

Announcements

- Final project: 45% of the grade, 10% presentation, 35% write-up
 - Presentations: in lecture Dec 1 and 3
 - If you have constraints, inform me by email by Wednesday night, we will assign the others at random on Thursday
- PS2: due Friday 23:59pm.
- Tuesday 4-5pm 540 Cory: Hadas Kress-Gazit (Cornell)

High-level tasks to correct low-level robot control

In this talk I will present a formal approach to creating robot controllers that ensure the robot satisfies a given high level task. I will describe a framework in which a user specifies a complex and reactive task in Structured English. This task is then automatically translated, using temporal logic and tools from the formal methods world, into a hybrid controller. This controller is guaranteed to control the robot such that its motion and actions satisfy the intended task, in a variety of different environments.

CS 287: Advanced Robotics Fall 2009

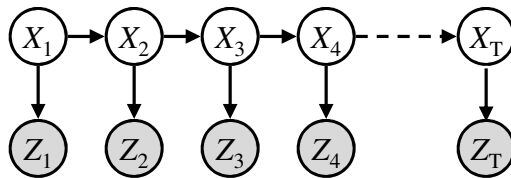
Lecture 23:
HMMs: Kalman filters, particle filters

Pieter Abbeel
UC Berkeley EECS

Hidden Markov Models

Joint distribution is assumed to be of the form:

$$P(X_1 = x_1) P(Z_1 = z_1 | X_1 = x_1) P(X_2 = x_2 | X_1 = x_1) P(Z_2 = z_2 | X_2 = x_2) \dots \\ P(X_T = x_T | X_{T-1} = x_{T-1}) P(Z_T = z_T | X_T = x_T)$$



Filtering in HMM

- Init $P(x_1)$ [e.g., uniformly]
- Observation update for time 0:

$$P(x_1|z_1) \propto P(z_1|x_1)P(x_1)$$

- For $t = 1, 2, \dots$

- Time update

$$P(x_{t+1}|z_{1:t}) = \int_{x_t} P(x_{t+1}|x_t)P(x_t|z_{1:t})dx_t$$

- Observation update

$$P(x_{t+1}|z_{1:t+1}) \propto P(z_{t+1}|x_{t+1})P(x_{t+1}|z_{1:t})$$

- For discrete state / observation spaces: simply replace integral by summation

Discrete-time Kalman Filter

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

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

with a measurement

$$z_t = C_t x_t + \delta_t$$

5

Kalman Filter Algorithm

Algorithm **Kalman_filter**(μ_{t-1} , Σ_{t-1} , u_t , Z_t):

Prediction:

$$\begin{aligned}\bar{\mu}_t &= A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t &= A_t \Sigma_{t-1} A_t^T + R_t\end{aligned}$$

Correction:

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

Return μ_t , Σ_t

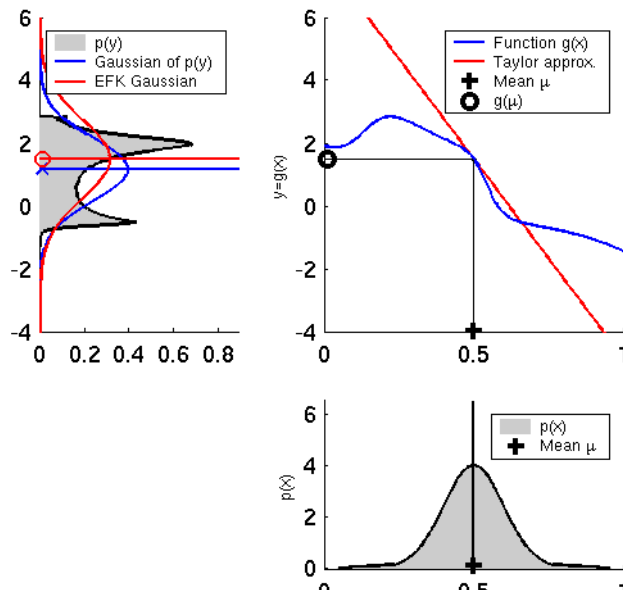
6

Nonlinear systems

$$x_t = g(u_t, x_{t-1}) + noise$$

$$z_t = h(x_t) + noise$$

Extended Kalman filter (EKF)



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

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

Prediction:

$$\begin{aligned} \bar{\mu}_t &= g(u_t, \mu_{t-1}) & \longleftarrow \bar{\mu}_t &= A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t &= G_t \Sigma_{t-1} G_t^T + R_t & \longleftarrow \bar{\Sigma}_t &= A_t \Sigma_{t-1} A_t^T + R_t \end{aligned}$$

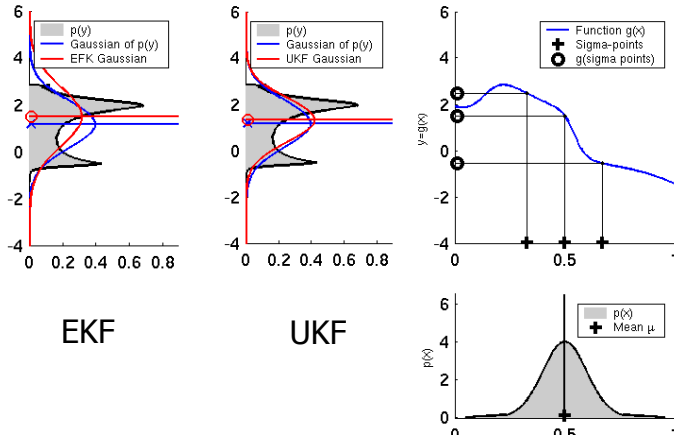
Correction:

$$\begin{aligned} K_t &= \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1} & \longleftarrow K_t &= \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \\ \mu_t &= \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t)) & \longleftarrow \mu_t &= \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \\ \Sigma_t &= (I - K_t H_t) \bar{\Sigma}_t & \longleftarrow \Sigma_t &= (I - K_t C_t) \bar{\Sigma}_t \end{aligned}$$

Return μ_t, Σ_t

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

Linearization via Unscented Transform

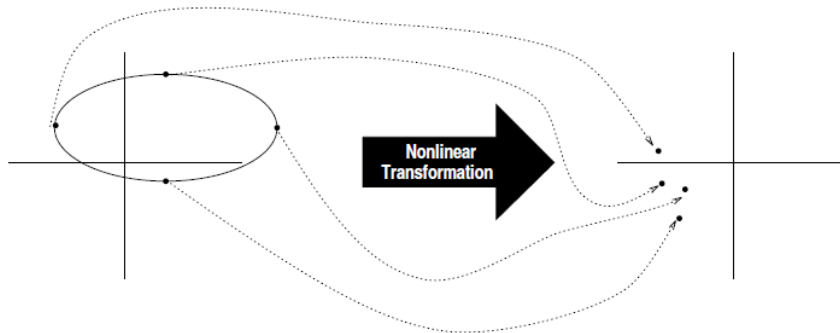


EKF

UKF

11

UKF Sigma-Point Estimate (4)



UKF intuition why it can perform better

- Assume we know the distribution over X and it has a mean \bar{x}
- $Y = f(X)$

$$\begin{aligned} \mathbf{f}[\mathbf{x}] &= \mathbf{f}[\bar{\mathbf{x}} + \delta\mathbf{x}] \\ &= \mathbf{f}[\bar{\mathbf{x}}] + \nabla\mathbf{f}\delta\mathbf{x} + \frac{1}{2}\nabla^2\mathbf{f}\delta\mathbf{x}^2 + \frac{1}{3!}\nabla^3\mathbf{f}\delta\mathbf{x}^3 + \frac{1}{4!}\nabla^4\mathbf{f}\delta\mathbf{x}^4 + \dots \end{aligned}$$

$$\bar{\mathbf{y}} = \mathbf{f}[\bar{\mathbf{x}}] + \frac{1}{2}\nabla^2\mathbf{f}\mathbf{P}_{xx} + \frac{1}{2}\nabla^4\mathbf{f}\mathbb{E}[\delta\mathbf{x}^4] + \dots$$

$$\begin{aligned} \mathbf{P}_{yy} &= \nabla\mathbf{f}\mathbf{P}_{xx}(\nabla\mathbf{f})^T + \frac{1}{2 \times 4!}\nabla^2\mathbf{f} \left(\mathbb{E}[\delta\mathbf{x}^4] - \mathbb{E}[\delta\mathbf{x}^2\mathbf{P}_{yy}] - \mathbb{E}[\mathbf{P}_{yy}\delta\mathbf{x}^2] + \mathbf{P}_{yy}^2 \right) (\nabla^2\mathbf{f})^T + \\ &\quad \frac{1}{3!}\nabla^3\mathbf{f}\mathbb{E}[\delta\mathbf{x}^4] (\nabla\mathbf{f})^T + \dots \end{aligned}$$

[Julier and Uhlmann, 1997]

UKF intuition why it can perform better

- Assume
 - 1. We represent our distribution over x by a set of sample points.
 - 2. We propagate the points directly through the function f .
- Then:
 - We don't have any errors in f !!
 - The accuracy we obtain can be related to how well the first, second, third, ... moments of the samples correspond to the first, second, third, ... moments of the true distribution over x .

Self-quiz

- When would the UKF significantly outperform the EKF?

Original unscented transform

- Picks a minimal set of sample points that match 1st, 2nd and 3rd moments of a Gaussian:

$$\begin{array}{ll}
 \mathbf{x}_0 & = \bar{\mathbf{x}} & W_0 & = \kappa / (n + \kappa) \\
 \mathbf{x}_i & = \bar{\mathbf{x}} + \left(\sqrt{(n + \kappa) \mathbf{P}_{xx}} \right)_i & W_i & = 1/2(n + \kappa) \\
 \mathbf{x}_{i+n} & = \bar{\mathbf{x}} - \left(\sqrt{(n + \kappa) \mathbf{P}_{xx}} \right)_i & W_{i+n} & = 1/2(n + \kappa)
 \end{array}$$

- $\bar{\mathbf{x}}$ = mean, \mathbf{P}_{xx} = covariance, $i \rightarrow i$ 'th row, $\mathbf{x} \in \mathbb{R}^n$
- κ : extra degree of freedom to fine-tune the higher order moments of the approximation; when \mathbf{x} is Gaussian, $n + \kappa = 3$ is a suggested heuristic

[Julier and Uhlmann, 1997]

Unscented Kalman filter

- Dynamics update:
 - Can simply use unscented transform and estimate the mean and variance at the next time from the sample points
- Observation update:
 - Use sigmapoints from unscented transform to compute the covariance matrix between x_t and z_t . Then can do the standard update.

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

1. $\mathcal{X}_{t-1} = (\mu_{t-1} \quad \mu_{t-1} + \gamma\sqrt{\Sigma_{t-1}} \quad \mu_{t-1} - \gamma\sqrt{\Sigma_{t-1}})$
2. $\bar{\mathcal{X}}_t^* = g(\mu_t, \mathcal{X}_{t-1})$
3. $\bar{\mu}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{X}}_t^{*[i]}$
4. $\bar{\Sigma}_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)(\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)^\top + R_t$
5. $\bar{\mathcal{X}}_t = (\bar{\mu}_t \quad \bar{\mu}_t + \gamma\sqrt{\bar{\Sigma}_t} \quad \bar{\mu}_t - \gamma\sqrt{\bar{\Sigma}_t})$
6. $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$
7. $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$
8. $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^\top + Q_t$
9. $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^\top$
10. $K_t = \bar{\Sigma}_t^{x,z} S_t^{-1}$
11. $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$
12. $\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^\top$
13. **return** μ_t, Σ_t

[Table 3.4 in Probabilistic Robotics]

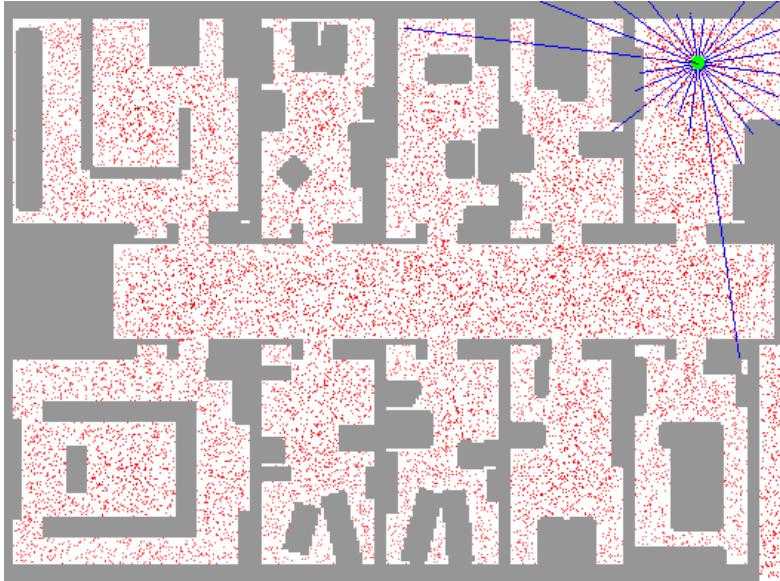
UKF Summary

- **Highly efficient**: Same complexity as EKF, with a constant factor slower in typical practical applications
- **Better linearization than EKF**: Accurate in first two terms of Taylor expansion (EKF only first term) + capturing more aspects of the higher order terms
- **Derivative-free**: No Jacobians needed
- **Still not optimal!**

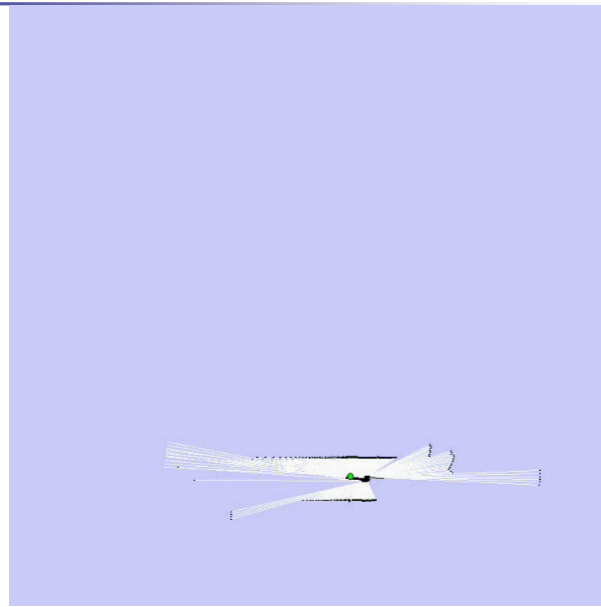
Particle filters motivation

- Particle filters are a way to **efficiently** represent **non-Gaussian distribution**
- Basic principle
 - Set of state hypotheses ("particles")
 - Survival-of-the-fittest

Sample-based Localization (sonar)

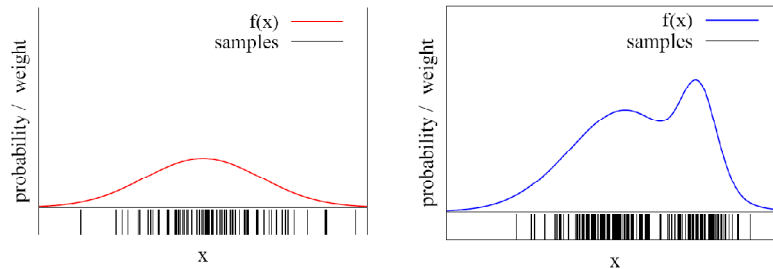


FastSLAM [particle filter + Rao-Blackwellization +
occupancy grid mapping + scan matching based odometry]



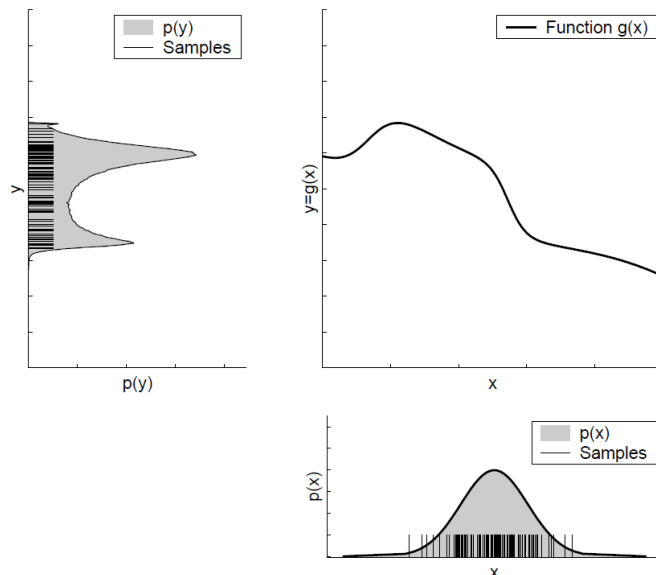
Sample-based Mathematical Description of Probability Distributions

- Particle sets can be used to approximate functions



- The more particles fall into an interval, the higher the probability of that interval
- If a continuous density needs to be extracted \rightarrow can place a narrow Gaussian at the location of each particle
- How to efficiently draw samples in an HMM?

Dynamics update with sample representation of distribution: sample from $P(x_{t+1} | x_t)$



Observation update

- Goal: go from a sample-based representation of

$$P(x_{t+1} | z_1, \dots, z_t)$$

to a sample-based representation of

$$P(x_{t+1} | z_1, \dots, z_t, z_{t+1}) =$$

$$C * P(x_{t+1} | z_1, \dots, z_t) * P(z_{t+1} | x_{t+1})$$

Importance sampling

- Interested in estimating:

$$\begin{aligned} E_{X \sim P}[f(X)] &= \int_x f(x)P(x)dx \\ &= \int_x f(x)P(x)\frac{Q(x)}{Q(x)}dx \text{ if } Q(x) = 0 \Rightarrow P(x) = 0 \\ &= \int_x f(x)\frac{P(x)}{Q(x)}Q(x)dx \\ &= E_{X \sim Q}\left[\frac{P(X)}{Q(X)}f(X)\right] \\ &\approx \frac{1}{m} \sum_{i=1}^m \frac{P(x^{(i)})}{Q(x^{(i)})} f(x^{(i)}) \text{ with } x^{(i)} \sim Q \end{aligned}$$

- **Hence we could sample from an alternative distribution Q and simply re-weight the samples == Importance Sampling**

Sequential Importance Sampling (SIS) Particle Filter

- Sample $x^{(1)}_1, x^{(2)}_1, \dots, x^{(N)}_1$ from $P(X_1)$
- Set $w^{(i)}_1 = 1$ for all $i=1, \dots, N$
- For $t=1, 2, \dots$
 - Dynamics update:
 - For $i=1, 2, \dots, N$
 - Sample $x^{(i)}_t$ from $P(X_{t+1} | X_t = x^{(i)}_t)$
 - Observation update:
 - For $i=1, 2, \dots, N$
 - $w^{(i)}_{t+1} = w^{(i)}_t * P(z_{t+1} | X_{t+1} = x^{(i)}_{t+1})$
- At any time t , the distribution is represented by the weighted set of samples $\{ \langle x^{(i)}_t, w^{(i)}_t \rangle ; i=1, \dots, N \}$

SIS particle filter major issue

- The resulting samples are only weighted by the evidence
 - The samples themselves are never affected by the evidence
- Fails to concentrate particles/computation in the high probability areas of the distribution $P(x_t | z_1, \dots, z_t)$

Resampling

- At any time t , the distribution is represented by the weighted set of samples

$$\{ \langle x_t^{(i)}, w_t^{(i)} \rangle ; i=1, \dots, N \}$$

- Sample N times from the set of particles
 - The probability of drawing each particle is given by its importance weight
- More particles/computation focused on the parts of the state space with high probability mass

1. Algorithm **particle_filter**(S_{t-1}, u_{t-1}, z_t):

2. $S_t = \emptyset, \eta = 0$

3. **For** $i=1 \dots n$ *Generate new samples*

4. Sample index $j(i)$ from the discrete distribution given by w_{t-1}

5. Sample x_t^i from $p(x_t | x_{t-1}, u_{t-1})$ using $x_{t-1}^{j(i)}$ and u_{t-1}

6. $w_t^i = p(z_t | x_t^i)$ *Compute importance weight*

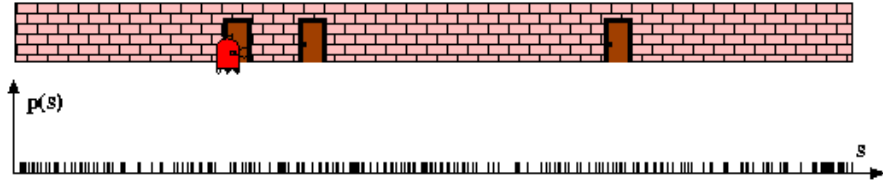
7. $\eta = \eta + w_t^i$ *Update normalization factor*

8. $S_t = S_t \cup \{ \langle x_t^i, w_t^i \rangle \}$ *Insert*

9. **For** $i=1 \dots n$

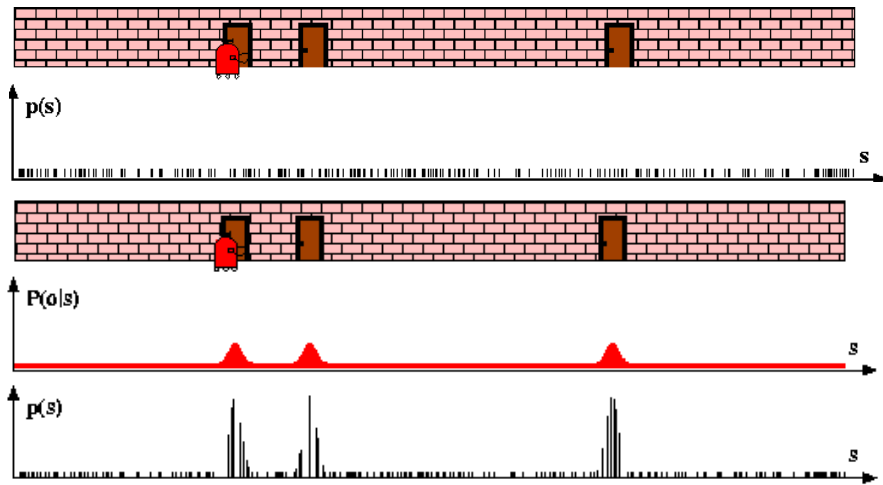
10. $w_t^i = w_t^i / \eta$ *Normalize weights*

Particle Filters



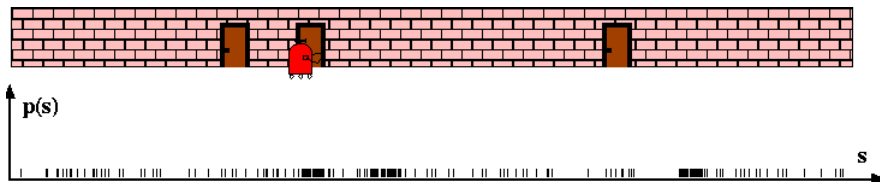
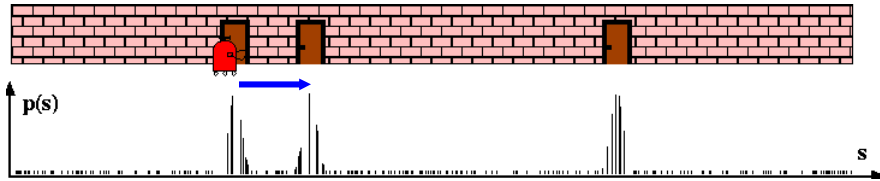
Sensor Information: Importance Sampling

$$\begin{aligned}
 Bel(x) &\leftarrow \alpha p(z|x) Bel^-(x) \\
 w &\leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x)
 \end{aligned}$$



Robot Motion

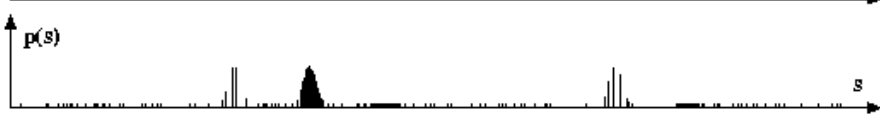
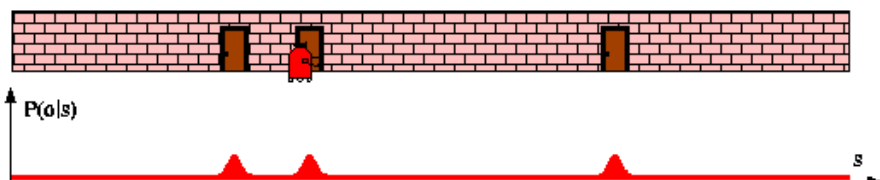
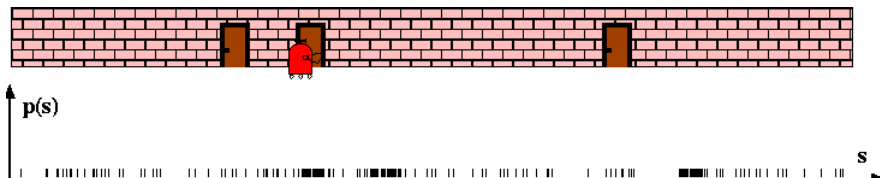
$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$



Sensor Information: Importance Sampling

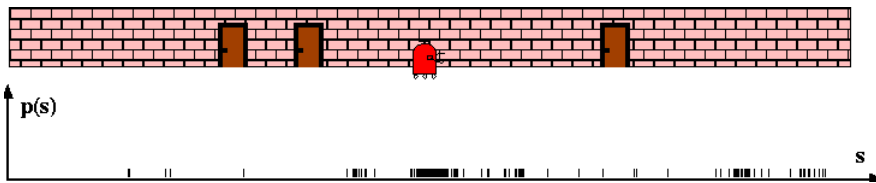
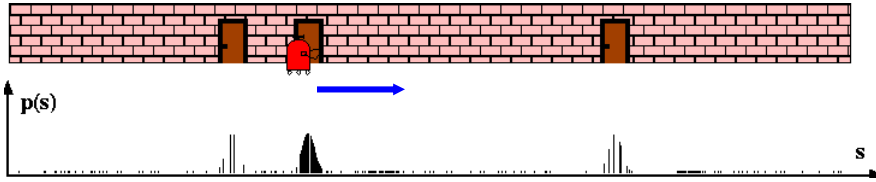
$$Bel(x) \leftarrow \alpha p(z|x) Bel^-(x)$$

$$w \leftarrow \frac{\alpha p(z|x) Bel^-(x)}{Bel^-(x)} = \alpha p(z|x)$$



Robot Motion

$$Bel^-(x) \leftarrow \int p(x|u, x') Bel(x') dx'$$



Resampling issue

- Loss of samples ...

Low Variance Resampling

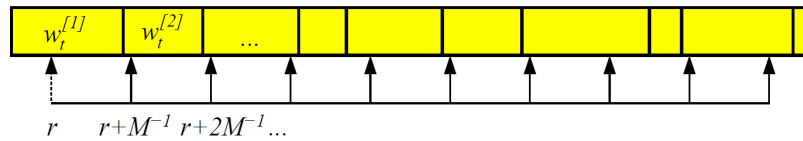


Figure 4.7 Principle of the low variance resampling procedure. We choose a random number r and then select those particles that correspond to $u = r + (m - 1) \cdot M^{-1}$ where $m = 1, \dots, M$.

Low Variance Resampling

- Advantages:
 - More systematic coverage of space of samples
 - If all samples have same importance weight, no samples are lost
 - Lower computational complexity

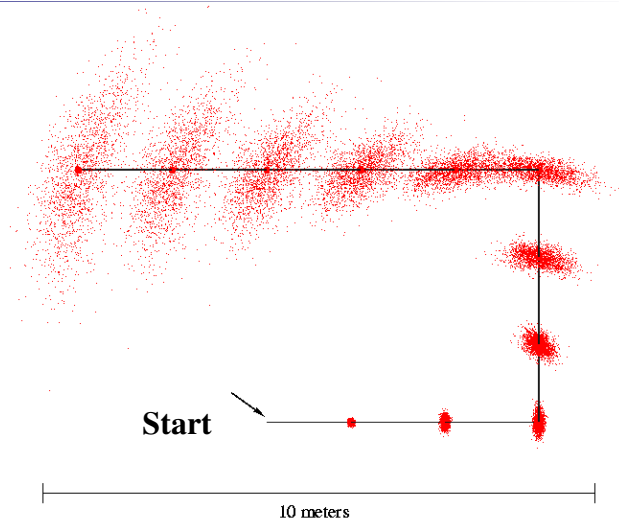
Regularization

- If no dynamics noise \rightarrow all particles will start to coincide
- \rightarrow regularization: resample from a (narrow) Gaussian around the particle

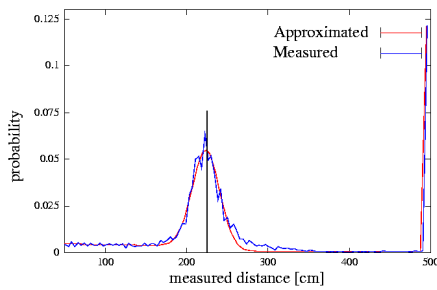
Mobile Robot Localization

- Each particle is a potential pose of the robot
- Proposal distribution is the motion model of the robot (prediction step)
- The observation model is used to compute the importance weight (correction step)

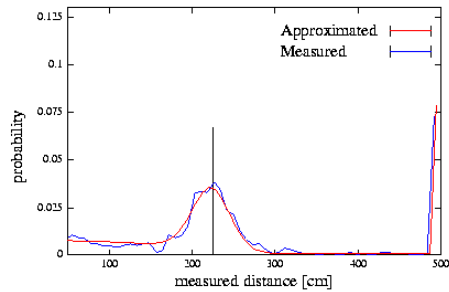
Motion Model Reminder



Proximity Sensor Model Reminder



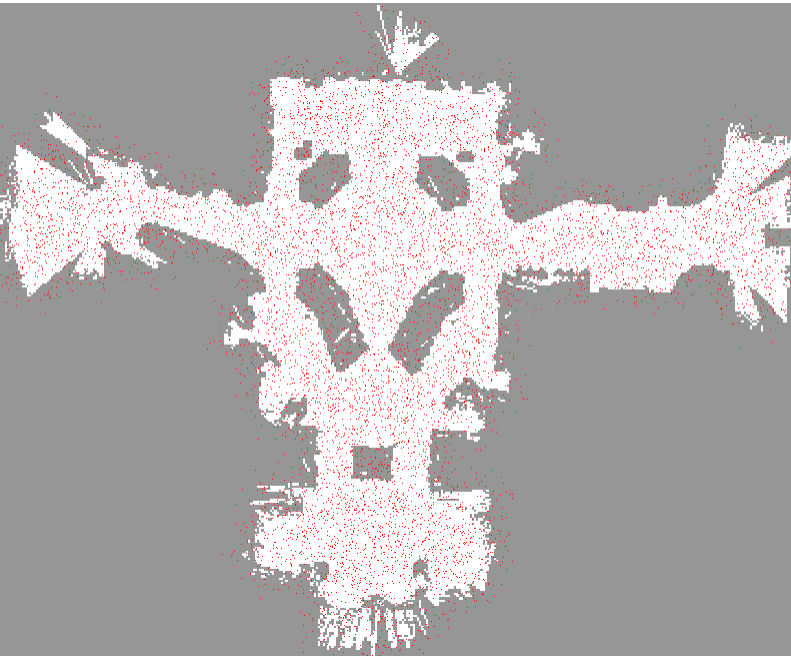
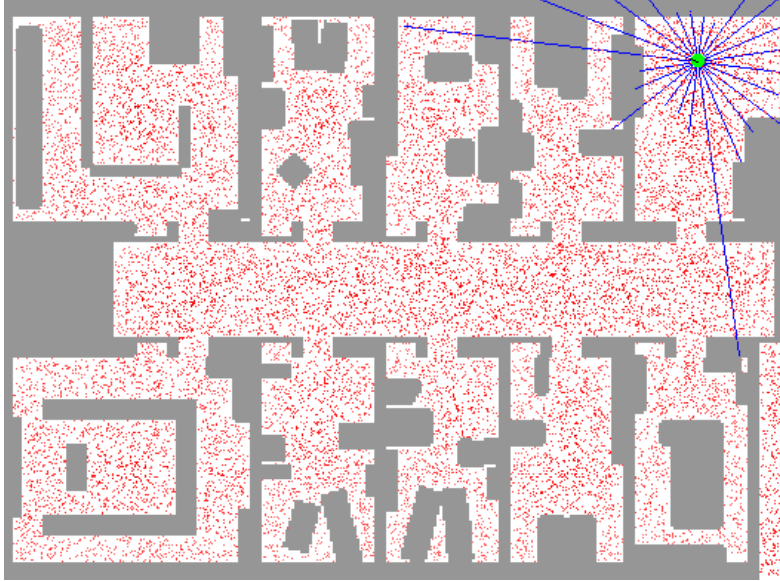
Laser sensor



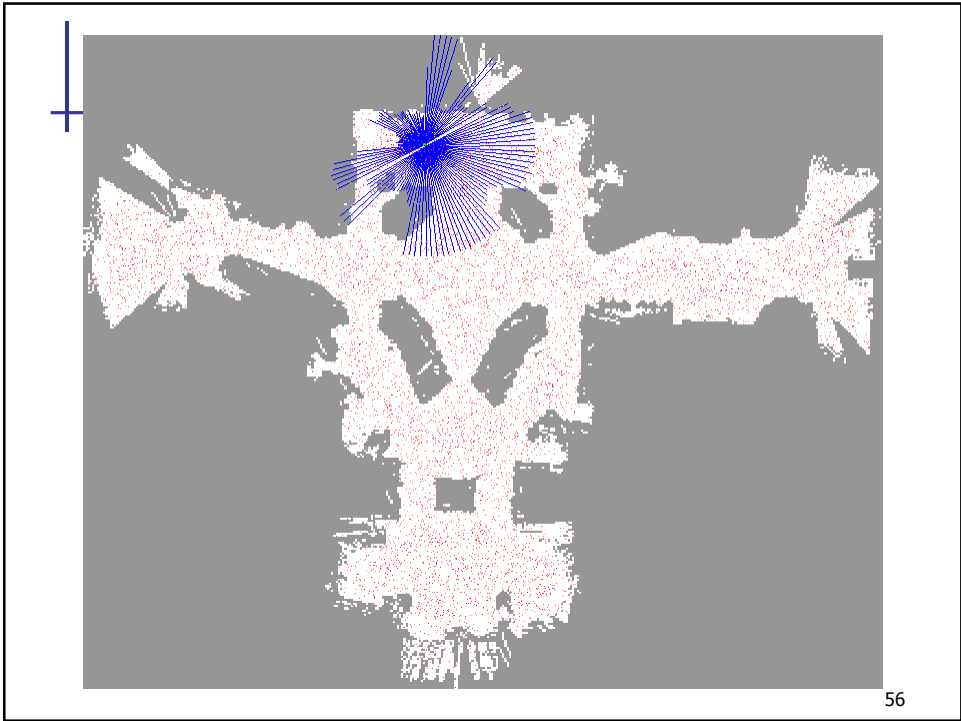
Sonar sensor

Note: sensor model is not Gaussian at all!

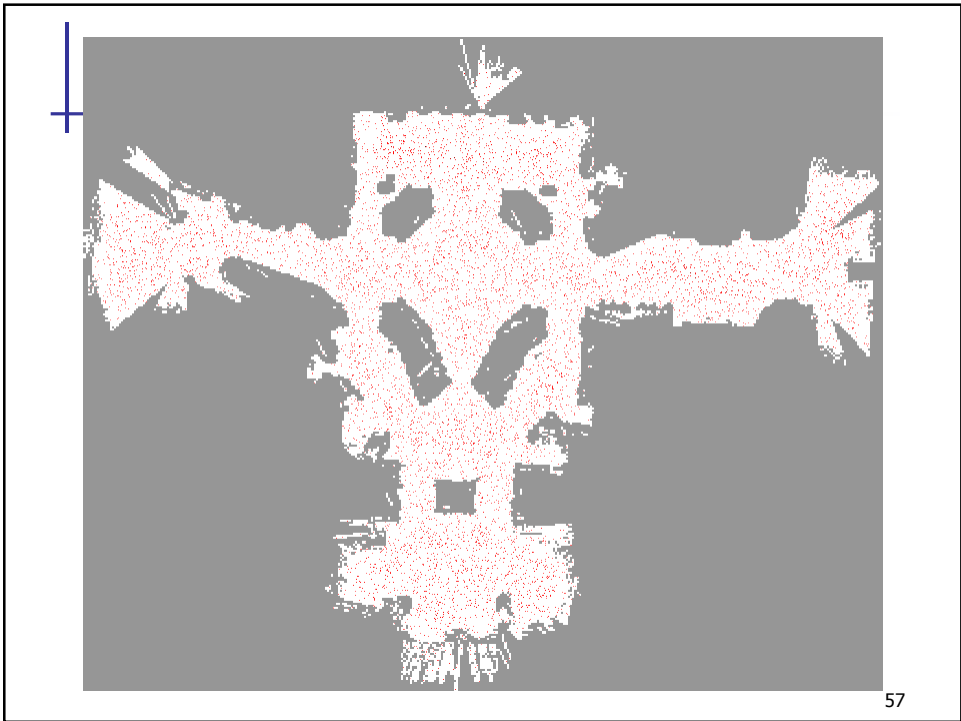
Sample-based Localization (sonar)



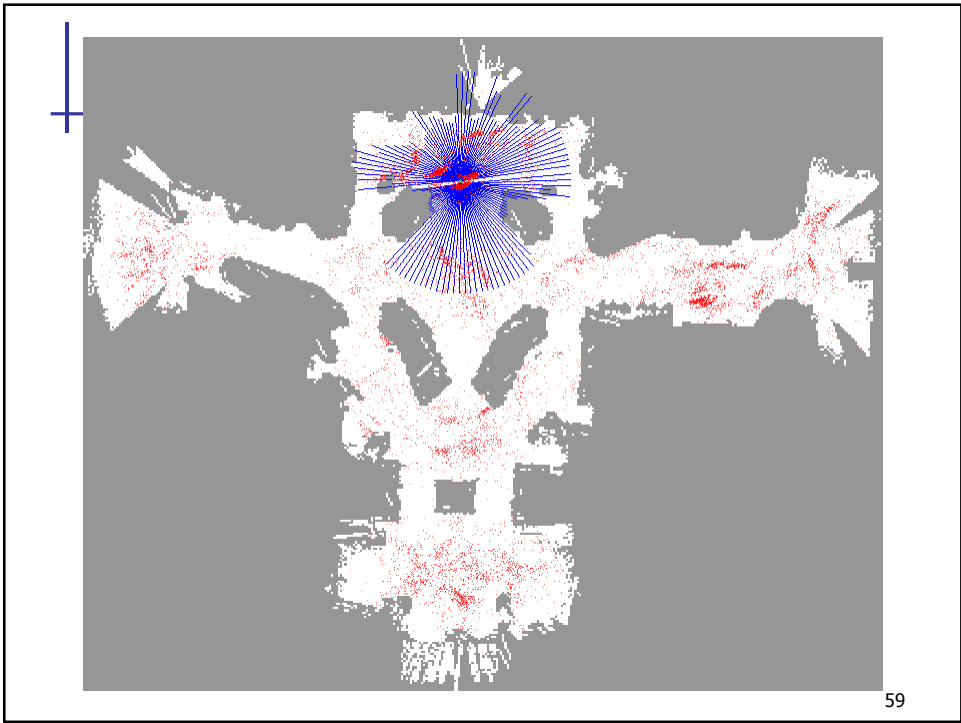
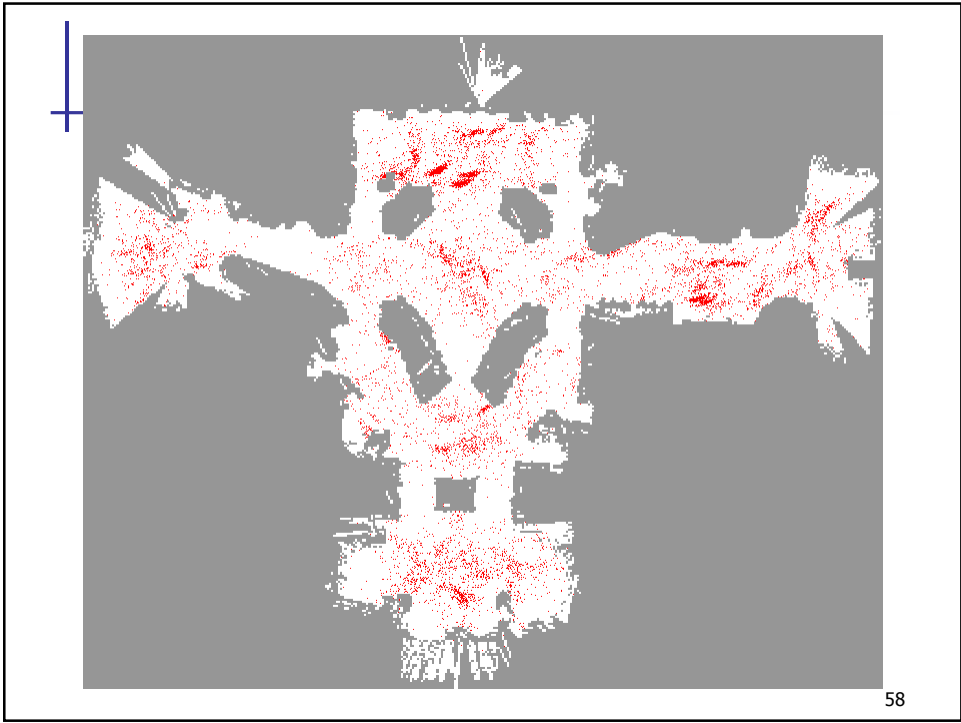
55

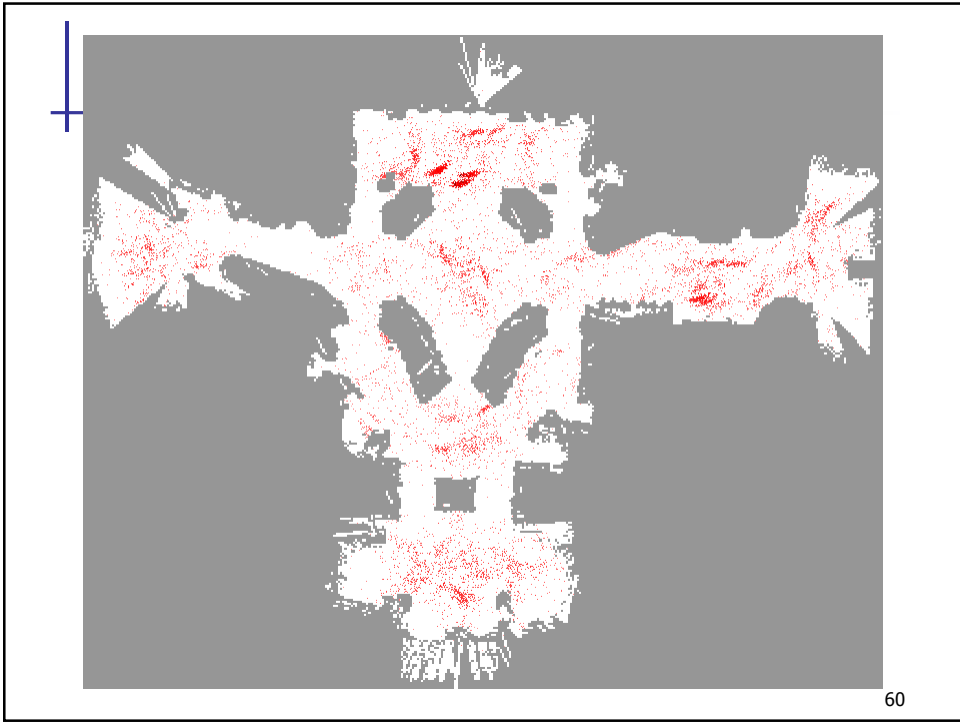


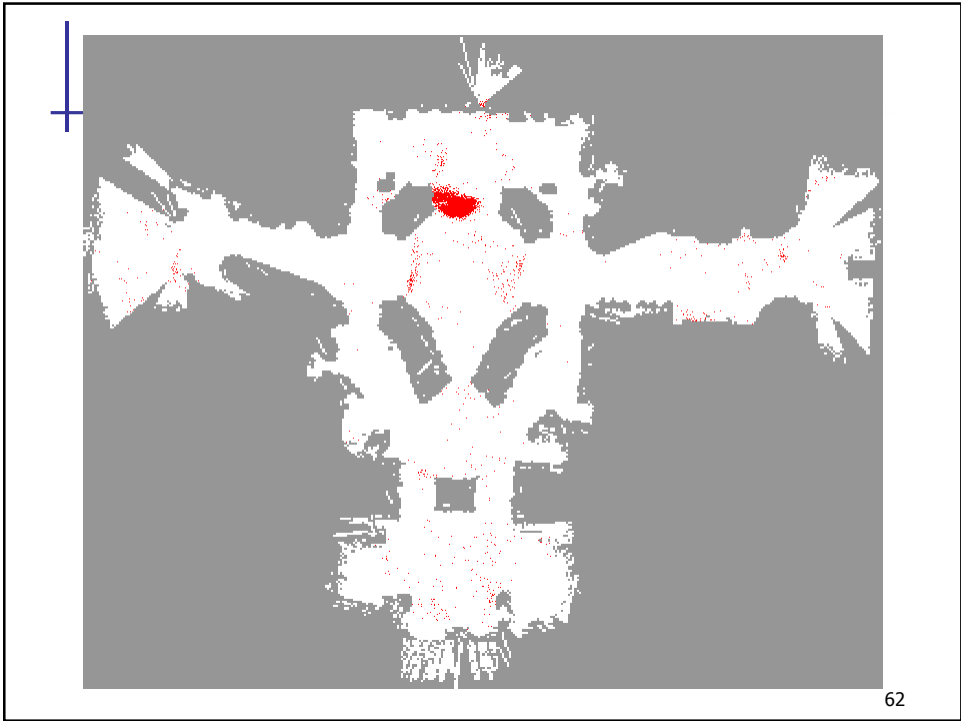
56



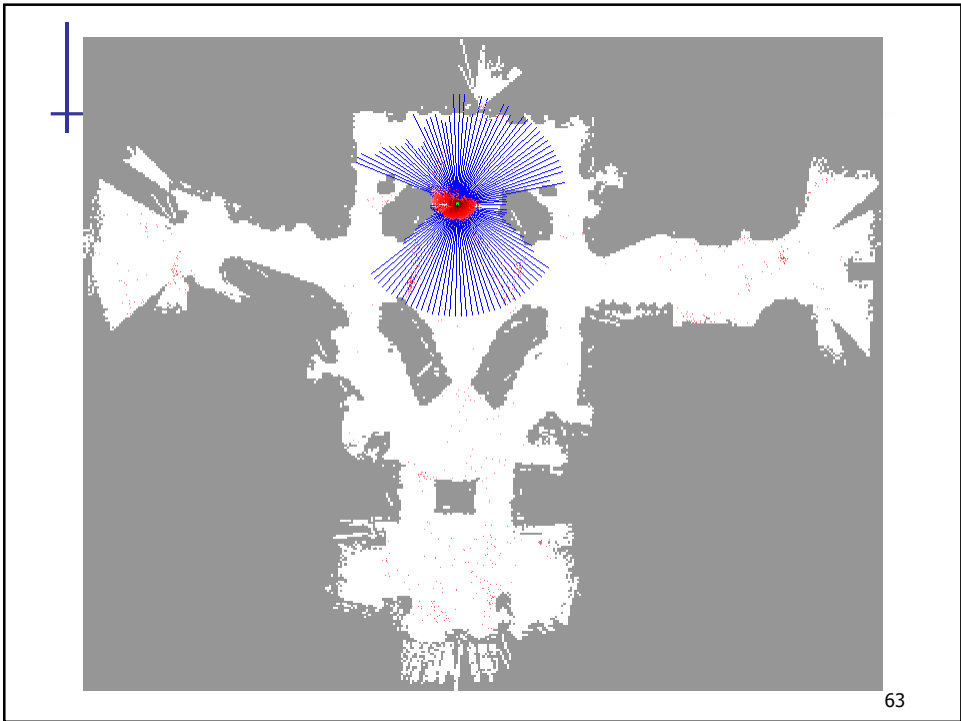
57



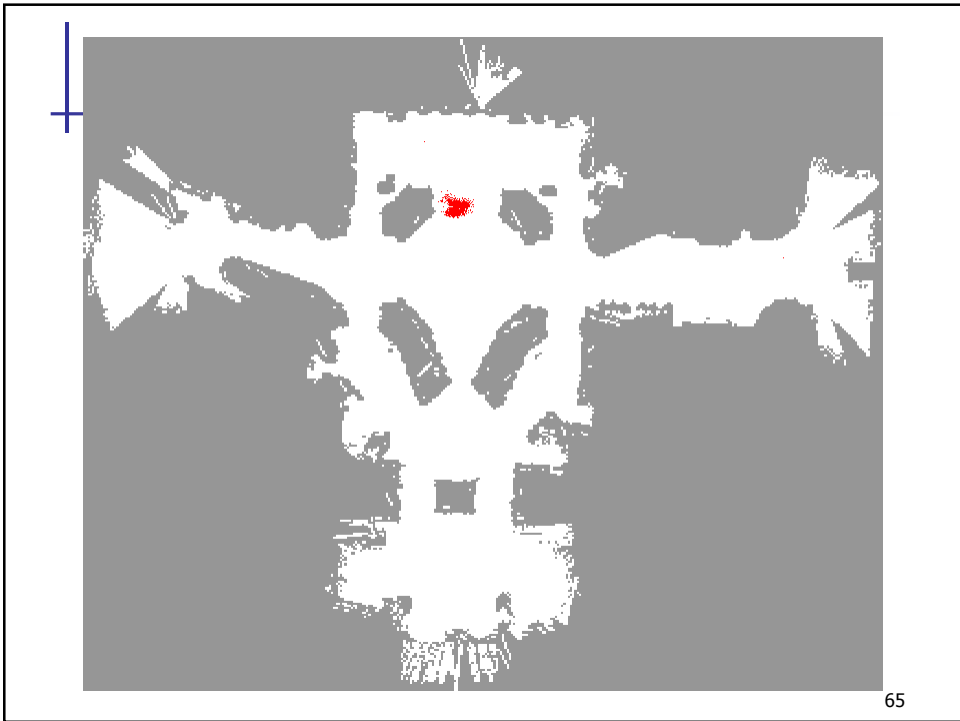
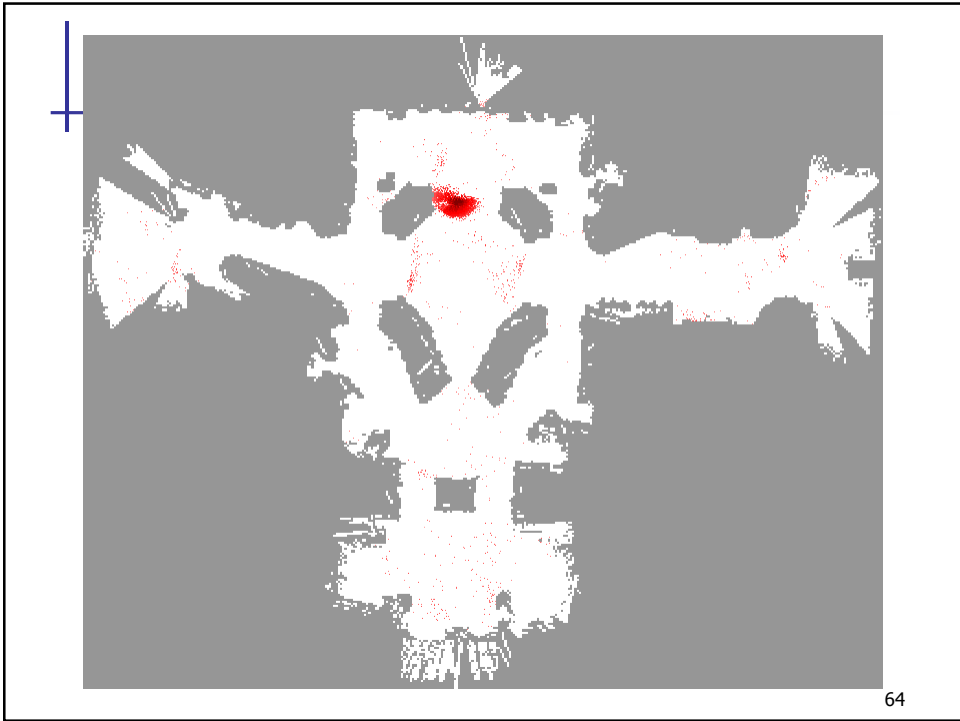


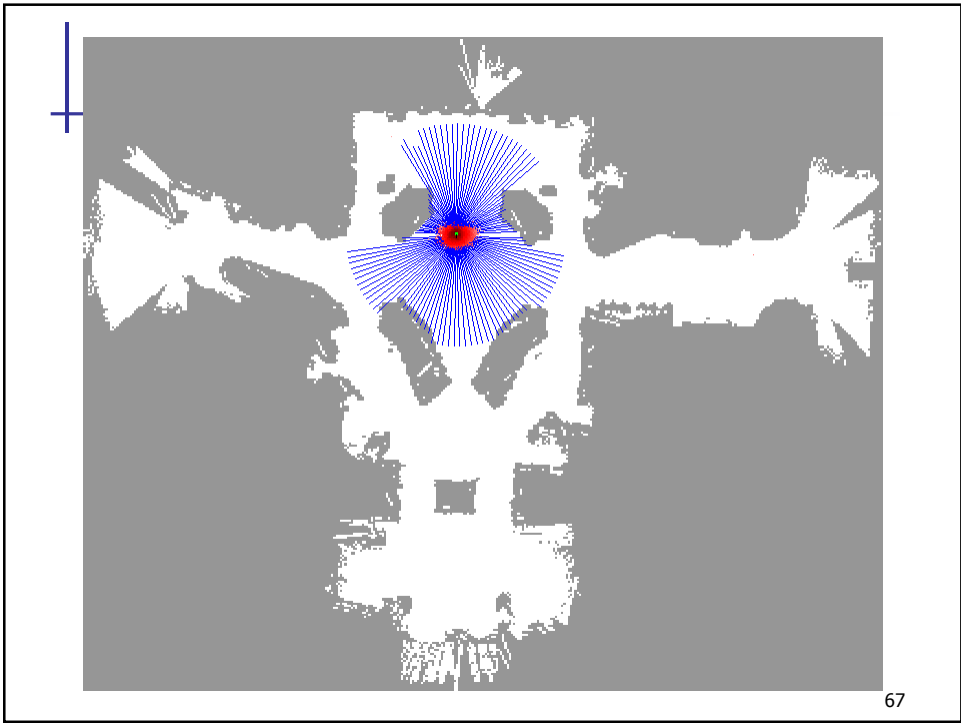
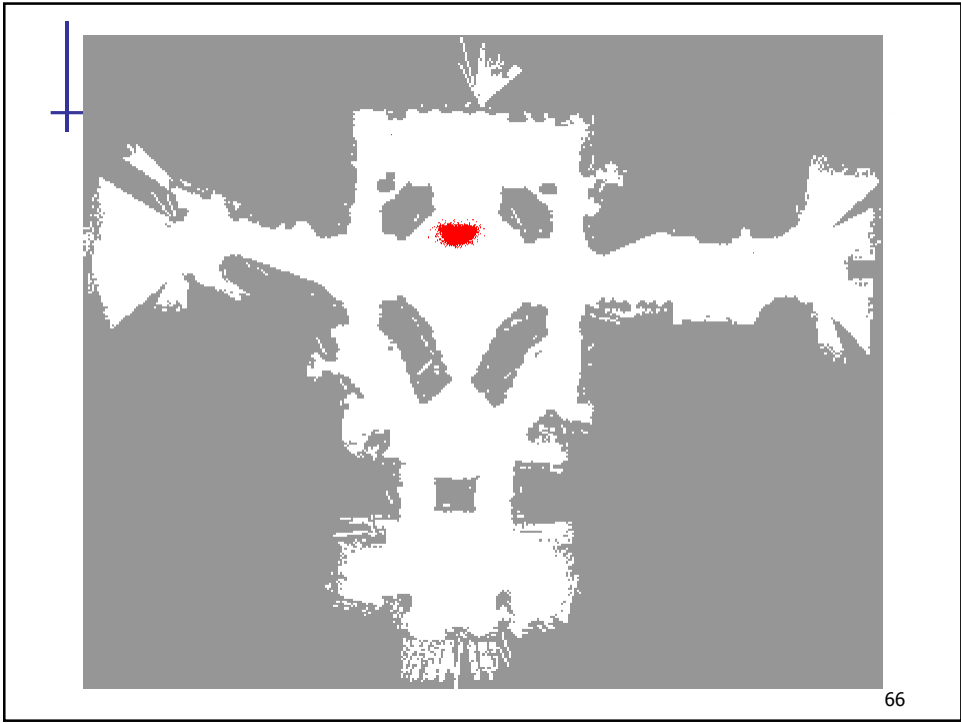


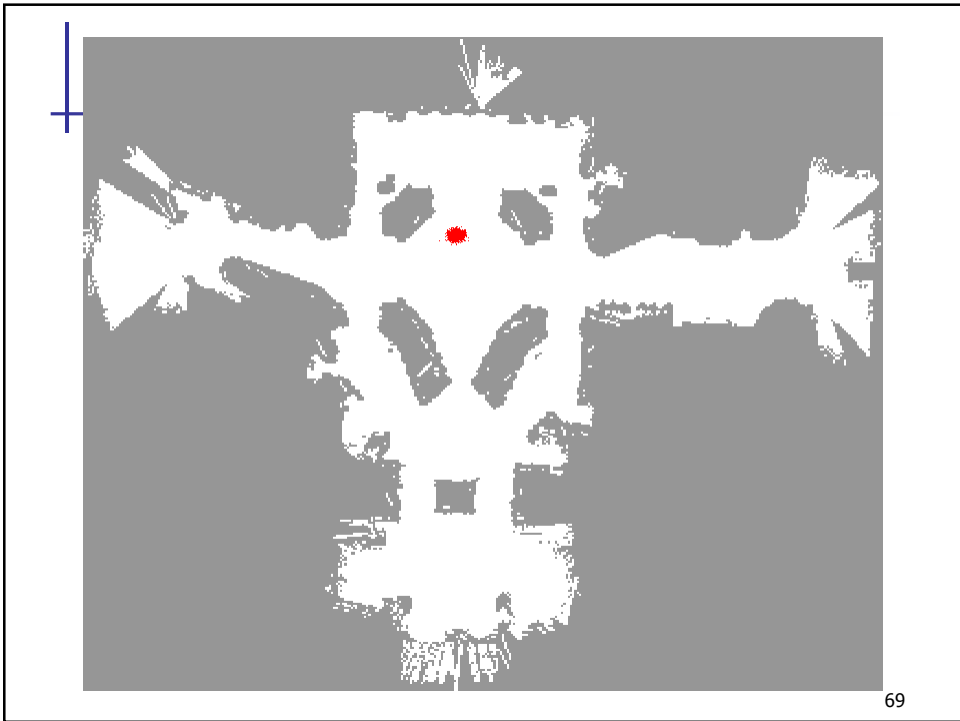
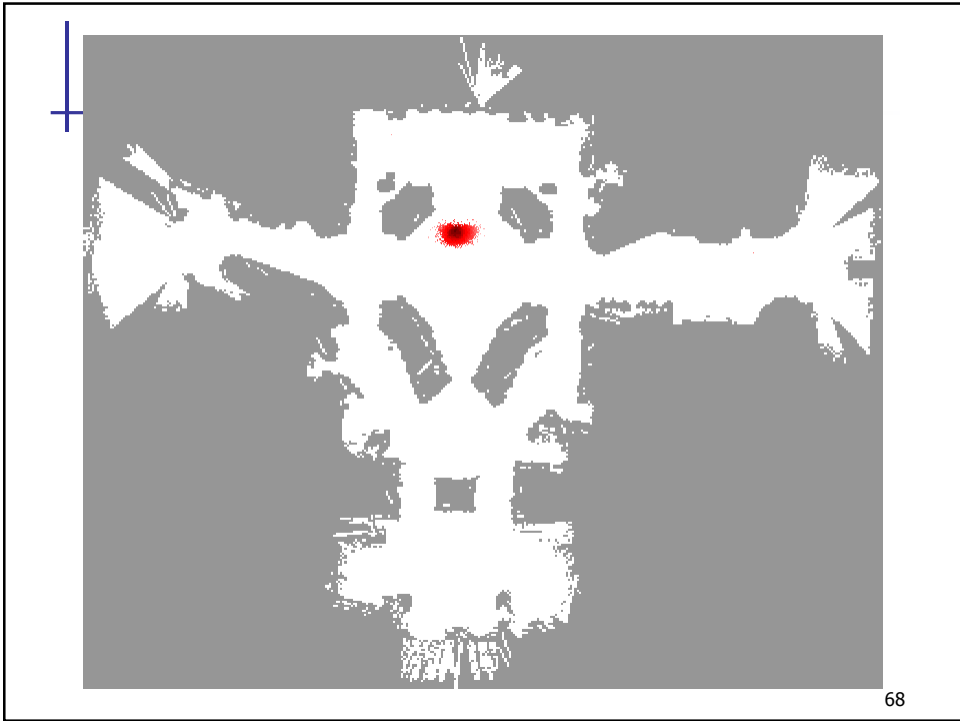
62

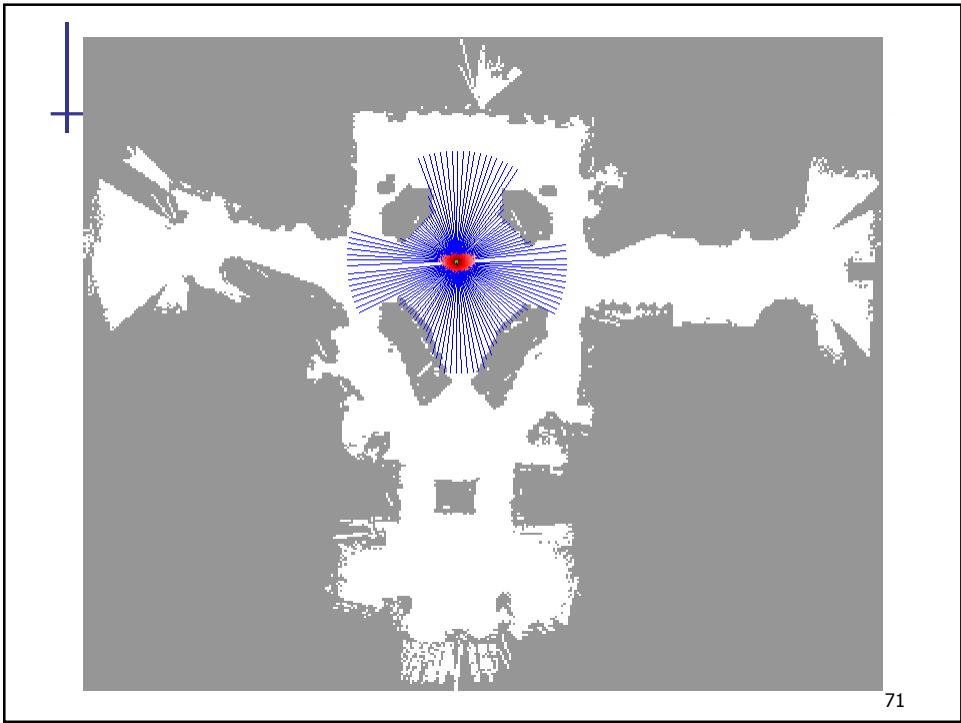
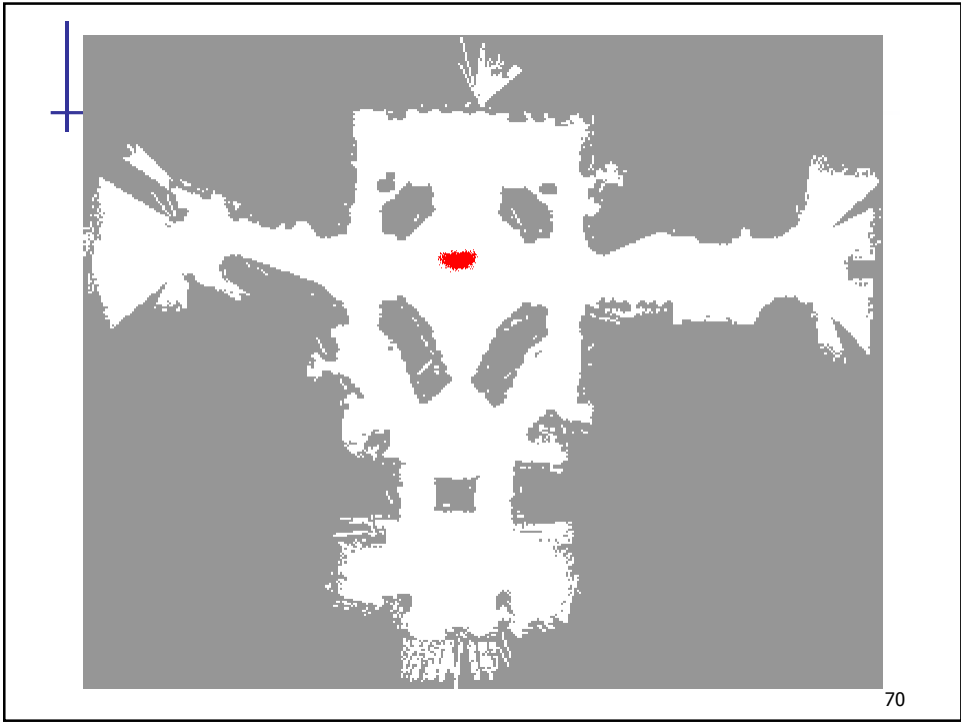


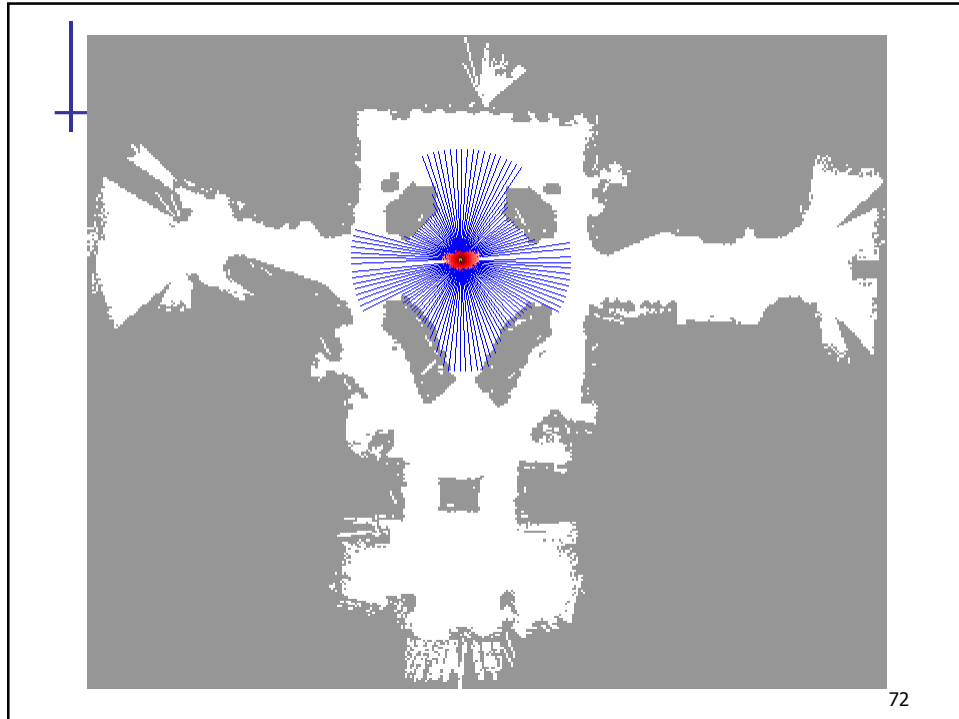
63











Summary – Particle Filters

- Particle filters are an implementation of recursive Bayesian filtering
- They represent the posterior by a set of weighted samples
- They can model non-Gaussian distributions
- Proposal to draw new samples
- Weight to account for the differences between the proposal and the target
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter

77

Summary – PF Localization

- In the context of localization, the particles are propagated according to the motion model.
- They are then weighted according to the likelihood of the observations.
- In a re-sampling step, new particles are drawn with a probability proportional to the likelihood of the observation.