Probabilistic Robotics

Mobile Robot Localization

Wolfram Burgard
Cyrill Stachniss

Probabilistic Robotics

Key idea: Explicit representation of uncertainty

(using the calculus of probability theory)

- Perception = state estimation
- Action = utility optimization

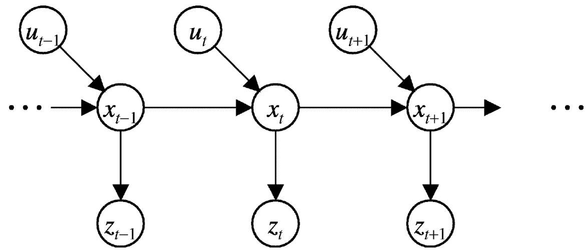
Bayes Filters: Framework

- **Given:**
  - Stream of observations *z* and action data *u:*

    $d_t = \{u_1, z_1$ ☺ $, u_t, z_t\}$
  - Sensor model *P(z|x).*
  - Action model *P(x|u,x').*
  - Prior probability of the system state *P(x).*
- **Wanted:**
  - Estimate of the state *X* of a dynamical system.
  - The posterior of the state is also called **Belief**:

$$Bel(x_t) = P(x_t \mid u_1, z_1 ☺, u_t, z_t)$$

Markov Assumption



$$p(z_t \mid x_{0:t}, z_{1:t}, u_{1:t}) = p(z_t \mid x_t)$$
$$p(x_t \mid x_{1:t-1}, z_{1:t}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

Underlying Assumptions
- Static world
- Independent noise
- Perfect model, no approximation errors

Bayes Filters

z = observation
u = action
x = state

$$Bel(x_t) = P(x_t \mid u_1, z_1 \text{☺}, u_t, z_t)$$

Bayes $$= \eta\ P(z_t \mid x_t, u_1, z_1, \text{☺}, u_t)\ P(x_t \mid u_1, z_1, \text{☺}, u_t)$$

Markov $$= \eta\ P(z_t \mid x_t)\ P(x_t \mid u_1, z_1, \text{☺}, u_t)$$

Total prob. $$= \eta\ P(z_t \mid x_t) \int P(x_t \mid u_1, z_1, \text{☺}, u_t, x_{t-1})\ P(x_{t-1} \mid u_1, z_1, \text{☺}, u_t)\ dx_{t-1}$$

Markov $$= \eta\ P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1})\ P(x_{t-1} \mid u_1, z_1, \text{☺}, u_t)\ dx_{t-1}$$

Markov $$= \eta P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1})\, P(x_{t-1} \mid u_1, z_1, \text{☺}, z_{t-1})\ dx_{t-1}$$

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

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

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 Filters are Frequently used Robotics

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

- Kalman filters
- Particle filters
- Hidden Markov models
- Dynamic Bayesian networks
- Partially Observable Markov Decision Processes (POMDPs)

Example: Robot Localization using a Bayes Filter

- Action: motion information of the robot
- Perception: compare the robots sensor observations to the model of the world
- Particle filters are a way to **efficiently** represent **non-Gaussian distribution**
- Basic principle
  - Set of state hypotheses (“particles”)
  - Kind of “survival-of-the-fittest”

Mathematical Description

- Set of weighted samples

$$S = \left\{ \left\langle s^{[i]}, w^{[i]} \right\rangle \mid i = 1, \ldots, N \right\}$$

State hypothesis — Importance weight

- The samples represent the posterior

$$p(x) = \sum_{i=1}^{N} w_i \cdot \delta_{s^{[i]}}(x)$$

Particle Filter Algorithm

- Action step: sample the next generation for particles using a probabilistic motion model (proposal distribution)

$$Bel^-(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\; \mathrm{d}x'$$

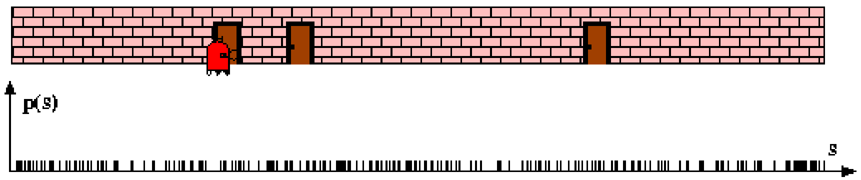
- Perception step: compute the importance weights to incorporate the observation:

$$Bel(x) \leftarrow \alpha\, p(z \mid x)\, Bel^-(x) = \alpha\, p(z \mid x) Bel^-(x)$$

- Resampling: Draw particle with a probability proportional to their importance weight

  "Replace unlikely samples by more likely ones"

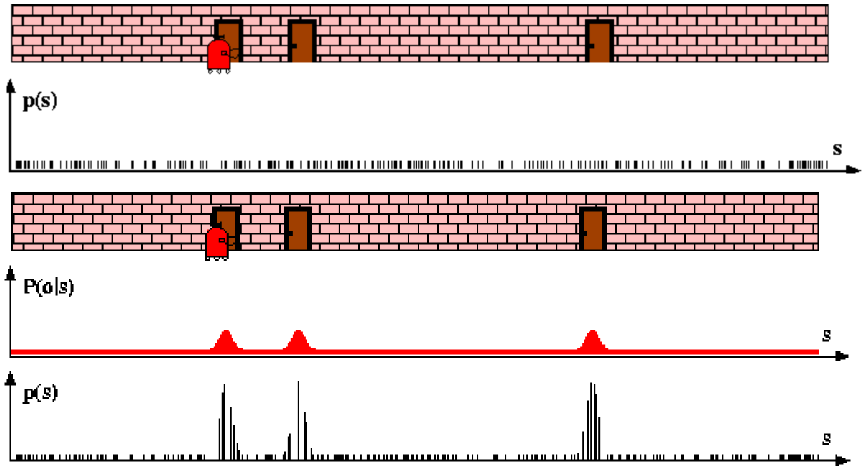
Particle Filters



Sensor Information: Importance Sampling

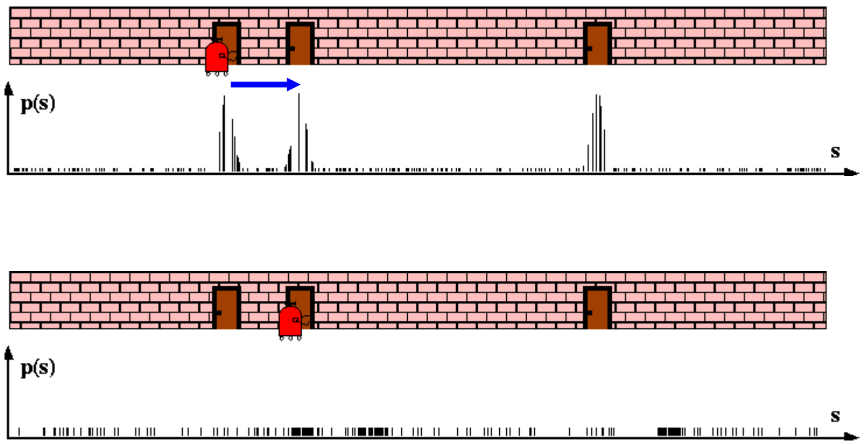
$$Bel(x) \leftarrow \alpha\, p(z \mid x)\, Bel^-(x)$$

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



Robot Motion

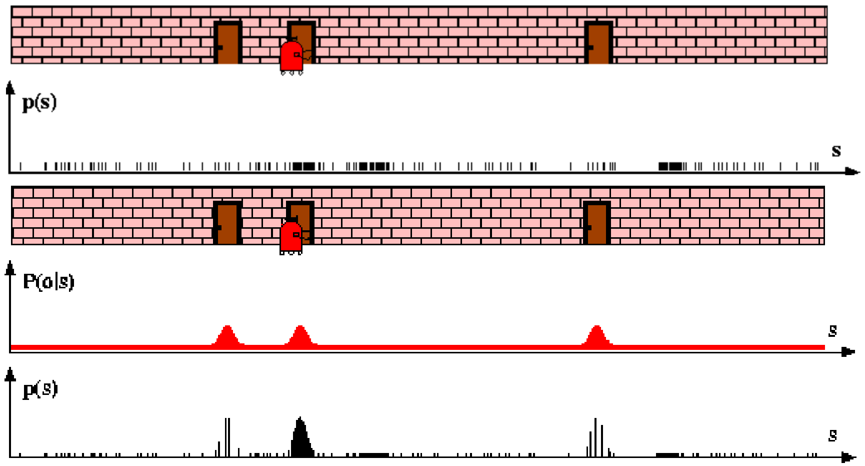
$$Bel^-(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'$$



Sensor Information: Importance Sampling

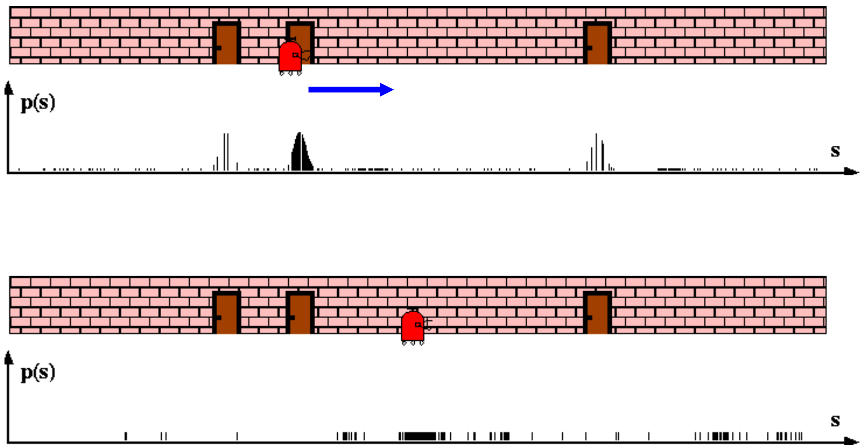
$$Bel(x) \leftarrow \alpha\, p(z \mid x)\, Bel^-(x)$$

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



Robot Motion

$$Bel^-(x) \leftarrow \int p(x \mid u, x')\, Bel(x')\, \mathrm{d}x'$$



Particle Filter Algorithm

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

- draw x^i_{t-1} from $Bel(x_{t-1})$
- draw x^i_t from $p(x_t \mid x^i_{t-1}, u_{t-1})$
- Importance factor for x^i_t:

$$w^i_t = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{\eta\; p(z_t \mid x_t)\; p(x_t \mid x_{t-1}, u_{t-1})\; Bel\,(x_{t-1})}{p(x_t \mid x_{t-1}, u_{t-1})\; Bel\,(x_{t-1})} \propto p(z_t \mid x_t)$$

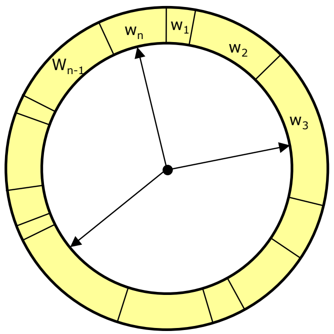
Particle Filter Algorithm

1. Algorithm **particle_filter**(S_{t-1}, u_{t-1} z_t):
2. $S_t = \emptyset, \quad \eta = 0$
3. **For** $i = 1$ ☹ n — ***Generate new samples***
4. Sample index $j(i)$ from the discrete distribution given by w_{t-1}
5. Sample x_t^i from $p(x_t \mid x_{t-1}, u_{t-1})$ using $x_{t-1}^{j(i)}$ and u_{t-1}
6. $w_t^i = p(z_t \mid x_t^i)$ — ***Compute importance weight***
7. $\eta = \eta + w_t^i$ — ***Update normalization factor***
8. $S_t = S_t \cup \{< x_t^i, w_t^i >\}$ — ***Insert***
9. **For** $i = 1$ ☹ n
10. $w_t^i = w_t^i / \eta$ — ***Normalize weights***

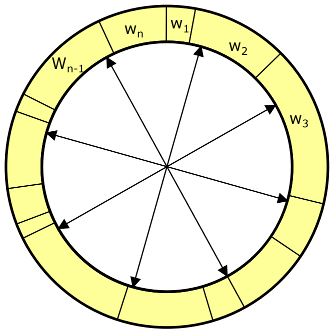
Resampling

- **Given**: Set S of weighted samples.
- **Wanted** : Random sample, where the probability of drawing x_i is given by w_i.
- Typically done n times with replacement to generate new sample set S'.

Resampling



- Roulette wheel
- Binary search, n log n



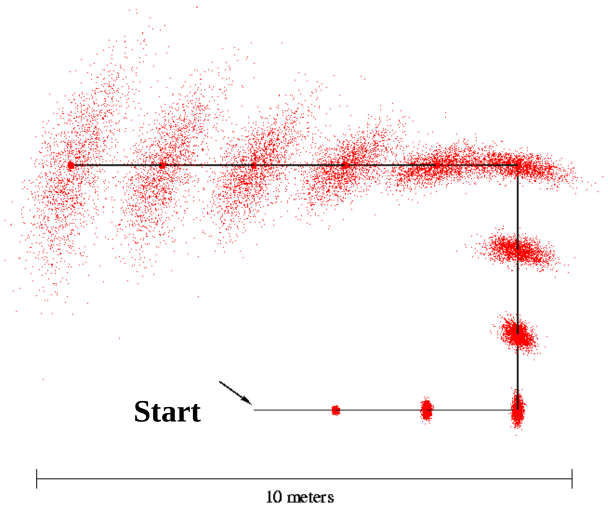
- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance

Resampling Algorithm

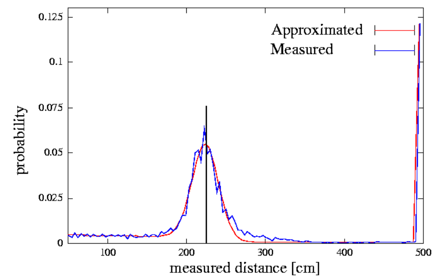
1. Algorithm **systematic_resampling**(S,n):
2. $S' = \emptyset, c_1 = w^1$
3. **For** $i = 2$ ☹ n — ***Generate cdf***
4. $c_i = c_{i-1} + w^i$
5. $u_1 \sim U]0, n^{-1}], i = 1$ — ***Initialize threshold***
6. **For** $j = 1$ ☹ n — ***Draw samples …***
7. **While** ($u_j > c_i$) — ***Skip until next threshold reached***
8. $i = i + 1$
9. $S' = S' \cup \{< x^i, n^{-1} >\}$ — ***Insert***
10. $u_{j+1} = u_j + n^{-1}$ — ***Increment threshold***
11. **Return** S'

Also called **stochastic universal sampling**

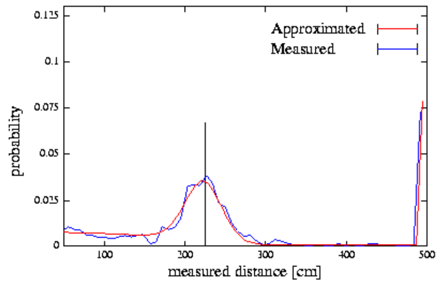
Motion Model Reminder



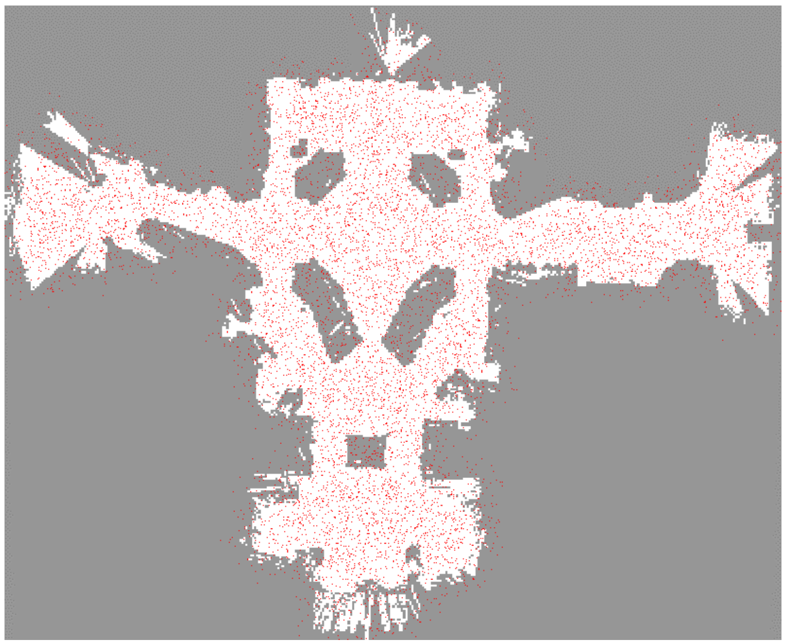
Proximity Sensor Model Reminder

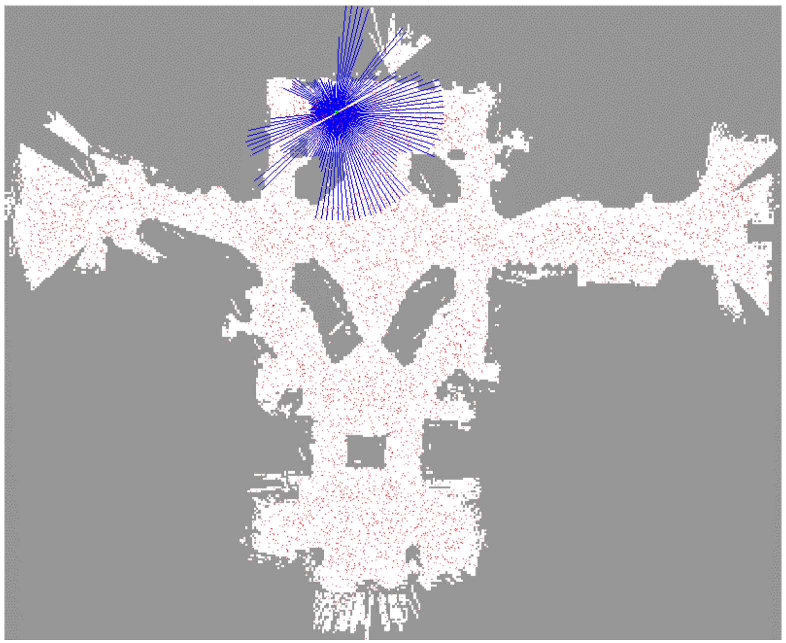


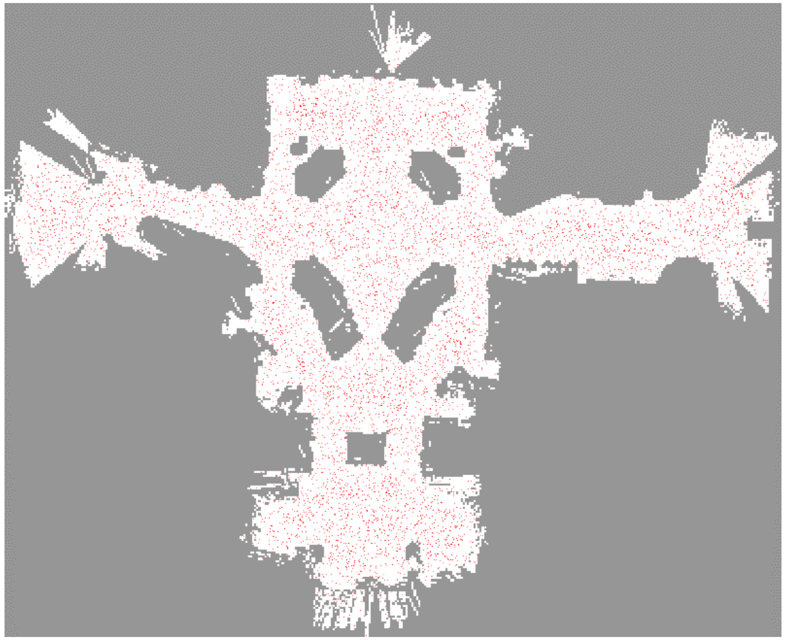
Laser sensor



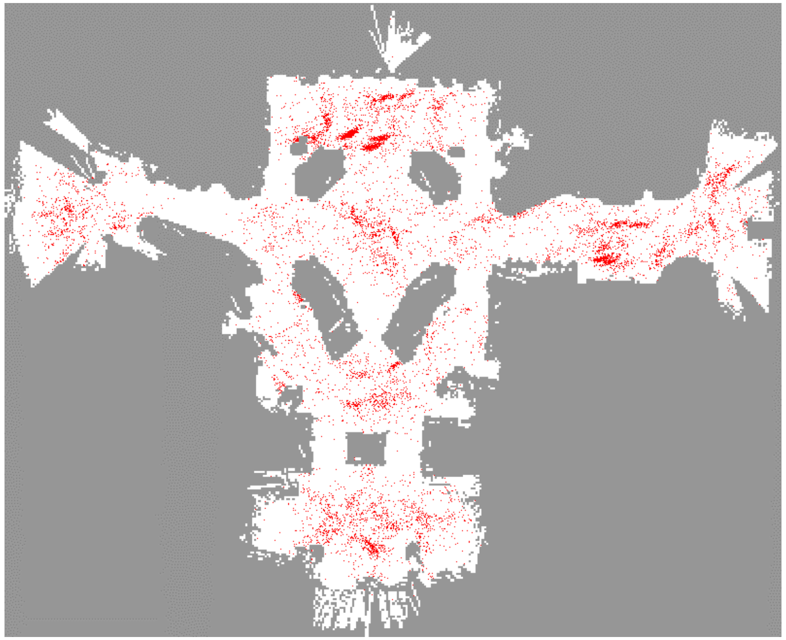
Sonar sensor



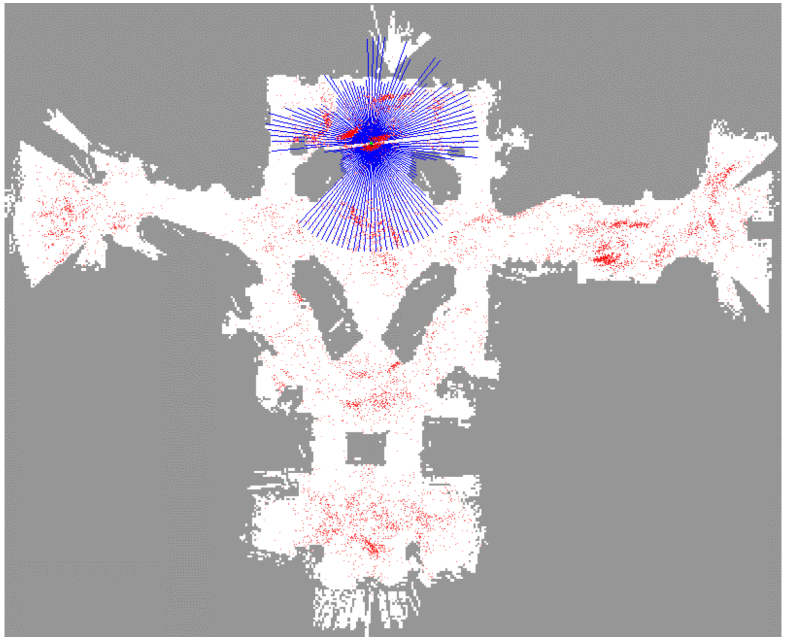




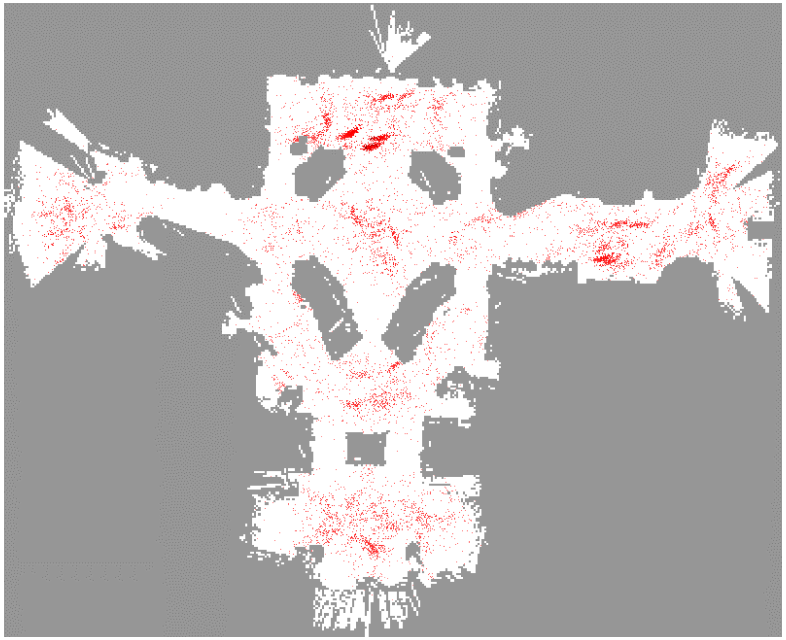
25



26

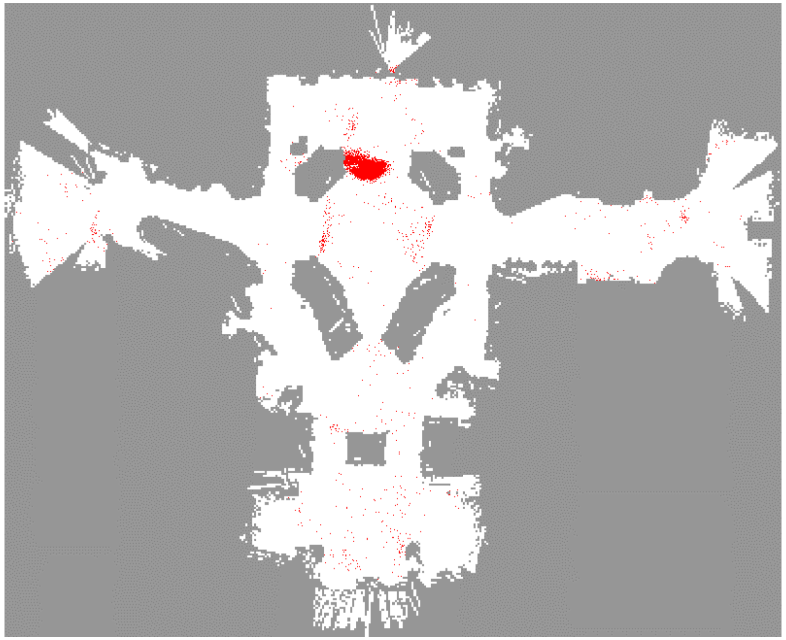


27

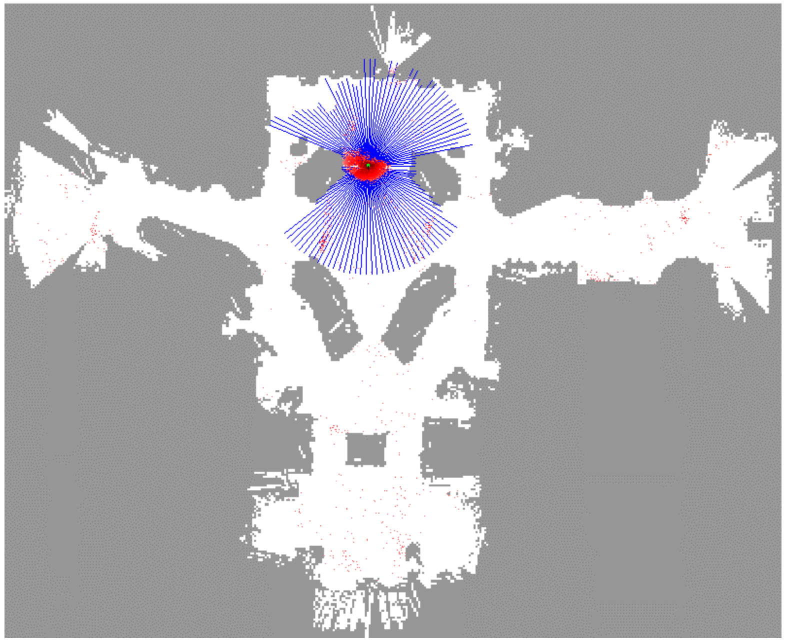


28

29

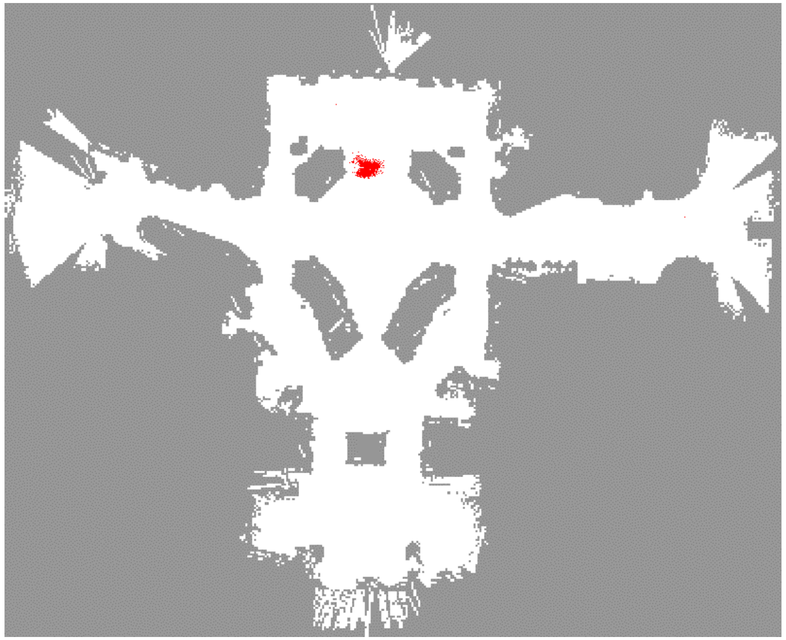


30

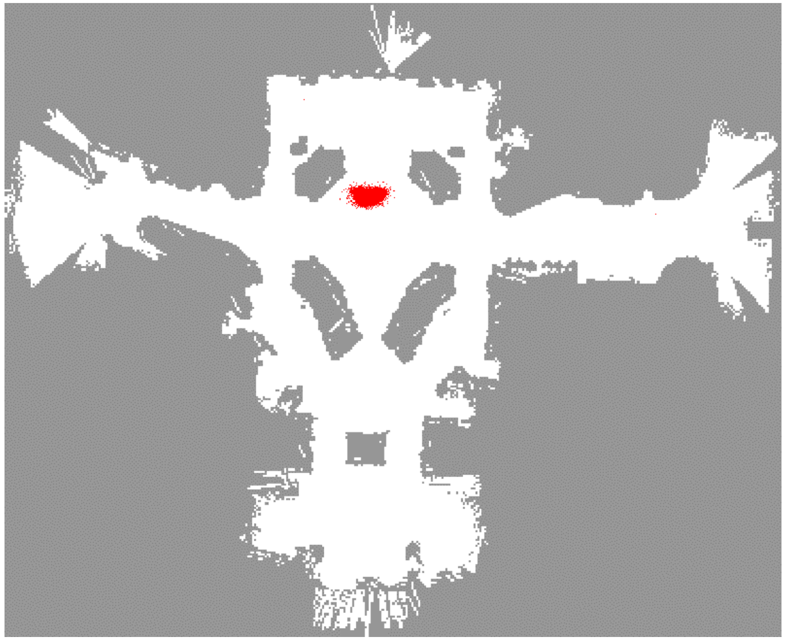


31

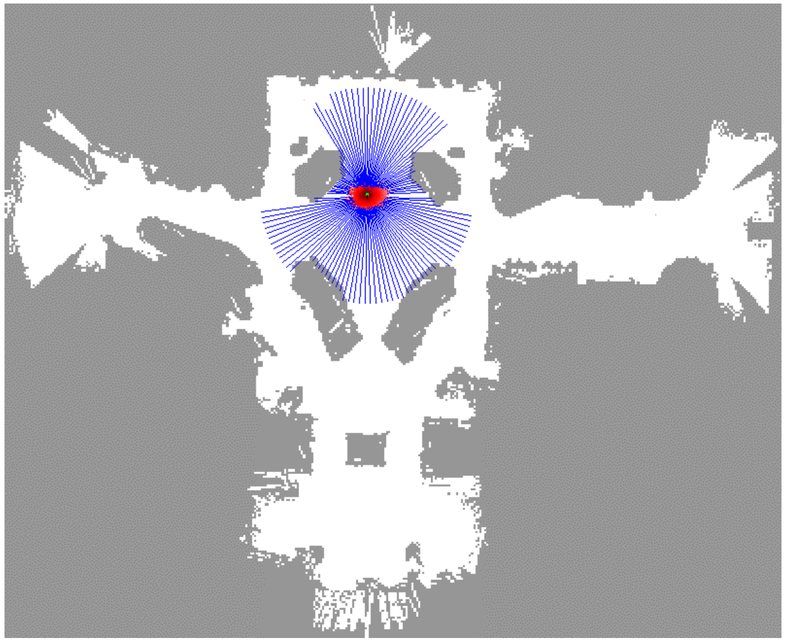
32



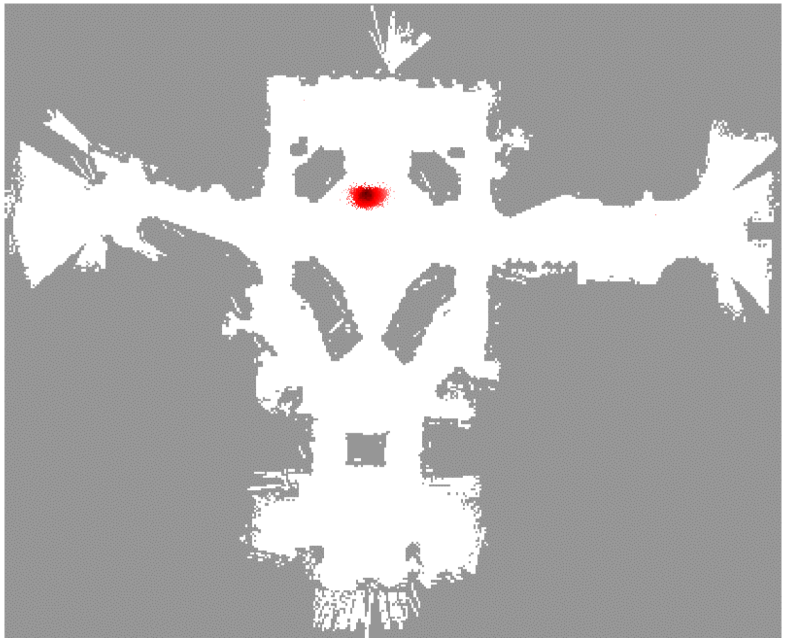
33



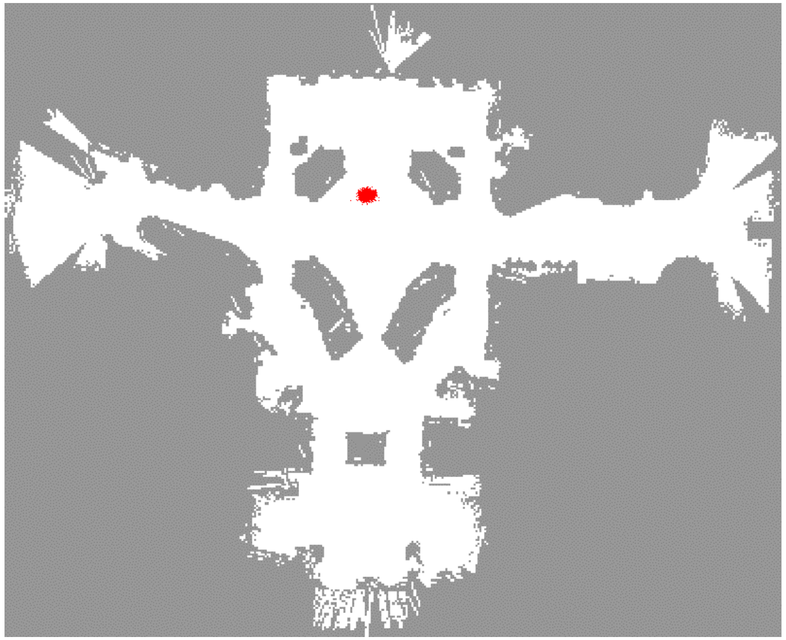
34



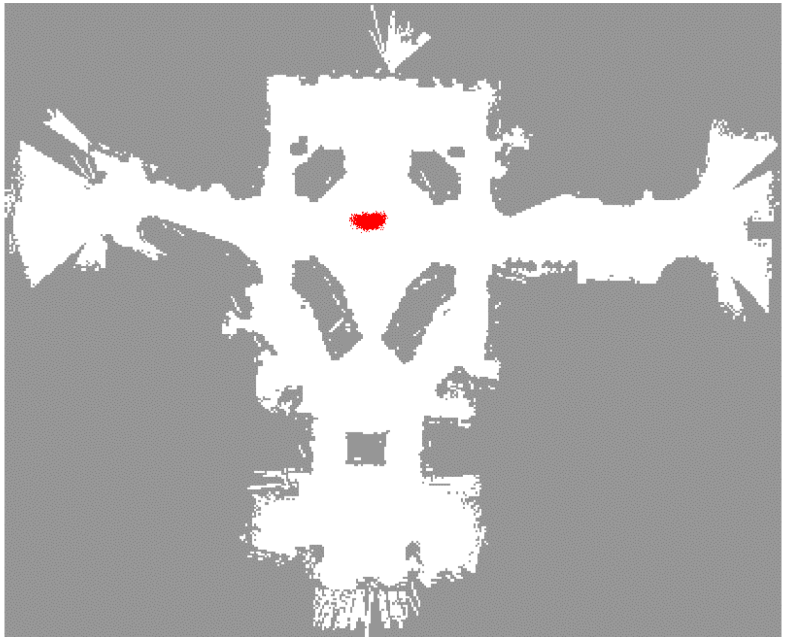
35



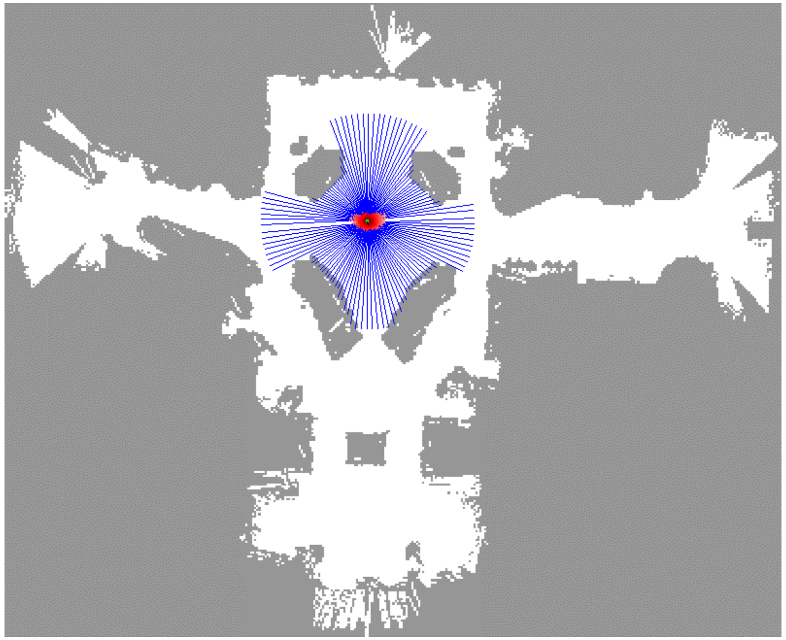
36



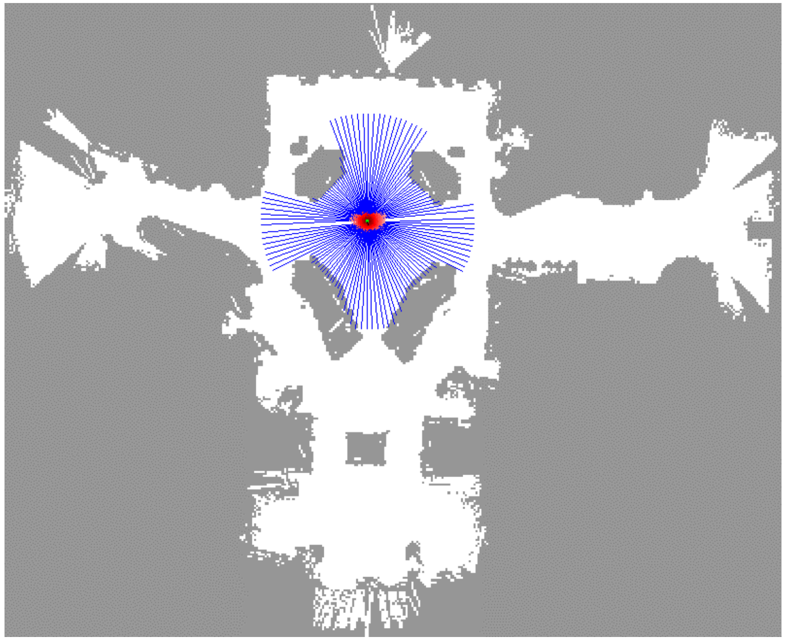
37



38

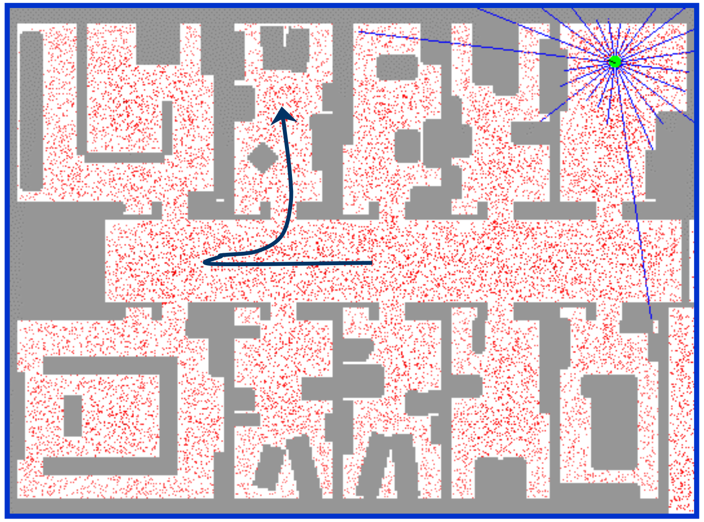


39



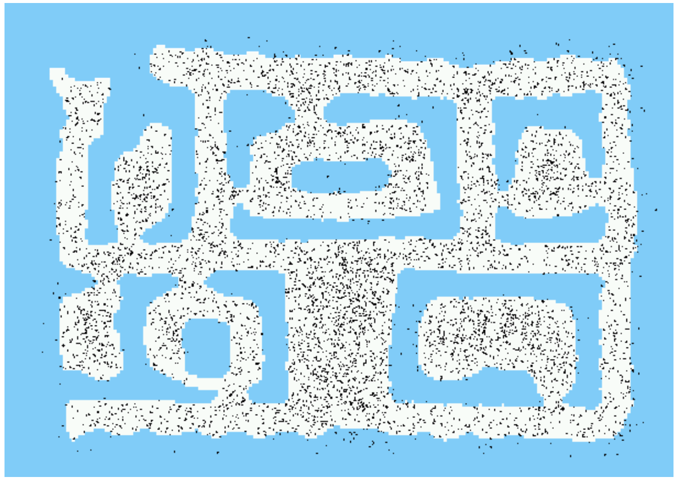
40

Sample-based Localization (Sonar)

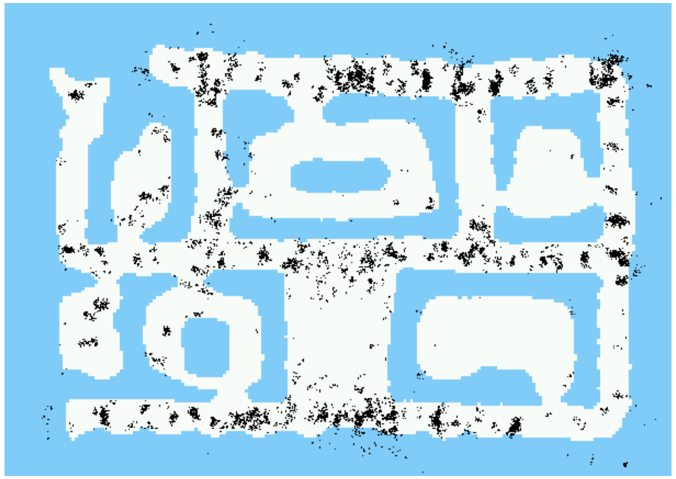


[VIDEO]

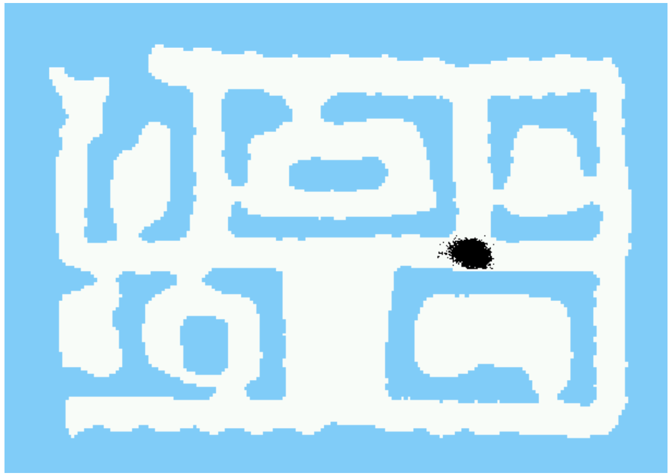
Initial Distribution



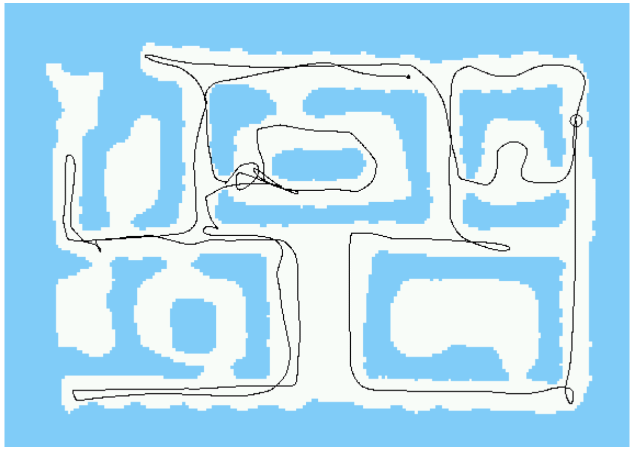
After Incorporating Ten Ultrasound Scans



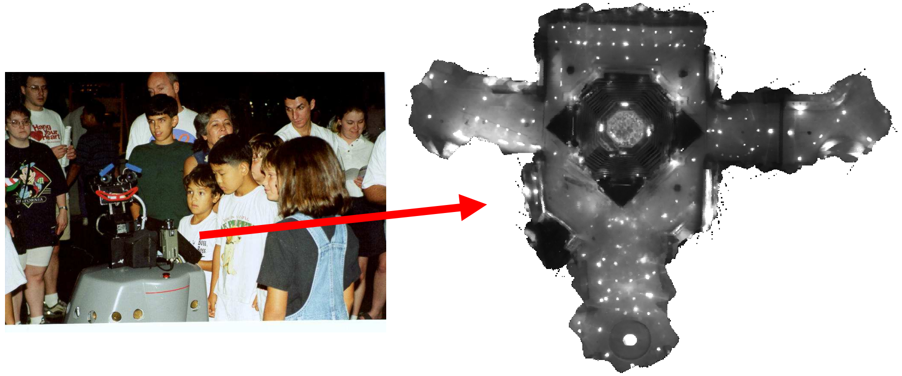
After Incorporating 65 Ultrasound Scans



Estimated Path

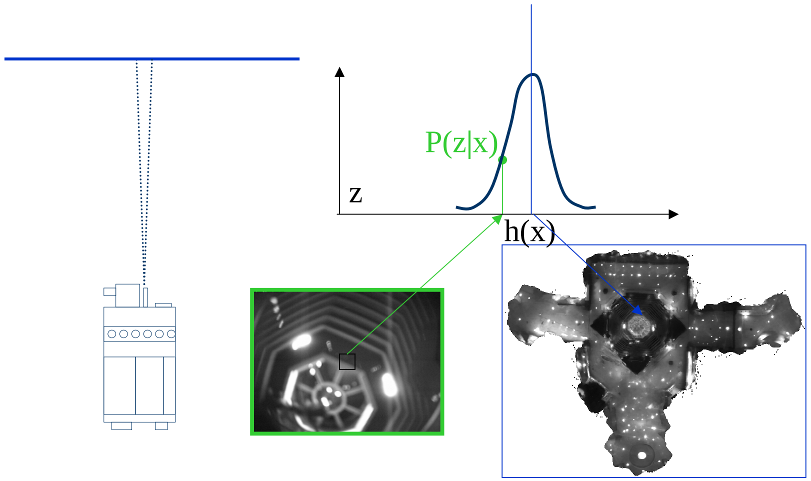


Using Ceiling Maps for Localization



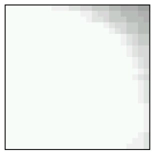
[Dellaert et al. 99]

Vision-based Localization

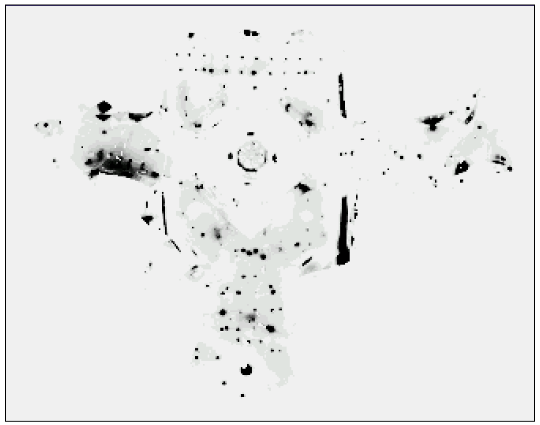


Under a Light

Measurement z:

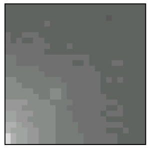


***P(z|x)*:**

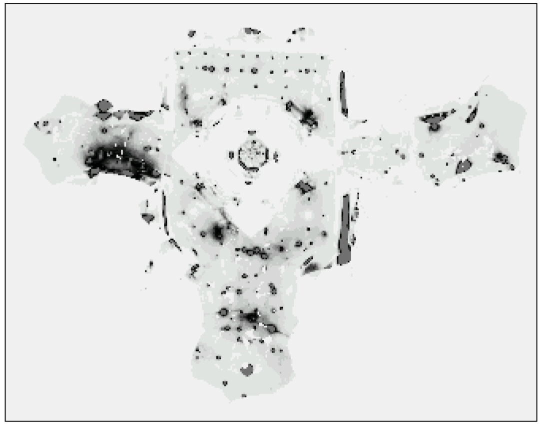


Next to a Light

Measurement z:



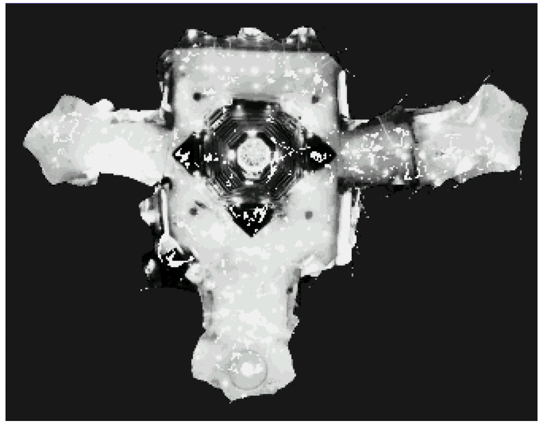
***P(z|x)*:**



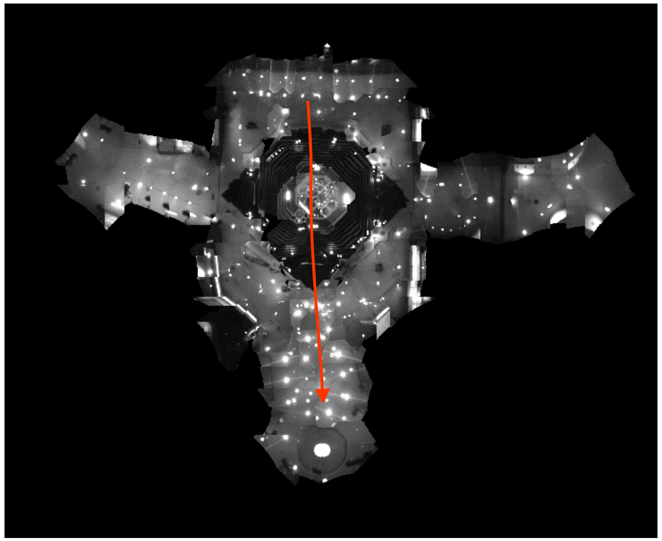
Elsewhere

Measurement z:

***P(z|x)*:**



Global Localization Using Vision



[VIDEO]

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 distribution 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

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