Monte Carlo Localization for Mobile Robots

Frank Dellaert[1], Dieter Fox[2],
Wolfram Burgard[3], Sebastian Thrun[4]
[1]Georgia Institute of Technology
[2]University of Washington
[3]University of Bonn
[4]Carnegie Mellon University

Take Home Message

Representing uncertainty using samples
is powerful, fast, and simple !

Outline

Robot Localization

Sensor ?

Density Representation ?

Monte Carlo Localization

Results

Outline

Robot Localization

Sensor ?

Density Representation ?

Monte Carlo Localization

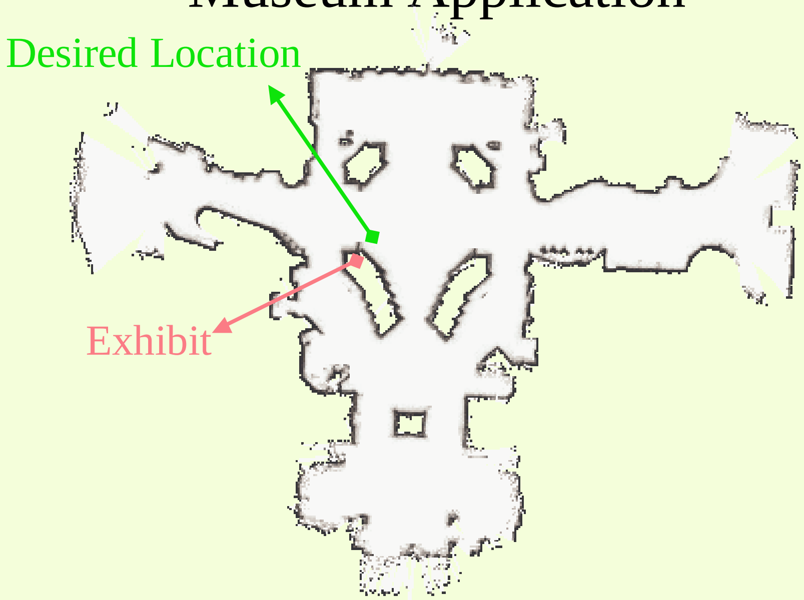
Results

Minerva

Motivation

- Crowded public spaces
- Unmodified environments

Museum Application



Global Localization

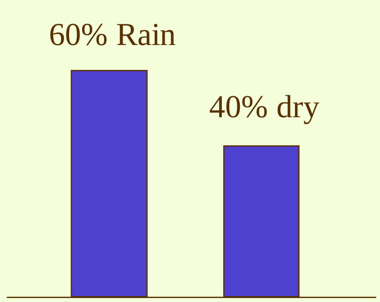
- Where in the world is Minerva the Robot ?
- Vague initial estimate
- Noisy and ambiguous sensors

Local Tracking

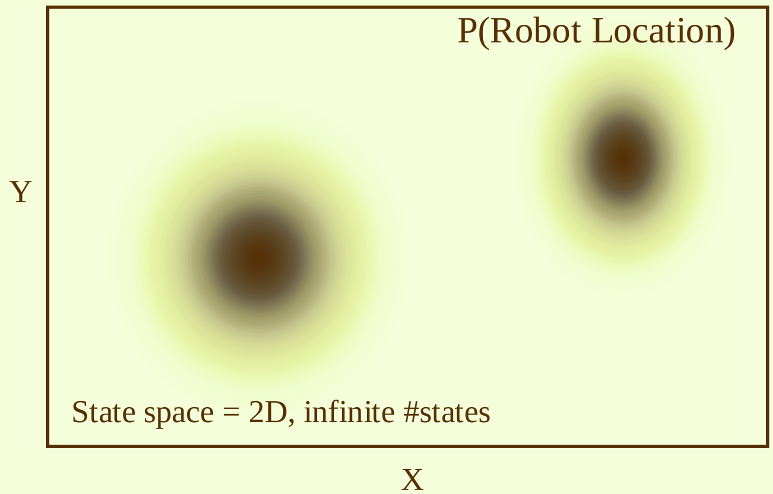
- Sharp initial estimate
- Noisy and ambiguous sensors

The Bayesian Paradigm

- Knowledge as a probability distribution

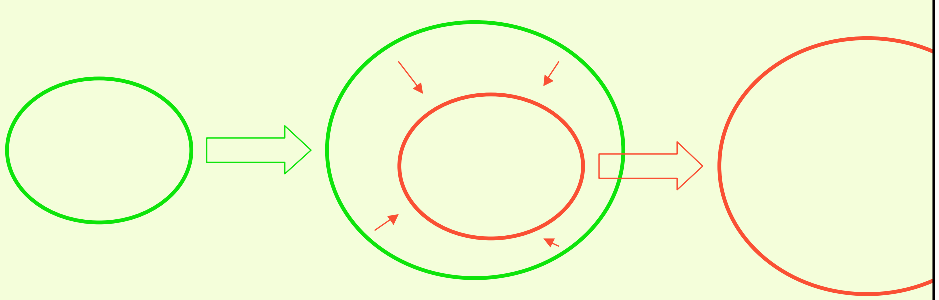


Probability of Robot Location

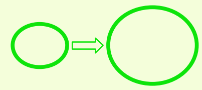


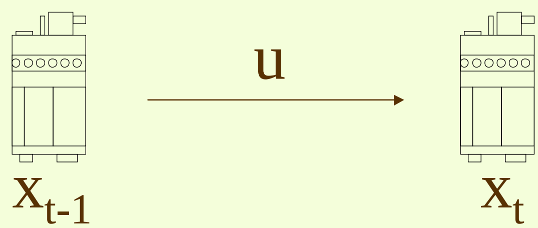
Bayesian Filtering

- Two phases: 1. Prediction Phase
  2. Measurement Phase



1. Prediction Phase

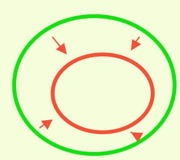


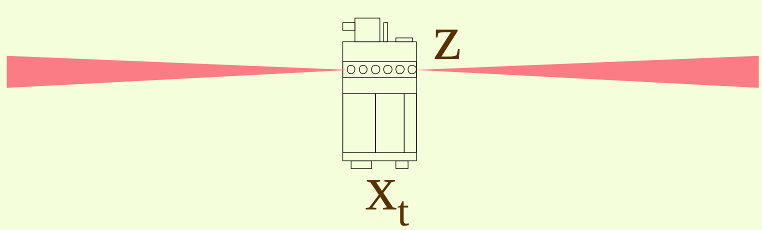


$$P(x_t) = \sum P(x_t|x_{t-1},u)\, P(x_{t-1})$$

Motion Model

2. Measurement Phase





$$P(x_t|z) = k\, P(z|x_t)\, P(x_t)$$

Sensor Model

Outline

Robot Localization

Sensor ?

Density Representation ?

Monte Carlo Localization

Results

What sensor ?

- Sonar ?
- Laser ?
- Vision ?

Problem: Large Open Spaces

- Walls and obstacles out of range
- Sonar and laser have problems
- One solution: Coastal Navigation

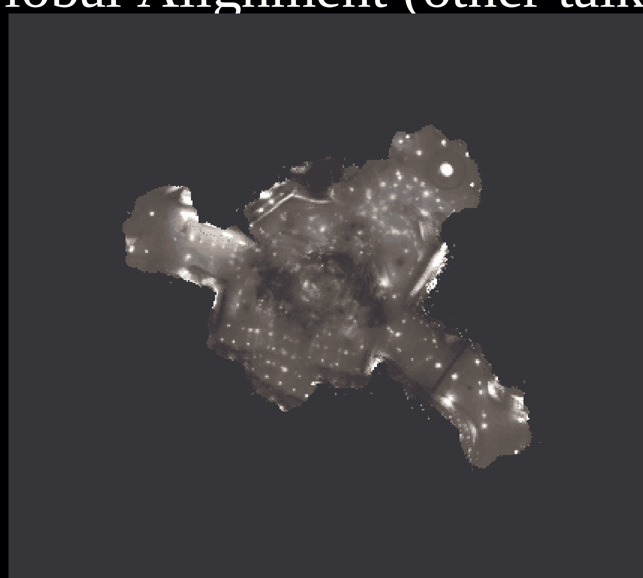
Problem: Large Crowds

- Horizontally mounted sensors have problems
- One solution: Robust filtering

Solution: Ceiling Camera

- Upward looking camera
- Model of the world = Ceiling Mosaic

Global Alignment (other talk)



Ceiling Mosaic

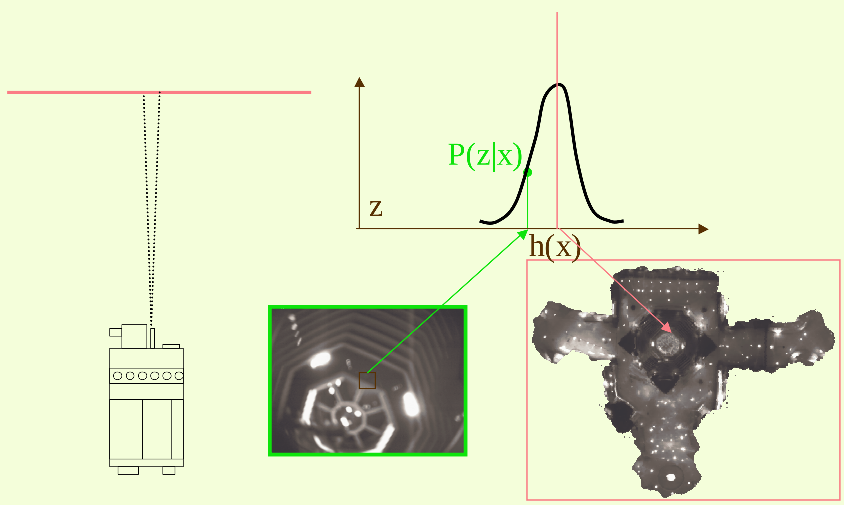
Large FOV Problems

- 3D ceiling -> 3D Model ?
- Matching whole images slow

Small FOV Solution

- Model = orthographic mosaic
- No 3D Effects
- Very fast

Vision based Sensor



Outline

Robot Localization

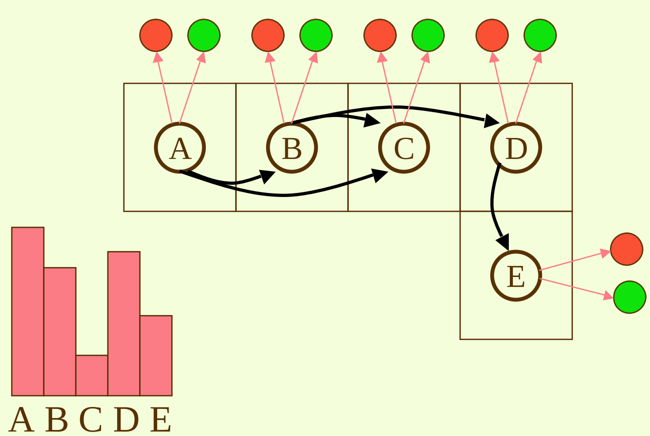
Sensor ?

Density Representation ?

Monte Carlo Localization

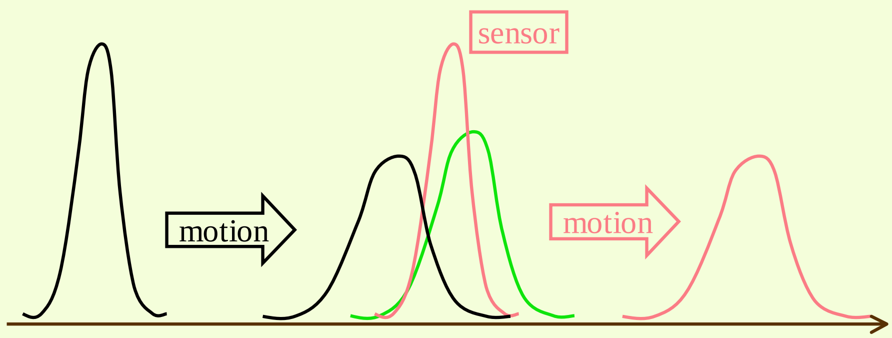
Results

Hidden Markov Models



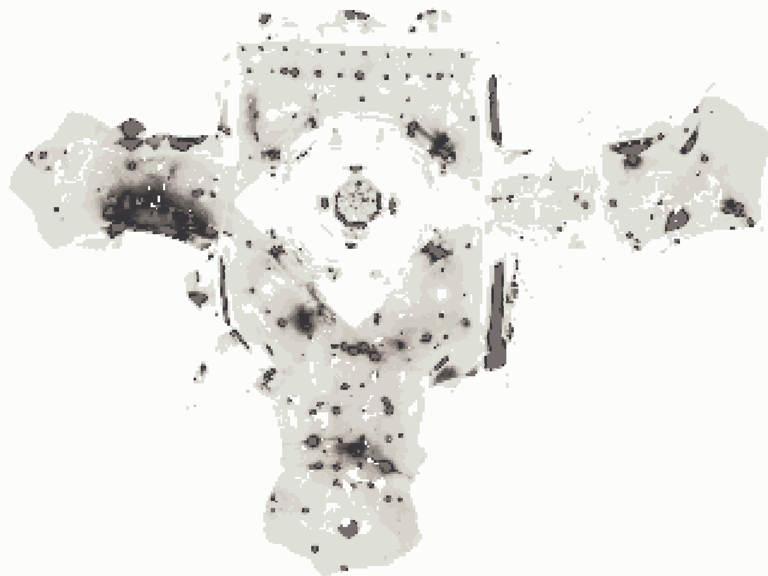
Kalman Filter

- Very powerful
- Gaussian, unimodal



Under Light

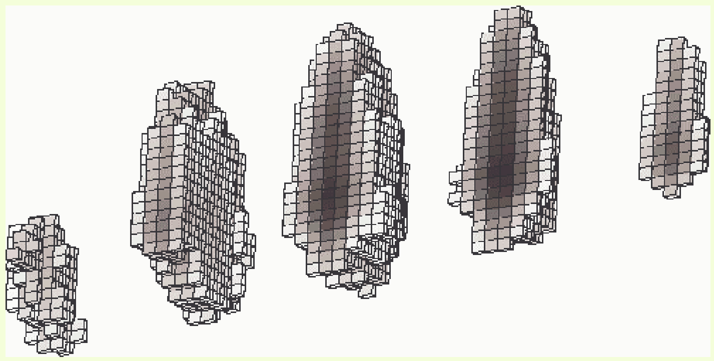
Next to Light



Elsewhere

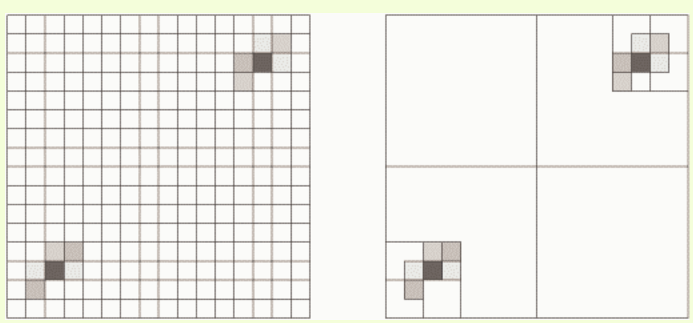
Markov Localization

- Fine discretization over {x,y,theta}
- Very successful: Rhino, Minerva, Xavier…



Dynamic Markov Localization

- Burgard et al., IROS 98
- Idea: use Oct-trees



Sampling as Representation

Samples <=> Densities

- Density => samples
  Obvious

- Samples => density
  Histogram, Kernel Density Estimation

Sampling Advantages

- Arbitrary densities
- Memory = O(#samples)
- Only in “Typical Set”
- Great visualization tool !

- minus: Approximate

Outline

Robot Localization

Sensor ?

Density Representation ?

Monte Carlo Localization

Results

Disclaimer

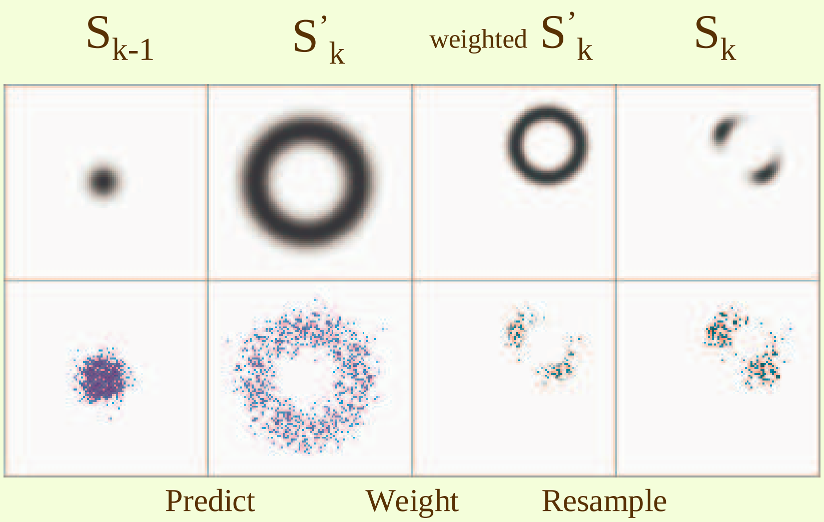
- Handschin 1970 (!)
  lacked computing power

- Bootstrap filter 1993 Gordon et al.
- Monte Carlo filter 1996 Kitagawa
- Condensation 1996 Isard & Blake

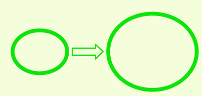
Added Twists

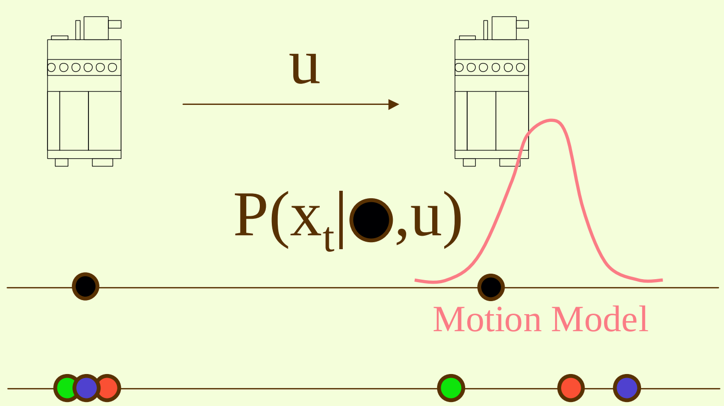
- Camera moves, not object
- Global localization

Monte Carlo Localization

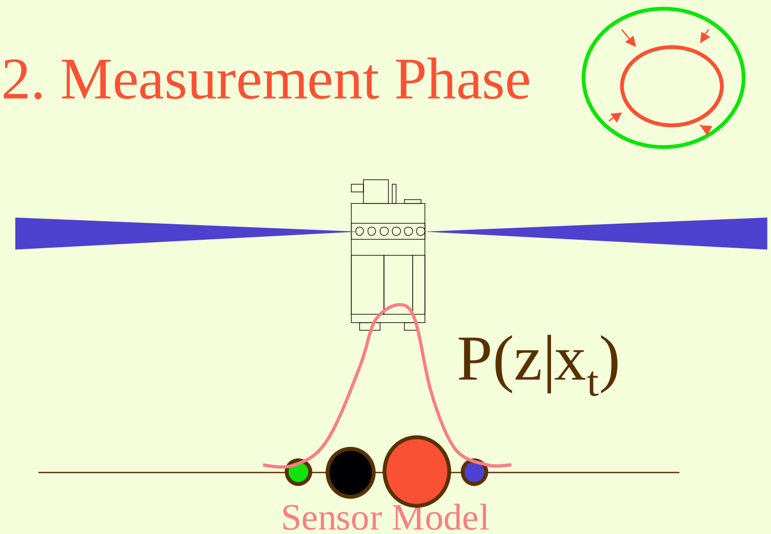


1. Prediction Phase

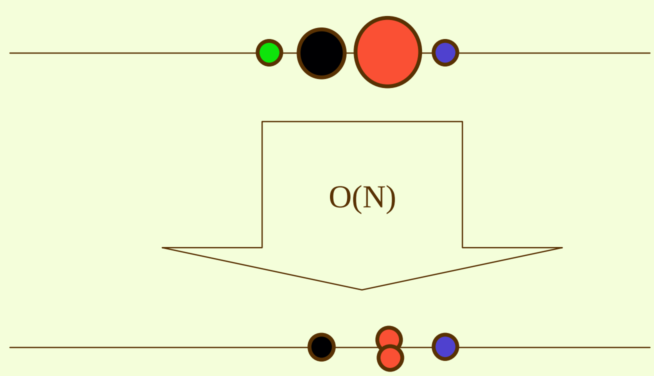




2. Measurement Phase



3. Resampling Step



A more in depth look

Bayes Law, new look

- Densities:
  update prior p(x) to p(x|z) via l(x;z)

- Samples
  update a sample from p(x) to a sample from the posterior p(x|z) through l(x;z)

Bayes Law Problem

- We really want $p(x|z)$ samples
- But we only have $p(x)$ samples !
- How can we upgrade $p(x)$ to $p(x|z)$?

More General Problem

- We really want $h(x)$ samples
- But we only have $g(x)$ samples !
- How can we upgrade $g(x)$ to $h(x)$?

Solution = Importance Sampling

- 1. generate x_i from $g(x)$
- 2. calculate $w_i = h(x_i)/g(x_i)$
- 3. assign weight $q_i = w_i / \sum w_i$

- Still works if $h(x)$ only known up to normalization factor

Mean and Weighted Mean

- Fair sample:
  obtain samples x_i from $p(x|z)$
  $E[m(x)|z] \sim \sum m(x_i)/N$

- Weighted sample:
  obtain weighted samples (x_i,q_i) from $p(x|z)$
  $E[m(x)|z] \sim \sum q_i \, m(x_i)$

Bayes Law using Samples

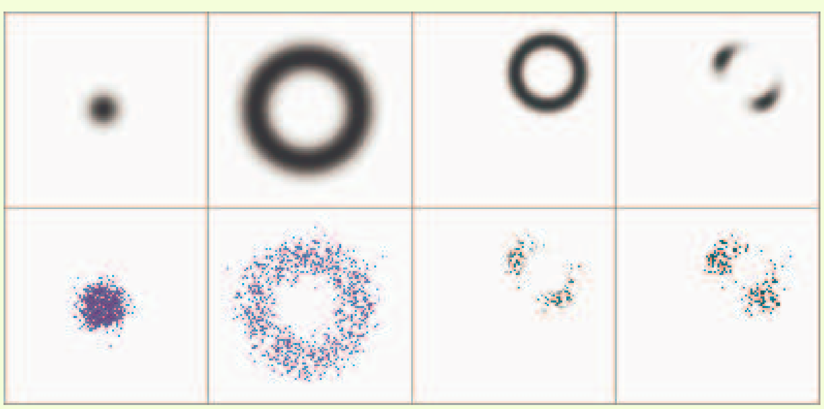
- 1. generate x_i from $p(x)$
- 2. calculate $w_i = l(x_i;z)$
- 3. assign weight $q_i = w_i / \sum w_i$

- Indeed: $w_i = p(x|z)/p(x) = l(x;z)\, p(x) / p(x) = l(x;z)$

- 4. if you want, resample from (x_i, q_i)

Monte Carlo Localization

S_{k-1} S'_k weighted S'_k S_k



Predict Weight Resample

Outline

Robot Localization

Sensor ?

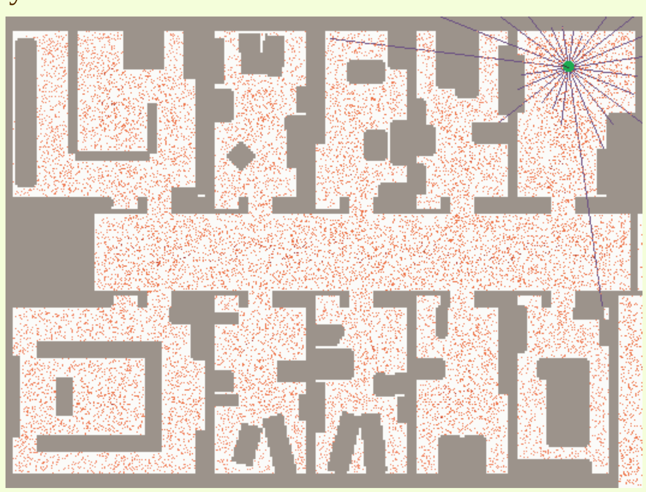
Density Representation ?

Monte Carlo Localization

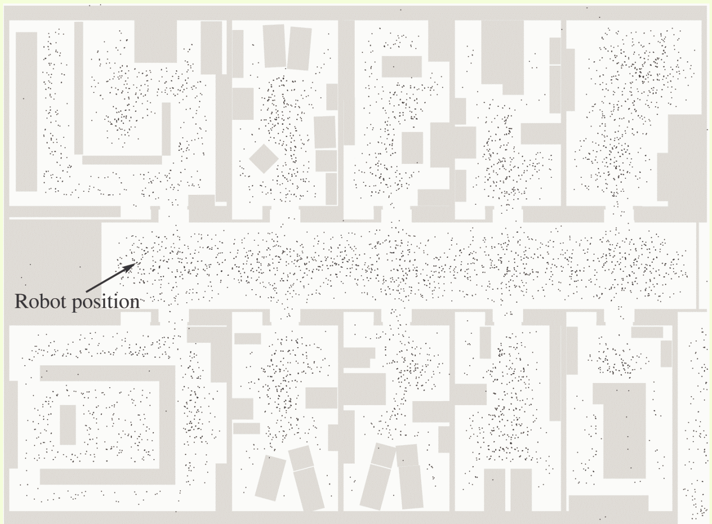
Results

- Office Environment
- Sonar Sensors
- Global Localization
- Symmetry confusion

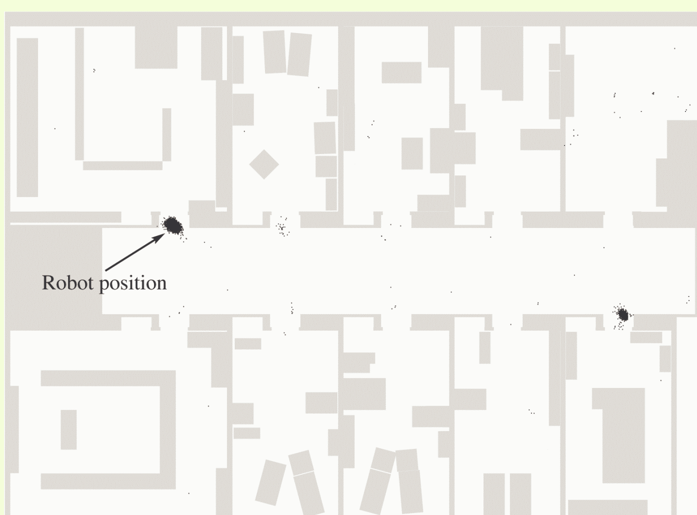
Video A



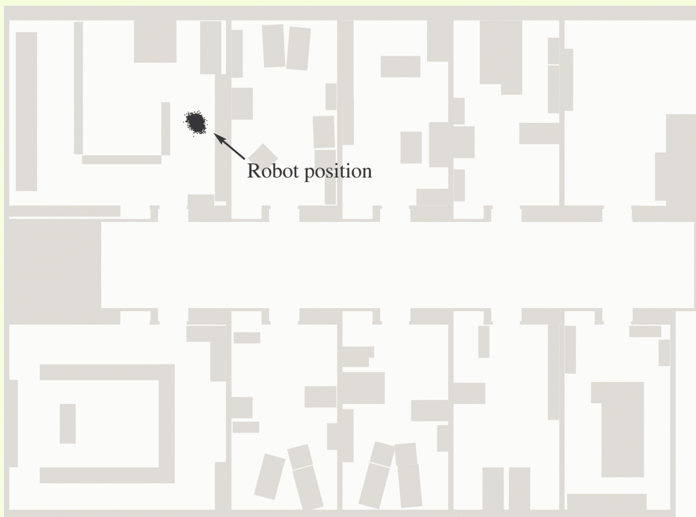
Global Localization



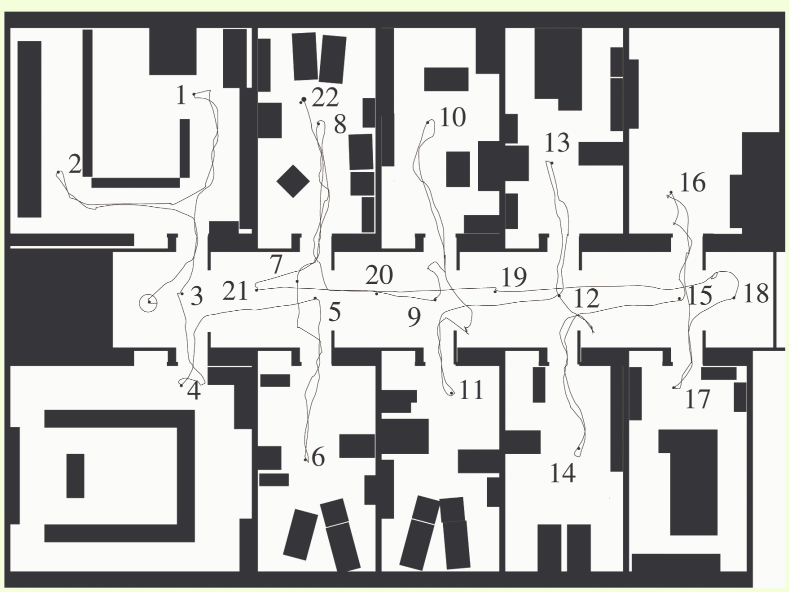
Global Localization (2)



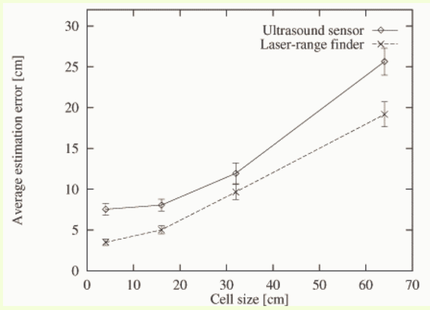
Global Localization (3)

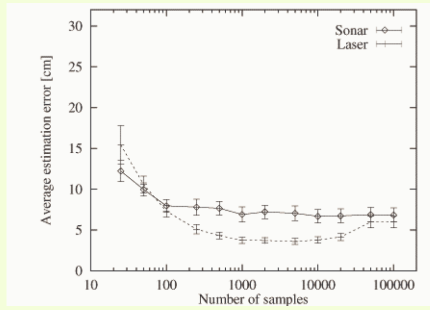


Reference Path



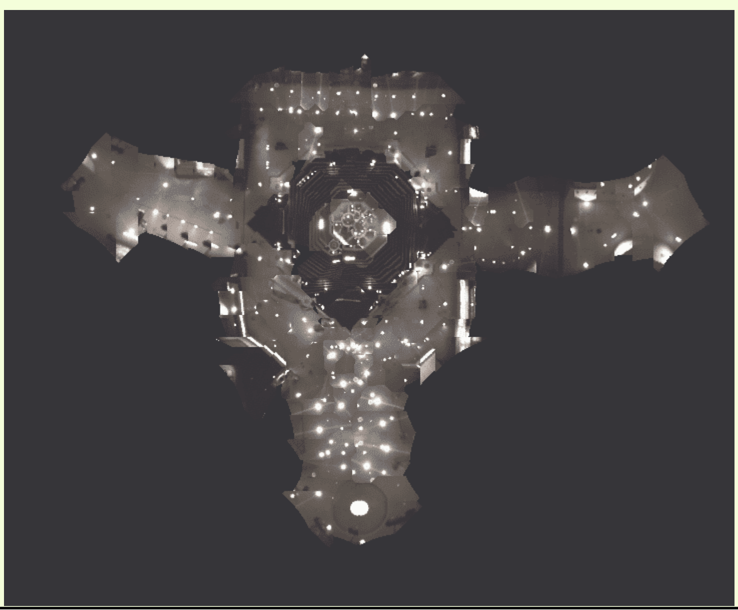
Accuracy



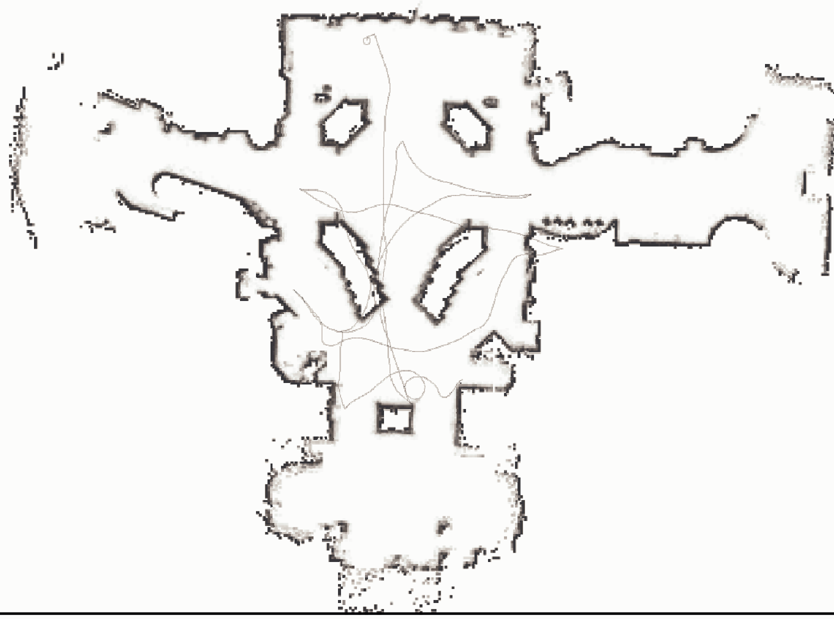


- Smithsonian Museum of American History
- Ceiling Camera, Global Localization

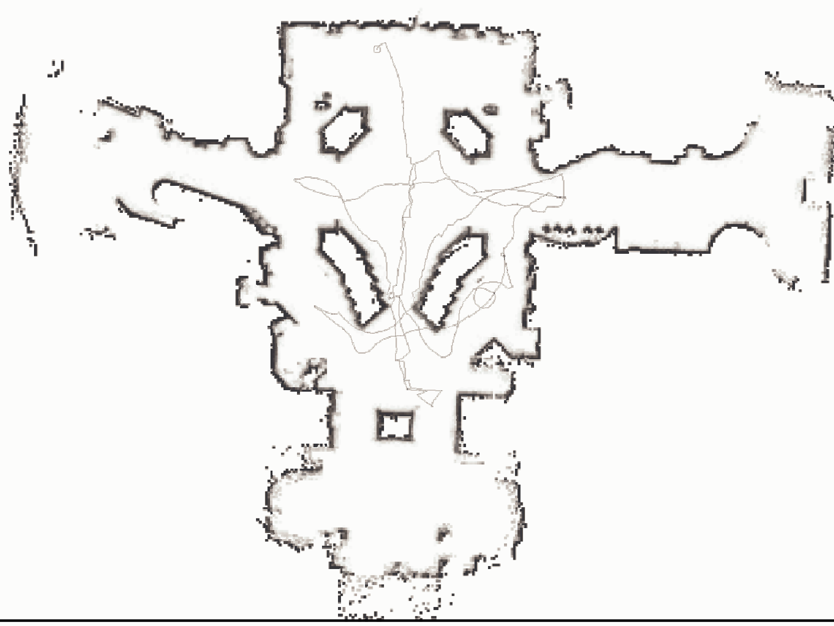
Video B



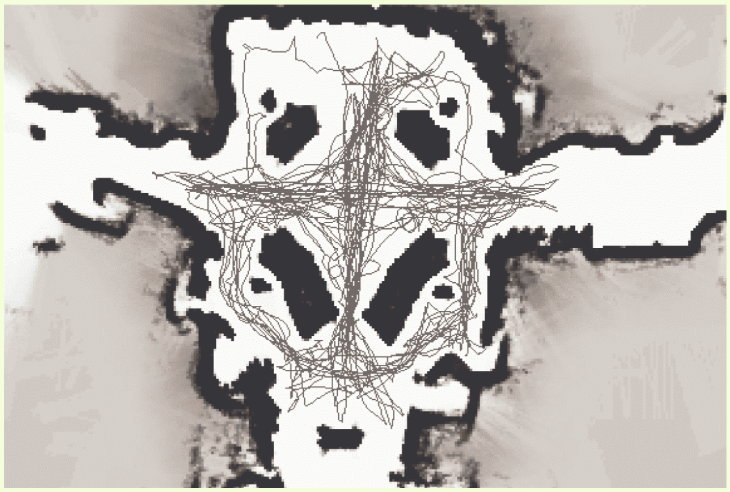
Odometry Only

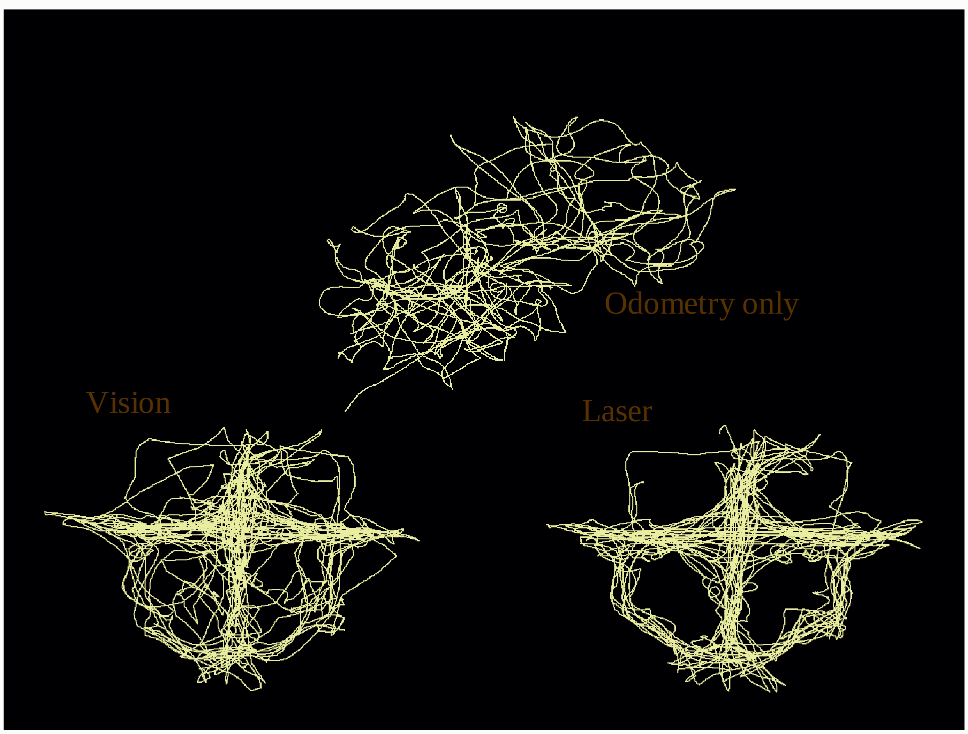


Using Vision



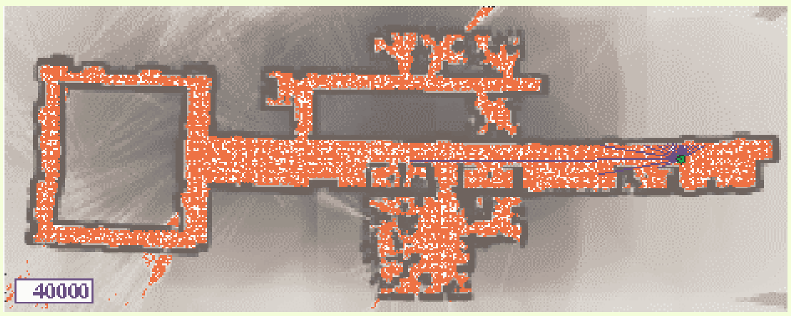
Fast Internet Morning





- UW Sieg Hall
- Laser

Video C



Conclusions

- Monte Carlo Localization:
  Powerful yet efficient
  Significantly less memory and CPU
  Very simple to implement

- Future:
  discrete states, rate information, distributed

Take Home Message

Representing uncertainty using samples
is powerful, fast, and simple !