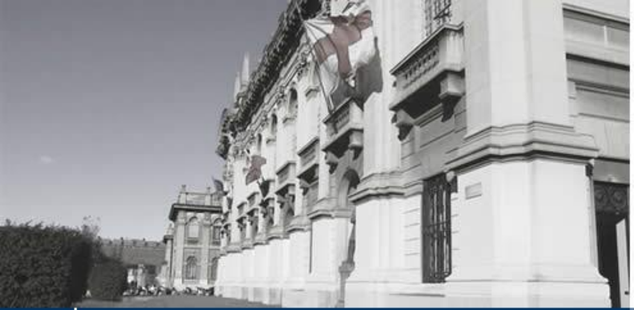
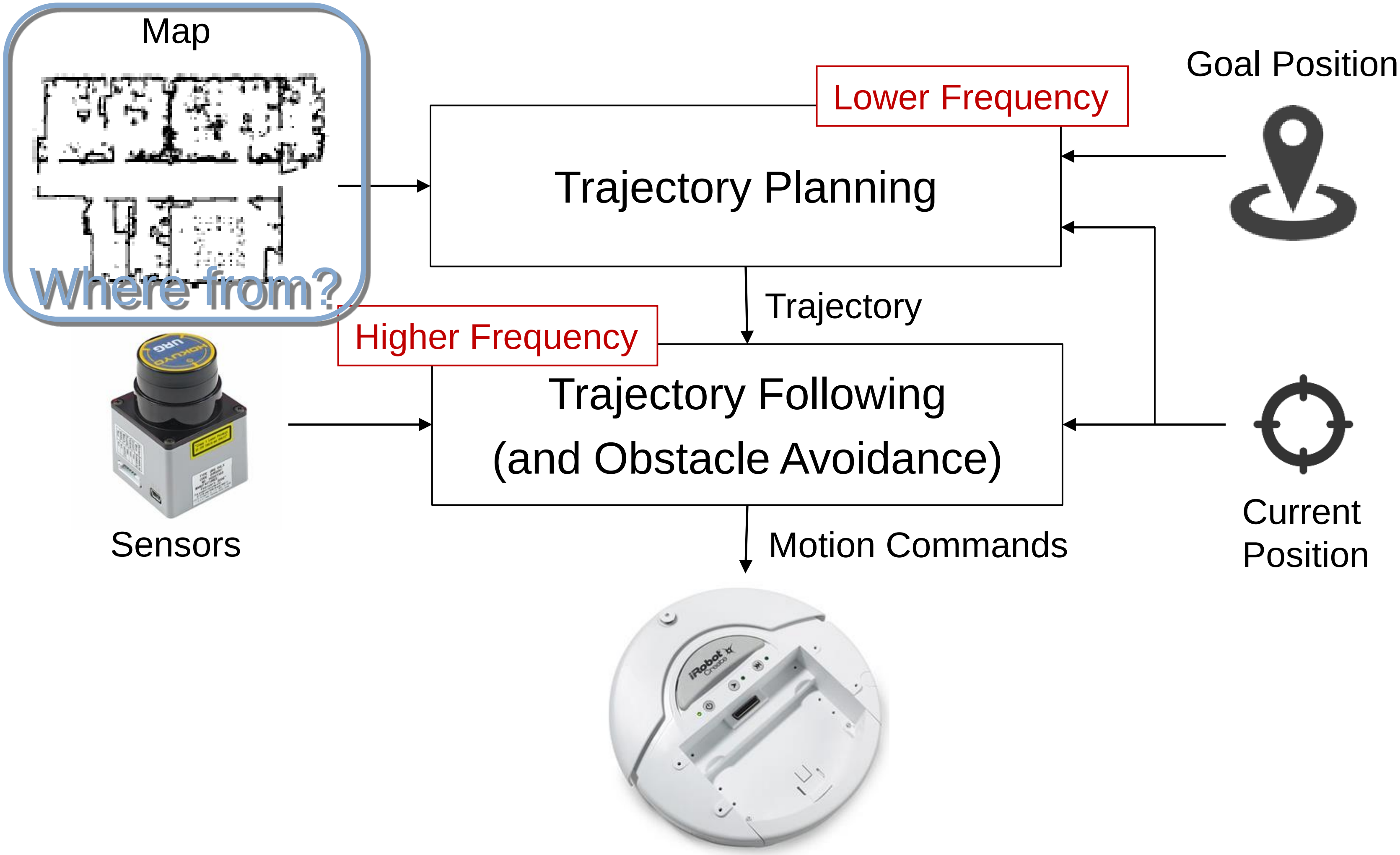


POLITECNICO DI MILANO

SLAM with Lasers

Matteo Matteucci – matteo.matteucci@polimi.it

A Two Layered Approach



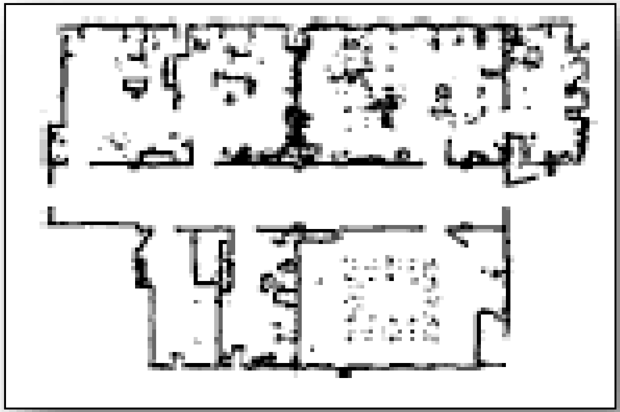
Where Am I?

To perform their tasks autonomous robots and unmanned vehicles need

- To know where they are (e.g., Global Positioning System)
- To know the environment map (e.g., Geographical Institutes Maps)

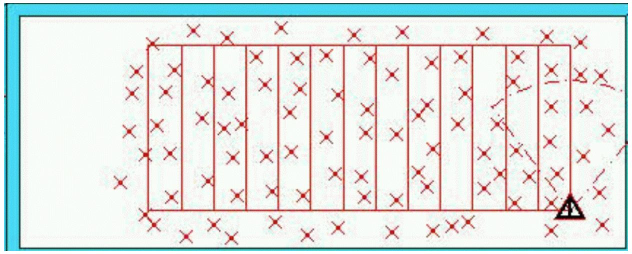
These are not always possible or reliable

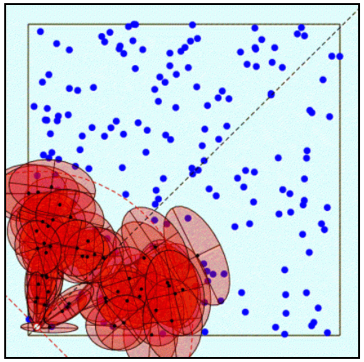
- GNSS are not always reliable/available
- Not all places have been mapped
- Environment changes dynamically
- Maps need to be updated

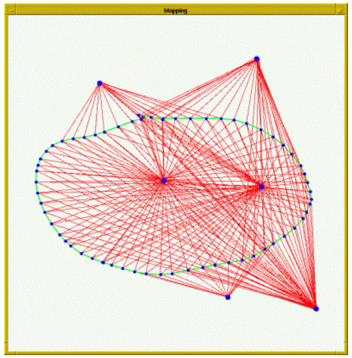


Representations

Landmark-based

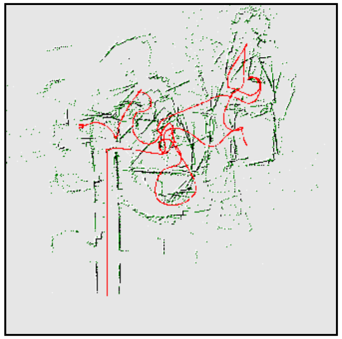


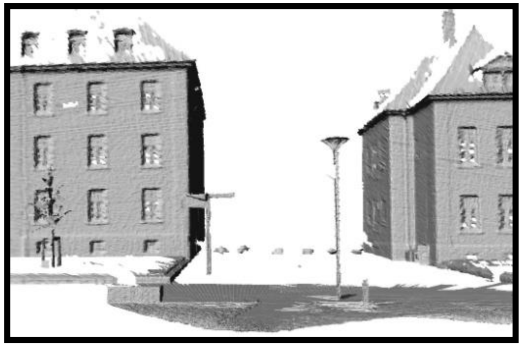


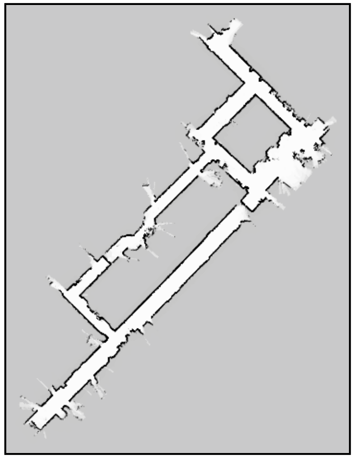


[Leonard et al., 98; Castelanos et al., 99: Dissanayake et al., 2001; Montemerlo et al., 2002;…]

Grid maps or scans

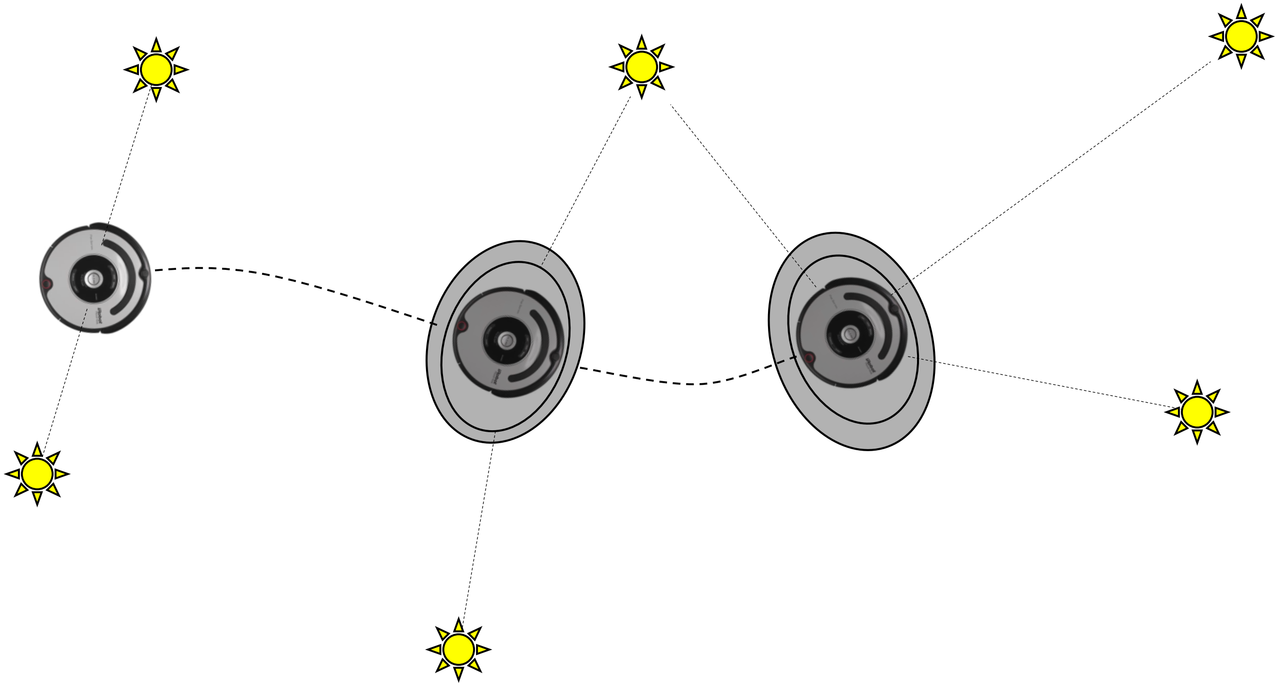




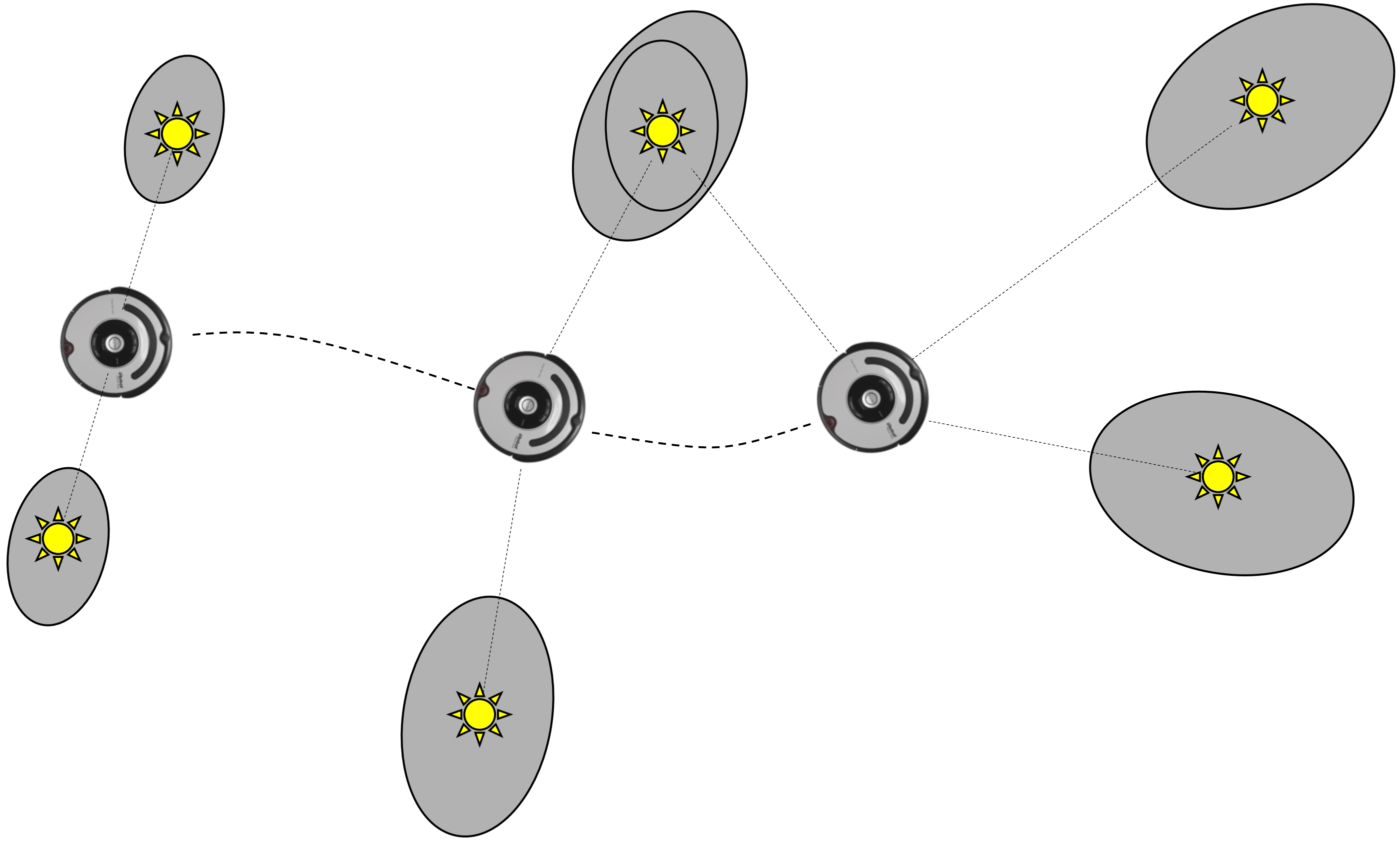


[Lu & Milios, 97; Gutmann, 98: Thrun 98; Burgard, 99; Konolige & al., 00; Thrun, 00; Arras, 99; Haehnel, 01;…]

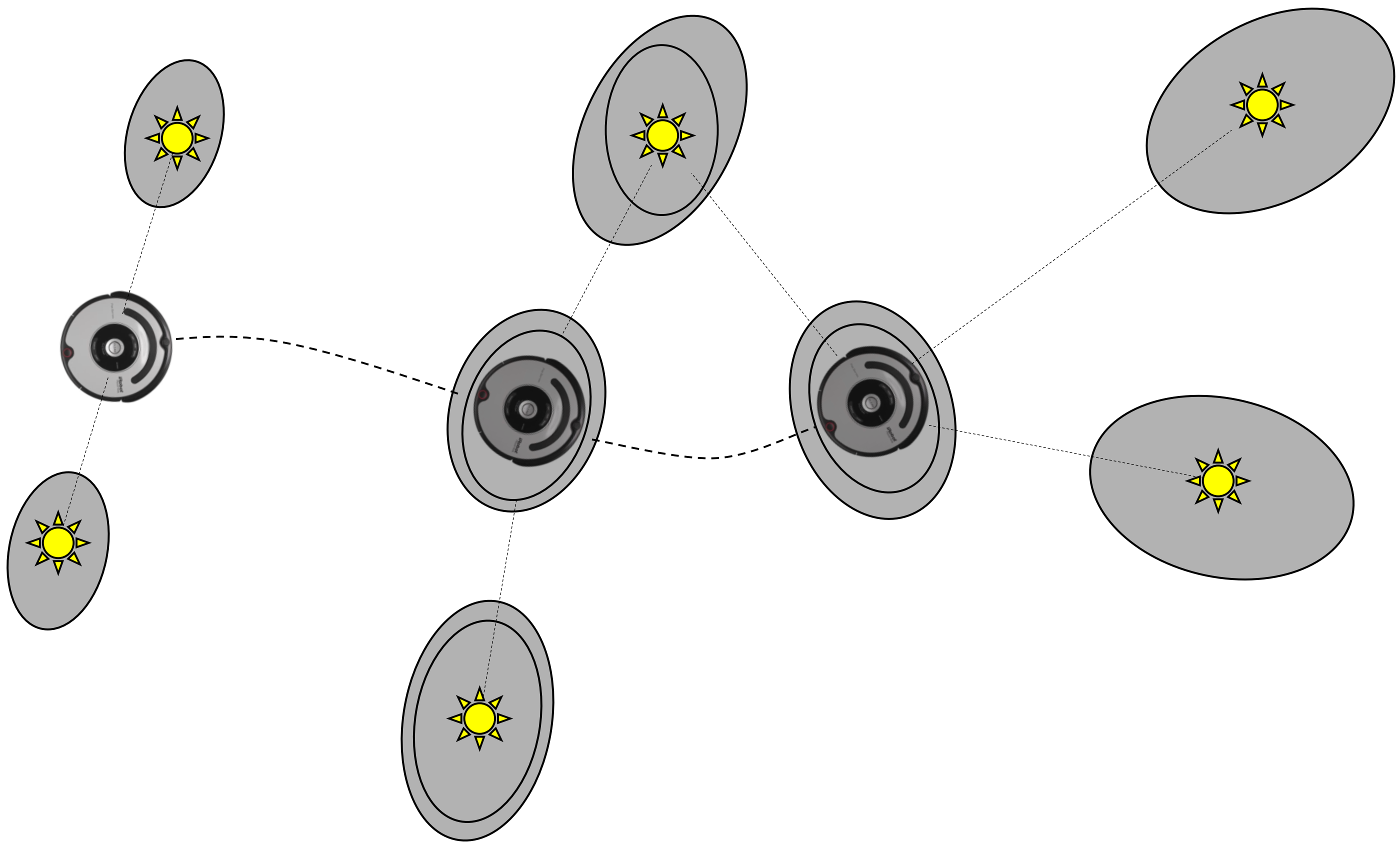
Localization … with known map



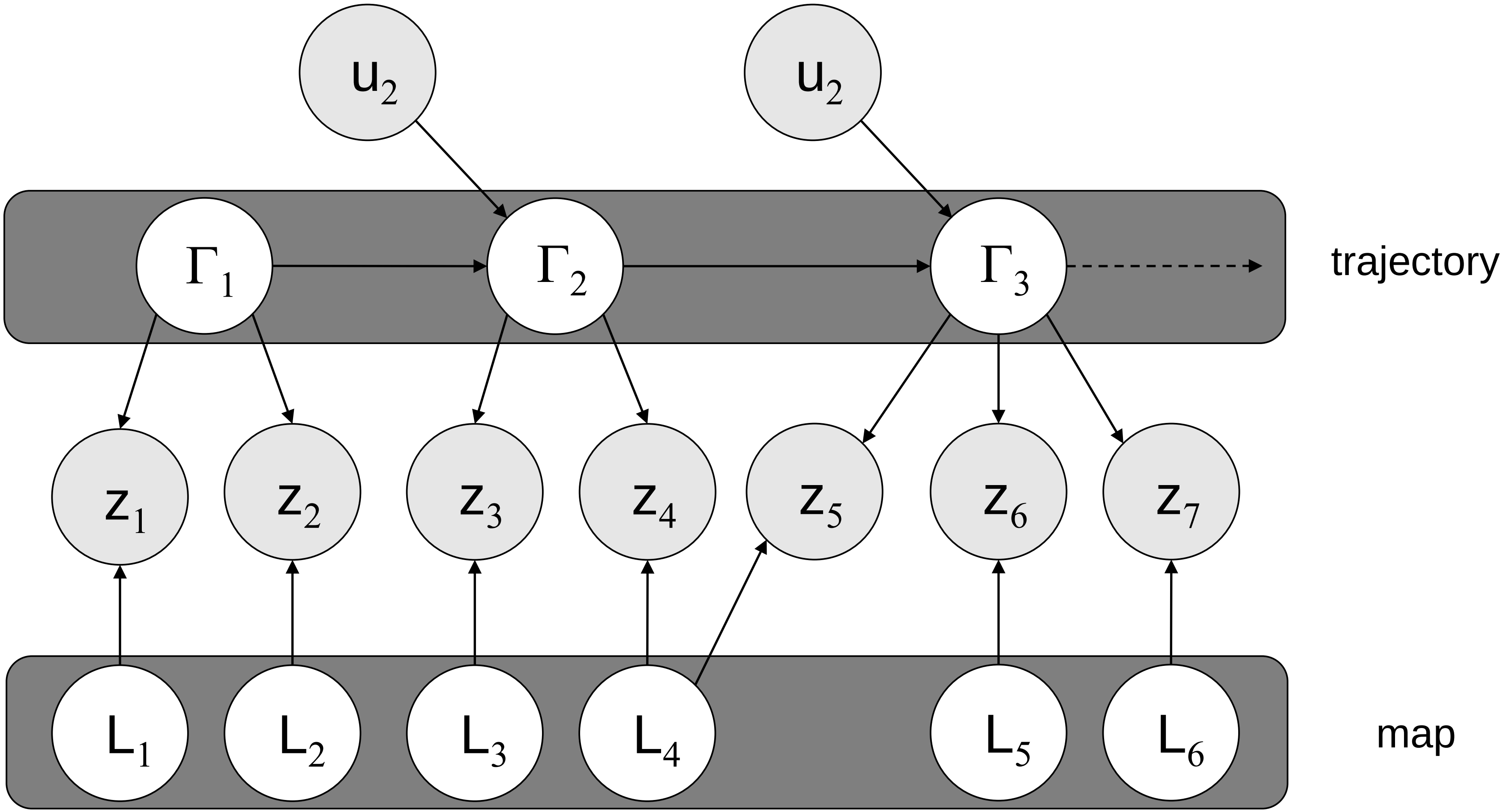
Mapping … with known poses



Simultaneous Localization and Mapping

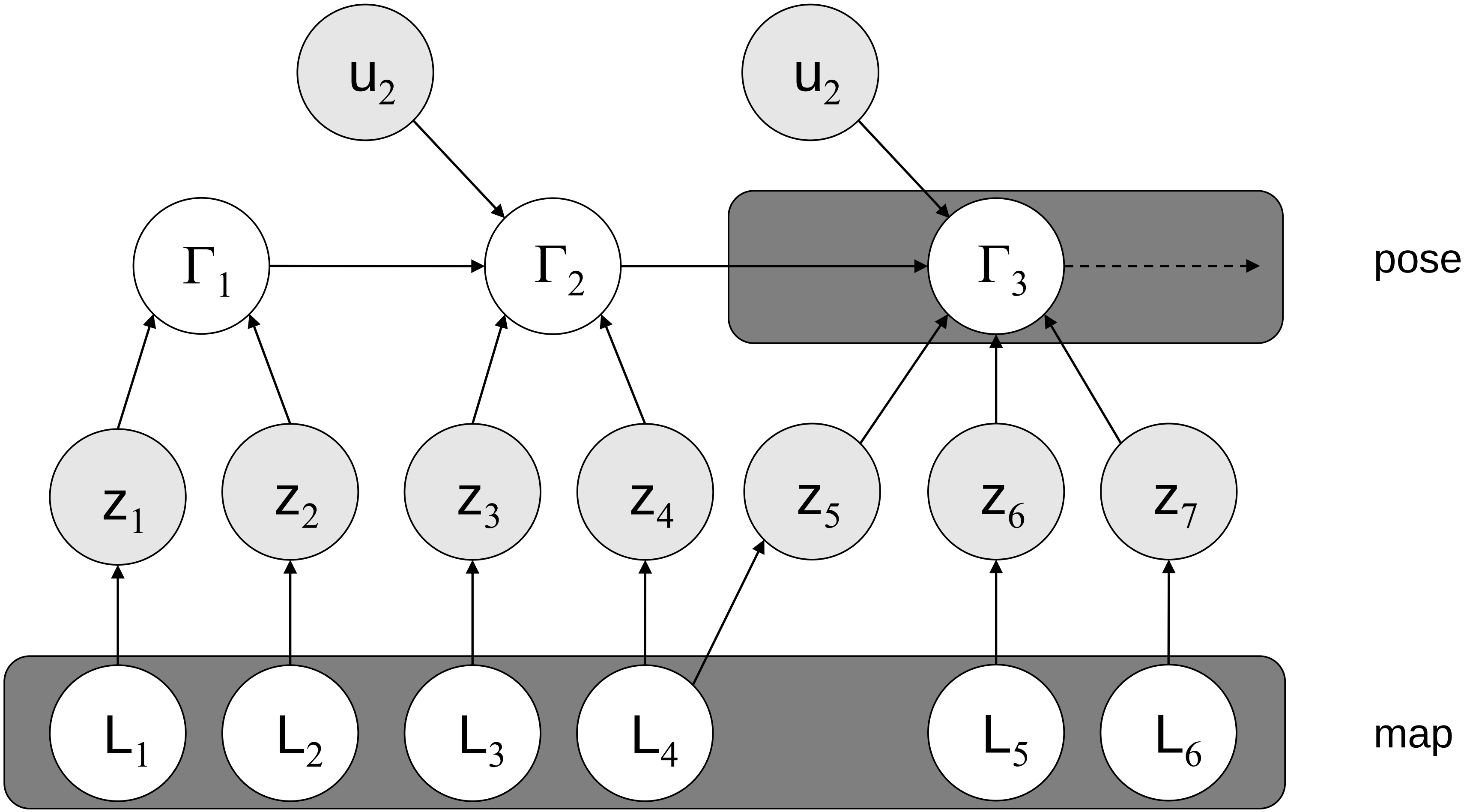


Dynamic Bayes Network Inference and Full SLAM



$$\text{Smoothing} \quad : \quad p(\Gamma_{1:t}, l_1, \ldots, l_N \mid Z_{1:t}, U_{1:t})$$

Dynamic Bayes Network Inference and Online SLAM

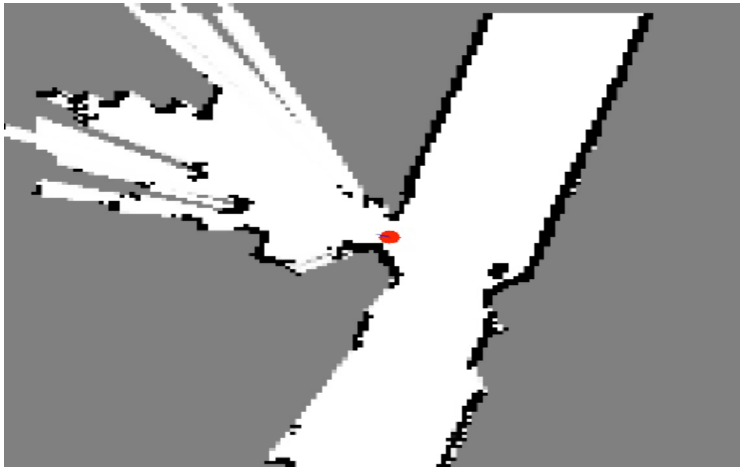


Filtering : $p(\Gamma_t, l_1, \ldots, l_N \mid Z_{1:t}, U_{1:t}) = \iiint\limits_{1:t-1} p(\Gamma_{1:t}, l_1, \ldots, l_N \mid Z_{1:t}, U_{1:t})$

Occupancy Grids

A simple 2D representation for the map

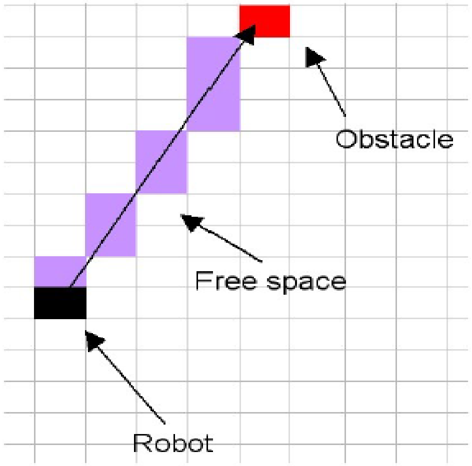
- Each cell is assumed independent
- Probability of a cell being occupied is being estimated through the Bayes theorem



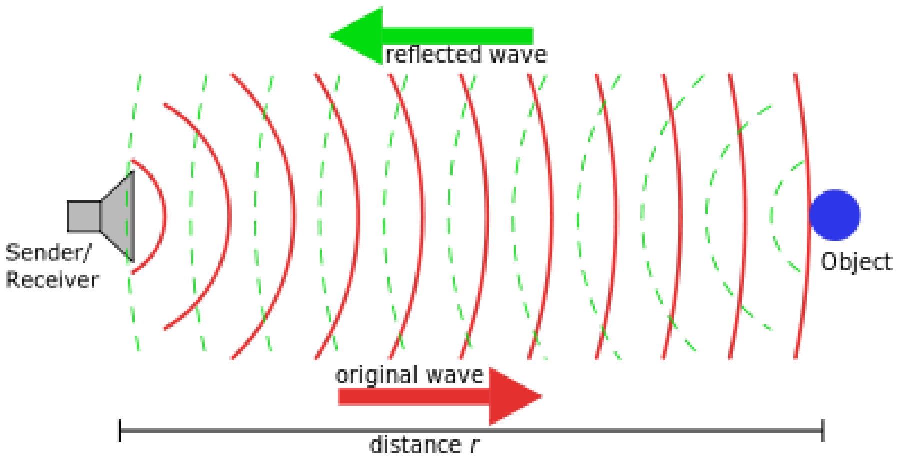
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\neg A)P(\neg A)}$$

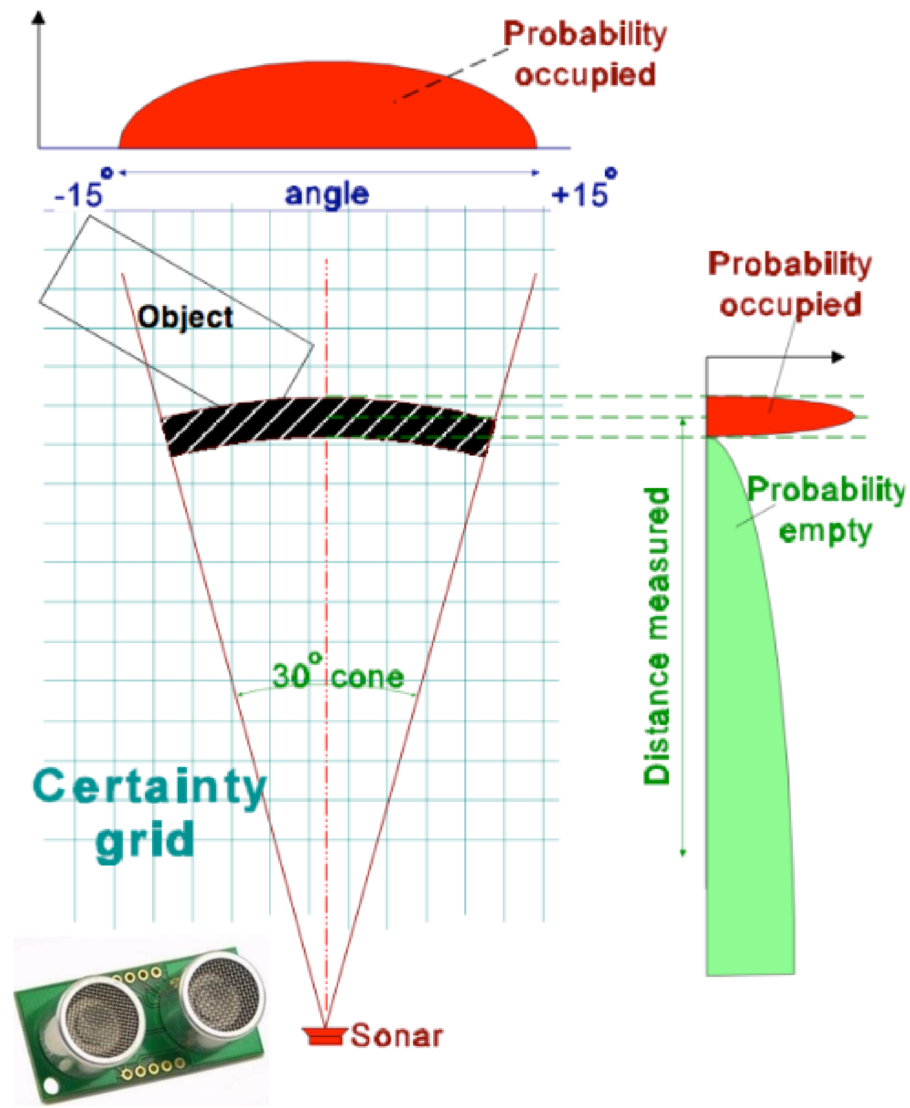
Maps the environment as an array of cells

- Cell sizes range from 5 to 50 cm
- Each cell holds a probability value that the cell is occupied
- Useful for combining different sensor scans, and even different sensor modalities



Sonar Sensor Model (Ultrasound Wave)

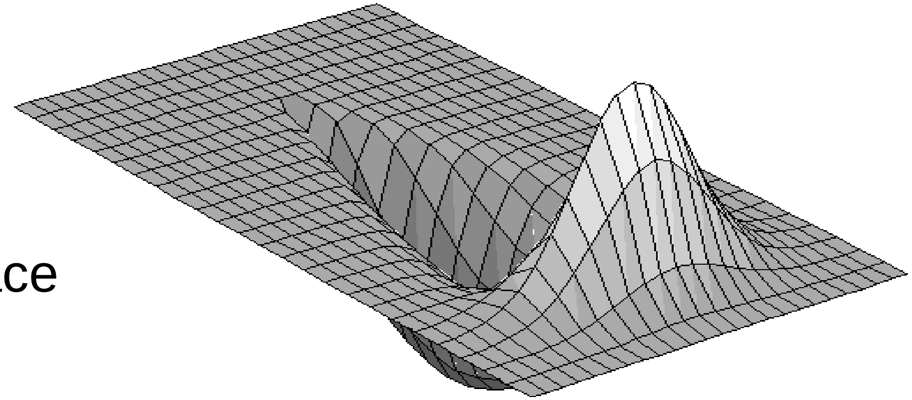




An US wave is sent by a transducer

- Time of flight is measured
- Distance is computed from it
- Obstacle could be anywhere on the arc at distance D
- The space closer than D is likely to be free.

Occupancy from Sonar Return



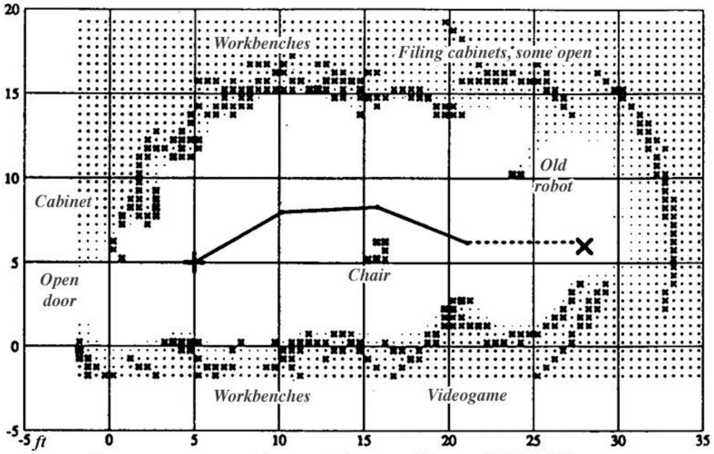
The most simple occupancy model uses

- A 2D Gaussian for information about occupancy
- Another 2D Gaussian for free space

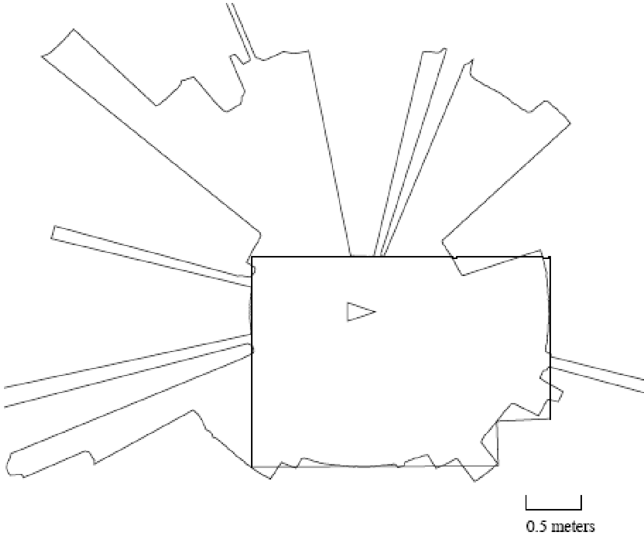
Sonar sensors present several issues

- A wide sonar cone creates noisy maps
- Specular (multi-path) reflections generates unrealistic measurements

Moravec 1984

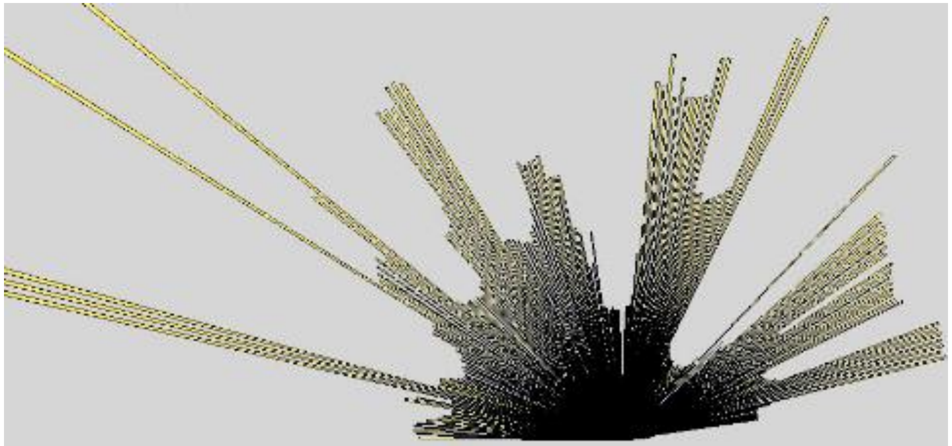


Room traverse by grid map from SONAR



Lasers are definitely more accurate sensors

- 180 ranges over 180°
  (up to 360 in some models)
- 1 to 64 planes scanned
- 10-75 scans/second
- <1cm range resolution
- Max range up to 50-80 m
- Problems only with mirrors, glass, and matte bla



< 1000 €

~ 6000 €

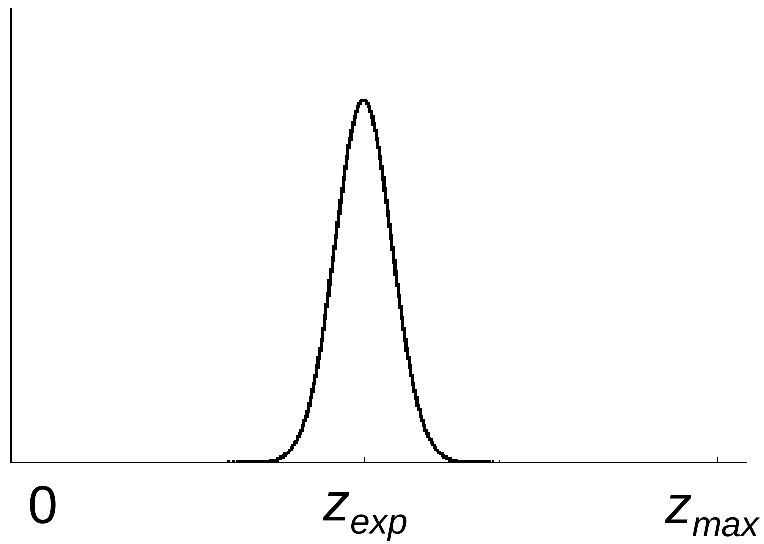
~ 40.000 €

> 80.000 €

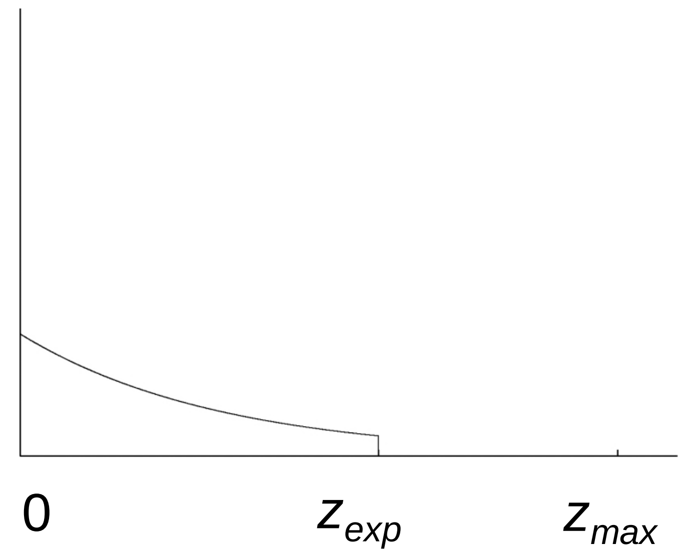
Beam Based Sensor Model (I)

The laser range finder model describe each single measurement using

Measurement noise



Unexpected obstacles

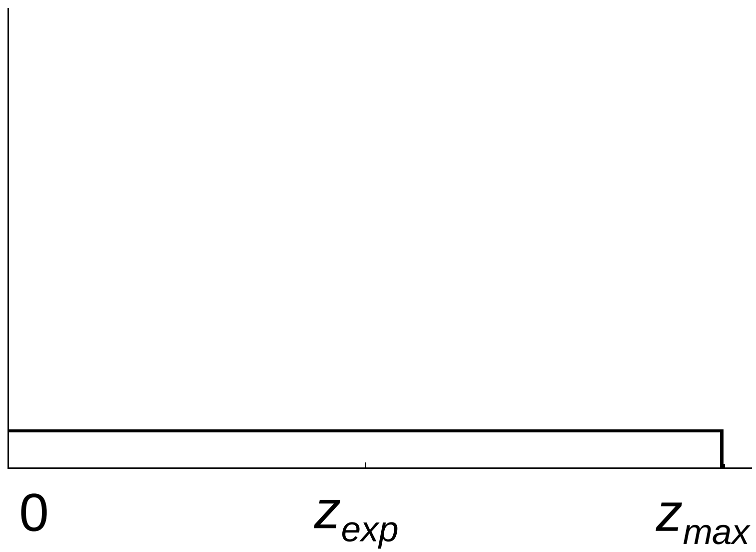


$$P_{hit}(z \mid x, m) = \eta \frac{1}{\sqrt{2\pi b}} e^{-\frac{1}{2}\frac{(z - z_{\exp})^2}{b}}$$

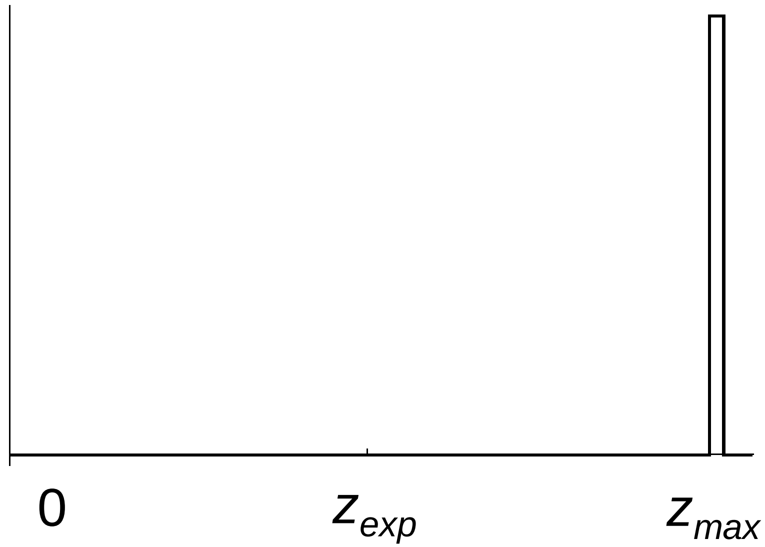
$$P_{\mathrm{unexp}}(z \mid x, m) = \begin{cases} \eta \, \lambda \, \mathrm{e}^{-\lambda z} & z < z_{\mathrm{exp}} \\ 0 & otherwise \end{cases}$$

The laser range finder model describe each single measurement using

Random measurement



Max range

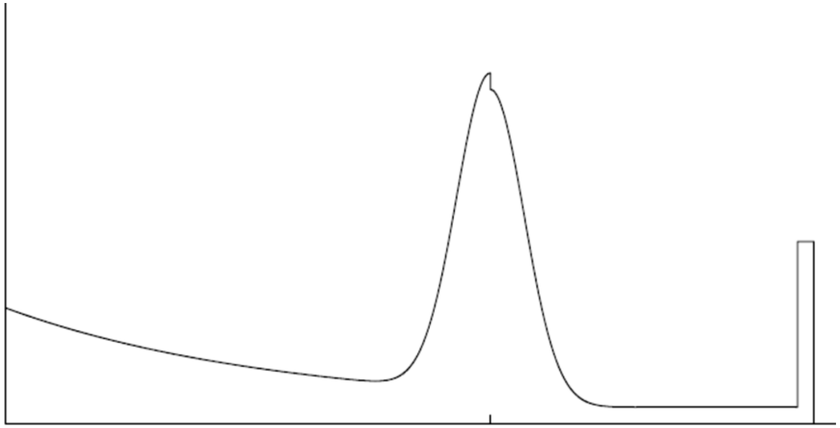


$$P_{rand}(z \mid x, m) = \eta \frac{1}{z_{\max}}$$

$$P_{\max}(z \mid x, m) = \eta \frac{1}{z_{small}}$$

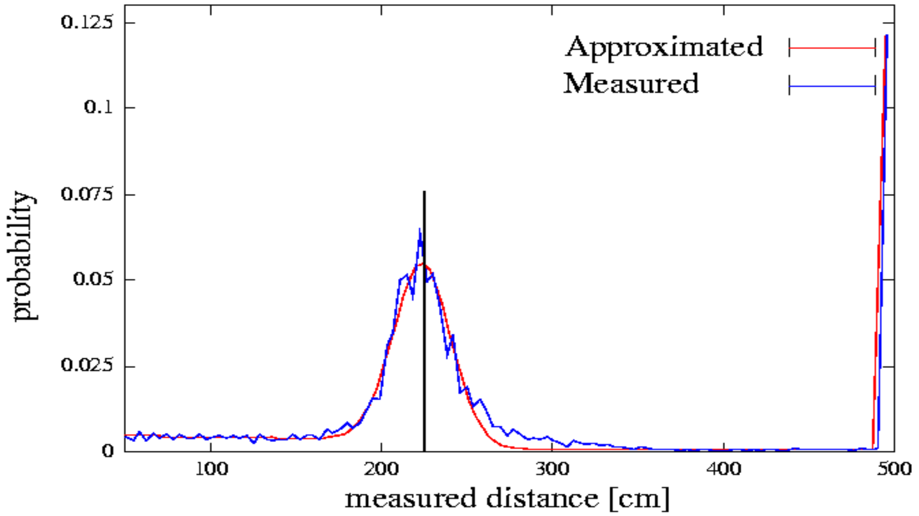
Beam Based Sensor Model (III)

The laser range finder model describe each single measurement using



$$P(z \mid x,m) = \begin{pmatrix} \alpha_{\text{hit}} \\ \alpha_{\text{unexp}} \\ \alpha_{\text{max}} \\ \alpha_{\text{rand}} \end{pmatrix}^T \cdot \begin{pmatrix} P_{\text{hit}}(z \mid x,m) \\ P_{\text{unexp}}(z \mid x,m) \\ P_{\text{max}}(z \mid x,m) \\ P_{\text{rand}}(z \mid x,m) \end{pmatrix}$$

Which, in practice, turns out to be quite accurate!



Matteo Matteucci – matteo.matteucci@polimi.it

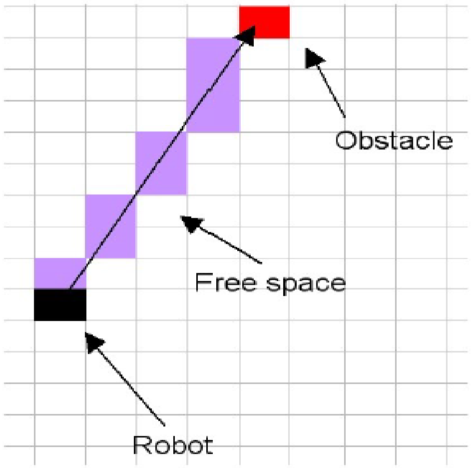
Occupancy Grid Cell Update

Let $occ(i,j)$ mean cell C_{ij} is occupied, then we have

- Probability: $p(occ(i,j))$ has range $[0,1]$
- Odds: $o(occ(i,j))$ has range $[0,+\infty)$

$$o(A) = \frac{P(A)}{P(\neg A)}$$

- Log odds: $\log o(occ(i,j))$ has range $(-\infty,+\infty)$
  - Each cell Cij holds the value $\log o\big(occ(i,j)\big)$
  - $Cij = 0$ corresponds to $p\big(occ(i,j)\big) = 0.5$



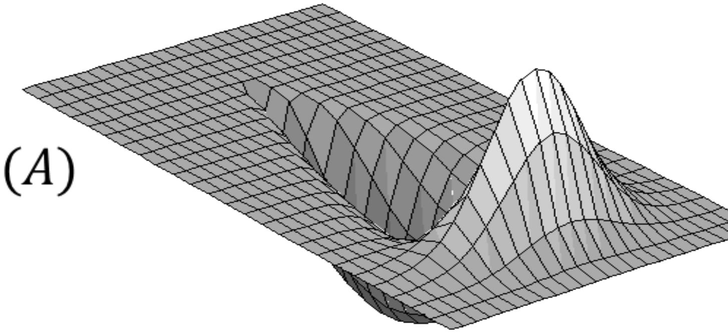
We will apply Bayes Law

- where *A* is *occ*(*i*,*j*)
- and *B* is an observation *r* = *D*

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

We can simplify this by using the log odds representation …

Occupancy by Log Odds



Lets consider Bayes law

- $o(A|B) = \frac{p(A|B)}{P(\neg A|B)} = \frac{p(B|A)P(A)}{P(B|\neg A)P(\neg A)} = \tau(B|A)o(A)$
- $\log o(A|B) = \log \tau(B|A) + \log o(A)$

To update the log odds of a cell at distance D

- $\log o(occ(i,j) \mid r = D) = \log \tau(\mathrm{r = D \mid occ(i,j)}) + \log o\left(occ(i,j)\right)$

Assume cell C_{ij} holds $\log o\left(occ(i,j)\right)$

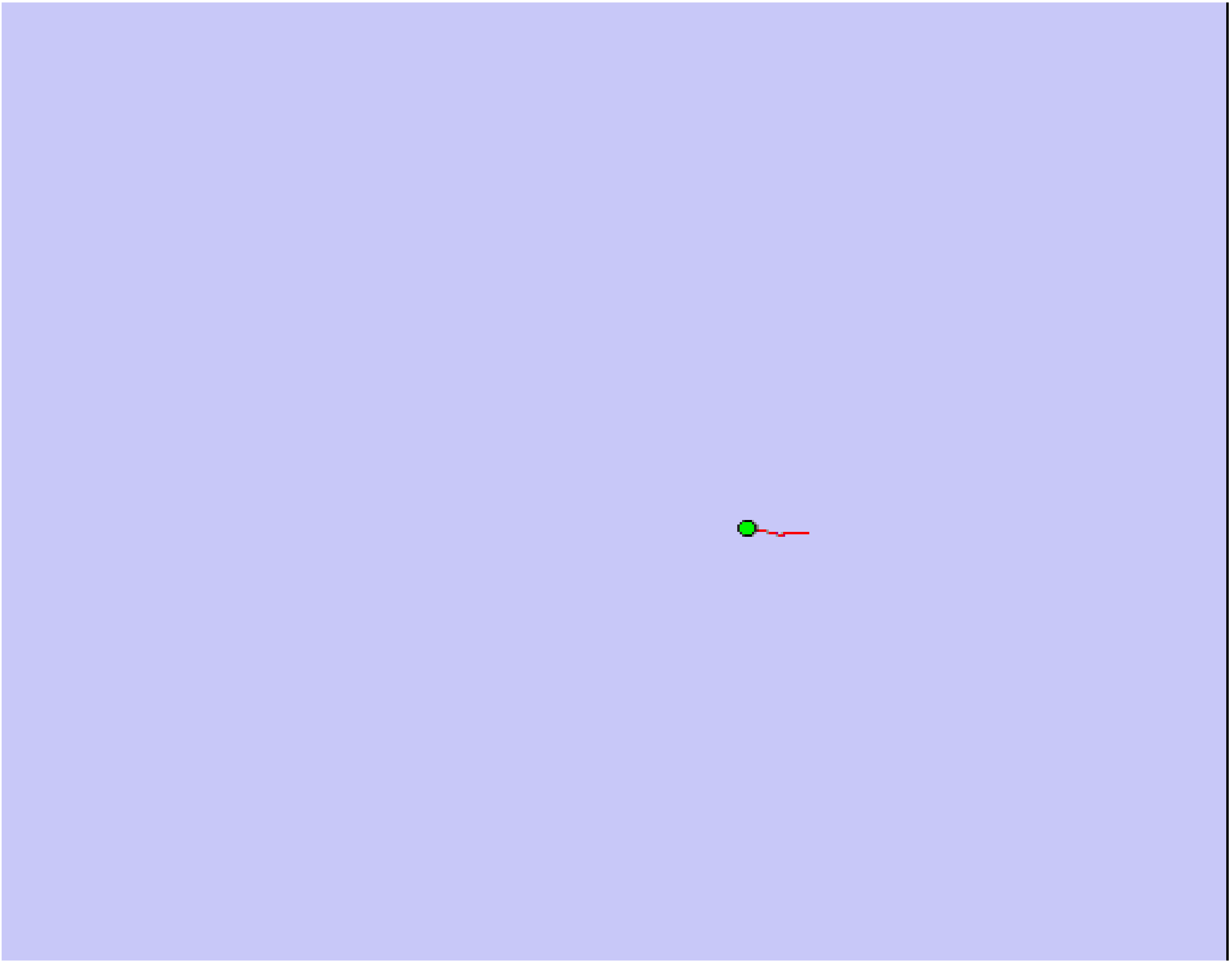
- Let be r the measurement from the sensor
- Let D be the distance of the cell
- For each cell C_{ij} accumulate evidence from each sensor reading

$$\tau(r = D|occ(i,j)) = \frac{p(r = D|occ(i,j))}{p(r = D|\neg occ(i,j))} \approx \frac{.06}{.005} = 12 \quad \rightarrow \quad log_2 \tau = 3.5$$

$$\tau(r > D|occ(i,j)) = \frac{p(r > D|occ(i,j))}{p(r > D|\neg occ(i,j))} \approx \frac{.45}{.90} = .5 \quad \rightarrow \quad log_2 \tau = -1$$

Matteo Matteucci – matteo.matteucci@polimi.it

Mapping with Raw Odometry (with known poses)



Scan Matching

Maximize the likelihood of the *i-th* pose and map relative to the *(i-1)-th* pose and map.

$$\hat{x}_t = \arg\max_{x_t} \left\{ p(z_t \mid x_t, \hat{m}^{[t-1]}) \cdot p(x_t \mid u_{t-1}, \hat{x}_{t-1}) \right\}$$

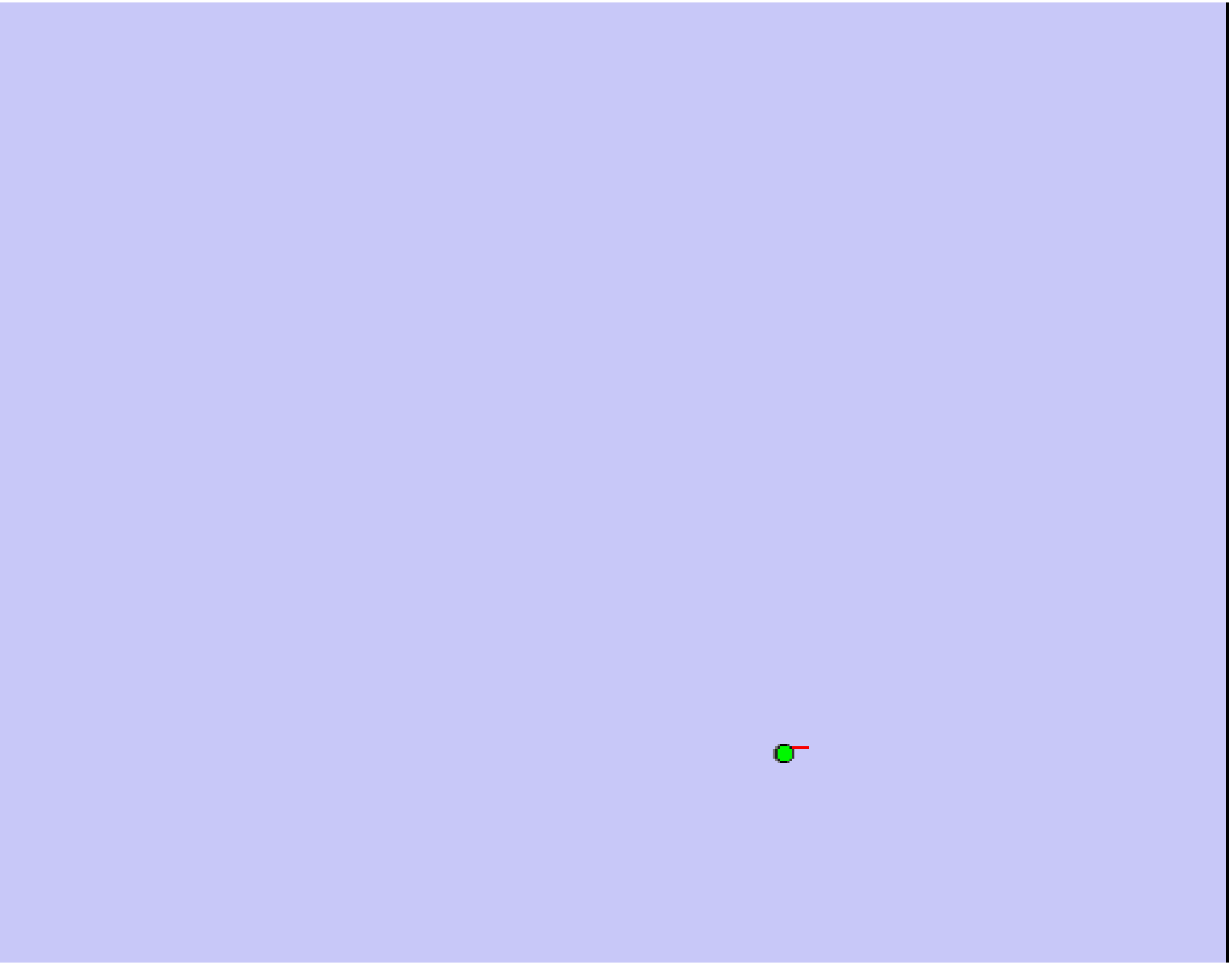
current measurement

robot motion

map constructed so far

Calculate the map $\hat{m}^{[t]}$ according to “mapping with known poses” based on the poses and observations.

Scan Matching Example



Techniques for Generating Consistent Maps

Several techniques have been studied to obtain a consistent estimate of the joint probability of pose and map

- Scan matching
- EKF SLAM / UKF SLAM
- Fast-SLAM (Particle filter based)
- Probabilistic mapping with a single map and a posterior about poses (Mapping + Localization)
- Graph-SLAM, SEIFs
- ...

We won't see the all of them! ☺

SLAM: Simultaneous Localization and Mapping

Full SLAM:

$p(x \; m \mid z \; u \;)$

Two famous example of this!

Extended Kalman Filter (EKF) SLAM

- Solves online SLAM problem
- Uses a linearized Gaussian probability distribution model

FastSLAM

- Solves full SLAM problem
- Uses a sampled particle filter distribution model

Online Sl

dx_{t-1}

Integrations typically done one at a time

Bayes Filters: Framework

Given:

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

$$d_t = \{u_1, z_1 \ldots, u_t, z_t\}$$

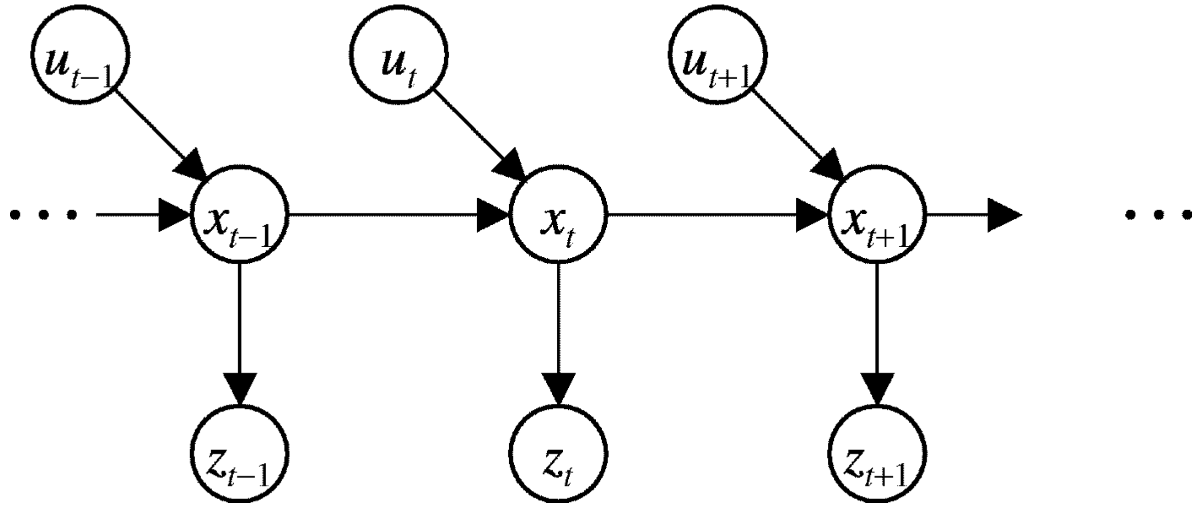
- Sensor model *P(z|x).*
- Action model *P(x|u,x')*.
- Prior probability of the system state *P(x).*

We want to compute:

- Estimate of the state *X* of a dynamical system.
- The posterior of the state is also called **Belief**:

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

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

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

Bayes
$$= \eta\ P(z_t \mid x_t, u_1, z_1, \ldots, u_t)\ P(x_t \mid u_1, z_1, \ldots, u_t)$$

Markov
$$= \eta\ P(z_t \mid x_t)\ P(x_t \mid u_1, z_1, \ldots, u_t)$$

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

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

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

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

Bayes Filter: The Algorithm

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

How to represent such belief?

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)*

Map with N landmarks:(3+2N)-dimensional Gaussian

$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

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

$$\boxed{p(x_t \mid u_t, x_{t-1}) = N\left(x_t; A_t x_{t-1} + B_t u_t, R_t\right)}$$

Map with N landmarks:(3+2N)-dimensional Gaussian

$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

$$\boxed{z_t = C_t x_t + \delta_t}$$

$$\boxed{p(z_t \mid x_t) = N\left(z_t ; C_t x_t , Q_t\right)}$$

Bayes Filter: The Algorithm

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

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)*

correction

prediction

Kalman Filter Algorithm

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

Prediction:

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

Correction:

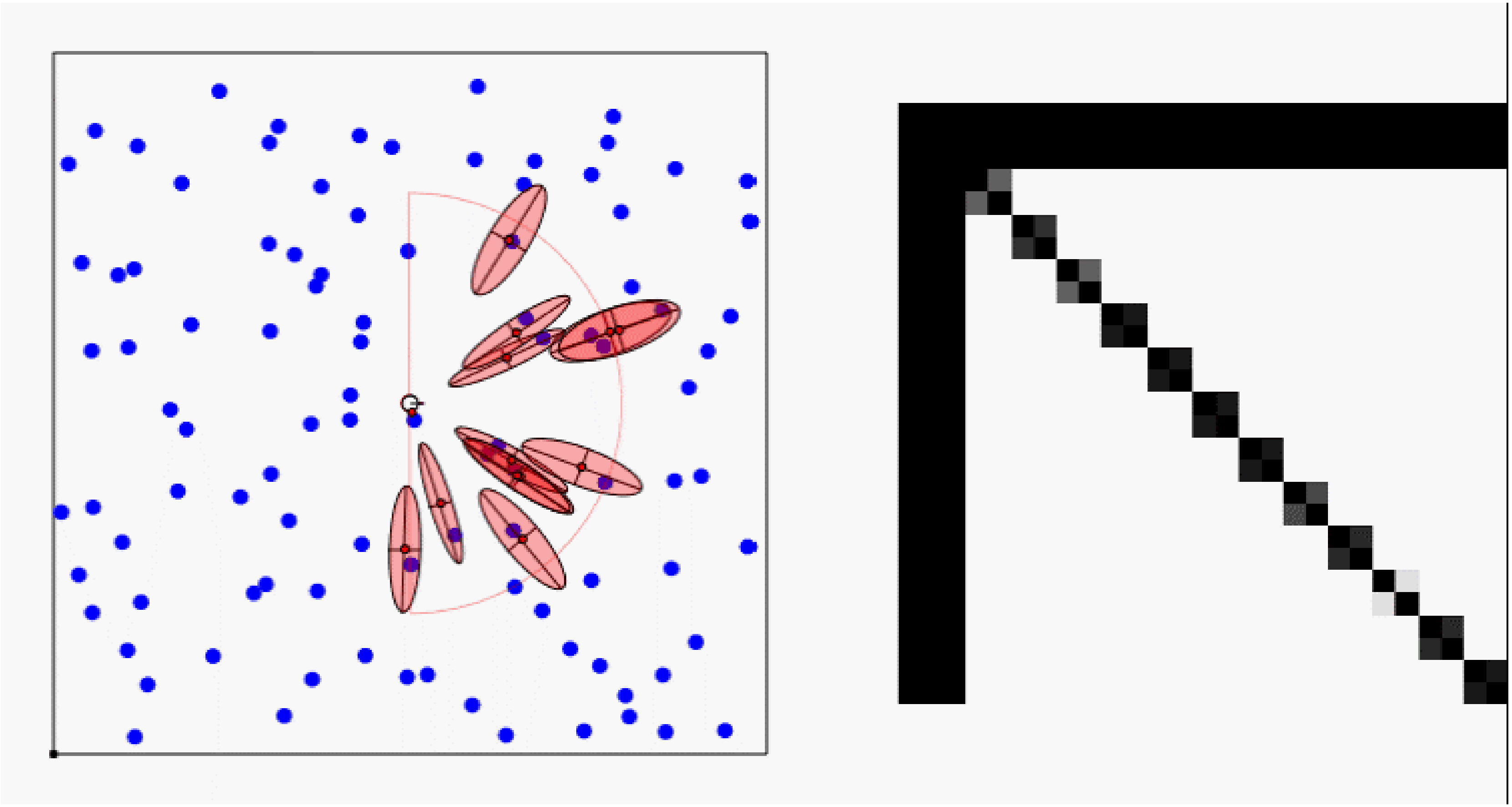
$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$
$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$$
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

Return μ_t, Σ_t

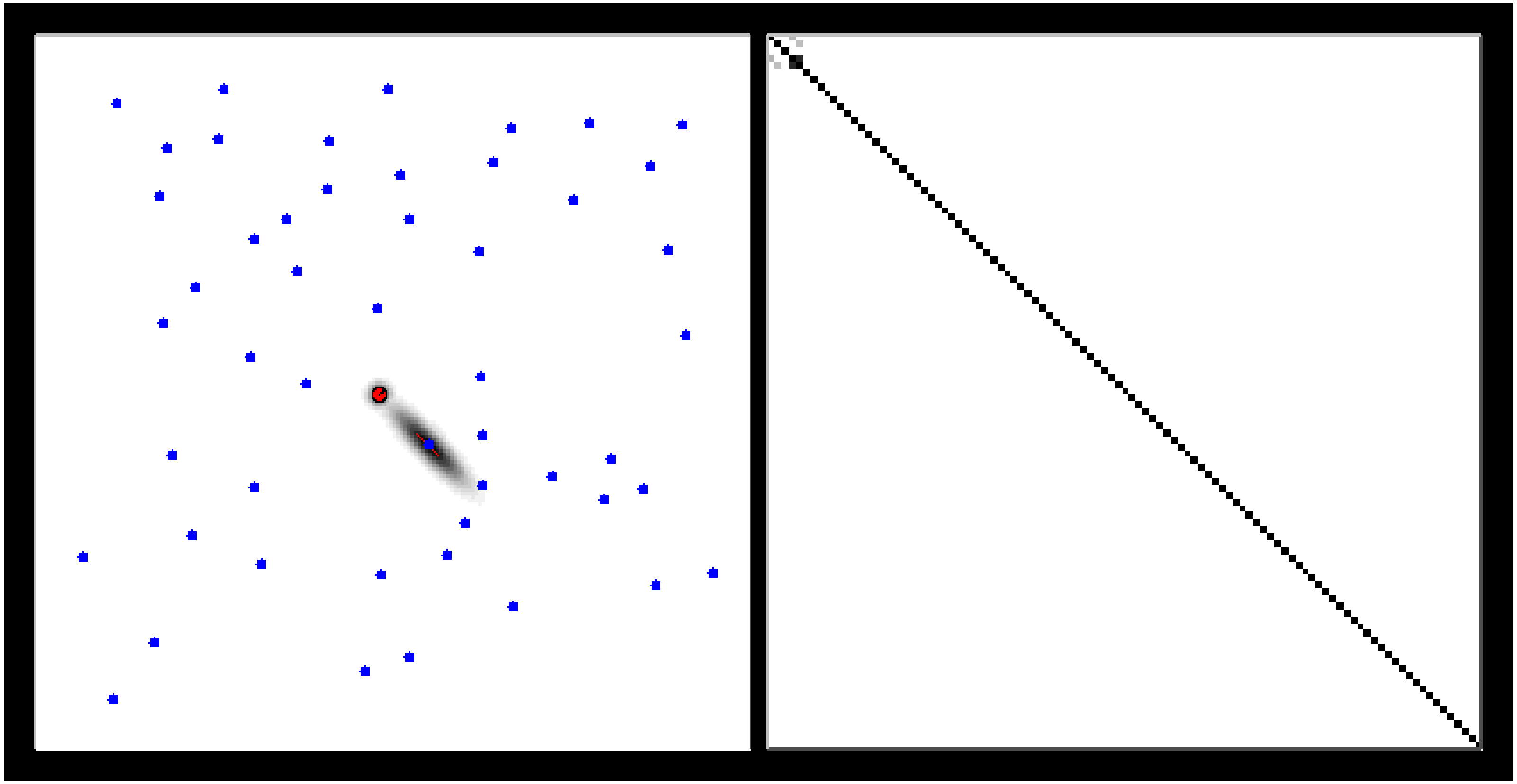
$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

Classical Solution – The EKF

Approximate the SLAM posterior with a high-dimensional Gaussian

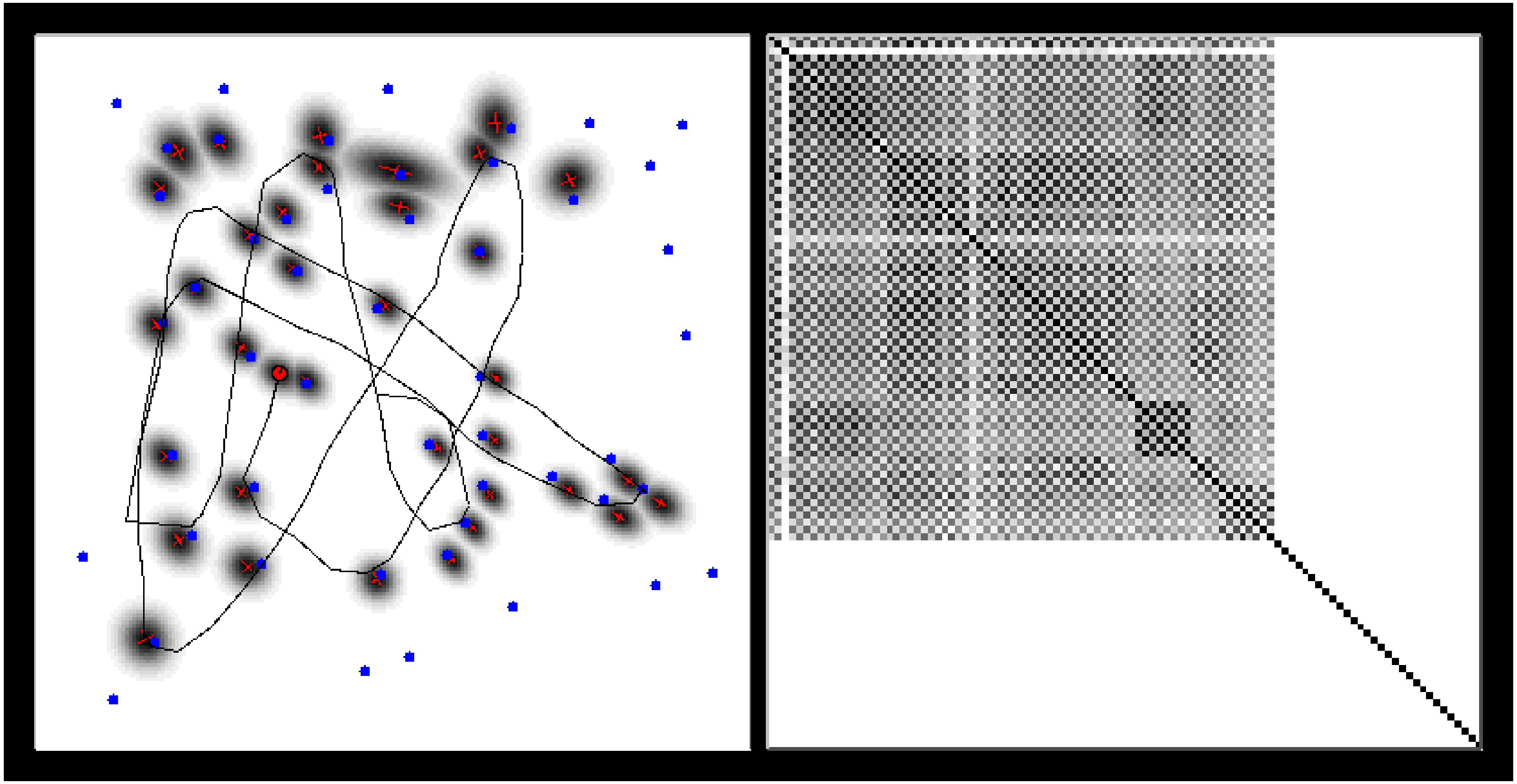


Blue path = true path **Red path** = estimated path **Black path** = odometry



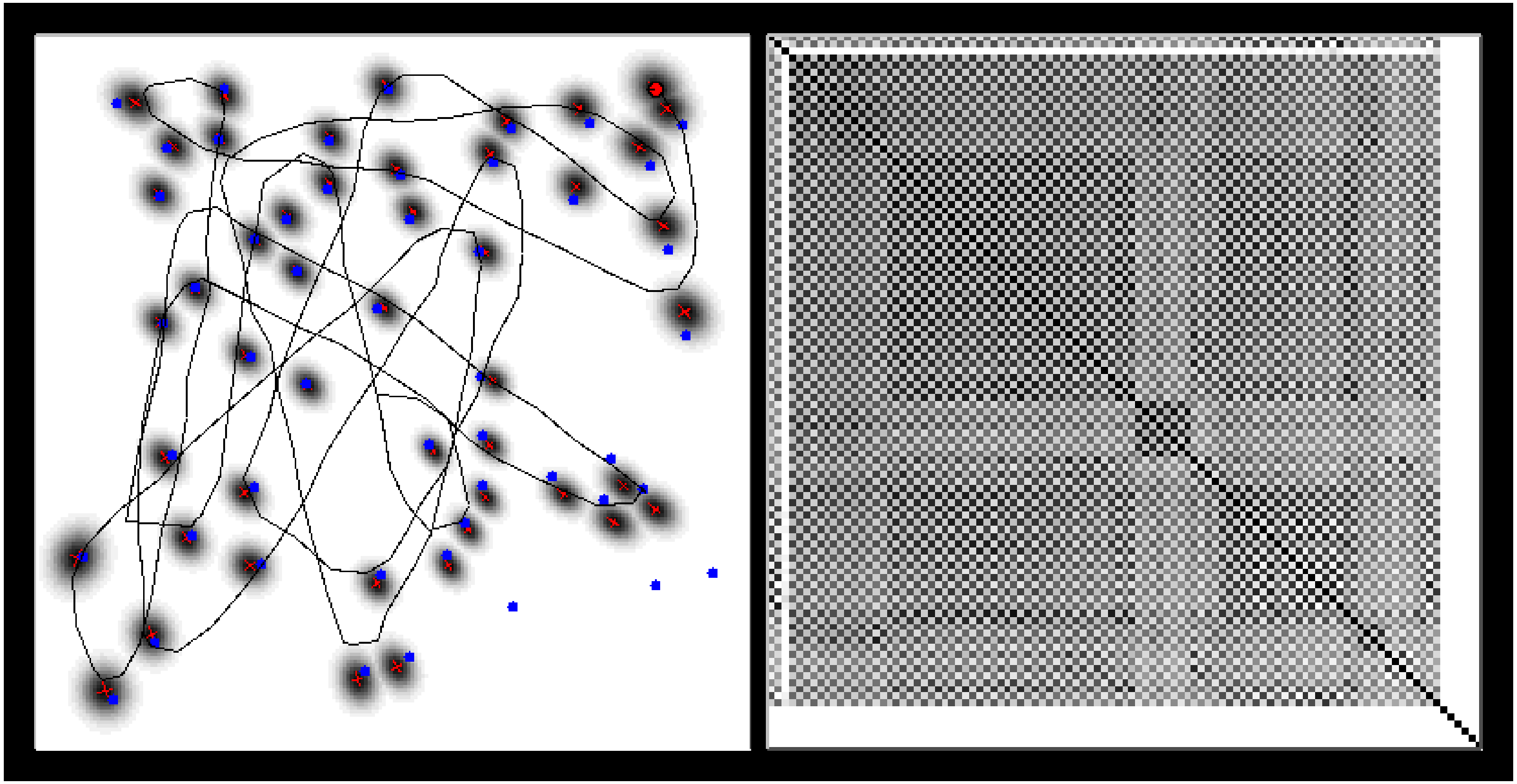
Map Correlation matrix

EKF-SLAM



Map Correlation matrix

EKF-SLAM



Map Correlation matrix

Properties of KF-SLAM (Linear Case)

Theorem:

[Dissanayake et al., 2001]

The determinant of any sub-matrix of the map covariance matrix decreases monotonically as successive observations are made.

Theorem:

In the limit the landmark estimates become fully correlated

Are we happy about this?

- Quadratic in the number of landmarks: $O(n^2)$
- Convergence results for the linear case.
- Can diverge if nonlinearities are large!
- Have been applied successfully in large-scale environments.
- Approximations reduce the computational complexity.

EKF Drawbacks

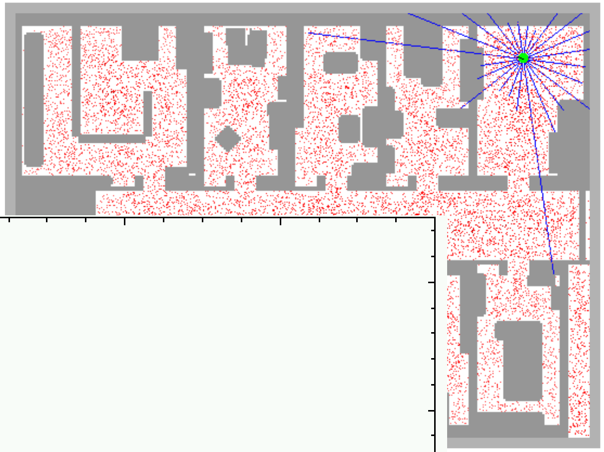
EKF-SLAM works pretty well but ...

- EKF-SLAM employs linearized models of nonlinear motion and observation models and so inherits many caveats.
- Computational effort is demand because computation grows quadratically with the number of landmarks.

Possible solutions

- Local submaps [Leonard & al 99, Bosse & al 02, Newman & al 03]
- Sparse links (correlations) [Lu & Milios 97, Guivant & Nebot 01]
- Sparse extended information filters [Frese et al. 01, Thrun et al. 02]
- Thin junction tree filters [Paskin 03]
- Rao-Blackwellisation (FastSLAM) [Murphy 99, Montemerlo et al. 02, Eliazar et al. 03, Haehnel et al. 03]
  - Represents nonlinear process and non-Gaussian uncertainty
  - Rao-Blackwellized method reduces computation

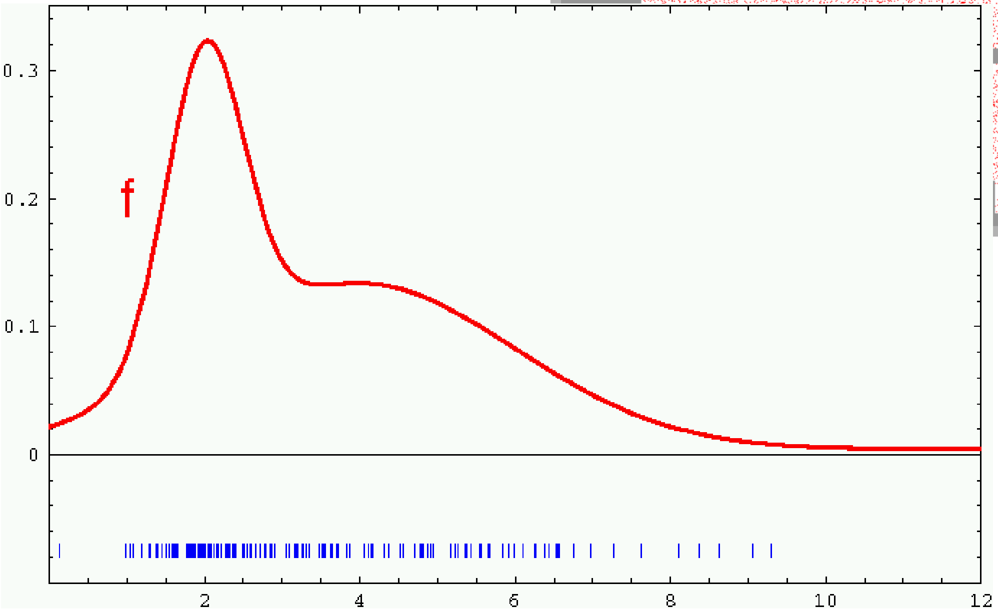
Particle Filters



Represent belief by random samples

Estimation of non-Gaussian, nonlinear processes

- Monte Carlo filter
- Survival of the fittest
- Condensation
- Bootstrap filter
- Particle filter
- …



Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96]

Computer vision: [Isard and Blake 96, 98]

Dynamic Bayesian Networks: [Kanazawa et al., 95]

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 $Bel(x_{t-1})$

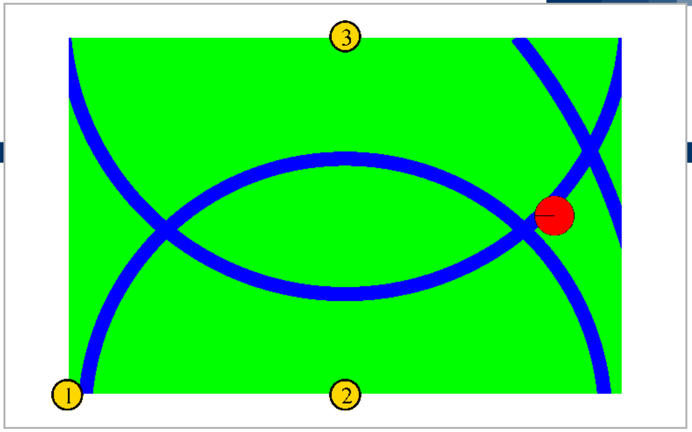
draw x^i_t from $p(x_t \mid x^i_{t-1}, u_{t-1})$

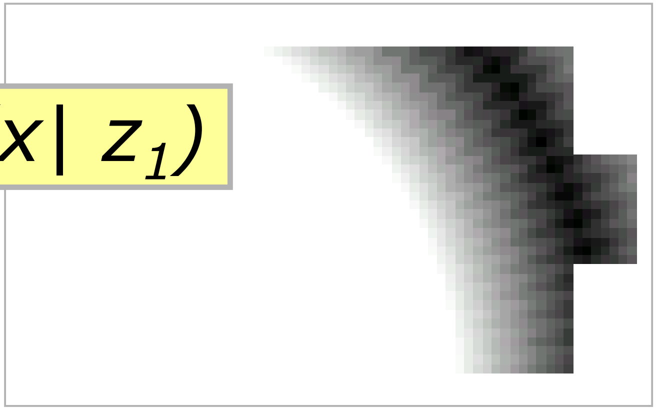
Importance factor for x^i_t:

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

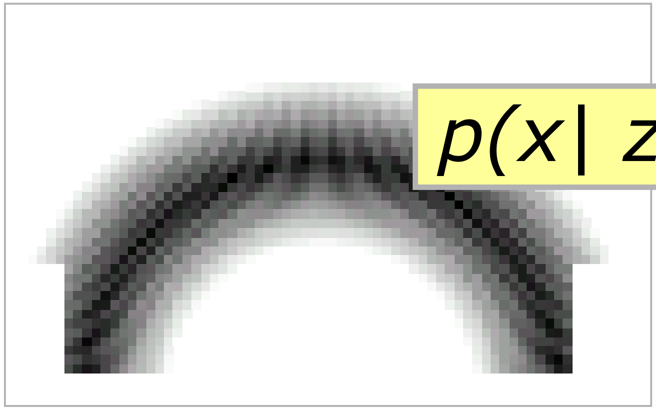
A Four Legged Example …

Involved Distributions

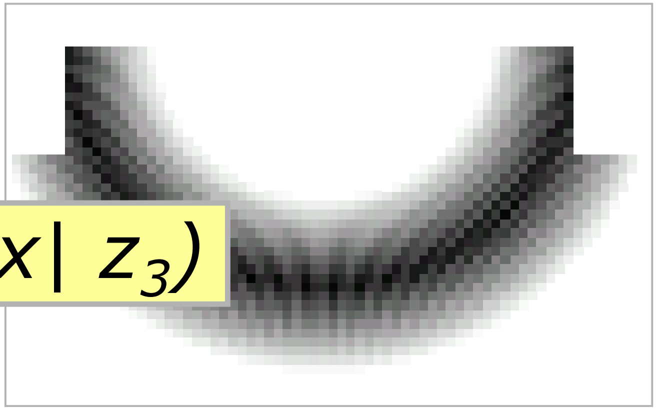




$p(x|\, z_1)$

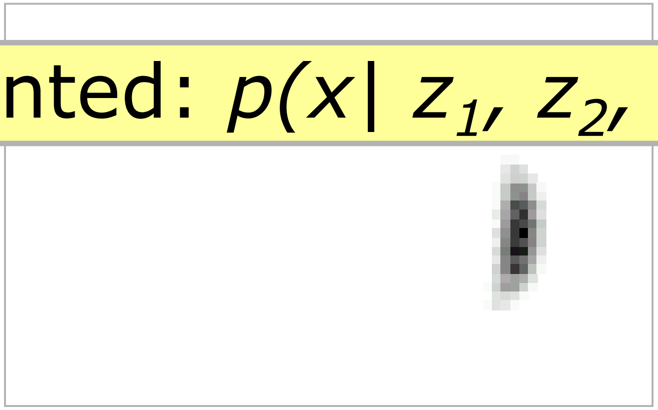


$p(x|\, z_2)$



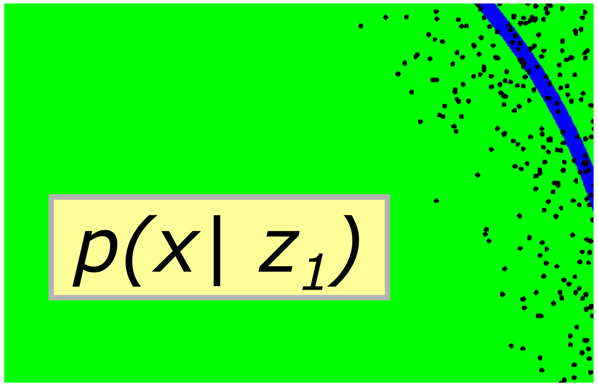
$p(x|\, z_3)$

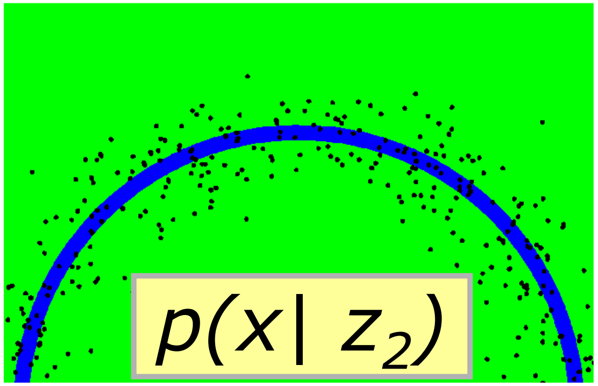
Wanted: $p(x|\, z_1, z_2, z_3)$

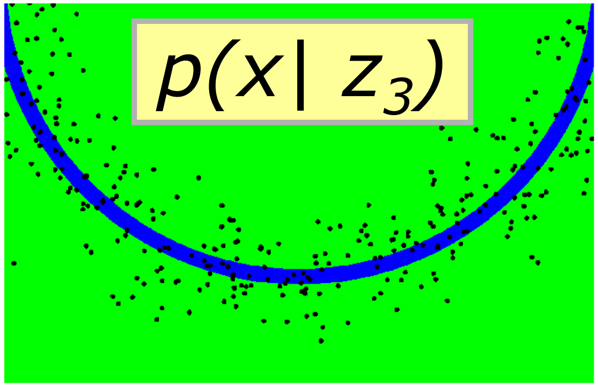


This is (somehow) easy!

Draw samples from $p(x|z_i)$ using the detection parameters and some noise





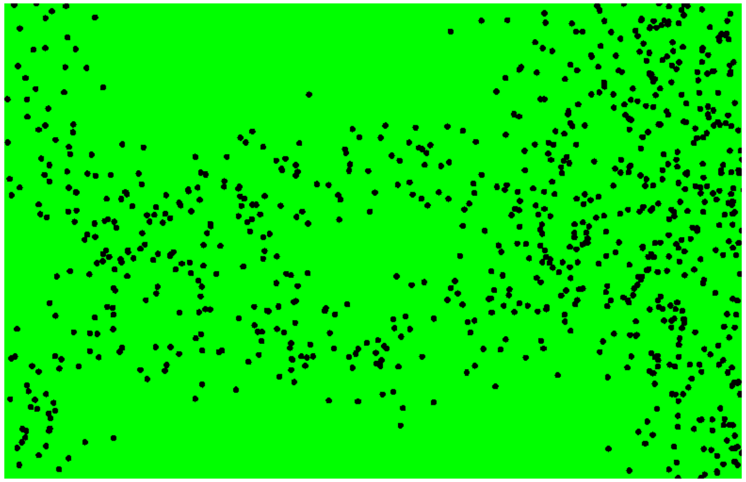


$$\text{Target distribution f}: p(x \mid z_1, z_2, \ldots, z_n) = \frac{\prod_k p(z_k \mid x) \quad p(x)}{p(z_1, z_2, \ldots, z_n)}$$

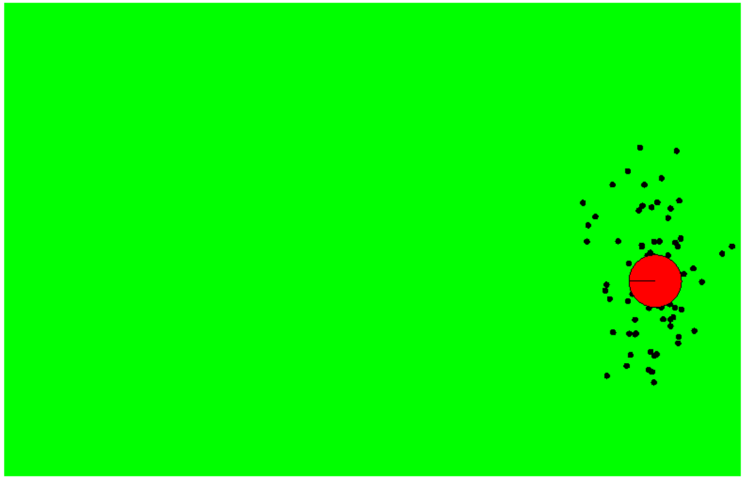
$$\text{Sampling distribution g}: p(x \mid z_l) = \frac{p(z_l \mid x)\, p(x)}{p(z_l)}$$

$$\text{Importance weights w}: \frac{f}{g} = \frac{p(x \mid z_1, z_2, \ldots, z_n)}{p(x \mid z_l)} = \frac{p(z_l) \prod_{k \neq l} p(z_k \mid x)}{p(z_1, z_2, \ldots, z_n)}$$

Importance Sampling with Resampling

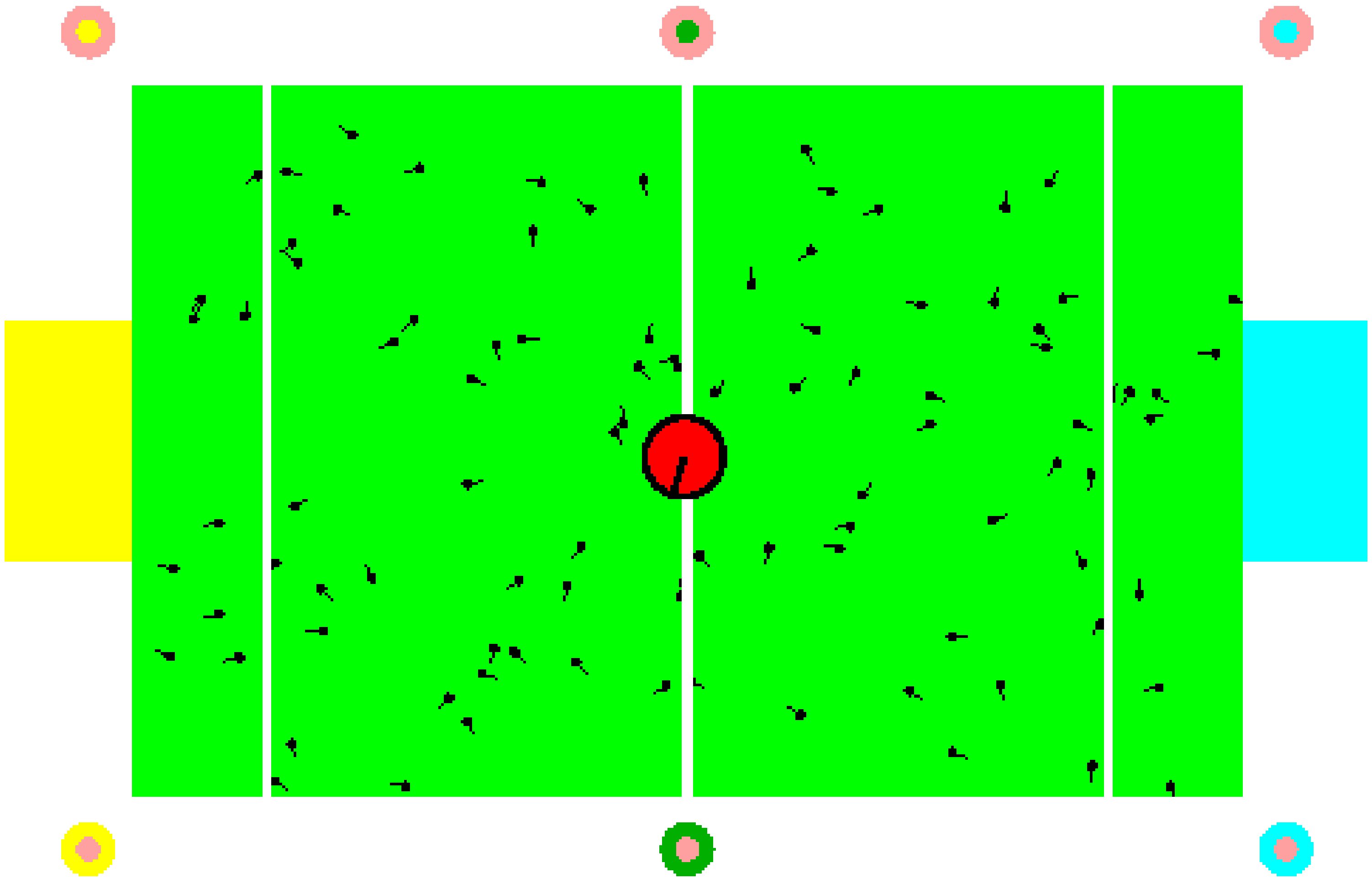


Weighted samples



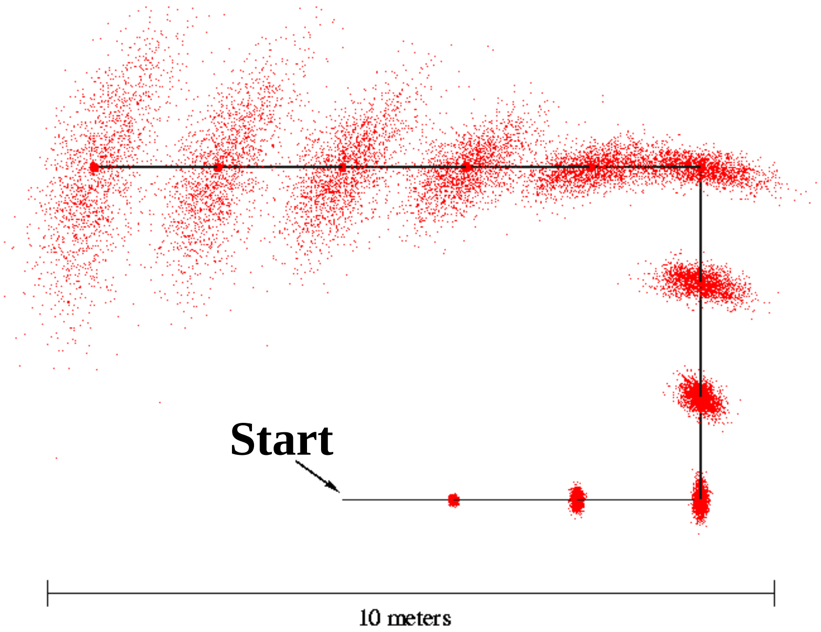
After resampling

Localization for AIBO robots

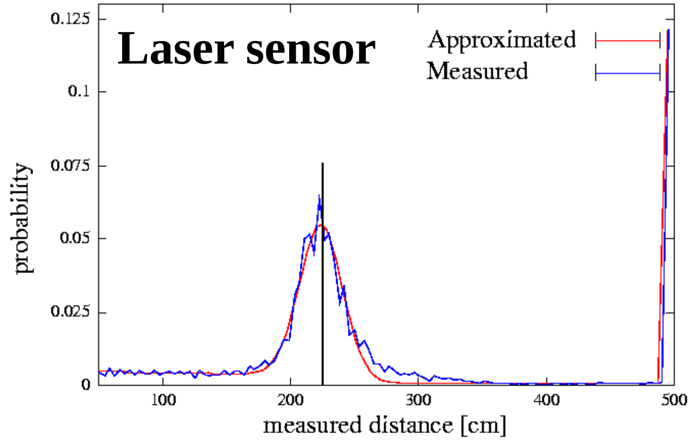


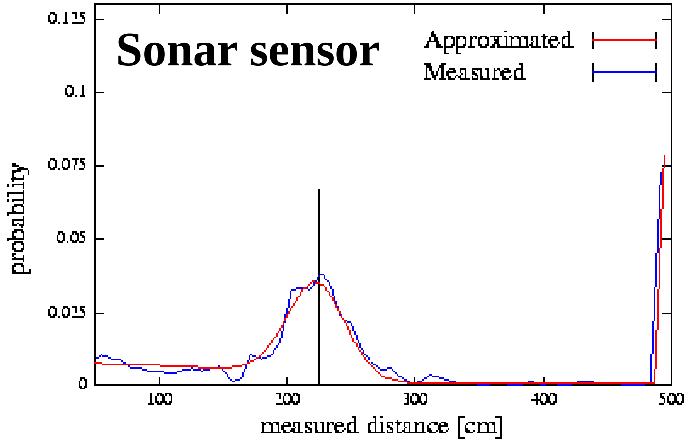
Monte Carlo Localization with Laser

Stochastic motion model

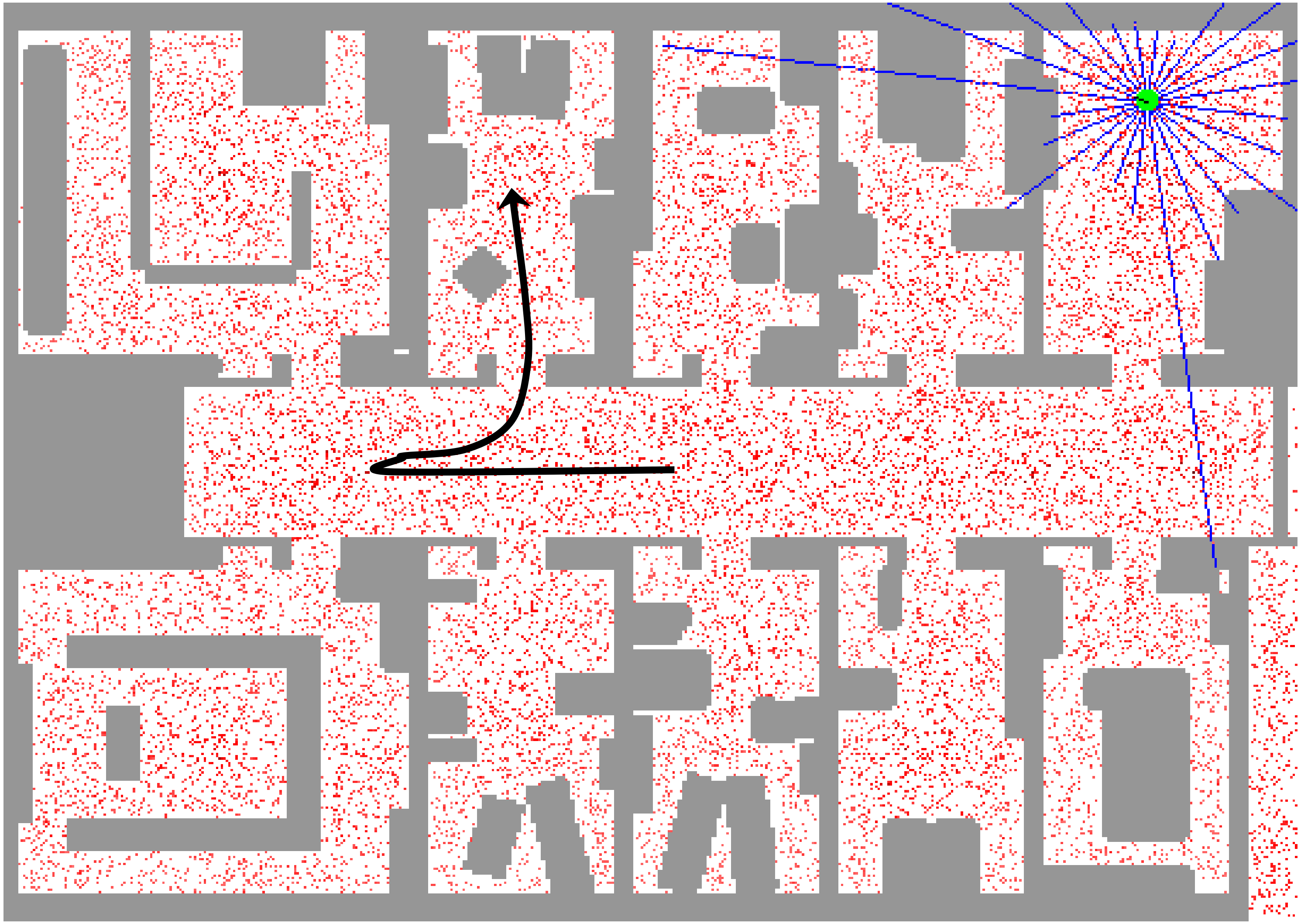


Proximity sensor model





Sample-based Localization (sonar)



The FastSLAM Idea (Full SLAM)

In the general case we have

$$p(x_t, m \mid z_t) \neq P(x_t \mid z_t) P(m \mid z_t)$$

However if we consider the full trajectory X_t rather than the single pose $x_{t.}$

$$p(X_t, m \mid z_t) = P(X_t \mid z_t) P(m \mid X_t, z_t)$$

In FastSLAM, the trajectory X_t is represented by particles $X_t(i)$ while the map is represented by a factorization called Rao-Blackwellized Filter

$$P(m \mid X_t^{(i)}, z_t) = \prod_j^M P(m_j \mid X_t^{(i)}, z_t)$$

- $P(X_t \mid z_t)$ through particles
- $P(m \mid X_t, z_t)$ using an EKF

FastSLAM Formulation

Decouple map of features from pose ...

- Each particle represents a robot trajectory
- Feature measurements are correlated thought the robot trajectory
- If the robot trajectory is known all of the features would be uncorrelated
- Treat each pose particle as if it is the true trajectory, processing all of the feature measurements independently

poses map observations & movements

$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) =$$

$$\underbrace{p(x_{1:t} \mid z_{1:t}, u_{0:t-1})}_{\text{Robot path posterior}} \cdot \underbrace{p(l_{1:m} \mid x_{1:t}, z_{1:t})}_{\text{Landmark positions}}$$

SLAM posterior

Factored Posterior: Rao-Blackwellization

$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1})$$
$$= \underbrace{p(x_{1:t} \mid z_{1:t}, u_{0:t-1})}_{\text{Robot path posterior (localization problem)}} \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t})$$
$$= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} \underbrace{p(l_i \mid x_{1:t}, z_{1:t})}_{\text{Conditionally independent landmark positions}}$$

Robot path posterior (localization problem)

Conditionally independent landmark positions

Dimension of state space is drastically reduced by factorization making particle filtering possible

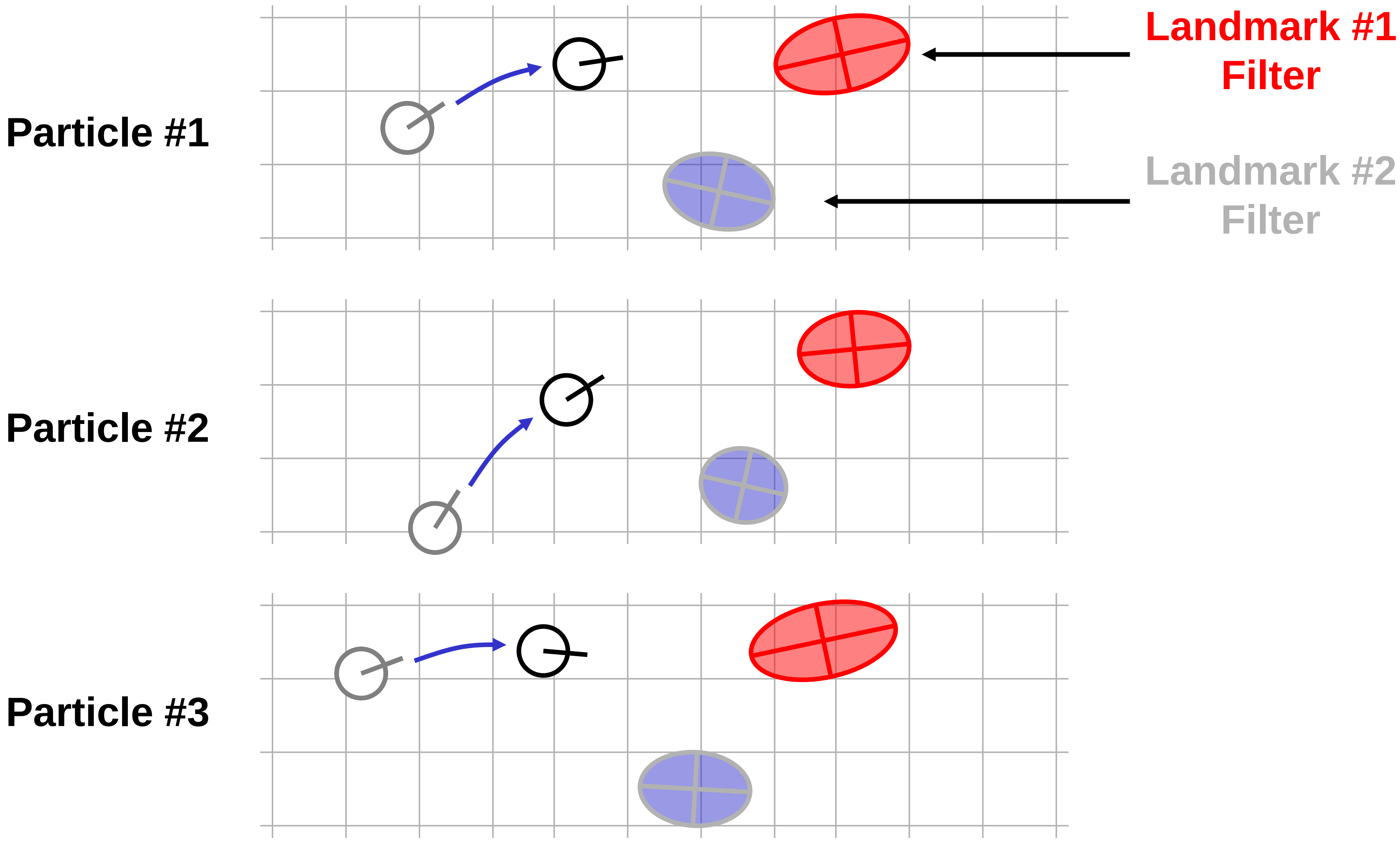
$$p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) = p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} p(l_i \mid x_{1:t}, z_{1:t})$$

FastSLAM in Practice

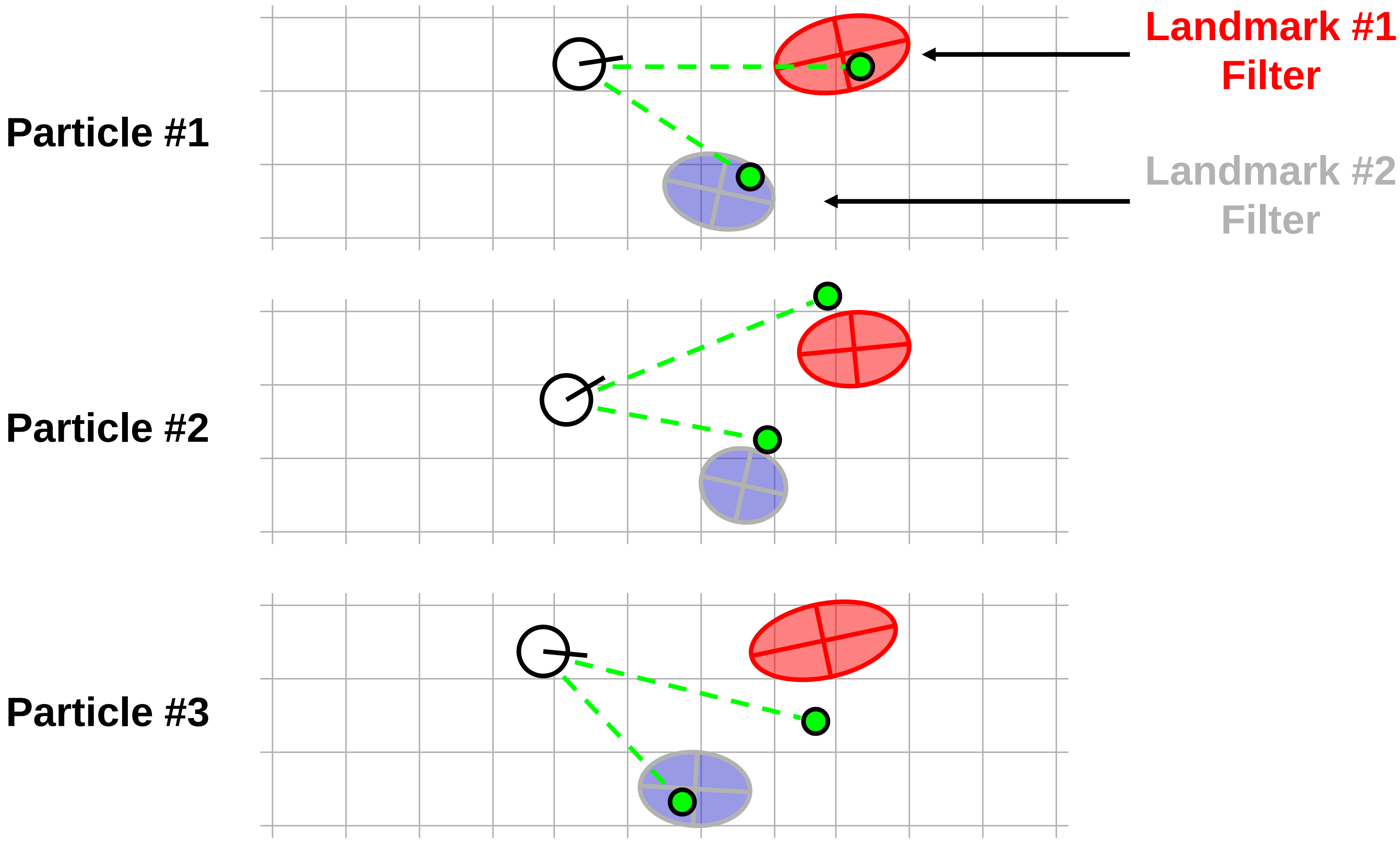
- Rao-Blackwellized particle filtering based on landmarks [Montemerlo et al., 2002]
- Each landmark is represented by a 2x2 Extended Kalman Filter (EKF)
- Each particle therefore has to maintain M EKFs

Particle #1	x, y, θ	**Landmark 1**	**Landmark 2**	**…**	**Landmark M**
Particle #2	x, y, θ	**Landmark 1**	**Landmark 2**	**…**	**Landmark M**
⋮					
Particle N	x, y, θ	**Landmark 1**	**Landmark 2**	**…**	**Landmark M**

FastSLAM – Action Update

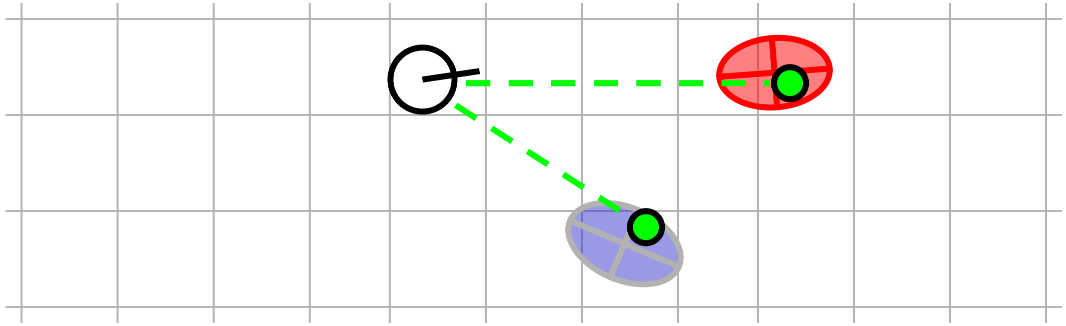


FastSLAM – Sensor Update



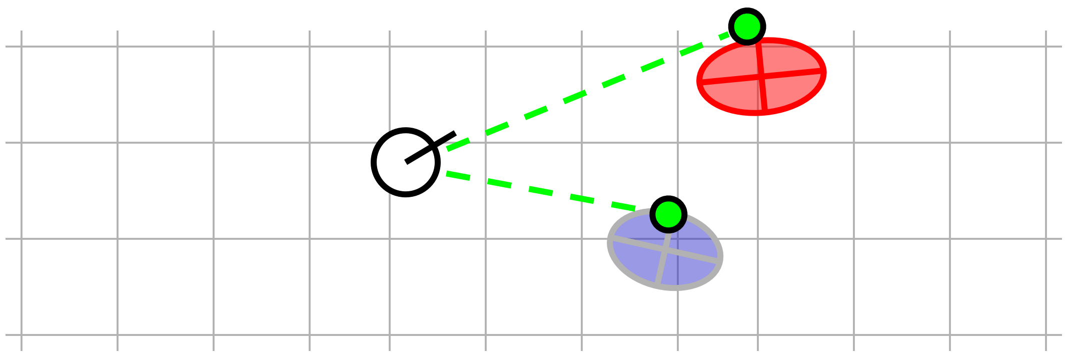
FastSLAM – Sensor Update

Particle #1



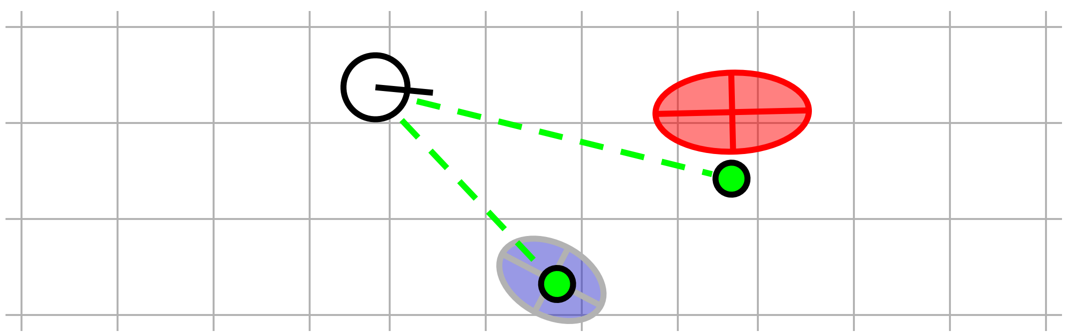
Weight = 0.8

Particle #2



Weight = 0.4

Particle #3



Weight = 0.1

FastSLAM Complexity

Update robot particles based on control u_{t-1}

$O(N)$

Constant time per particle

Incorporate observation z_t into Kalman filters

$O(N \cdot \log(M))$

Log time per particle

Resample particle set

$O(N \cdot \log(M))$

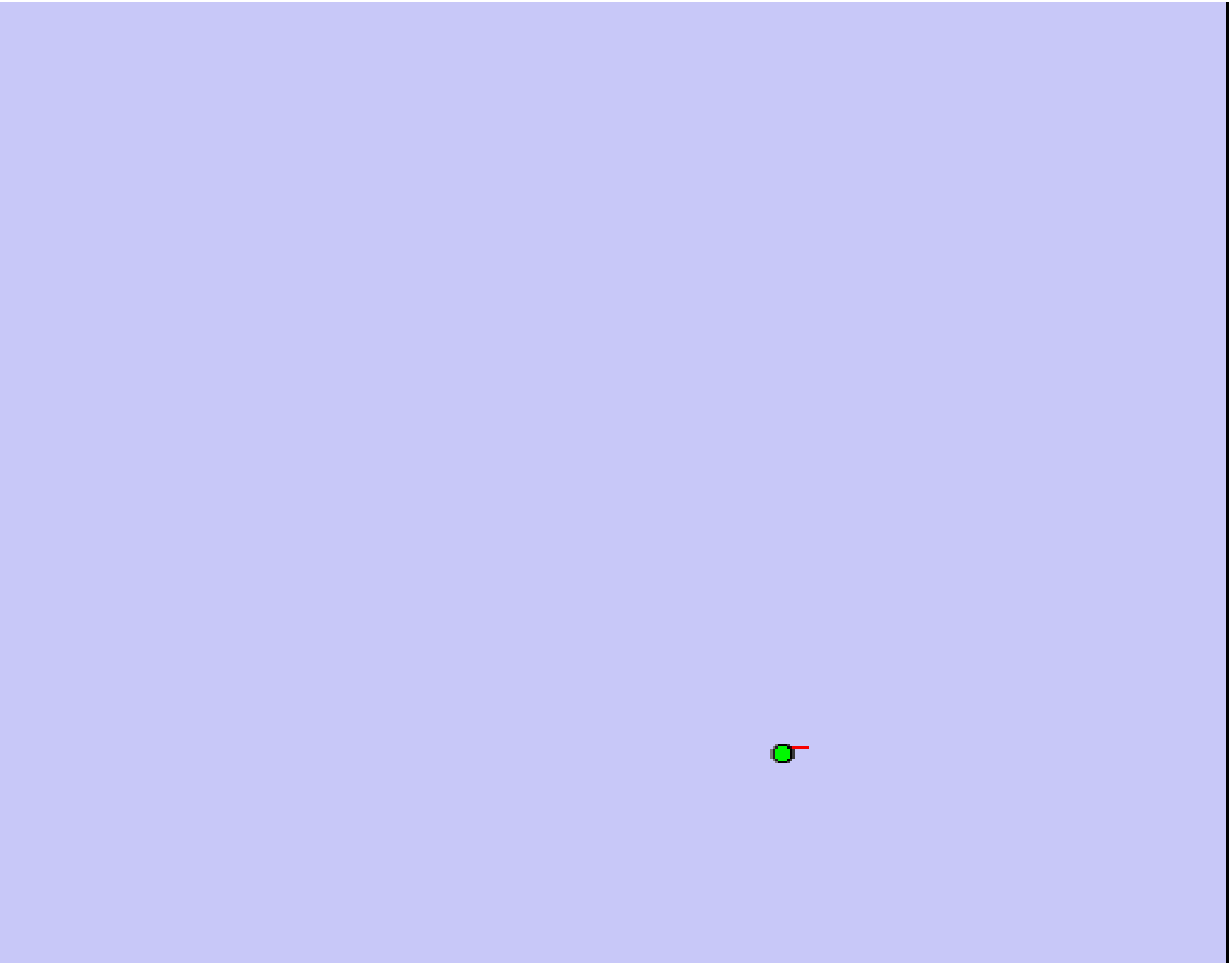
Log time per particle

$O(N \cdot \log(M))$

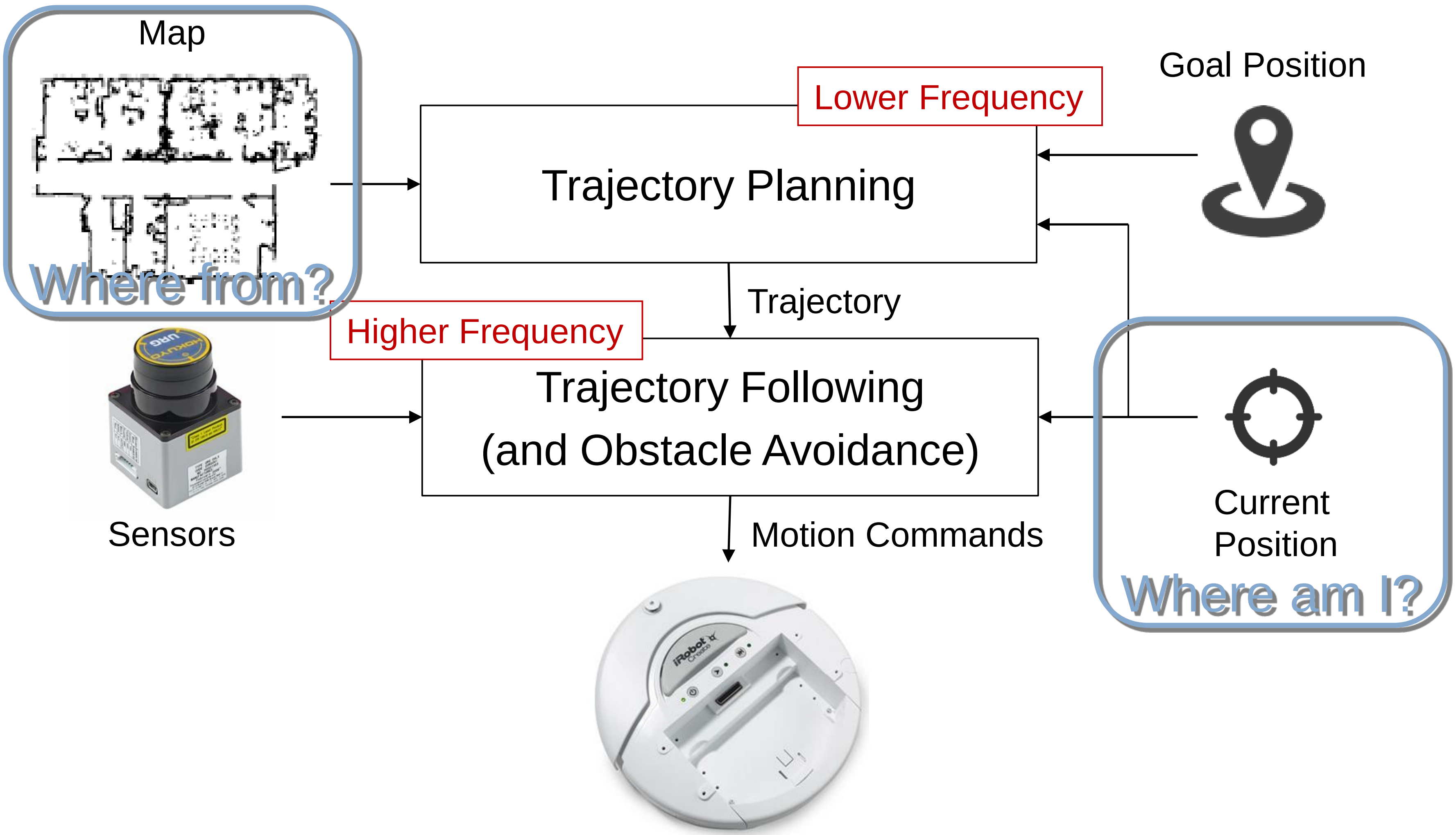
Log time per particle

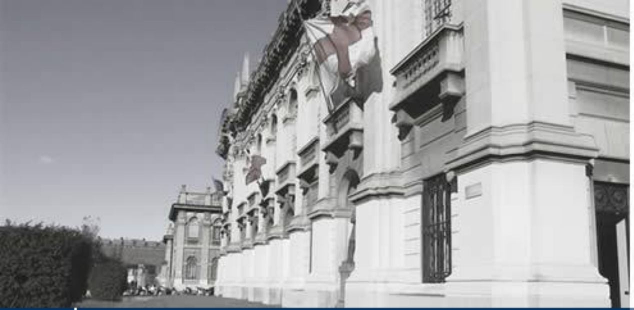
N = Number of particles
M = Number of map features

Fast-SLAM Example



A Two Layered Approach





POLITECNICO DI MILANO

Cognitive Robotics –SLAM with Lasers

Matteo Matteucci – matteo.matteucci@polimi.it