Autonomous Robot Navigation

Localization and Mapping Techniques for Mobile Robots

Panos Trahanias
Professor of Computer Science, University of Crete
and
Head, Computational Vision and Robotics Lab
Foundation for Research and Technology – Hellas (FORTH)
trahania@ics.forth.gr

July 2006

Robots Intelligently Interacting With People
2006 ONASSIS LECTURE SERIES in COMPUTER SCIENCE

Autonomous Navigation
Problem Statement

The ability of robots to navigate safely and reliably within their environments

- Operation in industrial environments
- Tour-guiding visitors in museums/exhibition sites
- Helping in household tasks
- Exploring unfriendly environments
  (volcanoes, sewer systems, underwater)
- Space applications
- …. (the list goes on)

Historical Walkthrough

… in the beginning: Robotics in ancient times - Talos

… and then: a multitude of robotic systems; Industrial robots

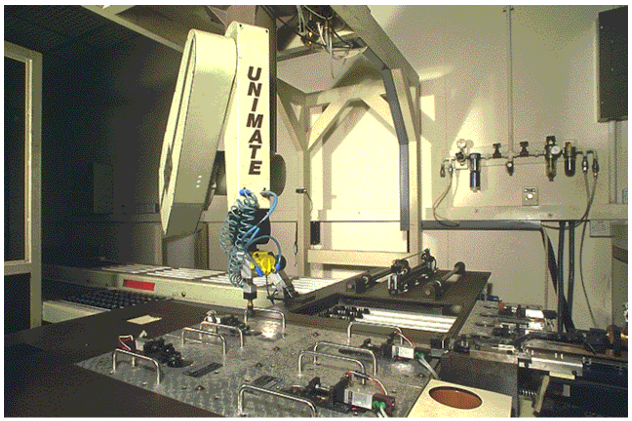
More Recently

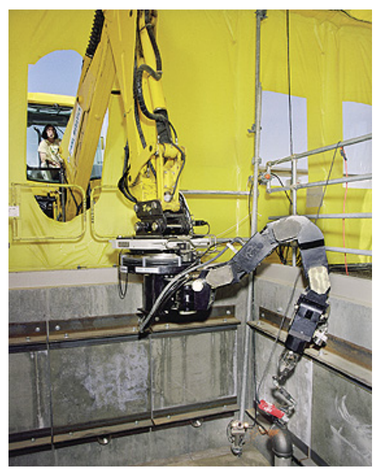
Trend towards intelligent systems

↓

Basic keyword: autonomous systems

Existing, non-autonomous systems…

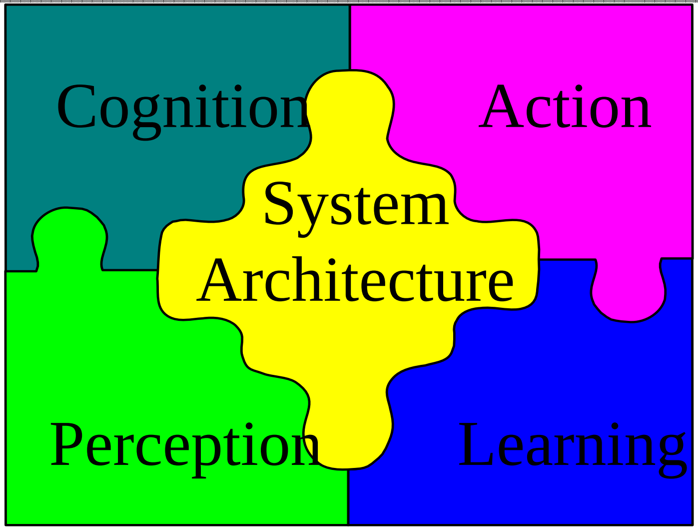




... and non-existent, fully autonomous robots

General research goal

Development of robotic systems able to exhibit autonomous behavior in complex and dynamic environments



An Interesting Research Area

Theoretical interest…

- **Mathematical and computational modeling of perception and action**

… with important applications

- **“Intelligent” robotic wheelchairs**
- **Robotic tour-guides in museums and exhibition sites**
- **Exploration of unknown and, possibly hostile, environments**
- **Routine tasks (surveillance, cleaning, etc)**

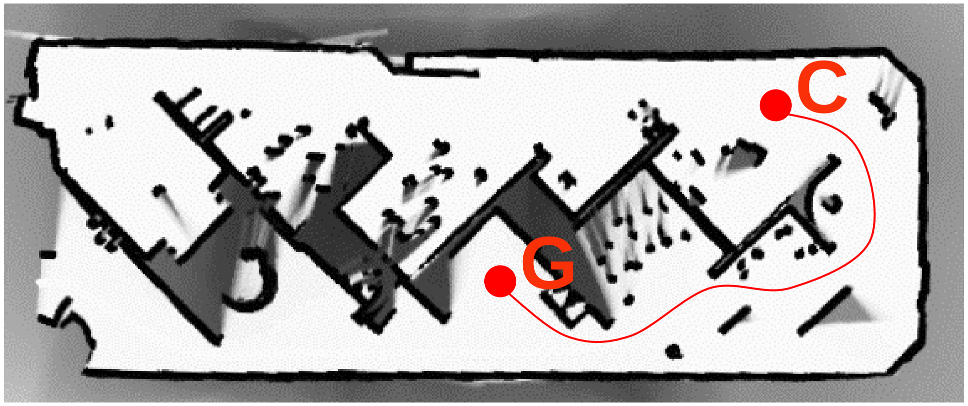
Research Directions

The above are based to a great extent on the ability of <u>autonomous navigation</u>

Autonomous Navigation Research Directions

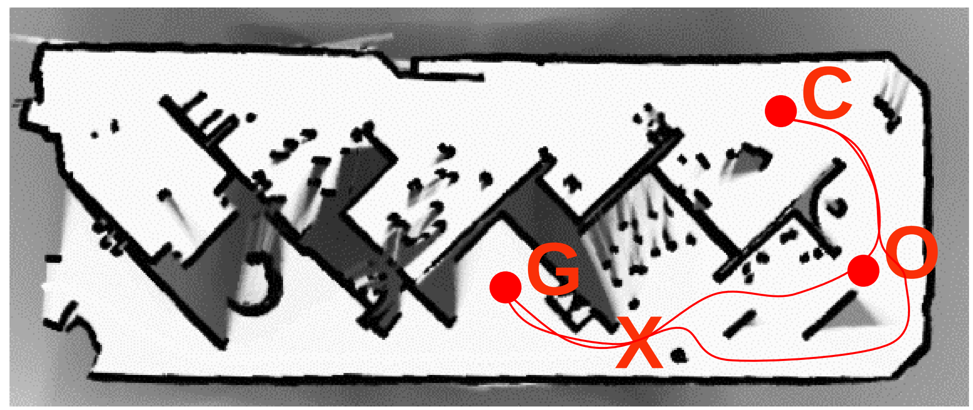
- Given
- An environment representation - Map
- Knowledge of current position C
- Target position G

- A path has to be planned and tracked that will take the robot from C to G



Autonomous Navigation Research Directions

- During execution (run-time)
- Objects / Obstacles O may block the robot
- The planned path is no-longer valid
- The obstacle needs to be avoided and the path may need to be re-planned

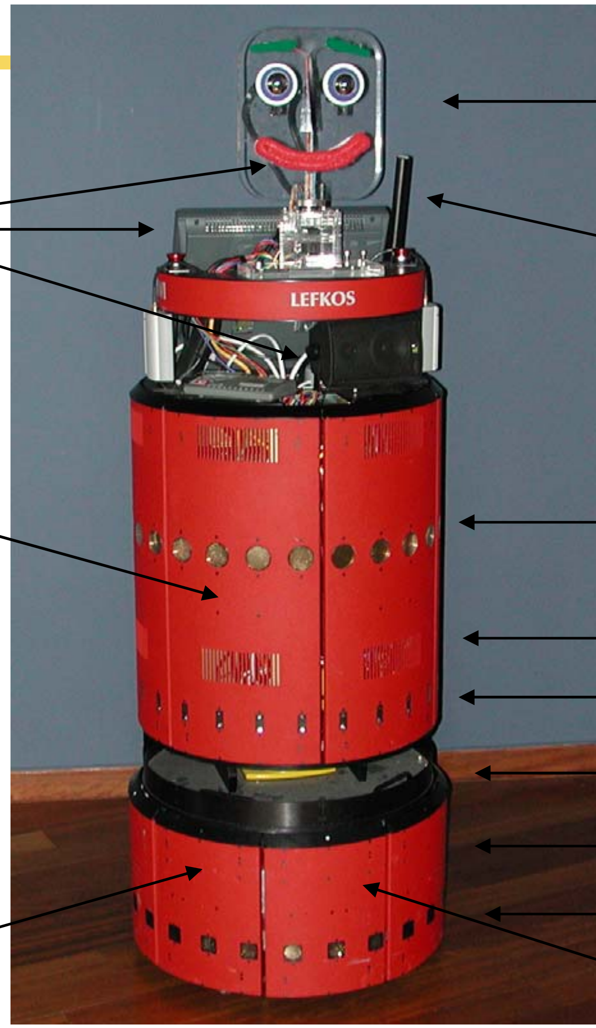


Navigation Issues

Important questions (Levitt et al '91)	*Important navigation issues*
Where am I	Robot localization
Where are other places relative to me	Map building
How do I get to other places from here	Path/m

Wednesday

Navigation Issues – Underlying HW



Interaction

Processing Power

Motors

Sensors

Stereo vision

Communications

Sonars

Bump sensors

Infrared sensors

Laser scanner

Laser Scanner

Bump sensors

Sonars

Odometry

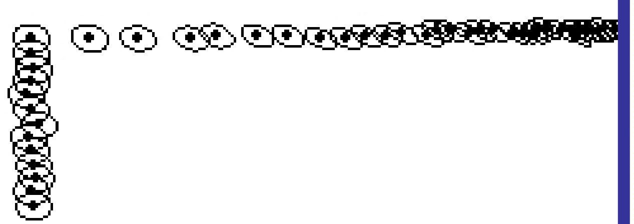
Range Sensor Model

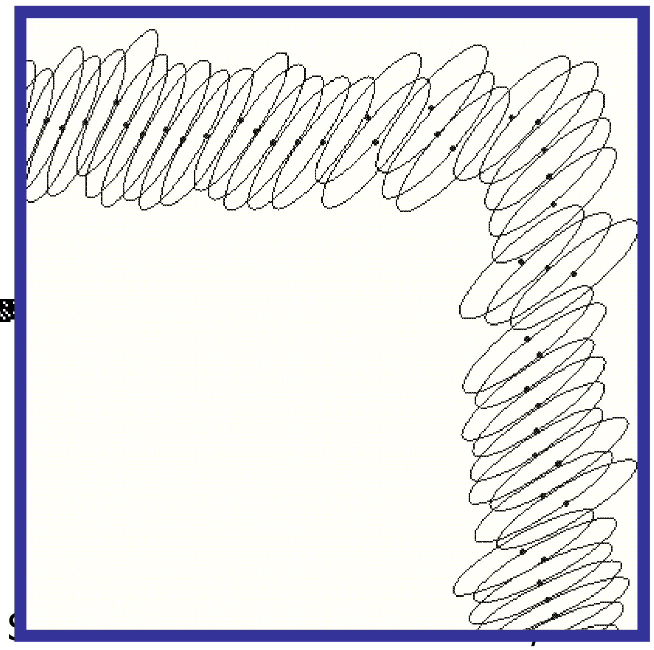
- Laser Rangefinder
- Model range and angle errors.

$$[x, y]^T = Exp\left(R(r,\phi)\right) = \left[r\cos(\phi), r\sin(\phi)\right]^T$$

$$\Sigma_{polar} = \begin{bmatrix} k_{\phi}\phi & 0 \\ 0 & k_{\rho_0} + k_{\rho_1} r \end{bmatrix}$$

$$\Sigma_p = \nabla R \Sigma_{polar} \nabla R^T$$





Need for Modeling

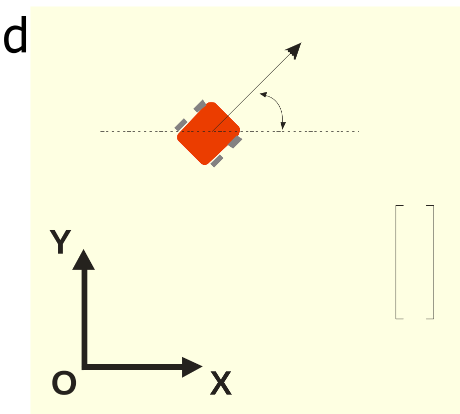
Robot
\+ Environment

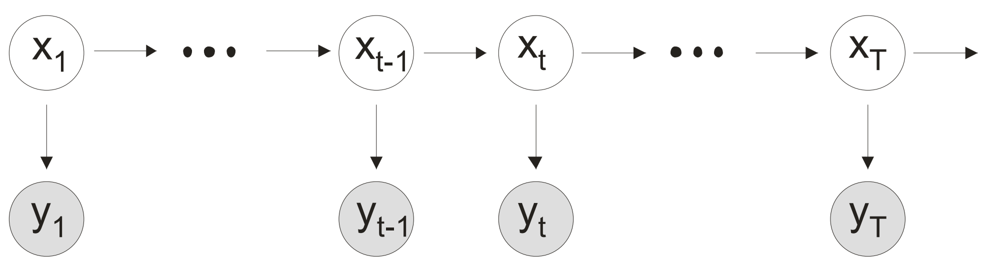
Extremely Complex
Dynamical System

⇨ Need for Appropriate Modeling

Markov Assumption

- State depends only on previous state and observations
- Static world assumption
- Hidden Markov Model (HMM)

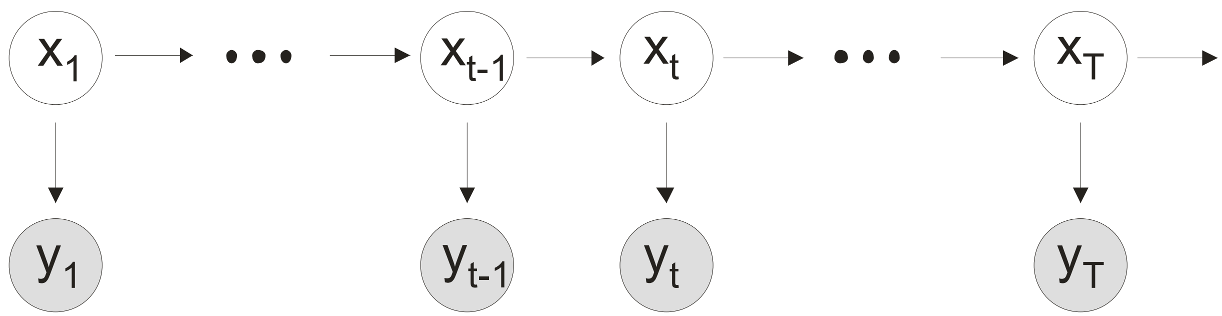




$$P\left(x_t \middle| y_1, y_2, \ldots, y_T\right) \Longrightarrow P(x_1)P\left(y_1 \middle| x_1\right)\prod_{t=2}^{T} P\left(x_t \middle| x_{t-1}\right)P\left(y_t \middle| x_t\right)$$

Bayesian estimation: Attempt to construct the posterior distribution of the state given all measurements

A Dynamic System



- Most commonly - Available:
  - Initial State $\mathbf{x}_1 \leftrightarrow P(\mathbf{x}_1)$
  - Observations $\mathbf{y}_1 .. \mathbf{y}_T$
  - System (motion) Model $\mathbf{x}_k = f_k(\mathbf{x}_{k-1}) \quad \leftrightarrow \quad p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$
  - Measurement (observation) Model

$$\mathbf{y}_k = h_k(\mathbf{x}_k) \quad \leftrightarrow \quad p(\mathbf{y}_k \mid \mathbf{x}_k)$$

Inference - Learning

- Localization (inference task)
  Compute the probability that the robot is at pose z at time t given all observations up to time t (forward recursions only)

$$P\left(x_t = z \middle| y_1, y_2, \ldots, y_t\right)$$

- Map building (learning task)
  Determine the map m that maximizes the probability of the observation sequence.

$$m^* = \arg\max_m P\left(m \middle| y_1, y_2, \ldots, y_T\right)$$

Belief State

$$\boxed{P(x_t \mid y_1, y_2, \ldots, y_t)} = \frac{1}{c_t} P(y_t | x_t) \int_Z P(x_t | x_{t-1} = z) \boxed{P(x_{t-1} = z \mid y_1, y_2, \ldots, y_{t-1})} dz$$

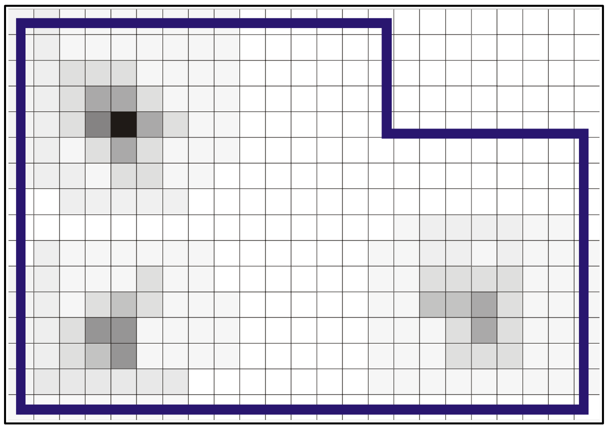
How is the posterior distribution calculated?	How is the prior distribution represented?

- Discrete representation
  - Grid (Dynamic) — (Dynamic) Markov localization (Burgard98)
  - Samples — Monte Carlo localization (Fox99)
- Continuous representation
  - Gaussian distributions — Kalman filters (Kalman60)

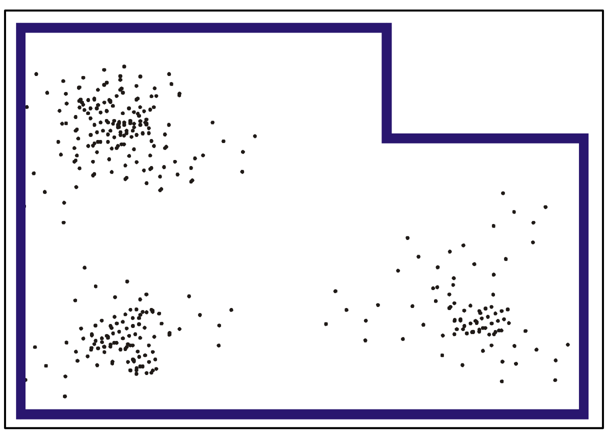
Example: State Representations for Robot Localization

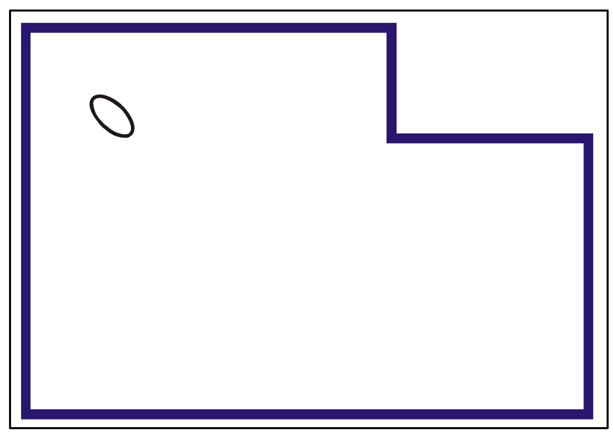
Discrete Representations | *Continuous Representations*

Grid Based approaches (Markov localization)



Particle Filters (Monte Carlo localization)





Kalman Tracking

Kalman Filters - Equations

$$\left.\begin{aligned} P(x_t | x_{t-1}) &\approx N(Ax_{t-1}, \Gamma) \\ P(y_t | x_t) &\approx N(Cx_t, \Sigma) \end{aligned}\right\}$$

A: State transition matrix (n x n)

C: Measurement matrix (m x n)

w: Process noise ($\in R^n$),

v: Measurement noise($\in R^m$)

$$\left.\begin{aligned} x_t &= A_t x_{t-1} + w_t \\ y_t &= C_t x_t + v_t \\ w_t &\approx N(0, \Gamma) \\ v_t &\approx N(0, \Sigma) \end{aligned}\right\}$$

Process dynamics (motion model)

measurements (observation model)

Where : $N(x; m, V) = \dfrac{1}{|2\pi V|^{1/2}} \exp\left(-\dfrac{1}{2}(x-m)^T V^{-1}(x-m) \right)$

Kalman Filters - Update

$$\left.\begin{aligned} x_t &= A_t x_{t-1} + w_t \\ y_t &= C_t x_t + v_t \\ w_t &\approx N(0, \Gamma) \\ v_t &\approx N(0, \Sigma) \end{aligned}\right\}$$

Predict

$$\hat{x}_t^- = A\hat{x}_{t-1}$$

$$P_t^- = AP_{t-1}A^T + \Gamma$$

Compute Gain

$$K_t = P_t^- C^T (CP_t^- C^T + \Sigma)^{-1}$$

Compute Innovation

$$J_t = \hat{y}_t - C\hat{x}_t^-$$

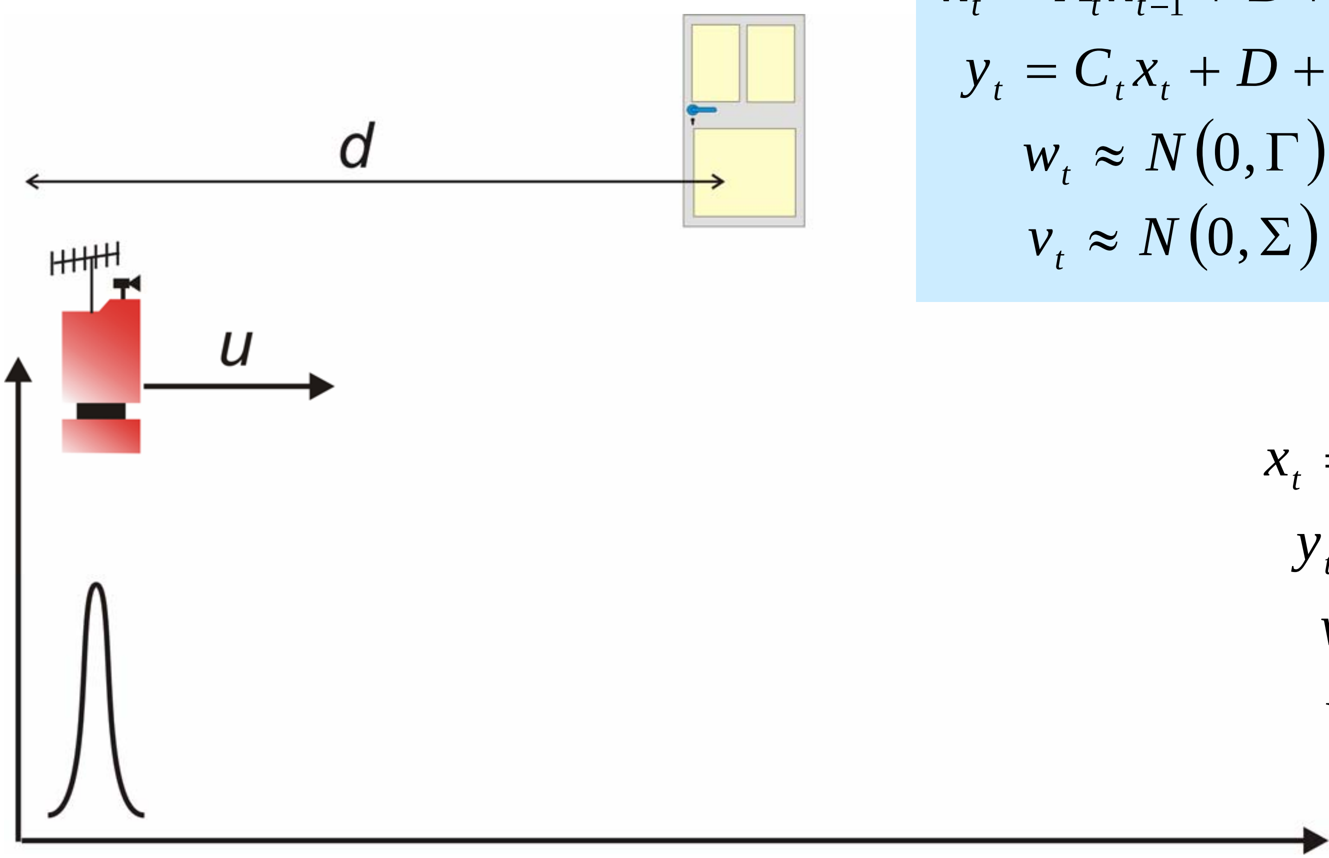
Update

$$\hat{x}_t = \hat{x}_t - \mathrm{K}_t J_t$$

$$P_t = (I - K_t C)P_k^-$$

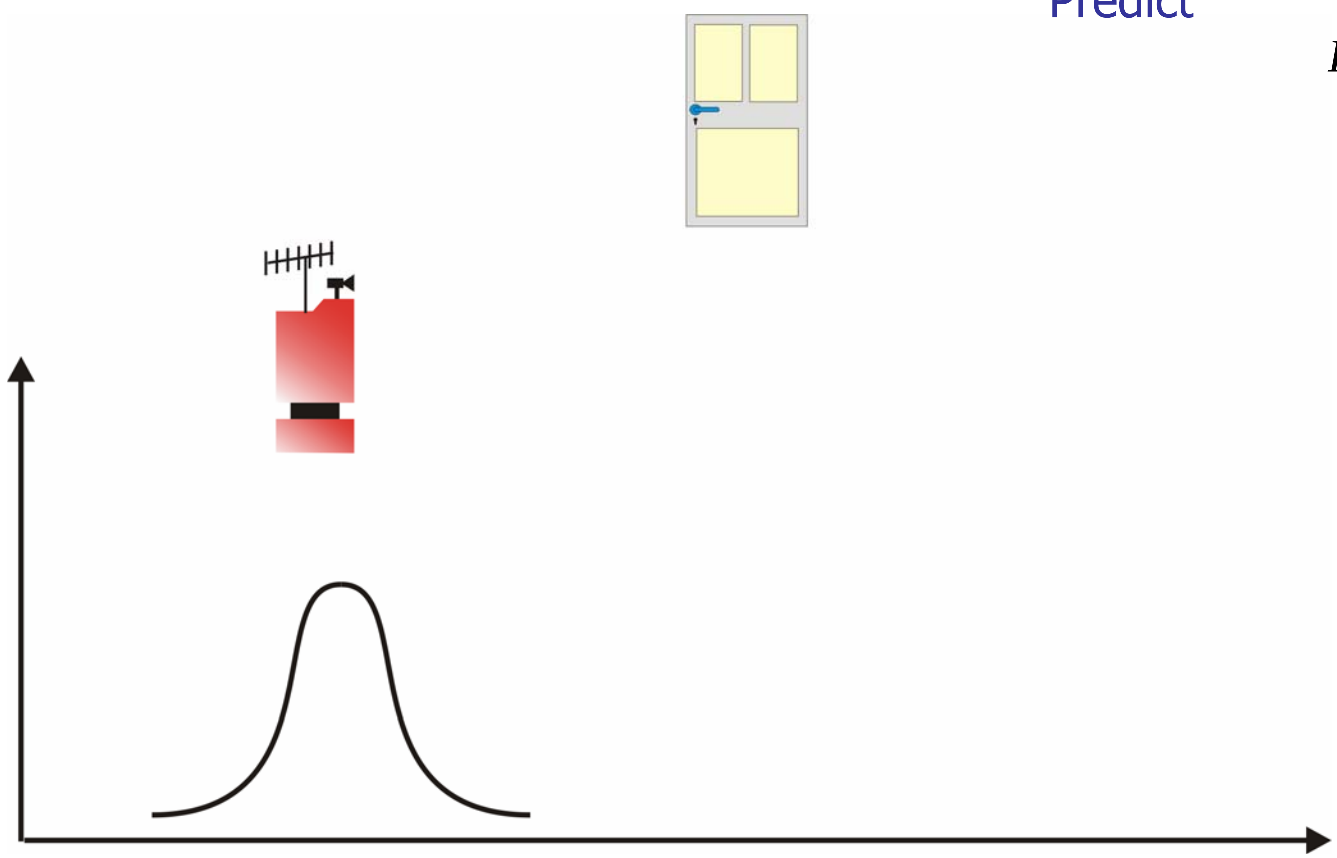
Kalman Filter - Example

$$
\begin{aligned}
x_t &= A_t x_{t-1} + B + w_t \\
y_t &= C_t x_t + D + v_t \\
w_t &\approx N(0, \Gamma) \\
v_t &\approx N(0, \Sigma)
\end{aligned}
\qquad
\begin{aligned}
A_t &= [1] \\
B_t &= [u_t] \\
C_t &= [-1] \\
D_t &= [1]
\end{aligned}
$$



$$
\left.
\begin{aligned}
x_t &= x_{t-1} + u_t + w_t \\
y_t &= d - x_t + v_t \\
w_t &\approx N(0, \Gamma) \\
v_t &\approx N(0, \Sigma)
\end{aligned}
\right\}
$$

Kalman Filter - Example



Predict

$$\hat{x}_t^- = A\hat{x}_{t-1} + B$$

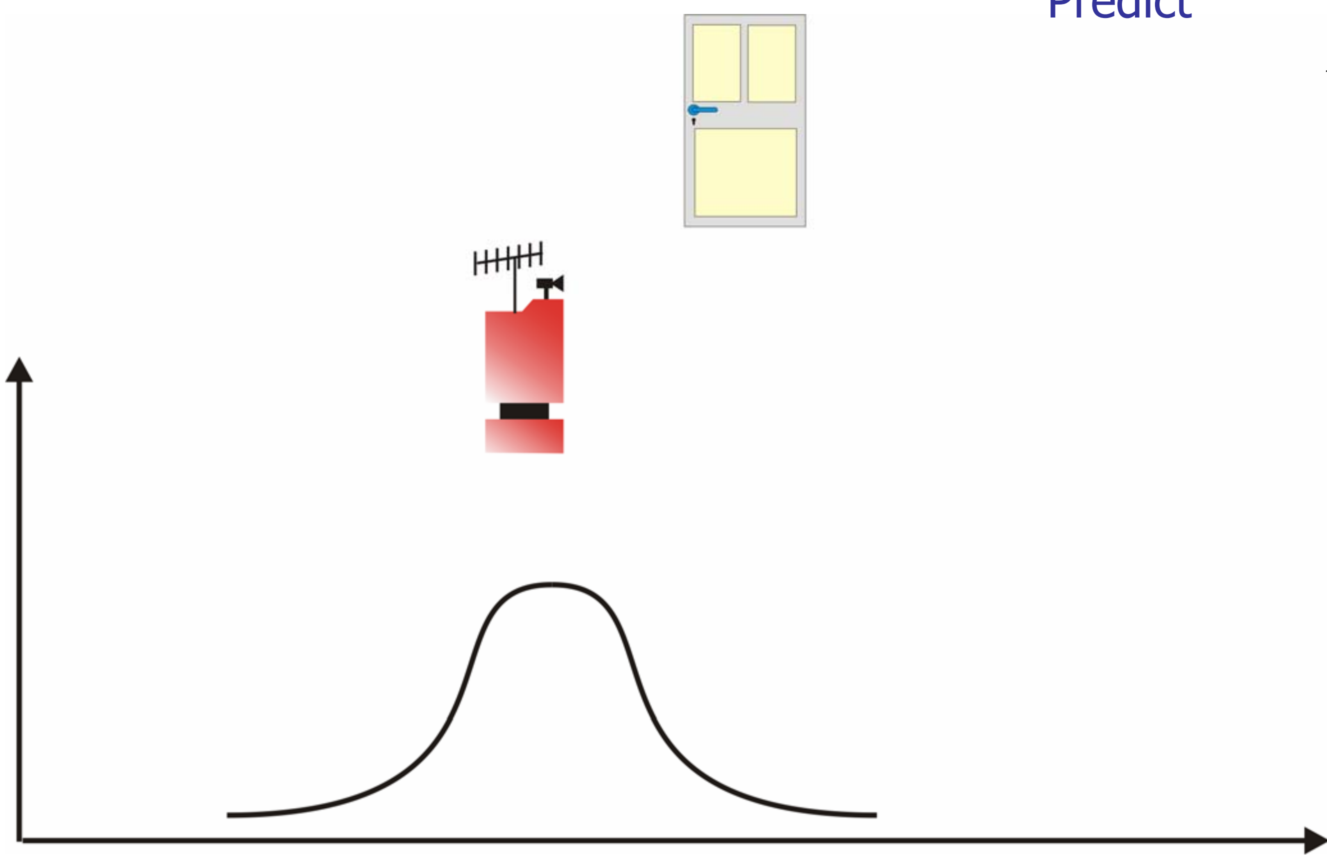
$$P_t^- = AP_{t-1}A^T + \Gamma$$

Kalman Filter - Example

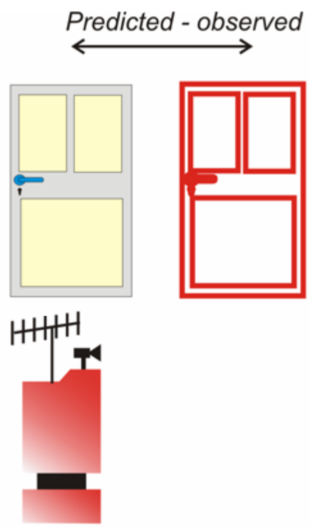
Predict

$$\hat{x}_t^- = A\hat{x}_{t-1} + B$$

$$P_t^- = AP_{t-1}A^T + \Gamma$$



Kalman Filter - Example



Predict

$$\hat{x}_t^- = A\hat{x}_{t-1} + B$$

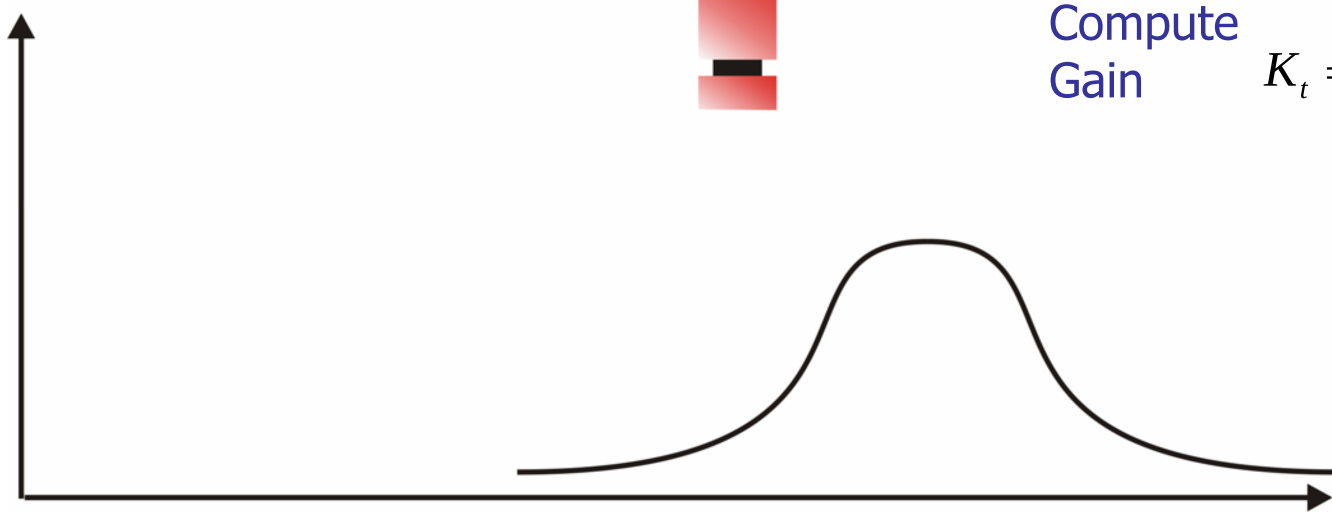
$$P_t^- = AP_{t-1}A^T + \Gamma$$

Compute Innovation

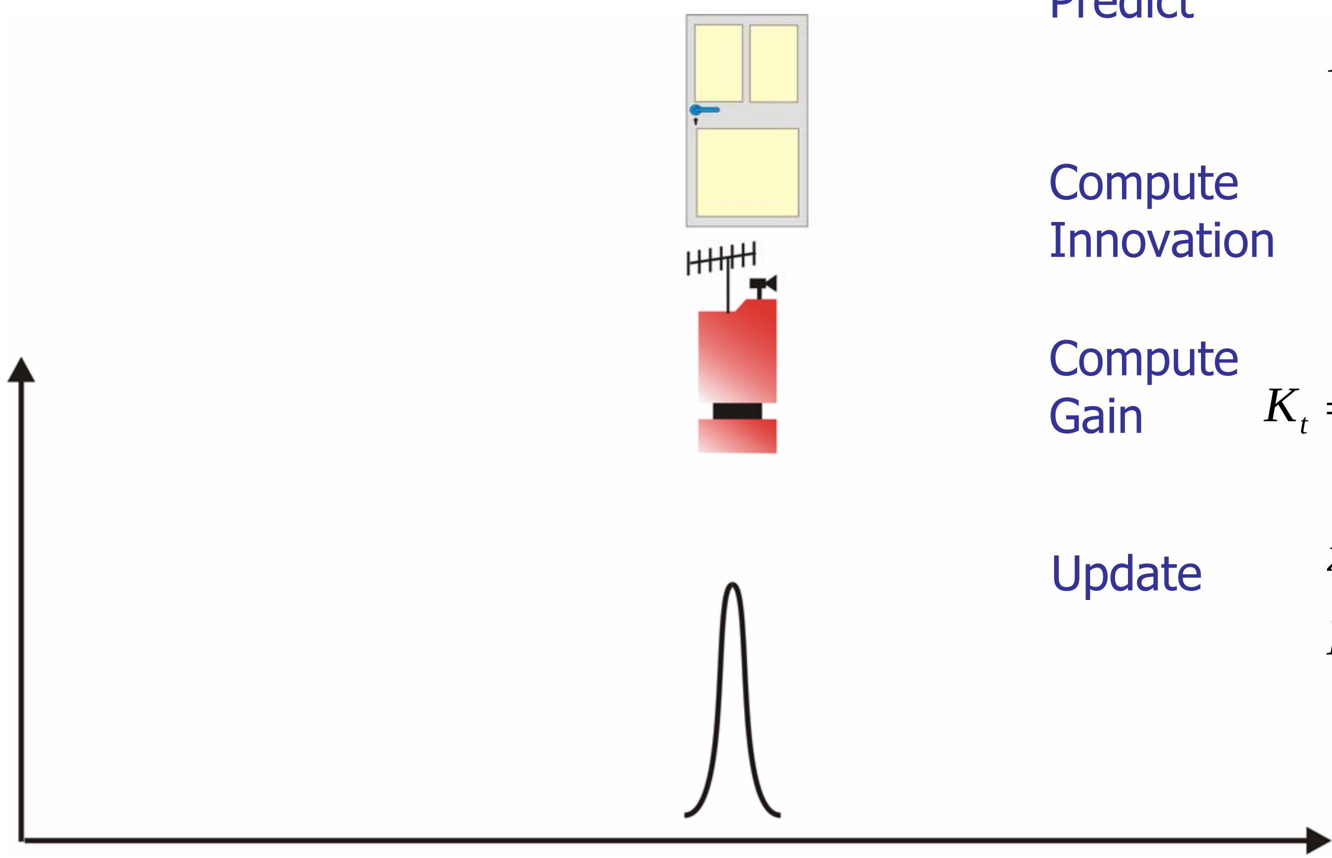
$$J_t = \hat{y}_t - C\hat{x}_t^-$$

Compute Gain

$$K_t = P_t^- C^T (CP_t^- C^T + \Sigma)^{-1}$$



Kalman Filter – Example



Predict

$$\hat{x}_t^- = A\hat{x}_{t-1} + B$$

$$P_t^- = AP_{t-1}A^T + \Gamma$$

Compute Innovation

$$J_t = \hat{y}_t - C\hat{x}_t^-$$

Compute Gain

$$K_t = P_t^- C^T (CP_t^- C^T + \Sigma)^{-1}$$

Update

$$\hat{x}_t = \hat{x}_t - \mathrm{K}_t J_t$$

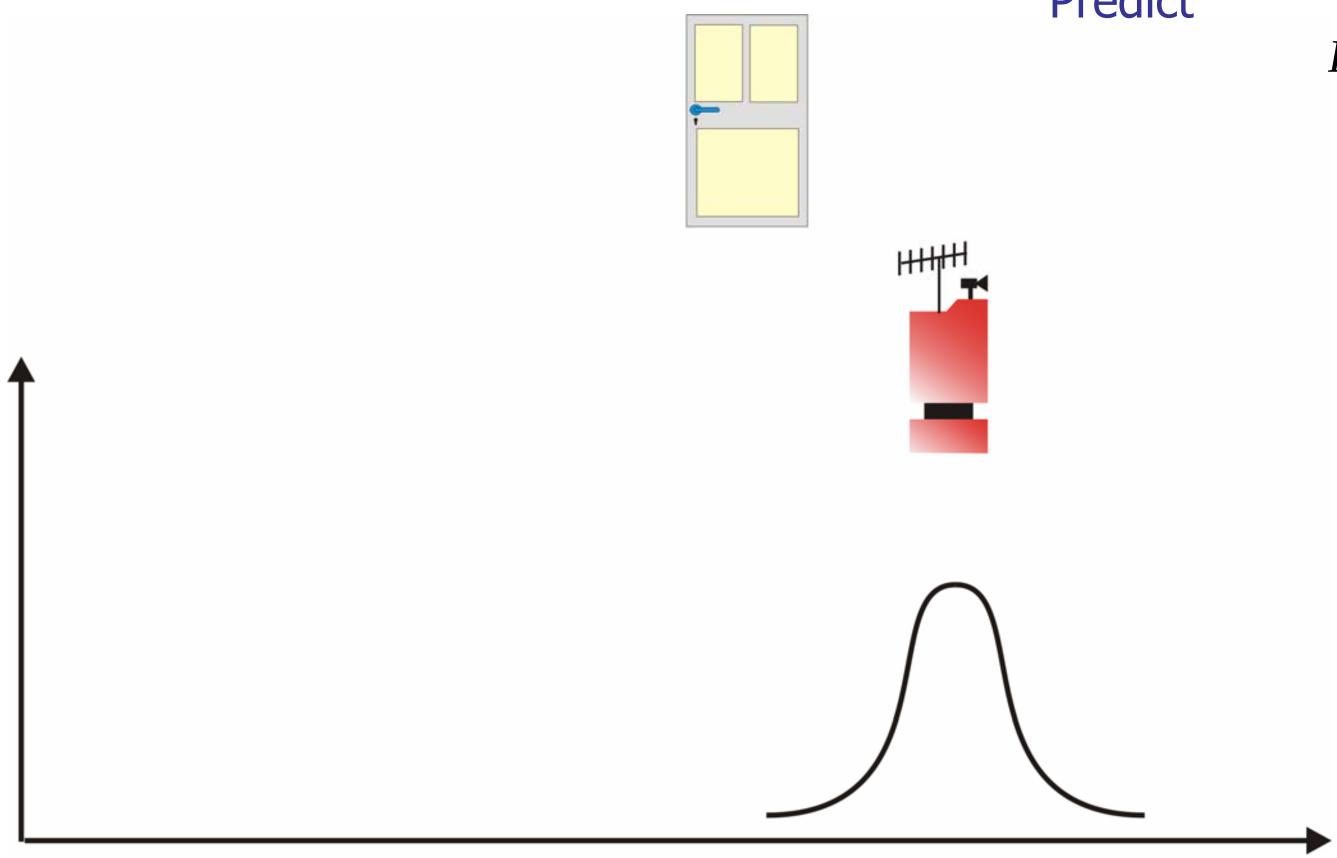
$$P_t = (I - K_t C)P_k^-$$

Kalman Filter – Example

Predict

$$\hat{x}_t^- = A\hat{x}_{t-1} + B$$

$$P_t^- = AP_{t-1}A^T + \Gamma$$

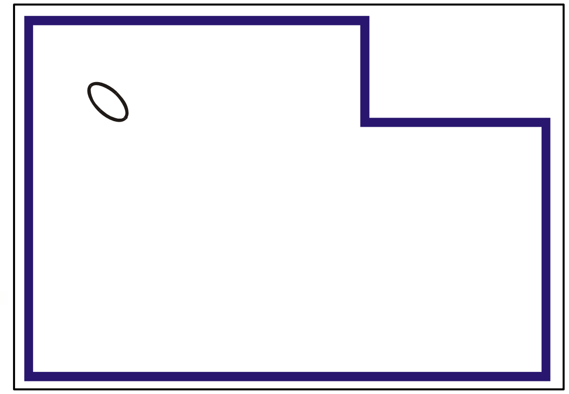


Non-Linear Case

- Kalman Filter assumes that system and measurement processes are linear
- Extended Kalman Filter -> linearized Case

$$\left.\begin{aligned} x_t &= A_t x_{t-1} + w_t \\ y_t &= C_t x_t + v_t \\ w_t &\approx N(0,\Gamma) \\ v_t &\approx N(0,\Sigma) \end{aligned}\right\} \longrightarrow \left.\begin{aligned} x_t &= f(x_{t-1}) + w_t \\ y_t &= g(x_t) + v_t \\ w_t &\approx N(0,\Gamma) \\ v_t &\approx N(0,\Sigma) \end{aligned}\right\}$$

Example: Localization – EKF

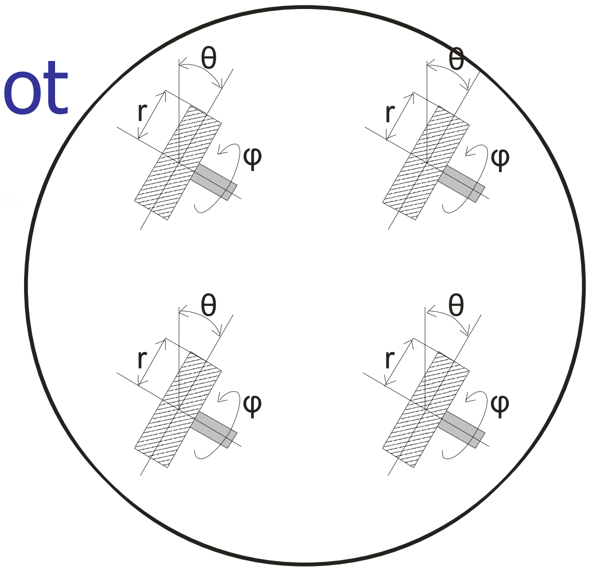


- Initialize State
  - Gaussian distribution centered according to prior knowledge – large variance
- At each time step:
  - Use previous state and motion model to predict new state
    (mean of Gaussian changes - variance grows)
  - Compare observations with what you expected to see from the predicted state – Compute Kalman Innovation/Gain
  - Use Kalman Gain to update prediction

Extended Kalman Filter

Project State estimates forward (prediction step)	$\mu_{x_{t+1}^-} = Exp(F(\mu_{x_t}, \alpha_t))$ $\Sigma_{x_{t+1}^-} = \nabla F_x \Sigma_{x_t} \nabla F_x^T + \nabla F_\alpha \Sigma_{\alpha_t} \nabla F_\alpha^T$
Predict measurements	$l_{t+1}^- = H(\mu_{x_{t+1}^-})$
Compute Kalman Innovation	$r_{t+1} = l_{t+1} - l_{t+1}^-$ $\Sigma_{r_{t+1}} = \nabla F_{x_{t+1}^-} \Sigma_{x_{t+1}^-} \nabla F_{x_{t+1}^-}^T + \Sigma_{l_{t+1}}$
Compute Kalman Gain	$K_{t+1} = \Sigma_{x_{t+1}^-} \nabla F_{x_{t+1}^-} \Sigma_{r_{t+1}^-}^{-1}$
Update Initial Prediction	$\mu_{x_{t+1}} = \mu_{x_{t+1}^-} + K_{t+1} r_{t+1}$ $\Sigma_{x_{t+1}} = \Sigma_{x_{t+1}^-} - K_{t+1} \Sigma_{r_{t+1}} K_{t+1}^T$

EKF – Example
motion model for mobile robot

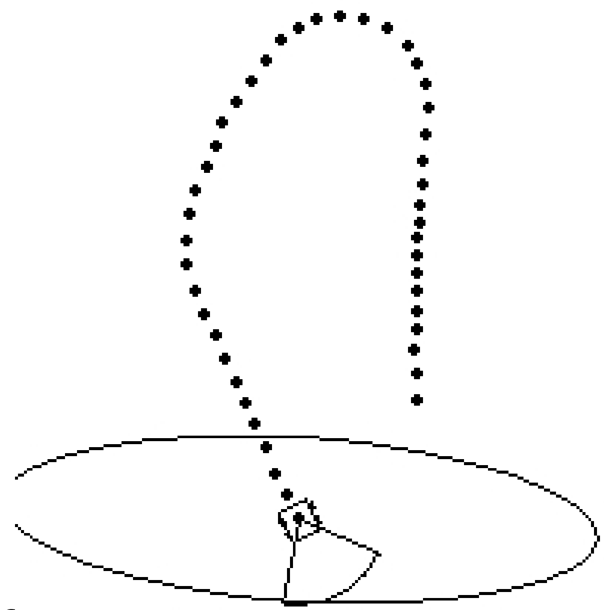


- Synchro-drive robot
- Model range, drift and turn errors

$$\Sigma_{a_t} = \begin{bmatrix} k_r d_t & 0 \\ 0 & k_t f_t + k_d d_t \end{bmatrix}$$

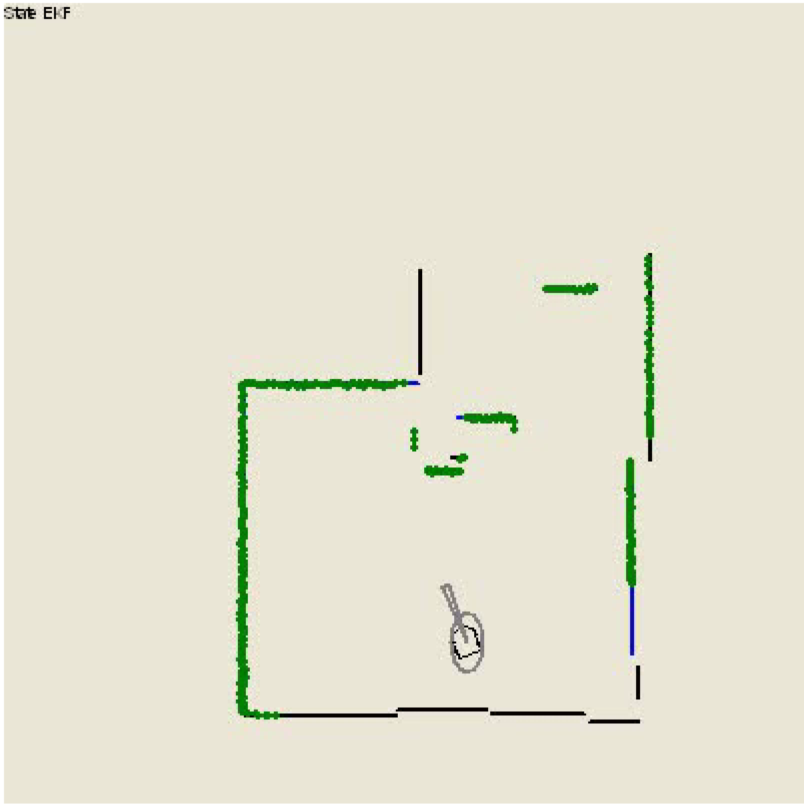
$$\mu_{x_{t+1}} = Exp\big(F(\mu_{x_t}, a_t)\big) = \begin{bmatrix} x_t - d_t \sin(\theta_t) \\ y_t - d_t \cos(\theta_t) \\ \theta_t + d_t \end{bmatrix}$$

$$\Sigma_{x_{t+1}} = \nabla F_x \Sigma_{x_t} \nabla F_x^T + \nabla F_a \Sigma_{a_t} \nabla F_a^T$$



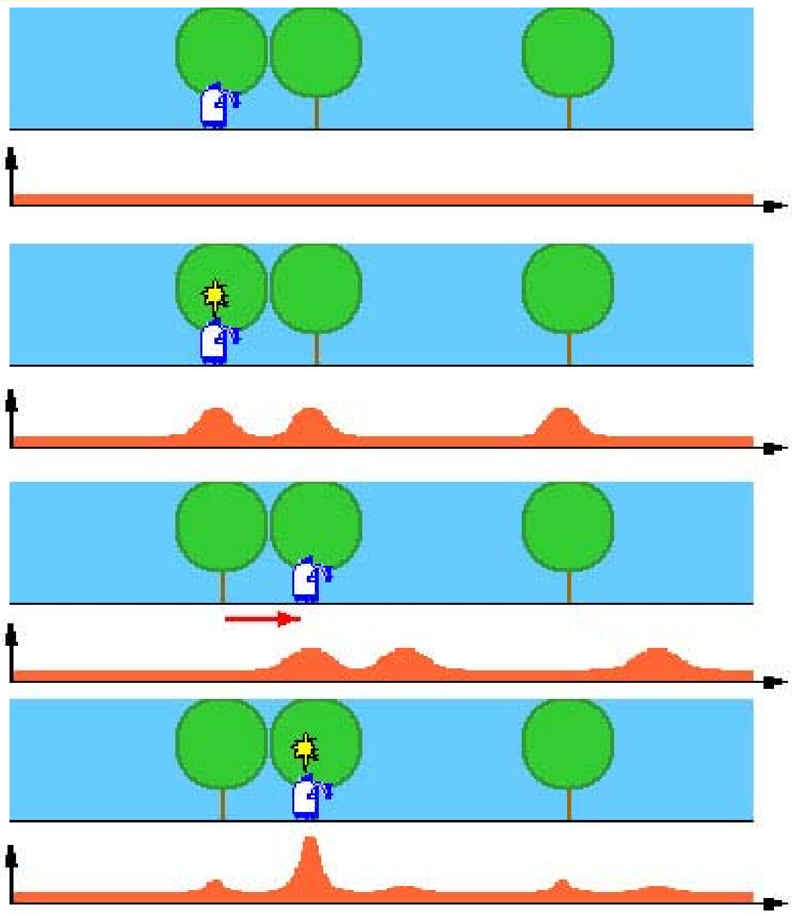
EKF – Example

simulated run



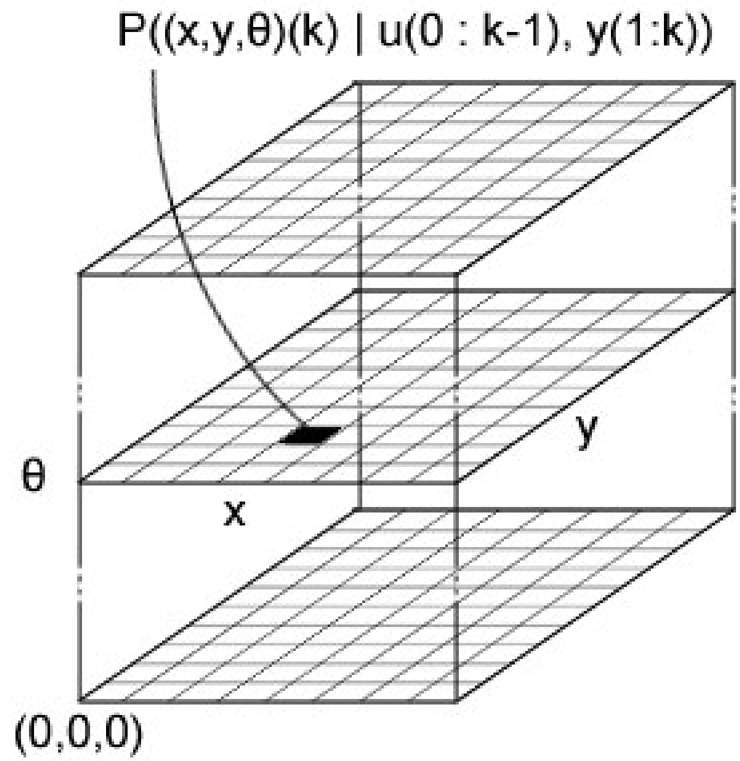
Bayesian Methods
Discrete Representation

Probabilistic localization – the case of global localization



Bayesian Methods
Discrete Approaches

Grid-based representation of the state-space

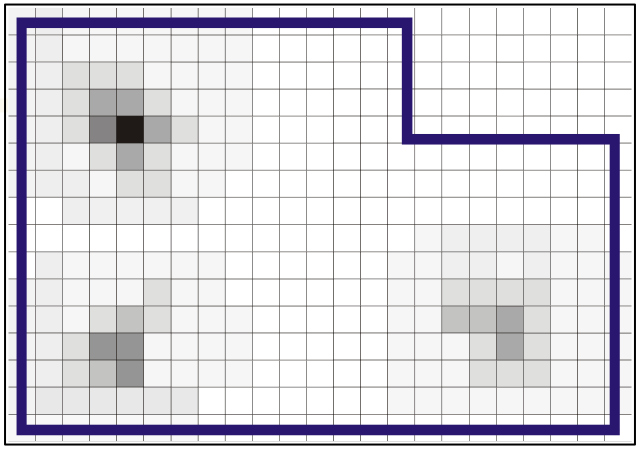


Bayesian Methods
Discrete Approaches

$$\overbrace{P\big(x(k)\,|\,u(0:k-1),y(1:k)\big)}^{\textbf{Posterior}} = n(k)\overbrace{P\big(y(k)\,|\,x(k)\big)}^{\textbf{Sensor model}} \bullet$$

$$\sum_{x(k-1)\in X}\Big[\underbrace{P\big(x(k)\,|\,u(k-1),x(k-1)\big)}_{\textbf{Motion model}}\underbrace{P\big(x(k-1)\,|\,u(0:k-2),y(1:k-1)\big)}_{\textbf{Past history}}\Big]$$

Example: Localization – Grid Based



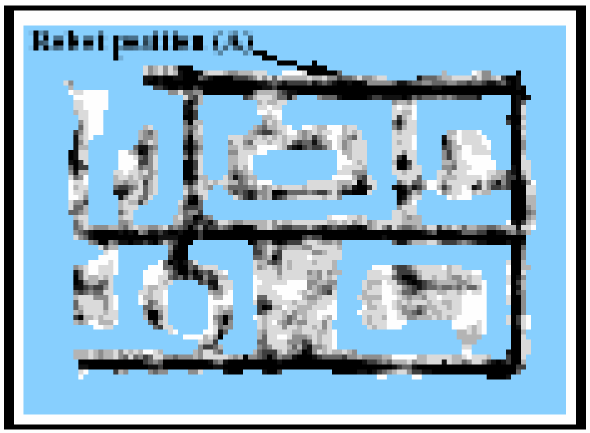
- Initialize Grid
  (Uniformly or according to prior knowledge)
- At each time step:
  - For each grid cell
    - Use observation model to compute $P\big(y(k)\,|\,x(k)\big)$
    - Use motion model and probabilities to compute

$$\sum_{x(k-1)\in X}\Big[P\big(x(k)\,|\,u(k-1),x(k-1)\big)\,P\big(x(k-1)\,|\,u(0:k-2),y(1:k-1)\big)\Big]$$

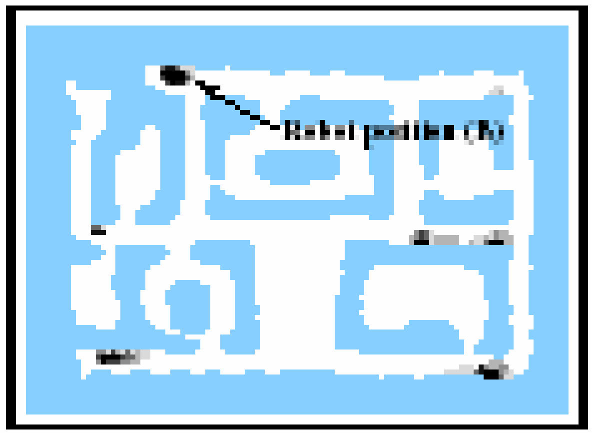
  - Normalize

Bayesian Methods
Discrete Approaches

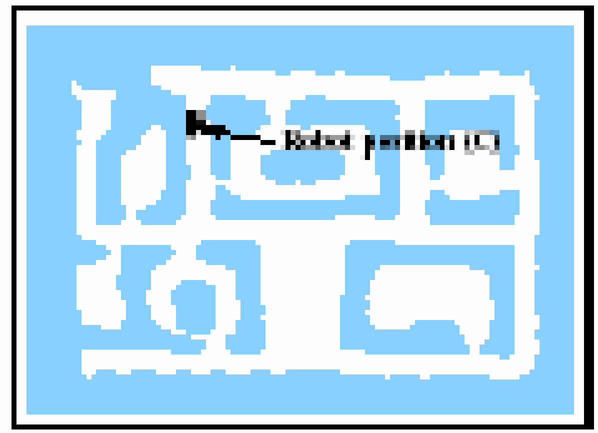
Density plots of the robot state



… beginning

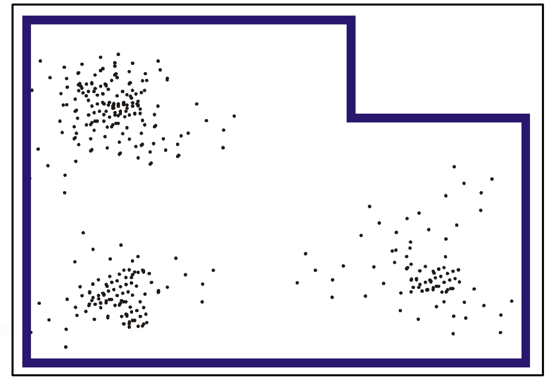


small No of cycles



sufficient No of cycles

Particle Filters



- Often models are non-linear and noise in non gausian.
- Use particles to represent the distribution
- "Survival of the fittest"

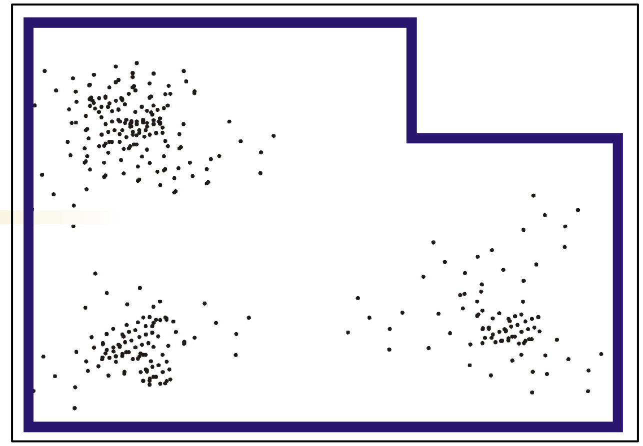
$$P(x_t \mid y_{1:t}) = \frac{1}{c_t} P(y_t | x_t) \boxed{\int_Z P(x_t | x_{t-1} = z) P(x_{t-1} = z \mid y_{1:t-1}) dz}$$

Motion model

Observation model (=weight)

Proposal distribution

Particle Filters
SIS-R algorithm



- Initialize particles randomly
  (Uniformly or according to prior knowledge)
- At each time step:

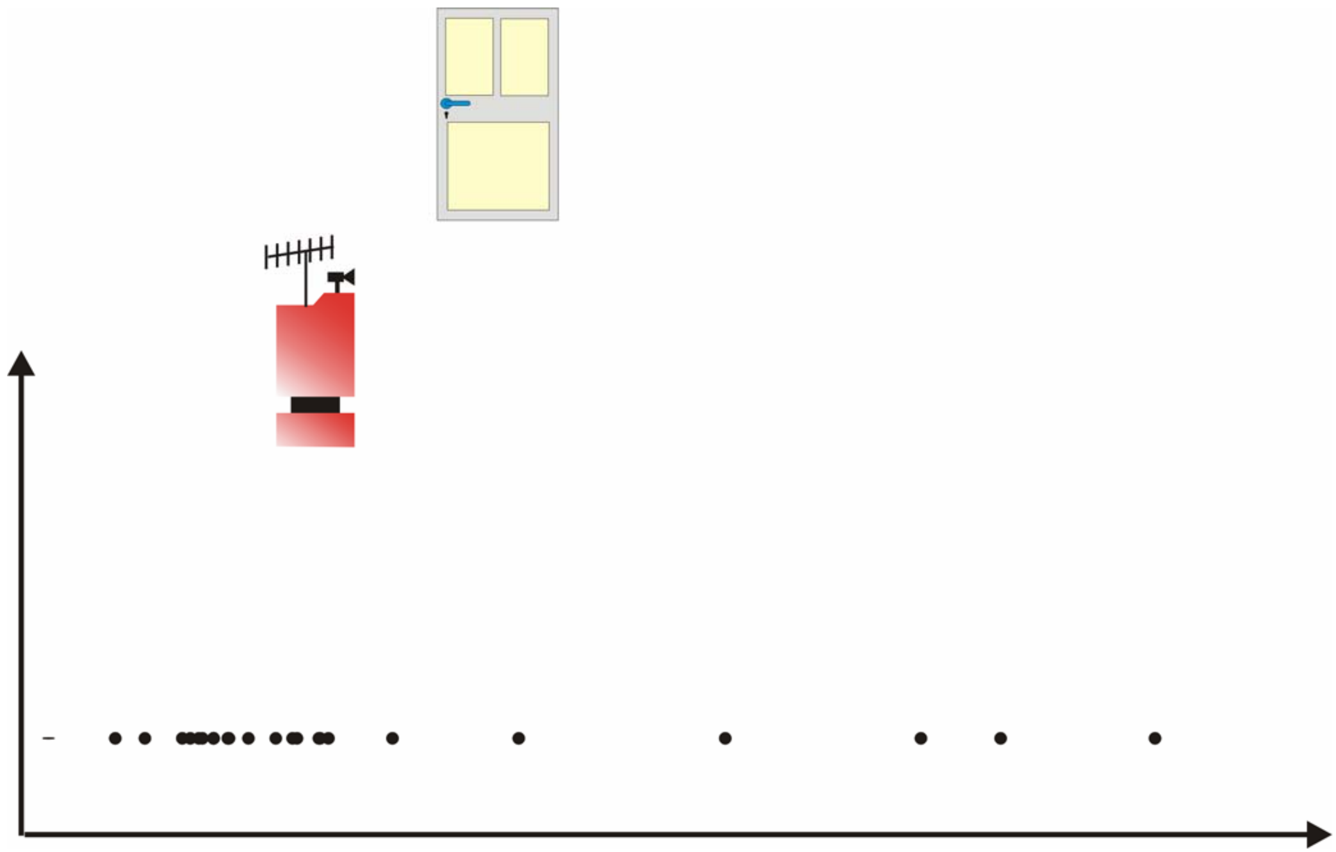
Sequential importance sampling

- For each particle:
  - Use motion model to predict new pose (sample from transition priors)
  - Use observation model to assign a weight to each particle (posterior/proposal)

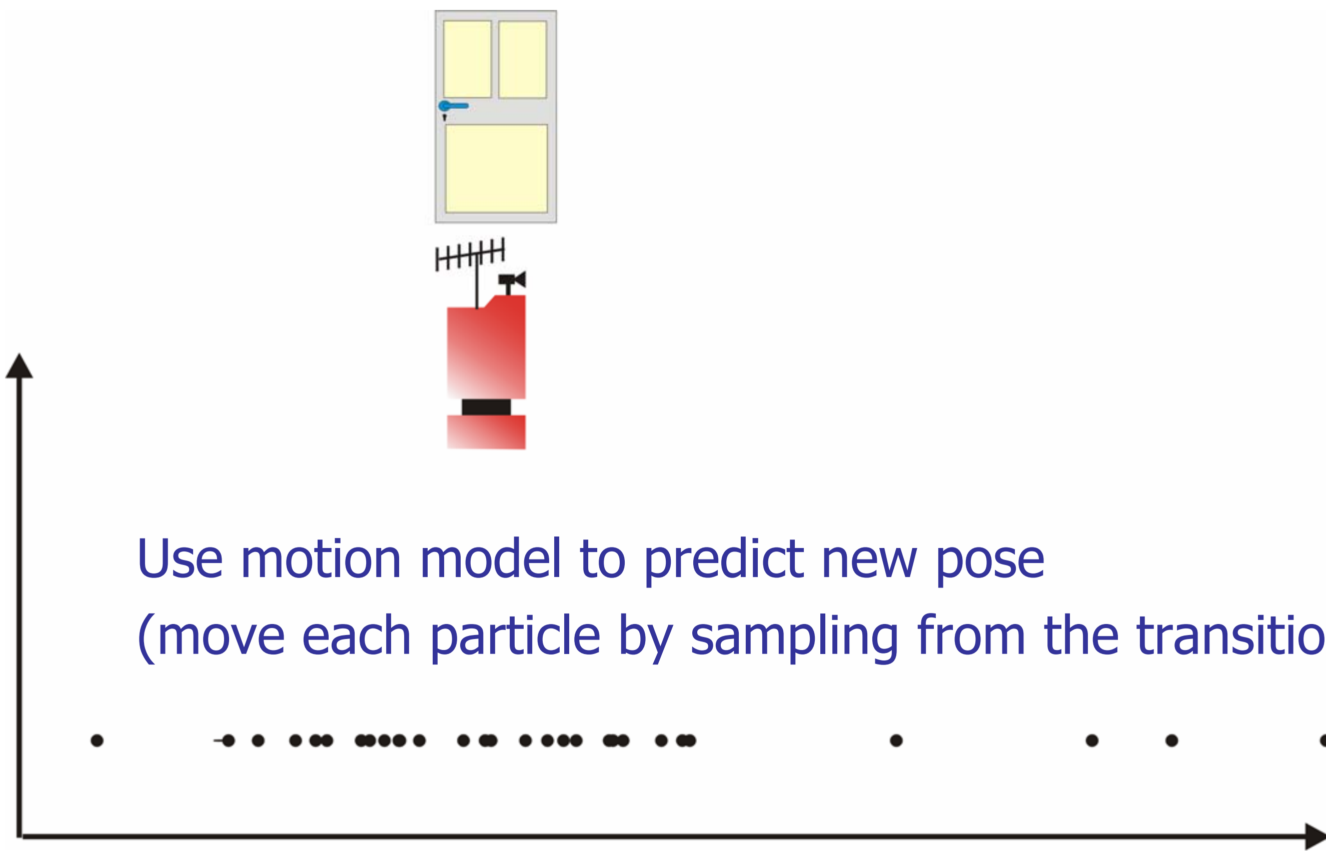
Selection: Re-sampling

- Create A new set of equally weighted particles by sampling the distribution of the weighted particles produced in the previous step.

Particle Filters – Example 1

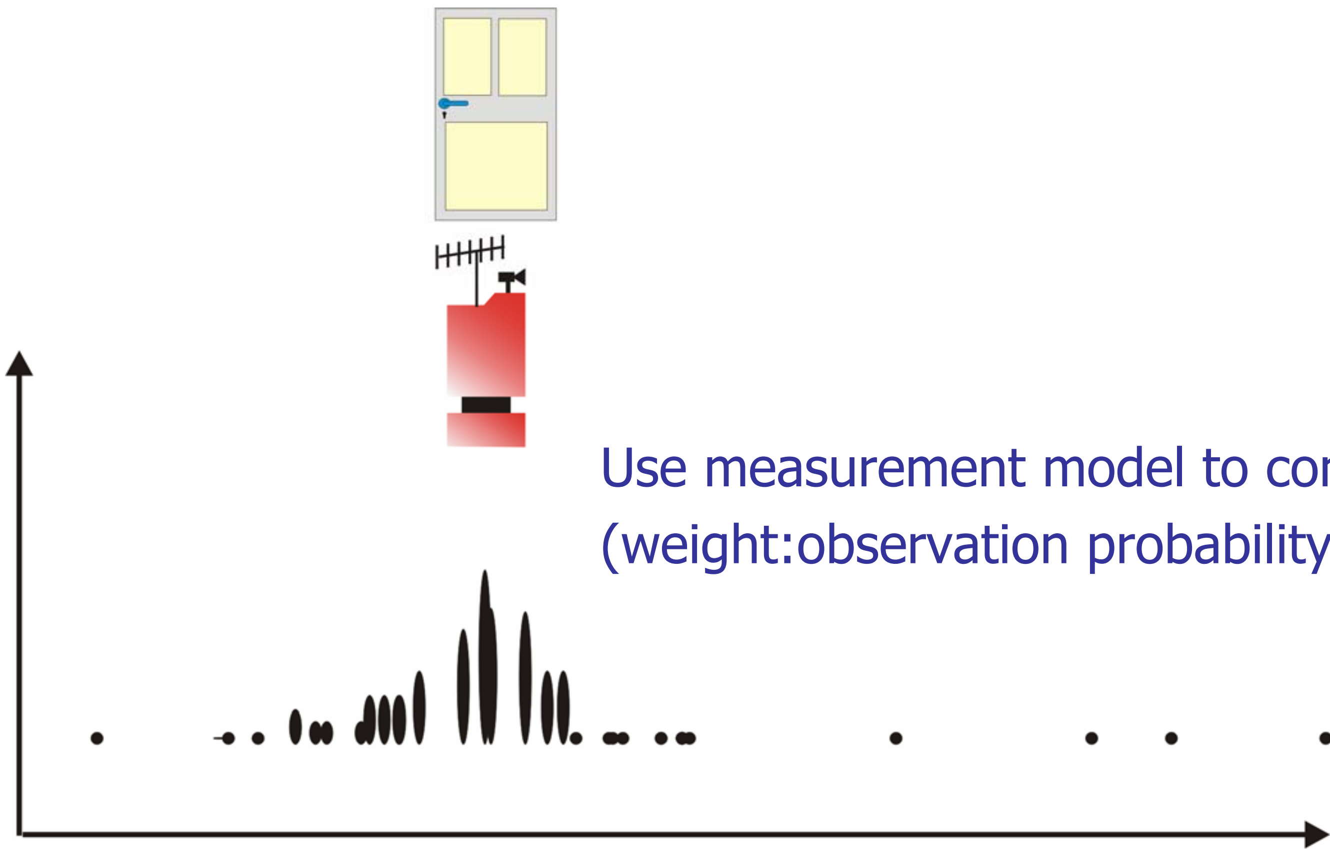


Particle Filters – Example 1



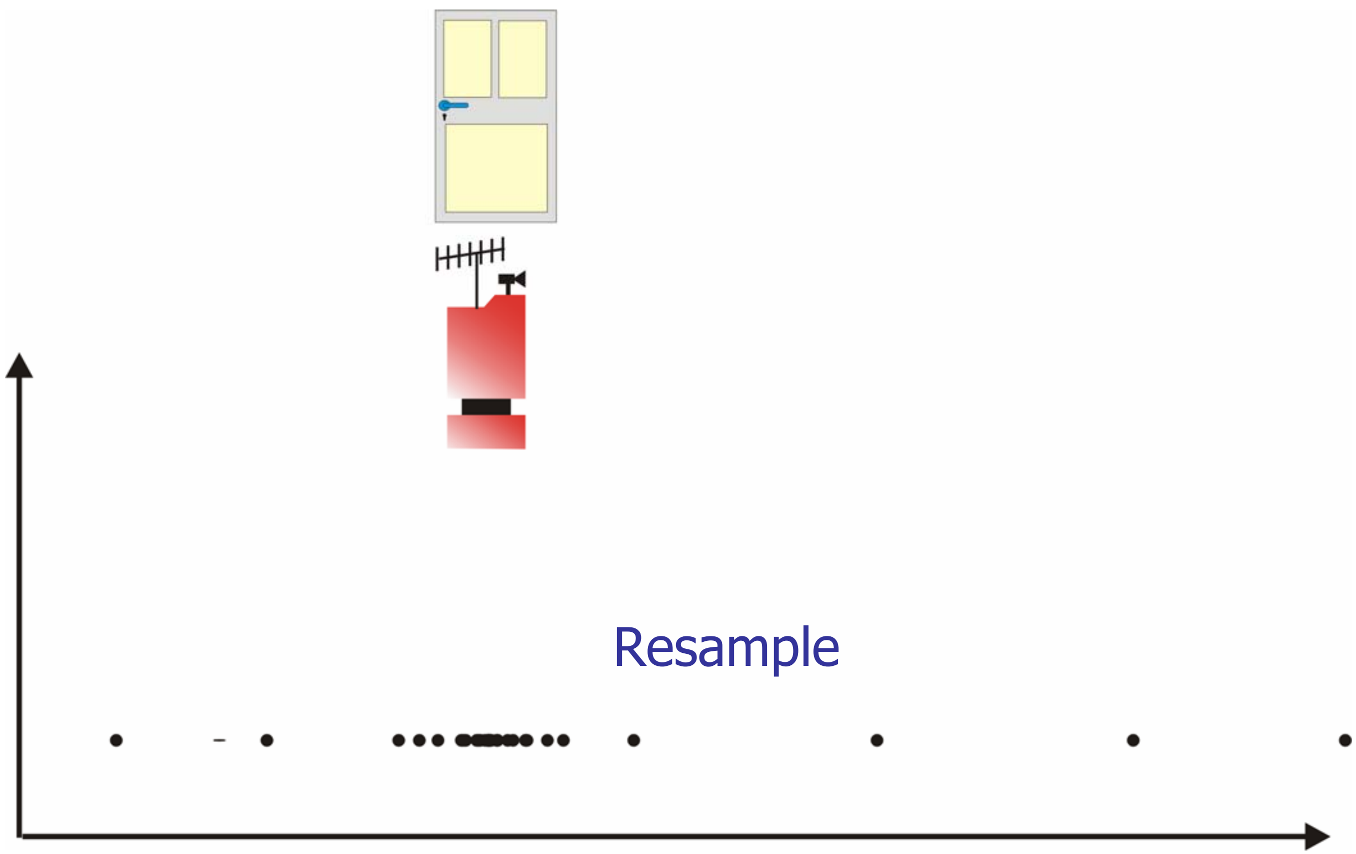
Use motion model to predict new pose
(move each particle by sampling from the transition prior)

Particle Filters – Example 1

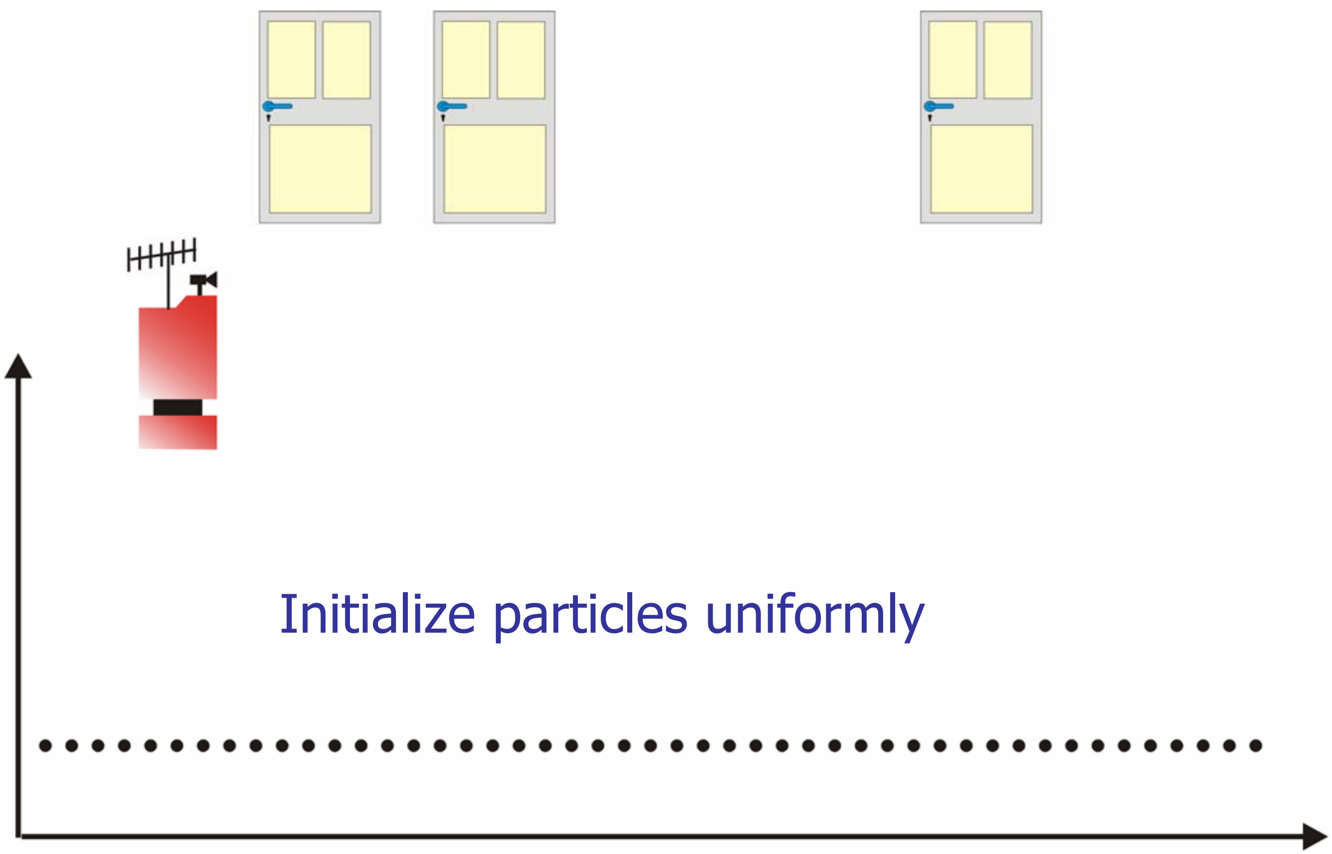


Use measurement model to compute weights (weight:observation probability)

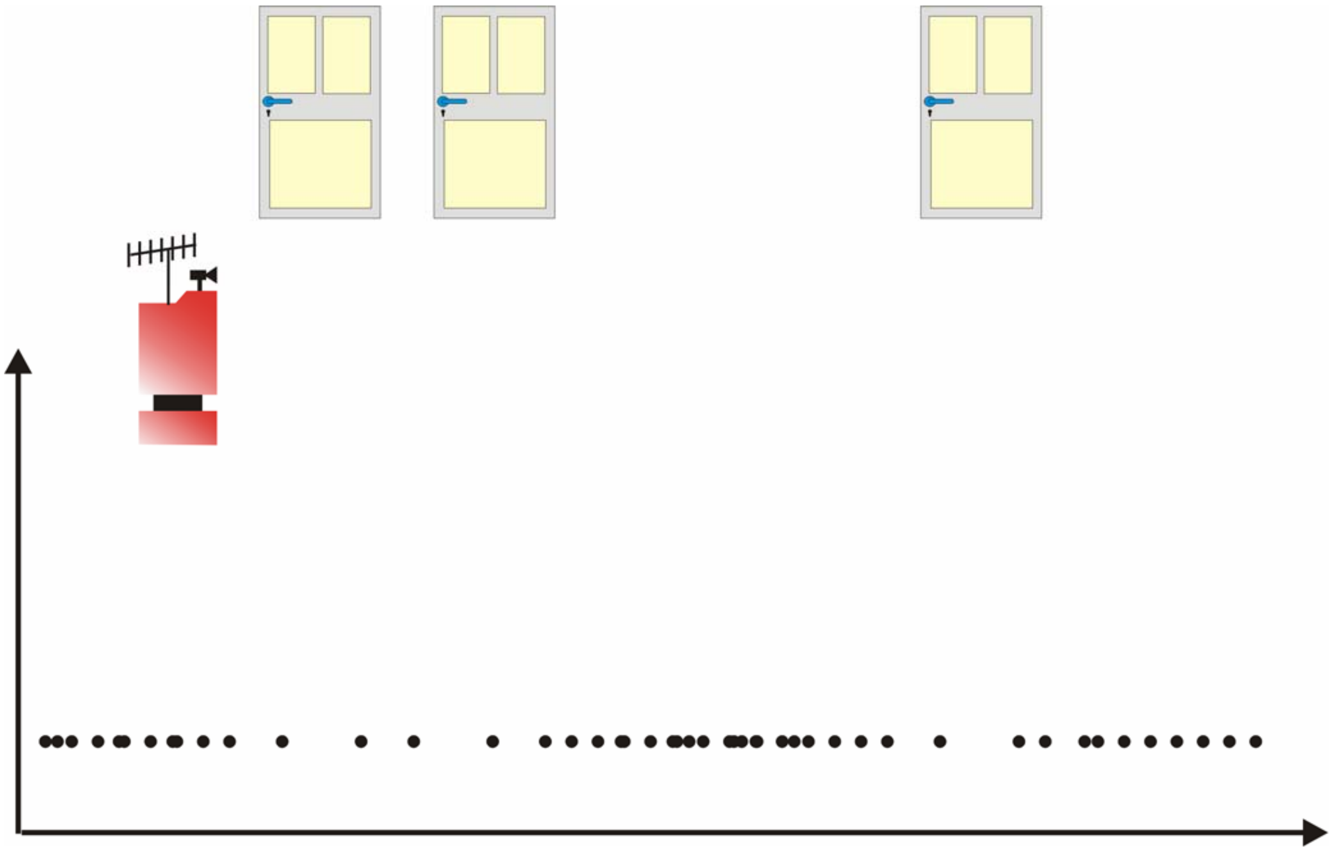
Particle Filters – Example 1



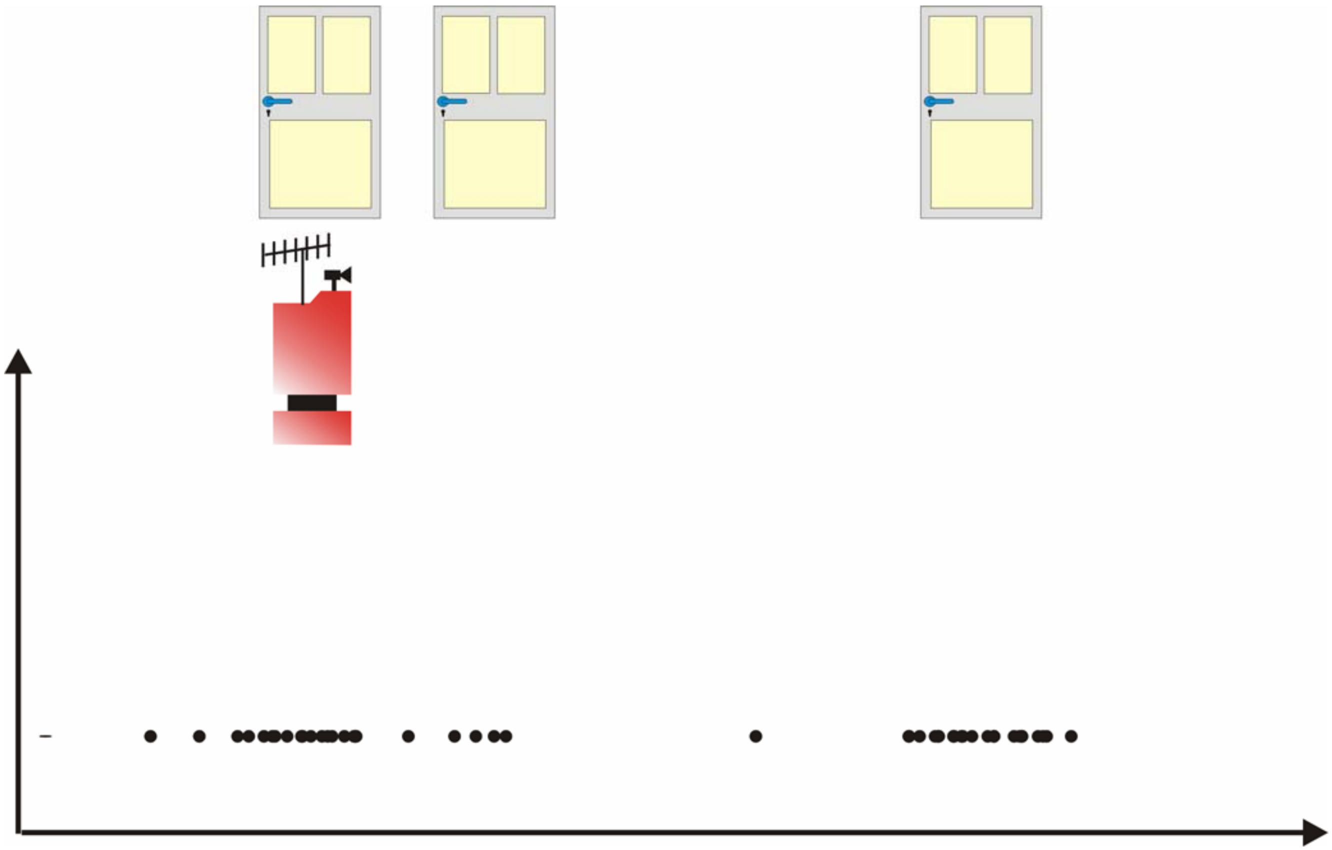
Particle Filters – Example 2



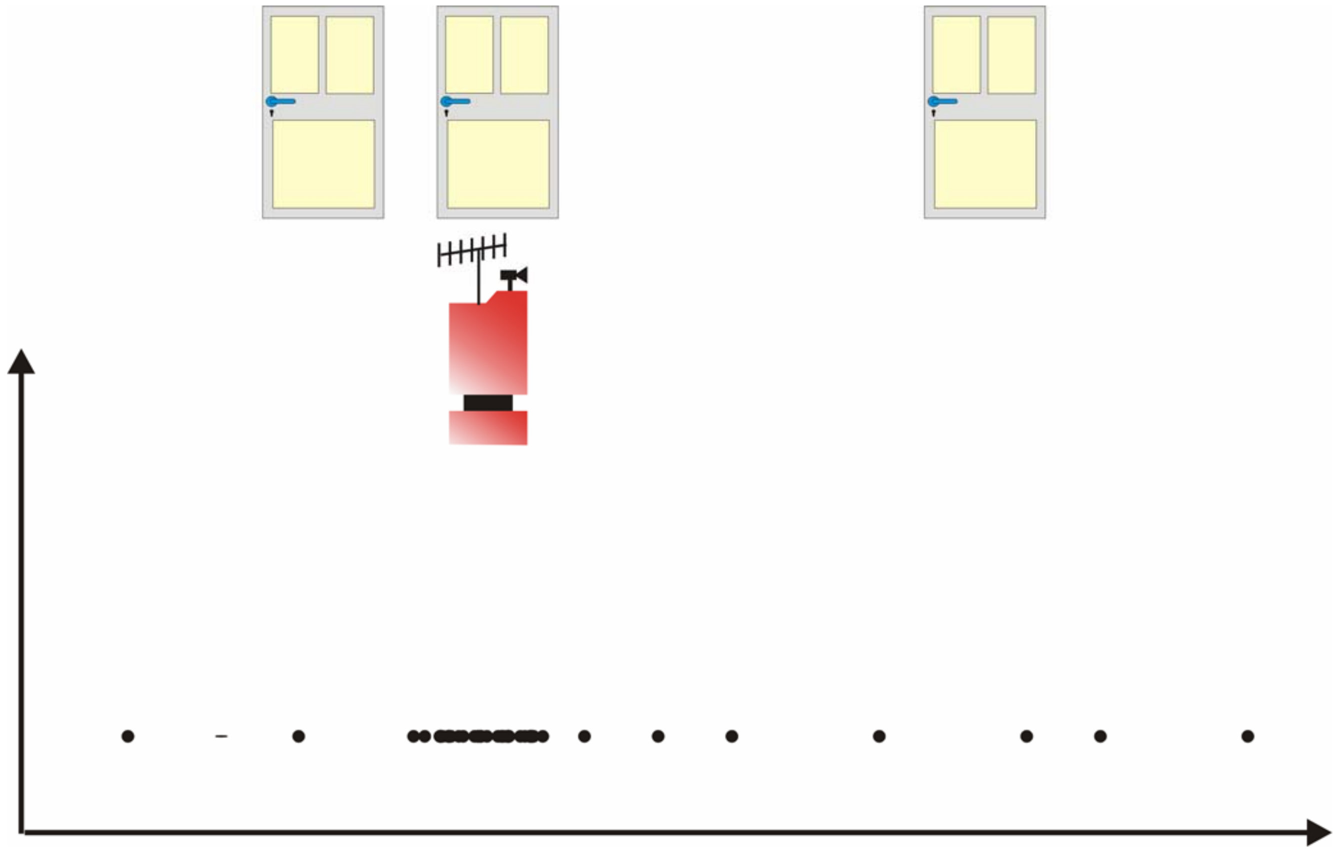
Particle Filters – Example 2



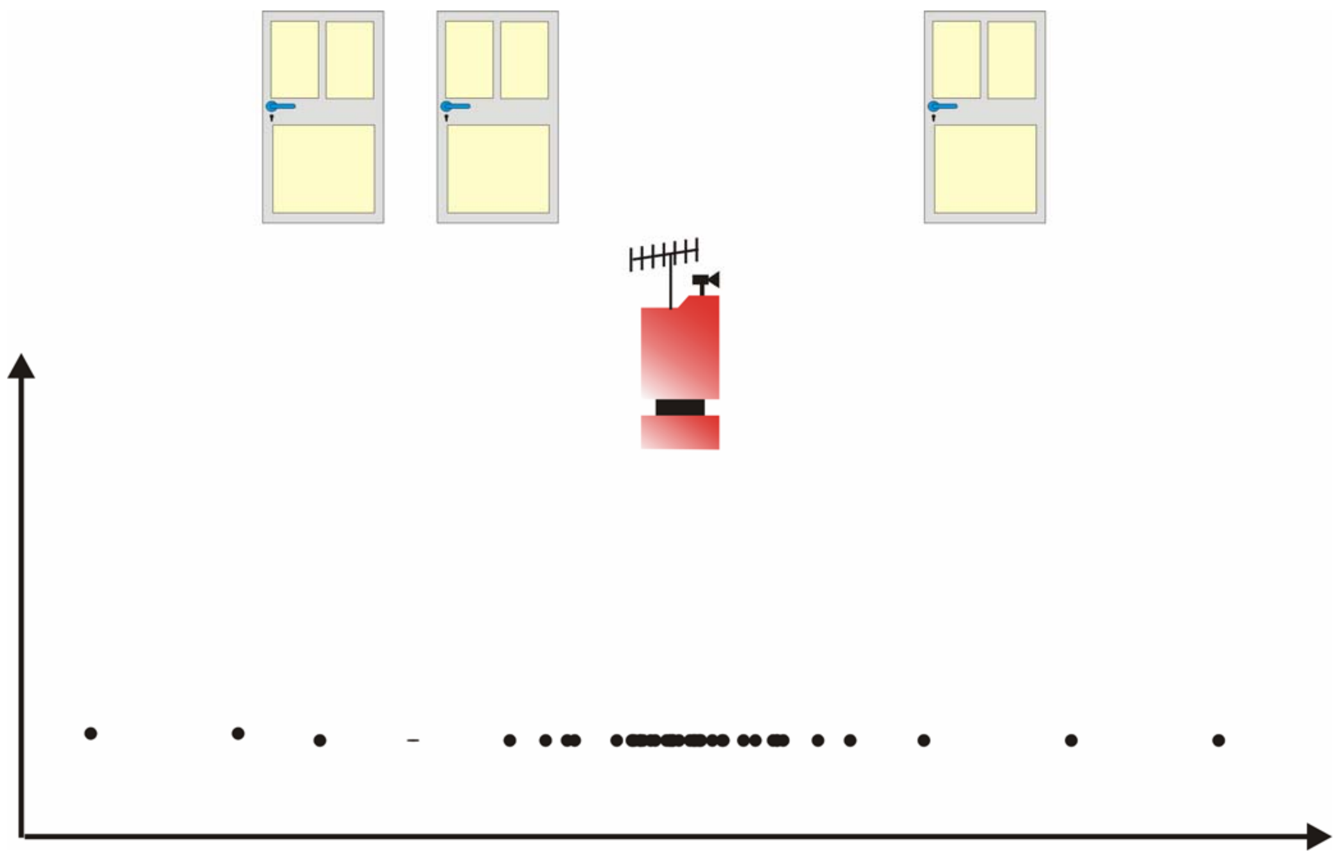
Particle Filters – Example 2



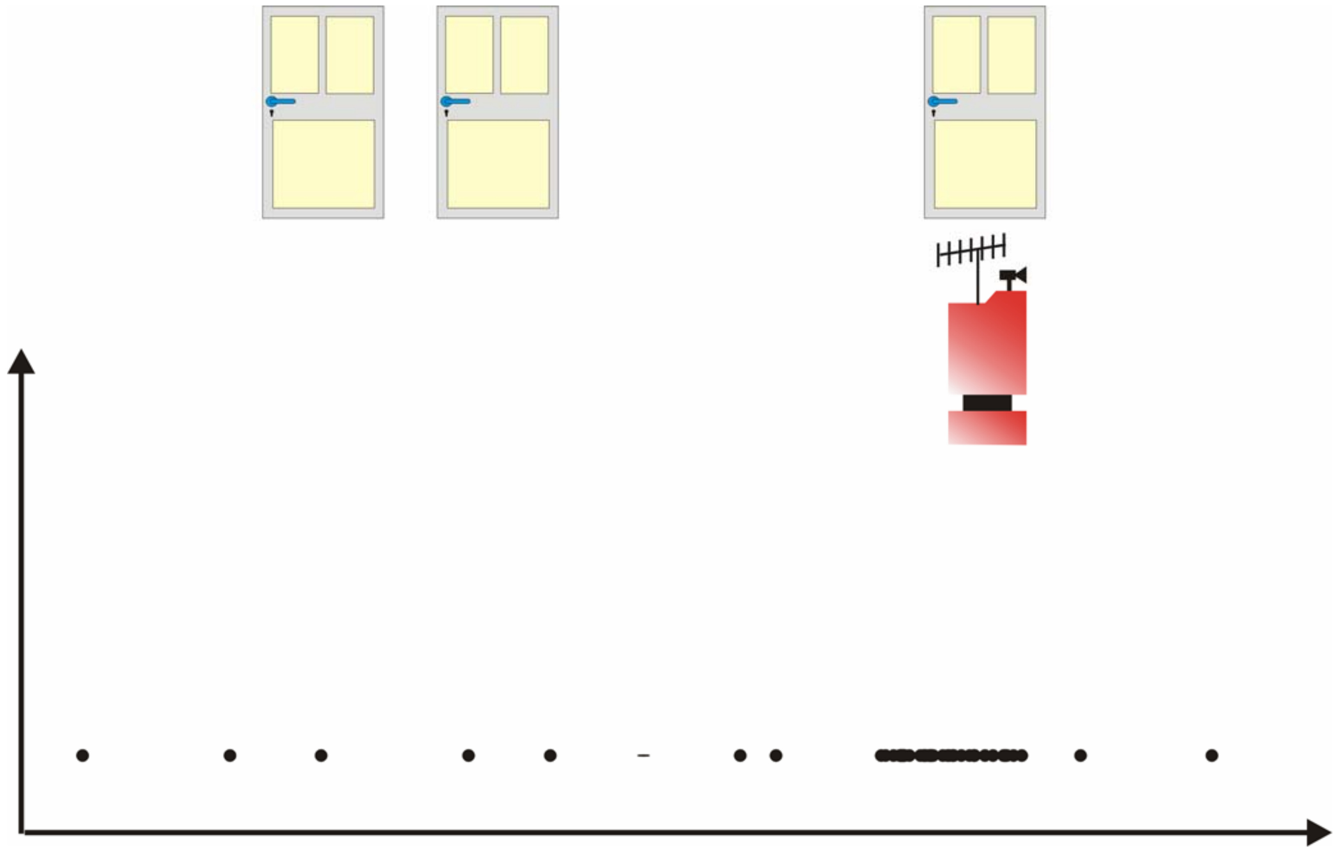
Particle Filters – Example 2



Particle Filters – Example 2



Particle Filters – Example 2



Discrete State Approaches

- Ability (to some degree) to localize the robot even when its initial pose is unknown.
- Ability to deal with noisy measurements, such as from ultrasonic sensors.
- Ability to represent ambiguities.
- Computational time scales heavily with the number of possible states (dimensionality of the grid, size of the cells, size of the map).
- Localization accuracy is limited by the size of the grid cells.

Continuous State Approaches

- Perform very accurately if the inputs are precise (performance is optimal in the linear case).
- Computational efficiency.

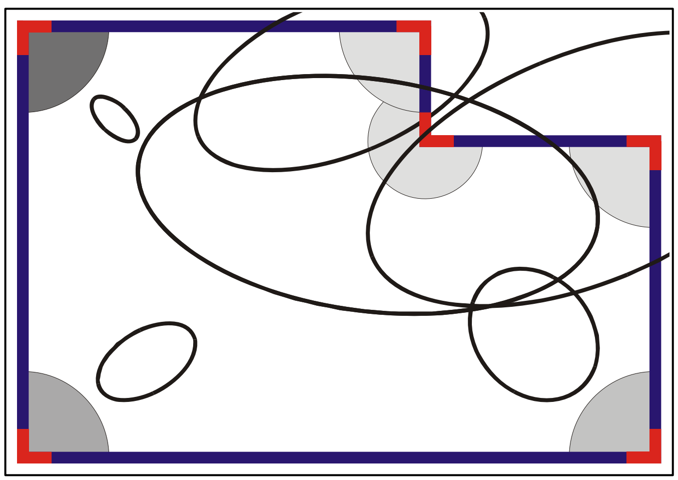
- Requirement that the initial state of the robot is known.
- Inability to recover from catastrophic failures caused by erroneous matches or incorrect error models.
- Inability to track Multiple Hypotheses about the location of the robot.

Hybrid Approaches

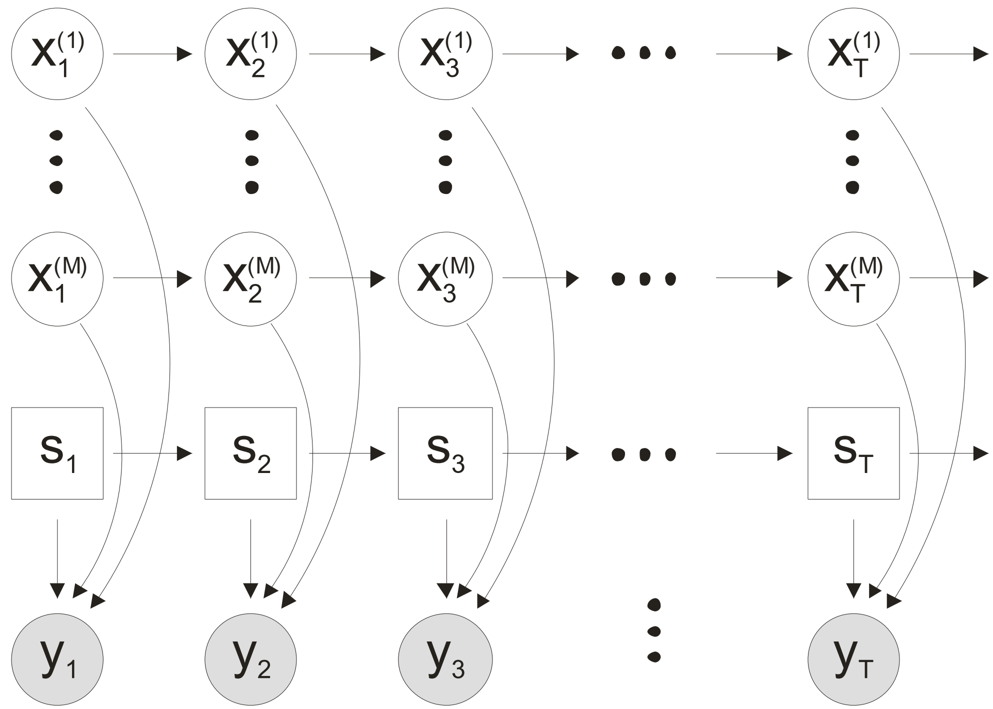
- Combination of characteristics from both methods
- Hybrid methods very popular in many scientific areas
  - Control theory
  - Economics

Proposed Model
switching state-space model (SSSM)

The Switching State-Space model
- M continuous State Vectors
- One discrete "switch" variable



Example Belief State



H. Baltzakis and P. Trahanias, Autonomous Robots 2003

Switching state-space Model

Combines both models

Inherits strengths
Eliminates weaknesses

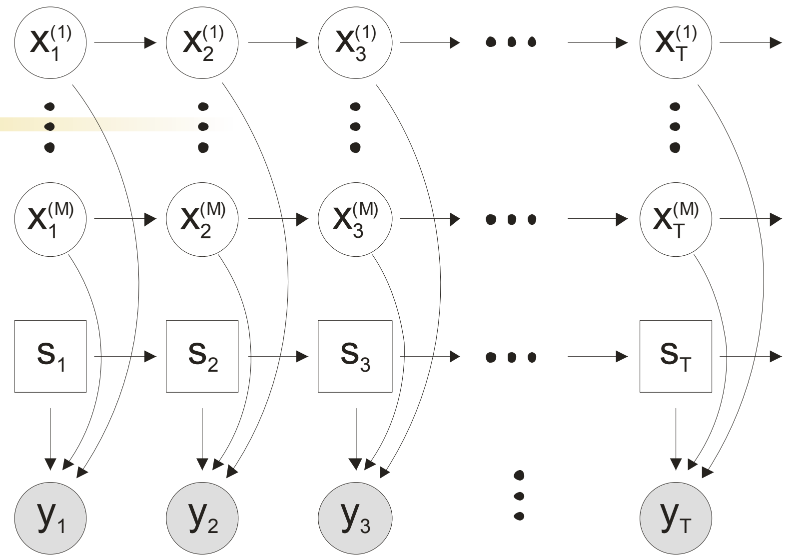
Continuous Model

- Accurate performance
- Computational efficiency
- ~~Initial state must be known~~
- ~~Inability to recover from catastrophic failures~~
- ~~Inability to track Multiple Hypotheses~~

Discrete Model

- Perform even when initial pose is unknown
- Deal with noisy measurements
- Represent ambiguities
- ~~Computational time scales heavily~~
- ~~Localization accuracy limited~~

Localization



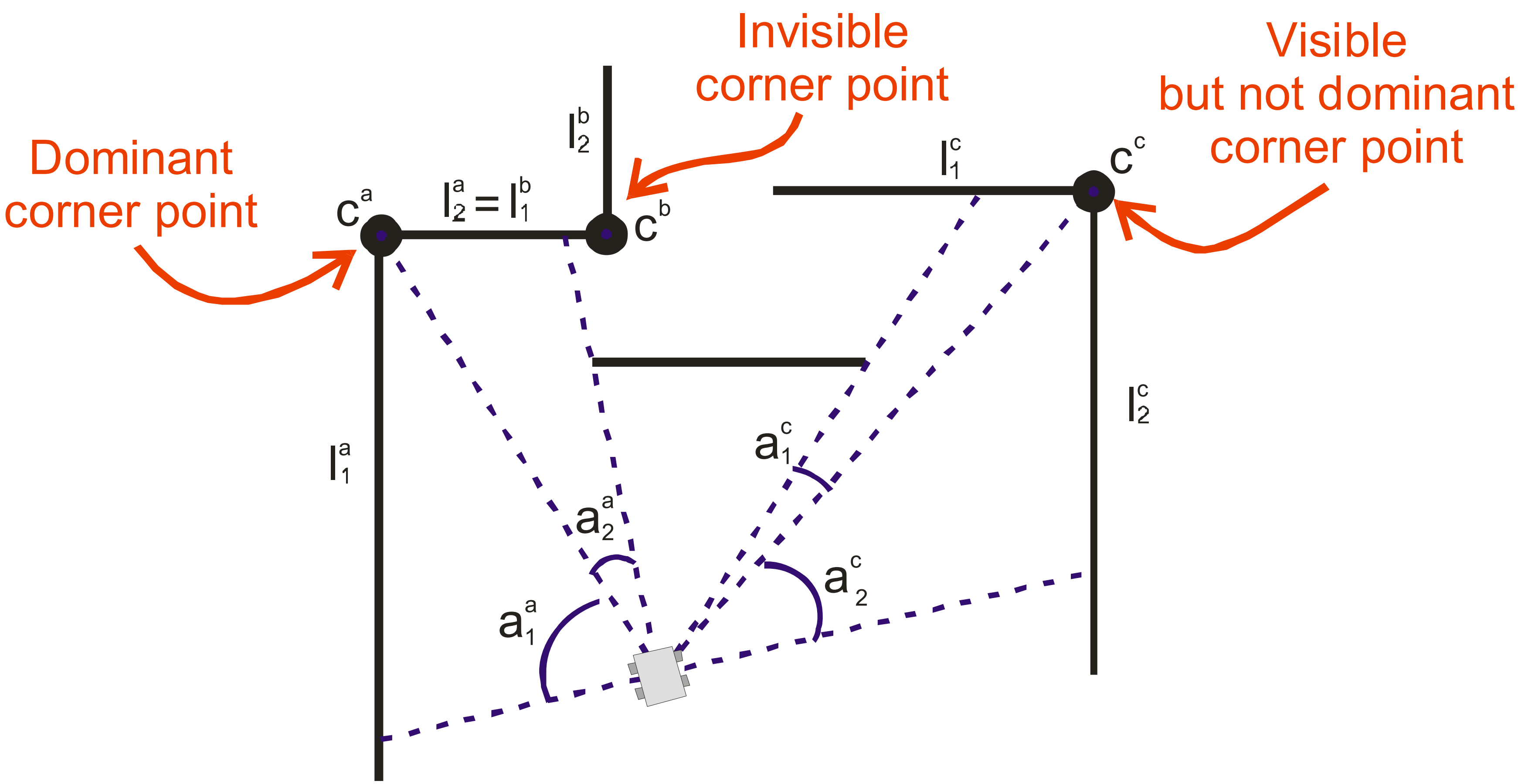
Belief state is intractable

- Mixture of M^T Gaussians
- Grows exponentially with time

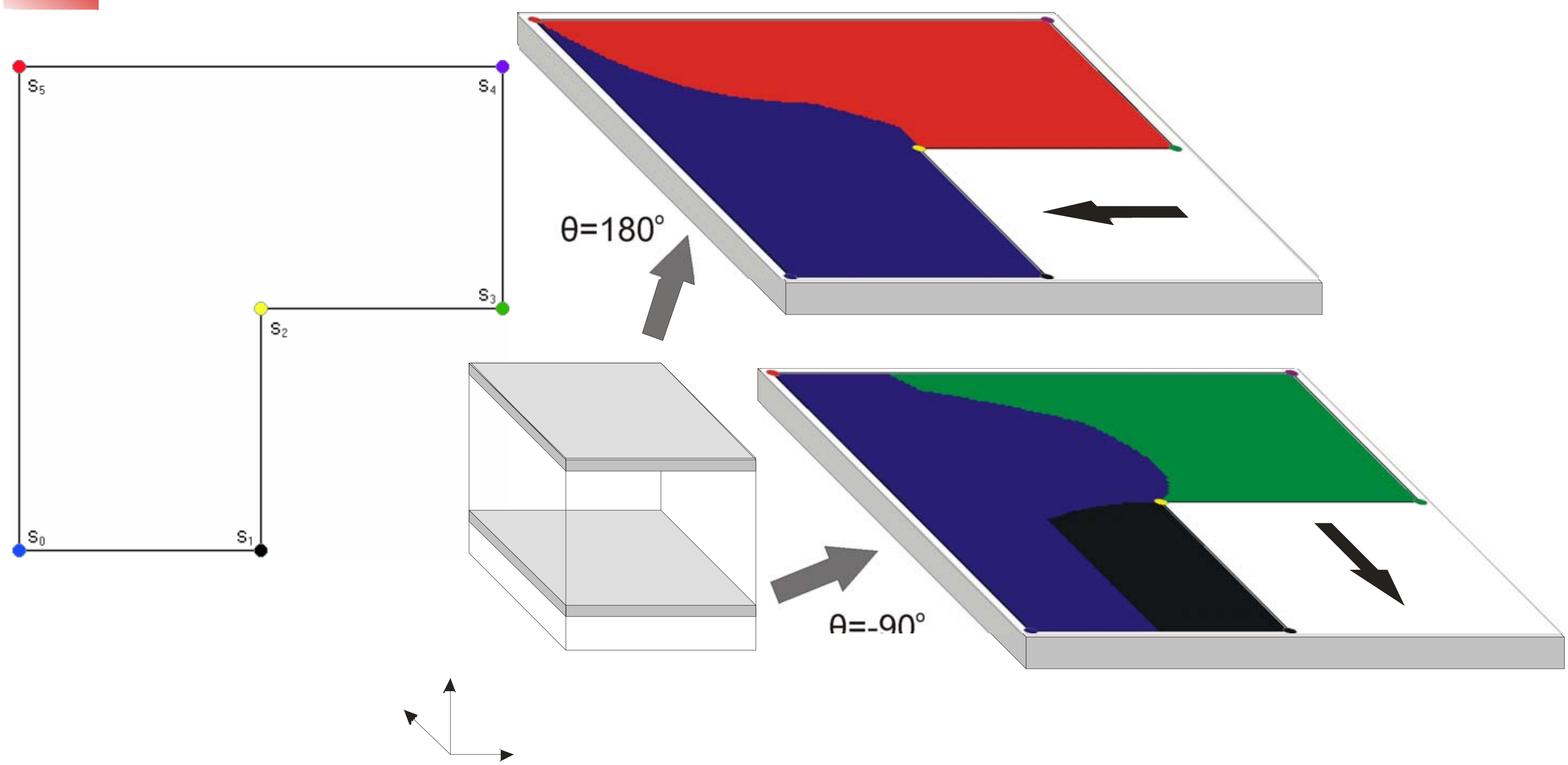
Solution

- Selection (eg. Cox94, Jensfelt99, Roumeliotis00 ,Duckett01)
  Only keep the most probable paths in model histories (Multiple Hypothesis Tracking)
- <u>Collapsing (eg. Murphy98)</u>
  Approximate the mixture of M^T Gaussians with a mixture of M^r Gaussians (r: small number, eg. 1,2,3)

Localization – Discrete Model Corner Point Visibility

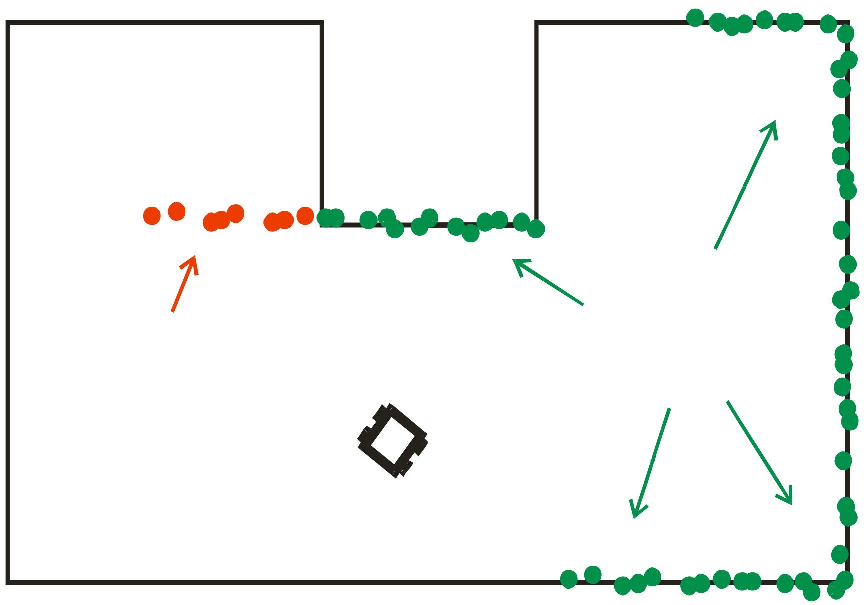


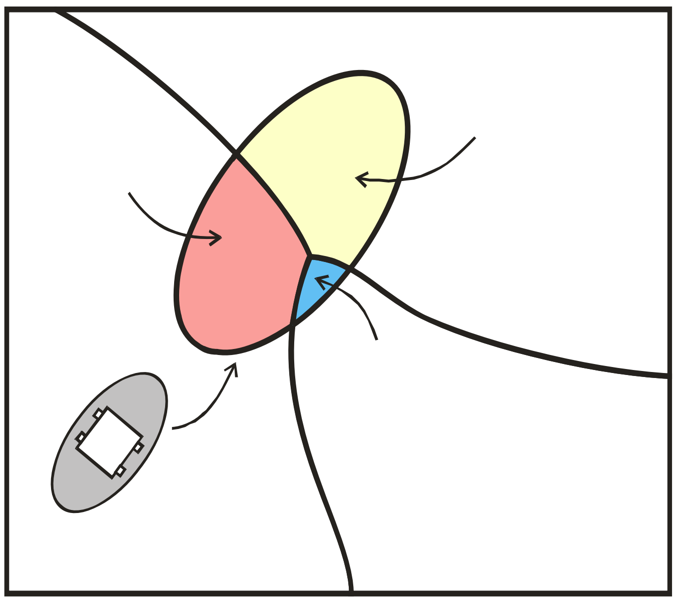
Localization – Discrete Model
Corner Point Visibility



Localization - Discrete Model (Observation – Transition)

$$P\left(s_t^k \mid y_1, y_2, \ldots, y_t\right) = \frac{1}{c_t} P\left(y_t \middle| s_t^k\right) \int_i P\left(s_t^k \middle| s_{t-1}^i\right) P\left(s_{t-1}^i \mid y_1, y_2, \ldots, y_{t-1}\right)$$

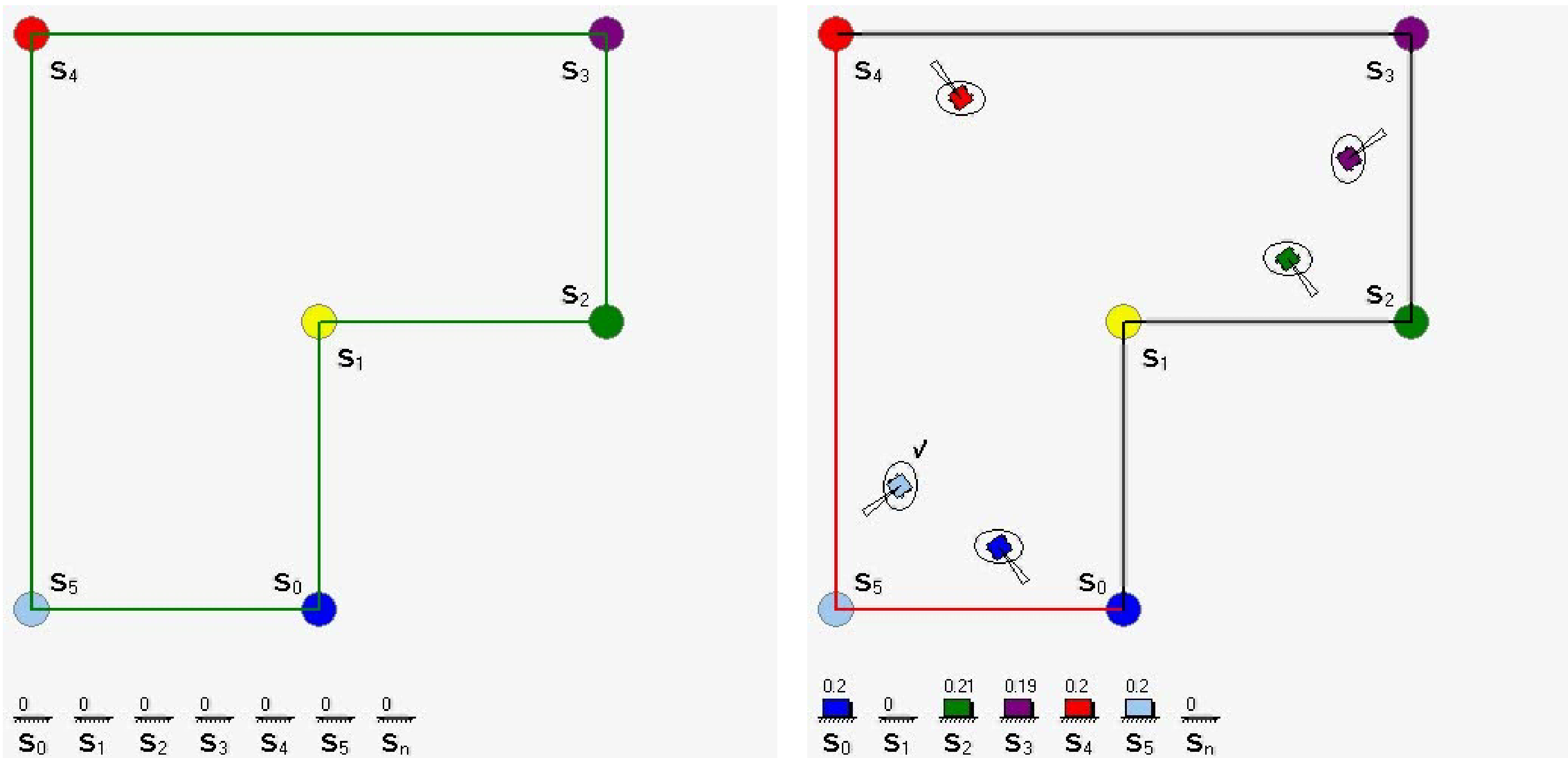




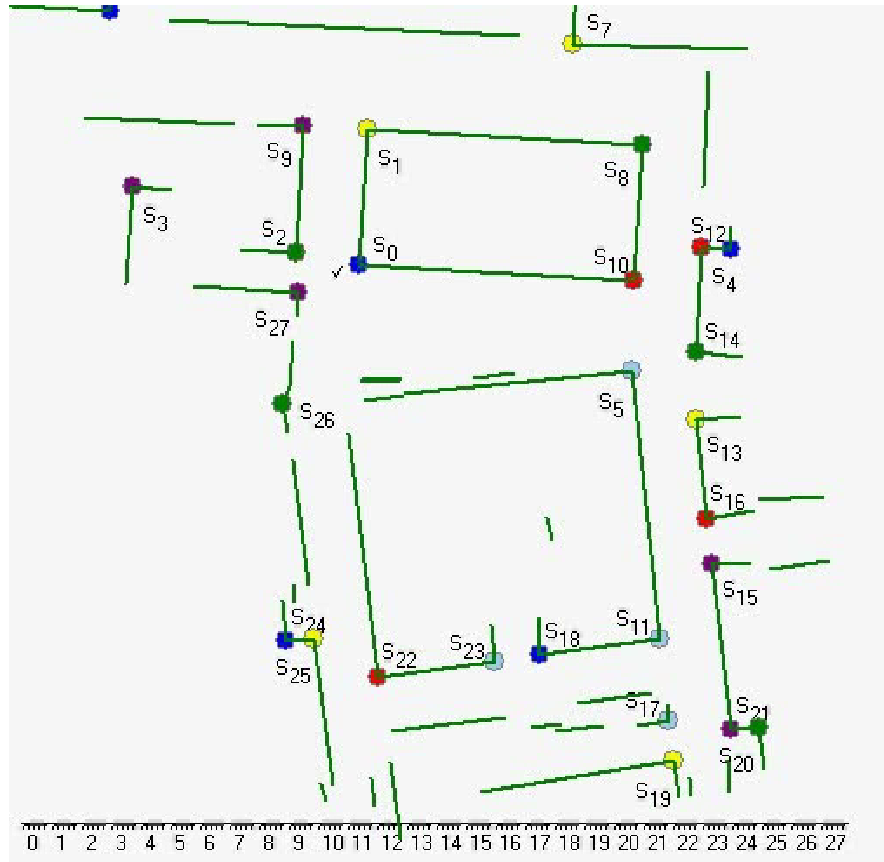
Localization – Continuous Model (EKF)

Project State estimates forward (prediction step)	$\mu_{x_{t+1}^-} = Exp(F(\mu_{x_t}, \alpha_t))$ $\Sigma_{x_{t+1}^-} = \nabla F_x \Sigma_{x_t} \nabla F_x^T + \nabla F_\alpha \Sigma_{\alpha_t} \nabla F_\alpha^T$
Predict already mapped features to the predicted state	$l_{t+1}^- = H(\mu_{x_{t+1}^-})$
Compute Kalman Innovation	$r_{t+1} = l_{t+1} - l_{t+1}^-$ $\Sigma_{r_{t+1}} = \nabla F_{x_{t+1}^-} \Sigma_{x_{t+1}^-} \nabla F_{x_{t+1}^-}^T + \Sigma_{l_{t+1}}$
Compute Kalman Gain	$K_{t+1} = \Sigma_{x_{t+1}^-} \nabla F_{x_{t+1}^-} \Sigma_{r_{t+1}^-}^{-1}$
Update Initial Prediction	$\mu_{x_{t+1}} = \mu_{x_{t+1}^-} + K_{t+1} r_{t+1}$ $\Sigma_{x_{t+1}} = \Sigma_{x_{t+1}^-} - K_{t+1} \Sigma_{r_{t+1}} K_{t+1}^T$

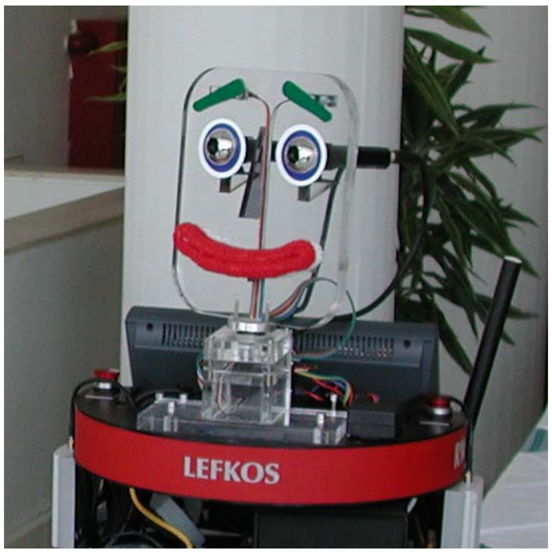
Localization - Results
(Simulated)

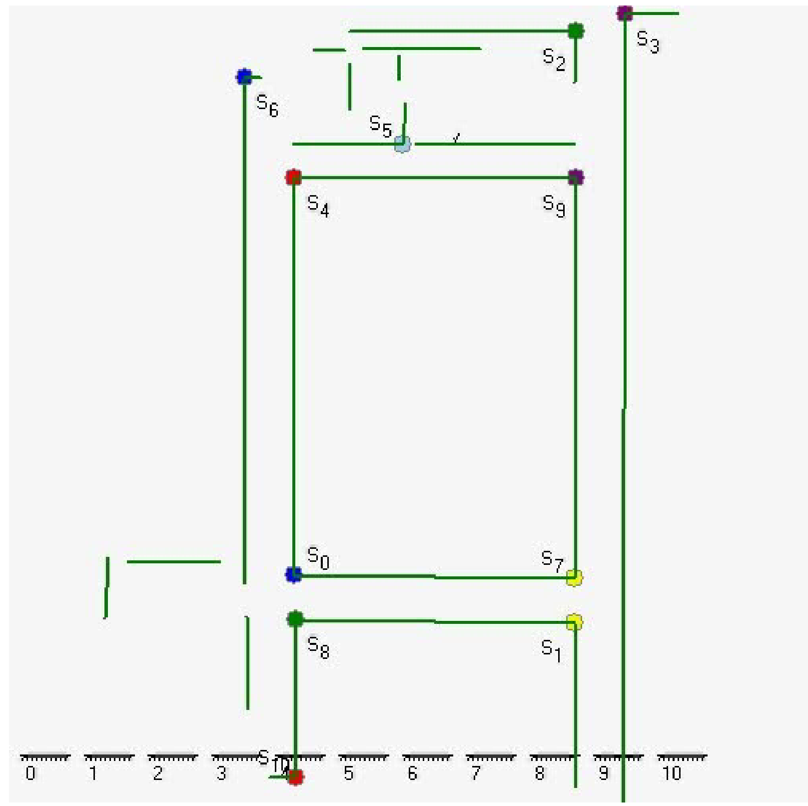


Localization - Results
(Real world – FORTH 1st floor)

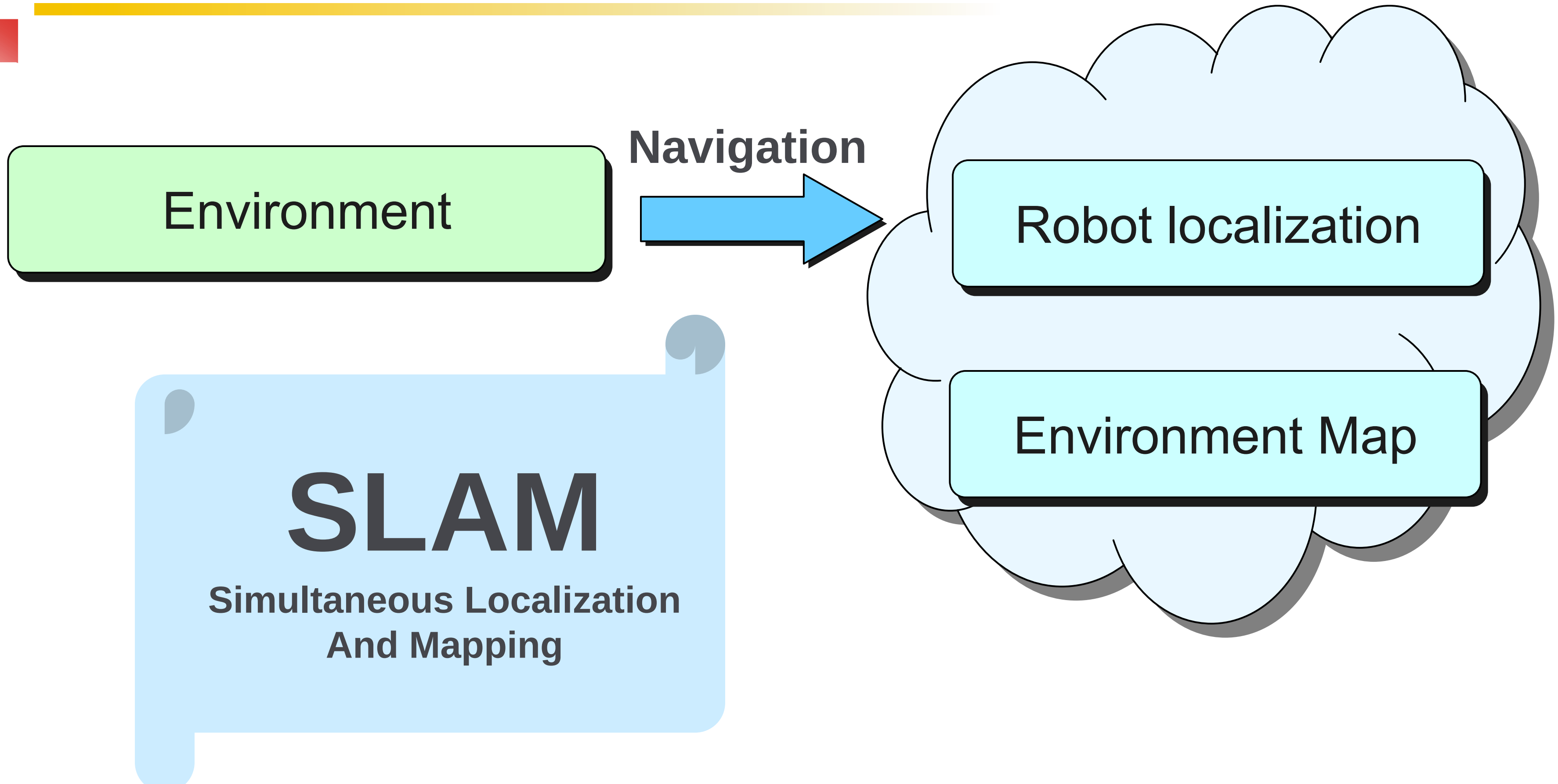


Localization - Results
(Real world – Outside our lab)





Mapping
Problem Statement



Mapping – Kalman Tracker

- Simultaneously estimate the robot position as well as the positions of landmarks (stochastic mapping)
  - Augment state vector to also include landmark positions

$$x = \begin{bmatrix} x_r & y_r & x_{l1} & y_{l1} & x_{l2} & y_{l2} & \cdots & x_{\ln_l} & y_{\ln_l} \end{bmatrix}^T$$

Mapping – Kalman Tracker

$$x(k+1) = x(k) + \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix} \begin{pmatrix} u_x(k) \\ u_y(k) \end{pmatrix} + \begin{pmatrix} v_{rx}(k) \\ v_{ry}(k) \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$x(k+1) = Fx(k) + Gu(k) + v(k)$$

Mapping – Discrete Bayesian Approach

- Recursive Bayesian filtering for estimating the robot positions along with a map of the environment

$$P(x(1:k), m \mid u(0:k-1), y(1:k))$$
$$= aP(y(k) \mid x(k), m) \int A \bullet B \bullet dx(1:k-1)$$

$$A = (P(x(k) \mid u(k-1), x(k-1))$$
$$B = P(x(1:k-1), m \mid u(0:k-2), y(1:k-1)))$$

Mapping – Discrete Bayesian Approach

- Estimating the full posterior is not tracktable
- Incremental scan matching

- Let at time k-1 the localization and map estimates:

$$\hat{x}(k-1) \qquad \hat{m}(\hat{x}(1:k-1), y(1:k-1))$$

- At time k – after moving and getting a new measurement y(k)

$$\hat{x}(k) = \arg\max_{x(k)} \left\{ P(y(k) \mid x(k), \hat{m}(\hat{x}(1:k-1), y(1:k-1))) P(x(k) \mid u(k-1), \hat{x}(k-1)) \right\}$$

Mapping – Discrete Bayesian Approach

- $\boxed{\text{Estimating the full posterior is not tracktable}}$
- FastSLAM

$$P(x(1:k), m \mid u(0:k-1), y(1:k))$$
$$= P(m \mid u(0:k-1), y(1:k)) \bullet P(x(1:k) \mid u(0:k-1), y(1:k))$$

- Usually implemented via particle filters

Mapping Challenge: Loops in the robot’s path

- As the robot moves and maps features, errors in both the state and the mapped features tend to increase with time
- When already mapped areas are visited (a loop is detected), the mapping algorithm should be able to correct its state and eliminate the accumulated errors

- Complicated robot paths, nested loops or loops that close simultaneously are difficult cases.

Our approach

Off-line Feature-mapping algorithm:

Loop detection is accomplished via a hybrid localizer with global localization capabilities (SSSM) that creates hypotheses whenever known areas (corner points) are visited

- All hypotheses created by the localizer, whenever loops are detected, are tracked individually within their own copy of the map.
- The best path through hypotheses histories is selected, a Kalman smoother redistributes errors and an iterative procedure corrects the map

H. Baltzakis and P. Trahanias, ICRA 2006

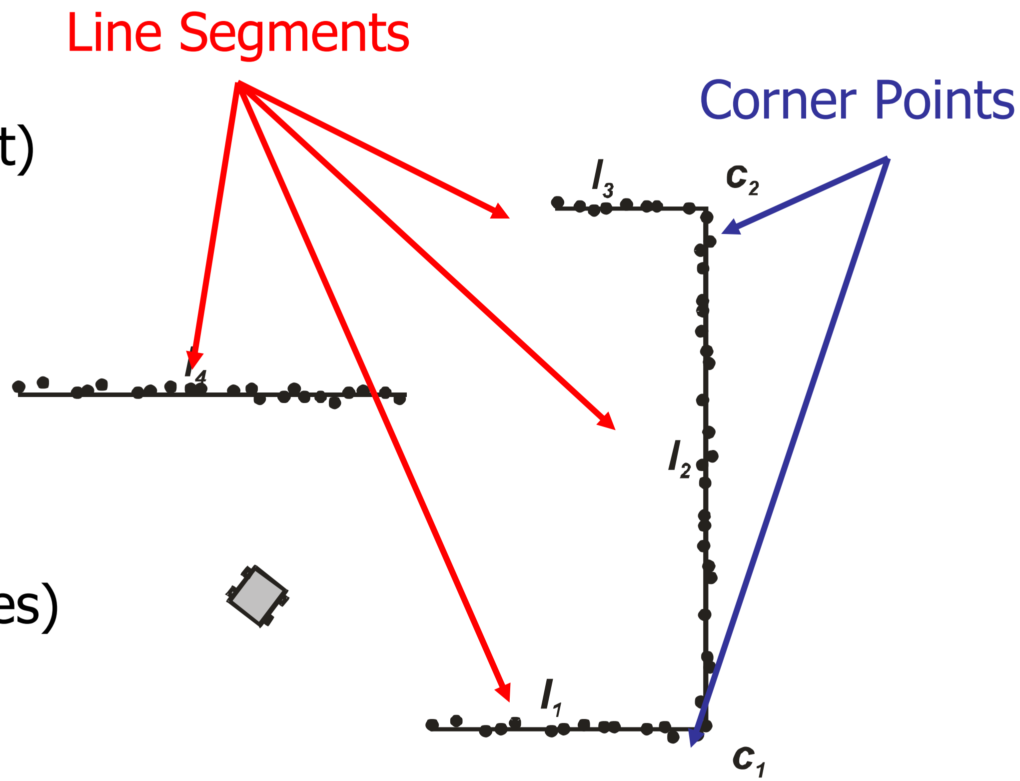
Features

- **Line Segments**
  (used to localize the robot)

$$l \approx N\left((l_f, l_d), \Sigma_f\right)$$

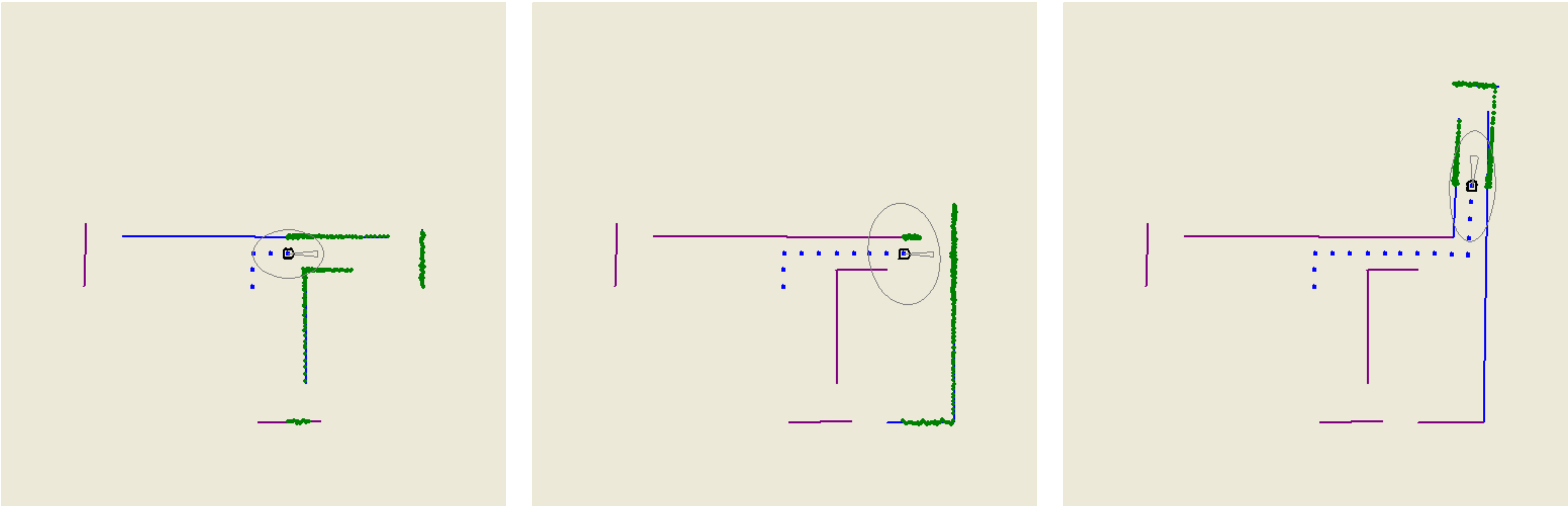
- **Corner Points**
  (used to create hypotheses)

$$c \approx N\left((c_x, c_y), \Sigma_c\right)$$



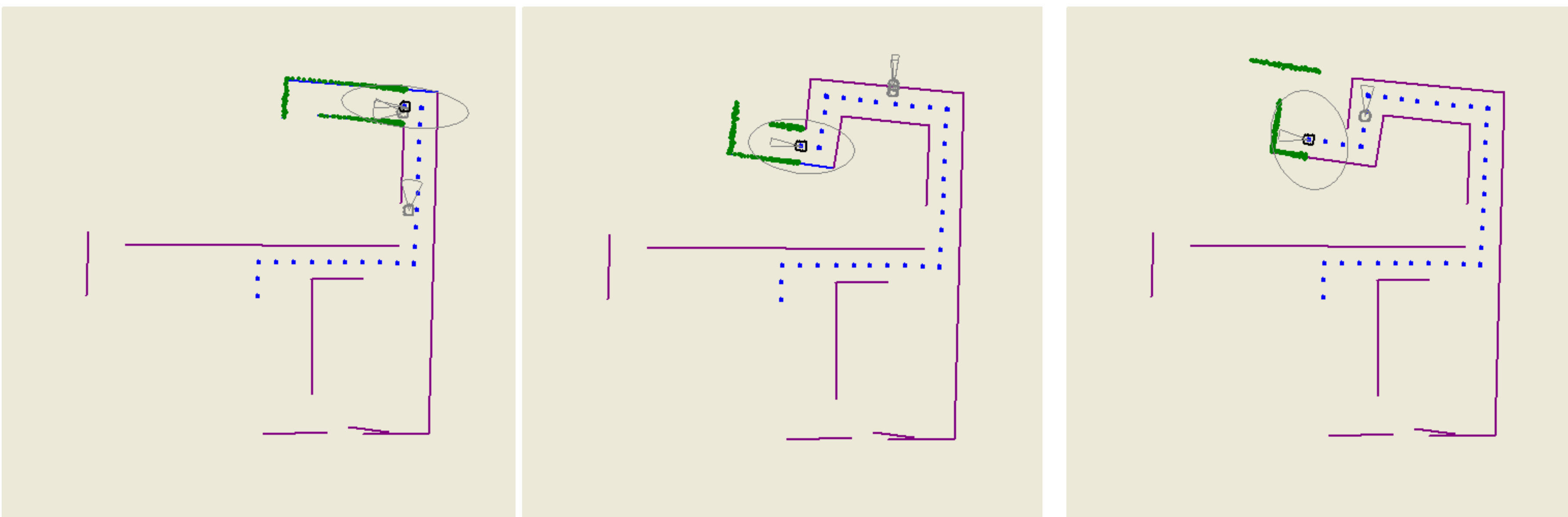
Algorithm overview 1

- Mapping starts with one hypothesis (dominant)
- Existing line segments used for localization while the map is created.
- Non existing segments are inserted in the map



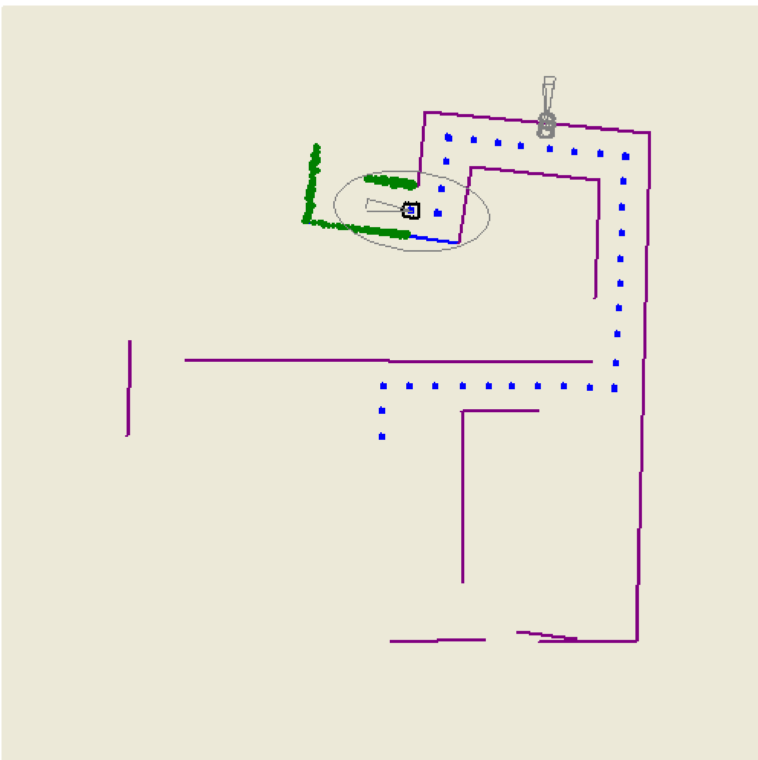
Algorithm overview 2

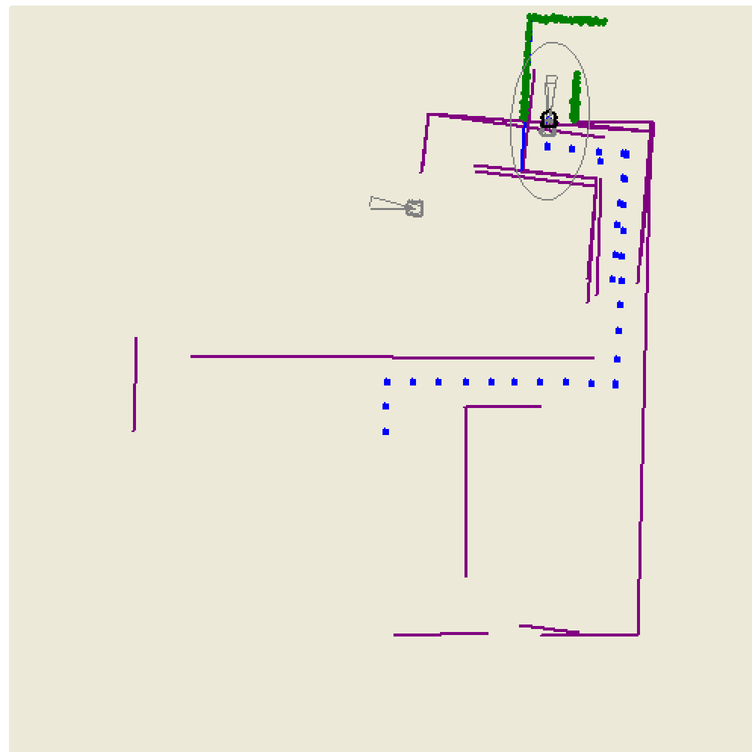
- Detected corner points result in creation of new hypotheses
- Hypotheses eventually vanish if observation sequences do not confirm their validity



Algorithm overview 3

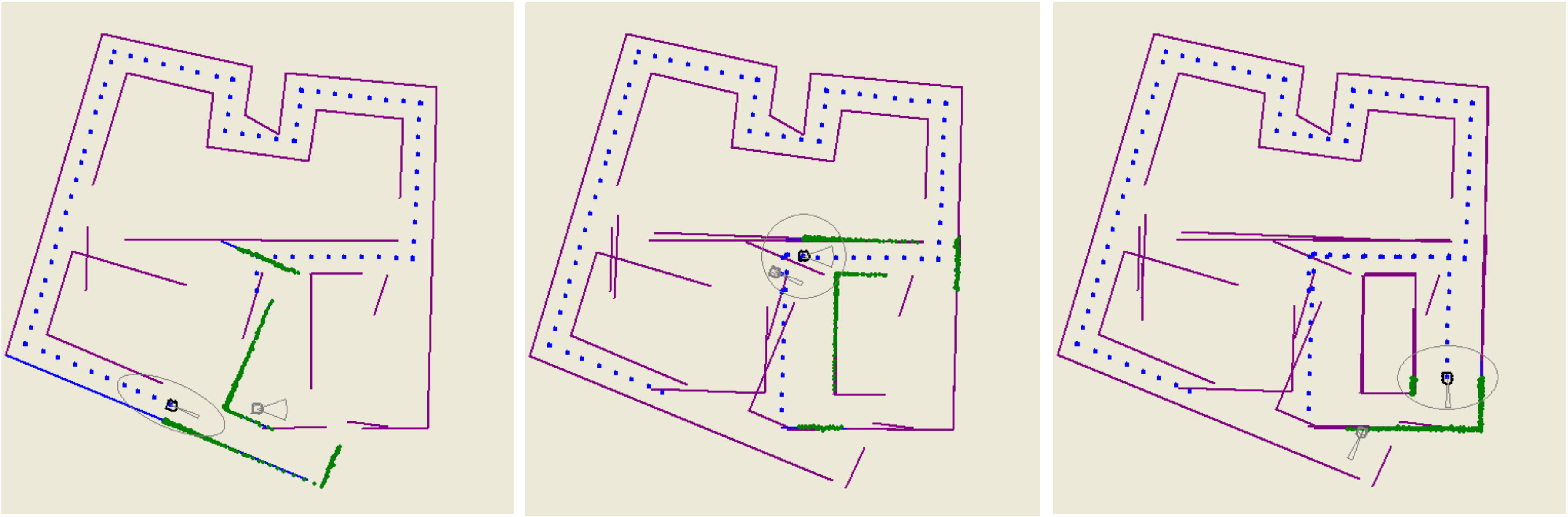
- Hypotheses maintain their own copy of the map (as many maps as hypotheses)





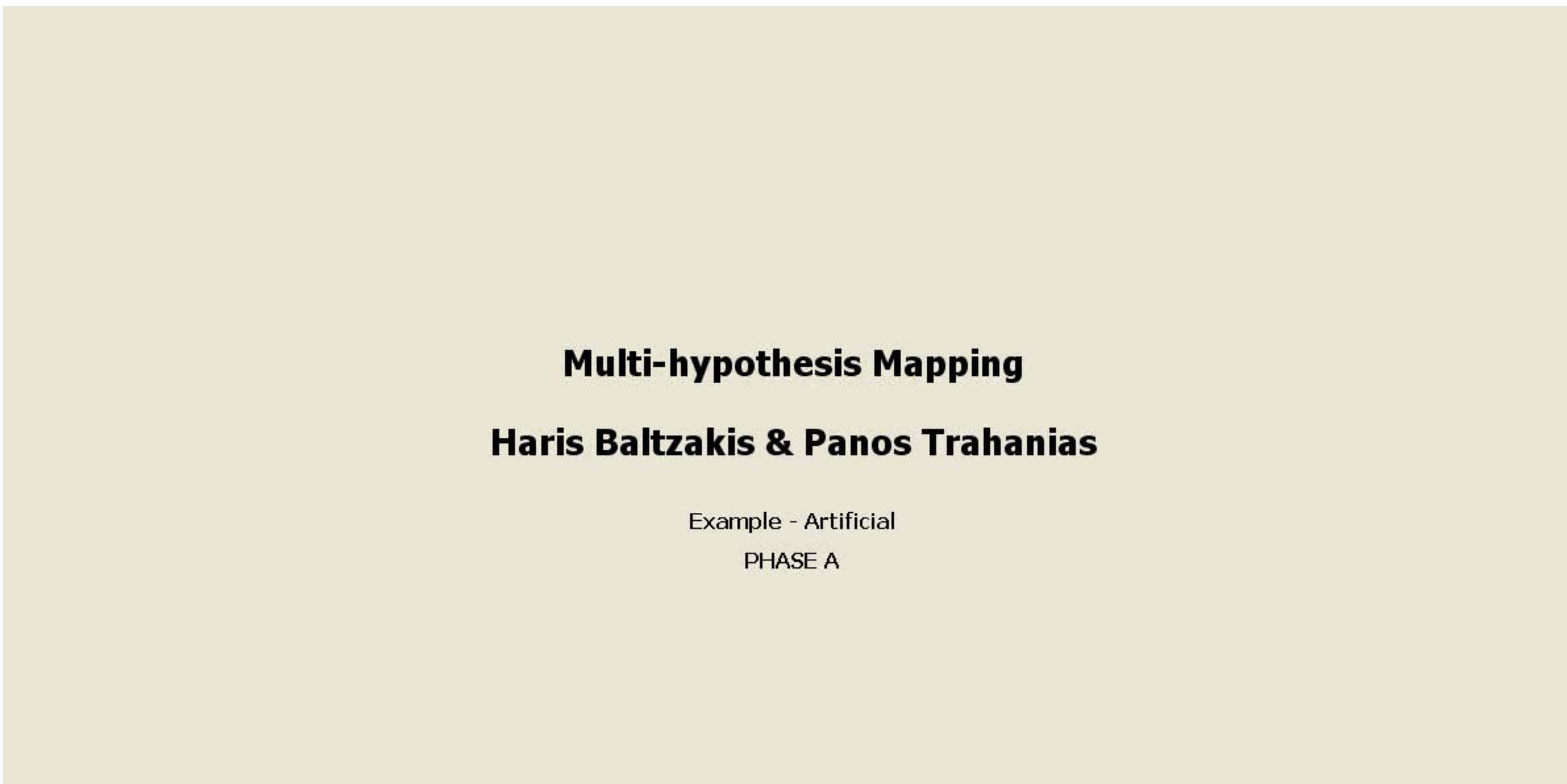
Algorithm overview 4

- Upon entering previously mapped areas (corners detected), new hypotheses are created at the correct robot poses
- Correct hypotheses will eventually become more probable since observations confirm their validity



Algorithm overview 5

- All hypotheses are tracked within their own copy of the map.

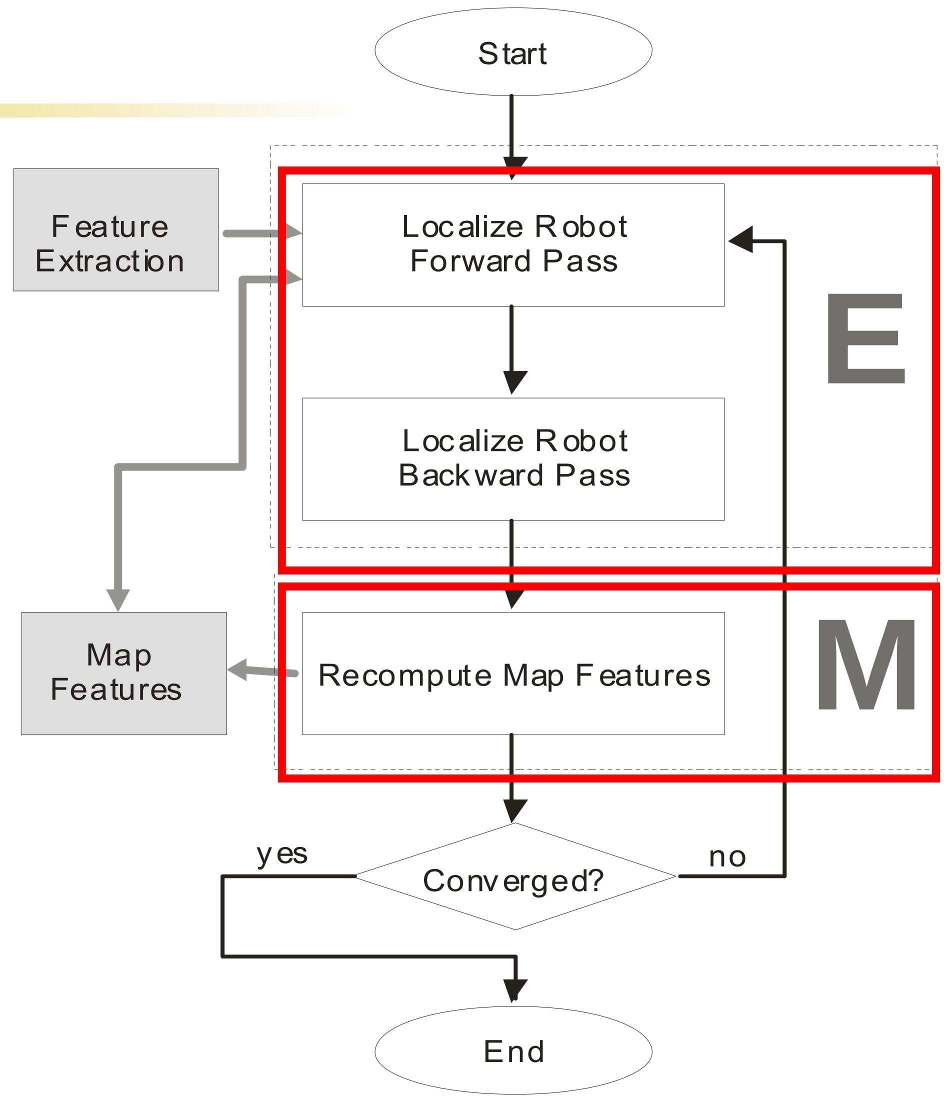


Map Rectification - Iterative Algorithm

Treat map features as parameters of the dynamical system according to which the robot's state evolves

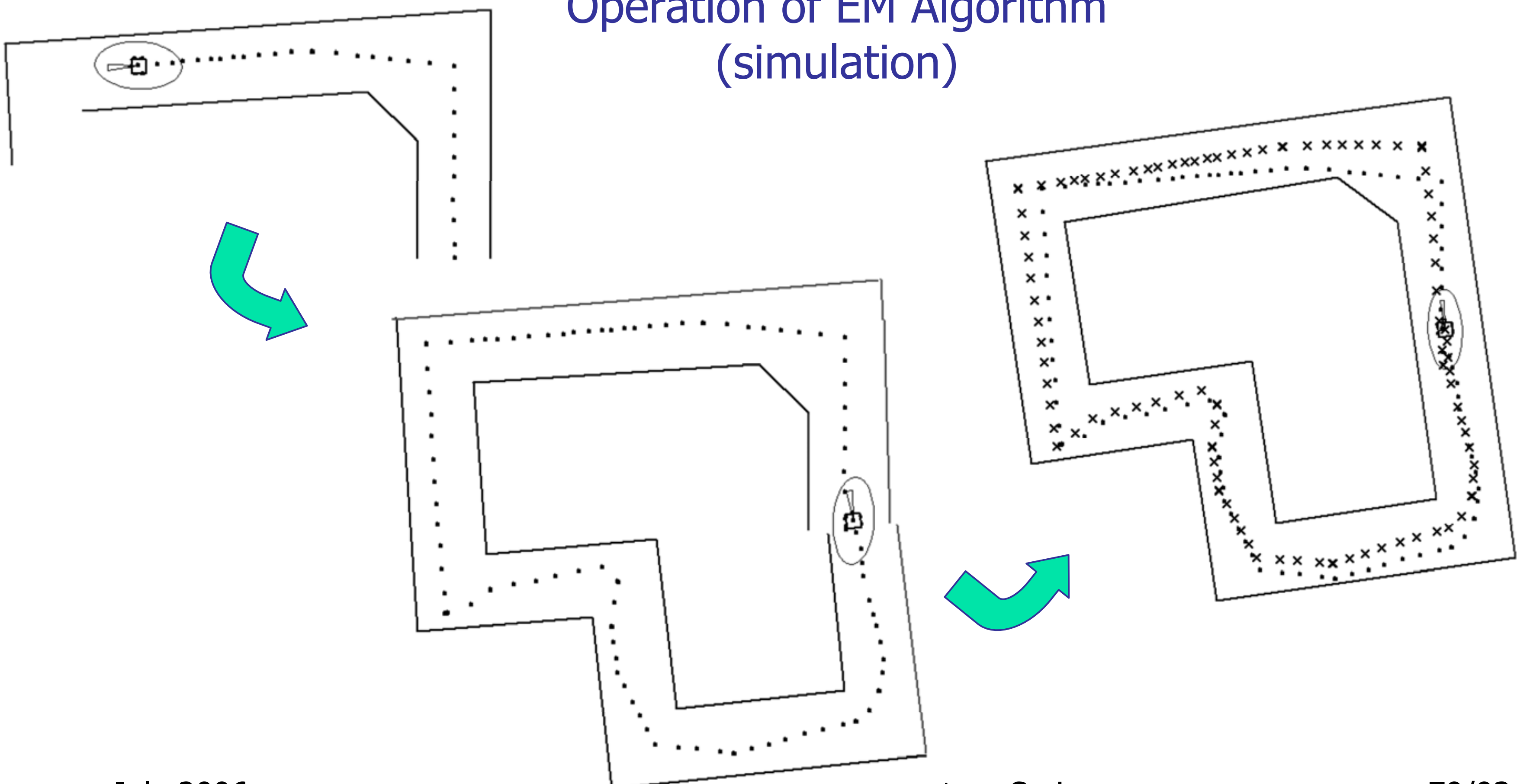
- E-STEP: Localize the robot using all available measurements. (obtain max a-posteriori estimates of robot states)

- M-STEP: Recalculate map features



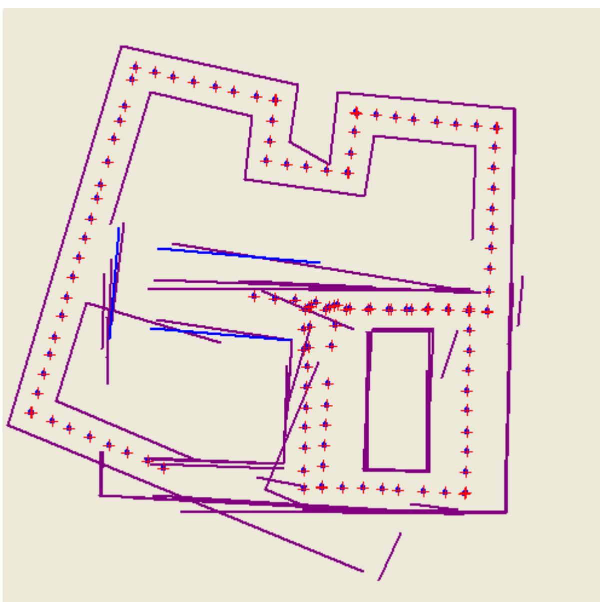
EM Algorithm - Example

Operation of EM Algorithm
(simulation)



Results (simulated running example)

Initial map



Multi-hypothesis Mapping

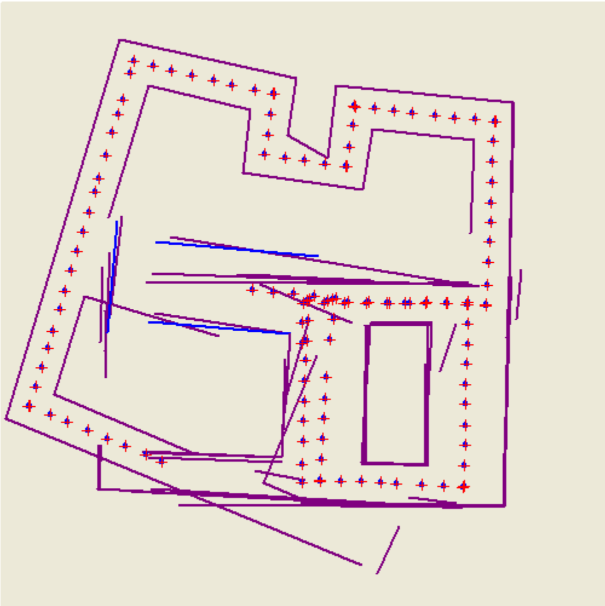
Haris Baltzakis & Panos Trahanias

Example - Artificial

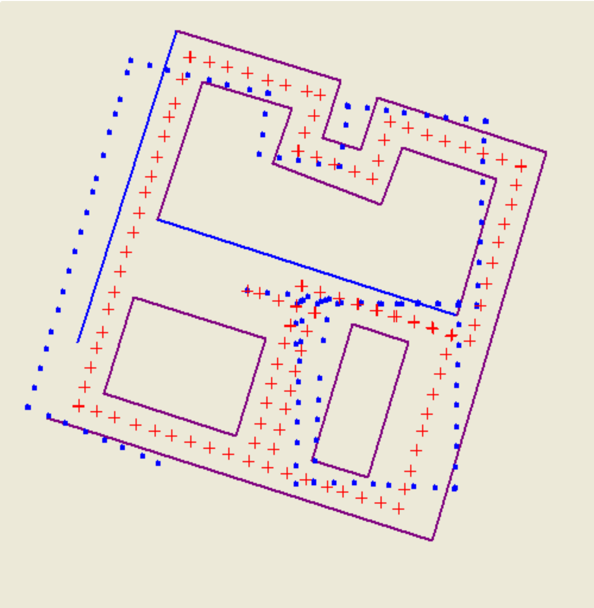
PHASE B

Results (simulated running example)

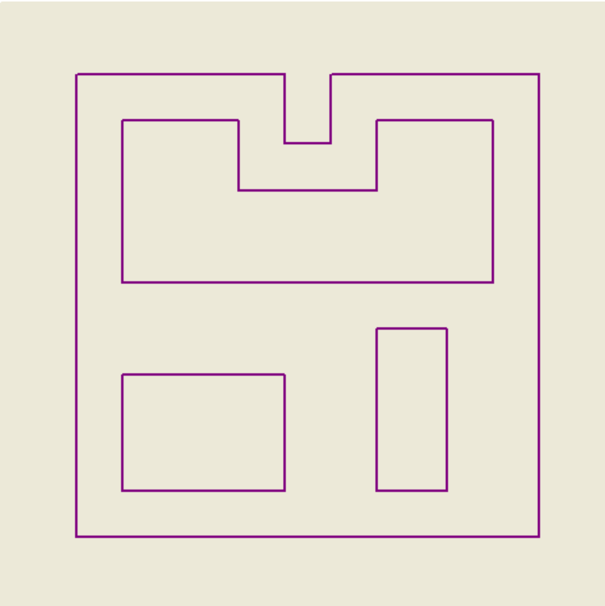
Initial Map



Rectified Map

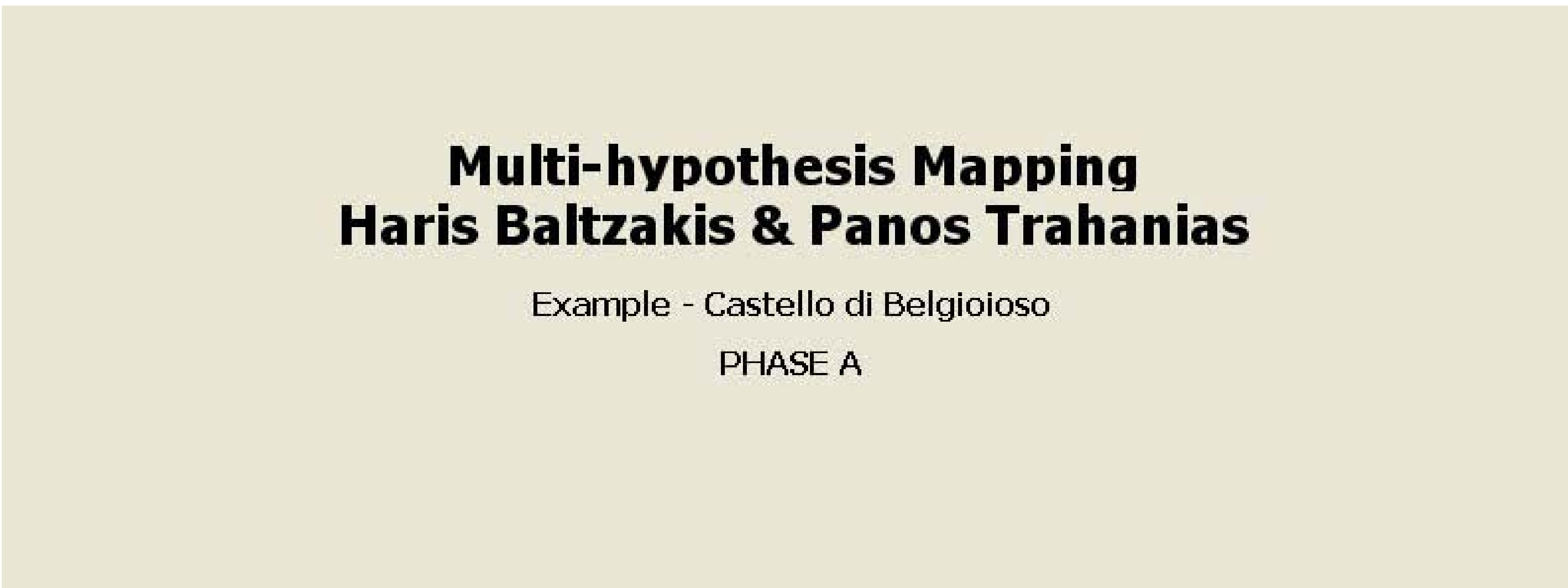


Ground Truth



Results (Di Castello Belgioioso)

Phase A



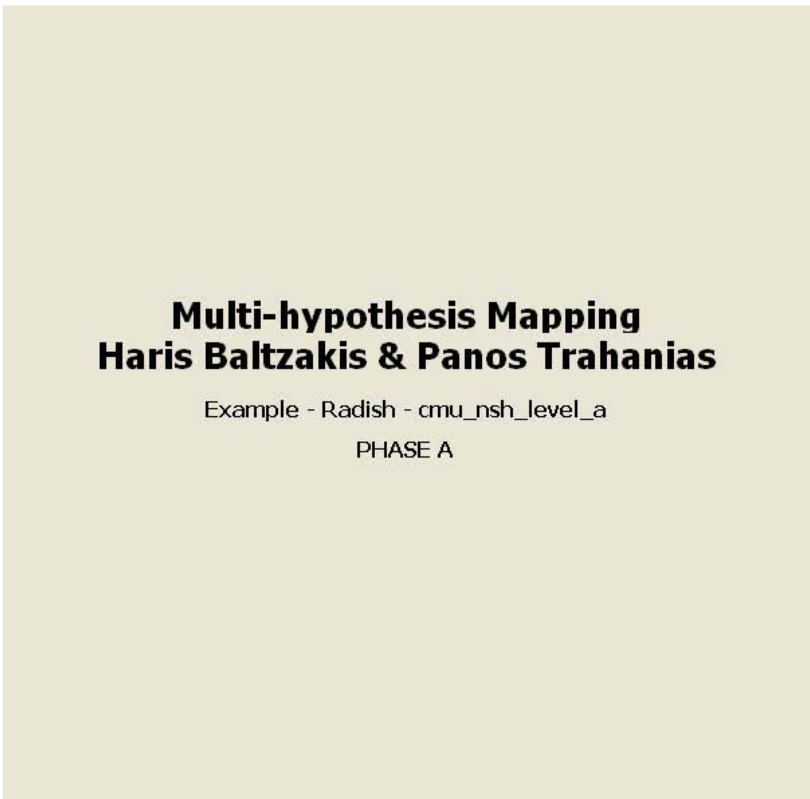
Belgioioso dataset available from university of Freiburg

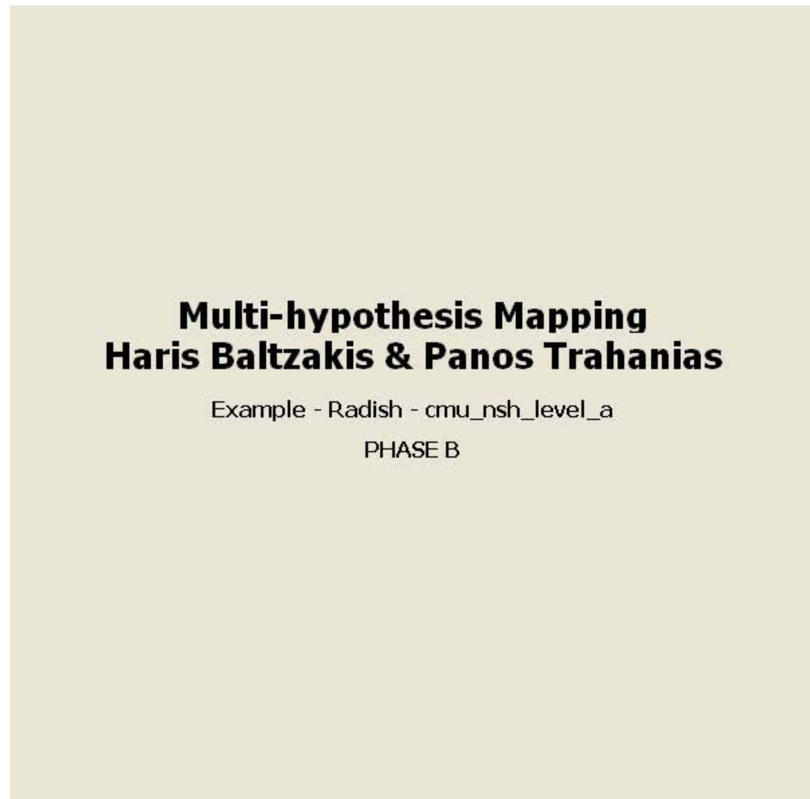
Results (Di Castello Belgioioso)

Phase B

Belgioioso dataset available from university of Freiburg

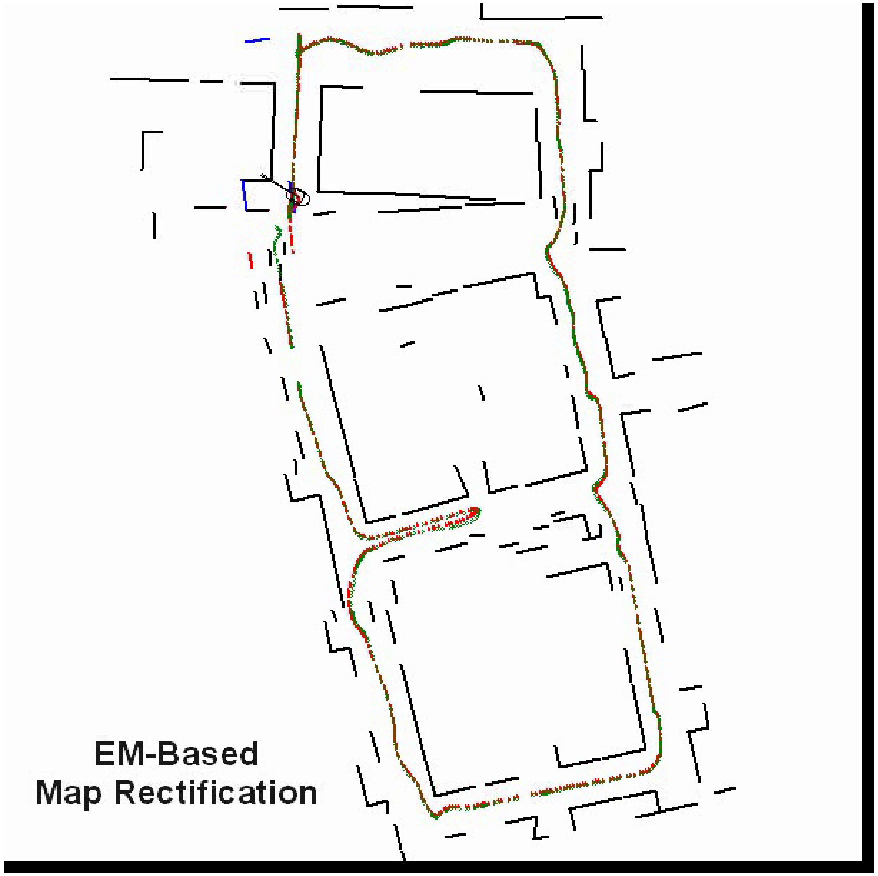
Results (Radish - cmu_nsh_level_a)





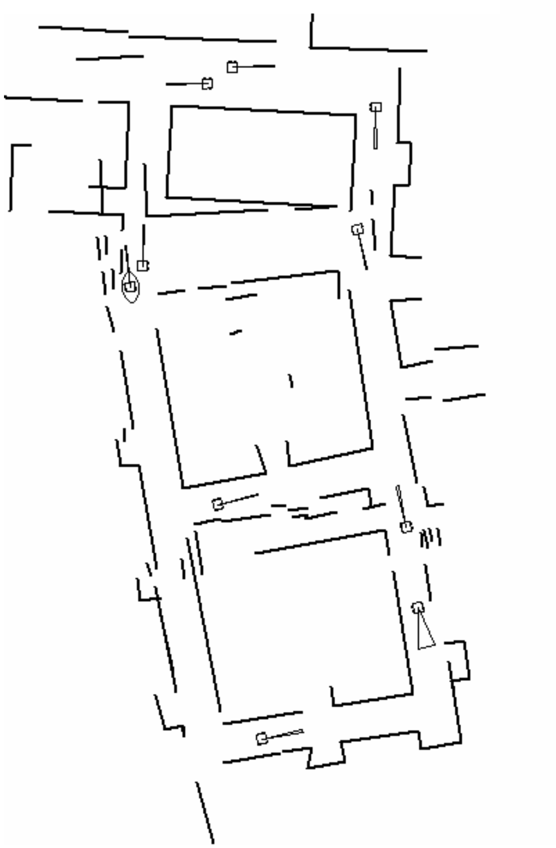
Radish cmu_nsh_level_a data set submitted by Nick Roy

Mapping - Results
(Real world – FORTH 1st floor)

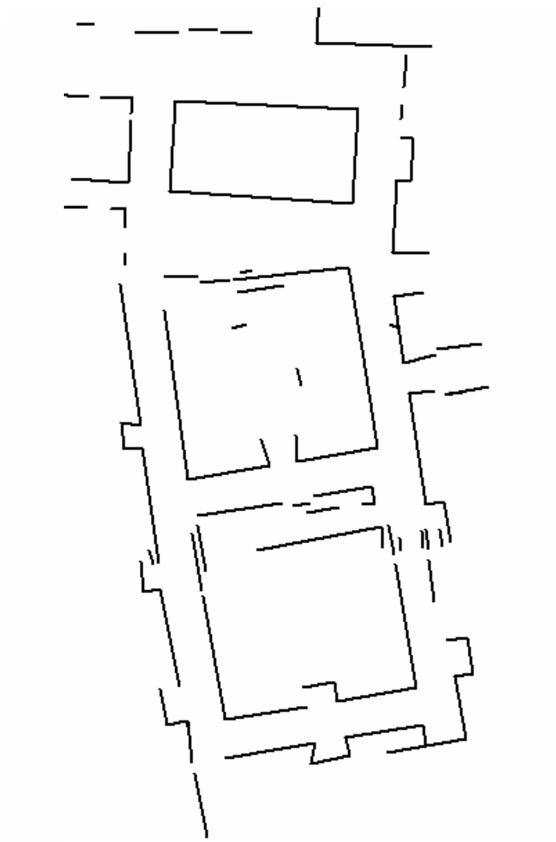


Mapping - Results
(Real world – FORTH 1st FLOOR)

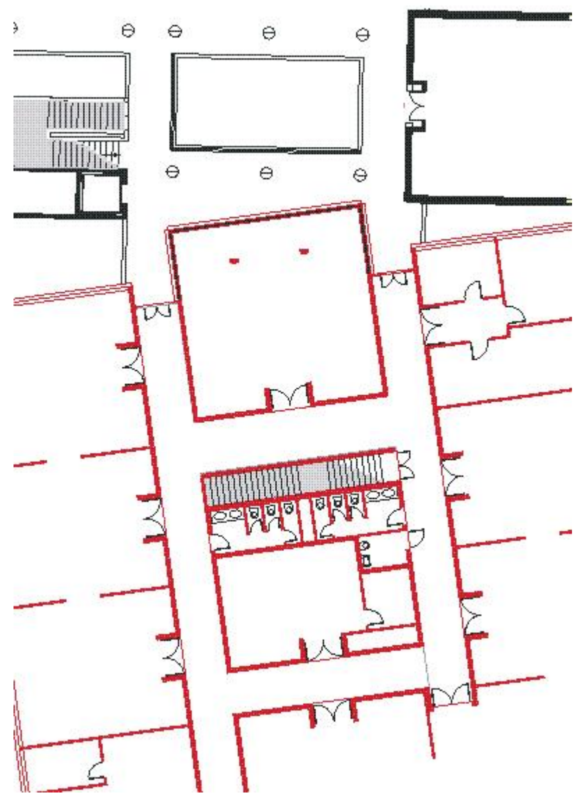
Initial Map



Rectified Map

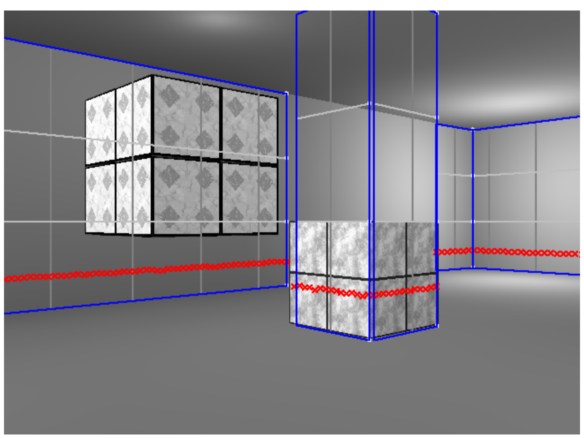


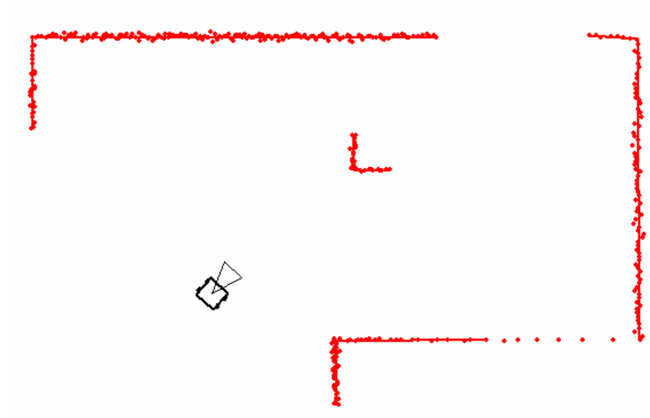
Ground Truth



Visual Information Processing

- Laser range finders provide fast and accurate depth information for 2D slices of the environment
- Various objects are invisible to the laser range finder.
- Vision can provide extra information for crucial tasks such as obstacle avoidance.



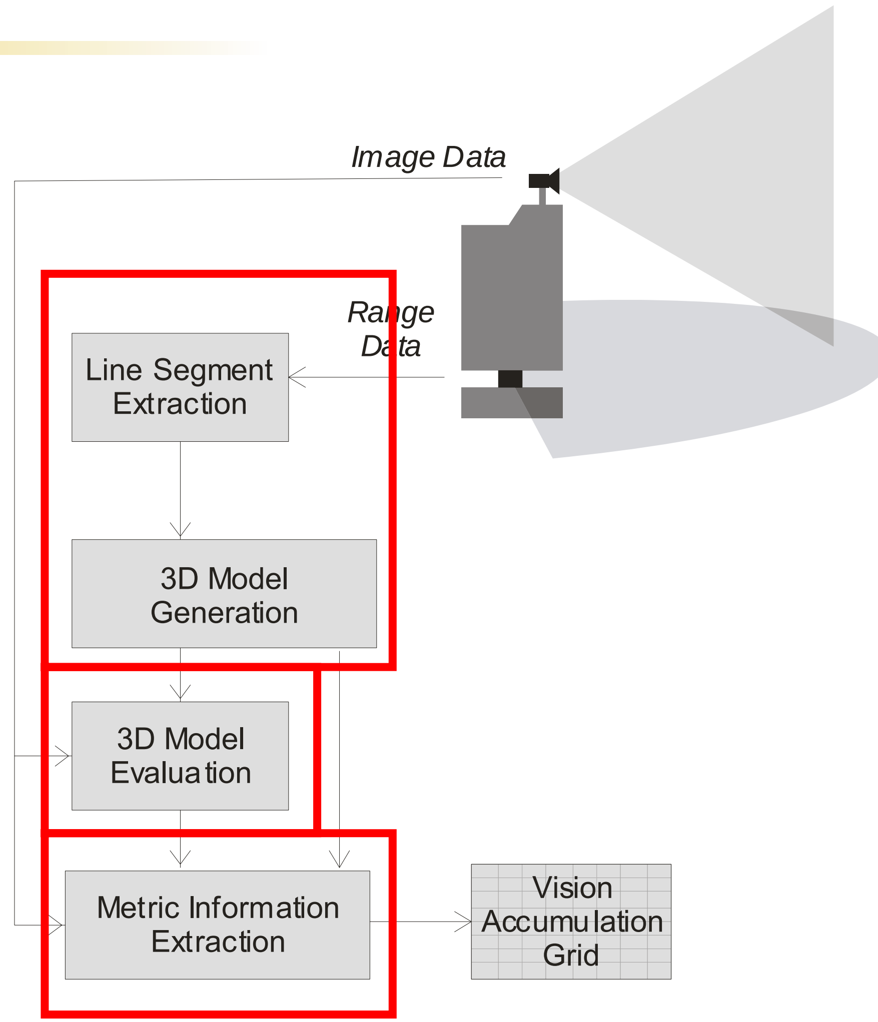


Visual Information Processing

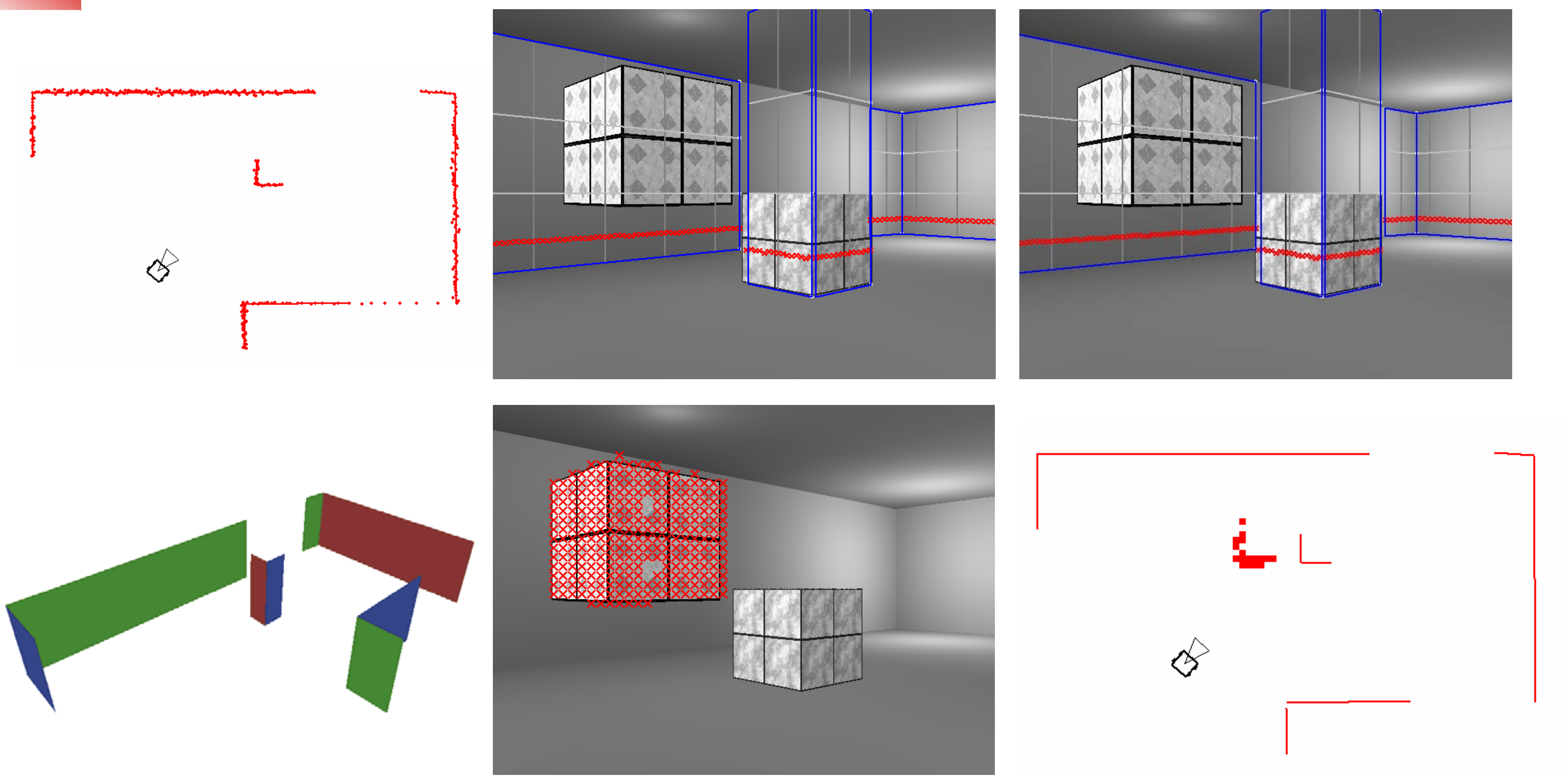
Step 1: From 2D laser measurements, form 3D model hypothesis

Step 2: From 1^{st} camera visual data and 3D model hypothesis predict visual data at the 2^{nd} camera. Then, compare actual and predicted visual data to evaluate the 3D model

Step 3: Wherever 3D model hypothesis is invalid, extract metric structure information



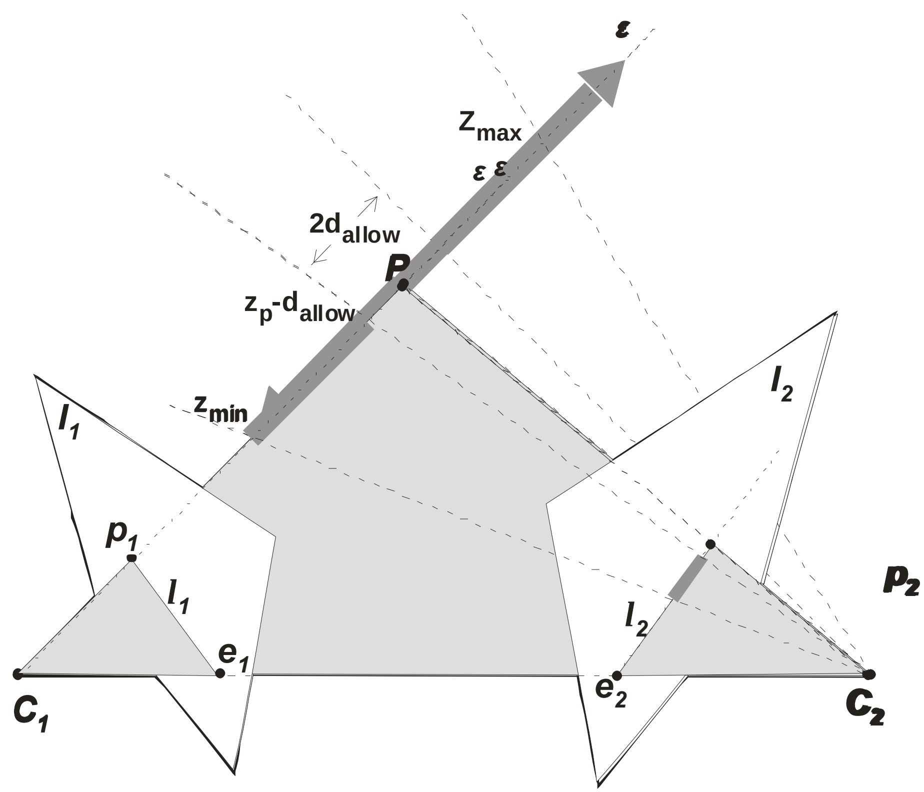
Visual Information Processing
(simulated example)



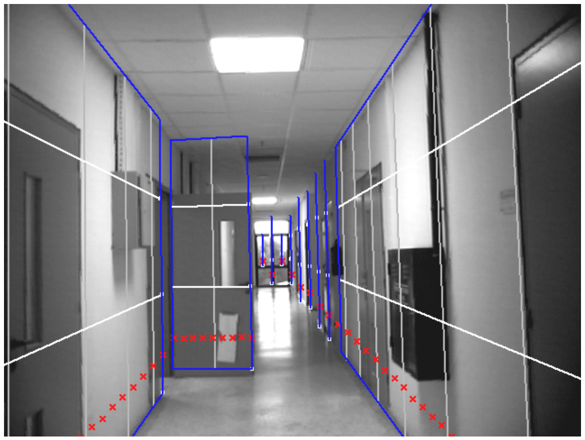
Visual Information Processing (Efficiency considerations)

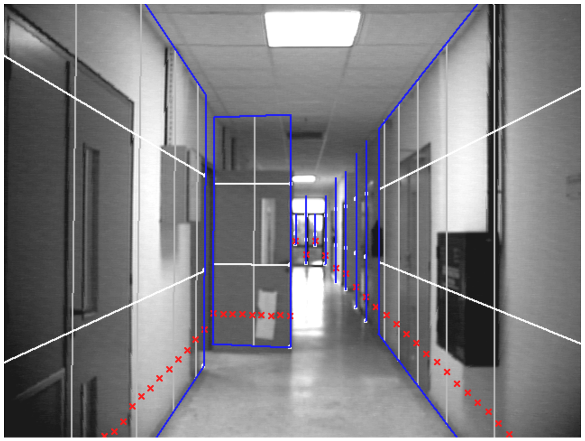
- Depth computations take place only where inconsistencies are detected
- For collision avoidance depth computations can be further eliminated.
  - Criterion 1. Visual range defers significantly to laser range data
  - Criterion 2. Visual range is shorter that corresponding laser suggests
  - Criterion 3. Visual range is neither too far nor to close to the robot.

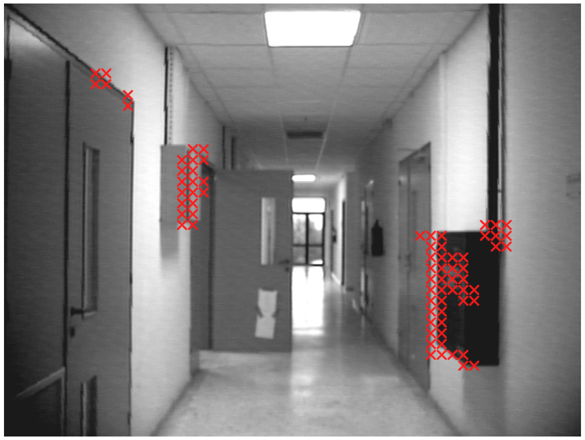
Crit Combination of Criteria max low

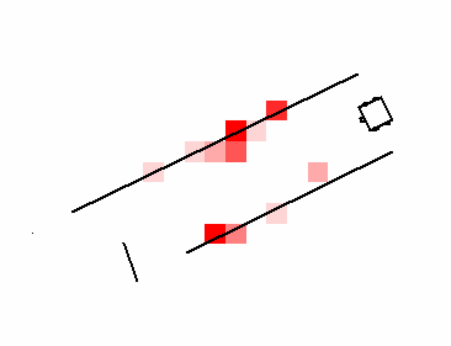


Visual Information Processing
(Real world example – outside our lab)









H. Baltzakis, A. Argyros and P. Trahanias, MVA 2003

Real Application
(Robotic Tour-guide in exhibition site)

The TOURBOT & WebFAIR Projects:

- Autonomous mobile robots in populated environments (serving real-visitors)
- Also operating over the web (serving web-visitirs)

P. Trahanias et al, IEEE RAM 2005

Thanks!

Thanks for your attention! Time for a break…