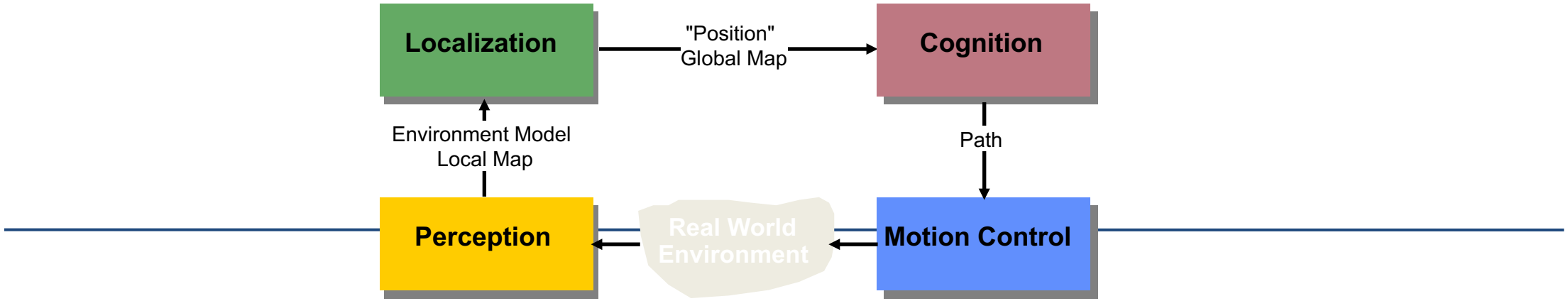
上海科技大学

ShanghaiTech University

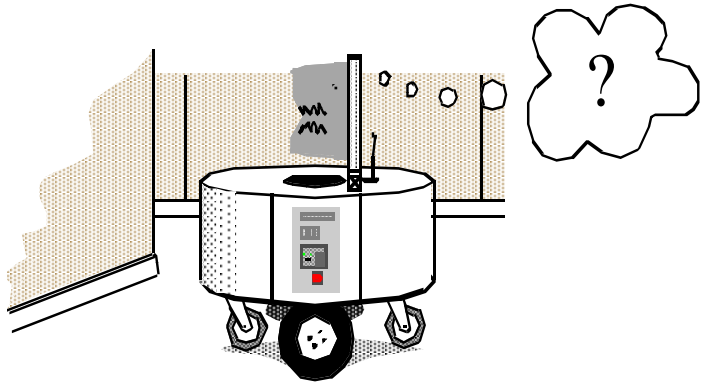
CS283: **Robotics Fall 2016: Localization**

Sören Schwertfeger / 师泽仁

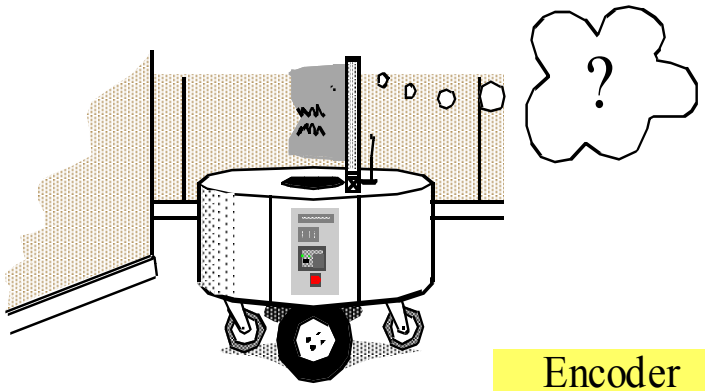
ShanghaiTech University

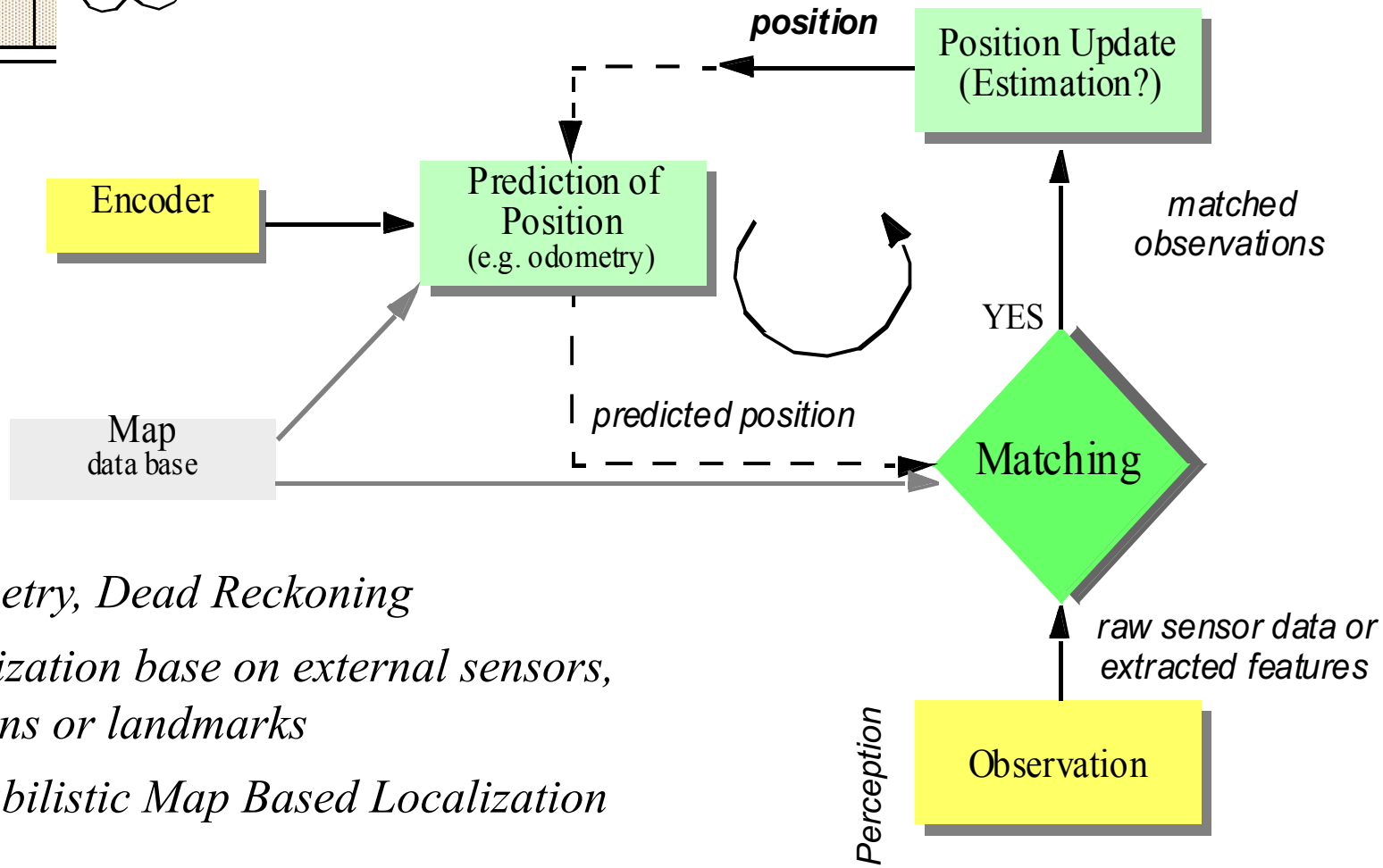


LOCALIZATION



Map based localization





- *Odometry, Dead Reckoning*
- *Localization base on external sensors, beacons or landmarks*
- *Probabilistic Map Based Localization*

Exteroceptive Sensor Noise

- Sensor noise is mainly influenced by environment
  e.g. surface, illumination …
- and by the measurement principle itself
  e.g. interference two Kinects
- Sensor noise drastically reduces the useful information of sensor readings.
  The solution is:
  - to model sensor noise appropriately
  - to take multiple readings into account
  - employ temporal and/or multi-sensor fusion

Effector Noise: Odometry, Deduced Reckoning

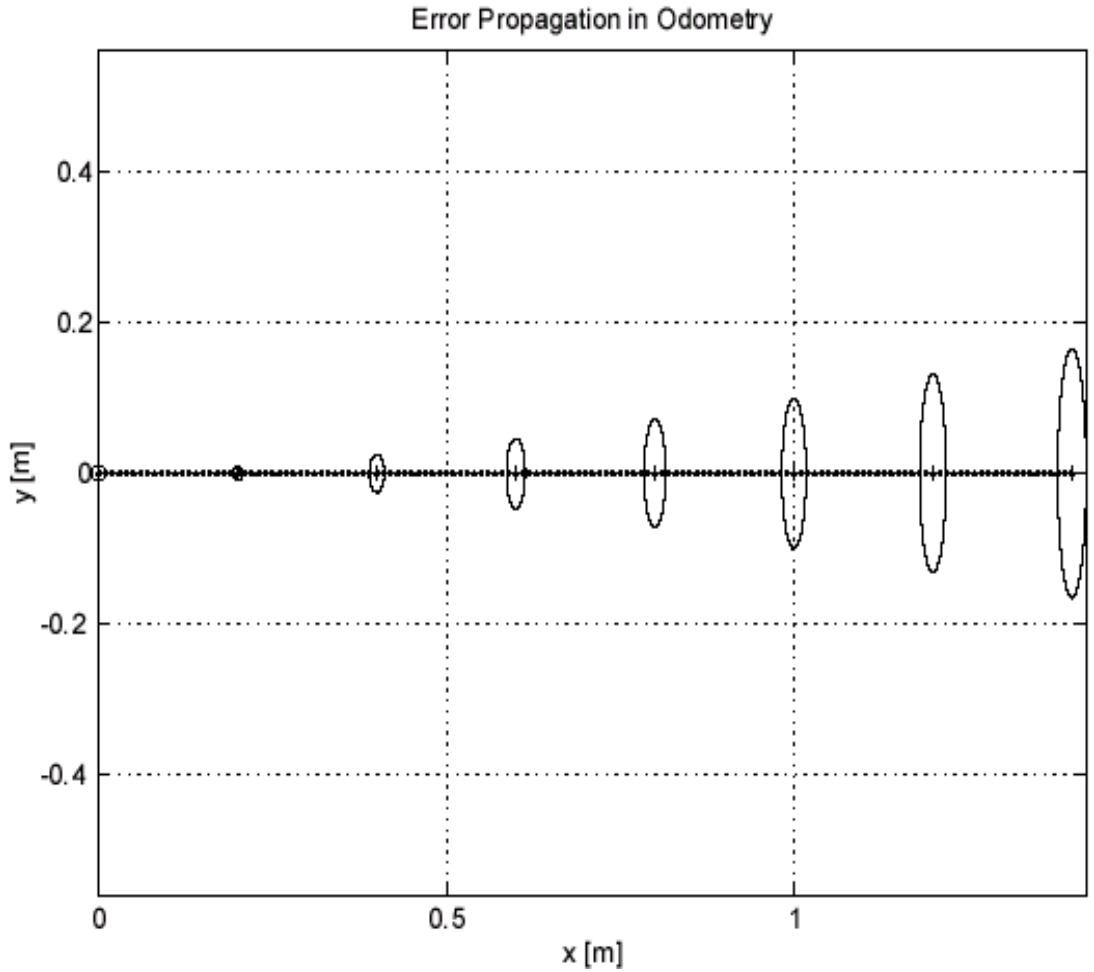
- Odometry and dead reckoning:
  Position update is based on proprioceptive sensors
  - Odometry: wheel sensors only
  - Dead reckoning: also heading sensors

- The movement of the robot, sensed with wheel encoders and/or heading sensors is integrated to the position.
  - Pros: Straight forward, easy
  - Cons: Errors are integrated -> unbound

- Using additional heading sensors (e.g. gyroscope) might help to reduce the cumulated errors, but the main problems remain the same.

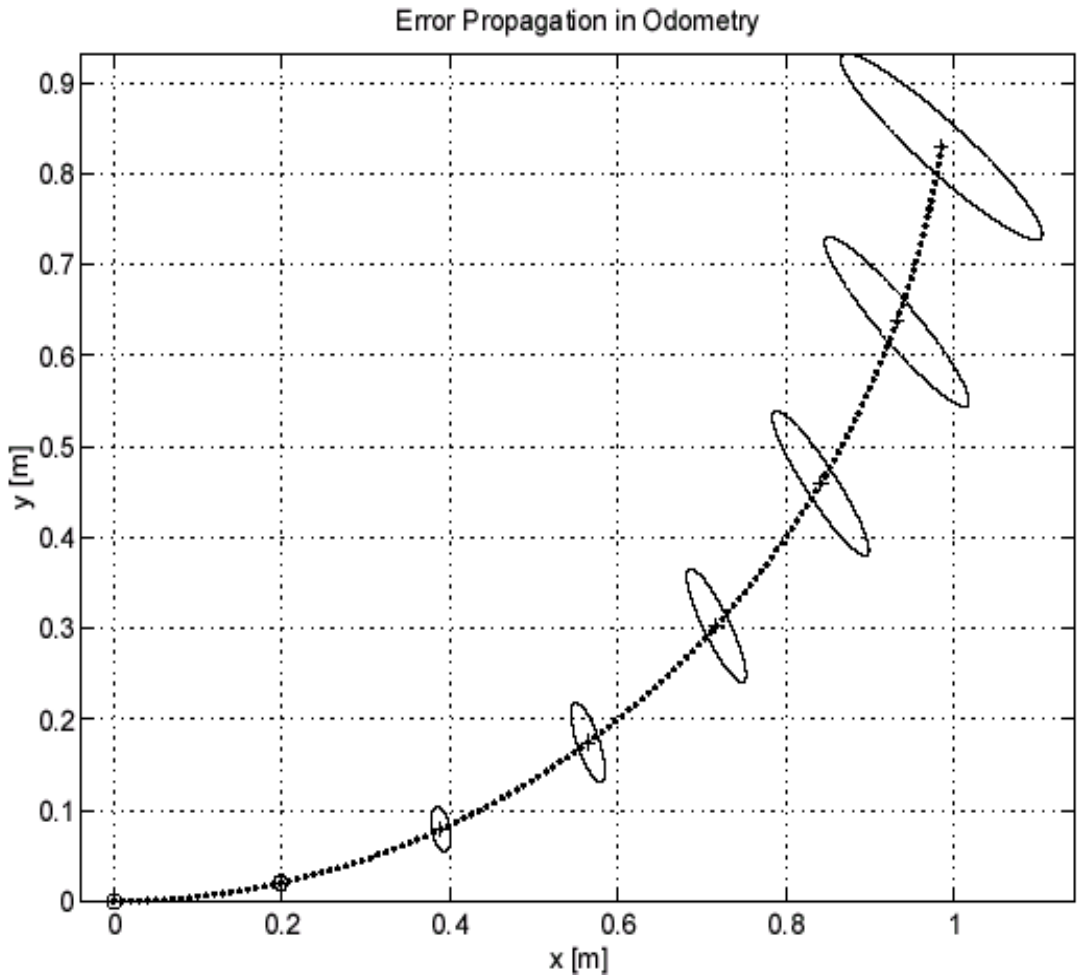
Odometry: Growth of Pose uncertainty for Straight Line Movement

- Note: Errors perpendicular to the direction of movement are growing much faster!



Odometry: Growth of Pose uncertainty for Movement on a Circle

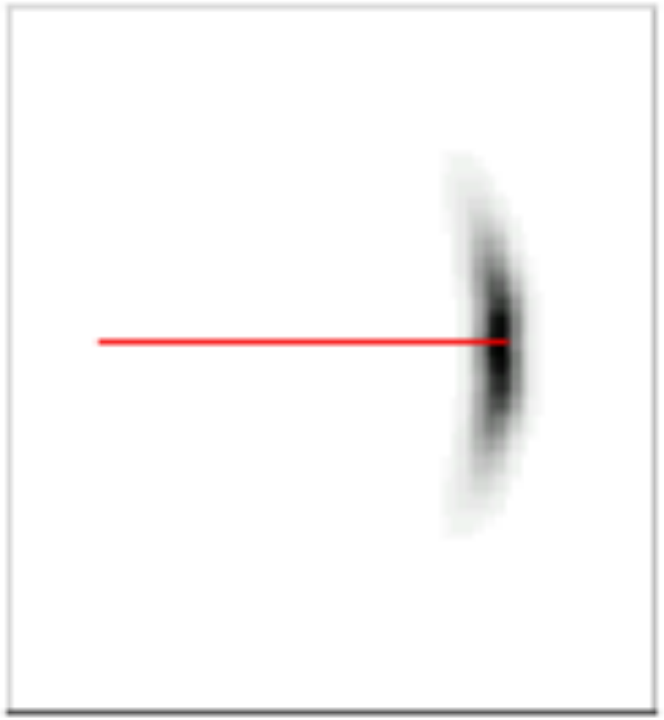
- Note: Errors ellipse in does not remain perpendicular to the direction of movement!

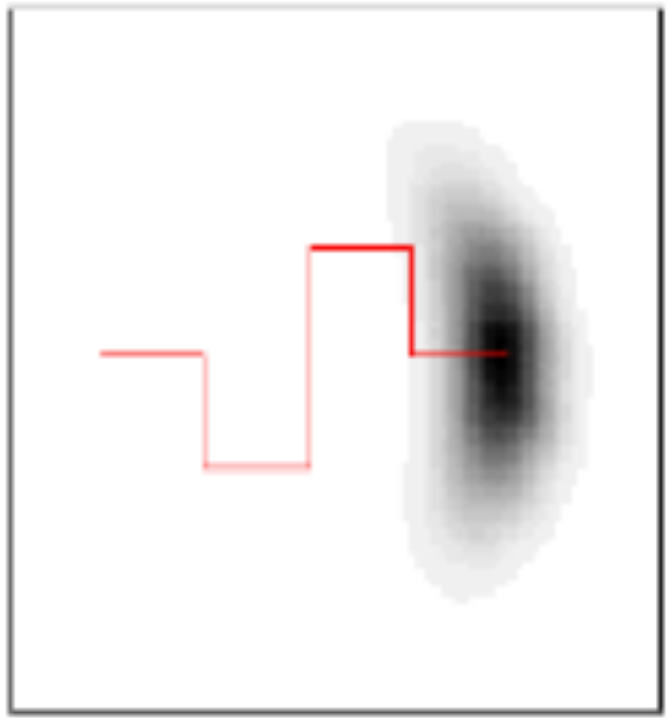


Odometry: example of non-Gaussian error model

- Note: Errors are not shaped like ellipses!

Courtesy AI Lab, Stanford

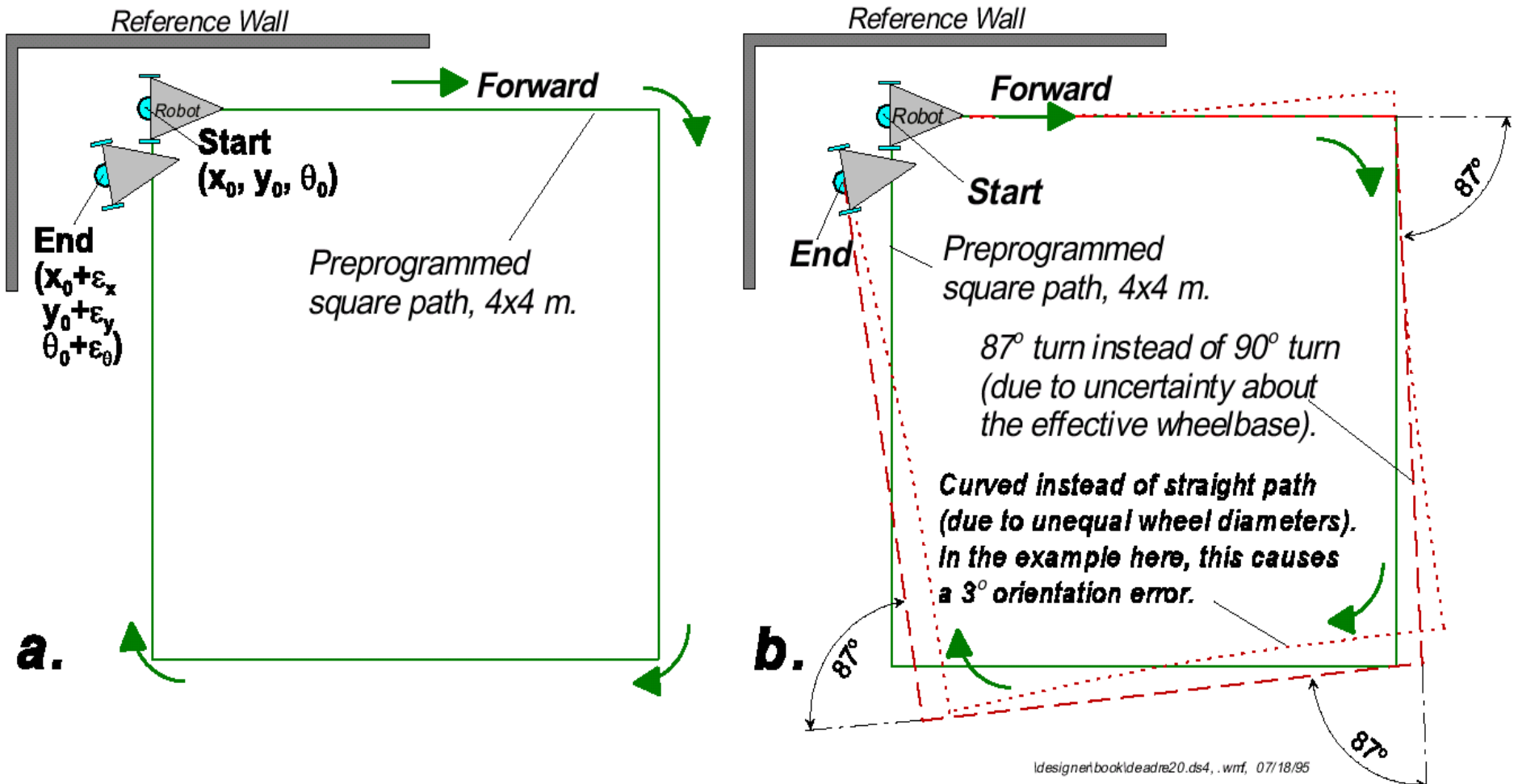




[Fox, Thrun, Burgard, Dellaert, 2000]

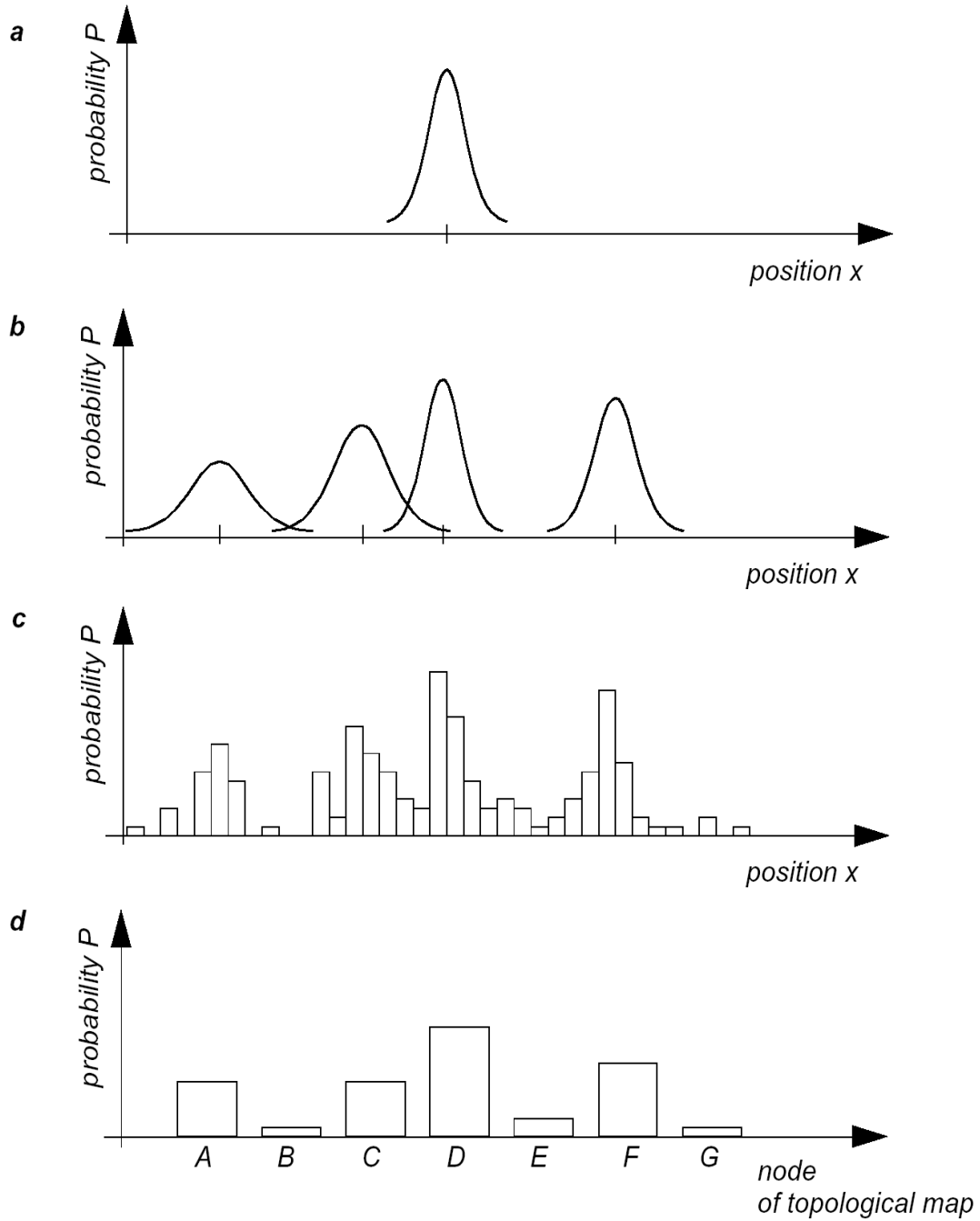
Odometry: Calibration of Errors

- The unidirectional square path experiment

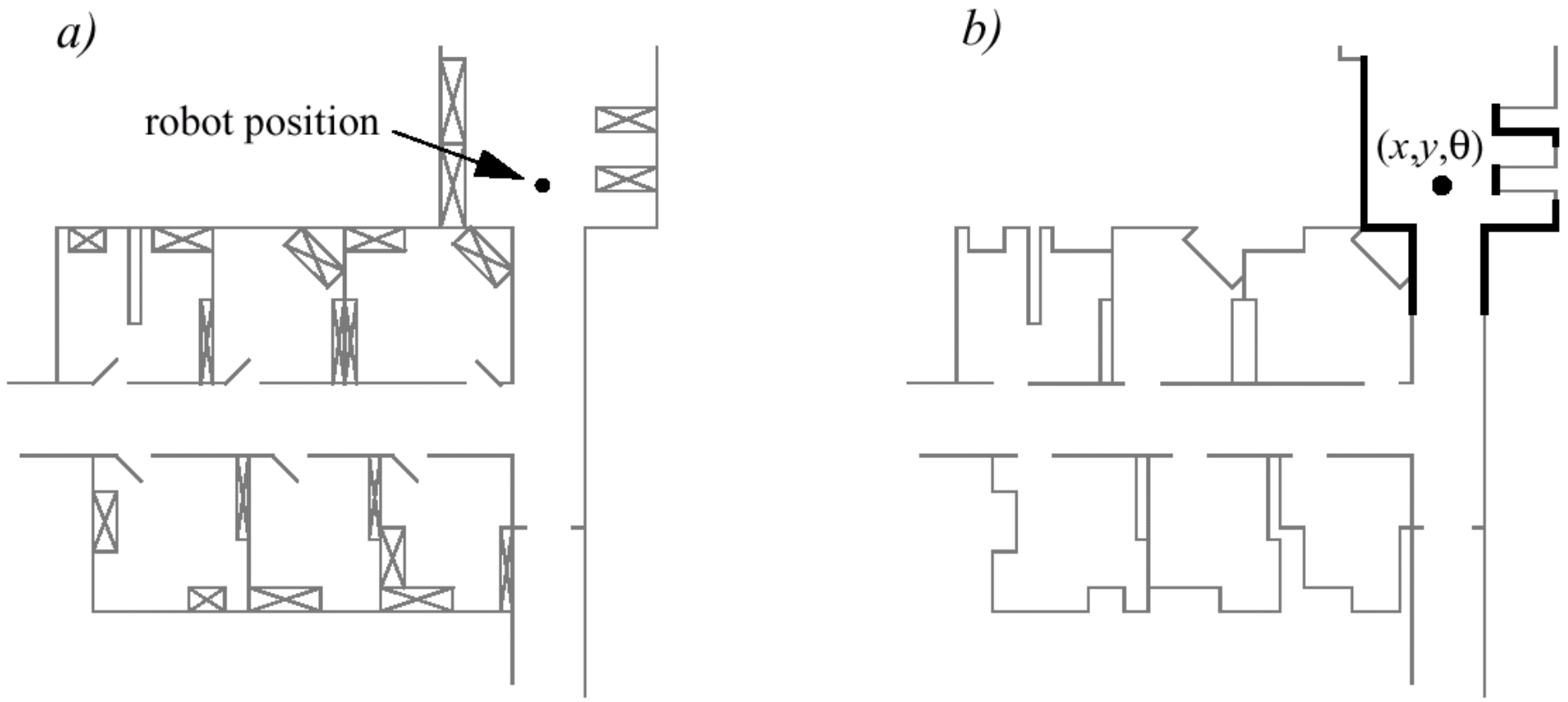


Belief Representation

- How do we represent the robot position, where the robot “believes” to be?
- a) Continuous map with single hypothesis probability distribution
- b) Continuous map with multiple hypothesis probability distribution
- c) Discretized map with probability distribution
- d) Discretized topological map with probability distribution

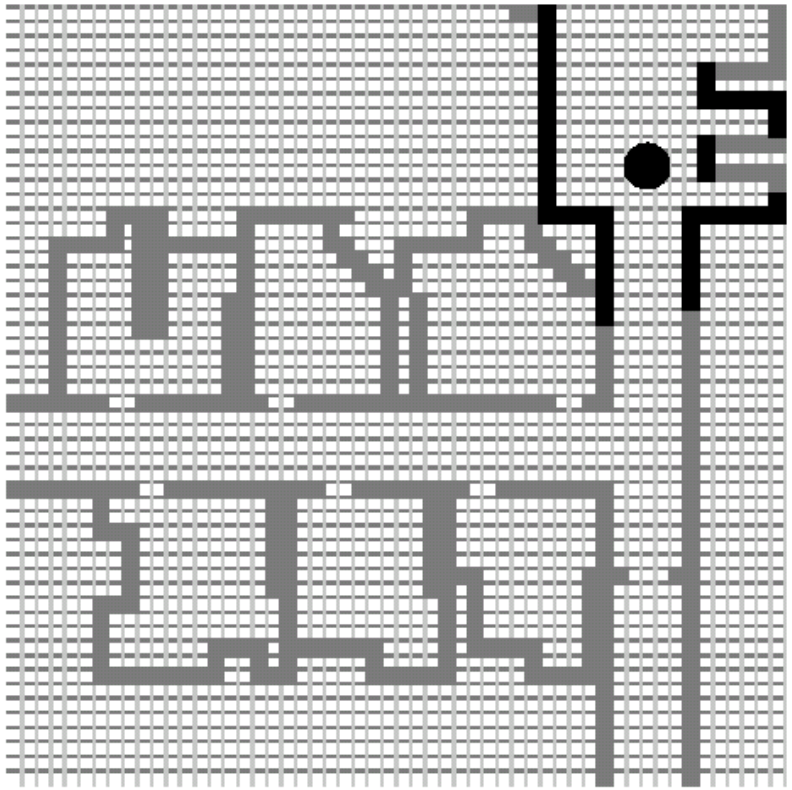


Single-hypothesis Belief – Continuous Line-Map

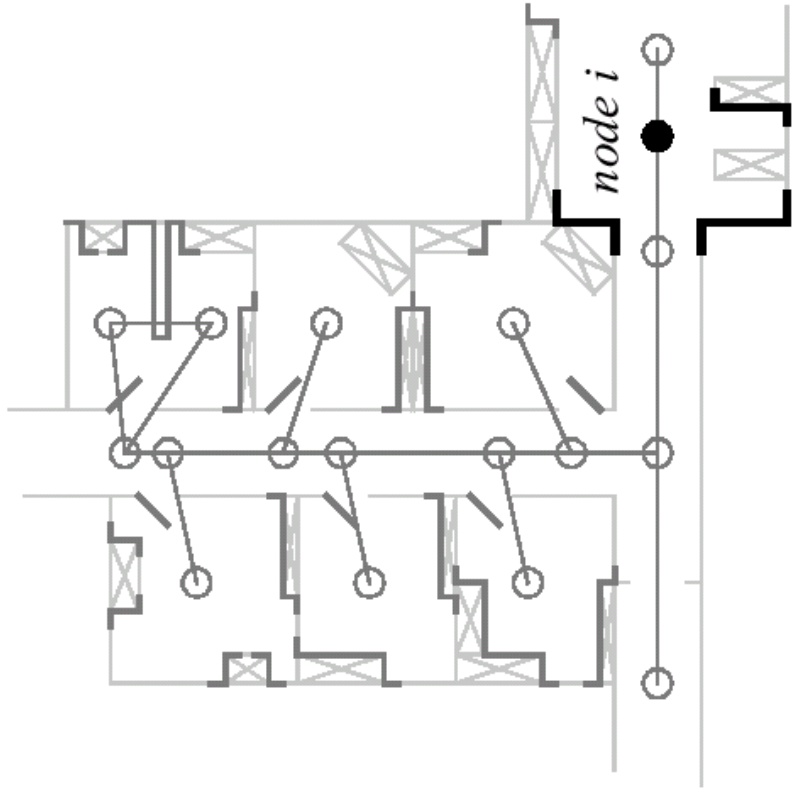


Single-hypothesis Belief – 2D Grid and Topological Map

c)

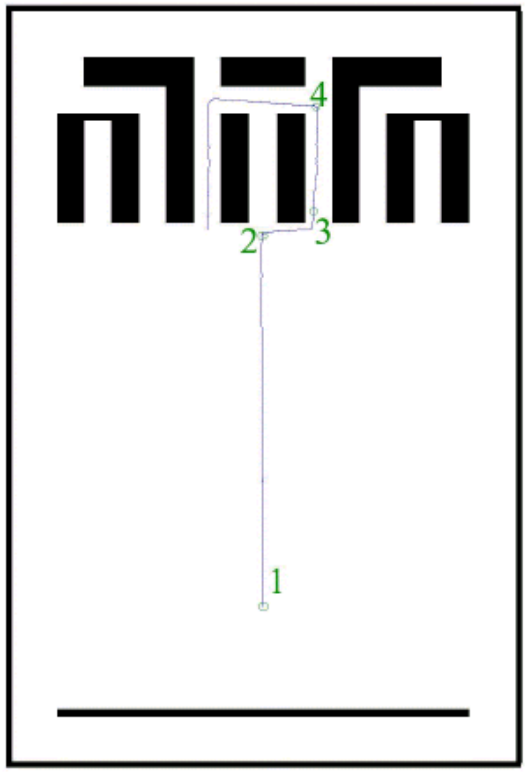


d)

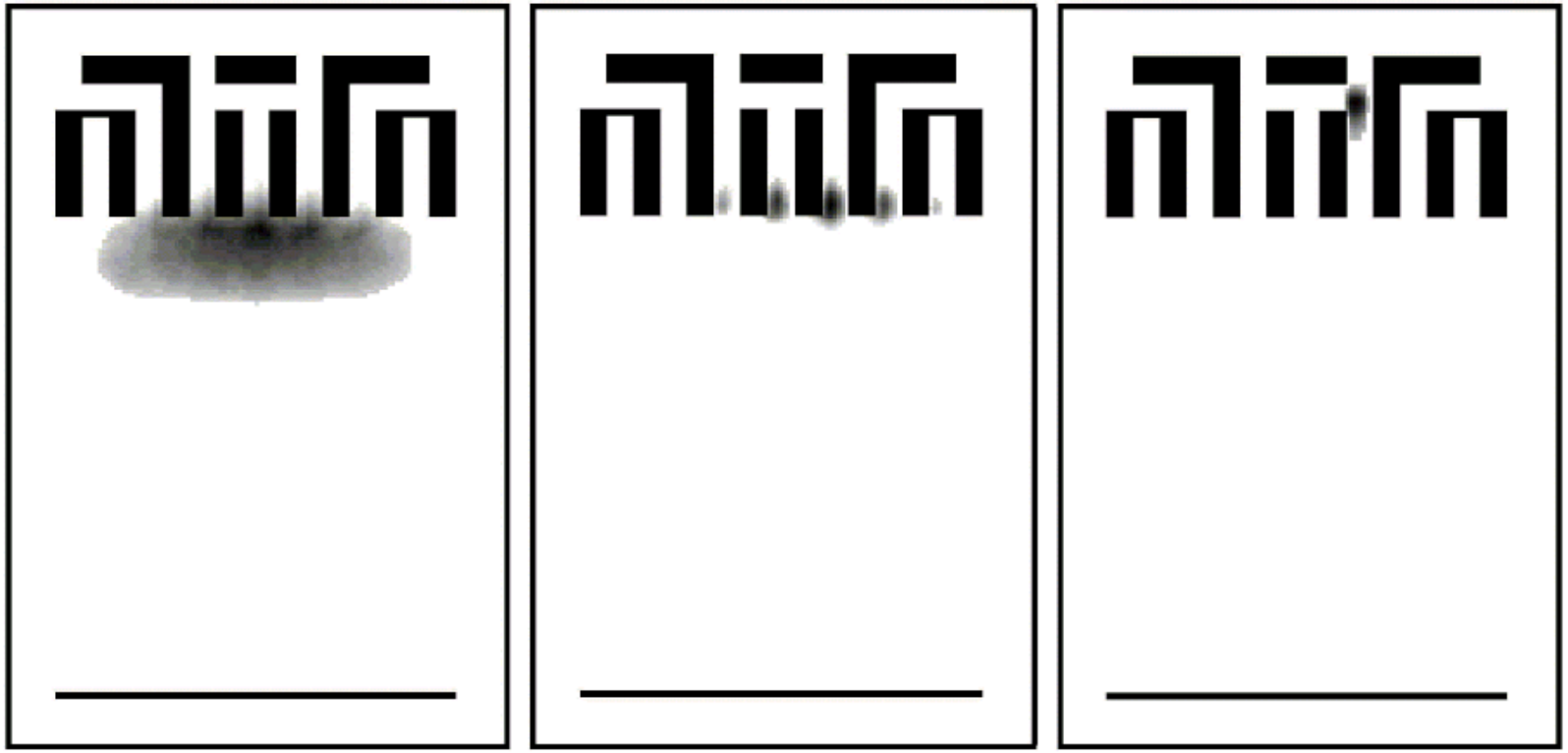


Grid-based Representation - Multi Hypothesis

Courtesy of W. Burgard



Path of the robot



Belief states at positions 2, 3 and 4

Representation of the Environment

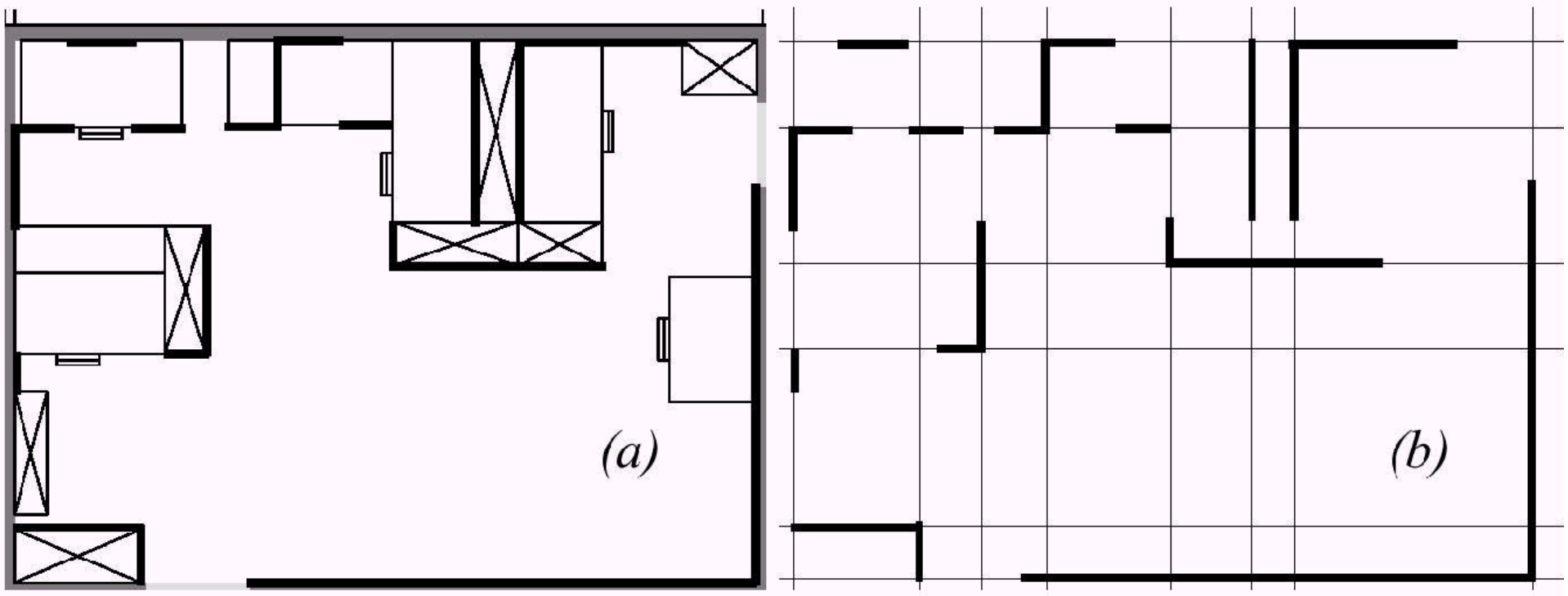
- Environment Representation
  - Continuous Metric → x, y, θ
  - Discrete Metric → metric grid
  - Discrete Topological → topological grid

- Environment Modeling
  - Raw sensor data, e.g. laser range data, grayscale images
    - large volume of data, low distinctiveness on the level of individual values
    - makes use of all acquired information
  - Low level features, e.g. line other geometric features
    - medium volume of data, average distinctiveness
    - filters out the useful information, still ambiguities
  - High level features, e.g. doors, a car, the Eiffel tower
    - low volume of data, high distinctiveness
    - filters out the useful information, few/no ambiguities, not enough information

Map Representation: Continuous Line-Based

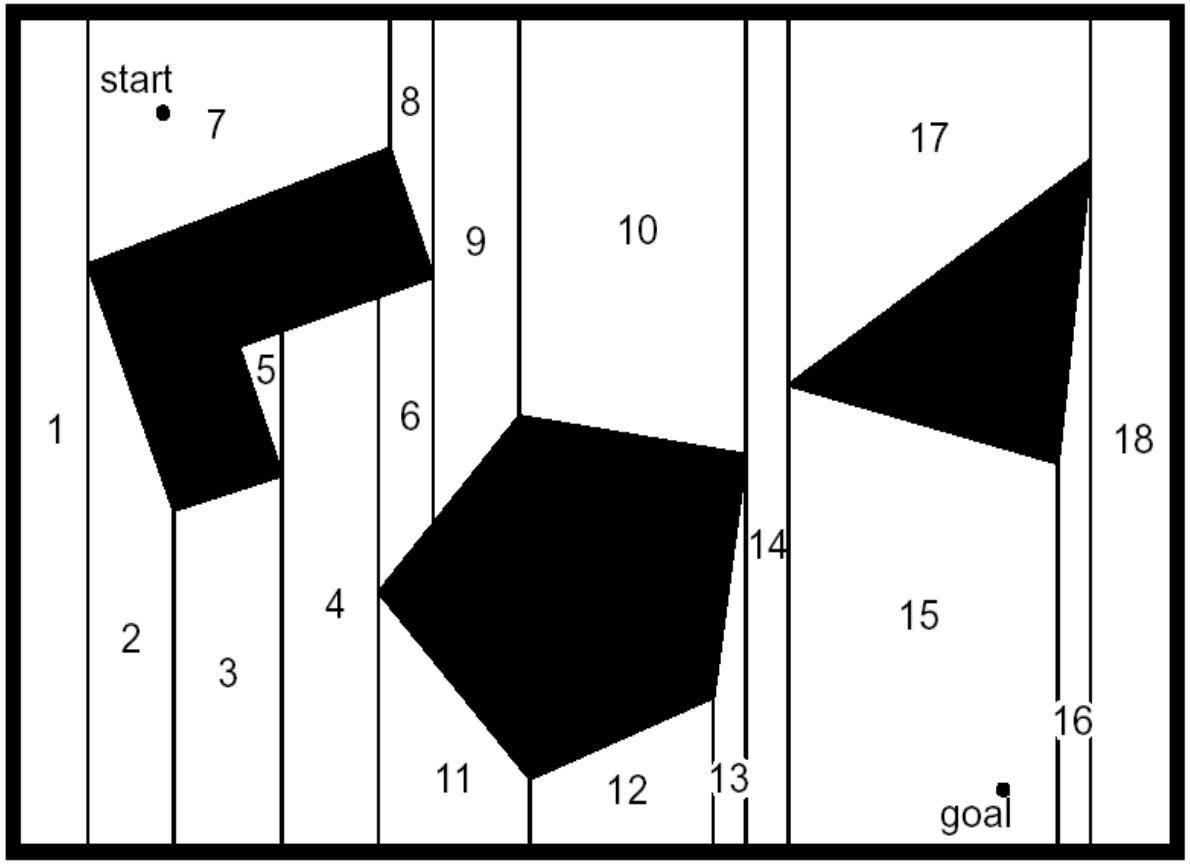
a) Architecture map

b) Representation with set of finite or infinite lines



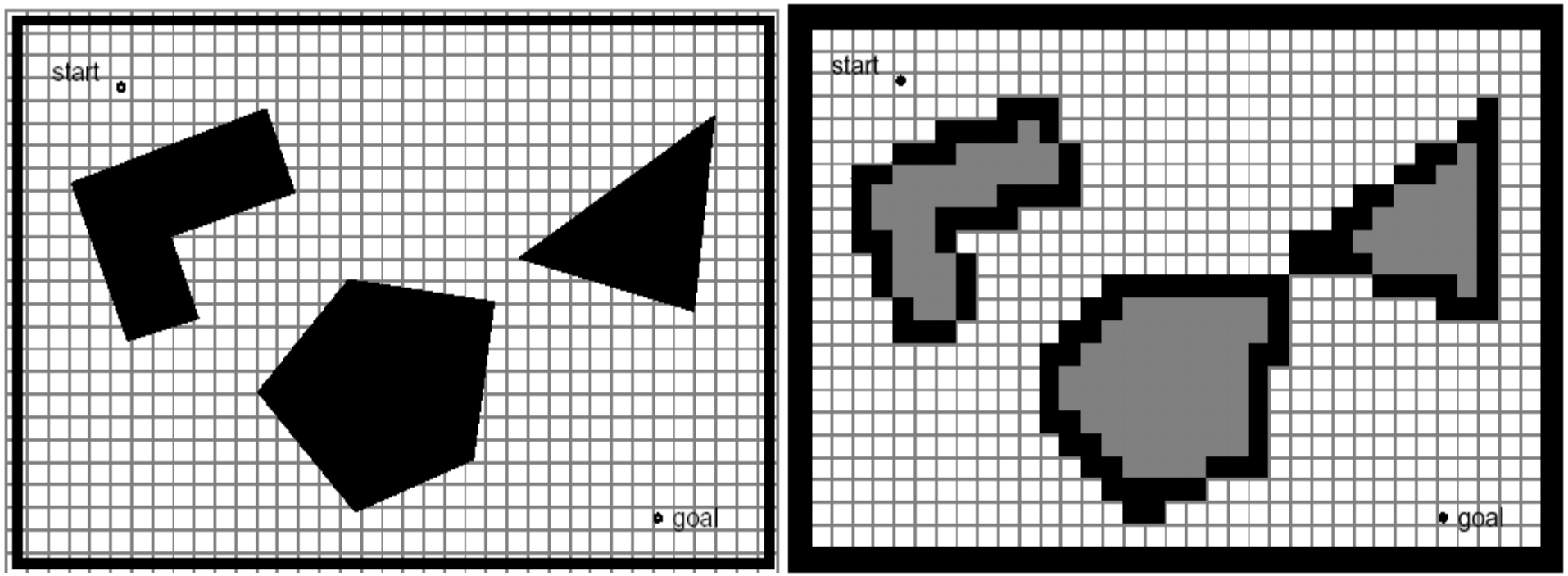
Map Representation: Exact cell decomposition

- Exact cell decomposition - Polygons



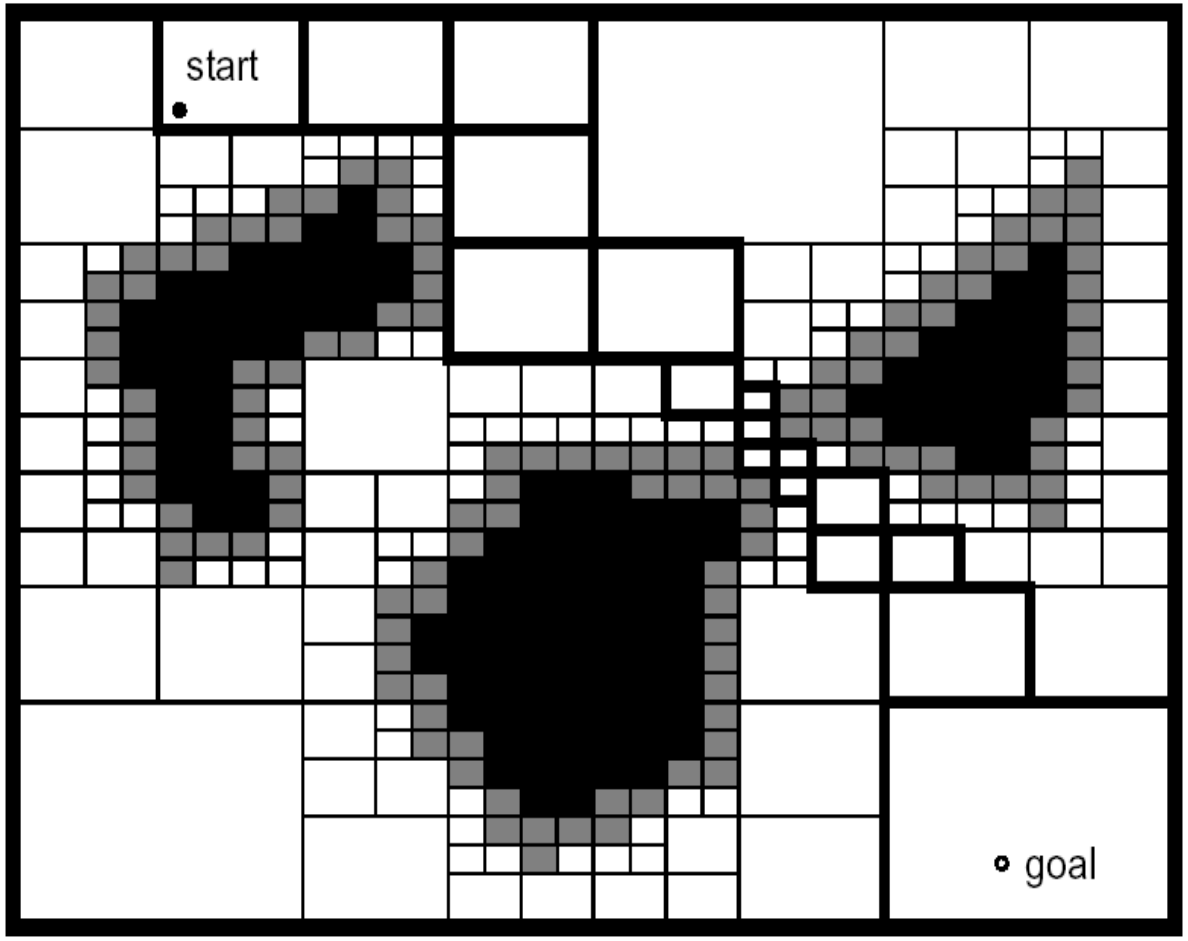
Map Representation: Approximate cell decomposition (1)

- Fixed cell decomposition
  - Narrow passages disappear



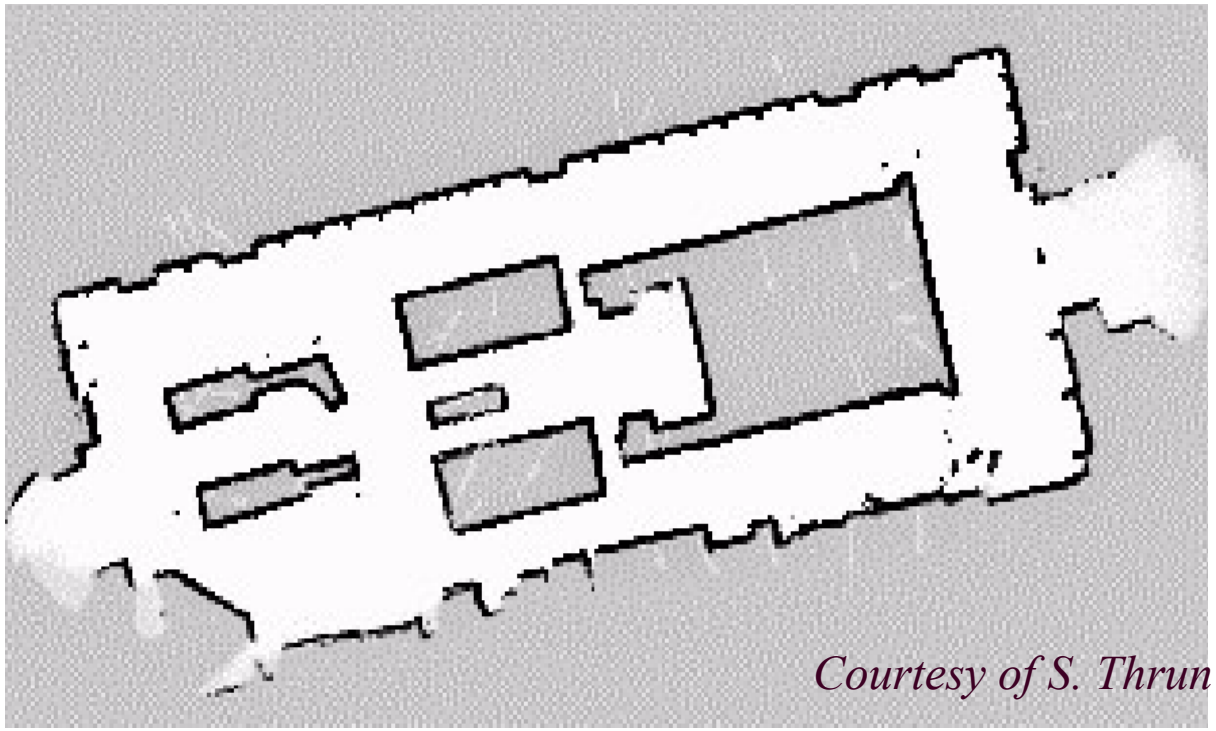
Map Representation: Adaptive cell decomposition (2)

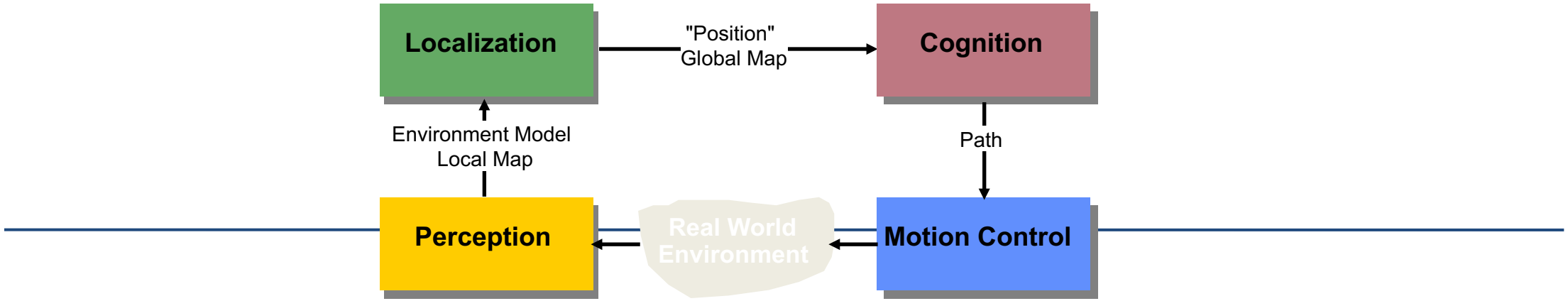
- For example: Quadtree



Map Representation: Occupancy grid

- Fixed cell decomposition: occupancy grid example
  - In occupancy grids, each cell may have a counter where 0 indicates that the cell has not been hit by any ranging measurements and therefore it is likely free-space. As the number of ranging strikes increases, the cell value is incremented and, above a certain threshold, the cell is deemed to be an obstacle
  - The values of the cells are discounted when a ranging strike travels through the cell. This allows us to represent “transient” (dynamic) obstacles

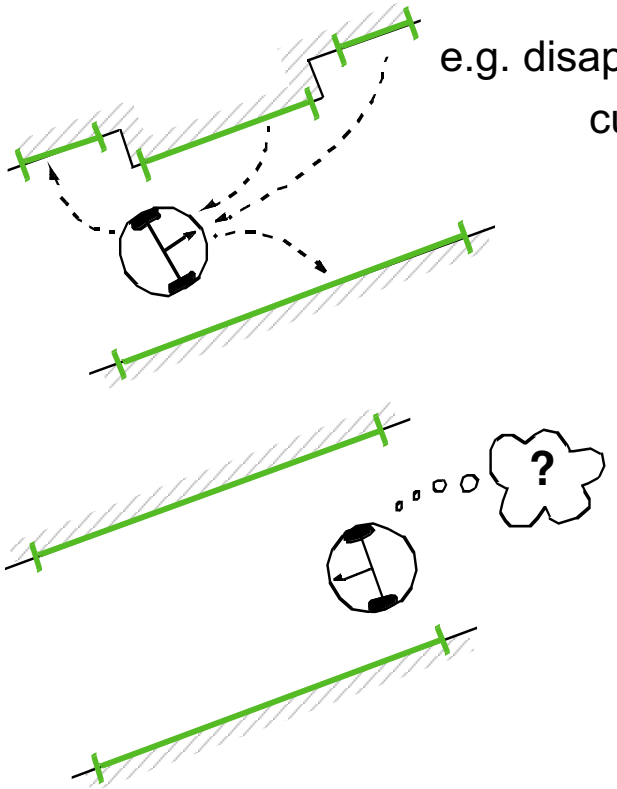




SLAM: SIMULTANEOUS LOCALIZATION AND MAPPING

Map Building: The Problems

1. Map Maintaining: Keeping track of changes in the environment

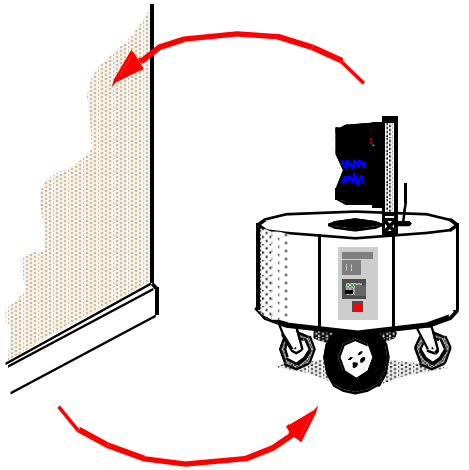


e.g. disappearing cupboard

- e.g. measure of belief of each environment feature

2. Representation and Reduction of Uncertainty

position of robot -> position of wall

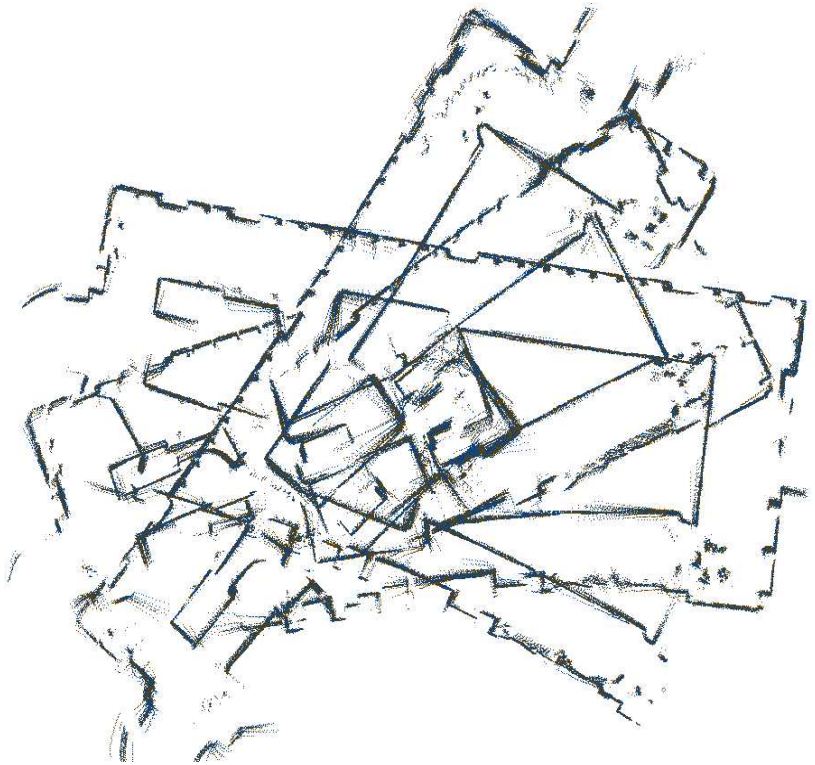


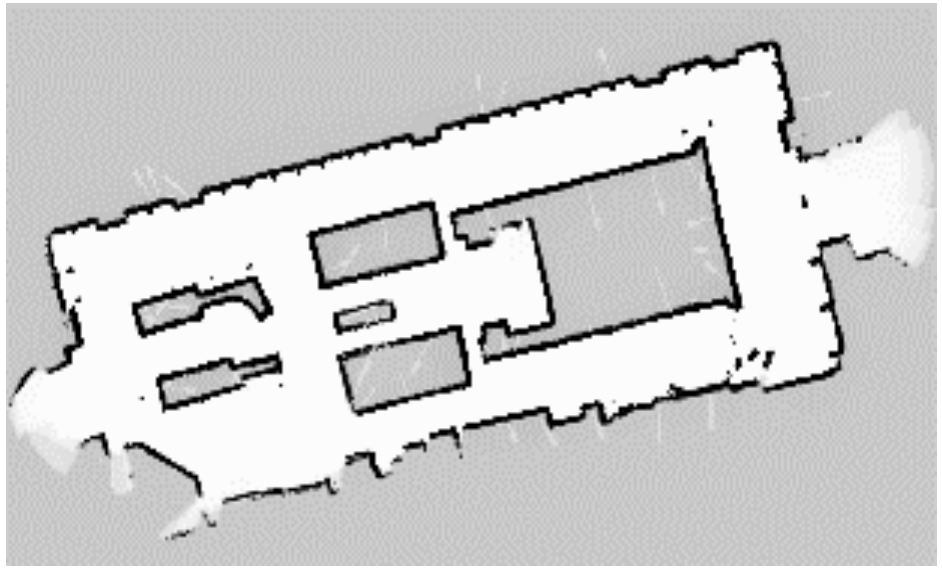
position of wall -> position of robot

- probability densities for feature positions
- Inconsistent map due to motion drift

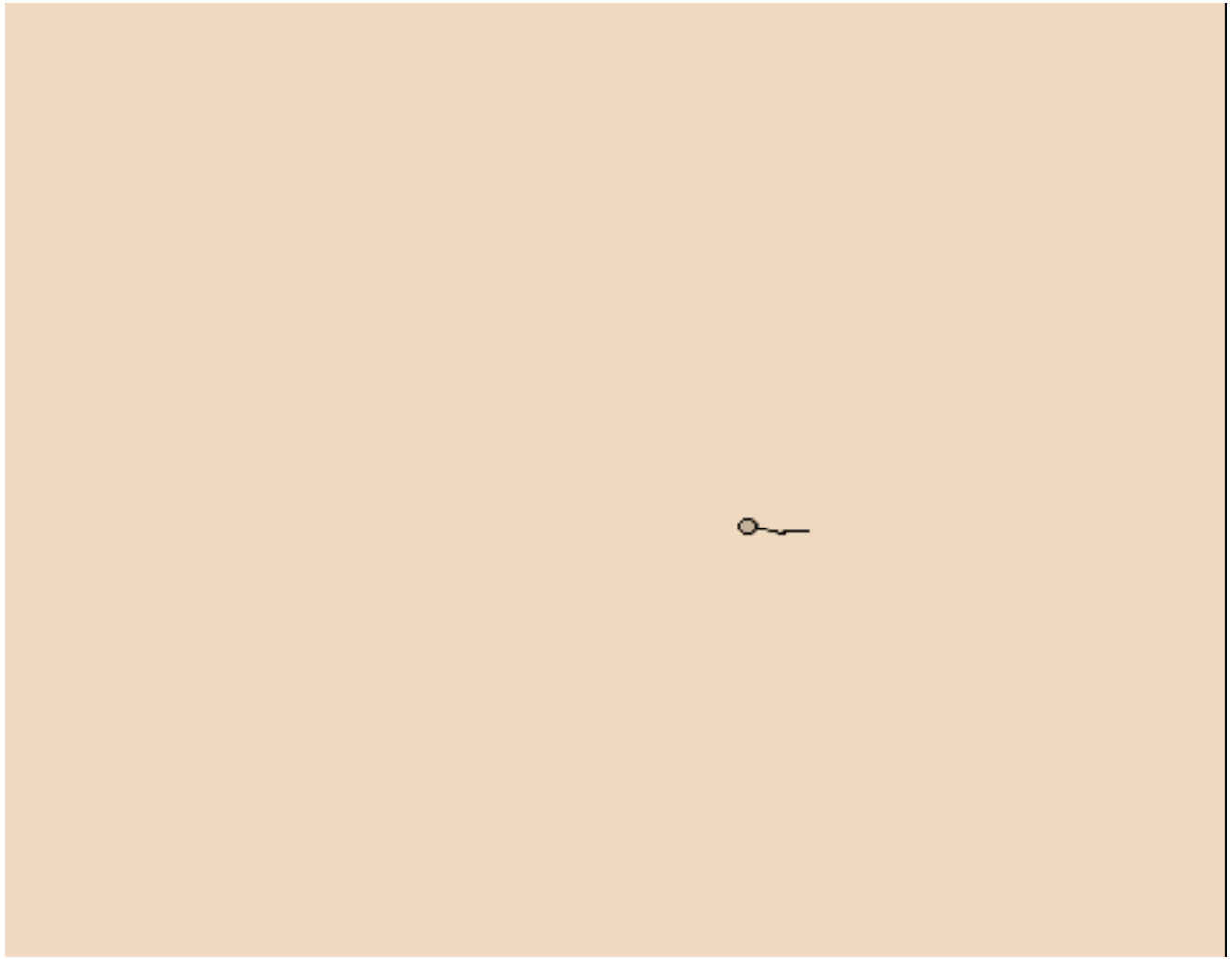
Cyclic Environments

- Small local error accumulate to arbitrary large global errors!
- This is usually irrelevant for navigation
- However, when closing loops, global error does matter



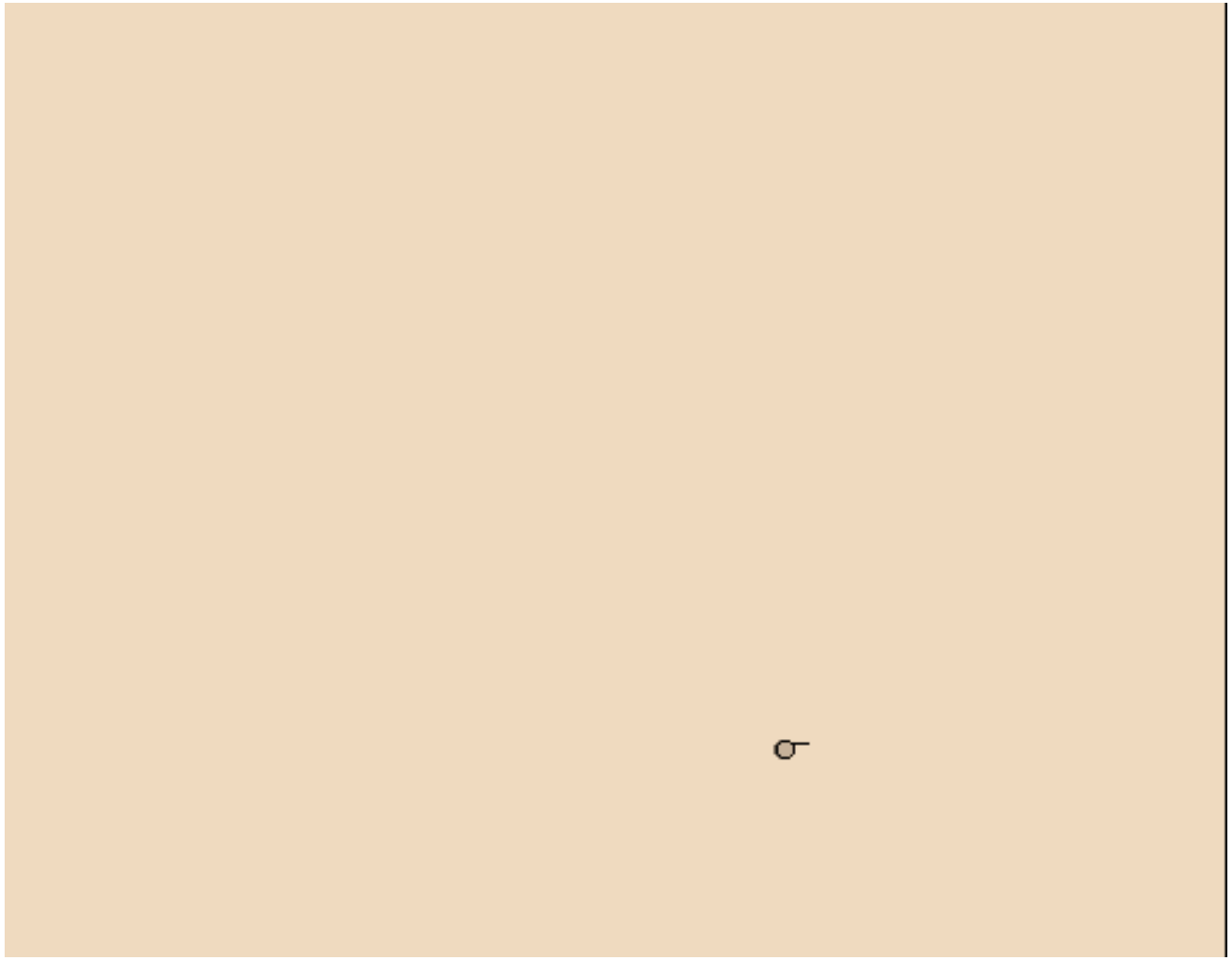


Raw Odometry …



Courtesy of S. Thrun

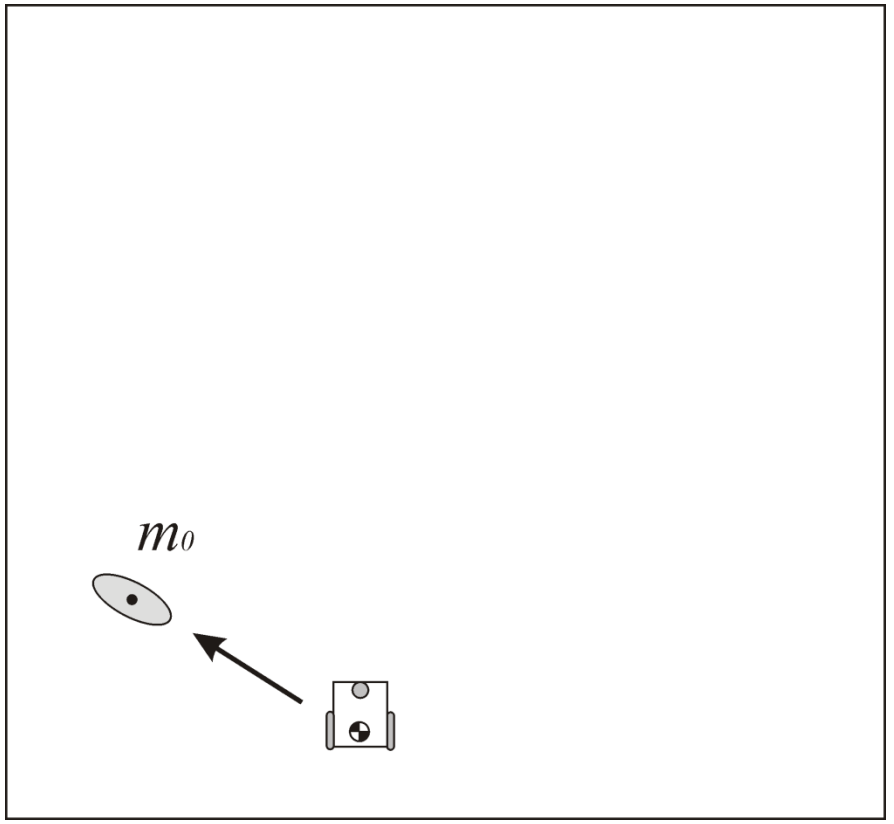
Scan Matching:
compare to sensor data from previous scan



Courtesy of S. Thrun

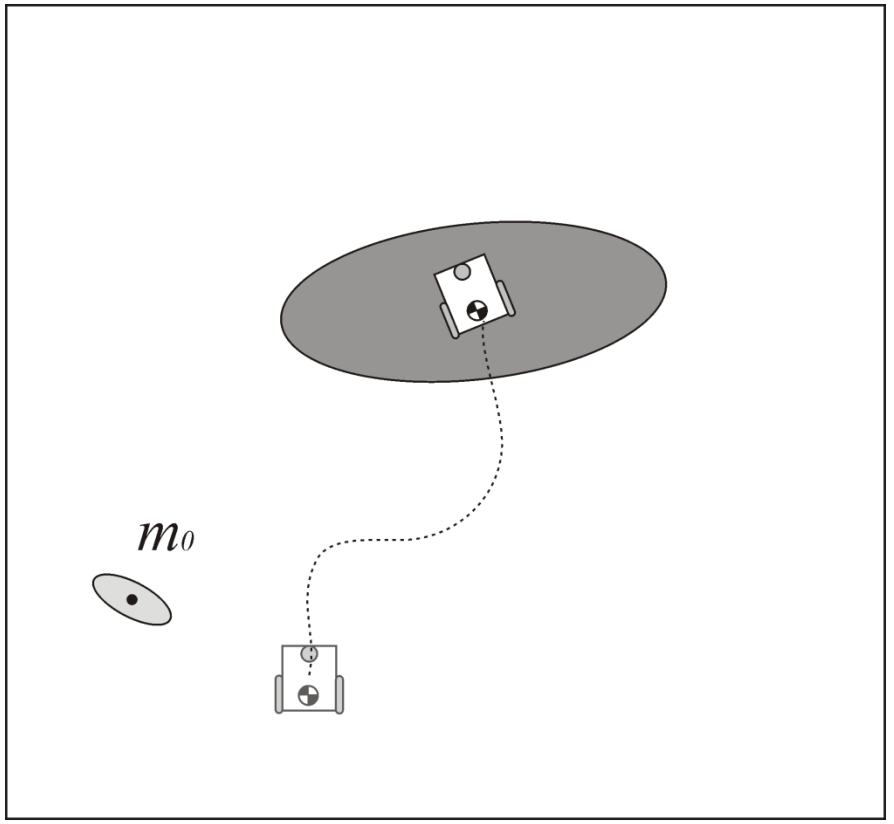
SLAM overview

- Let us assume that the robot uncertainty at its initial location is zero.
- From this position, the robot observes a feature which is mapped with an uncertainty related to the exteroceptive sensor error model



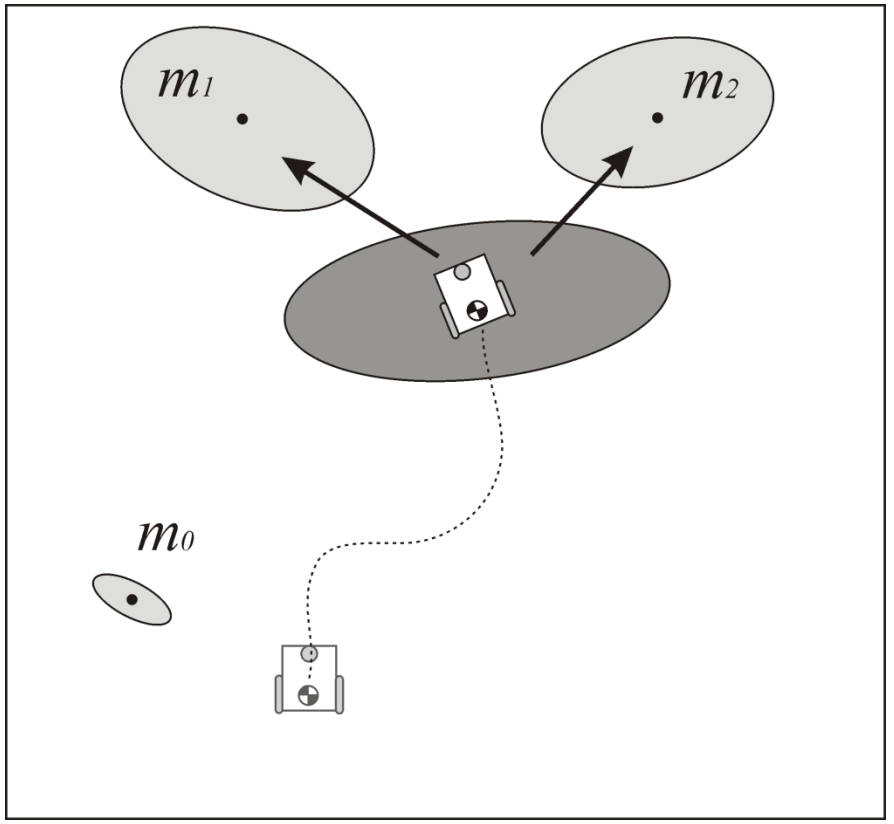
SLAM overview

- As the robot moves, its pose uncertainty increases under the effect of the errors introduced by the odometry



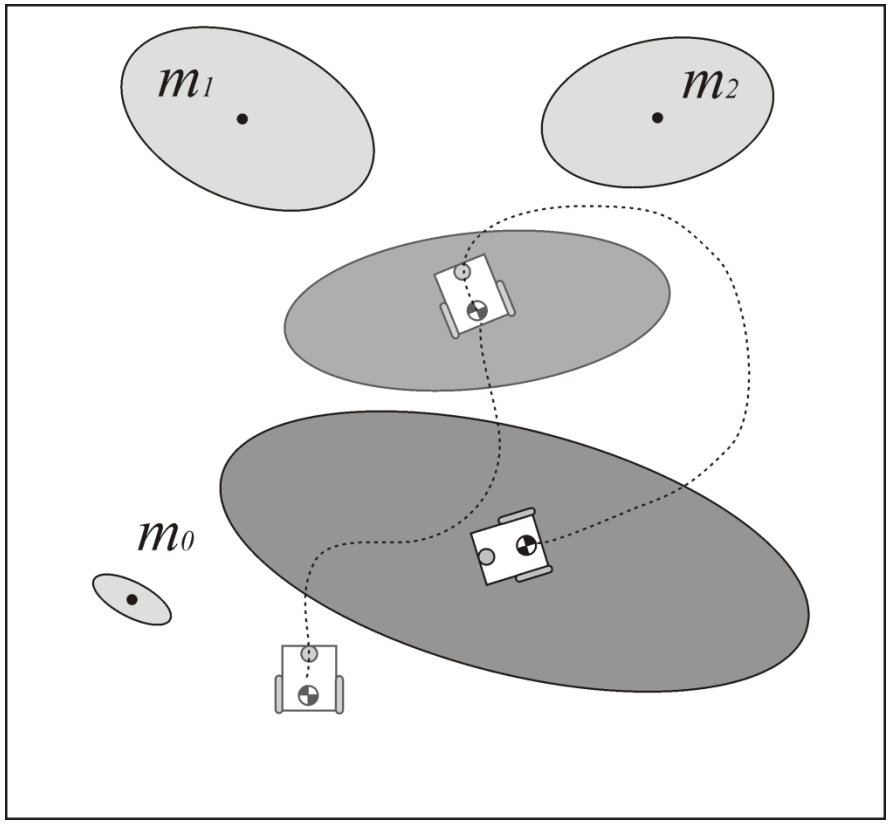
SLAM overview

- At this point, the robot observes two features and maps them with an uncertainty which results from the combination of the measurement error with the robot pose uncertainty
- From this, we can notice that the map becomes correlated with the robot position estimate. Similarly, if the robot updates its position based on an observation of an imprecisely known feature in the map, the resulting position estimate becomes correlated with the feature location estimate.



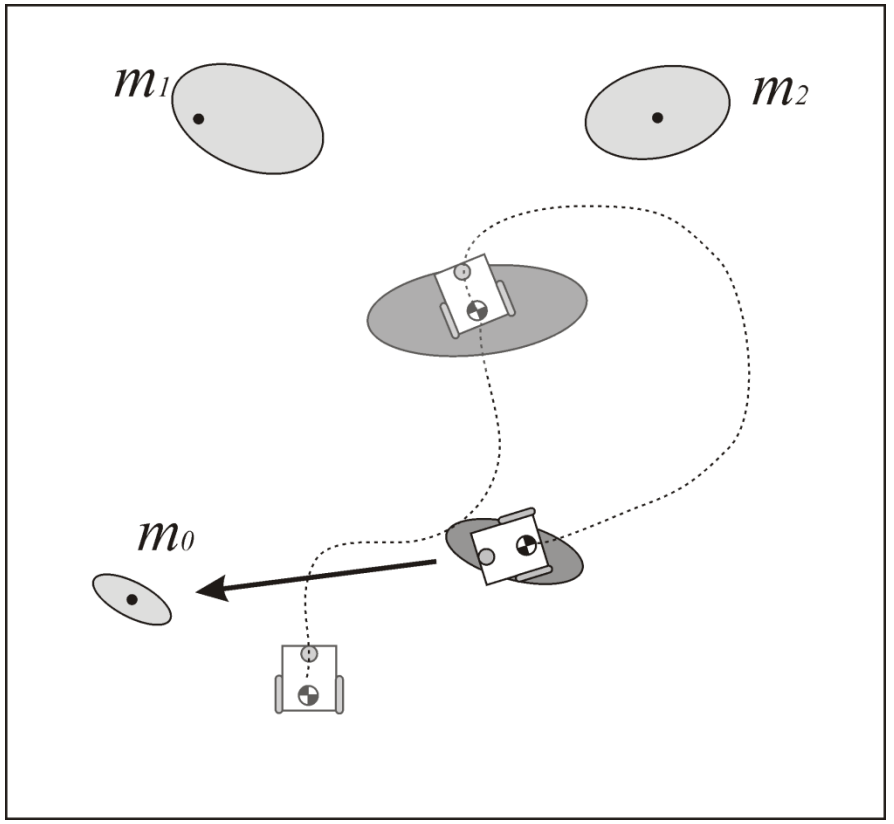
SLAM overview

- The robot moves again and its uncertainty increases under the effect of the errors introduced by the odometry



SLAM overview

- In order to reduce its uncertainty, the robot must observe features whose location is relatively well known. These features can for instance be landmarks that the robot has already observed before.
- In this case, the observation is called *loop closure detection.*
- When a loop closure is detected, the robot pose uncertainty shrinks.
- At the same time, the map is updated and the uncertainty of other observed features and all previous robot poses also reduce



The Three SLAM paradigms

- Most of the SLAM algorithms are based on the following three different approaches:
  - Extended Kalman Filter SLAM: (called EKF SLAM)
  - Particle Filter SLAM: (called FAST SLAM)
  - Graph-Based SLAM

EKF SLAM: overview

- **extended state vector** y_t : robot pose x_t + position of all the features m_i in the map:

$$y_t = [x_t, m_0, \ldots, m_{n-1}]^T$$

- Example: 2D line-landmarks, size of y_t = $3+2n$: three variables to represent the robot pose + $2n$ variables for the n line-landmarks having vector components

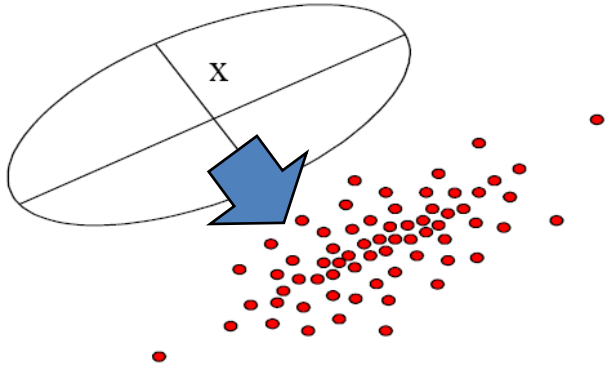
$$(\alpha_i, r_i)$$

$$y_t = [x_t, y_t, \theta_t, \alpha_0, r_0, \ldots, \alpha_{n-1}, r_{n-1}]^T$$

- As the robot moves and takes measurements, the state vector and covariance matrix are updated using the standard equations of the extended Kalman filter
- Drawback: EKF SLAM is computationally very expensive.

Particle Filter SLAM: FastSLAM

- **FastSLAM approach**
  - Using particle filters.
  - Particle filters: mathematical models that represent probability distribution as a set of discrete particles that occupy the state space.



probability distribution (ellipse) as particle set (red dots)

- Particle filter update
  - Generate new particle distribution using motion model and controls

a) For each particle:

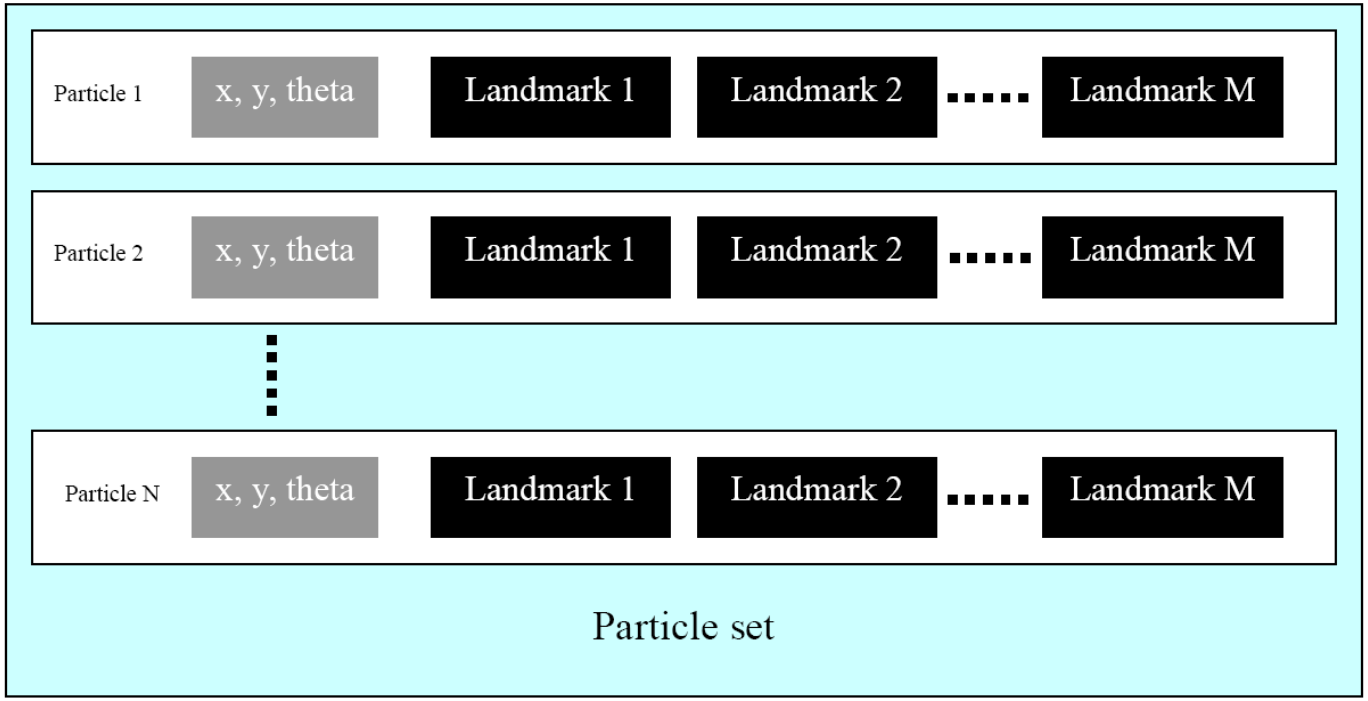
1. Compare particle's prediction of measurements with actual measurements
2. Particles whose predictions match the measurements are given a high weight

b) Filter resample:

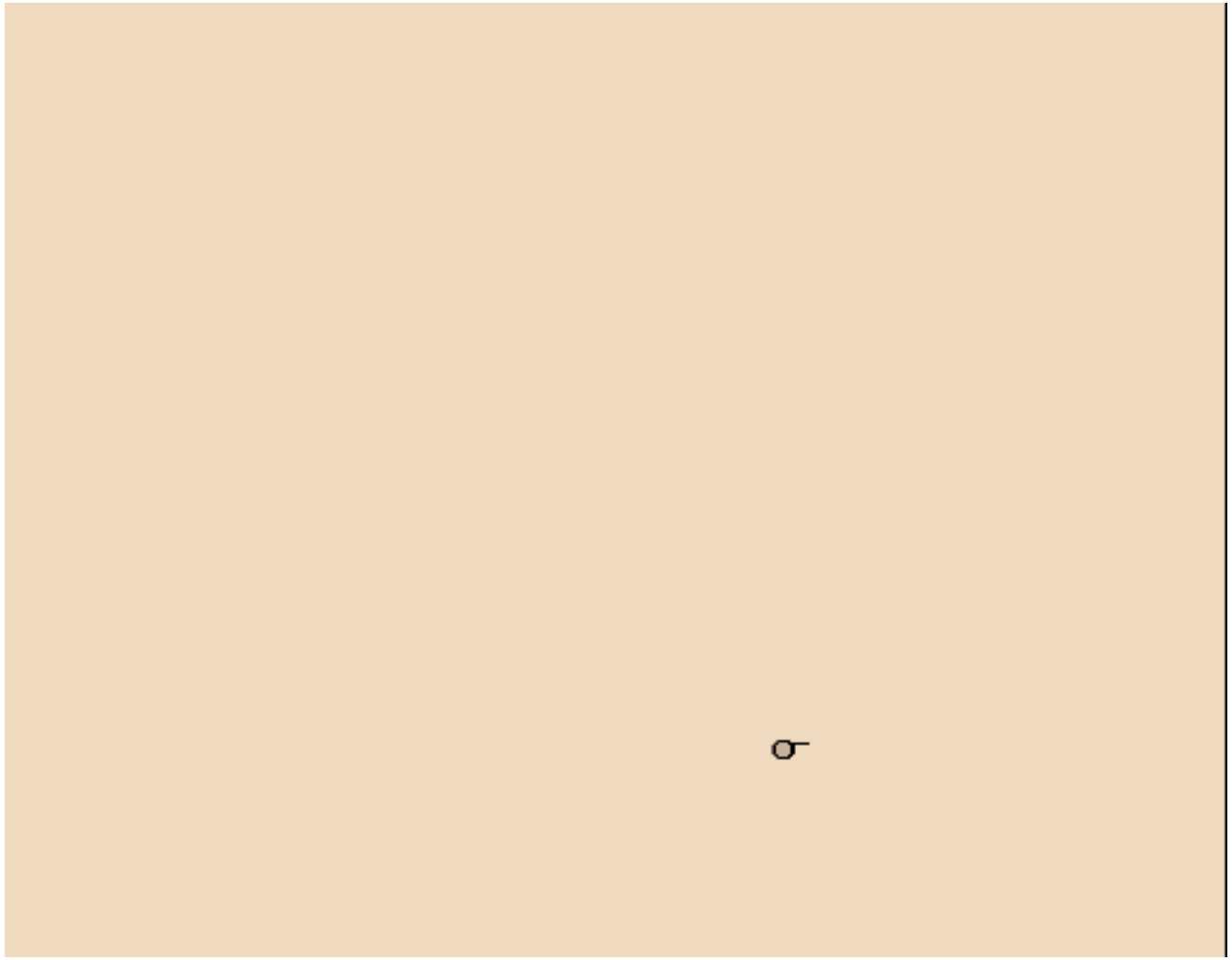
- Resample particles based on weight
- Filter resample
  - Assign each particle a weight depending on how well its estimate of the state agrees with the measurements and randomly draw particles from previous distribution based on weights creating a new distribution.

Particle Filter SLAM

- FastSLAM approach
  - Particle set:

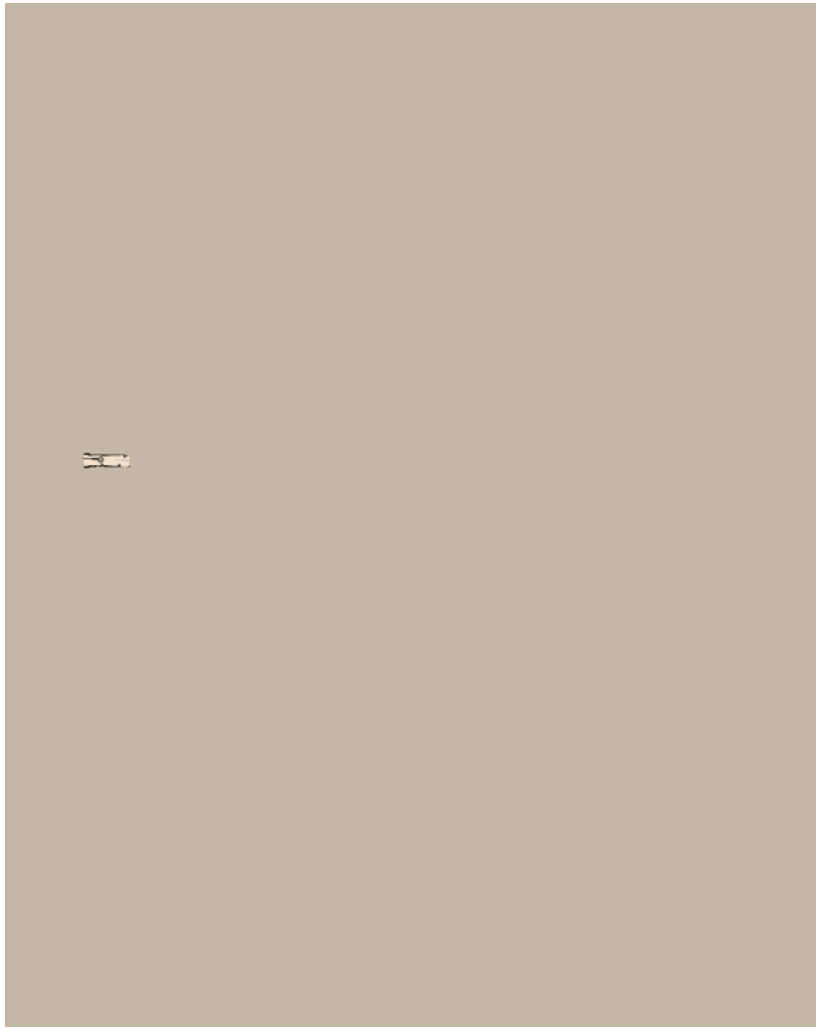


FAST SLAM example



Courtesy of S. Thrun

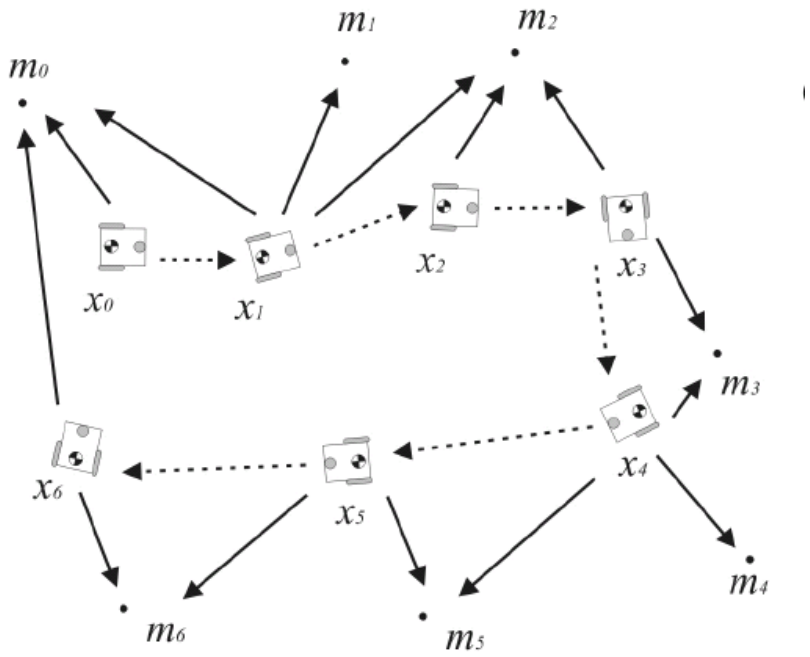
FAST SLAM example



Courtesy of S. Thrun

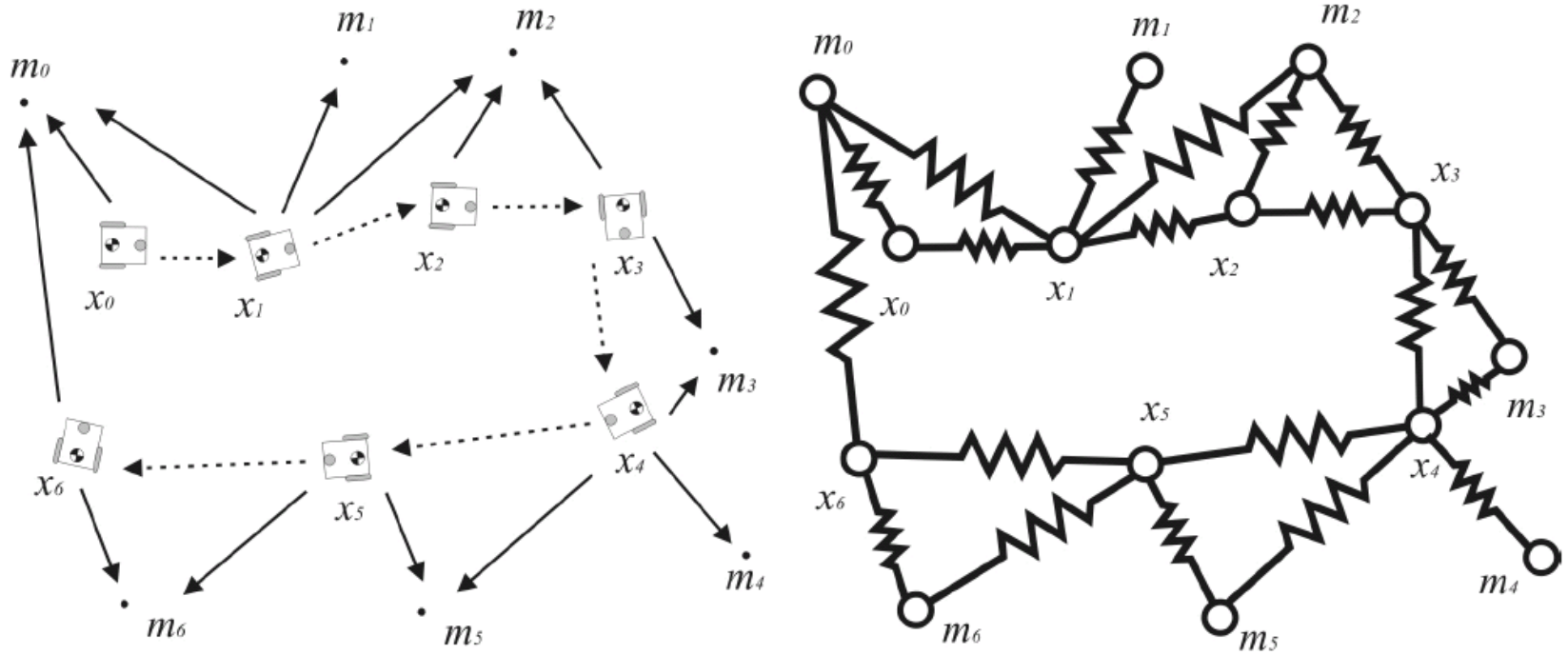
Graph-Based SLAM (1/3)

- SLAM problem can be interpreted as a sparse graph of nodes and constraints between nodes.
- The nodes of the graph are the robot locations and the features in the map.
- Constraints: relative position between consecutive robot poses , (given by the odometry input ***u***) and the relative position between the robot locations and the features observed from those locations.



Graph-Based SLAM (2/3)

- Constraints are not rigid but soft constraints!
- Relaxation: compute the solution to the full SLAM problem =>
  - Compute best estimate of the robot path and the environment map.
  - Graph-based SLAM represents robot locations and features as the nodes of an elastic net. The SLAM solution can then be found by computing the state of minimal energy of this net



Graph-Based SLAM (3/3)

- Significant advantage of graph-based SLAM techniques over EKF SLAM:
  - EKF SLAM: computation and memory for to update and store the covariance matrix is quadratic with the number of features.
  - Graph-based SLAM: update time of the graph is constant and the required memory is linear in the number of features.
- However, the final graph optimization can become computationally costly if the robot path is long.