Lecture 7-2

Going Places:

Navigation, Localization

The Robotics Primer (Ch. 19)

C.A. Berry

Course Announcements

- Quiz on **Monday, 5/04/09** on **Navigation and Localization**
- Lab 5 Demo due **Thursday, 4/30/09**
- Lab 5 Memo and code due by midnight on **Friday, 5/01/09**
- Upload memo and code to Angel
- Bring your laptop and robot everyday
- **DO NOT** unplug the network cables from the desktop computers or the walls

Quote of the Week

“We are what we repeatedly do. Excellence, then, is not an act, but a habit.”

Aristotle

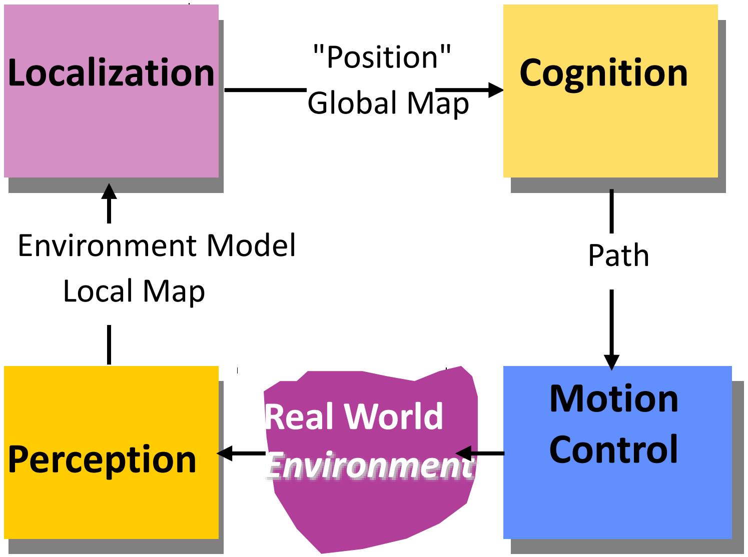
“Genius is 1 percent inspiration and 99 percent perspiration. As a result, genius is often a talented person who has simply done all of his homework.”

Thomas Edison

Navigation

Mobile Robot Navigation

- *Navigation* is one of the most challenging mobile robot competencies
- Successful *navigation* requires
  - Perception
  - Localization
  - Cognition
  - Motion control



Mobile Robot Navigation

- ***Perception***
  - The robot must interprets its sensors to extract meaningful data
- ***Localization***
  - The robot must determine it’s position in the environment
- ***Cognition***
  - The robot must decide how to act to achiever its goals
- ***Motion Control***
  - The robot must modulate its motor outputs to achieve the desired trajectory

Navigation

- *Navigation* refers to the way a robot finds its way in the environment
- This is a difficult problem because it is rooted in uncertainty
- It is difficult for a robot to know exactly where it is and how to get to its next destination

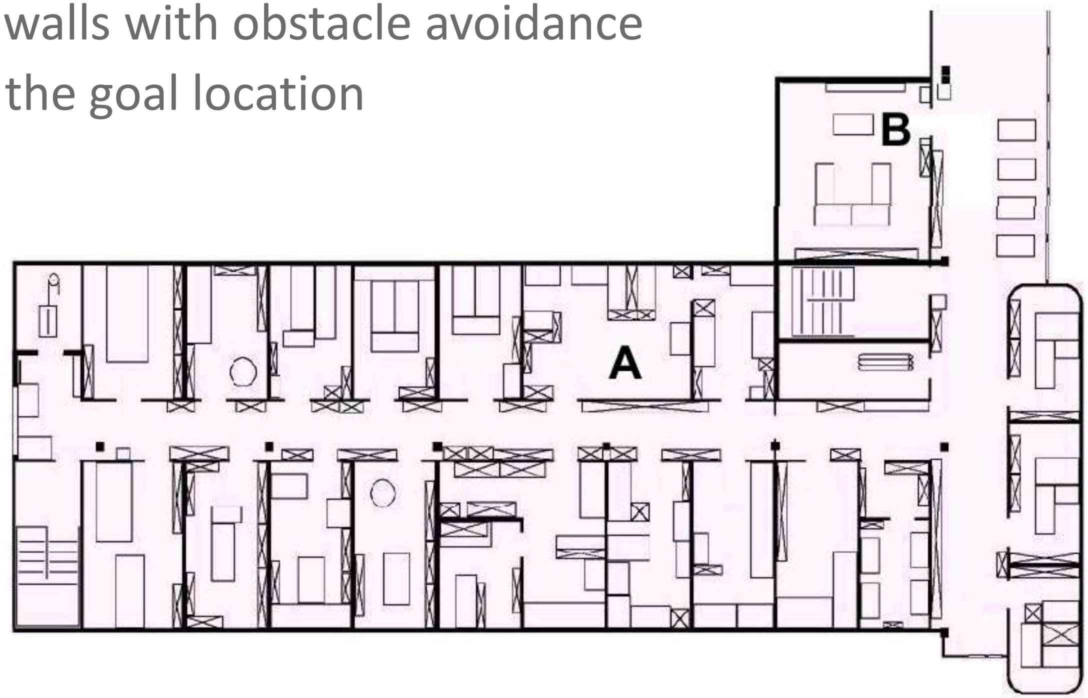
Navigation problems

- The robot may need to use a map for *path planning* assuming that the map is correct or that the world does not change
- The robot may need to also find itself on a map and this is the *localization* problem
- If the robot is trying to find a location on a map without prior knowledge and it must use a good search strategy this the *coverage* problem
- If the robot does not have a map of its world then it must build a map as it goes along and this is the *mapping* problem (i.e. SLAM)

Two navigation methods

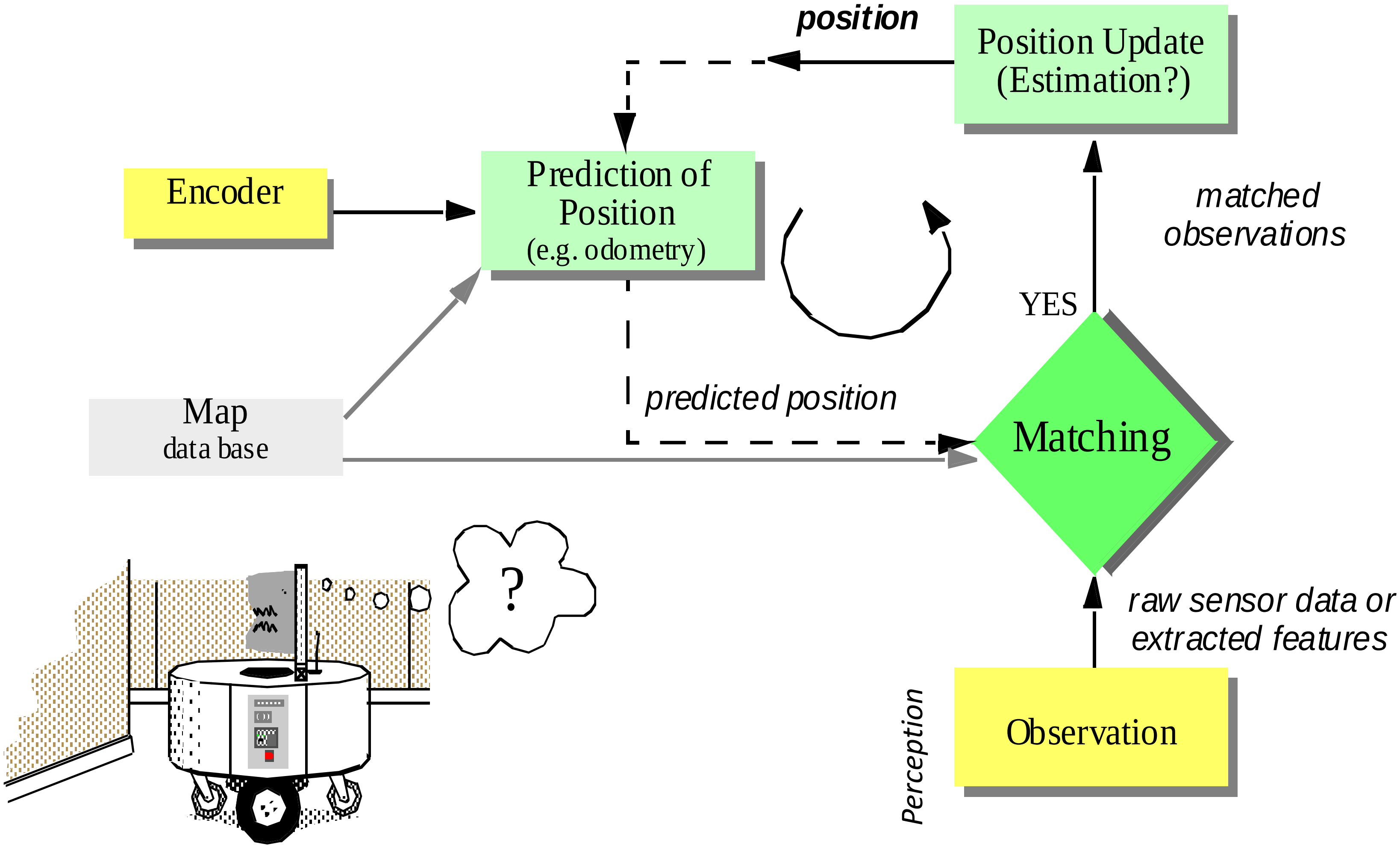
How to navigate between A and B,

- □ Use *localization* with respect to a map to navigate to the goal B
- □ Use *behavior-based navigation* without hitting obstacles
  - Follow walls with obstacle avoidance
  - Detect the goal location



Localization

Where am I?



position

Position Update (Estimation?)

Encoder

Prediction of Position (e.g. odometry)

matched observations

YES

Map data base

predicted position

Matching

?

raw sensor data or extracted features

Perception

Observation

Localization

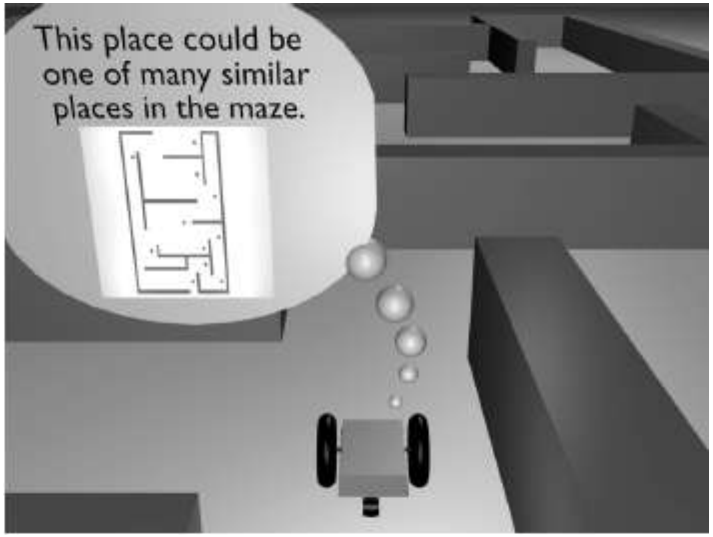
- One way for a robot is to use odometry or *path integration*
- Because of accumulation error, the robot will eventually need to recognize a landmark to reset the odometer
- This is localization relative to the start or reference point (i.e. GPS map)
- Localization is also treated as a *state estimation* problem
- State estimation is the process of estimating the state of a system from measurements

Localization Methods

- Odometry
- Using external sensors (beacons or landmarks)
- Probabilistic Map Based Localization

Localization problems

- The estimation process is indirect
- Measurements are noisy
- Measurements may not be available all the time



Frame of Reference

- Frame of reference is important
  - Local/Relative: Where am I vs. where was I?
  - Global/Absolute: Where am I relative to the world frame?
- Location can be specified in two ways
  - Geometric: Distances and angles
  - Topological: Connections among landmarks

Absolute Localization

- Proximity to reference
  - Landmarks/Beacons
- Angle to reference
  - Visual: manual triangulation from physical points
- Distance from reference
  - Time of Flight
    - Radio frequency (RF)
    - Global positioning system (GPS)
    - Acoustic
  - Signal Fading
    - Electromagnetic
    - Radio frequency
    - Acoustic

Relative Localization

- If you know your speed and direction, you can calculate where you are relative to where you were (integrate).
- Speed and direction might, themselves, be absolute (compass, speedometer), or integrated (gyroscope, accelerometer)
- Relative measurements are usually more accurate in the short term -- but suffer from accumulated error in the long term
- Most robotics research seems to focus on this

Challenges of Localization

- Knowing the absolute position (e.g. GPS) is not sufficient
- *Localization* may also be required on a relative scale with respect to humans
- *Cognition* may require more than position, it may need to build an environmental model, *map*, to plan a path to a goal

Sensor Noise

- Perception (sensors) and motion control (effectors) play an integral role in localization
  - Sensor noise
  - Sensor aliasing
  - Effector noise
  - Odometric position estimation

Belief Representation

- The fundamental issue that differentiates map-based localization systems is *representation*
  - *Map representation*
    - Robot's model of the environment, or a map
    - At what level of fidelity does the map represent the environment?
  - *Belief representation*
    - Robot's belief of its position on the map
    - Does the robot identify a single unique position?
    - Does the robot describe its position in terms of a set of possible positions?
    - How are multiple positions ranked

Localization Methods

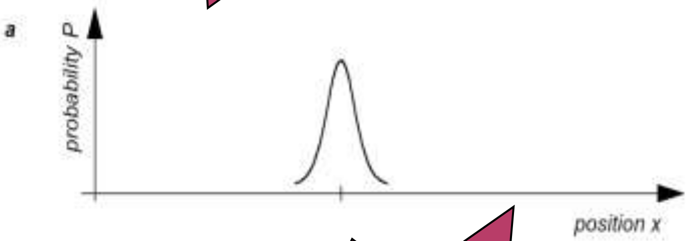
- Markov Localization:
  - Represent the robot’s belief by a probability distribution over possible positions and uses Bayes’ rule and convolution to update the belief whenever the robot senses or moves
- Gaussian
  - Represents the continuous hypothesis belief as a normal distribution

Belief Representation Characteristics

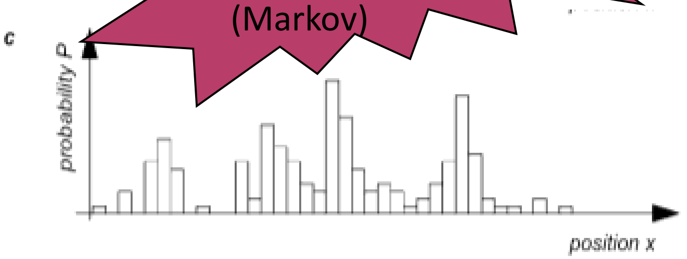
- Continuous
  - Precision bound by sensor data
  - Typically single hypothesis pose estimate
  - Lost when diverging (for single hypothesis)
  - Compact representation and typically reasonable in processing power.
- Discrete
  - Precision bound by resolution of discretization
  - Typically multiple hypothesis pose estimate
  - Never lost (when diverges converges to another cell)
  - Important memory and processing power needed. (not the case for topological maps)

Types of Belief Representation

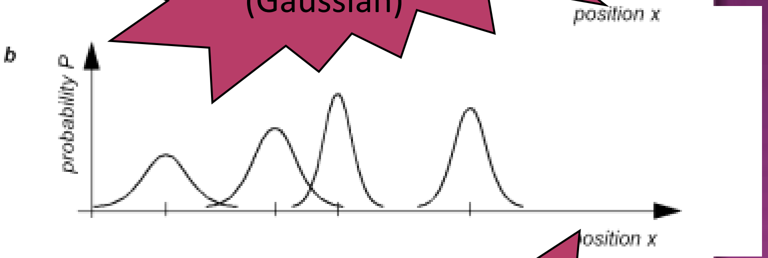
Continuous
Single hypothesis belief
(Gaussian)



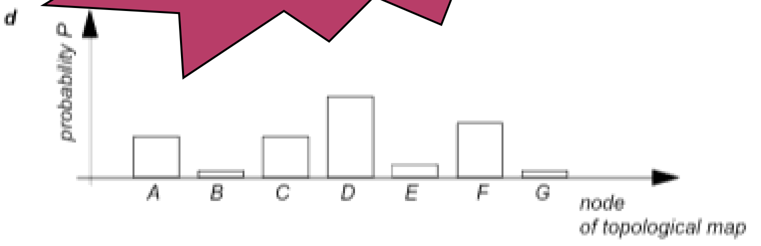
Discretized grid map
with probability values
for all possible robot
positions
(Markov)



Continuous
Multiple hypothesis
belief
(Gaussian)



Discretized topological map
with probability values
for all possible robot nodes
(Markov)



Single Hypothesis Belief

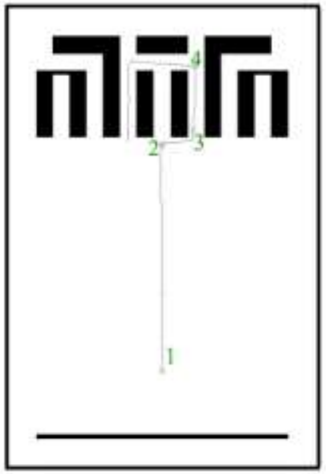
- Advantages:
  - Given a unique belief, there is no position ambiguity
  - Facilitates decision-making at robot’s cognitive level (e.g. path planning)
- Disadvantages:
  - Robot motion induces uncertainty due to effector and sensor noise
  - Forcing the position update to always generate a single hypothesis of position is challenging

Multiple Hypothesis Belief

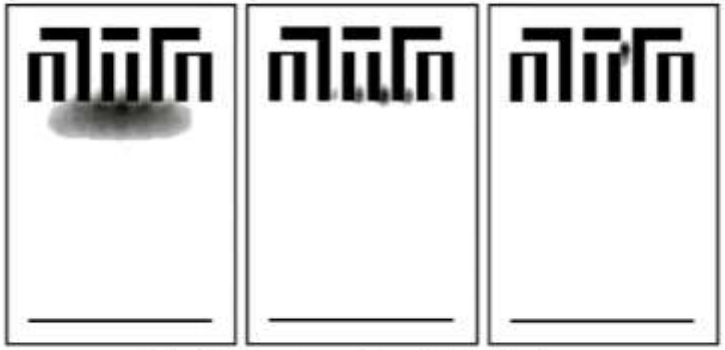
- The robot tracks an infinite set of possible positions
- This set can be described geometrically as a convex polygon positioned on a 2D map (continuous or discrete)
- In this method, the possible robot positions are not ranked
- To rank the positions requires a model of the beliefs as a mathematical distribution (Gaussian probability density function)

Multiple Hypothesis Grid-Based Representation

- There are discrete markers for each possible position
- Each position is noted along with a confidence or probability parameter
- Thousands of possible positions for a highly tessellated map



Path of the robot

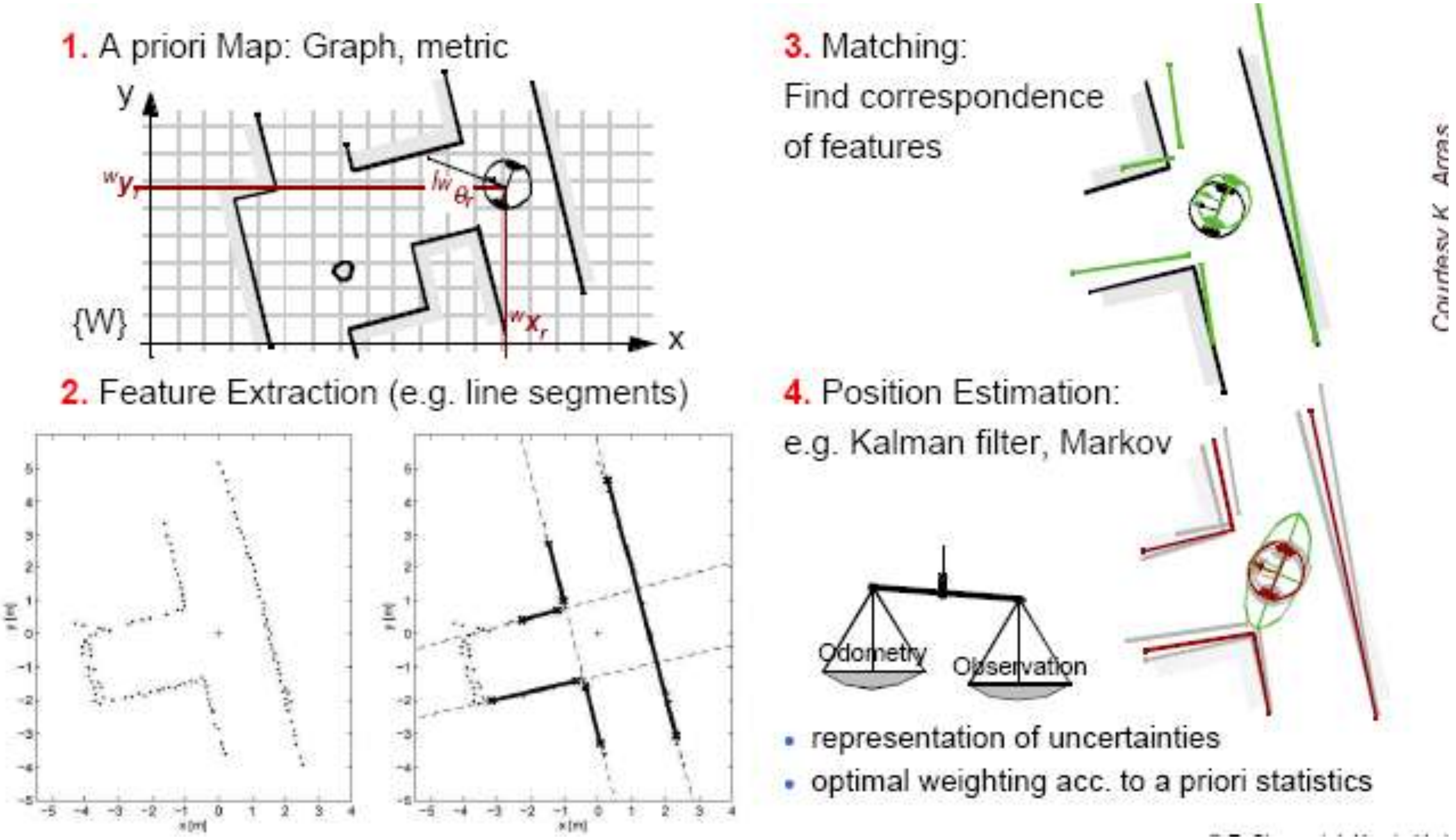


Belief states at positions 2, 3 and 4

Multiple Hypothesis Grid-Based Advantages and Disadvantages

- Advantages:
  - Robot maintains a sense of position while explicitly annotating its own uncertainty about the position
  - Partial information from sensors and effectors can update the belief
  - Robot is able to explicitly measure its own degree of uncertainty regarding position
- Disadvantages:
  - In decision making, how does the robot decide what to do next?
  - Each position must have an associated probability
  - Computationally expensive

Methods for Localization: Quantitative Metric Approach



1. A priori Map: Graph, metric

2. Feature Extraction (e.g. line segments)

3. Matching:
Find correspondence
of features

4. Position Estimation:
e.g. Kalman filter, Markov

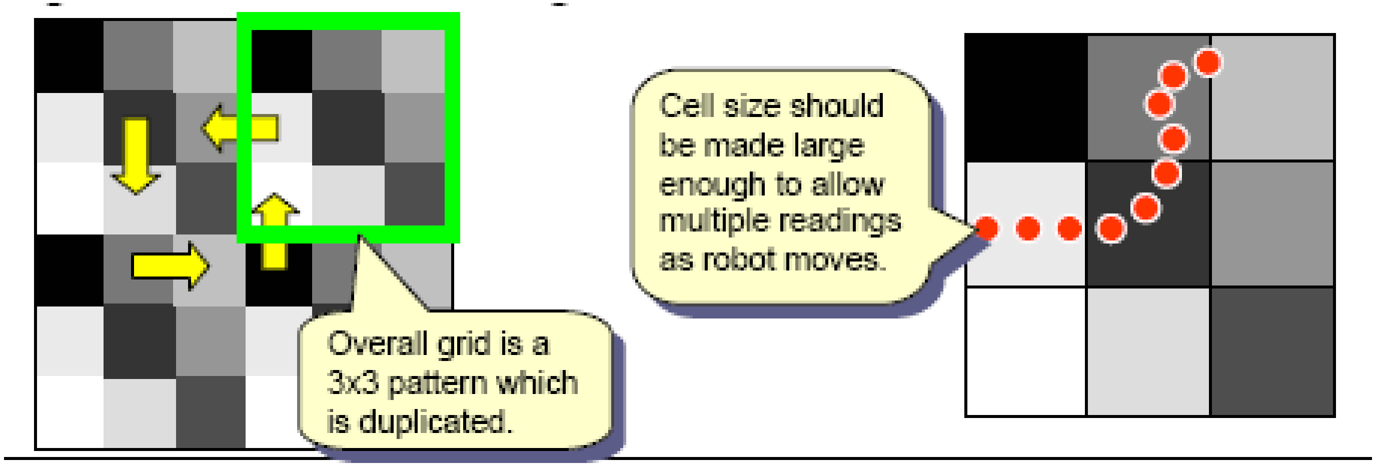
- representation of uncertainties
- optimal weighting acc. to a priori statistics

Grid-based Localization

- Another strategy for position estimation is to do grid tracking
- Place a grid on the floor with clearly identifiable cells
- The robot senses change from one cell to another

Grid design

- A robot is equipped with a light sensor
- Grid must be designed to distinguish changes from one cell to another
- Must maximize the contrast between adjacent cells
- Grid cells must be larger when the robot moves faster



Overall grid is a 3x3 pattern which is duplicated.

Cell size should be made large enough to allow multiple readings as robot moves.

Grid tracking

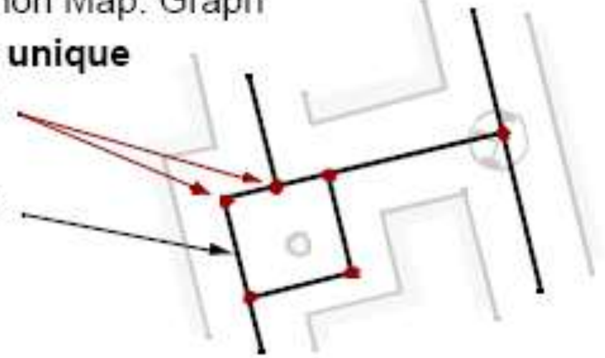
- Advantages
  - Can re-confirm location after short distances, eliminate errors within 1 cell range
  - Simple to implement
- Disadvantages
  - Cell size limits accuracy
  - Requires many sensor readings and large cells for truly reliable estimations
  - Requires modification of the environment
  - Result depends on print quality and sensor calibration

Topological Localization Quantitative Approach

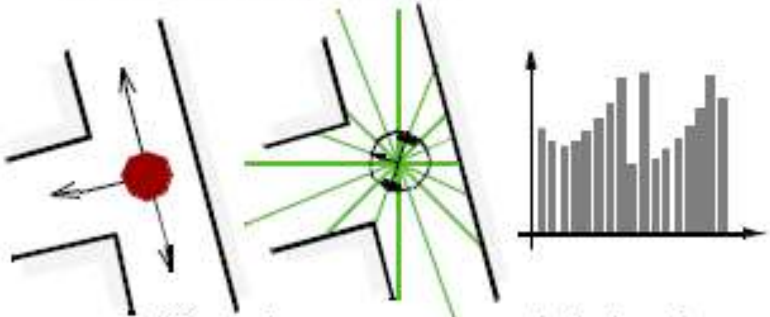
1. A priori Map: Graph

locally **unique** points

edges



2. Method for determining the **local uniqueness**

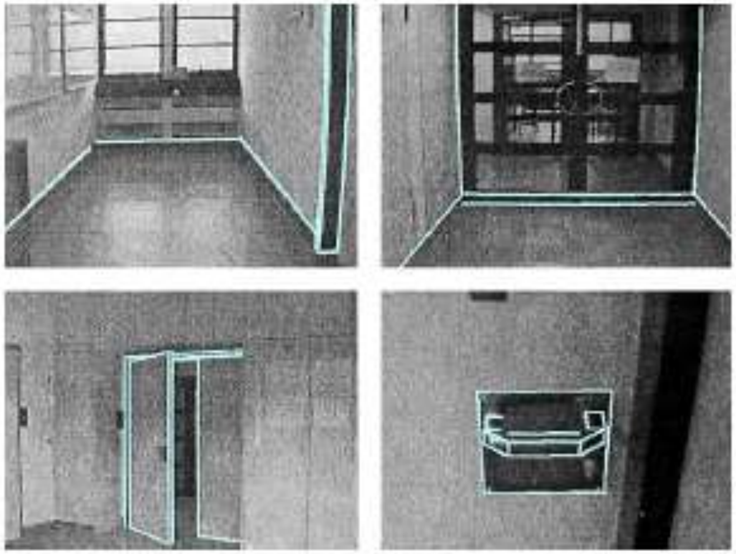


e.g. striking changes on raw data level or highly distinctive features

3. Library of **driving behaviors**

e.g. wall or midline following, blind step, enter door, application specific behaviors

Example: Video-based navigation with natural landmarks



Courtesy of [Lanser et al. 1996]

Probabilistic Map-Based Localization

- One geometric approach to *multi-hypothesis representation* identifies the possible positions of a robot
- *Probabilistic techniques* identifies probabilities with the possible robot positions
- Two classes of probabilistic localization are:
  - *Markov localization*
  - *Kalman filter localization*

Probabilistic Localization Classes

- *Markov localization*
  - Uses an explicitly defined probability distribution across all robot positions
- *Kalman filter localization*
  - Uses a Gaussian probability density representation of robot position and scan matching for localization
  - Unlike Markov, it does not independently consider each possible robot pose
  - Kalman results from the Markov axioms if the robot's position uncertainty is assumed to be Gaussian

Updating Robot Position

- The process of updating robot position based upon *proprioceptive* and *exteroceptive* sensor values are separated logically into a general two-step process
  - *Action Update*
    - Proprioceptive
    - Represents the application of some action model
  - *Perception Update*
    - Exteroceptive
    - Represents the application of some perception model

Markov and Kalman Filter Localization

- *Markov localization* is the robot's belief state usually represented as separate probability assignment for every possible pose on the map
  - Special case of probabilistic state estimation applied to mobile robot localization
- *Kalman filter localization* represents the robot's belief state using a single, well-defined Gaussian probability density function
  - It retains a μ and σ parameterization of the robot's belief about position with respect to the map

Markov vs. Kalman Filter Localization

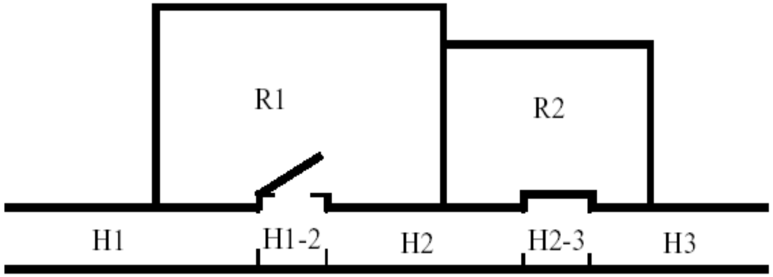
- Markov
  - Allows localization starting from any unknown position
  - Recovers from ambiguous situations because the robot can track multiple, complete disparate possible positions
  - Requires discrete representation of the space (geometric grid or topological graph)
  - Required memory and computational power can limit precision and map size

- Kalman Filter
  - Tracks the robot from a known position
  - Is both precise and efficient
  - Can be used in continuous world representations
  - If robot uncertainty becomes too large and not unimodal, it can fail to capture the multitude of possible robot positions and can become irrevocably lost

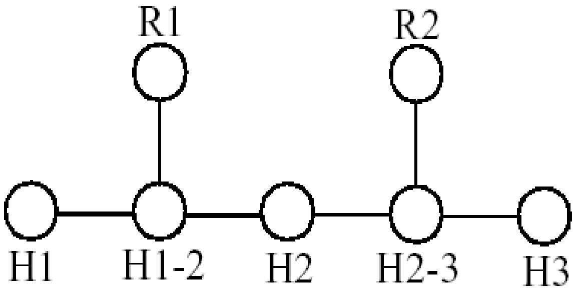
Markov Localization Topological Map

- Identical in abstraction and information to the environment map
- Decision involves assignment of nodes and connectivity between nodes
- Node boundaries are marked by doorways, hallways, and foyers
- Note there is no geometric information on the nodes

Environment map

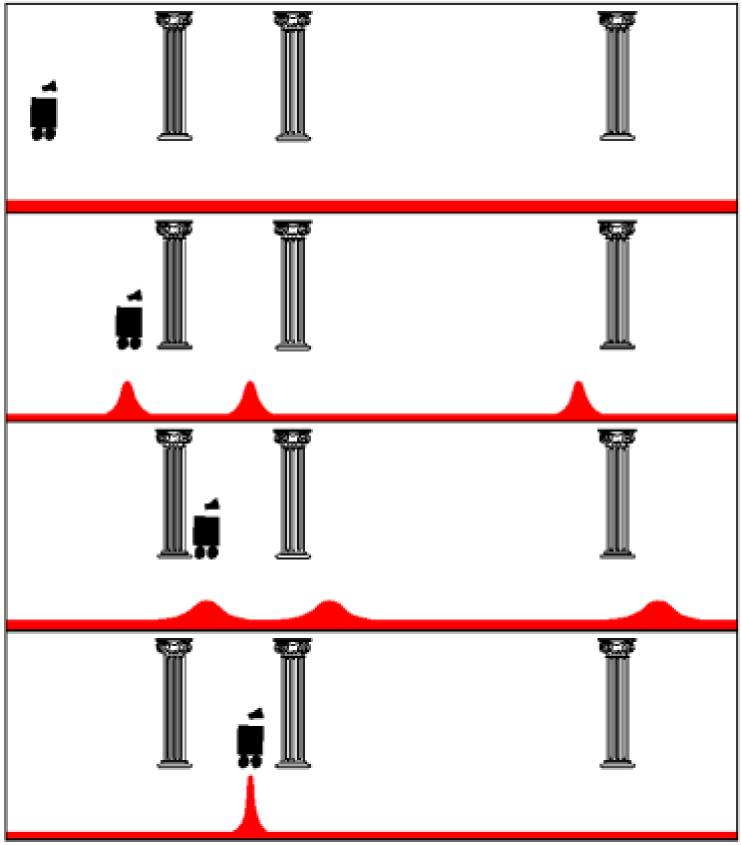


Topologic map



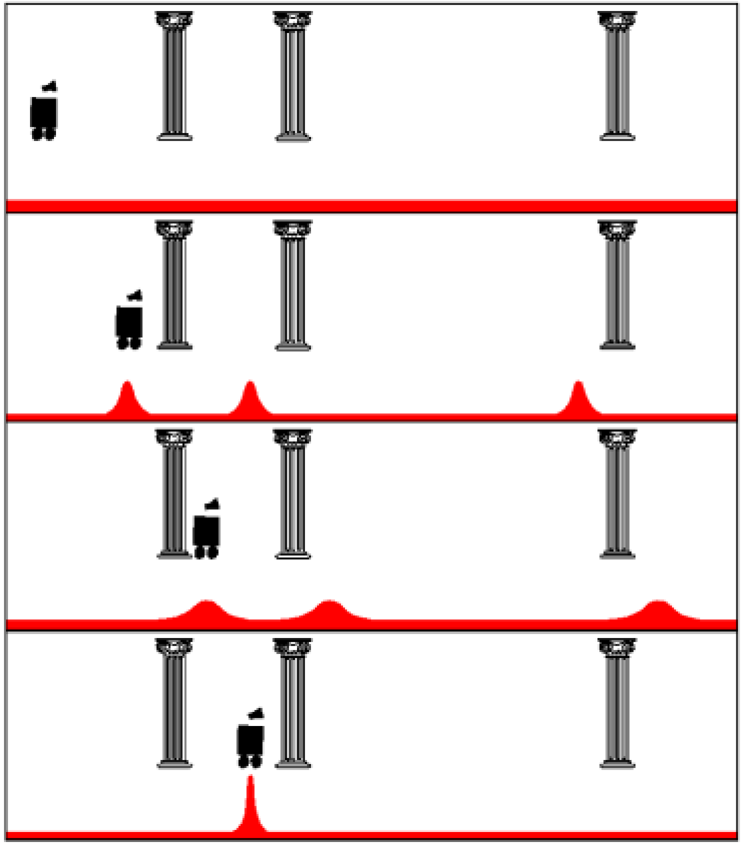
Markov Localization Grid Map

1. Start
   - No knowledge at start, thus there is
     a uniform probability distribution
2. Robot perceives first pillar
   - Seeing only one pillar, the probability
     being at pillar 1, 2 or 3 is equal.
3. Robot moves
   - Action model enables the estimate of the new probability distribution based
     on the previous one and the motion.
4. Robot perceives second pillar
   - Based on all prior knowledge the probability being at pillar 2 becomes
     dominant

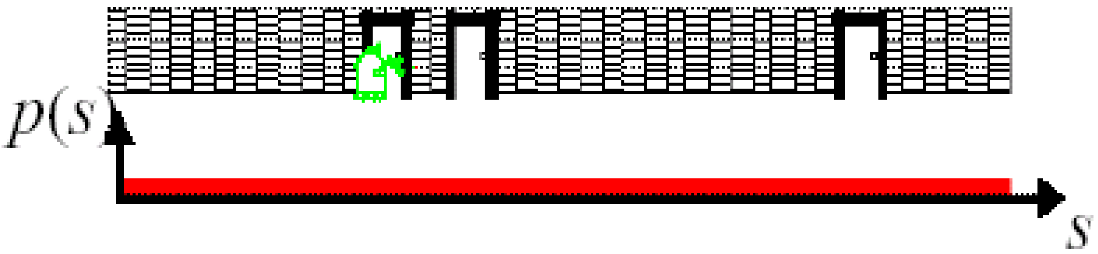


Markov Localization Grid Map

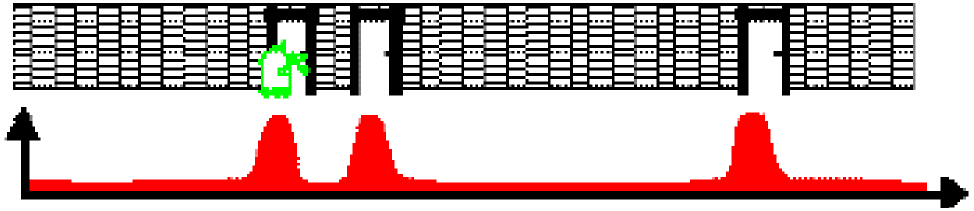
- As the robot encounters one pillar and then a second pillar, the probability density function over possible positions becomes multimodal, unimodal and then sharply defined
- The ability of a Markov localization system to *localize the robot from an initially lost belief state* is its key distinguishing feature
- This is a challenging application because of the dynamic nature of the environment



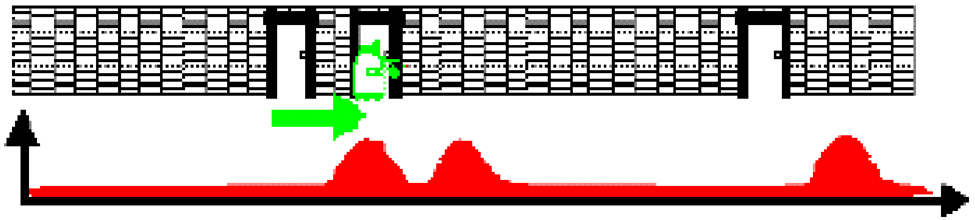
Markov Localization Examples



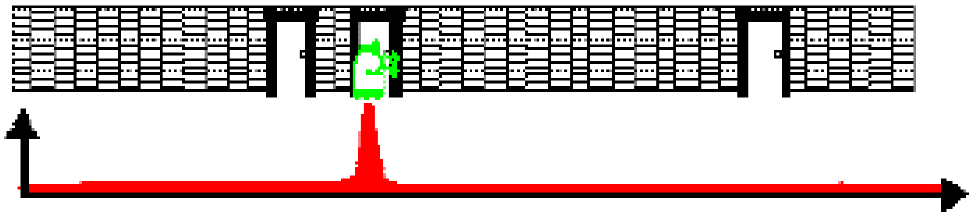
The robot is placed somewhere in the environment but it is not told its location



The robot queries its sensors and finds out it is next to a door



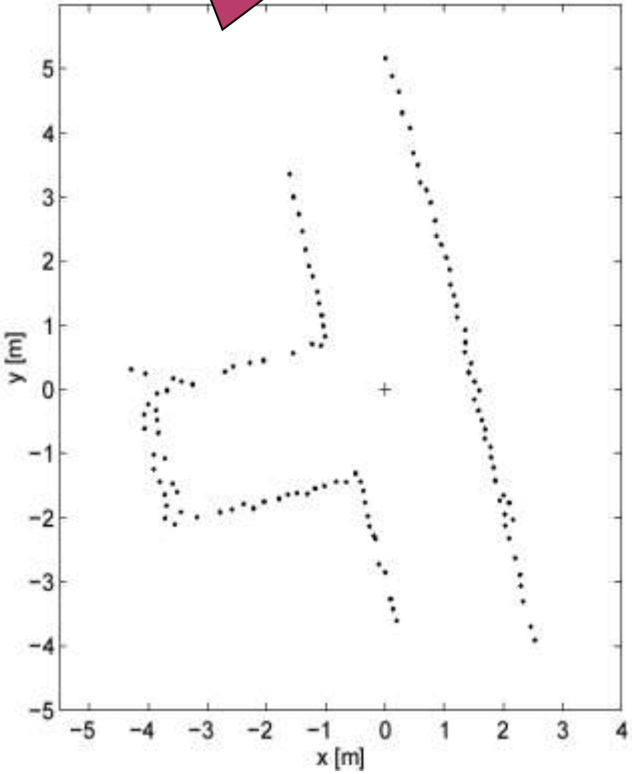
The robot moves one meter forward. To account for inherent noise in robot motion the new belief is smoother

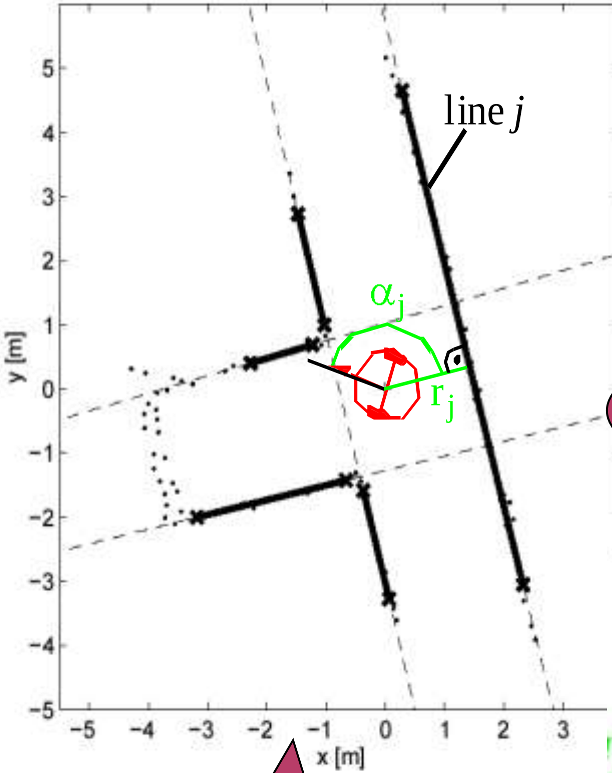


The robot queries its sensors and again it finds itself next to a door

Kalman Filter Localization

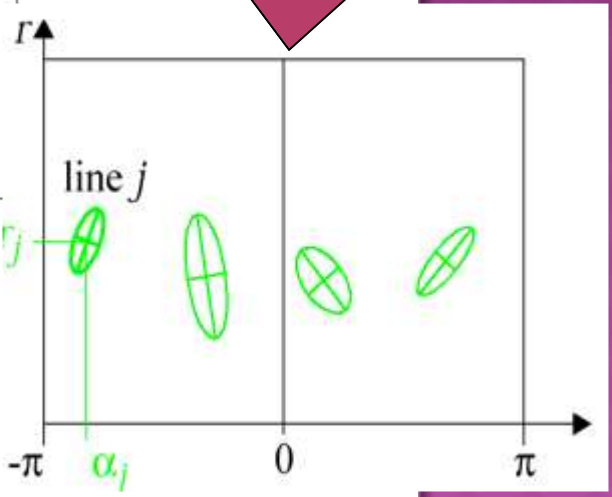
raw data from the laser scanner at time k+1, extracted lines





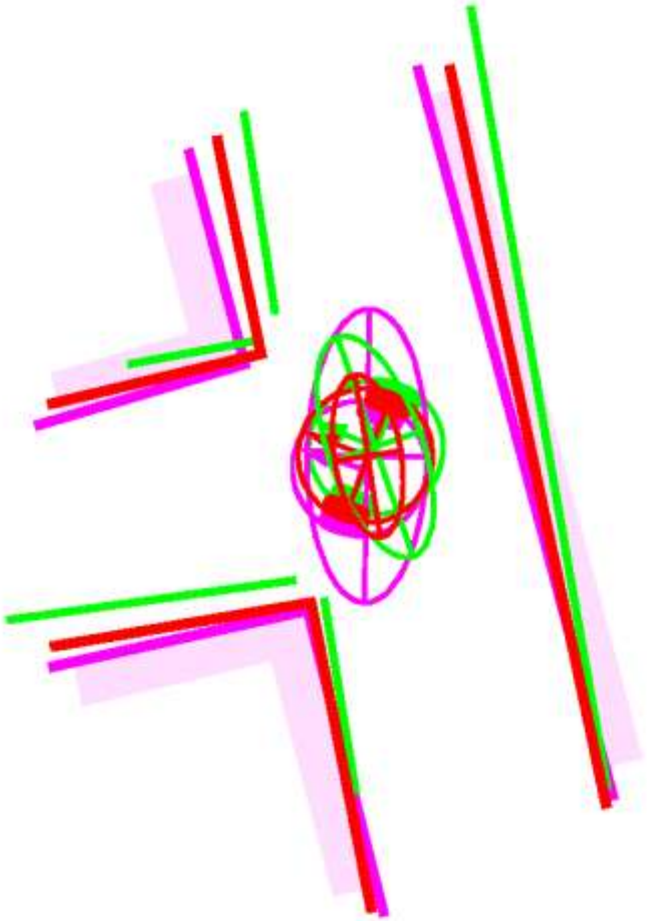
lines extracted from the raw data

the extracted lines uncertainties represented in the model space



Kalman Filter Localization Case Study

- Kalman filter estimation of the new robot position
  - By fusing the prediction of robot position (magenta) with the innovation gained by the measurements (green)
  - we get the updated estimate of the robot position (red)
- this final pose estimate corresponds to the weighted sum of the
  - pose estimates of each matching pairing of observed and predicted features
  - robot position estimation based on odometry and observation positions



Autonomous Map Building (SLAM)

A robot that localizes successfully has the right sensors for detecting the environment and the robot ought to build its own map

- starting from an arbitrary initial point,
- a mobile robot should be able to autonomously explore the environment with its sensors,
- gain knowledge about it,
- interpret the scene,
- build an appropriate map
- and localize itself relative to this map

Simultaneous Localization and Mapping (SLAM)

- When the robot does not have a map and does not know where it is and chooses to build a map as it goes along in order to localize this is called *simultaneous localization and mapping (SLAM)*
- SLAM is also referred to as concurrent mapping and localization (CML)
- *Simultaneous Localization and Mapping (SLAM)* is one of the most difficult problems specific to mobile robot systems
- SLAM is one of the most difficult tasks in robotics because it is based upon the interaction between the position updates it uses to localize and the mapping actions

SLAM Dilemmas

- If the robot updates its position based on an observation of an imprecisely known feature, the results position estimate becomes correlated with the feature location estimate
- The map becomes correlated with a position estimation if an observation taken from an imprecisely known position is used to update or add a feature to the map
- The complete and optimal solution has to consider correlations between *position* and *feature location estimation*
- Cross-correlated maps are called *stochastic maps*