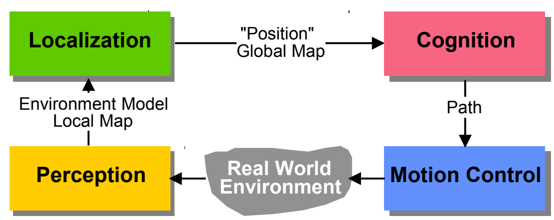
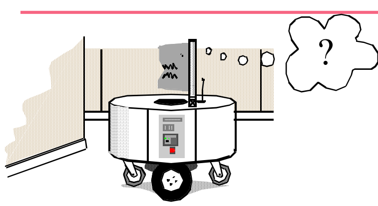
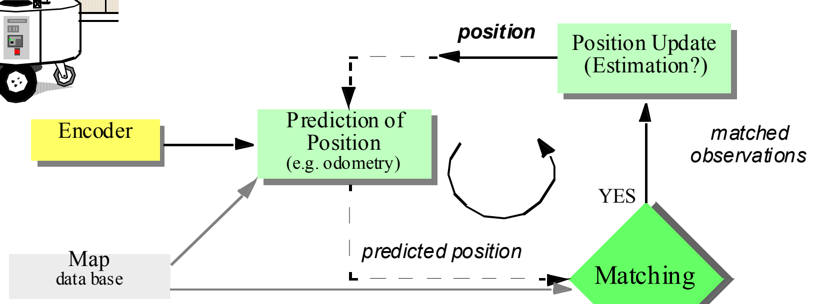
Localization and Map Building

- Noise and aliasing; odometric position estimation
- To localize or not to localize
- Belief representation
- Map representation
- Probabilistic map-based localization
- Other examples of localization system *(maybe)*
- Autonomous map building *(maybe)*

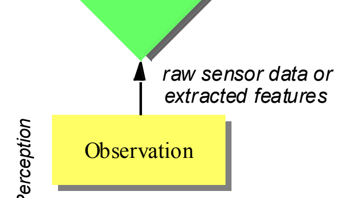


Localization, Where am I?





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



Challenges of Localization

- Knowing the absolute position (e.g. GPS) is not sufficient
- Localization in human-scale in relation with environment
- Planning in the *Cognition* step requires more than only position as input
- Perception and motion plays an important role
  - *Sensor noise*
  - *Sensor aliasing*
  - *Effector noise*
  - *Odometric position estimation*

Sensor Noise

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

Sensor Aliasing

- In robots, non-uniqueness of sensors readings is the norm
  - *Note that this is the case even if the sensors are working perfectly (which of course they typically don't).*
- Even with multiple sensors, there is a many-to-one mapping from environmental states to robot's perceptual inputs
- Therefore the amount of information perceived by the sensors is generally insufficient to identify the robot's position from a single reading
  - *Robot's localization is usually based on a series of readings*
  - *Sufficient information is recovered by the robot over time*

Effector Noise: Odometry, Dead 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: Error sources

deterministic (systematic) — non-deterministic (non-systematic)

- *deterministic errors can be eliminated by proper calibration of the system.*
- *non-deterministic errors have to be described by error models and will always leading to uncertain position estimate.*

- Major Error Sources:
  - *Limited resolution during integration (time increments, measurement resolution …)*
  - *Misalignment of the wheels (deterministic)*
  - *Unequal wheel diameter (deterministic)*
  - *Variation in the contact point of the wheel*
  - *Unequal floor contact (slipping, not planar …)*
  - *…*

Odometry: Classification of Integration Errors

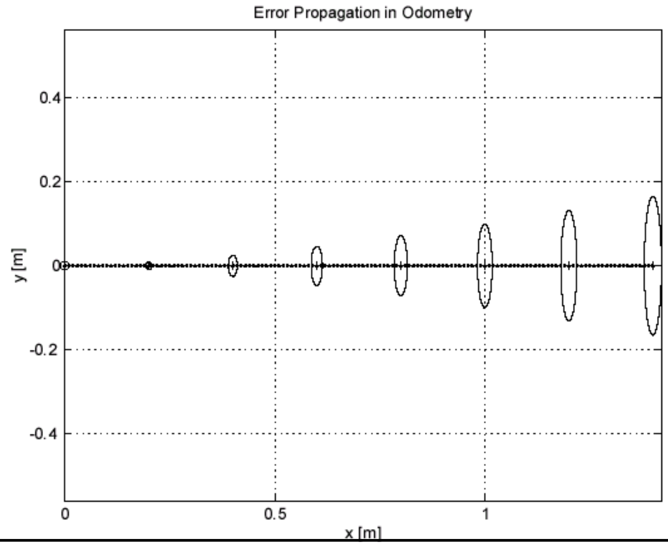
- Range error: integrated path length (distance) of the robots movement
  - *sum of the wheel movements*
- Turn error: similar to range error, but for turns
  - *difference of the wheel motions*
- Drift error: difference in the error of the wheels leads to an error in the robots angular orientation

Over long periods of time, turn and drift errors far outweigh range errors!

- *Consider moving forward on a straight line along the* **x** *axis. The error in the* **y***-position introduced by a move of* **d** *meters will have a component of* **dsin**$\Delta\theta$*, which can be quite large as the angular error* $\Delta\theta$ *grows.*

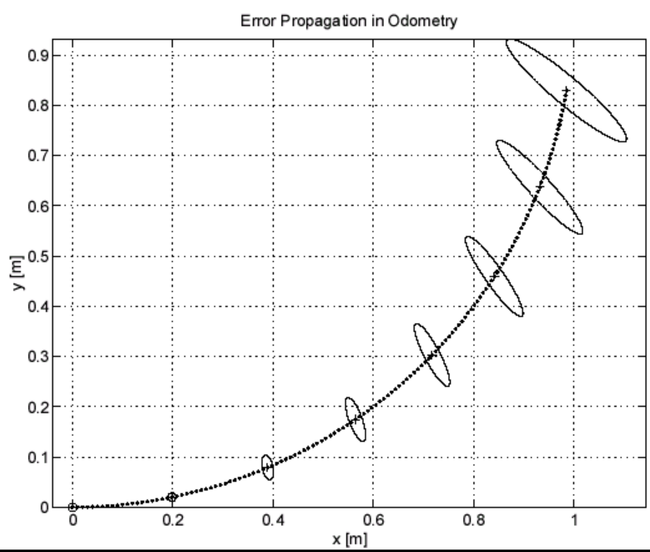
Odometry: Growth of Pose uncertainty for Straight Line Movement

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



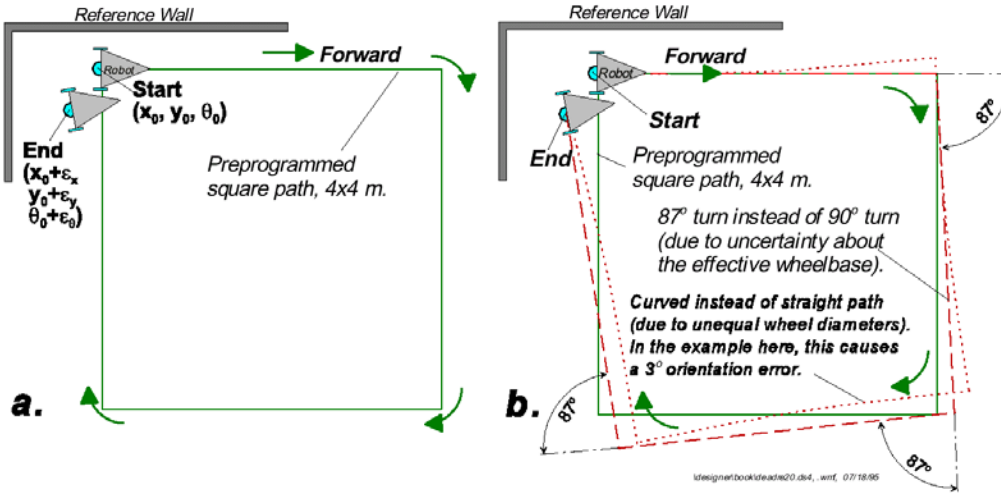
Odometry: Growth of Pose uncertainty for Movement on a Circle

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



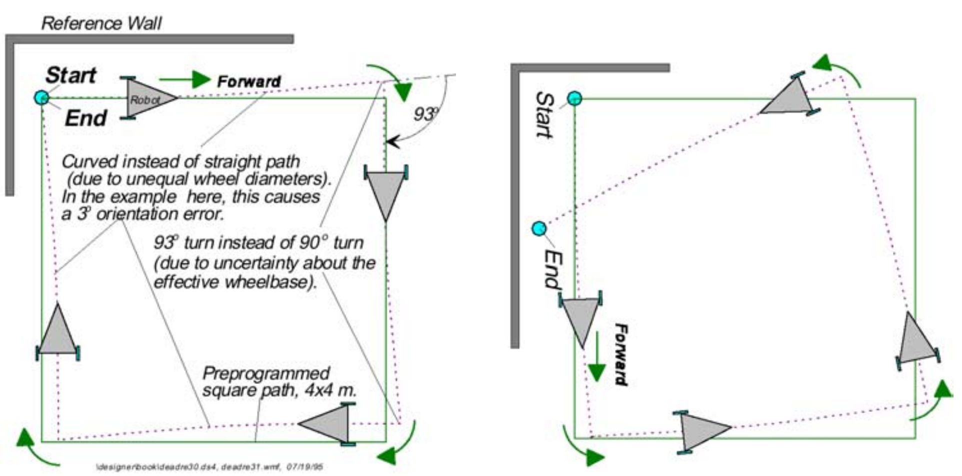
Odometry: Calibration of Errors I (Borenstein [5])

- The unidirectional square path experiment



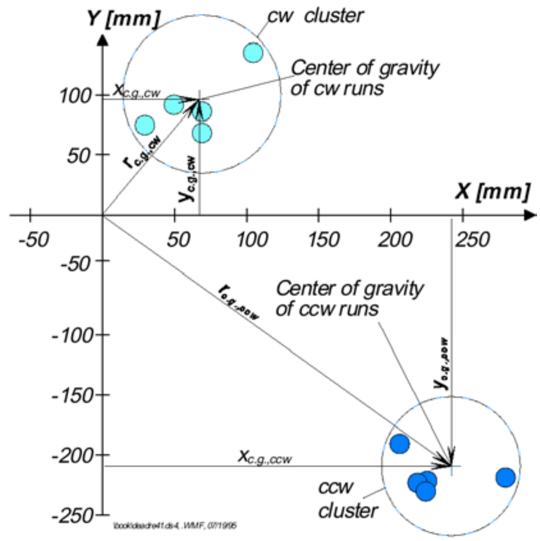
Odometry: Calibration of Errors II (Borenstein [5])

- The bi-directional square path experiment



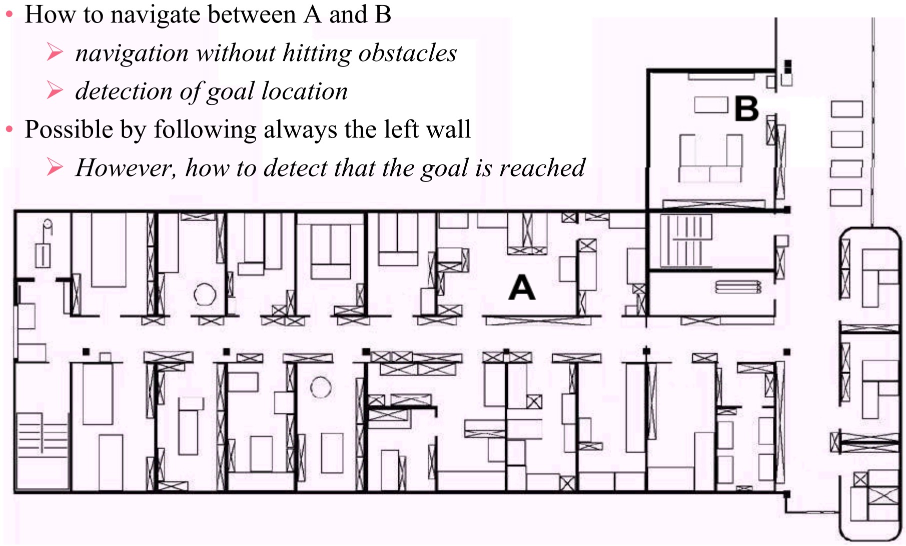
Odometry: Calibration of Errors III (Borenstein [5])

- The deterministic and non-deterministic errors



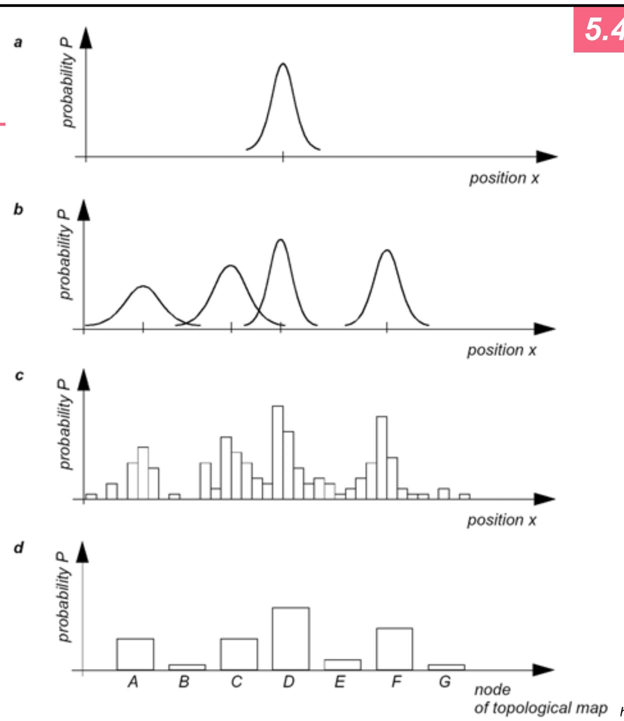
To localize or not?

- How to navigate between A and B
  - ➢ *navigation without hitting obstacles*
  - ➢ *detection of goal location*
- Possible by following always the left wall
  - ➢ *However, how to detect that the goal is reached*



Belief Representation

- a) Continuous map
  with *single hypothesis*
- b) Continuous map
  with *multiple hypothesis*
- d) Discretized map
  with probability distribution
- d) Discretized topological
  map with probability
  distribution

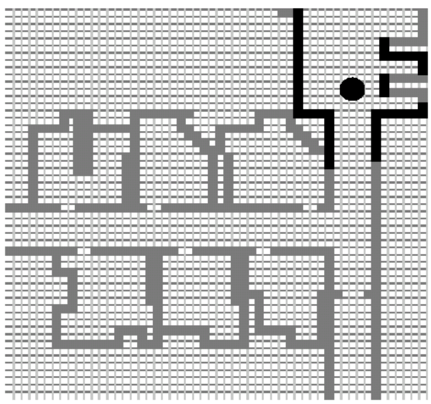


Belief Representation: Characteristics

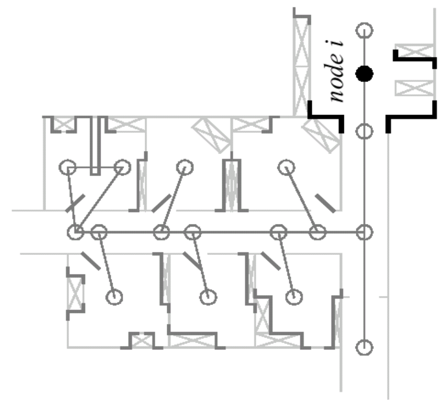
- Continuous
  - *Precision bound by sensor data*
  - *Typically single hypothesis pose estimate*
  - *Lost when diverging (for single hypothesis)*
  - *Compact representation and typically reasonable in processing power.*
- Discrete
  - *Precision bound by resolution of discretisation*
  - *Typically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

Single-hypothesis Belief – Grid and Topological Map

c)



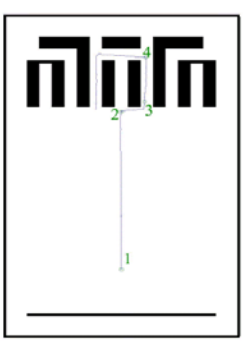
d)



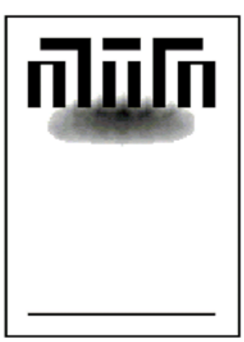
Grid-base Representation - Multi Hypothesis

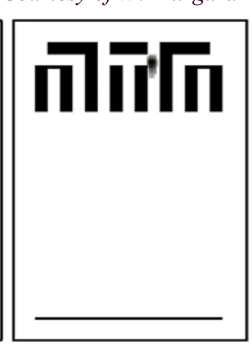
- Grid size around 20 cm^2.

Courtesy of W. Burgard



Path of the robot





Belief states at positions 2, 3 and 4

Map Representation

1. Map precision vs. application
2. Features precision vs. map precision
3. Precision vs. computational complexity

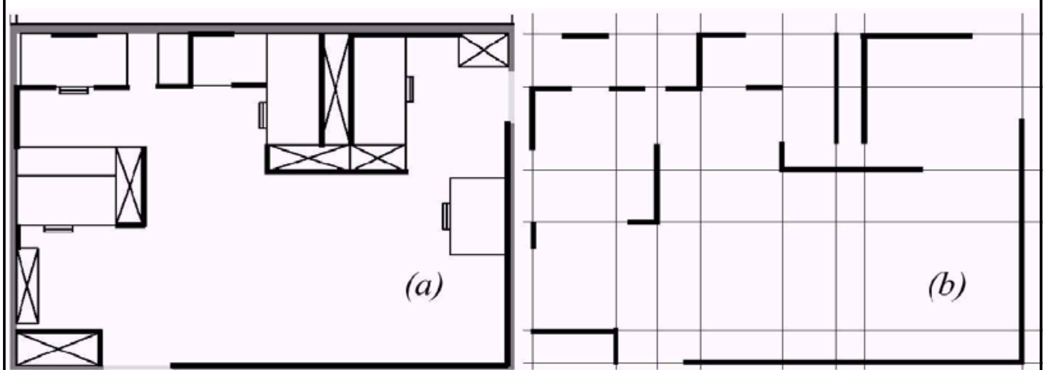
- Continuous Representation
- Decomposition (Discretization)

Representation of the Environment

- Environment Representation
  - *Continuous Metric* $\rightarrow x, y, \theta$
  - *Discrete Metric* $\rightarrow$ *metric grid*
  - *Discrete Topological* $\rightarrow$ *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*

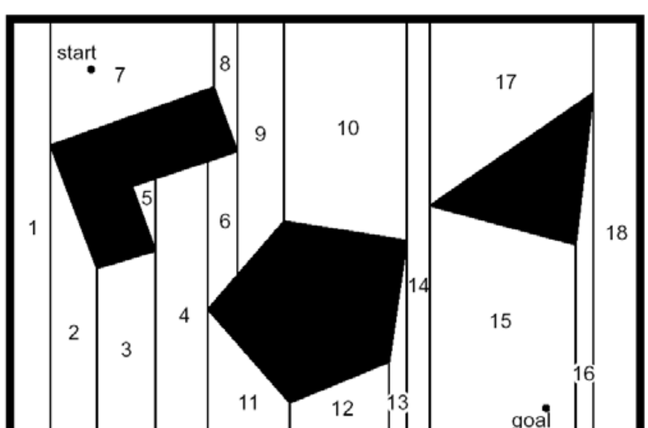
Map Representation: Continuous Line-Based

a) Architecture map
b) Representation with set of infinite lines



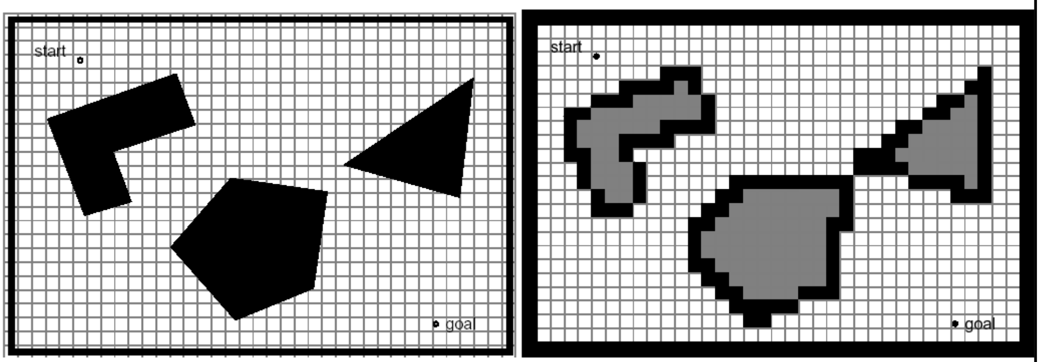
Map Representation: Decomposition (1)

- Exact cell decomposition



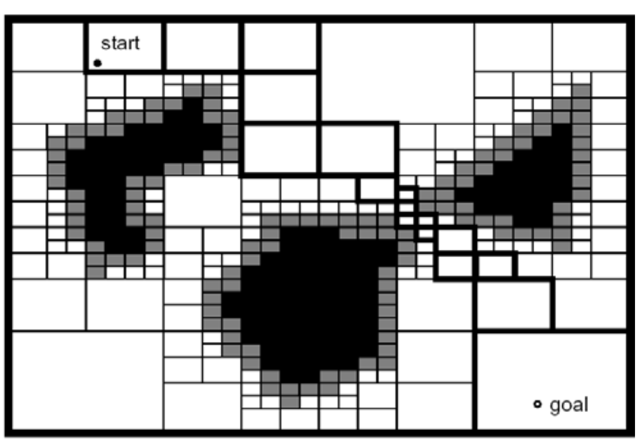
Map Representation: Decomposition (2)

- Fixed cell decomposition
  - *Narrow passages disappear*



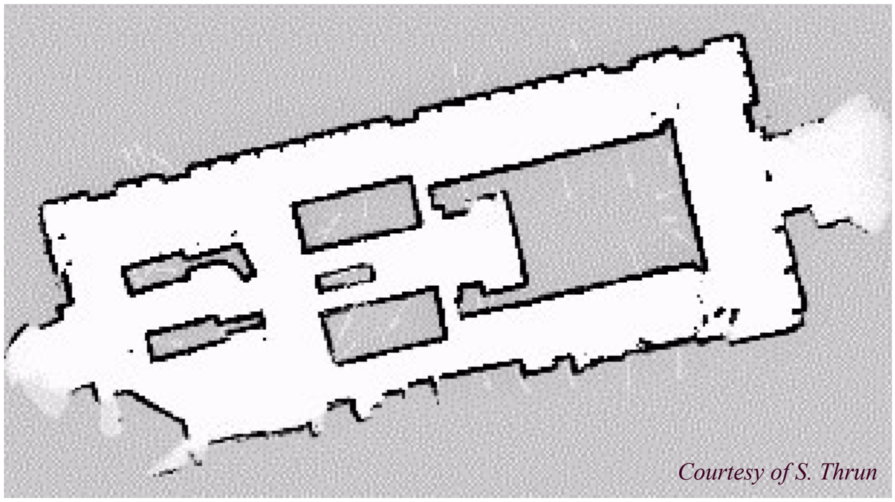
Map Representation: Decomposition (3)

- Adaptive cell decomposition



Map Representation: Decomposition (4)

- Fixed cell decomposition – Example with very small cells



Courtesy of S. Thrun

Probabilistic, Map-Based Localization (1)

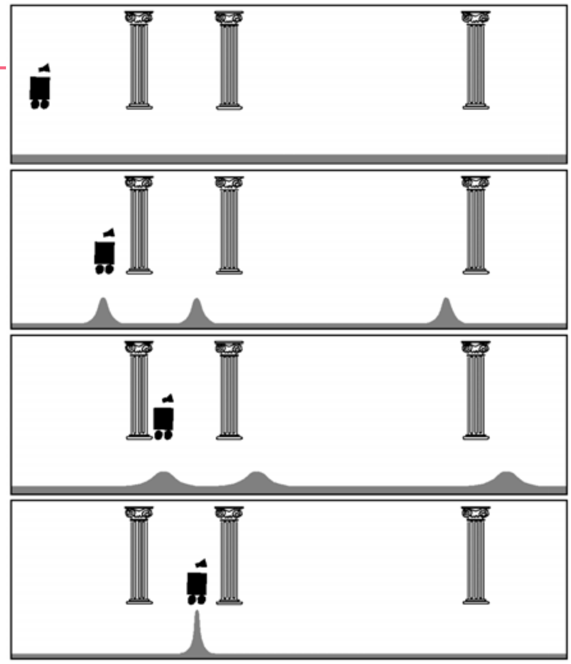
- Consider a mobile robot moving in a known environment.

- As it start to move, say from a precisely known location, it might keep track of its location using odometry.
- However, after a certain movement the robot will get very uncertain about its position.

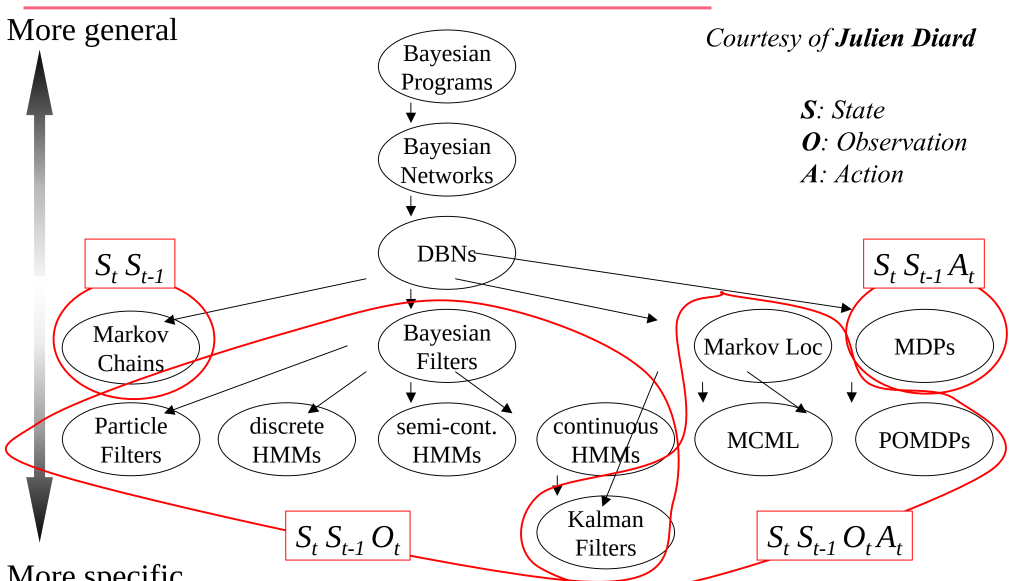
➔ update using an observation of its environment.

- observation lead also to an estimate of the robots position which can than be fused with the odometric estimation to get the best possible update of the robots actual position.

- Improving belief state by moving



Bayesian Approach: A taxonomy of probabilistic models



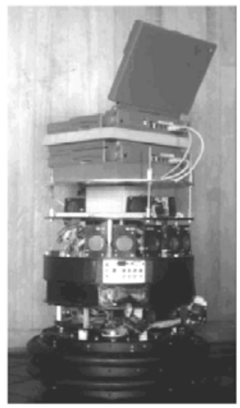
Interlude 1 …

Markov Localization

- Markov localization uses an explicit, discrete representation for the probability of *every* position in the state space.
- This is usually done by representing the environment by a grid or a topological graph with a finite number of possible states (positions).
- During each update, the probability for each state (element) of the entire space is updated.

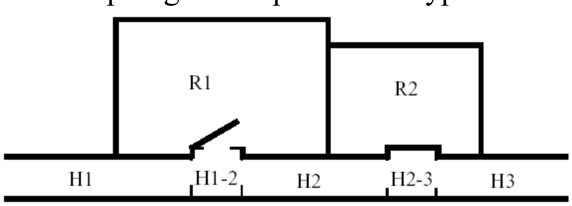
Markov Localization: Case Study 1 - Topological Map (1)

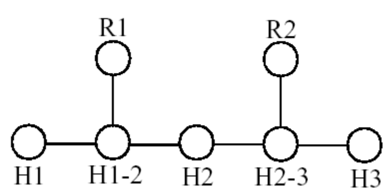
- The Dervish Robot
- Topological Localization with Sonar



Markov Localization: Case Study 1 - Topological Map (2)

- Topological map of office-type environment





	Wall	Closed door	Open door	Open hallway	Foyer
Nothing detected	0.70	0.40	0.05	0.001	0.30
Closed door detected	0.30	0.60	0	0	0.05
Open door detected	0	0	0.90	0.10	0.15
Open hallway detected	0	0	0.001	0.90	0.50

Markov Localization: Case Study 1 - Topological Map (3)

- Update of belief state for position *n* given the percept-pair *i*

  $$p(n|i) = p(i|n)p(n)$$

  - *p(n|i): new likelihood for being in position* n
  - *p(n): current belief state*
  - *p(i|n): probability of seeing* i *in* n *(see table)*

	Wall	Closed door	Open door	Open hallway	Foyer
Nothing detected	0.70	0.40	0.05	0.001	0.30
Closed door detected	0.30	0.60	0	0	0.05
Open door detected	0	0	0.90	0.10	0.15
Open hallway detected	0	0	0.001	0.90	0.50

- No action update !
  - *However, the robot is moving and therefore we can apply a combination of action and perception update*

  $$p(n_t|i_t) = \int p(n_t|n'_{t-i}, i_t)p(n'_{t-i})dn'_{t-i}$$

  - *t-i is used instead of t-1 because the topological distance between n' and n can very depending on the specific topological map*

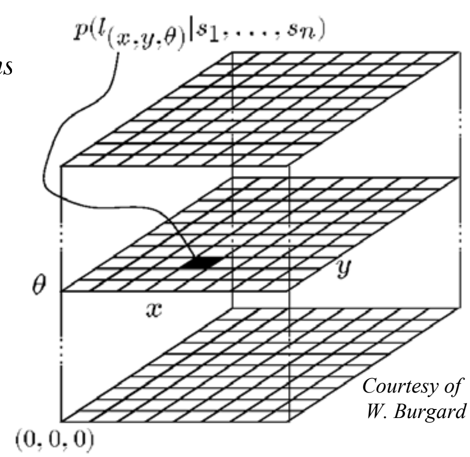
Markov Localization: Case Study 2 – Grid Map (1)

- Fine *fixed decomposition* grid (x, y, θ), 15 cm x 15 cm x 1°
  - *Action and perception update*
- Action update:
  - *Sum over previous possible positions and motion model*

  $$P(l_t|o_t) = \sum_{l'} P(l_t|l'_{t-1}, o_t) \cdot p(l'_{t-1})$$

  - *Discrete version of eq. 5.22*
- Perception update:
  - *Given perception i, what is the probability to be a location l*

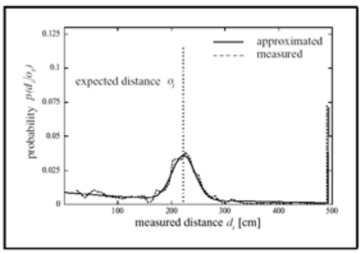
  $$p(l|i) = \frac{p(i|l)p(l)}{p(i)}$$



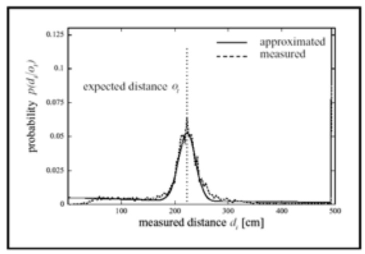
Courtesy of W. Burgard

Markov Localization: Case Study 2 – Grid Map (2)

- The critical challenge is the calculation of $p(i|l)$ $\quad p(l|i) = \frac{p(i|l)p(l)}{p(i)}$
  - *The number of possible sensor readings and geometric contexts is extremely large*
  - *$p(i|l)$ is computed using a model of the robot's sensor behavior, its position l, and the local environment metric map around l.*
  - *Assumptions*
    - *Measurement error can be described by a distribution with a mean*
    - *Non-zero chance for any measurement*



Ultrasound.

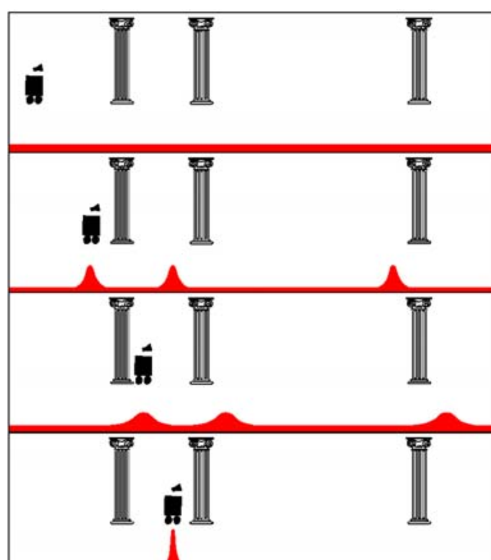


Laser range-finder.

Courtesy of W. Burgard

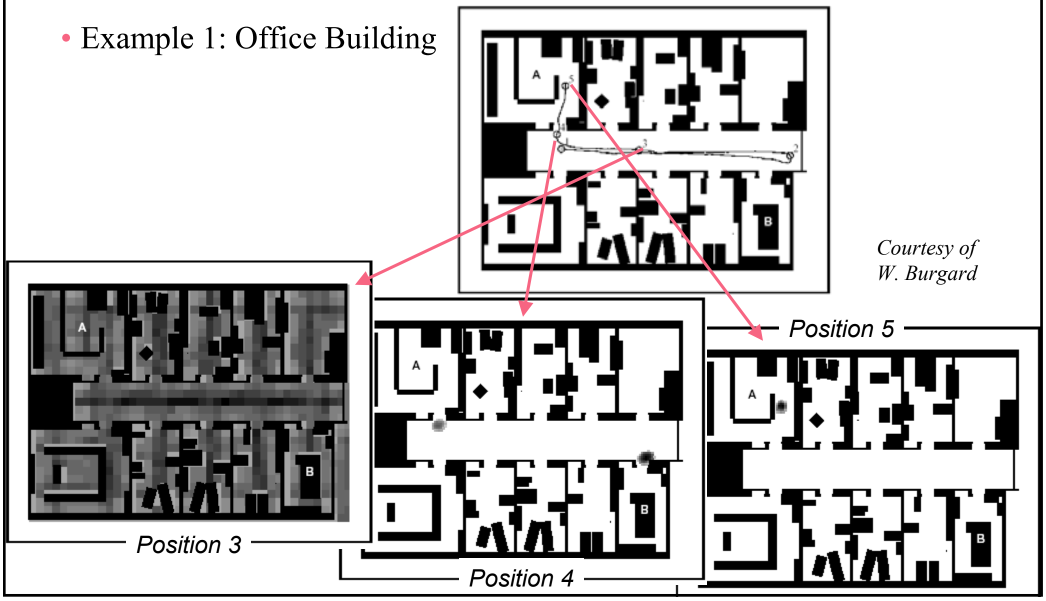
Markov Localization: Case Study 2 – Grid Map (3)

- The 1D case
  1. *Start*
     - *No knowledge at start, thus we have an 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 to estimate the new probability distribution based on the previous one and the motion.*
  4. *Robot perceives second pillar*
     - *Base on all prior knowledge the probability being at pillar 2 becomes dominant*



Markov Localization: Case Study 2 – Grid Map (4)

- Example 1: Office Building



Courtesy of W. Burgard

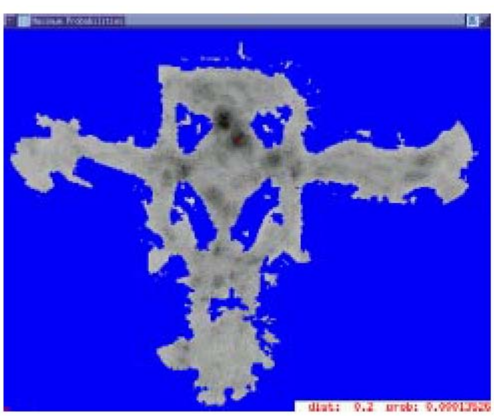
Autonomous Mobile Robots, Chapter 5

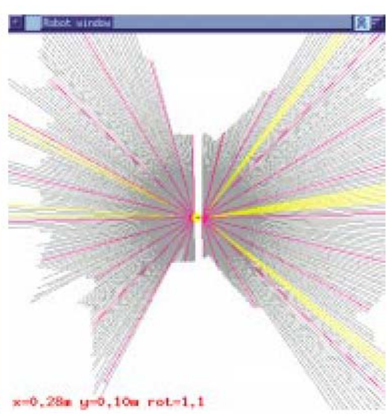
5.6.2

Markov Localization: Case Study 2 – Grid Map (5)

Courtesy of W. Burgard

- Example 2: Museum
  - *Laser scan 1*

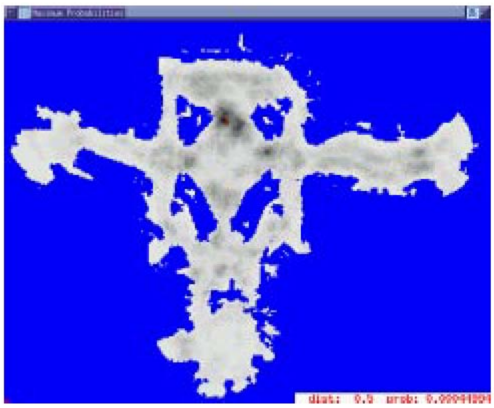
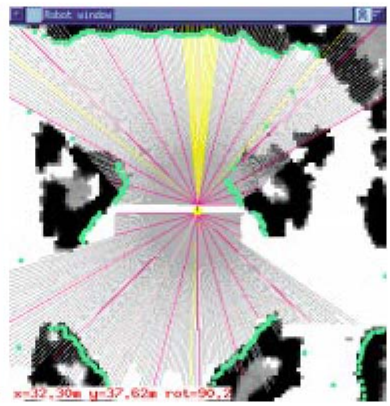




Markov Localization: Case Study 2 – Grid Map (6)

Courtesy of W. Burgard

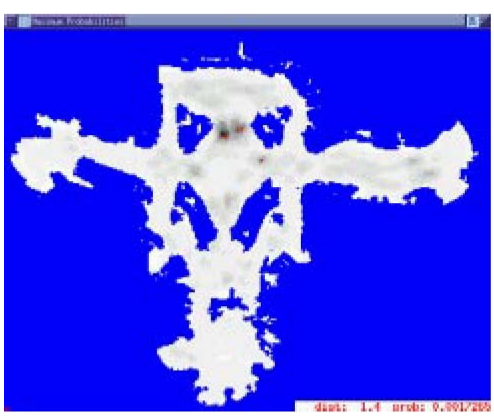
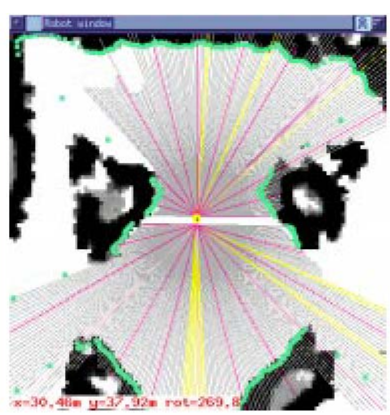
- Example 2: Museum
  - *Laser scan 2*

Markov Localization: Case Study 2 – Grid Map (7)

Courtesy of W. Burgard

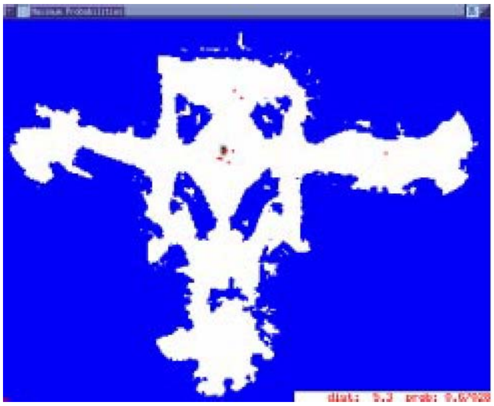
- Example 2: Museum
  - *Laser scan 3*

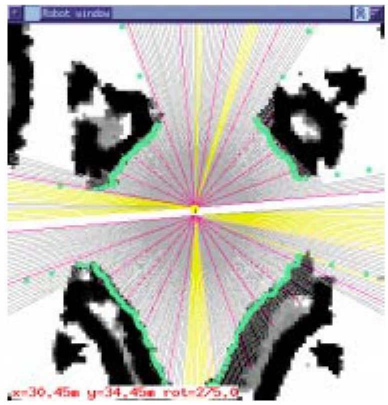



Markov Localization: Case Study 2 – Grid Map (8)

Courtesy of W. Burgard

- Example 2: Museum
  - *Laser scan 13*

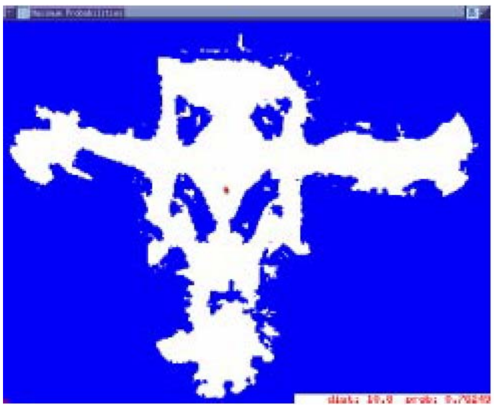


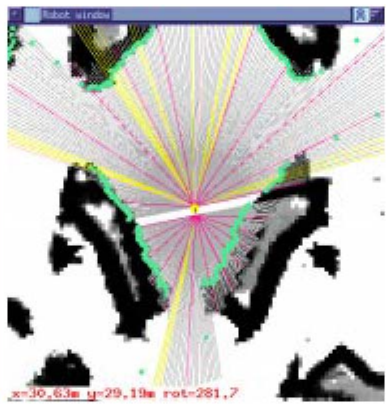


Markov Localization: Case Study 2 – Grid Map (9)

Courtesy of W. Burgard

- Example 2: Museum
  - *Laser scan 21*





Markov Localization: Case Study 2 – Grid Map (10)

- Fine *fixed decomposition* grids result in a huge state space
  - *Very important processing power needed*
  - *Large memory requirement*
- Reducing complexity
  - *Various approached have been proposed for reducing complexity*
  - *The main goal is to reduce the number of states that are updated in each step*
- Randomized Sampling / Particle Filter
  - *Approximated belief state by representing only a 'representative' subset of all states (possible locations)*
  - *E.g update only 10% of all possible locations*
  - *The sampling process is typically weighted, e.g. put more samples around the local peaks in the probability density function*
  - *However, you have to ensure some less likely locations are still tracked, otherwise the robot might get lost*

Interlude 2 …

Other Localization Methods (not probabilistic)

Localization Based on Artificial Landmarks

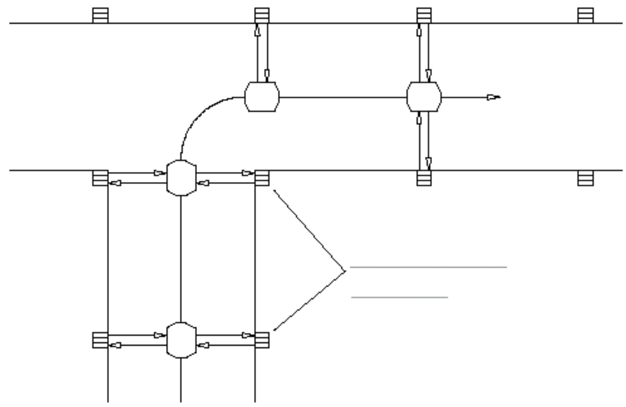


Figure 7.5: Polarized retroreflective proximity sensors are used to locate vertical strips of retroreflective tape attached to shelving support posts in the Camp Elliott warehouse installation of the MDARS security robot [Everett et al, 1994].

Other Localization Methods (not probabilistic)

Positioning Beacon Systems: Triangulation

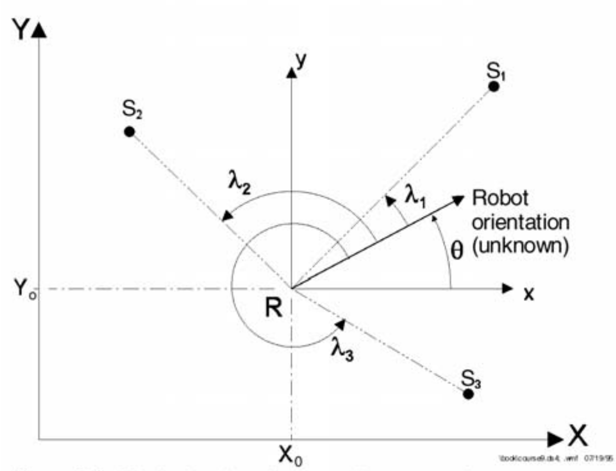
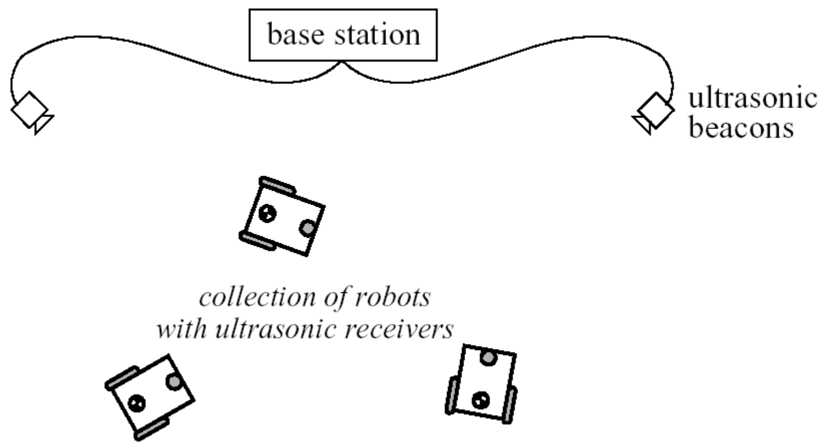


Figure 6.1: The basic triangulation problem: a rotating sensor head measures the three angles λ_1, λ_2, and λ_3 between the vehicle's longitudinal axes and the three sources S_1, S_2, and S_3.

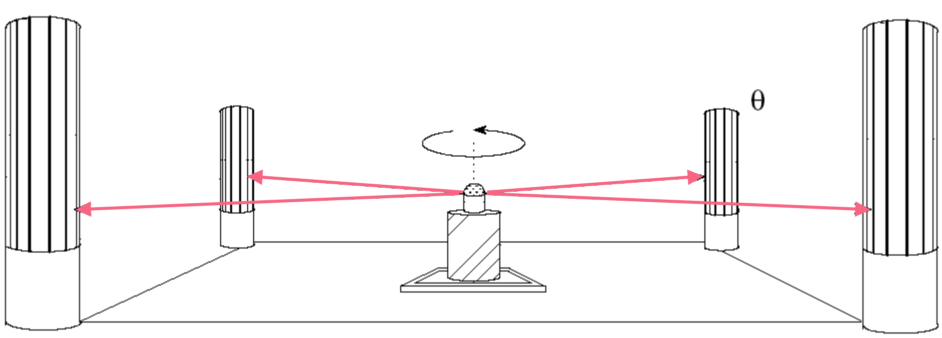
Other Localization Methods (not probabilistic)

Positioning Beacon Systems: Triangulation



Other Localization Methods (not probabilistic)

Positioning Beacon Systems: Triangulation



Other Localization Methods (not probabilistic)

Positioning Beacon Systems: Docking

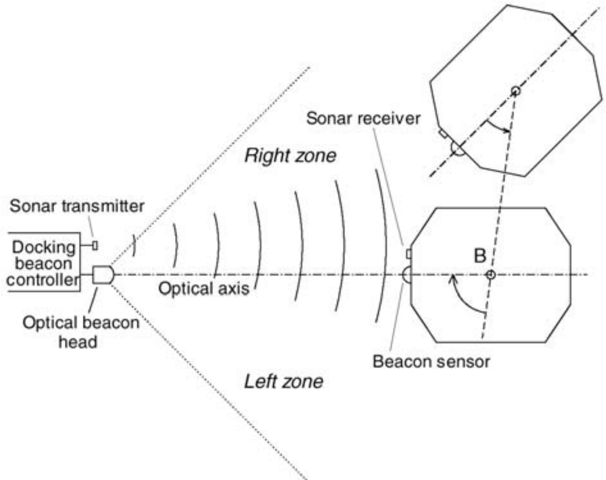


Figure 6.6: The structured-light near-infrared beacon on the Cybermotion battery recharging station defines an optimal path of approach for the *K2A Navmaster* robot [Everett, 1995].

Other Localization Methods (not probabilistic)

Positioning Beacon Systems: Bar-Code

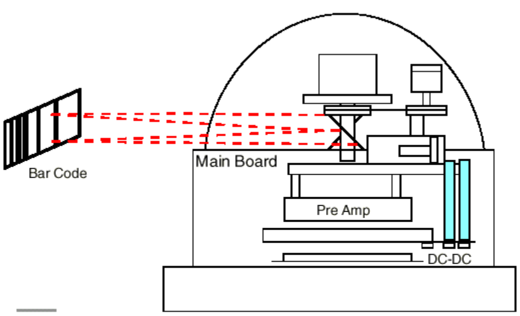


Figure 6.14: Schematics of the Denning Branch International Robotics *LaserNav* laser-based scanning beacon system. (Courtesy of Denning Branch International Robotics.)

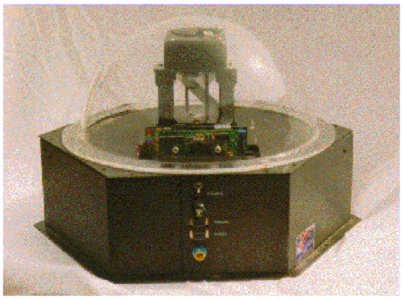


Figure 6.15: Denning Branch International Robotics (DBIR) can see *active targets* at up to 183 meters (600 ft) away. It can identify up to 32 active or passive targets. (Courtesy of Denning Branch International Robotics.)

Other Localization Methods (not probabilistic)

Positioning Beacon Systems

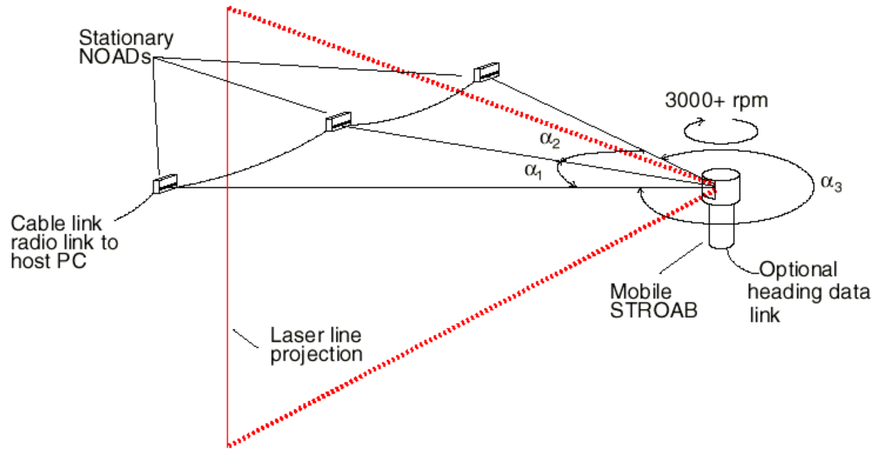


Figure 6.21: The Computerized Opto-electronic Navigation and Control (CONAC™) system employs an onboard, rapidly rotating and vertically spread laser beam, which sequentially contacts the networked detectors. (Courtesy of MTI Research, Inc.)

Autonomous Map Building

Starting from an arbitrary initial point,

a mobile robot should be able to autonomously explore the environment with its on board sensors,

gain knowledge about it,

interpret the scene,

build an appropriate map

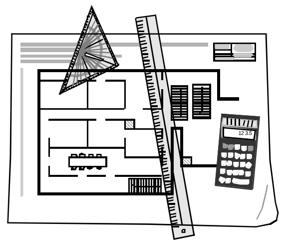
and localize itself relative to this map.

SLAM

The Simultaneous Localization and Mapping Problem

Map Building:
How to Establish a Map

1. By Hand



2. Automatically: **Map Building**

The robot **learns** its environment

Motivation:
- by hand: hard and costly
- dynamically changing environment
- different look due to different perception

3. Basic Requirements of a Map:

- *a way to incorporate newly sensed information into the existing world model*
- *information and procedures for estimating the robot's position*
- *information to do path planning and other navigation task (e.g. obstacle avoidance)*

- Measure of Quality of a map
  - *topological correctness*
  - *metrical correctness*

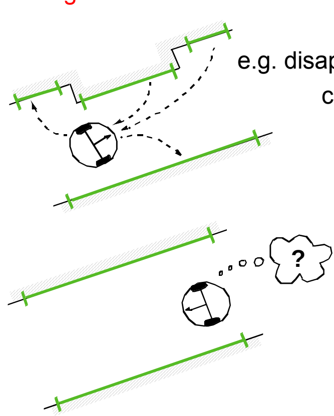
  → *predictability*

- But: Most environments are a mixture of predictable and unpredictable features
  → hybrid approach

  model-based vs. behaviour-based

Map Building:
The Problems

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

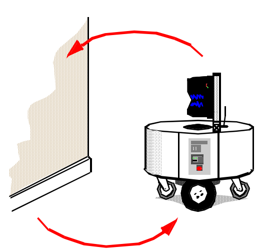


e.g. disappearing cupboard

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

2. Representation and Reduction of Uncertainty

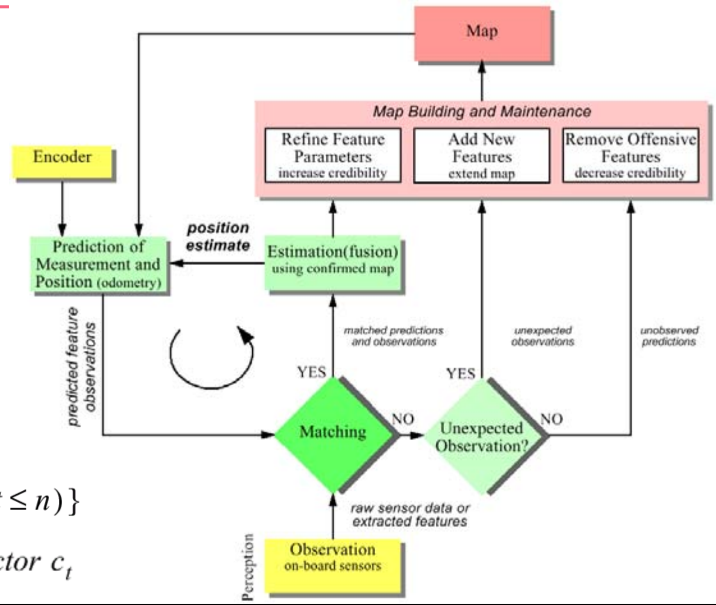
position of robot -> position of wall



position of wall -> position of robot

- probability densities for feature positions
- additional exploration strategies

General Map Building Schematics



$$M = \{\hat{z}_t, \Sigma_t, c_t | (1 \le t \le n)\}$$

credibility factor c_t

Map Representation

- M is a set n of probabilistic feature locations
- Each feature is represented by the covariance matrix Σ_t and an associated credibility factor c_t

$$M = \{\hat{z}_t, \Sigma_t, c_t | (1 \le t \le n)\}$$

- c_t is between 0 and 1 and quantifies the belief in the existence of the feature in the environment

$$c_t(k) = 1 - e^{-\left(\frac{n_s}{a} - \frac{n_u}{b}\right)}$$

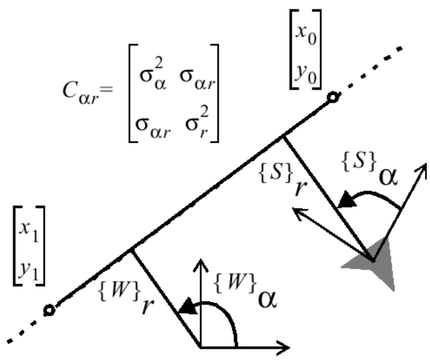
- a and b define the learning and forgetting rate and n_s and n_u are the number of matched and unobserved predictions up to time k, respectively.

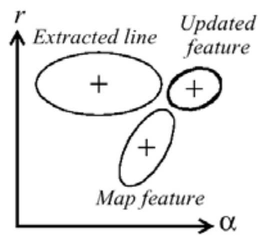
Autonomous Map Building

Stochastic Map Technique

- Stacked system state vector: $X = \left[x_r(k)\ x_1(k)\ x_2(k)\ \dots\ x_n(k)\right]^T$
- State covariance matrix:

$$\Sigma = \begin{bmatrix} C_{rr} & C_{r1} & C_{r2} & \dots & C_{rn} \\ C_{1r} & C_{11} & \dots & \dots & C_{1n} \\ C_{2r} & \dots & \dots & \dots & C_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ C_{nr} & C_{n1} & C_{n2} & \dots & C_{nn} \end{bmatrix}$$

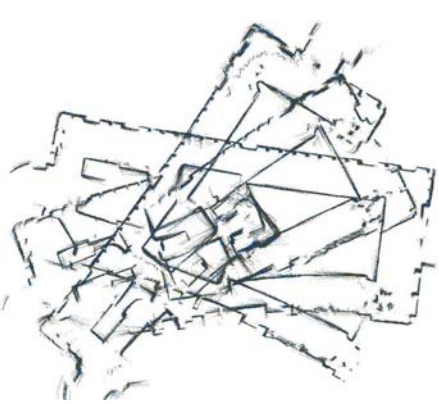


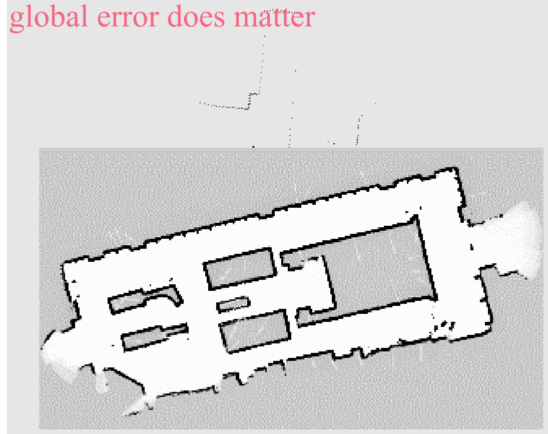


Cyclic Environments

Courtesy of Sebastian Thrun

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

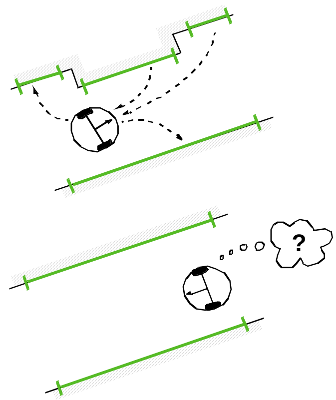




Dynamic Environments

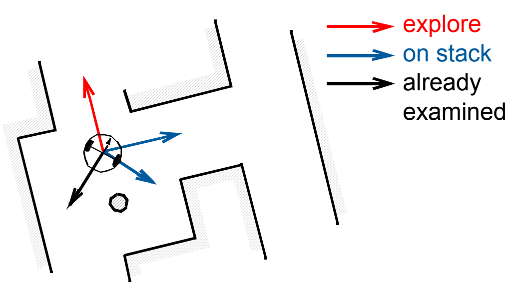
- Dynamical changes require continuous mapping

- If extraction of high-level features would be possible, the mapping in dynamic environments would become significantly more straightforward.
  - *e.g. difference between human and wall*
  - *Environment modeling is a key factor for robustness*



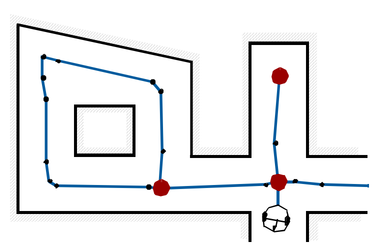
Map Building:
Exploration and Graph Construction

1. Exploration



- provides correct topology
- must recognize already visited location
- backtracking for unexplored openings

2. Graph Construction



Where to put the nodes?

- Topology-based: at distinctive locations

- Metric-based: where features disappear or get visible

Summary

- This lecture has explored issues relating to *localization*
  - *Odometry and the errors therein*
  - *Sensor noise*
  - *Map representation*
  - *Representation of robots within maps*
- We also looked at two (related) probabilistic approaches to performing localization.
  - *Markov localization*
  - *Monte-Carlo localization*
- These are both Bayes filters, and between them offer a promising solution to the localization problem.