Robotics and Autonomous Systems

Mobile Robot Localization

Li Xing
Susanne Petsch
Pedro Teixeira

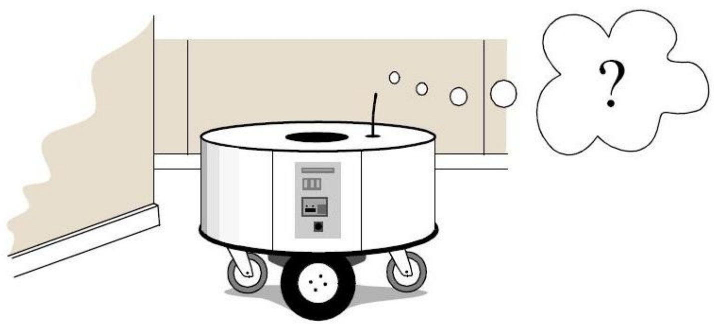
20th November 2008, KTH Stockholm

Content

- Navigation: Noise and Aliasing, Error Model
- Localisation-Based Navigation versus Programmed Solutions
- Belief Representation
- Map Representation
- Probabilistic Map-Based Localization
- Other Examples of Localization Systems
- Autonomous Map Building

What is Navigation?

- Where am I?



What is Navigation?

- Navigation is one of the most challenging competences required of a mobile robot.
- Success in navigation requires success in four building blocks of navigation:
  - Perception
    - Interpret sensors to extract meaningful data
  - Localization
    - Determine position in the environment
  - Cognition
    - Decide how to act to achieve the goals
  - Motion control
    - Modulate motor output to achieve the designed trajectory

Why is localization challenging? - sensor noise

- Examples:
- In vision-based sensor systems, illumination dependence, picture jitter, signal gain, blooming, and blurring are all possible noise.
-
- Solution:
- Take multiple readings into account, employing temporal fusion or multisensor fusion to increase the overall information content of the robot's inputs.

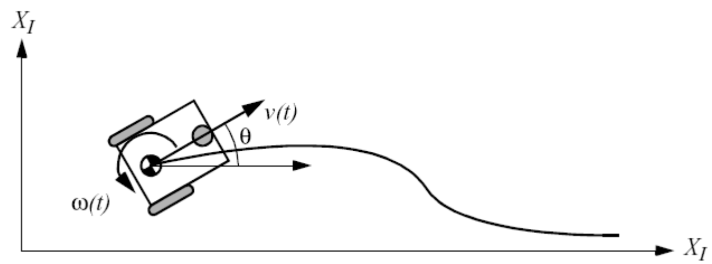
Why is localization challenging? - sensor aliasing

- Sensor aliasing is from the nonuniqueness of sensor readings
- Example:
- With sonar robots have difficulty in distinguishing between human and inanimate objects in an indoor setting.
-
- Solution:
- Techniques must be employed by the robot programmer that base the robot's localization on a series of readings and, thus, sufficient information to recover the robot's position over time.

Why is localization challenging? - effector noise

-
- Mobile robot effectors introduce uncertainty about future state. The true source of error generally lies in an incomplete model of environment.
-
- Examples:
- The robot does not model the fact that the floor may be sloped, the wheels may slip, and a human may push the robot.

An Error Model for Odometric Position Estimation



An Error Model for Odometric Position Estimation

- For a differential-drive robot the position can be estimated starting from a known position by integrating the movement.
- For a discrete system with a fixed sampling interval Δt
- The incremental travel distance $(\Delta x ; \Delta y ; \Delta \theta)$
- $\Delta x = \Delta s \cos(\theta + \Delta\theta / 2)$
- $\Delta y = \Delta s \sin(\theta + \Delta\theta / 2)$
- $\Delta\theta = \frac{\Delta s_r - \Delta s_l}{b}$
- $\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$
- Where
- $\Delta s_r ; \Delta s_l$ = traveled distance for the right and left wheel respectively
- b = distance between the two wheels of differential-drive robot

An Error Model for Odometric Position Estimation

- The updated position:

$$p' = \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = p + \begin{bmatrix} \Delta s \cos(\theta + \Delta\theta / 2) \\ \Delta s \sin(\theta + \Delta\theta / 2) \\ \Delta\theta \end{bmatrix}$$

An Error Model for Odometric Position Estimation

- Establish an error model for the integrated position p' to obtain the covariance matrix of the odometric position estimate.
- Kr and Kl are errors constants representing the nondeterministic parameters of the motor drive and the wheel-floor interaction.
- Assume the initial covariance matrix for p is known and the following covariance matrix:

$$\Sigma_\Delta = \mathrm{cov}ar(\Delta r_r, \Delta s_l) = \begin{bmatrix} k_r \mid \Delta s_r \mid & 0 \\ 0 & k_l \mid \Delta s_l \mid \end{bmatrix}$$

An Error Model for Odometric Position Estimation

- Make the following assumptions:

1.The two errors of the individually driven wheels are independent

2.The covariance of the errors (left and right wheels) are proportional to the absolute value of the travel distances

An Error Model for Odometric Position Estimation

$$\Sigma_{p'} = \nabla_p f \cdot \Sigma_p \nabla_p f^T + \nabla_{\Delta_{rl}} f \cdot \Sigma_{\Delta_{rl}} f^T$$

- The covariance matrix is given by the covariance matrix of the previous step and thus can be calculated after specifying an initial value.

- Also we can develop the two Jacobians:

$$F_p = \nabla_p f$$
$$F_{\Delta_{rl}} = \nabla_{\nabla_{rl}} f$$

To Localize or Not to Localize:

Explicit localization with reference to a map is not the only strategy that qualifies as a goal detector.

•Localization-based navigation

•Programmed solution

Behaviour-Based Approach

- Advantage: when possible, it may be implemented very quickly for a single environment with a small number of goal positions.

- Disadvantages:
- The method does not directly scale to other environments
- The underlying procedures must be carefully designed to produce the desired behavior
- A behavior-based systems may have multiple active behaviors at any one time.

Map-based approach

- Key advantages:
- The explicit, map-based concept of position makes the system's belief about position transparently available to the human operators.
- The existence of the map itself represents a medium for communication between human and robot: the human can simply give the robot a new map if the robot goes to a new environment.
- The map, if created by the robot, can be used by human as well, achieving two uses.
-
- Disadvantage and Risk:
- The map-based approach will require more up-front development effort to create a navigating mobile robot.
- An internal representation, rather than the real world itself, is being constructed and trusted by the robot.

Belief Representation

- Design questions for belief representation:

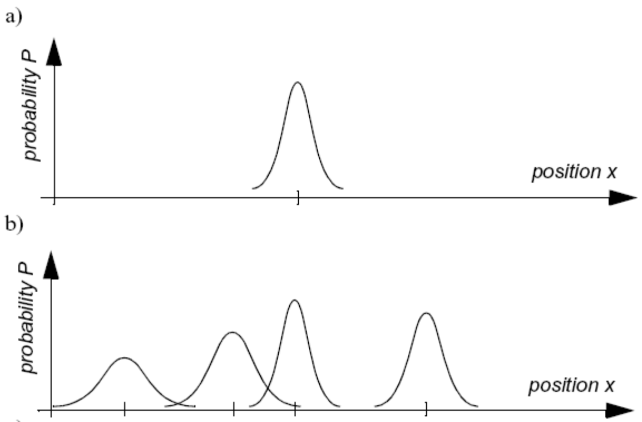
•Does the robot identify a single unique position as its current position?
•Does it describe its position in terms of a set of possible positions?
•If multiple possible positions are expressed in a single belief, how are those multiple positions ranked, if at all?

Belief Presentation

•Single-belief hypothesis
Given some environmental map, the robot's belief about position is expressed as a single unique point on the map.

•Multiple-belief hypothesis
The robot tracks not just a single possible position but a possibly infinite set of positions.

a) Continuous map with single-belief hypothesis
b) Continuous map with multiple-belief hypothesis

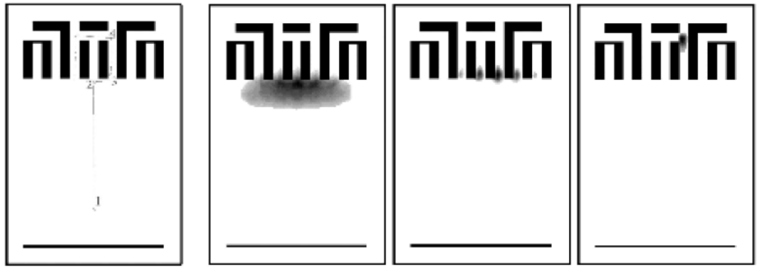


Single Belief Hypothesis

- Advantage:
  Decision-making and updating robot's belief regarding position is facilitated.

- Disadvantage:
  Robot motion often induces uncertainty due to effector and sensor noise, so forcing the position update process to always generate a single hypothesis of position is challenging, often, impossible.

Multiple-Belief Hypothesis



Path of the robot *Belief states at positions 2, 3, and 4*

Figure 5.11
Example of multiple-hypothesis tracking (courtesy of W. Burgard [49]). The belief state that is largely distributed becomes very certain after moving to position 4. Note that darker coloring represents higher probability.

Multiple-Belief Hypothesis

Advantage:

- Robot can explicitly maintain uncertainty regarding its position.
- Around the robot's ability to explicitly measure its own degree of uncertainty regarding position.

Disadvantage:

- Decision making
- Computational very expensive

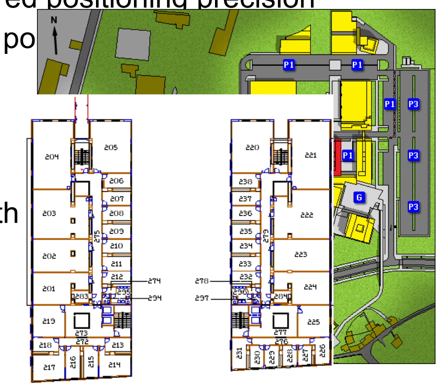
Map Representation

- *Map* - visual representation of an area
- Used for the purpose of localization
- Key aspects when choosing a particular type of map representation:
- Fidelity/Precision
- Features
- Complexity

Map Representation - Precision

- Acts as the lower bound on position representation precision
- Map must :
- be as precise as the required positioning precision
- match the precision of the po

- Street/road maps ~ meter
- Floor plans ~ cm
- Example: parking a car with



Map Representation – Features & Complexity

- The features represented in the map must match the type of sensors being used
- If a ground robot is using tilt/attitude sensors, it might be useful to use a topographical map
- The chosen type of map (cont./disc.,features, precision...) will impact computational complexity in terms of:

–Reasoning (e.g.: path planning)

–Localization

–Mapping

- It will also impose requisites on

–Processing power

–Storage space

Map Representation – Continuous Representation

•Allows exact decomposition of the environment through continuously-valued annotation.

•Advantages: high accuracy and expressiveness (fidelity to the real world)

•Disadvantages: High memory usage, computationally costly

•Solutions:

–Abstraction – capture only the relevant features.

–Closed-world assumption – represent only objects and features (instead of empty space).

–Simplification – approximate features using simple shapes (lines, polygons...)

Map Representation – Discrete Representation

•Approximate decomposition of the environment

•Fixed/Variable Cell Occupancy grids, Topological maps...

•Advantages:

–Adjacency/connectivity properties

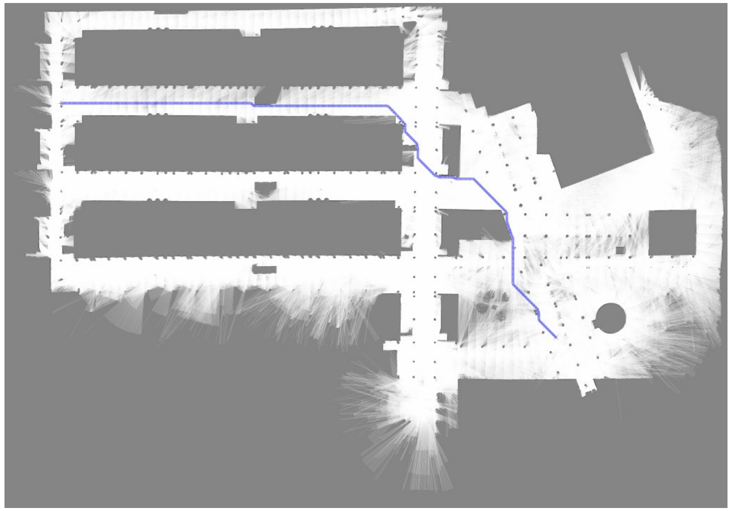
–Adequate for robots based on range sensors

•Disadvantages:

–High memory usage for small-sized cells(high precision)

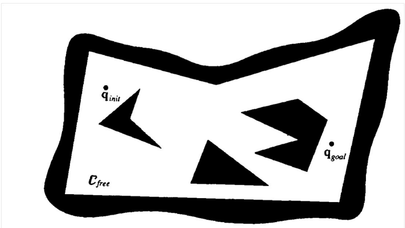
–Incompatibility with the close world assumption (empty spaces are represented -continuous representations may be smaller for sparse environments)

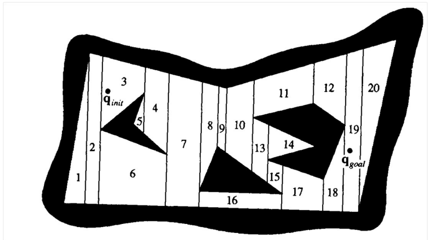
Map Representation – Discrete Representation



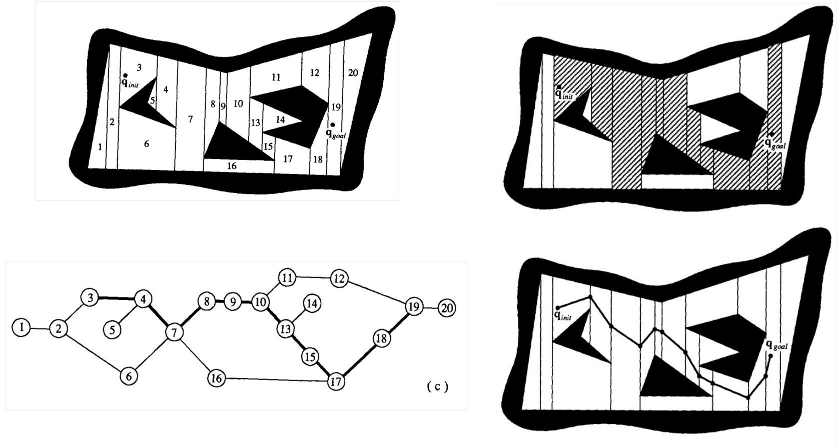
Map Representation – Decomposition Strategies

- •Exact Cell/Trapezoidal Decomposition (lossless)
- •Assumes that the position of the robot within each area (cell) doesn't matter – cells can be stored as nodes!
- •Compact representation, captures node adjacency
- •Facilitates path planning (connected graph)



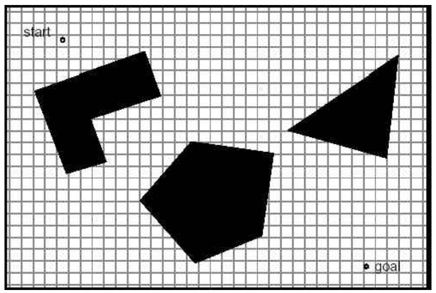


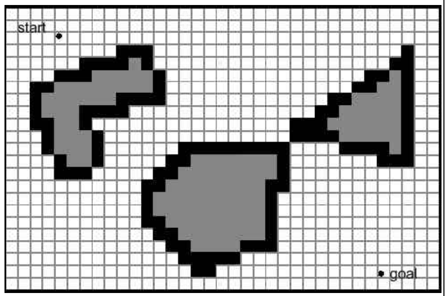
Map Representation – Decomposition Strategies



Map Representation – Decomposition Strategies

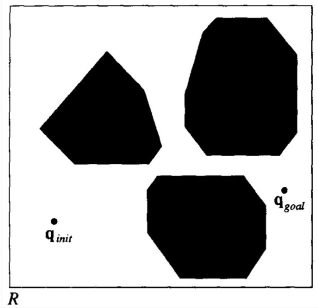
- •Fixed Decomposition (inexact, lossy)
- •Discrete grid approximation of the original map using fixed-size cells
- •Approximation may result in loss of connectivity

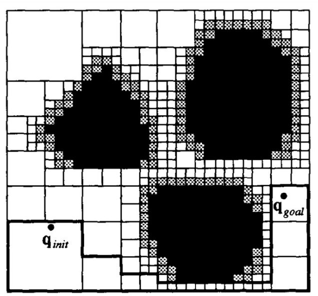




Map Representation – Decomposition Strategies

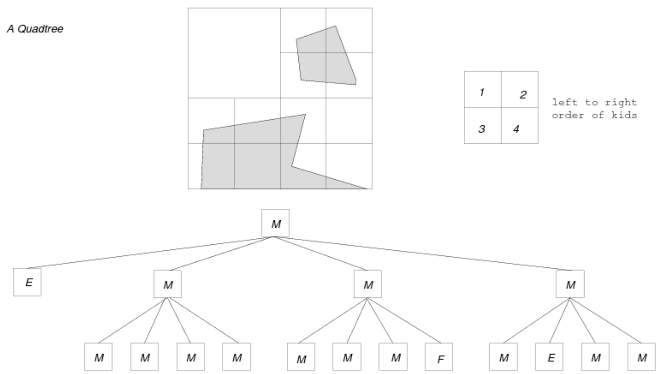
- •Adaptive/2^n-Tree/Approximate Variable Cell Decomposition (inexact, lossy)
- •Discrete grid approximation of the original map using variable-size cells





Map Representation – Decomposition Strategies

•2^n - tree $(n=2)$



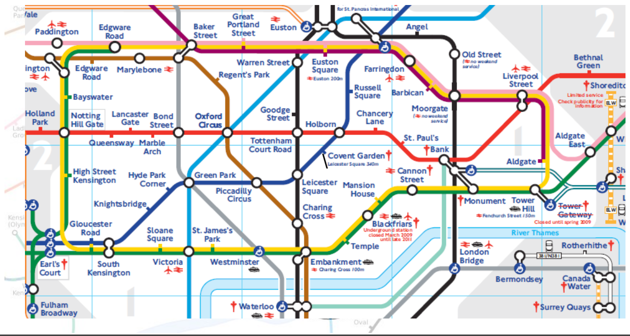
Map Representation – Decomposition Strategies

•Purely topological representations

–Graphs specifying nodes and connectivity

–Nodes capture geometric space

–Arcs represent connectivity



Map Representation – Current Challenges

•Dynamic environments –

–Transient/Temporary Obstacles (boxes, shipping packages)

–Moving Obstacles (Humans)

•Perception

–Errors

–Information Extraction

•Open spaces

–Sparseness

–Long range finding

•Sensor fusion – the only general implementation for it are neural networks.

Probabilistic Map-Based Localization

•Combination of information:

–Aim: position estimation

–Information: map and on-board sensors

•Uncertainties: error in sensor information

•Refined belief state with combined information

Combination of information

•action-update: $s'_t = Act(o_t, s_{t-1})$

–action model Act

–encoder measurement o_t

•perception-update: $s_t = See(i_t, s'_t)$

–perception model See

–exteroceptive sensor inputs i_t

Markov localization

•Arbitrary probabilistic density function

•Finite discrete number of possible positions

•Principle: prior belief state + new info = new state

•Bayes formula:

$$p(A|B) = \frac{p(B|A) \cdot p(A)}{p(B)}$$

Markov localization - perception update

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

•aim p(l|i): position l given sensor inputs i

•p(i|l): model needed, e.g. from the map

•p(l): belief state before perception update

•p(i): denominator, effectively constant, therefore dropped out, normalization at the end

Markov localization – Action update

$$p(l_t|o_t) = \int p(l_t|l'_{t-1}, o_t) p(l'_{t-1}) \, \mathrm{d}l'_{t-1}$$

•Integral over all possible ways l

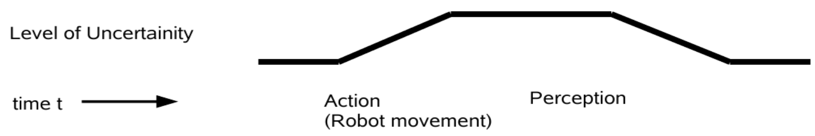
•Uncertain encoder measurement o_t

•Markov assumption:

–output = function of the previous state and its most recent actions and perceptions

–not true, but good simplification

Markov localization – principle

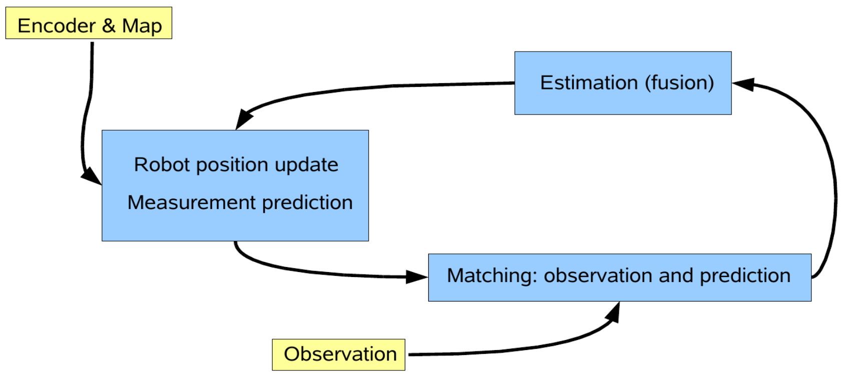


- Action update: $p(l_t|o_t) = \int p(l_t|l'_{t-1}, o_t) p(l'_{t-1})\, \mathrm{d}l'_{t-1}$
- Perception update: $$p(l|i) = \frac{p(i|l) \cdot p(l)}{p(i)}$$

Kalman filter

- Problem: sensor fusion in localization
- Knowledge: system and measuring device
- Assumption: unimodal Gaussian noise
- Aim: minimize error by least-squares technique

Kalman filter - Principle



Least-squares technique

- Sum of errors: $S = \sum_{i=1}^{n} w_i (\hat{q} - q_i)^2$
- Minimum: $\hat{q} = \dfrac{\sum_{i=1}^{n} w_i q_i}{\sum_{i=1}^{n} w_i}$
- Weight $w_i = \dfrac{1}{\sigma_i^2}$
- $\Rightarrow \hat{q} = q_1 + \dfrac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}(q_2 - q_1)$

Kalman filter – Static estimation

- Position update:
  $\hat{x}_{k+1} = \hat{x}_k + K_{k+1}(z_{k+1} - \hat{x}_k)$
- Kalman gain: $K_{k+1} = \dfrac{\sigma_k^2}{\sigma_k^2 + \sigma_z^2}$
- Variance update:
  $\sigma_{k+1}^2 = \sigma_k^2 - K_{k+1}\sigma_k^2$

Kalman filter I

1. Robot position update:
   $\hat{p}(k+1|k) = f(\hat{p}(k|k), u(k))$
   $\Sigma_p(k+1|k) = \nabla_p f \cdot \Sigma_p(k|k) \cdot \nabla_p f^T +$
   $\nabla_u f \cdot \Sigma_u(k) \cdot \nabla_u f^T$
2. Observation:
   sensor measurements $Z(k+1)$
   transformation h_i: global frame $\rightarrow$ S

Kalman filter II

3. Measurement prediction:
   $\forall n_t$ predicted features $\hat{z}_i(k+1) = h_i(z_t, \hat{p}(k+1|k))$
   set $\hat{Z} = \{\hat{z}_i(k+1) | (1 \leq i \leq n_t)\}$
4. Matching:
   innovation: observed - predicted measurements
   $\nu_{ij}(k+1) = [z_j(k+1) - h_i(z_t, \hat{p}(k+1|k))]$
   innovation covariance (incl. measurement covariance):
   $\Sigma_{IN,ij}(k+1) = \nabla h_i \Sigma_p(k+1|k) \nabla h_i^T + \Sigma_{R,j}(k+1)$
   validation gate: validity of the correspondence
   Mahalanobis distance:
   $\nu_{ij}^T(k+1) \cdot \Sigma_{IN,ij}^{-1}(k+1) \cdot \nu_{ij}(k+1) \leq g^2$

Kalman filter III

5. Estimation: applying the Kalman filter

   Kalman gain:
   $K(k+1) = \Sigma_p(k+1|k) \cdot \nabla h^T \cdot \Sigma_{IN}^{-1}(k+1)$

   Update estimation of position:
   $\hat{p}(k+1|k+1) = \hat{p}(k+1|k) + K(k+1) \cdot \nu(k+1)$

   Update error covariance matrix:
   $\Sigma_p(k+1|k+1) = \Sigma_p(k+1|k) - K(k+1) \cdot \Sigma_{IN} \cdot K^T(k+1)$

Comparison: Markov localisation and Kalman filter

	Markov localization	**Kalman filter**
starting position	everywhere	initially known position
tracked positions	multiple	single
ambiguous situation	recover possible	irrevocably lost
density functions	arbitrary	unimodal Gaussian
representation	discrete	continuous
precision	limited precision	precise, efficient
possible problems	memory, computation	no unimodal uncertainity

Other Localization Sys. - Landmark-based

•Landmark - Passive object in the workspace

•Provides highly accurate localization when in sight

•Dead-reckoning betw
to ensure short paths

•Strong formal theory
(under certain conditi

•Requires significant e

•Robotic soccer leagu

•Manufacturing

•Space transportation

Other Localization Sys. - Global Unique Localization

•Assume that localization is perfect and unique, no matter where.

•Depends heavily on the sensing system (sensors + software)

•Proposed solutions:

–Histogram-based – analysis of one or more histograms – each room has a different hist. "signature"

–Mosaic-based – analysis of floor patterns (scans the whole environment)

–Localization is a matter of matching in both cases

Other Localization Sys. - Positioning Beacon

•Uses geometric principles to achieve localization

•Active beacons

–Radio – ground & air robots - GPS, LORAN...

–Ultrasonic – AUVs – Acoustic (sonar) beacons

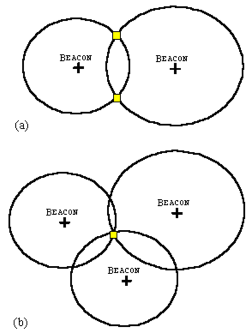
–Optical – AUVs, ground robots - IR

•Passive beacons

–Optical (retroreflective)

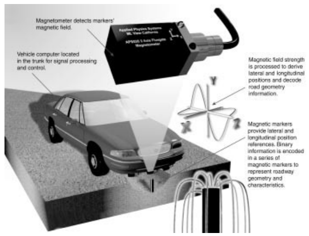
•Highly accurate localization (+)

•Environment modification (-)



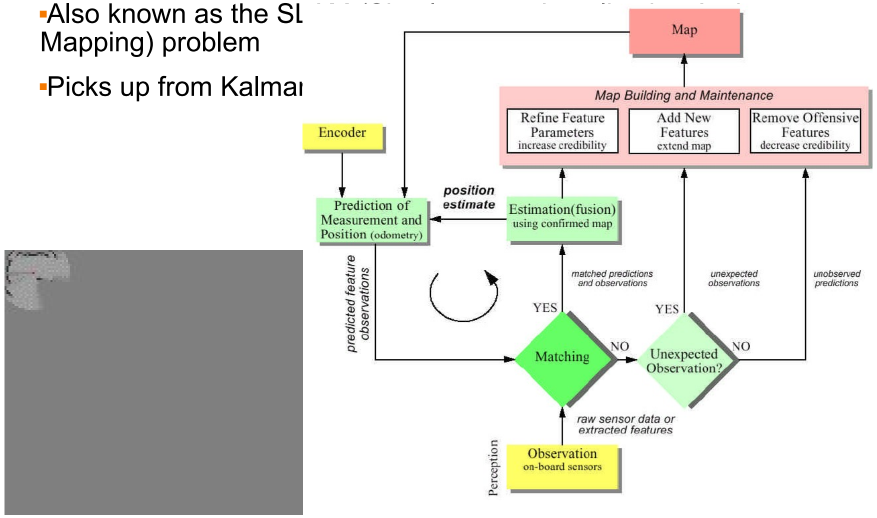
Other Localization Sys. - Route-based

- Localization relative to the path instead of a global frame
- The route is explicitly marked
  - Optical markers (e.g. UV-reflective paint)
  - Magnetic markers (e.g. Magnets, inductive coils)
- Highly accurate at the cost of environmental modification
- Highly inflexible



Autonomous Map Building

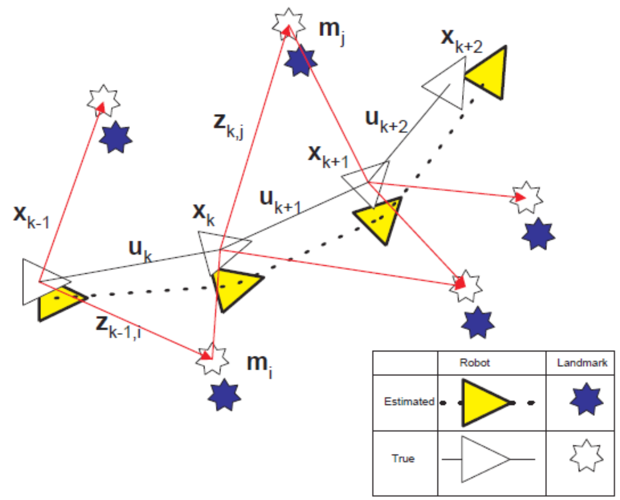
- Also known as the SL
Mapping) problem
- Picks up from Kalma



Autonomous Map Building – Stochastic Map Technique

- *From "SLAM: Part I The essential Algorithms" (Bailey, Durrant-Whyte)*
- At time instant k the following quantities are defined
  - x(k) – state vector
  - u(k) – control applied at k-1 to drive to x(k)
  - m(i) – location of the ith landmark
  - z(i,k) – observation of the ith landmark
- In addition, the following sets are defined
  - X(0:k) = {x(0),x(1)...,x(k)} – the history of vehicle locations
  - U(0:k) = {u(0),u(1)...,u(k)} – the history of control inputs
  - m = {m1,m2,...,mn} – the set of all landmarks
  - Z(0:k) = {z(0),z(1)...,z(k)} – the set of all landmark observations

Autonomous Map Building – Stochastic Map Technique



Autonomous Map Building – Stochastic Map Technique

- We want to compute, for every k, the joint posterior density of the vehicle state and landmark locations
  - P(x(k),m | Z(0:k), U(0:k), x(0))
- given:
  - the set of all landmark observations, Z(0:k)
  - the set of all control inputs, U(0:k)
  - the initial state, x(0)
- We also want a recursive solution, so we calculate this estimate from the previous estimate P(x(k-1),m | Z(0:k-1), U(0:k-1),x(0)) followed by a control input u(k), and an observation z(k)

Autonomous Map Building – Stochastic Map Technique

- This is done using Bayes' theorem, for which we need
- The **observation model**, describing the probability of making an observation z(k) when the vehicle state and landmark locations are known:
  - P(z(k)|x(k),m)
- The **motion model**, describing the probability of the state transition, from the preceding state x(k-1), which depends only on the control input applied at k-1, u(k):
  - P(x(k)|x(k-1),u(k))
- From here we can implement the SLAM algorithm in a sequential form!

Autonomous Map Building – Stochastic Map Technique

- Time update
  - $P\left(x(k), m | Z(0:k-1), U(0:k), x(0)\right) =$
  - $\int P\left(x(k), m | x(k-1), u(k)\right) \cdot P\left(x(k), m | Z(0:k-1), U(0:k-1), x(0)\right) dx(k-1)$
- Measurement update
  - $P\left(x(k), m | Z(0:k), U(0:k), x(0)\right) =$

$$\frac{P(z(k)|x(k),m) \cdot P(x(k),m|Z(0:k-1),U(0:k),x(0))}{P(z(k),m|Z(0:k-1),U(0:k))}$$

Thank you for your attention!

Questions?

References:

R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots.* Cambridge MIT Press, 2004.