Environment representation and modeling

Miroslav Kulich

Intelligent and Mobile Robotics Group
Gerstner Laboratory for Intelligent Decision Making and Control
Czech Technical University in Prague

Wednesday 27/07/2011

Why mapping?

- Learning maps is one of the fundamental problems in mobile robotics.
- Maps allow robots to efficiently carry out their tasks, allow localization ...
- Successful robot systems rely on maps for localization, path planning, activity planning etc.
- Mapping as a Chicken and Egg Problem
  - Mapping involves to simultaneously estimate the pose of the vehicle and the map. The general problem is therefore denoted as the simultaneous localization and mapping problem (SLAM).
  - Throughout this section we will describe how to calculate a map given we know the pose of the robot.

Problems in mapping

- Sensor interpretation
  - How do we extract relevant information from raw sensor data?
  - How do we represent and integrate this information over time?
- Robot locations have to be estimated
  - How can we identify that we are at a previously visited place?
  - This problem is the so-called data association problem.

Environment representation and modeling

- Environment Representation
  - Continuous Metric → x, y, ϕ
  - Discrete Metric → metric grid
  - Discrete Topological → topological grid
- Environment Modeling
  - Raw sensor data, e.g. laser range data, gray-scale images
    - large volume of data, low distinctiveness
    - 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

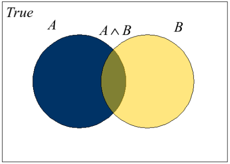
Choose the appropriate type of the map according to task you are solving!

Lecture outline

- Introduction to probability
- Spatial decomposition
  - Grid maps
  - Structures, we already know ...
  - Geometric representation
- Topological maps

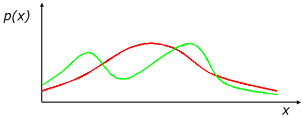
Gentle introduction to probability theory

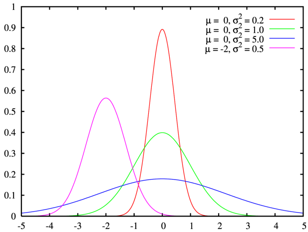
- Key idea: explicit representation of uncertainty using the calculus of probability theory
- p(X=x) probability that the random variable X has the value x
- $0 \leq p(x) \leq 1$
- $p(true) = 1, \ p(false) = 0$
- $p(A \vee B) = p(A) + p(B) - p(A \wedge B)$



Discrete and continuous random variable

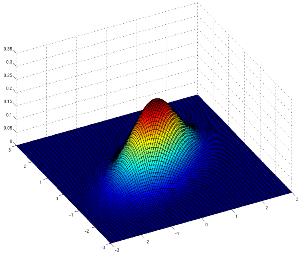
- **Discrete**: X is finite, i.e. $X = x_1, x_2, \ldots, x_n$
- **Continuous**: X takes on values in the continuum
- p is called probability mass function
- Several distributions
- Mostly known: Normal distribution (Gaussian)
- $p(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

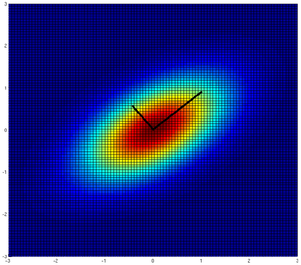




Multivariete normal distribution

$$p(x) = \frac{1}{\sqrt{2\pi|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$





- Eigenvectors and eigenvalues of covariance matrix determine elipses.

Joint and conditional probability

- $p(X = x \text{ and } Y = y) = p(x, y)$
- If X and Y are independent then

$$p(x, y) = p(x)p(y)$$

- $p(x|y)$ is the probability of x given y

$$p(x|y) = p(x, y)/p(y)$$

$$p(x, y) = p(x|y)/p(y)$$

- If X and Y are independent then

$$p(x, y) = p(x)$$

Law of Total probability, Marginals

Discrete case

$$\sum_x p(x) = 1$$

$$p(x) = \sum_y p(x,y)$$

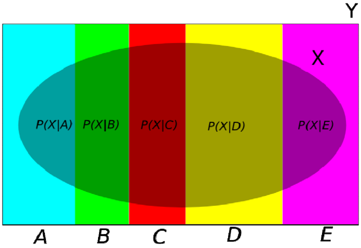
$$p(x) = \sum_y p(x|y)p(y)$$

Continuous case

$$\int_x p(x)dx = 1$$

$$p(x) = \int_y p(x,y)dy$$

$$p(x) = \int_y p(x|y)p(y)dy$$



Bayes formula

$$p(x,y) = p(x|y)p(y) = p(y|x)p(x)$$

$$\Rightarrow$$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}}$$

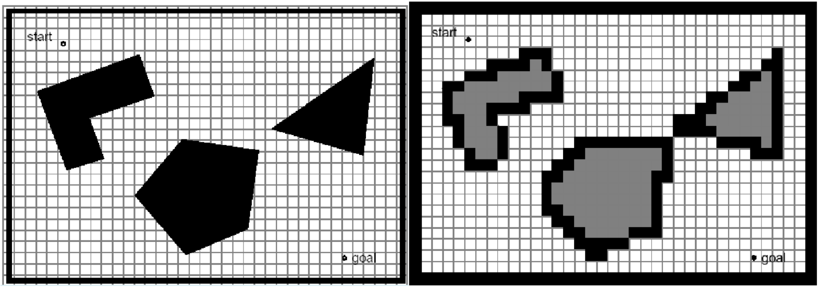
$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \eta p(y|x)p(x)$$

$$\eta = p(y)^{-1} = \frac{1}{\sum_x p(y|x)p(x)}$$

Spatial decomposition

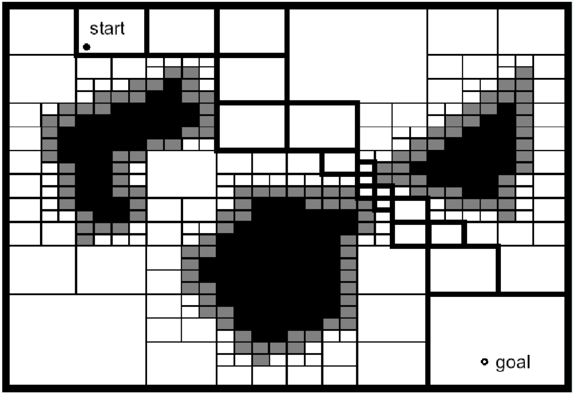
Fixed cell decomposition

- We loose details - narrow passages disapper



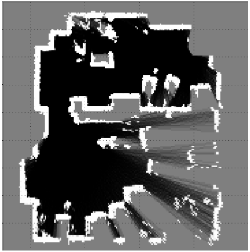
Spatial decomposition

Adaptive cell decomposition



Occupancy grid maps

- Introduced by Moravec and Elfes in 1985
- Because of intrinsic limitations in any sonar, it is important to compose a coherent world-model using information gained from multiple reading
- Represent environment by a grid.
- Estimate the probability that a location is occupied by an obstacle.



  - Key assumptions
    - Occupancy of individual cells ($m[xy]$) is independent

$$Bel(m_t) = p(m_t | u_1, z_2, \ldots, u_{t-1}, z_t) = \prod_{x,y} Bel(m_t^{[xy]})$$

    - Robot positions are known!

Updating occupancy grid maps

- Idea: Update each individual cell using a binary Bayes filter.

$$Bel(m_t^{[xy]}) = \eta p(z_t | m_t^{[xy]}) \int p(m_t^{[xy]} | m_{t-1}^{[xy]}, u_{t-1}) Bel(m_{t-1}^{[xy]}) dm_{t-1}^{[xy]}$$

- Additional assumption: Map is static.

$$Bel(m_t^{[xy]}) = \eta p(z_t | m_t^{[xy]}) \int Bel(m_{t-1}^{[xy]})$$

Occupancy grid cells

- The proposition $occ(i,j)$ means:
  - The cell C_{ij} is occupied.
- Probability: $p(occ(i,j))$ has range $[0,1]$.
- Odds: $o(occ(i,j))$ has range $[0,+\infty)$.

$$o(A) = \frac{p(A)}{p(\neg A)}$$

- Log odds: $\log o(occ(i,j))$ has range $(-\infty,+\infty)$
- Each cell C_{ij} holds the value $\log o(occ(i,j))$

Probabilistic occupancy grids

- We will apply Bayes rule:

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

  - where A is $occ(i,j)$
  - and B is an observation $r = D$
- We can simplify this by using the log odds representation.

Bayes rule using odds

- Bayes rule:

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

- Likewise:

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

- so:

$$o(A|B) = \frac{p(A|B)}{p(\neg A|B)} = \frac{p(B|A)p(A)}{p(B|\neg A)p(\neg A)} = \lambda(B|A)o(A)$$

- where:

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

and

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

Easy update using Bayes

- Bayes rule can be written:

$$o(A|B) = \lambda(B|A)o(A)$$

- Take log odds to make multiplication into addition:

$$\log o(A|B) = \log \lambda(B|A) + \log o(A)$$

- Easy update for cell content.

Occupancy grid cell update

- Cell C_{ij} holds $\log o(occ(i,j))$.
- Evidence $r = D$ means sensor r returns D.
- For each cell C_{ij} accumulate evidence from each sensor reading:

$$\log o(A|B) = \log \lambda(B|A) + \log o(A)$$

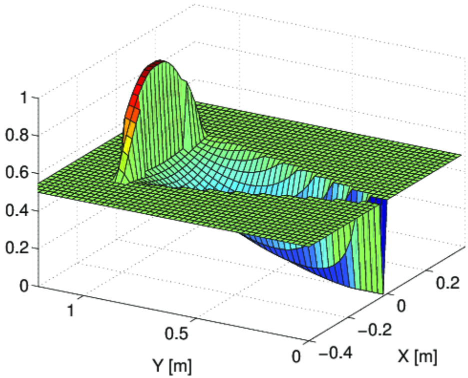
$$\log o(occ(i,j)|r = D) = \log o(occ(i,j)) + log\lambda(r = D|occ(i,j))$$

Sensor model for sonar

Probability density $p(z_t|m_t^{[xy]})$ is defined:

$$p(z_t|m_t^{[xy]}) = \frac{1 + model_O^{z_t}(\alpha, d) - model_V^{z_t}(\alpha, d)}{2},$$

where (α, d) are polar coordinates of the cell $m_t^{[xy]}$ in sensor coordinate system and z_t is measured distance.

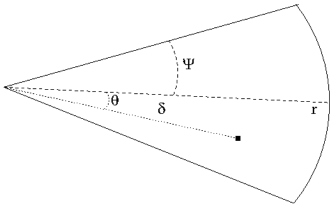


Sensor model for sonar (Elfes)

Model is defined by:

- width of the signal: Ψ
- precision of sensor measurement: ϵ

For measured distance r we get:



$$\begin{aligned} model_v^r(\delta,\phi) &= V_r(\delta)A_n(\phi) \\ model_o^r(\delta,\phi) &= O_r(\delta)A_n(\phi), \end{aligned}$$

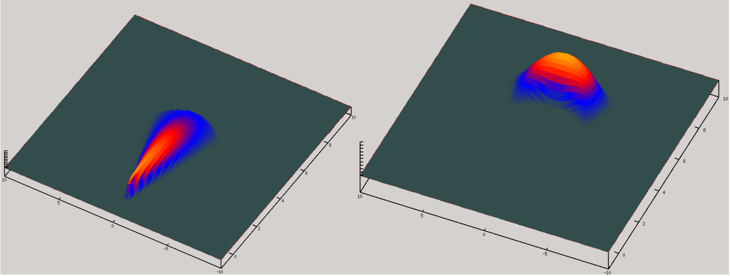
where

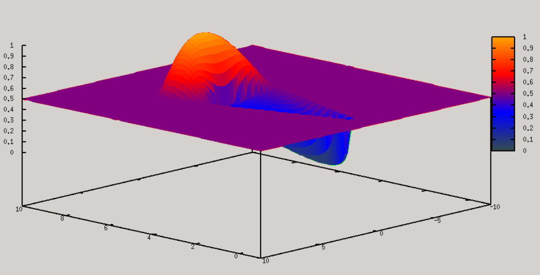
$$V_r(\delta) = \begin{cases} 1-\left(\frac{\delta}{r}\right)^2, & \textit{for} \quad \delta \in < 0, r-\epsilon > \\ 0 & \text{otherwise} \end{cases}$$

$$O_r(\delta) = \begin{cases} 1-\left(\frac{\delta-r}{\epsilon}\right)^2, & \textit{for} \quad \delta \in < r-\epsilon, r+\epsilon > \\ 0 & \text{otherwise} \end{cases}$$

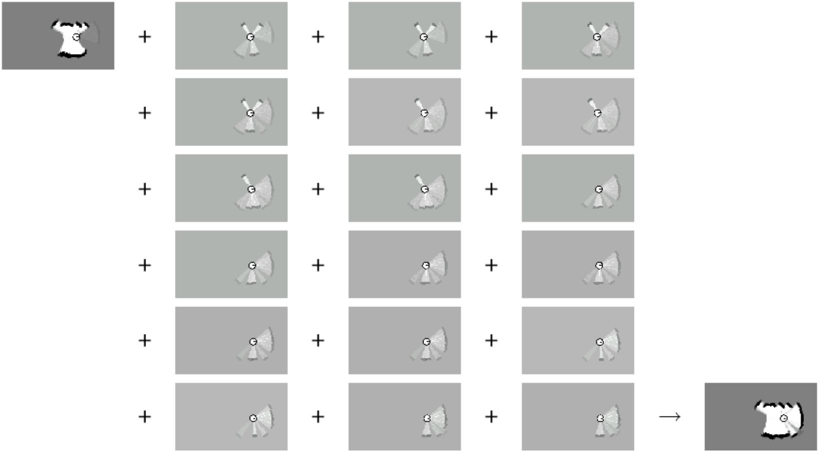
$$A_n(\phi) = \begin{cases} 1-\left(\frac{2\phi}{\Psi}\right)^2, & \textit{for} \quad \phi \in \left\langle -\frac{\Psi}{2}, \frac{\Psi}{2} \right\rangle \\ 0 & \text{otherwise} \end{cases}$$

Sensor model for sonar

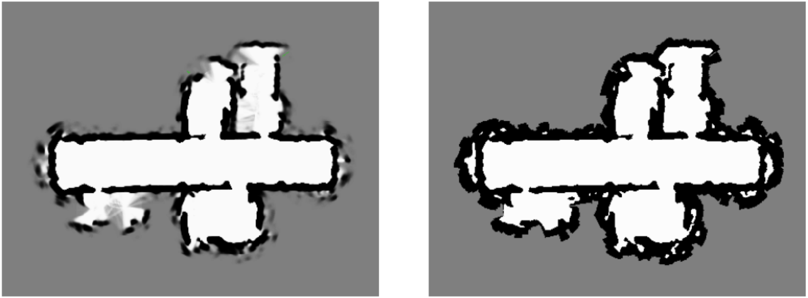




Example - incremental updating of occupancy grids



Example - map obtained with ultrasound sensors



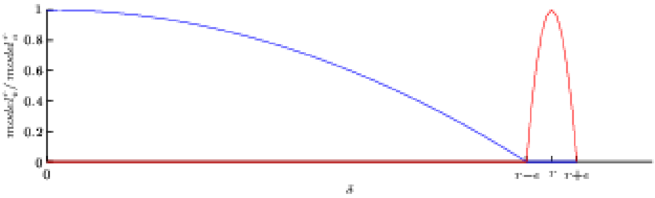
The maximum likelihood map is obtained by clipping the occupancy grid map at a threshold of 0.5

Sensor model for a laser range-finder

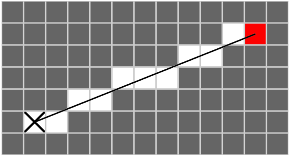
The model filters measurements longer than X:

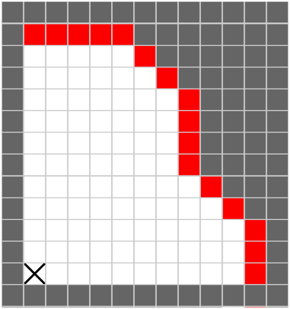
$$
model_v^r(\delta) = \begin{cases} 1 - \left(\frac{\delta}{r-\epsilon}\right)^2, & \text{for } \delta \in < 0, r-\epsilon > \\ 0 & \text{otherwise} \end{cases}
$$

$$
model_o^r(\delta) = \begin{cases} 1 - \left(\frac{\delta - r}{\epsilon}\right)^2, & \text{for } r < X \wedge \delta \in < r-\epsilon, r+\epsilon > \\ 0 & \text{otherwise} \end{cases}
$$



Laser model - a practical approach





- Connect a cell corresponding the sensor position with the hit cell.
- Set all cells on the line as empty.
- Set the hit cell as occupied.
- Apply Bayes rule to update the grid.
- Use some line drawing algorithm (Bresenham).
- Improvement: use flood-fill algorithm to draw the whole scan.

Alternative: Simple counting

Reflection maps

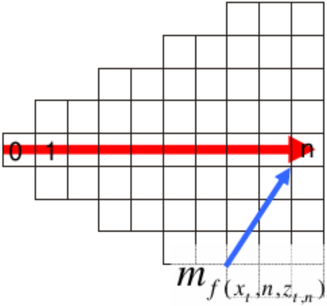
- For every cell count
  - *hits*(x, y): number of cases where a beam ended at $\langle x, y\rangle$
  - *misses*(x, y): number of cases where a beam passed through $\langle x, y\rangle$

$$Bel(m^{[xy]}) = \frac{hits(x,y)}{hits(x,y) + misses(x,y)}$$

- Value of interest: $p((reflects(x, y)))$

The measurement model

pose at time t:	x_t
beam n of scan t:	$z_{t,n}$
maximum range reading:	$\zeta_{t,n} = 1$
beam reflected by an object:	$\zeta_{t,n} = 0$



$$p(z_{t,n}|x_t, m) = \begin{cases} \prod_{k=0}^{z_{t,n}-1}(1 - m_{f(x_t,n,k)}) & \text{if } \zeta_{t,n} = 1 \\ m_{f(x_t,n,z_{t,n})} \prod_{k=0}^{z_{t,n}-1}(1 - m_{f(x_t,n,k)}) & \text{if } \zeta_{t,n} = 0 \end{cases}$$

Computing the most likely mapping

- Compute values for m that maximize

$$m^* = \arg\max_m p(m|z_1, z_2, \ldots, z_t, x_1, x_2, \ldots, x_t)$$

- Assuming an uniform prior probability for $p(m)$, this is equivalent to maximizing (apply Bayes rule):

$$\begin{aligned} m^* &= \arg\max_m p(z_1, z_2, \ldots, z_t|m, x_1, x_2, \ldots, x_t) \\ &= \arg\max_m \prod_{t=1}^{T} p(z_t|m, x_t) \\ &= \arg\max_m \sum_{t=1}^{T} \ln p(z_t|m, x_t) \end{aligned}$$

Computing the most likely mapping

$$m^* = \arg\max_m \left[\sum_{j=1}^{J} \sum_{t=1}^{T} \sum_{n=1}^{N} (I\,(f(x_t, n, z_{t,n}) = j)(1 - \zeta_{t,n}) \ln m_j \right.$$
$$\left. + \sum_{k=0}^{z_{t,n}-1} I\,(f(x_t, n, k) = j) \ln(1 - m_j)) \right]$$

Suppose the number of times a beam

- that is not a maximum range beam ended in cell j ($hits(j)$).

$$\alpha_j = \sum_{t=1}^{T} \sum_{n=1}^{N} (I\,(f(x_t, n, z_{t,n}) = j)\,(1 - \zeta_{t,n})$$

- intercepted cell j without ending in it ($misses(j)$).

$$\beta_j = \sum_{t=1}^{T} \sum_{n=1}^{N} \left[\sum_{k=0}^{z_{t,n}-1} I\,(f(x_t, n, k) = j) \right]$$

Computing the most likely mapping

We assume that all cells m_j are independent:

$$m^* = \arg\max_m \left(\sum_{j=1}^{J} \alpha_j \ln m_j + \beta_j \ln (1 - m_j) \right)$$

If we set

$$\frac{\partial m}{\partial m_j} = \frac{\alpha_j}{m_j} - \frac{\beta_j}{1 - m_j}$$

we obtain

$$m_j = \frac{\alpha_j}{\alpha_j + beta_j}$$

$\Downarrow$

Computing the most likely map amounts to counting how often a cell has reflected a measurement and how often it was intercepted.

Difference between Occupancy Grid Maps and Counting

- The counting model determines how often a cell reflects a beam.
- The occupancy model represents whether or not a cell is occupied by an object.
- Although a cell might be occupied by an object, the reflection probability of this object might be very small.

Comparison

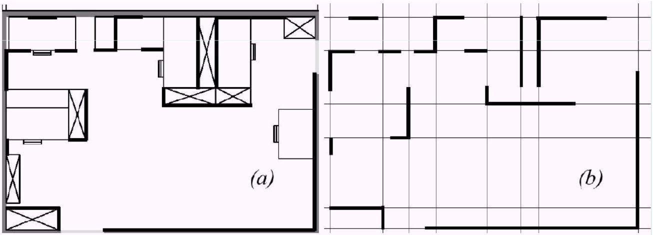
Occupancy map × Reflection map

Grid maps - summary

- Occupancy grid maps are a popular approach to represent the environment of a mobile robot given known poses.
- In this approach each cell is considered independently from all others.
- It stores the posterior probability that the corresponding area in the environment is occupied.
- Occupancy grid maps can be learned efficiently using a probabilistic approach.
- Reflection maps are an alternative representation.
- They store in each cell the probability that a beam is reflected by this cell.
- We provided a sensor model for computing the likelihood of measurements and showed that the counting procedure underlying reflection maps yield the optimal map.

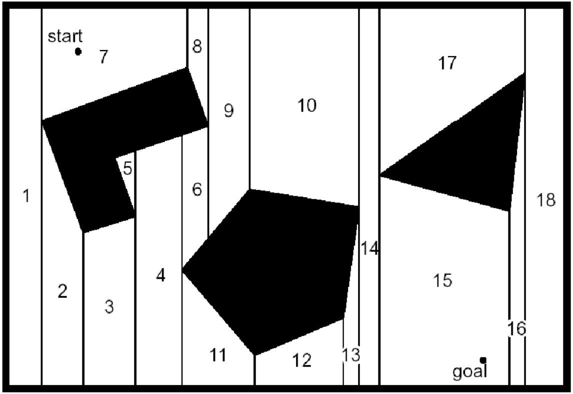
Geometric representation

- Environment modeling by geometric primitives.
- The environment can be approximated:
  - line segments - most frequent, high precision → large number of segments.
  - second order curves - better approximation, computationally expensive, how to plan?
- Pros: maps available, easy planning.
- Cons: difficult to build from sensor data.



Exact cell decomposition

- Trapezoidal
- Cylindrical
- Triangulation



How to create a geometric map

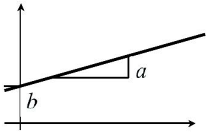
line based

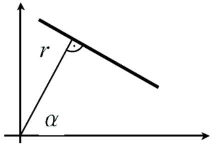
- Directly from raw sensor data
  - Detection of line segments.
  - Correspondence finding.
  - Adding new segments
- From a grid map
  - Building a grid map.
  - Detecting line segments in the grid map.

Line segment description

Many possibilities

End points	(A, B)
Slope–intercept form	$y = ax + b$
Normal form	$x \cos(\alpha) + y \sin(\alpha) = r$
Covariance matrix	





Covariance matrix

- Suppose that points $\{P_i\}_{i=1}^n$, where $P_i = (x_i, y_i)$ form a line u.
- Covariance matrix is defined:

$$C = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix},$$

where σ_x^2 a σ_y^2 are variances in x and y coordinates and σ_{xy} is their covariance:

$$\sigma_{xy} = \frac{\sum_{i=1}^n (x_i - m_x)(y_i - m_y)}{n} = \frac{\sum_{i=1}^n x_i y_i}{n} - m_x m_y,$$

where $m_x = \frac{\sum_{i=1}^n x_i}{n}$ a $m_y = \frac{\sum_{i=1}^n y_i}{n}$.

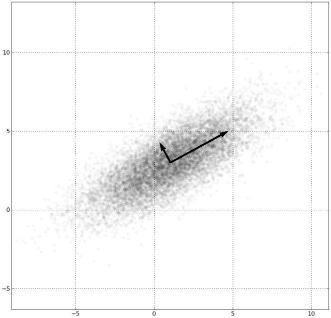
Covariance matrix as an ellipse

- We can express covariance (line segment) as an ellipse.
- The directions of semi-axes correspond to the eigenvectors of this covariance matrix and
- their lengths to the square roots of the eigenvalues.

Eigenvalues can be determined as:

$$\lambda_1 = \frac{\sigma_x^2 + \sigma_y^2 + \sqrt{(\sigma_x^2 - \sigma_y^2)^2 + 4\sigma_{xy}^2}}{2}$$

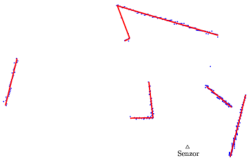
$$\lambda_2 = \frac{\sigma_x^2 + \sigma_y^2 - \sqrt{(\sigma_x^2 - \sigma_y^2)^2 + 4\sigma_{xy}^2}}{2}$$

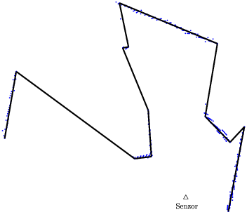


Ratio of the eigenvalues $\Lambda = \frac{\lambda_1}{\lambda_2}$ describe quality of the segment.

Detection of line segments

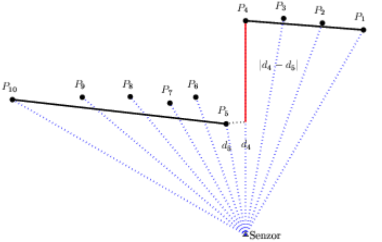
- Problem: Find line segments approximating a given set of points (scan).
- Approaches:
  - sequence - points treated one by one.
  - iterative - processes whole scan
- Our approach:
  - Use sequence algorithm to split the input set into „continuous" sub-sets.
  - Use iterative algorithm to find line-segments for each sub-set.
  - Use covariance matrix to describe the line-segments.





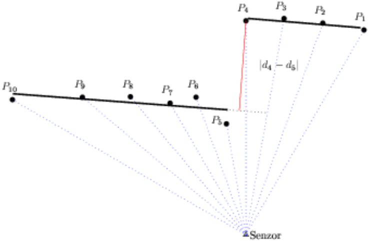
Sequence algorithms

Successive Edge Following (SEF)



- Processes a raw scan (measured distances).
- if $|r_i - r_{i-1}| > \textit{Threashold}$ then start new segment.

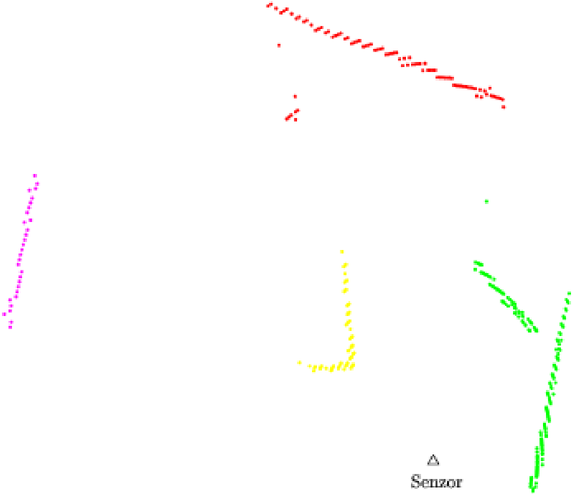
Line Tracking (LT)



- Processes data points.
- Actual segment is approximated by line (least squares).
- if $d(l_k, p_i) > \textit{Threashold}$ then start new segment.

Successive Edge Following

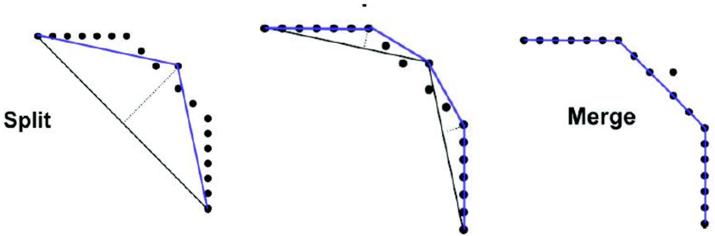
Example



Iterative algorithm

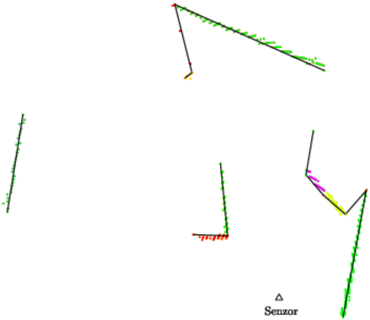
Iterative End Point Fit

1. Connect the first and last points with a line.
2. Detect a point with a maximum distance to the line
3. If the distance $d(l_k, p_m) > \mathit{Threashold}$ then split the point into two groups.
4. Perform steps 1-3 for each of the groups.
5. Join pairs of adjoining segments if the resulting segment is ,,good".

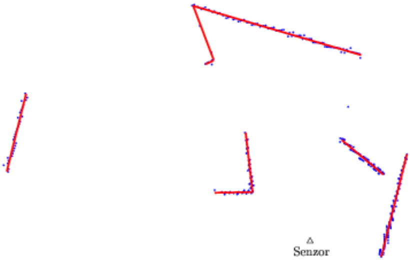


Iterative End Point Fit

Example



After IEPF



Final result

Correspondence finding

- Problem1: are two segments the same?
- Problem2: how to merge them?

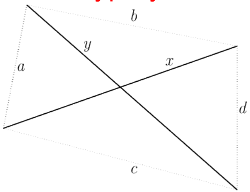
Crowley

$\left(\phi_i, \sigma^2_{\phi_i}, \rho_i, \sigma^2_{\rho_i}, x_i, y_i, h_i\right)$, where ϕ_i - slope, ρ_i - distance to origin, variances ϕ_i and ρ_i $\sigma^2_{\phi_i}$ and $\sigma^2_{\rho_i}$, (x_i, y_i) center h_i half length.

Two segments are the same if:

$$\mid \phi_1 - \phi_2 \mid \leq \sigma^2_{\phi_1} + \sigma^2_{\phi_2}$$
$$\mid \rho_1 - \rho_2 \mid \leq \sigma^2_{\rho_1} + \sigma^2_{\rho_2}$$
$$(x_1 - x_2)^2 + (y_1 - y_2)^2 \leq h_1 + h_2$$

Skrzypczynski



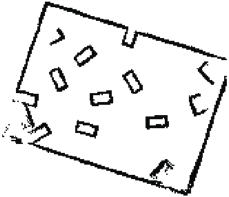
Two segments are the same if:

$$a + b < x + Tol$$
$$c + d < x + Tol$$
$$a + c < y + Tol$$
$$b + d < y + Tol$$

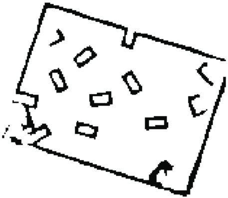
Map building from a grid map

- Based on occupancy grid proccessing using mathematical morphology.

Input grid

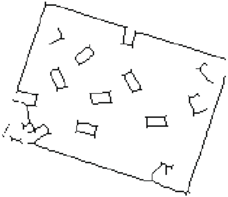


Segmentation

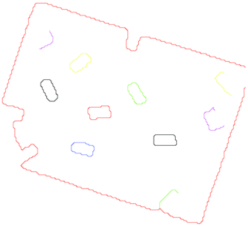


Dilation & erosion

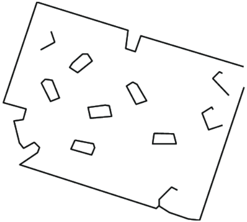
Map building from a grid map



Skeleton



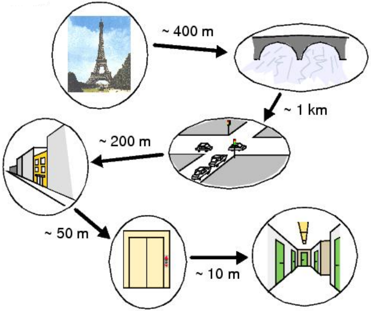
Skeleten splitting

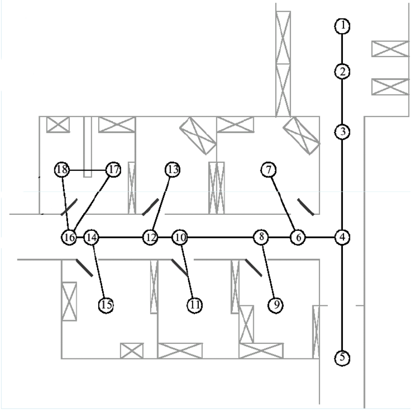


Final approximation

Topological map

- defined as a graph - nodes and connections





Building topological map from occupancy grid

S. Thrun, A. Bücken