

CSC477

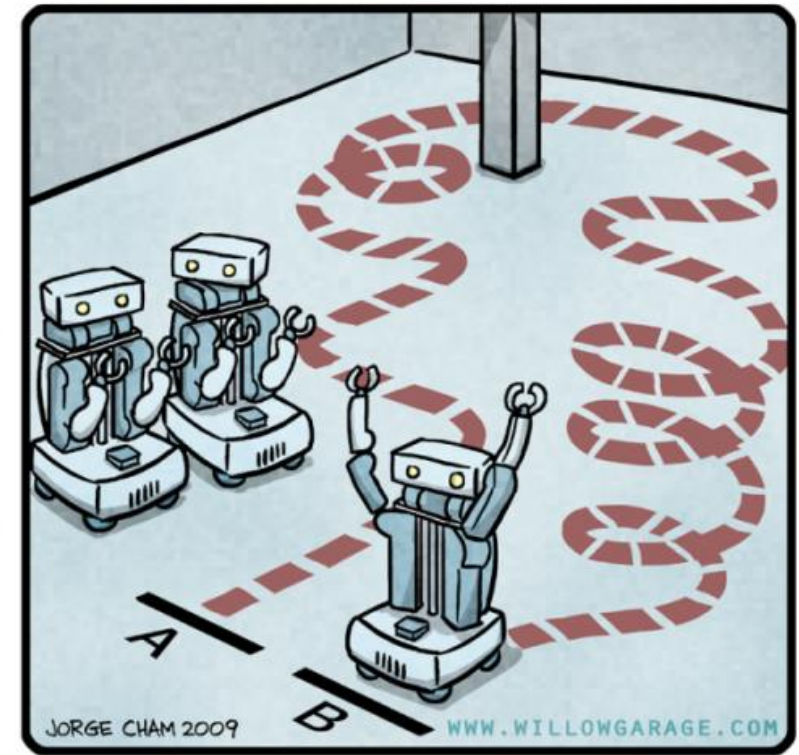
Introduction to Mobile Robotics

Florian Shkurti

Week #6: Mapping

Today's agenda

- How to represent maps
- Probabilistic occupancy grid mapping



"HIS PATH-PLANNING MAY BE
SUB-OPTIMAL, BUT IT'S GOT FLAIR."

Categories of maps

- Metric
 - Map accurately represents lengths and angles
- Topological
 - Map is reduced to a graph representation of the structure of free space
- Topometric
 - Atlas: a combination of local metric maps (nodes) connected via edges
- Sequence of raw time-series observations (e.g. video)
 - No metric or topological information directly represented by the map

Typical operations on maps

- Distance and direction to closest obstacle
- Collision detection: is a given robot configuration in free space?
- Map merging / alignment
- Occupancy updates
- Raytracing

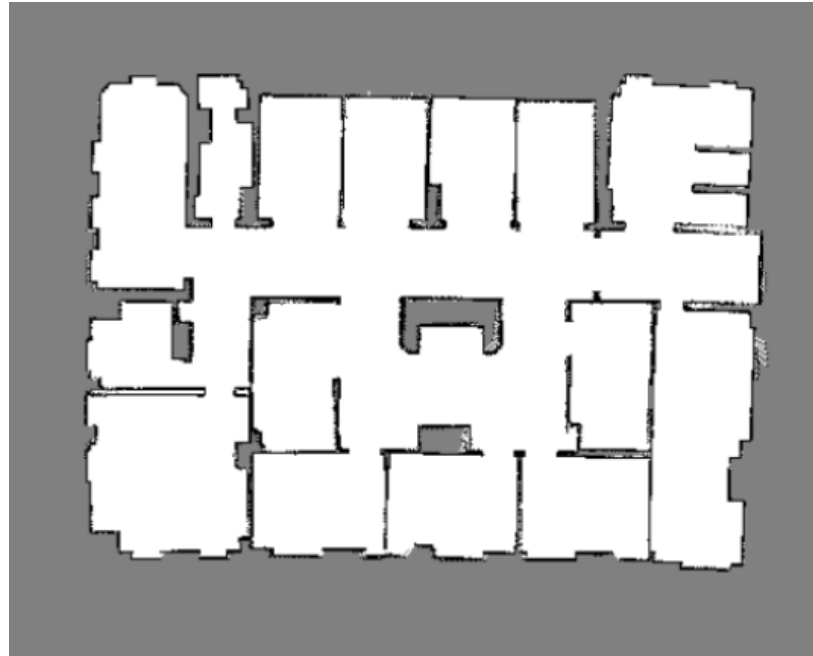
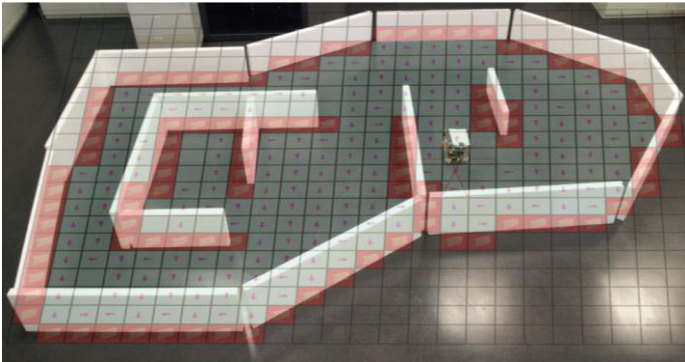
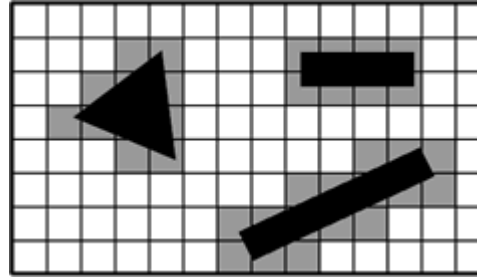
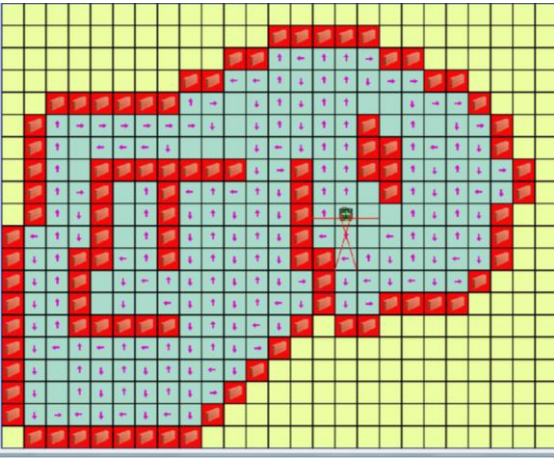
Typical operations on maps

- Distance and direction to closest obstacle
- Collision detection: is a given robot configuration in free space?
- Map merging / alignment
- Occupancy updates
- Raytracing

Common operations in
computer graphics

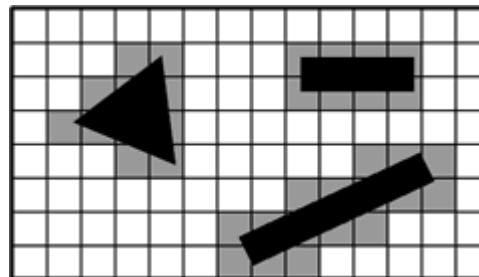
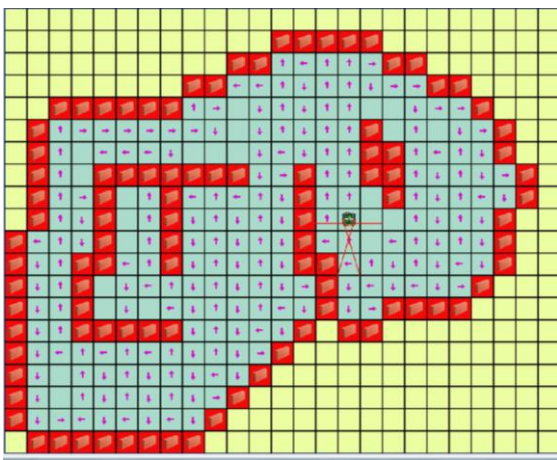
Metric Maps

Occupancy Grids



- Each cell contains either:
- unknown/unexplored (grey)
 - probability of occupation

Occupancy Grids

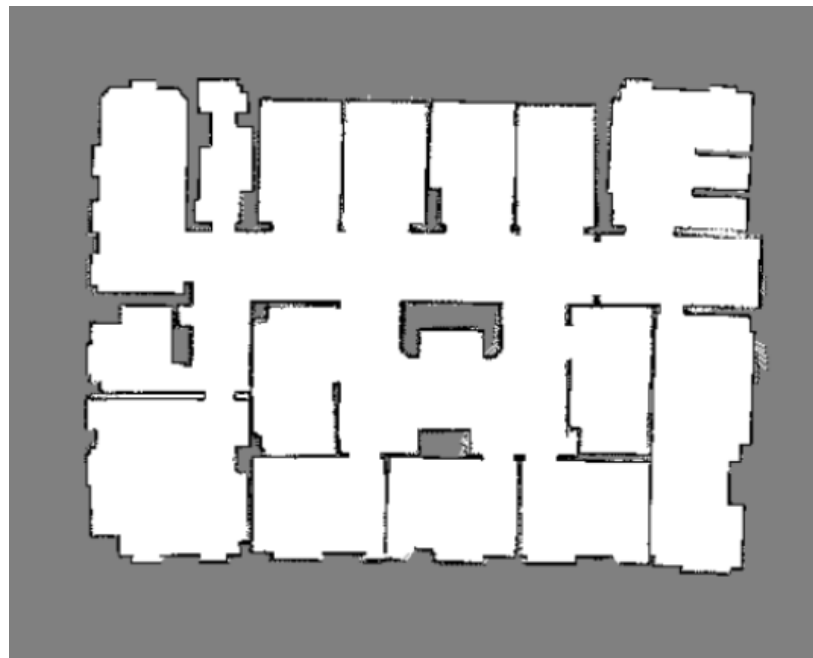
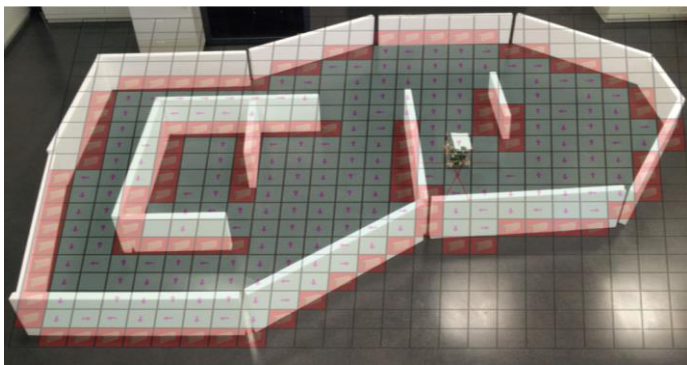


Advantages:

- $O(1)$ occupancy lookup and update
- Supports image operations

Disadvantages:

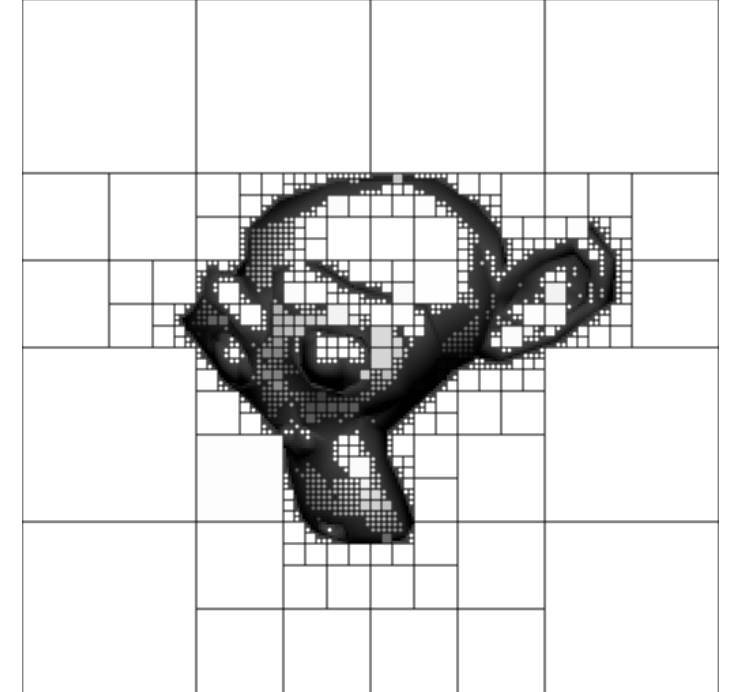
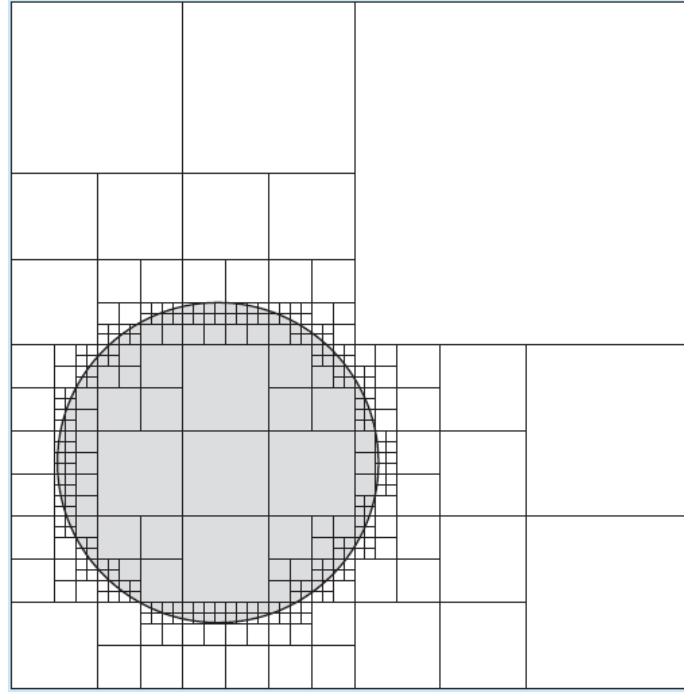
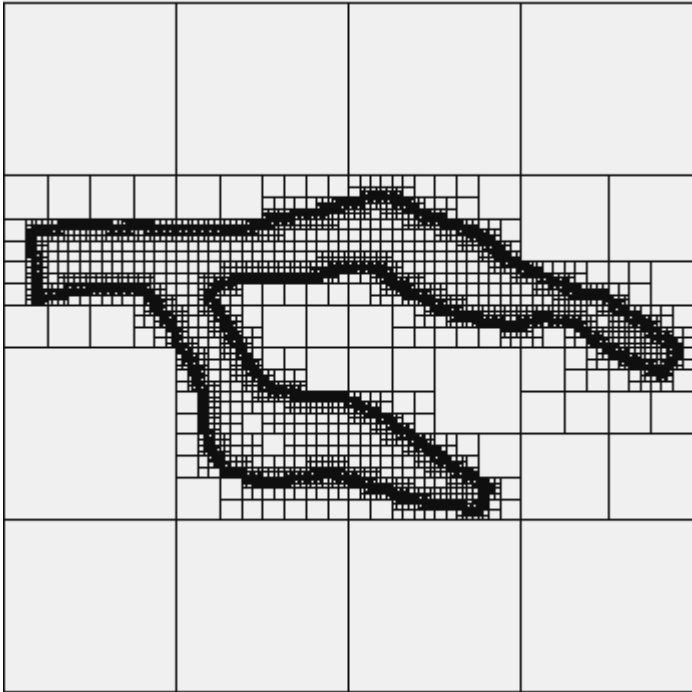
- Doesn't scale well in higher dimensions



Each cell contains either:

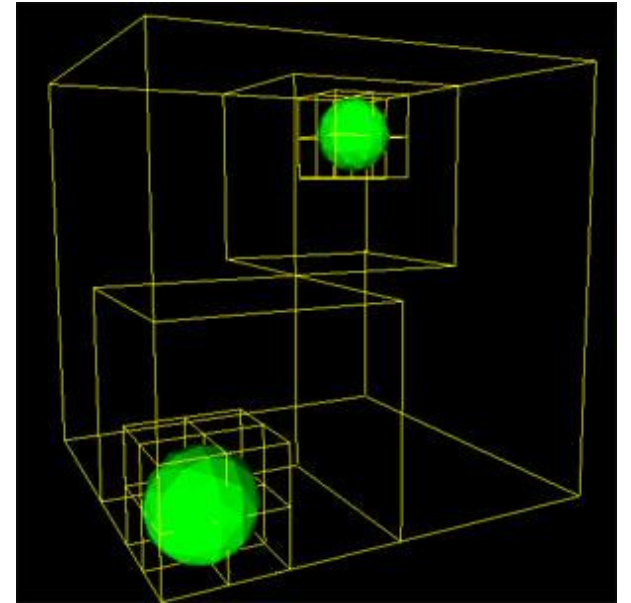
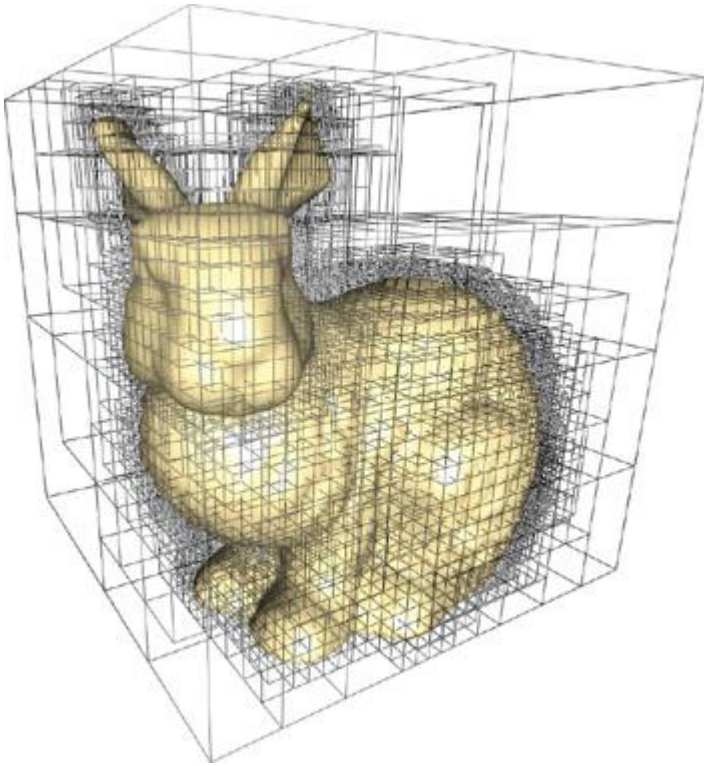
- unknown/unexplored (grey)
- probability of occupation

Quadtrees



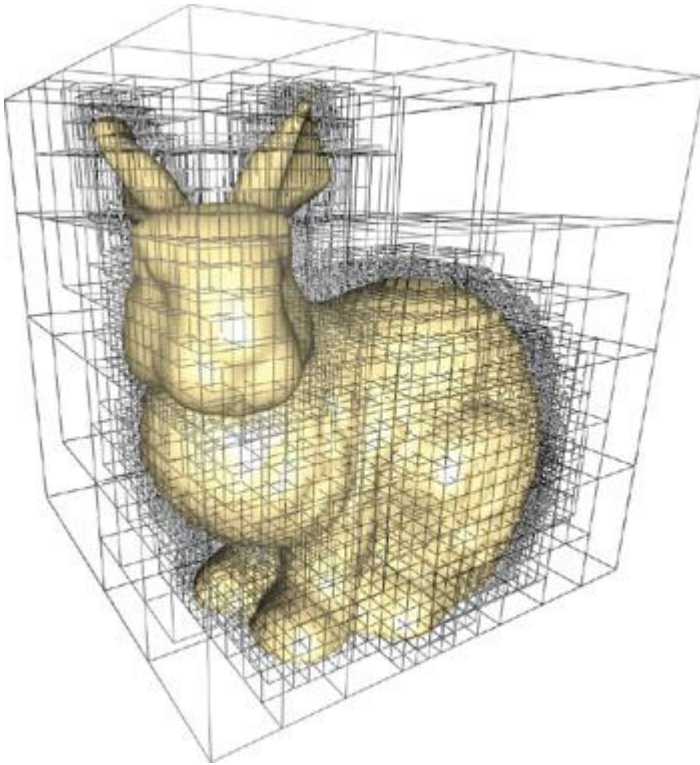
Each node represents a square. If the node is fully empty or fully occupied it has no children.
If it is partially occupied it has four children. Subdivision stops after some minimal square size.

Octrees

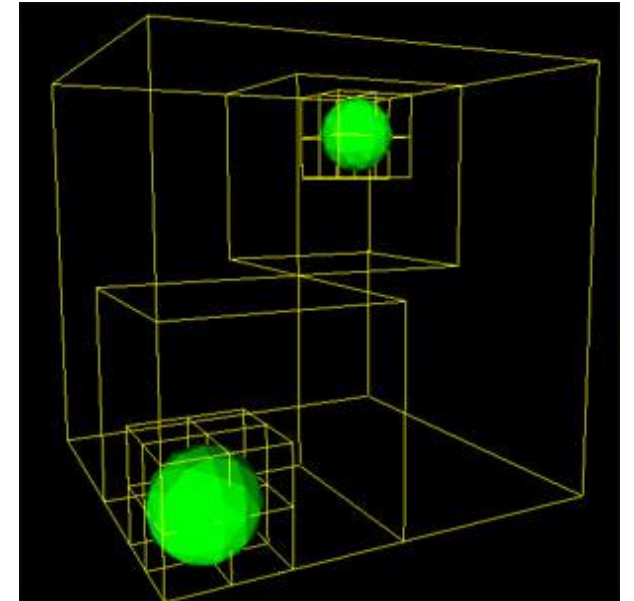


Each node represents a cube. If the node is fully empty or fully occupied it has no children.
If it is partially occupied it has eight children. Subdivision stops after some minimal cube size.

Octrees

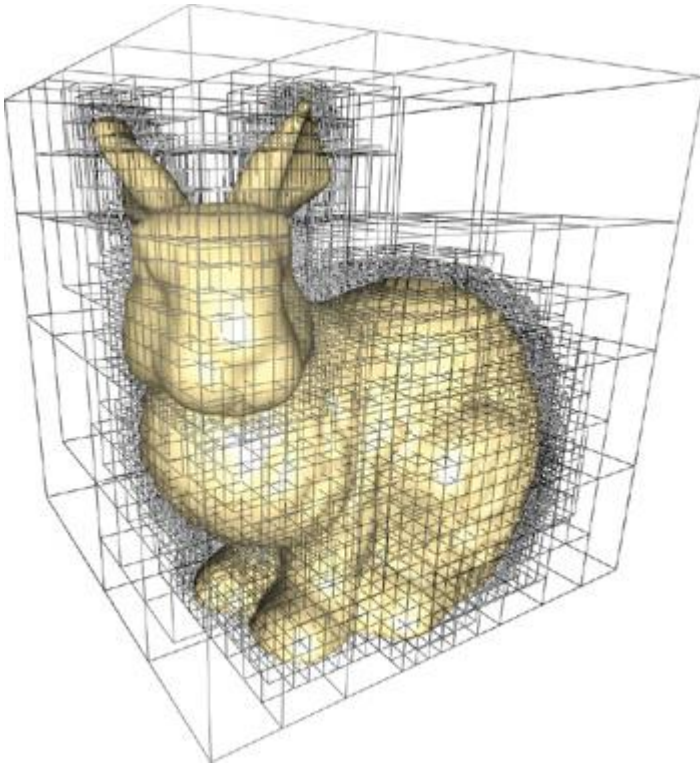


Problem 1: quadtrees and octrees are not balanced trees. So, in the worst case an occupancy query could be $O(n)$ in the number of nodes.



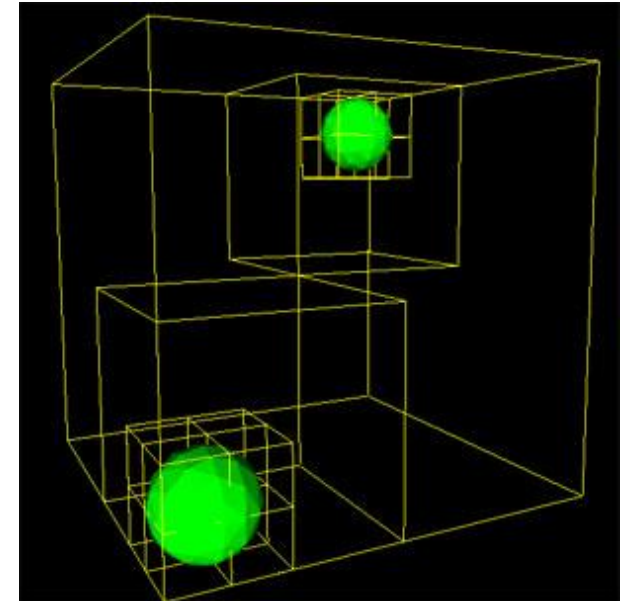
Each node represents a cube. If the node is fully empty or fully occupied it has no children.
If it is partially occupied it has eight children. Subdivision stops after some minimal cube size.

Octrees



Problem 1: quadtrees and octrees are not balanced trees. So, in the worst case an occupancy query could be $O(n)$ in the number of nodes.

Problem 2: quadtrees and octrees are sensitive to small changes in the location of obstacles.



Each node represents a cube. If the node is fully empty or fully occupied it has no children. If it is partially occupied it has eight children. Subdivision stops after some minimal cube size.

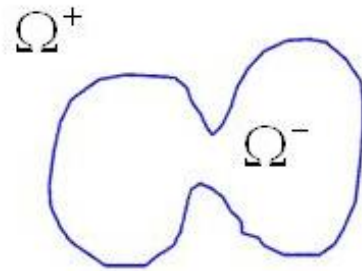
Octree Example: Octomap

Open source
as a ROS package

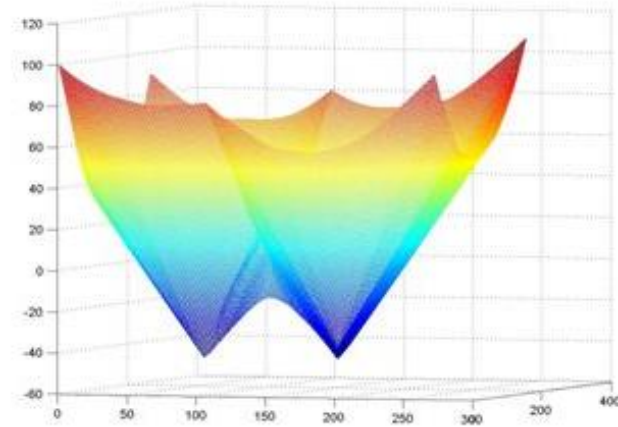


Implicit Surface Definitions: Signed Distance Function

Contour C



Signed distance function



This distance function
is defined over any point
in 3D space.

A signed distance function is defined by :

$$\phi(x) = \begin{cases} \text{dist}(x, C) & \text{if } x \text{ is outside } C \\ 0 & x \in C \\ -\text{dist}(x, C) & \text{if } x \text{ is inside } C \end{cases}$$

SDF Example

Real-Time Camera Tracking and 3D Reconstruction Using Signed Distance Functions

Erik Bylow, Jürgen Sturm, Christian Kerl,
Fredrik Kahl, Daniel Cremers

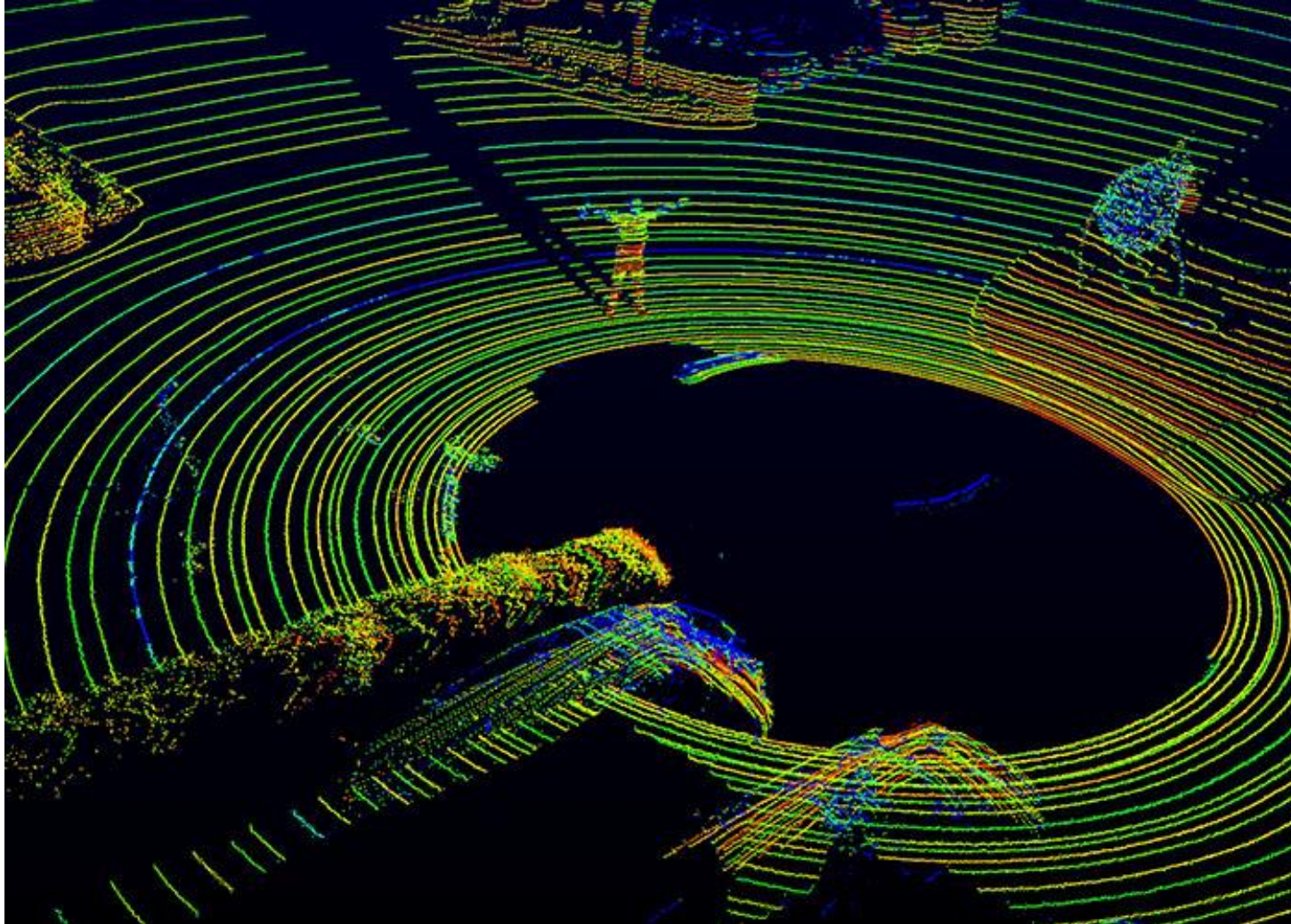
Robotics: Science and Systems (RSS)
June 2013



Computer Vision Group
Department of Computer Science
Technical University of Munich



Pointclouds



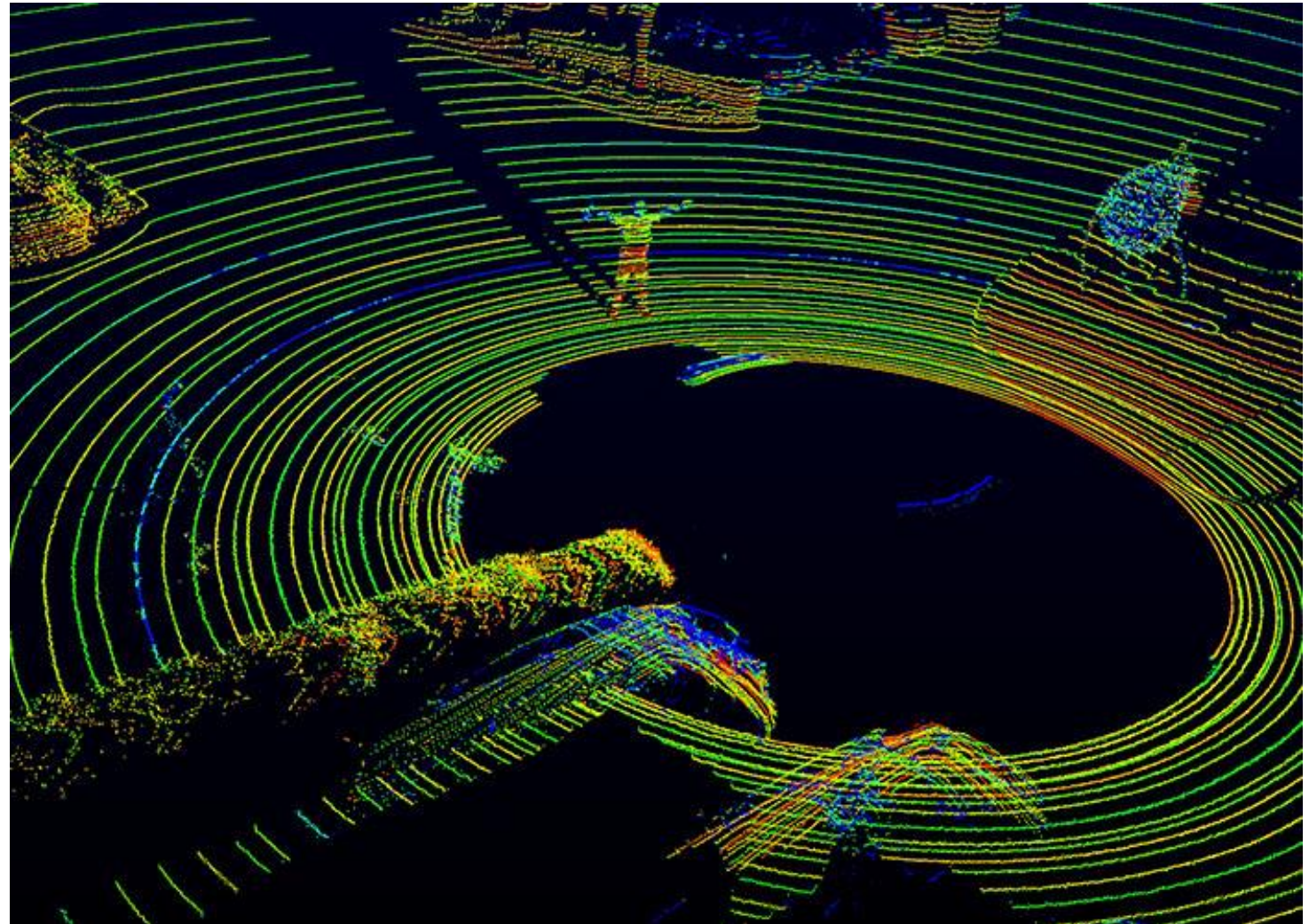
Pointclouds

Advantages:

- can make local changes to the map without affecting the pointcloud globally
- can align pointclouds
- nearest neighbor queries are easy with kd-trees or locality-sensitive hashing

Disadvantages:

- need to segment objects in the map
- raytracing is approximate and nontrivial



Topological Maps

Topology: study of spatial properties that are preserved under continuous deformations of the space.

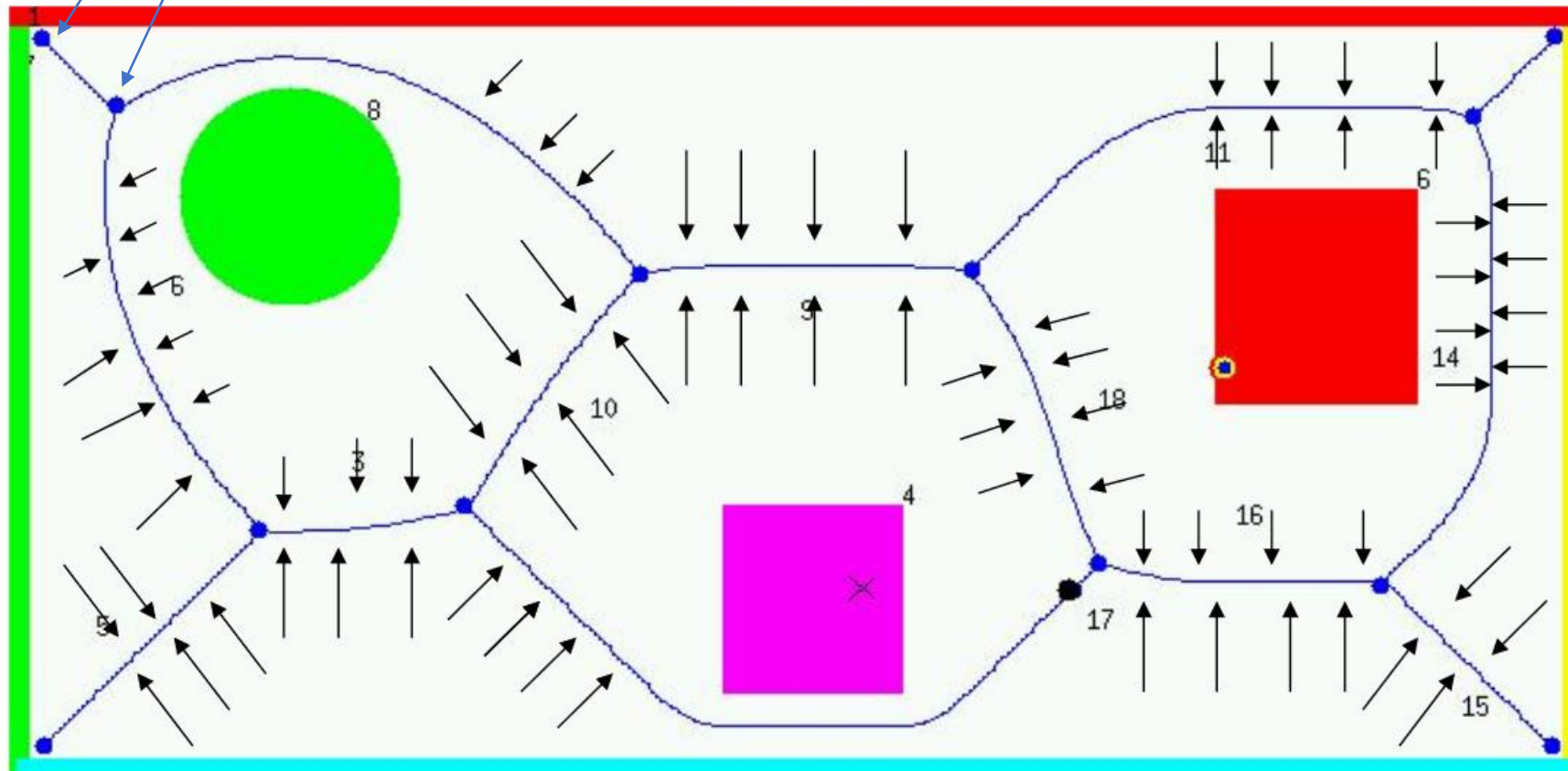
Generalized Voronoi Graph (GVG)



Deformation Retraction: GVG in Plane

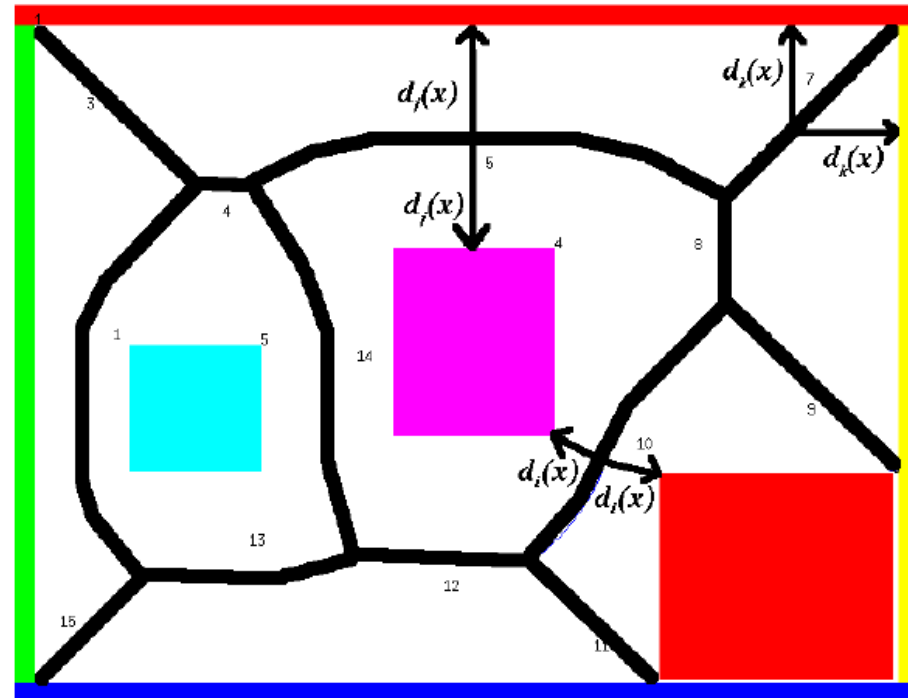
GVG nodes: points that are equidistant to 3 or more obstacle points

Retractions are also called roadmaps.



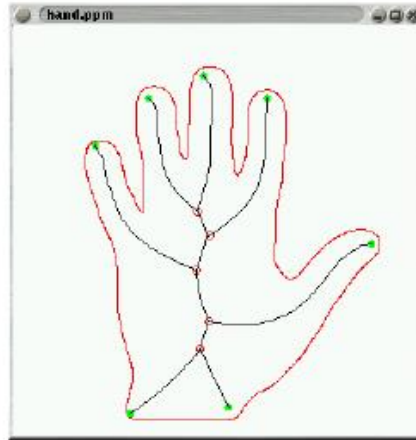
Roadmap: Voronoi diagrams

- GVG is formed by paths equidistant from the two closest objects
- maximizing the clearance between the obstacles.



- This generates a very safe roadmap which avoids obstacles as much as possible

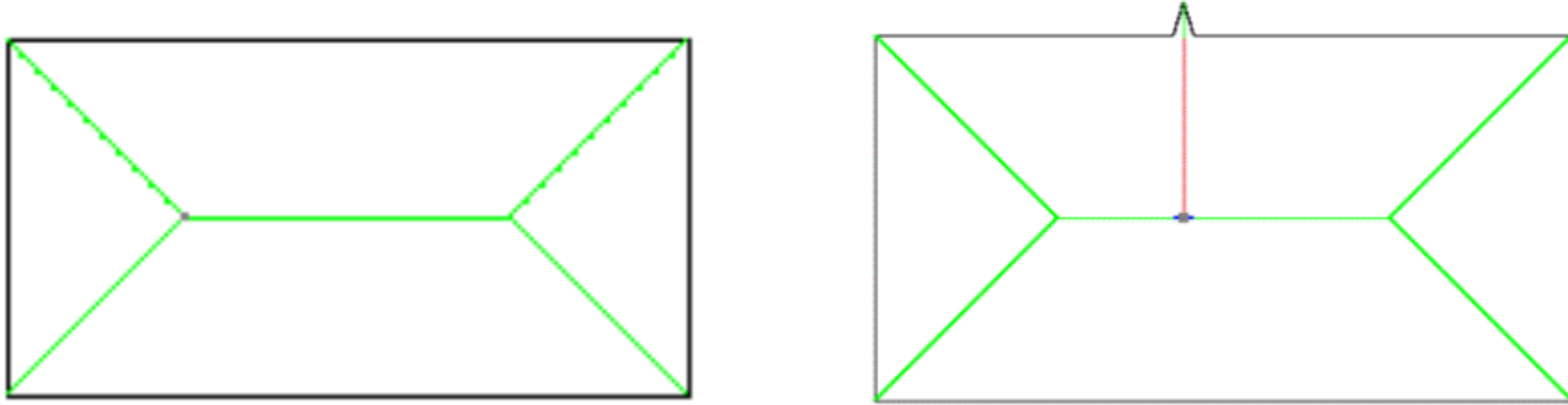
Generalized Voronoi Graphs (GVG)



in 2d, it's called
a medial axis

Turns comparison between pixels to comparison between graphs.

GVG: sensitivity



The skeleton is sensitive to small changes in the object's boundary.

GVG: advantages

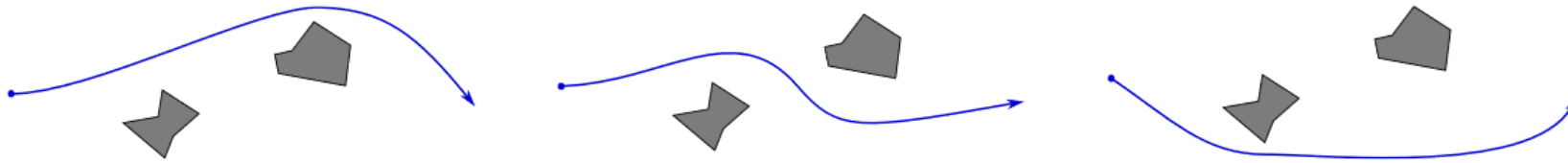
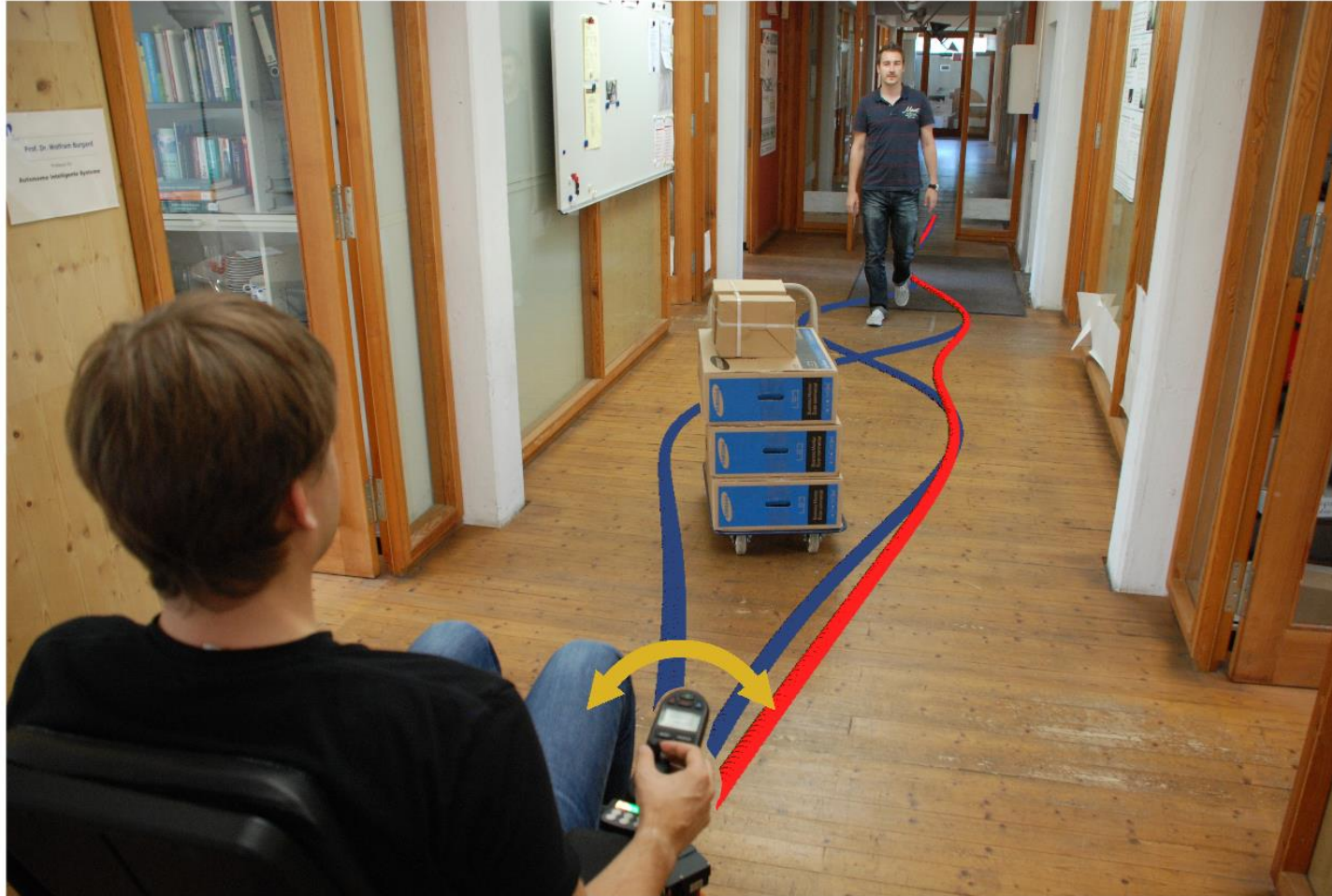


Figure 4.1: Example of three homotopically distinct composite trajectories in the same situation. The agent depicted in blue passes the solid obstacles on different sides. With fixed start- and end points these trajectories cannot be continuously transformed into each other without colliding with obstacles.

Can specify whether we pass on the “left” or “right” of each obstacle on our way to the goal.

GVG: advantages



How a curve winds around an obstacle

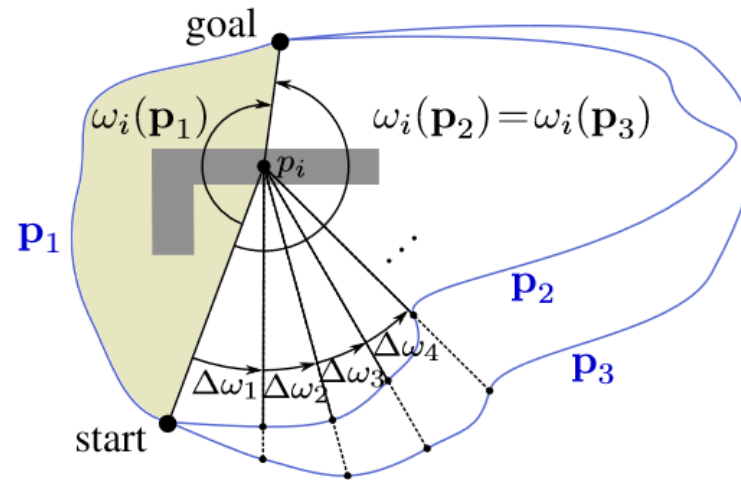


Figure 4.7: Computation of the winding numbers with respect to obstacles. The figure shows three paths \mathbf{p}_1 , \mathbf{p}_2 and \mathbf{p}_3 that bypass the obstacle i . The infinitesimal angles $\Delta\omega$ sum up to the winding number ω_i around the representative point p_i of the obstacle. The two paths on the right yield the same winding number $\omega_i(\mathbf{p}_2) = \omega_i(\mathbf{p}_3)$ in contrast to the path on the left.

How a curve winds around an obstacle

Note: winding angle of a path can be more than 360 degrees

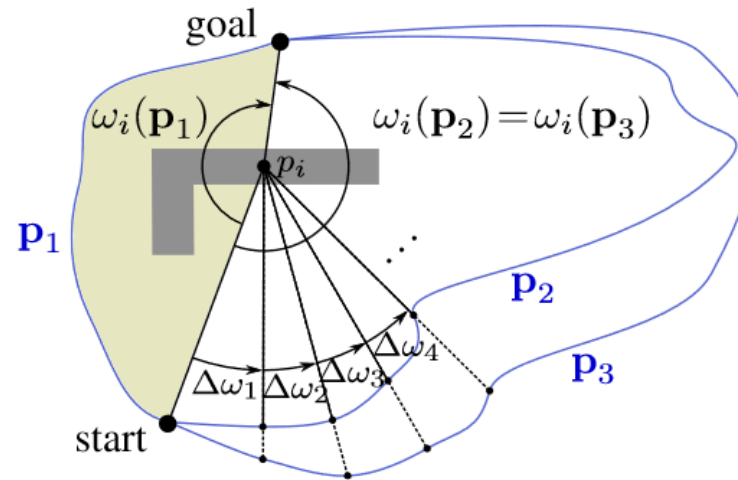


Figure 4.7: Computation of the winding numbers with respect to obstacles. The figure shows three paths p_1 , p_2 and p_3 that bypass the obstacle i . The infinitesimal angles $\Delta\omega$ sum up to the winding number ω_i around the representative point p_i of the obstacle. The two paths on the right yield the same winding number $\omega_i(p_2) = \omega_i(p_3)$ in contrast to the path on the left.

Homotopy classes

Two paths with the same endpoints are homotopic or belong to the same homotopy class iff one can be deformed continuously (without hitting obstacles) into the other. Formally, the paths:

$$\tau_1 : [0, T] \rightarrow \mathbb{R}^2$$

$$\tau_2 : [0, T] \rightarrow \mathbb{R}^2$$

with

$$\tau_1(0) = \tau_2(0)$$

$$\tau_1(T) = \tau_2(T)$$

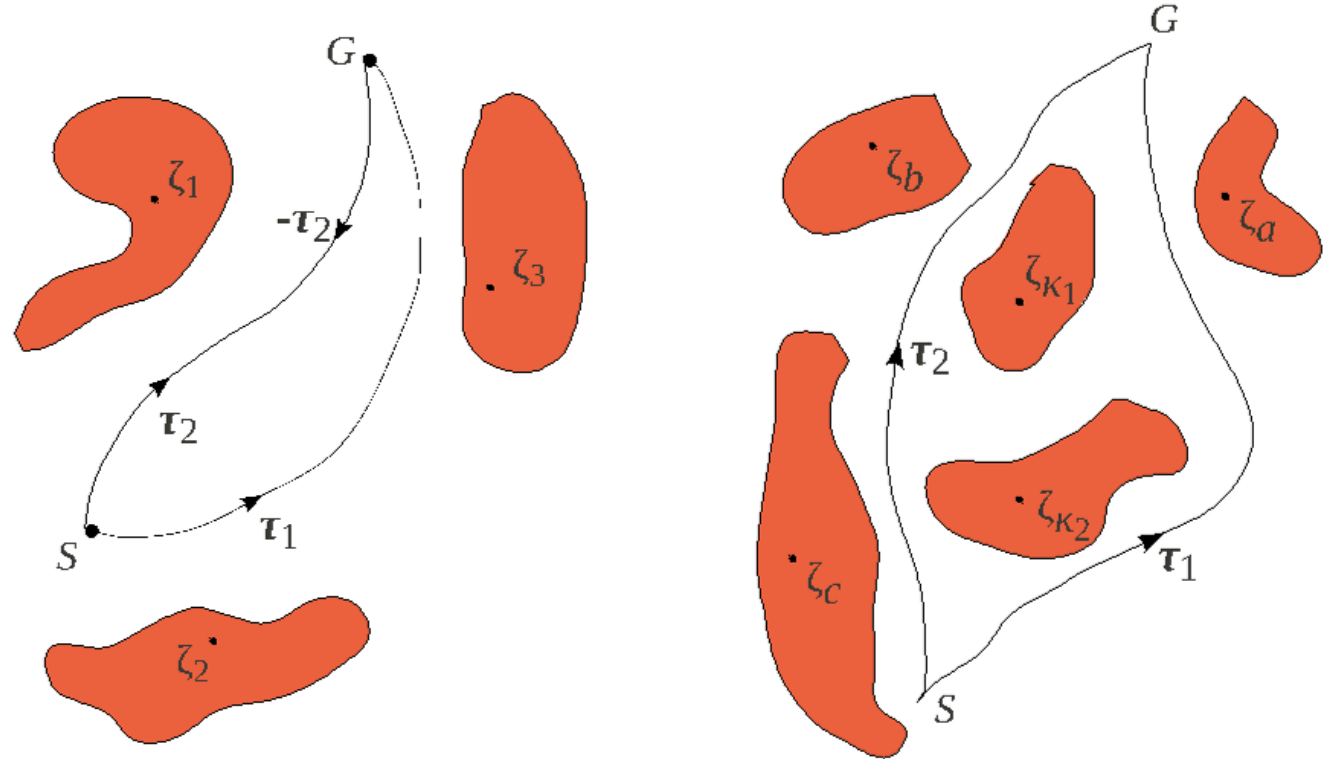
are homotopic iff there exists a continuous function

$$H : [0, 1] \times [0, T] \rightarrow \mathbb{R}^2$$

such that for any time t :

$$H(0, t) = \tau_1(t)$$

$$H(1, t) = \tau_2(t)$$



(a) In same Homotopy class, forming a closed contour.

(b) In different Homotopy classes, enclosing obstacles.

Homotopy functions for deformations

Two paths with the same endpoints are homotopic or belong to the same homotopy class iff one can be deformed continuously (without hitting obstacles) into the other. Formally, the paths:

$$\tau_1 : [0, T] \rightarrow \mathbb{R}^2$$

$$\tau_2 : [0, T] \rightarrow \mathbb{R}^2$$

with

$$\tau_1(0) = \tau_2(0)$$

$$\tau_1(T) = \tau_2(T)$$

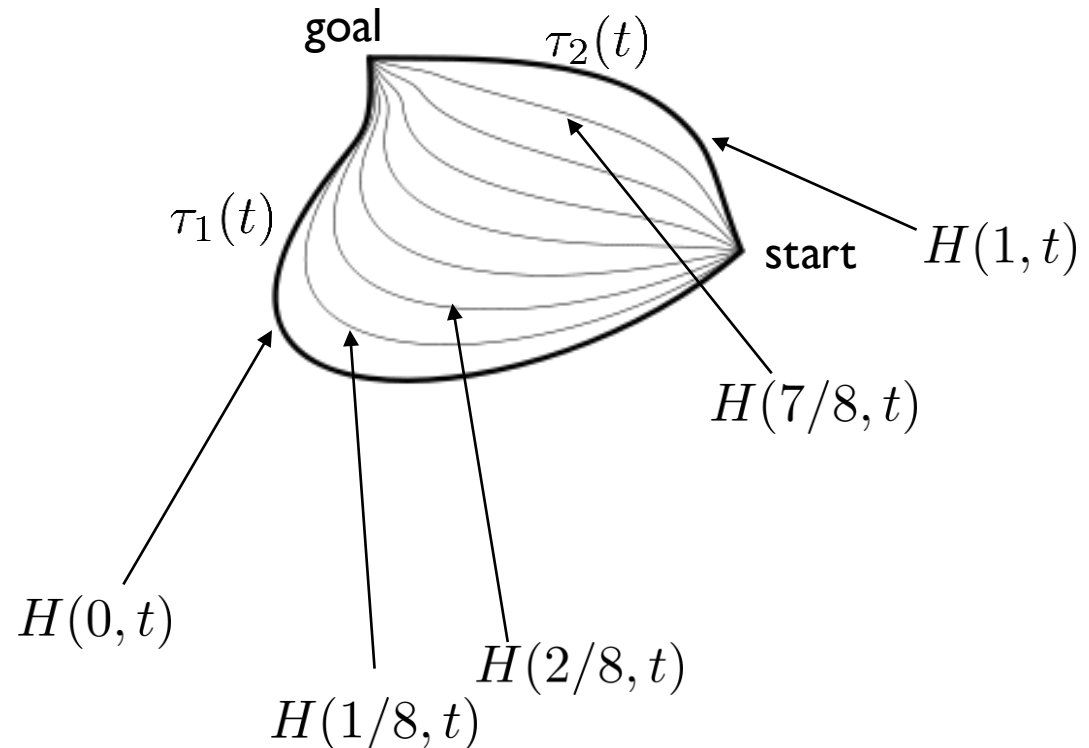
are homotopic iff there exists a continuous function

$$H : [0, 1] \times [0, T] \rightarrow \mathbb{R}^2$$

such that for any time t :

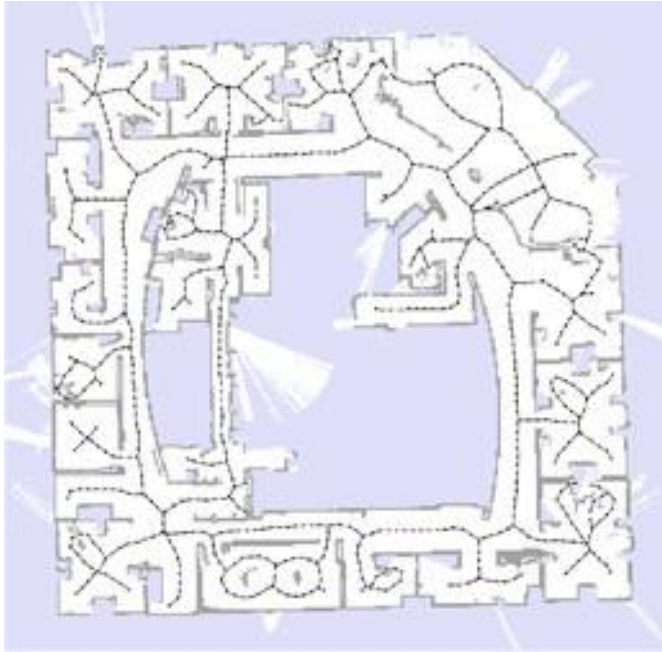
$$H(0, t) = \tau_1(t)$$

$$H(1, t) = \tau_2(t)$$

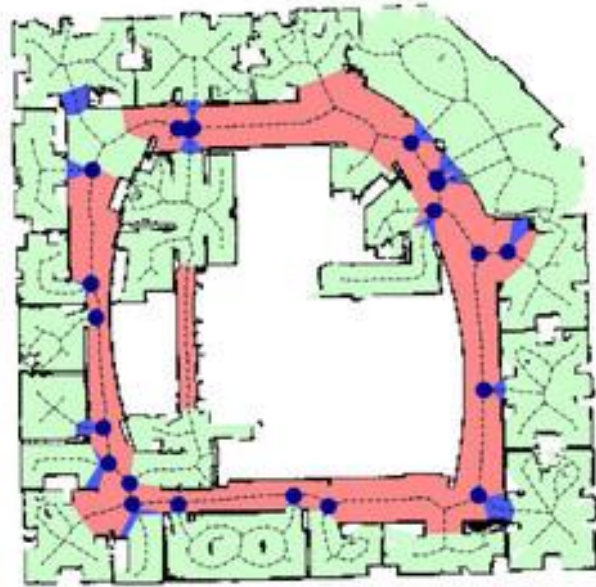


Topometric Maps

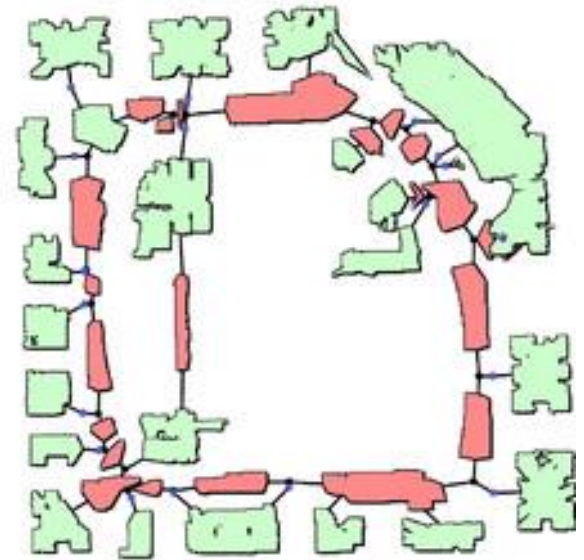
Topometric maps



Occupancy grid

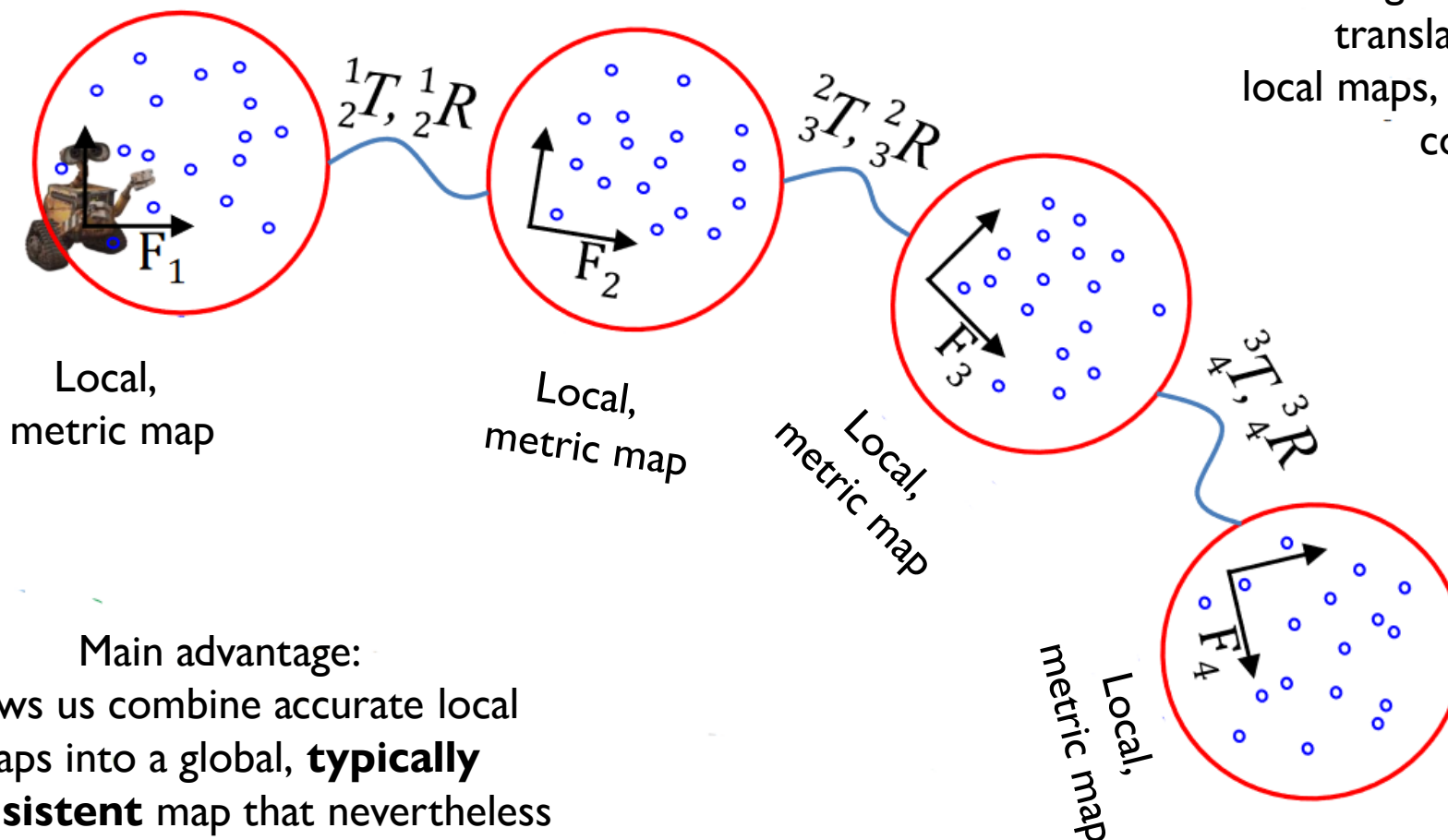


Topological map



Topometric map

Topometric maps



Edges: rotations and translations between local maps, but also topological connectivity

Main advantage:
allows us combine accurate local maps into a global, **typically inconsistent** map that nevertheless provides sufficient navigation information.

Maps of Raw Observations

Main Idea

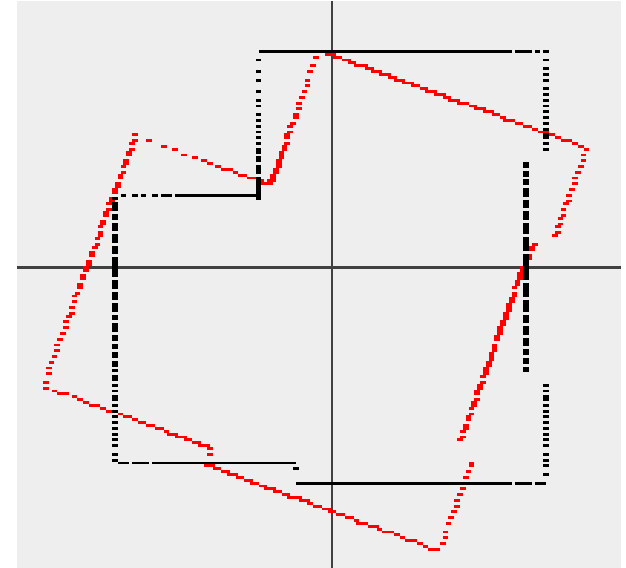
- Map = entire (unprocessed) sequence of observations, e.g. video.
- Do not try to support distance, collision, and raytracing queries.
- Instead, provide only a similarity/nearest neighbors query
 - “Find the image in the video that is most similar to the one I’m seeing now.”
- History of observations determines a (set of) location(s) in the map

Metric Map Alignment

a.k.a. scan matching, a.k.a. iterative closest point (ICP), a.k.a. registration

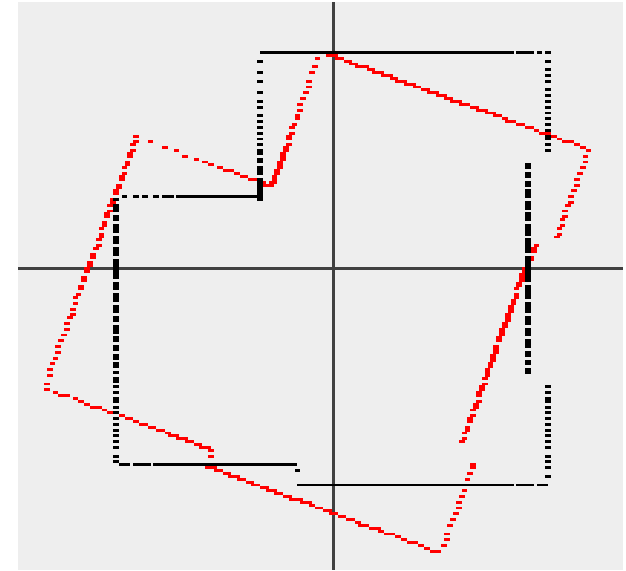
Problem definition

- Given
 - two pointclouds or
 - a (local) laser scan and a pointcloud (global map) or
 - two maps
- find the rotation and translation that aligns them



Problem definition

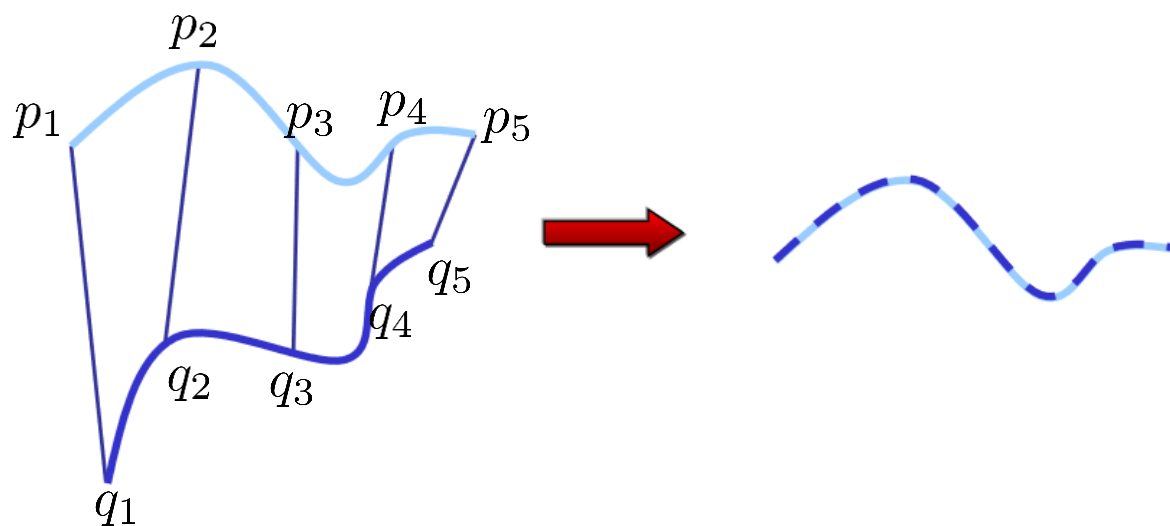
- Given
 - two pointclouds or
 - a (local) laser scan and a pointcloud (global map) or
 - two maps
- find the rotation and translation that aligns them



- Assumption: We are assuming in these slides for simplicity that rigid-body transformations are sufficient to align the scans. They might not be. We might need to also model scaling, non-uniform stretching and other nonlinear transformations.

Scan alignment with known correspondences

If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form.

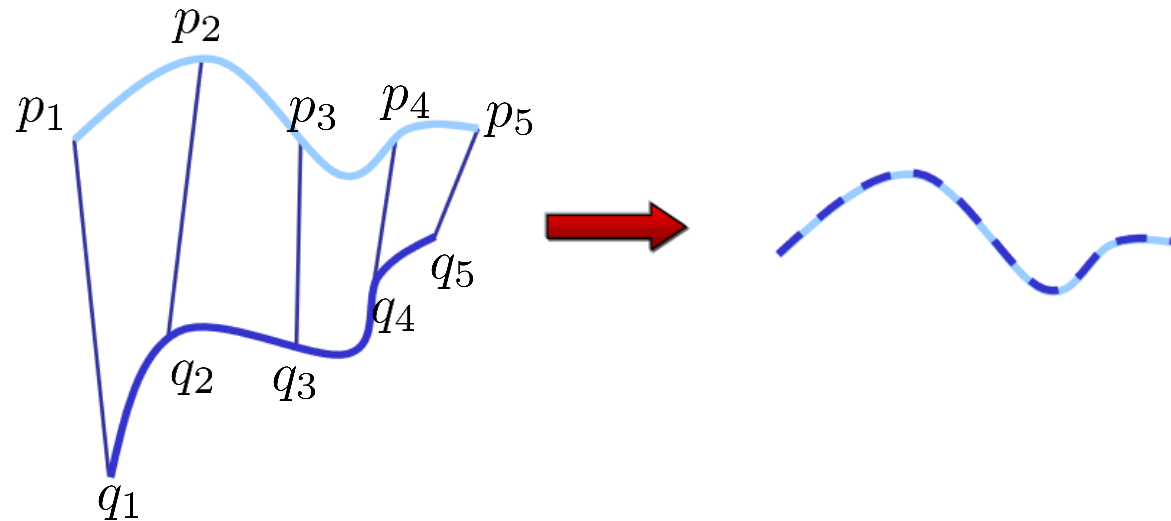


Scan alignment with known correspondences

If the correct correspondences are known, the correct relative rotation/translation can be calculated in closed form.

When correct correspondences are known we say that **data association** is known/unambiguous.

In general, data association is a real and hairy problem in robotics.

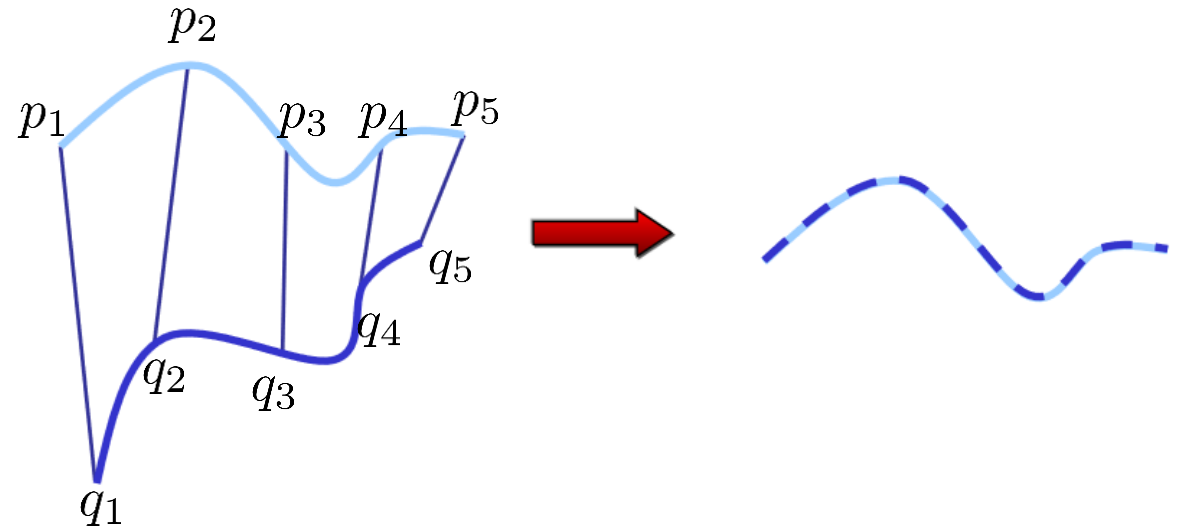


Scan alignment with known correspondences

Find the 3D rotation matrix R and the 3D translation vector t that will best align the corresponding points

$$\text{error}(R, t) = \frac{1}{N} \sum_{i=1}^N \|p_i - (Rq_i + t)\|^2$$

$$R^*, t^* = \underset{R, t}{\operatorname{argmin}} \text{error}(R, t)$$

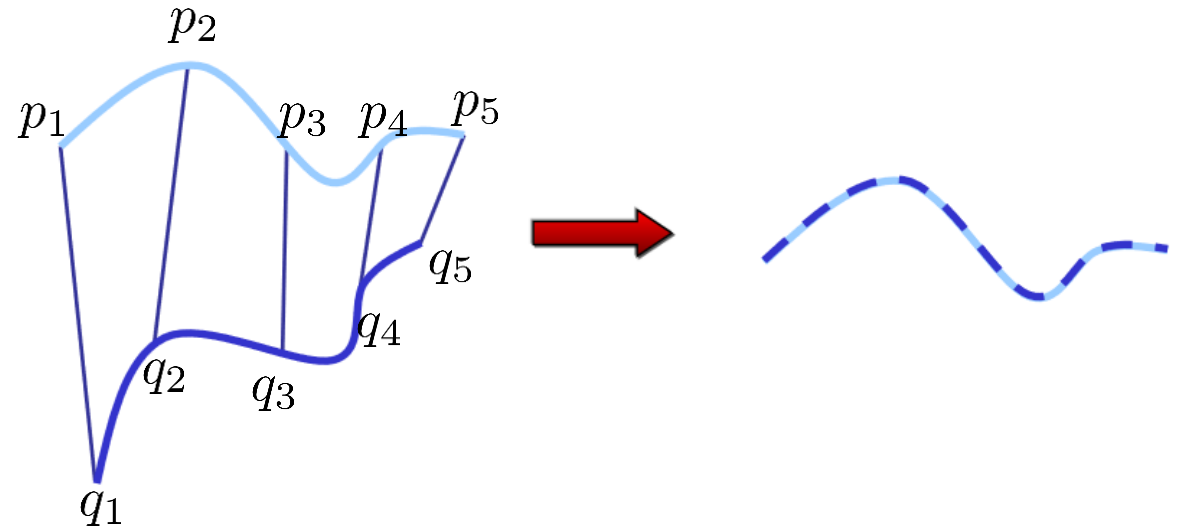


Scan alignment with known correspondences

Find the 3D rotation matrix R and the 3D translation vector t that will best align the corresponding points

$$\text{error}(R, t) = \frac{1}{N} \sum_{i=1}^N \|p_i - (Rq_i + t)\|^2$$

$$R^*, t^* = \underset{R, t}{\operatorname{argmin}} \text{error}(R, t)$$



Q: How do we minimize this error?

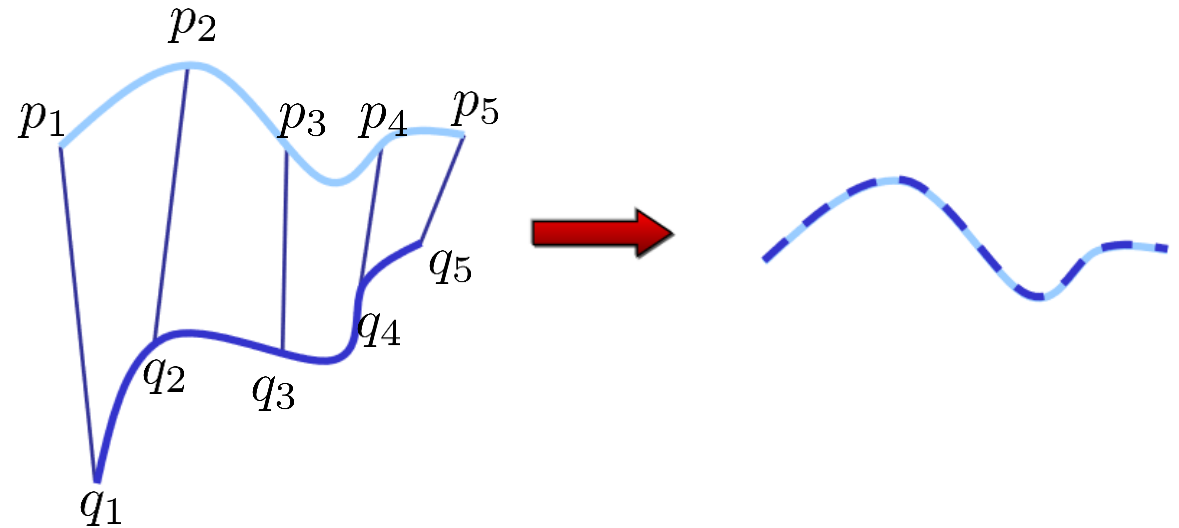
A: Turns out it has a closed-form solution.

Closed form solution of scan alignment with known correspondences

Find the 3D rotation matrix R and the 3D translation vector t that will best align the corresponding points

$$\text{error}(R, t) = \frac{1}{N} \sum_{i=1}^N \|p_i - (Rq_i + t)\|^2$$

$$R^*, t^* = \underset{R, t}{\operatorname{argmin}} \text{error}(R, t)$$



Step 1: compute the means of the two scans

$$\mu_p = \frac{1}{N} \sum_{i=1}^N p_i \quad \mu_q = \frac{1}{N} \sum_{i=1}^N q_i$$

Step 2: subtract the means from the scans

$$\bar{p}_i = p_i - \mu_p \quad \bar{q}_i = q_i - \mu_q$$

Step 3: form the matrix

$$W = \sum_{i=1}^N \bar{p}_i \bar{q}_i^T$$

Closed form solution of scan alignment with known correspondences

Find the 3D rotation matrix R and the 3D translation vector t that will best align the corresponding points

$$\text{error}(R, t) = \frac{1}{N} \sum_{i=1}^N \|p_i - (Rq_i + t)\|^2$$

$$R^*, t^* = \underset{R, t}{\operatorname{argmin}} \text{error}(R, t)$$

Step 1: compute the means of the two scans

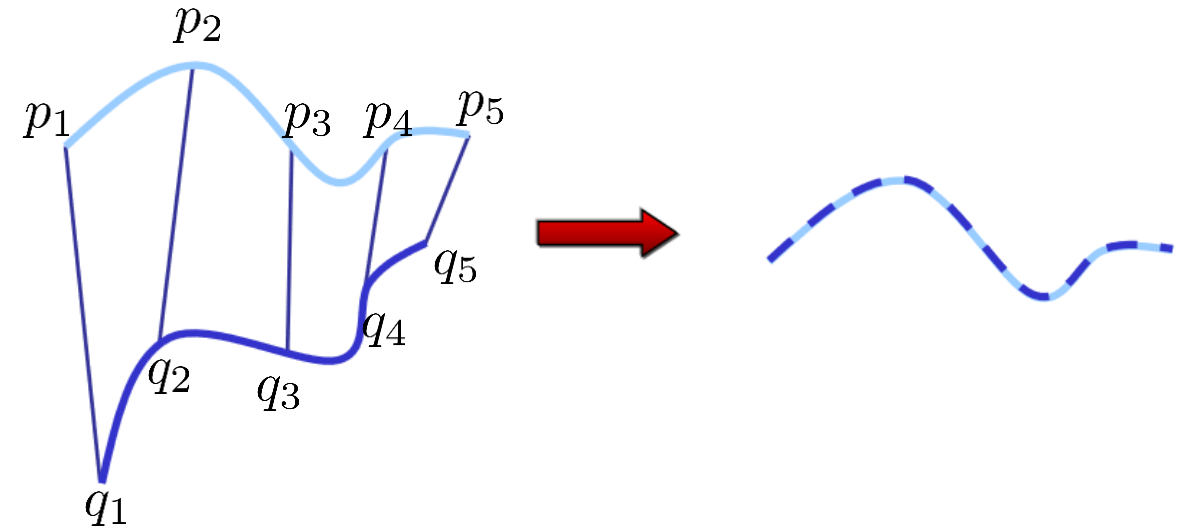
$$\mu_p = \frac{1}{N} \sum_{i=1}^N p_i \quad \mu_q = \frac{1}{N} \sum_{i=1}^N q_i$$

Step 2: subtract the means from the scans

$$\bar{p}_i = p_i - \mu_p \quad \bar{q}_i = q_i - \mu_q$$

Step 3: form the matrix

$$W = \sum_{i=1}^N \bar{p}_i \bar{q}_i^T$$



Step 4: compute the SVD of the matrix W

$$W = U \Sigma V^T$$

Step 5: if $\text{rank}(W)=3$, optimal solution is unique:

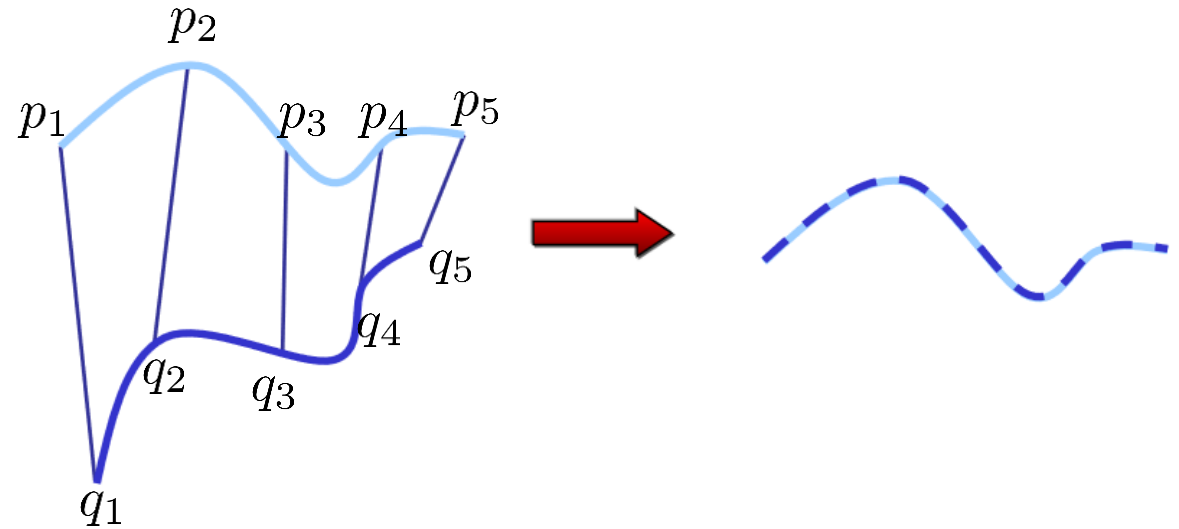
$$R^* = UV^T \quad t^* = \mu_p - R^* \mu_q$$

Closed form solution of scan alignment with known correspondences

Find the 3D rotation matrix R and the 3D translation vector t that will best align the corresponding points

$$\text{error}(R, t) = \frac{1}{N} \sum_{i=1}^N \|p_i - (Rq_i + t)\|^2$$

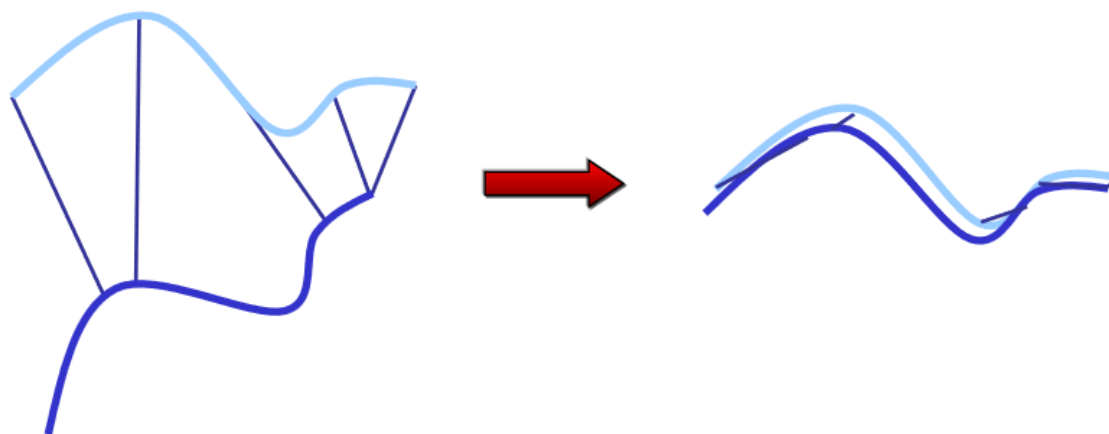
$$R^*, t^* = \underset{R, t}{\operatorname{argmin}} \text{error}(R, t)$$



If you're interested, the proof of the closed-form solution can be found in:
K. S. Arun, T. S. Huang, and S. D. Blostein. Least square fitting of two 3-d point sets.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(5):698 – 700, 1987

Scan alignment with unknown correspondences

- | If correct correspondences are not known, it is generally impossible to determine the optimal relative rotation/translation in one step



Scan alignment with unknown correspondences

ICP-Algorithm

- Idea: iterate to find alignment
- Iterated Closest Points (ICP)
[Besl & McKay 92]
- Converges if starting positions are “close enough”



Main idea for data association:

- associate each point in the source scan to its nearest neighbor in the target scan

Find optimal rotation and translation for this correspondence.

Repeat until no significant drop in error.

libpointmatcher

STUDY CASE: **Mapping a Campus**

Velodyne HDL-64e

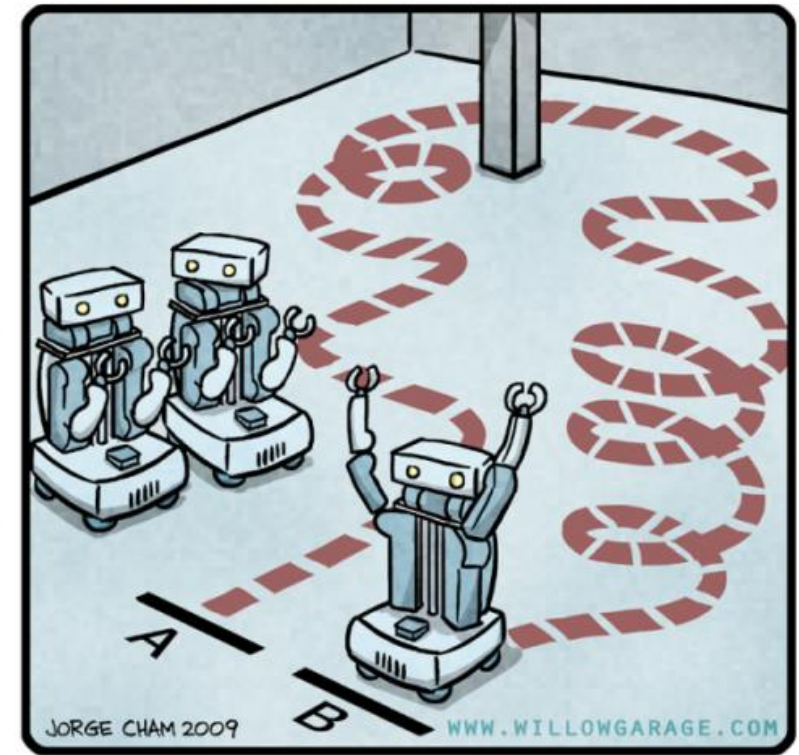
François Pomerleau
Tyler Daoust
Tim Barfoot

Feb 9th, 2016



Today's agenda

- How to represent maps
- Probabilistic occupancy grid mapping



"HIS PATH-PLANNING MAY BE
SUB-OPTIMAL, BUT IT'S GOT FLAIR."

What we want to do

EECS568

Mobile Robotics: Methods & Algorithms

Instructor: Prof. Ryan M. Eustice

Mobile Robot Occupancy Grid Mapping

Algorithm implemented in MATLAB
Footage from ZZ's course homework 4

Terminology

- Pose: the rotation and translation of a robot
- Odometry: the transformation of the body frame with respect to its initial pose (fixed frame of reference).

$${}^{B_0}T_{B_t}$$

- Dynamics model: what is the next state given current state and control?

$$x_{t+1} = f(x_t, u_t)$$

- Sensor measurement model: what is the expected measurement given the robot's current state?

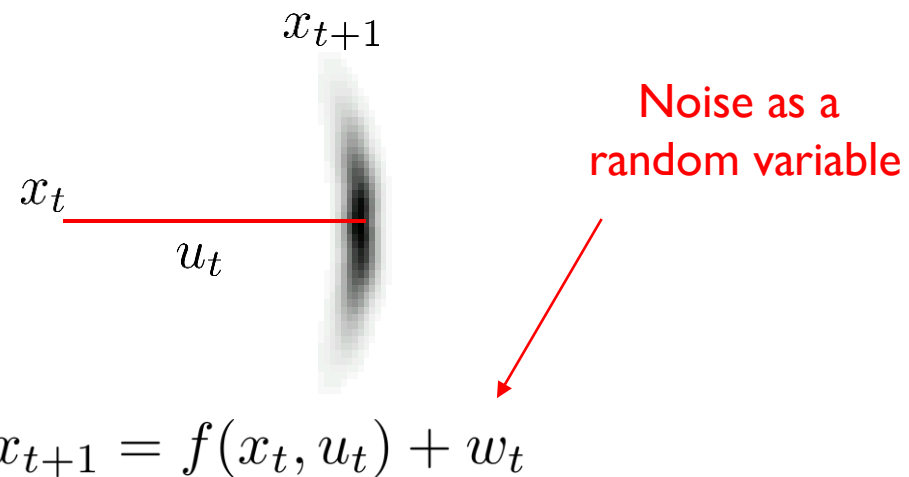
$$z_t = h(x_t)$$

Perfect models vs. Reality

Dynamics



$$x_{t+1} = f(x_t, u_t)$$



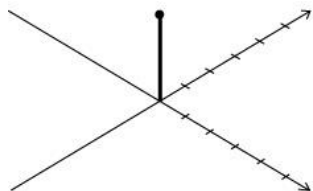
Perfect models vs. Reality

Sensor
Measurements

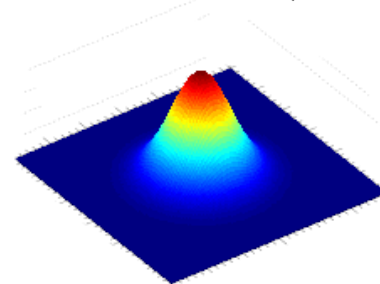
$$z_t = h(x_t)$$

e.g. GPS (simplified)

$$z_t = x_t$$



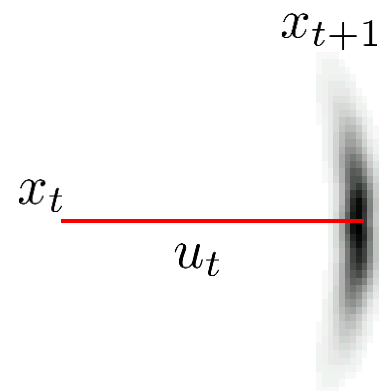
$$z_t = x_t + v_t, \quad v_t \sim \mathcal{N}(0, \sigma^2 I)$$



Perfect models vs. Reality

Dynamics

$$x_{t+1} = f(x_t, u_t)$$



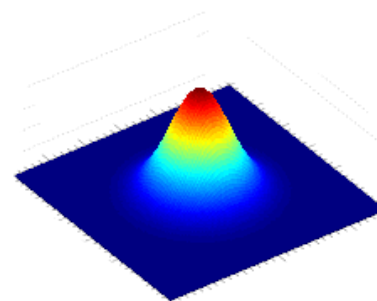
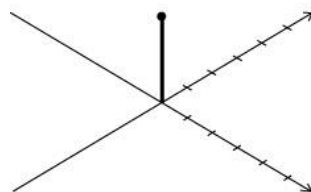
$$p(x_{t+1} | x_t, u_t)$$

probabilistic
dynamics model

Sensor
Measurements

$$z_t = h(x_t)$$

$$z_t = x_t$$



$$p(z_t | x_t)$$

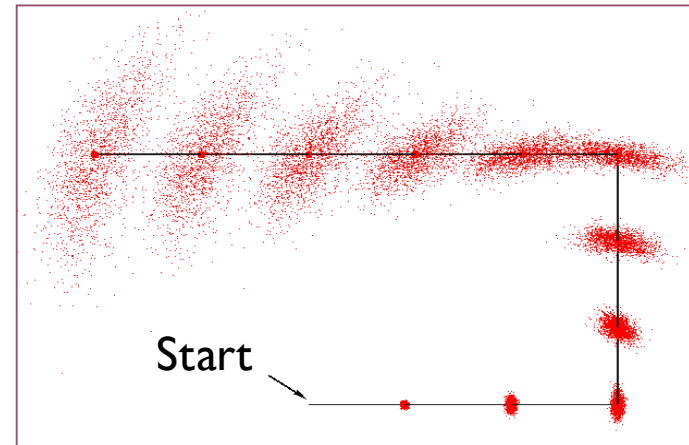
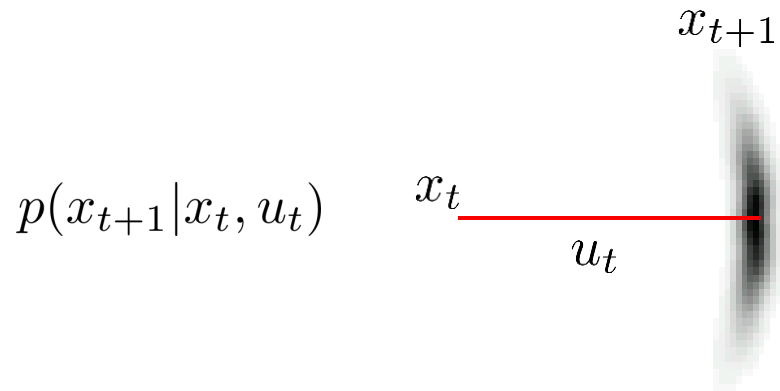
probabilistic
measurement model

e.g. GPS (simplified)

Why is mapping a problem?

Don't we have all the information we need to build a map?

- Two main sources of uncertainty:
 - accumulating uncertainty in the dynamics



Uncertainty in the dynamics compounds into increasing uncertainty in odometry, as time passes.

Why is mapping a problem?

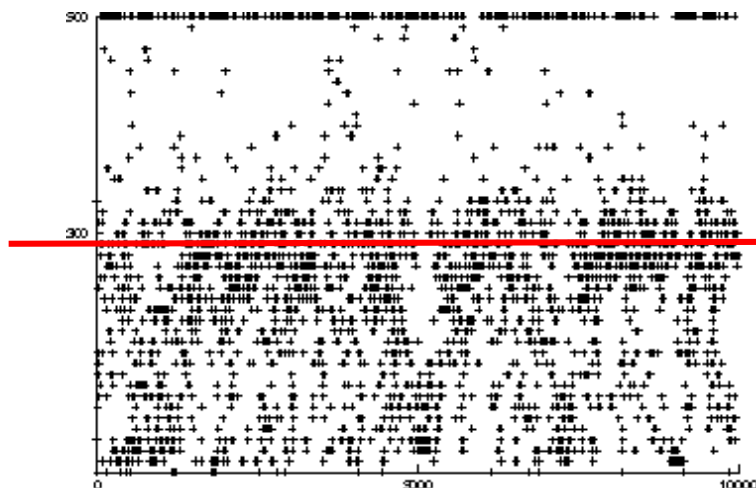
Don't we have all the information we need to build a map?

- Two main sources of uncertainty:
 - uncertainty in the dynamics $p(x_{t+1}|x_t, u_t)$
 - uncertainty in sensor measurements

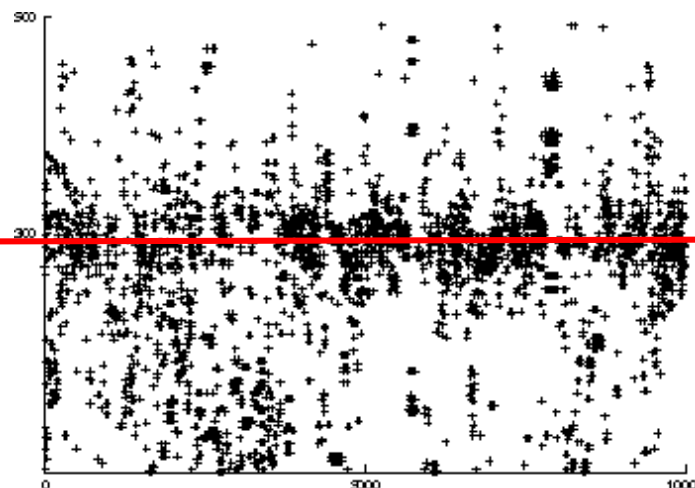
Why is mapping a problem?

Don't we have all the information we need to build a map?

- Two main sources of uncertainty:
 - uncertainty in the dynamics $p(x_{t+1}|x_t, u_t)$
 - uncertainty in sensor measurements



Sonar



Laser

True value: 300cm

but measurements have noise

Why is mapping a problem?

Don't we have all the information we need to build a map?


- If we had no uncertainty, i.e. $x_{t+1} = f(x_t, u_t)$ and $z_t = h(x_t)$ then mapping would be trivial.
- Today we will assume perfect dynamics and odometry, but noisy sensor measurements. $p(z_t|x_t)$
- We are also going to assume a static map, no moving objects

Defining the problem

- The occupancy grid map is a binary random variable

$$\mathbf{m} = \{m_{ij}\} \in \{0, 1\}^{W \times H}$$

width = #columns
height = #rows
of the occupancy grid



Defining the problem

- The occupancy grid map is a binary random variable

$$\mathbf{m} = \{m_{ij}\} \in \{0, 1\}^{W \times H}$$

width = #columns
height = #rows
of the occupancy grid

- The path of the robot up to time t is a sequence of random variables

$$\mathbf{X}_{1:t} = \mathbf{X}_1, \dots, \mathbf{X}_t \text{ with } \mathbf{x}_i = (x_i, y_i, \theta_i)$$

Odometry coordinates

Defining the problem

- The occupancy grid map is a binary random variable

$$\mathbf{m} = \{m_{ij}\} \in \{0, 1\}^{W \times H}$$

width = #columns
height = #rows
of the occupancy grid

- The path of the robot up to time t is a sequence of random variables

$$\mathbf{X}_{1:t} = \mathbf{X}_1, \dots, \mathbf{X}_t \quad \text{with} \quad \mathbf{x}_i = (x_i, y_i, \theta_i)$$

Odometry coordinates

- At each time step the robot makes a measurement (sonar/laser).

Measurements up to time t are a sequence of random variables

$$\mathbf{Z}_{1:t} = \mathbf{Z}_1, \dots, \mathbf{Z}_t \quad \text{with} \quad \mathbf{z}_i = \{(r_i, \psi_i)\}^K$$

(range, angle) in
the laser's local
coordinates

K = #beams, or
#points in the scan

The goal of mapping

- To estimate the probability of any map, given path and measurements

$$belief_t(\mathbf{m}) = p(\mathbf{m} | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

Sensor
Measurements

Odometry / Robot Poses

The goal of mapping

- To estimate the probability of any map, given path and measurements

$$p(\mathbf{m} | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- This is intractable. E.g. for a 100 x 100 grid there are 2^{10000} possible binary maps.

The goal of mapping

- To estimate the probability of any map, given path and measurements

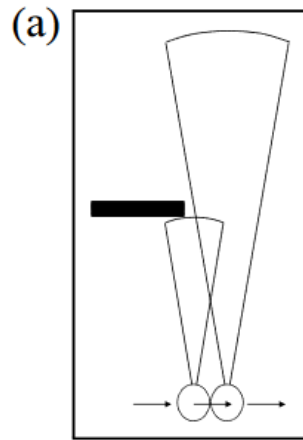
$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- This is intractable. E.g. for a 100 x 100 grid there are 2^{10000} possible binary maps.

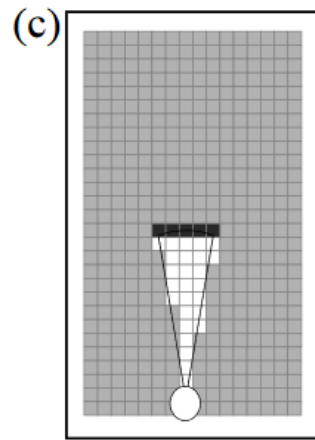
- We can approximate $p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \simeq \prod_{i,j} p(m_{ij}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$

Approximation ignores all dependencies
between map cells, given known info.
Assumes (for tractability) that cells are
independent given path and measurements

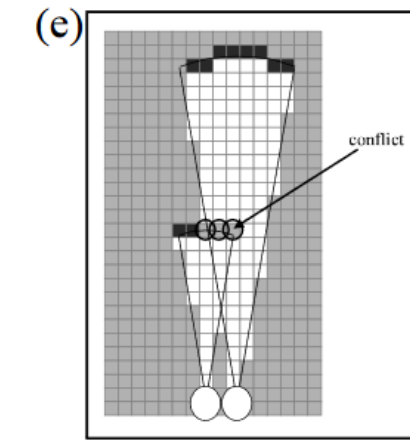
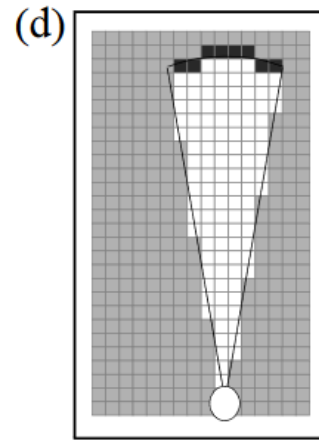
Why is it an approximation?



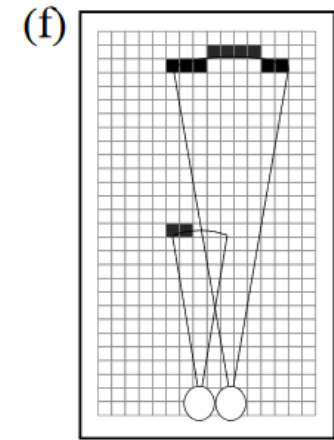
Scenario



Nearby
measurements



Resulting map when
considering cells
independently



Resulting map when
considering cells
jointly

Evaluating the occupancy of a map cell

- How do we evaluate $p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$?

Evaluating the occupancy of a map cell

- How do we evaluate $p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$?

Bayes' Rule

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

Conditional Bayes' Rule

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

Evaluating the occupancy of a map cell

- How do we evaluate $p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$?
- Using conditional Bayes' rule we get

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t}, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Bayes' Rule

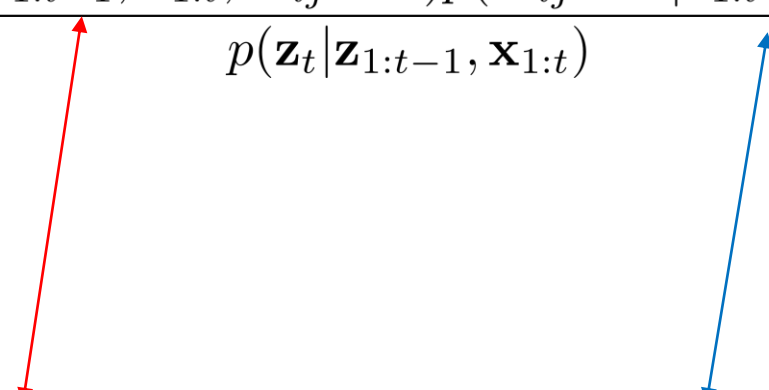
$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

Conditional Bayes' Rule

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

Evaluating the occupancy of a map cell

- How do we evaluate $p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$?
- Using conditional Bayes' rule we get

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t}, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$


- And we simplify

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

If C is independent of A given B, then C provides no extra information about A after we know B

$$p(A | B, C) = p(A | B)$$

Evaluating the occupancy of a map cell

- How do we evaluate $p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$?
- Using conditional Bayes' rule we get

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t}, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Current measurement
only depends on current
state and map cell

Current state without
current measurement provides
no additional information
about the occupancy of the map cell

- And we simplify

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Evaluating the occupancy of a map cell

- And we simplify:

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

- Another way to write this:

$$belief_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) belief_{t-1}(m_{ij} = 1)$$

- Belief at time t-1 was updated to belief at time t based on likelihood of measurement received at time t.

Evaluating the occupancy of a map cell

- And we simplify:

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

- Another way to write this:

$$belief_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) belief_{t-1}(m_{ij} = 1)$$

So, as long as we can evaluate
the measurement likelihood

$$p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)$$

and the normalization factor

$$\eta = 1 / p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})$$

we can do the belief update.

Evaluating the occupancy of a map cell

- And we simplify:

$$p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) p(m_{ij} = 1 | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

- Another way to write this:

$$belief_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) belief_{t-1}(m_{ij} = 1)$$

So, as long as we can evaluate
the measurement likelihood

$$p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)$$

and the normalization factor

$$\eta = 1 / p(\mathbf{z}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})$$

we can do the belief update.

Problem: this is hard to
compute. How can we avoid it?

The log-odds trick for binary random variables

- We showed $believe_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) believe_{t-1}(m_{ij} = 1)$
- Define the log odds ratio $l_t^{(ij)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(m_{ij} = 0 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \log \frac{believe_t(m_{ij} = 1)}{believe_t(m_{ij} = 0)}$

The log-odds trick for binary random variables

- We showed $\text{belief}_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) \text{belief}_{t-1}(m_{ij} = 1)$ (1)
- Define the log odds ratio $l_t^{(ij)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(m_{ij} = 0 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \log \frac{\text{belief}_t(m_{ij} = 1)}{\text{belief}_t(m_{ij} = 0)}$

The log-odds trick for binary random variables

- We showed $belief_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) belief_{t-1}(m_{ij} = 1)$ (1)
- Define the log odds ratio $l_t^{(ij)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(m_{ij} = 0 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \log \frac{belief_t(m_{ij} = 1)}{belief_t(m_{ij} = 0)}$
- Then (1) becomes $l_t^{(ij)} = \log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} + l_{t-1}^{(ij)}$

The log-odds trick for binary random variables

- We showed $believe_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) believe_{t-1}(m_{ij} = 1)$ (1)
- Define the log odds ratio $l_t^{(ij)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(m_{ij} = 0 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \log \frac{believe_t(m_{ij} = 1)}{believe_t(m_{ij} = 0)}$
- Then (1) becomes $l_t^{(ij)} = \log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} + l_{t-1}^{(ij)}$
- We can recover the original belief as
$$believe_t(m_{ij} = 1) = \frac{1}{1 + \exp(-l_t^{(ij)})}$$

The log-odds trick for binary random variables

- We showed $belief_t(m_{ij} = 1) = \eta p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) belief_{t-1}(m_{ij} = 1)$ (1)
- Define the log odds ratio $l_t^{(ij)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(m_{ij} = 0 | \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \log \frac{belief_t(m_{ij} = 1)}{belief_t(m_{ij} = 0)}$
- Then (1) becomes $l_t^{(ij)} = \log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} + l_{t-1}^{(ij)}$

So, as long as we can evaluate
the log odds ratio for the
measurement likelihood:

$$\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$$

we can do the belief update.

Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ to do the belief update
- We apply conditional Bayes' rule again:
$$p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) = \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t) p(\mathbf{z}_t | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$$

Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ to do the belief update
- We apply conditional Bayes' rule again: $p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) = \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t) p(\mathbf{z}_t | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- If we take the log-odds ratio: $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0 | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$

Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ to do the belief update
- We apply conditional Bayes' rule again: $p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) = \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t) p(\mathbf{z}_t | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- If we take the log-odds ratio: $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0 | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- We can simplify further:

Knowing the current state provides no information about whether cell is occupied, if there are no observations


$$\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0)}{p(m_{ij} = 1)}$$

Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ to do the belief update
- We apply conditional Bayes' rule again: $p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) = \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t) p(\mathbf{z}_t | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- If we take the log-odds ratio: $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0 | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- We can simplify further:

$$\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0)}{p(m_{ij} = 1)}$$

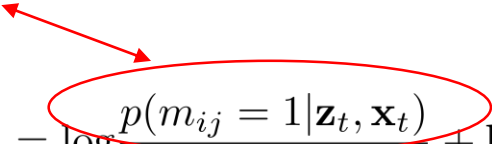
Prior probability of cell being occupied.
Can choose uniform distribution, for example.



Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ to do the belief update
- We apply conditional Bayes' rule again: $p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1) = \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t) p(\mathbf{z}_t | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- If we take the log-odds ratio: $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0 | \mathbf{x}_t)}{p(m_{ij} = 1 | \mathbf{x}_t)}$
- We can simplify further:

Inverse measurement model:
what is the likelihood of the map
cell being occupied given the current
state and current measurement?


$$\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0)}{p(m_{ij} = 1)}$$

Summary:

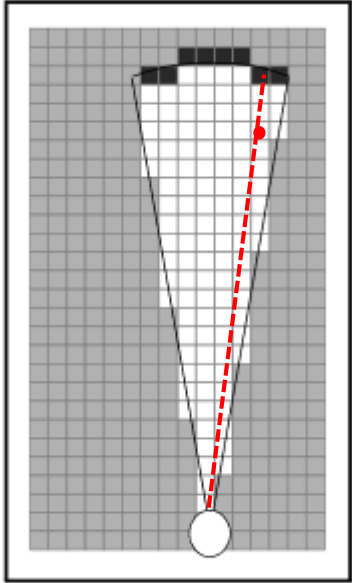
Log-odds ratio for the measurement likelihood

- We want to compute $\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)}$ but it's hard
- Instead, we can compute the log-odds ratio of the measurement likelihood in terms of the inverse measurement model:

$$\log \frac{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t | \mathbf{x}_t, m_{ij} = 0)} = \log \frac{p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0 | \mathbf{z}_t, \mathbf{x}_t)} + \log \frac{p(m_{ij} = 0)}{p(m_{ij} = 1)}$$

Inverse measurement model:
what is the likelihood of the map
cell being occupied given the current
state and current measurement?

Inverse sensor measurement model

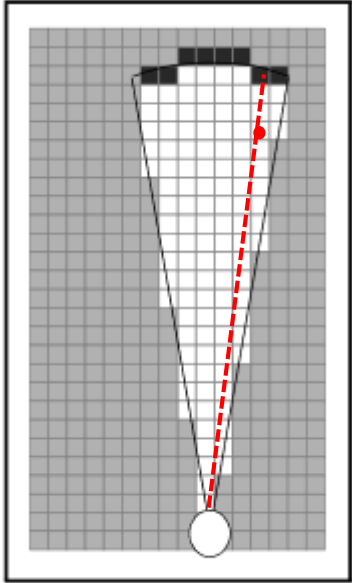


Given map cell (i, j) the robot's state $\mathbf{x} = (x, y, \theta)$, and beams $\mathbf{z} = \{(r_k, \psi_k)\}$

Find index k of sensor beam that is closest in heading to the cell (i, j)

$$p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)$$

Inverse sensor measurement model



Given map cell (i, j) the robot's state $\mathbf{x} = (x, y, \theta)$, and beams $\mathbf{z} = \{(r_k, \psi_k)\}$

Find index k of sensor beam that is closest in heading to the cell (i, j)

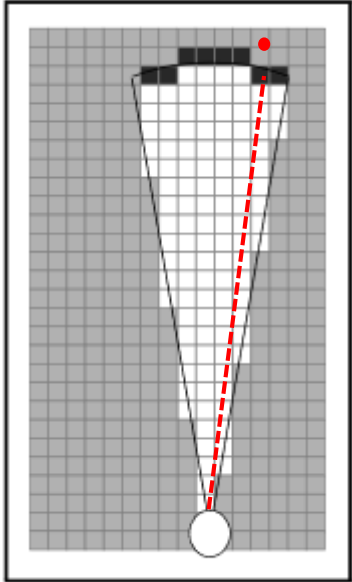
$$p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)$$

If the cell (i, j) is sufficiently closer than r_k

// Cell is most likely free

Return p_{occupied} that is well below 0.5

Inverse sensor measurement model



Given map cell (i, j) the robot's state $\mathbf{x} = (x, y, \theta)$, and beams $\mathbf{z} = \{(r_k, \psi_k)\}$

Find index k of sensor beam that is closest in heading to the cell (i, j)

If the cell (i, j) is sufficiently farther than r_k or out of the field of view

// We don't have enough information to decide whether cell is occupied

Return prior occupation probability $p(m_{ij} = 1)$

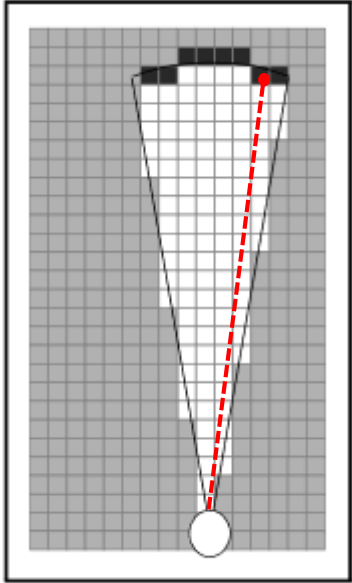
$$p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)$$

If the cell (i, j) is sufficiently closer than r_k

// Cell is most likely free

Return p_{occupied} that is well below 0.5

Inverse sensor measurement model



$$p(m_{ij} = 1 | \mathbf{z}_t, \mathbf{x}_t)$$

Given map cell (i, j) the robot's state $\mathbf{x} = (x, y, \theta)$, and beams $\mathbf{z} = \{(r_k, \psi_k)\}$

Find index k of sensor beam that is closest in heading to the cell (i, j)

If the cell (i, j) is sufficiently farther than r_k or out of the field of view

// We don't have enough information to decide whether cell is occupied

Return prior occupation probability $p(m_{ij} = 1)$

If the cell (i, j) is nearly as far as the measurement r_k

// Cell is most likely occupied

Return p_{occupied} that is well above 0.5

If the cell (i, j) is sufficiently closer than r_k

// Cell is most likely free

Return p_{occupied} that is well below 0.5

inverse_sensor_measurement_model((i, j) $\mathbf{x} = (x, y, \theta)$ $\mathbf{z} = \{(r_k, \psi_k)\}$)

From Probabilistic Robotics, chapter 9.2

- Let (x_i, y_i) be the center of the cell (i, j)
- Let $r = \|(x_i, y_i) - (x, y)\|$
- Let $\phi = \text{atan2}(y_i - y, x_i - x) - \theta$ // Might need to ensure this angle difference is in $[-\pi, \pi]$
- The index of the closest-in-heading beam to (x_i, y_i) is $k^* = \underset{k}{\operatorname{argmin}} |\phi - \psi_k|$
- If $r > \min\{r_{\max}, r_k + \alpha/2\}$ or $|\phi - \psi_k| > \beta/2$
 - Return the log odds ratio of the prior occupancy $\log \frac{p(m_{ij} = 1)}{p(m_{ij} = 0)}$
- If $r_k < r_{\max}$ and $|r - r_k| < \alpha/2$
 - Return the log odds ratio of being occupied (corresponding to occupation probability > 0.5)
- If $r \leq r_k$
 - Return the log odds ratio of being free (corresponding to occupation probability < 0.5)

Recap

- We wanted to compute the likelihood of any map based on known states and observations

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \simeq \prod_{i,j} p(m_{ij}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

Recap

- We wanted to compute the likelihood of any map based on known path and observations

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \simeq \prod_{i,j} p(m_{ij}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- To evaluate $p(m_{ij} = 1|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \text{belief}_t(m_{ij} = 1)$ we had to apply Bayes' theorem, which revealed a way to recursively update the belief

$$\text{belief}_t(m_{ij} = 1) = \eta p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1) \text{belief}_{t-1}(m_{ij} = 1)$$

Very frequent
reasoning in
probabilistic
robotics

Recap

- We wanted to compute the likelihood of any map based on known path and observations

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \simeq \prod_{i,j} p(m_{ij}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- To evaluate $p(m_{ij} = 1|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \text{belief}_t(m_{ij} = 1)$ we had to apply Bayes' theorem, which revealed a way to recursively update the belief

$$\text{belief}_t(m_{ij} = 1) = \eta p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1) \text{belief}_{t-1}(m_{ij} = 1)$$

- To avoid evaluating η we used the log odds ratio

$$l_t^{(ij)} = \log \frac{p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 0)} + l_{t-1}^{(ij)}$$

Can do this for binary
random variables

Recap

- We wanted to compute the likelihood of any map based on known path and observations

$$p(\mathbf{m}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \simeq \prod_{i,j} p(m_{ij}|\mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- To evaluate $p(m_{ij} = 1|\mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \text{belief}_t(m_{ij} = 1)$ we had to apply Bayes' theorem, which revealed a way to recursively update the belief

$$\text{belief}_t(m_{ij} = 1) = \eta p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1) \text{belief}_{t-1}(m_{ij} = 1)$$

- To avoid evaluating η we used the log odds ratio

$$l_t^{(ij)} = \log \frac{p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1)}{p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 0)} + l_{t-1}^{(ij)}$$

- Computing the forward measurement model $p(\mathbf{z}_t|\mathbf{x}_t, m_{ij} = 1)$ was hard, so we applied Bayes' rule again, to get an inverse measurement model $p(m_{ij} = 1|\mathbf{z}_t, \mathbf{x}_t)$ and an easier-to-compute log-odds ratio:

$$l_t^{(ij)} = l_{t-1}^{(ij)} + \log \frac{p(m_{ij} = 1|\mathbf{z}_t, \mathbf{x}_t)}{p(m_{ij} = 0|\mathbf{z}_t, \mathbf{x}_t)} - \log \frac{p(m_{ij} = 1)}{p(m_{ij} = 0)}$$

Occupancy Grid Algorithm

- Upon reception of a new laser/sonar/scan measurement $\mathbf{z}_t = \{(r_k, \psi_k)\}$
- Let the robot's current state be $\mathbf{x}_t = (x_t, y_t, \theta_t)$
- Let the previous log-odds ratio of the occupancy belief be the 2D array $l_{t-1}^{(ij)}$ where i is a row, j is a column
In the beginning we set the prior $l_0^{(ij)} = \log \frac{p(m_{ij} = 1)}{p(m_{ij} = 0)}$ where the occupancy probability is a design decision.

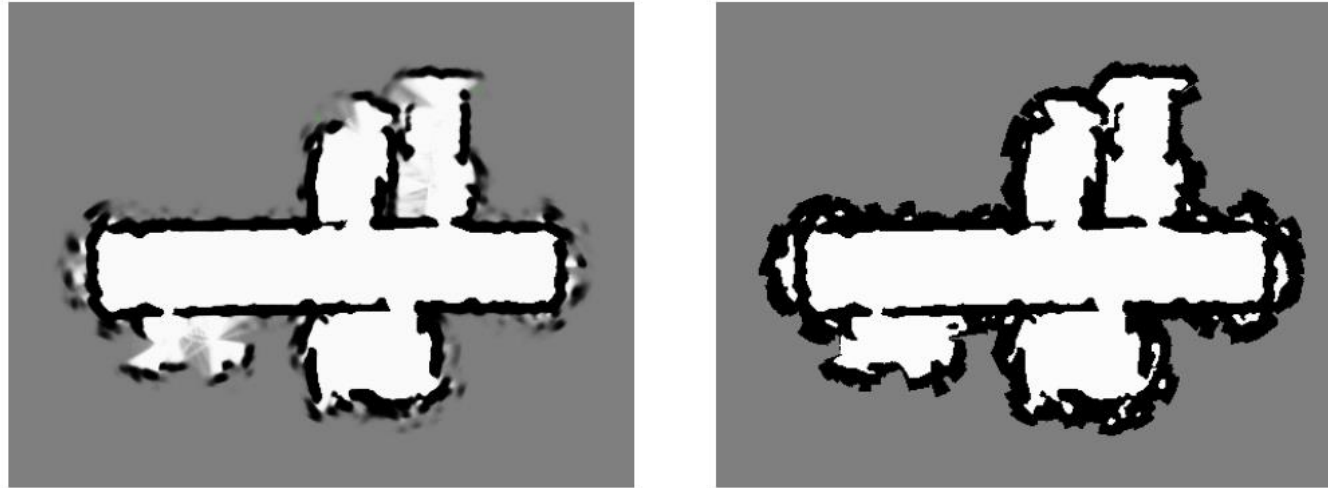
Occupancy Grid Algorithm

- Upon reception of a new laser/sonar/scan measurement $\mathbf{z}_t = \{(r_k, \psi_k)\}$
- Let the robot's current state be $\mathbf{x}_t = (x_t, y_t, \theta_t)$
- Let the previous log-odds ratio of the occupancy belief be the 2D array $l_{t-1}^{(ij)}$ where i is a row, j is a column
 In the beginning we set the prior $l_0^{(ij)} = \log \frac{p(m_{ij} = 1)}{p(m_{ij} = 0)}$ where the occupancy probability is a design decision.
- For all cells (i,j) in the grid
 - If the cell (i,j) is in the field of view of the robot's sensor at state \mathbf{x}_t

$$l_t^{(ij)} = l_{t-1}^{(ij)} + \text{inverse-sensor-measurement-model}((i, j), \mathbf{x}_t, \mathbf{z}_t) - l_0^{(ij)}$$
 - Else

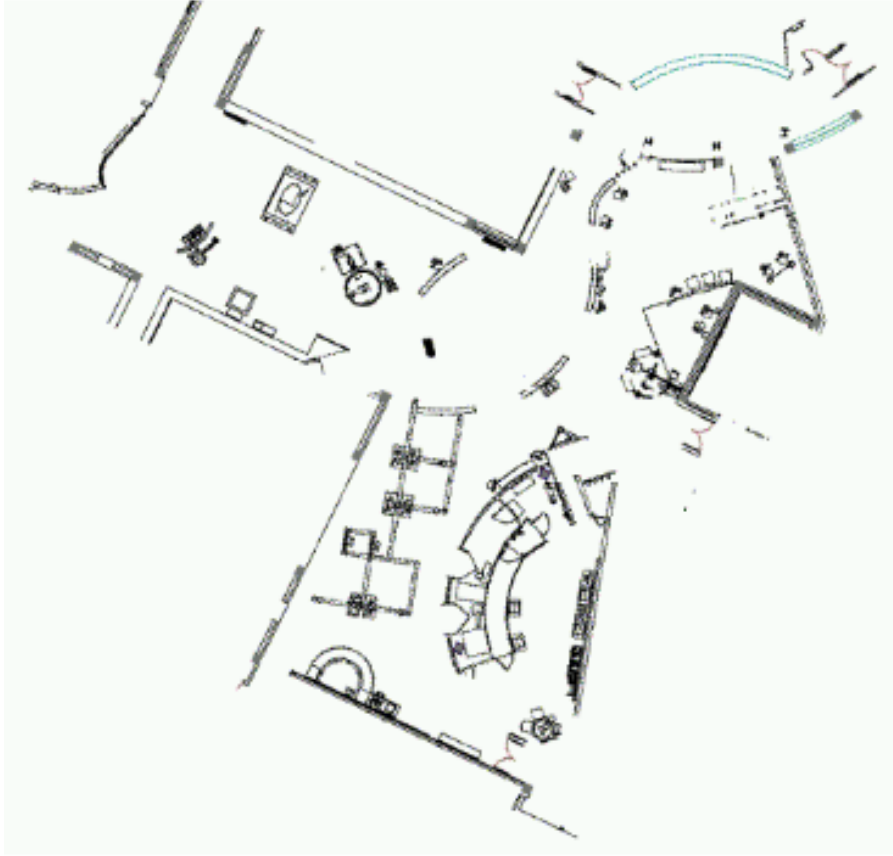
$$l_t^{(ij)} = l_{t-1}^{(ij)}$$
- If asked, return the following 2D matrix of occupancy probabilities: $belief_t(m_{ij} = 1) = 1 - \frac{1}{1 + \exp(l_t^{(ij)})}$

Results



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

Tech Museum, San Jose



CAD map



occupancy grid map