Robotics and Autonomous Systems

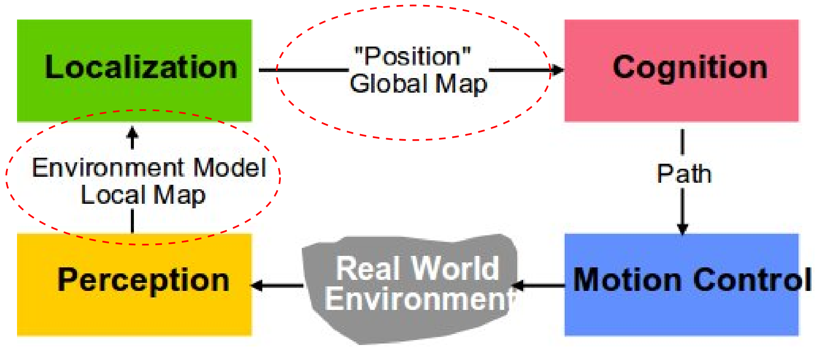
Lecture 8: Maps and mapping

Richard Williams

Department of Computer Science
University of Liverpool

UNIVERSITY OF LIVERPOOL

Today



- Since you need it for the assignment, we'll talk about mapping.

Features

- In the first class we said that navigation begins with what the robot can "see".

There are several forms this might take, but it will depend on:

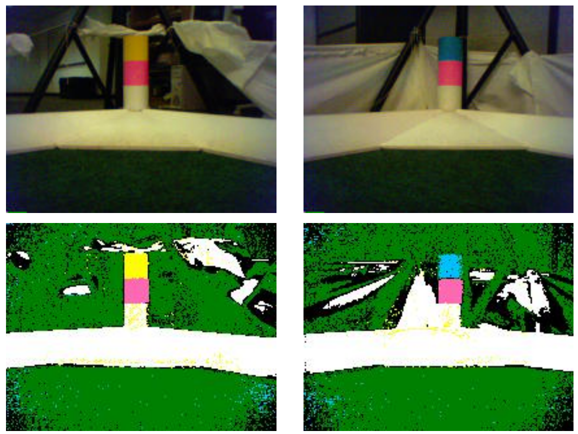
What sensors the robot has

What features can be extracted.

- This is not a particularly likely set of features.

Features

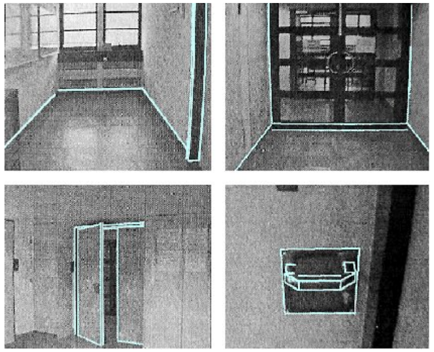
- More likely features are things that can be extracted from images:



- Simple color segmentation.
  (UT Austin RoboCup team)

Features

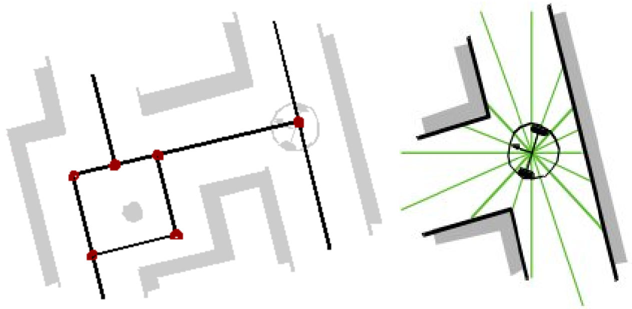
- The results of more complex image processing:



- Edge detection, template matching.
  (Lanser et al (1996))

Features

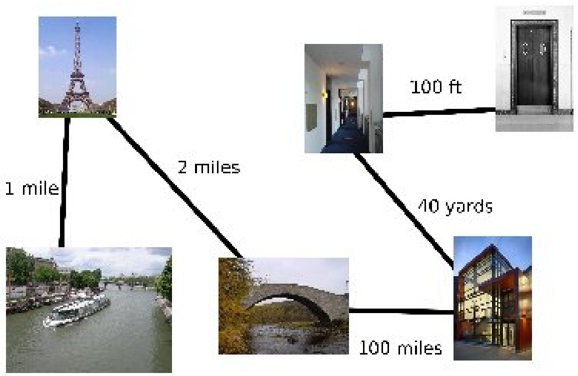
- One can also identify features with other kinds of sensor.



- Patterns of range finder readings that are identifiable.
  - Meaning we can tell when the sensor spots them.

Map

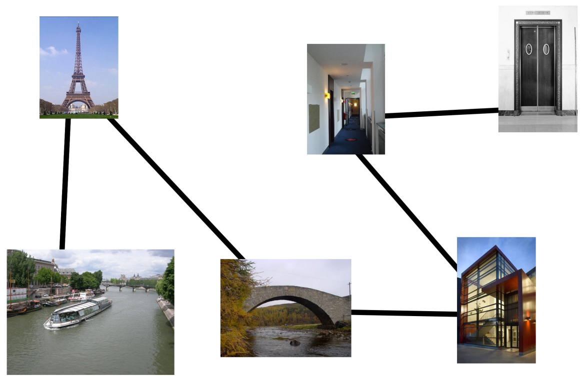
- Once we have a set of features we can build a map.



- A map says how features sit relative to one another.

Types of map

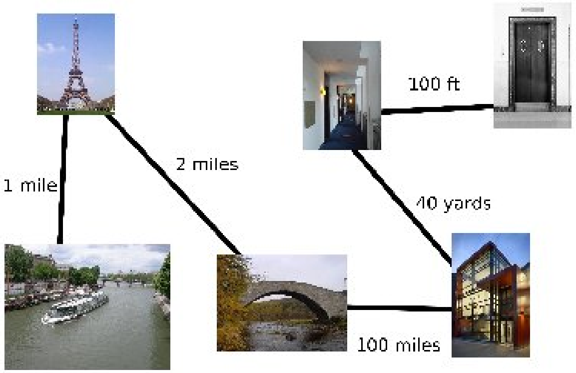
- Topological map



- Just says what the relationship between features is.

Types of map

- Metric topological map



- Provides some information on distances between features.

Types of map

- Metric map

Latitude: 48º 51' 32" North
Longitude: 002º 17' 45" East

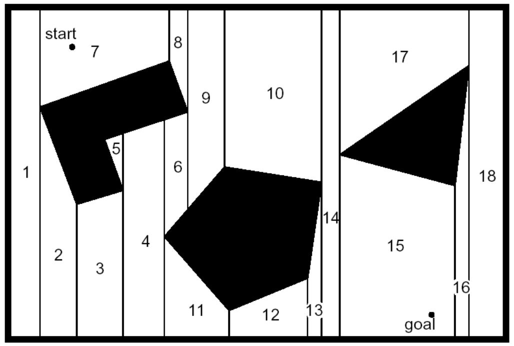
- Gives the precise location of the features.
  - In whatever coordinate system is most appropriate.
  - Frequently as a pose $\langle x, y, \theta \rangle$
- Continuous or discrete measurements.

Cell-based maps

- A common way to create a map is to break up the map into a series of cells.
- A number of ways one might do this.
- As ever, there are trade-offs.
  - Different approaches are better or worse depending on what you are trying to do.

Cell-based maps

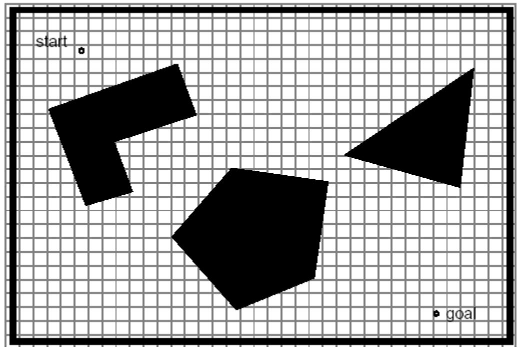
- Exact cell decomposition.



- Split up the space based on features of the objects in the space.
- Naturally lends itself to a topological map.

Cell-based maps

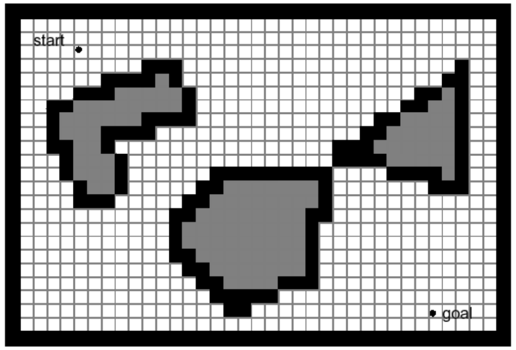
- Fixed cell decomposition.



- The "graph paper" approach.
- Can get a nasty interaction between object boundaries and cell size.

Cell-based maps

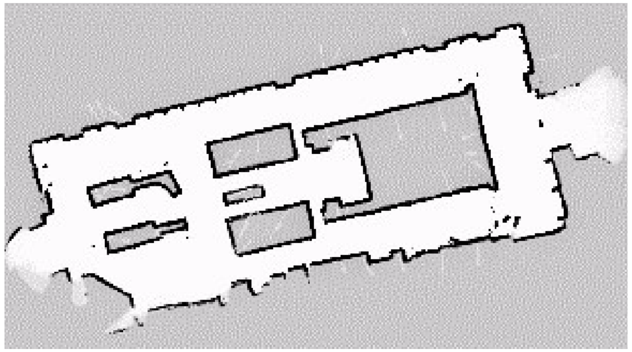
- As here:



- If we are looking for free squares to traverse, they tend to disappear in narrow spaces.
- What can we do?

Cell-based maps

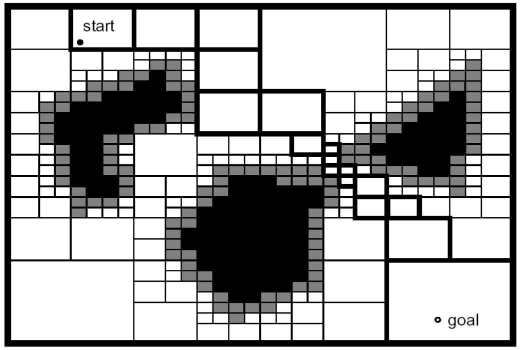
- A grid-based map with a small cell size:



- This creates its own problems.
  - Like?

Cell-based maps

- Another approach is to use an adaptive cell decomposition.



- Large cells where there is free space, smaller cells near the obstacles.

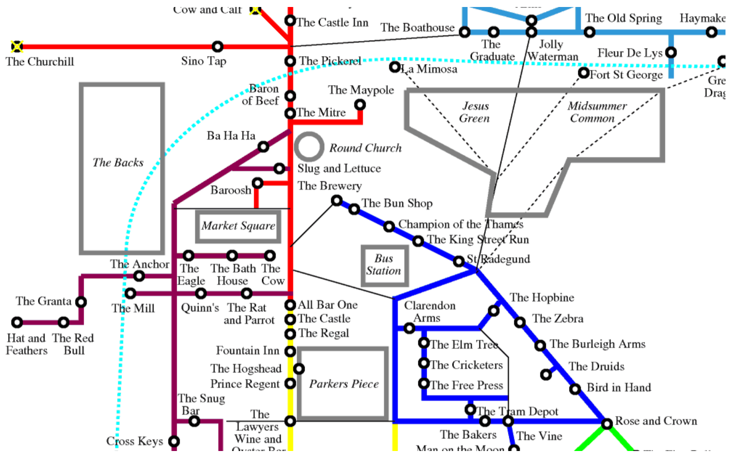
Topological maps

- Topological maps just concentrate on connections.

- London underground map is a classic topological map
  - Exact positions of stations are misleading

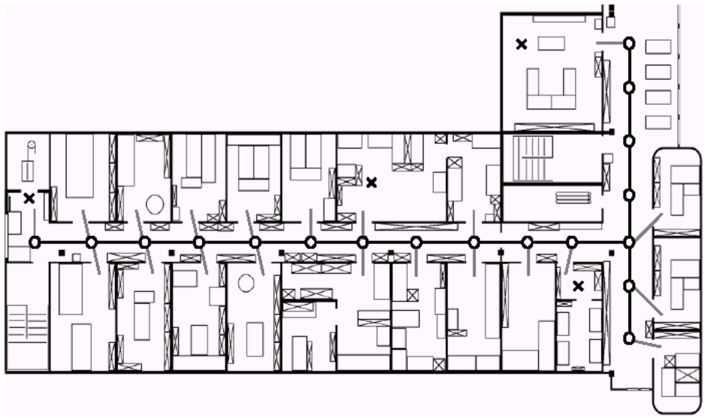
Topological maps

- Pubs in Cambridge, UK:



Topological maps

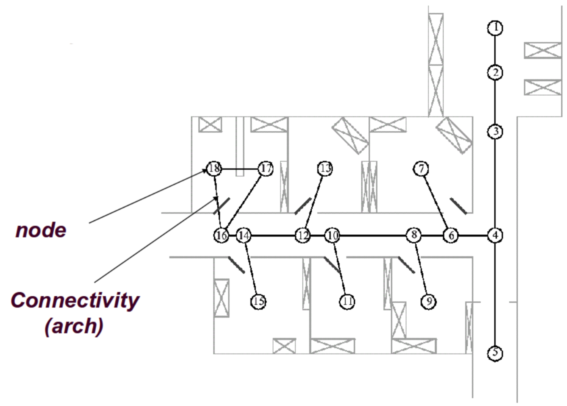
- How are topological maps useful?



- Sometimes it is enough to know which direction to head.

Topological maps

- Topological decomposition



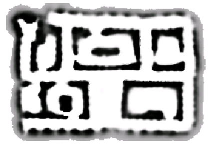
Occupancy grids

- A common form of fixed cell map is an occupancy grid (Elfes 1987)

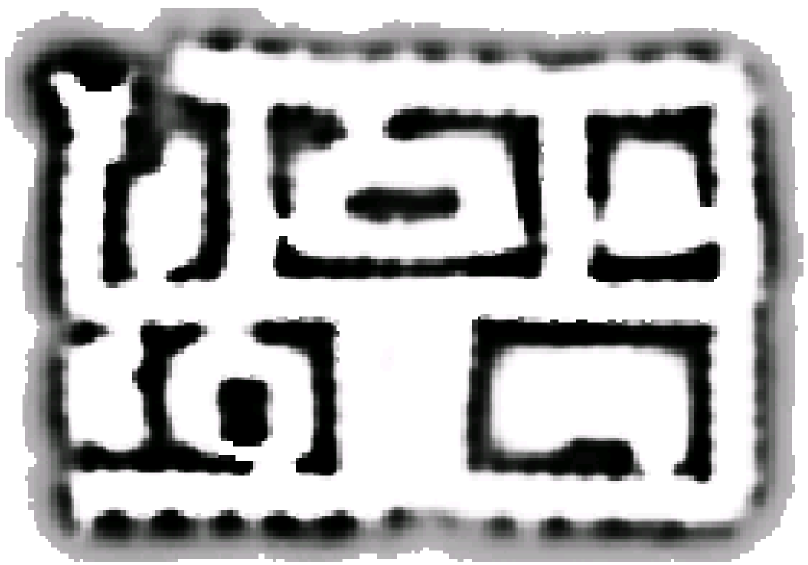
- Each square in the grid is “marked” as either containing an obstacle or being free space.
- Occupancy grids are (relatively) easy to construct as the robot moves around.

Occupancy grids

- A common form of fixed cell map is an occupancy grid (Elfes 1987)



- Each square in the grid is “marked” as either containing an obstacle or being free space.
- Occupancy grids are (relatively) easy to construct as the robot moves around.
- That is why we will use them for the assignment

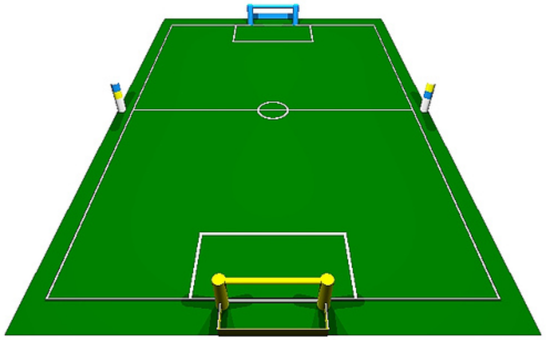


From maps to localization

- The reason we want maps is to be able to have our robots navigate.
- So a key question is what we need to do in addition to just having the map in order to localize.
- To some extent this depends on what kind of map we have.
  - In turn that depends on what kind of features we are extracting.
- It is all connected!

Towards localization

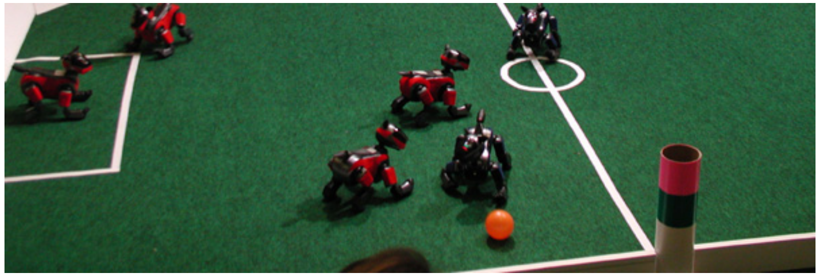
- Maps with beacons and individual features.



- If we spot a unique beacon, and we know the distance and bearing to it, that is all we need to apply a sensor model.
- Each observation is quite informative.

Towards localization

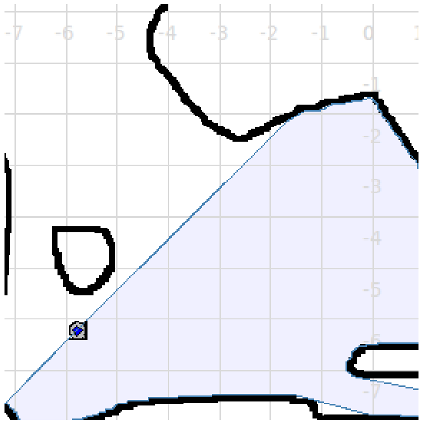
- All the robots in the picture below will be able to position themselves on the arc of a circle around the beacon on the bottom right.



- Seeing another object will allow them to localize (modulo errors in their observations).
- The same can be true for other visual data and some features with clear sensor signatures.

Towards localization

- Data from sonar, laser and other range data are less helpful.
- All a reading typically tells us is that the robot is some distance from an obstacle.



- Could match multiple locations.

Towards localization

- To use scan data to localize we have to:
  - Work out what the ranger would measure at every location in the map.
  - Compare this with what the scanner reports.
  - Apply the sensor model
- Each scan typically gives less information than feature recognition
  - More locations match
- Range scanners work because they generate so much data — tens to hundreds of points a second rather than a measurement every few seconds for vision data.

Map building

- Clearly it is possible for people to build maps.
  - Maps can be constructed from floor plans.
  - Or just by motivated people with rulers and compasses or other angle-measuring devices.
- But that is not so interesting as having robots build maps for themselves.
- One of the reasons for the popularity of occupancy grid maps is that it is easy to get started:
  - Have the robot wander around and measure where its sensors say there are cells that are occupied.

Building an occupancy grid map

- Fixed cell map.
- Each cell has a number associated with it.
  - Probability of occupancy
- 0 indicates the cell is not occupied.
- 1 indicates that it is occupied.
- Doing the precise probability estimate is somewhat difficult, but there is a simple and robust approximate way to build the map.

Building an occupancy grid map

- In essence you drive around, keeping track of where you are, and looking into each cell.
- You keep count of how many time your "see" an obstacle in a cell with your sensors.
- You increase the count for the cell each time you detect an object in it.
- You decrease the count for the cell each time you detect that it is clear.
- If you track how many times you have "seen" a cell in total, you can estimate the probability that the cell is occupied.

Making that a bit more formal

- Start with an array:

$$M = \begin{bmatrix} M_{0,0} & M_{1,0} & \dots & M_{m,0} \\ M_{0,1} & M_{1,1} & \dots & M_{m,1} \\ \vdots & \vdots & \vdots & \vdots \\ M_{0,n} & M_{1,n} & \dots & M_{m,n} \end{bmatrix}$$

  where m and n are the maximum x and y indices of the occupancy grid.
- This is a model of the occupancy grid
- Which itself is a model of the world.

Making that a bit more formal

- Set every element $M_{x,y}$ of M to 0.
- Create a similar array:

$$C = \begin{bmatrix} C_{0,0} & C_{1,0} & \dots & C_{m,0} \\ C_{0,1} & C_{1,1} & \dots & C_{m,1} \\ \vdots & \vdots & \vdots & \vdots \\ C_{0,n} & C_{1,n} & \dots & C_{m,n} \end{bmatrix}$$

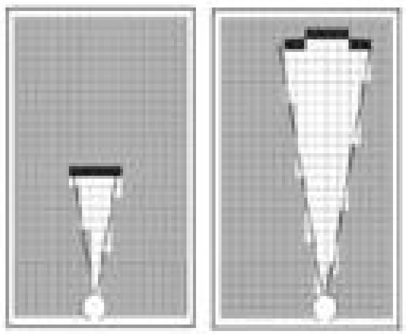
  and set each of its elements to 0 also.
- (You could also create an array of objects each of which held an M value and a C value.)

Making that a bit more formal

- As the robot moves, it figures out which cells are observed by the sensors. and updates the relevant matrix cells.
- For each cell (x, y) in the sensor field:
  - Set $C_{x,y}$ to $C_{x,y} + 1$.
  - If the cell is occupied, set $M_{x,y}$ to $M_{x,y} + 1$.
  - If the cell is unoccupied, set $M_{x,y}$ to $M_{x,y} - 1$.
- Repeat as often as possible.

Sensor model

- To establish which cells are observed, we apply a sensor model:



- Says which cells the sensor "sees" relative to the robot.

  (Sensor model from Thrun, Fox and Burgard)

Building an occupancy grid

- If $C_{x,y} = 0$, we don't know anything about a cell
  - And should probably find out.
- If $C_{x,y} > 0$, we know something.
- The larger $C_{x,y}$, the more sure we are that what we know is correct.

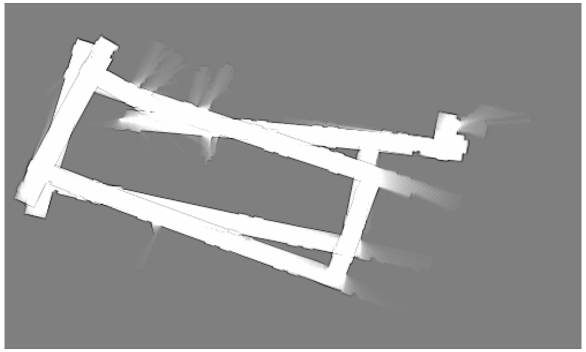
Building an occupancy grid

- For cells that have $C_{x,y} > 0$.
- If $M_{x,y} > 0$ we can consider that the cell is more likely than not to be occupied
- If $M_{x,y} < 0$ we can consider that the cell is more likely than not to be unoccupied
- We can also compute a probability of the cell being occupied:

$$\Pr(O_{x,y}) = \frac{M_{x,y} + C_{x,y}}{2C_{x,y}}$$

A preview of the problem you'll have

- Can't localize without a map.
  - Have to rely on odometry

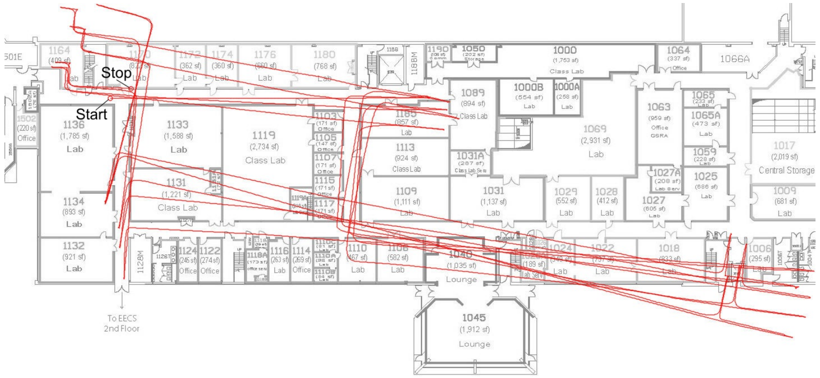


- Odometry sucks

(image from Andrew Howard's pmap site).

A preview of the problem you'll have

- Seconded by:



(Johann Borenstein).

However

- You can add beacons that give you approximate location.

  . . . and that can be enough, especially at the coarse scale you'll be using.
- That is the reason for the colored paper etc.
- (We'll talk about this issue more later.)

Beacons

Summary

- Maps are essential if we are going to have robots navigate.
- We discussed some of the different kinds of map
  - There are others
- We looked at how they might be used.
- And we looked at how our kind of map might be built autonomously.