G54ARS
Autonomous Robotic Systems
Lecture 5

Localisation and Mapping

Material adapted from:

Autonomous Mobile Robots, Ch. 5,
Siegwart & Nourbakhsh
Figs. 5.3, 5.4, 5.5, 5.6, 5.9, 5.11, 5.13, 5.14, 5.15, 5.18 © Siegwart & Nourbakhsh
Fig. 5.11 Courtesy of W. Burgard

Introduction to AI Robotics, Ch. 11,
Robin R. Murphy
Figs. 11.2, 11.4, 11.9 © Robin R. Murphy

University of Nottingham
UK | CHINA | MALAYSIA

Postgraduate Study Roadshow

Win a £1,000 bursary for a masters or PhD

When: Wednesday 8 November 2017
11.30am-3pm

Where: The Great Hall, Trent Building
University Park Campus

Register now at www.postgraduateroadshow.com

RUSSELL GROUP UNIVERSITIES
POSTGRADUATE STUDY
ROADSHOW

Last week…

- Fuzzy Logic Control
  - Fuzzy Logic Control Principles
  - Origins and History
  - Fuzzy Sets and FLC components
  - FLC control examples
  - FLC design and tuning
  - FLC implementation notes

Kahoot!
https://kahoot.it/

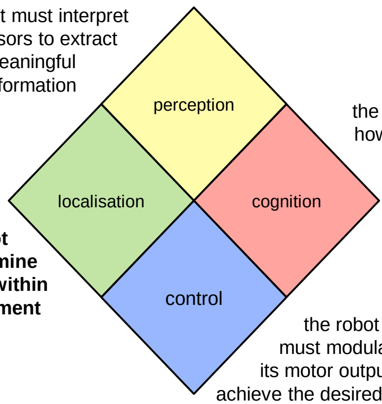
This week…

- Localisation issues
  - general scheme
  - sources of uncertainty
    - sensors
    - actuators
- Alternatives to localisation
  - behaviour-based navigation
- Sensor models
  - Bayesian Sonar Sensor Model
  - HIMM
- Representation
  - belief representation
  - map representation

Localisation Issues

Autonomous Mobile Robots - Navigation

the robot must interpret its sensors to extract meaningful information

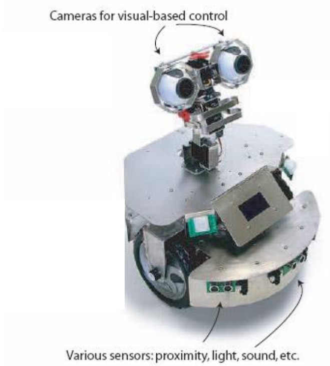


the robot must decide how to act to achieve its goals

the robot must determine its position within the environment

the robot must modulate its motor outputs to achieve the desired trajectory

The Challenge of Localisation

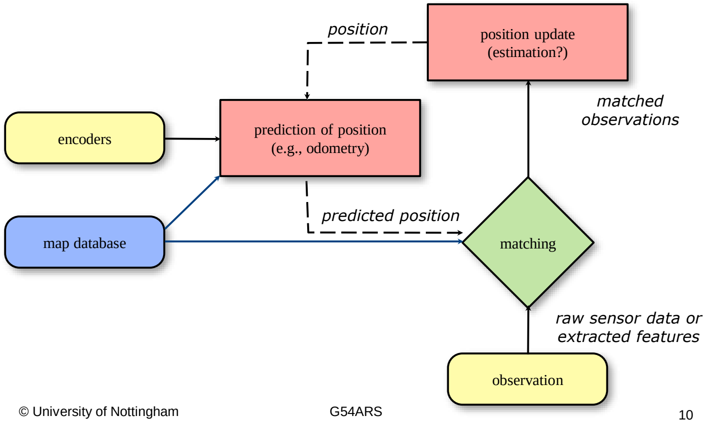


Gordon Wyeth
University of Queensland

The Ultimate?

- Even if we had an extremely precise, completely accurate 'GPS' type positioning system
  - would still not provide local positioning information
    - how far away is the wall?
    - what is the position of the target?
    - where is the blue door?
  - GPS problems
- Even in abstract terms, in order to be useful, accurate GPS+ would be needed for *every object*
  - precision/accuracy required is task/robot dependent
  - complete accuracy is *fundamentally impossible!*
    - *Heisenberg's Uncertainty Principle*

General Schematic for Localisation



Sensor Noise

- Sensor noise induces a limitation on the consistency of sensor readings obtained from the given environmental state
  - a limit on the number of useful bits of information
- Often, the source of sensor noise is that some environmental features are not captured by the robot's representation
  - CCD colour camera
    - hue values obtained are dependent on lighting conditions (e.g. sun, clouds, etc.)
  - sonar range-measurements
    - are dependent on angle of incidence / surface texture

Sensor Aliasing

- The same information is returned by a sensor for two different local states
  - rarely encountered by human visual system
    - every place looks different
    - most robotic sensors are far less rich
  - however, aliasing can occur for humans
    - remove / significantly reduce visuals (e.g., dark cave)
    - in some special circumstances / unusual environments (e.g., mazes in stately homes / theme parks)
- For robotic sensors, aliasing is the norm
  - sonars only provide distance
    - rotate 90 degrees in a square environment
    - cannot differentiate between a wall and a human, etc.

Effector Noise

- Also a fundamental physical limitation
  - a single action has many different possible results
    - even if the initial state is completely known
  - uncertainty is introduced into future state (e.g., robot thinks it has moved distance x but has actually moved distance y)

- Prediction of future state, based on knowledge of current state and an accurate system model, can be effectively combined with sensor information to *reduce uncertainty*
  - incomplete model of the system
    - e.g., lack of modelling of individual motor characteristics
  - incomplete model of the environment
    - e.g., lack of modelling of wheel slip

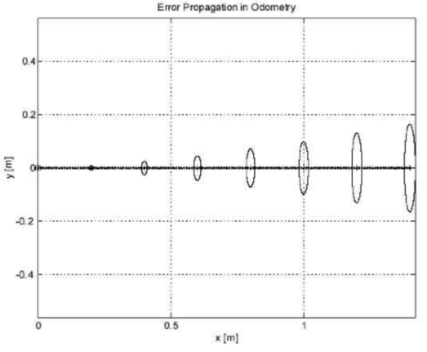
Odometry and Dead Reckoning

- Odometry: use of wheel sensors
- Dead reckoning: also using heading sensors (e.g., gyroscope)
- Robot movement is integrated over time, thus the position error accumulates over time
- Sources of error include:
  - limited resolution during integration
  - misalignment of wheels
  - uncertainty in wheel diameters (particularly unequal)
  - variation in contact point of the wheel (tyre deformation)
  - unequal floor contact (slipping, rough surface, etc.)
- Errors may be
  - deterministic
    - may be eliminated by calibration
  - non-deterministic
    - random errors which remain

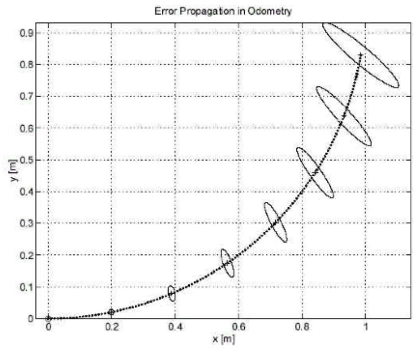
Classification of Errors

- Errors may be classified into three types
  - range error
    - integrated path length (distance) of robot's movement
      - sum of wheel motions
  - turn error
    - similar to range error, but for turns
      - difference of wheel motions
  - drift error
    - difference in the error of the wheels leads to an error in the robot's angular orientation
- Over time, turn & drift far outweigh range errors
  - small error in angle is amplified over time

Pose Uncertainty Over Time

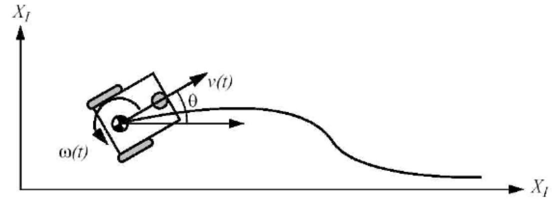


Pose Uncertainty with Turn

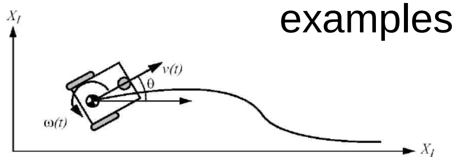


Differential Drive Robot

$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad p' = p + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \theta \end{bmatrix}$$



Differential Drive robot – worked examples



$$\Delta\theta = \frac{\Delta s_r - \Delta s_l}{b}$$
$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$

where Δs_l and Δs_r are the distances travelled for the left and right wheel respectively and b is the distance between the two wheels of the differential-drive robot

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

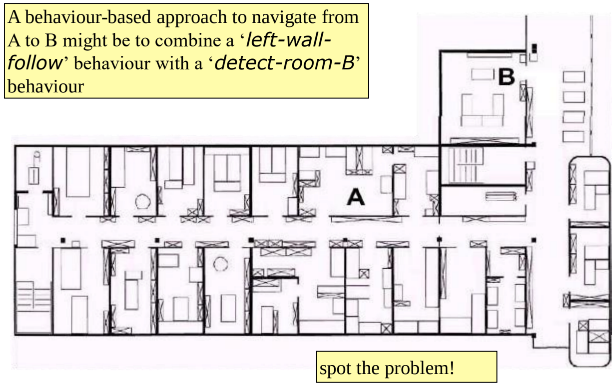
Examples

See Worksheet

Alternatives to Localisation

Localise or Not? Is Localisation a must?

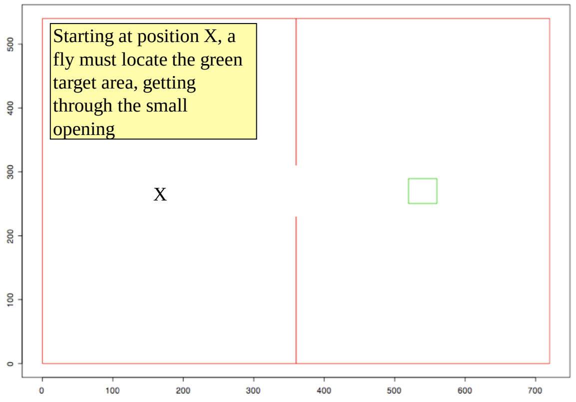
A behaviour-based approach to navigate from A to B might be to combine a '*left-wall-follow*' behaviour with a '*detect-room-B*' behaviour



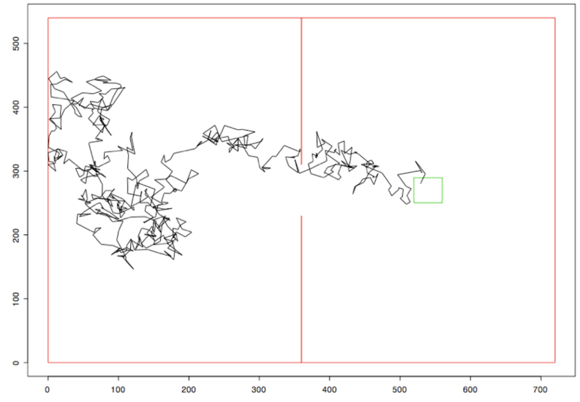
spot the problem!

Simple Behaviours

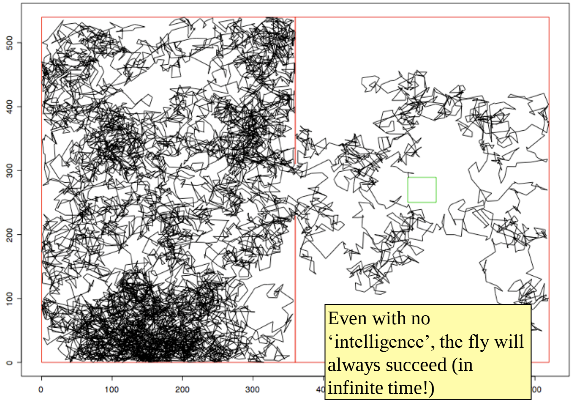
Starting at position X, a fly must locate the green target area, getting through the small opening



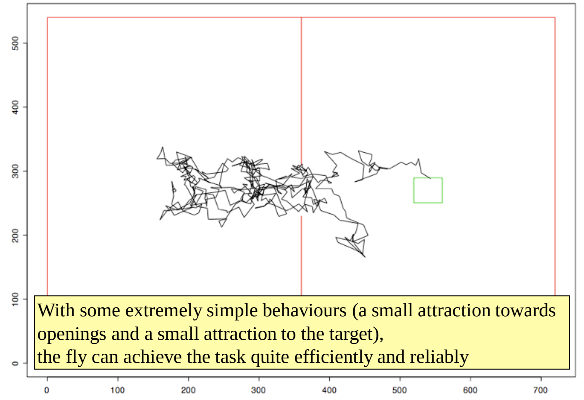
A Good Solution



A Bad Solution



Simple Effective Behaviour



So – to localise or not?

- Problem dependent question.
- For some problems behaviour based architectures excel and allow achieving robust solutions using a very simple set of behaviours.
- HOWEVER, behaviours **alone** can often make it very hard to solve specific problems, e.g., in mobile robotics, a robot frequently needs to reason about:
  - Where am I?
  - Have I been here before?
  - How best to get there?
  - Etc.

- To answer these questions localisation and potentially from there – mapping is required.
- Two main categories of localisation are commonly distinguished:
  - Iconic (using occupancy-grids)
  - Feature-based (e.g., for topological maps)

A practical case – Sonar Sensor Models

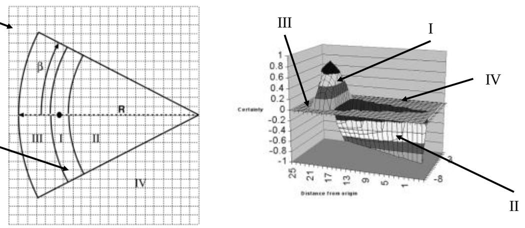
- Sonar sensors are common for map making and obstacle avoidance
- However, (sonar) sensor data is subject to uncertainty (e.g., sensor noise)

- What to do?
  - **Empirical modelling**: testing the sensor and using the frequency of correct readings to build up a belief model
  - **Analytical model**: generating the model directly from known properties of the sensor (e.g., from the manufacturer)

Sonar Sensor Model

grid elements

Note: the thickness of Region I arises from the resolution or *tolerance* of the sonar (e.g., + or – 2 cm)



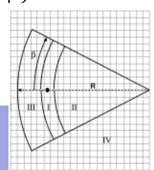
β : the field of view of the sonar sensor (half-angle)
R : the maximum range of the sonar
Region I : where the grid elements are probably occupied
Region II: where the grid elements are probably empty
Region III: where it is not known if the elements are occupied or not
Region IV: outside current field of view

A Bayesian Approach – Recap.

- A probability function scores evidence towards a hypothesis $\boldsymbol{H}$ as a number between 0 and 1.
- An example of $\boldsymbol{H}$ is the event whether a given grid element, grid[i][j] is occupied or empty, i.e.:
  $$\boldsymbol{H} = \{H, \neg H\} \ or \ \boldsymbol{H} = \{Occupied, Empty\}$$
- The probability of H is captured by:
  $$0 \leq \mathrm{P}(H) \leq 1$$
- Remember:
  $$1 - \mathrm{P}(H) = P(\neg H)$$
- Unconditional Probabilities: known in advance, e.g., in a structured environment we may have: $\mathrm{P}(H = Occupied) = 0.75$.
- Conditional Probabilities: the probability of a hypothesis given some evidence such as a sensor reading s. For example:
  $P(H|s)$ is the probability that H has occurred <u>given</u> s.
- Remember:
  $$P(H|s) + P(\neg H|s) = 1$$

Updating Occupancy Grids

- For each grid element, we capture the probabilities $P(Occupied|s)$ and $P(Empty|s)$.
- To update the probabilities for a given element/cell, we use a set of functions dependent on the position of the element in relation to the sonar, e.g.:



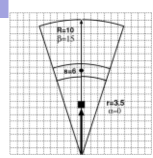
The closer to the origin of the sonar, the higher the belief

The closer to the acoustic axis, the higher the belief

A reading is never fully believable, e.g., $Max_{occupied} = 0.98$

- **Region I**:

$$P(Occupied) = \frac{\frac{R-r}{R} + \frac{\beta - \alpha}{\beta}}{2} \times Max_{occupied}$$
$$P(Empty) = 1.0 - P(Occupied)$$

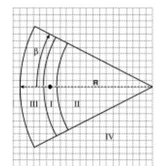


Note:
r : the distance from the sonar to the grid element/cell
α : the angle to the grid element

Updating ctd.

- **Region II:**

$$P(Occupied) = 1.0 - P(Empty)$$
$$P(Empty) = \frac{\frac{R-r}{R} + \frac{\beta - \alpha}{\beta}}{2}$$



Note: in Region II, a grid cell can have the probability $P(Empty)$=1, while in Region I, $P(Occupied) < 1$.

- **Example:**

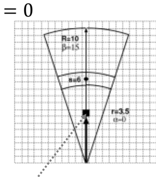
The sonar returns a reading of 6 feet, with a tolerance of ± 0.5.
$$R = 10, \quad \beta = 15, \quad Max_{occupied} = 0.98, \quad r = 3.5, \quad \alpha = 0$$

Note: *r : distance between origin and target cell,*
α : arctangent between origin and target cell

→ *Step 1 - Determine region:* $3.5 < (6.0 - 0.5)$, *thus: Regions II*

→ $P(Empty) = \frac{\frac{R-r}{R} + \frac{\beta-\alpha}{\beta}}{2} = \frac{\frac{10-3.5}{10} + \frac{15-0}{15}}{2} = \frac{\frac{6.5}{10} + \frac{15}{15}}{2} = 0.825$

→ $P(Occupied) = 1.0 - P(Empty) = 1 - 0.825 = 0.175$



Target cell at [i][j]

Updating ctd.

- **Problem**: The sensor model provides $P(s|H)$, i.e. the probability that the sensor would return the value considered given that the cell is really occupied.
- **BUT**: We would like to know $P(H|s)$, i.e. the probability that a given area of the grid is occupied given a particular sensor reading.
- We can apply Bayes' rule and derive the following:

$$P(Occupied|s) = \frac{\boldsymbol{P(s|Occupied)}P(Occupied)}{\boldsymbol{P(s|Occupied)}P(Occupied) + \boldsymbol{P(s|Empty)}P(Empty)}$$

We know the terms in bold from the sensor model – the others are so-called priors (i.e. prior knowledge). Priors may be known for example in structured environments, otherwise commonly we set $P(Occupied) = P(Empty) = 0.5$.

Thus, for example 1, i.e. regarding the occupancy of grid [i][j], we have:

$$P(Occupied|s=6) = \frac{0.175 \times 0.5}{0.175 \times 0.5 + 0.825 \times 0.5} = \frac{0.0875}{0.0875 + 0.4125} = 0.175$$

$$P(Empty|s=6) = \frac{0825 \times 0.5}{0.825 \times 0.5 + 0.175 \times 0.5} = 1 - 0.175 = 0.825$$

Updating with Bayes' rule

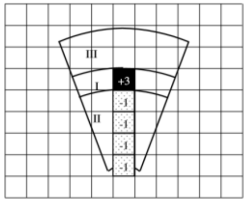
- As new information becomes available over time, we can update the probability at time t_{n-1} with the current observation at time t_n.
- We can derive:

$$P(H|s_n) = \frac{P(s_n|H)P(H|s_{n-1})}{P(s_n|H)P(H|s_{n-1}) + P(s_n|\neg H)P(\neg H|s_{n-1})}$$

- Thus we can update the probabilities $P(Occupied)$ and $P(Empty)$ for each grid cell after each observation/measurement.
- **Conclusion**: Bayesian Sensor model provides excellent capacity to modelling with uncertain information – but, it is computationally expensive.

Bayesian vs HIMM Sensor Model

- The Histogrammic in Motion Mapping (HIMM) sensor model was developed to provide a computationally efficient sensor model.
- Employed to enable reliable obstacle avoidance while travelling at speed.
- Important – the sonar signal is always the same – it is the modelling which changes….



Your tasks:
- Review HIMM (Chapter 11 in Introduction to AI Robotics – paper copy distributed in lecture)
- What are the core features and differences of the HIMM model (in comparison to the Bayesian approach)?
- How do they compare computationally and performance-wise?
- Review worked example (Section 11.6.1)

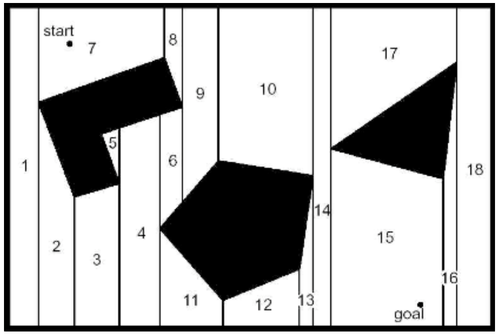
More on Mapping…

Utility of Localisation & Mapping

- Maps provide a basic tool for ordering spatial information (i.e. memory structure for what the robot has encountered and where)
- The map-based concept of the robot's position makes the robot's belief about position visible to the human. (i.e. this is where the robot thinks it is)
- A map provides a medium of communication between human and robot, i.e. humans can give a robot a map and ask it to go to location x.
- Maps can be a useful output when they are automatically created by a robot.

Map Representations

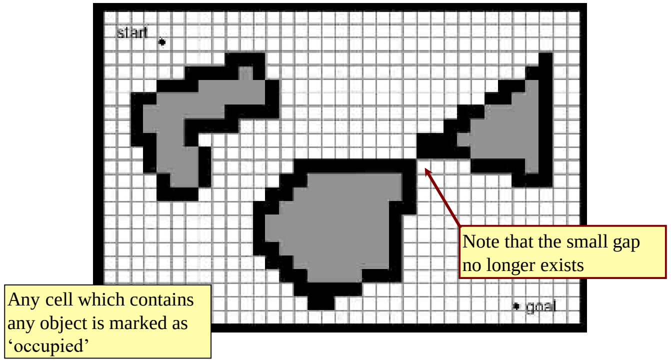
Exact Cell Decomposition



Exact Cell Decomposition

- Space is tessellated into areas of free space
- Each area is represented by a single node
- Assumptions:
  - actual position of a robot within an area is not important
  - What IS important is the potential to traverse from one area of free space to another
- The information required for ECD is not always easily available (e.g., hard to collect by a robot)

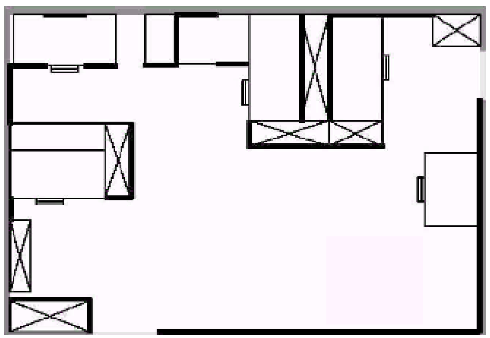
Fixed Cell Occupancy



Fixed Cell Occupancy

- Real world is tessellated, i.e. continuous space is transformed into a discrete representation
- FCO is highly dependent on cell-size (smaller cells – higher precision but higher cost)
- Imposes artificial "grid" on real world
- Small cells risk omitting important detail
- But – efficient and easy to work with

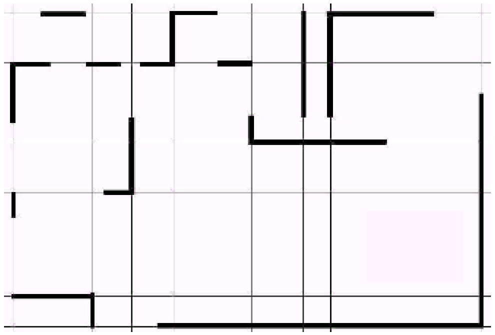
Continuous Polygons



Continuous Polygons

- Generally employed when an environment is simple and well known
- E.g., robot labs and robot scenarios
- Less applicable for mapping of the real world (how can a robot perceive continuous polygons?)

Continuous Line-Based



Continuous Line-Based

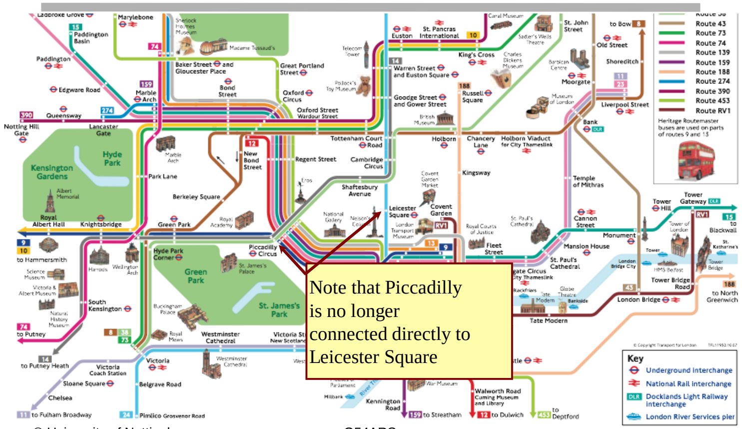
- Abstraction of the real world based on continuous lines
- Can be generated on the fly, for example using laser range finder data (1000s of points are mapped to lines)
- Very good for indoor environments (rooms, corridors, etc.)

Topological Map

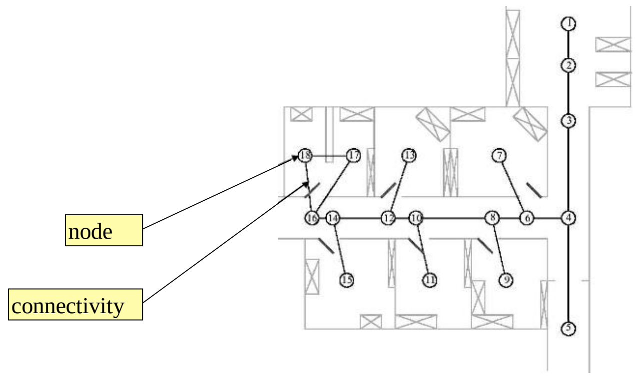
Topological Maps

- Represent nodes and connectivity between nodes
- Topological maps rely on the robot to:
  - employ any (at least one) of its sensors to recognise and "exploit" the connectivity between nodes. (E.g., perceive the corridor to traverse to the other room)
  - be physically capable of "exploiting" the connectivity, e.g., use its wheels to move on a smooth surface
- As topological maps are not necessarily based on the geometry of space (e.g., London tube map), the potential for them to describe the robot's precise position is very limited.

Topological Map



Topological Decomposition



Map Representation

- The method for representing the map is closely linked to the method used for the belief representation (i.e. where is the robot on the map)
  - the precision of the map must appropriately match what the robot needs in order to achieve its goals
  - the precision of the map and the type of features represented must match the precision and data types of the robots sensors
  - the complexity of the map representation has a direct impact on the computational complexity of reasoning about mapping, localisation and navigation

A note on Belief Representation

Belief Representation

- Assuming we want to implement explicit localisation, we must decide on representation
  - belief representation
    - single-hypothesis belief
      - the robot postulates its unique position
      - Kalman filter localization
    - multiple-hypothesis belief
      - the robot describes the degree to which it is uncertain about its position, by postulating possible alternatives
      - particle filter localization
  - map representation
    - continuous representations
    - decomposition strategies

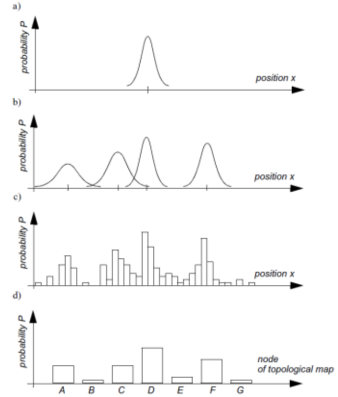
Example Representations

a) continuous map with single hypothesis

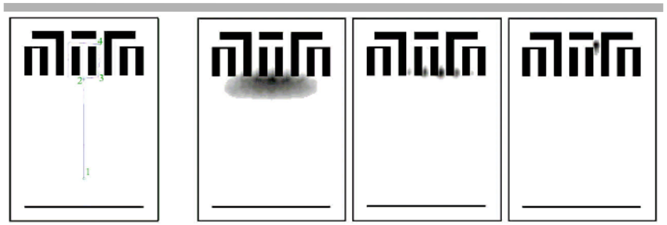
b) continuous map with multiple hypothesis

c) discretised map with probability distribution

d) discretised topologic map with probability distribution



Multiple-Hypothesis Tracking



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

- Multiple-hypothesis approaches can be computationally expensive
  - at position 3, the robot's belief state is distributed across five different passageways
  - need methods to control and eliminate ambiguities

Single vs Multiple Hypothesis

- Single Hypothesis
  - Robot is lost when diverging from hypothesized state
  - typically reasonable in processing power
  - Simple for decision making
- Multiple Hypothesis
  - Robot can maintain and reason about the level of uncertainty of its position
  - Uncertainty resulting from incomplete sensor information can be modelled through the updating of the multiple states
  - Challenge for decision making (e.g., do x when at y – what if I am not sure I am at y?)
  - Potentially computationally expensive

Summary

- Localisation issues
  - general scheme
  - sources of uncertainty
    - sensors
    - actuators
- Alternatives to localisation
  - behaviour-based navigation
- Sensor models
  - Bayesian Sonar Sensor Model
  - HIMM
- Representation
  - belief representation
  - map representation