Robótica Móvel

Mestrado em Robótica e Sistemas Inteligentes

Ano letivo 2023/2024

Nuno Lau, Vitor Santos

The Representation Problem

Representation is the form in which information is stored or encoded in the robot (Mataric)

- Representation is more than memory
- It has a significant impact on robot control

What can the robot represent

- Self
  - Stored proprioception, self-limitations, goals, intentions, plans
- Environment
  - Navigable spaces, structures
- Objects, people, other robots
  - Detectable things in the world
- Actions
  - Outcomes of specific actions in the environment
- Task
  - What needs to be done, where, in what order, how fast, etc.

Navigation challenges

- Path planning problem
  - Robot has a map, knows own and target positions
- Localization problem
  - Robot has a map showing target, doesn’t know own position
- Coverage problem
  - Robot has a map, knows where it is, but doesn’t know where the target is
- Mapping problem
  - Robot does not have a map, may known own position
- Simultaneous localization and mapping
  - Robot does not have a map, and doesn’t know own position

Navigation Questions

- Where am I?
  - Localization
- Where have I been?
  - Mapping
- Where should I be going?
  - Decision
- What is the best way to get there?
  - Path planning
- How do I get there?
  - Path following and Obstacle Avoidance

Different types of representation

- Maze navigator robot
  - Exact path it has taken: “Go straight 2m, turn left 90 deg, go straight…”. This is an **odometric path**
  - Sequence of moves at particular landmarks: “Left at 1st junction, right at 2nd junction, straight…”. This is a **landmark-based path**
  - What to do at each landmark: “At the green/red junction go left, at the red/blue junction go right, …”. This is a **landmark-based map**
  - The map of the maze. This is a **metric map**

Mapping

- Process of building an internal estimate of the map of the environment
  - How does the world look like (for navigation purposes)?
- This is what the robot sees:

Mapping

- To build the map the robot needs to know where it is (Localization)
  - But to know where it is, it needs a map
- Simultaneous Localization and Mapping - SLAM
  - Solve Localization and Mapping simultaneously
  - Chicken-egg problem
- Mapping only
  - Assume pose of the robot is known
    - Example: GPS or DGPS

Mapping

- What should be represented:
  - Path Map
  - Topological Map
  - Features Map
  - Cell Decomposition
  - Metric Map
    - Occupancy grid map

Metric maps and Topological maps

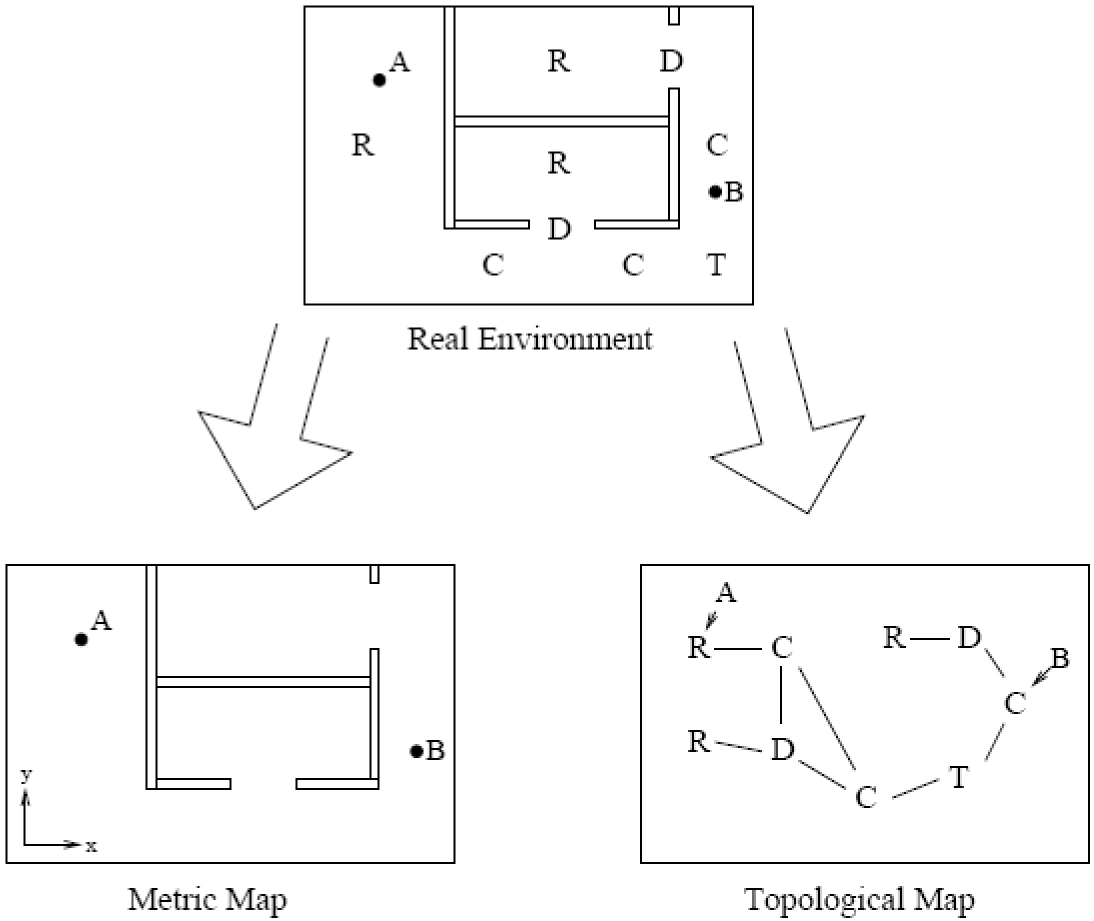
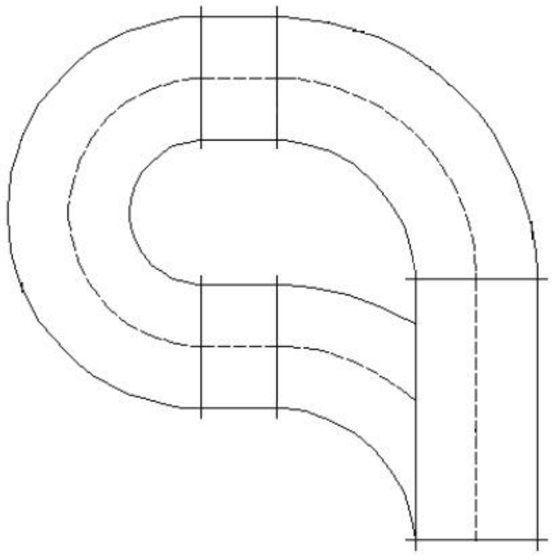
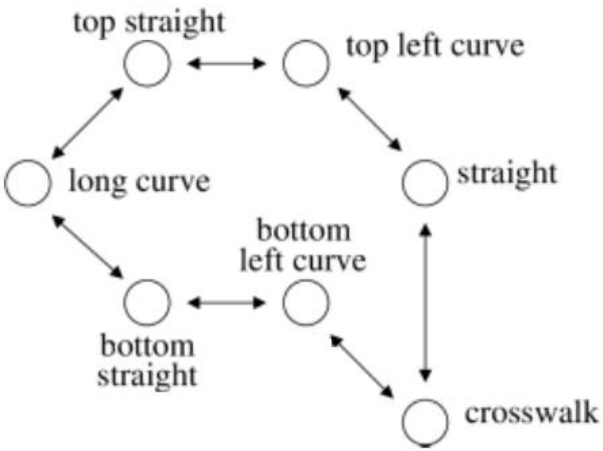


Figure from Meyer, “Map-based navigation in mobile robotics”, 2003, some other figures follow

Topological map

Topological map





Occupancy Grid

Occupancy Grid

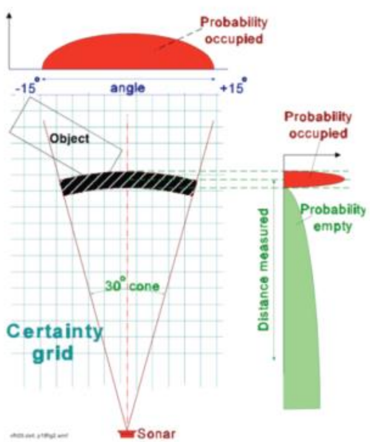
- Map is composed of cells of equal dimension
- Cells may be **occupied** of **free**
- Cells keep the probability of being occupied
  - Probability that each cell is occupied **P(H)** and probability that each cell is free **P(~H)**

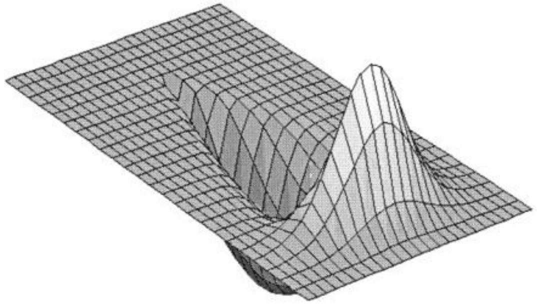
    $$0 \leq P(H) \leq 1$$

    $$P(\sim H) = 1 - P(H)$$
- Other methods store 2 different functions: one for occupied probability and another for empty probability

Mapping

- We want to determine P(H|s)
  - Probability cell is occupied given a certain measure s
  - We can use the Inverse sensor model





Mapping

Given sensor data $z_{1:t}$ and the poses $x_{1:t}$, estimate the map

$$p(m|z_{1:t}, x_{1:t}) = \Pi_i \, p(m_i|z_{1:t}, x_{1:t})$$

Applying static state binary bayes filter

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(z_t|m_i, z_{1:t-1}, x_{1:t}) \, p(m_i|z_{1:t-1}, x_{1:t})}{p(z_t|z_{1:t-1}, x_{1:t})}$$

Applying Markov assumption and independence

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(z_t|m_i, x_t) \, p(m_i|z_{1:t-1}, x_{1:t-1})}{p(z_t|z_{1:t-1}, x_{1:t})}$$

Applying Bayes Rule on 1st term and independence

$$p(m_i|z_{1:t}, x_{1:t}) = \frac{p(m_i|z_t, x_t) p(z_t|x_t) \, p(m_i|z_{1:t-1}, x_{1:t-1})}{p(m_i) \, p(z_t|z_{1:t-1}, x_{1:t})}$$

As there are only binary states, so for exact opposite state event, we get

$$p(-m_i|z_{1:t}, x_{1:t}) = \frac{p(-m_i|z_t, x_t) p(z_t|x_t) \, p(-m_i|z_{1:t-1}, x_{1:t-1})}{p(-m_i) \, p(z_t|z_{1:t-1}, x_{1:t})}$$

Mapping

Now computing the ratio of both probabilities, we obtain

$$\frac{p(m_i|z_{1:t},x_{1:t})}{p(-m_i|z_{1:t},x_{1:t})} = \frac{p(m_i|z_t,x_t)}{1-p(m_i|z_t,x_t)}\frac{p(m_i|z_{1:t-1},x_{1:t-1})}{1-p(m_i|z_{1:t-1},x_{1:t-1})}\frac{1-p(m_i)}{p(m_i)} \quad \ldots(1)$$

To simplify the expression we turn the odds ratio into the probability

As Odds(x) = $\frac{p(x)}{1-p(x)}$, which gives us

$$p(x) = [1 + Odds(x)^{-1}]^{-1}$$

Now using the expression above, we have

$$p(m_i|z_{1:t},x_{1:t}) = [1 + \frac{1-p(m_i|z_t,x_t)}{p(m_i|z_t,x_t)}\frac{1-p(m_i|z_{1:t-1},x_{1:t-1})}{p(m_i|z_{1:t-1},x_{1:t-1})}\frac{p(m_i)}{1-p(m_i)}]^{-1}$$

For efficiency reasons, we perform the calculations in log odds notation i.e. $l(x) = \log\frac{p(x)}{1-p(x)}$

$$l(m_i|z_{1:t},x_{1:t}) = l(m_i|z_t,x_t) + l(m_i|z_{1:t-1},x_{1:t-1}) - l(m_i)$$

In short,

$$l_{t,i} = invsensormodel(m_i,x_t,z_t) + l_{t-1,i} - l_0$$

Mapping

Algorithm 1: OccupancyGridMapping($\{l_{t-1,i}\}$,x_t, z_t)

foreach *m_i of the map m* **do**

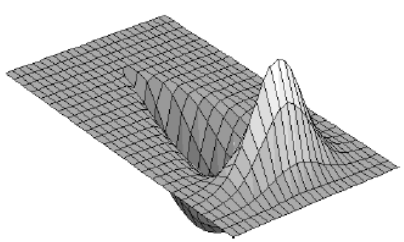
  if *m_i in the perceptual field of z_t* **then**

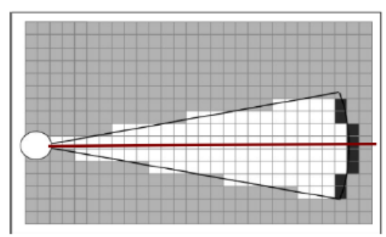
    $l_{t,i} := l_{t-1,i} + \text{inv\_sensor\_model}(m_i, x_t, z_t) - l_0$;

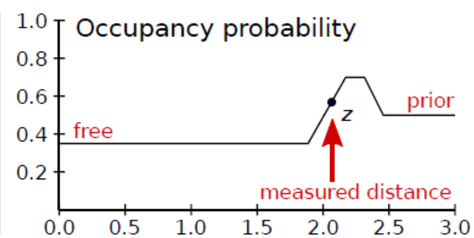
  else

    $l_{t,i} := l_{t-1,i}$;

return $\{l_{t,i}\}$

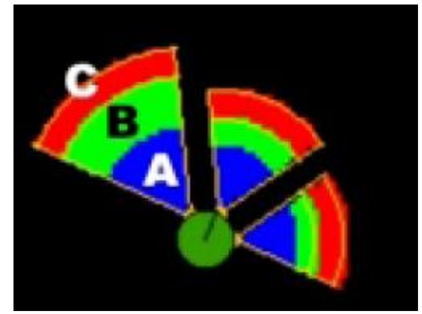




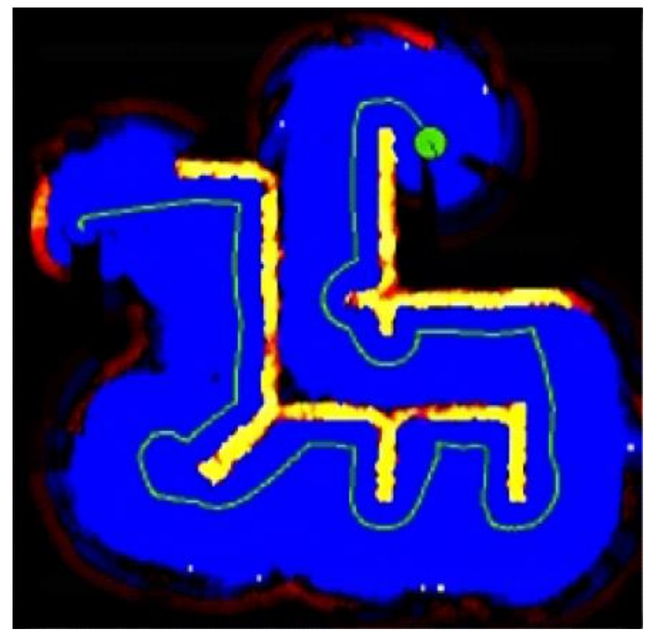


Mapping example

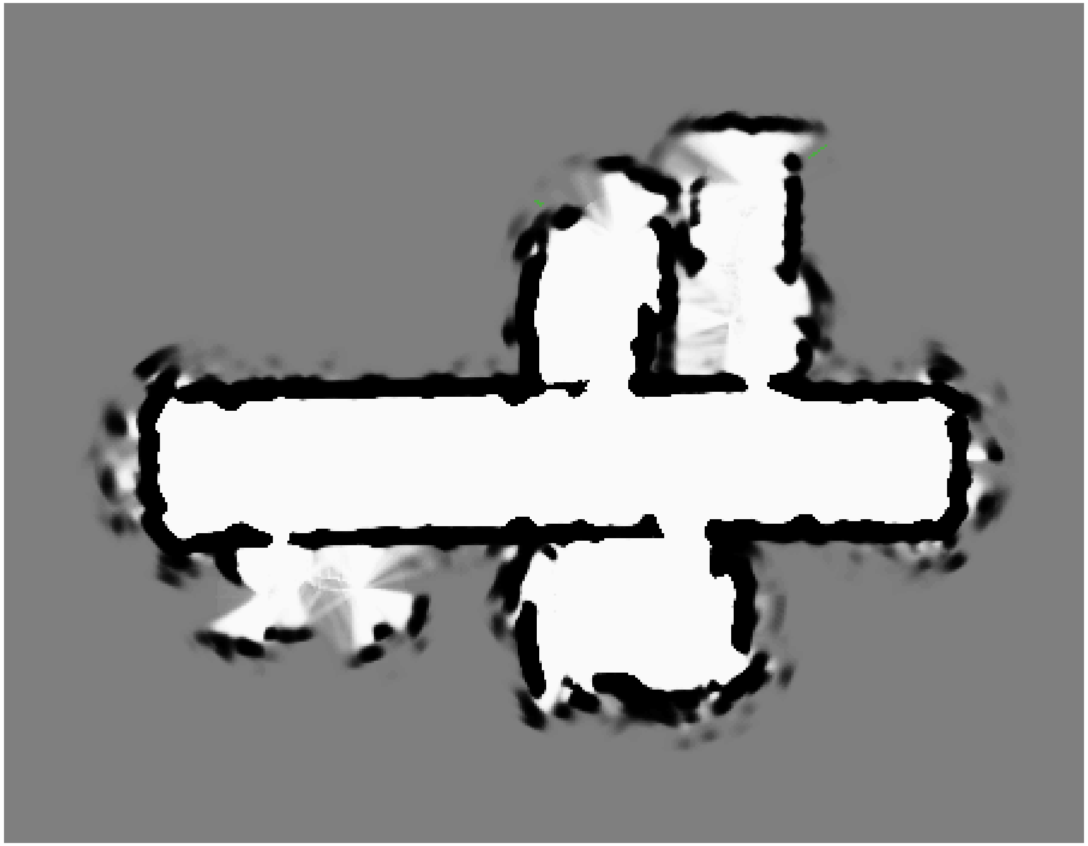
- YAM's algorithm
  - Probabilities (?) may be positive or negative
    - Positive means probably occupied
    - Negative means probably empty
  - Estimated position of the robot is assigned minimum probability
  - Field of view of obstacle sensors is divided in 3 regions:
    - Cells in Region A are assigned a low probability
    - Cells in Region B decrease their probability
    - Cells in Region C increase their probability
    - Increase in region C is 4 times the value of decrease in region B



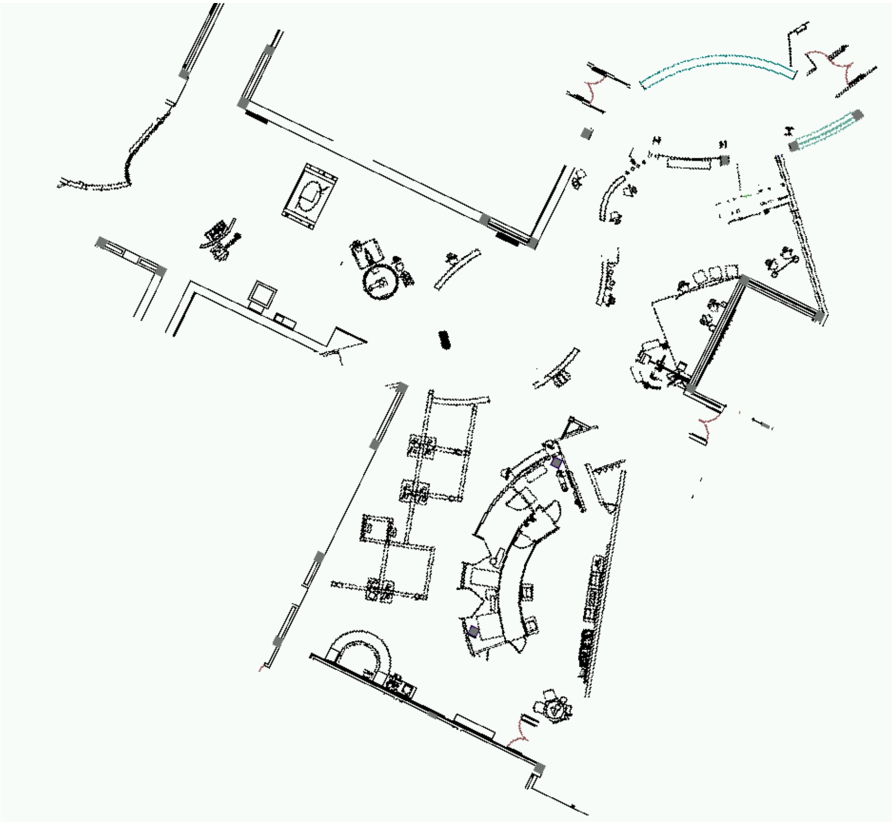
YAM

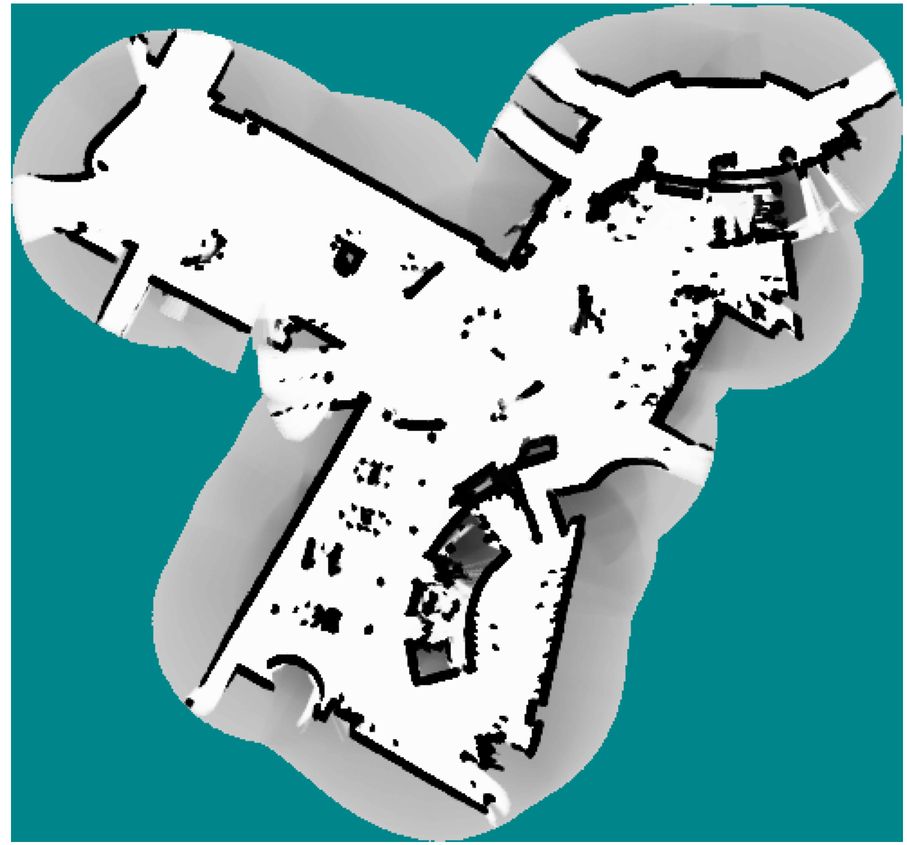


Mapping with Sonar sensors



Mapping with LRF





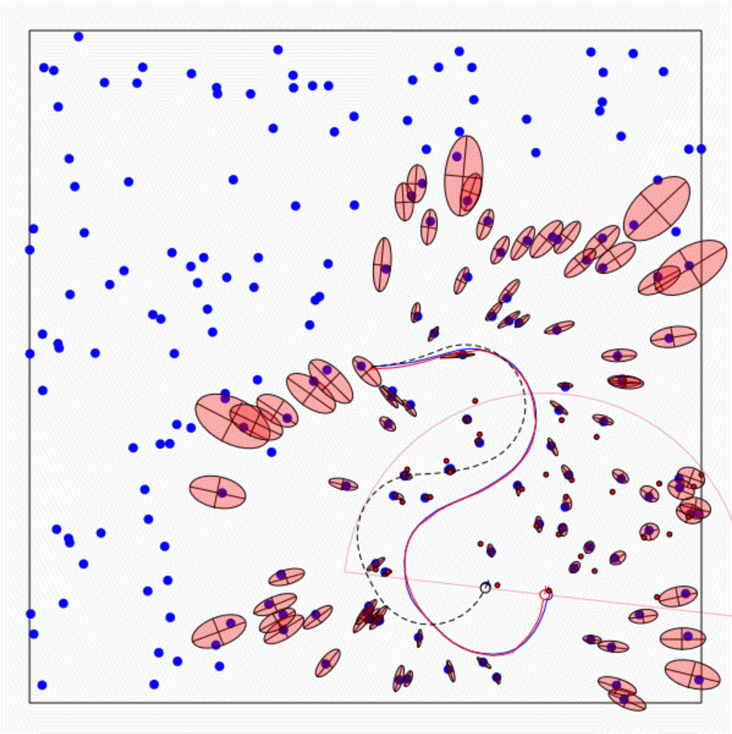
Mapping: Chicken and egg

- Mapping, as presented in the previous slides, assumes the pose of the robot is known
- Often this is not true and, while mapping, the robot must simultaneously estimate its pose.
  - To estimate its pose it may use the already known map
- The general problem is denoted as the Simultaneous Localization and Mapping problem (SLAM)

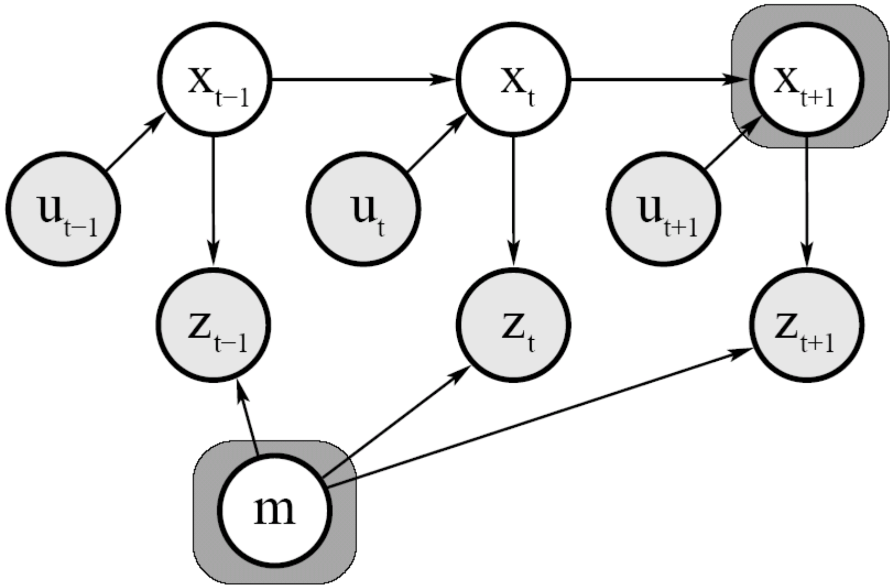
SLAM problem

A robot is exploring an unknown, static environment

- Given:
  - The robot's controls
  - Observations of nearby features
- Estimate:
  - Map of features
  - Path of the robot



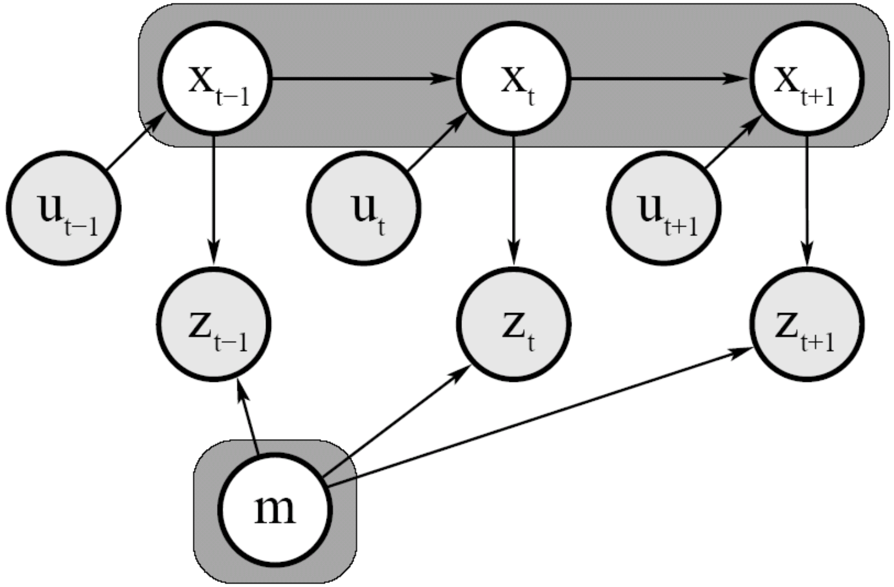
Online SLAM



$$p(x_t, m \mid z_{1:t}, u_{1:t}) = \int\int \ldots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t})\, dx_1\, dx_2 \ldots dx_{t-1}$$

- Estimates map and current position given measures and controls

Full SLAM



$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

- Estimates map and robot path given measures and controls

Some SLAM Techniques

- Scan matching
- EKF SLAM
- Fast-SLAM
- Graph-SLAM, SEIFs
- gMapping
- Hector Mapping