Robot Mapping

IR Lab, 16th Oct 2007

Zeyn Saigol

Outline

1. Introduction
2. Occupancy Grids
   - Forward vs. Inverse Models
   - Update Rule
   - Issues and Limitations
3. Kalman Filters
4. SLAM and Particle Filters

Introduction

- Aim is to build a map of the local environment of the robot
- Wide variety of sensors are used, with different charateristics (e.g. sonar, laser, IR, cameras)
- The robot may not have perfect knowledge of its own position. Estimating the map and the pose together is known as SLAM (Simultaneous Localisation And Mapping)
- Probabilistic methods have proved very effective for mapping, and have been universally adopted

Introduction / cont'd

- Map can be of obstacles (walls), or of features (e.g. landmarks, hydrothermal vents)

  Map landmarks => limited dimensionality (but still high)

  Map obstacles => very high dimensional

- Static World assumption

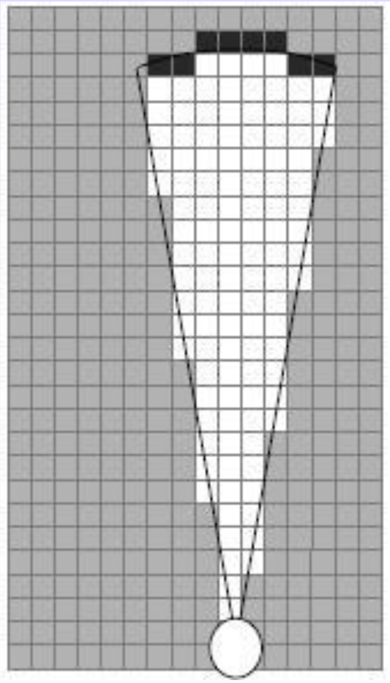
  Mapping is made much simpler by assuming that the map **m** is fixed, and is completely responsible for sensor readings z:

$$P(z_{t+1} \mid z_{1:t}, \mathbf{m}) = P(z_{t+1} \mid \mathbf{m})$$

Occupancy Grids

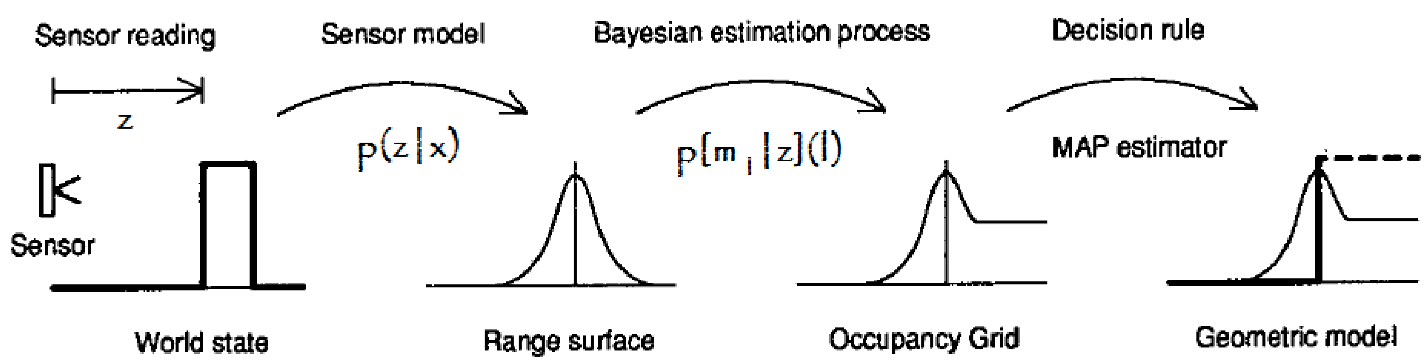
- Developed by Hans Moravec and Alberto Elfes at CMU in the 1980s
- Search area is split into cells
- $m_{x,y}$ - cell at $\langle x, y \rangle$ is occupied

  $\overline{m}_{x,y}$ - cell at $\langle x, y \rangle$ is empty
- Originally developed for sonar
- Use a Bayesian update rule to find the odds ratio
- Estimate $P(m_{x,y} \mid z_{1:t})$ **independently** for each cell



Forward and Inverse Models

- Probabilistic mapping requires a sensor model
- Sensors are naturally described by a generative or forward model, giving P(reading|true-distance), $P(z \mid x)$
- Occupancy grids need an inverse model, $P(m_x \mid z)$
- Diagram from Elfes (1989) (*Using Occupancy Grids for Mobile Robot Perception and Navigation*)



Occupancy Grid Updates

- Iterative update

$$P\left[m_{x,y} \mid z_{1:t+1}\right] = \frac{P\left[z_{t+1} \mid m_{x,y}\right] P\left[m_{x,y} \mid z_{1:t}\right]}{P\left[z_{1:t}\right]}$$

- Calculate posterior *odds ratio*, $\frac{P\left[m_{x,y} \mid z_{1:t+1}\right]}{P\left[\overline{m}_{x,y} \mid z_{1:t+1}\right]}$, to avoid needing to know $P\left[z_{1:t}\right]$
- Prior for each cell normally [0.2, 0.5]
- "After ten years of development, the Cart program was still unreliable at crossing a room, but the very first grid program succeeded every time" (Martin and Moravec, 1996)

OG Issues

- **Robot pose**

  The OG method requires exact knowledge of the location of the sensor at all points

- **Independency assumption**

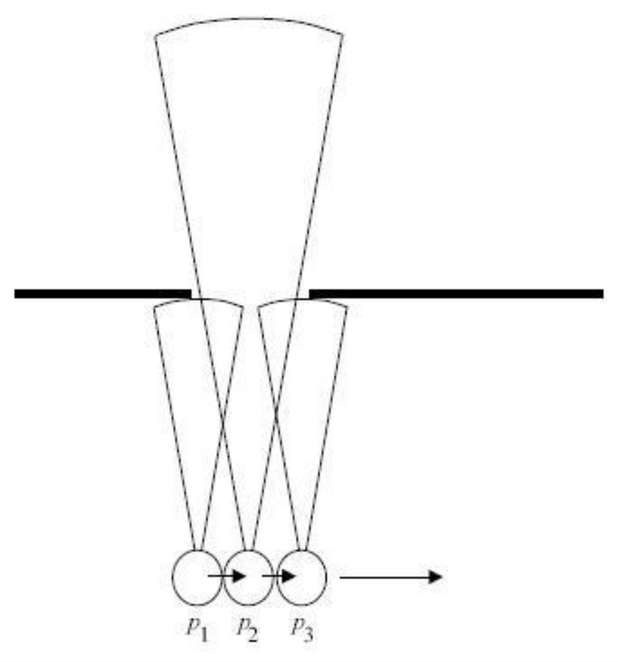
  The OG update rule assumes

  that sensor readings are conditionally

  independent given knowledge of a

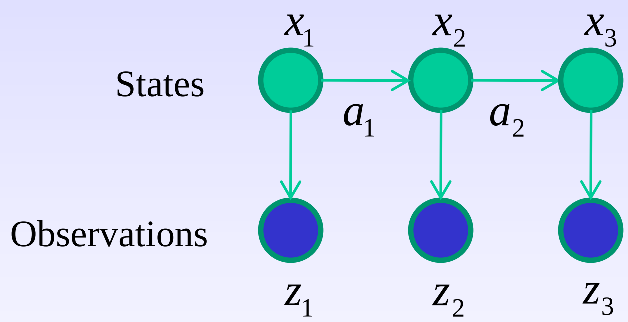
  single cell:

$$P\left[z_{t+1} \mid z_{1:t}, m_{x,y}\right] = P\left[z_{t+1} \mid m_{x,y}\right]$$



Dynamic Systems

- Want to model the robot's motion => state x_t must evolve
- Hidden Markov Models, HMMs



- Governed by two distributions
  1. State transition function or motion model, $P(x_{t+1} \mid x_t, a_t)$
  2. Observation or sensor model, $P(z_t \mid x_t)$

Kalman Filters 1

- Extension of HMMs to continuous states and observations, originally used for tracking moving objects
- Can also be used for estimating any hidden state, including a map **m** and robot pose s: $x_t = (s_t, \mathbf{m})^T$
- Both the motion and sensor models are linear with additive Gaussian noise

$$z_t = Cx_t + v_t \qquad v_t \sim N(0, R)$$

$$s_{t+1} = As_t + Gw_t \qquad w_t \sim N(0, Q)$$

- Initial conditions must have Gaussian uncertainty
- Given these, $P(x_t \mid z_{1:t})$ is also Gaussian, and there is an analytic solution for its mean and covariance

Kalman Filters 2

- Analytic solution comes from Bayesian update equation:

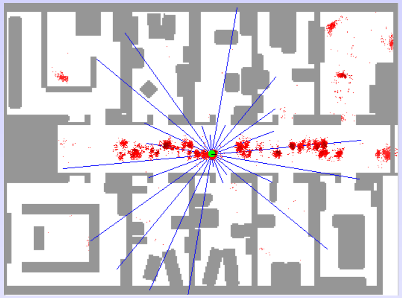
$$P(s_{t+1}, \mathbf{m} \mid \mathbf{z}_{1:t+1}, a_{1:t})$$
$$= \alpha P(\mathbf{z}_{t+1} \mid s_{t+1}, \mathbf{m}) \int P(s_{t+1} \mid s_t, a_t) P(s_t, \mathbf{m} \mid \mathbf{z}_{1:t}, a_{1:t-1}) ds_t$$

- In practice, motion and sensor models are usually significantly nonlinear
- Therefore, Extended Kalman Filters (EKFs) are frequently used in real mapping problems
- EKFs work by linearising the models about each point, using the first few terms in a Taylor expansion

SLAM

- The SLAM problem: attempt to reduce the correlated errors in robot pose and map by estimating both simultaneously
- When a previously-observed feature is seen again, the uncertainty over pose is reduced, and therefore the uncertainty over new feature locations is also reduced
- EKFs have been widely and successfully used for SLAM
- However, posterior over map and pose is Gaussian
- Also relies on known, identifiable landmarks (the *correspondence problem*)

Particle Filter-based SLAM



- Solve Gaussian posterior issue by using a particle filter
- However, number of particles needed is exponential in the number of dimensions
- ... and mapping is very high dimensional
- Solution: Use particles to track robot's pose, and associate a Kalman filter with each particle to estimate the map
- Essentially Rao-Blackwellization (Sebastian Thrun's FastSLAM , e.g. *Simultaneous Localisation and Mapping with Unknown Data Association Using FastSLAM*, Montemerlo and Thrun, ICRA 2003)

Summary

- Occupancy grids good for generating complete maps, given accurate localisation of the robot
- SLAM problem commonly solved by EKFs, or a combined EKF/particle filters method
- Other major technique is EM (slightly complicated)
- EM is an iterative procedure to find the maximum likelihood solution by searching over possible maps
- RatSLAM seems to use non-Cartesian grid, where connections between visitable points are more important than absolute location