Occupancy Grid Mapping

EE468/CE468: Mobile Robotics

Dr. Basit Memon

Electrical and Computer Engineering
Habib University

November 13, 2023

Table of Contents

Table of Contents

Why do we need maps?

- Autonomous robots use maps for localization, path planning, mission planning, etc.

- Learning maps is one of the most fundamental problems for autonomous robots.

Do we need to learn maps online?

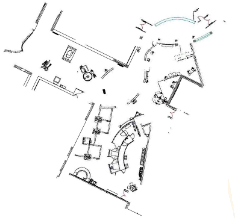


Figure: Floor plans can be wrong

- Furniture, etc. is not captured in architectural maps.
- Reduces set up time for installation of mobile robots.
- Mapping unstructured environments.

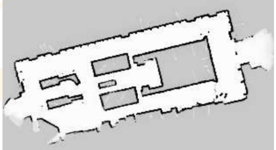


Figure: (Top) Raw sensor data (Bottom) Processed map

- **Noise in perception and actuation**
- If sensing and actuation were noise free, mapping will be a simple problem.
- Practically, learning maps is “chicken-and-egg” problem. Robot needs to localize itself and learn the map.

Mapping problem can be set up as probabilistic problem.

- Given the sensor and control data,

$$d = \{z_1, z_2, \cdots, z_t, u_1, u_2, \cdots, u_t\}$$

calculate the most likely map

$$m^* = \arg\max_m \; p(m|d).$$

- Today, we'll solve the simpler problem of learning map, assuming robot pose is known perfectly.

Learning maps with known poses is a simple problem.

- Given the sensor and **pose** data,

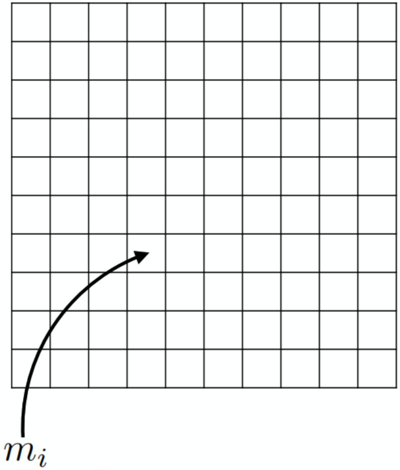
$$d = \{z_1, z_2, \cdots, z_t, x_1, x_2, \cdots, x_t\}$$

calculate the most likely map

$$m^* = \arg\max_m p(m|d).$$

Table of Contents

Learning maps with known poses is a simple problem.

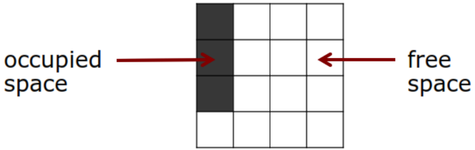


- Represent the world as a grid made up of uniform cells, m_i,

$$\text{Map: } m = \{m_i\}.$$

- This is **occupancy grid map**. Introduced by Moravec and Elfes in 1987.
- Map is assumed to be static.

Each cell is modeled as a binary random variable.



- Area that corresponds to a cell is either completely free or occupied.

$$m_i = \begin{cases} 1 & \text{if occupied,} \\ 0 & \text{if free.} \end{cases}$$

- Algorithm will assign a probability value, $[0, 1]$, to each cell:

$$\begin{aligned} P(m_i) &= 1 \rightarrow & &\text{Cell is occupied,} \\ P(m_i) &= 0 \rightarrow & &\text{Cell is free,} \\ P(m_i) &= 0.5 \rightarrow & &\text{No information about cell.} \end{aligned}$$

Estimating occupancy grid map from this data is still intractable.

- Given **sensor data** from past till current time, $z_{1:t}$, and **pose data**, $x_{1:t}$, estimate the map

$$p(m|z_{1:t}, x_{1:t}).$$

- If we map $25m \times 25m$ area at $25cm$ resolution, we have a 100×100 grid, i.e. $10,000$ cells. **How many maps are possible?** 2^{10000}!

- Computing this posterior probability over the space of all maps is computationally intractable.

Assume that map cells are independent of each other.

- Approximate the map distribution assuming independence across cells,

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

Find probability of single cell using Bayes filter

$$P(m_i|z_{1:t}, x_{1:t}) = P(m_i|\underbrace{z_{1:t-1}, x_{1:t-1}}_{\text{old data}}, \underbrace{z_t, x_t}_{\text{new data}})$$

$$(Bayes) = \eta\, P(z_t|m_i, z_{1:t-1}, x_{1:t-1}, x_t)\, P(m_i|z_{1:t-1}, x_{1:t-1}, x_t)$$

$$(Markov) = \eta\, P(z_t|m_i, x_t)\, \underbrace{P(m_i|z_{1:t-1}, x_{1:t-1})}_{\text{Previous estimate}}$$

Sensor model, $P(z_t|m_i, x_t)$, is hard to find.

- If the robot has a lidar, this is distribution of the entire scan at a time conditioned occupancy of one cell.
- Apply Bayes rule to get an inverse sensor model.

$$P(z_t|m_i, x_t) = \frac{P(m_i|z_t, x_t)\,P(z_t|x_t)}{P(m_i|x_t)}$$

Plugging inverse sensor model in our occupancy filter

$$P(m_i|z_{1:t}, x_{1:t}) = P(m_i|\underbrace{z_{1:t-1}, x_{1:t-1}}_{\text{old data}}, \underbrace{z_t, x_t}_{\text{new data}})$$

$$(\text{Bayes}) = \eta\, P(z_t|m_i, z_{1:t-1}, x_{1:t-1}, x_t)\, P(m_i|z_{1:t-1}, x_{1:t-1}, x_t)$$

$$(\text{Markov}) = \eta\, P(z_t|m_i, x_t)\, \underbrace{P(m_i|z_{1:t-1}, x_{1:t-1})}_{\text{Previous estimate}}$$

$$(\text{Bayes}) = \eta\, \frac{P(m_i|z_t, x_t)\, P(z_t|x_t)}{P(m_i|x_t)}\, P(m_i|z_{1:t-1}, x_{1:t-1})$$

$$(\text{Markov}) = \eta\, \frac{P(m_i|z_t, x_t)\, P(z_t|x_t)}{P(m_i)}\, P(m_i|z_{1:t-1}, x_{1:t-1})$$

There are still terms in expression we don't want to compute.

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

Let's compute the probability of the opposite event:

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

Let's look at the following ratio, called **odds**:

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

Note that we don't have to compute some terms.

Expression in log odds ratio form is more computationally elegant.

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

$$= \underbrace{\frac{P(m_i|z_t,x_t)}{1-P(m_i|z_t,x_t)}}_{\text{Inverse Sensor Model}}\,\underbrace{\frac{1-P(m_i)}{P(m_i)}}_{\text{Prior}}\,\underbrace{\frac{P(m_i|z_{1:t-1},x_{1:t-1})}{1-P(m_i|z_{1:t-1},x_{1:t-1})}}_{\text{Previous belief}}$$

Taking log of all terms:

$$\log\left(\frac{P(m_i|z_{1:t},x_{1:t})}{P(\neg m_i|z_{1:t},x_{1:t})}\right) = \log\left(\frac{P(m_i|z_t,x_t)}{1-P(m_i|z_t,x_t)}\right) - \log\left(\frac{P(m_i)}{1-P(m_i)}\right) + \log\left(\frac{P(m_i|z_{1:t-}}{1-P(m_i|z_{1:}}\right.$$

$$l_t = l(m_i|z_t,x_t) - l_0 + l_{t-1}$$

Occupancy Grid Mapping Algorithm

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

 for all cells $\mathbf{m}_i$ *do*

  if $\mathbf{m}_i$ *in perceptual field of* z_t *then*

   $l_{t,i} = l_{t-1,i} + \textbf{inverse\_sensor\_model}(\mathbf{m}_i, x_t, z_t) - l_0$

  else

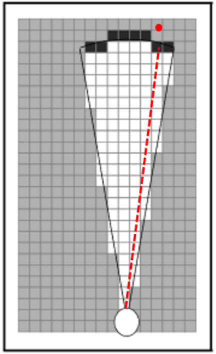
   $l_{t,i} = l_{t-1,i}$

  endif

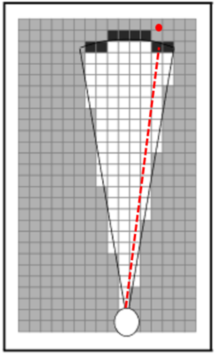
 endfor

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

How do we find inverse sensor model, $p(m_i|x_t, z_t)$?



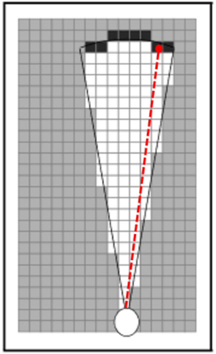
- Thrun learned model using a neural network [3, 9.3].
- Elfes used a probabilistic model for determining this quantity [2].



- Given cell i and robot pose, (x, y, θ),
  - Find index, k, of sensor beam that is closest in heading to cell i.
  - Find distance, r, between robot position and center of mass of m_i.
- If cell is sufficiently farther than obtained range measurement or out of field of view of sensor, then

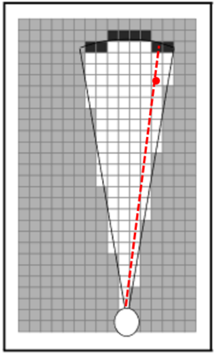
$$P(m_i) = P(m_i) \text{ at previous time.}$$

An inverse sensor model for laser range finders



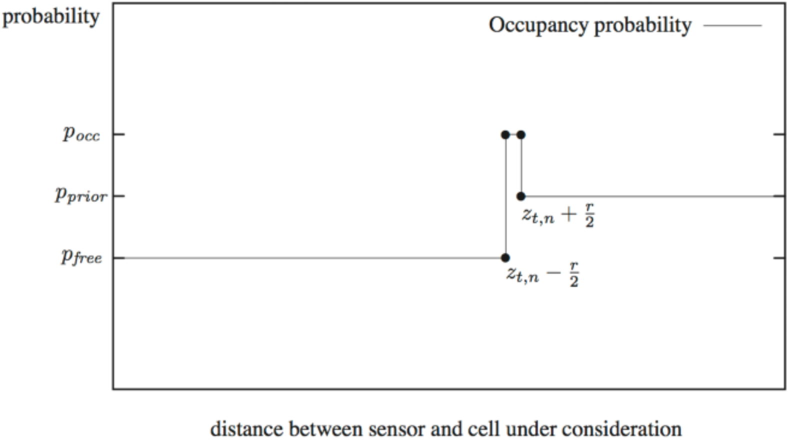
- Given cell i and robot pose, (x, y, θ),
  - Find index, k, of sensor beam that is closest in heading to cell i.
  - Find distance, r, between robot position and center of mass of m_i.
- If cell is nearly as far as the obtained range measurement, then **return probability value well above 0.5**

An inverse sensor model for laser range finders



- Given cell i and robot pose, (x, y, θ),
  - Find index, k, of sensor beam that is closest in heading to cell i.
  - Find distance, r, between robot position and center of mass of m_i.
- If cell is sufficiently closer than the obtained range measurement, then **return probability value well below 0.5**

An inverse sensor model for laser range finders



Courtesy: C. Stachniss

Inverse Sensor Model Algorithm

```
Algorithm inverse_range_sensor_model(m_i, x_t, z_t):
    Let x_i, y_i be the center-of-mass of m_i
    r = sqrt((x_i - x)^2 + (y_i - y)^2)
    φ = atan2(y_i - y, x_i - x) - θ
    k = argmin_j |φ - θ_{j,sens}|
    if r > min(z_max, z_t^k + α/2) or |φ - θ_{k,sens}| > β/2 then
        return l_0
    if z_t^k < z_max and |r - z_t^k| < α/2
        return l_occ
    if r ≤ z_t^k
        return l_free
    endif
```

Algorithm inverse_range_sensor_model($\mathbf{m}_i, x_t, z_t$):

- *Let* x_i, y_i *be the center-of-mass of* $\mathbf{m}_i$
- $r = \sqrt{(x_i - x)^2 + (y_i - y)^2}$
- $\phi = \mathrm{atan2}(y_i - y, x_i - x) - \theta$
- $k = \mathrm{argmin}_j \, |\phi - \theta_{j,\mathrm{sens}}|$
- *if* $r > \min(z_{\max}, z_t^k + \alpha/2)$ *or* $|\phi - \theta_{k,\mathrm{sens}}| > \beta/2$ *then*
  - *return* l_0
- *if* $z_t^k < z_{\max}$ *and* $|r - z_t^k| < \alpha/2$
  - *return* l_{occ}
- *if* $r \leq z_t^k$
  - *return* l_{free}
- *endif*

Figure: α is grid cell size and β is width of the beam.

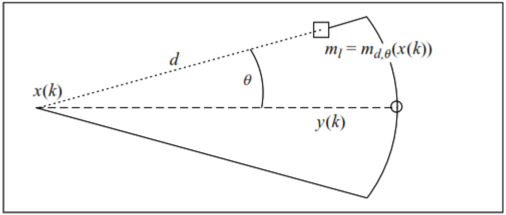


Figure 9.12 The occupancy probability of a cell $m_l = m_{d,\theta}(x(k))$ depends on the distance d to $x(k)$ and the angle θ to the optical axis of the cone.

- $y(k)$ is measurement z_t.
- θ is angle of cell from optical axis.

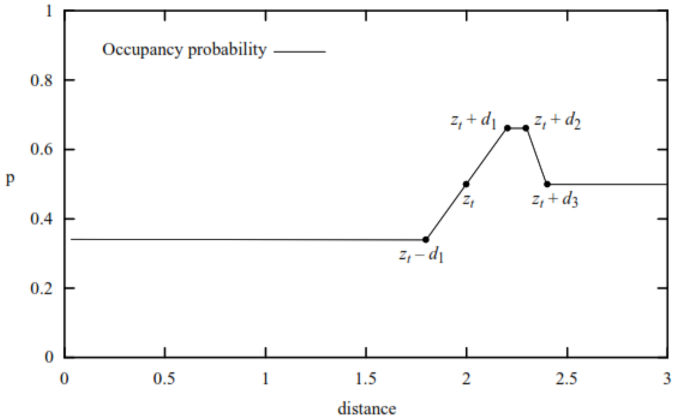


Figure: Occupancy probability wrt distance at $\theta = 0^\circ$.

Inverse sensor model is a mixture of Gaussian and linear

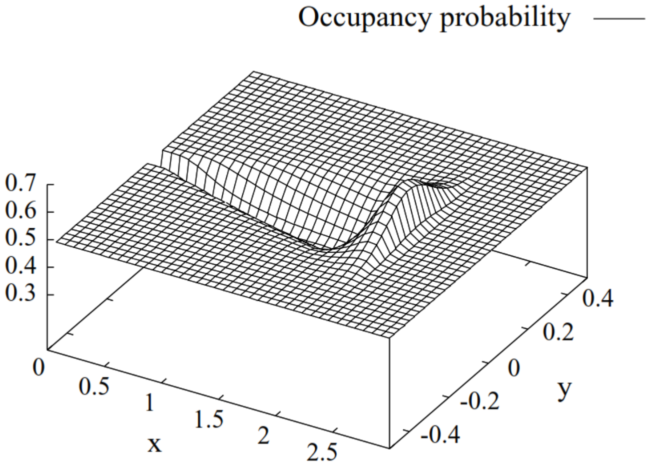


Figure: Occupancy probability for different cells when measurement is 2 m.

Occupancy Grid Map Example

Figure 9.18 Occupancy probability map for the corridor of the Autonomous Intelligent Systems Lab at the University of Freiburg (left) and the corresponding maximum-likelihood map (right).

Figure: Right map is obtained by rounding off to 1 or 0.

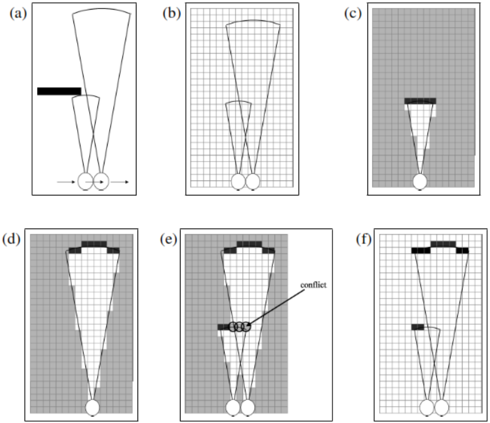


Figure: One solution is to use MAP estimator.

Table of Contents

[1] **Howie Choset, Kevin M Lynch, Seth Hutchinson, George A Kantor, and Wolfram Burgard.**
Principles of robot motion: theory, algorithms, and implementations.
MIT press, 2005.

[2] **Alberto Elfes.**
Using occupancy grids for mobile robot perception and navigation.
Computer, 22(6):46–57, 1989.

[3] **Sebastian Thrun, Wolfram Burgard, and Dieter Fox.**
Probabilistic robotics.
2006.