Lecture 10 - Grid Maps

- all previous filters require/assume Gaussian distributions
  ↳ limitation when representing high probability of robot being in 2 distinct places ("i'm sure the bot is in one of these two poses")
  ↳ Gaussians cannot represent Bimodal Distributions (or multimodal)

Features (what we used so far...)

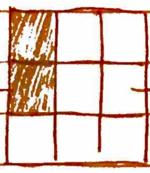
- natural choice for Kalman-based SLAM
- compact representation
- multiple observations improve landmark position estimate (EKF)
- system will depend on having a good feature detector

Grid Maps

- Discretize the world into cells
  ↳ Grid structure is rigid
- each cell is either fully occupied or fully free
- non-parametric model
- hard to scale because it's space-extensive
  ↳ BUT; we don't rely on a feature detector, work directly with the raw sensor data
- non-perfect alignment of grid map & objects can result in grey borders → eg. Diagonal walls

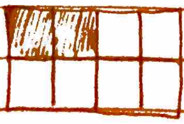
Assumption 1

- cell either free or not

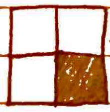
not free ←  → free

↳ Representation:

- each cell is a binary random variable that models occupancy

$P(m_i)=1$ →  ← $P(m_i)=0$ No Knowledge; $P(m_i)=0.5$

Assumption 2



→ always free

→ always occupied

- the world is static
  - ↳ (most mapping systems make this assumption)

Assumption 3

- all cells (the bin. random vars) are independent of eachother
  - ↳ not necessarily bad but be aware of this

Representation;

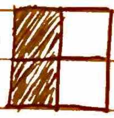
$$P(m) = \prod_i p(m_i)$$

↳ map   ↳ cell

→ find probability that the world looks like this

eg.

m_1	m_2
m_3	m_4

$m_1 = 0.9$, $m_3 = 0.8$ lets say $m =$ 

$m_2 = 0.5$, $m_4 = 0.1$

$$P(M=m) = \prod_{i=1}^{4} P(M_i = m_i) = 0.9(1-0.5)0.8(1-0.1) \boxed{= (0.9)(0.5)(0.8)(0.9)}$$

Estimating a Map from Data

- Given sensor data $z_{1:t}$ & poses of robot $x_{1:t}$ (not SLAM) estimate the map

$$P(m \mid z_{1:t}, x_{1:t}) = \prod_i P(m_i \mid z_{1:t}, x_{1:t})$$

- Assumption: poses of the robot are known
- * - Do not need a prediction step
  - ↳ because of assumption 2 static world. we don't have any controls, only observations → so you know where you are
- ↳ will use a Binary Bayes Filter for Static State for this

Bayes Rule: $P(A|B,C) = \frac{P(B|A,C)P(A|C)}{P(B|C)}$ $\quad P(B|A)P(A) = P(A,B)$

Static State Binary Bayes Filter

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

Prev. observ. useless since we have m_i

x_t useless since we don't have z_t

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

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

Knowing where u are w/o obser. doesn't help the map

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

↳ now do the same for opposite event (exploiting assumption 1)

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

- now lets take ratio of these. we get (after canceling things out):

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

$$\frac{P(m_i | z_{1:t}, x_{1:t})}{1 - P(m_i | z_{1:t}, x_{1:t})} = \underbrace{\frac{P(m_i | z_t, x_t)}{1 - P(m_i | z_t, x_t)}}_{\text{uses curr. observ } z_t} \cdot \underbrace{\frac{P(m_i | z_{1:t-1}, x_{1:t-1})}{1 - P(m_i | z_{1:t-1}, x_{1:t-1})}}_{\text{recursive term of recursive Bayes filter}} \cdot \underbrace{\frac{1 - P(m_i)}{P(m_i)}}_{\text{Prior}}$$

↳ now how to go from ratio to probability: $\frac{P(x)}{1-P(x)} = y$

$$P(x) = y - yP(x)$$

$$P(x)(1+y) = y$$

$$\boxed{P(x) = \frac{y}{(1+y)} = \frac{1}{\frac{1}{y}+1}} = (1+y^{-1})^{-1}$$

- So if we sub the last step we had with Y; $\to$ this inverse is costly

$$P(m_i \mid z_{1:t}, x_{1:t}) = \left[1 + \frac{1-P(m_i \mid z_t, x_t)}{P(m_i \mid z_t, x_t)} \frac{1-P(m_i \mid z_{1:t-1}, x_{1:t-1})}{P(m_i \mid z_{1:t-1}, x_{1:t-1})} \frac{P(m_i)}{1-P(m_i)}\right]^{-1}$$

<u>Log Odds Notation</u> $l(x) = \log\left(\frac{P(x)}{1-P(x)}\right)$

$$P(x) = 1 - 1/(1+\exp l(x))$$

- So product turns to sum;

$$l(m_i \mid z_{1:t}, x_{1:t}) = \underbrace{l(m_i \mid z_t, x_t)}_{\text{inverse sensor model}} + \underbrace{l(m_i \mid z_{1:t-1}, x_{1:t-1})}_{\text{recursive term}} - \underbrace{l(m_i)}_{\text{prior}}$$

- Or in short;

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

$\hookrightarrow$ just uses sums which is very efficient.

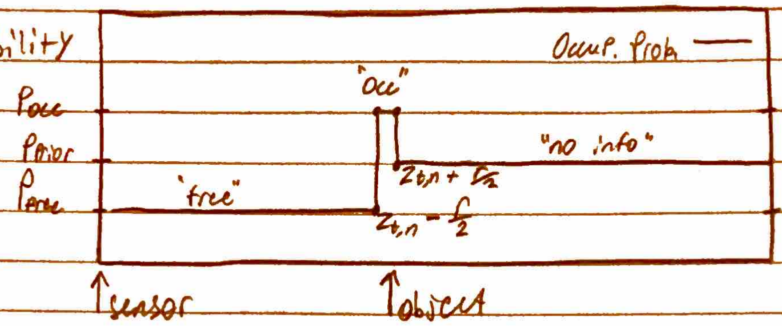
Occupancy Mapping Algorithm

```
Occupancy_grid_mapping({l_{t-1,i}}, x_t, z_t):
    for all cells m_i do:
        if m_i in perceptual field of z_t then
            l_{t,i} = l_{t-1,i} + inv_sensor_model(m_i, x_t, z_t) - l_0
        else
            l_{t,i} = l_{t-1,i}
    return {l_{t,i}}
```

- Proposed in mid 80's
- Developed for noisy sonar data
- also called "mapping with known poses

- what does the inverse sensor model look like?

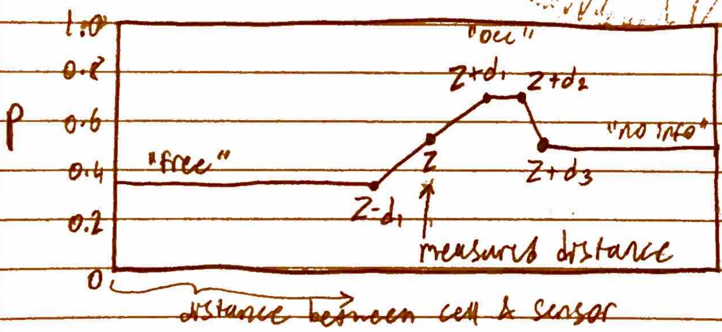
Inverse Sensor Model for Lazer Range Finders



r = dim of grid cell ?

... for Sonar Range Sensors

- here a cone rather than streit line



→ use linear function to accomidate for noise of this type of sensor

Which cell to update for a single lazer beem ?
↳ Bresenham's line algorithm

Summary
- Static State Binary Bayes Filter per cell
- mapping with known poses – easy
- log odds model makes it fast to compute
- no need for predifined features (astiny range)
- assumed known poses so far
- can be applied to 3D