NA 568 - Winter 2022

Occupancy Grid Mapping

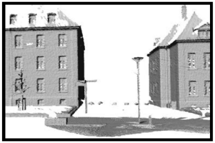
Maani Ghaffari

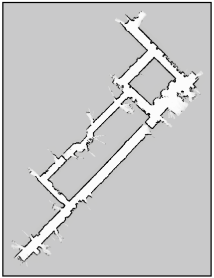
Slides: Courtesy of Ryan Eustice

February 22, 2022

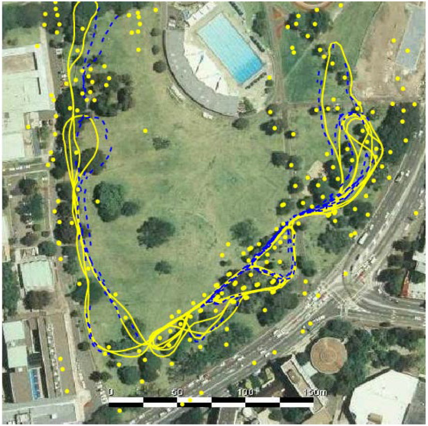
UNIVERSITY OF MICHIGAN

Features vs. Volumetric Maps





Courtesy: D. Hähnel



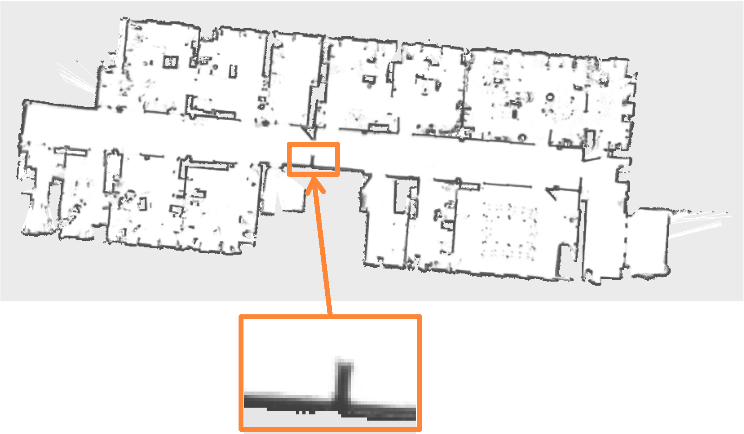
Courtesy: E. Nebot

Grid Maps

- Discretize the world into cells
- Grid structure is rigid
- Each cell is assumed to be occupied or free space
- Non-parametric model
- Large maps require substantial memory resources
- Do not rely on a feature detector

Courtesy: C. Stachniss

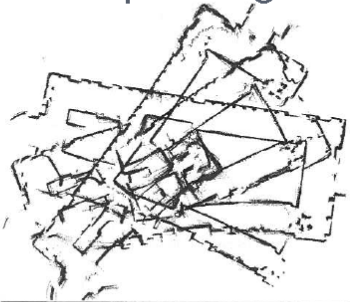
Example

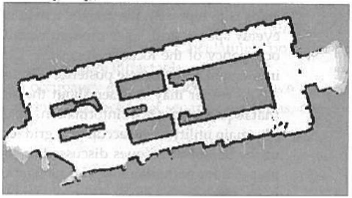


Courtesy: C. Stachniss

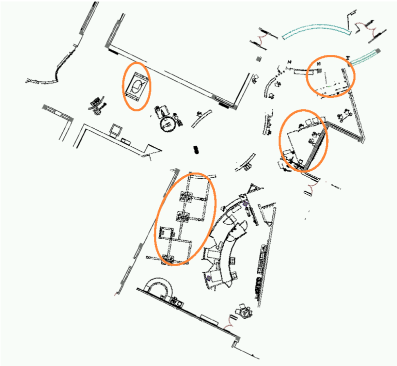
Occupancy maps provide a notion of "free space" – important for planning

- Raw laser scans placed using odometry
  - Inconsistent
- Solve for poses using SLAM
  - Consistent, but no notion of "free space"
- Use SLAM-derived poses as input to generate an occupancy map
  - Planning, localization

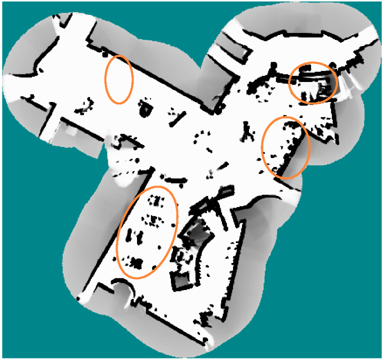




Even when we have prior maps, they can be inaccurate…



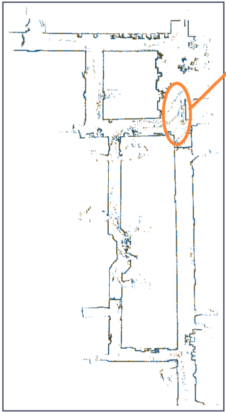
CAD map



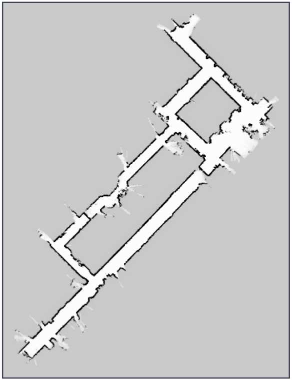
occupancy grid map

Tech Museum, San Jose

Occupancy Grids: From scans to maps



False "hits" from people in the environment



Occupancy Grid Mapping

- Moravec and Elfes proposed occupancy grid mapping in the mid 1980's
- Developed for noisy sonar sensors
- Also called "mapping with know poses"

Courtesy: C. Stachniss

Occupancy Grid Maps (OGM)

- Introduced by Moravec and Elfes in 1987
- Represent environment by a grid
  - e.g. $25\ \mathrm{m} \times 25\ \mathrm{m}$ area at $25\ \mathrm{cm}$ resolution yields a 100×100 grid = $10{,}000$ cells
- Estimate the probability that a cell is occupied by an obstacle.

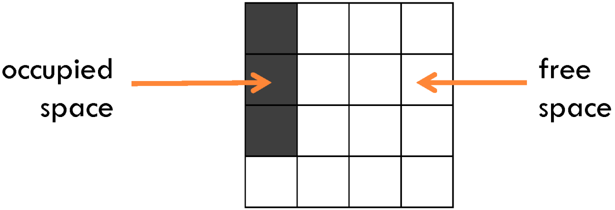
  Binary state: $\mathbf{m}_i = \begin{cases} 1 & \text{occupied} \\ 0 & \text{free} \end{cases}$

- Map: $m = \{m_i\}$ Belief: $bel_t(m) = p(m \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$
  - Discrete Bayes estimation problem.
  - In our example above, how many possible maps?

That's 10^{3010} maps!!!

Assumption 1

- The area that corresponds to a cell is either completely free or occupied

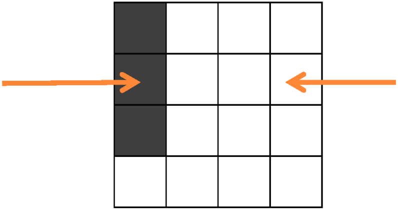


Courtesy: C. Stachniss

Representation

- Each cell is a **binary random variable** that models the occupancy

$p(m_i) \rightarrow 1$



$p(m_j) \rightarrow 0$

Courtesy: C. Stachniss

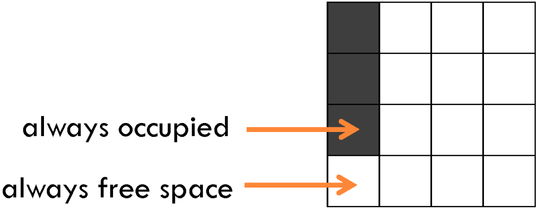
Occupancy Probability

- Each cell is a **binary random variable** that models the occupancy
  - Cell is occupied: $p(m_i) = 1$
  - Cell is not occupied: $p(m_i) = 0$
  - No knowledge: $p(m_i) = 0.5$

Courtesy: C. Stachniss

Assumption 2

- The world is **static** (most mapping systems make this assumption)

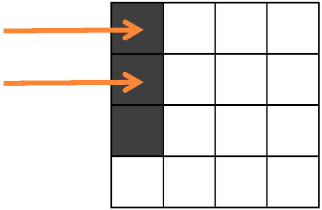


Courtesy: C. Stachniss

Assumption 3

- The cells (the random variables) are **independent** of each other

no dependency
between the cells



Courtesy: C. Stachniss

Representation

- The probability distribution of the map is given by the product over the cells

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

map

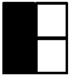
cell

Courtesy: C. Stachniss

Representation

- The probability distribution of the map is given by the product over the cells

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



example map
(4-dim state)

4 individual cells

Courtesy: C. Stachniss

Estimating a Map From Data

- Given sensor data $\mathbf{z}_{1:t}$ and the poses $\mathbf{x}_{1:t}$ of the sensor, estimate the map

$$p(m \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \prod_i p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

binary random variable

⇨ Binary Bayes filter
(for a static state)

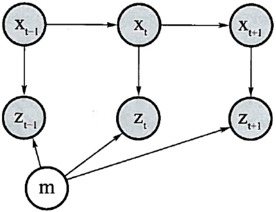
Courtesy: C. Stachniss

Occupancy Grid Maps (OGM)

- Key assumptions (for tractability)
  - Robot positions are known!
  - Occupancy of individual cells (m_i) are independent

$$bel_t(m) = p(m \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \prod_i p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})$$

- OGM graphical model
  - $\mathbf{z}$ and $\mathbf{x}$ are known (shaded)
  - Goal is to infer map m
  - Controls $\mathbf{u}$ play no roll in the belief since $\mathbf{x}$ are given



Static State Binary Bayes Filter

$$p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

$$p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

$$p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\; p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\; p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

Forward model

- When the measurement space is more complex than the state space, an inverse sensor model may be easier to come by.
  - e.g., determining if a door is open or closed from a camera image

- Rewriting in terms of inverse sensor model we have:

$$p(\mathbf{z}_t \mid m_i, \mathbf{x}_t) \overset{\text{Bayes rule}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\; p(\mathbf{z}_t \mid \mathbf{x}_t)}{p(m_i \mid \mathbf{x}_t)}$$

Inverse model

Static State Binary Bayes Filter

$$p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}$$

$$p(\mathbf{z}_t \mid m_i, \mathbf{x}_t) \overset{\text{Bayes rule}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)}{p(m_i \mid \mathbf{x}_t)}$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

$$
\begin{aligned}
p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) &\overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Bayes rule}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i \mid \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}
\end{aligned}
$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

$$
\begin{aligned}
p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) &\overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Bayes rule}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i \mid \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{indep.}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}
\end{aligned}
$$

- *Made for sheer convenience (actually the pose of the robot tells us that the cell must be free!)*

Static State Binary Bayes Filter

$$
\begin{aligned}
p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) &\overset{\text{Bayes rule}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Markov}}{=} \frac{p(\mathbf{z}_t \mid m_i, \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{Bayes rule}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i \mid \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})} \\
&\overset{\text{indep.}}{=} \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(m_i)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}
\end{aligned}
$$

Do exactly the same for the opposite event:

$$
p(\neg m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) \overset{\text{the same}}{=} \frac{p(\neg m_i \mid \mathbf{z}_t, \mathbf{x}_t)\, p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(\neg m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{p(\neg m_i)\, p(\mathbf{z}_t \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})}
$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

- By computing the ratio of both probabilities, we obtain:

$$\frac{p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(\neg m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \frac{\dfrac{p(m_i|\mathbf{z}_t,\mathbf{x}_t)\ \cancel{p(\mathbf{z}_t|\mathbf{x}_t)}\ p(m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{p(m_i)\ \cancel{p(\mathbf{z}_t|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}}}{\dfrac{p(\neg m_i|\mathbf{z}_t,\mathbf{x}_t)\ \cancel{p(\mathbf{z}_t|\mathbf{x}_t)}\ p(\neg m_i|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t-1})}{p(\neg m_i)\ \cancel{p(\mathbf{z}_t|\mathbf{z}_{1:t-1},\mathbf{x}_{1:t})}}}$$

  - Note how this eliminates difficult to come by quantities

Courtesy: C. Stachniss

Static State Binary Bayes Filter

- By computing the ratio of both probabilities, we obtain:

$$
\begin{aligned}
&\frac{p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{p(\neg m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} \\
&= \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\; p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})\; p(\neg m_i)}{p(\neg m_i \mid \mathbf{z}_t, \mathbf{x}_t)\; p(\neg m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})\; p(m_i)} \\
&= \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}{1 - p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}\; \frac{p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{1 - p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}\; \frac{1 - p(m_i)}{p(m_i)}
\end{aligned}
$$

Courtesy: C. Stachniss

Static State Binary Bayes Filter

- By computing the ratio of both probabilities, we obtain:

$$\frac{p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{1 - p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}$$

$$= \frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)\ p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})\ p(\neg m_i)}{p(\neg m_i \mid \mathbf{z}_t, \mathbf{x}_t)\ p(\neg m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})\ p(m_i)}$$

$$= \underbrace{\frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}{1 - p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}}_{\text{uses } \mathbf{z}_t} \underbrace{\frac{p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{1 - p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}}_{\text{recursive term}} \underbrace{\frac{1 - p(m_i)}{p(m_i)}}_{\text{prior}}$$

Courtesy: C. Stachniss

Log Odds Notation

- Log odds ratio is defined as

$$l(x) \quad = \quad \log \frac{p(x)}{1 - p(x)}$$

- and with the ability to retrieve $p(x)$

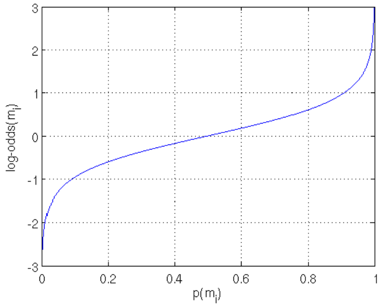
$$p(x) \quad = \quad 1 - \frac{1}{1 + \exp l(x)}$$

Courtesy: C. Stachniss

Why Log-Odds form?

- Computationally elegant for updating beliefs in log-odds form because updates are additive and avoids truncation problems that arise for probabilities close to 0 or 1

- $\ell(x) \in [-\infty, \infty]$



Occupancy Mapping in Log Odds Form

$$\frac{p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})}{1 - p(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t})} = \underbrace{\frac{p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}{1 - p(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}}_{\text{uses } \mathbf{z}_t} \underbrace{\frac{p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}{1 - p(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}}_{\text{recursive term}} \underbrace{\frac{1 - p(m_i)}{p(m_i)}}_{\text{prior}}$$

- The product turns into a sum

$$l(m_i \mid \mathbf{z}_{1:t}, \mathbf{x}_{1:t}) = \underbrace{l(m_i \mid \mathbf{z}_t, \mathbf{x}_t)}_{\text{inverse sensor model}} + \underbrace{l(m_i \mid \mathbf{z}_{1:t-1}, \mathbf{x}_{1:t-1})}_{\text{recursive term}} - \underbrace{l(m_i)}_{\text{prior}}$$

- or in short

$$l_{t,i} = \text{inv\_sensor\_model}(m_i, \mathbf{x}_t, \mathbf{z}_t) + l_{t-1,i} - l_0$$

Courtesy: C. Stachniss

Occupancy Mapping Algorithm

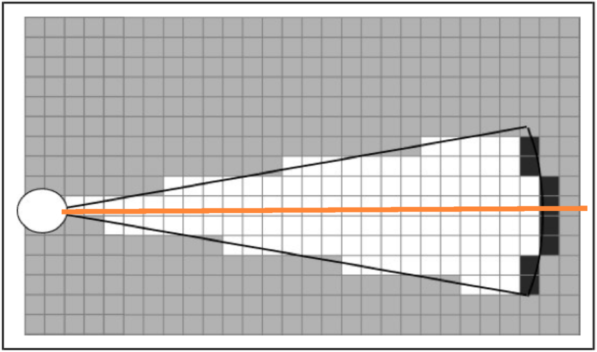
$\textbf{occupancy\_grid\_mapping}(\{l_{t-1,i}\}, \mathbf{x}_t, \mathbf{z}_t)$**:**

for all cells m_i *do*
    if m_i *in perceptual field of* $\mathbf{z}_t$ *then*
        $l_{t,i} = l_{t-1,i} + \text{inv\_sensor\_model}(m_i, \mathbf{x}_t, \mathbf{z}_t) - l_0$
    else
        $l_{t,i} = l_{t-1,i}$
    endif
endfor
return $\{l_{t,i}\}$

highly efficient, we only have to compute sums

Courtesy: C. Stachniss

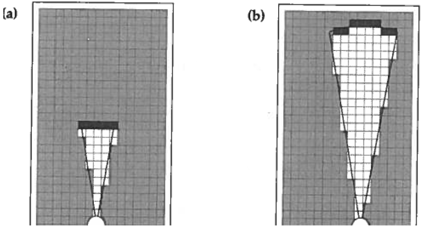
Inverse Sensor Model for Sonar Range Sensors

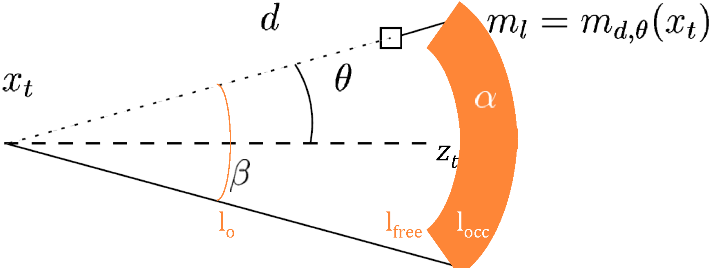


In the following, consider the cells along the optical axis (orange line)

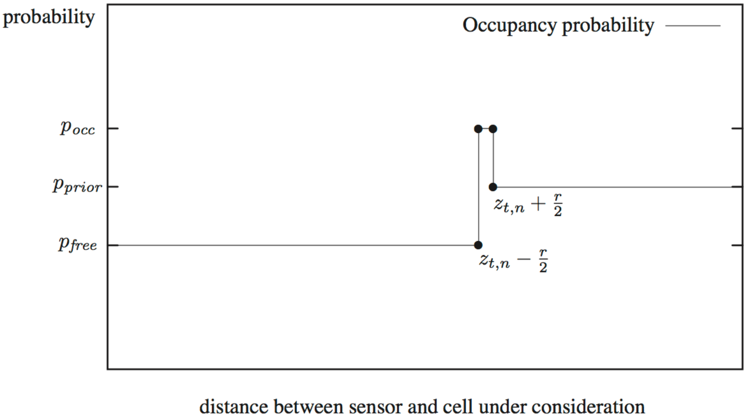
Courtesy: Thrun, Burgard, Fox

Example of a (Crude) Inverse Sensor Model





Inverse Sensor Model for Laser Range Finders



Courtesy: C. Stachniss

(Crude) Inverse Sensor Model

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

2: *Let* x_i, y_i *be the center-of-mass of* $\mathbf{m}_i$

3: $r = \sqrt{(x_i - x)^2 + (y_i - y)^2}$

4: $\phi = \text{atan2}(y_i - y, x_i - x) - \theta$

5: $k = \text{argmin}_j \, |\phi - \theta_{j,\text{sens}}|$

6: *if* $r > \min(z_{\max}, z_t^k + \alpha/2)$ *or* $|\phi - \theta_{k,\text{sens}}| > \beta/2$ *then*

7:     *return* l_0

8: *if* $z_t^k < z_{\max}$ and $|r - z_t^k| < \alpha/2$

9:     *return* l_{occ}

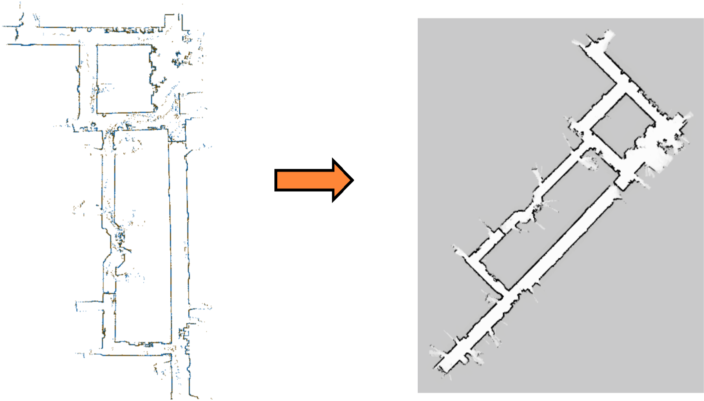
10: *if* $r \leq z_t^k$

11:     *return* l_{free}

12: *endif*

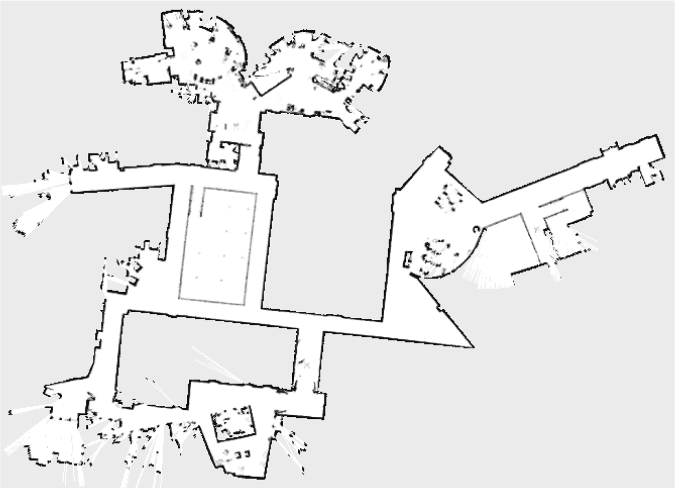
Table 9.2 A simple inverse measurement model for robots equipped with range finders. Here α is the thickness of obstacles, and β the width of a sensor beam. The values l_{occ} and l_{free} in lines 9 and 11 denote the amount of evidence a reading carries for the two different cases.

Occupancy Grids
From Laser Scans to Maps



Courtesy: D. Hähnel

Example: MIT CSAIL 3rd Floor



Courtesy: C. Stachniss

Occupancy Map Summary

- Occupancy grid maps are a popular approach to represent the environment of a mobile robot given known poses.
- In this approach each cell is considered independently from all others.
- It stores the posterior probability that the corresponding area in the environment is occupied.
- Occupancy grid maps can be learned efficiently using a probabilistic approach.

Readings

- Probabilistic Robotics: Ch. 6 and 9