Perception Models

EE468/CE468: Mobile Robotics

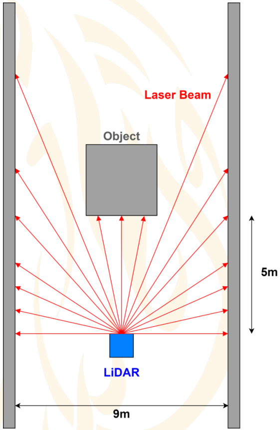
Dr. Basit Memon

Electrical and Computer Engineering
Habib University

November 13, 2023

Table of Contents

Table of Contents



- Say, we have a LiDAR installed on the robot.
- Where does the following come from?

$$p(\underbrace{z_t}_{\text{laser scan}} \mid \underbrace{x_t}_{\text{current state}}, \underbrace{m}_{\text{map}})$$

- Measurement model depends on the sensor and the map representation.

Different physical principles for different robot sensors

- **Proprioceptive Sensors:**
  - Acceleronmeters
  - Gyroscopes
  - Compass
- **Proximity Sensors:**
  - Sonar (time of flight)
  - LiDAR (tof, phase)
- **Visual Sensors:** Cameras
- GPS

Why probabilistic model? Example: Ultrasonic sensor

1. Beams reflected by obstacles
2. Beams reflected by persons / caused by crosstalk
3. Random measurements
4. Maximum range measurements

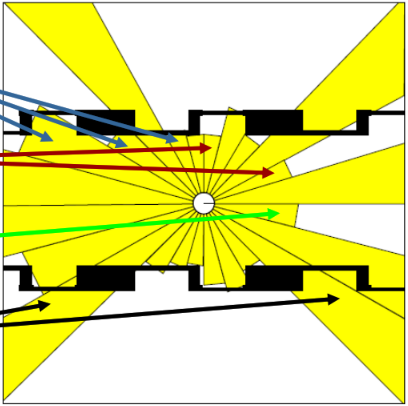


Figure: Slide Credit: Burgard

Example: Lidar Scan

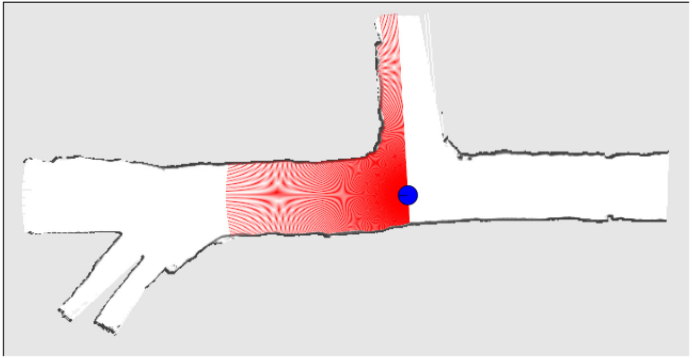


Figure: Image Credit: [3]

Table of Contents

Scan z_t at time t consists of K measurements.

- People
- Errors in m
- Approximations of posterior

- $z_t = \{z_t^1, z_t^2, \cdots, z_t^K\}$
- Assume individual measurements are independent, given the state and map. Then,

$$p(z_t | x_t, m) = \prod_{i=1}^{K} p\left(z_t^i | x_t, m\right).$$

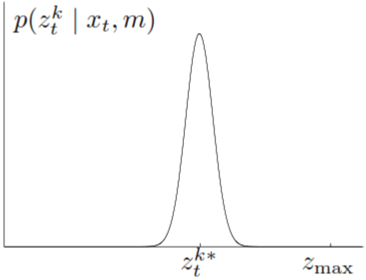
- When is this assumption invalid?

Basic measurement model for single beam, z_t^k

- Applicable to LiDAR, measuring range along beam, or Sonar, measuring within a cone.

- Probabilistic model is a mixture of four factor models, given common errors.

Factor 1: Measurement Noise



- $z_t^k = z_t^{k*} + Error$, where z_t^{k*} is correct range to nearest object.
- Sources of error are: sensor resolution, atmospheric effects, etc.
- Modeled as Gaussian with mean at z_t^{k*}, and variance, σ_{hit}^2.

$$p_{hit}(z_t^k|x_t, m) = \begin{cases} \eta \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{-\frac{1}{2}\frac{(z_t^k - z_t^{k*})^2}{\sigma_{hit}^2}} & \text{if } 0 \leq z_t^k \leq z_{max} \\ 0 & \text{otherwise,} \end{cases}$$

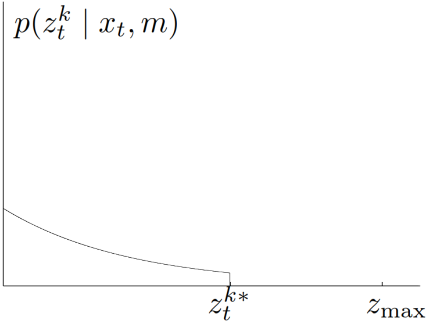
where η normalizes PDF as $z_t^k \leq z_{max}$.

How do you find z_t^{k*}?

- **Location-based Maps:** Information is indexed by location. At each location, properties of that location are stored.
- Find z_t^{k*} by ray casting, i.e. given x_t and m, what should the beam in this direction encounter?

- **Feature-based maps:** Features are indexed. At each index, properties of feature and its location are stored.
- Search for closest feature in a measurement cone.

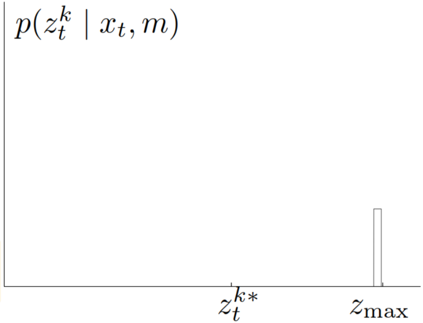
Factor 2: Unexpected Objects



- Objects not in the map, m, produce readings shorter than z_t^{k*}.
- Likelihood of sensing unexpected objects decreases with range.
- Modeled as exponential distribution, with parameter λ_{short}.

$$p_{short}(z_t^k | x_t, m) = \begin{cases} \eta \lambda_{short} e^{-\lambda_{short} z_t^k} & \text{if } 0 \le z_t^k \le z_t^{k*} \\ 0 & \text{otherwise.} \end{cases}$$

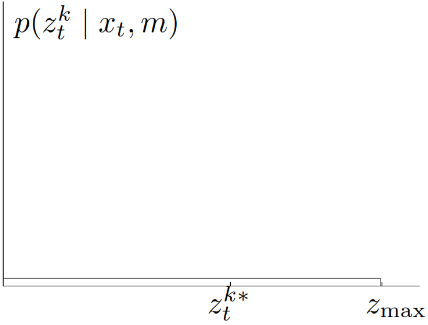
where η again normalizes PDF.



- Obstacles are missed altogether, e.g. Specular reflections, Laser absorption, etc.
- Sensor returns z_{max}.
- Modeled as point-mass distribution, centered at z_{max}.

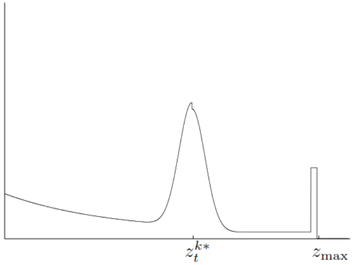
$$p_{max}(z_t^k|x_t, m) = \begin{cases} 1 & \text{if } z_t^k = z_{max} \\ 0 & \text{otherwise.} \end{cases}$$

Factor 4: Unexplainable Measurements



- Sensor gives phantom readings.
- Possible sources: Cross-talk, multiple reflections, etc.
- Modeled as uniform distribution, in range $[0, z_{max}]$.

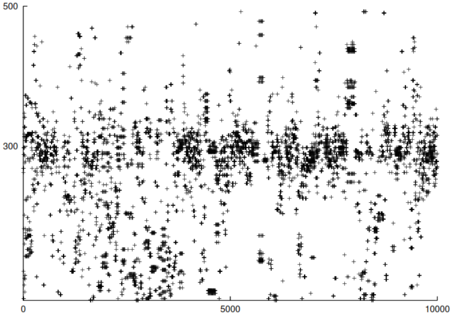
$$p_{rand}(z_t^k | x_t, m) = \begin{cases} \frac{1}{z_{max}} & \text{if } 0 \leq z_t^k < z_{max} \\ 0 & \text{otherwise.} \end{cases}$$



Weighted combination of all four factors:

$$p(z_t^k|x_t, m) = \begin{pmatrix} \alpha_{hit} \\ \alpha_{short} \\ \alpha_{max} \\ \alpha_{rand} \end{pmatrix}^T \cdot \begin{pmatrix} p_{hit}(z_t^k|x_t, m) \\ p_{short}(z_t^k|x_t, m) \\ p_{max}(z_t^k|x_t, m) \\ p_{rand}(z_t^k|x_t, m) \end{pmatrix}$$

How do we find parameters? [3, 6.3.2]



Collect lots of data and find the parameters, Θ, that maximize the log likelihood of data, i.e.

$$\arg\max_{\Theta} \log p(z|x, m, \Theta)$$

EM Algorithm for finding parameters [3, 6.3.2]

Algorithm learn_intrinsic_parameters(Z, X, m):

repeat until convergence criterion satisfied

for all z_i in Z do

$$\eta = [\, p_{\text{hit}}(z_i \mid x_i, m) + p_{\text{short}}(z_i \mid x_i, m) + p_{\text{max}}(z_i \mid x_i, m) + p_{\text{rand}}(z_i \mid x_i, m)\,]^{-1}$$

calculate z_i^*

$$e_{i,\text{hit}} = \eta\; p_{\text{hit}}(z_i \mid x_i, m)$$

$$e_{i,\text{short}} = \eta\; p_{\text{short}}(z_i \mid x_i, m)$$

$$e_{i,\text{max}} = \eta\; p_{\text{max}}(z_i \mid x_i, m)$$

$$e_{i,\text{rand}} = \eta\; p_{\text{rand}}(z_i \mid x_i, m)$$

$$z_{\text{hit}} = |Z|^{-1} \sum_i e_{i,\text{hit}}$$

$$z_{\text{short}} = |Z|^{-1} \sum_i e_{i,\text{short}}$$

$$z_{\text{max}} = |Z|^{-1} \sum_i e_{i,\text{max}}$$

$$z_{\text{rand}} = |Z|^{-1} \sum_i e_{i,\text{rand}}$$

$$\sigma_{\text{hit}} = \sqrt{\frac{1}{\sum_i e_{i,\text{hit}}} \sum_i e_{i,\text{hit}} (z_i - z_i^*)^2}$$

$$\lambda_{\text{short}} = \frac{\sum_i e_{i,\text{short}}}{\sum_i e_{i,\text{short}}\, z_i}$$

return $\Theta = \{z_{\text{hit}}, z_{\text{short}}, z_{\text{max}}, z_{\text{rand}}, \sigma_{\text{hit}}, \lambda_{\text{short}}\}$

Limitations

- Computationally intensive
  - Subsampling
  - Table lookup
- Overconfidence (Beams are not independent)
  - Subsampling
  - $p(z_t^k | x_t, m)^\alpha$ for $\alpha < 1$
- Lack of smoothness
  - Distribution can be unsmooth in cluttered environment with small obstacles.
  - Poses constraints on accuracy of approximations to ensure accuracy of posterior

Table of Contents

Other measurement models

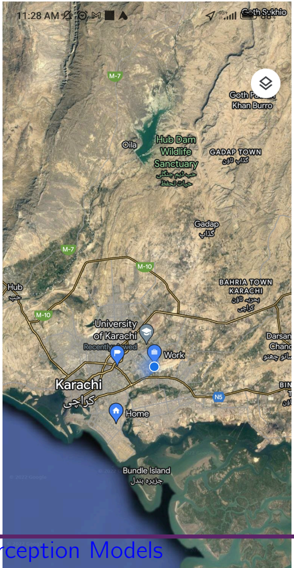
- Feature-based measurement model
  - Landmark features (Homework 3)
- Likelihood Fields (Homework 4)
- Correlation-based measurement model
  - Map Matching

Table of Contents

What is a map? [1, 9.1]

- Maps encode information about the world.
- A map is any data structure that implements long term memory of the location of measurements in raw or processed form, and makes it accessible later.


11:28 AM
Khan Burro
Hub Dam
Wildlife
Sanctuary
Oila
GADAP TOWN
Gadap
Hub
BAHRIA TOWN
KARACHI
University
of Karachi
Work
Karachi
Home
Bundle Island


11:28 AM
Khan Burro
Hub Dam
Wildlife
Sanctuary
Oila
GADAP TOWN
Gadap
Hub
BAHRIA TOWN
KARACHI
University
of Karachi
Work
Karachi
Home
Bundle Island


11:27 AM
Khan Burro
Hub Dam
Wildlife
Sanctuary
Oila
GADAP TOWN
Gadap
Hub
BAHRIA TOWN
KARACHI
University
of Karachi
Work
Karachi
Home
Bundle Island

What do we want from maps?

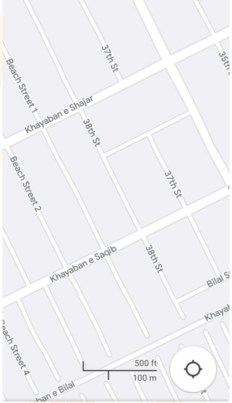
- Information: What kind? It depends on task (Localization, planning, object manipulation, etc.)
- Query: Access information from a map. Can we do it online? How quickly can we search for information?
- Updatable: Can we update it online? Can it deal with sensor and motion uncertainties?
- Memory: How much storage does it require? How does it scale with time and with more sensor data?

Table of Contents

Three common ways to represent environment [1, 9.1]

- **Field View or Metric Maps**: Explicitly represent all of space and what is known about each piece of space.
- **Topological View**: It represents only a network of paths that the robot may move along.
- **Object View**: Explicitly represents the discrete physical things in environment and what is known about them.

Metric Maps



- Distance in map = Distance in real world

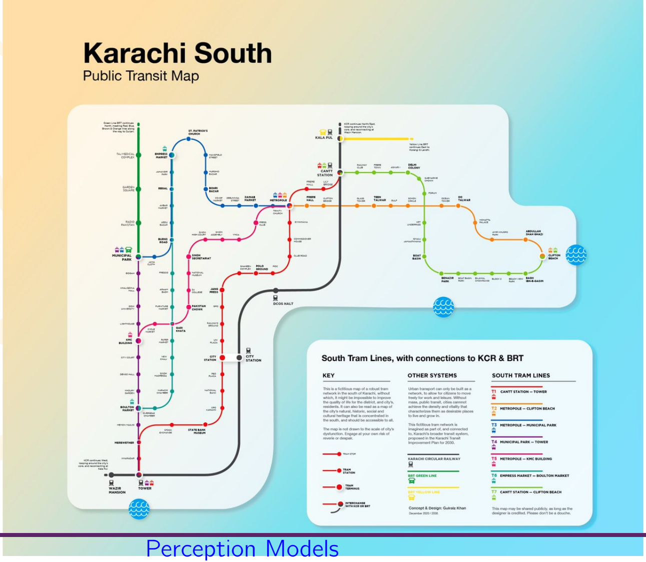

Karachi South
Public Transit Map
South Tram Lines, with connections to KCR & BRT
KEY
OTHER SYSTEMS
SOUTH TRAM LINES

Table of Contents

Example: Occupancy Grid Maps

- Field View. Decomposes the continuous space into fixed cells.
- Information: Likelihood of occupancy of each cell.
- Query: Cheap, $O(1)$
- Update: Log likelihood update equation. It can deal with noisy sensors.
- Storage: Depends on the resolution. It does not scale well with resolution or increase in size of environment.

Example: Line Maps [2, 45]

- Field View. Keeps the space continuous.
- Information: Walls, corners, lines in world.
- Update: Yes, parameters are updated.
- Storage: Less storage than grid maps as parameters of infinite lines are stored.

Example: Topological map

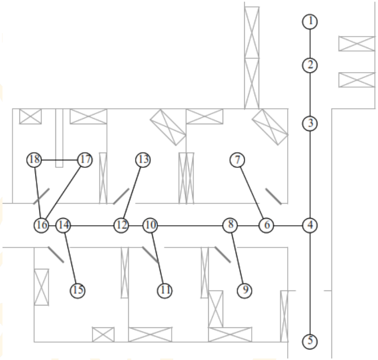


Figure: Courtesy: Autonomous Mobile Robots

- Graph where nodes are locally distinguishable places and edges indicate connectivity between nodes.
- Information: Localization, High-level navigation
- Query: Cheap graph query
- Update: Not easy. Mostly offline.
- Storage: Low memory requirements.e

Example: Landmark maps

Figure: Courtesy: Autonomous Mobile Robots

- List of landmarks.
- Information: Localization
- Query: Give closest landmark.
- Update: Easy to update
- Storage: Usually small as landmarks are few.

Example: Elevation maps or Height maps

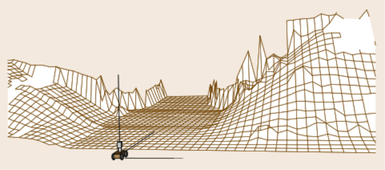


Fig. 45.8 Example elevation map built by accumulating 3-D data from a range sensor

- Each cell contains height.
- Information: Natural environment mapped from overhead.
- Query: $O(1)$
- Update: It can handle noisy data by adding filter.
- Storage: Less memory consumption compared to other maps for same purpose.

Example: Point Clouds

- Store raw range data.
- Information: Accurate surface data.
- Query: Give the closes point, $O(N)$, but N is huge.
- Update: Easy, just add point data. It cannot deal with noisy measurements.
- Storage: Depends on N

Example: Occupancy Trees (Octomaps)

OctoMap:
An Efficient Probabilistic 3D Mapping Framework Based on Octrees

Armin Hornung · Kai M. Wurm · Maren Bennewitz ·
Cyrill Stachniss · Wolfram Burgard

Figure: https://octomap.github.io/

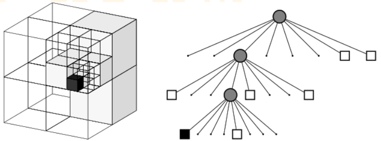


Fig. 2 Example of an octree storing free (shaded white) and occupied (black) cells. The volumetric model is shown on the left and the corresponding tree representation on the right.

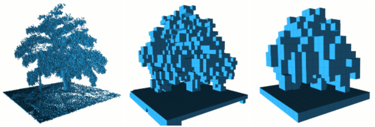


Fig. 3 By limiting the depth of a query, multiple resolutions of the same map can be obtained at any time. Occupied voxels are displayed in resolutions 0.08 m, 0.64 , and 1.28 m.

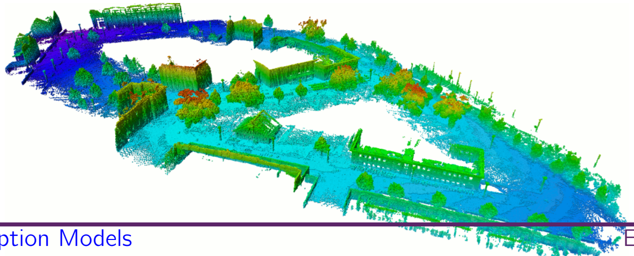


Table of Contents

[1] Alonzo Kelly.
Mobile robotics: mathematics, models, and methods.
Cambridge University Press, 2013.

[2] Bruno Siciliano and Oussama Khatib.
Springer handbook of robotics.
springer, 2016.

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