

Unit 4: Localization Part 3

Motion and Measurement Models + Grid Localization

Computer Science 4766/6912

Department of Computer Science
Memorial University of Newfoundland

June 13, 2018

1 Odometry Motion Model

2 Measurement Model

3 Grid Localization

Odometry Motion Model: $p(x_t | u_t, x_{t-1})$

- We require a general motion model to apply Bayes filter to mobile robot localization
- The motion model described here (from [Thrun et al., 2005]) is based on odometry
- Odometry gives a direct estimate of position \rightarrow unfortunately this estimate exhibits cumulative error
- We employ the difference between the current odometry pose vector \bar{x}_t and the last odometry pose vector \bar{x}_{t-1}

$$\bar{x}_t = [\bar{x}', \bar{y}', \bar{\theta}']^T \quad \bar{x}_{t-1} = [\bar{x}, \bar{y}, \bar{\theta}]^T$$

- Define the *control* or *action* as,

$$u_t = [\bar{x}_{t-1}, \bar{x}_t]^T$$

- The difference between \bar{x}_t and \bar{x}_{t-1} is a good estimate of the difference between x_t and x_{t-1}

We transform u_t into a sequence of three steps:

- An initial rotation of angle δ_{rot1}
- A translation of length δ_{trans}
- A final rotation of angle δ_{rot2}

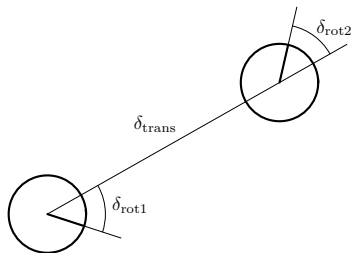


Figure 5.7 Odometry model: The robot motion in the time interval $(t - 1, t]$ is approximated by a rotation δ_{rot1} , followed by a translation δ_{trans} and a second rotation δ_{rot2} . The turns and translations are noisy.

We model the robot's motion using these three parameters; Yet the actual motion may have been quite different (e.g. rotating while translating)

Given $u_t = [\bar{x}_{t-1}, \bar{x}_t]^T$ we can derive δ_{rot1} , δ_{trans} , and δ_{rot2} via simple geometry.

COVERED ON BOARD

$$\begin{aligned}\delta_{rot1} &= \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \\ \delta_{trans} &= \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2} \\ \delta_{rot2} &= \bar{\theta}' - \bar{\theta} - \delta_{rot1}\end{aligned}$$

We assume that these *measured values* are corrupted by independent noise, such that the true values are given as follows,

$$\begin{aligned}\delta_{rot1}^{true} &= \delta_{rot1} - \epsilon_{rot1} \\ \delta_{trans}^{true} &= \delta_{trans} - \epsilon_{trans} \\ \delta_{rot2}^{true} &= \delta_{rot2} - \epsilon_{rot2}\end{aligned}$$

where the three ϵ 's are Normal random variables assumed to have zero mean.

We don't know these true parameters. But given a pair of poses x_t and x_{t-1} we can compute the parameters we would expect

$$x_t = [x', y', \theta']^T \quad x_{t-1} = [x, y, \theta]^T$$

$$\begin{aligned}\hat{\delta}_{rot1} &= \text{atan2}(y' - y, x' - x) - \theta \\ \hat{\delta}_{trans} &= \sqrt{(x - x')^2 + (y - y')^2} \\ \hat{\delta}_{rot2} &= \theta' - \theta - \hat{\delta}_{rot1}\end{aligned}$$

In order to work out the probability of a given movement, we need to know the variance of the noise processes, ϵ_{rot1} , ϵ_{trans} , and ϵ_{rot2} . We assume that the amount of noise is proportional to the amount of movement, as measured by odometry.

We utilize the equations below to model the relationship between the amount of motion and the amount of uncertainty:

$$\begin{aligned}\sigma_{rot1}^2 &= \alpha_1 \delta_{rot1}^2 + \alpha_2 \delta_{trans}^2 \\ \sigma_{trans}^2 &= \alpha_3 \delta_{trans}^2 + \alpha_4 (\delta_{rot1}^2 + \delta_{rot2}^2) \\ \sigma_{rot2}^2 &= \alpha_1 \delta_{rot2}^2 + \alpha_2 \delta_{trans}^2\end{aligned}$$

where the α parameters specify the noise characteristics of a particular robot (to be determined experimentally).

Note: This model differs from the book which relates the σ quantities to the $\hat{\delta}$ quantities.

The odometry motion model computes $p(x_t | u_t, x_{t-1})$ as follows:

- Compute δ_{rot1} , δ_{trans} , and δ_{rot2} from odometry
 - \bar{x}_t is computed by integrating wheel motion; \bar{x}_{t-1} is simply the same result from the last time step
- Compute $\hat{\delta}_{rot1}$, $\hat{\delta}_{trans}$, and $\hat{\delta}_{rot2}$ for a specific pair of x_t and x_{t-1}
- Assuming for the moment that the “hat” parameters are true, determine the probability of obtaining δ_{rot1} , δ_{trans} , and δ_{rot2}

This probability is computed as follows:

$$\begin{aligned} p_1 &= \text{prob}(\delta_{rot1} - \hat{\delta}_{rot1}, \sigma_{rot1}^2) \\ p_2 &= \text{prob}(\delta_{trans} - \hat{\delta}_{trans}, \sigma_{trans}^2) \\ p_3 &= \text{prob}(\delta_{rot2} - \hat{\delta}_{rot2}, \sigma_{rot2}^2) \\ p(x_t | u_t, x_{t-1}) &= p_1 \cdot p_2 \cdot p_3 \end{aligned}$$

where $\text{prob}(v, \sigma^2)$ returns the probability density for value v of a zero-mean Normal distribution with variance σ^2 . The product of probabilities $p_1 \cdot p_2 \cdot p_3$ is the joint probability under the assumption that the three noise processes are independent.

The shape of $p(x_t | u_t, x_{t-1})$ depends on the α_i parameters, which should be chosen to match (or exceed) the robot's actuator and odometry errors

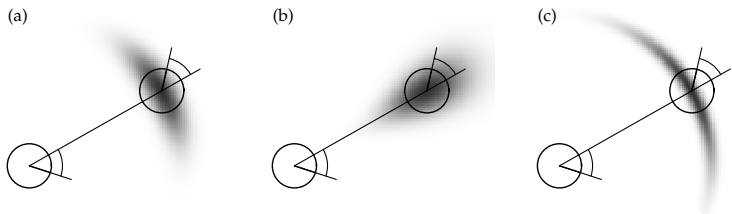


Figure 5.8 The odometry motion model, for different noise parameter settings.

Measurement Model: $p(z_t|x_t)$

We must determine the probability of observing z_t given a pose x_t . The measurement model should incorporate all of the sensor's various failure modes:

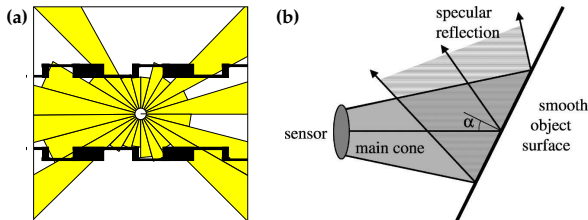


Figure 6.1 (a) Typical ultrasound scan of a robot in its environment. (b) A misreading in ultrasonic sensing. This effect occurs when firing a sonar signal towards a reflective surface at an angle α that exceeds half the opening angle of the sensor.

The measurement model depends heavily on the map, m

A particular sensor observation z_t may be composed of a number of different sensor readings. The identity of these individual readings are indicated in superscript,

$$z_t = \{z_t^1, \dots, z_t^K\}$$

where there are K individual sensors.

We assume that individual sensors are independent,

$$p(z_t|x_t) = \prod_{k=1}^K p(z_t^k|x_t)$$

The **beam model** for range finding sensors is a good model for both laser rangefinders and ultrasonic sensors. We will assume that z_t^k is a range.

$p(z_t^k | x_t)$ is computed as a mixture of four distributions. These four distributions come from the probability of the following conditions:

1. The sensor has measured the correct range, with some noise

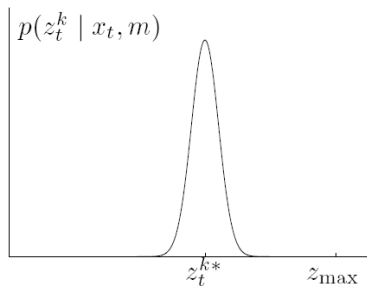
Let the true range be z_t^{k*} . The random variable z_t^k is assumed to be Normally distributed with mean z_t^{k*} and standard deviation σ_{hit} .

Thus, if we know z_t^{k*} and σ_{hit} we can calculate $p_{hit}(z_t^k | x_t)$. But how do we determine z_t^{k*} ?

We **cast a ray** from position x_t in the map in the direction that our sensor is facing. When this ray hits its first obstacle we set z_t^{k*} to the distance it has travelled.

We may wish to restrict the distribution to have zero probability after the sensor's maximum range z_{max} , as we know the sensor will not produce a larger value. If this is done it would also be necessary to normalize the distribution to ensure its sum is 1.

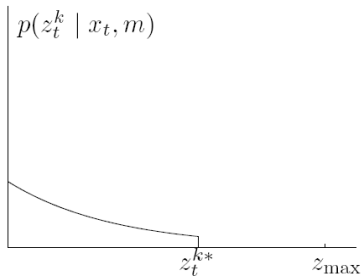
(a) Gaussian distribution p_{hit}



2. Unexpected objects

The map may omit many objects (e.g. people). The presence of such object will tend to *reduce* the reported range. The probability of an object interposing itself between the robot and a mapped part of the environment decreases with range. This probability can be modelled as an exponential distribution, truncated at the true range.

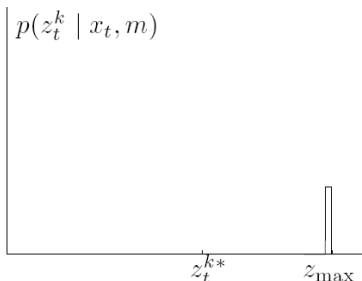
(b) Exponential distribution p_{short}



3. Missed objects

The sensor may miss an object altogether. Or the emitted beam may experience specular reflection and never return to the sensor. This possibility is modelled by a “spike” in probability at the maximum range of the sensor z_{max} .

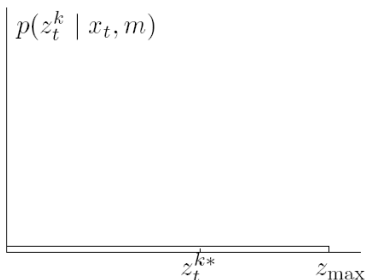
(c) Uniform distribution p_{max}



4. Unexplainable measurements

Sometimes sensors produce inexplicable measurements, for no apparent reason. We “model” such occurrences by a uniform distribution.

(d) Uniform distribution p_{rand}



The following shows all four of these densities individually:

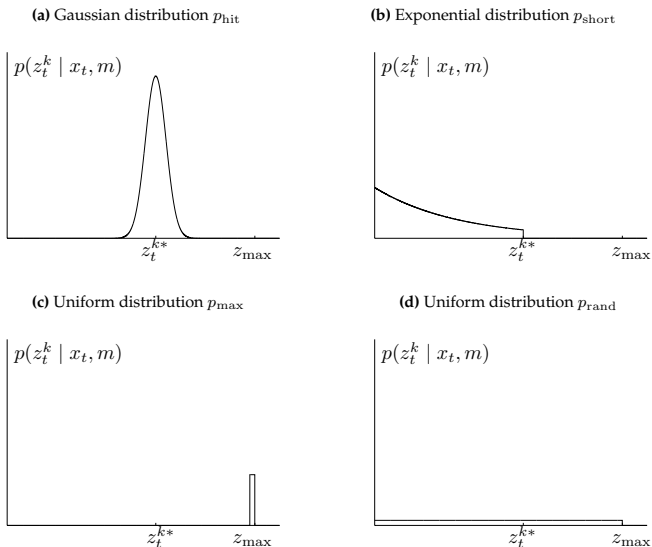


Figure 6.3 Components of the range finder sensor model. In each diagram the horizontal axis corresponds to the measurement z_t^k , the vertical to the likelihood.

A weighted combination of the four densities gives $p(z_t^k | x_t)$:

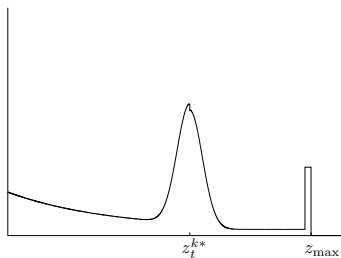


Figure 6.4 "Pseudo-density" of a typical mixture distribution $p(z_t^k | x_t, m)$.

(a) Laser scan and part of the map



(b) Likelihood for different positions

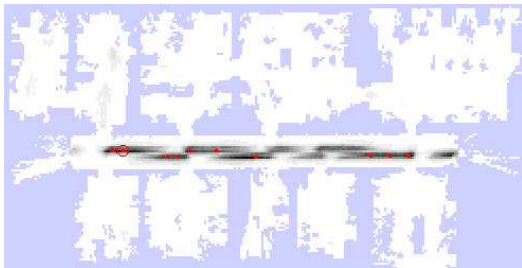


Figure 6.7 Probabilistic model of perception: (a) Laser range scan, projected into a previously acquired map m . (b) The likelihood $p(z_t | x_t, m)$, evaluated for all positions x_t and projected into the map (shown in gray). The darker a position, the larger $p(z_t | x_t, m)$.

- **Grid localization** is the direct application of Bayes filter to mobile robot localization using a discrete grid (usually 3-D) as the belief representation
 - Uses the **discrete Bayes filter** (integrals replaced with summation)
- Can be applied on a fine-grained grid or on a topological decomposition of the belief state
- Fine-grained approaches yield good results but with a high computational cost
- Has the ability to track multiple hypotheses
- The measurement model is usually defined on raw sensor values → feature extraction not required

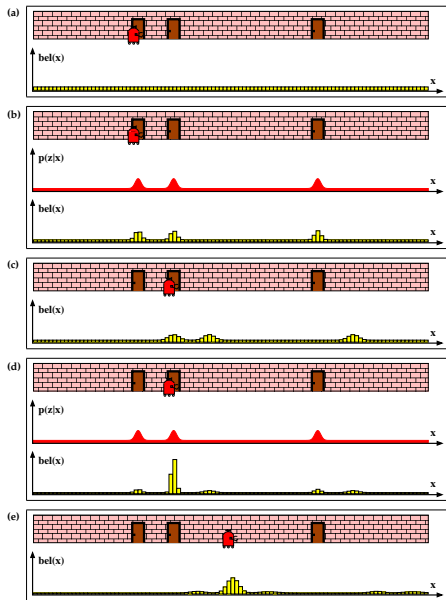


Figure 8.1 Grid localization using a fine-grained metric decomposition. Each picture depicts the position of the robot in the hallway along with its belief $bel(x_t)$, represented by a histogram over a grid.

For robots operating in the plane we require a 3-D probability cube to represent our belief in the robot's pose $x_t = [x, y, \theta]^T$

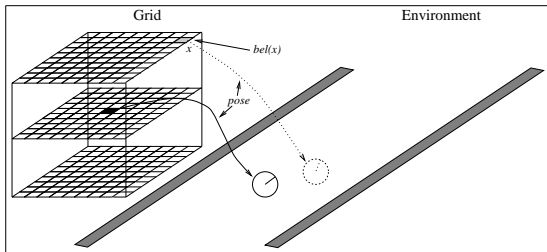


Figure 8.2 Example of a fixed-resolution grid over the robot pose variables x , y , and θ . Each grid cell represents a robot pose in the environment. Different orientations of the robot correspond to different planes in the grid (shown are only three orientations).

Each grid cell contains the probability that the robot has the corresponding pose

Grid localization requires a map so that the measurement model can be applied. The following is a typical occupancy grid map:

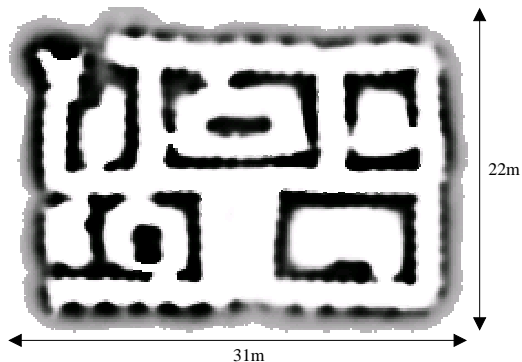


Figure 8.8 Occupancy grid map of the 1994 AAI mobile robot competition arena.

- Typical resolution of probability cube: 15 cm in x and y , 5° in θ
- With greater (i.e. more fine grained) resolution the accuracy increases, but so does the computation time
- If the resolution is very coarse, transitions between cells become increasingly unlikely for slow-moving robots (e.g. with 1 m cell size and 10 cm movements, the probability of transitioning between cells is very remote)
 - This can be accounted for by inflating the amount of noise in the motion model, but this decreases accuracy
- A naive implementation of grid localization can be quite slow:
 - Application of the motion model involves iterating over all x_t , and for each x_t one must iterate over all x_{t-1} . For a $n \times n \times n$ probability cube this is a $O(n^6)$ operation!
 - Application of the measurement model iterates over all x_t , and for each x_t one must iterate over all k sensor values. For range sensors, each scan point requires a ray casting operation.

There are a number of ways of speeding up grid localization:

- Reduce frequency of updates.
 - Note: If we reduce the frequency of applications of the motion model, we must integrate the motion for a longer period in between updates.
- Decrease grid resolution.
 - Problems: decreases accuracy, may require exaggerated noise in motion model.
- **Selective updating:**
 - Based on probability threshold: Update only those cells of the belief cube which have a probability greater than some fixed threshold.
 - Problem: if localization errors occur (e.g. the kidnapped robot problem) the robot may end up in an area which is not updated.
 - Apply updates in local neighbourhood: When updating a cell, restrict the space of possible prior poses to those centred around that cell (*other variants on this scheme are possible*)

The measurement model can also be accelerated:

- Pre-caching results during initialization:
 - For each possible 3D pose, cast rays for each sensor to determine beforehand the expected ranges for each pose. Store these ranges in a table for later look-up.
 - Pre-compute the p.d.f.'s for all possible ranges.
- Reduce the number of sensors used.

An example of grid localization. It is important to note that the probabilities in the grid have been projected from 3-D to 2-D. The distribution of belief over θ is very important, but it is not shown below.

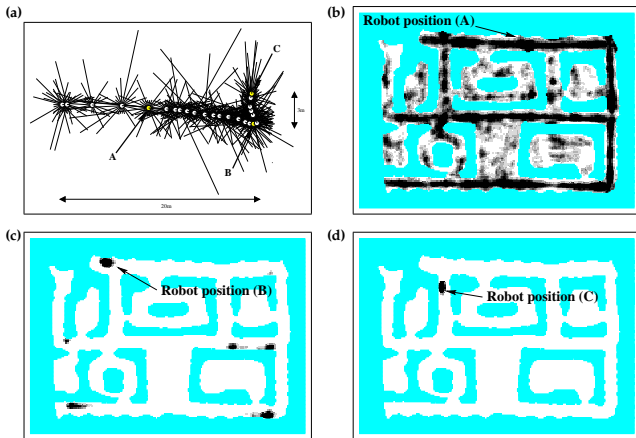


Figure 8.9 (a) Data set (odometry and sonar range scans) collected in the environment shown in Figure 8.8. This data set is sufficient for global localization using the grid localization. The beliefs at the points marked "A," "B" and "C" are shown in (b), (c), and (d).

References



Thrun, S., Burgard, W., and Fox, D. (2005).
Probabilistic Robotics.
MIT Press.