Chapter 2

Basic Techniques

This chapter explains two techniques which are frequently used throughout this thesis. First, we will introduce the concept of particle filters. A particle filter is a recursive Bayesian technique for estimating the state of a dynamic system. We then explain the ideas of grid maps and "mapping with known poses". Note that elementary laws in the context of probability theory can be found in the Appendix A.1 of this thesis.

2.1 Introduction to Particle Filters

A particle filter is a nonparametric implementation of the Bayes filter and is frequently used to estimate the state of a dynamic system. The key idea is to represent a posterior by a set of hypotheses. Each hypothesis represents one potential state the system might be in. The state hypotheses are represented by a set S of N weighted random samples

$$S = \left\{ \left\langle s^{[i]}, w^{[i]} \right\rangle \mid i = 1, \ldots, N \right\}, \tag{2.1}$$

where $s^{[i]}$ is the state vector of the i-th sample and $w^{[i]}$ the corresponding importance weight. The weight is a non-zero value and the sum over all weights is 1. The sample set represents the distribution

$$p(x) = \sum_{i=1}^{N} w_i \cdot \delta_{s^{[i]}}(x), \tag{2.2}$$

where $\delta_{s^{[i]}}$ is the Dirac function in the state $s^{[i]}$ of the i-th sample. Such set S of samples can be used to approximate arbitrary distributions. The samples are drawn from the distribution they should approximate. To illustrate such an approximation, Figure 2.1 depicts two distributions and their corresponding sample sets. In general, the more

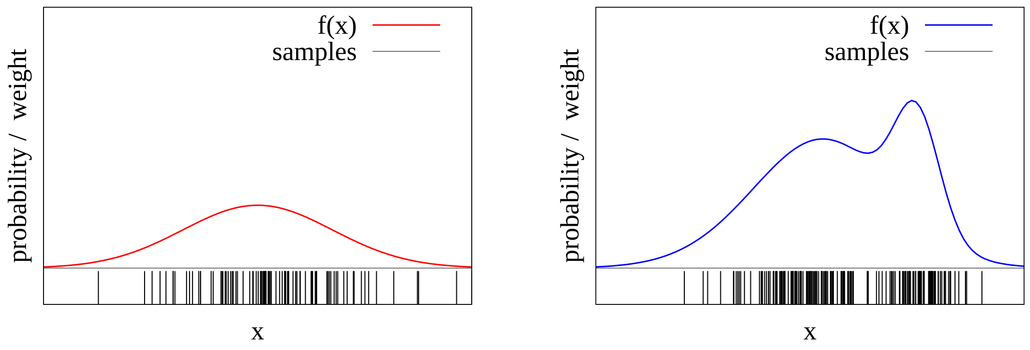

Figure 2.1: Two functions and their approximations by samples with uniform weights. The samples are illustrated by the vertical bars below the two functions.

samples that are used, the better the approximation. The ability to model multi-modal distributions by the set of samples is an advantage compared to a series of other filters. The Kalman filter [Kalman, 1960], for example, is restricted to Gaussian distributions.

Whenever we are interested in estimating the state of a dynamic system over time, we can use the particle filter algorithm. The idea of this technique is to represent the distribution at each point in time by a set of samples, also called particles. The particle filter algorithm allows us to recursive estimate the particle set S_t based on the estimate S_{t-1} of the previous time step. The *sampling importance resampling* (SIR) particle filter can be summarized with the following three steps:

1. **Sampling:** Create the next generation S'_t of particles based on the previous set S_{t-1} of samples. This step is also called sampling or drawing from the proposal distribution.

2. **Importance Weighting:** Compute an importance weight for each sample in the set S'_t.

3. **Resampling:** Draw N samples form the set S'_t. Thereby, the likelihood to draw a particle is proportional to its weight. The new set S_t is given by the drawn particles.

In the following, we explain these three steps in more detail. In the first step, we draw samples in order to obtain the next generation of particles for the next time step. In general, the true probability distribution to sample particles from is not known or not in a suitable form for sampling. We show that it is possible to draw samples from a different distribution than the one we want to approximate. This technique is known as *importance sampling*.

We are faced with the problem of computing the expectation that $x \in A$, where A is a region. In general, the expectation $E_p[f(x)]$ of a function f is defined as

$$E_p[f(x)] \;=\; \int p(x) \cdot f(x)\, dx. \tag{2.3}$$

Let B be a function which returns 1 if its argument is true and 0 otherwise. We can express the expectation that $x \in A$ by

$$\begin{aligned} E_p[B(x \in A)] &= \int p(x) \cdot B(x \in A)\, dx & (2.4)\\ &= \int \frac{p(x)}{\pi(x)} \cdot \pi(x) \cdot B(x \in A)\, dx, & (2.5) \end{aligned}$$

where π is a distribution for which we require that

$$p(x) > 0 \;\;\Rightarrow\;\; \pi(x) > 0. \tag{2.6}$$

Thus, we can define a weight $w(x)$ as

$$w(x) \;=\; \frac{p(x)}{\pi(x)}. \tag{2.7}$$

This weight w is used to account for the differences between p and the π. This leads to

$$\begin{aligned} E_p[B(x \in A)] &= \int \pi(x) \cdot w(x) \cdot B(x \in A)\, dx & (2.8)\\ &= E_\pi[w(x) \cdot B(x \in A)]. & (2.9) \end{aligned}$$

Let us consider again the sample-based representations and suppose the sample are drawn from π. By counting all the particles that fall into the region A, we can compute the integral of π over A by the sum over samples

$$\int_A \pi(x)\; dx \;\approx\; \frac{1}{N} \cdot \sum_{i=1}^{N} B(s^{[i]} \in A). \tag{2.10}$$

If we consider the weights in this computation, we can compute the integral over p as

$$\int_A p(x)\; dx \;\approx\; \sum_{i=1}^{N} w^{[i]} \cdot B(s^{[i]} \in A). \tag{2.11}$$

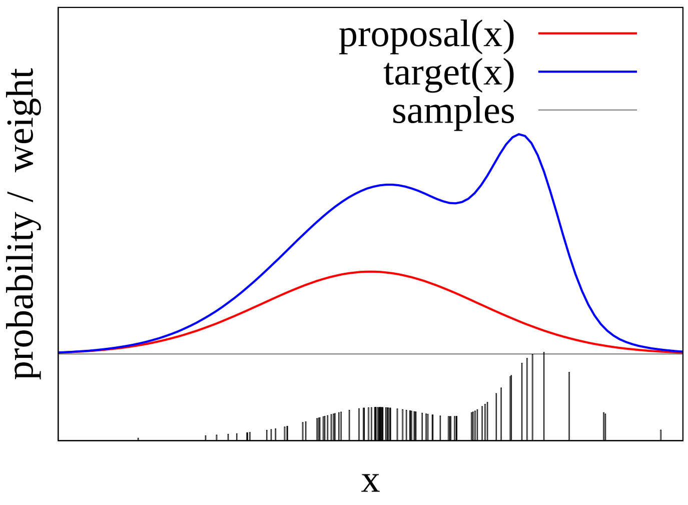

Figure 2.2: The goal is to approximate the target distribution by samples. The samples are drawn from the proposal distribution and weighted according to Eq. (2.13). After weighting, the resulting sample set is an approximation of the target distribution.

It can be shown, that the quality of the approximation improves the more samples that are used. For an infinite set of samples, the sum over the samples converges to the integral

$$\lim_{N\to\infty} \sum_{i=1}^{N} w^{[i]} \cdot B(s^{[i]} \in A) \quad = \quad \int_A p(x)\, dx. \tag{2.12}$$

Let p be the probability distribution which is not in a suitable form for sampling and π the one we actually sample from. In the context of importance sampling, p is typically called the *target distribution* and π the *proposal distribution*.

This derivation tells us that we can sample from an arbitrary distribution π which fulfills Eq. (2.6) to approximate the distribution p by assigning an importance weight to each sample according to Eq. (2.7). This condition is needed to ensure that a state which might be sampled from p does not have zero probability under π. An example that depicts a weighted set of samples in case the proposal is different from the target distribution is shown in Figure 2.2. Note that the importance sampling principle requires that we can point-wise evaluate the target distribution. Otherwise, the computation of the weights would be impossible.

Let $p(s_{1:t} \mid d)$ be the posterior to estimate, where d stands for all the data or background information. The importance weighting performed in Step 2 of the particle filter implementation (see Page 28) accounts for the fact one draws from the proposal π by setting the weight of each particle to

$$w_t^{[i]} \quad = \quad \eta \cdot \frac{p(s_{1:t}^{[i]} \mid d)}{\pi(s_{1:t}^{[i]} \mid d)}, \tag{2.13}$$

where η is the normalizer that ensures that the sum over all weights is 1.

The resampling step within a particle filter removes particles with a low importance weight and replaces them by particles with a high weight. After resampling, the weights are set to $1/N$ because by drawing according to the importance weight, one replaces "likelihoods" by "frequencies".

Resampling is needed since we use only a finite number of samples to approximate the target distribution. Without resampling, typically most particles would represent states with a low likelihood after some time and the filter would loose track of the "good" hypotheses. On the one hand, this fact makes resampling important, on the other hand removing samples from the filter can also be problematic. In practice, it can happen that samples are replaced even if they are close to the correct state. This can lead to the so-called particle depletion or particle deprivation problem [Doucet, 1998, Doucet *et al.*, 2001, van der Merwe *et al.*, 2000].

To reduce the risk of particle depletion, one can apply low-variance resampling. This technique does not draw the particles independently of each other in the resampling step. Instead of generating N random numbers to select N samples, the approach uses only a single random number to choose the first particle. The others are drawn depended on the first draw but still with a probability proportional to the individual weights. As a result, the particle set does not change during a resampling in case the weights are uniformly distributed. A detailed explanation on low-variance resampling as well as on particle filters in general can be found in [Thrun *et al.*, 2005]. The complete particle filter algorithm is listed in Algorithm 2.1.

2.1.1 Mobile Robot Localization using Particle Filters

In the context of mobile robotics, particle filters are often used to track the position of the robot. Since this technique is used in this thesis, we briefly illustrate the most important facts of Monte-Carlo localization [Dellaert *et al.*, 1998]. In this scenario, the state vector s is the pose of the vehicle. Mostly, the motion estimate of the robot resulting from odometry is used to compute the proposal distribution in Step 1. The so-called motion model $p(x_t \mid x_{t-1}, u_{t-1})$ is used to draw the next generation of particles. In this case, the importance weight $w_t^{[i]}$ of the i-th sample has to be computed based on the observation likelihood $p(z_t \mid m, x_t^{[i]})$ of the most recent sensor observation z_t given a map m of the environment and the corresponding pose of the particle. This becomes

Algorithm 2.1 The particle filter algorithm

```
Input: Sample set $S_{t-1}$ and the data $d$.
 1: $S'_t = \emptyset$
 2: for i=1 to N do
 3:     draw $\hat{s} \sim \pi(s_t \mid s_{t-1}^{[i]}, d)$
 4:     $\hat{w} = \eta \cdot \left[p(\hat{s} \mid s_{t-1}^{[i]}, d)\right] \cdot \left[\pi(\hat{s} \mid s_{t-1}^{[i]}, d)\right]^{-1}$ // where $\eta$ is a normalizer
 5:     $S'_t = S'_t + \langle \hat{s}, \hat{w} \rangle$
 6: end
 7: $S_t = \emptyset$
 8: for j=1 to N do
 9:     draw a sample $s_t^{[i]}$ from $S'_t$. Thereby, $s_t^{[i]}$ is drawn with probability $w_t^{[i]}$
10:     $S_t = S_t + \left\langle s_t^{[i]}, 1/N \right\rangle$
11: end
12: return $S_t$
```

clear by considering the following derivations. We can transform the full posterior $p(x_{1:t} \mid m, z_{1:t}, u_{1:t-1})$ and obtain a recursive formula

$$\begin{aligned} p(x_{1:t} \mid m, z_{1:t}, u_{1:t-1}) &\overset{\text{Bayes' rule}}{=} \eta \cdot p(z_t \mid m, x_{1:t}, z_{1:t-1}, u_{1:t-1}) \\ &\quad \cdot p(x_{1:t} \mid m, z_{1:t-1}, u_{1:t-1}) \qquad (2.14) \\ &\overset{\text{Markov}}{=} \eta \cdot p(z_t \mid m, x_t) \\ &\quad \cdot p(x_{1:t} \mid m, z_{1:t-1}, u_{1:t-1}) \qquad (2.15) \\ &\overset{\text{product rule}}{=} \eta \cdot p(z_t \mid m, x_t) \\ &\quad \cdot p(x_t \mid m, x_{1:t-1}, z_{1:t-1}, u_{1:t-1}) \\ &\quad \cdot p(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-1}) \qquad (2.16) \\ &\overset{\text{Markov}}{=} \eta \cdot p(z_t \mid m, x_t) \cdot p(x_t \mid x_{t-1}, u_{t-1}) \\ &\quad \cdot p(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-2}), \qquad (2.17) \end{aligned}$$

where η is the normalizer resulting from Bayes' rule. Under the Markov assumption, we can transform the proposal as

$$\begin{aligned} \pi(x_{1:t} \mid m, z_{1:t}, u_{1:t}) &= \pi(x_t \mid m, x_{t-1}, z_t, u_{t-1}) \\ &\quad \cdot \pi(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-2}). \qquad (2.18) \end{aligned}$$

The computation of the weights needs to be done according to Eq. (2.13). In the sequel, we drop the normalizer that ensures that all weights sum up to 1. This leads to

$$
\begin{aligned}
w_t &= \frac{p(x_{1:t} \mid m, z_{1:t}, u_{1:t-1})}{\pi(x_{1:t} \mid m, z_{1:t}, u_{1:t-1})} && (2.19)\\
&= \frac{\eta \cdot p(z_t \mid m, x_t) \cdot p(x_t \mid x_{t-1}, u_{t-1})}{\pi(x_{1:t} \mid m, z_{1:t}, u_{1:t-1})} \cdot p(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-2}) && (2.20)\\
&= \frac{\eta \cdot p(z_t \mid m, x_t) \cdot p(x_t \mid x_{t-1}, u_{t-1})}{\pi(x_t \mid m, x_{t-1}, z_t, u_{t-1})} \cdot \underbrace{\frac{p(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-2})}{\pi(x_{1:t-1} \mid m, z_{1:t-1}, u_{1:t-2})}}_{w_{t-1}} && (2.21)\\
&= \frac{\eta \cdot p(z_t \mid m, x_t) \cdot p(x_t \mid x_{t-1}, u_{t-1})}{\pi(x_t \mid m, x_{t-1}, z_t, u_{t-1})} \cdot w_{t-1}. && (2.22)
\end{aligned}
$$

If we choose the motion model as the proposal, we obtain for the i-the sample

$$
\begin{aligned}
w_t^{[i]} &= \frac{\eta \cdot p(z_t \mid m, x_t^{[i]}) \cdot p(x_t \mid x_{t-1}^{[i]}, u_{t-1})}{p(x_t \mid x_{t-1}^{[i]}, u_{t-1})} \cdot w_{t-1}^{[i]} && (2.23)\\
&= \eta \cdot p(z_t \mid m, x_t^{[i]}) \cdot w_{t-1}^{[i]} && (2.24)\\
&\propto p(z_t \mid m, x_t^{[i]}) \cdot w_{t-1}^{[i]}. && (2.25)
\end{aligned}
$$

Since the resampling step resets the weights of the whole set by $1/N$, we can ignore the weight of the previous time step and obtain

$$
w_t^{[i]} \propto p(z_t \mid m, x_t^{[i]}). \qquad (2.26)
$$

This derivation shows that by choosing the motion model to draw the next generation of particles, we have to use the observation likelihood $p(z_t \mid m, x_t)$ to compute the individual weights.

To summarize this section, particle filters are a nonparametric implementations of the recursive Bayes filter. They use a set of weighted samples and can represent arbitrary distributions. The samples are drawn from a proposal distribution. After determining the importance weights which account for the fact that the target distribution is different from the proposal distribution, the resampling step replaces particles with a low weight by particles with a high importance weight.

Throughout this thesis, we apply particle filters to solve the simultaneous localization and mapping problem. Furthermore, we apply them in the context of information gain-based exploration and to localize a mobile robot in dynamically changing environments.

2.2 Grid Maps

There exist different types of models for representing the environment which are frequently used in mobile robotics. The most common ones are feature maps, geometric maps, and grid maps. A feature map stores a set of features detected in the environment. Typical features are lines and corners when proximity sensors are used. Other possibilities are visual features based on the scale invariant feature transform (SIFT) [Lowe, 1999] whenever a camera is used to perceive the environment. For each feature, these maps store the feature information together with a coordinate and eventually an uncertainty measure. This can be realized by a list of features or by using more efficient data structures like KD-trees [Friedman *et al.*, 1977, Bentley, 1980].

Geometric maps represent all obstacles detected by the robot as geometric objects, like circles or polygons. This kind of representation is comparably compact and needs only few memory resources.

Throughout this thesis, we use grid maps to model the environment. Grid maps discretize the environment into so-called grid cells. Each cell stores information about the area it covers. Most frequently used are occupancy grid maps that store for each cell a single value representing the probability that this cell is occupied by an obstacle. The advantage of grids is that they do not rely on predefined features which need to be extracted from sensor data. Furthermore, they offer a constant time access to grid cells and provide the ability to model unknown (unobserved) areas, which is an important feature in the context of exploration. However, they have the disadvantages of discretization errors and of requiring a lot of memory resources.

In this section, we first introduce the occupancy mapping algorithm, developed by Moravec and Elfes [1985]. Afterwards, we briefly describe a variant called reflection probability maps. Both approaches are also referred to as "mapping with known poses."

2.2.1 Occupancy Probability Mapping

Grid maps discretize the environment into equally sized cells. Each cell represents the area of the environment it covers. It is assumed that each cell is either free or occupied by an obstacle. Occupancy grids store for each cell c a probability $p(c)$ of being occupied by an obstacle. In the following, we will derive the map update algorithm introduced by Moravec and Elfes which computes the occupancy probability $p(m)$ for the grid map m.

The algorithm takes into account a sequence of sensor observations $z_{1:t}$ obtained by the robot at the positions $x_{1:t}$ and seeks to maximize the occupancy probability for the grid map. One assumption in the algorithm of Moravec and Elfes is that the different cells are independent. Therefore, the probability of a map m is given by the product over the probabilities of the individual cells

$$p(m) = \prod_{c \in m} p(c). \tag{2.27}$$

In the following, we concentrate on the estimation of the occupancy probability of the individual cells $c \in m$. By applying Bayes' rule using $x_{1:t}$ and $z_{1:t-1}$ as background knowledge, we obtain

$$p(c \mid x_{1:t}, z_{1:t}) = \frac{p(z_t \mid c, x_{1:t}, z_{1:t-1}) \cdot p(c \mid x_{1:t}, z_{1:t-1})}{p(z_t \mid x_{1:t}, z_{1:t-1})}. \tag{2.28}$$

We assume that z_t is independent from $x_{1:t-1}$ and $z_{1:t-1}$. This leads to

$$p(c \mid x_{1:t}, z_{1:t}) = \frac{p(z_t \mid c, x_t) \cdot p(c \mid x_{1:t}, z_{1:t-1})}{p(z_t \mid x_{1:t}, z_{1:t-1})}. \tag{2.29}$$

We apply Bayes' rule for the term $p(z_t \mid c, x_t)$ in Eq. (2.29) and obtain

$$p(z_t \mid c, x_t) = \frac{p(c \mid x_t, z_t) \cdot p(z_t \mid x_t)}{p(c \mid x_t)}. \tag{2.30}$$

We can now combine Eq. (2.30) and Eq. (2.29). Let us furthermore assume that x_t does not carry any information about c if there is no observation z_t. This leads to

$$p(c \mid x_{1:t}, z_{1:t}) = \frac{p(c \mid x_t, z_t) \cdot p(z_t \mid x_t) \cdot p(c \mid x_{1:t-1}, z_{1:t-1})}{p(c) \cdot p(z_t \mid x_{1:t}, z_{1:t-1})}. \tag{2.31}$$

If we exploit the fact that each cell is a binary variable, we can derive the following equation in an analogous way

$$p(\neg c \mid x_{1:t}, z_{1:t}) = \frac{p(\neg c \mid x_t, z_t) \cdot p(z_t \mid x_t) \cdot p(\neg c \mid x_{1:t-1}, z_{1:t-1})}{p(\neg c) \cdot p(z_t \mid x_{1:t}, z_{1:t-1})}. \tag{2.32}$$

By dividing Eq. (2.31) by Eq. (2.32), we obtain

$$\frac{p(c \mid x_{1:t}, z_{1:t})}{p(\neg c \mid x_{1:t}, z_{1:t})} = \frac{p(c \mid x_t, z_t) \cdot p(\neg c) \cdot p(c \mid x_{1:t-1}, z_{1:t-1})}{p(\neg c \mid x_t, z_t) \cdot p(c) \cdot p(\neg c \mid x_{1:t-1}, z_{1:t-1})}. \tag{2.33}$$

Finally, we use the fact that $p(\neg c) = 1 - p(c)$ which yields

$$\begin{aligned} &\frac{p(c \mid x_{1:t}, z_{1:t})}{1 - p(c \mid x_{1:t}, z_{1:t})} = \\ &\quad \frac{p(c \mid x_t, z_t)}{1 - p(c \mid x_t, z_t)} \cdot \frac{1 - p(c)}{p(c)} \cdot \frac{p(c \mid x_{1:t-1}, z_{1:t-1})}{1 - p(c \mid x_{1:t-1}, z_{1:t-1})}. \end{aligned} \tag{2.34}$$

If we define

$$\mathrm{Odds}(x) \quad = \quad \frac{p(x)}{1-p(x)}, \tag{2.35}$$

Eq. (2.34) turns into

$$\begin{aligned}&\mathrm{Odds}(c \mid x_{1:t}, z_{1:t}) = \\ &\qquad \mathrm{Odds}(c \mid x_t, z_t) \cdot \mathrm{Odds}(c)^{-1} \cdot \mathrm{Odds}(c \mid x_{1:t-1}, z_{1:t-1}).\end{aligned} \tag{2.36}$$

This equation has a recursive structure similar to that of a recursive Bayesian update scheme. The corresponding $\log \mathrm{Odds}$ representation of Eq. (2.36) is given by

$$\begin{aligned}&\log \mathrm{Odds}(c \mid x_{1:t}, z_{1:t}) = \\ &\quad \log \mathrm{Odds}(c \mid z_t, x_t) \\ &\quad -\log \mathrm{Odds}(c) \\ &\quad +\log \mathrm{Odds}(c \mid x_{1:t-1}, z_{1:t-1}).\end{aligned} \tag{2.37}$$

The usage of the $\log \mathrm{Odds}$ notation has advantage that it can be computed efficiently. It is only necessary to compute a sum in order to update a cell based on sensory input. To recover the occupancy probability from the Odds representation given in Eq. (2.36), we use the following formula which can easily be derived from Eq. (2.35):

$$p(x) \quad = \quad \frac{\mathrm{Odds}(x)}{1+\mathrm{Odds}(x)} \tag{2.38}$$

This leads to the following *occupancy update formula*

$$\begin{aligned}&p(c \mid x_{1:t}, z_{1:t}) = \\ &\quad \left[1 + \frac{(1-p(c \mid x_t, z_t))}{p(c \mid x_t, z_t)} \cdot \frac{p(c)}{(1-p(c))} \cdot \frac{1-p(c \mid x_{1:t-1}, z_{1:t-1})}{p(c \mid x_{1:t-1}, z_{1:t-1})}\right]^{-1}.\end{aligned} \tag{2.39}$$

Eq. (2.39) tells us how to update our belief $p(c \mid x_{1:t}, z_{1:t})$ about the occupancy probability of a grid cell given sensory input. In practice, one often assumes that the occupancy prior is 0.5 for all cells so that $\frac{p(c)}{(1-p(c))}$ can be removed from the equation.

It remains to describe how to compute the occupancy probability $p(c \mid x_t, z_t)$ of a grid cell given a *single* observation z_t and the corresponding pose x_t of the robot. This quantity strongly depends on the sensor of the robot and has to be defined manually for each type of sensor.

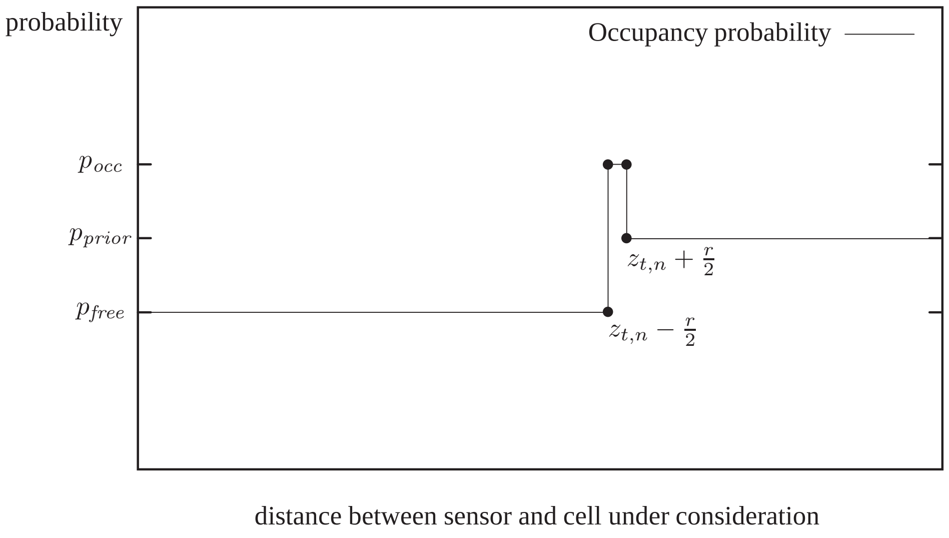

Figure 2.3: Sensor model for a laser range finder. It depicts the probability that a cell is occupied depending on the distance of that cell from the laser sensor.

2.2.2 Sensor Model for a Laser Range Finder

In case a laser range finder is used, a quite simplistic model can be applied. Each cell c that is covered by the n-th beam $z_{t,n}$ of the observation z_t and whose distance to the sensor is shorter than the measured one, is supposed to be unoccupied. The cell in which the beam ends is supposed to be occupied. The function $dist(x_t, c)$ refers to the distance between the sensor and the center of the cell c. This can be formulated

$$p(c \mid z_{t,n}, x_t) = \begin{cases} p_{prior}, & z_{t,n} \text{ is a maximum range reading} \\ p_{prior}, & c \text{ is not covered by } z_{t,n} \\ p_{occ}, & |z_{t,n} - dist(x_t, c)| < r \\ p_{free}, & z_{t,n} \geq dist(x_t, c), \end{cases} \tag{2.40}$$

where r is the resolution of the grid map. Furthermore, it must hold $0 \leq p_{free} < p_{prior} < p_{occ} \leq 1$. Figure 2.3 depicts an example for such a sensor model for laser range finder data.

2.2.3 Sensor Model for a Sonar Sensor

In case a sonar sensor is used, the sensor model is slightly more complicated, since the sensor is not a beam sensor and the observations are more noisy than the ones of

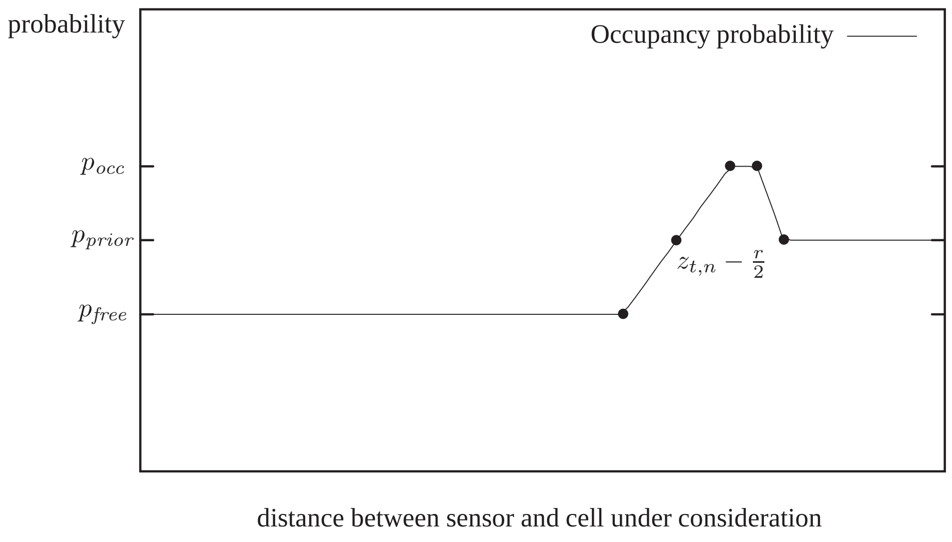

Figure 2.4: Probability that a cell on the optical axis of the sensor is occupied depending on the distance of that cell from the sensor.

a laser range finder. In practice, one typically uses a mixture of three functions to express the model. First, the influence of an observation (which is represented by the difference between p_{prior} and p_{occ} as well as between p_{prior} and p_{free}) decreases with the measured distance.

Second, the proximity information of a sonar is substantially affected by noise. Therefore, one typically uses a piecewise linear function to model a smooth transition from p_{free} to p_{occ} as illustrated in Figure 2.4.

Finally, the sonar sensor should not be modeled as a beam sensor, since it sends out a conic signal. The accuracy of an observation decreases with the angular distance between the cell under consideration and the optical axis of the observation. This is expressed by the derivation from the prior and is typically modeled using a Gaussian with zero mean. Therefore, it is maximal along the optical axis and decreases the bigger the angular distance form the optical axis is.

Two examples for a resulting model are depicted in Figure 2.5. It shows two three-dimensional plots of the resulting occupancy probabilities for a measurement of 2 m (left image) and 2.5 m (right image). In this figure, the optical axis of the sensor cone was identical with the x-axis and the sensor was placed in the origin of the coordinate frame. As can be seen, the occupancy probability is high for cells whose distance to x_t is close to $z_{t,n}$. It decreases for cells with shorter distance than $z_{t,n}$ as well as with

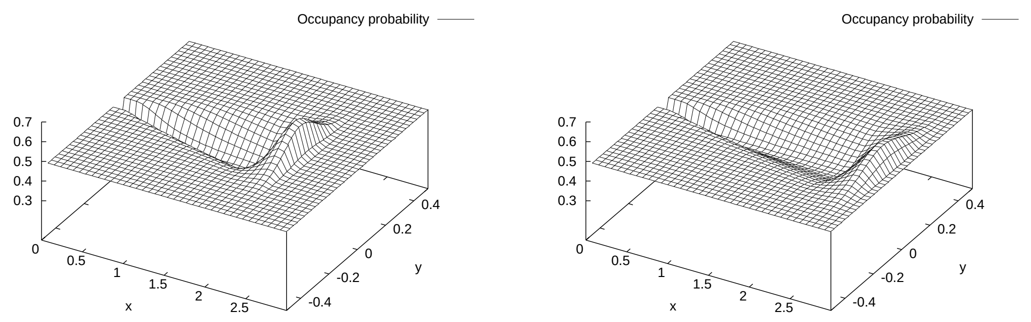

Figure 2.5: Occupancy probability introduced by a single ultrasound measurement of $z_{t,n} = 2.0m$ (left image) and $z_{t,n} = 2.5m$ (right image).

increasing values of the angular distance.

Figure 2.6 depicts the mapping process for a sequence of observations recorded with an iRobot B21r robot. The first row shows a map was built from a sequence of previous ultrasound scans. Afterwards the robot perceived a series of 18 ultrasound scans each consisting of 24 measurements. The occupancy probabilities for these 18 scans are depicted in the rows from 2 to 7. The occupancy probability grid obtained by integrating the individual observations into the map is shown in the last row of this figure. As can be seen, the belief converges to a representation of the corridor structure in which the scans where recorded.

2.2.4 Reflection Probability Mapping

Beside occupancy probability grids, there exist alternative realization of grid maps. A frequently used model is the so-called reflection probability map or counting model. In contrast to occupancy grid maps, they store for each cell a reflection probability value. This value provides the probability that a measurement covering the cell is reflected. Note that the occupancy model and the counting model are similar but not identical.

In this model, we are interested in computing the most likely reflection probability map m^* given the observations and poses of the robot.

$$m^* \quad = \quad \underset{m}{\operatorname{argmax}}\, p(m \mid x_{1:t}, z_{1:t}) \tag{2.41}$$

By series of mathematical transformations (see [Burgard, 2005] for the details), one can derive that the most likely map m^* is the map for which each grid cell c has the

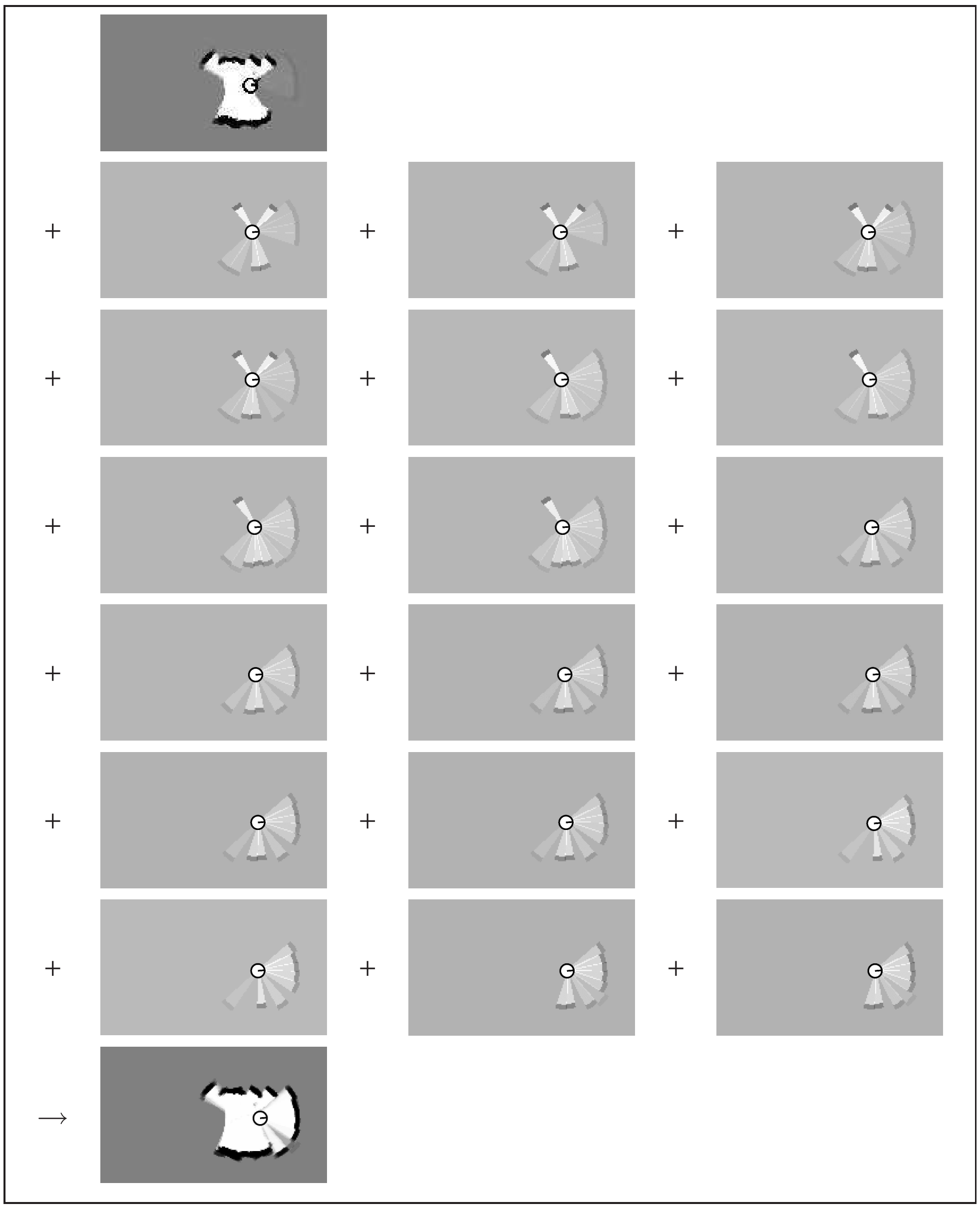

Figure 2.6: Incremental mapping in a corridor environment. The upper left image shows the initial map and the lower one contains the resulting map. The maps in between are the local maps built from the individual ultrasound scans perceived by the robot.

value

$$p(c \mid x_{1:t}, z_{1:t}) \quad = \quad \frac{\#hits(c, x_{1:t}, z_{1:t})}{\#hits(c, x_{1:t}, z_{1:t}) + \#misses(c, x_{1:t}, z_{1:t})}, \qquad (2.42)$$

where $\#misses(c, x_{1:t}, z_{1:t})$ is the number of times a beam $z_{t,n}$ taken from x_t passed through the grid cell c and $\#hits(c, x_{1:t}, z_{1:t})$ is the number of times a beam ended in that cell. Since the value of each cell can be determined by counting, this technique is also called counting model.

The differences between occupancy probability and reflection probability maps is that the occupancy probability typically converges to 0 or 1 for each cell which is frequently observed. In contrast to that, reflection probability values converge to values *between* 0 and 1. Values significantly different from 0 or 1 often occur when mapping objects much smaller than the grid discretization or, for example, for glass panes which are repeatedly observed with a laser range finder.