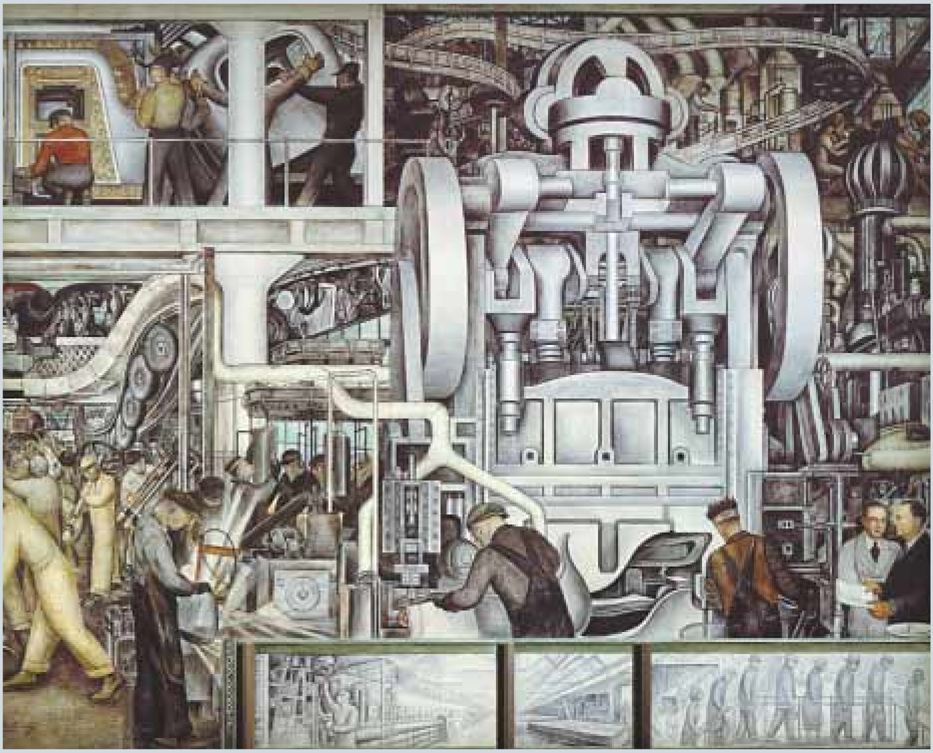
INTRODUCTION TO
AI ROBOTICS



ROBIN R. MURPHY

11 *Localization and Map Making*

Chapter Objectives:

- Describe the difference between *iconic* and *feature-based* localization.
- Be able to update an occupancy grid using either Bayesian, Dempster-Shafer, or HIMM methods.
- Describe the two types of formal exploration strategies.

11.1 Overview

The two remaining questions of navigation are: *where am I?* and *where have I been?* The answers to these questions are generally referred to as *localization* and *map-making*, respectively. Both are closely related, because a robot cannot create an accurate map if it does not know where it is. Fig. 11.1 shows a hallway in black in a building. The hallway makes a complete circuit around the center of the building. The gray shows the hallway as sensed by a mobile robot. The mobile robot senses, updates the map with the portions of the hallway that have come into view, then moves, updates, and so on. In this case, it uses shaft encoders to determine where it has moved to and how to update the map.

As can been seen from the figure, as well as discussions in Ch. 6, shaft encoders are notoriously inaccurate. Worse yet, the inaccuracies are highly dependent on surfaces. For example, the robot's wheel will slip differently on carpet than on a polished floor. Developing an error model to estimate the slippage is often unrealistically difficult. The shaft encoder problem might appear to be eliminated by new hardware technology, especially GPS and MEMS (micro electrical-mechanical systems) inertial guidance systems (INS).

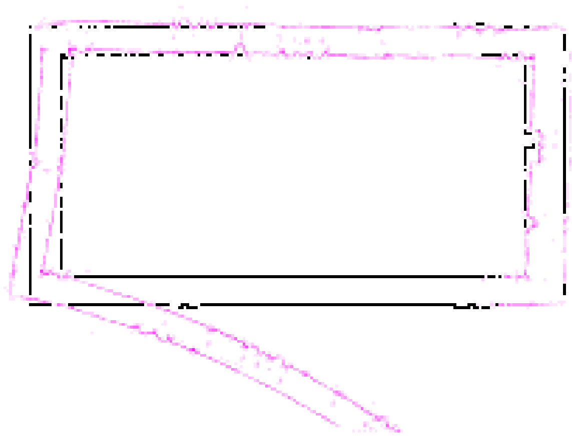


Figure 11.1 A map of a circuit of a hallway created from sonars by a Nomad 200 showing the drift in localization. The ground truth is in black.

However, GPS only works reliably outdoors. The signal is often unobtainable indoors, in tunnels, or in cities with large buildings (sometimes referred to as *urban canyons*). MEMS inertial navigation devices are small, but suffer from significant inaccuracies and have not been packaged in a way to be easily used with robots.

URBAN CANYONS

Researchers have attempted to solve the localization problem in a number of ways. The first approach was to simply ignore localization errors. While this had the advantage of being simple, it eliminated the use of global path planning methods. This was part of the motivation and appeal of purely reactive systems, which had a "go until you get there" philosophy. Another approach was to use topological maps, which have some symbolic information for localization at certain points such as gateways, but don't require continuous localization. Unfortunately, for reasons discussed in Ch. 9, it is hard to have unique gateways. The move to topological mapping gave rise to a whole subfield of reasoning about indistinguishable locations.

More sophisticated systems either identified natural landmarks which had noticeable geometric properties or added artificial landmarks. One robot proposed for a Canadian mining company intended to navigate through relatively featureless mine shafts by dropping beacons at different intersections, much like Hansel and Gretel dropping cookie crumbs for a path. (This inspired much humorous discussion of the merits of the biological equivalent of robot droppings and robots imitating animals that "mark" their territory in the wild.) Other techniques attempted to match the raw sensor data to an *a priori* map using interpretation trees or similar structures. One of the many problems with these techniques is that the sensor data rarely comes in a form amenable to matching against a map. Consider attempting to match noisy sonar data to the layout of a room. In the end, the basic approach used by most systems is to move a little, build up a small map, match the new map to the last map, and merge it in, then merge the small map with the overall map. The use of small, local maps for localization brings the process back full circle to the need for good map-making methods.

Localization algorithms fall into two broad categories: *iconic* and *feature-based*. Iconic algorithms appear to be the more popular in practice, in part because they usually use an occupancy grid. Occupancy grids are a mechanism for fusing sensor data into a world model or map. Fusion is done either following an algorithm provided by a formal theory of evidence, either Bayesian or Dempster-Shafer, or by a popular quasi-evidential method known as HIMM. Since occupancy grids fuse sensor data, the resulting map does not contain as much sensor noise. Many Hybrid architectures also use the occupancy grid as a *virtual sensor* for obstacle avoidance.

The chapter first covers occupancy grids, which are also known as certainty and evidence grids. Since sonars are a popular range sensor for mapping and obstacle avoidance, the chapter next covers sonar sensor models and the three methods for using sensor models to update a grid: Bayesian, Dempster-Shafer, and HIMM. The Bayesian and Dempster-Shafer methods can be used with any sensor, not just range from sonar. The comparison of the three methods discusses practical considerations such as performance and ease in tuning the method for a new environment. Iconic localization is described next. It is useful for metric map building and generally uses an occupancy grid-like structure. Feature-based localization, which is better suited for topological map building, is discussed next. Feature-based methods have become popular with the advent of partially ordered Markov decision process (POMDP) methods to simplify reasoning about them; POMDPs are outside the scope of this book but the basic localization strategy is presented. The chapter ends with a brief description of frontier and Voronoi methods of using the data in an occupancy grid to direct exploration of an unknown environment.

11.2 Sonar Sensor Model

All methods of updating uncertainty require a sensor model. Models of sensor uncertainty can be generated in a number of ways. *Empirical methods* for generating a sensor model focus on testing the sensor and collecting data as to the correctness of the result. The frequency of a correct reading leads to a belief in an observation; the set of beliefs from all possible observations form the model. *Analytical methods* generate the sensor model directly from an understanding of the physical properties of the device. *Subjective methods* rely on a designer's experience, which are often an unconscious expression of empirical testing.

One robotic sensor which has been heavily studied is the Polaroid ultrasonic transducer, or sonar. This chapter will use Polaroid sonars as an example; however, the principles of scoring and fusing belief apply to any sensor. Most roboticists have converged on a model of sonar uncertainty which looks like Fig. 11.2, originally presented in Ch. 6.

SONAR MODAL PARAMETERS

The *basic model of a single sonar beam* has a field of view specified by β, the half angle representing the width of the cone, and R, the maximum range it can detect. This field of view can be projected onto a regular grid. The grid will be called an *occupancy grid*, because each element l (for eLement) in the grid will hold a value representing whether the location in space is occupied or empty. As shown in Fig. 11.2, the field of view can be divided into three regions:

OCCUPANCY GRID ELEMENT L

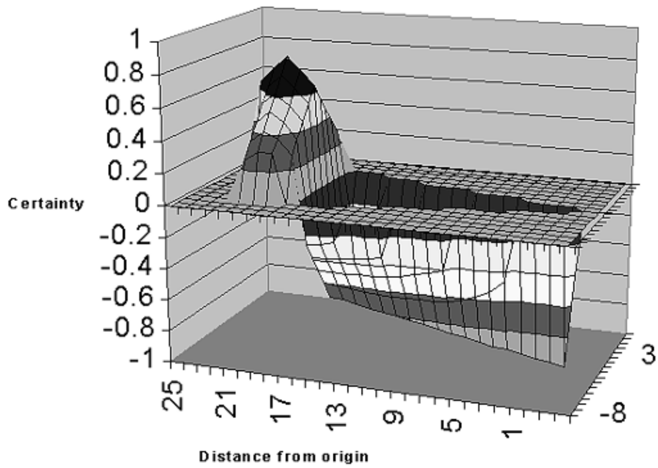
Region I: where the affected elements are probably occupied (drawn as a "hill"),

Region II: where the affected elements are probably empty (drawn as a "valley"), and

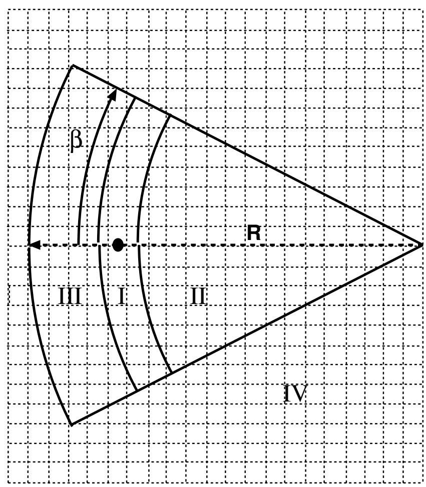
Region III: where the condition of the affected elements is unknown (drawn as a flat surface).

Given a range reading, Region II is more likely to be really empty than Region I is to be really occupied. Regardless of empty or occupied, the readings are more likely to be correct along the acoustic axis than towards the edges. Recall that this is in part because an obstacle which was only along one edge would be likely to reflect the beam specularly or generate other range errors.

While the sensor model in Fig 11.2 reflects a general consensus, there is much disagreement over how to convert the model into a numerical value



a.



b.

Figure 11.2 A sensor model for a sonar: a.) three dimensional representation and b.) two dimensional representation projected onto an occupancy grid.

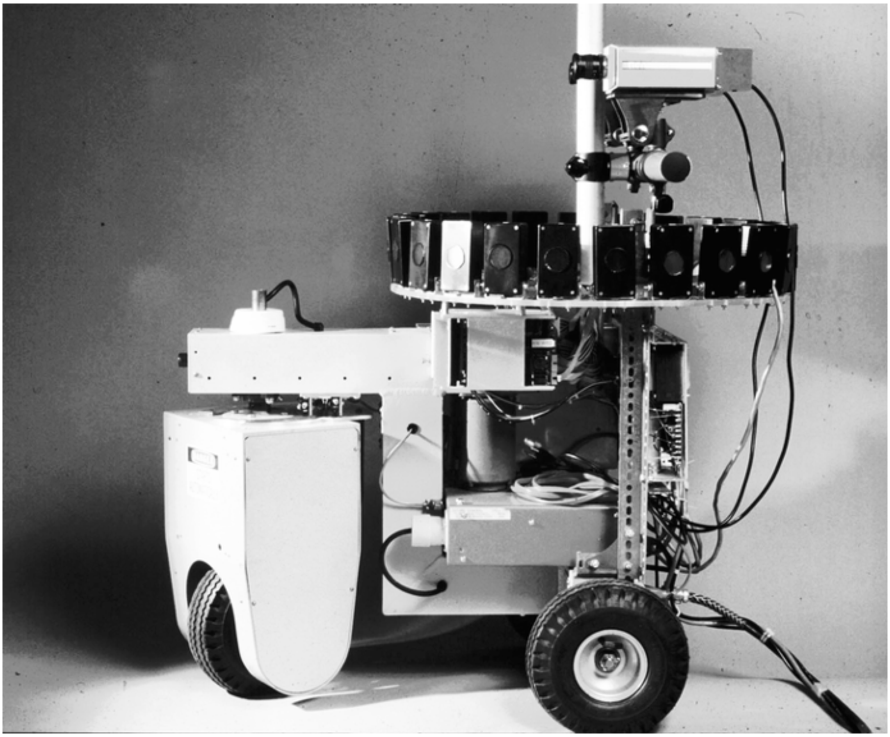


Figure 11.3 Neptune, a robot using occupancy grids during the early 1980's. (Photograph courtesy of Hans Moravec.)

for belief. Each of the three methods covered in the following sections does the translation slightly differently.

11.3 Bayesian

The most popular evidential method for fusing evidence is to translate sensor readings into *probabilities* and to combine probabilities using *Bayes' rule*. Elfes and Moravec at Carnegie Mellon University pioneered the probabilistic approach in the early 1980's. Later Moravec turned to a form of Bayes' Rule which uses probabilities expressed as *likelihoods* and *odds*.[95] This has some computational advantages and also side-steps some of the problems with priors. The likelihood/odds formulation is equivalent to the traditional approach presented here. In a Bayesian approach, the sensor model generates conditional probabilities of the form $P(s|H)$. These are then converted to $P(H|s)$ using Bayes' rule. Two probabilities, either from two different sensors sensing at the same time or from two different times, can be fused using Bayes' rule.

11.3.1 Conditional probabilities

PROBABILITY FUNCTION

To review, a *probability function* scores evidence on a scale of 0 to 1 as to whether a particular event **H** (**H** stands for "hypothesis") has occurred given an experiment. In the case of updating an occupancy grid with sonar readings, the experiment is sending the acoustic wave out and measuring the time of flight, and the outcome is the range reading reporting whether the region being sensed is Occupied or Empty.

Sonars can observe only one event: whether an element $grid[i][j]$ is Occupied or Empty. This can be written $\mathbf{H} = \{H, \neg H\}$ or $\mathbf{H} = \{Occupied, Empty\}$.

The probability that H has really occurred is represented by $P(H)$:

$$0 \leq P(H) \leq 1$$

An important property of probabilities is that the probability that H didn't happen, $P(\neg H)$, is known if $P(H)$ is known. This is expressed by:

$$1 - P(H) = P(\neg H)$$

As a result, if $P(H)$ is known, $P(\neg H)$ can be easily computed.

UNCONDITIONAL PROBABILITIES

Probabilities of the form $P(H)$ or $P(\neg H)$ are called *unconditional probabilities*. An example of an unconditional probability is a robot programmed to explore an area on Mars where 75% of the area is covered with rocks (obstacles). The robot knows in advance (or *a priori*) that the next region it scans has $P(H = Occupied) = 0.75$.

Unconditional probabilities are not particularly interesting because they only provide *a priori* information. That information does not take into account any sensor readings, S. It is more useful to a robot to have a function that computes the probability that a region $grid[i][j]$ is either Occupied or Empty given a particular sensor reading s. Probabilities of this type are called *conditional probabilities*. $P(H|s)$ is the probability that H has really occurred given a particular sensor reading s (the "|" denotes "given"). Unconditional probabilities also have the property that $P(H|s) + P(\neg H|s) = 1.0$.

CONDITIONAL PROBABILITIES

In an occupancy grid, $P(Occupied|s)$ and $P(Empty|s)$ are computed for each element, $grid[i][j]$, that is covered by a sensor scan. At each grid element, the tuple of the two probabilities for that region is stored. A tuple can be implemented as a C `struct`

```
typedef struct {
  double  occupied;
  double  empty;
} P;

P occupancy_grid[ROWS][COLUMNS];
```

Probabilities provide a representation for expressing the certainty about a region $grid[i][j]$. There still needs to be a function which transfers a particular sonar reading into the probability for each grid element in a way that captures Fig. 11.2. One set of functions which quantify this model into probabilities is given below.

For every grid element falling into **Region I**:

$$\begin{aligned} P(Occupied) &= \frac{(\frac{R-r}{R}) + (\frac{\beta-\alpha}{\beta})}{2} \times Max_{occupied} \\ P(Empty) &= 1.0 - P(Occupied) \end{aligned} \tag{11.1}$$

where r and α are the distance and angle to the grid element, respectively. The $\frac{\beta-\alpha}{\beta}$ term in Eqn. 11.1 captures the idea that the closer the grid element is to the acoustic axis, the higher the belief. Likewise, the nearer the grid element is to the origin of the sonar beam, the higher the belief (the $\frac{R-r}{R}$ term). The $Max_{occupied}$ term expresses the assumption that a reading of occupied is never fully believable. A $Max_{occupied} = 0.98$ means that a grid element can never have a probability of being occupied greater than 0.98.

It is important to note that Region I in Fig. 11.2 has a finite thickness. Due to the resolution of the sonar, a range reading of 0.87 meters might actually be between 0.82 and 0.92 meters, or 0.87±0.05 meters. The ± 0.05 is often called a *tolerance*. It has the impact of making Region I wider, thereby covering more grid elements.

TOLERANCE

Each grid element in **Region II** should be updated using these equations:

$$\begin{aligned} P(Occupied) &= 1.0 - P(Empty) \\ P(Empty) &= \frac{(\frac{R-r}{R}) + (\frac{\beta-\alpha}{\beta})}{2} \end{aligned} \tag{11.2}$$

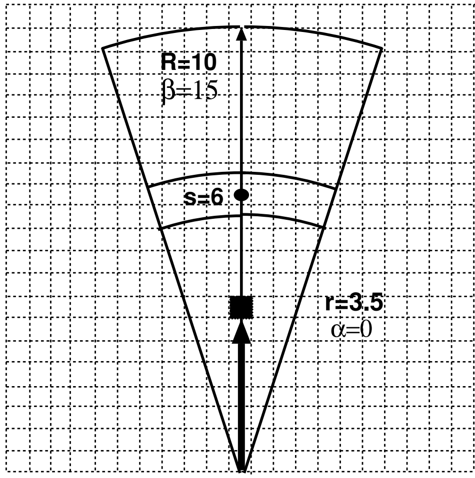


Figure 11.4 Example 1: Updating an element in Region II (sonar reading of 6).

Note that unlike an element in Region I, an element in Region II can have a probability of being empty of 1.0.

To see how these formulas would be applied, consider the example in Fig. 11.4. The sonar has returned a range reading of 6.0 feet with a tolerance of ± 0.5 feet. The $Max_{occupied}$ value is 0.98. The robot is shown on a grid, and all elements are measured relative to it. The element of interest $grid[i][j]$ is shown in black, and is at a distance $r = 3.5$ feet and an angle of $\alpha = 0°$ from the robot. In a computer program, r and α would be computed from the distance and arctangent between the element of interest and the element representing the origin of the sonar, but for the sake of focus, these examples will give r and α.

The first step is to determine which region covers the element. Since $3.5 < (6.0 - 0.5)$, the element is in Region II. Therefore, the correct formulas to apply are those in Eqn. 11.2:

$$\begin{array}{rclcl} P(Empty) & = & \frac{(\frac{R-r}{R})+(\frac{\beta-\alpha}{\beta})}{2} & = & \frac{(\frac{10-3.5}{10})+(\frac{15-0}{15})}{2} & = & 0.83 \\ P(Occupied) & = & 1.0 - P(Empty) & = & 1 - 0.83 & = & 0.17 \end{array}$$

The example in Fig. 11.5 shows an element in Region I. The probability for the element in black is computed the same way, only using the equations for that region.

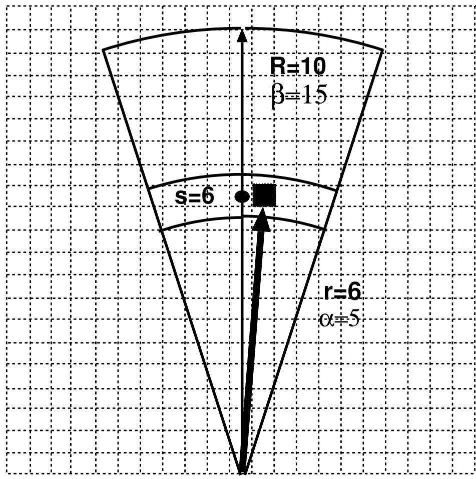


Figure 11.5 Example 2: Updating a element in Region I (sonar reading at 6).

$$\begin{array}{rclclcl} P(Occupied) & = & \frac{(\frac{R-r}{R})+(\frac{\beta-\alpha}{\beta})}{2} \times Max_{occupied} & = & \frac{(\frac{10-6}{10})+(\frac{15-5}{15})}{2} \times 0.98 & = & 0.52 \\ P(Empty) & = & 1.0 - P(Occupied) & = & 1 - 0.52 & = & 0.48 \end{array}$$

11.3.2 Conditional probabilities for $P(H|s)$

The sensor model represents $P(s|H)$: the probability that the sensor would return the value being considered given it was really occupied. Unfortunately, the probability of interest is $P(H|s)$: the probability that the area at $grid[i][j]$ is really occupied given a particular sensor reading. The laws of probability don't permit us to use the two conditionals interchangeably. However, Bayes' rule does specify the relationship between them:

$$P(H|s) = \frac{P(s|H)P(H)}{P(s|H)P(H) + P(s|\neg H)P(\neg H)} \quad (11.3)$$

Substituting in $Occupied$ for H, Eqn. 11.3 becomes:

$$P(Occupied|s) = \frac{P(s|Occupied)\,\boxed{\text{P(Occupied)}}}{P(s|Occupied)\,\boxed{\text{P(Occupied)}} + P(s|Empty)\,\boxed{\text{P(Empty)}}} \quad (11.4)$$

$P(s|Occupied)$ and $P(s|Empty)$ are known from the sensor model. The other terms, $P(Occupied)$ and $P(Empty)$, are the unconditional probabilities, or *prior probabilities* sometimes called *priors*. The priors are shown in

Eqn. 11.4 in boxes. If these are known, then it is straightforward to convert the probabilities from the sonar model to the form needed for the occupancy grid.

In some cases, such as for a planetary rover, there may be some knowledge that produces the prior probabilities. In most cases, that knowledge isn't available. In those cases, it is assumed that $P(Occupied) = P(Empty) = 0.5$. Using that assumption, the probabilities generated for the example in Fig. 11.5 can be transformed as follows.

For $grid[i][j]$:

$$\begin{aligned} P(s=6|Occupied) &= 0.62 \\ P(s=6|Empty) &= 0.38 \\ P(Occupied) &= 0.5 \\ P(Empty) &= 0.5 \end{aligned}$$

Substituting into Eqn. 11.4 yields:

$$\begin{aligned} P(Occupied|s=6) &= \frac{(0.62)(0.5)}{(0.62)(0.5)+(0.38)(0.5)} = 0.62 \\ P(Empty|s=6) &= \frac{(0.38)(0.5)}{(0.38)(0.5)+(0.62)(0.5)} = 0.38 \end{aligned}$$

The use of 0.5 for the priors made $P(Occupied|s)$ numerically equivalent to $P(s|Occupied)$, but in general $P(H|s) \neq P(s|H)$.

11.3.3 Updating with Bayes' rule

Now that there is a method for computing conditional probabilities of the correct form, the question becomes how to fuse it with other readings. The first update is simple. Each element in the occupancy grid is initialized with the *a priori* probability of being occupied or empty. Recall that this is generally implemented as a data structure consisting of two fields. If the *a priori* probabilities are not known, it is assumed $P(H) = P(\neg H) = 0.5$. The first observation affecting $grid[i][j]$ can use Bayes' rule to compute a new probability and replace the prior $P(H) = 0.5$ with the new value.

But what about the second observation? Or an observation made from another sonar at the same time? It turns out that in both cases, Bayes' rule can be used iteratively where the probability at time t_{n-1} becomes the prior and is combined with the current observation (t_n).

To see this, consider the following derivation. For n multiple observations, $s_1, s_2, \ldots s_n$, Bayes' rule becomes:

$$P(H|s_1, s_2, \ldots s_n) = \frac{P(s_1, s_2, \ldots s_n|H)P(H)}{P(s_1, s_2, \ldots s_n|H)P(H) + P(s_1, s_2, \ldots s_n|\neg H)P(\neg H)} \quad (11.5)$$

This introduces the problem of generating $P(s_1, s_2, \ldots s_n|H)$. Ideally, this requires a sonar model of getting occupied and empty values for all $grid[i][j]$ with n combinations of sensor readings. Fortunately, if the reading from s_1 can be considered the result of a different experiment than s_2 and the others, $P(s_1, s_2, \ldots s_n|H)$ simplifies to $P(s_1|H)P(s_2|H)\ldots P(s_n|H)$. Now, the program only has to remember all previous $n-1$ readings. Since there is no way of predicting how many times a particular grid element will be sensed, this creates quite a programming problem. The occupancy grid goes from being a two dimensional array with a single two field structure to being a two dimensional array with each element a potentially very long linked list. Plus, whereas Eqn. 11.3 involved 3 multiplications, updating now takes $3(n-1)$ multiplications. The computational overhead begins to be considerable since an element in a hallway may have over 100 observations.

Fortunately, by clever use of $P(H|s)P(s) = P(s|H)P(H)$, a recursive version of Bayes' rule can be derived:

$$P(H|s_n) = \frac{P(s_n|H)P(H|s_{n-1})}{P(s_n|H)P(H|s_{n-1}) + P(s_n|\neg H)P(\neg H|s_{n-1})} \quad (11.6)$$

So at each time a new observation is made, Eqn. 11.6 can be employed and the result stored at $grid[i][j]$. The rule is commutative, so it doesn't matter in what order two or more simultaneous readings are processed.

11.4 Dempster-Shafer Theory

An alternative theory of evidence is *Dempster-Shafer theory* which produces results similar to Bayesian probabilities. It is a much newer theory, originating in the work of A.P. Dempster, a mathematician at Harvard, during the 1960's with extensions by Glen Shafer in 1987.[126] Whereas Bayes' rule relies on evidence being represented by probability functions, Dempster-Shafer theory represents evidence as a *possibilistic* belief function. Possibilistic means that the function represents *partial evidence*. For example, a reading