Building Maps for Indoor Mobile Robots Using Ultrasonic and Laser Range Sensors

Abstract

A new method for learning probabilistic grid-based maps of indoor environments by a mobile robot is described. New contributions on two major components of map learning, namely, sensor data fusion and exploration are proposed. In particular, new models of sensors and a way of sensor data fusion that takes advantage of multiple viewpoints are presented. The exploration approach merges a local strategy, similar to wall following to keep the robot close to obstacles, within a global search frame, based on a dynamic programming algorithm. We introduce the concept of travel space as a way to map costs to grid cells based on distances to obstacles. This method is tested using a simulated and a real mobile robot with odometer, ultrasonic and laser range sensors (implemented with a laser line generator and a camera) with promising results.

Keywords: Map Building, Exploration, Sensor Data Fusion, Mobile Robots.

1 Introduction

This paper deals with the task of learning a model of an environment by an indoor mobile robot using its sensors. The map learned is commonly used by the mobile robot navigation system while doing a high level task. Sometimes there is no hand–coded map of the environment, it is difficult or dangerous to build a map by a human being, or the hand–coded map does not take into account the characteristics of the sensors (Lee, 1996). The ability to learn a map of the environment increases the flexibility and usefulness of mobile robots.

The problem of learning a model of an environment by a mobile robot while it is moving is difficult and far from being solved (Thrun, 1998). The following factors impose practical limitations (Thrun, 1998):

- *Sensors limitations.* Sensors are often not capable of directly measuring the quantity of interest.
- *Perceptual limitations.* The perceptual range of most sensors is limited to a small range around the robot.
- *Sensor noise.* Sensor measurements are typically corrupted by noise.
- *Drift/Slippage.* Robot motion is inaccurate and odometric errors accumulate over time.

This paper presents an approach for an indoor mobile robot using its sensors, to learn a *Probabilistic Grid-based Map* (PGM) (Elfes, 1989; Thrun *et al.*, 1998) of an environment. A PGM is a two dimensional map where the environment is divided in square regions or cells of the same size that have occupancy probabilities associated to them. A PGM is appealing because it can be easily used to fuse sensor data. See (Gallistel, 1990) for other types of maps and (Lee, 1996) for a discussion of advantages and disadvantages of each type of map .

This work considers a mobile robot with a ring of ultrasonic range sensors, or *sonars* for short, a common arrangement used in other previous research (Thrun, 1998). The robot has as well a laser sensor, which measures proximity of nearby objects using a combination of a laser line generator and a camera. The laser line is parallel to the floor. Distances to objects can be estimated considering the height of the *laser points* within images.

Building a PGM can be described by three major components:

Sensor data fusion. Multiple sensor data are mapped and integrated over time to occupancy probabilites of grid cells. An extended approach to the sonar data fusion described in (Howard & Kitchen, 1996) is given. A new laser range model is also presented. Finally, a way to fuse data from different sensor types is described.

Exploration. The robot explores its environment trying to reach the nearest unexplored grid cell minimizing the travel cost. A novel approach to merge local strategies, like *wall following*, within

a global search frame, based on a dynamic programming algorithm, is given.

Position estimation. The position of the robot is continuously tracked, minimizing odometric errors. This approach estimates the position of the robot based on correlations between laser range data and the map.

The remainder of this paper is organized as follows. Sections 2, 3 and 4 describe each of the three components for building a PGM. Experimental results using a mobile robot simulator and a real mobile robot are presented in section 5. Finally, some conclusions and future research directions are given.

2 Sensor Data Fusion

The environment is modeled as a set of cells arranged in a regular two–dimensional grid. The occupancy of a cell at location (x, y) is measured by the state variable $O(x, y)$ which can have one of two values: *occ* (occupied) and *free*. For each cell, the probability that $O(x, y) = occ$, denoted shortly by $P(O)$, is estimated using the sensors of the robot. In this work it is considered that the robot has three types of sensors:

Ultrasonic range sensors. In this case, let $P(O_s)$ be the occupancy probability of the cell (x,y) detected only by sonars.

Laser range sensors. In the same way, let $P(O_l)$ be the occupancy probability detected only by this type of sensors.

Maneuverability. The mere fact that a robot moves to a location (x,y) makes it unlikely that this location is occupied. Let $P(O_m)$ be the occupancy probability detected by this type of sensor.

A cell will be considered occupied if it is detected occupied by at least one sensor. In this way, the probability that a given cell is occupied can be estimated using a logical OR operation among the occupancy states detected by each type of sensor:

$$P(O) = P(O_s \ OR \ O_l \ OR \ O_m) \tag{1}$$

To expand the right hand side of (1), it is assumed that the events O_s, O_l and O_m are statistically independent. With this assumption, and after some algebra, equation (1) becomes:

$$P(O) = 1 - \prod_{i=s,l,m} (1 - P(O_i)) \tag{2}$$

This expression can be used to compute the probability that a cell is occupied once we have determined the probability that a cell is occupied by each type of sensor. The prior probabilities $P(O)$, are initially set to 0.5 to indicate ignorance. This implies that the prior probabilities for the variables associated to each type of sensor i $(i = s, l, m)$ for every cell are given by:

$$P_{prior}(O_i) = 1 - (0.5)^{1/3} \tag{3}$$

A description of how to compute the probability that cells are occupied using each type of sensor is given in the following sections.

2.1 Ultrasonic Range Sensors

There are two main difficulties using sonars: 1) they have a wide beam, like a 30 degrees cone and 2) a specular reflection occurs whenever an ultrasonic pulse encounters a smooth extended surface. In ordinary office environments which contain smooth walls and glass doors specular reflection is common (Howard & Kitchen, 1996). Elfes (1989) and Moravec (1988) describe an occupancy grid approach in which range measurements from multiple viewpoints are combined into a PGM. Each cell in the grid is assigned a *single* value indicating the probability that the cell is occupied. Unfortunately, the occupancy grid approach does not work well in specular environments (Howard & Kitchen, 1996). Howard and Kitchen (1996) propose an improvement of grid-based approaches by introducing the concept of *response grid*. The basic idea is that a cell may generate a response (e.g. appears to be occupied) when viewed from one direction, but will not generate a response when viewed from another.

Following the approach described in (Howard & Kitchen, 1996), when an ultrasonic pulse entering a cell with some direction is reflected back to the detector, the cell is said to have a *response* in that direction. The occupancy value of the cell is determined by assuming that any cell that generates a response in one or more directions must contain at least one surface and therefore it is occupied. The response of a cell (x, y) in some direction ϕ is measured by the variable R (res means response): $R(x, y, \phi) = res$. The full interval $[0, 360^o]$ is divided into n intervals. Let R_i, be the proposition that a given cell (x, y) generates a response for the direction interval ϕ_i. The probability that the cell is occupied using sonars is given by:

$$P(O_s) = P(R_1 \ OR \ \cdots \ OR \ R_n) \tag{4}$$

To expand the right hand side of (4), it is assumed that the events R_i are mutually independent. With this assumption, (4) becomes:

$$P(O_s) = 1 - \prod_{i=1}^{n}(1 - P(R_i)) \quad (5)$$

The prior probability $P(R_i)$ can be computed from equation (2):

$$P_{prior}(R_i) = 1 - (1 - P_{prior}(O_s))^{\frac{1}{n}} \quad (6)$$

The following sensor model is used to estimate the probability of a response given a measurement r, $P(R_i|r)$. Let s be the distance between the cell (x, y) and the sensor. $P(R_i|r)$ for all grid cells under the sonar cone are estimated by:

$$P(R_i|r) = \quad (7)$$

$$\begin{cases} 1 - (1 - P_{prior}(R_i))(1 - Ks_{max})^{\frac{1}{Nc}} & if \quad s < r \\ P_{prior}(R_i) & if \quad s > r \\ P_{prior}(R_i) Ks_{min}{}^{\frac{1}{Nc}} & if \quad s = r \end{cases}$$

where Ks_{max} is a constant close to 1, Ks_{min} is another constant close to 0, and Nc is the number of grid cells ($Nc >= 1$) at range s covered by the sonar. In this way, the probability of a cell whose distance s is greater than the sonar reading r is not changed; when $s = r$ short sonar readings tend to significantly increase the probability of occupancy, while long sonar readings tend to slightly increase it; and when $s < r$ short sonar readings tend to significantly decrease the probability of occupancy, while long sonar readings tend to slightly decrease it.

To compute $P(R_i)$ given m sensor readings, denoted $r^{(1)}$,$r^{(2)}$, ... ,$r^{(m)}$, one has to assume conditional independence between $r^{(i)}$ and $r^{(j)}$ $(i \neq j)$ given that $(R_i(x, y) = res)$ is true. In other words:

$$P(r^{(i)}|R_i, r^{(1)}, \ldots, r^{(i-1)}, r^{(i+1)}, \ldots, r^{(m)}) = P(r^{(i)}|R_i) \quad (8)$$

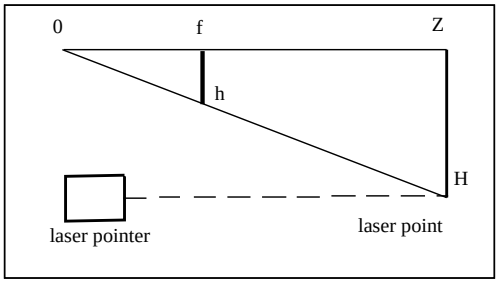
With this assumption, a probabilistic update rule for $P(R_i)$ can be deduced (Thrun *et al.*, 1998), (Thrun, 1998):

$$P(R_i|r^{(1)}, \ldots, r^{(m)}) = 1 - [1 + \frac{P(R_i)}{1 - P(R_i)} Prod_1]^{-1} \quad (9)$$

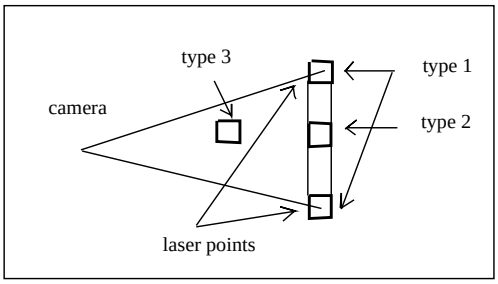
where

$$Prod_1 = \prod_{j=1}^{m} [\frac{P(R_i|r^{(j)})}{1 - P(R_i|r^{(j)})}][\frac{1 - P(R_i)}{P(R_i)}]$$

This equation can be used to update the probabilities in an incremental way.



a)



b)

Figure 1: Laser model. (a) Laser range sensor implemented with laser pointers and a camera. (b) Types of cells within the laser model.

2.2 Laser range sensor

A camera, together with a laser line generator parallel to the floor, are aligned to implement a laser range sensor. This arrangement can rotate to cover the whole area around the robot. In the following analysis we consider the laser line generator as a set of laser pointers in a radial arrangement.

Let us now consider the uncertainty in the estimated distances due to image resolution. Let h be the distance between the x–axis of the image and the laser point within the image, H be the height of the laser point (a constant), f be the camera focal length and Z be the distance from the camera center to the laser point (see Fig. 1 (a)). The relative error of Z due to image resolution (Δh) is given by (Shigang *et al.*, 1992):

$$|\frac{\Delta Z}{Z}| = \frac{1}{h}|\Delta h| \quad (10)$$

Considering similar triangles, the absolute error $E_l = |\Delta Z|$, for a given distance Z, can be expressed as:

$$E_l(Z) = K_l Z^2 \quad (11)$$

where $K_l = \frac{\Delta h}{fH}$ is a constant that depends on the specific camera (f and Δh) and the vertical distance between the camera and the laser pointers (H). In order to update the probability $P(O_l)$ given a set of

laser range data, the proposed model considers the laser data in consecutive pairs[1]. Three types of cells (see Fig. 1 (b)), for two consecutive readings are considered:

1. Cells associated to each reading.
2. Cells between cells of type (1), considering a linear interpolation.
3. Cells inside the polygon formed by the camera and the two readings.

For each type of cell, the probability $P(O_l)$, given two consecutive readings $Z^{(1)}, Z^{(2)}$ is updated in the following way. Let $Z^{(i)}$ be the reading associated to one cell of type (1). Then, the probability that this type of cell is occupied given the reading is given by:

$$P(O_l|Z^{(i)}) = 1 - (1 - P_{prior}(O_l))(1 - Kl_{max})^{E_l(Z^{(i)})} \quad (12)$$

where Kl_{max} is a constant close to 1. This expression takes into account the absolute error associated to the reading, and assigns high values to short readings and low values to large readings. The update rule for cells of type 1, chooses the minimum value between the old and the new probability values. In this way, short laser readings overwrite probabilities due to longer readings. To update cells of type 2, if the minimum of $P(O_l)|Z^{(1)})$ and $P(O_l)|Z^{(2)})$ is greater than the old value, then the minimum value between $P(O_l)|(Z^{(1)}+Kl_a)$ and $P(O_l)|(Z^{(2)}+Kl_a)$ is choseen. This means that short readings, as before (considering the minimum of both readings), overwrite probabilities due to longer readings. Kl_a depends on the belief that these cells are occupied. This value could be a function on the distance between the two laser points. To update cells of type 3, if the minimum of $P(O_l)|Z^{(1)})$ and $P(O_l)|Z^{(2)})$ is greater than the old value, then we multiply the old value by a factor Kl_i in the interval $(0, 1)$. Kl_i depends on the belief that these cells are free.

This is a much simpler approach than that of equation (9) which can be used in this case since laser range data are less noisy than sonar data.

2.3 Maneuverability

In this case, we use a simple update rule. To update $P(O_m)$ given the position of the robot, all the cells under the robot are decreased by a factor K_m in the interval $(0, 1)$.

[1] That is, adjacent laser points on an angular sequence.

3 Exploration

Research on exploration strategies has developed two general approaches: *reactive* and *model based*. By far the most widely–used exploration strategy in reactive robotics is *wall following*. Model based strategies vary with the type of model being used, but they are based on the same underlying idea: *go to the least–explored region* (Lee, 1996). A critical issue while exploring the environment is to estimate the robot position. There are several successful localization methods that can estimate the robot's position using its sensors (Gutmann *et al.*, 1998). However, most localization methods fail when the sensors of the robot are beyond its perceptual capability (Roy *et al.*, 1999) (i.e. the robot is too far from obstacles). This paper introduces a novel approach to explore a static indoor environment. The idea is to reach the nearest unexplored grid cell minimizing the travel cost. The travel cost takes into account the perceptual limitations of the sensors and tries to maintain a fixed distance to obstacles while the robot is moving (*wall following*). The concept of *travel space* is introduced to assign costs to grid cells. The motion policy of the robot is computed using a dynamic programing algorithm that includes the costs associated to the travel space. This approach merges local or *reactive* strategies with a global or *model based* strategy.

The general idea for exploration is to move the robot on a minimum–cost path to the nearest unexplored grid cell (Thrun, 1998). The minimum–cost path is computed using *value iteration*, a popular dynamic programming algorithm. In (Thrun, 1998) the cost for traversing a grid cell is determined by its occupancy value, while in (McKerrow, 1991) the cost is determined by the distance between cells (see chapter 8 in (Lee, 1996)). This paper proposes an approach that combines local search strategies within a modified version of value iteration described in (McKerrow, 1991). When the robot starts to build a map, all the cells have the same probability of occupancy $P(O) = 0.5$. A cell is considered *unexplored* when its occupancy probability is in the interval (close to 0.5) defined by two constants $[Pe_{min}, Pe_{max}]$ $(Pe_{min} < 0.5 < Pe_{max})$ and *explored* otherwise.

Cells are defined as *free* or *occupied*. A cell is considered occupied when its $P(O)$ reaches a threshold value Po_{max} and continues to be occupied while its $P(O)$ does not fall below a threshold value Po_{min} (where $Po_{min} < Po_{max}$). It is considered *free* in other case. This mechanism prevents changes in the state of occupancy of a cell by small probability changes. We assume that $Pe_{max} < Po_{min}$, so an unexplored cell is

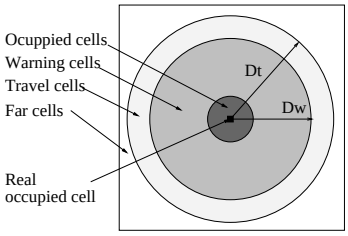


Figure 2: Travel space due to a single occupied cell

Figure 3: A travel space. From darker to lighter: occupied cell (black), warning cells (dark gray), travel cells (light gray), and far cells (white)

also a free cell. In this way, the PGM becomes a binary map when cells are classified as occupied or free. This binary map will be called *occupied–free* map.

In this work, a cylindrical (circular base) robot was used, so the configuration space (c–space) (Latombe, 1991) can be computed by growing the occupied cells by the radius of the robot. In fact, the c-space is extended to form a *travel space*. The idea behind the travel space is to define a way to control the exploration by a kind of *wall following strategy*. Wall following is a local method that has been used to navigate robots in indoor environments, but unfortunately it can easily get trapped in loops (Lee, 1996). The travel space together with a dynamic programming technique has the advantages of both, local and global strategies: robustness and completeness.

Consider the travel space due to a single real occupied cell in the occupied–free map (see Figure 2).

The travel space splits the free cells of the occupied–free map in four categories:

1. *Occupied cells.* These cells are inside the circle given by the radius of the robot (as in the c–space) with center in the real occupied cell. After this expansion, the robot is considered as a single cell.

2. *Warning cells.* Cells close to an occupied cell. Let D_w be the maximum distance between a cell of this type and its closest real occupied cell. These cells are called *warning cells* because their purpose is to warn the robot about its closeness to an obstacle. The value of D_w takes into account the perceptual limitations of the sensors.

3. *Travel cells.* Cells close to a warning cell. Let D_t be the maximum distance between a cell of this type and its closest real occupied cell. These cells are called *travel cells* because their purpose is to suggest to the robot a path to follow.

4. *Far cells.* Any free cell (in the occupied–free space) that is not a warning or a travel cell.

In order to assign a higher cost to warning cells closer to obstacles, each warning cell must record, besides its type, the distance to the nearest occupied cell d_{min}. For travel and far cells it is enough to record the cell's type. A linear function is used to get the cost of a warning cell depending on the distance to the nearest occupied cell.

The travel space can be computed incrementally after each change of state of a cell in the occupied–free map while the robot is exploring the environment (Romero *et al.*, 2000). An example of a travel space is shown in Figure 3.

A policy to move to the unexplored cells following minimum–cost paths is computed using the travel space and a modified version of value iteration. The algorithm uses two variables, V and M, associated to each cell. $V(x,y)$ denotes the travel cost from cell (x,y) to the nearest unexplored cell. $M(x,y)$ represents the optimal movement to choose, given that the robot is in that cell. We consider 8 possible movements of the robot, one per cell in its vicinity. If $M(x,y) = (dx, dy)$, where dx is the change in x and dy is the change in y, the set of valid movements is $M_v = \{(1,0),(1,1),(1,0),(-1,1),(-1,0),(-1,-1),(0,-1),(1,-1)\}$. The idea is to associate costs to cells depending on its type. If warning cells and far cells have costs higher than travel cells, then a wall following strategy for exploration is taken into account.

For simplicity, cells of type *warning*, *travel* or *far*, will be call *free* cells in the travel space. Using the variables M and V, the algorithm has two steps:

1. Initialization. Unexplored cells (x,y) that are free in the travel space, are initialized with $V(x,y) = 0$, all the other explored cells that are free in the travel space are initialized with $V(x,y) = \infty$. All the free cells in the travel space are initialized with an undefined value to M.

2. Update. Let (x_r, y_r) be the position before the last movement of the robot. For all the explored free cells $(x,y) \neq (x_r, y_r)$ in c–space do:

$$V(x,y) \leftarrow \quad min_{(dx,dy)\in M_v}\{V(x+dx,y+dy) + Cost((x,y),(dx,dy)\}$$

$$M(x,y) \leftarrow \quad arg\_min_{(dx,dy)\in M_v}\{V(x+dx,y+dy) + Cost((x,y),(dx,dy)\}$$

where $Cost((x,y),(dx,dy))$ measures the cost of moving from the cell (x,y) to the cell $(x+dx,y+dy)$. This function punishes changes in direction and takes the value $C(x+dx,y+dy)+Dist((x,y),(x+dx,y+dy))+Kp_c$. Where $Dist(p_1,p_2)$ is the distance between cells p_1 and p_2 (1 or $\sqrt{2}$), and Kp_c represents the cost of the rotation of the robot to reach the next cell. $C(x,y)$ represents the cost associated with cell (x,y) in the travel space, based on its type. This assignment that punishes direction changes of the robot makes sense if we consider that rotation changes become a major source of uncertainty about the position of the robot.

The update rule is iterated, and when the values $(V(x,y),\ M(x,y))$ converge, the robot executes the movement indicated by M.

Exploration ends when $V = \infty$ for the cell where the robot is placed, which means that there is no way to reach an unexplored cell. Each time the value iteration algorithm is called, only the V values around the robot are initialized, in a similar way to the *bounding box* described in (Thrun, 1998).

There are two related works. In (Roy *et al.*, 1999), a *coastal navigation* method is described, once the PGM has been built. Each cell in the map contains a notion of information content available at that point in the map, which corresponds to the ability of the robot to localize itself. The information content is based on the concept of entropy and some assumptions are considered to reduce the complexity of the method. Our approach generates a similar form of coastal navigation with a simpler and more efficient method. The incremental algorithm developed in this paper also fits the time requirements for the exploration task. In another related work (Thrun, 1998), the probability of occupancy of the cell is used as a cost associated to the cells. The motion policy, given by the value iteration algorithm, needs to be postprocessed in order to keep the robot near to the center of narrow passages (but they do not describe how to do that). In our approach there is no need to modify the policy given by the value iteration algorithm. The local guides, in the form of costs, are inside the value iteration algorithm, so the policy is optimal given the costs associated to cells and the cost to move to an adjacent cell.

Figure 4: The real mobile robot

4 Position Tracking

Position tracking is the problem of estimating the robot position while it is moving, using its sensors. As noted in (Thrun *et al.*, 1998), position tracking is particularly difficult to solve if map learning is interleaved with localization. A recent survey in (Borenstein *et al.*, 1996) dedicated to this topic illustrates the importance of localization and the large number of existing approaches. The approach used in this paper is a simple version of the techniques described in (Weib *et al.*, 1994). The position tracking algorithm computes the actual location of the robot in two steps:

1. Rotation. From the last position of the robot a *laser visibility view* is computed from the map, using only the probabilities associated to the laser range sensors ($P(O_l(x,y))$. For reference, this view will be called a *map view*. Then, the map view is converted to a *polar map view* taking the robot position as the center, and using linear interpolation to get a smooth curve. In a similar way, a *polar sensor view* is obtained from the current laser range data (*sensor view*). The rotation between the last position and the new position can be estimated using a correlation between the polar map view and the polar sensor view.

2. Traslation. The position of the robot can be estimated using a correlation between the map and the sensor view, once the sensor view has the same orientation that the map view. In this correlation only the sensor readings above some threshold value Kd are considered

5 Experimental Results

This section presents the results obtained using a mobile robot simulator and a real mobile robot. The mobile robots have odometer, ultrasonic and a low cost laser range sensor (about 300 US$ vs 8000 US$ of *laser–based time of flight* range sensor). We use a Super Scout Mobile Robot with a ring of 16 sonars, a camera and a laser line generator (see Figure 4). See (Romero & Morales, 1999) for details about the sonar model implemented in the simulator. Figures

Figure 5: Laser range sensor operation. From left to right: (a) Image from the camera. (b) Image with the laser line generator turned on. (c) Image of (b) - (a)

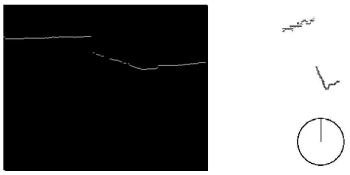


Figure 6: Laser range sensor operation (cont.). From left to right: (a) Image with a single laser point per column. (b) Obstacles in front of the robot

5 and 6 illustrate the laser range sensor operation. Fig. 5 (a) and (b) show images taken from the camera before and after the laser line generator is turned on. If we subtract image (a) from image (b) and apply a median filter (to deal with the noise), the laser points are emphasized, as it is shown in the Fig 5 (c). To get a single laser point per column of the image, we compute the *center of mass* of every column of the image, considering the gray level as the mass of each pixel (black pixels have no mass). The result of this operation is shown in Figure 6(a) (considering only points with a total mass above a given threshold value). Finally, Figure 6 (b) shows the mapping from the laser points in (a) to a map of obstacles in front of the mobile robot. Note the uncertainty associated to long readings. In fact, this behavior of the laser range sensor makes difficult the comparison of our map building method with other methods. Most works use *laser–based time of flight* ranging systems where the uncertainty associated to readings does not depend of the distance to the obstacle.

Figure 7 (a) shows the PGM built by the simulated robot without using the travel space (i.e. assigning null costs to warning and travel cells). The grid cells are $10\text{x}10cm^2$ and the map is $10\text{x}10m^2$. The simulator introduces an uniform random error on displacements of $\pm 10\%$ and a uniform random orientation error of about ± 7 degrees, per movement. The lighter trace on the map is given by the odometer and it shows the path followed by the robot. Note that sometimes the robot gets very close to obstacles. Figure 7 (b) shows the map built using the travel space and Fig. 7 (c) shows the map built using the real mobile robot

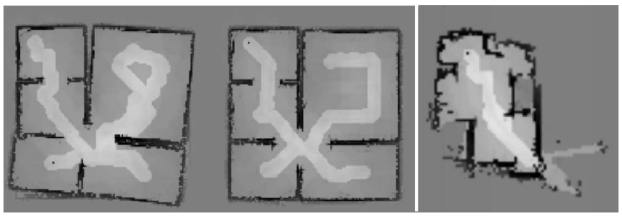


Figure 7: PGMs. White areas represent cells with occupancy probabilities near to 0. From left to right: (a) Using the simulator without the travel space. (b) Using the simulator and the travel space. (c) Using the real mobile robot and the travel space

within an office environment with desks, chairs, bookshelfs, etc. (note the effect of a glass door in the lower right corner). In these cases, warning cells have costs in the interval $[13, 1]$ (a linear function was used to estimate costs depending on the distance from the cell to the nearest occupied cell), travel cells have a cost of 0.001 and far cells have a cost of 6. The warning cells form a layer of 100 cm. and the travel cells form a layer of 20 cm. These cost values implement the wall following strategy to explore the environment, as can be observed in the map. Also the robot does not get too close to obstacles even in narrow passages. Instead, in narrow passages (where there are only warning cells) the robot tends to maximize the clearance between the robot and the obstacles. The map of the Fig. 7 (b) is also more accurate than the map built without using the travel space (Fig. 7(a)). This is because the travel space approach tends to move the robot to positions where the sensor readings are more reliable and hence the position tracking algorithm gives better estimations.

Some experiments were performed to evaluate the changes due to Kp_c, the cost of making orientation changes in the robot movements, during the exploration phase using the simulator. In these experiments we assign a cost of Kp_c for rotation of 45 degrees, $2Kp_c$ for 90 degrees, and so on. Table 1 shows the results, considering the length d (in cm.) of the path followed by the robot, the total number of movements made by the robot (n), the amount (θ) of orientation changes (in 45 degrees units) made by the robot, and the ratio (θ/n). These results suggest that higher Kp_c values tend to decrease the number of movements that change the orientation of the robot. The effect of Kp_c is analog to the effect of *inertial mass*: it tends to keep the orientation of the robot unchanged.

Kp_c	d	n	θ	θ/n
0	3352	284	162	0.5704
1	3640	310	163	0.5258
2	3596	304	155	0.5098
3	3536	306	145	0.4738

Table 1: Some experimental results for different costs of orientation changes (Kp_c)

6 Conclusions

A new method for map learning for indoor mobile robots using ultrasonic and laser range sensors was presented. This paper extends the approach described in (Howard & Kitchen, 1996) to fuse ultrasonic range data, gives a new probabilistic model of laser range sensors, and presents a method for integrating sensor data of different types. The experimental results show that this approach is adequate for sensor data fusion.

Additionally, a new approach for a mobile robot to explore in an indoor environment that combines local control (via cost associated to cells in the *travel space*) with a global exploration strategy (using a dynamic programming technique) has been described. As the experimental results confirm, the exploration follows a kind of *wall following* technique to reduce uncertainty in terms of localization, as well as to guide the robot through narrow passages maximizing the distance between the robot and the obstacles. This combination of local and global strategies takes the advantages of both: robustness of local strategies and completeness of global strategies. Also a heuristic to minimize the number of orientation changes, trying to minimize the accumulated odometric error, is also introduced.

We plan to explore some dynamic extensions to the travel space approach considering the specular degree of cells captured by the sonar model.

References

Borenstein, J., Everett, B., & Feng, L. *Navigating Mobile Robots: Systems and Techniques.* Wellesley, MA: A.K. Peter, Ltd. 1996.

Elfes, A. "Using Occupancy Grids for Mobile Robot Perception and Navigation". *IEEE Computer*, **22**(6), 46–57. 1989.

Gallistel, C. *The Organization of Learning.* MIT Press. 1990.

Gutmann, J.-S., Burgard, W., Fox, D., & Konolige, K. "An experimental Comparison of Localization Methods". *In: Proc. International Conference on Intelligent Robots and Systems (IROS'98).* 1998.

Howard, H., & Kitchen, L. "Generating Sonar Maps in Highly Specular Environments". *In: Proceedings of the Fourth International Conference on Control, Automation, Robotics and Vision.* 1996.

Latombe, J-C. *Robot Motion Planning.* Kluwer Academic Publishers. 1991.

Lee, D. *The Map–Building and Exploration of a Simple Sonar–Equipped Robot.* Cambridge University Press. 1996.

McKerrow, P. J. *Introduction to Robotics.* Addison-Wesley. 1991.

Moravec, H. P. "Sensor Fusion in Certainty Grids on Mobile Robots". *AI Magazine*, **9**(2), 61–74. 1988.

Romero, L., & Morales, E. "Uso de una red neuronal para la fusion de lecturas de sonares en robots moviles". *In: Segundo Encuentro Nacional de Computacion (ENC99) Mexico.* 1999.

Romero, L., Morales, E., & Sucar, E. "A Robust Exploration and Navigation Approach For Indoor Mobile Robots Merging Local and Global Strategies". *In: IBERAMIA, LNIA.* Springer. 2000.

Roy, Nicholas, Burgard, Wolfram, Fox, Dieter, & Thrun, Sebastian. "Coastal Navigation – Mobile Robot Navigation with Uncertainty in Dynamic Environments". *In: Proc. IEEE Conf. Robotics and Automation (ICRA).* 1999 (May).

Shigang, L., Ichiro, M., Hiroshi, I., & T., Saburo. "Finding of 3D Structure by an Active–vision–based Mobile Robot". *In: Proceedings of the IEEE International Conference on Robotics and Automation.* 1992.

Thrun, S. "Learning Maps for Indoor Mobile Robot Navigation". *Artificial Intelligence*, **99**(1), 21–71. 1998.

Thrun, S., Bucken, A., Burgar, W., *et al.* "Map Learning and High-Speed Navigation in RHINO". *In:* **Kortenkamp, D., Bonasso, R. P., & Murphy, R** (eds), *Artificial Intelligence and Mobile Robots.* AAAI Press/The MIT Press. 1998.

Weib, G., Wetzler, C., & Puttkamer, E. "Keeping Track of Position and Orientation of Moving Indoor Systems by Correlation of Range-Finder Scans". *In: Intelligent Robots and Systems.* 1994.