

An Hybrid Approach to Solve the Global Localization Problem For Indoor Mobile Robots Considering Sensor's Perceptual Limitations

Leonardo Romero, Eduardo Morales and Enrique Sucar

ITESM, Campus Morelos, Temixco, Morelos, 62589, Mexico
lromero@zeus.umich.mx, {emorales,esucar}@campus.mor.itesm.mx

Abstract

Global localization is the problem of determining the position of a robot under global uncertainty. This problem can be divided in two phases: 1) from the sensor data (or *sensor view*), determine the set of locations where the robot can be; and 2) devise a strategy by which the robot can correctly eliminate all but the right location. The approach proposed in this paper is based on Markov localization. It applies the principal component method to get rotation invariant features for each location of the map, a Bayesian classification system to cluster the features, and polar correlations between the *sensor view* and the *local map views* to determine the locations where the robot can be. In order to solve efficiently the localization problem, as well as to consider the perceptual limitation of the sensors, the possible locations of the robot are restricted to be in a roadmap that keep the robot close to obstacles, and correlations between the possible local map views are pre-computed. The hypotheses are clustered and a greedy search determine the robot movements to reduce the number of clusters of hypotheses. This approach is tested using a simulated and a real mobile robot with promising results.

1 Introduction

Numerous tasks for a mobile robot require it to have a map of its environment and knowledge of where it is located in the map. Determining the location (position and orientation) of the robot in the environment is known as the *robot localization problem*. A recent survey [Borenstein *et al.*, 1996] illustrates the importance of the localization problem and illustrates the large number of existing approaches.

Traditionally, localization addresses two subproblems: *position tracking* and *global localization* [Thrun *et al.*, 1998]. Position tracking refers to the problem of estimating the location of the robot while it is moving. Drift and slippage impose limits on the ability to estimate the location of the robot within its global map. The robot knows its initial location and tracks its location after small movements using sensor data. Global localization is the problem of determining the location of the robot under global uncertainty. This problem

arises, for example, when a robot uses a map that has been generated in a previous run, and it is not informed about its initial location within the map.

The global localization problem can be seen as consisting of two phases: hypothesis generation and hypothesis elimination [Dudek *et al.*, 1998]. The first phase is to determine the set of *hypothetical locations* H that are consistent with the sensing data obtained by the robot at its initial location. The second phase is to determine, in the case that H contains two or more hypotheses, which one is the true location of the robot, eliminating the incorrect hypotheses. Ideally, the robot should travel the minimum distance necessary to determine its exact location.

This paper presents an approach to solve the global localization problem in a known indoor environment modeled by an occupancy grid map, a two dimensional map where the environment is divided in square regions or cells of the same size. This approach is based on *Markov localization* (see [Gutmann *et al.*, 1998]) and takes into account the perceptual limitations of sensors in two ways: 1) the robot tries to keep a fixed distance to obstacles while the robot is moving and 2) the sensors have a limited range of operation. Following the approach described in [Romero *et al.*, 2000] to build a *roadmap* using the first restriction, the number of possible locations where the robot can be is significantly reduced. Taking into account the second restriction, *local map views* are computed from the map for each cell of the roadmap. The general form of each local map view can be compressed into two real numbers using the *principal component* method [Gonzalez and Wintz, 1987] with the advantage that these features are rotation independent. These compressed views are then clustered by Autoclass [Cheeseman and Stutz, 1996], a Bayesian classification system. The idea to generate the hypotheses is to compute the principal components of the sensor data (or *sensor view*), then use Autoclass to predict the best class (or classes) and finally perform polar correlations between the *sensor view* and the predicted local map views. To eliminate hypotheses, a greedy search is performed upon the correlations of the local map views of the same class.

The rest of this paper is organized as follows. Section 2 reviews some related relevant literature. Section 3 describes our approach to generate hypotheses. Section 4 presents the framework of Markov localization. Section 5 explains the approach to eliminate hypotheses. Experimental results using

a mobile robot simulator and a mobile robot with a low cost laser range sensor (implemented by a laser line generator and a camera) are shown in Section 6. Finally, some conclusions are given in Section 7.

2 Related work

Theoretical work on the subject has been reported in the literature. A solution to the hypothesis generation phase is given in [Guibas *et al.*, 1995]. In [Dudek *et al.*, 1998] it is shown that the problem of localizing a robot with minimum travel is NP-hard. These works make strong assumptions: first, perfect knowledge about the orientation of the mobile robot is considered; second, they assume the availability of noise-free sensor data.

Practical work has also been reported (see [Thrun, 1998; Castellanos and Tardos, 1999]). *Landmark methods* rely on the recognition of landmarks to keep the robot localized geometrically. While landmark methods can achieve impressive geometric localization, they require efficient recognition of features to use as landmarks. In contrast, *dense sensor methods* attempt to use whatever sensor information is available to update the location of the robot. They do this by matching dense sensor scans against a surface map of the environment, without extracting landmark features. See [Gutmann *et al.*, 1998] for an experimental comparison of two dense sensor matching methods: Kalman Filter and Markov localization. Some correlation techniques of laser range-finder scans are given in [Weib *et al.*, 1994]. In [Fox and Burgard, 1998] a method to move the robot in order to eliminate hypotheses is given. The idea is to move the robot so as to minimize future expected uncertainty, measured by the entropy of future belief distributions.

3 Hypothesis generation

In this section and the two following sections a simple occupancy grid map is used as an example to show the ideas behind the proposed approach. Figure 1 (a) shows this simple map built using a mobile robot simulator. Figure 1 (b) shows a local map view, extracted from the map for the position of the robot shown in (a), considering that the robot direction is aligned with a global fixed direction (pointing downwards in this case). Figure 1 (c) shows the sensor view (generated by the simulator) considering that the robot is aligned with the global direction. Both views have an angular resolution of 0.5 degrees, a common value found in laser range sensors, and include the robot position for reference as a black cell. Perceptual limitations are taken into account setting a maximum range of 3 meters.

The problem consist in estimating the set H of possible locations that have local map views consistent with the sensor view.

3.1 Polar correlation

To have a simple model of robot motion and hence a small state space in the Markov localization, we assume that the robot should be in one of 8 possible directions ($\theta_i = 45 * i$ degrees, $i = 0, ..7$), with respect to the global fixed direction, one for each adjacent cell. A polar correlation, using a sum of

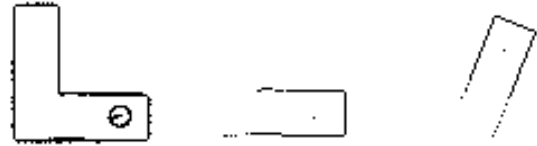


Figure 1: A simple environment. From left to right: (a) Occupancy grid map. (b) Local map view for the robot location showed in (a). (c) Sensor view

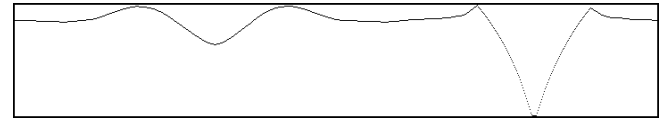


Figure 2: Correlation results. Angular displacements are from 0 (left) to 359.5 degrees (right).

absolute differences, can be used to find the match between a local map view and the sensor view. Figure 2 shows the correlation results for all the possible angular displacements of the sensor view against the local map view shown in Figure 1. From the minimum difference an angular displacement can be computed to align the robot with one of the directions θ_i . Obviously, the right angular displacement should be indicated considering the most probable position of the robot. In the case of Figure 2 the angular displacement corresponds to -21 degrees, and the best estimated direction is 270 degrees.

As the Markov localization needs a probabilistic value $p(s|l)$ of perceiving a sensor view s given that the robot is at location l (cell $\langle x, y \rangle$ with direction θ_i) a difference value $d(x, y, \theta_i)$ can be computed for direction θ_i from the correlation results between the sensor view s and the local map view at the cell $\langle x, y \rangle$, and then a probabilistic value can be obtained from $d(x, y, \theta_i)$. We compute $d(x, y, \theta_i)$ as the minimum difference (in the correlation results) for an angular interval with center at θ_i . The desired probability is computed by $p(s|l) = e^{-\alpha d(x, y, \theta_i)}$ where α is positive real number.

Considering that this procedure is expensive, next sections show a fast procedure to find a small set of candidate cells to apply this procedure, instead of all the free cells in the map.

3.2 Roadmap

Following the ideas described in [Romero *et al.*, 2000], the set of possible cells, where the robot is allowed to move, tries to keep a fixed distance k to obstacles. It associates higher costs to free cells close to obstacles and the lowest costs to cells at distance k from obstacles. Figure 3 (a) shows the full set of free cells where the robot can be as white pixels, while Figure 3 (b) shows the cells that form the roadmap. There is a significant reduction in the number of cells. Let R be the set of cells of the roadmap.

In the next section we are going to consider only the cells $\langle x, y \rangle \in R$.

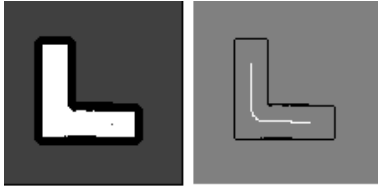


Figure 3: A roadmap. From left to right: (a) Full set of free cells. (b) The roadmap for $k = 1$ m.

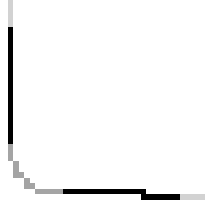


Figure 4: Clusters found by Autoclass. Different gray level are associated to different classes

3.3 Principal component analysis

We apply the principal component analysis described in [Gonzalez and Wintz, 1987] to find two rotation invariant features from a local map view or sensor view. The coordinates of each occupied cell of the view can be interpreted as two dimensional random variables with mean $\mathbf{m}$ and covariance $\mathbf{C}$. It is noted that $\mathbf{m}$ is a two dimensional vector and $\mathbf{C}$ is a 2×2 matrix. The eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ of $\mathbf{C}$ point in the directions of maximum variance (subject to the constraint that they are orthogonal) and the eigenvalues λ_1 and λ_2 are equal to the variance along eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ respectively. Our approach uses the two eigenvalues as rotation invariant features for a local map view or sensor view.

3.4 Bayesian clustering

Since the eigenvalues associated to adjacent cells of the roadmap normally do not change a lot, it makes sense to cluster cells according to their eigenvalues. For this purpose we use Autoclass [Cheeseman and Stutz, 1996]. Figure 4 show with different gray level the three classes found for the cells of the roadmap shown in Figure 3 (b). Autoclass found the same class for cells close to the end cells of the roadmap, another class for cells in the middle of the roadmap and the rest of cells in a third class.

At the beginning of the localization process, to take the robot to the roadmap, we apply the algorithm described in [Romero *et al.*, 2000] to learn a map of the environment. Using that approach it is easy to know if the robot is in the roadmap or need to execute some movements. Joining all these parts, the hypothesis generation does the following steps:

1. Compute the eigenvalues $\mathbf{e}$ from the sensor view.
2. With $\mathbf{e}$, use Autoclass to predict the most probable classes.

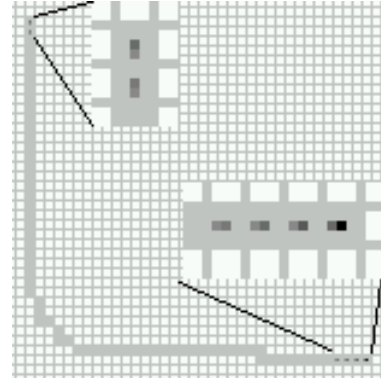


Figure 5: Most probable locations for the robot

3. For all cells in the classes found in the previous step, apply the correlation technique to get the angular displacement that align the robot with one of the possible directions and compute the conditional probabilities $p(s|l)$. The conditional probability for other cells of the roadmap are set to a very small value.

For the sensor view shown in Figure 1 (c) the conditional probabilities are shown in Figure 5. If the probability of finding the robot at location l of the roadmap is equal for all the cells in the roadmap, the same figure shows the most probable locations for the robot. The locations in the left top part of the figure point downwards and the locations in right bottom part of the figure point to the left. The following section describes the process to update the probability of hypotheses after the robot senses or moves.

4 Markov Localization

The key idea of Markov localization is to compute a probability distribution over all possible locations in the environment. $p(L_t = l)$ denotes the probability of finding the robot at location l at time t . Here, l is a location in $x - y - \theta_i$ space where x and y are Cartesian coordinates of cells and θ_i is a valid orientation. $p(L_0)$ reflects the initial state of knowledge and it is uniformly distributed to reflect the global uncertainty. $p(L_t)$ is updated whenever ...

1. ... the robot moves. Robot motion is modeled by a conditional probability, denoted by $p_a(l|l')$. $p_a(l|l')$ denotes the probability that motion action a , when executed at l' , carries the robot to l . $p_a(l|l')$ is used to update the belief upon robot motion:

$$p(L_{t+1} = l) \leftarrow \frac{\sum_{l'} p_a(l|l') p(L_t = l')}{p(s)} \quad (1)$$

Here $p(s)$ is a normalizer that ensures that $p(L_{t+1})$ sums up to 1 over all l . An example of $p_a(l|l')$ is shown in Figure 6, considering that the robot moves in the thick roadmap, and the orientation of the robot is aligned to one of the possible 8 directions θ_i , as it was described in a previous section.

2. ... the robot senses. When sensing s ,

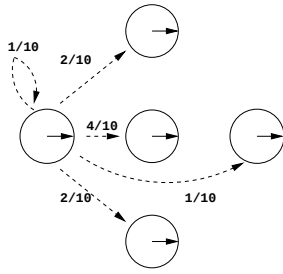


Figure 6: Robot motion for one direction

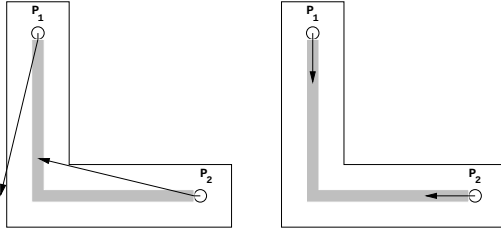


Figure 7: Hypotheses after a movement. From left to right: (a) One hypotheses. (b) Two hypothesis

$$p(L_{t+1} = l) \leftarrow \frac{p(s|l)p(L_t = l)}{p(s)} \quad (2)$$

Here $p(s|l)$ is the probability of perceiving s at location l . In order to get a efficient procedure to update the probability distribution, cells with probability below some threshold are set to zero.

To get a robust and efficient procedure, our approach considers a *thick* roadmap which include the cells in the neighborhood of the roadmap (*thin roadmap*) besides the cells in the roadmap. The possible hypotheses for the robot are restricted to be in the thick roadmap.

5 Hypothesis elimination

Let $p(L_t = l)$ be the probability of finding the robot at location l at time t . We consider as hypotheses only those locations l with $p(L_t = l) > \beta$, where β is a threshold value. Let n be the number of hypotheses.

To eliminate hypotheses, the robot should move through the roadmap. After a robot move, there are two different scenarios for the cells that correspond to the different hypotheses: 1) the cells are in the roadmap under all the hypotheses, and 2) the cells are in the roadmap for less than n hypotheses. In the first scenario, the movement does not help to eliminate hypotheses; in the second scenario the movement helps to decrease the number of hypotheses. Figure 7 shows an example of both scenarios. If there are two hypotheses P_1 and P_2 then the movement, *represented by a rotation followed by a translation*, showed in Figure 7 (a) helps to reduce the number of hypotheses while the other movement is not useful.

To get an efficient procedure, our approach considers that the mobile robot is at the most probable location and then considers all the cells of the thin roadmap as valid movements

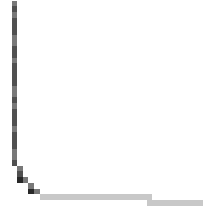


Figure 8: Number of hypotheses associated to cells of the roadmap using the roadmap criteria. Darker cells have lower number of hypotheses

for the robot. If we assign to locations the number of possible hypotheses, a good movement to eliminate hypotheses is to direct the robot toward the nearest cell with less than n hypotheses.

If we consider the hypotheses shown in Figure 5, Figure 8 shows the number of possible hypothesis as an image. Darker cells have lower values for the number of hypotheses. In this case the best hypothesis is located in the right bottom part of the roadmap and it is pointing to the left. The robot should move to the nearest darker cell of the roadmap. Let this cell be called the *goal* cell.

If we have a criteria to consider two local map views as *similar*, we can use it to detect shorter movements than using only the previous criteria. The two scenarios discussed have two variants: a) the cells are *similar* considering its local map views, and b) not all the cells look similar. The robot should move to a cell with scenario (b).

A similarity measure can be computed using the correlation technique previously presented. Two cells look similar if the computed probability is above some threshold. If there are m cells in the roadmap, let c_i , ($i = 1, \dots, m$) denote the m cells of the roadmap, and $\text{sim}(c_i, c_j)$ be the similarity between the local map view associated to cells c_i and c_j . $\text{sim}(c_i, c_j)$ for all $i, j = 1, \dots, m$ form a *similarity matrix* and it can be computed from the map and the roadmap, before the localization process starts. To compute this matrix we can take advantage of the fact that this matrix is symmetric and have zero values in the main diagonal.

Figure 9 shows the similarity matrix for our example. In this case, the cells of the thin roadmap were ordered following a depth first search trying to keep cells with adjacent indices associated to adjacent cells in the roadmap. The image shows that there are only a few cells with significant similarities that are not close to each other. This makes sense if we consider the map shown in Figure 1 and the limited range of the sensors.

Two cells c'_i and c'_j in the thick roadmap are considered similar if $\text{sim}(c_i, c_j)$ is above some threshold, where c_i and c_j are the closest cells in the thin roadmap to c'_i and c'_j respectively.

If we consider the hypotheses shown in Figure 5 and the similarity criteria, Figure 10 shows the number of possible hypotheses as an image. In this case, the goal cell is closer to the starting location than the one in Figure 8.

Another improvement over the last criteria is to cluster the

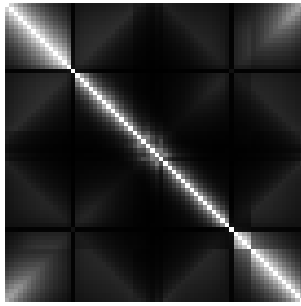


Figure 9: Similarity matrix. Darker pixels mean lower similarities

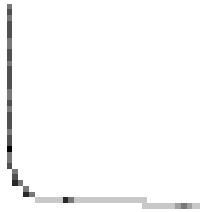


Figure 10: Number of hypotheses associated to cells of the roadmap using the roadmap and the similarity criteria. Darker cells have lower number of hypotheses

hypotheses and then the goal cell is the cell that reduces the number of clusters. In other words, the best movement for the robot should reduce the number of clusters, not the total number of hypotheses. For instance, if there are two clusters (c_1 , c_2) and c_1 has 4 hypotheses and c_2 has only 1 hypothesis, and the most probable hypothesis is in cluster c_1 , then a movement that eliminates one hypothesis in cluster c_2 is better than one movement that eliminates one or two hypotheses in cluster c_1 .

For the clustering process we use Autoclass with the coordinates of the hypotheses. If there is only one cluster, then the goal cell is the nearest cell that reduces the number of hypotheses. The robot is localized if there is only one cluster and there is no cell that reduces the number of hypotheses.

Once a goal cell is found, the robot should perform some movements towards that cell, updating the probability distribution.

6 Experimental Results

This section presents preliminary results obtained using a mobile robot simulator and a real mobile robot. Both robots use (or simulate) sonars and a low cost laser range sensor, implemented with a laser line generator and a camera. The laser sensor gives good measurements within a range of 3 m.

Figure 11 (a) shows a more complex simulated environment. Figure 11 (b) shows the map built with its roadmap. The map is about $10 \times 10 \text{ m}^2$ and the roadmap considers a distance to obstacles of 1m.

Figure 12 shows 7 clusters found by Autoclass, and Figure 13 shows the associated similarity matrix. The symmetric nature of the map is captured in the similarity matrix.

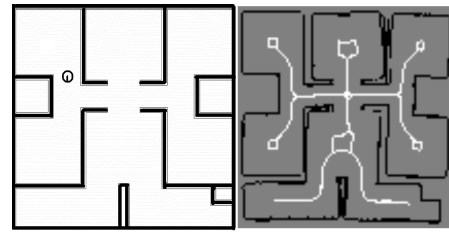


Figure 11: A more complex map built using a mobile robot simulator. From left to right: (a) Simulated environment. (b) Map built with its roadmap

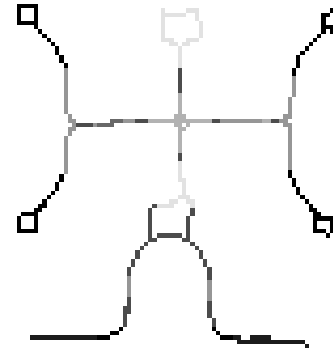


Figure 12: Clusters found by Autoclass for environment of the Figure 11.

If the robot is located as it is shown in Figure 11 (a), then Figure 14 shows the most probable locations for the robot. Autoclass found two clusters C_1 (pointing downwards) and C_2 (pointing upwards) and the location with the highest probability is in C_1 . The hypotheses elimination process found a goal cell pointed by G in the figure. This result makes sense because the possible hypotheses after that movement have different sensor views.

Figure 15 shows a map built using the real mobile robot. Autoclass found in this case 6 clusters and the similarity matrix is shown in Figure 15. In this real environment the robot localization is easier than in the previous simulated environ-

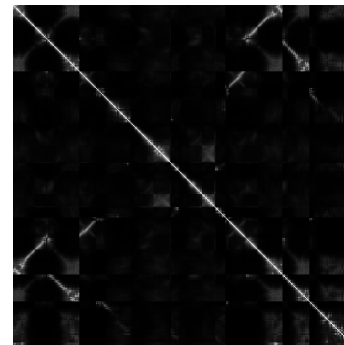


Figure 13: Similarity matrix for environment of the Figure 11. Darker pixels mean lower similarities

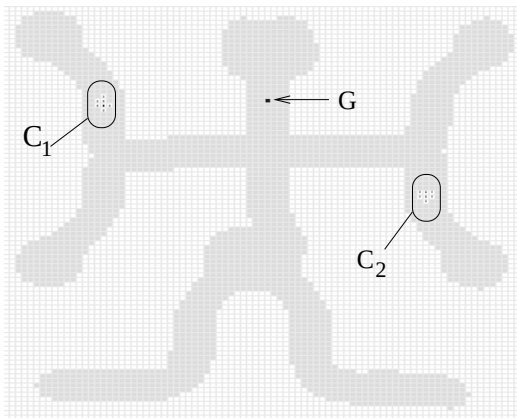


Figure 14: Most probable locations for the robot for the environment showed in Figure 11 (a).

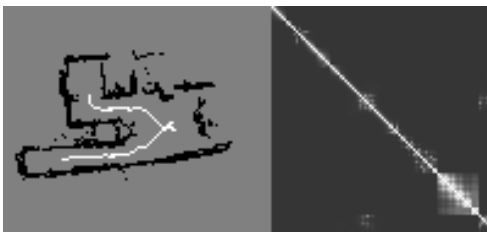


Figure 15: A real mobile robot in a real environment. From left to right: (a) Map and its roadmap. (b) Similarity matrix

ment.

7 Conclusions

An hybrid approach to solve the global localization problem in indoor environments has been presented. It can be seen as a merge of a landmark approach and a dense sensor method based on the Markov localization. Landmarks are generated and recognized by a Bayesian clustering technique (Auto-class) from rotation invariant features (principal components) of sensor or local map views. These landmarks are used to restrict the set of possible locations for the robot and so increase the speed of the Markov localization process.

In the hypothesis elimination phase a fast greedy search is introduced. This search uses a similarity matrix to compute a similarity criteria between the map views of cells. The search considers movements towards cells of the thin roadmap that reduce the number of clusters of hypotheses.

To get an efficient localization algorithm, the approach uses the thick roadmap concept, instead of the full set of free cells; restricts the number of possible orientations of the robot, takes advantage of a correlation technique to align the robot and pre-compute the similarity matrix.

We plan to explore a non deterministic kind of movement in the hypotheses elimination phase using the Markov framework to perform sequence of (move-sense) for each cluster of hypotheses. We think this kind of movements will be able to predict a goal cell even when the map is not very accurate or in the case of robots with high odometric errors.

References

- [Borenstein *et al.*, 1996] J. Borenstein, B. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques*. A.K. Peter, Ltd., Wellesley, MA, 1996.
- [Castellanos and Tardos, 1999] J.A. Castellanos and J. D. Tardos. *Mobile robot localization and map building: a multisensor fusion approach*. Kluwer Academic Publishers, 1999.
- [Cheeseman and Stutz, 1996] P. Cheeseman and J Stutz. Bayesian classification (autoclass): Theory and results. In U.-M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. AAAI Press/MIT Press, 1996.
- [Dudek *et al.*, 1998] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. *SIAM J. Comput*, 27(2):583–604, 1998.
- [Fox and Burgard, 1998] D. Fox and W. Burgard. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 1998.
- [Gonzalez and Wintz, 1987] R. C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison-Wesley, 2 edition, 1987.
- [Guibas *et al.*, 1995] L. Guibas, R. Motwani, and P. Raghavan. The robot localization problem. In K. Goldberg, D. Halperin, and J. C. Latombe, editors, *Algorithmic Foundations of Robotics*. A. K. Peters, 1995.
- [Gutmann *et al.*, 1998] J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *Proc. International Conference on Intelligent Robots and Systems (IROS'98)*, 1998.
- [Romero *et al.*, 2000] L. Romero, E. Morales, and E. Sucar. A robust exploration and navigation approach for indoor mobile robots merging local and global strategies. In M. C. Monard and J.S. Sichman, editors, *Advances in Artificial Intelligence, LNAI 1952*. Springer, 2000.
- [Thrun *et al.*, 1998] S. Thrun, A. Bucken, W. Bugar, et al. Map learning and high-speed navigation in rhino. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *Artificial Intelligence and Mobile Robots*. AAAI Press/The MIT Press, 1998.
- [Thrun, 1998] S. Thrun. Learning maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.
- [Weib *et al.*, 1994] G. Weib, C. Wetzler, and E. Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. In *Intelligent Robots and Systems*, 1994.