Autonomous Robots 19, 53–65, 2005

Mobile Robot Simultaneous Localization and Mapping in Dynamic Environments*

DENIS F. WOLF AND GAURAV S. SUKHATME
Robotic Embedded Systems Laboratory, Center for Robotics and Embedded Systems, Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA
denis@robotics.usc.edu
gaurav@robotics.usc.edu

Abstract. We propose an on-line algorithm for simultaneous localization and mapping of dynamic environments. Our algorithm is capable of differentiating static and dynamic parts of the environment and representing them appropriately on the map. Our approach is based on maintaining two occupancy grids. One grid models the static parts of the environment, and the other models the dynamic parts of the environment. The union of the two grid maps provides a complete description of the environment over time. We also maintain a third map containing information about static landmarks detected in the environment. These landmarks provide the robot with localization. Results in simulation and real robots experiments show the efficiency of our approach and also show how the differentiation of dynamic and static entities in the environment and SLAM can be mutually beneficial.

Keywords: mapping, occupancy grids, dynamic environments, SLAM

1. Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in mobile robotics and has been studied extensively in the robotics literature (Thrun et al., 1998, 2000; Castellanos et al., 2001; Guivant and Nebot, 2001, 2002; Tardos et al., 2004; Newman et al., 2002; Montemerlo et al., 2002; Nieto et al., 2002; Liu and Thrun, 2002; Fenwick et al., 2002; Haehnel et al., 2003). For the most part, research has concentrated on SLAM in static environments. In this article, which is an extended version of Wolf and Sukhatme (2004), we explicitly consider the SLAM problem in dynamic environments. The presence of moving objects in the environment can lead SLAM algorithms to mistakes and result in incorrect maps. The explicit identification of dynamic entities can certainly improve the accuracy of the localization and mapping processes.

The approach presented in this paper is divided into two parts: mapping (and detection of) dynamic environments (i.e. maintaining separate representations for the dynamic and static parts of the environment), and robot localization. These two tasks are interleaved, allowing the robot to perform simultaneous localization and mapping.

Our mapping algorithm extends the occupancy grid technique introduced in Elfes (1986, 1989) and Moravec (1988) to dynamic environments. The resulting algorithm is capable of detecting dynamic objects in the environment and representing them on a map. Non-stationary objects are detected even when they move out of the robot's field of view, when the robot revisits the already mapped area where the changes happened. In order to do this, we maintain two occupancy grids maps. One map (S) is used to represent occupancy probabilities which correspond to the static parts of the environment and the other map (D) is used to represent occupancy probabilities of the moving parts of the environment. A complete description of the environment is obtained by the union of the information

*Part of this work has been presented in the International Conference in Robotics and Automation—ICRA 2004.

present in the two maps ($S \cup D$). In this case, the non-static entities will appear in the last position that they were seen. This information is updated when the robot revisits these already mapped areas.

Our localization algorithm is based on the well-known SLAM approach by Dissanayake et al. (2001). We use an Extended Kalman Filter (EKF) to incrementally estimate the correct position of the robot and landmarks in the environment. Corners have been used as landmarks.

Experimental tests have been performed using ActiveMedia Pioneer Robots equipped with laser range finders in the California Science Center in Los Angeles. The results show that our algorithm is able to successfully differentiate dynamic and static parts of the environment, and simultaneously localize the robot.

2. Related Work

Mapping of static environments has received considerable attention recently, but in most cases, these algorithms cannot be directly applied to dynamic environments. A good overview of mapping methods is given in Thrun (2002). Examples of localization in dynamic environments can be seen in Aycard et al. (1998) and Fox et al. (1999).

Usually, the presence of moving objects causes errors and compromises the overall quality of the map. This is a considerable problem since many realistic robot applications are in non-static environments. Mapping dynamic environments has been addressed in recent years (Haehnel et al., 2002, 2003; Wang and Thorpe, 2002; Wang et al., 2003; Biswas et al., 2002; Wolf and Sukhatme, 2003; Andrade-Cetto and Sanfeliu, 2002; Avots et al., 2002), but still has many open questions. These include, how to differentiate static and dynamic parts of the environment and, how to represent such information in the map.

Before discussing the details regarding related approaches to this problem, we introduce a classification of objects in a dynamic environment. There are two types of moving objects to be considered: objects that are permanently in motion and objects that are stationary part of the time (some times most of the time) and move occasionally. The first category refers to objects that have changed location each time they are observed by the robot's sensors. The second category refers to objects that may or may not have moved since they were observed previously. They are however known to have moved at least once. In this work, when an object moves at least once, it will be always considered dynamic. The approach presented in this paper deals with both categories of moving objects. In Haehnel et al. (2002), a technique is presented to identify moving objects in the environment based on the difference of consecutive sensor readings. The objective is filter out those sensor readings corresponding to dynamic objects in order to improve the quality of the map. Sample-based Joint Probabilistic Data Association Filters have been used. In Wang and Thorpe (2002) a combination of SLAM and DTMO (Detection and Tracking of Moving Objects)is proposed in which scan matching algorithms are used to SLAM and moving objects are detected and tracked. In Wang et al. (2003) a framework is given for solving the simultaneous mapping, localization, detection and tracking of moving objects. The idea is identify and keep track of moving objects in order to improve the quality of the map. All the approaches cited above only deal with objects that move in the field of view of the robot. They do not address changes in the environment when those changes occurred behind the robot's back.

As far as we know, the only works that deal with this case are (Biswas et al., 2002; Wolf and Sukhatme, 2003; Haehnel et al., 2003). The work presented in Biswas et al. (2002) uses an off-line Bayesian approach (based on the Expectation-Maximization algorithm) that can detect changes over time in an environment. The basic idea of this approach rests on a map differencing technique. Maps of the same environment are created at different points in time. By comparing those maps, the algorithm is able to identify the parts of the environment that changed over time.

The approach in Haehnel et al. (2003) uses the EM algorithm to differentiate dynamic and static parts of the environment off-line. In the expectation step it estimates which measurements might correspond to static parts of the environment. In the maximization step, the position of the robot in the map is calculated. The algorithm iterates until no further improvement can be achieved. Both Biswas et al. (2002) and Haehnel et al. (2003) are off-line methods.

In prior work (Wolf and Sukhatme, 2003), we presented an on-line mapping algorithm capable of differentiating static and dynamic parts of the environment even when the moving objects change position out of the field of the view of the robot. The algorithm could also uniquely classify each moving object and keep track of its location on the map. On the other hand, this approach assumed ideal localization, which is a fairly narrow assumption.

3. The Mapping Approach

In our mapping approach, two distinct occupancy grid maps (S and D) are used. The static map S contains only information about the static parts of the environment such as the walls and other obstacles that have never been observed to move. The dynamic map D contains information about the objects which have been observed to move at least once.

In the static map S, the occupancy probability of a cell represents the probability of a static entity being present at that cell. As dynamic entities are not represented in this map, if that cell is occupied by an moving object, the occupancy probability will indicate a free space (i.e. not occupied by a static entity). In the same manner, static parts of the environment are not represented the dynamic map D. Thus when a cell in D has a occupancy probability indicating free space, it means simply that no moving entity is currently occupying the cell. It does not exclude the possibility of the cell being occupied by a static part of the environment. By the use of these two maps, the algorithm presented here is able to detect moving objects even if these objects move out of the view of the robot in an already mapped area.

In our formulation both maps S and D are independently updated. The set of sensor readings is represented by o. We use the discrete time index as superscript of the variables, for example o^t means the sensor readings at time t. Therefore, the problem is to estimate the following:

$$p(S^t|o^1 \dots o^t) \tag{1}$$

$$p(D^t|o^1 \dots o^t) \tag{2}$$

The first step in order to correctly update both maps S and D is to differentiate static and dynamic entities in the environment. This can be performed if we include some previous information about the static parts of the environment (S^{t-1}) in Eqs. (1) and (2). As the static parts of the environment never change over time, they can be used as a reference to determine which sensor readings are generated by static and dynamic obstacles. Dynamic parts of the environment cannot be used for this purpose. Thus Eqs. (1) and (2) become:

$$p(S^t \mid o^1 \dots o^t, S^{t-1}) \tag{3}$$

$$p(D^t \mid o^1 \dots o^t, S^{t-1}) \tag{4}$$

We are interested in estimating these quantities above, thus updating the static and dynamic maps.

3.1. *Static Map Update*

The update equation for the static map S (Eq. 3) is slightly different from the regular occupancy grid technique, which assumes the environment does not change over time. Since we explicitly deal with situations in which parts of the environment change their position over time, we use some previous knowledge about the environment and compare with the current set of observations in order to differentiate static and dynamic obstacles and keep only the static parts of the environment in the map S.

Applying Bayes rule to Eq. (3) we obtain:

$$p(S^t \mid o^1 \dots o^t, S^{t-1}) = \frac{p(o^t|o^1 \dots o^{t-1}, S^{t-1}, S^t) \cdot p(S^t|o^1 \dots o^{t-1}, S^{t-1})}{p(o^t \mid o^1 \dots o^{t-1}, S^{t-1})} \tag{5}$$

As we are mapping the static part of the environment, in the term $p(o^t \mid o^1 \dots o^{t-1}, S^{t-1}, S^t)$ we assume that all information from previous observations ($o^1 \dots o^{t-1}$) is incorporated in S^{t-1} therefore the term $o^1 \dots o^{t-1}$ can be dropped. It is possible to rewrite that term as $p(o^t \mid S^{t-1}, S^t)$ and applying this modification to Eq. (5), we obtain:

$$p(S^t \mid o^1 \dots o^t, S^{t-1}) = \frac{p(o^t|S^{t-1}, S^t) \cdot p(S^t|o^1 \dots o^{t-1}, S^{t-1})}{p(o^t \mid o^1 \dots o^{t-1}, S^{t-1})} \tag{6}$$

Applying Bayes rule to the first term of Eq. (6), we obtain:

$$p(S^t|o^1 \dots o^t, S^{t-1}) = \frac{p(S^t|o^t, S^{t-1}) \cdot p(o^t|S^{t-1}) \cdot p(S^t|o^1 \dots o^{t-1}, S^{t-1})}{p(S^t|S^{t-1}) \cdot p(o^t|o^1 \dots o^{t-1}, S^{t-1})} \tag{7}$$

Following an analogous derivation, it is possible to calculate the non-occupancy of S, denoted by $\bar{S}$.

$$p(\bar{S}^t|o^1 \dots o^t, S^{t-1}) = \frac{p(\bar{S}^t|o^t, S^{t-1}) \cdot p(o^t|S^{t-1}) \cdot p(\bar{S}^t|o^1 \dots o^{t-1}, S^{t-1})}{p(\bar{S}^t|S^{t-1}) \cdot p(o^t|o^1 \dots o^{t-1}, S^{t-1})} \tag{8}$$

Dividing Eq. (7) by Eq. (8), we can eliminate some terms, obtaining:

$$\frac{p(S^t|o^1 \dots o^t, S^{t-1})}{p(\bar{S}^t|o^1 \dots o^t, S^{t-1})} = \frac{p(S^t|o^t, S^{t-1})}{p(\bar{S}^t|o^t, S^{t-1})} \cdot \frac{p(\bar{S}^t|S^{t-1})}{p(S^t|S^{t-1})} \cdot \frac{p(S^t|o^1 \dots o^{t-1}, S^{t-1})}{p(\bar{S}^t|o^1 \dots o^{t-1}, S^{t-1})} \tag{9}$$

We can rewrite Eq. (9) substituting $p(\bar{S})$ by $1 - p(S)$.

$$\frac{p(S^t|o^1 \dots o^t, S^{t-1})}{1 - p(S^t|o^1 \dots o^t, S^{t-1})} = \frac{p(S^t|o^t, S^{t-1})}{1 - p(S^t|o^t, S^{t-1})} \cdot \frac{1 - p(S^t|S^{t-1})}{p(S^t|S^{t-1})} \cdot \frac{p(S^t|o^1 \dots o^{t-1}, S^{t-1})}{1 - p(S^t|o^1 \dots o^{t-1}, S^{t-1})} \tag{10}$$

The term $p(S^t|S^{t-1})$ can be rewritten as $p(S)$. This is a time-invariant term, which represents the prior knowledge about the occupancy of any cell in the map. The term $p(S^t|o^1 \dots o^{t-1}, S^{t-1})$ is the occupancy of S^t given all observations up to the time step $t - 1$ and previous information about the map. It can be rewritten as $p(S^{t-1})$. Substituting the terms we obtain:

$$\frac{p(S^t|o^1 \dots o^t, S^{t-1})}{1 - p(S^t|o^1 \dots o^t, S^{t-1})} = \frac{p(S^t|o^t, S^{t-1})}{1 - p(S^t|o^t, S^{t-1})} \cdot \frac{1 - p(S)}{p(S)} \cdot \frac{p(S^{t-1})}{1 - p(S^{t-1})} \tag{11}$$

Equation (11) can be converted to a log-odds form, which can be more efficiently computed.

$$\log \frac{p(S^t|o^1 \dots o^t, S^{t-1})}{1 - p(S^t|o^1 \dots o^t, S^{t-1})} = \log \frac{p(S^t|o^t, S^{t-1})}{1 - p(S^t|o^t, S^{t-1})} + \log \frac{1 - p(S)}{p(S)} + \log \frac{p(S^{t-1})}{1 - p(S^{t-1})} \tag{12}$$

Equation (12) gives a recursive formula for updating the static map S. The $p(S)$ term is the prior for occupancy. If it is set to 0.5 (unbiased uncertainty), it can be canceled. The occupancy for the static map $p(S^t)$ is now calculated based on the previous information about this map $p(S^{t-1})$ and the inverse sensor model $p(S^t \mid o^t, S^{t-1})$.

Notice that the information about the previous occupancy is also part of the inverse sensor model. That information allows us to determine if some previously free space is now occupied, which means that some dynamic entity has moved to that place. It is also possible to detect if some entity that was previously considered static has moved. Table 1 shows the possible inputs to the inverse sensor model and the resulting values.

The first column in Table 1 represents the possible occupancy states of the cells in the previous static map S^{t-1}. The possible states are: *Free*, *Unknown*, and *Occupied*. To be considered *Free*, the occupancy probability of a grid cell must be below a pre-determined low threshold (we used 0.1 in our experiments). A very small occupancy probability means a high confidence that the cell is not occupied by a static entity in the environment. If the occupancy probability has a value above a high threshold (0.9 in our experiments) that cell is considered *Occupied*. If the occupancy probability is in the middle of the low and high thresholds, it is considered *Unknown*. The second column o^t represents the information provided by the sensors. In this case, each grid cell can be *Free* or *Occupied*, according to the sensor readings at the current robot position. The values of the resulting inverse observation model are represented, for simplicity, as: *high* value or *low* value. High values are values above 0.5 (that will increase the occupancy probability of that cell) and low values are values below 0.5 (that will decrease the occupancy probability of that cell). The third column shows the six possible combinations. For the first three rows $o^t = Free$. These

Table 1. Inverse observation model for the static map.

S^{t-1}	o^t	$p(S^t\|S^{t-1}, o^t)$
Free	Free	Low
Unknown	Free	Low
Occupied	Free	Low
Free	Occupied	Low
Unknown	Occupied	High
Occupied	Occupied	High

are the trivial cases where no obstacles are detected, and, independent of the information about previous occupancy, the inverse sensor model will result in a low value, which will decrease the occupancy probability. The fourth row ($S^{t-1} = \mathit{Free}$ and $o^t = \mathit{Occupied}$) is this case where there is strong evidence that the space was previously free of static entities and is now occupied. In this case the observation is considered consistent with the presence of a dynamic object and the static occupancy probability will decrease. In the fifth row ($S^{t-1} = \mathit{Unknown}$ and $o^t = \mathit{Occupied}$) there is uncertainty regarding the previous occupancy of that region in the map and the obstacles detected will be initially considered static until they are detected to have moved. Therefore, the sensor model will result in a high value, which will increase the occupancy probability of static entities in that region of the map. This situation occurs when the robot is initialized once all the grid cells reflect uncertainty about their occupation. The last row of the table is also trivial and shows the case where the space was previously occupied by a static obstacle and the sensors still confirm that belief. In this case the sensor model will result in a high value, which will raise the occupancy probability of a static obstacle on that region of the map.

3.2. *Dynamic Map Update*

By definition, the dynamic map D only contains information about the moving parts of the environment. We denote by $p(D^t)$ the occupancy probability of determined region of the map being occupied by an moving object at time t. Based on the sensor readings and information about previous occupancy in the static map (S^{t-1}), it is possible to identify the moving parts of the environment and represent then in the dynamic map D. It is important to mention that the information about the previous occupancy of the dynamic map (D^{t-1}) cannot be used as a reference because it may change over time.

Similar to Eqs. (3) and (4) can be rewritten in the following manner:

$$\log \frac{p(D^t|o^1 \dots o^t, S^{t-1})}{1-p(D^t|o^1 \dots o^t, S^{t-1})} = \log \frac{p(D^t|o^t, S^{t-1})}{1-p(D^t|o^t, S^{t-1})} + \log \frac{1-p(D)}{p(D)} + \log \frac{p(D^{t-1})}{1-p(D^{t-1})} \tag{13}$$

Equation (13) is similar to Eq. (12) in the sense that the new estimation for the occupancy of $p(D^t)$ is based on the previous occupancy of that map $p(D^{t-1})$ and the sensor model $p(D^t \mid o^t, S^{t-1})$. Usually, we make $p(D) = 0.5$, which means no a priori knowledge about the occupancy in D and the entire term can be canceled. In order to update the dynamic map D Eq. (13) also takes into account the information about previous occupancy of the static map S^{t-1} in its sensor model.

It is important to state that we are not interested in keeping all information about the occupancy of dynamic objects over time. The objective of the dynamic map is to maintain information about the dynamic objects only at the present time. For example, if a particular grid cell was occupied by an moving object in the past and it is currently free, it it will be considered just free. We do not keep any history about previous occupancy of the cells in D. The information in the map just needs to be set to represent the current occupancy of each cell. Of course, in order for the changes in the environment to be reflected on the map, those changes must be sensed by the robot. If those changes occur in the robot's field of view, they will be reflected immediately in the map, otherwise the robot needs to revisit the regions of the environment where the changes occurred, in order to detect them.

Table 2 shows the values of the inverse observation model used to update the dynamic map. The first and second columns are identical to the table used in the static map. However, for the dynamic map, the behavior of the inverse observation model is slightly different. In the three first rows, as the observation o^t indicates a free space, the occupancy probability will be trivially updated with a low value independent of the previous occupancy on the static map. In the fourth row, the previous occupancy in the static map states that the space was free $S^{t-1} = \mathit{Free}$ but the sensor readings show some obstacle in that cell $o^t = \mathit{Occupied}$. This case characterizes the presence of a moving object and consequently the dynamic map will be updated with a

Table 2. Inverse observation model for dynamic map.

S^{t-1}	o^t	$p(D^t \mid o^t, S^{t-1})$
Free	Free	Low
Unknown	Free	Low
Occupied	Free	Low
Free	Occupied	High
Unknown	Occupied	Low
Occupied	Occupied	Low

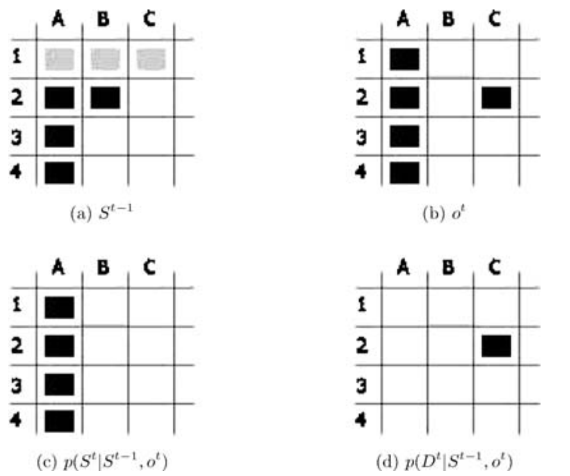


Figure 1. Updates for maps S and D.

high occupancy probability. In the fifth row we have the case where $S^{t-1} = \textit{Unknown}$ and $o^t = \textit{Occupied}$. As we do not have any information about the previous occupancy of that area, we cannot know what kind of obstacle is being detected by the sensors, by default it is considered static until some movement is detected. Therefore we keep a low occupancy update on the dynamic map. The sixth row where $S^{t-1} = \textit{Occupied}$ and $o^t = \textit{Occupied}$ is trivial, and the inverse sensor model results in a low value.

Figure 1 shows an example of map update. In Fig. 1(a), the black spaces in the grid represent occupied regions, the white spaces represent free regions , and the gray spaces represent unknown regions. These three possibilities are equivalent to column 1 in Tables 1 and 2 (S^{t-1}). In Fig. 1(b), similarly to column 2 of Tables 1 and 2, we have two possibilities for the observations, black spaces representing an *occupied* observation while white spaces signify a *free* observation (O^t). Figure 1(c), equivalent to column 3 of Table 1, shows the results for the inverse sensor models, which will be applied to update the cells on the static map S ($p(S^t \mid o^t, S^{t-1})$). Figure 1(d) represents the inverse sensor model, which will be applied to update the dynamic map D, column 3 of Table 2 ($p(D^t \mid o^t, S^{t-1})$).

This example illustrates two interesting cases of map update. In the first case, the cell B2 was occupied (time step $t-1$) but the sensor readings indicate a free space at that place (time step t). This means that a moving object that was probably stopped has been mapped as a static part of the environment. As the object moved, the static map has been correctly updated to a free space. In the second case, the cell C2 was free (time step $t-1$) but the sensor readings indicate that region as occupied (time step t). It means that a moving object moved to that space. The update applied to that cell on the static map (Fig. 1(c)) will represent it as free space because the moving object will not be represented in the static map S. it will be represented only on the dynamic map D as seen in Fig. 1(d).

4. Localization

In order to build consistent occupancy grid maps, good localization is required. For most commercial robots odometric information is not accurate enough for reasonable localization. With time, the odometer tends to accumulate error without bound. As the identification of the moving objects is based on previous maps of the same region, accuracy errors to determine the exact position of the robot can lead to mistakes such as considering static parts of the environment as moving objects.

We use a localization method based on landmarks—features in the environment that can be detected by the sensors of the robot. If the robot has some a priori information about the position of the landmarks, it is possible to estimate its position as it detects the landmarks. If there is no previous information about the position of the landmarks, both the position of the robot and the position of the landmarks have to be estimated simultaneously. As the approach presented in this paper assumes that the robot does not have any a priori information about the environment, the algorithm given in Dissanayake et al. (2001) has been used to simultaneously estimate the position of both the landmarks and the robot. The landmarks used in our experiments are corners, which are commonly present in indoor environments. Corners are detected (Tomasi and Kanade, 1991) using the measurements provided by a laser range finder and represented by the triple x, y, f, where x and y are position coordinates in the Cartesian space and f is a flag which indicates whether the corner is convex or concave (used for data association).

As most corners have basically the same shape, they cannot be uniquely identifiable. Therefore, the data association problem has to be solved in order to correctly assign the landmarks detected by the sensors to the landmarks present in the map. The nearest neighbor filter has been used to address the data association problem.

Besides the two occupancy grid maps, the robot keeps a third map, which contains the information about the position of the detected landmarks. But as

Table 3. Static and dynamic landmark classification.

Occupancy in S	Type of landmark
Free	Dynamic landmark
Unknown	Dynamic landmark
Occupied	Static landmark

Table 4. Mapping dynamic environment results.

	Real corridors	Real arena	Simulated corridors	Simulated arena
Total objects moved	20	20	20	20
Correct detected	19	19	20	20
Not detected	1	1	0	0
Detected static	3	0	0	0

we are dealing with dynamic environments, there is a possibility of moving objects being detected as landmarks. As moving objects change their position over time, they may be used as references for localization. Using them as references can lead to errors in localization and (eventually) and mapping. Therefore, it is clearly necessary to differentiate static landmarks, which are suitable for localization.

The strategy used to differentiate static landmarks from dynamic landmarks (moving objects that can be detected as landmarks) is based on the information provided by the static map S. As shown in Table 3, a landmark is considered static if, and only if the occupation probability of the corresponding region in the static map is classified as occupied. If a landmark is found in an empty area, it will be considered as a dynamic object that moved to that position, so it will not be used as a reference to localize the robot. If the occupation probability is unknown, the landmark is also considered dynamic and is not used for localization.

5. Experimental Results

In order to validate the ideas presented in this paper, extensive simulated and experimental tests have been performed. Experimental tests have been done using ActiveMedia Pioneer Robots equipped with SICK laser range finders and an Athlon XP 1600+ machine. Player[1] has been used to perform the low level control of the robots (Gerkey et al., 2001) and Player/Stage have been used in the simulated experiments. Two experiment sets are reported here, both in Stage simulator and real robots. Only mapping was performed in the first set of experiments, while SLAM was done in the second.

5.1. *First Set of Experiments: Mapping*

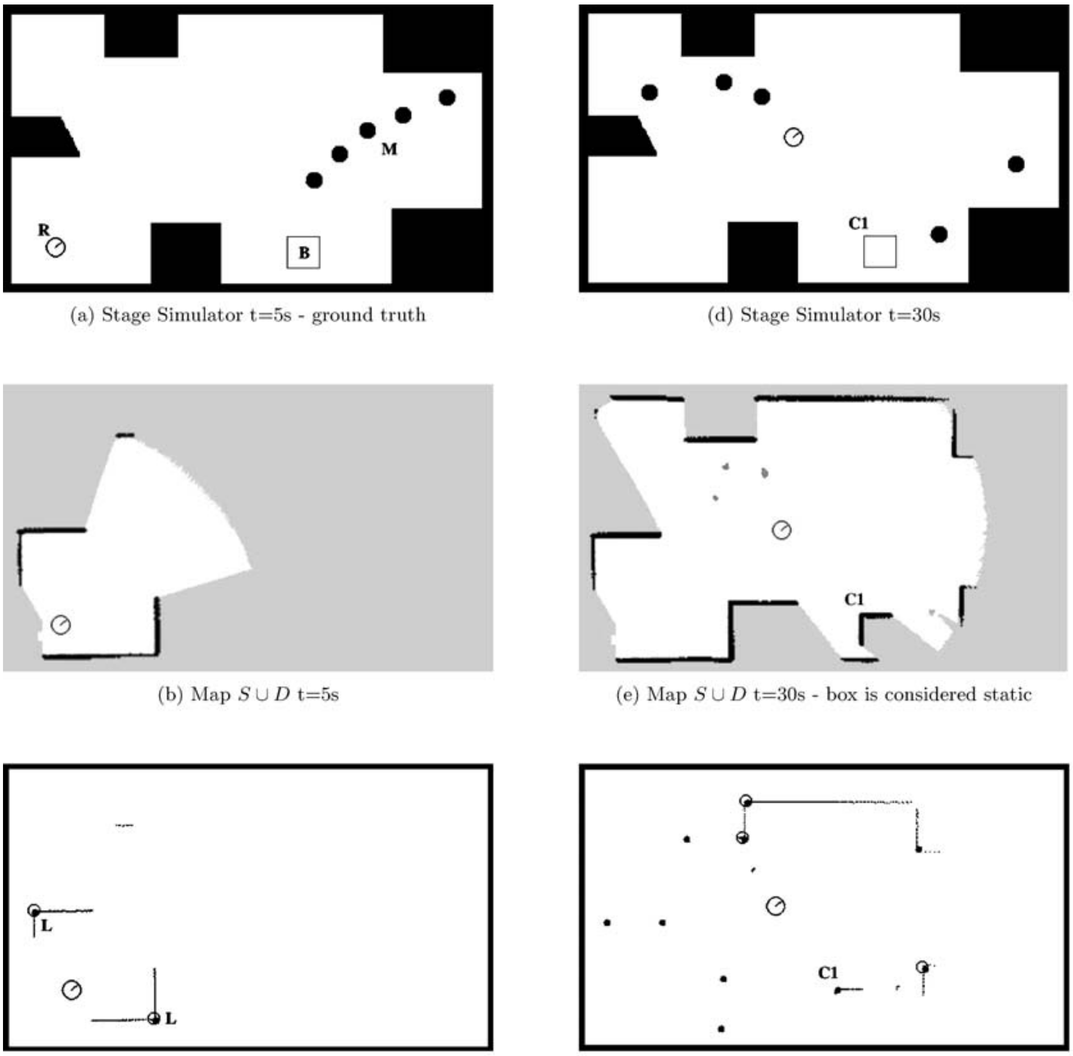
The purpose of the first set of experiments is to test the mapping part of the proposed approach in an environment with objects that are stationary most of the time, moving occasionally. It is also important to mention that the Kalman Filter localization has not been used during these experiments. In order to have localization, beacons (reflexive pieces of paper) have been put in the environment. The robot had some a priori information about the position of those beacons. The experiments have been performed on the corridors of the computer science department at USC and in Player/Stage simulator. The robot built maps of two different environments, corridors and an open arena. Cylinders of paper ranging from 10 to 60 cm diameter were used as moving objects. The basic difference between these two environments is that during the corridor experiments the objects were not moved in front of the robot (they were hidden by the walls). Only after revisiting the region where the objects were moved, the robot was able to notice the changes in the environment and update the map. In the open arena, the objects were moved in front of the robot.

The objects were moved 20 times in each experiment. As shown in Table 4, the algorithm was able to successfully detect the moving objects. Some of these objects were not been detected in the experimental tests because of pathological cases where small objects are positioned close to walls and other static obstacles in the environment. As the grid size used in this experiment was 10 cm, the rounding of some laser readings plus some small errors in localization erroneously considered the moving object as part of the wall. The same errors in localization lead the robot to classify static parts of the environment as dynamic as well.

5.2. *Second Set of Experiments: SLAM*

In the second set of experiments, objects that move actively have been included in the test set. These experiments has also been performed in simulation and using real robots. In the simulation experiments, a robot was required to localize itself and build a map of its environment (19 m $\times$ 11 m) while many other entities (between 2 to 10) were moving independently in the vicinity. Besides those moving objects, some other objects (square shaped boxes) were added to the environment. The moving entities used in these experiments were robots, which were the only autonomously controlled entities provided by the Stage simulator. These boxes were (manually) occasionally moved out of the field of the view of the robot. In order to have a more realistic simulation, considerable error was added to the odometric information (which is accurate in the simulator). The effect of error can be seen in the Fig. 3(a), which is a map created when the localization was only provided by the noisy odometric information.

Figure 2 shows the Stage simulator, the occupancy grid map (S and D combined), and the landmark map at four different time steps. Figures 2(a), (d), (g), and (j) are screenshots of the Stage simulator, which are used as ground truth. In Fig. 2(a), the white circle marked as R represents the robot, the moving entities are represented by black circles (marked as M), and the box the that occasionally moved is represented by the square marked as B.



(a) Stage Simulator t=5s - ground truth

(d) Stage Simulator t=30s

(b) Map $S \cup D$ t=5s

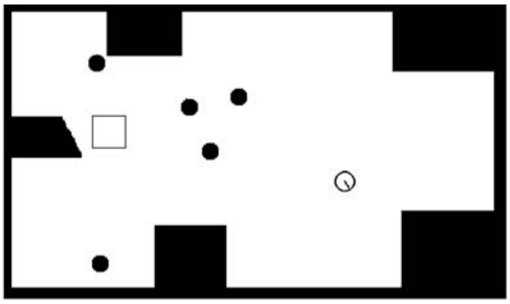
(e) Map $S \cup D$ t=30s - box is considered static

(c) Landmarks t=5s - corners in the environment

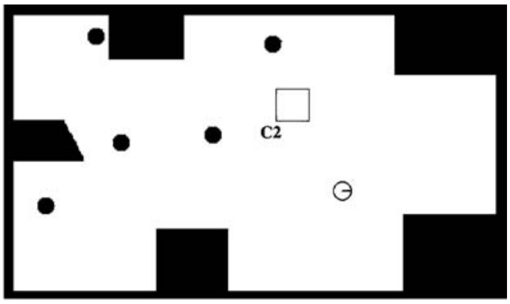
(f) Landmarks t=30s - moving objects are not considered landmarks

Figure 2. Simulation with 6 moving objects: the white circle R represents the robot, the moving entities M are detected as dynamic, and the box B is initially represented as static (e). After it has been detected to move, the box B is correctly represented as dynamic (k).

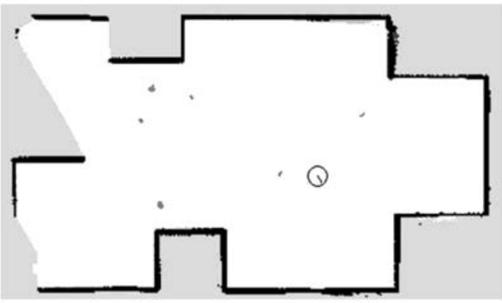
(*Continued on next page.*)



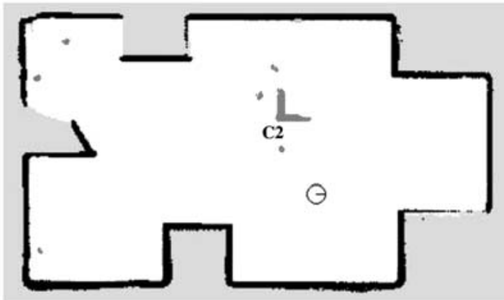
(g) Stage Simulator t=100s - box is moved



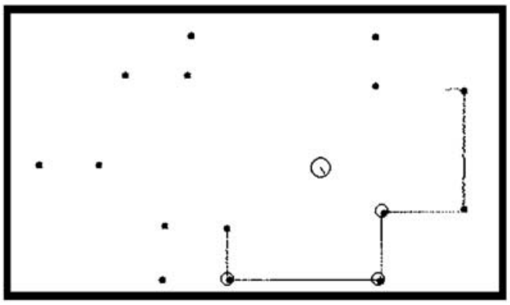
(j) Stage Simulator t=140s - robot detect box again



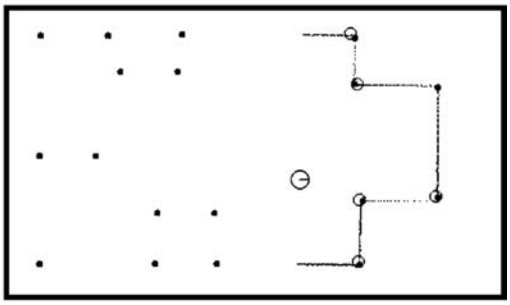
(h) Map $S \cup D$ t=100s - map is updated in the box region



(k) Map $S \cup D$ t=140s - box is considered dynamic

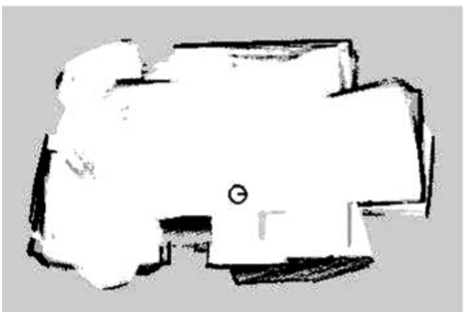


(i) Landmarks t=100s

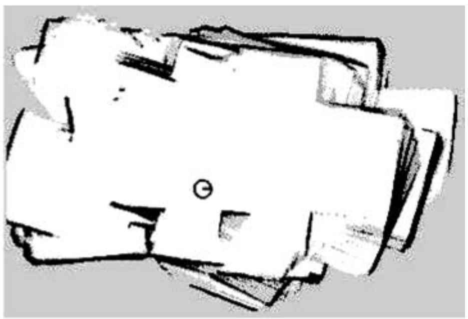


(l) Landmarks t=140s - box is not a landmark anymore

Figure 2. (*Continued*).

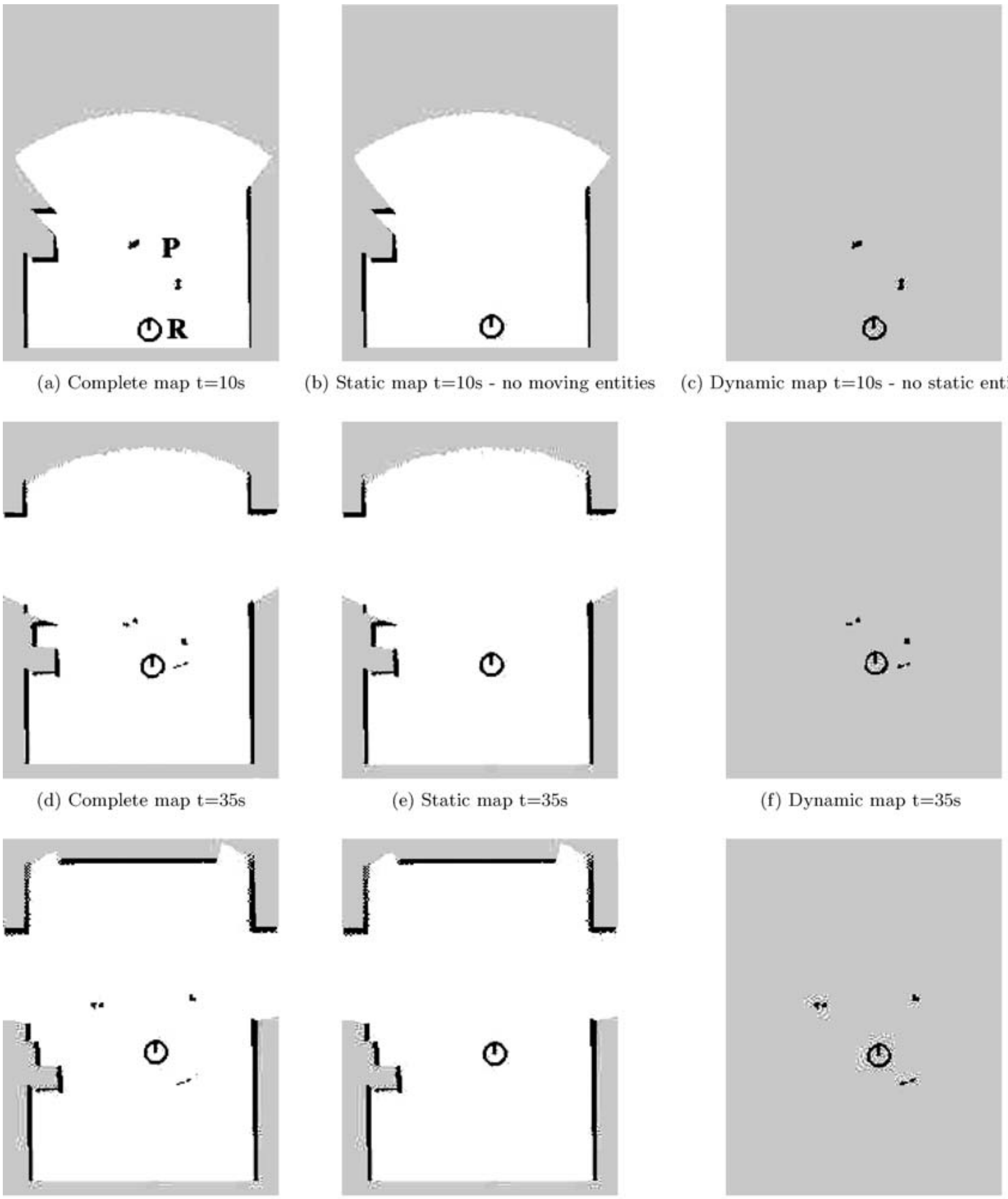


(a) No localization



(b) No dynamic obstacles detection

Figure 3. Poor quality maps due to lack of localization and dynamic obstacles detection.



(a) Complete map t=10s (b) Static map t=10s - no moving entities (c) Dynamic map t=10s - no static entities

(d) Complete map t=35s (e) Static map t=35s (f) Dynamic map t=35s

(g) Complete map t=70s (h) Static map t=70s - no errors due to dynamic entities (i) Dynamic map t=70s - moving entities tracked during mapping

Figure 4. Map with people moving around the robot. The robot (R) detected all the moving entities (P) and represented them appropriately in the map D.

It is important to notice that both robots and the box have corners which are identified as possible landmarks. Static parts of the map are represented in black while the dynamic entities are gray in color (Fig. 2(b), (e), (h), and (k)). The small black circles in Fig. 2(c), (f), (i), and (l) represent the static landmarks (corners of the walls). The corners generated by the moving object are not represented in the landmark map because they cannot provide localization. The lines following the walls in the landmark maps are the laser scans and the circles around determined landmarks identify the landmarks that are being detected at that time.

In Fig. 2(e), it is possible to see that the square box has been detected as static part of the environment. The robot had not seen the box move at that time, therefore it was considered static by default. The corner marked as C1 has also been detected as a landmark which could be used to provide localization. During the experiment, the box was moved (out of the view of the robot). As the robot revisited the part of the map, the region where was the box was updated as free space (Fig. 2(h)). After a while, the box was detected in a different position in the map and correctly classified as dynamic part of the environment (marked as C2 on Fig. 2(k)), even after the robot has never seen it moving. A short movie of the simulation is available at: http://robotics.usc.edu/denis/videos/slam_sim.avi.

As a result, the robot was able to differentiate the static and dynamic objects and show them appropriately on the map. All the moving objects in the environment were correctly detected and only the static landmarks have been used for localization.

In order to show how detection of dynamic entities and SLAM can be mutually beneficial, we performed the same experiment not using the EKF localization (Fig. 3(a)) and without differentiating dynamic entities in the environment (Fig. 3(b)). On both tests the robot failed to create reasonable maps of the environment. Without the EKF the robot accumulated considerable error in its localization estimate, which resulted in errors in detecting dynamic entities after revisiting an already mapped region of the environment. Without differentiating dynamic entities the robot considered moving objects as landmarks, which resulted in errors in localization and consequently in mapping.

In order to test the robustness of the algorithms in real situations, a set of experiments was conducted in the California Science Center (CSC) in Los Angeles. During the experiments in the CSC, a large open space (14 m $\times$ 10 m) was mapped while three people actively walked around the robot. The robot could correctly identify the static landmarks (corners) and successfully create maps of the environment differentiating static and dynamic entities. Figure 4 shows the static and dynamic maps, and their union. The robot his marked as R in Fig. 4(a). The three rows in Fig. 4 represent three distinct time steps. Figures 4(b), (e), and (h) show the occupancy grid of the static map (S). Figures 4(c), (f), and (i) show the occupancy grid of the dynamic map (D). The small black regions on the dynamic maps represent the position of the moving objects (people) at that point in time (marked as P in the Fig. 4(a)). Figures 4(a), (d), and (g) show the complete map of the environment, where both static and dynamic entities are represented. The results presented in Fig. 4 show the applicability of our approach in real world situations. The robot was able to robustly create a map of the environment where static and dynamics entities are correctly identified and appropriately represented. As ground truth was not available during the experiments, the error in the robot's pose could not be measured precisely. We estimate that the uncertainty in the robot's position indicated by the EKF localization algorithm is on the order of few centimeters.

As identification of moving objects is based on the comparison of the sensor measurements and the information contained in the map of the environment, small localization/rounding errors lead to mistakes in differentiating static and dynamic parts of the environment. These mistakes are avoided if instead of comparing only the occupancy of a determined cell we take into account the occupancy of the neighborhood of that cell.

6. Conclusions and Future Work

We have proposed an approach to SLAM in dynamic environments which uses features that are likely to be static. We also demonstrated that detection of dynamic entities and SLAM are mutually beneficial. Experimental and simulated tests show that our approach is able to successfully create maps of dynamic environments, correctly differentiating static and dynamic parts of the environment and represent them in an occupancy grid. As the localization is based on corner detection, the algorithm is also able to differentiate landmarks provided by the static and dynamic entities, and use the static landmarks to do localization. The algorithm is robust enough to detect dynamic entities both when they move in and out robot's field of view.

As future work, we will investigate the use of different localization algorithms. They will be implemented in order to deal with non-structured environments. Alternative algorithms to detect moving entities may also be incorporated to our approach in order to improve the efficiency of the dynamic objects detection. We also plan to address the case where dynamic objects move very slowly and close to static parts of the environment, which created problems in the neighborhood comparison used in the present experiments. We also plan to investigate how to use these techniques in outdoor environments, which introduce different challenges for mapping, localization, and detection of dynamic entities.

7. Resources

The data-sets used in this article are available on the Radish (Robotics Data Sets Repository) web-site (Howard and Roy, 2003).

Acknowledgments

The authors thank Boyoon Jung and Julia Vaughan for their valuable support during the tests in the California Science Center. This work is supported in part by the DARPA MARS program under grants DABT63-99-1-0015 and 5-39509-A (via UPenn), ONR DURIP grant N00014-00-1-0638 and by the grant 1072/01-3 from CAPES-BRAZIL.

Note

1. Player is a server and protocol that connects robots, sensors and control programs across the network. Stage simulates a population of Player devices, allowing off-line development of control algorithms. Player and Stage were developed jointly at the USC Robotics Research Labs and HRL Labs and are freely available under the GNU Public License from http://playerstage.sourceforge.net.

References

Andrade-Cetto, J. and Sanfeliu, A. 2002. Concurrent map building and localization on indoor dynamic environments. *International Journal on Pattern Recognition and Artificial Intelligence*, 16:361–374.

Avots, D., Lim, E., Thibaux, R., and Thrun, S. 2002. A probabilistic technique for simultaneous localization and door state estimation with mobile robots in dynamic environments. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 521–526.

Aycard, O., Laroche, P., and Charpillet, F. 1998. Mobile robot localization in dynamic environments using places recognition. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3135–3139.

Biswas, R., Limketkai, B., Sanner S., and Thrun, S. 2002. Towards object mapping in non-stationary environments with mobile robots. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pp. 1014–1019.

Castellanos, J., Neira, J., and Tards J. 2001. Multisensor fusion for simultaneous localization and map building. *IEEE Trans on Robotics and Automation*, 17(6):908–914.

Dissanayake, M.W.M.G., Newman, P., Durrant-Whyte, H.F., Clark, S., and Csorba, M. 2001. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation*, 17(3):229–241.

Elfes, A. 1986. Sonar-based real-world mapping and navigation. *IEEE Transactions on Robotics and Automation*, 3(3):249–265.

Elfes, A. 1989. Using occupancy grids for mobile robot perception and navigation. *Computer*, 12(6):46–57.

Fenwick, J., Newman, P., and Leonard, J. 2002. Collaborative concurrent mapping and localization. In *Proc. of IEEE Conf on Robotics and Automation*, pp. 1810–1817.

Fox, D., Burgard, W., and Thrun, S. 1999. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 2:391–327.

Gerkey, B.P., Vaughan, R.T., Stoy, K., Howard, A., Sukhatme, G.S., and Mataric, M.J. 2001. Most valuable player: A robot device server for distributed control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1226–1231.

Guivant, J.E. and Nebot, E. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257.

Guivant, J. and Nebot, E. 2002. Improving computational and memory requirements of simultaneous localization and map building algorithms. *IEEE International Conf on Robotics and Automation ICRA*, pp. 2731–2736.

Haehnel, D., Schulz, D., and Burgard, W. 2002. Map building with mobile robots in populated environments. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 496–501.

Haehnel, D., Triebel, R., Burgard, W., and Thrun, S. 2003. Map building with mobile robots in dynamic environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1557–1563.

Haehnel, D., Fox, D., Burgard, W., and Thrun, S. 2003. A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. of the Conference on Intelligent Robots and Systems (IROS)*, pp. 206–211.

Howard, A. and Roy, N. 2003. Radish: The robotics data set repository http://radish.sourceforge.net.

Liu, Y. and Thrun, S. 2003. Results for outdoor-SLAM using sparse extended information filters. In *Proc. of IEEE International Conference on Robotics and Automation*, pp. 1227–1233,

Lu, F. and Milios, E. 1994. Globally consistent range scan alignment for environment mapping. *Autonomous Robots* 4:333–349.

Moravec, H. 1988. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74.

Montemerlo, M., Thrun, S., Koller D., and Wegbreit, B. 2002. Fast-SLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pp. 593–598.

Newman, P.M., Leonard, J.J., Neira, J., and Tardos, J. 2002. Explore and return: Experimental validation of real time concurrent mapping and localization. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 1802–1809.

Nieto, J., Guivant, J., Nebot, E., and Thrun, S. 2003. Real time data association for FastSLAM. In *Proc. of IEEE International Conference on Robotics and Automation*, pp. 512–518.

Tardos, J.D., Neira, J., Newman, P.M., and Leonard, J.J. 2002. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 21(4):311–330.

Thrun, S. 2002. Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium*, G. Lakemeyer and B. Neberl (Eds.). Morgan Kaufmann.

Thrun, S., Burgard, W., and Fox, D. 1998. A probabilistic approach to concurrent mapping and localization for mobile robots'. *Machine Learning*, 31:29–53.

Thrun, S., Burgard, W., and Fox, D. 2000. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 321–328.

Tomasi, C. and Kanade, T. 1991. Detection and tracking of point features, *Carnegie Mellon University Technical Report* CMU-CS-91-132.

Wang, C. and Thorpe, C. 2002. Simultaneous localization and mapping with detection and tracking of moving objects. *IEEE International Conference on Robotics and Automation*, pp. 2918–2924.

Wang, C.-C., Thorpe, C., and Thrun, S. 2003. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 842–849.

Wolf, D.F. and Sukhatme, G.S. 2003. Towards mapping dynamic environments. In *In Proceedings of the International Conference on Advanced Robotics (ICAR)*, pp. 594–600.

Wolf, D. and Sukhatme, G. 2004. Online simultaneous localization and mapping in dynamic environments. In *Proc. of IEEE International Conference on Robotics and Automation*, New Orleans, pp. 1301–1306.

Dennis F. Wolf is a Ph.D. student in the Computer Science department at University of Southern California (USC). His research interests include localization and mapping in urban environments and mapping dynamic environments with mobile robots. Denis received his B.Sc. in Computer Science from Federal University of Sao Carlos, Brazil in 1999 and his M.Sc. in Computer Science from University of Sao Paulo, Brazil in 2001.

Gaurav S. Sukhatme is an Assistant Professor in the Computer Science Department at the University of Southern California (USC). He received his M.S. and Ph.D. in Computer Science from USC. He is the co-director of the USC Robotics Research Laboratory and the director of the Robotic Embedded Systems Laboratory which he founded at USC in 2000. His research interests are in distributed mobile robotics and sensor/actuator networks. Prof. Sukhatme has served as PI on several NSF, DARPA and NASA grants. He is a member of AAAI, IEEE and the ACM and has served on several conference program committees. He is on the editorial boards of two leading journals in robotics, IEEE Transactions on Robotics and Automation, and Autonomous Robots and a leading magazine in ubiquitous computing, IEEE Pervasive Computing. He has published over 100 technical papers and is a receipient of the NSF CAREER award.