Online Simultaneous Localization and Mapping in Dynamic Environments

Denis 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, California, USA
{denis,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 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 with physical robots show the efficiency of our approach and and show how the differentiation of dynamic and static entities in the environment and SLAM can be mutually beneficial.

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in mobile robotics and has been studied extensively in the robotics literature recently. For the most part, research has concentrated on SLAM in static environments. In this paper we explicitly consider the SLAM problem in a dynamic environment. An algorithm which differentiates between the dynamic and static parts of the environment can considerably contribute to the efficiency of both localization and mapping. Two successful techniques used to perform SLAM are based on alignment of sensor readings and detection of landmarks. Both techniques can fail in the presence of dynamic entities. The explicit identification of those dynamic entities can improve SLAM efficiency.

The approach presented in this paper can be divided into two parts: mapping 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 do simultaneous localization and mapping.

The mapping algorithm extends the occupancy grid technique introduced in [1] to dynamic environments. The resulting algorithm is capable of detecting dynamic objects in the environment and representing them in the map. The non-stationary objects are detected even when they move out of the robot's field of view. 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 present in the two maps $(S \cup D)$.

The localization algorithm is based on the well-known SLAM approach [2]. We use a Kalman Filter to incrementally estimate the correct position of the robot and landmarks (e.g. corners).

Experimental tests have been performed using ActiveMedia Pioneer Robots 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.

II. RELATED WORK

Mapping of static environments has received considerable attention recently (see [3] for a survey), but in most cases, these algorithms cannot be directly applied to dynamic environments. Usually, the presence of moving objects lead these approaches to make mistakes and compromising the overall quality of the maps. This is a considerable problem since many realistic applications for robots are in non-static environments. Mapping dynamic environments has been addressed in recent years [4,5,6,7], 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 about related approaches to this problem, it is interesting to clarify the notion of moving objects in a dynamic environment. There are two different 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 approach presented in this paper deals with both categories of moving objects.

In [4], Biswas et. al. present an off-line Bayesian approach (based on the EM 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 presented by Haenel et. al. [5] uses the EM algorithm to differentiate (off-line) dynamic and static parts of the environment. 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.

In [6], Wang et. al. present a framework for solving the simultaneous mapping, localization, detection and tracking of moving objects. The idea is identify and keep tracking of moving objects in order to improve the quality of the map. This approach, however, can only identify moving objects when they move into the robot's field of view.

In prior work [7], 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, the approach in [7] assumed ideal localization, a fairly narrow assumption.

III. THE MAPPING APPROACH

In our approach, two distinct occupancy grid maps (S and D) are used. The static map S, only contains 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 approach 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.

The probabilistic occupancy of each grid cell can usually be estimated based on the position of the robot and its sensor measurements. Let $p(S_{x,y}^t)$ and $p(D_{x,y}^t)$ denote the occupancy probabilities of the grid cell with coordinates < x, y > in the static and dynamic maps S and D, respectively. The set of sensor readings is represented by o and the position of the robot 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 o. Therefore, the problem is to estimate the following:

$$p(S_{x,y}^t, D_{x,y}^t | o^1 \cdots o^t, u1 \cdots u^t) \tag{1}$$

The position of the robot u is only used to calculate which region of the map (grid cells) will be updated. It is

not used to calculate the value that will be used to update the occupancy probability of those grid cells. Therefore, for simplicity, the information regarding the position of the robot (u) and coordinate of the grid cells (< x, y >) will not be included in the following equations.

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 add some previous information about the static parts of the environment to Equation 1. This information allows us to separate the sensor readings provided by static and dynamic obstacles, and correctly update both maps. The quantity to be estimated is:

$$p(S^t, D^t | o^1 \cdots o^t, S^{t-1})$$
 (2)

As the occupancy information of S and D maps is mutually exclusive (an entity cannot be part of the static and dynamic maps at the same time), it is possible to rewrite Equation 2 as a set of two equations.

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

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

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

A. Static Map Update

The update equation for the static map S (Equation 3) is slightly different from the regular occupancy grid technique, which assumes the environment does not change over time. We use the previous knowledge about the environment and compare with the current set of observations in order to keep only the static parts of the environment in the map S.

As shown in [7], the quantity in Equation 3 can be rewritten as follows:

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

Equation 5 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 cancelled. 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 $p(S^t|o^t,S^{t-1})$.

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 I shows the possible inputs to the inverse sensor model and the resulting values. The first column represents the possible occupancy states of the cells in the previous static map S^{t-1} . The

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

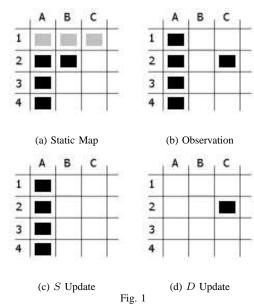
TABLE I
INVERSE OBSERVATION MODEL FOR THE STATIC MAP

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).

Table I shows the six possible combinations. For the first three rows $o^t = Free$. These 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 $(S^{t-1} = Free \text{ and } o^t = Occupied)$ 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 $(S^{t-1} = Unknown)$ and $o^t = Occupied$) 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.

B. Dynamic Map Update

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



UPDATE FOR THE STATIC AND DYNAMIC MAPS.

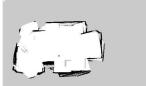
occupied by an moving object at time t. Based on the sensor readings and information about previous occupancy in the static map S, it is possible to identify the moving parts of the environment and represent then in the dynamic map D.

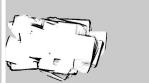
Similar to Equation 3, Equation 4 can be rewritten in the following manner [7]:

$$\frac{p(D^t|o^1 \cdots o^t, S^{t-1})}{1 - p(D^t|o^1 \cdots o^t, S^{t-1})} = \frac{p(D^t|o^t, S^{t-1})}{1 - p(D^t|o^t, S^{t-1})} \cdot \frac{1 - p(D)}{p(D)} \cdot \frac{p(D^{t-1})}{1 - p(D^{t-1})}$$
(6)

Equation 6 is similar to Equation 5 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 $p(D^t|o^t,S^{t-1})$. In order to update the dynamic map D, Equation 6 also takes into account the information about previous occupancy of the static map S 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 has already been occupied by an object in the past and it is currently free, we do not keep any history about previous occupancy 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 [7].





(a) No localization

(b) No DO detection

Fig. 2
Maps with errors.

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

TABLE II
INVERSE OBSERVATION MODEL FOR DYNAMIC MAP

Table II 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 $S^{t-1} = Free$ but the sensor readings show some obstacle in that cell $o^t = Occupied$. This case characterizes the presence of a moving object and consequently the dynamic map will be updated with a high occupancy probability. In the fifth row we have the case where $S^{t-1} = Unknown$ and $o^t = Occupied$. 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 $S^{t-1} = Occupied$ and $o^t = Occupied$ is trivial, and the inverse sensor model results in a low value.

Figure 1 shows an example of map update. In the Figure 1a, 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 I and II (S^{t-1}) . In Figure 1b, similar to column 2 of Tables I and II, we have two possibilities for the observations, black spaces representing an 'occupied' observation while white spaces signify a 'free' observation (O^t) . Figure 1c, equivalent to column 3 of Table I, shows the results for the inverse sensor models, which will be applied to update the cells on the static map S $(p(S^t|o^t, S^{t-1}))$. Figure 1d represents the inverse sensor model, which will be

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

TABLE III
STATIC AND DYNAMIC LANDMARK CLASSIFICATION.

applied to update the dynamic map D, column 3 of Table II $(p(D^t|o^t,S^{t-1}))$.

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. 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 (Figure 1c) will represent it as free space because the moving object will not be represented in the static map, it will be represented only on the dynamic map as seen in Figure 1d.

IV. LOCALIZATION

In order to build consistent occupancy grid maps, good localization is required. For most commercial robots the odometric information is not accurate enough to have reasonable localization. After some time, the odometer tends to accumulate error without bounds. 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 a mistakes such as considering static parts of the environment as moving objects.

The localization method used in this approach is 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 [2] 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 [8] using the measurements provided by a laser range finder.

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. In addition, the corner detection algorithm has been modified to differentiate convex

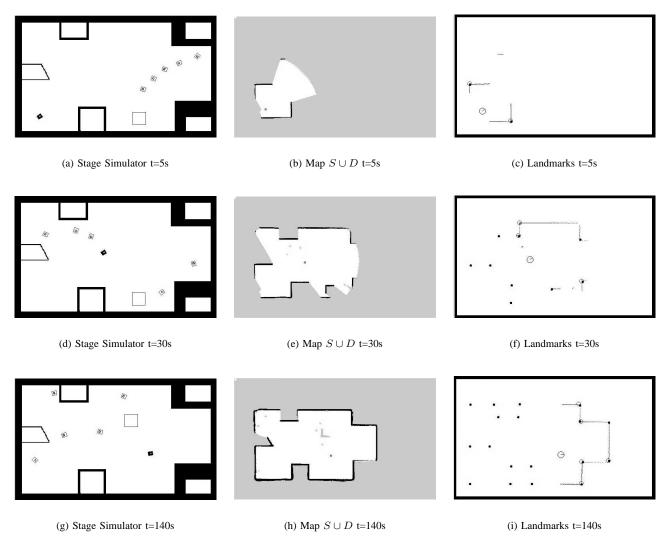


Fig. 3
SIMULATION WITH 5 MOVING OBJECTS.

and concave corners. That information is also used to help in the data association.

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 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.

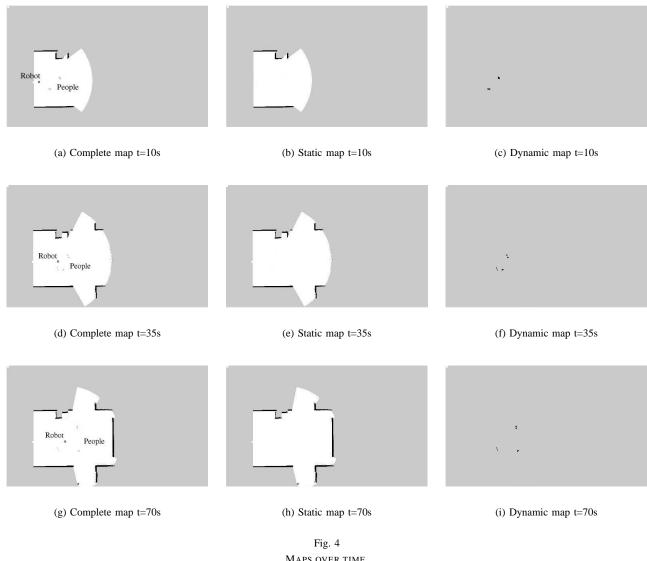
The strategy used to differentiate static landmarks from dynamic landmarks (moving objects that can be detected as landmarks) is based on the information provided by the static map S. Let $o_{x,y}$ denote the observation of a landmark at the

coordinates $\langle x, y \rangle$.

As shown in Table III, a landmark is only considered static if the occupation probability of that region in the static map is classified as occupied. If some 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 considered for localization.

V. 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



MAPS OVER TIME.

XP 1600+ machine. Player¹ has been used to perform the low level control of the robots [9] and Player/Stage have been used in the simulated experiments.

In the simulation experiments, a robot was required to localize itself and make a map of its environment while many other robots (between 2 to 10) were wandering around (these were the moving objects). Besides these moving objects, some other objects were added to the environment. These objects were only moved (manually) out of the field of the view of the robot. Considerable error was added to the odometric information (which is accurate in the simulator) in order to obtain more realistic results.

Figure 3 shows the occupancy grid map (S and D combined), the landmark map and the simulator at three different points in time. In this experiment, the black robot had to perform SLAM while five other robots were randomly moving in the environment and a rectangular box was moved out of the view of the robot. 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 (Figures 3b, 3e, and 3h). The small black circles in Figures 3c, 3f, and 3i 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 point in

¹Player 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.

time.

The results obtained show that the robot is able to differentiate the static and dynamic objects and show them appropriately on the map. In order to show how detection of dynamic entities and SLAM can be mutually beneficial, we performed the same experiment without localization (Figure 2a) and without detecting the dynamic entities (Figure 2b). It is clear that the results are much worse. Figure 2 shows the noise which has been added to the odometric information on the simulator.

In order to test the robustness of the algorithms in real situations, a set of experiments were performed in the hallways of the Computer Science Department at USC and in the California Science Center in Los Angeles. In the experiments in the hallways, the robot built a map of the environment. After that some objects were moved out of the field of view of the robot. When the robot revisited the spaces where the changes occurred, it was able to correctly identify the parts of the environment that changed their position.

During the experiments in the California Science Center, a large open space was mapped while three people actively walked around the robot. The robot could correctly identify the landmarks (corners) and successfully create maps of the environment differentiating static and dynamic entities. Figure 4 shows the static and dynamic maps, and the union of both as well. The three rows in Figure 4 represent three distinct points in time. Figures 4b, 4e, and 4h show the occupancy grid of the static map (S). Figures 4c, 4f, and 4i 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. Figure 4a, 4d, and 4g show the complete map of the environment, where both static and dynamic entities are represented. The results presented in Figure 4 show the efficiency 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.

VI. 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 the differentiating the static and dynamic parts of the environment and doing SLAM simultaneously is 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 then in an occupancy grid map. 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 robot's field of view and also when they move out of the field of view of the robot.

As future work, we are investigating the use of different localization algorithms will be implemented in order to deal with non-structured environments. The use of other algorithms to detect moving entities may also be incorporated to our approach in order to improve the efficiency of the dynamic objects detection. As the 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 can be avoided if instead of comparing only the occupancy of a determined cell we take into account the occupancy of the neighborhood of that cell. We also plan to address the case where dynamic objects move very slowly, which creates problems in the neighborhood comparison used in the present experiments.

VII. ACKNOWLEDGEMENTS

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

REFERENCES

- Elfes, A "Sonar-based Real-World Mapping and Navigation," IEEE Transactions on Robotics and Automation, 3(3):249-265, 1986.
- [2] Dissanayake, M. W. M. G., Newman, P., Durrant-Whyte, H. F., Clark, S. and Csorba, M. "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotic and Automation* 17(3), 229–241, 2001.
- [3] Thrun, S., "Robotic Mapping: A Survey," In G. Lakemeyer and B. Neberl, editors, Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann, 2002.
- [4] Wang, C.-C. and Thorpe, C. and Thrun, S. "Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas," In Proceedings of the International Conference on Robotics and Automation (ICRA), 842-849, 2003.
- [5] Hähnel, D., Triebel, R., Burgard, W., Thrun, S. "Map Building with Mobile Robots in Dynamic Environments," In Proceedings of the International Conference on Robotics and Automation (ICRA), 1557-1563, 2003
- [6] Biswas, R., Limketkai, B., Sanner S., Thrun, S. "Towards Object Mapping in Non-Stationary Environments with Mobile Robots," In Proceedings of the International Conference on Intelligent Robots and Systems (IROS), 1014-1019, 2002.
- [7] Wolf, D., Sukhatme, S. "Towards Mapping Dynamic Environments," In Proceedings of the International Conference on Advanced Robotics (ICAR), 594-600, 2003.
- [8] Tomasi, C., Kanade, T. "Detection and Tracking of Point Features," Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991.
- [9] Gerkey, B. P., Vaughan, R. T., Stoy, K., Howard, A., Sukhatme, G. S., Mataric, M. J. "Most Valuable Player: A Robot Device Server for Distributed Control," In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 1226-1231, 2001.