19th IEEE International Conference on Tools with Artificial Intelligence

Maximum Likelihood SLAM in Dynamic Environments

Nikos Mitsou and Costas Tzafestas
School of Electrical and Computer Engineering
Division of Signal, Control and Robotics
National Technical Univ. of Athens
Zografou Campus, Athens 15773, Greece
Email: nmitsou@mail.ntua.gr, ktzaf@softlab.ntua.gr

Abstract

Simultaneous Localization and Mapping in dynamic environments is an open issue in the field of robotics. Traditionally, the related approaches assume that the environment remains static during the robot's exploration phase. In this work, we overcome this assumption and propose an algorithm that exploits the dynamic nature of the environment during robot exploration so as to improve the localization process. We use a Histogram Grid to store all the past occupancy values of every cell and thus to select the most probable pose of the robot based on the occupancy evolution. Experiments on a simulated robot indicate the effectiveness of the proposed approach.

1. Introduction

During the last two decades much research has been conducted in the field of robotics on the problem of Simultaneous Localization and Mapping (SLAM). Kalman Filtering, Expectation Maximization, Maximum Likelihood and Particle Filtering are some of the most common techniques used to solve the SLAM problem ([2], [6], [11], [15]).

However, most of these approaches assume that the environment is static during the robot exploration. Obviously, this assumption is wrong since there exist objects such as doors, chairs or people that change their position during the robot exploration phase. Lately, several approaches have been presented that deal with this problem by first detecting these dynamic areas and then filtering them out ([5], [4]). In this way, however, important features of the environment are ignored, reducing the quality of the localization process. These approaches use a single map to model the environment. However, when the environment changes, a single map cannot capture the dynamics of the environment.

In this paper, we propose a new SLAM algorithm for non-stationary environments, which extends the widely used Maximum Likelihood SLAM (*ML-SLAM*) algorithm ([15]). Our algorithm makes use of a new storing structure, the Histogram Occupancy Grid. Every cell of the grid preserves a histogram where the occupancy values of the cell at every time point are stored. When a new sensor measurement is available, the algorithm calculates the probability of the measurement based on the values that are stored in the histograms (and not by comparing with the previous most likely map as the common ML-SLAM does). So, even if a door is correctly identified as closed at time point $t-1$ and is opened at time point t, the algorithm will successfully match the door with a previous occurrence stored in the cells histograms.

The rest of the paper is organized as follows: In Section 2, the related work is presented. In Section 3, we provide an introduction to the SLAM problem and a description of the popular Maximum Likelihood SLAM algorithm. In Section 4, we present our method for SLAM over dynamic environments and the Histogram Occupancy Grid. Experiments are presented in Section 5. Conclusions and future work are presented in Section 6.

2 Related Work

In the last few years, many attempts have been made towards mapping of and localization in dynamic environments. Some attempts are based on the identification of moving objects. For example, in [8] a sonar range sensor, a camera and differentiating techniques were used, in [7] an expectation maximization algorithm was applied and in [12] a probabilistic filtering algorithm was implemented all to detect moving objects in the robot environment. The identified objects are then removed from the localization and mapping process.

On the other hand, some works aim at the identification of different environmental configurations. For example, the authors of [13] presented a fuzzy clustering technique to capture the typical configurations of the dynamic environment. Snapshots were collected in different points in time and were clustered into possible configurations. The localization algorithm was not only responsible for estimating the robot pose but also for determining the active map of the environment. In [1], an offline approach based on the EM algorithm was implemented. Different snapshots of the environment were used to identify objects with the same shape placed in different positions. In both works, the en-

1082-3409/07 $25.00 © 2007 IEEE
DOI 10.1109/ICTAI.2007.168

vironment was assumed to be static during the collection of the snapshots. The main difference of the presented algorithm with these works is that they do not examine the evolution of the cells occupancy while our approach preserves the history of the past estimations.

In the next section, a short introduction to the SLAM problem is presented.

3 The SLAM problem in static environments

In this section, we describe the basic concepts of the SLAM problem, which are used throughout the paper. A detailed description of the problem can be found in [14].

The SLAM problem is defined as the problem of estimating the pose (x, y location and ϑ orientation) of the mobile robot and constructing, at the same time, a map of the unknown environment that the robot is exploring. Let s_t be the robot position and m_t be the map of the environment at timepoint t. The map is modeled either as a $n \times n$ grid where each cell contains a probability of being occupied ([3]) or as a set of poses and sensor scans ([9]). In this work, we adopt the first representation.

SLAM aims at finding the robot pose and the map of the environment that maximizes the probability:

$$argmax_{m_t,s_t} p(m_t, s_t | z_{1:t}, u_{1:t}) \quad (1)$$

where $[1:t]$ is the period of observation, m_t is the estimated map of the environment at time point t, s_t is the estimated pose of the robot at t, $z_{1:t}$ are the sensor observations during the period $[1:t]$ and $u_{1:t}$ are the control motion commands followed by the robot during the period $[1:t]$.

Maximizing Equation 1 is a very expensive task, since searching should be performed in the space of all maps and poses. The existing solutions can successfully solve this problem but under specific assumptions. The most common assumption made is that the environment is static. Under this assumption, the Equation 1 is simplified as follows:

$$argmax_{m,s_t} p(m, s_t | z_{1:t}, u_{1:t}) \quad (2)$$

As already stated, we adopt the Occupancy Grid representation to model the environment. Occupancy Grid divides the environment into cells and attempts to determine the probability of the cells occupancy. A common way to calculate the occupancy probability of a cell i is to use the log-likelihood sensor model for the i cell at time t:

$$R_{i,t} = \log \frac{p(z_t | m_i == occ)}{1 - p(z_t | m_i == emp)} \quad (3)$$

where $p(z_t | m_i == state)$ denotes the probability that we get the observation z_t given that the state of the cell equals to $state$. The overall occupancy of every cell can be calculated by the following formula:

$$R_i = \sum_t R_{i,t} \quad (4)$$

In the next section, we shortly describe one of the most popular algorithms for solving the SLAM problem in static environments, the Maximum Likelihood SLAM (ML-SLAM) algorithm.

3.1 Maximum Likelihood SLAM (ML-SLAM)

The main idea of the ML-SLAM algorithm is to incrementally build a single map of the environment as the sensor data arrive, without keeping track of any residual uncertainty at the robot pose. Given a sensor measurement z_{t-1}, a motion command u_{t-1}, the previous most likely pose s^*_{t-1} and map m^*_{t-1}, the most likely pose s^*_t and map m^*_t are estimated. These estimates, once made, are frozen and can never be changed in the future.

In more detail, a series of maximum likelihood maps $m^*_1, m^*_2, \ldots$, along with a series of maximum likelihood poses $s^*_1, s^*_2, \ldots$ is maintained. The t^{th} map and pose are constructed from the $(t-1)^{th}$ map and pose via maximization of the marginal likelihood:

$\langle m^*_t, s^*_t \rangle = argmax_{m_t,s_t} \mathbf{p}(s_t, m_t | z_t, u_t, m^*_{t-1}, s^*_{t-1}) =$
$argmax_{m_t,s_t} \mathbf{p}(z_t | s_t, m_t) \mathbf{p}(s_t, m_t | u_t, s^*_{t-1}, m^*_{t-1}) =$
$argmax_{m_t,s_t} \mathbf{p}(z_t | s_t, m_t) \mathbf{p}(s_t | u_t, s^*_{t-1}) \mathbf{p}(m_t | m^*_{t-1}) \quad (5)$

In the above equation, the first probability $\mathbf{p}(z_t | s_t, m_t)$ corresponds to the Perception Model of the robot (how probable a sensor measurement z_t is given the robot pose s_t and the map m_t) while the second one $\mathbf{p}(s_t | u_t, s^*_{t-1})$ represents the Motion Model of the robot (how probable a pose s_t is given the previous pose s_{t-1} and the motion command u_{t-1}). The last probability $\mathbf{p}(m_t | m^*_{t-1})$ represents the Map Model and indicates how probable a new map is with respect to the previous most likely map. Although the solution to the above equation is an optimization of both robot pose and environmental map, in practice it usually suffices to search in the space of all poses s_t as the map m_t can be uniquely determined once the pose s_t is known.

4 Maximum Likelihood SLAM in Dynamic Environments

The traditional ML-SLAM assumes that the environment is static (c.f. Equation 5). In this section, we will describe how we can apply the ML-SLAM on dynamics environments.

To deal with a dynamic environment, we rely on the observation that this environment cannot be captured in a single static map. Actually, a set of maps should be maintained. One such map m_t should represent the probability of occupancy at a specific time interval $[t, t + \Delta t)$ where Δt denotes the frequency of map construction. The m_t map acts as an Occupancy Grid for the interval $[t, t + \Delta t)$ and new sensor measurements are added with the update rule of the Occupancy Grid:

$$R_{i,t} = \sum_{t \le t' < t + \Delta t} R_{i,t'} \quad (6)$$

By using this history of grids, the proposed algorithm can estimate the current state of the environment not only by the last observed map but from the history of the observations. With the use of the history of grids, the Equation 5 that describes the ML-SLAM algorithm is transformed as follows:

$$\langle m_t, s_t^* \rangle = argmax_{m_t,s_t} \mathbf{p}(s_t, m_t | z_t, u_t, m_{1:t-1}, s_{t-1}^*)$$
$$= argmax_{m_t,s_t} \mathbf{p}(z_t | s_t, m_t, m_{1:t-1}) \mathbf{p}(s_t, m_t | u_t, s_{t-1}^*, m_{1:t-1})$$
$$= argmax_{m_t,s_t} \mathbf{p}(z_t | s_t, m_t) \mathbf{p}(s_t | u_t, s_{t-1}^*) \mathbf{p}(m_t | m_{1:t-1}) \quad (7)$$

Comparing the Dynamic ML-SLAM Equation 7 with the ML-SLAM Equation 5, we can observe that difference between the two approaches can be located in the Map Model $\mathbf{p}(m_t|m_{1:t-1})$. As we can see, the probability of a map does not only depend on the previous most likely map, m_{t-1}^*, as with the ML-SLAM, but on the entire history of maps $m_1 - m_{t-1}$.

4.1 Map Model

In order to evaluate a map based on the history of maps, we make two assumptions. At first, we assume that the prediction for individual cells of the map is independent of the predictions for the other cells given the history of maps. Next, we assume that each cell evolves independently from the others. Thus, we decompose the problem of estimating the Map Model into a number of problems, the estimations of the Cell Model for all the cells of the environment:

$$\mathbf{p}(m_t|m_{1:t-1}) = \prod \mathbf{p}(m_{i,t}|m_{1:t-1}) = \prod \mathbf{p}(m_{i,t}|m_{i,1:t-1}) \quad (8)$$

where $m_{i,t}$ denotes the i^{th} cell of the map m_t.

In order to estimate how probable an occupancy value is for the i^{th} cell given the history of its occupancies values, we can count the occurrences of this value in its history. For example, imagine a cell that contains a static object (i.e. a wall). The cells history will be filled with the "occupied" value. So, the probability of the cell being occupied will be 1 and the probability of being free will be 0. In the case, however, that the cell corresponds to a closed door configuration, when the door is closed the cell occupancy will be zero and when the door is open the cell occupancy will be one. So, the history of the cell will contain different occupancy values, thus we will calculate different non-zero values for the probabilities of being occupied and of being free.

The above can be summarized in the following formula:

$$\mathbf{p}(m_{i,t} == state | m_{i,1:t-1}) = \frac{|\{j : m_{i,j} == state\}|}{|\{j : m_{i,j}\}|} \quad (9)$$

4.2 Histogram Occupancy Grid

Equation 7 implies that a history of all maps must be available at any time point. One structure that can be used to store the history is the Temporal Occupancy Grid as described in [10]. On the other hand, Equation 9 indicates that we do not need to preserve the whole history of the cell occupancy evolution. Instead, we only need to know how many times a cell has taken specific occupancy values. So, to save memory, we use the Histogram Occupancy Grid, a grid that each cell contains a histogram. Every histogram contains a list of probabilities that indicate how probable the occupancy of a cell is. An example of a histogram can be found in Figure 1. We have to note here that instead of a histogram we could use any other representation of probability distributions (e.g. particles).

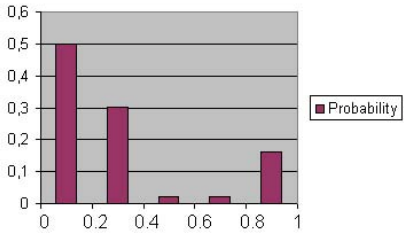

Figure 1. Example of a cell histogram

4.3 Most likely pose determination

On the ML-SLAM, to find the most likely pose, we can apply the gradient ascent technique (hill climbing) on the Equation 5, as indicated in [15]. In this way, we can find the local maximum of the available robot poses (an example in Figure 2 where the orientation is ignored). However, on the Dynamic ML-SLAM, we cannot apply a simple hill climbing. When moving in a dynamic world, there might exist more than one local maxima. A spurious measurement might be explained either by a dynamic effect of the environment or by an erroneous pose estimation. In Figure 3, an example of a local maximum that could result in a failure of the gradient ascent technique is presented. Instead, a randomized and hill climbing search could be applied. Random restarts with different starting points will eventually detect the global maximum.

4.4 Restrictions

We should note here that the Dynamic ML-SLAM inherits the inability to detect Loop Closures from the ML-SLAM algorithm. Tiny odometry errors could result in enormous error in the estimated robot pose. This causes the inability to detect if the robot has returned to a previous pose. The nature of this algorithm, the fact that previous estimations of the robot pose are frozen and can never be changed when new sensor measurements arrive, makes it inapplicable to cyclic environments.

For the same reason, this algorithm will fail when applied in a highly dynamic environment (i.e. an environment with many objects that move arbitrarily). On these environments, the pose probability function will probably contain

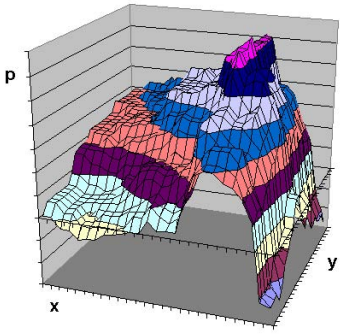

Figure 2. Example of a pose probability in static environment

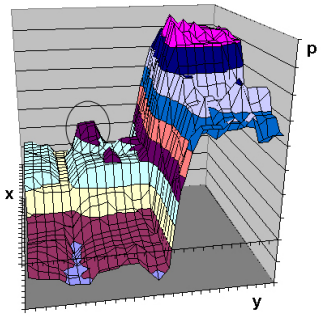

Figure 3. Example of a pose probability in dynamic environment

more than one local maximum with similar probabilities. We will not be sure which of these maxima will correspond to the correct pose. The fact that we do not preserve any uncertainty on the robot pose, makes it impossible to recover from erroneous estimations of the robot pose that have been made in the past.

5 Experiments

In order to evaluate the proposed algorithm, we performed a number of experiments with a simulated pioneer robot equipped with a laser range finder. The environment used throughout the experiments was a short passage with eight doors. Errors were added in the robot odometry in order to create more challenging conditions (systematic 3 degrees rotational error on every 5 cm of traveled distance). We have to mention that if no odometry errors were added, both the ML-SLAM and the dynamic ML-SLAM will perform almost identically. That is because the ML-SLAM would silently ignore the dynamic areas and use the static areas to localize the robot. The occupancy grid map of the raw odometry data is presented in Figure 4.

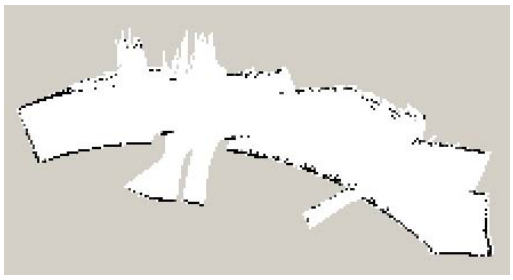

Figure 4. Raw odometry data in static environment

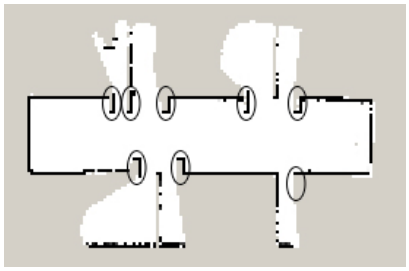

Figure 5. Occupancy Grid generated by the ML-SLAM applied in a static environment

At first, an experiment in a static environment (the doors preserved their states during the experiment) was performed. Both ML-SLAM and Dynamic ML-SLAM performed excellent under these conditions. The result of the ML-SLAM is presented in Figure 5. The eight circles indicate the doors of the environment. An identical map was created by the Dynamic ML-SLAM.

A second experiment was conducted in the same environment but at this time the states of the doors were changing every 5 to 20 seconds (at random). The robot followed the same path as with the previous experiment and the ML-SLAM and Dynamic ML-SLAM algorithms were applied.

In figure 6, the Occupancy Grid map generated by the ML-SLAM algorithm in this dynamic environment is presented. We can observe that the map error is significant. The left part of the map is badly positioned with respect to the rest of the map. In figure 7, the map of the environment obtained by the execution of the ML-SLAM is presented. The generated map is of higher quality in comparison to the one in Figure 7. The main difference with the map in Figure 5 is that the left part of the map is slightly turned as the algorithm failed to match the two lines indicated in the spotted area. The dynamics of the area is higher in comparison to other areas as there exist two doors near this area. So, on highly dynamic areas, the algorithm will fail due to the fact that we do not preserve the uncertainty on the robot pose and thus we can not correct previous estimations.

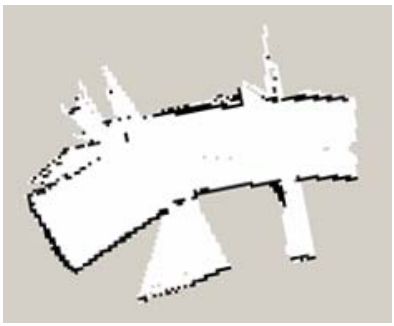

Figure 6. Occupancy Grid generated by the ML-SLAM applied in dynamic environment

Figure 7. Occupancy Grid generated by the Dynamic ML-SLAM

6 Conclusions

In this work, a new SLAM algorithm that can be applied in dynamic environments and the appropriate statistical framework that this algorithm is based on were presented. We introduced the Histogram Grid structure that preserves the history of the cells probability and we used this structure to identify dynamic areas and enhance the localization algorithm. The obtained maps are shown to be more accurate because of the improved pose selection process. Simulated results prove the correctness of this approach.

As a future plan, we will conduct a number of experiments with a real robot in different dynamic environments. The results are expected to agree with those shown in this work. To deal with the loop closure problem that was described earlier, it is in our plans to combine the Dynamic ML algorithm with the Particle Filtering algorithm in order to model the uncertainty on the robot pose with particles. This can yield in a robust algorithm that might be able to successfully localize the robot in a highly dynamic environment and produce a valid map of the environment.

References

[1] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in non-stationary environments with mobile robots. In *Proc. of the Conference on Intelligent Robots and Systems (IROS)*, pages 1301–1306, 2002.

[2] W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. Sonar-based mapping of large-scale mobile robot environments using em. In *Proc. of the 16th International Conference on Machine Learning (ICML '99)*, 1999.

[3] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, 1989.

[4] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[5] D. Fox, W. Burgard, S. Thrun, and A. Cremers. Position estimation for mobile robots in dynamic environments. In *Proc. of the AAAI Fifteenth National Conference on Artificial Intelligence*, 1998.

[6] J. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 318–325, Monterey, California, November 1999.

[7] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Freiburg, Germany, 2003. University of Freiburg.

[8] G. Q. Huang, A. B. Rad, and Y. K. Wong. A new solution to map dynamic indoor environments. In *International Journal of Advanced Robotic Systems, Vol. 3*, 2006.

[9] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal of Intelligent and Robotic Systems*, 18:249–275, 1997.

[10] N. Mitsou and C. Tzafestas. Temporal occupancy grid for mobile robot dynamic environment mapping. In *Proceedings of the 15th Mediterranean Conference on Control and Automation IEEE International Conference*. IEEE, 2007.

[11] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proc. of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.

[12] D. Schulz, W. Burgard, D. Fox, and A. Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association, 2001.

[13] C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *Proc. of the National Conference on Artificial Intelligence*, Pittsburgh, PA, USA, 2005.

[14] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[15] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *In Proc. of the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.