Dynamic Object Detection and Representation for Mobile Robot Application

Jose J. Lopez-Perez, Victor Ayala-Ramirez(✉), and Uriel H. Hernandez-Belmonte

División de Ingenierías, Campus Irapuato-Salamanca DICIS, Universidad de Guanajuato DICIS, Carr. Salamanca-Valle Km. 3.5+1.8, Palo Blanco, 36700 Salamanca, Mexico
{jjlp,hailehb}@laviria.org, ayalav@ugto.mx

Abstract. This paper presents how to classify the sensorial information to set the label of the cells in the occupation grid map of an unknown environment where several robots are wandering and how to use these labels to complete a path execution task. Reactive navigation of a mobile robot needs to identify the objects that encounters in its way towards a specified goal. The robot does not know the environment but it knows the position of its goal. The robot needs to build a map during its navigation to the goal. A pattern classification task is needed to update the state of the map grid cells as more information is acquired by the sensors. We propose to use a finite state machine approach to set the correct label according to the objects in the environment. The occupancy grid map is crucial to complete the navigation task safely. We have tested our approach in several scenarios with a varying number of mobile obstacles. The results show that our method works efficiently in a set of benchmark scenarios of varying degree of complexity.

1 Introduction

Autonomous navigation in unknown and dynamic environments is classified as one of the more challenging tasks in the mobile robotic community [4]. This capability is needed for the robot to navigate safely in environments populated either by other robots or people. In such an environment, the other robots or people behave in ways difficult to model and the autonomous robot has to react in order to guarantee the safety of all the entities sharing the same environment. Another scenario where safe navigation is required is the development of autonomous vehicles. Safety also needs to be guaranteed to let the autonomous vehicles to complete the navigation tasks without colliding among them.

We propose to use an occupancy grid map representation for the unknown environment. The occupancy grid is a well known approach in mobile robotics that has shown to be robust even if it shows also several disadvantages [1]. Some recent works use occupancy grids in the safe navigation context. For example, Li and Ruichek use it to map urban environments [5]; Wang *et al.* use it to track

J.F. Martínez-Trinidad et al. (Eds.): MCPR 2016, LNCS 9703, pp. 84–93, 2016.
DOI: 10.1007/978-3-319-39393-3_9

dynamical objects [9] and Xin *et al.* use it for representing the dynamic environment of an autonomous vehicle [10]. In this paper, we also use an occupancy grid in a reactive safe navigation task.

We use the information provided by a laser range finder (LRF) to classify the cells in the occupancy grid and a finite state machine (FSM) to determine if obstacles are moving or steady.

Rest of this paper is organized as follows: in Sect. 2, we present the main details of our approach. The implementation details, the tests and the results are presented in Sect. 3. Finally, our conclusions are presented in Sect. 4.

2 Methodology

2.1 Path Execution Task

The robot considered in this paper has to execute a safe navigation task. The current position $S = (x_s, y_s)$ and the goal position coordinates $G = (x_g, y_g)$ are known. The environment dimensions $W \times H$ are also known.

The mobile robot is provided with proprioceptive and exteroceptive sensors. The proprioceptive sensors are used to compute the current position of the robot. The exteroceptive sensors let the robot to acquire information about its surrounding area. In this navigation task, the robot uses a laser range finder (LRF) to get the measures from the obstacles that are near to it.

A LRF sensor operates by sending a laser beam in a given number L of orientations from the current position of the mobile robot. The time of flight lectures that the LRF records are used to compute how far the obstacles are. However, these measurements give no information about the nature of the obstacle.

The robot uses an occupancy grid to represent the information acquired from the environment during the execution of the path towards its goal. The occupancy grids were originally proposed by Elfes in [1] and they are widely used for mobile robot path planning tasks [10]. This approach discretizes the environment into a grid of cells of some dimensions. Figure 1(a), shows how an environment is discretized into an occupancy grid and the implicit inaccuracies in the scenario representation. Essentially, each cell can be labelled as a free cell or as an obstacle cell (see Fig. 1(b)). If the dimensions are too small, the representation of a small map can take a lot of memory space and can not be useful for the planning task. In our case, we discretize the environment as a grid of cells of dimensions $0.5\,m \times 0.5\,m$. In our work, we use the occupancy grid for two objectives: (i) to have a partial representation of the explored part of the scenario and (ii) to ease the planning algorithm when a re-planning is needed.

Using the occupancy grid approach, the mobile robot computes a path from the start position S to the goal position G by using a $A^\star$ planning algorithm [7,8]. Initially, the robot does not know any information about the environment, so it considers that all the map is free space.

2.2 Occupancy Grid Update

As the robot advances in the execution of its planned path, it needs to update the current map representation M according to the planned path P and to the obstacle information obtained from the LRF sensor L. After processing the current lectures of the LRF sensors, an updated map M' and a re-planning flag f_{rp} are computed. This enables the robot to handle the mobile and steady obstacles present in the environment.

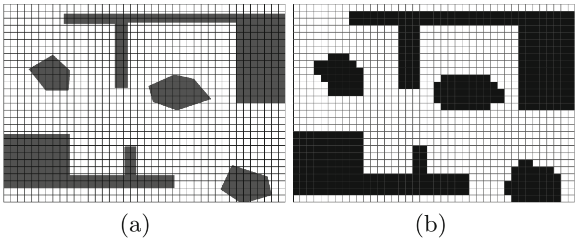


Fig. 1. (a) Scenario representation using occupancy grids. (b) Cells are labeled as *obstacle cells* (in black color) and as *free cells* (in white color) (Color figure online)

The cells in the maps M and M' can be labeled using four possible values: *non explored cell*, *explored cell*, *steady obstacle cell*, and *mobile obstacle cell*. The labeling procedure depends on the computation of two functions:

f1 that serves to determine which cells are occupied by an obstacle, either steady or mobile obstacle. This function uses the collision points of each of the laser measures $L = \{l_1, l_2, \ldots, l_L\}$ with L being the maximum number of lectures taken by the LRF.

f2 that serves to update the label of all the cells inside the polygon formed by the laser measures to the *explored cell* status.

For the function f_1, we obtain the collision points $l_j(x_j, y_j)$ for each of the measurements of the LRF. The distance from these points $l_j(x_j, y_j)$ to each of the center positions $c_i(x_i, y_i)$ of the map grid are computed. If the distance $d(lj, c_i)$ is smaller than the safety radius of the mobile robot r, the function return 1, otherwise it returns 0 (See Fig. 2).

$$d(l_j, c_i) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} = \|l_j - c_i\| \tag{1}$$

$$f_1 = \begin{cases} 1 \;\; \text{si} \;\; d(l_j, c_i) \leq r \\ \\ 0 \;\; \text{si} \;\; d(l_j, c_i) > r \end{cases} \tag{2}$$

For the function f_2, we generate a polygon using the laser collision points $l_j(x_j, y_j)$ as vertices. If a cell is inside the polygon, f_2 takes the value of 1,

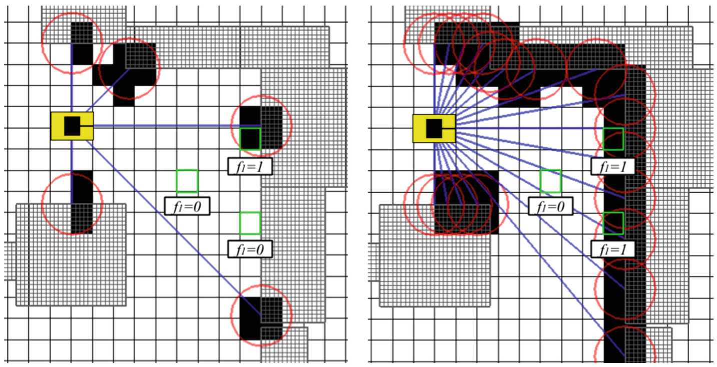


Fig. 2. Computation of f_1 function. *Steady obstacle cells* in black, *non explored cells* in white, measurement of the LRF in blue (Color figure online).

otherwise it returns a value of 0. The computation to know if the point $c_i(x_i, y_i)$ is in inside the polygon is made by tracing a vertical line and counting the number of intersections with the laser measurement polygon. If there is an odd number of intersections, the function f_2 returns 1, otherwise it returns 0 (See Fig. 3).

The transition between the states that a cell can have is described via the FSM diagram in Fig. 4. Note that the combination $(f_1, f_2) = (1, 1)$ is not represented there. That is because it can not occur in any state. When f_1 is 1, f_2 takes automatically the value of zero. All the cells are initially labeled as *non explored cells.*

If a cell changes from the state *non explored* to the state steady obstacle or from the state *explored* to the state *mobile obstacle*; and if this cell is also part of the planned path P, the f_{rp} flag is enabled. If this flag is enabled, the robot must do a re-planning using the partially known environment.

3 Results

3.1 Implementation Details

The proposed system was developed in a modular form to be able to implement it in frameworks like ROS (*Robot Operating System*) or ARIA (*Advanced Robot Interface for Applications*). These are the frameworks used by the robots available at the Laboratory for Vision, Robotics and Artificial Intelligence (LaViRIA) of the University of Guanajuato: a Husky A200 mobile robot named CompaBot (in Fig. 5(a)) and a Pioneer 3-AT named XidooBot (in Fig. 5(b)).

3.2 Test Protocol

The tests were performed using a benchmark set of 34 scenarios shown in Fig. 6. They are available from the *motion planning repository* [2]. These scenarios are

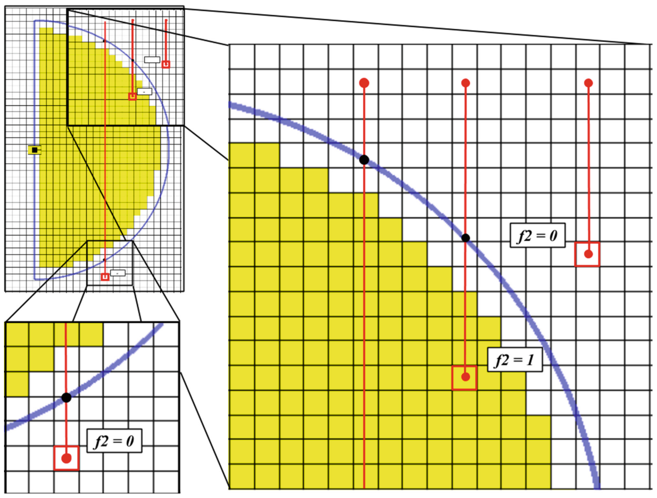


Fig. 3. Computation of the f_2 function. *Explored cells* in yellow, *non explored cells* in white, LRF measurement polygon in blue (Color figure online).

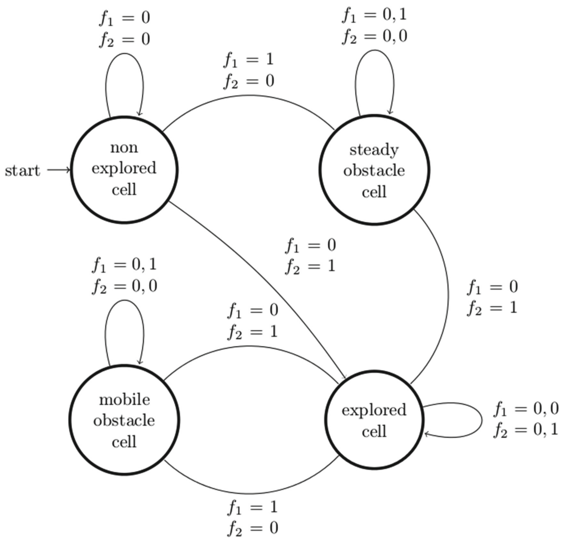


Fig. 4. State transition diagram for each cell of the occupancy grid.

very similar to those used for real world applications. These set include some difficult problem like robot traps, narrow corridors, zones with a large density of obstacles, labyrinths, rooms, etc.

For each of the test scenarios, 3 different navigation tasks were generated, i.e., a different start and goal position of the robot were chosen. For each navigation

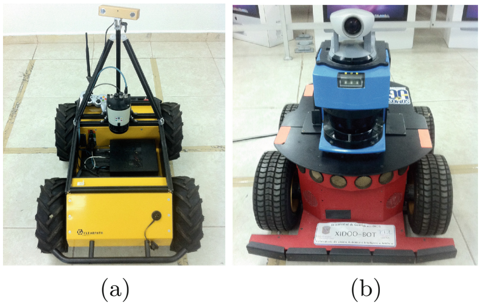


Fig. 5. (a) CompaBot, a Husky A200 mobile robot from Clearpath Robotics Inc. (b) XidooBot, a Pioneer 3-AT mobile robot designed by Adept MobileRobot.

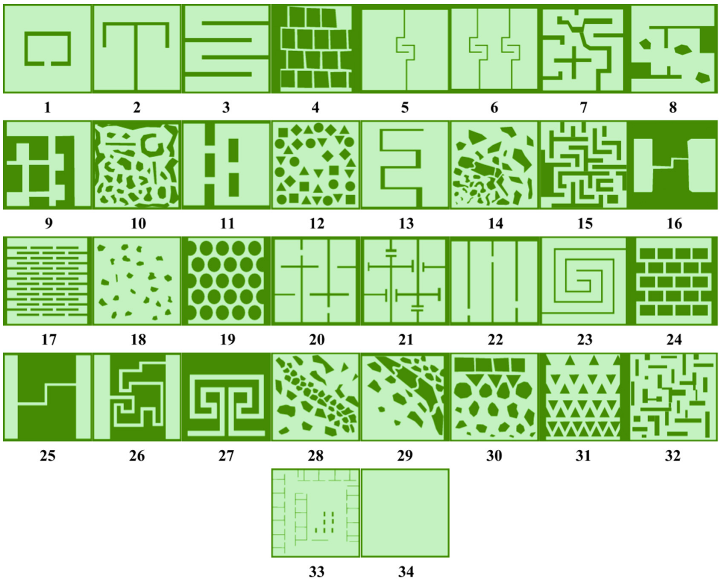


Fig. 6. Benchmark set of test scenarios from the repository of the Technical University of Prague [2].

task, both the start and goal position are in the free space of the environment under use. This procedure was repeated 30 times for each configuration and navigation task with a number of existing mobile obstacles on stage that was increased from 0 to 10. For the mobile obstacles, we have used several instances of the Pioneer 3-AT mobile robot that wander around the free space, simulating the human-robot interaction in a scenario.

3.3 Gazebo and RViz Simulation Using ROS

In order to implement the proposed system for safe navigation of mobile robots in dynamic unknown environments, we implement it on the ROS testbed. This enable us to implement it in a Husky A200 platform given that this platform is supported by ROS. The main simulation tools for ROS are Gazebo and RViz. In the following paragraphs, we describe the main features of these tools:

ROS is a development platform for writing robot software. It is composed of a collection of tools, libraries and standard messaging protocols that simplify the robot software development task for different robotic hardware [6].

Gazebo is a robot simulator that is useful for rapid prototyping and test of robot algorithms. It can be used to design robots and to perform the analysis of the simulation data in realistic environments. It also offers the capability of simulation of teams of robots in complex outdoor and indoor environments with accuracy and efficiency. Another important features of Gazebo are the robust physics engine and its high quality graphics rendering [3].

RViz is a 3D data visualization tool available in ROS. It can show the sensor data and the current state of ROS. RViz can be used to visually show the current state of the robot in a virtual model. It can also be used to display real-time sensor information, including data from the cameras, the sonar, the infrared devices and the map representation, for example [6].

3.4 Quantitative Results

Figure 7 shows the success rate for each test scenario. In most of the scenarios under test the success rate is over 90 %. However, we can observe that the scenarios with narrow corridors and with only one route to access the goal position, the system exhibits a low success rate (Fig. 6 scenario 15 - *maze*, 23 - *square_spiral*, 26 - *tunnel_twisted* and 27 - *tunnel*). The factors that cause the mobile robot failing to reach the goal position is the continuous obstruction of the mobile obstacles in its route or the presence of an error in the detection and classification of the mobile obstacles.

3.5 Qualitative Results

Here we present the qualitative results of applying the proposed method in a Husky A200 mobile robot in the Gazebo simulator. The robot model uses a LRF with 640 measurements and a vision field of 180° in front of the robot. We show the results in the *back_and_forth* test scenario (Fig. 6 scenario 3). The aforementioned scenario was built in Gazebo using the same size specifications than in the benchmark set. Figure 8 depicts the component modules of the ROS implementation. The proposed navigator, labeled as *dpn* (dynamic planner navigator), publishes the map that is building (*/compabot/map*) and the robot velocities (v y ω) to the simulator (*/compabot/husky/cmd_vel*). It also publishes the

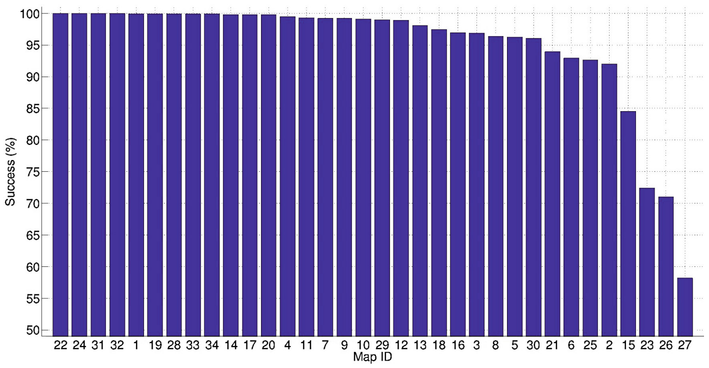


Fig. 7. Result of the success rate (%) vs map. In most scenarios test the success rate is over 90 %.

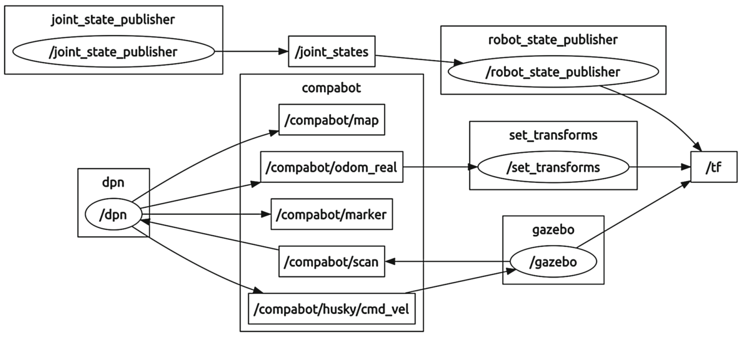
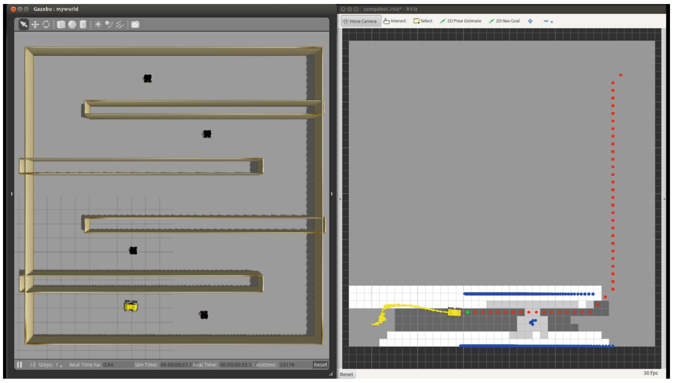


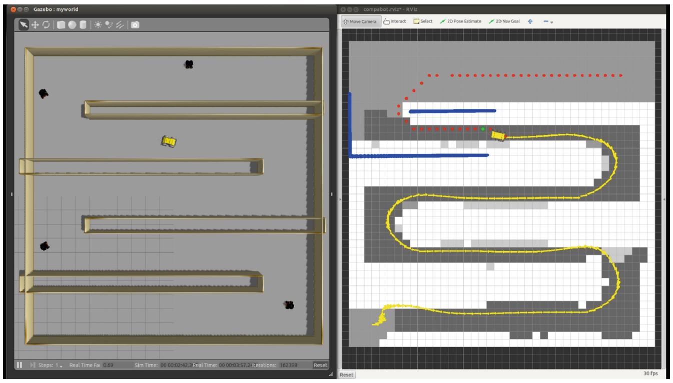
Fig. 8. Connection diagram for the ROS implementation of the proposed system.

current position of the mobile robot (*/compabot/odom_real*) and the information about its planned path (*/compabot/marker*), and it gets the laser scan data (*/compabot/scan*) from the simulator.

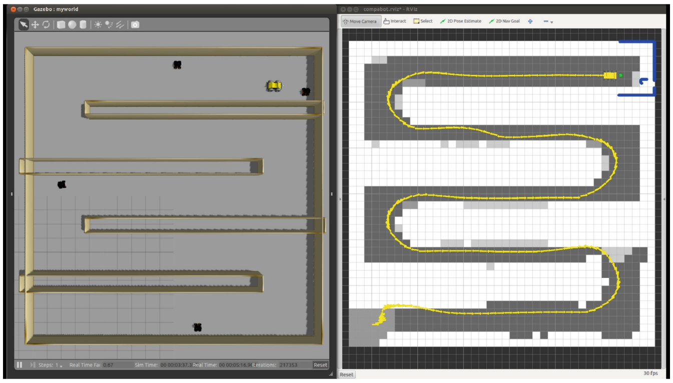
Some aspects of a mobile robot simulation are shown in Fig. 9. In the left column, we present the Gazebo display output and in the right column we present the RViz display output. The Gazebo output serves to render the current state of the virtual world. In our case, it renders the Husky A200 model and the models of the P3-AT robots that are wandering in the environment. In the RViz output, we can observe the model of the Husky A200 mobile robot and the executed path (in yellow), the planned path (in red),the map created by the mobile robot where the white cells are explored cells. The steady obstacles are represented using black cells and the moving obstacles are colored using a gray color, We can observe how the map is evolving to reflect the current knowledge about the environment.



$t = 33s$.



$t = 237s$.



$t = 316s$.

back_and_forth.

Fig. 9. Simulation results for the proposed navigator in Gazebo (in the left) and RViz (in the right). The timestamp t of each frame is shown below the snapshot.

4 Conclusions

In this work, we have proposed a cell labeling method for building the map of a dynamic unknown scenario. The method is based in a Finite State Machine approach. We have presented the rules that govern the transition for the cell labels of the occupancy grid map as the robot acquires information from its sensors. The method was tested using scenarios of different degrees of complexity and it has shown to be efficient and accurate. For the tests, we have used the ROS testbed. Future work will include to extend the tests to real robots. The use of this approach as a part of a simultaneous localization and mapping (SLAM) method will also be performed.

Acknowledgments. Jose J. Lopez-Perez and Uriel H. Hernandez-Belmonte would like to acknowledge CONACYT for the financial support through the educational scholarships with numbers 291047/736591 and 229784/329356 respectively.

References

1. Elfes, A.: Using occupancy grids for mobile robot perception and navigation. Computer **22**(6), 46–57 (1989)
2. Intelligent and Mobile Robotics Group: Motion planning maps. http://imr.ciirc.cvut.cz/planning/maps.xml. Accessed 29 Sep 2015
3. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 2149–2154. IEEE (2004)
4. Laugier, C., Chatila, R.: Autonomous Navigation in Dynamic Environments. Springer Tracts in Advanced Robotics. Springer, Heidelberg (2007). Chap. Preface
5. Li, Y., Ruichek, Y.: Occupancy grid mapping in urban environments from a moving on-board stereo-vision system. Sensors **14**, 10454–10478 (2014)
6. Martinez, A., Fernández, E.: Learning ROS for Robotics Programming. Packt Publishing Ltd, Birmingham (2013)
7. Murphy, R.: Introduction to AI Robotics. MIT Press, Cambridge (2000)
8. Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D.: Introduction to Autonomous Mobile Robots. MIT Press, Cambridge (2011)
9. Wang, D.Z., Posner, I., Newman, P.: Model-free detection and tracking of dynamic objects with 2D lidar. Int. J. Robot. Res. **34**(7), 1039–1063 (2015)
10. Xin, Y., Liang, H., Mei, T., Huang, R., Du, M., Sun, C., Wang, Z., Jiang, R.: A new occupancy grid of the dynamic environment for autonomous vehicles. In: Proceedings of the 2014 Intelligent Vehicles Symposium, pp. 787–792. IEEE (2014)