# Robotic mapping and navigation in unknown environments using adaptive automata

Miguel Angelo de Abreu de Sousa and André Riyuiti Hirakawa

Escola Politécnica da Universidade de São Paulo,
Departamento de Engenharia de Computação e Sistemas Digitais,
Av. Prof. Luciano Gualberto, travessa 3, nº 158. São Paulo - SP - Brasil - CEP 05508-900
{miguel.sousa, andre.hirakawa}@poli.usp.br

## Abstract

Real mobile robots should be able to build an abstract representation of the physical environment, in order to navigate and work in such environment. This paper presents an adaptive way to make such representation without any *a priori* information of the place. The proposed system allows the robot to explore the entire environment and acquire the information incoming from the sensors while it travels and, due to the adaptability inherent to the mapping method, the system is capable to increase the memory usage according to the already mapped area. The map, built using the adaptive technique, is useful to provide navigation information for the robot, allowing it to move on the environment.

## 1 Introduction

Robotic mapping is one of the most important requirements for a truly autonomous mobile robot. Such robots should be able to build their own abstract representation of the physical environment instead of use a preliminary map stored in its memory. The main reasons are: the repetitive and hard task to manually map each house, office, factory, street and agricultural field which robots are intent to work and manually register that map in the their memory. The second reason is related to the possibility of robots work in hazardous, unstructured and unknown environments. Such places may be dangerous or impossible for humans to construct a map. Then, robots must build their own map before being able to execute others tasks [1].

Another question related to robotic mapping is the way to acquire such maps. Complete geometrical representation of the environment increases the data amount and the computational complexity for the database searching during the robot localization and planning process.

This work employs adaptive mechanisms to collect information from the world and to navigate in that world. Such adaptive mechanism allows the system to build maps without *a priori* information of the place, and also, allows memory space occupied just grows up accordingly to the already mapped area.

In order to present these adaptive mechanisms, the following section presents the model used in this work, section 3 sketches the adopted formalism, section 4 describes the mapping structure, section 5 describes the automaton responsible for steer the robot during the exploration of the environment and section 6 presents the navigation mechanism using the built map.

## 2 The Model

The robotic mapping and navigation model proposed in this work is depicted in figure 1:

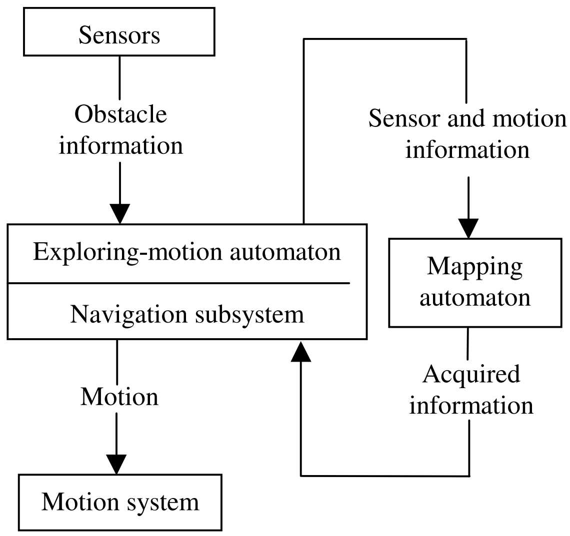


Fig. 1. System model.

In the model, the Exploring-motion automaton, receives data coming from the sensors and the current neighborhood information extracted from the Mapping automaton. Its output contains information on the next move of the robot during its exploration's movement. After the conclusion of the mapping process, decisions on navigation movement are performed by the Navigation subsystem, which also receives data from the sensors and the map. The Mapping automaton is responsible for storing all the sensor information on the presence or absence of obstacles through the robot travel path. The environment, sensors and motion information have been simulated in order to validate the proposed mechanisms.

## 3 Adaptive Automata

Adaptive automata extend the concept of finite automata by incorporating the feature of performing dynamic self-reconfiguration in response to external stimulus. Consequently, their behavior may be changed according to collected information. Such feature represents a trustful way for modeling physical environments and to conduct the robot, despite the complexity of the place. It has been shown that adaptive automata are Turing-powerful devices [7]. Figure 2 shows the graphic representation of adaptive automata, where:

e: current state before the transition;
e': current state after the transition;
a: input stimulus before the transition;
B: adaptive action before applying the transition;
A: adaptive action after applying the transition.

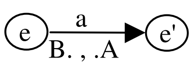


Fig. 2. Adaptive automata transition.

Adaptive actions A and B are both optional. Three different elementary adaptive actions are allowed: inspection – search the current state set for a given transition; deletion – erase a given transition from the current state set; and insertion – add a given transition to the current set of states. Such actions are denoted by preceding the desired transition by the signs ?, – and +, respectively. Reference [7] details the concept and notation of Adaptive Automata.

## 4 The Mapping Automaton

This paper extends a previous work that represents physical environments by using adaptive automata [3].

The proposed Mapping automaton has the initial configuration of a square lattice (see figure 3a), consisting of nine states connected by a set of transitions denoting areas to be mapped. The central state is the initial state of the automaton, and represents the starting point of the robot's exploring path.

In order to complete the representation of the initial automaton, figure 3b presents the dot-marked state (●), corresponding to the actual position of the robot, special tags (X), marking corner states, and special transitions, which are provided for supporting expansions in the lattice.

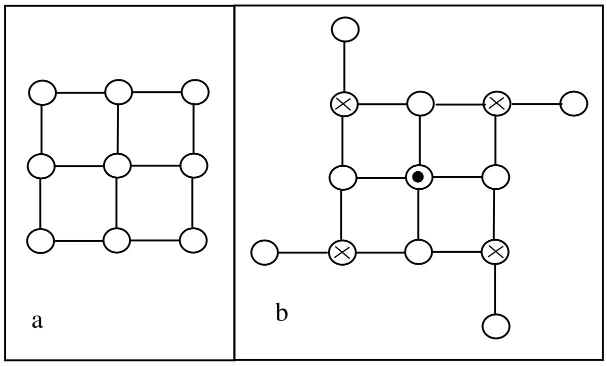


Fig. 3. (a) Initial lattice in an adaptive automaton. (b) Detailed initial automaton lattice.

Once the automaton is supplied with the data collected by the robot's sensors while it performs exploring motions, the four adjacent non-filled transitions are properly replaced according to that information. The information collected by the sensors contain indications on the direction – north (N), south (S), east (E) or west (W) – and its conditions – free or busy. Figure 4a shows one example of the four-data information collected by sensors. In the figure, double arrows indicate non-obstructed areas (directions S and E) and bold lines denote obstructed ways (directions N and W).

Robot's movement in the environment is represented by some consistent state change in the Mapping automaton. As the robot moves, the current lattice is expanded in the direction of the movement. This expansion is performed by adding a line or a column to the existent lattice. Figure 4b exemplifies the result of an N-move, after reconfiguring the vacant marks corresponding to all free directions.

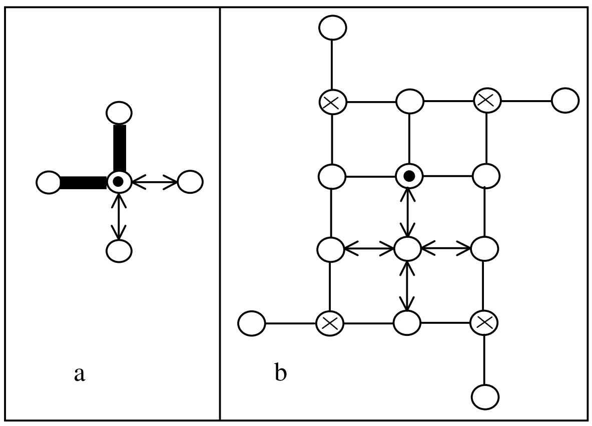


Fig. 4. (a) Example of information coming from the sensors: two directions obstructed and two free directions. (b) Expanded lattice after an N-move.

Figure 5 shows a complete map with the representation of the information acquired by the automaton after exploring an empty 'L' room. The robot has completed the exploration at the rightmost upper space of the room, which is represented by the dot-marked state.

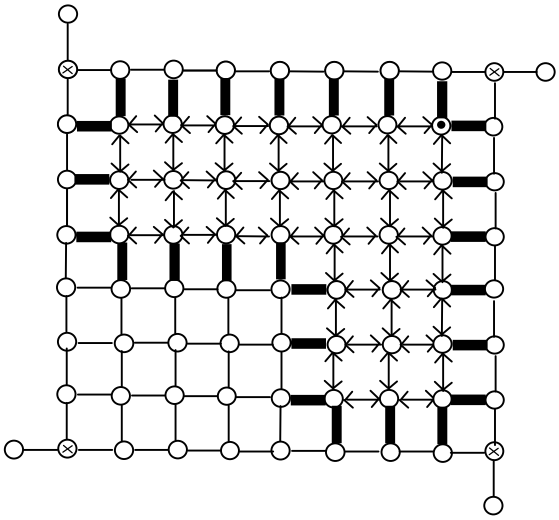

Fig. 5. Example of an empty 'L" room mapped.

The use of memory space is proportional to the existence of transitions and states. Then, although the memory usage is not proportional to the mapped area (how the vacant transitions indicate in figure 5), this memory usage increases with the actually mapped area.

To keep the relation between the piece already built of the map and the real environment, the initial state of the automaton is adopted as representing the origin of the map. This state corresponds to the initial mapping place. So, any point in the map is associated to any point in the physical environment through the association of each transition in the map's representation to some corresponding displacement performed by the robot in the real world.

## 5 The Exploring-Motion Automaton

An adaptive automaton is employed for determining the robot's next move. For this purpose, it is supplied with four-data information collected by the sensors (figure 4) and with neighborhood eight-data information previously modeled in the map (figure 6). The Exploring-motion is basically conducted in a zigzag behavior.

The zigzag path, guided by the Exploring-motion automaton, may be adapted according to the type of environment detected and according to the stage of exploration. To perform this adaptation, there are several branches of the Exploring-motion, one for each type of possible situations. All of those branches may be connected to the initial state of the Exploring-motion automaton by a transition. That connection is changed to the branch which should deal with the information coming from the sensor and from the map in a specific stage of exploration.

As described in section 3, that changing of connection between the initial state and the first state of the branch may be performed by adaptive actions, which allow insertion and deletion of transitions.

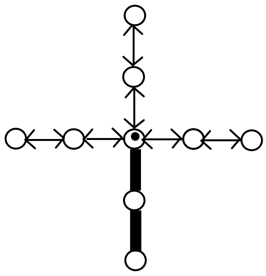

Fig. 6. Example of information extracted from the map representing two obstructed transitions and six free transitions.

While the environment is explored, the Exploring-motion automaton may sign some special states at the Mapping automaton, called landmarks, which are properly marked on the map. During the navigation process such landmarks are helpful for plan a trajectory.

If an obstacle is detected during the exploration's movement, the proposed automaton marks the middle point on the free space before and/or after the obstacle and it signs to the Mapping automaton that this point is a landmark. Figure 7 depicts the landmarks defined for an example environment. The Mapping automaton performs that representation by properly adding a transition connecting the signed state to a special state. This special state indicates all the landmark states. Figure 8a shows an example of two states, which are close, signed as landmarks, in which the tag (X) marks the special state.

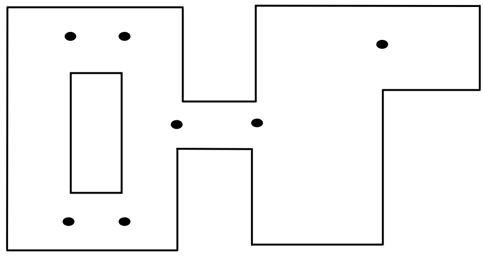

Fig. 7. Landmarks defined for an environment-example.

## 6 Navigation

During the navigation process, the landmarks are helpful for plan a trajectory from some initial position to a target position. The system calculates the path between such landmarks before the navigation and, during navigation, it must find which landmarks are nearest to the initial and to the target positions [4], [6]. Then, those landmarks may be viewed as sub-goals to the navigation process.

The calculus of the path between two landmark-states in the Mapping automaton is performed by the using a sequence generator. The sequences generated become the input of the Mapping automaton, which is configured to present the landmark states as initial and final states. Then, the accepted sequences by the Mapping automaton are saved as the sequences which represent the path between two landmarks.

The four sequences firstly generated contains each one of the four main directions: N, S, E or W. Then, the following sequences generated are extensions of the previous ones. For example, the three sequences generated from the N direction are NN, NE and NW (the sequence NS does not lead to a displacement, then it is not generated). Figure 8b shows the covered states by this example from the N direction (the triangle indicates the reached states and the dot-marked state (●) corresponding to the origin of searching).

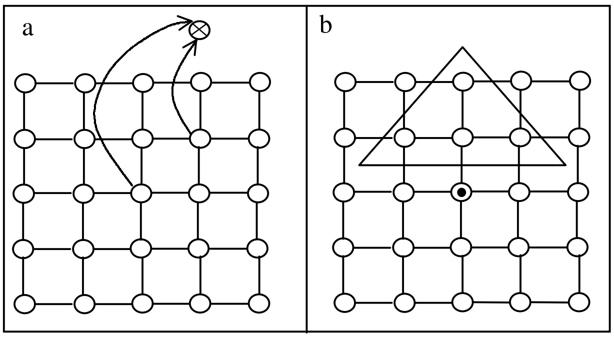


Fig. 4. (a) Example of two close states defined as landmarks. (b) Covered states by the first N-sequence generated.

## 7 Conclusions and Future Works

Robotic mapping is an important process for getting truly mobile robots and is also an essential feature to allow robots to complete certain tasks in hazardous, unstructured or unknown environments. The present work has shown an alternative to the classic robotic mapping approaches: adaptive mechanisms provide a new way to build maps and conduct the robot to the environment.

During the exploration, sensors attached to the robot scan the environment for the direction and presence or absence of obstacles, and such information is collected into the model by enabling the Mapping automaton to perform appropriate self-modifications. The Exploring-motion automaton allows the robot to cover the environment and provides special landmarks to the Mapping automaton, which may be used as sub-goals during navigation. The navigation process calculates the path between two of these landmarks.

This purpose contrasts with some classic approaches (e.g., [2], [5], [8]) by presenting features of building a map without *a priori* knowledge of the environment and memory space usage increasing with the actually mapped area.

Future works should deal with constructing landmarks with a serial identification attached in order to restrict the searching in the database during the calculus of the path between landmarks in the navigation process.

## References


[1] Brooks, R. A. *"Visual Map Making for a Mobile Robot",* Proceedings IEEE International Conf. Robotics and Automation., St. Louis, MO, 1985, pp. 824–829.

[2] Donald, B. R.; Howell J. *"Practical Mobile Robot Self-Localization"*, Proceedings International Conference on Robotics and Automation, San Francisco, CA, 2000.

[3] Hirakawa, A. R; Júnior, J. R. A.; Neto, J. J. *"Adaptive Automata for Independent Autonomous Navigation in Unknown Environment"*, International Conference on Applied Simulation and Modelling, Banff, Alberta, 2000.

[4] Innocenti, C. et al. *"Trajectory planning and real-time control of an autonomous mobile robot equipped with vision and ultrasonic sensors"*, Proceedings of the International Conference on Intelligent Robots and Systems IROS'94, Munich, September 12-16, 1994, pp.1861-1866.

[5] Jennings, J.; Kirkwood-Watts, C.; Tanis, C. *"Distributed Map-Making and Navigation in Dynamic Environments"*, Proceedings of the International Conf. on Intelligent Robots and Systems, Canada 1998, pp. 1695-1710.

[6] Ma, C.; Liu, L.; Li, W. *"Mobile Robot Motion by Integration of Low-Level Behavior Control and High-Level Global Planning"*, Proceedings of the IEEE International Conference on System, Man and Cybernetics, China, 1996, pp.310-315.

[7] Neto, J. J. "Adaptive Automata for Context-Dependent Languages", *ACM SIGPLAN Notices*, 1994, v.29, n.9, pp.115-24.

[8] Zimmer, U. R. *"Embedding local map patches in a globally consistent topological map"*, Proceedings of the International Symposium on Underwater Technology, Tokyo, Japan, 2000, pp.301-305.