Journal of Physics: Conference Series **90** (2007) 012088 doi:10.1088/1742-6596/90/1/012088

Maps Managing Interface Design for a Mobile Robot Navigation Governed by a BCI

Fernando A Auat Cheeín[1], Ricardo Carelli[1], Wanderley Cardoso Celeste[2], Teodiano Freire Bastos[2] and Fernando di Sciascio[1]

[1]Institute of Automatic, National University of San Juan. San Martin, 1109 - Oeste 5400 San Juan, Argentina.
[2]Electrical Engineering Department, Federal University of Espirito Santo. Fernando Ferrari, 514 29075-910 Vitoria-ES, Brazil.

E-mail: fauat@inaut.unsj.edu.ar

Abstract. In this paper, a maps managing interface is proposed. This interface is governed by a Brain Computer Interface (BCI), which also governs a mobile robot's movements. If a robot is inside a known environment, the user can load a map from the maps managing interface in order to navigate it. Otherwise, if the robot is in an unknown environment, a Simultaneous Localization and Mapping (SLAM) algorithm is released in order to obtain a probabilistic grid map of that environment. Then, that map is loaded into the map database for future navigations. While slamming, the user has a direct control of the robot's movements via the BCI. The complete system is applied to a mobile robot and can be also applied to an autonomous wheelchair, which has the same kinematics. Experimental results are also shown.

1. Introduction

The fusion of Brain-Computer Interfaces (BCI) with Robotics field is an area of increasing interest in the last few years. This is so because of the great advantages of robotics in rehabilitation treatments and the interactions human-machines in people with some kind of disabilities ([1], [2], [3]). The main objective of BCIs is to improve the life condition for those people, giving them more independence and the possibility of executing ordinary tasks ([4]).

In a BCI, the operator, or user, is capable to generate commands using his/her electroencephalographic signals (EEG). The BCI acquires those signals, processes them and generates a command that is sent to a device, which is the objective of the BCI ([5]). Such an objective could be the control of a robot manipulator, a text processor or a simple on-off controller. In this paper, the objective of the BCI is a control of the mobile robot Pioneer 2DX. This kind of robot has the same kinematics of a wheelchair. According to this, the system proposed in this work is applicable to a wheelchair governed by a BCI.

The BCI used in this work is an ERS/ERD (Event Related Synchronization and Desynchronization) based BCI ([6]).

The system proposed in this work is a graphical interface that manages several maps of real environments for robot navigation. The entire interface is governed by the same BCI. According to this, when the robot is inside a known environment (such an environment with a map in the database) the corresponding map is loaded and the user can generate several trajectories in order to navigate through it. On the other hand, if the robot is inside an unknown place, then a Simultaneous

Localization and Mapping algorithm ([7]) is settled and a probabilistic grid map of the environment is acquired. Once the map is complete, it is added to the database for future navigations. Thus, this graphical interface can represent, for example, the different rooms of a house and those places that do not belong to the house and are visited by the first time can be mapped and then loaded into the graphical interface.

This work is organized as follows. Section II, III, IV and V show the complete system architecture, including an explanation of the SLAM algorithm used as long as the probabilistic mapping process; section VI shows some experimental results and section VII are the conclusions of this work.

2. Main System Architecture

Figure 1 shows the interface proposed in this work.

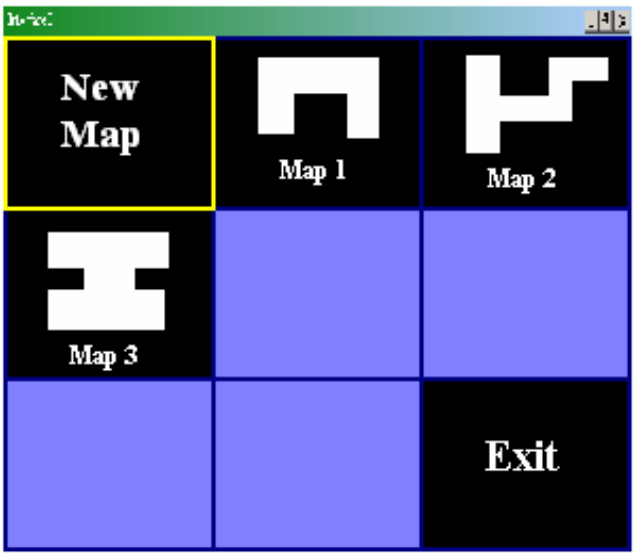

Figure 1. The graphical interface.

In figure 1 four different types of cells can be distinguished. They are as follows.

I. *New Map* cell allows new maps to be loaded into the interface.
II. *Exit* cell allows quitting the interface.
III. *Map #* cell, where # refers a specific map's number. This option is represented by an icon of the map loaded into the interface. If the interface is empty, then no maps are shown. The interface developed in this work can show up to seven different maps but this limitation is changeable.
IV. *Empty* cells are those cells that do not have yet a map associated with them.

Figure 1 shows the graphical interface that the user sees. The actual interface architecture is shown in figure 2.

In figure 2 one can see that the interface presented in this work is composed by three interfaces. Each interface's functionality is explained in the following sections.

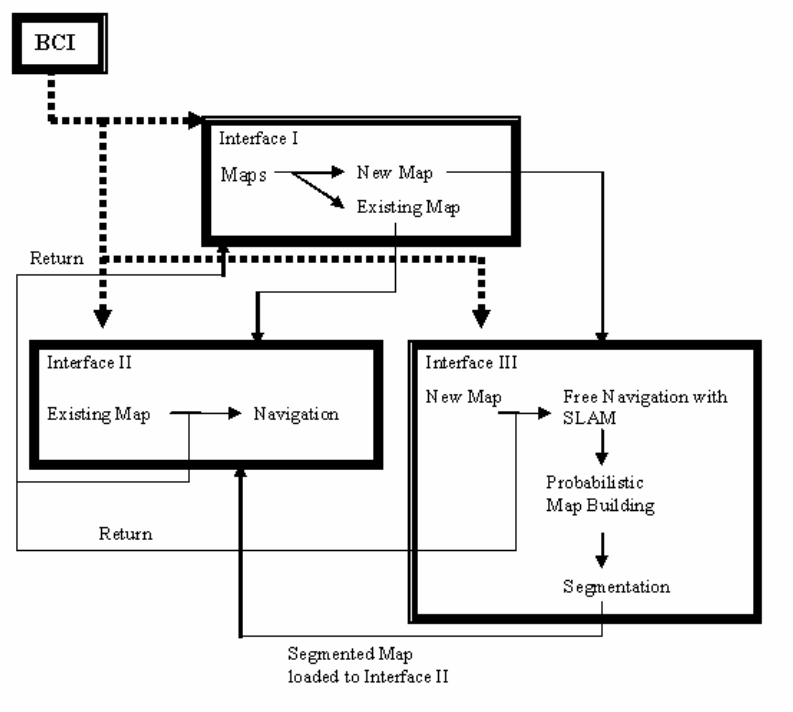

Figure 2. Complete System Architecture.

3. Architecture of Interface I

The *Interface I* or *Main Interface* is the one shown in figure 1. It is purely graphic and the user only has to choose between three options: the generation of a new map, the load of an existent map or quit the program. The first option is associated with the *New Map* cell; the second one is associated with any of the map icons at the interface. The third option corresponds to the *Exit* cell.

In order to choose a cell, a sequential scan mode is executed, highlighting cell by cell, starting from *New Map* cell and ending at the *Exit* cell. If no cell is chosen, the scan mode starts all over again. To choose a specific cell, an ERS event has to be detected by the BCI. More about ERS detection can be seen in [6].

If the user chooses to load, for example, *Map 3*, then the *Map 3* is shown in the screen and the control passes to *Interface II*. The *Interface II* manages the environment map chosen by the user and generates navigational trajectories.

On the other hand, if the user chooses to generate a new map of the environment where he/she is placed, then the *Interface III* is loaded into screen. The *Interface III* has a SLAM algorithm incorporated that allows it to build a map of the environment.

In both Interfaces, *II* and *III*, the option *Return* is available in order to allow returning to the *Main Interface*.

4. Architecture of Interface II

Figure 3 shows what user sees when *Interface II* is loaded. It represents a segmented map of an environment. The white cells in that map represent free obstacles cells, while black cells are those that are occupied. In this work each cell is 0.5×0.5 meters sized. This area is the same than the area of the mobile agent used (Mobile Robot Pioneer 2DX). The *Interface II* has a cell-by-cell sequential scan mode algorithm implemented. This algorithm is executed until an ERS event is detected by the BCI. The way this scan mode works is the same than the *Interface I* scan mode does. In *Interface II*, the scan mode is executed between all free cells (white cells) including the *Return* option. Thus, in a map with N white cells, the scan mode woks over $N+1$ options.

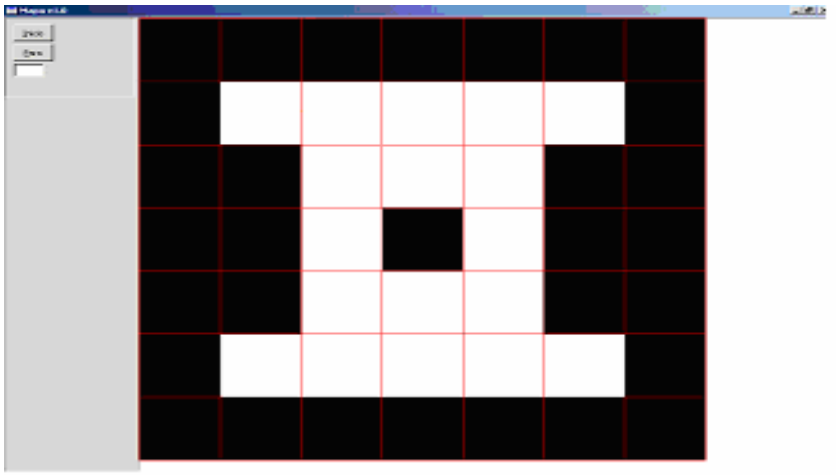

Figure 3. Segmented map of an environment.

Two processes were implemented in Interface II:

- Trajectory generation algorithm.
- Scan Mode system.

4.1. Trajectory generation for navigational purpose

Once a map is loaded into screen, the user has to determine his/her initial position inside that map. To do so, a first scan is released. When the BCI detects an ERS event generated by the user, the scan stops on a cell, which will become in the initial position of the mobile agent. Figure 4 shows this situation.

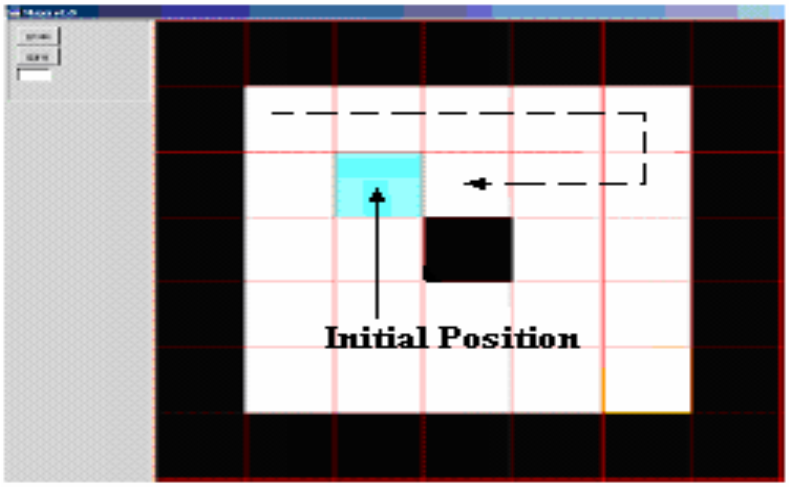

Figure 4. Initial Position selection.

Once the initial position is settled, the final destination should be chosen. This is done by the scan mode algorithm as it is shown in figure 5. Each navigational cell on a map has a weight associated with it. The way this weight is obtained is explained in the architecture of the *Interface III*.

With the initial position information among with the final position desired, a trajectory for a navigational purpose is generated. This generation is done by a DIJKSTRA ([8]) algorithm based on the white cells' weights. As the weight of a cell is associated to the center of mass of such a cell, then the trajectory is generated considering only the center of each cell. Figure 5 shows an example of a

trajectory generated by the DIJKSTRA algorithm. Additional information about the trajectory generation algorithm and navigation can be found in [9].

4.2. Scan Mode System

Interfaces I and *II* are based on an automatic scan mode system. This system allows scanning all available cells in both interfaces. The user only has to choose a specific cell and generate the ERS event.

The scan mode has a sampled time of 5 seconds. This value is needed in order to detect an ERS ([6]). To select a cell in both interfaces, the user has to hold an ERS event for at least two sampled times. If there is no ERS event, the scan mode continues normally highlighting the next cell.

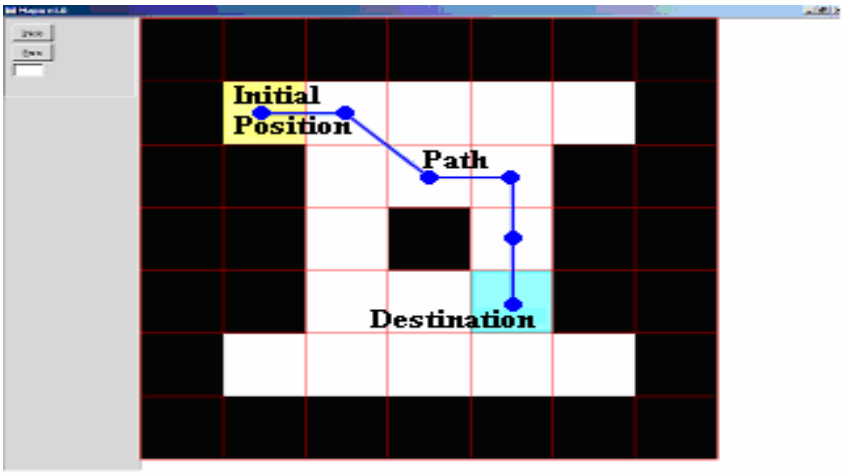

Figure 5. Trajectory Generation.

More information about the scan mode system can be seen in [9].

5. Architecture of Interface III

When *New Map* cell is chosen from *Interface I*, *Interface III* is loaded into screen. Its aspect can be seen in figure 6.

In *Interface III*, the user directly controls the mobile agent movements through the BCI. Figure 6 shows two windows of *Interface III*. One window shows the commands that are generated by the user while the other program, shows the real time movements executed by the mobile robot.

The *Interface III* allows the free navigation of the vehicle. This means that the user is the one who generates the movement controls for the robot. According to this, *Interface III* demands more of user's efforts.

While the free navigation mode is on, without any user's action, a SLAM algorithm based on an Extended Kalman Filter (EKF) is executed. The SLAM algorithm obtains a real time map of the environment. This map is a probabilistic occupancy grid map. Once the user leaves the *Interface III*, the probabilistic map is segmented. An example of the final map is shown in figure 5. The new segmented map is loaded into *Interface I* and a previously empty cell in that interface starts to show an icon of the map. If the user pretends to navigate inside a previously mapped environment, its map should be loaded from *Interface I*.

Interface III has three main functions associated with it: free navigation of the mobile agent, SLAM algorithm and segmentation of a map. These functions are explained as follows.

5.1. Free Navigation System

The *Interface III* does not work under a sequential mode scan. In this case, the user generates six basic movements' commands by an ERS/ERD combination. These six commands are as follows.

1. Move forward- the mobile robot moves forward with a constant velocity of 0.2 m/s.
2. Move backward- the mobile robot moves backward with a constant velocity of 0.2 m/s.
3. Turn to the left 90º and move forward.
4. Turn to the right 90º and move forward.
5. Stop movement.
6. Return.

The way the commands are generated by a combination of ERS/ERD events can be seen in [6]. This interface was tested on a population of 25 people with an average learning time of 15 minutes. The results of these testes are also shown in [6].

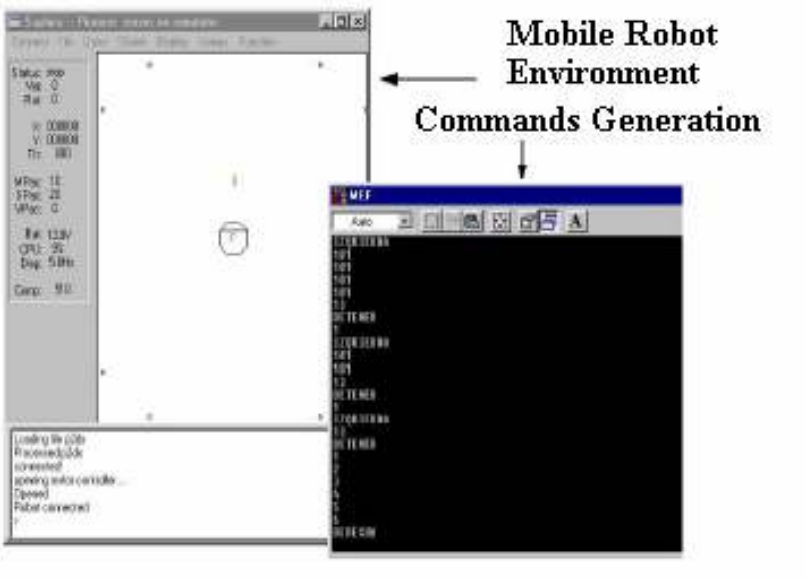

Figure 6. Screen view of Interface III.

During free navigation process is necessary to incorporate some control strategy to the mobile agent to protect its integrity. In order to do this, a behavioral strategy was developed. The main characteristics of this strategy are:

- It protects the mobile agent against wall collisions;
- It has an obstacle avoidance control law implemented;
- It does not allow invalid movements according to the environment's constrains.

More about the behavioral strategy can be found in [6]. To quit the *free navigation system*, the user has to choose the *return* option.

5.2. SLAM algorithm

The SLAM implemented on *Interface III* is an Extended Kalman Filter based algorithm that builds a grid map of the environment [7]. The probabilistic map is then built based on the grid map information. In order to do this, the mobile robot is equipped with 16 sonar sensors. To create the probabilistic map, Gaussian models of the sensors were used ([10]). Thus, the probabilistic information of each grid (cell) at the map is obtained by the recursive Bayes rule shown in equation (1).

$$P(C \mid s_n) = \frac{P(s_n \mid C)P(C \mid s_{n-1})}{P(s_n \mid C)P(C \mid s_{n-1}) + P(s_n \mid \overline{C})P(\overline{C} \mid s_{n-1})} \quad (1)$$

Where equation (1) is read as follows. $P(C \mid s_n)$ is the occupancy probability of cell C given the n-measurement sensor, $P(s_n \mid C)$ is the probability that the n-measurement shown by the sensor correspond to cell C. $P(s_n \mid \overline{C})$ is the probability that the n-measurement shown by the sensor correspond to every cell but C and $P(C \mid s_{n-1})$ is the probability of cell C given the s_{n-1} reading of that cell.

5.3. Map segmentation

Once the probabilistic map of the environment is obtained, it is segmented in cells of the size of the mobile agent. In this work, those cells were of 0.5×0.5 meters sized. Considering that, each cell in the grid map is of 0.1×0.1 meters ([9]), then each cell of the segmented map is composed of 25 cells of the probabilistic one. The occupancy of a cell at the segmented map is determined as follows.

- If a segmented map cell contains at least one probabilistic map cell with occupancy probability near one ($P(C) > 0.9$), then the segmented map cell has the status of *occupied*. Otherwise, the segmented map cell is *free*.
- The segmentation starts at the start mapping point, i.e., *(x,y) = (0,0)* at the global map.
- The set of all *free* cells form an area available for mobile agent navigation.

6. Experimental Results

This section shows the probabilistic map of a grid map and how it is converted into a segmented map. Figure 7 shows a grid map of a corridor.

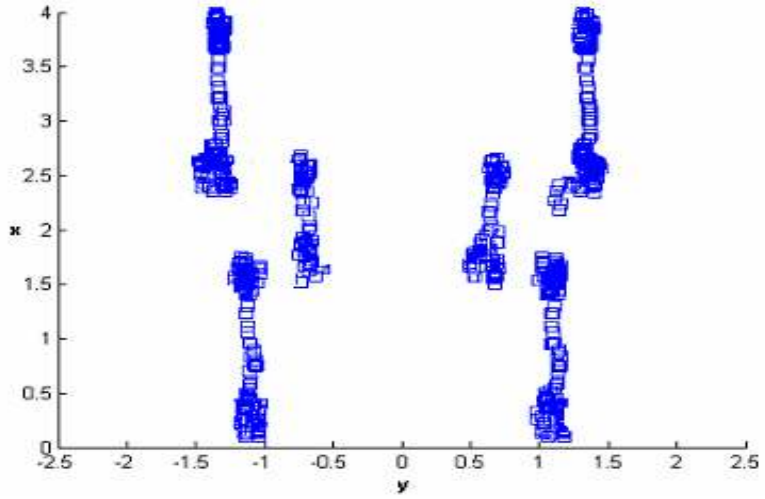

Figure 7. Grid map of a corridor.

Figure 8 shows the probabilistic map of the corridor showed by figure 7. Both maps were built in real time.

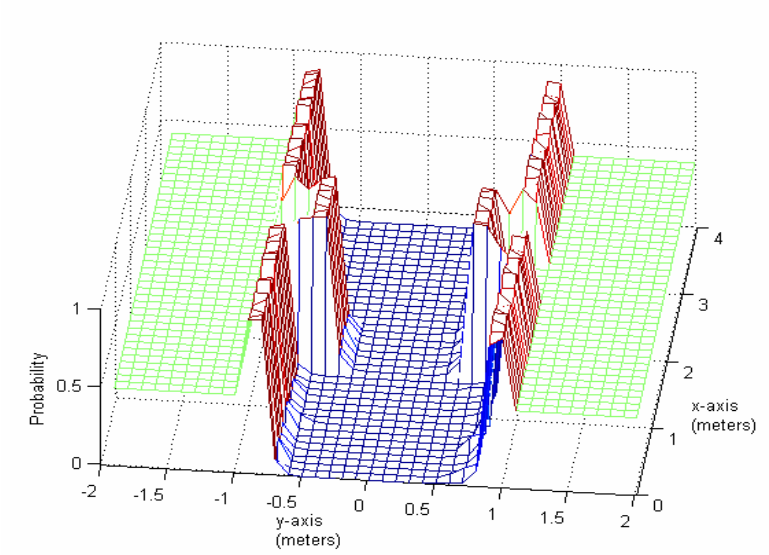

Figure 8. Probabilistic grid map of a corridor.

Figure 9 shows the segmented map corresponding to the one showed in figure 8.

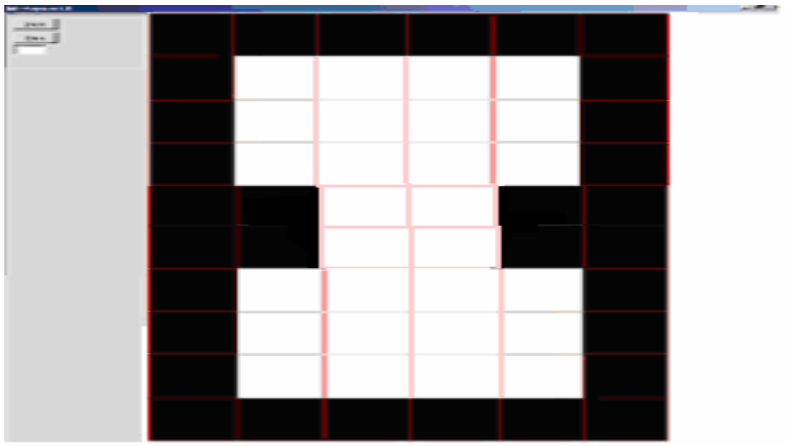

Figure 9. Final segmented map.

Figure 10 shows how *Interface I* -initially empty- is affected once a segmented map -*Map 1*- is loaded into it.

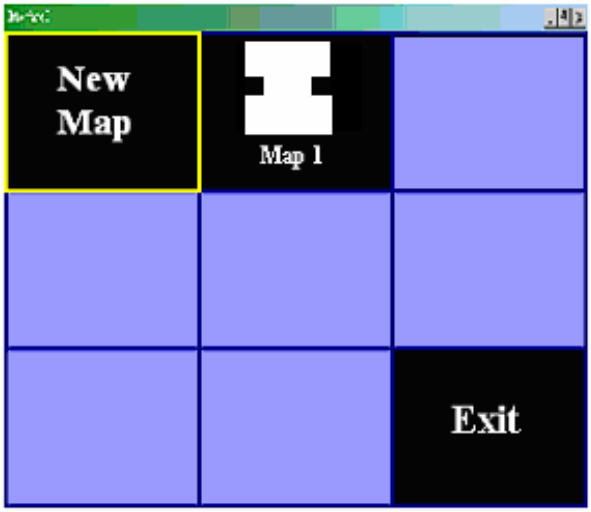

Figure 10. Main interface view once a new map is loaded into it.

7. Conclusions

In this paper, a graphical interface for maps managing was presented. This interface is governed by an ERS/ERD-based Brain Computer Interface although is not limited to this kind of BCIs. The Interface allows the user to control a mobile agent's movements (a mobile robot that can be interpreted as a wheelchair).

If the mobile agent is located inside a pre-mapped environment, the user is able to generate trajectories and navigate through that environment. Otherwise, if the map of the environment does not exist, then the user can obtain a map of it using an EKF-based SLAM algorithm. Once a map is complete, a segmented version of it is added to the main interface and it is available for future navigations.

Although the user can navigate inside a known environment or navigate an unknown environment in order to get a probabilistic map of it, the effort needed for both tasks is not the same.

As future work, an optimization algorithm is proposed in order to decrease the time needed to detect and classify ERS/ERD events.

References

[1] Kubler A, Kotchoubey B, Kaiser J, Wolpaw J R and Birbaumer N 2001 *Psychol Bul* **127** 358-375

[2] Wolpaw J R, Birbaumer N, McFarland D J, Pfurtscheller G, and Vaughan T M 2002 *Clin Neurophysiol* **113** 767-791

[3] Lehtonen J 2003 EEG-based brain computer interfaces Master's thesis, Helsinky University of Technology, Helsinky, Finlandia

[4] Felzel T 2001 On the possibility of developing a brain-computer interface (BCI) Technical University of Darmstadt, Darmstadt, Germany, Tech. Rep.

[5] Millán J, Renkens F, Mouri.ño J and Gerstne W 2003 Non-invasive brain-actuated control of a mobile robot *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico

[6] Auat Cheeín F A 2005 Diseño de una interfase cerebro-computadora para la navegación de un robot móvil Master's thesis, Facultad de Ingeniería de la Universidad Nacional de San Juan, San Juan, Argentina

[7] Zunino G and Christensen H I 2001 Simultaneous Localization and Mapping in Domestic Environments *International Conference on Multisensor Fusion and Integration for Intelligent Systems* 67-72

[8] Tenembaun A S 2003 *Computer Networks* (Prentice Hall)
[9] Neto F, Auat Cheeín F A, Cardoso Celeste W, Toniolo C C, Bastos Filho T F and Carelli R 2006 Navegação de um Veículo Móvel a Rodas em Ambientes Mapeados Utilizando Tabuleiro Eletrônico com Varredura Automática *IV IBERDISCAP: Congreso Ibero-Americano sobre Tecnologías de Apoyo a Portadores de Discapacidad* Vitória Espírito Santo Brasil
[10] Elfes A 1989 Using occupancy grids for Mobile Robot perception and navigation. *IEEE Computer Society* **22** 46-57