Map Building Using Fuzzy ART, and Learning to Navigate a Mobile Robot on an Unknown World

Rui Araújo, and Aníbal T. de Almeida
Institute for Systems and Robotics (ISR); and
Electrical Engineering Department; University of Coimbra;
Pólo II, Pinhal de Marrocos; 3030 Coimbra - Portugal; Email: rui@isr.uc.pt

Abstract
This paper introduces a new approach, based on the application of the Fuzzy ART neural architecture, for on-line map building from actual sensor data collected with a mobile robot. This method is then integrated, as a complement, on the parti-game learning approach, allowing the system to make a more efficient use of collected sensor information. Also, a predictive on-line trajectory filtering method, is introduced on the learning approach. Instead of having a mechanical device (the robot) moving to search the world, the idea is to have the system analysing trajectories in a predictive mode, by taking advantage of the improved world model. The real robot will only move to try trajectories that have been predicted to be successful, allowing lower exploration costs. This results on an overall new and powerful method for simultaneous and cooperative construction of a world model, and learning to navigate from an initial position to a goal region on an unknown world. It is assumed that the robot knows its own current world location. It is additionally assumed that the mobile robot is able to perform sensor-based obstacle detection (not avoidance), and straight-line motions. Results of experiments with a real Nomad 200 mobile robot will be presented, demonstrating the effectiveness of the proposed methods.

1 Introduction
It is important for an autonomous mobile robot to be able to navigate on unknown environments, where the location, shape and size of obstacles is unknown, and where there is no map or model of the world initially available. In fact, it is difficult to provide the robot control system with a global map model of its world. This may easily become a tedious and time consuming programming task. In addition, robot programming and control architectures must be equipped to face unstructured environments, which may be partially or totally unknown at programming time. A variety of approaches to motion planning, such as road-map, cell decomposition, and potential field methods (see [9] for an overview and further references) have been proposed. However, few methods have been able to fully cope with the above problem of model building and learning to navigate.

Both simulation [2], and real-robot experiments [1], [3], have demonstrated the parti-game multiresolution cell-based learning approach [10], as a powerful method for the simultaneous learning of a world model, and learning to navigate a mobile robot from a specified initial position to a known goal region on an unknown world. In [2], it is shown that the constructed world model is general-purpose, in the sense that its usefulness is not restricted to be used on self-learning a particular path, but is valuable for learning paths with different (Start,Goal) pairs. Some forgetting of accumulated world knowledge takes place, when cell-splitting occurs on the system [10], [3]. This leads to additional, time-consuming, exploration with the robot. In [1], modifications on the original parti-game cell-splitting strategy [10] were introduced to improve its operation by decreasing the exploration and modelling efforts. However, most of the received sensor information is lost, not being explicitly integrated into the constructed world model, and not used to plan robot trajectories.

In this paper we introduce, and demonstrate the effectiveness, of a new approach for sensor-based map building with geometric primitives, that is based on the application of the Fuzzy ART neural architecture [6]. This approach is then integrated, as a complement, in the original parti-game learning approach, resulting on a new method, allowing the system to make a more efficient use of collected sensory information for simultaneous and cooperative construction of a world model and learning to navigate from an initial location to a goal region on an unknown world. In this context, a predictive on-line trajectory filtering method is introduced on the learning approach, allowing a very significant reduction on the time-consuming exploration effort that is associated with searching the world with a real robot. Instead of having a mechanical device (the robot) searching the world, the idea is to have the system analysing trajectories in a predictive mode, by taking advantage of the improved world model. The real robot will only move to try a trajectory that has been predicted to be successful. The organisation of the paper is as follows. Section 2 summarises the basic learning architecture. Section 3 presents the new method for sensor-based map building, that in section 4 is integrated on the parti-game algorithm, yielding an improved approach to navigate a mobile robot. Section 5 presents experimental navigation results with a real Nomad 200 mobile robot. Finally in section 6 we make some concluding remarks.

2 Learning Architecture
In this section we briefly discuss the parti-game learning approach [10] that constitutes the original core of the method we use for learning to navigate a mobile robot. With the method, the robot can simultaneously learn a kind of map of its environment, and learn to navigate to the goal on an unknown world, having the predefined abilities of doing straight-line motion to a specified position in the world, and obstacle detection (not avoidance) using its own distance sensors. The system also requires the knowledge of the robot current position. However, in this paper we do not deeply address the problem of mobile robot localisation. We simply use accumulation of
encoder information to perform robot localisation. Even though this simple approach reduces errors, it was sufficient to validate the learning approach in our experiments. The two concurrent learning abilities may be seen as cooperating and enhancing each other in order to improve the overall system performance. For a more extensive discussion see [10], [5], [2].

The parti-game algorithm is based on a selective and iterative partitioning of the state-space. It is a multiresolution approach, beginning with a large partition, and then increasing resolution by subdividing the state-space (see Fig. 4) where the learner predicts that a higher resolution is needed. In order to reach the goal, the mobile robot path is planned to traverse a sequence of cells. The ability of straight-line motion is used as a greedy controller to move from one cell to the next cell on the path. This request to move to the next cell on the path (which is a neighbouring cell) may fail — usually due to an unexpected obstacle that is detected to be obstructing the robot path. A database of cell-outcomes, observed when the system aims at a new cell, is memorised and maintained in real-time. The information in this database may be seen as being organised on a graph data structure. In addition, cells are simultaneously organised in a kd-tree [7], for fast state-to-cell mapping. The database is in turn used to plan the sequence of cells to reach the goal cell, using a game-like minimax shortest path approach. In fact the next-cell outcomes observed as a result of a cell-aim (which is not guaranteed to succeed) may be viewed as “moves” available to an imaginary adversary that would be working against our objective of reaching the next cell, and ultimately reach the goal. The next cell on the path is chosen taking into account a worst case assumption. i.e. we imagine that for each cell we may aim, the adversary is able to place us on the worst position on the current cell such that the next cell that results from the aim is also the worst. In this way we always aim at the neighbouring cell with the best worst-outcome. For this purpose, a minimax problem is solved using Dynamic Programming methods [5]. Spatial resolution is robustly chosen using a game-theoretic cell-splitting criterion. Cells are split when the robot is caught on a losing cell - a cell for which the distance to the goal cell is ∞. Intuitively this means that for each sequence of cell-aims we may choose, our “adversary” may “respond” with cell outcomes that permanently prevent us from reaching the goal, i.e. for the current resolution, the game of arriving at the goal cell is lost. In those situations, as explained in [10], [3], [1], cells in the neighbourhood between losing and non-losing cells are split.

Algorithms 1, and 2 (Fig. 1) describe the overall parti-game learning method. In these figures, NEIGHS(i) represents the set of (cell-) neighbours of cell i. OUTCOMES(i, j) is the set of cells that were previously observed when the system was on cell i and aimed at cell j. P is the world partition, and D is a database of observed outcomes, represented as a set of triplets of the form: (starting-cell, aimed-cell, actually-attained-cell). Algorithm 1 (see Fig. 1) keeps applying the local greedy controller, aiming at the next cell, on the “minimax shortest path” to the goal, and accumulating observed cell-aim outcome-experience, until

**Algorithm 1**

```
REPEAT FOREVER
1. FOR each cell i and each neighbour j ∈ NEIGHS(i), compute the OUTCOMES(i, j) in the following way:
   1.1 IF there exists some k′ for which (i, j, k′) ∈ D THEN OUTCOMES(i, j) = {k′ (i, j, k) ∈ D }
   1.2 ELSE, use the optimistic assumption in the absence of experience: OUTCOMES(i, j) = {j}
2. Compute JWC(k′) for each cell using minimax.
3. Let i := the cell containing the current real-valued state s.
4. IF i = GOAL THEN exit, signalling SUCCESS.
5. IF JWC(i) = ∞ THEN exit, signalling FAILURE.
6. ELSE
   6.1 Let j := j′ ∈ NEIGHS(i) s ∈ OUTCOMES(i, j′) JWC(k)
   6.2 WHILE ( s is not in the goal cell )
      6.2.1 Actuate local greedy controller aiming at j.
      6.2.2 i := new real-valued state.
   6.3 Let is := the identifier of the cell containing s.
   6.4 D := D ∪ {i, j, is}.
LOOP
```

**Algorithm 2**

```
REPEAT FOREVER
WHILE ( s is not in the goal cell )
1. Run Algorithm 1 on s and P. Algorithm 1 returns the updated database D, the new real-valued state s, and the success/failure signal.
2. IF FAILURE was signalled THEN
   2.1 Let Q := All losing cells in P (JWC = ∞).
   2.2 Let Q′ := The members of Q who have any non-losing neighbours.
   2.3 Let Q″ := Q′ and all non-losing neighbours of Q′.
   2.4 Split each cell of Q″ in half along its longest axis producing a new set K of twice the cardinality.
   2.5 P := P + K.
   2.6 Recompute all new neighbour relations, and delete from the database D, those triplets that contain a member of Q″ as a start point, an aim-for, or an actual outcome.
LOOP
```

Figure 1: Algorithm 1, and top-level Algorithm 2.

either the robot is caught on a losing cell (JWC = ∞), or reaches the goal cell. The top-level Algorithm 2 (see Fig. 1), is responsible for subdividing a selected set of cells whenever the robot is caught on a losing cell.

3 Map Building

This section introduces the Fuzzy ART architecture [6] as a new approach for map building, based on geometric primitives. A map building algorithm should ideally have a set of characteristics. (1) The Fuzzy ART model allows self-organisation: to be autonomous, the mobile robot must organise in a useful way the sensor data it collects from the environment. (2) Multifunctionality: for representing the environment, we want a compact model that allows for efficient sensor-based map building, motion planning, self-referencing, etc. The application of the Fuzzy ART model for map-building is discussed in sections 3.1 and 5. An example of its usefulness and application in motion planning is shown in sections 4 and 5. Under the algorithmic point of view, if possible, we also want a similar model to be useful for other pattern recognition, control, and reasoning problems. (3) Updatability: the model should be easy to update according to new information arriving from sensors; the Fuzzy ART model can be updated by learning each isolated data point as it is received on-line, with the
same result as if the update were made in conjunction with a set of other data points - model update is made on a point by point basis not requiring the simultaneous consideration of a possibly large, set of data points. This is a significant convenience, allowing the robot to use new sensor data as soon as it arrives, thus enabling other system components, such as path planning and localisation, to take advantage of an updated model as soon as possible. See section 3 for a further discussion on the updatability of the Fuzzy ART model. The Fuzzy ART model enables (4) a compact geometric representation allowing small data requirements, and (5) low computational complexity. (6) Unlimited dimensions: the Fuzzy ART model is easy to extend to the modelling of data that is represented in higher dimensions (e.g. mapping of $\mathbb{R}^3$ objects) without adversely impacting on the data size or complexity.

Grid-based certainty maps are widely used to store and maintain occupancy information because they are easy to build and maintain [12]. However, it is difficult to select a resolution for the grid, that is suitable for representing, and to serve as a basis for reasoning on, the entire world. A very localised feature of the world may impose a very high (constant-) resolution grid over the entire state-space. This implies higher data requirements, and induces excessive detail on world modelling, on reasoning (higher computational costs), and on the paths that result from reasoning under such a model. The difficulties on the direct application of grid-based models on localisation have also been pointed out [11]. Geometric representations (e.g. [8], [11]), on the other hand, have been difficult to build, but are significantly more compact, less complex, and fully applicable to high- and low-level motion planning (e.g. this paper) and localisation approaches (e.g. [11]). With higher dimensions the geometric model data requirements become exponentially smaller than the requirements of constant-resolution cellular models.

### 3.1 Map Building with Fuzzy ART

Other works, e.g. [8], [11], have used different methods to extract geometric primitives. In this subsection, we give a compact overview of the Fuzzy ART learning architecture [6], and discuss its application to map building. With the approach we are able to extract a set of (hyper-) rectangles, whose union represents occupied space, where sensor data points associated with objects have been perceived - a kind of unsupervised clustering.

A Fuzzy ART system includes a field $F_0$ of nodes representing a current input vector; a field, $F_1$, that receives both bottom-up input from $F_0$ and top-down input from a field, $F_2$, that represents the active code or category (Fig. 2(a)). The $F_0$ activity vector is denoted by $I = (I_1, \ldots, I_M)$, with $I_i \in [0,1]$, $i = 1, \ldots, M$. The $F_1$ and $F_2$ activity vectors are respectively denoted by $y_1 = (y_{11}, \ldots, y_{1N})$ and $y_2 = (y_{21}, \ldots, y_{2N})$. The number of nodes in each field is arbitrary. Associated with each $F_0$ category node $j$ ($j = 1, \ldots, N$) is a vector $w_j = (w_{j1}, \ldots, w_{jM})$ of adaptive weights, or LTM traces. Initially weights are set to $w_{j1}(0) = \cdots = w_{jM}(0) = 1$, and all categories are said to be uncommitted. After a category is selected for coding it becomes committed. The Fuzzy ART operation is controlled by a choice parameter $\alpha > 0$, a learning rate parameter $\beta \in [0,1]$, and a vigilance parameter $\rho \in [0,1]$. For each presentation of input $I$, and $F_0$ node $j$, a choice function is defined by $T_j(I) = |I \land w_j|/(\alpha + |w_j|)$, where, for any $M$-dimensional vectors $p$ and $q$, $\land$ denotes the ‘min’ version of the fuzzy AND operator defined by $(p \land q)_i = \min(p_i, q_i)$, and $|I|$ denotes the norm defined by $|I| = \sum_{i=1}^{M} |I_i|$. For notational simplicity, $T_j(I)$ is often written as $T_j$ when input category $I$ is fixed.

The system is said to make a category choice when at most one $F_2$ node can become active at a given time. The category choice is indexed by $J$, where $T_j = \max(T_j : j = 1, \ldots, N)$. If more than one $T_j$ is maximal, the category $j$ with the smallest index is chosen. In particular, nodes become committed in order $j = 1, 2, 3, \ldots$. When the $j$th $F_2$ category is chosen, $y_{j1} = 1$, $y_{j2} = 0$ for $j \neq J$, and the $F_1$ activity vector is given by $y_1 = I \land w_J$. Resonance occurs if the match function $|I \land w_J|/|I|$ of the chosen category meets the following vigilance criterion: $|y_J| = |I \land w_J| \geq \rho |I|$. If so, then learning takes place as defined below. Mismatch reset occurs if $|y_J| = |I \land w_J| < \rho |I|$. In this situation, the match function $T_j$ is set to 0 for the duration of the current input presentation to avoid the persistent selection of the same category during search. A new index $J$ maximising the choice function is chosen, and this search process continues until the chosen $J$ leads to resonance. Once search ends, learning takes place by updating weight vector $J$ according to the following equation: $w_{J}^{new} = \beta (I \land w_{J}^{old}) + (1 - \beta) w_{J}^{old}$. By definition fast learning corresponds to setting $\beta = 1$. To avoid proliferation of $F_2$ categories, a complement coding input normalisation rule is used. With complement coding, if the input is an $M$-dimensional vector $x$ (in our case, a sensor data point), then field $F_0$ receives the $2M$-dimensional vector $\mathbf{I} = (x \ x^\prime) = (x_1, \ldots, x_M, x_1 \cdots, x^\prime_M)$, where the complement of $x$ is denoted by $x^\prime$, with $x_1 = 1 - x_1$. The weight vector, $w_j$, can also be written in complement coding form: $w_j = (w_{j1}, w_{j2})$, where $w_{j1}$ and $w_{j2}$ are $M$-dimensional vectors. Let a (hyper-) rectangle $R_j$ be defined by two of its corners (in diagonal) as illustrated in Figure 2(b). The size of $R_j$ is defined as $|R_j| = |v_j - u_j|$, which in the 2D case, is equal to the sum of the height and width. A Fuzzy ART system with complement coding fast learning, and con-

![Fuzzy ART neural architecture](image)

**Figure 2:** (a) Fuzzy ART neural architecture. (b) Rectangle associated to category $j$. 

[Diagram of Fuzzy ART neural architecture]
stant vigilance forms hyperrectangular categories, $R_j$ that grow monotonically in all dimensions, and converge to limits in response to an arbitrary sequence of input vectors [6]. Rectangle $R_j$ includes/represents the set of all data points which have activated Fuzzy ART category $j$ without reset [6]. Additionally [6], the maximum size of the rectangles $R_j$ can be controlled with the vigilance parameter: $|R_j| \leq (1 - \rho) M$. Recall that input data $I$ presented to Fuzzy ART must have components satisfying the condition $I_j \in [0, 1]$. However, if we have sensor data that is assumed to initially belong to one (any) axis-aligned hyperrectangle, then a linear transformation enables the satisfaction of this condition.

3.2 Sensor Data Filtering

Due to the accumulating nature of the Fuzzy ART system, when applying it to modelling real sensor data, it is useful to perform some prior filtering for removing noisy exemplars. In our implementation, we have used two filtering operations on sensor data points. First experience with the infrared range sensors we have used, shows that above a certain limit, distance readings were not very reliable, and thus were rejected. A second filtering operation, probably more generally applicable to other types of sensors, was performed. Let $S$ be a set of sensor points. A point $x$ is rejected, if no other data point is found inside a circle of radius $r_x$ and centre at $x$. Since this second operation requires the presence of a set of points $S$, we are not able to take full advantage of the isolated-point learning capability of Fuzzy ART. However, excellent results were obtained with small data sets, $S$, composed of points coming from a number of as low as two consecutive sensor-ring scans.

4 Predictive On-line Planning

In our previous work we demonstrated the application of the algorithm of section 2, to navigate a mobile robot [3], [1]. However, this work also enabled the identification and understanding of some aspects where this powerful method could be improved. Two comments emerge in this context. First, the parti-game model is based on its multiresolution partition, and on its aims-outcomes database ($D$) that summarises the information collected while exploring the world. The system gets rid of almost all sensor information received from the environment and, in spite of its clear usefulness, the information maintained on database $D$ is somewhat indirect, scarce, and implicit, in its description of the world. The second comment is that $D$ must be subject to some forgetting, when cell-splitting takes place (cf. point 2.6 of Algorithm 2 - Fig. 1). In fact when a set of cells is split, the associated aim-outcome information is not directly inheritable from "parent-cells” to "son-cells”. This constitutes one of the basic ideas of the parti-game operation: e.g. in spite of an aim-failure between two “parent-cells”, it is quite possible to have aim-success (and the method is searching for it) between two corresponding “son-cells”. But from an external point of view, this induces redundant exploration.

These two comments have motivated two developments on the navigation architecture. First, a new map building method, based on the Fuzzy ART model, and making better use of the received sensor information, was developed (section 3), and integrated in the parti-game system of section 2, for improving its world model. Second, the parti-game learning approach was extended by the introduction of a method for Predictive On-line Trajectory Filtering (POTF), allowing a very significant reduction on the time-consuming exploration effort that is associated with searching the world with a real robot. Instead of having a mechanical robot exploring the world, the idea is to have the system analysing trajectories in a predictive mode, by taking advantage of the improved world model. The real robot will only move to explore a planned trajectory, when the system is on real mode, and the system will enter this mode, only after a predictive success has occurred. In real mode, obstacle detection is performed using the real distance sensors of the robot. In predictive mode, on the other hand, exploration trajectories have an on-line predictive/simulation nature not involving any real-robot motion, and obstacle detection is performed using the Fuzzy ART world model, possibly enlarging the rectangles by a percentage of the robot radius in a border gap. In both modes, path planning is performed using the parti-game approach, with the parti-game model (partition, and aims-outcomes database $D$) being incrementally updated, according to the results of both predictive and real exploration. However, only in real mode is the Fuzzy ART model incrementally updated, because only in this mode is real sensor data available for this purpose.

As already described, one of the main ideas of the method, is to reduce real-robot exploration by giving priority to predictive exploration. However, the “extent” of the predictive effort may be controlled by configuring the exigency level of the “predictive success” condition that is used to trigger the transition from predictive mode to real mode. Two options may be used to establish this condition: a predictive success may be said to occur when, (1) $N$ consecutive predictive cell-aim successes (or the predictive arrival at the goal cell whichever comes first), or (2) a predictive arrival at the goal cell, take(s) place after starting from the current robot location. Also, the “frequency” of predictive effort may be controlled, by configuring the condition that is used to trigger the transition from real mode to predictive mode. The system always starts in predictive mode, and the following four options (listed in increasing order of predictive frequency) may be used: enter predictive mode (1) after cell splitting that takes place when the robot is caught on a losing cell, or (2) at the end of every failed cell-aim, or (3) at the end of every cell-aim, or (4) at the end of every motion sampling interval. Figure 3 illustrates the ideas introduced in this section.

Figure 3: Predictive on-line trajectory filtering.
5 Experimental Results and Discussion
The methods presented in this paper have been implemented on a zero turning radius real Nomad 200 mobile robot which includes 3 wheels, and 16 infrared range sensors (equally spaced around its body) that were used for obstacle detection.

In this section we present results of 3 experiments. Each experiment was organised as a sequence of trials to navigate in a world with obstacles, from a start location to a goal region. The first trial starts with no model of the world. Subsequent trials start with, and build upon, the world model that was learned until the end of the previous trial. The first experiment (Figs. 4(a)-(d)) is included for comparative purposes, and was discussed in [3]. In this experiment, the method of section 2 was used without the integration of the Fuzzy ART model (section 3), or the use of POTF (section 4). We do not deeply address the problem of mobile robot localisation. We simply used accumulation of encoder information to perform robot localisation, with localisation accumulators being set to correct values at the beginning of each trial. Even though this simple approach induces localisation errors, it was sufficient to demonstrate the mobile robot map-building and navigation methods of the previous sections. However, it should be interesting to improve the methods, in order to make them more robust to uncertainty in localisation. Figure 4 includes: infrared information (not in (g), (j)), robot trajectories (not in (h), (l)), state-space partition at the end of trials. Fuzzy ART model (not in (a)-(d)), and the predictive trajectories at the beginning of trials (only in (h), (l)). In all the experiments of figure 4, the dimensions of the state-space were 7.42m x 6.73m.

Experiment 2 was similar to experiment 1, except that: (1) the path to the goal was somewhat greater with the goal region set at 1.7 m to the left, of its location on experiment 1, and (2) the Fuzzy ART map building approach of section 3 was activated, and used to make predictive on-line trajectory filtering (POTF - section 4). In figures 4(e)-(g) the efficient trajectories of the first 3 trials of experiment 2 can be observed, with very direct navigation to the goal starting from the very first trial. These results demonstrate that the introduction of the new methods of sections 3 and 4, lead to a new, and very effective, approach for simultaneous model building, and learning to navigate a mobile robot on an unknown world. Additionally, from a relative point of view, when comparing with experi-
ment 1 (Figs. 4(a)-(d)), the improvements are very significant. In figures 4(e)-(g), it can also be observed how the Fuzzy ART approach of section 3, was able to create a geometric-primitive-based map with the location of objects as they were perceived by the infrared range sensors. In both experiments 2 and 3, sensor data filtering (section 3.2) was performed with sensor readings above 13 inches being rejected, and $r_f = 67 mm$. On trial 2, a few rectangles are present in places, that are slightly apart from where the infrared sensors have perceived objects on this trial. There are 2 reasons for this: (1) slight differences in localisation positions (e.g. from trial to trial) imply different locations for the perceived objects, and (2) the Fuzzy ART model has an accumulative nature, and currently is not able to detect and remove primitives from places where no objects are perceived anymore. In both experiments 2 and 3, the system was configured to enter predictive mode (section 4) at the end of each cell-aim, and to signal a predictive success after a predictive arrival at the goal cell. Also, a border gap of 80% of the robot radius was used on the Fuzzy ART rectangles. Figure 4(h) illustrates the predictive trajectories that were analysed at the beginning of trial 2: even before any real robot motion. Note that, since the robot enters predictive mode at the end of cell-aims, this is just a part of the predictive trajectories analysed during trial 2. As can be seen, a considerable amount of predictive exploration takes place. This enables a significant decrease of the exploration effort that would have to be done with the real mechanical robot (a more time consuming operation). Also, the lower amounts exploration motion enabled by the introduction of POTF, lead to a reduction in the severity of the localisation error problem without, however, solving it. In fact it may be seen that, when comparing with trials 1 and 2 of Experiment 1 (Figs. 4(a), (b)), the localisation errors have clearly decreased in Experiments 2 and 3.

Experiment 3 (Figs. 4(i)-(l)), used the same navigation controller that was used in experiment 2, and is an additional experimental evidence of the effectiveness of the overall navigation method that was introduced in this paper. Note how the system is able to backtrack from the dead end at the upper-left corner of the world, and subsequently this area does not need to be visited by the real robot anymore. An effective use of the available sensor information is made, which leads to efficient navigation trajectories from the very first trial (see Figs. 4(i)-(k)). The predictive trajectories at the beginning of trial 2 are again shown in figure 4(l). Further discussion of the results of experiment 3 would be very similar to the discussion of experiment 2. Finally, we remark that, in all experiments, the time expended in computational costs is only a small fraction of the total operation time that includes the sampling intervals when the mechanical robot was moving. A quantitative evaluation of the effectiveness of the introduction of POTF will be presented in [4].

Future work on Fuzzy ART map building includes relevant aspects of model updatability which, at present state, have not yet been considered: the division, and pruning (an obstacle may be no longer present), of geometric primitives in order to better model the world data. However, from a structural point of view, it will not be difficult to delete geometric primitives. Thus, we are optimistic on the feasibility of overcoming the above two aspects (especially pruning), provided that suitable tests are integrated to detect the two situations. This optimism is further supported by the fact that the Fuzzy ART world model may be used to make (sensor) measurement predictions, and those predictions may be compared with real sensor readings. This would be especially useful in non-static worlds. See section 3 for the discussion on another aspect of updatability, where Fuzzy ART is clearly strong. Exploring the application of the Fuzzy ART world model for place recognition and localization is another line of future research.

6 Conclusion

A new approach, based on the Fuzzy ART neural architecture, has been introduced for on-line map building from actual sensor data. This method was then integrated, as a complement, on the part-game learning approach, allowing the system to make a more efficient use of sensor information. Also, a predictive on-line trajectory filtering method, was introduced on the learning approach. This resulted on an overall new and powerful method for simultaneous/interactive construction of a world model, and learning to navigate from an initial position to a goal region on an unknown world. Results of experiments demonstrated the application of the learning approach to a real mobile robot.

References