Building Geometric Certainty Maps and Application to the Navigation of a Mobile Robot

Rui Araujo, and Anibal T. de Almeida

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

Abstract— Autonomous robots must be able to learn and maintain models of their environments. This paper introduces a new approach, based on 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 robot moving to search the work, 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, and 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.

I. 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.

Applications where there is no previously available world model, usually rely on sensor systems, such as laser range finders, and infrared and sonar range sensors, to collect information composed of thousands of range points which are difficult to handle and interpret in order to construct a model of the environment. Grid-based certainty maps are widely used (e.g. [11]) to store and maintain occupancy information because they are easy to build and maintain. They were introduced by Moravec and Elfes [8], and further enhanced and modified by other researchers (e.g. [3]). Grid-based certainty maps represent the environment as a two-dimensional array of cells, with each cell holding a value which represents the confidence in whether it is occupied space or free space. 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 [10]. Geometric representations (e.g. [6], [10]), 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. Section III) and localisation approaches (e.g. [10]). With higher dimensions the geometric model data requirements become exponentially smaller than the requirements of constant-resolution cellular models.

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 Fuzzy ART neural architecture [5], [4]. The system generates a model composed of rectangular geometrical primitives. The model is learned on-line from perceived range data points, with familiar inputs being directly associated to their rectangular categories, while novel exemplars continue to trigger the generation of new categories.

The proposed mapping building method is then integrated, as a complement, into the parti-game learning approach [7], [2], [1], resulting on an improved 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, leading to a 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 article is organised as follows. Section II presents the new method for sensor-based map building, that in section III is integrated on the parti-game approach, yielding an improved method to navigate a mobile robot. Section IV presents experimental navigation results with a real Nomad 200 mobile robot. Finally in section V we make some concluding remarks.
II. MAP BUILDING

This section introduces the Fuzzy ART architecture [5] as a new approach for map building, based on geometric primitives. In general a map building algorithm should ideally have a set of characteristics [6].

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 II-A and IV, and an example of its usefulness and application in motion planning is shown in Sections III and IV. The Fuzzy ART model performs a clustering operation that has potential application on other pattern recognition, control, and reasoning problems involved in the operation of a mobile robot. (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 without 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 IV 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 $R^3$ objects) without adversely impacting on the data size or complexity.

A. Map Building with Fuzzy ART

This subsection gives an overview of the Fuzzy ART architecture [5], [4], and discusses its application as a novel approach for map building, based on geometric primitives. Other works, have used different methods to extract geometric primitives. For example in [6] occupied space is represented by ellipsoids, and in [10], the world model is composed of lines and circles. With the approach described in this subsection, 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, $F_j$, that receives both bottom-up input from $F_0$, and top-down input from a field, $F_j$, that represents the active code, or category (Fig. 1(a)). The $F_j$ activity vector receives the current sensor data point, and is denoted by $I = (i_1, \ldots, i_M)$. Each component is assumed to satisfy the condition $I_i \in [0,1]$. However, if we have sensor data that is assumed to belong to one (any) axis-aligned hyperrectangle, then the application of a linear transformation, $L$, enables the satisfaction of this condition. The $F_1$ and $F_j$ activity vectors are respectively denoted by $y_1 = (y_{11}, \ldots, y_{1M})$ and $y_j = (y_{j1}, \ldots, y_{jM})$.

The number of nodes in each field is arbitrary. Associated with each $F_j$ category node $j$ ($j = 1, \ldots, N$) is a vector $w_j = (w_{j1}, \ldots, w_{jM})$ of adaptive weights. Initially weights are set to $w_j(0) = \cdots = w_{jN}(0) = 1$, 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 $F_j$ node $j$, a choice function is defined by $T_j(I) = ||I \land w_j||/(\alpha + ||w_j||)$, where $\land$ denotes the 'min' version of the fuzzy AND operator defined by $p \land \rho = \min(p, \rho)$, and $||I||$ denotes the norm defined by $||p|| = \sum_{i=1}^{M} |p_i|$. 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 $F_j$ node can become active at a given time. The category choice is indexed by $J$. If $T_j = \max\{T_j : j = 1, \ldots, N\}$. 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 $F_j$ category is chosen, $y_{j1} = 1$, $y_{j2} = 0$ for $j \neq J$, and the $F_j$ activity vector is given by $y_j = I \land w_j$. 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 process continues until the chosen $J$ leads to resonance. Once search ends, learning takes place by updating weight vector $w_j$ 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 $F_j$ 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 $I = (x, x') = (a_1, \ldots, a_M, a'_1, \ldots, a'_M)$, where the complement of $x$ is denoted by $x'$, with $a'_i = 1 - a_i$. The weight vector, $w_j$, can also be written in complement coding form: $w_j = (u_j, v_j)$, where $u_j$, and $v_j$ are $M$-dimensional vectors. Let an (hyper-) rectangle $R_j$ be defined by two of its corners (in diagonal) as illustrated in Fig. 1(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 constant vigilance forms hyperrectangle categories. $R_j$ that grow monotonically in all dimensions, and converge to limits in response to an arbitrary sequence of input vectors [5], [4]. Rectangle, $R_j$, includes/represents the set of all data points which have
activated Fuzzy ART category \( j \) without reset [5, 4]. Additionally [5, 4], the maximum size of the rectangles \( R_j \) can be controlled with the vigilance parameter: \( |R_j| \leq (1 - \rho) M \). In our case, rectangle size limitation is important in order to avoid the free-space between two obstacles to be modelled by a single encompassing rectangle primitive. In this case, instead of growing a single rectangle, we want the size limitation mechanism to give rise to two separate primitives with free-space in between. After applying to the rectangles \( R_j \), the inverse, \( L^{-1} \), of the linear transformation \( L \) we get a new set of rectangles \( R_j^W \) that include/represent all the original sensor data points.

**B. 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 point observations. 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 \( \mathbf{x} \) is rejected, if no other data point \( \mathbf{x}^r \in S \) is found inside a circle of radius \( r_f \) and centre at \( \mathbf{x} \) (see Fig. 1(c)). 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.

**C. Sensor Measurement Prediction**

In this subsection we will discuss the application of the Fuzzy ART world model for making predictions of sensor distance readings. These predictions can be integrated with real sensor data in order to perform place recognition and localisation.

The Fuzzy ART world model is composed of a set of rectangular geometric primitives \( \mathcal{P} = \{ P_i : i = 1, \ldots, N_p; N_p \leq N \} \), where \( P_i \) is a vector containing the coordinates of the lower-left and upper-right corners of rectangle \( i \): \( P_i = (x_{il}, y_{il}, x_{iu}, y_{iu}) \). \( N_p \) is the number of primitives in the current model. Clearly, \( N_p \) is not higher than the number of \( F_2 \) categories: \( N_p \leq N \).

Robot range sensors, like for example a laser range finder, or infrared or sonar range sensors, continuously provide the system with distance readings. Each of these readings provides a measurement vector, \( \mathbf{q}_D^S \), relative to the corresponding sensor frame, \( \{ S \} \), and which in general may be expressed as \( \mathbf{q}_D^S = [d_D^S \ \theta_D^S]^T \), where \( d_D^S \) is a range, and \( \theta_D^S \) is an angle, to a measured point \( P \) (see Fig. 2). For infrared and sonar range sensors we usually have \( \theta_D^S = 0 \). The position of point \( P \) relative to the sensor frame, \( \{ S \} \), may be expressed by the following position vector:

\[
\mathbf{x}_P^S = \begin{bmatrix} x_P^S & y_P^S \end{bmatrix}^T = d_P^S \cos \theta_P^S \sin \theta_P^S \begin{bmatrix} x_a^S & y_a^S \end{bmatrix}^T \quad (1)
\]

To perform the sensor predictions it will be assumed that the pose vector of the sensor frame, \( \{ S \} \), relative to the robot frame, \( \{ R \} \), is known and given by \( \mathbf{X}_S^R = \begin{bmatrix} (\mathbf{x}_S^R)^T & \theta_S^R \end{bmatrix}^T = \begin{bmatrix} x_S^R & y_S^R & \theta_S^R \end{bmatrix}^T \), and the pose vector

\[
\mathbf{X}_S^R = \begin{bmatrix} (\mathbf{x}_S^R)^T & \theta_S^R \end{bmatrix}^T = \begin{bmatrix} x_S^R & y_S^R & \theta_S^R \end{bmatrix}^T
\]
of \( \{ R \} \), relative to the world frame, \( \{ W \} \), is known and given by
\[
X^W_R = [x^W_R \ y^W_R \ \theta^W_R] = [x^W_p \ y^W_p \ \theta^W_p]T.
\]

The problem to be solved may be stated as follows: given \( \mathbf{P} \), \( X^W_i, X^W_{i+1}, \) and \( \theta^W_{i+1} \), then calculate a prediction of the measurement range \( d^p_i \). To each geometric primitive \( \mathbf{p}_i \) in \( \mathbf{P} \), it corresponds one predicted measurement point \( P_i \). Thus it is clear that the measurement prediction \( d^p_i \) corresponds to the predicted point that is closest to the origin of the sensor frame. (S). Thus each primitive may be analysed separately and then the smaller predicted distance, \( d^p_i \), is retained. For each primitive, one of two cases may occur: (1) the predicted point belongs to a vertical segment (see Fig. 2), or (2) to an horizontal segment of the rectangular geometric primitive. In the first case, if the predicted distance is given by \( d = w \tau_{w_0} \) (where \( w \tau_{w_0} = \tau_i \) or \( w \tau_{w_0} = \tau_o \)), then \( d^p \) may be obtained as follows (see Fig. 2):

\[
d^p = \frac{w \tau_{w_0} - x^W_{w_0}}{\cos \theta^W_{w_0}}.
\]

Likewise, in the second case, if the horizontal segment is given by \( d = w \tau_{y_0} \) where \( w \tau_{y_0} = \tau_i \) or \( w \tau_{y_0} = \tau_o \), then:

\[
d^p = \frac{w \tau_{y_0} - y^W_{y_0}}{\sin \theta^W_{y_0}}.
\]

and

\[
\begin{align*}
x^W_{w_0} &= x^W_w + x^W_{w_0} \cos \theta^W_{w_0} - y^W_{w_0} \sin \theta^W_{w_0}, \\
y^W_{y_0} &= y^W_y + x^W_{y_0} \sin \theta^W_{y_0} + y^W_{y_0} \cos \theta^W_{y_0}.
\end{align*}
\]

If at sampling interval \( k \), the real and the predicted sensor range readings are respectively given by \( d^p(k) \) and \( d^p(k) \), then an error variable, \( e^p(k) = d^p(k) - d^p(k) \), may be formed. We may define a, so far observed, mean value for this error variable, \( m_e(k) = \left( \frac{1}{k} \sum_{j=1}^{k} e^p(j) \right) / k \). This mean error can be added to the predicted distance, \( d^p(k) \), to obtain a new corrected prediction, \( d^p(k) = d^p(k) - m_e(k) \). This new estimate of sensor measurement will have a mean closer to zero.

III. Navigation Architecture

In this section we discuss the navigation architecture into which the map building method of section II was integrated. The parti-game multiresolution cell-based learning approach [7] constitutes the original core of the method that we use for learning to navigate the mobile robot. With the method, the robot can simultaneously, learn a kind of map of its environment, and learn to navigate from an initial location to a known goal region 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. It is also assumed that the robot knows its own current world coordinates. For a more extensive discussion see [7], [2], [1]. The parti-game algorithm is based on a selective and iterative partitioning (P) of the state-space (e.g. Fig. 5). The path to the goal cell is planned as 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 neighboring cell) may fail – usually due to an unexpected obstacle that is detected to be obstructing the robot path (a stopping mechanism is triggered). A database, \( D \), of cell-outcomes, observed when the system aims at a new cell, is memorised in real-time. 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. Cells are split when the robot is caught on a losing cell - a cell for which the distance to the goal cell is \( \infty \) (this means that, for the current resolution, the game of arriving at the goal cell is lost). In those situations, as explained in [7], [2], [1], the partition resolution is selectively increased by splitting cells in the neighbourhood between losing and non-losing cells.

A. Predictive On-line Trajectory Filtering

In our previous work we demonstrated the application of the parti-game algorithm to navigate a mobile robot [2], [1]. However, this work also enabled the identification and understanding of some aspects where this method could be improved. Two comments emerge in this context. First, in spite of its clear usefulness, the information maintained on the parti-game model (\( P \) and \( D \)) is somewhat indirect, scarce, and implicit, in its description of the world because most of the received sensor information is lost, not being explicitly integrated into the constructed world model, and nor used to plan robot trajectories. The second comment is that the database of cell outcomes, \( D \), is required to undergo some forgetting when cell-splitting takes place on the system [7], [2]. But from an external point of view, this induces redundant exploration on the operation of the method.

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 II), and integrated in the parti-game system, 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. Specifically, two distinct modes of operation of parti-game are introduced: real mode and predictive mode (Fig. 3). Instead of having a mechanical robot exploring the world, the idea is to have the system analysing trajectories in 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 in 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 (P, and D) being incrementally updated, according to the results of both predictive and real exploration. However, only in real mode is the Fuzzy ART model 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, takes 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. Fig. 3 illustrates the ideas introduced in this section.

IV. EXPERIMENTAL RESULTS

The methods presented in this paper have been implemented on a zero turning radius circular shaped real Nomad 200 mobile robot [9] (Fig. 4), which has a diameter of 60 cm and includes 3 wheels, and 16 infrared range sensors (equally spaced around its body) that were used for obstacle detection, and to measure the distance to objects less 60 cm away in the environment. The infrared range sensors have a resolution of 25.4 mm. The results reported in this article were obtained using a 100 MHz Pentium processor running the “Nomadic Software Environment” [9]. This software environment allows the control of a real or simulated robot. The translational and rotational speeds were 15 cm/sec and 20 deg/sec.

In this section we present results of two real-robot navigation experiments. Each experiment was organised as a sequence of trials to navigate, in a world with obstacles, from a starting 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 robot must know its current location in the world. We have simply used accumulation of encoder information to perform robot localisation, with localisation accumulators being set to correct values at the beginning of each trial. This induces localisation errors (e.g. Fig. 5(b)), but was sufficient to demonstrate the 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. Figs. 5 and 6 include: infrared information, robot trajectories (not in 6(c)), state-space partition. Fuzzy ART model (not in 5(a)-(d)), and the predictive trajectories at the beginning of trials (only in 6(c)).

Experiment 1 (Figs. 5(a)-(d)) is included for comparative purposes, and was discussed in [2]. In this experiment, the method of Section III was used without the integration of the Fuzzy ART model (Section II), or the use of POTF method (Section III-A). In experiment 2 (Figs. 6(a)-(c)) the Fuzzy ART map building approach of section II was activated, and used to make predictive on-line trajectory filtering (POTF - Section III-A). In Figs. 6(a)-(b) the trajectories and the constructed geometric-primitive-based Fuzzy ART map (with the location of objects as they were perceived by the infrared range sensors) of trials 1 and 2 can be observed. These results demonstrate that the introduction of the new methods have significantly improved the navigation approach, with very direct navigation to the goal starting from trial 1. Fig. 6(c) illustrates the subset of predictive trajectories that were analysed at the beginning of trial 2, even before any real robot motion. The system was configured to enter predictive mode at the end of every cell aim, and to signal a predictive success at a predictive arrival at the goal cell. Also, a border gap of 80% of the robot radius was used on the Fuzzy ART rectangles. 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). 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 (Fig. 6).

Simulation experiments were conducted regarding the application of the sensor measurement prediction method of Section II-C. Typical values obtained for the absolute value of the mean of the prediction error, before and after correction (Section II-C), were 6 cm and 0.3 cm respectively.

V. Conclusion

This article described a new approach, based on the Fuzzy ART neural architecture, for sensor-based online map building. 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. To achieve this, a predictive online trajectory filtering method was introduced on the learning approach, resulting on an improved method to navigate a mobile robot from an initial location to a goal region on an unknown world. Results of experiments demonstrated the application of the described map-building method and the navigation methods on real mobile robot.

References


