Modular Neural Network and Classical Reinforcement Learning for Autonomous Robot Navigation: Inhibiting Undesirable Behaviors

Eric A. Antonelo, Albert-Jan Baerveldt, Thorsteinn Rögnvaldsson, and Mauricio Figueiredo

Abstract— **Classical reinforcement learning mechanisms and a modular neural network are unified to conceive an intelligent autonomous system for mobile robot navigation. The conception aims at inhibiting two common navigation deficiencies: generation of unsuitable cyclic trajectories and ineffectiveness in risky configurations. Different design apparatuses are considered to compose a system to tackle with these navigation difficulties, for instance: 1) neuron parameter to simultaneously memorize neuron activities and function as a learning factor, 2) reinforcement learning mechanisms to adjust neuron parameters (not only synapse weights), and 3) a inner-triggered reinforcement. Simulation results show that the proposed system circumvents difficulties caused by specific environment configurations, improving the relation between collisions and captures.**

I. Introduction

Robot navigation has caused frequent admiration in human beings, certainly because of the intriguing capabilities announced (and sometimes confirmed). Besides, it is easily associated to a prosperous and intense technical progress. A vast range of applications are foreseen, e.g., surveillance, rescue, and space exploration.

Despite the great appeal, it is not easy to conceive a navigation system, being considered one of the main challenges in artificial system research. Different approaches are adopted for designing navigation systems, each of which more suitable to some classes of environments and tasks.

Traditional control techniques are not suitable for designing such systems because of the huge difficulty for modeling physically the problem. The unstructured character of the environment (including the simplest one) and the nonholonomic character of common robots are among the critical points to overcome.

On the other hand, navigation systems based on computational intelligence techniques (neural networks [1], fuzzy systems, evolutionary computation [2] and swarm intelligence [3]) have reached important results. These systems show desirable characteristics, e.g., robustness and; learning and adaptation capacities.

Systems can be divided in two main classes: deliberative and reactive. Deliberative systems take decisions considering all information captured (or stored), including past information. In general such systems consider internal maps of the environment and the trajectory is planned completely a priori. Reactive systems take decisions based only on sensory information captured currently.

A special feature, the learning capability, allows identifying a distinct and important class, namely, autonomous navigation systems (considered here as those systems that learn their navigation strategies independently from the designer). These systems are of special concern to cope with inherent difficulties associated to unknown environments, e.g., unpredictability and hazardousness. The design of systems for navigating in unknown environments is even more difficult, being associated with the most interesting applications.

Most proposed intelligent autonomous systems are reactive systems [4]. Different types of navigation problems are considered. Usually autonomous systems learn two behaviors: obstacle avoidance and target seeking. Sometimes there is a specific sensory field for detecting targets [5] [6] [8]. If this is not the case, the design is more complex. In general, it is observed that systems generate undesirable behaviors, e.g., cyclic trajectories, if attractive objects are eclipsed [7].

This is the case considered in this paper: a reactive autonomous navigation system for mobile robots without a target sensory field. The main purpose of this work is to describe learning mechanisms that acquire navigation strategies that avoid undesirable behaviors. The system is composed of hierarchical modular neural networks that learn according to the classical reinforcement learning procedure [9]. Learning proceeds continuously (as the robot interacts with different classes of objects) and provides the base for acquisition of target seeking and obstacle avoidance behaviors. Despite the lack of a target sensory field, simulation results show that the autonomous system learns to generate behaviors free of cyclic trajectories and the robot efficiently avoids risky configurations.

The paper is organized as follows. Existent deficiencies of certain autonomous systems are presented in Section II. Environment and robot models are described in Section III. The following section gives an overview about the autonomous system. In Section V the proposed enhancements for the system are described. Section VI

This work is supported by the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America, scholarship no. E04M027421BR.

E. Antonelo, A.-J. Baerveldt and T. Rögnvaldsson are with the School of Information Science, Computer and Electrical Engineering, Halmstad University, Sweden (eric.antonelo@gmail.com, albert@ide.hh.se and denni@ide.hh.se, respectively).

M. Figueiredo is with Department of Computer Science, State University of Maringá, Brazil (mauricio@din.uem.br).

shows simulation results that confirm the effectiveness of the proposals. Finally some discussion points come in Section VII.

II. Common Navigation Deficiencies

The environment configuration influences the performance of navigation systems, in particular autonomous navigation systems. Navigation system deficiencies may be observed when the robot navigates through specific environments. Two kinds of navigation deficiencies are considered in this section: cyclic trajectories and inefficiencies in risky configurations.

From this point on, including next sections, some figures show a global view of the navigation environment after a simulation. The key for understanding them is: a green semicircle and a black line represent the robot and its trajectory; and blue and yellow polygons represent repulsive and attractive objects, respectively. Repulsive contacts (collisions) are marked with small green circles and attractive contacts with red ones.

Undesirable Cyclic Trajectories: Navigating in rich environments (e.g., an environment with a diverse configuration and several objects inside), an autonomous system is able to learn suitable navigation strategies (avoid repulsive objects and approach attractive objects). However, if the environment is *not rich*, the system may develop strange and undesirable behaviors, trapping the robot in an endless cyclic trajectory [8].

This is briefly illustrated below. We have showed that the system is able to learn acceptable navigation strategies [7]. However, a deeper analysis has revealed that the robot ends up in cyclic trajectories when moving/learning in certain environments (Fig. 1). This can be observed after a long series of experiments (38 trials, each lasting for 30000 iterations). Results also show that the robot is able to capture both attractive objects only once [13].

It is important to point out that certain aspects of the system and environment favor the occurrence of this kind of undesirable behavior: the attractive objects may be eclipsed depending on the robot position, the system is of a reactive type and the robot model does not have a target sensory field.

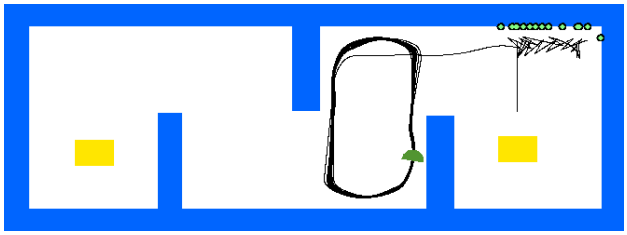

Fig. 1: Autonomous navigation system generates an endless cyclic trajectory in a simple environment without obstacles inside.

Inefficiency in Risky Configurations: It is not uncommon that the autonomous system proposed in [7] generates ineffective trajectories that cause collisions when the robot navigates in a corner-like region, even after it has acquired an efficient collision avoidance strategy (considering distinct configurations) (Fig. 2). In general, recurrent direction adjustments move the robot closer and closer to the repulsive objects and, at the end, a fatal collision occurs [6]. Poor performance is also observed if the robot is located in an irregular corridor-like region (e.g. a corridor composed of a wide wall and a small object) (Fig. 2). A similar recurrence of events takes place also in the latter case. Particularly, irregular corridor configurations may bring about collisions problems due to an excessive repulsion behavior, e.g., to the extended wall.

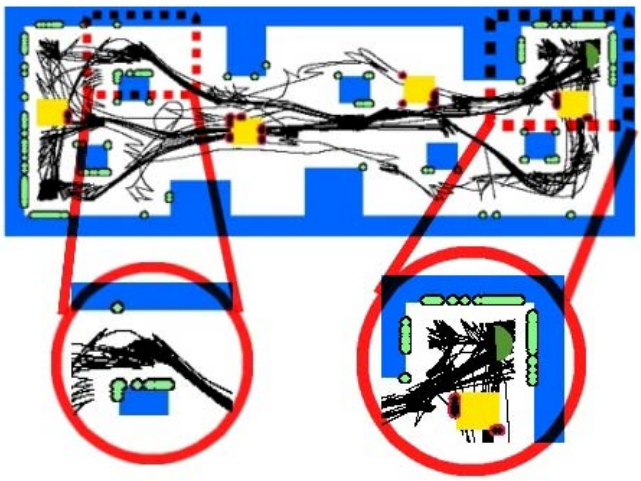

Fig. 2: Two common risky configurations (dotted line cuts): irregular corridor (left), and corner (right).

Besides the damage caused by collisions, the aforementioned undesirable behavior brings another negative consequence: the robot may fail to explore the environment and reach its goal, which is to capture targets.

Both these deficiencies, undesirable cyclic trajectories and inefficiency in risky configurations, are the focus of this paper. The next sections describe learning mechanisms that improve the performance of the autonomous system proposed in [7].

III. Environment and Robot Models

The environment of the robot is composed of repulsive and attractive objects. Each object has a particular color, denoting its respective class. Obstacles are considered repulsive objects.

The robot model is shown in Fig. 3. The robot interacts with the environment by distance, color and contact sensors; and by one actuator that controls the adjustment on the movement direction. Sensor positions are distributed homogenously over the front of the robot (from -90º to +90º). Each position holds three sensors (for distance, color and contact perception). Further details about robot sensors are given in [7].

The velocity of the robot is constant. At each iteration the

robot is able to execute a direction adjustment to the left or to the right in the range [0, 15] (degrees).

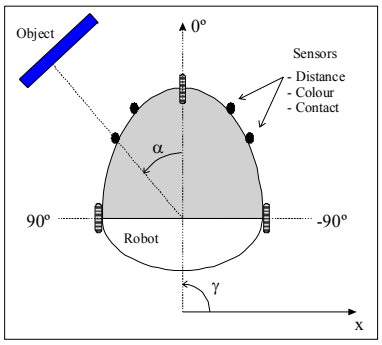

Fig. 3: Robot Model.

IV. System Overview

A. Introduction

It is argued that biological systems are the best reference for designing successful autonomous intelligent systems [10]. So, among several design options, the most biologically plausible one is the alternative most suitable to be chosen. This is one of the main principles adopted to guide the design of the system, called here *The Best Biologically Plausible Choice Principle* (*BBC Principle*).

A general view of the system is described next [13]. This work extends the system proposed in [7]. All design steps, namely, architecture, reasoning and learning, are conducted according to the BBC Principle.

The autonomous system is of the reactive type. The learning proceeds as the robot navigates. The robot eventually learns a suitable navigation strategy even though the initial performance is poor (with respect to the number of collisions and the robot's exploration ability).

In this section are only the aspects common to the system proposed in [7] considered. Innovations are detailed in the next section.

B. Architecture

The intelligent autonomous system corresponds to a neural network arranged in three layers (Fig. 4).

In the first layer there are two neural repertoires: Proximity Identifier repertoire (PI) and Color Identifier repertoire (CI). Distance sensors stimulate PI repertoire whereas color sensors feed CI repertoire. Both repertoires receive stimuli from contact sensors. The second layer is composed by two neural repertoires: Attraction repertoire (AR) and Repulsion repertoire (RR). Each one establishes connections with both networks in the first layer as well as with contact sensors. The actuator network, connected to AR and RR repertoires, outputs the adjustment on direction of the robot.

PI and CI repertoires: The neural network associated with each repertoire corresponds to a set of spatially-distributed columns of neurons. Each column of neurons is topologically fixed with a one-dimensional structure. There is no spatial influence among neurons of different columns (i.e. no cross-column connections).

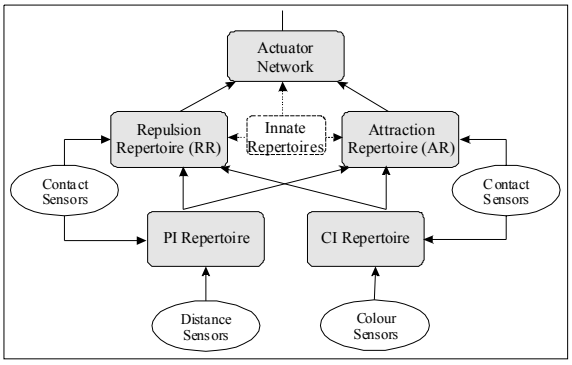

Fig. 4: Autonomous system architecture.

AR and RR Repertoires: A single layer of neurons form the AR and RR networks. The size of each network corresponds to the number of columns in the IP or IC networks. Each neuron in this layer connects to every neuron in the corresponding column of IP and IC networks, respectively.

Actuator Network: There are two layers in the actuators network. Each neuron in the first layer is associated to a fixed and predefined direction adjustment value. Each neuron receives stimuli from neurons in RA and RR networks and it is also connected to the unique neuron in the output layer.

Innate Repertoires: They are connected to the contact sensory field (not shown in Fig. 3 to improve clarity), and to the AR, RR, and actuator networks. The role of innate repertoires is to generate (attractive or repulsive) innate behaviors when the robot touches an object [13].

C. Learning

Following the BBC Principle, the classical reinforcement learning theory is adopted when designing the learning mechanisms for the proposed neural system.

According to this theory, reinforcement stimuli trigger learning processes. There are different types of reinforcements, some (few) innate and other acquired (second order reinforcement) [11] [12] [14]. Each type of innate reinforcement is able to elicit a specific innate behavior, essential to the integrity and development of the animal (e.g., sucking, grasping; and hunger and fear sensations). Innate behaviors are not associated to particular life experiences, but are a priori incorporated in the nervous system, during the epigenesis. They have a preponderant function in biological intelligent systems (including human beings), being the basis for acquiring more complex behaviors [12].

Collision avoidance and target seeking behaviors are acquired in an analogous way shown in [7], i.e. a reinforcement stimulus is generated when any type of contact is detected. Two types of reinforcements exist,

punishment and reward, detected when the robot interacts with repulsive or attractive objects, respectively. Whenever a reinforcement stimulus is received, an innate behavior is elicited and a learning mechanism is triggered.

V. Autonomous System Enhancements

A. Introduction

This section describes changes in the system model proposed in [7], changes with the purpose of reducing the deficiencies mentioned earlier. First, improvements are devised in order to overcome the deficiencies associated with unsuitable cyclic trajectories. This system is called System 1. Secondly, new changes are considered for improving the system performance when the robot is moving in a risky configuration region (reducing the number of collisions). The system resulting from this second set of improvements is referred to as System 2.

B. Suppressing Cyclic Trajectories

New models are described for the PI and RR repertoires (reasoning and learning) and a new innate behavior is defined. This section focuses on the design of System 1.

1) Innate Monotony Module

Experiments have shown that cyclic trajectories are associated with a monotonic state. A monotonic state is a sequence of navigation decisions in which the system does not detect any repulsive or attractive contacts during a long time interval [8].

The proposed system exploits this correlation, being able to detect monotonic states. This is an important role of the innate monotony module, considered as a part of the innate repertoires shown in Fig. 4. Besides, this innate module also triggers a particular learning process by an internally triggered reinforcement.

A fairly short time interval (2300 iterations) without contacts is initially sufficient to trigger the learning process. This time interval is then increased each time a monotony event is detected (see equation below), being bounded from above by a maximum of 8000 iterations:

$$\lambda(n+1) = \lambda(n) + 0.2\lambda(n)$$

2) PI Repertoire Reasoning

The new neuron model for the PI repertoire is as follows: Consider $\mathbf{x}^k(n) = [x_1, x_2, ..., x_m]^T$ and $\mathbf{w}_j^k(n) = [w_1, w_2, ..., w_m]^T$, the input (distance sensors) and synaptic weight vectors, respectively. The output of neuron j in column k; $k = 1,2,...,q$; $j = 1,2,...,l$; at iteration n, is defined as in (1):

$$y_j^k(n) = \begin{cases} u_j^k(n) f(\phi, n, j) & \text{if } j = i^k(\mathbf{x}^k(n), \Theta) \\ 0 & \text{if } j \neq i^k(\mathbf{x}^k(n), \Theta), \end{cases} \quad (1)$$

$$f(\phi, n, j) = \exp\left(-\left\|\mathbf{x}^k(n) - \mathbf{w}_j^k(n)\right\|\right)^2 / \phi). \quad (2)$$

$$i^k(\mathbf{x}^k(n), \Theta) = \arg\min_j \left\|\mathbf{x}^k(n) - \mathbf{w}_j^k(n)\right\|,$$

$$j \in \begin{cases} \{u \in \{1,2,...,l\} / u \notin W^k\} \text{ if } P \equiv \varnothing, \\ P = \{u / u \in W^k \text{ and } \left\|\mathbf{x}^k(n) - \mathbf{w}_u^k(n)\right\| < \Theta\}, \text{ otherwise;} \end{cases} \quad (3)$$

$$W^k = \{v \in \{1,2,...,l\} / v \equiv i^k(\mathbf{x}^k(m), \Theta), m = 1,...,(n-1)\},$$

where: ϕ and Θ are constant parameters; $u_j^k(n)$ is an adjustable parameter defined in (5) and (6).

Note that, according to (1), only the winner neuron $i^k(\mathbf{x}^k(n), \Theta)$ is fired at each iteration n.

The set W^k consists of neurons in column k that have previously been winners. Neurons in W^k, the *proud* neurons, are fired only if the input pattern is sufficiently similar to the synaptic weight vector (this similarity is determined by the acceptance parameter Θ) [13].

3) PI Repertoire Learning:

A learning mechanism is triggered if either a repulsive contact or monotony state is detected.

Repulsive Contact Event: If a repulsive contact event occurs, at iteration n, the learning adjusts parameters as follows. Consider the k^{th} column selected for learning (see [7]). Only winner neuron parameters, synaptic weights $\mathbf{w}_j(n)$ and activation potential $u_j(n)$, are adjusted according to (4) and (5):

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta_j(n)(\mathbf{x}(n) - \mathbf{w}_j(n))$$
$$\eta_j(n) = \varepsilon\eta_j(n-1), \quad (4)$$

where: $j = j^k(n, \mathrm{O})$ (the winner neuron is determined analogously to (3), substituting O for Θ) and; O and ε are constants.

$$u_j(n+1) = u_j(n) + \hat{\eta}(\gamma - u_j(n)) \quad (5)$$

where $\hat{\eta}$ and γ are constants.

This new learning is more accurate than in [7] because of the addition of "proud" neurons: a neuron outside the set W^k is selected for storing a new pattern only if the latter is sufficiently distinct from a stored pattern in W^k (first statement in the calculation of j in (3)): in this case the set W^k increases by one; otherwise, a synaptic adjustment on a neuron which has already been winner is accomplished (second statement in (3)).

Monotony Event: If a monotony event occurs, then only the activation potential $u_j(n)$, j = 1,2,…,l, is adjusted according to

$$u_j^k(n+1) = u_j^k(n) - \eta_m e_k(n) u_j^k(n) \quad (6)$$

where: $k \in \{r / e_r(n) > \tau \text{ and } r = 1,2,...,q\}$; $e_k(n)$ is defined in (8) and τ and η_m are constants.

4) RR Repertoire Reasoning

It is observed that the RR network develops a particular activation when unsuitable cyclic trajectories occur (see [13]). In order to find out which neurons contribute to the occurrence of this undesirable behavior, a new model is proposed for the RR repertoire. The model includes a parameter, called *degree of activity*, which has memory-like properties, providing a way to identify these neurons [13].

The neuron model for the RR repertoire is as follows. Consider $\mathbf{x}^k(n) = [x_1, x_2, ..., x_m]^T$ and $\mathbf{w}_j^k(n) = [w_1, w_2, ..., w_m]^T$, the input (distance sensors) and synaptic weight vectors, respectively. The output of neuron j in RR network, j = 1,2,...,q; at iteration n, is defined in (7):

$$y_j(n) = \begin{cases} 0 & \text{if } s_j <= \beta(n) \\ \alpha(s_j - \beta(n)) & \text{if } \beta(n) < s_j < (\beta(n) + 1/\alpha) \\ 1 & \text{if } s_j >= (\beta(n) + 1/\alpha), \end{cases} \quad (7)$$

$$\beta(n) = \beta(n) + \eta_\beta \beta_0,$$

where: $s_j = \mathbf{x}_j(n) \cdot \mathbf{w}_j(n)$ is the inner product between the input vector and the synaptic weight vector; and α and η_β are constants.

The parameter $e_j(n)$, called degree of activity, is defined as

$$e_j(n+1) = \begin{cases} e_j(n) + \varphi e_j(n) & \text{if } y_j > \kappa; \\ e_j(n) - \sigma e_j(n) & \text{otherwise,} \end{cases} \quad (8)$$

where: φ, σ and κ are constants.

Observe that $e_j(n)$ functions as a kind of memory for the neuron activity, having an important role in the repertoire learning mechanism.

5) RR Repertoire Learning

If a monotony event occurs at iteration n, then synaptic weights and parameters are adjusted only for neurons whose degree of activity is greater than a threshold. The adjustments are given by (9) and (10):

$$w_k(n+1) = w_k(n) - \eta e_h(n) w_k(n), \quad (9)$$

where $k \in \{r / e_r(n) > \tau \text{ and } r = 1,2,...,q\}$, $e_r(n)$ is defined in (8) and η is a constant.

$$\beta(n) = \beta_1; \quad (10)$$

The adjustment above influences the activation function of neurons in RR repertoire. Note that $\beta(n)$ evolves to β_0 according to (7).

C. Avoiding Risky Configurations

This section focuses on the design of System 2. The proposed scheme is integrated to the one described in the previous section, that is, System 2 is composed of System 1 as well as of the enhancements proposed next in this section.

This section describes new improvements to the neuron model for the PI repertory. In fact only the parameters ϕ and Θ are considered (see Equations (1), (2) and (3)). These parameters, constant in the previous neuron model, become adjustable, i.e. they are $\phi(n)$ and $\Theta(n)$ from now on.

The parameters are defined as follows:

$$\phi(n+1) = \begin{cases} \phi(n) + \alpha_1(\tilde{\phi} - \phi(n)) & \text{if } \psi(n) > \overline{\psi}, n \bmod \delta = 0; \\ \phi(n) + \alpha_2(\hat{\phi} - \phi(n)) & \text{if } v(n) > \overline{v}, n \bmod \delta = 0; \\ \phi(n) + \alpha_3(\hat{\phi} - \phi(n)) & \text{if } \psi(n) < \overline{\psi}, v(n) < \overline{v}, n \bmod \delta = 0; \\ \phi(n) & \text{if } n \bmod \delta \neq 0, \end{cases}$$

$$\psi(n) = \sum_{k=1}^{l} \begin{cases} 1 & \text{if } y_j^k > \overline{y}_a; j = i^k(n, \Theta) \\ 0 & \text{otherwise,} \end{cases} \quad (11)$$

$$v(n) = \sum_{k=1}^{l} \begin{cases} 1 & \text{if } y_j^k < \overline{y}_b; j = i^k(n, \Theta) \\ 0 & \text{otherwise,} \end{cases}$$

where: $\overline{\psi}$, $\overline{v}$, $\overline{y}_a$, $\overline{y}_b$, α_1, α_2, α_3, $\tilde{\phi}$, $\hat{\phi}$, and δ are constants.

$$\Theta(n+1) = \begin{cases} \tilde{\Theta} & \text{if } \psi(n) > \overline{\psi}, n \bmod \delta = 0; \\ \Theta(n) + \beta_1(\hat{\Theta} - \Theta(n)) & \text{if } v(n) > \overline{v}, n \bmod \delta = 0; \\ \hat{\Theta} & \text{if } \psi(n) < \overline{\psi}, v(n) < \overline{v}, n \bmod \delta = 0; \\ \Theta(n) & \text{if } n \bmod v\delta \neq 0, \end{cases}$$

……(12)

where β_1, $\tilde{\Theta}$ and $\hat{\Theta}$ are constants, and δ is such as in (11).

(Note that the dynamics of ϕ and Θ are different from the neuron output's dynamics.)

VI. Simulation Results

A. Introduction

This section shows evaluations for both systems 1 and 2, considering robustness as the general criterion. That is, their performances are evaluated for the cases in which navigation problems are brought about by specific environments.

Experiments are organized in two parts. The first part aims at confirming the capability of both systems for suppressing cyclic trajectories. The second part shows evaluations of the systems when the robot navigates in risky configuration regions.

B. Simulation Parameters

Environments and robot: The environments considered are those shown in Section II. The robot model has 67 sensor positions and its velocity is 0.28 distance units per iteration.

System 1: The parameters considered for System 1 are:

1- *RR Repertoire:*

It is structured in a single layer of 12 neurons (l). The weights are randomly initialized in the interval [0, 0.07]. Other initial values are $\kappa = 0.001$; $\varphi = 0.02$; $\sigma = 0.0005$; η_β=0.002; β_0=1.01; $\alpha = 1.3$; $\beta_1 = 2.5$; $e_j(0)$=1; $e_j(n) \in [0.1, 5]$; $\beta(0) = 1.1$; $\eta = 0.65$.

2- *PI Repertoire:*

The structure consists of 22 columns (q) each with 12 neurons (l). The weights are randomly initialized in the interval [0,1]. Other initial values are $\Theta = 0.63$; $O = 0.26$; $\hat{\eta}$=0.7; ϕ= 0.5; η_m=0.09; τ=0.1; ε = 0,6; γ = 0,6; $\eta_j(0) = 0.7$; $u_j(0) = 0.06$;

3- *CI Repertoire (see [7]):*

CI repertoire is structured identically to PI repertoire. The weights are randomly initialized considering the interval [0,1]. Synaptic adjustments in CI network (defined in [7]) use a varying learning rate (similar to (4)) with $\eta_j(0) = 0.9$; $m_j(0) = 5$;

4- *AR Repertoire (see [7]):*

$\eta' = 0.8$;

5- *Monotony Detection:*

$\lambda(0) = 2300$; $\lambda(n) \in [2300, 8000]$

System 2: The parameter configuration for System 1 is also considered for System 2. Also, the parameters associated with (11) and (12) are $\tilde{\phi}$ =0.007; $\hat{\phi}$=0.4; α_1=0.9; α_2=0.03; α_3=0.4; $\tilde{\Theta}$=0.2; $\hat{\Theta}$=0.63; β_1=0.08; $\bar{y}_a$=0.55l; $\bar{y}_b$=0.93l; $\Theta(0) = \hat{\Theta}$; $\phi(0) = \hat{\phi}$.

C. Experiments: Cyclic Trajectories

The system is evaluated by considering its capability for suppressing cyclic trajectories. Simulation experiments are carried out in the environment shown in Fig. 1 (Section II), where cyclic trajectories are observed for the system described in [7].

The experiment consists of 30 runs (for each system), each with 150000 iterations. Three different initial robot positions are considered (each third of the experiments is done considering a distinct initial robot position).

System 1: An example run, randomly chosen, is shown in Fig. 5 and the respective performance evolution, in Fig. 6. After the last monotony event is detected, the robot develops a suitable trajectory; the cumulative number of collisions becomes constant (i.e. the robot does not collide anymore) and the cumulative number of target captures grows linearly with time.

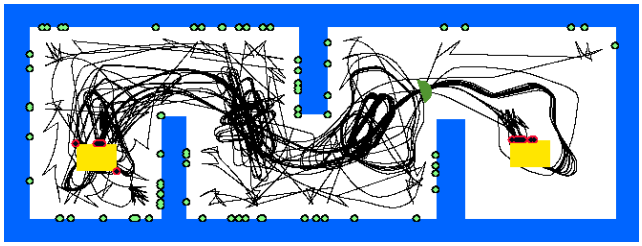

Fig. 5: A suitable navigation strategy is finally found after sufficient monotony events (not shown) – System 1.

The autonomous system is able to learn a suitable navigation strategy even in simple environments (where the number of objects is reduced) since monotonic events provide stimuli to trigger the learning mechanisms. Free of cyclic trajectories (and collisions), the robot keeps effectively and permanently working to reach targets, considering both systems 1 and 2.

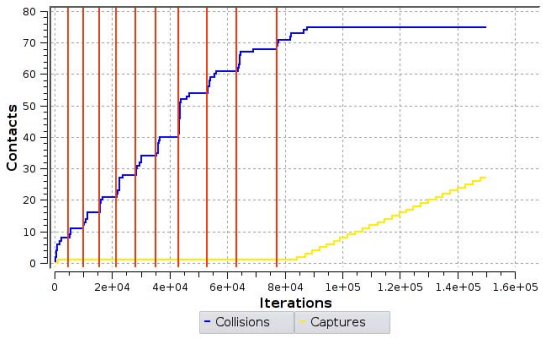

Fig. 6: Learning evolution (System 1): monotony events, number of collisions (with obstacles) and number of target captures are represented by vertical lines and, blue (upper) and yellow (lower) lines, respectively.

System 2: Fig. 7 shows the trajectory of a run (randomly chosen). The respective performance evolution is registered in Fig. 8. The general aspects are similar to System 1, i.e. System 2 is able to suppress cyclic trajectories, achieving an

acceptable performance.

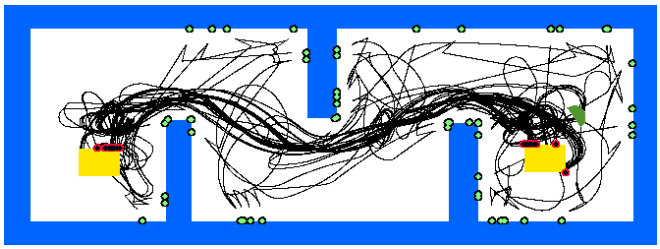

Fig. 7: After sufficiently many monotony stimuli, the robot develops a suitable navigation strategy - System 2.

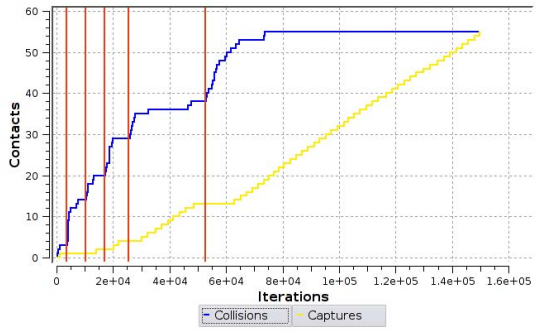

Fig. 8: Learning evolution for System 2 in Fig. 7. The cumulative number of collisions (blue) becomes constant after 80000 iterations (i.e. the robot does not collide anymore) and the number of captures (yellow) grows rapidly with time (i.e. the robot is an efficient hunter).

Discussion: Simulation results are summarized in Table I. The number of monotonic events is relatively small, considering the number of captures, collisions and total iterations. System 1 is better, on average, than System 2 considering the number of collisions and number of captures. However, the results are not fully unbiased since the number of runs (30) is small [13]. (Parameter tuning for System 2 may improve system performance in this case.)

TABLE I
EXPERIMENT STATISTICS

Mean	Collisions	Captures	Monotonies
System 1	72	47	5
System 2	84	44	7

D. Experiments: Risky Configuration

The comparison criterion adopted for evaluating Systems 1 and 2 is the ability to guide the robot through a risky region. The environment shown in Fig. 2 is considered for evaluation since its complexity is sufficient for causing poor autonomous navigation system performance (as shown in Section II).

The experiment consists of 22 runs for System 1 and 18 runs for System 2, each with 150000 iterations.

System 1: A run, randomly chosen, is shown in Fig. 9. The robot continually collides against obstacles, mainly when it crosses corridors or approaches corners.

Fig. 10 shows the respective performance evolution. Repulsive contacts keep increasing until the end of the simulation, indicating that System 1 is unable to learn a suitable strategy for navigating in risky configuration areas.

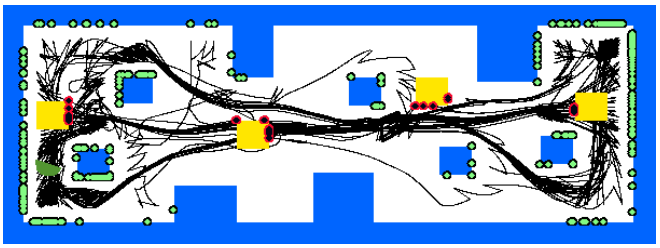

Fig. 9: System 1 displays a poor performance.

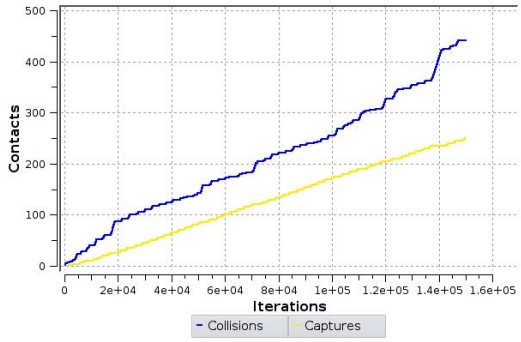

Fig. 10: Learning evolution for system 1 in Fig. 9. The cumulative number of collisions (blue) never becomes constant (i.e. the system continues to collide with obstacles) and the number of captures (yellow) grows slowly with time (i.e. the robot is not an efficient hunter).

Note that System 1 squanders part of navigation taking collision trajectories. This behavior, as expected, affects the target seeking behavior negatively, reducing the number of target captures (compare with System 2, see Table II).

System 2: A run, randomly chosen, is shown in Fig. 11. The robot follows a trajectory primarily characterized by a target seeking behavior.

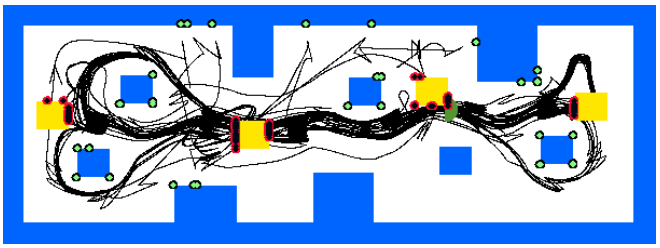

Fig. 11: System 2 achieves a good performance in the same environment where System 1 fails (compare with Fig. 9).

The good performance is confirmed in Fig. 12. The autonomous system learns how to explore the environment effectively after a few collisions (collisions are imperative so any autonomous system will develop an acceptable collision avoidance behavior). System 2 generates a suitable trajectory, nearly completely free of collisions, and accumulates a large number of captures.

Discussion: Table II summarizes experiment statistics. The difference in number of collisions in favor of System 2 is impressive. There are 527% more collisions for System 1 and 21% more captures for System 2.

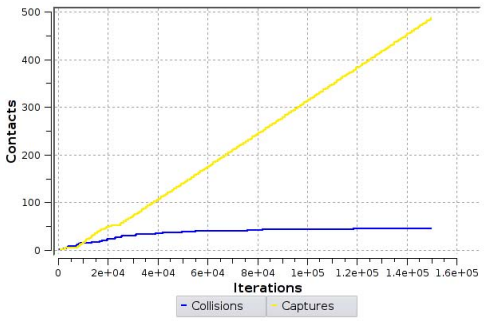

Fig. 12: Learning evolution for System 2 in Fig. 11. The cumulative number of collisions (blue) becomes approximately constant after 80000 iterations (i.e. the robot does not collide anymore) and the number of capture (yellow) grows quickly with time (i.e. the robot is an efficient hunter). This figure should be compared to Fig. 10 which shows how System 1 performed in the same environment.

TABLE II
EXPERIMENT STATISTICS

Mean	Collisions	Captures
System 1	445	317
System 2	71	384

VII. CONCLUSION

Autonomous navigation systems may be vulnerable to environment configuration, that is, undesirable robot behaviors, associated with a poor performance, can be observed depending on the environment configuration. Two such kinds of behaviors are considered: cyclic trajectories and inefficiency in risky configurations.

An autonomous navigation system is proposed based on two intelligent computation techniques: neural networks and classical reinforcement learning. The robot model does not have a specific target sensory field. As it guides the robot, the autonomous system learns to distinguish attractive objects from repulsive ones, developing coherently the appropriate responses (to approach or to avoid).

Several instruments are considered to create a system able to autonomously circumvent navigation difficulties, for instance: 1) design of a neuron parameter for functioning simultaneously as a memory of the neuron activity and as a learning factor, 2) conception of learning mechanisms for adjusting neuron parameters (not only synaptic weights) and 3) definition of a inner-triggered reinforcement.

For comparison purposes, different versions of the system are evaluated. Distinct environments are considered in order to investigate the learning capabilities for suppressing cyclic trajectories and achieving good performance when navigating in risky areas. Simulation results confirm that only the most refined system develops a relative good performance for both kinds of situations, particularly considering the second one (navigation in risky areas). Experiments show that the number of collisions is drastically reduced and the number of captures is increased as the navigation evolves, independent of narrow corridors, corners or desert (void) areas.

It is observed that the constant parameter values chosen are not critical, that is, they can vary over a wide range without degrading the system performance. Despite this favorable first impression, this characteristic may be an interesting theme for future works, including coupling parameter degree and adjustment mechanisms (supported by classical reinforcement learning strategies) in order to find suitable parameter values.

REFERENCES

[1] S. Haykin, *Neural Networks: a comprehensive foundation*, Prentice Hall, New York, USA, 1994.

[2] D. B. Fogel, *Evolutionary Computation – Toward a New Philosophy of Machine Intelligence*, 2nd edition, IEEE Press, 1999.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*, Oxford University Press, 1999.

[4] M. Dorigo, "Introduction to the Special Issue on Learning Autonomous Robots"; *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 26, (3), pp. 365-380, 1996.

[5] P. R. Crestani, M. Figueiredo, and F. Von Zuben, "A hierarchical neuro-fuzzy approach to autonomous navigation," in *Proc. of 2002 International Joint Conference on Neural Networks*, Honolulu, USA, 2002.

[6] R. Calvo, and M. Figueiredo, "Reinforcement learning for hierarchical and modular neural network in autonomous robot navigation," in *Proc. of 2003 International Joint Conference on Neural Networks – IJCNN*, Oregon, USA, 2003.

[7] E. A. Antonelo, M. Figueiredo, A-J Baerveldt and R Calvo, "Intelligent Autonomous Navigation for Mobile Robots: Spatial Concept Acquisition and Object Discrimination," in *Proc.6th IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Helsinki, Finland, 2005.

[8] R. Cazangi, and M. Figueiredo, "Simultaneous emergence of conflicting basic behaviors and their coordination in an evolutionary autonomous navigation system," *Proc. of 2002 IEEE Congress on Evolutionary Computation*, Honolulu, USA, 2002.

[9] P. Vershure, and T. Voegtlin, "A bottom up approach towards the acquisition and expression of sequential representations applied to behaving real-world device: distributed adaptive control III," *Neural Networks*, vol. 11, pp. 1531 – 1549, 1998.

[10] G. Edelman, *Neural Darwinism: the theory of neuronal group selection*, Basic books, New York, USA, 1987.

[11] E. Kandell, J. Schwartz, and T. Jessel, *Principles of neural science*, Elsevier, New York, 1991.

[12] J. W. Donahoe and D. C. Palmer, *Learning and complex behavior*, Allyn and Bacon, Needham Heights, Massachusetts, USA, 1994.

[13] E. A. Antonelo, "A neural reinforcement learning approach for behavior acquisition in intelligent autonomous systems," M.S. thesis, School of Information Science, Computer and Electrical Engineering, Halmstad University, Halmstad, Sweden, 2006.

[14] B. F. Skinner, *Science and human behavior*, New York, 1953.