Memory-Enhanced Evolutionary Robotics: The Echo State Network Approach

Cédric Hartland, Nicolas Bredeche, Michèle Sebag

Memory-Enhanced Evolutionary Robotics: The Echo State Network Approach

Cedric Hartland & Nicolas Bredeche & Michèle Sebag
TAO – CNRS – INRIA
Université Paris-Sud, F-91405 Orsay, France
FirstName.LastName@lri.fr

***Abstract*— Interested in Evolutionary Robotics, this paper focuses on the acquisition and exploitation of memory skills. The targeted task is a well-studied benchmark problem, the Tolman maze, requiring in principle the robotic controller to feature some (limited) counting abilities. An elaborate experimental setting is used to enforce the controller generality and prevent opportunistic evolution from mimicking deliberative skills through smart reactive heuristics.**

The paper compares the prominent NEAT approach, achieving the non-parametric optimization of Neural Nets, with the evolutionary optimization of Echo State Networks, pertaining to the recent field of Reservoir Computing. While both search spaces offer a sufficient expressivity and enable the modelling of complex dynamic systems, the latter one is amenable to robust parametric, linear optimization with Covariance Matrix Adaptation-Evolution Strategies.

I. Introduction

One prominent challenge in Autonomous Robotics is to go beyond reactive control [1] and to feature deliberative control, involving at least to some extent planning abilities [2]. One key difference between reactive and deliberative control is that the latter requires the robot to be endowed with some memory capacities. The robot situation, ruling the action selection task, cannot be determined from its current sensor values only and some additional information about the past is needed for disambiguation[1]. Quite a few approaches investigating explicit or implicit memory modelling have been proposed in the literature. On the explicit modelling side, a most simple approach is to retain all sensory information in the last T time steps, as done in classifier systems [3]; a more elaborate and demanding approach proceeds by encoding every possible situation of the robot in the search space [4]. Implicit memory modelling mostly proceeds by representing robotic controllers as recurrent neural nets (NN) [5], [6], [7], allegedly coding the memory of the previous time steps within the neuron states.

Most generally, the controller search space is required to be sufficiently expressive, in order to enable the representation of the desired dynamical system; while (recurrent) neuronal nets indeed meet this requirement [8], this result is by no means a constructive one. An equally important requirement thus regards the training feasibility: the search for satisfactory solutions must be computationally tractable. Focussing specifically on Evolutionary Robotics (ER) [5], an additional requirement regards the amount of human effort needed to find satisfactory solutions (e.g. through tuning the ER parameters). Prominent approaches in the ER literature address the above requirements in different ways. For instance Elman (recurrent) NNs are amenable to parametric continuous optimization; the expressiveness of the search space is ruled by the user-supplied number of neurons N, and the size of the optimization search space (the weight vector) quadratically increases with N [7]. In the NEAT framework, the number of neurons and NN topology are automatically adjusted (non parametric optimization); the efficiency of the approach is ruled by quite a few ER parameters, hand-tuned by the user [9].

This paper focuses on memory-enhanced Evolutionary Robotics, featuring two main contributions. The first contribution is a memory testing benchmark problem together with an appropriate experimental methodology. This problem, referred to as Tolman maze, is inspired from ethology experiments conducted by Tolman in the early 30s and related to latent learning in animals, specifically rats [10]. While the Tolman maze has been tackled in the classifier system setting using a discretized representation [11], a continuous representation is considered in this paper for the sake of scalability. The target behavior is to reach the third avenue in a maze, thus demonstrating some limited counting abilities (section III). Preliminary experiments show however that the well-known evolution opportunism makes it feasible to reach the goal through reactive-like behaviors. An original experimental setting, involving stochastic perturbations of the maze, is proposed to prevent reactive controllers from doing the job (section III-C).

As second contribution, the paper investigates the search space of Echo State Networks (ESNs) [12], pertaining to the so-called field of *Reservoir Computing*. ESNs differ from recurrent NNs in two main ways. Firstly, the topology and internal weights of the network are specified from macro-parameters, such as the connectivity rate and the highest eigenvalue of the connexion matrix; secondly, only the output weights are tuned in the training phase (section IV). Overall, ESNs are thus amenable to parametric continuous optimization, with linear complexity in the number of neurons. Covariance Matrix Adaption-based Evolution Strategies (CMA-ES),

[1]Notably, some memory capacities might also be required to handle reactive tasks in the case of perceptual aliasing, that is when same sensory information is acquired from different robot locations (e.g. due to sensor limitations); if these different locations ask for different reactions, then some memory capacity is required to disambiguate the task as well.

state of the art continuous evolutionary algorithms [13], are used in the rest of the paper to optimize ESNs.

The comparative experimental validation of the proposed approach, using NEAT as baseline, demonstrates that although both approaches can solve the problem, ESNs are amenable to frugal, robust and quasi parameterless optimization; the number of parameters left to hand-tuning is reduced by one order of magnitude compared to NEAT.

The paper is organized as follows. Section II briefly reviews and discusses the state of the art in Evolutionary Autonomous Robot Control. Section III presents the proposed memory-modelling benchmark problem, and discusses how to prevent evolution from tackling the problem in a reactive fashion. Section IV describes the proposed search space, presenting Echo State Networks and their design parameters.

The experimental setting is described in section III-C and validation results are presented in section V, comparing the performances of ESNs and NEAT [9] and discussing their sensitivity with respect to the experimental setting. The paper concludes with some perspectives for further research.

II. State of the art

Well-posed problems in robotics are handled using control theory, modelling the target goal and the environment in terms of differential equations [14]. The control framework gives optimality guarantees regarding the solution controller and its stability. This framework however relies on strong assumptions, including a comprehensive model of the world.

More general settings include Evolutionary Robotics (ER) [5] and Reinforcement Learning (RL) [15]. In both settings, the goal is expressed via a fitness or reward function; solutions are policies maximizing the fitness function in the ER setting, or maximizing the (discounted) reward expectation in the RL setting. The main difference between both settings regards the optimization method and search space. RL proceeds by estimating value functions, namely the reward associated to each state in the search space, or to each pair (state, action) through the so-called Hamilton-Jacobi-Bellman equations [16]; an optimal policy can henceforth be derived by selecting in every state the action maximizing the associated reward or leading to the state with maximal reward. While RL provides sound guarantees of optimality, it hardly scales up w.r.t. the size of the state and action spaces, particularly so when no model of the environment is provided[2]. Quite the opposite, ER directly explores the controller search space; while it does not offer optimality guarantees, it handles large-scale and complex specifications, e.g. related to swarm robotics [17] or morphogenesis (optimizing both the robot design and policy [18]); specific search spaces (rulesets [3], trees. graphs or dynamic systems [19]) can be considered; additional heuristics, e.g. related to controller diversity, can be used [20].

Memory modelling in RL is mostly based on redesigning the state space [4]; for instance, each spatial state might be duplicated depending on whether it has been formerly visited, or depending on the current stage of the robot w.r.t. the planned goal. In ER, classifier system approaches [3] likewise rely on explicit memory modelling, with the difference that the controller is provided with the sensory information in the last L time steps; how to exploit this information is left to the ER engine. Neural net-based ER approaches rely on the recurrent topology properties [21], or on the neuron type itself, e.g. using spiking neurons [22], to enable memory modelling. The question is to find a good tradeoff between the size of the search space (number of weights) and the memory span supported by the architecture. Other approaches, chiefly NEAT [23], optimize both the topology and weights of the NN. While NEAT does adjust the size of the search space, it requires quite a few evolutionary control parameters to be properly adjusted by trials and errors. An alternative to NEAT is based on the parametric optimization of the so-called Echo State Networks proposed by Jaeger [12]. After recent investigations in the domain of Autonomous Robotic or Optimal Design [24], [25], [26], ESNs are amenable to frugal and efficient optimization (more in section IV).

Another critical issue regards the design of the reward/fitness function. As noted by all practitioners, Evolutionary Computation tends to fall into every "hole in the fitness function", which is referred to as Evolution Opportunism[3]. Notably, heuristics enforcing the controller population diversity [27] might counteract Evolution Opportunism, enforcing the discovery of more diverse and thus non-trivial solutions. In the general case however, efficient fitness functions often result from some co-evolution between the evolutionary engine and the designer, barring the discovery of undesired solutions while yielding a tractable fitness landscape.

III. The Tolman Maze Experiment

This section first describes the proposed memory-testing benchmark problem. Preliminary experiments illustrate the so-called Evolution Opportunism and gives additional insights into the distinction between reactive and deliberative behaviors. Lastly, a robust experimental methodology is proposed, enforcing the controller generality, the neutrality of the fitness landscape, and the robustness of evolutionary selection with respect to the experimental noise.

A. The Tolman Maze

The proposed benchmark is remotely inspired from the one used in ethology in the early 1930's by Tolman [10], to demonstrate the *latent learning* of rats. This benchmark problem is a maze with strong regularities (Fig. 1), where the robot location cannot be determined from its only short-range sensors. Typically, the entries of every branch look alike (perceptual aliasing), preventing the robot from directly reaching

[2]When the model is provided, policy learning can be directly tackled in terms of maximum a posteriori estimation; see [2] for a Bayesian approach.

[3]More precisely, the fitness function may admit inappropriate behaviors as optimal solutions; if these are simple, they tend to crowd the population and the evolutionary search never recovers. For instance, if the fitness function heavily penalizes the robot bumping into obstacles, then the controller population is soon crowded with "optimal" inappropriate behaviors, staying motionless or rotating on oneself.

Fig. 1. The robot starts from the extreme left position, heading to the right; its target location is indicated with a cross.

the target location (here the end of the third branch) based upon its only sensor values. The simplest way of overcoming perceptual aliasing is to provide the robot with better sensors, e.g. a long-range camera would enable to distinguish between the branches of the Tolman maze. Better sensors however would not allow for making the distinction between branch n and $n+1$ in a generalized Tolman maze; furthermore, the richer the sensors, the higher the dimension of the controller input space is, severely hindering the training/learning process.

A principled way of overcoming perceptual aliasing is through endowing the robot with some form of memory, compensating lesser spatial skills with better temporal ones. As mentioned earlier on, the use of explicit memory has already been investigated in relation with the Tolman maze within the classifier system framework [11], with two main limitations. Firstly, as the controller is given access to the sensor values in the last L time steps, the size of the input space is increased by a factor L, making the controller training significantly more difficult. Secondly, classifier systems mostly handle discrete input spaces, scaling up to a limited extent; in practice, classifier systems hardly face medium scale arenas (above a few hundred squares). For these reasons, the proposed approach will consider a continuous setting, and tackle implicit memory modelling.

B. Evolution Opportunism

The controller goal is to reach as fast as possible the target location, the end of the third maze branch noted x^*, and to stay there. After the Evolutionary Robotics standards [5], the controller fitness measures its behavior averaged over K epochs to account for the intrinsic variability of robotic control. An epoch includes T time steps; at the beginning of the epoch the robot is located at the extreme left of the maze and heads toward the right (Fig. 1). The length T of the epoch is about four times the time needed to directly reach x^*. When bumping in a wall, the robot stays motionless thereafter.

Denoting $x(t)$ the robot position at time t, a first fitness function $\mathcal{F}$ was defined as the MSE between $x(t)$ and x^*:

$$\sum_{t=0}^{T} ||x(t) - x^*||^2$$

Experiments, based on a standard Khepera II with 8 proximity infrared sensors and 2 effectors (right and left wheels), were done *in silico* using the SIMBAD robot simulator [28]. Various controller architectures were considered and optimized using Evolutionary Algorithms. A multi-layer perceptron (MLP) was used as baseline architecture for a sanity check. MLPs indeed enable reactive control only, as the next action (moves of the right and left wheels) only depends on the instant sensor values.

Unexpectedly however, evolved MLPs[4] were found to solve the Tolman maze. A closer look at the robot behavior (Fig. 2) reveals the "hole in the fitness function": the robot is found to go until the extreme right of the maze; at this point, it bounces at a given angle; when close to the upper wall, it makes a 90 degree turn on its left and eventually arrives at the target location.

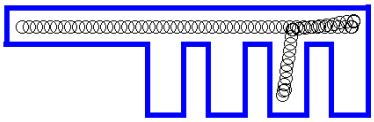

Fig. 2. The reactive behavior implemented by a multi-layer perceptron succeeds in finding the third branch.

In short, the Tolman maze together with the above fitness function admits a reactive solution: the MLP controller defines a trajectory which is not prone to perceptual aliasing and robustly reaches the target location despite the sensor noise and the noise on the initial conditions. This trajectory is however fragile: it is bound to fail if the third branch is slightly moved toward the right or left, if there is no wall on the extreme right of the maze. In short, the MLP controller implements a smart reactive heuristics with no generality.

C. Enforcing Controller Generality

The desired generality property can be best understood by reference to the Machine Learning (ML) framework. In supervised ML settings, not only should the solution be compliant with the available examples (training set); more importantly, it should be accurate on further unseen examples (test set). The generality property in ML is enforced through two main ingredients [30], increasing the information in the training set (considering more examples and more diverse ones) and requiring the solution to be sufficiently simple (Occam's razor).

While the simplicity requirement conveniently restricts the ML search space, only elementary simplicity constraints (e.g. limiting the number of hidden neurons) can be easily enforced in an evolutionary framework; more elaborate simplicity constraints (e.g. penalty terms) require significant efforts to be adjusted. Therefore the proposed experimental setting is only based on diversifying the training environments. Firstly, only open mazes are considered (no wall after the last branch); secondly, the distance d between the extreme left of the maze and the first branch, and the distance d' between two consecutive branches (Fig. 3) are varied. Formally, in each epoch d and d' are uniformly drawn in a given range (respectively $[5, 16]$ and $[\frac{1}{3}, \frac{5}{3}]$). The maze defined from d and d' is used to measure the fitness of all individuals in the population during

[4]CMA-ES with default parameters [29] was used to evolve the MLP weight vector (in $\mathbb{R}^{46}$); evolution is stopped after 200,000 fitness evaluations.

this epoch. The fitness of every individual is then averaged over 16 independent epochs[5].

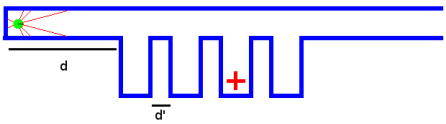

Fig. 3. Stochastic Tolman Maze: d and d' are perturbed in each epoch.

Along this new experimental setting, MLP performances fall down as expected; in the best case, the controller proceeds by exploring all branches (Fig. 4).

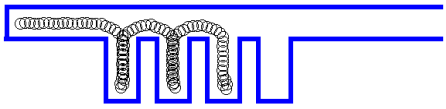

Fig. 4. A Successful MLP behavior in a Stochastic Tolman Maze

D. Prior Knowledge and Neutrality

Another way of guiding evolution towards robust and general controllers is to design educated fitness functions, incorporating more knowledge about the problem domain and/or the specificities of evolutionary search.

Fitness $\mathcal{F}_d$, inspired from the RL progress estimator [31], incorporates extensive domain knowledge. The task at hand actually involves two sub-tasks: i) arriving at the entry of the third branch noted z^*; ii) arriving at the end of the third branch x^*. Accordingly, fitness $\mathcal{F}_d$ multiplicatively aggregates two terms, the minimal distance between the robot trajectory and z^*, and the minimal distance between the trajectory and x^*:

$$\mathcal{F}_d = (min_{t=1...T}||x(t)-z^*||^2+1)\times(min_{t=1...T}||x(t)-x^*||^2)$$

Arguably, $\mathcal{F}_d$ both is crafted to the problem and defines a nice and smooth fitness landscape; the results obtained with $\mathcal{F}_d$ will thus serve to assess the controller architectures at their best.

Quite the contrary, fitness $\mathcal{F}_i$ simply measures the minimum distance to the target location, and carries minimal information about the environment:

$$\mathcal{F}_i = min_{t=1...T}||x(t) - x^*||^2$$

The $\mathcal{F}_i$ landscape involves a high amount of neutrality; anything the robot can do after the time step where it is closest to x^* is "for free", favoring the exploration of the maze. In the meanwhile, this landscape features a local optimum: the end of the second branch is closer to the goal than the entry of the third branch; any controller arriving at the end of the second branch is more fit than a controller cruising in the top avenue of the maze.

[5]A few preliminary experiments, not shown in the paper, showed that this number of epochs achieves a good trade off between the overall computational load and the solution robustness. More in section VI.

E. Noise and Vickrey Auction

Evolutionary Robotics faces yet another critical issue ruling the controller generality, namely the fitness noise, reflecting the variability of sensors, effectors and initial conditions. Actually, the average behavior over a limited number of epochs gives limited indications about the controller performance (see section VI). Further, a lucky controller getting an overly optimistic fitness can crowd the population and prevents evolution from discovering better individuals. Increasing the number of epochs indeed decreases the noise strength; it however increases the overall computational cost which is already quite high [5]. General heuristics are proposed in the EC literature to deal with noisy fitness optimization, e.g. ranging from the control of the mutation step [32] to that of the fitness averaging based on ranking tests [**?**]. Another mechanism relying on fitness neutrality is investigated in this paper.

Let $\mathcal{F}_t^*$ denote the best fitness value reached up to generation t. Let $\{(x_i, \mathcal{F}(x_i))\}$ denote the set of individuals in generation $t+1$ together with their fitness. The selection procedure is modified by thresholding the fitness of individual x_i to[6] $max(\mathcal{F}_t^*, \mathcal{F}(x_i))$ for x_i ranging in generation $t+1$.

The advantage of this mechanism is twofold. Firstly, it somehow relaxes the competition between individuals in the same generation; in a phenotypic perspective, individuals are only compared with the best individual in the previous generation. Secondly, all individuals improving on the previous best individual are being attributed the same fitness, which better preserves the population diversity.

This mechanism can be likened to the so-called Vickey auction, or second-price sealed-bid auction, [33], where the winner is the individual with highest bid although it pays the price proposed by the second highest bid. A most interesting property of the Vickrey auction is to provide an incentive to bid truthfully (the individual utility is not increased by over- or under-bidding) while it does not lead to ultimately disclosing the "true"market price.

Analyzing noisy artificial evolution in terms of bidding mechanisms, and understanding the properties of the proposed mechanism in more depth, is left for further study.

IV. Echo State Networks

This paper, concerned with memory-enhanced evolutionary robotics, investigates a new recurrent neuronal architecture referred to as Echo State Networks. This section briefly introduces ESNs, referring the reader to [12] for a more comprehensive presentation, and describes their evolutionary optimization of ESNs for autonomous robot control.

A. Formal Background

Despite major applicative successes (notably related to pattern recognition), neuronal nets (NNs) have been out-passed by Support Vector Machines (SVMs) in the field of Machine Learning since the early 90s [30]. One reason for this fact

[6]Formally, the Covariance Matrix Adaptation step [29] used in the experiments is based on individuals $(x_i, max(\mathcal{F}_t^*, \mathcal{F}(x_i))$ (remind that the goal is to minimize $\mathcal{F}$).

is that SVM learning is amenable to quadratic optimization and thus provides optimality guarantees, contrasting with NN learning based on gradient approaches.

The NN revival since the early 2000, through the emergence of the so-called *Reservoir Computing* field [12], [22], involves massively unstructured networks, only constrained from global and/or statistical properties. Formally, let an ESN network be defined from its input $x_1, \ldots, x_K$, its output $y_1, \ldots, y_L$, and N hidden neurons $e_1, \ldots, e_N$ (Fig. 5), where each hidden neuron implements a standard sigmoidal activation function. In the general case, inputs are connected to hidden neurons through the $K \times N$ matrix $\mathcal{W}_{in}$. Likewise, hidden neurons are connected to the outputs through the $N \times L$ weight matrix $\mathcal{W}_{out}$. Finally, hidden neurons are connected together through the $N \times N$ matrix $\mathcal{W}$, where $w_{i,j}$ commonly denotes the connexion weight from neuron e_i to neuron e_j.

The specificity of ESNs compared to standard NNs is twofold. Firstly, ESN training only modifies matrix $\mathcal{W}_{out}$, referred to as readout matrix. Therefore, the underlying optimization problem has linear size in the number N of hidden neurons (contrasting with the quadratic size of e.g. Elman recurrent NNs [21]). Secondly, the ESN core, made of the hidden connexion matrix $\mathcal{W}$, is specified from two macro-parameters, namely the connectivity rate ρ and the highest eigenvalue α of $\mathcal{W}$, referred to as damping factor. Formally, $w_{i.j}$ is set to 0 with probability $1-\rho$. Otherwise, $w_{i,j}$ is set to a or $-a$, where a is such that α complies with the prescribed damping factor, usually $\alpha = .95$. The control of the damping factor thus ensures that the ESN is a stable recurrent neural net.

Overall, designing an ESN amounts to controlling two global features (the connectivity rate ρ and damping factor α); and a few shallow features (the weights on the output connections). In contrast, NNs rely on controlling every and all features: global ones (e.g. the number and size of layers) and shallow ones (every weight).

B. Discussion

The appeal of ESNs is explained from the fact that, since only the output weight matrix $\mathcal{W}_{out}$ needs be trained to address supervised learning tasks[7], ESN learning is amenable to a well-posed optimization problem. For instance, ESN-based regression is achieved by minimization of the mean square error, akin linear regression despite the fact that ESN models offer a much higher expressiveness than linear models. Formally, ESNs (like neural nets) are universal approximators of L_2 functions [34].

The rationale behind training $\mathcal{W}_{out}$ only, is that the hidden neurons, also referred to as reservoir, encode a large variety of behaviors; the target dynamic system can thus be approximated as a weighted sum thereof. In short, the breakthrough of ESNs and more generally Reservoir Computing is to make the modelling complex dynamic systems a well-posed optimization problem, with linear size in the number of hidden neurons.

[7]The number N of hidden neurons, connectivity rate ρ and damping factors α are handled as hyper-parameters, akin SVM hyper-parameters.

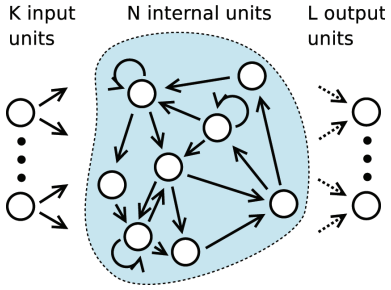

Fig. 5. Structure of an Echo State Network

ESNs have been used with good results for tasks involving temporal information, such as time series prediction or Reinforcement Learning, exploiting the fact that the memory of the system is encoded within the hidden neuron state [35]. For instance, ESNs have been used to tackle standard RL benchmark problems such as the inverted pole [25], using Evolution Strategies with Covariance Matrix Adaptation (CMA-ES) [29] to optimize the readout matrix, demonstrating the robustness of the approach with respect to earlier results [36].

Focusing on Autonomous Robotics, earlier work involving ESNs includes [26] and [37]. The former work is concerned with the detection of complex events and the robot localization within the environment. The latter one is concerned with Apprenticeship Learning [38], [39]: the traces of the teacher's behavior define a regression problem (reproducing the teacher actions depending on the sensor values), which is tackled using ESNs.

V. Experimental Study

This section considers the memory-testing benchmark problem defined in section III. Two main evolutionary approaches are compared, the prominent NEAT [9] and the continuous evolutionary optimization of Echo State Networks.

A. Experimental Setting

Evolutionary ESNs (EESNs) are optimized using Evolution Strategies with Covariance Matrix Adaptation[8], a state-of-the-art, quasi-parameterless evolutionary optimization algorithm [29]. EESNs involve three parameters: the number N of hidden neurons (dimension of the search space), the connectivity rate ρ and the damping factor α. After a few preliminary experiments, ρ and α were respectively set to 10% and .9, while N was varied in $\{50, 100, 200\}$. Due to space limitations, only the results obtained with $N = 100$ are reported in the following.

NEAT [9], used at first with its default parameters, failed to solve the problem. Its failure was a posteriori blamed on an insufficient population size, preventing NEAT from exploiting solution clusters. In the remainder of the paper, the NEAT parameters were taken from [24], with a population size set to 500.

[8]The java implementation of CMA-ES, available at http://www.bionik.tu-berlin.de/user/niko/cmaesintro.html, was used.

Each controller simulation involves $T = 200$ time steps; the fitness of each controller is averaged on K epochs (independent simulations) with K ranging in 8, 16, 32; due to space limitations, only results with $K = 16$ are reported. Each run is stopped after 200,000 simulations.

Two success criteria are considered. The online success is measured from the fitness of the best individual at each generation, averaged over 100 simulations to filter out the noise. The success rate measures whether the robot gets "sufficiently" close to the target location after a tolerance parameter ϵ, experimentally set to half the branch length. Parameter ϵ however is not very sensitive as every robot either goes very close to the target location, or stays far away. All results are averaged over 11 independent runs with same parameters.

For both EESNs and NEAT, the computational time is circa 48 hours on Dual Core AMD Opteron 1.8GHz computers.

B. Experimental Results

Fig. 6 displays all results obtained with EESNs and NEAT, respectively considering fitness $\mathcal{F}_d$ (first and second rows) and fitness $\mathcal{F}_i$ (third and fourth rows); the fifth row reports the EESNs results with fitness $\mathcal{F}_i$ and Vickrey-based option (section III-E). The left column reports on the fitness (median value, 25 and 75% quantiles); the middle column indicates the success rate; the right column is the time (number of steps) needed by successful controllers to reach the target location.

Overall, EESNs and NEAT reach comparable performances; they however diversely react to the two fitness functions. The fitness landscape defined by the (prior knowledge-based) fitness $\mathcal{F}_d$ is more amenable to CMA-ES optimization than to NEAT, as demonstrated by NEAT stagnating median fitness. The results also show that, although $\mathcal{F}_d$ was hand-crafted to the task at hand, it is hardly relevant; an improved fitness value does not necessarily imply that the success rate likewise increases. In practice NEAT reaches a success rate of circa 80% after 80,000 simulations only, and its success rate thereafter decreases.

A different picture is offered by the $\mathcal{F}_i$ fitness function. On average, NEAT individuals significantly outperform EESNs, with a much smaller fitness variance and a significantly higher success rate. In the meanwhile, EESNs yield a wide variety of ESNs and provide competent individuals (success rate 98%) much sooner than NEAT; ESNs with success rate 98% appear after 55,000 simulations (vs 100,000 for NEAT).

All results are summarized in Table I; the average fitness value is omitted due to the order of magnitude differences among the runs. The only statistically significant difference between EESNs and NEAT concerns the controller speed (rightmost column), the number of time steps needed to reach the target location. With fitness $\mathcal{F}_d$, the average number of time steps required by EESNs is 48 ± 5 with $\mathcal{F}_d$, vs 60 ± 21 for NEAT. With fitness $\mathcal{F}_i$, the average number of time steps required by EESNs is 46.5 ± 5 vs 55 ± 10 for NEAT. A tentative interpretation for this difference, suggesting that EESNs undergo a stronger pressure in favor of fast controllers than NEAT, goes as follows. By definition the target behavior relies on the memory of the past trajectory; the memory is encoded in the hidden neuron states. Within an ESN, this memory vanishes exponentially fast as time goes by (remind that the damping factor is .9). The faster the ESN controller, the more it can rely on its memory; efficiency and speed are thus tightly related. In opposition, NEAT does not set any explicit constraint on the damping factor of the recurrent NNs; further study will be devoted to the structure of the NEAT individuals (more in section VI).

Finally, the Vickrey option (section III-E) is shown to significantly improve the EESNs results, both in terms of fitness value and success rate (last row in Table I). Even more importantly, it significantly speeds up the convergence of evolution; as shown in Fig. 6 (bottom row), the first competent controller is found in about 20,000 evaluations against respectively 40,000 and 50,000 for $\mathcal{F}_d$ and $\mathcal{F}_i$ without Vickrey option. In the meanwhile, NEAT was adversely affected by the Vickrey option; a tentative interpretation for this fact is that the neutrality induced by thresholding the individual fitness prevents NEAT from forming appropriate solution clusters.

VI. Discussion and Perspectives

This paper has investigated the feasibility of an autonomous robot controller with counting abilities, comparing two evolutionary approaches: the well-known NEAT achieves the non-parametric optimization of a recurrent neuronal net [9]; a new recurrent neuronal architecture, Echo State Networks [12], is amenable to efficient parametric optimization through CMA-ES. Experimental results suggest that Evolutionary ESNs open promising research avenues for implicit memory modelling, with similar performances and significantly lesser hyper-parameters to tune than NEAT.

Two main advances have been made regarding the experimental methodology. On the one hand, stochastic perturbations of the Tolman maze have been used to enforce the generality of the controller solution and discard trivial solutions. Secondly, an original heuristics inspired from Computational Economics, the Vickrey option, has been proposed to handle the fitness noise; it has been shown to dramatically speed up the discovery of accurate solutions. An in-depth analyzis of the Vickrey effects is left for further studies.

The presented work opens a new research avenue in Evolutionary Robotics, based on the comparison of the neural net architectures respectively built by NEAT and EESNs. Preliminary investigations show that the NEAT solutions have a low connectivity rate and a number of hidden neurons ranging from 50 to 80, making their structure close to that of the ESN solutions. It naturally comes to consider the eigenvalues of the NEAT NNs, and to investigate how these relate to the memory skills and speed of the encoded controller. The ultimate question, along the same lines as [**?**], is whether the performance of the controller relates to some deep characteristics of the neural connexion matrix.

		Fitness value		Success rate (/100)			Time to target		
		Best	Median	Best	Median	Average	Best	Median	Average
F_d	NEAT	$4.6.10^{-2}$	0.97	98	78	57.4 ± 37.2	44	49	59.6 ± 21.26
	ESN	$4.5.10^{-2}$	0.18	99	70	59.06 ± 31.52	40	47	47.7 ± 5.43
F_i	NEAT	1.10^{-2}	0.94	95	76	72.87 ± 14.8	45	51	58.6 ± 14.89
	ESN	$9.33.10^{-4}$	0.84	98	24	43.75 ± 32.0	40	46	46.45 ± 4.86
F_{iV}	ESN	1.10^{-3}	$7.1.10^{-2}$	98	91	68.0 ± 31.5	39	44	45.25 ± 5.73

TABLE I

THE TOLMAN MAZE: EESNS AND NEAT RESULTS. FITNESS VALUES (LEFTMOST PART), SUCCESS RATE (MIDDLE PART) AND CONTROLLER SPEED (RIGHTMOST PART) WITH FITNESS $\mathcal{F}_d$, $\mathcal{F}_i$ AND $\mathcal{F}_i$ WITH VICKREY OPTION, AVERAGED ON 11 INDEPENDENT RUNS.

REFERENCES

[1] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Winning the darpa grand challenge," *Journal of Field Robotics*, 2006.

[2] M. Toussaint, V. Willert, J. Eggert, and E. Korner, "Motion segmentation using inference in dynamic bayesian networks," in *British Machine Vision Conference 2007*, 2007.

[3] P. L. Lanzi, "Adding Memory to XCS," in *Proceedings of the IEEE Conference on Evolutionary Computation (ICEC98)*. IEEE Press, 1998.

[4] N. Meuleau and R. I. Brafman, "Hierarchical heuristic forward search in stochastic domains," in *IJCAI*, M. M. Veloso, Ed., 2007, pp. 2542–2549.

[5] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press/Bradford Books, 2000.

[6] J. Urzelai and D. Floreano, "Evolution of adaptive synapses: Robots with fast adaptive behavior in new environments," *Evol. Comput.*, vol. 9, no. 4, pp. 495–524, 2001.

[7] E. Tuci, V. Trianni, and M. Dorigo, "Evolving the feeling of time through sensory-motor coordination: a robot-based model," in *The 8th International Conference on Parallel Problem Solving From Nature (PPSN 2004)*, ser. Lecture Notes in Computer Science, X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervós, J. Bullinaria, J. Rowe, P. Tiňo, A. Kabàn, and H. Schwefel, Eds., vol. 3242. Springer Verlag, Berlin, Germany, 2004, pp. 1001–1010.

[8] R. Hecht-Nielsen, *Neuro-computing*. Addison Wesley, 1989.

[9] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[10] E. Tolman and C. Honzik, "Insights in rats," *University of California Publications in Psychology*, vol. 4, no. 14, pp. 215–232, 1930.

[11] P. L. Lanzi, "An analysis of the memory mechanism of XCSM," in *Genetic Programming 1998: Proceedings of the Third Annual Conference*, J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, Eds. University of Wisconsin, Madison, Wisconsin, USA: Morgan Kaufmann, 22-25 1998, pp. 643–651.

[12] H. Jaeger, "A tutorial on training recurrent neural networks. covering bptt, rtrl, ekf, and the echo state network approach," in *GMD report 159. German National Research Center for Information Technology*, 2002.

[13] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[14] J.-P. P. Laumond, *Robot Motion Planning and Control*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1998.

[15] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Advances in Neural Information Processing Systems 19*, B. Scholkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 1–8.

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[17] G. Baldassarre, D. Parisi, and S. Nolfi, "Distributed coordination of simulated robots based on self-organisation." *Artificial Life*, vol. 12, no. 3, pp. 289–311, Summer 2006.

[18] H. Lipson and J. B. Pollack, "Automatic design and manufacture of robotic lifeforms." *Nature*, vol. 406, no. 6799, pp. 974–978, August 2000.

[19] S. Nolfi and D. Parisi, "Auto-teaching networks that develop their own teaching input," in *Proceedings of the Second European Conference on Artificial Life*, J. Deneubourg, H. Bersini, S. Goss, G. Nicolis, and R. Dagonnier, Eds., Brussels, Free University of Brussels, 1993.

[20] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer, November 2003.

[21] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[22] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: a new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, no. 11, pp. 2531–2560, 2002.

[23] R. M. K.O. Stanley, B.D. Bryant, "Evolving adaptive neural networks with and without adaptive synapses," *Evolutionary Computation*, vol. 4, pp. 2557–2564, 2003.

[24] A. Devert, N. Bredeche, and M. Schoenauer, "Unsupervised learning of echo state networks: A case study in artificial embryogeny," in *Artificial Evolution*, 2007, pp. 278–290.

[25] F. Jiang, H. Berry, and M. Schoenauer, "Supervised and evolutionary learning of echo state networks," in *PPSN X, 10th International Conference on Parallel Problem Solving from Nature*, 2008, pp. 215–224.

[26] E. Antonelo, B. Schrauwen, X. Dutoit, D. Stroobandt, and M. Nuttin, "Event detection and localization in mobile robot navigation using reservoir computing," in *Proceedings of the International Conference on Artificial Neural Networks*. Porto: Portugal, 9 2007.

[27] J. Lehman and K. O. Stanley, "Exploiting open-endedness to solve problems through the search for novelty," in *Proceedings of the Eleventh International Conference on Artificial Life (ALIFE XI)*. Cambridge, MA: MIT Press, 2008.

[28] L. Hugues and N. Bredeche, "Simbad : an autonomous robot simulation package for education and research," in *Proceedings of The Ninth International Conference on the Simulation of Adaptive Behavior (SAB'06)*. Rome, Italy: Published in Springer's Lecture Notes in Computer Sciences / Artificial Intelligence series (LNCS/LNAI) n.4095, 2006, pp. 831–842.

[29] N. Hansen and S. Kern, "Evaluating the CMA evolution strategy on multimodal test functions," in *Parallel Problem Solving from Nature PPSN VIII*, ser. LNCS, X. Yao *et al.*, Eds., vol. 3242. Springer, 2004, pp. 282–291.

[30] V. Vapnik, *Statistical learning theory*. Wiley, 1998.

[31] M. J. Mataric, "Reinforcement learning in the multi-robot domain," *Autonomous Robots*, vol. 4, no. 1, pp. 73–83, 1997.

[32] H.-G. Beyer, "Mutate large, but inherit small ! On the analysis of mutations in $(1, \lambda)$-ES with noisy fitness data," in *Proceedings of the 5^{th} Conference on Parallel Problems Solving from Nature*, T. Bäck, G. Eiben, M. Schoenauer, and H.-P. Schwefel, Eds. Springer Verlag, 1998, pp. 109–118.

[33] P. Cramton, Y. Shoham, and R. Steinberg, "Combinatorial auctions," University of Maryland, Department of Economics - Peter Cramton, Papers of Peter Cramton 04mit, 2004.

[34] M. Ozturk, D. Xu, and J. Principe, "Analysis and design of echo state

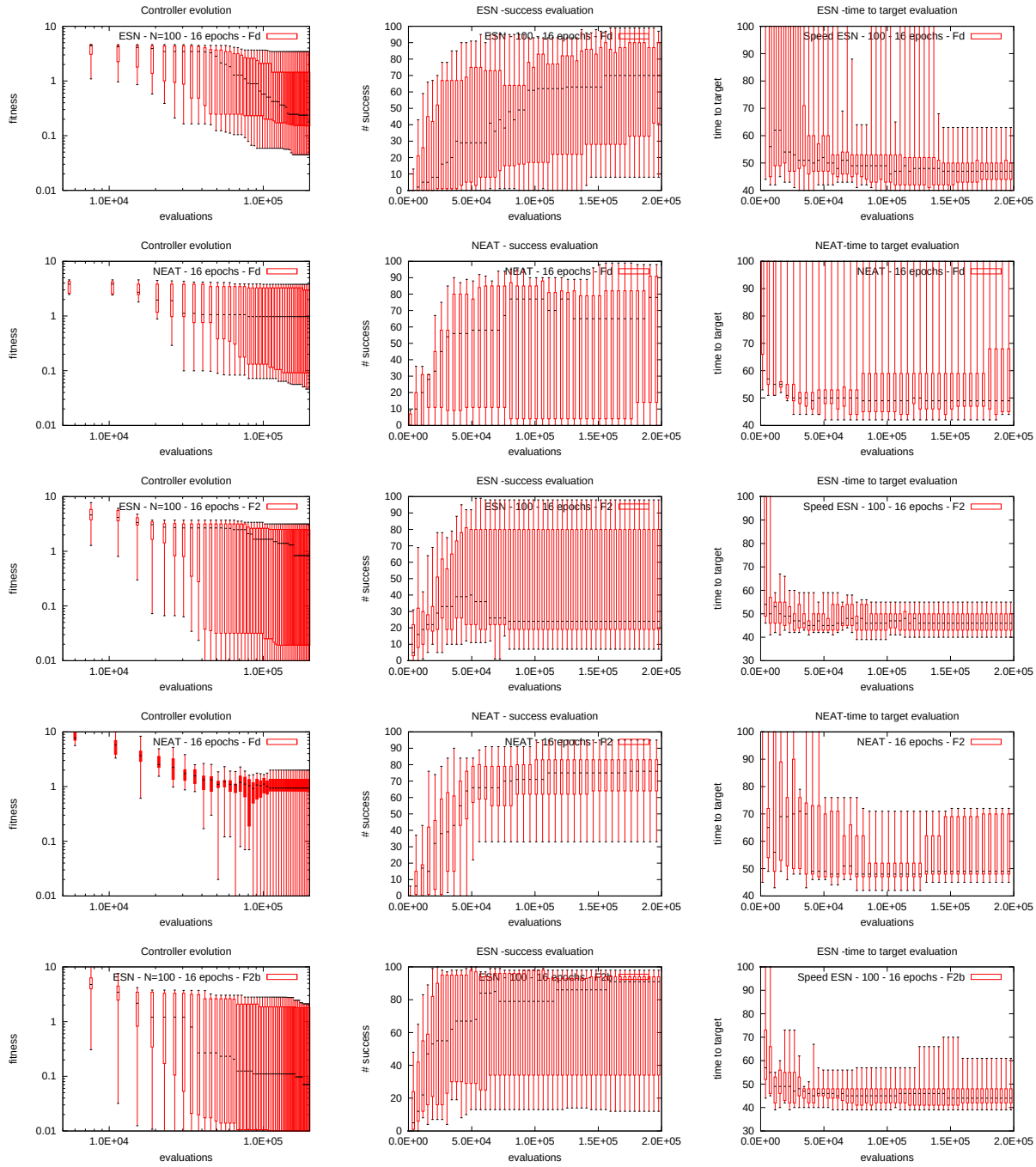

Fig. 6. From left to right : logscale controller evolution median and quartiles, target reach success over 100 trials and overall time to target, from top to bottom : ESN then NEAT using F_d fitness, ESN then NEAT using F_i fitness and finally ESN using $F_{vickrey}$ fitness.

networks,," in *Neural Computation, 19:1*. MIT Press, 2007, pp. 111–138.

[35] M. Salmen and P. Ploger, "Echo state networks used for motor control," in *In Proceedings of the 2005 IEEE international conference on robotics and automation*, 2005, pp. 1953–1958.

[36] C. Igel, "Neuroevolution for reinforcement learning using evolution strategies," in *Congress on Evolutionary Computation*, R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds., vol. 4, Sanya, China, December 2003, pp. 2588–2595.

[37] C. Hartland and N. Bredche, "Using echo state networks for robot navigation behavior acquisition," in *IEEE International Conference on Robotics and Biomimetics (ROBIO07)*. Sanya, China: IEEE Computer Society Press, December 2007.

[38] A. Billard and R. Siegwart, "Special issue on robot learning from demonstration," *Robotics And Autonomous Systems*, vol. 47, no. 2-3, pp. 65–67, 2004.

[39] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *In Proc. ICML*. ACM Press, 2004.