A Modular Neurocontroller for Creative Mobile Autonomous Robots Learning by Temporal Difference

Helmut A. Mayer
Department of Scientific Computing
University of Salzburg
A–5020 Salzburg, Austria
helmut@cosy.sbg.ac.at

Abstract – *One of the most prominent research goals in the field of mobile autonomous robots is to create robots that are able to adapt to new environments, i.e., the robots should be able to learn during their "lifetime" possibly without (or a minimum) of human intervention. When employing artificial neural networks (ANNs) to control the robot, reinforcement learning (RL) techniques are a good candidate for achieving continuous on–line learning. A problem with RL applied to robot learning is that the state (and action) space of a robot is typically not discrete. Thus, the robot had to evaluate an infinite number of possible actions at every time step in order to select the best. To overcome this problem we add a second network module to the neurocontroller acting as a memory of previous decisions (state–action pairs) of the robot. The robot's actual decisions, then, are based on previous decisions retrieved from memory. Additionally, intrinsic noise in the memory network gives the robot the possibility to evaluate new "ideas", hence it becomes creative. We analyze the potential of the above approach by measuring the ability of (simulated) robots to learn simple tasks using temporal difference (TD) learning.*

Keywords: Soft Computing, Mobile Autonomous Robots, Temporal Difference Learning, Modular Neurocontrollers, Creativity Machines, Robot Simulators.

1 Introduction

Today, a variety of control techniques are implemented for mobile autonomous robots, e.g., fuzzy control, machine learning systems, or artificial neural networks. Some of these systems have reached an impressive level of performance, however, most of these systems are not adaptive in the sense that they cannot change their behavior by feedback from the environment. Although, a human observer of these systems might get the impression that the robots can adapt to different situations, they can only adapt to situations the control program is aware of in advance. Even with evolutionary approaches to the construction of robotic neurocontrollers[1], learning is taking place in a generational time frame (*phylogenetic* learning), but not during the "lifetime" of a robot (*ontogenetic* learning).

Obviously, the main problem of (most) current control systems is that they cannot "reprogram" themselves during their exploration of the environment. This static behavior of an artificial structure is the most fundamental difference to *Biological Neural Networks* (BNNs) exhibiting highly dynamic properties not only throughout their lifetime, but also within very short time spans of activity [2].

A biologically plausible method to achieve a combination of phylogenetic and ontogenetic learning (as seen in nature) are evolved network structures, whose parameters are altered by *Artificial Neuromodulators* (ANMs) [3]. The ANMs influence learning by defining the type of *Hebb* learning based on the combination of modulators received by each neuron [3]. However, our attempt to utilize arbitrary network structures with well–defined learning reactions (triggered by ANMs) showed that the low–level, unsupervised Hebb learning cannot be linked to high–level concepts the robot should learn, when using a standard sensor–motor architecture [4].

Dynamic changes in a neurocontroller may be induced employing *Reinforcement Learning* (RL) techniques [5], i.e., the robot's brain (consequently, its behavior) is shaped during exploration of the environment. A popular RL method applicable to neurocontrollers is *Temporal Difference* (TD) learning. With this method the neurocontroller is not generating motor signals driven by sensor input, but evaluates potential (motor) actions. Learning is achieved by the difference of predictions in consecutive time steps (temporal difference) and scalar feedback signals (reward or punishment) from the envi-

ronment or a teacher.

The main problem associated with RL of robotic neurocontrollers is that the space of potential robot actions (and states) is infinite. In Q–Learning, where the values for all states (and actions) are explicitely stored and changed by reinforcement, the latter problem commands approximation of the value function [6]. With TD learning the values are represented implicitely by the function represented by the TD network being able to approximate (generalize). When evaluating potential actions, the robot has to restrict itself to a subset of potential actions. In a simple, but efficient approach a specific action subset is pre–defined by the human operator.

Even with a very large action subset another general problem of most robotic RL systems is the missing state–action memory, which arguably is the most important source of human behavior. If the robot remembered previous situations (states) and its corresponding behavior (actions) that has been proven successful, it could exploit this knowledge in current, similar situations. We realize such a memory by a second ANN in the neurocontroller, which additionally creates novel actions based on internal noise slightly altering the connection weights [7].

This idea is greatly inspired by *Creativity Machines*, more precisely, by the *Imagination Engine* (IE) presenting its output to a critic called *Alert Associative Center* (AAC), which has been trained to recognize, if the "idea" generated by the IE is consistent with general concepts of a problem domain. The IE generates its ideas by random alterations of a subset of the weights of a network trained with examples of the problem domain [8].

2 The Modular Neurocontroller

The structure and the functionality of the modular neurocontroller is schematically displayed in Figure 1.

Let us first describe the mechanisms associated with the evaluation module (TD network) learning by temporal difference. The TD network implements the on–policy method *SARSA* [5], where network input is not only the state of the robot, but a state–action pair. The state the robot is in is given by the current sensor signals (the five left inputs in Figure 1), while potential actions (two motor signals) are suggested by an action generation method. A basic approach to action generation is a pre–defined set of fixed actions [9]. In a single time step all available actions are sequentially presented to the evaluation network without changing the sensor input values (state). Each state–action pair is mapped to a value represented by the single output neuron. The selection of a specific action is performed according to the ϵ–greedy method [5], where the action with the highest value is selected, but with a small probability ϵ a random action is chosen instead. The network is trained with the TD(0)–algorithm [5] given by

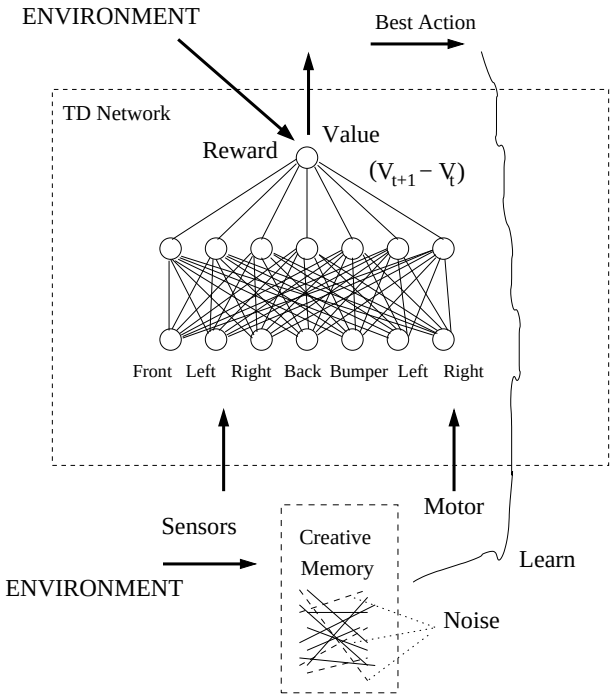

Figure 1: The modular neurocontroller with the evaluation module (TD network) and the creative memory.

$$\vec{w}_{t+1} = \vec{w}_t + \eta[r_{t+1} + \gamma V_{t+1} - V_t]\nabla_{\vec{w}_t} V_t, \qquad (1)$$

where $\vec{w}$ are the weights of the network, V is the value of the selected action, r is the reward, η is the learn rate, and γ is the discount factor. By multiplying the gradient of the value with the TD error (the term in square brackets), the value for the action selected at time step t is increased or decreased depending on the value of the best action at $t+1$. Thus, actions leading to higher values in subsequent time steps are reinforced, i.e., are trained to trigger a higher value themselves.

The creative memory (a feed-forward ANN) is a standard sensor–motor network employed to generate a set of potential actions, hence, implementing a more sophisticated action generation method. The current sensor signals are the input to the memory net generating a potential action (two motor signals) at the output. This procedure would always suggest a single action in any specific time step. By applying noise to a subset of the network weights different actions can be generated (keeping the sensor input fixed). However, these "ideas" are not purely random, but are slight deviations from the "main idea", which has been formed by training the memory network with previously selected state–action pairs. In each time step $t+1$ the memory is trained with the state (sensors) and selected action of the previous time step t utilizing standard back–propagation. As a consequence, the creative memory suggests actions based on past experience, hence the action space is sam-

pled in a promising region. The number of proposed actions is only limited by time considerations, as each perturbation of the memory creates a new (potential) action.

It should be explicitely mentioned that the reward signal is directly derived from the environment, i.e., reward is identical to specific sensor signals in the following experiments. This should more closely model biological systems, where often there is no explicit teacher giving reward, but is mediated by cognitive and emotional processes. E.g., when humans touch a hot piece of metal, they do not need a teacher to realize the painful sensations.

3 Experimental Setup

We perform experiments with two simple tasks executed in a Java simulator (cycle time $t_c = 0.1s$), namely, *Wall Avoidance*, where the robot should learn to stay away from the walls of a rectangular arena, and *Spot Finding*, where the robot is taught to move towards a circular spot in the arena, which can be smelled by the robot. The learning behavior is evaluated by a *Learn Ability* assessing the robot's behavior before and after a training session.

The cylindrical robot shown in Figure 2 is equipped with four distance sensors (front, back, left, right) and a contact sensor for the wall avoidance experiment, and a single nose sensor for spot finding (in Figure 2 all sensors are shown at once).

The distance sensor simulates a nonlinear, noise–free, real device measuring the reflection of a physical signal emitted exactly in direction of the line from robot center to the sensor positioned at the perimeter of the robot. The sensor signal s_d fed into the modular neurocontroller is given by

$$s_d = \frac{1}{(1+d)^{10}}, \tag{2}$$

where d is the distance from the sensor position to the nearest object (wall).

The simulated contact sensor generates a signal $s_c = -1$, if the robot collides with an object, otherwise $s_c = 0$. The negative value is commanded by the fact that the sensor signal is identical to the reward, which in case of wall avoidance must be negative (punishment) in order to teach the robot to stay away from the walls of the arena.

The nose sensor detects odor within a certain angular range given by the *Frustum Angle* α. The nose center is exactly in the forward direction of the robot, and the sensor signal s_n is scaled by the distance d and the angle β (between front direction and spot direction) to the center of the odorous spot according to

$$s'_n = o\ \frac{d}{r}(1 - \frac{2\beta}{\alpha}) \quad d \geq r, \beta \leq \frac{\alpha}{2} \quad s_n = \frac{s'_n}{1+s'_n} \tag{3}$$

with s'_n being the raw sensor signal, o the "amount of odor" (in *mol*), and r the radius of the odorous spot. If the nose is inside the spot ($d < r$), the robot smells the "full load" ($s'_n = o$). If the spot is outside the sensitivity range ($\beta > \frac{\alpha}{2}$), it does not smell anything ($s'_n = s_n = 0$).

Both networks in the modular neurocontroller are standard *One–Hidden Layer* networks (seven hidden neurons in the evaluation net, five in the memory net) composed of neurons with logistic activation function. Each sensor is associated with an input neuron of the evaluation network. Additionally, the potential actions (left and right motor signal) are fed into the latter network and evaluated by means of the activation of the single output neuron with an identity activation function. The reward is either identical to the contact sensor signal, or to the nose sensor signal for wall avoidance and spot finding, respectively. The learn rate $\eta = 0.01$, the discount factor $\gamma = 0.9$, and the parameter for the ϵ–greedy policy $\epsilon = 0.01$.

The creative memory receives the sensor signals as input (characterizing the robot's current state), and suggests corresponding motor actions (left and right motor) at the output. The motor values in the range of $[0.0, 1.0]$ generated by the network are linearly transformed to the range $[-0.5, 0.5]$ so as to cover both motor directions (negative values are backward). The creative memory's internal noise is imposed by randomly adding or subtracting a constant value σ to a fixed number of random weights of the memory net. As suggested in [7] a characteristic operational value of the internal noise level is given by the quantity $\frac{n\sigma}{N} = 0.19$ (n is the number of noisy connections, N the total number of connections). We applied a σ of 0.76 in all experiments resulting in noise in a quarter of all connections. Each time step the memory is trained with the state–action pair selected by the evaluation network in the previous time step. The back–propagation learn rate is given by the TD error of the current time step. Note that the TD error might be negative resulting in unlearning of a specific state–action pair. Also, a potential reward influences the magnitude of the learn rate, i.e., "good" actions are learnt more thoroughly.

In order to assess the potential of the creative memory, we compare it to two other basic action generating methods: a set of fixed, pre–defined actions, and a set of random actions. In the fixed action set we included the motor values -0.5, -0.25, 0.0, 0.25, and 0.5 for each motor resulting in 25 different actions (e.g., straight ahead with maximal speed is $[0.5, 0.5]$). Note that these actions cover all basic motions of the robot including standing still. The set of random actions simply contains 25 actions generated randomly anew in each simulation cycle. The same number of 25 potential actions is generated by the creative memory with the current state (sensor signals) of the robot as fixed input and successive perturbations of the network yielding action

"ideas" based on past experience.

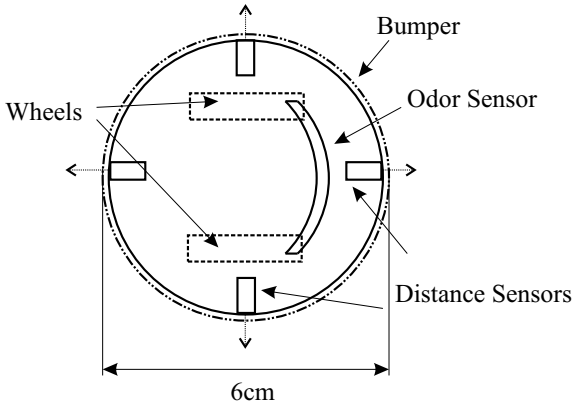

Figure 2: The cylindrical robot.

3.1 The Wall Avoidance Experiment

In this experiment the robot is placed in a rectangular arena (1.05 × 0.70 m) and should learn to avoid wall contact. We performed experiments with 500 simulated robots initialized with different random weights and biases from the interval $[-1.0, 1.0]$. The learning behavior is evaluated by the *Learn Ability* L calculated in the following way:

1. Every robot is placed into the upper left corner (in a distance of ten cm to the walls facing the corner) and then moves freely (without learning) for ten minutes. Then, the same procedure is repeated in the upper right corner. In these 20 minutes we measure the time t_{pre} it is in contact with the wall.
2. For the next two hours the robot with learning activated is placed at random positions (every five minutes) in the arena.
3. After learning the procedure described in 1 is repeated measuring the wall contact time t_{post}.

The learn ability L_{WA} is defined as

$$L_{WA} = \frac{t_{pre} - t_{post}}{t_{pre} + t_{post}}. \quad (4)$$

We compare the three different action generation methods using the mean learn ability $\overline{L}$ of the robots. Note that a number of robots never touch the wall, which we labelled *Genius*, as they perfectly master the task right from the time of "birth". Genius robots are not considered for calculation of the mean learn ability. The learn ability L is 1.0, if the robot has learned the task perfectly, e.g., never touches the wall after training. An $L > 0.0$ indicates an improvement after learning, while an $L < 0.0$ is the sign of a negative effect of training, i.e., the robot exhibits a worse behavior.

3.2 The Spot Finding Experiment

Again, the robot is placed into the rectangular arena, but this time it should learn to move towards a circular spot in the arena, which can be smelled by the robot. The only sensor is the nose sensor detecting odors with a frustum angle α of 90 degrees. The (virtual) odorous spot is a circle with $r = 6\ cm$ and $o = 10\ mol$.

The evaluation of the robots is performed as follows:

1. Each robot is placed in the upper left corner (ten cm to the walls facing the corner), while the spot is placed in the upper right corner (ten cm to the walls). During the next hour (without learning) we measure the time t_{pre} the robot is inside the spot.
2. In the next four hours learning is activated and every hour the spot is placed at a new random position.
3. Finally, learning is turned off, again, and the procedure described in 1 is repeated measuring the time inside the spot t_{post}.

As the measured times are now indicating wanted behavior, the learn ability $L_{SF} = -L_{WA}$ (Equation 4). Robots never moving inside the spot are labelled *Ignorants* and are not considered for calculation of the mean learn ability.

4 Results

The results of the wall avoidance (WA) experiments including the learn ability's mean $\overline{L}$ and standard deviation L_σ are shown in Table 1. Note that only the Creative method employs a second ANN in the neurocontroller, while the other methods only rely on the single evaluation network and deterministic (Fixed) or Random action generation.

Table 1: Learn abilities L of 500 wall avoiding robots using different action generation methods.

WA	Fixed	Random	Creative
$L = 1$	260	196	282
$L > 0$	12	215	65
$L = 0$	13	1	0
$L < 0$	19	51	80
(Genius)	196	37	73
$\overline{L}$	0.8232	0.7150	0.6109
L_σ	0.4959	0.5137	0.7068
$\overline{t}_{post,\%}$	0.0584	0.0160	0.0402

In this experiment the best learn ability is achieved by the Fixed method, however, there might be a simple explanation to this outcome. With all three compared methods a great majority of robots developed a spinning behavior, often with a slow movement towards the

center area of the arena. Only the Fixed method guarantees the availability of pure spinning (without lateral movement), as this action is contained five times in the action set (with different angular velocities including zero). The other methods are based on random effects, which may allow pure spinning in a single time step, but not in a long sequence of steps. This general observation is also backed up by the large number of genius robots with Fixed, as a "spinner by nature" never [1] receives a negative reward (being induced by wall contact).

On a similar note, the Random method yields few genius robots, as, though, the evaluation network might assign spinning the highest value, it will often not be present in the random action set. This also explains the large number of robots in the category $L > 0$ with Random. Many of these robots might have developed a perfect evaluation network, but occasionally in situations close to the wall, the correct "escape" move, e.g., turning, is not present in the action set, which forces the robot to select the best of the available actions, e.g., moving straight ahead.

It can also be seen that Creative is of less probabilistic nature than Random, as the number of genius robots and those in $L = 1$ is higher. The latter is even higher than with the Fixed method indicating that the creative memory is gradually acquiring spinning (or turning) behavior not present at the robot's "birth". The main problem with Creative is the rather large number of robots in $L < 0$. This might be attributed to *Catastrophic Interference* [10] in the memory, as many of these robots did not have any wall contact during pre–training. Forgetting of previous knowledge may also occur in the evaluation network, but with a second ANN (memory) in the neurocontroller this effect may be more pronounced.

The mean wall contact time percentage after training $\overline{t}_{post,\%}$ evaluated on all 500 robots sees Random in front of Creative, which can be explained by the non–deterministic nature of these methods. Even robots with a tendency to run into the walls frequently retreat from the wall due to random effects, which is not the case for Fixed.

The results of the spot finding (SF) experiment are presented in Table 2.

Generally, the mean learn abilities are much lower than in the wall avoidance experiment, which may mainly be attributed to the task being more complex. Also, the duration of the training session for a single robot (6 hours real time) might be too short, however, in special circumstances the training time may be even exceedingly long as outlined in the following.

The most striking result in Table 2 is the huge number of ignorant robots (those are never inside the spot during pre– and post–training, and are not considered for the mean learn ability). Again, this seems to be a consequence of the pure spinning behavior of many robots operating under the Fixed action generation method. Possible initial spinning (without lateral movement) is frequently rewarded, when the robot smells the spot while spinning, which reinforces spinning even more. Moreover, with all methods it could be observed that robots finding the spot initially follow the contours of the spot, but gradually change their behavior to pure spinning inside the spot, where this behavior is rewarded constantly for long periods of time. When robot and spot are separated into the two (distant) upper corners in post–training, many robots keep spinning, which is most pronounced with Fixed, as non–spinning actions are more distant (from spinning actions) in the fixed action space.

With Random a large number of ignorant robots move into the $L < 0$ category, which is mainly based on the fact that the random action sets let the robot explore the whole arena easier, accidentally moving inside the spot (even one time step is enough to no longer be counted as ignorant). Once trained to find the spot, spinning is also dominant for Random robots, and in post–training, though, the robots show lateral movement it is often not consistent towards the spot. Again, the problem is the mere random action set, which often may not contain the actions, which would be considered to be best by the evaluation network.

The creative memory method is comparable to the Fixed method in this experiment and allows much more robots into $L = 1$ than the other methods. As with wall avoidance this nice result comes at the cost of a large number of robots in $L < 0$. Still, this number is smaller than with Random confirming again, that Creative has a more deterministic flavor. Surprisingly, in pre–training quite a few robots move consistently towards the spot and stay in there (without learning). Most of these robots are later easily trained to find the spot, however due to the acquired spinning behavior their "flexibility" is greatly decreased, and in post–training many Creative robots move consistently, but very slowly towards the spot. Hence, it may take 20-30 minutes until

Table 2: Learn abilities L of 500 spot finding robots using different action generation methods.

SF	Fixed	Random	Creative
$L = 1$	22	20	109
$L > 0$	2	16	63
$L = 0$	1	0	1
$L < 0$	20	323	170
(Ignorant)	455	141	157
$\overline{L}$	0.0800	-0.8064	0.0045
L_σ	0.9681	0.5631	0.8829
$\overline{t}_{post,\%}$	0.0440	0.0296	0.2581

[1] Actually, the ϵ–greedy policy of action selection during training also introduces a small random component.

they reach the spot resulting in a reduced time inside the spot compared to pre–training, thus a negative learn ability. Most of the robots would move much faster and more directly to the spot, if training would be abandoned, when they start to circle around the spot (some also move straight through the spot turning immediately, when they leave the spot). In a very literal sense the robot becomes "saturated", when staying too long inside the spot. A larger memory network may reduce this effect, possibly being able to discern and suggest different basic behaviors inside and outside the spot.

The mean in–spot time percentage after training $\overline{t}_{post,\%}$ evaluated on all 500 robots shows that Creative enables the robots to find the spot much more consistently than the compared methods. With spot finding the robot cannot only rely on a single behavior like spinning, but has to combine different behaviors, namely locating, approaching, and staying inside the spot. To a certain degree the creative memory seems to enhance this ability, also promoted by the evaluation network.

5 Summary

We have presented experiments with mobile autonomous robots steered by neurocontrollers being trained on–line via reinforcement learning. We compared fixed, random, and creative action subset generation, the latter being implemented by a second ANN acting as a memory of actions the robot has chosen in past similar situations (states). The proposed modular neurocontroller enhances the creativity machine approach [8] with a dynamic network (shaped by temporal difference learning) as compared to the original, static (pre–trained) critic. The creative memory allows a sampling of the infinite action space based on the robot's previous experience. More importantly, it suggests novel actions (generated by noise in the creative memory), which are essential for adaptations of the robotic behavior to a changed environment. The experiments demonstrated the potential of the creative robots, specifically, with the more complex (but still simple) spot finding experiment. In future research some of the problems identified with the creative memory will be addressed, namely, catastrophic interference, and the strong interactions of different parameters. The "forgetting" of learned patterns as a consequence of training with new, different patterns is a problem common to artificial and natural systems, and could be reduced by more complex ANN architectures, e.g., recurrent associative networks. The aptitude of neurocontrollers for reinforcement learning could be increased by artificial evolution of the relevant parameters including learn rates, reward signals, policy parameters, sensor signals, and network structures. Essentially, we believe that the presented modular neurocontroller could be a step towards robots adapting to unknown environments in a human–like manner.

References

[1] Stefano Nolfi and Dario Floreano. *Evolutionary Robotics – The Biology, Intelligence, and Technology of Self–Organizing Machines*. MIT Press, 2000.

[2] G. M. Shepherd. *Neurobiology*. Oxford University Press, 3rd edition, 1994.

[3] Akio Ishiguro, Siji Tokura, Toshiyuki Kondo, Yoshiki Uchikawa, and Peter Eggenberger. Reduction of the Gap between Simulated and Real Environments in Evolutionary Robotics: A Dynamically–Rearranging Neural Network Approach. In *IEEE Systems, Man, and Cybernetics Conference*, pages III – 239–244. IEEE, October 1999.

[4] Helmut A. Mayer and Gerald Wiesbauer. Dynamic Regulation of Hebb Learning by Artificial Neuromodulators in Mobile Autonomous Robots. In *IEEE International Conference on Systems, Man & Cybernetics*, pages 2107–2112. IEEE, October 2003.

[5] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[6] William D. Smart and Leslie Pack Kaelbling. Effective Reinforcement Learning for Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002.

[7] Steve L. Thaler. Is Neuronal Chaos the Source of Stream of Consciousness? In *World Congress on Neural Networks*, pages 1255–1258, Mahwah, NJ, 1996. International Neural Network Society, Lawrence Erlbaum Associates.

[8] Steve L. Thaler. A Proposed Symbolism for Network–Implemented Discovery Processes. In *World Congress on Neural Networks*, pages 1265–1268, Mahwah, NJ, 1996. International Neural Network Society, Lawrence Erlbaum Associates.

[9] Martin Riedmiller and Barbara Janusz. Using Neural Reinforcement Controllers in Robotics. In *Proceedings of the 8th Australian Conference on Artificial Intelligence*, pages 491–496, Singapore, 1995. World Scientific Publishing.

[10] Bernard Ans, Stéphane Rousset, Robert M. French, and Serban Musca. Preventing Catastrophic Interference in Multiple–Sequence Learning Using Coupled Reverberating Elman Networks. In *Proceedings of the 24th Annual Meeting of the Cognitive Science Society*. Lawrence Erlbaum Associates, August 2002.