ORNL/CP-96046

Article Submitted to
NEURAP'98
Marseille, France

CONF-980338--

March 1998

Automatic Tuning of the Reinforcement Function*

Claude Touzet
Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6355

and

Juan Miguel Santos
Universidad de Buenos Aires
Cdad. Universitaria Depto. Computación
1428 Bs As, Argentina

*Research supported in part by U.S. Department of Energy, Office of Energy Research, Basic Energy Sciences Research Program under contract DE-AC05-96OR22464 with Lockheed Martin Energy Research Corporation.

DISCLAIMER

Submitted to NEURAP'98, Marseille, France, March 1998.

Automatic Tuning of the Reinforcement Function

Claude Touzet[1] (*) and **Juan Miguel Santos** (#)
(*) Center for Engineering Systems Advanced Research, Oak Ridge National Laboratory,
Oak Ridge, TN 37831-6355, USA.
(#)Universidad de Buenos Aires, Cdad. Universitaria Depto. Computación, 1428 Bs As, Argentina.

Abstract: The aim of this work is to present a method that helps tuning the reinforcement function parameters in a reinforcement learning approach. Since the proposal of neural-based implementations for the reinforcement learning paradigm (which reduced learning time and memory requirements to realistic values) reinforcement functions have become the critical components. Using a general definition for reinforcement functions, we solve, in a particular case, the so-called exploration versus exploitation dilemma through the careful computation of the RF parameter values. We propose an algorithm to compute, during the exploration part of the learning phase, an estimate for the parameter values. Experiments with the mobile robot Nomad 200 validate our proposals.

Key words: Reinforcement learning, adaptive robotics, autonomous robot, reinforcement function, behavior-based approach.

1. Introduction

Reinforcement Learning (RL) is a machine learning technique that allows a robot to learn complex behaviors using only a performance measure of the desired behavior [Kaelbling, 1996]. Models of the environment or of the desired behaviors are not required. The reinforcement signal provided by the reinforcement function (RF) evaluates the entered situation relative to the desired behavior. There has been a lot of research related to the issues of convergence [Dayan, 1994]; generalization [Mahadevan, 1992]; exploration [Thrun, 1992], [Zhang, 1996]; and memorization [Lin, 1992]. A major milestone in the course of RF development has been the proposal of neural-based implementations, which reduced learning time and memory requirements to realistic values, allowing true RL applications [Sehad, 1994]. Therefore, the RF has become the critical component. Despite all the ongoing research, there have been few efforts (if any) to propose a methodology or define a robust set of heuristics for the design of RFs. Authors usually report the building of the RF as an emergent process involving lots of trials and errors [Santos, 1997].

In section 2, we introduce a general definition for a RF. In section 3, we propose an algorithm to compute during the learning phase the parameter values. In section 4, simulations and experiments involving the synthesis of avoidance obstacles are conducted with a mobile robot (Nomad 200). Finally, we discuss and conclude the obtained results and future work.

---

[1] This research was funded in part by the Office of Energy Research, Basic Energy Sciences of the U.S. Department of Energy, under contract No. DE-AC05-96OR22464 with Lockheed Martin Energy Research Corporation.

2. A Definition of the Reinforcement Function (RF)

It has been reported in [Touzet, 1997a] that a RF should always imply positive, negative, and null rewards[2]. Thus, a desired ratio of positive and negative rewards over time during the exploratory part of the learning phase is mandatory. Let us consider the following RF expression, which has only one parameter per case (positive and negative rewards):

$$RF(s_1,...,s_u) = \begin{cases} +1 \ if \ g_1(s_1,...,s_u) > \theta_+ \\ -1 \ if \ g_2(s_1,...,s_u) < \theta_- \\ 0 \quad in\ other\ case \end{cases}$$

where $(s_1,...,s_u)$ is the output readings of the sensors.

For the need of illustration, we take the case of an obstacle avoidance behavior synthesis for a mobile robot. How should we define the value of θ_+ (res. θ_-) so that it respects the desired ratios? As shown on figure 1, the sum of the rewards over the number of moves value varies between 0.0 and 0.7. If θ_+ is small, the robot receives lots of positive rewards, and then the sum of the rewards will be large. The extreme case is when θ_+ is 0. As we increase θ_+, the total amount of positive rewards diminishes. It must not reach 1000; otherwise, the robot would never receive positive rewards. The same reasoning applies to θ_-.

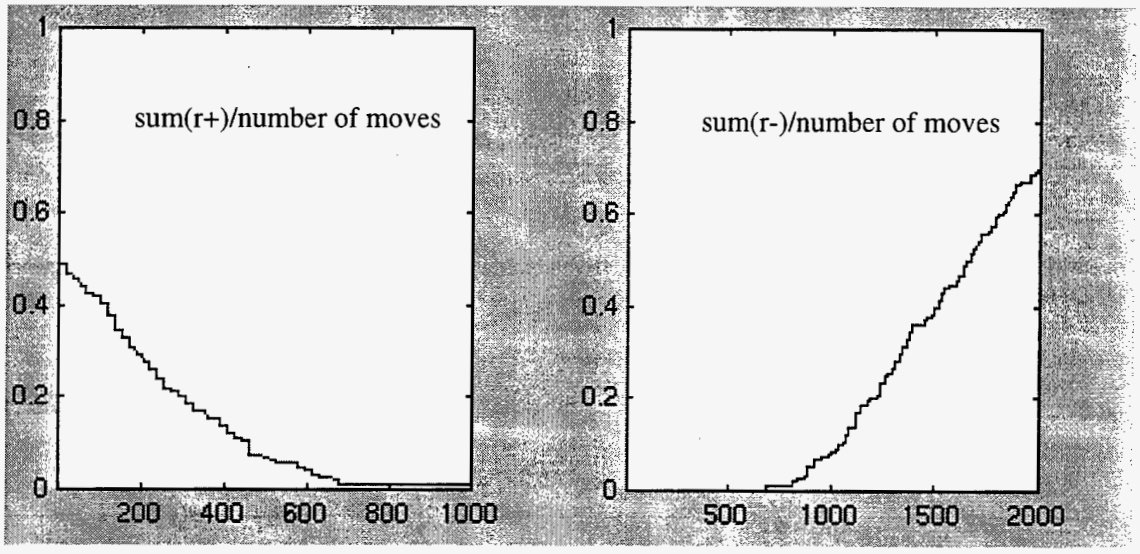

Figure 1. Each point of the graphs was obtained by performing 500 moves at random (using a random action generator for the mobile robot):
a/ Sum of positive rewards over the number of moves versus the value of θ_+ [0 .. 1000].
b/ Sum of negative rewards over the number of moves versus the value of θ_- [0 .. 2000].

[2] If there is no positive reward, the evaluation function built during the learning phase will have "0" as maximum value and the policy cannot select effective actions. If there is no negative reward, the robot can remain in a dead-end situation forever. If there is no null reward, the evaluation function will be non-continuous at the frontier between positive and negative situation-action pairs.

3. Dynamic Estimation of Parameters in Reinforcement Function

We have derived an algorithm to dynamically compute during the exploration part of the learning phase the parameter values. Let us assume that the functions $g_1(\,)$ and $g_2(\,)$ are non-linear but monotonous (here decreasing for $g_1(\,)$ and increasing for $g_2(\,)$ as shown on fig. 1).

Let us call p_+ and $p\_$ the desired values (ideal ratios) for the observers $\frac{\#r_+}{t}$ and $\frac{\#r\_}{t}$. The problem is to adjust dynamically the parameters θ_+ and $\theta\_$ so that we observe the following convergences $\frac{\#r_+}{t} \to p_+$ and $\frac{\#r\_}{t} \to p\_$.

Initialization:
Define ε, Δt k, k_1 *and* k_2 $(0 < k_1, k_2 < 1)$,
Define the values of p_+ *and* $p\_$,
Choose some initial value for θ_+ *and* $\theta\_$,
Set the initial step for $\Delta\theta_+$ *and* $\Delta\theta\_$,

1. Compute $\#r_+^{\Delta t}$ and $\#r\_^{\Delta t}$ as

$$\#r_+^{\Delta t} = \sum_{i=t-\Delta t}^{t} \delta(r^t, 1)$$

$$\#r\_^{\Delta t} = \sum_{i=t-\Delta t}^{t} \delta(r^t, -1)$$

where δ is the Kronecker function.

2. If $(\Delta r_+ - p_+)\left(\frac{\#r_+^{\Delta t}}{\Delta t} - p_+\right) < 0$ then $\Delta\theta_+ = k_1 \Delta\theta_+$

If $(\Delta r\_ - p\_)\left(\frac{\#r\_^{\Delta t}}{\Delta t} - p\_\right) < 0$ then $\Delta\theta\_ = k_2 \Delta\theta\_$

3. Set $\theta_+ = \theta_+ + sign\left(\frac{\#r_+^{\Delta t}}{\Delta t} - p_+\right)\Delta\theta_+$

Set $\theta\_ = \theta\_ - sign\left(\frac{\#r\_^{\Delta t}}{\Delta t} - p\_\right)\Delta\theta\_$

5. Store in Δr_+ and $\Delta r\_$ the actual values of observer estimators,

$$\Delta r_{+=} \frac{\#r_+^{\Delta t}}{\Delta t} \; ; \; \Delta r_{-=} \frac{\#r\_^{\Delta t}}{\Delta t}$$

6. If not terminate, then go to 1.

Figure 2. Update Parameters Algorithm.

The principle of the Update Parameters Algorithm (UPA) as described in figure 2 is to estimate the value of the observers $\frac{\#r_+}{t}$ and $\frac{\#r_-}{t}$. To this end, two variables $\#r_+^{\Delta t}$ and $\#r_-^{\Delta t}$ are defined. They are the number of positive and negative rewards occurring in the last Δt iterations (Δt is an integer value).

If the value of the estimator $\frac{\#r_+^{\Delta t}}{\Delta t}$ is greater than p_+, then we have to change the value of θ_+ to obtain in the next iterations a value of $\frac{\#r_+^{\Delta t}}{\Delta t}$ closer to the desired value p_+. Since the relation ($g_1(\ )$) between $\#r_+$ and θ_+ is monotonously decreasing, a positive value ($\Delta\theta_+$) is added to θ_+. In the opposite case ($\frac{\#r_+^{\Delta t}}{\Delta t}$ is smaller than p_+), then a negative value ($\Delta\theta_+$) is substracted to θ_+.

If the value of the estimator $\frac{\#r_-^{\Delta t}}{\Delta t}$ is greater than p_-, then we have to change the value of θ_- to obtain in the next iterations a value of $\frac{\#r_-^{\Delta t}}{\Delta t}$ closer to the desired value p_-. Since the relation ($g_2(\ )$) between $\#r_-$ and θ_- is monotonously increasing, a negative value ($\Delta\theta_-$) is substracted to θ_-. In the opposite case ($\frac{\#r_-^{\Delta t}}{\Delta t}$ is smaller than p_-), then a positive value ($\Delta\theta_-$) is added to θ_-. Each iteration, the values of $\#r_+^{\Delta t}$ and $\#r_-^{\Delta t}$ are updated using the last Δt values of $\#r_+$ and $\#r_-$ (each reinforcement signal at time t is noted as r^t).

We may have crossed the desired value for the parameter, in which case the product $\left(\Delta r_+ - p_+\right)\left(\frac{\#r_+^{\Delta t}}{\Delta t} - p_+\right)$ is negative (as also $\left(\Delta r_- - p_-\right)\left(\frac{\#r_-^{\Delta t}}{\Delta t} - p_-\right)$). In this case, the absolute value of $\Delta\theta_+$ (or/and $\Delta\theta_-$) is decreased by a factor k_1 (or k_2).

The stop conditions are given by:

i) $\left|\frac{\#r_+^{\Delta t}}{\Delta t} - p_+\right| < \varepsilon$ and $\left|\frac{\#r_-^{\Delta t}}{\Delta t} - p_-\right| < \varepsilon$ during k iterations
(ε is a small positive real number), or

ii) if the exploration part has finished.

4. Experiments

We have used the Nomad 200 (figure 3) robot in an experiment of synthesizing an obstacle avoidance behavior using its IR sensors. The typical arena is shown on figure 4. The robot has 16 sensors uniformly distributed around its body. The value returned by an IR sensor is 255 when there is nothing in front and 0 with a nearby obstacle (however, each value is coded on only 4 bits).

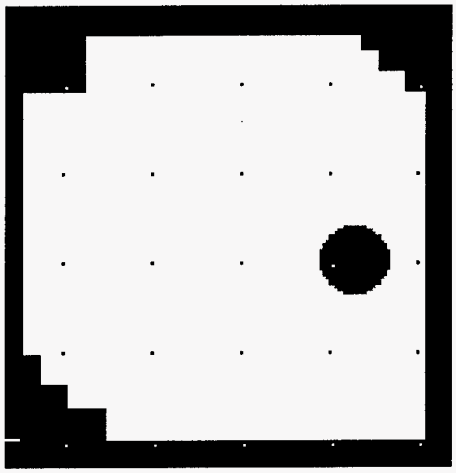

Figure 4. One of the arenas used in our experiments. The Nomad 200 robot is represented inside the arena.

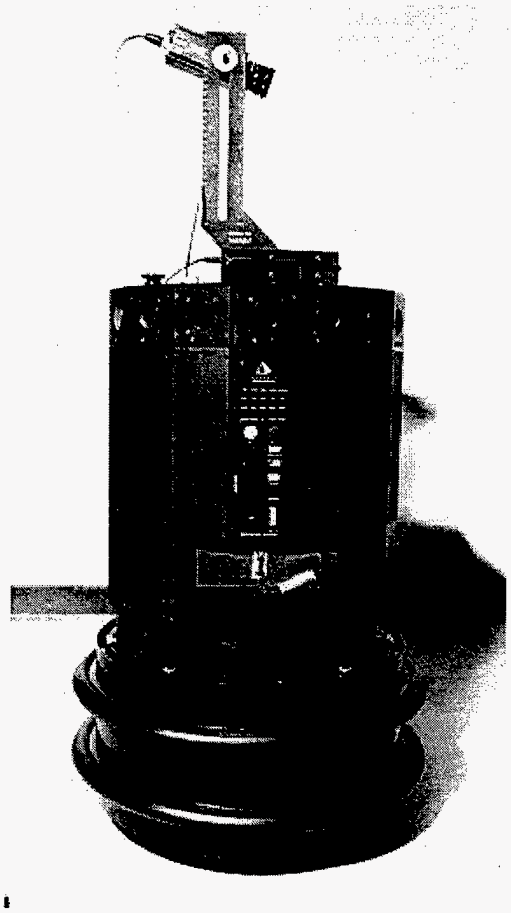

Figure 3. The Nomad 200 mobile robot.

The complete definition of the RF function is:

+1 if it is avoiding,
-1 if a collision occurs,
or 0 otherwise

The robot is avoiding when the current sum of sensor values[3] is greater than the previous one, the difference being greater than θ_+ and a collision occurs when the sum of the 8 front sensors is lesser than θ_-.This RF can be rewritten as:

$$RF((s_1,\ldots,s_{16})^t,(s_1,\ldots,s_{16})^{t-1}) = \begin{cases} +1 \text{ if } g_1((s_1,\ldots,s_{16})^t,(s_1,\ldots,s_{16})^{t-1}) > \theta_+ \\ -1 \text{ if } g_2(s_1,\ldots,s_{16}) < \theta_- \\ 0 \quad \text{in other case} \end{cases}$$

[3] The sensor values measure the distance to the obstacles (the greater the value, the farther the obstacles).

where $g_1((s_1,...,s_{16})^t,(s_1,...,s_{16})^{t-1}) = (g_2(s_1,...,s_{16})^t + \sum_{i=7}^{9} s_i^t) - (g_2(s_1,...,s_{16})^{t-1} + \sum_{i=7}^{9} s_i^{t-1})$

$g_2(s_1,...,s_{16})^t = \sum_{i=1}^{4} s_i^t + \sum_{i=13}^{16} s_i^t$

Figure 5 shows the learning curve for θ_+ and θ_- during the exploration part when UPA is active. The initial values for θ_+ and θ_- have been set to 0.0 and 1000.0, $k_1 = k_2 = 0.99$, $\Delta t = 20$, $\Delta\theta_+ = \Delta\theta_- = 1.0$. The desired ratio of r+ and r- to respect is 0.2 (for both). At the end of the UPA phase, the values for θ_+ and θ_- are 390.0 and 1150.0 (respectively). The behavior of r+ and r- during the 500 UPA iterations is displayed on figure 6.

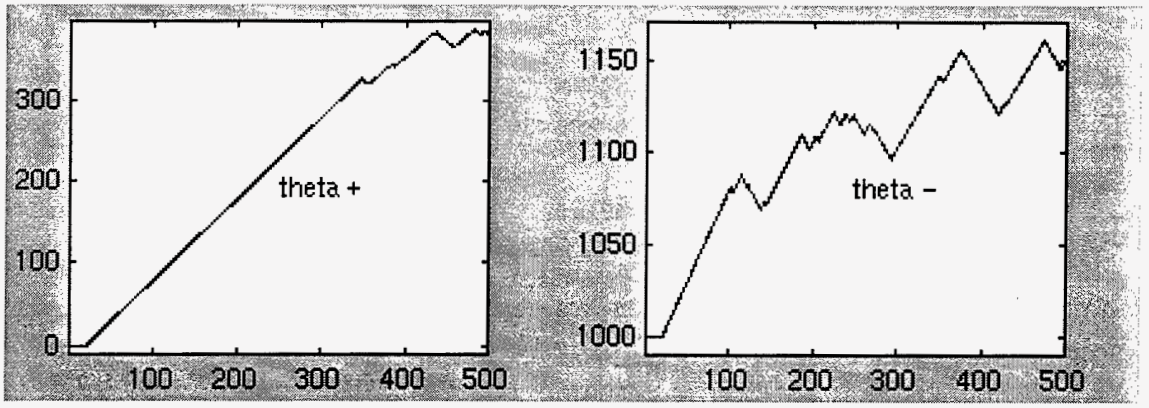

Figure 5. θ_+ and θ_- values during the 500 iterations of the UPA phase. The initial values are set randomly to 0.0 and 1000.0, respectively.

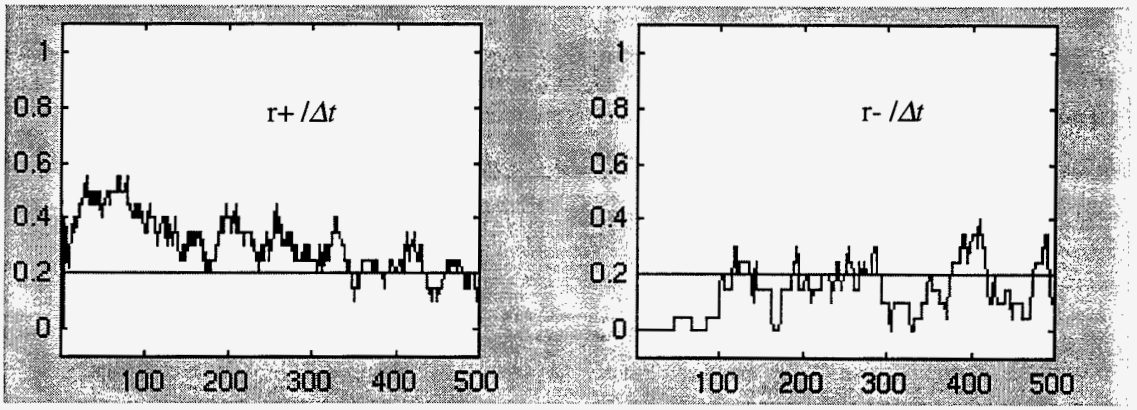

Figure 6. Values of r+/Δt and r-/Δt during the 500 UPA iterations. The oscillations are due to the limited size of the integration window (here Δt = 20 time steps) and the non-uniform distribution of the encountered situations. The objective is a ratio of 0.2 for r+ and r-.

After the 500 UPA iterations, Q-learning started the pure synthesis of the behavior using a self-organizing map implementation for memorizing and generalizing [Touzet, 1997b]. There were 16 neurons, a neighborhood of 4 and 18 inputs (16 IR sensor values, 1 action value, 1

Qvalue). Figure 7 shows the evolution of r+/ *t* and r-/ *t* during the experiments. During the initial 200 iterations, we can see the increase of r+/ *t* and the corresponding diminution of r-/ *t*, representative of a good learning phase. The remaining 100 iterations are used to verify the level of performance (test phase).

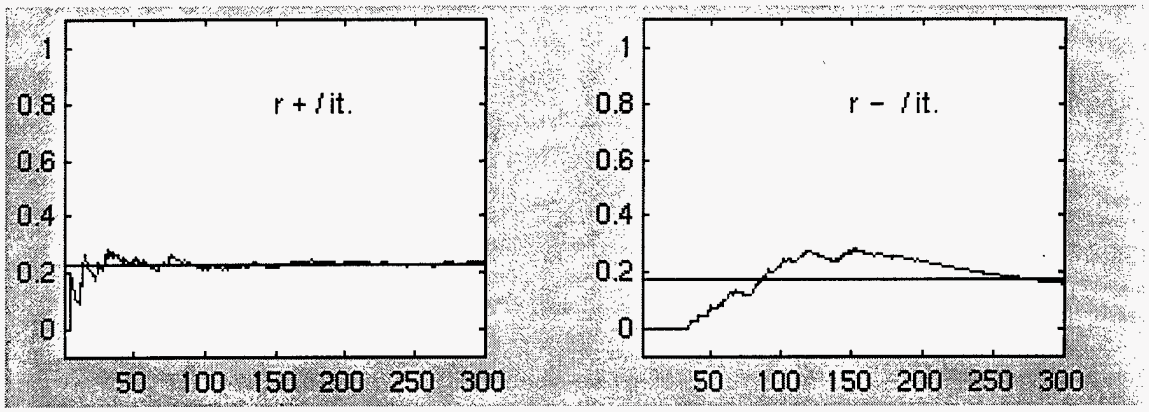

Figure 7. Graphs of r+/ *t* and r-/ *t* during the 200 learning + 100 test iterations.

Figure 8 displays an "objective" measure of the performance of the behavior: The distance to the obstacles. The only combination allowing a distance to the obstacles greater than a random move selection behavior corresponds to the threshold configuration given by the UPA, meaning that the used RF parameters (θ_+ and θ_-) are accurate for this task. There are 200 learning iterations, after which there is no more learning for the last 100 iterations. We see that the learned behaviors avoid obstacles from a greater distance.

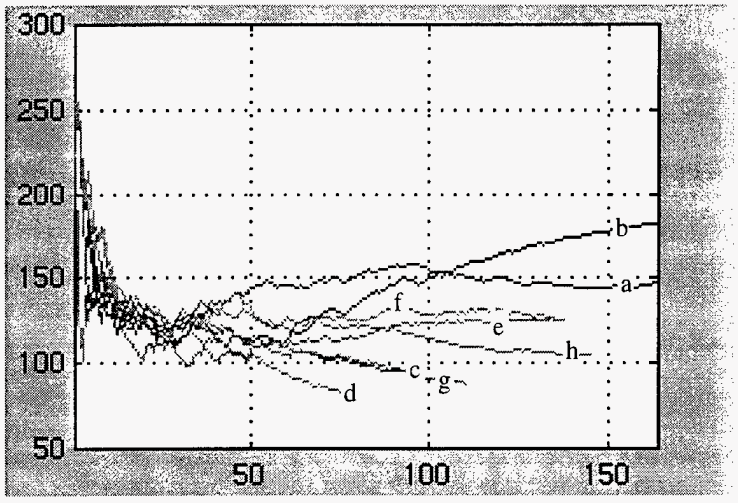

Figure 8. Distance to the obstacles in respect to the number of obstacles encountered during the 300 learning iterations.
(a) is obtained by a random move selection behavior (no learning involved),
(b) is the learned behavior corresponding to the values given by UPA ($\theta+ = 390$, $\theta- = 1150$),
(c) corresponds to ($\theta+ = 195$, $\theta- = 2300$), (d) corresponds to ($\theta+ = 780$, $\theta- = 575$),
(e) corresponds to ($\theta+ = 195$, $\theta- = 1150$), (f) corresponds to ($\theta+ = 780$, $\theta- = 1150$),
(h) corresponds to ($\theta+ = 390$, $\theta- = 2300$) , (i) corresponds to ($\theta+ = 390$, $\theta- = 575$).

5. Conclusion

In this paper, we propose to solve in a particular case the so-called exploration/exploitation problem. Based on the assumption that the relation between the parameters and the rewards is monotonous (it can be non-linear), the proposed Update Parameter Algorithm (UPA)allows one to compute during a pure exploration part an estimation of the RF parameter values. We test the tuned parameters using a mobile Nomad 200 robot on a task of synthesis of an obstacle avoidance behavior. The performance of the learning was evaluated along two indexes: Ratio of positive and negative rewards over time (r+ and r-) and distance to the obstacles. Self-organizing maps were used to implement the RL. Several experiments have been made with values of the parameters in the RF definition both close to or far from those obtained with the UPA. The results showed the validity of the RF parameters obtained, suggesting that further experiments are certainly needed to really measure the impact of these first steps in RF design.

The UPA use has been restricted here to a pure (without learning) exploration phase. It would certainly be interesting to be able to use the UPA during the learning phase as well. However, the non-uniform distribution of the rewards during the learning phase imposes modifications of the algorithm previously described. It is our desire to address this issue in the near future.

References

1. Leslie Pack Kaelbling, Michael L. Littman, Andrew W. Moore, "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, 4(1996), 237-285.
2. P. Dayan and T. Sejnowski, "TD(l) Convergences with Probability 1," *Machine Learning*, 14(3), 1994.
3. Sridhar Mahadevan and J. Connell, "Automatic Programming of Behavior-based Robots using Reinforcement Learning," *Artificial Intelligence* **55**, 311-365, 1992.
4. S.B. Thrun, "Efficient Exploration In Reinforcement Learning," *Technical Report CMU-CS-92-102,* Carnegie-Mellon University, 1992
5. Ping Zhang and Stéphane Canu, "Indirect Adaptive Exploration in Entropy-based Reinforcement Learning," *Proceedings of ICANN'96*, 1996.
6. L.-J. Lin and T. M. Mitchell, "Memory Approaches to Reinforcement Learning in Non-Markovian Domains," *Technical Report CMU-CS-92-138,* Carnegie Mellon University, School of Computer Science, 1992.
7. Samira Sehad and Claude Touzet, "Reinforcement Learning and Neural Reinforcement Learning," in *Proc. of ESANN,* Bruxelles, 1994.
8. Juan Miguel Santos and Claude Touzet, "First Results in Reinforcement Function Design," submitted to *Machine Learning and Autonomous Robots, Special Issue on Learning in Autonomous Robots.*
9. Claude Touzet and Sandip Sen, "Learning Agents," Invited paper *Autonomous Agents'97*, Marina del Rey, Los Angeles, CA, USA, February 1997a.
10. Claude Touzet, "Neural Reinforcement Learning for Behaviour Synthesis," to appear in *Robotics and Autonomous Systems*, Special issue on Learning Robot: the New Wave, N. Sharkey Guest Editor, 1997b.

Report Number (14) ORNL/CP--96046
CONF-980338--

Publ. Date (11) 199712
Sponsor Code (18) DOE/ER, XF
JC Category (19) UC-400, DOE/ER

DOE