computing@tanet.edu.te.ua
www.tanet.edu.te.ua/computing

ISSN 1727-6209
International Journal
of Computing

LEARNING FROM THE ENVIRONMENT WITH A UNIVERSAL REINFORCEMENT FUNCTION

Diego Ariel Bendersky, Juan Miguel Santos

Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires.
Pabellón I, Ciudad Universitaria
1428 Ciudad de Buenos Aires, Argentina.
{dbenders, jmsantos}@dc.uba.ar

Abstract: *Traditionally, in Reinforcement Learning, the specification of the task is contained in the reinforcement function (RF), and each new task requires the definition of a new RF. But in the nature, explicit reward signals are limited, and the characteristics of the environment affects not only "how" animals perform particular tasks, but also "what" skills an animal will develop during its life. In this work, we propose a novel use of Reinforcement Learning that consists in the learning of different abilities or skills, based on the characteristics of the environment, using a fixed and universal reinforcement function. We also show a method to build a RF for a skill using information from the optimal policy learned in a particular environment and we prove that this method is correct, i.e., the RF constructed in this way produces the same optimal policy.*

Keywords: *Reinforcement learning, environment influence, skills, autonomous robots.*

1. INTRODUCTION

In Reinforcement Learning (RL), an agent finds the optimal strategy for solving a particular task by interacting with the environment and receiving rewards and punishments based on the executed actions. This type of learning has been studied in humans and animals since the beginning of the 20th century [1], modeled mathematically using dynamic programming tools and adopted as an Artificial Intelligence method for machine learning [2].

Traditionally, the specification of the task is contained exclusively in the function that models rewards and punishments, called the reinforcement function (RF). Hence, each new learning task requires the specification of a new RF and most of the times this RF is built from scratch, based on the intuition and experience of the developer and tested by trial and error on realistic environments.

But in the nature, we can observe that explicit reward signals are limited, and external stimuli influence the behaviors of animals and humans [3] to the extent that it can affects not only *how* animals perform particular tasks, but also *what* skills an animal will develop during its life. For example, in laboratory experiments, a rat can learn how to pull a knob if this action opens a box with food. But the same rat can learn how to escape from a maze if it is put inside the maze and the food is put on the outside. In both cases, the reward (the positive reinforcement), expressed as a satisfaction felling, is obtained when the rat eat the food and not when the rat succeed to pull the knob or succeed to find the way out of the maze. In this example, it is the environment and not the RF which induces the skills that are going to be learned.

Another fact observation related to RL as seen in the nature is the use of information from past experience as a replacement for an explicit RF. This fact may be observed on humans and animals, who after the successful learning of a particular task, can construct new reinforcement functions and use it later in another task. The learning of these new tasks can then be produced without explicit external feedback. For example, humans associate reinforcements with approval or disapproval of other persons, with love and hate, or simple with a ``Right!" or ``Wrong!" yell [4]. This new type of reinforcements, often called *secondary reinforcements* or *conditioned reinforcements*, has been first identified by Ivan Pavlov in his experiments with animals.

Although these ideas have been studied by psychologists and biologists, as far as we know they have never been used for machine learning. Our aim is to incorporate them in an RL framework in a systematic and formal manner in order to develop a robust learning method less dependent to external

specifications.

In this work, we propose a novel use of RL that consist in the learning of different abilities or *skills*, based on the characteristics of the environment, using the same fixed and universal RF for all the skills. We also show a method to construct a RF for a skill based on the optimal policy learned in a particular environment with our method. We illustrate our idea in a robot simulator, showing how the robot learns different skills when the learning process takes place in different environments.

2. HYPOTHESES AND PROOF OF CONCEPT

In this section, we propose two hypotheses and we describe a series of simple experiments as a proof of concept. The hypotheses express the ideas of learning skills influenced by the environment and building RFs internally:

Hypothesis 1: Using a fixed reinforcement function that specifies a general task, RL can be used to learn different skills by modifying the characteristics of the environment.

Hypothesis 2: Using a fixed reinforcement function that specifies a general task and a policy that solves the task on a particular environment, it is possible to construct a RF for a skill that is part of the optimal policy for the general task.

These hypotheses are related to problems of interest for the RL field, such as: environment generalization and optimal environment construction (if two environments allow the learning of a given task, which one is better? Could we obtain the optimal environment?) and mappings between behaviors and environments (which characteristics should be present in the environment to allow an agent to learn the desired skill?).

3. LEARNING FROM THE ENVIRONMENT

As a proof of concept for our first hypothesis, we designed a series of experiments and we carried out several simulation tests. On these experiments, a light represent a food source, and the general task consist in reaching the light from any initial position. The task is considered episodic, and an episode ends either when the light source or a boundary of the environment is reached. The robot has a light sensor that consists in a pair of values that indicate the distance and angle from the front of the robot to the light source and nine proximity sensors distributed around the robot body. This task is expressed with a simple RF that assigns a positive reinforcement if the robot reaches the light and a negative reinforcement if it is too far.

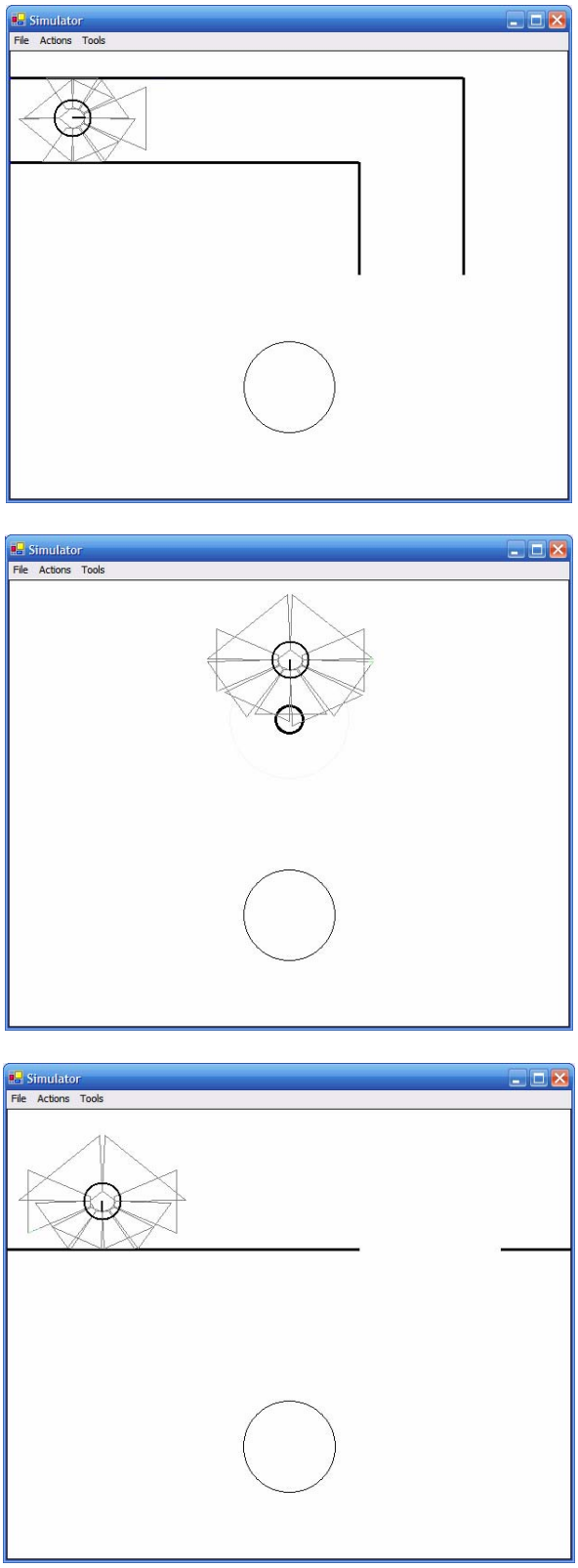
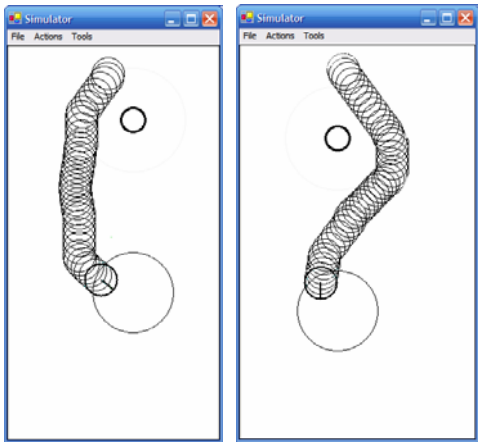
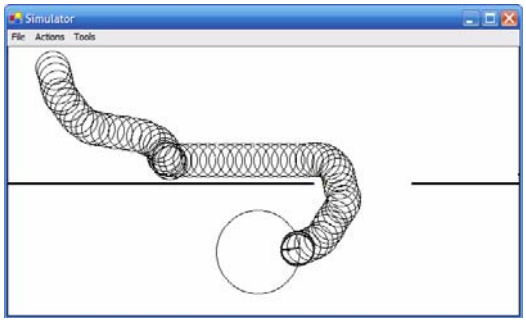
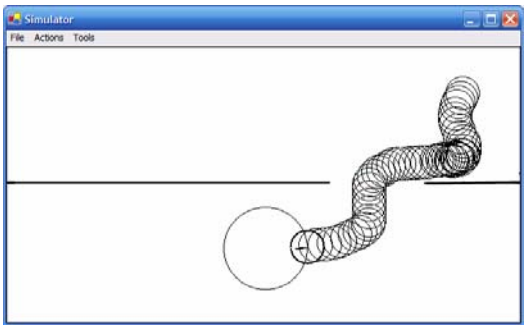


Fig. 1 – Environments used for the learning process. The big circle represents the light source, small circles are rounded obstacles, and straight lines are walls. The triangles around the robot represent the proximity sensors.

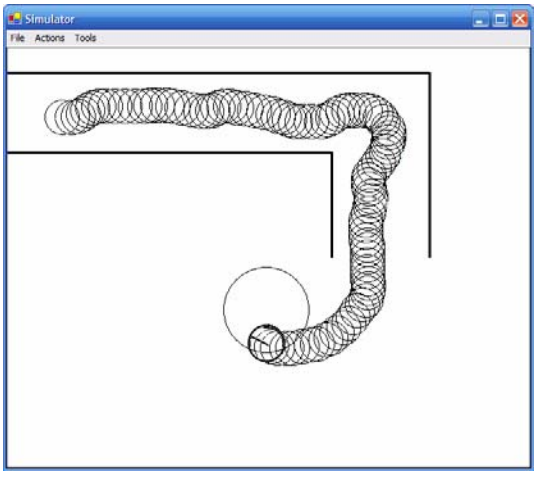


a)





b)



c)

Fig. 2 – Trajectories of the policies obtained on each environment: a) the robot avoids rounded obstacles, b) the robot finds a hole in a wall, and c) the robot walks through a corridor.

This RF can be expressed with the following formula:

$$rf(s) = \begin{cases} 100 & \text{if } light\_dist \le K_{min} \\ -1 & \text{if } light\_dist \ge K_{max} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where K_{min} and K_{max} are thresholds for distance to the light.

Using the same RL algorithm and the same parameters, we execute the learning process in three different environments: an environment with rounded obstacles, an environment with a wall and a hole on it and an environment with a corridor (see Figure 1). For all the experiments, we used the Q-Learning algorithm [5]. Figure 2 shows trajectories of the obtained behavior for each environment.

If we would have seen the results before reading the explanation of the experiment, we would have concluded that the robot on Figure 2.a knows how to avoid obstacles, the robot on Figure 2.b can find a hole in a wall and pass through it and the robot in Figure 2.c can traverse corridors. But there is *nothing* in the RF that determines these skills. A first question arises: Where does the information needed to learn the skill come from? Since the three experiments were set up exactly in the same way except for the definition of the environment, we can conclude that *the characteristics of the environment and the relationship between the RF and the environment implicitly contain the information needed to learn the skills*. If we give credit to the previous sentence, then it should be possible to extract this information, and make it explicit in the form of a specific RF for the learned skill.

4. EXTRACTING A RF FROM PAST EXPERIENCE

In this section we will show a method to extract a RF for a skill from a policy already learned in a particular environment.

Let $M = <S,A,T,R>$ be a Markov Decision Process that represent the global task (reach the light source in this case) for a particular environment, where S is the state set, A the action set, T a transition function and R the reinforcement function. Consider S_{skill}, the subset of S where the skill is expressed (notice that the skill does not cover the entire state space; for example, some parts of the environment are common to all the experiments and, conceptually, are not part of the skill) such that $R(s,a,s') = 0$ for all $s \in S_{skill}$. We define a new MDP for the skill $M' = <S',A',T',R'>$ where $S'=S_{skill} \cup \tau$, $A'=A$ and $T'(s,a,s') = T(f(s),a,f(s'))$, where f is defined as follows:

$$f: S \rightarrow S'$$

$$f(s) \begin{cases} s \text{ if } s \in S_{skill} \\ \tau \text{ if } s \notin S_{skill} \end{cases} \qquad (2)$$

A RF for a skill can be extracted from a learned policy if we can build R' such that $\pi^*_{M'}(s)$, the optimal policy for M' is equal to $\pi^*_M(s)$, the optimal policy for M, for all states $s \in S_{skill}$. If R' is defined as follows, we will show that then the previous property holds:

$$R'(s,a,s') \begin{cases} Q^*_M(s,a) & \text{if } s \in S_{skill} \text{ and } s' \notin S_{skill} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

where s is the previous state, a the executed action, s' the current state and $Q^*_M(s,a)$ the Q-value function for the optimal policy of M.

We will prove now that, for any policy π, $Q^\pi(s,a)$ in M (using R as the reinforcement function) is equal to $Q^\pi(s,a)$ in M' (using R') for all states $s \in S_{skill}$, for all actions a. From this, it follows immediately that the optimal policy for M is also the optimal policy for M' for states in S_{skill}.

Let $\Phi^\pi_{M\#}(s,a)$ be the set of trajectories $\{s_0,a_0,s_1,a_1,...\}$ induced by a policy π on an MDP $M^\#$ where $s_0=s$ and $a_0=a$. We can map each trajectory in the MDP M to a trajectory in the MDP M' with the function $g: \Phi^\pi_M(s,a) \rightarrow \Phi^\pi_{M'}(s,a)$, $g(\{s_0,a_0,s_1,a_1,...\}) = \{f(s_0),a_0,f(s_1),a_1,...\}$. Notice that g is a surjection and, hence, induces a partition in the domain. We will call $[\varphi']_g$ the set of all $\varphi \in \Phi^\pi_M(s,a)$ such that $g(\varphi)=\varphi'$.

Given a trajectory $\varphi' = \{s_0,a_0,s_1,a_1,...\} \in \Phi^\pi_{M'}(s,a)$, consider now the expected return for the trajectories $\varphi \in [\varphi']_g$, or $E_{\varphi \in [\varphi']_g}\{Ret(\varphi)\}$. We will prove that this quantity is equal to $Ret(\varphi')$.

If $s_i \in S_{skill}$ $\forall i$, there is only one trajectory $\varphi \in [\varphi']_g$, both $Ret(\varphi')$ and $Ret(\varphi)$ are equal to zero and the property holds. Otherwise, there exist a state s_{k+1} such that $s_{k+1+j} = \tau$ for all $j \geq 0$. By definition of R', $Ret(\varphi') = \gamma^k Q(s_k,a_k)$. On the other hand, since the first k returns of any trajectory $\varphi \in [\varphi']_g$ are zero, we can rewrite $E_{\varphi \in [\varphi']_g}\{Ret(\varphi)\}$ as $\gamma^k E_{\varphi\# \in \Phi\pi,M(sk,ak)}\{Ret(\varphi\#)\}$, which is equal to $\gamma^k Q(s_k,a_k)$ by definition of Q. Then, the property holds for any φ'.

Finally, observe that $Q^\pi_{M'}(s_0,a_0)$ is equal to $E_{\varphi' \in \Phi\pi M'(s0,a0)}\{Ret(\varphi')\}$, which is indeed equal to $E_{\varphi' \in \Phi\pi,M'(s0,a0)}\{E_{\varphi \in [\varphi']_g}\{Ret(\varphi)\}\}$ and, since g induces a partition on trajectories in M, and the transition probabilities of both M and M' are equal for S_{skill}, the probability of a trajectory $\varphi' \in \Phi^\pi_{M'}(s_0,a_0)$ is equal to the sum of the probabilities of all the trajectories of $[\varphi']_g$. Hence, $E_{\varphi' \in \Phi\pi,M'(s0,a0)}\{E_{\varphi \in [\varphi']g}\{Ret(\varphi)\}\} = E_{\varphi \in \Phi\pi,M(s0,a0)}\{Ret(\varphi)\} = Q^\pi_M(s_0,a_0)$. We can conclude then that $Q^\pi_{M'}(s_0,a_0) = Q^\pi_M(s_0,a_0)$ for all $s_0 \in S_{skill}$.

The figure 3 shows, as an example, the definition of the RF for our experiments, considering S_{skill} as the set of states with nearby obstacles. The dots represent final states, and the color of the dots their reinforcement value (lighter colors represent higher values).

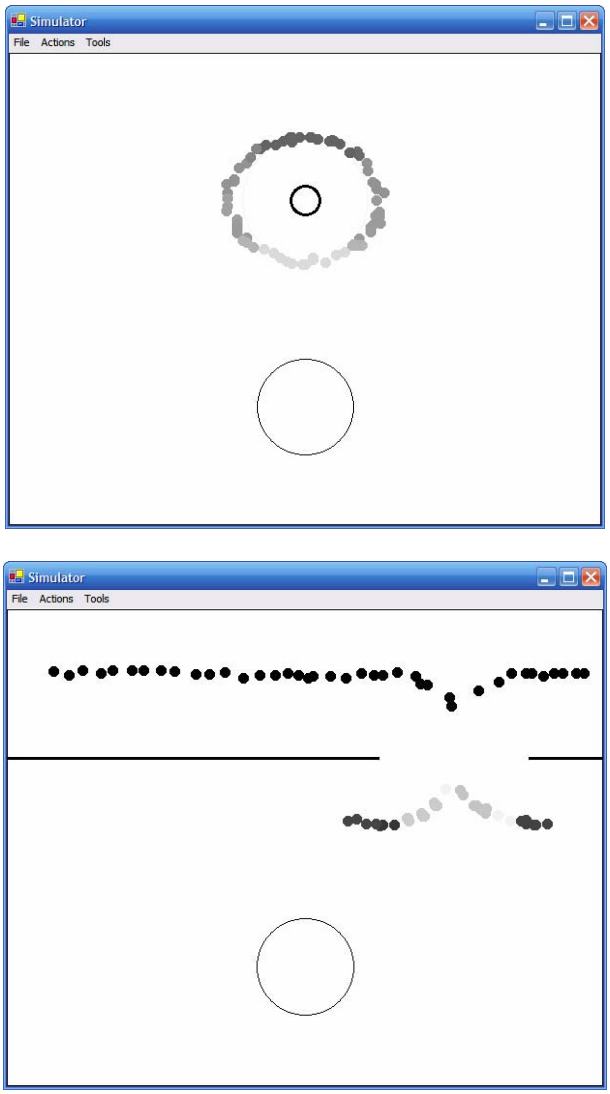


Fig. 3 – RF for the skills constructed from the learned policy. All dots are final states and the color represent the reinforcement value (lighter colors for higher values).

5. SCALING UP

Although the skills learned and extracted in the previous section were not trivial, they were somewhat limited. Based on the formal background presented in this section, we will show an approach to scale up our learning method. Our aim is to

synthesize more complex tasks incrementally, using the learned skills as "building blocks" for more complex policies, without altering the main characteristic of our proposed learning method: a unique RF.

For this purpose, we will use the Options Framework. The Options Framework, developed by Sutton, Precup and Singh [13], extends the usual notion of action to include *options*: closed-loop policies for taking actions over a period of time. Formally, an option consists of three components: a policy π, a termination condition $\beta: S \rightarrow [0, 1]$, and an initiation set $I \subseteq S$. An option $<I,\pi,\beta>$ is available in state s_t if and only if $s_t \in I$. If the option is taken, then actions are selected according to π until the option terminates stochastically according to β.

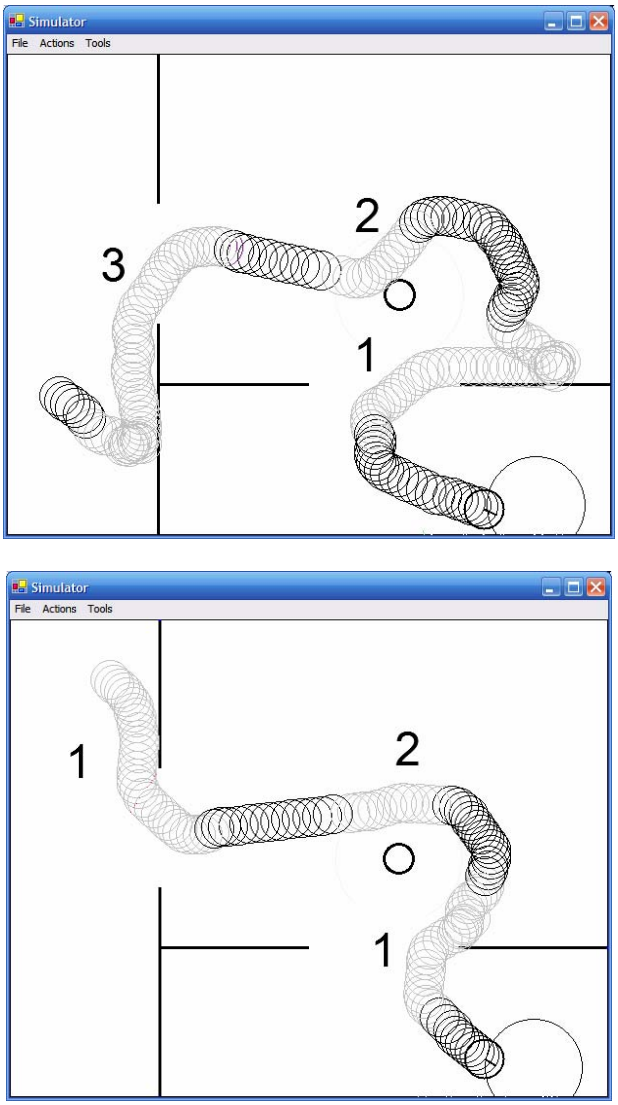


Fig. 4 – Trajectories of the policy learned using the skills. Lighter traces are displayed when the agent executes a skill and black traces when the agent executes basic actions

We will use the option framework by defining an option for each learned skill. To illustrate this approach, we propose an experiment that consists in reaching a light source in a maze-like environment, with two walls, an obstacle and the light source in a corner. The following skills were included as options: "find a hole on the left", "find a hole on the right" and "avoid rounded obstacles". Without these skills, the task would be extremely hard to learn using the universal RF, and very hard anyway if learned with a tailored RF. The results of the learning are shown in figure 4. Dark traces are displayed when the robot executes basic actions and light traces when the robot executes options. Label 1 indicates the execution of the skill *hole on the right*, label 2 the skill *obstacle* and label 3 the skill *hole on the left*. We can see that the agent can reach the light source from any initial position, choosing the appropriate skill on each state. The skill extraction method can be applied to the new learned policies as well to obtain new *higher level* skills to be used in even more complex contexts, obtaining a scaling up methodology. Notice that in the Options Framework the control is not hierarchical in the sense that options do not replace simple actions, but both can be used interchangeably.

5. MOTIVATIONS AND DISCUSSION

When RL is used in robot learning, some human intervention is needed in order to specify *what* tasks are to be learned and, for each task, what will it be considered a success and what will it be considered a failure. In other words, a human RF designer has to figure out which situations and actions should be reinforced and the magnitude of each reinforcement for each different task. But animals and humans can learn some skills completely alone. Understand how RL can be used on scenarios with no human presence can be promising and very useful for some robotic applications. Apart from this theoretical aspect, this method has technical advantages, since the definition of a proper RF for a nontrivial task can be very difficult. RFs are specified by hand and often fine tuned by trial and error. There is no general, direct method to deduce a RF from a high level definition of a task, although research is being made in this direction (for example, see [6]). But even if such a method exists, the description of the task in a high-level language may be ambiguous and lead to unexpected behaviors.

One of the most common behaviors used for testing learning algorithms in robotics is *obstacle avoidance*. At first sight, it is not difficult to define a reinforcement function for this task: a negative reinforcement should be given when a collision is produced. But guided with this function only, the best (optimal) behavior can be *don't move, don't matter what happens*, *rotate in place* or *move a*

small step forward and a small step backwards (why would the robot take the risk of exploring new and challenging regions?). Definitely, this behavior is not what anybody expects from *obstacle avoidance*. We think that the problem here is caused by an incomplete definition of the task: the correct definition should be *avoid obstacles while exploring the terrain*, or *avoid obstacles while moving from one point to another*. But even if we make some effort to specify the task with more detail, there are a lot of optimal strategies for *obstacle avoidance*. For example, when the robot approaches an obstacle, it can circumvent the obstacle, or it can turn around and go away from the obstacle. Both are optimal policies, according to the RF we have defined above. Which is the behavior the designer is trying to achieve? As this example shows, even the description of a reinforcement function in natural language can be ambiguous and may lead to unexpected behavior.

A second problem arises when we try to formalize the function. On real robots, the information gathered from the sensors is noisy, uncertain, incomplete and sometimes too low-level, and it is not easy to map this information to the high-level concepts used to express the RF in natural language. Some approaches to solve this problem includes the parameterization of the RF, the automatic tuning of the parameters during learning ([7] and [8]) and the formalization of the RF in terms of the configuration space ([6]).

Another potential problem produced by the translation of the RF from a high-level definition to a definition based on the agent's sensors is that the mapping may be one-to-many. Since a complete observability of the environment is often not possible, different situations can be indistinguishable by the agent. This phenomenon is called *perceptual aliasing* [9] and can cause that the same action executed on the same (sensed) situation can produce different results.

Finally, on some occasions the information available for a robot is local. Since tasks are more easily expressed in terms of global information, sometimes it is not easy to define an RF in terms of local data. See for example the Figure 2.b. In this environment, a robot with infrared sensors has to cross the wall by walking through a whole. How can the task been expressed with a RF in terms of local sensors?

As a conclusion, we can say that the definition of a proper RF for a task can be difficult. If the robot could learn different skills with a general RF and a careful design of the environment, and it could generate new RFs from past experience, we would have a powerful tool for the development of autonomous robots with more complex capabilities.

On the other hand, the influence of the environment in the learning process and the obtained behaviors has been studied by other authors. Jette Randlov has demonstrated the convergence of RL algorithms to the optimal policy if the transition function (i.e., a formal representation of the agent/environment interaction) is modified in a continuous manner and converges to the final function [10]. Andreas Matt proposes a modification to RL algorithms that allows the simultaneous learning of a task in different environments, obtaining the policy that work better considering all the environments [11]. Sebastian Thrun shows a method for *continual learning*, in which the dynamics of the environment is learned while the agent is learning to solve a particular task [12]. When the agent needs to learn another task, this information is used to speed up the learning. Despite the mentioned works and according to our knowledge, there are no antecedents in the study of our hypotheses and their consequences.

6. CONCLUSIONS

In this work, we described the influence of the environment in the acquisition of new skills and abilities in humans and animals. This influence affects what skills are learned, apart from how they are carried out. On the other hand, both humans and animals can associate rewards with new stimulus, based on previous experience and on the chaining of previous causes and effects.

We propose a novel use of Reinforcement Learning where different tasks or skills are not defined by a Reinforcement Function, but are induced by the characteristics of the environment. We carried out a series of simple experiments with a robot simulator as a proof of concept. On these experiments, a robot learned different skills (avoid round obstacles, find a whole in the wall and pass over it and traverse a corridor) using the same learning algorithm and the same reinforcement function, but changing the characteristics of the environment. After this experiment, we propose a method for the construction of a reinforcement function for these skills based on information gathered from the value function of the learned policy, and we prove that the optimal policy according to this new RF, restricted to a subset of states, is the same as the original learned policy.

These preliminary results show the relevance and the practical utility of our learning method for the synthesis of behaviors in Autonomous Robots, especially in environments with no human presence. Currently our ongoing research is focused on some problems that are tightly related to the hypothesis that we propose in this work, such as: mappings

between behaviors and environments, generalization and definition of partial orders over environments and construction of optimal environments for learning a particular task. We are also trying to scale up this method, including some type of hierarchical learning framework in order to solve more difficult tasks and interact with more complex environments.

7. REFERENCES

[1] B. F. Skinner. *About Behaviorism*, Random House, 1974.

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

[3] R. A. Brooks. Intelligence without reason. *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, John Myopoulos and Ray Reiter, Eds., Sydney, Australia, 1991, pp. 569–595, Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[4] F. S. Keller. *Learning: Reinforcement Theory*. Random House, New York, 1969.

[5] C. J. Watkins. *Learning from delayed rewards*, Ph.D. thesis, Cambridge university, 1989.

[6] A. Bonarini, C. Bonacina, M. Matteucci. An approach to the design of reinforcement functions in real world, agent-based applications, *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 31, no. 3, pp. 288–301, 2001.

[7] J. M. Santos. *Contribution to the study and the design of reinforcement functions*. Ph.D. thesis, Universidad de Buenos Aires, Universite d'Aix-Marseille III, 1999.

[8] A. Ng, D. Harada, S. Russell. Policy invariance under reward transformations: theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*. 1999, pp. 278–287, Morgan Kaufmann, San Francisco, CA.

[9] R. Matuk. J. M. Santos. The clustering aliasing problem in reinforcement learning for robots. In *Proceedings of the Fifth European Workshop on Reinforcement Learning*, Utrecht, The Netherlands, 2001, pp. 33–35.

[10] J. Randlov. Shaping in reinforcement learning by changing the physics of the problem. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

[11] A. Matt, G. Regensburger. *Reinforcement Learning for Several Environments: Theory and Applications*. Ph.D. thesis, University of Innsbruck, 2003.

[12] S. Thrun, T. Mitchell. Lifelong robot learning. *Robotics and Autonomous Systems*, vol. 15, pp. 25–46, 1995.

[13] R. S. Sutton, D. Precup, S. P. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

Diego Ariel Bendersky *is a PhD student at Universidad de Buenos Aires. His main area of research is reinforcement learning applied to autonomous robots. In this field, his main interests are: the influence of the environment in the development of behaviors, automatic discovery of skills and hierarchical learning. He is also interested in neural networks and software development in general. He got a Computer Science degree in Universidad de Buenos Aires, and he has worked in the software development industry in international companies for 10 years.*

Juan Miguel Santos*. He is Professor at the Departamento de Computación, Facultad de Ciencias Exactas (FCEN) y Naturales of the Universidad de Buenos Aires (UBA). He got his PhD degree in Computer Science in 1999, from FCEN-UBA and (DIAM-IUSPIM)-Universite de Aix-Marseille III, France. Currently, he is in charge of the Computational Intelligence Applied to Robotics Project at FCEN-UBA. He is mainly devoted to robot learning and to the development of robots that do not replace human work.*