Animats Adaptation to Complex Environments as Learning Guided by Evolution

Mario Martin, Màrius Garcia, Ulises Cortés
Dept. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Pau Gargallo 5, 08028, Barcelona (Catalunya), SPAIN
mmartin@lsi.upc.es, marius@lsi.upc.es, ia@lsi.upc.es

Abstract

In this paper, a new approach to adapt animats to complex environments is proposed. It is based on the advantages and drawbacks of two known strategies: learning and evolution. The proposed approach uses a new learning by reinforcement mechanism guided by an innate and general knowledge -obtained by an evolutionary mechanism- that triggers off a developmental process of learning general behavior. This developmental process increases the animats set of behaviors, characterized by sequences of actions, and facilitates the solving of more complex tasks. This approach is studied, described and finally illustrated with a set of experiments in a complex environment.

1 Introduction

Several proposals of frameworks for autonomous systems (*animats*) have been made in the latest years. One of the most successful has been the reactive approach, which states that stimulus-driven behaviors can accomplish complex tasks.

There are different kinds of reactive systems, ranging from systems based on simple architectures as look-up tables or stimulus-action rules, to structured and complex systems as "Sonja" [6] or those designed with the "subsumption architecture" [2].

Unfortunately, one of the main drawbacks of this kind of systems is their inability to adapt autonomously to complex environments [1]. Usually, reactive systems must be hand-designed in order to obtain adequate results. In this case, we should say that they are born adapted to their environment.

There exists some mechanisms developed to overcome this problem, basically divided into two strategies: learning and evolution.

[1]The term "complex environment" is defined more precisely in the next section, but we can advance that the environment complexity depends on the relationship between the animat and the tasks it must solve in the environment.

The first one considers the adaptation of the animat to the environment as a learning process.

This idea has been exposed by several authors [15] [7] [13]. For them, the adaptation process consists in learning (usually by means of a reinforcement signal) a mapping between the set of actions and the set of perceptions that reveals an adequate behavior to the environment.

These methods frequently begin the learning process without any previous knowledge of the environment and, although they are a valid approach for some environments, as we will see in the next section, they are not adequate for complex environments (specially when the animat starts with a set of simple actions in comparison with the needed to solve the tasks proposed in there).

The other strategy is the evolution by means of selection and variation of the best adapted animats, proposed for instance in [3], [12] or [11].

This approach usually states that animats are born completely hard-wired (adapted), not by the design made by an engineer, but by the natural process of evolution simulated by genetic algorithms [10]. This approach is borrowed from nature and reflects the fact that animals are born with a high level of behavioral information.

Nevertheless, this proposal presents several drawbacks. It implies that for every situation, the behavior of the animat is "innate". It's true that living beings have to be born with an enormous behavioral knowledge, but it is not complete.

There are objective reasons that support this fact. First of all, genes act in this case as predictors of the actions the animat has to do in any situation to survive in its environment [8]. A codification of the architecture that always indicates the animat what action has to do, in complex environments, is really expensive, and needs a big genoma and a very long evolutionary process.

Secondly, but more important, an animat born with all its knowledge innate doesn't have any robustness when the environment changes or is unpredictable in certain aspects. A little change in the environment can provoke a total inadaptation of the animats in the next generation.

Our proposal is an intermediate approach between these two. On one hand, we consider learning as an essential process that lets the animat tune its behavior to the concrete environment where it lives. But we don't expect the animat to learn beginning without any knowledge of the environment. As we will see in the next section, learning from nothing is very hard and almost impossible, specially when the actions the animat can take are so simple that require a long chain of actions to solve any problem.

In fact, an animat can learn to solve a complex task depending on the set of actions available to reach its objective. If it is necessary, the animat must learn new behaviors (complex actions) to achieve solving more complex tasks. What we propose is a developmental process of the animat in order to be sure that it gets new knowledge, which will let it solve more complex problems.

This developmental process needs to know which behaviors to learn in every moment. We have two options. The first is to have a teacher, but it is not realistic in an animat environment. The other one, more appropriate, is to have an innate guide as a result of an evolutionary process.

This integration pretends to obtain the good benefits of both worlds. On one hand, the animat learns all the particularities of its environment and it is sensitive to its changes. This makes a robust animat. On the other, a simplified genetic process, drives the animat to achieve enough development letting it successfully accomplish complex tasks.

This new approach is supported by animal learning research [4], where we can clearly observe the influence of innate mechanisms to guide and ease the learning process. It is also supported by cognitive science research when we speak about a *developmental process in an environment*, instead of only a learning process.

In this article we will expose this proposal in separated parts.

In the next section, we'll study the learning and the genetic requirements of our proposal. First we will study deeply the limitations of the classical reinforcement learning mechanisms as well as its causes, establishing which ought to be the constraints that a learning adaptation mechanism has to satisfy in our framework. Afterwards we'll study the kind of information the genetic mechanism has to manage to guide the animat development.

Taking both parts of our proposal, the more elaborated is the learning one, in which we have experimental results. The genetic part has been studied but we still don't have implemented results.

In section 3 we'll show the learning mechanism developed under the constraints deduced in section 2.

In section 4 we'll see the experimental results obtained for an animat in a complex environment, applying the learning mechanism of section 3 with an optimum guide that we'll consider achieved by a detailed evolutionary process.

Finally, in the last part, we'll expose the conclusions obtained in our research.

2 Adapting to Complex Environments

In the introduction, we have proposed to consider the adaptation to a complex environment as a learning process guided by an innate knowledge obtained from an evolutionary process. In this section we'll study these two aspects.

2.1 The learning process

There have been several approaches, in animats learning research, mainly based on the idea of reinforcement.

Reinforcement consists on a signal (internal or external) that indicates in a concrete situation whether the executed behavior is appropriated. The environment adaptation comes from the finding of a correct mapping between the available set of actions and the set of perceivable stimulus, in order to maximize the positive reinforcement. We believe this is the correct approach when we are dealing with animats.

The reinforcement signal can be *immediate* or *delayed.* Immediate means that a reinforcement is given instantaneously for every executed action. In the delayed case, the reinforcement signal is given to evaluate a sequence of actions after its execution.

Although it is more complex to learn behaviors from a delayed reinforcement, there are good reasons to choose it.

The most important reason is that it is more realistic for animats: an animal gets a reinforcement when it is eating, but not while it is chasing what to eat.

Moreover, simplifying the mechanism that gives the reinforcement signal is one of the advantages of the delayed reinforcement, because it only has to recognize when the final state has been reached.

Finally, an immediate reinforcement, forces the mechanism to know exactly which is the best solution in every moment, and consequently the animat would only *copy* a previously known behavior. On the other hand, the delayed reinforcement mechanism does not need to know how to reach the objective. In this case, we can say that the behavior to solve the problem is not copied but *emerged* from the interaction of the animat with its environment using the reinforcement signal. In the experiments section, we'll show how different general behaviors completely valid can emerge in this way.

The reinforcement is given to solve a concrete problem, but an important feature the learning mechanism must show, is the ability of learning *general* behaviors to solve problems. Problems with similar situations ought to be solved with the same learnt method.

Another important feature that the learning process must show is the ability for learning with a limited world perception. It is known that a realistic animat can't perceive the whole environmental information instantaneously. This limited perception implies that the information the animat has of its environment is always partial, incomplete and ambiguous. To solve a problem, the system must learn to search and perceive the required information. This kind of activity has been called *animated perception* [1]. In many other learning mechanisms it is supposed that the animat is able to perceive all the information of its environment. When these methods have to deal with ambiguities, the problem of "perceptual aliasing" [17] appears, disabling the learning process. The limited perception is a realistic constraint, but it is not a negative one. It is widely known that a limited environmental perception can produce a very profitable generalization process [5].

The problem resolution for an animat in the environment is characterized by a sequence of actions which will receive a reinforcement signal when the entire sequence has been executed. Without any initial knowledge, the animat has only the trial and error procedure to solve any problem.

With the existing methods of trial and error, it is known that it's very hard to learn how to accomplish goals which imply a long sequence of actions. If it happens, we say this is a complex task. Moreover, if it is necessary to solve this kind of tasks in order to survive, we say that we are dealing with a complex environment. It's important to note that the complexity of a task comes from the adequacy of the actions the animat can take to solve it. An adequate set of general actions lets solving the problem in a few steps. On the contrary, a primitive set of actions will provoke the learning failure.

The animat is born with a set of primitive actions and a set of sensors provided to adapt itself to its environment. There are some tasks too difficult to learn from the beginning with this set of actions, but fortunately the animat can learn more general actions which let it improve its skills. In order to do this, the animat proposes itself intermediate general tasks to solve. The behaviors learned to solve this tasks are then included as new available actions to solve more complex problems. So, we take an incremental point of view where the animat stays in different developmental stages, defined by its own set of actions (primitives and learned) which let it solve tasks of different complexity level.

All the observations we have made let us define the important features of the learning mechanism we have to design to obtain an adaptation of the animat to complex environments. Summarizing, these are:

1. Learning with delayed reinforcement signal.
2. Working with ambiguous information.
3. Problem solutions characterized as sequences of actions.
4. Learning new general complex actions (behaviors) useful for accomplish complex tasks.

A learning mechanism we propose satisfying these constraints is exposed in the next section.

Nevertheless, this considerations misses the mechanism that indicates the system which behaviors has to learn in order to achieve a high complexity level of solving tasks. We state that an evolutionary mechanism should do this job.

2.2 The evolutionary process

According to the exposed learning considerations, the possibility of learning complex tasks is subordinated to the set of actions available to the animat. If the set of actions is not adequate, the animat tries to learn new behaviors before facing a complex task.

We propose that a genetic mechanism, inherited from fathers to sons, guides the learning process, giving a sequence of general tasks to be learned to solve in order to achieve the animats adaptation to the complex environment.

The information this genetic mechanism must convey, consists in a sequence of tasks to learn. The order of this tasks in the sequence is very important, because they are interrelated being impossible to be learnt some tasks if some others haven't already been learnt.

For every task there must exist the next information:

1. The objective state and therefore, when the positive reinforcement signal must be given.
2. When an error has occurred, i.e., the maximum number of actions to be taken to solve a problem and the dangerous states (if any).
3. The set of actions to use to solve a task from the whole set of possible actions, as well as the subset of sensors.

The objective state is easy to characterize, because it normally consists on a concrete perception, a set of sensors with an appropriate associated value. Finding this objective state innately provokes an internal signal of positive reinforcement which is assimilated by the animat.

It is also necessary to know when there has been no success. The failure is easy to characterize too. It appears when a number of actions have been executed and the objective hasn't been reached. The failure can also appears when a painful action has been executed.

Sensor 1	Sensor 2	.....	Sensor n	Last action	New action

Figure 1: Entries to obtain the new action

Finally the last part tries to ease the learning mechanism, indicating which is the set of actions and sensors to use when we are solving a problem.

All this information should be codified in genes and chromosomes, and evolved by means of a genetic algorithm.

The main difference between this approach and the others within the genetic framework consists in, while the others completely codify the information of the animat architecture, this only specifies general aspects that are complemented with a learning process in order to obtain the complete adapted animat.

This evolutionary mechanism hasn't been implemented yet and therefore we cannot give results on it. What we will do is simulate that this process works, to show that the learning mechanism exposed in the next section, guided by this kind of knowledge, is able to accomplish complex tasks (tasks that cannot be accomplished with classic learning mechanisms).

3 The proposed learning mechanism

In this section, we'll expose the learning mechanism developed according to the requirements obtained from the study of the previous section.

The mechanism has to learn to solve a problem with a sequence of actions. At the beginning, the animat will not have any knowledge of which actions do in any situation, and therefore it'll execute randomly chosen actions trying to find useful sequences. The knowledge is obtained by a reinforcement signal which indicates whether the proposed sequence of actions solves the problem.

Reactive systems, use perceptual information in order to decide what actions execute. As we are characterizing solutions as sequences of actions, we have added the last executed action to the set of sensors as a necessary data for the animat to choose a new action.

The animat's architecture developed by the learning process consists in an indexed table (see figure 1). The entries of the table are formed by the sensors used in the problem solving and the last executed action. The output is a list of the possible executable actions with statistics to compute the probability to obtain a positive reinforcement if any of them is executed. The statistical information is composed of the number of successes and failures occurred when executing the action in a concrete situation.

There exists one entry to the table for every situation the animat has faced. In order to find the best action to execute, it will try to calculate which is the action with the highest probability of leading the animat to a positive reinforcement.

The probability to get a positive reinforcement when executing an action a, given the actual situation s, is estimated as the ratio between the successes obtained when this action has been executed in this situation versus the total number of activations in the same situation:

$$P(s,a) = \frac{\#Successes(s,a)}{\#Successes(s,a) + \#Failures(s,a)}$$

This estimation is only significative if the entire sequence of actions to solve the problem is obtained with the method of finding the action with the highest probability of reinforcement. This mechanism, that will be used for problem solving, is now explained:

1. Repeat until there's an error or a success (problem solving phase).
   (a) Choose the action with the highest probability to obtain a positive reinforcement from the actual situation. In case there is more than one, choose one randomly.
   (b) Keep the chosen action as the next of the sequence.
   (c) Execute it.
2. Actualize the statistics for each of the actions belonging to the constructed sequence of actions (learning phase).

When the first action has to be chosen, we consider that the last action executed is `NIL`.

An error occurs when the system executes a dangerous action or when certain number of actions have been executed without any success. This is an innate information, obtained from the genetic algorithm (see the previous section).

When the animat faces a new situation s, it creates a new entry of the specified type to the table, with the initialization value of $N/2$ for the $\#Successes(s,a)$ and the $\#Failures(s,a)$ of every executable action a. At this time, all the actions have the same probability estimation of obtaining a positive reinforcement.

The actualization of the statistical information is described in the next procedure:

- For every action a (taken in the situation s) of the sequence of actions, do:
  1. If a positive reinforcement is obtained, add 1 to $\#Successes(s,a)$
  2. If a negative reinforcement is obtained, add 1 to $\#Failures(s,a)$

3. Normalize the two counters, multiplying them by $\frac{N}{N+1}$

The normalization process is used to emphasize the last sequences of actions obtained, over the previous. With this procedure $\#Successes(s,a) + \#Failures(s,a)$ remains constant, with value N.

With this learning process, the animat learns which are the actions to execute in each situation. The animat learns form its successes and also from its failures, because failures inhibit the tendency to repeat an action in the same situation.

The learning cycle stops when the system doesn't learn anything in a determined number of new problems presented. To determine when the animat has finished its learning, we will use a table called *policy*. This table keeps the information of the best possible action to execute in every situation. If this table doesn't change during a large number of experiments, the system is not learning anything new and the learning process stops. In this case, the knowledge of how to solve a problem is kept in the policy table, which will be used to increase the animat set of actions and to achieve solving more complex problems.

4 Experiences

A set of experiments have been designed in order to show the performance of the proposed adaptation mechanism.

The environment selected, where the animat will solve problems, is the "blocks world". This environment has been used by Chapman [5] and Whitehead and Ballard [17] to show the possibilities and drawbacks of reactive systems in complex domains.

Chapman demonstrates, solving the "fruitcake" problem [2] that reactive systems can accomplish complex goals in this domain without the combinatorial explosion of stimulus-action rules predicted by Ginsberg [9]. The problem is solved using a "deictic" perceptual system that reduces enormously the number of states of the system.

Starting from this work, Whitehead and Ballard developed a mechanism to enable *learning* in reactive systems with deictic perception. Chapman has shown that this kind of systems can solve complex tasks, but only if they can operate with an incomplete description of the world. The mechanism they developed, tries to overcome the incapacity of the Q-learning [16] method to solve problems when it is fed with ambiguous information. Their results are positive, but far from learning to solve the "fruitcake" problem.

The blocks world is not a typical environment for an animat, but (as we will see) it fulfills the required characteristics to consider it complex: incrementally complex tasks can be proposed, animat's perceptual system gives incomplete or ambiguous reports, there is an initial set of actions too simple for solving complex tasks, etc. We have selected this environment because it has been previously used in works about reactive systems for complex tasks, but we can easily imagine more standard environments for animats with similar characteristics (a labyrinth with movable boxes, etc).

[2] How to build, from a set of blocks labeled with characters, a stack where the word "fruitcake" can be read.

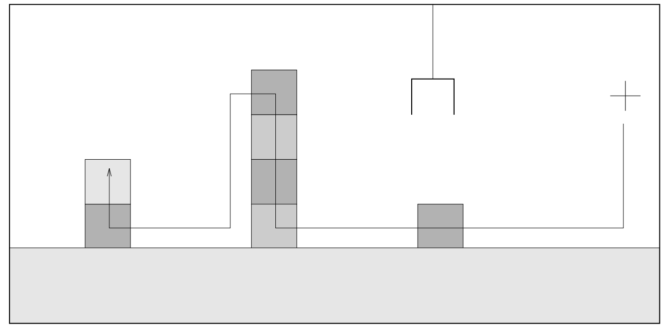

Figure 2: A standard setting of the environment.

A standard setting of the environment (see figure 2) is composed of a window where we can see a table and a set of blocks of different colors resting on the table or forming a stack over other blocks. In the picture we can also see the visual focus (in a standard learned trajectory to find a block) and the hand of the animat.

The animat of this environment has been designed with a set of sensors that reports ambiguous information of the world (i.e., it does not express univocously the state of the world) and a very simple set of actions. These sets of sensors and actions have been chosen in order to be as realistic as possible for a future physical implementation of the animat.

The sensory system of the animat is composed of:

- Visual sensors:
  - Foveal vision: A sensor that represents the visual focus of the animat and that gives complete information of the visualized zone contents (kind of object, color, etc.).
  - Peripheral vision: Four sensors that give partial information of the zones surrounding the visual focus. Each sensor indicates only the presence or absence of an object different from vacuum in the zone on the right of, on the left of, above or bellow the visual focus respectively.
  - Visual memory: A sensor that reminds the animat the contents of the zone where the visual mark is placed. Marks are pointers to the mental map of the animat and are used to locate point of particular interest for the problem solving process.

- Propioceptive[3] sensors:
  - Hand-Fovea angle: A sensor that reports the relative position between the hand and the visual focus on the world. It can take four values: up, down, right and left.
  - Hand-Fovea distance: A sensor that expresses the relative distance between the hand and the visual focus on the world. It can take three values: Far, near and very-near.
- Tactile sensors:
  - Content: A sensor that indicates whether the hand is grasping and object.
  - Nearness: Three sensors that express whether the hand is touching something on the left, right or bottom, respectively.
  - Open hand: A sensor that reports whether the hand is open or closed.

This set of sensors, enables the animat to operate adequately in the blocks world.

Initially, the set of actions of the animat is composed of:

- Actions over sensors:
  - To move the visual focus one position: Four actions that can move the visual focus up, down, right or left, respectively.
  - To move the visual focus to the mark: An action that moves directly the visual focus to the position where the mark lies.
  - To put a mark: An action that keeps in mind the local information and position where the visual focus currently is.
  - To remove a mark: An action that resets the visual memory.
- Actions over actuators:
  - To move the hand: Four actions that can move the hand one position up, down, left or right, respectively.
  - To open and to close the hand: Two actions that allow to open and close the hand, grasping or releasing and object respectively if it is the case.

The task the animat must solve in this environment consists in putting an unlocalized block of a given color in the position where the mark lies [4]. Initially, the animat only has the mentioned set of actions and sensors. Then, a solution to the problem proposed will be composed of a large chain of actions that, as we have seen in section 3, is the main problem that prevents learning from trial and error.

In order to solve the problem with our method, we will suppose that the genetic information of the system provides a set of general actions to learn which allows to solve the initial problem. In our case, we suppose that this innate knowledge proposes to learn the following general tasks:

1. Searching for a block: The animat must search for a colored block in its world. To do that the visual focus (fovea) of the system must land over a block of the desired color. To search for the block, the animat has to use its local visual system taking advantage of the environment features (for instance, that any block lies over the table or over other blocks).
2. Moving the hand to the visual focus: This task consists in moving the hand towards the visual focus avoiding to collide with any stack of blocks. The visual focus can be situated on a block or on the vacuum. In the former case, the block must be cleared.
3. Removing the top of a stack: This task consists in removing a block from the top of a stack. At the starting state, the fovea and the hand are placed in the target block. This block must be removed by the hand without colliding against any other block.
4. Grasping any block: This task consists is taking a block with the hand. The block can lie at the top of a stack or in the middle of it. In the last case, the animat must previously unstack all the blocks placed upon the target block.

The solving process for each of these tasks generates a behavior that can be considered as a new general action. These new actions increase the repertory of actions known by the animat and can be used for solving more complex problems.

In the following subsections we will show the results of applying the learning mechanism proposed in section 3 for each of these tasks.

4.1 *Task 1: Searching for a block*

This task consists in searching for a block of a desired color from the set of blocks in the world. This is a non-trivial task, due to the limited perceptual system of the animat. To solve this task, the animat must take advantage of the invariant features that it finds by experimentation in the environment, as the empirical fact that blocks rest on the table or over other blocks.

In order to solve this task, there is not necessary the whole set of sensors and actions. We suppose that the

[3] Propioception is the direct perception of the relative position of parts of the animats body via, for instance, muscle tension.

[4] With this action, substituting colors by letters, it is not very hard to solve the fruitcake problem.

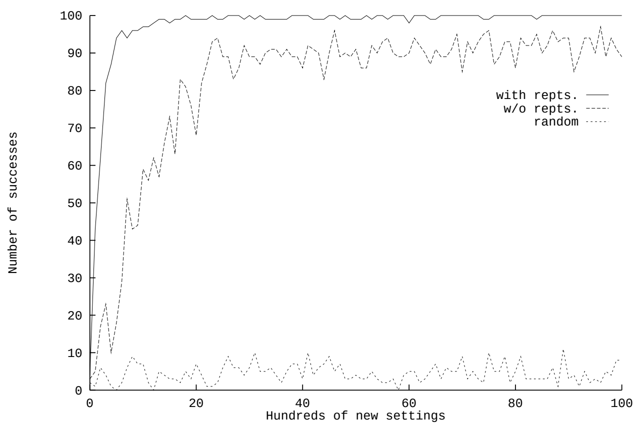

Figure 3: Graphic for searching for a block

innate knowledge of the animat, provided by its genetic code, suggests the following set of sensors and actions.

- Useful sensors: Foveal vision, peripheral vision.

- Useful actions: To move the visual focus up, down, left and right.

The animat will receive a positive reinforcement when it perceives by the foveal vision a block of the desired color. The animat will receive a negative reinforcement when it does not reach the goal in 25 actions or when the visual focus goes out of the environment window.

The learning cycle consists in presenting new situations (settings) of the environment to the animat from which to solve the task. These new situations are randomly generated.

The results obtained in the "searching for a block" task can be seen in figure 3. It represents the number of successes obtained (y axis) for a hundred of settings presented to the animat (x axis).

The first experiment for this task was a presentation of settings to the animat with the learning ability inhibited. In this case, the animat could only solve the problem choosing randomly an action for each situation. The outcomes of this experiment can be used as an index of the task's complexity. For this task, the percentage of successes has an average of the 5%, with a standard deviation of 5.2.

In the second experiment, the learning ability was allowed. In the graphic for this task, the effective learning of the animat can be observed, although near the 2.000 presentations of different settings the learning rate was stabilized at the 90% of successes. This percentage shows that there are a 10% of very complex problems that the animat is not able to solve. In this case, the animat has found a good policy and it considers that it is not useful to change this policy for solving this little percentage. We can say that the system has reached a local maximum. We must improve the mechanism to get a better learning for the animat.

The third experiment is a variation of the previous one. The change consists in forcing the animat to repeat the settings that it has not solved until it has a success or the number of repetitions has exceeded a predetermined number n. The repetition of not solved experiments punishes the policy function and forces the animat to find a solution for them. The results obtained with this strategy, can also be seen in the graphic. There are only counted as successes the problems solved at the first trial. A repetition of a setting it is not considered as a new setting. As it can be noted, the solving rate shows a fast growing that reaches the 95% of successes with 500 different settings and finally achieves a stabilized 100% of successes with 10.000 settings,

At the end of the learning process, the animat has discovered an adequate policy for searching for a block. The obtained policy depends on the initial conditions of the environment. For instance, if the world has few blocks forming stacks, then the obtained policy will consist in going down to the table, moving left or right until finding a stack, examining it and if it has not the desired block then finding another stack, etc. The trajectory of the visual focus in figure 2 illustrates this policy. On the other hand, if there are a lot of blocks forming heaps then the policy will consist in finding heaps and examining them by stages.

It is important to note that the obtained policies have emerged by means of the interaction of the animat and the environment, because these policies does not exist "a priori" in the reinforcement mechanism.

The policy found for this task will be kept in a table in order to be used as an action for more complex tasks.

4.2 Task 2: Moving the hand to the visual focus

This task consists in learning a general action to coordinate the hand and the eye of the animat. This action must be able to move the hand towards the place where the visual focus lies.

The hand, as depicted in figure 2, is hanging on the world ceiling and thus, the hand can only reach the visual focus if it is situated on the vacuum or over a cleared block. While moving, the hand must avoid colliding with any stack.

The following are the sets of actions and sensors needed to learn this new action:

- Useful sensors: Fovea, Nearness, Hand-Fovea distance and Hand-Fovea angle.

- Useful actions: To move the hand.

The animat learns to solve the proposed task. Figure 4 shows the learning progression until 10.000 new settings.

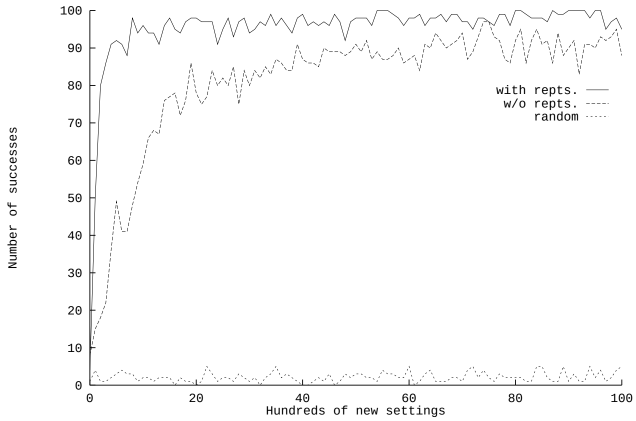

Figure 4: Graphic for moving the hand to the visual focus

The performance without learning shows a success rate of 2%. The learning without repetition of failed settings reaches the 87% of successes. Finally, the learning with repetition of failed settings achieves with 700 settings the 90% of successes.

4.3 Task 3: Removing the top of a stack

This task consist in removing a block from the top of a stack. At the starting state, the fovea and the hand are placed in the target block. The block must be removed by the hand, avoiding to collide with any other block.

The following are the useful sets of actions and sensors to solve the task:

- Useful sensors: Nearness and Open hand.
- Useful actions: To move the hand, to close hand and to open hand.

This task is easily learned by the animat, as can be seen in figure 5. With only 300 settings, the number of successes reaches the 95%.

4.4 Task 4: Grasping a block.

In this task, the animat must take a block with the hand. At the initial situation, the visual focus is on the target block. This is a non-trivial task because the target block can be cleared or placed on the middle of a stack. In the last case, the animat must learn to unstack the blocks placed over the target block.

To learn how to solve this task, the actions and sensors used are:

- Useful sensors: Fovea, peripheral vision, open hand and content.
- Useful actions: To move the visual focus, to move the hand towards the visual focus and to remove a block.

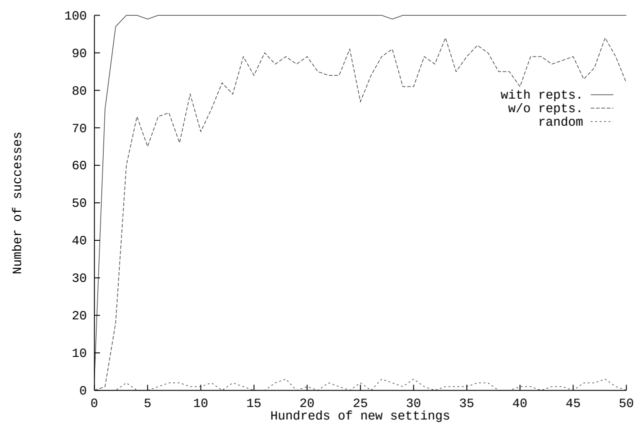

Figure 5: Graphic for removing a block

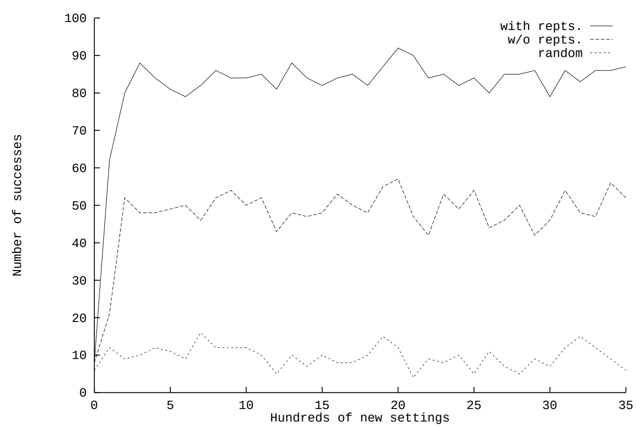

Figure 6: Graphic for grasping a block

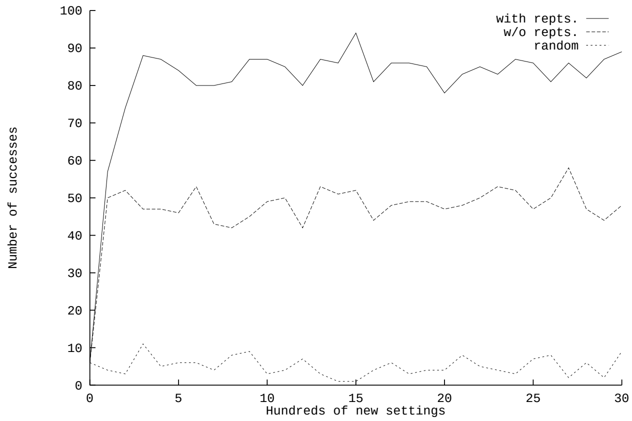

Figure 7: Graphic of the final task

This is the first experiment that uses previously learned actions to solve the task. This can be noted in figure 6 which shows the learning performance for this task. The results for learning with repetition of failed settings can only achieve the 85% of successes. This is mainly due to the use of actions previously learned that does not have a 100% of successes.

4.5 Final task: Moving a block to the mark.

This is the initial task proposed to the animat. The task consists in moving a block of a desired color to the place where the mark lies. To solve this task, the animat must find the block, grasp it and move it to the target place.

To solve this task, the set of actions and sensors used are:

- Useful sensors: Fovea, open hand and content.
- Useful actions: To move the visual focus to the mark, to move the hand towards the visual focus, to grasp a block and to close the hand.

Figure 7 shows the results of the learning process. The solving rate is stabilized at about 85% of successes.

5 Conclusions

In this paper, it has been presented a new approach for the adaptation of animats to complex environments. This new approach is base on the integration of learning and the innate knowledge produced by an evolutionary process. The required features for an adequate learning mechanism have been studied. It has also been studied the kind of information that the evolutionary process must convey.

Finally, a mechanism for learning has been developed and tested for the blocks world.

The research presented in the paper lead us to several conclusions:

On one hand, we have demonstrate that it is possible to learn how to solve complex tasks. The task proposed in the blocks world, considering the initial set of actions the animat knows, is not currently resolvable by any other direct learning method.

On the other hand, there are some aspects that must be developed, mainly the evolutionary mechanism proposed in the subsection 2.2.

Finally, we can note that several aspects can be improved. One of them is the limits for learning observed in tasks 4 and 5 that are caused by the use of learned actions not experienced in all situations. This problem could be solved enhancing the tuning of the previously learned behavior to the task when the last has reached a high percentage of successes.

6 Acknowledgments

We would like to thanks Neus Català for her support developing the software in which to visualize the problem solving behavior of the animat in the blocks world. We also would like to thank the people that collaborates in the PhD seminaries on Artificial Intelligence at the *Universitat Politècnica de Catalunya* for their criticism and suggestions.

References

[1] D. Ballard. Reference frames for animate vision. In *Proceedings of IJCAI89*, 1989.

[2] R.A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[3] R.A. Brooks. Artificial life and real robots. In *Proceedings of ECAL91*. MIT Press/Bradford Books, 1992.

[4] S. Carey and R. Gelman, editors. *The Epigenesis of Mind: Essays on Biology and Cognition*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

[5] D. Chapman. Penguins can make cake. *AI Magazine*, 10:45–50, winter 1989.

[6] D. Chapman. *Vision, Instruction and Action*. MIT Press, Cambridge, MA, 1991.

[7] D. Chapman and L. Kaelbling. Learning from delayed reinforcement in a complex domain. Technical Report 90-11, Teleos Research, 1990.

[8] R. Dawkins. *The Selfish Gene*. Oxford University Press, 1989.

[9] M.L. Ginsberg. Universal planning: An (almost) universally bad idea. *AI Magazine*, 10:41–44, winter 1989.

[10] D.E. Goldberg. *Genetic Algorithms*. Addison Wesley, 1989.

[11] I. Harvey, P. Husbands, and D. Cliff. Issues in evolutionary robotics. In *Proceedings of SAB92*. MIT Press/Bradford Books, 1993.

[12] J.R. Koza. Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Dept. of Computer Science, Stanford University, 1990.

[13] S. Mahadevan and J. Connell. Automatic programming of behaviour-based robots using reinforcement learning. In *Proceedings of AAAI91*, 1991.

[14] M. Martin and U. Cortés. A developmental approach for integrated architectures. In *ILA Workshop notes, ECML-93*, 1993.

[15] R.S. Sutton. Reinforcement learning architectures for animats. In *Proceedings of SAB91*, 1992.

[16] C. Watkins. *Learning from delayed rewards*. PhD thesis, Cambridge University, 1989.

[17] S.D. Whitehead and D.H. Ballard. Learning to percive and act by trial and error. *Machine Learning*, 7:45–83, 1991.