Learning agents in adaptive interface

C. Destruel, H. Luga, Y. Duthen, R. Caubet
E-mail: {destruel | luga}@irit.fr
IRIT-UPS, 118 route de Narbonne, 31062 TOULOUSE Cedex, France

Abstract : *We present in this paper a genetic-algorithm-learning-based approach of the adaptive interface problem. We have built a classifiers system in order to control a learning micro-agent which is going to watch over a very small set of parameters. We present the structure we use to define these agents and to organize their communication. Finaly, we are going to see how we can use this messages-based system in a virtual application.*

I. Introduction

Our images synthesis group is developing a complete virtual world system which includes a virtual reality applications design platform [Torguet95], a behaviour module and a 3D interface system. The computer world is becoming more and more complex for the users. Systems, hardware or software, are becoming more and more powerful and they offer more and more possibilities (approximately 600 functions for UNIX, 1000 for AutoCad, etc). This is the reason the user should be now more and more attentive to his system.

In spite of the complexity of an application, our goal is to adjust its interface presentation (forms, organisation, etc.) to make the system utilization easier. Using new hardware devices is a rather difficult adaptation. In fact, working with new tools (cyber eyes, 3D mouse, etc) requires an obligatory and incompressible learning period. In both software and hardware cases, we try to simplify the human-machine relation by working on the user's point of view.

A different or complementary approach is necessary to try to modify the interface. Our goal is to build an adaptive interface, able to modify its presentations and its actions according to the characteristics of every user.

Now we are going to see how we can achieve this adaptation. What is the interest of artificial-life-based system and what can we accomplish ?

II. Adaptive interface

The interface we know as adaptive or self-adaptive due to its ability to observe the user and decide to modify itself. This modification decision is meant to simplify the user's job. We are not going to speak about adaptable interfaces which are simply configured by each user.

An adaptive interface should enable the user to minimize his efforts to evaluate the tool and maximize his concentration on the resolution of the problem[Innocent82].

1) To adapt an interface

First, an interface could be used by each of the users. The first solution, applied to many systems, consists in dividing them in user levels. Each user is able to select his own level position. Automatically increasing from beginner to expert is the first self-adaptivity level. This evolution can be done after a certain time or when the user realizes some specific actions. Accordingly, we can present useful functions to the user at any time.

After selecting the user's level, the interface should be able to arrange its presentation to reply in the best way the user's requests. This new organization affects its aspect by adding or simply modifying some elements. Creating new macro-command regrouping some frequently used actions is an important point of the adaptation process. Reorganizing the user's working space sets a more important problem: we must minimize the ratio of the performance damages following the modifications and the expected gains.

An other important adaptation point concerns the indications the system can provide the user. It's very important to always present information, error or help messages, so that they can be clearly understand by the user. An explanation can be quite useful for a user and not very informative for another one. By using different user levels, we can provide some explanation steps. These explanations can present both, a video on how to use the function step by step, and a simple reminder of the function parameters [Meyer94].

2) The purposes

We need to follow three important steps to perform an interface adaptation. First, we should determine the user's knowledge. Second, we need to know his goals in order to be able to modify the system.

The interface cannot decide to modify itself unless it has attentively observed the user. These pieces of information are stored in a chronological recording file. We can imagine a different variety of files [Crow93]. The simplest one only records every sequential act, but a more complex one can store every information (all used functions, all complete actions, cancel with help, cancel and retry, etc.). When using these recordings, the interface produces a user model [Berthome94]. Considering a descriptive model, we dispose of all the information which characterizes each user. This step that allows to build a user representation from the chronological record is based on an heuristic use. These heuristics can for instance exclude parasite noise, select specific information and recurrent schemes. In the same way, by using heuristics and user mode, the interface can determine the modification the system needs.

3) Some problems

Realizing an adaptive interface arises several difficulties. The first one is about the user's reactions concerning the modifications the system makes [Sukaviriya93]. In fact, the modifications which seem to be imposed by the system (and not selected by the user) may not be accepted. A too large modification of the user's habits (modification of communication mode, menu organisation or who takes the dialogue initiative) produces a loss of performance and a discomfort for the user. To minimize this problem, the interface can ask the user to manage modification about his working method (principally a dialogue type modification). Then the interface will manage all the other modifications. Otherwise, we can try to split up adaptation to compose a modification unit. Each step of modification should not be detected by the user.

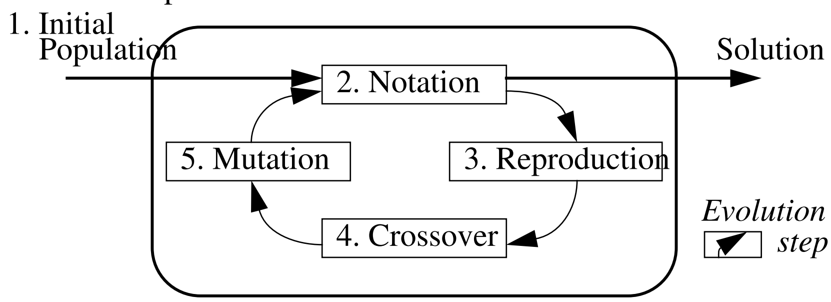
Another problem is adaptation diversity. When using heuristics, interface modification capability depends on the knowledge experts place on it. Hence we must anticipate not only what is going to be adapted but also how and when. In fact, the applied adaptation is the best one that as been built by a specialist for a similar situation.

III. Artificial life adds

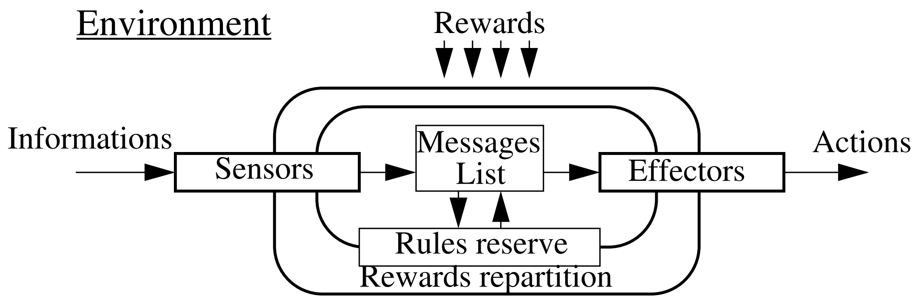
Artificial life technology and more specially genetics algorithms (G. A.) based systems provide some solutions for evolutionary or adaptation systems. First we discuss the solution we are considering and then its application.

1) Genetics algorithms

Genetics algorithms are learning methods inspired by natural life [Holland75]. They perform efficient optimization in deceptive and complex problems. They represent the candidate solution in a logical form and use artificial selection, cross-over, reproduction and mutation to build a final solution. This optimization method is very useful when the problem is not totally defined. The best example can be seen in the mathematical function optimization. Using G.A., we can find solution without verifying the derivative function and its continuity. G.A. just needs function expression.



Our goal is to realize an adaptive interface. This interface needs a learning module which is used to identify and apprehend the user's characteristics. The classifiers system is a learning method that amalgamates a genetic algorithm and a real time rule system [Goldberg89]. It acts through sensors and effectors which produce and respond to messages. A message consists of a bit string which triggers rules (classifiers) in the system. Rules are logic expressions which need conditions to be matched to produce new messages. The classifier (01101 & 10001 → 11010) produce a new message (11010) when (01101) and (10001) were posted. These new messages can be used by effectors to create actions or by rules to create new messages. A valuable action will reinforce the rule which has provided it. After a certain period of time, a genetic algorithm acts on rules to produce new rules and to strengthen the most efficient ones.



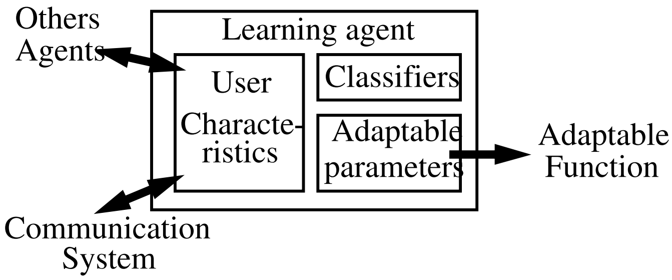
Classifier systems can be seen as neural networks [Smith94] where the internal messages represent internal states (hidden layers) and the external messages represent the input-output layer. Therefore, they have the same learning abilities but always provide evolving solutions. Moreover, rules can be easily listed to be analysed. This property could be very valuable to get an insight into what the system is really doing. Therefore this system with its embedded learning method fits our requirements.

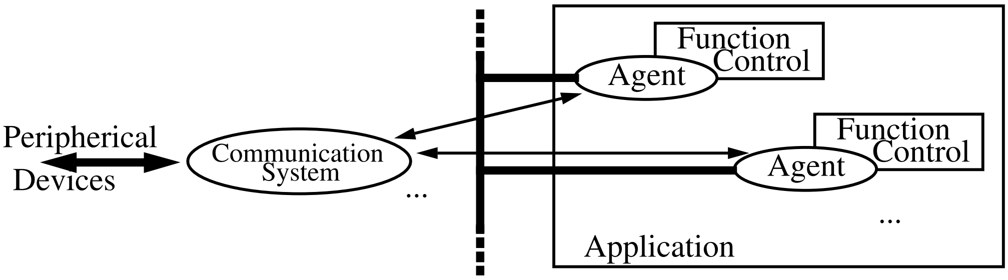
2) Learning agents introduction

In order to limit the drawbacks described above, we decided to build a set, taking into account some principles of classical adaptive interface with a learning system using classifiers [Maes96].

Hence we keep a well-known functioning. Information issued by the user and his environment is grabbed by a detection system. These pieces of information are stored in a smart way, in customed historics. For example, we use a historic which stores information on absolute and relative position of the virtual representation of the hand, and another historic for actions and cancellations. These historics help us decide which alterations are required. These alterations are estimated and then made.

We use an evolutionist system to select which alterations should be adopted. Then we define a micro-agent, fully autonomous, which is able to observe the different historics. This entity is able to alter the interface and the application parameters. The needs in expert knowledge are therefore limited, since the system is only altered by evolution. However this knowledge is not entirely suppressed. Indeed it is necessary to keep some of it, in order to initialize the learning mechanism. We can then solve the expert point of view problem and the rules initialization. We can modelize the set as follows:





The communication system manages the building of the different historics. It is the same system which hastens suitable interface agents, according to these historics. At any time, any of these agents can seek information in one of these historics, or can produce some alterations.

Each of these interface agents is linked with an adaptive functionality. It knows the parameters it deals with. Its job is to observe the system-user set of events and to alter its function parameters according to the user's preferences. Each agent can communicate with another one, in an indirect way, using communication module, or in a direct way, with a separate communication system. This second way of communication is exclusively reserved to information exchanges about the user's detected informations. Hence, as soon as some knowledge about the user is acquired, all the agents can use it in order to enhance the system.

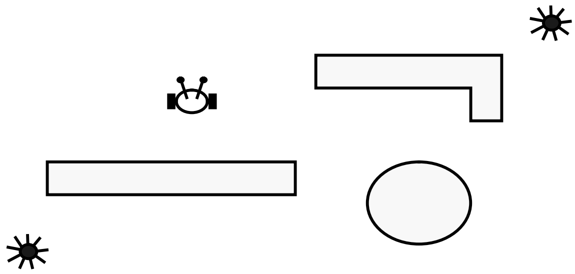
Accordingly, a set of autonomous entity is created, with learning and adapting capacities. We are thinking to use these entities to manage interface adaptation (thanks to its parameters) for the user. Now we are going to study how such an application can be developed with this system.

IV. Examples definition

Our main challenge was to verify the learning agent behaviour in a real situation, immersed in an environment and being used. First, we only use purely virtual entities and consider the relation between entity and environment instead of the human-robot relations. System learning liabilities are exactly the same in both cases but we suppress the human error source in evolving the evaluation. Further, we are going to see another example in which we use the same system in human-machine learning relation. That example also works in virtual 3D space.

1) Learning system

In this application, we dispose of a virtual autonomous robot, which can move in a virtual space [Luga96]. It has contact sensors which react when it touches obstacles.These sensors are placed in front of the robot and shield all its height. This robot cannot move back but is supplied with “turn right” and “turn left” functions.

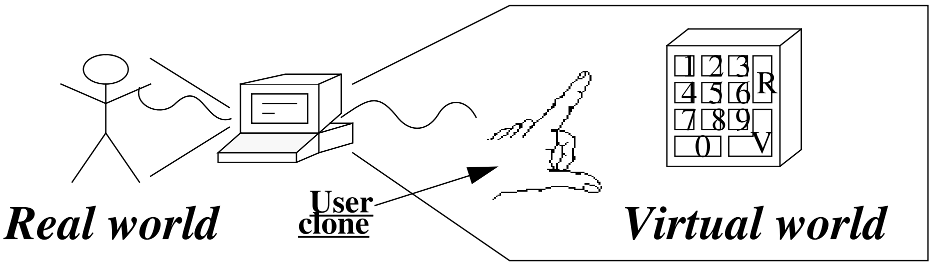


In the virtual world, we set obstacles (random placed) and goals to reach. The robot must walk around avoiding obstacles to reach its goals. In this environment we define two learning agents. The first one is attached to the robot and should define and adapt the “turn” order. It can use all the robot sensors’ information. The second learning agent is meant to make the robot reach its goal. This agent only knows the purpose and the robot’s position. This example shows us two positive characteristics about the learning agents system:

- it is able to resolve unknown problems. If all the obstacles are rectangle walls, when a round hole hinders its progress, the robot finds a solution.
- the communication between the agents in an identical system is very useful. Some problems cannot be solved by only one agent (right angle walls for example).

2) Human machine interaction

After observing the learning agents effects in 3D virtual world, we tried to apply them to human computer interaction. In this definition, the user is presented to the virtual world through its clone. This user clone can be considered like every environment entity. In this case, the learning agents are in charge of interface adaptation. In our simple example, we define a virtual drawing desk. The user can use several tools (pen, rubber, scissors, etc.) on a virtual paper sheet placed on the desk. We define a learning agents set to adapt this desk interface to each user.



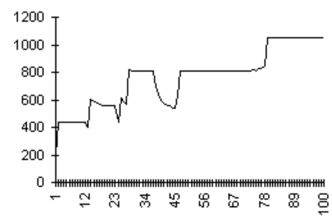
The first adaptation step concerns the user's gesture. Each user is going to use tools the way he always does. The interface first goal is to coordinate the gesture and its corresponding tool. As a matter of fact, the same movement can have a different meaning and provoke different actions to different users. This adaptation allows the user to maintain his own gesture. The application learns how to recognize the user. The second adaptation step works on the tool configuration. The interface is able to link a movement to a tool. It should then adapt each tool to the ability of each user.

To be compatible with classifiers system, all this application is binaries-messages-based. Each sensor watches a world parameter (in this case, an angle of the virtual hand) and posts messages which presents this parameter evolution. Effectors of the module start the action in the virtual world by launching the correct function. Each function of the application has an active sensors-effectors link and a learning micro-agent which can translate some of its rules in an active link. These links can be view as follow :

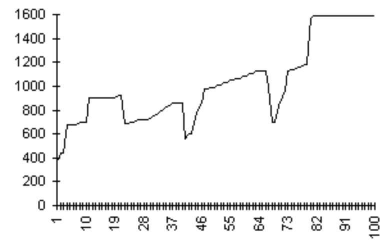
$$\underbrace{011101_t}_{\text{Increase of angle '011'}} \text{ <and> } \underbrace{011001_{t+i}}_{\text{Decrease of angle '011'}} \text{ <implicate> } \underbrace{100}_{\text{Scissors function}}$$

We decide to reward learning system rules according to active link action. The system estimates benefits or damages following this action and pay or tax classifiers depending on their similarity with the real link. When an active rule cannot please the user, the best classifier of the system is proposed to replace it.

The first figures we can present are about the learning system and its ability to recognize a situation. In association with a rule protection mecanism, we are using a hight crossover and mutation rate (about 15 percent mutation and 90 percent crossover). The safety device is able to protect a classifier which has proved its efficiency. In this way, this rule will take part in building the new generation (selection and reproduction steps) but will not be replaced by a new classifier. So, we can use a very hight mutation rate in order to maintain a good variety proportion in the system. We can see here the evolution of rules fitness in a such system.



Fitness



Generations

Generations

V. Conclusion

In spite of autonomous agents creation difficulties, learning agents present a solution for adaptive interface realization in virtual world. The problems we have encountered concerned to the classifiers system adjustments. Now we are going to work on how to minimize learning stages (which length is actually unknown). A solution seems to be a simplier learning system for the learning agents. We are also very interested in cooperation between two user.

VI. References

[Berthome94] BERTHOME-MONTOY A., “Building a user model for self-adaptive menu-based interfaces”, UM’94, International Conference on user modeling, Hyannis, Massachusetts, USA, 1994

[Crow93] CROW D., SMITH B., “The role of built-in knowledge in adaptive interface systems”, Proceedings of the 1993 international workshop on intelligen t user interfaces, Orlando, USA, 1993

[Goldberg89] GOLDBERG D. E. “Genetic algorithms in search, optimization and machine learning.” Addison-Wesley, Reading 1989.

[Holland75] HOLLAND J. “Adaptation in natural and artificial systems.” University of Michigan Press. Ann Harbor, 1975.

[Innocent82] INNOCENT P.R., “Towards self-adaptive interface systems”, International journal of man-machine studies,16

[Luga96] LUGA H., DESTRUEL C., DUTHEN Y., CAUBET R., “Adaptive interaction in virtual worlds using evolutionary algorithms”, IEEE Roman’96, Robot and human communication, 1996, Tsukuba, Japan

[Maes96] Maes P. “Modeling adaptive autonomous agents” in “Artificial Life: An overview” by Langton C. MIT Press

[Meyer94] MEYER B., “Adaptive performance support: user acceptance of a self-adapting system”, UM’94, International conference on user modeling, Hyannis, Massachusetts, USA, 1994

[Smith94] SMITH R.E., BROWN CRIBBS H., ”Is a learning classifier system a type of neural network?”, Departments of engineering science and mechanics, and electrical engineering, University of Alabama, 1994

[Sukaviriya93] SUKAVIRIYA P.N., FOLEY J.D., “Supporting adaptive interfaces in a knowledge-based user interface environment”, Proceedings of the 1993 international workshop on intelligent user interfaces, Orlando, USA, 1993

[Torguet95] TORGUET P, CAUBET R, “VIPER (VIrtuality Programming EnviRonment) - A Virtual Reality Applications Design Platform.” 2nd Eurographics Workshop on Virtual Environments, Monaco, 1995