Experiences from the Development and Use of Simulation Software for Complex Systems Education

Elpida S. Tzafestas*

Intelligent Robotics and Automation Laboratory
Electrical and Computer Engineering Department
National Technical University of Athens
Zographou Campus, Athens 15773, GREECE.
brensham@softlab.ece.ntua.gr
http://www.softlab.ece.ntua.gr/~brensham

Abstract. In this paper, we present our lessons from the design and use of two educational software tools for teaching behavioral modeling to graduate students of digital art. The tools, PainterAnts and VLab, allow the experimentation and control of simulated ants that paint, or Braitenberg vehicle-like agents (Braitenberg 1984). To better address the target audience of digital art students, we have transcribed both systems in ways that would excite art students, that is, by finding graphical equivalents of behavior. This arrangement allows us to exploit the users' visual experience and motivation to manipulate and experiment with complex visual forms. Experience with using the systems reveals that the artists show a high motivation for experimenting with them, which is partly due to the fact that they tend to regard them as simple abstract art tools that may produce interesting complex forms. Our experience has also identified several methodological and theoretical issues on educational system design that have been partly tackled and that merit extended study and research.

1 Introduction

Within the framework of a master in digital art, we have developed a set of educational tools for artificial life and the complexity sciences. These software tools constitute a laboratory curriculum that is used to supplement the theoretical courses on the subject and contain, among other things, an ant-based painting tool, called PainterAnts (Our reference 1), and an educational tool for behavioral modeling, called VLab (Our reference 2). "PainterAnts" is a special-purpose ant population demo that tries to convey and teach regulation principles using graphical means. VLab allows the experimentation and control of simulated robotic agents that are Braitenberg vehicle-like, and its purpose is to make students familiar with simple behavioral modeling.

To better address the target audience of digital art students, we have transcribed both systems in ways that would excite art students, for example in Vlab we have introduced an additional behavioral parameter, the brush, used by the simulated agents to draw/paint

* Also with: Digital Art Laboratory, Athens School of Fine Arts, Peiraios 256, 18233 Agios Ioannis Rentis, GREECE. tzafesta@asfa.gr, http://www.asfa.gr/~tzafesta

while moving around. This arrangement allows us to exploit the users' visual experience and motivation to manipulate and experiment with complex visual forms.

Both tools have been developed in Java on top of the RAGS software (Recursive Agents Simulator), that is a testbed for simulation and experimentation in artificial life. RAGS allows the recursive definition of agents that are composed of other agents at any depth and the simulation of such recursive multi-agent systems. Furthermore, it allows the definition of worlds with different spatial properties and the definition of observers of statistical nature (for instance, data curves). Apart from these features, RAGS comprises a set of interfaces that may be invoked automatically and dynamically during the system operation. These interfaces allow the control of the simulation, the edition and/or processing of all simulation components (agents, observers, worlds, etc.), as well as a limited set of graphical programming functions.

In a more general line of work, the education of artificial life with the aid of graphical tools allows us to put the user in the place of an external modeler that tries to understand the operation of a system by forming a model of the system. In this sense, our long-term objective is to study the relation between artificial worlds and the models that humans construct about them.

2 PainterAnts

The goal of "PainterAnts" is to support teaching of regulation principles and their effects by studying patterns of color in space. These patterns are generated in a distributed way by a population of simulated agents, called painter ants.

The painter ant model is inspired from the model of an ant that gathers food samples and brings them back home. Since there are generally many fairly big food sources in the ants' environment, the solution to the problem of food gathering is to allow the ants to deposit and pick up chunks of another substance ("crumbs"). Crumbs therefore serve as a communication means between agents and resemble the pheromones used by real ants. In (Tzafestas 1998) we have presented an ant model that bypasses the problems presented by previous models, by using a mechanism of internal crumb regulation. This model has been transcribed to an ant painter model by replacing the variable "crumbs" with a variable "color", so that a painter ant deposits or picks up a quantity of color, while seeking to bring food home. It is supposed that each position of the grid-based world has a color that may be modified during execution.

The main interface of PainterAnts is shown in figure 1 and has two parts, the simulation view or ***system view*** (right) and the processing/editing panel or ***control view*** (left). The user may modify the agents' parameters from the control view or from specialized popup dialogs. In every case, the user may modify among other things the current color of an ant. The control view allows also the modification of other simulation or worlds parameters, such as the visibility of agents and objects, the selective visibility of color components (R, G, B), the addition or deletion of food, the background color etc. Finally, the popup simulation view menu gives additional possibilities, such as saving. Some visual results of using the painter ant system are given in figure 2.

3 VLab

The goal of "VLab" is to support teaching of behavioral modeling by studying colored robotic paths in 2D space. These paths are generated by one or more simulated robots or "vehicles", which can draw/paint on the canvas while moving around.

A simplified robot model, inspired from the Khepera robot (Mondada et al. 1993), was adopted for supporting the vehicle models developed and examined. Each vehicle's sensors perceive stimuli sources and are directly connected to its motors that control motion, without elaborate processing. The topology of the connections between sensors and motors as well as the properties of those connections give rise to diverse behavioral phenomena that have been identified and studied by Braitenberg (1984). Various Braitenberg-type vehicle models are supported by the system. Beside behavioral functionality, each vehicle is endowed with a programmable "brush" that its uses to draw/paint on its environment while moving around.

The main interface of VLab is shown in figure 3 and, just like PainterAnts, it includes two areas : the simulation visualization area or ***system view*** (right) and the editing/processing area or ***control view*** (left). The user may modify the vehicles' parameters from the control view or from specialized popup dialogs. More specifically, the user may select the vehicle type, edit its parameters (velocity, sensors' range, etc.), control the type and intensity of environmental stimuli and program the brush. Finally, the popup simulation view menu gives additional possibilities, such as saving. Some visual results of using VLab are given in figure 4.

4 On users and learning

The students work was to try to understand the systems' operation as well as the individual effects of each of the ant or vehicle model variants. Around this central educational goal of us, another three satellite goals of a methodological nature appeared :

- ***Learning to experiment.*** The artists, having practically no formal or even fundamental scientific background, demonstrated a tendency to play around with the system facilities, without any discipline or organization of observations and experiments. It was immediately necessary then to have them follow manually certain predefined experimentation protocols. The subsequent version of the tools will encompass a set of predefined experiments with assorted protocols that will be launched automatically upon user request. Since this particular need has been identified, it will become necessary in the near future to clearly define in a programming language (preferably a visual one) the concept and the structure of an experiment.
- ***Learning to observe.*** Once the experimentation problem was identified and manually tackled, we realized that it is even more difficult for the students to concentrate on the detailed examination of the visual results of an experiment, for instance for comparing two images and deducing reasonable qualitative results. Keeping action/modification record files, as well as a very limited user action recording facility, allowed the students to draw a few qualitative conclusions. Our medium term

goal is to introduce to the system a component for user-definition of the observation method, for instance to compare the results after a thousand execution steps. This component will have to rely on the same visual programming language as the experiment description component mentioned earlier.

- ***Learning to learn.*** Finally, after having completed long series of experiments and having observed the similarities and differences between them, a major part of the users remain indifferent to the relation between a model and a result. What is missing in our default methodology, is the personal involvement or participation of the user in the system operation, which would incur a higher motivation for learning. In a future version of the system, we plan to introduce to the system a set of specialized criteria, so that different categories of users will have different but equally high motivations to use the system and learn behavioral modeling principles, for instance ask to artists questions such as “What can we do with only pure colors ?”.

Despite those methodological problems, we realized that the artists finally arrived at establishing a stable user relation with both systems, even without having well understood or evaluated the underlying behavioral models. The characteristics of this user-system relation are as follows :

- ***Learning to use.*** In the beginning, the artists take the place of an explorer-user, who starts manipulating and evaluating forms (indirectly, via experimentation with the system parameters) mostly randomly. Then, they stabilize on a set of created forms idiosyncratically selected, but they are ready at any moment to abandon the selected form and restart from the beginning if they are not happy with the current state of the form. Instead of learning to experiment with the systems and to observe, the artists learn to use the systems as simple form creation tools, without actually controlling them or even trying to control them.
- ***Desiring to modify the systems.*** After having acquired some experience with the systems, the artists start desiring to create forms similar to the ones already created and start searching blindly for the proper behavioral configuration. At this point, we intervened and asked questions of the type : “How can you be sure that the form that you have in mind exists and may be created with this system ?”. Hence at this point, the artists start to really want to learn what the systems do, i.e. to describe and understand the range of forms that the systems can support. As a consequence, the artists start wanting to modify the systems according to their tastes, even if the relation of a model with the resulting form is still little appreciated.

5 On tools and design

Simple behavioral models have been used by Resnick (1994) for education in the high school, while some educational applications have been presented by Pagliarini et al. (1996). On top of the purely educational achievement, our own experience with both educational tools has revealed several technical and theoretical issues involved in the design and development of educational software for artificial life and the complexity sciences at an undergraduate or postgraduate level.

An educational software tool has to encompass both general- and special- purpose

functions. General-purpose functions are necessary to support education of the subject addressed (here, it is behavioral modeling in both cases), *independently of the target audience*. Special-purpose functions are necessary to respond to the particularities of the audience (here, it is a class of graduate students in digital art). For example in VLab, the vehicles behavior library and the simulator are reusable in any university-level class, independently of discipline, whereas the brush models and the visual experimentation protocols are specially designed for classes of art students. Of course, we expect to have a different balance of general-versus special- purpose functions for different education subjects and audiences. On the other hand, thoroughly special-purpose educational software tools for particular audiences and a given subject, work sometimes better, but they are rare. What is truly necessary, is a modular software methodology that will allow customization of given tools for different audiences. We expect to be able to tackle this issue after acquiring experience with particular educational software tools, adapted to different user classes.

Another issue that has been already mentioned, is the need for involvement or participation of the user to the system operation, to ensure high motivation for learning. This can be done in a number of ways, such as an intrusive adventure game that addicts the user or an artistic creation tool that attracts a user who happens to be an artist. A methodology that invites participation in a particular educational tool is generally special-purpose for a given subject and audience. Such an approach that seeks to design participatory environments, has the special feature that teaching (teachers) and learning (students) are separate, and what actually connects the two is the software tool. It is the role of this tool to translate what the teacher says to what the student understands, and vice versa.

Education and learning, through participation or otherwise, is not enough. What we would like is to integrate a particular software-driven course to the general educational policy that applies to the target audience. Integration means that students should be able to use and reuse ideas and material in other contexts within their educational environment. In our case, we have observed that students are systematically importing images created with PainterAnts and VLab to other tools that they are using, namely standard tools, such as Adobe Photoshop and Macromedia Director, or home tools created in our laboratory. This is an indication that the particular educational tool has gained recognition among our students as a valid image creation and processing tool.

The latter observation also unravels a difficult issue, that of the objectiveness of education. What exactly do we expect students to learn ? Is learning something objective or something profoundly personal ? While we think it is impossible to give a general and definite answer to this question, we have fewer problems of such philosophical nature in our case. This is because education of artists relies precisely on personal acquisition and expression by each student, and only rarely on objective or systematic views, at least when imagery is involved.

6 Conclusion

We have presented two educational software tools for teaching behavioral modeling to graduate students of digital art. The systems, called PainterAnts and VLab, use ant

models or Braitenberg vehicle behaviors, respectively, and special graphical representation. The Java-based environment on which both tools are programmed provides many features for editing/processing the behavioral parameters and the environment, as well as visualization, recording and saving functions. Our experience with using the systems revealed that the artists show a high motivation for experimenting with them, which is partly due to the fact that they tend to regard them as simple abstract art tools that may produce interesting complex forms. Those forms are possible thanks to the versatility of the underlying models. We have also identified several methodological and theoretical issues that have to be addressed by large-scale educational software tools, for instance by the future expanded version of PainterAnts and VLab. Those issues include the balance that has to be found between general- and special- purpose functions and the need for active participation of the user to the system operation so as to ensure high motivation for learning. We also identify the problem of integration of a software-driven educational process to the general educational environment and policy applied, as well as the question of objectiveness of learning.

References

Braitenberg, V. (1984) *Vehicles: Experiments in Synthetic Psychology*, MIT Press, Cambridge, MA.

Mondada, F., E. Franzi, and P. Ienne (1993) Mobile robot miniaturisation: A tool for investigation in control algorithms. *Proceedings Third International Symposium on Experimental Robotics*, Kyoto, Japan, October.

Pagliarini, L., H. Hautop Lund, O. Miglino, D. Parisi (1996) Artificial life: A new way to build educational and therapeutic games, *Artificial Life V, Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, C.G. Langton and K. Shimohara (Eds.), MIT Press, 1996.

Resnick, M. (1994) *Turtles, termites and traffic jams: Explorations in massively parallel microworlds*, MIT Press/Bradford Books, Cambridge, MA.

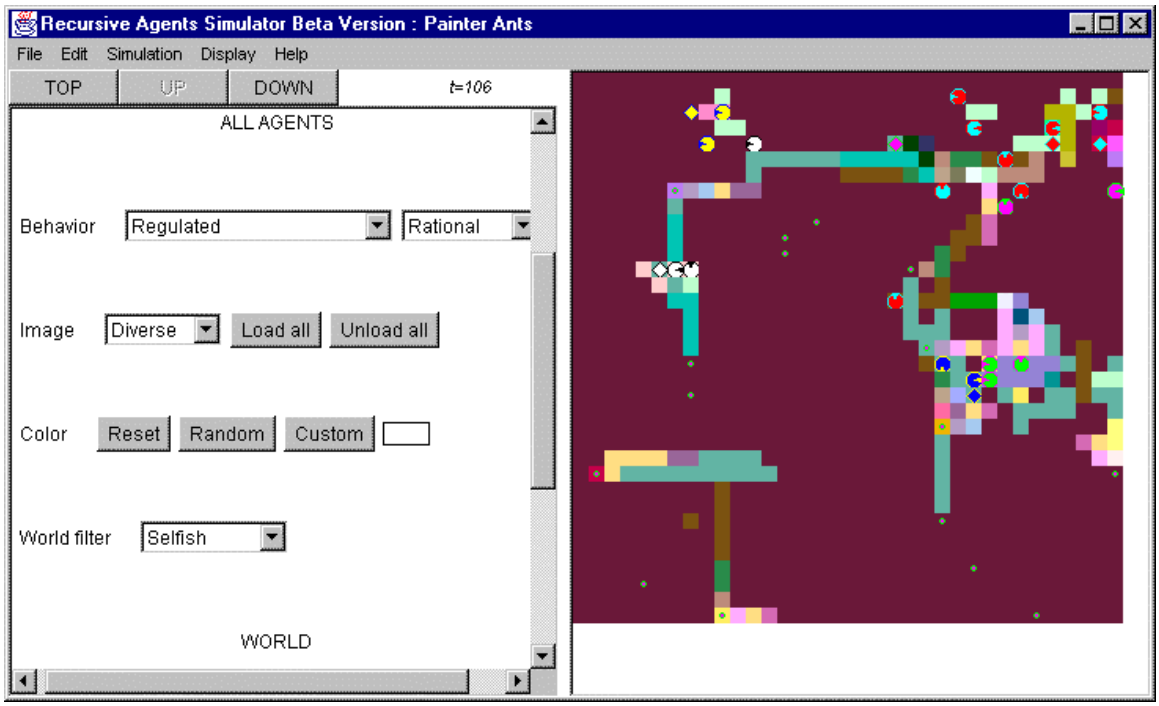

Figure 1. *Main interface of "Painter Ants"*

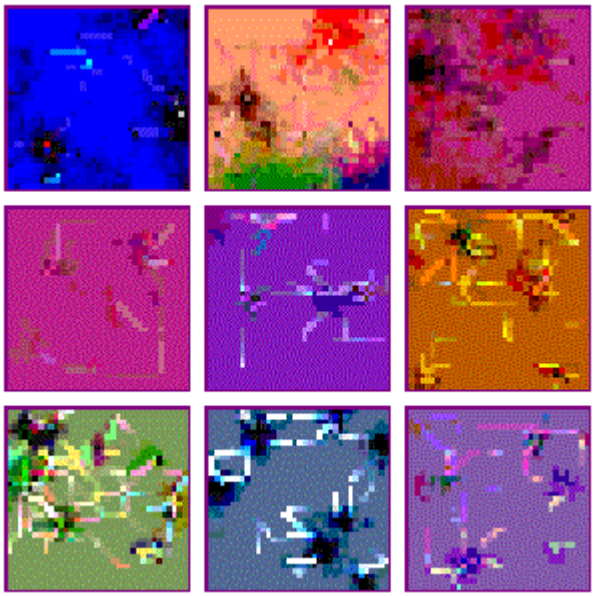

Figure 2. *Some snapshots taken with "Painter Ants" (http://www.softlab.ece.ntua.gr/~brensham/PainterAnts/)*

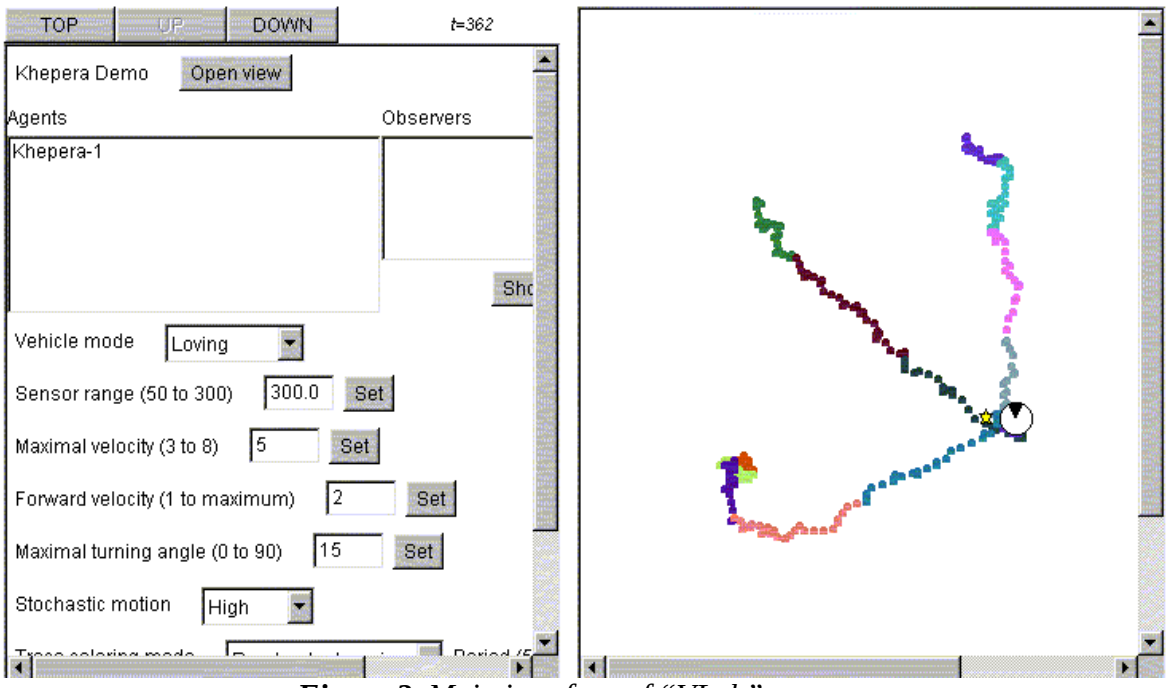

Figure 3. *Main interface of "VLab"*

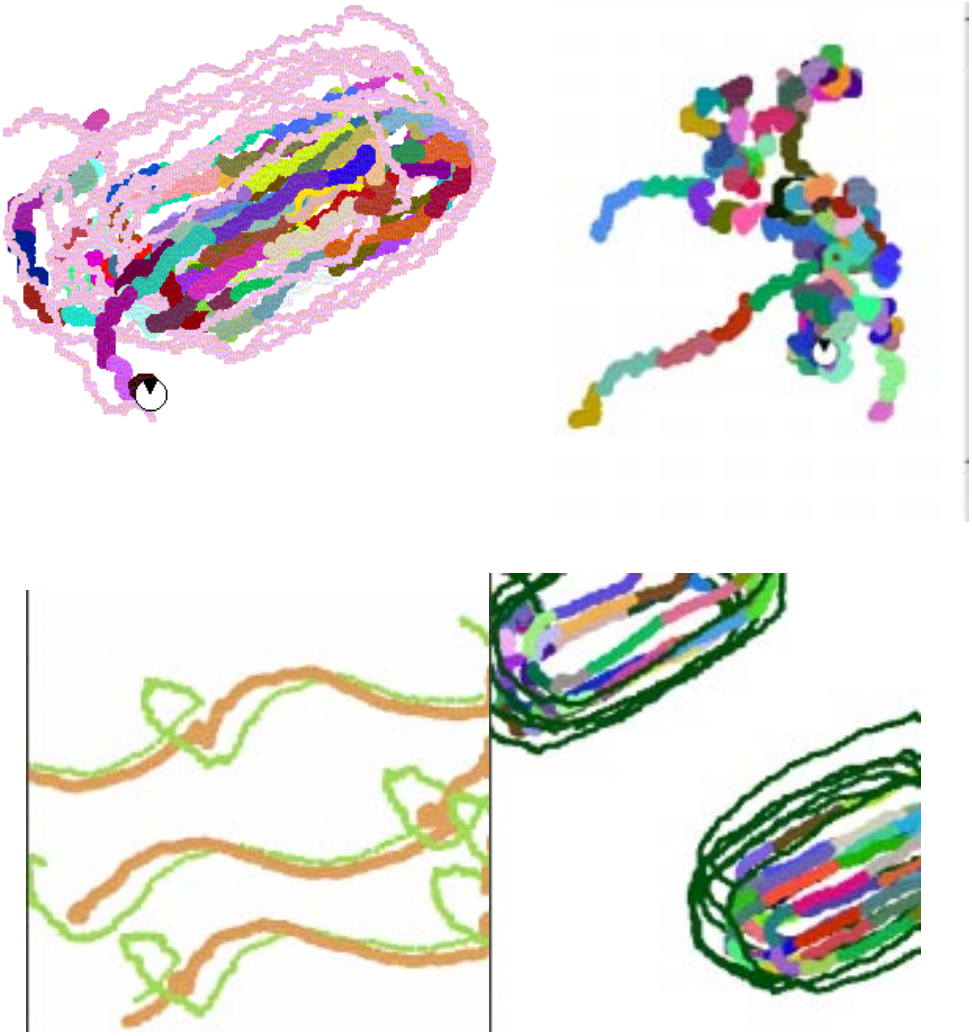

Figure 4. *Some snapshots taken with "VLab"*