Modeling Virtual Ecosystems with the Proactive Guidance of Agents

Bedřich Beneš and Enrique David Espinosa
ITESM, Campus Ciudad de México
{Bedrich.Benes | Enrique.Espinosa}@itesm.mx

Abstract

In mainstream geometric modeling, cultivating virtual plant ecosystems is a difficult task. Algorithms for realistic scene generation are rooted in procedural models with no explicit or poor external control. We propose that virtual ecosystems modeling may be boosted using software agents as behavioral tools. An ecosystem grows and is driven by its internal rules of development. If it is left to its own fate, it will reach stability on the edge of chaos. Agents interact with ecosystems by adding plants, cutting or killing them, watering, stepping-over, or favoring some plant species. An agent is a characterization artifact that shows proactive conduct and is described by its set of sensors, effectors, internal states, and habits. Habits are defined as continuous functions and allow for characterizing a wide variety of behaviors.

1. Introduction and Previous Work

This paper shows that a set of virtual agents interacting in and with a virtual environment can be used as tools for geometric modeling. In mainstream graphics technology, cultivating plants as a geometrical model is a difficult, but an important task in realistic scene development. Without it, artificiality shows up in graphical depictions of these models. In our case, behavior exhibited by growing, dying, plus shape and size, must be realistically modeled by computerized models. For this purpose, we introduce a concept of a virtual (software) agent that can sense an environment and react to it by providing high-level behavioral states. Such reaction is a result of external stimuli, internal states, and habits of the agent. The external stimuli are perceived by sensors, transformed into signals and processed. Habits are described as multivariate continuous functions and are diverse in a random fashion as an agent is created and exhibits personality. The internal states are set in a feedback loop based on the stimuli, habits, and the previous values of the internal states.

The combination of agent-based conduct automation plus graphical modeling of an ecosystem development according to its own internal rules, yields in innovative approach to virtual realism. This makes up for the key relevance in our work. Similar work has been proposed by researchers on artificial life using genetic models as their behavior framework, but their simulations lack the realism of virtual models. For this reason, we believe that our mixed approach is unique.

1.1 Agents in Virtual Ecosystems

Plant ecosystem modeling has been introduced into computer graphics within last few years. Probably the first paper dealing with ecosystem modeling and rendering was introduced by Deussen *et al* [3]. The later works include plant competition for space and resources [1, 7].

Agents interacting with plants were first described in the context of L-systems in [10]. Another paper dealing with more elaborated models of insects attacking plants has been recently published [4]. The paper presents examples of a formal specification by means of L-systems of insects interacting with plant or plants in different ways, ranging from a single insect foraging on a plant to insects flying in an ecosystem. The paper also discusses plant response to a damage, behavior modeling, insect perception, etc.

1.2 Agents in Another Context

Agents have been around for a number of years. It is hard to establish a clear definition for a software agent because of its diverse mainstream characteristics: proactivity, independence, capacity for reasoning under uncertainty, negotiation, collaboration, and mobility, can all be placed on other types of software machinery.

Most efforts in agent technology have dealt with information systems. For example, Maes and Moukas developed the `Firefly` project as an early purchasing broker prototype [8] using a number of ontologies such as probabilistic reasoning, neural networks or reasoning under uncertainty as a software engineering approach [5].

Virtual agents were mentioned and used many times in computer graphics. One of the pioneer's work was the paper

Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA'03)
1087-4844/03 $17.00 © 2003 IEEE

of Reynolds [11] who introduced a simulation of polarized behavior of flocks of birds and fishes. These agents called *boids* have a wide usage and assisted for example in the making of the Batman Return and The Lion King movies. Laws in these domains are more on the side of physical and natural phenomena. They are, to some extent, more certain, and can be described functionally. Two levels of realism are thus confronted: *behavioral*, and *virtual*. The following discussion deepens into the latter. At the end of the paper, we will go back to the former.

Sims has introduced evolved artificial creatures [12]. These amazing virtual animals are generated by genetic algorithms where the fitness function is evaluated by competition. The winners are crossed so the best features are inherited to the future generations. The emergent phenomena are the strategy of behavior and the creature's shape.

Terzopulos *et al* [14] have introduced a virtual aquarium with artificial fish. The model of a fish is based on physics and is described as a set of springs. The motion occurs when the springs act and the water generates corresponding response moving the fish forward or up and down. The fish has three mental stages (hunger, libido, and fear), sense the environment, and the intention generator generates responses that drive the motion.

Thalmann and his collaboratives have an impressive ongoing work on artificial agents in VR [6, 9, 13] used extended physics-based modeling to introduce a wide variety of virtual humans with feelings, emotions, etc. They communicate, visit the supermarket, feel and sense environment, travels in metro, ride skateboard, and so on.

Badler and his collaboratives developed the EMOTE project [2] that focuses emotions expression and gestures of virtual agents.

For this paper, we will establish that *an agent is a characterization artifact capable of delivering proaction guidance for a graphical character that must exhibit some sort of high level behavior inside a virtual environment*. Information processing is still part of the job, of course. What we propose as an innovative use for agents is their melding into a virtual modeling engine where an ecosystem will be generated. This novel approach to generating artificial life builds on top of work such as the *boids* described in the middle of this section.

In the next section, the ecosystem modeling is briefly described. The Section 3 describes completely the agents and their behavior. The next section deals with the agent description and the next section 5 shows some results of the simulations and the last Section 6 concludes the paper.

2. Ecosystems

A virtual ecosystem is a planar continuous area where different virtual plant species exist. It is supposed the conditions are equal in all directions and all places. Plants have the same amount of soils, nutrients, light, water, etc. *i.e.*, they have the same initial chance to survive.

There are two basic approaches used for the ecosystem modeling [7], the global-to-local where the plant distribution is given once by user interaction, for example as a gray-scale image. We use the second approach, the local-to-global. Here a scene is described and the plants are planted. Plants "know" their rules of behavior and grow according to them. Since this is a typical artificial life approach we focus on the emergent phenomenon that is the spatial plant distribution. Figure 6 (top) shows a result of such a simulation.

In the local-to-global approach, the plants develop according to the local rules of behavior and compete with the other plants for space and resources. The competition is detected as a collision of ecological neighborhoods that are circles corresponding to the area of influence of a plant. As the plant grows, the neighborhood is changing. If two neighborhoods interpenetrate there is a collision, the winner is determined and the looser is either diminished or eliminated from the simulation. The aim of this paper is not to describe the plant ecosystems, so we refer reader to [1, 7] for precise description of the techniques used here.

3. Agents

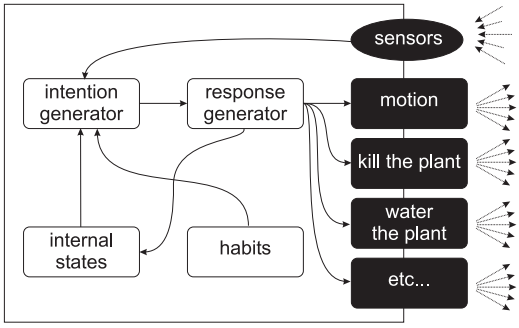

Figure 1. Internal structure of an agent

We consider an agent as a characterization artifact capable of delivering proaction guidance for a graphical character that is situated and acts in a virtual environment. It has its position defined by the two coordinates $[x, y]$ and orientation defined by its viewing direction.

The agents work in the following way (see in the Figure 1). They sense their environment with the sets of sensors that generates stimuli signals that are transported into the intention generator. The intention generated is also fed by internal states that are a set of variables corresponding to certain feelings (laziness, tiredness, hunger, etc). Habits are the last input of the intention generator. They are represented a set of functions describing a probability that a

certain action will be performed. There are also global variables like time, rain, and so on.

The intention generator sends a signal to the response generator. The response feeds back the internal states, for example a signal for walking very far will increase tiredness. At the same time the effectors perform the corresponding actions.

3.1 Sensing

An agent senses its environment by a set of sensors. We have implemented agents based on the visibility sensor, but the concept introduced here allows for arbitrary extension.

The area that each agent sees is called the *visibility range* and is defined by the viewing direction, the viewing angle α, and the distance of visibility d. The visibility angle is fixed and the distance d can vary according to the real conditions (day/night, fog/clear air, tiredness, etc.).

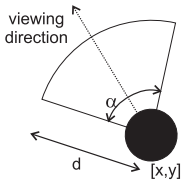

Figure 2. The visibility range

3.2 Motion

An agent performs a goal-oriented walk. The agent moves to the closest plant or plants and performs actions generated by the response generator. If there is no action to do the agent performs "idle" task, i.e., continues in the viewing direction and, after some time, it stops for a while, continues in a random direction, etc.

The typical motion of gardeners is from plant to plant as well as farmers eliminating weed. Farmers in the time of harvest walk in the direction of the high plant density.

3.3 Internal States

Internal states are variables defined by user. These variables depend on response generators that can change them immediately but they also depend on time and other variables, like the distance passed, day/night etc.

function	Figure
$f_a(x) = 2x^3 - x^2$	4a)
$f_b(x) = 2x^3 - 5x^2 + 4x$	4b)
$f_c(x) = 6.75x^3 - 13.5x^2 + 6.75x$	4c)
$f_d(x) = -6.75x^3 + 6.75x^2$	4d)
$f_e(x) = -64x^2 + 64x - 15$	4e)
$f_f(x) = -64x^2 + 48x - 8$	4f)
$f_g(x) = -0.2x + 0.9$	4g)
$f_h(x) = -2x + 1$	4h)
$f_i(x) = -2x^3 + 3x^2$	4i)
$f_j(x) = 6.75x^3 - 6.75x^2 + 1$	4j)

An example of an immediate change of the internal status is an agent eating grass. At the moment the grass is consumed, the value of the "hunger" variable is decreased. This variable is increased by time. The Figure 4 and the table show the functions we use in our implementation either to change internal states or to define habits. These functions allow agents to select their own internal state in an autonomous manner, and thus characterize the beforementioned proactive guidance provided by software agents. For simplicity we will refer to these functions as the function Figure 4 x), instead of $f_x(x)$, because the graph of the function can be read more easily.

3.4 Actions and the Intention Generator

Most important are the actions that can be performed by agents. All the functions that we use are polynomial because they can be promptly evaluated. All the functions are normalized. The following example explains a generated action.

Let's say the function in Figure 4a) describes the probability that the agent-animal will eat a plant. Here the variable x corresponds to the internal state - the hunger of the animal. When the agent encounters a plant this function is evaluated and a random number is generated, if the random number is smaller than the $f(x)$ the plant is eaten.

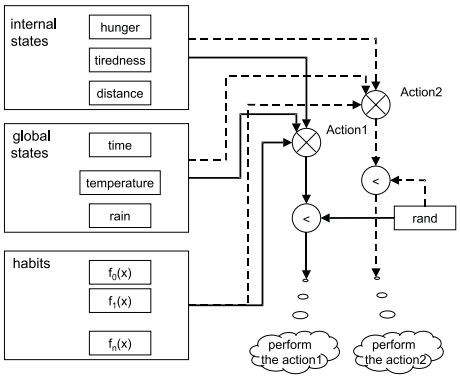

Figure 3. The action generator

Figure 3 shows the entire process how the actions are generated. Each action can depend on internal states, global states, and habits. The states are inputs of the above-described functions and all meets in the "cross" operator. Here the probability of the action is calculated and scaled according to the internal states of the agent. The output of the operator is compared with a random number. If the random number is smaller the action is performed. Let's demonstrate it on another, more precise, example.

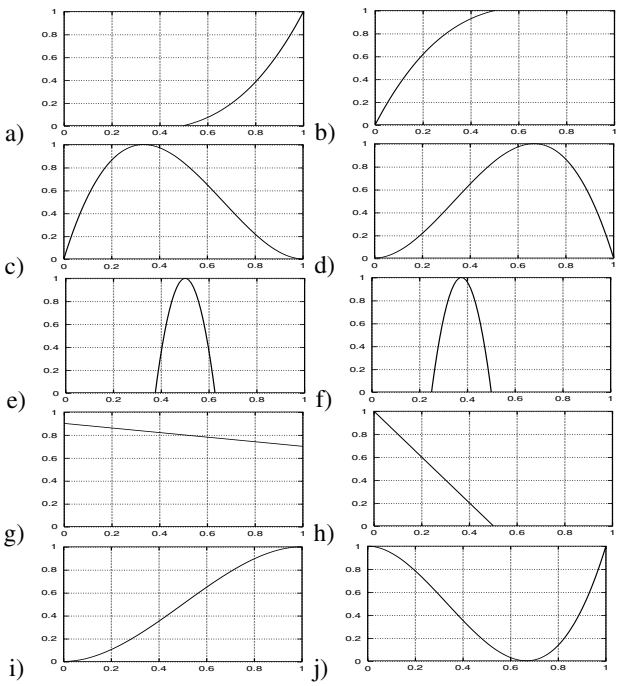

Figure 4. A set of functions describing needs, feelings, habits, and changes of the internal states of virtual agents

A farmer-agent works for four hours. Its tiredness depends on the normalized time it works and is described by the function Figure 4 g). The probability that the agent will kill the weed is described by the function Figure 4 d) that depends on the plant normalized age. In other words, the young weed cannot be seen very well and the probability to be eliminated is small. If the weed is too old, it can be left on the field, because it will die anyway. Suppose the agent is in the middle of its work (normalized time $t = 1/2$) and encounters a weed that has its normalized age equal to $3/4$. The tiredness function will give value $4/5$. The probability function of the plant will return 0.95. Multiplying these values gives the probability that the plants will be killed by this agent equal to 0.76.

4. Habits and States Definition

The above-described concept is not difficult to implement. A coding problem is the way a specific agent is programmatically described. We use a XML scene definition file that is processed when the program starts. An example follows.

$< global >$
t=0 Δt=0.5
rain=0 Δrain=20+rand(5)
p_1=daisy p_2=grass p_3=bush p_4=wheat p_5=corn
$< /global >$
$< agent1 >$#farmer
states: hunger=$f_a(time)$ hunger=$1 - f_a(food)$
tiredness=$1 - f_b(time)$
kill: $p_1 = f_d(age(p_1))$ $p_2 = f_d(age(p_2))$
$p_3 = f_h(age(p_3))$ $p_4 = 0$ $p_5 = 0$
water:$p_1 = 0$ $p_2 = 0$ $p_3 = 0$
$p_4 = f_e(dry)$ $p_5 = f_e(dry)$
$< /agent1 >$
$< agent2 >$#lawnmaker
states: hunger=$f_a(time)$ hunger=$1 - f_a(food)$
tiredness=$1 - f_b(time)$
kill: $p_1 = 0$ $p_2 = 0$ $p_3 = 0$
$p_4 = f_d(age(p_4))$ $p_5 = f_d(age(p_5))$
cut: $p_1 = 0$ $p_2 = f_b(age(p_2))$ $p_3 = f_h(age(p_3))$
$p_4 = 0$ $p_5 = 0$
water:$p_1 = f_b(dry)$ $p_2 = f_b(dry)$ $p_3 = f_b(dry)$
$p_4 = 0$ $p_5 = 0$
$< /agent2 >$

This is an incomplete description and demonstrates just the essential actions. The global section defines the time step of the simulation $\Delta t = 0.5$ day. Rain will occur every 20 ± 5 days. It also defines a set of plants called p_i, $i = 1, 2, \ldots, 5$.

Each agent's description starts with definition of the internal states (hunger increases with time and decreases with food) and continues with the action description.

The first agent has two actions defined for each plant: killing and watering. The agent will kill daisies and grass that are considered as a weed if they are in the middle of their productive age. Small plants can be skipped and old ones are omitted. The bushes are treated in a different way. If they are small they are eliminated but grown bushes are left in the field.

The second agent, the lawnmaker, is interested in daisies and grass and the other plants are considered as weed. The grass can sometimes be cut. The same applies to bushes.

The agent's description is easy to understand. The problem we have detected is checking the consistency of the rules. It is very difficult to guess from the behavior that some rule is wrong or does not apply.

5. Implementation and Results

The entire system is written in C++ and runs under Windows and UNIX. We use OpenGL for preview and Persistence of Vision raytracer (www.povray.org) for photorealistic images. The runtime of a simulation of ten years growth

of one scene with average 80 thousands plants with the half day time step takes on 1GHz IBM PC less than ten minutes. Ray tracing of the resulting scene (80 thousands plants and one million geometric objects) requires more than 500MB of memory and takes less than 20 minutes. The simulation area was $20m^2$.

The ecosystem grows respecting the biological rules and agents enter every week for four hours. They perform their tasks and leave the ecosystem that continues growing. Since the time step of the ecosystem development is half day we suppose the plants not to be growing while the agents are working. The average number of agents was ten.

The following example demonstrates an application of agents to a virtual ecosystem. An ecosystem that includes wheat, three different kinds of grass, some bush-like plant, and campanula (blue bell) is influenced by different agents. Figure 6 (top) shows a close-up of the ecosystem after ten years if it is left without influence of any agent. The grass dominates the area and the other plants are occasionally presented there. The average number of plants of one species is constant but can vary during the time. It is very rare if some plant specie excints without an external influence. The system is stable.

The next figure shows the same area after half year of treatment by agents-farmers that eliminate everything else than wheat (grass, bushes, and flowers). The area is dominated by wheat but at some places some other plant species appear. The others plants cover at most 2% of the area of the ecosystem. This kind of ecosystem is very unstable because the other plant species have very high probability to survive.

The example in Figure 6 (down) shows the same area after an intensive influence of a set of ten agents lawnmakers. The area is mostly covered by grasses but some other plants at some places appear (mostly campanulas). The area consists of 80 thousands grass blades, the other plants cover 1%.

Figure 5 shows similar ecosystem that was plundered by walking herd of virtual cows.

6. Conclusions

A new approach to modeling using virtual agents is presented. The aim is to provide to user a tool that permits to model phenomena that are very difficult to model by hand or by other techniques. User decides which plant species will be presented and the ecosystem is randomly seeded. Then the user "programs" the virtual agents that enter the ecosystem and perform their work. At the same time the ecosystem develops according to the governing biological rules. The result is a model of a 3D scene.

The agent description presented here is a quite low-level since the user must specify functions that influence the agent's behavior. We believe that the agent description introduced here is easy to understand and can be applied very fast. It is possible to write libraries of agents that could be used later.

The majority of the work in the system implementation was devoted to the modeling the plants and calibrating the ecosystem. The agent-based modeling in this paper is linear and is not very complex in its programming. We believe that the memoryless agents that react to external stimuli can be used efficiently to model some difficult tasks, especially when social or cognitive factors are taken into consideration. This is currently being undertaken as future work in reasoning on uncertain domains that may deliver additional realism to virtual environments like the ones presented in this paper.

Future work also includes agents that are able to learn. For example an agent can evaluate some area, enter it, and perform certain actions. Consequently, it can compare the results with the estimation and assign corresponding weights to the actions. In this way the agent can modify its internal functions. Diverse machine learning techniques may be applied in conjunction with the behavioral ontologies discussed throughout this paper.

As a result, future and improved algorithms and heuristics may prove to effectively aid in boosting virtual modeling where artificiality is diminished.

References

[1] B. Beneš. A Stable Modeling of Large Plant Ecosystems. In *Proceedings of the International Conference on Computer Vision and Graphics*, pages 94–101. Association for Image Processing, 2002.

[2] D. Chi, M. Costa, L. Zhao, and N. Badler. The EMOTE Model for Effort and Shape. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 173–182. ACM Press/Addison-Wesley Publishing Co., 2000.

[3] O. Deussen, P. Hanrahan, B. Lintermann, R. Měch, M. Pharr, and P. Prusinkiewicz. Realistic Modeling and Rendering of Plant Ecosystems. In *Proceedings of SIGGRAPH'98*, Annual Conference Series 1998, pages 275–286. ACM Press, 1998.

[4] J. Hanan, P. Prusinkiewicz, M. Zalucki, and D. Skirvin. Simulation of Insect Movement with Respect to Plant Architecture and Morphogenesis. *Computers and Electronics in Agriculture*, 2002, to appear.

[5] N. Jennings and M. Wooldridge. Agent Oriented Software Engineering. In *Handbook of Agent Technology*, pages 23–29. AAAI/MIT Press, 2000.

[6] M. Kallmann and D. Thalmann. Direct 3D Interaction with Smart Objects. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 124–130. ACM Press, 1999.

[7] B. Lane and P. Prusinkiewicz. Generating Spatial Distribution for Multilevel Models of Plant Communities. In *Proceedings of Graphics Interface'02*, Volume I, pages 69–80, 2002.

[8] M. Maes, R. H. Guttman, and A. G. Moukas. Agents that Buy and Sell. In *Communications of the ACM*, pages 81–98. ACM Press, 1999.

[9] R. Musse, F. Garat, and D. Thalmann. Guiding and Interacting with Virtual Crowds in Real-Time. In *Computer Animation and Simulation'99*, Springer Computer Science, pages 23–34. Springer–Verlag Wien New York, 1999.

[10] P. Prusinkiewicz, M. James, R. Měch, and J. Hannan. The Artificial Life of Plants. In *SIGGRAPH '95 Course Notes*, volume 7, pages 1-1–1-38. ACM SIGGRAPH, 1995.

[11] C. Reynolds. Flocks, Herds and Schools: A Distributed Behavioral Model. In *Proceedings of SIGGRAPH '87*, volume 21(3) of *Annual Conference Series*, pages 25–34. ACM Press, 1987.

[12] K. Sims. Evolving 3D Morphology and Behavior by Competition. In *Artificial Life IV Proceedings*, volume I, pages 28–39, 1994.

[13] D. Thalmann. The Virtual Human as a Multimodal Interface. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 14–20. ACM Press, 2000.

[14] X. Tu and D. Terzopoulos. Artificial Fishes: Physics, Locomotion, Perception, Behavior. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 43–50. ACM Press, 1994.

Figure 5. An ecosystem after a visit of a herd of cows

Figure 6. Ecosystem that is grown without influence of any agent (top) cultivated by lawnmakers (middle) and by farmers (down)