A New Architecture for Simulating the Behavior of Virtual Agents

F. Luengo[1,2] and A. Iglesias[2⋆]

[1] Department of Computer Science, University of Zulia, Post Office Box #527, Maracaibo, Venezuela
fluengo@cantv.net

[2] Department of Applied Mathematics and Computational Sciences, University of Cantabria, Avda. de los Castros, s/n, E-39005, Santander, Spain
iglesias@unican.es
http://personales.unican.es/iglesias

Abstract. This paper focuses on modeling the behavior of virtual agents living in a virtual 3D world. Our aim is to apply the most typical human behavior features to our virtual agents so that they behave as realistic as possible. To this end, a new architecture for the behavioral engine that incorporates a number of these typical characteristics of human behavior is introduced. This new proposal allows the virtual agents to interact among them and with the environment in a quite realistic way. The main features of this new architecture, such as perception, knowledge management, motion control and action selection (using internal states, world information, goals, and others) are carefully analyzed in the paper. Finally, some relevant functions (those describing sensations such as tiredness, agent's resistance and recovery capacities, happiness and anxiety) and parameters (those determining the vision range or sociability) are also described in the paper.

1 Introduction

One of the most exciting current applications of computer graphics and virtual reality is the simulation and animation of virtual worlds. They are commonly used in the entertainment industry, ranging from virtual and augmented reality in movies to video games. Although it is unanimously accepted that the quality of the current computer generated scenes is very high, there is still a long way ahead to handle the behavior of the virtual agents specially for real-time applications. A major restriction in these applications is that virtual agents must satisfy a twofold objective: on one hand, they must react instantaneously to user's actions. On the other hand, they must interact with other virtual agents and the surrounding environment in an intelligent way. For the first requirement, virtual agents should be controled by users. For the second one, virtual agents must have the capability to take decisions by themselves.

⋆ Corresponding author

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2657, pp. 935–944, 2003.

Several architectures have been recently proposed to fulfill a compromise between both requirements (see Section 2 for details). Most of them consist of a large set of deterministic rules and an inference engine able to make deductions from the input sentences (either provided by the user or acquire from the information acquisition subsystem). From this point of view, they can be classified as rule-based expert systems. The complexity of these systems is mainly determined by the number of rules and the design of the inference engine. Sophisticated systems also include several subsystems for different tasks such as learning, information acquisition, coherence control, action executation, etc.

In general, these systems suffer serious limitations: rule-based systems do not deal with uncertainties because objects and rules are treated deterministically. In other words, similar conditions and knowledge always yield the same output. This is a totally deterministic scheme in which what virtual agents can do is to follow user's instructions only. If we do not know in advance what is going on in the future is only because we are not able to store and manipulate the large number of rules on-the-fly. However, in real life uncertainty is the rule rather than the exception. It is well-known that human behavior often depends on a number of internal factors associated with each human and different from those of anyone else. In addition, external factors can also model the human behavior. For instance, human beings placed in the same environment and subjected to the same conditions can react in a very different way, depending on a number of different factors: internal states, physical condition, environmental conditions, etc. Therefore, this set of internal and external factors should be taken into account in order to create a more realistic animation of virtual humans. This is the aim of the present work.

The structure of this paper is the following: in Section 2 we present some previous work in this field while Section 3 describes our new architecture for the behavioral engine. This section also includes a brief description of each subsystem of this behavioral engine. Some implementation technical details are given in Section 4. Finally, Section 5 closes the paper with the main conclusions and some further remarks.

2 Previous Work

Several researchers have worked in behavioral animation of virtual agents during the last few years. At the beginning emphasis was put on the animation and control of human motion [13,22]. Most of the work at that time was based on the so popular nowadays motion capture systems. The reader is referred to [15] for a recent survey on this topic. More details can be found in, among others, [1,2] and references therein. Subsequently, more attention was devoted to the integration of motion and behavior of the virtual agents [4,5,7,12,16,17,18,21] with applications to real-time virtual environments [3,8,10,11]. Most recent developments in the field, as those of Prof. Thalmann and collaborators, include the possibility to give the virtual actors some kind of autonomy without losing control [6,14,19]. In addition, more sophisticated systems incorporating complex features such as

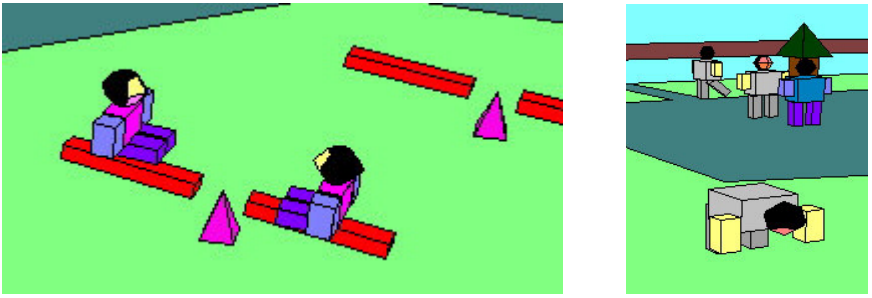

Fig. 1. Virtual agents evolving in a playground: (left) two children playing; (right) some adults chating, walking and doing physical exercices.

perception, knowledge, reasoning and learning have been recently developed by, among others, Prof. Terzopoulos and his group at the University of Toronto[9, 20]. As shown in the next paragraphs, many of these features have also been incorporated into the system presented in this paper.

3 Behavioral Engine Architecture

Since our primary goal is the accurate animation of complex behavioral features of virtual agents we will focus on describing the behavioral engine only. To this aim, let us consider a typical scene of a virtual 3D world, namely, a playground such as that shown in all figures throughout the paper. In an open environment like this, our virtual agents can find many different things to do: children can play, as those shown in Figure 1(left), a couple of adults can talk to each other, some people can make physical exercices while others are just talking a walk around, as shown in Fig. 1(right). The most important point here is that all these actions are performed by the virtual agents themselves without any user intervention, furthermore, with no external control at all.

The key to achieving this level of complexity is to create virtual agents which are totally autonomous in the sense that they incorporate all they need to evolve in an independent but still very realistic way. In particular, our behavioral engine architecture consists of a set of subsystems to perform specific tasks such as:

- to capture information from the sorrounding environment. This task will be performed by the *perception subsystem* (see Section 3.1),
- to analyze the world information acquired by the perception subsystem and then to update the knowledge base accordingly. This task is performed by the *analyzer subsystem* described in Section 3.2,
- to store the new world information captured by the sensors and then filtered by the analyzer subsystem into the *knowledge base subsystem* (see Section 3.3),

– to update some internal states such as tiredness, happiness and others, that are managed by the *internal states subsystem* (see Section 3.4),
– to determine the next goals and sort them by some prescribed criteria. This task is achieved by a combination of the *goal engine subsystem* and the *goal controler subsystem* described in Section 3.5,
– to move on the scene. To this end, a *motion subsystem* has been created that allows the agent to walk, turn, stay upright, sit down, do physical exercises and avoid obstacles, among others (see Section 3.6).

A scheme of our behavioral engine architecture is shown in Figure 2. All these subsystems are briefly described in the next paragraphs.

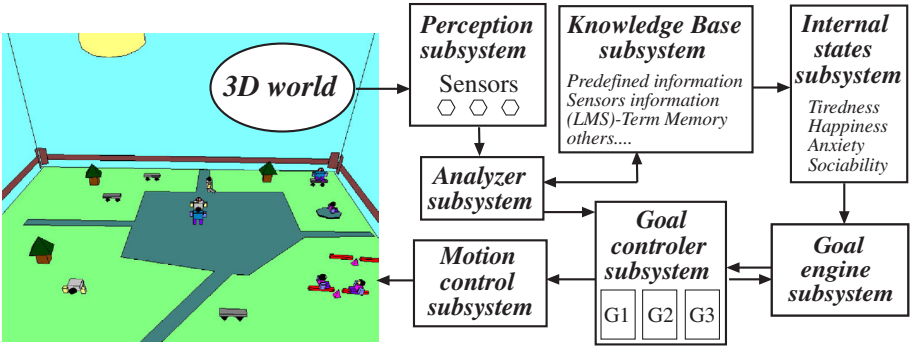

Fig. 2. Behavioral engine architecture scheme

3.1 The Perception Subsystem

This system includes a set of individual sensors so that each virtual agent is able to analyze the environment, capturing the most relevant information and sending it to the analyzer. At the time of this paper, the vision is the unique human sense actually incorporated into our virtual agents. By this we mean that a virtual agent placed in any arbitrary environment is able to take a look at the neighborhood and recognize the different things within. This recognition includes the determination of distances and positions of the different objects of the scene, so that the agent can move in this environment, avoid the obstacles, identify other virtual agents and take decisions, as it will be shown later. To fulfill our aim of simulating reality as accurate as possible, each virtual agent can effectively see the objects placed just in front and beside, but not the objects placed behind the agent (although the agent can always turn on the left/right and see the objects, as any human being can also do). Further, each agent has a predefined vision range (given by a numerical threshold value of the distance), and hence, objects far away from the agent can be considered as visible for that

agent only if the distance from the agent to the object is less than such threshold value. Otherwise, the object becomes invisible for the agent even although it could eventually be visible for other agents at the same scene and distance. This vision threshold also depends on the agent, and its corresponding value is determined by the user during the initialization step. Note that eagle eye, near-sighted and even blind individuals can be easily simulated in our scheme.

3.2 The Analyzer Subsystem

As remarked above, this subsystem receives the world information acquired by the perception subsystem and then analyzes it to update the knowledge base accordingly. In its turn, a new entry in the knowledge base might modify the previous analysis and, consequently, the agent behavior as well. On the other hand, the analyzer can modify the goal controler subsystem when, for example, new information acquired by the sensors or coming from the knowledge base is attained. This is a reasonable assumption, as human behavior changes to adjust to new circumstances and/or knowledge.

3.3 The Knowledge Base Subsystem

This subsystem is basically a database including all the relevant information. This information comes from many different sources. On one hand, some information is provided at the initialization step. This initial information tells the agent about him/herself (his/her sex, name, age, etc.) or others (friends, colleagues). This information is static, meaning that it cannot be modified during the simulation process.

Once the simulation is launched, new information is generated at each step of the running process. This dynamic information is also stored in a different field of the knowledge base. Finally, each agent has short, medium and long-term memory. For example, let us suppose that our virtual agent is talking a walk and sees a bank in the park. If after a brief span the agent is getting tired, he/she will look for a bank to sit down. Short-term memory allows him/her to recall not only that there is a bank in the park but also its relative position and distance with respect him/herself. Hence, the agent is able to go directly towards the bank and find it very quickly. On the contrary, if the agent decides to walk longer, short-term memory information is removed and only a part of this information will be transferred to the medium-term memory. Thus, the virtual agent recalls he/she has seen a bank nearby, but the information about distance and position is not available anymore. As a consequence, the agent is forced to explore around him/her in a random-like way. As the span increases, even this information is lost and after some days, the agent can only use the long-term memory to recall that he/she went to a park with a bank somewhere. Finally, even this information is removed from memory after a long time.

3.4 The Internal States Subsystem

This subsystem manages some internal states of the virtual agent. Currently, only four internal states are implemented in the system: tiredness T, happiness H, anxiety A and sociability S, which are functions of different variables. The tiredness T starts from an initial value $T_{ini} \in [0, 100]$ and then changes as the time t goes to infinity. Basically, this function T increases when the virtual agent is moving or doing some kind of physical effort, whereas it decreases when the agent is resting (for example, when the agent sits down or stays exactly at the same place for a while). The function T can be expressed as:

$$T(R, r, t) = \sum_{n=1}^{\infty} [g_n(R, t) + h_n(r, t)] \tag{1}$$

where $g_n(R, t)$ is a function of the resistance capacity R of the virtual agent and the time t, and $h_n(r, t)$ depends on the recovery capacity r of the agent and also on the time t and n is used to indicate the number of local maxima of function T. In other words, the tiredness is seen as a combination of two different functions g_n and h_n which account for the tiredness and the recovery spans, respectively. Each time the agent moves, the function g_n varies as:

$$g_n(R, t) = Exp\left\{ \frac{92}{200} \left[(t + 200) - \frac{2}{n} R \right] - 4.6 \right\} - 1 \tag{2}$$

which is a strictly increasing function while h_n is set to 0. This means that when a virtual agent moves, he/she is always getting tired (clearly, a very reasonable assumption). The agent can move until this $T = 100$, when the agent stops the physical activity (g_n set to 0) and the tiredness T starts to decrease through the recovery function h_n.

Usually, activation of either g_n or h_n (note that they are never activated simultaneously) implies a change in agent's goals. For example, when the tiredness is too low (let us say, $T_{ini} = 10$) the agent is plenty of energy and hence, anxious to walk, run or do some kind of physical effort in order to decrease its energy level (or equivalently, to increase his/her tiredness T). After a span, the tiredness reaches an upper threshold value $T_{sup} < 100$, and the agent immediately starts to look a place to rest (perhaps, a bank to sit down). Note that this value T_{sup} is strictly less than 100 in order to give the agent the chance to move and look for such a place. Otherwise, tiredness would be equal to 100 and the agent has not energy at all, thus forcing him/her to stay upright (to sit down on the court is absolutely forbidden in our urban park).

Another internal state is happiness. It increases when the agent is enjoying the activity he/she is doing at that time. For example, when children are playing with the seesaw (See Fig. 1(left)) its happiness function increases a lot. After reaching an upper threshold value H_{sup}, children are getting bored because they are doing the same activity for a long time, and suddenly happiness H starts to decrease. Of course, there are many different ways to become happy and they

depend on the particular agent we are dealing with. Some people can enjoy talking to their friends, while others will prefer to walk alone, play with something, read the newspaper or something else.

The third internal state, anxiety, is a measure of the frustration caused by trying to make something and failing in this attempt. For instance, the seesaw requires two people to play with. If a child is alone but he/she wants to play with the seesaw, he/she sits down on it and waits for another child to play together. If nobody is going to play with the child, he/she becomes anxious, that is, his/her anxiety function A increases abruptly.

Finally, the sociability function S measures agent's wishes to socialize. Due to the difficulty to associate this factor with an analytic expression, S is assigned a constant value, which is set during the initialization step.

3.5 The Goal Engine and Goal Controller Subsystems

The goal engine subsystem is the component that updates, sorts and finally stores agent's goals into a priority list. Currently, our agents are able to do a very limited number of things, such as: (1) do nothing (default agent's goal); (2) walk around; (3) sit on a bank; (4) do physical exercices; (5) play alone with a wheel; (6) talk to others, and (7) play with the seesaw with others. All these goals are illustrated in Figure 3.

Firstly, priority criteria are determined by agent's internal states. For example, if an agent is not very tired, $T \leq 70$, he/she could choose among goals 2, 4, 5, 6 and 7. Depending on the sociability level, goals 6 and 7 might be rejected/accepted (low/high values of S). Also, the happiness will determine the feasible goals, as the agent will prefer the goal which gives him/her the highest level of happiness. Finally, the anxiety for pursuing a goal and to fail is other factor that could modify the elements of the list of goals as well as its order in the priority list.

Once the lists of goals and priorities are defined, they are sent to the goal controler subsystem. This component will determine which goals the agent is actually going to get. Firstly, this subsystem check for those goals that cannot be carried out and consequently must be rejected. For example, if the agent is tired, he/she will look for a bank to sit down. If there is no seats available or they are not free at this time, the agent must modify his/her goals. This leads to the concept of feasibility, that is used to check which goals are actually feasible at any time. From this point of view, the goal controler subsystem acts like a filter modifying the goals and priorities received from the goal engine subsystem.

3.6 The Motion Subsystem

Once the goals and priorities are defined by the goal controler subsystem, this information is sent to the motion subsystem. This component is responsible for all the motion routines, including avoiding static obstacles, avoiding dynamic obstacles, walking, sitting-down, playing with the seesaw, playing with the wheel

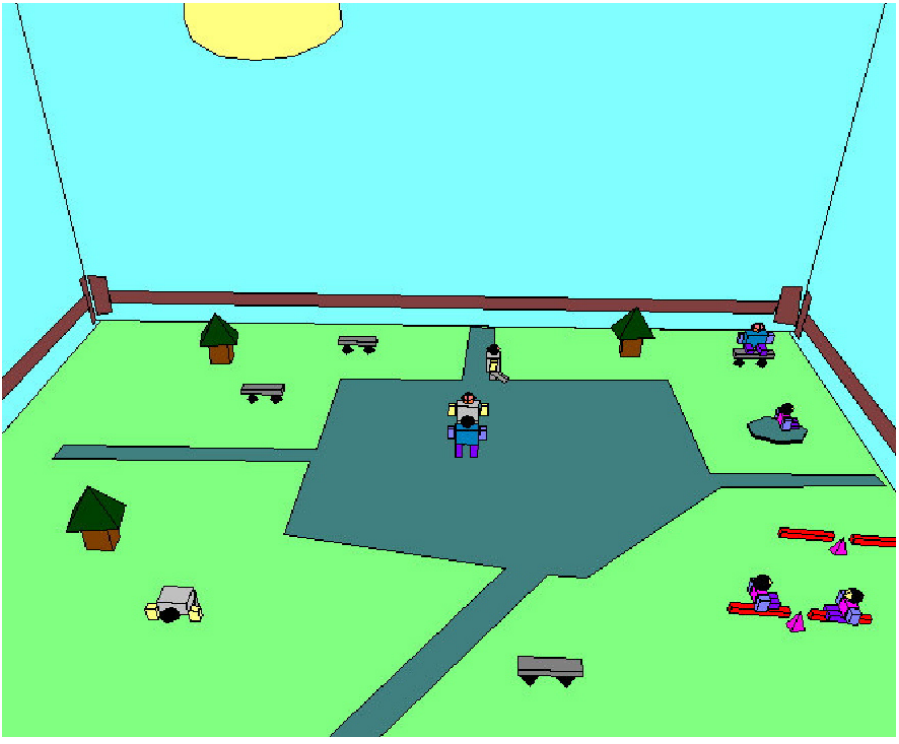

Fig. 3. This figure illustrates different goals the virtual agents can do: they can walk around, play with either the wheel or the seesaw, talk to others, sit on the bank and do physical exercices

and doing physical exercices. Some much subtler motion routines, such as competing routines and others, are also incorporated in our system. They are not described here because of limitations of space. However, agents motion is still very simple and further work is currently being developed to create more and better motion routines.

4 Implementation of the System

The behavioral engine described in the previous paragraphs as well as all graphical output have been implemented in C++. Since our focus is the simulation of virtual agents' behavior, we do not worry too much about the quality of the graphical environment. Thus, instead of using the OpenGL graphics library or similar we decided to create everything. The obvious advantage of this "autonomous" solution is that it gives us more degrees of freedom in making decisions and allows us for a better control. However, it requires a very important

programming effort because all non-specific tools need to be developed. For example, problems such as collision detection among objects, algorithms for hidden surfaces and the design of agents, objects and environments (to quote only three examples) were to be solved. The counterpart is that the behavioral engine is totally integrated into the graphical module. As a consequence, any new improvement in the behavioral engine is instantaneously reflected in the graphical output, avoiding the use of sockets, TCP/IP or any other communication and/or file transfer protocols, which could eventually lead to substantial delays during the real-time simulation process.

5 Conclusions and Further Remarks

The core of this paper is the accurate simulation of human behavior by virtual agents living in a virtual 3D world. To this end, the paper introduces a new architecture for the behavioral engine that incorporates a number of the most typical human behavior actions. This new proposal allows the virtual agents to interact among them and with the environment in a quite realistic way. Some remarkable features of this new architecture, such as perception, knowledge management, motion control and action selection (using internal states, world information, goals, and others) are carefully analyzed in the paper. Finally, some relevant functions (those describing sensations such as tiredness, agent's resistance and recovery capacities, happiness and anxiety) and parameters (those determining the vision range or sociobility) are also described in the paper.

Although the system presented here could be a remarkable first step, there is still a long way to walk. On one hand, we expect that future versions will include a better graphical output, including textures, shadows, reflections and illumination models. At this time, we are using OpenGL to incorporate all these graphical improvements. On the other hand, the accurate simulation of human behavior requires to modify the system substantially. The list of new functions and parameters to be defined is virtually infinite, and it is still unclear nowadays which are the functions modeling most of the human actions and decisions. These and other tasks, such as the addition of new sensations, feelings, beliefs and capabilities (such as speech) to the virtual agents, the improvement of agents' motion, knowledge base and deduction engine and a more accurate modeling of many human behavior functions are the following steps to be done in this work. The obtained results will be reported elsewhere.

References

1. Badler, N.I., Barsky, B., Zeltzer, D. (eds.): Making Them Move. Morgan Kaufmann, San Mateo, CA (1991)
2. Badler, N.I., Phillips, C.B., Webber, B.L.: Simulating Humans: Computer Graphics Animation and Control. Oxford University Press, Oxford (1993)
3. Blumberg, B.M., Galyean, T.A.: Multi-level direction of autonomous creatures for real-time virtual environments. Proc. of SIGGRAPH'95, ACM, New York (1995) 47–54

4. Boulic, R., Becheiraz, P., Emering, L., Thalmann, D.: Integration of motion control techniques for virtual human and avatar real-time animation. Proc. of ACM Symposium on Virtual Reality Software and Technology, ACM, New York (1997) 111–118
5. Brogan, D.C., Metoyer, R.A., Hodgins, J.K.: Dynamically simulated characters in virtual environments. IEEE Computer Graphics and Applications (1998) 58–69
6. Caicedo, A., Thalmann, D.: Virtual humanoids: let them to be autonomous without losing control, Proceedings of the Fourth International Conference on Computer Graphics and Artificial Intelligence, D. Plemenos (ed.) University of Limoges, Limoges (2000) 59-70
7. Cerezo, E., Pina, A., Seron, F.J.: Motion and behavioral modeling: state of art and new trends. The Visual Computer, **15** (1999) 124–146
8. Farenc, N., Boulic, R., Thalmann, D.: An informed environment dedicated to the simulation of virtual humans in urban context, Proceedings of EUROGRAPHICS'99 (1999) 309–318
9. Funge, J., Tu, X. Terzopoulos, D.: Cognitive modeling: knowledge, reasoning and planning for intelligent characters, Proceedings of SIGGRAPH'99, ACM, New York (1999) 29-38
10. Granieri, J.P., Becket, W., Reich, B.D., Crabtree, J., Badler, N.I.: Behavioral control for real-time simulated human agents, Symposium on Interactive 3D Graphics, ACM, New York (1995) 173–180
11. Kallmann, M.E., Thalmann, D.: A behavioral interface to simulate agent-object interactions in real-time, Proceedings of Computer Animation'99, IEEE Computer Society Press, Menlo Park (1999) 138–146
12. Maes, P., Darrell, T., Blumberg, B. Pentland, A.: The alive system: full-body interaction with autonomous agents, Proceedings of Computer Animation'95, IEEE Computer Society Press, Menlo Park (1995) 11–18
13. McKenna, M., Pieper, S., Zeltzer, D.: Control of a virtual actor: the roach, Proceedings of SIGGRAPH'90, Computer Graphics, **24**(2) (1990) 165–174
14. Monzani, J.S., Caicedo, A., Thalmann, D.: Integrating behavioral animation techniques, Proceedings of EUROGRAPHICS'2001, Computer Graphics Forum, **20**(3) (2001) 309–318
15. Multon, F., France, L., Cani-Gascuel, M.P., Debunne, G.: Computer animation of human walking: a survey, Journal of Visualization and Computer Animation, **10** (1999) 39–54
16. Perlin, K., Goldberg, A.: Improv: a system for scripting interactive actors in virtual worlds, Proceedings of SIGGRAPH'96, ACM, New York (1996) 205–216
17. Renault, O., Magnenat-Thalmann, N., Thalmann, D.: A vision-based approach to behavioral animation, Journal of Visualization and Computer Animation, **1** (1990) 73–80
18. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model, Computer Graphics, **21**(4) (1987) 25–34
19. Thalmann, D., Noser, H.: Towards autonomous, perceptive and intelligent virtual actors, Lecture Notes in Artificial Intelligence, **1600** (1999) 457–472
20. Tu, X. Terzopoulos, D.: Artificial fishes: physics, locomotion, perception, behavior, Proceedings of SIGGRAPH'94, ACM, New York (1994) 309–318
21. Wilhelms, J., Skinner, R.: A "notion" for interactive behavioral animation control, IEEE Computer Graphics and Applications, **10**(3) (1990) 14–22
22. Zeltzer, D.: Motor control techniques for figure animation, IEEE Computer Graphics and Applications, **2**(9) (1982) 53–59