Employing virtual humans for education and training in X3D/VRML worlds

Lucio Ieronutti*, Luca Chittaro

HCI Lab

Dept. of Math and Computer Science,University of Udine,

via delle Scienze 206, 33100 Udine, Italy

Abstract

Web-based education and training provides a new paradigm for imparting knowledge; students can access the learning material anytime by operating remotely from any location. Web3D open standards, such as X3D and VRML, support Web-based delivery of educational Virtual Environments (EVEs). EVEs have a great potential for learning and training purposes, by allowing one to circumvent physical, safety, and cost constraints. Unfortunately, EVEs often leave to the user the onus of taking the initiative both in exploring the environment and interacting with its parts. A possible solution to this problem is the exploitation of virtual humans acting as informal coaches or more formal instructors. For example, virtual humans can be employed to show and explain maintenance procedures, allowing learners to receive more practical explanations which are easier to understand. However, virtual humans are rarely used in Web3D EVEs, since the programming effort to develop and re-use them in different environments can be considerable. In this paper, we present a general architecture that allows content creators to easily integrate virtual humans into Web3D EVEs. To test the generality of our solution, we present two practical examples showing how the proposed architecture has been used in different educational contexts.

Keywords: interactive learning environments, virtual reality, distance education and telelearning, human-computer interface.

* Corresponding author.

E-mail addresses: ieronutt@dimi.uniud.it (L. Ieronutti), chittaro@dimi.uniud.it (L. Chittaro)

1. Introduction

Web-based teaching provides a new paradigm for imparting knowledge, enabling both on-campus and distance-learning models (Sanz & Iskander, 2000; Zahorian et al., 2000); students can access the learning material anytime by operating remotely from any location. As a result, learners can progress on their own initiative to study the content of the course (Kayssi, El-Haji, Assir, & Sayyid, 1999). In particular, the Web can give the learner the means to access information flexibly and individually: she can choose what to view, how long to view it and how many times to view it (Lin & Hsieh, 2001). Moreover, the possibility of exploiting personalization techniques allows Web-based learning applications to provide tailored teaching methods taking into account individual students knowledge, interests, preferences and needs (Oreta, 1999).

Besides traditional HTML materials (e.g., text, images and hyperlinks), more complex multimedia elements can be integrated into the learning experience. In particular, Web3D open standards, such as X3D (2004) and VRML (1997), support the delivery of Educational Virtual Environments (EVEs) through the Web. EVEs have a great potential for learning and training purposes since they allow one to circumvent physical, safety, and cost constraints. As a result, Web3D standards allow content creators both to exploit EVEs benefits and to provide individualized instruction for a massive number of learners.

Unfortunately, unlike traditional scenarios, EVEs often leave to the user the onus of taking the initiative both in exploring the environment and interacting with its parts. A possible solution to this problem is the exploitation of virtual humans acting as informal coaches or more formal instructors. However, virtual humans are rarely used in Web3D EVEs, since the programming effort to develop and re-use them in different environments can be considerable. For example, at lower levels, one has to consider aspects concerning the geometric definition of the virtual human model and appearance while, at higher levels, one should be able to define how the virtual human behaves in the EVE. As a result, to include a virtual human in an EVE, the content creator has to write a considerable amount of code. We have thus developed a general architecture that allows one to easily integrate virtual humans into EVEs. In particular, our solution can be run on the Web since it i) is completely based on Web standards (such as X3D/VRML and Java), and ii) is compact in size and then it can be quickly downloaded.

The paper is structured as follows. In Section 2, we illustrate the key features of EVEs. In Section 3, we present the main benefits of using virtual humans in EVEs. In Section 4, we illustrate the different aspects that have to be considered for modeling virtual humans. Section 5 describes the software architecture we developed and Section 6 presents two practical examples of its application to different educational contexts. Finally, in Section 7, we discuss the current limitations and future developments of our work.

2. Educational Virtual Environments (EVEs)

Virtual Reality (VR) allows to immerse the user in at three-dimensional, interactive, computer-generated world that is able to detect user's actions and change in accordance. The user can interact with the virtual environment (VE) either through traditional input devices (e.g., keyboard and mouse) or by employing special devices (e.g., head-mounted display and data glove).

Educational VEs (EVEs) provide students with visual, experiential, and self-directed learning. Students can i) experience directly some physical proprieties (e.g., shape, size and time duration) of objects and events, ii) change point of view to access new and/or unusual perspectives (Ferrington & Loge, 1992), iii) interact with objects either to discover and study their hidden elements (Sonnet et al., 2004) or to evaluate the effects of manipulations (Larijani, 1994). Moreover, EVEs for training can (Vince, 1995; Kozak et al., 1993): i) provide a low-cost alternative to creating full-scale physical training scenarios, ii) offer the opportunity to create a wide variety of scenarios including those rarely (or never previously) encountered in the real world, iii) simulate training scenarios that can be run repeatedly and v) include a monitoring of progress during training sessions to evaluate learners skills. A thorough discussion of the pedagogical basis and motivations for EVEs is provided in (Chittaro & Ranon, 2005).

Advances in computer graphics and hardware performance in the last 10 years have made EVEs economically more feasible and several new EVEs have appeared in the literature. For example, Bell and Fogler (1996) have developed Vicher, an EVE for chemical engineering education that deals with the topics of catalyst deactivation and non-isothermal effects in chemical reaction engineering. In this context, the EVE provides students with unlimited access to virtual chemical manufacturing facilities, without risks and without disrupting real operations.

VRPS (Virtual Reality Physics Simulation) (Kim et al., 2001) allows one to learn physics concepts, such as wave propagation and relative velocity. The application interface brings together 3D models of real apparatus and a visualization of physical situations in an interactive manner. Students can investigate the functioning of AC/DC electric generators; they can watch the output voltage in a voltmeter while switching the direction of the magnetic field, changing the intensity of the magnetic field, and adjusting the frequency of revolutions.

Virtual Water (Trindade et al., 1999) exploits the same EVEs advantages; the application introduces students to the molecular bonding and structure of water by employing 3D graphics to represent different scientific concepts (e.g., molecular geometry, orbitals and densities).

An interesting Web3D EVE (based on the use of VRML and Java) has been developed by Ong and Mannan (2004); the proposed Web-based tool allows students to experience with CNC (i.e. Computer Numerical Control) control machine tools, which deal primarily with the NC control of the motion of cutting tools in manufacturing. This application allows learners to enhance the understanding of the topic without requiring to machine a new workpiece every time a student needs to test new NC code.

3. Virtual humans in EVEs

One typical problem of EVEs is the lack of structuring facilities (Kaur et al., 1999); the onus is often on the user to take the initiative both in exploring the environment and interacting with its parts. The lack of proper assistance clashes with the traditional learning scenario, where a real teacher structures the presentation of material and learning activities (Hertz-Lazarowitz & Shachar, 1990). Embodied agents provide a possible solution to the problem of unstructuredness (Economou et al., 2000): the agent is embodied (i.e. it has a visual representation) and it acts as a teacher. While it is not feasible to provide a human tutor for every learner in the real world, embodied agents can offer assistance to anyone with access to a computer, enabling an individualized instruction for a massive number of learners.

In the context of EVEs, embodied agents take often the form of virtual humans, i.e. 3D simulations of human beings. Virtual humans can employ locomotion abilities and perform actions to navigate the VE and interact with objects. These capabilities enable virtual humans to explain physical and procedural human tasks (e.g., maintenance procedures), allowing learners to receive more practical explanations which are easier to understand. Moreover, virtual humans can communicate with students in a natural way by exploiting both verbal and nonverbal communication. In particular, they can perform deictic gestures to focus the students' attention towards a particular aspect, while facial expressions can be used to make the dialogue with learners more engaging and natural. For example, virtual human gaze can regulate turn-taking in a mixed-initiative dialogue (Cassell et al., 1994), while head nods and facial expressions can provide unobtrusive feedback on students' actions. Moreover, as proved by Lester et al. (1997), the presence of an anthropomorphic agent may increase the students' arousal and motivation.

Due to their peculiar features, virtual humans are being increasingly used in different application domains. Simulating human motion and behavior using a computer is an active research area (e.g., Ponder et al., 2003; Traum & Rickel, 2002) and different approaches have been proposed in literature. Each approach considers the appearance, function and autonomy of the virtual human with respect to specific levels of detail and accuracy, and given application areas.

In the context of EVEs, virtual humans are used for different purposes, such as explaining physical and procedural human tasks (Rickel & Johnson, 1999), simulating dangerous situations (Traum & Rickel, 2002), assisting users during the navigation both by showing where relevant objects/places are and providing users with additional information (Chittaro et al., 2003). Virtual humans can also be the main subject of study e.g., for training civilian officers to recognize and interact with mentally ill people (Hubal et al., 2003) and in applications for medical staff training (Kizakevich et al., 1998; Cavazza & Simo, 2003; Ponder et al., 2003). In the following, we present an overview of representative systems, architectures and applications employing virtual humans in EVEs.

JUST-TALK (Hubal et al, 2003) is a system designed to teach policemen basic techniques for managing encounters with mentally ill people through a series of one-to-one scenarios with a simulated subject. Through observations of the VE and dialog with the virtual subject, the learner has to stabilize the situation and decide whether to release or detain the subject.

Virtual Medical Trainer (VMET) (Kizakevich et al., 1998), combines medical procedures databases and virtual patients to produce an interactive environment that can mimic the cognitive pre-hospital assessment and care demands of a real emergency; moreover, VMET provides the opportunity to experience a range of trauma scenarios.

An interesting EVE for training medical staff has been presented by Cavazza and Simo (2003): the system simulates the cardio-vascular system to generate clinical syndromes and derives the consequences of the trainee's therapeutic interventions, by employing the virtual patient's appearance as the most direct manifestation of the gravity of the situation. The VE (representing an operating room) is accurately designed to reproduce the atmosphere and tension created by the critical nature of the situation. The system employs physiological models for representing the internal model of the patient.

STEVE (Soar Training Expert for Virtual Environments) (Rickel & Johnson, 1999) is a virtual human that helps students learn to perform physical and procedural tasks. In particular, STEVE demonstrates how to perform tasks and can also monitor students while they practice tasks, providing assistance when needed. Besides teaching students individual tasks, the system can also help them learn to perform multi-person team tasks: STEVE can either behave as a tutor for a student learning a particular role in the team, or play the role of a teammate when a human teammate is unavailable. STEVE has been tested on a variety of naval operating procedures, where the virtual human operated several consoles controlling the engines aboard ships and inspected air compressors on the engines. Unfortunately, STEVE is not suitable for a common personal computer because it has to run different modules of the architecture in parallel as separate processes, possibly on different computers.

VHD++ (Ponder et al., 2003) is a real-time software framework for developing virtual and augmented reality applications employing advanced virtual character simulation. VHD++ is an efficient, flexible framework based on modern 3D game-engine design principles. VHD++ can be employed in a large number of different scenarios; it has been used in i) cultural heritage environments involving architecture, acoustics and cloths, ii) augmented reality applications for maintenance training, iii) edutainment (i.e. entertainment that is intended to be educational) systems based on storytelling and iv) immersive training applications.

Unlike the above mentioned systems, the framework proposed by Sims (2005) is a Web-compliant system. In particular, the author used it to develop an interactive training simulation aimed at teaching elements of the Iraqi dialect, nonverbal language, culture and customs. In this application, H-Anim virtual humans are used both as role-playing actors, and as instructors/mentors who present important elements of the dialect and culture, explain best courses of actions and provide feedback on the learner's choices.

4. Modeling different virtual human aspects

Virtual human modeling and simulation requires knowledge of different disciplines, such as computational geometry, kinematics, artificial intelligence, computer graphics, and bio-mechanics. The complexity of modeling virtual humans encourages to divide the problem in several sub-problems; this can be done with the five-levels hierarchy (Fig. 1) proposed by (Funge et al., 1999).

The lowest layer of the hierarchy is the *geometric layer* that concerns the definition of the virtual human model and its appearance. The choice of the most suitable virtual human model depends both on the purpose of the application and on its intended users. For example, in an EVE designed for children, a funny, cartoon-like model could be very appreciated as virtual teacher, while in an EVE for training medical staff realistic human models have to be employed.

At the *kinematic layer*, the virtual human is represented as a set of rigid bodies, called *segments*, hierarchically organized and connected by *joints*, that represent articulations of the virtual human. From this point of view, an animation can be defined either by specifying different *joints* rotation values in time, or by defining (or automatically computing) different positions of specific parts of the virtual human body (called *end-effectors*) in time. The latter approach is based on *inverse kinematics* (i.e., systems that compute the joints rotation values needed to put end-effectors in any given position) and is commonly used to control feet and hands movements (e.g., when the virtual human has to reach and grasp an object).

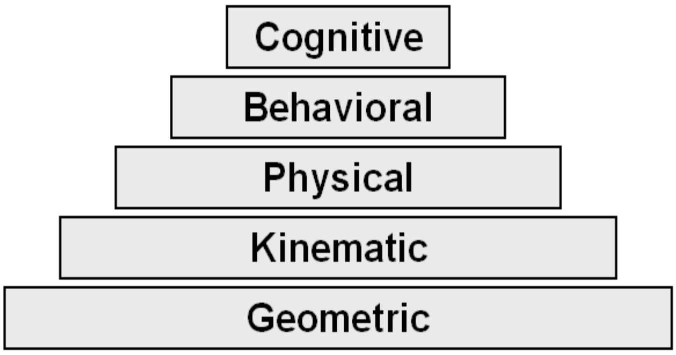


Fig. 1. The modeling hierarchy proposed by (Funge et al., 1999).

At the *physical layer*, animation is obtained by applying physical laws to different parts of the virtual human body; this approach is used to compute complex animations, such as hair movement and skin deformation. In the context of EVEs, the physical layer allows virtual humans to produce facial expressions through the computation of corresponding skin deformations (allowing the virtual human to communicate with the user in more effective, engaging and natural ways).

The *behavioral layer* represents the behavior of the virtual human (e.g., in terms of stimulus-action associations) by defining how the virtual human responds to users' actions (e.g., it can specify how the virtual human reacts when a student makes an error).

The highest layer, the *cognitive* one, binds various stimuli with complex reasoning processes that allow the virtual human to search for the most suitable reaction. Cognitive models go beyond behavioral models in that they govern what the virtual human knows, how that knowledge is acquired, and how it can be used to plan actions.

However, the development of virtual humans for learning purposes does not necessarily require to implement all five layers; the behavioral layer is needed for modeling interactive virtual humans (whose behavior is influenced by user's actions), the kinematic and physical layers allow the virtual human to perform a variety of movements by changing animation parameters, and the cognitive layer is required for simulating complex reasoning processes.

Unfortunately, H-Anim (2004), the standard for representing humanoids in X3D/VRML worlds, deals only with the geometric layer of the hierarchy by defining the hierarchical structure of the virtual human in terms of segments and joints. H-Anim does not specify neither how to define the virtual human behavior nor how the different layers can be integrated. As a result, when the Web3D content creator wants to include an interactive virtual human into a X3D/VRML EVE, she has to write a considerable amount of code that refers to the different layers of the hierarchy.

In the following, we introduce the *Virtual Human Architecture* (VHA), a software architecture that we developed to i) control H-Anim virtual humans acting as instructors/mentors by clearly separating the geometrical, kinematic, physical and behavioral layers, and ii) integrate such virtual humans into Web3D EVEs.

5. Virtual Human Architecture

The Virtual Human Architecture (VHA) is a software architecture that allows one to program H-Anim virtual humans whose behavior is based on the *Sense-Decide-Act* paradigm; the virtual human is able to perceive what happens in the surrounding environment (Sense process), to decide a proper reaction (Decide process) and to perform the related actions in the Web3D world (Act process).

Unlike other more sophisticated architectures (e.g., Rickel & Johnson, 1999; Ponder et al., 2003), our system can be employed for Web-based distance-learning applications since it i) is completely based on Web standards (such as X3D/VRML, H-Anim and Java), and ii) is compact in size.

To integrate VHA into a Web3D EVE, one has to define the following set of application-dependent data: i) how the virtual human reacts to environment events, ii) information on the virtual environment topology (e.g., navigable areas), iii) objects information (e.g., position and geometry of an object), iv) a set of virtual human animations and v) the H-Anim model of the virtual human. A technical description of VHA and its integration with the VRML language is provided in (Ieronutti & Chittaro, 2005).

The main internal modules of the architecture (Fig. 2) are the *Behavioral Engine*, the *Execution Engine* and the *Presentation module*. The Behavioral Engine concerns the Behavioral layer of the modeling hierarchy (it controls the Sense and Decide processes), the Execution Engine deals with the kinematic, physical and geometric layers (it controls the Act process), while the Presentation module is used for presenting textual information to the user.

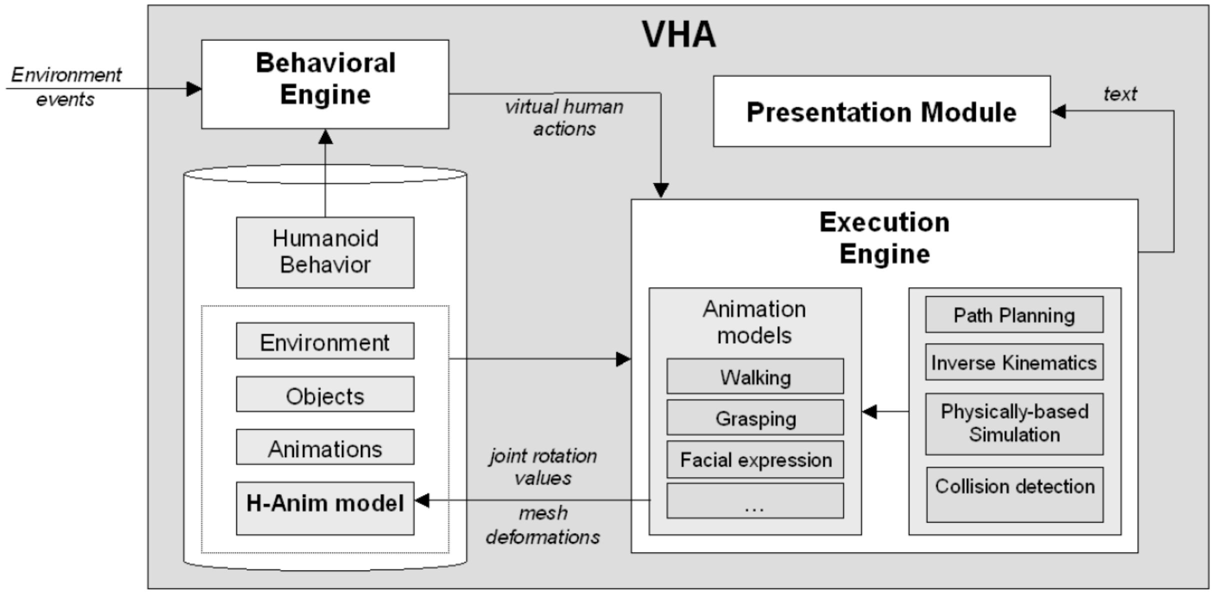


Fig. 2. The proposed Virtual Human Architecture

The Behavioral Engine senses environment events (e.g., an object has been touched by the user) and chooses the set of high-level actions (e.g., the virtual human presents the object that has been touched by the user) the virtual human has to perform in response. These actions are sent to the Execution Engine, which animates the virtual human and retrieves information that has to be provided to the user. This information is sent to the Presentation module that adapts the textual data to a format suitable for the presentation.

In the following Sections, we present each VHA module, focusing mainly on representation and control of the virtual human behavior.

5.1. The Behavioral Engine

In VHA, *Finite-State Machines* (FSMs) and *Hierarchical-State Machines* (HSMs) allow the developer to specify how the virtual human behaves, given its current state and by considering perceived environment events (that the Behavioral Engine stores into an internal buffer, called *Events Buffer*).

A FSM is represented by a directed graph G = (V, E), where V represents a set of nodes n_i, and E represents a set of oriented edges (n_i, n_j). Each node corresponds to a particular state of the virtual human, while each edge corresponds to a transition that allows the virtual human to change its current state. Conventionally, node n_0 represents the *initial state*, and nodes that do not have outgoing edges are called *end states*.

Each transition (n_i, n_j) is characterized by conditions-actions (c_{ij}, a_{ij}) pairs: c_{ij} is the set of conditions that determine the applicability of the transition, while a_{ij} is the set of actions the virtual human has to execute if the corresponding transition is activated.

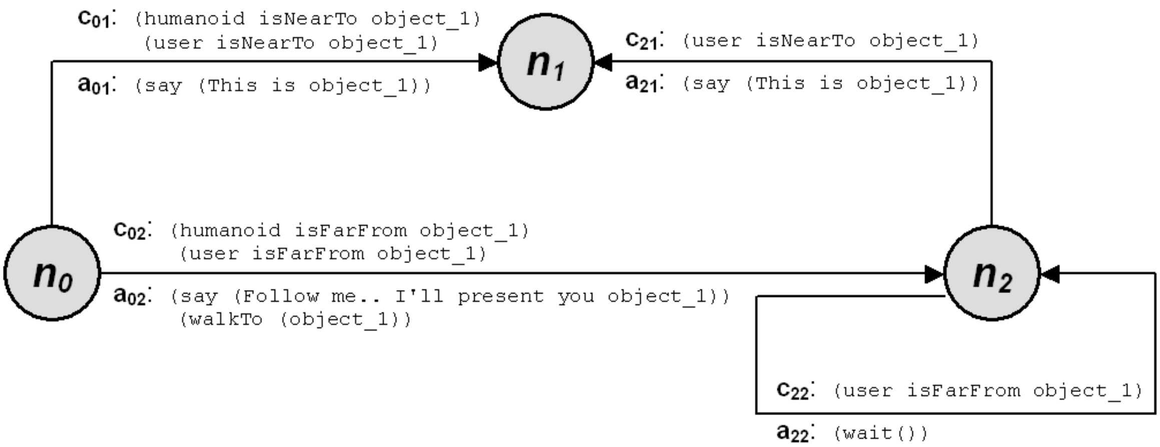


Fig. 3. An example of Finite-State Machine (FSM)

Figure 3 shows a FSM that describes a simple virtual human behavior. The FSM is represented by three states (n_1, n_2 and the initial state n_0) and four transitions ((n_0, n_1), (n_0, n_2), (n_2, n_2) and (n_2, n_1)), each one characterized by one conditions-actions (c_{ij}, a_{ij}) pair.

The correspondent virtual human behavior is the following. If both the user and the virtual human are close to an object called `object_1` (c_{01}), the virtual human presents the object (a_{01}). If they are far from the object (c_{02}), the virtual human firstly invites the user to follow it, and then it walks to the object (a_{02}). The virtual human waits (a_{22}) until the user is close enough to the object (c_{22}); when this happens, i.e. the user is near the object (c_{21}), the virtual human presents it (a_{21}).

A transition (n_i, n_j) can be activated by the Behavioral Engine if and only if the corresponding conditions c_{ij} are satisfied; to check satisfaction, conditions are matched with events stored into the Events Buffer. Conditions can concern events generated by explicit user's actions (e.g., c_{ij} = `(object_1 touched)`, the object called `object_1` has been touched by the user), they can refer to spatial relations (e.g., c_{ij} = `(user isNearTo humanoid)`, the user is close to the virtual human) or temporal conditions (e.g., c_{ij} = `(inactivityTime > 10)`, more than ten seconds have passed since the last interaction between the user and any object). The decision process is activated whenever all actions of the previously activated transition have been executed by the system; in other words, the execution of a set of actions a_{ij} is not interruptible. Actions can concern animations that the virtual human has to perform and information that has to be presented to the user.

Once a transition (n_i, n_j) is activated, the Behavioral Engine deletes all events stored into the Events Buffer, sends to the Execution Engine the list of actions a_{ij}, and then starts to sense new events.

When the virtual human acts as virtual teacher, a FSM can represent the structure of an interactive lesson; for example, in an application aimed at teaching the user how to perform a physical activity (e.g., a maintenance procedure), nodes can represent single physical activities, while edges can define the order (not necessarily sequential) in which these activities have to be carried out. Moreover, additional edges can be used to manage learner's mistakes with proper virtual human reactions (e.g., by encouraging and advising the learner). As a result, a student actively participates to virtual lessons by influencing the order in which different concepts are presented, since user's behavior (e.g., interactions of the user with objects and the virtual human) determines the way in which the graph is explored.

VHA allows one to organize large FSMs through HSMs, i.e. hierarchical compositions of FSMs. Each graph node can be either a basic state (i.e. it belongs to *V*) or a state that contains the description of a particular behavior (represented through a HSM itself). The Behavioral Engine can activate a transition if and only if the current virtual human state is an end state belonging to the underlying level (e.g., the transition (n_1, n_2) in Fig. 4 can be activated if and only if the current virtual human state is n_{11}). The hierarchy is multilevel, since each HSM can contain nodes that do not represent basic states. For a formal treatment of HSMs, see e.g. (Mikk et al., 1997).

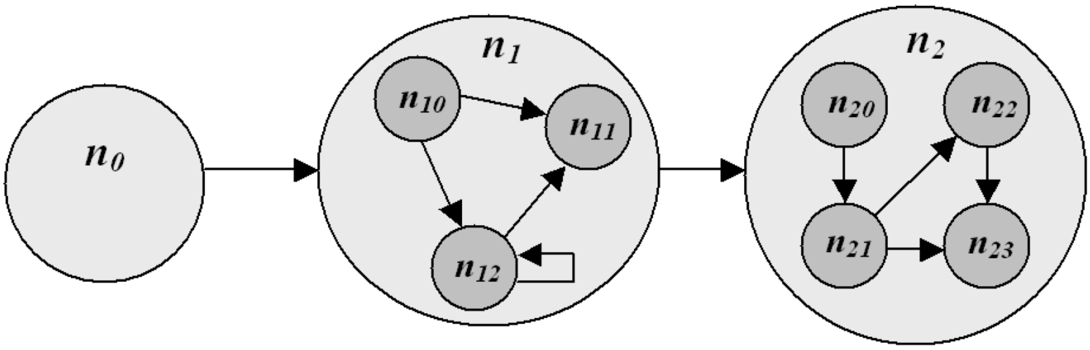


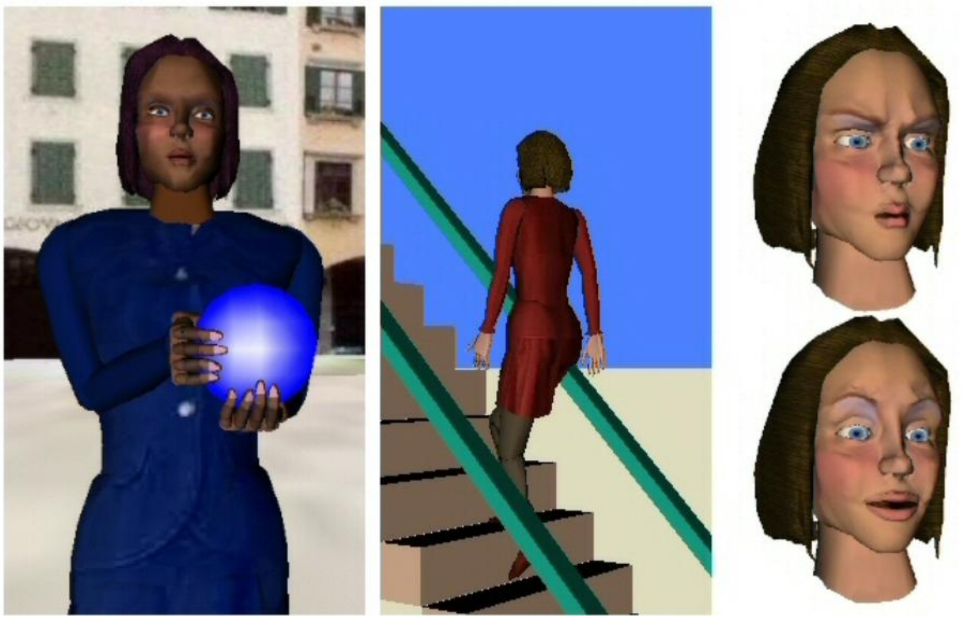
Fig. 4. An example of a Hierarchical-State Machine (HSM).

Figure 4 shows an example of a HSM; the considered HSM is composed by two levels and represents a simple virtual guide behavior (the virtual human leads the user through the environment by presenting sequentially two objects). The top level, consists of three nodes (n_0, n_1 and n_2) and two transitions ((n_0, n_1) and (n_1, n_2)), and represents the high-level description of the virtual guide behavior. In particular, node n_0 represents the initial state of the virtual guide, while n_1 and n_2 correspond to the presentation of two different objects. The second level (the lowest in the considered example) is represented by two HSMs corresponding to descriptions of more specific virtual guide behaviors. For example, node n_1 can include the FSM (depicted in Fig. 3) describing the particular virtual human behavior that corresponds to the `object_1` presentation.

The possibility of hierarchically organizing different FSMs by using a HSM allows one to define the virtual human behavior at different levels of abstraction. While the top level of a HSM represents the high-level description of the virtual human behavior (e.g., the structure of an interactive lesson), lower levels represent descriptions of more specific virtual human behaviors (e.g., the explanation of atomic concepts). As a result, one can reuse the same behavior (at any abstraction level) in different contexts, without being forced to rewrite the entire code needed for describing recurrent virtual human behaviors. Moreover, this solution allows one to organize large FSMs such that it is conceptually more clear for the developer.

5.2. The Execution Engine

To define an animation for a H-Anim virtual human, one has to specify different joints rotation values in time; the resulting virtual human motion is generated by smoothly interpolating specified rotation values. Similarly, to define a mesh[1] deformation one has to specify different positions of mesh vertexes in time; the resulting deformation (e.g., the skin of the virtual human face is modified to obtain a facial expression) is obtained by linearly interpolating specified position values. This kind of animations are called *pre-stored animations*, since the complete description of the movement has to be explicitly described.



a) b) c)

Fig. 5. Example of parametrized animations supported by VHA: a) grasping, b) walking on uneven surfaces and c) facial expressions.

Another kind of animations, called *parametrized animations*, is more convenient to animate virtual humans. These animations generate a motion as a function of use a small set of parameters. For example, parametrized animations are more general and flexible than pre-stored ones; they allow one to generate a variety of movements only by changing parameters. Parametrized animations are usually based on inverse kinematics to control end-effectors movements (e.g., virtual human feet and hands) and employ path planning algorithms to generate collision-free motions. The Execution Engine is able both to use pre-stored animations (e.g., recorded through motion capture devices, i.e. hardware devices that collect

[1] A mesh is a three-dimensional geometric object composed by triangles or quadrilaterals that share vertexes and edges.

animation data starting from a real actor motion) and to generate parametrized animations at execution-time.

Each parametrized animation is defined by an *animation model* that describes how the motion is generated starting from a set of parameters. An animation model can use information on the topology of the environment (e.g., areas that can be traveled by the virtual human), object information (e.g., object position and geometry) and parameters explicitly defined by the Web3D content creator to generate the virtual human animation.

In our system, each (pre-stored and parametrized) animation is characterized by its name; to specify an action that has to result in a pre-stored animation one has simply to use the name of the animation (e.g., a_{ij} = `wait()`), while for a parametrized animation one has also to specify the set of parameters required by the animation model (e.g., a_i = `walkTo(coordinates(5,2,4))`).

The Execution Engine contains different animation models; the most relevant ones are the grasp model (that allows the architecture to support actions such as a_{ij} = `(grasp(object_1, leftHand))`), the walk model (e.g., a_{ij} = `(walkTo(object_1))`), and the model that allows the virtual human both to produce facial expressions (e.g., a_{ij} = `(expression(smile))`) and to synchronize lip movement with speech (e.g., a_{ij} = `(say("I'm your virtual assistant"))`). These models use pre-stored data, Inverse Kinematics [Boulic & Thalmann, 1992], Path Planning [Chittaro et al., 2003], Collision Detection [Lin & Gottschalk, 1998] and Physically-based simulation [Chittaro & Corvaglia, 2003] modules to generate the motion.

5.3. The Presentation Module

The Presentation module allows the virtual human to present textual information to the user through a semi-transparent On Screen Display (OSD) (Fig. 6) and/or using a synthesized voice. The module formats the given textual information to adapt the visualization according to the display size and, at the same time, controls the speech volume by taking into account the distance between the virtual human and the user.

Fig. 6. Textual information displayed on the OSD.

6. Case Studies

The proposed architecture can be easily integrated into Web3D EVEs. In particular, we tested our solution by considering two different case studies. First, we employed a virtual human as a guide into an architectural virtual reconstruction of cultural heritage (see Fig. 7). Second, we used the same virtual human into a learning environment aimed to explain the functioning of computer devices of the 70s (see Fig. 8).

Fig. 7. The virtual human providing architectural information.

Fig. 8. The virtual human explaining the functioning of a card punch.

In particular, in the first environment, the virtual human is used to tell the history of different buildings and highlight the main architectural differences, while in the second application the same virtual human is able to provide technical information by demonstrating how different devices worked.

Since the considered Web3D environments differ in purposes and contents, to move from the first to the second application, one has to redefine the behavior of the virtual human, environment information and objects information. In these applications, VHA allowed us to focus mainly on high-level aspects, such as the definition of the virtual human behavior, rather than spend time on low-level details (e.g., the definition of many virtual human paths connecting different points of interest in the VE).

Recently we have evaluated some of the effects of using virtual humans in the context of the second application and in making the learning environment more lively and attractive for users (Chittaro et al, 2004). As a navigation aid, the virtual human proved to be appreciated in the evaluation; it was mostly rated as simple to use, and it had the advantage of being unobtrusive for the expert user. Moreover, using the virtual human demanded a very short learning time, probably because, from a human-computer interaction point of view, the virtual human metaphor has the advantage of being consistent with the real-world experience of users. On the other hand, by analyzing the most frequent concerns expressed by subjects, there is a clear need for personalization capabilities. For example, walking and talking speed of the virtual human were both rated too slow or too fast by some users: the ideal solution would be to adapt these features on the basis of each user's preferences.

In the following, we describe in more detail the features of the virtual museum application.

6.1. The 3D Computer Science museum

The 3D Computer Science museum is based on the virtual reconstruction of a typical data processing centre of the 70s, reproducing hardware from the Univac/Sperry 90/30 line.

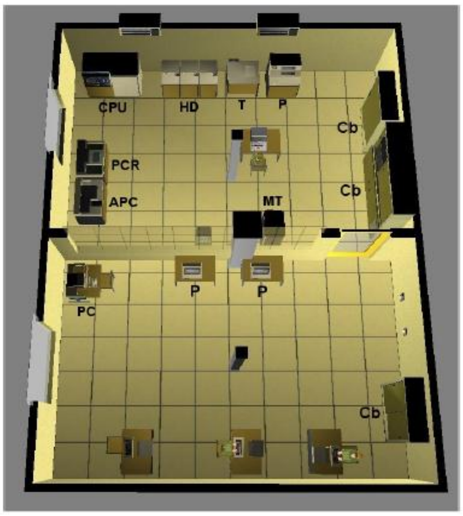


Fig. 9. Top view of 3D Computer Science museum, with the different devices on display

Fig. 10. An operator working at a video terminal.

The main pedagogical goals for this virtual museum are concerned with pointing out the large differences between data processing centers in the 70s and current computers, e.g. by illustrating the mainframe, terminals, printers and punch cards.

The virtual data processing center is divided into two main rooms (as shown in Fig. 9): a computer room, containing the main system, devoted to data processing under the control of technical staff, and a terminal room, containing punch card units and video terminals, devoted to activities that are preparatory to data processing. To increase realism, we added the typical furniture and working people (Fig. 10). Moreover, the audio channel is used to add typical noises and sounds of objects and human actions (for instance, printers, operator's typing, etc).

Museum visitors have the possibility to: i) observe the different devices in their original context of operation ii) manipulate the devices to observe their internal parts. For example, the user can open cabinet units to examine their internal details and working (Fig. 11 illustrates the case of hard disks).

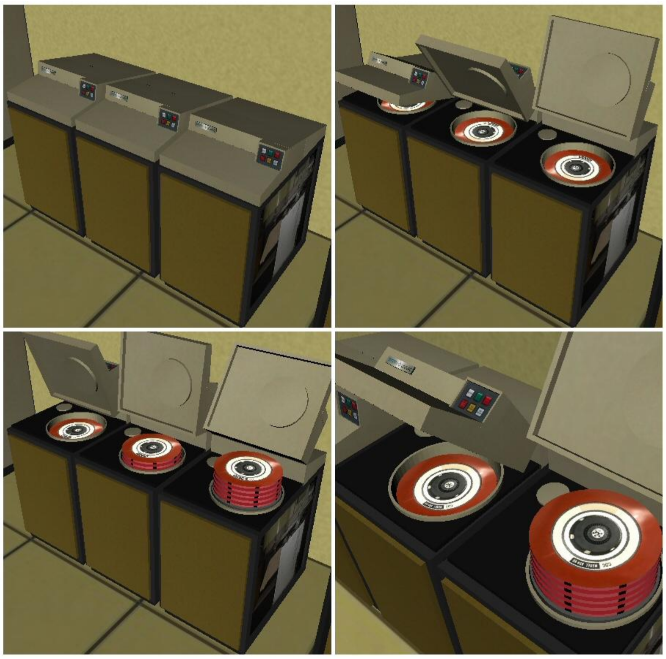


Fig. 11. Interactive components: hard disks

The virtual human leads the student through the environment, it presents different devices by following a logic order and by considering what information has been already presented to the student during the visit. If needed, the virtual human provides more detailed information, provides comparisons among different devices (e.g., punch card and card reader), by organizing logically the lesson, e.g. the virtual human explains what memory devices were used in the 70s before providing the detailed punch card description.

We have defined different HSMs corresponding to different virtual human behaviors, each one designed to highlight a different aspect of the Computer Science Museum (e.g., an high-level introduction to the overall environment, an explanation of the hardware architecture, a description of work activities). The user, according to her personal interests and needs, can choose (and change during the visit) the topics the virtual human explains.

7. Conclusions

This paper introduced the use of virtual humans in EVEs and presented a software architecture that allows Web3D content creators to integrate interactive H-Anim virtual humans acting as teachers, assistants or guides into these environments. The proposed solution enables distance-learning models completely based on Web standards.

We plan to integrate VHA with systems that are able to dynamically generate Web contents for educational purposes, e.g., the AHA! System (De Bra et al., 2003); student's knowledge, preferences and needs could be exploited by the virtual human to provide learners with tailored lessons. Moreover, since the time required for describing complex (in terms of number of states and transitions) virtual human behavior can be considerable (the content creator has to write by hand the specifications of the HSM), we plan to develop an authoring tool that allows the content creator to define the virtual human behavior either through a graphical user interface or by employing a more abstract programming language e.g., by using the Multimodal Presentation Markup Language (Prendinger et al., 2004), a XML-based language to easily describe multimodal presentation based on (2D and 3D) embodied agents.

References

Bell, J. T., & Fogler, H. S. (1996). Vicher: A Prototype Virtual Reality Based Educational Module for Chemical Reaction Engineering. *Computer Applications in Engineering Education*, *4*(4), 285-296.

Boulic, R., & Thalmann, D. (1992). Combined Direct and Inverse Kinematic Control for Articulated Figure Motion Editing. *Computer Graphics Forum*, *11*(4), 189-202.

Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Douville, B., Prevost, S., & Stone, M. (1994). Animated conversation: Rule-based generation of facial expression, gesture and spoken intonation for multiple conversational agents. In *Proceedings of SIGGRAPH'94: 21st Conference on Computer Graphics and Interactive Techniques* (pp. 413-420). New York: ACM Press.

Cavazza, M., & Simo, A. (2003). A virtual patient based on qualitative simulation. In *Proceedings of IUI'03: International Conference on Intelligent User Interfaces* (pp. 19-25). New York: ACM Press.

Chittaro, L., & Corvaglia, D. (2003). 3D Virtual Clothing: from Garment Design to Web3D Visualization and Simulation. In *Proceedings of Web3D'03: 8th International Conference on 3D Web Technology* (pp. 73-84). New York: ACM Press.

Chittaro, L., Ieronutti, L., & Ranon, R. (2004). Navigating 3D Virtual Environments by Following Embodied Agents: a Proposal and its Informal Evaluation on a Virtual Museum Application. *PsychNology Journal (Special issue on Human-Computer Interaction)*, *2*(1), 24-42.

Chittaro, L., & Ranon, R. (2005). Web3D Technologies in Learning, Education and Training: Motivations, Issues, Opportunities. *Computers & Education*, this issue.

Chittaro, L., Ranon, R., & Ieronutti, L. (2003). Guiding Visitors of Web3D Worlds through Automatically Generated Tours. In *Proceedings of Web3D'03: 8th International Conference on 3D Web Technology* (pp. 27-38). New York: ACM Press.

De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D., & Stash, N. (2003). AHA! The adaptive hypermedia architecture. In *Proceedings of the 14th ACM conference on Hypertext and hypermedia* (pp. 81-84). New York: ACM Press.

Economou, D., Mitchell, W., & Boyle, T. (2000). Requirements elicitation for virtual actors in collaborative learning environments. *Computers & Education*, *34*, 225-239.

Ferrington, G., & Loge, K. (1992). Virtual Reality: A New Learning Environment. *The Computing Teacher*, *19*(7), 16-19.

Funge, J., Tu, X., & Terzopoulos, D. (1999). Cognitive Modeling: Knowledge, Reasoning, and Planning for Intelligent Characters. In *Proceedings of SIGGRAPH'99*: *26th Conference on Computer Graphics* (pp. 29-38). New York: ACM Press.

H-Anim (2004). ISO/IEC FCD 19774. http://www.h-anim.org/Specifications/H-Anim200x/ISO_IEC_FCD_19774/ (last access on April 2005).

Hertz-Lazarowitz, R., & Shachar, H. (1990). Teacher's verbal behaviour in cooperative and whole-class instruction. In S. Sharan, *Cooperative learning theory and research* (pp. 77-94). New York: Praeger.

Hubal, R. C., Frank, G. A., & Guinn, C. I. (2003). Lessons learned in modeling schizophrenic and depressed responsive virtual humans for training. In *Proceedings of IUI'03: 7th International Conference on Intelligent User Interfaces* (pp. 85-92). New York: ACM Press.

Ieronutti, L., & Chittaro, L. (2005). A Virtual Human Architecture that Integrates Kinematic, Physical and Behavioral Aspects to Control H-Anim Characters. In *Proceedings of Web3D'05: 10th International Conference on 3D Web Technology* (pp. 75-83). New York: ACM Press.

Kaur, K., Maiden, N., & Sutcliffe, A. (1999). Interacting with virtual environments: An evaluation of a model of interaction. *Interacting with Computers*, *11*(4), 403-426.

Kayssi, A., El-Haji, A., Assir, M. E., & Sayyid, R. (1999). Web-based tutoring and testing in a computer networks course. *Computer Applications in Engineering Education*, *7*(1), 1-7.

Kim, J. H., Park, S. T., Lee, H., Yuk, K. C., & Lee, H. (2001). Virtual Reality Simulations in Physics Education. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*. http://imej.wfu.edu/articles/2001/2/02/ printver.asp (last access on April 2005).

Kizakevich, P. N., McCartney, M. L., Nissman, D. B., Starko, K., & Smith, N. T. (1998). Virtual Medical Trainer: Patient Assessment and Trauma Care Simulator. In J. D. Westwood, H. M. Hoffman, D. Stredney, & S. J. Weghorst, *Medicine Meets Virtual Reality - Art, Science, Technology: Healthcare (R)Evolution* (pp. 309-315). Amsterdam: IOS Press.

Kozak, J. J., Hancock, P. A., Arthur, E. J., & Chrysler, S. T. (1993). Transfer of training from virtual reality. *Ergonomics*, *36*(7), 777-784.

Lariani, L. C. (1994). *The virtual reality primer*. New York: McGraw-Hill.

Lester, J. C., Converse, S. A., Kahler, S. E., Barlow, S. T., Stone, B. A., & Bhogal, R. S. (1997). The persona effect: Affective impact of animated pedagogical agents. In *Proceedings of CHI'97: Conference on Human Factors in Computing Systems* (pp. 359-366). New York: ACM Press.

Lin M., & Gottschalk S. (1998). Collision Detection between Geometric Models: A Survey. In *Proceedings of IMA Conference on Mathematics of Surfaces* (pp. 37-56). Berlin: Springer-Verlag.

Lin, B. S., & Hsieh, C. T. (2001). Web-based teaching and learner control: A research review. *Computers & Education*, *37*, 377-386.

Mikk, E., Lakhnech, Y., & Siegel, M. (1997). Hierarchical Automata as Model for Statecharts. In *Proceedings of ASIAN'97: Asian Computing Science Conference on Advances in Computing Science* (pp. 181-196). Berlin: Springer-Verlag.

Ong, S. K., & Mannan, M. A. (2004). Virtual reality simulations and animations in a web-based interactive manufacturing engineering module. *Computers & Education*, *43*, 361-382.

Oreta, A. W. C. (1999). Developing a web-based learning module in a basic civil engineering course. *Computer Applications in Engineering Education*, *7*(4), 235-243.

Ponder ,M., Herbelin, B., Molet, T., Schertenlieb, S., Ulicny, B., Papagiannakis, G., Magnenat-Thalmann, N., & Thalmann, D. (2003). Immersive VR Decision Training: Telling Interactive Stories Featuring Advanced Virtual Human Simulation Technologies. In *EGVE'03: 9th Eurographics Workshop on Virtual Environments* (pp. 97-106). New York: ACM Press.

Ponder, M., Papagiannakis, G., Molet, T., Magnenat-Thalmann, N., & Thalmann, D. (2003). VHD++ Development Framework: Towards Extendible, Component Based VR/AR Simulation Engine Featuring Advanced Virtual Character Technologies. In *Proceedings of CGI'03: Computer Graphics International* (pp. 96-104). Los Alamitos: IEEE Computer Society Press.

Prendinger, H., Descamps, S., & Ishizuka, M. (2004). MPML: A markup language for controlling the behavior of life-like characters. *Visual Languages and Computing*, *15*(2), 183-203.

Rickel, J., & Johnson, W. (1999). Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control. *Applied Artificial Intelligence*, *13*, 343-382.

Sanz, S., Iskander, M. F., & Yu, L. (2000). Development of an interactive multimedia module on antenna theory and design. *Computer Applications in Engineering Education*, *8*(1), 11-17.

Sims, E. (2005). Reusable, Lifelike Virtual Humans for Mentoring and Role-Playing. *Computers & Education*, this issue.

Sonnet, H., Carpendale, S., & Strothotte, T. (2004). Integrating expanding annotations with a 3D explosion probe. In *Proceedings of AVI'04: International Conference on Advanced Visual Interfaces* (pp. 63-70). New York: ACM Press.

Traum, D., & Rickel, J. (2002). Embodied agents for multi-party dialogue in immersive virtual worlds. In *Proceedings of AAMAS'02: International Conference on Autonomous Agents and Multiagent Systems* (pp. 766-773). New York: ACM Press.

Trindade, J., Fiolhais, C. & Gil V. (1999). Virtual Water, an application of virtual environments as an education tool for physics and chemistry. In *Proceedings of ICCE'99: International Conference on Computers in Education* (pp. 655-658), 2. Ohmsha: IOS Press.

Vince, J. (1995). *Virtual Reality Systems*. Reading, MA: Addison-Wesley.

VRML, (1997). The Virtual Reality Modeling Language.
http://www.web3d.org/x3d/specifications/vrml/ISO_IEC_14772-All/index.html (last access on April 2005).

Wexelblat, A. (1993). *Virtual Reality: Applications and Explorations*. San Diego: Academic Press.

X3D, (2004). ISO/IEC FDIS 19775. http://www.web3d.org/x3d/specifications/index.html (last access on April 2005).

Zahorian, S., Swart, W., Lakdawala, V., Leathrum, J., & Gonzales, O. (2000). A modular approach to using computer technology for education and training. *International Journal of Computer Integrated Manufacturing*, *13*(3), 286-297.