Why Computerized Models to Control Virtual Humans ?

Daniel Thalmann, EPFL, Switzerland, Member, IEEE

Abstract—Interactive systems, games, VR and multimedia systems require more and more flexible Virtual Humans with individualities. There are mainly two approaches: 1) Recording the motion using motion capture systems, then to try to alterate such a motion to create this individuality. This process is tedious and there is no reliable method at this stage. 2) Creating computational models which are controlled by a few parameters. One of the major problem is to find such models and to compose them to create complex motion. Such models can be created for walking, grasping, but also for groups and crowds.

Index Terms—animation, virtual human, walking, grasping, groups, crowds

I. INTRODUCTION

Virtual humans simulations are becoming each time more popular. Nowadays many systems are available to animate virtual humans. Such systems encompass several different domains, as: autonomous agents in virtual environments, human factors analysis, training, education, virtual prototyping, simulation-based design, and entertainment. In the context of Virtual Humans, a Motion Control Method (MCM) specifies how the Virtual Human is animated and may be characterized according to the type of information it privileged in animating this Virtual Human. For example, in a keyframe system for an articulated body, the privileged information to be The problem is basically to be able to generate variety among a finite set of motion requests and then to apply it to either an individual or a member of a crowd. A single autonomous agent and a member of the crowd present the same kind of 'individuality'. The only difference is at the level of the modules that control the main set of actions. With this formulation, one can also see that the personality of an agent (i.e. the set of noisy actions) can be preserved whenever it is in a crowd, alone. Figure 1 shows a group of Virtual Humans in a room and Figure 2 Virtual Humans in city park.

The problem is basically to be able to generate variety among a finite set of motion requests and then to apply it to either an individual or a member of a crowd. A single autonomous agent and a member of the crowd present the same kind of 'individuality'. The only difference is at the level of the modules that control the main set of actions. With this formulation, one can also see that the personality of an agent (i.e. the set of noisy actions) can be preserved whenever it is in a crowd, alone.

Figure 1. A group of Virtual Humans

Figure 2. Virtual Humans in a city park

To create this flexible Virtual Humans with individualities, there are mainly two approaches:

- Recording the motion using motion capture systems (magnetic or optical), then to try to alterate such a motion to create this individuality. This process is tedious and there is no reliable method at this stage.
- Creating computational models which are controlled by a few parameters. One of the major problem is to find such models and to compose

Computer Graphics Lab, EPFL, CH 1015 Lausanne, Switzerland
htttp://ligwwww.epfl.ch

them to create complex motion. Such models can be created for walking, running, grasping, but also for interaction, groups, and crowds.

II. MOTION CAPTURE AND RETARGETING

A. Introduction

The first approach consists in recording the motion (Fig. 3) using motion capture systems (magnetic or optical), then to try to alterate such a motion to create this individuality. This process is tedious and there is no reliable method at this stage. Even if it is fairly easy to correct one posture by modifying its angular parameters (with an Inverse Kinematics engine, for instance), it becomes a difficult task to perform this over the whole motion sequence while ensuring that some spatial constraints are respected over a certain time range, and that no discontinuities arise. When one tries to adapt a captured motion to a different character, the constraints are usually violated, leading to problems such as the feet going into the ground or a hand unable to reach an object that the character should grab. The problem of *adaptation* and *adjustment* is usually referred to as the *Motion Retargeting Proble*m.

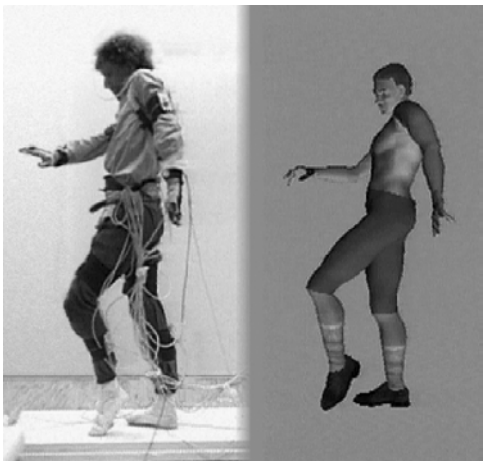

Figure 3. Motion capture

Witkin and Popovic [1] proposed a technique for editing motions, by modifying the motion curves through warping functions and produced some of the first interesting results. In a more recent paper [2], they have extended their method to handle physical elements, such as mass and gravity, and also described how to use characters with different numbers of degrees of freedom. Their algorithm is based on the reduction of the character to an *abstract character* which is much simpler and only contains the degrees of freedom that are useful for a particular animation. The edition and modification are then computed on this simplified character and mapped again onto the end user skeleton. Bruderlin and Williams [3] have described some basic facilities to change the anima-tion, by modifying the motion parameter curves. The user can define a particular posture at time t, and the system is then responsible for smoothly blending the motion around *t* . They also introduced the notion of *motion displacement ma*p, which is an offset added to each motion curve. The *Motion Retargeting Problem* term was brought up by Michael Gleicher [4] . He designed a space-time constraints solver, into which every constraint is added, leading to a big optimisation problem. He mainly focused on optimising his solver, to avoid enormous computation time, and achieved very good results. Bindiganavale and Badler [5] also addressed the motion retargeting problem, introducing new elements: using the zero-crossing of the second derivative to detect significant changes in the motion, visual attention tracking (and the way to handle the gaze direction) and applying Inverse Kinematics to enforce constraints, by defining six sub-chains (the two arms and legs, the spine and the neck). Finally, Lee and Shin [6] used in their system a coarse-to-fine hierarchy of B-splines to interpolate the solutions computed by their Inverse Kinematics solver. They also reduced the complex-ity of the IK problem by analytically handling the degrees of freedom for the four human limbs

Lim and Thalmann [7] have addressed an issue of solving customers' problems when applying evolutionary computation. Rather than the seemingly more impressive approach of wow-it-all-evolved- from-nothing, tinkering with existing models can be a more pragmatic approach in doing so. Using interactive evolution, they experimentally validate this point on setting parameters of a human walk model for computer animation while previous applications are mostly about evolving motion controllers of far simpler creatures from scratch. Figure 4 shows an example of such application of evolutionary computation.

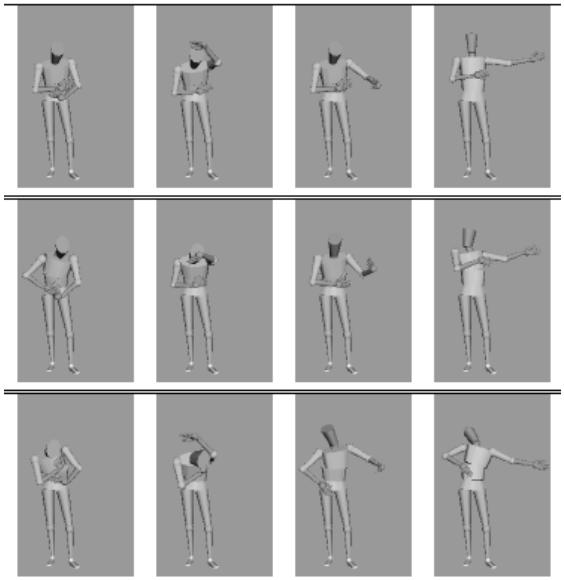

Figure 4. Evolutionary Computation: The original motion of the first row has evolved into the motion in rows 2 and 3.

B. Using an intermediate skeleton

Given a captured motion associated to its Performer Skeleton, we decompose the problem of retargeting the motion to the *End User Skeleton* into two steps

- First, computing the Intermediate Skeleton matrices by orienting the Intermediate Skeleton bones to reflect the Performer Skeleton posture *(Motion Converte*r).
- Second, setting the End User Skeleton matrices to the local values of the corresponding Intermediate Skeleton matrices.

The first task is to convert the motion from one hierarchy to a completely different one. We introduce the *Intermediate Skeleton* model to solve this, implying three more subtasks: manually set at the beginning the correspondences between the two hierarchies, create the Intermediate Skeleton and convert the movement. We are then able to correct the resulting motion and make it enforce Cartesian constraints by using Inverse Kinematics. When considering motion conversion between different skeletons, one quickly notices that it is very difficult to directly map the Performer Skeleton values onto the End User Skeleton, due to their different proportions, hierarchies and axis systems. This raised the idea of having an *Intermediate Skeleton*: depending on the Performer Skeleton posture, we reorient its bones to match the same directions. We have then an easy mapping of the Intermediate Skeleton values onto the End User Skeleton. The first step is to compute the Intermediate Skeleton *(Anatomic Binding* module). During the animation, motion conversion takes two passes, through the *Motion Converter* and the *Motion Composer* (which has a graphical user interface). The next sections discuss the creation of the Intermediate Skeleton, the motion conversion and demonstrate the importance of the initial Intermediate Skeleton posture.

III. CREATING COMPUTIONAL MODELS

The second approach consists in creating computational models which are controlled by a few parameters. One of the major problem is to find such models and to compose them to create complex motion. Such models can be created for walking or grasping objects, but also for groups and crowds.

A. Walking

Walking has global and specific characteristics. From a global point of view, every human-walking has comparable joint angle variations. However, at a close-up, we notice that individual walk characteristics are overlaid to the global walking gait.

We use the walking engine described in [8] which has been extended in the context of a european project on virtual human modeling [9] . Our contribution consists in integrating the walking engine as a specialized action in the animation framework. Walking is defined by the following characteristics (Figure 5-9): • it is a motion where the centre of gravity alternatively balances from the right to the left side.

- at any time, at least one foot is in contact with the floor, the 'single support' duration *(d*s).
- there exists a short instant during the walk cycle, where both feet are in contact with the floor, the 'double support' duration *(dd*s).
- it is a periodic motion which has to be normalized in order to adapt to different anatomies.

The joint angle variations are synthesized by a set of periodic motions which we briefly mention here:

- sinus functions with varying amplitudes and frequencies for the humanoid's global translations (vertical, lateral and frontal) and the humanoid's pelvic motions (forward/backward, left/right and torsion)
- periodic functions based on control points and interpolating hermite splines.They are applied to the hip flexion, knee flexion, ankle flexion, chest torsion, shoulder flexion and elbow flexion.

The parameters of the joint angle functions can be modified in a configuration file in order to generate personalized walking gaits, ranging from tired to energetic, sad to happy, smart to silly. The algorithm also integrates an automatic speed tuning mechanism which prevents sliding on the supporting surface. Many high level parameters can be adjusted dynamically, such as linear and angular velocity, foot step locations and the global walk trajectory. The walk engine has been augmented by a specialized action interface and its full capacity is therefore available within the animation framework. The specialized action directly exports most common high level parameter adjustment functions. For fine-tuning, it is still possible to explicitly access the underlying motion generator. The walk animation engine has been developed in the early nineties. However it suffered from not being easily combined with other motions, for example a walking human giving a phone call with a wireless phone was hardly possible. Now, that the walking engine is integrated as a specialized action, a walking and phoning human is easily done, simply by performing the walk together with a 'phone'-keyframe for example. In Figure 5, we show some of the parameterized gaits achieved through the specialized action interface.

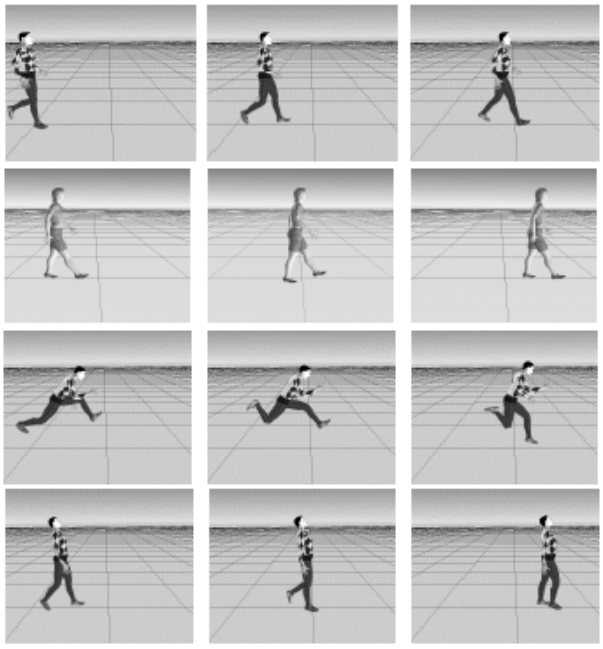

Figure 5. Individualized walking

B. Grasping

For grasping object, our approach is based on three steps:

- **Heuristic grasping decision.** Based on a grasp taxonomy, Mas and Thalmann [10] proposed a completely automatic grasping system for synthetic actors. In particular, the system can decide to use a pinch when the object is too small to be grasped by more than two fingers or to use a two-handed grasp when the object is too large.
- **Inverse kinematics to find the final arm posture**
- **Multi-sensor hand.** Our approach [11] is adapted from the use of proximity sensors in Robotics [12], the sensor-actuator networks [13] and our own work on human grasping [10]. In our work, the sphere multi-sensors have both touch and length sensor properties, and have been found very efficient for synthetic actor grasping problem. Multi-sensors are considered as a group of objects attached to the articulated figure. A sensor is activated for any collision with other objects or sensors. Here we select sphere sensors for their efficiency in collision detection.

In case of large objects, such as furniture, grasping simultaneously involves two or more persons. Therefore, we focused on a multi-agent grasp action for encumbering objects. As the object's weight and geometry is distributed over several hand support points of different agents, the heuristic motion planning schemes have to be different than the ones for an object grasp performed by a single individual. For example, a large object might be grasped frontally by the first agent and from behind by the second agent (see Figure 6).

The humanoid is the active agent, the balloon the passive agent. We can reverse the role of active and passive agent, e.g. the balloon can be active and the human passive (Figure 7). The choice of the active and passive agents depends on which agent is supposed to control the other one – is the human carrying the balloon or is the balloon lifting the human into the air? By extension, any agent can be active and passive at the same time, e.g. a box attaches a balloon and is attached to a humanoid.

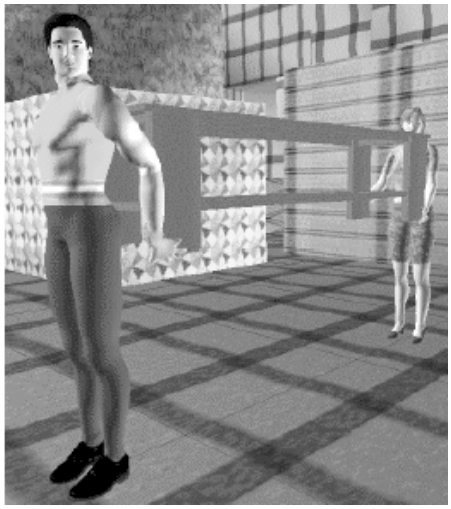

Figure 6. Multi-agent carrying

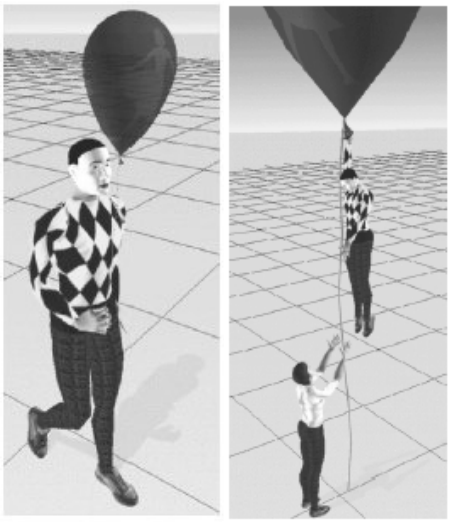

Figure 7. Is the human carrying the balloon or is the balloon lifting the human into the air?

IV. CROWDS AND GROUPS

Animating crowds [14] is challenging both in character animation and a virtual city modeling. Though different textures and colors may be used, the similarity of the virtual people would be soon detected by even non-experts, say, "everybody walks the same in this virtual city!" . It is, hence, useful to have a fast and intuitive way of generating motions with different personalities depending on gender, age, emotions, etc., from an example motion, say, a genuine walking motion. The problem is basically to be able to generate variety among a finite set of motion requests and then to apply it to either an individual or a member of a crowd. It also needs very good tools to tune the motion [15].

The proposed solution addresses two main issues: i) crowd structure and ii) crowd behavior. Considering *crowd structure*, our approach deals with a hierarchy composed of crowd, groups and agents, where the groups are the most

complex structure containing the information to be distributed among the individuals. Concerning *crowd behavior*, our virtual agents are endowed with different levels of autonomy. They can either act according to an innate and scripted crowd behavior *(programmed behavio*r), react as a function of triggered events *(reactive or autonomous behavio*r) or be guided by an interactive process during simulation *(guided behavio*r). We introduced the term <guided crowds> to define the groups of virtual agents that can be externally controlled in real time [16]. Figure 8 shows a crowd guided by a leader.

Figure 8. Crowd guided by a leader

In our case, the intelligence, memory, intention and perception are focalized in the group structure. Also, each group can obtain one leader. This leader can be chosen randomly by the crowd system, defined by the user or can emerge from the sociological rules. Concerning the crowd control features, The crowd aims at providing autonomous, guided and programmed crowds. Varying degrees of autonomy can be applied depending on the complexity of the problem. Externally controlled groups, *<guided groups>*, no longer obey their scripted behavior, but act according to the external specification. At a lower level, the individuals have a repertoire of basic behaviors that we call *innate behavior*s. An innate behavior is defined as an "inborn" way to behave. Examples of individual innate behaviors are goal seeking behavior, the ability to follow scripted or guided events/reactions, the way trajectories are processed and collision avoided. While the innate behaviors are included in the model, the specification of scripted behaviors is done by means of a script language. The groups of virtual agents whom we call *<programmed groups>* apply the scripted behaviors and do not need user intervention during simulation. Using the script language, the user can directly specify the crowd or group behaviors. In the first case, the system automatically distributes the crowd behaviors among the existing groups. Events and reactions have been used to represent behavioral rules. This reactive character of the simulation can be programmed in the script language (scripted control) or directly given by an external controller. We call the groups of virtual agents who apply the behavioral rules *<autonomous groups>*.

The train station simulation (Figure 9) includes many different actions and places, where several people are present and doing different things. Possible actions include "buying a ticket", "going to shop", "meeting someone", "waiting for someone", "making a telephone call", "checking the timetable", etc. This simulation uses external control (RBBS [12][24]) to guide some crowd behaviors in real time.

Figure 9. Train station simulation.

CONCLUSION

In order to develop truly interactive multimedia systems with Virtual Humans, games, and interactive movies, we need a flexible way of animating these Virtual Humans. Altering motion obtained from a motion capture system is not the best solution. Only computational models can offer this flexibility unless powerful motion retargeting methods are developed, but in this case they will look similar to computational models.

ACKNOWLEDGMENTS

The author would like to thank all people who have contributed to these projects especially Luc Emering, Soraia Musse, Ik Soo Lim, and Mireille Clavien. Research has been partly funded by the Swiss National Foundation for Research and the Federal Office for Education and Science in the framework of the CROSSES project.

REFERENCES

[1] A.Witkin, Z.Popovic. Motion warping. *Proceedings of SIGGRAPH 9*5, pages 105–108, August 1995. ISBN 0-201-84776-0. Held in Los Angeles, California.

[2] Z.Popovic, A.Witkin. Physically based motion transformation. Proceedings of SIGGRAPH 99, pages 11–20, August 1999. ISBN 0-20148-560-5. Held in Los Angeles, California.

[3] A.Bruderlin, L.Williams. Motion sig-nal processing. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceeding*s, Annual Conference Se-ries, pages 97–104. ACM SIGGRAPH, Addison Wes-ley, August 1995. held in Los Angeles, California, 06- 11 August 1995.

[4] M.Gleicher. Retargeting motion to new characters. In Michael Cohen, editor, *SIGGRAPH 98 Con-ference Proceeding*s, Annual Conference Series, pages 33–42. ACM SIGGRAPH, Addison Wesley, July 1998. ISBN 0-89791-999-8.

[5] R.Bindiganavale, N. I. Badler. Motion abstraction and mapping with spatial constraints. In N. Magnenat-Thalmann and D. Thalmann, editors, *Modeling and Motion Capture Techniques for Virtual Environment*s, Lecture Notes in Artificial Intelligence, pages 70–82. Springer, November 1998. held in Geneva, Switzerland, November 1998.

[6] L.Jehee, S.Y.Shin. A hierarchical approach *Proceedings of SIGGRAPH 9*9, pages 39–48, Au-gust 1999. ISBN 0-20148-560-5. Held in Los Angeles, California.

7 I.S.Lim, D.Thalmann, Solve Customers' Problems: Interactive Evolution for Tinkering with Computer Animation, Proc. 2000 ACM Symposium on Applied Computing (SAC2000), pp. 404-407
.

8 R.Boulic, N.Magnenat-Thalmann, D.Thalmann, A Global Human Walking Model with Real-time Kinematics Personification ,The Visual Computer, Vol.6, No6, December 1990, pp.344-358.

9 R.Boulic, T.Capin, Z.Huang, L.Moccozet, T.Molet, P.Kalra, B.Lintermann, N.Magnenat-Thalmann, I.Pandzic, K.Saar, A.Schmitt, J.Shen, D.Thalmann, The HUMANOID Environment for Interactive Animation of Multiple Deformable Human Characters, Proc. Eurographics `95, Maastricht, August 1995, pp.337-348.

10 R.Mas, D.Thalmann, A Hand Control and Automatic Grasping System for Synthetic Actors, Proc. Eurographics '94, Oslo

11 Z.Huang, R.Boulic, N.Magnenat-Thalmann, D.Thalmann, A Multi-sensor Approach for Grasping and 3D Interaction, Proc. Computer Graphics International `95, Leeds, Academic Press, pp.235-254.

12 B.Espiau, R.Boulic Collision avoidance for redondants robots with proximity sensors, Proc. of Third International Symposium of Robotics Research, Gouvieux, October.Espiau and Boulic 1985

13 M.van de Panne, E.Fiume Sensor-Actuator Network, Computer Graphics, Annual Conference Series, 1993, pp.335-342.

14 S.R. Musse, D.Thalmann, A Behavioral Model for Real-Time Simulation of Virtual Human Crowds, IEEE Transactions on Visualization and Computer Graphics, Vol.7, No2, 2001, pp.152-164.

15 L.Emering, R.Boulic, T.Molet, D.Thalmann, Versatile Tuning ofHumanoid Agent Activity, Computer Graphics Forum

16 Musse, S.R., Babski, C., Capin, T. and Thalmann, D. Crowd, Modelling in Collaborative Virtual Environments. ACM VRST ‘98, Taiwan