Embedding a Motion-Capture Interface in a Control Structure for Human–Like Agent Behavior Achievement

LUIZ GARCIA[1] FERNANDO DA SILVA[2,3] ANTONIO OLIVEIRA[2] LUIZ VELHO[3] JONAS GOMES[3]

[1] Laboratoire d'Analyse et d'Architecture des Systemes (LAAS/CNRS)
7, Avenue du Colonel Roche, 31077 Toulouse France
lmgarcia@laas.fr

[2]LCG - Laboratório de Computação Gráfica, COPPE - Sistemas / UFRJ
21945-970, Rio de Janeiro, RJ, Brazil, Caixa Postal 68511
(nando,oliveira)@lcg.ufrj.br

[3]VISGRAF Laboratory, IMPA–Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina, 110, 22460 Rio de Janeiro, RJ, Brazil
(nando,lvelho,jonas)@visgraf.impa.br

Abstract. In this work we provide methodology to embed a motion-capture interface in a software control architecture, with which to implement artificial animated agents with more human like behavior. The resulting architecture deals with dynamics and kinematics issues and provides a well structured way to control the agent movements, making them more natural and smooth over time. Demonstrations using an artificial agent, human-like shaped, show the versatility of the architecture. As a practical result, by using the control structure, the above artificial animated human agent is able to perform tasks involving decisions for motion reparametrization, achieving more human-like motion and behavior.

Keywords. Motion Capture, Agent Simulation, Computer Animation, Motion Control.

1 Introduction

Over the past decades, several software architectures has been designed to support agent simulation. See for example [1,2,3,4], and others. In general, these architectures do not provide sensing capabilities. Mainly because simulating sensors is kind of a hard task, due to the lack of knowledge about what is really important to be sensed (or perceived) by artificial agents. Also, the architectures do not use a control theory approach to design the manipulators control laws. In general, physics (dynamics and kinematics) issues are neglected in the manipulation of the resources, or degrees of freedom (DOFs), of the agent. Also, in general, a geometric model of a scene is previously given to a virtual agent, which acts directly based on the geometric information contained in the model. Finally, certain useful human skills like motion characteristics and behavior are not embedded. As a result of the above issues, in general, the agent's pose and perceptual state are arbitrated, and its movements, generally attached to geometric models, are disorganized and irregular. This turns it into a simple task executor, without decision or reaction capabilities. Also, this generates agents with notedly unreal behaviors.

Here, we extend the concepts and the architecture presented in [5] used to perform general animated agents, by embedding a motion capture interface in order to achieve more human-like behavior. The model also supports sensing capabilities and deals with physics constraints as dynamics and kinematics issues. In the adopted philosophy, the creature's processes do not deal with the construction of a geometric model, in order to represent the environment. From sensory information (the "sensory buffer") the agent processes compute a "perceptual buffer", which is joined to other type of information, like the agent's pose and functional state, to define the current "perceptual state". So, the agent can make decisions and act mainly based on the information contained in this perceptual state. Furthermore, the attached motion-capture interface allows to perform a on-line reparametrization on the agent's motion, during the execution of each step of motion, in order to pursue the task goal. This embedding makes the computer animation of human agents more realistic and also allows to deal with unexpected changes in its perceptual state in a more natural way. A key aspect in computer animation is to produce movements that are smooth over time. This can be guaranteed by using a control-theory approach and by considering motion dynamics and kinematics aspects. But, only by using these, our agent acts like an automata (a robot). If a rapid change occurs in its perceptual state, the agent would change its direction or make movements that are not natural.

So, here, we combine our previous control-theory approach with a motion-capture interface to produce more natural and realistic movements for our human agent in real-time. The resulting architecture also allows the agent to have reaction capabilities.

By looking the results of the experiments and demonstrations performed, we can see that this approach contributes with some improvements. First, it is easier for the creature to react, choosing actions, based on its own perception of the world rather than by using directly a geometric model. Second, the system deals with less data, what substantially improves its performance. Third, by embedding the motion capture interface and by using a control theory approach, considering physics constraints as dynamics and kinematics issues, we approximate simulation and reality. Our agent behaves more naturally. Finally, the resulting architecture allows to deal with dynamic environments in a more efficient way, providing real-time feedback to a eventual stimuli change. Our agent system has human movements recorded in its memory and at a time of a decision, one action is mapped into the sub-movements necessary to perform it. So, the main advantage of using such architecture is to allow the agent to act autonomously, imitating human motion behavior.

2 The Control Architecture

Figure 1 shows the main aspects of the control architecture. Briefly, we use a simple model, a world description that is not an inherent part of the agent kernel system, and some computer graphics techniques to derive sensory information. So, the information in the agent sensory buffer is provided by a high-level application, which simulates acquisition devices like cameras or haptics sensors (including tactile and proprioceptive information). Then, information from the sensory buffers are abstracted to construct a perceptual buffer (feature maps). The creature uses this set of features plus its pose and functional state to define its perceptual state. Using this perceptual state, the task controller evaluates the current task goal achievement. Based on the results of this evaluation, it re-parameterizes the current task, allowing the creature to choose the right set of actions to perform, or eventually it selects a new task, if goal is achieved. After a task selection or reparametrization, the attentional mechanism operates based on the current task parameters. Based on a task dependent policy, it selects a new action. If this action involves changing attention, it changes the current attention window (topic of interest), telling the agent where to put its attentional focus. Then, the motion-capture interface, takes the generated displacements and defines which set of sub-movements (or its parameters) will be retrieved from its data-base to be applied to the degrees of freedom of our agent to attend the new position. Basically, it maps the immediate actions to recorded human movements, constrained by a certain "metric" inherent to the task goal achievement. These parameters are sent trough the agent controllers, which effectively brings the agent to a new pose and also puts a new set of information in the sensory buffers, re-starting the cycle (feature extraction).

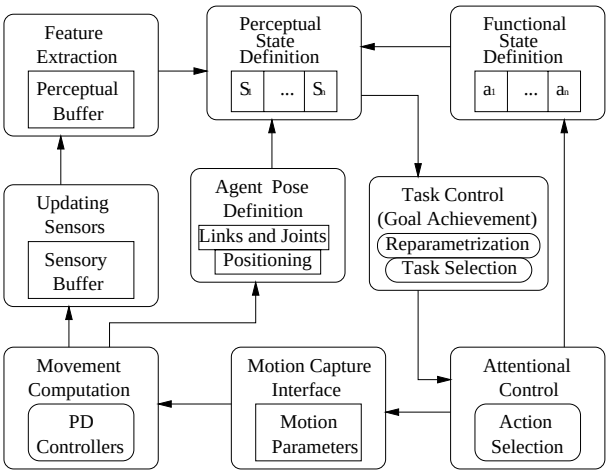

Figure 1: System Control Architecture

3 Agent Representation

In our context, an artificial animated human agent can be simply understood as a device that can partially imitate a human being. The main characteristic of an artificial human agent is its "real-time" sensing and reacting capability. Such being is a dynamic entity that learns in some automated way, and reacts to stimuli performing right related actions as feedback.

In our architecture, the agent is treated as a skeleton. Its topology is represented by a graph formed by joints and links. Its geometry is represented by a set of connected links (Figure 2), that may assume any level of realism depending on the application. This description is adequate to be used in most animation systems available today, since it reflects the structure of an articulated figure.

At the programming level, the actor is represented by using a modified version of Zeltzer's APJ (*Axis Position Joint*) structure [6], adapted to work with motion captured data. As we will describe later, such representation allows us to deal with motion processing techniques in a straightforward way.

4 Capturing and Representing Human Motions

We have decided to use motion capture as the basic component of locomotion for our human-like agent because this

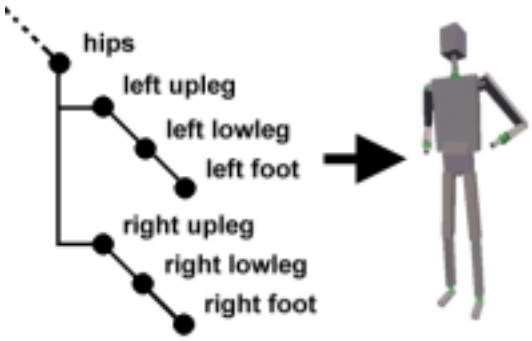

Figure 2: Topological and geometrical description of the agent.

provides more realism and makes possible the real–time visualization. On the other hand, captured motions act like scripts and are hard to modify in order to be adapted to different situations according to user interaction. However, such drawback can be avoided by using well known motion processing techniques.

4.1 The Motion Capture Process

The information generated by a motion capture device is composed by a data set of samples, which represent the position and global orientation of a real object at uniformly spaced instants of time. In the case of human motion capture, the position and orientation of several joints of an actor are recorded, generating a set of 1D signals also known as motion curves. These curves are then processed and mapped onto a skeleton hierarchy, which will drive a virtual actor (that is, an agent) in the computer [7].

There is a large diversity of motion capture hardware available nowadays, from simple mechanical devices to sophisticated optical systems. Mechanical systems are composed of potentiometers (or sliders) that measure the position or orientation of the joints in an object. Its similarity with conventional stop-motion techniques that are widely used in movie production allows a natural migration of traditional animators, thus increasing the popularity of this technique. However, the realism of mechanically captured motions still depends, in great part, on the ability and patience of the animator.

Systems based on magnetic technology are probably the most popular ones. Both positional and angular data of the joints of a real subject are captured, using a set of sensors that measure the magnetic field generated by a source. Their main advantage is the possibility of real-time animation of virtual characters, thus offering to the TV industry new possibilities in the field of virtual sets. Some drawbacks of this technology are: the sensitivity to metals in the capturing area, what introduces some noise into the final data; the high level of encumbrance, due to the great number of cables attached to the actor; and the sampling rate, very low for fast sport motions.

Optical systems are based on high contrast video imaging of retro-reflective markers, that are placed on objects whose motion is being recorded. This technique provides high sampling rates, but the recorded motion data must be post-processed using computer vision tracking techniques. In the tracking process, the centroids of markers are matched in images from pairs of cameras, using a triangulation to compute the positional data of these markers in 3D space. This process introduces artifacts (offsets) into the final data. Some disadvantages of the optical process are the occlusion of one or more markers during the capturing session, the lack of angular data, and the sensitivity to background light and reflective objects.

5 Motion Combination and Control

As seen in the previous section, the signal–like nature of captured data suggests that it should be treated using the paradigms of signal processing theory [8]. Motion capture data processing has become an important field of research in recent years [9]. The crescent demand of powerful tools for motion editing has led to the development of several techniques such as warping [10], concatenation [11] and reparametrization [12]. We use these tools to combine small pieces of motion such as to generate new movements according to user interaction, in a process that will drive the agent in the virtual world. The set of motion processing tools used by our agent includes a set of well defined operations, discussed next.

Motion Filtering

Filters may be used to eliminate or attenuate specific frequencies of a motion signal. A low-pass filter can be applied to the motion curves of the agent in order reduce the noise introduced by the capturing process, thus making the motion more natural and smooth. Also, special filters can be used to create slow-motion and accelerated-time effects in a motion sequence [12].

Motion Concatenation

Concatenation is a very simple operation which is widely used in many commercial applications involving avatars and user-driven agents, such as FIFA Soccer and Virtual Fighter. This technique consists on sequencing a number of motions, chosen from a memory-based motion library according to user interaction, performing a smooth transition between them. This can be done by using special motion interpolators such as spacetime [11] and physically based constraints.

Motion Warping

Motion can be deformed by applying a warping function on its motion curves. This causes changes in the orientation and position of specific sets of joints of the agent, and may be used to avoid obstacles in virtual scenes. Figure 3 shows an example of motion warping. In this case, the original motion was predicted as a straight line walk. But, when the agent walks over a virtual scenario with an obstacle, a collision is expected to occur (Figure 3, a). Using the perceptual information extracted from the sensory buffers, the agent's internal motion engine can calculate the necessary warp factor that must be applied to a group of joints such as to avoid the collision with the obstacle (Figure 3, b).

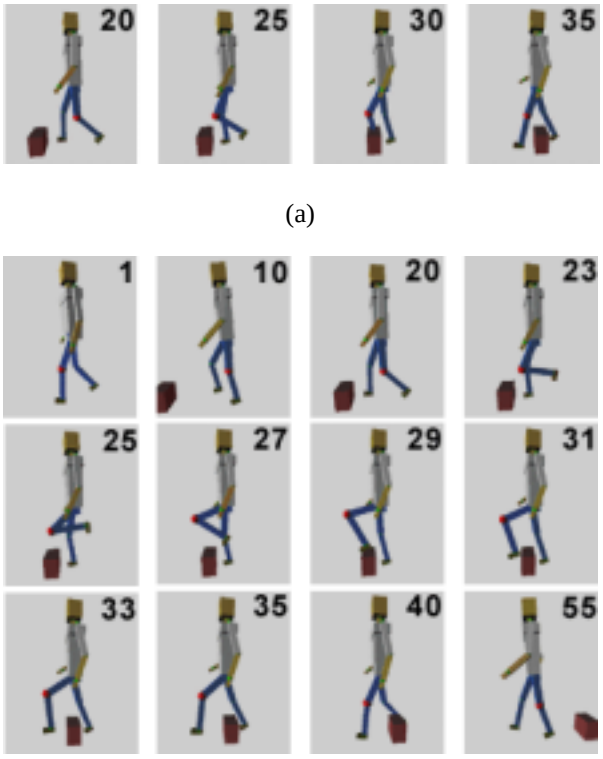

Figure 3: Agent performing motion warping.

Motion Time-Warping

By using time-warping algorithms, the agent motions can be extended or compressed in time. The basic goal is to deform the motion without producing changes in the frequency contents of its curves. In the expansion case, this causes a cycling in the motion [13]. Similar techniques are used in soccer games in order to repeat a basic motion over time (e.g. a player pursuing the ball: cycling of a running motion).

6 Conclusions and Future Work

By using the motion-capture interface embedded in our previous architecture we have produced more natural motion behaviors for our human agent. Besides this basic contribution, this interface also has some other functionalities. First, a set of human-like movements of artists can be captured and a data-base containing these motion-parameters generated and easy maintained. Second, by applying some training strategy the agent has a way to learn what set of movements is appropriated to what task or subtask. One could use for example reinforcement learning [14] or other approach. In our basic obstacle avoidance experiments, we have determined the operation (warping) and motion parameters empirically, based on the task goal. After this learning phase, the agent can be more autonomous, choosing the right parameters, according to changes in its perceptual state.

Here we have also showed how the human-like agent can deal with unexpected events on-line, for example avoiding obstacles. Based on the parameters given by the attentional process (a direction of motion) and on the information of the agent perceptual state (the discovery of an obstacle in the direction of motion), the system gives as answer the right set of motion operations, also retrieving the related set of movements. These movements would be performed until the task goal is achieved (obstacle is avoided) or other eventual stimuli appears, changing its perceptual state again. Every change in the perceptual state would require a new operator set, as well as motion parameters.

So, we have counter-balanced the drawback of using previously recorded motion sequences by producing (hand-coding) a real-time decision strategy which tells what motion processing technique shall be used next, based on the information currently in the agent perceptual state. The motion operations produce the desired real-time changes in our agent without needs of a programmer interference. The immediate extension of this work is to make our agent "learn" this capability autonomously, according to the information existent in its perceptual state and a task set. The result would be an inherently reactive agent, which takes decisions based on its own perception of the world, perhaps closer to a biological model in its motion behavior.

7 References

[1] PERLIN, K., Real-time Responsive Animation with Personality. In *IEEE Transactions on Visualization and Computer Graphics*, Vol 1, No.1, March 1995.

[2] MOORE, M. AND WILHELM, J. Collision detection and response for computer animation. In *SIG-*

GRAPH'88 Conference Proceedings, volume 4, 1988, pages 289–298. ACM SIGGRAPH.

[3] COSTA, M. AND FEIJO, B An architecture for concurrent reactive agents in real-time animation.In *Proc. of SIBGRAPI'96*, 1996, pages 281–288.

[4] FARENC, N. BOULIC, R. AND THALMAN, D. An informed environment dedicated to the simulation of virtual humans in urban context. In *Proc. Eurographics'99*, 1999, pages 309–318.

[5] GARCIA, LUIZ M. G. AND DA SILVA, FERNANDO Towards an Architecture for Artificial Animated Creatures. To appear in Proc. of CGS/IEEE Computer Animation 2000 Conference. Philadelphia PA, USA, May 3-5th 2000.

[6] ZELTZER, D., Motor Control Techniques for Figure Animation. In *IEEE Computer Graphics and Applications*, volume 2, no. 9, pp. 53–59, 1982.

[7] SILVA, F., VELHO, L., CAVALCANTI, P., GOMES, J., An Architecture for Motion Capture Based Animation. In *Proceedings of SIBGRAPI'97, X Brazilian Symposium of Computer Graphics and Image Processing* (October 1997), pp. 49–56, IEEE Press.

[8] WILLIAMS, L., BRUDELIN, A., Motion Signal Processing. In *Computer Graphics (SIGGRAPH'95 Proceedings)* (August 1995), pp. 97–104.

[9] SILVA, F., Um Sistema de Animação Baseado em Movimento Capturado. Master Thesis, Federal University of Rio de Janeiro - COPPE/UFRJ, March 1998.

[10] WITKIN, A., POPOVIC, Z., Motion Warping. In *Computer Graphics (SIGGRAPH'95 Proceedings)* (August 1995), pp. 105–108.

[11] COHEN, M., ROSE, C., GUENTER, B., BODENHEIMER, B., Efficient Generation of Motion Transitions Using Spacetime Constraints. In *Computer Graphics (SIGGRAPH'96 Proceedings)* (August 1996), pp. 147–154.

[12] SILVA, F., VELHO, L., GOMES, J., Motion Reparametrization. In *EUROGRAPHICS Technical Note (short-papers proc.)* (September 1998), pp. 1.5.1–1.5.4, Springer-Verlag.

[13] SILVA, F., VELHO, L., GOMES, J. AND GOLDENSTEIN, S., Motion Cyclification by Time x Frequency Warping. In *Proceedings of SIBGRAPI'99, XII Brazilian Symposium of Computer Graphics and Image Processing* (October 1999), pp. 49–58, IEEE Press.

[14] SUTTON, RICH AND BARTO, ANDREW *Reinforcement Learning: an Introduction*. The MIT Press, Cambridge, MA, 1998.