DESCRIPTION-DRIVEN GENERATION OF 3D HUMANOIDS WITHIN AUTHORING 744

V. Fernández-Carbajales*, J.M. Martínez* and F. Morán♣

Grupo de Tratamiento de Imágenes

*Escuela Politécnica Superior, Universidad Autónoma de Madrid,
Av. Francisco Tomás y Valiente, 11, Ciudad Universitaria de Cantoblanco,
Crta. Colmenar Viejo, km. 15, E-28049 Madrid, SPAIN
{Victor.Fernandez,JoseM.Martinez}@uam.es
http://www-gti.ii.uam.es/

♣E.T.S. Ing. Telecomunicación, Universidad Politécnica de Madrid,
Ciudad Universitaria s/n
E-28040 Madrid, SPAIN
Francisco.Moran@gti.ssr.upm.es
http://www.gti.ssr.upm.es/

Keywords: Graphics, Descript Metadata and 3D content.

Abstract

This paper presents a new trend within the Authoring744 research initiative: the creation of 3D humanoids. Authoring744 proposes the creation of content using descriptions that drive content synthesis. Here we present the first steps towards the generation of 3D humanoids and their future animation within the Authoring744 framework, which proposes to use the MPEG-7 and MPEG-4 standards for, respectively, semantically describing and representing content. One of the main objectives is the creation of more intuitive and easier to use 3D content authoring tools.

1 Introduction

Currently, the creation of 3D contents is achieved with not very intuitive programs and requires that the 3D content author follow long and costly learning curves. In addition, usually these programs do not use standard representation formats, which would make it possible to interchange authored content among them, but use instead their own format for optimizing 3D content handling.

Since its early development phases, MPEG-4 [1] has addressed, among other more well known topics, the standardization of (2D, 3D and 4D) graphics representation formats. Therefore, MPEG-4 is a good candidate, at least from a technical point of view, for overcoming the limitation in the interoperability among different graphics products: authoring tools, story-editors, players, etc. Regarding authoring, it seems more natural to describe what we want than to use the common 3D authoring tools. The Autoring744 research initiative envisioned [2] the use of high-level descriptions in order to ease the creation of multimedia content. First results [3] showed that the vision could reach promising results, although those results were constrained to editing multimedia scenes composed by different media arranged in a spatio-temporal synchronized scenario.

Authoring744 proposes the use of MPEG-4 [1] for content representation and MPEG-7 [4] for content description (both structurally and semantically), promoting the use of standards instead of proprietary formats (although adaptation or migration to them should be relatively easy).

This paper introduces the creation of 3D humanoids within the Authoring744 framework [3].

2 State of art

Currently, the authoring of 3D contents is achieved with programs that are very complex for the average users, with long curves of understanding and learning. Maya and 3ds max are among the most popular and powerful programs, and currently they are at least the starting point for the creation of 3D animated graphics.

Additionally to the standard interface for 3D content authoring, these programs usually provide different plug-ins for added functionalities: creation of humanoids, architectural models, particle systems, or clothes; texturing of eyes and skin; and so on. The problem of interchanging designs between programs creates difficulties in the collaborative authoring of complex 3D scenes by different work groups using different tools: e.g., a design created in Maya 6.0 is not portable to 3ds max 7.0. This inconvenient could be solved easily with the use of a standard representation format, namely MPEG-4 AFX [5].
Regarding the complexity of use, more intuitive programs could be designed based on the creation driven by descrip-

tions of the content we want to generate, for example, in the case of humanoids authoring the metaphor for the user interfaces will be the "robot portrait" (identikit picture).

There are plenty of standards (both de jure and de facto) for the creation of 3D content. In the first place, we have OpenGL, DirectX, etc.: standards which have an Application Programming Interface (API) for writing applications that create and manage 3D content via "methods" of that API. In order to run such programs, only the (appropriate version of the) corresponding library or drivers should be installed in the rendering terminal. The MPEG-4 [1] standard (like VRML97 [8]) has another philosophy, as it is based on standardizing the representation of the content, not the way for creating it. Therefore, the only requirement for interoperability is that the authoring tools and the players are compliant with the specified MPEG-4 representation format.

Another way for authoring 3D animated content is the use of a script describing the scene, both structurally and semantically, to create (this was part of the Authoring744 vision [2]). In the script file, the humanoids, the dialogs, the foreground of the scene, the soundtrack, etc. can be described and the system may consult content databases for searching or may synthesize the contents required for the scene. There are different systems proposing the script approach (although we have found none where the creation of content is part of the system). For example, in the EMM system [6] the primitive objects and actions are pre-stored in the system databases and knowledge system, and the script allows to combine them in order to create animated movies with their "base" objects and events. Authoring744 [2] objective targets both the creation of base objects and the scripting based on them. Currently, we are focusing on the creation of 3D content.

MPEG-4 was originally based on VRML97 [8], but has added huge improvements to it. In what concerns media synchronization issues (tackled in MPEG-4's Part 1, "Systems"), the biggest advantage is probably that MPEG-4 supports streaming while VRML97 does not. But also, and more importantly in the context of this paper, MPEG-4 has greatly extended the 3D graphics assets of VRML97. In its two first versions, MPEG-4's Part 2, "Visual", already featured the FBA (Face and Body Animation) toolset for the efficiently compressed animation of humanoids. More recently, a whole new and independent Part of MPEG-4, Part 16, "AFX (Animation Framework eXtension)" [5], has been devoted to 3D graphics, and it contains specific tools for the animation of completely generic virtual characters, based on the BBA (Bone-Based Animation) paradigm.

FBA defines control points for animating the humanoid's face and body, providing knobs on the face for expressing emotions by moving all its important parts (eyes, eyebrows, lips, ears, etc.), and on the body joints (articulations). The animation of humanoids with the BBA tools is based on the creation of a humanoid with a skeleton, i.e., a group of bones with their associated muscles and skin. Each bone has its own axis system and is then integrated in the common skeleton axis. With this integration, also kinematics constraints are imposed. All bones can suffer rotation, scaling and translation, allowing the deformation and animation of the "base" humanoid as we wish (the modifications propagate to the skin associated to the bones). The BBA specification is more powerful than FBA because BBA offers more usage functionalities for generic virtual characters, and because the control points of FBA are not as useful as the bones of BBA for the generation and animation of humanoids. Besides, BBA improves the quality of the specified graphics, as the system of muscles coupled to the bones generates more realistic movements.

3 Objectives

One of the main objectives of the Authoring744 initiative [2] is the creation of authoring tools which are more intuitive for the average user. In the scope of this paper, we are interested in the creation of 3D humanoids via high-level descriptions, in a similar way "robot portraits" are generated, that is, following a high-level semantic description and modification of it. The final aim is to ease the creation of this type of 3D content to the common user (for example, for customizing the user's avatar in a computer graphics game). Afterwards, the integration of their animation within this intuitive authoring framework will be provided.

Regarding the creation of 3D content, there is another objective regarding the independence of the created content (or its authored description) from the rendering environment, that is, the terminal and the network delivering the description (if the synthesis is done at the terminal). In any moment, the visualization of the contents must adapt to the conditions of the environment, looking for the best quality of this contents, but taking into account parameters as monitoring capacity, rendering capacity, processing capacity, etc. of the terminal: the described content does not change (in its "being" or structure) depending on where it is played, but should be adapted to the terminal and network characteristics at each moment.

The 3D humanoids descriptions will follow the MPEG-7 [4] framework, and we aim to contribute the extensions for supporting 3D content description to the MPEG-7 standardization process (future MPEG-7 amendments). Besides this, contributions to BBA are also expected (in fact, we have already contributed one upgrade to the BBA module MPEG-4's Part 5, "Reference software").

4 From deformation description to content modification

The idea for creating the humanoids via the "robot portrait" metaphor is based on the possibilities offered by BBA for modifying a base humanoid via BBA commands. The expected functionalities of the "robot portrait" application will be presented by example over Samuel, the gargoyle shown in **Figure 1** in its original form.

Figure 1: Original Model

Figure 2: Left Leg 10% Longer

Figure 3: Chopped Off

Figure 4: Head Twice As Big

The first example is the lengthening of Samuel's left leg by a 10%. For performing this deformation, all bones of Samuel's leg must be scaled by 1,1 times in the *y* axis (which corresponds to the longitudinal direction of this particular bone), but not in the *x* or *z* axes, to avoid a simultaneous widening of the leg. The result of this transformation is shown in **Figure 2**. The second example is doubling the size of Samuel's head. For performing this deformation, we need to scale the bone of head by 2 its three axes (*x*, *y*, and *z*). The result of this transformation is shown in **Figure 4**. The third example is the amputation of the Samuel's left arm. For this, we scale Samuel's left arm to zero in all of its coordinates. The result of this transformation is shown in **Figure 3**.

All these examples of modifications of an MPEG-4 humanoid show that "high-level" descriptions (e.g., longer leg, bigger head, no left arm) can be written for deforming (and animating) a base humanoid, as result of the user's interaction, which consists in providing parameterized descriptions within an application. These descriptions (to be specified within the MPEG-7 framework) will be mapped to specific modifications in BBA to a base humanoid. We also plan to provide descriptions to create the base humanoid from scratch.

5 Software architecture

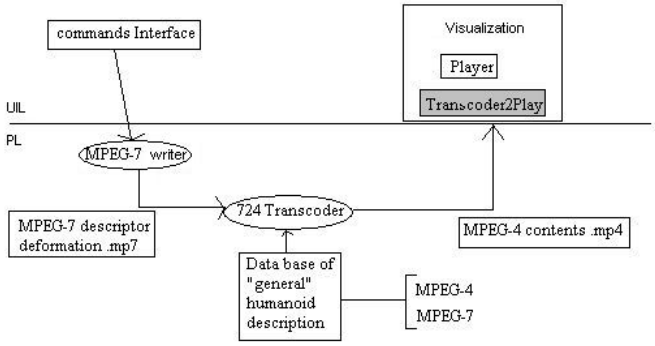

Figure 5: Software architecture

Following the Authoring744 high-level architecture [3], the software architecture for the authoring of humanoids via descriptions can be divided in two layers, as shown the **Figure 5:** UIL (User Interface Layer), and the PL (Processing Layer).

The UIL is in charge of the interactions with the user of the application. This layer controls the inputs/outputs of the system from/to the user. The inputs that the UIL layer handles are two: the selection (or creation in a near future) of the base humanoid and the deformation commands. The UIL captures from the user the base humanoid over which the deformations will have effect. The deformation commands are to be selected within a set of instructions for modifying the characteristics of the base humanoid: height, weight, musculature, etc. These commands can be associated in scripts which can be made persistent for future reuse. The UIL manages two outputs: the visualization of the humanoid that the user is modelling and the corresponding MPEG-7 [4] and MPEG-4 [1] schemas. Regarding visualization, each new command up-

dates the rendering of the humanoid (and there is the possibility of redo and reset), after the MPEG-7 description and MPEG-4 nodes have been updated via the processing layer. The schema views (MPEG-7 description and MPEG-4 nodes) allow the advanced user to modify the humanoid directly. Any change of the schemas also triggers a revisualization of the humanoid. The MPEG-4 content (i.e., 3D humanoids described thanks to BBA) could be transcoded to feed an appropriate player for its rendering (currently there is no need of this module, and therefore it is shaded in grey in the diagram).

The PL is in charge of the parsing/translation of the commands launched by the user to the high-level description in MPEG-7 and the translation of the MPEG-7 descriptions into MPEG-4. Currently the approach is to use a base humanoid in MPEG-4 and to modify it via MPEG-7 descriptions, although the target is to have a whole MPEG-7 description soon allowing full creation in MPEG-7 and not only deformation. After each change in the MPEG-7 description, the 724Transcoder module translates the new description of the humanoid, and afterwards the PL triggers the UIL player for updating the rendering.

Currently, we have checked the power of using MPEG-4's BBA [5] for the creation, deformation, and animation of virtual humanoids. We have developed an application that implements some parts of the system presented in Section 5, namely part of the 724Transcoder. The current development of this module covers the application of deformations to a base humanoid using BBA. Starting from a base humanoid, other "derived" humanoids may be created by selecting a set of deformations to be applied to the different accessible bones. Both the base and modified humanoids are visualized, as can be seen in **Figure 6**. The applied transformations are saved and can be applied to other base humanoids (the deformations applying only if the same bone exist).

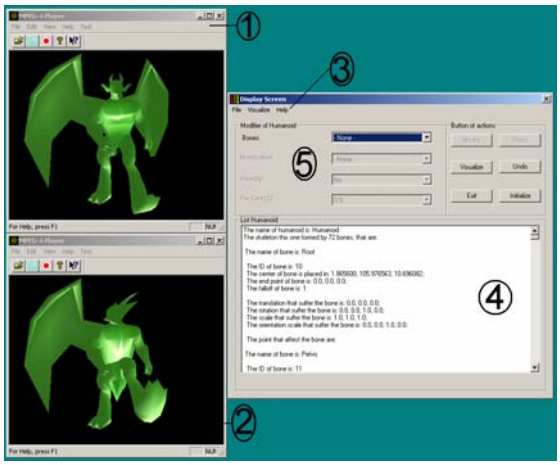

Figure 6: Screen Shot Application

The base humanoid may be developed with any 3D modelling tool that can output content MPEG-4-compliant files/bitstreams, and is used as the primary model for the "derived" humanoids, which are obtained by applying the chosen transformations from the ones available in the GUI. A transformation is any modification that is performed on the humanoid by means of a high level command, like "get bigger", "get shorter", etc., and that is applied to individual bones or to groups of them, depending of the level of abstraction (currently only the complete skeleton level is implemented).

In the Screen Shot Application in **Figure 6**, we can see the different main windows:

- Number 1 points at the window visualizing the base humanoid.
- Number 2 indicates the window visualizing the "derived" humanoid.
- Number 3 points at the toolbar giving access to the different menus: load, save, visualize, and transform humanoids.
- Number 4 is the MPEG-4 file code that defines the "derived" humanoid, after applying the transformations to the base one.
- Number 5 indicates the most important part of the application: the series of combo boxes and buttons allowing the user to choose and apply a deformation on a bone or group of bones of the humanoid.

Currently the application allows to apply up to 25 different transformations (listed in **Table 1**) on any bone of the humanoid – actually in any of the 88 bones that can be handled by 3ds max 7.0 and INT's plug-in for exporting these bones into the MPEG-4 file format. Any transformation can be applied to a selected bone only, or propagated to all its "inheritors" inside the MPEG-4 structure of the humanoid.

Note that the coordinate system of each bone is local and therefore leads to what are known in the 3D Graphics context as MCs (Modelling Coordinates), as opposed to WCs (World Coordinates). It is chosen by the creator of the bone [5], and so the axes mentioned in **Table 1** are completely dependent on that choice.

Our tests show that, with a small amount of transformations, we can adapt in a fair way the humanoid to what we want to "see", like in a "robot portrait".

Table 1 Modifications handled by the application

Number	Transformation	Description
1	Slim	Shrink the bone by scaling it in the *x* axis.
2	Fatten	Stretch the bone by scaling it in the *x* axis.
3	Shorten	Shrink the bone in the *y* axis.
4	Lengthen	Stretch the bone in the *y* axis.
5	Smooth	Shrink the bone in the *z* axis.
6	Get_Fat	Stretch the bone in the *z* axis.
7	Make_Slender	Shrink the bone in the *x*

		axis, and stretch it in the *y* axis.
8	Make_Dwarf	Shrink the bone in the *y* axis, and stretch it in the *x* and *z* axes.
9	Bigger	Stretch the bone in the *x*, *y* and *z* axes.
10	Reduce	Shrink the bone in the *x*, *y* and *z* axes.
11	Amputate	Shrink the bone to zero, which virtually eliminates it.
12	Separate_Origin	Separate the bone from the world coordinates origin.
13	Mover_Close_Origin	Brings the bone over to the origin of coordinates of the world.
14	Move_Right	Moves the bone towards the positive *x* axis.
15	Move_Left	Moves the bone towards the negative *x* axis.
16	Take_Up	Moves the bone towards the positive *y* axis.
17	Take_Down	Moves the bone towards the negative *y* axis.
18	Bring_Over	Moves the bone towards the positive *z* axis.
19	Separate	Moves the bone towards the negative *z* axis.
20	Twist_Right	Turn clock-wise around the positive *y* axis.
21	Twist_Left	Turn counter-clock-wise around the positive *y* axis.
22	Turn_Right	Turn clock-wise around the positive *z* axis.
23	Turn_Left	Turn counter-clock-wise around the positive *z* axis.
24	Raise	Turn clock-wise around the positive *x* axis.
25	Descend	Turn counter-clock-wise around the positive *x* axis.

7 Current and future work

Currently we are creating the higher abstraction levels of the humanoid description, allowing to describe the humanoid in terms of groups of bones composing high level units, like torso, head, neck, arm, etc. We are working in different levels of abstractions, e.g., hand versus palm and fingers. Also, the transformations will be mapped to a more natural set of expressions.

These descriptions, both for humanoids and transformations will be represented using the MPEG-7 [4] framework, our aim being to propose new description tools. More long-term work will include the animation of the humanoid within the proposed framework.

8 Conclusion

The extension of Authoring744 from multimedia scenarios edition to the creation of 3D content has started, targeted for instance at the creation and animation of virtual humanoids. Currently preliminary results show that the approach is also valid in this scenario and that the authoring of avatars may be easy for any user, therefore paving the ground for any other applications of convenient personalization of avatars.

Besides the easy of creation of 3D content, the associated MPEG-7 description that we foresee will allow to transmit it much more compactly than through the equivalent MPEG-4 representation, in general more complex and bigger. Of course this approach implies that the terminal should be able to run not only an MPEG-4 player, but also a 724Transcoder. Adaptability will be another major functionality regarding the description and its translation to MPEG-4.

Acknowledgements

This work has been partially supported by the 6th Framework Programme of the European Commission, within its research project FP6-IST-1-507926: On-Line GAming (OLGA). The authors wish to thank Marius Preda and Octavian Folea of INT (Institut National des Télécommunications) for their support regarding AFX, and Fabrice Léte of Larian Studios, who allowed the use of the Samuel character.

This work is also supported by the Ministerio de Ciencia y Tecnología of the Spanish Government under project TIN2004-07860 (MEDUSA).

References

[1] R. Koenen (ed.): “MPEG-4 Overview v21”, ISO/MPEG doc. N4668, 2002.

[2] J.M. Martínez and F. Morán: “From Descriptions To Content: Inverting The Sense”, Proc. Intl. Conf. Media Futures, 111-114, 2001.

[3] J.M. Martínez and F. Morán: “Authoring 744: Writing Descriptions to Create Content”, IEEE Multimedia October/November, 94-98, 2003.

[4] J.M. Martínez (ed.): “Overview of the MPEG-7 standard v9.0”, ISO/MPEG doc. N5525, 2003.

[5] MPEG (Moving Picture Experts Group), formally ISO/IEC JTC1/SC29/WG11: “ISO/IEC 14496-16” (a.k.a. “MPEG-4 Part 16: Animation Framework eXtension”), ISO/IEC standard, 2003.

[6] H. Seo, F. Cordier and N. Magnenat-Thalmann: “'Synthesizing Animatable Body Models with Parameterized Shape Modifications'”, ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 120-125, July 2003.

[7] J. Shen, T. Aoki, H. Yasuda and S. Miyazaki: “Emovie Creation By Rule-Based Reasoning From The Director’s Viewpoint – E-movie: Computer Animation & Real Images”, Knowledge-Based Media Analysis for Self-Adaptive and Agile Multimedia Technology, 119-126, 2004.

[8] VRML (Virtual Reality Modelling Language) Consortium Inc. (now Web3D Consortium Inc.) and ISO/IEC JTC1/SC24: “ISO/IEC 14772 1:1997” (a.k.a. “VRML97”), ISO/IEC standard, December 1997.