Virtual Modelling

Szilárd Kiss, Anton Nijholt and Job Zwiers

Department of Computer Science, University of Twente, The Netherlands
{szilard | anijholt | zwiers}@cs.utwente.nl

Abstract

We concentrate our efforts on building virtual modelling environments where the content creator uses controls (widgets) as an interactive adjustment modality for the properties of the edited objects. Besides the advantage of being an on-line modelling approach (visualised just like any other on-line VRML content), our approach provides also instant visualisation and an intuitive, graphical editing modality.
Although visual modelling environments are more powerful in content creation than a text editor, our aim is to include certain domain knowledge to extend their capabilities. Our most recent system is a 3D character modeller capable of handling H-anim or other types of hierarchies (based also on H-anim components) and the geometry attached to them. The modelling is based on the properties of regular grid geometries, on H-anim hierarchies and where applicable, on 3D character symmetry.
The novelty of our approach consists of 1) natural graphics components; 2) integrating the interface elements into the virtual environment itself; 3) rule based modelling of 3D characters; 4) on-line modelling.
Keywords: Visual modelling, 3D characters, virtual environments, VRML, interaction.

1. Introduction

The paper deals with modelling concepts for virtual reality. We try a method of bringing visual, real-time modelling possibilities to programmers and occasional content creators who know the basics of the VRML language, but do not necessarily know about graphics packages and their complex operations. We achieve that by visualising the editable properties of VRML components. We built modelling prototypes aimed at geometries, and for other, higher level VRML constructs, such as bone hierarchies.

If we provide a visual modelling possibility for low level VRML graphics, like geometry coordinates, texture coordinates, etc., we reduce the overhead needed for switching between text-mode editing and viewing of the results. In that situation, the simple positioning of a chair next to a table may require several window-switching and scene reloading operations, while some drag operations using visual controls would position intuitively the chair in the desired position.

The novelty of the system consists of taking the modelling interface into the virtual environment (VE). The result is an instant preview method, as well as the online modelling possibility inside the VRML plugin. We also take in account properties of the edited entities to provide rules which ease the construction of 3D characters.

2. Background

VE content can be handled using text editors or graphics modellers. We like the visualising properties of the latter, but we are concerned about their complexity. We would also like to take advantage of the user's knowledge about VRML, which is constrained mainly in favour of text editors.

Simple text editors are the basic tools for creating VE content. They are suited for users who know the file format, but the result has to be previewed separately. For fine-grained geometrical and positional control, this can turn tedious. Tools like VRMLPad[11] or X3D Edit[5] provide additional assistance, but visualising still needs another application.

Graphics modellers that provide VRML export generate huge files, since they work with complex, high-precision data. They also require a steep learning curve. Graphics modellers aimed at VRML also tend to be complex. CosmoWorlds for instance does offer a visual approach to modelling, but is overwhelming to occasional users.

Our attention was caught also by VRML constructs (widgets) that bring interaction possibilities into VEs. Sliders for colour picking, illumination control, VCR-style controls, custom navigation, etc.[13] provide the possibility to alter the environment. In this light, a VE with saving routines could be used for modelling purposes.

The modelled characters have graphical (skin geometry) and structural (hierarchy) components. We have seen graphics components similar in construction[12], where regular grid geometries are used for easier calculations. Our case is different by providing conceptual groupings for the geometry gridpoints, and the tools to handle these as groups, resulting in a more powerful editing possibility. We chose the H-anim[6] specification as the base for our hierarchies, consisting of Joints and their Segments (geometries). We also allow the users to extend these hierarchies as wished.

The benefits of natural interaction and the integration of the modelling tools into the visualisation environment is recognised in the literature. Emphasising the importance of 3D widgets, Conner et al.[3] mentions advantages like reduced size, naturalness, visual control. Together with others[1, 10, 4], they advocate the view of integrating the modelling tools completely in the VE, providing consistency and even a higher level of immersion in modelling tasks.

3. Framework

Our goal is to provide modelling systems that are capable of handling geometry, structure and animation for 3D (virtual) characters in a visual, easy to use manner. For this purpose, we analysed the existing technical possibilities regarding visualisation and interaction.

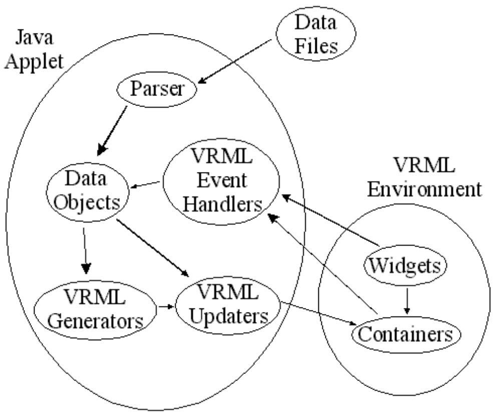

Figure 1: *Structure of the modelling system.*

If we look at the visualisation of VRML content, the obvious choice is a web browser with a compatible plugin, choice inherent from the medium of VRML, which is the world wide web. Building modelling systems is generally regarded as building the controls around the visualisation environment. As we can see from various examples, a VE can be built to influence its own content. We extend this concept to the more unusual components of the VRML content: coordinates, whole geometrical elements and hierarchy properties.

For visualising the underlying structure of graphical elements and hierarchies (the coordinates and Joints) we use small graphical elements (for now cubes), which we connect with sensors and make them selectable. Other types of selections also exist, aiming at coordinate collections or Segments. Widgets and buttons represent a minimal number of operations (we opt for few operations in favour of easy usage), inherent from VRML basic properties like coordinate position, scaling and number of children.

Our system becomes through the visualisation choice an on-line modelling system. It is an on-demand application for the simplicity (brevity) of VRML content in a visual form, providing intuitive handling with instant preview which shortens the preview action. With the right setup, it would be also possible to use it in a collaborative scenario.

The modelling systems take advantage of the Java programming language through the External Authoring Interface[2] (EAI) which makes possible the communication between an applet and a VRML plugin within a web browser. The different connections of the system are depicted in Figure 1.

4. Geometry modeller

First, we implemented a geometry modeller based on regular grid coordinates, but without automatic symmetry handling. As our goal involved the creation of human characters, we realized that an automatic symmetry handling approach would ease the work for a content creator. Our second prototype for geometry modelling contains the necessary routines for automatically handling the symmetric coordinates. An asymmetric and symmetric geometry model is shown in Figure 2.

The regular grid coordinates made possible the use of natural geometry primitives: the coordinate rings and columns that define the regular grid. To this, a number of natural geometry modelling operations can be added, with the premise that these preserve the regularity of the geometry.

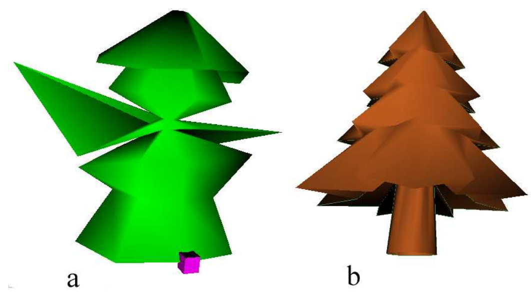

Figure 2: *a) Unsymmetric, b) Symmetric geometry model.*

The combined geometry modeller included in the 3D character modelling system provides the possibility to edit all the skin Segments that make up the structure of a whole 3D character defined by its hierarchy indifferently whether they are self-symmetric or not. The skin is visualised by selectable coordinates which are bound to the regular grid criteria. Due to this criteria, the selectable entities are single coordinates and collections of coordinates which can be either ring or column collections. In case of symmetry only a half grid of coordinates is selectable and the modelling operations are replicated on the other half model also.

We use for our systems custom data files. This gives the possibility for the use of a simple format which requires less parsing, and also provides a means of automatic coordinate handling. These files contain information like the geometry grid size and the actual coordinates.

Following our guidelines for integrated tools and simplicity, the geometry modelling interface is situated completely in the VE and the limited number of tools make it easier to learn it and use it. Figure 3 is a screen-shot of the system with a 3D character and a selected geometry component.

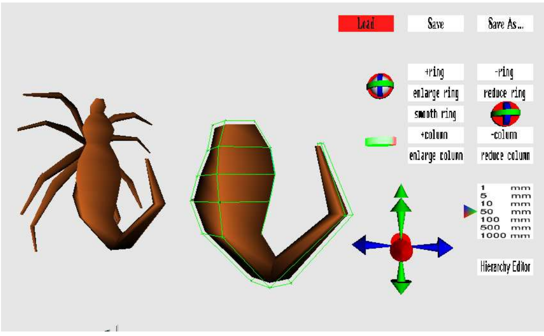

Figure 3: *Geometry modelling subsystem. From left to right: edited character, selected geometry component (tail), modelling tools. The modelling tools: file handling buttons (top), geometry handling buttons (middle), different widgets (round rotation widgets, scale widget are in the middle while displacement related widgets are on the bottom).*

Conversion to NURBS surfaces is a relatively straightforward conversion from simple coordinates to knot values considering the similarities between the regular grid criteria of our geometries and the NURBS knot vectors. This way, simple polygon geometries become parametric surfaces that can present enough surface curvature detail if needed. A modified version of our geometry modelling system is capable of modelling NURBS based on such a conversion.

5. Hierarchy modeller

The hierarchy modeller extends the possibilities of the geometry modeller by adding structure (hierarchy) manipulating possibilities. With this system, it is possible to build complex objects, made up from a number of graphics elements. The underlying, compositional structure for these complex objects is a hierarchy structure based on the H-anim specification, since our primary aim is to provide tools for the creation and animation of 3D characters to be used in VEs. H-anim defines a bone structure with 'Joints' and the geometry 'Segments' attached to them. We use these components, but we do not enforce the strict requirements for the humanoid Joints for the sake of extendibility: both H-anim and custom templates are available. Figure 4 depicts an H-anim compliant and a non-compliant hierarchy structure while Figure 5 shows their corresponding geometries.

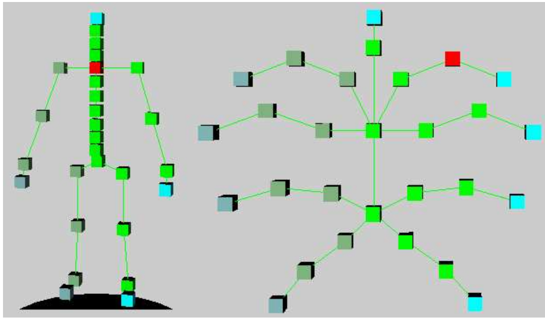

Figure 4: *H-anim compliant and non-compliant hierarchies.*

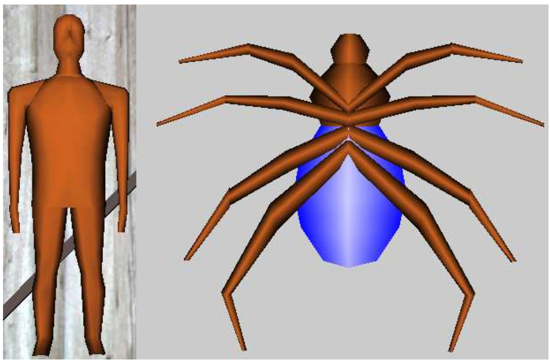

Figure 5: *Humanoid and non-humanoid characters.*

The Joint structure of H-anim based hierarchies is bound to properties like: each Joint has one parent Joint (except the root Joint), each Joint is a parent or terminal Joint and each Joint has one Segment child. We set hierarchy manipulation rules for working with the hierarchies, these being:

- Refining the hierarchy by inserting a Joint.
- Extending the hierarchy by adding a Joint at leaf level.
- Collapsing the hierarchy by removing a Joint.

The selection sensors in this case are pretty straightforward, since the Joints are not grouped like in the case of coordinates. The available manipulation options take in account the selection context based on the hierarchy and manipulation rules mentioned previously.

Symmetry in this case is two-fold. The geometrical symmetry was renamed to internal symmetry, and a new type of symmetry was introduced describing pairs of symmetric Joints and Segments, which we call external symmetry (e.g. human legs are externally symmetric in this classification). The modelling operations are designed to automatically resolve both types of symmetry constraints.

The hierarchy modelling system also takes advantage of the EAI connection possibility to resolve the interaction and the file handling requirements. The file format used is a proprietary, simple one to ensure the regular grid, hierarchy and easy parsing requirements. To this extent, one record of the hierarchy file specifies a Joint name (which is a file describing the positional, rotational and other properties of the Joint), its parent, its Segment child (the connecting geometry) and its symmetry property.

The interface, similar to the geometry modelling interface, is integrated into the environment and contains a number of tools that effectuate the operations defined by the hierarchy manipulation rules enumerated above and handle the Joint positions within the hierarchy. Figure 6 depicts the visualisation and usage options of the hierarchy modelling system, which can be easily mastered.

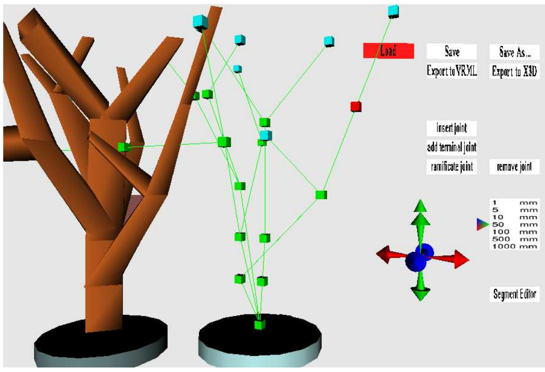

Figure 6: *Hierarchy modeller. From left to right: 3D character (a tree in this case), its underlying hierarchy, modelling tools. The tools are: file handling buttons (top), hierarchy complexity handling buttons (middle) and Joint displacement widgets (bottom).*

6. Conclusions and future work

We provide modelling tools for occasional users interested in creating 3D character models usable as avatars, agents, etc., tools that are easy to use, simple and at the same time natural in their interface and editing options. H-anim and other types of hierarchies can be created in a visual, direct manipulation modelling environment with embedded controls. A serious drawback is the use of regular grid geometries, but this is also the source for the simplicity and naturalness regarding the geometry primitives and modelling options. The format used provides the on-line modelling possibility.

We are currently working on extending our 3D character modeller with animation handling. A literature survey[9] showed us different possibilities of animation systems, from which the most promising is an approach based on captured data. This is due to the advancement of capture technologies as well as the different possibilities for analysing animations, extracting characteristics and generating combined or new animations. We are also working on adding a texture handling component to the system.

References

1. E. A. Bier. Skitters and Jacks: Interactive 3D positioning tools. In *ACM Workshop on Interactive 3D Graphics*, pages 183–196, 1986. ACM Press, New York. 2
2. Chris Marrin - Silicon Graphics, Inc. *External Authoring Interface Reference* http://www.graphcomp.com/info/specs/eai.html 2
3. D. B. Conner, S. S. Snibbe, K. P. Herndon, D. C. Robbins, R. C. Zeleznik, and A. van Dam. Three-dimensional widgets. In *Computer Graphics*, volume 25(2), pages 183–188, 1992. ACM Press. 2
4. J. Döllner and K. Hinrichs. Interactive, Animated 3D Widgets. In *Computer Graphics International*, 1998. 2
5. Don Brutzman. X3D Edit. www.web3d.org/TaskGroups/x3d/translation/README.X3D-Edit.html. 1
6. Humanoid Animation Working Group of the Web3D Consortium. H-Anim: Specification for a Standard VRML Humanoid, version 1.1, 1999. http://www.h-anim.org/Specifications/H-Anim1.1/ 2
7. Sz. Kiss. Web Based VRML Modeling. In *Information Visualisation*, pages 612–617, 2001. IEEE Press.
8. Sz. Kiss. 3D Character Modeling in VR. In *Information Visualisation*, pages 541–548, 2002. IEEE Press.
9. Sz. Kiss. Computer Animation for Articulated 3D Characters. University of Twente, Technical Report TR-CTIT-02-45, ISSN 1381-3625, October 2002. 4
10. L. Matejic. LOG: Building 3D User Interface Widgets by Demonstration. Technical Reports, Brown University, CS-93-22, May 1993. Technical Report. 2
11. Parallel Graphics, Inc. VrmlPad Text Editor. www.parallelgraphics.com/products/vrmlpad/. 1
12. P. Volino and N. Magnenat-Thalmann. *Virtual Clothing: Theory and Practice*. Springer, 2000. 2
13. Web3D Consortium. PROTO Repository. sdk.web3d.org/summer2002disk2/content/PROTORepository/. 1