Behavior Authoring for VRML Applications in Industry

R. Dörner, C. Elcacho and V. Luckas

Dept. Animation and Image Communication, Fraunhofer Institute for Computer Graphics, Darmstadt, Germany

Abstract

As VRML offers behavior definition on different levels (animation behavior, interaction behavior, hypermedia behavior, application behavior) an authoring tool for VRML content creation has to comprise a dedicated behavior editor. An editor for this purpose has been conceived based upon object-oriented concepts. The concept stresses reusability and support for non-expert users. Reusable elements with object-specific behavior are used Clipart-like to author the scene easily. The editor itself offers a library of generally applicable behaviors, too. This enables the author to use a wide range of pre-defined behavior. The editor keeps track of the authors' actions and out of this information a VRML file is created automatically. A first implementation of this editor focuses on special elements and behaviors needed in industrial applications.

1. Introduction

VRML (Virtual Reality Modeling Language) is the new ISO/IEC standard for exchanging 3D models, environments and applications especially in a WWW-based context [1]. As opposed to traditional 3D data exchange formats VRML provides new forms of behavior and integrates behavior facilities such as 3D animation, interaction with the models and the scene, multimedia and hypermedia contexts, allows the use of annotations inside a scene and provides an interface for programming logical connections of a scene. The behavior components and modeling facilities of VRML go far beyond the functionality provided by traditional 3D animation systems. This implies the need for new, powerful behavior authoring tools for VRML applications and environments.

2. Concepts

In our approach we suggest to use an object oriented concept, where scene elements comprises a geometrical as well as a behavioral description. The authoring tool provides special behavior components in a behavior component library, which can be associated with scene elements.

The behavior descriptions are stored together with the VRML scene elements using Java. Technically VRML97 supports the Java ScriptNode interface, which allows the Java programming of a VRML scene. We have used this interface to specify behavior components that can be associated to VRML scene elements, i.e. geometry.

In order to define behavior in a hypermedia context the Java EAI (External Authoring Interface) has been suggested as an extension to the VRML97 standard [2] by Chris Marrin. We have used the Java EAI to specify behavior components that can be associated to VRML scene elements in order to provide hypermedia and multimedia integration.

Authoring is a three step process: The first step being the definition and generation of the VRML scene elements and the second step being the association of the behavior attributes to the VRML scene elements. The application elements and the hypermedia /multimedia integration elements are using a well defined interface for communication with the VRML environment. This well defined interface allows to reuse the elements in different applications. The third step in application authoring consists in assorting the application elements to form a complex scene or environment.

3. Behavior Editor System

Based upon the concepts presented in the last chapter a behavior editor has been conceived and partly implemented. The editor allows the author to search for appropriate elements and to insert them into the scene. Once an element is selected in a scene, its pre-defined behavior is presented to the author. For example, if the author inserts the „forklift truck" element into the scene

its object-specific methods like „move_forward“ or „fork_up“ are displayed. The author can select one and specify the parameters needed. For instance, if the author wants to use the „move_forward“ method he may choose to specify two points in the scene, a start point of the movement and an endpoint respectively. Then, the author has to decide about the conditions the behavior he has specified is triggered. In our example the author may specify that the forklift truck moves forward at a specific time (animation behavior) or after the viewer of the scene clicks on a specific part of the geometry like the throttle control (interaction behavior). The „move_forward“ method itself takes care that the behavior is animated appropriately in the resulting VRML scene, i.e. the wheels are rotating and a motor sound is played.

Apart from these object specific methods the editor offers a library with certain behaviors, too. For example, the editor offers annotation behavior. This means that an element may be annotated with a text supplying additional information to the element. The author can accomplish this by selecting the element to be annotated, providing a text string and choosing the annotation form. The editor offers three forms: a billboard that is placed next to the annotated element, a billboard that is only displayed when the user is near the element or an HTML page that is presented in the browser (in this case the text string needs to be an URL). Most behaviors the editor offers are related to more than one element, for instance snap behavior.

In the end of the authoring session the VRML file is generated, while traversing the data structure in which the behavior specifications are stored. In this process the VRML file is generated. The elements needed are inserted into the scene as external prototypes and the prototype definitions as well as the Java classes are added to the VRML scene. Thus, the author need not be familiar with VRML and its concepts such as routings or external prototypes, since the editor creates the VRML scene description automatically.

4. Applications

We have applied the authoring concepts to applications in the industrial sector, such as plant manuals for production planning and VRML based product information systems. Other application areas in industry are learning and training, visualization of simulation results and marketing.

The behavior attributes used in the context of learning and training applications are snap-behavior, reference-point behavior, explosion animation and assembly as well as disassembly behavior attributes. Snap behavior is used to define a relation between two or more scene elements, facilitating the action of interactively putting together the elements. Reference point behavior allows to specify two or more reference points on a scene element such as e.g. the top of a fork of forklift truck.

Figure 1: *3D Exploded View of a Compressor: Dive-In Point to a Product Based Plant Manual.*

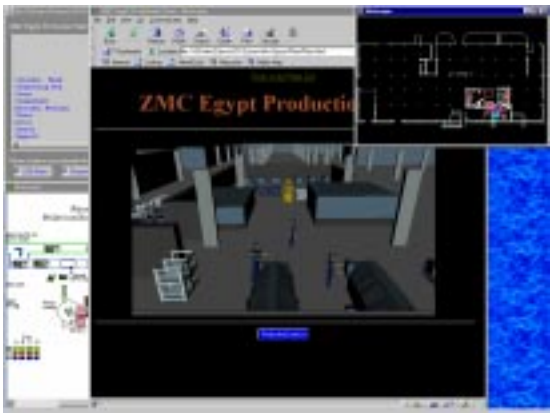

Figure 2: *VRML Based Plant Manual: View of the Drilling Machines Area*

They allow to explicitly define an animation, without knowledge of the exact coordinates. The explosion, assembly and disassembly animation attributes provide a scene element that is composed of more than one geometrical part, such as a product model of a motor, which consists of parts such as a crankcase, crankshaft, piston, etc., with the behavior to display an animation leading to a 3D exploded view of its parts, an assembly or a disassembly animation.

The explosion animation attributes have also been used in a product information system, such as the product based dive-in point to a VRML-based factory information system, depicted in figure 1.

Moreover a context attribute and a list-interaction attribute have been used for VRML-based hypermedia applications, such as the plant manual depicted in figure 2. An other view of an animation attribute added industrial scene is shown in figure 3.
In the future we will continue adding behaviors to our editor and will use it for evaluation purposes in projects with industry partners. Moreover, we are developing a supplemental editor that allows the authoring of element behavior itself.

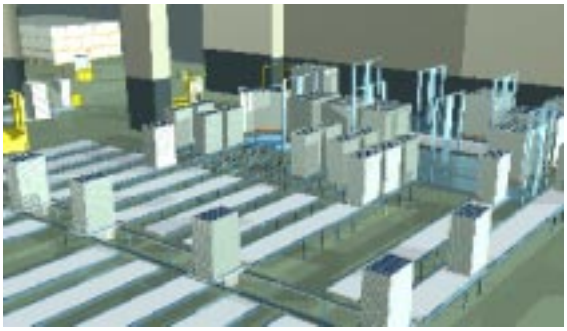

Figure 3: *Example of a Simulation Visualization in the Area of Production and Logistics.*

References

1. *The Virtual Reality Modeling Language, VRML97 Specification*, ISO/IEC 14772-1:1997, http://www.vrml.org/Specifications/VRML97/.

2. Marrin, C.: *The External Authoring Interface Proposal for a VRML2.0, Informative Annex*, November 21, 1997, http://www.marrin.com/vrml/eaiwg/ExternalInterface.html.