ITEA

Information Technology for European Advancement

Guidelines for 3D representations

Deliverable 5

Public version

User Centred Intelligence:
BEYOND (the GUI)

Edited by Marc Proesmans, Jo Dotremont, Jari Karvonen & Mikko Kerttula
Contributors in addition to previous: Marc Pollefeys, Fabian Ernst, Harri Kyllönen
Version: 22 December 2000

This document is part of the work of the EUREKA 2023 - ITEA 99002 project BEYOND.
Copyright © 1999-2001 BEYOND consortium.

Published via the Beyond Website:
www.extra.research.philips.com/euprojects/beyond

No part of this publication may be reproduced, stored or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without written permission of the BEYOND consortium.

Requests for publications can be send to: al_publicrequest@natlab.research.philips.com.

Table of Contents

1. Purpose and structure of this document

This document is the Deliverable 5 of BEYOND project and reports on the 3D representations that are needed for use in a simulation environment.

The deliverable has been done under Simulation workpackage (WP3) and the contributing partners and their expert areas are following:

- Eyetronics, Belgium, http://www.eyetronics.com: Eyetronics is specialized in 3D acquisition technologies. It offers solutions ranging from consumer-oriented products (e.g. low-cost sculptures of people), to high-end special effects for the movie industry. Its goal is to design cost-effective 3D acquisition systems that use off-the-shelf components.
- CCC/Cybelius Software Oy, Finland, http://www.cybelius.com: The partner has a special interest in web-based functional virtual prototypes, that are actually interactive, functional and photorealistic 3D computer models of the products.
- K.U.Leuven, Belgium, http://www.esat.kuleuven.ac.be/psi: The center for Processing of Speech and Images (ESAT-PSI) of the K.U.Leuven has a strong expertise in the acquisition of photorealistic 3D models from images.
- Philips Research, DSP, the Netherlands, http://www.research.philips.com: The contribution of the Digital Signal Processing group of Philips Research is focused on the acquisition of 3-D scenes and converting them into 3-D models.
- VTT Electronics, Finland, http://www.vtt.fi/ele: In this project VTT has a focus in virtual prototyping technology especially for future wireless telecommunication devices and environments.

Previous partners can be organised quite clearly in a number of distinct groups – say modelling groups – depending on the way they deal with 3D-modelling and the applications they are involved in.

- 3D acquisition: measurement / capture of 3D objects and/or environments
- Modelling: modelling of 3D objects, including interaction.
- Integration of 3D models and environments for use in simulation.

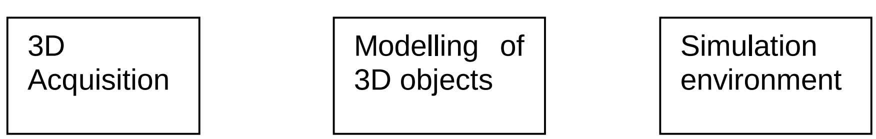

Figure 1: Representation of 3D models at different levels of the creation of 3D environments.

Obviously, each of these groups has its own way of representing 3D models and environments. The goal of this deliverable is to define a set of guidelines, that first of all describe the way in which 3D models or environments should be represented for the applications in mind, and secondly, at the same time, to establish a link between the different modelling groups.

This deliverable is therefore organised as follows:

First an overview is given of the different 3D representations that are typically generated and/or used within each of the above modelling groups by each of the partners separately. This can be considered to be a state-of-the-art on different types of 3D representations.

Secondly, a set of requirements will be defined that describe what kind of characteristics the 3D representation should possess in order to be usable within industrial simulation environments.

Finally, based on the requirements stated above, a set of guidelines will be defined to unify the different types of 3D representations into a more coherent model for simulation environments.

2. 3D representations

This paragraph gives an overview of the 3D representations used by each of the contributing partners. The purpose of this overview is to give a clear idea about the 3D representations that are currently used within different fields related to 3D modelling. This will give the reader a clear understanding of the requirements and guidelines that are put forward later on in this report.

This section is also subdivided in three parts, according to the "modelling groups" defined in section 1. As it will become clear, the strategies followed by the partners within each individual group will be quite similar.

2.1. 3D acquisition

Eyetronics

The native format of Eyetronics is related to the way the 3D structure of the models is acquired. By projecting structured light, in particular a square grid pattern, onto the object, the deformation of the grid as seen from a camera (from a different viewpoint) yields the 3D structure of the object. The ShapeSnatcher software allows converting the 2D grid pattern into a 3D patch, which is represented as polygonal mesh of quadrangles (due to the square grid), and can be exported in the most common 3D formats including VRML. Also texture images can be assigned to the 3D model for texture mapping.

Obviously, one single image is not sufficient to build complete 360 degrees models. The ShapeMatcher software allows importing different partial 3D patches (either by changing the camera viewpoints, or by rotating the object in the camera/projector set-up). After aligning and blending the different patches and their textures, one single surface topology can be generated. This is achieved by stitching together the different parts on the overlapping area. Depending on the integration mechanism used, the resulting 3D model is represented by a polygonal mesh of either triangles or quadrangles, which can be exported in the same formats as mentioned above, e.g. VRML.

K.U.Leuven

The native format of the K.U.Leuven method is related to the method of acquisition. This representation consists of the images themselves with the associated recovered depth maps (a distance value from the camera centre to the object surface along the ray associated with every pixel) and the pose/calibration for each of these images.

Starting from this representation several representations can be obtained. Typically, a polygonal mesh with associated texture is generated (in VRML format). However, other representations such as parametric surface models could also be constructed from the native representation or through an intermediate polygonal representation.

The native representation also directly supports other applications such as augmented reality or plenoptic/lightfield rendering (rendering new images by recombining the light rays found in a specific collection of images).

Philips

In the approach used by Philips, 3D-models are created from a set of images by a two step approach. In the first step, depth maps are created from two subsequent images using a Structure-from-Motion (SfM) algorithm. The second step, the so-called "over-time integration", combines the 2.5-D data (images with their associated depth maps) into a full 3D model.

For the geometry of the 3-D model, we distinguish between an internal representation and an external representation. The internal representation is the current best estimate for the 3D model, stored in such a way that it can be efficiently stored and easily be augmented when new data (images with associated depth maps) come in. This internal representation is hidden from the user. The external representation is the output given to the user. From the internal representation, an implicit surface is generated which is triangulated. The triangulated mesh forms the external representation, and is provided to the user as a VRML model. Currently, only the geometry of the VRML model is defined. Representation of the texture on this triangulated mesh is still a subject of ongoing research. The choice for representing the model as VRML is based on its standardised format and its widespread use for 3D-purposes.

Note that for the generation of 3D-models, no assumptions have to be made regarding the type or number of objects; there is no difference in generating a single 3D-object or multiple 3D objects.

As a preliminary result, the 2.5D-data can also be provided as pseudo 3D-models. Here again, VRML is used augmented with a global texture map, which is in this case a scaled version of the original image.

2.2. Modelling

CCC Cybelius

The Cybelius Software has technology to create functional 3D virtual models that simulate the behaviour of the actual physical product. The Java-based technology enables development of lightweight solutions, which can be accessed in the Internet environment. The small footprint technology allows a wide range of solutions:

- Interactive 3D interface
- Online configuration
- Shared Web presentations

The technology enables the 3D models to be connected to other relevant software through interactive 3D interface in order to enhance configuration capabilities and real time use of the functional virtual products.

The production of interactive 3D models is a twofold process. The geometry of the models is defined by using 3rd party modelling software (typically 3D Studio MAX). The selected modelling tool must be capable of exporting models to an Internet enabled format, or appropriate file format converters must be available. Currently, the Internet enabled formats supported by Cybelius are:

- Viewpoint
- Shout3D
- VRML

The implementation of the 3D functionality is rather strictly determined by the selected format. Therefore, the 3D model is seen as just a thin UI layer on top of the actual application logic, which is encapsulated in a way that allows changing the 3D format transparently. The user actions in the 3D model (UI) trigger must be able to communicate with the underlying application logic via following methods.

- Make HTTP requests
- Call JavaScript functions on the current HTML page

2.3. Simulation

VTT

VTT's VRP (Virtual Prototyping Environment)

The development of the user interface of future mobile communication and information devices is challenged by several development trends. First, these devices will provide complex functionality and applications based on their ever-growing embedded software. According to common predictions, the mobile information devices will become one of the most important application platforms in the near future. The user interfaces of these devices are rapidly adapting multimedia and multimodal features, but due to requirements set by mobility, the devices itself are becoming increasingly smaller. A small-sized device can not have a large display or convenient keypad. Instead of conventional mouse and keyboard these devices will use alternative input methods. In longer-term visions, mobile information devices will become truly ubiquitous computing devices that have fast, wireless access to Internet and that will provide user interfaces based on virtual and augmented reality techniques. These whole new kind of mobile user interfaces will require a lot of research and experimenting, which in turn, implies introduction of novel simulation tools and methods. To meet these challenges and to facilitate the development of future mobile user interfaces, we are developing a simulation framework that is based on the virtual reality prototyping technology developed at VTT Electronics. The technology utilises *smart virtual prototypes* [1] to simulate the various features of the target device. The key objective in the development of the simulation framework was to define an advanced platform for a diverse range of prototyping tools and applications. The objective is to support rapid development and testing of future mobile communication and information devices and their user interfaces. The framework utilises object-oriented and dynamic Java technologies to implement both the smart virtual prototypes and the integrated development environment. The applications of the framework will shorten developments cycles of mobile user interfaces and improve the quality of designs.

Smart virtual prototypes

Smart virtual prototypes are digital product models that enable realistic and interactive simulation of product's user interface and functionality. They are built using digital product components as building blocks. Digital product components are implemented using platform independent Internet technologies, such as Java. Smart virtual prototypes employ a layered architecture of three main layers: *UI simulation layer*, *logic simulation layer* and an intermediate *connection layer* that connects the two former layers. The UI simulation layer consists of a user interface simulation and rendering components. The logic simulation models the functionality and behaviour of the target device and its components. UI objects are based on bitmaps, Java Foundation Classes and Java Swing components. Alternatively the UI objects can be three-dimensional. In that case the UI objects are based on VRML97 prototype nodes. The connection layer is an abstract interface whose implementation depends on the applied UI components. The main task of the connection layer is to pass events between UI components and logic simulation components.

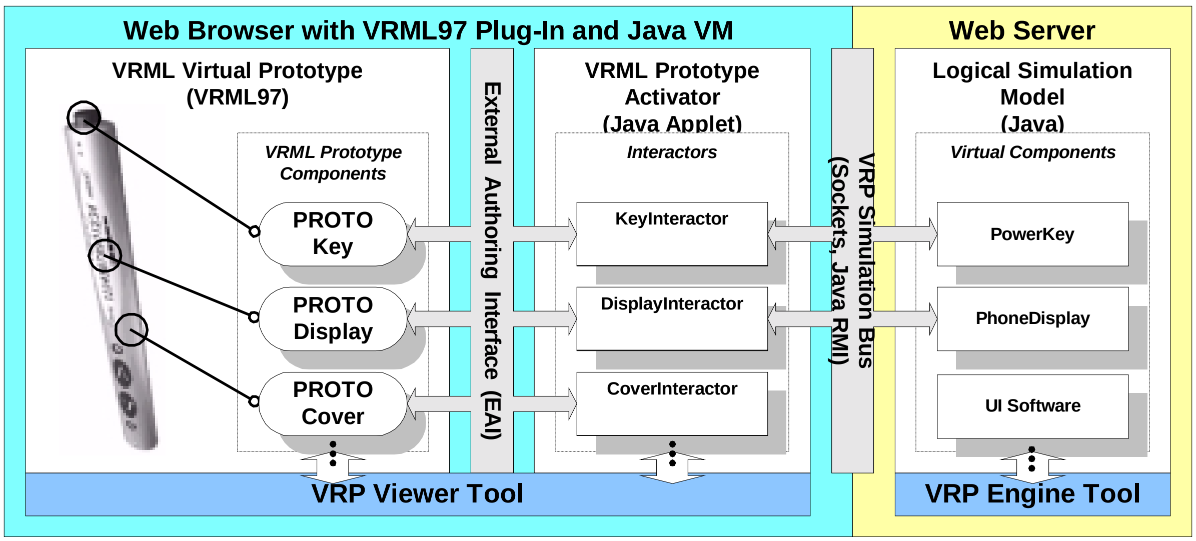

Figure 2. Layered architecture of a 3D virtual prototype.

Digital product components

One of the key ideas in the framework is to build the virtual prototypes using digital product components as building blocks. A digital product component aims at simulate its physical counterpart at an adequate degree of functionality and realism. The digital product components have the generic ternary architecture. This architecture reflects the three main layers in the architecture of the simulation framework. All the components of the target product are represented as a combination of one or more of the following modelling components: a UI component, an interactor component and a logic component. These modelling components are combined so as to communicate with each other. (Fig. 2)

3D representation

The 3D representation of the virtual prototype is based on VRML97 [2]. In VRML it is possible to create new objects called prototypes. Prototype nodes allow the reuse of any VRML node or a combination of nodes. Using prototypes it is possible to create reusable complex VRML structures.

A library of VRML prototype nodes has been created. The nodes in the library represent the most common part types in virtual products e.g. button, display, cover etc. These prototype nodes have an interface to communicate with the interactor components. Each instance of a prototype node is also named to identify it when sending and receiving events from the interactor components. Geometry and material properties can be attached to the prototype nodes. Some of the prototype nodes have animations defined in them to allow simple interactions (pushing a button triggers an animation sequence). Animation sequences and geometry are independent. It is possible to change the geometry without affecting the animation.

The VRML model of the virtual prototype is usually converted from a 3D-CAD model. Most 3D-CAD programs can export VRML. Some processing is always required before a model is feasible to be used as a virtual prototype. VTT has created a tool called VRP Creator [3] for facilitating the processing. VRP Creator can automatically parse a VRML model and attach geometries to appropriate prototype nodes. Only requirement is that a certain naming convention should be followed. This way we can easily create functional 3D models from CAD models.

3. Requirements

This paragraph sets out a number of requirements the 3D models and environments should satisfy in order to be applicable in an industrial simulation environment.

3.1. Real time interaction

Within a simulation environment, the user should be able to interact with that environment, be it pushing a button, getting status information, viewing the environments from different angles, etc. Whenever the user requests for a certain action to be performed, the reply from the simulation environment should be immediate, i.e. real time.

3.2. Exact representation

The virtual objects or environment within the simulation program, should be exact copies of the real object or environment. Of course, for some part of the simulation environment (e.g. background) a detailed representation may not be necessary. However, it should be possible that at least for those objects where accuracy and consistency with the real world is necessary, the corresponding 3D representation of these models allows to represent this kind of realism.

3.3. Perceptually convincing

The objects and the simulation environment should be perceptually convincing to the user of the simulation environment. Since the goal of the simulation is to show or design objects in real life, the virtual objects should look as real as possible in the simulation environment.

3.4. Show functionality / physical interaction

An object is not only characterised by its shape and outlook, but also by the way the user or designer can interact with the object. The simulation environment should therefore allow to show all the functionalities of the object (buttons, display, etc..) and provide the means to interact with the object (press buttons, …)

3.5. Reusability

The objects within the simulation environment should be reusable. Whenever a specific characteristic of an object needs to changed, it is necessary that the remaining characteristics of that object are not altered or influenced by that change, so that the object can be reintroduced in the simulation environment with minimum effort.

3.6. Exchangeable

The object representations need to be exchangeable among different users or designers, without having to worry about dedicated formats or specific software that is being used at either end.

3.7. Collaborative design / remote user access

The simulation environment needs to be accessible by the user or the designer, also from remote sites. Preferably, multiple users or designers should be able to access the environment at the same time in order to exploit the collaborative efforts within the simulation environment.

4. Guidelines

In this section the requirements as specified in section 3 will be translated into guidelines for a 3D-representation.

4.1. Low complexity

In order to achieve real-time response on a user's request, the object needs to be represented with as little amount of polygons possible (in case the 3D representation is polygonal of course). One way of achieving this is to make the size of the polygons adaptive to the local geometric variation and required resolution. Parts of the objects which are large, flat areas or which need to be seen only with low resolution allow a much larger polygon size than areas with a large amount of geometrical detail.

The requirement of low complexity may be contradictory to other requirements; therefore it may be necessary to use hierarchical models containing both low and high complexity models. In this last case mechanisms of level-of-detail and automatic 3D-model simplification could be very useful/required.

4.2. Smooth surface representation

The most typical 3D representation is polygonal, however for some applications it is necessary that the dimensions and shape of the virtual object corresponds to the actual shape of the object. Rather than using a polygonal topology, spline and nurbs surfaces may be much more accurate to represent the object. An important issue here is that the parameterisation of such a more advanced 3D representation is standardised well enough.

4.3. Texture mapping

Most objects can be characterised for a great deal by their shape, especially in industrial environments. However, shape alone does not give the user or designer a feeling of realism, therefore it is necessary that the objects are modelled and visualised with texture maps. In some cases (and depending on future availability on graphics hardware) other complementary surface details maps could be used (e.g. bump maps or displacement maps).

If the 3D-model has been created from 2D-images, the texture should be extracted from these original 2D-images. The issue then is that these images might partly overlap, and might have been taken with differing lighting conditions, such that parts of the 3D-model might look different in different images. In that case, the texture information from different images has to be fused.

Another point of interest is the way the texture is stored. Considering the requirements discussed in 3.4 and 3.5 (interaction and reusability), it is necessary to store the texture per functional block or per polygon basis.

The last point of interest is whether the exact texture is stored, or whether efficiency gains are possible when using an inexact, but perceptually adequate, texture.

4.4. Separate parts

a) Decomposition of model to separate parts

Furthermore, in order to clearly show the functionality of the object, and to provide the necessary interaction such as moving parts, the object needs to be built up out of different separate parts. To support some further requirements (such as independence of geometry), it is advised that these parts be associated with logical entities (e.g. door, button).

b) Definition of interactions of the parts

If a 3D-model is not just a passive object, but should also allow interaction with it, a conventional 3D-representation describing geometry and texture should be augmented with description of the possible interactions of each building block. This description should not only contain the functionality itself (e.g., if a button is pushed a bell should ring), but also the change in position of the building block itself (e.g., if a button is pushed, the button should really move downwards).

4.5. Animation & geometry independent

Any kind of interaction a particular object is provided with (pushing buttons, opening a valve, etc) should be independent of the actual geometry of that object. This means that if anything needs to be changed to the shape of the object – for instance when a different version of a similar object needs to be simulated – the interaction that has been defined can be kept.

This constrains the 3D-model in the sense that different functional parts should also be different geometrical parts in the 3D-model. For a polygonal mesh, the polygon boundaries should coincide with the boundaries of functional blocks. This also allows easy interchange of different functional blocks.

4.6. Open standard, platform independence

3D-models need to be interchanged among users or designers. Therefore the object representation has to be platform independent, and the format has to satisfy an open standard. A well known standard that satisfies these features is VRML (which stands for Virtual Reality Modelling Language).

4.7. Networking (access rights, etc.)

The simulation environment needs to be accessible through a network. Typically when trying to engineer or interact with object within a simulation environment, multiple people are involved in the design or simulation process. This also calls for special attention with respect to user access and user restrictions to avoid conflicts when manipulating objects in the simulation environment with multiple users at once.

Below, in table 1 we once again list the different requirements against the different guidelines that are discussed in detail here above. Naturally, as is shown in the table, guidelines may contribute to achieve one or more of the requirements listed in section 3.

Table 1. Requirements and guidelines for 3D graphics presentation

		Requirements						
		Real time Interaction	Exact Representation	Perceptually convincing	Functionality	Reusable	Exchangeable	Remote access
Guidelines	Low complexity	X						
	Smooth surface representation		X					
	Texture mapping			X				
	Separate parts				X			
	Animation / Geometry independent	X			X	X		
	Open standard						X	
	Networking						X	X

5. Conclusion

From the description of the partners, it can be seen that the VRML modeling language is currently the preferred representation for the BEYOND partners involved in this work package.

When judging VRML against the guidelines described in section 4, we see that VRML models already go quite far in satisfying requirements. Changing the level of detail allows for a low polygon count, while texture mapping is also possible. It is an open standard, and is also a very popular way of visualizing 3D content over the internet. Hence the current VRML models of the participants already can satisfy the guidelines and requirements to some extent.

6. References

[1] Salmela, M. Kyllönen, H. "Smart Virtual Prototypes: Distributed 3D Product Simulations for Web-Based Environments", in proceedings of the VRML2000 Conference, Monterey, CA, February 20-24, 2000.

[2] VRML97 Specification, ISO/IEC 14772-1:1997. Information technology -- Computer graphics and image processing -- The Virtual Reality Modeling Language (VRML) -- Part 1: Functional specification and UTF-8 encoding., http://www.web3d.org/technicalinfo/specifications/vrml97/index.htm.

[3] Kyllönen, H. Salmela, M. "VRP Creator, a process modelling tool for building smart virtual prototypes", in proceedings of NWPER'2000 Nordic Workshop on Programming Environment Research, Lillehammer, Norway, May 28-30, 2000.