



# A Content Adaptation Approach for On-Line Gaming on Various Networks and Terminals

F. Morán<sup>1</sup>, M. Preda<sup>2</sup>, G. Lafruit<sup>3</sup>, P. Villegas<sup>4</sup> and R.-P. Berretty<sup>5</sup>

<sup>1</sup> Universidad Politécnica de Madrid, ES <sup>2</sup> Institut National des Télécommunications, FR  
<sup>3</sup> Interuniversitair Micro Electronica Centrum, BE <sup>4</sup> Telefónica I+D, ES <sup>5</sup> Philips Research, NL  
<sup>1</sup> Francisco.Moran@upm.es <sup>2</sup> Marius.Preda@int-evry.fr <sup>3</sup> Gauthier.Lafruit@imec.be  
<sup>4</sup> Paulo@tid.es <sup>5</sup> Robert-Paul.Berretty@philips.com

**Abstract:** Most current multi-player 3D games can only be played on dedicated platforms, requiring specifically designed content and communication over a predefined network. To overcome these limitations, the OLGA (On-Line GAMing) consortium has devised a framework to develop real distributive, multi-player 3D games. Scalability at the level of content, platforms and networks is exploited to achieve the best complexity vs. quality. Additionally, standardised content representation and compression formats (MPEG-4, JPEG 2000) are used in OLGA's framework, enabling easy deployment over existing infrastructure, while keeping hooks to well-established practices in the game industry.

**Keywords:** 3D games, scalable coding, content adaptation, ISO standards.

## 1. Overview of the OLGA Project

OLGA ([www.ist-olga.org](http://www.ist-olga.org)) is the short name of a research project partially funded by the European Commission under FP6-IST ([cordis.europa.eu/ist](http://cordis.europa.eu/ist)). Its full name is "A unified scalable framework for **On-Line GAMing**", as its ultimate goal is: to provide a framework for developing **scalable** 4D (animated 3D) game content that can be adaptively streamed to a variety of terminals over heterogeneous networks; and to do so by using (and improving, whenever possible!) de jure international **standard** coding formats such as MPEG-4's AFX (Animated Framework eXtension) [5].



Figure 1: Screen shots from both the PC and CP versions of GOAL, OLGA's game.

Thanks to OLGA's scalable 4D content authoring and compression tools, it is possible to render the same textured 4D content at wildly different qualities and frame rates, according to each network and terminal profile. Figure 1 gives an idea of how OLGA's game test bed, named GOAL, runs on a PC and a CP (Cell Phone). Besides, we decided to enable the players to publish their own 4D content for its use in the game: OLGA's tools are not only provided to game designers, but also to end users.

But OLGA's mission was not only producing scalable 4D content authoring and compression tools. Another two of its main goals were to deploy a scalable game platform (an infrastructure consisting of both servers and network), and to provide a set of terminals to validate OLGA's technology by implementing GOAL on them.

## 2. OLGA's 4D Content Authoring and Compression Tools

Only a few years ago, high quality 3D graphics were a crucial asset for making a computer game successful. Nowadays, they are practically taken for granted: for current players, 4D content looking great is not a bonus but nearly a must. And creating compelling 4D objects and characters is a very time-consuming task even when that content is not scalable.

A first key ingredient of OLGA is its software toolset for content creation, conversion and compression, which provides game designers (as well as end users) with flexible solutions to create scalable 4D content from scratch, or to recycle already existing 4D content to have it be scalable, and to compress it efficiently. Scalable (off-line) coding is of the utmost importance for OLGA to enable the continuous adaptation (at run-time and under constrained system resources) of the 4D content parameters, so that the best trade-off between instantaneous 3D rendering quality and animation speed can be achieved. Such adaptation is possible thanks to progressive bit-streams (low to high resolution information ordering) that can be stripped through packet selection mechanisms for view-dependent decoding (or even streaming) scenarios, in which only the visible portions of a 3D object geometry and texture, given the current user's viewpoint, are transmitted and decoded at the appropriate quality.

A second key ingredient of OLGA was compliance to the maximum possible extent to international standards. MPEG-4 was chosen because it already featured a rich set of tools for scalable 4D content when OLGA started: WSS (Wavelet Subdivision Surfaces) for **3D geometry** [5, 6]; JPEG2000 multiplexed within MPEG-4 streams for **2D textures**, and BBA (Bone-Based Animation) for **animation** [5, 7].

With respect to **3D Geometry**, several 3ds Max plug-ins have been designed and implemented. They enable an artist to automatically simplify an arbitrary connectivity 3D mesh, hierarchically model it and code it in a scalable manner. Our 3D mesh simplification 3ds Max plug-in is based on the QEM (Quadric Error Metrics) technique by Garland [3] and yields significant improvements over 3ds Max's native mesh simplification plug-ins: the geometry obtained is much more efficient (in terms of triangle count for a given approximation error) and the texture coordinates are correctly handled. In addition it allows the artist to control more closely the mesh decimation and obtain more subjectively faithful final results by selecting certain regions to be preserved. As for the coding, it can be compliant to the "WaveSurf" tool already in

MPEG-4's AFX, or follow the PLTW (Progressive Lower Trees of Wavelet coefficients) technique [1], developed within OLGA and proposed to MPEG for its adoption in a future Amendment of AFX. The two corresponding decoders are both integrated in OLGA's software framework for the PC platform (for the CP platform, only the PLTW-based decoder has been ported to Symbian OS v8, since it has less memory requirements than MPEG-4's "WaveSurf" tool).

For **2D Textures**, after carrying out a comparative study between JPEG, JPEG 2000 and MPEG-4's VTC with respect to desired functionalities within OLGA, JPEG 2000 was selected, and several tools have been developed. First, a plug-in enabling 3ds Max to import and export JPEG 2000 textures was provided. We continued with tools enabling view-dependent texture streaming thanks to JPEG 2000 and JPIP (JPEG 2000 Internet Protocol), in which a bit-stream packet selection mechanism takes the user's viewpoint information into account. Implementations have been made for both the PC and CP platforms, and both the JPEG 2000 and JPIP decoders were optimised towards their usage in a 3D graphics texture context, and extended with additional control tools tailored to a view-dependent texture streaming scenario. The JPIP cache mechanism was adapted to minimize the memory usage in the CP platform. Finally, a JPEG 2000 bit-stream packet selector has been integrated in the *Simplificator* module running on server side, that supports resolution scaling and bit-plane removal. The LOD selection takes into account both the available bandwidth between server and terminal, and the terminal screen resolution. In addition, drawbacks in the current IFS (Indexed Face Set) tool of MPEG-4 [4] were detected, and we defined the so-called "IFS++" format for 4D meshes supporting vertex attributes such as texture coordinates and bone-vertex influence coefficients.

Concerning **Animation**, the work was concentrated on virtual characters, the most complex objects in a 3D game. Indeed, OLGA's main vision, using scalable content within a standardized framework, was applied for virtual characters as well. We used as a basis the BBA specification, which defines a complete framework for representing and animating skinned models. In addition to the compression of the object graph, based on MPEG-4's generic scene coding tool, BIFS (BInary Format for Scenes) [4], BBA defines a compressed representation of the animation parameters. In order to visualize MPEG-4 content for both PC and CP terminals, we selected a small number of scene graph nodes defined by BIFS, that is enough to represent static and animated textured 3D objects. We first implemented a light BIFS decoder, we plugged in both JPEG and JPEG 2000 decoders and, to support animation, we developed BBA decoders for Windows and Symbian OS v8. Finally, we developed the rendering layer by using DirectX 9 for PC and OpenGL ES for CP.

### 3. Servers, Network and Terminals

The work related to servers and networks comprised the design, development and testing activities for the integration of the game test bed versions with the various versions of the network architecture. Both the PC and CP clients communicate and authenticate with the central lobby server, which manages the game logic servers deployed in the network. Load balancing and recovery mechanisms for the distributed

game logic servers were implemented and successfully tested. The game logic server has basic gaming functionality and can handle non-player characters, both static and dynamic content. Instead of using a completely centralised solution, or one with a grid of homogeneous servers, we decided to have many heterogeneous zone game servers and local content servers, potentially hosted at the most powerful PCs of the players themselves. This allows a high degree of network scalability against the number of clients.

As for the terminals, they range from high-end gaming PCs to laptops but, more importantly, also mobile terminals have been used within the project: GOAL is available on CPs based on Symbian OS v8 and supporting J2ME, notably the Nokia 6630. Game logic was implemented on both platforms, and decoders for the simplified content downloaded from the network are integrated in both versions of the game.

**PC:** Special attention has given to the final aspect of 3D gaming content. To make the game experience more immersive, we have endowed some terminals with auto-stereoscopic 3D displays. Multiple images associated to different viewpoints could be rendered on the device, but this solution is not optimal in terms of bandwidth or computational complexity. Instead, we render only one viewpoint and provide the depths information that is available in the z-buffer of the GPU to the 3D display; then, a dedicated processor renders the desired viewpoints at high quality [2].

**CP:** A part of the software is programmed in Java, and the content decoders are programmed in Symbian, the Symbian framework being connected through a socket with the Java game engine. Rendering of the final graphics is done in software on the ARM embedded in the OMAP processor of the Nokia 6630. In fact, the OLGA consortium was the first to report a fully functional application rendering MPEG-4's AFX content on a Symbian OS CP. Besides, as OLGA partners are consistently providing input to the MPEG-4 standardisation process, the project results will be available on any other future MPEG-4-compliant platforms, breaking open the 3D gaming market on mobile devices.

## References

1. M. Avilés, F. Morán and N. García: "Progressive Lower Trees of Wavelet Coefficients: Efficient Spatial and SNR Scalable Coding of 3D Models", Procs. PCM (Pacific-rim Conference on Multimedia), p. 61-72, November 2005.
2. R.-P. M. Berretty, F. J. Peters and G. T. G. Volleberg: "Real Time Rendering for Multiview Autostereoscopic Displays", Procs. Stereoscopic Displays and Applications Conference, SPIE vol. 6055, p. 208-219, January 2006.
3. M. Garland and P. S. Heckbert, "Surface Simplification Using Quadric Error Metrics", Procs. ACM SIGGRAPH, p. 209-216, August 1997.
4. ISO/IEC JTC1/SC29/WG11, a.k.a. MPEG (Moving Picture Experts Group): "Standard 14496-1", a.k.a. "MPEG-4 Part 1: Systems", ISO, 1999.
5. ISO/IEC JTC1/SC29/WG11: "ISO/IEC 14496-16", a.k.a. "MPEG-4 Part 16, AFX (Animation Framework eXtension)", ISO/IEC standard, 2004.
6. F. Morán and N. García: "Comparison of Wavelet-Based 3D Model Coding Techniques", IEEE Trans. Circuits and Systems for Video Technology, vol. 14-7, p. 937-949, July 2004.
7. M. Preda and F. Prêteux: "Virtual Character within MPEG-4 AFX", IEEE Trans. Circuits and Systems for Video Technology, vol. 14- 7, p. 975-988, July 2004.