OLGA: On-Line GAming over Heterogeneous Platforms Thanks to Standard Scalable Content

Francisco Morán[1], Marius Preda[2], Gauthier Lafruit[3], Paulo Villegas[4] and Robert-Paul Berretty[5]
[1] *Universidad Politécnica de Madrid* [2] *Institut National des Télécommunications,*
[3] *Interuniversitair Micro Electronica Centrum* [4] *Telefónica I+D* [5] *Philips*
[1] *Francisco.Moran@upm.es* [2] *Marius.Preda@int-evry.fr* [3] *Gauthier.Lafruit@imec.be*
[4] *Paulo@tid.es* [5] *Robert-Paul.Berretty@philips.com*

Abstract

Most current multi-player 3D games can only be played on a single dedicated platform such as PCs, video-consoles, or (very recently) cell-phones, requiring specifically designed content and inter-terminal 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 where scalability at the level of content, platforms and networks is exploited to achieve the best complexity vs. quality and load-balancing trade-offs given the distributive resources available over the end-to-end delivery chain. Additionally, standardized content representation and compression formats (XML, 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.

1. Introduction

OLGA (www.ist-olga.org) is a research project partially funded, from April 2004 to September 2006, by the European Commission under FP6-IST. Its full name is "A unified scalable framework for **O**n-**L**ine **GA**ming", and it targets technologies for multi-player 3D games, which nowadays often rely on dedicated/ proprietary technological solutions (e.g., massively parallel, brute-force grid computing) or a priori scaling down paradigms to the weakest node in the infrastructure. This presents problems for:

- game developers, who cannot provide the adequate quality for every platform/network combination;
- platform builders, who want to be able to diversify their platform characteristics while still being able to allow for game playing;
- above all, end-users, who want to be able to roam attractive games in different usage contexts, inside and outside the home, without "being trapped" into a single platform/network configuration.

With respect to the latter, the main inherent difference/heterogeneity among PCs (Personal Computers) is their rendering power, assuming that a broadband Internet connection is available. However, OLGA's set of terminals includes also CPs (Cell Phones) using Symbian OS v8, which are not only limited in terms of 3D graphics acceleration, but perhaps even more importantly in terms of bandwidth, as they are linked to the Internet through 3G connections — at best!

The OLGA consortium has therefore targeted the roaming of on-line 3D games over heterogeneous networks and terminals thanks to the efficient and standardized compression of the game content, and the distribution of the game servers and network architecture, allowing adaptation by:

- providing a framework for developing **scalable** 4D game content that can be adaptively streamed to a variety of terminals over heterogeneous networks,
- using (and improving, whenever possible!) de jure international **standard** coding formats such as the ones produced by MPEG, notably MPEG-4's AFX (Animated Framework eXtension) [12].

Thanks to the scalable 4D content authoring and compression tools produced within OLGA it is possible to:

- use popular commercial 3D modelling and animation software packages such as 3ds Max and export MPEG-4-compliant scalable 4D content;
- analyse existing 3D models described by classic geometry modelling paradigms (e.g., triangular meshes), or 2D textures coded with older standards, and adapt that content for hierarchical transmission and rendering to permit scalability;
- attach the appropriate semantics to a generic articulated object hierarchy that will be used to continuously tune the animation resolution.

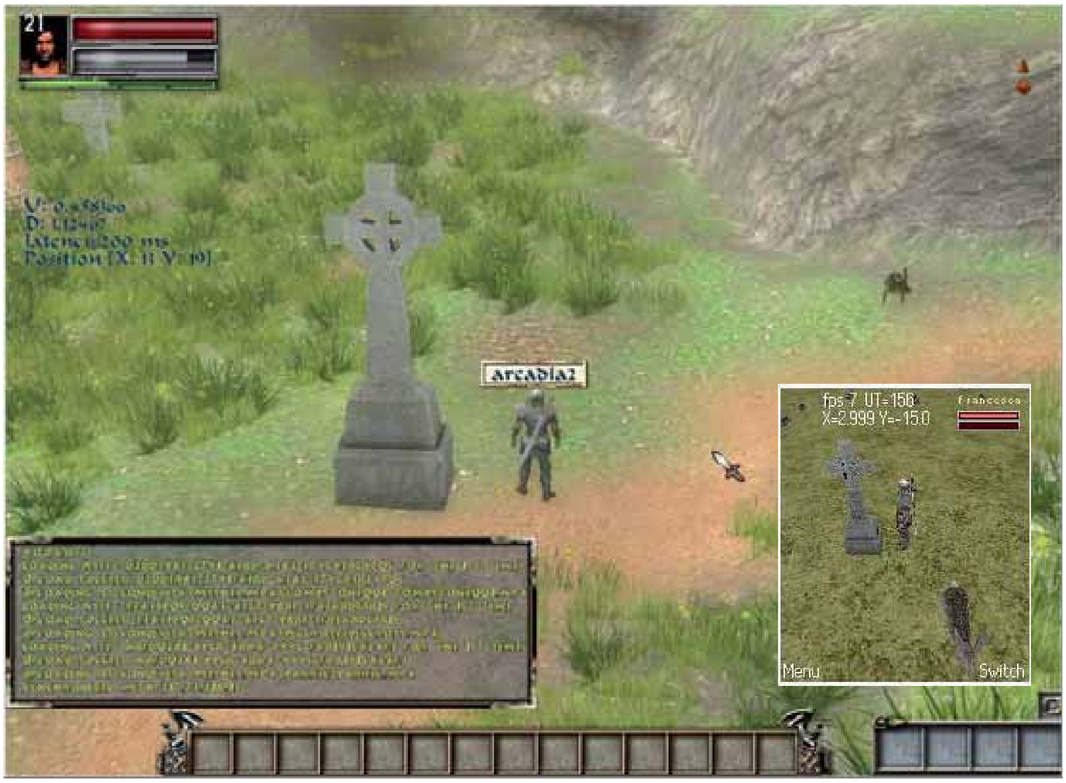


Figure 1: Screen shots from both the PC and CP versions of GOAL, OLGA's game.

In particular, thanks to the OLGA framework it is possible to render the same textured 4D content at wildly different qualities and frame rates, according to each particular network and terminal profile: see Figure 1. Besides, game the players can publish their own 4D content for its use within the game, so OLGA's tools are not only provided to game designers, but also to end users.

Within this document, Section 2 explains how scalable coding can be exploited for adapting the execution time in a specific terminal, under excellent quality vs. bit-rate vs. memory vs. execution time trade-offs of the 3D geometry, textures and animation. Section 3 comments on other main goals of OLGA: to design and implement 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 on them a multi-platform, multi-player game test bed, named GOAL. Finally, Section 4 concludes our presentation.

2. Standard scalable 4D content

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 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 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 are transmitted and decoded at the appropriate quality. The animation quality can also be scaled by performing only those transformations yielding a visible effect for the player.

As stated in the introduction, another key ingredient of OLGA, besides scalability, was compliance to the maximum possible extent to international standards. MPEG-4 was chosen because it already featured the following tools for scalable 4D content when OLGA started:

- **3D geometry:** among its several tools targeting the compression of polygonal meshes, we chose the one based on WSSs (Wavelet Subdivision Surfaces), known as "WaveSurf" [12], [16];
- **2D textures:** VTC (Visual Texture Coding) [10] is also wavelet-based, as is JPEG 2000 [8], which is not part of MPEG-4, but that MPEG-4 now supports as a format for textures thanks to a proposal from the OLGA consortium, and that was chosen for OLGA after a comparative study against VTC;
- **animation:** BBA (Bone-Based Animation) [12], permits to animate generic articulated characters based on the "skeleton and skin" paradigm.

2.1. 3D geometry

Several 3ds Max plug-ins have been implemented that enable an artist to automatically simplify an arbitrary connectivity 3D mesh, remesh it to have subdivision connectivity, and code it in a scalable manner:

- Our 3D mesh simplification 3ds Max plug-in, olgaQAttSimp, is based on the QEM (Quadric Error Metrics) technique by Garland [6] and yields significant improvements over 3ds Max's native Optimize: the geometry obtained is much more efficient (in terms of triangle count for a given approximation error) and the texture coordinates are correctly handled. Compared to the MultiRes modifier that comes also standard with 3ds Max, olgaQAttSimp is very efficient when it comes to smooth content, and roughly equivalent for "not-well-rounded" shapes. But, in all cases, 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. A simplificator software module has also been developed based on olgaQAttSimp, and integrated in the LCSs (Local Content Servers: see Section 3) to allow run-time vertex removal.
- The coding 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], explained below and proposed to MPEG for its adoption in a future Amendment of AFX. The two corresponding decoders (MPEG-4-compliant and PLTW-based) are both integrated in OLGA's software framework for the PC platform. As 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.

2.1.1. Geometry quality/bit-rate/memory trade-off.

Once a 3D shape is modelled as a WSS, it is fit for multi-resolution coding. Our research in this field focussed on new methods that could be more suitable for resource-limited devices than the SPIHT-based ones, like the WaveSurf tool in MPEG-4's AFX [12]. For a decade already, the SPIHT (Set Partitioning In Hierarchical Trees) technique [19] has been the reference against which to compare other coding techniques based on the wavelet transform. It was originally designed to code scalar wavelet coefficients, but has been extended to handle 3D coefficients, such as the ones resulting from colour images or 3D surfaces modelled thanks to WSSs [13], [16]. The problem of SPIHT is that, although its bit-streams are SNR scalable, they are not spatially scalable, and cannot be easily parsed according to a given maximum resolution (i.e., number of pixels or triangles) or LOD (Level Of Detail) tolerated by the decoder. There is little point in encoding a 3D mesh with thousands of triangles if the CP that must render it can barely handle hundreds. Furthermore, from the memory viewpoint, having a perfectly SNR scalable bit-stream that may have bits corresponding to details of LOD 3 before those of LOD 1 makes also little sense, as the decoding process alone will completely eat up all the CP resources: even if memory is not allocated for the triangles of LOD 3 (which will never be rendered), their detail trees must be created to follow the SPIHT algorithm.

The main novelty of the PLTW technique [1] is that the resulting bit-stream does not impose on the less powerful decoders the need of building detail trees as deep as required by the maximum LOD encoded, because the wavelet coefficients are sent on a per-LOD basis, thus achieving "local SNR scalability" within "global spatial scalability". With PLTW, the set of coefficients is also hierarchically traversed, but they are scanned in LODs, which yields a spatially scalable bit-stream. The decoder first receives all the coefficients corresponding to a LOD and, only when it has finished reading them, it proceeds (if it has enough resources) with those from the next. However, with the introduction of bit-plane encoding, bits from each LOD are ordered in such a way that the first to arrive are the ones that contribute more to lower the reconstruction error, while bits from negligible coefficients arrive last. A comparison of our PLTW coder vs. two other SPIHT-based coders is illustrated by Figure 2, which plots, for two different 3D models, the rate distortion curves for: the PLTW coder, which does include AC (Arithmetic Coding) as a final step; a version of the SPIHT algorithm with AC; and the "WaveSurf" tool of MPEG-4, which also uses the SPIHT, but without AC. Except at very low rates, where the PLTW is still reconstructing upper LODs and does not benefit from the smoothing effect of subdivision (while its competitors do), PLTW always results in higher PSNRs for the same bit-rate. It is also noticeable how none of the SPIHT-based coders is able to reach the same PSNR as the PLTW coder even employing 160% (SPIHT-AC) or 330% (MPEG-4) of the bits used by PLTW for the same quantisation set of values. The poor results of the MPEG-4 "WaveSurf" coder are mostly due to the overhead introduced to support view-dependent transmission of coefficient trees.

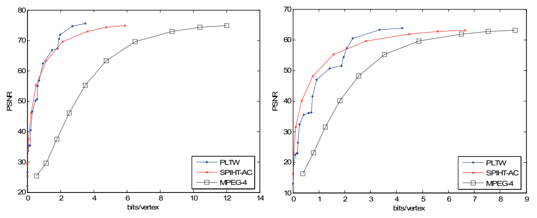


Figure 2: PLTW vs. SPIHT and MPEG-4's "WaveSurf" for the Max Planck and bunny models.

2.1.2. Geometry quality/bit-rate/execution time trade-off. The use of compressed, multi-resolution content enables the adaptation of its complexity (and hence also its visual quality) to the available bandwidth and terminal resources. WSSs permit to code the shape of a 3D model in a multi-resolution manner with very good compression, but require a large CPU overhead for a fine-grained, on-the-fly control of the content complexity in execution time regulated applications such as networked, interactive 3D games. In fact, the CPU overhead for controlling the execution time with MPEG-4's "WaveSurf" tool is sometimes as large as the 3D graphics rendering execution time itself [22]. Moreover, typical implementations of WSSs multiply by four the number of triangles in every subdivision step, which enables only very discrete LOD management, and therefore yields abrupt and often disturbing quality changes while only supporting coarse-grained adaptation to a target execution time. Besides improving the compression efficiency and the adequacy to weak terminals with the PLTW technique, we have introduced some add-ons to enable a low-complexity, yet efficient fine-grained quality/execution time trade-off in execution time control.

To achieve this target, the MPEG-4 WSS mesh regions are progressively decoded in a continuous LOD fashion, by subdividing only the important regions of the geometry. The importance and order for subdividing the triangles is given by their impact on the error to the target mesh, i.e. the triangles that decrease this error the most are subdivided first [15].

These non-uniformly subdivided WSS meshes allow a fine-grained control of the resolution of the geometry, resulting in small variations of the visual quality while achieving a target execution time. With special subdivision platform mapping techniques using LOD-based moving windows [21], the complexity of the subdivision control is largely reduced, resulting in an overhead of only a small percentage in the final decoding and rendering execution time for two different platforms: a high-end PC and a low-end CP [22].

In order to actually steer the execution time control, the execution time, and especially the rendering time, should be estimated for a large range of triangle budgets. We have used previously reported performance models for the software [20] and hardware [23] rendering pipelines, according to which the most important parameters are the number V of processed vertices (for the vertex processing) and the number F of fragments (for the rasterisation); additional parameters important for the software model are the number S of spans and the number T of visible triangles. The coefficients of the performance model are derived with an off-line calibration procedure that first measures on the device the rendering time for many different objects with different sizes (F) and complexity (V and T), and then computes the average values of the coefficients c_α ($\alpha \in \{T, F, S\}$) with multi-linear regression analysis.

2.2. 2D textures

After carrying out a comparative study between JPEG, JPEG 2000 and MPEG-4's VTC with respect to the considered criteria and desired functionalities within OLGA, the JPEG 2000 technology was selected, and several tools have been developed:

- A plug-in enables 3ds Max to import and export JPEG 2000-compliant textures (at the time of writing, Autodesk had just released 3ds Max 9, which did not support natively JPEG 2000 yet).
- 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 is adapted to minimise the memory usage in the CP platform.
- A JPEG 2000 bit-stream packet selector has been integrated in the simplificator module running on LCSs, that supports resolution scaling and bit-plane removal. The LOD selection takes into account both the available bandwidth between LCS and terminal, and the terminal screen resolution.

But OLGA's most important contribution with respect to textures has little to do with JPEG 2000 (except for having succeeded at having MPEG-4 support it as one of its native image formats), as in fact the work described above has mostly consisted in implementing and porting already existing algorithms and software. At least conceptually, OLGA's main contribution has been detecting drawbacks in the current IFS (Indexed Face Set) tool of MPEG-4, and defining the so-called "IFS++" format for 4D meshes with enriched vertex attributes such as multiple texture coordinates and bone-vertex influence coefficients. This activity has led to another MPEG proposal, which will hopefully be included as well in a future Amendment of AFX.

2.2.1 Geometry + 2D texture quality/bit-rate trade-off. Besides the execution time variation with the platform and content parameters [20], the linearity of the cost with the object parameters has also been observed in the bit-rate of the textured MPEG-4 objects: with a regression coefficient of 93% measured over 60 objects, the original MPEG-4 file size s decreases roughly bi-linearly with decreasing JPEG 2000 texture LOD (with negative slope m_1) and decreasing object mesh LOD (with negative slope m_2).

Small file sizes s with large (absolute values of) m_1 and m_2 correspond to small bit-rates that decrease very rapidly with decreasing LOD: the corresponding objects representing only a small fraction of the total bit-rate at all LOD levels, they have low priority to be scaled for global (over all objects) bit-rate adaptation. On the other extreme, large s with small m_1 and m_2 correspond to large bit-rates that decrease very slowly with decreasing LOD, hence representing barely any opportunity of down-scaling for global bit-rate adaptation. Consequently, large s with large m_1 and/or m_2 are the most appealing candidates for bit-rate adaptations: starting from a large full resolution bit-rate contribution, they scale very well by adjusting the texture and/or mesh LOD.

Together with the improvements introduced by the PLTW and BBA tools, a global quality/bit-rate/execution time control can be obtained over all objects. The precise details of this intelligent global adaptation are beyond the scope of this paper, since it is mainly related in finding heuristics for approximately solving an NP-hard knapsack problem: the reader is referred to [3], [17] for a framework of 3D inter-object adaptation using some tabular characteristics of each object.

2.3. Animation

Virtual characters are the most complex objects in a 3D game, and a special interest was foreseen with respect to them from the initial stages of OLGA. 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 BIFS (BInary Format for Scenes) [9], BBA defines a compressed representation of the animation parameters: bone transforms, muscle deformations and morphing weights.

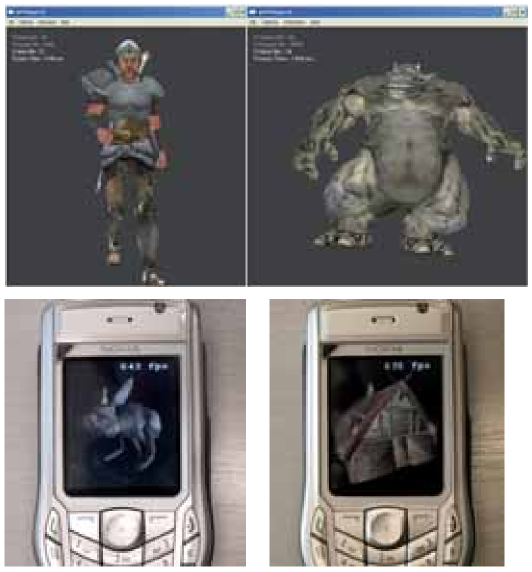


Figure 3: Some OLGA models loaded in the PC (top) and CP (bottom) versions of the 3D Graphics MPEG-4 player.

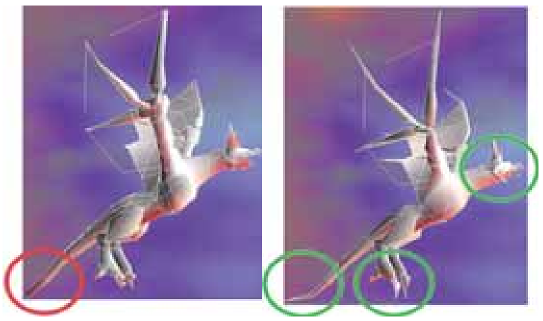
To visualize MPEG-4 content for both PC and CP terminals (see Figure 3), we selected a small number of the scene graph nodes defined in the BIFS specification, ensuring to represent static and animated textured 3D objects. Starting from GPAC [14], an open source BIFS decoder, we derived a simplified BIFS decoder by implementing only the selected nodes. To allow texture mapping, we plugged-in both JPEG and JPEG 2000 decoders. To support animation, we optimized the initial BBA decoder we had developed for

PC and also ported it to Symbian OS v8 for the selected CP (Nokia 6630). Finally, we developed the rendering layer by using DirectX 9 for PC and OpenGL ES for CP.

2.3.1. Animation quality/bit-rate trade-off. To represent compactly the data required by the animation of textured 3D models (varying vertex attributes: essentially spatial coordinates but also normals or texture coordinates), some kind of redundancy in the animation is usually exploited: either temporal, and then linear or higher order interpolation is used to obtain the value of the desired attribute between its sampled value at certain key frames; or spatial, and then nearby vertices are clustered and a unique value or transform is assigned to each cluster. MPEG standardized an approach for compression of generic interpolated data [9], able to represent coordinates and normal interpolation. While generic, this approach does not exploit the spatial redundancy. Concerning avatar animation, one of the most used animation content for games, ISO/IEC published in 1999 [10] and 2000 [11], under the umbrella of the MPEG-4 specifications, a set of tools named FBA (Face and Body Animation) allowing compression at very low bit-rates. The limitations of FBA consist mainly in the rigid definition of the avatar and the difficulty to set up the proposed deformation model. Some other methods reported in the literature are: quantization of the motion type [5], data transmission scalability by exploiting the 3D scene structure [7], and quantization to achieve data compression and incorporate intelligent exploitation of the hierarchical structure of the human skeletal model [4]. At the time the OLGA project started, we were in the final stage of standardizing BBA, an extension of FBA within MPEG-4's AFX.

BBA allows to represent animated, generic 3D objects based on the skin and bones paradigm, and to transmit the animation data at very low bit-rates by exploiting both the temporal and spatial redundancies of the animation signal. Within OLGA, we addressed the terminal/network adaptation, compression and rendering of BBA-based content. We considered the adaptation of animated content at two levels: geometry simplification constrained by dynamic behaviour [18] and animation frame reduction. The dynamic behaviour was expressed as constraints used to parameterise the well-known mesh simplification QEM technique. We introduced a weighting factor to specify how a given set of bones influences the simplification procedure. The biomechanical characteristics (i.e., the relationships between skin and bones) were directly exploited to constrain and control the simplification procedure. We applied the developed algorithm to OLGA animated objects, previously converted into MPEG-4-compliant skinned models. Figure 4 shows the comparative results of animated model simplification for the developed approach, called AC-QEM, vs. plain QEM.

Decoding and rendering animation data on small memory devices such as CPs requires server-side animation adaptation. Our approach was to reduce the number of the animation key frames so that the CP must only store a small quantity of information and use temporal interpolation. Animation simplification based on frame reduction was achieved by considering a progressive approach. Given an original animation sequence of n frames, to obtain a simplified sequence with m frames ($m < n$) approximating better the original curve, one has to minimise the area between the original curve and the reconstructed one. Considering this condition for all bones (or the subset of extreme bones), the optimisation problem becomes difficult to solve. To overcome the complexity, we adopted an incremental approach: for each pair of three frames and for each extreme bone, we compute the area between the original signal and the one reconstructed by linear interpolation. We sum these areas for all the extreme bones and the minimum of the sums indicates the frame that has to be removed. We repeat the algorithm until the number of removed frames equals $n - m$. After frame reduction, a new BBA stream is obtained by encoding the m frames, and indicating for each frame the number of intermediate frames to be obtained by interpolation on the terminal.



QEM-simplified model (491 vertices) AC-QEM-simplified model (497 vertices)

Figure 4: AC-QEM vs. QEM: qualitative results for the dragon model.

2.4. Complete 4D scene exporting in MPEG-4

Finally, three 3ds Max plug-ins have been released that are able to export whole scenes containing several 4D objects: the first exports fully MPEG-4-compliant textual ("*.xmt") and binary ("*.mp4") files; the second exports MPEG-4-compliant textual ("*.txt") for animated characters: object graph definition and animation data; and the third outputs bit-streams that are not yet fully MPEG-4-compliant in that they follow OLGA's IFS++ format and the PLTW-based coding of WSSs if the user so wishes.

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. Both the PC and CP clients communicate and authenticate with a central lobby server, which manages a distributed network of game logic servers, called ZGSs (Zone Game Servers), and content adaptation and delivery servers, called LCSs (Local Content Servers). Load balancing and recovery mechanisms for this distributed network of 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 centralized 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.

4. Conclusions

Today's multi-player 3D games often rely on dedicated/proprietary technological solutions for their servers (e.g., massively parallel, brute-force grid computing), and scale down content a priori, according to the bandwidth or rendering power of the "weakest" node in the infrastructure. The OLGA (On-Line GAming) consortium has opted for a completely different paradigm: exploiting the scalability at the level of content, platforms and networks, possibly adapting the content, network and processing load to the distributive resources available over the end-to-end delivery chain. OLGA's 4D (animated 3D) content is not stored locally on one single server or local storage medium (e.g., DVD), but is rather distributed over a multitude of servers spread all over the network with adequate load-balancing and fault-tolerance policies, and possibly hosted at the most powerful PCs of the players themselves!

The 4D content is actively pushed from the available servers to the gaming terminals but, since OLGA's 4D content authoring and compression tools are provided to end users as well as to game designers, the players can develop and publish their own content, which then becomes part of the persistent world, and benefits from OLGA's standardised framework for adapting scalable content to the varying processing and bandwidth capacities of a heterogeneous infrastructure, and to the very different rendering power of heterogeneous terminals. OLGA's 4D content authoring and compression tools do not impose constraints on the content complexity: game developers and players are free in their creativity, and OLGA's tools take care to adapt to any circumstances — not the other way around, as is usually the case…

Summarising, we have managed to integrate a chain of content conversion, transmission and rendering technologies into a heterogeneous infrastructure and terminal set, demonstrating real-time interactive 4D content adaptation. We have developed a distributive multi-player 4D game but, more importantly, we have developed a framework to develop distributive multi-player 4D games (or other multimedia applications with heavy and highly variable bandwidth and rendering requirements), and our framework hooks to a complete toolkit of standardised content representation and compression formats (XML, MPEG-4's AFX, JPEG 2000), enabling easy deployment over existing infrastructure, while not impeding well-established practices in the game development industry.

5. References

[1] M. Avilés, F. Morán and N. García: "Progressive Lower Trees of Wavelet Coefficients: Efficient Spatial and SNR Scalable Coding of 3D Models", Proceedings of PCM (Pacific-rim Conference on Multimedia), LNCS (Lecture Notes in Computer Science) vol. 3767, p. 61-72, Springer, November 2005.

[2] R.-P. M. Berretty, F. J. Peters and G. T. G. Volleberg: "Real Time Rendering for Multiview Autostereoscopic Displays", Proceedings of Stereoscopic Displays and Applications Conference, SPIE vol. 6055, p. 208-219, January 2006.

[3] J. Bormans, N. Pham Ngoc, G. Deconinck and G. Lafruit: "Terminal QoS: Advanced Resource Management for Cost Effective Multimedia Applications", chapter of "Ambient Intelligence: Impact on Embedded System Design", p. 183-201, Kluwer, 2003.

[4] S. Chattopadhyay, S.M. Bhandarkar and K. Li: "Virtual People & Scalable Worlds: Efficient Compression and Delivery of Stored Motion Data for Virtual Human Animation in Resource Constrained Devices", Proceedings of VRST (Virtual Reality Software and Technology) Symposium, p. 235-243, ACM, November 2005.

[5] M. Endo, T. Yasuda and S. Yokoi: "A Distributed Multi-User Virtual Space System", Computer Graphics and Applications, vol. 23, nr. 1, p. 50-57, IEEE, January 2003.

[6] M. Garland and P. S. Heckbert, "Surface Simplification Using Quadric Error Metrics", Proceedings of SIGGRAPH Conference, p. 209-216, ACM, August 1997.

[7] T. Hijiri, K. Nishitani, T. Cornish, T. Naka and S. Asahara: "A Spatial Hierarchical Compression Method for 3D Streaming Animation", Proceedings of Web3D-VRML Symposium, p. 95-101, ACM, February 2000.

[8] ISO/IEC JTC1/SC29/WG1, a.k.a. JPEG (Joint Photographic Experts Group): "Standard 15444-1", a.k.a. "JPEG 2000 Part 1: Core coding system", ISO, 2004.

[9] 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.

[10] ISO/IEC JTC1/SC29/WG11, a.k.a. MPEG: "Standard 14496-2", a.k.a. "MPEG-4 Part 2: Visual", ISO, 1999.

[11] ISO/IEC JTC1/SC29/WG11: "Standard 14496-2/ AMD1", a.k.a. "MPEG-4 Part 2: Visual, Amendment 1: Visual extensions", ISO, 2000.

[12] ISO/IEC JTC1/SC29/WG11: "Standard 14496-16", a.k.a. "MPEG-4 Part 16: Animation Framework eXtension (AFX)", ISO, 2004.

[13] A. Khodakovsky, P. Schröder and W. Sweldens: "Progressive Geometry Compression", Proceedings of SIGGRAPH Conference, p. 271-278, ACM, July 2000.

[14] J. Le Feuvre: "GPAC", available on-line below "http://gpac.sourceforge.net./".

[15] A.W.F. Lee, W. Sweldens, P. Schröder, L. Cowsar and D. Dobkin: "MAPS: Multiresolution Adaptive Parameterization of Surfaces", Proceedings of SIGGRAPH Conference, p. 95-104, ACM, July 1998.

[16] F. Morán and N. García: "Comparison of Wavelet-Based 3D Model Coding Techniques", Transactions on Circuits and Systems for Video Technology, vol. 14, nr. 7, p. 937-949, IEEE, July 2004.

[17] N. Pham Ngoc, G. Lafruit, J. Mignolet, S. Vernalde, G. Deconinck and R. Lauwereins: "A Framework for Mapping Scalable Networked Multimedia Applications on Run-Time Reconfigurable Platforms", Proceedings of ICME (International Conference on Multimedia and Expo), vol. 1, p. 469-472, IEEE, July 2003.

[18] M. Preda, S. Tran and F. Prêteux: "Adaptation of Quadric Metric Simplification to MPEG-4 Animated Object", Proceedings of PCM, LNCS vol. 3767, p. 49-60, Springer, November 2005.

[19] A. Said and A. Pearlman: "A New, Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", Transactions on Circuits and Systems for Video Technology, vol. 6, nr. 3, p. 243-250, IEEE, June 1996.

[20] N. Tack, F. Morán, G. Lafruit and R. Lauwereins: "3D Graphics Rendering Time Modeling and Control for Mobile Terminals", Proceedings of Web3D Symposium, p. 109-117, ACM, April 2004.

[21] K. Tack, G. Lafruit, F. Catthoor and R. Lauwereins: "Eliminating CPU Overhead for On-the-fly Content Adaptation with MPEG-4 Wavelet Subdivision Surfaces", Transactions on Consumer Electronics, vol. 52, nr. 2, p. 559-565, IEEE , May 2006.

[22] K. Tack, G. Lafruit, F. Catthoor and R. Lauwereins: "Platform Independent Optimisation of Multi-Resolution 3D Content to Enable Universal Media Access", The Visual Computer, vol. 22, nr. 8, p. 577-590, Springer, August 2006.

[23] M. Wimmer and P. Wonka: "Rendering Time Estimation for Real-Time Rendering", Proceedings of Eurographics Symposium on Rendering, p. 118-129, June 2003.