The 5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage
VAST (2004)
K. Cain, Y. Chrysanthou, F. Niccolucci, N. Silberman (Editors)

A case study of a virtual audience in a reconstruction of an ancient Roman odeon in Aphrodisias

Pablo de Heras Ciechomski, Branislav Ulicny, Rachel Cetre and Daniel Thalmann

Virtual Reality Lab, EPFL, CH-1015 Lausanne, Switzerland

Abstract

The benefits of including virtual humans into cultural heritage reconstructions are twofold: the realism of architectural models is increased by populating them; and, as well, it allows to preserve the intangible heritage describing how people in historical times behaved. We present a case study, where we create an interactive real-time scenario of a virtual audience in an ancient Roman odeon in Aphrodisias. Based on historical sources, we reconstruct both the building and the people. Inhabited virtual heritage applications require careful balancing of computational resources between the visualization of environment and the visualization of people. We describe several techniques that allow us to achieve high visual quality for a large number of virtual humans rendered together with a complex architectural model while still keeping interactive framerates. An important part of the heritage recontruction is the authoring, we propose a comprehensive framework for authoring of crowd scenarios.

1. Introduction

Our work is part of a larger effort in the context of the European project called ERATO. The goals of ERATO are identification, evaluation and revival of the cultural heritage of theaters and odea from the ancient Roman period. The ancient odeon of Aphrodisias was selected to be reconstructed as a 3D model populated by a large animated virtual audience following an interactive scenario.

The ancient city of Aphrodisias, once the capital of the province of Lydia in the Roman empire, is located near the village of Geyre in the district of Karacasu 38 kilometers south of Nazilli in southern Turkey. It is one of the most interesting archeological sites of the Aegean region. In Roman times, Aphrodisias had a famous sculpture academy, most probably because of the high-quality marble quarried in its vicinity.

The odeon, a small white marble theater, was constructed in 2 AD. At that time, it was covered with a wooden roof. The floor of the circular orchestra section was decorated with mosaics. Inside and outside were many statues and magnificent sculptures, produced by the school which was located next door. Time has taken its toll on the building, and additionally, in the fourth century, it was damaged by an earthquake. Currently, only the lower part is preserved. Figure 1 shows a photograph of the current state of the odeon.

Figure 1: *Photograph of the present state of the Aphrodisias odeon.*

Our goal is to reconstruct the odeon in virtual reality according to historical sources and surveys of the site in present time (see Section 2), and furthermore to recreate life inside the theater as it was in ancient times (see Section 3). For the ERATO project, we target to handle an order of magnitude larger crowd than in previous works on crowds in vir-

Figure 2: *Real-time virtual audience in the Aphrodisias odeon.*

tual heritage. This brings new challenges for the design of the real-time rendering, animation and crowd control software, and for the scenario content creation and management.

Several other works have been dealing with virtual humans in virtual heritage reconstructions, however none of the these were targeting real-time applications with a combination of similar scale and visual quality. In the non-real time domain, Magnenant-Thalmann *et al.* presented a reconstruction of the Xian terra-cotta warrior army [MTPM97]. Real-time virtual heritage applications usually deal only with a smaller number of virtual characters: DeLeon [DeL99] and Frohlich *et al.* [FLKB01] created a virtual cathedral with a single "tour guide"; Foni *et al.* [FPMT02] and Papagiannakis *et al.* [PFMT03] added a small number of very detailed worshippers into virtual mosques. Ulicny and Thalmann [UT02] presented a 3D animated crowd peforming virtual prayer, however the number of agents was an order of magnitude smaller than in our scenario. More recently, Heigeas *et al.* described a simulation of a larger non-photorealistic impostor crowd in an ancient Greek agora of Argos [HLTC03]. While being well suited for their particular application of an open space area, nevertheless, the visual quality of the characters can decrease significantly when the camera gets closer.

In our case, we could not use impostors, as the viewer will be too close to the audience because of the space constraints of the actual architectural design of the odeon. Therefore we needed a method that could render high quality virtual humans even from a close zoom. We create a real-time crowd rendering and animation engine in order to display and animate the 3D virtual audience inhabiting the ancient theatre. Section 4 presents an overview of our rendering approach and discusses design issues.

No matter how good crowd rendering or behavioral model, a virtual crowd simulation is not very useful, if it is too difficult to produce content for it. The authoring possibilities are an important factor influencing the usability of a crowd simulation system, especially when going beyond a limited number of "proof-of-concept" scenarios. Usability and flexibility of the authoring process is especially important for virtual heritage applications. A large part of the final result depends not only on the creators of the system, but also on the artists using it while creating the content. Additionally, while doing the reconstructions based on histor-

ical sources, it is common to change both the scenario and geometrical configuration of the architectural model several times, as various sources sometimes do not agree. Section 5 describes our approach for crowd scenario authoring.

2. Reconstruction of the Aphrodisias Odeon

The first task was to reconstruct the building of the Aphrodisias odeon (see Figures 3 and 4). We followed a methodology similar to the one outlined by Papagiannakis *et al.* [PLFMT01]. A three-dimensional geometrical model was created based on architectural plans, visual measurements and historical literature [Eri86, Ize92] using the *3ds max* modelling package [3DS04]. The textures have been created based on the pictures taken in situ. Images were restored, where the shape was redefined to compensate for lens distortion and colors have been adjusted to correct for the effects of different lighting conditions while taking pictures on the site. Figure 5 shows a comparision between the real and the virtual Aphrodisias odeon's cavea.

Figure 3: *Exterior view of the virtual odeon.*

One of the important considerations was to minimize both polygon count and texture size of the odeon model, as the graphics card resources have to be divided between handling of both audience and building rendering.

During the modelling process, a higher level of detail for texture and geometry is used. When the model is exported for real-time viewing, the details are reduced. To create details (such as lion's feet), we are using a combination of mesh and texture in order to keep a good balance between number of polygons and realistic representation. The current version of the odeon model has around fourteen thousand triangles.

In order to increase the visual realism of the odeon, we computed a global illumination corresponding to the geographical location of the actual odeon in Aphrodisias, and baked the lighting information into the textures of the model. The models of the virtual audience have then been visually

Figure 4: *Interior view of the virtual odeon.*

Figure 5: *Comparison between the real and the virtual Aphrodisias odeon's cavea.*

integrated into the lighting conditions of the odeon model (see Figure 2) by using real-time directional light corresponding to the light used while baking the textures and by modifying mesh colors to emulate the effect of shadows and patches of the light (see also Section 5). In the process of baking a texture the original color information is mixed with shadows and extra light information from the radiosity algorithms. This creates some extra space that could be overcome by using light maps.

3. Virtual Roman audience

Theatres are environments where the actors perform plays in front of an audience. In this work, we focus on the creation of the virtual audience. The creation of stage actors will be another part of the ERATO project. We survey histori-

cal sources concerning the appearances of the Romans (such as clothes, shoes, hairstyles and bodies) [Sym87, Cro02], and the distribution and behaviors of spectators in ancient Roman times [Kin79]. The sample population data should cover a spectrum of social classes present in the audience in ancient Roman theaters.

Our goal is to create an interactive scenario, which would specify some plausible behaviors for the audience in the odeon, and also would allow the user to have a certain influence on these behaviors. The audience is composed of numerous agents, which are able to react in different emotional ways depending on what is happening both on stage and around them.

According to Kindermann [Kin79], Roman audiences were very noisy: they were chatting and flirting, and it was very difficult to get them to calm down in order to start the show (often a theatre police had to intervene). Nobody was allowed to leave the odeon during the show. Famous actors were treated as superstars; they had their own "fan-club"; and as their popularity was measured by their fan's support, they often paid people in the audience to acclaim them and humiliate their rivals. Fans were over-reacting to such extent that sometimes the hissing could last until the actor left the stage. Positive reactions included clapping of hands, shouting "da capo!" or "bravo!", unison clapping, and so on. Examples of negative reactions were hissing, feet rattling, or shouting. The audience was not only reacting to their favorite actors, but they were reacting to the play as well: approving, disapproving, laughing out loud, crying, hailing a character, or discussing with a neighbor.

3.1. Scenario

Our scenario of the virtual audience is based on the descriptions above. However, as the work on the ERATO project is still in progress, instead of the audience reacting to the actual actors on the stage, the user controls a scenario flow by means of a user interface.

Several configurations of the audience in different emotional states are prepared by a designer (see Section 5). Configurations correspond to the reactions of the audience to different actors performing various parts of the play on the stage. The user selects the actor from the interface, and then the corresponding audience configuration is activated. For example, actor "A" elicits positive reactions in his fanclub, but negative reactions in the fanclub of actor "B" and vice versa. When there are no triggered events, agents apply their default behaviours such as changing of position, touching the hair, or looking at other agents.

Another option in the scenario is a free control of emotions, where the user can use a subset of the authoring tools to change the emotional state of the audience directly (see Section 5).

We use a reactive behavioral engine, similar to the one presented in [UT02], which is based on behavioral rules that can be activated by events taking into account the state of the agent. In our case, the state of the agent includes emotional state, social class and gender. The behavioral rules have several roles including handling of idle behavior, a contagion of emotions, and a management of variety, where, for example, people of different social classes react in a different manner even while being in the same emotional state.

3.2. Rendering Considerations

For creation of the virtual crowds, one of the biggest challenges is how to create variety while keeping speed of rendering and memory requirements at acceptable levels. The basic idea is to create several template meshes (see Figure 6) which are then in run-time cloned to produce a large number of people. These clones can then be modified by applying different textures, colors and scaling factors to create variety as in [UdHCT04].

Figure 6: *Texture combinations.*

Textures for virtual humans have to be designed carefully, trying to get the best visual impact while optimising the size of the texture. In our case, a particularly difficult task was the reproduction of togas' folds in order to compensate for the models' lack of details to create the illusion of real clothes. Since our crowd rendering engine demands that only one texure per mesh is used, we had to unwrap the geometry of the models (see Figure 7).

Another factor that we needed to consider was design of meshes. Based on performance tests of our rendering engine (see Section 6), it was clear that using level-of-details meshes leads to significant improvements in performance. Ideally, this should be done automatically. However, we found that automatic tools do not give very good results in terms of reliable simplifications, especially considering that we already deal with meshes of low polygonal counts (our high definition mesh is around 1000 triangles).

There are a few points that need to be addressed for automation to be possible. Facial polygons have to be reduced at a slower rate than the rest of the body since we give

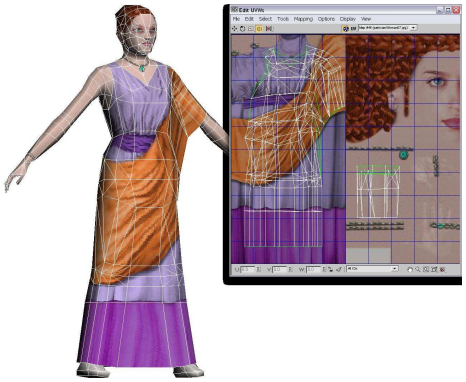

Figure 7: *Single texture mapping.*

Figure 8: *Screenshot from the real-time performance test application with 10 000 Romans.*

more importance to facial features. Joint vertices cannot always be reduced since they are specifically placed there for the bending deformations that occur. Edge-collapsing techniques [GH97] cannot re-arrange the vertices for the lower detail levels to fit the bending constraints, since they assume that vertices stay static, when in some cases they would have to be re-arranged. Texturing also suffers from warping problems, since we optimize the texture space by re-use of texture coordinates. For example, faces are usually mirrored on the triangles thus it is necessary to store only half of the face along the nose symmetry line. Because none of the available automatic tools were able to produce good enough results, we found it more practical to create all detail levels manually.

Finally, as our crowd rendering engine is restricted by memory use, the animation length has to be taken into account while creating behaviors. One second of animation contributes with around 2 megabytes of memory depending on how much the mesh representation can be packed (see Section 4.3).

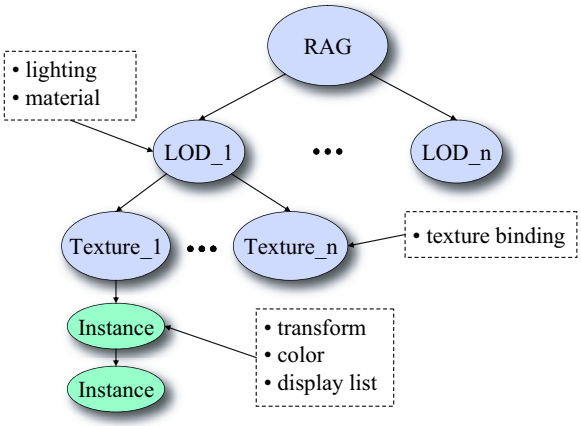

Figure 9: *Rendering acceleration graph.*

4. Crowd Rendering

Today's graphics hardware can render large amounts of geometry, however it is important to use this rendering power to the fullest. If the entire rendered scene was one mesh using one texture, only the geometrical complexity and its hardware representation would matter. However, in our case, we have humans represented by several meshes, one for each animation frame and template (an example rendering can be seen in Figure 8). The humans have differences in appearance in form of textures, positions, sizes and colors. A straightforward way of rendering these humans would be one by one, applying the desired texture, animation, position, size and color. It is improbable that this method would use the graphics card to its full capabilities. The performance of the straightforward approach depends on the size of the on-board graphics memory.

A more refined approach can be derived by taking a closer look at the internals of the grapics card. Graphics boards include, for example, texture caches, vertex caches that can reuse transformations applied to vertices and normals, vertex buffers keeping geometry like positions, normals and colors, triangle index buffers for indexing vertices, and normalizations of lighting calculations. Some rules of thumb of how to use these capabilities are given by graphics cards manufacturers [SCGK03]. The operations ordered by descreasing cost are: texture changes, followed by lighting calculations and modifying lighting parameters, then transformations performed by matrix operations and lastly vertex data arrangements. In the following sections we explain, how all of these optimisation techniques were utilised.

We used the OSG scenegraph[OSG04] and added one unique custom renderign node that contained the entire crowd. The roman odeon was imported from 3ds max using and OSG exporter along with textures.

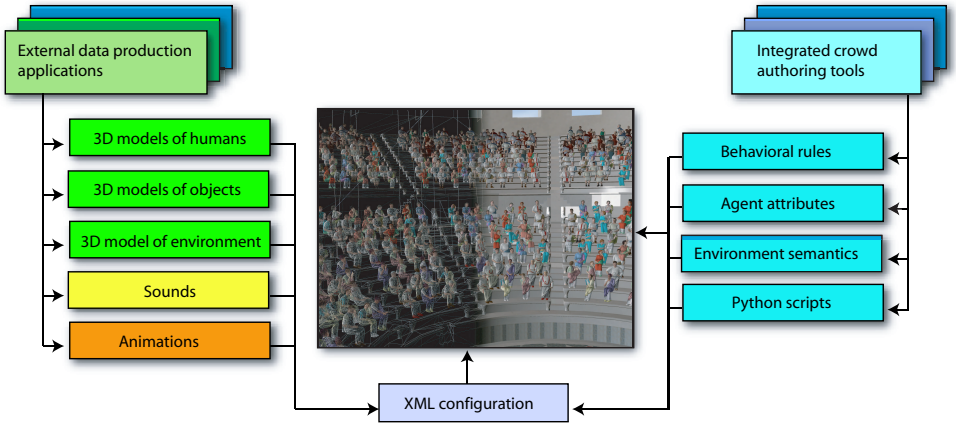

Figure 10: *Overview of crowd authoring framework.*

4.1. Tested Approaches

At an early stage, we considered adding variety in appearance by having several textures per human, i.e. to divide the mesh into zones for the torso, legs, head and arms. The performance of the straightforward approach was significantly decreased since textures for every zone had to be re-bound, thus creating cache misses in the graphics card texture cache. We tried to optimise performance by sorting humans by zones. Nevertheless, the extra costs of sorting (even while using a very optimised radix sort algorithm) an increased number of calls for glPush, glPop and glMultMatrix nullified the advantage. It also resulted in more complex logic in the rendering code leading to additional performance hits.

After these trials, we decided to use only one texure per human and to add variety by creating more textures. This data simplification made other ways of arranging data available.

4.2. Rendering Acceleration Graph

The Rendering Acceleration Graph (RAG) is a way of ordering geometry for the rendering pass to use minimal graphical state changes and to perform less complex logic. It sorts the humans by level-of-detail and by texture (see Figure 9). Level-of-detail sorting gives us two major benefits. Firstly, lighting parameters are set only once giving speed for the lowest level-of-detail human, since it has turned off lighting and re-normalisation of normals. Secondly, we get a partial front-to-back rendering, which speeds up rasterisation.

Since the humans are inserted and extracted from the RAG often due to changes in level-of-detail and visibility, these operations were optimised. We made sure that humans had all the necessary rendering parameters stored in atomic data structures because the rendering pass traverses the RAG sequentially resulting in memory coherent access.

Furthermore the memory used by these atomic structures was compressed by using shorter data types and bit fields for boolean variables.

4.3. Mesh Representations

We needed to reduce the amount of memory needed for mesh representation inside of display lists and also to use a more efficient data structure for mesh rendering. Triangle stripification of a mesh reduces the size necessary to describe which vertices the triangles are using [ESV96]. Since our characters always have one texture, we can make one connected strip using degenerate triangles, called a stitched strip (we use the NvTriStrip library). This gave use a memory efficient representation. For rendering optimisation, we used indexed interleaved vertex arrays.

Nevertheless, a straightforward combination of stripification and indexed vertex arrays would lead to problems with texturing. It is often the case that two triangles in our mesh share the same vertex index, thus positional information from a vertex is shared but the texture mapping coordinates for the same vertex index are not. If this is not considered, the triangles will be forced to use one of the uv-mappings, because the vertex array rendering in OpenGL assumes 1 to 1 mapping for uv-coordinates and positional information. To resolve this texturing problem, before passing the mesh to the stripifier, we have to duplicate the vertices that are indexed with several different uv-coordinates.

After the duplication and stripification steps, the mesh can be indexed in a un-ordered manner, leading to inefficient rasterisation. Additional performance can be gained by rearranging the vertices taking into account the vertex cache size, which can usually contain 16 transformed vertex values (see [Hop99]). This will create a second mapping step on top of the duplication step. To be able to fill the interleaved array, we need to compute the mapping from the basic mesh to

the vertex cache re-ordered one. Otherwise, the duplicated vertices would have to re-compute normals and to deform positions unnecessarily.

5. Scenario Authoring

The scenario authoring process is an important part of the virtual heritage application creation. It is common that many people with different expertise are involved, and most often the technical experts are not the ones creating content. Additionally, in the lifecycle of a typical virtual heritage project, the exact details of the targeted scenario can change several times as new facts are collected, or the known information is reinterpreted. Therefore, the authoring process should be easy to use and flexible enough to allow for rapid changes.

Crowd scenes are composed of different elements, both visual and non-visual, such as 3D models of agents and environment, sounds, behavioral rules and agents' attributes, environment semantics data or scenario configuration (see Figure 10). Creation of the crowd scenario involves selection of those elements out of libraries of already existing assets, or eventually the creation of new elements for a particular scenario. As crowd simulations can involve large numbers of entities, many times only templates of entities are stored in the library, and most of the content is then generated procedurally in run-time of the simulation (for instance, audience members are cloned out of template humans, see Section 3.2).

Different aspects of the simulation (as spatial, temporal or procedural) require different methods for authoring. Our approach is to use a collection of tools, each of them tailored for a different task: we use a graphical user interface with classical widgets for autoring of behavioral rules and environmental semantics; a scripting language to handle more complex initialization procedures or scenario logic evaluation; and a CrowdBrush tool to handle spatial distribution of agents and their characteristics [UdHCT04].

For scenario creation productivity it is beneficial to see the effects of changes as soon as possible, which holds true especially for real-time productions. Therefore for all our tools, the authoring is happening in real-time, the changes being propagated directly to the running simulation. The authoring tools coexist with the simulation engine in a single application.

5.1. Audience configuration

One of the tasks particular to the creation of the virtual audience is the distribution of spectators in the odeon and subsequent distribution of features to the people. We use several tools from our authoring framework to achieve this goal.

As a first step, we use a Python script to procedurally create grid of valid seating positions in the odeon. This allows us, for example, to easily add or remove whole rows of seats, to change seats' spacing or to move around the location of stairs to match changing archeological data as they become available. The process is interactive - it is necessary only to reload the new odeon model and change parameters of the script.

Then, the CrowdBrush tool allows the designer to fill the odeon with spectators by interactively adding and removing virtual humans at desired locations. Figure 11 shows, on the right side, the plan of the audience distribution according to social classes of people based on historical sources [Kin79]. Starting from the center, there are places for nobles, patricians and plebeians. On the left side of the figure is the distribution of 3D humans in the actual model of the Aphrodisias odeon according to the plan. The designer uses a "social class" brush to add new people from three sets of available templates depending on the social class. This tool affects the template of the virtual human both concerning the rendering and behavioral aspects and according to the value of the brush, the agents are being changed into nobles, patricians or plebeians.

Using another user interface widget, the designer can then interactively configure parameters of real-time lights (as position, orientation, color, or intensity) to adjust the illumination of the scene, so that the audience visually matches the look of the odeon. Furthermore, to get an approximation of shadows and patches of the light coming through odeon windows, the designer interactively adjusts colors of the audience meshes by using a "light" brush tool (see Figure 2).

Finally, the designer sprays emotions to different groups in the audience corresponding to "fan-clubs" of actors (see Section 3). The configuration of the audience can be saved and later re-loaded when the user selects the options of the scenario from the user interface (see also Section 3.1).

6. Performances

We measured the performance of the crowd rendering and animation engine on a PC equipped with a Pentium4 2.4 GHz processor, 1 GB of RAM, with NVIDIA GeForceFX 5950 Ultra graphics card with 256 MB of memory. Figure 12 summarizes the performance results. We performed two set of tests with varying number of virtual humans: first, without the use of any level-of-details optimization; and second, using a model with 4 level-of-detail meshes. The number of triangles per LOD was the following: the highest detail mesh had 1038 triangles, and the three lower resolution meshes used 662, 151, and 76 triangles respectively. A screenshot from these tests can be seen in Figure 8.

The performance with a level-of-details optimization is several times higher than, for example, in the recent work by Ulicny *et al.* [UdHCT04] which used a similar method but without level-of-details. Our system is able to handle more than 10,000 animated fully 3D virtual humans at interactive framerates.

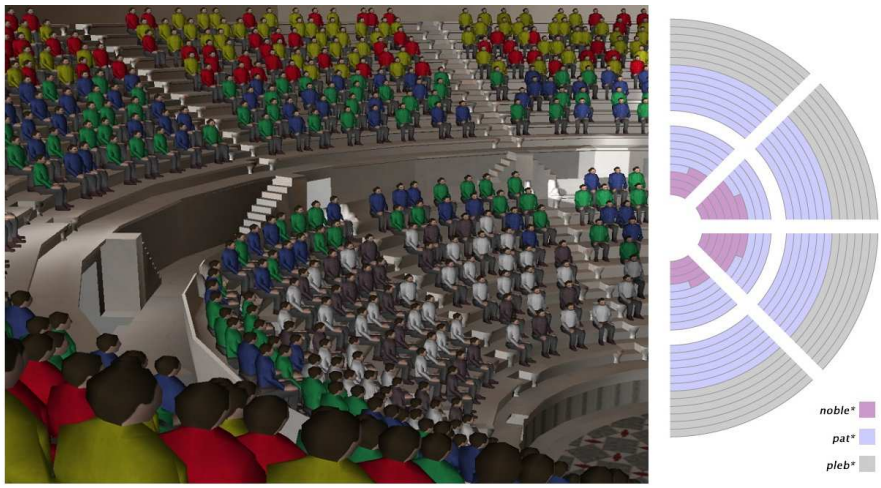

Figure 11: *Distribution of the audience in the odeon (different colors represent different social classes and gender).*

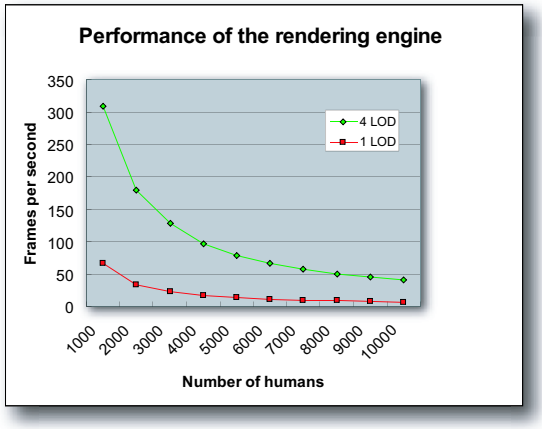

Figure 12: *Performance of the crowd rendering engine with and without level-of-detail optimization.*

Even while the full capacity of the Aphrodisias odeon is only around 700 people, the speed-up of crowd rendering (up to several hundreds of frames per second for an audience of the size needed to completely fill the odeon) has an impact on the overall visual quality of the reconstruction, as it allows to increase details of the architectural model.

7. Conclusions and Future Work

We proposed a methodology to create realtime virtual heritage reconstructions involving complex architectural models inhabitated by a large number of virtual humans. We demonstrated a set of techniques and practices concerning high quality realtime rendering and flexible authoring in the case study of the virtual audience in the reconstructed ancient odeon in Aphrodisias.

The ERATO project is in the middle of its duration. Our work presents a base, on which a more complex virtual heritage simulation will be built. The next step will be the integration of virtual stage actors in the scenario and the extension of behavioral repertoire of the audience using dynamic generated motions. Furthermore, the additional realism will be achieved by adding virtual sound generation based on the acoustical model of the odeon.

References

[3DS04] 3ds max, 2004. http://www.discreet.com/3dsmax.

[Cro02] CROOM A. T.: *Roman clothing and fashion*. Tempus Publishing, 2002.

[DeL99] DELEON V. J.: VRND: NOTRE-DAME CATHEDRAL: A globally accessible multi-user real-time virtual reconstruction. In *Proc. Virtual Systems and Multimedia 1999* (1999).

[Eri86] ERIM K. T.: *Aphrodisias: city of Venus Aphrodite*. Muller Blond & White, London, 1986.

[ESV96] EVANS F., SKIENA S., VARSHNEY A.: Optimizing triangle strips for fast rendering. In *IEEE VISUALIZATION '96* (1996), pp. 319–326.

[FLKB01] FRŒHLICH T., LUTZ B., KRESSE W., BEHR J.: The virtual cathedral of Siena. *Computer Graphik topics 3* (2001).

[FPMT02] FONI A., PAPAGIANNAKIS G., MAGNENAT-THALMANN N.: Virtual Hagia Sophia: Restitution, visualization and virtual life simulation. In *Proc. UNESCO World Heritage Congress* (October 2002).

[GH97] GARLAND M., HECKBERT P.: Surface simplification using quadric error metrics. In *Proc. SIGGRAPH '97* (1997), pp. 209–216.

[HLTC03] HEÏGEAS L., LUCIANI A., THOLLOT J., CASTAGNÉ N.: A physically-based particle model of emergent crowd behaviors. In *Proc. Graphikon '03* (2003).

[Hop99] HOPPE H.: Optimization of mesh locality for transparent vertex caching. In *Proc. SIGGRAPH '99* (1999), pp. 269–276.

[Ize92] IZENOUR G.: *Roofed theaters of Classical Antiquity*. Yale University Press, New Haven & London, 1992.

[Kin79] KINDERMANN H.: *Das Theater Publikum der Antike*. Otto Müller, Salzburg, 1979.

[MTPM97] MAGNENAT-THALMANN N., PANDZIC I. S., MOUSSALY J.-C.: The making of the terracotta Xian soldiers. *Digital Creativity 8*, 2 (1997), 66–73.

[OSG04] OpenSceneGraph, 2004. http://www.openscenegraph.org.

[PFMT03] PAPAGIANNAKIS G., FONI A., MAGNENAT-THALMANN N.: Real-time recreated ceremonies in vr restituted cultural heritage sites. In *Proc. CIPA '03* (2003).

[PLFMT01] PAPAGIANNAKIS G., L'HOSTE G., FONI A., MAGNENAT-THALMANN N.: Real-time photo realistic simulation of complex heritage edifices. In *Proc. Virtual Systems and Multimedia 2001* (2001), pp. 218–227.

[SCGK03] SHREINER D., COMMIKE A., GRANTHAM B., KUEHNE B.: Performance OpenGL: Platform independent techniques. In *SIGGRAPH '03 Course* (2003).

[Sym87] SYMONS D. J.: *Costume of Ancient Rome*. Chelsea House, New York, 1987.

[UdHCT04] ULICNY B., DE HERAS CIECHOMSKI P., THALMANN D.: Crowdbrush: Interactive authoring of real-time crowd scenes. In *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'03)* (2004), pp. 243–252.

[UT02] ULICNY B., THALMANN D.: Crowd simulation for virtual heritage. In *Proc. First International Workshop on 3d Virtual Heritage* (Geneva, 2002), pp. 28–32.