Lowering the Cost of Virtual Human Rendering With Structured Animated Impostors

Amaury Aubel, Ronan Boulic and Daniel Thalmann

Computer Graphics Lab
Swiss Federal Institute of Technology (EPFL)
CH 1015 Lausanne, Switzerland
Tel: +41 21 693 52 14 Fax: +41 21 693 53 28
e-mail: {aubel, boulic, thalmann}@lig.di.epfl.ch

ABSTRACT

We present in this paper an image-based rendering technique to represent virtual humans in real-time. Though we deal only with virtual humans, the concept can be applied to any articulated character. Our technique is termed *animated impostors* because it extends the notion of dynamic impostors to take into account changes in the character's appearance. We also show how impostors can be embedded into large-scale simulations/applications.

Keywords: virtual humans, impostors, LOD, sprites, image-based rendering.

1. INTRODUCTION

A lot of research has been devoted over the past decades to model and animate virtual humans in a realistic fashion. As a result, today's animators have tools that enable them to create so compelling synthetic actors that our visual system is sometimes mistaken, as in the recent film "Titanic". But realistic synthetic characters are not confined to the sole movie industry. They are being used increasingly in real-time applications such as computer games, collaborative virtual environments and for digital communities. However, one of the current limitations to displaying numerous virtual humans lies in the number of polygons that can be processed by the rendering pipeline. Geometric level of detail (LOD) is the most common answer to this problem. Pratt et al. [Pratt97] applied this technique to virtual humans in large-scale, networked virtual environments. In their application each simulated human has four different resolutions ranging from about 500 polygons down to only three polygons. This may be satisfying from a simulation point of view, for the system manages to sustain a high refreshment rate. In counterpart, the result is necessarily poor in terms of realism and aesthetics because there is no way a virtual human made up of a few dozen polygons can look good, no matter the viewing distance and the designer's skills.

Our approach to rendering virtual humans is based on the fact that there is a high degree of coherence between two successive frames. Even when rendering a complex, moving, self-deforming figure such as a virtual human, the changes from frame to frame in the animation remain small. Besides, since the human brain tends to reconstruct the missing data when shown only certain frames, a few key

postures are often sufficient to understand the whole animation. This principle has been demonstrated successfully for decades in the cartoon industry. Our technique takes advantage from this temporal coherence. It is an image-based rendering technique termed *animated impostors* because it extends the concept of dynamic impostors to take into account changes in the character's appearance. The impostor is *structured* in the sense that several planes may be used in order to achieve a correct visibility. Our system is currently implemented in software for want of a dedicated architecture. Hardware requirements include texture mapping and alpha channel support, which are commonplace on today's workstations.

The next section gives some insight concerning image-based rendering techniques. After briefly describing our virtual human model, we explain in section 3 how virtual humans can be represented with impostors. Section 4 deals with the image cache invalidation mechanism. The last-but-one section shows how this method can be embedded in real applications. Finally, we present in section 6 a summary of our results and give some hints for future work.

2. RELATED WORK

Image-based rendering is a technique that has received much attention in the past years. There are several reasons behind this:

- Rendering time is (sometimes) independent of the complexity of the scene/object rendered thus making it profitable for highly complex objects like trees [Nelson96].
- Faster than traditional rendering.
- 3D rendering from photos and movies.

Texture mapping is the most familiar image-based rendering technique. To create the illusion of fine details, an image or texture is mapped onto a three-dimensional surface thus reducing the scene complexity. Impostors share the same idea: parts of the scene or objects are replaced with some very simple textured geometry that retains the visual characteristics. This technique has been used to navigate in large environments [Maciel95] using pre-computed textures for each object. In [Shade96, Schau96] textures are generated dynamically to accelerate the walkthroughs of large environments. Impostors augmented with 3D information were also used to view architectural models [Sillon97] and accelerate the rendering of numerous virtual humans [Aubel98].

Recent work has focused on how to add depth information to achieve a correct visibility. [Schau97] proposed to directly modify the depth value for every texel. In [Schau98] the impostor is decomposed into several layers, each rendered with a different depth value. The approach in [Snyder98] is to detect potential occlusions between objects and create appropriate image layers.

From a hardware point of view, two systems dedicated to image-based rendering have been proposed [Regan94, Torbo96]. They exploit the same idea: parts of the frame buffer content are re-used over several frames thus avoiding rendering the whole scene each frame from scratch.

3. REPRESENTATION OF VIRTUAL HUMANS WITH IMPOSTORS

As traditionally the case in the existing literature, what we mean by impostor is a set of transparent polygons onto which we map meaningful opaque images. The number of polygons varies depending on what goal we want to achieve. If correct visibility is required a multi-plane impostor must be used. Otherwise, a

single-plane impostor is preferable because of greater speed.

3.1. The Virtual Human Model

Our real-time virtual human model consists of an invisible skeleton and a skin. The underlying skeleton is a hierarchy of joints that correspond to the real human main joints. Each joint has a set of degrees of freedom, rotation and/or translation, which are constrained to have authorized values based on real human mobility capabilities. The skin of our virtual human is generated by means of an in-house modeler: ellipsoidal metaballs and ellipsoids are employed to represent the gross shape of bone, muscle and fat tissue [Shen96]. Animating our virtual human model consists in updating the skeleton hierarchy including the global mobility joints (special joints used to position and orient the virtual human in the world). For each posture assumed by the skeleton, appropriate skin deformations are computed using a technique based on cross-sections [Shen96].

3.2. Representation with a Single Quadrilateral

Virtual humans can be accurately modeled using a single textured plane, in which case they are sometimes referred to as sprites. The quadrilateral is placed so that it faces the camera continuously and its texture is updated regularly enough for it not to be visually distracting. [Aubel98] showed how to generate textures representing virtual humans. A parallel projection can be used without causing too much infidelity.

3.2. Multi-plane Impostor

Another way is to decompose the virtual human into several planes that are rendered at different depth values. Our approach is to divide the virtual human into coherent parts, that is parts within which no overlapping can occur. Our virtual human is divided to this effect into the following seventeen parts: waist, torso, neck, head, upper arms, lower arms, upper legs, lower legs, hands and feet. Each body part is associated with a quadrilateral, which is placed in the scene so that it faces the viewer (fig. 1). Textures for each body part are generated dynamically in an off-screen buffer and copied to texture memory.

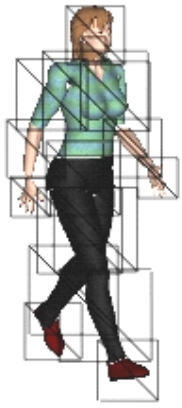
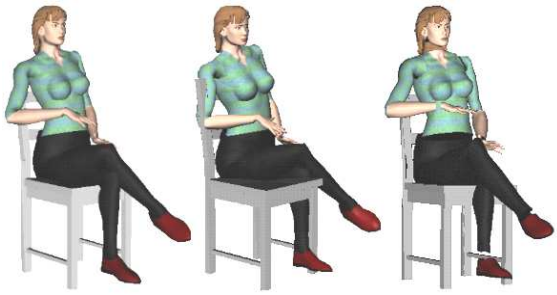


Figure 1: a multi-plane impostor

Our multi-plane impostor is still a draft model. For example, the texture refreshment algorithm presented below does not discriminate between the planes: either all sixteen textures are regenerated or none. A more advanced version should decide which textures must be refreshed depending on the motion. Besides, all quadrilaterals always remain upright for simplicity's sake. It would probably be profitable to authorize rotations of the quadrilaterals in the projection plane. Finally, there are cases for which the visibility is still incorrect e.g. long hair is likely to interfere with the neck (fig. 1,2).

3.3. Visibility Issue

Fig.2 shows that a single-plane sprite may produce an incorrect visibility whereas the multi-plane one does not.



Left: actual geometry; Middle: sprite; Right: multi-plane impostor.

Figure 2

3.4. Comparison

Both versions are equally fast to render because the number of pixel drawn is approximately the same. Texture storage requirements are also roughly the same. However, the multi-plane impostor is in practice slower because more calculations are required (e.g. because of skin deformations, the size of each body part changes and must be recomputed). In addition, the model looks better in our current system with a single sprite because body parts are more smoothly connected and no overlapping occurs.

4. CACHE INVALIDATION

For static objects camera motion is the one and only factor to take into consideration for cache invalidation. The algorithm that decides whether a snapshot of a virtual human is stale is obviously a bit more complex. Still, that algorithm has to execute very quickly because it must be performed for every virtual human. We rely primarily on two factors: self-deformation of the actor and its global motion with respect to the camera. Then, we explain how to take into account other factors to better fit simulations where lots of actors are involved.

4.1. Posture Change

The basic idea is that the viewer need not see every posture of the virtual humans to fully understand what they are doing. A few key postures are often meaningful enough. Our approach reflects this idea of sub-sampling the motion. Once we have updated the skeleton hierarchy we have direct access to joints' positions in world space. It is therefore quite straightforward to compute distances (or squared distances) between some particular joints. In concrete terms we compare four distances (fig. 3) with those stored when the texture was last generated. As soon as the variation exceeds a certain threshold the texture is to be re-generated. We found out that four tests suffice to reflect any significant change. Nevertheless, other tests should be performed if the simulation is meant to underscore some peculiar actions e.g. grasping of an object requires additional testing of the hand motion.

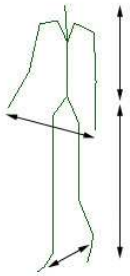


Figure 3: tests for posture change

4.2. Relative Motion of the Actor in the Camera's Frame

Instead of testing independently camera motion and actor's orientation, both are checked in a single test. This idea stems from the fact that every virtual human is always seen under a certain viewing angle which varies during simulation whether because of camera motion or actor's motion. Yet, there is basically no need to know what factor actually caused the variation.

In practice we start by computing a "view" matrix, which corresponds to the transformation under which the viewer sees the virtual human. It is plainly the product of the scene's camera matrix and

that of the articulated character. We then construct the transformation matrix **M** from the old view matrix to the new one, where the old view matrix corresponds to that computed when the texture was last generated. **M** being purely a rotation matrix, its equivalent axis-angle representation can be extracted (only the angle is needed afterwards). All that remains to be done is check the angle against a pre-defined threshold.

4.3. Cache Invalidation Mechanism

The two tests described above are applied on an individual basis. In a real simulation involving several actors however, restricting the cache invalidation mechanism to these two factors would yield a jerky animation. Put differently, the frame rate would be likely to be high on average but uneven. The idea for a global texture refreshment mechanism is to perform these two basic tests to cull very rapidly the actors that need not be refreshed. Then, the virtual humans are treated as a whole, which guarantees a quasi-constant frame rate among other things. The cache invalidation algorithm runs in pseudo-code as follows:

```
For every frame N>0    /* Simulation */
  For every virtual human
     Compute viewing angle variation
     Angle ratio = viewing angle variation /
     angle threshold
     Compute Posture Variation
     Posture ratio = posture variation / posture
     threshold
     Variation ratio = max (posture ratio *
     angle ratio, max (posture ratio, angle
     ratio))
     If (Variation ratio > 1)
       Insert character into list of "to be
       updated" characters
     End if
  End for
  K= Sort list of "to be updated"
  Refresh textures of first K actors
End for
```

Instead of refreshing an actor's texture at once if it must be, a pointer to the character is inserted into a list of candidates for refreshment. After every potential character has been inserted, a global weight used to sort the list is computed for each candidate according to several criteria. The first and most important factor is the variation ratio. Among other things the effect of the posture ratio and that of the angle ratio should be cumulated if both are greater than one. Second, the (normalized) distance from the virtual character to the viewer is taken into consideration so that distant characters and those on the periphery contribute less. Third, the rendering cost (based e.g. on the number of polygons) is normalized with respect to the maximal rendering cost of the candidates for refreshment. Simulation criteria may also be brought into play. For example, the application can keep track of the texture refreshments to avoid updating the same characters too frequently or assign priorities to the actors. To weigh the contribution of each factor, each criterion can be raised to a power proportional to its importance. Our choice is to square the variation ratio because it reflects how poorly the current texture represents the new position/posture of the virtual human. Once the list of candidates sorted, the first **K** actors are refreshed. We chose **K** so that the number of polygons to render does not exceed a predefined threshold. We did not set a minimum threshold but it could be done by checking in a first pass the size in terms of polygons of the list of candidates and adding, if necessary, other actors to the list.

5. INTEGRATION

Our work on impostors originated from previous research on human crowd control [Musse97], where it is desirable to be able to preview scenes quickly. The next section logically elaborates a bit more about what can be achieved in terms of number of virtual humans rendered. The

second one shows how to adapt the cache invalidation mechanism to the context of virtual environments.

5.1. Human Crowd Rendering

We simulated a circular wave motion propagating through a gathered crowd of 120 virtual humans (see fig. 4 or http://ligwww.epfl.ch/~aubel/crowd/ for an MPEG movie), each consisting of around 800 triangles (the actors' head, hands and feet are replaced with simple textured geometry). We set a fairly high angular threshold in order to avoid camera motion causing too frequent texture updates. By doing so we managed to view and navigate through the crowd at an interactive frame rate on a bi-R10000 Octane workstation. We achieved a steady rate of 24 frames a second by setting the posture variation threshold to 10 percent and allowing ten texture refreshments (about 8000 triangles) per frame at most. In this configuration there was no visible degradation of the animation. Setting the posture variation threshold to 30 percent increased the frame rate to 36Hz with very small loss of quality. By comparison, the frame rate dropped to 3Hz when the actual geometry was rendered instead.

Figure 4: crowd performing a wave motion

In this test-bed impostors perform so well as opposed to the real geometry because most characters are "idle" at any given time. However, this case is not specific. In fact most simulations involving virtual humans are likely to exhibit this fundamental pattern: most actors are either "idle" or walking. Since very few key postures are needed to create the illusion of walking, impostors turn out effective in most cases.

5.2. Collaborative Virtual Environments (CVEs)

Rendering humanoids in the context of CVEs differs in three respects. Firstly, the number of virtual humans on screen is much lower, typically up to a dozen in the same part of the world. Therefore, the user has more time to watch each character. Secondly, the virtual humans lie in general closer to the virtual camera than in crowd simulations, which means that insufficient texture refreshments are more obvious to the viewer. Lastly, participants often communicate through facial expressions and gestures. Because all this contributes to make animated impostors less suitable for rendering avatars, the texture refreshment mechanism must be tailored to the CVE.

Our approach is to maintain a list of update priorities of the avatars for each participant. The lists are constructed by defining a sphere of interest for each participant. Each sphere of interest encompasses all the other participants that are of some special interest and whose avatars need to be updated regularly. Whether a virtual human belongs to this sphere depends on four factors: field of view, distance, orientation and interaction. A character is discarded if it does not lie in the participant's field of view or if it is too far. Similarly, a virtual human with its back turned to the participant will have fewer chances of belonging to his/her sphere of interest. Finally, the character should be discarded if it does not try to communicate with the participant [Beche96], verbally or not, and if it is not engaged in a common activity. Two other factors we have not experimented with could be added: accounting for the guidance system of the avatar and distinguishing between autonomous characters and avatars.

6. CONCLUSION

This paper shows how to represent efficiently virtual humans with impostors and how to integrate them into applications. We proposed a global cache invalidation mechanism to that end that executes reasonably fast. In summary animated impostors provide an extremely powerful technique for the following reasons:

- Software technique that could yet benefit from a dedicated architecture.
- Usable with geometric LOD.
- Simulation considerations can easily be integrated (quasi-constant frame rate, priorities etc.)
- Texture memory consumption can be monitored and acted upon.

The weaknesses are the following:

- Usage of texture memory.
- Visibility not always entirely correct.
- Small animations (e.g. breathing, fidgeting) that improve the visual realism are sacrificed.

Four major issues could be addressed in future work:

1. The visibility problem has been only partially tackled in this paper. Our suggestion is to use a multi-plane representation when an actor is close to objects or other actors. Yet, our multi-plane representation is only a rough draft. In particular, issues such as smoother connections between body parts and overlapping parts remain to be addressed.
2. Investigation of lighting changes and shadows generation.
3. The current system only relies on posture information and therefore rules facial animation out. A more complete system would integrate changes in facial expressions, at least for close characters in CVEs.
4. As the number of articulated characters increases, so does the computation time in such a way that it may rapidly exceed the rendering time. A complete update of our virtual human model's hierarchy requires over a hundred matrix multiplications. A tighter coupling with level of detail in the animation is therefore needed to actually simulate great numbers of articulated characters.

ACKNOWLEDGEMENTS

This work was partly sponsored by the Federal Office of Education and Science in the framework of the European project eRENA.

REFERENCES

Aubel A., Boulic R., Thalmann D. 1998 *Animated Impostors for Real-time Display of Numerous Virtual Human*, Proceedings of Virtual Worlds '98, Paris, France, pp.14-28.

Becheiraz P., Thalmann D. 1996 *The Use of Nonverbal Communication Elements and Dynamic Interpersonal Relationship for Virtual Actors*, Proceedings of Computer Animation '96, pp.58-67.

Maciel P., Shirley P., 1995 *Visual Navigation of Large Environments using Textured Clusters in* 1995 Symposium on Interactive 3D Graphics, pp. 95–102

Musse S., Thalmann D., 1997 *A Model of Human Crowd Behaviour*, Computer Animation and Simulation '97, Budapest, Eurographics Workshop, Wien, Austria, pp.39-51.

Nelson M., 1996
Hierarchical Rendering of Trees from Precomputed Multi-Layer Z-Buffers, Proceedings of the 7th Eurographics Workshop on Rendering'96, Porto, Portugal, pp.165-174.

Pratt D., Pratt S., Barham P., Barker R., Waldrop M., Ehlert J., Chrislip C. 1997
Humans in Large-scale, Networked Virtual Environments, Presence, Vol.6, No.5, pp. 547-564.

Regan M., Post R., 1994
Priority Rendering with a Virtual Reality Address Recalculation Pipeline, SIGGRAPH '94 Proceedings, pp.155-162.

Schaufler G. 1996
Exploiting Frame to Frame Coherence in a Virtual Reality System, Proceedings of VRAIS'96, Santa Cruz, California, pp.95-102.

Schaufler G. 1997
Nailboards: A Rendering Primitive for Image Caching in Dynamic Scenes, Proceedings of 8th Eurographics Workshop'97 on Rendering, St. Etienne, France, pp. 151-162.

Schaufler G., 1998
Per-Object Image Warping with Layered Impostors, Proceedings of the 9th Eurographics Workshop on Rendering'98, Vienna, Austria, pp.145-156.

Shade J., Lichinski D., Salesin D., DeRose T., Snyder J., 1996
Hierarchical Image Caching for Accelerated Walthroughs of Complex Environments, SIGGRAPH'96 Proceedings, pp. 75-82.

Shen J., Chauvineau E., Thalmann D. 1996
Fast Realistic Human Body Deformations for Animation and VR Applications, Computer Graphics International'96, pp. 166-173.

Sillon F., Drettakis G., Bodelet B., 1997
Efficient Impostor Manipulation for Real-Time Visualization of Urban Scenery, Proceedings of Eurographics'97, pp. C-207 - C-217.

Snyder J., Lengyel J., 1998.
Visibility Sorting and Compositing without Splitting for Image Layer Decompositions, SIGGRAPHG'98 proceedings, pp.219-229.

Torborg J., Kajiya J., 1996
Talisman: Commodity Real-time 3D Graphics for the PC, SIGGRAPH '96 Proceedings, pp.353-363.