EUROGRAPHICS Workshop on ... (2013), pp. 1–3
N.N. and N.N. (Guest Editors)

Crowd Rendering with per-joint Impostors

A. Beacco, C. Andujar, N. Pelechano and B. Spanlang

MOVING Research Group, Universitat Politecnica de Catalunya, Spain

Abstract

In this poster we present two methods for rendering thousands of animated characters in real-time. We maximize rendering performance by using a collection of pre-computed impostors sampled from a discrete set of view directions. The first method is based on relief impostors [BSAP11] and the second one in flat impostors [BAPS12]. Our work differs from previous approaches on view-dependent impostors in that we use per-joint rather than per-character impostors. Characters are animated by applying the joint rotations directly to the impostors, instead of choosing a single impostor for the whole character from a set of predefined poses. This representation supports any arbitrary pose and thus the agent behavior is not constrained to a small collection of predefined clips. To the best of our knowledge, this is the first time a crowd rendering algorithm encompassing image-based performance, small GPU footprint and animation-independence is proposed.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Real-time rendering of detailed animated characters in crowd simulations is still a challenging problem in computer graphics. State of the art approaches can render up to several thousand by consuming most of the GPU resources, leaving little room for other GPU uses such as driving the crowd simulation. Polygonal meshes deformed through skinning in real time is suitable for simulations involving a relatively small number of agents, since the rendering cost of each animated character is proportional to the complexity of its polygonal representation.

A number of techniques have been proposed to accelerate the rendering of animated characters. Besides view-frustum and occlusion culling techniques, related work has focused mainly on providing level-of-detail (LOD) representations. Unfortunately, most surface simplification methods do not work well with dynamic articulated meshes. As a consequence, the simplified versions of each character are often created manually and they still suffer from a substantial loss of detail. Image-based precomputed impostors [TLC02] provide substantial speed improvements by rendering distant characters as a textured polygon, but suffer from two major limitations: all animation cycles have to be known in advance (and thus animation blending is not supported), and resulting textures are huge (As for each view angle and animation frame an image has to be stored). Using separate impostors for different body parts provides a more memory-efficient approach. We also benefit from the use of LOD by simplifying the input bone hierarchy grouping joints (letting some parent nodes absorb small child nodes). We created bone hierarchies with 21, 7, and 1 joint (no animation).

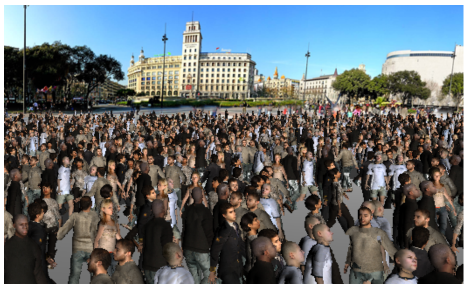
Figure 1: *Crowd rendered with per-joint relief impostors.*

Polypostors [KDC*08] use 2D polygonal impostors for each body part. Unfortunately, the animation sequence still has to be known at construction time, and the character decomposition is done manually. Our work aims at encompassing the performance benefits of view-dependent impostors with the flexibility of animation-independent approaches. On the one hand, view-dependent impostors minimize the geometry to be transformed as well as per-fragment computations, and thus achieve the maximum performance while minimizing the use of programmable hardware. On the other hand, animation-independent approaches are more flexible in terms of animation clips and animation blending. The two papers presented in this poster are based on per-joint impostors that are computed fully automatically.

2. Per-joint Relief Impostors

Each character is encoded through a small collection of textured boxes storing color and depth values (Figure 2). At runtime, each box is animated according to the rigid transformation of its associated bone and a fragment shader is used to recover the original geometry using a dual-depth version of relief mapping. This compact representation is able to recover high-frequency surface details and reproduces view-motion parallax effectively. It drastically reduces both the number of primitives being drawn and the number of bones influencing each primitive, at the expense of a very slight per-fragment overhead. We show that, beyond a certain distance threshold, our compact representation is much faster to render than traditional level-of-detail triangle meshes. The user study results carried out demonstrated that replacing polygonal geometry by per-joint relief impostors produces negligible visual artifacts [BSAP11].

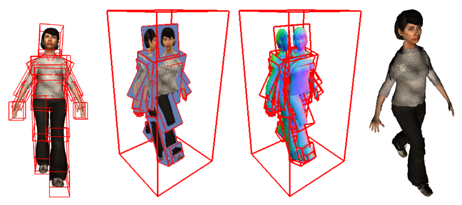

Figure 2: *During pre-process, color, normal and depth information is projected onto the 6 box faces. During real-time, each character is rendered through relief mapping.*

3. Per-joint Flat Impostors

Instead of using six orthogonal relief maps for each joint, which requires multiple dependent texture accesses per fragment, we use flat impostors created by sampling each joint from multiple view directions. We compute a spherical Voronoi map for the desired number of samples, and build a cube map by projecting the Voronoi cells onto the cube faces. At runtime, a single texture lookup is enough to retrieve the fragment color, which is one order of magnitude faster than relief mapping.

Since our impostors are intended to be valid for any pose, a key issue is to properly define which part of the geometry influenced by each joint must be represented as opaque pixels in the corresponding impostor (*mask*) (see Figure 3) .

We provide an efficient algorithm for computing optimized masks which considers how the geometry of each bone is affected by the transformation of neighboring joints (details can be found in [BAPS12]). This approach clearly outperforms competing animation-independent approaches for crowd rendering, being over 5 times faster than per-joint relief impostors .

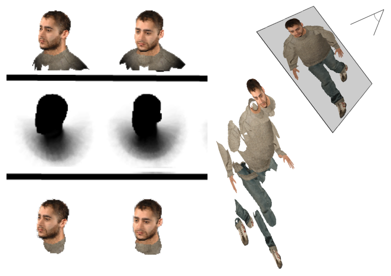

Figure 3: *On the left, pre-process to obtained each per-joint per-view impostor for the head of the character after applying the corresponding mask. On the right, real time rendering composing the per-joint textures for a given view point.*

4. Conclusions

Per-joint impostors allows us to render tens of thousands of characters in real-time. Encoding per-joint geometry and appearance with relief maps provides the highest image quality at the expense of a higher per-fragment overhead, which in practice limits their applicability to distant characters. View-depedent flat impostors are more demanding in terms of texture memory and construction time, but provide the highest runtime performance even for close-up characters. With properly choosen switch distances, both representations outperform polygonal meshes with negligible visual artifacts. Regardless of the particular encoding, per-joint impostors support arbitrary animation cycles and animation blending, a missing feature in competing per-character impostors.

References

[BAPS12] Beacco A., Andújar C., Pelechano N., Spanlang B.: Efficient rendering of animated characters through optimized per-joint impostors. *Journal of Computer Animation and Virtual Worlds 23*, 2 (2012), 33–47. 1, 2

[BSAP11] Beacco A., Spanlang B., Andujar C., Pelechano N.: A flexible approach for output-sensitive rendering of animated characters. *Computer Graphics Forum 30* (2011). 1, 2

[KDC*08] Kavan L., Dobbyn S., Collins S., Žára J., O'Sullivan C.: Polypostors: 2d polygonal impostors for 3d crowds. In *I3D '08: Proceedings of the 2008 symposium on Interactive 3D graphics and games* (New York, NY, USA, 2008), ACM, pp. 149–155. 1

[TLC02] Tecchia F., Loscos C., Chrysanthou Y.: Image-based crowd rendering. *IEEE Comput. Graph. Appl. 22*, 2 (2002), 36–43. 1