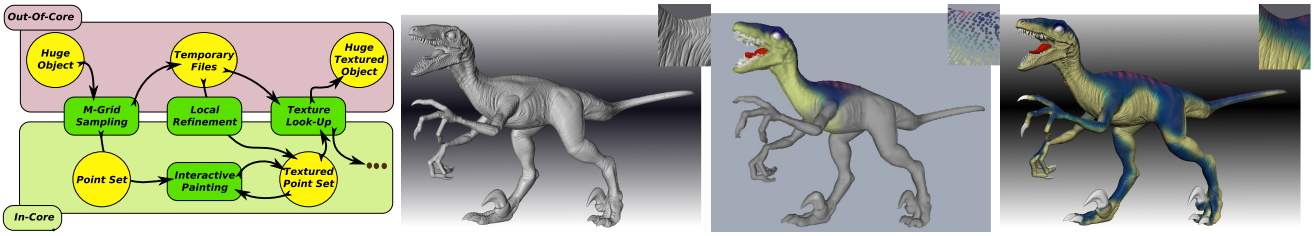
SIGGRAPH 2006 Sketch : Interactive Out-Of-Core Texturing

Tamy Boubekeur Christophe Schlick

LaBRI - INRIA - CNRS - University of Bordeaux*



(a) (b) (c) (d)

Figure 1: (a) General framework for interactive out-of-core texturing. (b) Initial untextured out-of-core mesh (8M polygons). (c) Interactive texturing of an in-core point-sampled approximated geometry (150k samples). (d) Point-sampled texture applied on the initial full-size mesh.

1 Introduction

Interactive rendering of huge objects becomes available on common workstations thanks to highly optimized data-structures and out-of-core frameworks for rendering. However, interactive editing, and in particular interactive texturing [Hanrahan and Haeberli 1990] of such objects, is still a challenging task, since the dynamic information added during this editing step would break any highly-optimized data-structures, such as GPU vertex buffers or specific out-of-core representations of huge objects. We propose *Point-Sampled Textures (PST)* for interactive texturing of large models at various scales without requiring 2D parameterization (complex and expensive for large models). This framework (see Figure 1(a)) allows the user to interactively set any appearance property of the original object, from per-sample color to complex BRDFs.

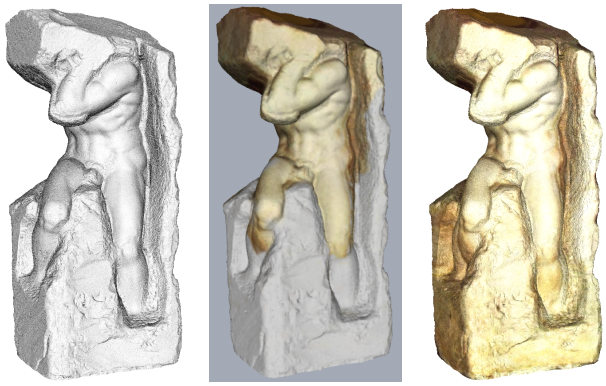
2 Texturing Framework

Out-Of-Core Sampling By performing an adaptive out-of-core point-sampling of the large object, we are able to represent arbitrary topology from various kind of surface definitions (e.g. meshes, point clouds), without dealing with any local structure-preserving operator. We propose a *constrained multi-grid* (or *m-grid*) approach combining the efficiency of regular grids with the adaptivity of octrees for fast out-of-core sampling. During the first step, a coarse grid is initialized and a streaming pass from the hard-drive is performed in order to evaluate the density of the input model. All points falling in a given cell are stored in its associated temporary file, useful for speeding up future queries. Then, a local grid is generated at each cell of the coarse one, whose resolution is set according to the cell density and the target user-defined resolution. Finally, a second streaming pass is performed through this *m*-grid, and one sample point is kept for each intersected local cell. The resulting Point-Based Surface (PBS) will be used as an intermediate representation for interactive texturing.

Interactive multiresolution texturing The in-core PBS (Figure 1(c)) can now be interactively textured using usual point-based texturing tools such as PointShop 3D [2002]. Multiresolution editing is simply enabled by implementing two additional features: first, user-controlled *refinement scheme*, that locally retrieves additional points from the original huge object, in the areas where the user wants to add some complex features, and second, an automatic *point-reduction scheme*, that locally simplifies the PBS by storing *Least Recently Used* local point sets on the hard-drive and replacing them by single samples (useful when the in-core PBS grows and becomes too large for interactive processing). These two operations are performed on a per-coarse-cell basis (first level of the *m*-grid). Each new point sample inserted in the active PBS is textured using the same point-based texture definition than for final back texturing. Here, the point-based representation makes straightfoward this local up-sampling/down-sampling in selected areas.

Back texturing Once textured, the in-core PBS is used to generate a Point-Sampled Texture (PST) which is a 3D continuous vector-field generated by using a *quadratic* weighted average of a local neighborhood over the PBS. The feature-preserving filtering is intuitively user-controlled by adjusting the size of local neighborhood and the shape of the quadratic kernel. Finally, the PST is applied on the original huge object in a streaming process.

This framework is easy to implement and provides a *size-insensitive* multiresolution texturing tool for huge objects entirely based on streaming processes (see Figure 1(d) and 2). It is particularly suitable when artists want to focus on the appearance of an automatically acquired (laser range scanners) complex 3D geometry. Compared to the widely used *octree textures* [Benson and Davis 2002], PST offer a much better local control, and do not suffer from directional artefacts that are common with axis-aligned data structures.



(a) 500M polygons (b) 278k samples (c) 500M textured pol.

Figure 2: Recoloring the Michelangelo's Atlas. (a) Original uncolored large mesh. (b) Interactive multi-scale texturing with our system, using several photos from the original statue and stone samples textures. (c) Application of the PST to the original large model and real-time out-of-core visualization.

References

BENSON, D., AND DAVIS, J. 2002. Octree textures. In *SIGGRAPH '02*, 785–790.

HANRAHAN, P., AND HAEBERLI, P. 1990. Direct wysiwyg painting and texturing on 3d shapes. In *SIGGRAPH '90*, 215–223.

ZWICKER, M., PAULY, M., KNOLL, O., AND GROSS, M. 2002. Pointshop 3d: An interactive system for point-based surface editing. In *SIGGRAPH '02*, 322–329.

*e-mail:{boubek,schlick}@labri.fr