View-Dependent Impostors for Architectural Shape Grammars

C. Jia, M. Roth, B. Kerbl, M. Wimmer

TU Wien, Institute of Visual Computing & Human-Centered Technology

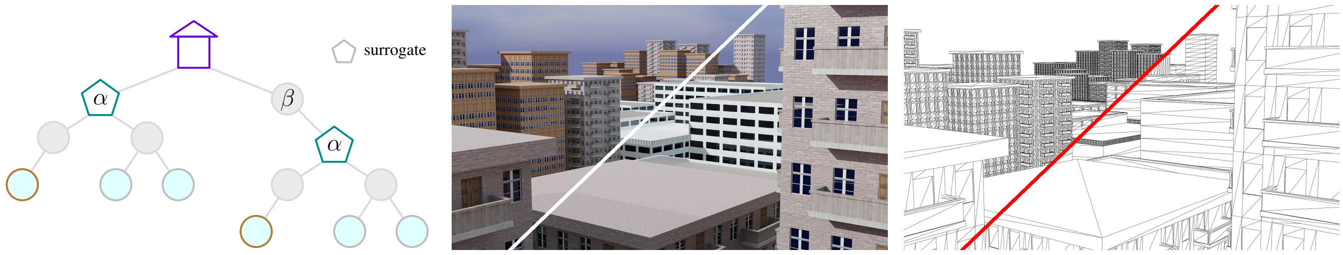


Figure 1: *(left) Our approach automatically identifies opportunities for placing surrogate rules (α) in shape grammar derivation trees, either directly or only for partial subtrees (β). At runtime, surrogate rules can evaluate to simple impostors that replace detailed geometry in procedurally generated scenes. (center) Our approach uses view-dependent factors to decide on the applicability of impostors, achieving high visual fidelity (original/ours). (right) Wireframes indicate the significantly reduced geometry in the rendered scene (original/ours).*

Abstract

Procedural generation has become a key component in satisfying a growing demand for ever-larger, highly detailed geometry in realistic, open-world games and simulations. In this paper, we present our work towards a new level-of-detail mechanism for procedural geometry shape grammars. Our approach automatically identifies and adds suitable surrogate rules to a shape grammar's derivation tree. Opportunities for surrogates are detected in a dedicated pre-processing stage. Where suitable, textured impostors are then used for rendering based on the current viewpoint at runtime. Our proposed methods generate simplified geometry with superior visual quality to the state-of-the-art and roughly the same rendering performance.

1. Introduction

As realism in computer games and movies rapidly advances, the demand for massive and detailed virtual environments has surged. It is becoming increasingly infeasible to create such compelling models within a reasonable amount of time and budget using the traditional process. Procedural approaches have found great use in automating the process of modeling scene geometry [**DBLP:journals/tog/WonkaWSR03**] and can easily generate vast, detailed environments. However, any geometry that is rendered in real-time must eventually be transferred to the Graphics Processing Unit (GPU), which, for potentially billions of triangles, incurs a significant overhead. Several approaches have been proposed to produce procedural geometry via so-called shape grammars in parallel, directly on the GPU, such as the parallel generation of architecture (PGA) system presented by **DBLP:journals/cgf/SteinbergerKKWS14** [**DBLP:journals/cgf/SteinbergerKKWS14**]. But even then, as **DBLP:journals/cgf/SteinbergerKKWS14** pointed out, further optimizations are required to relieve memory bottlenecks for highly detailed procedural models. The authors proposed using techniques such as visibility pruning and frame-to-frame coherence. They also suggested the use of level-of-detail (LOD) mechanisms as an important performance factor, but their solutions did not offer methods for exploiting LODs automatically. Their hand-crafted LOD models were further selected only based on viewing distance, thus ignoring orientation, projection and parallax effects. To address these issues, we propose an automatic, view-dependent impostor generation for shape grammars via *surrogate rules*.

2. Method

Our surrogate rules are capable of either evaluating the original shape grammar rules of their corresponding subtree or returning a simplified impostor instead (Figure 1). Starting from the root of the shape grammar's derivation tree, at each node α representing a rule, we check whether it is sensible to insert a surrogate rule. The policy for this decision is use case-dependent and made based on the rules and parameter ranges in α's subtree. We then evaluate the

© 2021 The Author(s)
Eurographics Proceedings © 2021 The Eurographics Association.

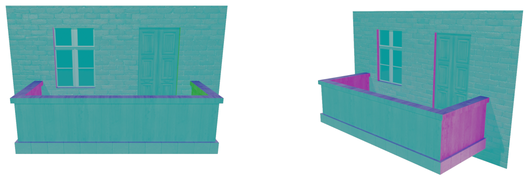


Figure 2: *A facade viewed from different perspectives. Parts that are invisible from one view direction become visible from another.*

grammar from α down to its leaves to generate the detailed geometry. To avoid redundant computation, we run a full evaluation of the whole derivation tree in one pass and store the generated geometry in global vertex and index buffers. From the detailed geometry, textures are created for each surrogate rule's impostor. We achieve this by rendering the detailed geometry with orthographic projection oriented towards the current rule's impostor shape, a fitted quadrilateral. Due to the parallax effect, one impostor texture made from a single view direction is very limited in capturing the appearance of detailed geometry, especially when it significantly extrudes out of the impostor shape (Figure 2). These errors may still be noticeable, even if viewed from a great distance. Therefore, we generate multiple textures for each impostor by rendering the corresponding detail geometry from several view directions into an *impostor grid* and defer selection of the impostor texture to runtime.

Given point Q on plane $\mathscr{P}$ and a point $P \notin \mathscr{P}$, we define

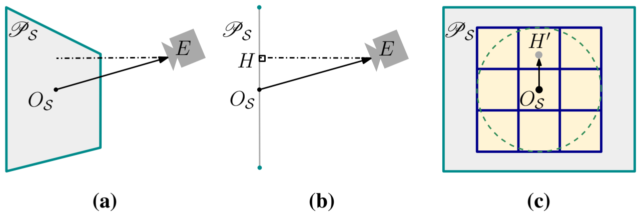
$$povi_{\mathscr{P},Q}(P) = \frac{\mathbf{QH}}{\|\mathbf{QP}\|}, \tag{1}$$

where H is the orthogonal projection of point P onto plane $\mathscr{P}$. For a given impostor shape $\mathcal{S}$, we define the *impostor plane* $\mathscr{P}_\mathcal{S}$ to be the plane in which $\mathcal{S}$ resides, and the *origin* $O_\mathcal{S}$ of plane $\mathscr{P}_\mathcal{S}$ to be the center of $\mathcal{S}$. For the camera position E, the *POV indicator* of E is computed as $\boldsymbol{\phi}_{\mathcal{S},E} = povi_{\mathscr{P}_\mathcal{S},O_\mathcal{S}}(E)$, which can be seen as the orthogonal projection of the *view vector* $\mathbf{O_\mathcal{S}E}$ onto plane $\mathscr{P}_\mathcal{S}$, normalized by the distance between the camera and the center of $\mathcal{S}$ (see Figure 3). Thus, $\boldsymbol{\phi}_{\mathcal{S},E}$ is independent of $\|\mathbf{O_\mathcal{S}E}\|$. We subdivide the unit square in the impostor plane $\mathscr{P}_\mathcal{S}$ into a $(2k+1)\times(2k+1)$ uniform grid, $k \geq 1$. Given a certain view distance d, for each grid cell with center O', corresponding camera position E' is computed:

$$E' = O_\mathcal{S} + d \cdot (\mathbf{O_\mathcal{S}O'} + \mathbf{n}_\mathcal{S} \cdot \sqrt{1 - \|\mathbf{O_\mathcal{S}O'}\|^2}),$$

where $\mathbf{n}_\mathcal{S}$ is the outgoing unit normal vector of $\mathcal{S}$. Obviously $\mathbf{O_\mathcal{S}O'}$ is the POV indicator of the camera positioned at E'. Then we render the detailed geometry for each surrogate from those camera positions and generate $(2k+1)^2$ textures from the rendered images.

Based on the implicit information stored in the impostor grid, we can now select view-dependent impostors at runtime. During the evaluation of the derivation tree when geometry is procedurally generated, we calculate the POV indicator at each added surrogate rule. The texture for the impostor is chosen based on which cell of the impostor grid the POV indicator lies in. Subtree rules will be fully evaluated if the POV indicator lies outside the impostor grid.



$\mathscr{P}_\mathcal{S}$
E
$O_\mathcal{S}$
$\mathscr{P}_\mathcal{S}$
H
E
$O_\mathcal{S}$
$\mathscr{P}_\mathcal{S}$
H'
$O_\mathcal{S}$
(a)
(b)
(c)

Figure 3: *(a) An impostor shape $\mathcal{S}$ centered at $O_\mathcal{S}$ resides in plane $\mathscr{P}_\mathcal{S}$, with a camera positioned at point E; (b) Vector $\mathbf{O_\mathcal{S}H}$ is the orthogonal projection of the view vector $\mathbf{O_\mathcal{S}E}$ onto $\mathscr{P}_\mathcal{S}$; (c) POV indicator $\mathbf{O_\mathcal{S}H'} = povi_{\mathscr{P}_\mathcal{S},O_\mathcal{S}}(E) = \mathbf{O_\mathcal{S}H}/\|\mathbf{O_\mathcal{S}E}\|$. Clearly, $\mathbf{O_\mathcal{S}H'}$ lies inside the unit circle centered at point $O_\mathcal{S}$.*

3. Preliminary Results and Future Work

To evaluate the overall performance of our method, we tested four static scenes of varying complexity (when fully evaluated): **balcony** (167.8M triangles), **town** (6.1M triangles, Figure 1), **dorms** (836.7K triangles) and **facade** (3.2M triangles). For these four scenes, we randomly sampled 647 camera configurations in total, with different positions and view directions. In our tests, we used a 3×3 impostor grid by setting $k = 1$, and the resolution of the impostor texture for each point of view is set to 128×128. Then we evaluated the unmodified grammar and modified grammars with adaptive level of details regarding performance and visual quality and compared to the PGA approach presented in [**DBLP:journals/cgf/SteinbergerKKWS14**]. With our method, we achieved better visual scores than PGA in all three test scenes (20% on average), according to the Butteraugli metric [**DBLP:journals/corr/AlakuijalaOSSVW17**]. Compared to the unmodified version, our method was able to eliminate 83% of triangles on average and achieve roughly the same rendering frame rate as **DBLP:journals/cgf/SteinbergerKKWS14**'s PGA. However, compared to **DBLP:journals/cgf/SteinbergerKKWS14**, our method may still generate about 5–20× as many geometry primitives. This seemingly drastic difference is easily explained by the fact that **DBLP:journals/cgf/SteinbergerKKWS14**'s LOD method depends solely on the view distance to determine the applicability of an impostor; hence, their approach trivially achieves a hugely beneficial side effect of occlusion culling. However, during the analysis of the derivation tree, we could easily bake outer and inner hulls of procedural geometry into our surrogate rules, which should enable us to apply proper occlusion culling and produce favorable results to the current state of the art and, alongside further improvements, bring our work to fruition.

Acknowledgements

This work was supported by the Research Cluster "Smart Communities and Technologies (Smart CT)" at TU Wien.