Paper Review

Procedural Modeling of Buildings

Yueyang Li

2024-01-30

1. Paper Title, Authors, and Affiliations

- **Title**: Procedural Modeling of Buildings
- **Year**: 2006
- **Authors**: Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, Luc Van Gool
- **Affiliations**: ETH Zürich, Arizona State University, K.U. Leuven

2. Main Contribution of the Paper

This paper presents CGA shape, an innovative shape grammar tailored for the procedural generation of architectural forms. The framework facilitates the creation of intricate building exteriors with rich geometric complexity and lifelike detail. It's key contributions includes:

- A procedural modeling technique that maintains structural coherence through flexible volumetric forms, regardless of orientation.
- Creation of Context-aware rules that govern dynamic relationships between elements like windows, doors, and terraces, enhancing architectural realism.
- A highly efficient algorithm in generating complex building geometry and facades, which is proving to support scalable and highly complex generation of city geometry.
- A framework that combines **mass modeling** and **facade modeling** for generating surprisingly detailed cities that overcomes the previous limitations of unsatisfied blocking and trimming issues.

3. Outline of Major Topics

1. **Introduction**
   (a) Procedural modeling is an efficient alternative to manual 3D modeling, reducing costs and time.
   (b) The proposed CGA shape grammar generates realistic building shells and facades.
   (c) It enables scalable and adaptive city modeling.
2. **CGA Shape Grammar**

   (a) Extends previous shape grammars (L-systems, Chomsky grammars).
   (b) Proposed a production rule that evolve a city iteratively by first Generate the mass model, then structure the facades, and add windows, doors, floors and ornaments.

3. **Mass Modeling**
   (a) Building masses are generated using scaling, translation, and splits.
   (b) Introduces roof structures, volumetric primitives, and rotations.
   (c) GIS data can be integrated for real-world footprints.

4. **Shape Rules and Further Refinement Details**
   (a) Introduces snap lines to align floors and facades.
   (b) Uses component splits to define edges, faces, vertices for refinement.

5. **Implementation of the CGA Shape Grammar**
   (a) Implementation of the grammar is through C++ based on City Engine Framework
   (b) Optimized data structures: Uses octrees for spatial queries.
   (c) Integration with RenderMan for high-detail rendering.

6. **Discussion and Result**
   (a) Strengths: Allows customization and rule reuse.
   (b) Strengths: Fast generation of massive urban models.
   (c) Weaknesses: Some generated configurations are unrealistic.
   (d) Weaknesses: Complex geometric details still require manual modeling.

4. Two Things Liked or Found Interesting

1. The procedural method enables the creation of billions of polygons in a single day, making large-scale city modeling practical and efficient using a generic way based on bounding volumes(mass).
2. The occlusion handling and snap-line-based alignment help maintain realism in the generated buildings by avoiding awkward placements of windows, doors, and balconies, which is not observed in previous works that do poorly using single extrusion and extension rules.

5. What Did You Not Like About the Paper?

- While the grammar rules ensure some realism, they do not fully model real-world structural stability, which might lead to implausible building structures in certain cases.
- Certain rules are still maintained manually, which establishes some learning curve for latent users. Furthermore, the generated buildings are not as diverse as human-generated ones, which might need further research and investigation.

6. Questions for the Authors

1. How does CGA Shape handle irregular building footprints in GIS data? Could Data-driven methods be utlized some way in creating trustworthy rules and grammars to facillitate this system?
2. Does the procedural method account for terrain variations (hills, slopes, or uneven ground), or does it assume a flat environment?