AN ARCHITECTURAL MODELING METHOD FOR GAME ENVIRONMENTS AND VISUALIZATION

CETIN TUKER¹ and HALIL ERHAN²

¹Mimar Sinan Fine Arts University, Istanbul, Turkey
cetintuker@gmail.com
²Simon Fraser University, Surrey, Canada
herhan@sfu.ca

Abstract. Modeling 3D architectural environments for games and design visualization is different than modelling for other purposes, such as for construction. These models include only the outer surfaces as ‘skin’ structures of the facades for rendering for which existing tools are too complex. After interviewing fourteen domain experts and evaluating available modelling tools, we observed a need for new modelling methods for rapid visualizations that leaves redundant model parts out for efficiency. We have developed a surface modelling method and a formalism for modelling architectural environments by slicing a building into layers with strips of façade element sequences. In the first prototype, we focused on parametric structures using user-defined architectonic vocabulary such as voids and solids. We conducted an expert review study with four participants: two user-experience and two domain experts. All participants responded that the method is easy to learn even for non-experts. Based on the tasks completed, they agreed that the method can speed the process of modelling large continuous façades, single-mass single-storey geometries, and repetitive floor layers; they also made suggestions for improvement. The results from the initial evaluation show that the method presented has some merits to be used in practice.

Keywords: 3D modelling; facade reconstruction; game; visualization.

1. Introduction

Creating digital models of buildings has become common practice for different purposes among which visualisation takes an important part. Models of architectural environments are efficient representations to communicate design ideas between architects, clients or other stakeholders. With the help of
photorealistic renderers, architectural environments can be visualized as still images or for animated walkthroughs. While architects use highly specialized tools for these purposes, game design engines and libraries are gradually taking their place in practice. With these systems, designers and other stakeholders can study the spatial characteristics of a design and details while roaming freely in the digital environment in real time.

Modeling 3D architectural environments for games and design visualization is essentially different than modeling for other purposes, such as for construction. Existing CAD tools are either semantic-modelers where each geometric element in a model has a corresponding meaning, or geometric modelers independent from what they refer to in design. BIM software can create precise and editable models but these models are heavy in detail and polygon count. In order to reduce polygon count—which is one of the main reasons of longer rendering times—these models are edited to include only the outer surfaces of the facades as ‘skins’. Other modeling tools—varying from CAD tools to pure 3D polygon, NURBS geometry modelers—are available for general purposes. Therefore they are equipped with low-level tools that lack specific features to create ‘skin’ facades. They also fall short in line-based modeling, which the modelers must create complex polygons or solids even for simple linear geometries. Furthermore editing these geometries with dimensional precision is required. Instead of parametrically defining architectonic elements, these tools create an additional task layer where designers should manage computer graphics. Some expert users improvise strategies for facade modeling, but this only increases the complexity of process by adding an extra task layer into the modeling workflow.

In this study we introduce a method that we believe can streamline creating geometric modeling of architectural environments for rapid visualization. We implemented a prototype system that adapts this method and we evaluated the prototype through expert-user interviews.

2. Motivation

During the first decade of 2000s, video games began to include highly realistic and large scaled 3D virtual urban and architectural environments. In these games, it is even possible for player to drive a virtual car or walk even for hours to experience the virtual environment without following the storyline. These are called as free roaming or open world game environments. These virtual cities include thousands of unique buildings that are designed and modeled to create a realistic experience of the city. Some of the well-known games with virtual cities are Grand Theft Auto (by Rockstar Games) and Mafia (by Illusion Softworks). During an interview, Aaron Garbut, the art
director of Grand Theft Auto, said: "We’ve simply not copied buildings around the map or procedurally generated the terrain to pad it out. It’s all handcrafted, all unique…" (Bernstein, 2013).

These virtual cities are designed to fulfill the narrative and artistic requirements of a particular scenario. Therefore they are art directed and modelled by expert 3D modelers manually and building-by-building. Procedural city modeling technique relies on an algorithm to generate a city from a pre-defined building topology. This makes the creation of environments very hard to be art directed. Therefore, the generated cities mostly share the visual monotonous look that all buildings share the same topologic family and have limited or no unique identity. Mass-modelling is not desirable.

After expert interviews and evaluation of available modeling software, we observed a need for rapid and more effective modeling methods for visualizations. In particular, we realise that digital architectural visualization have commonalities with virtual city creation for games. Visualizations are mostly presented in either rendered walkthrough animations or large size colour prints of still frame renderings. Rendering is essentially a very time consuming process requiring significant computational power particularly for polygon heavy models and animations. Rendering models with large polygon counts in real time is more critical for game engines. As improvisation, modelers manually remove polygons from the views.

Both game and visualization modelers use same type of software equipped with low-level modeling, photorealistic rendering, and animation tools. These tools require special training, where modelers must learn the vocabulary of computer graphics along with architecture. Therefore, training takes a long time and the learning curve is considerably steep.

Modeling process is iterative with many trial and error cycles. A tool to support this process must be flexible for editing any part of the model at any phase. For creating variations and reusing a model preferably we expect such tools provide some form of parametric features. Considering these observations, the method we present aims to improve modeling of architectural environments by using a simple parametric syntax to definite and edit building elements rapidly for mainly visualization purposes. We aim to better support the workflow in modeling game environment and architectural visualization.


3.1. AUTOMATED METHODS

Models of architectural environments are created using various input sources such as drawings, vector-based CAD models, and data from laser scans or
photographic images of facades from the existing buildings. An input source provides information on geometry, surface quality, openings, and other architectural clues. Below we discuss how different input sources determine selection of modeling methods and tools.

**Vector-based extrusion:** Yin et al. (2009) presents a survey of methods for generating 3D building models from both 2D architectural vector drawings and scanned images of architectural plans on paper. In the drawing-based modeling methods, models are created through automatic and semi-automatic extrusion of 2D plans from existing CAD drawings (Lewis & Séquin, 1998). The result is a 2½D surface polygon model that is highly dependent on the quality of the source, algorithms for finding and clearing errors and redundancies in the CAD drawings. Furthermore architectural symbols, dimension lines, text objects and hatched areas must be manually removed. During the process of converting scanned images of the floor plans to vector drawings, several image-processing methods must be used for noise removal, text extraction, vectorisation, and error cleaning.

**Image-based modeling:** The modelers scan and map architectural features from photographs of real scenes to the geometric models. Modeling process can be based on ordered or unordered image collections of inner spaces, facades or building masses. Klavdianos et al (2013) classifies these as techniques of photogrammetry. Lebegue and Aggarwal (1994) use a mobile robot with a camera mounted on top to capture image sequences from the inside of a structure and they create models by using these photographic sequences using image recognition or manual tools.

**Procedural modeling:** A set of rules defined and these rules are interpreted in a generative (rule-based) system to create various instances of the structure. L-Systems (Li, 2012), fractals (Wen, Hong, and Xia, 2009) and shape grammars (Stiny, 2006) are some examples of available procedural systems. Müller et al (2006) demonstrates a novel shape grammar to create procedural architectural shell (skin) models for computer games and movies. Their user groups are game designers and computer graphics professionals working for the movie industry. However, they give few suggestions on how their method can be used for architectural visualisation in architectural design firms. Although procedural systems can be efficient in creating multiple models based on rules in a short time, their use in design visualisation can be limited due to the cyclic and iterative nature of design process. It is hard to integrate changes and revisions both on the rules and models generated.

CAD drawings can vary in detail from layout of the plan to construction details. They give incomplete information about the building facades to generate an entire model. In addition, game designer has different needs and has no time to prepare CAD drawings before starting the 3D building model. In
addition a free roaming urban environment representation do not need interior plans of the buildings but do need a fast modeling tool which can create editable 3D building skins quickly. Although rule-based systems can generate building models quickly, the compilation of rules to create diverse set of models is a very challenging task. Besides, their use in a model of an existing urban environment is highly limited.

3.2. MANUAL MODELING METHODS

Since little research studied the manual methods for modeling for visualization and games, we interviewed fourteen field experts and studied the tutorials of commonly used software. Our goal was to identify common modeling methods and workflows in practice. Among the experts all were active 3D modelers for minimum five years. Six of them had background in architecture; two in industrial design; one in city planning; three game design; and two as visualization specialist. They currently work in different industries: architecture (6), game development (5), and 3D modeler (3). We asked them questions on how they create building models for visualization.

The modelers we interviewed use mainly facade and plan drawings to create a model. These can be vector-based 2D drawings on paper or hand drawn sketches. They mostly use general-purpose polygon modeling tools to transfer the models where representation of lines is not available and only accept polygon models. Their workflow focuses on recreating polygon geometries and exporting them into modeling tools to continue with line extrusion and further polygon modeling. If the floor plans are on paper, common practice is to scan the drawing and import the image into the modeling software to use as a template for polygon models. They complain about the loss of dimension precision when scanning drawings, which make creating a complete representation highly difficult. Therefore a model created as an end product of this workflow is a look-alike representation with geometric gaps. Since these polygonal models are not parametric, after many changes on the model, the modelers mentioned that they find it difficult and labour intensive to keep the model clean and precise.

In addition to extrusion of polygon models, some of the modelers we interviewed use push-pull methods and boolean operations as their favourite techniques to create openings on solid walls. These techniques have the same risk as polygon modeling on keeping the dimensions precise. Both methods have a tendency to create unnecessarily slim triangles that can lower the surface quality in complex geometries.

The architects use BIM for visualization because they are already created for construction-focused building models. They are efficient in boolean op-
erations, use architectonic vocabulary, enable precision and have libraries for building components. However BIM software creates models with unnecessary details for visualization and the model geometry is not easily scalable.

3.3. USING GAME ENGINES FOR VISUALISATION

Using game engines for architectural representations is a new method as game engines can introduce interactivity and free roaming of users (viewers) in a virtual site. Fritsch (2004) shows how game engines can be used to create interactive 3D applications for both indoor visualisation and outdoor visualisation using texture mapped geospatial data. Yan et al. (2011) demonstrate a BIM - Game prototype that integrates BIM and gaming tools into architectural visualization. Although their work is focused on indoor, it presents a framework for integrating BIM and games with a simple example of real time walkthrough, to use computer games in education.

4. Proposed Method

4.1. APPROACH

In this paper, we present a method for building modeling for visualization. A building mass can be sliced into horizontal layers and divisions defining the borders between these layers. For example, the building in Figure 1 (Left) can be decomposed into three sections: a base, a cluster of repeating floors, and an entablature. The façade surfaces on every layer is modelled as a sequence of parametric structures of solids $s$ (e.g. walls) and voids $v$ (e.g. windows), and turn angles $r$. Hence a layer essentially is a 2½D model of each floor that stacking with other layers forms a complete 3D model. Only information needed to build such models is the angles between different façade orientation, lengths and heights of the elements. Most buildings can be modelled by parametric definition of $s$, $v$, and $r$ and their sequences.

![Figure 1. (Left) Horizontal layers can be stacked in modeling. (Right) Decomposition of a façade and its expression in a sequence of solids and voids.](image-url)
This simple approach yields to a highly simple representation of a building that is complete enough for games and architectural renderings. The approach described is different from polygon modeling as (a) preparing vector plan drawings is omitted, and (b) creating parametric façade elements potentially can reduce the effort needed to complete repetitive buildings features.

Figure 1 (Right) shows an example of a layer cut out as a strip from the building façade. The origin is indicated by an anchor or turn location. The solid and void elements are outlined to define sequencing in a linear structure. For realistic rendering, texture mapping can be applied at solid or void level, or on the complete façade layer. The border elements can also be added to define multiple horizontal blocks as part of the same layer. The vertical surfaces are automatically created to complete the building envelope or manually added to define variations. The formal representation of the method is presented in Table 1, which can produce a complete scriptable model.

Table 1. Formal representation of the method in Extended Backus Naur Form syntax.

| Building: B | $B = \text{Origin}, \ (\text{Layer} \ | \ \text{Border} \ | \ \text{Cluster});$ |
| Origin: O | $O = %d \ x, %d \ y, [%d \ z];$ |
| Layer: L | $L = \text{Façade}, %d \ \text{height}, [O];$ |
| Façade: F | $F = \{s \ | \ v \ | \ r\}, [%d \ \text{overridenHeight}];$ |
| Solid: s | $s = %d \ \text{length}, [%d \ \text{overridenHeight}];$ |
| Void: v | $v = %d \ \text{length}, [%d \ \text{baseLength}],[%d \ \text{baseHeight}];$ |
| Rotation: r | $r = %d \ \text{rotationAngle}? (* 0 \leq r \leq 360*);$ |
| Border: D | $D = \{s \ | \ r\}, %d \ \text{height};$ |
| Cluster: C | $C = %n \ %l? \ (*L \ is \ unique \ and \ n: \ repetition*);$ |

The proposed method is analogous to imperative programming which is different than rule-based methods: we define the geometry in individual steps, loops, and conditionals rather than generic rules.

4.2. PROTOTYPE:

We developed an interactive prototype adapting the method and formalism we described above. The tool defines solid and void elements in strips as parametric nodes. Each node has its parameters and location relative to the preceding node. We strictly limit direct manipulation to selection and low-level editing of lines to focus on the modeling method only in this phase. Figure 2 (Left) shows a simple two-story building where its origin is indicated by "start or S". The building is decomposed into two different layers: an L-shape base layer and a second layer as a box. The base layer has six different façades identified by five 90° turns and a series of solids and voids.
Figure 2. (Left) An example building and its corresponding façade mapping where s: solid, v: void, and \( r \in \{90, -90\} \) rotation angle. Façade A can be written as: \( A = \{svsvsvsvs\} \), and the first floor \( F1 = \{svsvsvsvs, r90, svsvsvsvs, r90, svsvsvs, r90, svsvsvs, r90, svsvsvsvsvs\} \). (Right) A snapshot from the prototype interface.

The prototype is mainly used to evaluate if and how the method is suitable for modeling architectural environments for visualization. It includes a 3D model-view area, strip-skin input and editing area (construction line) made up of nodes, a properties panel and a tool bar (Figure 2 Right). Although we envision a scripting area, it is not active in this version.

5. Evaluation of the Proposed Method

5.1. DESIGN AND PROCEDURE

We conducted an expert review study with four participants: two user experience (UX) and two domain experts. The UX experts have academic degrees in architecture and CAD. Although they are not expert modelers, they have experience on developing different tools for modeling. One of the domain experts was a 3D generalist in the entertainment and game industry and the other is a designer with extensive knowledge in 3D modeling. All experts have experience with polygon or NURBS based freeform modeling.

The user study is composed of three sessions. In the first session, the moderator introduced the method and the basics of the prototype. After this session, the users were given four consecutive tasks that required them to model four buildings with growing complexity levels (Figure 3). The easiest task is to model a single-storey building. The fourth task was to model a two-storey building with three separate columns supporting the second-storey. The moderator described the tasks and presented the drawings of the buildings with dimensions on paper. The participants were asked to write their reaction to, experiences with, and suggestions for the method and the approach on the given handouts. We asked them to ignore the particulars of
the UI as much as possible. In this session, we recorded the computer screen and the participants’ comments as they completed the tasks.

During the study, the experts were able to complete all given tasks. Finally, the experts were encouraged to discuss their impressions of the method and the prototype in an open-ended question and answer session.

5.2. RESULTS
All participants responded that the method is easy to learn even for the non-experts. Based on the tasks completed, they agreed that the method can speed the process of modeling large continuous façades, single-mass single-storey geometries, and repetitive floor layers. The experts found modeling of discrete faces or objects (e.g. a column) as challenging. They agreed that the parametric node structure enabled them to edit the models locally and propagate the changes to all related parts.

The participants made some suggestions for improvement. One of them is to create a repository of recurring objects that can be selected and used repeatedly. They can be easily inserted in the sequence script by name or id. We strictly omitted direct manipulation except for object selection. The participants reported that when starting a new floor on an existing floor, calculating offset values relative to the previous floor is a difficult task. They suggested some kind of snap or align function to locate anchor points. As users appreciated the horizontal strips, particularly they commented that the composition of multi-story spanning façade elements would be hard to model. They suggested a feature to create vertical strips spanning multiple layers.

The results from the initial evaluation show that the method presented here has some merits to be used in practice. To further understand the approach, the second phase of evaluation will include multiple real-life users working on real-world modeling tasks on the next version of the prototype.

6. Conclusions and Future Work
We have developed a surface modeling method and formalism for modelling architectural environments by slicing a building into layers with strips of
parametric node sequences. In the first prototype, we focused on a nodal parametric structures using user-defined architectonic vocabulary. Reducing direct manipulation on the model, we aim for a fast and easy to learn parametric modelling that is extendable with simple scripting language based on the formalism presented. We evaluated the method through an expert user study revealing interesting advantages and opportunities for our future work.

The method is currently limited to creating flat surfaces perpendicular to the horizontal model surface. Although we can model large and repetitive façades, to build more complex forms our formalism must extend to include definition of basic geometry such curves. In addition buildings with multiple masses like detached columns on the same floor our method, formalism, and prototype need further development. We believe that adding a scripting feature for direct coding of parametric structures can enhance the modeling experience. Functions for identifying the origin of a geometric element or a point to resume modeling will be significantly helpful.

Acknowledgment

This research has been partially supported by the Council of Higher Education, Turkey, and Simon Fraser University Computational Design Lab.

References


Klavdianos, P.; Zhang, Q., and Izquierdo, E.:2013, A concise survey for 3D reconstruction of building façades. In *the Proceeding of the 14th International Workshop on Image Analysis for Multimedia Interactive Services* (pp. 1-4)


