Roll Your Own City [*]

Graham Whelan
Department of Informatics
Institute of Technology
Blanchardstown
Dublin, Ireland
graham.whelan@itb.ie

George Kelly
Department of Informatics
Institute of Technology
Blanchardstown
Dublin, Ireland
kellygp@gmail.com

Hugh McCabe
Department of Informatics
Institute of Technology
Blanchardstown
Dublin, Ireland
hugh.mccabe@itb.ie

ABSTRACT

The creation of detailed virtual urban environments is an increasingly common task being faced by both the film and games industries. It is a time-consuming and expensive process that requires artists to model vast amounts of geometric detail, including terrain, roads, buildings and other associated features. We demonstrate a system that allows an artist to rapidly generate 3D cityscapes of arbitrary size, complexity and layout. This is achieved by means of an interactive interface that enables close control over key parameters while the system procedurally generates the required detail in real time. The results can be viewed and edited or exported in a format suitable for use in 3D modelling packages and game engines.

Categories and Subject Descriptors

I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling; I.3.7 [**Computer Graphics**]: Three Dimensional Graphics and Realism; I.3.7 [**Simulation and Modeling**]: Applications; J.6 [**Computer-Aided Engineering**]: Computer-aided design (CAD)

General Terms

Algorithms

Keywords

Procedural Modelling, City Generation, Building Generation, Computer-Aided Design

1. DESCRIPTION

Our aim in designing this work was to create a system that enables the generation of cityscapes in an easy and intuitive manner and that therefore facilitates and encourages experimentation with the notion of creating imagined virtual cities. Our system is inspired by existing procedural city generation systems[2] but these systems tend to base the generation process on statistical input, such as population density maps, that may not be readily available. Our system is unique in that it requires little more of the user than to boot it up and start clicking with the mouse. The graphical interface allows for the direct manipulation of structural elements such as road intersections and also allows the user to alter the various parameters that control the procedural generation of both the roads and the buildings.

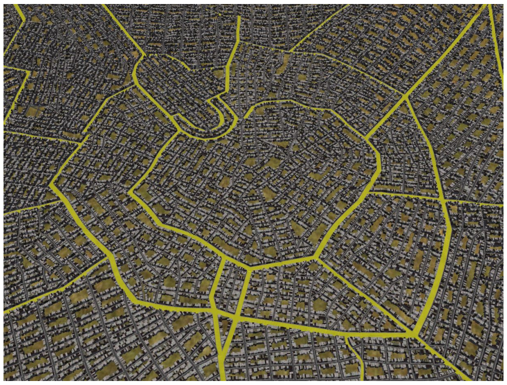

Figure 1: An example of a cityscape generated using Citygen.

Our work builds upon *Citygen* [1] as part of our ongoing research. Citygen is an application that employs procedural techniques for the automatic generation of large scale road networks and cityscapes. We extend this work through the addition of a procedural building generation framework which provides the ability to create buildings in greater levels of detail and in any number of different architectural styles. By incorporating this new framework within Citygen we are able to produce virtual city models with a far greater degree of realism.

The process commences with the user importing a predefined terrain model upon which the city will be built. The central idea is that cities are created around the primary road network which has a strong influence on city character and defines the constituent elements of city structure such as neighbourhoods. Therefore, the first stage involves the user creating this network by clicking points on the 3D terrain which then become major road intersections of the city.

[*]Futher details regarding this work can be found at `http://www.gamesitb.com/city.html`

DIMEA '08, September 10-12, 2008, Athens, Greece.
Copyright 2008 ACM 978-1-60558-248-1/08/09... $5.00

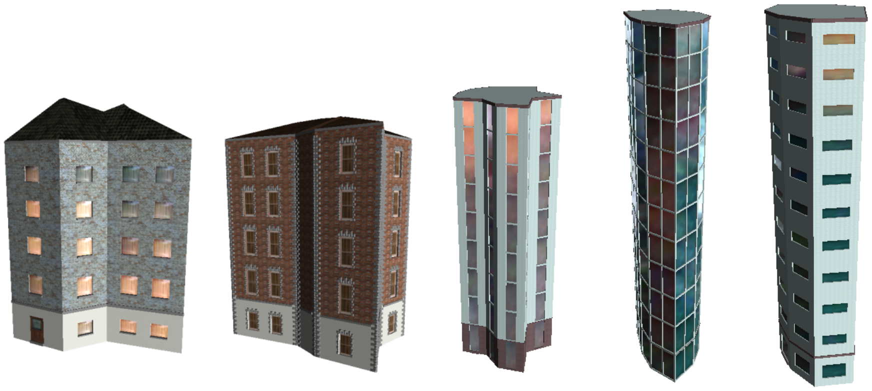

Figure 2: These buildings were created using our procedural building generation framework.

These intersections can be moved around at will to achieve different city layouts and the system will automatically generate primary roads, mapped to the underlying terrain, that connect these intersections in a realistic and cohesive manner. The sections of terrain enclosed by primary roads form neighbourhoods which we refer to as *city cells*.

These city cells are then automatically populated with secondary roads. Secondary road generation is accomplished using a growth based algorithm similar to L-systems, a technique often employed in the simulation of natural phenomena. This growth process begins at the borders of the city cells and proceeds inwards in a parallel fashion. We incorporate a snapping algorithm that tests the proximity of proposed new secondary road segments to the existing segments and intersections. In this way we can modify the proposed segments based upon a set of control parameters which can be specified on a cell by cell basis. These control parameters can be chosen to achieve different types of secondary road networks e.g. grid-like, meandering and so on. This has the effect of providing variety across the city and allows fine-grained control over the generation of each neighbourhood.

The areas enclosed by secondary roads form lots, upon which buildings are then inserted. The user can adjust various parameters to control the size and shape of these lots which in turn influences what type of buildings the system will create. In order to generate detailed building models we have implemented a grammar-based system[3]. This allows the construction of a building to be defined in terms of ordered rule sets that operate on some initial parameters. These parameters provide information such as the size and shape of the building footprint, the desired height of the building, what textures to apply, and values to control the generation of other features such as windows, doors, roofs etc. The result of this approach is that the system can generate an endless variety of different buildings by simply altering the initial parameters. Different rule sets define different architectural styles and advanced users can create their own styles and add them to the system. Users can also modify existing rule sets in order to produce variations on an existing style, for instance, rewriting the rule that controls window generation to create a building with customised windows.

Although our system provides a broad interface for user interaction, by exposing many of the underlying parameters which are used to control and manipulate the generation process, there is no requirement on the user to specify each and every one. The system provides a series of templates containing default values for certain parameters. These templates encapsulate a particular design or style and allow the user to quickly generate useful content. For example, we can provide a template describing a particular architectural style. A user can select this template and quickly generate a building in that style. If the user is not entirely satisfied with the outcome, they can select a different template or modify the control parameters directly in order to produce the desired result. Templates are also provided to control other aspects of city generation such as the creation of secondary roads within city cells. The provision of these templates enables us to create a city in a completely automated fashion by allowing the application to automatically select the series of templates to use.

The system is implemented as a C++ program that uses the OGRE open source game engine for high-quality rendering of the results in real time. It can also export the created city in the COLLADA format which facilitates import into many game engines or 3D modelling packages for further refinement. We believe it is both a useful tool for 3D artists who need to rapidly create urban environments and a compelling and engaging experience for the user that wants to experiment with creating their own city.

2. REFERENCES

[1] G. Kelly and H. McCabe. Citygen: An interactive system for procedural city generation. In *Fifth International Conference on Game Design and Technology*, pages 8–16, 2007.

[2] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. V. Gool. Procedural modeling of buildings. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 614–623, New York, NY, USA, 2006. ACM Press.

[3] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky. Instant architecture. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 669–677, New York, NY, USA, 2003. ACM Press.