Inverse design of urban procedural models

Carlos Vanegas

How to quote this text: Vanegas, C. Inverse Design of Urban Procedural Models. *V!RUS,* 11. [e-journal] Available at: <http://www.nomads.usp.br/virus/virus11/?sec=5> [Accessed dd mm yyyy].

Carlos Vanegas is the Chief Technology Officer and a co-founder at Synthicity, where he leads the engineering team in the development of software tools for 3D urban planning and design. His work is focused on concurrent behavioral and geometric methods for fast design, editing and visualization of 3D urban spaces.

KEYWORDS
parametrization, v!11, urban planning, modeling, public space.

VIDEO: https://www.youtube.com/watch?v=zOkWmwD3OV0

How are 3D models of planned cities typically created? The most common approach is to model by hand every object in the planned development, including streets, lots, buildings and facades. This approach is as ubiquitous as it is inefficient. As a response to the limitations of manual modeling, procedural urban modeling is becoming increasingly popular. Procedural urban modeling refers to a set of techniques that are used to programmatically generate 3D geometric models of urban spaces. The most typical techniques use shape rules that are sequentially applied to starting shapes in order to generate increasingly complex derived shapes. A key basis for the popularity of city-scale urban procedural modeling is that it enables creating sets of rules that encapsulate the complex interdependencies within realistic urban spaces. Users who may or may not be aware of the internal details of the procedural model itself can work with procedural tools to quickly create large, complex 3D city models.

A common property of many procedural methods is that a small set of succinct input rules and input parameters can yield very complex and coherent outputs. While this constitutes an advantage in terms of efficiency and succinctness, it is also a significant limitation in terms of controllability. Obtaining a 3D urban model with complex user requirements is a challenging task that requires experience and in depth knowledge of the underlying procedural model. Imagine as an example that an area of a city comprising a few dozen blocks will be redeveloped and you are asked to create a master plan for the site. The requirements for the new site include locating each new home within at most 1 kilometer from the closest park, ensuring that a landmark in the downtown of the city is visible from the top floors of at least half of the new buildings, a minimum number of residential units, and observing maximum

V!RUS 11
It's parametrization, baby!
revista do nomads.usp | nomads.usp journal
issn 2175-974x | CC BY-NC
www.nomads.usp.br/virus| vnomads@sc.usp.br

floor-to-area ratios (total area of a building divided by the total area of the lot the building is located on). How can a designer use procedural tools to create a model that meets such requirements?

Solving this problem with a traditional manual modeling approach would be prohibitively expensive and inefficient. Using procedural tools is likely a step in the right direction, but can still be a significant burden. The task involves multiple iterations in which a user must manually try many combinations of input parameters, observe the resulting 3D model, and adjust the parameters and rules as an attempt to produce a model that is closer to the requirements. What if instead of trying all these combinations manually we let the computer explore the vast space of combinations and find the ones that better satisfy our design goals?

In our paper 'Inverse Design of Urban Procedural Models', presented at SIGGRAPH Asia 2012, we propose a framework that helps the user discover how to alter the parameters of an urban procedural model so as to produce a desired 3D output. Our framework combines a forward procedural urban modeling process with an inverse urban modeling algorithm that optimizes the input parameters to satisfy user-specified goals. Both forward and inverse modeling strategies can be applied during an interactive editing session for either local or global model modifications.

Our approach consists of a forward procedural modeling engine that given a set of input parameters produces a 3D model, a set of tools to measure indicators on the 3D model, and a tool to quickly sample a solution space in search of combinations that generate models exhibiting desired indicator values. Indicators can be thought of as an additional output layer that can vary from simple evaluation of geometric properties to complex semantic metrics. The average distance of a house from the street and the floor-to-area ratio, are examples of simple indicators. Visibility of a landmark structure, the amount of sun exposure per building, and suitability of roofs for solar energy panels are examples of more complex indicators that do not have an obvious relationship to input parameters.

Our approach to sampling a solution space combines optimization with resilient back propagation. The optimization consists in simultaneously running many sets of Markov Chains over a large number of iterations and choosing the best solution states. The indicator values for a state, or combination of input parameters, are computed in each iteration. With the goal of allowing the MCMC engine to evaluate the target indicators for a large number of iterations and instances of urban models, we use a resilient back-propagation engine to imitate the indicator behavior of the urban procedural model without having to explicitly produce a 3D model.

We have used our framework to create and edit a variety of city-scale 3D models. All editing and rendering is done interactively using sliders to alter parameter values (forward modeling) or indicator values (inverse modeling). All example sessions were completed in under 5 minutes and most took less than one minute. Results of these models are shown in our SIGGRAPH Asia 2012 paper and video.