Interactive Tool for Procedural City Generation in Houdini
Master Thesis

Soham Ramteke
NCCA, Bournemouth University

August 19, 2011

Contents

List of Figures

Abstract

In this thesis an interactive tool for procedural city creation is presented which involves different procedural content creation techniques used to generate various aspects of a city like terrain, roads, buildings, etc. separately which complement and affect one another for the final creation of the city which is further customizable by the user with the help of an interactive interface that lets the user have total control over the look and development of the city.

Chapter 1

Introduction

Now a days procedural content has been used extensively in the films, commercials, animation and games industry which arose the need to find more and more ways to recreate the natural phenomenon procedurally yet making it look realistic and convincingly believable. There is no doubt that something that is created procedurally may not have the same level of detail as something that is created manually by the artists but it is also true that creating something procedurally is a much more efficient way for content generation and as the complexity of the content increases the creation time and the cost of creation increases as well. In such situations, adopting the procedural content creation approach will allow the artists to use the time in polishing and improving the areas of higher important and the areas of lower importance can be created procedurally which eventually results to be more cost effective as well. This can be further optimized by giving high level of artistic control to the user for all the content that is generated procedurally minimising the need for the artists to create anything manually and so increasing the productivity of procedural content creation as well.

In this project Houdini Object Model (HOM), an application program interface (API) that allows the user get information from and control Houdini using Python scripting language is used along with many different powerful nodes available in Houdini. Houdini was chosen to create the tool because of its easy and user-friendly node-based interface and the availability of numerous powerful nodes which can be explained as node-based representation of most of the real-life calculation and computations which can be networked together to represent an algorithm and to extend the control for further complexity each node can be edited and embedded with Python or Hsrcipt scripting language which gives the user much higher control for procedural content creation and also debugging the errors in the system is fairly easy due to the network based view of all the nodes which reflects a warning or error on the node if there is a possibility of mismatch of attributes to go on to the further nodes or if the user may have incorrectly typed in any part of script in the expressions and this helps to know where exactly in the network is the bug and can be debugged. Houdini also has a powerful feature that allows the user to compress everything in the network into one single node (Houdini Digital Asset) that can have a Graphical User Interface (GUI) which gives you the control to access the features of the HDA without the need to go inside the network. The HDA can then be passed on to anyone to recreate one or more instances of the tool for developing or improvement.

Chapter 2

Previous Work

Procedural generation can be explained in simple words as to automate the process of creating geometries and textures using a set of computer instructions. This phenomenon is not new to computer graphics. It has been used for over 2 decades now for creating the textures of a more natural appearance by a combination of different levels of Perlin Noise [Perlin 1985]. For a cellular patterns like cracked or dry ground or skin cells Voronoi Diagrams were used. Fractals and Lindenmayer-Systems (L-Systems) were most commonly used for generating objects like trees and foliage. In 1990 A. Lindenmayer and P. Prusinkiewicz had released a book titled *"Algorithmic Beauty of plants"* which introduced a system of generating plants based on rewriting system which basically accepted a rule or grammar for initial modelling and the iterating over the grammar for further modeling of the tree. L-System later on became very common to be used for many different types of the procedural content creation[Prusinkiewicz and Lindenmayer 1990]. It has recently been extended to create more complex natural objects like trees, clouds, terrains, etc. [Ebert et al. 2003]. Nearly a decade ago the first version of a real-time tree generation system called SpeedTree RT was released by Interactive Data Visualization Inc. and since has been used by many games studios and in films. Most recently it was extensively praised for and used in Avatar and the game The Witcher 2: Assassins of Kings [IDA Inc. 2011].

"Procedural Modeling of Cities" presented by Yoah I H Parish and Pascal Muller at ACM SIGGRAPH 2001 described a system titled "CityEngine" which procedurally generates an entire city including procedurally created buildings and the city layout based on the data provided by the user in the form of image maps. User provides the information like street map, height map, vegetation map, population density map , etc. which are collectively used for the formation of city layout that is then populated with the roads and building which are again procedurally generated using two different types of L-System which user can modify to vary the buildings and road formation respectively. CityEngine got very popular with time and has been widely used in the films and commercials. Most recently it was used to recreate the London city model procedurally which

Figure 2.1: SpeedTree Cinema

Figure 2.2: CityEngine in CARS 2

was used in a car chase scene in the film CARS 2 [PROCEDURAL Inc. 2011].

In the paper "*CityGen: An interactive system for procedural city generation*" presented at the Fifth International Conference on Game Design and Technology in 2007 by Kelly and McCabe describes an interactive stand alone application which generates a procedural urban city in three step process. Primary Road Generation which uses sampling technique to generate the road path which is again subdivided in the second step and in the third step Split-Grammar concept is used to generate the building and populate the city.

Procedural Modeling is also used in a different and entertaining way in the form of city-building simulation games. Where instead of letting the algorithm compute everything from the city layout to the positioning of buildings and roads, etc. user can have a range of city creation elements to select from where user can use an interface to create and manipulate the city pattern, road layout, building positions, house positions, tree positions and any other elements available for the city. This makes this type of procedural creation of city more interesting to the user as although most of the content is managed algorithmi-

Figure 2.3: Cities in Motion

cally inside the game network like depending on the modification of the terrain allowing the user to put city elements like road and buildings accordingly and restrict from the areas where elements can not be positioned and so on. This is not a very new concept. It has been used in games like SIMCITY which was first released in 1989 and has grown in popular since. In April 2011, Paradox Interactive released a building simulation game titled Cities In Motion which uses the same technique. Additionally the user has implement and improve a public transport system in European cities. Here, most of the computations are done procedurally yet the user has very high level of control for the creation of the city which makes the game-play interesting and entertaining [PARADOX INTERACTIVE 2011].

Chapter 3

Technical Background

In this chapter different procedural content generation techniques and their application in the computer graphics is discussed briefly and how a combination of two or more techniques can be more efficient for achieving realism and making the the creation process more interesting is explained. The most commonly used techniques are L-Systems, Fractals, Perlin Noise, Voronoi Diagram.

3.1 Fractals

Fractal, derived from the Latin word *fractus* which means "broken" or "fractured". A fractal is generally "a rough or fragmented geometric shape that can be split into parts, each of which is (at least approximately) a reduced-size copy of the whole," a property called self similarity [Prusinkiewicz and Lindenmayer 1990]. Fractal are created by a procedural technique of using a recursive algorithm where the basic fractal shape is recurred a certain number of time which defines the detail of the fractal shape. Fractals are useful while creating irregular shaped geometries with repetitive textures like clouds and mountains. Although Fractals can create a geometry of complexity to a certain level they are still not as efficient as L-Systems in procedural creation.

3.2 L-Systems

L-System or Lindenmayer system is a parallel string rewriting system originally developed to study bacteria replication and the growth of simple organisms. This was further extended to generate more complex natural phenomenon such as plants and branching structures. Adopting this, Prusinkiewicz and Lindenmayer together extended the application of L-System to computer graphics and described it in the book "Algorithmic Beauty of plants" which has since been very useful and the basis of many procedural creation techniques [Prusinkiewicz and Lindenmayer 1990]. Basically, L-Systems works on the concept of rewriting, a technique for defining complex objects by successively replacing parts of a simple object using rewriting rules. L-Systems in the most basic form can be explained as follows:

Rule 1 = A $\rightarrow$ AB
Rule 2 = B $\rightarrow$ BA

Iteration 1 : A
Iteration 2 : AB
Iteration 3 : ABBA
Iteration 4 : ABBABAAB
. . .

This Concept is then applied to generation of geometries which can be further extended to create extremely complex geometries by adding more complex rules depending on the requirement [Prusinkiewicz and Lindenmayer 1990].

3.3 Perlin Noise

Perlin Noise can generate a set of random values relative to a range specified by the user. A combination of multiple Perlin Noise values added, multiplied or subtracted together can give a range of complex Noise value which can be used to create geometries with irregular shapes which as a whole still have a uniformity like in a terrain or clouds.

3.4 Voronoi Diagram

Voronoi Diagrams are cellular patterns generated by scattering points on a surface and each point has a Voronoi cell where each edge of a voronoi cell clamps itself to the nearest point's voronoi cell edge so that every edge is equidistant from its two nearest points. This technique can be used to generate cellular pattern geometries like cracks on a dry land etc.

3.5 Point Selection

This technique gives the highest level of control to the user than any other procedural techniques. It is quite similar to the technique of taking some part of information as input from the user and then using that information to generate the procedural content. For example, calculating data from the Density Map or Population Map provided by the user as an input and then use that information to categorise the positioning of different geometries which can again be combined with any of the techniques explained earlier. In this project Point Selection will be used partly for the city generation.

Chapter 4

Implementation

This chapter describes the development of the tool and the various techniques used.

4.1 Approach

The basic idea was to create an interactive tool that gives artistic control over the creation of a complex and customizable city using procedural generation approach and at any point user can add, delete or modify the elements of the city like buildings, roads, road pattern, etc. as he see fit. Instead of using and being limited to the existing techniques for city generation like L-Systems a new or extended technique should be implemented. To give variety of control to the user two different type of approaches are available in the tool to generate a city. User can select a Map Based city creation where user can provide a predefined layout of the city and then configure and control distribution of the city elements or User can can choose to create a city entirely from scratch with the help of artistic content creation controls which allow the user to create/modify road patterns dynamically and also paint buildings, houses and trees, etc. depending on the users choice.

4.2 City Generation Tool

4.2.1 Overview

A Houdini Digital Asset (HDA) using the identifier "CITY" was created. It can be found in the otl folder. To install the Digital Asset in Houdini go to File → Install Digital Asset Library. Browse and select the CITY.otl file. It will be visible in the Digital Asset menu when u click Tab in the Network view inside a geometry node. Select the Digital Asset to create an instance of the tool.

4.2.2 Creation Type: User Defined

This is the main part of the tool which allows the user to have the most control over the creation and the customizability of the city. This method of city creation depicts the method of creation in various city building simulation games like Cities in Motion and SimCity. It controls the "Switch_Creation_Type" node in the network which further disables the unselected part of the network and works only with the selected creation type.

4.2.3 Year

4.1.3 This option allows user control the age of the city. There are three options that user can select from, 1950, 2000, 2050. Each selection has different effect on the creation of the city every age has different set of buildings and houses which change according to the selection. This also affects in the increase in the average level of the city. Older the age, higher will be the geometry level. This is also directly proportional to the density / population of the city and the average number of buildings to be placed on the city will be higher with time. This does not affect the city scale or size in any way bearing in mind the fact that with time only the city buildings and density will grow and not the city scale itself. This options was mainly meant to give an overall growth appearance to the city with the change in time. Many nodes in the network are controlled by this option which will be further explained as and when they are discussed.

4.2.4 Use Dummy Geometry

This is an important option to make the tool much more efficient while at the creation stage of the city. A city is bound to have a very large number of geometries when it is being created and working with high mesh geometries or textured geometries could make the network extremely heavy which will make the working of the tool slower. To avoid this "Use Dummy Geometry" option is provided which can be activated or deactivated at any point and accordingly heavy geometries will be replaced by dummy low poly geometries making it easier and faster to set up the positioning of all the city elements and once everything is set, it can be deactivated again to get the dummy geometries replaced by original geometries. This affects the all the switch nodes which control the Switch To Dummy Geometry option respectively.

4.2.5 City Size

This controls the scale of the ground plane or the terrain. This also modifies the available area for the geometries to be positioned.

4.3 Road Creation

Road Creation is one the key points of this tool. This option has gives user the control that most of the procedural city creation applications do not or cant not because of their complete procedural creation system. This is the plus point here that though the road is getting generated procedurally since it basically is a curve which has various other nodes and expressions affecting it to make the curve a road, and a customizable road. The user has the control to select any point on the road and move it and the road will modify itself accordingly. Similarly the user can delete any point from the road to rearrange the road network. To add a new point somewhere in the road user can do so by SHIFT + Left Mouse Button (LMB) Click to increase the complexity of the road. User can can also click anywhere on the terrain and it will create a new point and connect it with the end of the road. User can adjust the width and depth of the road. Also the colour can be adjusted. To make the tool more efficient there is an option to "Show only road on terrain for faster results". It is a handy option while modifying the road as the tool might get heavy with a large number of buildings available in the city, even dummy geometries as even though they are low poly geometries collectively at such a large number they can be very heavy.

The Display Grid option must be ON in the Display Options toolbar since the Base curve for the road creation gets created on X-Z axis i.e. on the grid and not keeping the Display Grid option will not have an axis for the points to be placed.

Since the Road is getting generated on X-Z axis it is necessary to procedurally adjust it to the position of the terrain depending on the terrain height which can be variable as user has the control to modify the terrain as well. This will be explained in the algorithm below:

```
Algorithm for Road Creation:
\Take Base Curve as input for road profile
\Snap the points very close to each other
\Increase the subdivisions of the curve for
smoother corners and surface
\Reposition every point on the curve to be
equidistant from the previous and next point
\Keep the Beginning and End points at their position
\Sort the curve_point_numbers in order so that
 positioning is correct at all times
\Modify the Normal of each point so that the Y is
always pointing upwards and X is always point to
the next point
\Add a line at the beginning of the Base curve
\Copy the line on each point on the Base curve
\Make it always perpendicular to the Base curve
\Skin/Join the lines so that they follow the path
of the curve
\Make the line customizable by the user to control
the width and depth of road
\Take the Y position of the every point on the terrain
\Project the curve on to the Terrain's so that
Y position of the Terrain matches with the point
 position of the curve
\Use an instance of the Terrain and make
Terrain-Minus-Road so that only they rest of the
area is available for painting geometries
\Join the curve and the original Terrain
```

4.4 Paint Buildings

This is the most interesting and artistic option of the tool. It allows user the control to select a building and paint it anywhere in the city. User can remove the buildings already painted as well and the user can reset the selected building and start painting again. To achieve this each individual geometry was passed on to a GroupPaint node which allows user to paint and area on the grid. the selected points were then passed on as the position for the buildings respectively for each building.

4.5 Map based

This options allows the user to provide a layout map which will used as the road layout of the city and buildings will be distributed automatically on the Terrain-Minus-Road area and user has the control to manipulate the city size, displacement of the terrain, density of the distributed buildings and the rotation offset for the buildings.

Chapter 5

Conclusion

The project has succeeded in most areas. The tool has many new and interesting features compared to the existing city-building tools. Despite of such complexity in the network the Tool is fairly fast and efficient. The tool surely struggles to achieve the realism, but it shows fair amount of customizability and artistic control to the user. In the current state the tool is not efficient enough to produce assets at the leave of detail required in the industry but it is still an efficient tool for inspiration and an prototype for implementing new and different techniques.

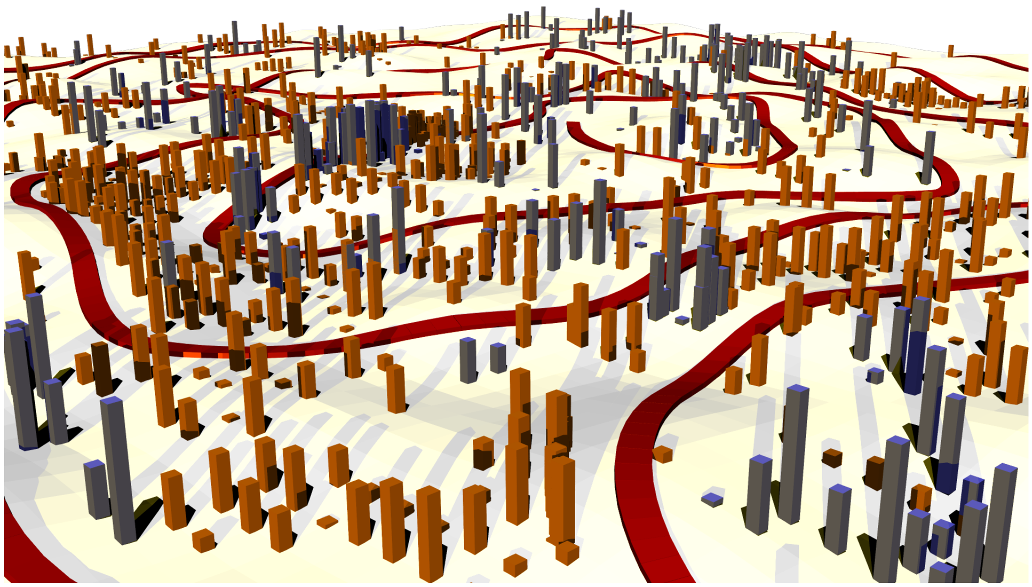

Figure 5.1: With Dummy Geometries

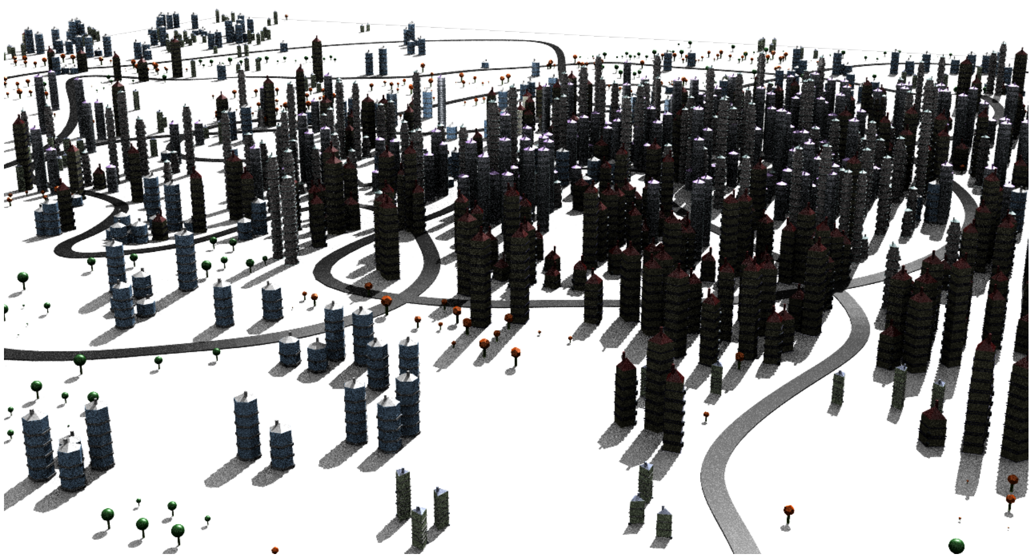

Figure 5.2: With Original Geometries

Chapter 6

References

Goffredo, E. (2010), 'A tool for procedural destruction in houdini'. Available from: http://nccastaff.bournemouth.ac.uk/jmacey/MastersProjects/MSc2010/04EmanuelleGoffredo/ [Accessed 10 May 2011].

odforce (2011), 'Curve on Mountain?'. Available from: http://forums.odforce.net/index.php?/topic/13802-curve-on-mountain/ [Accessed 10 May 2011].

INTERACTIVE DATA VISUALIZATION INC. 2011. SpeedTree RT. Available from: http://www.speedtree.com [Accessed 14 August 2011].

PROCEDURAL Inc., 2011. CityEngine. Available from: http://www.procedural.com/showcases/showcase/ [Accessed 14 August 2011].

PARADOX INTERACTIVE, 2011. Cities In Motion. Available from: http://www.citiesinmotion.com [Accessed 14 August 2011].

SideFX (2011b), 'Houdini 11 documentation – nodes'. Available from: http://www.sidefx.com/docs/houdini11. [Accessed 10 May 2011].

Figures:

CityEngine in Cars 2.png - PROCEDURAL, 2011. Cars 2. Image. Available from: http://www.procedural.com/showcases/showcase/cars_2.html [Accessed 14 August 2011].

SpeedTree Cinema.jpg - INTERACTIVE DATA VISUALIZATION INC. 2011. SpeedTree Cinema. Available from: http://www.speedtree.com/gallery/ [Accessed 14 August 2011].

Cities in Motion.jpg – PARADOX INTERACTIVE, 2011. Cities In Motion. Available from: http://www.citiesinmotion.com [Accessed 14 August 2011].