Open Streets Race

Anders Mousten, Michele Ermacora

I. Introduction

Open Streets Race is a Unity[1] plug-in created for generate a racing game from data obtained from the open Street Maps[2] site. Open Street Maps is a site similar to Google Maps[3] where the user can select different parts of the earth, but with an important difference: the user can also download parts of the maps obtaining informations about that zone (for example position of the roads, position of the buildings and size of them). These informations can be download as xml files, with nodes representing different objects (for example buildings, streets, parks, rivers, ecc.). The goal of our plug-in is let the user import the data inside the Unity[1] engine in a simple way, and create buildings and roads from generate levels using real cities as bottom line using procedurally generated algorithm. The user then can modify the map generated using the game play layer, where he can decide to smooth the roads, placing checkpoints and cars. In this way, the generation of maps for car racing games are becoming more fast, while giving him control on the results. Another important aspect of our plug-in is that is easy to include other kind of constraints inside the game play layer, making it easy extendible also for include other kind of genres (like FPS).

II. Backgorund

This work was inspired by the SketchaWorld system[4]. In this system, the user can concentrate in what they want to create instead of how by using a constraint solving and semantic modelling. This is achieved by dividing the different part of the level creation in different layers. For example, in the first layer (called landscape) the user can create the terrain, modifying the highness of it applying an height map or use the tools inside the editor. In the second (water), the user can create rivers. For do this, he can click on different part of the maps and the pcg algorithm automatically generate the river taking in consideration also the below layer, modifying it if necessary. In fact, in their framework, the layers on top of each other can modify the lower ones, using the constraints inside each of them for resolve the conflicts between layers. Although, the suer can select to create a world from scratch, or import data from Open Street Maps and generate from there the world.
While this framework can generate very interesting worlds, with rivers, mountains, roads and cities nice fit together, one critique that can be done is that the, even though was created game developers, the developers haven't add also a game play layer (or take in consideration the game play element). This cause the generation of realistic worlds but no interesting for a game play prospective. For example, in an fps set in a city background, we don't want only the generation of buildings,rivers,roads ecc., but also covers, short cut between buildings, stairs that can make the players reach the roof.

Another paper that inspired this work was Generating Interesting Monopoly Boards from Open Data[5]. In this paper, the authors use Open Data information about the population,reachness and other parameters to generate Monopoly boards. This is done by using an evolving algorithm in which are evolved the different weights for the street creation inside the Monopoly game. The user can decide which indicators use for defining what means prosperity inside an area, and then, after some calculations, the board is generated. This paper is interesting, as it uses real world data for generating Monopoly boards letting the user create is own board based on his knowledge of what it means reach are.

III. Game Design

While the main goal of our project is to give to the final user a tool for generating levels from real world data, in our project we focus on building a game play layer for a car racing game from real world cities. The game play is inspired by other racing games as Need For Speed[6], in which the player can race inside a city as in a free roaming game, and in which the different tracks can have also short cuts (that are difficult to take as, for comparison, continue for the main street). Because modelling an entire city from scratch would require too much effort we decide to creating it using procedurally generated algorithms for generate the building and the roads from the data of Open Street Map. In this way, we can also modify the different elements without had the need to re-create big models from scratch, or modify them. For example, instead of try to make fit different shapes of curves, we need only to click where we want the road start and ends, and the algorithm take care of generating the entire street (that can also be modified by the game play layer). Although, having at our disposal this layer make more easy for us try different kind of solutions for the track, improving the final quality of it.

IV. Methods

In this section we are going to explain the different elements composing our plug-in.

A. Xml Parser

The first element the user will use is the xml parser. This is a dll file in which the user can select which kind of data import from the xml document download from open Street Maps. For the moment the dll support the export of buildings, streets, rivers and parks. Anyway this is easy extendible by overwriting the right method. The dll then create a text file with the information that will be processed by the different layers implemented inside Unity. Because Open Street maps cant export very big maps as xml files, the dll give the possibility also to attached to the same text file nodes from different xml files, leaving to the user completly freeedom on which zones (also caming from different places) he want to use for his level creation and, and how much big the level should be.

B. Building Layer

C. Roads Generation

For the road generation we decide to use an external plug-in[7] for have the basic functionality of generating a mesh from a predefined set of points. For work, this script need to be attached to a terrain. This give the user the freedom of generated procedurally part of the background, or modify the terrain itself (as creating hills for example), and the roads, as the buildings, automatically position themselves on top of the terrain created. This generator simple take as input the different positions in which the user click on the terrain, search inside the data structure which nodes of the world street map are the most near to them and generate the street. For do that, we have used the A* algorithm, in which the heuristic function is the euclidean distance between the nodes that the user have clicked and the streets inside our list. After we have found the right road, we use the nodes composing the road and pass them to the external plug-in for the mesh generation. This plug-in apply the cubic spline algorithm for approximate the curve between the different nodes and create a smooth path. The cubic spline is an algorithm in which, given If, between the points that the user has clicked, there is a hill, the algorithm take also care of placing the different parts of the mesh on top of the terrain.

V. Results

VI. Discussion

References

[1] Unity

[2] open Street Maps

[3] Google maps

[4] sketchaworld

[5] Generating Interesting Monopoly Boards from Open Data Marie Gustafsson Friberger and Julian Togelius

[6] need for speed

[7] plugin for street generation

[8] A. Great, *This is the book title*. This is the name of the publisher, 2006.

[9] F. Author, S. Author, and T. NonRelatedAuthor, “This is the paper title,” in *This is the proceedings title*, 2008, pp. 1–8.

[10] B. Myself, “This is the title of the journal article,” *This is the name of the journal*, pp. 1–30, 2007.