Procedural Generation of Roads with Conditional Generative Adversarial Networks

Lin Ziwen Kelvin
Computer Science Department
National University of Singapore
Singapore
e0014961@u.nus.edu

Bhojan Anand
Computer Science Department
National University of Singapore
Singapore
banand@comp.nus.edu.sg

ABSTRACT

Procedural terrain generation refers to the generation of terrain features, such as landscaping, rivers or road networks, through the use of algorithms, with minimal input required from the user. In the process of game development, generating terrain is often an important part of the game development process. Traditional generation methods are often too time consuming especially with larger terrain maps. On the other hand, procedural methods that generate terrain automatically often do not have much user control over the output. We explore the usage of conditional generative adversarial networks in the creation of road maps, as well as the application of such road maps in the creation of game levels in game development engines such as Unreal Engine 4.

ACM Reference Format:
Lin Ziwen Kelvin and Bhojan Anand. 2020. Procedural Generation of Roads with Conditional Generative Adversarial Networks. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH '20 Posters), August 17, 2020.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3388770.3407422

1 INTRODUCTION

Procedural Terrain Generation (PTG) refers to the generation of terrain using algorithms, specifically with little to no human input. This is in contrast to traditional terrain creation, where the landscaping was sculpted by hand, and the secondary features are added on manually. PTG allows for terrain and levels to be generated dynamically without needing humans to create the entire landscape by hand.

Road networks are frequently used in games in some form or another. In city-based games such as Sims or Grand Theft Auto, the game takes place within cities, thus road networks directly influence player navigation. In other games such as Player Unknown's Battleground, road networks allow for safer transport of players, and can potentially affect player strategy through usage of chokepoints. Outside of games, road generation may be of use in urban planning, such as for predicting traffic movements and future urban development. However, to the best of our knowledge, most road network generation methods do not have much customizability, and users are often unable to control the types of roads being generated.

SIGGRAPH '20 Posters, August 17, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7973-1/20/08.
https://doi.org/10.1145/3388770.3407422

We thus explore the usage of conditional Generative Adversarial Networks (cGAN) in creating a framework that allows users to generate road networks that can be used for game development.

2 RELATED WORK

2.1 Example-based Generation Algorithms

Growth-based algorithms were proposed by [Yu and Steed 2012] and [Beneš et al. 2014], where an initial road network is provided, and a city comprising of the road networks is generated by considering the surrounding terrain.

Generative Adversarial Networks (GANs) were used by [Beckham and Pal 2017] in the generation of landscape heightmaps, by using satellite images as the training dataset. In addition, the predicted texturing of the generated terrain for the given heightmap is also generated, which can be imported into the Unity Engine to create landscapes.

[Hartmann et al. 2017] proposed using GANs in the generation of road networks. The resulting GAN from [Hartmann et al. 2017] however makes use of random noise in generating new tiles, leading to a lack of control over the output, making them similar to traditional PTG algorithms in output controllability.

Conditional GANs were used by [Guérin et al. 2017] in the generation of landscapes. Unlike [Beckham and Pal 2017], [Guérin et al. 2017] uses heightmap data from the United State Geological Survey Earth explorer. [Guérin et al. 2017] also features a real-time user sketching tool, that allows users to generate terrain in real time.

3 OUTLINE OF APPROACH

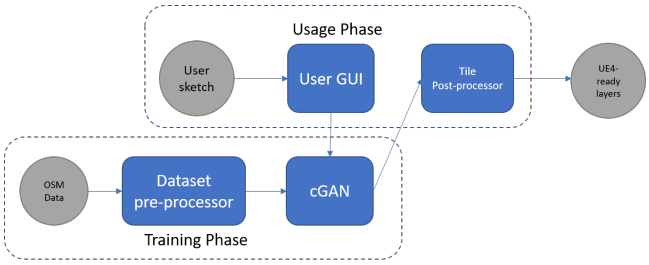

Figure 1: Overall Roadmap Generation Framework

Figure 1 describes the overall pipeline used to generate a full road network, given an initial sketching of primary roads. It consists of a data pre-processor, a conditional Generative Adversarial Network, a simple graphical user interface to input the initial network, and a

post-processor that formats the generated network for use in game engines such as Unreal Engine 4.

3.1 Dataset Pre-processor

We use OpenStreetMap map tiles as our dataset due to its ease of access, high customizability through stylesheets, hence allowing the cGAN to be trained to generate different types of features, and road networks generated from this dataset are likely to be realistic as well.

The dataset is prepared by obtaining two sets of OpenStreetMap tiles, one that contains only the primary roads, which constitutes the input tiles, and one that contains the primary, secondary roads, and buildings, which constitutes the target ouput.

3.2 cGAN Architecture and Training

The architecture for the cGAN comprises of a discriminator network d, and a generator network g. The cGAN is implemented using the Keras and TensorFlow libraries, and is largely adopted from the Pix2Pix network proposed by [Isola et al. 2017].

The discriminator network d takes in a $< generatedtile, targettile >$ pair and assigns it an accuracy score from 0 to 1. This score is based on how similar the target tile is to the training dataset in general. It is implemented as a standard Convolutional Neural Network, where a 4 x 4 sized window is used as a filter. MaxPooling and BatchNormalization is carried out on each layer. To prevent overfitting, a dropout rate of 0.4 is applied in the first Conv2D layer, and the binary cross-entropy is used for the loss function.

The generator network takes in a base tile i, and outputs a generated tile $g(i)$. The generator is implemented as an encoder-decoder network with skip connections, with a 4 x 4 sized window as a filter. Each encoder block consists of a Convolutional layer with BatchNormalization, and each decoder block consists of a transposed Convolutional layer. Dropout is applied in the decoder blocks to prevent overfitting.

3.3 Graphical Interface and Post-Processing

To allow for game developers to create their own road networks, we also provide a simple graphical user interface, implemented using the tkinter library. The GUI comprises of a drawing canvas, where the user uses their mouse to draw the initial primary road network. The user can then select the 'Generate' option to see what kind of road network the cGAN produces, and if they are satisfied with the outcome, save the generated tile for further use, be it to use as a reference image, or for use in Unreal Engine 4.

4 RESULTS

Our results show that a cGAN can be used to effectively generate full road networks from an initial sketch.

Using the graphical interface, user-drawn sketches were input to simulate the user drawing, and the subsequent weight layers were input into a fresh Unreal Engine 4 level asset. Using this landscape, foliage was planted into the map, and cuboid polygons were placed into spots designated as building locations, represented by the orange textures, to simulate the placement of buildings.

From this and other free assets provided in Unreal Engine 4, a simple driving simulator was created using the generated landscape as a basis. A sample screenshot from the driving simulator is shown in Figure 2.

Figure 2: Driving Simulator created in Unreal Engine 4 using the generated tiles

5 CONCLUSION AND FUTURE WORK

By using a Conditional Generative Adversarial Netowrk, we were able to create entire road networks from a simple initial input of primary roads. Users can also make use of a GUI to provide their own input sketches to generate their own tiles. These generated tiles were able to be adapted for use in Unreal Engine 4 in the creation of game landscapes.

This work is supported by Singapore Ministry of Education Academic Research grant T1 251RES1812, Dynamic Hybrid Real-time Rendering with Hardware Accelerated Ray-tracing and Rasterization for Interactive Applications.

REFERENCES

Christopher Beckham and Christopher Pal. 2017. A step towards procedural terrain generation with gans. *arXiv preprint arXiv:1707.03383* (2017).

Jan Beneš, Alexander Wilkie, and Jaroslav Křivánek. 2014. Procedural modelling of urban road networks. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 132–142.

Éric Guérin, Julie Digne, Eric Galin, Adrien Peytavie, Christian Wolf, Bedrich Benes, and Benoît Martinez. 2017. Interactive example-based terrain authoring with conditional generative adversarial networks. *Acm Transactions on Graphics (TOG)* 36, 6 (2017), 228.

Stefan Hartmann, Michael Weinmann, Raoul Wessel, and Reinhard Klein. 2017. Streetgan: Towards road network synthesis with generative adversarial networks. (2017).

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1125–1134.

Qizhi Yu and Anthony Steed. 2012. Example-based Road Network Synthesis.. In *Eurographics (Short Papers)*. 53–56.