Terrain Generation Using Progressive Growing GAN

Byung Il Choi*
University of Illinois
at Urbana-Champaign

Eric Gene Shaffer†
University of Illinois
at Urbana-Champaign

Abstract

Terrain generations can be done using Generative Adversarial Networks as recent works have shown that GAN can generate various kinds of objects, including terrains. Terrain generations would require high resolution for terrain's topological data. As NVidia's Progressive Growing GAN has proven to generate object's images in high resolutions, we chose to generate terrains using Progressive Growing GAN. The GAN doesn't only generate colors of surface of the earth, but it also generates the heights together. The training process provides how to utilize a Convolutional Neural Network to achieve high relevance between the colors and heights.

Index Terms: Human-centered computing—Visualization—Visualization application domains—Geographic visualization; Computing Methodologies—Computer graphics—Shape modeling—; Computing Methodologies—Machine learning—Machine learning approaches—Neural networks

1 Introduction

The procedural terrain generation is a popular way of generating terrains; the diamond square [2] method and Perline noise [10] are two well known methods. And recently, machine learning methods were introduced, which uses generative adversarial networks [3]. In particular, C. Beckham and C. Pal tried to use DCGAN [11] and pix2pix GAN [5] to generate terrains [1]. But two networks were trained separately and seems to have some mismatching problem between surface color and its heights.

The purpose of this study is to generate terrains using GAN that can combine the training of surface colors, topological data, and hydrology data, which provides a stable training process. Progressive Growing Of GAN [6] exactly satisfies the purpose.

2 Previous Works

Progressive Growing GAN uses many distinctive features that make itself different from other GANs.

2.1 GAN

Generative adversarial networks have two networks which consist of a discriminator D and a generator G. D tries to discriminate better and G tries to fool D. This problem can be formulated as a minimax game [3]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))] \quad (1)$$

Progressive Growing GAN uses only one critic, meaning D and G alternates on a per-minibatch basis [6]

*e-mail: bichoi2@illinois.edu
†e-mail: shaffer1@illinois.edu

2.2 Loss

As WGAN-GP [4] is genrally more stable loss function [6], WGAN-GP is used in this study, which is formulated as below:

$$L = \underset{\tilde{x}\sim\mathbb{P}_g}{\mathbb{E}}[D(\tilde{x})] - \underset{x\sim\mathbb{P}_r}{\mathbb{E}}[D(x)] + \lambda \underset{\hat{x}\sim\mathbb{P}_{\hat{x}}}{\mathbb{E}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2] \quad (2)$$

where $\underset{\tilde{x}\sim\mathbb{P}_g}{\mathbb{E}}[D(\tilde{x})] - \underset{x\sim\mathbb{P}_r}{\mathbb{E}}[D(x)]$ is for critic loss and
$\lambda \underset{\hat{x}\sim\mathbb{P}_{\hat{x}}}{\mathbb{E}}[(||\nabla_{\hat{x}} D(\hat{x})||_2 - 1)^2$ is for the gradient-penalty.

2.3 Pixelwise normalization

Progressive Growing GAN uses Pixelwise normalization [6] to prevent abnormal changes of magnitudes in generator and discriminator, which is a variant of local responsive normalization [9].

$$b_{x,y} = a_{x,y} / \sqrt{\frac{1}{N}\sum_{j=0}^{N-1}(a^j_{x,y})^2 + \varepsilon} \quad (3)$$

where N is the number of features, a is original feature vector, b is normalized vector for position (x,y), and $\varepsilon = 10^{-8}$ [6].

2.4 Equalized learning rate

We use ADAM [7] to optimize Progressive Growing GAN. The GAN initializes the weights w with $N(0,1)$ an $\hat{w}_i = w_i/c$ where w_i are weights and c is the per-layer normalization constant, which ensures that the dynamic ranges of weights becomes same [6], i.e. same learning rate.
ADAM's hyperparameters are $\alpha = 0.001$, $\beta = 0$, $\beta = 0.99$, and $\varepsilon = 10^{-8}$ with no learning rate decay [6].

2.5 Minibatch standard deviation

A terrain can have many features which would look unnatural if the features get omitted. GANs often use a subset of theses features [6], but minibatch discrimination [8] can help to increase the variation captured from the training batch. Progressive Growing GAN has simplified this without learnable parameters nor hyperparameters [6] which is formulated as below:

$$\sqrt{\mathbb{E}[(X-\mu)^2] + \varepsilon} \quad (4)$$

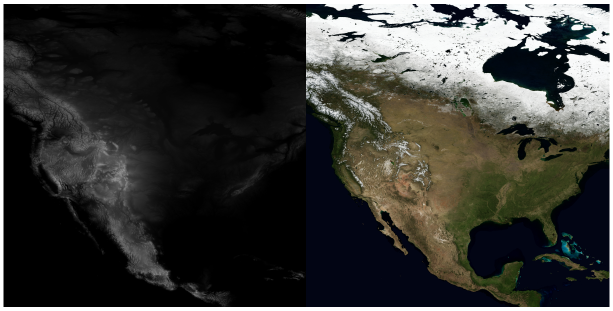
This is a standard deviation of features which is used to be concatenated to all spatial locations over the minibatch [6] as an additional feature map.

2.6 Progressively growing network layers

Progressive Growing GAN's key feature is how the network grows layers to stabilize the training. From low resolution to high resolution, the GAN incrementally trains layers to work well. In each step of adding a layer, the GAN doubles the resolution and increase α linearly that blends the results of two connected layers [6].
For blending of two layers, downsampling is required for the discriminator and upsampling is required for the generator [6].

3 TRAINING

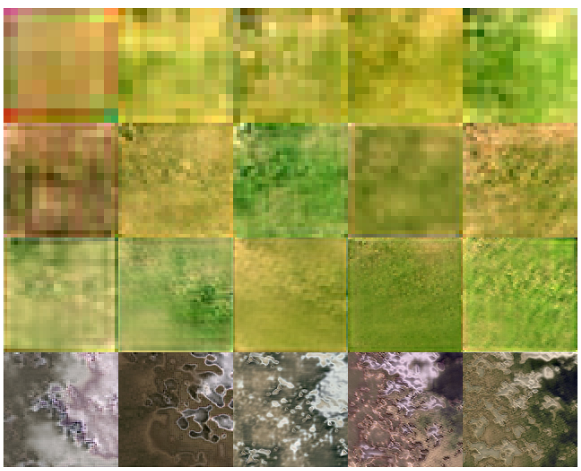
We used earth's surface image and topological data from NASA's Visible Earth, which looks like below:



Then we combined the color channels with topology channel for the input of the image sampler. And then, our image sampler randomly cropped above input with more cropping probability on the United States. The trainer samples 30k random crops for training.

3.1 Progressively growing layers

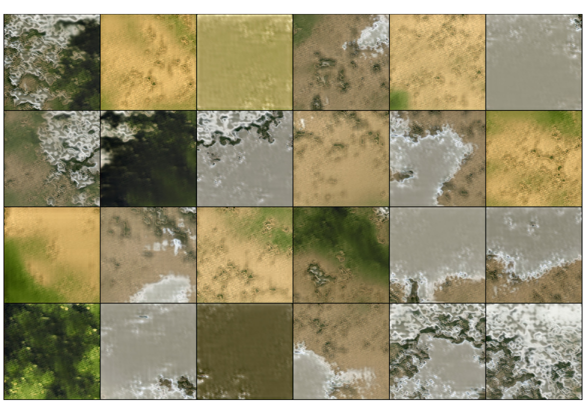
The progressive improvements of growing resolution can be visualized as below:

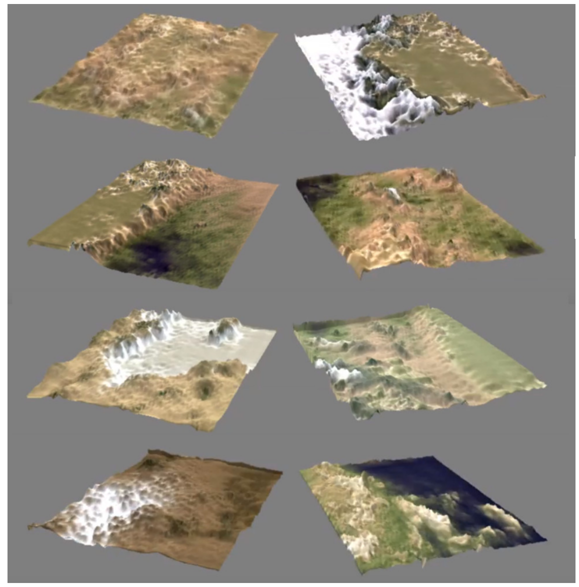


From left to right, we can see how fade-in is affecting the training progress. From top to bottom, we can see how resolution is growing with additional layers.

4 RESULT

The GAN can generate various types of terrain with the combination of various features as below:





The transition from snowed terrain to other types of terrain in the $\text{Output}_{1,2}$ is natural as we would see from the northern part of America. And, we can see dirts and grass in $\text{Output}_{3,1}$, the mountains and oceans in $\text{Output}_{4,1}$, canyons in $\text{Output}_{2,1}$, etc.

5 CONCLUSION

The GAN was trained as intended, but there's a need for improvements for both visualization and the network.

6 FURTHERMORE

Since we have the surface type as colors, we need to increase the surface detail of the terrains that the GAN generated and find a way to train hydrological data together for applying surface erosion. As the network uses a lot of RAMs, we can also consider serializing the network to distribute the computation.

ACKNOWLEDGMENTS

Thanks to Professor Eric Gene Shaffer for advising this study. And, thanks to the designer of Progressive Growing GAN. NASA's visible earth data-set is a great resource for GAN.

REFERENCES

[1] C. Beckham and C. Pal. A step towards procedural terrain generation with gans, 2017.

[2] A. Fournier, D. Fussell, and L. Carpenter. Computer rendering of stochastic models. *Commun. ACM*, 25(6):371–384, June 1982. doi: 10.1145/358523.358553

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[4] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[5] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[6] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[7] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[8] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds., *Advances in Neural Information Processing Systems 29*, pp. 4743–4751. Curran Associates, Inc., 2016.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. doi: 10.1145/3065386

[10] K. Perlin. An image synthesizer. *SIGGRAPH Comput. Graph.*, 19(3):287–296, July 1985. doi: 10.1145/325165.325247

[11] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.