CS230

Adapting Image Outpainting for Unbounded Two-Dimensional Landmass Generation

Robathan Harries
Department of Computer Science
Stanford University
rharries@stanford.edu

Abstract

Have you ever seen a map and wondered what exists just beyond its borders? This project adapts state of the art methods from the field of image outpainting to procedurally expand large-scale terrain maps repeatedly in any and all directions. First an image outpainting model is trained to generate map tiles consistent with a single adjacent source tile. Then, by adapting the generator architecture to two-dimensional case, transfer learning is applied to quickly train a second model to generate map tiles consistent with two adjacent source tiles. By using both models in concert, large-scale terrain maps can be extended with procedurally generated content in any direction.

1 Introduction

Procedural Terrain Generation is the task of creating realistic environments for simulations, video-games, or other purposes. Many algorithms exist for generating terrains at a near-human scale, using methods ranging from deep neural networks to physical erosion simulations. However, far fewer methods exist for procedurally generating terrain on the scale of islands, continents, and world maps. The methods which do exist for this generally require hand-designed features, like mountains, plateaus, and troughs, which can be warped and overlayed onto a height map. Though this approach can create realistic maps occasionally, it is inconsistent and requires significant human oversight.

The goal of this project is to use methods and insights from image outpainting research to expand large-scale maps containing realistic features on land, sea and the shorelines in between. The input to this algorithm is an array of topological heights covering a fixed-size square tile. Two generative models are then used in concert to generate multiple new heightmap tiles which can be attached around the input tile.

2 Related work

2.1 Image Outpainting: NS Outpaint

Unlike image inpainting, image outpainting operations can be repeated recursively to expand an image arbitrarily. Unfortunately, most image outpainting methods degrade significantly in quality over multiple generation steps, and converge towards a kind of eigenvalue of low-quality, unchanging images. The NS-Outpaint algorithm, developed by Yang et al. (2019), was designed specifically to combat this issue. Figure 1 shows the performance of this method over multiple generation steps, compared to other outpainting methods which exhibit degrading quality.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Figure 1: Multi-step outpainting comparison against NS-Outpaint (labeled RCT+SHC). The Pix2Pix and Contextual Attention (CA) methods highlight the degrading quality and convergence of most outpainting methods over multiple generation steps. Yang et al. (2019)

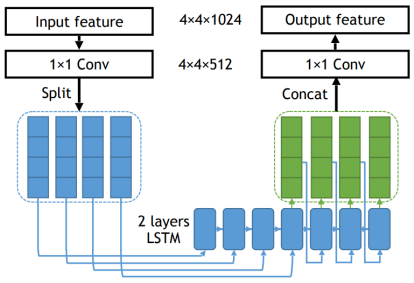


Figure 2: Recurrent Content Transfer (RCT) Module. Yang et al. (2019)

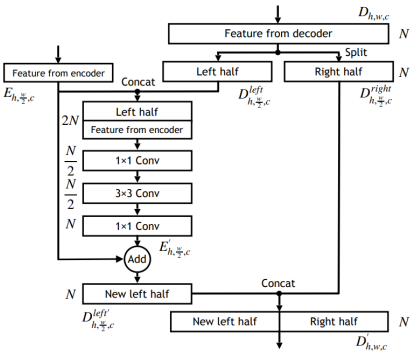


Figure 3: Skip Horizontal Connection (SHC) Module. Yang et al. (2019)

The general architecture of NS-Outpaint is one common in both inpainting and outpainting methods: An encoder-decoder network that is trained adversarially against a local discriminator (focusing only on newly generated content) and a global discriminator (focusing on the expanded image as a whole). This occurs in Gardias et al. (2020), Zhang et al. (2019), and many more inpainting and outpainting methods. The latent state between encoder and decoder represents encoded information about the original image, and is used to generate an adjacent latent state of the same shape, which represents encoded information about the original image's adjacent neighbor. Both latent states are decoded together, producing a reconstruction of the source image adjacent to a newly generated image.

The algorithm achieves sustained quality and long-distance information transfer with two particularly important modules.

First, the Recurrent Content Transfer (RCT) module (Figure 2) uses a two-layer LSTM to generate a new latent state conditional on the original encoded latent state. This sequence model approach allows significantly improved ability to communicate detailed feature information across long distances.

Second, the Skip Horizontal Connection (SHC) module (Figure 3) operates in the decoder stage, and provides a skip connection from the partially-encoded features of the original image to the corresponding partially-decoded features of the original (not newly generated) half of the decoded image. This significantly improves the reconstruction of the original given image, and improves consistency and smoothness along the border between the two images, since features from the original image can propagate across the border to affect nearby features in the generated image.

3 Dataset and Features

3.1 Data Source

The dataset that chosen for this project is the SRTM 15+ dataset, a compilation of height measurements both on land and underwater, spanning the entire globe at a resolution of 15 arc seconds per pixel, which is approximately 50,000 m^2 at the equator. To limit geographic warping at the poles, this

project dataset only accesses latitudes between -80 and 80 degrees. The overall image over those bounds has dimensions 86,400 x 38,400 pixels. Scripts automatically download and store this entire area in 576 separate windows with 10 degrees to a side. A separate algorithm can then stitch together these windows to create an image corresponding to any longitude and latitude bounds, down-sampled to any desired resolution.

3.2 Data Format

Pixel values have only 1 channel, which is their height in meters. Given the importance of the zero meter threshold (sea level), the dataset is not completely normalized, but has been scaled down to facilitate model training.

To allow experiments at different coordinate scales, different resolutions, and varying filters, the original SRTM 15+ dataset is stored separately from specific dataset versions which can be compiled for particular models. To generate a specific dataset for training and testing a model, a sliding window with fixed width and height is passed over all available coordinates, and each resulting image is down-sampled to a fixed resolution, resulting in a dataset which contains many partially overlapping images. These compiled datasets also support filtering samples by minimum landmass and height variance, as well as augmentation through vertical and horizontal mirroring.

For this project, models were trained on image tiles with 20 degrees longitude/latitude to a side. This tile size was chosen since samples of that size often contain interesting information about landmasses and shorelines, while limiting the number of samples which contain near-uniform heights, especially underwater, which would otherwise risk biasing models to just generate near-uniform outputs. The final datasets were also augmented with vertical and horizontal mirroring, resulting in a dataset with 10,944 samples. Appendix A shows some examples from the dataset.

3.3 Dataset Difficulty Scaling

In order to facilitate fast iteration, especially during the early exploration stages of the project, in which many differing architectures were implemented, each architecture went through a few training tests on different datasets before full-scale training. First, a toy version of the architecture, with fewer layers and channels, was trained on a dataset compiled with lower resolution (64x64), filtered to have a standard deviation of at least 10m and at least 5% of total area above sea level. After confirming initial results match expected results, the model was scaled up to its full architecture, and the dataset increased to its intended resolution (128x128), with filters removed.

4 Methods

4.1 Problem Specification

The overall algorithm operates on square map tiles, each with 128x128 one-channel pixels corresponding to 20x20 degrees. There are two sub-problems which must each be solved in order to arbitrarily extend source map tiles.

- **One-dimensional expansion**: Generating a new tile given an adjacent source tile. This considers consistency along only one edge.
- **Two-dimensional expansion**: Generating a new tile given three neighboring source tiles, attached along two sides and the diagonal in between.

4.2 Generator Architecture

The core model of the project is a version of the NS-Outpaint architecture with fewer convolutional channels. The base architecture of NS-Outpaint is incredibly intensive, with 119 million parameters trained for 1,500 epochs. Due to the relative simplicity of the heightmap dataset, and the time/computation limitations of a single-person course project, I reduced the maximum number of channels in any layer of the network from 1024 to 256, resulting in a parameter count of 9.75 million while retaining much of the generative power of the original network. The encoded representations under this architecture are now 256 channels at each of 16 spatial locations in the image.

Additionally, I made one further alteration to the core NS-Outpaint architecture. Initial models were predisposed towards producing images with a grid-spaced distortions, which Odena et al. (2016) attributes to strided transposed convolutions, since they often have uneven kernel overlap in their outputs. At their suggestion, I replaced each transposed convolution layer with a nearest-neighbor upsampling layer followed by a normal convolution layer, which immediately removed the artifacts, as seen in Appendix B. It's likely that enough training would mitigate these artifacts, but given limitations on training time, faster convergence to realistic outputs was a high priority.

Multiple other additions to the architecture were tested, including the addition of NoisyAdaIn blocks inspired by StyleGAN in Karras et al. (2018), which renormalize channel activation statistics to a distribution determined by a transform on the latent state, but the best-performing architecture in the end was the downsized NS-Outpaint architecture with nearest-neighbor upsampling layers.

4.3 Two-Dimensional Generator Adaptation

This core model can only solve the one-dimensional expansion sub-problem, which is also the limit of the NS-Outpaint architecture. It is not designed to maintain feature consistency along more than one side at a time. To achieve two-dimensional expansion while maintaining the benefits of the core one-dimensional architecture, I adapted the NS-Outpaint architecture into a two-dimensional format.

Instead of taking a single map tile (128x128) and producing two attached map tiles (256x128), the two dimensional adaptation takes three attached map tiles (three quadrants of 256x256) and produces four attached map tiles (256x256).

- Since convolutional layers are spatially independent, the convolutional components of the NS-Outpaint architecture can be applied to 256x256 images instead of 256x128 images without any change.
- Since SHC modules only affect the partially-decoded features which spatially correspond to the source map tile (by passing a skip-connection from that source map tile's partially encoded features), they can also be easily adapted to the two-dimensional case. The same SHC module and parameters can be used three times to apply the skip-connection operation to the partially decoded features for each source map tile.
- To adapt RCT layers (which use sequence models to generate new latent representations along one axis) to the two-dimensional case, I apply the same RCT module and parameters from both directly adjacent source tiles. The final latent state for the target tile is just the average of the latent states generated by expanding from each directly adjacent latent state.

One crucial advantage of adapting NS-Outpaint modules, rather than using a separate approach, is that it is amenable to transfer learning, since the generator modules are unchanged, and are simply applied multiple times on different parts of the image. As such, after training the one-dimensional generator, the two-dimensional generator could be initialized with the trained parameters from the one-dimensional generator, and fine-tune from there. Initial performance of the two-dimensional generator using only the pretrained weights can be found in Appendix D.

4.4 Discriminator Architecture

Local and global discriminators were implemented based on the NS-Outpaint specification, which uses modified Wasserstein discriminators (Arjovsky et al. (2017)) to output an unbounded score for each input image, which characterizes how likely the image is to be real.

4.5 Loss Functions

The overall loss function for each generator had three components: masked reconstruction loss, global discriminator loss, and local discriminator loss. Masked reconstruction loss is computed as the L1 norm of the difference between generated and real images, with diminishing weights for pixels further into the generated region. Global and local discriminator loss contributions were weighed relatively at 0.18% and 0.02% compared to reconstruction loss, based off the relative weights in the NS-Outpaint implementation.

The generator was subject to L2 regularization with a constant of 0.00002, while discriminators were subject to gradient-magnitude penalty with a weight of 10.

4.6 Training and Evaluation

The one-dimensional generator was trained on inputs with two adjacent map tiles, and was required to reproduce the both map tiles using only information from the left tile. The two-dimensional generator was trained on inputs with four adjacent map tiles in a square, and was required to reproduce all four without using the top-right map tile. Evaluation was done based on loss values, computed regularly against a held-out validation set. Example generations were also saved at regular intervals for qualitative assessment.

5 Experimental Results

After only 10 training epochs, the one-dimensional generator could successfully extend large-scale structures, specifically shorelines, into the generated tile, though low-level details and textures were wildly inaccurate. After 50 training epochs, the one-dimensional generator could extend particular textures into the generated tile, even introducing new local details like the beginnings of new landmasses. After this point, training curves become more unpredictable as discriminator loss becomes comparable to reconstruction loss. After 100 training epochs, however, the one-dimensional generator can create remarkably realistic extensions, which can predict large-scale structures such as seas and even small-scale textures such as mountainous regions. At this point, validation loss plateaus, and the model improves at a much slower rate.

The two-dimensional generator learned even faster, since its weights were initialized from the trained one-dimensional generator. Within 10 epochs it was successfully extending landmass shapes and adding rudimentary textures. After 50 epochs, its performance on single generation steps was more realistic than the one-dimensional generators, a consequence of extrapolating from more adjacent image tiles.

With both models trained to produce realistic 1-step extensions of map tiles, they can finally be used in concert to expand a map tile indefinitely in any directions. The results are consistent between map tiles, and individual map tiles contain realistic low-level textures. However, repeated generation steps also revealed model flaws which were not visible when training on single generation steps. Firstly, as noise compounds over generation steps, the boundaries between tiles become far clearer. Secondly, both models are biased towards producing thin wavy island chains, and has trouble maintain straight lines over multiple tiles, resulting in chaotic shorelines.

Appendices C and D show examples throughout the training of each model, and Appendices E and F show the final results of repeated expansion in one and two dimensions. Given the size of these images, they require appendix pages to properly display.

6 Conclusion

The main drawback of this approach is the large model size and long training times, which make it difficult to iterate over different architectures and hyperparameters. Since I did not have the time for an exhaustive architecture search, my best-performing architectures are likely unnecessarily large, and future work could search for smaller successful model sizes.

There are two main avenues for improving results further in future work. Firstly, simply training for longer periods. The original NS-Outpaint architecture was trained for 1,500 epochs, whereas my models were trained for a maximum of 100 epochs at a rate of one epoch per hour. Secondly, these models can be trained directly on multiple generation steps, rather than just one at a time. This could even apply to training both the one-dimensional and two-dimensional generators in concert, so they must reconstruct three quadrants of an image given just one.

Over all, the application of NS-Outpaint architecture to long-distance heightmap expansion in two dimensions was a success. A reduced-size NS-Outpaint architecture with upsampling layers can successfully expand heightmaps in cardinal and diagonal directions, even with reduced training times.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

Gardias, P., Arthur, E., and Sun, H. Enhanced residual networks for context-based image outpainting, 2020.

Karras, T., Laine, S., and Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. *arXiv e-prints*, art. arXiv:1812.04948, December 2018.

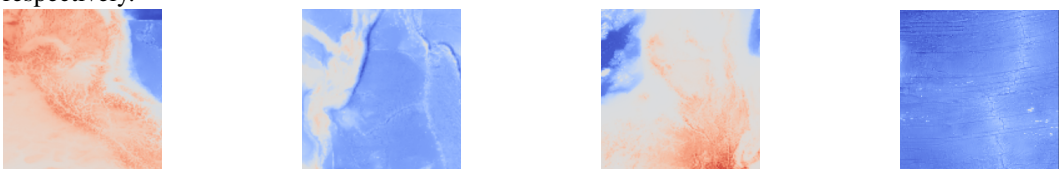
Odena, A., Dumoulin, V., and Olah, C. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

Yang, Z., Dong, J., Liu, P., Yang, Y., and Yan, S. Very long natural scenery image prediction by outpainting. *CoRR*, abs/1912.12688, 2019. URL http://arxiv.org/abs/1912.12688.

Zhang, L., Wang, J., and Shi, J. Multimodal image outpainting with regularized normalized diversification, 2019.

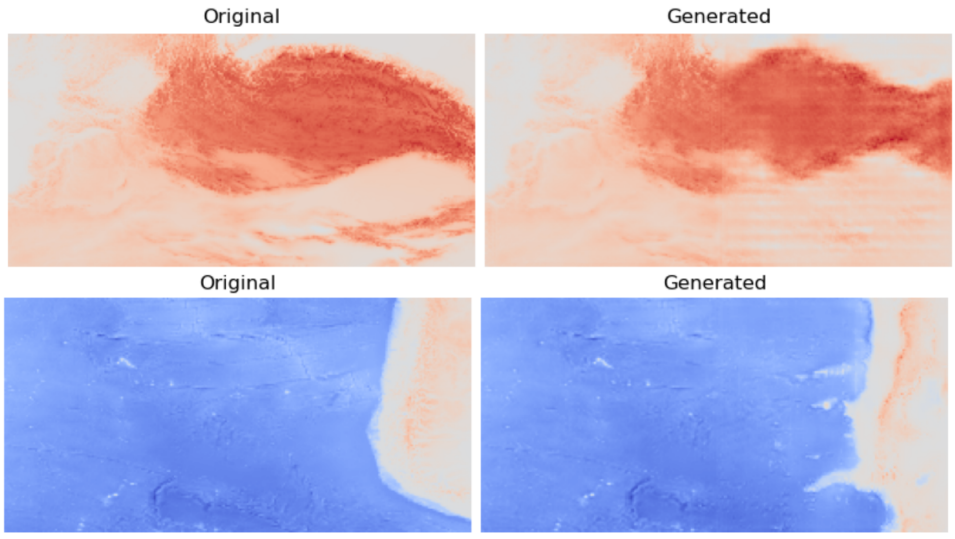
A Dataset Examples

Examples from the SRTM 15+ data source, compiled into map tiles of 20x20 degrees and 128x128 pixels. White represents sea level, while red and blue represent height above and below sea level respectively.



B Grid Artifacts: Transpose Convolution vs Upsampling

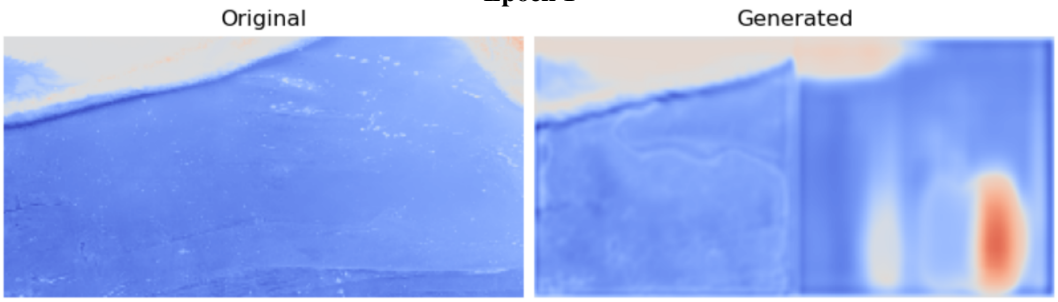
Illustration of the predisposition towards grid artifacts when using strided transpose convolutions as opposed to upsampling layers. In both cases, the generator has no access to the right half of the original image. The model using transpose convolutions (top) shows clear grid-pattern artifacts even after 60 training epochs. The model using nearest-neighbor upsampling layers (bottom) has no such issue after just 50 training epochs.



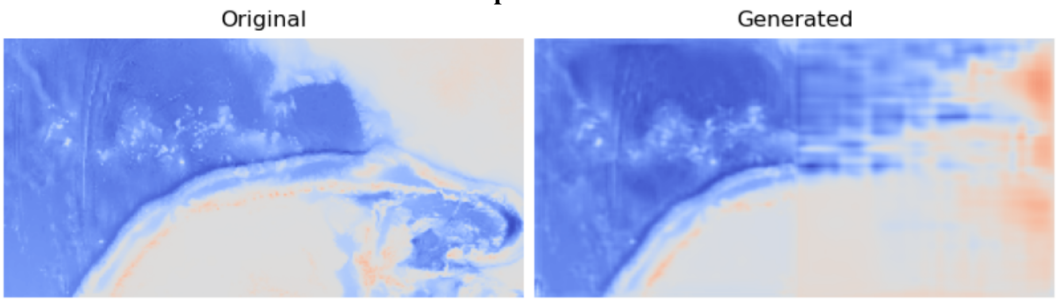
C One-Dimensional Generator Training

The examples below demonstrate the training progress of the one-dimensional generator inspired by NS-Outpaint architecture. The generator learns to extend large-scale landmasses very quickly, and slowly refines the level of detail it adds to the generated image.

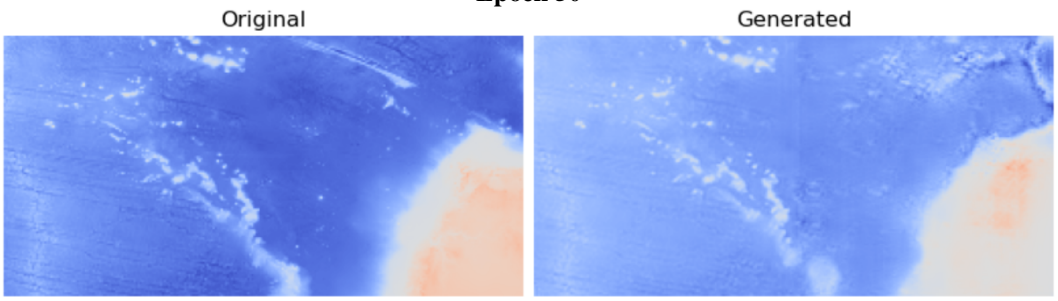
Epoch 1



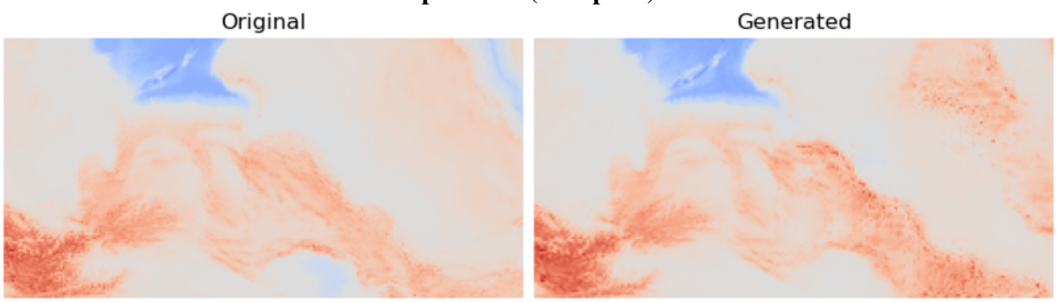
Epoch 10



Epoch 50



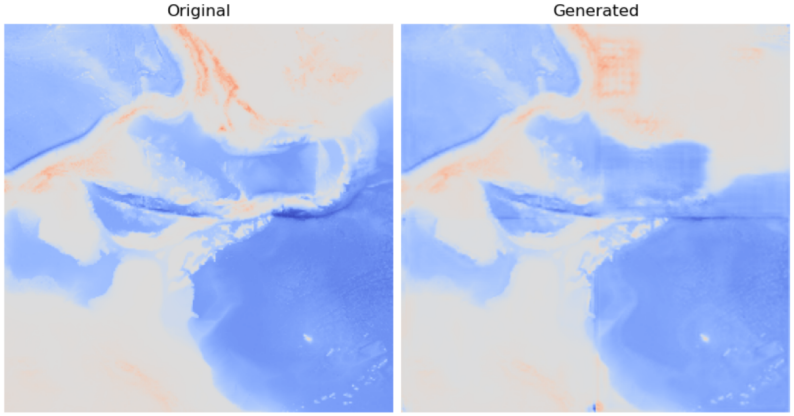
Epoch 100 (Complete)



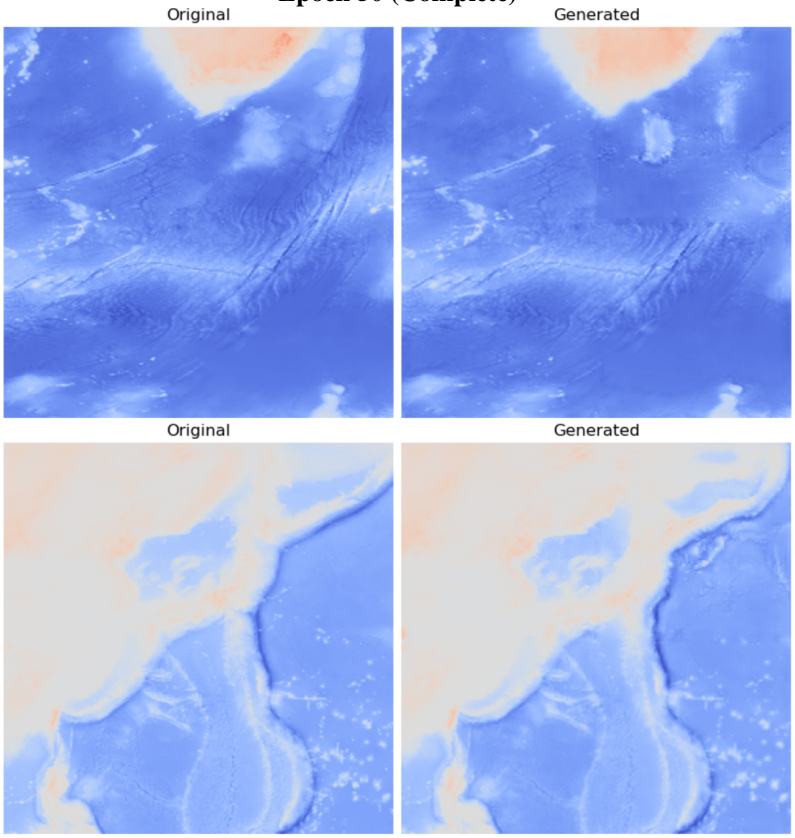
D Two-Dimensional Generator Transfer Learning

The following images illustrate the training progress of the two-dimensional adaptation of NS-Outpaint architecture. The generator has access to only the three quadrants (top-left, bottom-left, bottom-right) of the shown original image, and must generate the missing quadrant. Since generator parameters are initialized with the trained values from the one-dimensional generator, initial performance is quite good, with only a slight loss of detail and a few visual glitches along the quadrant borders to show for the transfer. By the end of training, the two-dimensional generator can generate realistic extensions to landmasses, and even predict lakes and islands which occur only in the generated region.

Epoch 1 with Pretrained Weights

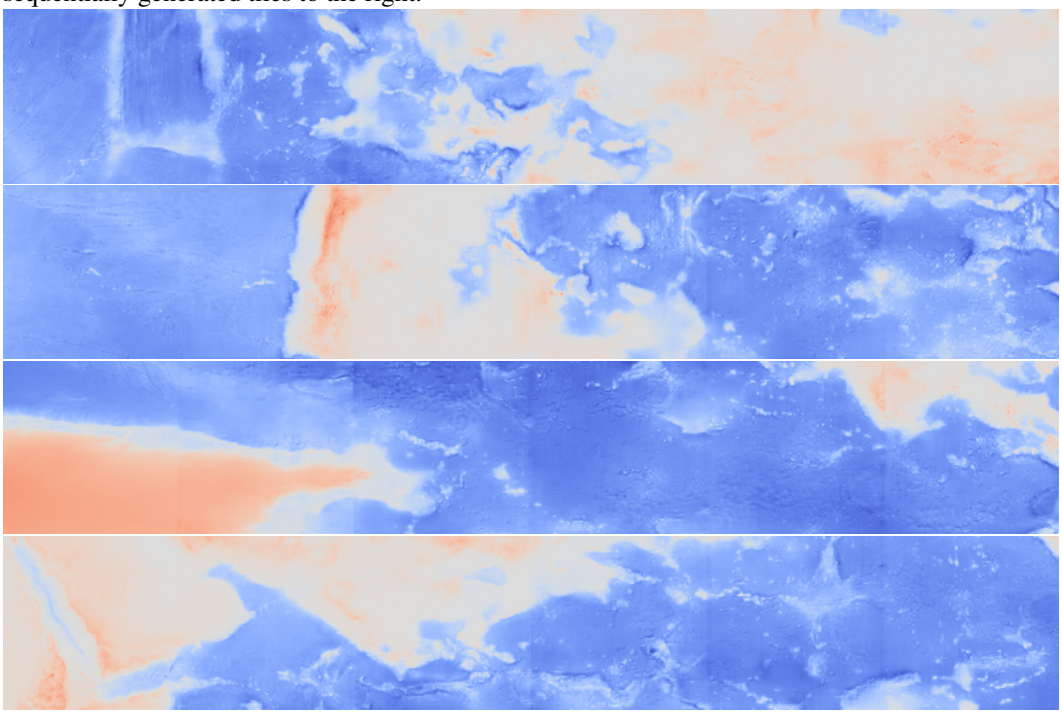


Epoch 50 (Complete)



E Long Distance One-Dimensional Generation

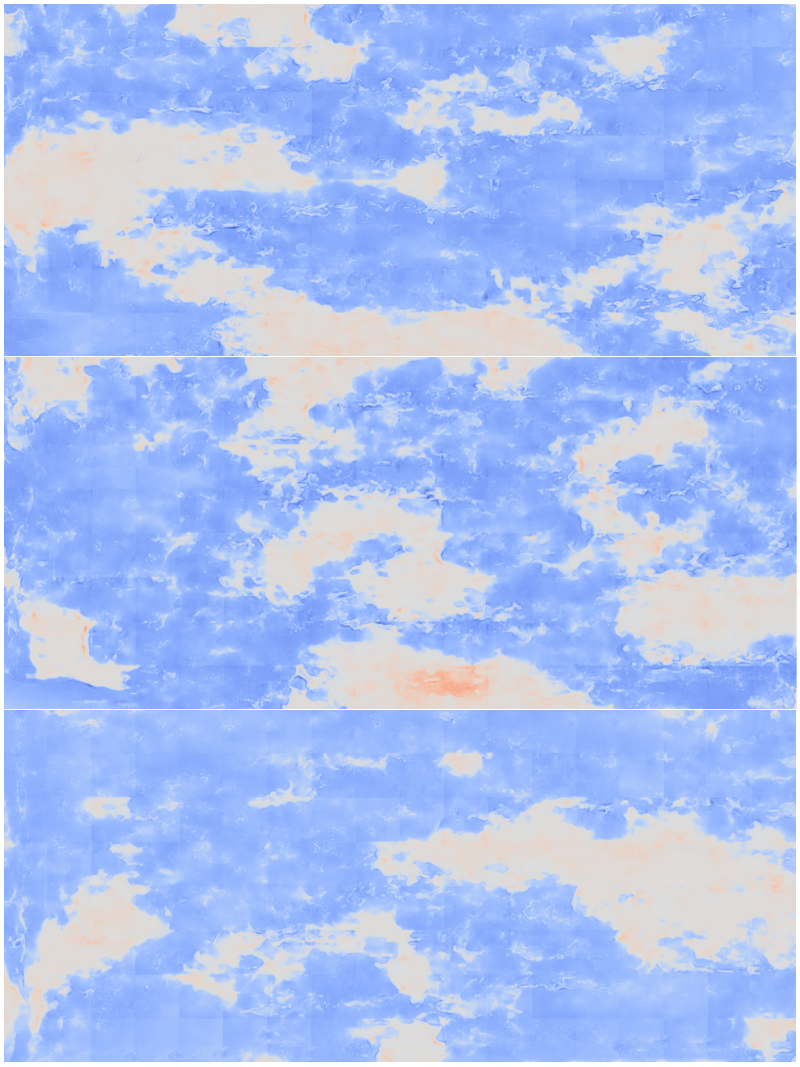
The following examples contain 6 map tiles, with one real map tile on the left, followed by 5 sequentially generated tiles to the right.



F Long Distance Two-Dimensional Generation

The following generated examples are seeded with a single real map tile in the bottom left, and has a total of 143 generated map tiles.

Generated (360 degrees longitude by 160 degrees latitude)



Real-World Comparison