Exemplar-based Terrain Synthesis

Leandro Cruz, Francisco Ganacim, Djalma Lucio, Luiz Velho, Luiz Henrique de Figueiredo
VISGRAF Lab - Instituto de Matematica Pura e Aplicada - IMPA
Rio de Janeiro, Brazil

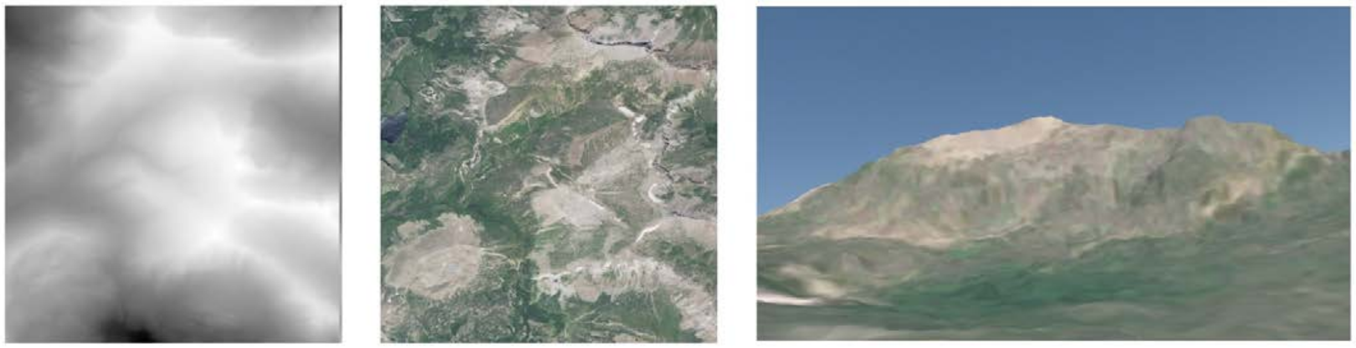

Fig. 1. Given a small exemplar of a terrain, with height data (left) and texture (middle), we synthesize a new terrain (right) adapting texture synthesis methods.

***Abstract*—The first terrain modeling techniques were introduced in the late 70's and in early 80's. They used procedural methods based on fractals to generate geometry. These models are good enough for some applications, but are not geomorphologically correct. Later, some works introduced methods that create terrains simulating physical process and using real data.**

In this work we discuss a method to generate terrains based on real samples of digital elevation models. The synthesis process is an adaptation of modern texture synthesis algorithms. Our method goes toward creating non-homogeneous terrains using some controllable features.

***Keywords*-Terrain synthesis, exemplar-based approach, real data**

I. Introduction

A common representation of the macro structure of a terrain consists of a Digital Elevation Model (DEM) and of a texture. To represent a more complex model, with more geometric details such as overhangs, vertical forms or loose rocks, it is necessary to use a more general representation [1], [2].

Another representation choice refers to the topology of the model. We can represent a terrain with the topology of a sphere (like a planet) or with the topology of a disk (like a flat landscape). To represent a planet (a model with the topology of a sphere) we need to decompose it in an atlas (a set of charts), ensuring that the transitions between glued charts are continuous (and smooth, depending on the case).

Since DEMs are basically matrices of heights, there is a seemingly trivial association between this representation and a gray-scale images. With this motivation, in this work, we aim to adapt texture synthesis methods to generate macro structure of terrain models. Hence, the DEM representation is suitable for our purposes. Furthermore, to avoid handling chart transitions we opt to create only "flat" terrains.

A. *Related work*

The first methods used to create artificial terrains were based on procedural approaches. The original method, introduced by Mandelbrot [3], [4], uses fractals to generate height fields. Later some procedural techniques based on noise were proposed [5], [6], [7].The main goal of these techniques is to create models that are perceptually good; they do not aim to create a geomorphologically correct model. More recently, new techniques were introduced that rely on simulating physical phenomena [8] or exploiting real data [9], [10] to guide the creation of new landscapes. This latter trend is the one we explore in this research.

Zhou et al. [10] exploit pieces of real datasets to create new terrain models. They use geometric information related to the geometry of ridges and valleys of the landforms to create new terrains with the same features of the input exemplar. They show that the results can be improved when considering the geometric features of the input terrains.

To generate micro structure one could use procedural methods such as described by Ebert et al. [11] or Peytavie et al. [2]. The former shows many methods based on fractals and noise. The latter uses implicit functions to models landform features, such as loose rocks, overhangs, and caves.

Wei and Levoy [12] introduced a technique to synthesize a texture, in a multiresolution way, matching neighborhoods. Lefebvre and Hoppe [13] introduced a technique to create textures using neighborhood matching. It is a parallel and controllable technique to texture synthesis that can be used to create terrain models (as the authors show in one example of the paper). Their approach was extended by Han et al. [14] to use more than one input exemplar and to allow multiscale, improving the ability to generate heterogeneous

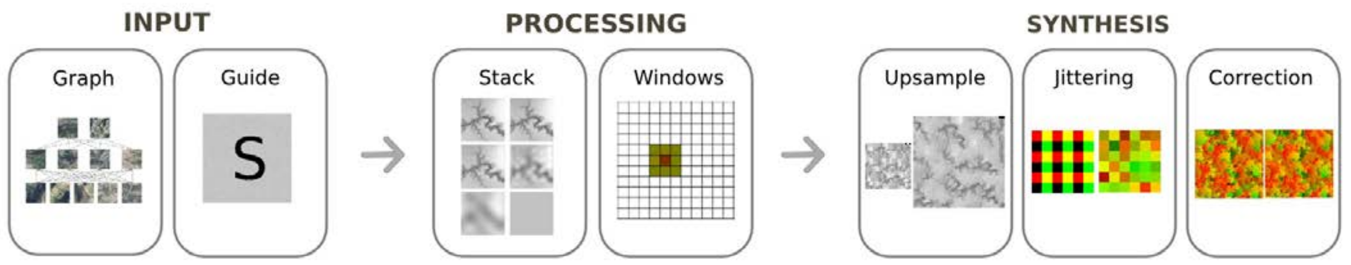

Fig. 2. Pipeline of our method

results. Their method is based on a graph of similarity between the exemplars (specified by the user). However, it is not clear how to choose good exemplars and how to define the similarity graph.

One of the main differences between the approach by Wei and Levoy [12] and the one by Lefebvre and Hoppe [13] is the multiresolution representation of the input sample. Wei and Levoy create a Gaussian pyramid of the sample, while Lefebvre and Hoppe create a Gaussian stack. The second case is more general than the first, because it contains all possibilities of pyramids. Accordingly, the results obtained using the stack are better than the ones obtained with the pyramid.

Han et al. [14] extended the previous approach by including the super exemplar concept. It is a graph with samples and their relations, which enables us to create an heterogeneous model with features in several resolutions.

II. The Terrain Synthesis Method

Figure 2 shows the pipeline of our method. We begin by specifying the input data: the set of samples (RGB+DEM), the graph (which could be empty), the jittering scale to be applied in each level, and the guide. The input is then processed to create auxiliary data structures, such as the Gaussian stacks [13]. Finally, we perform the synthesis by creating the output from the coarsest resolution to the finest.

The synthesis pass that we employ was first introduced by Wei and Levoy [12]. In that work they synthesize a texture in multiresolution. They begin by initializing the coarsest level with random colors collected from the input exemplar. For each new resolution level, the method applies upsampling followed by a correction phase.

The correction step consists on finding in the input exemplar, neighborhoods of pixels that are the closest, by some given metric, to the neighborhood of the pixel in the texture level being synthesized. This choice may not be unique, since many neighborhoods in the input can share the same distance to the texture neighborhood being corrected. In this case, we can choose randomly or base our decision on some other criterion. Besides that, we can choose not only from the set of closest neighborhoods, but from a slight larger set of the closer neighborhoods. This change introduces more variability and can give interesting results in some cases. Both nearest neighbor and approximate nearest neighbor can be implemented using a kd-tree, reducing the overall algorithm cost.

Our multiresolution representation of the exemplar is the Gaussian stack [13]. If the input is a square patch with side 2^L then the stack has $L + 1$ levels. The finest level ($l = L$) equals to the input, and the coarsest level ($l = 0$) contains only a constant value. All intermediary levels are representations of the input at different resolutions or levels of detail. Generally speaking, level l has half the amount of frequency information than level $l + 1$, that is, level l is level $l + 1$ convolved with a low-pass filter. Figure 4 shows an example of an exemplar stack.

The core of the method, according to Lefebvre and Hoppe [13], consists in the following three steps: (1) upsample the texture level; (2) jittering the coordinates, and (3) correct the pixels. The correction step may be performed more than once. This is not a convergent process. Because of this, it is not clear how many times it should be repeated for optimal results. For the generated examples shown in this paper, we have performed two correction steps.

A. Neighborhoods

The neighborhood is a $k \times k$ window around of the pixel. We choose k odd in order to have the evaluated pixel at the center of the window.

The neighborhood matching is made using an Euclidean norm in vectors of dimension $k \times k \times d$, where d is the number of channels of the data. This is a naive norm, since it does not take in account the specific domain of the problem. One

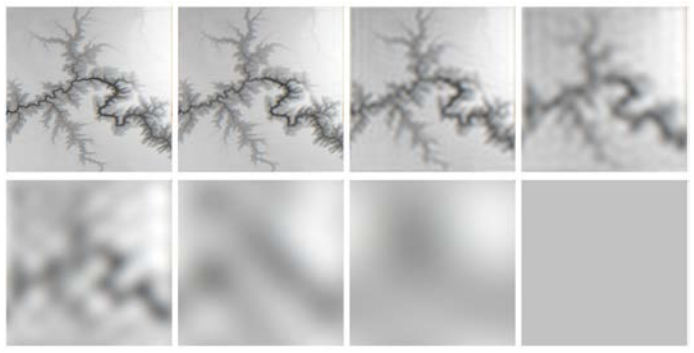

Fig. 4. The Gaussian stack of an input exemplar of size 128×128, with 8 levels.

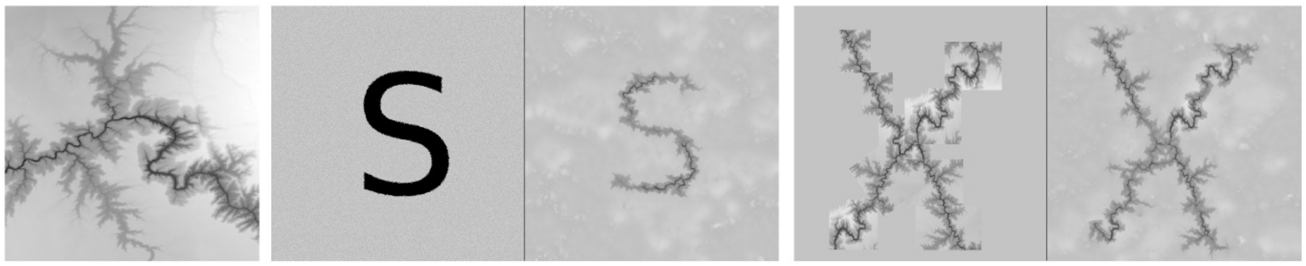
Fig. 3. Guide: (left) exemplar, (middle) sktech-based guide, and (right) gluing approach.

objective of this research is to find a better metric to match neighborhoods and to evaluate the quality of the results.

Once the neighborhood size is defined, we can improve the matching performance reducing the dimensionality of the neighborhoods with Principal Components Analysis [13]. We select the eigenvectors that represent the directions containing at least 95% of the energy of the input.

To the best of our knowledge, in all texture synthesis works the window size is an arbitrary parameter. If this parameter is small, the neighborhoods do not capture structural informations of the exemplar, and the matching is inefficient. If it is too large, the matching process is too expensive, and the result can not show a desired variability. We calculate the optimal size of the window as the one that minimizes the ratio between the PCA dimension and the neighborhood dimension ($k \times k \times d$). So far, all generated terrains have used exemplars of size 128×128 or 256×256. In these cases, we found for the optimal window size with $k \in \{3, 5, 7, 9\}$.

In many examples of texture synthesis works, the authors use toroidal exemplars to avoid the creation of spurious artifacts. These artifacts are created due to discontinuity on the borders of the exemplar. However, this is not an alternative for terrain synthesis, because the type of exemplars we use do not have this characteristic. To aleviate this problem, we use, in the same spirit as Lefebvre and Hoppe [13], larger exemplars, employing only the data inside a predefined safety margin.

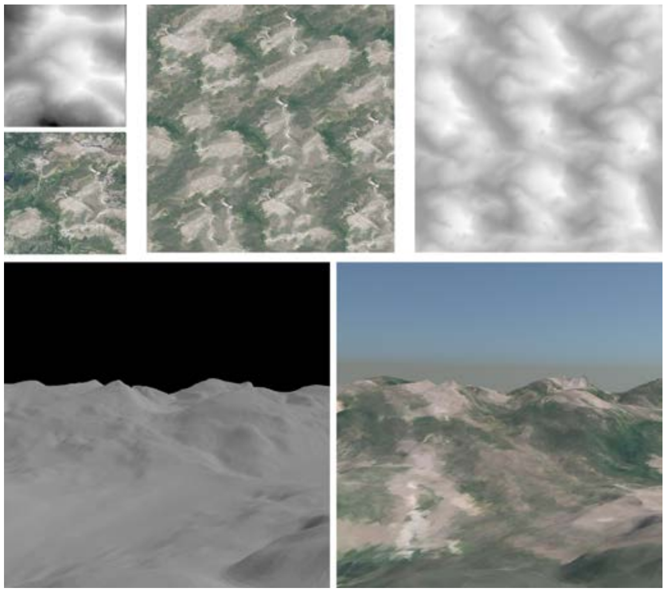
Fig. 5. Synthesized Terrain.

We have created some terrains using the presented approach. Figure 5 shows one of the results. The first line shows the exemplars and the generated result. The second line shows the rendering of a piece of this terrain. On left we are showing only geometry and in right a complete rendering.

B. Guided synthesis

To obtain more control of the terrain creation we are proposing to introduce a guide to the synthesis process. We represent the guide the same way we represent a terrain model, using a DEM. Furthermore, it is important to ensure that the guide's scale is the same of the terrain.

In the processing stage, the guide is decomposed in a multiresolution pyramid (Laplacian pyramid). For each level of the synthesis we add the respective resolution information from the Laplacian pyramid. Following, we apply the correction. This way we maintain the multiresolution approach, but we introduce a new step in the synthesis pipeline. This process offer a control of the synthesized result, ensuring that the macro structure of the guide will appear in the synthesized terrain.

The guide building is a sensitive process, because it must contain the structures existent into the exemplars. If the guide features are too different, the correction stage will create spurious structures.

Currently, we are building a guide in two different ways: (1) a sketch-based approach, and (2) gluing features of the exemplar. In the former we sketch the landforms structures (using a brushing approach). It is important to identify some features of the brush, to ensure that the guide will only have features of the exemplar. In the latter, we create a guide gluing patches of the exemplar, possibly warping them to fit the desired output. Figure 3 shows some results created with both approaches. We are still investigating the best way of building and using guides in the synthesis process.

The neighborhood representation is invariant to translations. In order to successfully execute the guided synthesis we need to be able to create neighborhoods that do not exist in the input, unless we apply some sort of transformation, such as rotations, on the input. For that reason, the translation invariance is not general enough for the matching process.

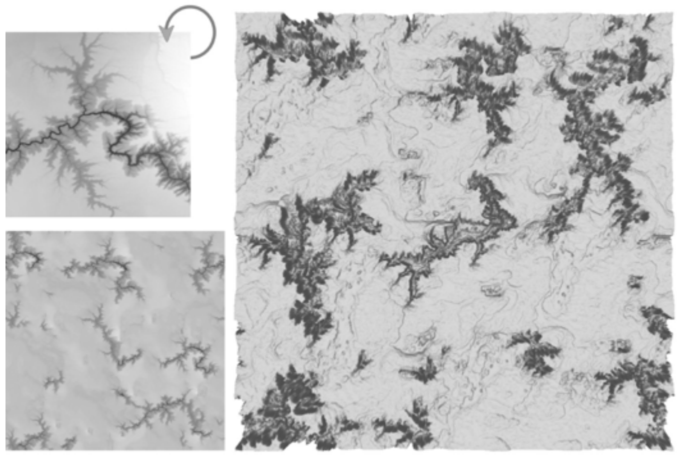

Fig. 6. Terrain created with a graph self similatity graph.

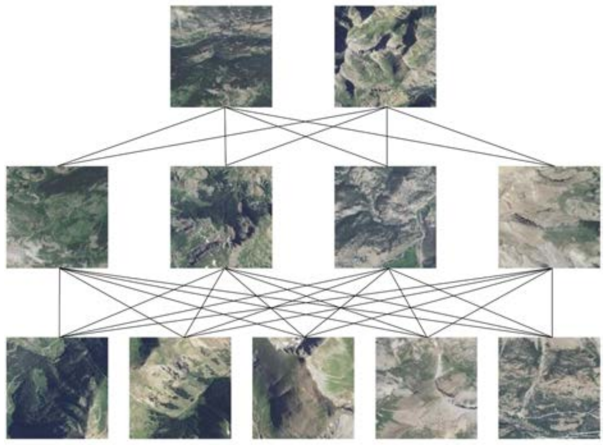

Fig. 7. Input graph.

To simulate a representation that is invariant to rotation, we have added rotated versions of the exemplars (we use 4 rotated versions). It has aggregated more informations for the correction step, and improved the quality of the results. This is not a solution. We still do not have an invariant representation to rotations. In the following stages of this research, we intend to search for better representations.

III. Future Work

One of the challenges of this work is to create ways to control the synthesis. The input data allows us to define desirable features, and the graph allows us to describe scale relations between the input exemplars. However, it is not possible to control the features positions and sizes. To improve the control we are using the guide to add macro structure to the initial texture.

However, the guide creation is a task that must improved. We have proposed two approaches to guide building. Both of these approaches are a proof of concept that indicates promising directions. We intend to follow investigating on how to build the guide keeping a good trade of between control and simplicity.

Another possibility is to create a map with the probability of samples that should be prioritized in certain regions. It could take advantage of the multiple choices in matching of neighborhoods.

Another problem is how to find exemplars with appropriated structure. In general, the exemplars have only parts of some landform in some resolution. We intend to improve our exemplars using summarization techniques [15] to create exemplars that contain more structure.

Finally, we intend to use the multiscale synthesis introduced by Han et al. [14]. Another improvement of our method consists in automatically creating the graph. The relations definition between the exemplars is not a trivial task. Figure 7 shows an example of a graph manually created. To the best of our knowledge, no work addresses the creation of the input graph. So far, we have only tested using graphs with only one exemplar with a self similarity. This example is shown in Figure 6. The top-right figure shows the exemplar. The right figure shows the mesh with the result terrain.

Acknowledgment

The first and second authors are supported by CNPq. The fourth and fifth authors a partially suported by CNPq. This work was done in the Visgraf laboratory at IMPA, which is sponsored by CNPq, FAPERJ, FINEP and IBM Brasil.

References

[1] M. N. Gamito and F. K. Musgrave, "Procedural landscapes with overhangs." In Proceedings of 10th Portuguese Computer Graphics Meeting, 2001.

[2] J. G. S. M. A. Peytavie, E. Galin, "Arches: a framework for modeling complex terrains." Computer Graphics Forum (Proceedings of Eurographics), 2009.

[3] B. B. Mandelbrot, "Stochastic models for the earth's relief, the shape and the fractal dimension of the coastlines, and the number-area rule for islands." In Proceedings of the National Academy of Sciences (USA), 1975.

[4] ——, *The Fractal Geometry of Nature*. Freeman and Co, 1983.

[5] A. Fournier, D. Fussell, and L. Carpenter, "Computer rendering of stochastic models." Communications of the ACM, 1982.

[6] K. Perlin, "An image synthesiser." ACM SIGGRAPH, 1985.

[7] G. S. P. Miller, "The definition and rendering of terrain maps." ACM SIGGRAPH, 1986.

[8] H. Hnaidi, E. Gurin, S. Akkouche, A. Peytavie, and E. Galin, "Feature based terrain generation using diffusion equation." In Proceedings of Pacific Graphic, 2010.

[9] F. Belhadj, "Terrain modeling: a constrained fractal model." AFRIGRAPH, 2007.

[10] H. Zhou, J. Sun, G. Turk, and J. M. Rehg, "Terrain synthesis from digital elevation models." IEEE Transactions on Visualization and Computer Graphics, 2007.

[11] D. Ebert, F. K. Musgrave, D. Peachey, K. Perlyn, and S. Worley, *Texturing and Modeling: A procedural approach*. Academic Press, 1998.

[12] L.-Y. Wei and M. Levoy, "Order-independent texture synthesis," 2003, stanford Technical Report.

[13] S. Lefebvre and H. Hoppe, "Parallel controllable texture synthesis." ACM SIGGRAPH, 2005.

[14] R. R. C. Han, E. Risser and E. Grinspun, "Multiscale texture synthesis." ACM SIGGRAPH, 2008.

[15] L.-Y. Wei, J. Han, K. Zhou, H. Bao, B. Guo, and H.-Y. Shum, "Inverse texture synthesiss." ACM SIGGRAPH, 2008.