Terrain Modeling: A Constrained Fractal Model

Farès Belhadj*
Université Paris 8
Laboratoire d'Intelligence Artificielle
Saint-Denis, France

Abstract

We present an algorithm mainly designed to reconstruct Digital Elevation Maps (DEM). Our approach relays on a fast and highly controllable fractal-based algorithm, we are able to create DEMs according to given constraints. Thus, these constraints can be given as scattered dataset of elevations obtained by satellite, our method supersamples this data and creates the according smooth terrain surface. Moreover, as a painter can make a sketch of his model, the final user can give or edit the main characteristics, local details and morphology, of his wanted DEM instantaneously obtaining the resulting terrain surface. Note that there is no limitation on the number of local constraints (that could vary from 0 to the number of points of the final DEM). Thus, the method we propose gives the ability to modify the global aspect (the surface behavior) as well as to constrain any local detail of the final terrain model. This paper presents the algorithm and reconstruction examples. Using a Root Mean Square Error computation between an original model and its downsampled-then-reconstructed version, the results confirm the method good behavior and show its efficiency. Other various terrain models and alternative applications are presented.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Fractals; I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems;

Keywords: terrain, fractals, surface reconstruction, surface modeling, midpoint displacement

1 Introduction

Generating realistic terrain models is an important topic in computer graphics and it has been addressed with different approaches for four decades. Terrain models are useful in many applications including virtual reality, army, geographical information systems, regional planing, geology, cinema, video games and especially flight simulations. Today with satellites, even if we can get gigabytes of more detailed real elevation datasets, the importance of the topic remains the same; we always want more details and even have more application fields like problems of data compression or restoration. Thus, the successive methods or software suites try to provide more and more detailed terrain models and sometimes their erosion along time.

This paper focuses on creating terrain surfaces starting from scattered datasets of elevations that can be obtained by satellite or other sources of geological data acquisition. We want to obtain, much more rapidly, the best approximation of the original DEMs. We are also interested in generating terrains from scratch or according to the user sketches. Thus, to create models in an interactive way, the efficiency of the chosen method becomes very important.

There are many differences between the algorithms used in terrain modeling. Nevertheless, we can classify them into families, the fractal-based methods, the fractal and physically based methods, the physically-based methods, and other miscellaneous methods that can have common points with the others.

The fractal algorithms are generally faster, produce varied relief maps (these methods can produce approximations of terrain roughness), most of time from scratch: ones are geometric [Mandelbrot 1983; Fournier et al. 1982; Miller 1986; Lewis 1987; Arakawa and Krotkov 1996] and others are procedural [Perlin 1985; Ebert et al. 2003], but often are not easily restrictable and also suffer from lack of realism. For example, hydraulic erosion is not or roughly imitated. Rare are those which are able to create terrains according to given fixed constraints. In [Kelley et al. 1988], Kelley et al. propose an approach where a generated water drainage network constrains a simple terrain surface, thus the relief is modeled through the water erosion process, an improvement of this approach was given in [Nagashima 1998]. In [Prusinkiewicz and Hammel 1993] Prusinkiewicz and Hammel describe a method, in a single integrated process, that models mountains with rivers using context-sensitive rewriting mechanisms. Otherwise, we have the fractal and physically based methods. In [Musgrave et al. 1989], Musgrave et al. have proposed one of the most realistic results by using simulations of hydraulic and thermal erosion on fractal terrains. Note that these three last methods do not give the end user the ability to fix his own constraints and that the last one can suffer from time complexity. A first solution is given by Belhadj and Audibert [Belhadj and Audibert 2005a] where they propose to generate, using fractal-based algorithm, realistic-looking terrains around simply precomputed ridge and river networks. Their second approach, presented in [Belhadj and Audibert 2005b], produces better interpolations using a bottom-up algorithm. Moreover the user is able to fix his own ridge lines, obtaining the corresponding river network and finally the entire DEM.

In addition, there is the family of the physically based methods. All of them are too complex and can not lead to the final result in an interactive way. Nevertheless, we can find variable degrees of complexity, i.e. a simple model based on velocity fields of water flow [Chiba et al. 1998] or a more complex one [Benjamin Neidhold 2005] that runs in interactive mode. Generally, these methods are used to reproduce the various erosion phenomena, the used data structures are sometimes more complex than a DEM, i.e. layered [Benes and Forsbach 2001a; Benes and Arriaga 2005] representation is used to get closer to the geological model, the results have more realistic aspect but the complexity is usually greater.

Finally, there are other various methods that can produce models under constraints and that focus on: surface approximations [Vemuri et al. 1997; Pouderoux et al. 2004], decomposition [Danovaro et al. 2003] or recomposition [Zhou et al. 2007; Brosz et al. 2006; Chiang et al. 2005; Ong et al. 2005] and deformation [Stachniak

*e-mail: amsi@ai.univ-paris8.fr

and Stuerzlinger 2005]. First can consist in generating a terrain model by computing the interpolation of point clouds or contour lines. In [Pouderoux et al. 2004] the authors managed to obtain good approximations of scattered (downsampled) elevation datasets using radial basis functions. Other methods use patches [Zhou et al. 2007], small-scale [Brosz et al. 2006] or microscopic [Chiang et al. 2005] features of existing terrains to synthesize new ones that satisfy the user macroscopic constraints (global constraints). The method presented in [Stachniak and Stuerzlinger 2005] deforms an initial fractal terrain by applying local modifications to satisfy a set of various fixed constraints. Most of these methods suffer from either time and/or manipulation complexity.

Thus, for its efficiency, a fractal-based approach is preferable and we have chosen to base a part of our work on the "Bottom-Up" approach of Midpoint Displacement methods proposed in [Belhadj and Audibert 2005b] : the Midpoint Displacement Inverse process (MDI).

We present a fractal-based algorithm, called the Morphologically Constrained Midpoint Displacement (MCMD) and based on improvements of the classical midpoint displacement methods and the MDI algorithm. Our new model satisfies several kinds of initial constraints given by the user. Our model can either rebuild DEMs from partial elevation datasets, or generate them from scratch. The given constraints are expressed as fixed elevations (local constraints) on the initial DEM and as curvature constraints on the global aspect of the final surface (i.e. bumped or ridged). The obtained models do not suffer from steep slope artifacts.

In the following, we will focus on the approach presented in [Belhadj and Audibert 2005b] and underline its drawbacks. We will explain why this method can not satisfy reconstruction constraints and give our specific implementation of the classical midpoint displacement methods (MD). This improvement of the MD methods is a part of the MCMD model, it constraints the obtained surface curvature. After we explain why constraints can not be satisfied when only the MD methods are used (cf. section 3.2) and then we present the MCMD model composed with both the MD process and an "MDI correctness" the Midpoint Displacement Bottom-Up process (MDBU). Finally, applications and results are presented and future works are discussed.

2 Background

In [Belhadj and Audibert 2005b] the authors propose an approach in two steps to generate, from scratch, realistic-looking terrain models. First, randomly generated ridge lines deform the initial mesh in order to obtain smooth ridges. Then, a simplified physically-based model is used to simulate the construction of the corresponding river network. In the second step, only the ridge lines and the river network will be kept in a DEM called skeleton-DEM. This DEM is used as an input to a fractal-based method that will finalize the generation process: the MDI algorithm add elevations to the map in order to prepare the interpolation and send the result to the according midpoint displacement method.

The method proposed in [Belhadj and Audibert 2005b] suffers from some main drawbacks. The most significant one refers to the interpolation function: we can not get any control over the interpolation even if it is in the top-down (MD) or in the bottom-up (MDI) process; thus specifying curvature constraints on the surface is impossible. This drawback does not affect the terrain generation when ridge lines and rivers networks are precomputed because the interpolation is naturally done between the ridges (high elevations) and the rivers (low elevations). But, for example, when we have either high or low elevation constraints the resulting surface remains flat. Figure 1 shows the result obtained using this previous method; here

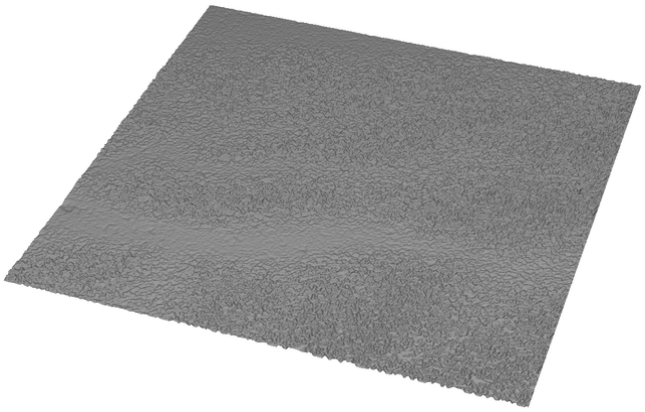


Figure 1: *Old approach drawbacks: reconstruction of Mount Washington USGS DEM after the downsampling of the original dataset to 3.3%.*

the given initial constraint map is a downsampled dataset of Mount Washington USGS (United States Geological Survey) DEM where the downsampling factor is 30. The local constraints (here 3.3% of the entire DEM) are satisfied but the generated surface does not anymore look like the original one (cf. figure 9-(a)). This is due to an incorrect computing of the interpolation in the MDI algorithm and also due to total impossibility of describing the shape of the interpolation curve. Thus we introduce in each process (MD and MDBU) completely configurable interpolation functions that give us the ability to tune the global aspect of the final shape.

The other drawbacks concern the MDI algorithm. In practice, the recursion tree that describes the top-down process does not need to be stored. Thus, in this paper, in the MDBU algorithm we save hundreds of MB ($\approx$ 200MB for a one million point DEM) by just using a function that simulates the MD process in order to get the ascendant list for a given child. On the other hand, in the MDI process, the children elevations are not well weighted in the computing of their ascendant elevation(s) (jerked modifications). Now these ascendant children are all stored in a hashtable before proceeding to elevation computing. Finally, a better implementation and managing of the FIFO queue results in an improvement of the algorithm performances.

3 Morphologically Constrained Midpoint Displacement

Under very favorable conditions, we manage to fix some local constraints and force the general aspect of surfaces produced using a midpoint displacement interpolation. In a classical top-down process, children elevations depend on their ascendants ones. Thus, if this condition is met then a smooth interpolation is guaranteed. But this is not sufficient to obtain a control over the aspect of the interpolation curve. Then, to respectively constraint the local and the global aspect of the generated DEM : favorable conditions must be reproduced by a preliminary process (here the MDBU process) to obtain smooth interpolations and controls over the interpolation computing are included in both MD and MDBU process to control the general aspect of the obtained surface.

3.1 The Midpoint Displacement

We present our implementation of the midpoint displacement methods. Here we introduce changes in the interpolation computing in

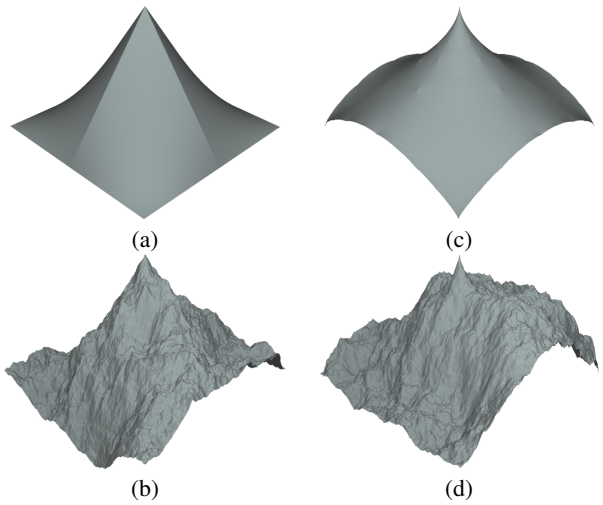


Figure 2: *Midpoint Displacement interpolation under five local constraints: the corner point elevations and the center point elevation are fixed. (a) / (b) triangle-edge resulting interpolation without / with noise; (c) / (d) diamond-square resulting interpolation without / with noise.*

order to constraint the global aspect of the final surface.

In a 1D space and for a given sub-interval, the midpoint elevation is interpolated by the elevations of extreme points; a signed random displacement δ is added to each interpolation in order to obtain a Fractional Brownian Motion [Mandelbrot and Ness 1968]; this random value is taken proportionally to the sub-interval size:

$$\delta = (su\_rand() + rt) \times rs \times 2^{-rnH} \quad (1)$$

where su_rand is a uniform pseudo-random number generator in $[-1, 1]$, rt is used to translate su_rand interval, rs is a scale factor, r is the recursion level, n is the space dimension, and H, an approximation of the Hurst's parameter, controls the fractal dimension.

In a 2D space, interpolating the midpoint elevations differs according to the used subdivision algorithm. We study variations of the triangle-edge and the diamond-square subdivision methods [Miller 1986]. A DEM is used to store values computed with the MD method. We consider a DEM as a 2D array $E[H][W]$ of cells. Each cell $E[y][x]$ is given by the couple $elevation$ ($E[y][x].e$ a two Byte integer value) and $state$ ($E[y][x].s \in \{es\_unknown, es\_known\}$). We use cell states to avoid modifications on known, or already computed, elevations. Thus by interpolating an initial DEM where favorable conditions are present (see figure 2-(a) and (c), here corner cells are set to a medium elevation and the center cell is set to the maximum elevation) we obtain two different results according to the chosen method. In figure 2-(c) and (d), a fractal map is generated by adding a random deviation δ to each interpolated elevation. Now, in order to get better control over the final surface, we bring modifications on the average computation: we weight each ascendant elevation according to the distance from the interpolated point (the child). Thus our implementation allows adjustable nonlinear interpolation:

$$\Delta(e, d) = e \times (1 - \sigma(I) \times (1 - (1 - \tfrac{d}{d_{max}})^{|I|}))$$
$$\sigma(I) = \begin{cases} 1 & I \geq 0 \\ -1 & I < 0 \end{cases} \quad (2)$$

where $\Delta(e, d)$ gives the weighted elevation used in the average computation according to e the elevation of the ascendant cell and

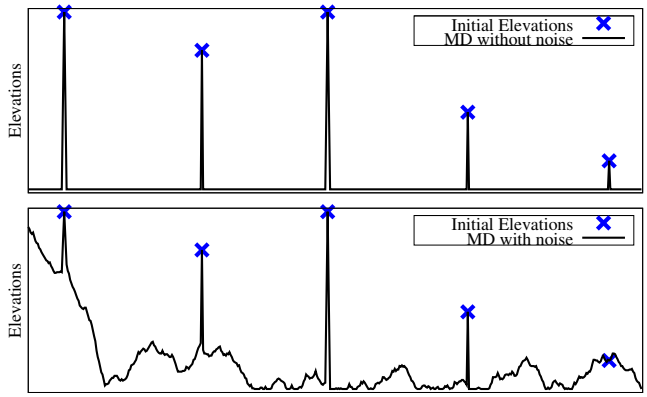


Figure 4: *The 1D-MD drawbacks appear when initial elevations are given. The top diagram shows that the MD interpolation (here without noise) produces a flat curve. The result is not better when noise is added (the bottom diagram).*

d the euclidean distance from the ascendant cell to the child cell. Here d_{max} is the DEM diagonal and I is used to tune the interpolation curve; depending on I we have: $\Delta(e, d) = e$ when $I = 0$, otherwise $\Delta(e, d)$ respectively decreases or increases according to the distance d when $I > 0$ or $I < 0$; it respectively follows a bell curve, a line or a ridged curve when $0 < |I| < 1$, $|I| = 1$ or $|I| > 1$. Finally, a displacement $d \times \delta$ (cf. equation (1)) is added to each average computation in order to obtain a fractal surface. Figure 3 shows some surface behaviors according to each parameter values. Note that when $rs = 0$, we produce an interpolated surface without fractal noise; thus rt and H have no effects over the obtained surface.

3.2 When do the MD methods produce smooth interpolations under local constraints ?

Figures 2 and 3 present various MD interpolations where the elevations of corners and center cells have been constrained. In those very particular cases, the interpolation process works well and does not produce discontinuity artifacts. When ascendant cell elevations are known before (or at the same time as) their children elevations, the interpolation process does not produce discontinuities. Thus in a 1D space, discontinuities can appear if elevations at the extremities of a sub-interval are unknown and if at the same time we know the elevations of some cells inside this same sub-interval. An example of such discontinuities is shown in figure 4; the top diagram shows the result of an MD interpolation in an interval where initial elevations are given. Figure 5 shows 2D example where some parts of the result contain discontinuities and the others don't. A centered circle and four segments (on the map diagonals) are respectively initialized to maximum and medium elevations; the bottom left segment is shifted of a unit to the left. Well interpolated elevations are those who are on the sub-rectangle diagonals. Discontinuities are clearly visible around the circle and the shifted segment.

3.3 The Midpoint Displacement Bottom-Up Process

Figure 4 shows discontinuity problems when a cell elevation is known while, at the same time, those of its ascendants are unknown. We detect these situations by testing the cell states during a simulation of the subdivision process. For these cells, we avoid discontinuities by running a bottom-up progression that computes the

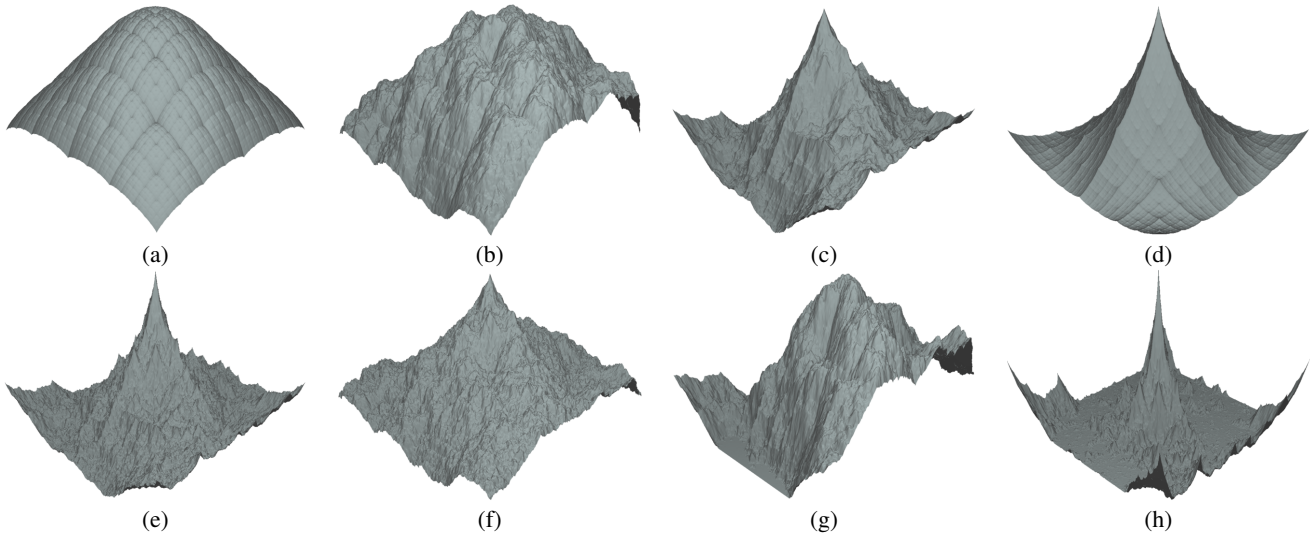


Figure 3: *DEM variations obtained with our implementation of the triangle-edge method: (a)* $I = -0.4$, H *has no effect (**ne**),* rt *(**ne**),* $rs = 0$ *(b)* $I = -0.4$, $H = 0.5$, $rt = 0$, $rs = 0.25$ *(c)* $I = -0.4$, $H = 0.5$, $rt = -1$, $rs = 0.25$ *(d)* $I = 0.4$, H *(**ne**),* rt *(**ne**),* $rs = 0$ *(e)* $I = 0.4$, $H = 0.4$, $rt = 0$, $rs = 0.15$ *(f)* $I = 0.4$, $H = 0.4$, $rt = 1$, $rs = 0.15$ *(g)* $I = -2$, $H = 0.5$, $rt = -0.9$, $rs = 0.55$ *(h)* $I = 2$, $H = 0.5$, $rt = -0.5$, $rs = 0.6$

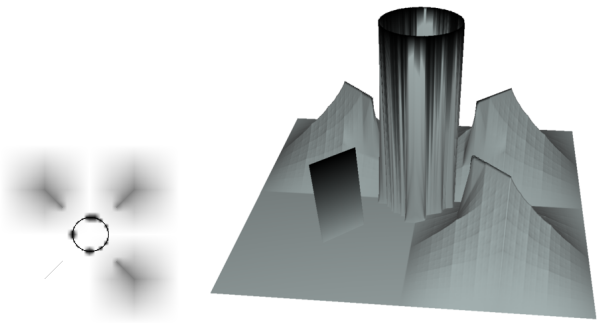


Figure 5: *MD drawbacks: the DEM is initialized with cells describing one circle and four segments; discontinuities appear around the circle and the fourth segment. The left image shows the DEM after a triangle-edge interpolation; white pixels depict the lowest elevations while black pixels depict the highest ones. The right image shows 3D view of the DEM where black material depicts the constraints.*

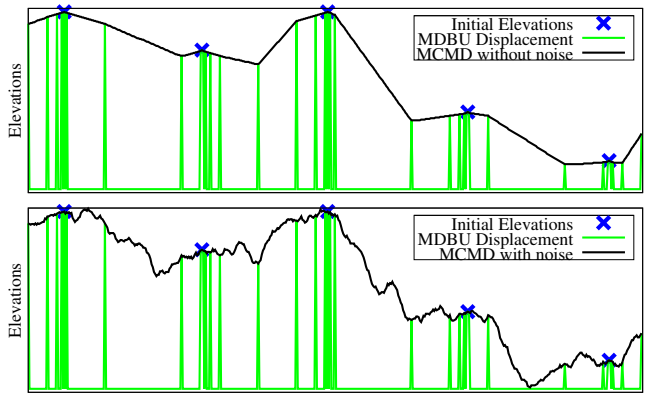


Figure 6: *Application of the 1D-MCMD method on a prefilled interval: the top curve is obtained without adding noise and the bottom one by adding noise during the MD process.*

ascendant elevations according to their children ones. Finally, the top-down process (MD) finalize the interpolation. Figure 6 shows resulting interpolations on a pre-filled initial interval after the 1D-MCMD algorithm. The initial elevations are the same as those used in figure 4. The top diagram shows an MCMD interpolation without noise. On the bottom diagram, noise is added only during the MD process.

In a 2D space, we start with an initial DEM where local constraints are given as elevations of known state cells, the DEM is pre-processed with the MDBU method and the associated MD method finalizes the interpolation. Note that MDBU depends on the chosen subdivision method (MD). Indeed, the subdivision scheme is used to obtain an ascendant list for a given child cell. Then we can store unknown-state ascendants in a hashtable and have an access to the list of their known-state children. Thus, the elevation of an ascendant cell (with an unknown state) can be computed according to those of all its children (with a known state). Figure 7 shows the process of a triangle-edge-MDBU on a 5×5 DEM where two

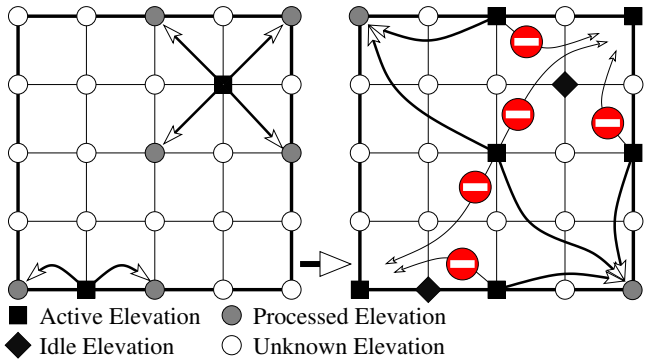


Figure 7: *Interpolation process of a triangle-edge-MDBU on a* 5×5 *DEM. For each step, "Active Elevations" are used to interpolate their ascendants ("unknown" → "processed"); then for the next step, "Active Elevations" become "Idle Elevations" and "Processed Elevations" become "Active Elevations". The process is stopped when there is no more "Active Elevations".*

initial constraints are given ("Active Elevation" on the left scheme). After the MDBU process we get ten initial constraints instead of two; the new constraints help the MD process to produce a smooth interpolation. Finally, in order to control the interpolation curve of the bottom-up process, we define Δ_{BU} (same as Δ in equation (2) but uses its own I constant called I_{BU}), an interpolation function as:

$$\Delta_{BU}(e,d) = \quad e \times (1 - \sigma(I_{bu}) \times (1 - (1 - \tfrac{d}{d_{max}})^{|I_{bu}|})) \quad (3)$$

where e is an elevation, d an euclidean distance, I_{bu} tunes the interpolation curve and σ is the same as the one defined in equation (2). Thus, the algorithm 1 gives the MDBU method details.

```
Put all initial cells in a FIFO Queue FQ;
while not_empty(FQ) do
    while E ← get(FQ) do
        for all A ascendant of E do
            if A.s = es_unknown then
                add, if does not exist, A in hashtable T;
                add E in T[A]: list of known child of A;
            end if
        end for
    end while
    for all cells A in T do
        e ← n ← 0;
        for all C known child of A in T[A]; do
            e ← e + Δ_BU(C.e, euclidean_distance(A, C));
            n ← n + 1;
        end for
        A.e ← e/n;
        A.s ← es_known;
        Remove A (and its known children) from T;
        Put A in FQ;
    end for
end while
```

Algorithm 1: *The MDBU algorithm.*

Figure 8 shows that the MCMD method suppresses the defects appearing in figure 5: (a) shows the preliminary result after the MDBU algorithm; (b)/(c) shows the (DEM)/(3D view) after the entire interpolation process and (d) shows an interpolation process where fractal noise is added.

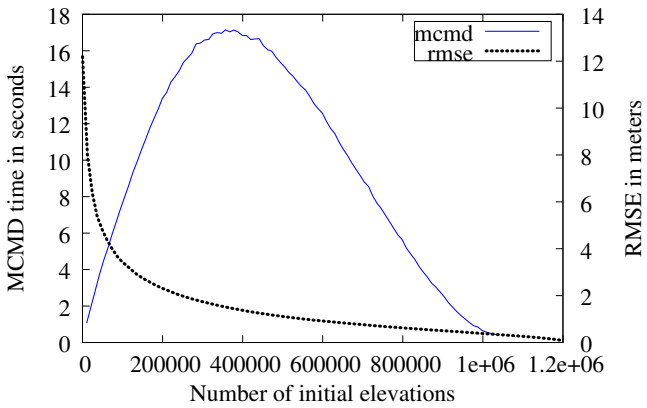


Figure 10: *Interpolating downsampled data of Mount Washington USGS DEM: MCMD time consuming and RMSE are given for each downsampling rate (from 99% to 1%).*

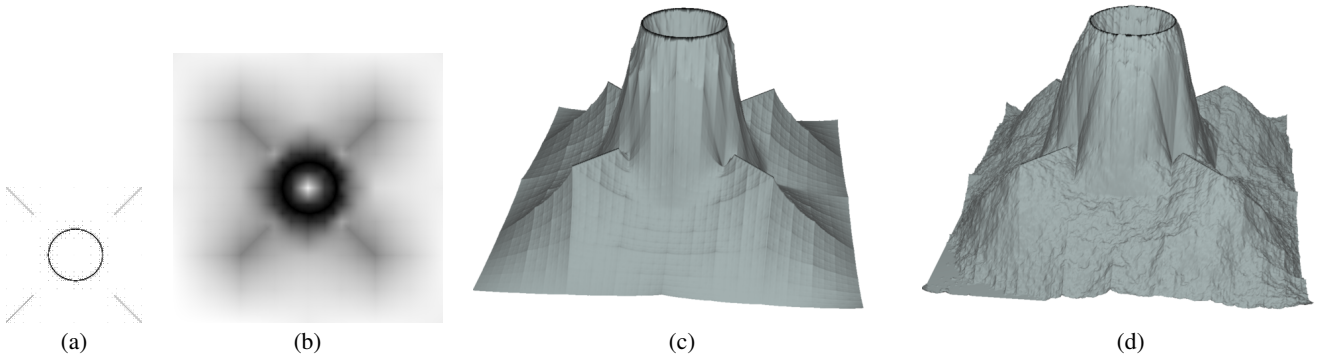
4 Applications & Results

The algorithm main application is to reconstruct downsampled DEMs or fill no-data holes in DEMs like the SRTM DEMs [SRTM 2005]. In [Pouderoux et al. 2004] the authors downsampled an USGS DEM of Mount Washington to 3.3% and use their algorithm to interpolate the downsampled dataset. New DEM is generated and compared to the original one using a Root Mean Square Error (RMSE) computation. Here we choose to make the same test in order to evaluate our method efficiency. Starting with a 1050625 elevation dataset of Mount Washington DEM (each elevation is given in two bytes, the terrain unit is equal to 0.019836 meters and we have 58405 different samples), we downsample randomly it to 35721 elevations (3.3% of the original one) and then reconstruct the dataset using the MCMD method. Figure 9-(a) shows a 3D view of the original Mount Washington DEM and figure 9-(b) shows a 3D view of the reconstructed DEM using our MCMD algorithm. As in [Pouderoux et al. 2004], the reconstructed dataset and the original one are compared using an RMSE computation. We obtain an RMSE that is about 5.8 meters against 5.04 meters in [Pouderoux et al. 2004]. The reconstruction time using our method is about 2.97 seconds (MDBU: 2.73s + MD: 0.24s)[1]. The approximation we obtain is as good as the one in [Pouderoux et al. 2004] and our algorithm is more than 150 times faster (2.97s against 531s). The curves, figure 10, confirm these good results for any downsampling rate. From a complexity viewpoint, as the MD process does not depend on local constraints (its complexity is $O(Nlog_4N)$ where $N = W \times H$) this means that the time consuming variation on figure 10 depends only on the bottom-up process. Actually, MDBU becomes time consuming when initial constraints are in deep part of the subdivision process tree. Thus, one worst case can be obtained when all the tree leaves are known and all other tree nodes not (odd index data when $W = 2^n + 1$ and $H = 2^m + 1$); here the complexity is about $O(C(log_4N)^2)$ where $C \approx \frac{N}{4}$ is the constraint number.

Please view the movie that illustrates the reconstruction process of the Mount Washington DEM. The movie is mpeg4-encoded and is available at:
`http://www.ai.univ-paris8.fr/~amsi/papers/afrigraph07/`

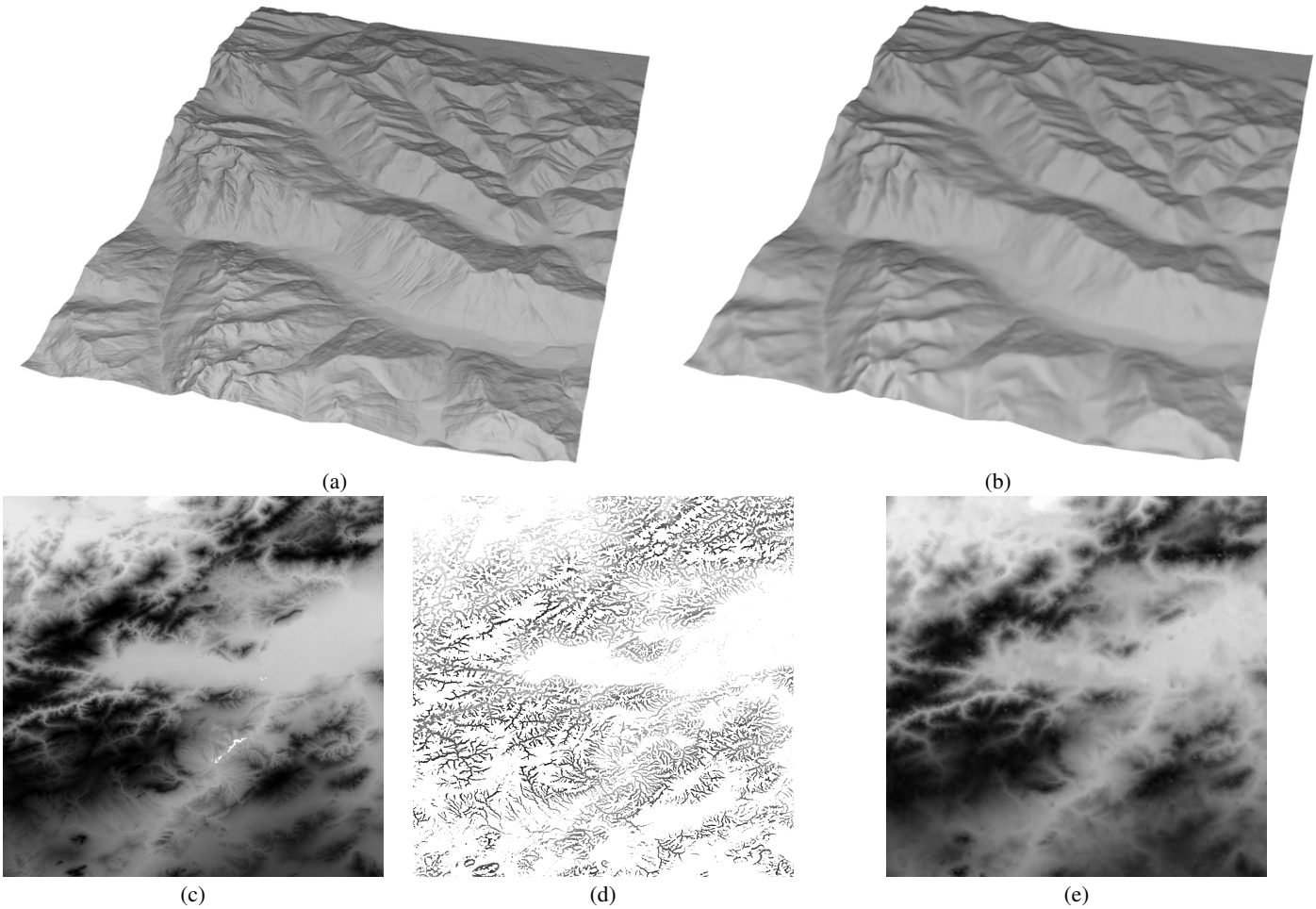
Figures 9-(c), (d), and (e) show a second interesting result in DEM reconstruction. (c) shows the SRTM DEM `N36E008.hgt` ob-

[1] All our computational times have been clocked on a Pentium IV 4Ghz with 1GB of memory



(a) (b) (c) (d)

Figure 8: *Steps and some variations of the MCMD method: (a) The DEM after the MDBU algorithm: cells are added around the initial circle and segments; (b) The DEM after a triangle-edge-MCMD interpolation: ($I_{bu} = 6$, $I = 1$); (c) 3D view of (b); (d) 3D view of a triangle-edge-MCMD interpolation with noise: ($I_{bu} = 5$, $I = -0.7$, $H = 0.4$, $rt = -1$, $rs = 0.2$).*



(a) (b)

(c) (d) (e)

Figure 9: *Reconstructing / Filling no-data holes of satellite DEMs. (a) Shows a 3D view of the original Mount Washington USGS DEM. This DEM is randomly downsampled to 35721 elevations (3.3% of the original DEM). Then the MCMD method is used to interpolate the downsampled data in order to obtain (b) in 2.97 seconds and with an RMSE = 5.8 meters. (c), (d) and (e) : Reconstructing / Filling no-data holes of an SRTM DEM: (c) The original SRTM DEM; (d) More than 90% of data from (c) are deleted using a Sobel edge detection; (e) Rebuilding the SRTM DEM by applying the MCMD method on (d). Note that no-data hole from (a) is filled in (c).*

tained from the Shuttle Radar Topography Mission database (cf. [SRTM 2005]). In (d), more than 90% of data from (c) are deleted, we use a silhouette detection in order to keep a minimum of information. And then in (e), the MCMD method is used to fill the "lost" data. We can see that there is a striking resemblance between (c) and the result of the rebuilding of (c) starting from (d). This resemblance, is also confirmed by an RMSE computation between the two DEMs where we obtain less than 8 meters. Moreover, due to data capture errors, the SRTM data contains no-data holes within, we can see a white little splash on the middle bottom of (c) which disappeared on (e).

Another main application for our algorithm is to help the user to easily generate realistic-looking landscape models. Initial constraints can be given through a DEM editor, an image editor or can be automatically generated. For example, we can use skeleton-DEMs presented in [Belhadj and Audibert 2005b]. To have a ridged behavior behind ridge line and bumped behavior behind rivers, we bring up some modifications to the MDBU process in order to differentiate "ridge cells" (es_ridge) and "river cells" (es_river) from the other "known cells" (es_known). Thus, each cell with a specific state (here es_ridge or es_river) can produce a specific behavior around it. This can be done by giving different I and I_{bu} parameters for those specific states. The left image of figure 11 shows a realistic rendering of an eroded terrain model generated using this approach. We can notice that the entire model is generated from scratch. The right image of figure 11 shows another model produced from scratch; here a Bracketed L-System[2] [Prusinkiewicz and Lindenmayer 1990] path is used as an initial constraint map. Other models generated around the user sketches are shown in figure 12. Here, to obtain the initial constraints, Africa silhouette and the text "Afrigraph '07" are added in a grayscale image under an image editor. The image is loaded as an initial DEM where white pixels are considered as unknown-state cells. We obtain 12-(a) and 12-(b) by using the MCMD method with different parameters.

Finally, our method can also be used in material modeling. We show an example in figure 13. Here, we use an image editor as in the previous application example. A grayscale image is filled with 1% of white noise. The Afrigraph logo, Africa silhouette and the text "Afrigraph '07" are added. Then this "Salt and Pepper Noised" image (cf. figure 13-(a)) is loaded as initial constraints of the MCMD method that produces (cf. figure 13-(b)) a realistic rough-coat wall painting model [Deguy and Benassi 2001a]. Note that, for this application example, we use two passes of a 5×5 median filter on the MCMD model in order to reduce the fractal noise.

5 Conclusion

This paper has presented an efficient fractal-based algorithm for terrain surface reconstruction. The method can be used to supersample DEMs, complete partial datasets of elevations, fill no-data holes in DEMs or produce realistic-looking models from scratch or by using a user sketch as initial constraints. The algorithm is efficient enough to model high resolution maps in a very short time which gives the end user the ability to work his model interactively. The topics for future work aimed at reducing the RMSE in the interpolation of downsampled DEMs in order to use our method in data compression. We are also planning to improve the method results when a contour lines are used in DEM reconstruction. Actually, we only have one behavior around a fixed elevation (a local constraint). With contour lines two behaviors are necessary in order to interpolate upper neighbors (one side of the contour line) and lower ones (the other side).

[2]Here is the description of the L-System:

θ_0 ← $\frac{\pi}{2}$
θ ← $\frac{\pi}{25}$
ω ← ++++++++++++X
X ← F[@.5+++++++++X] -F[@.4 - - - - - - - - - - -!X]@.6X

References

ARAKAWA, K., AND KROTKOV, E. 1996. Fractal modeling of natural terrain: Analysis and surface reconstruction with range data. *Graphical models and image processing: GMIP 58*, 5, 413–436.

BELHADJ, F., AND AUDIBERT, P. 2005. Modeling landscapes with ridges and rivers. In *VRST '05: Proceedings of the ACM symposium on Virtual reality software and technology*, ACM Press, New York, NY, USA, 151–154.

BELHADJ, F., AND AUDIBERT, P. 2005. Modeling landscapes with ridges and rivers: bottom up approach. In *GRAPHITE '05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM Press, New York, NY, USA, 447–450.

BENES, B., AND ARRIAGA, X. 2005. Table mountains by virtual erosion. In *Eurographics Workshop on Natural Phenomena*, 33–39.

BENES, B., AND FORSBACH, R. 2001. Layered data representation for visual simulation of terrain erosion. In *SCCG '01: Proceedings of the 17th Spring conference on Computer graphics*, IEEE Computer Society, Washington, DC, USA, 80.

BENES, B., AND FORSBACH, R. 2001. Parallel implementation of terrain erosion applied to the surface of mars. In *AFRIGRAPH '01: Proceedings of the 1st international conference on Computer graphics, virtual reality and visualisation*, ACM Press, New York, NY, USA, 53–57.

BENJAMIN NEIDHOLD, OLIVER DEUSSEN, M. W. 2005. Interactive physically based Fluid and Erosion Simulation. *Eurographics Workshop on Natural Phenomena*.

BROSZ, J., SAMAVATI, F. F., AND SOUSA, M. C. 2006. Terrain synthesis by-example. In *GRAPP*, 122–133.

CHIANG, M.-Y., TU, S.-C., HUANG, J.-Y., TAI, W.-K., LIU, C.-D., AND CHANG, C.-C. 2005. Terrain synthesis: An interactive approach. In *International Workshop on Advanced Image Technology*.

CHIBA, N., MURAOKA, K., AND FUJITA, K. 1998. An erosion model based on velocity fields for the visual simulation of mountain scenery. *The Journal of Visualization and Computer Animation 9*, 4, 185–194.

DANOVARO, E., FLORIANI, L. D., MAGILLO, P., MESMOUDI, M. M., AND PUPPO, E. 2003. Morphology-driven simplification and multiresolution modeling of terrains. In *GIS '03: Proceedings of the 11th ACM international symposium on Advances in geographic information systems*, ACM Press, New York, NY, USA, 63–70.

DEGUY, S., AND BENASSI, A. 2001. A flexible noise model for designing maps. In *International Workshop on Vision, Modelling and Visulization*, 299–308.

DEGUY, S., AND BENASSI, A. 2001. A flexible noise model for designing maps. In *VMV'01: Proceedings the Vision Modeling and Visualization Conference*, 299–308.

(a)(b)

Figure 11: *Realistic renderings of models generated from scratch (quarter million point models). The model rendered in (a) was generated in 0.61 seconds (0.5s to generate the skeleton of the ridge and rivers network and 0.11s to produce the surface using our MCMD method); the used parameters are:* $I = 1.0, I_{BU} = 6.0, H = 0.49, rt = 0.0, rs = 0.5$. *The model rendered in (b) was generated in 0.12 seconds; the used parameters are:* $I = 1.1, I_{BU} = 10.0, H = 0.48, rt = 0.0, rs = 0.5$.

EBERT, D. S., MUSGRAVE, F. K., PEACHEY, D., PERLIN, K., AND WORLEY, S. 2003. *Texturing & Modeling: A Procedural Approach*. Morgan Kaufmann, ch. 20 – MOJOWORLD: Building Procedural Planets, 565–615.

FOURNIER, A., FUSSELL, D., AND CARPENTER, L. 1982. Computer rendering of stochastic models. *Commun. ACM 25*, 6, 371–384.

KELLEY, A. D., MALIN, M. C., AND NIELSON, G. M. 1988. Terrain simulation using a model of stream erosion. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 263–268.

LEWIS, J. P. 1987. Generalized stochastic subdivision. *ACM Trans. Graph. 6*, 3, 167–190.

MANDELBROT, B. B., AND NESS, J. V. 1968. Fractional brownian motions, fractional noises and applications. In *SIAM Review*, vol. 10, 422–437.

MANDELBROT, B. B. 1983. *The fractal geometry of nature*. W.H. Freeman and Co., New York.

MILLER, G. S. P. 1986. The definition and rendering of terrain maps. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 39–48.

MUSGRAVE, F. K., KOLB, C. E., AND MACE, R. S. 1989. The synthesis and rendering of eroded fractal terrains. *Computer Graphics 23*, 3, 41–50.

NAGASHIMA, K. 1998. Computer generation of eroded valley and mountain terrains. *The Visual Computer 13*, 9–10, 456–464.

ONG, T. J., SAUNDERS, R., KEYSER, J., AND LEGGETT, J. J. 2005. Terrain generation using genetic algorithms. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, ACM Press, New York, NY, USA, 1463–1470.

PERLIN, K. 1985. An image synthesizer. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 287–296.

POUDEROUX, J., GONZATO, J.-C., TOBOR, I., AND GUITTON, P. 2004. Adaptive hierarchical rbf interpolation for creating smooth digital elevation models. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*, ACM Press, New York, NY, USA, 232–240.

PRUSINKIEWICZ, P., AND HAMMEL, M. 1993. A fractal model of mountains with rivers. In *Proceeding of Graphics Interface '93*, 174–180.

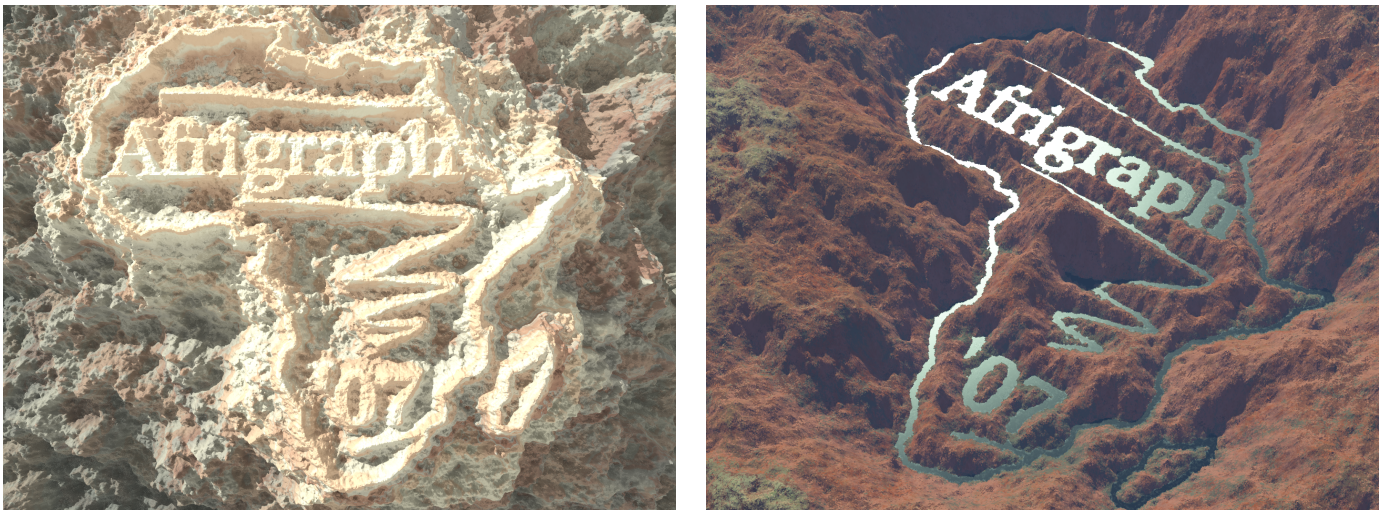
PRUSINKIEWICZ, P., AND LINDENMAYER, A. 1990. *The algorithmic beauty of plants*. Springer-Verlag New York, Inc., New York, NY, USA.

SRTM. 2005. Shuttle radar topography mission. http://srtm.usgs.gov/.

STACHNIAK, S., AND STUERZLINGER, W. 2005. An algorithm for automated fractal terrain deformation. In *Proceedings of the 7th international conference on Computer Graphics and Artificial Intelligence*, Ed. D. Plemenos, 64–76.

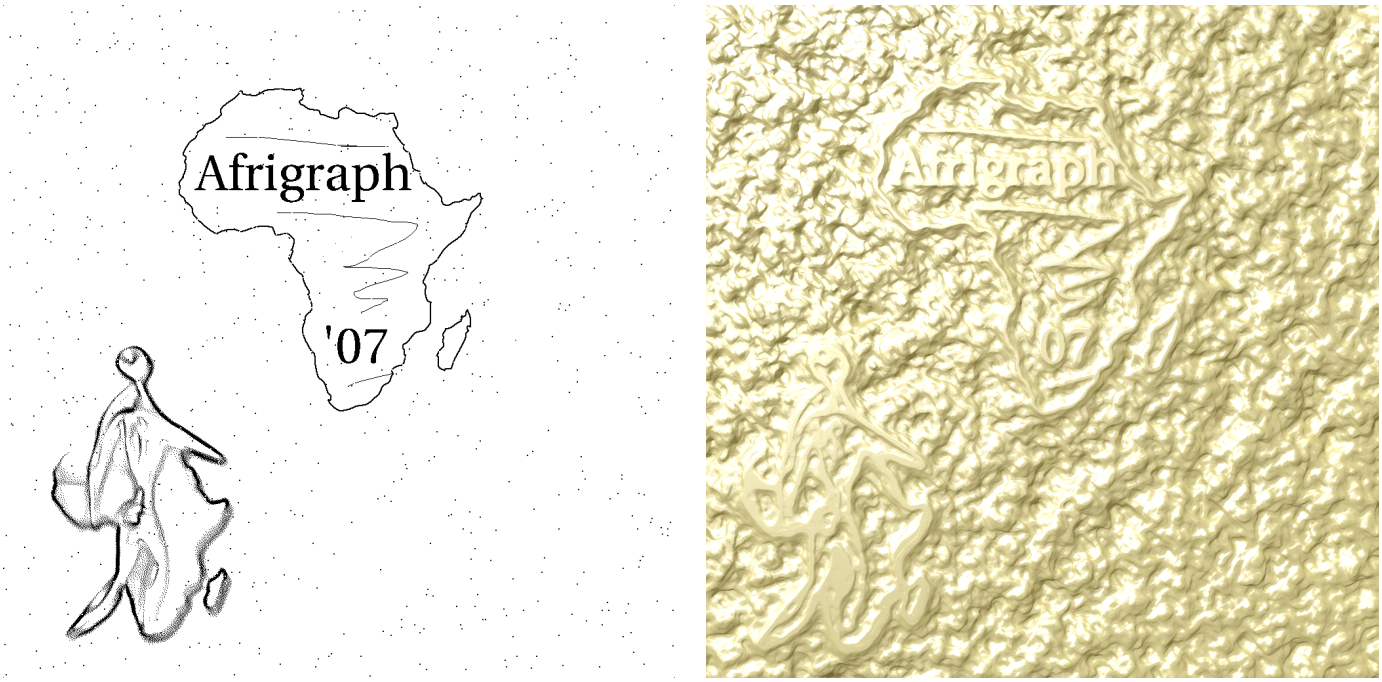
VEMURI, B. C., MANDAL, C., AND LAI, S.-H. 1997. A fast Gibbs sampler for synthesizing constrained fractals. *IEEE Transactions on Visualization and Computer Graphics 3*, 4, 337–351.

ZHOU, H., SUN, J., TURK, G., AND REHG, J. M. 2007. Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13*, 4 (July/August), 834–848.



(a) (b)

Figure 12: *Realistic renderings of models (a million point) generated using a user sketch as local constraints. The model rendered in (a) was generated in 0.25 seconds; the used parameters are:* $I = -1.1, I_{BU} = -20.0, H = 0.45, rt = -0.2, rs = 1.0$. *The model rendered in (b) was also generated in 0.25 seconds (here we use the same local constraints); the used parameters are:* $I = 1.1, I_{BU} = 4.0, H = 0.45, rt = 0.0, rs = 1.1$.



(a) (b)

Figure 13: *Application of the MCMD method in material modeling. Here a rough-coat wall painting effect is reproduced. This million point model was generated in 3.66 seconds (0.34s for the MCMD method and 3.32s for the median filter) ; the used parameters are:* $I = 3.0, I_{BU} = 7.5, H = 0.5, rt = 0.1, rs = 1.3$.