Dynamic Terrain Generation Based on Multifractal Techniques

Joost van Lawick van Pabst
Hans Jense
TNO Physics and Electronics Laboratory
The Hague, The Netherlands

Abstract

The work, described in this paper, covers three topics: real-world terrain analysis, synthetic terrain generation based upon the results of the analysis stage, and dynamic terrain generation. Each topic involves the theory of multifractals. First, a summary of four known fractal-based techniques for terrain generation is presented including fractional Brownian motion, Mid-point displacement, Iterated Function Systems and the multifractal formalism. The multifractal formalism was chosen for both the analysis of real-world terrain data, as well as the generation of synthetic terrain surfaces. The authors implemented a multifractal terrain analysis algorithm that captures terrain characteristics of real-world data, into five parameters. These parameters were put into a multifractal terrain generation algorithm that produced synthetic terrain with features similar to those in the terrain that was analysed. Also, an algorithm for zooming in on synthetic terrain was developed. Finally, an application was developed that generates terrain dynamically.

1 Introduction

The work described in this paper aims at the investigation of fractal based techniques for the representation of terrain models. From the earliest point in fractal research, applications for terrain generation were developed [1, 2, 3, 4]. In particular we are interested in the possibilities to use these methods for the on-line generation of terrain models in real-time for visual simulation applications. In visual simulation systems, the geometrical model of the world lies at the basis of the system. This representation of the environment is the input based on which the images are rendered. It contains the mathematical definition of every element in the simulated world. Terrain databases for visual simulation purposes are usually based on data that have been acquired either by surveying, by digitizing existing maps, or by spatially sampling real-world terrain using, e.g., aerial photography or satellite born height measuring equipment.

In order to achieve real-time generation of images, based on these very large databases, techniques such as spatial indexing, database culling, and level-of-detail management have been developed [5]. These allow the image generation process to minimize the amount of data that has to be retrieved from the database, and thus to maximize the rendering speed.

For certain visual simulation applications, either military or in the civil domain, difficulties in the data-acquisition process severely limit the availability of detailed terrain databases. For instance, for the simulation of remotely controlled planetary rover vehicles on Mars, only very small parts of that planet have been captured in 3D terrain models of sufficient detail. In order to evaluate the performance of new rover designs, or training operators, databases of wider areas with different terrain features are required. It is often not required that these databases represent actual real-world terrain, as long as the characteristic features are represented correctly.

Another way of representing natural phenomena uses the theory of fractals [1]. This mathematical theory is based on numerous iterations or recursions of a simple rule, producing complex shapes. By introducing small random deviations in every iteration, the resulting shape becomes irregular. From a fractal definition it is possible to extract output at any required resolution, higher output resolutions requiring more iterations. In this way, fractal-based methods provide implicit level-of-detail control.

It is possible to synthesize realistically looking representations of natural phenomena, e.g., terrain models, based on fractal techniques [2]. Conversely, real world data can be analysed and their fractal properties extracted [3, 6]. These fractal parameters, which characterise certain features of, e.g., a terrain, can then be used to control the terrain generation process. Thus, a synthetic but visually realistic terrain model can be generated when only limited amounts of real-world terrain data are available.

Finally, because the basic rule is very simple, the amount of input data needed is very small. The realistic output comes from the numerous iterations of the rule. This property of fractal algorithms has been recognised as very important for data compression applications [7].

The above mentioned aspects of fractal methods, i.e., ability to represent natural terrain features, implicit level-of-detail control, and compact input data representation, make them interesting candidates for the on-line generation of terrain models as opposed to the off-line construction of geometric terrain databases. However, the generation of fractal based terrain models requires considerable processing power, because the number if iterations of the basic rule can be large. Therefore, the parallellisation of fractal based terrain generation algorithms in order to boost their performance, is an important issue.

In the remainder of this paper, we first present a brief overview of various fractal based methods for terrain generation in section 2. Section 3 describes our implementation of an experimental terrain analysis and generation system, based on the multifractal approach. We also provide preliminary results of synthetic Martian terrain models. Finally, in section 4, the possibilities for parallellisation of the generation algorithm, which is a subject for future research, are discussed.

2. Fractal techniques for terrain modelling

In this chapter, a summary of fractal theory and algorithms for constructing fractal landscapes, is presented. In 2.1, fractal Brownian motion is discussed. Paragraph 2.2 gives an overview of the Mid-point displacement technique and Iterated Function Systems (IFS) are presented in 2.3. The last paragraph, 2.4, treats the multifractal formalism. Multifractals are especially suitable to capture the characteristics of terrain features and to generate new terrain based on these characteristics.

2.1 Fractional Brownian motion

The characterization and modelling of random fractals can be based on generalisations of fractional Brownian motion [1], denoted as fBm. Fractional Brownian motion is an extension to the concept of Brownian motion. A fractional Brownian motion is a single valued function $V_H(t)$. Its increments $\Delta V_H(\Delta t) = V_H(t_2) - V_H(t_1)$ have a Gaussian distribution with variance:

$$\langle \Delta V_H^2(\Delta t) \rangle \propto \Delta t^{2H} , \tag{1}$$

where $\Delta t = |t_2 - t_1|$ and the angular brackets denote averages over many samples of $V_H(t)$. The parameter H has a value $0 < H < 1$. The value $H = {}^1/_2$ gives Brownian motion with $\Delta V^2 \propto \Delta t$, and its increments are independent. For $H > {}^1/_2$, the increments have a positive correlation and for $H < {}^1/_2$, a negative correlation. If the time scale Δt is changed by a factor r, then the increments ΔV_H change by a factor r^H,

$$\langle \Delta V_H^2(r\,\Delta t) \rangle \propto r^H \langle \Delta V_H^2(\Delta t) \rangle . \tag{2}$$

This non-uniform scaling is known as self-affinity. Uniform scaling is known as self-similarity. The fractal dimension D for a trace of $V_H(t)$ is:

$$D = 2 - H \tag{3}$$

The random function $V(t)$ is characterised by its spectral density function $S_V(f)$. If $V(t)$ is Gaussian white noise, $S_V(f) \propto 1/f^0$. For simple power laws where $S_V(f) \propto 1/f^\beta$ with $1 < \beta < 3$, it can be shown that $\beta = 2H + 1$. This is a direct relation between the

spectral density S_V and the fractional Brownian motion function V. For Brownian motion, $H = {}^1/_2$. which corresponds to a spectral density function $S(f)$ with $\beta = 2$ or $1/f^2$ noise.

Because a trace of fractional Brownian motion looks very much like a mountain horizon, it can be used for artificial terrain generation. To do this, $V_H(t)$ must be replaced by $V_H(x,y)$ representing altitude, where x and y represent the coordinates in the plane. Parameter t is replaced by r with $\Delta r^2 = \Delta x^2 + \Delta y^2$. The fractal dimension of the surface is:

$$D = 3 - H \tag{4}$$

Terrain can be numerically generated using the relation between function V and its spectral function S_V by filtering a surface of samples of white noise to give directly the desired power law $1/f^\beta$ for variations in any desired direction. Filtering can be done using the Fast Fourier Transform [1, 8].

2.2 Mid-point displacement

The mid-point displacement method (MPD) is a recursive generating technique which approximates fBm. Given a square grid of unit size δ, the MPD method generates a new set of interpolation points to the grid, making its resolution $\delta/\sqrt{2}$ (see Figure 1). Displacing again results in a resolution of $\delta/2$ etc. In each stage, resolution is scaled with a factor $r = 1/\sqrt{2}$. A new point is calculated from its neighbours by linear interpolation and random noise is added to it.

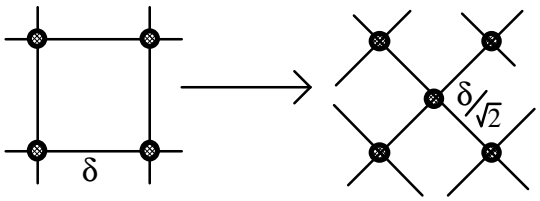


Figure 1: *Resolution scaling by a factor* $\delta/\sqrt{2}$

The MPD method sometimes causes some defects in the resulting terrain, e.g., unnaturally large jumps in the terrain height. One way to overcome these artifacts is to add displacements with suitable variance to all of the points and not just the midpoints This process is called *successive random addition*. In this case, twice as many displacements are necessary compared to the basic MPD method. Details and algorithms can be found in [1, 2].

2.3 Iterated function systems

IFS theory is a practical tool for the generation of images including clouds, smoke and landscapes. The method is developed by *Barnsley* [3]. It concerns deterministic geometry and is an extension to classical geometry. IFS uses affine transformations, i.e., scaling, rotations, and translations, to express relations between parts of geometrical objects. IFS can be used to *extract* characteristic features from an object; to *model* the geometry of an object and to *visualise* the object.

To extract features from objects and model them, an algorithm based on the so called *Collage Theorem* is used in IFS. The output of this algorithm is *IFS code*. IFS code is a set of affine transformations, which define the geometry of the underlying model, and some additional parameters. Given an IFS code, there is a unique geometrical object A associated, called the attractor of the IFS. There is also a unique associated measure μ which may be thought of as a distribution of material of the object. (A,μ) defines the underlying model associated with the IFS code.

The second purpose of IFS is the rendering part. IFS code (the affine transformations) forms the input of the IFS rendering algorithm and, based on random iterations, the rendering algorithm produces a deterministic geometrical object together with rendering values. Given a viewing window, a resolution and an IFS code (A,μ), the IFS image can be calculated and rendered.

2.4 Multifractal algorithms for terrain generation

The terrain models generated by the fractal techniques from the previous sections can all be classified as so-called *monofractals*. Monofractal sets are characterised by a single fractal dimension. *Multifractal* sets on the other hand can be divided in a number of subsets, each with its own fractal dimension.

For an explanation of the multifractal approach we refer to Figure 2 which shows a so-called turbulent discrete cascade model. The cascade model has originally been used to model turbulent (fluid) flows [10]. In particular, singularities (sudden changes in behaviour of the turbulence) can be modelled with the multifractal technique. In the context of terrain modelling, mountain peaks can be modelled as singularities in the landscape.

A turbulent cascade model is scale invariant and has an energy quantity at each level which is equal to the overall energy flux. At each successive step, the turbulent energy flux is distributed over smaller scales according to some probability density function and renormalised. At each step, energy fluxes are getting stronger, remain equal, or getting weaker according to some multiplicative incremental factor. In the end, this leads to the appearance of a full hierarchy of levels of “survival” of the flux energy, hence of a hierarchy of dimensions of the set of survivors for these different levels.

A raw multifractal field, e.g., an elevation grid with (square) resolution λ, denoted with $\varepsilon_{\lambda,}$ does not look like a terrain surface (compare Colour plate 1). It must first be fractal-integrated to obtain a map that can be interpreted as terrain surface. Fractal integration, described in §2.1, introduces the $1/f^{\beta}$ relation in the multifractal field ε_{λ}.

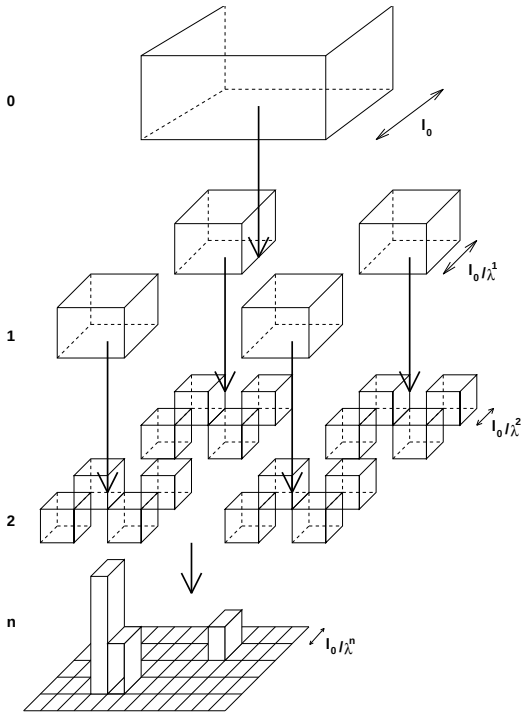


Figure 2: *A discrete multiplicative cascade process (adapted from [10])*

Multifractal fields are multi-scalable which means that different statistical moments of the field are controlled by different power-law relations:

$$\left\langle \varepsilon_{\lambda}{}^{q} \right\rangle = \lambda^{K(q)} \left\langle \varepsilon_{1}{}^{q} \right\rangle \tag{5}$$

where q are the moments of ε_{λ}. The moment scaling function $K(q)$ describes how the statistical properties of each moment behave under isotropic dilations and contractions. The moment scaling function characterises the multifractal field ε_{λ} and will be used in the estimation process when terrain surfaces are analysed. According to [10], $K(q)$ can be parameterised with two parameters, α and C_1, which will be explained below.

If the logarithm is taken of the multifractal field ε_λ, the multiplicative increments (i.e. the factors controlling the increase or decrease of the energy fluxes in the cascade process) become additive increments. Let Γ_λ be the logarithm of ε_λ. Γ_λ is called the additive increment generator of the field. If the increments are random variables, Γ_λ corresponds to sums of random noises. If these noises have unit mean and unbounded variance, the normalised sum of these noises $x_1, x_2,..., x_n$ tends towards a Lévy distribution:

$$S_n = \frac{x_1 + x_2 + \ldots + x_n}{n^{1/\alpha}} \tag{6}$$

where S_n represents a random Lévy variable. The Lévy parameter α ranges between 0 and 2. If $\alpha = 2$, S_n becomes a lognormal distribution. In that case, there will be only small fluctuations (singularities) in the multifractal field. If $\alpha < 2$, the variances are not necessarily bounded anymore, so the probability of fluctuations increases as α decreases and more singularities in the energy field are expected.

Thus, the first stage in obtaining a multifractal field ε_λ, is to generate a Lévy noise field with a certain α which is subsequently normalised.

The second parameter in the moment scaling function *K(q)* that controls terrain characteristics in a multifractal field, is the code-dimension of the mean of the field, represented by C_1. It characterises the sparseness of the mean of the field. For Lévy noise generated multifractal fields, the moment scaling function is defined as:

$$K(q) = \frac{C_1}{\alpha - 1}\left(q^\alpha - q\right) \tag{7}$$

The second stage in obtaining a multifractal field ε_λ is to filter the output of stage one to get a multi-scaling behaviour according to the characteristic function *K(q)*. The third stage is to exponentiate and normalise the output of stage two to obtain a multifractal field ε_λ. The last stage is to fractal integrate ε_λ to obtain a terrain surface-like map [10].

3 Implementation and results

We adopted the multifractal technique because it provides a framework for both the analysis of real world data, as well as a generator to construct synthetic terrain. The analysis stage provides estimates for the multifractal parameters that can then be used in the generator to build new synthetic terrain with similar characteristics as the analysed terrain. Furthermore, we developed a module that can perform a zoom operation on the newly generated terrain, and we constructed a setup for dynamic terrain generation.

3.1 Synthetic terrain generation

Our implementation of the multifractal terrain generation algorithm is based on IRIS Explorer [12]. IRIS Explorer is a modular interactive visualisation environment based on the data flow paradigm. It is a powerful and versatile visualisation system that allows researchers to quickly and easily explore their data by interactively creating, modifying, and executing there own visualisation applications.

The multifractal terrain generation process consists of two Explorer modules: the *Multifractal Generator* and the *Integrator*. In the Multifractal Generator, a raw multifractal field $\varepsilon_\lambda$is generated that is exactly determined by four parameters: α which determines the occurrence of peaks (singularities) in the raw terrain, C_1 which controls the sparseness of the mean terrain height ("roughness"), S as an initial number (seed) for the random number generator, and R for the (square) resolution of the terrain surface. For each seed the random generator produces exactly the same sequence of random numbers that are used in the generation process and thus each set of four parameters exactly specifies the raw terrain surface patch.

Computationally, the multifractal field ε_λ is generated according to a procedure adapted from Pecknold et.al. [10]. They present a method for the direct generation of an extremal Lévy-stable variable $S(\alpha)$ with index α. We used this method for equation (6). To implement the second stage of generating a multifractal field, as discussed in paragraph 2.4, the grid of normalised $S(\alpha)$ noises is transformed to the Fourier domain using a FFT. To obtain multi-scaling behaviour, the transformed $S(\alpha)$ is weighted with a factor $w(k) \propto |k|^{-d/\alpha'}$ $(1/\alpha + 1/\alpha' = 1)$. Finally, the noise must be band-limited between $[1,\lambda]$. Therefore it is multiplied by a filter $f(\lambda,k)$ which has value 1 for $|k| < \lambda$ and decays exponentially for $|k| > \lambda$. Then, an inverse Fourier transformation is applied to get the (additive) generator from paragraph 2.4:

$$\Gamma_\lambda = F^{-1}\left\{ N_1 \, S(\alpha) |k|^{-d/\alpha'} \, f(\lambda, k) \right\} \tag{8}$$

N_1 is a normalisation factor. ε_λ becomes:

$$\varepsilon_\lambda = N_2 \exp(\Gamma_\lambda) \tag{9}$$

The raw multifractal field ε_λ is then fractal integrated to introduce the $1/f^\beta$ relation. This is done in the second Explorer module, the Integrator. The integration process is controlled by the H parameter that is related to the β parameter by $\beta = 2H + 1$. The larger H becomes, the more smoother the terrain will be. If $H = 0$ then no smoothing is performed and the raw multifractal field is displayed. Fractal

integration is also performed in the Fourier domain and therefore also involves a FFT.

The rendering of the generated terrain models was done with the standard Explorer modules that are available for this purpose. See Figure 3 for a schematic view of the synthetic terrain generation sequence. Two basic Explorer applications (maps) were developed around these modules. The first application shows the increase in detail in the terrain as the user zooms in (§3.2), while the second application demonstrates the capability to generate new terrain "on-the-fly" as one moves around the landscape (see §3.3).

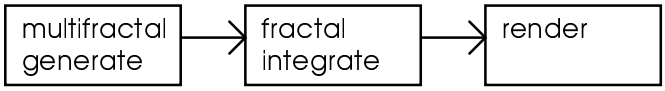


Figure 3: *Diagram of the synthetic terrain generation operation*

The raw multifractal field in Colour plate 1 was fractal integrated and the resulting terrain map is shown in Colour plate 2. The total time to generate and integrate a 128×128 resolution terrain map, is typically 1.5 seconds on an SGI Indigo workstation equipped with a MIPS R4000 CPU.

3.2 Zooming in on synthetic terrain

The zoom operation operates on the raw multifractal field. Therefore, the outcome of the zoom operation must be fractal integrated to obtain again a realistic terrain surface. Figure 4 shows the context diagram of the zoom procedure. The zoom operation was adapted from [9].

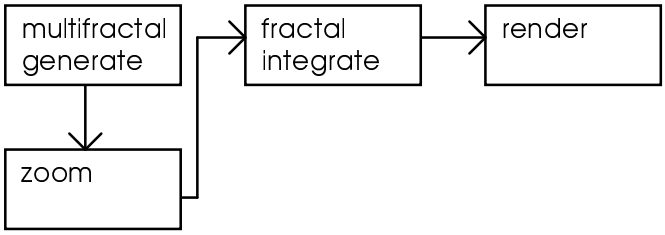


Figure 4: *Context diagram of the zoom operation process*

The user defines a point on a synthetic terrain surface *S*, around which a zoom window is constructed. The size of the zoom window depends on a zoomfactor *zf*, specified by the user, and the resolution r_S of *S*. Zooming is done in two stages: blowing up the original data in the zoom window and, secondly, generating new detail. Blowing up the data in the zoom window is straightforward: all data points in the zoom window are magnified in resolution by a factor *zf* to form squares of size zf^2, and their values are set to the corresponding original points in the zoom window. Detail is generated the same way as was done in the Multifractal module.

The only difference is hidden in the $f(\lambda,k)$ filter. Finer detail is characterised by higher frequencies in the Fourier domain, thus if the filter is shifted towards higher frequencies, finer detail will be the result.

Colour plate 3 and 4 show two examples of the zoom operation. Plate 3 is the result of zooming in twice on the lower right quadrant of the terrain surface, depicted in Colour plate 2. The global shape is preserved but more detail is shown. Colour plate 4 is an example of zooming in 4 times on the lower right quadrant of Colour plate 2. Again, the global structure remains. The three dominant peaks are visible and can be found in the original picture, Colour plate 2 and also in Colour plate 3.

3.3 Dynamic terrain generation

The Integrator module does not need to integrate the complete raw multifractal field. If we integrate only a part of the multifractal field, we can look at the integrator as a sliding window over the raw multifractal field. If a user has some sort of control to steer the integration window, the rendered fractal integrated part of the multifractal field will give the appearence of movement. We implemented this feature into IRIS Explorer module. Figure 5 shows the Explorer Render module. The Generator module was used to generate a raw multifractal field of a certain size. However, instead of integrating the whole field, only a smaller sub-area is integrated. In this case we chose an integration window one fourth the size of the underlying field. Thus, the Render screen shows only one fourth of the underlying raw multifractal field.

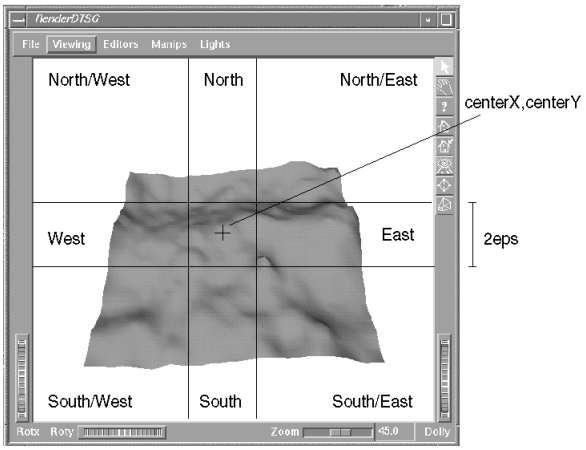


Figure 5: *Eight possible moving directions*

The user can steer the integration window across the underlying raw multifractal field by manipulating the mouse. By clicking the mouse button in one of the screen areas indicated in the figure, the center of the integration window is moved to that location and the integration operation is subsequently performed on the area

centered around that point. Because now only the integration has to be performed instead of both the generation and the integration, the time required to visualise the synthetic terrain is greatly reduced and the user can "pan" around the multifractal landscape interactively.

3.4 Real world terrain analysis

In addition to the terrain generation modules, we also developed a module for the analysis of multifractal parameters. This module takes 2D images or Digital Elevation Maps as input and provides estimates for the values of the multifractal parameters α, C_1 and H. The analysis method is based on the Structure function and it provides rough estimates for these parameters [11]. The method involves counting occurrences of height-differences in the terrain at a number of increasing scale levels (i.e., decreasing resolution). Colour plate 6 shows a sample from a Mars Digital Elevation Model (DEM) with resolution 64×64. Its actual size is 150x300 km. The difference between highest and lowest point on the DEM measures 4,5 km. We estimated the H parameter from the structure function, and calculated the moment scaling function $K(q)$. From $K(q)$, α and C_1 were estimated. Figure 6 shows a plot of its moment scaling function $K(q)$. We found the following estimates: H = 0.8; C_1 = 0.1; α = 1.7.

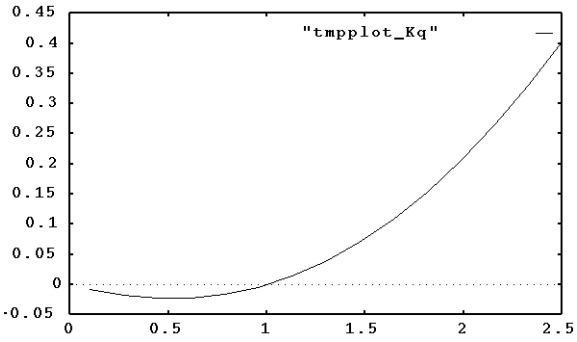


Figure 6: *A plot of the Moment scaling function K(q) of the sample Mars DEM depicted in Colour plate 6*

These same values were then used to generate Colour plate 7. Colour plate 6 can be characterised as a rather calm terrain surface without sudden changes and these features are again found in Colour plate 7.

4 Towards a parallel approach

Although the preliminary performance result of 1.5 s to generate a 128×128 digital elevation model is encouraging, a considerable performance increase is required in

order to achieve real-time terrain generation speeds. Fortunately, the structure of the algorithms and the architecture of our generator suggest several possibilities for performance gain.

4.1 Distribution of the generation of raw terrain patches

Instead of having a serial connection between a single Multifractal Generator that generates the raw multifractal field and the Integrator, a feasible set-up would be one with multiple generation processes, each running on their own processor, feeding the Integrator that in turn generates DEM data that are fed into the rendering stage. The Integrator does not need to filter the entire multifractal field in one piece. Instead, only the data in a local window (with respect to the current viewpoint and -direction in the terrain) are integrated and rendered. See Figure 7. By employing predictive algorithms to determine where to generate a new patch depending on the movement of the viewpoint, new terrain patches can be generated before they become visible (i.e. have to be integrated and rendered).

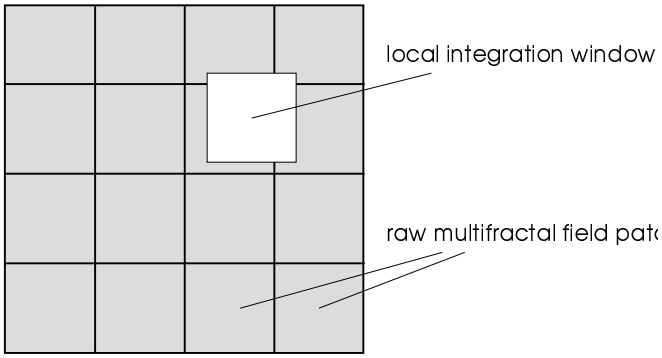


Figure 7: *Different processors can be used to generate multifractal patches that are partially integrated, depending on location and size of the local integration window.*

4.2 Parallellisation of the multifractal generation and integration algorithms

The multifractal generation and the integration (smoothing) algorithms themselves are also candidates for parallellisation. For instance, in both the generation and the integration steps (inverse) Fast Fourier Transforms are used, for which a parallel implementation of the FFT algorithm might be considered.

4.3 Further research

In addition to the previously mentioned issues, two more subjects for futher work can be mentioned. One deals with better methods for parameter estimation for

terrain analysis, while the second is related to the generation of terrain features with a directional preference.

The analysis method that we used for real world terrain analysis, provides rather poor estimates. It is better to use for example the Double Trace Moments method [13] for more accurate parameter estimation. The Structure function could still be used for initial estimates.

So far we have used the multifractal formalism to generate terrain models that show height features that have no directional preference. We have experimented somewhat with several parameters in the generation and integration processes that influence the directional distribution of height values, but this aspect requires still more study. One of these parameters we have called the *skewness* factor. Colour plate 5 shows an example of this. Colour plate 5 was made in exactly the same way as Colour plate 2 was, but in the former, the skewness factor was changed so that the terrain in plate 5 exhibits more structures oriented from left to right across the image, while in plate 2 they show a directional preference from top to bottom.

Acknowledgements

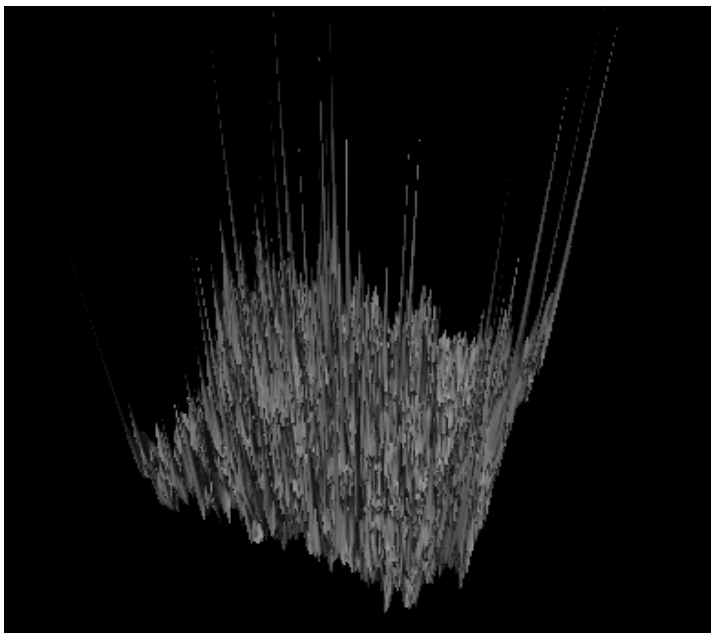
The work described here was performed under ESA/ESTEC contract 10475/93/NL/JG(SC), and also sponsored by the Netherlands Agency for Aerospace Programs (NIVR) under contract NRT 2305 FE. Thanks are due to Felix Herrmann for fruitful discussions about the multifractal formalism.

References

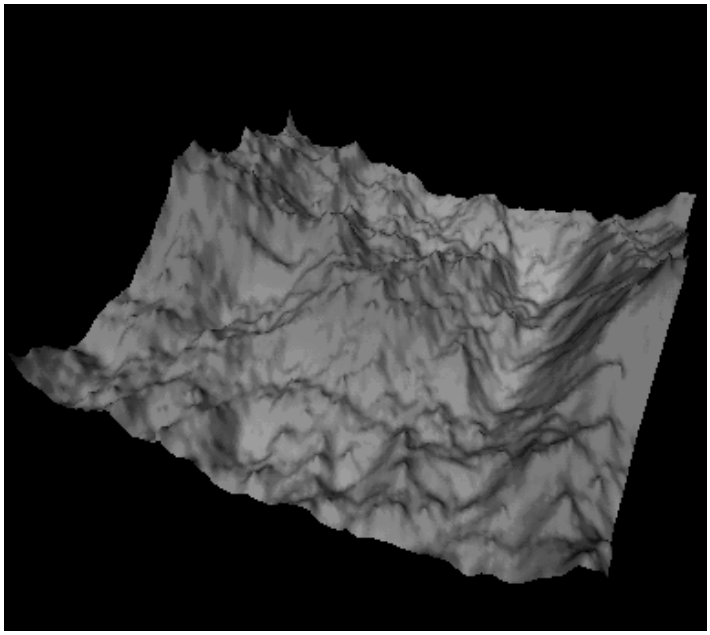
1. B.B. Mandelbrot, The Fractal geometry of Nature, Freeman and Company, 1977
2. H.-O. Peitgen and D. Saupe; The Science of Fractal Images; Springer-Verlag, New York, 1988.
3. M. F. Barnsley; Fractals Everywhere; Academic Press, 1988.
4. C.A. Pickover, Generating Extraterrestrial Terrain, IEEE Computer Graphics and Applications, Vol. 15, No. 2, pp 18-21, March 1995.
5. J. Rohlf and J. Helman, IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics, Proc. Siggraph, 1994.
6. A. Malinverno, A Simple Method to Estimate the Fractal Dimension of a Self-Affine Series, Geophysical Research Letters, Vol. 17, No. 11, pp. 1953-1956, October 1990.
7. Y. Fisher, Fractal Image Compression, SIGGRAPH '92 Course Notes.
8. R.J. Voss; Random Fractals: Self-affinity in Noise, Music, Mountains, and Clouds; in: Proceedings of the International Conference honouring Benoit B. Mandelbrot on his 65th birthday; Vence, France, October 1989 / editors A. Aharony, J. Feder.

9. D. Schertzer and S. Lovejoy (eds.), Non-linear variability in geophysics: scaling and fractals. Kluwer Academic, 1991.
10. S. Pecknold, S. Lovejoy, D. Schertzer, C. Hooge, and J. F. Malouin, The simulation of universal multifractals, Cellular Automata, World Scientific, 1993.
11. J. F. Muzy, E. Bacrye, and A. Arneodo, Multifractal formalism for fractal signals: The structure-function approach versus the wavelet-transform-modulus-maxima method, Journal of Statistical Physics, vol. 17, no. 34, pp. 635-674.
12. IRIS Explorer User's Guide, Silicon Graphics, Inc., Doc. no. 007-1371-020.
13. D. Schertzer, S. Lovejoy (editors); Non-linear variability in Geophysics 3: scaling and multifractal processes; Lecture Notes EGS Richardson Memorial Conference; september 10-17; 1993

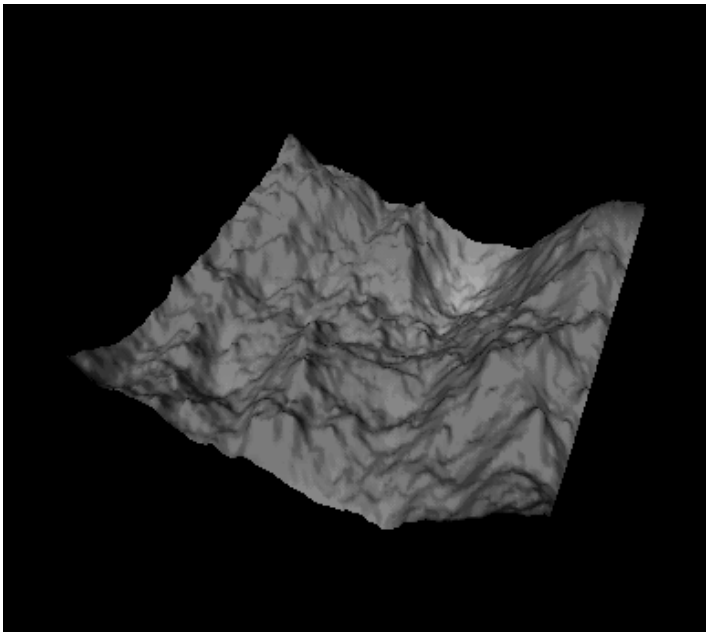
Colour plates



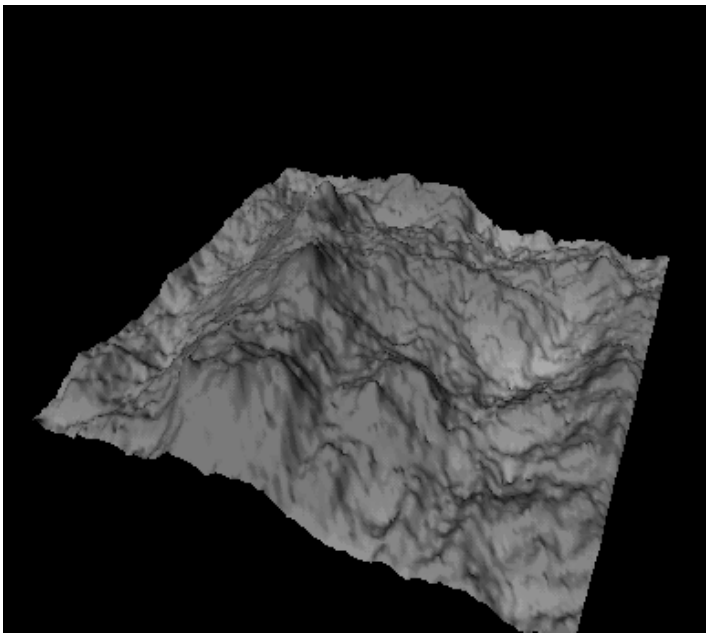
Colour plate 1. Raw multifractal field



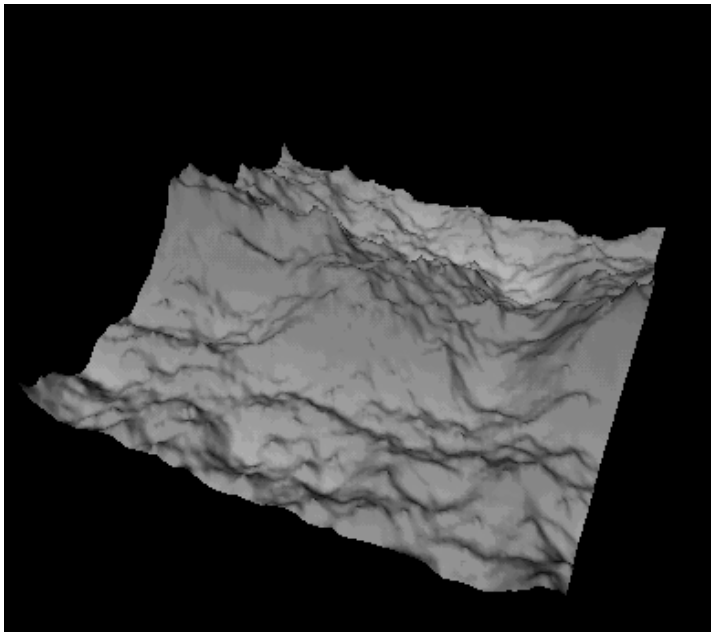
Colour plate 2: Fractal integrated multifractal field



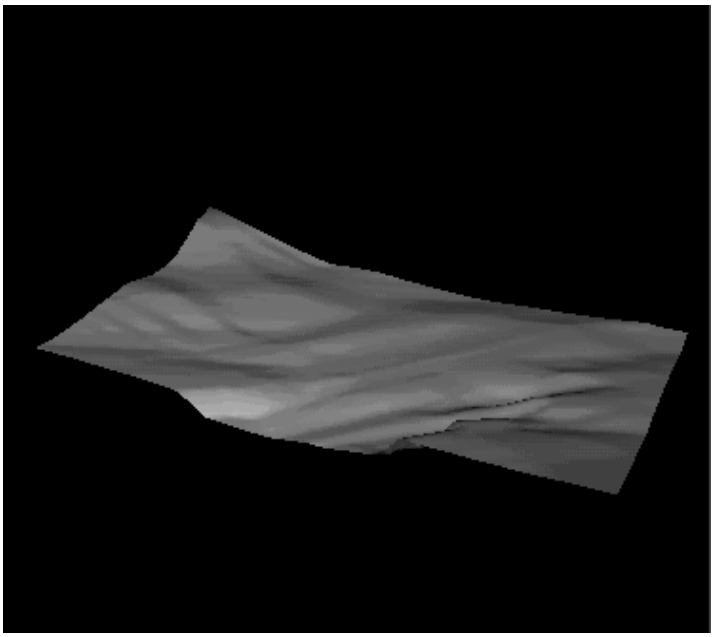
Colour plate 3: Example of the zoom operation with zoomfactor 2



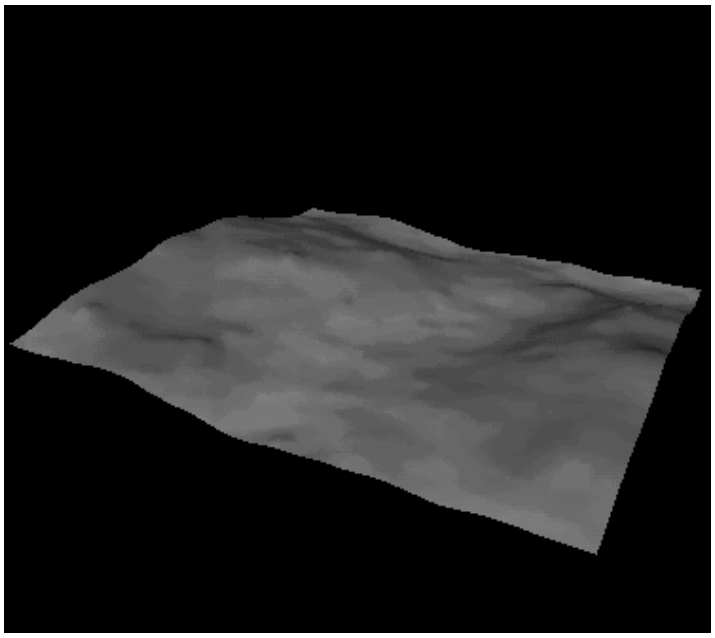
Colour plate 4: Example of the zoom operation with zoomfactor 4



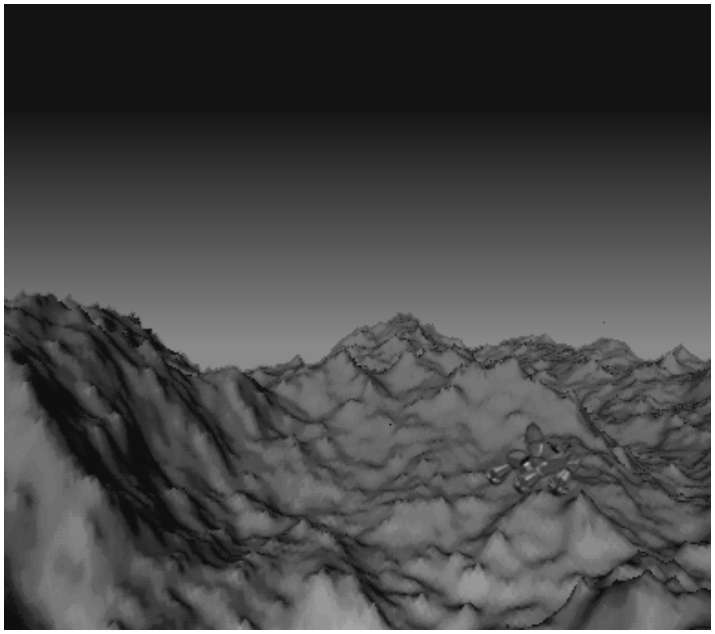
Colour plate 5: Skewed terrain surface



Colour plate 6: Analysed Mars terrain patch: $H = 0.8$; $C_1 = 0.1$; $\alpha = 1.7$.



Colour plate 7: Generated terrain with $H = 0.8$; $C_1 = 0.1$; $\alpha = 1.7$.



Colour plate 8: Synthetic Martian Terrain Surface