V-variable Fractals: Theory and Implementation

Bram Kuijvenhoven

June 13, 2007

Abstract

This report gives an overview of V-variable fractal theory, as presented in [2003a], and briefly discusses computer implementations suitable for rendering such fractals in three dimensionsal space.

1 Introduction

Random and deterministic fractals generated by Iterated Function Systems are used to model phenomena in a wide range of applications. In many applications, a finer control on local variability is required than what can be accomplished by using deterministic fractals, which are locally too similar at different points, and random fractals, which by contrast exhibit too few correlation between different points [BHS]. Moreover, computing random fractals is a slow and difficult task.

Therefore, the class of *V-variable fractals* is introduced, which accomodate mainly for two things:

- Fine control over local variability.
- A fast algorithm for computing and sampling standard random fractals.

Barnsley, Hutchinson and Stenflo published an expository article about V-variable fractals in [BHS], while [2003a] and [2003b] provide the more technical backgrounds, including V-variable code trees and dimension results.

One of the applications given in [2003a] is the area of computer graphics. Examples are given of fractals in $\mathbb{R}^2$, discretized using *bitmaps* (two-dimensional arrays of pixels). Section 2 through 5 give an overview of the theory on V-variable fractals as presented in [2003a]. Some additional remarks are placed in footnotes. Finally, in Section 6 is presented my own work regarding computer implementations – in particular those suitable for rendering three-dimensional fractals.

2 IFSs and fractals

A traditional way to generate deterministic and random fractals is by using *Iterated Function Systems* (IFSs). These IFSs express the self-similarity of the fractals and can be used to actually construct (samples of) the fractals.

2.1 IFSs

First, we will introduce some notation and the basics of IFSs. Throughout, M, N and V will denote positive integers, m will be a number in the set $\{1, \ldots, M\}$, and similarly $n \in \{1, \ldots, N\}$, and $v \in \{1, \ldots, V\}$.

Consider a compact, metric space $(\mathbb{X}, d_{\mathbb{X}})$. An *Iterated Function System* (*IFS*) on $\mathbb{X}$ is denoted by

$$F = (\mathbb{X}; f_1, \ldots, f_M; p_1, \ldots, p_M), \tag{1}$$

with the $f_m : \mathbb{X} \to \mathbb{X}$ being M component maps and the p_m being M non-negative weights adding up exactly to 1. Sometimes the weights are omitted and then the version with weights is referred to as an *IFS with weights* or a *weighted IFS*.

The maps f_m will usually be contractive maps. Examples of the space $\mathbb{X}$ are $\mathbb{R}^2$ and $\mathbb{R}^3$ equipped with the Euclidean metric. In this case, the maps f_m will usually be affine transformations, which can be described by a single matrix and vector.

The IFS F does not operate on $\mathbb{X}$ itself, but on a derived space. Of particular interest are the space $\mathbb{H} = \mathbb{H}(\mathbb{X})$ of all non-empty, compact subsets of $\mathbb{X}$ and the space $\mathbb{P}(\mathbb{X})$ of all probability measures on $\mathbb{X}$. For the associated metrics $d_{\mathbb{H}}$ and $d_{\mathbb{P}}$ we take the Hausdorff metric and the Monge-Kantorovitch metric that are induced by $d_{\mathbb{X}}$. (See [2003a] for a precise definition of the these metrics.) This yields two compact metric spaces $(\mathbb{H}(\mathbb{X}), d_{\mathbb{H}})$ and $(\mathbb{P}(\mathbb{X}), d_{\mathbb{P}})$. The Borel sets of $\mathbb{X}$ are denoted by $\mathbb{B}(\mathbb{X})$.

A map $f : \mathbb{X} \to \mathbb{X}$ is extended to $f : \mathbb{H}(\mathbb{X}) \to \mathbb{H}(\mathbb{X})$ by setting $f(K) = \{f(x) : x \in K\}$ for all $K \in \mathbb{H}(\mathbb{X})$. It is extended to $f : \mathbb{P}(\mathbb{X}) \to \mathbb{P}(\mathbb{X})$ by taking the push-forward map defined by $f(\mu) = \mu \circ f^{-1}$ for all $\mu \in \mathbb{P}(\mathbb{X})$.

The IFS F itself can now operate on $\mathbb{H}(\mathbb{X})$ and $\mathbb{P}(\mathbb{X})$ as follows:

$$F(K) = \bigcup_{m=1}^{M} f_m(K) \qquad \forall K \in \mathbb{H}(\mathbb{X}), \tag{2}$$

$$F(\mu) = \sum_{m=1}^{M} p_m f_m(\mu) \qquad \forall \mu \in \mathbb{P}(\mathbb{X}). \tag{3}$$

If the components maps f_m are contractions — i.e. there exists a constant $l \in [0, 1)$ such that $d(f_m(x), f_m(y)) \leq l \cdot d(x, y)$ for all $x, y \in \mathbb{X}$ — then so is F, on both spaces $\mathbb{H}(\mathbb{X})$ and $\mathbb{P}(\mathbb{X})$ and using the same constant. As a result, there exists a unique attractor set $A \in \mathbb{H}(\mathbb{X})$ satisfying $A = F(A)$ and a unique attractor measure $\mu \in \mathbb{P}(\mathbb{X})$ satisfying $\mu = F(\mu)$.

Hence there are two *IFS attractors*: the set attractor A, called a *fractal set*, and the measure attractor μ, called a *fractal measure*. Both attractors are called *fractals*. The component maps f_m define the self-similarity of the fractals; they define how the fractals are composed of (transformed) copies of itself.

From now on we will assume that the component maps of the IFSs that we will discuss are all strict contractions with the same factor $l \in [0, 1)$.

2.2 Fractal computation

There are basically two algorithms for the computation of IFS-generated fractals: the *deterministic process*, or *backward process*, and the *random iteration process*, or *forward process*, which is also called the *chaotic process* or *chaos game*.

The backward process follows a standard way to obtain the fixed point of a contractive map: start with any set $A_0 \in \mathbb{H}$ and iterate $A_{k+1} = F(A_k)$ for $k = 1, 2, \ldots$; the process will converge (in the Hausdorff metric) to the attractor set A. Similarly, one can start with a $\mu \in \mathbb{P}$ and iterate $\mu_{k+1} = F(\mu_k)$ for $k = 1, 2, \ldots$ and the process will converge (in the Monge-Kantorovitch metric) to the attractor measure μ.

The forward process does not start with a set, but with a point, $x_1 \in \mathbb{X}$. The iteration step is given by $x_{k+1} = \hat{f}_k(x_k)$, $k = 2, 3, \ldots$, where in each step $\hat{f}_k$ is chosen randomly from $(f_1, \ldots, f_M)$ using the respective weights $(p_1, \ldots, p_M)$. The result is a random orbit $\{x_k\}_{k \geq 1}$ of points which do accumulate unevenly in $\mathbb{X}$. In particular, it has been proved that for all starting points x_1 almost surely the weighted sums of point measures $\frac{1}{k}(\delta_{x_1} + \cdots + \delta_{x_k})$ converges in the weak sense to the attractor measure μ.

2.3 Code spaces

A very useful concept associated with the IFS F is that of its *code space*, $\Sigma = \{1, \ldots, M\}^\infty$. The members of this space are infinite sequences over the alphabet $\{1, \ldots, M\}$, indexed by the positive numbers, $\mathbb{N}$.

The space Σ is made a compact metric space by equipping it with the metric d_Σ, which is defined for $\sigma \neq \tau$ by

$$d_\Sigma(\sigma, \tau) = \frac{1}{M^k} \tag{4}$$

where k is the smallest index such that $\sigma_k \neq \tau_k$.

There exists a continuous onto mapping $F : \Sigma \to A$, which is defined for all $\sigma_1\sigma_2 \cdots \in \Sigma$ by

$$F(\sigma_1\sigma_2 \ldots) = \lim_{k \to \infty} f_{\sigma_1} \circ \ldots \circ f_{\sigma_k}(x), \tag{5}$$

a limit that is independent of $x \in \mathbb{X}$ and whose convergence is uniform in x. The sequence $\sigma_1\sigma_2 \cdots \in \Sigma$, the code space, is called an *address* of the point $F(\sigma_1\sigma_2 \ldots) \in A$, the attractor set of F. Note that the map $F : \Sigma \to A$ is onto, but not one-to-one in general.

The map $F : \Sigma \to A$ also characterizes μ, the measure attractor of F, but to this end we must first define a fundamental IFS on Σ.

For $m \in \{1, \ldots, M\}$, the shift operator $s_m : \Sigma \to \Sigma$ is defined by

$$s_m(\sigma_1\sigma_2 \ldots) = m\sigma_1\sigma_2 \ldots \tag{6}$$

for all $\sigma_1\sigma_2 \cdots \in \Sigma$. The shift operators are contractive with factor $\frac{1}{M}$, hence the IFS $S := \{\Sigma; s_1, \ldots, s_M; p_1, \ldots, p_m\}$ has a unique set attractor, which is Σ

itself, and a unique measure attractor, which equals a measure that is denoted by $\pi \in \mathbb{P}(\Sigma)$. On *cylinder sets*, subsets of Σ of the form

$$[\sigma_1 \dots \sigma_k] = [\sigma_1 \dots \sigma_k]_\Sigma := \{\tau_1 \tau_2 \dots \in \Sigma : \sigma_1 \dots \sigma_k = \tau_1 \dots \tau_k\}, \tag{7}$$

with $k \geq 1$ and $\sigma_1 \dots \sigma_k \in \{1, \dots, M\}^k$, this measure π is defined by

$$\pi([\sigma_1 \dots \sigma_k]) = p_{\sigma_1} \cdots p_{\sigma_k}, \tag{8}$$

and this has a unique extension to $\mathbb{P}(\Sigma)$.

A theorem now relates the measure attractor μ of the IFS F on $\mathbb{X}$ to the measure attractor π of the fundamental IFS S on Σ by asserting that

$$\mu = F(\pi) = \pi \circ F^{-1}, \tag{9}$$

where $F : \Sigma \to A$ is the map defined in (5).

3 V-variable fractals

The construction of V-variable fractals corresponds to the forward process of a *superfractal*, which we will be discussed in greater detail below. A V-variable fractal is constructed using the set

$$\mathcal{F} := \left\{\mathbb{X}; F^1, \dots, F^N; P_1, \dots, P_N\right\} \tag{10}$$

whose N components F^n are IFSs in their own right, accompanied by non-negative weights P_n that add up to 1. For each IFS we write

$$F^n := \left\{\mathbb{X}; f_1^n, \dots, f_M^n; p_1^n, \dots, p_M^n\right\}. \tag{11}$$

The maps f_m^n are all maps operating on $\mathbb{X}$ with contraction factor smaller than or equal to $l \in [0, 1)$, and the non-negative weights p_m^n sum up to 1 over the index m. Finally, V is a positive integer which determines the 'variability' of the fractal that will be constructed.

Consider the space of V-tuples of non-empty compact subsets of $\mathbb{X}$, denoted by $\mathbb{H}^V = \mathbb{H}(\mathbb{X})^V$, as well as the space of V-tuples of measures on $\mathbb{X}$, denoted by $\mathbb{P}^V = \mathbb{P}(\mathbb{X})^V$. These V-tuple spaces are equipped with metrics $d_{\mathbb{H}^V}$ and $d_{\mathbb{P}^V}$, which are induced from $d_{\mathbb{H}}$ and $d_{\mathbb{P}}$ by taking the component-wise maximum. That is, for a metric space $(\mathbb{X}, d_{\mathbb{X}})$ one defines the metric space $(\mathbb{X}^V, d_{\mathbb{X}^V})$ by

$$d_{\mathbb{X}^V}(x, y) = \max_{v=1\dots V} d_{\mathbb{X}}(x_v, y_v) \tag{12}$$

for all $x = (x_1, \dots, x_V), y = (y_1, \dots, y_V) \in \mathbb{X}^V$, and the same contruction is used to obtain the metric spaces $(\mathbb{H}^V, d_{\mathbb{H}^V})$ and $(\mathbb{P}^V, d_{\mathbb{P}^V})$ from $(\mathbb{H}, d_{\mathbb{H}})$ and $(\mathbb{P}, d_{\mathbb{P}})$. These metrics allow one to speak of convergence in these V-tuple spaces, as well as of contraction maps on these spaces.

An iteration step in the construction of a V-variable fractal takes V 'input buffers' and produces V 'output buffers'. The iteration step also involves a

number of random choices: for each v-th output buffer an IFS F^{n_v} is chosen as well as M input buffers for this IFS, indexed by $(v_{v,1}, \ldots, v_{v,M})$. These choices together are captured in a single index $a \in \mathcal{A}$ by writing

$$a := (a_1, \ldots, a_V) \qquad \in \mathcal{A} := \left\{ \{1, \ldots, N\} \times \{1, \ldots, V\}^M \right\}^V, \tag{13}$$

$$a_v := (n_v; v_{v,1}, \ldots, v_{v,M}) \in \qquad \{1, \ldots, N\} \times \{1, \ldots, V\}^M. \tag{14}$$

In general, each choice $a \in \mathcal{A}$ is chosen with probability $\mathcal{P}^a$. These probabilities are determined for example by requiring that the choices of IFSs are independent of each other, with weights $(P_1, \ldots, P_N)$, and that the choices of input buffers are also independent but follow a uniform distribution. In that case they are given by

$$\mathcal{P}^a = \frac{P_{n_1} \cdots P_{n_V}}{V^{MV}}. \tag{15}$$

For each choice $a \in \mathcal{A}$, the iterations steps on $\mathbb{H}(\mathbb{X})^V$ and $\mathbb{P}(\mathbb{X})^V$ are given by maps $f^a : \mathbb{H}(\mathbb{X})^V \to \mathbb{H}(\mathbb{X})^V$ and $f^a : \mathbb{P}(\mathbb{X})^V \to \mathbb{P}(\mathbb{X})^V$, defined by

$$f^a(K) = \left(\bigcup_{m=1}^{M} f_m^{n_v}(K_{v_{v,m}}) \right)_{v=1}^{V}, \tag{16}$$

$$f^a(\mu) = \left(\sum_{m=1}^{M} p_m^{n_v} f_m^{n_v}(\mu_{v_{v,m}}) \right)_{v=1}^{V} \tag{17}$$

for all $K = (K_1, \ldots, K_V) \in \mathbb{H}(\mathbb{X})^V$ and $\mu = (\mu_1, \ldots, \mu_V) \in \mathbb{P}(\mathbb{X})^V$. Essentially, the construction of V-variable fractals corresponds to the chaotic or forward process of the *superIFSs*

$$\mathcal{F}_V := \left\{ \mathbb{H}(\mathbb{X})^V; f^a, \mathcal{P}^a, a \in \mathcal{A} \right\}, \tag{18}$$

$$\tilde{\mathcal{F}}_V := \left\{ \mathbb{P}(\mathbb{X})^V; f^a, \mathcal{P}^a, a \in \mathcal{A} \right\}. \tag{19}$$

These IFSs also have each a unique attractor, called the *superfractal set*, respectively *superfractal measure*. This follows from the fact that the f^a are contractions with the factor $l \in [0, 1)$ in $(\mathbb{H}(\mathbb{X})^V, d_{\mathbb{H}(\mathbb{X})^V})$ and $(\mathbb{P}(\mathbb{X})^V, d_{\mathbb{P}(\mathbb{X})^V})$. (For a proof of this, see Theorems 15 and 20 of [2003a].)

4 Random fractals

Associated to the weighted collection $\mathcal{F}$ of IFSs in (10) is also a canonical random fractal. The construction of this random fractal can be understood in the context of associated code trees and a probability distribution on the space of such code trees, which is induced naturally by the weights $(P_1, \ldots, P_N)$.

4.1 Code trees

Let T denote the M-fold *tree*: the set of finite sequences of elements from the set $\{1, \ldots, M\}$ including the empty sequence, which is denoted by $\emptyset$. The elements $i = i_1 \ldots i_k \in T$ of this tree are called its *nodes*; the number $|i| = k$ is called the *level* of the node. The M-fold *level-k tree* T_k is a subset of T containing only those nodes up to and including level k.

A *labelled tree* is a function with domain T. A *code tree* is a labelled tree with range $\{1, \ldots, N\}$. The space of all infinite code trees is denoted by

$$\Omega := \{\tau : \tau : T \to \{1, \ldots, N\}\}. \tag{20}$$

Obviously, a *level-k code tree* τ has domain T_k, and we write $|\tau| = k$ in such cases.

A level-k code tree τ is an *initial segment* of another code tree σ of at least level k (including infinity) if they agree on their common domain, T_k. This relation is denoted by $\tau \prec \sigma$. The *cylinder set* of a level k code tree τ is the set set of all code trees extending it:

$$[\tau] = [\tau]_\Omega := \{\sigma \in \Omega : \tau \prec \sigma\}. \tag{21}$$

That is, a cylinder set is the set of infinite code trees whose first k levels correspond to the level-k code tree τ.

A metric d_Ω on Ω is defined by taking for $\tau \neq \sigma$ in Ω

$$d_\Omega(\tau, \sigma) := \frac{1}{M^k} \tag{22}$$

where k is the smallest level k for which there exist a level-k node $i \in T$ such that $\sigma(i) \neq \tau(i)$. This metric makes (Ω, d_Ω) a compact metric space. The cylinder sets form exactly the open as well as the closed balls in this space.

The probabilities $(P_1, \ldots, P_N)$ induce a natural probability distribution ρ on Ω. For cylinder sets $[\tau]$ where τ is a level-k code tree ($k \geq 0$), it is defined by[1]

$$\rho([\tau]) := \prod_{i \in T_k} P_{\tau(i)} \tag{23}$$

and $\rho(\Omega) = 1$. The cylinder sets generate the Borel σ-algebra of (Ω, d_Ω), denoted by $\mathbb{B}(\Omega)$, hence the extension of ρ to a probability distribution on Ω can be done in the usual way.

4.2 The random fractal

The labels on the code trees correspond to the choices of IFSs in the random fractal that they describe, and the probability distribution ρ on the space Ω consisting of such code trees forms the underlying probability distribution. The

[1] Equation (3.3) of [2003a] states that product is to be taken over nodes i with $1 \leq |i| \leq |\tau|$, but in fact the root node $\emptyset$ should also be included, which is denoted here by $i \in T_k$.

tree node hierarchy corresponds to the way the components maps of the chosen IFSs are applied to an initial set $K \in \mathbb{H}(\mathbb{X})$ or initial measure $\mu \in \mathbb{P}(\mathbb{X})$.

Define for all $k \geq 1$ and $\sigma \in \Omega$ the compositions of maps

$$\mathcal{F}_k(\sigma)(K) = \bigcup_{\{i \in T : |i| = k\}} f_{i_1}^{\sigma(\emptyset)} \circ f_{i_2}^{\sigma(i_1)} \circ \ldots \circ f_{i_k}^{\sigma(i_1 i_2 \ldots i_{k-1})}(K), \tag{24}$$

$$\mathcal{F}_k(\sigma)(\mu) = \sum_{\{i \in T : |i| = k\}} \left(\prod_{j=1}^{k} p_{i_j}^{\sigma(i_1 \ldots i_{j-1})} \right) \cdot f_{i_1}^{\sigma(\emptyset)} \circ f_{i_2}^{\sigma(i_1)} \circ \ldots \circ f_{i_k}^{\sigma(i_1 \ldots i_{k-1})}(\mu). \tag{25}$$

From the strict contractivity of the maps f_m^n it follows that indepent of K and μ, the limits

$$\mathcal{F}(\sigma) := \lim_{k \to \infty} \mathcal{F}_k(\sigma)(K), \qquad \tilde{\mathcal{F}}(\sigma) := \lim_{k \to \infty} \tilde{\mathcal{F}}_k(\sigma)(\mu) \tag{26}$$

exist, and the convergence is uniform in σ, K and μ. Moreover, the functions

$$\mathcal{F} : \Omega \to \mathbb{H}(\mathbb{X}), \qquad \tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{X}) \tag{27}$$

are continuous.

The sets of fractal sets, respectively fractal measures, associated with $\mathcal{F}$ are given by

$$\mathfrak{H} := \{\mathcal{F}(\sigma) : \sigma \in \Omega\} = \mathcal{F}(\Omega), \qquad \tilde{\mathfrak{H}} := \{\tilde{\mathcal{F}}(\sigma) : \sigma \in \Omega\} = \tilde{\mathcal{F}}(\Omega), \tag{28}$$

and these sets are distributed according to the probability distributions

$$\mathfrak{B} := \mathcal{F}(\rho) = \rho \circ \mathcal{F}^{-1} \in \mathbb{P}(\mathfrak{H}), \qquad \tilde{\mathfrak{B}} := \tilde{\mathcal{F}}(\rho) = \rho \circ \tilde{\mathcal{F}}^{-1} \in \mathbb{P}(\tilde{\mathfrak{H}}). \tag{29}$$

5 V-variable fractals revisited

In the previous section we discussed a canonical random fractal associated with the weighted system of IFSs $\mathcal{F}$. We introduced a collection of code trees, Ω, generating a set of fractals, $\mathfrak{H}$, distributed according to a probabililty distribution, $\mathfrak{B}$, that was derived from a distribution on the code trees, ρ. We will now construct a collection of code trees suitable for describing V-variable fractals, and show how these fractals can be used to approximate and sample the canonical random fractal $(\mathfrak{H}, \mathfrak{B})$.

5.1 Code trees revisited

Before diving into the appropriate code trees, we introduce some extensions to our concept of trees. Recall the M-fold tree T. To each node $im \in T$, with $i \in T$ and $m \in \{1, \ldots, M\}$, corresponds a *limb*, (i, im); the limb corresponding to the root node $\emptyset$ is $(\emptyset, \emptyset)$, which we call the *trunk*. Limbs can also be labelled, or even both the nodes and the limbs.

Another concept, regarding code trees, is that of subtrees: a *subtree* $\tilde{\tau}$ of a given code tree τ, rooted at node $i \in T$ is defined by $\tilde{\tau}(j) = \tau(ij)$ for all $j \in T$.

5.2 An IFS on Ω^V

Now consider the space Ω^V, the space of V-tuples of code trees. An element of Ω^V is called a *grove*[2]. For convenience, the trunks of its V component trees will always be labelled from 1 to V. The metric d_{Ω^V} on this space takes the component-wise maximum of the metric d_Ω defined in (22), cf. (12).

Not all groves in Ω^V are of interest for our purposes. An IFS on Ω^V will be contructed whose attractor, Ω_V, will provide the code trees of interest for V-variable fractals. The component maps of this IFS are given by certain functions $\eta^a : \Omega^V \to \Omega^V$, with $a \in \mathcal{A}$. $\mathcal{A}$ is the index set defined in (13), in our first discussion of V-variable fractals. The weights are again the probabilities $\mathcal{P}^a$, indexed by $a \in \mathcal{A}$.

The maps η^a correspond intuitively to V-tuples of *level-1 function trees*. The v-th level-1 function of η^a consists of level-1 tree whose trunk is labelled with v, whose root node is labelled n_v, and whose M limbs are labelled $v_{v,1}$ through $v_{v,M}$. The V-grove $\eta^a(\omega)$, obtained after one application of η^a to a V-grove $\omega = (\omega_1, \ldots, \omega_V) \in \Omega^V$, consists of these V level-1 function trees with to each m-th limb attached $\omega_{v_{v,m}}$, the $v_{v,m}$-th tree of the V-grove ω.

Algebraically, the maps η^a are defined conveniently using the n-th shift mapping $\xi_n : \Omega^M \to \Omega$, defined by

$$\xi_n(\omega)(\emptyset) := n, \qquad \xi_n(\omega)(mi) := \omega_m(i), \tag{30}$$

for all $\omega = (\omega_1, \ldots, \omega_M) \in \Omega^M$, $m \in \{1, \ldots, M\}$ and $i \in T$. The interpretation of the maps η^a in terms of level-1 function trees is now formalized by

$$\eta^a(\omega_1, \ldots, \omega_V) = \left(\xi_{n_1}(\omega_{v_{1,1}}, \ldots, \omega_{v_{1,M}}), \ldots, \xi_{n_V}(\omega_{v_{V,1}}, \ldots, \omega_{v_{V,M}})\right), \tag{31}$$

for all $\omega = (\omega_1, \ldots, \omega_V) \in \Omega^V$ and $a \in \mathcal{A}$.

The IFS on Ω^V giving the right code space is

$$\Phi := \left\{\Omega^V; \eta^a, \mathcal{P}^a, a \in \mathcal{A}\right\}. \tag{32}$$

The maps η^a are contractive with constant $\frac{1}{M}$, as Theorem 9 of [2003a] points out. This is a natural consequence of the usage of the shift maps ξ_n in (31) and the way the metric d_Ω is defined. As a result, the IFS Φ has a unique attractor set $\Omega_V \in \mathbb{H}(\Omega^V)$ and a unique attractor measure $\mu_V \in \mathbb{P}(\Omega^V)$.

5.3 V-groves, V-trees and V-variability

The elements of Ω_V are called *V-groves* and in turn their elements are called *V-trees*. The V-groves are a special kind of groves, and their elements, V-trees, are a special kind of code trees, hence the prefix 'V-' in front of their names. There speciality is V-variability, as will be seen shortly.

[2]The *New Oxford American Dictionary* defines a grove as "a small wood, orchard, or group of trees"; other dictionaries add that it is usually without underbrush.

First, we show that we can restrict our attention to the code trees appearing on the first coordinate of the V-tuples in Ω^V. Specifically, one defines $\Omega_{V,v}$ to be the set of V-trees appearing on the v-th coordinate of the V-groves in Ω_V: [3]

$$\Omega_{V,v} := \{\omega_v \in \Omega : \omega = (\omega_1, \ldots, \omega_V) \in \Omega_V\}. \tag{33}$$

Similarly, ρ_V[4] denotes the marginal probability measure on the first coordinate:

$$\rho_V(B) := \mu_V(B, \Omega, \ldots, \Omega) \tag{34}$$

for all Borel sets $B \in \mathbb{B}(\Omega)$.

Theorem 10 of [2003a] states that independently of v, one has $\Omega_{V,1} = \Omega_{V,v}$, hence this equals the entire set of V-trees. If the probabilities $\{\mathcal{P}^a, a \in \mathcal{A}\}$ obey equation (15), then independently of v, starting at any initial grove, the random distribution of trees $\omega \in \Omega$ that occur on the v-th coordinate of groves produced after n iterations of the random iteration process of the IFS Φ, almost always converges weakly to the marginal distribution ρ_V as $n \to \infty$. (Of course, the random distribution of the V-groves themselves converges almost always to μ_V, as the IFS Φ is contractive. But this statement applies to the *marginal* distribution of trees occuring on the first coordinate.)

[2003a] is not very clear about the origin of the requirement that the $\mathcal{P}^a$ needs to obey equation (15), and it appears to be a little bit hidden in the proof. The proof states that the IFS Φ is invariant under a map $\Xi : \Omega^V \to \Omega^V$ that is a coordinate permutation, and that this means that the permuted IFS $\{\Omega^V : \Xi\eta^a\Xi^{-1}, \mathcal{P}^a, a \in \mathcal{A}\}$ equals Φ. The result would be that the attractors are also invariant under Ξ, which means that $\Xi\Omega_V = \Omega_V$ and $\Xi\mu_V = \mu_V$. In fact only the set of component maps does not change under transformation by Ξ. The invariance of Ω_V holds because it is independent of the weights $\mathcal{P}^a$, but the invariance of μ_V only follows only under a symmetry condition on these weights, and equation (15) satisfies this symmetry.

A set L of vectors of labelled trees, as well as the vectors themselves, are said to be *V-variable*, or to have the property of *V-variability*, when for each component of a vector $\omega \in L$ and for each level $k \geq 1$, the number of distinct subtrees rooted at the nodes at level k is at most V.

Theorem 11 of [2003a] shows that Ω_V equals exactly the set of V-variable groves in Ω^V, and that $\Omega_{V,1}$, the set of V-trees, equals exactly the V-variable trees in Ω. This is proved by showing that the set of V-variable groves in Ω^V equals the set attractor of the IFS Φ, which is unique.

One of the main reasons to introduce V-variable fractals is because they can be used to approximate canonical random fractals. Theorem 12 of [2003a] founds the basis of this fact by giving two limits regarding V-trees and code trees. It states that

$$d_{\mathbb{H}(\Omega)}(\Omega_{V,1}, \Omega) \leq \frac{1}{V}, \qquad \text{hence} \qquad \lim_{V \to \infty} \Omega_{V,1} = \Omega, \tag{35}$$

[3] [2003a] states that $\Omega_{V,v} \subset \Omega_V$, but in fact $\Omega_{V,v} \subset \Omega$ while $\Omega_V \subset \Omega^V$.

[4] Surely, a more canonical name for ρ_V would be $\mu_{V,1}$, and we could just as easily define $\mu_{V,v}$ for all $v \in \{1, \ldots, V\}$.

and that when the probabilities $\mathcal{P}^a$ obey equation (15), then also

$$d_{\mathbb{P}(\Omega)}(\rho_V, \rho) \leq 1.4 \left(\frac{M}{V}\right)^{\frac{1}{4}}, \qquad \text{hence} \qquad \lim_{V\to\infty} \rho_V = \rho, \tag{36}$$

where the limits are in the respective metrics $d_{\mathbb{H}(\Omega)}$ and $d_{\mathbb{P}(\Omega)}$.[5]

5.4 The code space Σ_V

The IFS Φ has as its associated code space $\Sigma_V := \mathcal{A}^\infty$. As Section 2.3 points out, there exists a continuous onto mapping $\Phi : \Sigma_V \to \Omega_V$, and an infinite sequence $a_1 a_2 \cdots \in \Sigma_V$ is called an address of the point $\Phi(a_1 a_2 \dots) \in \Omega_V$. This mapping is not one-to-one, however.

It is possible to compose level-1 functions trees to form level-k function trees. A *level-k function tree* is level-k labelled tree with the nodes of the first $k-1$ levels labelled with numbers from $\{1, \dots, N\}$ and the limbs from all levels labelled with numbers from $\{1, \dots, V\}$. Let $k \in \mathbb{N}$ be a positive number. A grove g of V level-k function trees has the trunks of each v-th component labelled with v. We write $|g| = k$, and let G_k denote the set of such groves. Furthermore, let $G := \cup_{k\geq 1} G_k$.

The composition of two function tree groves g and h in G, denoted $g \circ h$, has height $|g \circ h| = |g| + |h|$. To each level-k limb of g that is labelled with v, the v-th function tree from h is attached. Similarly, for each $g \in G$, the map $\eta^g : \Omega^V \to \Omega^V$ is defined by attaching to each level-$|g|$ node of g that is labelled v, the v-th component of $\omega \in \Omega^V$. This extends the notion of maps η^a, $a \in \mathcal{A}$, and their associated function trees.

In particular, the operation $\circ$ on function trees is associative, and has the property that

$$\eta^{g\circ h} = \eta^g \circ \eta^h. \tag{37}$$

for all $g, h \in G$. In particular,

$$\eta^{a_1 \circ \dots \circ a_k} = \eta^{a_1} \circ \dots \circ \eta^{a_k}. \tag{38}$$

for all $a_1 \dots a_k \in \mathcal{A}^k$, where we used $a \in \mathcal{A}$ to denote the corresponding level-1 function trees.[6]

[5] The constant 1.4 in equation (36) may be wrong, because from the proof in [2003a] itself, it appears that the constant must be $\frac{7}{2^{1/4}\cdot 3} \approx 1.96$.

[6] [2003a] explains that they use η^a, $a \in \mathcal{A}$, to denote both the mapping $\eta^a : \Omega^V \to \Omega^V$ and the level-1 function tree to which it bijectively corresponds. Similarly they let η^g, $g \in G$, denote both the mapping $\eta^g : \Omega^V \to \Omega^V$ and the function tree g itself. This seems to be a little confusing, especially in interpreting equations like (37) and (38).

In fact, it seems to be more appropriate to let the indices $a \in \mathcal{A}$ denote both an element in $\mathcal{A}$ and the level-1 function tree to which it bijectively corresponds. Similary, $g \in G$ is a function tree, which can be written as the composition of finitely many $a \in \mathcal{A}$, and η^g is the map on Ω^V to which it corresponds.

5.5 Superfractal sets and measures

Recall the definition of the superIFSs $\mathcal{F}_V$ and $\tilde{\mathcal{F}}_V$ in equations (18) and (19). We denote their set attractors — called superfractal sets — by $\mathfrak{H}_V \in \mathbb{H}(\mathbb{H}(\mathbb{X})^V)$ and $\tilde{\mathfrak{H}}_V \in \mathbb{H}(\mathbb{P}(\mathbb{X})^V)$, and we denote their measure attractors — called superfractal measures — by $\mathfrak{B}_V \in \mathbb{P}(\mathbb{H}(\mathbb{X})^V)$ and $\tilde{\mathfrak{B}}_V \in \mathbb{P}(\mathbb{P}(\mathbb{X})^V)$. The superfractal sets are said to be distributed by their superfractal measures.

Just as we did with the V-tuples of code trees Ω_V, we will restrict our attention to the first coordinate, that is, define for all $v, \in \{1, \dots, V\}$ the collections

$$\mathfrak{H}_{V,v} := \{K_v \in \mathbb{H}(\mathbb{X}) : (K_1, \dots, K_V) \in \mathfrak{H}_V\}, \tag{39}$$

$$\tilde{\mathfrak{H}}_{V,v} := \{\ \mu_v \in \mathbb{P}(\mathbb{X}) :\ \ (\mu_1, \dots, \mu_V) \in \tilde{\mathfrak{H}}_V\}, \tag{40}$$

and define the marginal probability measures $\mathfrak{B}_{V,1} \in \mathbb{P}(\mathbb{H})$ and $\tilde{\mathfrak{B}}_{V,1} \in \mathbb{P}(\mathbb{P})$ by

$$\mathfrak{B}_{V,1}(B) := \mathfrak{B}_V(B, \mathbb{H}(\mathbb{X}), \dots, \mathbb{H}(\mathbb{X})) \quad \forall B \in \mathbb{B}(\mathbb{H}(\mathbb{X})), \tag{41}$$

$$\tilde{\mathfrak{B}}_{V,1}(B) := \tilde{\mathfrak{B}}_V(B, \mathbb{P}(\mathbb{X}), \dots, \mathbb{P}(\mathbb{X})) \quad \forall B \in \mathbb{B}(\mathbb{P}(\mathbb{X})). \tag{42}$$

In a similar way as can be shown for the sets of V-trees $\Omega_{V,v}$ and the marginal distribution ρ_V, one can prove that $\mathfrak{H}_{V,v} = \mathfrak{H}_{V,1}$ for all v and that $\tilde{\mathfrak{H}}_{V,v} = \tilde{\mathfrak{H}}_{V,1}$ for all v, and that when the probabilities $\mathcal{P}^a$ that are used in the superIFSs $\mathcal{F}_V$ and $\tilde{\mathcal{F}}_V$ obey equation (15), then the marginal distributions $\mathfrak{B}_{V,1}$ and $\tilde{\mathfrak{B}}_{V,1}$ can be approximated by looking at the empirical distribution at any v-th coordinate of the random iteration process. This can also be proved by lifting the result on code trees to the associated fractals, something that will be justified shortly.

As for any IFS, there exist a continuous onto mapping $\mathcal{F}_V : \Sigma_V \to \mathfrak{H}_V$ that assigns to each address $a_1 a_2 \cdots \in \Sigma_V$ a V-tuple of non-empty compact sets $\mathcal{F}_V(a_1 a_2 \dots) \in \mathbb{H}^V$. Similarly, there exists a continuous onto mapping $\tilde{\mathcal{F}}_V : \Sigma_V \to \tilde{\mathfrak{H}}_V$ that assigns to each address $a_1 a_2 \cdots \in \Sigma_V$ a V-tuple of measures $\tilde{\mathcal{F}}_V(a_1 a_2 \dots) \in \mathbb{P}^V$. But these mappings are not one-to-one in general, just as the mapping $\Phi : \Sigma_V \to \Omega_V$ is not one-to-one in general.

The space Ω_V, however, does provide a very useful code space for $\mathcal{F}_V$ and $\tilde{\mathcal{F}}_V$. The maps $\mathcal{F} : \Omega \to \mathbb{H}(\mathbb{X})$ and $\tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{X})$ from equation (27) are extended to the domain Ω^V by setting for all $(\omega_1, \dots, \omega_V) \in \Omega^V$

$$\mathcal{F}(\omega_1, \dots, \omega_V) := (\mathcal{F}(\omega_1), \dots, \mathcal{F}(\omega_V)), \tag{43}$$

$$\tilde{\mathcal{F}}(\omega_1, \dots, \omega_V) := (\tilde{\mathcal{F}}(\omega_1), \dots, \mathcal{F}(\omega_V)). \tag{44}$$

Then Theorems 17 and 22 of [2003a] state that the following statements hold:

$$\mathcal{F}(\eta^a(\omega)) = f^a(\mathcal{F}(\omega)), \qquad \tilde{\mathcal{F}}(\eta^a(\omega)) = f^a(\tilde{\mathcal{F}}(\omega)) \tag{45}$$

for all $\omega \in \Omega^V$ and $a \in \mathcal{A}$, and

$$\mathcal{F}(\Omega_V) = \mathfrak{H}_V, \qquad \tilde{\mathcal{F}}(\Omega_V) = \tilde{\mathfrak{H}}_V, \tag{46}$$

$$\mathcal{F}(\Omega_{V,1}) = \mathfrak{H}_{V,1}, \qquad \tilde{\mathcal{F}}(\Omega_{V,1}) = \tilde{\mathfrak{H}}_{V,1}, \tag{47}$$

and when the weights $\mathcal{P}^a$ obey equation (15), then also

$$\mathcal{F}(\mu_V) = \mathfrak{B}_V, \qquad \tilde{\mathcal{F}}(\mu_V) = \tilde{\mathfrak{B}}_V, \tag{48}$$

$$\mathcal{F}(\rho_V) = \mathfrak{B}_{V,1}, \qquad \tilde{\mathcal{F}}(\rho_V) = \tilde{\mathfrak{B}}_{V,1}. \tag{49}$$

In particular, we call $\Phi(a_1 a_2 \dots)$ a *tree address* of the fractal set $\mathcal{F}(a_1 a_2 \dots)$, as well as of the fractal measure $\tilde{\mathcal{F}}(a_1 a_2 \dots)$.

The map from V-groves to V-variable fractals allows us to lift properties of Ω_V to these fractals. For example, the property of V-variability of V-trees has an interesting interpretation in terms of the fractal sets $\mathfrak{H}_{V,1}$: at any "level of magnification", any V-variable fractal sets is made of up to V "forms" or "shapes".

To be more precise, let $\varepsilon > 0$ represent any "scale". Then any V-variable fractal set in $\mathfrak{H}_{V,1}$ is a finite union of continuous transformations of at most V distinct non-empty compact subsets in $\mathbb{H}(\mathbb{X})$, and the diameter of these transformed sets is at most ε. Similarly, any V-variable fractal measure in $\tilde{\mathfrak{H}}_{V,1}$ is a finite weighted superposition of continuous transformations of at most V distinct normalized measures in $\mathbb{P}(\mathbb{X})$, and the diameter of the support of these measures is at most ε.

The second property that extends from Ω_V to the V-variable fractals is there ability to approximate canonical random code trees. That is, by using the continuity of the maps $\mathcal{F} : \Omega \to \mathbb{H}(\mathbb{X})$ and $\tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{X})$:

$$\lim_{V\to\infty} \mathfrak{H}_{V,1} \overset{(47)}{=} \lim_{V\to\infty} \mathcal{F}(\Omega_{V,1}) \overset{\mathcal{F}\text{ cont.}}{=} \mathcal{F}(\lim_{V\to\infty} \Omega_{V,1}) \overset{(35)}{=} \mathcal{F}(\Omega) \overset{(28)}{=} \mathfrak{H}, \tag{50}$$

$$\lim_{V\to\infty} \tilde{\mathfrak{H}}_{V,1} \overset{(47)}{=} \lim_{V\to\infty} \tilde{\mathcal{F}}(\Omega_{V,1}) \overset{\tilde{\mathcal{F}}\text{ cont.}}{=} \tilde{\mathcal{F}}(\lim_{V\to\infty} \Omega_{V,1}) \overset{(35)}{=} \tilde{\mathcal{F}}(\Omega) \overset{(28)}{=} \tilde{\mathfrak{H}}, \tag{51}$$

and when the probabilities $\mathcal{P}^a$ obey equation (15), then by using the continuity of the maps $\mathcal{F} : \Omega \to \mathbb{H}(\mathbb{X})$ and $\tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{X})$: [7]

$$\lim_{V\to\infty} \mathfrak{B}_{V,1} \overset{(49)}{=} \lim_{V\to\infty} \mathcal{F}(\rho_V) \overset{\mathcal{F}\text{ cont.}}{=} \mathcal{F}(\lim_{V\to\infty} \rho_V) \overset{(36)}{=} \mathcal{F}(\rho) \overset{(29)}{=} \mathfrak{B}, \tag{52}$$

$$\lim_{V\to\infty} \tilde{\mathfrak{B}}_{V,1} \overset{(49)}{=} \lim_{V\to\infty} \tilde{\mathcal{F}}(\rho_V) \overset{\tilde{\mathcal{F}}\text{ cont.}}{=} \tilde{\mathcal{F}}(\lim_{V\to\infty} \rho_V) \overset{(36)}{=} \tilde{\mathcal{F}}(\rho) \overset{(29)}{=} \tilde{\mathfrak{B}}. \tag{53}$$

6 Computer implementation

For applications, one is interested in the actual computation of fractals, and for the scope of this report, in the actual computation of V-variable fractals. To

[7] The proofs of Theorems 19 and 24 of [2003a] do not mention the correct continuous maps. In the second part of the proof of Theorem 19 a map $\mathcal{F} : \Omega \to \mathbb{P}(\mathbb{X})$ is mentioned, but apparently either the map $\mathcal{F} : \Omega \to \mathbb{H}(\mathbb{X})$, or the derived push-forward map $\mathcal{F} : \mathbb{P}(\Omega) \to \mathbb{P}(\mathbb{H}(\mathbb{X}))$ is meant.

In the second part of the proof of Theorem 24 a map $\tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{P})$ is mentioned, but apparently either the map $\tilde{\mathcal{F}} : \Omega \to \mathbb{P}(\mathbb{X})$, or the derived push-forward map $\tilde{\mathcal{F}} : \mathbb{P}(\Omega) \to \mathbb{P}(\mathbb{P}(\mathbb{X}))$ is meant.

this end, one must account not only for the random aspect of the process, but also for some way to represent elements of $\mathbb{H}(\mathbb{X})$ and $\mathbb{P}(\mathbb{X})$ in a computer and for a way to apply transformations f_m^n to them.

We will restrict ourselves to $\mathbb{X} = \mathbb{R}^d$, equipped with the corresponding Euclidean metric, and in particular to the cases $d = 2$ and $d = 3$, which are the most important cases appearing in computer graphics. The easiest transformations to apply to these spaces are affine transformations, which are of the form $\mathbf{x} \mapsto A\mathbf{x} + \mathbf{b}$ with $A \in \mathbb{R}^{d \times d}$ and $\mathbf{b}, \mathbf{x} \in \mathbb{R}^d$.

6.1 Representation

Two common ways to represent elements of $\mathbb{H}(\mathbb{R}^d)$ on computers are:

- Bitmaps; based on a division into a d-dimensional grid of cells (or pixels).
- Vector graphics; based on descriptions for the boundaries of objects.

The bitmap representation is used mainly to represent images in two dimensions. Most computer image formats, like PNG and JPEG are based on it. The vector graphics representation is used for example in the PostScript and PDF standards (though these also support inclusion of bitmap graphics).

Because the storage requirements for describing objects in $\mathbb{R}^3$ using bitmap graphics grow very fast with respect to the desired precision, a vector-like representation is often used for three dimensions. For example, computer programs using the OpenGL or Direct3D standards — such as games — describe the the shape of objects basically by their boundary, given as the composition of simple geometric shapes such as triangles. The process of converting such vector based descriptions to a 2-dimensional bitmap representation suitable for display on a computer screen is called *rendering*. Personal computers are often equipped with a *graphics card* which provides hardware support for rendering.

The surface of the represented objects is often furnished with 2-dimensional bitmap called *textures*, but other methods to specify their appearance exist. In particular, modern graphics cards support *pixel shaders* which are little programs that are executed for each pixel rendered to determine their color on the output screen.

Elements from $\mathbb{P}(\mathbb{R}^d)$ can be interpreted as greyscale images. It is practical to represent them by bitmaps in which every pixel gets assigned a number in a fixed range like $0 \ldots 255$ or $0 \ldots 65535$, a shade of grey.

6.2 Transformations

Applying a given transformation f to a bitmap is done most easily using its inverse f^{-1}, if it exists. To determine the value of an output pixel at coordinate $\mathbf{x} \in \mathbb{R}^d$, simply look at the pixel in the input at coordinates $f^{-1}(\mathbf{x})$. In general this procedure causes some loss of information (even if there is no scaling in f), and averaging techniques can be applied to smooth jagged boundaries a bit.

An alternative way of dealing with (affine) transformations is commonly employed in vector graphics systems: they allow the coordinate system to be transformed. Apart from that, they allow one to "push" the current "graphics state", which includes the current coordinate transformation, to a stack and to restore it later by a corresponding "pop" operation.

Suppose f is an affine transformation, then drawing a transformed version $f(K)$ of an object K is done by the following steps:

1. Push the current coordinate transformation onto the stack.
2. Transform the current coordinate transformation by f.
3. Draw the object K.
4. Pop the coordinate transformation from the stack.

6.3 Computing V-variable fractals

When generating V-variable fractals, a straightforward implementation uses a set of V bitmaps. At each iteration step, the selected transformations are applied to the selected "input buffers" to form the contents of the V "output buffers". All buffers are represented by bitmaps.

A vector representation could be chosen for the buffers, but one must take care of the increasing complexity that occurs as the union of (transformed) buffers is stored in the output buffer. One way to deal with that is by discarding objects that become too small, or perhaps replace them by simpler objects.

These methods are quite efficient. This is because we are dealing with the forward process, the forward process of a V-variable superfractal. The amount of information that describes the generated approximation at the n-th iteration is given only by the indices $a_1 a_2 \dots a_n \in \mathcal{A}^n$. Because it is a forward process, we only need to use the last V buffers. Of course, this is tightly bounded to the V-variable structure of the underlying fractal and code trees.

Another approach to rendering, especially suitable for $\mathbb{R}^3$, is to use coordinate transformations. Recall that the V-trees are code trees that specify the hierarchy of transformations that is to be applied. Each m-th limb of a node $i \in T$ labelled with the IFS index $n = \omega(i)$, $\omega \in \Omega_{V,v}$ corresponds to a transformation f_n^m. After having chosen k indices $a_1 a_2 \dots a_n \in \mathcal{A}^k$ with probability $\rho_V([\eta^{a_1 \circ \dots \circ a_k}]) = \mathcal{P}^{a_1} \cdots \mathcal{P}^{a_k}$, tree level-$k$ contruction tree $a_1 \circ \dots \circ a_k$ can be walked in depth-first order, meanwhile pushing and popping the transformations corresponding to each limb. Only when reaching a k-level node, whose limb is labelled with v, the v-th initial set is drawn.

Though this approach does not reuse the intermediate buffer contents, its runtime can be limited by pruning the construction tree whenever the current coordinate transformation has a scaling smaller than a certain limit. At such a point, something can be drawn to replace the subtree that is cut off. The replacement shape could be a fixed shape, possibly estimated from previously generated samples of the fractal, but this would break the V-variability at the given scale. Still, this can be acceptable for applications.

There is another way of dealing with pruning: don't prune on a small scale, but keep the rescursion level k small. Apply this procedure iteratively, where the replacement shapes are "simplified" versions of the v-th outputs of the previous iteration. The indices v correspond to the labels of the corresponding limbs at level k.

This method preserves V-variability nicely, but it may be difficult for particular implementations to obtain "simplified" versions of previous outputs. For bitmap implementations, it is simple however. In fact, it corresponds to the straightforward procedure described at the beginning of this subsection, applied to the IFS

$$\{\mathbb{X}^V; f^{a_1} \circ \ldots \circ f^{a_k}, \mathcal{P}^{a_1 \circ \ldots \circ a_k}, a_1 \ldots a_k \in \mathcal{A}^k\}, \tag{54}$$

where $\mathcal{P}^{a_1 \circ \ldots \circ a_k} := \mathcal{P}^{a_1} \cdots \mathcal{P}^{a_k}$, with corresponding code-space generating IFS

$$\{\Omega^V; \eta^{a_1 \circ \ldots \circ a_k}, \mathcal{P}^{a_1 \circ \ldots \circ a_k}, a_1 \ldots a_k \in \mathcal{A}^k\}. \tag{55}$$

6.4 An example

For the purpose of this report, I created a little program that renders V-variable fractals using the OpenGL standard. It does not exploit any advanced pruning or substitution techniques. See Figures 1 through 6 for some examples of images it produced.

References

[BHS] M.F. Barnsley, J.E. Hutchinson and Ö. Stenflo, *V-variable Fractals and Superfractals*

[2003a] M.F. Barnsley, J.E. Hutchinson and Ö. Stenflo, *A fractal value random iteration algorithm and fractal hierarchy*, Submitted to SIAM review

[2003b] M.F. Barnsley, J.E. Hutchinson and Ö. Stenflo, *Dimension and approximation properties of V-variable fractals*, In preparation

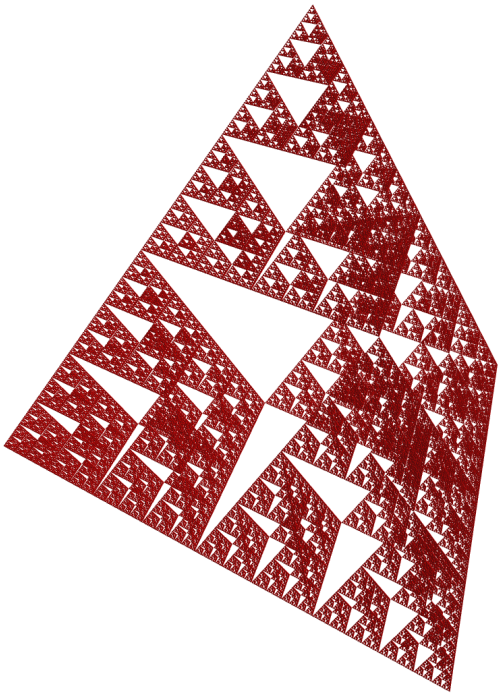

Figure 1: A 3-dimensional Sierpinski fractal.

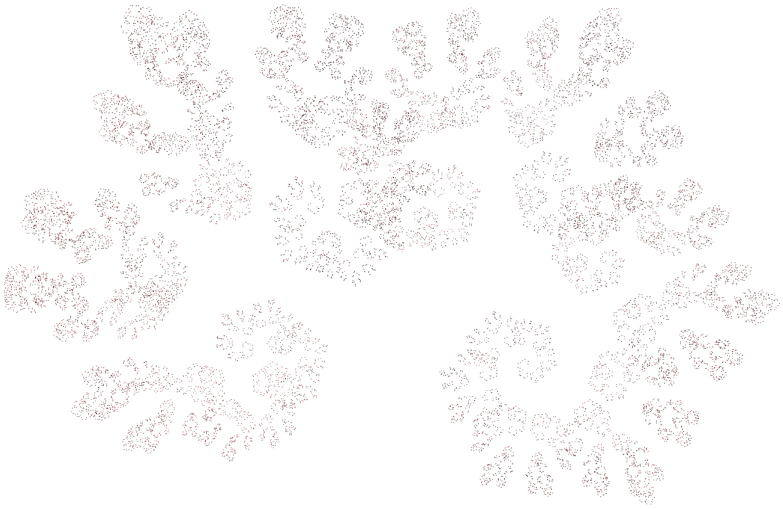

Figure 2: A 3-dimensional Pythagoras tree (normal mode: only the n-th approximation to the attractor set is shown).

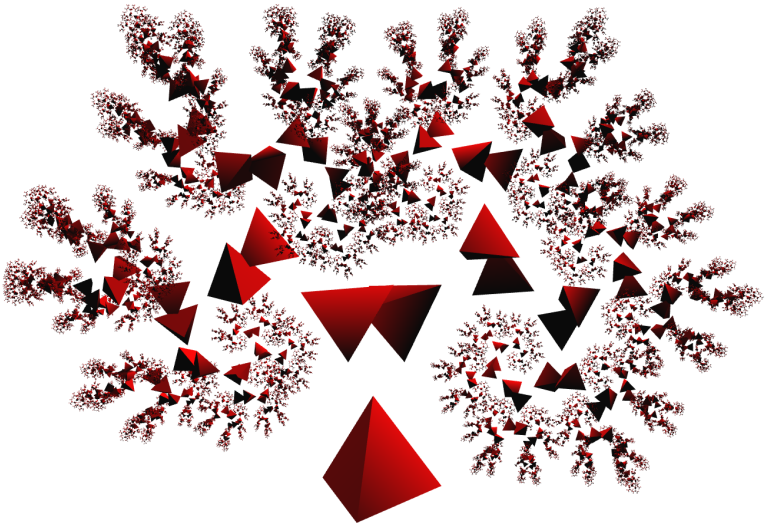

Figure 3: A 3-dimensional Pythagoras tree (sticky mode: approximations from previous iterations are shown as well).

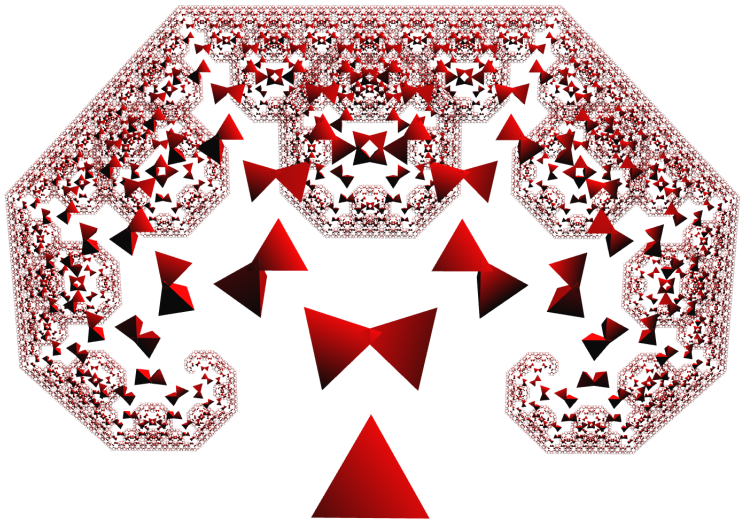

Figure 4: A 2-dimensional Pythagoras tree rendered in three-dimensional space (sticky mode: approximations from previous iterations are shown as well).

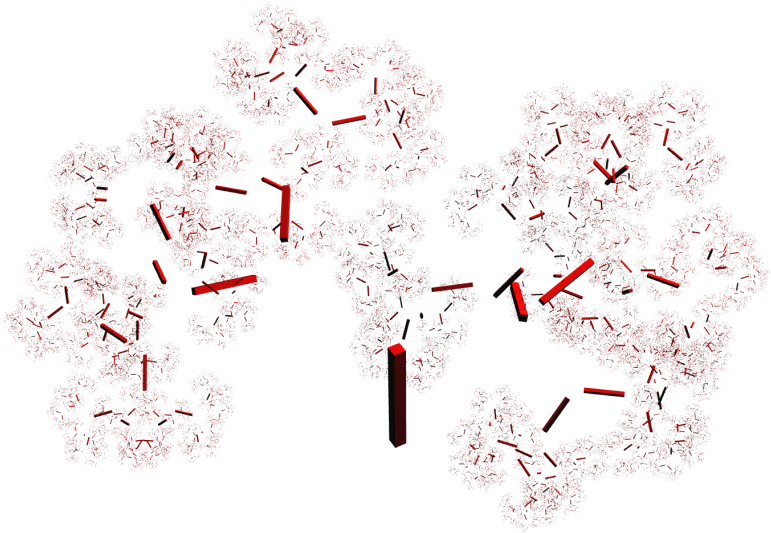

Figure 5: A 3-dimensional Pythagoras tree with an alternating pattern, simulated by using four component maps instead of the regular two (sticky mode: approximations from previous iterations are shown as well). The initial figure is a bar instead of a cube.

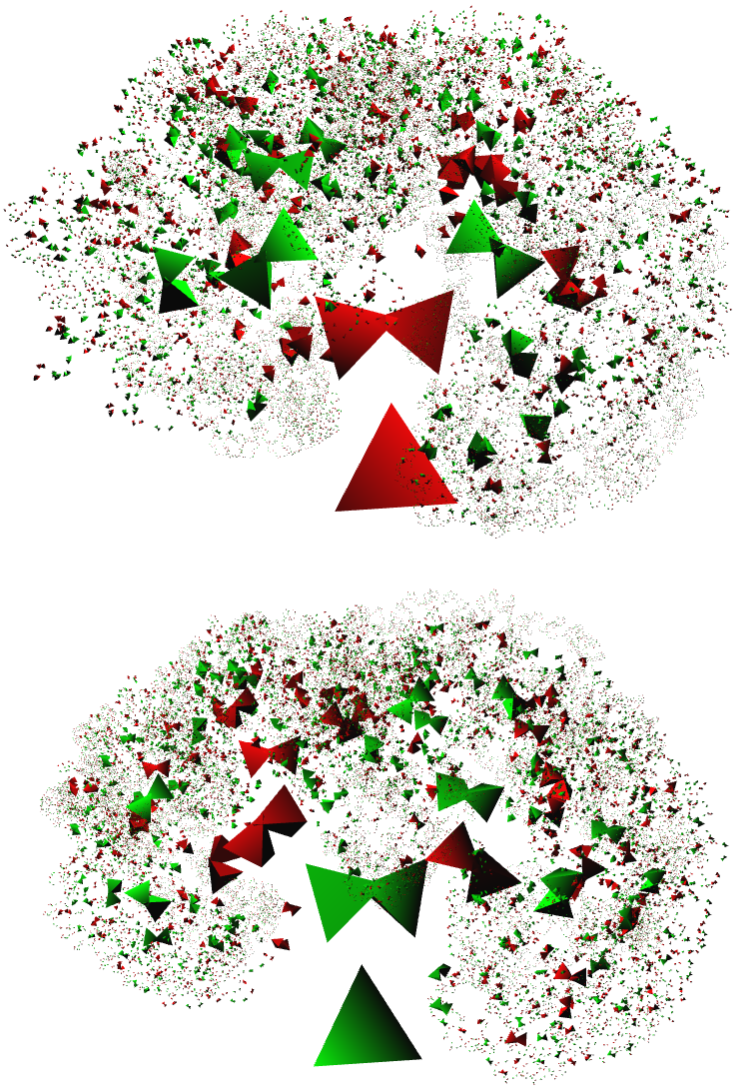

Figure 6: A 2-variable 3-dimensional Pythagoras tree (sticky mode: approximations from previous iterations are shown as well). The two colors correspond to the two initial buffers. The 2-variability is subtle, but it is clear the trees are not as regular as those in Figure 3 and 4. The two chosen IFSs are from these two figures respectively.