Language-Restricted Iterated Function Systems, Koch Constructions, and L-systems

Przemyslaw Prusinkiewicz and Mark Hammel
Department of Computer Science
University of Calgary
Calgary, Alberta, Canada T2N 1N4
e-mail: pwp|hammel@cpsc.ucalgary.ca

From *New Directions for Fractal Modeling in Computer Graphics*,
SIGGRAPH '94 Course Notes. ACM Press, 1994.

Language-Restricted Iterated Function Systems, Koch Constructions, and L-systems

Przemyslaw Prusinkiewicz and Mark Hammel

Department of Computer Science
University of Calgary
Calgary, Alberta, Canada T2N 1N4
(pwp@cpsc.ucalgary.ca, hammel@cpsc.ucalgary.ca)

1 Introduction

Linear fractals can be generated using a variety of methods. This raises the question of finding equivalent methods for generating the same fractal. Several aspects of this question have been addressed in the literature. These include:

- A method for converting Koch constructions to equivalent iterated function systems (IFS's) [11],
- Methods for converting selected classes of L-systems with geometric interpretation to extensions of IFS's, such as controlled iterated function systems (CIFS's) [10] and mutually recursive function systems (MRFS's) [4].

Previous course notes [9] introduced the notion of language-restricted iterated function systems (LRIFS's) encompassing CIFS's and MRFS's, and included a number of sample LRIFS's equivalent to L-systems with turtle interpretation. The present notes include the following further extensions to these results:

- Introduction of the notions of iterated transformations of coordinate systems (ITCS's) and their language-restricted generalization (LRITCS's),
- A characterization of Koch constructions in terms of ITCS's,
- A method for constructing an LRITCS equivalent to a given LRIFS,
- Expression of LRITCS's using parametric L-systems with turtle interpretation.

Readers are advised that this work is still in progress, and consequently the results are not presented with the rigor expected in final publications.

2 Iterated function systems

Iterated Function Systems (IFS's) are among the basic methods for generating fractals. The term itself was introduced by Barnsley and Demko [2], but the essential concept is usually attributed to Hutchinson [7]. Vrscay [12] traces the idea further back to Williams [13], who studied fixed points of finite compositions of contractive maps. A detailed introduction to IFS's is presented in [1].

Below we summarize the central notions of the IFS theory and introduce the notation used in subsequent parts of these notes.

Let X be a complete metric space with distance function d. The distance between point $a \in X$ and set $B \subset X$ is defined as:

$$d(a, B) = \inf_{b \in B} d(a, b).$$

The *half-distance* between set $A \subset X$ and set $B \subset X$ is equal to:

$$d'(A, B) = \sup_{a \in A} d(a, B).$$

Note that, in general, $d'(A, B) \neq d'(B, A)$. The *distance between sets* A and B is the greater of the two half-distances:

$$\rho(A, B) = \max\{d'(A, B), d'(B, A)\}.$$

The function $\rho(A, B)$ satisfies the distance axioms in the space $H(X)$ of all closed nonempty bounded subsets of the space X and is called the *Hausdorff metric* on this space.

A function (transformation) $F : X \to X$ is called a *contraction*, if there is a constant $r < 1$ such that

$$d(F_i(x), F_i(y)) \leq r d(x, y)$$

for all $x, y \in X$.

A transformation $F : X \to X$ is extended to the domain $H(X)$ of subsets of X:

$$F(A) = \{F(x) : x \in A\}.$$

Since the values of F are sets, it is possible to perform set-theoretic operations on them. Let $\mathcal{F} = \{F_1, \ldots, F_N\}$ be a set of functions $F_i : X \to X$, extended to the domain $H(X)$ as described above. The equation

$$\mathcal{S}(A) = \bigcup_{i=1}^{N} F_i(A)$$

defines a function $\mathcal{S} : H(X) \to H(X)$ associated with the set $\mathcal{F}$. Hutchinson [7, page 728] showed that if all functions F_i are contractions in space X with metric d, then $\mathcal{S}$ is a contraction in the space $H(X)$ with the Hausdorff metric ρ.

The space $H(X)$ and the transformation $\mathcal{S}$ satisfy conditions of *Banach's fixed point theorem* [3, page 778], presented here in a narrowed version.

Banach's fixed point theorem. Let M be a complete metric space, and suppose that $T : M \to M$ is a contractive transformation in M. Then for any initial element $x_0 \in M$, the iterative process $x_{n+1} = T(x_n)$, $n = 0, 1, 2, \ldots$, can be continued indefinitely, and the sequence $\{x_n\}$ converges to an element $x \in M$, which is the unique solution of the equation $x = T(x)$.

According to this theorem, there is a unique set $\mathcal{A} \in H(X)$, such that

$$\mathcal{A} = \mathcal{S}(\mathcal{A}) = \bigcup_{i=1}^{N} F_i(\mathcal{A}). \tag{1}$$

The set $\mathcal{F}$ of contractive mappings $F_1, F_2, \ldots, F_N$ is called an *iterated function system*, and the set $\mathcal{A}$ is called the *attractor* of $\mathcal{F}$.

Let $\mathcal{S}^n$ denote the n-fold *power* of the transformation $\mathcal{S}$, defined recursively by the formulae $\mathcal{S}^0(A) = A$ and $\mathcal{S}^n(A) = \mathcal{S}^{n-1}(\mathcal{S}(A))$, where $n = 1, 2, 3, \ldots$. The fixed point theorem states that, for any compact set A, the sequence $\mathcal{S}^n(A)$ converges to the attractor $\mathcal{A}$ in the space $(H(X), \rho)$,

$$\lim_{n \to \infty} \mathcal{S}^n(A) = \mathcal{A}. \tag{2}$$

Let $\mathcal{S}^\star$ denote the *iteration* of the transformation $\mathcal{S}$,

$$\mathcal{S}^\star(A) = \bigcup_{n=0}^{\infty} \mathcal{S}^n(A).$$

We will show that $\mathcal{S}^\star(A) = \mathcal{A}$ for any $A \subseteq \mathcal{A}$. Taking the definition of transformation $\mathcal{S}$ into account, the following inclusions hold:

$$\begin{array}{rcl}
A & \subseteq & \mathcal{A} \\
\mathcal{S}(A) & \subseteq & \mathcal{S}(\mathcal{A}) = \mathcal{A} \\
S^n(A) & \subseteq & \mathcal{A} \ \text{ for all } \ n = 0, 1, 2, \ldots \\
\mathcal{S}^\star(A) & \subseteq & \mathcal{A}.
\end{array}$$

On the other hand,

$$\mathcal{A} = \lim_{n\to\infty} \mathcal{S}^n(A) \subseteq \mathcal{S}^\star(A),$$

thus, in conclusion,

$$\mathcal{A} = \mathcal{S}^\star(A).$$

The above equation provides the basic method for constructing the attractor $\mathcal{A}$ by selecting a starting point $x_0 \in \mathcal{A}$ and applying to it all possible sequences of transformations from $\mathcal{F}$. Of course, in practice it is impossible to consider an infinite number of sequences, and the construction ends after a finite number of steps. Various strategies for choosing subsequent transformations and terminating the approximation of the attractor are discussed in [5, 6].

In the scope of these notes, we are interested in iterated function systems consisting of linear functions F_i in the plane. A legible method for expressing them is important in practice, since it can make IFS's easier to specify and understand. We will express transformations by composing operations of translation, rotation, and scaling. The following notation is observed:

- $t(a,b)$ is a translation by vector (a,b):

$$\begin{aligned} x' &= x + a \\ y' &= y + b \end{aligned}$$

- $r(\alpha)$ is a rotation by angle α with respect to the origin of the coordinate system:

$$\begin{aligned} x' &= x \cos\alpha - y \sin\alpha \\ y' &= x \sin\alpha + y \cos\alpha \end{aligned}$$

- $s(r_x, r_y)$ is a scaling with respect to the origin of the coordinate system:

$$\begin{aligned} x' &= r_x x \\ y' &= r_y y \end{aligned}$$

  If $r = r_x = r_y$, we write $s(r)$ instead of $s(r,r)$.

We compose transformations from left to right. For instance, if $F_1 = t(a,b)$, $F_2 = r(\alpha)$, and $F_3 = s(r)$, then

$$x \circ t(a,b) \circ r(\alpha) \circ s(r) = x \circ F_1 \circ F_2 \circ F_3 = F_3(F_2(F_1(x))).$$

The operator of composition is often omitted without ambiguity,

$$x \circ F_1 \circ F_2 \circ F_3 = xF_1F_2F_3.$$

Whenever a transformation cannot be (conveniently) represented in terms of translations, rotations, and scalings, we can use matrix notation.

Example. Figure 1(a) illustrates the operation of an iterated function system defined by the following transformations:

$$\begin{aligned} F_a &= s(0.5) \\ F_b &= s(0.5)t(0.5,0) \\ F_c &= s(0.5)t(0,0.5) \end{aligned}$$

For instance, the composed transformation $F_b \circ F_c \circ F_a$ takes the origin of the coordinate system $O(0,0)$ to point $P(0.125, 0.25)$.

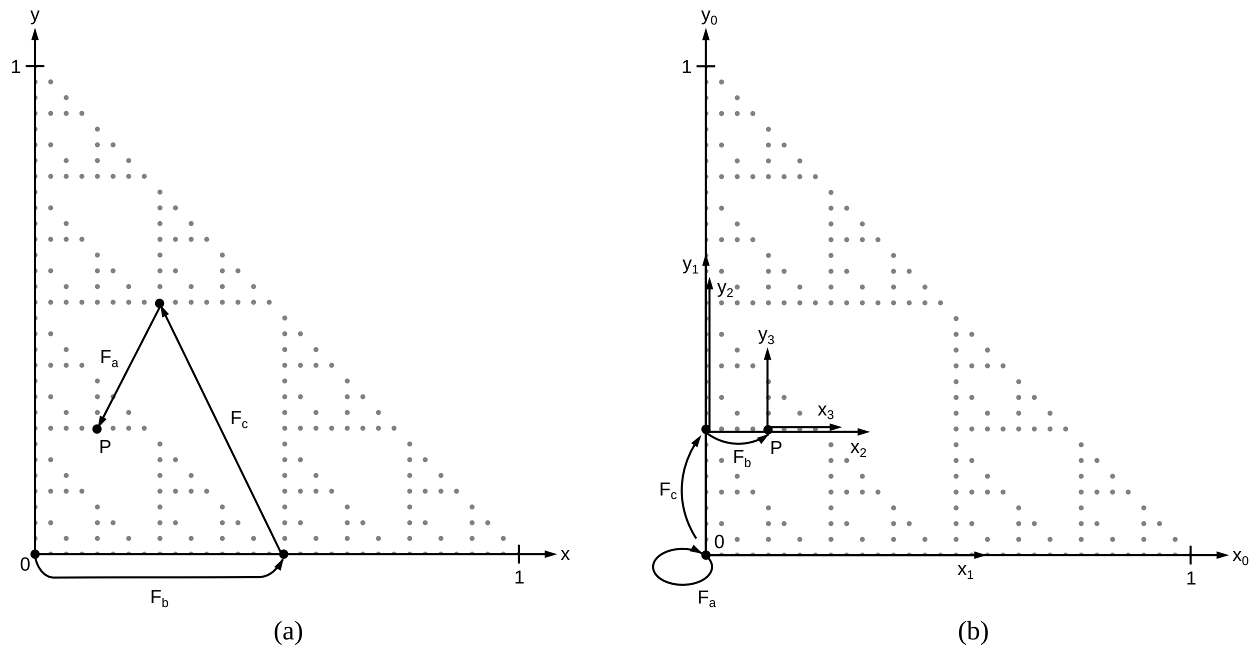

Figure 1: Sequences of transformations taking point O to point P. (a) All transformations are expressed in the same global coordinate system. (b) Each transformation is expressed in a local coordinate system resulting from the application of previous transformations.

3 Iterated transformations of coordinate systems

In the case of iterated function systems, all transformations being composed are expressed in the same global coordinate system. We may also consider a dual construct in which the application of any transformation changes the coordinate system in which the subsequent transformation will be expressed.

Example. Figure 1(b) illustrates a mapping of point O to point P through a sequence of transformations of coordinate systems. Transformation F_a, applied first, scales the original coordinate system xy by 0.5, yielding a system x_1y_1. Transformation F_c, *expressed in coordinate system* x_1y_1, scales x_1y_1 by 0.5 and translates it by vector $(0, 0.5)$. Coordinate system x_2y_2 results. The final transformation F_b, expressed in coordinate system x_2y_2, scales x_2y_2 by 0.5 and translates it by vector $(0.5, 0)$. Point P is the image of O in this sequence of transformations.

While comparing Figures 1(a) and 1(b) we may notice that the sequence of transformations F_b, F_c, F_a applied in the global coordinate system yields the same result as the reverse sequence of transformations, F_a, F_c, F_b, each applied in the local coordinate system obtained by previous transformations. We will show that this equality of results is a general property of arbitrary sequences of transformations.

Consider a transformation T of a coordinate system x_0y_0 to a system x_1y_1, a mapping R that relates figures A and B in the system x_0y_o, and the mapping $R^{(T)}$ that relates images $A^{(T)}$ and $B^{(T)}$ of A and B in the system x_1y_1. As illustrated by Figure 2, the following equalities hold:

$$\begin{aligned} A \circ R &= B, \\ A \circ T &= A^{(T)}, \\ B \circ T &= B^{(T)}, \\ A^{(T)} \circ R^{(T)} &= B^{(T)}. \end{aligned}$$

By combining these equations, we obtain:

$$A \circ T \circ R^{(T)} = B \circ T = A \circ R \circ T.$$

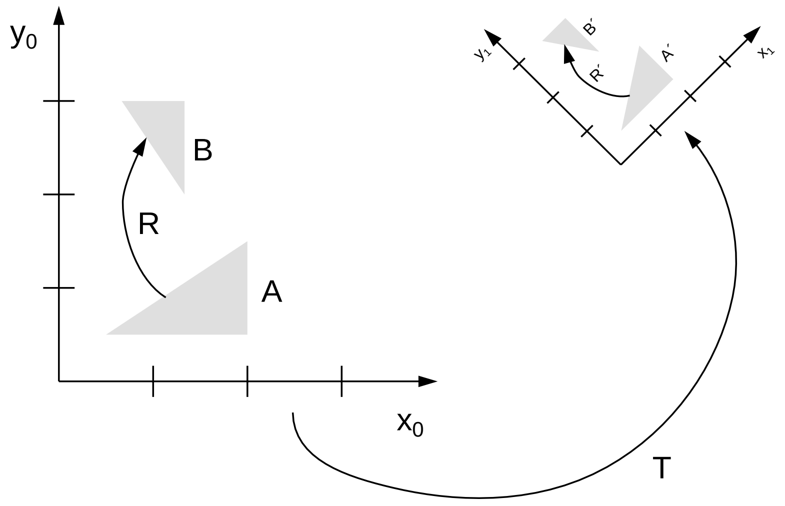

Figure 2: Relationship between figures A and B and their images in a transformed coordinate system.

As no assumptions regarding figure A were made, the above equality holds for all A, thus

$$T \circ R^{(T)} = R \circ T.$$

The following theorem extends this result to arbitrary sequences of transformations.

Theorem. Let $T_1 * T_2 * \cdots * T_n$ denote the result of composing of a sequence of $n \geq 1$ transformations $T_1, T_2, \ldots T_n$ in the following manner:

- transformation T_1 is expressed in the original coordinate system x_0y_0,
- for any $i \in \{1, 2, \ldots, n-1\}$, transformation T_{i+1} is expressed in the coordinate system x_iy_i that results from the application of transformation T_i to the system $x_{i-1}y_{i-1}$.

Then the following equality holds:

$$T_1 * T_2 * \cdots * T_n = T_n \circ \cdots \circ T_2 \circ T_1$$

Proof: by induction on n.

- For $n = 1$ the thesis reduces to the obvious equality $T_1 = T_1$.
- Assume the thesis true for some $n \geq 1$. Using the definition of the operation $*$ and the equality $T \circ R^{(T)} = R \circ T$, we obtain:

$$T_1 * T_2 * \cdots * T_n * T_{n+1} = (T_1 * T_2 * \cdots * T_n) \circ T_{n+1}^{(T_1 * T_2 * \cdots * T_n)} = T_{n+1} \circ (T_1 * T_2 * \cdots * T_n) = T_{n+1} \circ T_n \circ \cdots \circ T_2 \circ T_1 \quad \square$$

The above theorem indicates that any fractal generated by an iterated function system can be alternatively obtained by an iterated transformation of coordinate systems. Koch constructions are an important special case of such iterated transformations. Recall that a Koch construction consists of recursively replacing edges of an arbitrary polygon by an open polygon *reduced and displaced* so as to have the same endpoints as those of the interval being replaced [8]. The operations of reducing and displacing the successor polygon can be described as a transformation of the coordinate system in which this polygon is defined. The fact that lines rather than other figures (such as the origins of the coordinate system, for example) are drawn does not affect the limit figure produced. This is analogous to the observation that the attractor of an iterated function system does not depend on the choice of the initial figure subject to the transformations.

A convenient notation for specifying Koch constructions, and iterated transformations of coordinate systems in general, is provided by parametric L-systems with turtle interpretation, described in [10]. For example, the following

L-system generates the Sierpinski gasket in the manner illustrated in Figure 1(b):

$$\begin{array}{rll} \omega: & @o(0.01,1) & \\ p: & @o(d,s) & \rightarrow \quad [@o(d,s*0.5)][-f(s*0.5)+@o(d,s*0.5)][f(s*0.5)@o(d,s*0.5)] \end{array}$$

The angle associated with the symbols $+$ and $-$ is equal to 90°.

The expression $@o(d,s)$ denotes a circle of diameter d, thought of as the origin of a local coordinate system xy. Its position is determined by the current position of the turtle. The x and y axes are aligned with the vectors $\vec{H}$ and $-\vec{L}$, which specify the turtle's heading and direction to the right. Parameter s determines the unit length in the system xy. The essence of iterated transformations of coordinate systems is captured by production p, which replaces current system xy by three new systems, displaced with respect to each other, and scaled by 0.5.

We are now well positioned to describe language-restricted iterated function systems (LRIFS's) and their counterpart, language-restricted iterated transformations of coordinate systems (LRITCS's). In both cases, the main idea is to limit the set of allowable sequences of transformations. To this end, we identify sequences of transformations using words over the alphabet of transformation labels V. In consequence, sets of sequences of transformations can be viewed as formal languages over the alphabet V. The necessary notions of formal language theory are recalled in the next section.

4 Formal languages

An *alphabet* V is a finite nonempty set of *symbols* or *letters*. A *string* or *word* over alphabet V is a finite sequence of zero or more letters of V, whereby the same letter may occur several times. The string consisting of zero letters is called the *empty word* and is denoted by ϵ. The *concatenation* of words $x = a_1a_2\ldots a_m$ and $y = b_1b_2\ldots b_n$ is the word formed by extending the sequence of symbols x with the sequence y, thus $xy = a_1a_2\ldots a_mb_1b_2\ldots b_n$.

The set of all words over V is denoted by $V^\star$. A *formal language* over an alphabet V is any set L of words over V, hence $L \subset V^\star$. Since formal languages are sets, the set-theoretic operations can be performed on languages. In addition, concatenation is extended to languages by the formula:

$$L_1L_2 = \{xy : x \in L_1 \,\&\, y \in L_2\}.$$

The n-th *power* of a language L is defined recursively as $L^0 = \epsilon$, $L^n = LL^{n-1}$ for $n = 1,2,3,\ldots$. The *iteration* of language L is the union of all its powers,

$$L^\star = \bigcup_{n=0}^{\infty} L^n.$$

If $w = a_1a_2\ldots a_n$, then the word $w^R = a_n\ldots a_2a_1$ is called the *mirror image* of w. It can be shown that $(xy)^R = y^Rx^R$ for any words x and y. The mirror image of words is extended to languages as follows:

$$L^R = \{w^R : w \in L\}.$$

The problem of providing finite specifications for infinite languages takes a central place in formal language theory. The two main techniques are grammars and automata. In these notes we focus on *regular languages*, which can be specified using *finite-state automata*.

A **nondeterministic finite-state (Rabin-Scott) automaton** is a quintuplet:

$$\mathcal{M} = < V, S, s_0, E, I >,$$

where:

- V is a finite set of symbols, called the alphabet,
- S is a finite set of states,
- $s_0 \in S$ is a distinguished element of the set S, called the initial state,
- $E \subset S$ is a distinguished subset of S, called the set of final states,

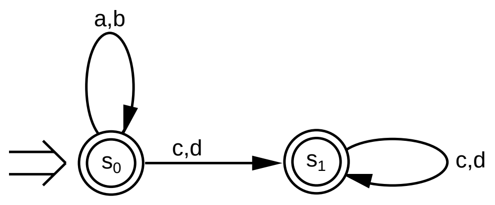

Figure 3: A sample finite automaton

- $I \subset V \times S \times S$ is a state transition relation.

Instead of $(a, s_i, s_k) \in I$, we often write $(a, s_i) \to s_k$.

At the beginning of the operation, the automaton is in initial state s_0. It processes a word $w = a_1 a_2, \ldots, a_n \in V^\star$ letter by letter, arriving at successive states $s_1, s_2, \ldots, s_n$, according to the state transition relation:

$$(a_1, s_0) \to s_1, \;\; (a_2, s_1) \to s_2, \ldots \;\; (a_n, s_{n-1}) \to s_n.$$

Processing stops when the string of symbols is exhausted. If the sequence of states $s_1, s_2, \ldots, s_n$ can be chosen in such a way that s_n is a final state, $s_n \in E$, then the word w is *accepted* by the automaton $\mathcal{M}$, otherwise it is *rejected*.

Formally, the state transition relation $\to$ is extended to the relation $\to^\star$ operating on words as follows:

- $(\epsilon, s_i) \to^\star s_i$ for all $s_i \in S$,
- if there exists such an $s_j \in S$ that $(w, s_i) \to^\star s_j$ and $(a, s_j) \to s_k$, then $(wa, s_i) \to^\star s_k$.

The set

$$L(s_k) = \{w \in V^\star : (s_0, w) \to^\star s_k\}$$

is called the *language associated with the state* s_k. The *language accepted by the automaton* $\mathcal{M}$ is defined as:

$$L(\mathcal{M}) = \bigcup_{s_k \in E} L(s_k) = \{w \in V^\star : (\exists s_k \in E)\; (s_0, w) \to^\star s_k\}.$$

Finite state automata are commonly represented as directed graphs, with the nodes representing states and arcs representing transitions. The initial state is indicated by a short arrow. If $E \neq S$, the final states are distinguished by double circles. A sample finite automaton is shown in Figure 3. It is known that given an automaton $\mathcal{M} = < V, S, s_0, E, I >$, the mirror image of the language $L(\mathcal{M})$ is accepted by the automaton:

$$\mathcal{M}^R = < V, S \cup \{s_0'\}, s_0', I^R >,$$

where

$$I^R = \{(\epsilon, s_0', s_K) : s_K \in E\} \cup \{(a, s_j, s_i) : (a, s_i, s_j) \in I\}.$$

Thus, the automaton $\mathcal{M}^R$ is obtained from $\mathcal{M}$ by:

- creating a new initial state $s_0' \notin S$,
- creating transitions labeled ϵ from s_0' to all final states of $\mathcal{M}$,
- reversing the direction of all other transitions,
- making s_0 the unique final state of $\mathcal{M}^R$.

For example, Figure 4 shows the automaton $\mathcal{M}^R$ corresponding to automaton $\mathcal{M}$ from Figure 3. Automaton $\mathcal{M}^R$ obtained using the above method may be nondeterministic, which means that it has more than one transition originating in the same state, and labeled by the same symbol. For example, the automaton $\mathcal{M}^R$ shown in Figure 4 has transitions (s_1, c, s_1) and (s_1, d, s_1) as well as (s_1, c, s_0) and (s_1, d, s_0). It is known that a deterministic automaton equivalent to a given nondeterministic automaton always exists and can be found algorithmically. The appropriate methods are presented in standard texts on finite automata theory. For example, Figure 5 presents a deterministic automaton $\mathcal{M}^{RD}$ equivalent to the automaton $\mathcal{M}^R$ in Figure 4.

We will now apply these notions to iterated function systems.

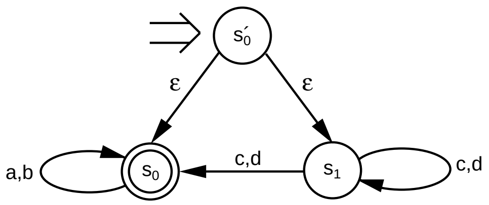

Figure 4: Automaton $\mathcal{M}^R$ defining the language $(L(\mathcal{M}))^R$, where $\mathcal{M}$ is shown in Figure 3.

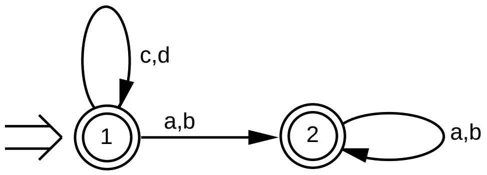

Figure 5: A deterministic automaton $\mathcal{M}^{RD}$ equivalent to the automaton $\mathcal{M}^R$ in Figure 4.

5 Language-restricted iterated function systems

A *language-restricted* iterated function system (LRIFS) is a quintuplet $\mathcal{F}_L = \langle X, \mathcal{F}, V, h, L\rangle$, where:

- X is the underlying metric space,
- $\mathcal{F}$ is an iterated function system in space X,
- V is an alphabet,
- $h : V \to \mathcal{F}$ takes letters of the alphabet V to mappings of the IFS $\mathcal{F}$,
- $L \subset V^\star$ is a language over the alphabet V.

The domain of the mapping h is extended from the set of letters V to the set of words $V^\star$ using the equation:

$$h(a_1a_2\ldots a_n) = h(a_1) \circ h(a_2) \circ \ldots \circ h(a_n).$$

According to this definition, the extended function h is a homomorphism from the monoid generated by the alphabet V with the operation of concatenation to the monoid generated by the iterated function system $\mathcal{F}$ with composition. The final extension of h from the domain of words to the domain of languages over V is given by the equation:

$$h(L) = \bigcup_{w\in L} h(w).$$

Thus, $H(L)$ is a function defined in the domain $H(X)$ of subsets of the space X.

If the language L consists of all words over alphabet V, then $h(L) = h(V^\star)$ is the set of all sequences of transformations in the IFS $\mathcal{F}$, and

$$h(L) = (F_1 \cup F_2 \cup \ldots \cup F_n)^\star = \mathcal{S}^\star.$$

In contrast, if L is a subset of $V^\star$, then only some composite transformations from $\mathcal{S}^\star$ belong to $h(L)$, hence $h(L) \subset S^\star$. If x_0 belongs to the attractor $\mathcal{A}$ of the IFS $\mathcal{F}$, and $\mathcal{A}_L(x_0)$ denotes the image of the set $\{x_0\}$ with respect to the transformation $h(L)$, the following inclusion holds:

$$\mathcal{A}_L(x_0) = x_0 \circ h(L) \subset x_0 \circ h(V^\star) = \mathcal{S}^\star(x_0) = \mathcal{A}.$$

Thus, the set $\mathcal{A}_L(x_0)$ generated by the LRIFS $\mathcal{F}_L$ with the starting point $x_0 \in \mathcal{A}$ is a subset of the attractor $\mathcal{A}$.

Example. Consider an LRIFS $\mathcal{F}_L = \langle X, \mathcal{F}, V, h, L\rangle$, where:

- the space X is the plane,

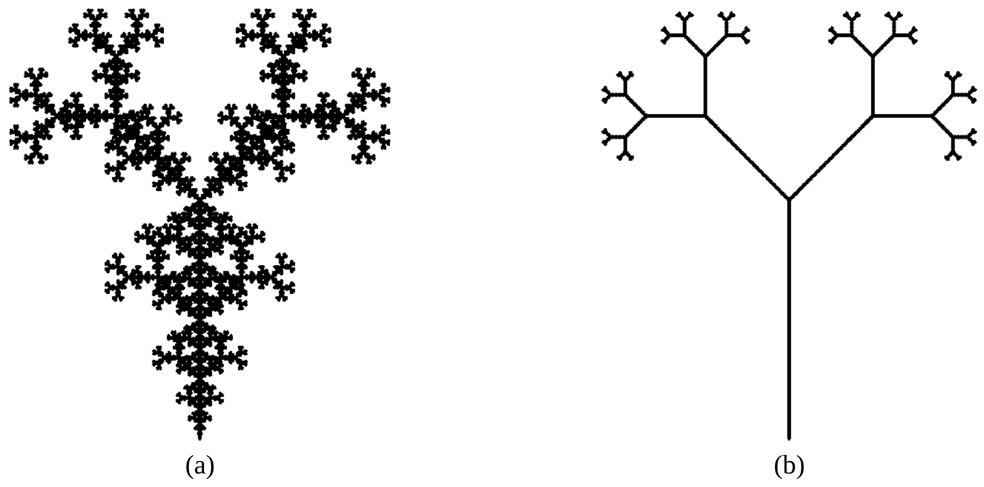

Figure 6: Attractor $\mathcal{A}$ and its subset $\mathcal{A}_L(x_0)$

- the IFS $\mathcal{F}$ consists of four transformations:

$$\begin{aligned} F_1 &= s(0.5) \\ F_2 &= s(0.5)t(0, 0.5) \\ F_3 &= s(0.5)r(45)t(0, 1) \\ F_4 &= s(0.5)r(-45)t(0, 1), \end{aligned}$$

- the alphabet V consists of four letters a, b, c, d,
- the homomorphism h is defined by:

$$h(a) = F_1, \;\; h(b) = F_2, \;\; h(c) = F_3, \;\; h(d) = F_4,$$

- the language L is defined by the regular expression

$$L = (a \cup b)^\star (c \cup d)^\star$$

  which corresponds to the automaton from Figure 3.

Figure 6 compares the attractor $\mathcal{A}$ of the IFS $\mathcal{F}$ with the set $\mathcal{A}_L(x_0)$ generated by the LRIFS $\mathcal{F}_L$ using the starting point $x_0 = (0, 0)$. Clearly, the branching structure of Figure (b) is a subset of the original attractor (a).

6 Conversion of LRIFS's to LRITCS's and L-systems

Language-restricted iterated transformations of coordinate systems (LRITCS's) are defined in the same manner as LRIFS's. The difference is that each transformation modifies the local coordinate system for the subsequent transformations. In other words, transformations are composed according to the operation $*$ rather than $\circ$.

We call an LRIFS $\mathcal{F}_{L_F}$ and an LRITCS $\mathcal{G}_{L_G}$ equivalent if they define the same limit set (have the same attractor $\mathcal{A}$). From Section 3 it follows that this equivalence occurs, if:

- both systems employ the same set of transformations $\mathcal{F}$, and
- the language L_G restricting the LRITCS $\mathcal{G}_{L_G}$ is the mirror image of the language L_F restricting the LRIFS $\mathcal{F}_{L_F}$:

$$L_G = L_F^R$$

For example, in order to control an LRITCS defining the attractor shown in Figure 6(b), we must consider the mirror image of sequences accepted by automaton $\mathcal{M}$ in Figure 3. The language L^R of the mirror sequence is specified by the automaton $\mathcal{M}^{RD}$ shown in Figure 5. Using the L-system notation to express this LRITCS, we obtain:

$$\begin{array}{llcll}
\omega : & @o(0.01, 1, 1) & & & \\
p_1 : & @o(d,s,t) : t == 1 & \rightarrow & [f(s) - (45)@o(d, s * 0.5, 1)] & /*F_3*/ \\
 & & & [f(s) + (45)@o(d, s * 0.5, q)] & /*F_4*/ \\
 & & & [@o(d, s * 0.5, 2)] & /*F_1*/ \\
 & & & [f(s * 0.5)@o(d, s * 0.5, 2)] & /*F_2*/ \\
p_2 : & @o(d,s,t) : t == 2 & \rightarrow & [@o(d, s * 0.5, 2)] & /*F_1*/ \\
 & & & [f(s * 0.5)@o(d, s * 0.5, 2)] & /*F_2*/
\end{array}$$

As in the L-system considered in the previous example, the module $@o(d, s, t)$ indicates the origin of a local coordinate system, which is translated, rotated, and scaled by productions. Parameter d determines the size of the circle marking this origin. Parameter s is the scaling factor, which determines the unit length in the local system. The value of parameter t corresponds to the state of automaton $\mathcal{M}^{RD}$ in Figure 5, and determines the admissible sequence of transformations.

A more involved example of a conversion of an LRIFS to an L-system is considered below. Figure 7 shows a structure $\mathcal{A}$ with an alternating-opposite branching pattern. This structure with all lines of the same length is generated by an LRIFS with the following transformations (c.f. [9]):

$$\begin{array}{lcl}
T_1 & = & s(0.5) \\
T_2 & = & s(0.5)t(0, 0.13) \\
T_3 & = & s(0.42)r(0.45) \\
T_4 & = & s(0.2)r(90)t(-0.05, 0.05) \\
T_5 & = & s(0.2)t(-0.05, 0.05) \\
T_6 & = & s(0.74)t(-0.078, 0.078) \\
T_7 & = & s(0.37)r(-45)t(0, 0.14) \\
T_8 & = & s(0.172)r(-90)t(0.05, 0.19) \\
T_9 & = & s(0.172)t(0.05, 0.19) \\
T_10 & = & s(0.74)t(0.0689, 0.1053) \\
T_11 & = & s(0.74)t(00.26)
\end{array}$$

The automaton $\mathcal{M}$ defining the language L of the allowable sequence of transformations is given in Figure 8. The corresponding deterministic automaton $\mathcal{M}^{RD}$ defining the mirror language L^R is shown in Figure 9. The resulting equivalent L-system is given below.

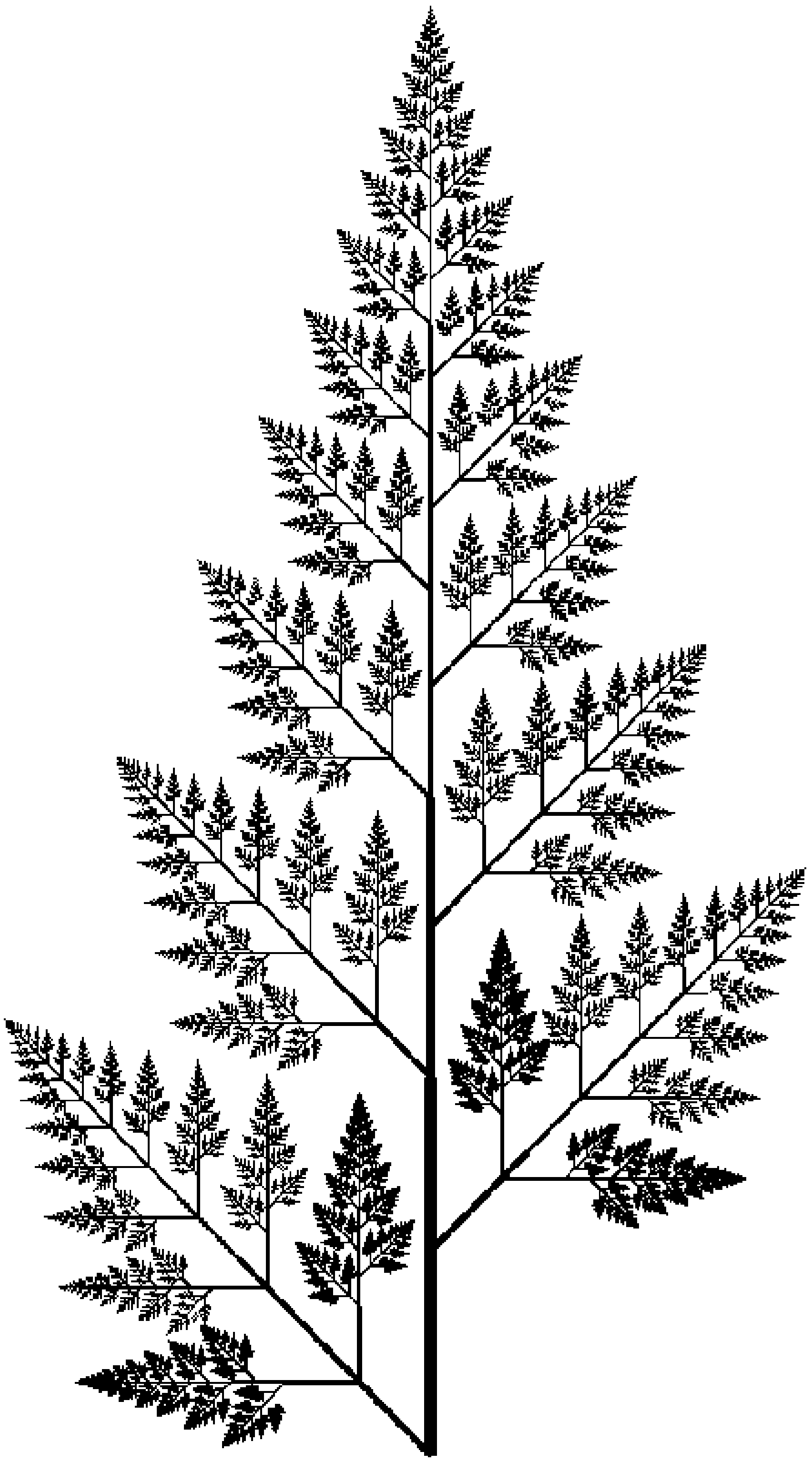

Figure 7: A structure $\mathcal{A}$ with an alternating-opposite pattern.

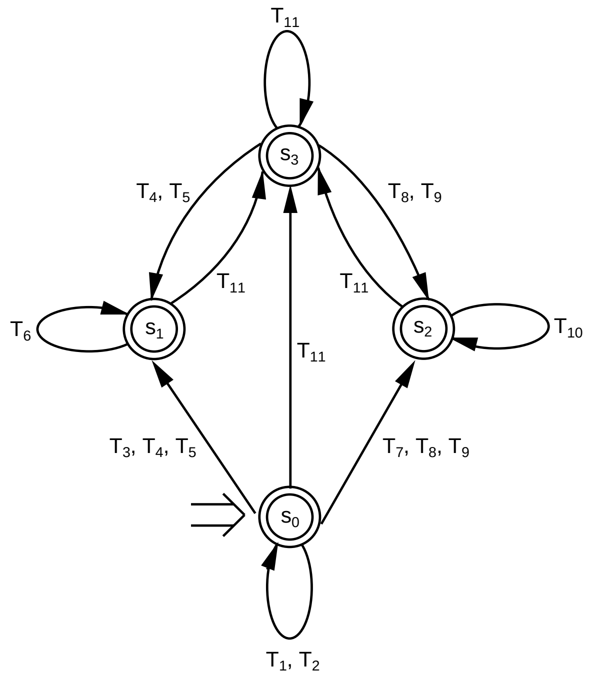

Figure 8: The automaton $\mathcal{M}$ defining a language of sequences of transformations T_1 — T_{11}.

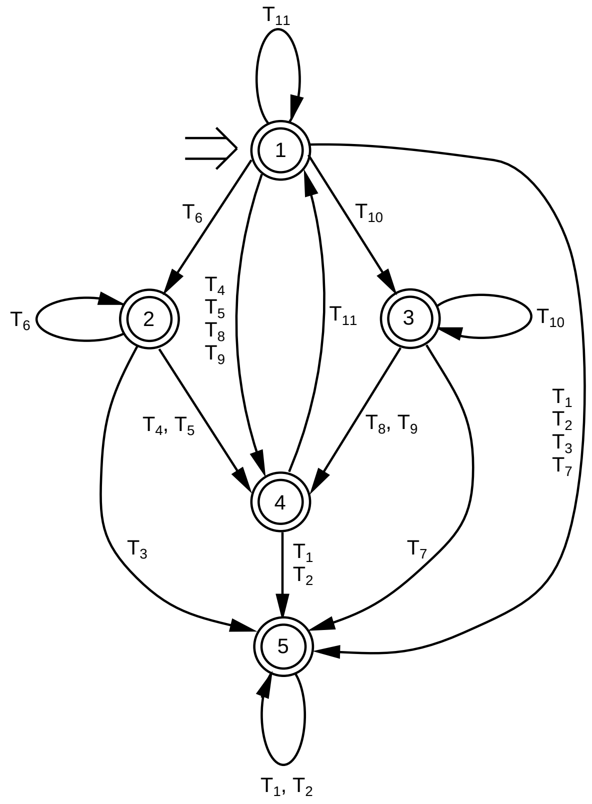

Figure 9: Automaton $\mathcal{M}^{RD}$ defining a language $(L(\mathcal{M}))^R$, where $\mathcal{M}$ is the automaton shown in Figure 8.

$$\begin{array}{lllll}
\omega: & @o(0.002,1,1) & & & \\
p_1: & @o(d,s,t): t == 1 & \rightarrow & [++f(s*0.05)--f(s*0.05)++@o(d,s*0.2,4)] & /*T_4*/ \\
 & & & [++f(s*0.05)--f(s*0.05)@o(d,s*0.2,4)] & /*T_5*/ \\
 & & & [--f(s*0.05)++f(s*0.19)--@o(d,s*0.172,4)] & /*T_8*/ \\
 & & & [--f(s*0.05)++f(s*0.19)@o(d,s*0.172,4)] & /*T_9*/ \\
 & & & [@o(diam,s/2,5)] & /*T_1*/ \\
 & & & [f(s*0.13)@o(diam,s/2,5)] & /*T_2*/ \\
 & & & [+@o(d,s*0.42,5)] & /*T_3*/ \\
 & & & [f(s*0.14)-@o(d,s*0.37,5)] & /*T_7*/ \\
 & & & [++f(s*0.078)--f(s*0.078)@o(d,s*0.74,2)] & /*T_6*/ \\
 & & & [--f(s*0.0689)++f(s*0.1053)@o(d,s*0.74,3)] & /*T_{10}*/ \\
 & & & [f(s*0.26)@o(d,s*0.74,1)] & /*T_{11}*/ \\
p_2: & @o(d,s,t): t == 2 & \rightarrow & [++f(s*0.05)--f(s*0.05)++@o(d,s*0.2,4)] & /*T_4*/ \\
 & & & [++f(s*0.05)--f(s*0.05)@o(d,s*0.2,4)] & /*T_5*/ \\
 & & & [+@o(d,s*0.42,5)] & /*T_3*/ \\
 & & & [++f(s*0.078)--f(s*0.078)@o(d,s*0.74,2)] & /*T_6*/ \\
p_3: & @o(d,s,t): t == 3 & \rightarrow & [--f(s*0.05)++f(s*0.19)--@o(d,s*0.172,4)] & /*T_8*/ \\
 & & & [--f(s*0.05)++f(s*0.19)@o(d,s*0.172,4)] & /*T_9*/ \\
 & & & [f(s*0.14)-@o(d,s*0.37,5)] & /*T_7*/ \\
 & & & [--f(s*0.0689)++f(s*0.1053)@o(d,s*0.74,3)] & /*T_{10}*/ \\
p_4: & @o(d,s,t): t == 4 & \rightarrow & [f(s*0.26)@o(d,s*0.74,1)] & /*T_{11}*/ \\
 & & & [@o(d,s/2,5)] & /*T_1*/ \\
 & & & [f(s*0.13)@o(d,s/2,5)] & /*T_2*/ \\
p_5: & @o(d,s,t): t == 5 & \rightarrow & [@o(d,s/2,5)] & /*T_1*/ \\
 & & & [f(s*0.13)@o(d,s/2,5)] & /*T_2*/
\end{array}$$

7 Conclusion

These notes present a method for constructing a parametric L-system equivalent to a given language-restricted iterated function system in which transformations are compositions of rotations, translations, and scalings.

References

[1] M. F. Barnsley. *Fractals Everywhere*. Academic Press, San Diego, 1988.

[2] M. F. Barnsley and S. Demko. Iterated function systems and the global construction of fractals. *Proceedings of the Royal Society of London Ser. A*, 399:243–275, 1985.

[3] I. N. Bronshtein and K. A. Semendyayev. *Handbook of mathematics*. Van Nostrand Reinhold Co., 1979.

[4] Karel Culik II and Simant Dube. L-systems and mutually recursive function systems. *Acta Informatica*, 1994. To appear.

[5] S. Dubuc and A. Elqortobi. Approximations of fractal sets. *Journal of Computational and Applied Mathematics*, 29:79–89, 1990.

[6] D. Hepting, P. Prusinkiewicz, and D. Saupe. Rendering methods for iterated function systems. In *Proceedings of FRACTAL '90, the First IFIP Conference on Fractals, Lisbon, Portugal*, June 1990.

[7] J. E. Hutchinson. Fractals and self-similarity. *Indiana University Journal of Mathematics*, 30(5):713–747, 1981.

[8] B. B. Mandelbrot. *The fractal geometry of nature*. W. H. Freeman, San Francisco, 1982.

[9] P. Prusinkiewicz and M. Hammel. Automata, languages, and iterated function systems. In J. C. Hart and F. K. Musgrave, editors, *Fractal Modeling in 3D Computer Graphics and Imagery*, pages 115–143. ACM SIGGRAPH, 1991. Course Notes C14.

[10] P. Prusinkiewicz and A. Lindenmayer. *The algorithmic beauty of plants*. Springer-Verlag, New York, 1990. With J. S. Hanan, F. D. Fracchia, D. R. Fowler, M. J. M. de Boer, and L. Mercer.

[11] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. Mechanisms of plant development. Video tape, University of Regina, 1988.

[12] E. R. Vrscay. Iterated function systems: Theory, applications and the inverse problem. In *Proceedings of the NATO Advanced Study Institute on Fractal Geometry held in Montreal, July 1989*. Kluwer Academic Pulishers, 1990.

[13] R. F. Williams. Composition of contractions. *Bol. Soc. Brasil. Mat.*, 2:55–59, 1971.