Geometry control of the junction between two fractal curves

Sergey Podkorytov, Christian Gentil, Dmitry Sokolov, Sandrine Lanquetin

HAL Id: hal-00755851

https://inria.hal.science/hal-00755851

Submitted on 22 Nov 2012

Geometry control of the junction between two fractal curves.

Sergey Podkorytov[1], Christian Gentil[1], Dmitry Sokolov[2] and Sandrine Lanquetin[1]

[1] *University of Burgundy, France*
[2] *University of Lorraine, France*

Abstract

The general objective of our work is to create a geometric modeller based on iterative processes. With this objective in mind, we have to provide tools that work with fractal objects in the same manner as with objects of classical topology. In this article we focus on the constructing of an intermediate curve between two other curves defined by different iterative construction processes. Similar problem often arises with subdivision surfaces, when the goal is to connect two surfaces with different subdivision masks. We start by dealing with curves, willing to later generalize our approach to surfaces. We formalize the problem with Boundary Controlled Iterated Function System model. Then we deduct the conditions that guaranties continuity of the intermediate curve. These conditions determine the structure of subdivision matrices. By studying the eigenvalues of the subdivision operators, we characterize the differential behaviour at the connection points between the curves and the intermediate one. This behaviour depends on the nature of the initial curves and coefficients of the subdivision matrices. We also suggest a method to control the differential behaviour by adding intermediate control points.

Key words: Iterative function system, attractor, curve, differentiability

1. Introduction

The global objective of our work is to develop a geometric modeller based on the paradigm of fractal geometry and more precisely on iterative processes. The iterative processes can be used to describe a wide array of shapes inaccessible to standard methods such as fractal curves or sets. Our work is based on Boundary Controlled Iterative Function System (BCIFS) (see [TBSG+06]). BCIFS upgrades the standard iterative process such as Iterated Function System (IFS) with B-Rep structure. We can describe objects with familiar B-rep structure, where each cell is a fractal object. For instance, if we consider a polyhedron, then each face is a fractal surface, and each edge is a fractal curve. Objects modelled with BCIFS not necessary have the fractal properties, objects such as B-splines curves and surfaces can be modelled as well. So with BCIFS formalism we can operate with both standard and fractal objects.

Our formalism covers traditional models like NURBS and subdivision surfaces and also a new world of shapes, not accessible by the polynomial models and having a particular aesthetic. With our model we can control independently the lacunarity (topology subdivision) and the roughness (geometry subdivision) (see figure 1). We want to provide tools to design new kind of shapes and to construct them

for industry applications in order to exploit specific physical fractal properties. For instance we can exploit lacunarity to have weightless objects (see figure 2), or roughness to have specific design and more rigid surface as illustrated by figure 3.
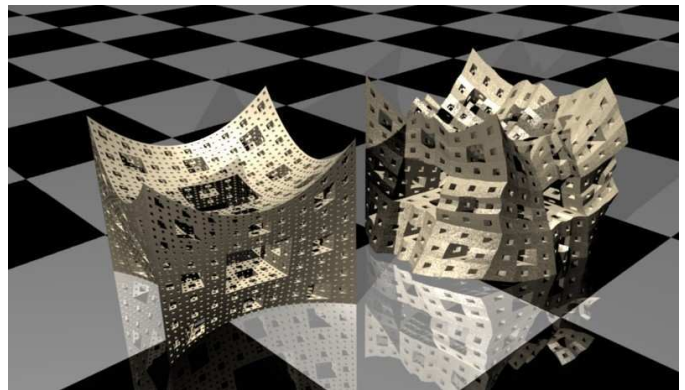

Figure 1. Left: the Menger's sponge with polynomial deformation. Right: the Menger's sponge with fractal deformation.

One difficulty arising is how to integrate those kind of objects in an existing environment designed with standard smooth geometry.

In this article we focus on the problem of connecting two shapes built by two different iterative processes. This
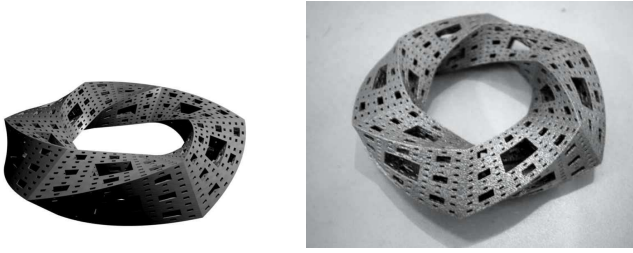
Figure 2. Left: a geometric model of a "Möbius-Menger torus" design by LE2I. Right: The corresponding "Möbius-Menger torus" constructed by Pôle Europeen de Plasturgie using selective laser melting process.
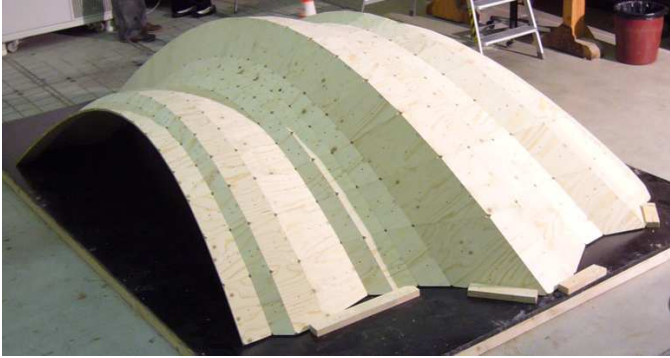

Figure 3. Architectural overlapping structures constructed by IBOIS of EPLF. A wooden model of a surface, smooth in one direction and fractal in the second direction with a low iterated level. The fractal structure increase the rigidity of the surface.

problem often arises in context of subdivision curves and surfaces [LL03]. Here we describe the general way to deal with the curves, that gives a continuous intermediate curve. We also study its differential properties. We use the formalism of Boundary Controlled Iterated Function System [TBSG+06] to describe two curves and construct the intermediate curve at the same time. An automaton describes the construction process. Adjacency and incidence equations guaranty the continuity and define the structure of the subdivision matrices.

In section 2, we introduce the reader to the concepts of iterative modelling like Iterative Function System (IFS), Controlled Iterative Function Systems (CIFS) and Boundary Controlled Iterated Function System (BCIFS). In that section we also describe concepts of the adjacency and incidence constraints. In section 3, we construct a BCIFS that will satisfy our needs. We analyse the differential properties of the obtained solution in section 4. After that, we refine our solution for more convenience in section 5. Finally we provide some examples of curves we were able to generate in section 6.

2. Background on IFS, CIFS and BCIFS

2.1. *IFS and projected IFS*

Given a complete metric space (E, d) an Iterated Function System (*IFS*) is a finite set of contractive operators

$\mathbb{T} = \{T_i\}_{i=0}^{N-1}$ acting on points of E.

It is possible to define an operator $\mathbb{T} : \mathcal{H}(E) \to \mathcal{H}(E)$, called Hutchinson operator. It maps non-empty compact subset K to $\bigcup_{i=0}^{N-1} T_i(K)$.

If the operators T_i are contracting in the space (E, d), then the operator $\mathbb{T}$ is contracting in $(\mathcal{H}(E), d_{\mathcal{H}})$, the space of non-empty compact subsets in E with the Hausdorff distance. [Bar88]

According to [Hut81], there exists a unique compact A such that $\mathbb{T}(A) = A$, i.e. the fixed point of $\mathbb{T}$. Furthermore, due to the contractivity of $\mathbb{T}$, A can be calculated as the limit: $A = \lim_{i \to \infty} \mathbb{T}^i(K)$. This limit does not depend on the initial compact K as long as it is not empty. This property is illustrated by the sequence of images in figure 5 (top).

The idea of projective IFS was introduced by Zair and Tosan [ZT96]. By separating the iterative space from the modelling space, it is possible to construct fractal shapes with control points. Similar to splines determined by the basic functions defined in a barycentric space, attractors are defined in barycentric space whose dimensions correspond to the number of control points: $A \subset BI^n = \{\lambda \in \mathbb{R}^n | \sum_{i=0}^{n-1} \lambda_i = 1\}$, where n is the number of control points. Then the attractor is projected to the modelling space with the transformation defined by control points $PA = \{\sum_{i=0}^{n-1} P_i \lambda_i | \lambda_i \in A\}$, where $P = [P_0 \ P_1 \cdots P_{n-1}]$ is the vector composed of control points.

We limit our study to linear operator acting in the barycentric spaces. Let $\mathbb{R}^n$ be a vector space, then the barycentric hyperplane is $BI^{n-1} = \{\boldsymbol{v} | \sum_{i=1}^{n} v_i = 1\}$. The operators on the barycentric space can be written as linear operators on $\mathbb{R}^n$, with a specific constraint on its matrix representation. That is each column of the matrix must have the sum of its elements equal to 1. The fixed point of such operator, as well as an attractor of the IFS composed of such operators, always belong to the barycentric plane. Later in the article we construct a special vector space associated with the barycentric space for the purpose of studying the differential behaviour (see section 4).

2.2. *Controlled IFS (CIFS)*

It is possible to extend this model by adding rules controlling the iterative process. This is the principle of the Controlled IFS (CIFS). It can be described with an automaton [MW88].

States of the automaton are associated with iterative spaces and arcs represent transformation applicable at the current state. This gives a new way to control the shape of the attractor.

The left scheme in figure 4 represents the automaton for the classic IFS which generates the BARNSLEY fern. The fern is self-similar, i.e. it is built from an infinite number of copies of itself.

We can add a new state with three transformations defining the SIERPINSKI triangle. We also change the destination
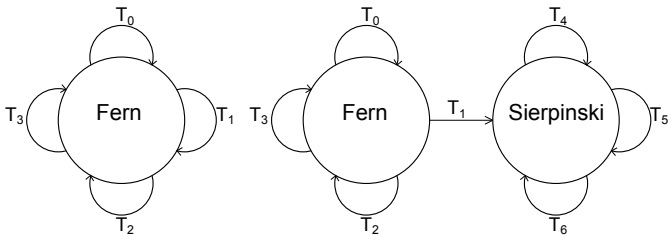
Figure 4. The automatons giving the transformation applying rules. Left: simple automaton corresponding to the ordinary IFS. Right: Modified automaton with two state. The attractor of the state Fern include the attractor of the state Sierpinski.

of the arc T_1 (see right part in figure 4). After the transformation T_1 is applied once, next steps will then follow the SIERPINSKI subdivision. The attractor of the new controlled IFS is composed of the infinite number of an SIERPINSKI triangles (see bottom row in figure 5).
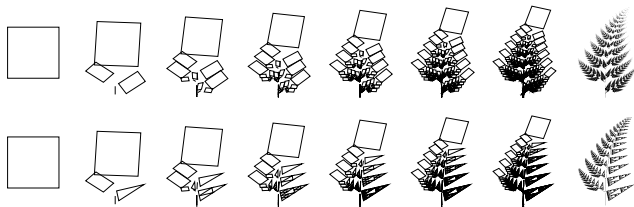

Figure 5. Top: BARNSLEY fern; Bottom: with a C-IFS we can mix attractors of different nature. The Barnsley fern is self-similar, it is built of an infinite number of copies of itself, while the fern on the bottom, is built of infinite number of SIERPINSKI triangles.

2.3. Boundary Controlled IFS (BCIFS)

The Boundary Controlled IFS (BCIFS) model enhances the CIFS model by adding the B-Rep notion [TBSG⁺06]. It gives a way to explicitly state the face-edge-vertex structure of the attractor [Gen92]. We can also write the incidence and adjacency constrains on the subdivision process and thus control the topology: classic (curve, surface, ...) or fractal topology.

B-Rep concepts used here are more general than the classical B-Rep concepts. Topological elements can be fractal objects. For example, a face can be a SIERPINSKI triangle, or an edge can be a CANTOR set, but the B-Rep structure remains consistent. This approach differs from the traditional model by the ability to clarify the relation of incidence and adjacency with the subdivision process of the given topological structure. For the sake of simplicity we will present its application to curves.

To describe a curve with BCIFS it is necessary to distinguish the different spaces in which different cells will be defined. Each cell of the B-Rep structure (here an edge or a vertex) is defined by:
– A state which represents this cell in the automaton
– An iterative space associated with it, more specifically a barycentric space. The dimension of this space is equal to the number of control points of the cell

– An IFS reflecting the subdivision of the cell
For the curve defined by n control points and whose vertices depend on p control points we obtain the following structure:
– For the edge:
  · a state called e
  · an iterative space = a barycentric space of dimension n
  · an IFS = a set of at least two matrices ($n \times n$) representing edge subdivision
– For each vertex (which can be different):
  · two states v^l and v^r (for the left and right vertices respectively)
  · an iterative space = a barycentric space of dimension p
  · an IFS = a set of one matrix ($p \times p$) representing vertex subdivision

At this point we have a set of iterated function systems, where each IFS describes a cell of the B-Rep structure. If the IFS are composed of arbitrary operators there is no guarantee that the edge is really bordered by vertices and that subdivision of the edge does result in continuous curve. To address this issue we will use additional constraints on BCIFS matrices, but before that we need to add a relation between different cells.

To do so we introduce boundary operators. In our example, different IFS associated with the edge and the vertices are defined in the barycentric spaces. Boundary operators create a link between the space defining the system of the nested subspaces, i.e. the space for the vertex is a subspace of the space for the edge. As an edge has two vertices we use two boundary operators.

For example, consider a curve that depends on three control points. If the first vertex depends on first two control points and while the second one depends on second and third control points, then their respective boundary operators are:

$$\partial_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}, \partial_1 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The general automaton for the curve with an edge subdivided in two parts and two different vertex subdivisions is presented in figure 6.
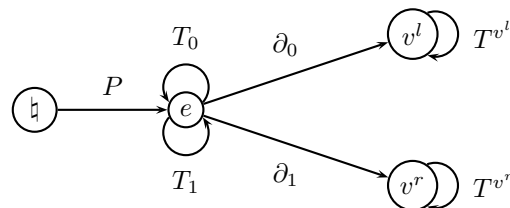

Figure 6. An automaton representing a curve with two edge subdivision $T_{0,1}$ and two different vertex subdivisions T^{v^l}, T^{v^r}.

3

Incidence and adjacency constraints can be easily identified from the graph representing the progression of the automaton. The first level of subdivision is depicted in figure 7.
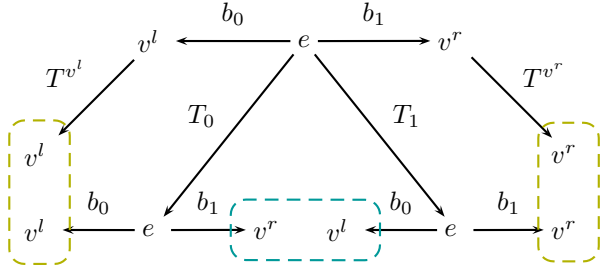

Figure 7. The unfolding of the automaton generating subdivision system. This system is built in such way, that it guaranties the topological structure. The incidence constraints are represented in red and adjacency constraints in blue.

Adjacency constraints The edge is subdivided in two parts, so the "left" part has to be connected to the "right" one through the intermediate vertices (see figure 7). First of all the states v^l and v^r have to be identical, or in other world the "left" and the "right" vertices need to have the same "nature" and be subdivided by the same operator, otherwise the continuity is not ensured. So equality $T^{v^l} = T^{v^r} = T^v$ is necessary.

Another condition is deduced by writing the equivalence of paths in the graph: the left vertex of the right subdivision has to correspond to the right vertex of the left subdivision:

$$T_0 \partial_1 = T_1 \partial_0.$$

Incidence constraints In the same manner, incidence constraints express the fact that vertices must remain at the ends of the edge during the subdivision process: subdivision of the left vertex of the edge has to correspond to the left vertex of the left subdivision of the edge (see figure 7). Writing down equivalences between paths gives the following equations:

$$\partial_0 T^v = T_0 \partial_0,$$
$$\partial_1 T^v = T_1 \partial_1.$$

Resolving the constraints, adjacency and incidence determine structures in the subdivision matrices in the form of equalities between columns and sub-matrices (see figure 8). Two examples of curves that can be described by a BCIFS are presented on the picture 9.

3. Constructing the junction.

In this section we are going to use the BCIFS formalism to describe the set of two curves and construct a third curve between them. We start with a BCIFS for two disjoint curves (see figure 10), wishing to build a joining curve
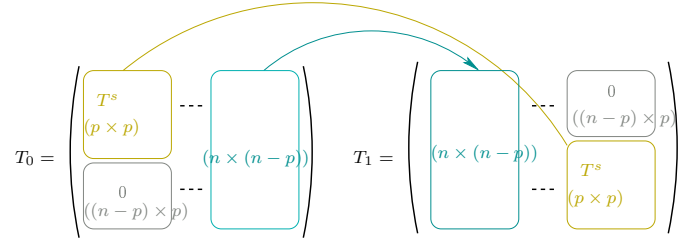

Figure 8. General structure of the subdivision matrices for the curve with n control points, whose vertices are controlled by p control points.

Figure 9. Left: cubic spline with 4 control points. Right: a fractal curve with 3 control points.

between the right endpoint of the left curve and the left endpoint of the right curve. The initial BCIFS consists of 5 states: state corresponding to the modelling space ♮, two states corresponding respectively to the initial curves l and r and two states for their respective vertices v^l and v^r. P^l and P^r are the vectors of control points for the initial
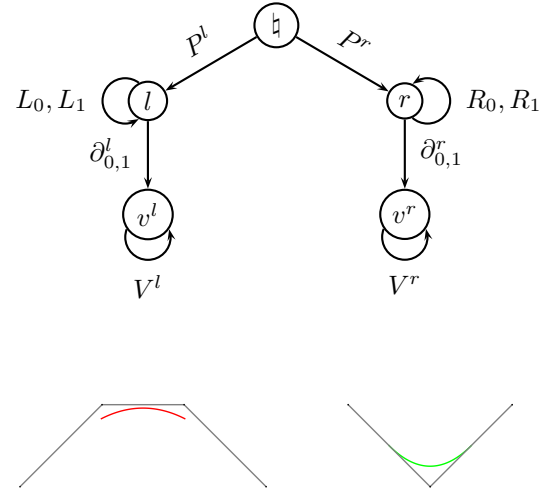

Figure 10. Top: initial BCIFS. Each branch of the automaton corresponds to an initial curve. Bottom: Possible attractor of that BCIFS: two B-spline of different degrees.

curves. L^0, L^1 and R^0, R^1 are the subdivision operators for respective curves and $b^l_{0,1}$ and $b^r_{0,1}$ are the boundary operators. We will denote the barycentric space for any state s as BI^s and the dimensions of BI^{v^l} and BI^{v^l} as n_l and n_r respectively.

Now for the purpose of building a joining curve we add an intermediate state i and three new transitions T^l, T^r, P^i as well as vertex states v^{il}, v^{ir} and their respective boundary operators $\partial^i_0, \partial^i_1$ (see figure 11). At this point our intermediate curve is not yet a curve, but two disjoint images of
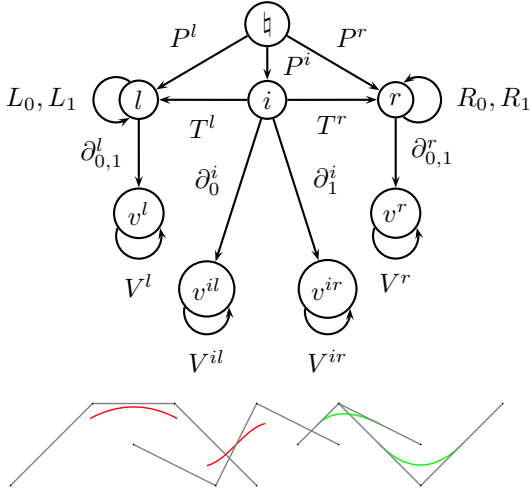
Figure 11. Top: modified BCIFS, with an intermediate state i and additional transitions leading to and from it. Bottom: The attractor of this BCIFS: two initial curves and two their copies.

the initial curves placed somewhere in the modelling space (by applying $T^l P^i$ and $T^r P^i$ respectively), but we will introduce conditions on transitions T^l, T^i and P^i so that new curves become a continuation of our initial curves. To do that we need to determine the dimension of BI^i (i.e. how many control points the intermediate curve depends on) and therefore the dimensions of T^l and T^r. We know that the endpoints of the initial curves respectively depends on n_l and n_r control points. As we seek the way to connect those points with our intermediate curve, it has to be controlled by at least $n_l + n_r$ control points.

The necessity to join the intermediate curve to the initial curves means that the left vertex of the intermediate curve must be equal to the right vertex of the left curve. It provides the following equations:

$$v^l = v^{il}$$

$$V^l = V^{il}$$

Similar equations have to be held true for the right curve:

$$v^r = v^{ir}$$

$$V^r = V^{ir}$$

To derive the other conditions we use the graph depicting unfolding of the automaton. It is presented in figure 12. Note that states v^{il} is merged with v^l as well as v^{ir} is merged with v^r. The constrains for the left part of the curve can be written as follows:

$$P^i \partial_0^i = P^l \partial_1^l \tag{1}$$

$$T^l \partial_0^l = \partial_0^i V^l \tag{2}$$

$$\partial_1^l V^l = L_1 \partial_1^l \tag{3}$$

Analogous conditions can be written down for the right part:

$$P^i \partial_1^i = P^r \partial_0^r \tag{4}$$
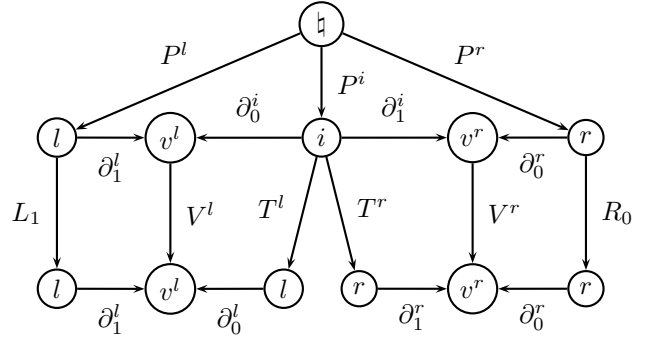
$$T^r \partial_1^r = \partial_1^i V^r \tag{5}$$

Figure 12. Unfolding of the automaton from figure 11.

$$\partial_0^r V^r = R_0 \partial_0^r \tag{6}$$

The equations (2),(3),(5) and (6) fixate n_l left columns for T^l and n_r right columns for T^r:

$$T^L = \begin{pmatrix} V^l & \mathbf{X} \\ 0 & \end{pmatrix},$$

$$T^R = \begin{pmatrix} \mathbf{Y} & 0 \\ & V^r \end{pmatrix}.$$

Here V^l and V^r are blocks equal to the respective vertex subdivisions, while X and Y consists of arbitrary columns, such that the sum of all elements for each column is equals to 1.
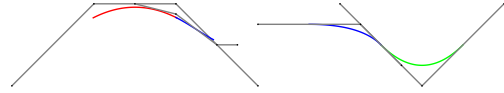

Figure 13. Possible attractor of the automaton from figure 11 with adjacency and incidence constraints in place. Two initial curves and two segment of the new intermediate curve are pairwise connected.

At this point BCIFS presented in figure 11 plus the aforementioned conditions still yields us a disjoint curve (see figure 3). But if we keep the X and Y blocks of T^l and T^r within certain bounds, we can ensure that the gap between the curves is smaller than the gap between the initial two. These bounds are derived after we introduce another transition to our automaton.

We add one more transition, applying the approach described before to the newly obtained curves. The new transformation acts on the image of the initial curves in BI^i and its effect is similar to what we achieved before by adding T^l, T^r, P^i. This transformation leads from the state i to itself and is denoted as T^i. The modified BCIFS is presented in figure 14. To obtain the constraints on T^i we use the graph depicting the unfolding of the new automaton (see figure 15.) And then we can deduce the following constrains on T^i:

$$T^r \partial_0^r = T^i \partial_1^i \tag{7}$$

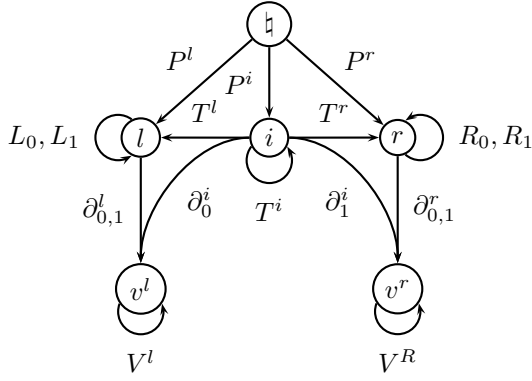$$T^l \partial_1^l = T^I \partial_0^i \tag{8}$$

Figure 14. Final BCIFS with an intermediate state and all required transitions.

As we already know the form of T^l, T^r, ∂_0^i and ∂_1^i, T^i can be described like this: it has n_l first columns equal to the last n_l columns of T^l and the n_r last columns equal to the first n_r columns of T^r.
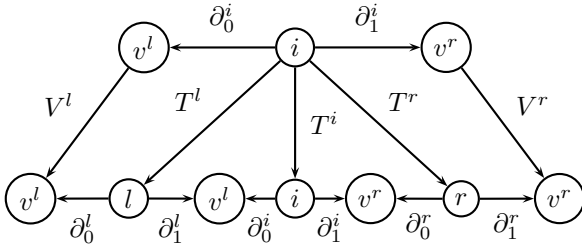

Figure 15. Unfolding of the automaton from figure 14.

Here we would like to give a brief and informal description of how this automaton works. Imagine that we have already built the initial curves inside their respective barycentric spaces. At first automaton places their "copy" into the barycentric space BI^i by applying T^l or T^r respectively. Then the automaton applies T^i to these two "copies", moving them closer to the fixed point of T^i. In the meantime the automaton uses T^l and T^r and "produces" a new pair of curves. Condition guarantee that the new curves are pairwise connected to the previous ones. As this process repeats the curves "grow" towards each other. As T^i is a contracting map, this process has a limit that is a continuous curve, that passes through the fixed point of T^i.

Here, having constructed the BCIFS for the intermediate curve we proceed to the next section, where we analyse the continuity of the curve it describes.

4. Continuity of degree 1 or higher

We will denote the two initial curves, each inside their respective barycentric space by C^l and C^r respectively. Now let us consider the intermediate curve in barycentric space BI^i. The curve can be divided into two parts by the fixed point of T^i. We denote the left one by C^{il} and the other one by C^{ir}. Each of those parts are built from copies of one of the initial curves. This means that if initial curve has some order of continuity, then all of its copies inherits it:

$$C^{il} = \bigcup_{n=0}^{\infty} (T^i)^n T^l C^l.$$

The nature of continuity between the duplicates of the initial curve depends on the eigenvector associated to the sub-dominant eigenvalue (see [SGB10]). If this eigenvector belongs to the subspace associated with the vertex, then the continuity is at least C_1 due to the presence of blocks V^l and V^r of the matrices T^l and T^r respectively.

Note that the initial left and right curves are projected to the space associated with the state i by the transformations T^l and T^r. From this point of view we can treat matrices T^l and T^r as vectors of control points in the space BI^i. The transformation T^i iterates this process creating copies of the curves inside the barycentric space. Constraints on T^i guaranties the C^0-continuity. And after an infinite number of iterations two curves are joined at the fixed point of T^i :

$$C^i = C^{il} \cup C^{ir} = \bigcup_{n=0}^{\infty} (T^i)^n (T^l(C^l) \cup T^r(C^r))$$

To study the differential behaviour of C^{il} we consider the control polygon P_0^l corresponding to the matrix T^l. By applying T_i to P_0^l we obtain a control polygon P_1^l for the next copy of C^l. Thus we can construct an infinite sequence of control polygons $\{P_k^l\}_{k=0}^{\infty}$. The lengths of the segments of the polygon P_k^l tend to 0 when k tends to ∞, but each element converges with different speeds in different direction. If the speed in one direction is considerably slower than in the others, than the control polygon becomes more and more stretched in the direction with each iteration. To study this behaviour we utilize the eigenvalues and eigenvectors of T^i.

Let us consider the eigenpairs (eigenvalue and associated eigenvector) of T^i: first eigenpair is $(1, \boldsymbol{v_1})$ where $\sum_{i=1}^n v_{1i} \neq 0$; while the rest are $(\lambda_j, \boldsymbol{v_j})$ where $|\lambda_j| < 1$ and $\sum_{i=1}^n v_{ji} = 0$ to assume the convergence. Without the loss of generality we can consider $\sum_{i=1}^n v_{1i} = 1$. As $T^i \boldsymbol{v_1} = \boldsymbol{v_1}$, $\boldsymbol{v_1}$ is a fixed point of T^i. Also note that because $\sum_{i=1}^n v_{1i} \neq 0$ all eigenvectors, except $\boldsymbol{v_1}$, are parallel to the barycentric space.

On the assumption of T^i have a full set of real eigenvectors we can express any vector on the barycentric hyperplane as a sum of $\boldsymbol{v_i}$:

$$BI^i = \{\boldsymbol{v} | \boldsymbol{v} = \boldsymbol{v_1} + \sum_{i=1}^n c_i \boldsymbol{v_i}, c_i \in \mathbb{R}\}.$$

Now we consider a vector space associated with the barycentric plane with a basis of $\{\boldsymbol{v_2}, \ldots, \boldsymbol{v_n}\}$. We associate the operator T^i to an operator that acts in this vector space. The matrix form of this operator is a diagonal matrix with $\lambda_2, \ldots, \lambda_n$ on the diagonal.

Now let us assume that $\lambda_2 \in \mathbb{R}$, $\lambda_2 > 0$ and $\forall i > 2, |\lambda_2| > |\lambda_i|$. When such a condition holds true, the successive con-

trol polygons P_k^l will be more stretched in the direction of $\boldsymbol{v_2}$ while approaching the fixed point.

As $|\lambda_2| < \lambda_1 = 1$, the differential behaviour depends on the nature of the sub-dominant eigenvalue: real positive eigenvalue, negative real eigenvalue or complex eigenvalue (see [Ben09,BGN09,SGB12]).

For instance if all vectors of P_0^l have a non-zero component in the direction of $\boldsymbol{v_2}$, $\boldsymbol{v_2}$ gives the direction of the left half-tangent. So the existence and the position of the left half-tangent depend on the existence of single positive sub-dominant eigenvalue of T^i and whether initial control polygon does not have a zero component in the sub-dominant eigenvector component.

If the sub-dominant eigenvalue is negative or complex there is no tangent. However if all vectors of the control polygons do have a zero component in the sub-dominant direction, we have to look at the next eigenvalue. If the absolute values of the next eigenvalue is strictly greater than the absolute value of the other eigenvalues and it is positive and if a control polygon has a member with a non-zero component, then that eigenvector gives the direction of the left half-tangent. This logic can be repeated in case there is no non-zero components in the control polygon.

Same conditions can be applied for the right part. Note that despite eigenvectors and eigenvalues are the same, the zero-component may exist only for the left control polygons, but not for the right one or vice versa. The reason for this is the difference between two control polygons.

4.1. Example

Here we give an example of joining a cubic and a quadratic splines with the proposed method. In this case the matrices T^i, T^r and T^l have the following form:

$$T^l = \begin{pmatrix} 0.5 & 0.125 & 0 & a \\ 0.5 & 0.75 & 0.5 & b \\ 0 & 0.125 & 0.5 & c \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & e \end{pmatrix}, \; T^r = \begin{pmatrix} k & 0 & 0 \\ l & 0 & 0 \\ m & 0 & 0 \\ n & 0.75 & 0.25 \\ p & 0.25 & 0.75 \end{pmatrix},$$

$$T^i = \begin{pmatrix} 0.125 & 0 & a & k & 0 \\ 0.75 & 0.5 & b & l & 0 \\ 0.125 & 0.5 & c & m & 0 \\ 0 & 0 & d & n & 0.75 \\ 0 & 0 & e & p & 0.25 \end{pmatrix},$$

where $a + b + c + d + e = 1$ and $k + l + m + n + p = 1$.

With the change of free variable we can obtain a different sub-dominant eigenvalue and therefore different behavior of the curve. Some examples are presented in figure 16.

It is worth noting that T^i has 8 degrees of freedom, and all of them affect eigenvalues of the matrix, and therefore af-
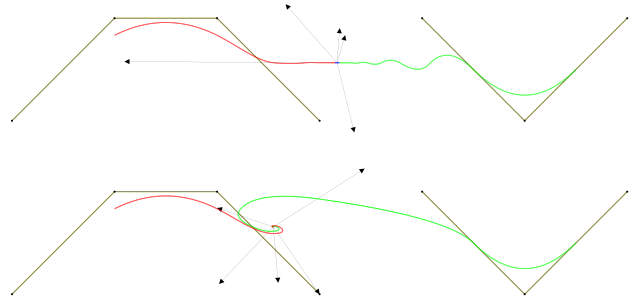

Figure 16. Example 1. Different ways to connect a cubic and a quadratic spline. Top: Positive sub-dominant eigenvalue yields G_1-continuity. $a = 0.131$, $b = 0.2$, $c = 0.38$, $d = 0.289$, $k = 0.11$, $l = 0.12$, $m = 0.1$, $n = 0.12$, $\lambda_2 = 0.696$ Bottom: Sub-dominant eigenvalue is complex and both parts of the curve are spiralling towards the fixed point. $a = 0.131$, $b = 0.745$, $c = -0.061$, $d = 0.289$, $k = 0.11$, $b = 0.097$, $c = 0.115$, $d = -0.53$, $\lambda_{2.3} = 0.112 \pm 0.449i$

fects the shape and differential properties of the final curve. In the next section we refine our automaton to avoid this overflowing with free variables.

5. Refining the BCIFS

While the method described above solves the problem, it is not very convenient in practical usage. Although free parameters give a flexible control of the curve, the influence of each separate parameter can not be singled out and described. Also establishing a parameter domain that guaranties smooth curve is difficult as it requires symbolic evaluation of eigenvectors and eigenvalues. In the following section we describe modifications to the automaton as well as additional constraints that eliminates this flaw.

In the previous section we showed that the sub-dominant eigenvector gives the tangent direction. Here we are going to change the automaton, so it would be possible to control the sub-dominant eigenvector directly with control points.

As we already noted, the intermediate curve can be split into two parts, we are going to reflect this in an automaton. We replace the state i with two states il and ir. New states have on common vertex denoted as v^i. The modified automaton is presented in figure 17.
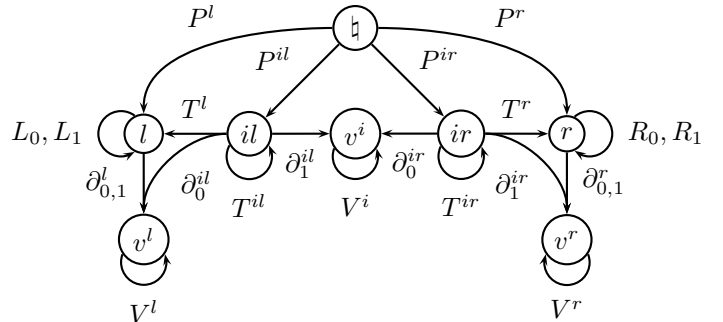

Figure 17. Refined BCIFS with two intermediate states.

Here we are free to choose any dimension for BI^{v^i}. For the sake of simplicity we choose it equal to 1. There is only

one 1×1 matrix possible in an IFS — an identity matrix. An identity matrix as a vertex subdivision essentially means that the vertex will be located exactly at the control point position. It gives us a direct control of the curve, as the curve always passes through the vertex and therefore through the control point.

Now we can repeat the process from section 3 and deduce the form of T^{il}, T^{ir}, T^l and T^r. We illustrate it with the cubic-quadratic splines from section 4.1. Let us first consider transformations T^l and $T^i l$. The left curve is a cubic spline. There are 3 control points for the vertex of the initial curve and 1 for the intermediate vertex. To better control the curve we can add another control point. We will later use it to explicitly control the half-tangent at the intermediate vertex. This means that the final dimension of BI^{il} is $3 + 1 + 1 = 5$.

We can write down the constraints and fix the form of T^{il}:

$$\begin{pmatrix} 0.125 & 0 & a & f & 0 \\ 0.75 & 0.5 & b & g & 0 \\ 0.125 & 0.5 & c & h & 0 \\ 0 & 0 & d & j & 0 \\ 0 & 0 & e & k & 1 \end{pmatrix},$$

where $a + b + c + d + e = 1$ and $f + g + h + j + k = 1$.

Now we would like to set the tangent at the intermediate vertex in the direction of the segment between the two rightmost control points. To achieve this a sub-dominant eigenvector must have the given direction. For this we assign values to the parameters of T^{il}. Also we would like to simplify the dependency between parameters and eigenvalues. So we assign 0 to a, b, c, e, f, g, h and 1 to d and obtain the following structure:

$$\begin{pmatrix} 0.125 & 0 & 0 & 0 & 0 \\ 0.75 & 0.5 & 0 & 0 & 0 \\ 0.125 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 1 & j & 0 \\ 0 & 0 & 0 & 1-j & 1 \end{pmatrix}$$

The idea behind this is to create a 2×2 sub-matrix in the bottom left corner and isolate the free variables in there. This means that free variables only affect eigenvalues of the sub-matrix. Sub-matrix $\begin{pmatrix} j & 0 \\ 1-j & 1 \end{pmatrix}$ has two eigenvalues 1 and j. These eigenvalue are also eigenvalues of T^{IL}. The eigenspace corresponding to the eigenvalue j is $(t, -t)$, where $t \in \mathbb{R}$. So the original 5×5 matrix has the following eigenspace: $(0, 0, 0, t, -t)$. Note that $\forall t \in \mathbb{R}$ a corresponding eigenvector is collinear to $\boldsymbol{e_4} - \boldsymbol{e_5}$ (where $\boldsymbol{e_i}$ is a component of the canonical basis) which is a desired tangent direction. Now we only have to ensure that j is the sub-dominant eigenvalue. Because all other eigenvalues are constant it is

easy to establish a domain for j. In the provided example $j \in (0.5, 1)$.

We can now apply the same ideas to the right half of the curve and obtain the following:

$$T^{ir} = \begin{pmatrix} 1 & 1-r & 0 & 0 \\ 0 & r & 1 & 0 \\ 0 & 0 & 0 & 0.75 \\ 0 & 0 & 0 & 0.25 \end{pmatrix}.$$

It is easy to deduct a domain for r which turn out to be $(0.25, 1)$.

We have built an intermediate curve with 3 control points. It passes through one of them, and we can use the other two to control the half-tangents at the first control point. As the curve has the same nature as before i.e. it is built of infinite numbers of copies of the initial curves, differential behaviour analysis from section 4 can be applied here as well. The only exception is that we can easily write down the conditions for the C^1 continuity at the middle point of the curve. Eigenvalues l and r have to be subdominant for their respective matrices T^{il} and T^{ir}, while the three control points have to be aligned on the same line.
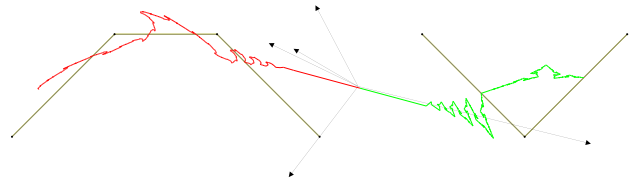
6. Examples

Figure 18. $\lambda_2 > 0$ yields G_1-continuity for the curve between two fractal curves.

In this section we give some examples of possible differential behaviours.

6.1. Positive eigenvalue, $\lambda_2 > 0$

If λ_2 is positive the half-tangents on right and on the left exist and have the same direction. We can obtain the G_1-continuity at the joining point (see figure 18).

6.2. Negative eigenvalue, $\lambda_2 < 0$

When λ_2 is negative each iteration mirrors both left and right parts along the next eigenvalue. There is no half tangent neither on the left nor on the right.

The oscillation amplitudes depend on the control polygons and the direction of the subsubdominant eigenvector $\boldsymbol{v_2}$. They can be different on the left and on the right (see figure 19).
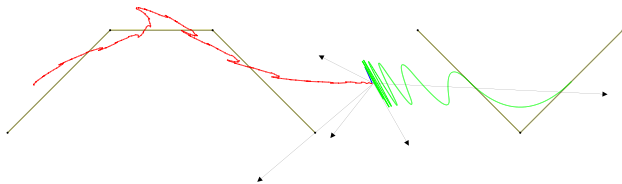
Figure 19. $\lambda_2 < 0$ with a fractal curve and a quadratic spline: there is no half-tangents.

6.3. *Complex eigenvalue, λ_2*

If the sub-dominant eigenvalue is complex both left and right parts of the curves spiral towards the fixed point on the plane defined by two associated eigenvectors (see figure 20).
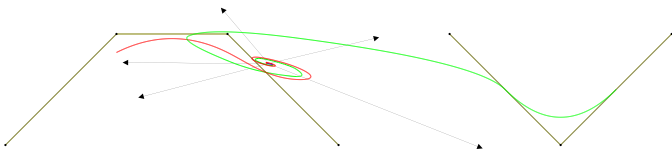

Figure 20. If λ_2 is complex both curves spiral towards the fixed point.

6.4. *Controlling half-tangent with control points*

Here are some examples demonstrating curves obtained with the modified automaton from section 5. Examples illustrates an intermediate curve between the fractal curve and a B-spline (figures 21, 22). The control points control the shape of the curve. The curve passes through the second control points, while half-tangent are defined by direction given by the first and third control point is relation to the second one. The first two examples illustrates curves with collinear half-tangent as all three control points are aligned. (figure 21). The third example illustrates the same curve, but with different control point configuration (figure 22). As control point are not aligned, they define two non-collinear half-tangents.
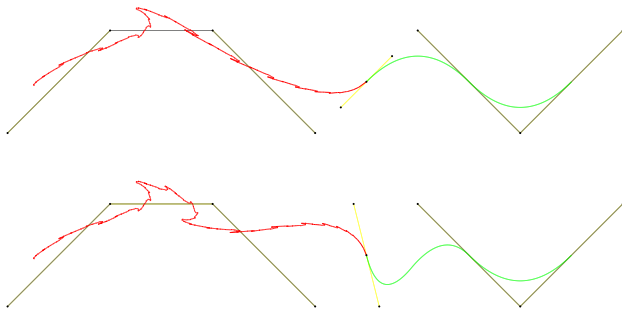

Figure 21. Two intermediate curves between the fractal curve and B-spline. Three control point are used to control the shape of the curve.
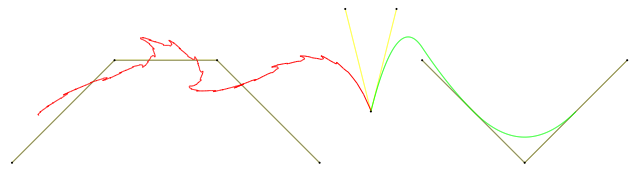

Figure 22. Intermediate curve between the fractal curve and B-spline. Such configuration of control points defines different half-tangents.

7. Conclusion

Using the BCIFS model we have shown how it is possible to construct the joining curve between two curves defined by two different iterative processes. With the automaton describing the subdivision process we wrote down the incidence and adjacency equations ensuring the C_0-continuity. Resolving them induced a structure on the subdivision matrices. We have analysed the differential properties of the final curve. We also describe the different behaviours depending on the eigenvectors and eigenvalues of the subdivision matrices.

Then we have refined the automaton to provide more control over the shape of the curve. We used the BCIFS formalism to describe an extra vertex for the curve. We used the eigenvector analysis to deduce constraint on the matrix form of the operators. Such constraints bound the sub-dominant eigenvector to the direction given by control points. This allowed us to provide convenient tools of controlling the curve behaviour.

For the future work, we aim at solving the similar problem for surfaces. According to the BCIFS formalism it is possible to use the similar construction process for surfaces and to apply the same spectral analysis to describe the differential behaviour. Indeed, we used the similar principle to derive the necessary conditions for the continuity of the intermediate surfaces (see figure 23). However the differential analysis is not yet complete. We are also trying to find out the best way to control the shape of the surface, which will be both intuitive and flexible.
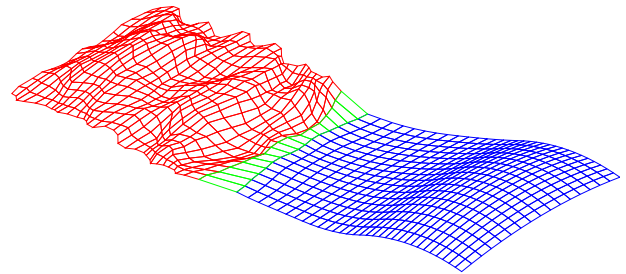

Figure 23. An example of the intermediate surface between two self-similar surfaces.

References

[Bar88]   Michael Barnsley. *Fractals everywhere*. Academic Press Professional, Inc., San Diego, CA, USA, 1988.

[Ben09]   Hicham Bensoudane. *Etude differentielle des formes fractales*. PhD thesis, Université de Bourgogne, 2009.

[BGN09]   Hicham Bensoudane, Christian Gentil, and Marc Neveu. Fractional half-tangent of a curve described by iterated function systems. *Journal of Applied Functional Analysis (JAFA)*, 4(2):311–326, April 2009.

[Gen92]   Christian Gentil. *Les fractales en synthèse d'image : le modèle IFS*. PhD thesis, Université LYON 1, 1992.

[Hut81]   John Hutchinson. Fractals and self-similarity. *Indiana University Journal of Mathematics*, 30(5):713–747, 1981.

[LL03]   Adi Levin and David Levin. Analysis of quasi-uniform subdivision. *Applied and Computational Harmonic Analysis*, 15(1):18 – 32, 2003.

[MW88]   R. Daniel Mauldin and S. C. Williams. Hausdorff dimension in graph directed constructions. *Transactions of the American Mathematical Society*, 309(2):811–829, 1988.

[SGB10]   Dmity Sokolov, Christian Gentil, and Hicham Bensoudane. Proprietes differentielles de courbes construites par procede iteratif. In *GTMG 2010, Journées du Groupe de Travail en Modélisation Géométrique, Dijon*, pages –, 31 Mars - 1er Avril 2010.

[SGB12]   Dmitry Sokolov, Christian Gentil, and Hicham Bensoudane. Differential behaviour of iteratively generated curves. In Jean-Daniel Boissonnat, Patrick Chenin, Albert Cohen, Christian Gout, Tom Lyche, Marie-Laurence Mazure, and Larry Schumaker, editors, *Curves and Surfaces*, volume 6920 of *Lecture Notes in Computer Science*, pages 663–680. Springer Berlin / Heidelberg, 2012.

[TBSG+06] E. Tosan, I. Bailly-Sallins, G. Gouaty, I. Stotz, P. Buser, and Y. Weinand. Une modélisation géométrique itérative basée sur les automates. In *GTMG 2006, Journées du Groupe de Travail en Modélisation Géométrique, Cachan*, pages 155–169, 22-23 Mars 2006.

[ZT96]   Chems Eddine Zair and Eric Tosan. Fractal modeling using free form techniques. *Comput. Graph. Forum*, 15(3):269–278, 1996.