Iterated Function Systems
A Direct Discrete Approach with Pyramids

Technical Report 13

10:20 — July 26, 1995

Contents

1 Introduction

Results of Barnsley [1, 3, 4] have shown that by the use of iterated function systems (in short IFS) image compression rates up to 1:10,000 and higher can be achieved. This is a significant improvement compared with traditional methods that provide rates up to 1:20. The main problem, also known as the *inverse problem*, is that of finding an IFS that generates a given image or an approximation. A method solving the inverse problem in general and automatically has not been published yet, although such a system is currently sold by Iterated Systems Inc. with which Barnsley is affiliated.

Recent papers by Stark [11, 12] have brought up a new way of generating images defined by IFS with neural networks: Using a unit per pixel the image as well as the gray levels can be computed sufficiently exact. The connections between units are determined by the IFS' transformations.

First of all, we improved these results by giving a closed form of representation and by using image pyramids to recognize empty regions of the image at an early stage of computation. Referring to Stark [11] we use the term *direct computation* for computing the digital image just with the accuracy needed for a given resolution.

This work has led to some new ideas of how to deal with IFS, especially with the inverse problem. Hitherto known methods use a search technique optimizing an error criterion to find an IFS [2, 13, 9, 10]. We think it is possible to avoid this expensive step by computing the IFS *directly* from the image, using pyramids.

2 Basics of IFS

Let (X, d) denote a complete metric space with distance function d. A mapping $w : X \to X$ is called a *contractive mapping* on X if there exists a real number $c \in [0, 1)$ such that $d(w(x), w(y)) \leq cd(x, y)$ for all $x, y \in X$. The smallest c for which this condition holds is the *contractivity factor* of w. For every contractive mapping w there exists a unique *fixed point* $x_F \in X$ with $w(x_F) = x_F$. The sequence $\{x, w(x), \ldots, w^i(x), \ldots\}$ is the *orbit* of x under w. The orbit of any $x \in X$ converges to the fixed point of the contractive mapping w. The rate of convergence coincides inversely with the contractivity factor of w.

Let $H(X)$ be the class of all non-empty compact subsets of X. A contractive mapping w on X can be extended to a mapping on $H(X)$ in the following way:

$$w(A) = \{w(a) : \text{for all } a \in A\} \qquad A \in H(X) \tag{1}$$

A distance function d_h, the so called *Hausdorff metric*, can be defined so that $(H(X), d_h)$ forms a complete metric space [5]. If a mapping w is contractive on (X, d) it is also contractive on $(H(X), d_h)$ [1, 6]. The Hausdorff metric is defined as

$$d_h(A, B) = \inf\{\delta : A \subseteq B_\delta \text{ and } B \subseteq A_\delta\} \qquad A, B \in H(X) \tag{2}$$

where A_δ is the *δ-parallel body* of $A \in H(X)$; this is the set of points within distance δ of A, i.e. $A_\delta = \{x \in X : d(x, a) \leq \delta \text{ for some } a \in A\}$.

2.1 Affine Transformations in $\mathbb{R}^2$

We will take a closer look at the special case of $\mathbb{R}^2$ because a binary image can be regarded as a compact subset of $\mathbb{R}^2$.

In the following $d(\bar{x}, \bar{y})$ will denote the *Euclidian metric*, i.e. the distance between two points $\bar{x}, \bar{y} \in \mathbb{R}^2$. $(\mathbb{R}^2, d)$ forms a complete metric space. The contractive mappings we will consider are *affine*

transformations, since they can be easily understood in terms of geometric operations, e.g. rotation, translation and scaling. An affine transformation is defined by

$$w(\bar{x}) = L\bar{x} + \bar{t} \qquad \bar{x} \in \mathbb{R}^2. \tag{3}$$

The 2×2 matrix L sums up the effects of rotations, skews, reflection, and scaling whereas $\bar{t} \in \mathbb{R}^2$ is a translation. If $\bar{x}_F \in \mathbb{R}^2$ is the fixed point of w, the transformation can also be written as

$$w(\bar{x}) = L(\bar{x} - \bar{x}_F) + \bar{x}_F. \tag{4}$$

From this we can conclude that the geometric operations contained in L operate in relation to the fixed point, e.g. $\bar{x}_F$ is the center of the rotation.

To show which geometric operations the matrix L performs it can be split up into the product of a rotation, a scaling/reflection, and another rotation:

$$L = \mathrm{R}(\beta)\mathrm{S}(c_x, c_y)\mathrm{R}(\alpha) \tag{5}$$

with $\alpha, \beta \in [0, 2\pi)$ and $c_x, c_y \in (-1, 1)$. The operators R and S generate the following matrices:

$$\mathrm{R}(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \quad \mathrm{S}(c_x, c_y) = \begin{pmatrix} c_x & 0 \\ 0 & c_y \end{pmatrix} \tag{6}$$

Now that the basics have been explained it is possible to illustrate the effect of affine transformations with some images of orbits in Fig. $1-3$. The starting point is $(1, 1)$ for all of them.

2.2 Deterministic IFS

A *deterministic iterated function system* is a finite set of two or more contractive mappings from X to itself [6].

The *Hutchinson operator* $\mathbf{w}(A)$ on a set $A \in H(X)$ for an IFS $\{w_i : i = 1, \ldots, n\}$ is the union of the images of the set A under the n mappings of the IFS:

$$\mathbf{w}(A) = \bigcup_{i=1}^{n} w_i(A) \tag{7}$$

The effect of the Hutchinson operator of the IFS

$$\begin{aligned} w_1(\bar{x}) &= \mathrm{S}(0.5, 0.5)\bar{x} \\ w_2(\bar{x}) &= \mathrm{S}(0.5, 0.5)\bar{x} + (0.25, \sqrt{3/16}) \qquad \bar{x} \in \mathbb{R}^2 \\ w_3(\bar{x}) &= \mathrm{S}(0.5, 0.5)\bar{x} + (0.5, 0) \end{aligned}$$

is illustrated in Fig. 4. The starting image is the gray filled rectangle in the upper left corner of the figure. Applying the Hutchinson operator on it yields the middle image in the upper row. As we proceed, the Sierpinski triangle in the lower right corner is approximated closer and closer. This happens always, no matter which starting image is used. The image that is approximated is called the *attractor* of an IFS.

Hutchinson proved [6] that $\mathbf{w}$ is a contractive mapping on $(H(X), d_h)$. Therefore $\mathbf{w}$ has a unique fixed point $\mathcal{A} \in H(X)$ which is the attractor of the corresponding IFS and the solution of

$$\mathcal{A} = \mathbf{w}(\mathcal{A}). \tag{8}$$

Fig. 5 shows some IFS and their attractors.

Figure 1: Orbit of transformation $w(\bar{x}) = \mathrm{S}(0.6, 0.9)\bar{x}$

Figure 2: Orbit of transformation $w(\bar{x}) = \mathrm{R}(10^{\circ})\mathrm{S}(0.9, 0.9)\bar{x}$

Figure 3: Orbit of transformation $w(\bar{x}) = \mathrm{R}(10^{\circ})\mathrm{S}(-0.9, 0.9)\bar{x}$

Figure 4: Effect of the Huchinson operator on a rectangle

$$w_1(\bar{x}) = \mathrm{R}(10^\circ)\mathrm{S}(0.8, 0.8)\bar{x} \qquad w_1(\bar{x}) = \mathrm{S}(0.9, 0.8)\mathrm{R}(-20^\circ)\bar{x}$$
$$w_2(\bar{x}) = \mathrm{R}(20^\circ)\mathrm{S}(0.64, 0.64)\bar{x} + (5, 3) \qquad w_2(\bar{x}) = \mathrm{S}(0.8, 0.9)\mathrm{R}(20^\circ)\bar{x} + (2, 1)$$

Figure 5: Some attractors

2.3 Probabilistic IFS

A *probabilistic iterated function system* is a deterministic IFS extended by assigning probabilities p_i to each transformation w_i, which sum up to 1 [6].

Let $P(X)$ denote the space of normalized Borel measures on X. Each w_i induces a Markov operator M_i on $P(X)$ in the following way:

$$M_i(\mu)(A) = \mu(w_i^{-1}(A)) \qquad A \in H(X),\ \mu \in P(X) \tag{9}$$

The operator $\mathbf{M} : P(X) \to P(X)$ for the whole IFS is then defined by

$$\mathbf{M} = \sum_{i=1}^{n} p_i M_i. \tag{10}$$

There exists an *invariant measure* [6] which is the unique solution $\mu_F \in P(X)$ of

$$\mu_F(A) = \mathbf{M}(\mu_F)(A) \qquad \text{for all } A \in H(X). \tag{11}$$

The measure defined by an IFS in $\mathbb{R}^2$ can be interpreted as color or gray levels.

2.4 Sampling the Attractor

An IFS in $\mathbb{R}^2$ defines an attractor that is a subset of $\mathbb{R}^2$. Now we must find a way to display this subset as a binary image on a screen. Therefore, the screen is identified with its index set which is a finite subset of $\mathbb{N}^2 \subset \mathbb{R}^2$. The area containing the attractor is subdivided into small squares that correspond to the pixels of the screen. A pixel is set to one if and only if the corresponding square contains at least one point of the attractor. The resulting binary image is called the *sampled attractor*.

The Chaos Game [1] is the mostly used strategy for sampling an attractor:

Initialization A starting point $x_0 \in \mathcal{A}$ is chosen, e.g. the fixed point of any transformation in the IFS. j is set to 0.

Iteration The pixel corresponding to x_j is set. The next step is to compute $x_{j+1} = w_{i_j}(x_j)$, where $i_j \in \{1, \ldots, n\}$ is selected at random with equal probability $p = \frac{1}{n}$.

The process should be stopped when the attractor is plotted densely enough. Since the random walk on the attractor shows chaotic behaviour, this should not take too much time [5].

For probabilistic IFS the transformations are selected according to their probabilities p_i. This has two effects: firstly the attractor is covered more regularly and therefore faster provided that the probabilities are chosen according to the area of the attractor under the transformations; secondly, using the invariant measure μ_F, it is possible to get an image with color or grey values of the attractor: Each pixel is assigned a color or gray value in respect of its measure. Concerning the chaos game this can be done by counting the number of hits on a pixel and normalizing it by the maximum number of hits. Colors or gray levels are assigned on behalf of the so obtained value by a mapping function.

3 From IFS to Images

The Chaos Game is a rather expensive way to sample an attractor, calculating it much more exactly than ever necessary. This is because the limit of the Chaos Game is the real attractor as a mathematical object and there is no obvious way to conclude the number of iterations that is sufficient for a given resolution.

Image pyramids are the key to many efficient algorithms. Our first step will be to show how the attractor and its invariant measure can be computed from an IFS directly in discrete space; subsequently we combine this with image pyramids.

3.1 Direct Computation of the Attractor and the Invariant Measure

To compute the attractor or its invariant measure directly for a given screen resolution it is necessary to represent them as finite sets. We will work on the discrete space $\mathbb{P}^2 \subset \mathbb{N}^2$ of screen pixels, for simplicity assuming a square screen. For a screen with dimension R

$$\mathbb{P}_R = \{0, 1, \ldots, R-1\} \tag{12}$$

and $\mathbb{P}_R^2$ is defined as $\mathbb{P}_R \times \mathbb{P}_R$. We identify the screen with $[0,R) \times [0,R) \subset \mathbb{R}^2$; each pixel $p = (p_x, p_y) \in \mathbb{P}_R^2$ represents a unit sqare $[p_x, p_x+1) \times [p_y, p_y+1) \subset \mathbb{R}^2$. For simplicity we assume the attractor lying entirely in $[0,R) \times [0,R)$ (any IFS can be transformed into an equivalent one for which this assumption holds).

We consider finite subsets $I \subseteq \mathbb{P}_R^2$. Every subset I defines a binary image in a way that the pixels $\in I$ are set and the pixels $\notin I$ are not set and vice versa. Hence, for simplicity we can call these subsets I images. The gray levels of an image I are defined by specifying the measures μ_p for all $p \in I$. We have to define the meaning of $\mathbf{w}$ and $\mathbf{M}$ on $\mathbb{P}_R^2$; therefore a correspondence between $\mathbb{P}_R^2$ and $\mathbb{R}^2$ has to be given. This is done by defining a mapping ϱ from $\mathbb{P}_R^2$ to $H(\mathbb{R}^2)$, i.e. ϱ relates a pixel to the subset of $\mathbb{R}^2$ representing it. We explore two possibilities:

1. Each pixel is represented by its center.

$$\varrho^+(p) = \{\bar{x} \in \mathbb{R}^2 : \bar{x} = (p_x + 0.5, p_y + 0.5)\} \qquad p \in \mathbb{P}_R^2 \tag{13}$$

2. Each pixel is represented by a unit square subset of $\mathbb{R}^2$.

$$\varrho^\square(p) = [p_x, p_x+1) \times [p_y, p_y+1) \qquad p \in \mathbb{P}_R^2 \tag{14}$$

The first choice will lead to simple algorithms. The second one may be more exact.

After applying an affine transformation on $\varrho(p)$ for a pixel p the result has to be mapped back to $\mathbb{P}_R^2$ again. This is done by the function Δ which maps subsets of $\mathbb{R}^2$ to subsets of $\mathbb{P}_R^2$:

1. If pixels are represented by their centers, Δ^+ has to operate on points in $\mathbb{R}^2$. It selects the pixel that contains a given point.

$$\Delta^+(\{\bar{x}\}) = \{p \in \mathbb{P}_R^2 : p = (\lfloor \bar{x}_x \rfloor, \lfloor \bar{x}_y \rfloor)\} \qquad \bar{x} = (\bar{x}_x, \bar{x}_y) \in \mathbb{R}^2 \tag{15}$$

2. If pixels are represented as square subsets of $\mathbb{R}^2$ the result of applying w_i on $\varrho(p)$ is a parallelogram. Hence, $\Delta^\square$ maps a subset of $\mathbb{R}^2$ (namely a parallelogram) to those pixels intersecting it.

$$\Delta^\square(A) = \{p \in \mathbb{P}_R^2 : \varrho^\square(p) \cap A \neq \emptyset\} \qquad A \subset \mathbb{R}^2 \tag{16}$$

Note that the result of Δ is guaranteed to lie inside of $\mathbb{P}_R^2$; everything outside the screen is discarded. When we write ϱ or Δ in the following, ϱ^+, Δ^+ or $\varrho^\square$, $\Delta^\square$ respectively are possible depending on the pixel representation.

3.1.1 Computing the Discrete Attractor

For the following, it is necessary to define the relation between a transformation $w : \mathbb{R}^2 \to \mathbb{R}^2$ and a *discrete transformation* w' that maps a pixel to a set of pixels. This is done in the following way:

$$w'(p) = (\Delta \circ w \circ \varrho)(p) \qquad p \in \mathbb{P}_R^2 \tag{17}$$

The pixel p is mapped by ϱ into $\mathbb{R}^2$ first, then the transformation w is applied and the resulting subset of $\mathbb{R}^2$ is finally transformed into a pixel-set by Δ. Notice that this approach also allows the use of transformations other than affine ones. In Fig. 6 you can see a pixel being mapped to others under some transformation. On the left, the pixel is represented by its center, on the right, representation as a square is used.

Figure 6: Different methods of pixel representation

The discrete Hutchinson operator $\mathbf{w}'$ of an IFS $\{w_i : i = 1, \ldots, n\}$ maps a pixel-set I into a pixel-set and is defined as

$$\mathbf{w}'(I) = \bigcup_{i=1}^{n} \bigcup_{p \in I} w_i'(p) \qquad I \subseteq \mathbb{P}_R^2. \tag{18}$$

Each pixel of image I is discretly transformed for each mapping of the IFS.

Any set $\mathcal{A}_d$ satisfying the following equation is called a *discrete attractor*:

$$\mathcal{A}_d = \mathbf{w}'(\mathcal{A}_d) \tag{19}$$

Due to discretization errors, this equation has generally no unique solution as in the continuous case (Eq. 8). Perhaps it has exactly one maximal solution for any IFS.

We think an improvement of discrete transformations can be achieved by a modification of the above. If a discrete transformation maps a pixel to itself, two cases can be distinguished: First, the fixed point of this transformation is situated in this pixel. The pixel stays mapped to itself. Secondly, the fixed point is not situated in this pixel. Then the transformation w is applied repeatedly until another pixel is reached. Rough experiments have shown that this may lead to discrete attractors that are closer to the sampled ones.

Algorithm RemoveBackground

The algorithm given here computes the maximal discrete attractor for an IFS $\{w_i : i = 1, \ldots, n\}$.

A matrix $\Pi = (\pi_{pq})_{p,q \in \mathbb{P}_R^2}$ can be built that tells where the pixels of $\mathbb{P}_R^2$ are mapped to under the transformations w_i:

$$\pi_{pq} = \begin{cases} 1 & \text{if } q \in w_i'(p) \text{ for some } i \in \{1, \ldots, n\} \\ 0 & \text{otherwise} \end{cases} \qquad p, q \in \mathbb{P}_R^2 \tag{20}$$

Π can be interpreted as the adjacency matrix of a directed graph. The nodes coincide with pixels and (p, q) is an arc of the graph if pixel p is mapped into the area of pixel q. The effect of affine transformations is therefore entirely described by a directed graph.

We can restrict the directed graph specified by Π to the pixels of a certain image I. Let its adjacency matrix be denoted by Π_I.

For a discrete attractor $\mathcal{A}_d$ all nodes of graph $\Pi_{\mathcal{A}_d}$ must have at least one incoming arc. This is because the attractor is self-tiling under its transformations. Hence, we only need to find the maximal subset of $\mathbb{P}_R^2$ for which this condition holds to get the maximal discrete attractor.

Algorithm RemoveBackground computes the maximal discrete attractor $\mathcal{A}_d$ for a given IFS:

- Set I to $\mathbb{P}_R^2$.
- Compute Π by using Eq. 20.
- The nodes with no incoming arc in Π_I are deleted from I recursively until none of that kind exist.

Then I is the maximal discrete attractor of the IFS.

Proof: The attractor lies entirely inside the area of the screen and RemoveBackground starts with whole $\mathbb{P}_R^2$; thus, it computes the **maximal** subset I of $\mathbb{P}_R^2$ for which Π_I is a graph whose nodes all have at least one incoming arc. So I has to be the maximal discrete attractor as stated above.

Algorithm SetForeground

This algorithm takes the assumption that the graph of each discrete attractor is completely connected, i.e. each node can be reached from every other.

Initialization We can start with any pixel of the attractor, e.g. a pixel where a fixed point is situated. This pixel is put on a list.

Iteration A pixel is taken from the list, marked and set on the screen. Each transformation is applied to the pixel, yielding a set of successor pixels. Every unmarked successor pixel is put on the list. The process continues until the list is empty.

It is still to explore if the connectivity assumption stated above holds for every IFS.

SetForeground has a lower memory and run-time requirement on regular sequential computers than RemoveBackground.

Differences Between Sampled and Discrete Attractor

Stark [11] has proved that the difference (i.e. the Hausdorff metric) between a sampled and a maximal discrete attractor is bounded by a term that depends only on the contractivity factor of the IFS.

Fig. 7 shows images of the fern attractor at a resolution of 256×256 pixels. The leftmost attractor

Figure 7: Sampled and discrete fern attractors

has been sampled with the chaos game. The following two were generated with RemoveBackground and SetForeground respectively. The rightmost is the result of both RemoveBackground and SetForeground extended with the improvement mentioned in section 3.1.1. Among the discrete attractors, it is closest to the sampled attractor. This gives a strong evidence for the usefulness of the improvement. Neither are the discrete attractors fully contained in the sampled attractor nor vice versa.

3.1.2 Computing the Invariant Measure

A discrete measure $\bar{\mu}$ on $\mathbb{P}_R^2$ is defined by specifying the measures for all pixels in $\mathbb{P}_R^2$, yielding a vector:

$$\bar{\mu} = (\bar{\mu}_p)_{p \in \mathbb{P}_R^2} \tag{21}$$

So the discrete Markov operator $\mathbf{M}'$ of an IFS is defined as

$$\mathbf{M}'(\bar{\mu}) = \bar{\mu}\Gamma \tag{22}$$

where $\Gamma = (\gamma_{pq})_{p,q \in \mathbb{P}_R^2}$ is a matrix giving the effect of the measures sent from p to q under the Markov operator:

$$\gamma_{pq} = \sum_{i=1}^{n} p_i \pi_{pqi} \qquad\qquad p, q \in \mathbb{P}_R^2 \tag{23}$$

p_i is the probability assigned to w_i. π_{pqi} indicates how much of p is mapped into the region of q under w_i:

1. If pixels are represented by their centers the whole measure of a pixel p is sent to that pixel q where its center is mapped into under w_i:

$$\pi_{pqi}^{+} = \begin{cases} 1 & \text{if } \{q\} = (\Delta^{+} \circ w_i \circ \varrho^{+})(p) \\ 0 & \text{otherwise} \end{cases} \qquad p, q \in \mathbb{P}_R^2,\ 1 \le i \le n \tag{24}$$

2. If pixels are represented as square subsets of $\mathbb{R}^2$ the measure of p is distributed among those pixels it overlaps under w_i in the ratio of the areas of the overlapping sections:

$$\pi_{pqi}^{\square} = \frac{\text{area}((w_i \circ \varrho^{\square})(p) \cap \varrho^{\square}(q))}{\text{area}((w_i \circ \varrho^{\square})(p))} \qquad p, q \in \mathbb{P}_R^2,\ 1 \le i \le n \tag{25}$$

   area(A) denotes the area of a subset A of $\mathbb{R}^2$.

The discrete invariant measure $\bar{\mu}_F = \mathbf{M}'(\bar{\mu}_F)$ is then a normalized solution of the homogenous system of linear equations defined by

$$\bar{\mu}_F(\Gamma - E) = 0. \tag{26}$$

E is the unit matrix. Only that solution is valid where the measures $\bar{\mu}_{Fp}$ of the pixels $p \notin \mathcal{A}_d$ are zero; therefore just the pixels that are part of the attractor have to be considered. They can be determined with the algorithms given in the last section.

3.2 Improving Performance Of Direct Computation Using Image Pyramids

Throughout this section we use a non-overlapping pyramid structure as it is described in [7].

It is useful to discover regions of the screen as large as possible where the attractor is guaranteed not to lie: These regions need not be considered in the RemoveBackground algorithm described in section 3.1.1. This is the aim of the following.

Fig. 8 shows the attractor of Barnsley's fern [1] computed with the Chaos Game for resolutions 256×256, 128×128, 64×64, 32×32, 16×16, 8×8, and 4×4. At a resolution of 8, for example, more than half of the pixels lie outside of the attractor and need no refinement at higher resolutions.

Figure 8: Pyramid of Barnsley's fern created by the Chaos Game

Suppose we have found the pixels of the attractor and their measures at level I_i. For each of these pixels we set up links to its sons, thereby determining the pixels to start the direct computation with at level I_{i-1}. Sons of pixels that are not part of the solution at level I_i can be neglected from level I_{i-1} on. The procedure starts at the top level I_N of the pyramid. Weights for links between the levels can be provided such that the measures of the sons can be established in terms of their father pixel's measure. When the process is finished at I_0, all information of the image is contained in the father-son links.

If pixels are represented by their centers, it may be that parts of the attractor's border area have been lost during the computation of level I_N to level I_i due to error in representation. We assume that they will be captured again at level I_{i-1} if they are big enough to be part of the solution defined by this level. Otherwise, the algorithm may find them at a lower level in the pyramid. The correctness of this is to be proved.

3.3 Efficient Image Reconstruction from Image Pyramids

The weighted top-down links in the pyramid can be used to generate the original image in $O(\log R^2)$ parallel steps. Imagine that all points within the pixels of the original image are summarized in the top pixel. Then they can be redistributed onto the higher pixels in proportion to the link weights. The original distribution of points is correctly reconstructed in one parallel top-down pass through the pyramid.

3.4 An Illustrative Example in $\mathbb{R}$

To help the reader to gain insight into the procedure described in section 3.2 we present an example in $\mathbb{R}$ (Fig. 9).

Figure 9: Pyramid for IFS $\{w_1, w_2\}$ in $\mathbb{R}$

The attractor of the IFS $\{w_1, w_2\}$ with

$$w_1(x) = \frac{1}{4}x \qquad w_2(x) = \frac{1}{2}(x - 16) + 16 \qquad\qquad x \in \mathbb{R} \tag{27}$$

is computed. Algorithm RemoveBackground deletes all pixels with no incoming arc resulting in the four graphs shown in Fig. 9. The arcs of the graph are labeled with the corresponding transformations. Pixels, i.e. intervals, are represented by their center (drawn as small circles). A pixel p is mapped to that pixel q into which p's center is transformed. The shape of the attractor is shown in bold intervals.

The construction of the example starts at the top with two nodes. Since all pixels have incoming arcs at this low resolution, all four sons at the next lower level are candidates for belonging to the attractor. This time one pixel is removed and the six sons of the remaining three pixels are inspected below. Dashed lines show these father-son relations in Fig. 9.

4 From Images to IFS — The Inverse Problem

The pyramidal approach has proved to be helpful to compute the attractor and the invariant measure of an IFS directly. Let's now explore the inverse way, i.e. the inverse problem.

Constructing a pyramid for a given image allows to access the inherent structural information of the image at multiple resolutions. Considering both this pyramid and the one defined by an IFS it seems that pyramids are a form of intermediate representation between images and IFS, narrowing the gap between them.

This and the fact that we have found solutions for quite constrained cases (see following sections) let the idea of computing an IFS directly from an image seem more realistic than hitherto.

When we look for a way to compute an IFS directly from an image, it seems reasonable to begin with simple cases. Therefore, we will constrain the IFS we are searching for to such ones with two transformations in the form of Eq. 4, where

$$L = \mathrm{S}(c, c) \tag{28}$$

with c being the contractivity factor. This means that only a contraction which is the same for horizontal and vertical direction, and maybe a reflection take place.

The attractor yielded by such an IFS is always situated on the line that is determined by the two fixed points, i.e. it will look like a straight line including the fixed points, maybe with some interruptions. Only one dimension is of importance here; there is no reason to stay in 2D. That is why we start our research looking for some IFS in $\mathbb{R}$.

4.1 Direct Computation of an IFS in $\mathbb{R}$

As an example we show how to compute a first approximation of a deterministic IFS for a row of pixels. Fig. 10 shows the pixel pattern we consider.

Figure 10: Pixel pattern

Given two transformations we compute the largest gap and an upper bound for the attractor. The transformations in $\mathbb{R}$ may be reflexive ($c_i < 0$) or not:

Figure 11: Boundaries of non-reflecting IFS

1. Assuming two non-reflecting transformations, boundaries f_1, f_2, e_1, e_2 are given (Fig. 11).

   In this case the parameters of $w_1(x) = c_1(x - f_1) + f_1$ and $w_2(x) = c_2(x - f_2) + f_2$ are given by

   $$c_1 = \frac{e_1 - f_1}{f_2 - f_1} \quad \text{and} \quad c_2 = \frac{f_2 - e_2}{f_2 - f_1}. \tag{29}$$

   Note that the fixed points are among the observed boundaries.

2. Assuming two reflecting transformations, we observe the boundaries m_1, m_2, e_1, e_2 (Fig. 12).

Figure 12: Boundaries of reflecting IFS

The parameters of $w_1(x) = c_1(x - f_1) + f_1$ and $w_2(x) = c_2(x - f_2) + f_2$ are computed as follows:

$$c_1 = -\frac{e_1 - m_1}{m_2 - m_1}, \qquad c_2 = -\frac{m_2 - e_2}{m_2 - m_1} \tag{30}$$

and

$$f_1 = \frac{1}{1 - c_1} m_1 - \frac{c_1}{1 - c_1} m_2, \qquad f_2 = \frac{1}{1 - c_2} m_2 - \frac{c_2}{1 - c_2} m_1 \tag{31}$$

4.2 Refinement of Direct Computation in $\mathbb{R}$

Until now only the rough structure (outer boundaries and largest gap, which divides the attractor in parts) of the pixel row has been regarded. For a better approximation of the pixel row it is necessary to examine its structure on a finer level (a pyramid is suited very well for this). Then, some smaller gaps will become visible.

We have two possibilities now: First of all we can choose the better matching IFS from reflecting and non-reflecting IFS. Secondly, if the approximation is still not close enough, additional transformations will be necessary. Considering a smaller gap, we will have to change the transformation that yields that part of the attractor which has been divided by the smaller gap in the finer level of resolution. This transformation will be split up into two transformations. Note that all other transformations stay the same, for the parts of the attractor they yield are not concerned.

One can gradually increase resolution, choosing between reflecting and non-reflecting transformations and splitting them up at each level until the desired pixel pattern is reached. In the worst case, this system results in as many transformations as the attractor has parts, but results may be better.

We think these issues are worth exploring because mastering the 1D case will help a great deal in solving the inverse problem for 2D.

4.3 Continuous $\mathbb{R}^2$-Orbits

In order to extend the above example into $\mathbb{R}^2$, we must find a way to represent an orbit as a continuous curve. This can be done by parameterizing the orbit curve.

For a transformation $w(\bar{x}) = \mathrm{S}(c,c)\mathrm{R}(\alpha)(\bar{x}-\bar{x}_F)+\bar{x}_F$ (i.e. every affine transformation that has a uniform contraction in all directions) the paramterized orbit of startpoint $\bar{x}_0 \in \mathbb{R}^2$ can be denoted by

$$o_w(t,\bar{x}_0) = \mathrm{S}(c^t,c^t)\mathrm{R}(t\alpha)(\bar{x}-\bar{x}_F)+\bar{x}_F \qquad t \in [0,\infty) \tag{32}$$

with $\alpha \in [0,2\pi)$ and $c \in [0,1)$, which means that only non-reflecting transformations can be parameterized.

For a transformation $w(\bar{x}) = \mathrm{S}(c_x,c_y)(\bar{x}-\bar{x}_F)+\bar{x}_F$ the paramterized orbit of startpoint $\bar{x}_0 \in \mathbb{R}^2$ can be denoted by

$$o_w(t,\bar{x}_0) = \mathrm{S}(c_x^t,c_y^t)(\bar{x}-\bar{x}_F)+\bar{x}_F \qquad t \in [0,\infty) \tag{33}$$

with $c_x, c_y \in [0,1)$.

The orbit of a reflecting transformation can be split up into two orbits of non-reflecting ones, where $\bar{x}_0$ and $w(\bar{x}_0)$ are the starting points and w^2 is used instead of w.

Fig. 13 shows the continuous orbit of transformation $w(\bar{x}) = \mathrm{S}(0.8,0.8)\mathrm{R}(30°)\bar{x}$ and starting point $(1,1)$.

Figure 13: Continuous orbit

4.4 Classes of IFS

In this section, we will only consider IFS with two non-reflecting transformations. Continuous orbits imply a division of transformations into classes. Each class consists of the transformations with the same continuous orbit. If $v(\bar{x}) = S(c,c)\mathrm{R}(\alpha)(\bar{x}-\bar{x}_F)+\bar{x}_F$ is a member of a class, then any transformation

$$w(\bar{x}) = \mathrm{S}(c^f,c^f)\mathrm{R}(f\alpha)(\bar{x}-\bar{x}_F)+\bar{x}_F \qquad f \in \mathbb{R},\ f>0 \tag{34}$$

is also a member of this class. If $v(\bar{x}) = \mathrm{S}(c_x, c_y)(\bar{x} - \bar{x}_F) + \bar{x}_F$ is a member of a class, then any transformation

$$w(\bar{x}) = \mathrm{S}(c_x^f, c_y^f)(\bar{x} - \bar{x}_F) + \bar{x}_F \qquad f \in \mathbb{R},\ f > 0 \tag{35}$$

is also a member of this class.

Using this we can divide also IFS into classes. Those IFS are in the same class whose first transformations lie in the same class and the second ones accordingly.

4.5 Border Curve of an IFS

The *border curve* of an IFS is an analytically defined curve that bounds the attractor of the IFS. In the case of IFS with two non-reflecting transformations we assume that the border curve is built of parts of the continuous orbits of two *border points*. There are two necessary conditions for a pair of border points (see Fig. 14):

1. Each border point must be situated on one continuous orbit of the other border point, i.e. M1 must be situated on M2's orbit to the fixed point F1 and vice versa.
2. The tangents of the two continuous orbits at each border point have to be equal, i.e. there has to be a smooth join of orbits.

Figure 14: Border curve of an IFS with two transformations

If these conditions are satisfied for a pair of points, then the attractor is sure to lie inside the so-defined border curve. It is not garantueed that there exists no other pair of points that define a tighter border curve.

It follows from the above that all IFS in an IFS class have the same border curve.

4.6 Largest Gaps of an IFS

Again we consider only IFS with two non-reflecting transformations. Looking at an attractor bounded by a border curve (Fig. 15) we can see the largest empty areas in the neighbourhood of the border points. We will call such areas *largest gaps* of an IFS. IFS of the same class will always have their largest gaps situated in quite the same place because their fixed points are the same. Nevertheless, shape and size of these gaps will differ. Continous variation of parameter f in Eq. 34, 35 will lead to a continous variation of shape and size of the largest gaps. The larger f is the larger the gaps will be.

Figure 15: Largest gaps of an IFS

4.7 Deriving IFS Parameters from Geometric Properties

This is a rough sketch of how to derive the parameters of an IFS with two non-reflecting transformations from the geometric properties of a given image. Looking at an image it is possible to draw a curve that bounds the object shown in it. Then the largest empty areas inside the bounded area can be concluded. These two geometric properties can be related to the border curve and the largest gaps. As it was possible to derive IFS parameters from borders and gaps in the one dimensional case (see 4.1) it may be possible to do this in two dimensions by using the properties described before (and maybe some others).

5 Conclusion

The following has been shown:

- The discrete attractor and its invariant measure can be computed directly for a given resolution.
- This method can be enhanced by using image pyramids.
- In a simple 1D case IFS can be computed directly from a row of pixels.
- A parameterized form of continous orbits in $\mathbb{R}^2$ was introduced.
- A division into classes was done for transformations and IFS.

We believe that it is worth investigating into the recovery of fractal parameters from images because current computer vision systems are not good in representing natural objects like plants, clouds, etc. The capability of recognizing fractals would therefore enhance the potential of current vision systems.

References

[1] Michael F. Barnsley. *Fractals Everywhere*. Academic Press, 1988.

[2] Michael F. Barnsley, V. Ervin, D. Hardin, and J. Lancaster. Solution of an inverse problem for fractals and other sets. *Proceedings of the National Academy of Sciences, USA*, 83, 1986.

[3] Michael F. Barnsley, Arnaud Jacquin, Francois Malassenet, Laurie Reuter, and Alan D. Sloan. Harnessing chaos for image synthesis. *Computer Graphics*, 22(4):131–140, August 1988.

[4] Michael F. Barnsley and Alan D. Sloan. A better way to compress images. *BYTE*, pages 215–223, January 1988.

[5] Kenneth Falconer. *Fractal Geometry, Mathematical Foundations and Applications*. Wiley & Sons, 1990.

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

[7] Walter G. Kropatsch. Kurvenrepräsentation in Pyramiden. In Walter G. Kropatsch and P. Mandl, editors, *Mustererkennung '86*, pages 16–51. Oldenbourg, Band 36, 1986.

[8] Walter G. Kropatsch, Michael A. Neuhauser, Irene J. Leitgeb, and Horst Bischof. Combining Pyramidal and Fractal Image Coding. Extended abstract submitted to the 11$^{\text{th}}$ ICPR '92, The Hague, October 1991.

[9] J. Lévy-Véhel and André Gagalowicz. Shape approximation by a fractal model. In Guy Maréchal, editor, *Proceedings of the EUROGRAPHICS 1987*, pages 159–179. Eurographics Association, Elsevier Science Publishers, August 1987.

[10] J. Lévy-Véhel and André Gagalowicz. Fractal approximation of 2-D object. In David A. Duce and Pierre Jancene, editors, *Proceedings of the EUROGRAPHICS 1988*, pages 297–311. Eurographics Association, Elsevier Science Publishers, September 1988.

[11] Jaroslav Stark. Iterated function systems as neural networks. *Neural Networks*, 4:679–690, 1991.

[12] Jaroslav Stark. A neural network to compute the hutchinson metric in fractal image processing. *IEEE Transactions on Neural Networks*, 2(1):156–158, January 1991.

[13] Edward R. Vrscay. Moment and collage methods for the inverse problem of fractal construction with iterated function systems. In *SIGGRAPH '91 Course Notes*, volume C14, pages 271–289. The Association for Computing Machinery, 1991.