Polar IFS + Individual Genetic Programming = Efficient IFS Inverse Problem Solving

Pierre Collet, Evelyne Lutton, Frédéric Raynal, Marc Schoenauer

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Polar IFS + Individual Genetic Programming
=
Efficient IFS Inverse Problem Solving

Pierre COLLET, Evelyne LUTTON, Frédéric RAYNAL,

Marc SCHOENAUER

N ° 3849

December 21, 1999

THÈME 4

ISSN 0249-6399 ISRN INRIA/RR--3849--FR+ENG

Polar IFS + Individual Genetic Programming
=
Efficient IFS Inverse Problem Solving

Pierre COLLET, Evelyne LUTTON, Frédéric RAYNAL,
Marc SCHOENAUER *

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet FRACTALES

Rapport de recherche n3849 — December 21, 1999 — 28 pages

Abstract: The inverse problem for Iterated Functions Systems (finding an IFS whose attractor is a target 2D shape) with non-affine IFS is a very complex task. Successful approaches have been made using Genetic Programming, but there is still room for improvement in both the IFS and the GP parts.

The main difficulty with non-linear IFS is the efficient handling of contractance constraints. This paper introduces Polar IFS, a specific representation of IFS functions that shrinks the search space to mostly contractive functions. Moreover, the Polar representation gives direct access to the fixed points of the functions, whereas the fixed point of general non-linear IFS can only be numerically estimated.

On the evolutionary side, the "individual" approach is similar to the Michigan approach of Classifier Systems: each individual of the population embodies a single function rather than the whole IFS. A solution to the inverse problem is then built from a set of individuals.

Both improvements show a drastic cut-down on CPU-time: good results are obtained with small populations in few generations.

Key-words: Genetic Algorithms, optimization, Iterated function systems, inverse problem.

(Résumé : tsvp)

* {Pierre.Collet, Evelyne.Lutton, Frederic.Raynal}@inria.fr, `http://www-rocq.inria.fr/fractales/`, Marc.Schoenauer@polytechnique.fr, `http://www.eeaax.polytechnique.fr`

Unit´e de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
T´el´ephone : 01 39 63 55 11 - International : +33 1 39 63 55 11
T´el´ecopie : (33) 01 39 63 53 30 - International : +33 1 39 63 53 30

IFS polaires et stratégie "individuelle" de programmation génétique : une méthode efficace de résolution du problème inverse pour les IFS

Résumé : Lorsque l'on s'intéresse aux IFS (systèmes de fonctions itérées) non affines, la résolution du problème inverse (c'est-à-dire trouver l'IFS dont l'attracteur approxime au mieux une forme bidimensionnelle donnée) devient un problème très complexe. Ce problème a déjà été résolu avec succès à l'aide de stratégies de programmation génétique, fondées sur une représentation des fonctions sous forme d'arbres. La principale difficulté de cette approche étant la gestion efficace des contraintes de contractance sur les fonctions, nous proposons ici l'emploi d'une représentation polaire des IFS non affines, centrée sur le point fixe de chaque fonction. Cette représentation a deux principaux avantages :

1. une contrainte simple sur la définition de la composante radiale de chaque fonction assure sa convergence vers un point fixe (le point central de sa representation polaire),
2. l'accès au point fixe de chaque fonction est direct (il n'est plus nécessaire de l'estimer comme dans l'approche en coordonnées cartésiennes).

Nous présentons ensuite une stratégie originale de programmation génétique, fondée sur une exploitation plus "économique" des stratégies évolutionnaires : l'approche "individuelle", où chaque individu de la population représente une seule fonction (au lieu d'un IFS complet). La solution au problème étant fournie par un ensemble d'individus de la population finale, des résultats sont obtenus de façon plus rapide et plus efficace que dans la version classique où tous les individus de la population finale sauf un (le meilleur) sont écartés.

Mots-clé : Algorithmes Génétiques, optimisation, systèmes des fonctions itérées, problème inverse.

1 Introduction

Iterated Functions System (IFS) theory is an important topic in fractals, and provides powerful tools for the investigation of fractal sets. The action of systems of contractive mappings to produce fractal sets has been considered by many authors (see for example [15, 2, 3, 9, 12]), and most fractal image compression techniques are based on IFS [5, 17, 10].

A major challenge of both theoretical and practical interest is the resolution of the inverse problem – finding an IFS whose attractor is a target 2D shape [19, 29, 28, 4]. An exact solution can be found in some particular cases, but in general, no exact solution is known.

1.1 IFS representations

The IFS inverse problem problem can be formulated as an optimisation problem: some computational solutions exist, based on deterministic or stochastic optimisation methods. As the function to be optimised is extremely complex, some *a priori* restrictive hypotheses are necessary. Usually, the search space is that of affine IFS, with a fixed number of functions [5, 16, 27]. Solutions based on Evolutionary Algorithms have recently been presented for affine IFS [28, 11, 26, 21].

Some previous work [18] dealt with general non-affine IFS using Genetic Programming (GP), termed *mixed IFS*:

- such IFS are capable to create a wide variety of shapes,
- GP offers an easy representation for evolving general functions.

However, without other guideline than target shapes, functions defined by GP parse-trees are rarely contractive. Moreover, their fixed point needs to be numerically estimated.

This paper considers an alternative representation of non-affine IFS. Each function is represented in polar coordinates with respect to a central point. The term "Polar IFS" will be used to designate an IFS built on such functions. There are many advantages to polar representation:

- a simple constraint on the radial coordinate ensures the convergence towards the central point, which happens to be the fixed point of the function (see section 3);
- polar IFS can be represented as GP parse-trees with a simple wrapper; the associated inverse problem can hence be solved using GP;
- handling of contractance constraints is simpler than with mixed IFS, though the contractance still has to be checked numerically; however, the proportion of contractive IFS in the set of Polar IFS is far greater than in the set of Mixed IFS (see section 3.2). Hence, polar IFS provide a more efficient (less sparse) search space to the optimisation algorithm than Mixed IFS.

1.2 IFS evolution

All of the above-mentioned works dealing with the inverse problem have used what can be called a standard approach to evolve IFS: an individual is a fully fledged IFS, made of several functions —independently of the representation (polar or mixed). In this approach, the solution is the best individual of the final population. All individuals but one are discarded, which seems a great waste, as the elaboration of each of them has used the same amount of CPU-time as the winner.

An alternative approach has been proposed in the Classifier framework: in Classifier Systems (CS), the "Pittsburgh" approach [7, 25] individuals are complete rule bases, whereas in the "Michigan" approach [14, 13, 30] individuals are single rules, and the solution is built using several individuals (rules) of the population.

In this paper, the "Michigan" approach is transposed in the framework of IFS. We named it *individual approach* (section 4): single contractive functions are evolved using GP, and IFS are built using individuals (functions) from the population. The main difficulty of this approach is the definition of the partial fitness for a single function — the direct transposition of the Bucket Brigade algorithm in CS is impossible. Section 5 presents a problem-specific approach.

Finally, section 6 describes how *Polar IFS* and the *individual approach* are implemented to solve some instances of IFS inverse problem. Results on three images, together with comparisons with previous results from [18] are presented and acknowledge the power of the proposed approach.

2 Iterated Function Systems

This section briefly recalls the basis of IFS theory and the numerical algorithms most widely used to compute the attractors of an IFS.

2.1 Notations and definitions

Definition 1: Let (F, d) be a complete metric space, and $(w_i)_{i=1,..,N}$ be a collection of functions defined from F into F.

$\Omega = \{F, (w_i)_{i=1,..,N}\}$ is called an *IFS (Iterated Function System).*

A central notion in IFS theory is that of contractive mappings:

Definition 2: A mapping $w : F \to F$, from a metric space (F, d) into itself, is called *contractive* if there exists a positive real number $s < 1$ such that:

$$\forall (x, y) \in F^2, \quad d\big(w(x), w(y)\big) \leq s.d(x, y)$$

The smallest of such numbers s is called the *contraction ratio* of w.

A crucial result about contractive mappings is the following:

Theorem *Contractive Mapping Fixed Point Theorem:*
If (F, d) is a complete metric space, and $W : F \to F$ is a contractive mapping, then W has a unique fixed point.

All mappings can also be applied to subsets of F, and give the following:
Definition 3: An IFS $\Omega = \{F, (w_i)_{i=1,..,N}\}$ induces an operator W defined on the space of subsets of F by:

$$\forall K \subset F, \quad W(K) = \bigcup_{i \in [0,N]} w_i(K)$$

Definition 4: An IFS $\Omega = \{F, (w_i)_{i=1,..,N}\}$ is called *hyperbolic* (or *contractive*) if all functions w_i are contractive. The *contraction ratio* of Ω is the minimum of the contraction ratio of the w_i.

Proposition If an IFS $\Omega = \{F, (w_i)_{i=1,..,N}\}$ is contractive, there exists a unique set $A \subset F$, called the *attractor* of the IFS Ω, such that:

$$W(A) = A$$

The uniqueness of an attractor for contractive IFS is a result of the Contractive Mapping Fixed Point Theorem for the mapping W acting on $\mathcal{P}(F), d_H$, which is contractive according to the Hausdorff distance d_H:
Definition 5: The *Hausdorff distance* between two subsets A and B of F is defined by:

$$d_H(A, B) = \max\left(\max_{x \in A}\left(\min_{y \in B} d(x, y)\right), \max_{y \in B}\left(\min_{x \in A} d(x, y)\right)\right)$$

2.2 Computing the attractor

There are two main techniques to compute the attractor of a contractive IFS.

- **Deterministic method:**
  From any kernel S_0, build the sequence $\{S_n\}$ of subsets of F:

  $$S_{n+1} = W(S_n) = \bigcup_{i=1}^{N} w_i(S_n)$$

  For large values of n, S_n is an approximation of the actual attractor of Ω.

- **Stochastic method (toss-coin):**
  Let x_0 be the fixed point of one of the w_i functions. We build the sequence of points x_n as follows: $x_{n+1} = w_i(x_n)$, i being randomly chosen in $\{1..N\}$.
  Then $\bigcup_n x_n$ is an approximation of the real attractor of Ω. The larger n, the more precise the approximation.

2.3 Inverse problem

The inverse problem for 2D IFS can be stated as follows:

> *Find a contractive IFS the attractor of which is exactly a given shape (a binary image).*

However, this problem is generally relaxed into:

> *Find a contractive IFS the attractor of which is as close as possible of a given shape for a pre-defined distance function.*

A tool that is usually used for the simplification of the previous problem is the *Collage theorem*, which states that finding an IFS Ω whose attractor is close to a given shape I, is equivalent to minimising the distance $d_H\left(I, \bigcup_{i=1}^{N} w_i(I)\right)$ with the constraint that all w_i are contractive functions.

Collage theorem [4]
Let A be the attractor of the hyperbolic IFS $\Omega = \{F, (w_i)_{i=1,..,N}\}$, and λ the contraction ratio of Ω. Then:

$$\forall K \subset F, \quad d_H\left(K, W(K)\right) < \varepsilon \qquad \Rightarrow \qquad d_H(K, A) < \frac{\varepsilon}{1-\lambda},$$

However, some difficulties arise when $d_H\left(I, \bigcup_{i=1}^{N} w_i(I)\right)$ is to be minimised:

- The fitness depends on the contractance of the mappings; if one of the mappings is poorly contractive (i.e. λ close to 1), then the term $\frac{1}{1-\lambda}$ may become very large, and the bound thus becomes meaningless.

  In the case of affine IFS, it is possible to estimate λ and thus to minimise $\frac{1}{1-\lambda} d_H\left(I, \bigcup_{i=1}^{N} w_i(I)\right)$ to overcome this difficulty. However, for non-linear IFS, the contraction ratio may not be uniform over the domain F which makes it almost impossible to estimate.

- Computing the Hausdorff distance itself is CPU-time consuming. Moreover, the Hausdorff distance often is counter-intuitive: Figure 1 presents two pairs of shapes [(a), (b)] and [(a'), (b')] with d_H[(a), (b)] = d_H[(a'), (b')]. While (a) and (b) are perceived as similar, (a') and (b') look quite different.

These drawbacks led some authors of this paper [18] to consider a fitness function based on the toss-coin algorithm and on more intuitive distances between shapes (namely pixels differences or Euclidian distance), instead of the Hausdorff distance.

Section 5 will demonstrate how the "individual" approach allows one to use informations stemming from both collage theorem and toss-coin algorithm in order to solve the inverse problem.

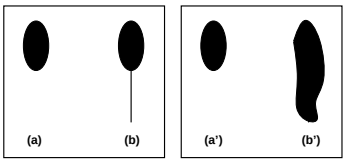


Figure 1: Hausdorff distance may be counter-intuitive. Here, d_H[(a), (b)] = d_H[(a'), (b')] though the shapes appear very different.

3 Polar IFS

This section introduces the Polar representation of non-affine IFS, and discusses the contractance issue. Experiments are presented to show the advantage of Polar IFS *versus* general non-affine IFS.

3.1 Polar representation

The main difficulty which arises when manipulating non-linear IFS is the handling of the contractance constraint. There is no general analytic way to check the contractance of a non-affine function, and experimental tests require heavy computations while only giving a hint on contractance. For instance, less that 15% of random functions actually are contractive (see [18], and also experiments of section 3.2). This empirical fact motivated the introduction of an alternative representation.

3.1.1 Local contractance

The first idea is to define a weaker contractance condition that will be easier to check:
Definition 6: A mapping $w : F \to F$, from a metric space (F, d) into itself, is called *locally contractive with respect to point* P if there exists a positive real number $s < 1$ such that:

$$\forall M \in F^2, \quad ||\overrightarrow{P\ w(M)}|| < s||\overrightarrow{PM}||$$

The smallest of such numbers s is called the *local contraction ratio* of w.

It is obvious that if w is locally contractive w.r.t. P, then P is the unique fixed point of w. Nevertheless, local contractance does not imply global contractance, as demonstrated by the following counter-example.

3.1.2 A counter-example

$$\begin{array}{lll}
\text{If} \quad x <= -1 & w_0(x,y) = \begin{pmatrix} 0.5x - 1 \\ 0.5y \end{pmatrix} \\
\text{If} \quad 1 <= x <= -1 & w_0(x,y) = \begin{pmatrix} 0.25(x^2 - 3x - 10) \\ 0.25(3+x)y \end{pmatrix} \\
\text{If} \quad x >= 1 & w_0(x,y) = \begin{pmatrix} x - 4 \\ y \end{pmatrix}
\end{array}$$

$$\begin{array}{lll}
\text{If} \quad x <= -1 & w_1(x,y) = \begin{pmatrix} x + 4 \\ y \end{pmatrix} \\
\text{If} \quad 1 <= x <= -1 & w_1(x,y) = \begin{pmatrix} 0.25(-x^2 - 3x + 10) \\ 0.25(3-x)y \end{pmatrix} \\
\text{If} \quad x >= 1 & w_1(x,y) = \begin{pmatrix} 1 + 0.5x \\ 0.5y \end{pmatrix}
\end{array}$$

Functions w_0 et w_1 are continuous. Their fixed points are respectively $P_0 = (-2, 0)$ and $P_1 = (2, 0)$, and they satisfy:

$$\forall X \in [-4,4] \times [-4,4], \quad d\big(P_n, w_n(X)\big) \leq s d(P_n, X) \quad \text{with} \quad s < 1$$

They are however not contractive and do not define a unique attractor for the $\{w_0, w_1\}$ IFS: there exists several sets, such that $A = W(A) = w_0(A) \cup w_1(A)$ (see figure 2) and the $W^n(X)$ sequences are not always convergent when $n \to \infty$ (see figure 3).

Though this counter-example seems to ruin all efforts to easily design contractive mappings to use to build IFS, the next sections will show that indeed, benefits can arise from using locally contractive functions in IFS.

3.1.3 GP representation and wrapper

When using GP trees to represent IFS, two approaches are possible to tackle the contractance requirement: transform GP trees into contractive mappings, or eliminate *a posteriori* non contractive mappings. The latter approach always works, but can be very time consuming, as many trees that are generated along evolution have to be eliminated (see [18]). This eugenistic approach was proposed because no cleaner way could be found to generate contractive mappings in two dimensions.

But, together with local contractance goes the idea of polar representation: if one considers a locally contractive mapping w with fixed point P, then points can be represented using (ρ, θ) coordinates centered on P [1], and w itself can thence be defined easily using

[1] i.e. $\vec{PM} = (\rho\cos(\theta), \rho\sin(\theta))$

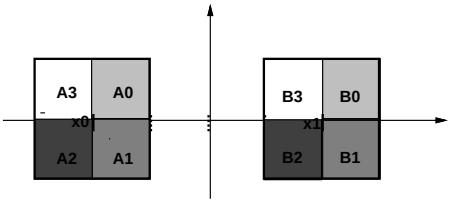


Figure 2: $W(A0 \cup B0) = A0 \cup B0$, $W(A1 \cup B1) = A1 \cup B1$, $W(A2 \cup B2) = A2 \cup B2$, $W(A3 \cup B3) = A3 \cup B3$, and $W(x_0 \cup x_1) = x_0 \cup x_1$.

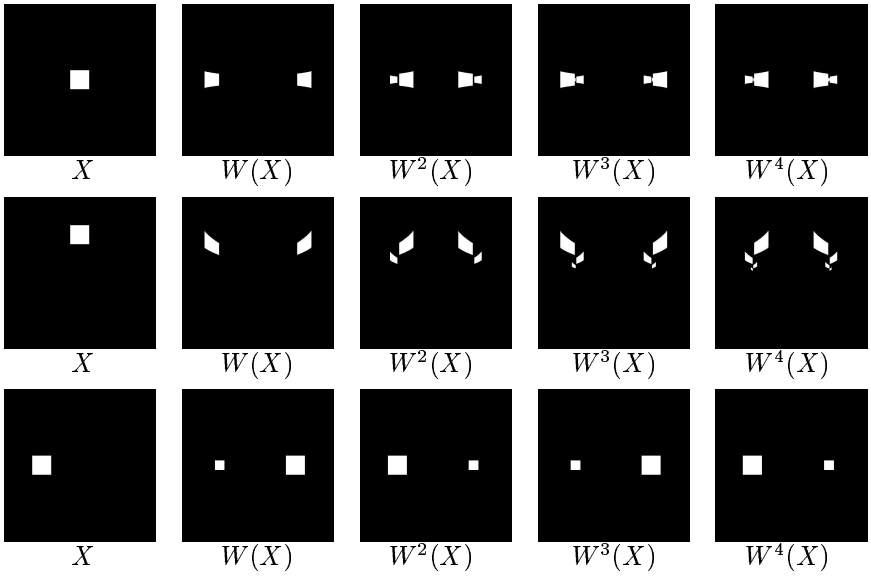


Figure 3: Three examples for which the sequence $W^n(X)$ diverges.

such coordinates. The important fact is that the local contractance condition becomes a contractance condition on the one-dimensional function giving ρ —and it is straightforward to transform a random one-dimensional function into a contractive function.

Definition 7: A Polar IFS is a set of locally contractive functions w_i with respective fixed points P_i such that each w_i is defined in polar coordinates w.r.t. P_i by two one-dimensional functions F_i and G_i by:

$$w_i \left(\begin{array}{c} \rho \\ \theta \end{array} \right)_{P_i} = \left(\begin{array}{c} \rho \dfrac{th(k_i * F_i(\rho,\theta)) + 1}{2} \\ G_i(\rho,\theta) \end{array} \right)_{P_i} \quad (1)$$

for some real number k_i.

The particular form of the ρ function ensures the local contractance —as $\frac{th(k_i * F_i(\rho,\theta))+1}{2} < 1$ whatever the value of F_i.

A locally contractive function can hence be represented by one point P in $[0,1]^2$ and two functions F and G, which can be evolved as GP trees. Equations (1) are used as wrapper, which ensures the local contractance of the mapping. Moreover, figure 4 shows that a wide variety of shapes can be obtained using that representation.

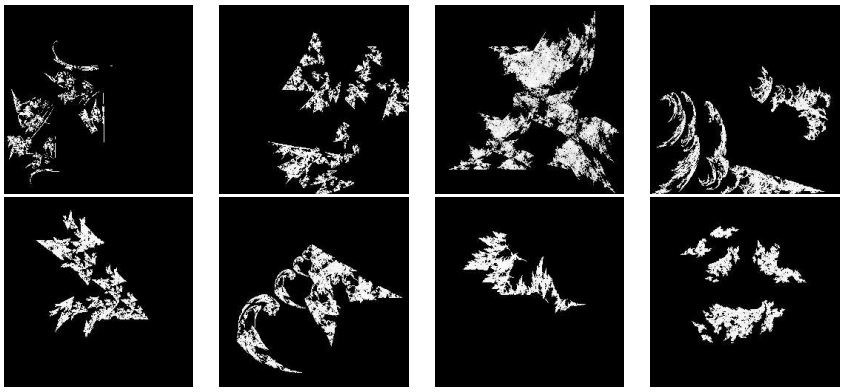


Figure 4: Examples of Polar IFS attractors

3.2 Contractance tests

The previous subsection highlighted the fact that contractance verification cannot be avoided for Polar IFS. However this section will experimentally show that the proportion of

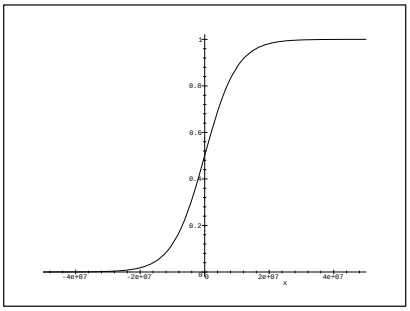


Figure 5: $y = \frac{th(kx)+1}{2}$ curve with $k = 10^{-7}$

contracting mappings in the set of locally contractive mappings defined by equation (1) is larger than for general mixed IFS. Hence the cost of contractance verification will be much lower for Polar IFS than for Mixed IFS.

This section presents some statistical tests that have been performed on two different sets of functions in order to experimentally approximate their respective proportion of contractive functions. The first set of functions is the set of mixed IFS, i.e. general non-linear IFS built from general random GP trees. The second set is the set of locally contractive functions defined by definition 7 whose components F and G are random GP trees. In both cases, the random trees are built from the basic nodes and terminals described in section 5.1.

As an analytical computation of the contraction ratio is generally out of reach, a numerical estimation of this ratio has to be done experimentally: for each mapping w, a sample of pairs of distinct points (P_i, Q_i) in $[0,1]^2$ is defined, and the minimum of $\frac{d\left(w(P_i),w(Q_i)\right)}{d(P_i,Q_i)}$ gives the approximate contraction ratio for w.

The cost of the approximation is of course proportional to the number of points in the sample: four tests with different samples have been tried on $n \times n$ images, and the results are presented in Tables 1, 2 and 3. Three tests are based on a pre-defined set of pairs (depending on the resolution of the image), while the fourth one selects points uniformly.

1. **All-pixels test** :
   Every pixel of the image is checked with every other pixel of the image. It is the most accurate test, but also the most CPU-consuming.

2. **Twist test** :
   This test starts from the two pixels at two opposite corners of the image, and each point scans symmetrically each half image, line by line, until the center pixel is reached (see figure 6).

3. **Square test** :
   All pixels of the image are tested against their immediate neighbour (see figure 7).

4. **Random test** :
   n pairs of points are selected randomly, uniformly in $[0,1]^2$.

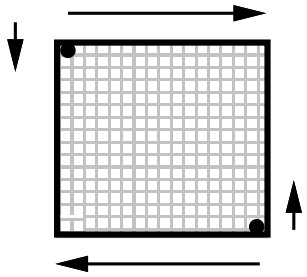


Figure 6: The Twist test.

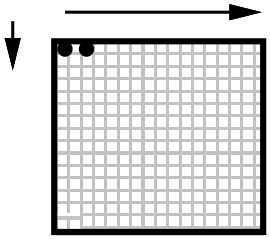


Figure 7: The Square test.

For some yet unclear reasons, the Twist test seems quite inefficient, while the Square test seems much more accurate (i.e. gives results closer to the All-pixels test) on Polar IFS than on mixed IFS. Hence the Square was found to realize a good compromise between efficiency and cost, and was used for all experiments presented in the following.

Tables 2 and 3 show two other major results :

1. The Square test results are not very far from the All-pixels test results for Polar IFS. Furthermore, the ratio of functions deemed contractive does not decrease drastically with the resolution of the images: less than 2.55% difference between 128×128 and 512×512. This means that the Square test can be considered as quite accurate, regardless of the image resolution above 128×128.

2. the proportion of contractive Polar IFS much larger than that of Mixed IFS : more than 50% compared to less than 10%.

4 Individual approach in evolutionary computation

The standard approach when using evolutionary methods as stochastic optimisers is to evolve a population of potential solutions to the problem at hand, each of them called individuals. The output of the algorithm is in that case the best individual encountered during the evolution. All other individuals are discarded, whatever information they might bear.

In some cases, however, potential solutions are lists of items, and an alternate possibility is to evolve a population of such items, and to build a solution by combining different items of the current population. Such an approach is very popular in Classifier Systems, where items are rules, and solutions are rule bases: The well-known Michigan approach evolves rules, and a solution is a rule base made of some of the best individuals of the final population

	# of w_i computed	# of comparisons
all pixels	n^2	$\frac{(n^2-1)(n^2-2)}{2}$
twist	n^2	$\frac{n^2}{2}$
square	n^2	n^2
random	$2n$	n

Table 1: Complexity of the various tests

n	# of w_i computed	all-pixels	twist	square	random
16	256	10.26	24.46	21.64	14.67
32	1024	10.46	24.16	21.80	13.73
64	4096	9.42	23.96	21.62	13.25
128	16384	-	24.03	22.08	13.39
256	65536	-	23.94	21.19	13.04
512	262144	-	23.91	21.18	-

Table 2: Proportion of contractive functions (%) for each test for *mixed* IFS

n	# of w_i computed	all-pixels	twist	square	random
16	256	62.48	90.56	64.94	74.29
32	1024	58.82	88.91	60.72	69.28
64	4096	56.12	88.08	57.91	64.30
128	16384	-	87.39	56.44	62.14
256	65536	-	87.24	55.26	60.00
512	262144	-	87.03	53.89	-

Table 3: Proportion of contractive functions (%) for each test for *polar* IFS

[14, 13, 30]. On the other hand, the so-called Pittsburgh approach evolves populations of complete rule bases [7, 25].

Similarly to the Michigan approach in CS, the *individual approach* proposed here consists in evolving a population of locally contractive functions, and to build an IFS by picking up functionsin the population.

Clearly, only problems where the solution can be set apart into separate components can be handled using an individual approach. However, another necessary condition is to be able to accurately evaluate the *local fitness* of a single component, i.e. its usefulness to the global fitness of the solution.

Fortunately, the IFS inverse problem offers some nice ways to evaluate the contribution of a single function of the system to its global performance: consider the target shape A; if an IFS (w_i) is a solution of the inverse problem, then $A = \cup w_i(A)$. Hence a local fitness for a w_i should consider positively the part of $w_i(A)$ lying inside A, and negatively the part of $w_i(A)$ lying outside A. Furthermore, the position of the fixed points also gives some indications. It should lie inside the target shape A, but the relative positions of the fixed points of two locally contractive functions also define semi-distances between individuals, which can be used to implement some sharing mechanism to keep diversity in the populations and prevent all functions in the populations to become similar.

5 Polar IFS + individual GP

5.1 GP components

As described in section 3.1, a locally contractive mapping is defined by two functions ($F(\rho, \theta)$ and $G(\rho, \theta)$) and a point P. Whereas P is simply represented by its two real-valued coordinates that do not need further description, F and G are modeled as GP-trees, similarly to Mixed IFS [18] .

Nodes and terminals: The trees are built from a set of terminals, which consist of the variables ρ and θ, constants in $[-1, 1]$, and a set of nodes, built from the following set of basic real-valued functions:

Unary nodes:[2]

- $-x$
- $\frac{1}{x}$ for $x \notin [-1, 1]$
- $\cos(x)$
- $\sin(x)$
- $th^{-1}(x)$
- $psqrt(x) = sign(x)\sqrt{|x|}$
- $plog(x) = sign(x)\log(|x|)$, if $x \neq 0$

Binary nodes:

- $+$
- $-$
- $\times$
- $\div$

Initialization: The initialization procedure for the GP trees is a simple recursive random choice into the joint set of node and terminals until either terminals are drawn on all branches or the maximum depth is reached.

As for the fixed points, they are randomly chosen among the contour points of the target shape A. This idea comes from the conjecture by Dekking [8] that there always exists solutions to the inverse problem where fixed points are on the edges of the target shape. This result has been proven in the case of affine IFS in [8].

Crossover: the standard GP crossover is used on both trees: it performs swaps of randomly selected sub-trees between the parents.

The fixed points are not modified by crossover.

Mutation : Different mutation operators are used, taking into account both the trees representing the F and G functions and P, the center of the locally contractive mapping at hand.

- No mutation acting on nodes has been retained, due to the too drastic effects of such a perturbation on the phenotypic behavior of the corresponding tree.
- For the same reason (too drastic effect), the only mutations of terminals finally retained are the mutation of the values of constant terminals and the mutation of variable terminals into constant terminals.
- *Constants*: the precise adjustment of constant terminals is critical when handling numerical trees (i.e. trees that represent a real-valued function). Whereas John Koza's seminal work considered only a finite number of possible values for constant terminal, relying on arithmetic to possibly generate any value (e.g. using $+$, $*$ and $\div$ it is possible

[2] $sign(x)$ returns the sign of x, i.e.: $sign(x) = 1$ if $x \geq 0$ $sign(x) = -1$ otherwise.

to approximate within a given precision any real value from integers in $[0, 10]$), more recent works consider that specific mutation operators are a better approach to the fine tuning of constant terminal values. In that perspective, constant terminal values are modified here by choosing a new value uniformly from a disk of fixed radius (another user-defined parameter).

Early experimental investigations highlighted the fact that constant terminals tend to disappear from the population. One possible explanation is that the numerical optimisation of the constant values is a more difficult task than the symbolic optimisation of the other nodes. The selection operator thus tends to rapidly eliminate functions having wrong constant values. This difference could come from the fact that the search spaces of the nodes and variables is finite while the search space of the constants is infinite. One way to tackle this difficulty is to implement an optimization loop for constant values as after the application of any variation operator (mutation or crossover). Though quite successful in other frameworks [24], this technique still needs to be tested in the IFS framework.

- *Variables*: mutations of variables considered here is either their transformation into constant terminals, the value of which is randomly drawn within $[0, 1]$, or their transformation into another variable.

- *Fixed points*: the mutation operator moves the fixed point of a contractive function by choosing another point uniformly in a neighborhood of radius 4 pixels from the parent fixed point, while staying on the boundary of the target shape.

A niching strategy is mandatory to prevent all individuals to rapidly become similar to the best covering mapping. Here, the dynamic niche sharing has been used [20]. This technique is based on a clusterisation of the populations with respect to a predefined inter-individuals distance. The selection operator is also adapted in order to fairly select a given percentage of best individual of each niche.

Table 5.1 summarizes the parameters used for the evolution. The implementation is based on the GALib library [1].

5.2 Fitness function for individuals

The fitness of an individual in the individual approach is divided into two main parts: the *local* fitness, computed from the characteristics of the individual itself, and the *global* fitness, that is based on a measurement of how a set of individuals actually solves the problem at hand. In the IFS inverse problem framework, the local fitness takes into account the relative positions of the attractor of a single contractive mapping and the target shape.

BINARY NODES	$+$, -, *, $\div$	
UNARY NODES	**neg, 1/., cos, sin, th^{-1}, psqrt, plog**	
POPULATION SIZE	see results, section 6	
OPERATORS	**Mutation probabilities**	
	constant $\rightarrow$ constant	0.15 according to a Gaussian law of variance SIGMA
	variable $\rightarrow$ constant	0.05 randomly chosen in $[-1, 1]$
	constant $\rightarrow$ variable	0.08
	variable $\rightarrow$ variable	0.08
	function $\rightarrow$ function (same arity)	0.08
	fixed points:	0.1, linearly decreasing along the generations.
	Crossover probability	
	PCROSS	0.95 for trees no crossover for fixed points
SELECTION	**Ranking**	
	selection pressure	1.35
REPLACEMENT	**Population replacement scheme**	
	replacement percentage	50% Overlapping populations
STOPPING CRITERION	**Based on approximation of target**	
		see section 5.3
WRAPPER	**Polar description of contractive mappings**	
		see section 3
SPECIAL FEATURES	**Dynamic niche sharing**	
	σ	0.05
	Max nb of clusters	0.5 of POP_SIZE

Table 4: Tableau for the evolution of polar IFS

5.2.1 First goal: $w_i(A)$ must lie inside A

Let $\#[A]$ be the number of pixels of $A \subset F$, and let us define $\mathcal{F}_1(w_i)$ as:

$$\mathcal{F}_1(w_i) = \frac{\#[w_i(A) \bigcap A]}{\#[w_i(A) \bigcap A] + \#[w_i(A) \backslash A]}$$

$\mathcal{F}_1(w_i)$ is maximum (and equals 1) iff $w_i(A) \subset A$.

5.2.2 Second goal: maximize the area of $w_i(A) \bigcap A$

Let us define $\mathcal{F}_2$, that corresponds to the maximization of the size of $w_i(A) \bigcap A$:

$$\mathcal{F}_2(w_i) = \frac{\#[w_i(A) \bigcap A]}{\#[A]}$$

$\mathcal{F}_2(w_i)$ is maximum (and equals 1) iff $A \subset w_i(A)$.

5.2.3 Integration of the contractance constraints

As described in section 3.2, the contractance test can be included in the computation of the image of the target $w_i(A)$. In the same time, the mean contraction ratio s_i can be estimated. If the function is not contractive, $\mathcal{F}_1$ and $\mathcal{F}_2$ defined above are not computed and are directly set to zero.

5.2.4 Local fitness

The local fitness is defined as a linear combination of $\mathcal{F}_1$, $\mathcal{F}_2$ and the distance to 1 of the estimated contraction ratio:

$$\mathcal{F}_{loc}(w_i) = \mathcal{F}_1(w_i) + \mathcal{F}_2(w_i) + (1 - s_i)$$

where s_i is the estimated contraction ratio of w_i. Due to the term $\mathcal{F}_2$, $w_i(A)$ tend to fill A, which is not satisfactory for an IFS. This effect is counterbalanced by the term $1 - s_i$ which forbids the trivial solution $w_i = Id$.

This fitness represents an interpretation of the collage property of an IFS, i.e.: one searches for the set of best w_i's such that $A = \bigcup w_i(A)$. This is yet another argument for the use of some niching mechanism in order to avoid that all individuals go to the same best coverage of the target shape. Moreover, a side effect might be an "economic" coverage of A, i.e. with the smallest possible $w_i(A) \bigcap w_j(A)$ for all i, j.

5.3 Global fitness and its repartition on individuals

A clusterisation of the current population with respect to a distance defined on the search space (the mean distance of the images of a set of sample points) yield to the set of the

N individuals to be globally evaluated. These N best individuals build an IFS Ω which represents a potential solution to the inverse problem at hand. A toss-coin algorithm is used in order to compute its attractor A_Ω, which is then compared to the target A using two quantities:

$$\begin{aligned} In_\Omega &= \frac{\#[A_\Omega \bigcap A]}{\#[A]} && \text{proportion of points of } A_\Omega \text{ within the target} \\ Out_\Omega &= \#[A_\Omega \backslash A] && \text{number of points of } A_\Omega \text{ without} \end{aligned}$$

The global fitness at generation n takes into account both the attractor of the current IFS $\Omega(n)$ and the attractor of the IFS constructed at generation $n-1$, $\Omega(n-1)$. A *global parsimony* term is added to favour solutions with smaller number of functions (nothing to do with usual GP parsimony that takes into account the size of the trees).

$$\begin{aligned} F_{glob}(n) = & \quad [In_{\Omega(n)} - In_{\Omega(n-1)}] - [Out_{\Omega(n)} - Out_{\Omega(n-1)}] \\ + & \quad \alpha[\text{Nb\_functions}(\Omega(n)) - \text{Nb\_functions}(\Omega(n-1))] \end{aligned}$$

For all results presented in section 6, the parsimony factor α was set to 0.075.

This global fitness is simply added to the individual fitness of the "active" individuals w_i of the population, according to their participation in the current IFS:

- if w_i just entered the IFS $\Omega(n)$ (it did not participate in $\Omega(n-1)$) then:
$$Fitness(w_i) = F_{loc}(w_i) + F_{glob}(n)$$
- if w_i was already present in $\Omega(n-1)$:
$$Fitness(w_i) = F_{loc}(w_i) + \frac{F_{glob}(n) + F_{glob}(n-1)}{2} \times \frac{1}{[age(w_i)]^2}$$
where $age(w_i)$ stands for the number of generations during which w_i has been part of the IFS.
- if w_i did just quit the IFS:
$$Fitness(w_i) = F_{loc}(w_i) - F_{glob}(n)$$
- if w_i does not belong to the IFS:
$$Fitness(w_i) = F_{loc}(w_i) + \frac{F_{glob}(n-1)}{2}$$
Its term corresponding to the global fitness decreases along generations.

The global contribution term distributed on each individual takes into account the past of this individual and its age with respect to the current IFS.

Termination criterion: F_{glob} is also used as a stopping criterion: the algorithm stops whenever the target is approximated within a fixed threshold, based on F_{glob} value.

5.4 Improving the toss-coin algorithm

The usual stochastic method known as *toss-coin* has been chosen here to actually compute the attractors, as it is acknowledged to be less CPU-consuming than the deterministic algorithm (see section 2.2). Nevertheless, the computation of the attractors is by far the most costly step of the whole inverse problem solving process. Hence, great care must be given to its actual implementation. The notion of *patience* has been introduced to cut down the computation time of unpromising evaluations of attractors with the toss-coin algorithm.

All calculations are done in the $[0,1] \times [0,1]$ unit square, discretized into a finite image (e.g. 512×512). The following possibilities might happen:

1. the attractor is almost uniformly spread within the target shape,
2. the attractor is almost uniformly spread across the 512×512 image,
3. the attractor lies mostly out of the target shape,
4. the attractor lies within a very small (e.g. 2×2 pixels) area.

IFS attractors have an incredible variety of shapes. Hence it is very unlikely that they should produce attractors of the first or second kind in the first generations.

Moreover, the number of pixels is finite. This means that the drawing speed (frequency of apparition of *new* pixels in the attractor) decreases along the iterations.

Adaptive optimization criterion Deciding once for all that the toss-coin routine should be iterated 10,000 times for instance is a very bad decision: if the attractor is of the third or fourth kind, a lot of cpu time is lost. On the other hand, 10,000 iterations may not be enough to draw a faithful representation of the attractor in the first two cases.

Optimizing the number of iterations is crucial. Ideally, one should stop iterating the toss-coin function as soon as the drawing speed of the attractor comes close to 0.

The notion of *patience* is introduced to adaptively adjust the number of iterations: suppose a patience of 1,000 iterations. If no newer pixel has been illuminated on the discretized image during the last 1,000 iterations, the algorithm stops – guessing that no significant amount of new pixels will come out of the toss-coin routine in the future.

Formally speaking, the value of the `Patience` parameter sets a threshold on the speed of occurrence of new pixels on the image.

The patience criterion automatically *adapts* to both the definition of the final image, and the required precision:

- On a reduced 64×64 image, each pixel represents a large subset of the $[0,1] \times [0,1]$ domain, and the toss-coin will rapidly stop;
- During the first generations, a very fine representation of the attractor is superfluous: one only needs to roughly know if the IFS lies out of the target shape or not. One

can then mischievously increase the value of the `Patience` variable along with the generations: in all experiments, the `Patience` is initially set to 50 and is incremented with every generation. A patience of 1,000 is used to produce the final image.

6 Results

6.1 Sample results

Figures 8 to 11 present some results of the proposed algorithm for some inverse IFS problems. The parameter settings are those of table 5.1.

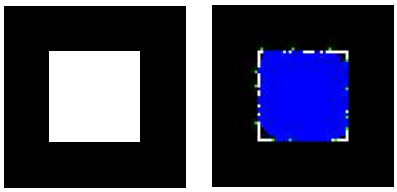
6.2 Comparative results

Comparison with previous approaches of the inverse problem for IFS is rather uneasy. Indeed, current improvements have come from two sources :

- the use of Polar IFS (i.e. a restriction of the search space) has provided an easier search space for the optimisation algorithm,
- the individual strategy for GP is a very particular way of handling the evolutionary optimization. Comparisons in terms of function evaluation with the classical approach is not very meaningful, as an evaluation in a classical evolutionary algorithm and in the individual approach do not represent the same thing with respect to the function to be optimised.

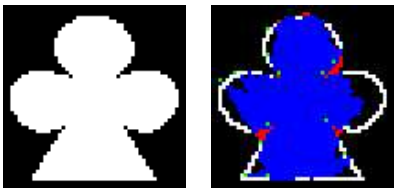
Figures 12 and 14 present results obtained on the square target. For performance comparisons in terms of CPU time, one can consider the number of GP tree evaluations. This is roughly equivalent to $Population\_Size \times Number\_of\_Generations + Number\_of\_Selected\_Individuals\_for\_a\_Gl$ $\times$ $Number\_of\_Generations$ as fas as the individual approach is concerned compared with 4 $\times$ $Number\_of\_Individuals \times Number\_of\_Generations$ for the classical approach.

- Figure 12 presents results obtained with a Genetic Algorithm for affine IFS (searching for a 4 functions IFS, i.e. for 24 real parameters) this result was obtained using approximatively 200,000 evaluations (a population size of 20 individuals during 10,000 generations). Figure 13 presents the result of an iterative implementation of the same algorithm where successive runs are made on more and more precise approximations of the target (best individuals of the previous run are included in the initial population of the next GA run). Fewer iterations were necessary in order to obtain similar results (around 55,000).
- Figure 14 presents results obtained with a Genetic Programming technique for mixed IFS (from [18]). These results were obtained using approximatively 45,000 evaluations (population size of 30 during 1,500 generations).



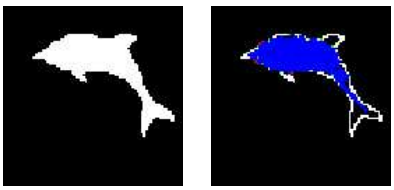
Target 49 generations
14 functions
pixels inside: 85% (blue)
pixels outside: 0% (red)

Figure 8: 64x64 target, with a population size of 60 individuals, the fixed points of each function are in green.



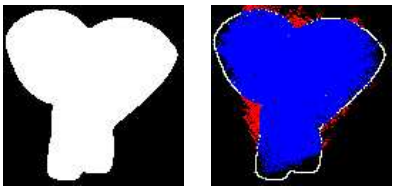
Target 60 generations
13 functions
pixels inside: 70% (blue)
pixels outside: 2.2% (red)

Figure 9: 50x50 target, with a population size of 100 individuals, the fixed points of each function are in green.



Target 400 generations
13 functions
pixels inside: 60% (blue)
pixels outside: 0.36% (red)

Figure 10: 80x80 Dolphin target, with a population size of 100 individuals, the fixed points of each function are in green.



Target

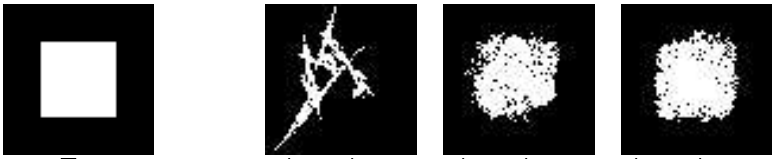
87 generations
31 functions
pixels inside: 84.52%
pixels outside: 5.03%

Figure 11: 128x128 target, with a population size of 200 individuals, the fixed points of each function are in green.

- Using an individual scheme on Polar IFS, only 2,940 evaluations were necessary to obtain results of figure 8.

Target | 10000 generations
4 functions
pixels inside: 88.4%
pixels outside: 17%

Figure 12: Classical GA for affine IFS on a 64x64 target, with a population size of 20 individuals.



Target | iteration 1
500 gen.
5 indiv.
inside: 31%
outside: 9% | iteration 2
1,500 gen.
10 indiv.
inside: 79%
outside: 10% | iteration 3
2,500 gen.
15 indiv.
inside: 88%
outside: 7%

Figure 13: Classical GA for affine IFS : iterative scheme for a 64x64 square (4 functions).

Figure 14: Genetic Programming for Mixed IFS on a 64x64 target. From left to right : target and best images of generations 10, 100, 300 and 1,500. Population size 30 individuals.

7 Conclusion and perspectives

This paper has introduced two original features in the framework of IFS inverse problem:

- on the fractal side, Polar IFS are an interesting model that simplifies the manipulation of non-linear IFS. Moreover, the very high proportion of contractive mappings in the set of locally contractive mappings, together with the simple wrapper transforming two general GP trees into a locally contractive mapping result in a very efficient search;
- on the evolutionary side, the IFS framework is another area where a Michigan approach is possible: single functions are evolved, and a subset of the population actually builds the solution to the inverse problem. The careful design of both the local and the global fitness functions, as well as their balanced aggregation into the final fitness used for selection, are crucial for the success of the method.

These first experiments lead us to think that making use of *a priori* information on the problem to solve, along with other methods can be quite efficient with the individual approach. Moreover an approximation of the target shape is obtained very rapidly while the representation of details needs more computation time.

Future work on this topic concern:

- cross-validation tests using the individual approach with mixed IFS representations, and the standard approach with the Polar IFS representation, in order to sort out the respective benefits of both original improvement introduced here;
- implementation of a individual GP technique in an interactive manner for artistic generation of fractal images;
- use of the Polar IFS representation as an alternative representation for mechanical structures, in the framework of topological optimum design [23].

Acknowledgements

The authors would like to thank Jean-Pierre Tillich for his help in the construction of the counter example of section 3, Jacques Lévy Véhel for the numerous discussions we had about inverse problem for IFS, and Laurent Balagué who helped us a lot in the final implementation of the techniques described in this paper.

References

[1] Galib: A c++ library of genetic algorithm components. http://lancet.mit.edu/ga/.

[2] M. Barnsley and S. Demko. Iterated function system and the global construction of fractals. *Proceedings of the Royal Society*, A 399:243–245, 1985.

[3] M. Barnsley, S. Demko, J. Elton, and Geronimo J. Invariant measures for Markov processes arising from iterated function systems with place-dependent probabilities. Georgia Tech. preprint.

[4] M. Barnsley, V. Ervin, D. Hardin, and J. Lancaster. Solution of an inverse problem for fractals and other sets. *Proc. Natl. Acad. Sci. USA*, 83, 1986.

[5] M. F. Barnsley. *Fractals Everywhere.* Academic Press,N Y, 1988.

[6] Pierre Collet, Evelyne Lutton, Frédéric Raynal, and Marc Schoenauer. Individual gp: an alternative viewpoint for the resolution of complex problems. In *GECCO99, Genetic and Evolutionary Computation Conference, July 13 - 17, 1999, Orlando, Florida, USA.*, 1999.

[7] K.A. DeJong. *The Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, Ann Harbor, 1975. *Dissertation Abstract International*, 36(10), 5140B. (University Microfilms No 76-9381).

[8] F.M. Dekking. *Fractal Image Coding : Some Mathematical Remarkd on Its Limits and Its Prospects*, chapter In 'Fractal Image Encoding and Analysis', pages 117–133. NATO ASI Series, Springer Verlag. Yuval Fisher, Ed, 1998. Series F : Computer and System Sciences, Vol. 159.

[9] J. H. Elton. An ergodic theorem for iterated maps. In *Georgia Tech. preprint*, 1986.

[10] Y. Fisher. Fractal image compression. In *Siggraph 92 course notes*, 1992.

[11] B. Goertzel. Fractal image compression with the genetic algorithm. *Complexity International*, 1, 1994.

[12] D. P. Hardin. *Hyperbolic Iterated Function Systems and Applications.* PhD thesis, Georgia Institute of Technology, 1985.

[13] J. Holland. Escaping brittleness : The possibilities of general purpose learning algorithms applied to parallel rule-based systems. In R.S Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: an artificial intelligence approach*, volume 2, pages 593–623. Morgan Kauffman, 1986.

[14] J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, 1975.

[15] J. Hutchinson. Fractals and self-similarity. *Indiana University Journal of Mathematics*, 30:713–747, 1981.

[16] E. W. Jacobs, Y. Fisher, and R. D. Boss. Fractal image compression using iterated transforms. In *Data Compression*, 1992.

[17] A.E. Jacquin. Fractal image coding: A review. *Proc. of the IEEE*, 81(10), 1993.

[18] E. Lutton, J. Lévy Véhel, G. Cretin, P. Glevarec, and C. Roll. Mixed ifs: resolution of the inverse problem using genetic programming. *Complex Systems*, 9:375–398, 1995. (see also Inria Research Report No 2631).

[19] G. Mantica and A. Sloan. Chaotic optimization and the construction of fractals : solution of an inverse problem. *Complex Systems*, 3:37–62, 1989.

[20] Brad L. Miller and Michael J. Shaw. Genetic algorithms with dynamic niche sharing for multimodal function optimization. Technical Report 95010, IlliGAL Report, December 1995.

[21] D. J. Nettleton and R. Garigliano. Evolutionary algorithms and a fractal inverse problem. *Biosystems*, 33:221–231, 1994. Technical note.

[22] Frédéric Raynal, Evelyne Lutton, Pierre Collet, and Marc Schoenauer. Manipulation of non-linear ifs attractors using genetic programming. In *CEC99, Congress on Evolutionary Computation, July 6-9, Washington DC. USA.*, 1999.

[23] M. Schoenauer. Representations for evolutionary optimization and identification in structural mechanics. In J. Périaux and G. Winter, editors, *Genetic Algorithms in Engineering and Computer Sciences*, pages 443–464. John Wiley, 1995.

[24] M. Schoenauer, M. Sebag, F. Jouve, B. Lamy, and H. Maitournam. Evolutionary identification of macro-mechanical models. In P. J. Angeline and Jr K. E. Kinnear, editors, *Advances in Genetic Programming II*, pages 467–488, Cambridge, MA, 1996. MIT Press.

[25] S.F. Smith. Flexible learning of problem solving heuristics through adaptive search. In A. Bundy, editor, *Proceedings of IJCAI83*, pages 422–425, 1983.

[26] L. Vences and I. Rudomin. Fractal compression of single images and image sequences using genetic algorithms., 1994. The Eurographics Association.

[27] J. Lévy Véhel. *Analyse et synthèse d'objets bi-dimensionnels par des méthodes stochastiques*. PhD thesis, Université de Paris Sud, Décembre 1988.

[28] E. R. Vrscay. *Fractal Geometry and Analysis*, chapter Iterated function Systems: theory, applications and the inverse problem, pages 405–468. 1991.

[29] R. Vrscay. Moment and collage methods for the inverse problem of fractal construction with iterated function systems. In *Fractal 90 Conference*, 1990. Lisbonne, June 6-8.

[30] S.W. Wilson and D.E. Goldberg. A critical review of classifier systems. In J. D. Schaffer, editor, *Proceedings of the* 3^{rd} *International Conference on Genetic Algorithms*, pages 244–255. Morgan Kauffman, 1989.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399