Curve fitting by fractal interpolation

Polychronis Manousopoulos, Vassileios Drakopoulos and Theoharis Theoharis

Department of Informatics and Telecommunications,
University of Athens, Panepistimioupolis, 157 84, Athens, Greece.
{polyman, vasilios, theotheo}@di.uoa.gr

Abstract. Fractal interpolation provides an efficient way to describe data that have an irregular or self-similar structure. Fractal interpolation literature focuses mainly on functions, i.e. on data points linearly ordered with respect to their abscissa. In practice, however, it is often useful to model curves as well as functions using fractal intepolation techniques. After reviewing existing methods for curve fitting using fractal interpolation, we introduce a new method that provides a more economical representation of curves than the existing ones. Comparative results show that the proposed method provides smaller errors or better compression ratios.

Key words: fractal interpolation, curve fitting, iterated function systems.

1 Introduction

Fractal interpolation has been developed as an alternative interpolation technique suitable for capturing data with inherent fractal structure, i.e. details at different scales or some degree of self-similarity. In contrast to traditional interpolation, which is built on elementary functions such as polynomials, fractal interpolation is based on the theory of iterated function systems producing interpolants that are convenient for fitting physical or experimental data.

Fractal interpolation literature focuses on functions, i.e. the data points are linearly ordered with respect to their abscissa and the interpolant is a function of (usually) non-integral dimension. This is often sufficient, e.g. when interpolating time series data. In practice, however, there are many cases where the data are suitable for fractal interpolation but define a curve rather than a function, e.g. when modelling coastlines or plants. So, it is useful to extend fractal interpolation to include curves as well as functions, an issue not fully addressed so far. Methods based on generalizations to higher dimensions are introduced in [1], [2] and [3]. The use of index coordinates is suggested in [4]. Non-affine fractal interpolation is employed in [5]. Various combinations of IFS models and free form curves are proposed in [6] and [7]. A method of data fitting by means of fractal interpolation functions is proposed in [8]. An interpolation method for multifractal structures is presented in [9].

In this paper we review existing approaches in this area and introduce a new method for curve fitting by fractal interpolation. Our motivation is to create a method that is more accurate and economical than the existing ones, thus being more suitable for practical applications such as shape representation. All methods are compared in practical applications showing the advantages of the proposed FCF method in terms of either accuracy or compression ratio. The paper is structured as follows. In Sect. 2 we present the necessary background on fractal interpolation functions. Section 3 contains existing appoaches to curve fitting by fractal interpolation, while Sect. 4 introduces the new method. Section 5 contains the application of our method in various practical cases and comparisons against previous approaches. Finally, Sect. 6 presents our conclusions and indicates areas of future work.

2 Fractal Interpolation Functions

Fractal interpolation functions as defined in [10] and [11] are based on the theory of *iterated function systems*. An iterated function system (IFS), denoted by $\{X; w_n, n = 1, 2, \ldots, N\}$, consists of a complete metric space (X, ρ), e.g. $(\mathbb{R}^n, ||\cdot||)$ or a subset, and a finite set of continuous mappings $w_n\colon X \to X$, $n = 1, 2, \ldots, N$. If w_n are contractions with respective *contractivity factors* s_n, $n = 1, 2, \ldots, N$, the IFS is termed *hyperbolic*. The transformation $W\colon \mathcal{H}(X) \to \mathcal{H}(X)$ with $W(B) = \cup_{n=1}^{N} w_n(B)$, where $\mathcal{H}(X)$ denotes the metric space of nonempty compact subsets of X with respect to the Hausdorff metric, has a unique fixed point $A_\infty = W(A_\infty) = \lim_{n\to\infty} W^n(B)$ for every $B \in \mathcal{H}(X)$, which is called the *attractor* of the IFS.

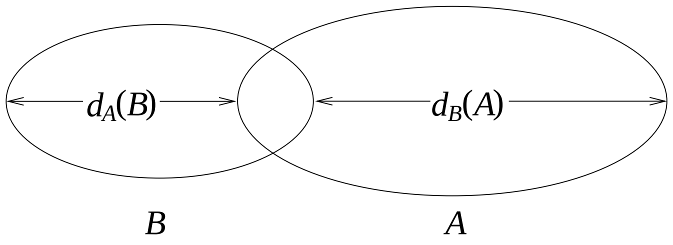

Fig. 1. The difference between $d_A(B)$ and $d_B(A)$.

The *Hausdorff distance* between the points A and B of $\mathcal{H}(X)$ is given by

$$h(A, B) = \max\{d_A(B), d_B(A)\},$$

where $d_B(A) = \max\{d(x, B) : x \in A\}$ and $d_A(B) = \max\{d(x, A) : x \in B\}$ (Fig. 1). The function $d_A(B)$ sometimes is called the *directed Hausdorff distance* from A to B. Because of the sensitivity of the Hausdorff metric to noise or isolated points that stems from its 'worst-case' nature, some Hausdorff-like metrics have been proposed such as the *Modified Hausdorff Distance (MHD)* (see [12]).

Specifically,

$$h_{MHD}(A,B) = \max\{d_{MHD}(A,B), d_{MHD}(B,A)\}$$

where $d_{MHD}(A,B) = (1/N_a)\Sigma_{a\in A} d(a,B)$, N_a denotes the number of points in A and $d(a,B)$ is the usual point to set distance.

2.1 Fractal Interpolation Functions in the Plane

Let us represent the given set of *data points* as $\{(u_m, v_m) \in \mathbb{R}^2 \colon m = 0, 1, \ldots, M\}$. In general, the interpolation is applied to a subset of them, the *interpolation points*, represented as $\{(x_i, y_i) \in \mathbb{R}^2 \colon i = 0, 1, \ldots, N\}$. Both sets are linearly ordered with respect to their abscissa, i.e. $u_0 < u_1 < \cdots < u_M$ and $u_0 = x_0 < x_1 < \cdots < x_N = u_M$. The interpolation points partition the set of data points into *interpolation intervals* and may be chosen equidistantly or not. The greater the number of interpolation points the better the fit of the data, but more interpolation points result in a smaller compression ratio since more information is required to describe the interpolation function.

Let $\{\mathbb{R}^2; w_n, n = 1, 2, \ldots, N\}$ be an IFS with affine transformations

$$w_n \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_n & 0 \\ c_n & s_n \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d_n \\ e_n \end{bmatrix}$$

constrained to satisfy

$$w_n \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} x_N \\ y_N \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$$

for every $n = 1, 2, \ldots, N$. Solving the above equations results in

$$\begin{aligned} a_n &= \frac{x_n - x_{n-1}}{x_N - x_0} \\ d_n &= \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0} \\ c_n &= \frac{y_n - y_{n-1}}{x_N - x_0} - s_n \frac{y_N - y_0}{x_N - x_0} \\ e_n &= \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n \frac{x_N y_0 - x_0 y_N}{x_N - x_0}, \end{aligned}$$

i.e. the real numbers a_n, d_n, c_n, e_n are completely determined by the interpolation points, while the s_n are *free* parameters of the transformations satisfying $|s_n| < 1$, in order to guarantee that the IFS is hyperbolic with respect to an appropriate metric. The transformations w_n are *shear transformations*: line segments parallel to the y-axis are mapped to line segments parallel to the y-axis contracted by the factor $|s_n|$. For this reason, the s_n are called *vertical scaling* (or *contractivity*) *factors*.

It is well known (see for example [11]) that the attractor $G = \bigcup_{n=1}^{N} w_n(G)$ of the aforementioned IFS is the graph of a continuous function $f \colon [x_0, x_N] \to \mathbb{R}$

that passes through the interpolation points. This function is called *fractal interpolation function (FIF)* corresponding to these points. It is a *self-affine* function since each affine transformation w_n maps the entire (graph of the) function to its *section*, i.e. function values between the interpolation points (x_{n-1}, y_{n-1}) and (x_n, y_n) for all $n = 1, 2, \ldots, N$. For example, let $\{(0,0), (0.4, 0.5), (0.7, 0.2), (1, 0)\}$ be a given set of data points. Figure 2 shows the graph of an affine FIF with $s_1 = 0.5$, $s_2 = -0.2$ and $s_3 = 0.4$.

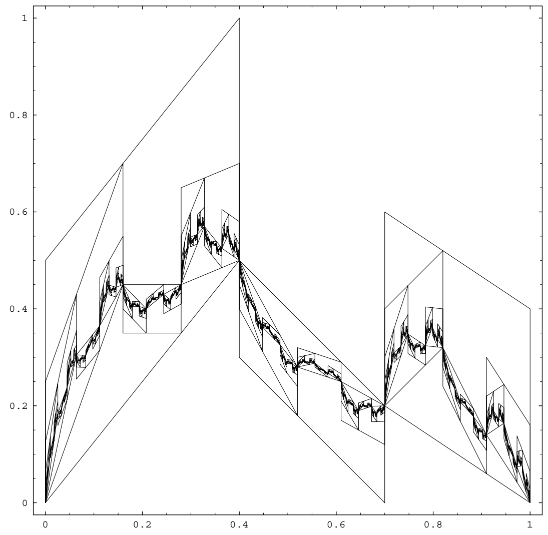

Fig. 2. The construction of an affine FIF starting from the unit square.

The graph of a FIF is bounded by the rectangle $[x_0, x_N] \times [a, b]$ if the vertical scaling factors s_n satisfy $s_n^{min} \le s_n \le s_n^{max}$ and $|s_n| < 1$, where

$$s_n^{min} = \max\left\{\frac{a - y_{n-1}}{b - y_0}, \frac{a - y_n}{b - y_N}, \frac{b - y_{n-1}}{a - y_0}, \frac{b - y_n}{a - y_N}\right\}$$

$$s_n^{max} = \min\left\{\frac{b - y_n}{b - y_N}, \frac{b - y_{n-1}}{b - y_0}, \frac{a - y_n}{a - y_N}, \frac{a - y_{n-1}}{a - y_0}\right\}$$

for every $n = 1, \ldots, N$ (see [5], [13]).

Although the FIF passes by definition through the interpolation points, this is not necessarily the case for the remaining data points $\{(u_m, v_m)\} \setminus \{(x_i, y_i)\}$. The accuracy of fit can be measured as the squared error between the ordinates of the original and the reconstructed points

$$\varepsilon = \sum_{m=0}^{M} (v_m - G(u_m))^2 \tag{1}$$

or, alternatively, as the *Hausdorff distance* between the two sets

$$\varepsilon = h(\{(u_m, v_m)\}, G).$$

The vertical scaling factors of a FIF are usually chosen so as to minimize such an error measure. For example, in [14] and [15] the minimization of (1) is achieved by algebraic or geometric methods. Moreover in [15], a greedy algorithm for finding some proper (but not necessarily globally optimal) interpolation points is presented.

An extension of the FIFs are the so-called *piecewise self-affine FIFs* (see [15]), which are essentially an application of the *recurrent IFSs* (see [11]). Their motivation is the fact that data often present self-affinity in subintervals and not in their whole length. This is modelled by the introduction of the *address points*, represented as $\{(\tilde{x}_{n,1}, \tilde{y}_{n,1}), (\tilde{x}_{n,2}, \tilde{y}_{n,2}) \in \mathbb{R}^2\colon n = 1, 2, \ldots, N\}$, that define the intervals of self-affinity. The affine transformations $w_n, n = 1, 2, \ldots, N$ are then constrained to satisfy

$$w_n \begin{bmatrix} \tilde{x}_{n,1} \\ \tilde{y}_{n,1} \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} \tilde{x}_{n,2} \\ \tilde{y}_{n,2} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix}.$$

Piecewise self-affine FIFs are more flexible than affine FIFs, but require the additional cost of determining the address points. Moreover, a greedy algorithm for locating both some proper (but not necessarily globally optimal) interpolation and address points is presented in [15].

2.2 Generalized Fractal Interpolation Functions

The FIF model described in the previous section can be extended to higher dimensions, producing functions that interpolate points in $\mathbb{R}^k$. Let $\{p_m \in \mathbb{R}^k\colon m = 0, 1, \ldots, M\}$ be the set of data points and $\{q_i \in \mathbb{R}^k\colon i = 0, 1, \ldots, N\}$ the set of interpolation points. Both sets are again assumed to be linearly ordered with respect to their abscissa. Let $\{\mathbb{R}^k; w_n, n = 1, 2, \ldots, N\}$ be an IFS with affine transformations

$$w_n \begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^k \end{bmatrix} = \underbrace{\begin{bmatrix} a_n & 0 & \cdots & 0 \\ c_n^1 & s_n^{1,1} & \cdots & s_n^{1,k-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_n^{k-1} & s_n^{k-1,1} & \cdots & s_n^{k-1,k-1} \end{bmatrix}}_{k \times k} \underbrace{\begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^k \end{bmatrix}}_{k \times 1} + \begin{bmatrix} d_n^1 \\ d_n^2 \\ \vdots \\ d_n^k \end{bmatrix}$$

constrained to satisfy

$$w_n(q_0) = q_{n-1} \quad \text{and} \quad w_n(q_N) = q_n$$

for every $n = 1, 2, \ldots, N$. The real numbers a_n, c_n^i, d_n^j for every $n = 1, \ldots, N$, $i = 1, 2, \ldots, k-1$ and $j = 1, 2, \ldots, k$ are completely determined by the interpolation points by solving the above equations, while the $s_n^{i,j}$, $i, j = 1, 2, \ldots, k-1$ are free parameters of the transformations chosen such that the contractivity factor s_n of the matrix (called *contractivity matrix*)

$$\begin{bmatrix} s_n^{1,1} & \cdots & s_n^{1,k-1} \\ \vdots & \ddots & \vdots \\ s_n^{k-1,1} & \cdots & s_n^{k-1,k-1} \end{bmatrix}$$

has modulus less than unity, in order to guarantee that the IFS is hyperbolic with respect to an appropriate metric. The exact values of $s_n^{i,j}$ can be determined by minimizing an error measure as in the planar case (see e.g. [2]) .

The attractor $G = \bigcup_{n=1}^{N} w_n(G)$ of the IFS is the graph of a continuous function $f\colon [q_0^1, q_N^1] \to \mathbb{R}^{k-1}$ that interpolates the points q_i, $i = 0, 1, \ldots, N$ (see [11]). It is a self-affine FIF in $\mathbb{R}^k$; however, its orthogonal projections to $\mathbb{R}^2$ are not necessarilly self-affine. The accuracy of fit of a Generalized FIF can be defined similarly to the planar case.

For example, in $\mathbb{R}^3$ we have the IFS $\{\mathbb{R}^3; w_n, n = 1, 2, \ldots, N\}$ with affine transformations

$$w_n \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_n & 0 & 0 \\ c_n^1 & s_n^{1,1} & s_n^{1,2} \\ c_n^2 & s_n^{2,1} & s_n^{2,2} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} d_n^1 \\ d_n^2 \\ d_n^3 \end{bmatrix}$$

constrained to satisfy

$$w_n \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} x_{n-1} \\ y_{n-1} \\ z_{n-1} \end{bmatrix} \quad \text{and} \quad w_n \begin{bmatrix} x_N \\ y_N \\ z_N \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix}$$

for $n = 1, 2, \ldots, N$. Solving the above equations results in

$$\begin{aligned}
a_n &= \frac{x_n - x_{n-1}}{x_N - x_0} \\
d_n^1 &= \frac{x_N x_{n-1} - x_0 x_n}{x_N - x_0} \\
c_n^1 &= \frac{y_n - y_{n-1}}{x_N - x_0} - s_n^{1,1} \frac{y_N - y_0}{x_N - x_0} - s_n^{1,2} \frac{z_N - z_0}{x_N - x_0} \\
c_n^2 &= \frac{z_n - z_{n-1}}{x_N - x_0} - s_n^{2,1} \frac{z_N - z_0}{x_N - x_0} - s_n^{2,2} \frac{z_N - z_0}{x_N - x_0} \\
d_n^2 &= \frac{x_N y_{n-1} - x_0 y_n}{x_N - x_0} - s_n^{1,1} \frac{x_N y_0 - x_0 y_N}{x_N - x_0} - s_n^{1,2} \frac{x_N z_0 - x_0 z_N}{x_N - x_0} \\
d_n^3 &= \frac{x_N z_{n-1} - x_0 z_n}{x_N - x_0} - s_n^{2,1} \frac{x_N y_0 - x_0 y_N}{x_N - x_0} - s_n^{2,2} \frac{x_N z_0 - x_0 z_N}{x_N - x_0}
\end{aligned}$$

i.e. the real numbers $a_n, d_n^1, c_n^1, c_n^2, d_n^2, d_n^3$ are completely determined by the interpolation points, while $s_n^{1,1}, s_n^{1,2}, s_n^{2,1}, s_n^{2,2}$ are *free* parameters of the transformations chosen such that the contractivity factor of the matrix

$$\begin{bmatrix} s_n^{1,1} & s_n^{1,2} \\ s_n^{2,1} & s_n^{2,2} \end{bmatrix}$$

has modulus less than unity.

For example, let $\{(0,2,1), (1,4,3), (2,8,5), (3,6,2), (4,5,6), (5,2,4), (6,3,7), (7,4,4), (8,2,3), (9,1,2)\}$ be a given set of data points in $\mathbb{R}^3$. Figure 3 shows the graph of a Generalized FIF with $s_n^{1,1} = s_n^{1,2} = s_n^{2,1} = s_n^{2,2} = 0.1$. The concept of

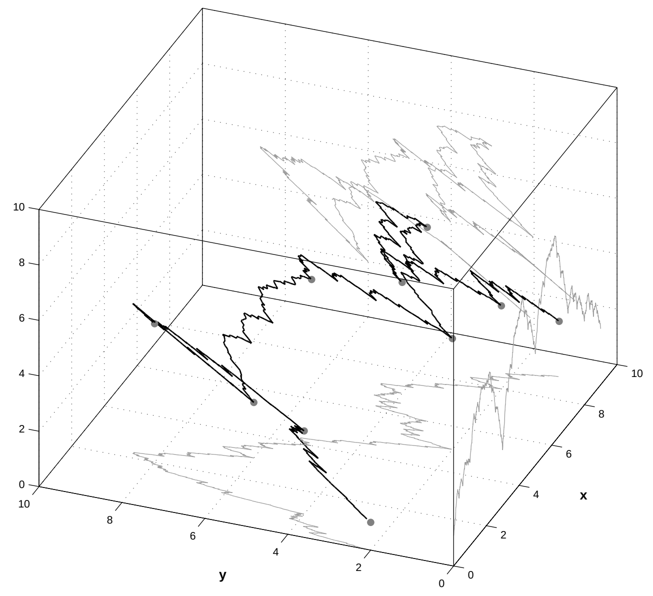

Fig. 3. A Generalized FIF in $\mathbb{R}^3$. The projections of the Generalized FIF on the xy, xz and yz planes are depicted in gray.

piecewise self-affine FIFs in $\mathbb{R}^2$ can be similarly extended to higher dimensions.

Hidden Variable Fractal Interpolation Functions The Generalized FIFs can also be used for interpolating points in $\mathbb{R}^2$. The idea is to extend the data to a higher-dimensional space, interpolate them by a Generalized FIF and project it back to $\mathbb{R}^2$ to obtain a function that interpolates the original data. Specifically, we apply to the interpolation points the mapping $T\colon \mathbb{R}^2 \to \mathbb{R}^3$ with $(x_i, y_i) \mapsto (x_i, y_i, H_i)$, $i = 0, 1, \ldots, N$, where the H_i are freely chosen. The new set of points $(x_i, y_i, H_i), i = 0, 1, \ldots, N$ is the generalized set of data corresponding to the original points and is interpolated by creating an IFS in $\mathbb{R}^3$ as described in the previous section. The attractor $G' = \bigcup_{n=1}^{N} w_n(G')$ of the IFS is the graph of a continuous function $f'\colon [x_0, x_N] \to \mathbb{R}^2$ that interpolates the points (x_i, y_i, H_i), $i = 0, 1, \ldots, N$. The orthogonal projection of the attractor to $\mathbb{R}^2$, defined by $P_H\colon G' \to G$ with $(x, y, H) \mapsto (x, y)$, is the graph of a continuous function $f\colon [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$. The extra coordinate H_i is called *hidden variable* and can be used to adjust the shape of the resulting interpolation function f which is thus called *hidden variable fractal interpolation function (HVFIF).* Note that although the attractor G' is self-affine, this is not necessarily the case for its projection G.

The hidden variable FIFs can be extended by introducing more than one hidden variables, having thus more free parameters in order to adust the shape of the resulting interpolation function.

3 Existing Applications of FIFs to Curve Fitting in the Plane

When the interpolation points define a curve rather than a function, i.e. they are not linearly ordered with respect to their abscissa, the direct use of a fractal interpolation function is not possible. In order to construct an IFS whose attractor interpolates the given points, and is therefore a curve, we can transform or extend the original points such that the application of a FIF is possible. This is then transformed or projected back to the plane to obtain a curve that interpolates the original points.

3.1 Curves as projections of Generalized FIFs

One possibility is to extend the idea of hidden-variable FIFs ([1], [2], [3]). We transform the original set of points to a higher-dimensional set that defines a function, then create the respective FIF and project it back to $\mathbb{R}^2$ to obtain a curve that interpolates the original points.

Let $\{(x_i, y_i) \in \mathbb{R}^2 \colon i = 0, 1, \ldots, N\}$ be the set of interpolation points. These points do not define a function but a curve on the xy-plane, i.e. it is not necessarily $x_i < x_j$ for $i < j$. We apply the transformation $T \colon \mathbb{R}^2 \to \mathbb{R}^3$ with $(x_i, y_i) \mapsto (t_i, x_i, y_i)$, $i = 0, 1, \ldots, N$, where the introduced index coordinates t_i satisfy $t_0 < t_1 < \cdots < t_N$; usually we set $t_i = i$. The new set of points $(t_i, x_i, y_i), i = 0, 1, \ldots, N$ is the *generalized set* corresponding to the original points and defines a function. We create a FIF that interpolates the generalized set as described in Sect. 2.2. The attractor $G' = \bigcup_{n=1}^{N} w_n(G')$ of the IFS is the graph of a continuous function $f' \colon [t_0, t_N] \to \mathbb{R}^2$ that interpolates the generalized set of points (t_i, x_i, y_i), $i = 0, 1, \ldots, N$. The projection of the attractor to $\mathbb{R}^2$, defined by $P_t \colon G' \to G$ with $(t_i, x_i, y_i) \mapsto (x_i, y_i)$, is the graph of a continuous curve $f \colon [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$ and is thus called *fractal interpolation curve (FIC)* [1]. Note that although the attractor G' is self-affine, its projection G is not necessarily self-affine.

The FIC defined above is open, assuming that the first and last interpolation points are different. In order to construct a closed fractal interpolation curve, we append to the original points an additional one that is the same as the first, i.e. we add $(x_{N+1}, y_{N+1}) = (x_0, y_0)$. The curve is afterwards constructed in the same way. A FIC of this kind is depicted in Fig. 4(a), which is constructed on a simple, manually selected set of 10 interpolation points. Specifically, the data points $\{(3, 1),\ (2, 2),\ (1, 4),\ (0, 3),\ (-1, 3),\ (-2, 1),\ (-1, -1),\ (0, -2),\ (2, -1),$

$(3.5, -0.5)\}$ have been used and the contractivity factors s_n have been set to 0.1.

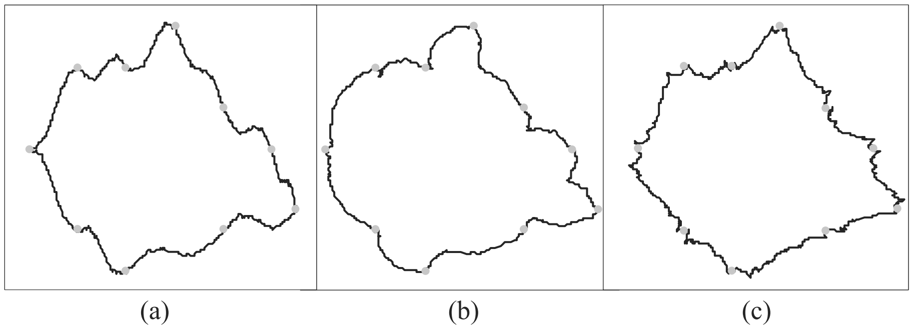
Fig. 4. (a) A FIC constructed by projecting a Generalized FIF. (b) A FIC constructed by coordinate separation. (c) A polar FIF. All three interpolation curves have been constructed from the same ten interpolation points (depicted in grey) using predefined vertical scaling factors.

3.2 Curves by Coordinate Separation

A similar way to construct a FIC involves the introduction of index coordinates without generalization to a higher-dimensional space ([4]). Specifically, we split the original set of points into two new sets by introducing an index for each coordinate. Then a fractal interpolation function is constructed for each new set, and these are finally combined in a single curve that interpolates the original points.

As previously, let $\{(x_i, y_i) \in \mathbb{R}^2\colon i = 0, 1, \ldots, N\}$ be the set of interpolation points. We apply the transformations $T_1\colon \mathbb{R}^2 \to \mathbb{R}^2$ with $(x_i, y_i) \mapsto (t_i, x_i)$, $i = 0, 1, \ldots, N$ and $T_2\colon \mathbb{R}^2 \to \mathbb{R}^2$ with $(x_i, y_i) \mapsto (t_i, y_i)$, $i = 0, 1, \ldots, N$, where the introduced index coordinates t_i satisfy $t_0 < t_1 < \cdots < t_N$; usually we set $t_i = i$.

Then, we create a fractal interpolation function for each of the two sets in the way described in Sect. 2.1. Let $G^x = (t_i^x, x_i)$ and $G^y = (t_i^y, y_i)$ be the attractors of the respective IFS. We can merge G^x and G^y in order to obtain $G = (x_i, y_i)$ which is the graph of a continuous curve $f\colon [x_0, x_N] \to \mathbb{R}$ that interpolates the points (x_i, y_i), $i = 0, 1, \ldots, N$ and is thus called fractal interpolation curve (FIC). Note that although the attactors G^x and G^y are self-affine, this is not necessarilly the case for G.

We can construct a closed curve, as previously, by appending to the original points an additional one that is the same as the first. A FIC of this kind is depicted in Fig. 4(b), where the same interpolation points and contractivity factors as in Fig. 4(a) have been used.

3.3 Curves as Polar Fractal Interpolation Functions

If the data points that define the curve are ordered by angle in their polar form, we can interpolate them using a class of non-affine FIFs, the polar FIFs (see [5]). Specifically, let $\{(x_i, y_i) \in \mathbb{R}^2 \setminus \{(0,0)\};\ i = 0, 1, \ldots, N-1\}$ be the set of interpolation points and $(r_i, \theta_i) \in (0, \infty] \times [0, 2\pi)$, $i = 0, 1, \ldots, N-1$ be their representation in polar coordinates obtained by the transformations $x = r\cos\theta$ and $y = r\sin\theta$. We assume that at least one point (x_i, y_i) exists in each quadrant and that it is $0 = \theta_0 < \theta_1 < \cdots < \theta_{N-1} < \theta_N = 2\pi$, i.e. the points define a function in the polar plane.

We create a FIF for the points $(r_i, \theta_i), i = 0, 1, \ldots, N$, where $(r_N, \theta_N) = (r_0, \theta_0)$ and we transform it back to the $xy-$plane to obtain a closed curve that interpolates the points (x_i, y_i). This FIF is called *polar fractal interpolation function*. Note that this curve is not self-affine since we have used a non-affine (polar) transformation. A polar FIF is depicted in Fig. 4(c), where the same interpolation points and contractivity factors as in Fig. 4(a) have been used.

4 Fractal Interpolation Curves in the Plane

We introduce a new method for creating fractal interpolation curves (FICs) in the plane without using index coordinates or generalizing to a higher-dimensional space. Our motivation is to create a more compact representation of curves using fewer parameters, as will be analyzed in the next section. We apply a reversible transformation to the data points in order to define a function in the plane. Then a FIF is constructed as usual and its attractor is transformed back to the original coordinates in order to obtain a curve that interpolates the original points.

Let us represent the given set of data points as $\{(u_m, v_m) \in \mathbb{R}^2 : m = 0, 1, \ldots, M\}$ and the set of interpolation points as $\{(u_{\mathbb{J}(i)}, v_{\mathbb{J}(i)}) \in \mathbb{R}^2 : i = 0, 1, \ldots, N\}$, where the labelling function $\mathbb{J}\colon \{0, 1, \ldots, N\} \to \{0, 1, \ldots, M\}$ defines the indices of the interpolation points. We apply the transformation $T_1(u_m, v_m) = (u'_m, v'_m)$, $m = 0, 1, \ldots, M$, where

$$u'_m = u_0 + \sum_{j=1}^{m}(|u_j - u_{j-1}| + \varepsilon) = u'_{m-1} + (|u_m - u_{m-1}| + \varepsilon)$$

$$v'_m = v_m,$$

and $\varepsilon > 0$ is an arbitrary constant necessary when all points in an interpolation interval have equal u-coordinates, i.e. $u_m = u_{m-1}$ for every $m = \mathbb{J}(i)+1, \ldots, \mathbb{J}(i+1)$ and some $i \in \{0, 1, \ldots, N\}$. Otherwise, we set $\varepsilon = 0$. The resulting points (u'_m, v'_m), $i = 0, 1, \ldots, M$ are linearly ordered with respect to their abscissa, i.e. $u'_m < u'_n$ for every $m < n$. This transformation is essentially arraying the data points so as to preserve their horizontal distances. This is shown in the example depicted in Fig. 5, where the same interpolation points as in Fig. 4 have been used. Note that this transformation preserves the distances between consecutive points, i.e. $d((u_m, v_m), (u_{m-1}, v_{m-1})) = d((u'_m, v'_m), (u'_{m-1}, v'_{m-1}))$ for all $m = 1, \ldots, M$.

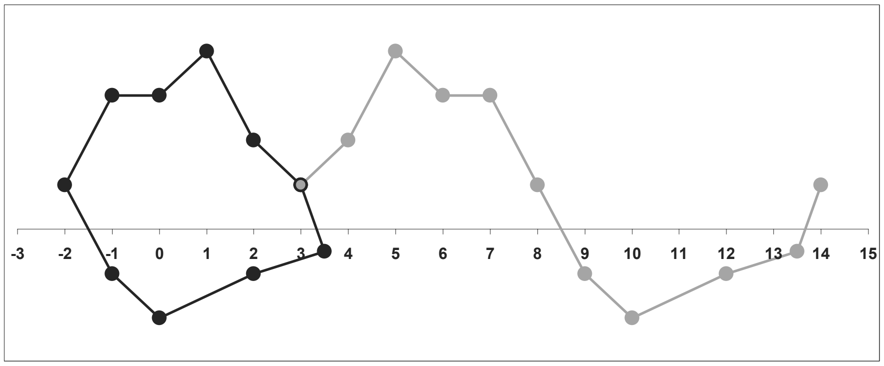

Fig. 5. The interpolation points (black) and their transformation (grey) for the FIC construction.

The next step is to create an IFS whose attractor is the graph of a function that interpolates the points $(u'_{\mathbb{J}(i)}, v'_{\mathbb{J}(i)})$, $i = 0, 1, \ldots, N$. This is achieved by using a $2D$ affine IFS (Sect. 2.1) and the result is its attractor G'.

The final step is to apply a second transformation to G' in order to obtain the graph G of a curve that interpolates the initial points $\{(u_m, v_m)\colon m = 0, 1, \ldots, M\}$. Let $(u', v') \in G'$ be a point of the attractor. We apply the transformation $T_2\colon G' \to G$ with $(u', v') \mapsto (u, v)$, where

$$\begin{aligned} u &= u_{m-1} + (u_m - u_{m-1}) \left(\frac{u' - u'_{m-1}}{u'_m - u'_{m-1}} \right), \quad u' \in [u'_{m-1}, u'_m] \\ v &= v'. \end{aligned}$$

Note that the overlapping at the endpoints of successive intervals $[u'_{i-1}, u'_i]$ in the above formula is not ambiguous, since the resulting u is the same in both cases. The transformation T_2 can be efficiently computed, if the points of the attractor are first sorted by u' and then the attractor and transformed data points are sweeped in parallel in order to calculate the appropriate (u, v).

The FIC defined above is open, assuming that the first and last points are different. To construct a closed FIC, we append an additional interpolation point that is the same as the first, i.e. we add $(u_{\mathbb{J}(N+1)}, v_{\mathbb{J}(N+1)}) = (u_{\mathbb{J}(0)}, v_{\mathbb{J}(0)})$. The curve is afterwards constructed in the same way. A fractal interpolation curve constructed by this method is depicted in Fig. 6, where the same interpolation points and contractivity factors as in Fig. 4(a) have been used. We note that the resulting curve is more similar to the one generated by the projection of Generalized FIF methods.

We call the proposed method *fractal curve fitting (FCF).* The advantage of the FCF method is that it offers a more compact representation using fewer parameters. As will be explained in the next section, for each interpolation interval it requires five parameters while the methods of Sect. 3.1 and 3.2 require

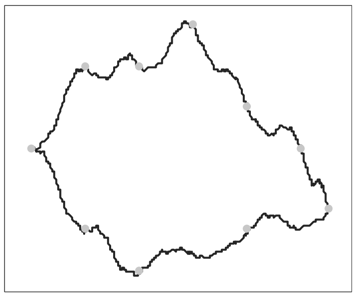

Fig. 6. A fractal interpolation curve constructed by the proposed FCF method.

ten. Moreover, it can be readily extended to represent curves in $\mathbb{R}^3$. The third coordinate has the same treatment as v, i.e. it remains unchanged by the transformations.

5 Results

In Fig. 7–9 the coastlines of the Greek islands Kimolos (A), Skyros (B) and Lemnos (C) consisting of 3897, 4663 and 7185 points, respectively, are presented. The coastlines have been extracted from digital aerial photographs of the islands using typical edge detection techniques. These data sets define closed curves and are suitable for fractal interpolation. A traditional interpolation method, using e.g. polynomials, would require very dense interpolation points in order to capture all the fine details of the coastlines.

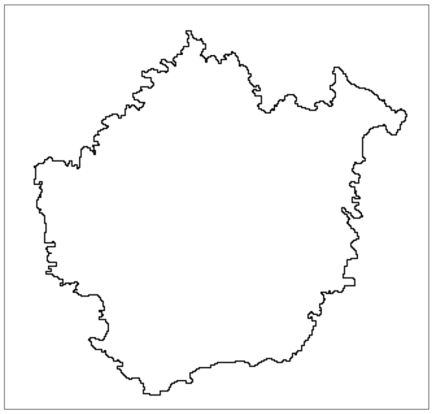

Fig. 7. A data set consisting of 3897 points and representing coastline A (Kimolos).

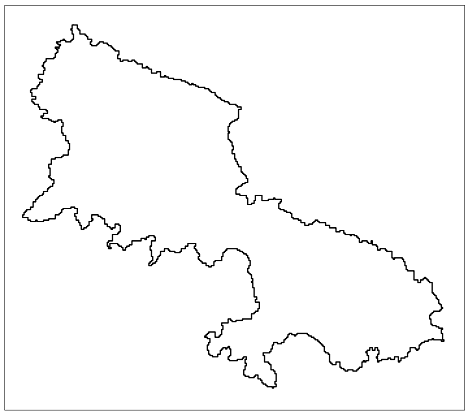

Fig. 8. A data set consisting of 4663 points and representing coastline B (Skyros).

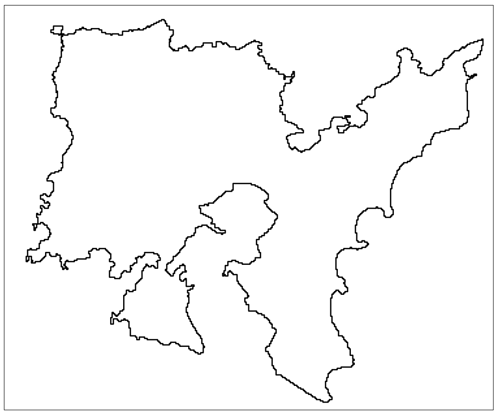

Fig. 9. A data set consisting of 7185 points and representing coastline C (Lemnos).

Tables 1–3 show the Hausdorff distance h and the Modified Hausdorff distance h_{MHD} between the original and reconstructed data for the three aforementioned coastlines, along with the total number of required transformation parameters p. We compare the projection of Generalized FIF method (Sec. 3.1), the coordinate separation method (Sec. 3.2) and the proposed FCF method (Sec. 4)[1]. The contractivity matrix for the projection of generalized FIF method is calculated with the algorithm of [3], while the vertical scaling factors for the other two methods are calculated with the analytic algorithm of [15]. The two algorithms are similar[2], both minimizing the sum of squared distances between original and reconstructed point coordinates using derivatives. Thus the results reflect the differences between the curve construction methods and not between the algorithms for calculating the scaling factors. The interpolation intervals have been chosen with a fixed increment L of 10 to 100, i.e. by taking every 10th to 100th point as interpolation point. As expected for all methods, the smaller the interpolation intervals/compression ratio, the smaller the distance between the original and reconstructed data. We also notice that in a few cases the increase in the length of the interpolation interval decreases the Hausdorff distance. This is rational, since the Hausdorff distance is sensitive to isolated, poorly approximated points. In these cases, the Modified Hausdorff distance provides a better overall comparison.

Table 1. The Hausdorff and Modified Hausdorff distance between original/reconstructed data and the number of required parameters for various interpolation interval lengths (coastline A).

L	**Method**								
	Proj. of Gen. FIF			**Coord. separation**			**Proposed FCF**		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	1.559	0.365	3890	2.974	0.519	3890	1.731	0.347	1945
20	2.651	0.486	1940	5.220	1.000	1940	3.157	0.506	970
30	2.976	0.683	1290	8.517	1.552	1290	5.047	0.704	645
40	4.503	0.874	970	11.493	1.939	970	5.545	0.855	485
50	4.987	1.107	770	13.865	2.449	770	6.262	1.113	385
60	10.161	1.330	640	13.930	2.982	640	9.234	1.278	320
70	10.430	1.592	550	16.396	3.378	550	9.225	1.524	275
80	9.967	1.829	480	19.057	3.644	480	14.469	1.766	240
90	10.162	2.085	430	17.758	3.828	430	11.369	1.811	215
100	14.481	2.238	380	18.363	3.928	380	15.559	2.135	190

In terms of Hausdorff and Modified Hausdorff distances, the proposed FCF method significantly outperforms the coordinate separation method in almost all cases and performs equally well to the projection of generalized FIF method.

[1] We have not compared the polar FIFs since the data are not ordered by angle in their polar form.

[2] The first is essentially an extention of the second.

Table 2. The Hausdorff and Modified Hausdorff distance between original/reconstructed data and the number of required parameters for various interpolation interval lengths (coastline B).

L	**Method**								
	Proj. of Gen. FIF			**Coord. separation**			**Proposed FCF**		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	1.712	0.370	4660	2.105	0.443	4660	1.861	0.339	2330
20	2.202	0.498	2330	4.017	0.755	2330	3.374	0.484	1165
30	3.017	0.664	1550	5.759	1.105	1550	4.364	0.680	775
40	3.788	0.858	1160	6.433	1.362	1160	5.274	0.849	580
50	4.926	1.033	930	9.327	1.701	930	6.211	1.049	465
60	7.596	1.263	770	10.846	1.939	770	7.885	1.278	385
70	7.946	1.460	660	11.908	2.404	660	8.650	1.501	330
80	7.721	1.624	580	11.411	2.356	580	10.538	1.520	290
90	11.632	1.875	510	15.132	3.099	510	12.063	1.907	255
100	11.571	2.007	460	13.173	3.245	460	13.874	2.126	230

Table 3. The Hausdorff and Modified Hausdorff distance between original/reconstructed data and the number of required parameters for various interpolation interval lengths (coastline C).

L	**Method**								
	Proj. of Gen. FIF			**Coord. separation**			**Proposed FCF**		
	h	h_{MHD}	p	h	h_{MHD}	p	h	h_{MHD}	p
10	2.403	0.372	7180	2.836	0.476	7180	2.397	0.353	3590
20	2.666	0.481	3590	5.375	0.742	3590	2.859	0.482	1795
30	3.433	0.627	2390	7.182	1.096	2390	4.564	0.645	1195
40	4.187	0.776	1790	10.063	1.451	1790	6.246	0.844	895
50	5.300	0.966	1430	11.532	1.812	1430	6.648	1.012	715
60	6.342	1.132	1190	13.486	2.326	1190	7.695	1.163	595
70	7.588	1.285	1020	16.582	2.640	1020	9.971	1.451	510
80	8.608	1.482	890	20.037	3.006	890	10.110	1.544	445
90	9.560	1.584	790	21.066	3.462	790	10.813	1.617	395
100	9.845	1.830	710	24.233	3.973	710	12.081	1.865	355

Moreover, the proposed method uses five parameters for each pair of consecutive interpolation points (one affine transformation of five parameters (Eq. 2.1)), while the other two methods require ten parameters. Specifically, the projection of generalized FIF method uses one affine transformation of ten parameters (Eq. 2.2), while the coordinate separation method uses two affine transformations of five parameters each (Eq. 2.1). This implies that for a specific compression ratio we can use the proposed FCF method with twice the number of interpolation points than the other two and thus obtain better results. Comparing the previous results from this point of view it is evident that, for a specific compression ratio, the proposed method clearly outperforms both others achieving smaller error or, conversely, for a specific error level it achieves better compression ratio. Moreover, it has the advantage that for each pair of consecutive interpolation points only one free parameter (vertical scaling factor) has to be determined, while two and four such parameters are required for the coordinate separation and projection of generalized FIF methods respectively. This is useful, for example, when using these free parameters to describe a family of curves, thus offering a more compact representation.

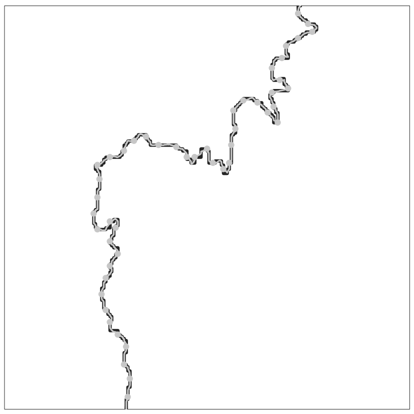

Fig. 10. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 10 ($r = 1\colon 4.01$).

In Fig. 10–12 parts of the reconstructed FICs for coastline A using the proposed FCF method with interpolation intervals of length 10, 30 and 50 respectively are presented. As shown in the figures, the reconstructed FICs provide an accurate representation of the coastline even with sparse interpolation points,

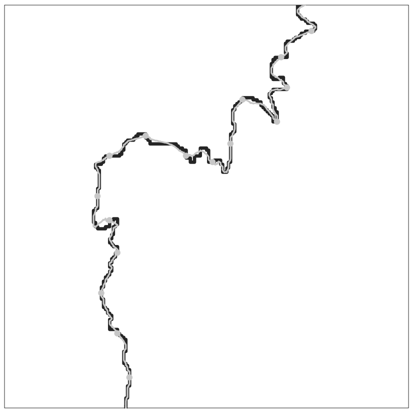

Fig. 11. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 30 ($r = 1{:}\,12.08$).

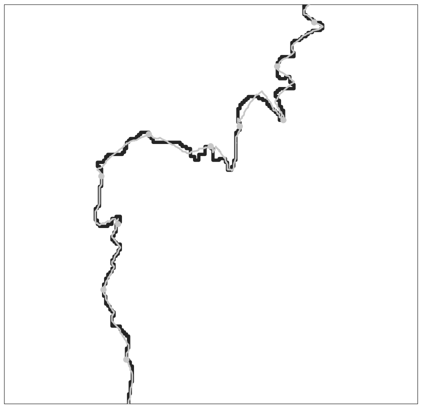

Fig. 12. The data points (black), interpolation points (grey circles) and FIC (grey) for coastline A using the proposed FCF method with interpolation intervals of length 50 ($r = 1{:}\,20.24$).

and therefore high compression ratios are achieved. Specifically, the compression ratios for these examples are 1: 4.01, 1: 12.08 and 1: 20.24 respectively[3].

In the left part of Fig. 13–15 three more coastlines are presented (Amorgos (D), Astypalaia (E), Tilos (F)), consisting of 4410, 7510 and 6172 points, respectively. In the right part of the figures, the reconstructed curves using FCF method are presented, achieving compression ratios of 1: 5.01, 1: 6.68 and 1: 8.02, respectively. As shown in the figures, the reconstructed FICs accurately represent the coastlines, while requiring considerably less data.

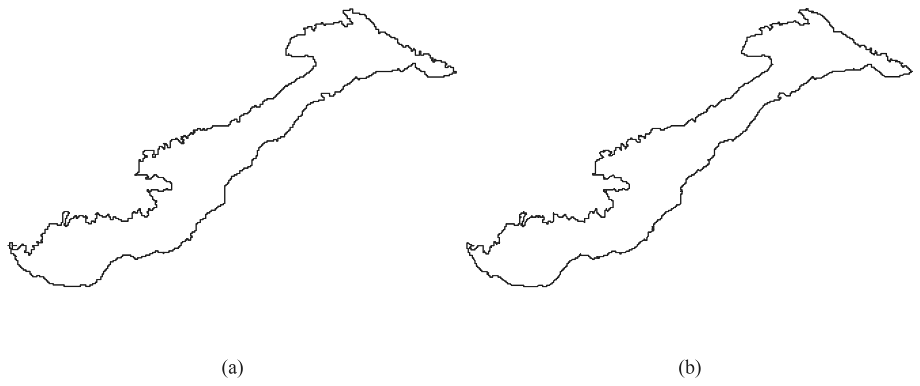

Fig. 13. (a) Coastline D (Amorgos) consisting of 4410 points. (b) The reconstructed curve using FCF method with $r = 1{:}\,5.01$.

In the previous examples we have used interpolation intervals of fixed length. It is possible to define intervals of variable length, e.g. using the iterative algorithm of [15]. In this case, we could achieve even better results by exploiting more efficiently the possible self-affinity of the data.

6 Conclusions

An accurate and economical new method for curve fitting using fractal interpolation has been introduced. Results show that, for a specific compression ratio, the proposed method clearly outperforms existing ones. Moreover, it has the advantage of offering a more economical representation using fewer bound and free parameters. Further work will focus on using piecewise self-affine FIFs ([15]) for interpolating the transformed data of the proposed method. This approach

[3] The compression ratio is calculated as $r = 5N/2M$, where N is the number of affine transformations and M is the number of data points. Note that the denominator is multiplied by 2 since the data points have two coordinates.

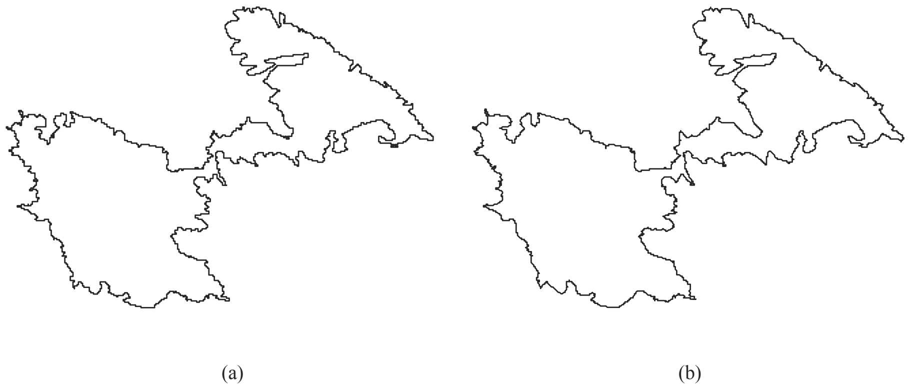
Fig. 14. (a) Coastline E (Astypalaia) consisting of 7510 points. (b) The reconstructed curve using FCF method with $r = 1\colon 6.68$.

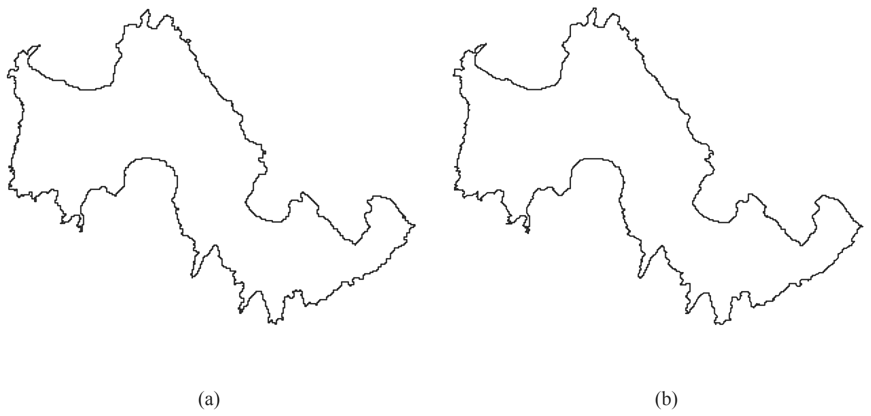
Fig. 15. (a) Coastline F (Tilos) consisting of 6172 points. (b) The reconstructed curve using FCF method with $r = 1\colon 8.02$.

is expected to be better for curves that present self-affinity in subintervals and not at their whole length. Moreover, it will be useful to define bounds for the contractivity factors such that the resulting curve is not self-intersecting.

Acknowledgements The authors would like to acknowledge financial support from the 03ED375 research project, implemented within the framework of the Reinforcement Programme of Human Research Manpower (PENED) and co-financed by National and Community Funds (25% from the Greek Ministry of Development-General Secretariat of Research and Technology and 75% from E.U.-European Social Fund).

References

1. Cochran, W.O., Hart, J.C., Flynn, P.J.: On approximating rough curves with fractal functions. In: Proc. Graphics Interface. Volume 1. (1998) 65–72
2. Mazel, D.S.: Representation of discrete sequences with three-dimensional iterated function systems. IEEE Trans. Signal Processing **42** (1994) 3269–3271
3. Mazel, D.S., Hayes, M.H.: Hidden-variable fractal interpolation of discrete sequences. In: Proc. Int. Conf. ASSP. Volume 1. (1991) 3393–3396
4. Uemura, S., Haseyama, M., Kitajima, H.: Efficient contour shape description by using fractal interpolation functions. In: IEEE Proc. ICIP. Volume 1. (2002) 485–488
5. Dalla, L., Drakopoulos, V.: On the parameter identification problem in the plane and the polar fractal interpolation functions. J. Approx. Theory. **101** (1999) 290–303
6. Guérin, E., Tosan, E., Baskurt, A.: Fractal coding of shapes based on a projected IFS model. In: ICIP. Volume 2., IEEE Computer Society (2000) 203–206
7. Guérin, E., Tosan, E., Baskurt, A.: A fractal approximation of curves. Fractals **9**(1) (2001) 95–103
8. Navascués, M.A., Sebastián, M.V.: Fitting curves by fractal interpolation: An application to the quantification of cognitive brain processes. In Novak, M.M., ed.: Thinking in patterns: Fractals and related phenomena in nature, Singapore, World Scientific (2004) 143–154
9. Cader, A., Krupski, M.: New interpolation method with fractal curves. In Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J., eds.: ICAISC. Volume 4029 of Lecture Notes in Computer Science., Berlin and Heidelberg, Springer-Verlag (2006) 1071–1081
10. Barnsley, M.F.: Fractal functions and interpolation. Constr. Approx. **2** (1986) 303–329
11. Barnsley, M.F.: Fractals everywhere. 2nd edn. Academic Press Professional, San Diego (1993)
12. Zhao, C., Shi, W., Y., D.: A new Hausdorff distance for image matching. Pattern Recognition Lett. **26** (2005) 581–586
13. Ruan, H.J., Sha, Z., Su, W.Y.: Counterexamples in parameter identification problem of the fractal interpolation functions. J. Approx. Theory. **122** (2003) 121–128
14. Marvasti, M., Strahle, W.: Fractal geometry analysis of turbulent data. Signal Processing **41** (1995) 191–201
15. Mazel, D.S., Hayes, M.H.: Using iterated function systems to model discrete sequences. IEEE Trans. Signal Processing **40** (1992) 1724–1734