Int. J. Appl. Math. Comput. Sci., 2011, Vol. 21, No. 4, 757–767
DOI: 10.2478/v10006-011-0060-8

PARTITIONED ITERATED FUNCTION SYSTEMS WITH DIVISION AND A FRACTAL DEPENDENCE GRAPH IN RECOGNITION OF 2D SHAPES

KRZYSZTOF GDAWIEC, DIANA DOMAŃSKA

Institute of Computer Science
University of Silesia, Będzińska 39, 41–200 Sosnowiec, Poland
email: kgdawiec@ux2.math.us.edu.pl, ddomanska@poczta.onet.pl

One of the approaches in pattern recognition is the use of fractal geometry. The property of self-similarity of fractals has been used as a feature in several pattern recognition methods. All fractal recognition methods use global analysis of the shape. In this paper we present some drawbacks of these methods and propose fractal local analysis using partitioned iterated function systems with division. Moreover, we introduce a new fractal recognition method based on a dependence graph obtained from the partitioned iterated function system. The proposed method uses local analysis of the shape, which improves the recognition rate. The effectiveness of our method is shown on two test databases. The first one was created by the authors and the second one is the MPEG7 CE-Shape-1PartB database. The obtained results show that the proposed methodology has led to a significant improvement in the recognition rate.

Keywords: fractal, partitioned iterated function system, shape recognition, dependence graph.

1. Introduction

The shape of an object (shape is a group of touching foreground pixels, i.e., a connected binary region (Burger and Burge, 2008)) is very important in object recognition. Using the shape of an object for object recognition is a growing trend in computer vision. Good shape descriptors and matching measures are a central issue in these applications. Based on the silhouette of objects, a variety of shape descriptors and matching methods have been proposed in the literature, e.g., shape contexts (Belongie *et al.*, 2002), the generative model (Tu and Yuille, 2004), the curvature scale space (Mokhtarian and Bober, 2003), polygonal multiresolution (Attalla and Siy, 2005), the inner distance (Ling and Jacobs, 2005), the shape-tree (Felzenszwalb and Schwartz, 2007) as well as the X-graph and Y-graph (Chang *et al.*, 2010).

The idea of fractals was first presented by Mandelbrot in the 1970s (Mandelbrot, 1983). Barnsley presented a couple of revolutionary ideas based on the hypothesis presented by Mandelbrot, emphasizing the practical aspect of fractals. He provided methods to model natural fractals and the idea of the Iterated Function System (IFS) as a tool to generate fractals (Barnsley, 1988). Since then fractals have found many applications, e.g., in image compression (Fisher, 1995; Nikiel, 2007), generating terrains (Meng *et al.*, 2009), generating plants (Prusinkiewicz and Lindenmayer, 1996), image processing (Ghazel *et al.*, 2003), or medicine (Dey, 2005) and economics (Peters, 1994). One such application is the use of fractals in pattern recognition. The motivation to use fractals in pattern recognition was the fact that with the help of fractals we are able to represent the shape much better than with the help of classical Euclidean geometry. Fractal recognition methods have found applications in face recognition (Chandran and Kar, 2002; Kouzani, 2008; Skarbek and Ignasiak, 1996), signature verification (Huang and Yan, 2000), character recognition (Linnell and Deravi, 2004; Mozaffari *et al.*, 2006), gait recognition (Zhao *et al.*, 2007), plant identification (Bruno *et al.*, 2008; Plotze *et al.*, 2005), or as a general recognition method (Neil and Curtis, 1997; Yokoyama *et al.*, 2004).

In this paper we present some weaknesses of fractal recognition methods and how to improve them. Moreover, we introduce a new fractal recognition method which will be used in the recognition of 2D shapes. As fractal features we used a dependence graph (Domaszewicz and Vaishampayan, 1995) achieved from a Partitioned Iterated Function System (PIFS) (Fisher, 1995).

In Section 2 we introduce the notion of a fractal (fractal as an attractor (Barnsley, 1988)) which is used in this paper and some basic information about fractals. In Section 3, we briefly present fractal image compression (Fisher, 1995), which gives us the PIFS code. Then, in Section 4 we present fractal pattern recognition with division. We show an observation presenting a weakness of the fractal recognition methods and introduce the notion of the PIFS with division and other related notions. Then, in Section 5 we present how to modify the existing fractal recognition methods using the PIFS with division. In Section 6 we present the definition of the dependence graph (Domaszewicz and Vaishampayan, 1995) and our fractal recognition method of 2D shapes which uses the dependence graph for recognition. The effectiveness of our method is shown on two test databases (Section 7). The first one is database created by the authors and the second one is the MPEG7 CE-Shape-1 Part B database. Finally, in Section 8 we present conclusions and plans for the future.

2. Fractal as an attractor

We can find many nonequivalent definitions of a fractal in the literature. Starting from an invariant measure (Kolumbán *et al*., 2003), through the definition of a fractal as an attractor (Barnsley, 1988) or as a set for which the Hausdorff dimension is greater than the topological dimension (Mandelbrot, 1983). In this section we will introduce the definition which we use in this paper. First we must introduce some notions.

Definition 1. A *metric space* is a pair (X, ρ), where X is a nonempty set and $\rho : X \times X \to [0, +\infty)$ satisfies the following conditions:

1. $\forall_{x,y\in X} \quad \rho(x, y) = 0 \iff x = y.$
2. $\forall_{x,y\in X} \quad \rho(x, y) = \rho(y, x).$
3. $\forall_{x,y,z\in X} \quad \rho(x, z) \leq \rho(x, y) + \rho(y, z).$

Definition 2. A metric space (X, ρ) is called a *complete* metric space if every sequence $(x_n)_{n\in\mathbb{N}}$ of elements of X satisfying the condition

$$\forall_{\varepsilon>0}\exists_{N\in\mathbb{N}}\forall_{n,m>N} \quad \rho(x_n, x_m) < \varepsilon \tag{1}$$

is convergent.

Let us take any complete metric space (X, ρ) and denote by $\mathcal{H}(X)$ the space of nonempty, compact subsets of X. In this space we introduce a metric $h : \mathcal{H}(X) \times \mathcal{H}(X) \to [0, +\infty)$ which is defined as follows:

$$h(R, S) = \max\{\max_{x\in R}\min_{y\in S}\rho(x, y), \max_{y\in S}\min_{x\in R}\rho(y, x)\}, \tag{2}$$

where $R, S \in \mathcal{H}(X)$.

The space $\mathcal{H}(X)$ with the metric h is a complete metric space (Barnsley, 1988). Another necessary notion is a contraction mapping.

Definition 3. A transformation $w : X \to X$ on a metric space (X, d) is called a *contraction mapping* if there exists a constant $0 \leq s < 1$ such that

$$\forall_{x,y\in X} \quad d(w(x), w(y)) \leq s \cdot d(x, y). \tag{3}$$

Any such number s is called a *contractivity factor* for w.

Definition 4. We say that a set $W = \{w_1, \ldots, w_N\}$, where w_i is a contraction mapping with contractivity factor s_i for $i = 1, \ldots, N$, is an *iterated function system*.

An IFS so defined determines the so-called Hutchinson operator (Barnsley, 1988), which is defined as follows:

$$\forall_{A\in\mathcal{H}(X)} \quad W(A) = \bigcup_{i=1}^{N} w_i(A) = \bigcup_{i=1}^{N}\{w_i(a) : a \in A\}. \tag{4}$$

The Hutchinson operator is a contraction mapping with a contractivity factor $s = \max\{s_1, \ldots, s_N\}$ (Barnsley, 1988). Let us consider the following recurrent sequence:

$$\begin{cases} W^0(A) = A, \\ W^k(A) = W(W^{k-1}(A)) \quad \text{if } k \geq 1, \end{cases} \tag{5}$$

where $A \in \mathcal{H}(X)$.

The next theorem is a consequence of the Banach fixed point theorem (Barnsley, 1988).

Theorem 1. *Let (X, ρ) be a complete metric space and $W = \{w_1, \ldots, w_N\}$ be an IFS. Then only one set $B \in \mathcal{H}(X)$ exist such that $W(B) = B$. Furthermore, the sequence defined by Eqn.* (5) *is convergent and*

$$\forall_{A\in\mathcal{H}(X)} \quad \lim_{k\to\infty} W^k(A) = B. \tag{6}$$

Definition 5. The limit (6) in Theorem 1 is called an *attractor* of the IFS or a fractal.

3. Fractal image compression

Fractal image compression can be described in many different ways (Barni, 2006; Fisher, 1995). In every method the fact that every image has partial self-similarity is used. Below we introduce a basic algorithm of fractal compression which is necessary in the proposed recognition method and many other fractal recognition techniques. First we describe the notion of a partitioned iterated function system (Fisher, 1995).

Definition 6. We call a set

$$P = \{(T_1, A_1), \ldots, (T_N, A_N)\}$$

a *partitioned iterated function system*, where T_i is a contraction mapping, A_i is the area of the image which we transform with the help of T_i for $i = 1, \ldots, N$.

In much the same way as in the case of the IFS, the PIFS has a unique fixed point, the image that it encodes.

In the image space we use affine transformations $T : \mathbb{R}^3 \to \mathbb{R}^3$ as the contraction mappings of the following form:

$$T\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} a_1 & a_2 & 0 \\ a_3 & a_4 & 0 \\ 0 & 0 & a_7 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} a_5 \\ a_6 \\ a_8 \end{bmatrix}, \quad (7)$$

where $x, y \in \mathbb{R}$ are the co-ordinates of the pixel that we transform, $z \in \mathbb{R}$ is the pixel intensity, the coefficients $a_1, a_2, a_3, a_4, a_5, a_6 \in \mathbb{R}$ describe a geometric transformation (translation, change of scale, rotation, shearing), and coefficients $a_7, a_8 \in \mathbb{R}$ are responsible for contrast and brightness.

The compression algorithm, whose idea is presented in Fig. 1, can be described as follows. We divide an image into a fixed number of nonoverlapping areas of the image called range blocks. We create a list of domain blocks. The list consists of overlapping areas of the image, larger than the range blocks (usually two times larger) and transformed using the following mappings:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \quad (8)$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \quad (9)$$

These four mappings are transformations of the rectangle (identity, 180° rotation and two symmetries of the rectangle). Next, for every range block R we look for the domain block D so that the value $\rho(R, T(D))$ is the smallest, where ρ is a metric (usually Euclidean) and blocks $R, T(D)$ are treated as vectors, T is a transformation determined by the position of R and D, the size of these in relation to itself and one of the four mappings defined by Eqns. (8) and (9), the coefficients a_7, a_8 are calculated with the help of Eqns. (10) and (11). This is the most time-consuming step of the algorithm:

$$a_7 = \frac{k \sum_{i=1}^{k} g_i h_i - \sum_{i=1}^{k} g_i \sum_{i=1}^{k} h_i}{k \sum_{i=1}^{k} g_i^2 - \left(\sum_{i=1}^{k} g_i \right)^2}, \quad (10)$$

$$a_8 = \frac{1}{k} \left[\sum_{i=1}^{k} h_i - a_7 \sum_{i=1}^{k} g_i \right], \quad (11)$$

where k is the number of pixels in the range block, $g_1, \ldots, g_k$ are the pixel intensities of the transformed and resized domain block, $h_1, \ldots, h_k$ are the pixel intensities of the range block. If

$$k \sum_{i=1}^{k} g_i^2 - \left(\sum_{i=1}^{k} g_i \right)^2 = 0,$$

then we fix (Fisher, 1995)

$$a_7 = 0, \quad a_8 = \frac{1}{k} \sum_{i=1}^{k} h_i.$$

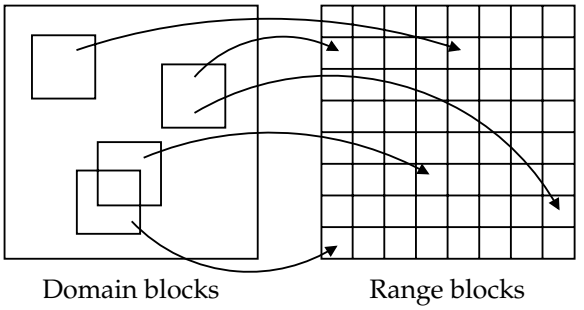


Fig. 1. Fractal image compression.

This algorithm is very simple and therefore used only in fractal image recognition. Moreover, in recognition of two-dimensional shapes in binary images, the coefficients a_7 and a_8 are omitted. In practice, when we compress an image, we use adaptive methods of partitioning such as the quad-tree, HV partition and others (Fisher, 1995).

4. Fractal pattern recognition with division

All fractal recognition methods of objects in which we use the PIFS obtained from fractal image compression are based on a global analysis of the shape. In practice we get better results using a local analysis. This is due to the fact that, when we compare two objects, the difference between them occurs only in regions where the object was changed, and the parts of the object which were not changed are identical. In this section we will present some observation which will help us to bring local analysis into fractal recognition methods. Moreover, we will present mathematical formalism of partitioned iterated function systems with division (Gdawiec, 2009b).

4.1. Fractal recognition methods and division into sub-images. Let us take a look at fractal compression (presented in Section 3) from the point of view of a domain block D. Further, assume that this block fits into several range blocks (Fig. 2(a)). Each of these fits corresponds to one mapping in the PIFS. Now let us suppose that block D was changed into block D' (e.g., the shape was cut or deformed). This situation is shown in Fig. 2(b). In this case, domain D' can not only fit into the same range blocks

as D (to all or only some), but it can also fit into some other range blocks. This change of fitting causes a change of the mapping in the PIFS. In the worst case all mappings can be changed.

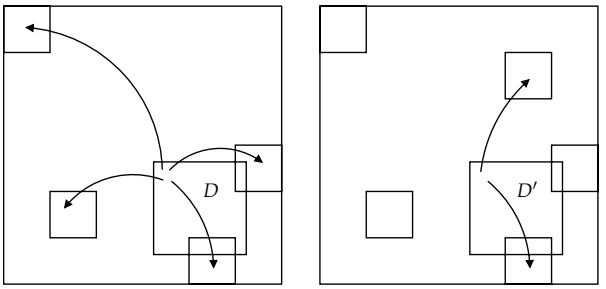


Fig. 2. Fractal image compression: starting situation (a), situation after the change of D into D' (b).

Now we divide the image into several non-overlapping sub-images, e.g., into four sub-images (Fig. 3(a)), and compress each of them independently (the coordinates of the domain and range blocks are determined in relation to the original image, not the sub-image). Again let us consider the same domain block D and the same range blocks. This time, block D fits only into the range blocks from the same sub-image, the other range blocks from different sub-images fit into other domain blocks D_1, D_2 (Fig. 3(a)). Now suppose that block D was changed in the same way as earlier (Fig. 3(b)). The change of the block influences only the sub-image in which block D' is placed. The fitting in other sub-images does not change. This time in the worst case only mappings corresponding to this sub-image change, and all the other mappings remain the same. So, a local change in the shape has only local influence on the transformations of the PIFS and not global one as in the previous case.

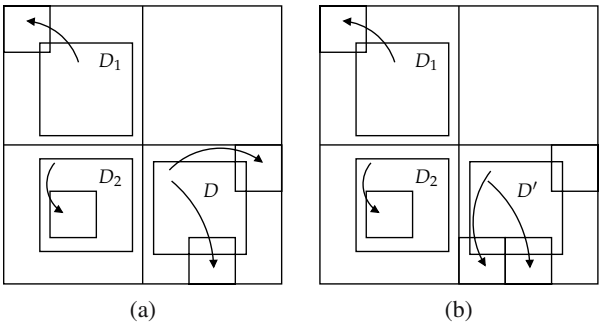


Fig. 3. Fractal image compression with division: starting situation (a), situation after the change of D into D' (b).

4.2. Partitioned iterated function system with division. Let us take a space of images $F = \{f : I^2 \to I\}$ with the supremum metric, where $I = [0, 1]$.

Definition 7. A *division* of the unit square I^2 is every family of sets $\Pi = \{P_1, \ldots, P_N\}$ such that

1. $\bigcup_{n=1}^{N} P_n = I^2$,
2. $\forall_{i \neq j, i,j \in \{1,\ldots,N\}} \quad P_i \cap P_j = \emptyset$.

Definition 8. Let $f \in F$. A *division into sub-images* of the image f is a family of functions $\Gamma = \{f_1, \ldots, f_N\}$ such that $f_i = f|_{P_i}$ for $i = 1, \ldots, N$, where $\Pi = \{P_1, \ldots, P_N\}$ is a division of the unit square I^2.

Let us notice that straight from Definition 8 we have that, if $\Gamma = \{f_1, \ldots, f_N\}$ is a division into sub-images of the image f, then

$$\bigcup_{i=1}^{N} f_i = f, \tag{12}$$

where the notation $\bigcup_{i=1}^{N} f_i$ is understood as a composition of functions graphs.

Let us fix a division of the unit square $\Pi = \{P_1, \ldots, P_N\}$, and let us take any $f \in F$. The division Π determines a division into sub-images of this image $\Gamma = \{f_1, \ldots, f_N\}$. Every sub-image $f_i \in \Gamma$ is compressed with the help of the fractal algorithm obtaining a PIFS W_i for $i = 1, \ldots, N$.

We define an operator $W : F \to F$ in the following way:

$$\forall_{g \in F} \quad W(g) = \bigcup_{i=1}^{N} W_i(g|_{P_i}), \tag{13}$$

and the n-th iteration of this operator as

$$\forall_{g \in F} \quad W^n(g) = \bigcup_{i=1}^{N} W_i^n(g|_{P_i}). \tag{14}$$

Then, for any $g \in F$ we have

$$\begin{aligned} \lim_{n\to\infty} W^n(g) &= \lim_{n\to\infty} \bigcup_{i=1}^{N} W_i^n(g|_{P_i}) \\ &= \bigcup_{i=1}^{N} \lim_{n\to\infty} W_i^n(g|_{P_i}). \end{aligned} \tag{15}$$

Because for every $i = 1, \ldots, N$ operator W_i is a PIFS, $\lim_{n\to\infty} W_i^n(g|_{P_i}) = f_i$. Therefore,

$$\lim_{n\to\infty} W^n(g) = \bigcup_{i=1}^{N} f_i = f. \tag{16}$$

From this we see that, if we introduce the division of image into sub-images and then compress each of this sub-image with the fractal image compression algorithm, then when we iterate the operator determined by the PIFSs of the sub-images we obtain the starting image. Let

us notice that the "usual" fractal image compression algorithm is a special case of compression with division into sub-images. Indeed, if we take a division of the image into one sub-image, i.e., the image is a division of itself, then the iterations of the operator W are simply next iterations of the PIFS of that image.

Definition 9. Let $\Pi = \{P_1, \ldots, P_N\}$ be a division of the unit square I^2 and $f \in F$. A set $W = \{W_1, \ldots, W_N\}$, where W_i is a PIFS of the sub-image f_i for $i = 1, \ldots, N$, is called a *partitioned iterated function system with division* (PIFS with division).

Let us fix a division of the unit square $\Pi = \{P_1, \ldots, P_N\}$. Mark with $\mathcal{P}_\Pi$ a set of all PIFSs with division Π. Consider a function $d_\rho : \mathcal{P}_\Pi \times \mathcal{P}_\Pi \to [0, +\infty)$ given by the following formula:

$$\forall_{W,V \in \mathcal{P}_\Pi} \quad d_\rho(W, V) = \sum_{i=1}^{N} \rho(W_i, V_i), \tag{17}$$

where ρ is an arbitrary metric for PIFSs.

Theorem 2. *Let $\Pi = \{P_1, \ldots, P_N\}$ be a fixed division of the unit square I^2. Then $(\mathcal{P}_\Pi, d_\rho)$ is a metric space.*

Proof. This follows from the fact that ρ is a metric. ■

Theorem 3. *Let $\Pi = \{P_1, \ldots, P_N\}$ be a fixed division of the unit square I^2 and W be a PIFS with division for an image $f \in F$. Then f is a unique fixed point of W, i.e., $W(f) = f$.*

Proof. First we show that f is a fixed point of W. For every $i = 1, \ldots, N$ the operator W_i is a PIFS of the sub-image f_i, so f_i is a unique fixed point of W_i, i.e. $W_i(f_i) = f_i$. Then

$$W(f) = \bigcup_{i=1}^{N} W_i(f|_{P_i}) = \bigcup_{i=1}^{N} f|_{P_i} = f. \tag{18}$$

Now we show that f is a unique fixed point of W. Let us assume that W possess at least two fixed points. Let g and h be two fixed points of W, i.e., $W(g) = g$ and $W(h) = h$.

From the fact that $W(g) = g$, we obtain that $\bigcup_{i=1}^{N} W_i(g|_{P_i}) = \bigcup_{i=1}^{N} g|_{P_i}$. Because $W_i : P_i \times I \to P_i \times I$, we get $W_i(g|_{P_i}) = g|_{P_i}$. Hence $g|_{P_i}$ is a fixed point of W_i. In an analogical way we can show that $W_i(h|_{P_i}) = h|_{P_i}$. Thus $h|_{P_i}$ is a fixed point of W_i.

For all $i = 1, \ldots, N$ the mapping W_i is a PIFS, so it possesses only one fixed point. Therefore, $g|_{P_i} = h|_{P_i}$ for $i = 1, \ldots, N$. Then

$$g = \bigcup_{i=1}^{N} g|_{P_i} = \bigcup_{i=1}^{N} h|_{P_i} = h. \tag{19}$$

So we have shown that if W possesses at least two fixed points, then they are equal. Moreover, from the first part of the proof we know that f is a fixed point of W. Therefore, we obtain that W has a unique fixed point, which is f. ■

5. Existing fractal recognition methods with division

The known fractal recognition methods use two different approaches to recognition. One is the application of the fractal dimension (Bruno *et al*., 2008; Plotze *et al*., 2005), and the other is the employment of the PIFS. In the latter case, we use the fact that the attractor is a fixed point of the PIFS (Kouzani, 2008; Neil and Curtis, 1997; Skarbek and Ignasiak, 1996) or we extract some features from the coefficients of the PIFS, e.g., multiple mapping vector accumulator matrices (Mozaffari *et al*., 2006), multiple domain-range co-location matrix (Mozaffari *et al*., 2006), mapping vectors (Yokoyama *et al*., 2004), or even only coefficients (Chandran and Kar, 2002).

In the case of methods that use the PIFS, we can modify them employing the PIFS with division from Section 4.2. Independently of the approach (attractor as a fixed point or extracting the features), we replace the PIFS with the PIFS with a fixed division. In the case of methods that use the fixed point fact, this is all we do, but in the case of methods that extract the features from the PIFS, we must also modify the extracting process. Having a PIFS with division we extract the features from each of the PIFS separately using the methodology from the examined (original) method. In this way we obtain a new set of features. Next, we modify the similarity measure. For this purpose we use the original similarity measure S from the method considered and define a new one

$$d_S(Q^1, Q^2) = \sum_{i=1}^{N} S(Q_i^1, Q_i^2), \tag{20}$$

where $Q^1 = \{Q_1^1, \ldots, Q_N^1\}$, $Q^2 = \{Q_1^2, \ldots, Q_N^2\}$ are sets of features.

6. Fractal dependence graph method

Domaszewicz and Vaishampayan (1995) introduced the notion of a dependence graph. The graph reflects how domain blocks are assigned to range blocks. They used this graph for three different purposes. The first was an analysis of the convergence of fractal compression. The second was a reduction of the decoding time, and the last one was the improvement upon collage coding.

Before we give the definition of the dependence graph, we will introduce the definition of a directed graph and the adjacency matrix of a graph (Harris *et al*., 2008).

Definition 10. We say that a pair $G = (V, E)$ is a *directed graph* (digraph) if V is a finite, nonempty set and E is a

set of ordered pairs from V. The elements of V are called *vertices*, and the elements of E are called *edges*.

Definition 11. Let $G = (V, E)$ be a directed graph and $|V| = n$, where $|A|$ means the cardinality of a set A. The *adjacency matrix* of G is a matrix $\mathbf{M}$ of size $n \times n$ whose elements m_{ij} for $i, j \in \{1, \ldots, n\}$ are defined by

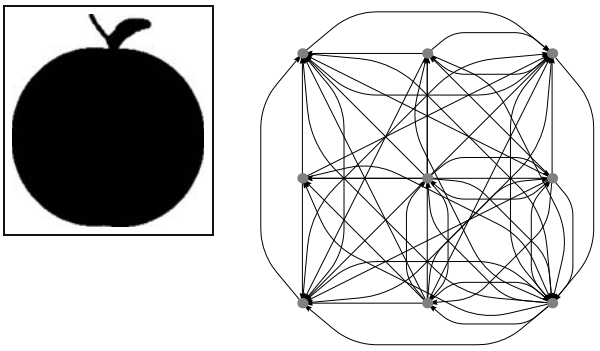
$$m_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E, \\ 0 & \text{if } (v_i, v_j) \notin E, \end{cases} \tag{21}$$

where $v_i, v_j \in V$.

The definition of a dependence graph is as follows.

Definition 12. Let W be the PIFS with a set of range blocks $\mathcal{R}$. The *dependence graph* of W is a directed graph $G = (V, E)$ where $V = \mathcal{R}$ and for all $R_i, R_j \in \mathcal{R}$ we have $(R_i, R_j) \in E$ if and only if the domain block D corresponding to R_j overlaps R_i, i.e., $D \cap R_i \neq \emptyset$.

Figure 4 presents an example of an image and a dependence graph corresponding to the PIFS which encodes the image. The fractal coding was done using partition into $3 \times 3 = 9$ range blocks.



(a) (b)

Fig. 4. Example of an image (a) and its dependence graph (b).

For the PIFS with division, the dependence graph is created by taking all the dependence graphs of the sub-image PIFSs.

Now we are ready to introduce the Fractal Dependence Graph (FDG) method. First, we fix the partition of the unit square $\Pi = \{P_1, \ldots, P_N\}$. Next, for each PIFS W_i encoding the sub-image f_i we fix the number of mappings of which the PIFS consists and denote it by n_i for $i = 1, \ldots, N$. Let

$$M = \sum_{i=1}^{N} n_i \tag{22}$$

be the number of all mappings of forming the PIFS with division.

The fractal dependence graph method is as follows:

1. Binarise the image and extract the object.
2. Find a set of correct orientations Θ.
3. Choose any correct orientation $\theta \in \Theta$ and rotate the object through θ.
4. Find a normalized PIFS with division W, i.e., for which the space is $[0, 1]^2$.
5. Determine the adjacency matrix $\mathbf{G}$ for the PIFS with division W.
6. In the base, find the adjacency matrix $\mathbf{H}$ which minimizes the Frobenius matrix norm (Golub and van Loan, 1996) of the matrix $\mathbf{G} - \mathbf{H}$, i.e.,

$$d_{\mathbf{H}} = \|\mathbf{G} - \mathbf{H}\| = \sqrt{\sum_{i=1}^{M} \sum_{j=1}^{M} |g_{ij} - h_{ij}|^2}. \tag{23}$$

7. Choose an image from the base which corresponds to $d_{\mathbf{H}}$.

The normalization of the PIFS is used to make the method robust to translations and change in scale of the object. The correct orientation is used to make the method robust to rotations and is defined as follows.

Definition 13. A *correct orientation* is an angle by which we need to rotate an object so that it fulfils the following conditions:

1. The area of the bounding box is the smallest.
2. The height of the bounding box is smaller than the width.
3. The left half of the object has at least as many pixels as the right.

Figure 5 presents examples of objects and their correct orientations. In the case of the triangle we see three different orientations. If we want to add such an object to the base for each of the correct orientations, we find the corresponding normalized PIFS with division, determine the dependence graph of this PIFS with division, and add the corresponding adjacency matrix to the base.

To find a normalized PIFS with division, firstly we find a PIFS with division P described in Section 4.2. Next, we make the normalization: For each transformation (T, A) belonging to P, we take A and divide the x-coordinate of the points defining the area by the width of the image, the y-coordinate of these points by the height of the image, and in similar way we normalize the translation vector $[a_5, a_6]^T$ of the transformation T.

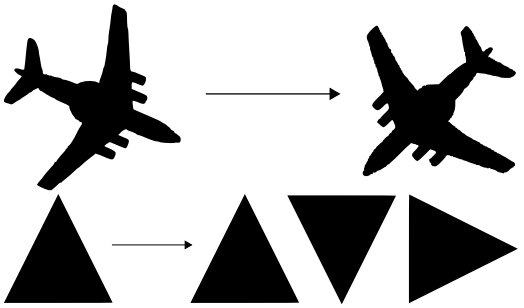


Fig. 5. Examples of objects and their correct orientations.

7. Experiments

Experiments were performed on two databases. The first was created by the authors, and the second was the MPEG7 CE-Shape-1 Part B database (Latecki *et al.*, 2000).

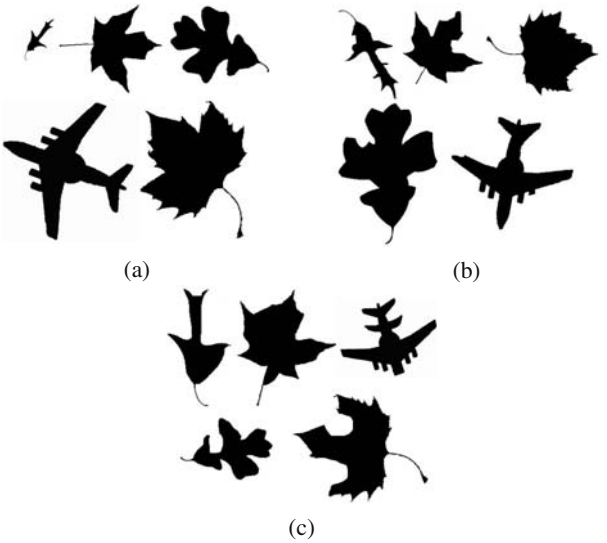
Our base consists of three datasets. In each we have 5 classes of objects, 20 images per class. Figure 6 presents the base images used to create the three datasets. In the first dataset we have base objects changed by elementary transformations, i.e., rotation, scaling, translation. Example images from this dataset are presented in Fig. 7(a). In the second dataset we have objects changed by elementary transformations, and we add small changes to the shape locally, e.g., shapes are cut and/or they have something added. Figure 7(b) presents example images from this dataset. Finally, in the third set, similar to the other two sets, the objects were modified by elementary transformations, and we add to the shape large changes locally. The large changes are made such that the shape is still recognizable. Example images from the third dataset are presented in Fig. 7(c).

Because the notion of a local small/large change is very subjective, we are not able to define it unambiguously. We could give some percent intervals indicating how the shape changed, but it would be good only for global changes. When we consider local changes as in our case, this kind of approach would not prove itself because locality is also a subjective notion. For a person locality means some small area of the shape and for another person practically the whole shape. So the notion of a local small/large change in the shape will be based on subjective feelings.

The MPEG7 CE-Shape-1 Part B database consists of 1400 silhouette images from 70 classes. Each class has 20 different shapes. Figure 8 presents some sample images from the MPEG7 CE-Shape-1 Part B base.

In the test we used several different divisions of the unit square. One was a partition into 1×1 parts which corresponds to global analysis. The other divisions were 2×2 and 4×4. Figure 9 presents the divisions of the image which were used in the experiments. The number of

Fig. 6. Base images used to create three datasets.



(a) (b)

(c)

Fig. 7. Example images from the authors' base: elementary transformations (a), local small changes (b), local large changes (c).

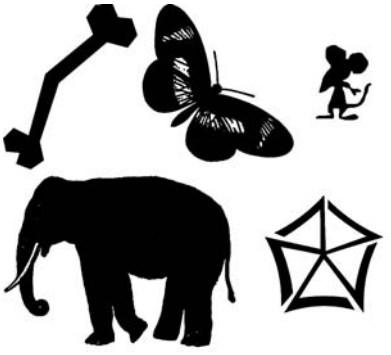
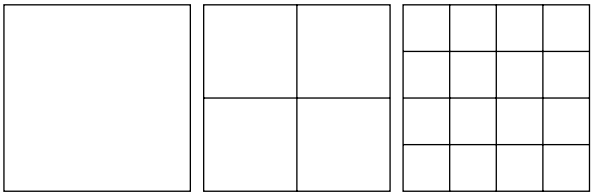


Fig. 8. Example images from the MPEG7 CE-Shape-1 Part B base.

transformations used in fractal compression of each sub-image depends on the division. For the 1×1 division we used 256 transformations per sub-image (16×16 range blocks division), for the 2×2 division 64 transformations per sub-image (8×8 range blocks division), and 16 transformations per sub-image (4×4 range blocks division) for the 4×4 division. So in each case the PIFS with division consists of 256 transformations.



(a) (b) (c)

Fig. 9. Divisions of the image into sub-images: 1×1 (a), 2×2 (b), 4×4 (c).

To estimate the error rate, we used the leave-one-out method (Witten and Frank, 2005) for the three datasets created by the authors, and for the MPEG7 CE-Shape-1 Part B base we used a stratified 10-fold cross validation (Witten and Frank, 2005).

To compare our method, in the test we used several fractal recognition methods: the Neil–Curtis method (Neil and Curtis, 1997) (N–C method), a method which uses PIFS coefficients (Chandran and Kar, 2002) (coeff. method), the multiple mapping vector accumulator method (Mozaffari *et al*., 2006) (MMVA method). In the case of the MMVA method, only matrices of size 5×5 and the Euclidean distance were used. Each of the method was implemented in the original form (partition 1×1) and in the modified form using the PIFS with division.

All the tested methods were implemented in Matlab, and the tests were performed on a computer with the AMD Athlon x2 6400+ processor, 4GB DDR2 RAM memory and with Microsoft Windows XP installed.

Tables 1(a)–(c) present the results of tests for the authors' base with elementary transformation. From the tables we clearly see that with smaller divisions into sub-images the error of all methods has decreased and the time of the test has shortened (even three times). Moreover, we see that the Neil–Curtis method and the fractal dependence graph method yielded the lowest values of the error (1%) for the 4×4 division, while the time of the test for the Neil–Curtis method was the longest. The fractal dependence graph method obtained the shortest time. From the results we see that all the methods for the 4×4 division yielded the error of 1–2%, which shows that these methods are not sensitive to elementary transformations (translation, rotation, change of scale).

Tables 2(a)–(c) present the results of tests for the authors' base with local small changes. From the results we see that the values of the error are bigger than in the case of the base with elementary transformations. In much the same way as in the previous case, the values of the error decreased with the smaller division into sub-images. The best results were also produced in the 4×4 case. The value of 1% was produced by the fractal dependence graph method, all the other methods yielded errors of 3–4%. The times of the test has shortened with the smaller division. The smallest reduction of the time was obtained for the Neil–Curtis method—only 0.007% between the time for the 1×1 division and 4×4 division, and the biggest reduction was for the fractal dependence graph method—70% between the time for the 1×1 division and 4×4 division. All the tested methods produced the error rate between 1% and 4%, which shows that the methods are robust to local small changes in shape.

Tables 3(a)–(c) present the results of tests for the authors' base with local large changes. From the obtained results again we see that with the smaller division of the image into sub-images the values of the error decreased and the times of the tests are shorter. Moreover, we see that the fractal dependence graph method and the MMVA method yielded the lowest value of the error (7%) for the 4×4 division. The Neil–Curtis method for the 4×4 division produced the value of the error only about 1% worse than the best methods. Similarly to the previous tests, the fractal dependence graph method yielded the shortest time (2190 s) for the 4×4 division.

Tables 4(a)–(c) present the results of tests for the MPEG7 CE-Shape-1 Part B base. From the tables we see that for the division 1×1 the best results are given by the Neil–Curtis method (29.45%). The fractal dependence graph method concedes the Neil–Curtis method by about 6.43%. When we look at the results for the 2×2 division, we see that the values of the error decreased. Also in this case the Neil–Curtis method is the best (19.17%), but the difference between this method and the fractal dependence method has decreased to 0.2%. The other methods have the error rate greater than 40%. And, finally, for the division 4×4, we see a further decrease in the error rate. The difference in the value of the error between the fractal dependence graph method and the Neil–Curtis method decreased to 0.14% on the advantage of the Neil–Curtis method which produced the error value of 18.02%. The times of the tests in the case of the MPEG7 base were not measured, but we noticed that the same dependency between the division into sub-images and time occurs that in the case of the authors' base, i.e., the time for the Neil–Curtis method decreased a little and for the other methods the time decreased significantly with the smaller division.

Table 5 presents the error rates for nonfractal methods known from the literature. The table comes from the work of Xu *et al*. (2009). Comparing fractal methods with the ones from Table 5, we see that only the Neil–Curtis and the proposed FDG method for division 4×4 are bet-

Table 1. Results of tests for the author's base—elementary transformations for divisions: 1×1 (a), 2×2 (b), 4×4 (c).

(a)

Method	Error [%]	Time [s]
FDG	2.00	8044
N-C	2.00	17917
coeff.	4.00	7915
MMVA	10.00	7964

(b)

Method	Error [%]	Time [s]
FDG	2.00	4824
N-C	2.00	17824
coeff.	3.00	4874
MMVA	3.00	5195

(c)

Method	Error [%]	Time [s]
FDG	1.00	2462
N-C	1.00	17738
coeff.	2.00	2671
MMVA	2.00	2962

Table 2. Results of tests for the authors' base—local small changes for divisions: 1×1 (a), 2×2 (b), 4×4 (c).

(a)

Method	Error [%]	Time [s]
FDG	6.00	7260
N-C	4.00	16448
coeff.	11.00	7123
MMVA	18.00	7054

(b)

Method	Error [%]	Time [s]
FDG	3.00	4065
N-C	3.00	16381
coeff.	10.00	4393
MMVA	12.00	4535

(c)

Method	Error [%]	Time [s]
FDG	1.00	2204
N-C	3.00	16331
coeff.	4.00	2465
MMVA	3.00	2706

Table 3. Results of tests for the authors' base—local large changes for divisions: 1×1 (a), 2×2 (b), 4×4 (c).

(a)

Method	Error [%]	Time [s]
FDG	14.00	6684
N-C	11.00	16625
coeff.	37.00	6772
MMVA	32.00	6734

(b)

Method	Error [%]	Time [s]
FDG	7.00	3998
N-C	9.00	16617
coeff.	20.00	4302
MMVA	13.00	4422

(c)

Method	Error [%]	Time [s]
FDG	7.00	2190
N-C	8.00	16622
coeff.	16.00	2639
MMVA	7.00	2667

Table 4. Results of tests for the MPEG7 CE-Shape-1 Part B base for divisions: 1×1 (a), 2×2 (b), 4×4 (c).

(a)

Method	Error [%]
FDG	35.88
N-C	29.45
coeff.	52.74
MMVA	49.03

(b)

Method	Error [%]
FDG	19.37
N-C	19.17
coeff.	42.89
MMVA	42.03

(c)

Method	Error [%]
FDG	18.16
N-C	18.02
coeff.	32.88
MMVA	24.59

ter than the three last methods from the table, i.e., the curvature scale space, the generative model, shape contexts. The other two methods (MMVA, coefficient method) for all the divisions considered are worse than all the methods from the table.

Table 5. Error rates of nonfractal methods obtained on the MPEG7 CE-Shape-1 Part B base.

Method	Error [%]
Contour flexibility	10.69
Shape-tree	12.30
HPM-Fn	13.65
Inner distance	14.60
Multiscale representation	15.07
Polygonal multiresolution	15.67
Chance probability function	17.31
Curvature scale space	18.88
Generative model	19.97
Shape contexts	23.49

8. Conclusions

A modification of the standard fractal image compression and a new recognition method of 2D shapes has been presented in this paper. The method was based on a dependence graph. The modification of the compression scheme using the PIFS with division led to a significant decrease in the recognition error. Moreover, the PIFS with division also brings an improvement in the speed of achieving the fractal description of the shape. This is due to the fact that, in the case of dividing the image into sub-images and then compressing them, the list of the domain blocks on which we perform the search process is smaller than in the classical case. We can obtain a further speed improvement in the compression using graphics hardware (Erra, 2005).

The proposed fractal dependency graph method produced the lowest values of the error for the authors' base and was the fastest method. In the case of the MPEG7 base, the best values of the error were obtained for the Neil–Curtis method, but the times compared with other methods, were huge. The fractal dependency graph me-

thod performed for the division 4×4 error only about 0.14% worse than the Neil–Curtis method, but the time was several times better.

In our further work we will concentrate on taking into account the number of matching sub-images in the similarity measure, which may bring a further decrease in the value of error. Moreover, we shall perform tests with other divisions of the image into sub-images to see the influence of different divisions on the recognition rate. Furthermore, because the value of the error decreases with the smaller sub-images division, we shall search for an optimal division into sub-images. We shall also try to bring the division into sub-images into other known fractal recognition methods. The correct orientation used to align objects is very simple so there is research under way to find a better method for aligning the objects.

All the fractal methods of 2D shape recognition are based on the silhouette of the shape and none of them uses the shape contour. Thus we shall try to develop a shape descriptor using fractal descriptions of the contour. In the literature there are many other methods, than fractal compression, of finding a fractal description of the contour (Gdawiec, 2009a; Skarbek *et al*., 1996), and we shall try to use these methods in our further research.

Acknowledgment

The authors would like to thank Professor W. Kotarski for his perspicacious comments and suggestions. Special thanks go to the anonymous reviewers for their valuable comments that helped to improve the presentation of this paper.

References

Attalla, E. and Siy, P. (2005). Robust shape similarity retrieval based on contour segmentation polygonal multiresolution and elastic matching, *Pattern Recognition* **38**(12): 2229–2241.

Barni, M. (2006). *Document and Image Compression*, CRC Press, Boca Raton, FL.

Barnsley, M. (1988). *Fractals Everywhere*, Academic Press, Boston, MA.

Belongie, S., Malik, J. and Puzicha, J. (2002). Shape matching and object recognition using shape contexts, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(4): 509–522.

Bruno, O.M., Plotze, R.d.O., Falvo, M. and Castro, M.d. (2008). Fractal dimension applied to plant identification, *Information Science* **178**(12): 2722–2733.

Burger, W. and Burge, M.J. (2008). *Digital Image Processing: An Algorithmic Introduction Using Java*, Springer, New York, NY.

Chandran, S. and Kar, S. (2002). Retrieving faces by the PIFS fractal code, *6th IEEE Workshop on Applications of Computer Vision, Orlando, FL, USA*, pp. 8–12.

Chang, Y.F., Lee, J.C., Mohd Rijal, O. and Syed Abu Bakar, S.A.R. (2010). Efficient online handwritten Chinese character recognition system using a two-dimensional functional relationship model, *International Journal of Applied Mathematics and Computer Science* **20**(4): 727–738, DOI: 10.2478/v10006-010-0055-x.

Dey, P. (2005). Basic principles and applications of fractal geometry in pathology—A review, *Analytical & Quantitative Cytology & Histology* **27**(5): 284–290.

Domaszewicz, J. and Vaishampayan, V.A. (1995). Graph-theoretical analysis of the fractal transform, *1995 International Conference on Acoustics, Speech, and Signal Processing, Detroit, MI, USA*, Vol. 4, pp. 2559–2562.

Erra, U. (2005). Toward real time fractal image compression using graphics hardware, *in* G. Bebis, R. Boyle, D. Koracin and B. Parvin (Eds.), *Advances in Visual Computing*, Lecture Notes in Computer Science, Vol. 3804, Springer, Heidelberg, pp. 723–728.

Felzenszwalb, P.F. and Schwartz, J. (2007). Hierarchical matching of deformable shapes, *IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA*, Vol. 1, pp. 1–8.

Fisher, Y. (1995). *Fractal Image Compression: Theory and Application*, Springer-Verlag, New York, NY.

Gdawiec, K. (2009a). Fractal interpolation in modeling of 2D contours, *International Journal of Pure and Applied Mathematics* **50**(3): 421–430.

Gdawiec, K. (2009b). *Local Fractal Analysis in Recognition of 2D Shapes*, Ph.D. thesis, University of Silesia, Sosnowiec, (in Polish).

Ghazel, M., Freeman, G.H. and Vrscay, E.R. (2003). Fractal image denoising, *IEEE Transactions on Image Processing* **12**(12): 1560–1578.

Golub, G.H. and van Loan, C.F. (1996). *Matrix Computations,* 3rd Edn., The Johns Hopkins University Press, Baltimore, MD.

Harris, J.M., Hirst, J.L. and Mossinghoff, M.J. (2008). *Combinatorics and Graph Theory,* 2nd Edn., Springer, New York, NY.

Huang, K. and Yan, H. (2000). Signature verification using fractal transformation, *15th International Conference on Pattern Recognition, Barcelona, Spain*, Vol. 2, pp. 855–858.

Kolumbán, J., Soós, A. and Varga, I. (2003). Self-similar random fractal measures using contraction method in probabilistic metric spaces, *International Journal of Mathematics and Mathematical Sciences* **2003**(52): 3299–3313.

Kouzani, A.Z. (2008). Classification of face images using local iterated function systems, *Machine Vision and Applications* **19**(4): 223–248.

Latecki, L.J., Lakamper, R. and Eckhardt, T. (2000). Shape descriptors for non-rigid shapes with a single closed contour, *IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, SC, USA*, Vol. 1, pp. 424–429.

Ling, H. and Jacobs, D.W. (2005). Using the inner-distance for classification of articulated shapes, *IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA*, Vol. 2, pp. 719–726.

Linnell, T.A. and Deravi, F. (2004). Mapping vector accumulator: fractal domain feature for character recognition, *Electronic Letters* **40**(22): 1406–1407.

Mandelbrot, B. (1983). *The Fractal Geometry of Nature*, W.H. Freeman and Company, New York, NY.

Meng, D., Cai, X., Su, Z. and Li, J. (2009). Photorealistic terrain generation method based on fractal geometry theory and procedural texture, *2nd IEEE International Conference on Computer Science and Information Technology, Beijing, China*, pp. 341–344.

Mokhtarian, F. and Bober, M. (2003). *Curvature Scale Space Representation: Theory, Applications, and MPEG-7 Standardization*, Springer, Heidelberg.

Mozaffari, S., Faez, K. and Faradji, F. (2006). One dimensional fractal coder for online signature recognition, *18th International Conference on Pattern Recognition, Hong Kong, China*, Vol. 2, pp. 857–860.

Neil, G. and Curtis, K.M. (1997). Shape recognition using fractal geometry, *Pattern Recognition* **30**(12): 1957–1969.

Nikiel, S. (2007). A proposition of mobile fractal image decompression, *International Journal of Applied Mathematics and Computer Science* **17**(1): 129–136, DOI: 10.2478/v10006-007-0012-5.

Peters, E.E. (1994). *Fractal Market Analysis: Applying Chaos Theory to Investment and Economics*, John Wiley & Sons Inc., New York, NY.

Plotze, R.d.O., Falvo, M., Páuda, J.G., Bernacci, L.C., Vieira, M.L.C., Oliveira, G.C.X. and Bruno, O.M. (2005). Leaf shape analysis using the multiscale Minkowski fractal dimension, a new morphometric method: A study with passiflora (passifloraceae), *Canadian Journal of Botany* **83**(3): 287–301.

Prusinkiewicz, P. and Lindenmayer, A. (1996). *The Algorithmic Beauty of Plants*, Springer-Verlag, New York, NY.

Skarbek, W. and Ignasiak, K. (1996). Asynchronous nonlinear fractal operators and their applications, *Image Processing & Communications* **2**(2): 3–20.

Skarbek, W., Ignasiak, K. and Ghuwar, M. (1996). Fractal representation of planar shapes, *in* S. Miguet, A. Montanvert and S. Ubéda (Eds.), *Discrete Geometry for Computer Imagery*, Lecture Notes in Computer Science, Vol. 1176, Springer, Heidelberg, pp. 73–84.

Tu, Z. and Yuille, A.L. (2004). Shape matching and recognition using generative models and informative features, *8th European Conference on Computer Vision, Prague, Czech Republic*, pp. 195–209.

Witten, I.H. and Frank, E. (2005). *Data Mining—Practical Machine Learning Tools and Techniques*, 2nd Edn., Morgan Kaufmann Publishers, San Francisco, CA.

Xu, C., Liu, J. and Tang, X. (2009). 2D shape matching by contour flexibility, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31**(1): 180–186.

Yokoyama, T., Sugawara, K. and Watanabe, T. (2004). Similarity-based image retrieval system using partitioned iterated function system codes, *Artifical Life and Robotics* **8**(2): 118–122.

Zhao, G., Cui, L. and Li, H. (2007). Gait recognition using fractal scale, *Pattern Analysis & Applications* **10**(3): 235–246.

Krzysztof Gdawiec received the M.Sc. degree in mathematics from the University of Silesia (Poland) in 2005, and the Ph.D. degree in computer science from the same university in 2010. Currently he is an assistant professor at the Institute of Computer Science of the University of Silesia. He is an author or co-author of several journal and conference publications. His main research interests include applications of fractal geometry, pattern recognition, and computer graphics.

Diana Domańska holds the B.Sc. degree in computer science and the M.Sc. degree in mathematics from the University of Silesia (Poland). Currently she is a Ph.D. student at the Institute of Computer Science of the same University. She is an author or co-author of several journal and conference publications. Her research interests cover data mining, fuzzy numbers, shape recognition, and fractals.

Received: 29 October 2010
Revised: 17 April 2011