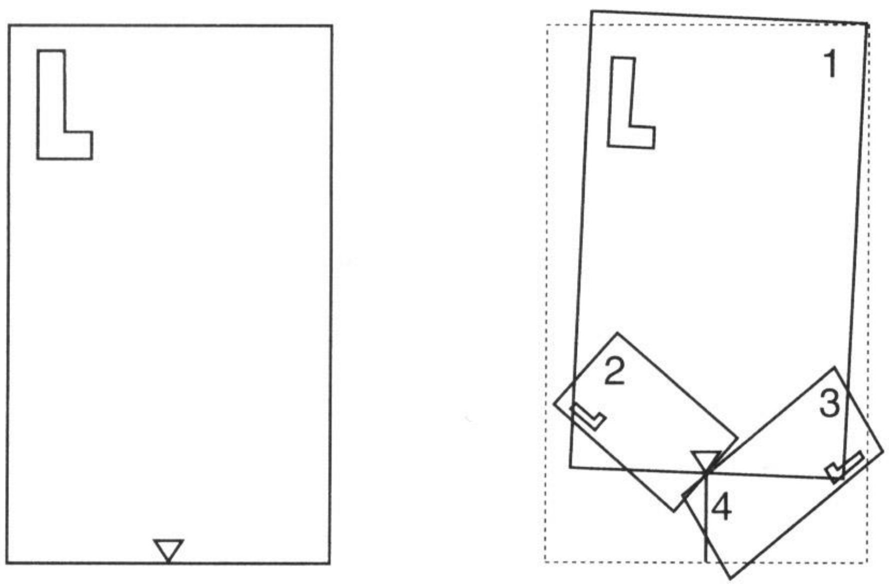
Fractals

Part 5 : Fractal compression

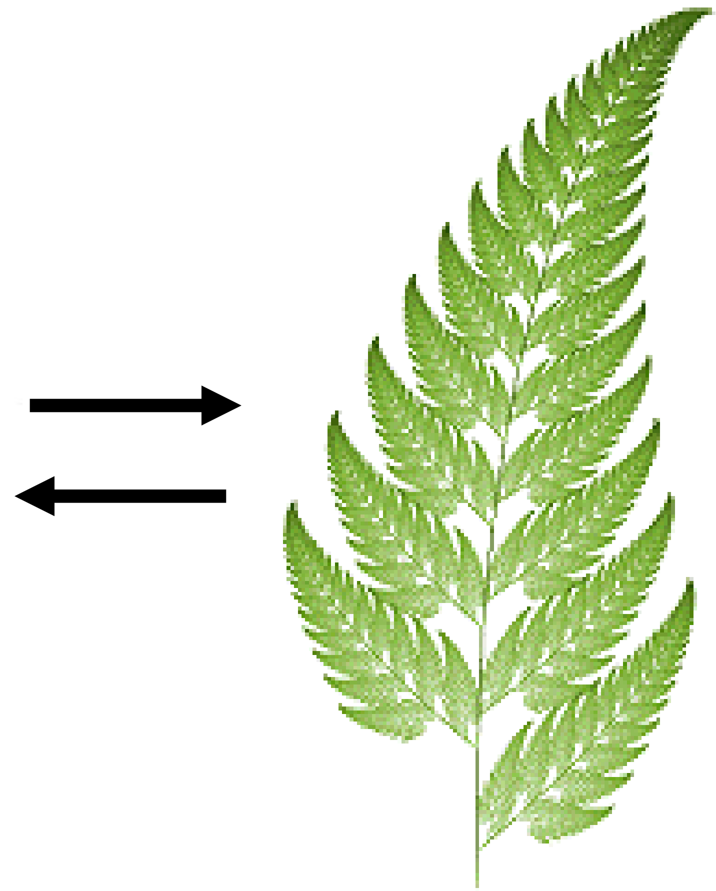
Martin Samuelčík
Department of Applied Informatics

IFS



Initial Image

Stage 1



Using IFS

- Model natural objects
- Simulate natural cases
- Inverse problem
- Reconstructing objects
- Fractal interpolation
- Image compression and coding

Coding of IFS

- Let’s have fractal as attractor of IFS
- We can describe this image very easily
- Perfect coding
- Independent of scale
- Perfect compression
- Complex shape <-> low count of bits

Generalization

- Using previous approach for any image
- Good compression
- Finding similarities in image
- We want image as attractor
- Or his good approximation
- Progressive showing of image

Image

- Image of pixels – in discrete space
- Finite resolution
- Generalized image for IFS
- I(x,y) in <0,1> for x,y in <0,1>
- Infinite resolution
- Then it is member of H(X)

Metric on Images

- Comparing 2 images
- Defining contractive transformation
- Basic metric
  - $\sup_{x,y \text{ in } <0,1>} |f(x,y)-g(x,y)|$
- Least square metric
  - Ai, bi are intensities of 2 images
  - Minimization of
  - R is then distance

$$R = \sum_{i=1}^{n} (s.a_i + o - b_i)^2$$

Compression of images

- In pixel space
- Storing pixels and using data compression
- Storing coefficients of DCT transformations
- All in discrete space
- Jpeg, Gif, Png, Bmp …

Collage theorem

- (X,d) complete metric space
- L in H(X), ε > 0
- IFS $\{X,f_1,\ldots,f_N\}$ with contr. factor *s*

$$h(L,\bigcup_{i=1}^{N} f_i(L)) < \varepsilon$$

- Then $h(L,x_W) < \dfrac{\varepsilon}{(1-s)}$
- Where x_W is attractor of IFS $\{X,f_1,\ldots,f_N\}$

Collage theorem 2

- If we cover precisely L with its affine copies, then attractor IFS approximate L with good precision
- For smaller ε we need more f_i
- Effective representation assume that transformations cover L and two transformation have no intersection

Improving IFS

- f_i:D_i -> R_i, i=1,…,N

$$f_i : \begin{pmatrix} x \\ y \\ z \end{pmatrix} : \begin{pmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \\ o_i \end{pmatrix} \qquad \bigcup_{i=1}^{N} R_i = I; R_i \cap R_j = 0, i \neq j$$

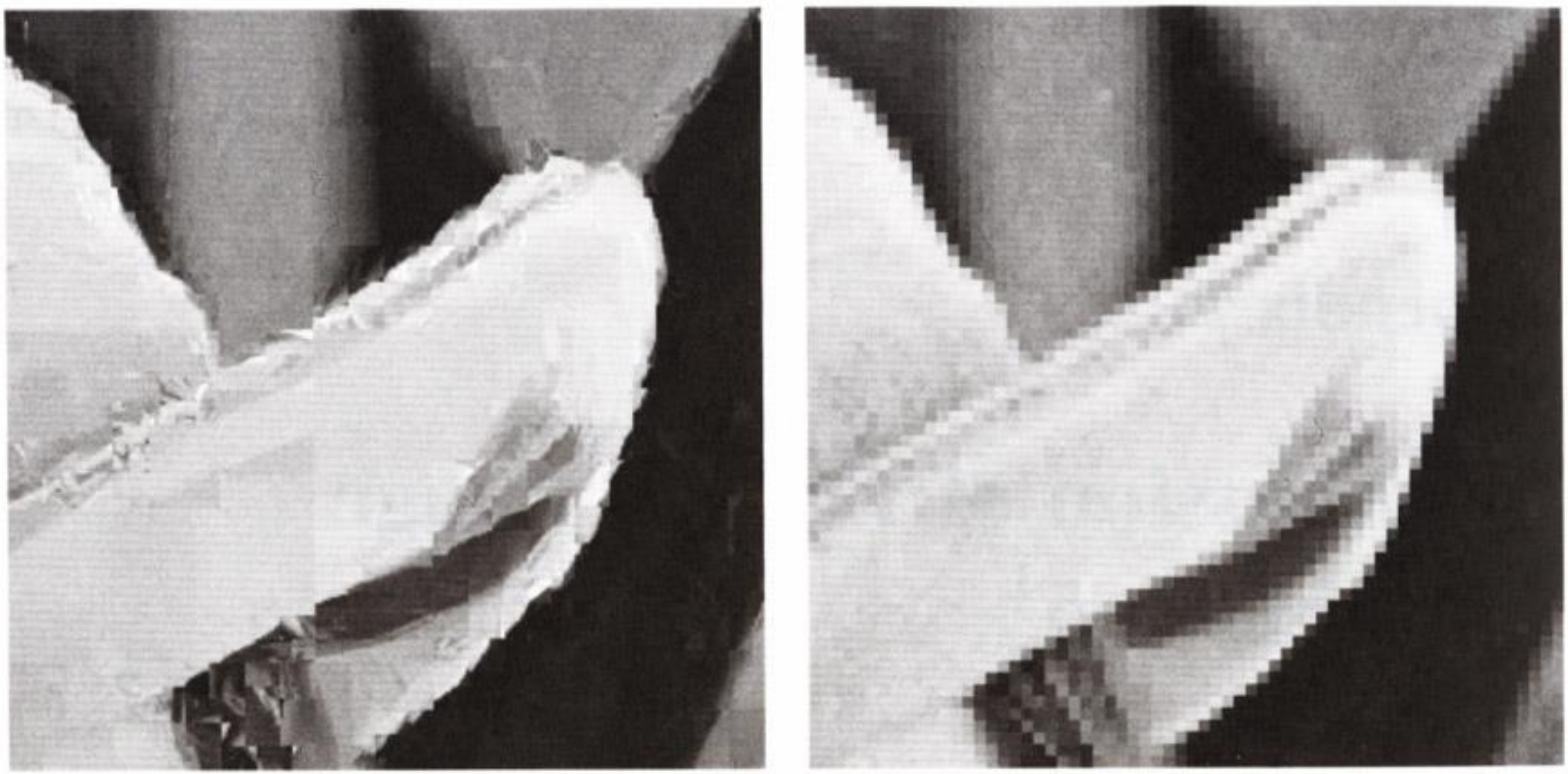
- $f_i(I)=f_i(x,y,I(x,y))$
- s_i,o_i – contrast, brightness
- f_i is restricted to D_i
- Partitioned IFS (PIFS)

Transformations

Fractal compression

- Barnsley
- Covered with copyright
- Perfect <-> useless
- Need to be optimalized
- Rapidly improved
- Now? Not used

Fractals vs Pixels



Encoding

- Now for given image I we want to find improved IFS $\{X, f_1, \ldots, f_N\}$
- W(I) should be very close to our image

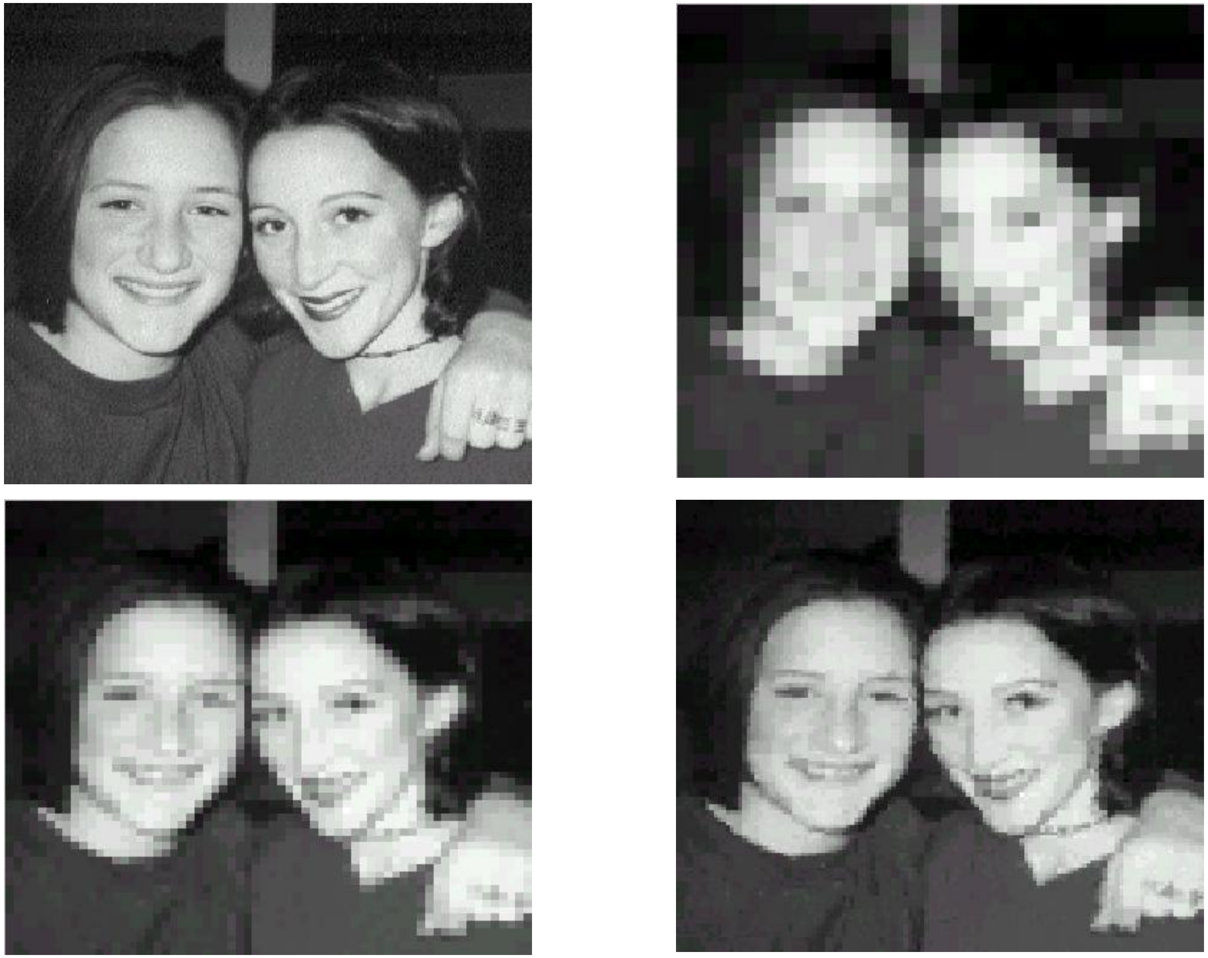
$$I \approx I' = W(I') \approx W(I)$$

- Minimal value of $h(I \cap R_i, f_i(I))$
- Have to find D_i, R_i, s_i, o_i
- Store only these values

Decoding

- We have IFS
- We can start with any image
- Should have lot of steps
- Using deterministic or stochastic algorithm
- No pixelization = scale detail
- Need to decode whole

Decoding 2



Simple example

- 256 x 256 image
- $R_1,..,R_{1024}$ – all non-overlaping 8x8 subimages
- $D_1,...,D_{58081}$ – all 16x16 subimages
- For each R_i find D_i so that minimizes

$$h(I \cap R_i, f_i(D_i))$$

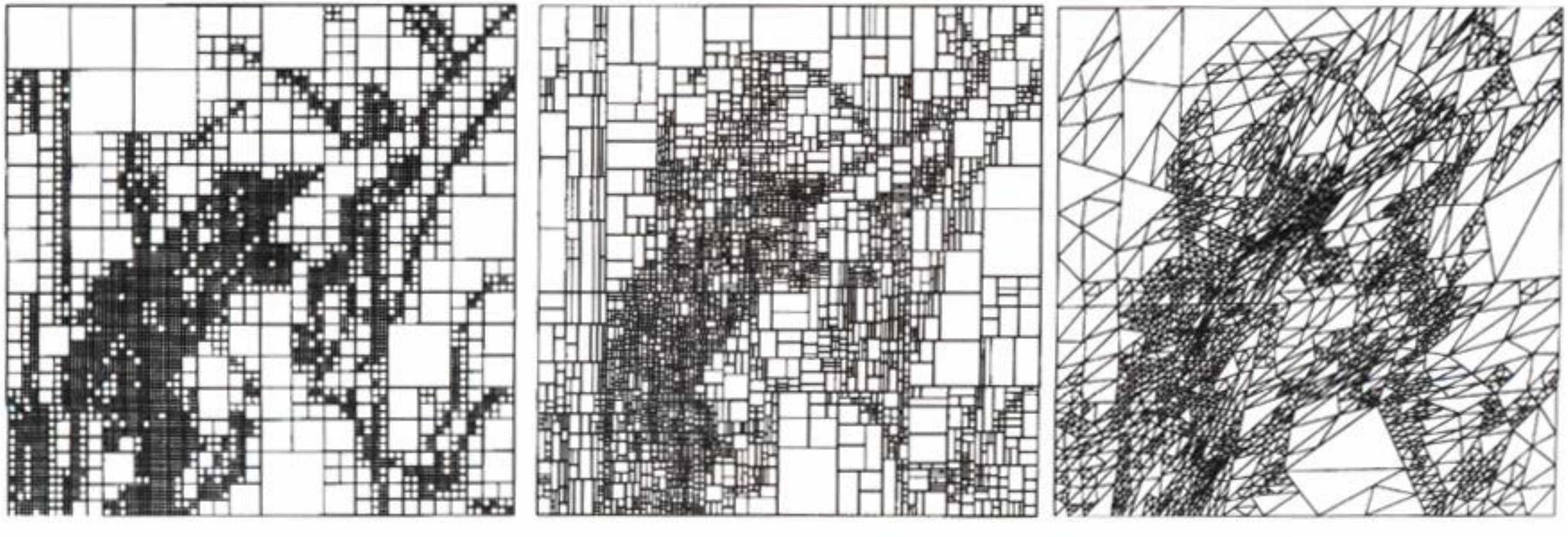
Simple example 2

- Minimalization using least square metric
- This metric gives us also s_i, o_i
- We get $f_1, \ldots, f_{1024}$
- We store this system
- 65536 -> 3968
- With decoding improving detail 8x8,4x4

Result

Partitioning image

- Uniform
- Quadtree, HV
- Triangular, Custom



Other results

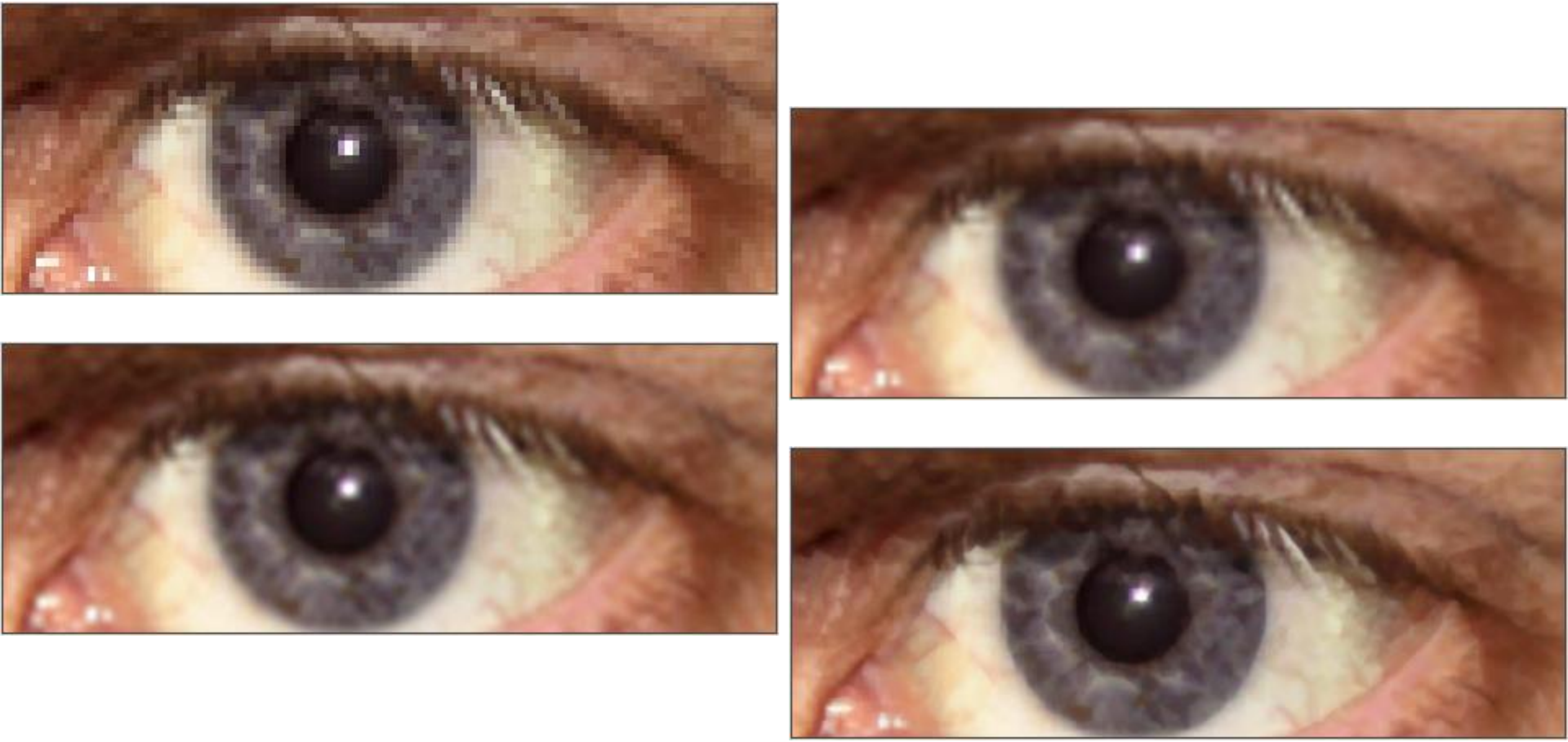
Optimizing

- Compression ratio <-> fidelity
- Encoding time
- Choosing ranges
- Choosing domains

Fractal interpolation

- Image encoded using fractal compression is “pixelation free”
- Good for expanding images – bilinear, cubic spline, statistical interpolation
- PIFS is created, used for expansion (contaction) of image and then discarded

Fractal interpolation



Search strategies

- Heavy brute force
- Light brute force
- Restricted area search
- Local spiral search
- Look same place
- Categorized search

Fractal compression

- Promising technology in the late 1980s
- Main competitor - JPEG
- Large advantage over JPEG at low image quality levels
- “Graduate Student Algorithm": lock a graduate student in a room with a computer until they solve your problem.

Fractal interpolation

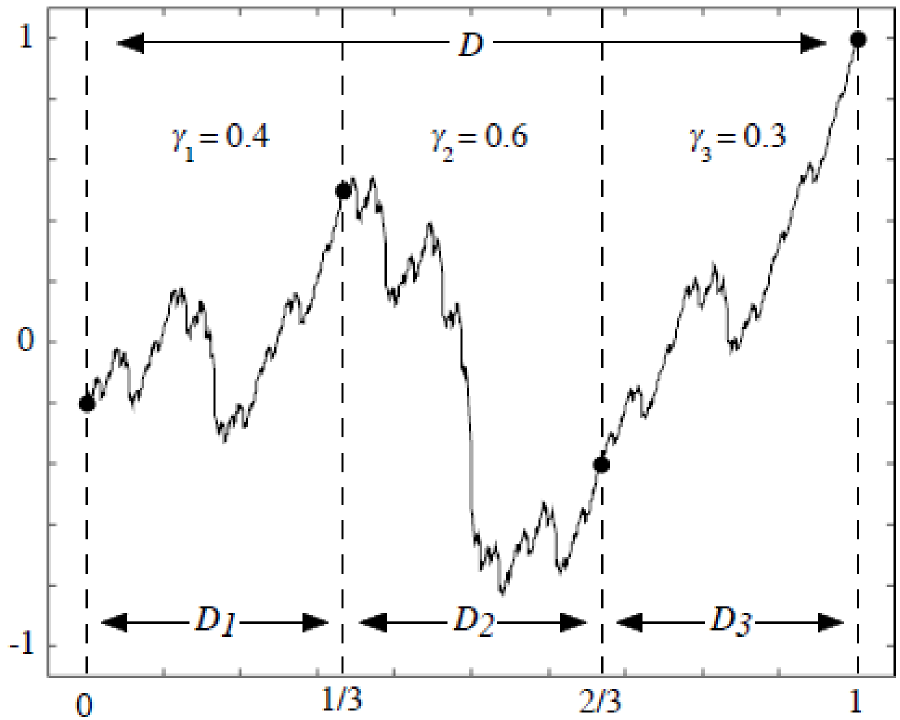
- Points $A_i=(x_i,y_i),i=0,..,M$
- Define function w such that $f(x_i)=y_i$
- f-attractor of IFS $(w_1,\ldots,w_M)$
- d_i is free parameter, $|d_i|<1$

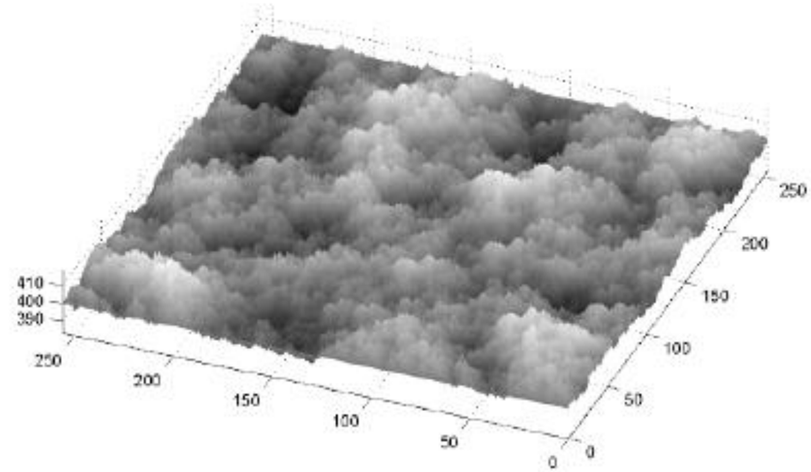
$$w_i\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_i & 0 \\ c_i & d_i \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \end{pmatrix}: \qquad i = 1,2,\ldots,M.$$

$$w_i\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} x_{i-1} \\ y_{i-1} \end{pmatrix} \quad \text{and} \quad w_i\begin{pmatrix} x_M \\ y_M \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix}: \qquad i = 1,2,\ldots,M$$

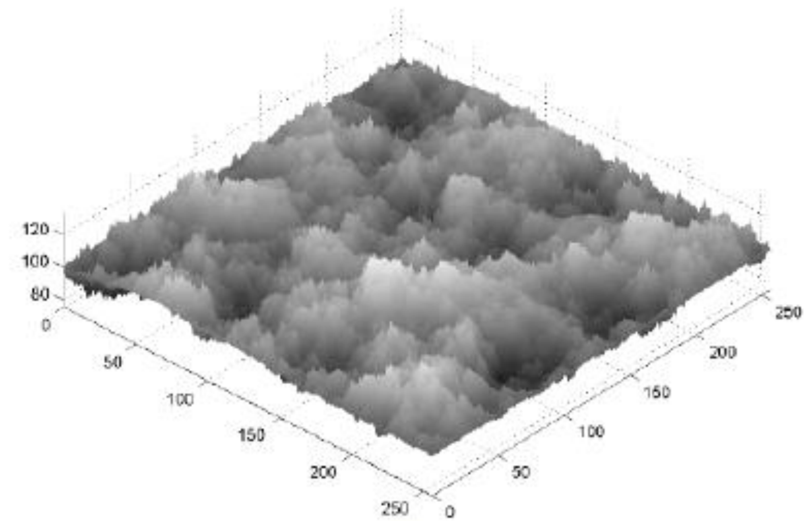
$$a_i = \frac{(x_i - x_{i-1})}{(x_M - x_0)}$$
$$c_i = \frac{(y_i - y_{i-1})}{(x_M - x_0)} - d_i\frac{(y_M - y_0)}{(x_M - x_0)}$$
$$e_i = \frac{(x_M x_{i-1} - x_0 x_i)}{(x_M - x_0)}$$
$$f_i = \frac{(x_M y_{i-1} - x_0 y_i)}{(x_M - x_0)} - d_i\frac{(x_M y_0 - x_0 y_M)}{(x_M - x_0)}$$

Interpolation





(a)



(b)

End

End of Part 5