Unit -V

Fractals and Self Similarity

S.Lakshmi Priya
Assistant Professor, CSE

The Fern

What are fractals?

- Fractals are geometric objects formed by iterations.
- Fractals are shapes that appear similar at all scales of magnification and is therefore often referred to as "infinitely complex."
- *Fractals are attractive because they exhibit similar characteristics to* the natural world.
- In computer graphics, we use fractals to create complex objects.
- Popularized by Benoit Mandelbrot

Fractal Geometry Methods

- Natural objects can be realistically described with fractal geometry methods.

- Procedures are used to model objects rather than equations.

- A fractal object has two basic characteristics
  - Infinite detail at every point - if we zoom in, we see more details of the object
  - Self-similarity between object parts and the overall features of the object

Fractal Dimension

- The amount of variations in the object detail is described with a number called **fractal dimension.**

$$D = \frac{\ln N}{\ln \frac{1}{s}}$$

- N – number of new line segments formed
- s- scaling factor.

Example of Graphics objects created using fractals

- Clouds
- Grass
- Fire
- Modeling mountains (terrain)
- Coastline
- Branches of a tree
- Surface of a sponge
- Cracks in the pavement

Fractal Property

- Level of detail remains the same as we zoom in
  - Example: surface roughness or profile same as we zoom in
- Types:
  - ***Exactly self-similar***: Region is enlarged the enlargement looks exactly like the original.
  - ***Statistically self-similar***: The wiggles and irregularities in the curve are same on the average

Fractal Generation Procedures

- A fractal object is generated by repeatedly applying a specified transformation function to points within a region of space
- If $P_0=(x_0,y_0,z_0)$ is initial point each iteration of a transformation function F generates successive levels

  $P_1=F(P_0)$, $P_2=F(P_1)$, $P_3=F(P_2)$, …
- Transformation function can be applied to points or to a set of primitives such as lines, curves, color areas or surfaces and solid objects

Fractal Generation Procedures

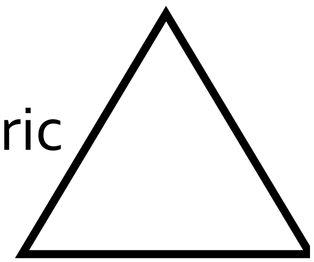
- Fractal objects by definition contain infinite detail , as the transformations functions are applied finite number of times.
- Objects we display will contain finite dimensions.
- The amount of detail in the graphical display of an object depends
  - On the number of iterations performed
  - The resolution of display system.

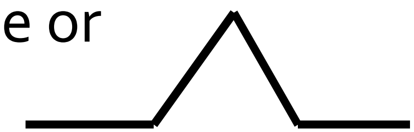
Classification of Fractals

- Self-Similar Fractals
- Self-Affine Fractals
- Invariant fractal sets

Classification of Fractals

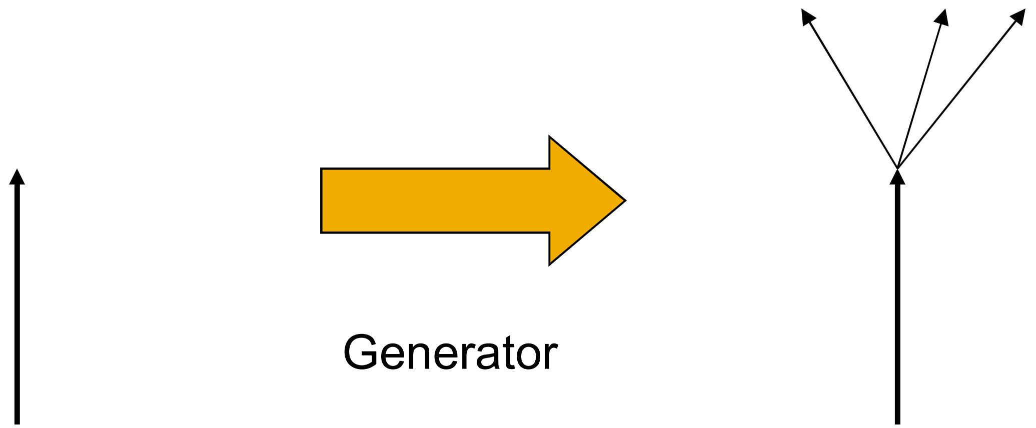
- **Self –similar:** Fractals have parts that are scaled-down versions of the entire object
  - Fractals start with an initiator, which is a given geometric shape
  - iteratively apply a generator, which is a pattern.
  - Apply same scaling parameter s to the overall shape or different scaling factor to scale down the subparts.



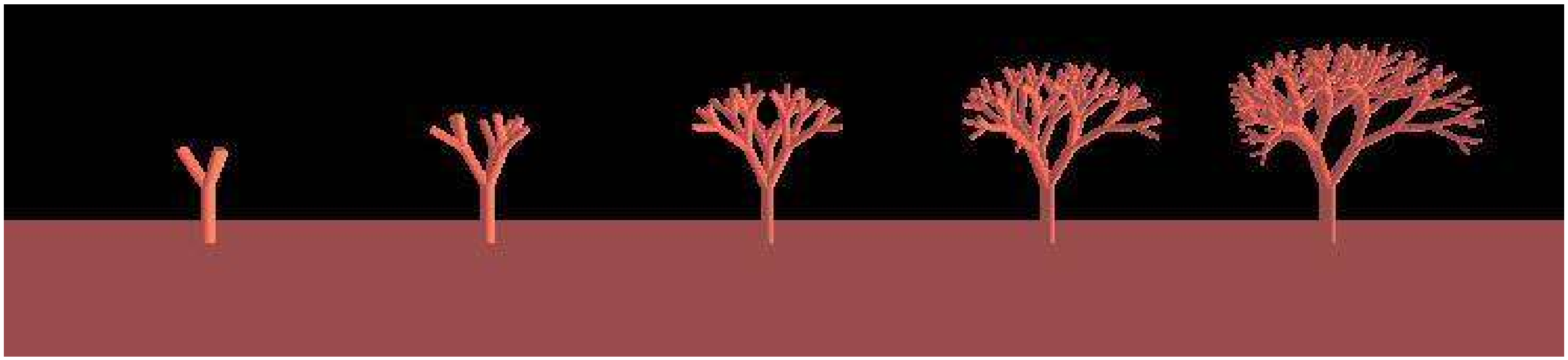


- When random variations are applied to scale down the subparts the fractal is called **statistically self-similar.**
  - Model trees, shrubs and other plants

Fractal Tree



Generator

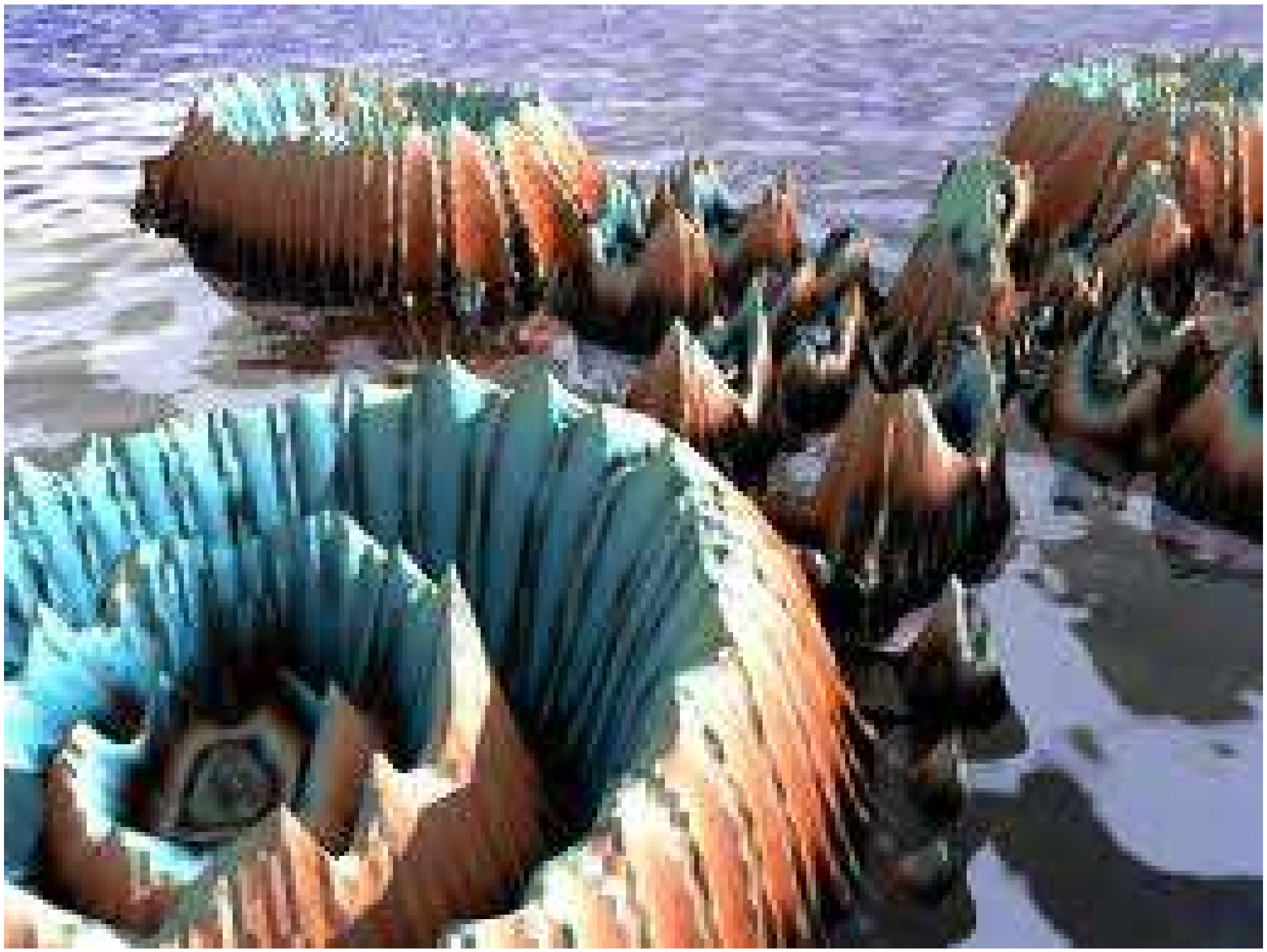


Iteration 1 Iteration 2 Iteration 3 Iteration 4 Iteration 5

Classification of Fractals

- **Self –Affine Fractals :** Formed with different scaling parameters sx, sy, sz in different coordinate systems.
  - Random variations are included to obtain statistically self affine fractals.
  - Terrain , water ,clouds are modeled using this method.

Example : Fractal Terrain



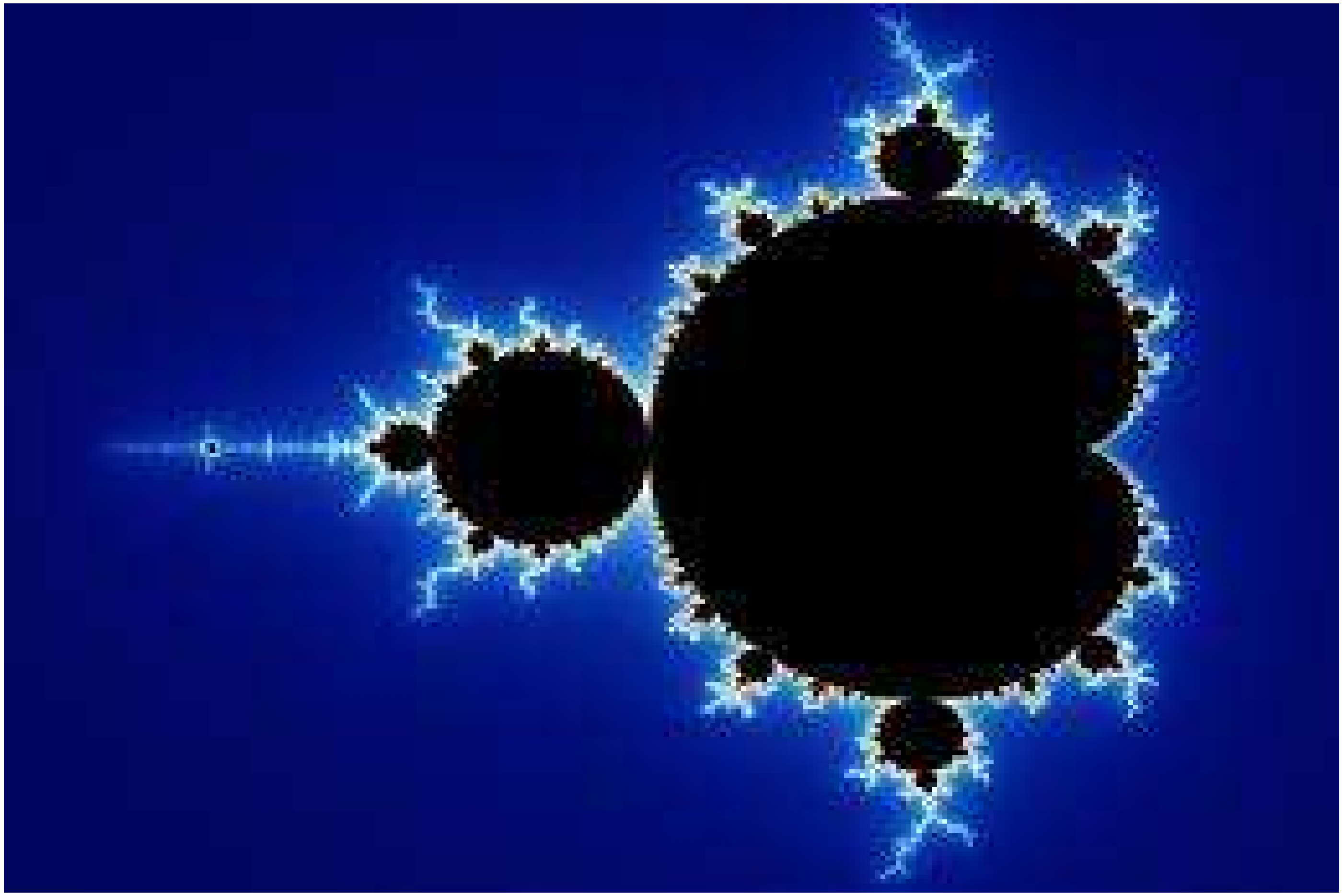
Example: Fractal Terrain

Courtesy: Mountain 3D Fractal Terrain software

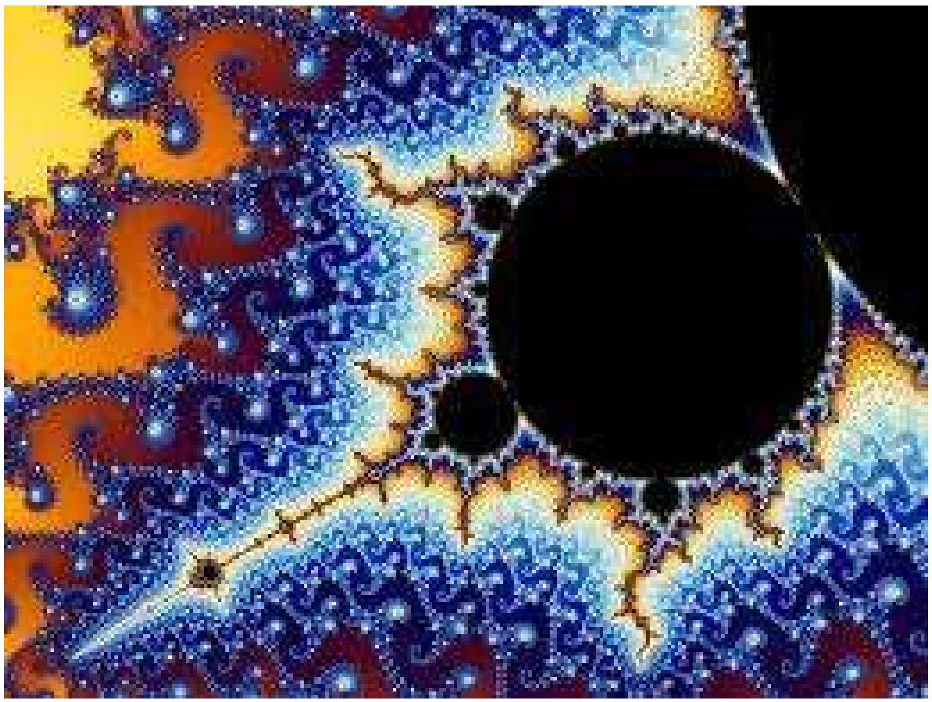
Classification of Fractals

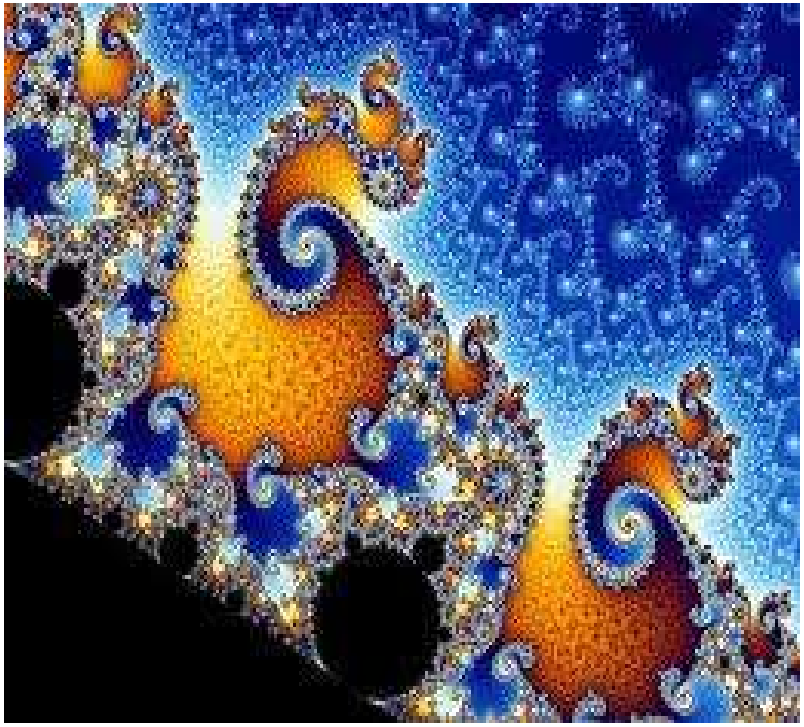
- **Invariant fractal sets:** Formed with nonlinear transformations.
  - Includes self-squaring fractals like Mandelbrot set formed with squaring functions in complex space.
  - Self inverse fractals formed with inversion procedures.

Example: Mandelbrot Set



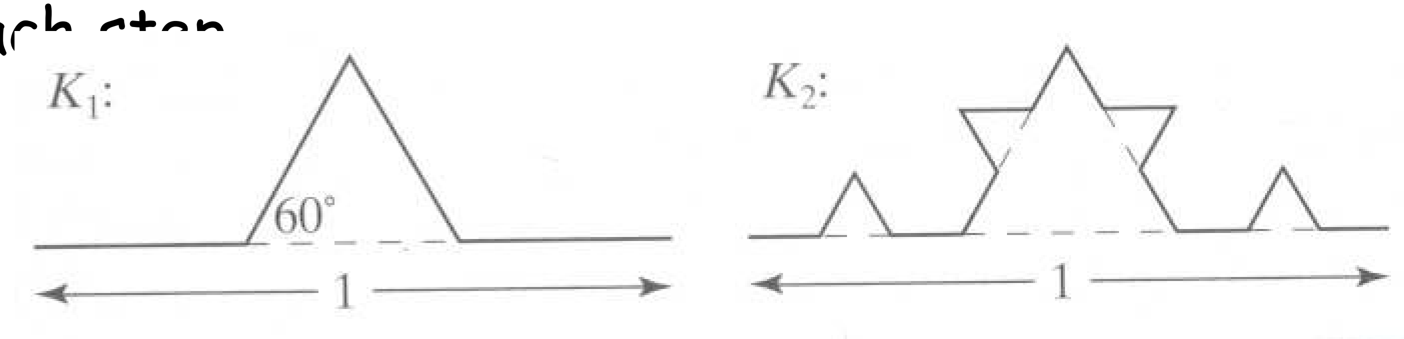
Example: Mandelbrot Set



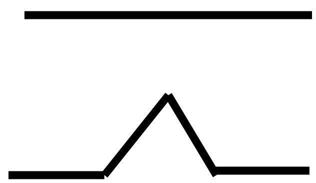


Koch Curves
(Self –Similar Fractal)

- Discovered in 1904 by Helge von Koch
- Start with straight line of length 1
- Recursively:
  - Divide line into 3 equal parts
  - Replace middle section with triangular bump, sides of length 1/3
  - Each line segment in the initiator is replaced with four equal length line segments at each step
  - Each new segment is increased by a factor 1/3 at each step



Koch Curves
(Self –Similar Fractal)



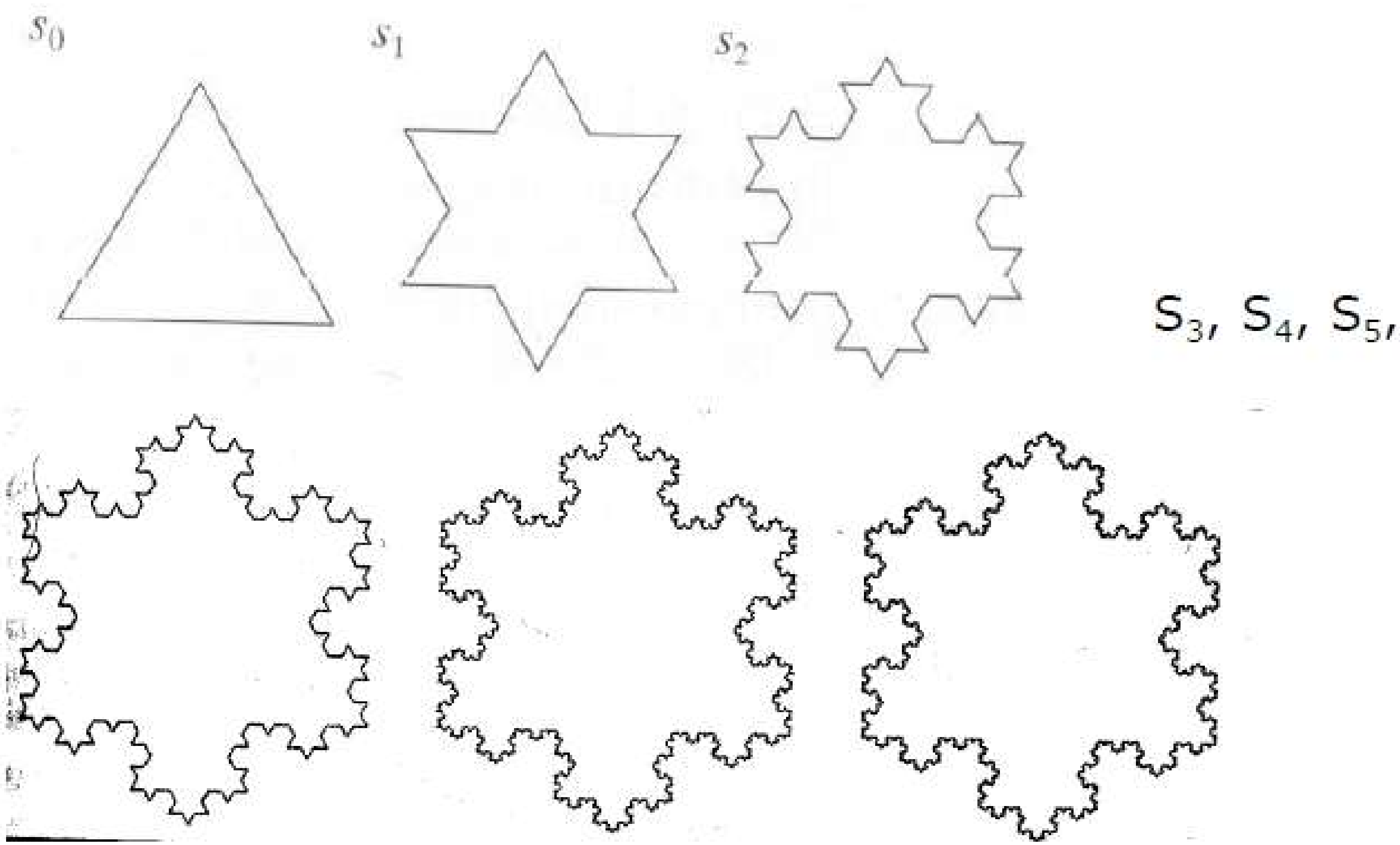
- Step 0: length = 1, s = 1/3
- Step 1: length = 4/3

- Scaling factor s is applied.

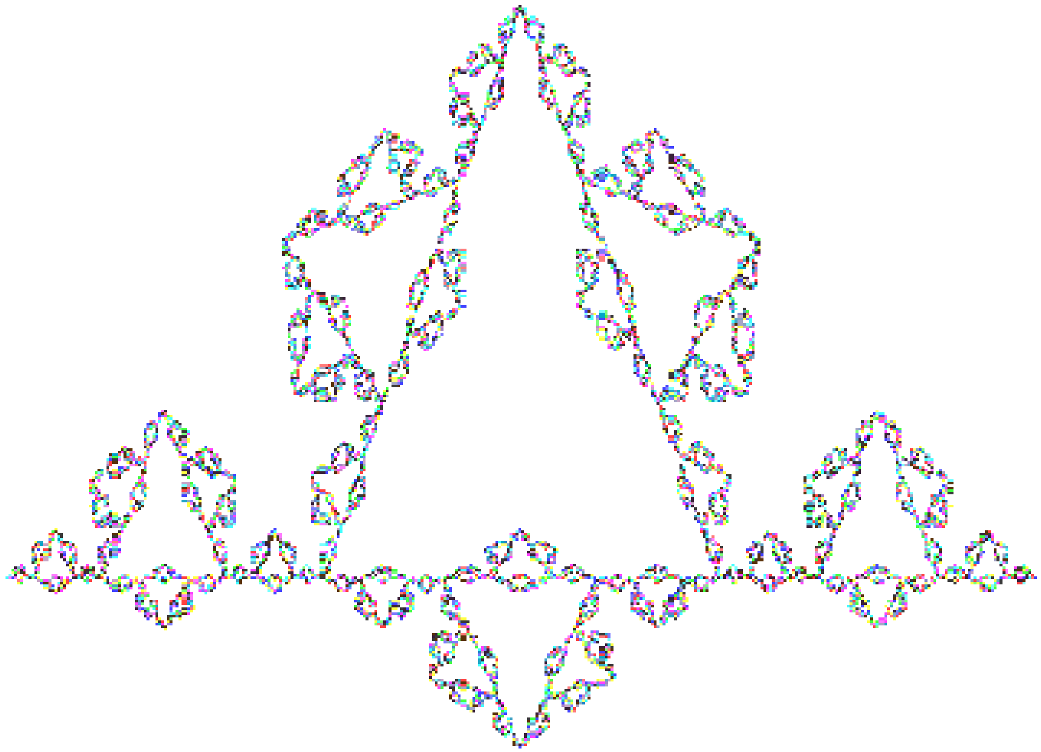
- This continues iteratively, where at each level, the segments are divided into thirds and the same pattern is repeated.

- The definition of the curve requires that this be repeated forever.
  - D = ln 4 / ln 3 = 1.26186

Koch Curves

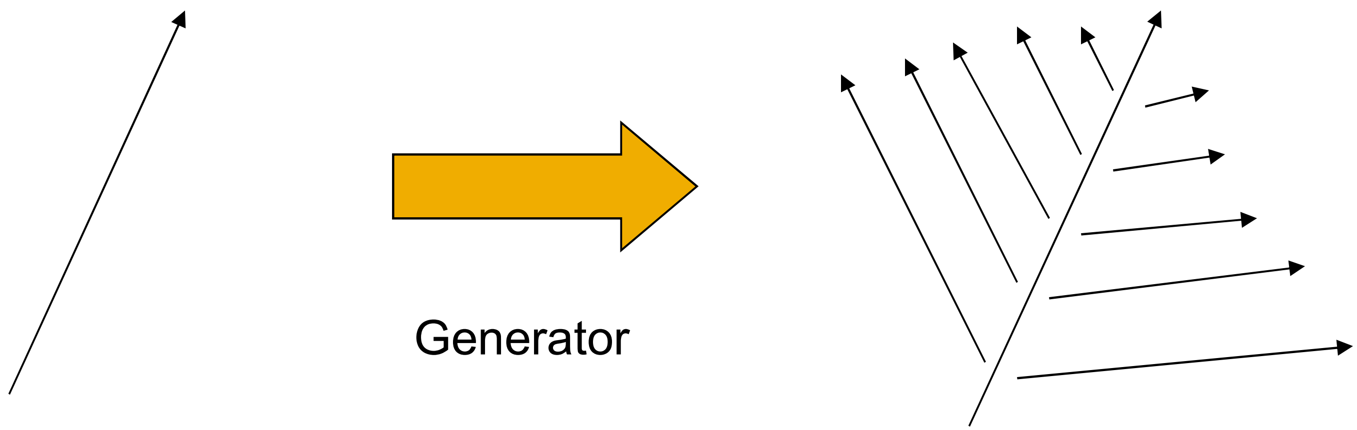


Koch Curve

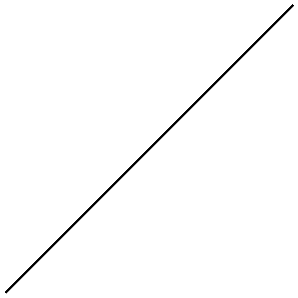


At level k, the length is $(4/3)^k$

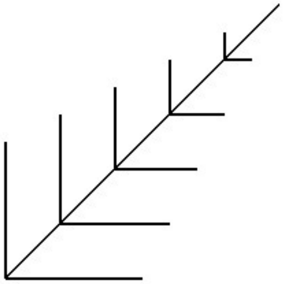
Fractal Fern



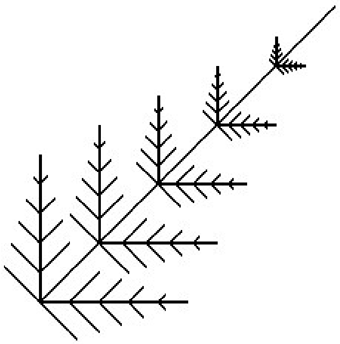
Generator



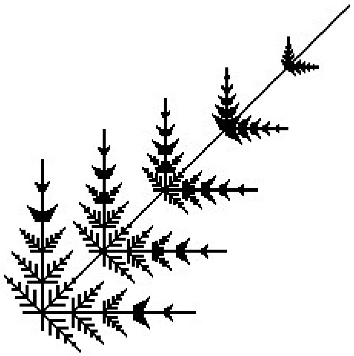
Iteration 0



Iteration 1



Iteration 2



Iteration 3

Application: Fractal Art

Iterated Function Systems (IFS)

- Recursively call a function
- IFS's converge to an image
- Self squaring fractals make use of this technique.
- Examples:
  - Julia- set
  - The Mandelbrot set

Self Squaring Fractals

- Repeatedly applying functions to points in the complex plane results in a remarkably complex image.
- A complex number can be represented as $z=x+iy$, where x and y are real numbers and i is the imaginary part.
- Use some self-squaring functions to generate fractal shapes.
- $f(z) = z^2+c$ where z and c are complex numbers.

Julia Set

- Suppose $f(z) = z^2 - 0.75$
- Here c= -0.75
- With c as constant, for different starting values of z, apply the function f(z). i.e.,

$$z_1 = f(z_0)$$

$$z_2 = f(z_1) \quad \text{and so on.}$$

- The set of points $\{z_0, z_1, z_2 \ldots\}$ is called the orbit of z_0 under f(z).
- These terms may tend towards infinity or remain bounded.

Julia Set

- The values of z for which the sequence remain bounded forms the set called as **Julia Set** for the given complex number c under f.
- Plot the points in the Julia set as black and others using different colors depending on how fast the sequence moves towards infinity.
- Need not iterate forever. Limit the number of iterations by using the math theorem that if the iterates fall beyond 2, then it will definitely tends towards infinity.

Julia set

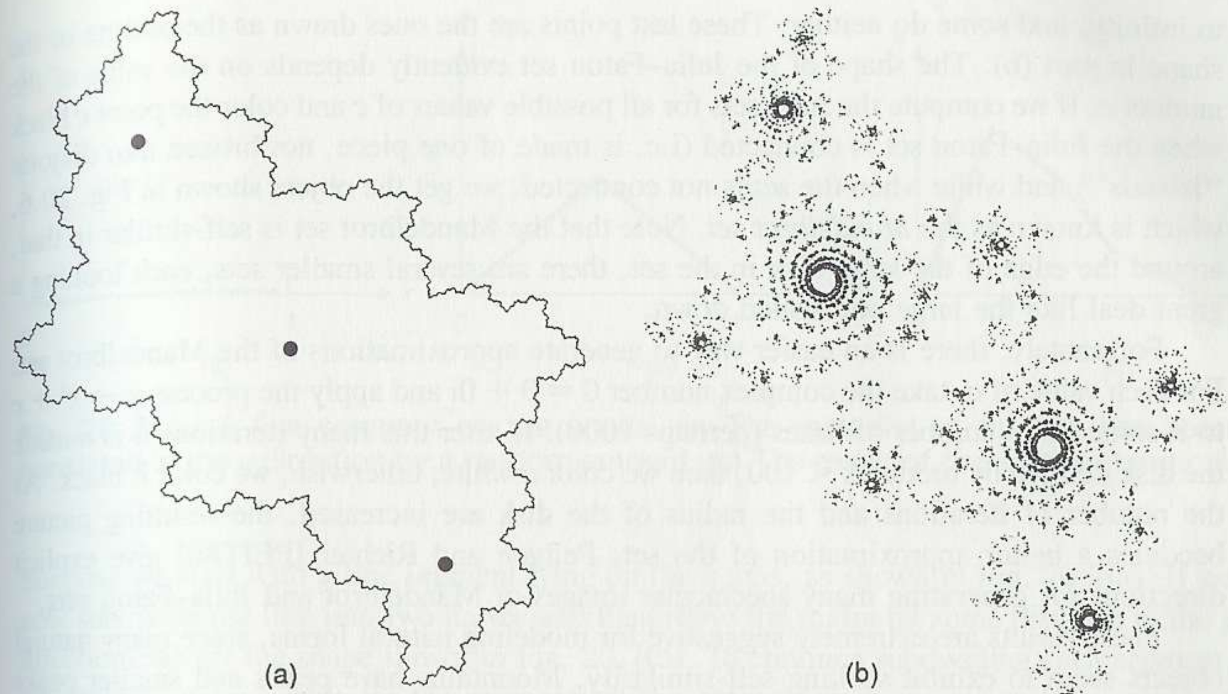
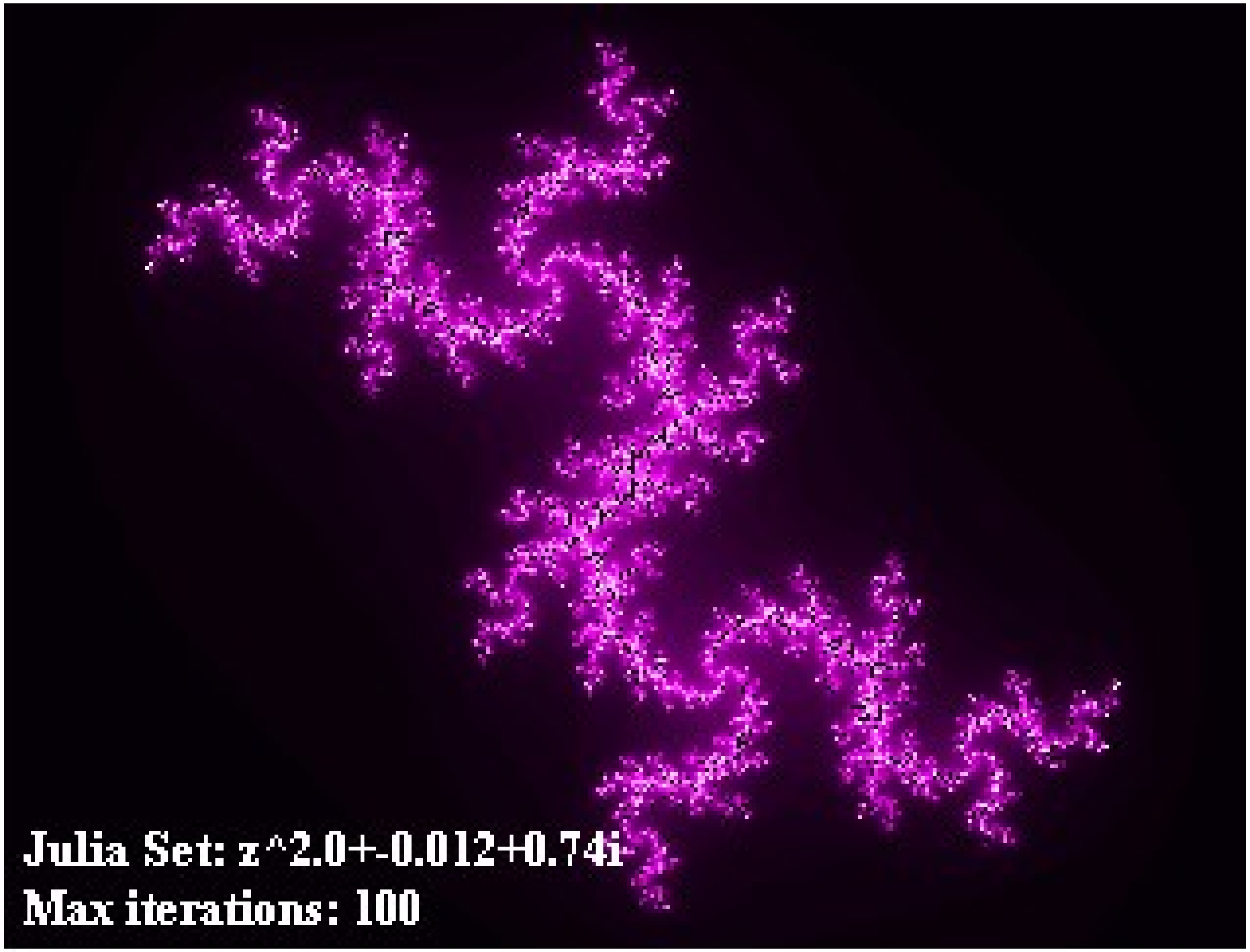


Fig. 20.5 (a) The Julia–Fatou set for $c = -0.12375 + 0.056805i$; (b) the Julia–Fatou set for $c = -0.012 + 0.74i$.

The most famous two fractal objects deserve mention here: the Julia–Fatou set and the

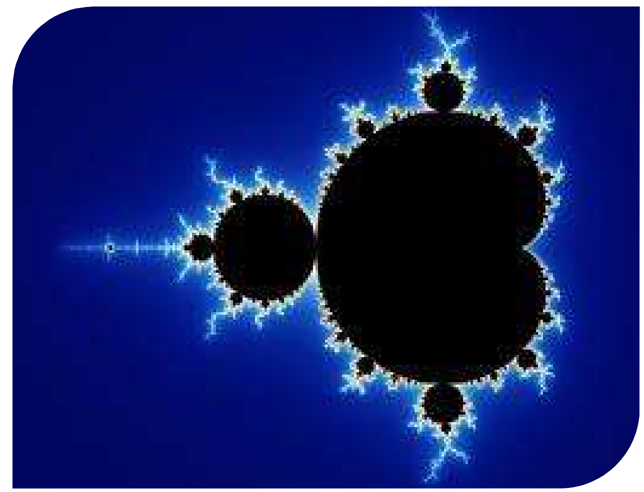
Julia sets



Mandelbrot Set

- Based on iteration theory
- Function of interest:

$$f(z) = (s)^2 + c$$



- Sequence of values:

$$d_1 = (s)^2 + c$$

$$d_2 = ((s)^2 + c)^2 + c$$

$$d_3 = (((s)^2 + c)^2 + c)^2 + c$$

$$d_4 = ((((s)^2 + c)^2 + c)^2 + c)^2 + c$$

Mandelbrot Set

- Orbit depends on s and c
- Basic question:
  - For given s and c,
    - does function stay finite? (within Mandelbrot set)
    - explode to infinity? (outside Mandelbrot set)
- Definition: if $|d| < 1$, orbit is finite else infinite
- Example orbits:
  - s = 0, c = -1, orbit = 0,-1,0,-1,0,-1,0,-1,.....finite
  - s = 0, c = 1, orbit = 0,1,2,5,26,677...... explodes

Mandelbrot Set

- Mandelbrot set: use complex numbers for c and s
- Always set s = 0
- Choose c as a complex number
- For example:
  - $s = 0, c = 0.2 + 0.5i$
- Hence, orbit:
  - $0, c, c^2 + c, (c^2 + c)^2 + c, \ldots\ldots\ldots$

- Definition: Mandelbrot set includes all finite orbit c

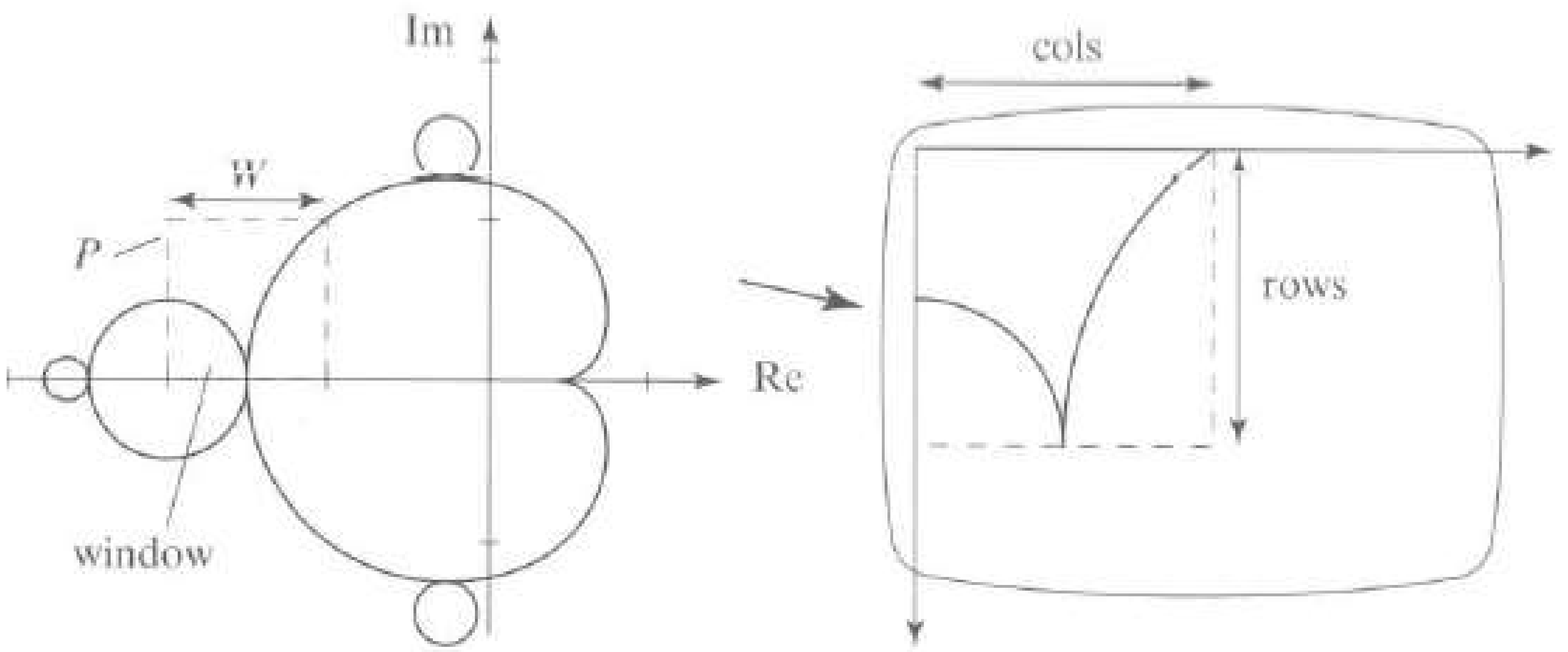
Mandelbrot Set

- Routine to draw Mandelbrot set:
- Cannot iterate forever: our program will hang!
- Instead iterate 100 times
- Math theorem:
  - if number (|d|) hasn’t exceeded 2 after 100 iterations, it never will!
- Routine returns:
  - Number of times iterated before modulus exceeds 2, or
  - 100, if modulus doesn’t exceed 2 after 100 iterations
  - See dwell( ) function in Hill (figure A4.5, pg. 755)

Mandelbrot set

- Map real part to x-axis
- Map imaginary part to y-axis
- Decide range of complex numbers to investigate. E.g:
  - X in range [-2.25: 0.75]
  - Y in range [-1.5: 1.5]
- Choose your viewport. E.g:
  - Viewport = [V.L, V.R, V.B, V.T]= [60,380,80,240]

Mandelbrot Set



References

- http://www.bugman123.com/Fractals/Mandelbrot.html