Procedural Modeling

Proceduralism in Computer Graphics

- Fixed models/primitives not robust enough
- Quest for extensibility and programmability
- Use a function or procedure to define the surface or structure of an object.
- Procedural Methods
  - Shading
  - Modeling
  - Animation

Procedural Models

- Topics
  - Fractals
    - Fractal terrains
    - L-Systems
  - Volumetric Models
    - Hypertexture
    - Particle Systems

Fractals

- A language of form for shapes and phenomena common in Nature
- *"Geometrical complex object, the complexity of which arises through the repetition of form over some range of scale".*
- Statistical self-similarity at all scales

Why Fractals?

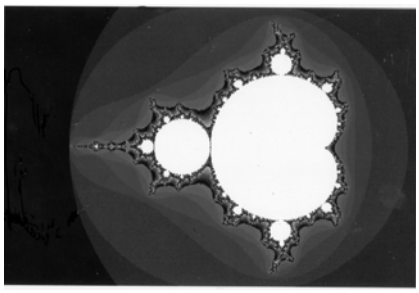
- Procedural way to add complexity to a scene
- Elements of nature posses fractal properties.
- Used to model nature
  - Terrain
  - Clouds
  - Coastlines
  - Trees / Landscaping

Fractals

Foley/VanDam/Feiner/Hughes

- Repetition of some underlying shape (basis function) at different scales
- Kock Snowflake Applet
  - http://www.arcytech.org/java/fractals/koch.shtml

Fractals – The Mandelbrot Set



Fractals - Mandelbrot Set

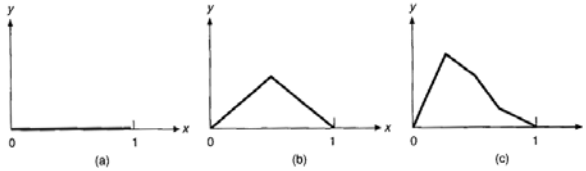
- Plot of a recursive mathematical function in the complex plane
  - $Z_n = Z_{n-1}^2 + C$
- For each complex number, the function will:
  - Move quickly to infinity (outside of the set)
  - Move slowly to infinity (on the border of the set)
  - Remain near the origin (inside the set)
- Create image by coloring based on how many iterations it takes to indicate divergence towards infinity.
- Function is self-similar as we zoom into different areas of the plot on the complex plane.

Fractals

- Fractals - Mandelbrot Set
  - Fractint

Fractal Terrain

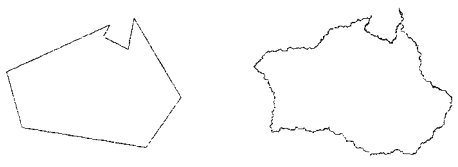
- Fractals -- a simple example



Foley/VanDam/Feiner/Hughes

Fractal Terrain

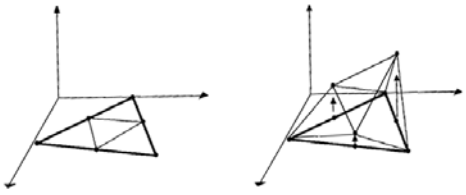
- Fractals - a simple example



[Fournier82]

Fractal Terrain

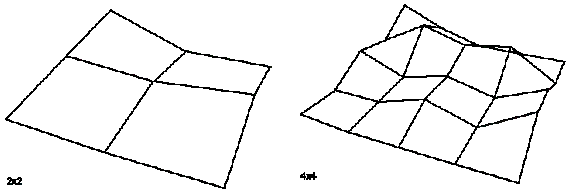
- Fractals - extend to 2d



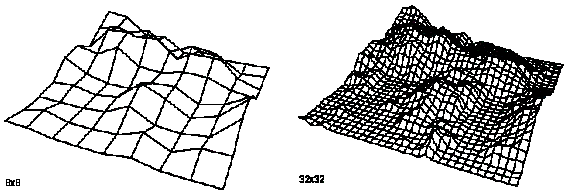
Foley/VanDam/Feiner/Hughes

Fractal Terrain

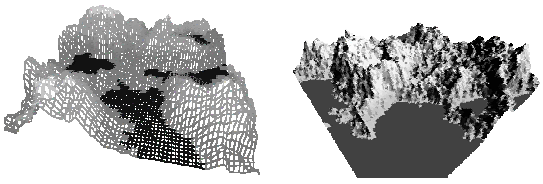
- Paul Bourke



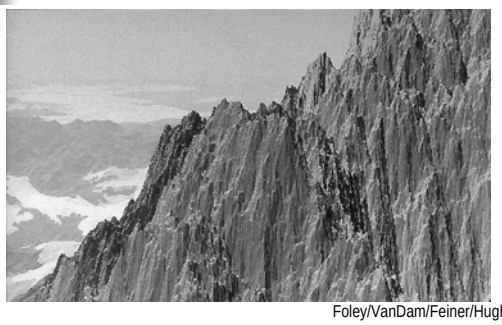
Fractal Terrain



Fractal Terrain



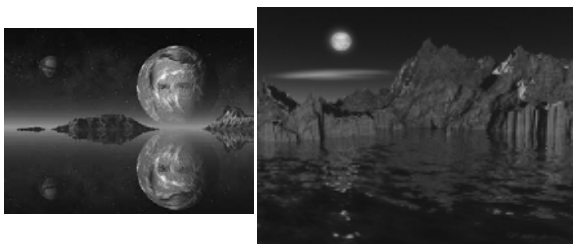
Fractal Terrain -Vol Libre Ridge



Foley/VanDam/Feiner/Hughes

Fractal Terrain

- my favorite fractal landscape



[F. Kenton Musgrave]

Fractals

- *Fractal* comes from fractal dimension

$$a.b$$

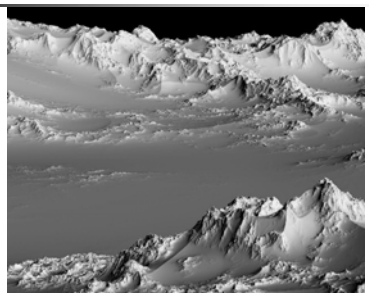
a = Euclidean dimension

b = fraction of filling up next dimension

Fractal Modeling

- Fractal modeling
  - Like Mandelbrot set, can zoom infinitely
  - Render at resolution that is most appropriate
  - Instant anti-aliasing.
  - Can model a whole planet procedurally

Fractal Modeling



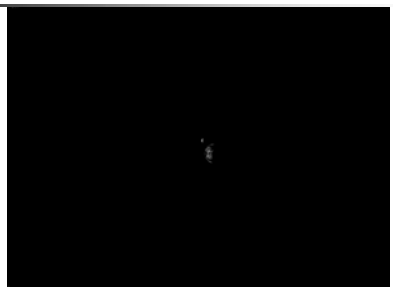
[Musgrave]

Fractals

"If one assumes that a certain error (e.g., pixel-sized) in the ray-surface intersection is acceptable, one can directly ray-trace a procedurally-defined height field with essentially perfect level of detail."

-- Ken Musgrave

Gaea Zoom



[Musgrave]

Fractal Modeling

- http://www.pandromeda.com
  - Explore fractal worlds on-line

F.K. Musgrave

  - Mojo World

Fractals

- For more info:
  - [Fournier82]
  - Pietgen, *The Science of Fractal Images*
  - Pietgen, *The Beauty of Fractals*
  - Ebert, et al, *Modeling & Texturing...*

Grammar Based Systems

- Building of models based on formal language grammars
- Method for creating fractals
- Grammar consists of:
  - Set of characters
  - Productions rules
  - Starting word

Grammar Based Systems - Example

Characters: {A, B,[,]}
Rules: A -> AA
B->A[B]AA[B]
Start word B

Iterations:

0: B

1: A[B]AA[B]

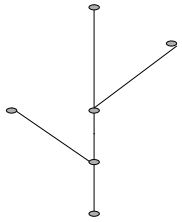
2: AA[A[B]AA[B]]AAAA[A[B]AA[B]]

Grammar Based Systems

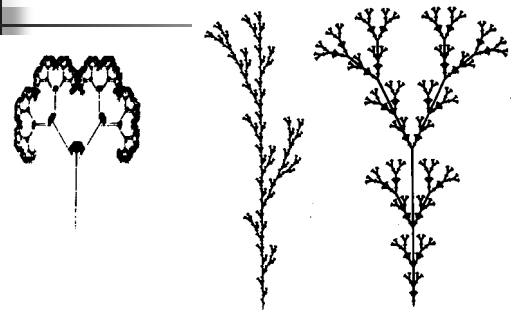
- But how does this help us create models?
- Assign a drawing action to each character:
- L-System (Lindenmeyer) used to create tree-like structures:
  - F move forward and draw
  - f move forward and do not draw
  - + increase angle with angle increment
  - - decrease angle by angle increment
  - [push state (i.e. branch)
  -] pop state (i.e finish branch)

Grammar Based Systems

- L-Systems
  - F=F[+F]F[-F]F
  - Applet
    - http://www-sfb288.math.tu-berlin.de/vgp/javaview/vgp/tutor/lsystem/PaLSystem.html



L-System Ferns



Grammar Based Systems

- L-Systems
  - Note that structures created using L-Systems are fractal like in the sense that they are self-similar at different levels
  - Self-similarity achieved by repeatedly applying production rules

Grammar Based Systems

- Koch curve as an L-system
  - F=F+F- -F+F
  - http://www.arcytech.org/java/fractals/lsystems.shtml

Foley/VanDam/Feiner/Hughes

L-Systems - Trees

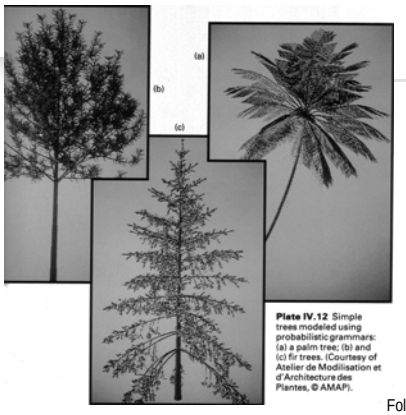


Plate IV.12 Simple trees modeled using probabilistic grammars: (a) a palm tree; (b) and (c) fir trees. (Courtesy of Atelier de Modilisation et d'Architecture des Plantes, © AMAP).

Foley/VanDam/Feiner/Hughes

L-System Trees

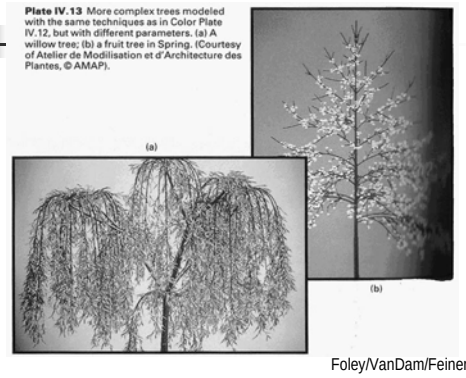
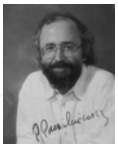


Plate IV.13 More complex trees modeled with the same techniques as in Color Plate IV.12, but with different parameters. (a) A willow tree; (b) a fruit tree in Spring. (Courtesy of Atelier de Modilisation et d'Architecture des Plantes, © AMAP).

Foley/VanDam/Feiner/Hughes

Grammar-Based Systems



- For more info:
  - Prusinkiewicz, *Lindenmayer systems, fractals, and plants*
  - Prusinkiewicz and Lindenmeyer, *The Algoritmic Beauty of Plants*
- *Questions?*

Volumetric Models

- Not all objects are "solid" models
  - water
  - fire
  - clouds
  - Rain
- Objects exists in a volume
  - Hypertexture
  - Particle Systems

Hypertexture [Perlin89]

- Extension of procedural textures
- Between surface + texture, i.e., spatial filling/volumetric
- Objects modeled as distribution of density
  - hard region - objects completely solid
  - soft region - object shape is malleable using a toolkit of shaping functions and CSG style operators to combine shapes

Hypertexture Uses

- Model shapes that don't have a well-defined boundary surface
  - Fur/hair
  - Fire/clouds/smoke
- Complex surface volumetics
  - Fluid flow
  - Erosion effects

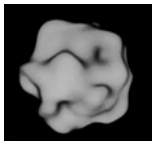
Hypertexture

- D(x) - Object Density Function over R^3
  - D (**x**) for all points **x** in 3D space [0,1]
  - Density of 3D shape
  - D (**x**) = 0 for all points outside the surface
  - D (**x**) = 1 for hard region of the object
  - 0 < D (**x**) < 1 for soft region of the object (fuzzy region)

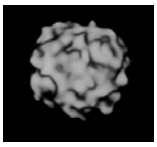
Hypertexture

- Toolbox of base DMFs
  - Bias – up / down control
  - Gain – controls gradiant
  - Noise (controlled randomness)
    - Won Ken an Academy Award!
  - Turbulence
    - Sum of noise at variety of frequencies
  - Mathematical functions

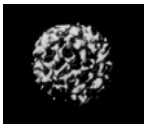
Hypertexture Noise Examples



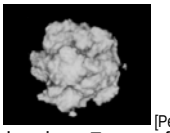
Noisy



2* frequency, 1/2 amplitude



High Amplitude, Noisy Sphere

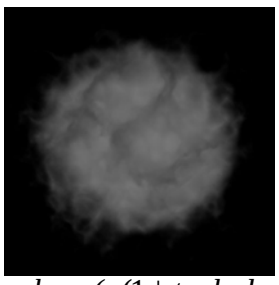


Fractal, noise - Σ many f's

[Perlin89]

Hypertexture Example - Fire

Red = low density
Yellow = high

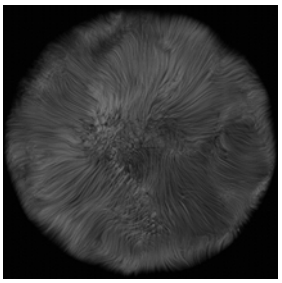


[Perlin89]

$$D(x) = sphere(x(1 + turbulence(x)))$$

Hypertexture Example - Fur/Hair

Here noise displaces *x* before projecting; uses variable to control curliness



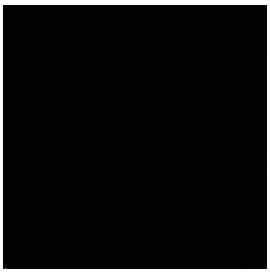
[Perlin89]

Tribble

Animated Hypertexture – *Pyroclastic*

[Musgrave]

Animated Hypertexture – *Fireball1*



[Musgrave]

Animated Hypertexture – *Cloud Dog*

[Musgrave]

Hypertexture

- Further Reading
  - [Perlin89]
  - Ebert, et. al, *Texturing & Modeling: A Procedural Approach.*

Particle Systems [Reeves83]

- Another volumetric modeling technique
- Abstraction provides control of animation and specification of objects
- Good for modeling volumetric natural phenomena:
  - water
  - fire
  - clouds
  - Rain
  - Snow
  - Grass
  - Trees

What are Particle Systems?

- A collection of geometric particles
- Algorithms governing creation, movement and death
- Attributes
- AND, randomness … can be applied to any of the above!

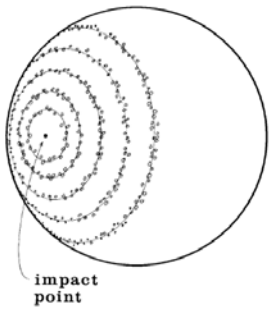
Attributes of Particles

- Initial Position
- Movement (velocity, rotation, acceleration, etc.)
- Color and Transparency
- Shape
- Volume
- Density
- Mass
- Lifetime (for particles)

Particle Systems

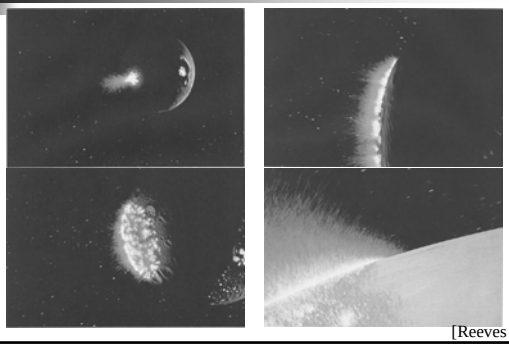
- The first example of particle systems was in the movie *Star Trek II: The Wrath of Khan*
- Particle systems were used to represent a wall of fire.

Particle Systems - *The Wrath Of Kahn*



[Reeves 83]

Particle Systems - *The Wrath Of Kahn*

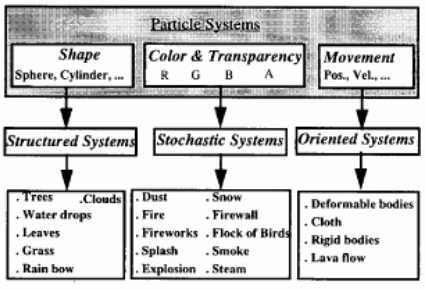


[Reeves 83]

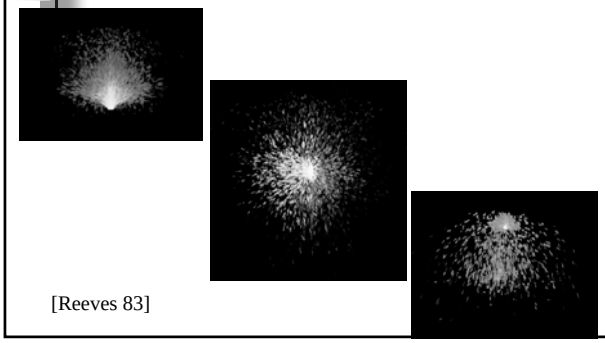
Why use Particle Systems?

- Excellent way to model complex natural objects.
- Allow us to model detailed man-made objects.
- Provides a solution to fuzzy object modeling problem.

Application of Particle Systems



Particle Systems - Fireworks



[Reeves 83]

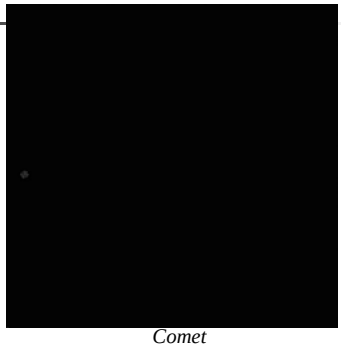
Particle Systems - Plants: *White Sand*

[Reeves 83]

Andreason & Zucca,

CG2 -19962

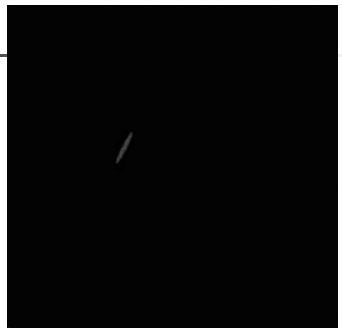
Fire and Smoke



Andreason & Zucca,

CG2 -19962

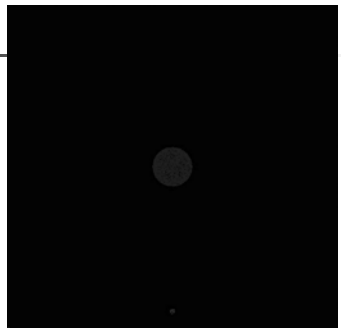
Comet



Andreason & Zucca,

CG2 -19962

Water



Andreason & Zucca,

CG2 -19962

Death Star

Procedural Models

- Summary
  - Use of a function to define objects
  - Types
    - Fractals/L-Systems
    - Volumetric Models
      - Hypertexture
      - Particle Systems
  - Good for
    - Natural objects
    - Landscapes
    - Non-solid type objects