CSCI-6962 Advanced Computer Graphics

Luxo Jr.

- Pixar Animation Studios, 1986
- Director: John Lasseter

Plan

- Introduction
- Overview of the Semester
- Administrivia
- Iterated Function Systems (Fractals)

Introductions

- Barb Cutler

  cutler@cs.rpi.edu
  http://www.cs.rpi.edu/~cutler/
  http://www.cs.rpi.edu/~cutler/classes/advancedgraphics/F05/info.html

- Who are you?

  Name, year/degree, graphics background (if any), research interests, & something fun, interesting or unusual about yourself

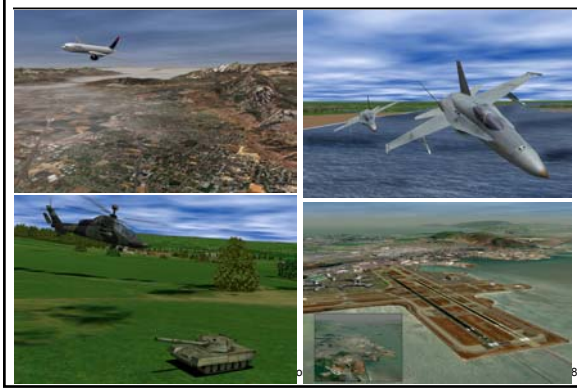
Why Computer Graphics?

- Movies
- Games
- Simulation
- CAD-CAM
- Architecture
- Virtual Reality
- Visualization
- Medical Imaging

Movies

Games

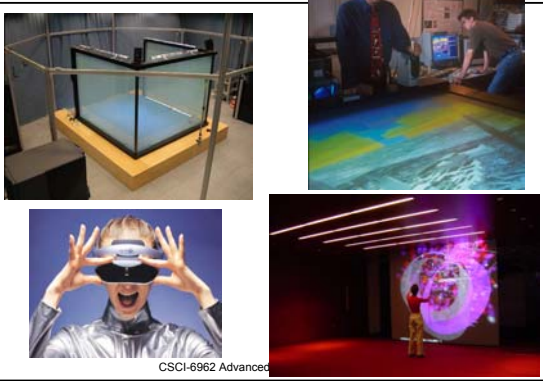
Simulation



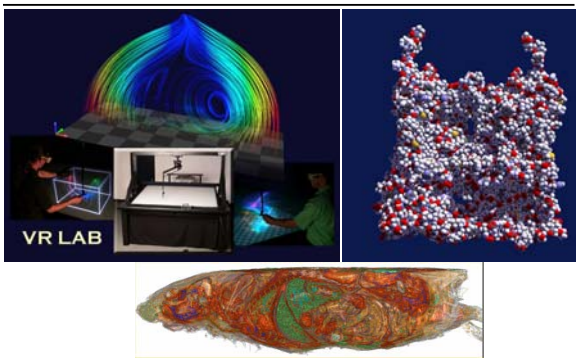
CAD-CAM & Design

Architecture

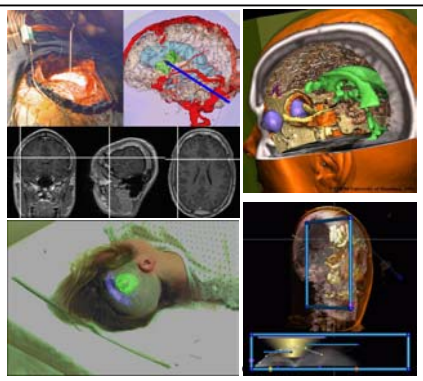
Virtual Reality



Visualization



Medical Imaging



What we will learn in CSCI-6962

- Advanced topics computer graphics algorithms
- How to tackle the challenges in many of the applications just shown

What we will NOT cover

- Software packages
  - CAD-CAM
  - Photoshop and other painting tools
- Artistic skills
- Game design
- Graphics API
  - Although you will be exposed to OpenGL

Questions?

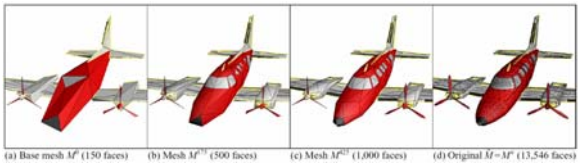
Plan

- Introduction
- Overview of the Semester
- Administrivia
- Iterated Function Systems (Fractals)

Overview of the Semester

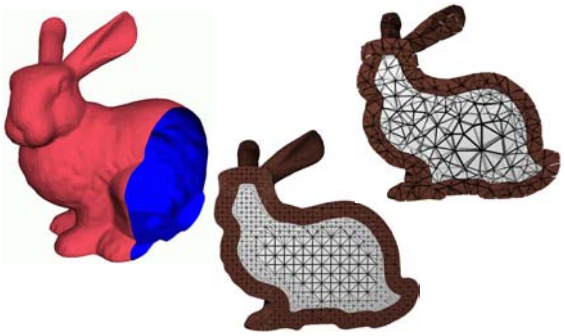
- advanced ray tracing: global illumination, photon mapping, subsurface scattering
- mesh generation and simplification
- subdivision surfaces
- appearance models
- volumetric modeling
- procedural modeling
- weathering
- simulation: particle systems, FEM, cloth
- texture synthesis
- & more …

Mesh Simplification

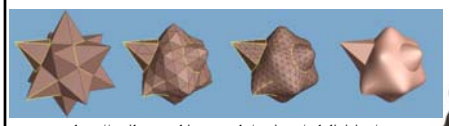


Hoppe "Progressive Meshes" SIGGRAPH 1996

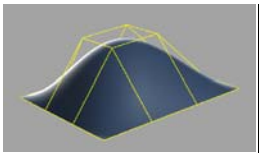
Mesh Generation & Volumetric Modeling



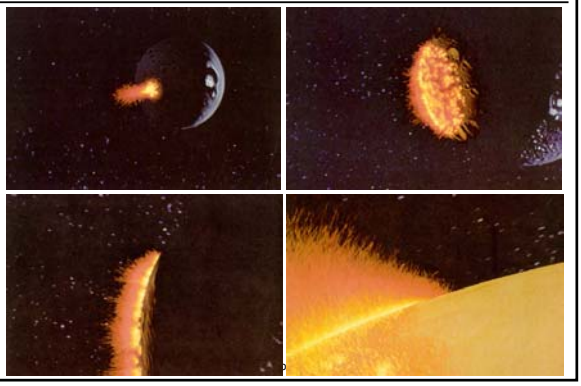
Modeling – Subdivision Surfaces



http://grail.cs.washington.edu/projects/subdivision/

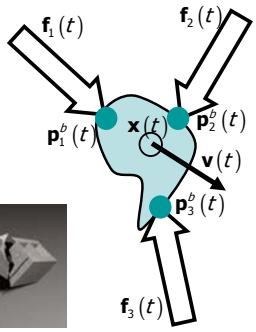


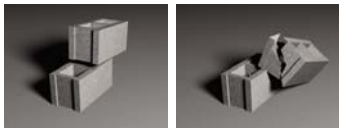
Particle system (PDE)



Physical Simulation

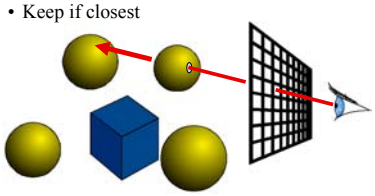
- Rigid Body Dynamics
- Collision Detection
- Fracture
- Deformation





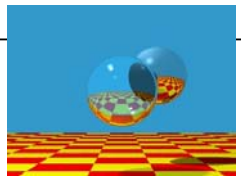
Ray Casting

- For every pixel
  construct a ray from the eye
  - For every object in the scene
    - Find intersection with the ray
    - Keep if closest

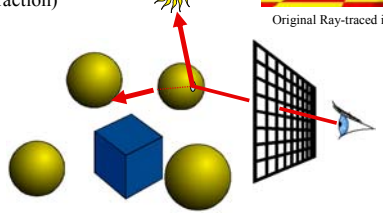


Ray Tracing

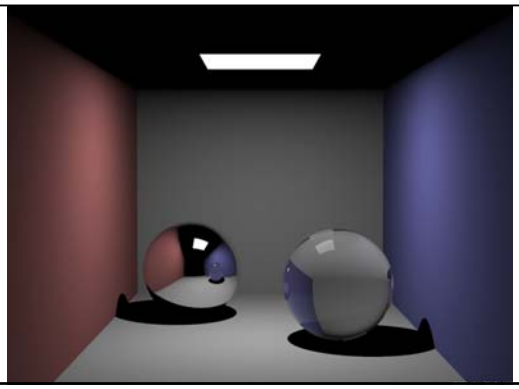
- Shade (interaction of light and material)
- Secondary rays (shadows, reflection, refraction)



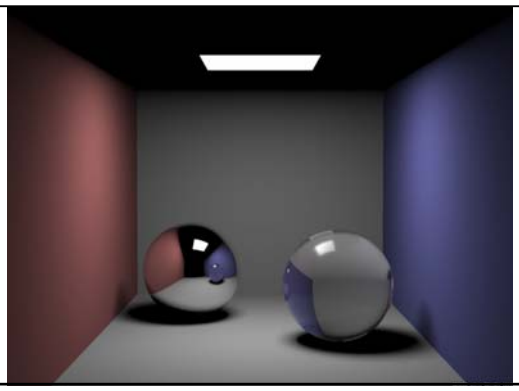
Original Ray-traced image by Whitted



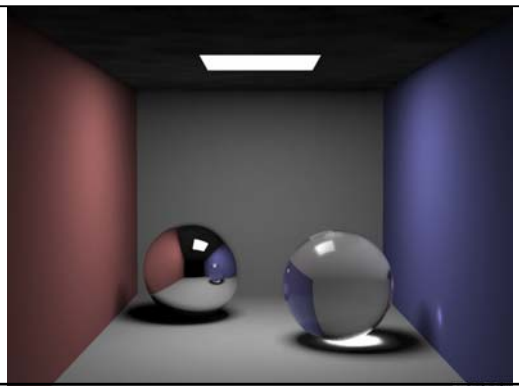
Traditional Ray Tracing



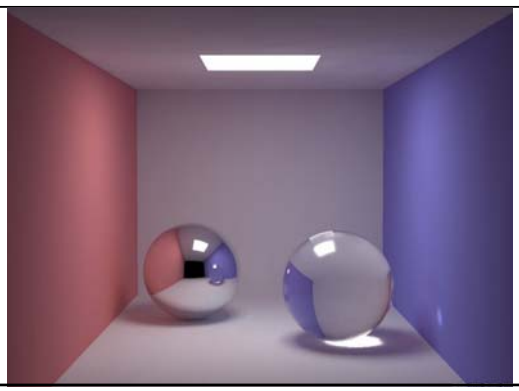
Ray Tracing + Soft Shadows



Ray Tracing + Caustics

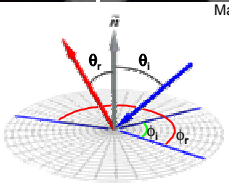


Global Illumination



Appearance Models

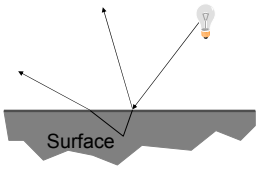
Matusik



Henrik Wann Jensen

Subsurface Scattering

Jensen et.al 01



Questions?

Plan

- Introduction
- Overview of the Semester
- Administrivia
- Iterated Function Systems (Fractals)

Prerequisites

- Not enforced
- Linear Algebra
  - vectors, matrices, basis, solving systems of equations
- Algorithms
  - Orders of growth, bounds, sorting, trees
- All assignments are in C++
- Previous coursework or experience in Computer Graphics an asset

Grading Policy

- Assignments: 40%
  - 4 programming assignments
  - Must be completed individually
- Final Project: 30%
- Quizzes: 15%
  - Tuesday, Oct 18th & Tuesday, Nov 17th (in class)
- Participation & Presentation: 15%
  - Lead discussion of one paper during the semester
  - Present your final project on Friday Dec 9th (in class)

Assignments

- Turn in code and executable (Linux or Windows…)
- Coding style important
  - Be concise & efficient, and *comment your code*
- Collaboration policy:
  - You can chat, but code on your own
  - Acknowledge your collaborators!
- Late policy:
  - Due Thursdays @ 11:59pm
  - Penalized 25% per day late
  - Extensions considered only if requested >1 week before due date

Final Project

- ~ 1 month effort
- Significant extension of previous assignment *OR* Exploration of other topic discussed in class
- Can be a component of a research project outside of class
- Suggestions throughout the semester
- I'll review your proposal to make sure the scope is appropriate

Office Hours

- Tuesdays & Fridays @ 1:30 (after lecture)
- Send email to make an appointment for some other time

Questions

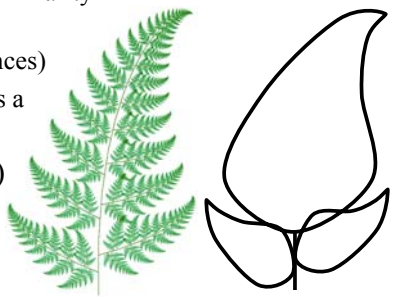
Plan

- Introduction
- Overview of the Semester
- Administrivia
- Iterated Function Systems (Fractals)

Iterated Function Systems (IFS)

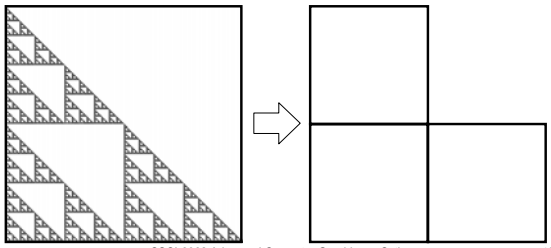
- Capture self-similarity
- Contraction (reduce distances)
- An attractor is a fixed point

$$A = \bigcup_i f_i(A)$$



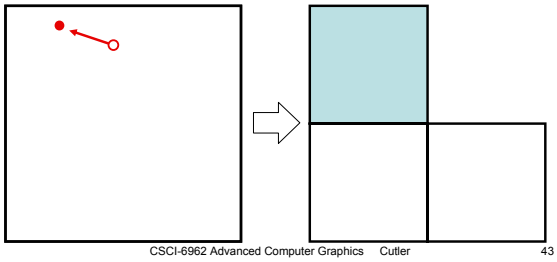
Example: Sierpinski Triangle

- Described by a set of n affine transformations
- In this case, $n = 3$
  - translate & scale by 0.5



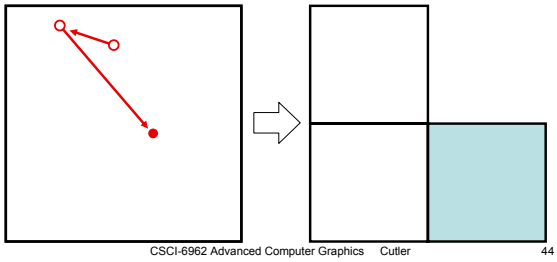
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



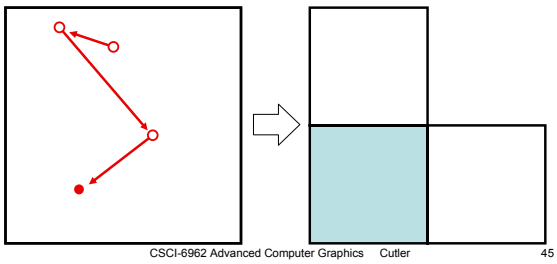
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



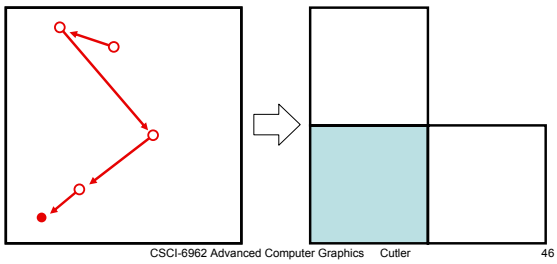
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



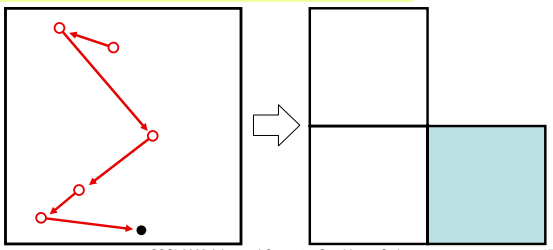
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



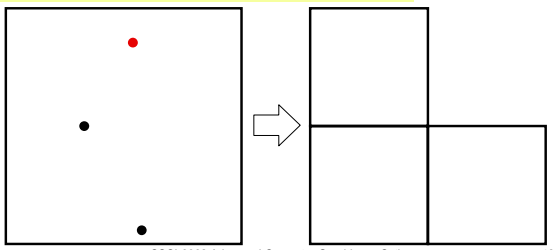
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



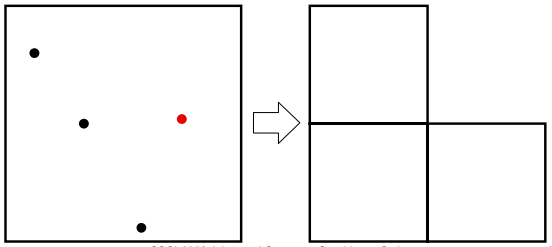
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



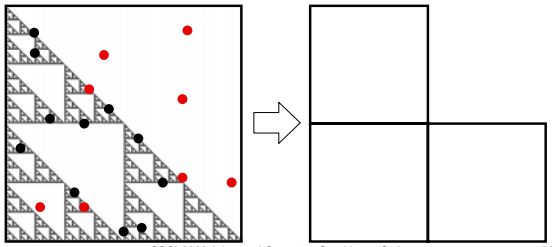
Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```



Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```

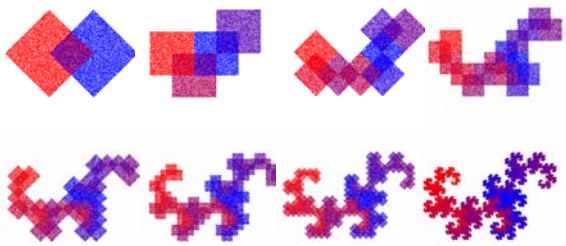


Example: Sierpinski Triangle

```
for "lots" of random input points (x0, y0)
    for j=0 to num_iters
        randomly pick transformation i
            (xk+1, yk+1) = fi (xk, yk)
    display (xk, yk)
```

Increasing the number of iterations

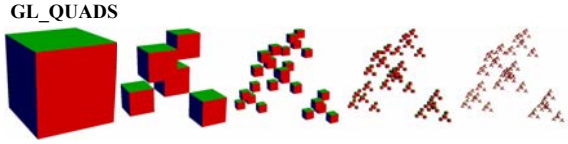
Another IFS: The Dragon



3D IFS in OpenGL

GL_POINTS

GL_QUADS

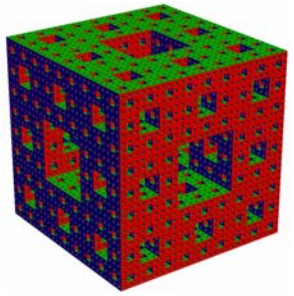


Application: Fractal Compression

- Exploit the self-similarity in an image

Compressed using Fractal Photo Lab

Assignment 1: OpenGL Warmup



- Get familiar with:
  - C++ environment
  - OpenGL
  - Transformations
  - simple Vector, Matrix & Image classes
- Have Fun!
- Due Thursday Sept 8th at 11:59pm

Questions?