6.837 Introduction to Computer Graphics

Luxo Jr.
- Pixar Animation Studios, 1986
- Director: John Lasseter

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

Team
- Lecturers
  - Frédo Durand
  - Barb Cutler
- TA
  - Rob Jagnow
- Course secretary
  - Bryt Bradley
  - 6.837-staff@graphics.csail.mit.edu
  - 6.837-ta@graphics.csail.mit.edu

Why Computer Graphics?
- Movies
- Games
- Simulation
- CAD-CAM
- Architecture
- Virtual Reality
- Visualization
- Medical Imaging

Movies
Medical Imaging

What you will learn in 6.837

- Fundamentals of computer graphics algorithms
- How to implement most of the applications just shown
- How graphics APIs and the graphics hardware work

What you will NOT learn in 6.837

- 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

- Ray Casting / Ray Tracing
- The Graphics Pipeline
- Textures, Shadows
- Sampling, Global Illumination
- Modeling, Animation, Color
- Advanced Topics
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)

Ray Tracing
- Original Ray-traced image by Whitted
  • Image computed using the Dali ray tracer by Henrik Wann Jensen
  • Environment map by Paul Debevec

The Graphics Pipeline
- Transformations
- Clipping
- Rasterization
- Visibility

Textures and Shading
- Model
- Model + Shading
- Model + Shading + Textures

At what point do things start looking real?

For more info on the computer artwork of Jeremy Tanner
see http://www.jeremytanner.com/arts/index.html
Shadows

Sampling & Antialiasing

Traditional Ray Tracing

Ray Tracing + Soft Shadows

Ray Tracing + Caustics

Global Illumination
Modeling

- Curved surfaces
- Subdivision surfaces

Animation: Keyframing

Particle system (PDE)

Rigid Body Dynamics

- Simulate all external forces and torques

Color

Image-Based Rendering

- Use images as inputs and representation
- E.g. Image-based modeling and photo editing
  Oh, Chen, Dorsey and Durand 2001
Questions?

Plan

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

Administrivia: Prerequisites

- Not enforced
- All assignments are in C++
  - Optional review/introductory session
    Monday Sept 13, 7:30-9pm, 4-231
- Linear Algebra (18.06)
  - vectors, matrices, basis, solving systems of equations
    Optional review/introductory session
    Monday Sept 20, 7:30-9pm, 2-105
- Algorithms (6.046)
  - Orders of growth, bounds, sorting, trees

Administrivia: Grading Policy

- Assignments: 75%
  - Weekly programming assignments
  - Must be completed individually
  - No final project this year
- Quiz: 10%
  - Tuesday, Oct 26 (in class)
- Final Exam: 10%
  - TBA during finals week
- Participation: 5%

Administrivia: Assignments

- Turn in code and executable (Linux or Windows)
- Coding style important
  - Many assignments are cumulative
- Collaboration policy:
  - You can chat, but code on your own
  - Acknowledge your collaboration!
- Late policy:
  - Due Wednesday @ 11:59pm
  - Penalized 25% per day late
  - Extensions only considered if requested 1 week before due date

"Create Your Own Assignment"

- Last assignment, ~ 1 week effort
- Extension of previous assignment OR Exploration of other topic discussed in class
- Suggestions throughout the semester
- We'll review your proposal to make sure the scope is appropriate
Administrivia: Lab & Office Hours

- http://graphics.csail.mit.edu/classes/6.837/F04/
- Fredo
  - Tuesday 6 PM in W20 55
- Barb
  - Wednesdays 6 PM in W20 55
- Rob
  - Wednesday 8 PM in W20 55
- 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
  \[ A = \bigcup f_i(A) \]

Example: Sierpinski Triangle

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

Example: Sierpinski Triangle

```plaintext
for "lots" of random input points (x_0, y_0)
for j=0 to num_iters
    randomly pick transformation i
    \( \{x_{nk}, y_{nk}\} = f_i(x_k, y_k) \)
display (x_k, y_k)
```
Example: Sierpinski Triangle

for "lots" of random input points \((x_0, y_0)\)

for \(j=0\) to \(\text{num\_iters}\)

randomly pick transformation \(i\)

\((x_{k+1}, y_{k+1}) = f_i(x_k, y_k)\)

display \((x_k, y_k)\)
Example: Sierpinski Triangle
for "lots" of random input points \((x_0, y_0)\)
for \(j=0\) to \(\text{num\_iters}\)
  randomly pick transformation \(i\)
  \((x_{k+1}, y_{k+1}) = f_i(x_k, y_k)\)
  display \((x_k, y_k)\)

Another IFS: The Dragon

Application: Fractal Compression
• Exploit the self-similarity in an image

Assignment 0: IFS
• Get familiar with:
  – C++ environment
  – Vector, Matrix & Image classes
• Due Wednesday Sept 15 at 11:59pm
• Optional C++ review/introductory session
  Monday Sept 13, 7:30-9pm, 4-231
• http://graphics.lcs.mit.edu/classes/6.837/F03/

Questions?