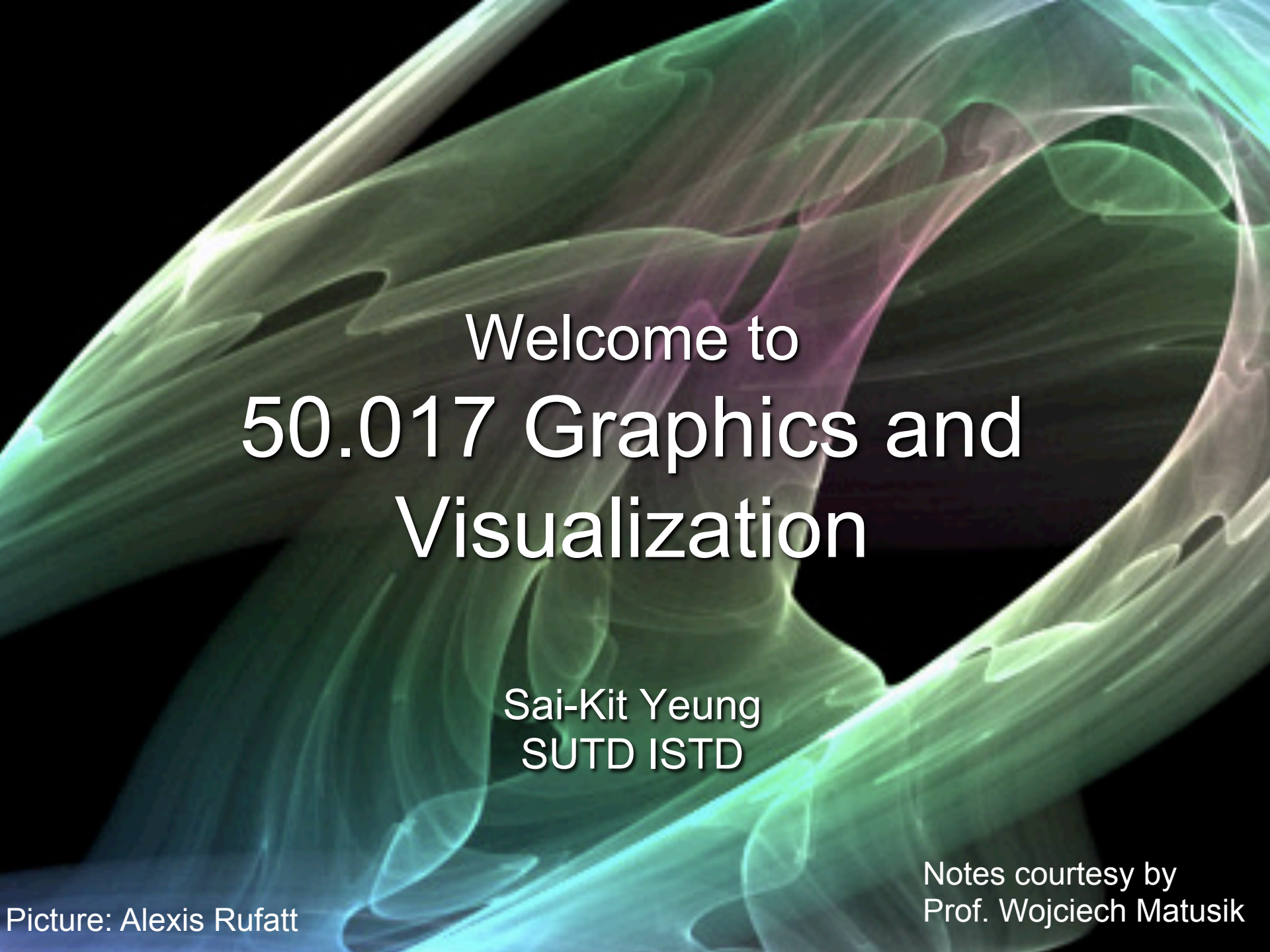
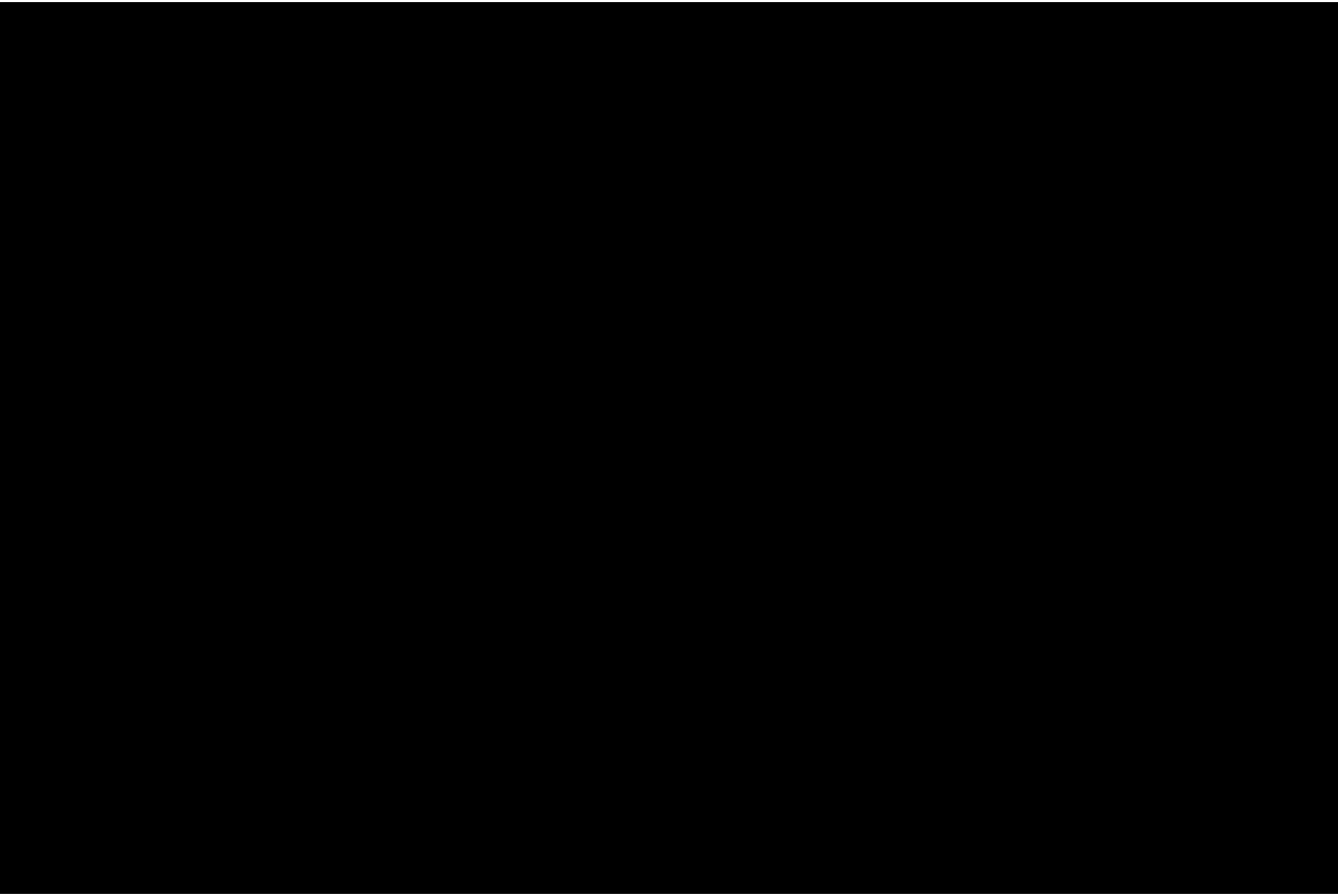


Welcome to 50.017 Graphics and Visualization

Sai-Kit Yeung
SUTD ISTD

Notes courtesy by
Prof. Wojciech Matusik

Picture: Alexis Rufatt



Luxo Jr.

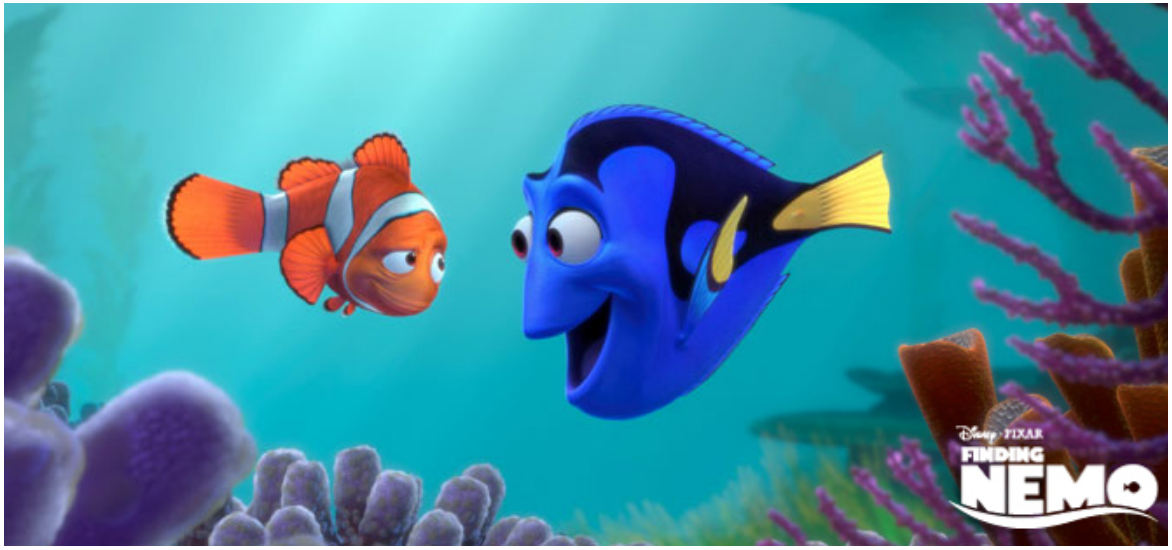
- Pixar Animation Studios, 1986
- Director: John Lasseter
- Shown in SIGGRAPH 1986

Plan

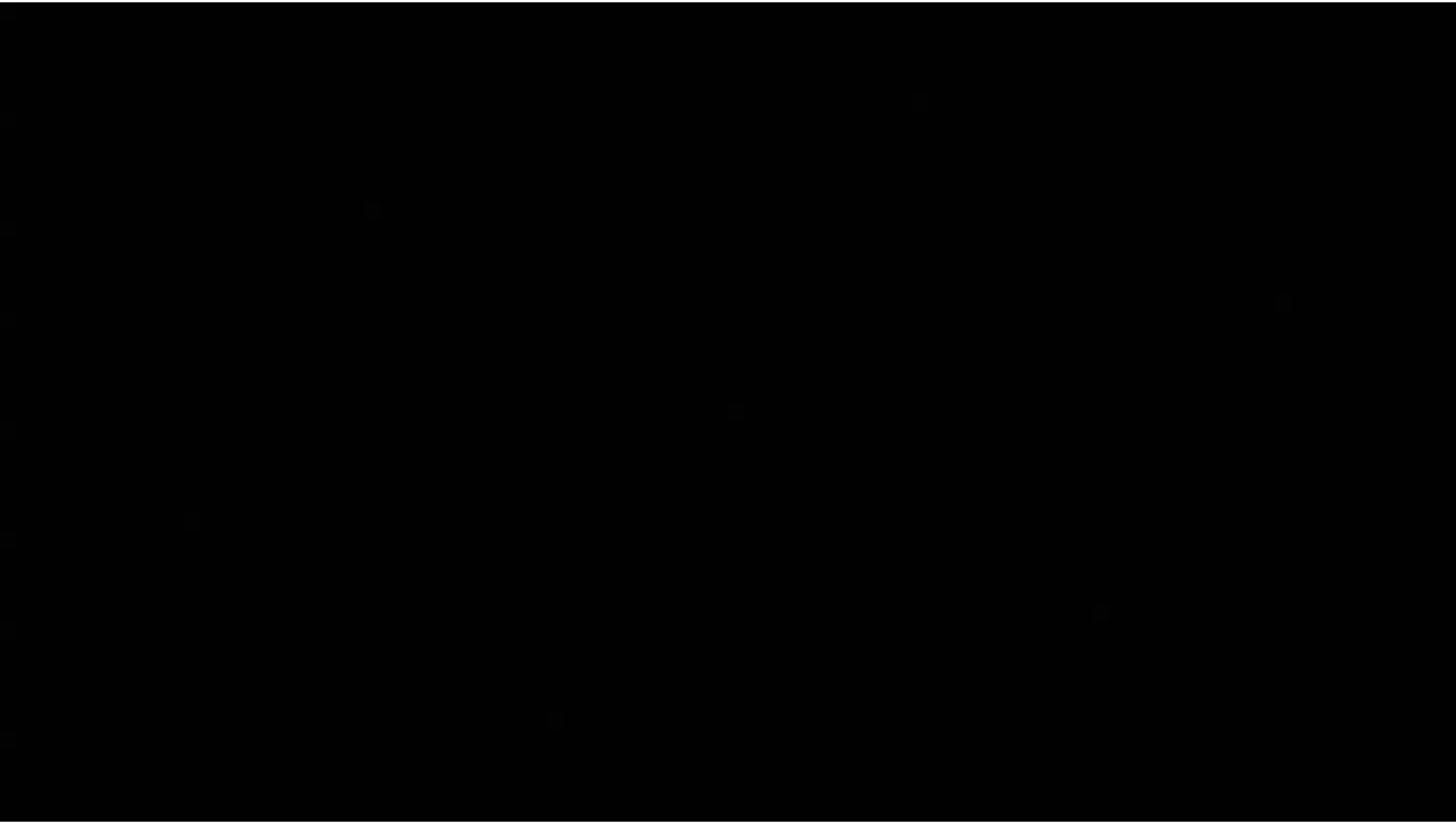
- Overview of computer graphics
- Administrivia
- Overview of the semester
- Overview of assignments
- Intro to OpenGL & assignment 0

What are the applications of graphics?

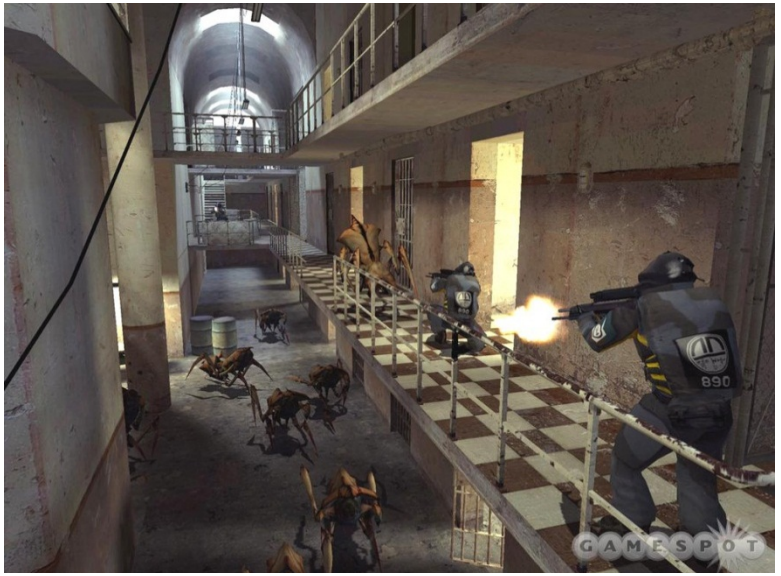
Movies/special effects



More than you would expect

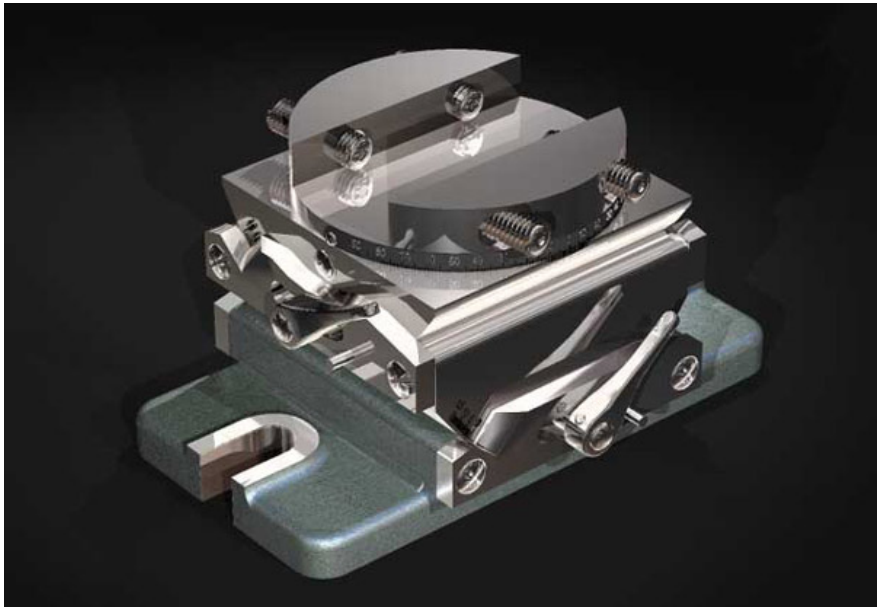


Video Games

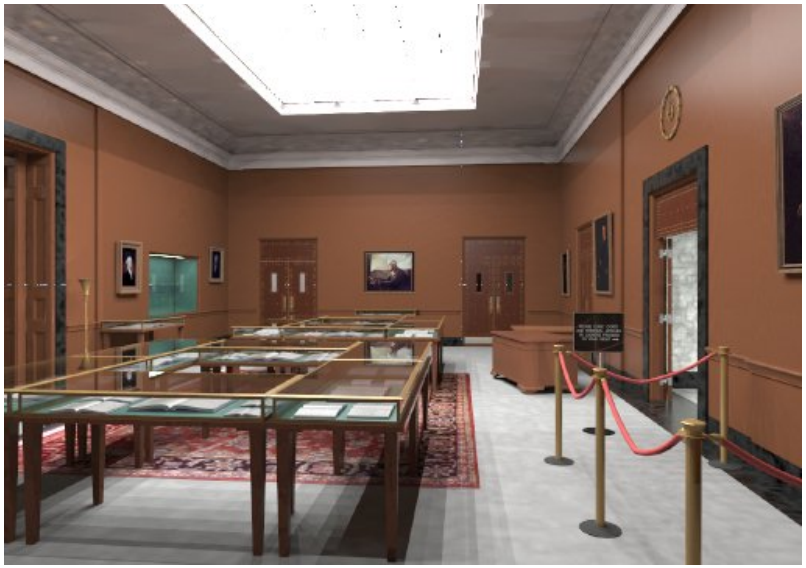


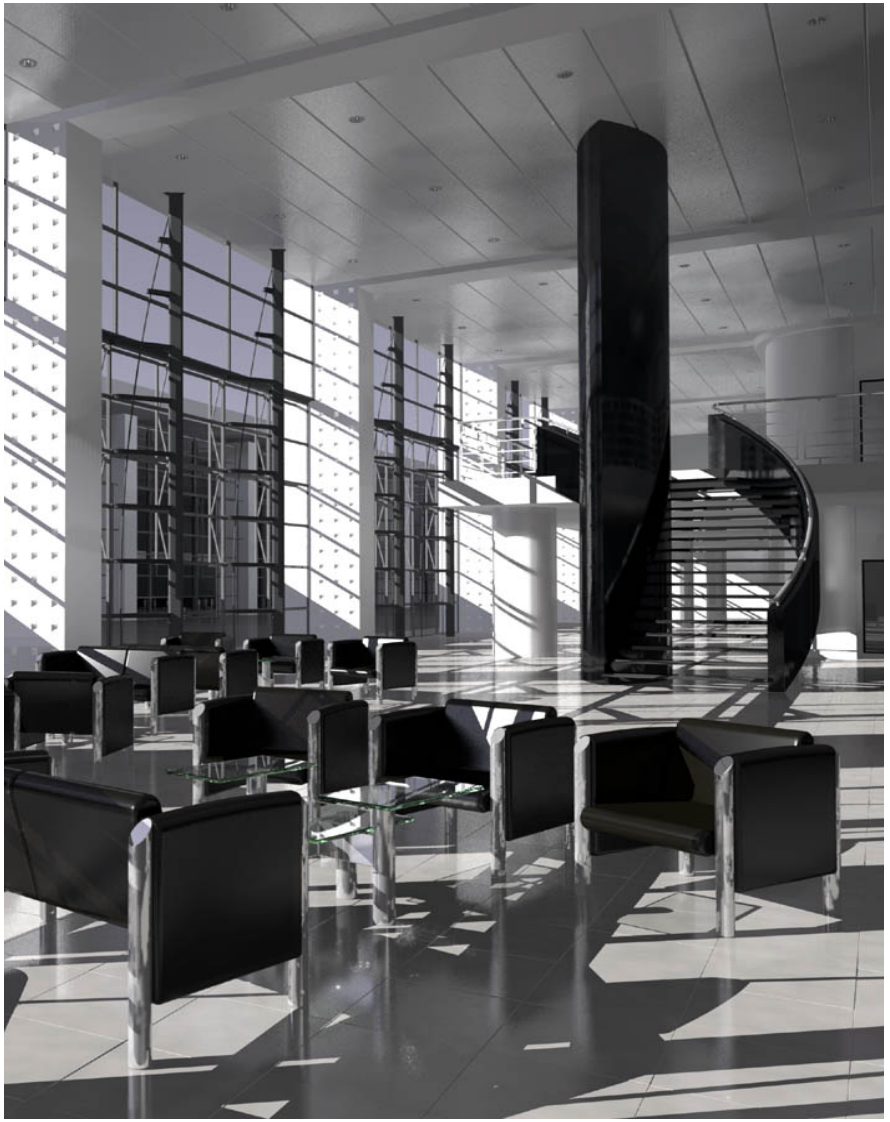
Simulation

CAD-CAM & Design

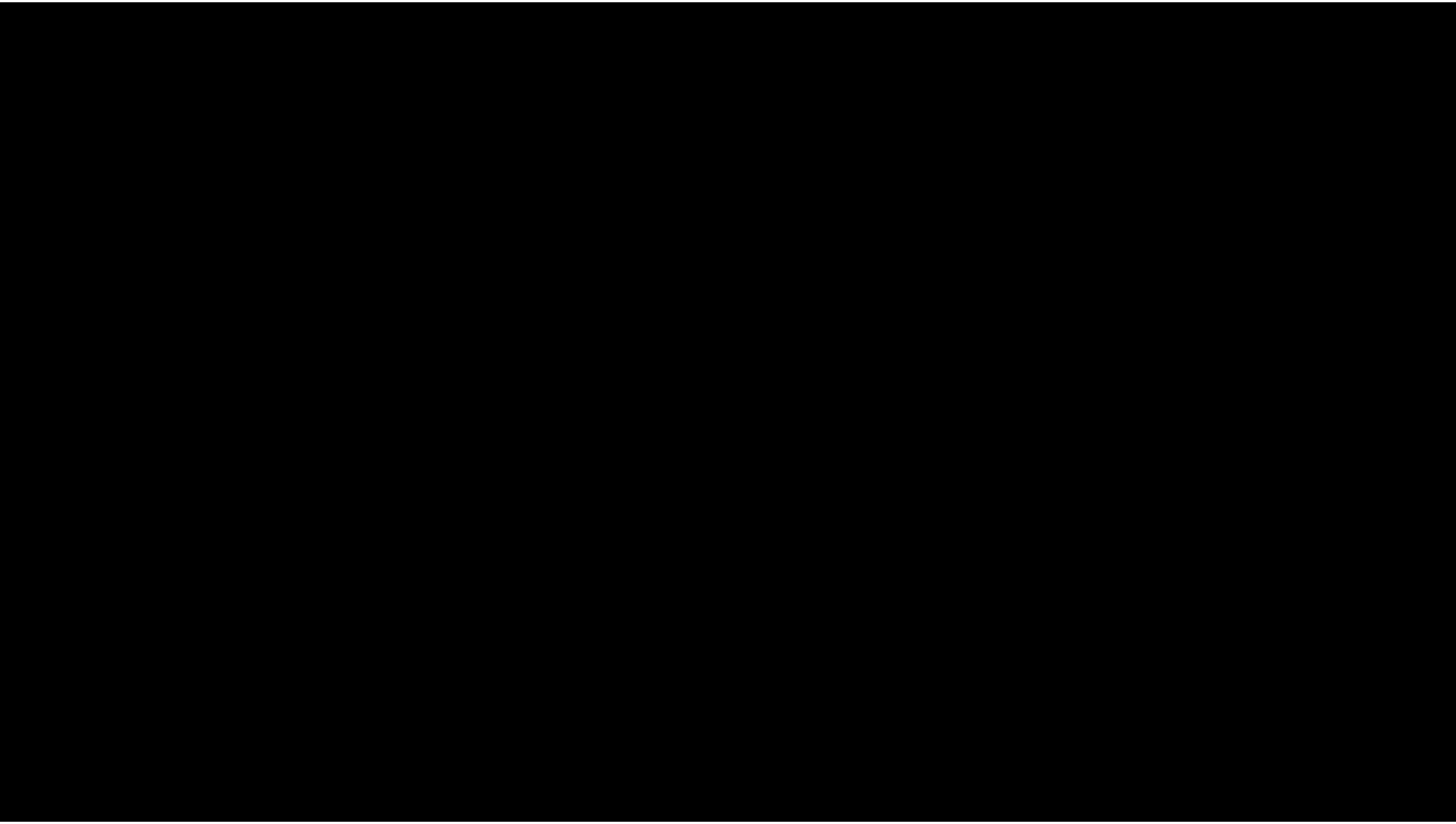


Architecture

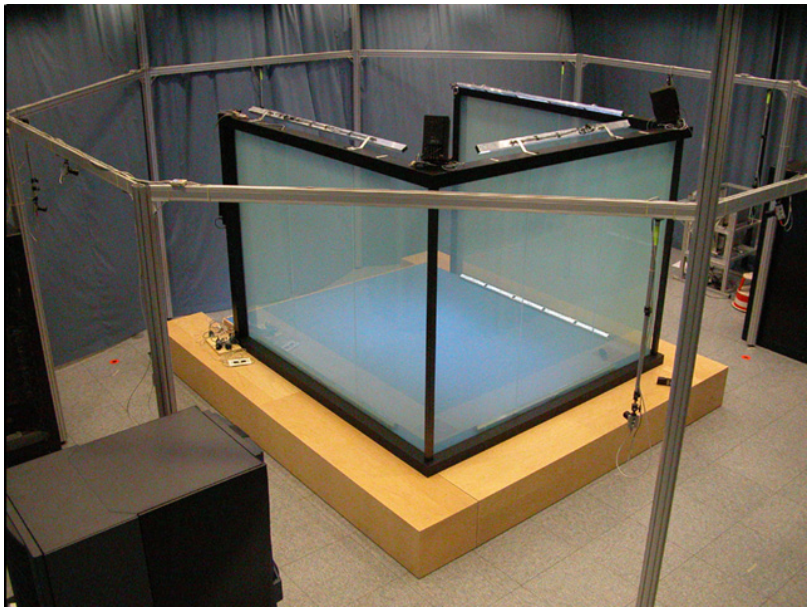


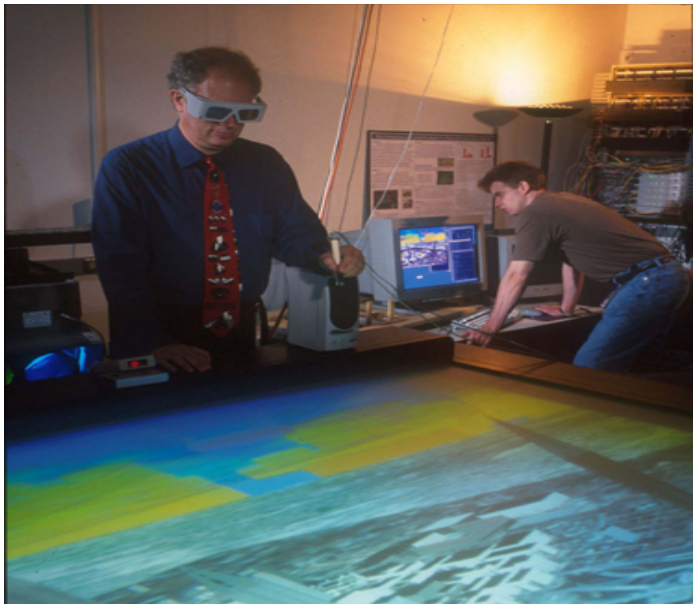


Game Engines

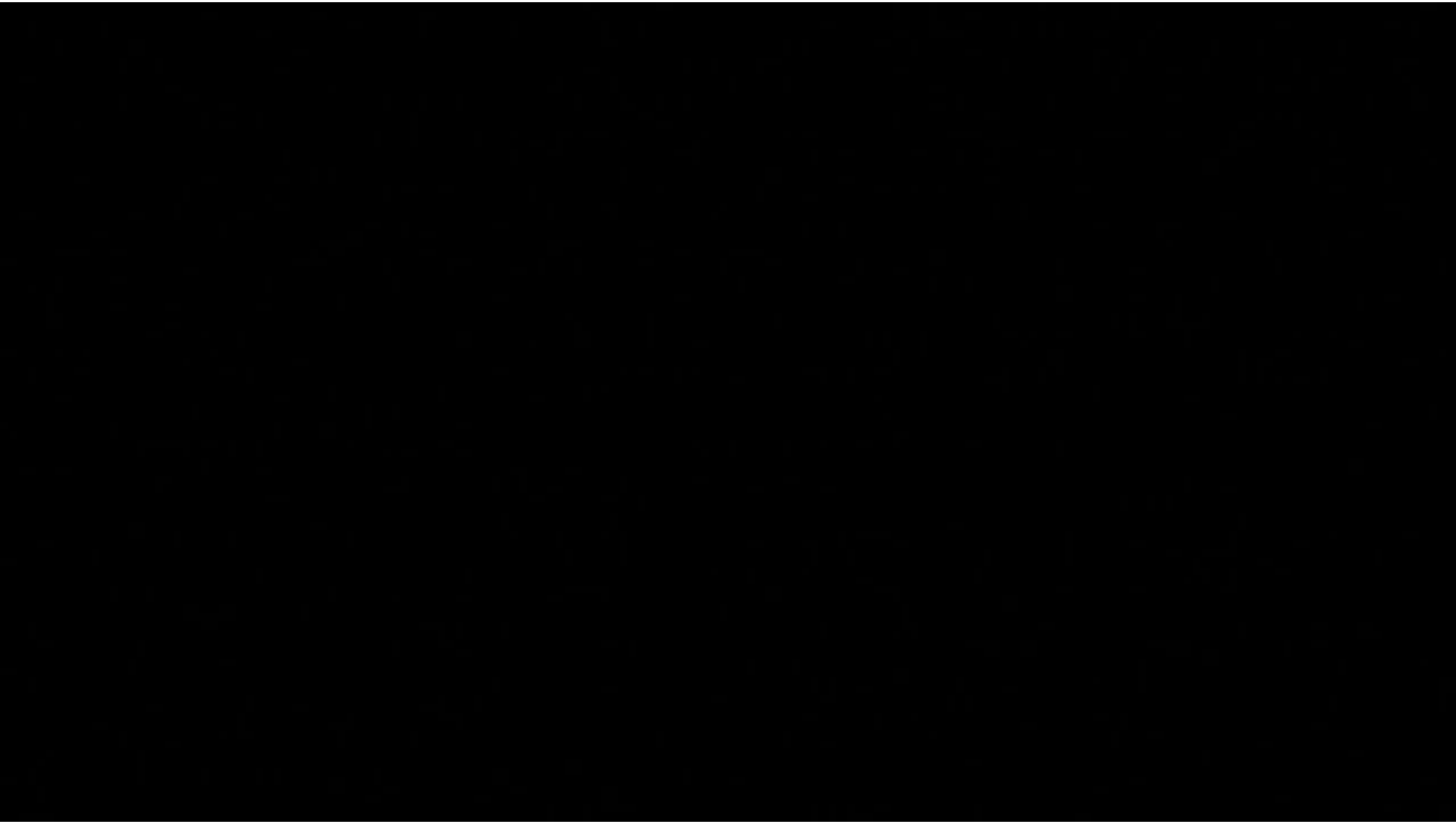


Virtual Reality

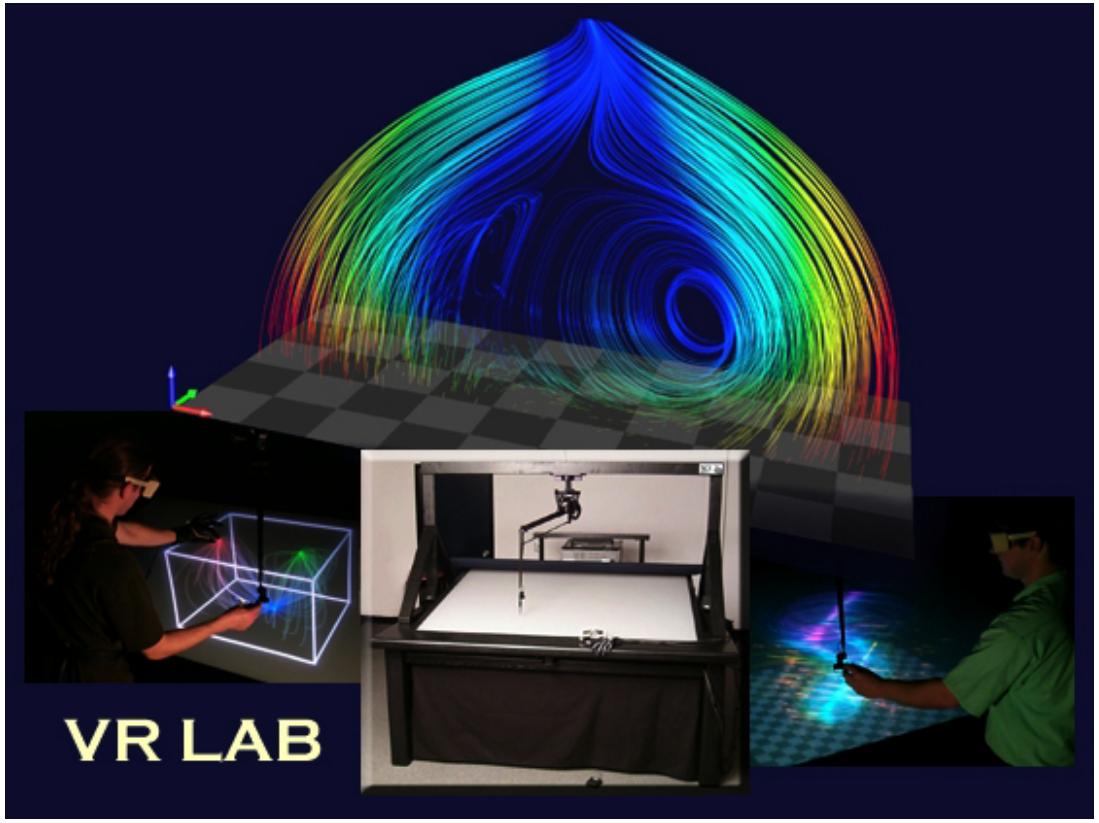


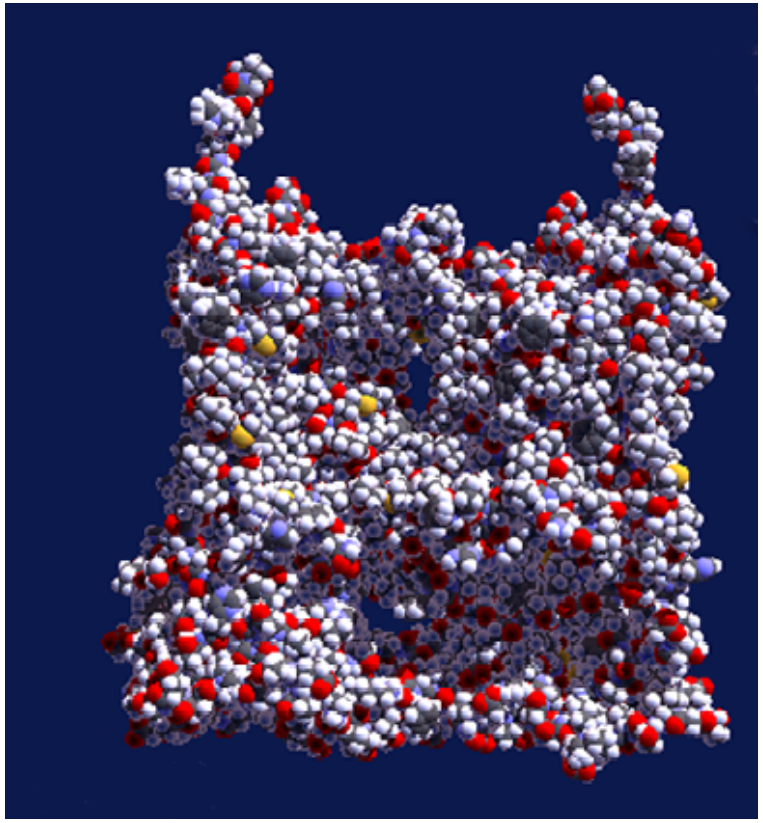


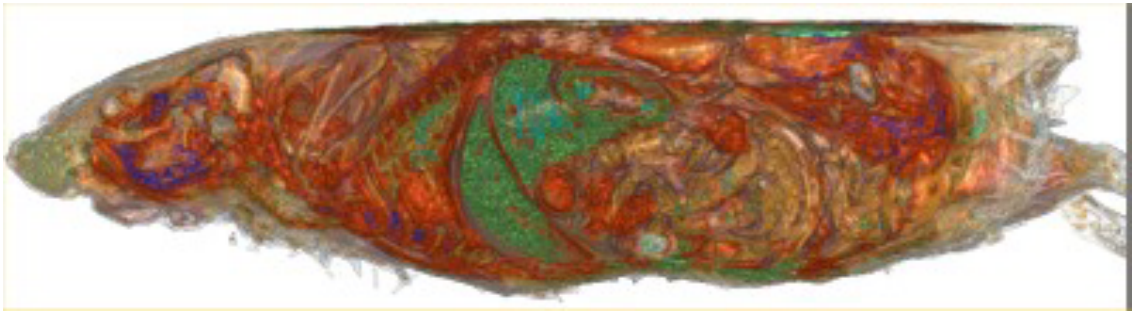
Virtual Reality



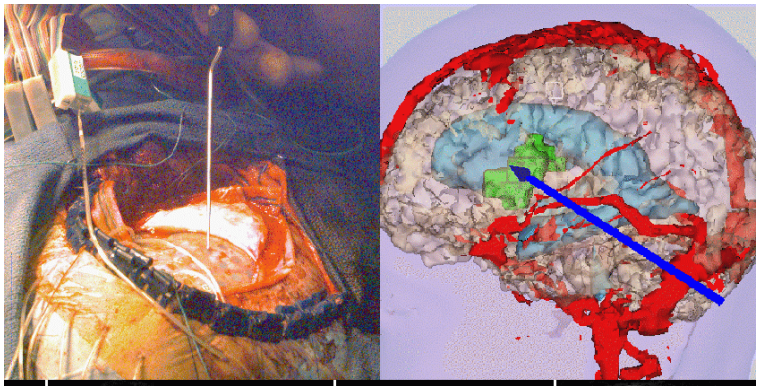
Visualization

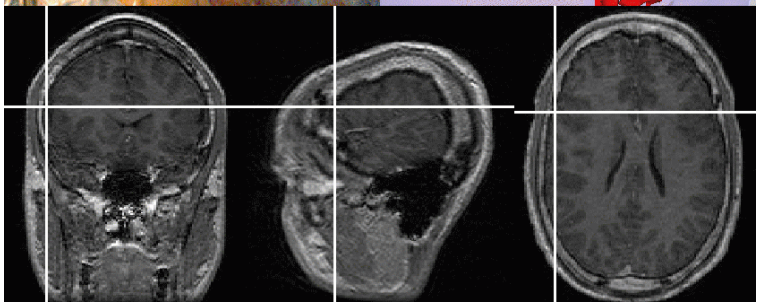


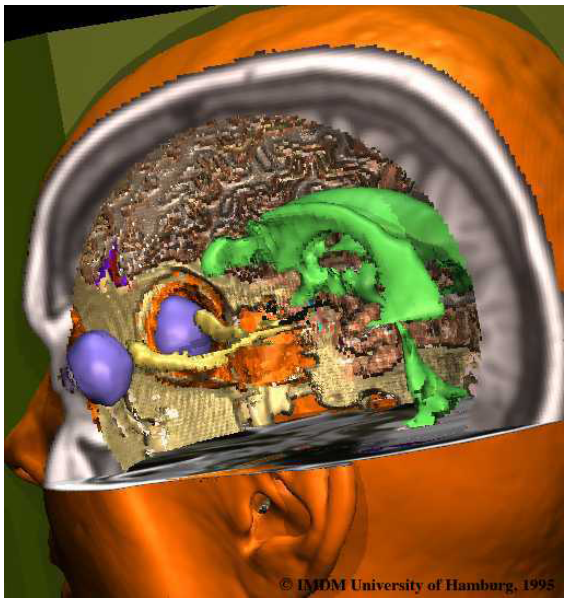


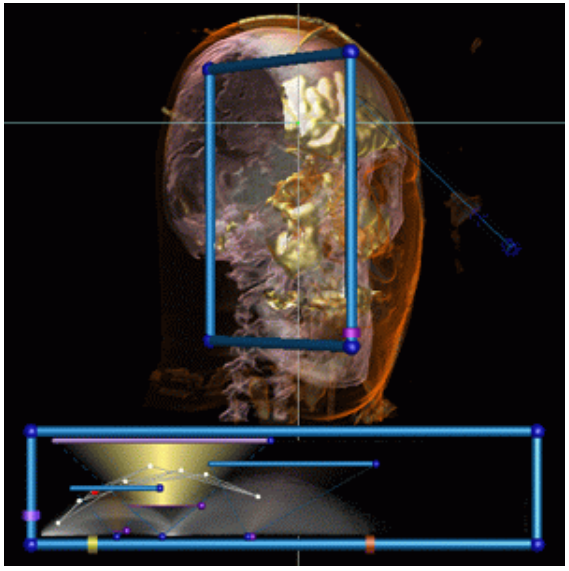


Medical Imaging

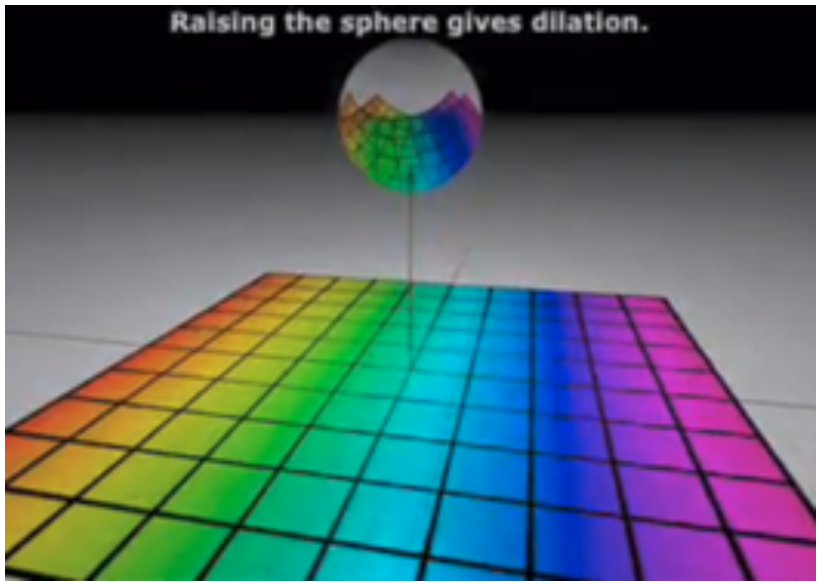


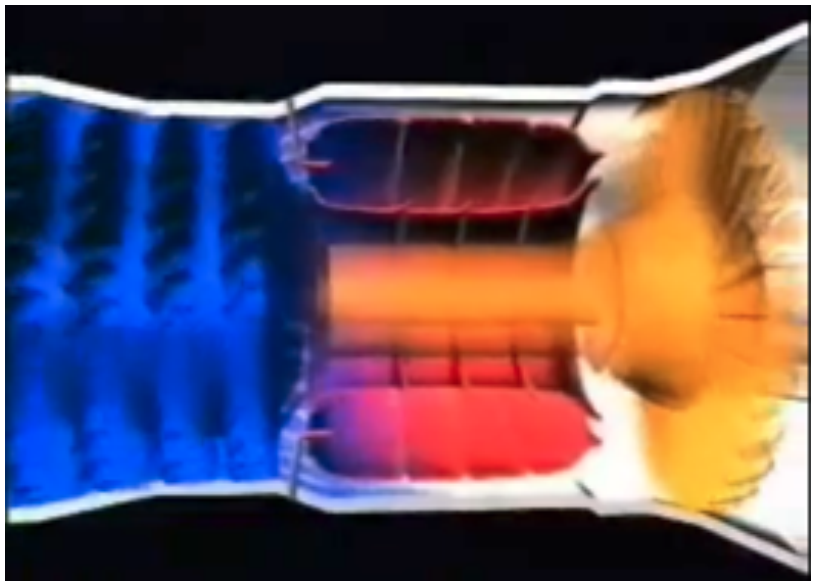




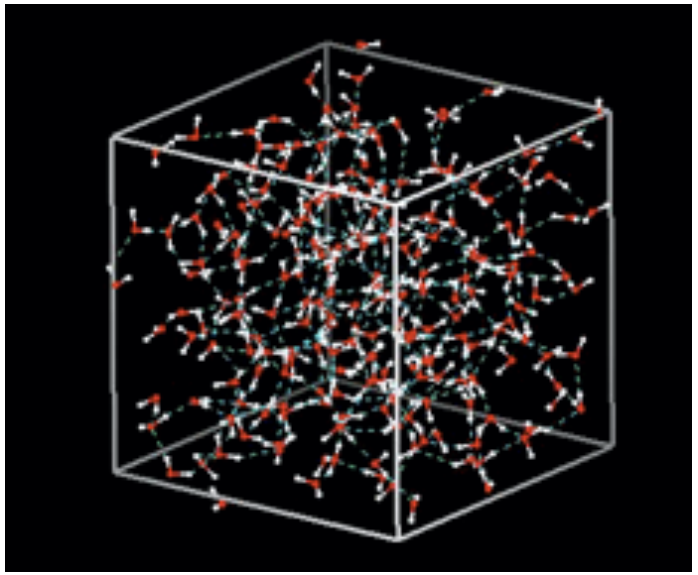


Education

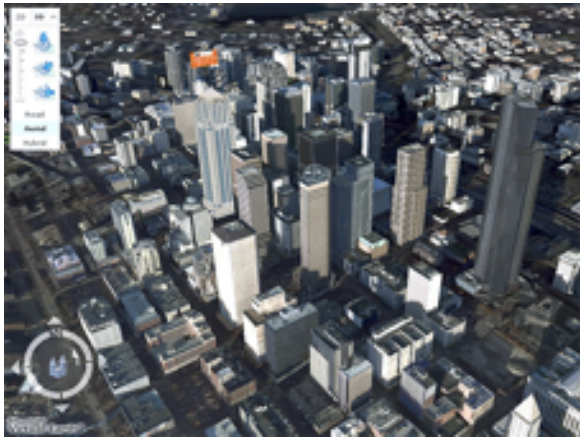

Raising the sphere gives dilation.

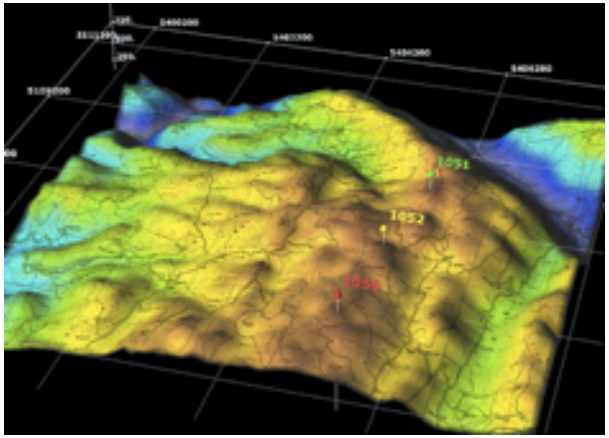


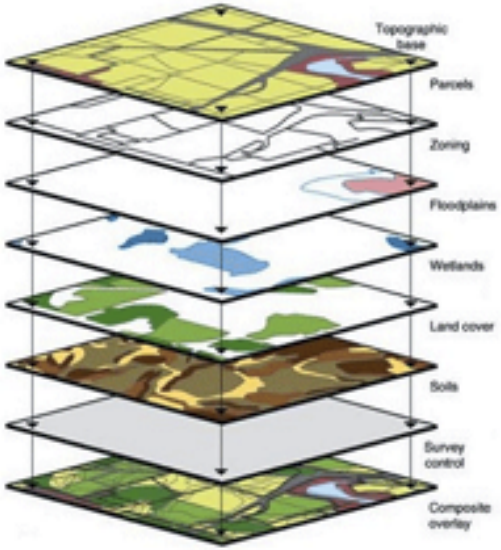

VISIBLE BODY
Hide, show, or fade the anatomy and access definitions.



Geographic Info Systems & GPS

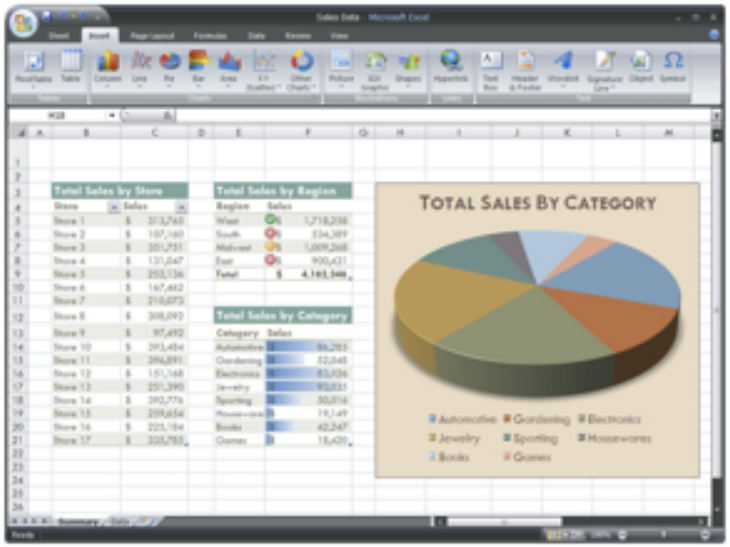


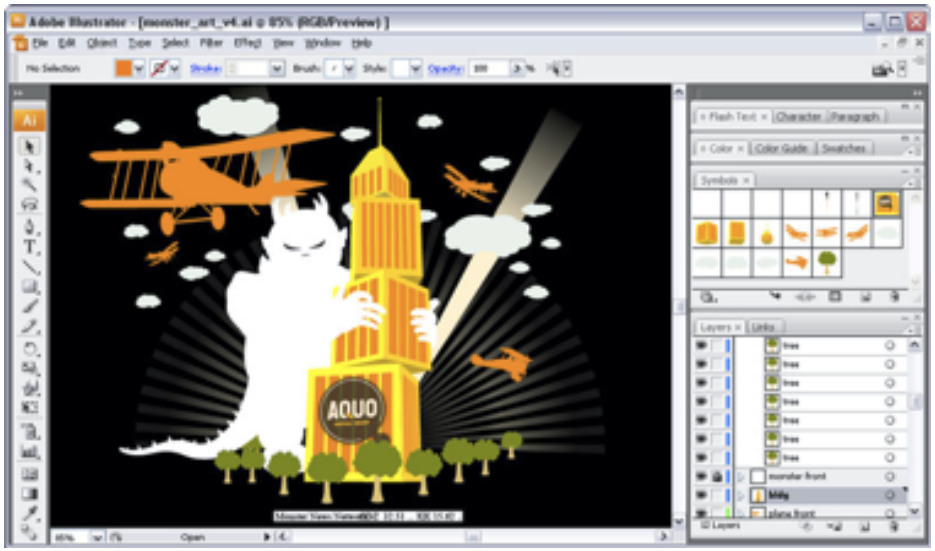




Any display

- Computers go through OpenGL and DirectX to display anything
- 2D graphics, Illustrator, Flash, Fonts





What do you expect to learn?

- And why?

What you will learn in 50.017

- Fundamentals of computer graphics algorithms
  - Will give a pretty good idea of how to implement lots of the things just shown
- We will concentrate on 3D,
  not 2D illustration or image processing
- Basics of real-time rendering
  Basic OpenGL
  - Not the focus, though: Means, not the end.
- You will get extensive C/C++ programming experience
- A little bit Unity (C#) and WebGL

What you will NOT learn in 50.017

- Software packages
  - CAD-CAM, 3D Studio MAX, Maya
  - Photoshop and other painting tools
- Graphics hardware
- Artistic skills
- Game design

How much Math?

- Lots of simple linear algebra
  - Get it right, it will help you a lot!
- Some more advanced concepts
  - Homogeneous coordinates
  - Ordinary differential equations (ODEs) and their numerical solution
  - Sampling, antialiasing (some gentle Fourier analysis)
- Always in a concrete and visual context

Beyond computer graphics

- Many of the mathematical and algorithmic tools are useful in other engineering and scientific context
- Linear algebra
- Splines
- Differential equations
- Monte-Carlo integration
- Machine Learning, Optimization…

Questions?

Plan

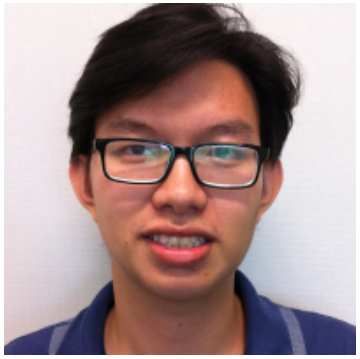
- Overview of computer graphics
- **Administrivia**
- Overview of the semester
- Overview of assignments
- Intro to OpenGL & assignment 0

Team

- Instructor
  - Sai-Kit Yeung
- Teaching Assistant
  - Zhipeng Mo
  - Benjamin Kang
  - Quang-Hieu Pham
  - Minh-Khoi Tran
  - Zheng Hui Er
  - Charmaine Wee








Administrivia: Website, Staff Email

- Course website
  - http://people.sutd.edu.sg/~saikit/courses/50.017/
  - Announcements
  - Slides (posted soon after each lecture)
  - Assignments, both instructions and turn-in
- Email
  - saikit@sutd.edu.sg
  - Zhipeng Mo (zhipeng_mo@mymail.sutd.edu.sg),
  - Minh-Khoi Tran minhkhoi_tran@mymail.sutd.edu.sg),
  - Quang-Hieu Pham (quanghieu_pham@sutd.edu.sg)
  - Benjamin Yue Sheng Kang (benjamin_kang@mymail.sutd.edu.sg)
  - Charmaine Wee (charmaine_wee@sutd.edu.sg)
  - Preferred method of communication

Administrivia: Grading Policy

- Assignments: 42%
  - 1 one-week programming assignments
  - 5 two-week programming assignments
  - Must be completed individually
- Labs: 10%
  - Assignment, project discussions
  - 4 lab sessions on Unity, 1 lab session on WebGL
- Final project: 15%
  - Project proposal: 2%, due on Mar 24th (Friday)
  - Project presentations and report: 13%
    - Presentations on Apr 17th,
    - Report due on Apr 27th
- Quizzes: 8%
  - Beginning of the class (highest 8), edimension
- Midterm Exam: 10%
  - Mar 1st Wed, 9am-1030am
- Final Exam: 15%
  - Apr 27th Thur, 9am-11am
  - Absent in final exam – straight F. Medical reason needs to support with proof.
  - Make up exam will be in oral format.

Administrivia: Prerequisites

- Not strictly enforced
- All assignments are in C/C++
  - Optional review/introductory session on Wed 5pm

- Calculus, Linear Algebra
  - Solving equations, derivatives, integral
  - vectors, matrices, basis, solving systems of equations

Administrivia: Labs

- Mainly conducted by TA and my RA
- Time: 830am Tuesday (not every week..)
- Mid-semester – 10%
  - 4 Sessions on Unity
  - 1 Session on WebGL

Administrivia: Assignments

- Turn in code and executable to edimension
- Always turn in a README file
  - Describe problems, explain partially-working code
    Say how long the assignment took
- Coding style is important
  - Some assignments are cumulative
- Collaboration policy:
  - You can chat, but code on your own
  - Acknowledge your collaboration! (in readme file)
- Late policy:
  - Due @ 11:55pm
  - They may be submitted late by no more than 24 hours, weekend and holiday counted. The penalty for late submission is 20% of the score
  - Medical problems must be documented

Collaboration policy

- Student Conduct:
  - [Important] Students are required to adhere to the University Policy on Academic Standards and Cheating, to the University Statement on Plagiarism and the Documentation of Written Work, and to the Code of Student Conduct.
  - [Important] Also note that posting projects or solutions on public websites requesting or offering to pay for outside assistance is strictly prohibited. It is also not allowed to publicly post your solutions, even this is for free, as it allows other students to copy them and hence results in plagiarism. These are all serious academic dishonesty. Students (whether past or current students) involved in these activities will be identified and sanctioned with no exception.
  - [Important] If Plagiarism is confirmed, straight F will be given to the final grade. Academic warning will be given with potential academic probation for repeated cases.

Collaboration policy

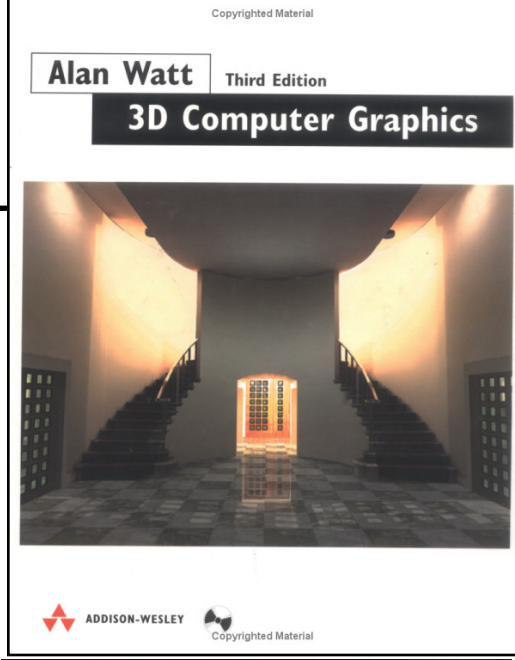
- You can chat, but code on your own (we use automated plagiarism detection software!)
- Acknowledge your collaboration (in README)
- Talk to each other, get a community going
  - Graphics is fun!

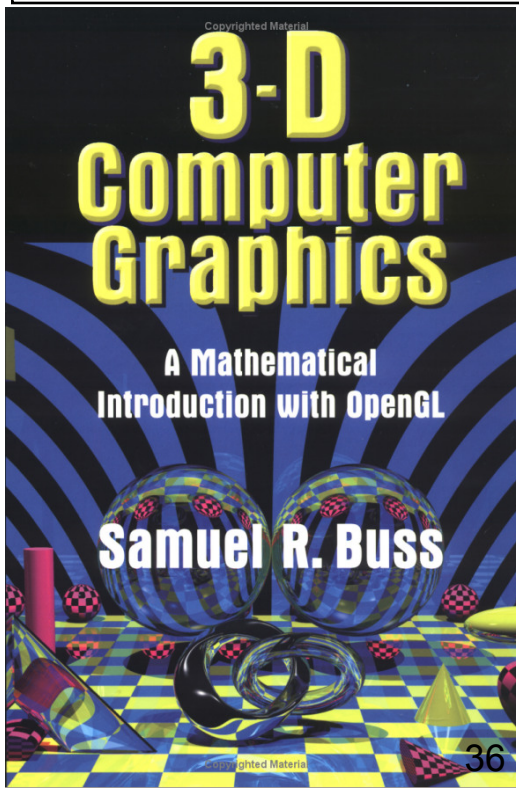
Administrivia: Assignments

- The assignments are a lot of work. Really.
  - Start early!

Textbooks

- No textbook is required
  - Notes will be self-contained
- Recommendations
  - Computer Graphics: Principles and Practice (3rd Edition) (Hughes, Dam, McGuire, Sklar, Foley, Feiner, Akeley)
  - 3D Computer Graphics (Watt)
  - 3D Computer Graphics: A Mathematical Introduction with OpenGL (Buss)
  - Real-Time Rendering, 3rd ed. (Akenine-Möller, Haines, Hoffman)
  - Fundamentals of Computer Graphics, 3rd ed. (Shirley, Marschner)





Questions?

Plan

- Overview of computer graphics
- Administrivia
- **Overview of the semester**
- Overview of assignments
- Intro to OpenGL & assignment 0

How do you make this picture?

Remedy Entertainment / Microsoft Games Studios

How do you make this picture?

How do you make this picture?

How do you make this picture?

- Modeling
  - Geometry
  - Materials
  - Lights
- Animation
  - Make it move
- Rendering
  - I.e., draw the picture!
  - Lighting, shadows, textures...

How do you make this picture?

- Modeling
  - Geometry
  - Materials
  - Lights
- Animation
  - Make it move
- Rendering
  - I.e., draw the picture!
  - Lighting, shadows, textures…

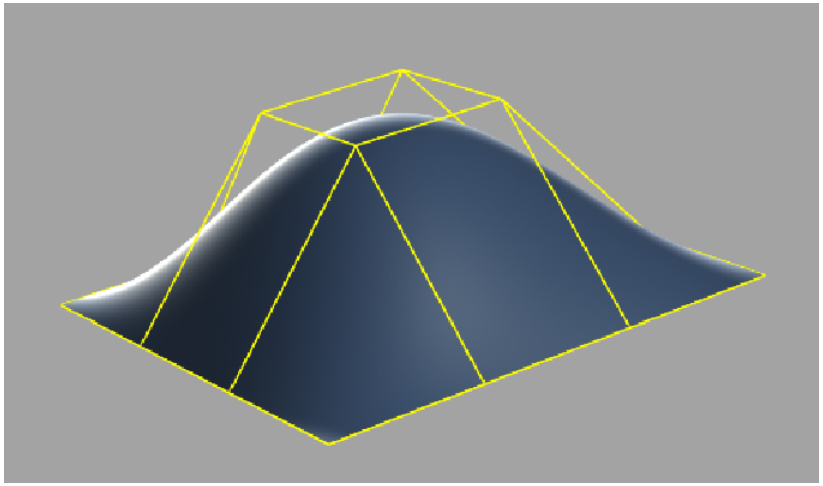
Questions?

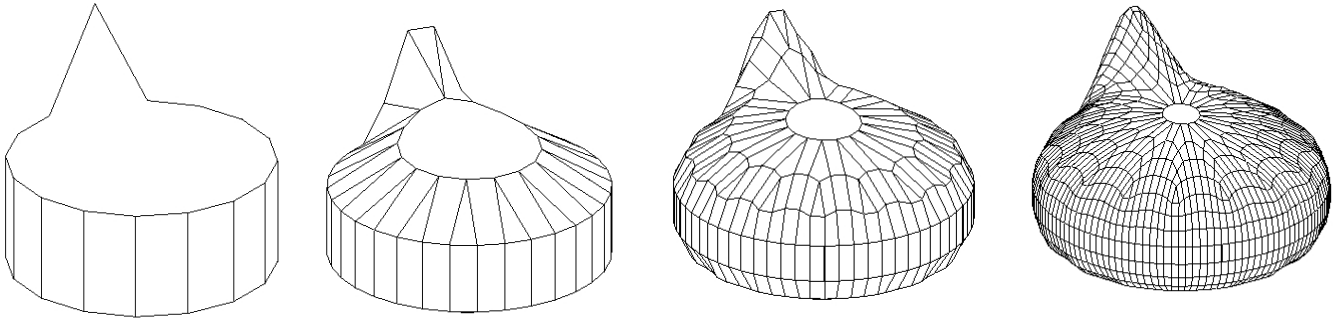
Overview of the Semester

- Modeling, Transformations
- Animation
- Ray Casting / Ray Tracing
- Ray Tracing Acceleration, Global Illumination
- Sampling, Textures
- The Graphics Pipeline

Modeling

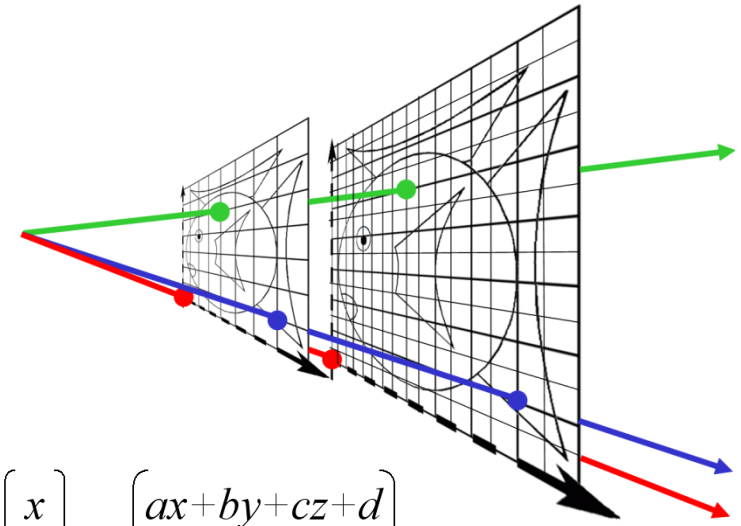
- Curves and surfaces
- Subdivision surfaces





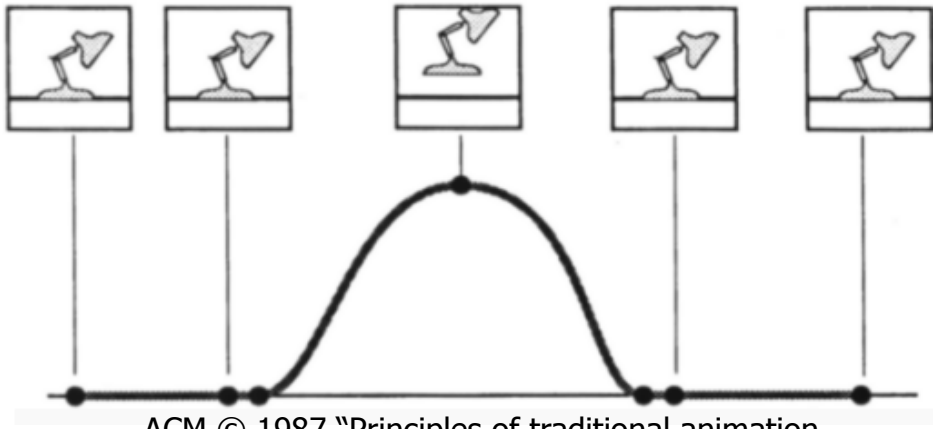
Transformations

- Yep, good old linear algebra
- Homogeneous coordinates
  - (Adding dimensions to make life harder)
- Perspective



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} ax+by+cz+d \\ ex+fy+gz+h \\ ix+jy+kz+l \\ 1 \end{bmatrix}$$

Animation: Keyframing



ACM © 1987 "Principles of traditional animation applied to 3D computer animation"

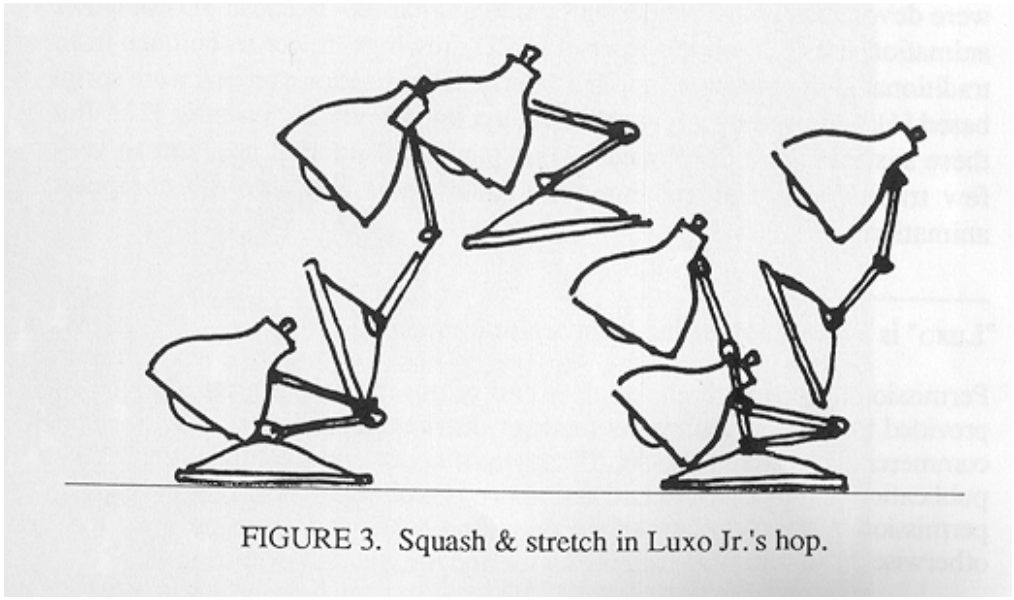
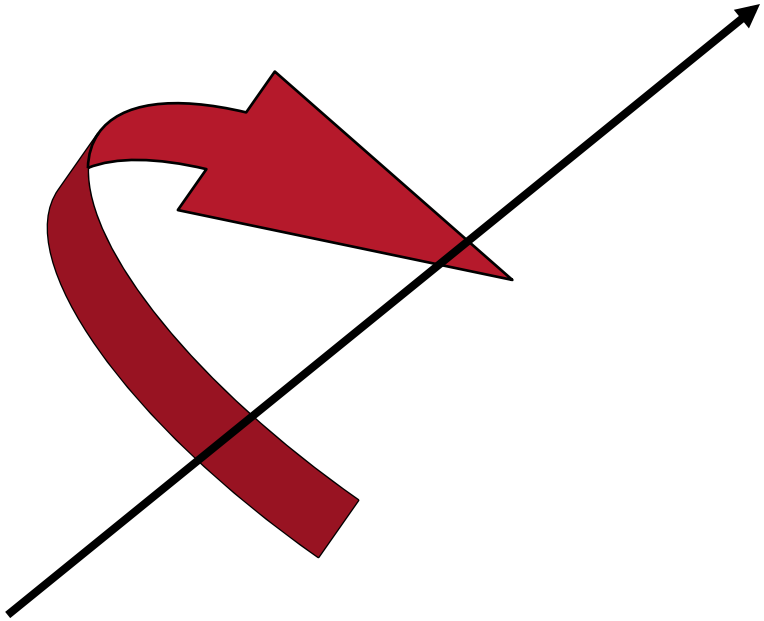
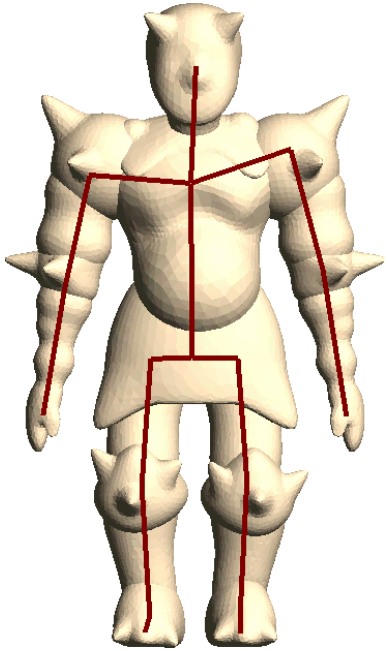


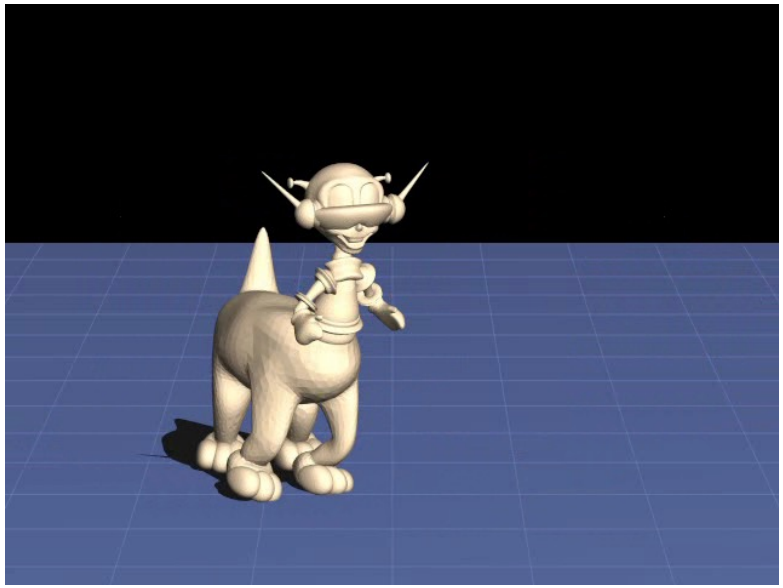
FIGURE 3. Squash & stretch in Luxo Jr.'s hop.



Character Animation: Skinning

- Animate simple “skeleton”
- Attach “skin” to skeleton
  - Skin deforms smoothly with skeleton
- Used everywhere (games, movies)



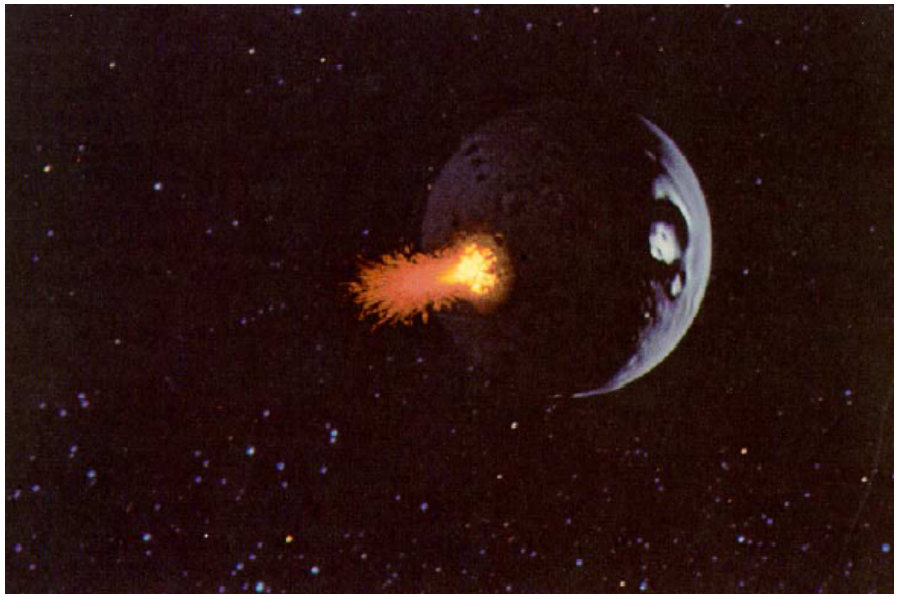


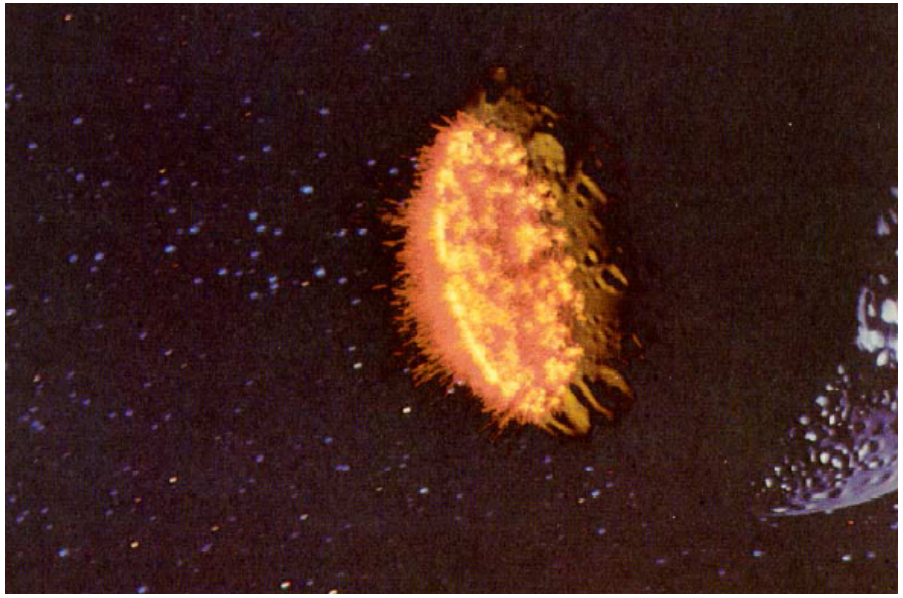
Ilya Baran

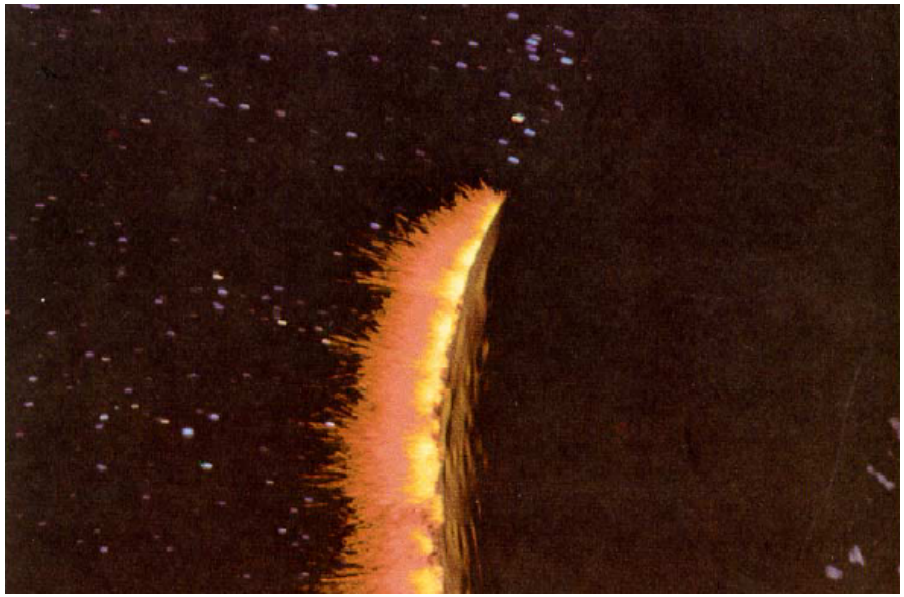
Epic Games

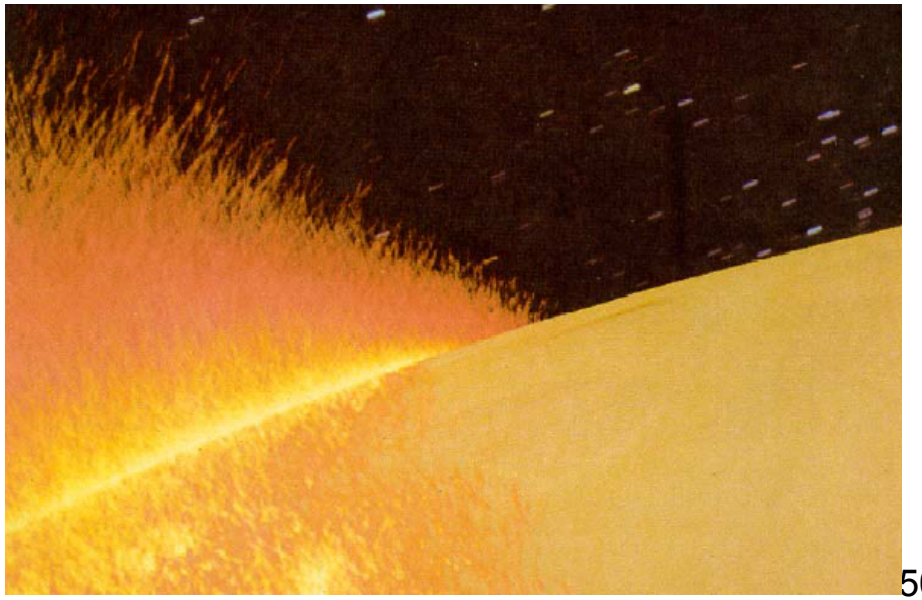
49

Particle systems



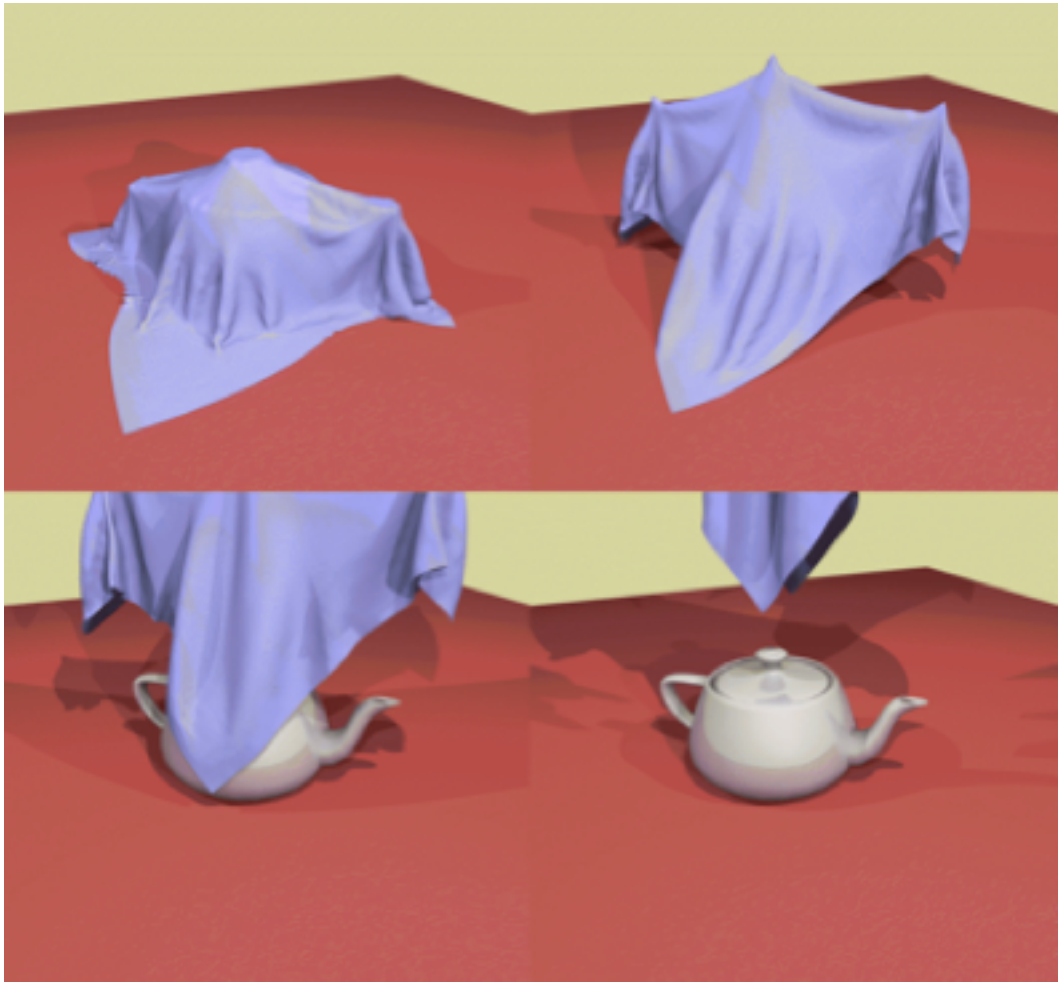






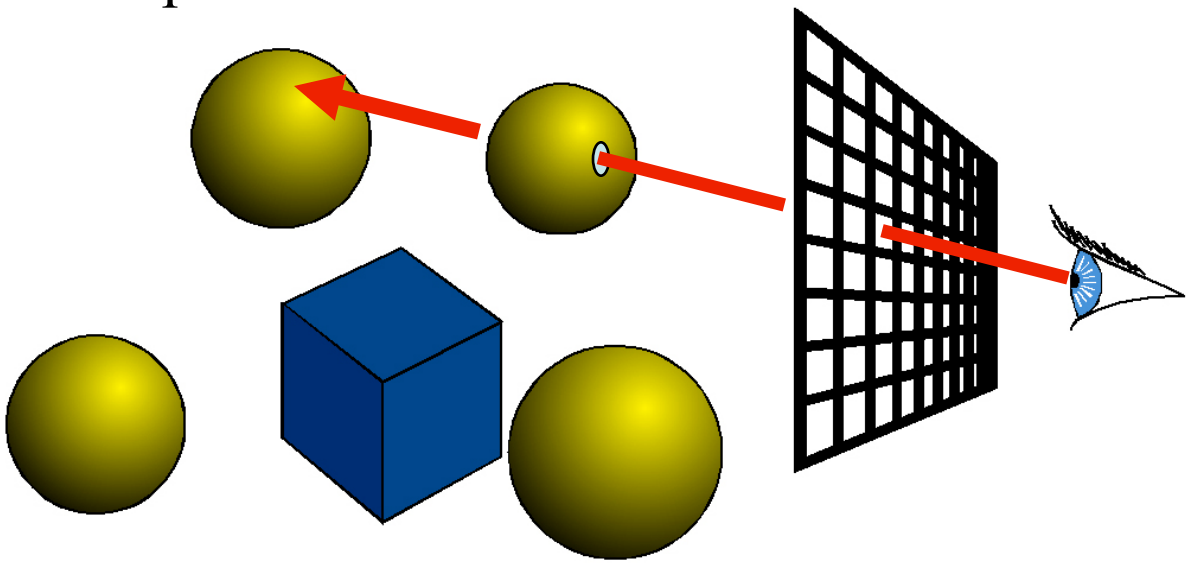
“Physics” (ODEs)

- Fire, smoke
- Cloth



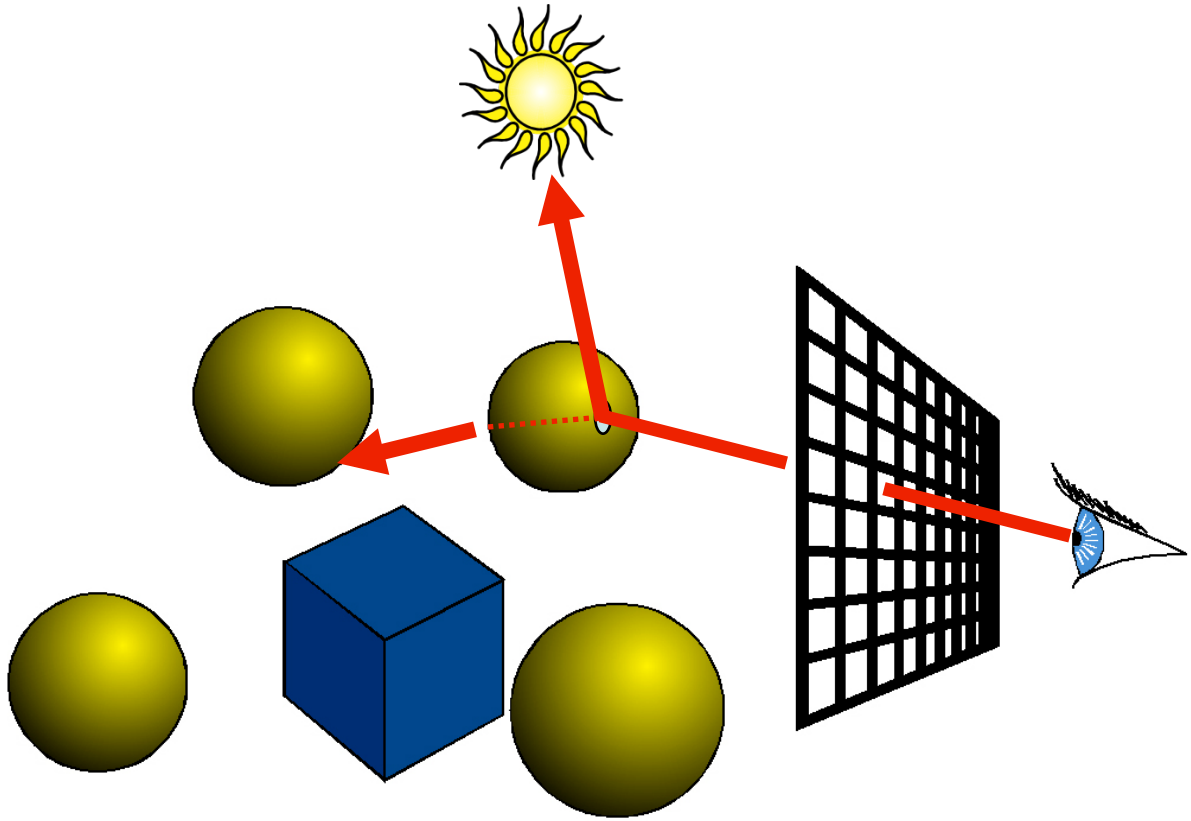
Ray Casting

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



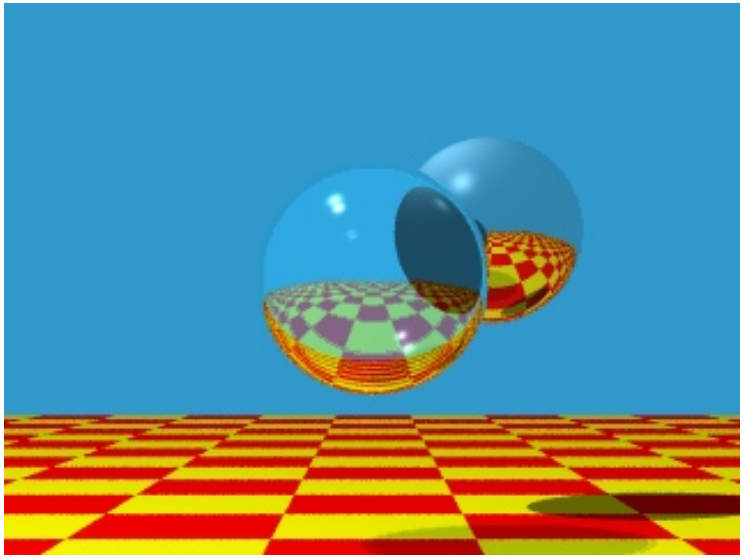
Ray Tracing

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



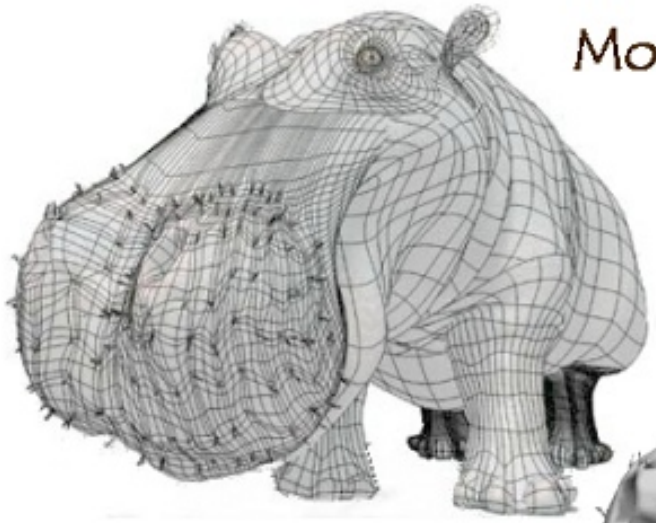
Ray Tracing

- Original Ray-traced image by Whitted



- Image computed using the Dali ray tracer by Henrik Wann Jensen
- Environment map by Paul Debevec

Textures and Shading



Model

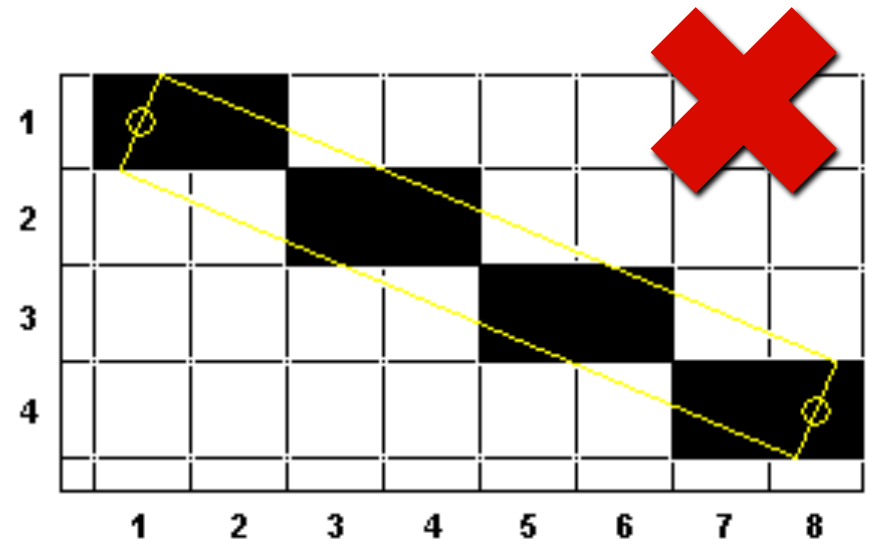
Model + Shading

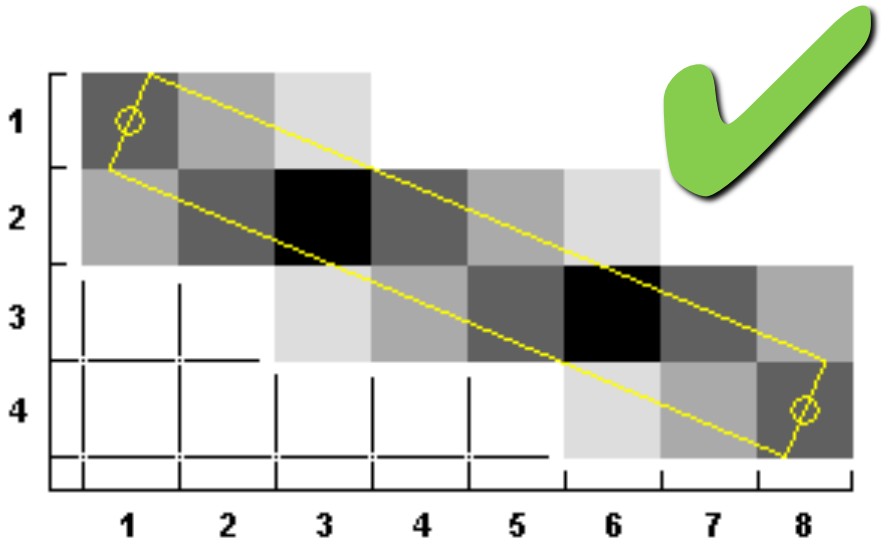
Model + Shading + Textures

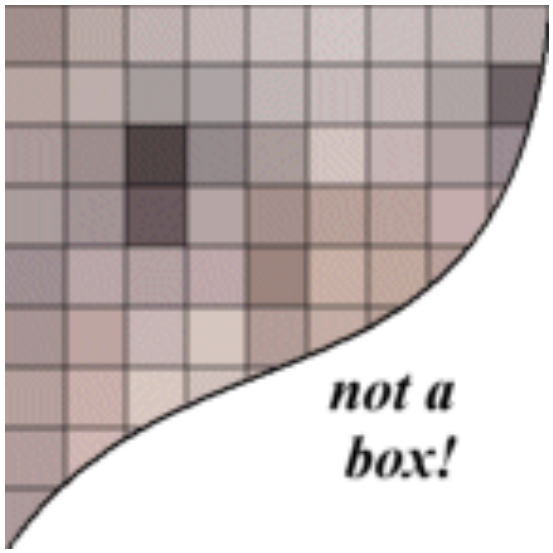
At what point do things start looking real?

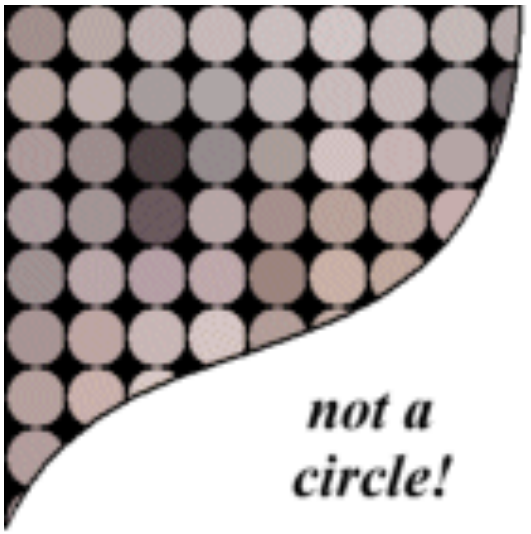
For more info on the computer artwork of Jeremy Birn see http://www.3drender.com/jbirn/productions.html

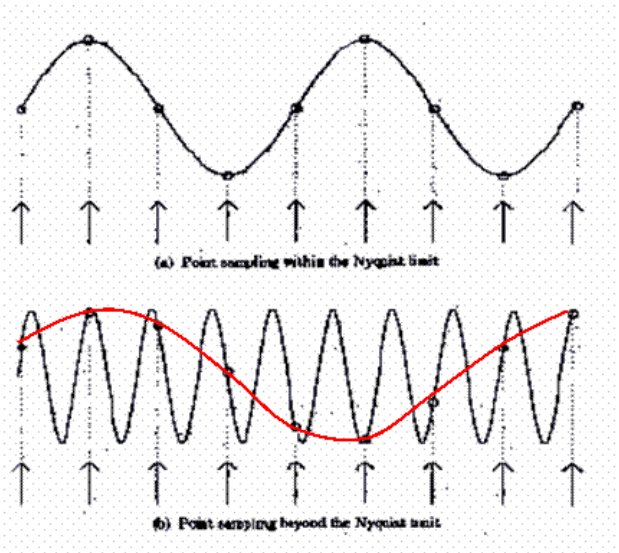
Sampling & Antialiasing



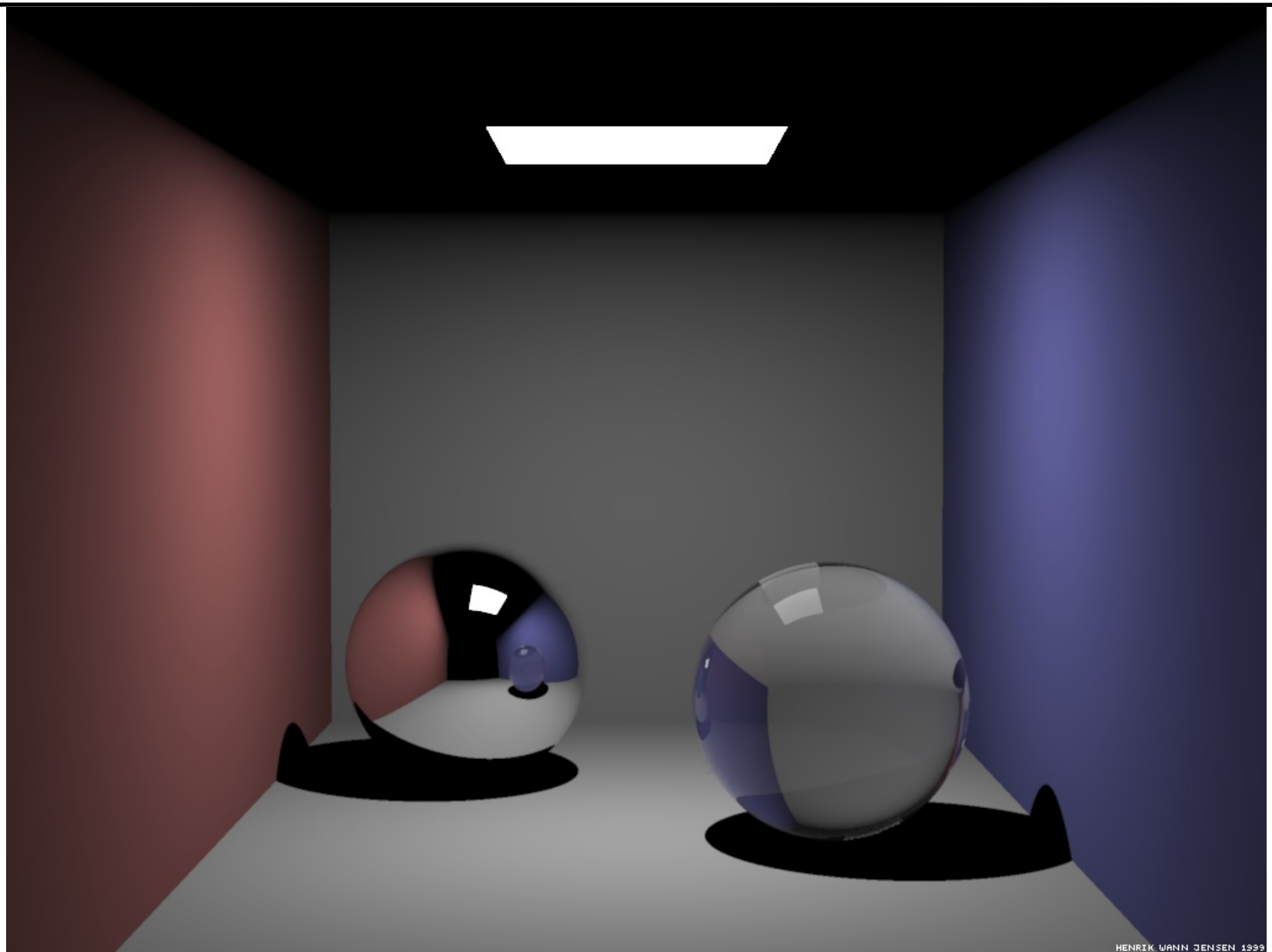




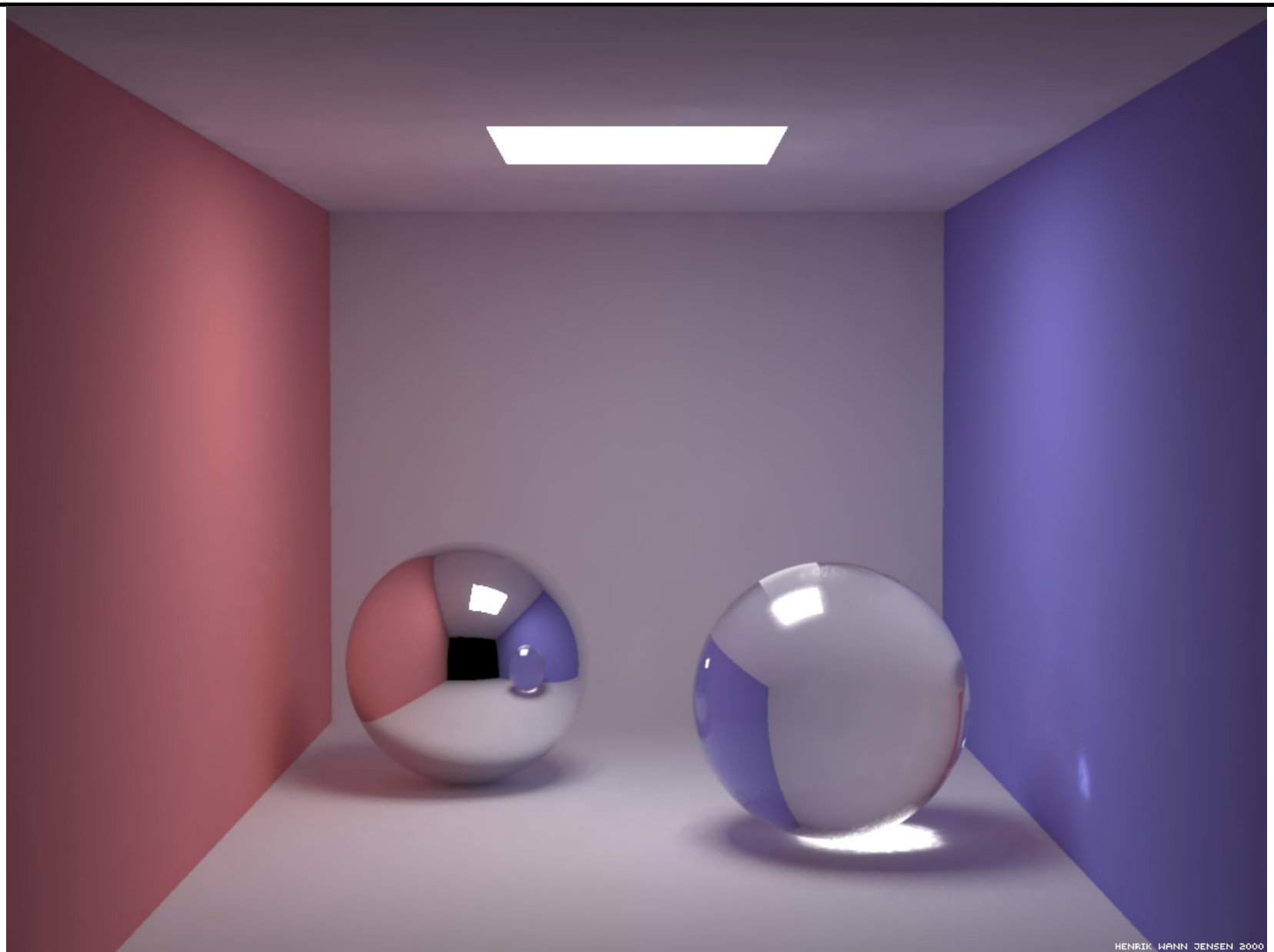




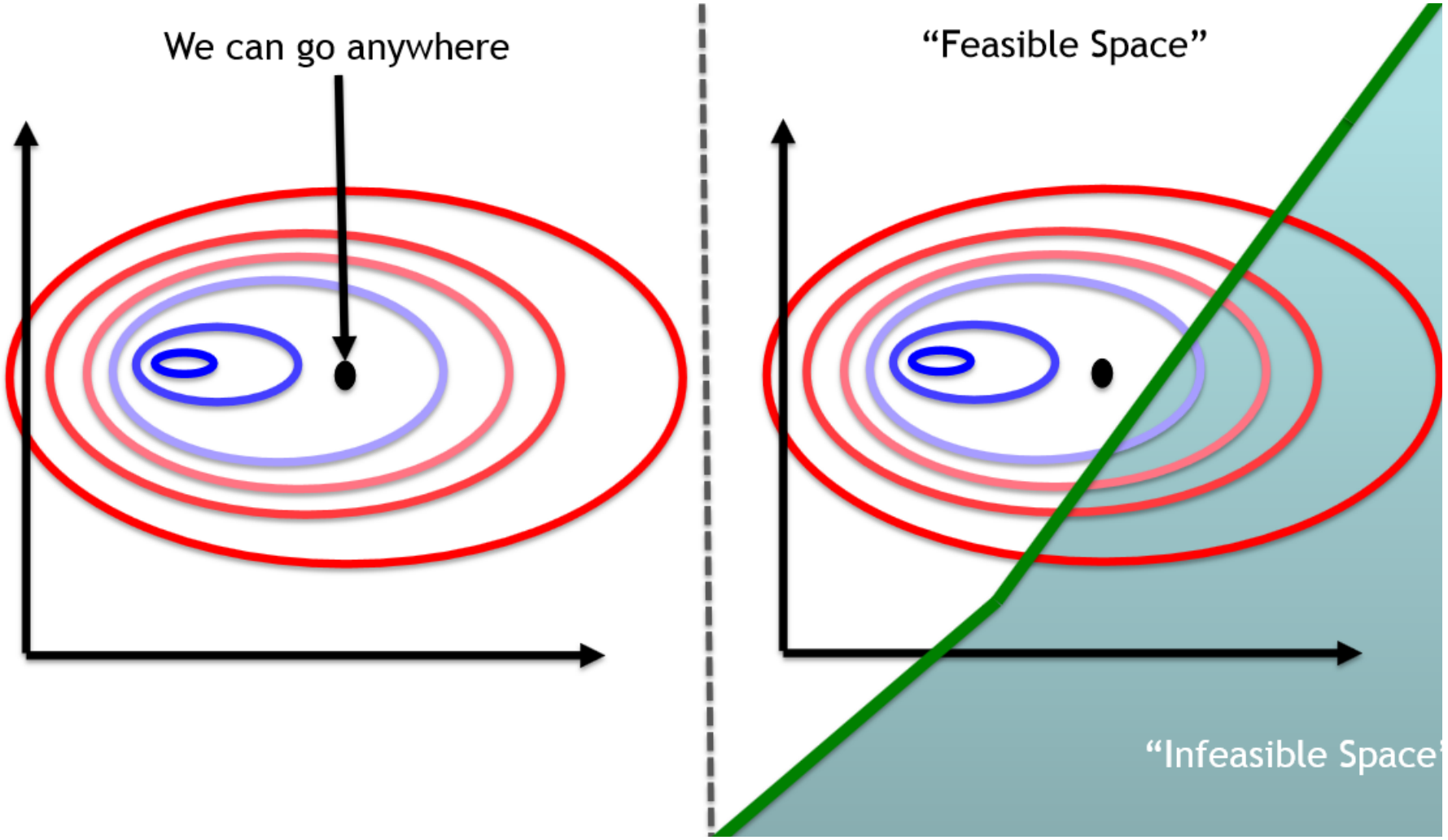
Traditional Ray Tracing



Global Illumination



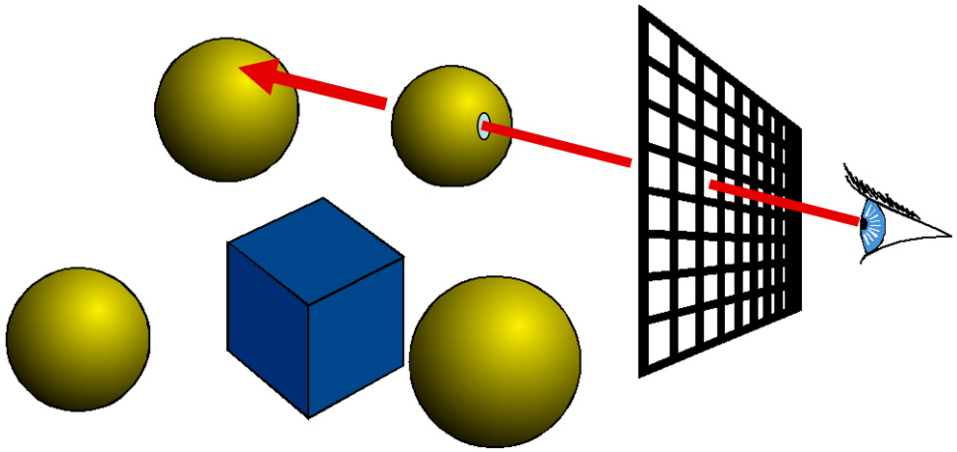
Machine Learning and Optimization



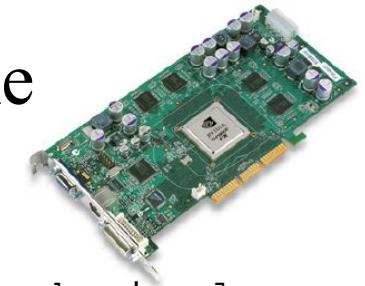
The Graphics Pipeline

Ray Casting

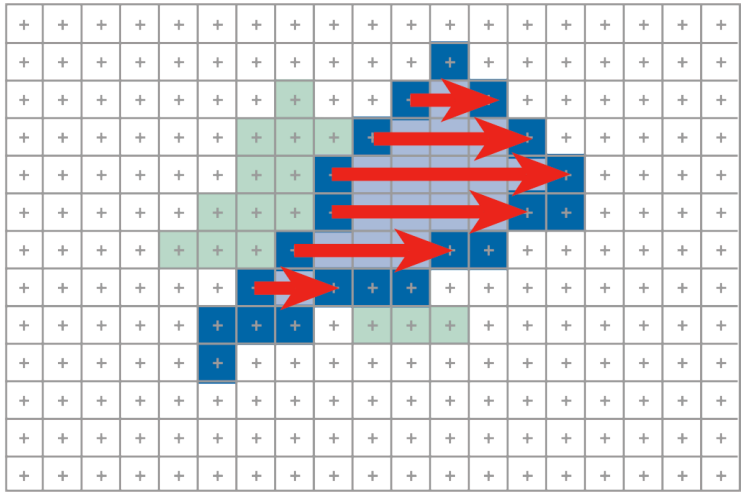
```
For each pixel
    For each object
        Send pixels to scene
```



Rendering Pipeline

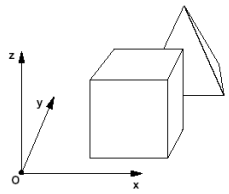


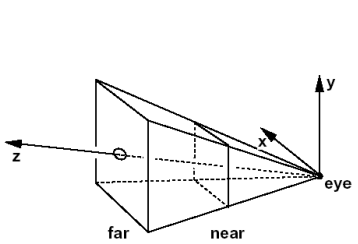
```
For each triangle
    For each projected pixel
        Project scene to pixels
```

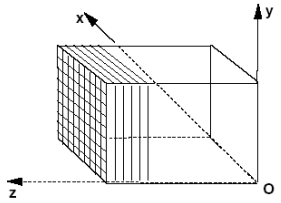


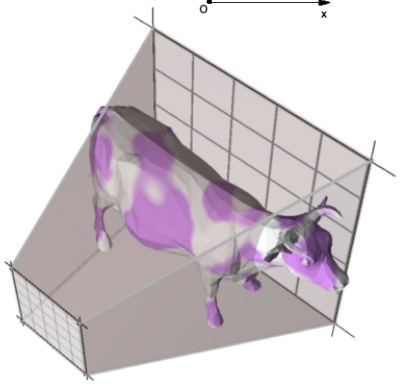
The Graphics Pipeline

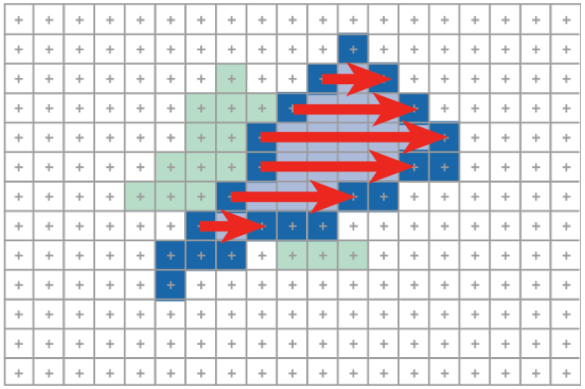
- Transformations
- Clipping
- Rasterization
- Visibility



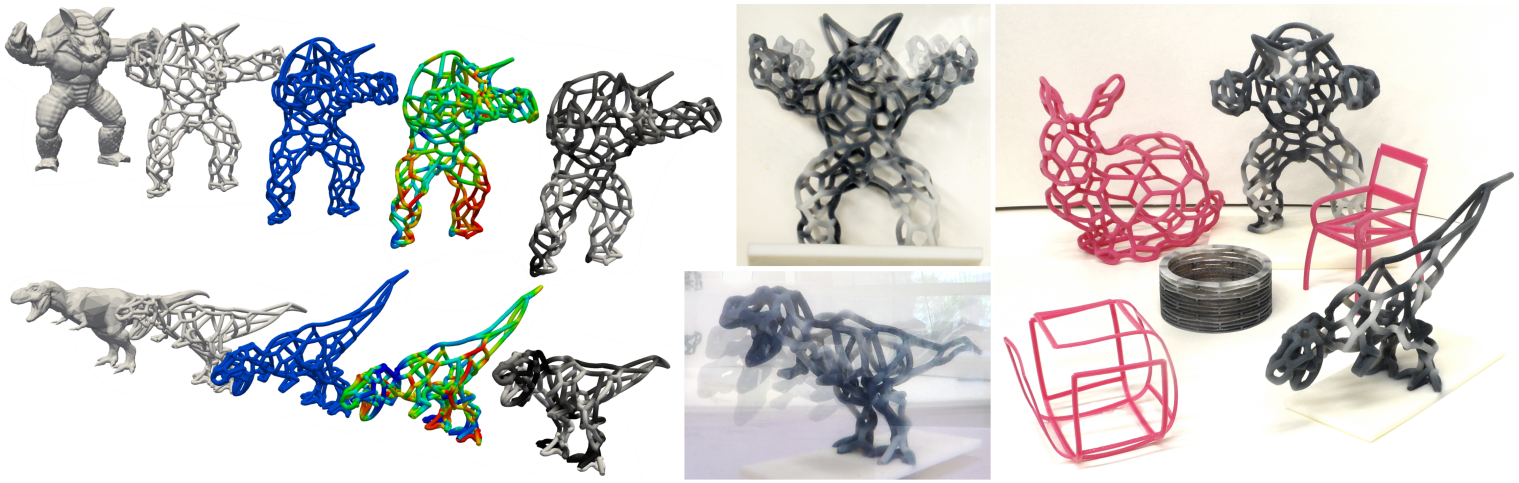


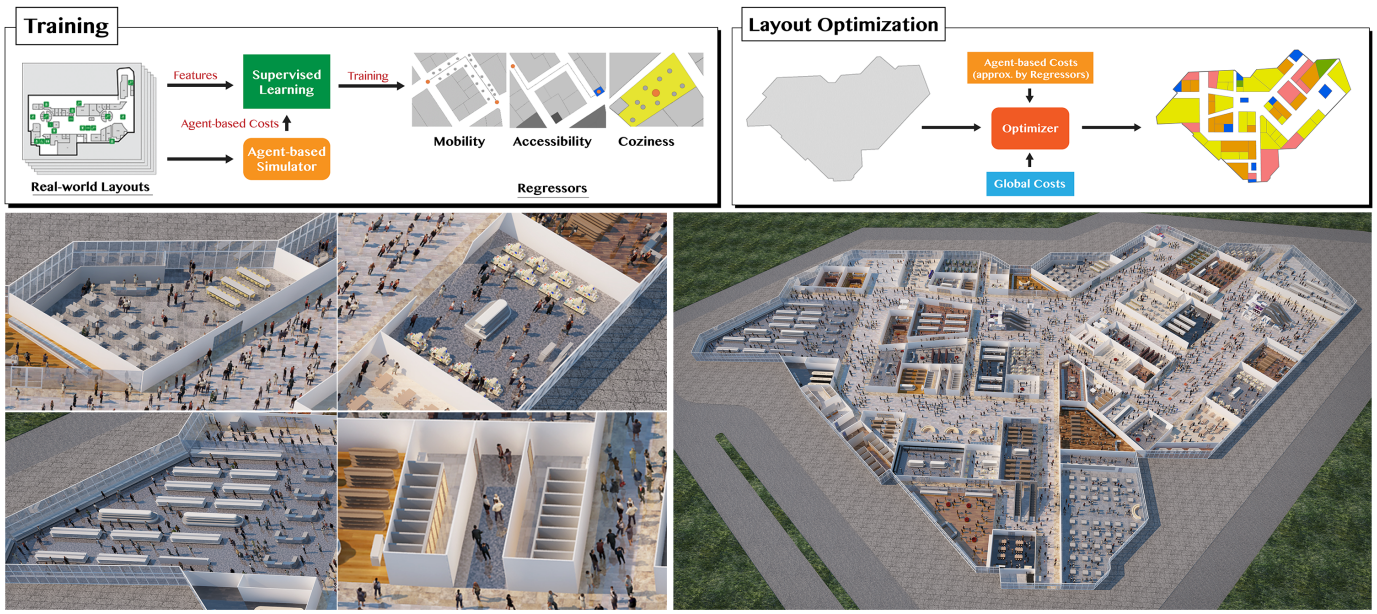






Latest Graphics Research

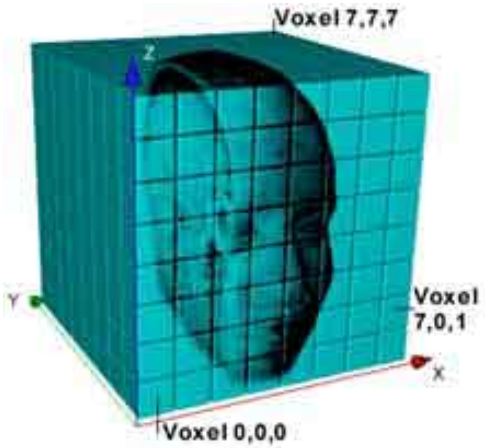


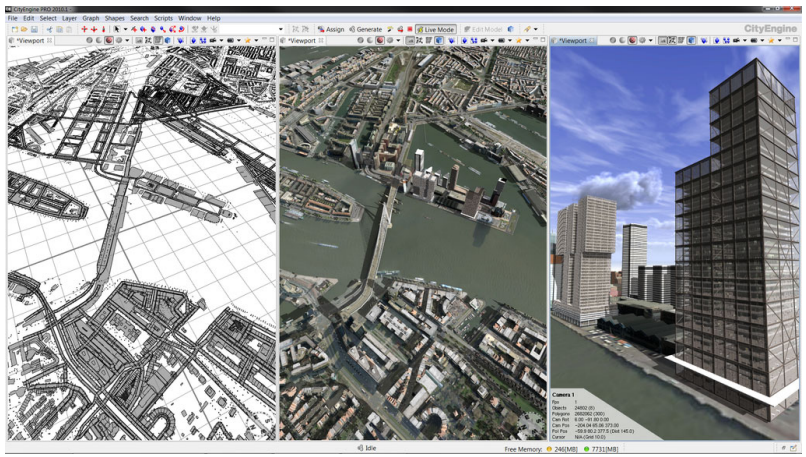


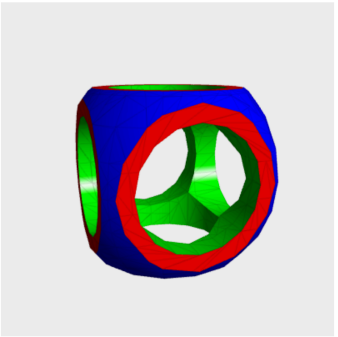
Questions?

Advanced Modeling

- Computational Fabrication (TAE) – Term 8
  - Solid Modeling
  - Constructive Solid Modeling (CSG)
  - Procedural Modeling
  - 3D Scanning
  - Simulation







Plan

- Overview of computer graphics
- Administrivia
- Overview of the semester
- **Overview of assignments**
- Intro to OpenGL & assignment 0

Assignments

0: Warm up (mesh display with OpenGL)

1: Curves & surfaces

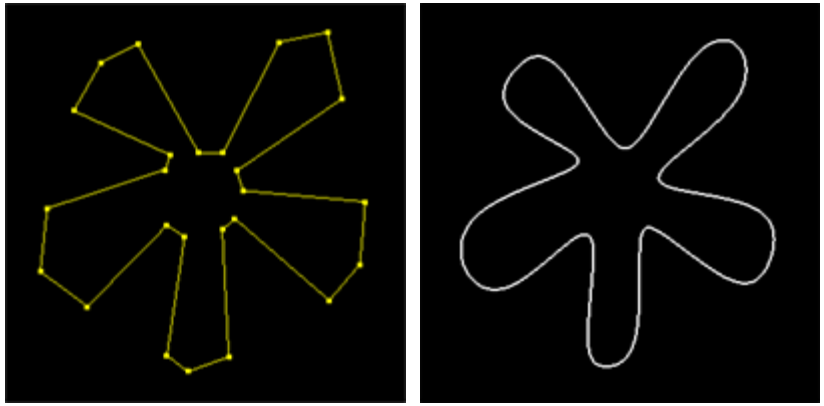
2: Hierarchical modeling, skinning

3: Physically-based simulation

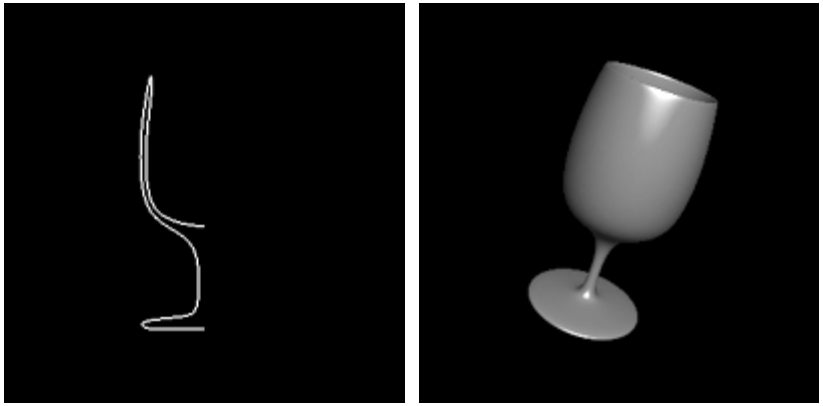
4: Ray casting

5: Ray tracing

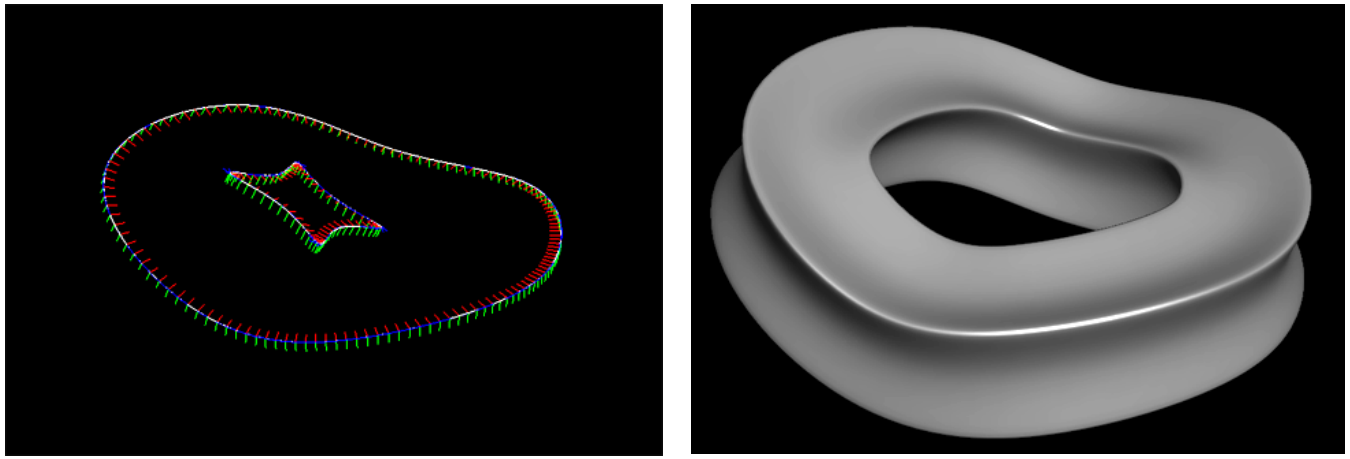
Assignment 1: curves & surfaces



Bezier curves



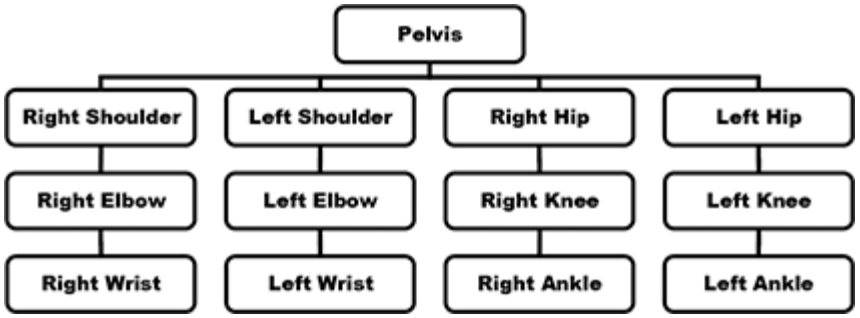
Surfaces of revolution

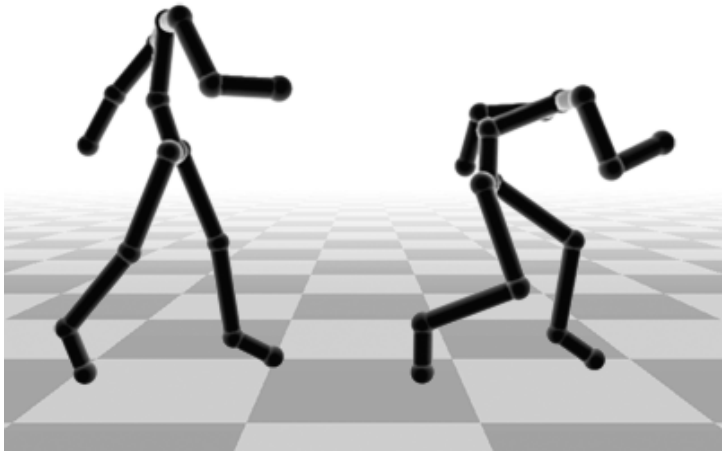


Sweep surfaces

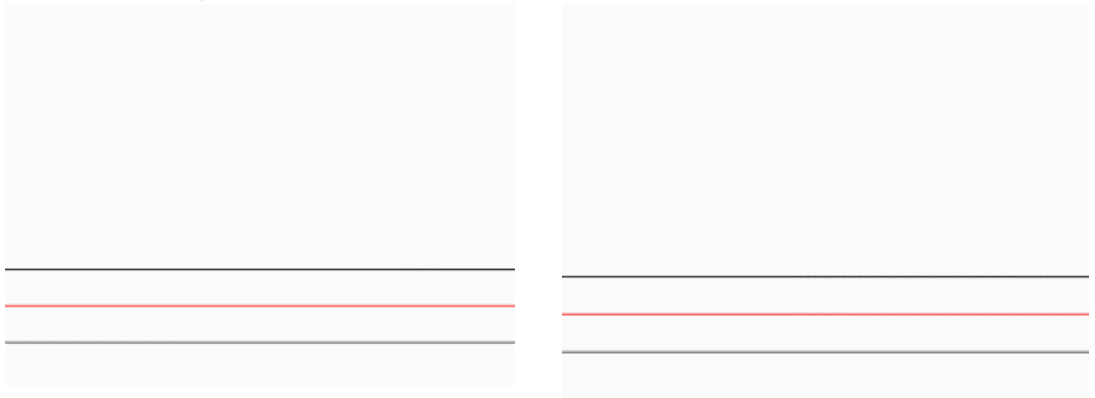
Assignment 2: hierarchical modeling

- Animate character skeleton as tree of transformations



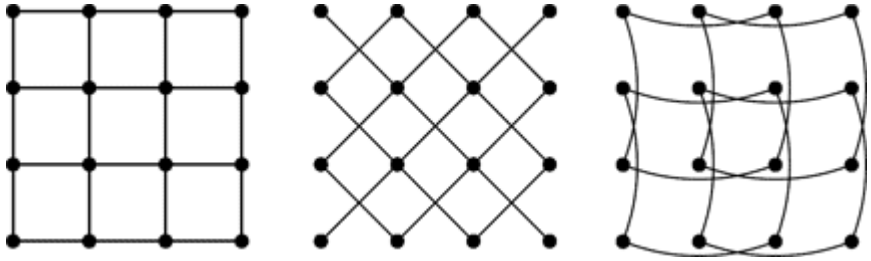


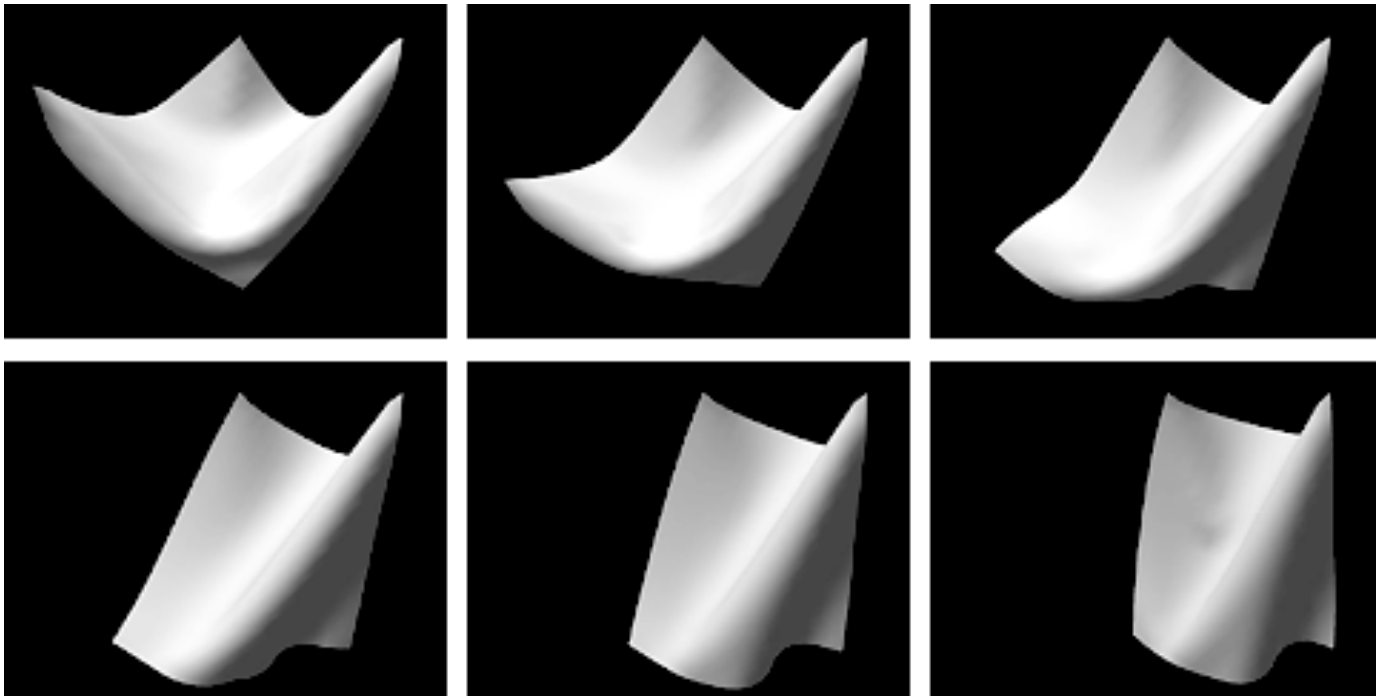
- Skinning: smooth surface deformation



Assignment 3: physics

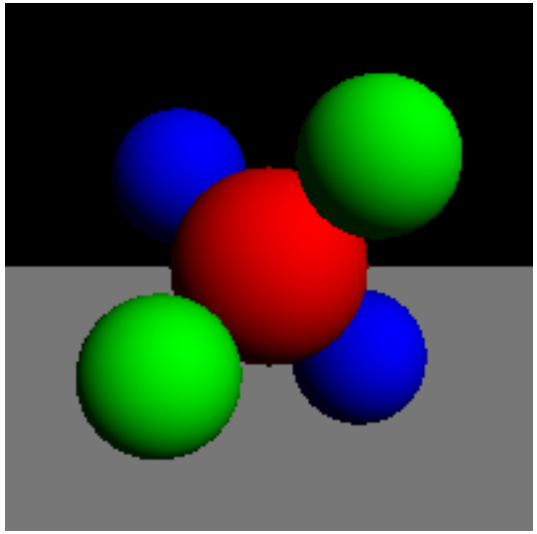
- Simulate cloth as a mass-spring network
  - ODE integration

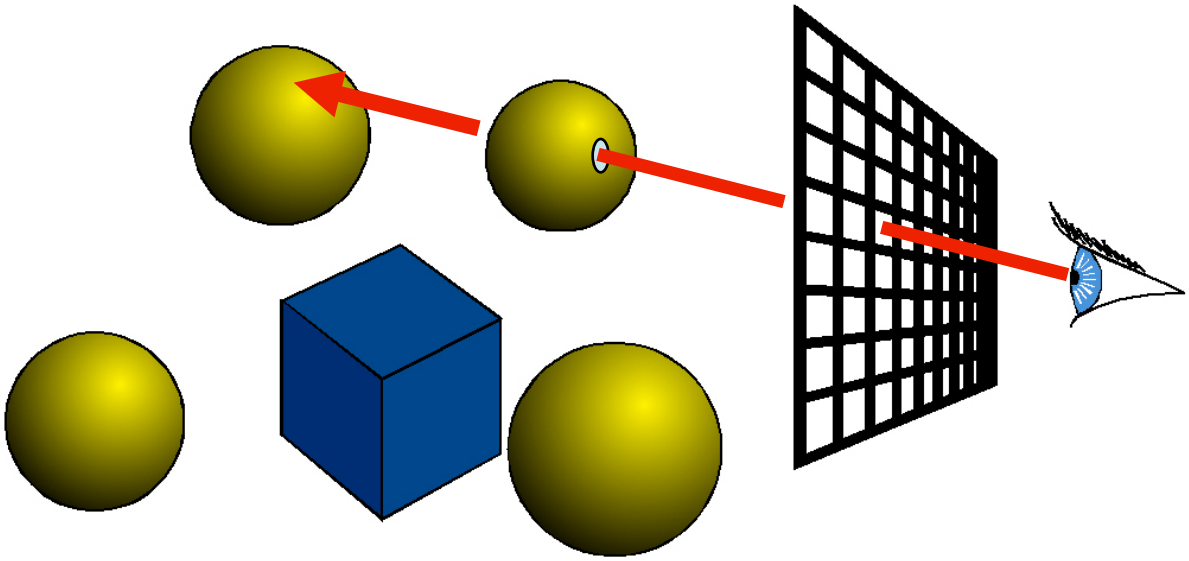




Assignment 4: ray casting

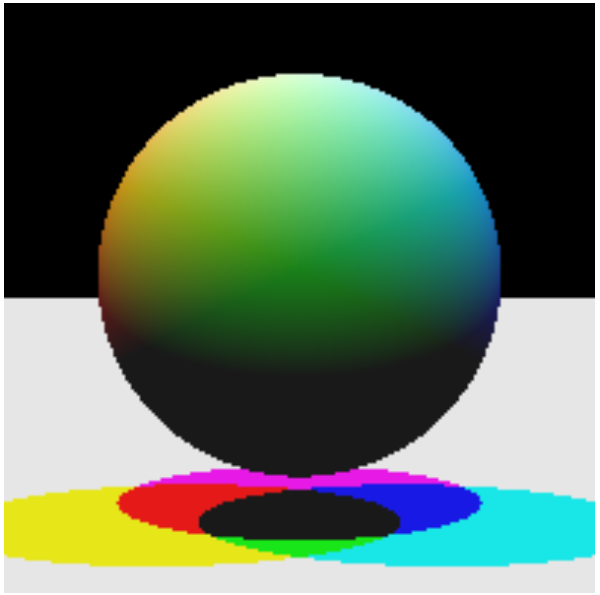
- Cast rays from the viewpoint
- Intersect with scene primitives

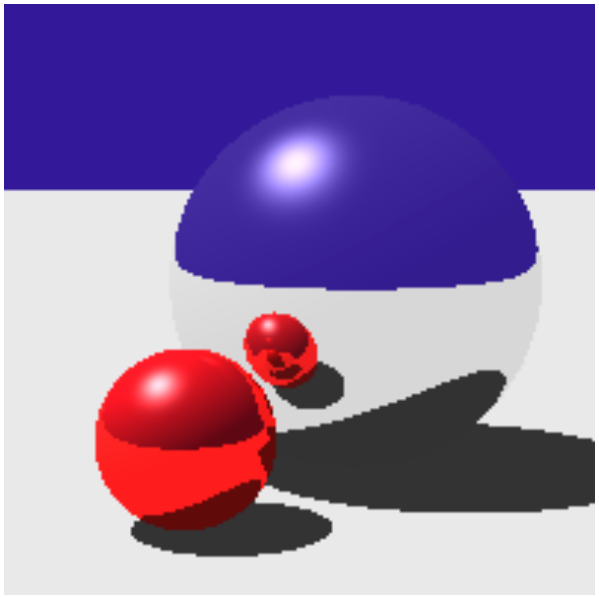


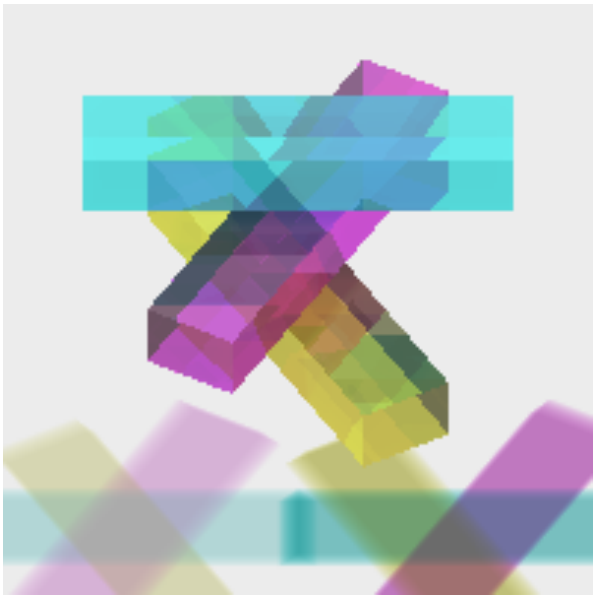


Assignment 5: ray tracing

- Shadows, reflection, refraction
- + flexible extension





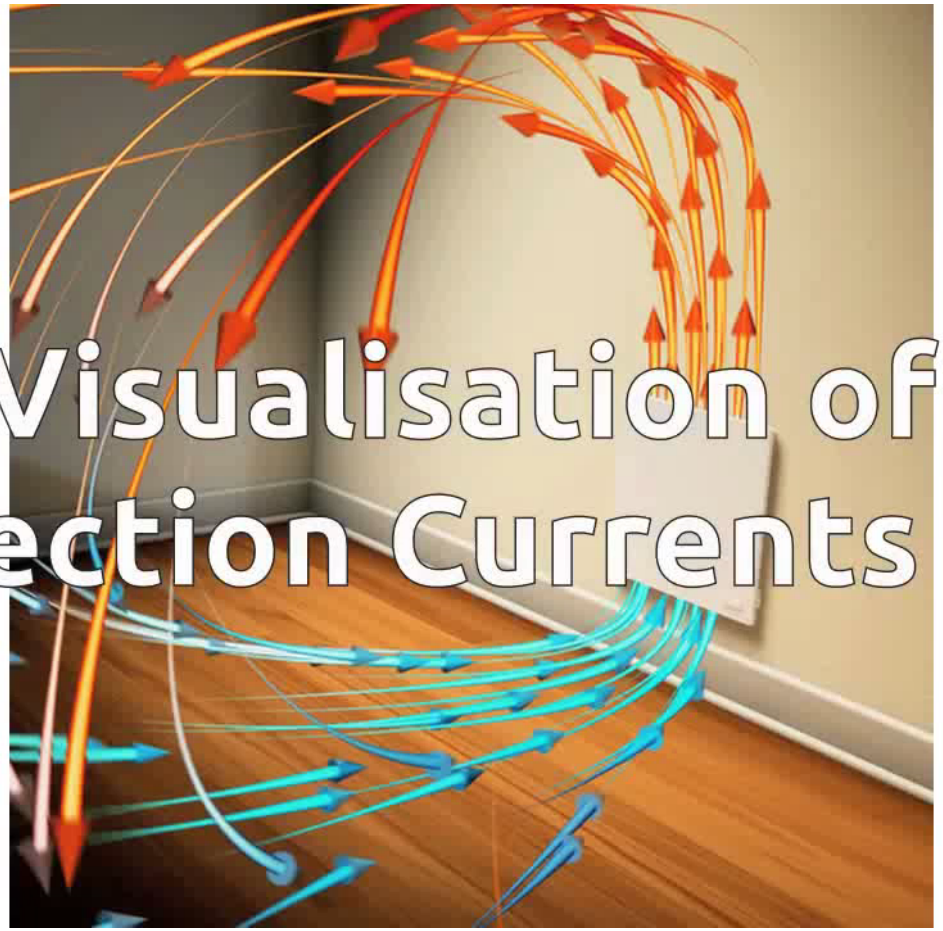


Final Projects

- Project proposals due on Mar 24th
- Presentation on Apr 17th
- Final report due on Apr 27th
- Scope
  - Built from scratch **OR**
  - A significant extension of one of the assignments
- Done in group
- Any platform (e.g., mobile, web, etc.)
- Projects judged by all the instructor and TA

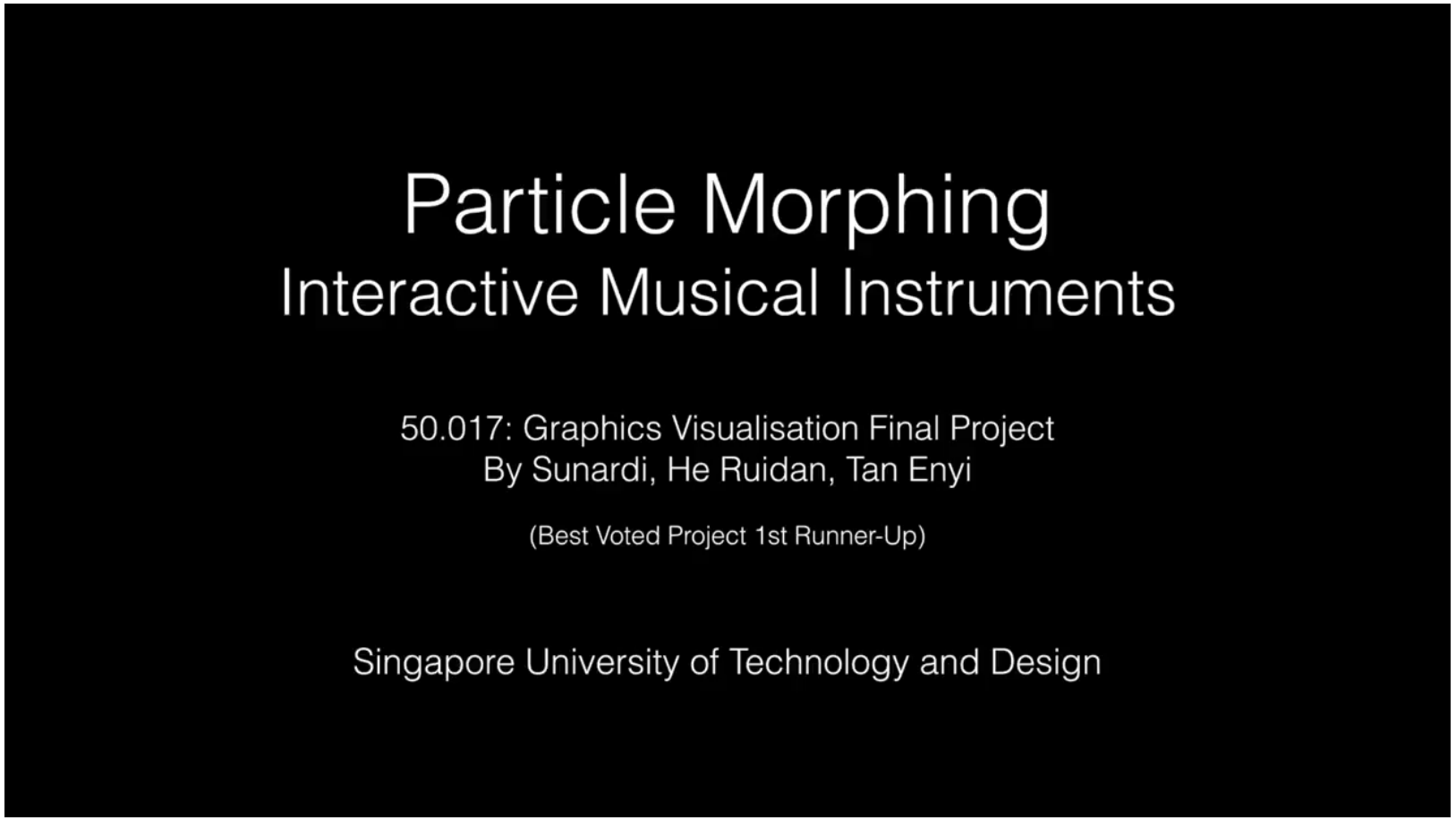
Final Projects

- Term 7, 2015
  - Student voted 2nd runner up



Final Projects

- Term 7, 2015
  - Student voted 1st runner up



Final Projects

- Term 7, 2015
  - Student voted best

Questions?

Plan

- Overview of computer graphics
- Administrivia
- Overview of the semester
- Overview of assignments
- **Intro to OpenGL & assignment 0**

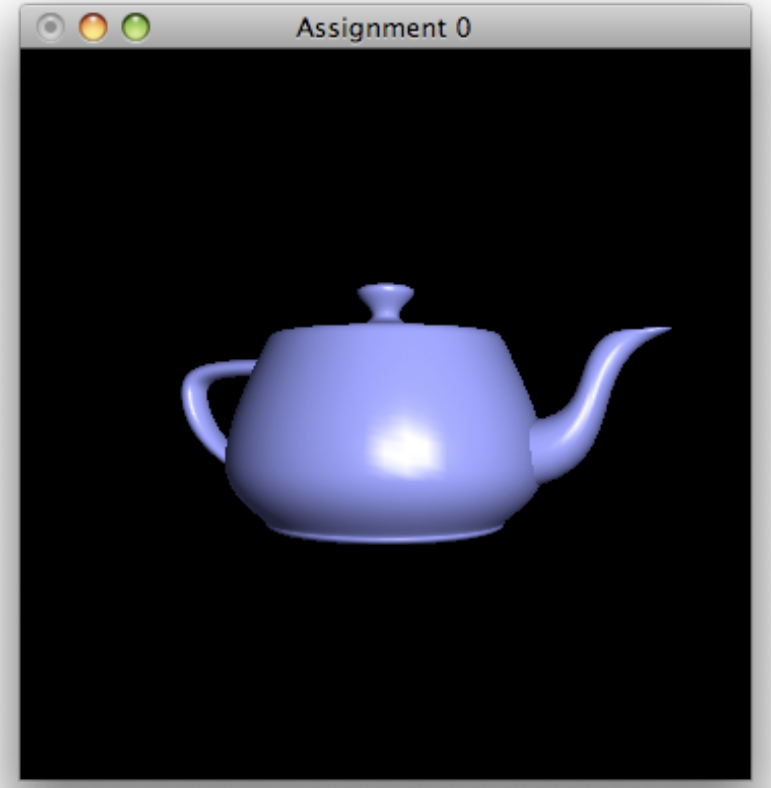
Simple 3D with OpenGL

- OpenGL is an API that allows you to send commands to the graphics card to draw 2D or 3D scenes
- At the beginning of the semester, we will use OpenGL as a black box to display 3D content
- Later, we will see what is under the hood

Assignment 0

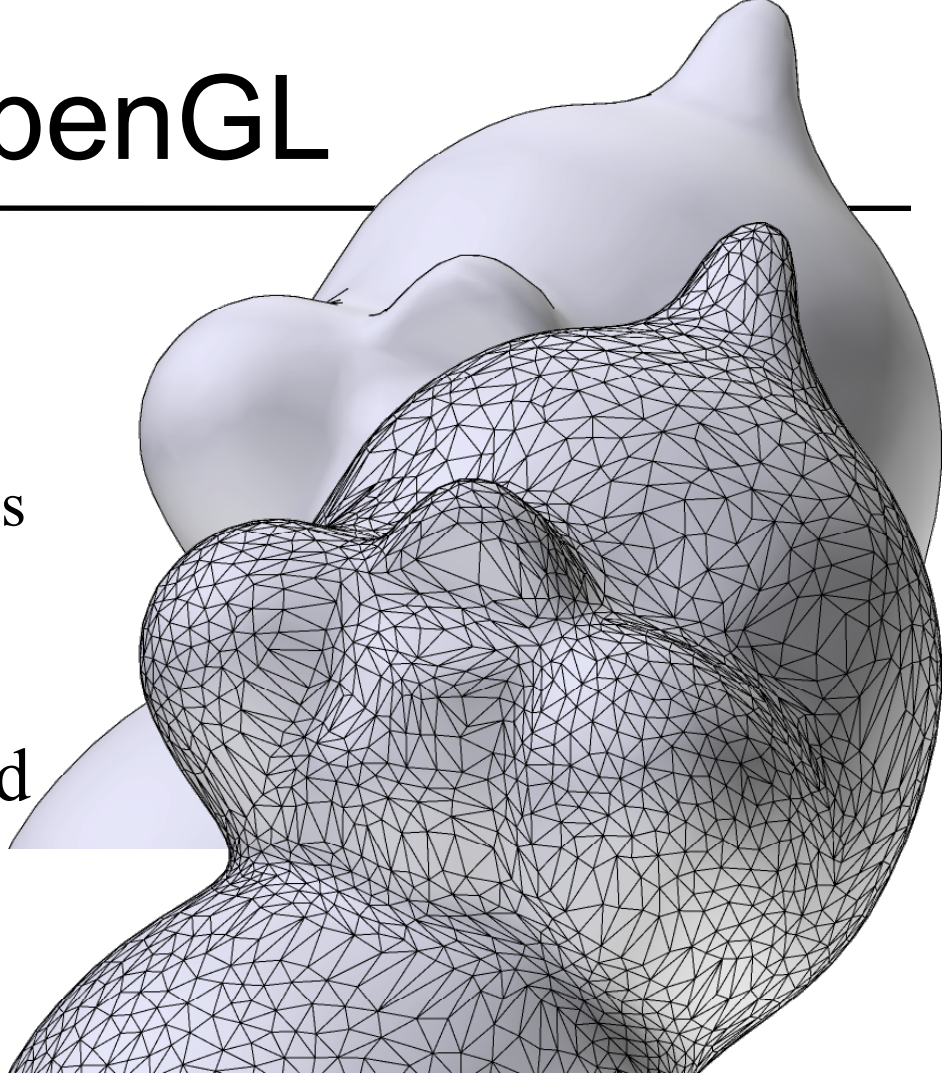
- Read a file with triangle mesh data
  - Including mesh normals
- Display it using OpenGL
  - Colors, simple movement

- Due next Monday!

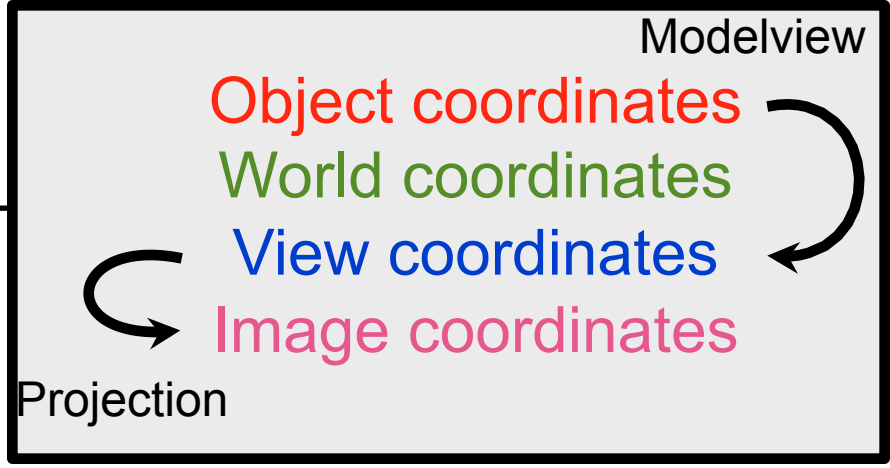


Simple 3D with OpenGL

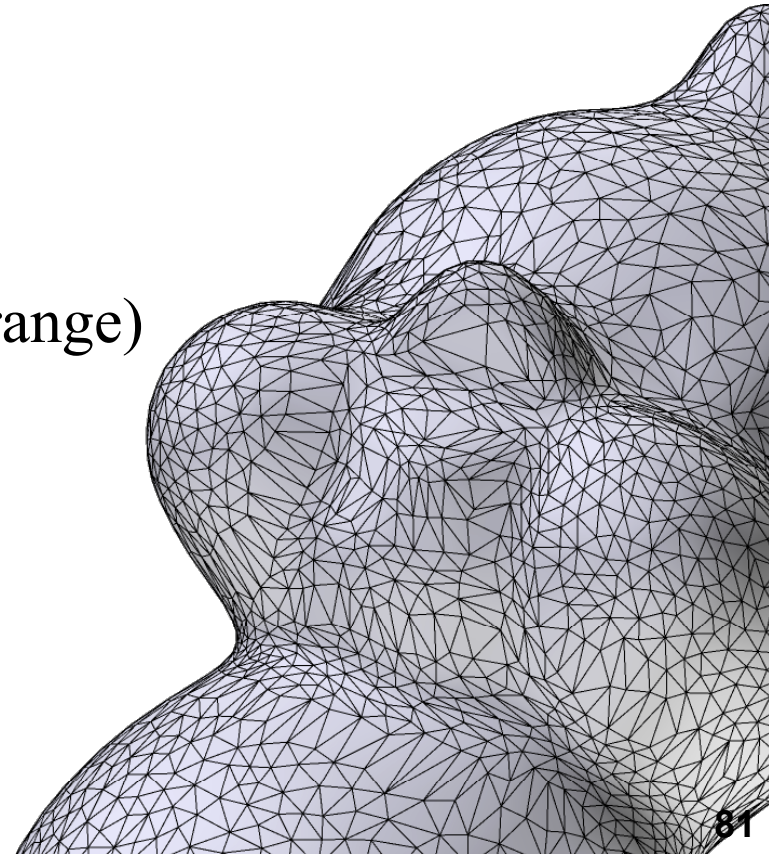
- Scene represented as triangles
  - A triangle is a set of 3 vertices
  - A vertex is a set of 3 floating point numbers (x, y, z)
- We will use OpenGL to send this to the graphics card (GPU)
  - The GPU will do its magic to display the scene from the current viewpoint



How to Draw?



- You need to tell OpenGL
  - The geometry of the object
    - Vertex positions
    - Vertex normals
    - 3 x vertex makes a triangle!
  - Camera parameters
    - Field of view, aspect ratio, (depth range)
    - The “projection matrix”



Questions?

OpenGL high-level pseudocode

- Initialize
  - (get graphics context, etc.)
- For each frame
  - Manage UI
  - Set appropriate viewpoint
  - Set light source directions
  - For each triangle

```
For i=0 to 2
    Send vertex data
```

OpenGL Example: Viewing

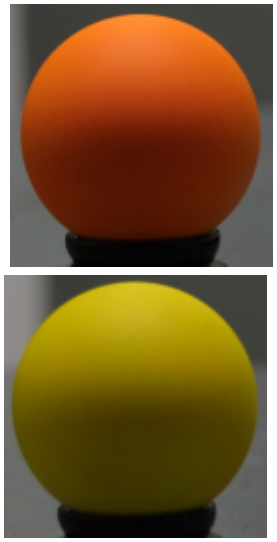
```
// Current matrix affects objects positions
glMatrixMode( GL_MODELVIEW );
// Initialize to the identity
glLoadIdentity();
// Position the camera at [0,0,5], looking
at // [0,0,0], with [0,1,0] as the up
direction.
gluLookAt(0.0, 0.0, 5.0,
          0.0, 0.0, 0.0,
          0.0, 1.0, 0.0);
// Rotate by -20 degrees about [0,1,0]
glRotated(-20.0, 0.0, 1.0, 0.0);

// Draw a teapot.
glutSolidTeapot(1.0);
```

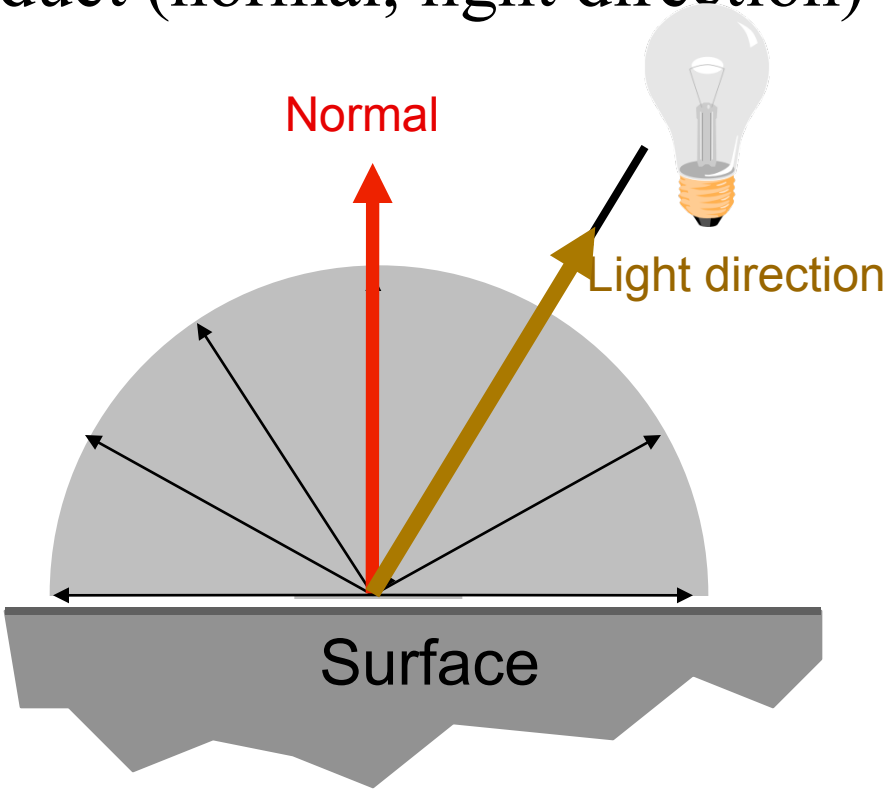
Vertex data

- What information do we need at each vertex?
  - Coordinates (3 floats)
  - Color (optional, 3 floats)
  - Normal information (optional, 3 floats)
  - Transparency (optional, 1 float)
  - More to come (texture information, shininess)

Why normals?

- To compute color as a function of light direction
- Simplest: Diffuse or Lambert model
  - Intensity = dot product (normal, light direction)





OpenGL Code

```
glBegin(GL_TRIANGLES); //what follows describes triangles
glColor3d (1,1,0); //red, green and blue components=>(yellow)
glNormal3d (0, 0, 1); //normal pointing up
glVertex3d (2,3,3); //3D position x, y, z
glColor3d (1,0,0);
glNormal3d (0, 0, 1);
glVertex3d (5,3,3);
glColor3d (1,0,1);
glNormal3d (0, 0, 1);
glVertex3d (3,6,3);
glEnd();
```

OpenGL high-level pseudocode

- Initialize
  - (get graphics context, etc.)
- For each frame
  - Manage UI
  - Set appropriate viewpoint
  - Set light source directions
  - For each triangle

```
For i=0 to 2
    Send vertex data
```

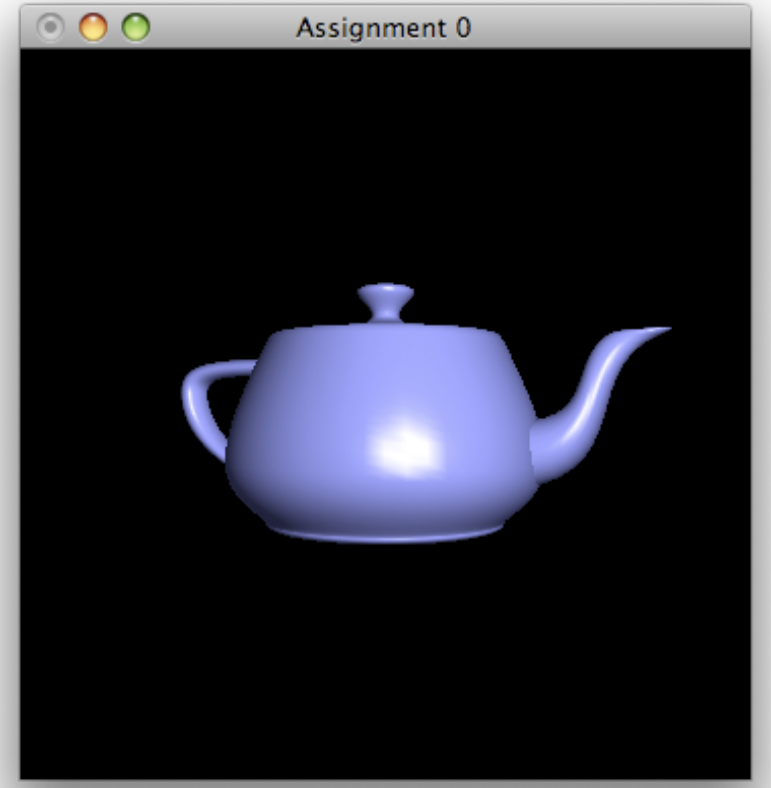
OpenGL is a state machine

- Each command changes the state
- For example, glColor3f changes the current color.
  - The color remains valid until we call glColorxx again
  - Use it before each vertex to get per-vertex color.
- Other state to manage lighting and other rendering aspects
- Can make it hard to debug
- (Note: This is conceptually simple, but not quite how you write efficient code these days.)

Assignment 0

- Read a file with triangle mesh data
  - Including mesh normals
- Display it using OpenGL
  - Colors, simple movement

- Due next Monday!



What is missing?

- Shadows
- Shininess
- Texture
- Etc.

- Be patient, you will have plenty enough

Linear Algebra is Everywhere

- Vertices are 3-vectors
- Normals are 3-vectors
  - Orthogonal to surface tangent plane
  - Cross product
- Colors are 3-vectors
- Diffuse shading is a dot product
- A non-bending object moving in a scene undergoes a rigid transformation
- Changing the viewpoint is a linear transformation of the scene coordinate
- Supplementary notes available on course website

After G&V

Crowd-driven Mid-scale Layout Design

Tian Feng[1] Lap-Fai Yu[2] Sai-Kit Yeung[1]
KangKang Yin[3] Kun Zhou[4]

[1]Singapore University of Technology and Design
[2]University of Massachusetts Boston
[3]National University of Singapore
[4]Zhejiang University

After G&V and CF

Optimal Design and Manufacture
of Active Rod Structures
with Spatially Variable Materials

Oliver Weeger Yue Sheng Benjamin Kang
Sai-Kit Yeung Martin L. Dunn

SINGAPORE UNIVERSITY OF
TECHNOLOGY AND DESIGN

Established in collaboration with MIT

What Makes Graphics Fun?

- Very interdisciplinary
  - Within CS: systems, compilers, languages, computer architecture, algorithms, numerical techniques
  - Math, physics, art, perception, architecture, manufacturing
- Helps you understand why the world looks the way it does
- You can "see" the result