

## Words of Wisdom



- Kevin Geiger, Walt Disney Feature Animation
- **What do you think is a current problem/issue facing the industry of computer graphics?**
  - "Lack of qualified talent. There are plenty of applicants out there who are software savvy, but relatively few who possess the aesthetic and conceptual training required to nail the shots. There's a misconception floating around that all you need to do is learn the major software packages, and you're set.

---

---

---

---

---

---

---

---

## Expanding a little



- Story → Visual Communication → Visual Effect → Software Tool
- No matter the tool,
- Being familiar with the common techniques used to create, compose, and animate renderings is essential.

---

---

---

---

---

---

---

---

## Visual Literacy



- Analyze, identify algorithms visually
- Advantages

---

---

---

---

---

---

---

---

## Visual Analysis



- Method of identifying graphics effects in images or animation by spotting specific visual cues.

---

---

---

---

---

---

---

---

## Why is it important?



- Communication of ideas hinges not only picture composition, but also on the effects chosen by the artists and programmers who create the picture.
- Story → Visual Communication → Visual Effect → Software Tool

---

---

---

---

---

---

---

---

## Common Misconceptions



- CG is about knowing how to use a bundle of graphic packages
- CG does not require programming, computer science does
- CG does not require mathematical foundations
- CG does not require knowing about drawing, composition, or editing

---

---

---

---

---

---

---

---

## The Truth is...

- In some instances, programming scripts are more suitable for the rendering needs
  - Maya - MEL
  - 3DSMax – MaxScript
  - Pixar - RenderMan



---

---

---

---

---

---

---

---

## The Truth is...

- Everything behind animation, visual effects, and modeling is math.
  - The algorithms used to generate visual effects are based on geometry, algebra, calculus, and in some cases, physics

---

---

---

---

---

---

---

---

## The Truth is..

- The composition of a scene requires artistic sensibility
  - Assembling models to convey a visual message
  - Lighting
  - Visual harmony



---

---

---

---

---

---

---

---

## The Good News...



- Software packages today have much of the math already implemented.
- However, that does not mean you should not know how these things are done.
- Software packages provide you with tools, not with ideas, knowledge or experience.

---

---

---

---

---

---

---

---

## In this class



- You will develop your visual literacy.
- You will generate renderings and possibly animations via scripting
- You will learn and apply some basic math used in computer graphics.

---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---



Pat Motoike  
DePaul University

---

---

---

---

---

---

---

---



Daiva Skuodyte  
DePaul University

---

---

---

---

---

---

---

---



Dan Mauer  
DePaul University

---

---

---

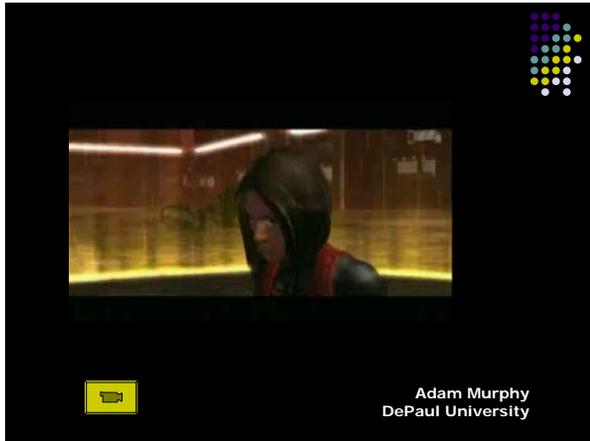
---

---

---

---

---



---

---

---

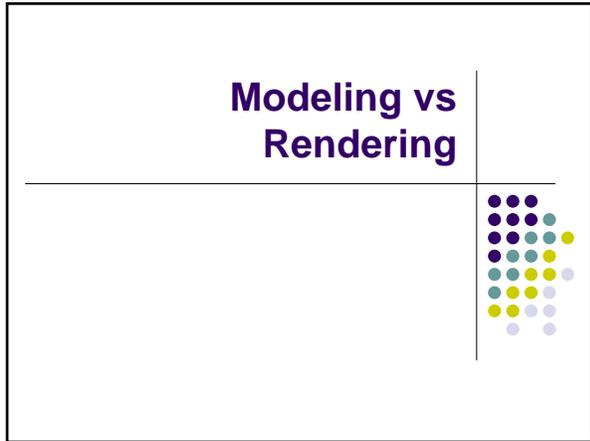
---

---

---

---

---



---

---

---

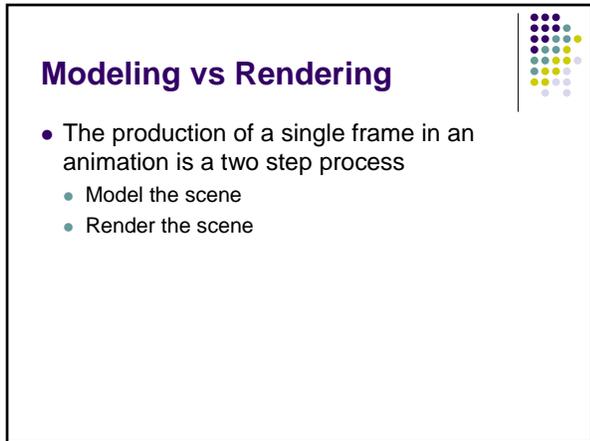
---

---

---

---

---



---

---

---

---

---

---

---

---

## Modeling



- Creating a representation of an object, scene or virtual world.

---

---

---

---

---

---

---

---

## Rendering



- Term comes from architecture
- Drawing an image of the model

---

---

---

---

---

---

---

---

## For every image ...



- Where do we place the observer of the scene?
- How much of the scene is seen?
- How much light?

---

---

---

---

---

---

---

---

## Viewing a Scene



- Where do we place the observer of the scene?
  - We need to place ourselves somewhere and look at some place on the scene
  - In CG we achieve this by creating a **Camera**

---

---

---

---

---

---

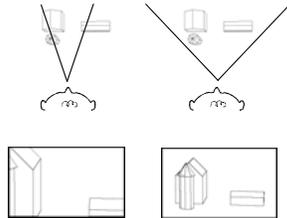
---

---

## Viewing a Scene



- How much of the scene is seen?
  - In CG this is called the **Field of View**



---

---

---

---

---

---

---

---

## Viewing a Scene



- Field of View
  - How much of the scene is seen



---

---

---

---

---

---

---

---

## How much light?



- How much would we see without light?
  - Sunlight
  - Artificial light
- We naturally need light in order to see
- In CG, a scene without light would look black

---

---

---

---

---

---

---

---

## Modeling + Rendering =



A high quality image



---

---

---

---

---

---

---

---

## Visual Effects



- Visible Surfaces
- Shadow
- Refraction
- Reflection
- Transparency
- Lighting

---

---

---

---

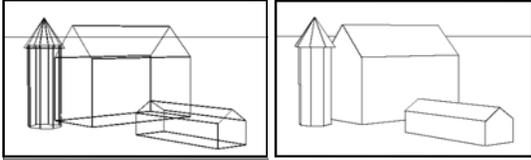
---

---

---

---

### Visibility of background



---

---

---

---

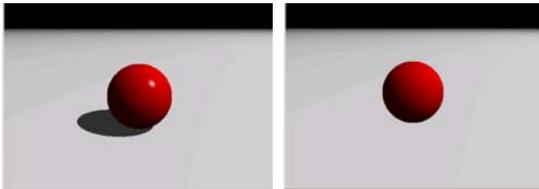
---

---

---

---

### Shadows and Shading



---

---

---

---

---

---

---

---

### Refraction



---

---

---

---

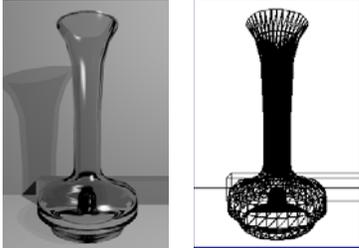
---

---

---

---

## Transparency



---

---

---

---

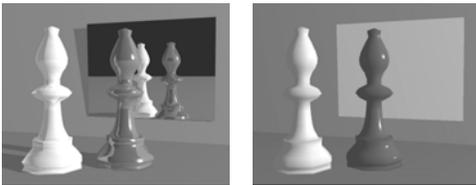
---

---

---

---

## Reflection



---

---

---

---

---

---

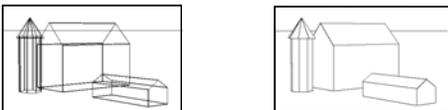
---

---

## Surface Algorithms



What surfaces are visible?



---

---

---

---

---

---

---

---

## Four main surface algorithms



- Wireframe
- Hidden Line
- Raytracing
- Z-buffer

---

---

---

---

---

---

---

---

- Outlined polygons
- Horizons, background objects completely visible.



**Wireframe**



---

---

---

---

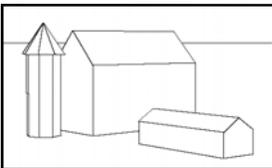
---

---

---

---

- Outlined polygons
- Occluded objects are invisible



**Hidden Line**



---

---

---

---

---

---

---

---



- Filled-in polygons
- no refraction, reflection or shadow



### Z-buffer

---

---

---

---

---

---

---

---



- Solid polygons
- Refraction, Reflection, Shadows



### Raytracing

---

---

---

---

---

---

---

---



### Visual Analysis

- Visual Cues
- Cues identify algorithms.

---

---

---

---

---

---

---

---

## Visual Analysis



- Find cues first.
- Then suggest an algorithm.

---

---

---

---

---

---

---

---

## Surface Algorithms



- Polygon appearance
- Visibility of far sides, horizons
- Refraction, reflection, shadow

---

---

---

---

---

---

---

---

## Polygon appearance



---

---

---

---

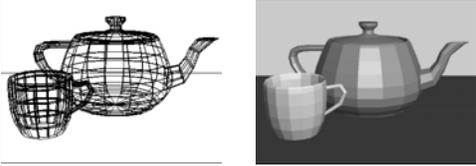
---

---

---

---

### Polygon appearance



---

---

---

---

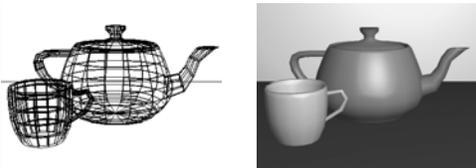
---

---

---

---

### Polygon appearance



---

---

---

---

---

---

---

---

### Refraction



---

---

---

---

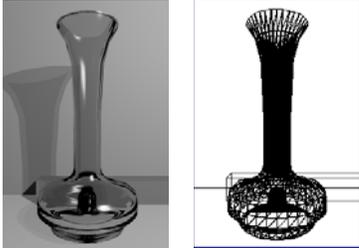
---

---

---

---

## Transparency



---

---

---

---

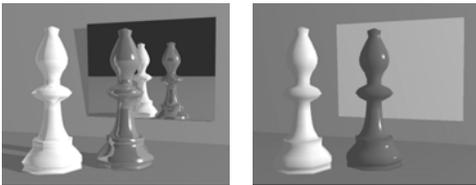
---

---

---

---

## Reflection



---

---

---

---

---

---

---

---

## Examples



---

---

---

---

---

---

---

---

### Surface Algorithm?



---

---

---

---

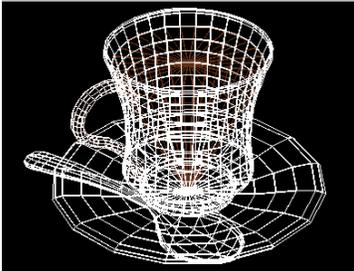
---

---

---

---

### Surface Algorithm?



---

---

---

---

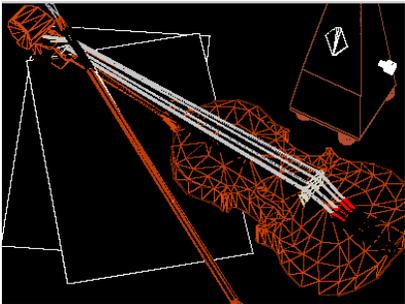
---

---

---

---

### Surface Algorithm?



---

---

---

---

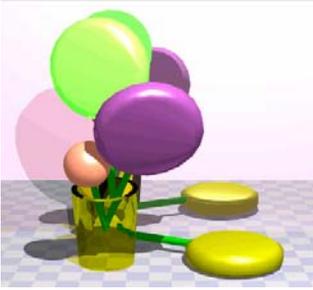
---

---

---

---

## Surface Algorithm?



---

---

---

---

---

---

---

---

## Acquiring visual literacy

- Practice :-)
- Render, render, and render....
- Tutorials
- The problem is that rendering is expensive.



---

---

---

---

---

---

---

---

## TERA

A tool to enhance visual literacy



---

---

---

---

---

---

---

---

## TERA



- Tool for Exploring Rendering Algorithms
- Simple, easy to install and easy to use
- Learn, test visual analysis
- 500,000 image combinations
- It comes with the textbook

---

---

---

---

---

---

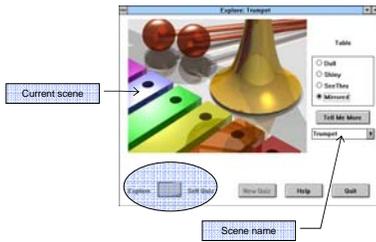
---

---

## TERA – Main Interface



- Two modes
  - Explore
  - Self quiz



---

---

---

---

---

---

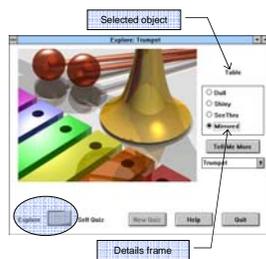
---

---

## TERA – Explore Mode



- In some scenes, you can click on an object in the picture and a rendering technique used to draw it.
- In the figure, the "Table" object was rendered as a "Mirrored" surface



---

---

---

---

---

---

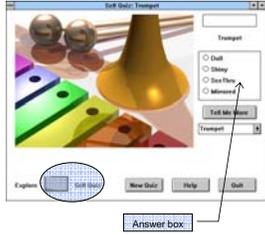
---

---

## TERA – Self Quiz Mode



- Test your visual literacy in this mode
- The “New Quiz” button generates a new quiz for you.
- Select an object and an answer in the answer box for that object.



---

---

---

---

---

---

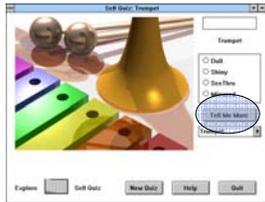
---

---

## TERA



- “Tell Me More” button
  - Provides additional cues about the selected object or scene



---

---

---

---

---

---

---

---

## TERA



- Contains practice scenarios for almost all the chapters of the textbook so users can see some of the different topics covered in action
- Main thing: Everything is pre-rendered so there is no waiting, everything is real-time

---

---

---

---

---

---

---

---

# Rendering Concepts

A Brief Discussion



---

---

---

---

---

---

---

---

## Coordinate systems

- Left handed
- Right handed



---

---

---

---

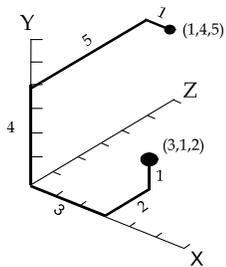
---

---

---

---

## A Point / Vertex



---

---

---

---

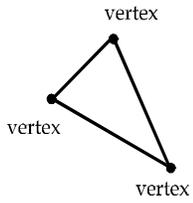
---

---

---

---

## A Polygon



---

---

---

---

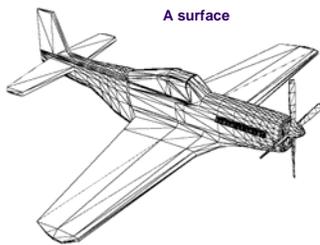
---

---

---

---

## Many Polygons =



---

---

---

---

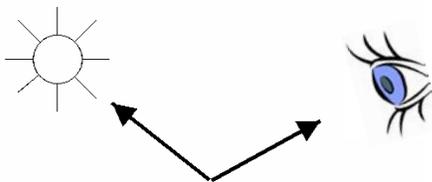
---

---

---

---

## Vector



---

---

---

---

---

---

---

---

# POV-Ray

Persistence of Vision Ray tracer



---

---

---

---

---

---

---

---

# POV-Ray

- Creates photo-realistic images using ray-tracing (a rendering technique).
- Reads a text file describing the objects, lighting and camera in a scene and generates an image from the view point of the camera
- It is NOT modeling software



---

---

---

---

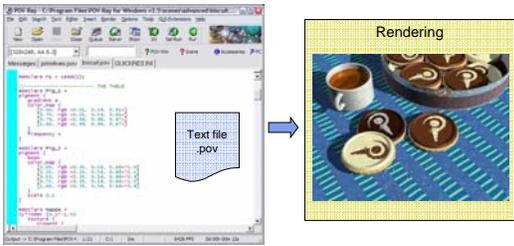
---

---

---

---

# POV-Ray



Text file .pov → Rendering



---

---

---

---

---

---

---

---

## POV-Ray



- You model the scene via scripting
- You can bring models created in other software packages (although we aren't interested in that here, for now)

---

---

---

---

---

---

---

---

## POV-Ray



- Free download
  - <http://www.povray.org>
- 8 Mb aprox.
- Online documentation

---

---

---

---

---

---

---

---

## POV-Ray

Some Samples



---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---



---

---

---

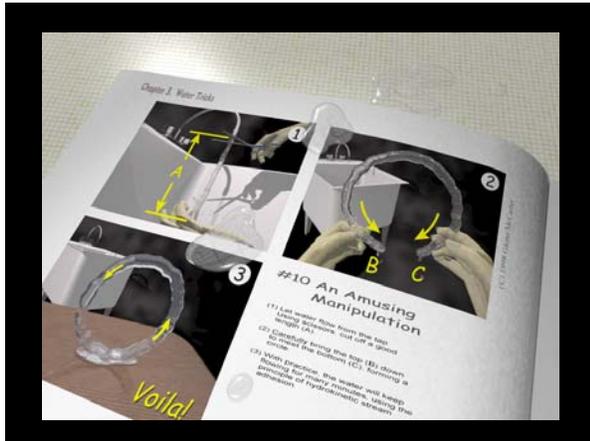
---

---

---

---

---




---

---

---

---

---

---

---

---

---

---

---

---




---

---

---

---

---

---

---

---

---

---

---

---

**POV-Ray - Running**

- Four main components
  - .ini files
  - .pov files
  - .inc files
  - Image files

---

---

---

---

---

---

---

---

---

---

---

---

# POV-Ray

## A First Scene



---

---

---

---

---

---

---

---

## Simple Scene

- Create an object.
- Create a camera.
- Create a light source.



---

---

---

---

---

---

---

---

## Example

```
#include "colors.inc"

camera {
  direction <0, 0, 1>
  location <0, 3, -6>
  look_at <0, 1, 0>
  up <0, 1, 0>
  right <4/3, 0, 0>
}

sphere { <0, 1, 0>, 1
  pigment {red}
  finish {
    ambient 0.15
    diffuse 0.75
    phong 1
    phong_size 100
  }
}

plane { <0,1,0>, 0
  pigment {White}
  finish {ambient 0.2
    diffuse 0.8}
}

light_source {<100, 120, -40>
  color White}
```



---

---

---

---

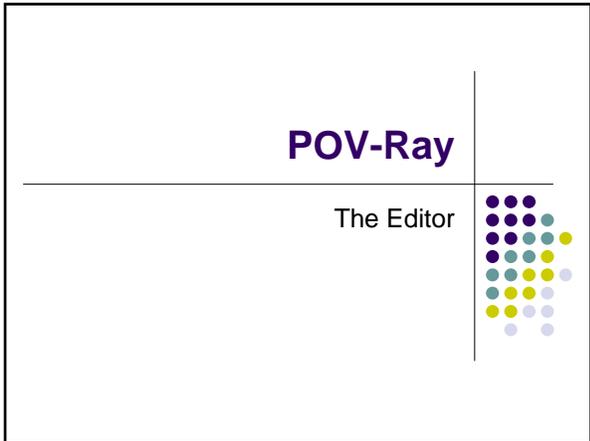
---

---

---

---






---

---

---

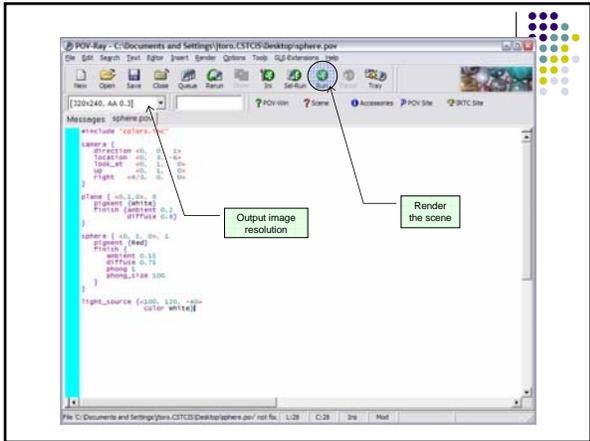
---

---

---

---

---




---

---

---

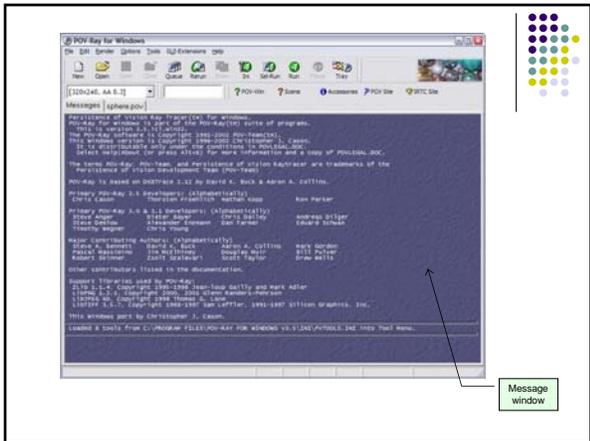
---

---

---

---

---




---

---

---

---

---

---

---

---