

Test 1

- Average 84
- Median 87
- High 97
- Low Below 50

Question 1

- They do cause a slight speedup for a simple reason that there is less overhead for calls to the routines. By having a single call as opposed to many, the overhead for passing arguments and returning to the main program is reduced.
- Large Data Set Advantages
 - Storage
 - Editing
 - Program maintenance/size/style

Question 2

- Most people got this one and there were a variety of approaches:
 - 4 triangles and a quad
 - A triangle fan and a quad
 - 1 triangle with 4 rotations and a quad.
- Many alternatives mentioned a vector approach but none of the actual code did one. I'd like to mention one – A triangle strip with all triangles. The base being split into two triangles.

Problem 3

- A glFrustum is more flexible than a gluPerspective (gluPerspective is implemented using glFrustum). A Frustum may have a asymmetric orientation where a perspective is always symmetric around the viewpoint.

Problem 4

- Most of the class were able to come up with the transformations. The problem was the order.
- Ry(45) Rx(90)S(1,3,1) P

Problem 5

- Shading differences. Many of the problems were not in what the 3 types did
 - Flat – constant intensity for all points in a polygon.
 - Gouraud – interpolated intensity from vertices of a polygon.
 - Phong – normals interpolated and point's intensity independently determined.
- The Problem was describing differences.
 - Highlights and transition across polygon are the keys!

Problem 6

- The height field was put in to make you think. There were a number of interesting approaches as to how to connect the varying heights together. Sorting points, determining neighboring points, interpolating heights and so forth.

Problem 7

- Display lists can speed it up when they save calculation time by the system. Images and transformations are good targets.
- Other Reasons
 - Code structure
 - Ease of modifying groups of attributes and objects.

Project 3

- We will add a couple of moving characters through the maze.
- The character should try to "mug" the person walking through the maze.
- Will need to use a timer so that the character can move even if the player doesn't.
- When the character "mugs" the player they should take one painting (random selection) from those the player has.

Modeling the Character

We will use a simple solid to model the character and try to put a reasonable texture map on it. Plan on using a cylinder to represent the character for the time being.

I will prepare a more detailed write-up and post it to the web.

Due Date: November 16, 2005

Illumination Models and Ray Tracing

Computer Graphics I
November 02, 2005

Light Source Types

- Point Sources
 - simplest version
 - emits light in all directions
 - Reasonable approximation for small (relatively) lights in a scene.
- Infinitely Distant or Directional Light Sources
 - The Sun
 - Rays are approximately parallel
- Extended Light Sources
 - A source that is more or less a surface.
 - The Warn Model is one way of producing such effects using a series of point sources.

Attenuation

- Radial intensity attenuation
 - light normally attenuated by the factor of $1/d^2$
 - Not normally used – too much variation to produce realistic pictures
- Used to make a spot light – Angular Intensity Attenuation

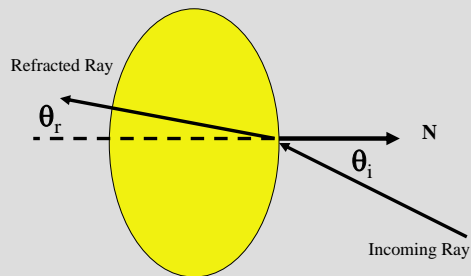
$$f_{angatten}(\phi) = \cos^{a_l}(\phi)$$
$$(V_{object} \bullet V_{light})^{a_l}$$

Transparent and Translucent Materials

- Transparent Surfaces – can have refraction and reflection
- Angle of refraction (Snell's law) =
 - θ_i -- angle of incidence
 - n_i is the index of refraction for the incident material
 - n_r is the index of refraction for the refracting material

$$\sin(\theta_r) = \frac{n_i}{n_r} \sin(\theta_i)$$

Refraction



Some Typical Indexes of Refraction

- Crown Glass 1.52
- Flint Glass 1.61
- Quartz 1.54
- Water 1.33

Gamma Correction

- When you present graphics on a monitor the brightness variations are nonlinear but the models produce a linear variation
- 64, 64, 64 would not appear half as bright as 128, 128, 128.
- A system may use a video lookup table to adjust the values for linearity.

$$V = \left(\frac{I}{a} \right)^{\frac{1}{\gamma}}$$

Ray Tracing

- Pixel ray – generated going from projection reference point (eye point) through an x-y pixel position.
- If the ray intersects an object it may generate secondary rays (reflection and refraction)
- New rays are computed and intersections are found.

Ray Tracing Tree (continued)

- This forms a tree.
- Tree branches terminate when
 - the ray intersects no surfaces
 - The ray intersects a light source that is not a reflecting surface
 - The tree has been generated to its maximum allowable depth

Ray – Surface Calculations

- We will start by determining the primary vector. We want this to be a unit vector to simplify later calculations.

$$u = \frac{P_{pixel} - P_{prp}}{|P_{pixel} - P_{prp}|}$$

Ray-Sphere Intersection Test

- P_c – The center of the sphere
- P – a point on the surface
- r – radius of the sphere
- P_0 – The origin of the ray
- u – the direction for the ray (unit vector)

The Equations

$$|P - P_c|^2 - r^2 = 0$$

$$|P_0 - su - P_c|^2 - r^2 = 0$$

$$s^2 - 2(u \bullet \Delta P)s + (|\Delta P|^2 - r^2) = 0$$

$$s = u \bullet \Delta P \pm \sqrt{(u \bullet \Delta P)^2 - |\Delta P|^2 + r^2}$$

Other Formulations

- Other formulations for this exist. The one that I like is the geometric solution which eliminates trivial misses sooner.

Ray Polygon Testing

Assuming N is the normal of the plane the polygon lies in and we have the equation for the plane $Ax+By+Cz+D$

Solve for P :

$$N \bullet P = -D$$

$$N \bullet (P_0 + su) = -D$$

$$s = -\frac{D + N \bullet P_0}{N \bullet u}$$

Ray Polygon Testing (cont)

- This gives a position on the infinite plane the polygon lies in.
- Still need to perform an inside-outside test

Speedup

- Speedup techniques tend to fall into a couple of categories
 - Reducing the Object-Intersection Calculations
 - Reducing the number of objects which we need to look at (Space-subdivision)

Other topics

- Antialiasing
- Distributed Ray Tracing

Test 2

- Test could be November 16th or 21st
- Should Cover (Green Book Sections):
 - Discrete Methods
 - Chapter 3
 - Chapter 4
 - Visible Surface Detection Methods
 - Chapter 9
 - Curves
 - Chapter 8
 - Input Methods
 - Chapter 11

Other Details

- Open Book and Notes again
- 5-6 questions
- Some Code
- While emphasis is not on earlier material it is still fair game as part of a problem.

Radiosity

- Uses the physical laws governing radiant energy transfers to model a scene.
- Also give a relatively detailed model

Radiant Energy Terms

- Energy in each Photon: $E_{\text{photon},f} = hf$
- Energy for all Photons $E_f = \sum_{\text{all_photons}} hf$
- Split into Frequencies $E = \sum_f \sum_{\text{all_photons}} hf$
- Radiant Flux $\Phi = \frac{dE}{dt}$
- Radiosity $B = \frac{d\Phi}{dA}$

Basic Model

- Need to compute the radiant energy interactions between all surfaces $B_k = E_k + \rho_k H_k$
 $H_k = \sum_j B_j F_{jk}$
- Simplified normally to small, opaque, ideal diffuse reflectors.
 $B_k = E_k + \rho_k \sum_j B_j F_{jk}$
 $(1 - \rho_k F_{kk}) B_k - \rho_k \sum_{j \neq k} B_j F_{jk} = E_k$

Radiosity (cont)

- To compute F, the form factor, we need to do some numerical integration
- Need to solve a set of n equations
- Time intensive – the text looks at one option to speed up the computation.

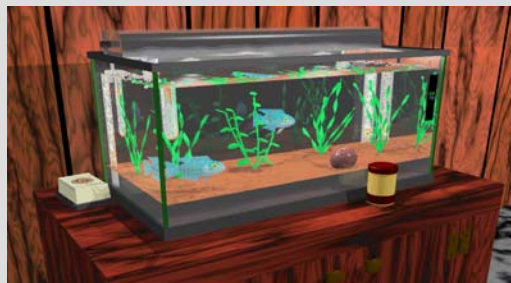
Radiosity Example



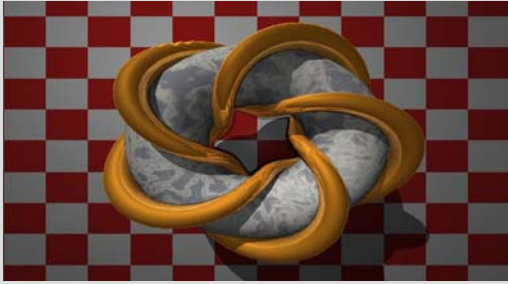
Radiosity Example



Raytracing Example



Raytraced Example



Another Raytraced Example



POVRAY

- Free ray tracer
- Relatively complete in terms of options
- Many tools available for use with it

Radiance

- Developed at Lawrence Livermore Labs, Berkeley
- High level of realism
- Not under a high level of development (POV ray has a much larger user group and more activity).

Open Source Radiosity

- RadiosGL
- POV-Ray (Also does Ray Tracing).