Administrative

- Homework 3 due on Nov 12
- We have been / are doing stuff from chapter 14. It is worth a skim.

Device independent colour imaging

- Problem: ensure that colours on a display, printer, etc. give the same experience that a viewer would have seeing relevant light spectra
- Difficulty: limited gamuts of most output devices
- Strategy: exploit a model of human experience
  - Simple model: The CIE XYZ matching paradigm
  - Being implemented in “Color Management Systems”
  - These try to relieve the user of the different color capabilities of devices
  - Complicated because every device needs to register properly with the CMS
- Defficiencies--as we have seen, the CIE systems does not count for spatial effects, illumination environments, etc., and these are important
- Some progress is being made but the models tend to be complicated

Shadows (without ray-tracing)

- Bonus topic--details won’t be covered in notes, but it is a good review exercise in to think about how you would design such an algorthm
- Book covers a few methods in chapter 14.

Computing shading values for coloured surfaces

- Simplest:
  - use appropriate shading model in 3 channels, instead of one
  - implies red albedo, green albedo, blue albedo, etc.
  - works because the shading model is independent of wavelength.
  - Can lead to somewhat inaccurate colour reproduction in some cases - particularly coloured light on coloured surfaces
- Better
  - use appropriate shading model at many different wavelength samples - 7 is usually enough
  - estimate receptor response in eye using sum over wavelength
  - set up pixel value to generate that receptor response

Recursive ray tracing

- Pixel brightness =
  radiance along ray to pinhole =
  ρ_d(diffuse) +
  ρ_s(specular) +
  ρ_t(transmitted)
- Diffuse component:
  - from sources alone (local shading model), usual case
  - from global illumination model
  - typically Lambertian, but better models do exist
- Reflected component is due to radiance along ray from intersection along specular direction
- Specular component (optional or sometines considered another name for reflected componant)
  - **could** cast rays to sources which are just off the reflected angle, and use Phong shading or phyisics based model. We can't cast all the rays we like, so we have to cast them to the bright stuff (usually lights).
  - **another** way is covered later
- Transmitted component is due to radiance along ray from intersection along transmitted direction

Recursive ray tracing rendering algorithm

- Cast ray from pinhole (projection center) through pixel, determine nearest intersection
- Compute components by casting rays
  - to sources = shadow ray
  - along specular dir = specular ray
  - along transmitted dir = refracted ray
- Each of the components has a weight
  - The main processes do not affect color much but a vector of weights for may be more convenient, or accurate depending on a color model
- Determine component and form sum, but ...
- To determine some of the components, the ray tracer must be called **recursively**.

Recursive ray tracing rendering (cont)

- Reflections (at least need to be attenuated)--no perfect reflectors
- We must stop the recurson at some point
  - when contributions are too small
    - need to track the cumulative effect
  - typically also limit the depth explicitly

Mechanics

- Primary issue is intersection computations.
  - E.g. sphere, triangle.
- Polygon (see book page 461-2, handout for a better way)
- Sphere (see book page 460, or handout)