Light Transport

CS224 Topic #2

John Hughes

Warmup Math

- Recall
  - *Angle* = ‘subset of the circle’; units: radians
  - *Solid Angle* = ‘subset of the sphere’; units: steradians
- *Light* is described by the *radiance* function $L(x, \omega)$ with units of Watt/m^2 sr (radiance)
- Radiance is *constant along rays in empty space!*

Raytracer

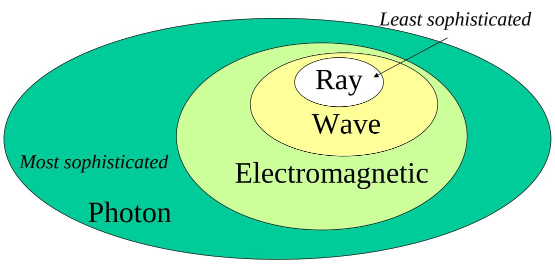
- When you make a raytracer, what are the units of the values you store at each sample point in your computation?

Light Transport

- Goal: Create realistic images
- Observation: Images are produced by light on film (or retina)
- Idea: Simulate the physical interaction of light and surfaces

How can we simulate light?

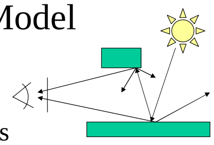
Many Optics Models



WARNING

We are now going to work with the **Ray Optics model**. Like any model, it simulates but is not actually reality. We us simplifications wherever their benefits outweigh the error they introduce. For the remainder of the lecture series, treat every statement of fact as a rule from our model and not a statement about the true physical properties of light.

Ray Optics Model



- Assume light travels in rays
- Assume geometric reflections
- Good for most human-observable effects
  - e.g. reflection, refraction
- Does not capture the full richness of light
  - e.g. interference on the wings of a butterfly

Other Assumptions

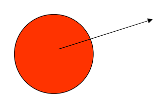
1. Only consider transport paths in the steady state
2. Reflection angle is independent of wavelength (color)
3. Light does not interact with air

Transport

What are we transporting along the light rays?

Photons

- A light ray is a stream of *photons*
- Photons are little balls of energy
- Always in motion
- Properties:
  - wavelength (λ) scalar meters
  - position (x) 3D vector meters
  - Direction of motion (ω) 3D unitless vector



Your Eye: A Photon Detector

- A group of N photons hit your retina
- All have the same wavelength λ
  - Assume 400nm < λ < 700nm
  - 1nm = 10^{-9}m
- What do you see?

Perception of Photons

N as brightness

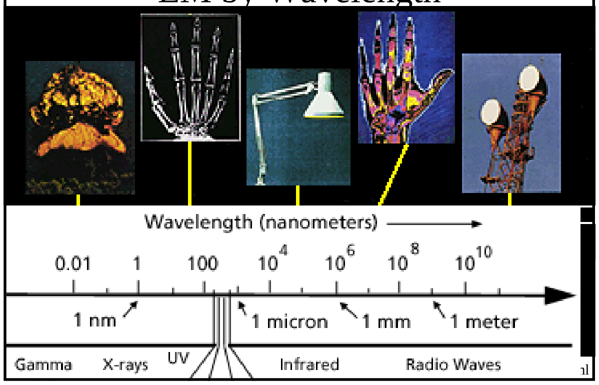
$b = \log N$

λ as color

λ_{red} = 700nm, …, λ_{blue} = 400nm

This is only a rough model to help you build intuition for photons. Lots of factors affect the actual perception of color and brightness (both neural and physiological).

EM by Wavelength



Simulating light transport

- We *should* consider transport at every single wavelength λ, but that's impractical
- Transport at exactly λ is always zero…but we can work with wavelengths λ … λ +d λ, compute transport, and divide by d λ, to get a transport density.
- In practice: divide the visible spectrum into a few bands; take one representative frequency in each

The "selected bands" assumption

- If a surface reflects lots of light at wavelength λ, it reflects a good deal at wavelengths near λ
- i.e., reflectivity is a slowly-changing function of λ
- …so we can sample at a few points and lose not too much information

Which bands/sample wavelengths?

- We choose three bands, with central wavelengths corresponding to monospectral lights that we perceive as red, green, and blue.
- To capture perceptual phenomenon of color as three-dimensional, need at least 3
- More than 3 generally not important
- For some scenes (sodium lamps, etc.) it's critical.

"Colored" Photons

- Goal: Track fewer photons
- Observation: Direction of reflection/refraction is often independent of wavelength
  - Not true for a glass prism or two-tone paint
- Idea: Encode a group of photons as a single "colored" photon

Colored Photon Representation

x Position (m, scalar)

ω Direction (unitless, 3D vector)

c Color (unitless, 3D vector)

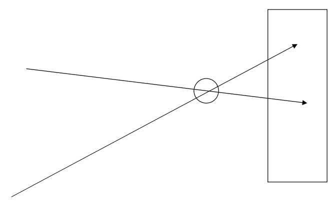
Interpretation:

a group of c_R photons of wavelength λ_R, c_G photons of wavelength λ_G, and c_B photons of wavelength λ_B that share position x and direction ω.

Photon Emitters

- A light source emits photons
- The photons are all emitted at the three “main” wavelengths.
- The emission rate is constant, and has a known dependence on direction and location.

The Image Plane



Factoring Illumination

- Not all paths are equally interesting
- What are the paths that create interesting effects?

Light Path Notation

- Goal: concise notation for discussing transport paths
- Idea: There are 4 kinds of interesting vertices
- L – Light
- E – Eye
- D – Diffuse bounce
- S – Specular bounce

”?

Direct Illumination

- L(S|D)E
- Also accounts for direct shadows

Caustics

- $LS^{+}DE$

Color Bleeding

- LD^+E
- a.k.a Diffuse Interreflection

Mirror Reflection and Refraction

- $L(D|S)^+SE$

How to account for image created by each path-type

- Those easier to trace forward from light
  - Caustics
  - Color Bleeding
- Those easier to trace backward from eye
  - Direct Illumination
  - Reflection and Refraction
- Hard to trace
  - Caustics from diffuse interreflection

The Rendering Equation

$$L(x,\omega) = L_e(x,\omega) + L_r(x,\omega)$$

- Outgoing = Emitted + Reflected
- At point x, in direction ω
- Radiance units are $W/(m^2 sr)$

Expand the reflected term

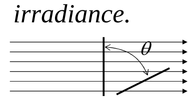
Reflected Radiance

$$L_r(x,\omega) = \int_{\Omega_i} f_r(x,\omega',\omega) L_i(x,\omega') \, |(\omega' \cdot n)| \, d\omega'$$

- f_r is the Bidirectional Reflectance Distribution Function (BRDF)
- L_i is the incident radiance
- n is the surface normal at x
- Integrate ω' over the hemisphere (Ω_i) of directions with $\omega' \cdot n < 0$ ('incoming light')
- Whole sphere for tranmissive materials…

Irradiance

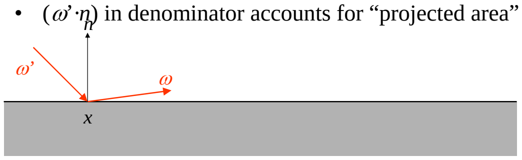
- Radiance is power per area per solid angle arriving along a direction ω hitting a surface of area A that's perpendicular to ω.
- How much power hits a surface that's not perpendicular to ω? cos θ as much.That's *irradiance.*



Bidirectional Reflectance Distribution Function

$$f_r(x,\omega',\omega)=\frac{L_r(x,\omega)}{L_i(x,\omega')(\omega'\cdot n)}$$

- Denominator is "amount of light, from direction ω, hitting a unit area of the surface.
- $(\omega'\cdot n)$ in denominator accounts for "projected area"



Substitute Back

$$L(x,\omega)=L_e(x,\omega)+\int_\Omega L(x,\omega')f_r(x,\omega',\omega)(\omega'\cdot n)d\omega'$$

- We can evaluate:
  - $L_e(x,\omega)$
  - $f_r(x,\omega',\omega)$
  - $(\omega'\cdot n)$

Integral is over a sphere---how to evaluate?

$$L(x,\omega)=L_e(x,\omega)+\int_\Omega L(x,\omega')f_r(x,\omega',\omega)(\omega'\cdot n)d\omega'$$

- Use constancy on rays!
- Look in direction ω until you see a surface point
- Rewrite to integrate over surface locations S instead of sphere Ω

- Do change of variables in 2D, show how it changes

Integrate Over Surfaces

$$d\omega'(x)=\frac{((x'\to x)\cdot n)dA'}{\|x'-x\|^2}$$

$$L_o(x,\omega)=L_e(x,\omega)+\int_S L_i(x,x'\to x)f_r(x,x'\to x,\omega)(\omega\cdot n')V(x,x')\left[\frac{((x'\to x)\cdot n)dA}{\|x'-x\|^2}\right]$$

- Let $x'\to x$ be the unit vector from x' to x
- Let visibility $V(x,x')$ be 1 if there is unobstructed line of sight between x and x'

"Geometry" Term

$$L_o(x,\omega)=L_e(x,\omega)+\int_S L_i(x,x'\to x)f_r(x,x'\to x,\omega)(\omega\cdot n')V(x,x')\left[\frac{(\omega'\cdot n)dA'}{\|x'-x\|^2}\right]$$

- Collect the projected area terms into a so-called "geometry" term, G

$$G(x,x')=\frac{(\omega\cdot n')((x'\to x)\cdot n)}{\|x'-x\|^2}$$

$$L_o(x,\omega)=L_e(x,\omega)+\int_S f_r(x,x'\to x,\omega)L_i(x,x'\to x)V(x,x')G(x,x')dA'$$

Plan of action

$$L_o(x,\omega) = L_e(x,\omega) + \int_S f_r(x, x' \to x, \omega) L_i(x, x' \to x) V(x,x') G(x,x') dA'$$

- Find a way to approximate the Li terms
- Find a way to approximate the integral
- This will produce Lo result.
- Do this only for points *x* and directions ω that contribute to the image you're making

Practical Constraints

- Our goal is often to produce the best image within a limited amount of time
- This means we can't perfectly simulate LT
- Variance Errors
  - Look like noise
- Bias (Mean) Errors
  - Physically wrong (e.g. too dark in certain places)

Noisy Estimators

- Say the *true* value is $L_{rT}(x, \omega)$
- Imagine some method that computes

$$L_{rN}(x,\omega) = L_{rT}(x,\omega) + \frac{1}{N}\sum_{i=1}^{N} rand()$$

- Limit as N goes to infinity is correct
- For any finite N, the result is noisy (has variance)

Biased Sampling Estimators

- Say the *true* value is $L_{rT}(x, \omega)$
- Imagine some method that computes

$L_{rB}(x, \omega) = L_{rT}(x, \omega) - K(x, \omega)/M$

- Limit as M goes to infinity is correct
- If K is everywhere positive, then for any finite M, our solution is too small

Sources of Bias

- May result from limiting assumptions
  - Radiosity assumes perfectly diffuse surfaces
- May result from biased sampling
  - Photon mapping emphasizes LS*DE and LDE paths

Joton

- Representation of a probabilistic photon group – a bunch of photons that we may want to sample.
- J = J(x, ω_I, Φ), where Φ is power arriving at the surface, and ω_I is the direction of incident light, x = pt on surface. Units of J = radiance.
- Photon map = record of lots of J-values.

Estimating light from a surface to the eye

- J values near the relevant surface point
- Reflectance function (fR) on the surface
- Combine by summation (low budget integration) to estimate L_R

Russian Roulette

- Suppose 100 Jotons of power 1 hit a surface that reflects diffusely with reflectance k.
- Naïve sim: 100 Jotons with power k leave surface
- Clever hack: (100 k) photons with power 1 leave surface.

Why?

- Fewer jotons (‘cuz k < 1)
- “Weak” jotons disappear and we don’t waste computation on them
- Photon map will only store photons with power ~ 1, so all contribute equally to estimate of integral, so variance is reduced.

How does Photon Mapping work?

- Reflect jotons just like photons…but instead of a fraction of incoming power, reflect with a *probability* proportional to reflectance.
- If not reflected, it gets dropped from simulation.
- $P(\text{bounce } A) = L_R(x, \omega_O) / LI(x, \omega_I)$
- (for diffuse surface, this is just diffuse reflectivity!)

Program Structure

- Psuedo-code for the Photon Mapping algorithm:

Forward Trace Caustic (Specular Interreflection) Paths into Caustic Photon Map

Forward Trace Diffuse Interreflection Paths into Diffuse Photon Map

Balance Caustic and Diffuse Trees

Backward Trace Photons

- Illumination = Caustic + Diffuse + direct illumination

Caustic tracing

```
repeat numCaustics times
  J := random photon from random light
  absorbed = false
   do
     S = first intersection between J and scene
     if (r = random(0,1)) < P(diffuse)
          if "LS+" path then write J to caustics map
          absorbed := true
          else if r < P(diffuse) + P(specular)
               J := mirror J about normal
               scale Jpower by specular color
```

```
        else if r < P(diffuse) + P(specular) + P(transmit)
                J : = refract J
                if total internal refraction then
                        absorbed = true
                scale Jpower by transmission color

        else
                absorbed := true
while not absorbed
```

Initial joton power

- For each point from which jotons are emitted:

Starting power = totalEmitterPower/numCaustics

Dealing with color:
(brdf.emissive/brdf.emissive.sum()) * (totalEmitterPower / numCaustics)

TotalEmitterPower = Sum(tri.triangle.area() * emissive.sum())

Summed over all emitters.

Diffuse tracing

```
repeat numDiffuse times
    J : = random photon from random light
    absorbed := false
    while not absorbed
        S = first intersection between J and scene
        r := random(0, 1)
        if r < P(diffuse)
            if (not "LS*D" path)
                write J to diffuse photon map
                scale Jpower by diffuse color
                J := random hemisphere direction
            else if r < P(diffuse) + P(specular)
                J := reflect J about normal
                scale Jpower by specular color
            else if r < P(diffuse) + P(specular) + P(transmission)
                J := refract J
                if total internal refraction then absorbed = true
                scale Jpower by transmission color
            else
                absorbed := true
```

Backward Tracing

```
for each pixel(x, y)
        R := ray from eye through (x, y)
        S := get first intersection(R, Scene)
        image(x,y) :=
                direct illumination at S
                from all lights (with shadowing)
        + caustic radiance estimate
                + diffuse radiance estimate
```

Direct Illumination

Direct Illumination (*x*, *N*)
C := 0
for each light L with normal N_L, radiosity B
for count := 1 … numShadowRays
    x_L := random point on L
    $\omega_L := (x_L - x) / \| xL - x \|$
    r := $\| x_L - x \|$
    if visible(x, x_L)
        C := C + $\max(N \cdot \omega_L, 0) * k_d *$
                $\max(-N_L \cdot \omega_L, 0) * B(x_L) / (\pi * r^2)$
        C := C * A_L / numShadowRays
return C

Radiance Estimate(x, N)

(used for both the diffuse and caustic maps)

C := 0

For each photon J in photon map within radius r of x

C := C + $\max(N \cdot -\omega_J, 0) * k_d * L_J$

return C / (πr^2)

Fin