TIEA311
Tietokonegrafiikan perusteet
kevät 2017

("Principles of Computer Graphics" – Spring 2017)

TIEA311 Tietokonegrafiikan perusteet – kevät 2017
(“Principles of Computer Graphics” – Spring 2017)

Adapted from: *Wojciech Matusik*, and *Frédo Durand*: 6.837 Computer Graphics. Fall 2012. Massachusetts Institute of Technology: MIT OpenCourseWare, https://ocw.mit.edu/.

Frontpage of the local course version, held during Spring 2017 at the Faculty of Information technology, University of Jyväskylä:
`http://users.jyu.fi/~nieminen/tgp17/`

TIEA311 - Today in Jyväskylä

This is a fine week. This is the week of Instanssi 2017 (`http://www.instanssi.org`).

I'll "do as I preach". I have 9 more hours (only!) left of my example project: `https://yousource.it.jyu.fi/tiea311-kurssimateriaalikehitys/tiea311-kurssimateriaali-avoin/trees/master/instanssi17_4k_intro_webgl`

The file `project_memo.txt` is a full dump of my brain activity (as interpreted by the cerebral cortex) during the process of learning a new language, a new library, and some new tools.

Bottom line: Don't do as I do – but you may want to pay some attention to what I think you'd better do. . . We all (talking about the Faculty staff now) want you to become an IT pro, and we sort of know what it usually requires!

TIEA311 - Today in Jyväskylä

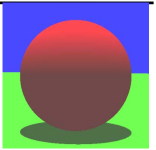
Our one-period course ends on Friday next week! After that, submissions can still be done, but **instruction from staff will not be available** (we'll be reassigned to new courses). Please, use the remaining time efficiently! Let's look at the plan:

- Ray Casting: cover fully. (DONE: necessary ideas covered on lecture; practicals in the final Assignments! One week to go!)
- Ray Tracing: basic idea, skip details $\rightarrow$ possible to continue as a "hobby project"; teachers of "TIEA306 Ohjelmointityö" may be contacted regarding credit for (any) hobby projects.
- Shading, texture mapping: Cover the principles up to Phong model and texture coordinates.
- Rasterization, z-buffering: cover basic ideas

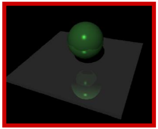
There's not much left, but it is all very important!

Overview of Today

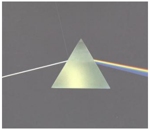
- Shadows



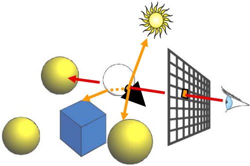
- Reflection



- Refraction

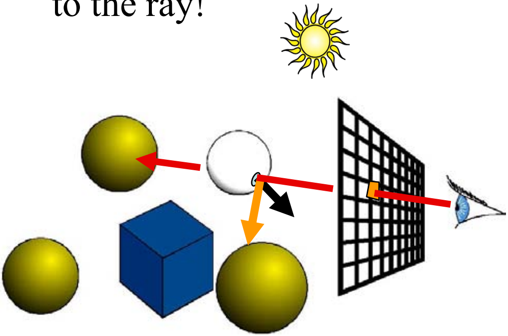


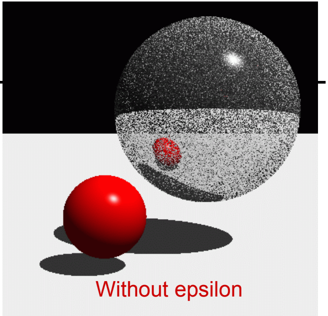
- Recursive Ray Tracing

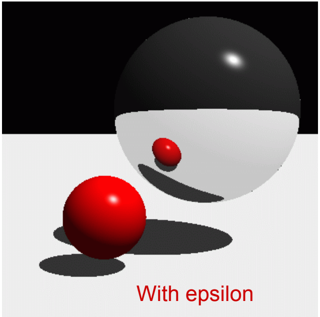


Mirror Reflection

- Cast ray symmetric with respect to the normal
- Multiply by reflection coefficient k_s (color)
- Don't forget to add epsilon to the ray!

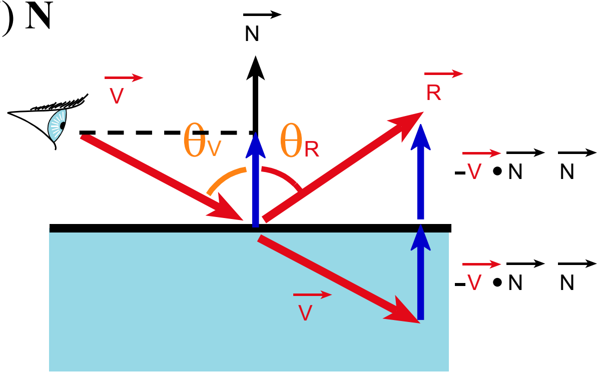






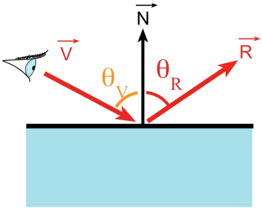
Perfect Mirror Reflection

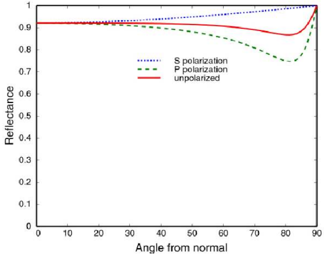
- Reflection angle = view angle
  - Normal component is negated
  - Remember particle collisions?
- $\mathbf{R} = \mathbf{V} - 2\,(\mathbf{V} \cdot \mathbf{N})\,\mathbf{N}$



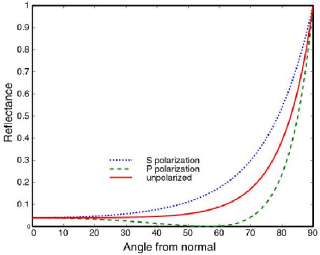
Amount of Reflection

- Traditional ray tracing (hack)
  - Constant k_s
- More realistic (we'll do this later):
  - Fresnel reflection term (more reflection at grazing angle)
  - Schlick's approximation: $R(\theta)=R_0+(1-R_0)(1-\cos\theta)^5$
- Fresnel makes a big difference!





metal

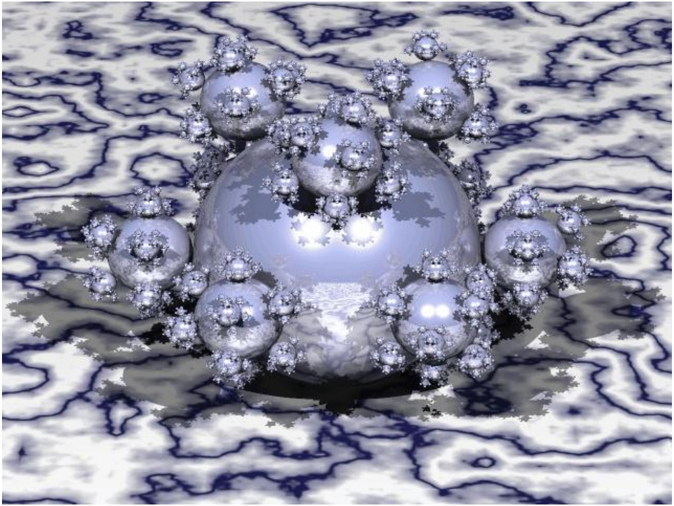


Dielectric (glass)

Questions?

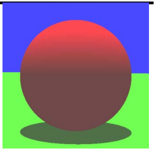
"Sphereflake" fractal

Henrik Wann Jensen

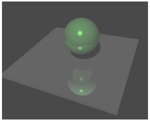


Overview of Today

- Shadows

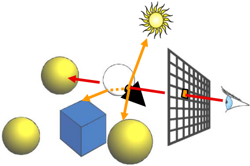


- Reflection



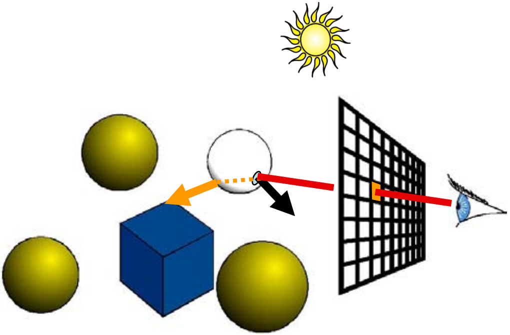
- Refraction

- Recursive Ray Tracing

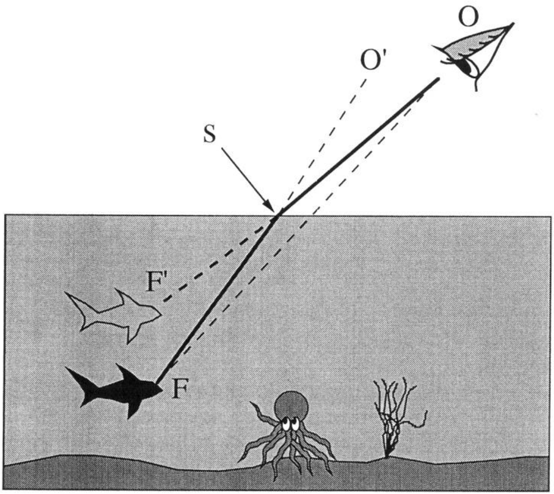


Transparency (Refraction)

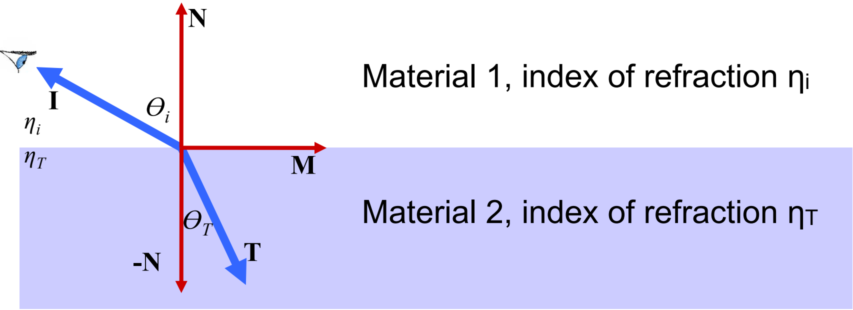
- Cast ray in refracted direction
- Multiply by transparency coefficient k_t (color)



Qualitative Refraction



Refraction



Snell-Descartes Law:

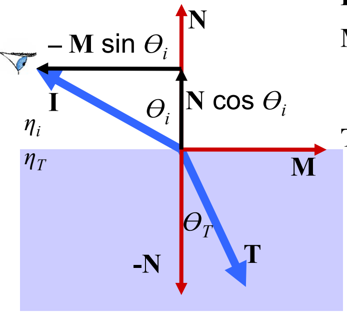
$$n_i \sin\theta_i = n_T \sin\theta_T$$

$$\frac{\sin\theta_T}{\sin\theta_i} = \frac{n_i}{n_T} = n_r$$

Relative index of refraction

Refracted direction **T**?

Refraction



$$\mathbf{I} = \mathbf{N}\cos\Theta_i - \mathbf{M}\sin\Theta_i$$

$$\mathbf{M} = (\mathbf{N}\cos\Theta_i - \mathbf{I}) / \sin\Theta_i$$

$$\mathbf{T} = -\mathbf{N}\cos\Theta_T + \mathbf{M}\sin\Theta_T$$

$$= -\mathbf{N}\cos\Theta_T + (\mathbf{N}\cos\Theta_i - \mathbf{I})\sin\Theta_T / \sin\Theta_i$$ *Plug M*

$$= -\mathbf{N}\cos\Theta_T + (\mathbf{N}\cos\Theta_i - \mathbf{I})\,\eta_r$$ *let's get rid of the cos & sin*

$$= [\,\eta_r\cos\Theta_i - \cos\Theta_T\,]\,\mathbf{N} - \eta_r\mathbf{I}$$

$$= [\,\eta_r\cos\Theta_i - \sqrt{1-\sin^2\Theta_T}\,]\,\mathbf{N} - \eta_r\mathbf{I}$$

$$= [\,\eta_r\cos\Theta_i - \sqrt{1-\eta_r^2\sin^2\Theta_i}\,]\,\mathbf{N} - \eta_r\mathbf{I}$$

$$= [\,\eta_r\cos\Theta_i - \sqrt{1-\eta_r^2(1-\cos^2\Theta_i)}\,]\,\mathbf{N} - \eta_r\mathbf{I}$$

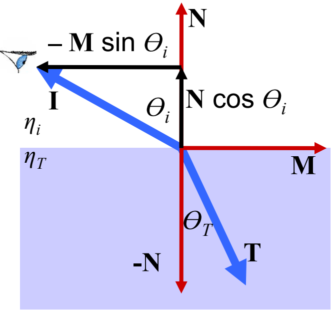
$$= [\,\eta_r(\mathbf{N}\cdot\mathbf{I}) - \sqrt{1-\eta_r^2(1-(\mathbf{N}\cdot\mathbf{I})^2)}\,]\,\mathbf{N} - \eta_r\mathbf{I}$$

Snell-Descartes Law:

$$n_i \sin\theta_i = n_T \sin\theta_T$$

$$\frac{\sin\theta_T}{\sin\theta_i} = \frac{n_i}{n_T} = n_r$$

Refraction



$\mathbf{I} = \mathbf{N} \cos \Theta_i - \mathbf{M} \sin \Theta_i$

$\mathbf{M} = (\mathbf{N} \cos \Theta_i - \mathbf{I}) / \sin \Theta_i$

- **Total internal reflection** when the square root is imaginary (no refraction, just reflection)

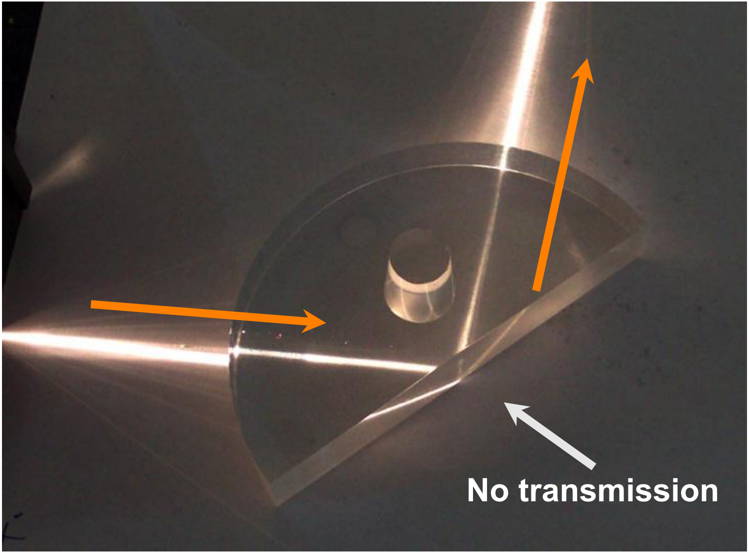
Snell-Descartes Law:

$$n_i \sin \theta_i = n_T \sin \theta_T$$

$$\frac{\sin \theta_T}{\sin \theta_i} = \frac{n_i}{n_T} = n_r$$

$$= [\, \eta_r (\mathbf{N} \cdot \mathbf{I}) - \sqrt{1 - \eta_r^2 (1 - (\mathbf{N} \cdot \mathbf{I})^2)}\,]\, \mathbf{N} - \eta_r \mathbf{I}$$

Total Internal Reflection



Total Internal Reflection

Fig. 3.7A The optical manhole. From under water, the entire celestial hemisphere is compressed into a circle only 97.2° across. The dark boundary defining the edges of the manhole is not sharp due to surface waves. The rays are analogous to the crepuscular type seen in hazy air, Section 1.9. (Photo by D. Granger)

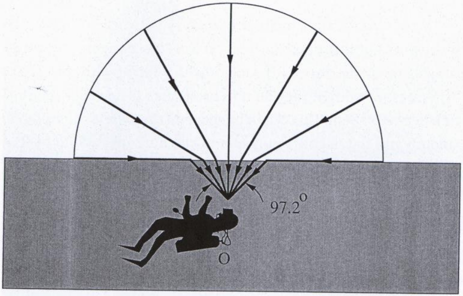
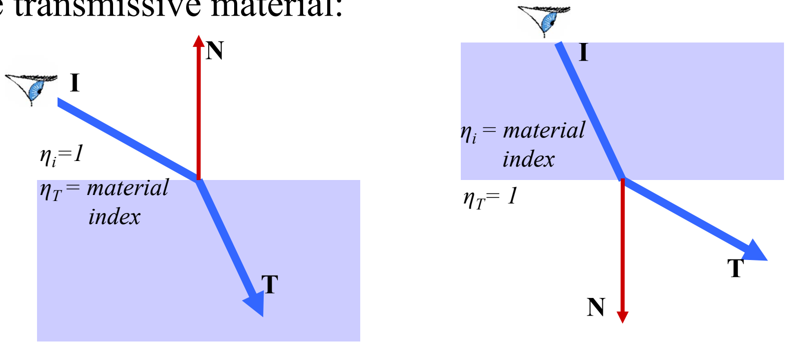


Fig. 3.7B The optical manhole. Light from the horizon (angle of incidence = 90°) is refracted downward at an angle of 48.6°. This compresses the sky into a circle with a diameter of 97.2° instead of its usual 180°.

Refraction & Sidedness of Objects

- Make sure you know whether you're entering or leaving the transmissive material:



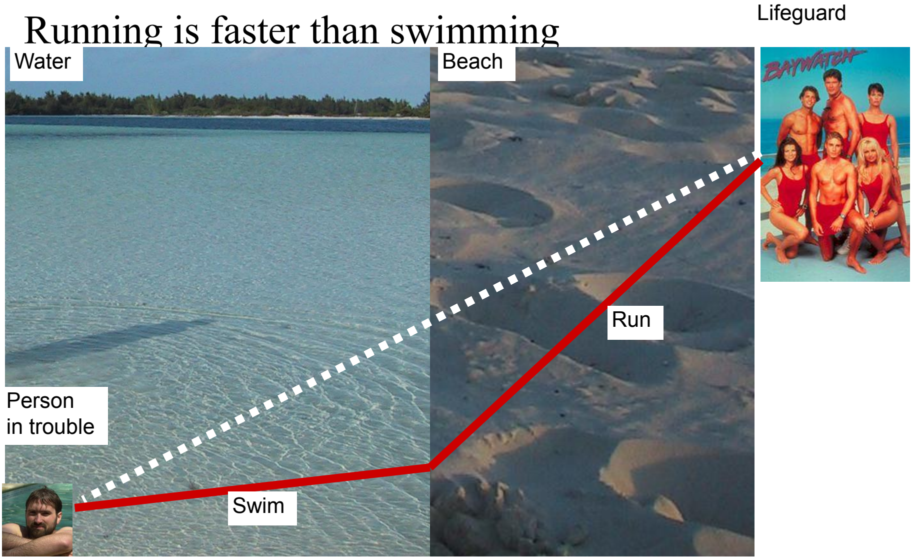
- Note: We won't ask you to trace rays through intersecting transparent objects **:-)**

Cool Refraction Demo

- Enright, D., Marschner, S. and Fedkiw, R., SIGGRAPH 2002

Refraction and the Lifeguard Problem

- Running is faster than swimming



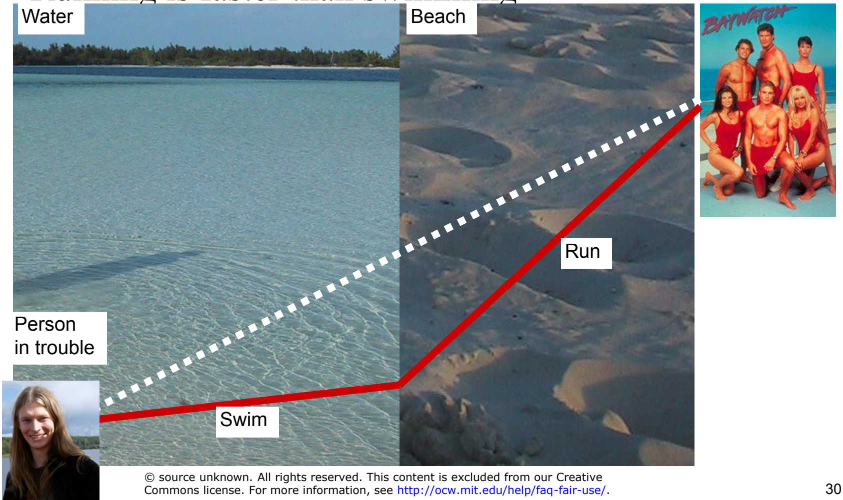
TIEA311 - Today in Jyväskylä

Wait a moment. . . who is in trouble on this course?

Refraction and the Lifeguard Problem

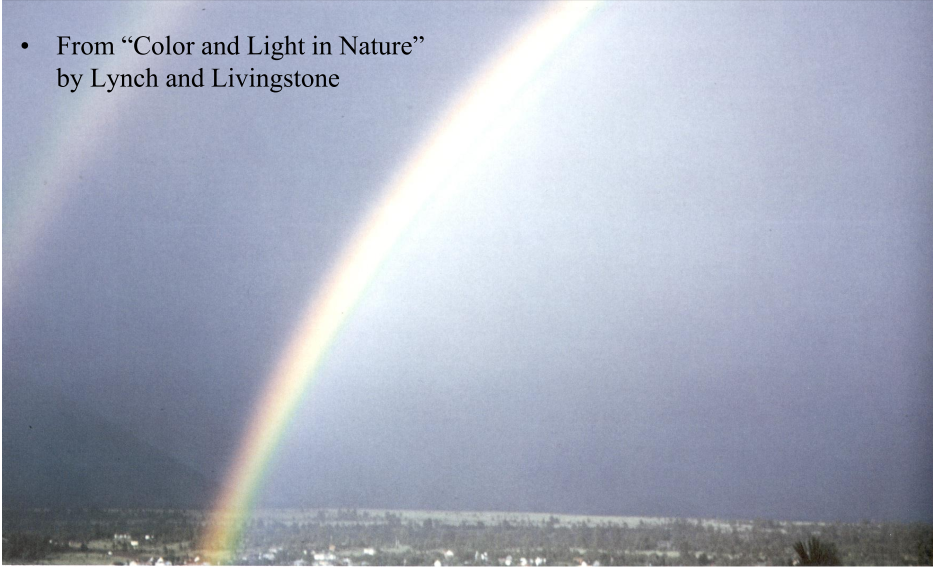
- Running is faster than swimming

Lifeguard



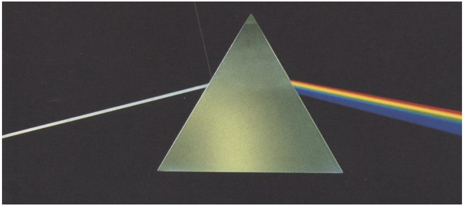
How Does a Rainbow Work?

- From “Color and Light in Nature” by Lynch and Livingstone



Wavelength

- Refraction is wavelength-dependent (dispersion)
  - Refraction increases as the wavelength of light decreases
  - violet and blue experience more bending than orange and red
- Newton's prism experiment
- **Usually ignored in graphics**

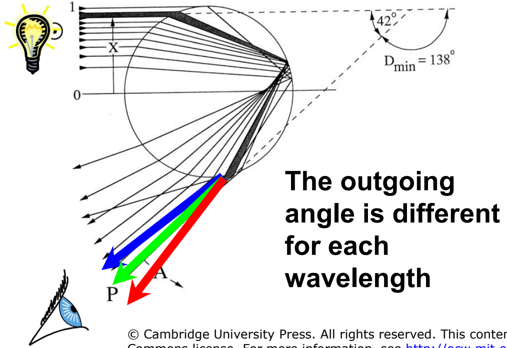


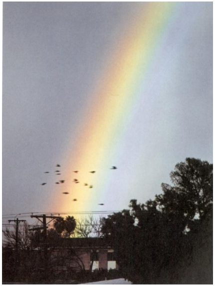
Pink Floyd, *The Dark Side of the Moon*

Pittoni, 1725, Allegory to Newton

Rainbow

- Rainbow is caused by refraction + internal reflection + refraction
- Maximum for angle around 42 degrees
- Refraction depends on wavelength (dispersion)

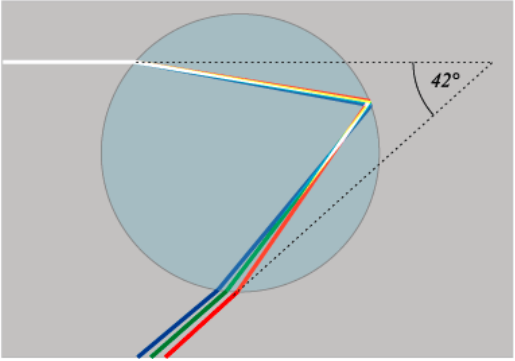




"Color and Light in Nature" by Lynch and Livingstone

Rainbow

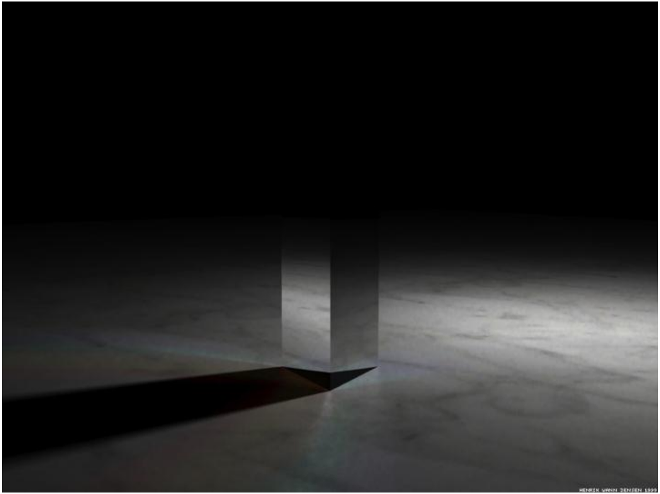
- Rainbow is caused by refraction + internal reflection + refraction
- Maximum for angle around 42 degrees
- Refraction depends on wavelength (dispersion)



This image is in the public domain. Source: Wikipedia.

Dispersion

- Image by Henrik Wann Jensen using Photon Mapping



Courtesy of Henrik Wann Jensen. Used with permission.

Application: CAD for lenses

- Has revolutionized lens design
  - E.g. zoom lenses are good now

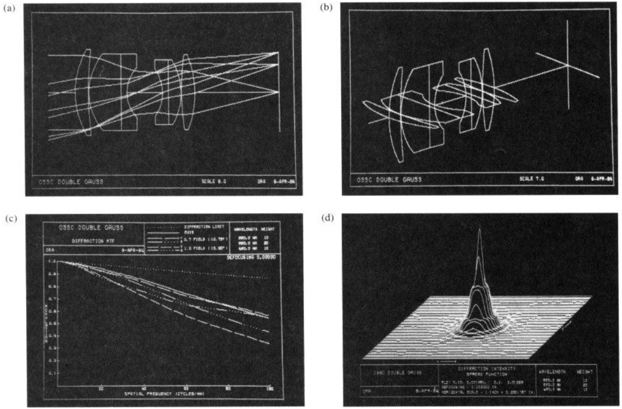


Figure 11.50 An example of the kind of lens design information available via computer techniques. (Photos courtesy Optical Research Associates.)

From Hecht's Optics

Lens design by Ray Tracing

- Used to be done manually, by rooms full of engineers who would trace rays.
- Now software, e.g. Zemax
- More in 6.815/6.865 Computational Photography

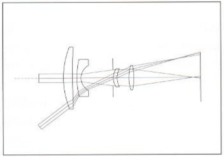


Figure-5

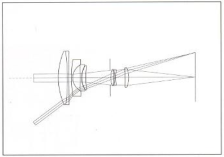


Figure-6

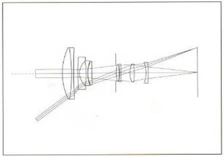


Figure-7

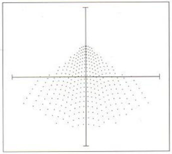


Figure-8

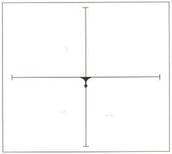


Figure-9

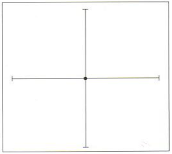
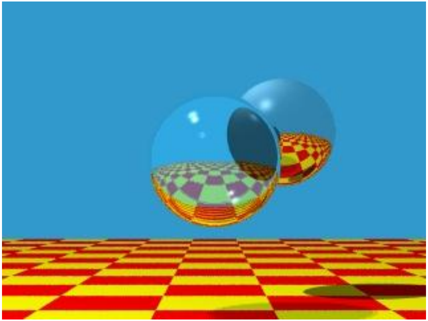


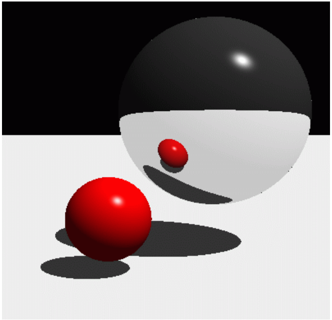
Figure-10

source: canon red book

Let’s Pause for a Moment...

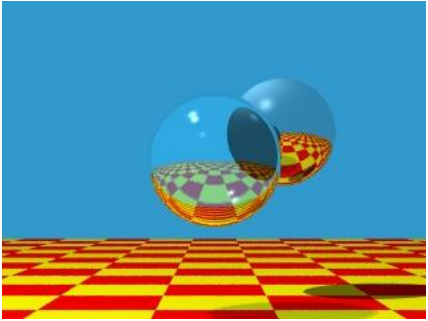
- Do these pictures look real?

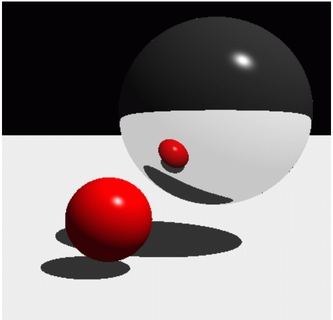




What's Wrong then?

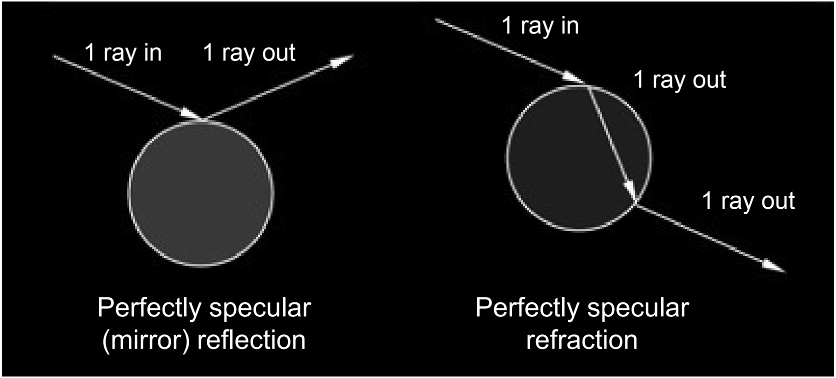
- No surface is a perfect mirror,
  no material interface is perfectly smooth





What's Wrong then?

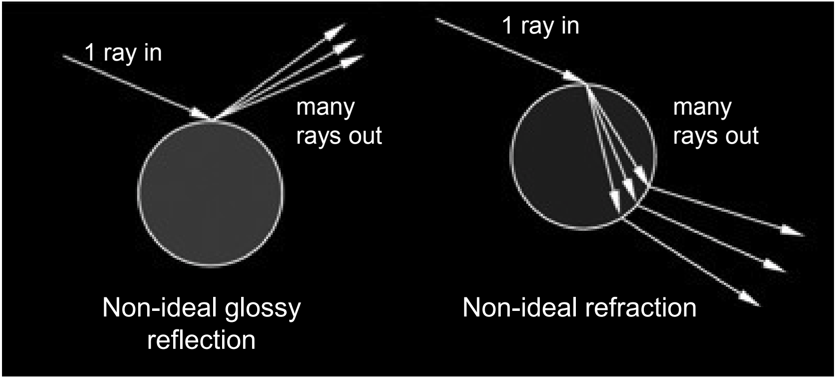
- No surface is a perfect mirror,
  no material interface is perfectly smooth



Adapted from blender.org

Non-Ideal Reflection/Refraction

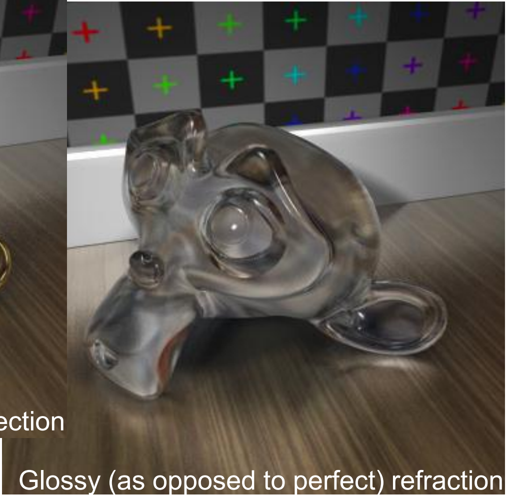
- No surface is a perfect mirror,
  no material interface is perfectly smooth



Adapted from blender.org

Non-Ideal Reflection/Refraction

Glossy (as opposed to mirror) reflection

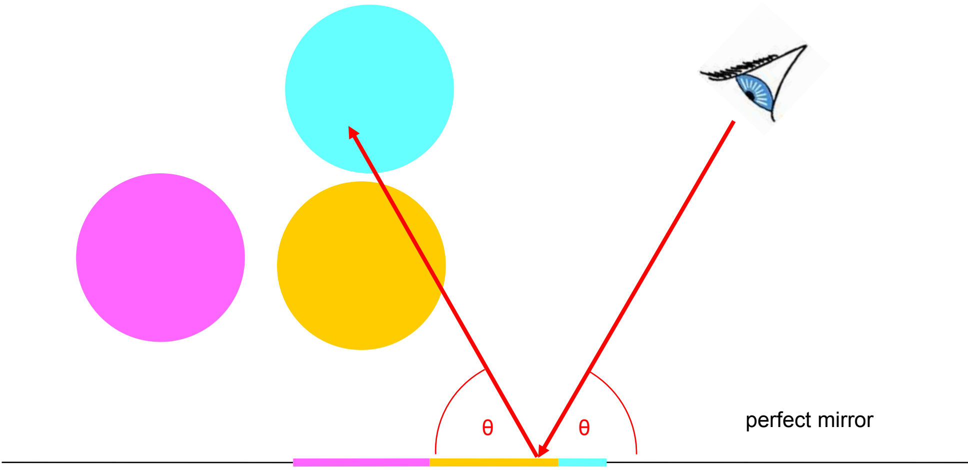


Glossy (as opposed to perfect) refraction

images from blender.org

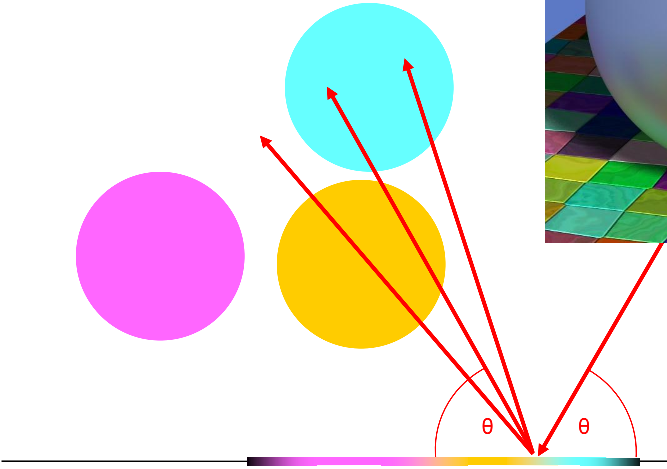
Reflection

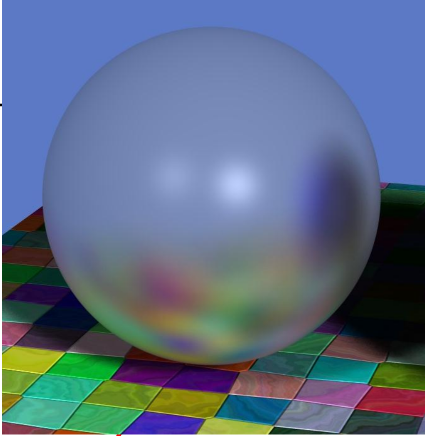
- One reflection ray per intersection



Glossy Reflection

- Multiple reflection rays



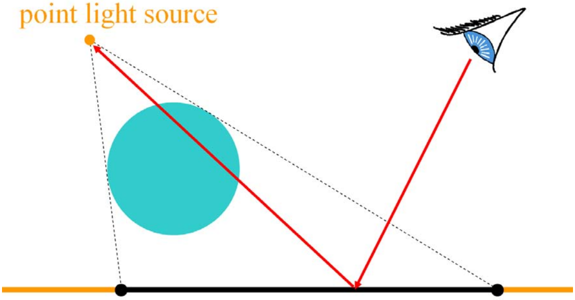


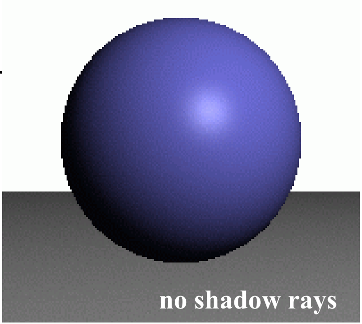
Courtesy of Justin Legakis.

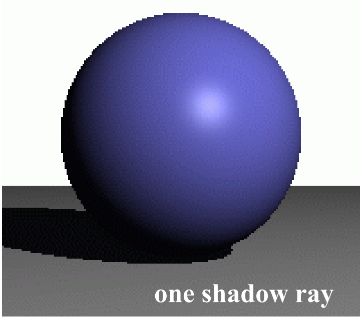
Justin Legakis

Shadows

- One shadow ray per intersection per point light source





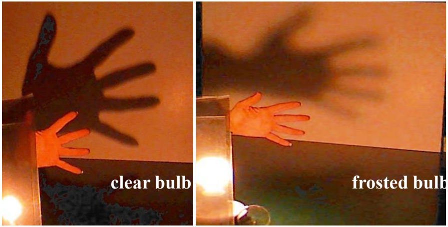


Shadows & Light Sources

Image removed due to copyright restrictions.

http://www.davidfay.com/index.php

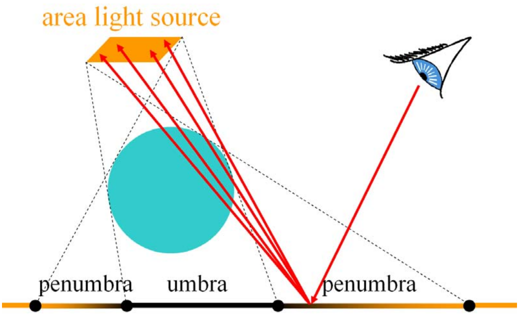
http://3media.initialized.org/photos/2000-10-18/index_gall.htm

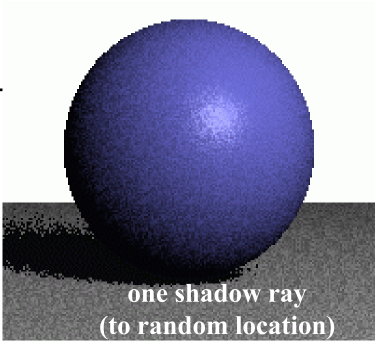


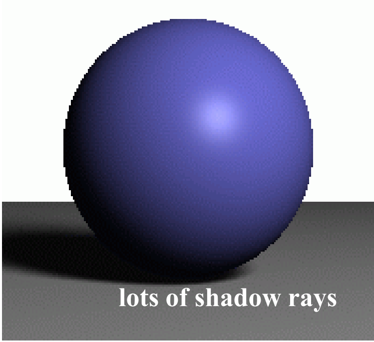
http://www.pa.uky.edu/~sciworks/light/preview/bulb2.htm

Soft Shadows

- Multiple shadow rays to sample area light source

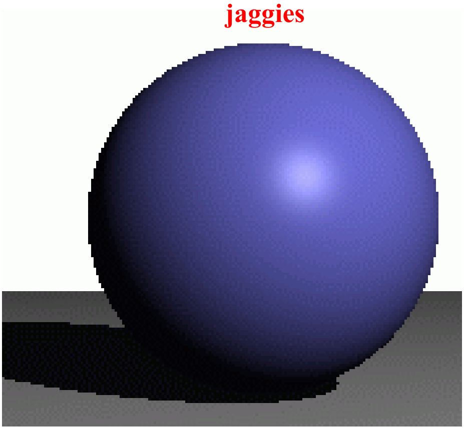


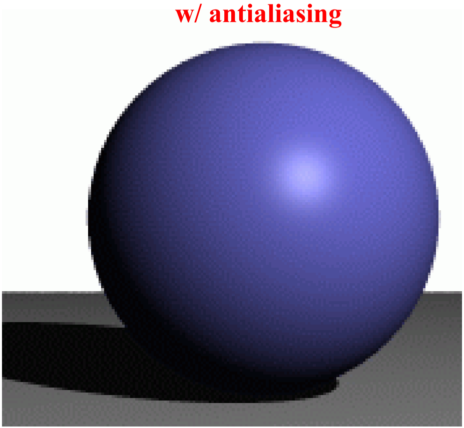




Antialiasing – Supersampling

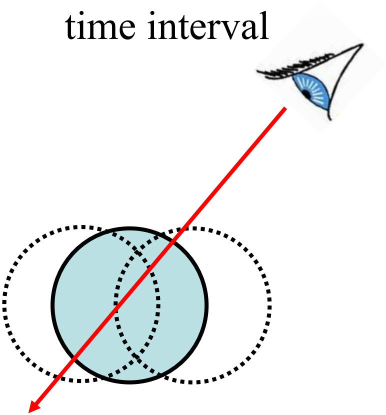
- Multiple rays per pixel





Motion Blur

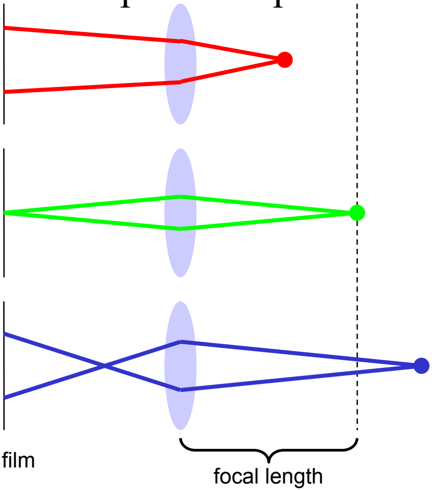
- Sample objects temporally over time interval

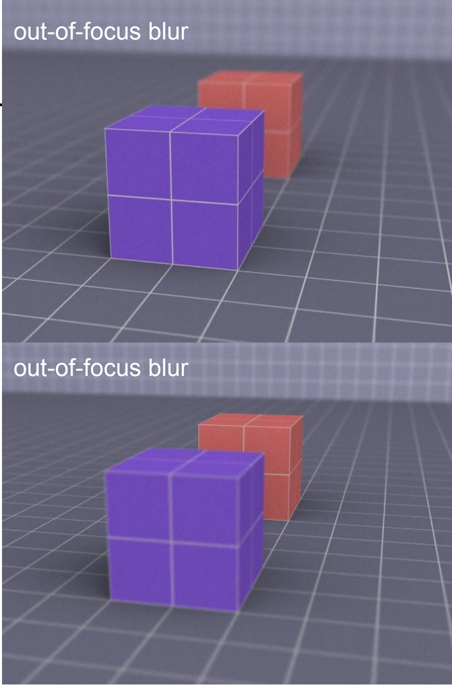


Rob Cook

Depth of Field

- Multiple rays per pixel: sample lens aperture



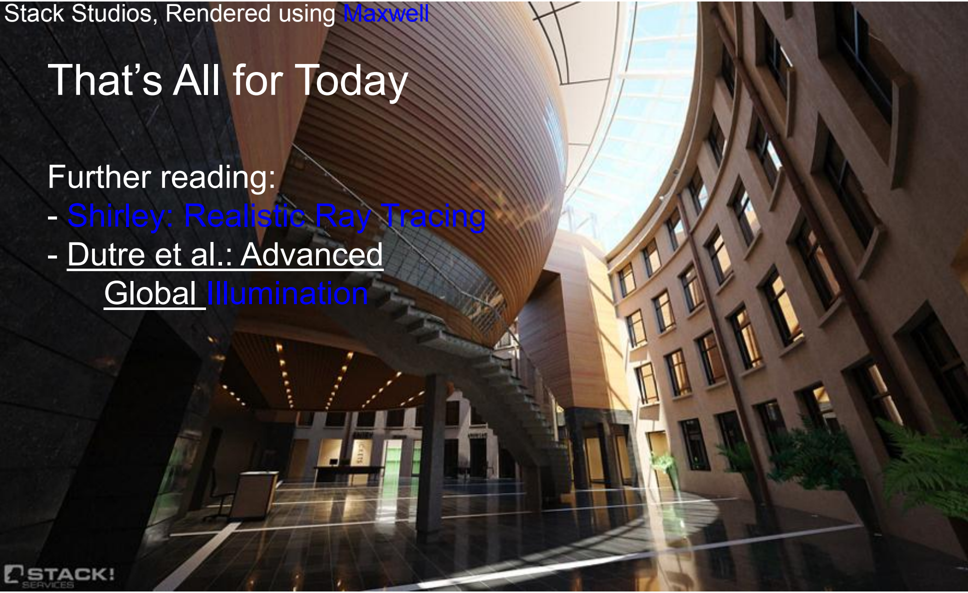


Justin Legakis

Questions?

Henrik Wann Jensen

Courtesy of Henrik Wann Jensen. Used with permission.



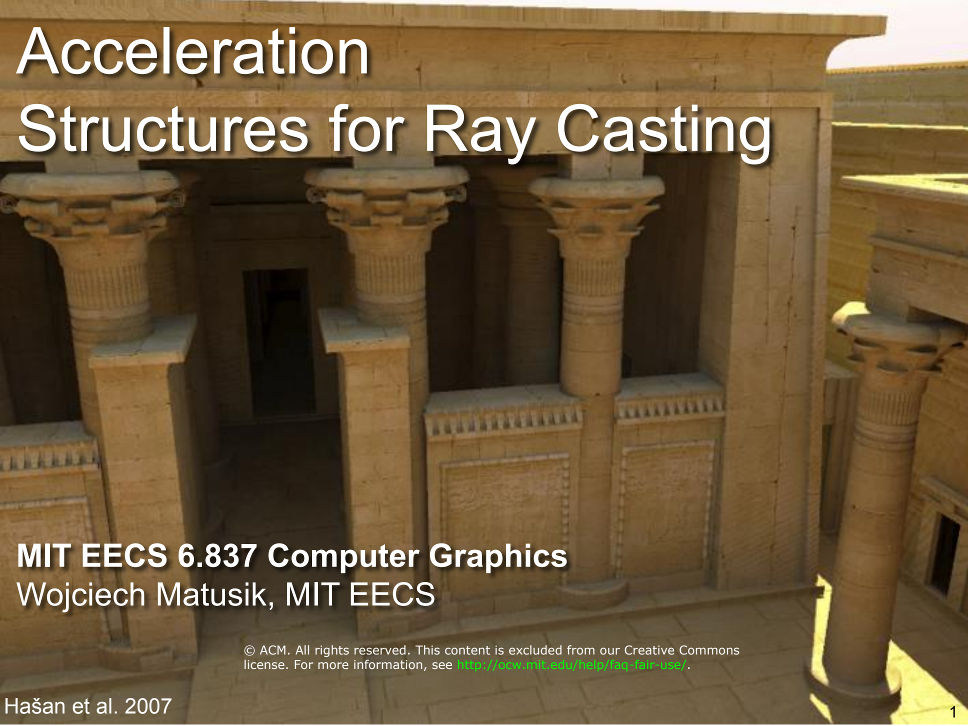
Stack Studios, Rendered using Maxwell

That's All for Today

Further reading:

- Shirley: Realistic Ray Tracing
- Dutre et al.: Advanced Global Illumination

Acceleration Structures for Ray Casting



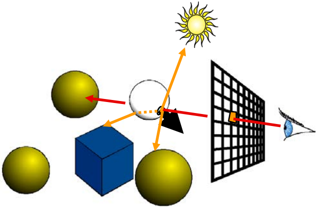
MIT EECS 6.837 Computer Graphics
Wojciech Matusik, MIT EECS

Hašan et al. 2007

Recap: Ray Tracing

```
trace ray
    Intersect all objects
    color = ambient term
    For every light
        cast shadow ray
        color += local shading term
    If mirror
        color += color_refl *
                  trace reflected ray
    If transparent
        color += color_trans *
                 trace transmitted ray
```

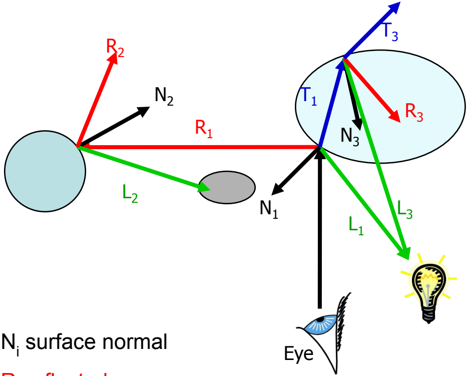
- ***Does it ever end?***



Stopping criteria:

- Recursion depth
  - Stop after a number of bounces
- Ray contribution
  - Stop if reflected / transmitted contribution becomes too small

The Ray Tree

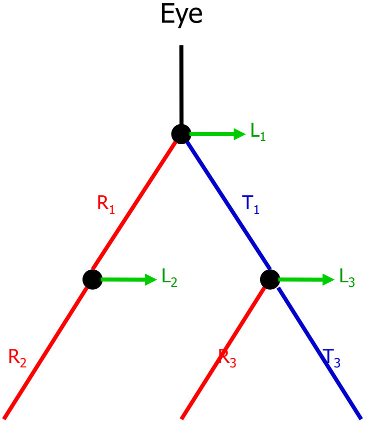


N_i surface normal

R_i reflected ray

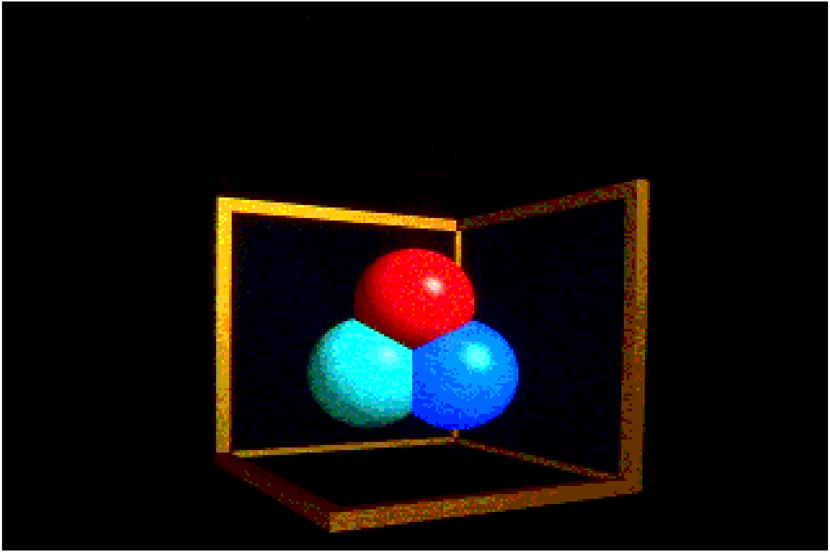
L_i shadow ray

T_i transmitted (refracted) ray

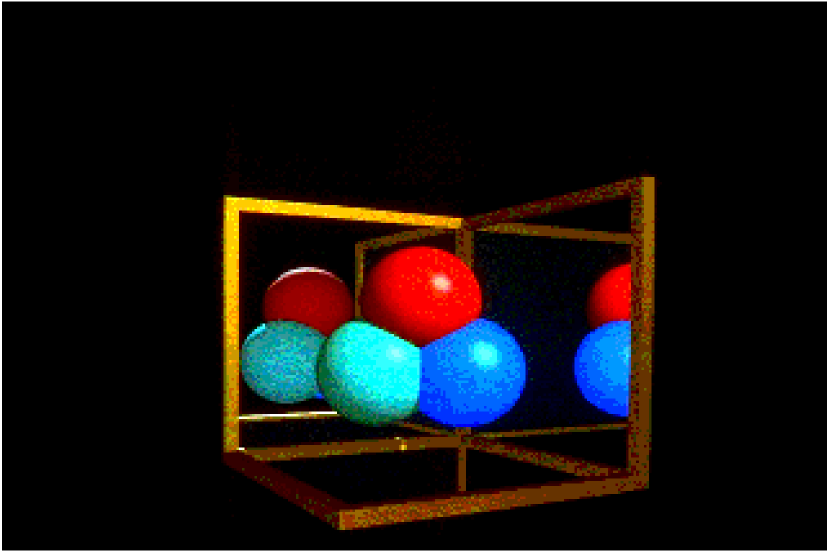


Complexity?

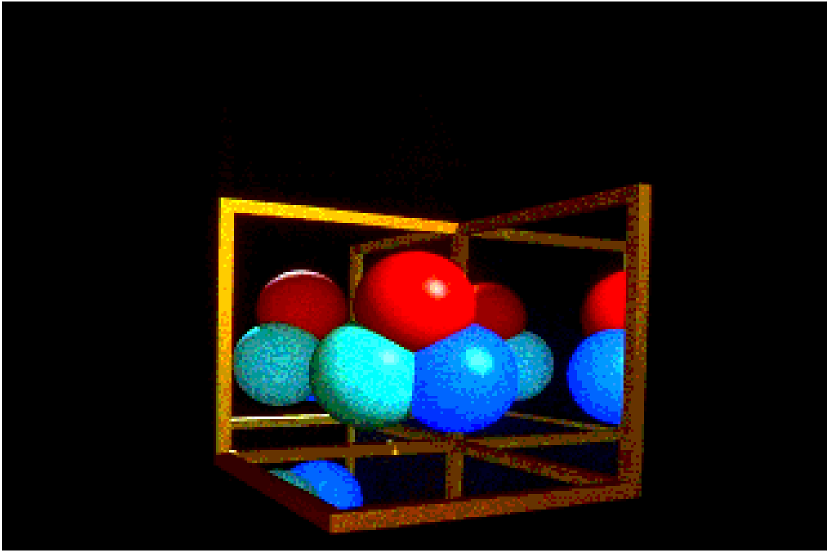
Recursion For Reflection: None



Recursion For Reflection: 1

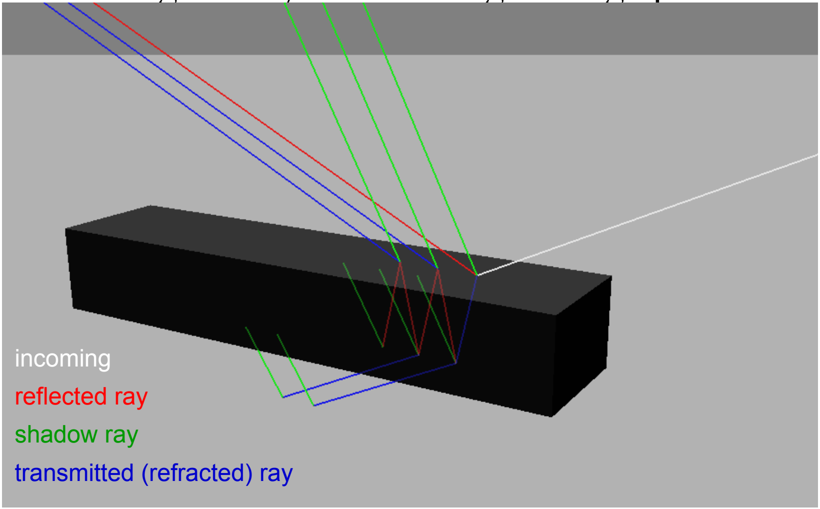


Recursion For Reflection: 2



Ray tree

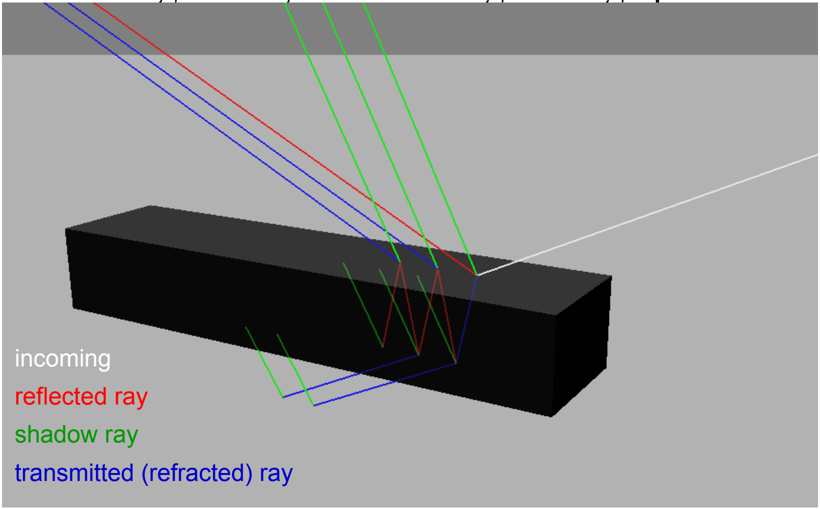
- Visualizing the ray tree for single image pixel



Ray tree

This gets pretty complicated pretty fast!

- Visualizing the ray tree for single image pixel



Ray Tracing Algorithm Analysis

- Lots of primitives
- Recursive
- Distributed Ray Tracing
  - Means using many rays for non-ideal/non-pointlike phenomena
    - Soft shadows
    - Anti-aliasing
    - Glossy reflection
    - Motion blur
    - Depth of field

cost ≈ height * width *
num primitives *
intersection cost *
size of recursive ray tree *
num shadow rays *
num supersamples *
num glossy rays *
num temporal samples *
num aperture samples *
. . .

Can we reduce this?

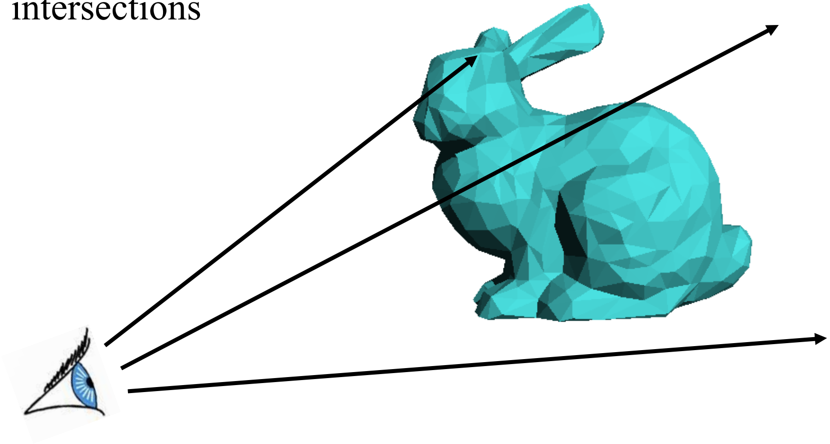
Today

- Motivation
  - You need LOTS of rays to generate nice pictures
  - Intersecting every ray with every primitive becomes the bottleneck
- Bounding volumes
- Bounding Volume Hierarchies, Kd-trees

```
For every pixel
  Construct a ray from the eye
  For every object in the scene
     Find intersection with the ray
     Keep if closest
      Shade
```

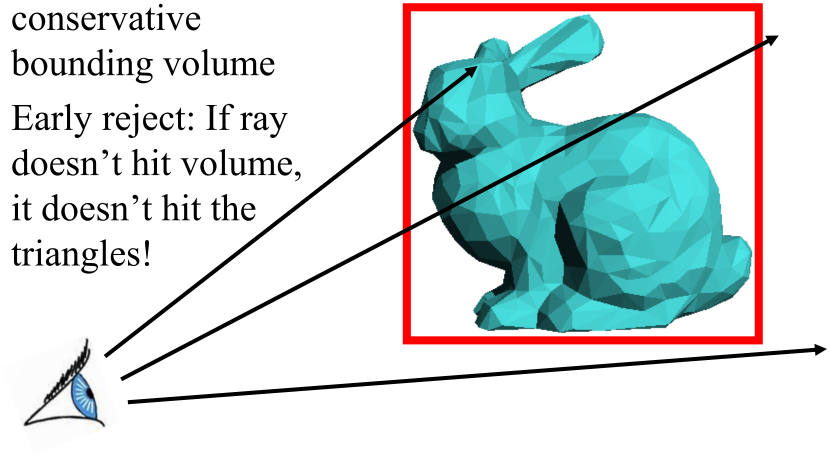
Accelerating Ray Casting

- Goal: Reduce the number of ray/primitive intersections



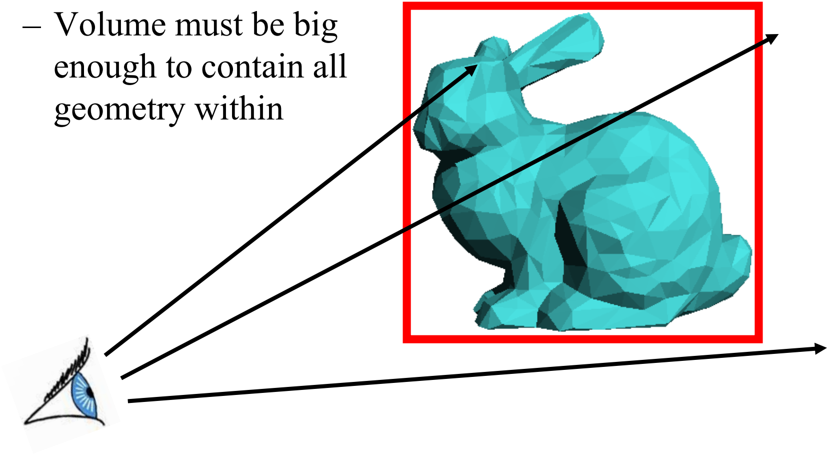
Conservative Bounding Volume

- First check for an intersection with a conservative bounding volume
- Early reject: If ray doesn't hit volume, it doesn't hit the triangles!



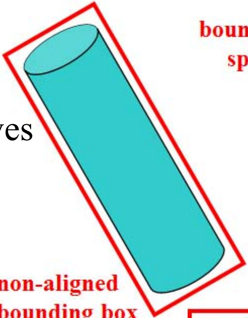
Conservative Bounding Volume

- What does "conservative" mean?
  - Volume must be big enough to contain all geometry within



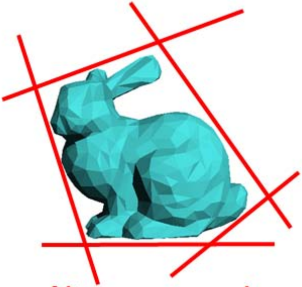
Conservative Bounding Regions

- Desiderata
  - Tight →
    avoid false positives
  - Fast to intersect

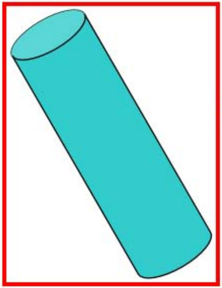


non-aligned
bounding box

bounding
sphere



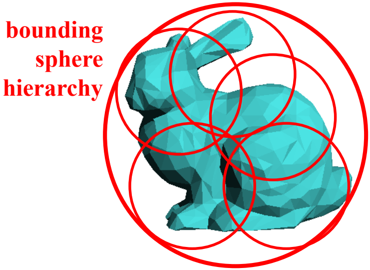
arbitrary convex region
(bounding half-spaces)



axis-aligned
bounding box

Bounding Volume Hierarchies

- If ray hits bounding volume,
  must we test all primitives inside it?
  - Lots of work, think of a 1M-triangle mesh
- You guessed it already, we'll split the primitives in groups and build recursive bounding volumes
  - Like collision detection, remember?



TIEA311 - Today in Jyväskylä

Sorry, guys. . . If you want to learn more about this stuff, you need to do it all by yourselves. At this point, we skip a lot of material about optimizing algorithms (all sorts of similar algorithms, even if graphics is an example, btw.).

To learn more, you may want to check out (on your own time) “Lecture 14” of the original course, and some books and articles about ray tracing. Should you want to use this stuff in your own hobby projects, do ask about possibilities of getting credit points. State-of-the-art methods are good topics for Bachelor / Master thesis projects.

This course will now teleport over algorithmic optimizations. (This is, of course, called “course optimization”!) – we haven’t covered all the fundamentals yet, so let’s not get stuck with details. The next few slides from the MIT course give references for further study (not necessary for our course).

Questions?

- Further reading on efficient Kd-tree construction
  - Hunt, Mark & Stoll, IRT 2006
  - Zhou et al., SIGGRAPH Asia 2008

Zhou et al.

Optimizing Splitting Planes

- Most people use the Surface Area Heuristic (SAH)
  - MacDonald and Booth 1990, "Heuristic for ray tracing using space subdivision", Visual Computer
- Idea: simple probabilistic prediction of traversal cost based on split distance
- Then try different possible splits and keep the one with lowest cost
- Further reading on efficient Kd-tree construction
  - Hunt, Mark & Stoll, IRT 2006
  - Zhou et al., SIGGRAPH Asia 2008

Hard-core efficiency considerations

- See e.g. Ingo Wald's PhD thesis
  - http://www.sci.utah.edu/~wald/PhD/
- Calculation
  - Optimized barycentric ray-triangle intersection
- Memory
  - Make kd-tree node as small as possible (dirty bit packing, make it 8 bytes)
- Parallelism
  - SIMD extensions, trace 4 rays at a time, mask results where they disagree

Pros and Cons of Kd trees

- Pros
  - Simple code
  - Efficient traversal
  - Can conform to data

- Cons
  - costly construction, not great if you work with moving objects

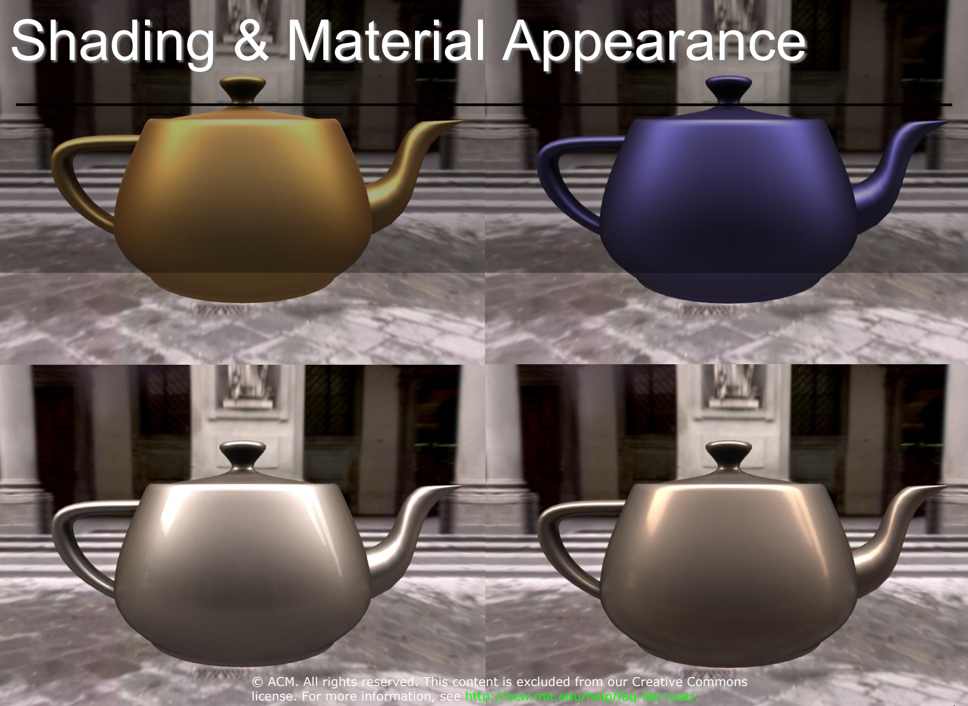
Questions?

- For extensions to moving scenes, see Real-Time KD-Tree Construction on Graphics Hardware, Zhou et al., SIGGRAPH 2008

TIEA311 - Today in Jyväskylä

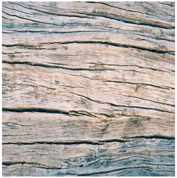
Back to basics. . .

Shading & Material Appearance



Lighting and Material Appearance

- Input for realistic rendering
  - Geometry, Lighting and Materials
- Material appearance
  - Intensity and shape of highlights
  - Glossiness
  - Color
  - Spatial variation, i.e., texture (next Tuesday)



Slide Addy Ngan

Unit Issues - Radiometry

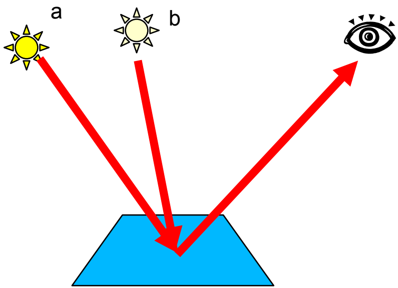
- We will not be too formal in this class
- Issues we will not really care about
  - Directional quantities vs. integrated over all directions
  - Differential terms: per solid angle, per area
  - Power? Intensity? Flux?

- Color
  - All math here is for a single wavelength only; we will perform computations for R, G, B separately
    - Do not panic, that just means
      we will perform every operation three times, that is all

Light Sources

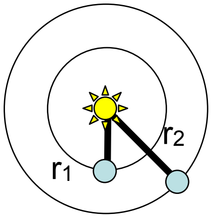
- Today, we only consider point light sources
  - Thus we do not need to care about solid angles
- For multiple light sources, use linearity
  - We can add the solutions for two light sources
    - I(a+b) = I(a) + I(b)
  - We simply multiply the solution when we scale the light intensity
    - I(s a) = s I(a)

Yet again, linearity is our friend!



Intensity as Function of Distance

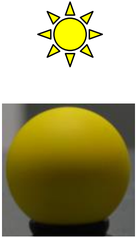
- $1/r^2$ fall-off for isotropic point lights
  - Why? An isotropic point light outputs constant power per solid angle ("into all directions")
  - Must have same power in all concentric spheres
    - Sphere's surface area grows with r^2 => energy obeys $1/r^2$

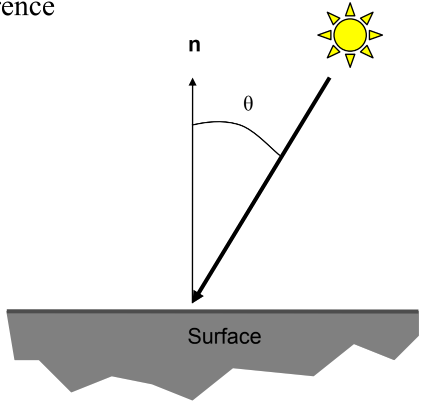


- … but in graphics we often cheat with or ignore this.
  - Why? Ideal point lights are kind of harsh
    - Intensity goes to infinity when you get close – not great!
  - In particular, $1/(ar^2+br+c)$ is popular

Incoming Irradiance

- The amount of light energy received by a surface depends on incoming angle
  - Bigger at normal incidence, even if distance is const.
    - Similar to winter/summer difference
- How exactly?
  - Cos θ law
  - Dot product with normal



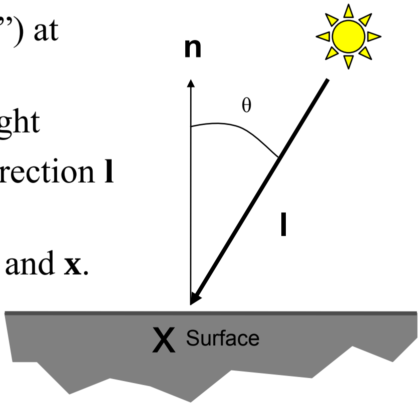


Incoming Irradiance for Pointlights

- Let's combine this with the $1/r^2$ fall-off:

$$I_{\text{in}} = I_{\text{light}} \cos\theta / r^2$$

  - I_{in} is the irradiance ("intensity") at surface point **x**
  - I_{light} is the "intensity" of the light
  - θ is the angle between light direction **l** and surface normal **n**
  - r is the distance between light and **x**.

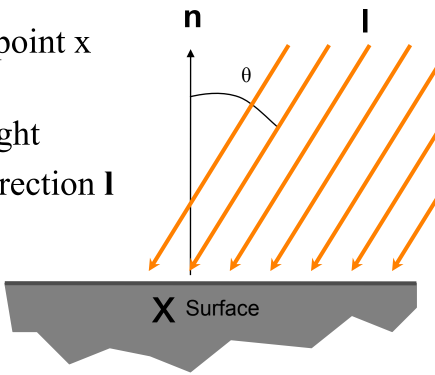


Directional Lights

- "Pointlights that are infinitely far"
  - No falloff, just one direction and one intensity

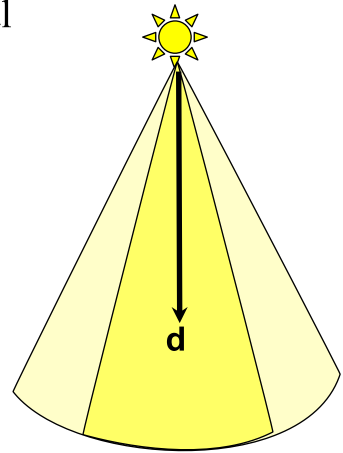
$$I_{\text{in}} = I_{\text{light}} \cos\theta$$

  - I_{in} is the irradiance at surface point x from the directional light
  - I_{light} is the "intensity" of the light
  - θ is the angle between light direction **l** and surface normal **n**
    - Only depends on **n**, not **x**!



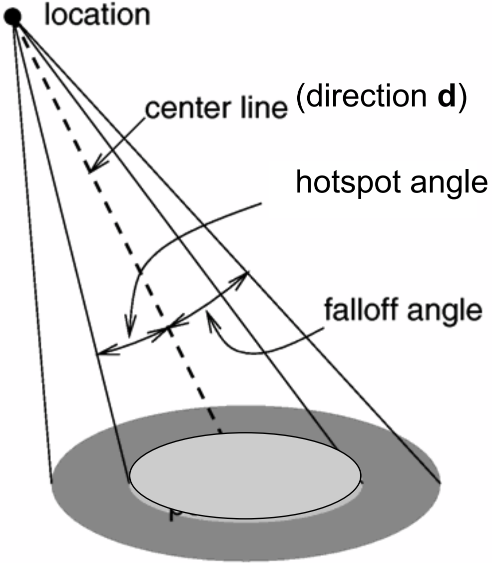
Spotlights

- Pointlights with non-uniform directional emission
- Usually symmetric about a central direction **d**, with angular falloff
  - Often two angles
    - "Hotspot" angle:
      No attenuation within the central cone
    - "Falloff" angle: Light attenuates from full intensity to zero intensity between the hotspot and falloff angles



- Plus your favorite distance falloff curve

Spotlight Geometry

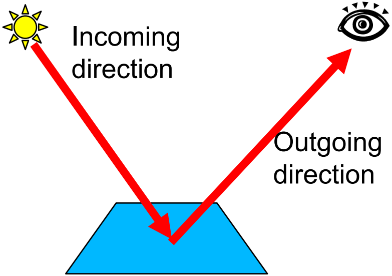


Adapted from
POVRAY documentation

Questions?

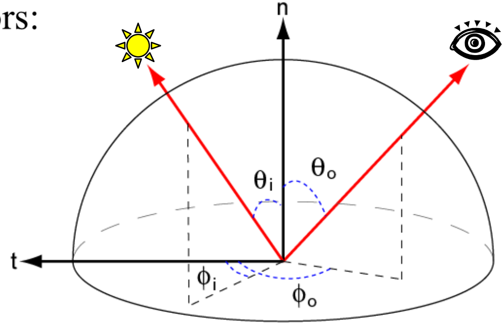
Quantifying Reflection – BRDF

- Bidirectional Reflectance Distribution Function
- Ratio of light coming from one direction that gets reflected in another direction
  - Pure reflection, assumes no light scatters into the material
- Focuses on angular aspects, not spatial variation of the material
- **How many dimensions?**



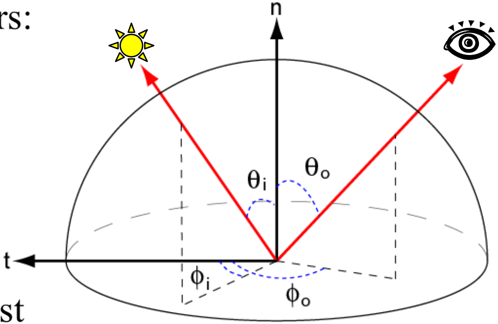
BRDF f_r

- Bidirectional Reflectance Distribution Function
  - 4D: 2 angles for each direction
  - BRDF = $f_r(\theta_i, \phi_i; \theta_o, \phi_o)$
  - Or just two unit vectors: BRDF = $f_r(\mathbf{l}, \mathbf{v})$
    - $\mathbf{l}$ = light direction
    - $\mathbf{v}$ = view direction



BRDF f_r

- Bidirectional Reflectance Distribution Function
  - 4D: 2 angles for each direction
  - BRDF = $f_r(\theta_i, \phi_i; \theta_o, \phi_o)$
  - Or just two unit vectors: BRDF = $f_r(\mathbf{l}, \mathbf{v})$
    - $\mathbf{l}$ = light direction
    - $\mathbf{v}$ = view direction
  - The BRDF is aligned with the surface; the vectors **l** and **v** must be in a local coordinate system

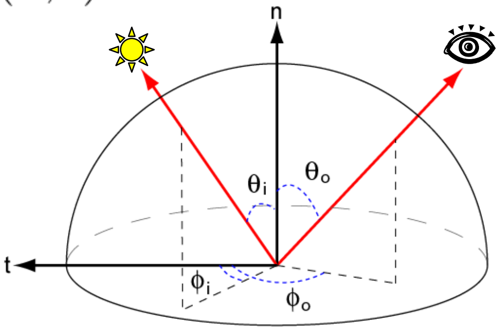


BRDF f_r

- Relates incident irradiance from every direction to outgoing light. How?

l = light direction (incoming)
v = view direction (outgoing)

$$I_{\text{out}}(\boldsymbol{v}) = I_{\text{in}}(\boldsymbol{l})\, f_r(\boldsymbol{v}, \boldsymbol{l})$$



BRDF f_r

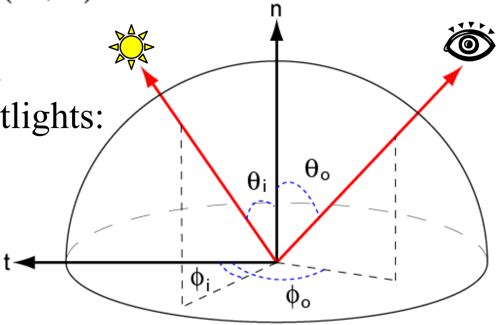
- Relates incident irradiance from every direction to outgoing light. How?

$$I_{\text{out}}(\boldsymbol{v}) = I_{\text{in}}(\boldsymbol{l})\, f_r(\boldsymbol{v}, \boldsymbol{l})$$

l = light direction (incoming)
v = view direction (outgoing)

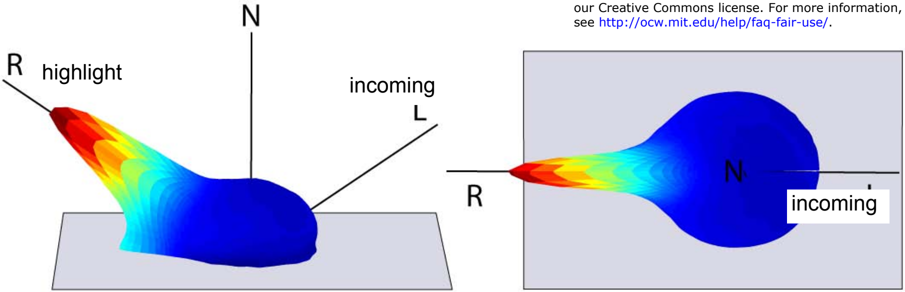
- Let's combine with what we know already of pointlights:

$$I_{\text{out}}(\boldsymbol{v}) = \frac{I_{\text{light}} \cos\theta_i}{r^2} f_r(\boldsymbol{v}, \boldsymbol{l})$$



2D Slice at Constant Incidence

- For a fixed incoming direction, view dependence is a 2D spherical function
  - Here a moderate specular component



Courtesy of Mitsubishi Electric Researh Laboratories, Inc. Used with permission.

Example: Plot of "PVC" BRDF at 55° incidence

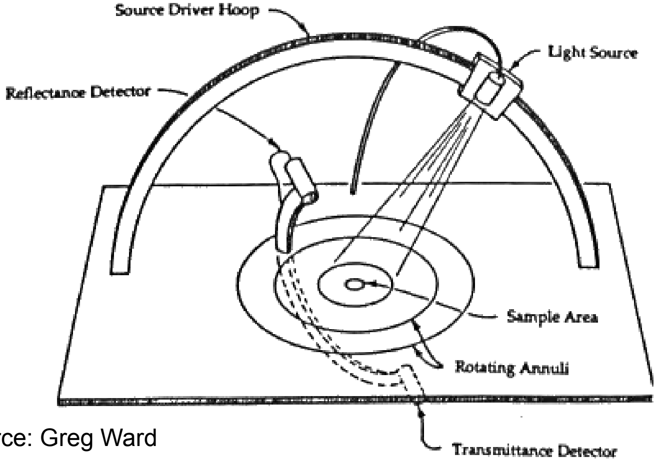
Isotropic vs. Anisotropic

- When keeping **l** and **v** fixed, if rotation of surface around the normal does not change the reflection, the material is called isotropic
- Surfaces with strongly oriented microgeometry elements are anisotropic
- Examples:
  - brushed metals,
  - hair, fur, cloth, velvet

Westin et.al 92

How do we obtain BRDFs?

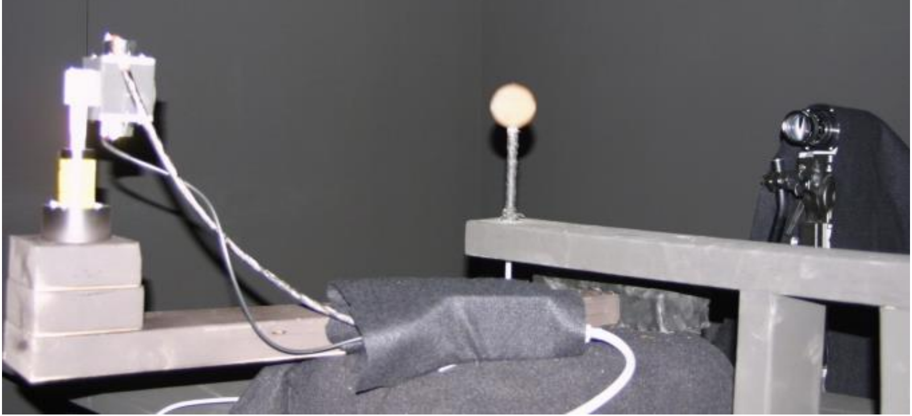
- One possibility: Gonioreflectometer
  - 4 degrees of freedom



Source: Greg Ward

How Do We Obtain BRDFs?

- Another possibility: Take pictures of spheres coated with material, rotate light around a 1D arc
  - This gives 3DOF => isotropic materials only



Matusik et al.

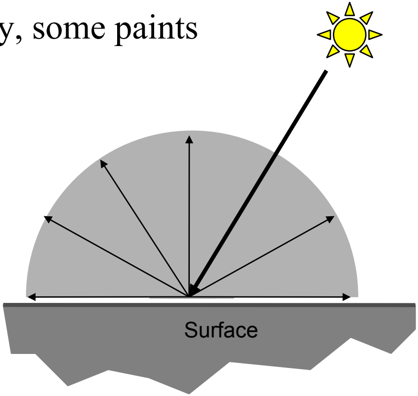
Parametric BRDFs

- BRDFs can be measured from real data
  - But tabulated 4D data is too cumbersome for most uses
- Therefore, parametric BRDF models represent the relationship between incident and outgoing light by some mathematical formula
  - The appearance can then be tuned by setting parameters
    - "Shininess", "anisotropy", etc.
  - Physically-based or Phenomenological
  - They can model with measured data (examples later)
- Popular models: Diffuse, Blinn-Phong, Cook-Torrance, Lafortune, Ward, Oren-Nayar, etc.

Questions?

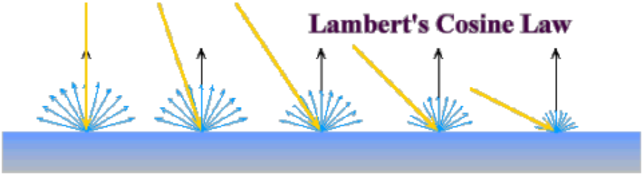
Ideal Diffuse Reflectance

- Assume surface reflects equally in all directions.
- An ideal diffuse surface is, at the microscopic level, a very rough surface.
  - Example: chalk, clay, some paints



Ideal Diffuse Reflectance

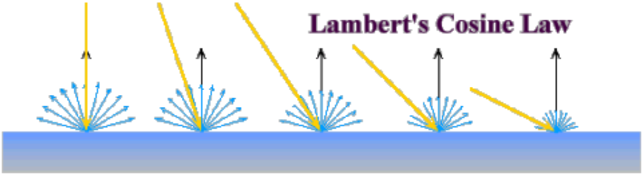
- Ideal diffuse reflectors reflect light according to Lambert's cosine law
  - The reflected light varies with cosine even if distance to light source is kept constant



Ideal Diffuse Reflectance

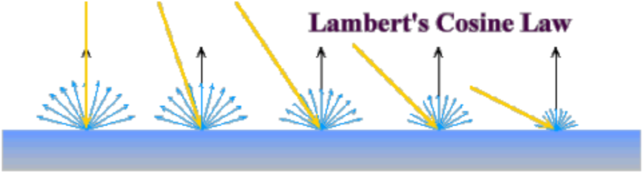
- Ideal diffuse reflectors reflect light according to Lambert's cosine law
  - The reflected light varies with cosine even if distance to light source is kept constant

Remembering that incident irradiance depends on cosine, what is the BRDF of an ideally diffuse surface?



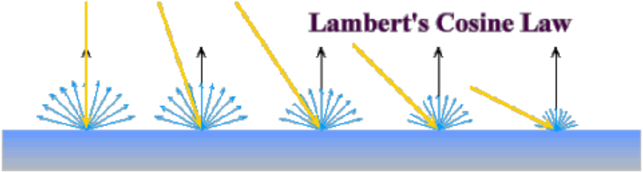
Ideal Diffuse Reflectance

- The ideal diffuse BRDF is a constant $f_r(\mathbf{l}, \mathbf{v})$ = const.
  - What constant ρ/π, where ρ is the *albedo*
    - Coefficient between 0 and 1 that says what fraction is reflected
  - Usually just called "diffuse color" k_d
  - You have already implemented this by taking dot products with the normal and multiplying by the "color"!



Ideal Diffuse Reflectance

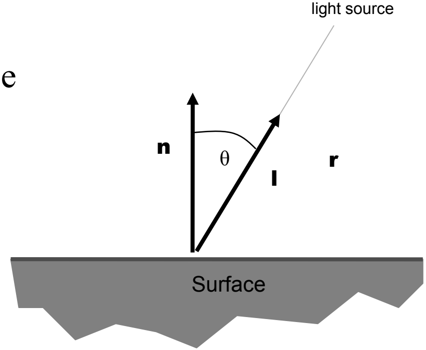
- This is the simplest possible parametric BRDF
  - One parameter: k_d
    - (One for each RGB channel)



Ideal Diffuse Reflectance Math

- Single Point Light Source
  - k_d: diffuse coefficient (color)
  - **n**: Surface normal.
  - **l**: Light direction.
  - L_i: Light intensity
  - r: Distance to source
  - L_o: Shaded color

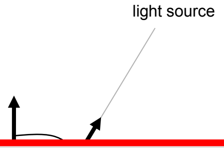
$$L_o = k_d \max(0, \boldsymbol{n} \cdot \boldsymbol{l}) \frac{L_i}{r^2}$$



Ideal Diffuse Reflectance Math

- Single Point Light Source
  - k_d: diffuse coefficient (color)
  - **n**: Surface normal.
  - **l**: Light direction.
  - L_i: Light intensity
  - r: Distance to source
  - L_o: Shaded color

$$L_o = k_d \max(0, \boldsymbol{n} \cdot \boldsymbol{l}) \frac{L_i}{r^2}$$



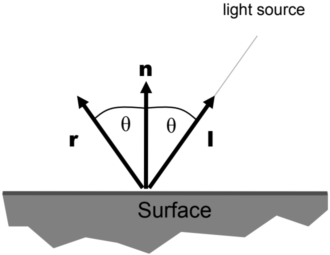
Do not forget to normalize your **n** and **l**!

We do not want light from below the surface! From now on we always assume (on this lecture) that **dot products are clamped to zero** and skip writing out the max().

Questions?

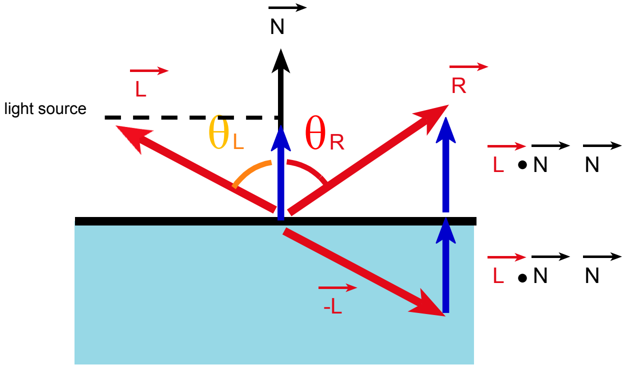
Ideal Specular Reflectance

- Reflection is only at mirror angle
- View dependent
  - Microscopic surface elements are usually oriented in the same direction as the surface itself.
  - Examples: mirrors, highly polished metals.



Recap: How to Get Mirror Direction

- Reflection angle = light angle
  - Both **R** & **L** have to lie on one plane
- $\mathbf{R} = -\mathbf{L} + 2\,(\mathbf{L} \cdot \mathbf{N})\,\mathbf{N}$



Ideal Specular BRDF

- Light **only** reflects to the mirror direction
- A Dirac delta multiplied by a specular coefficient k_s

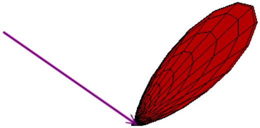
- Not very useful for point lights, only for reflections of other surfaces
  - Why? You cannot really see a mirror reflection of an infinitely small light!

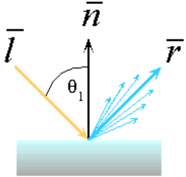
Non-ideal Reflectors

- Real glossy materials usually deviate significantly from ideal mirror reflectors
  - Highlight is blurry
- They are not ideal diffuse surfaces either …

Non-ideal Reflectors

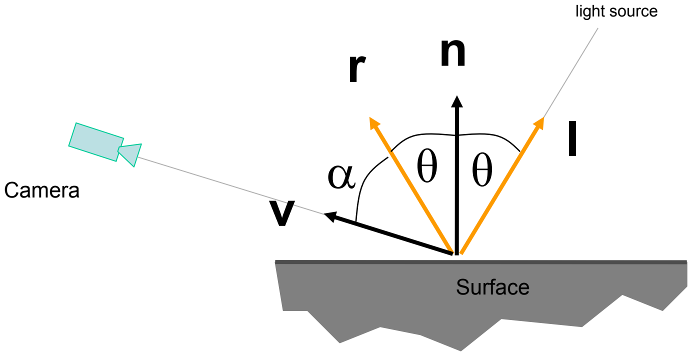
- Simple Empirical Reasoning for Glossy Materials
  - We expect most of the reflected light to travel in the direction of the ideal mirror ray.
  - However, because of microscopic surface variations we might expect some of the light to be reflected just slightly offset from the ideal reflected ray.
  - As we move farther and farther, in the angular sense, from the reflected ray, we expect to see less light reflected.





The Phong Specular Model

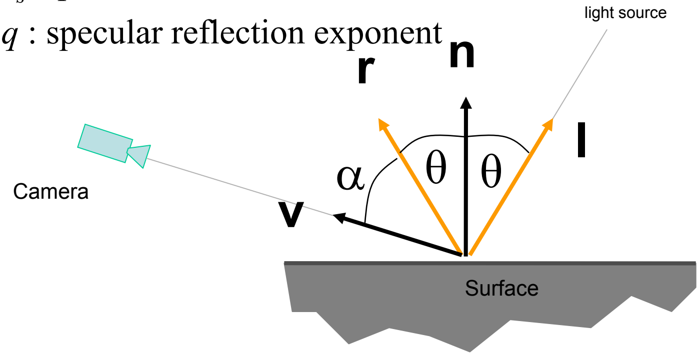
- How much light is reflected?
  - Depends on the angle α between the ideal reflection direction **r** and the viewer direction **v**.



The Phong Specular Model

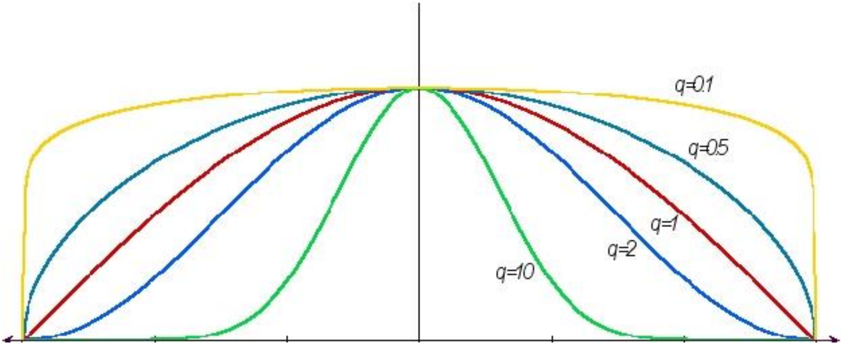
$$L_o = k_s (\cos\alpha)^q \frac{L_i}{r^2} = k_s (\boldsymbol{v} \cdot \boldsymbol{r})^q \frac{L_i}{r^2}$$

- Parameters
  - k_s: specular reflection coefficient
  - q : specular reflection exponent



The Phong Model

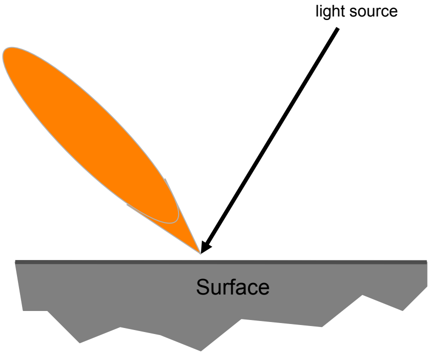
- Effect of q – the specular reflection exponent



$\alpha = -\pi/2$ $\alpha = 0$ $\alpha = \pi/2$

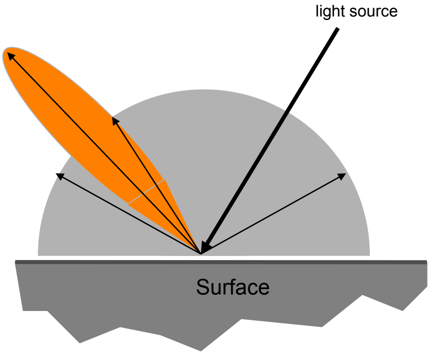
Terminology: Specular Lobe

- The specular reflection distribution is usually called a "lobe"
  - For Phong, its shape is $(\boldsymbol{r} \cdot \boldsymbol{v})^q$



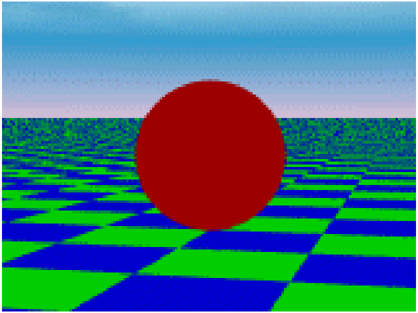
The Complete Phong Model

- Sum of three components:
  ideal diffuse reflection +
  specular reflection +
  "ambient".



Ambient Illumination

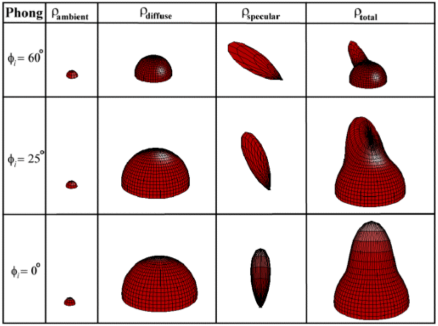
- Represents the reflection of all indirect illumination.
- This is a total hack!
- Avoids the complexity of indirect (“global”) illumination



Putting It All Together

- Phong Illumination Model

$$L_o = \left[k_a + k_d \left(\boldsymbol{n} \cdot \boldsymbol{l} \right) + k_s \left(\boldsymbol{v} \cdot \boldsymbol{r} \right)^q \right] \frac{L_i}{r^2}$$



Putting It All Together

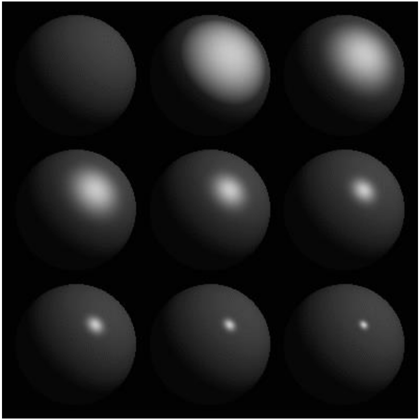
- Phong Illumination Model

$$L_o = \left[k_a + k_d\,(\boldsymbol{n}\cdot\boldsymbol{l}) + k_s\,(\boldsymbol{v}\cdot\boldsymbol{r})^q\right]\frac{L_i}{r^2}$$

- Is it physically based?
  - No, does not even conserve energy, may well reflect more energy than what goes in
  - Furthermore, it does not even conform to the BRDF model directly (we are taking the proper cosine for diffuse, but not for specular)
  - And ambient was a total hack

Phong Examples

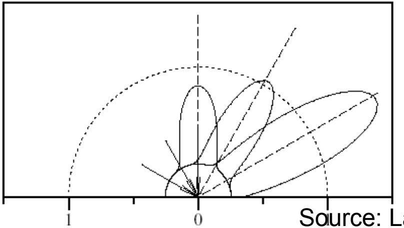
- The spheres illustrate specular reflections as the direction of the light source and the exponent q (amount of shininess) is varied.



$$L_o = \left[k_a + k_d \left(\boldsymbol{n} \cdot \boldsymbol{l} \right) + k_s \left(\boldsymbol{v} \cdot \boldsymbol{r} \right)^q \right] \frac{L_i}{r^2}$$

Fresnel Reflection

- Increasing specularity near grazing angles.
  - Most BRDF models account for this.



Source: Lafortune et al. 97

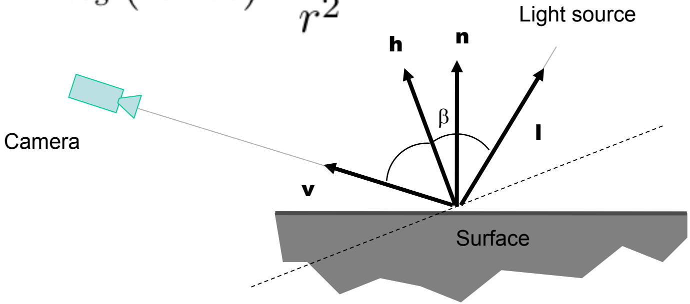
Questions?

Blinn-Torrance Variation of Phong

- Uses the "halfway vector" **h** between **l** and **v**.

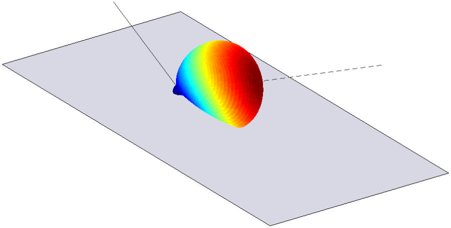
$$L_o = k_s \cos(\beta)^q \frac{L_i}{r^2}$$
$$= k_s (\boldsymbol{n} \cdot \boldsymbol{h})^q \frac{L_i}{r^2}$$

$$\boldsymbol{h} = \frac{\boldsymbol{l} + \boldsymbol{v}}{\|\boldsymbol{l} + \boldsymbol{v}\|}$$

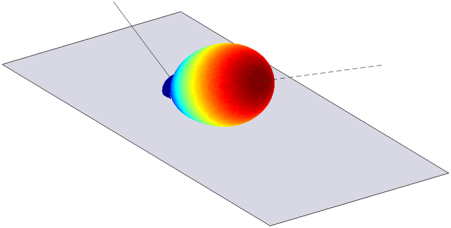


Lobe Comparison

- Half vector lobe
  - Gradually narrower when approaching grazing
- Mirror lobe
  - Always circular



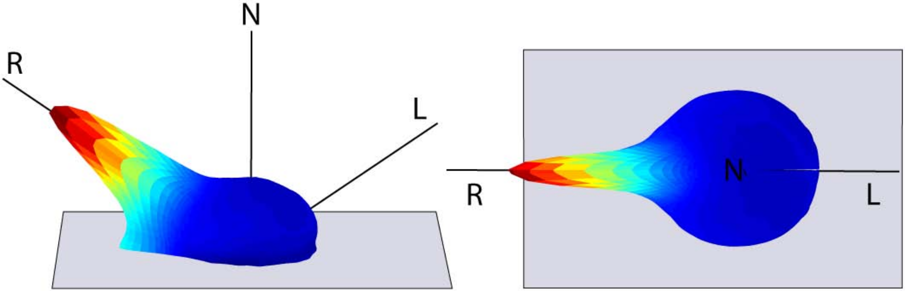
Half vector lobe



Mirror lobe

Half Vector Lobe is Better

- More consistent with what is observed in measurements (Ngan, Matusik, Durand 2005)



Example: Plot of "PVC" BRDF at 55° incidence

Questions?

Microfacet Theory

- Example
  - Think of water surface as lots of tiny mirrors (microfacets)
  - “Bright” pixels are
    - Microfacets aligned with the vector between sun and eye
    - But not the ones in shadow
    - And not the ones that are occluded

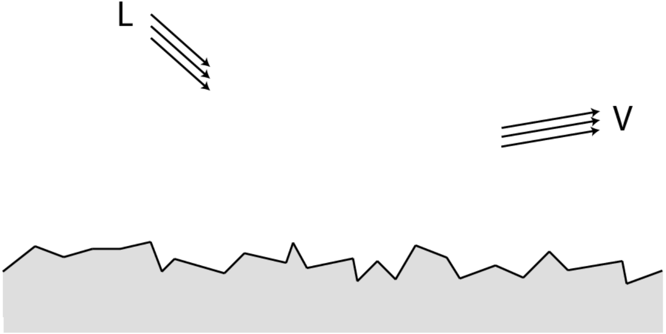
Image of sunset removed due to copyright restrictions.

Microfacet Theory

- Model surface by tiny mirrors
  [Torrance & Sparrow 1967]

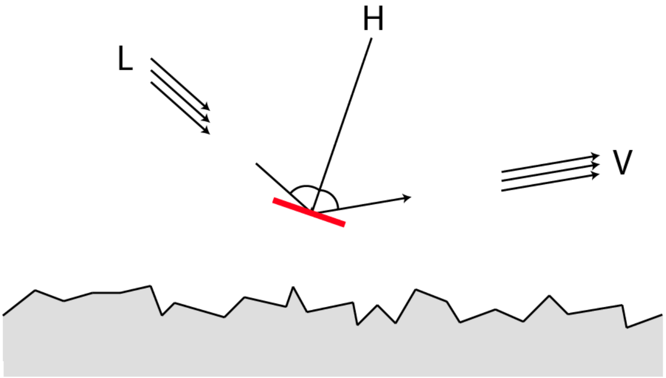
Microfacet Theory

- Value of BRDF at (L,V) is a product of
  - number of mirrors oriented halfway between L and V



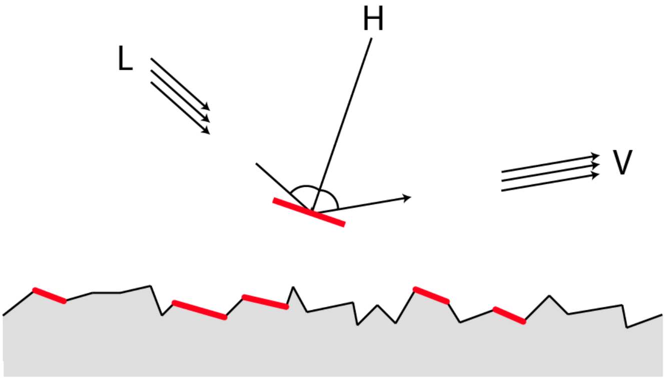
Microfacet Theory

- Value of BRDF at (L,V) is a product of
  - number of mirrors oriented halfway between L and V



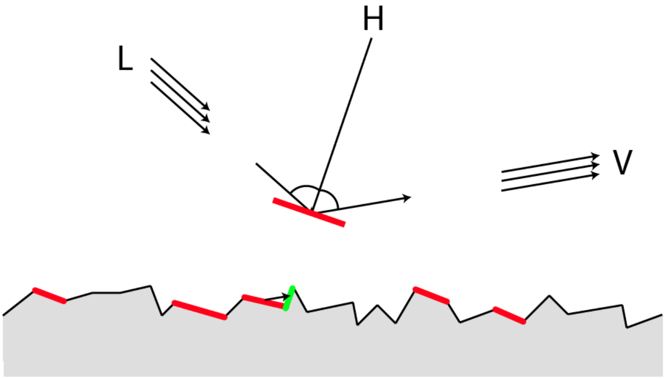
Microfacet Theory

- Value of BRDF at (L,V) is a product of
  - number of mirrors oriented halfway between L and V



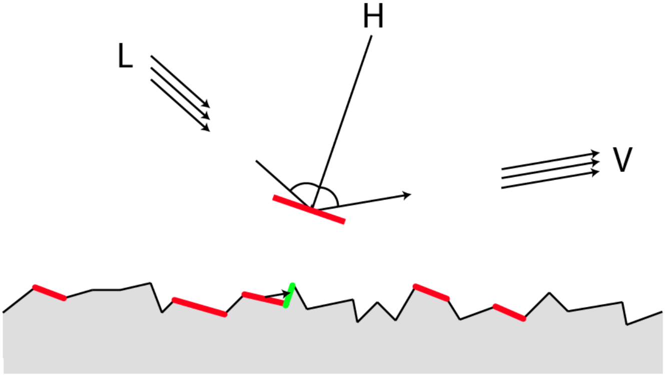
Microfacet Theory

- Value of BRDF at (L,V) is a product of
  - number of mirrors oriented halfway between L and V
  - ratio of the un(shadowed/masked) mirrors



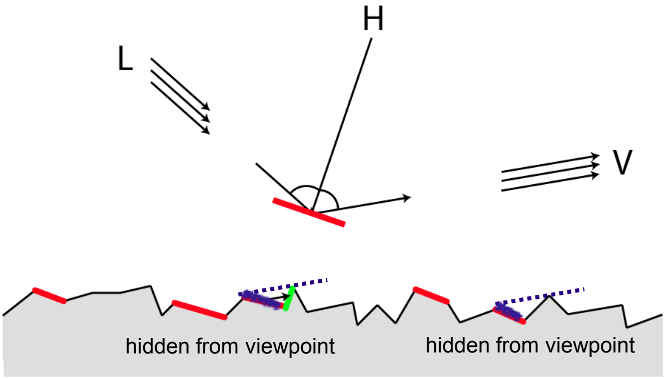
Microfacet Theory

- Value of BRDF at (L,V) is a product of
  - number of mirrors oriented halfway between L and V
  - ratio of the un(shadowed/masked) mirrors
  - Fresnel coefficient



Shadowing and Masking

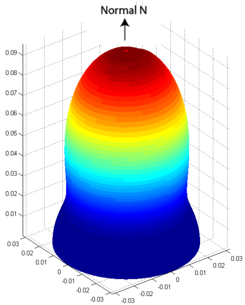
- Some facets are hidden from viewpoint
- Some are hidden from the light



Microfacet Theory-based Models

- Develop BRDF models by imposing simplifications [Torrance-Sparrow 67], [Blinn 77], [Cook-Torrance 81], [Ashikhmin et al. 2000]

- Model the distribution p(H) of microfacet normals
  - Also, statistical models for shadows and masking



spherical plot of a Gaussian-like p(H)

Full Cook-Torrance Lobe

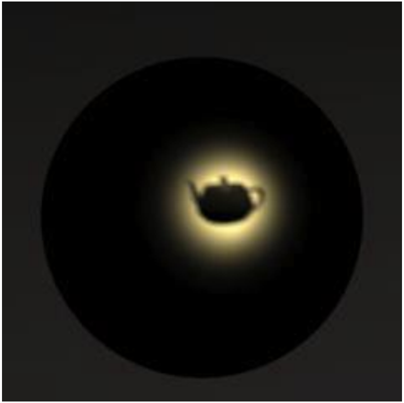
- ρ_s is the specular coefficient (3 numbers RGB)
- D is the microfacet distribution
  - δ is the angle between the half vector H and the normal N
  - m defines the roughness (width of lobe)
- G is the shadowing and masking term
- Need to add a diffuse term

$$K = \frac{\rho_s}{\pi} \frac{DG}{(N \cdot L)(N \cdot V)} Fresnel(F_0, V \cdot H)$$

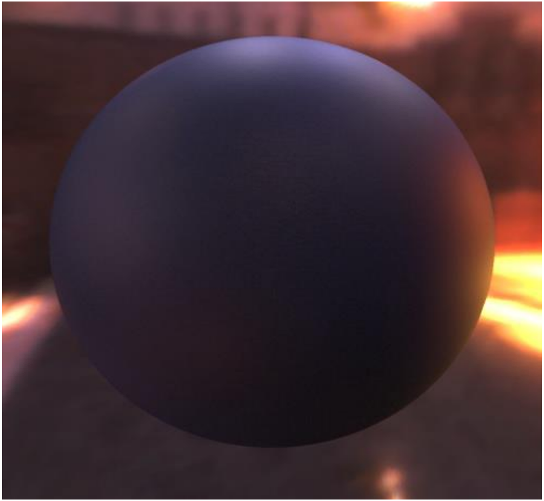
where $G = min\{1, \frac{2(N \cdot H)(N \cdot V)}{(V \cdot H)}, \frac{2(N \cdot H)(N \cdot L)}{(V \cdot H)}\}$ and $D = \frac{1}{m^2 \cos^4 \delta} e^{-[(tan\delta)/m]^2}$

Questions?

- "Designer BRDFs" by Ashikhmin et al.



BRDF Examples from Ngan et al.

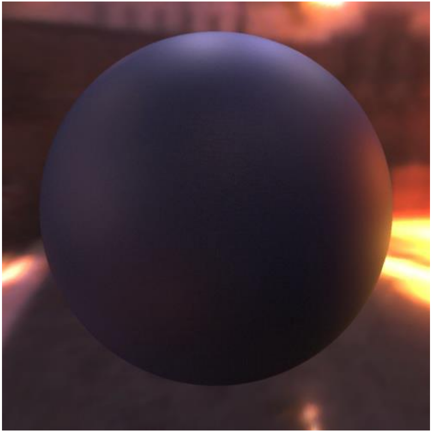


Material – Dark blue paint

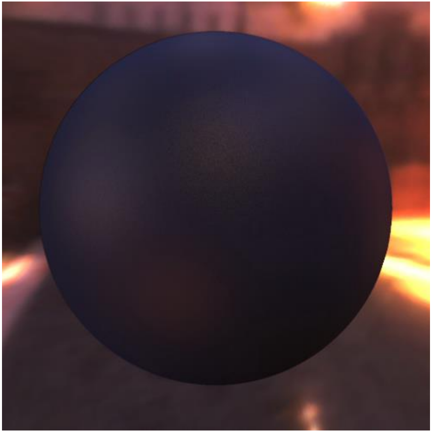
Lighting

Dark Blue Paint

Acquired data



Blinn-Phong

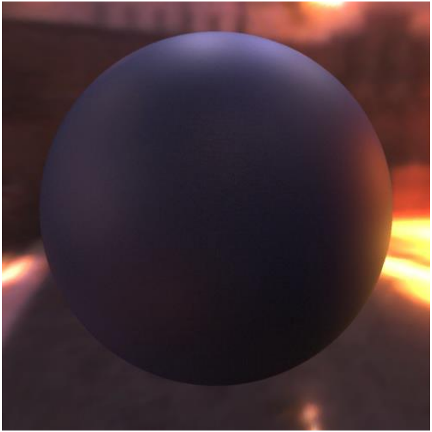


Finding the BRDF model parameters that best reproduce the real material

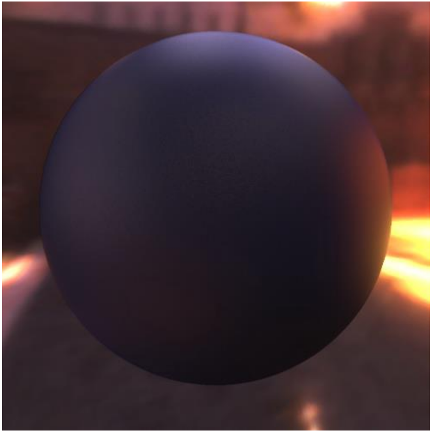
Material – Dark blue paint

Dark Blue Paint

Acquired data



Cook-Torrance



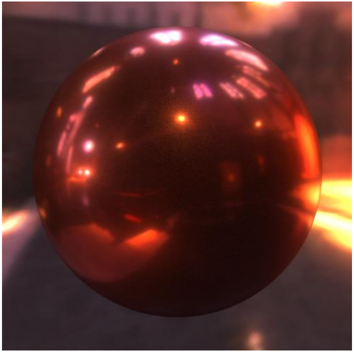
Finding the BRDF model parameters that best reproduce the real material

Material – Dark blue paint

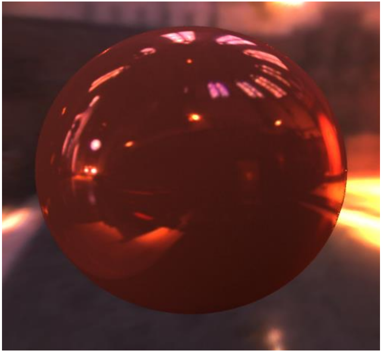
Observations

- Some materials impossible to represent with a single lobe

Acquired data



Cook-Torrance

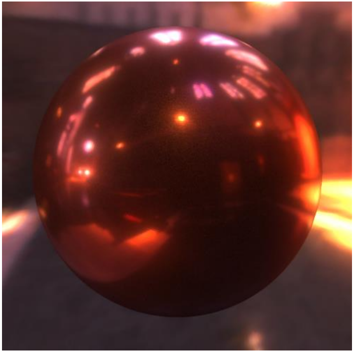


Material – Red Christmas Ball

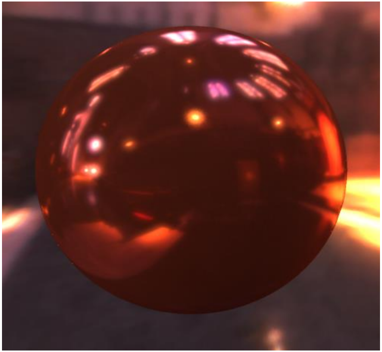
Adding a Second Lobe

- Some materials impossible to represent with a single lobe

Acquired data



Cook-Torrance 2 lobes



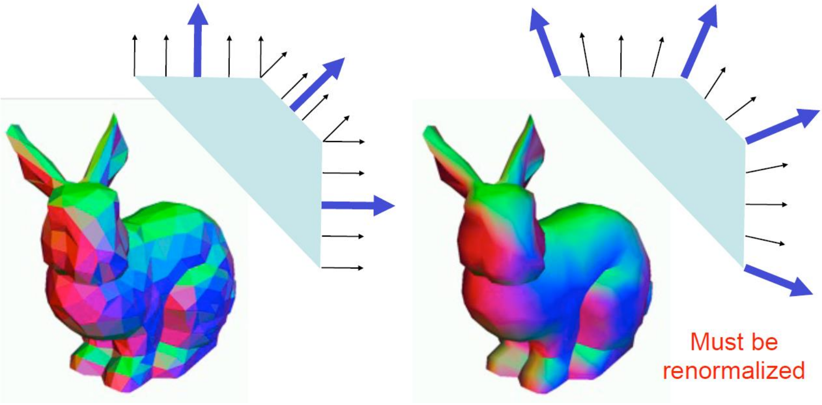
Material – Red Christmas Ball

Image-Based Acquisition

- A Data-Driven Reflectance Model, SIGGRAPH 2003
  - The data is available http://people.csail.mit.edu/wojciech/BRDFDatabase/

Phong Normal Interpolation (Not Phong *Shading*)

- Interpolate the average vertex normals across the face and use this in shading computations
  - Again, use barycentric interpolation!



That’s All for Today!

Images from the movie, “The Matrix,” removed due to copyright restrictions.