University of BRISTOL

COMS 30115

Light II

Carl Henrik Ek - carlhenrik.ek@bristol.ac.uk

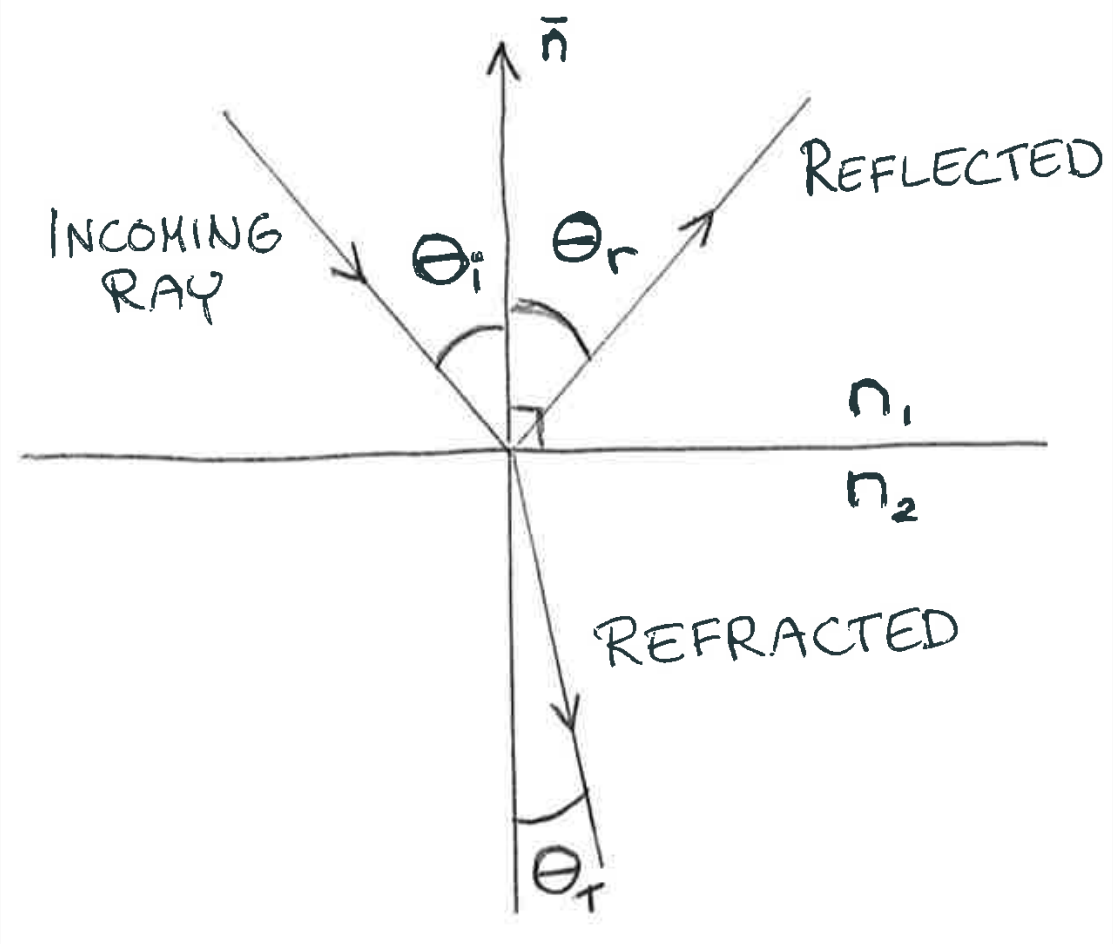
February 5th, 2018

http://www.carlhenrik.com

Last time

- Ray Surface interaction
  - Physics
- Approximations
  - Light factorisation
- Full rendering pipeline based on geometry

Last time


$\bar{n}$
INCOMING RAY
θ_i
θ_r
REFLECTED
n_1
n_2
REFRACTED
θ_t

Last time

$$
\begin{aligned}
\mathbf{i}_{tot} &= f(\mathbf{i}_{amb}, \mathbf{i}_{diff}, \mathbf{i}_{spec}) \\
&= \mathbf{m}_{emi} + \sum_{i=0}^{N-1} \Big(\mathbf{m}_{amb} \circ \mathbf{s}^i_{amb} \\
&\quad + \frac{\max((\mathbf{n}^{\mathrm{T}}\mathbf{l}^i), 0)\mathbf{m}_{diff} \circ \mathbf{s}^i_{diff} + \max((\mathbf{n}^{\mathrm{T}}\mathbf{h}^i), 0)^{m_{shi}} \mathbf{m}_{spec} \circ \mathbf{s}^i_{spec}}{s^i_c + s^i_l \left((\mathbf{s}_{pos} - \mathbf{p})^{\mathrm{T}}(\mathbf{s}_{pos} - \mathbf{p})\right)^{\frac{1}{2}} + s^i_q \left((\mathbf{s}^i_{pos} - \mathbf{p})^{\mathrm{T}}(\mathbf{s}^i_{pos} - \mathbf{p})\right)^{\frac{1}{2}}} \Big)
\end{aligned}
$$

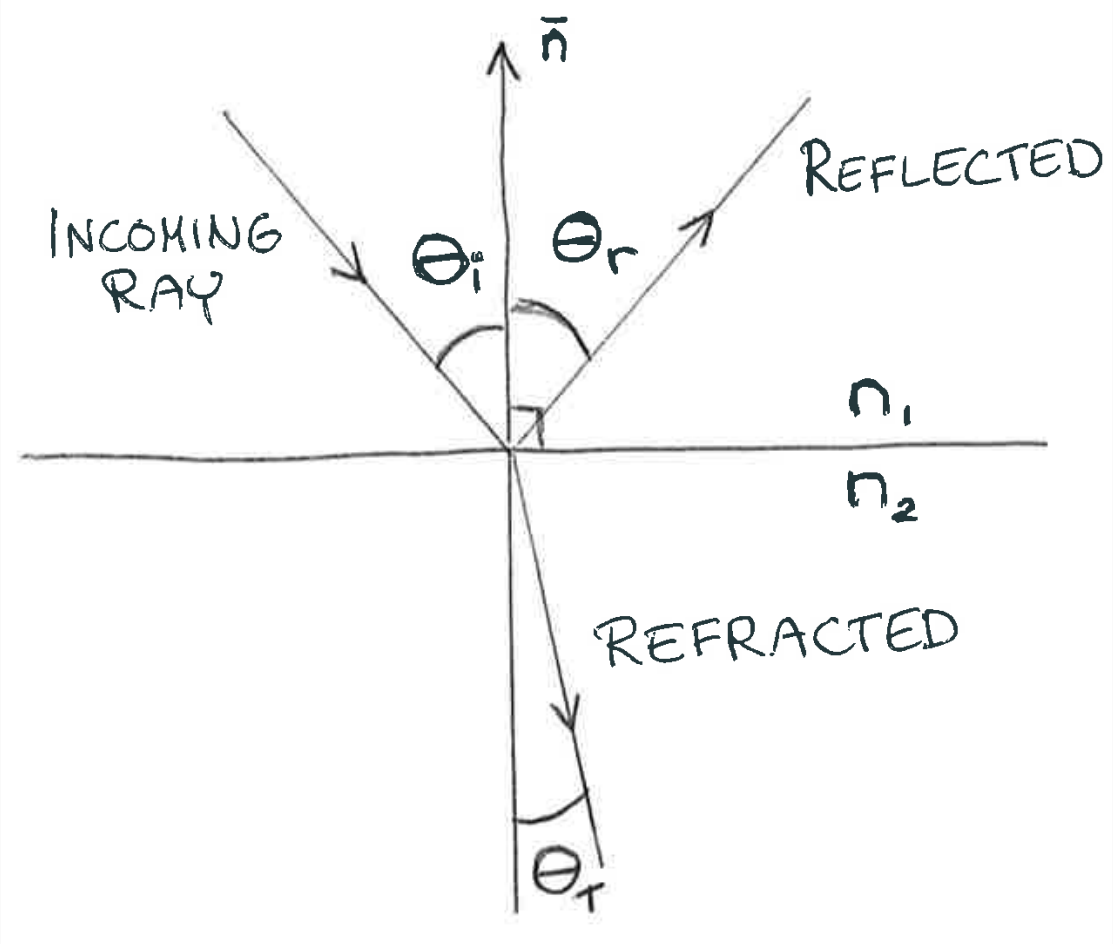
Today

- Continue with lighting
- BRDFs
    - a more realistic way to parametrise ray-surface interaction
- Approximations
    - Bump and Environment mapping

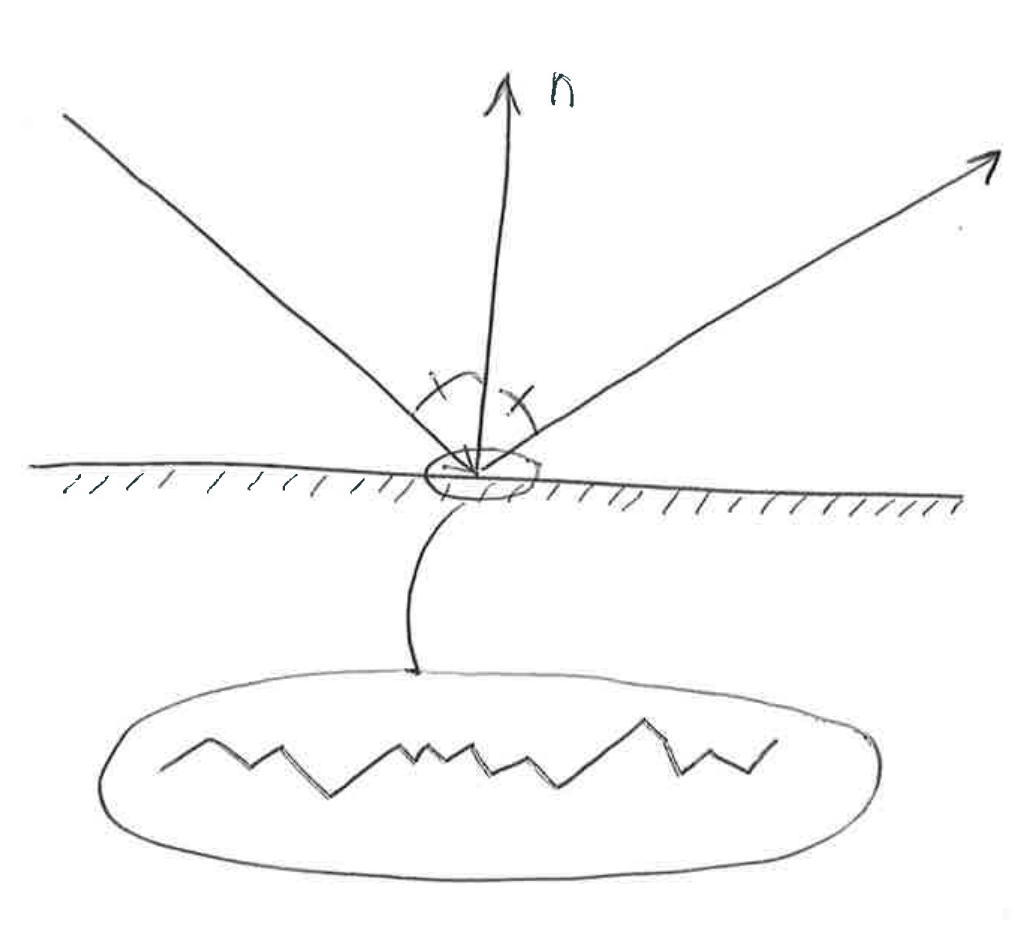
The Book

- BRDFs URL
- Bump/Normal mapping URL, URL
- ENV mapping URL, URL

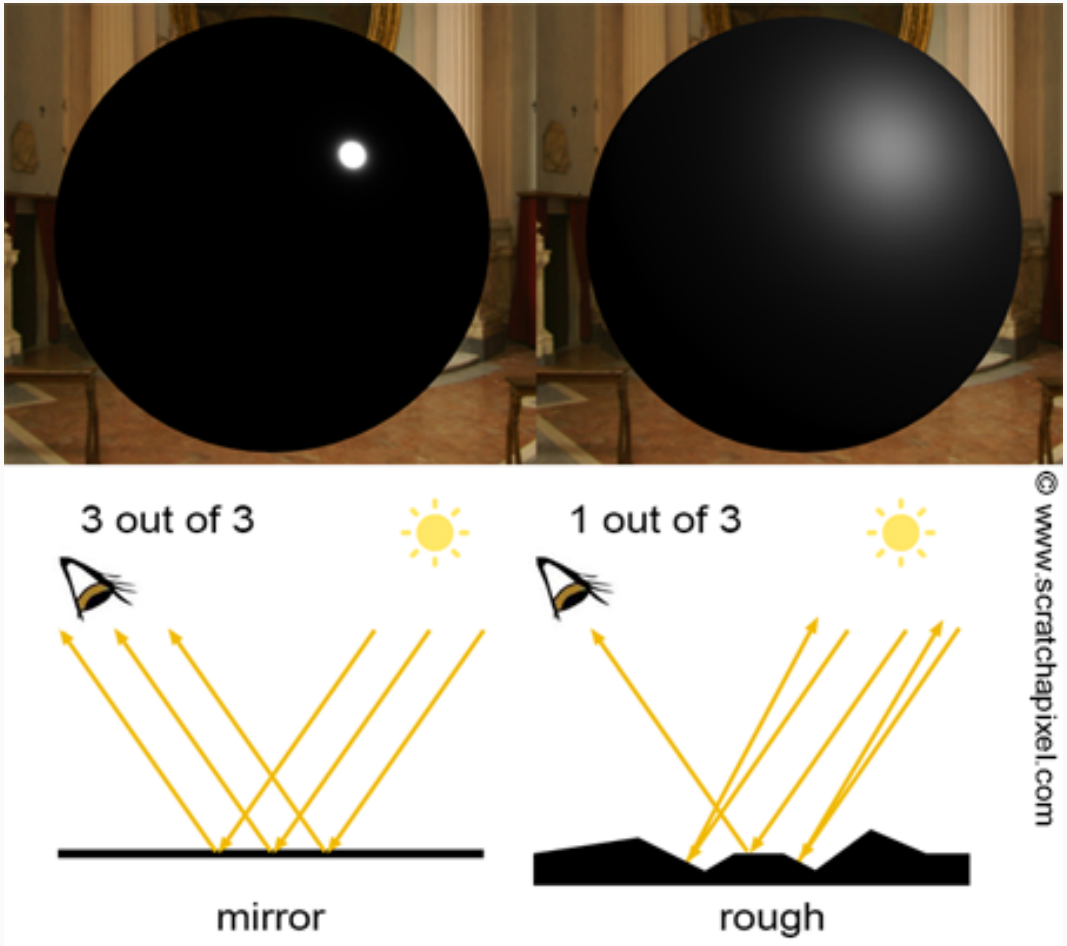
Motivation


$\bar{n}$
INCOMING RAY
θ_i
θ_r
REFLECTED
n_1
n_2
REFRACTED
θ_t

Motivation



Motivation



BRDF

Bi-Directional Reflectance Distribution Function

- We want to know the reflection
- Reflection depends on *incoming* and *view* direction
- Micro-facet structure is "unknown"
  - reflection is a "stochastic" process
- Phong model

$$f(\mathbf{L}, \mathbf{V})$$

Bi-Directional Reflectance Distribution Function

BRDF - Bi-Directional Reflectance Distribution Function

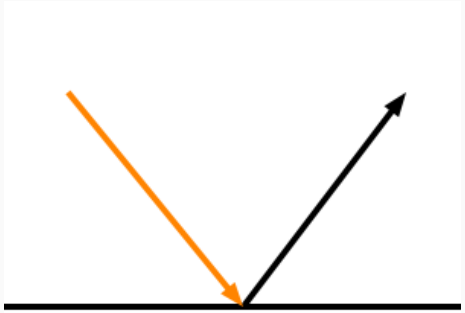
$$f(\mathbf{L}, \mathbf{V})$$

Positive: $f(\mathbf{L}, \mathbf{V}) \geq 0, \forall\{\mathbf{L}, \mathbf{V}\}$

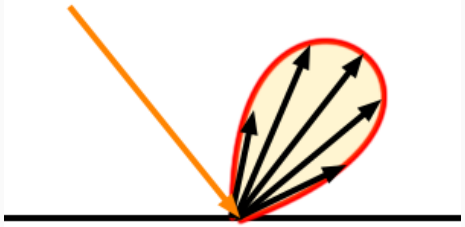
Reciprocal: $f(\mathbf{L}, \mathbf{V}) = f(\mathbf{V}, \mathbf{L})$

Energy Conserving: $\int f(\mathbf{L}, \mathbf{V}) \mathrm{d}\mathbf{L} \mathrm{d}\mathbf{V} = 1$

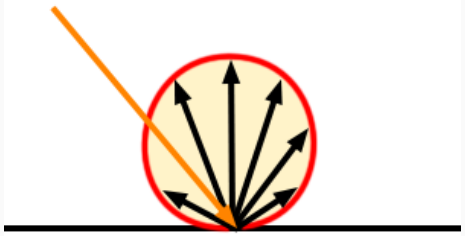
Completely characterises materials that have *no refracting part*



mirror reflection

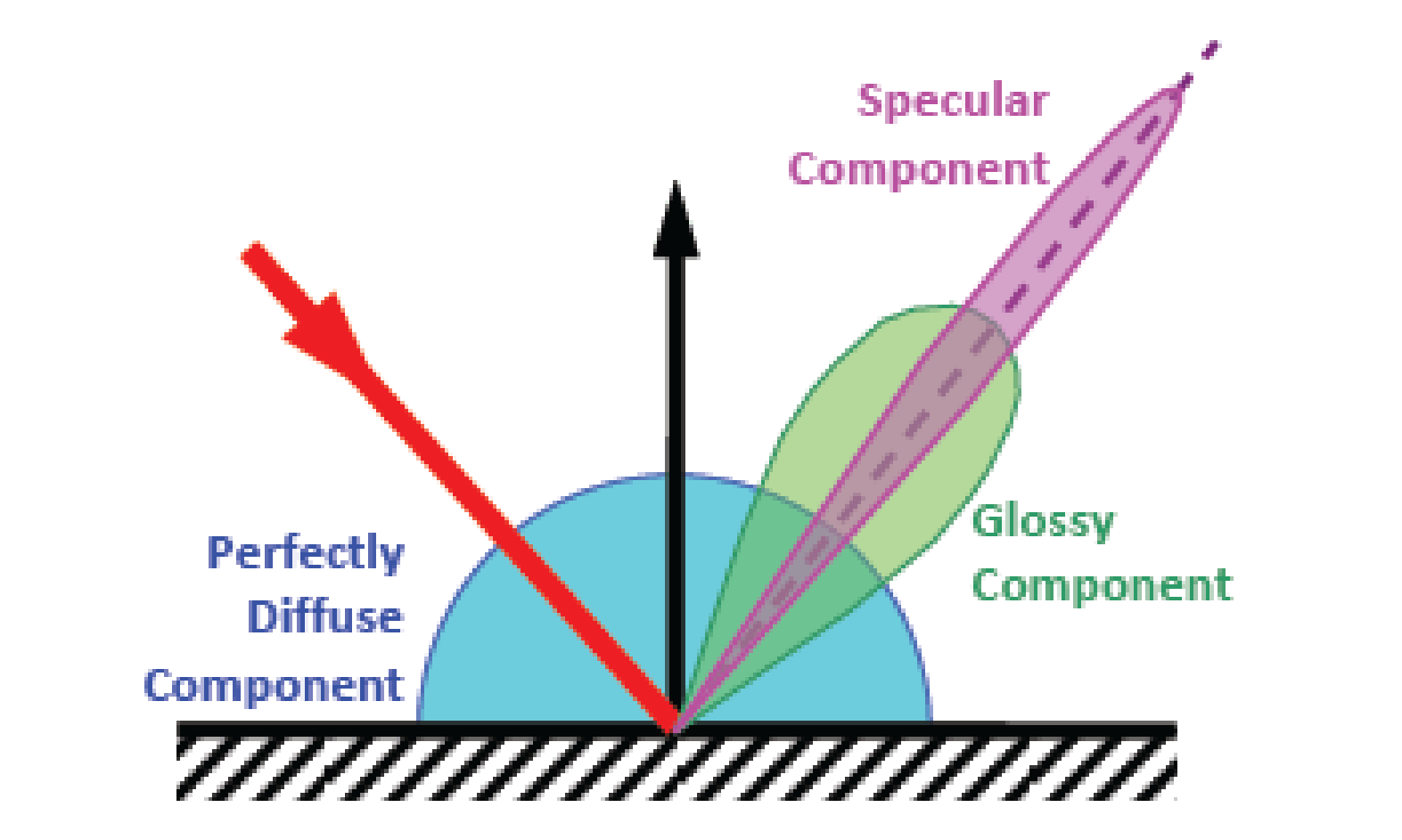


specular reflection



difuse reflection

Bi-Directional Reflectance Distribution Function

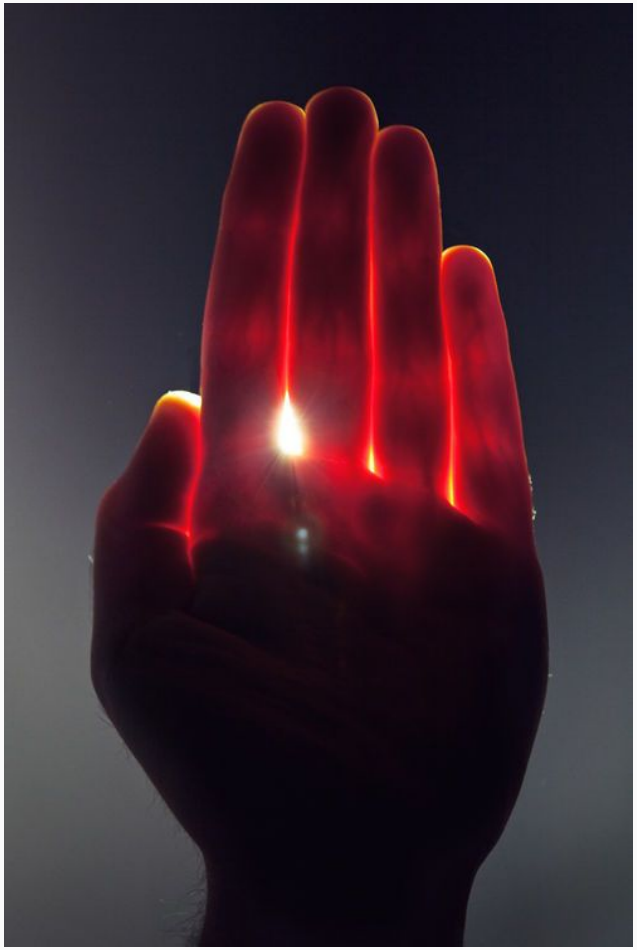


Bi-Directional Reflectance Distribution Function

Final Fantasy - The Spirit Within

Shrek

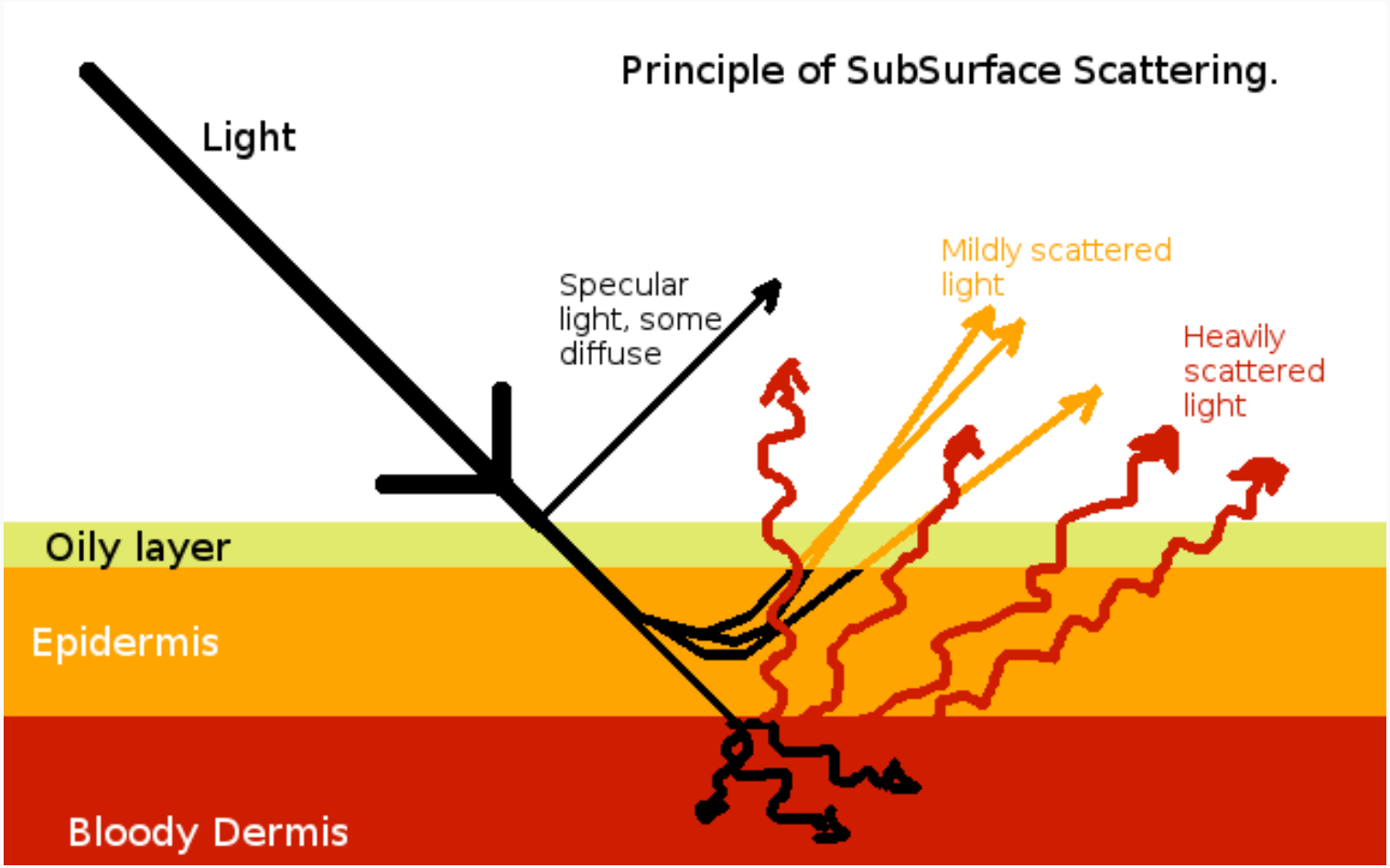
Bi-Directional Reflectance Distribution Function



Bi-Directional Reflectance Distribution Function

Bi-Directional Reflectance Distribution Function

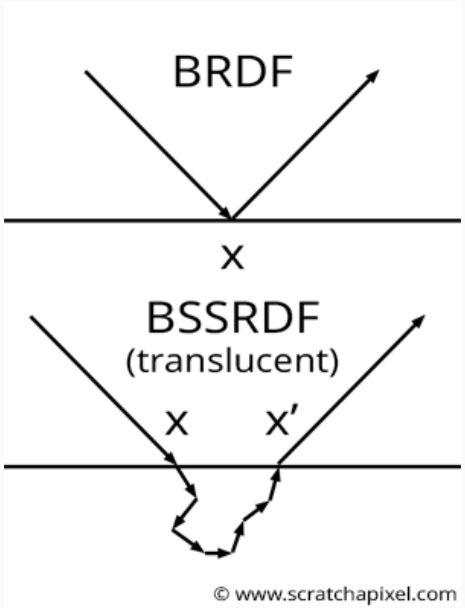
Bi-Directional Reflectance Distribution Function



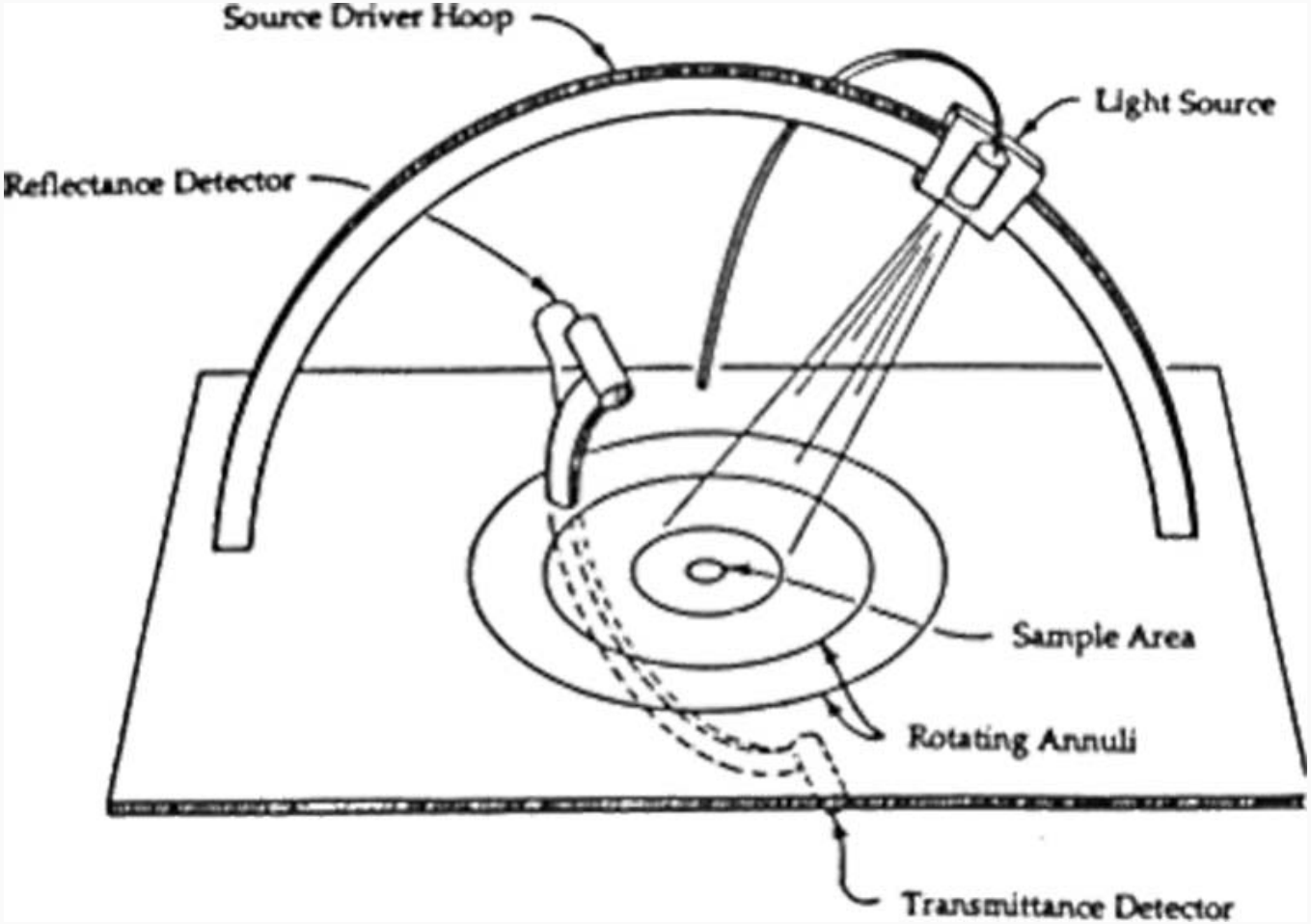
Bi-Directional Reflectance Distribution Function

BRSSDF - Bi-Directional Scattering-Surface Reflectance Distribution Function

- Very different from BRDFs
- Not all light arriving at point **p** leaves at **p**
- Needs two locations, i.e. you cannot shade each component independently



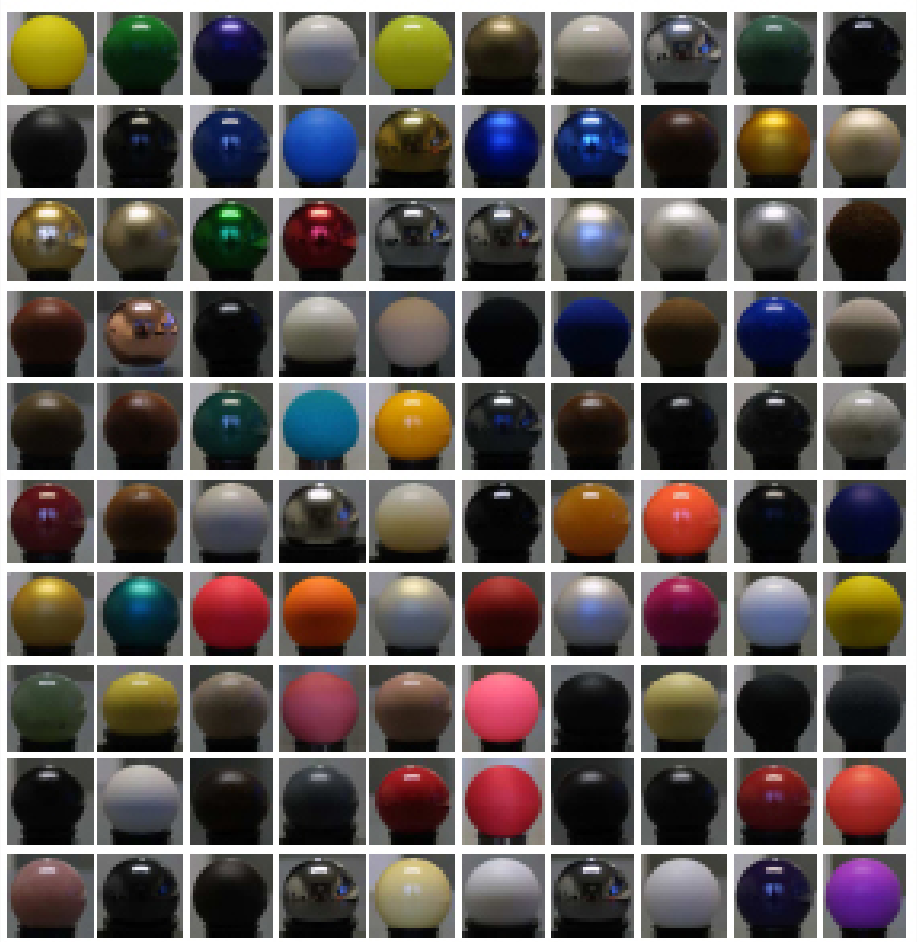
Aquiring BRDFs



1

[1] URL
[2] MERL Database URL

Aquiring BRDFs



2

[1] URL

[2] MERL Database URL

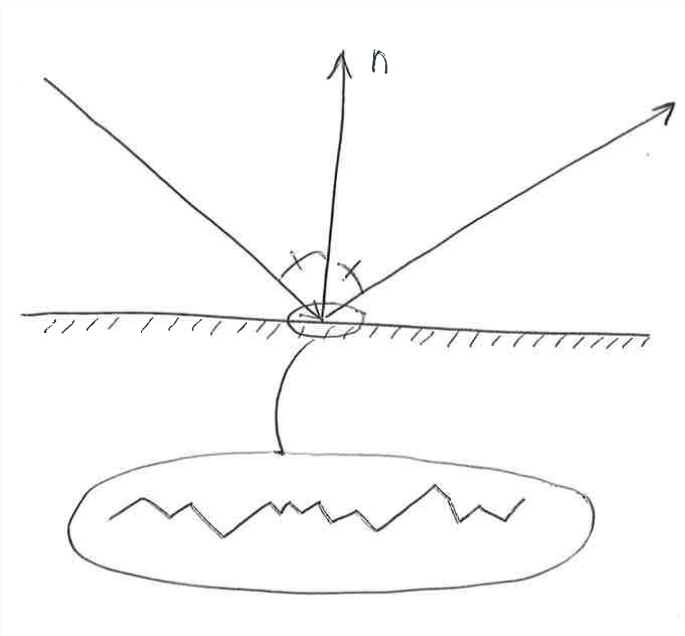
How to use BRDF

- the BRDF completely characterises the reflective properties of the surface
- we know the light direction of the incoming light
- we know the direction we are looking at the surface from
- this allows us to compute how much of the light is reflected
- *you can add BRDFs to your raytracer*

Normal mapping

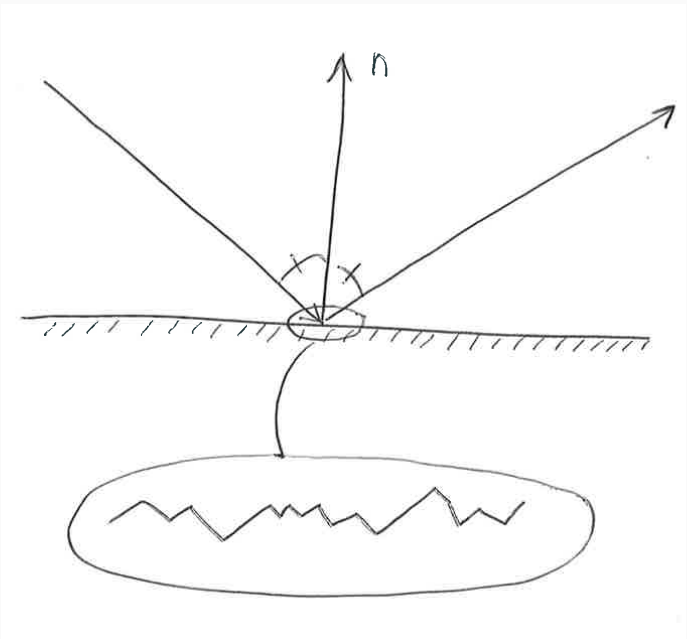
Introduction

- Glossy materials are small specular micro-facets
- Geometrically model facets expensive
  - many many more triangles
  - and intersections and ...



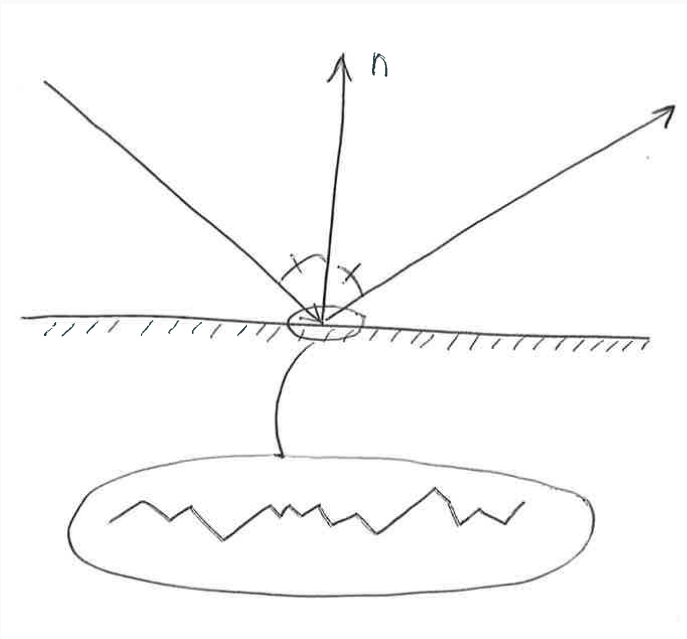
Introduction

- Glossy materials are small specular micro-facets
- Geometrically model facets expensive
  - many many more triangles
  - and intersections and ...
- Idea: Alter shading instead



Introduction

- Glossy materials are small specular micro-facets
- Geometrically model facets expensive
  - many many more triangles
  - and intersections and ...
- Idea: Alter shading instead
  - normal pertubation



Normalmap

- Surface: $\mathbf{P}(u, v)$
- Surface Normal in tangent plane

$$\mathbf{N} = \frac{\delta \mathbf{P}}{\delta u} \times \frac{\delta \mathbf{P}}{\delta v} = \mathbf{P}_u \times \mathbf{P}_v$$

Normalmap

- Surface: $\mathbf{P}(u, v)$
- Surface Normal in tangent plane

$$\mathbf{N} = \frac{\delta \mathbf{P}}{\delta u} \times \frac{\delta \mathbf{P}}{\delta v} = \mathbf{P}_u \times \mathbf{P}_v$$

- Normal, Bump or Displacement map $B(u, v)$

Normalmap

- Surface: $\mathbf{P}(u, v)$
- Surface Normal in tangent plane

$$\mathbf{N} = \frac{\delta \mathbf{P}}{\delta u} \times \frac{\delta \mathbf{P}}{\delta v} = \mathbf{P}_u \times \mathbf{P}_v$$

- Normal, Bump or Displacement map $B(u, v)$
- Displace surface point: $\tilde{\mathbf{P}}(u, v) = \mathbf{P}(u, v) + B(u, v)\mathbf{N}$

Normalmap

- Surface: $\mathbf{P}(u, v)$
- Surface Normal in tangent plane

$$\mathbf{N} = \frac{\delta \mathbf{P}}{\delta u} \times \frac{\delta \mathbf{P}}{\delta v} = \mathbf{P}_u \times \mathbf{P}_v$$

- Normal, Bump or Displacement map $B(u, v)$
- Displace surface point: $\tilde{\mathbf{P}}(u, v) = \mathbf{P}(u, v) + B(u, v)\mathbf{N}$
- Compute normal of displaced surface

$$\begin{aligned}\tilde{\mathbf{N}} = \mathbf{P}_u + B_u\mathbf{N} + B(u, v)\mathbf{N}_u \\ + \mathbf{P}_v + B_v\mathbf{N} + B(u, v)\mathbf{N}_v\end{aligned}$$

Normalmap

- Surface: $\mathbf{P}(u, v)$
- Surface Normal in tangent plane

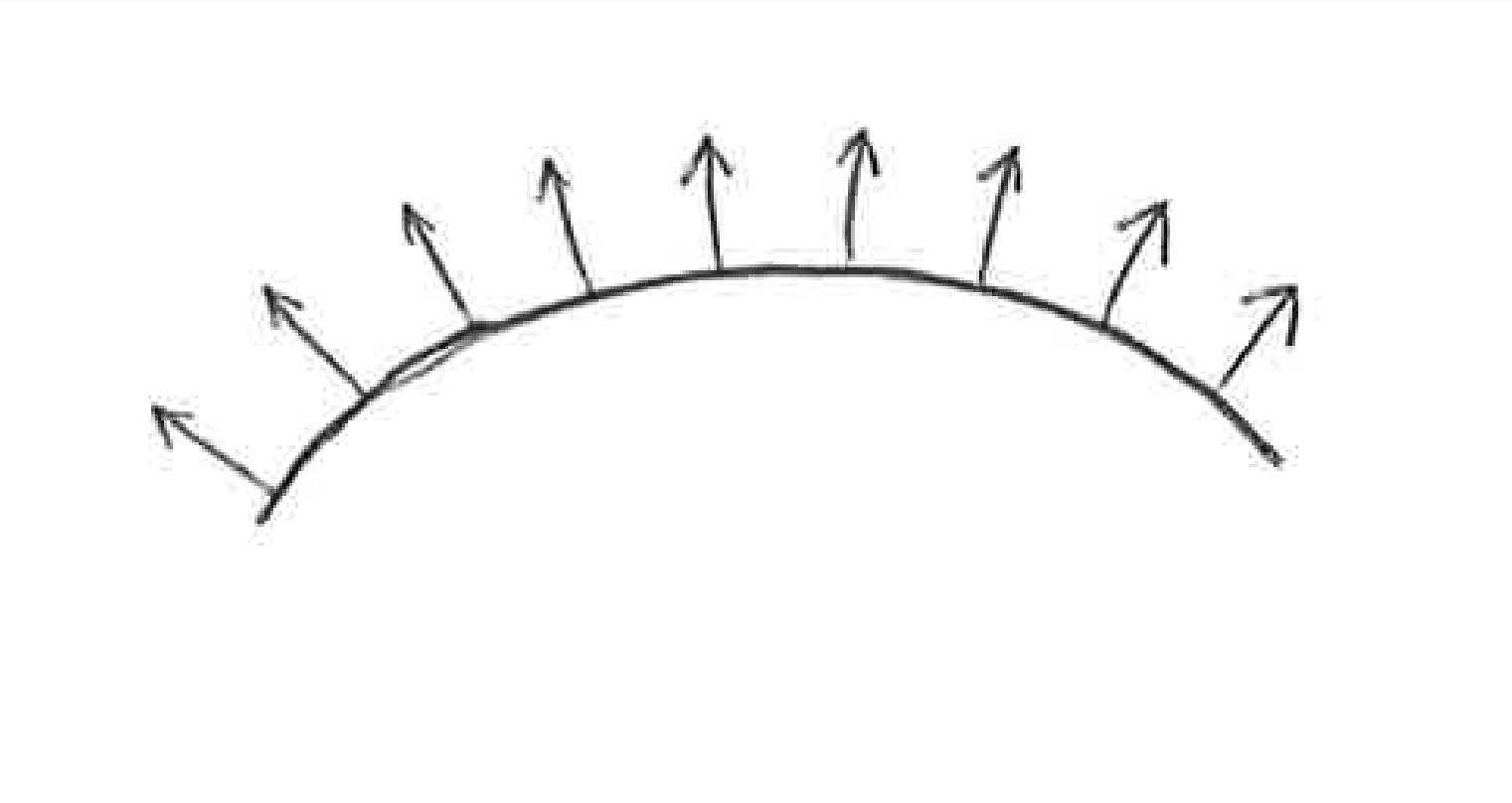
$$\mathbf{N} = \frac{\delta \mathbf{P}}{\delta u} \times \frac{\delta \mathbf{P}}{\delta v} = \mathbf{P}_u \times \mathbf{P}_v$$

- Normal, Bump or Displacement map $B(u, v)$
- Displace surface point: $\tilde{\mathbf{P}}(u, v) = \mathbf{P}(u, v) + B(u, v)\mathbf{N}$
- Compute normal of displaced surface

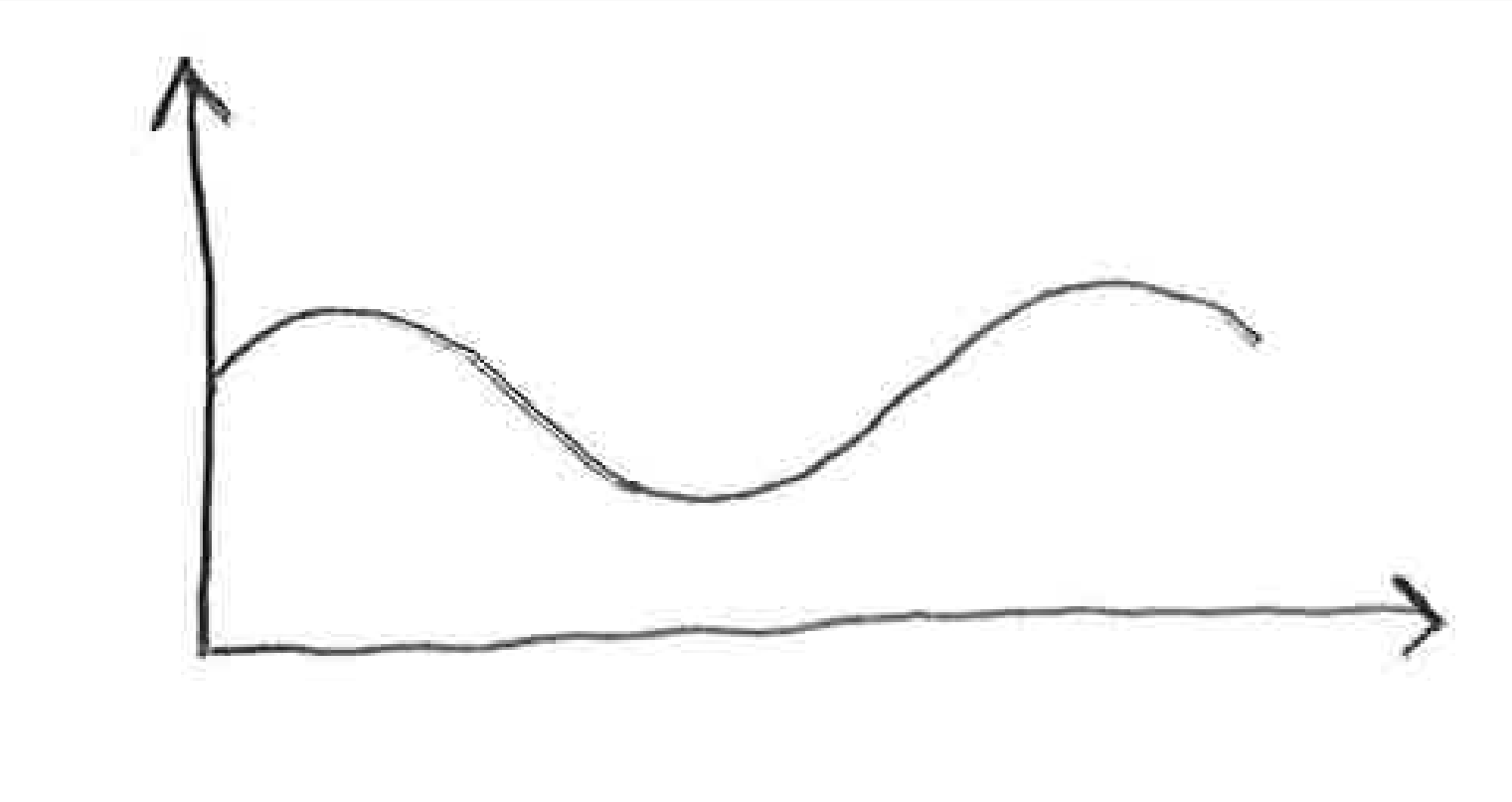
$$\begin{aligned}\tilde{\mathbf{N}} = \mathbf{P}_u + B_u\mathbf{N} + B(u, v)\mathbf{N}_u \\ + \mathbf{P}_v + B_v\mathbf{N} + B(u, v)\mathbf{N}_v\end{aligned}$$

- Shade with new normal

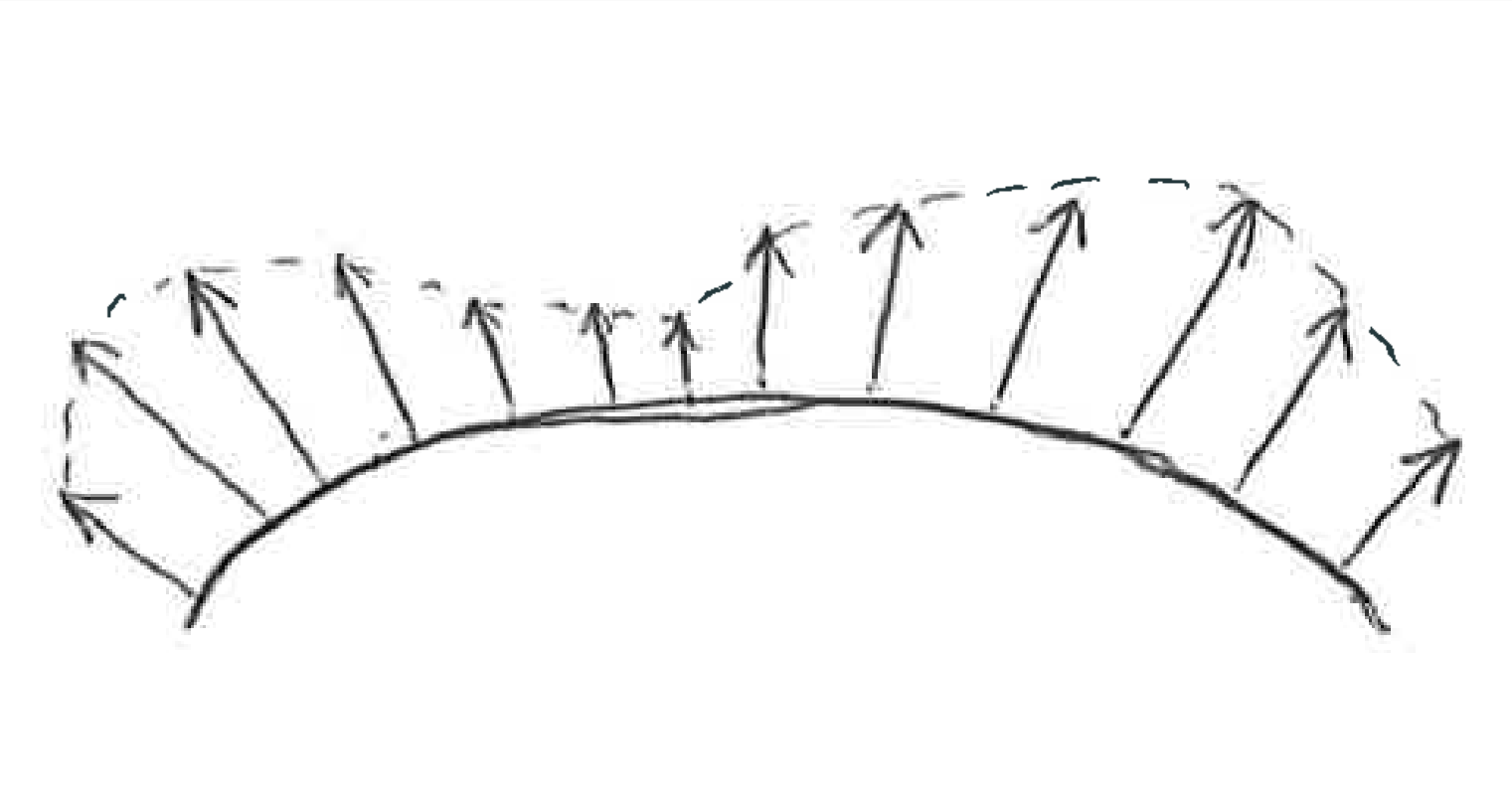
Normalmap



Normalmap

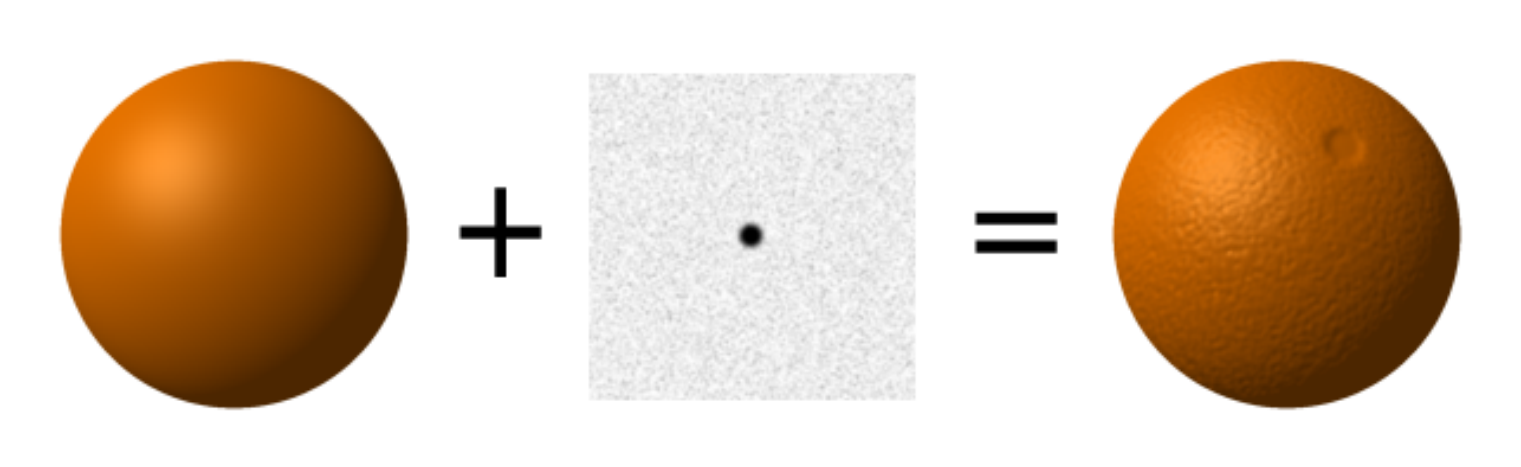


Normalmap



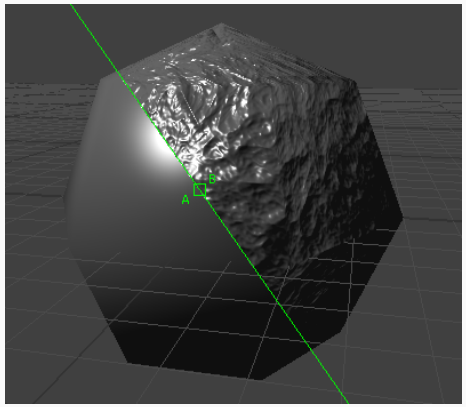
Normalmap

Normalmap



Normalmap

- Alter normals at shading step
- Cheap way to increase realism
- Lots of pre-computations possible
  - make look-up table
- avoid when contour dominant
- *there are lots of other things that you can alter with a map than normals*



Bump mapping

- Convert texture to grayscale

  ```
  convert <image_in> -set colorspace Gray -separate -avera
  ```

- Compute bump map

  $$B(u,v) = \{I(u+1,v) - I(u-1,v), I(u,v+1) - I(u,v-1)\}$$

- Shade

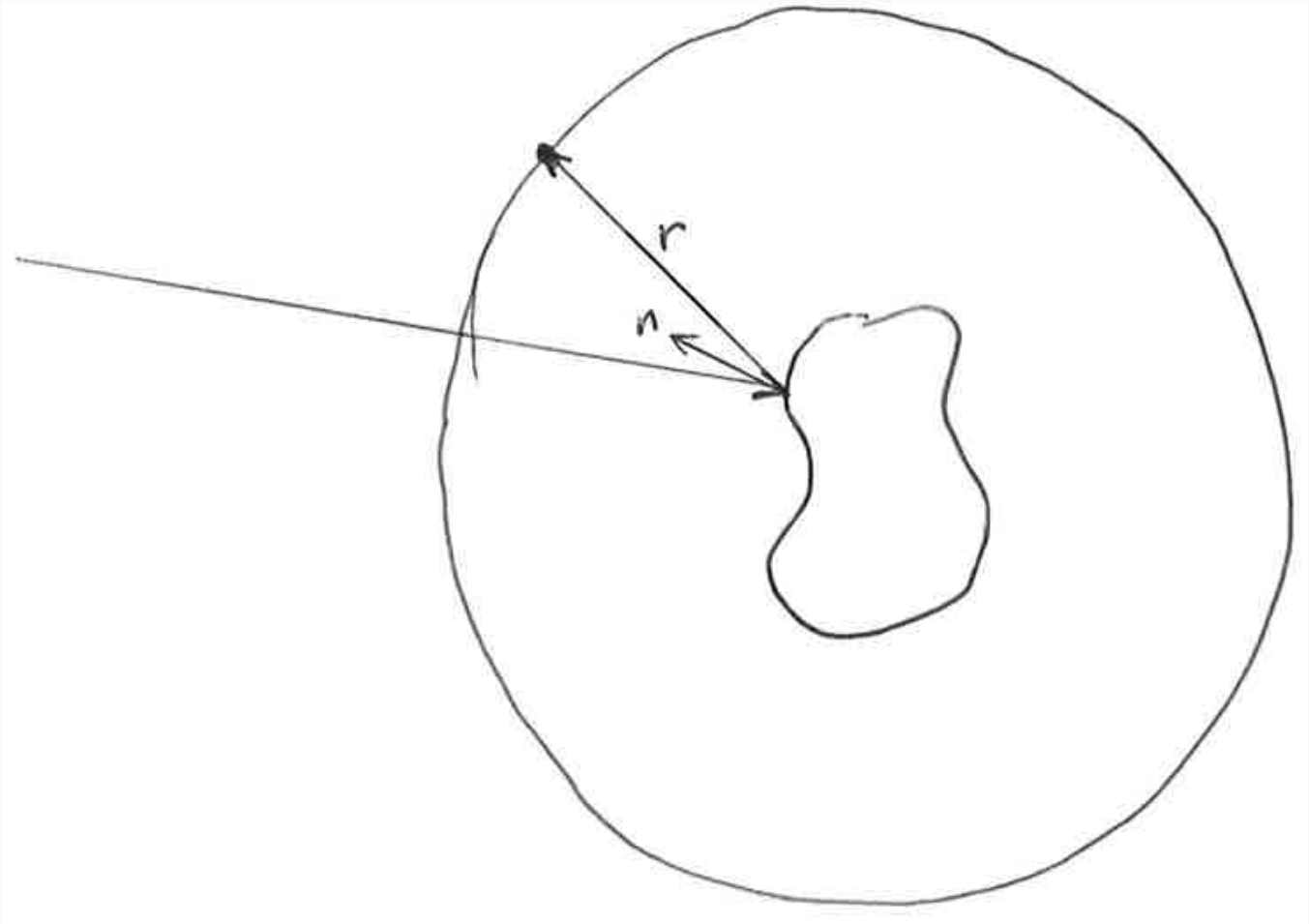
  $$\tilde{N}(u,v) = N + B(u,v)$$

- *Not correct but you can tune the amplitude to work well*

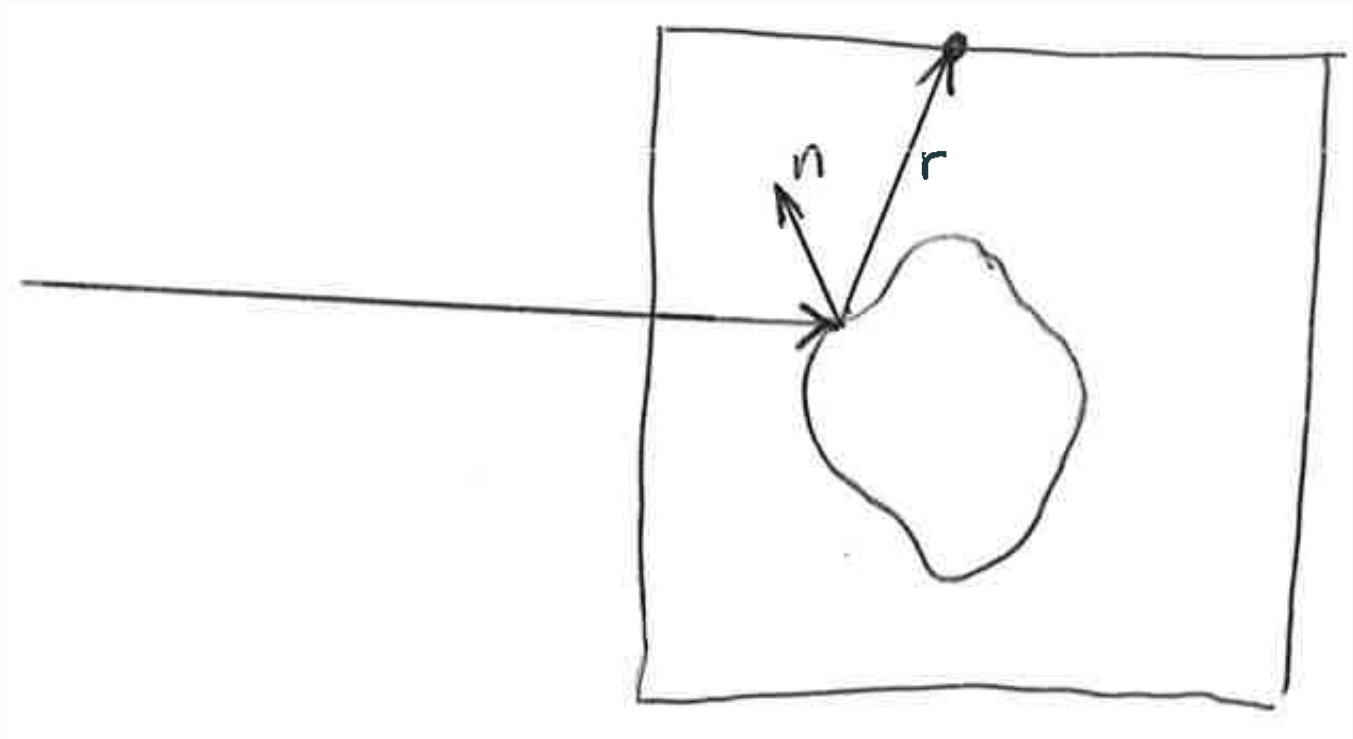
Env-Mapping

- Simple way to render shiny/reflective objects
- Often used in image-based rendering
- Secondary rays requires intersection calculations
- Sky-box

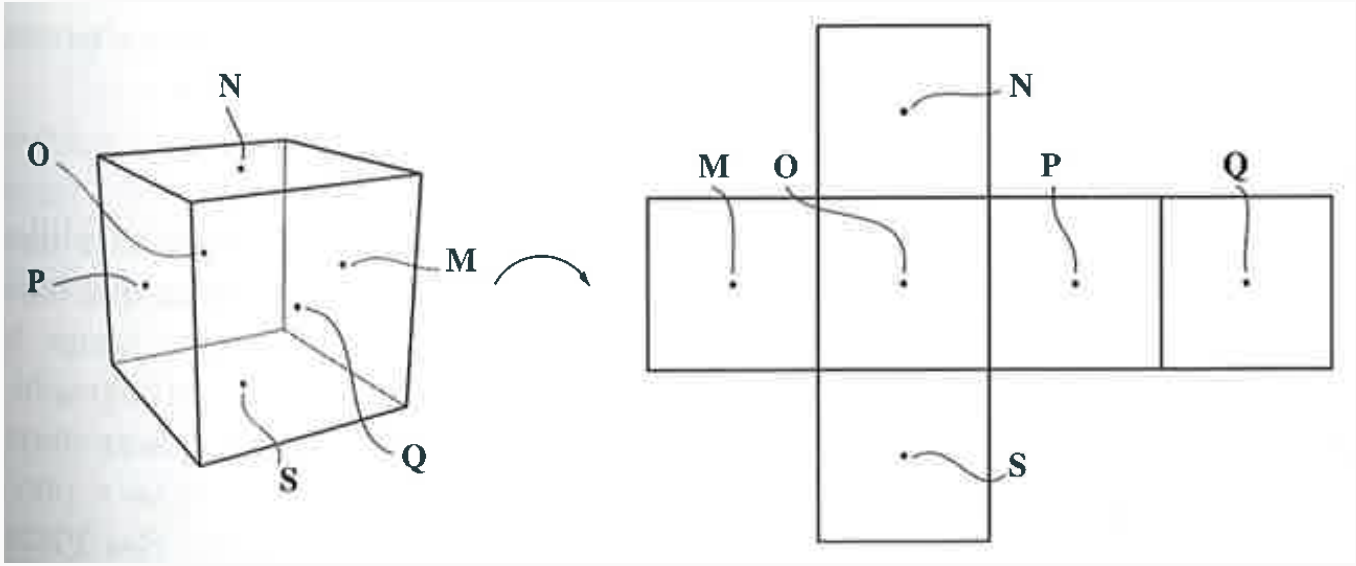
Env-Mapping



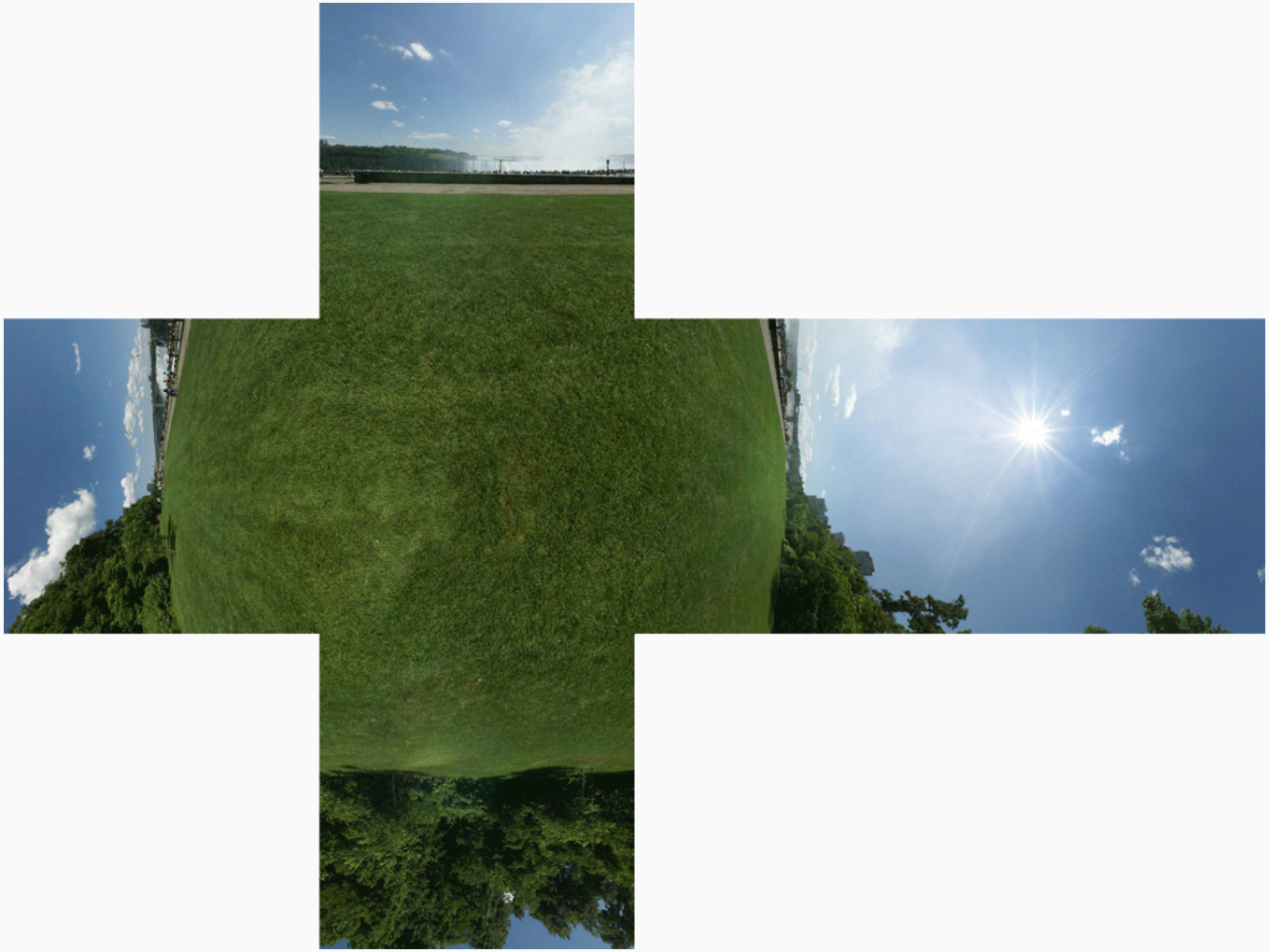
Env-Mapping



Env-Mapping



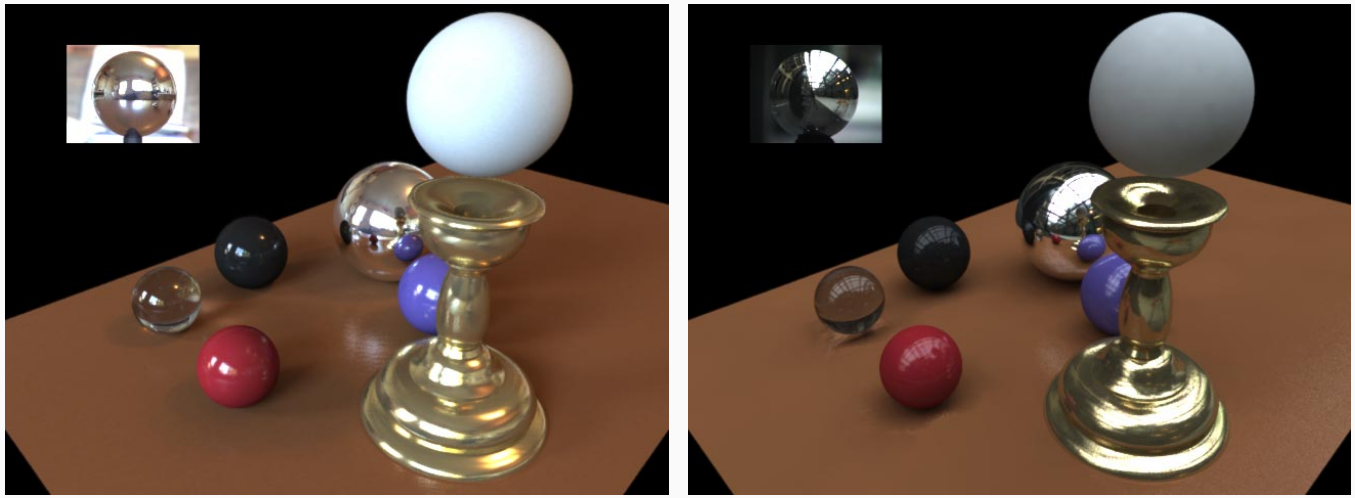
Env-Mapping



Capturing ENV-Maps[3]

[3]Paul Debevec, 1998

Capturing ENV-Maps[3]



[3] Paul Debevec, 1998

ENV-Mapping

- Compute reflection direction
  - $[-3.2, 5.1, -8.4]^{\mathrm{T}}$
- Largest magnitude determines face
  - eg. Negative $Z - \textit{face}$

ENV-Mapping

- Compute reflection direction
  - $[-3.2, 5.1, -8.4]^{\mathrm{T}}$
- Largest magnitude determines face
  - eg. Negative $Z - face$
- Find intersection point (eg. $z = -1$)
  - $[\frac{-3.2}{8.4}, \frac{5.1}{8.4}, -1]^{\mathrm{T}}$

ENV-Mapping

- Compute reflection direction
  - $[-3.2, 5.1, -8.4]^{\mathrm{T}}$
- Largest magnitude determines face
  - eg. Negative $Z - face$
- Find intersection point (eg. $z = -1$)
  - $[\frac{-3.2}{8.4}, \frac{5.1}{8.4}, -1]^{\mathrm{T}}$
- Re-map to texture coordinates

ENV-Mapping

- Compute reflection direction
  - $[-3.2, 5.1, -8.4]^{\mathrm{T}}$
- Largest magnitude determines face
  - eg. Negative Z – *face*
- Find intersection point (eg. $z = -1$)
  - $[\frac{-3.2}{8.4}, \frac{5.1}{8.4}, -1]^{\mathrm{T}}$
- Re-map to texture coordinates
- *Notice how reflection vector does not need to be normalised*

  $\Rightarrow$ *no square-root*

ENV-Mapping

Summary

Summary

- How to parametrise reflections
- Normal mapping
- Environment mapping
- *ideas of hacks*

Raytracing

- Know your two weapons
  - Inner product

$$\mathbf{x}^{\mathrm{T}}\mathbf{y} = ||\mathbf{x}||||\mathbf{y}||\cos(\theta)$$

  - Outer product

$$\mathbf{x} \times \mathbf{y} = \mathbf{z}$$

$$\mathbf{z} \perp \mathbf{x}$$

$$\mathbf{z} \perp \mathbf{y}$$

  - Think how things work physically
    - how can we "mimick" this behaviour?
  - *Thats rendering for you*

Next Time

Next Time

Lecture Friday 9th of February

- Shadows
- Realistic Cameras
- Optimizations
- Finish Raytracing

Lab Continue with Lab 1

eof