University of BRISTOL

COMS 30115

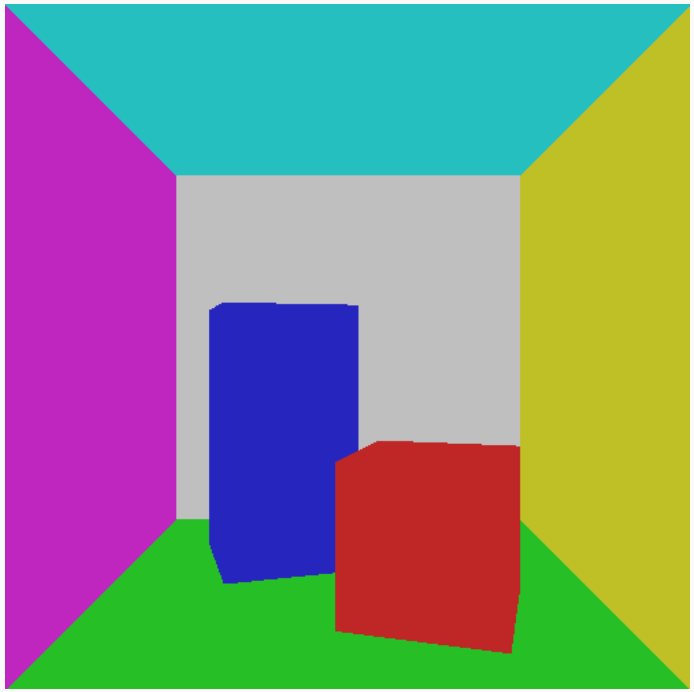
Light II

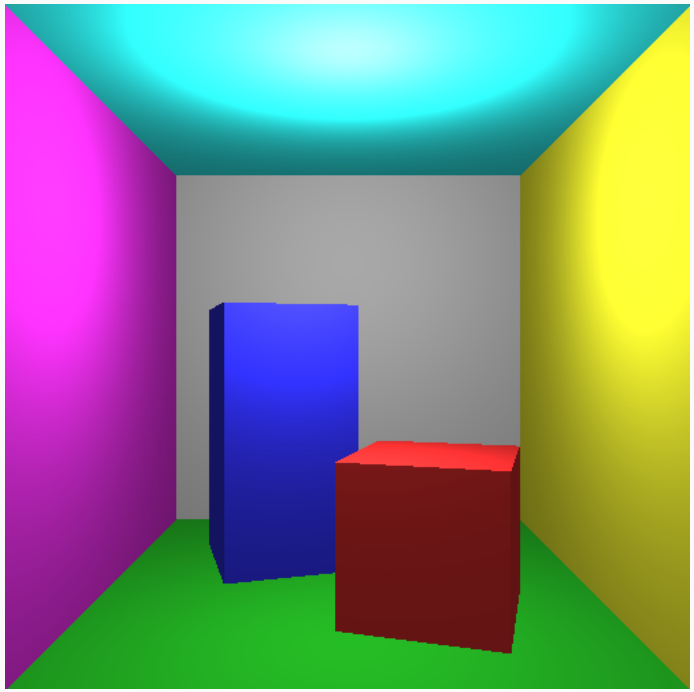
Carl Henrik Ek - carlhenrik.ek@bristol.ac.uk

February 15th, 2019

http://www.carlhenrik.com

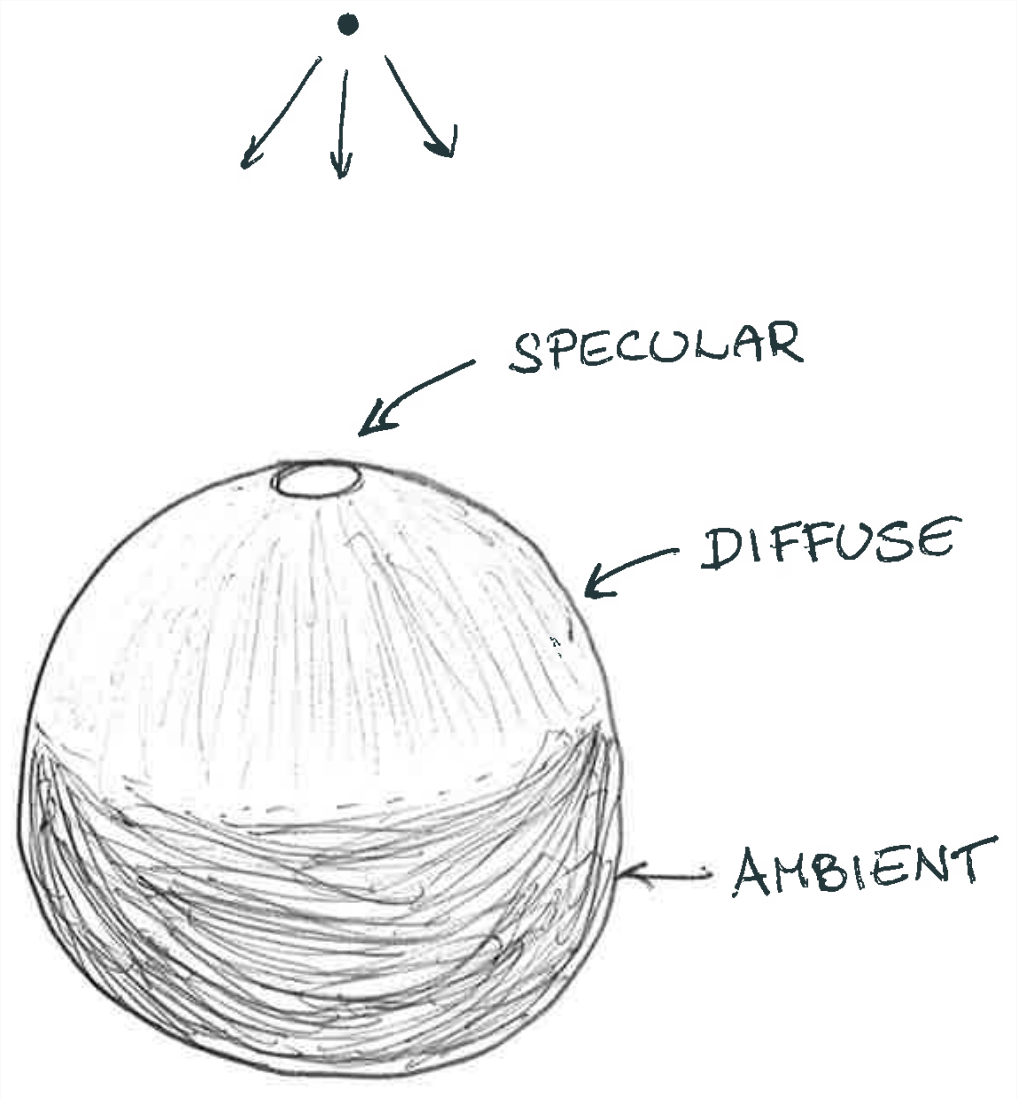
Where are we





Lights

Lights Ball



Light Factorisation

$$\mathbf{i}_{tot} = f(\mathbf{i}_{amb}, \mathbf{i}_{diff}, \mathbf{i}_{spec})$$

- There is only one type of light
- Approximation: Factorise into Ambient, Diffuse and Specular

Colour

$$\mathbf{i} = \mathbf{m} \circ \mathbf{s}$$

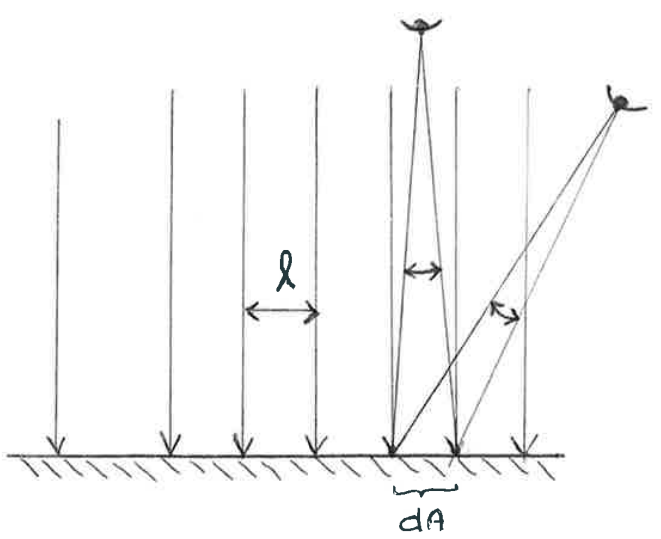
- $\mathbf{s}$ light intensity
- $\mathbf{m}$ material properties
- $\mathbf{i}$ colour of reflected light

Parametrisation

Notation	Description
$\mathbf{s}_{amb}$	Ambient intensity
$\mathbf{s}_{diff}$	Diffuse intensity
$\mathbf{s}_{spec}$	Specular intensity
$\mathbf{s}_{pos}$	Light source position

Notation	Description
$\mathbf{m}_{amb}$	Ambient material
$\mathbf{m}_{diff}$	Diffuse material
$\mathbf{m}_{spec}$	Specular material
m_{shi}	"Shininess"
$\mathbf{m}_{emi}$	Emitting

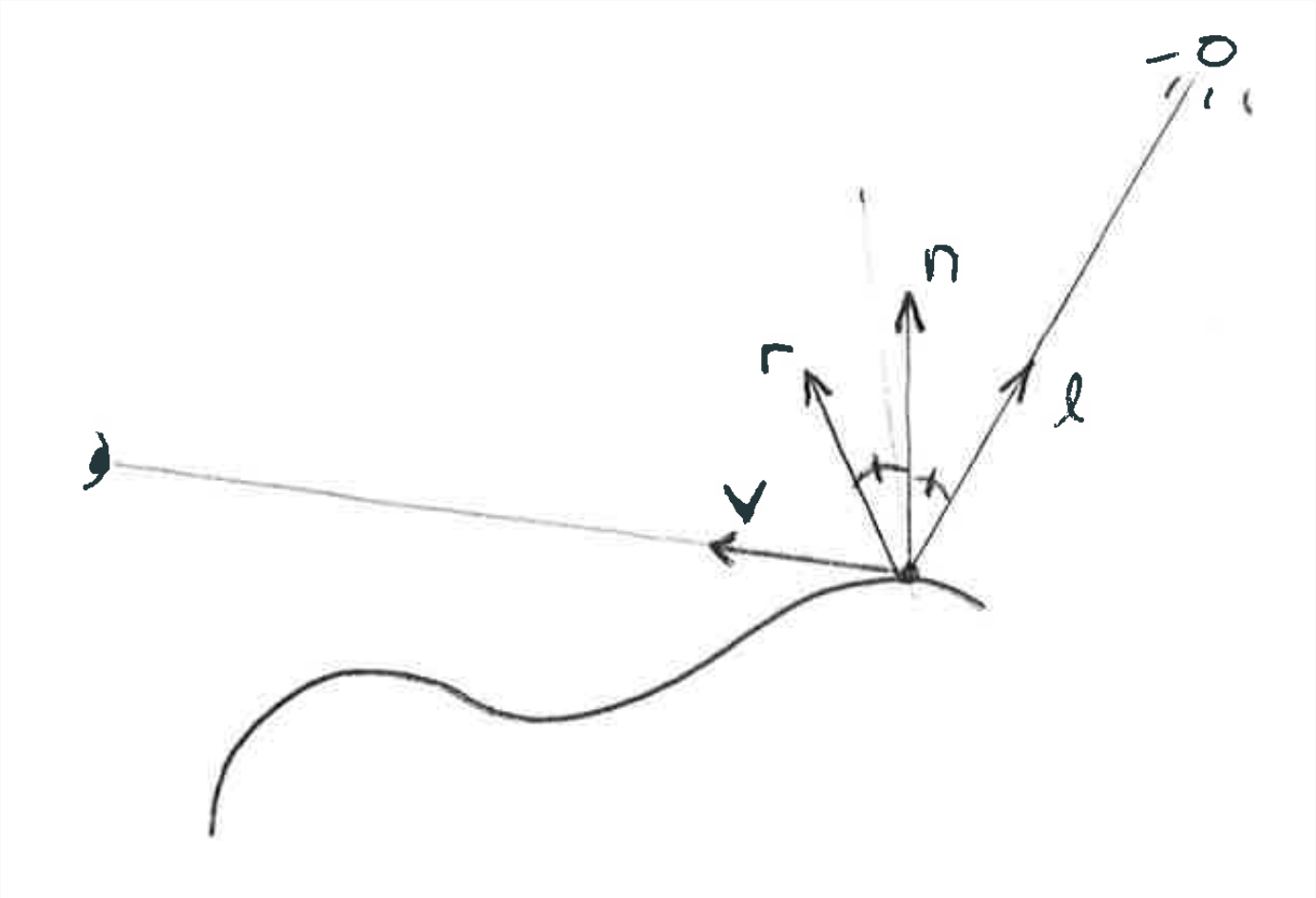
Diffuse/Lambertian



- View independent
- Lambertian Surface (Looks the same from all directions)

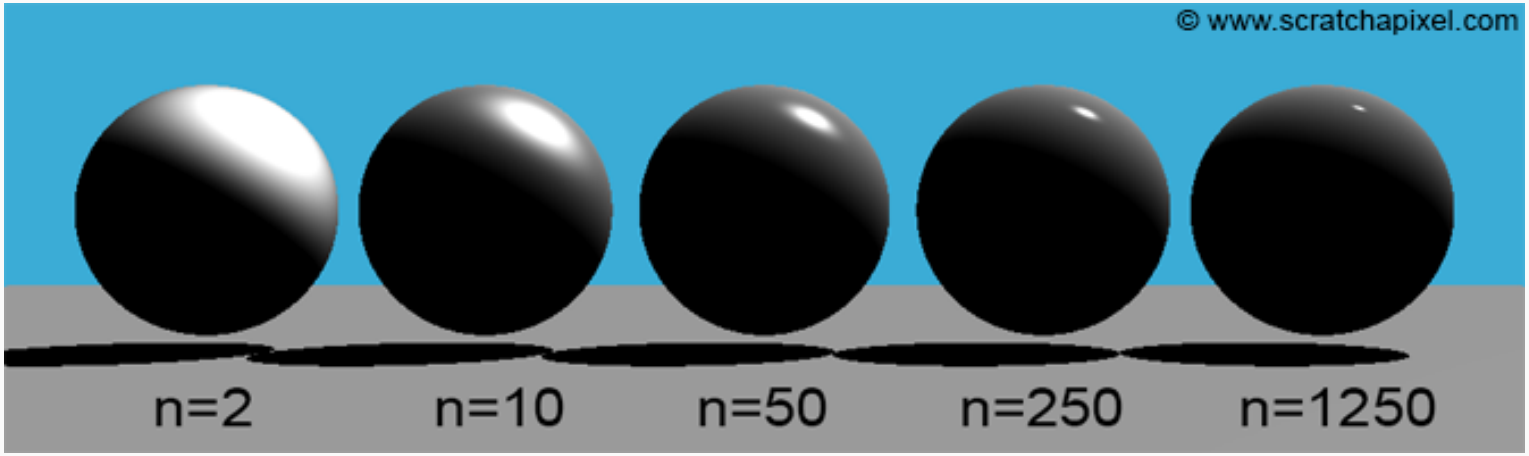
$$\mathbf{i}_{diff} = \max\left((0, \mathbf{n}^{\mathrm{T}}\mathbf{l})\right) \mathbf{m}_{diff} \circ \mathbf{s}_{diff}$$

Specular

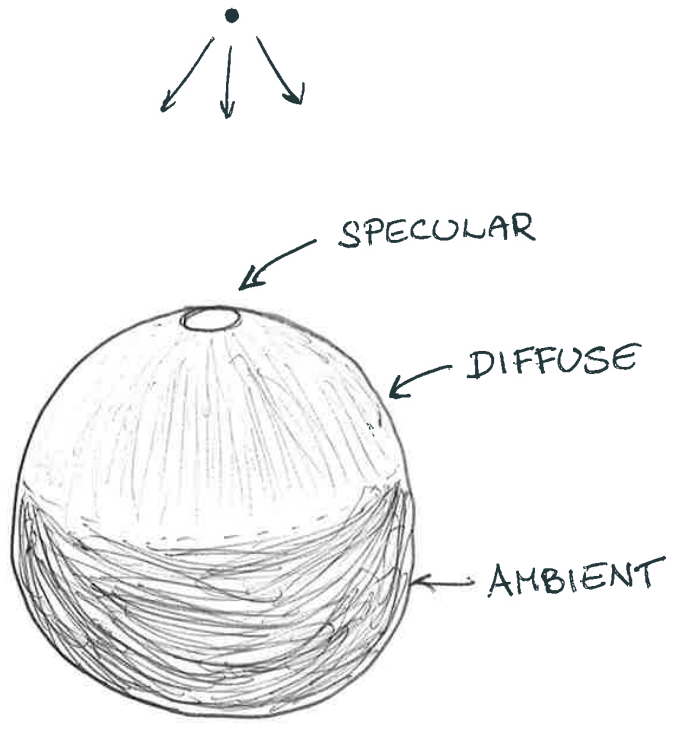


$$\mathbf{i}_{spec} = (\mathbf{r}^{\mathrm{T}}\mathbf{v})^{m_{shi}}\,\mathbf{m}_{spec} \circ \mathbf{s}_{spec}$$

Specular

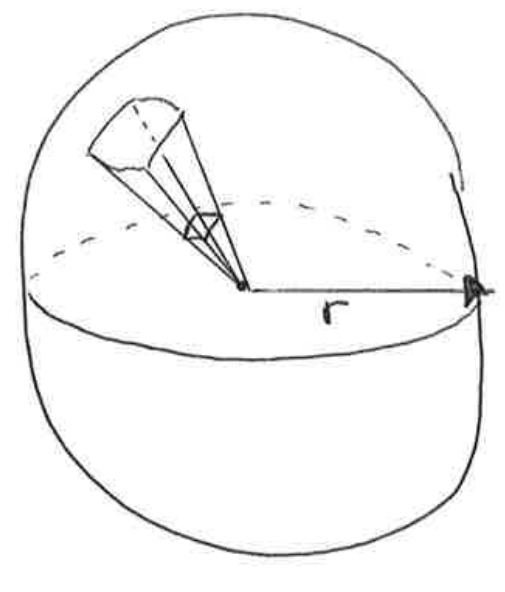


Ambient



$$\mathbf{i}_{amb} = \mathbf{m}_{amb} \circ \mathbf{s}_{amb}$$

Distance Attenuation



$$d = \left(s_c + s_l \cdot r + s_q \cdot r^2\right)^{-1}$$

$$r = \left((\mathbf{s}_{pos} - \mathbf{p})^{\mathrm{T}}(\mathbf{s}_{pos} - \mathbf{p})\right)^{\frac{1}{2}}$$

All Together

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

Code

```
/*Per Pixel*/
/*compute primary ray*/
for(int i=0;i<N_primitives;i++)
  {
    /*1. compute intersection
      2. check closest*/
  }
glm::vec3 i_tot = glm::vec3(0,0,0);
for(int i=0;i<N_lights;i++)
  {
    /*compute light*/
    i_tot += ;
  }
```

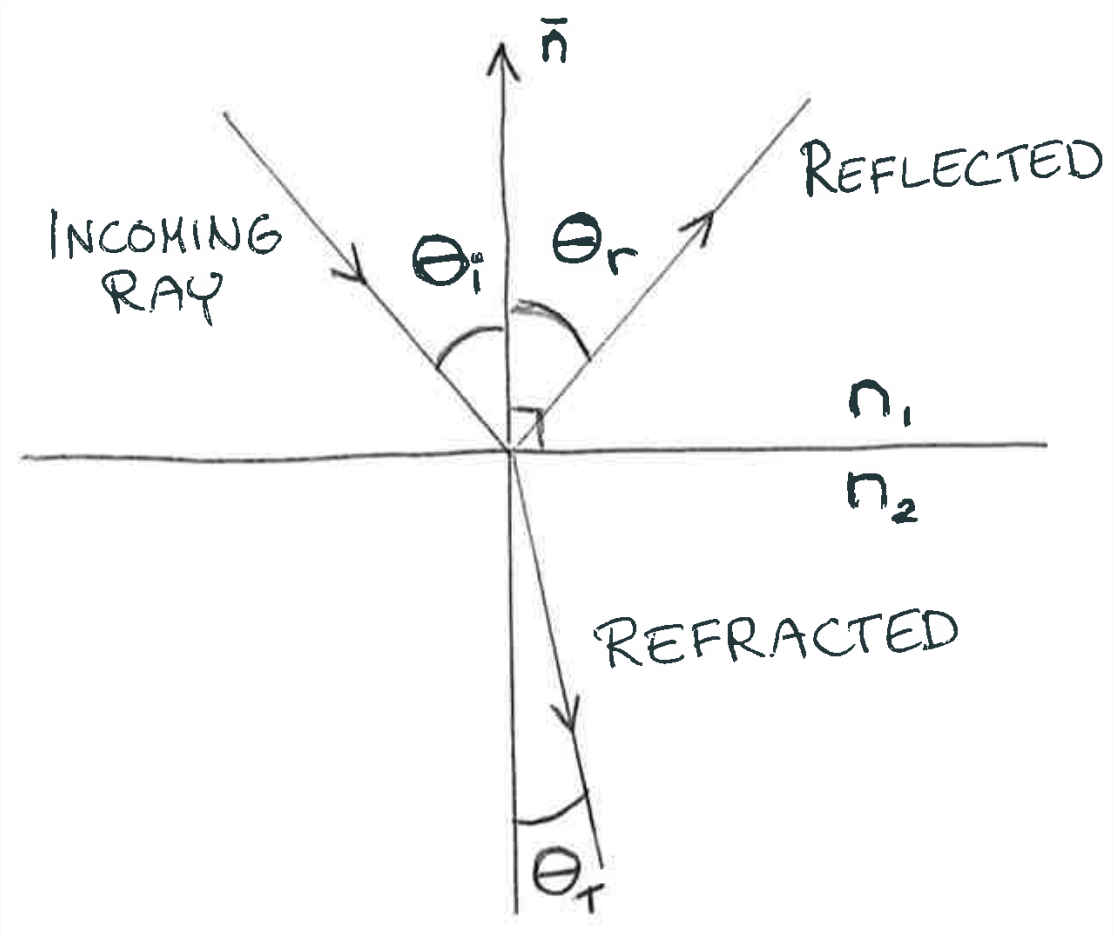
Today

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

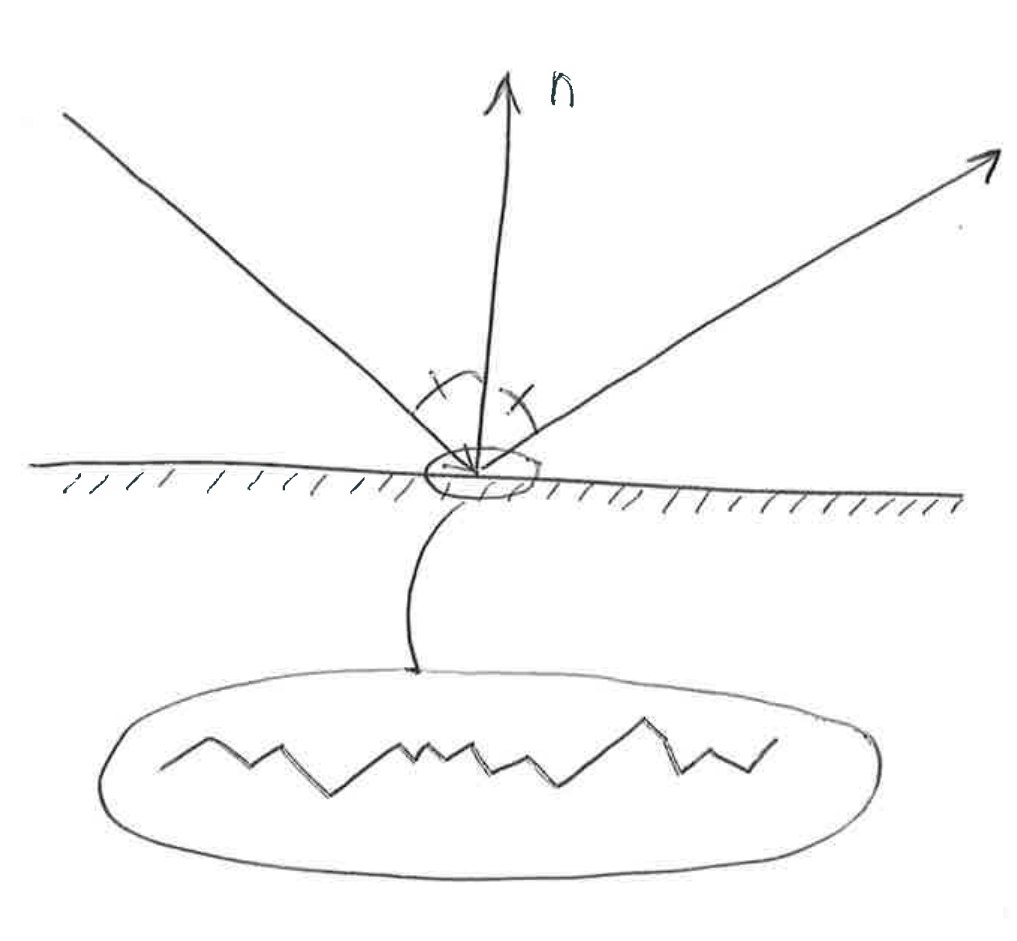
References

- Bidirectional-Reflectance-Distribution-Function
- Bump/Displacement-Maps, Environment Maps
- Reflection Mapping, Environment Mapping

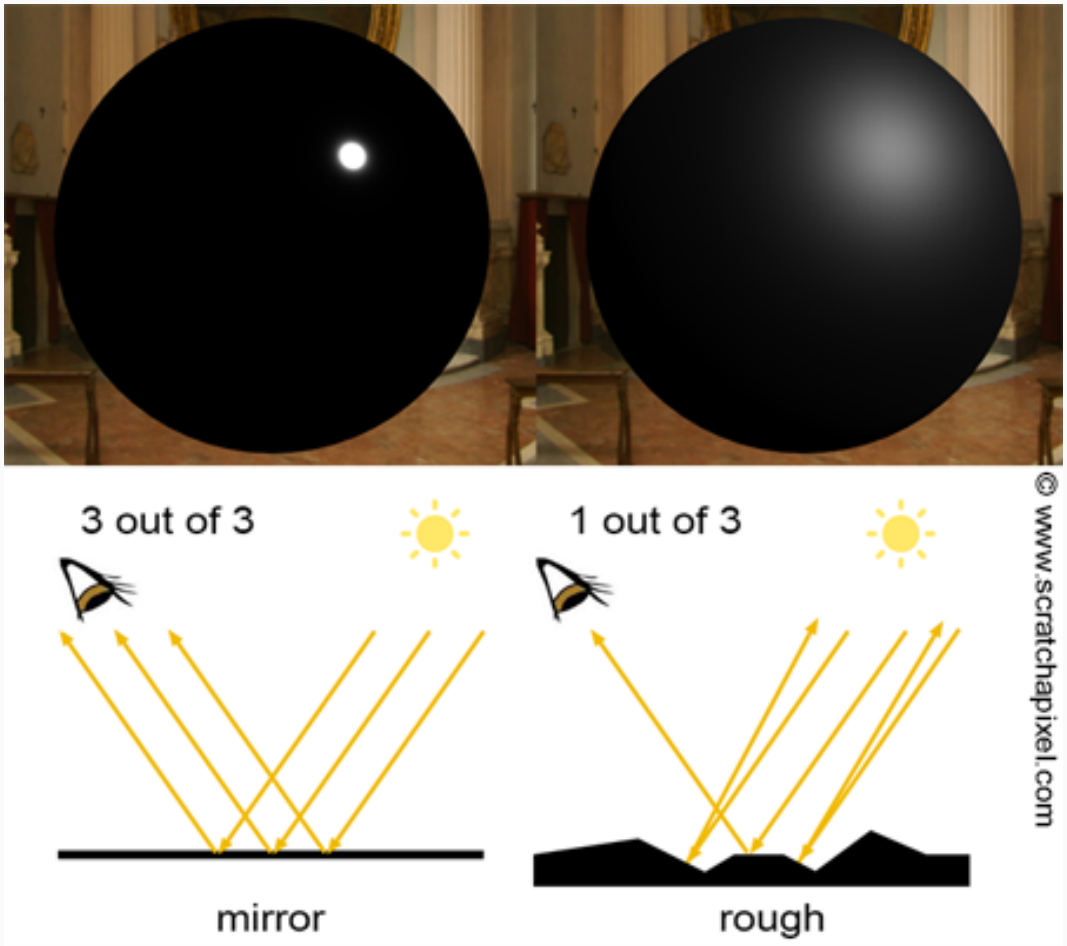
Motivation


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

Motivation



Motivation



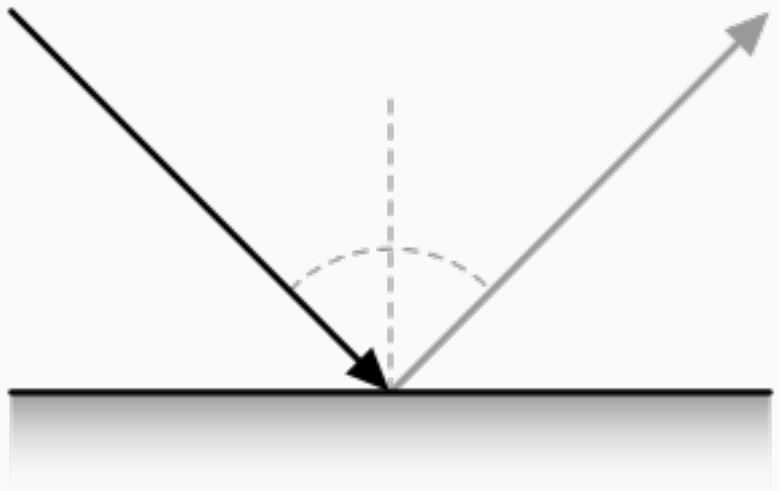
BRDF

Bi-Directional Reflectance Distribution Function

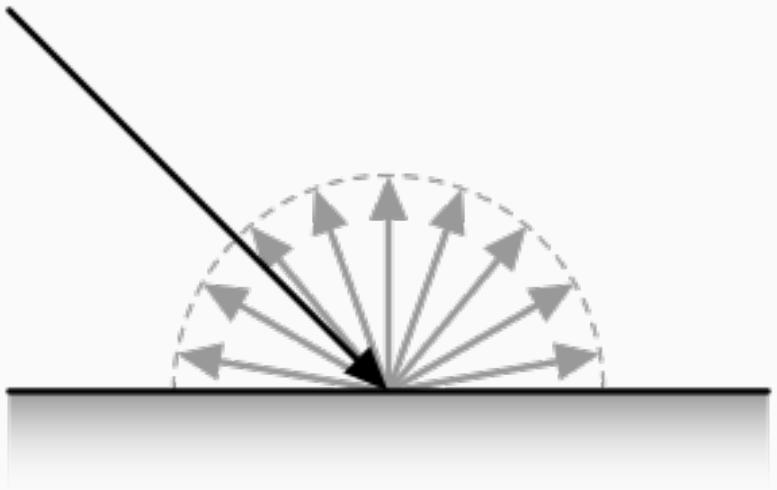
$$f(\mathbf{L}, \mathbf{V})$$

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

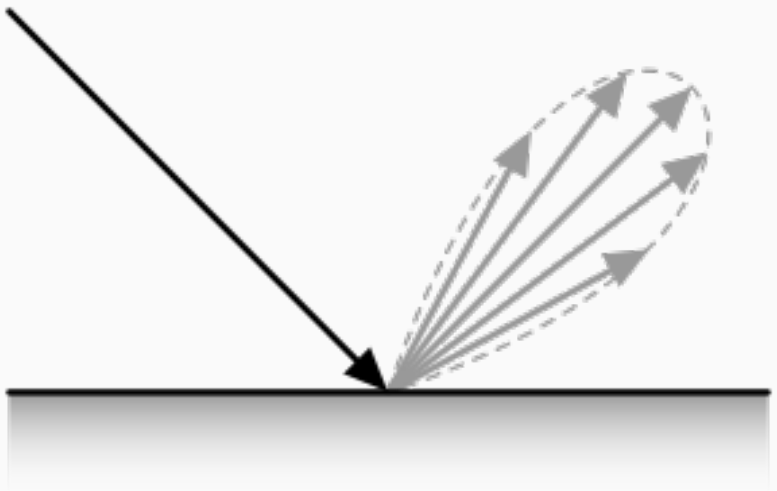
BRDF



BRDF



BRDF



BRDF: Requirements

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

Positive

BRDF: Requirements

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

Energy Conserving

BRDF: Requirements

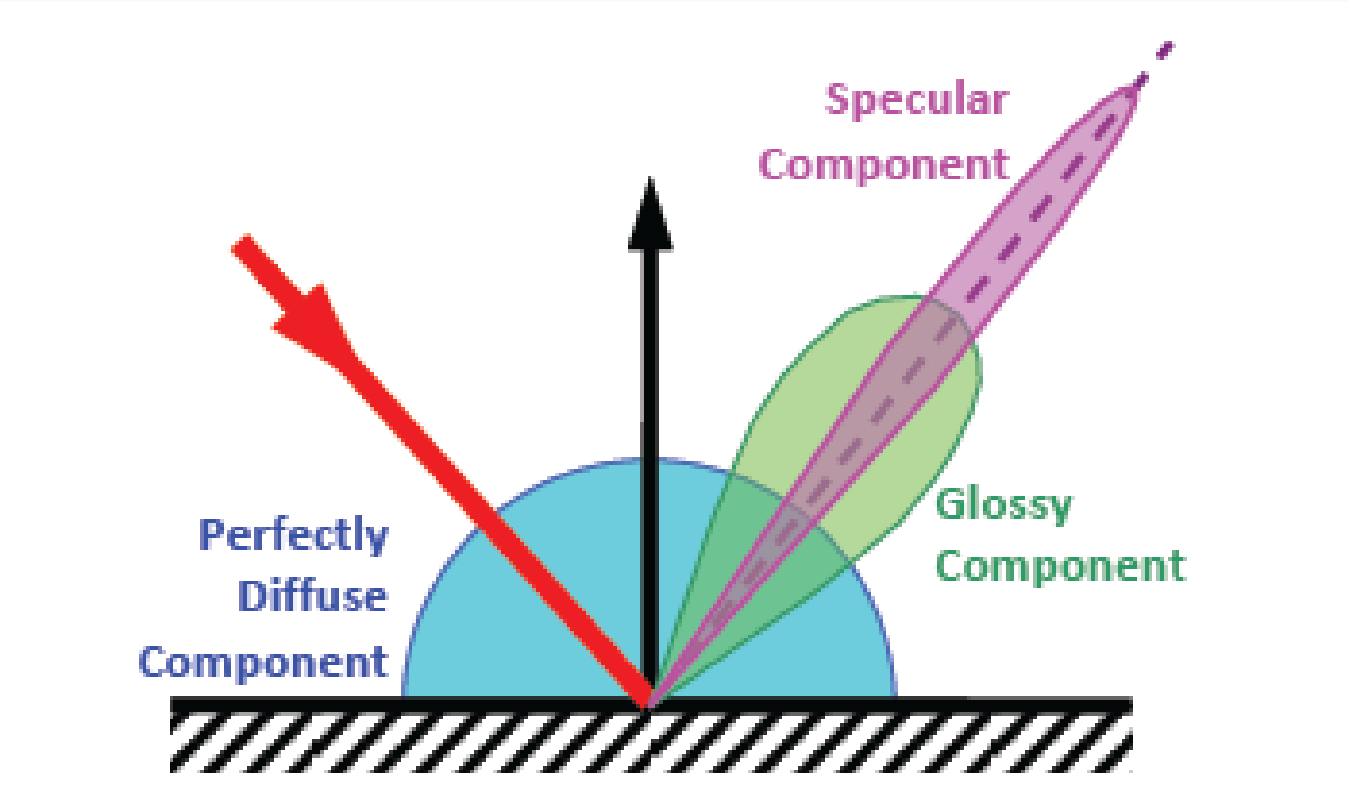
$$f(\mathbf{L}, \mathbf{V}) = f(\mathbf{V}, \mathbf{L})$$

Helmholtz reciprocity

Phong

$$\mathbf{i}_{spec} = (\mathbf{r}^{\mathrm{T}}\mathbf{v})^{m_{shi}}\,\mathbf{m}_{spec} \circ \mathbf{s}_{spec}$$

Combined BRDF

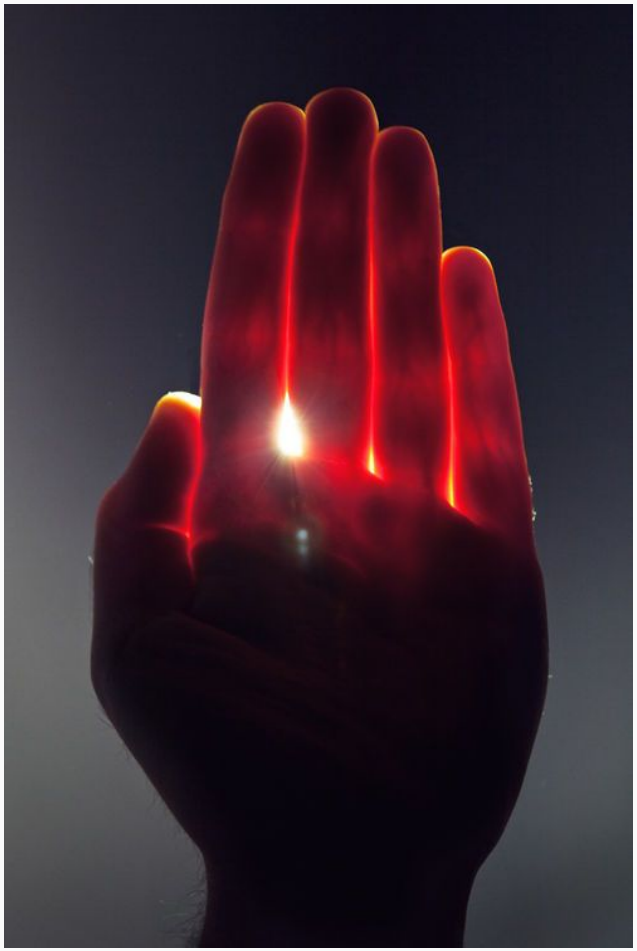


Translucent Objects

Final Fantasy - The Spirit Within

Shrek

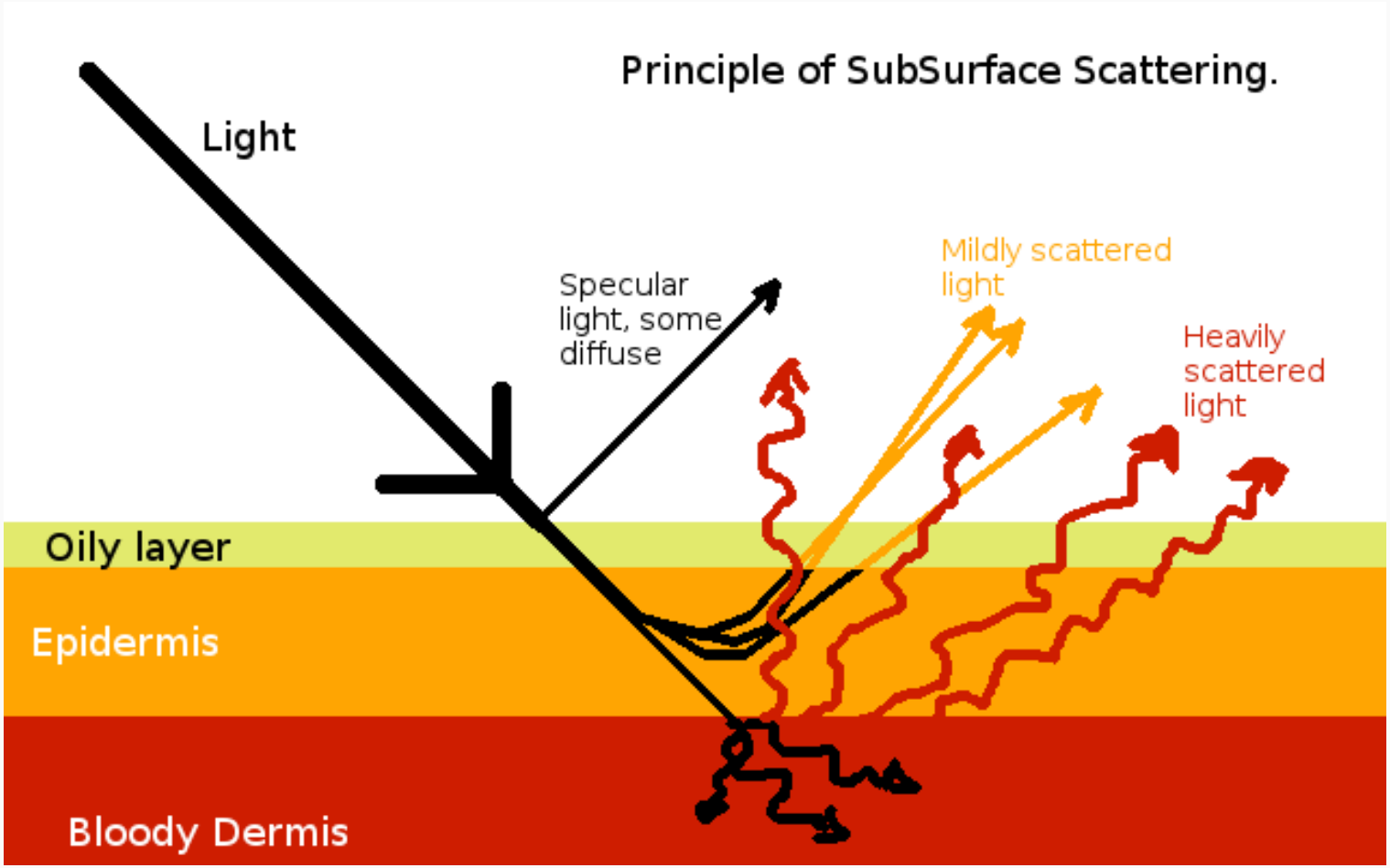
Translucent Objects



Translucent Objects

Translucent Objects

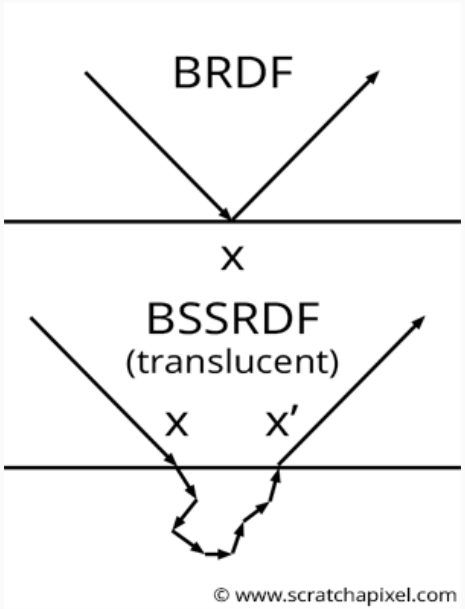
Skin


Principle of SubSurface Scattering.
Light
Specular light, some diffuse
Mildly scattered light
Heavily scattered light
Oily layer
Epidermis
Bloody Dermis

BRSSDF

BRSSDF - Bi-Directional Scattering-Surface Reflectance Distribution Function

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



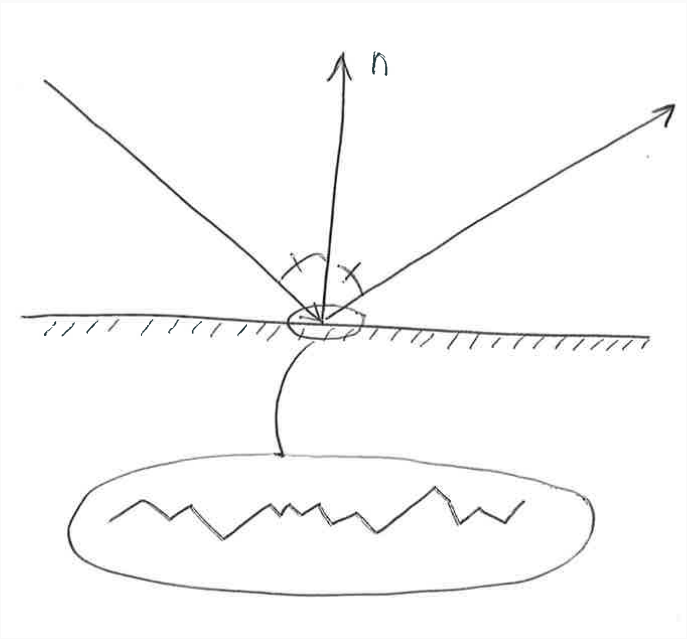
How to use BRDF

- 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

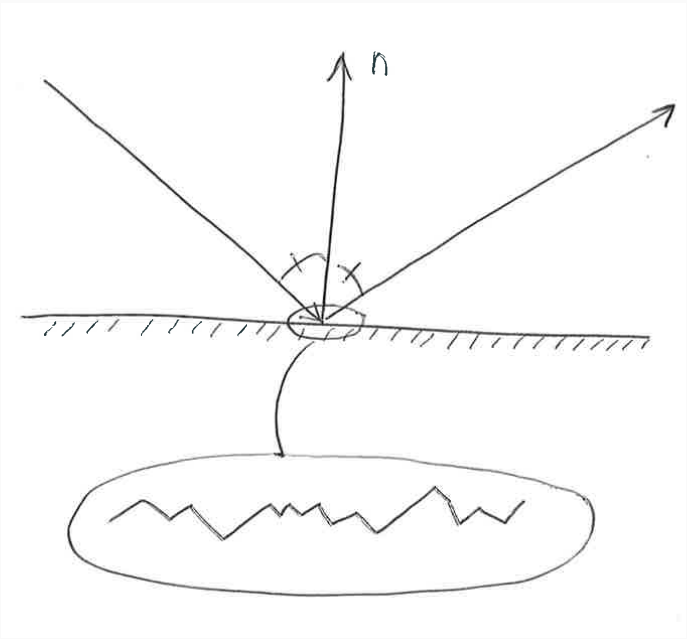
Introduction

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



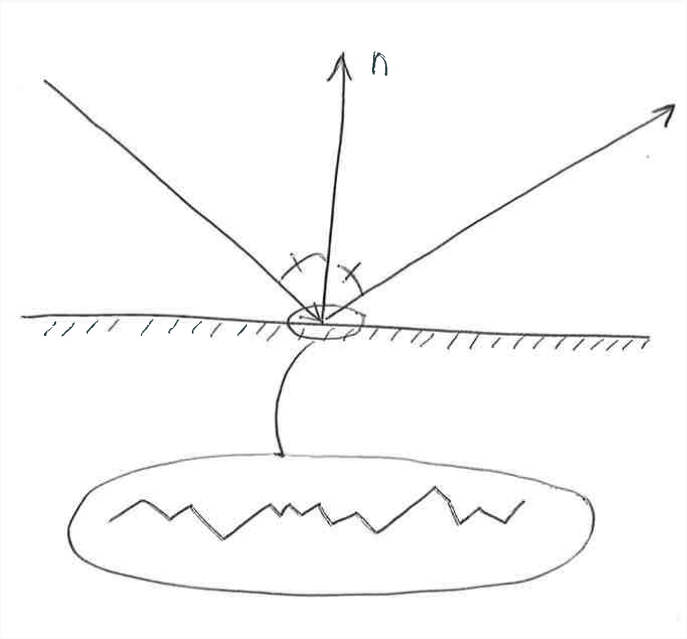
Introduction

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

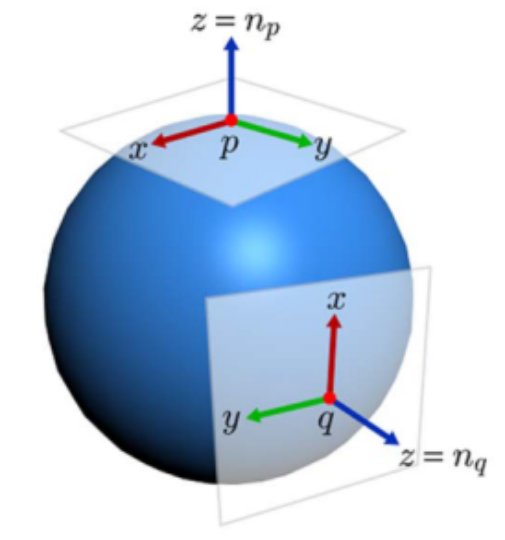


Introduction

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



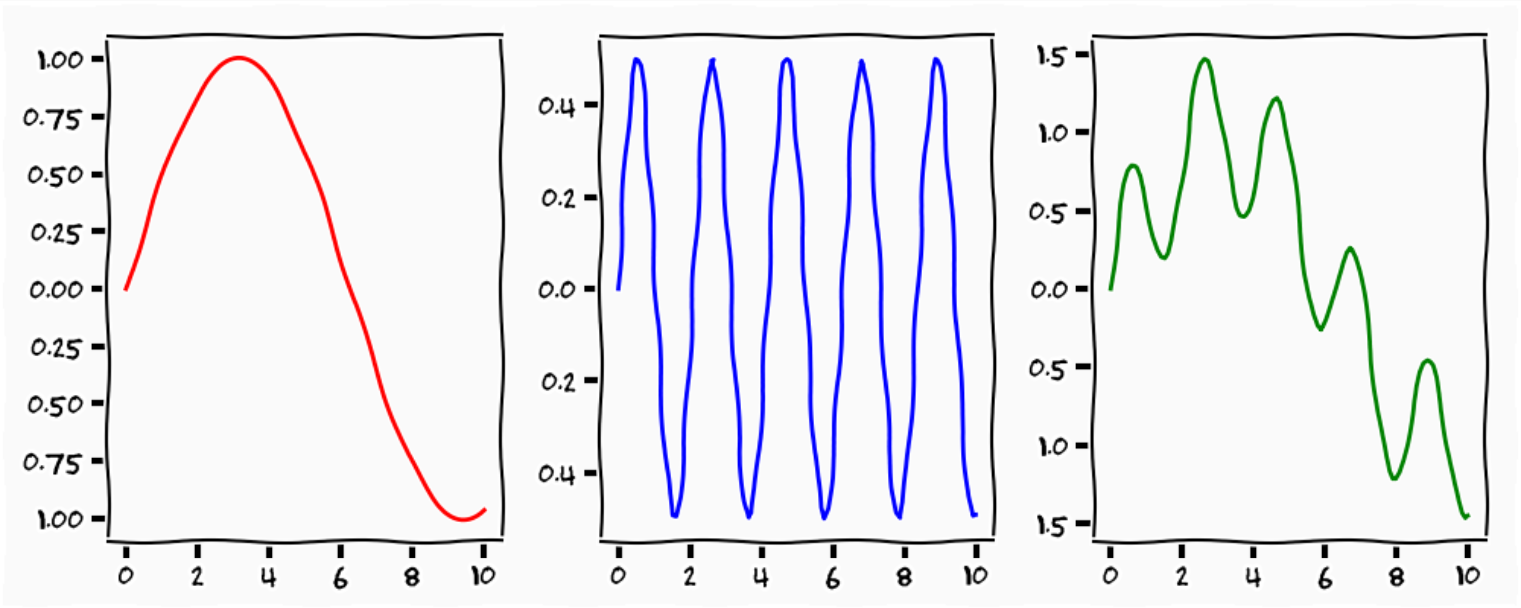
Normalmap



- Parametric surface $p(u, v)$
- Normal to surface

$$\mathbf{N} = \frac{\partial \mathbf{p}}{\partial u} \times \frac{\partial \mathbf{p}}{\partial v} = \mathbf{p}_u \times \mathbf{p}_v$$

Displacement



$$p'(u,v) = p(u,v) + b(u,v)\mathbf{n}$$

Displaced Normal

New normal

$$\begin{aligned}\mathbf{N}' &= \frac{\partial p'(u,v)}{\partial u} \times \frac{\partial p'(u,v)}{\partial v} \\ &= \frac{\partial}{\partial u}(p(u,v) + b(u,v)\mathbf{n}) \times \frac{\partial)}{\partial v}(p(u,v) + b(u,v)\end{aligned}$$

Partial derivative

$$\frac{\partial}{\partial u}(p(u,v) + b(u,v)\mathbf{n}) = \frac{\partial}{\partial u}p(u,v) + \left(\frac{\partial}{\partial u}b(u,v)\right)\mathbf{n} + b(u,v)\frac{\partial}{\partial u}\mathbf{n}$$

$$= \frac{\partial p(u,v)}{\partial u} + \frac{\partial b(u,v)}{\partial u}\mathbf{n}$$

Displaced Normal

$$\mathbf{N}' = \frac{\partial p'(u,v)}{\partial u} \times \frac{\partial p'(u,v)}{\partial v}$$

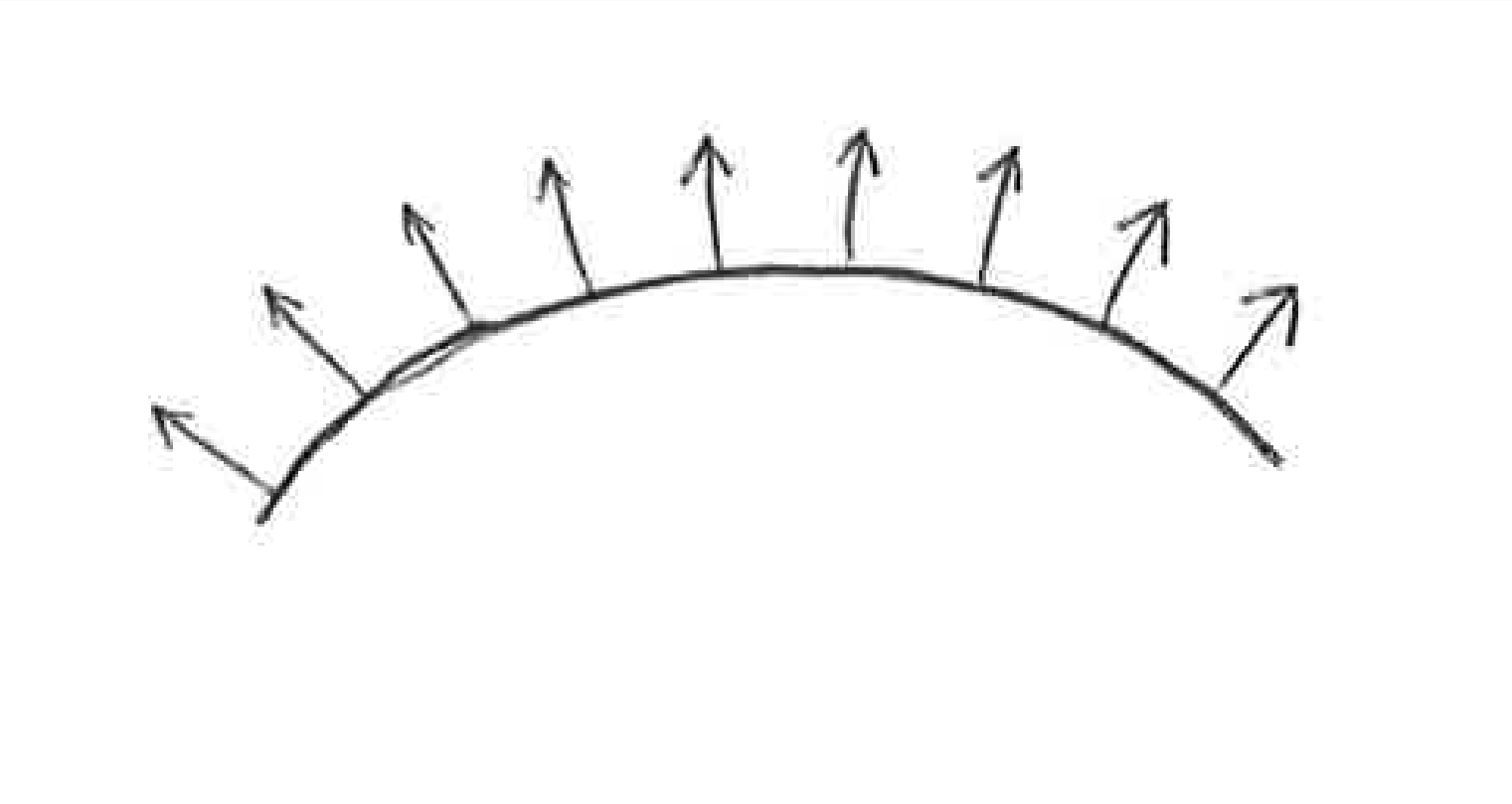
Displaced Normal

$$
\begin{aligned}
\mathbf{N}' &= \frac{\partial p'(u,v)}{\partial u} \times \frac{\partial p'(u,v)}{\partial v} \\
&= \frac{\partial}{\partial u} p(u,v) \times \frac{\partial}{\partial v} p(u,v) + \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial v} p(u,v)\right) \times \mathbf{n} \\
&+ \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial u} p(u,v)\right) \times \mathbf{n} + \frac{\partial}{\partial u} b(u,v) \frac{\partial}{\partial v} b(u,v)(\mathbf{n} \times \mathbf{n})
\end{aligned}
$$

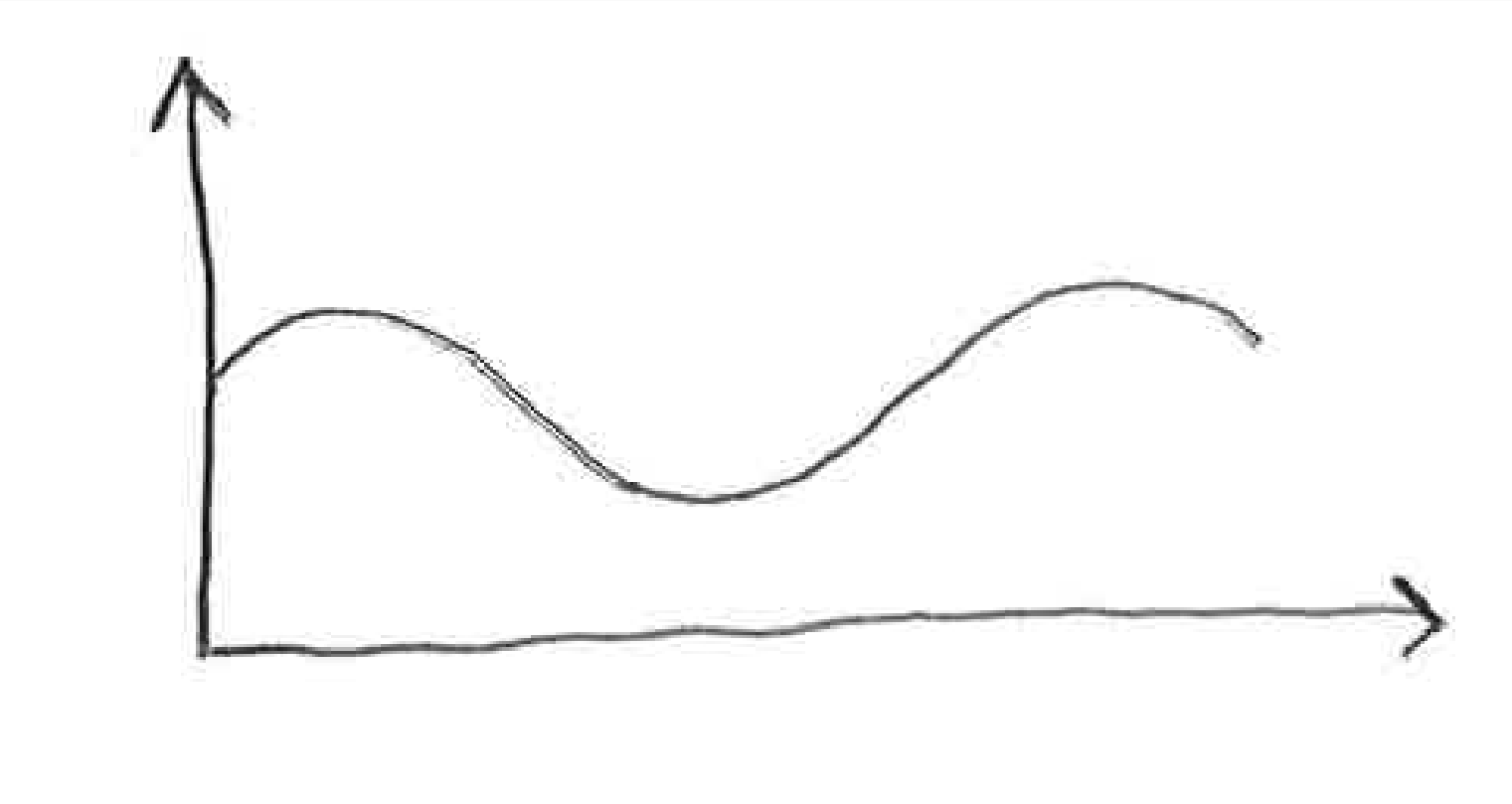
Displaced Normal

$$
\begin{aligned}
\mathbf{N}' &= \frac{\partial p'(u,v)}{\partial u} \times \frac{\partial p'(u,v)}{\partial v} \\
&= \frac{\partial}{\partial u} p(u,v) \times \frac{\partial}{\partial v} p(u,v) + \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial v} p(u,v)\right) \times \mathbf{n} \\
&+ \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial u} p(u,v)\right) \times \mathbf{n} + \frac{\partial}{\partial u} b(u,v) \frac{\partial}{\partial v} b(u,v)(\mathbf{n} \times \mathbf{n}) \\
&= \mathbf{N} + \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial v} p(u,v)\right) \times \mathbf{n} \\
&+ \left(\frac{\partial}{\partial u} b(u,v)\right)\left(\frac{\partial}{\partial u} p(u,v)\right) \times \mathbf{n}
\end{aligned}
$$

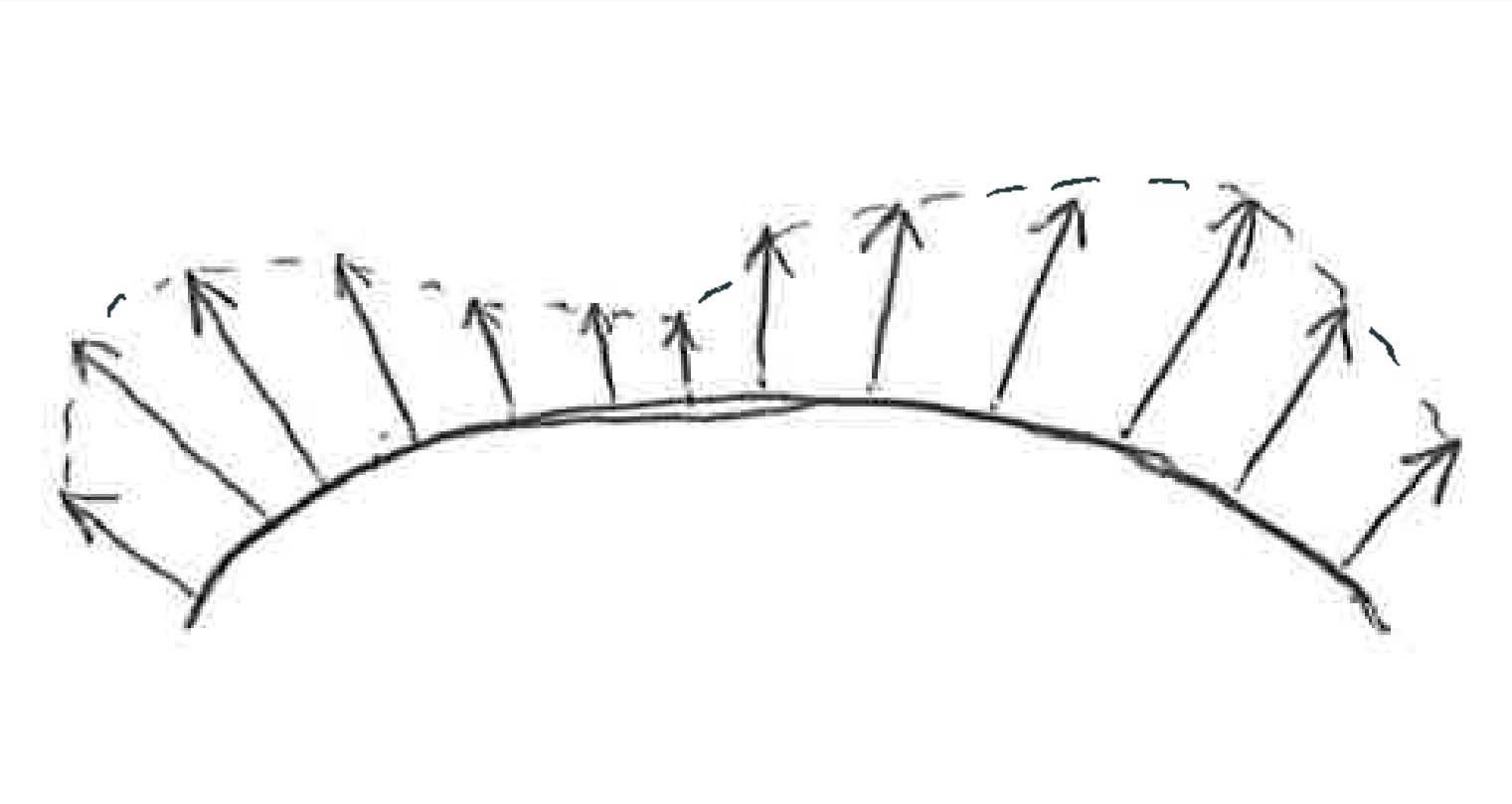
Normalmap



Normalmap

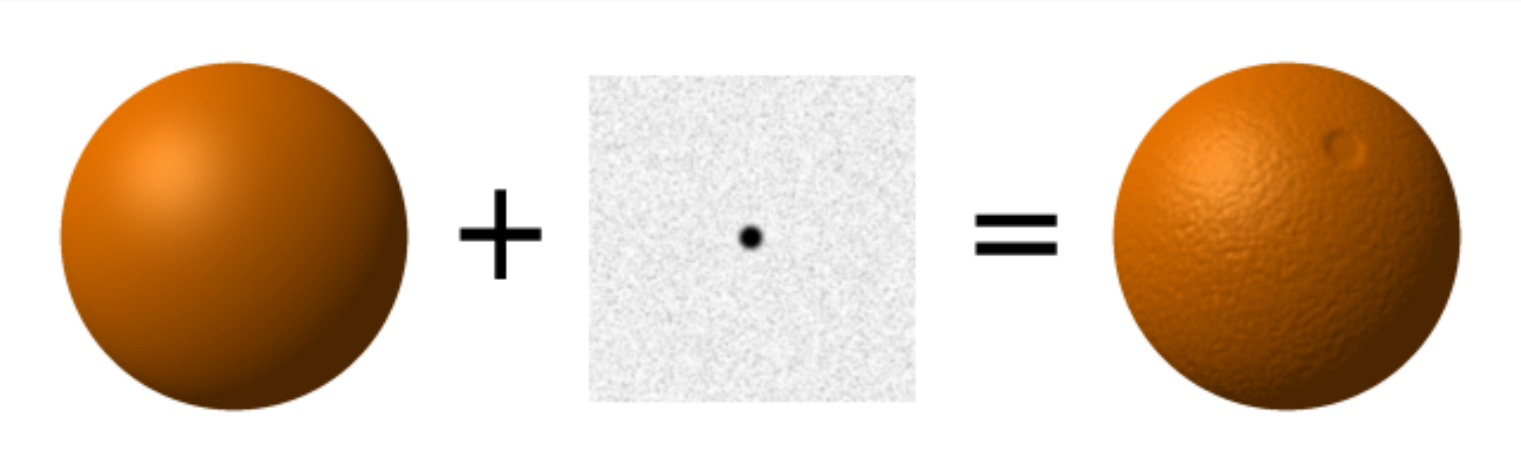


Normalmap



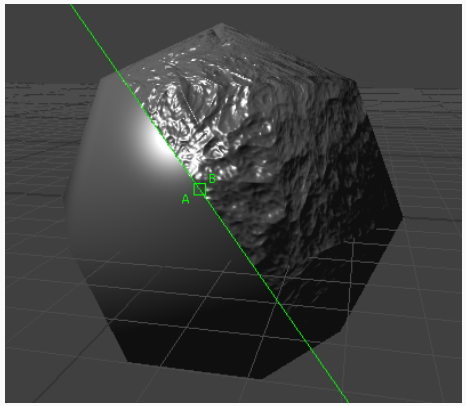
Normalmap

Bumpmap



Bumpmap

- 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

Code

```
convert <image_in> -set colorspace Gray
        -separate -average <image_out>
```

- Compute bump map

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

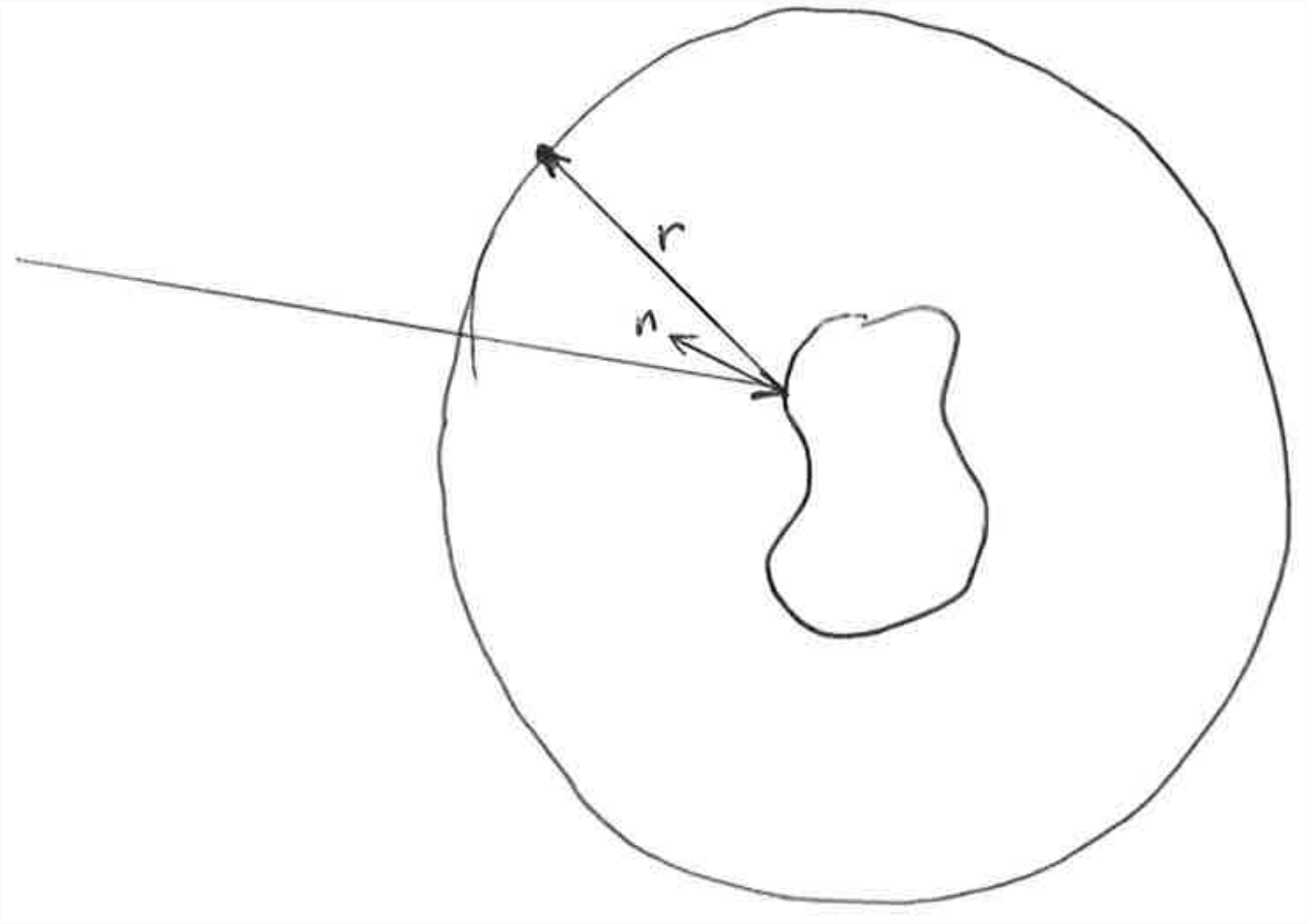
- Shade

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

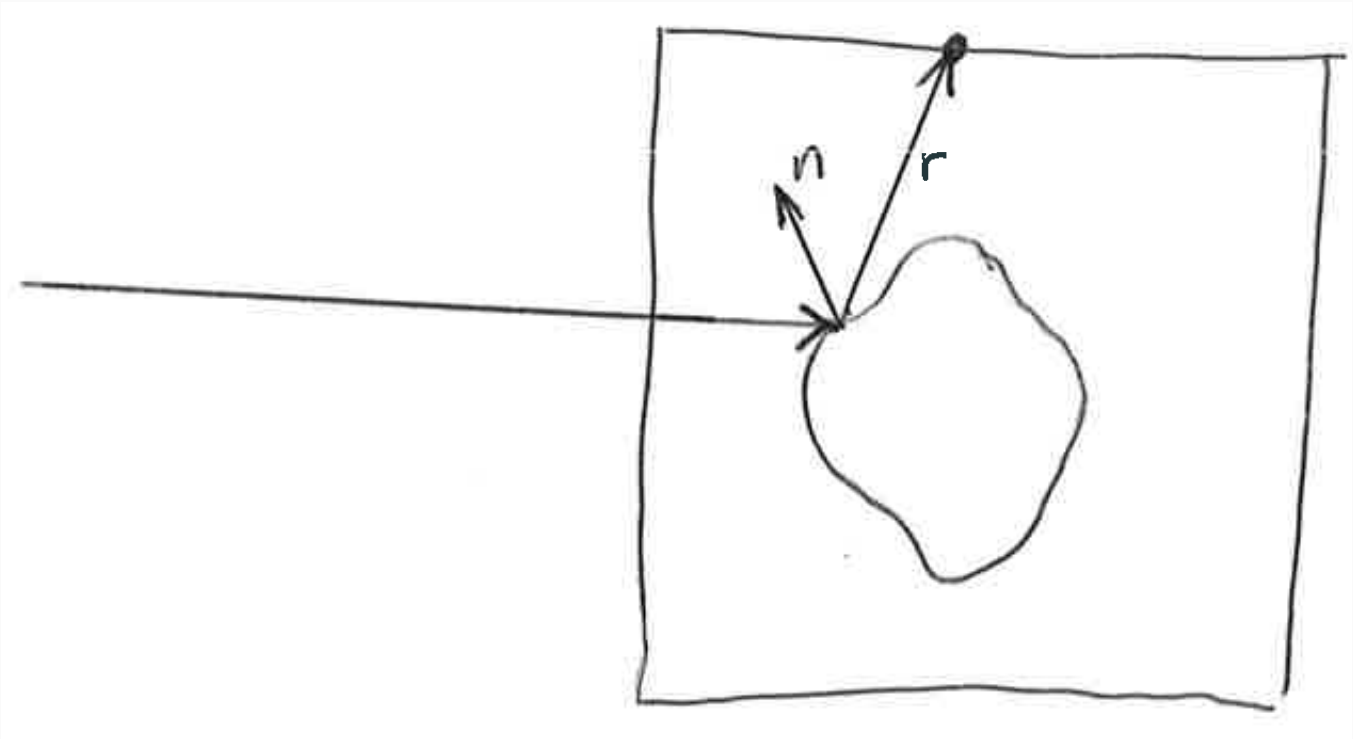
- *Not correct but you can tune the amplitude to work well*

Skybox

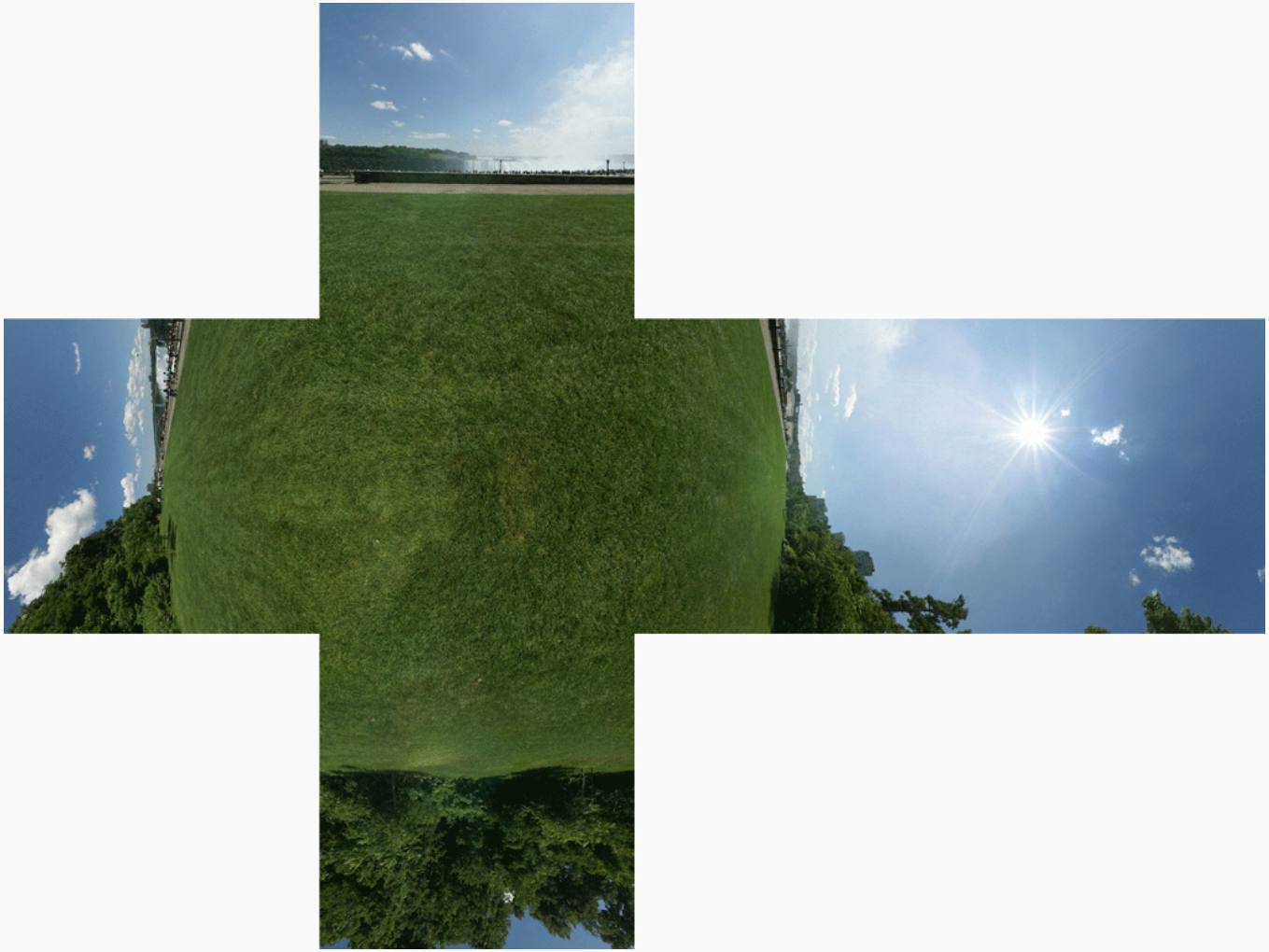
Env-Mapping



Env-Mapping



Cubemap



ENV-Mapping

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

Capturing ENV-Maps

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 Monday 18th of February

- Shadows
- Realistic Cameras
- Optimizations
- Finish Raytracing

Lab Continue with Lab 1

eof