09 Real time area lights

Steve Marschner

CS5625 Spring 2019

Area lighting

when lights are not point-like, we have an integration problem

$$L_r(\mathbf{v}) = \int_\Omega f_r(\mathbf{v}, \mathbf{l})\, L_i(\mathbf{l})\, (\mathbf{l} \cdot \mathbf{n})\, d\mathbf{l}$$

- for real time we need fast analytic solutions to (approximations of) this definite integral
- the form of the solution depends on the form of L_i and f_r.

diffuse

- polygonal lights: analytical solution due to Lambert (1760)
- environment lights: irradiance environment maps

specular

- polygonal lights: linearly transformed cosines
- environment lights: prefiltered environment maps

This is all with no shadows! That is a (large-ish) topic for later in the course.

Diffuse, polygonal lights

In this case the illumination integral simplifies to

$$L_r = \frac{R}{\pi} \int_{\Omega(S)} (\mathbf{n} \cdot \mathbf{l})\, d\mathbf{l}$$

Lambert worked out a solution to this in 1760

$$L_r = \frac{R}{2\pi} \sum_{i=1}^{n} \cos^{-1}(\mathbf{l}_i \cdot \mathbf{l}_{i+1}) \, \frac{\mathbf{l}_i \times \mathbf{l}_{i+1}}{\|\mathbf{l}_i \times \mathbf{l}_{i+1}\|}$$

- where the index is understood to wrap around for the last term
- (derivation on board; refer to Eric Heitz's notes linked on the schedule)

Specular shading, polygonal lights

Now the integral (for unit source radiance) reads

$$L_r(\mathbf{v}) = \int_{\Omega(S)} f_r(\mathbf{v}, \mathbf{l})\,(\mathbf{n} \cdot \mathbf{l})\,d\mathbf{l}$$

and analytic solutions are only available for special forms of the BRDF

- Phong BRDF: $f_r(\mathbf{v}, \mathbf{l}) \propto (\mathbf{r} \cdot \mathbf{l})^n$
  solution due to Arvo via a generalization of Lambert’s formula
- Other BRDFs: approximate the function $f_r(\mathbf{v}, \mathbf{l})$ by some convenient function $g(\mathbf{l})$
- Clever idea due to Heitz: linearly transformed cosine function:

$$g(\mathbf{l}) \propto (\mathbf{M}^{-1}\mathbf{l}) \cdot \mathbf{n}$$

Linearly transformed cosines

See Heitz’s slides, video, and WebGL demo

- https://eheitzresearch.wordpress.com/415-2/

Real-time environment illumination

Now incident radiance field is stored in a texture, not defined by a polygon

- this makes it easy to do mirror reflections: single cubemap lookup

Two special kinds of BRDFs are convenient

- diffuse BRDFs: irradiance depends only on surface normal (not **v**)
  …leads to **irradiance environment maps**
- rotationally symmetric lobes (e.g. Phong): it’s a convolution of the environment map
  …leads to **prefiltered environment maps**

Standard reflection map

For mirror-specular surface, the illumination integral reduces to

$$L_r(\mathbf{v}) = L_i(\mathbf{r}(\mathbf{v}, \mathbf{n}))$$

This is a function only of r

- so store it in a cubemap and look up using **r**(**v**, **n**).

Irradiance environment map

For diffuse surface, illumination integral reduces to

$$L_r = \frac{R}{\pi} \int_\Omega L_i(\mathbf{l})\,(\mathbf{n} \cdot \mathbf{l})\,d\mathbf{l}$$

This is a function only of n

- so store it in a cubemap and look up using **n**.

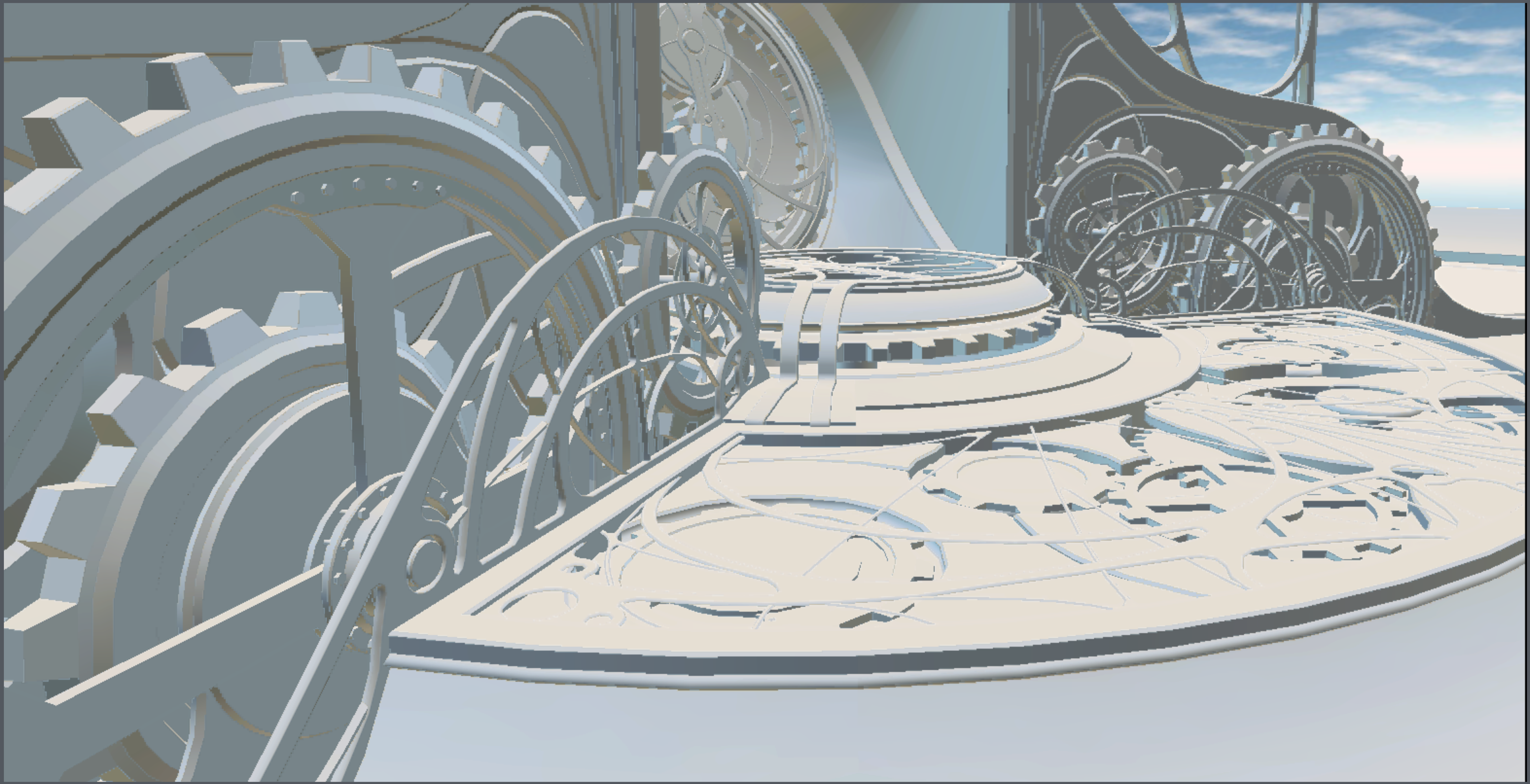
Irradiance environment map

environment map

irradiance map

Gary King in GPU Gems 2 http://http.developer.nvidia.com/GPUGems2/gpugems2_chapter10.html

Irradiance map illumination



McGuire et al. HPG '11 10.1145/2018323.2018327

Irradiance map illumination

McGuire et al. HPG ’11 10.1145/2018323.2018327

Prefiltered environment map

For a general specular surface, approximate the BRDF by a function

$$f_r(\mathbf{v}, \mathbf{l})\,(\mathbf{n} \cdot \mathbf{l}) \approx g_\sigma(\mathbf{l} \cdot \mathbf{r})$$

- where g defines a lobe of width σ that is symmetric around **r**

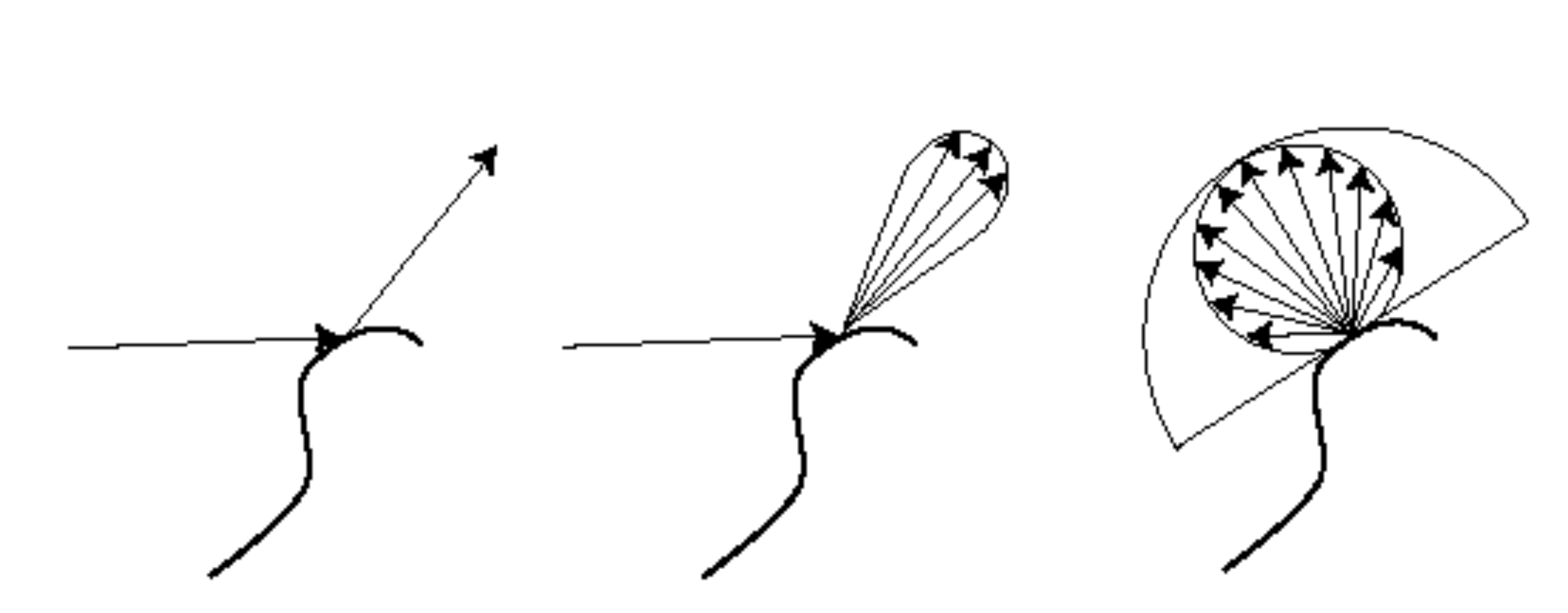
The illumination integral is

$$L_r(\mathbf{v}) \approx \int_\Omega g_\sigma(\mathbf{l} \cdot \mathbf{r})\, L_i(\mathbf{l})\, d\mathbf{l}$$

This depends only on r and the scalar σ

- so store it in an array of cubemaps indexed by σ, and look up using **r**

Irradiance environment mapping



environment map
for specular surface

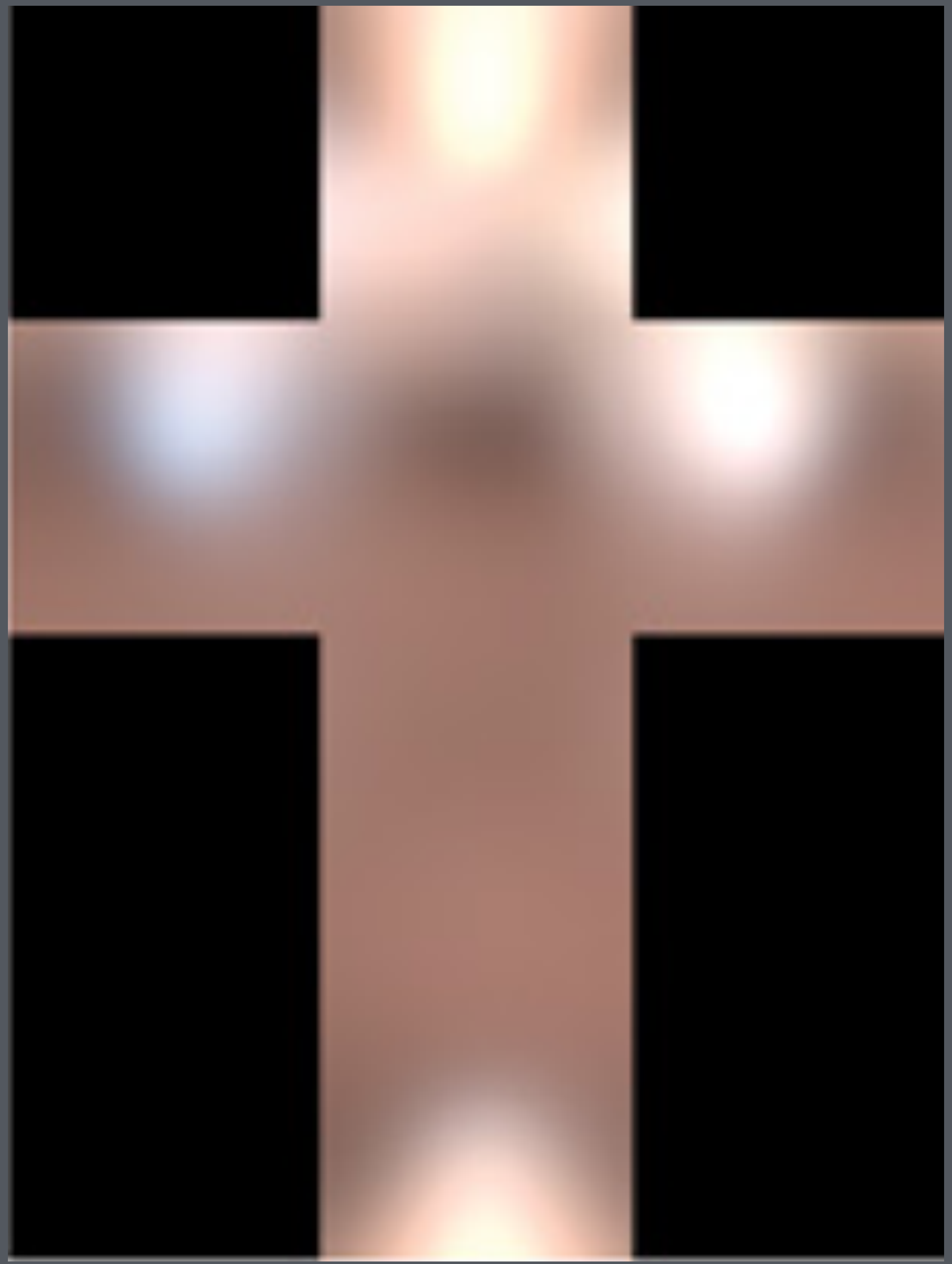
prefiltered map
for glossy surface

prefiltered map
for diffuse surface

Akenine-Möller et al. *RTR 3e*

Prefiltered environment map

environment map



prefiltered for Phong

Gary King in GPU Gems 2 http://http.developer.nvidia.com/GPUGems2/gpugems2_chapter10.html

Prefiltered map variants

Many approaches to approximating the BRDF with symmetric lobes

- classic is to use a single lobe, loses stretching at grazing angles
- many techniques use multiple samples, similar to anisotropic texture filtering
- modern thought process is to treat the whole precomputation as approximating more accurate BRDF lobes using multiple samples from arbitrary maps stored in mipmaps

Manson & Sloan 2016