Nonlinear Prefiltering for Surface Shading

Presenter:
Chun-Po Wang, Pramook Khungurn

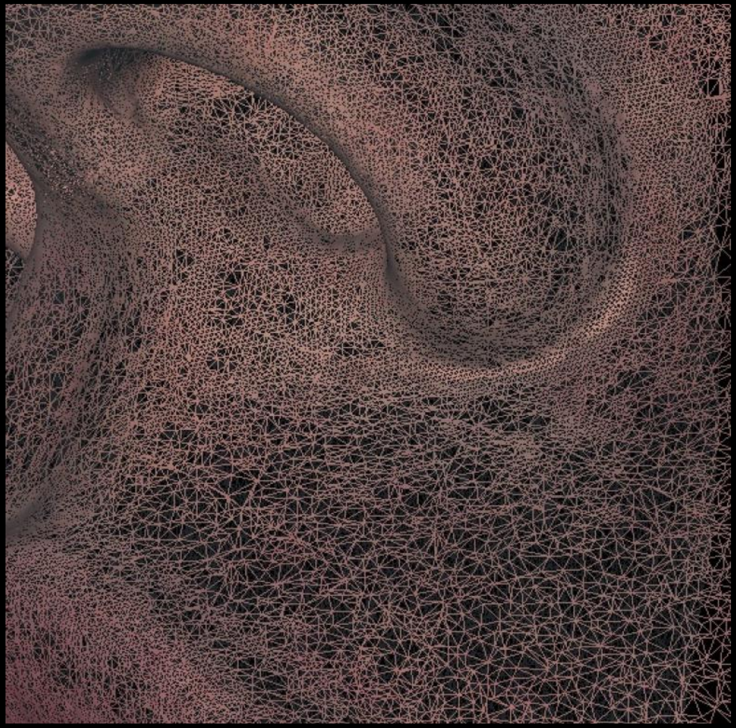
MOTIVATION

Motivation

- Real world objects have **surface details**.

Representing Surface Details

- Detailed meshes



Henrik van Jensen. "Digital Face Cloning." SIGGRAPH 2003 Sketch

Representing Surface Details

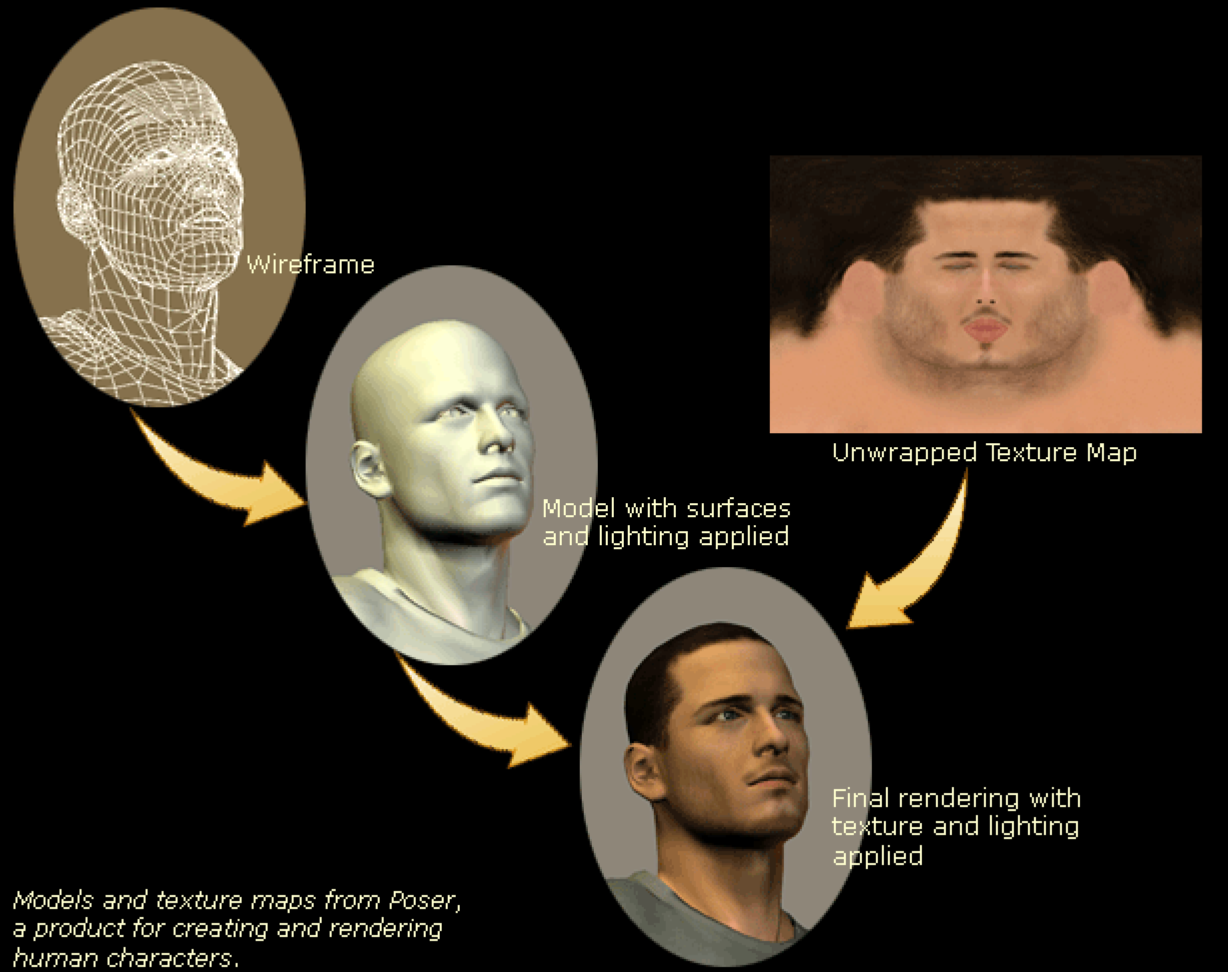
- Volume data

Samuli Laine, Tero Karras. “Efficient Sparse Voxel Octree.” I3D, 2010

Representing Surface Details

- The traditional approach
  - Coarse meshes
  - Texture maps
- Why not the last two approaches?
  - Huge memory requirement
  - Not available 10 years ago

Coarse Meshes + Texture Maps

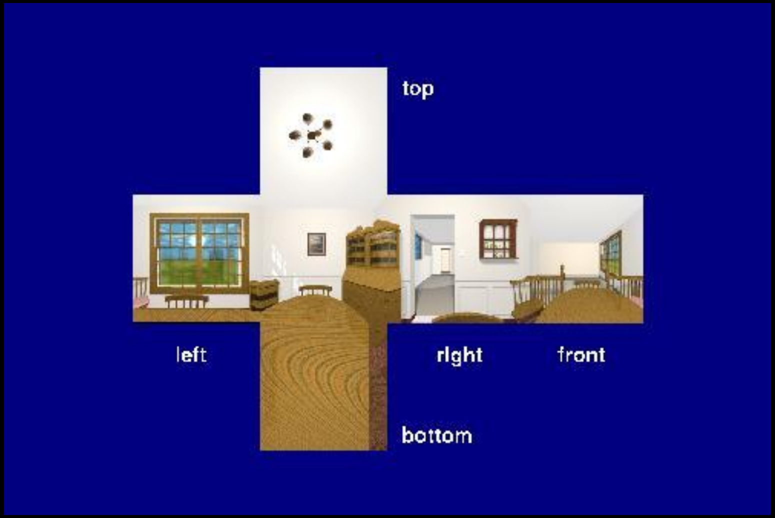


http://radoff.com/blog/2008/08/22/anatomy-of-an-mmorpg/

Types of Texture Maps

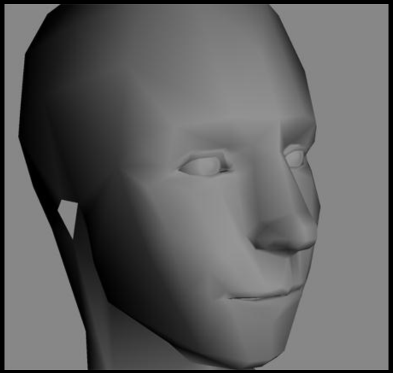
- Color maps
- Normal maps
- Horizon maps
- Shadow maps

Color Maps



http://www.siggraph.org/education/materials/HyperGraph/mapping/r_wolfe/

Normal Maps



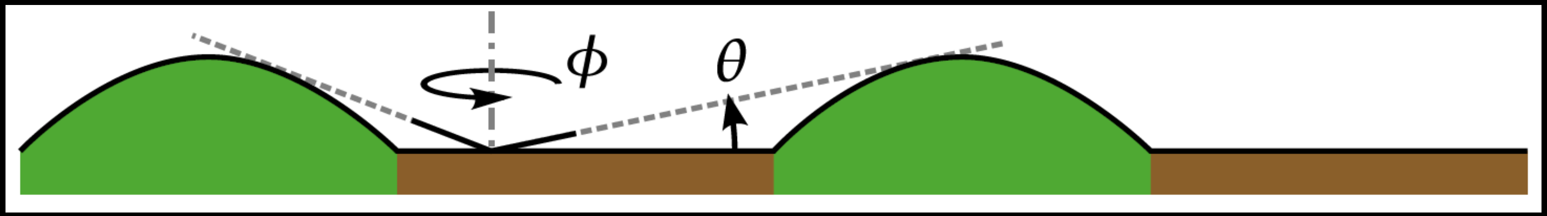
+

=

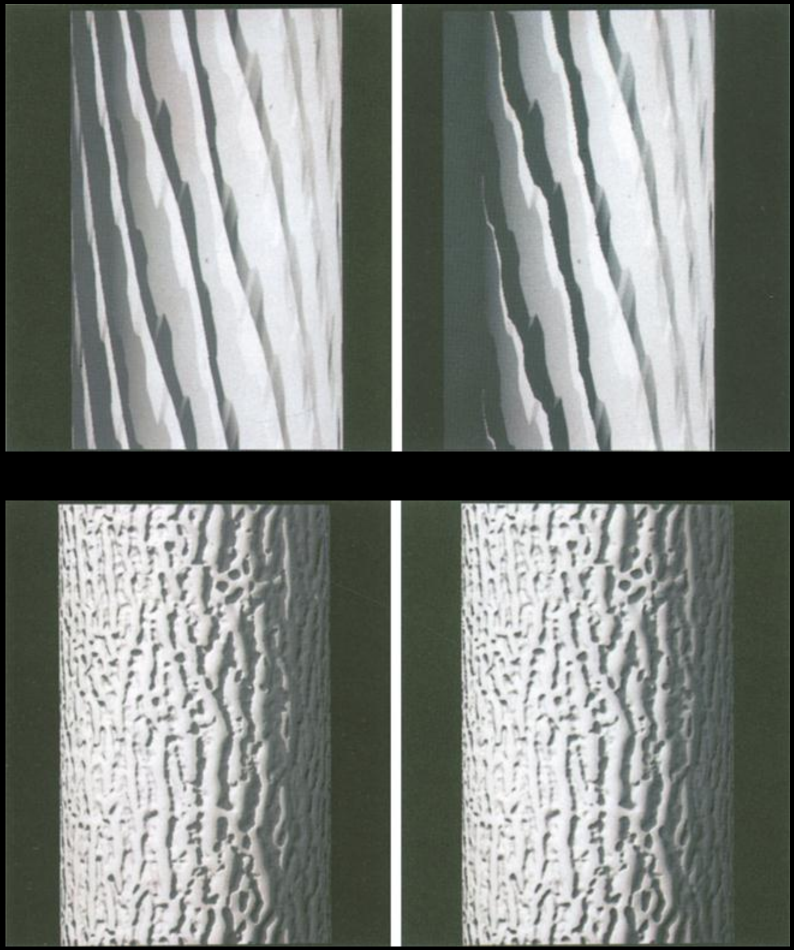
Horizon Maps

- Used to deal with self occlusion.

- Each texel stores a function $\Theta(\phi)$.
- $\Theta(\phi)$ = elevation angle beyond which the vision is not occluded by the object itself



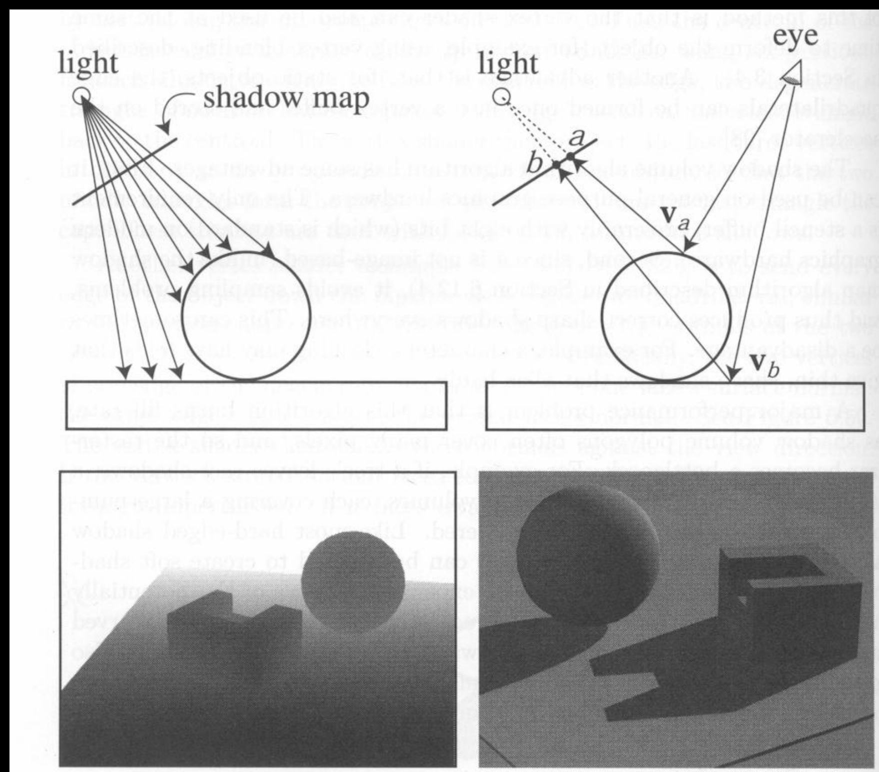
Horizon Maps



Nelson L. Max. “Horizon mapping: shadows for bump-mapped surfaces.” The Visual Computer, 1988

Shadow Maps

- Texels store the distance to the visible surface from light source.

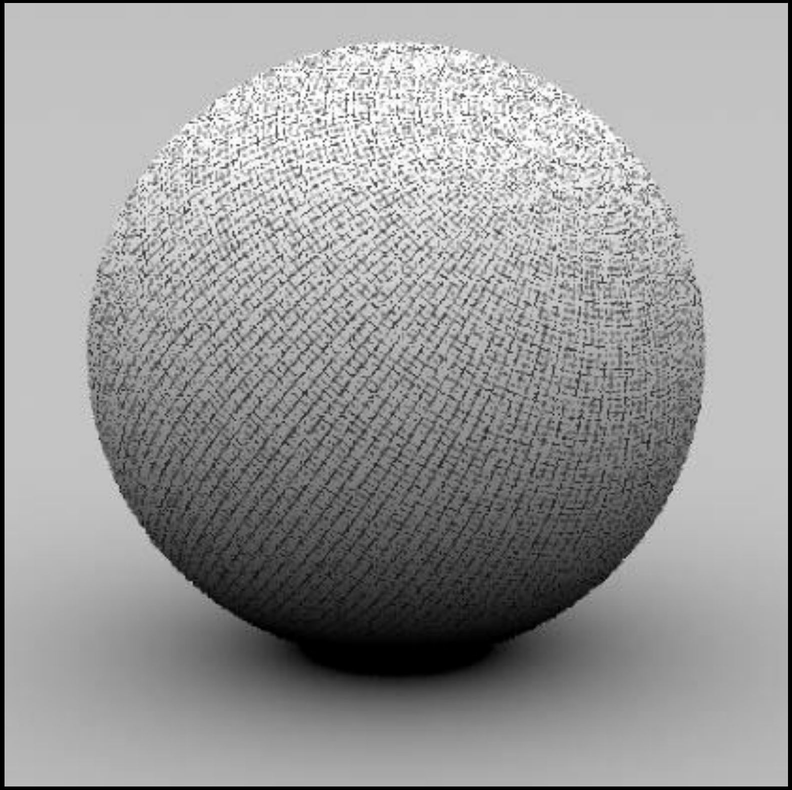


Foley et al. “Computer Graphics Principles and Practice”

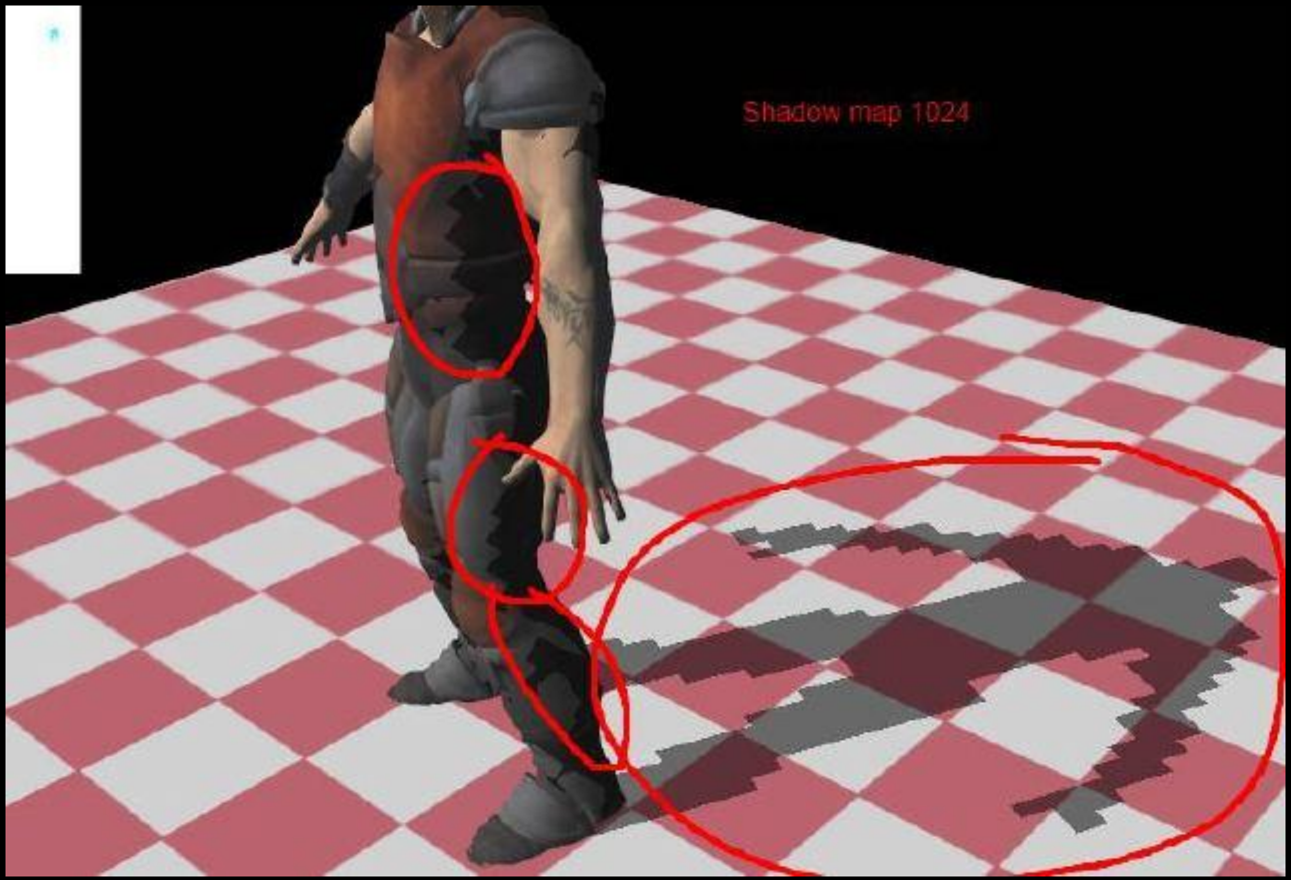
ALIASING

Problem: Aliasing

- Occur when a pixel covers many fine details.
- So much info in a pixel, but display just a little.

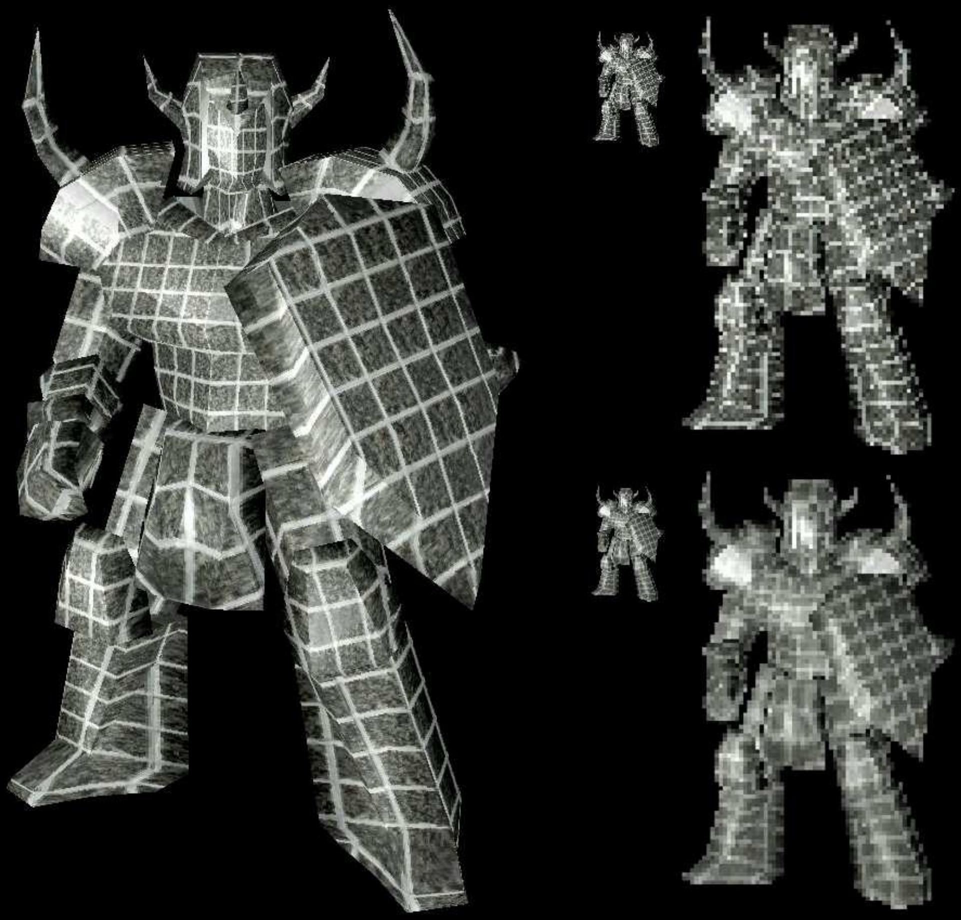


Problem: Aliasing



http://forums.create.msdn.com/forums/p/63434/388892.aspx

Problem: Aliasing



Ma et al. “Level-of-Detail Representation of Bidirection Texture Functions for Real-Time Rendering.” I3D, 2005

Removing Aliasing

- Two solutions
  - Supersampling
  - Prefiltering

- Supersampling
  - Throw in more samples in a pixel.
  - More samples = more info covered.
  - Always works.
  - But SLOW…

Prefiltering

- Come up with low-detail versions of the map.
  - "Filter" the map.

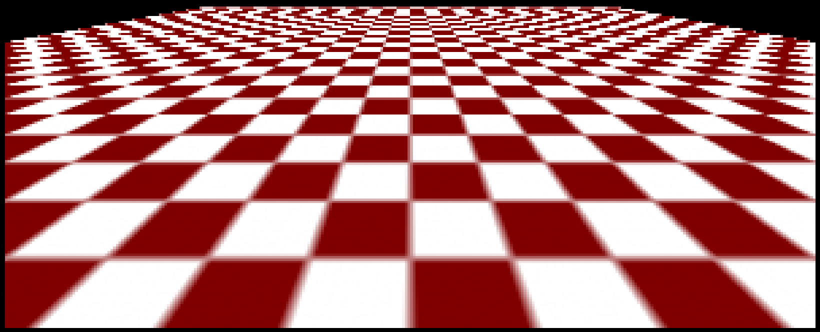
- Use the right low-detail version according to how many texels are covered.

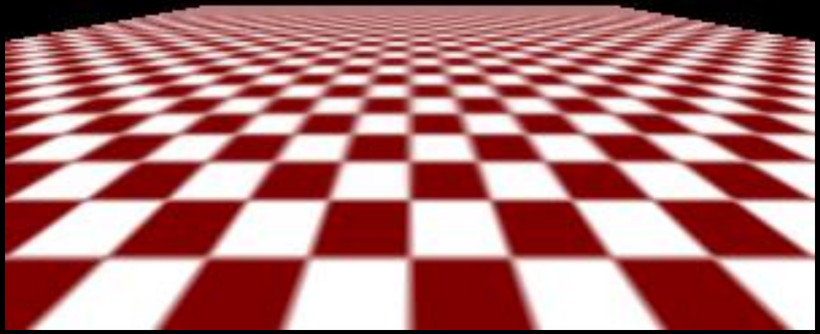
Prefiltering Example: Color Map

- MIP-mapping
  - Decimate the original color map by factor of 2.
  - Do so until getting a 1x1 image.
  - Use smaller image when pixel cover more texels.

http://www.gamedev.net/page/resources/_/technical/directx-and-xna/mip-mapping-in-direct3d-r1233

Prefiltering Example: MIP-map

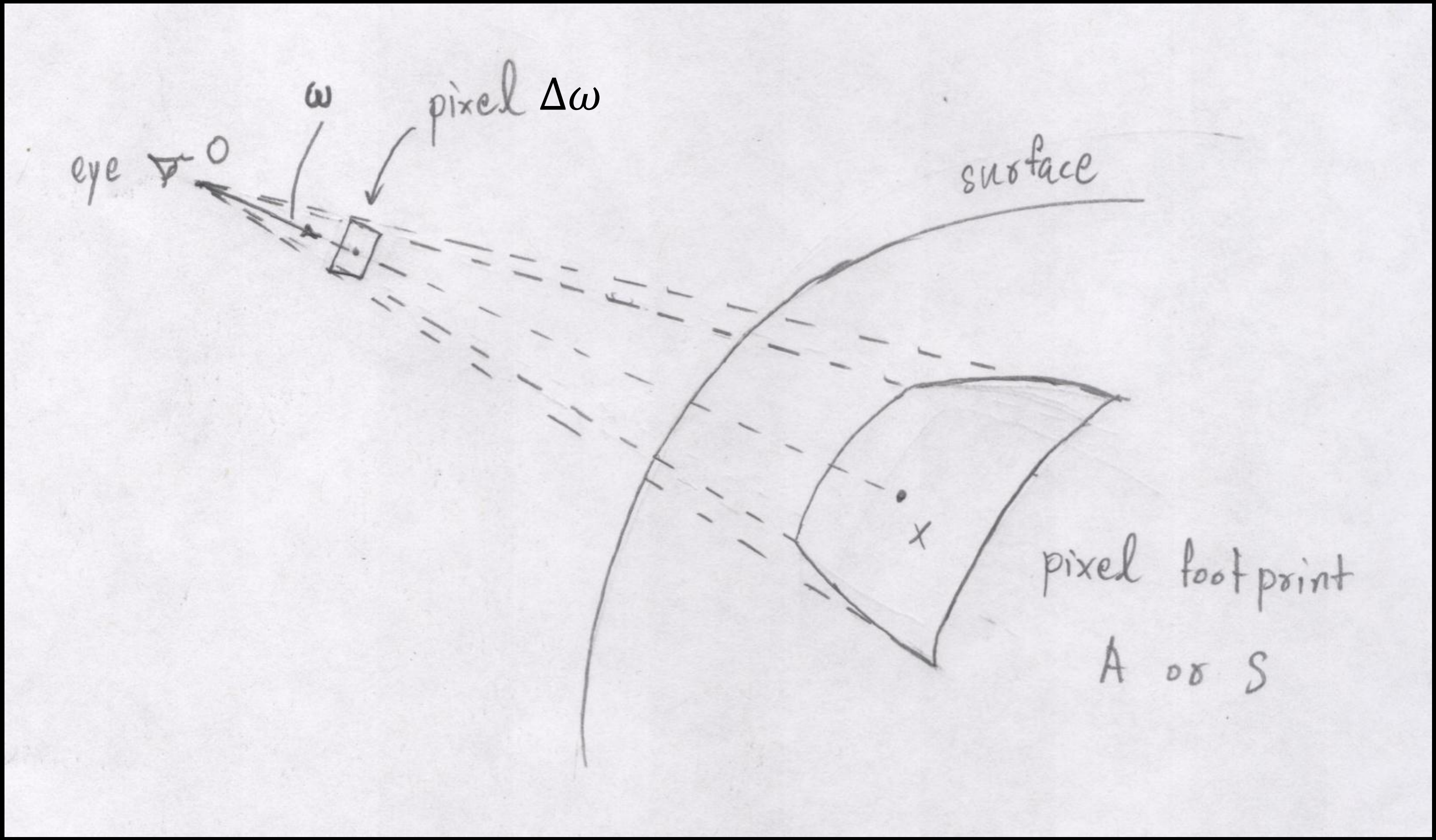




PREFILTERING FRAMEWORK

Prefiltering Framework

- Setting



Prefiltering Framework

- Measurement Equation

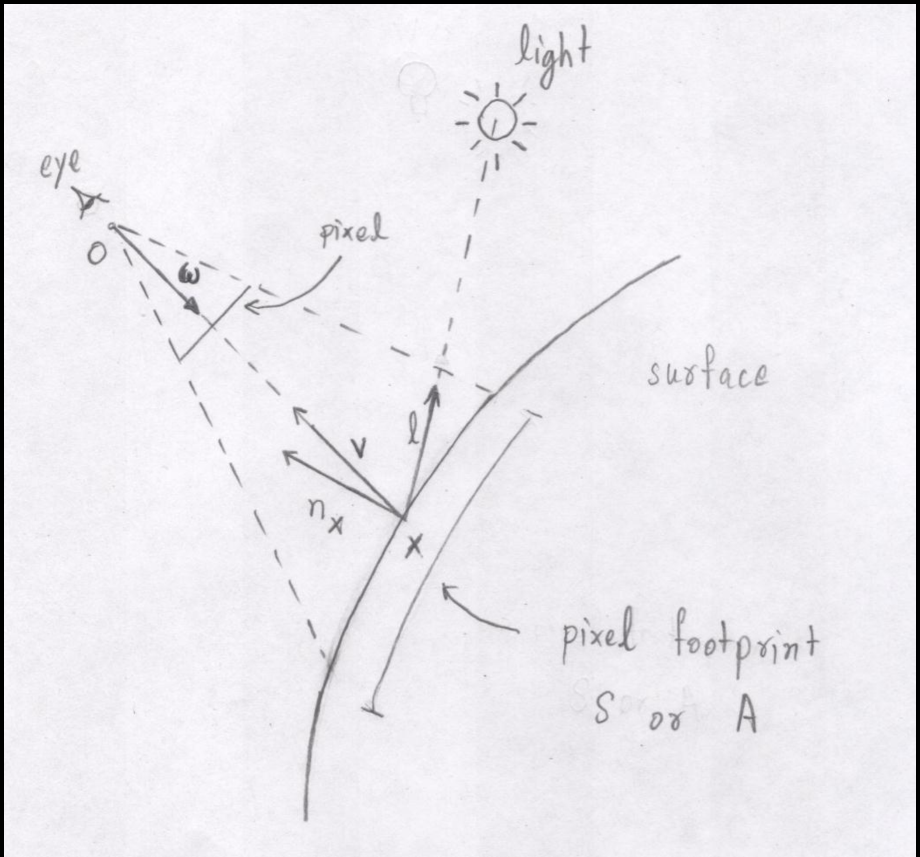
$$I = \int_{\Delta\omega} w(\omega) L(o, \omega) d\sigma(\omega)$$

  - I = pixel color
  - Weight function $w(\omega)$
    - Assign weight to each direction in the pixel.
    - Integrates to 1 over the pixel.

$$\int_{\Delta\omega} w(\omega) d\sigma(\omega) = 1$$

Prefiltering Framework

- Transform to integral over surface



Prefiltering Framework

- Transform to integral over surface
  - Weight function $\hat{w}(x)$ over points.
  - Expression for color:

$$I = \int_A \hat{w}(x) L(x, v) dA(x)$$

  - Weight function integrates to 1 over the pixel:

$$\int_A \hat{w}(x) dA(x) = 1$$

Bidirectional Texture Function

- $\tau_A(l, v)$ function
  - l = direction where light come from
  - v = view direction
  - A = pixel footprint

"fraction of light from l
that gets reflected to v
from pixel footprint A"

Bidirectional Texture Function

- Usage:

$$I = \int_{\Omega} \tau_A(v, l)\, E(l) d\sigma(l)$$

$E(l)$ = environment light from direction l

PREFILTERING APPROACHES

Brute Force

- Precompute $\tau_A(l, v)$
  - Different footprint sizes A
  - Different light direction l, and view direction v.
  - Around every point on the surface.

- Huge space requirement.
- Lots of literature on this topic.
  - Most on compression

Independent Prefiltering

- BTF approach assumes everything is *coupled.*
- We can prefilter each map *independently*.
  - Color map
  - Normal map (BRDF map)

- That is, we assume

$$\tau_A(l, v) \approx k_A \times \rho_A(l, v)$$

  - k_A is the prefiltered color over A
  - $\rho_A(l, v)$ is the prefiltered BRDF over A

Independent Prefiltering

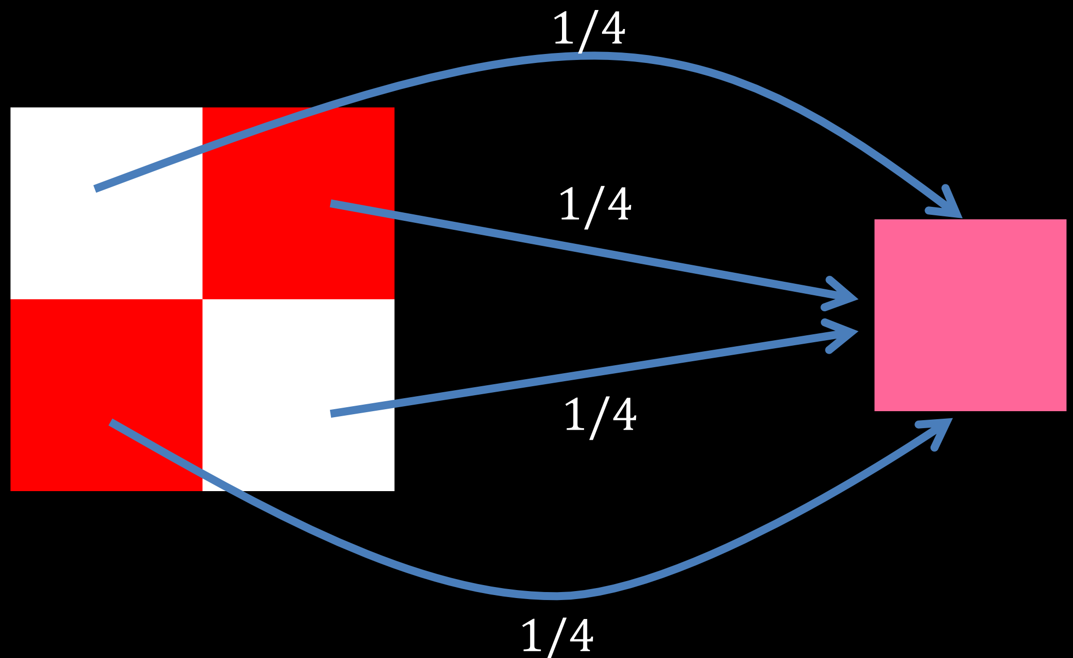
$$k_A = \int_A \hat{w}(x) k_x dA(x)$$

$$\rho_A(l, v) = \int_A \hat{w}(x) \rho_x(v, l) \;\; n_x l \;\; dA(x)$$

where $n_x l = \max(0, n_x \cdot l)$

Linear Prefiltering

- Filtering color map is "linear."
- Color in low-detail maps
  = weighted average of color in high-detail map



NORMAL MAP PREFILTERING

BRDF

- The BRDF is determined by the normal.
  - Lambertian: $\rho_x(l, v) = \max(0, n_x \cdot l) = (n_x l)$
  - Blinn-Phong: $\rho_x(l, v) = \frac{(n_x h)^\alpha}{(n_x l)}$ where $\mathrm{h} = \frac{l+v}{\|l+v\|}$

$$\rho_x(\mathbf{l}, \mathbf{v}) = \rho(\mathbf{n_x}, \mathbf{l}, \mathbf{v})$$

- In general, BRDF is a nonlinear function of n_x

BRDF Filtering: Filter the Normal Map

Want:

$$\rho_A(\mathbf{l}, \mathbf{v}) = \int_A \hat{w}(\mathbf{x})\, \rho_x(\mathbf{l}, \mathbf{v})\, \mathbf{n_x l}\, \mathrm{d}\mathbf{x}$$

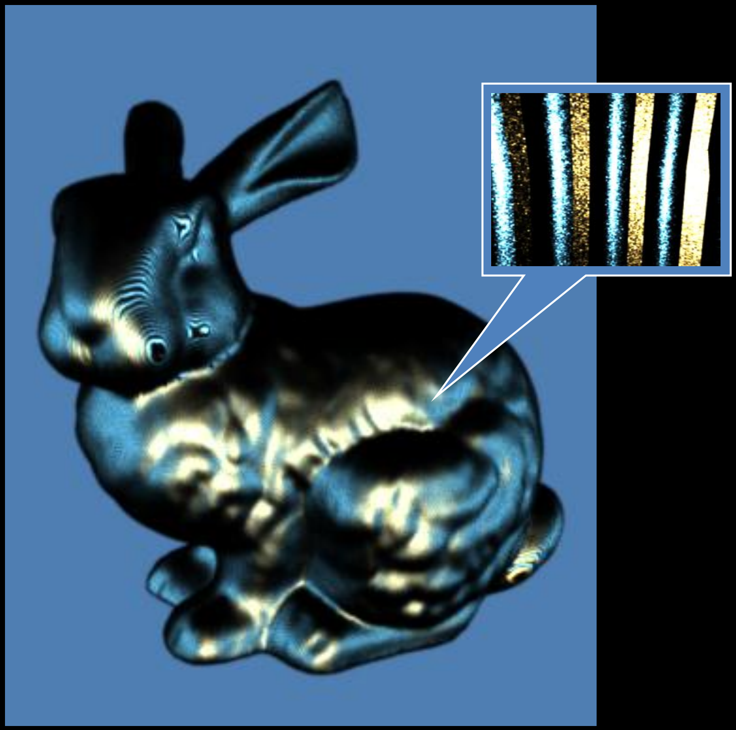
Can we filter the normal instead?

$$\mathbf{n_A} = \int_A \hat{w}(\mathbf{x})\, \mathbf{n_x}\, \mathrm{d}\mathbf{x}$$

$$\rho_A(\mathbf{l}, \mathbf{v}) = \rho(\mathbf{n_A}, \mathbf{l}, \mathbf{v})?$$

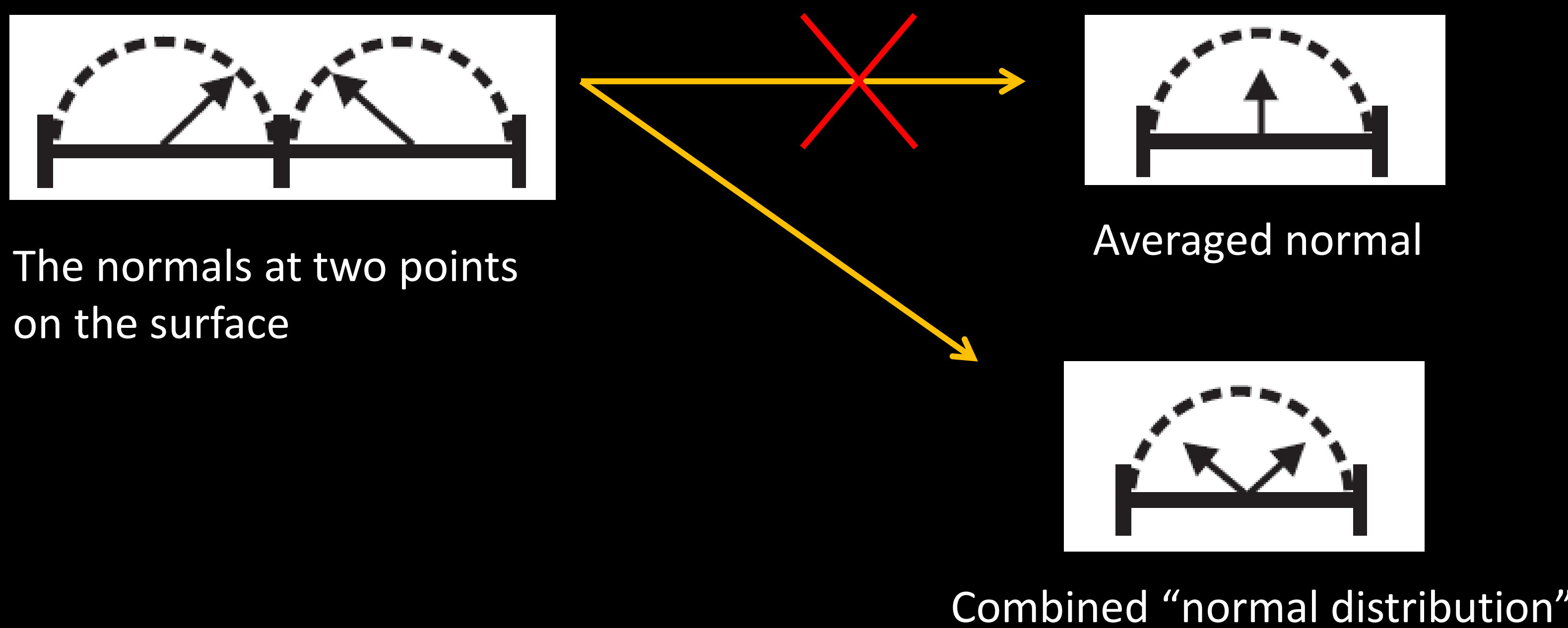
BRDF is a nonlinear function of normal

BRDF Map Filtering is NOT Linear



Correct Image

Linear Filtering


The normals at two points on the surface
Averaged normal
Combined “normal distribution”

Normal Distribution Funtion

Let's make assumptions on BRDF:

Normal Distribution Function (NDF)
h: half vector of l and v

$$\rho_x(\mathbf{l}, \mathbf{v}) = r(\mathbf{l}, \mathbf{v}) \frac{f_{\mathbf{x}}(\mathbf{h})}{\mathbf{n_x l}}$$

Does not depend on the surface property

Canceled in the equation

Normal Map Filtering

Want:

$$\rho_A(\mathbf{l}, \mathbf{v}) = \int_A \hat{w}(\mathbf{x})\, \rho_x(\mathbf{l}, \mathbf{v})\, \mathbf{n_x l}\, \mathrm{d}\mathbf{x}$$

$$= r(\mathbf{l}, \mathbf{v})\, f_A(\mathbf{h})$$

Where

$$f_A(\mathbf{h}) = \int_A \hat{w}(\mathbf{x})\, f_{\mathbf{x}}(\mathbf{h})\, \mathrm{d}\mathbf{x}$$

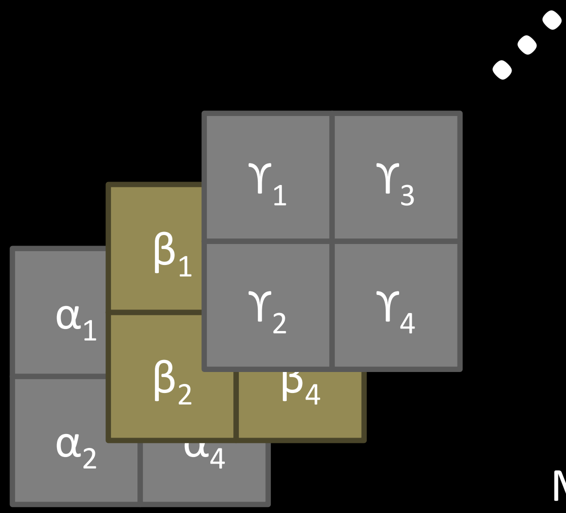
Want to find a parameterization of f_x so it's linear

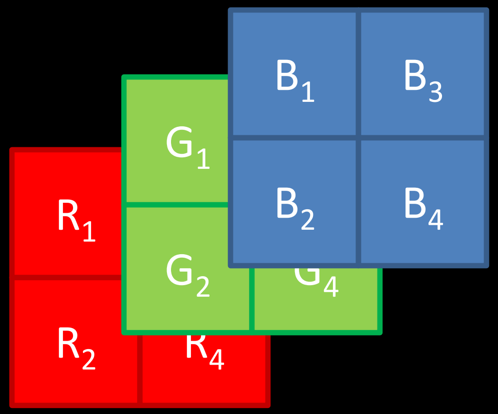
Normal distribution parameters map

Color map

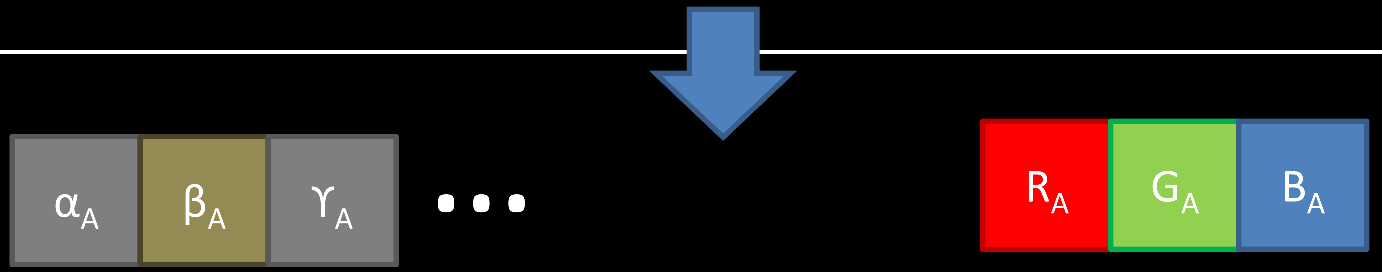
$$f_{\mathbf{x}}(\mathbf{h}) = f(\alpha_{\mathbf{x}}, \beta_{\mathbf{x}}, \gamma_{\mathbf{x}}, \ldots, \mathbf{h})$$

$$\mathbf{k}_{\mathbf{x}} = (R_{\mathbf{x}}, G_{\mathbf{x}}, B_{\mathbf{x}})$$





MIP-map filtering



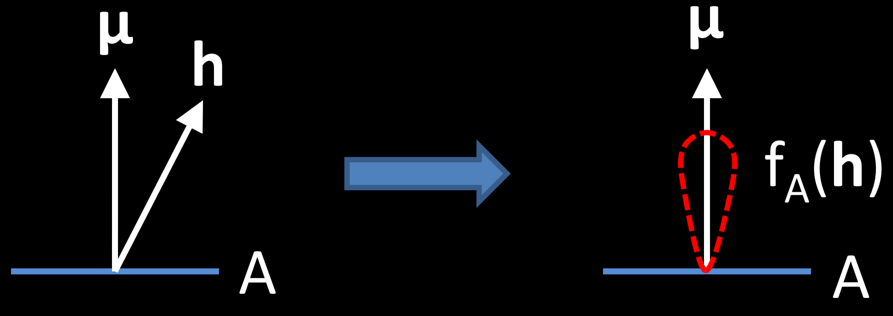
$$f_A(\mathbf{h}) = f(\alpha_A, \beta_A, \gamma_A, \ldots, \mathbf{h})$$

$$\mathbf{k}_A = (R_A, G_A, B_A)$$

Direct Method: 3D Gaussian

- [Olano and North, 1997]

$$f_A(\mathbf{h}) \propto \exp\left(-\frac{1}{2}(\mathbf{h}-\mu)^T \Sigma^{-1} (\mathbf{h}-\mu)\right)$$



Direct Method: 3D Gaussian

$$f_A(\mathbf{h}) \propto \exp\left(-\frac{1}{2}(\mathbf{h}-\mu)^T \Sigma^{-1} (\mathbf{h}-\mu)\right)$$

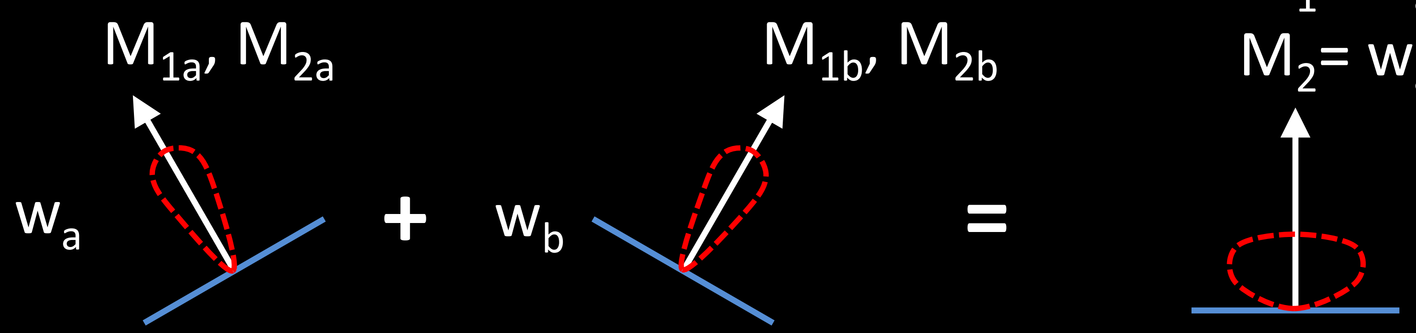
First moment M_1: μ

Second moment M_2: $\Sigma + \mu\mu^{\mathbf{T}}$

} 9 linear parameters

$M_1 = w_a M_{1a} + w_b M_{1b}$

$M_2 = w_a M_{2a} + w_b M_{2b}$



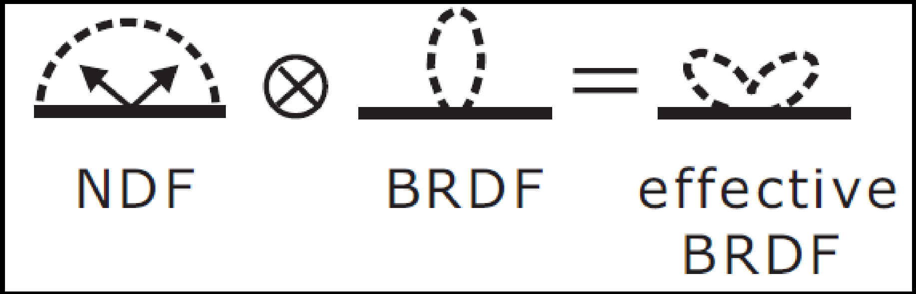
Convolution Method

$$f_A(\mathbf{h}) = \int_A \hat{w}(\mathbf{x})\, f_{\mathbf{x}}(\mathbf{h})\, \mathrm{d}\mathbf{x}$$

$$= \int_\Omega f(\mathbf{n}, \mathbf{h})\, p_A(\mathbf{n})\, \mathrm{d}\mathbf{n}$$

Underlying BRDF

Surface normal distribution (NDF)



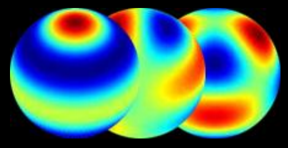
Convolution Method: Spherical Harmonics

- [Han et al., 2007] (Sec. 6)

Coefficients — Spherical Harmonics

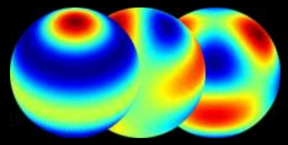
Surface NDF

$$p_A(\mathbf{n}) = \boxed{p_{lm}} \times$$



Underlying BRDF

$$f(\mathbf{n}, \mathbf{h}) = f_l \times$$

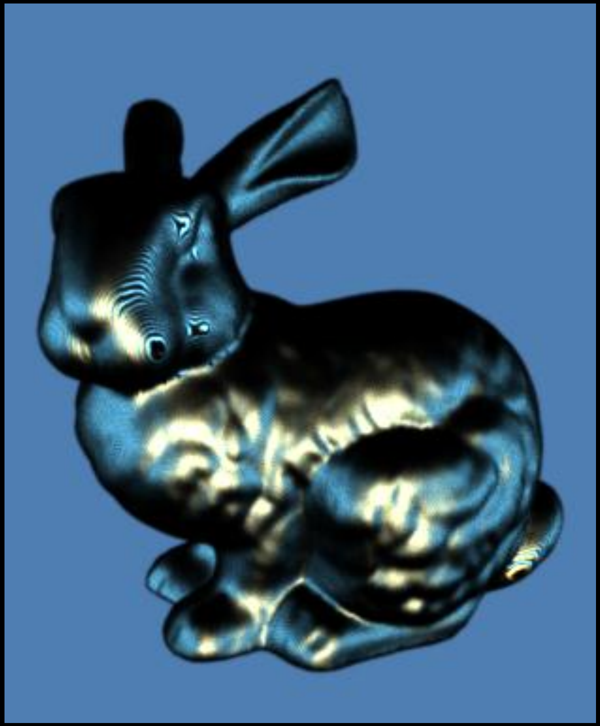


$$f_A(\mathbf{h}) = \int_\Omega f(\mathbf{n}, \mathbf{h})\, p_A(\mathbf{n})\, \mathrm{d}\mathbf{n}$$

$$= \sum_{l,m} f_l \boxed{p_{lm}} Y_{lm}(\mathbf{h})$$

- Multiplication instead of convolution
- p_{lm} can be linearly filtered (e.g., MIP-map)

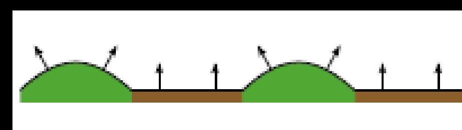
Images: http://www.maths.nottingham.ac.uk/personal/pcm/sphere/sphere.html



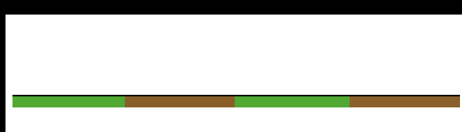
[Han et al., 2007]

Limitations

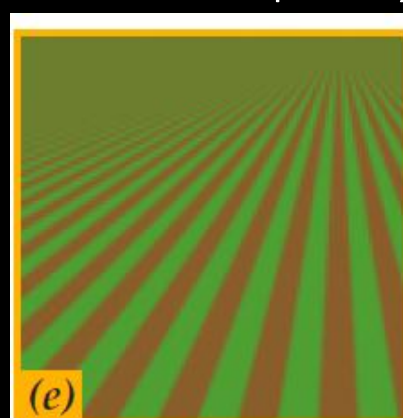
- Is the problem solved?
  - Answer: No
  - Maps are assumed to be uncorrelated
  - Parallax effect



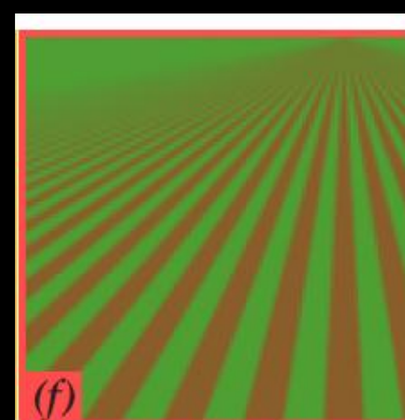
Correlated normal and color



Color filtered separately



Ground truth



Conclusion

- Correctly filtered surface maps are visually important
  - Normal, horizon, and shadow maps are non-linear
- Still far from solving the problem

Thank you