graphics and vision gravis University of Basel

Computer Graphics Basics

Probabilistic Morphable Models

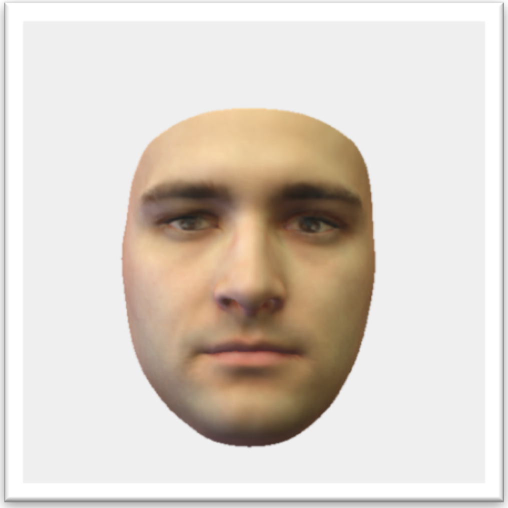
Summer School, June 2017

Sandro Schönborn

University of Basel

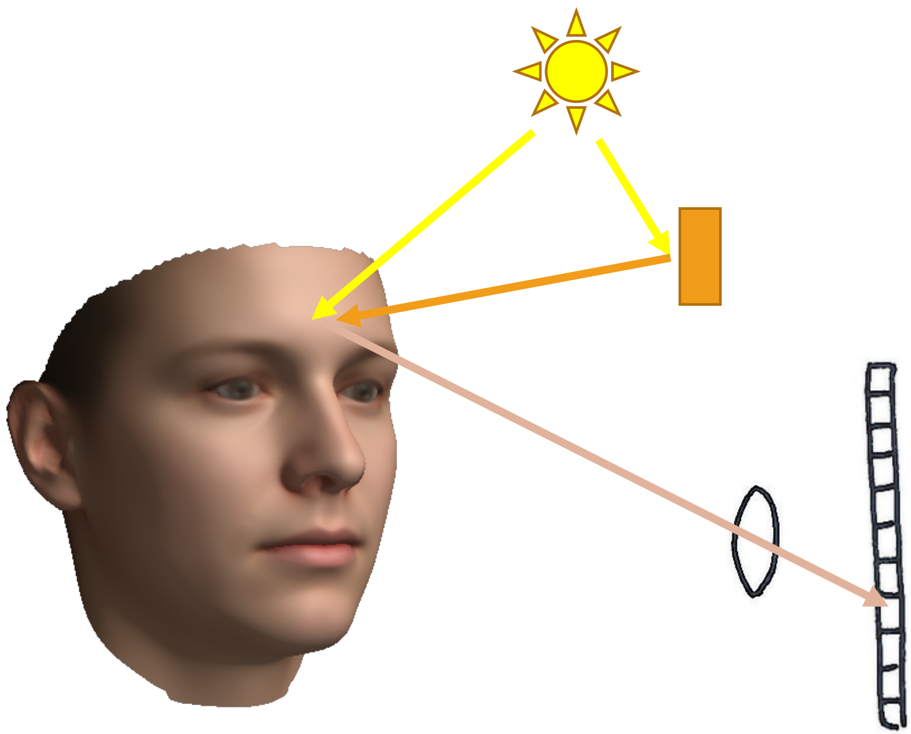
Computer Graphics

- Artificial Image Computation
- Focus: Photorealistic Rendering



- Computer graphics is more:
  visualization, non-photorealistic rendering, animation, …

Image Formation



Study of light:

- Light is emitted by source
- Light travels through space
- Light interacts with objects
- Light is reflected
- Light is refracted
- Light is captured by sensor

Computer Graphics: *Simulation of light*

Computer Graphics Compromise

- Inspired by physical cameras
  - Light and matter interaction
  - Light propagation
- Optimal goal:
  Simulation of physical reality
- Unrealistic! Infeasible
  - The perfect model?
  - Unknown parameters
  - Computational capacity

Compromise:

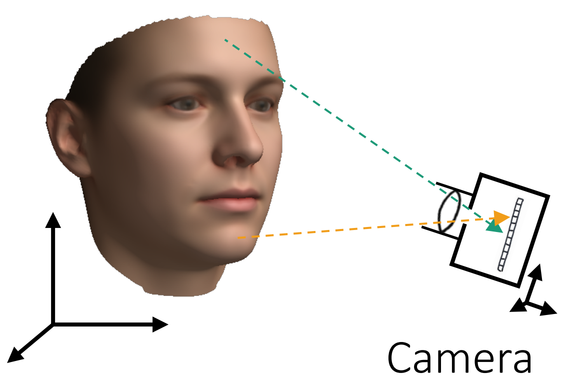
- Models to achieve results which are *good enough*
- Finding good-looking and simple *approximations*
- Simple models
  - Surface rendering (volume, interacting media, ...)
  - Lambert and Phong reflectance

Rendering

Geometry: *Correspondence*

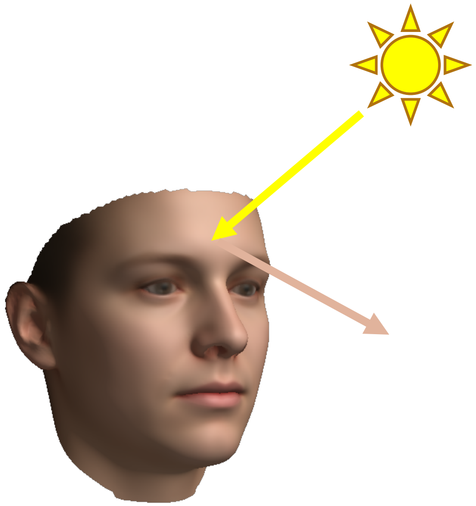
- Light transport & refraction
- Scene setup
- Correspondence

  Image point ↔ face point



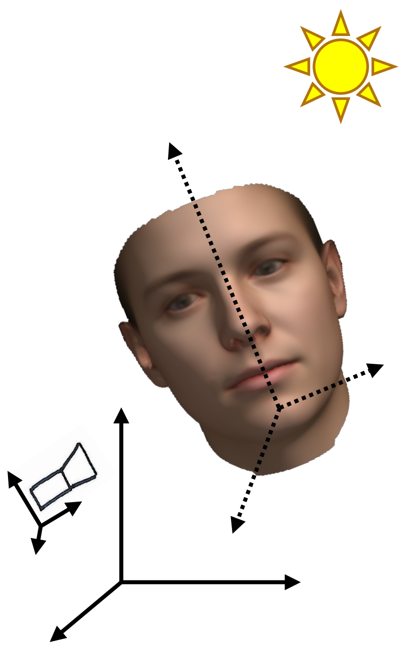
Shading: *Value*

- Light-matter interaction
- Color values
- Needs correspondence

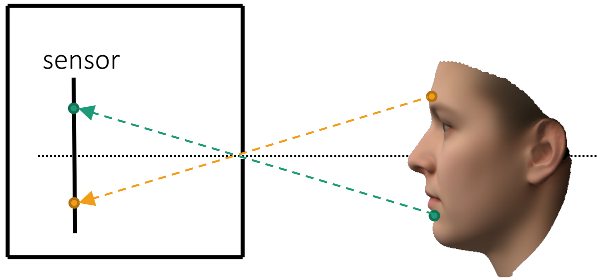


Geometry

Object
Mesh

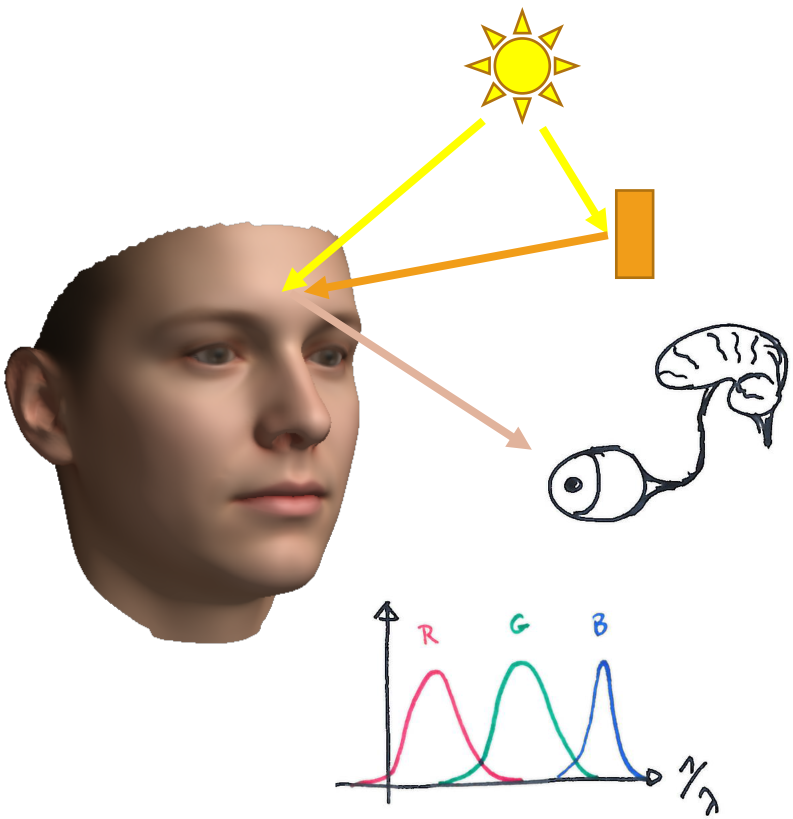


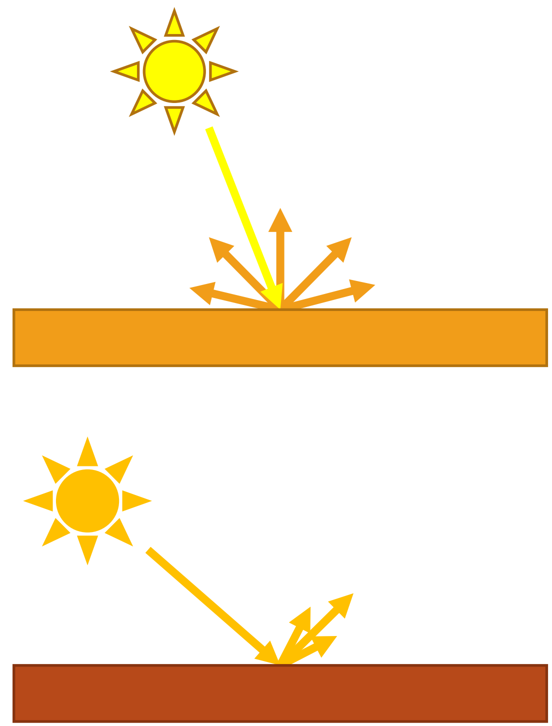
Coordinate transforms
Model, View transform



Camera model
Projection

Shading: Light-Matter Interaction



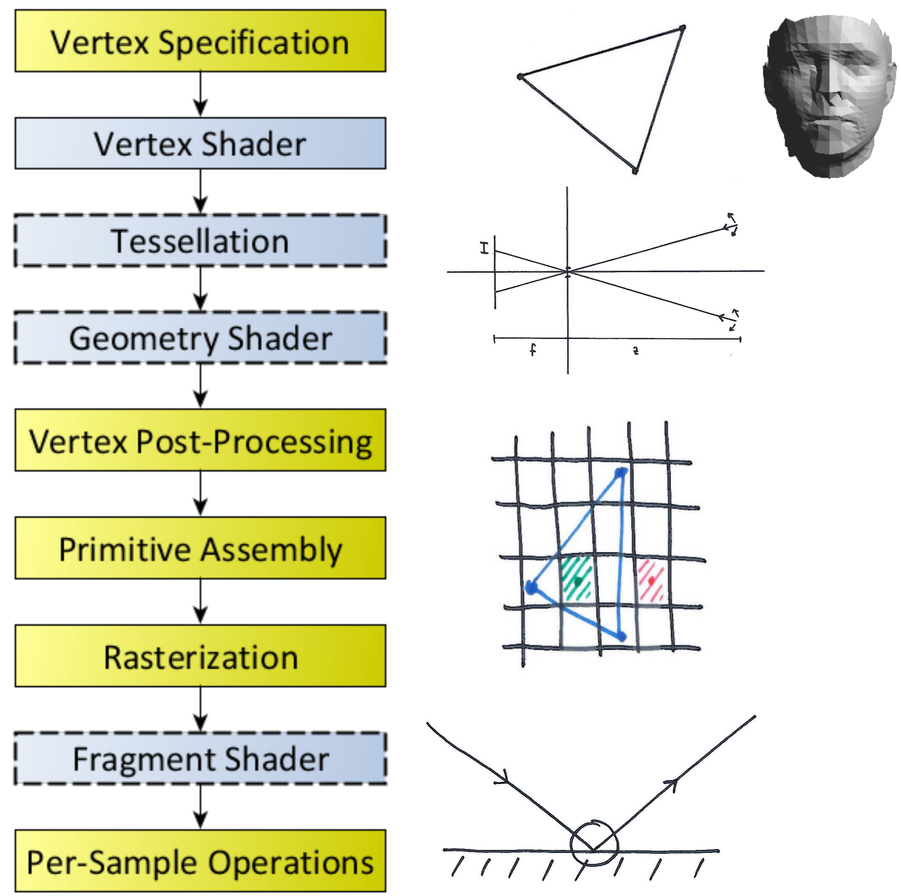


Reflectance Models

Transform incoming light into outgoing light

Modern Graphics Pipeline

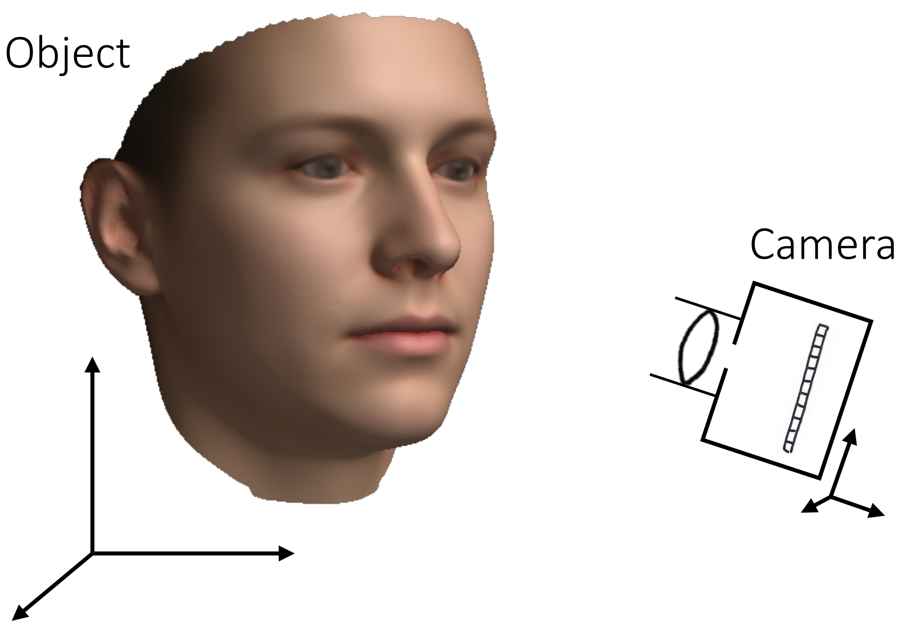
- Common design
- Specialized hardware
- Efficient, parallel
- Programmable: *Shaders* (blue parts)
- *OpenGL (ES, WebGL), Direct3D, Vulkan, Metal*



In scalismo-faces:
Close to standard design
fully controllable

Geometry

3D Scene



Multiple coordinate systems!

- 3D scene

  Objects in a world

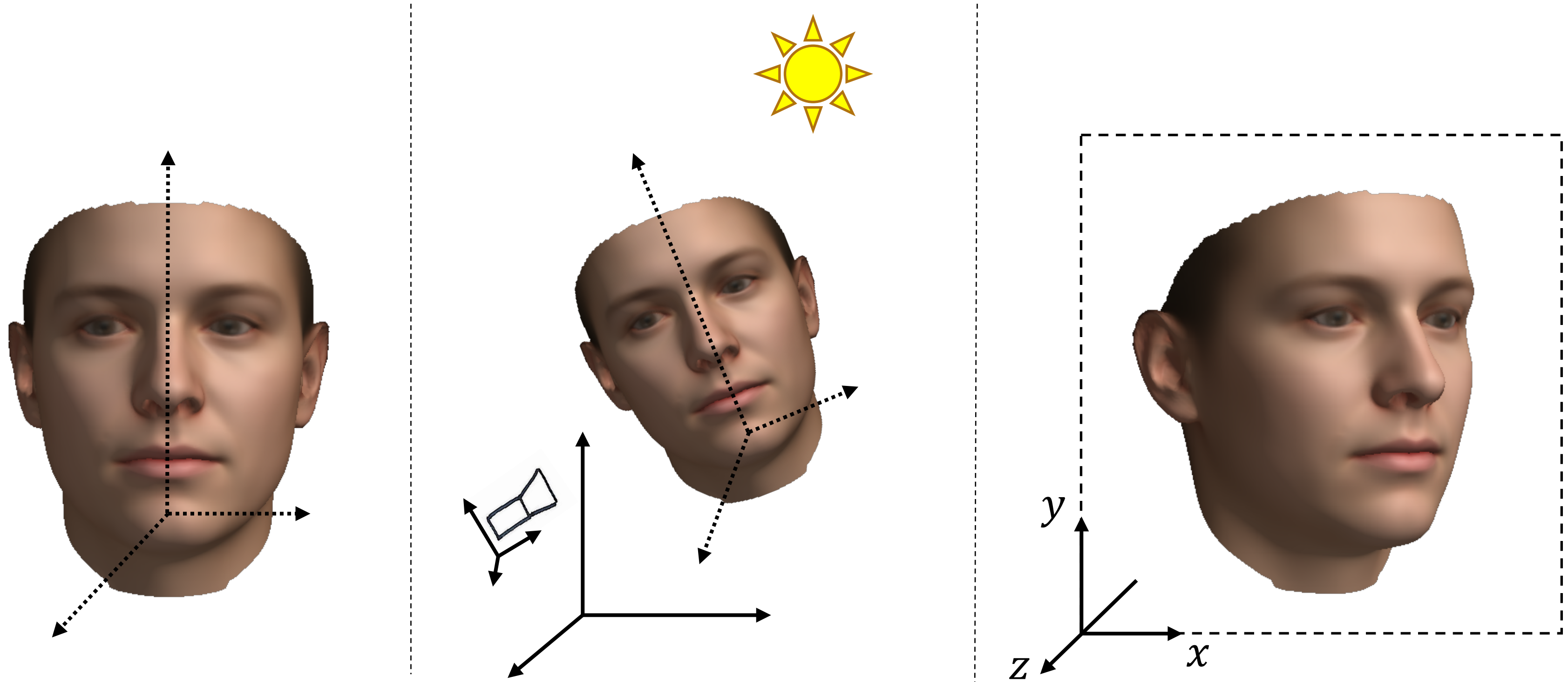
- Camera *takes* the picture

  Image lives on *image plane*

Typically 4 steps to image:

Model Transform
View Transform
Projection
Viewport Transform

Model and View Transform



Model

Model Transform T_M

World
Reference frame

View Transform T_V

Eye
Camera frame

Pinhole Camera: Perspective Projection

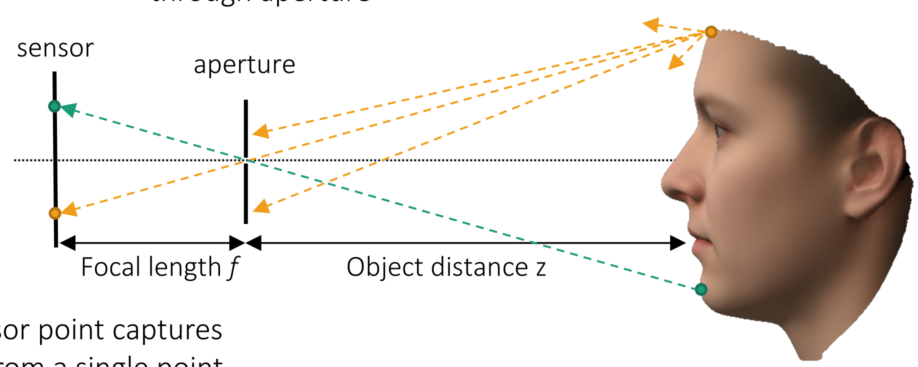
- Image formation on sensor
  image plane (3D -> 2D)
- Condition for sharp image:
  A sensor pixel captures light from a single point in scene
- Image plane coordinates by perspective division:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = f * \begin{bmatrix} x/z \\ y/z \end{bmatrix}$$

- Non-linear division operation

Pinhole camera

Single point aperture



Perspective Effect

- Perspective division distorts image non-linearly
- Effect depends on relation of object size and distance

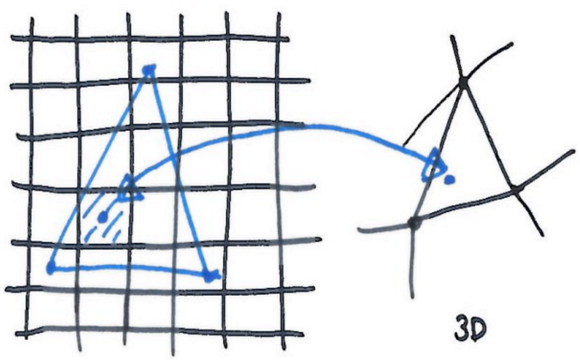
Our Transformations

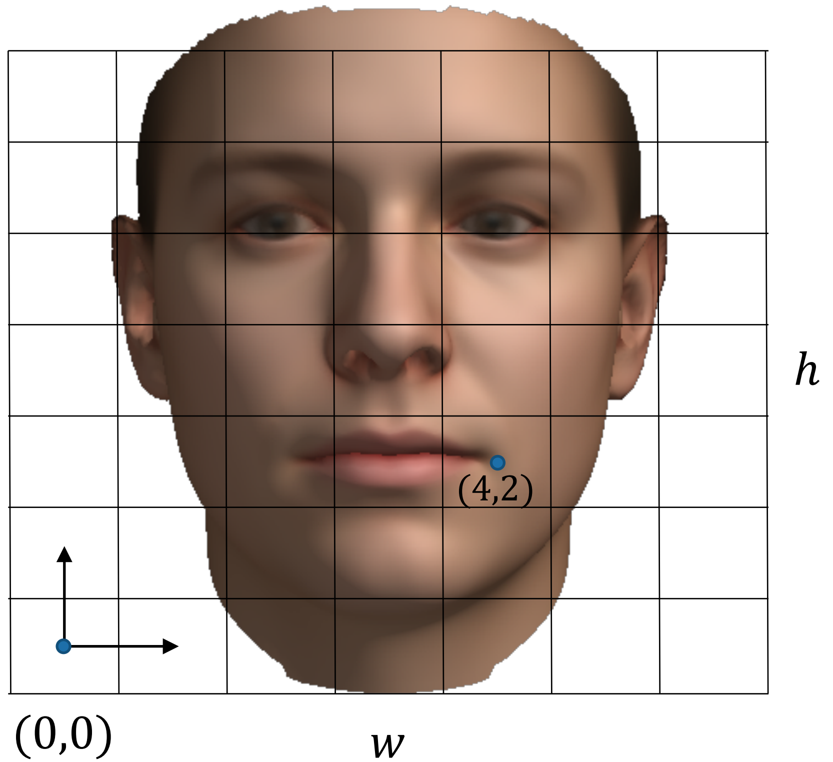
- Model-View
  $$T_{MV}(x) = R_{\varphi,\psi,\vartheta}(\boldsymbol{x}) + \boldsymbol{t}$$
- Projection
  $$\mathcal{P}(x) = \frac{f}{z}\begin{bmatrix} x \\ y \end{bmatrix}$$
- Viewport
  $$T_{VP}(x) = \begin{bmatrix} \frac{w}{2}(x+1) \\ \frac{h}{2}(1-y) \end{bmatrix} + \boldsymbol{t}_{pp}$$

- Describes our face-to-image transform completely
- 9 Parameters:
  - (3) Translation $\boldsymbol{t}$
  - (3) Rotation φ, ψ, ϑ
  - (1) Focal length f
  - (2) Image Offset $\boldsymbol{t}_{pp}$
- 2 Constants:
  - (2) Image size / sampling

Rasterization

- Camera: 3D → 2D transformation for *points*
- Raster Image in image plane
- Establishes correspondence to 3D surface for each *pixel*
- Basis: geometric *primitives*

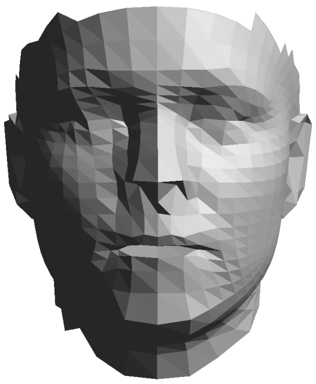


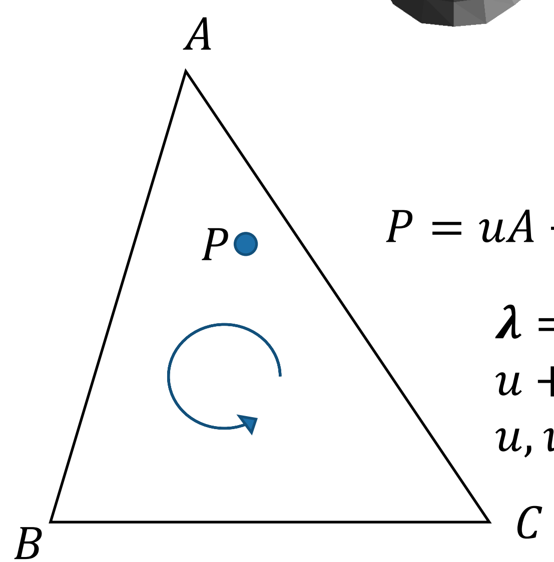


Pixel grid, cell-centered

Primitives: Triangles

- Triangle meshes for surface *parametrization:*
  - Triangle
  - Position within triangle
- Parametrization within triangle
  - Barycentric coordinates $\boldsymbol{\lambda}$
- Barycentric interpolation
  $f(P) = uf(A) + vf(B) + wf(C)$
- In/out Test:
  - All BCC valid (non-negative)



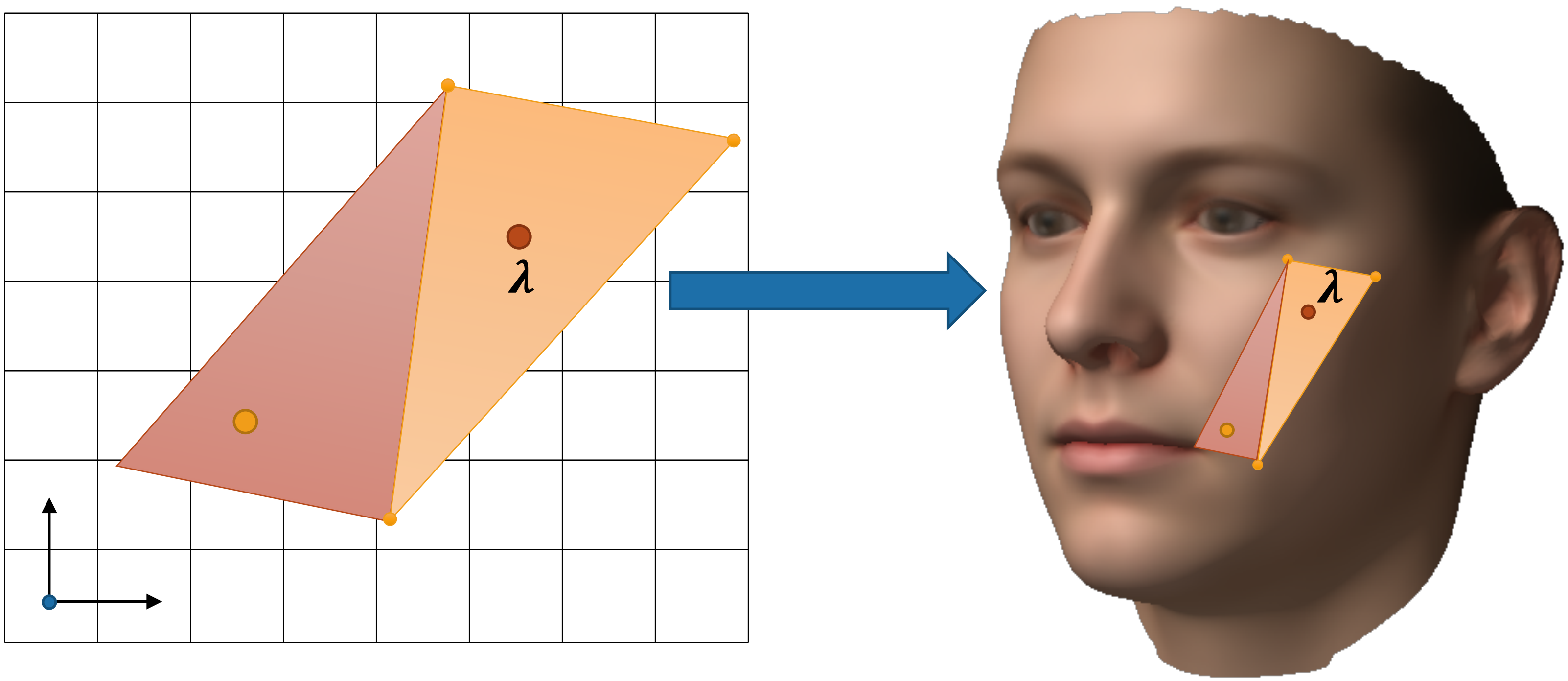


$$P = uA + vB + wC$$

$$\boldsymbol{\lambda} = (u, v, w)$$
$$u + v + w = 1$$
$$u, v, w \geq 0$$

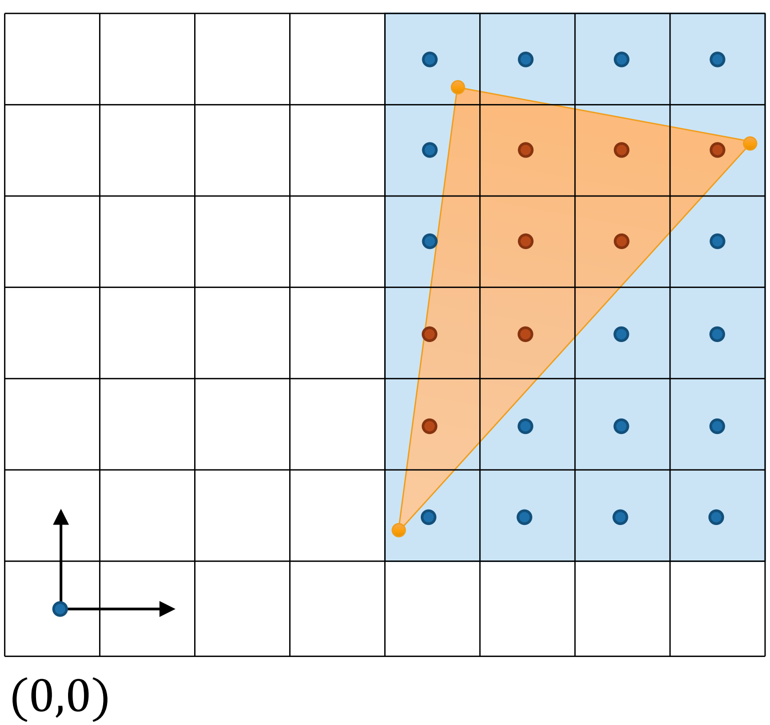
counter-clockwise winding!

Rasterization: Correspondence



- Each image *pixel* is mapped to surface point
- Point identification by parameterization with *triangle* and *barycentric coordinate*

Rasterization of Triangle Primitives



Vertex shader

- For each triangle:
  - Find Vertex position (corners)
  - Determine bounding box
  - For all pixels in box:
    - Inside triangle?
      - Find BCC in plane: *correspondence* to 3D through BCC
      - *Draw* the pixel

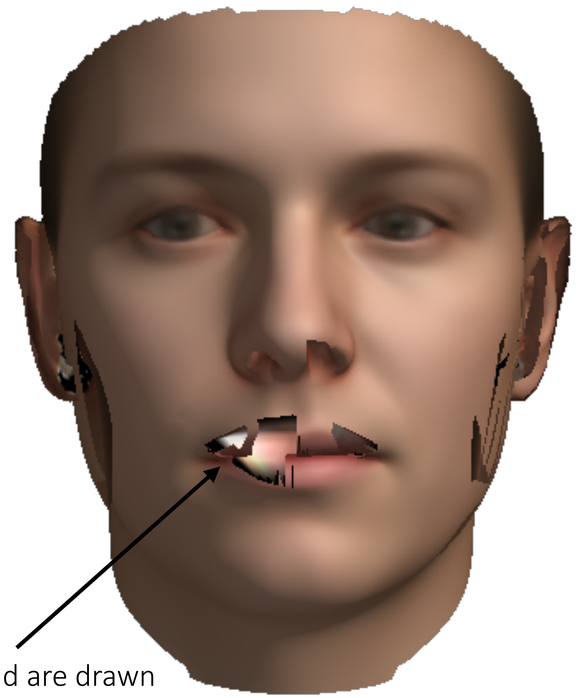
Efficient! No ray intersections (not perfect though)

Fragment shader

Visibility Issues

- Multiple triangles might cover the same pixel
- 3D surface *occludes* background
- Only the most *frontal* part is visible in the image
- Needs special care during rendering:

  Hidden Surface Removal

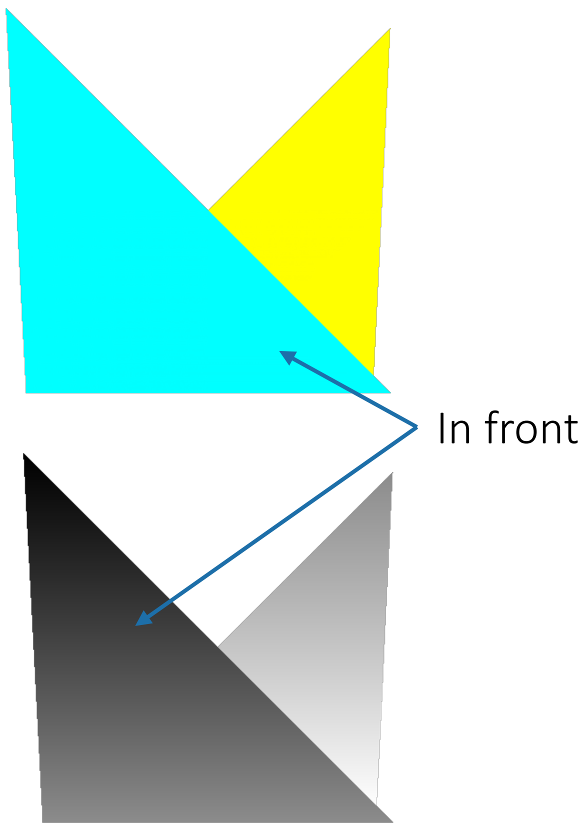


Triangles behind are drawn on top of those in front

Hidden Surface Removal: Z-Buffer

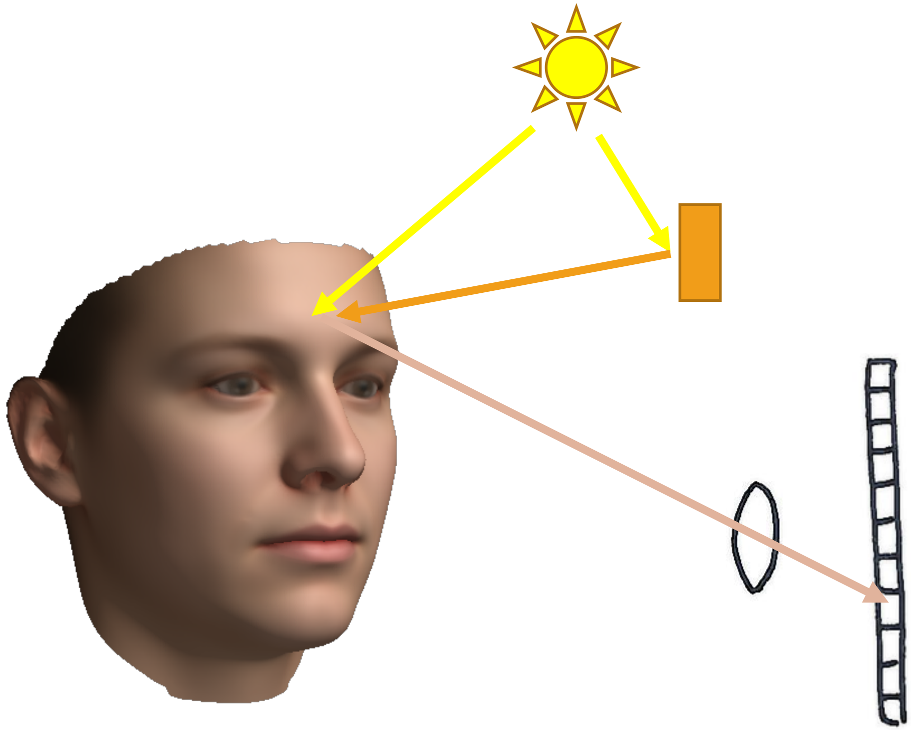
- Keep additional *Z-Buffer:*
  - Store depth information for each pixel
  - Draw a pixel only if it lies *in front* of previous drawing
- Standard approach
- Easy and versatile
  - Extensible to shadowing
- Issues:

  Precision, single value per pixel



Illumination

Illumination

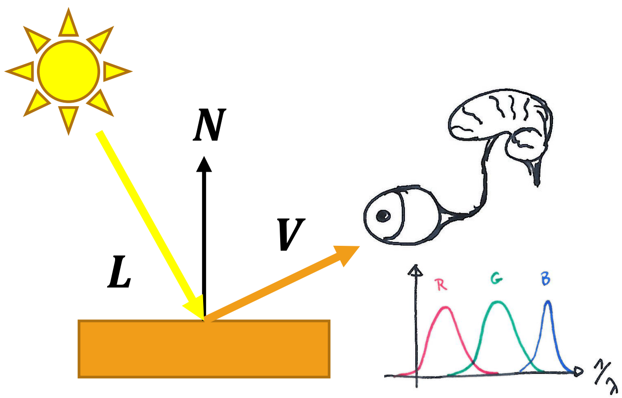


- Color in image is result of *light* and *surface* interactions
- Shading: *simulate* light interaction and transport
- Illumination is global:

  Lights scatters through scene, interaction with many objects

  - Global transport
  - Local interaction

Reflectance Models: BRDF



Bidirectional Reflectance Distribution Function

$$f(\lambda_i, \boldsymbol{L}, \lambda_r, \boldsymbol{V}, \boldsymbol{x}) = \frac{\mathrm{d}L_r(\boldsymbol{V})}{\mathrm{d}E_i(\boldsymbol{L})}$$

incoming light (*irradiance*) into outgoing light (*radiance*)

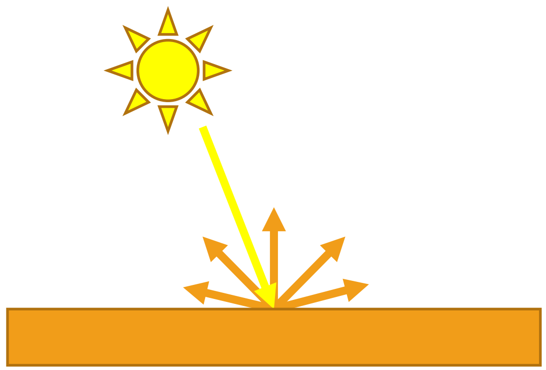
Geometry

- $\boldsymbol{L}$ Light direction
- $\boldsymbol{N}$ Surface normal
- $\boldsymbol{V}$ Viewing direction

Spectrum

- Albedo (color)
- Eye and most cameras: 3 color sensor types
- *RGB* Model: spectral distribution is sampled for *red*, *green* and *blue*
  $c = [r, g, b], \qquad r, g, b \in [0,1]$

Lambertian Reflectance



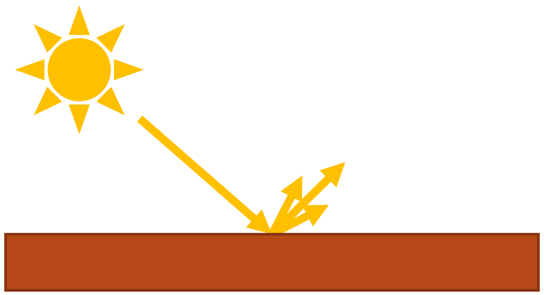
$$I_{\text{diff}} = k_{\text{diff}} * I_L * \cos(\boldsymbol{L}, \boldsymbol{N})$$

Observed intensity — Reflection coefficient — Light intensity

- Diffuse Reflectance
- Surface absorbs all radiation and reemits into every direction
  - Not directional
  - Constant BRDF
- Brightness of surface depends on incident energy
- Deep surface interaction: *Albedo:* k_{diff} (colored)

Lambertian Reflectance: Examples

Specular Highlights



$$I_{\text{spec}} = k_{\text{spec}} * I_L * \cos(\boldsymbol{R}, \boldsymbol{V})^n$$

- Mirror-like reflectance
  - Highly directional
- Reflection *cone* due to surface roughness
- Mostly without deep surface interaction

  k_{spec} not colored
- Parameter:

  n *Phong exponent*

  Width of specular cone

Phong Specularity: Examples

Specular reflection

Specular & diffuse

Global Illumination

Illumination is global in scene

Approximation levels:

- Ambient Light

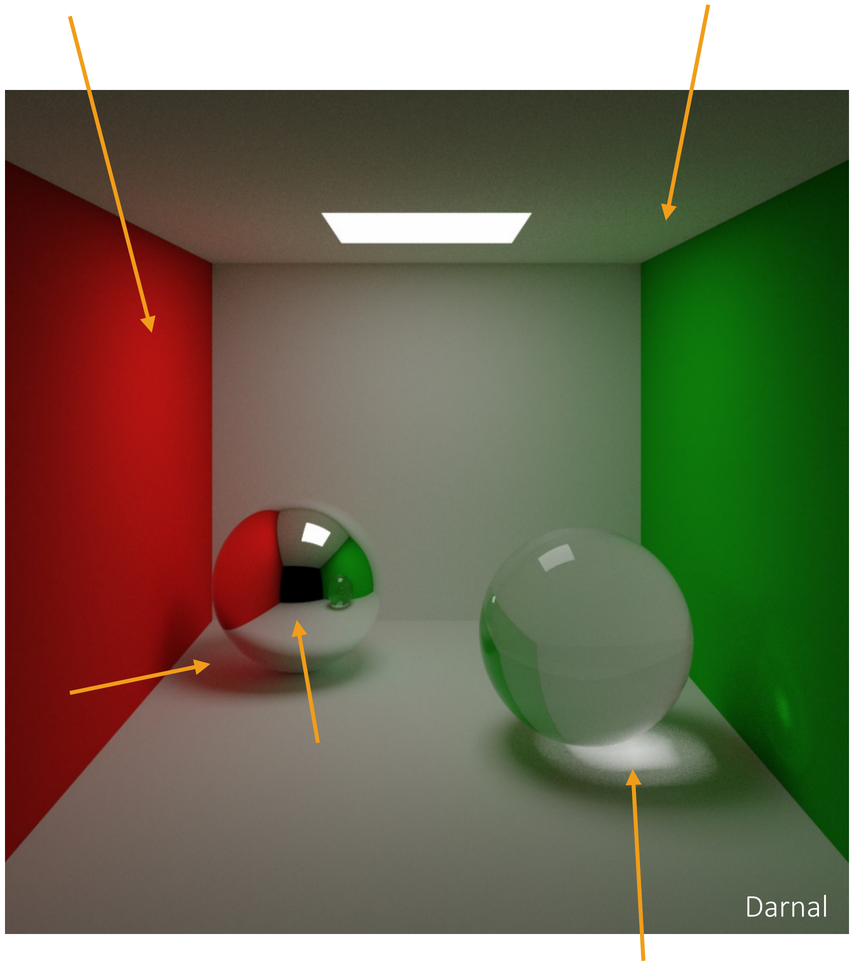
  Model scattered light as *constant average* value throughout scene

- Environment Map

  Incoming light intensity for each direction (empirically captured)

- Real global illumination

  Calculate light *scattering* through scene (extremely expensive)



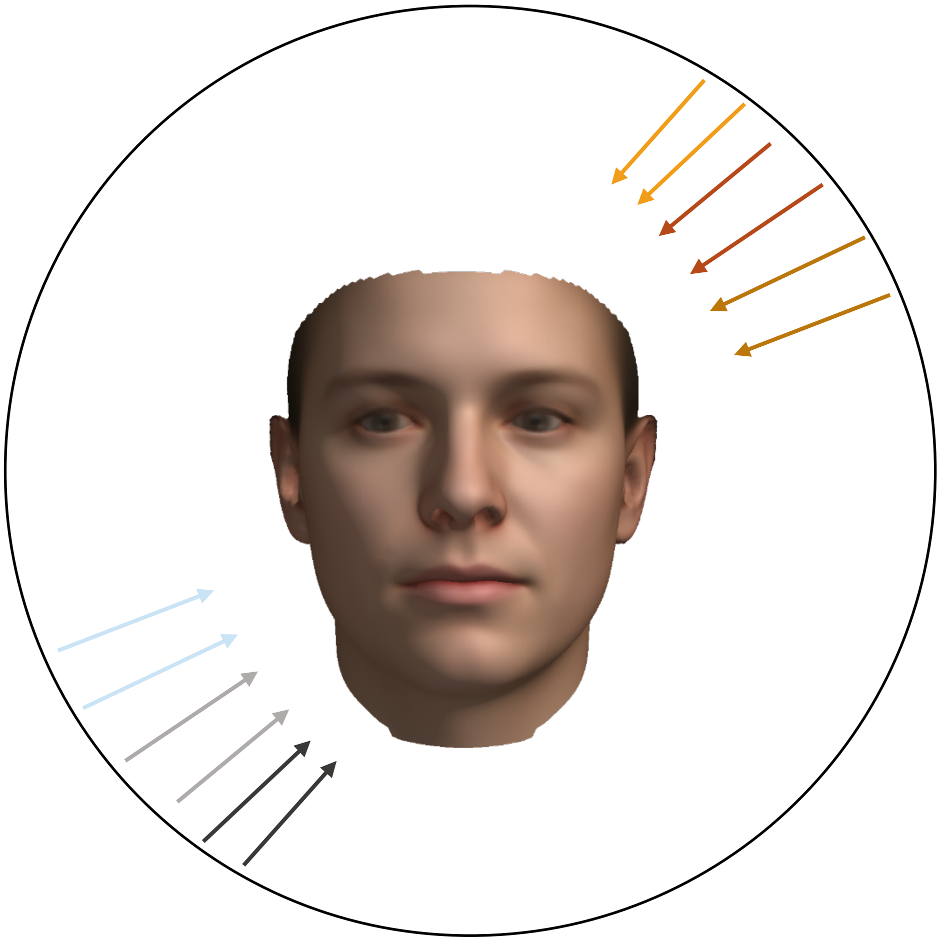
Phong Illumination Model

- Combination of three illumination contributions:
  - Lambert (diffuse) $k_{\text{diff}} * I_L * \cos(L, N)$
  - Specular $k_{\text{spec}} * I_L * \cos(\mathrm{R}, \mathrm{V})^n$
  - Ambient (global) $k_{\text{amb}} * I_A$
- Ambient is a scene *average* light intensity I_A
- Lambert and specular part for each light source

$$I' = k_{\text{amb}} * I_A + k_{\text{diff}} * I_L * \cos(L, N) + k_{\text{spec}} * I_L * \cos(\mathrm{R}, \mathrm{V})^n$$

usually colored

Environment Maps

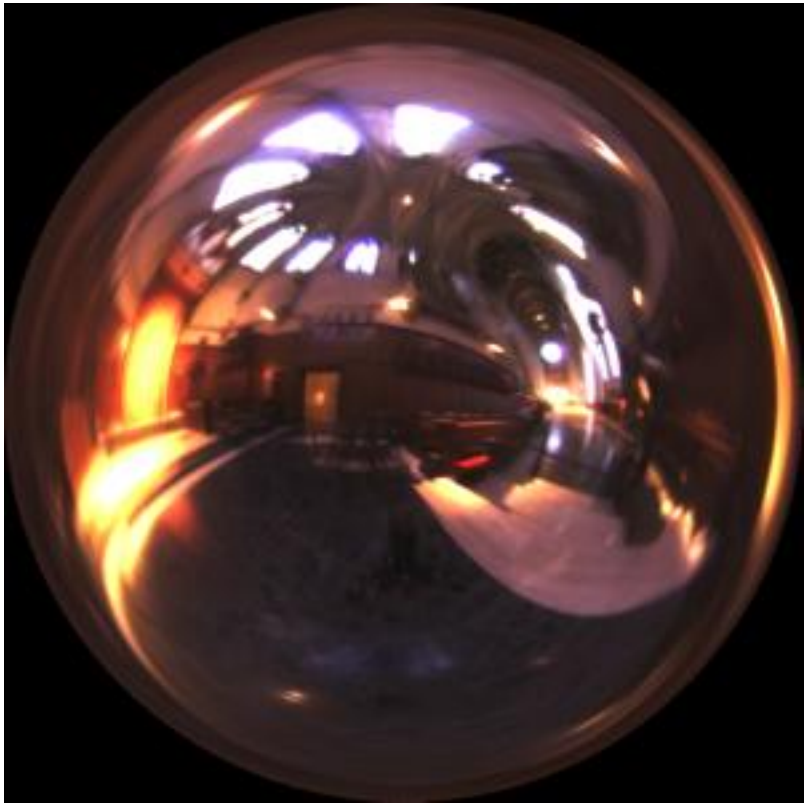


- Mapping of incoming light intensity from every direction

$$I_L^{\mathrm{RGB}}(\theta, \varphi)$$

- Modeled at infinity
- Typically *empirically* captured
- Shading with environment maps requires *integration* over all incoming directions

Environment Maps

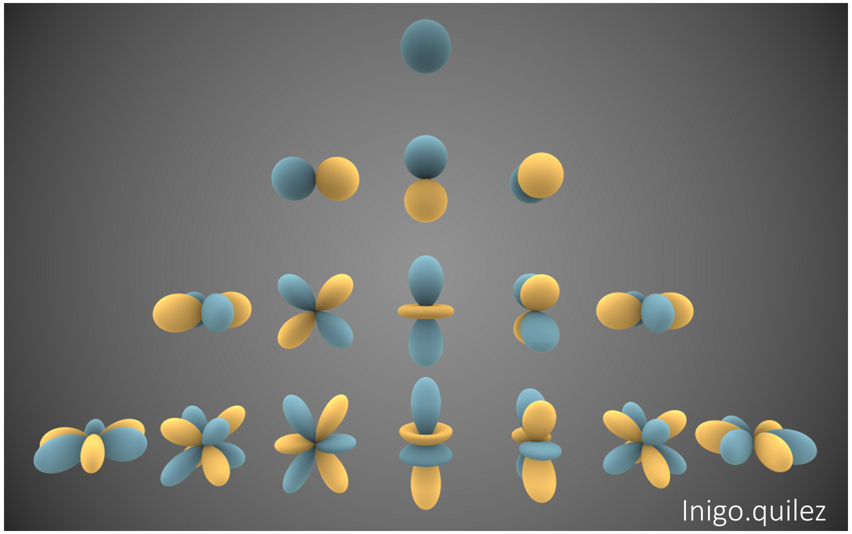


Grace Cathedral (San Francisco)
P. Debevec

White surface in Grace Cathedral

Spherical Harmonics Illumination

- Expand map $I_L^{\mathrm{RGB}}(\theta, \varphi)$ with *basis* functions
- Choose *Spherical Harmonics*: Eigenfunctions of Laplace operator on sphere surface
$$Y_{lm}(\theta, \varphi)$$
- Corresponds to Fourier transform
- Integration becomes multiplication of coefficients (→ *fast convolution*)
- Low frequency part is sufficient for Lambertian reflectance



Ramamoorthi, Ravi, and Pat Hanrahan. "An efficient representation for irradiance environment maps." Proceedings of the 28th annual conference on Computer graphics and interactive techniques. ACM, 2001.

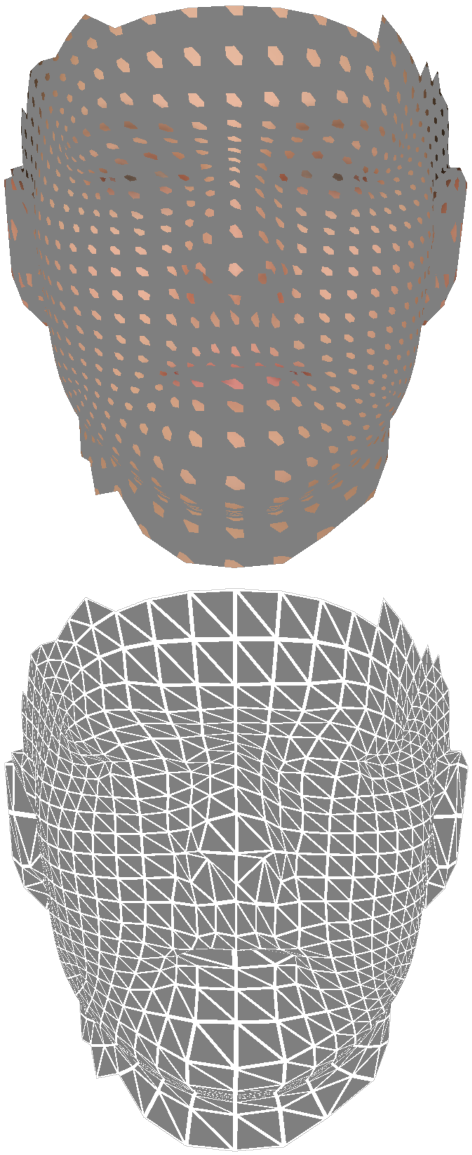
Environment Map Illumination

3DMM Rendering of Faces

3DMM Face Rendering

- Rigid Model Transform
- Pinhole camera model
- Mesh with *position* and *color* at each vertex (~albedo):

  Two independent, discrete low-rank Gaussian Processes ($\sim$30k points)
- Spherical Harmonics Illumination
  - Lambert
  - Environment map
    2 Bands, 9 coefficients x RGB



3DMM Random Faces

Summary: Rendering

- Computer Graphics:
  Artificial image computation
- Camera & Projection
  - Transformations in space and projection
  - Maps 3D space and 2D image plane
- Rasterization
  - Correspondence: image pixels ↔ surface
  - Z-Buffer: Hidden surface removal
- Shading
  - Illumination simulation
- Reflectance
  - Phong: Ambient, diffuse & specular
  - Global Illumination