https://tinyurl.com/dhcp7h3w

Computer Vision 1

Reflectance & Photometric Stereo Image Processing Basics

Dr. Martin Oswald, Dr. Dimitris Tzionas, Dr. Arun Mukundan,
(m.r.oswald, d.tzionas, a.mukundan)@uva.nl

University of Amsterdam

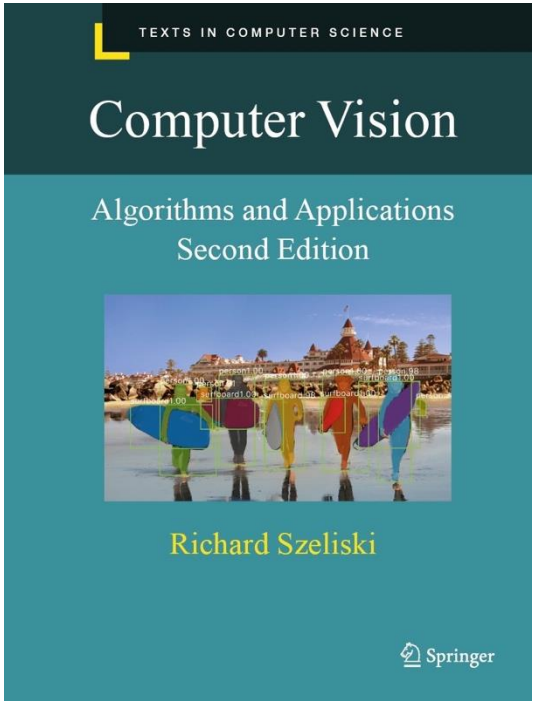
computer vision lab

Outline

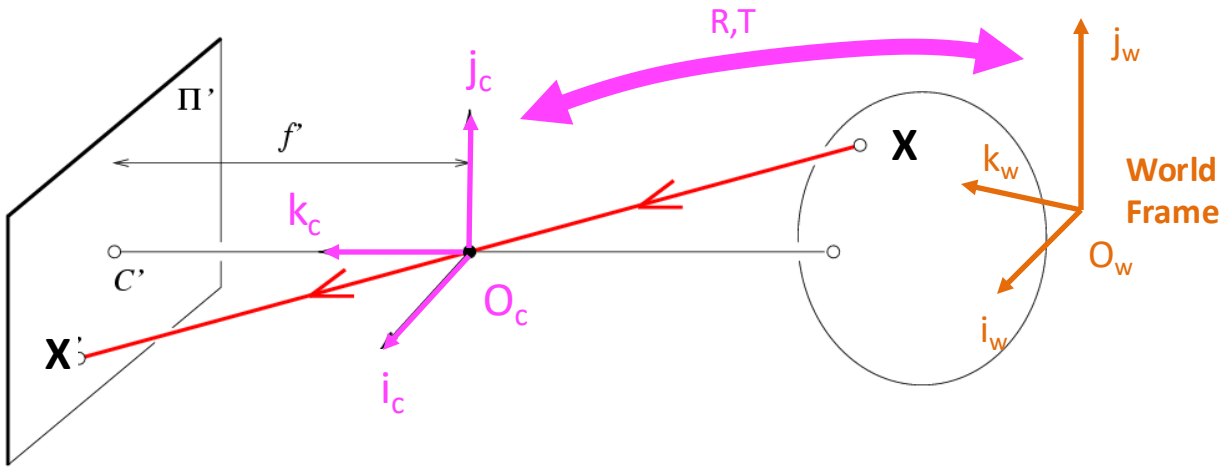
- Reflection and Photometric Stero
  - Reflection Models
  - Shape from Shading
  - Photometric Stereo
  - Intrinsic Decomposition
  - Applications
- Image Processessing Basics
  - Histogram
  - Histogram Modification
  - Histogram Equalization

Textbook

- 3.1.1
- 3.1.2
- 3.1.4
- 2.2.2



Camera Projection Matrix



Camera Frame posed (R, T) w.r.t. World Frame

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X}$$

Intrinsic Matrix

Extrinsic Matrix

O_w: Center of World Frame

O_c: Center of Camera Frame

X: 3D point in World Coordinates: (X,Y,Z,1)

x: 2D Image Coordinates: (u,v,1), up to scale w

K: Intrinsic Matrix (3x3)

R: Rotation (3x3)

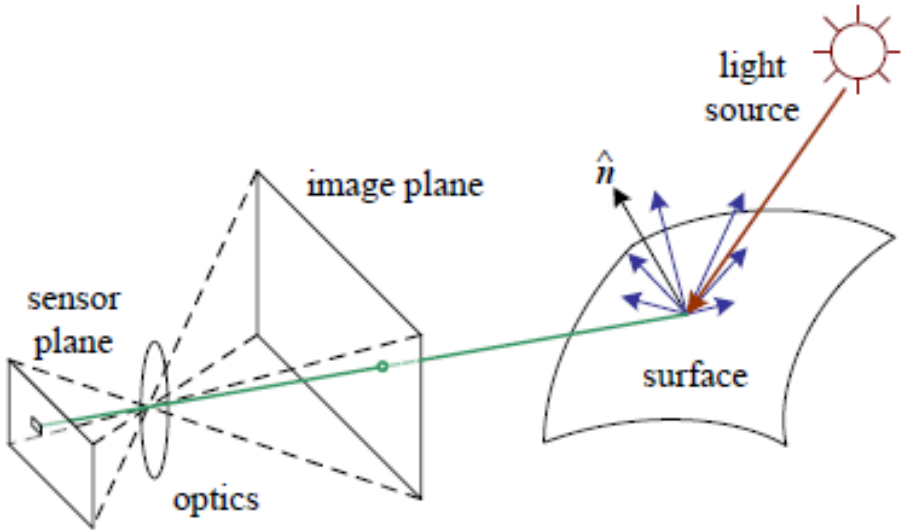
t: Translation (3x1)

Note: different books use different notation!

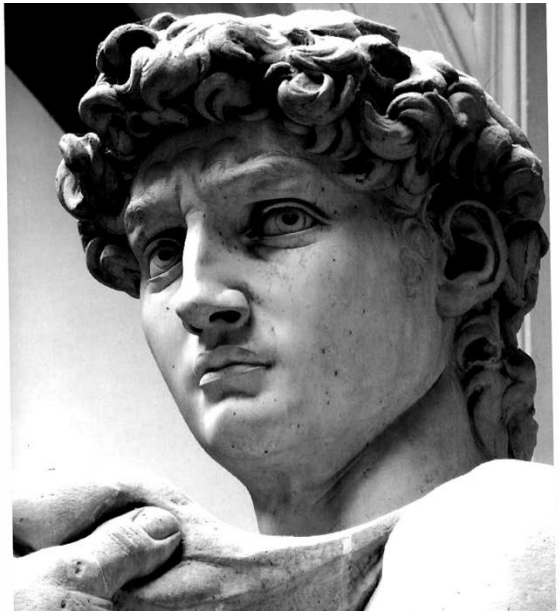
Slide Credit: S. Saverese

4 main factors which define intensity values

- the geometry of the scene
- the reflectances of the visible surface
- the illumination of the scene
- the viewpoint



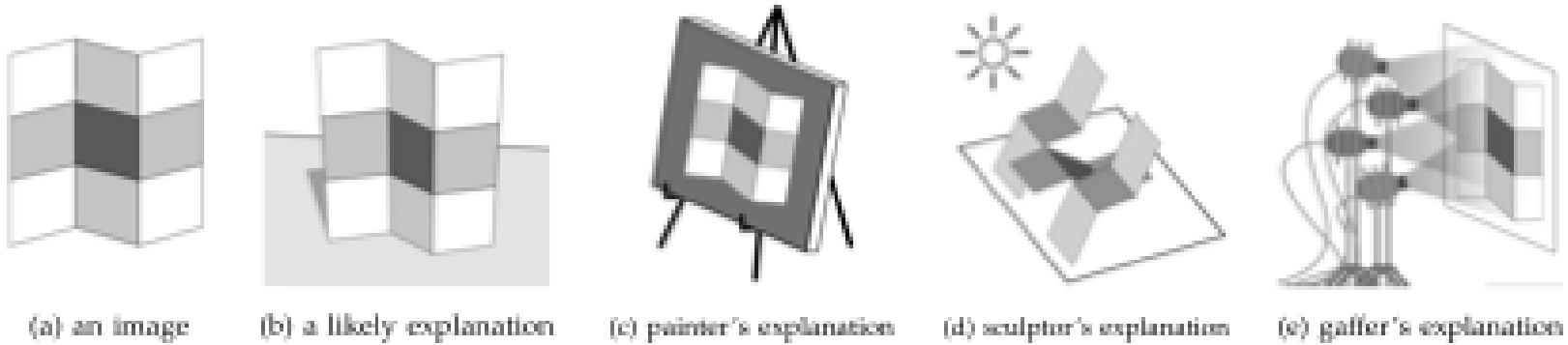
Can we recover shape from shading?



What is the relation between **intensity** and **shape**? Can we recover shape?

[Slide credit: Andreas Geiger]

Adelson and Pentland's Workshop Metaphor



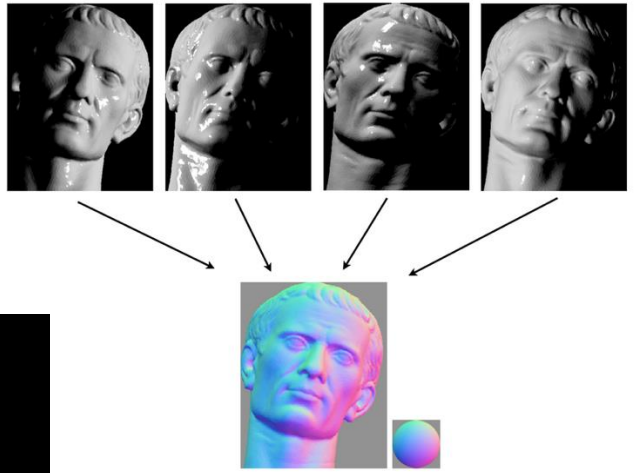
Can we recover **3D from a single image?**

- Inverting the graphics problem is very challenging
- The space of shapes, paint, and light that exactly reproduce an image is vast
- The image in (a) very likely corresponds to (b), but it could be a painting (c), sculpture (d), or an arrangement of lights (e)
- We are looking for a **simple explanation** and must use **prior knowledge**

[Slide credit: Andreas Geiger]

Motivation: Photometric Stereo

- Photometric stereo captures **multiple** images from the **same viewpoint**, but with **different light settings**
- Per-pixel estimation of **normal** and **albedo** or **material**
- Assumes far cameras/lights

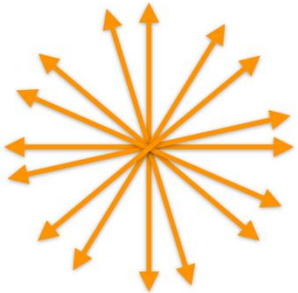


[Esper light stage]

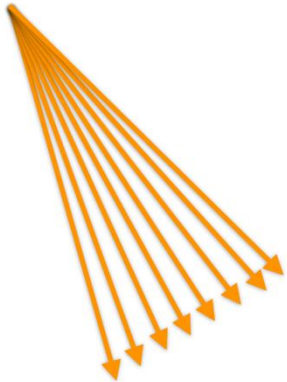
Total Relighting

[Pandey et al, SIGGRAPH'21, YouTube]

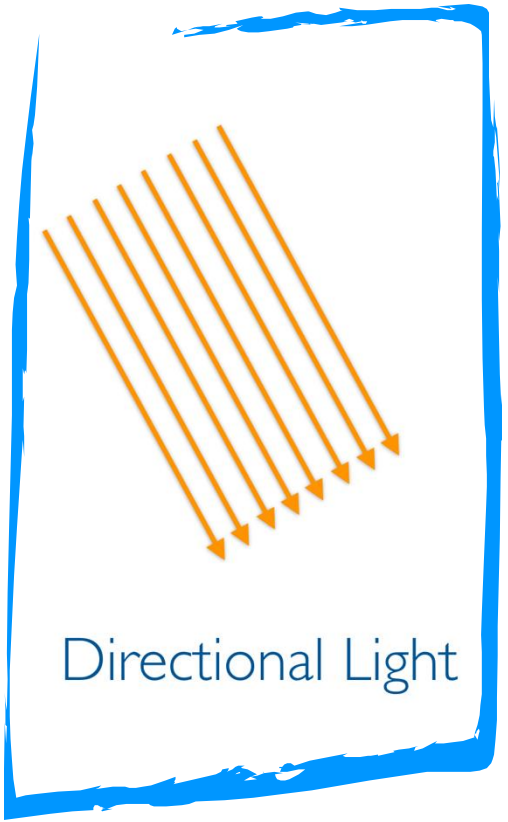
Light sources



Point Light



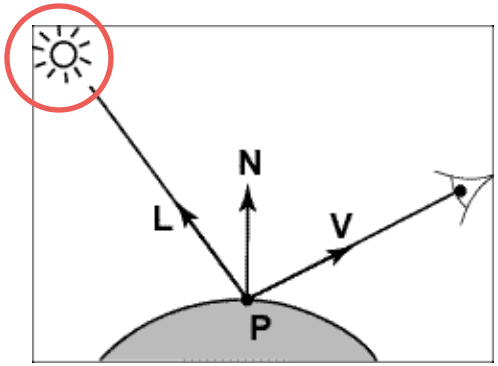
Spot Light



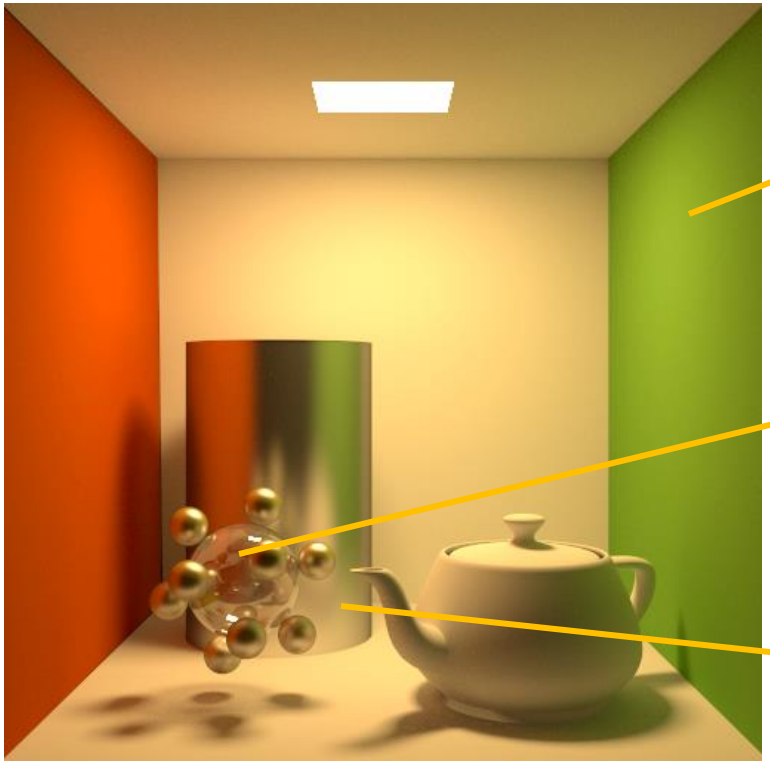
Directional Light

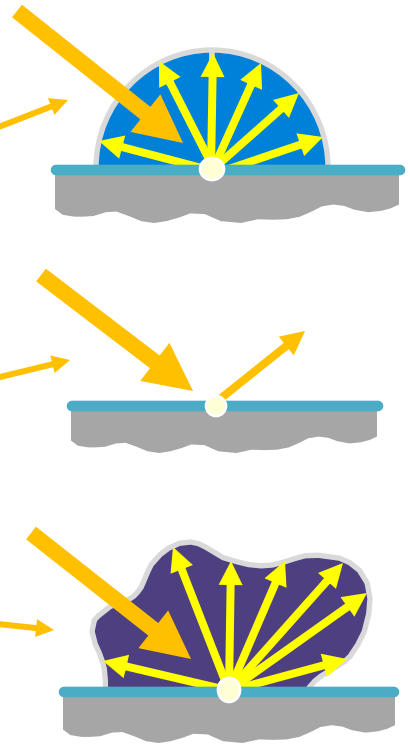
Directional Lighting and Reflectance

- Key property: all rays are parallel
- Equivalent to an infinitely distant point source



Materials - Three Forms





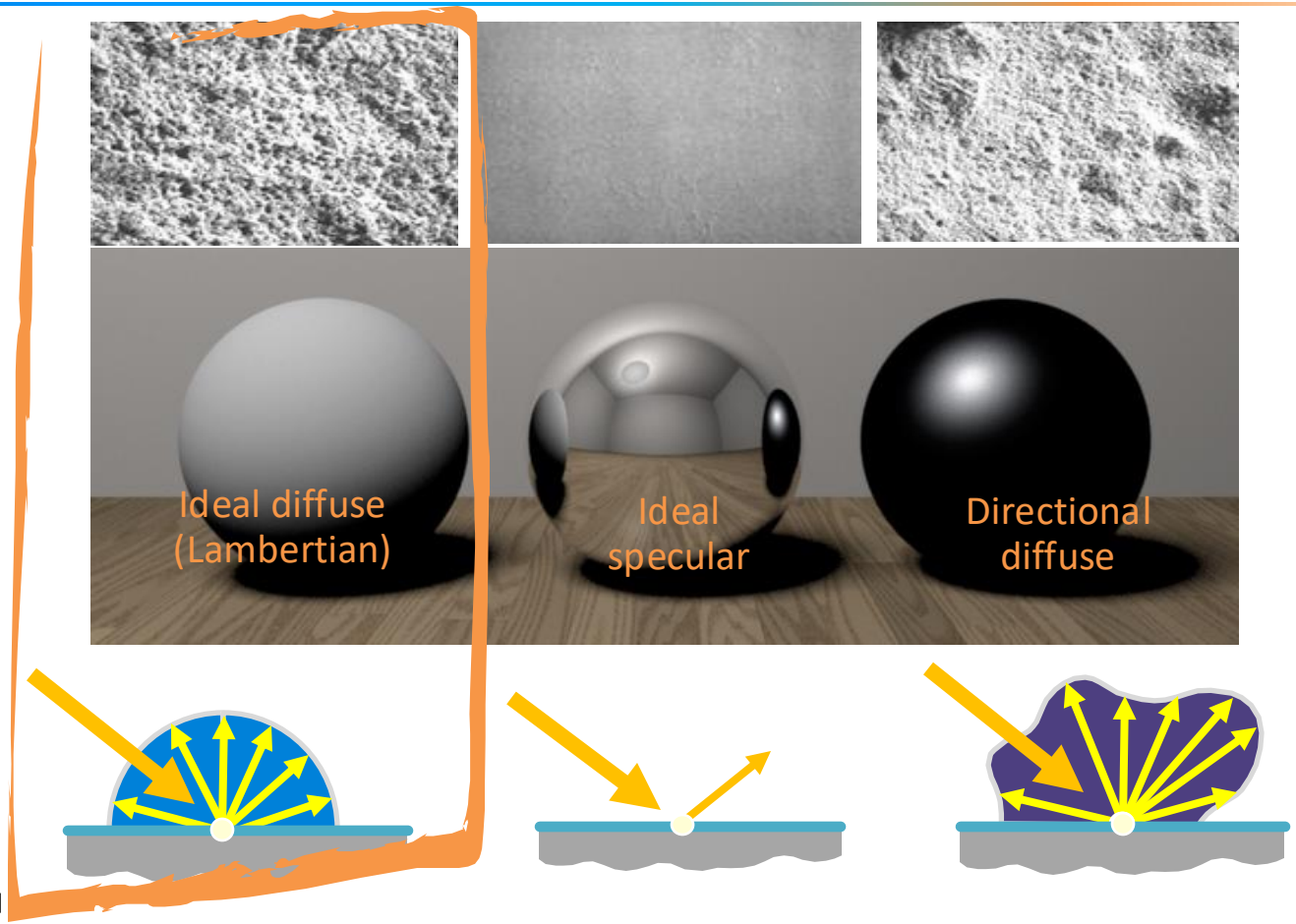
Ideal diffuse (Lambertian)

Ideal specular

Directional diffuse

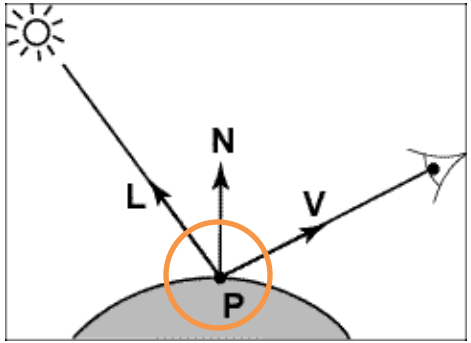
[Slide credit: Kavita Bala]

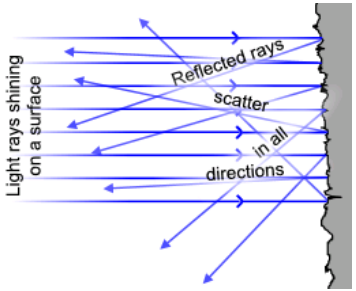
Reflectance - Three Forms

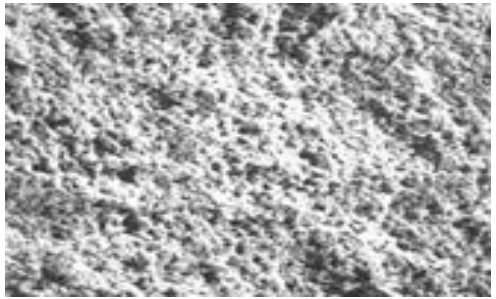


[Slide credit: Kavita Bala]

Reflectance - Three Forms



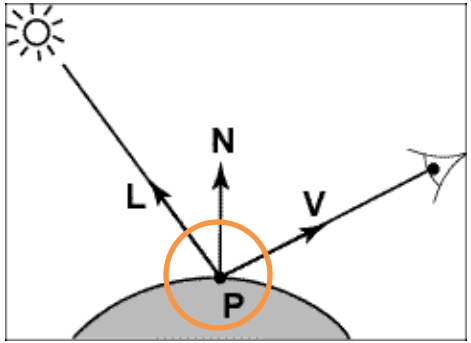


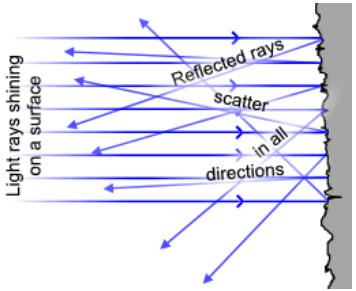


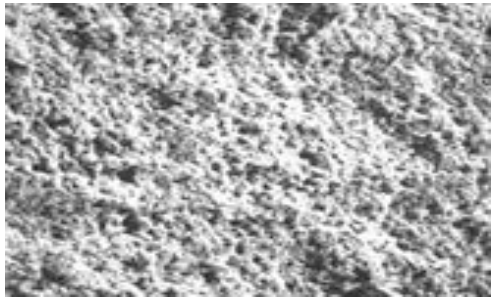
1. Reflected energy is proportional to cosine of angle between L and N **(incoming)**

2. Measured intensity is viewpoint-independent **(outgoing)**

Reflectance - Three Forms







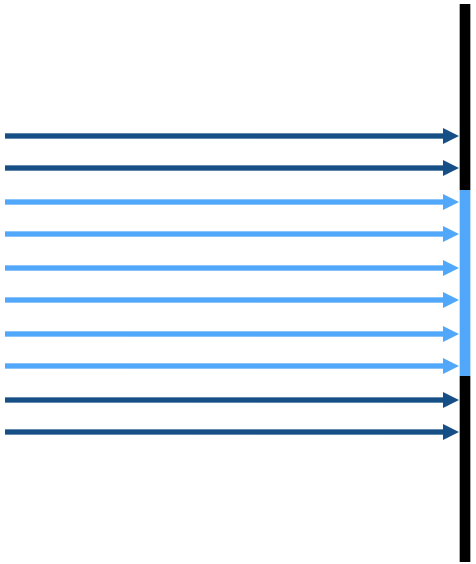
$$I \propto N \cdot L$$

Image intensity $\propto$ Surface normal $\cdot$ Light direction

Image intensity $\propto$ cos(angle between N and L)

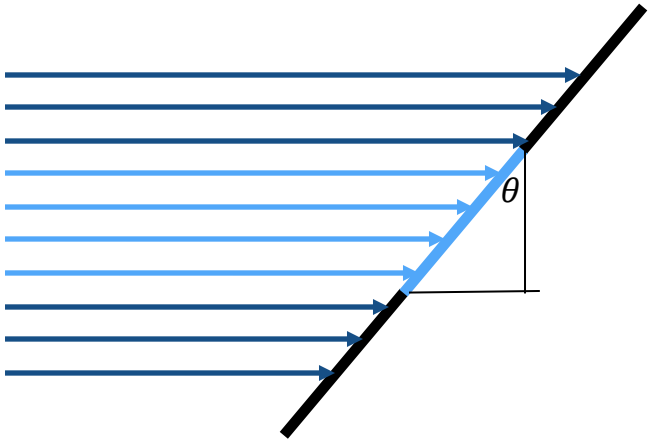
Lambertian Reflectance: Incoming

Reflected energy is proportional to cosine of angle between L and N.



Lambertian Reflectance: Incoming

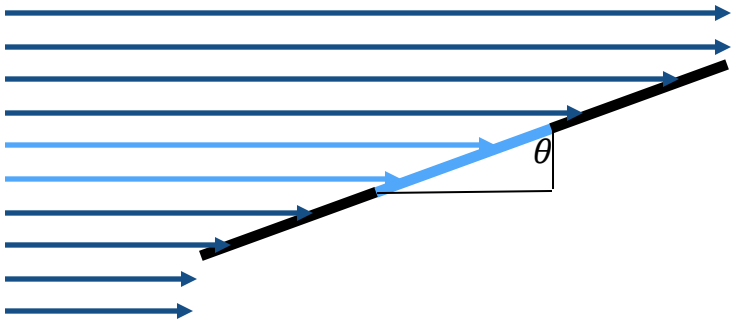
Reflected energy is proportional to cosine of angle between L and N.



$$I \propto \underbrace{N \cdot L}_{\cos(\theta)}$$

Lambertian Reflectance: Incoming

Reflected energy is proportional to cosine of angle between L and N.

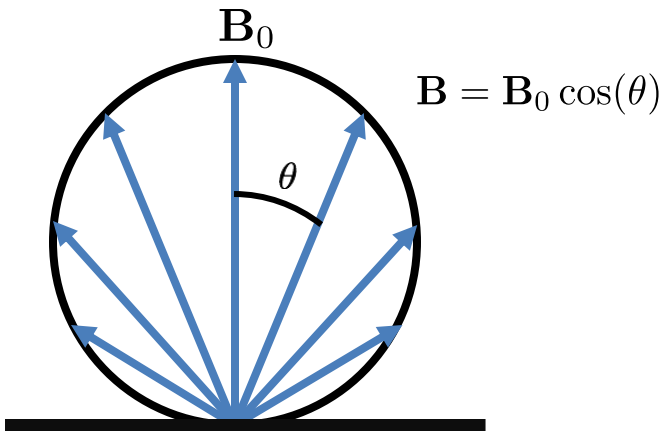


$$I \propto \underbrace{N \cdot L}_{\cos(\theta)}$$

Light hitting surface is proportional to the **cosine**.

Lambertian appearance is view-independent

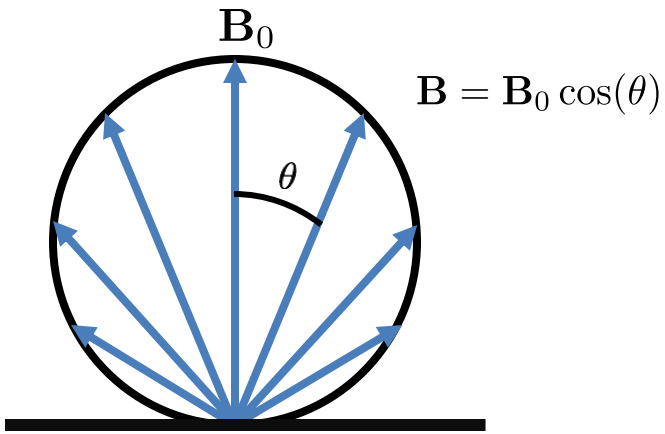
- Number of photons reflected to a given angle θ is proportional to $\cos(\theta)$



Lambert's cosine law: $\mathbf{B} = \mathbf{B}_0 \cos(\theta)$

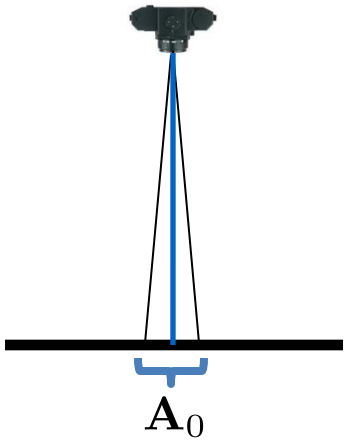
Lambertian appearance is view-independent

- Number of photons reflected to a given angle θ is proportional to $\cos(\theta)$



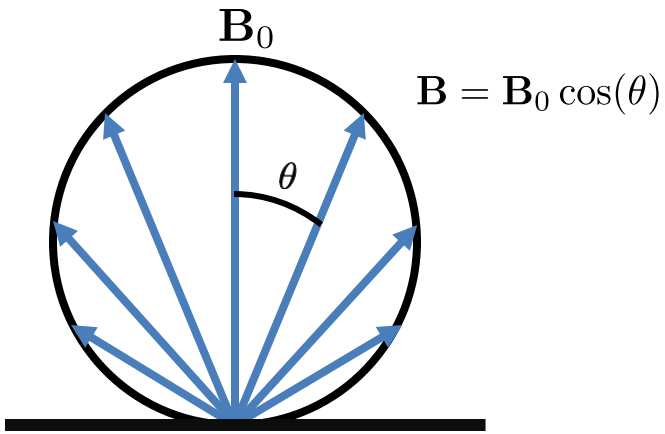
Lambert's cosine law: $\mathbf{B} = \mathbf{B}_0 \cos(\theta)$

- But appearance is the same from every angle due to larger pixel footprint at larger angles



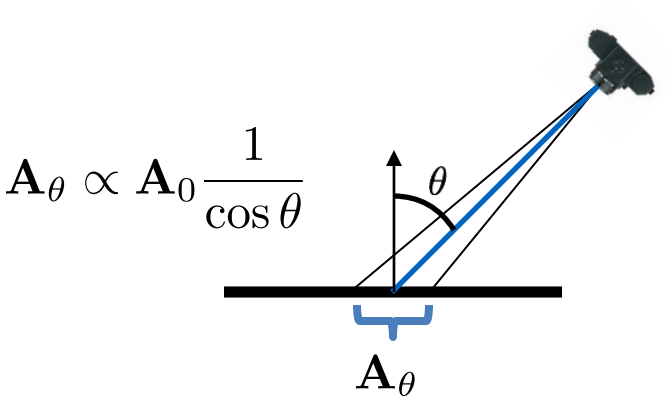
Lambertian appearance is view-independent

- Number of photons reflected to a given angle θ is proportional to $\cos(\theta)$



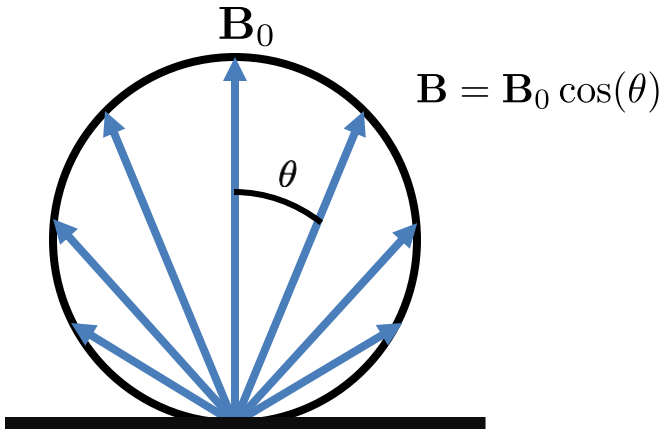
Lambert's cosine law: $\mathbf{B} = \mathbf{B}_0 \cos(\theta)$

- But appearance is the same from every angle due to larger pixel footprint at larger angles



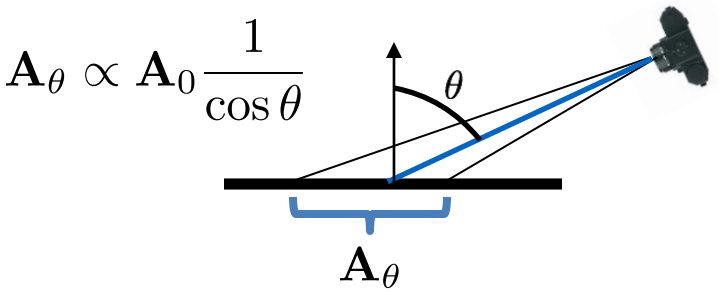
Lambertian appearance is view-independent

- Number of photons reflected to a given angle θ is proportional to $\cos(\theta)$



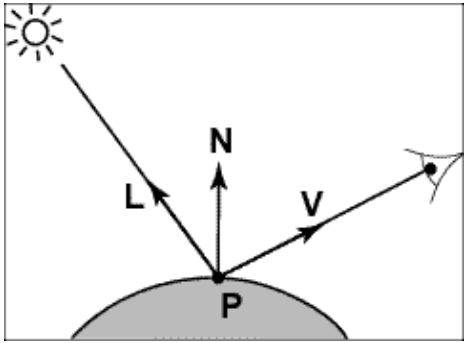
Lambert's cosine law: $\mathbf{B} = \mathbf{B}_0 \cos(\theta)$

- But appearance is the same from every angle due to larger pixel footprint at larger angles

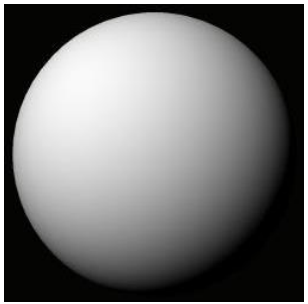


Radiance (what eye sees) $\propto \mathbf{B}_0 \mathbf{A}_0 \cancel{\cos(\theta)} \frac{1}{\cancel{\cos(\theta)}}$

Final Lambertian Image Formation Model

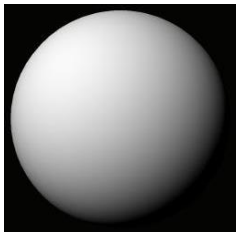


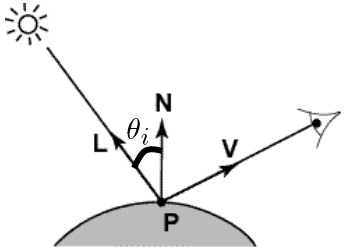
$$I = k_d \mathbf{N} \cdot \mathbf{L}$$



1. Diffuse **albedo**: what fraction of incoming light is reflected?
   - Introduce scale factor k_d
2. Light intensity: how much light is arriving?
   - Compensate with camera exposure (global scale factor)
3. Camera response function
   - Assume pixel value is linearly proportional to incoming energy (perform radiometric calibration if not)

A Single Image: Shape from Shading





Suppose (for now) $k_d = 1$

$$
\begin{aligned}
I &= k_d \mathbf{N} \cdot \mathbf{L} \\
&= \mathbf{N} \cdot \mathbf{L} \\
&= cos\ \theta_i
\end{aligned}
$$

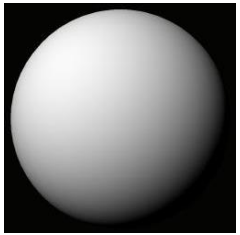
You can directly measure angle between normal and light source

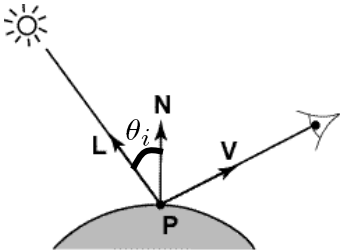
- Not quite enough information to compute surface shape
- But can be if you add some additional info, for example
  - assume a few of the normals are known (e.g., along silhouette)
  - constraints on neighboring normals—"integrability"
  - smoothness
- Hard to get it to work well in practice
  - plus, how many real objects have constant albedo?
  - But, deep learning can help

Shape from Shading

Shape from Shading assumptions:

- Diffuse material with spatially constant albedo
  - Reduces the number of material parameters to 1
- Known point light source at infinity
  - Keeps light direction L constant across all pixels
  - Makes L independent of geometry/depth
- Known camera at infinity (orthography)
  - Keeps view direction V constant across all pixels
  - Makes V independent of geometry/depth





[Slide credit: Andreas Geiger]

Lightsource Position Ambiguity

Plates and bowls or upside down or not?

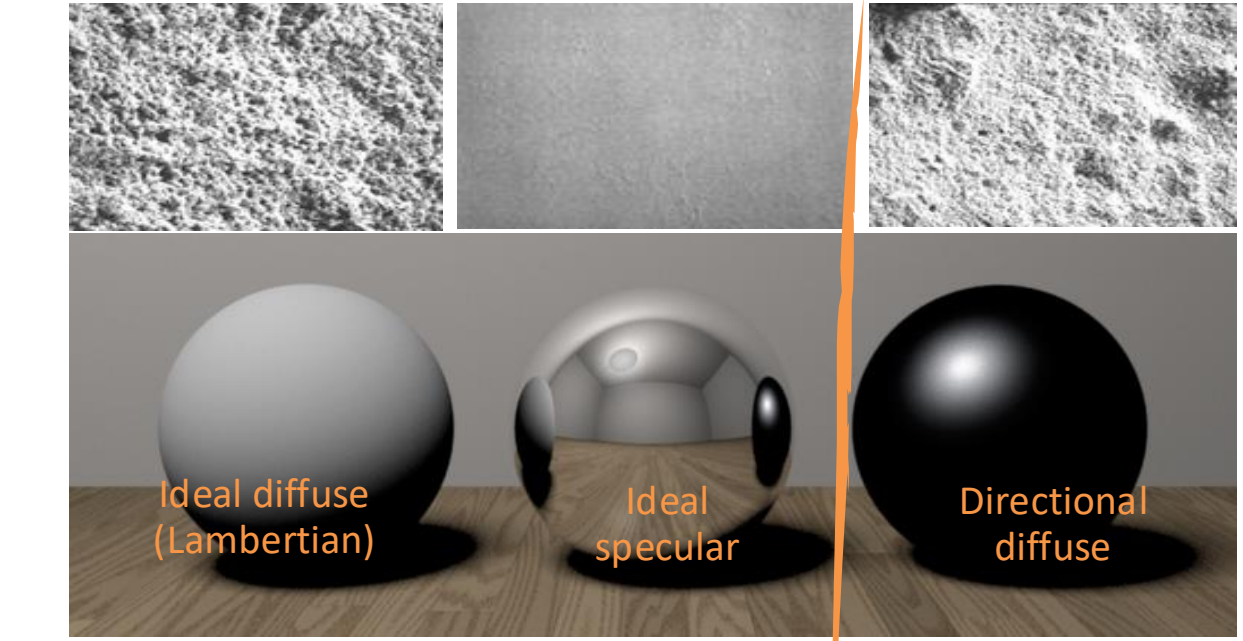
[https://www.good.is/optical-illusion-plates-and-bowls-upside-down-or-not]

Application: Detecting composite photos

Fake photo

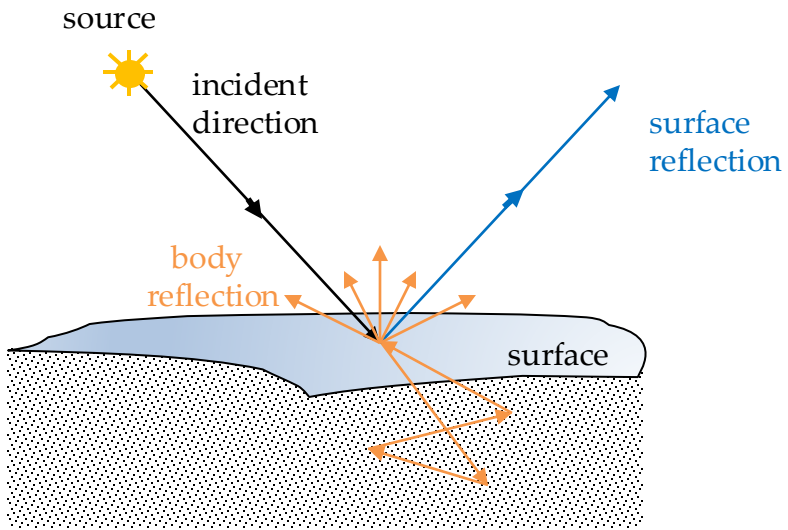
Real photo

Reflectance - Three Forms



[Slide credit: Kavita Bala]

Mechanisms of Surface Reflection



Body Reflection:

- Diffuse Reflection
- Matte Appearance
- Non-Homogeneous Medium
- Clay, paper, etc.

Surface Reflection:

- Specular Reflection
- Glossy Appearance
- Highlights
- Dominant for Metals

Image Intensity = Body Reflection + Surface Reflection

Mechanisms of Surface Reflection

Body Reflection:

- Diffuse Reflection
- Matte Appearance
- Non-Homogeneous Medium
- Clay, paper, etc.

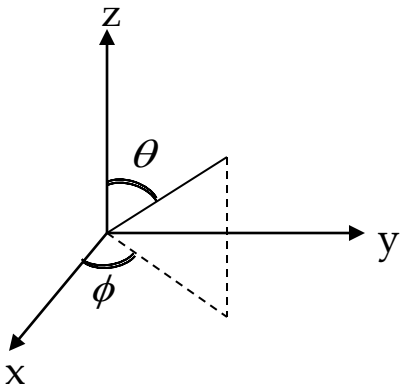
Surface Reflection:

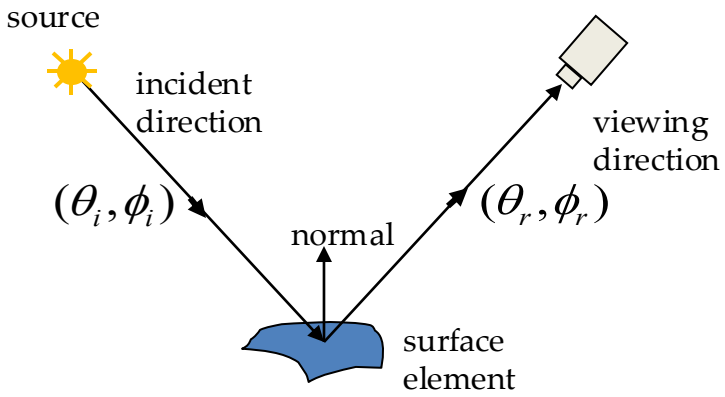
- Specular Reflection
- Glossy Appearance
- Highlights
- Dominant for Metals

Many materials exhibit both Reflections:

BRDF: Bidirectional Reflectance Distribution Function

θ : zenith
ϕ : azimuth



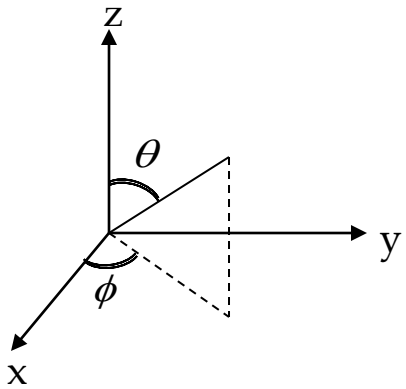


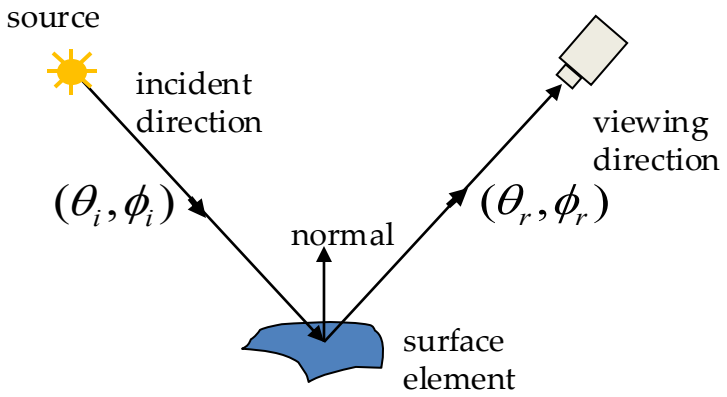
$E^{surface}(\theta_i, \phi_i)$ Irradiance due to source in direction (θ_i, ϕ_i)

$L^{surface}(\theta_r, \phi_r)$ Radiance of Surface in direction (θ_r, ϕ_r)

$$\text{BRDF}: \quad f(\theta_i, \phi_i; \theta_r, \phi_r) = \frac{L^{surface}(\theta_r, \phi_r)}{E^{surface}(\theta_i, \phi_i)}$$

BRDF Properties

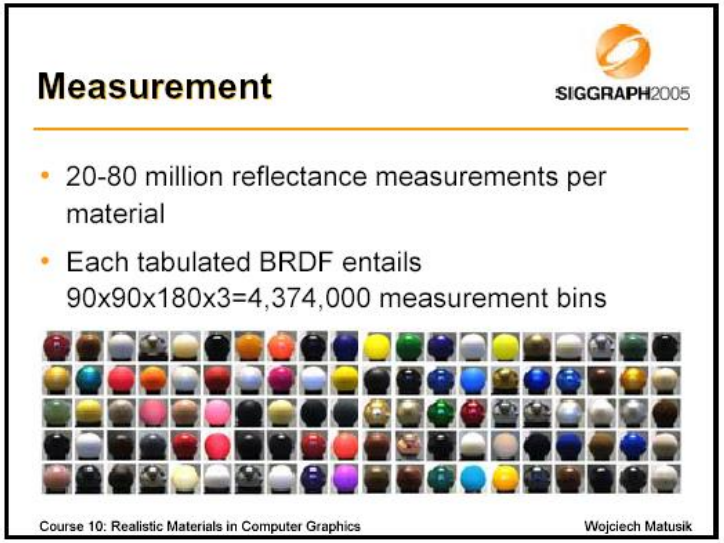




- BRDF is a 4D function
- Non-negative: $f(\theta_i, \phi_i, \theta_r, \phi_r) > 0$
- Helmholtz Reciprocity: $f(\theta_i, \phi_i, \theta_r, \phi_r) = f(\theta_r, \phi_r, \theta_i, \phi_i)$

BRDF databases

- MERL (Matusik et al.): 100 isotropic, 4 nonisotropic, dense



- CURET (Columbia-Utrect): 60 samples, more sparsely sampled, but also bidirectional texure functions (BTF)

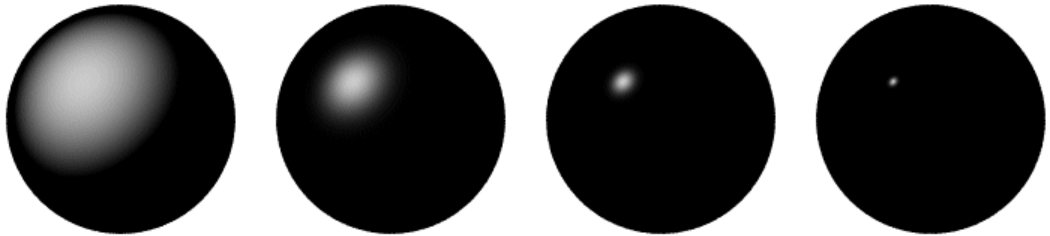
Sample albedos

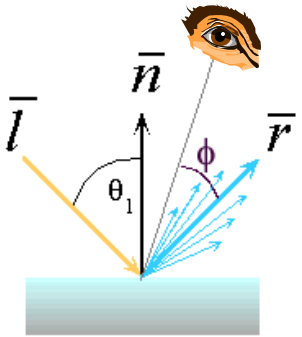
Surface	Typical albedo
Fresh asphalt	0.04[6]
Open ocean	0.06[7]
Worn asphalt	0.12[6]
Conifer forest, summer	0.08,[8] 0.09 to 0.15[9]
Deciduous forest	0.15 to 0.18[9]
Bare soil	0.17[10]
Green grass	0.25[10]
Desert sand	0.40[11]
New concrete	0.55[10]
Ocean ice	0.50 to 0.70[10]
Fresh snow	0.80[10]
Aluminium	0.85[12][13]

[Wikipedia: Albedo]

Phong Model

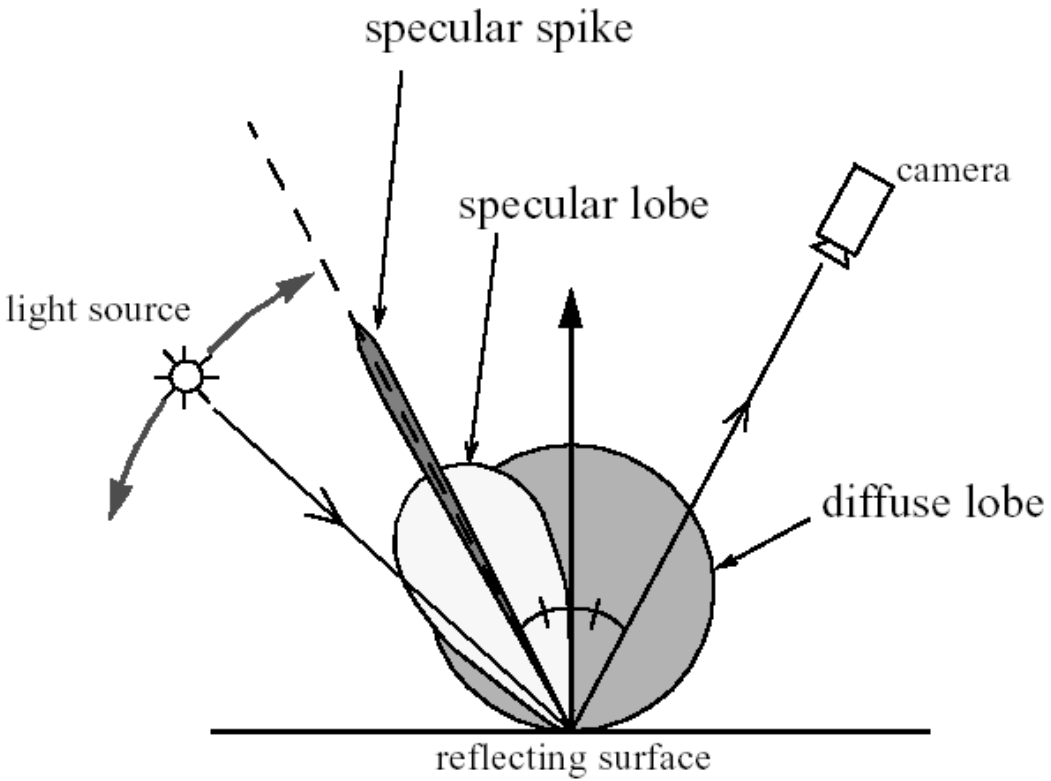
- These spheres illustrate the Phong model as *lighting direction* and n_{shiny} are varied:



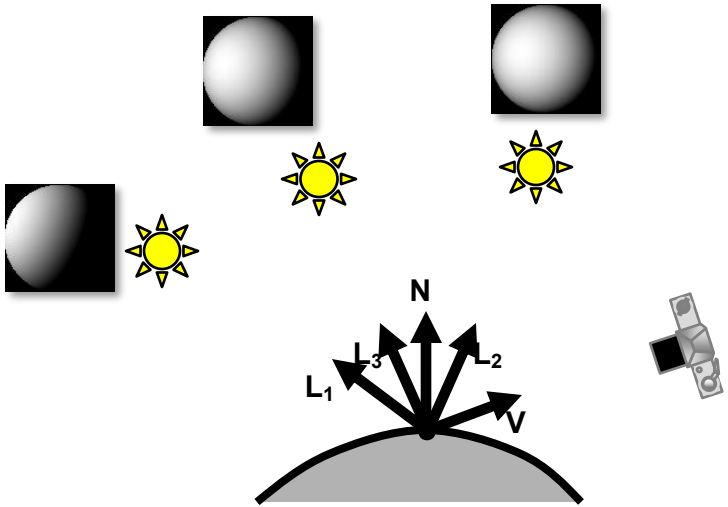


$$L = I\ \rho_s\ (\cos\phi)^{n_{shiny}}$$

All components of Surface Reflection



Multiple Images: Photometric Stereo



$$
\begin{aligned}
I_1 &= k_d \mathbf{N} \cdot \mathbf{L_1} \\
I_2 &= k_d \mathbf{N} \cdot \mathbf{L_2} \\
I_3 &= k_d \mathbf{N} \cdot \mathbf{L_3}
\end{aligned}
$$

Write this as a matrix equation:

$$
\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix} = k_d \mathbf{N}^T \begin{bmatrix} \mathbf{L_1} & \mathbf{L_2} & \mathbf{L_3} \end{bmatrix}
$$

Solving the Equations

$$\underbrace{\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix}}_{\mathbf{I}} = \underbrace{k_d \mathbf{N}^T}_{\mathbf{G}} \underbrace{\begin{bmatrix} \mathbf{L_1} & \mathbf{L_2} & \mathbf{L_3} \end{bmatrix}}_{\boldsymbol{L}}$$

$\mathbf{I}$: 1×3 $\quad$ $\mathbf{G}$: 1×3 $\quad$ $\boldsymbol{L}$: 3×3

Solving:

$$\mathbf{G} = \mathbf{I}\mathbf{L}^{-1}$$

Normalized the G, separate it into mode k_d and normal N

$$k_d = \|\mathbf{G}\|$$

$$\mathbf{N} = \frac{1}{k_d}\mathbf{G}$$

38

Solving the Equations

$$\underbrace{\begin{bmatrix} I_1 & I_2 & I_3 \end{bmatrix}}_{\mathbf{I}} = \underbrace{k_d \mathbf{N}^T}_{\mathbf{G}} \underbrace{\begin{bmatrix} \mathbf{L_1} & \mathbf{L_2} & \mathbf{L_3} \end{bmatrix}}_{\mathcal{L}}$$

$\mathbf{I}$: 1×3 $\quad$ $\mathbf{G}$: 1×3 $\quad$ $\mathcal{L}$: 3×3

$$\mathbf{G} = \mathbf{I}\mathbf{L}^{-1}$$

- When is L nonsingular (invertible)?
  - >= 3 light directions are linearly independent, or:
  - All light direction vectors cannot lie in a plane.
- What if we have more than one pixel?
  - Stack them all into one big system.

39

More than Three Lights

$$\begin{bmatrix} I_1 & \dots & I_n \end{bmatrix} = k_d \mathbf{N}^T \begin{bmatrix} \mathbf{L_1} & \dots & \mathbf{L_n} \end{bmatrix}$$

- Solve using least squares (normal equations):

$$\mathbf{I} = \mathbf{GL}$$

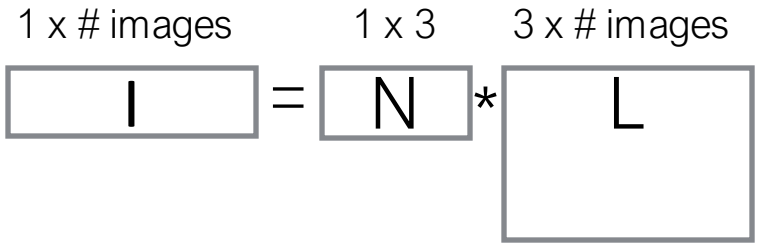
Find $\boldsymbol{G}$ s.t. minimizes $\|\boldsymbol{I} - \boldsymbol{GL}\|^2$

$$\mathbf{IL}^{\mathbf{T}} = \mathbf{GLL}^{\mathbf{T}}$$
$$\mathbf{G} = (\mathbf{IL}^{\mathbf{T}})(\mathbf{LL}^{\mathbf{T}})^{-1}$$

- Given G, solve for N and k_d as before.

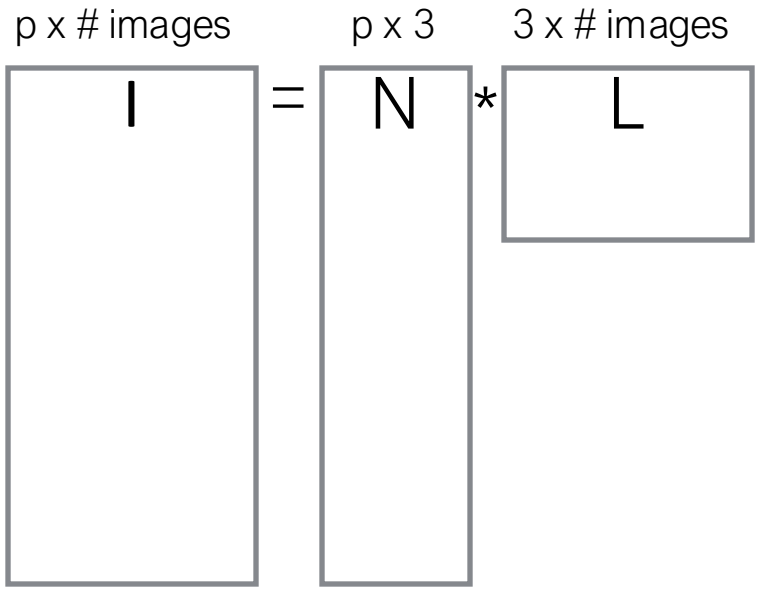
More than one pixel

Previously:



More than one pixel

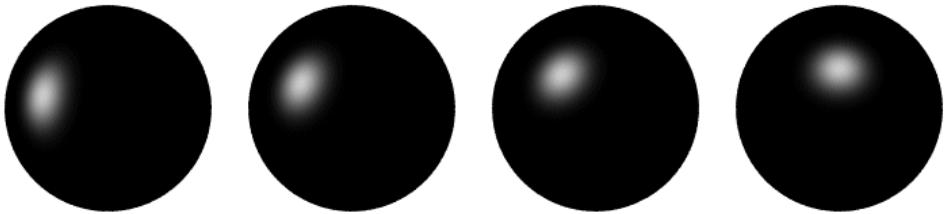
Stack all pixels into one system:



Solve as before.

Computing Light Source Directions

- Trick: Place a mirror ball in the scene.

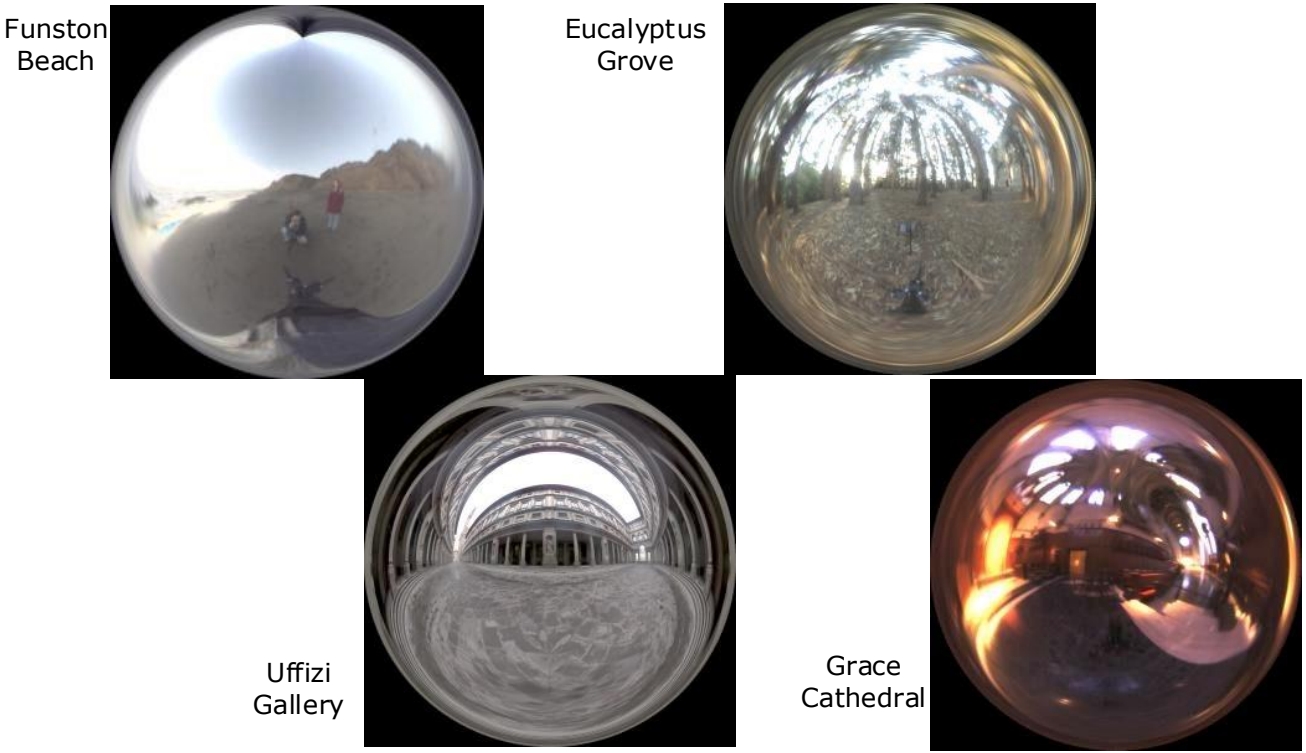


- The location of the highlight is determined by the light source direction.

HDRI Sky Probe

[Slide credit: Kavita Bala]

Real-World HDR Lighting Environments

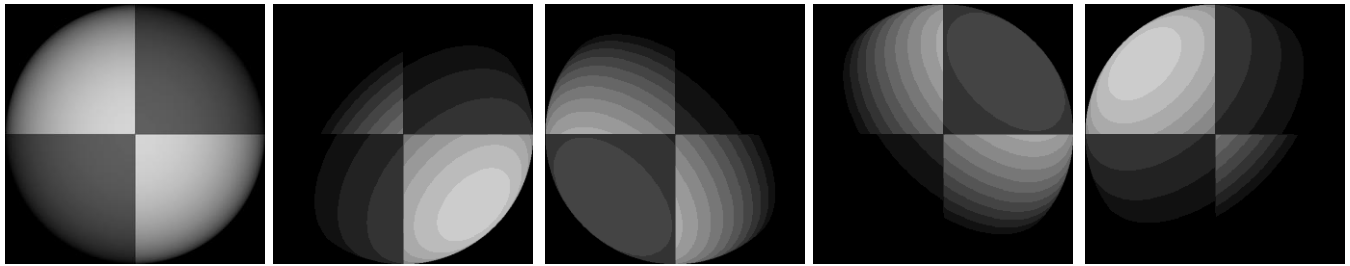


Lighting Environments from the Light Probe Image Gallery:
http://www.debevec.org/Probes/

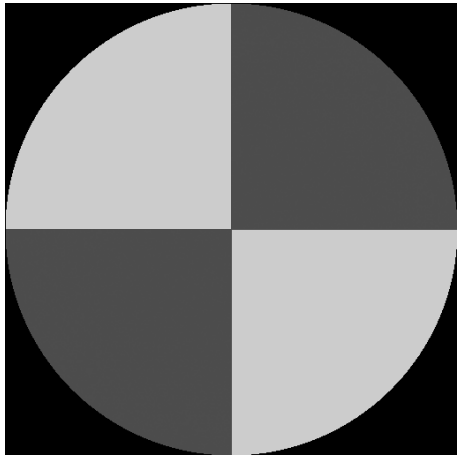
[Slide credit: Kavita Bala]

Example

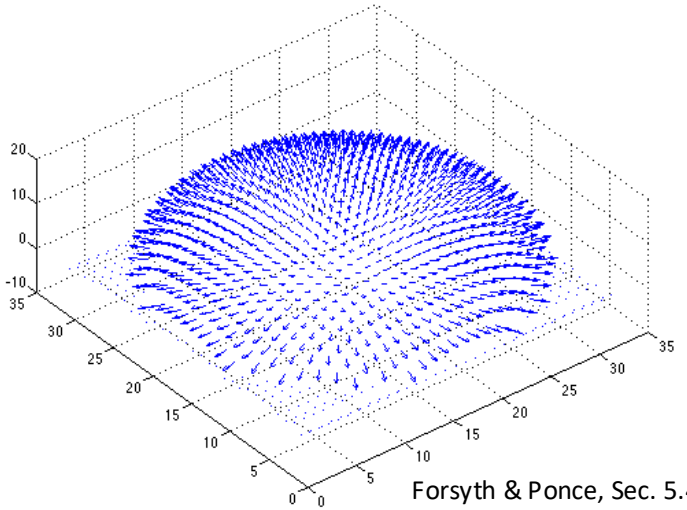
Input views



Recovered albedo



Recovered normal field



Forsyth & Ponce, Sec. 5.4

Depth from normals

- Solving the linear system per-pixel gives us an estimated surface normal for each pixel
- How can we compute depth from normals?
  - Normals are like the “derivative” of the true depth

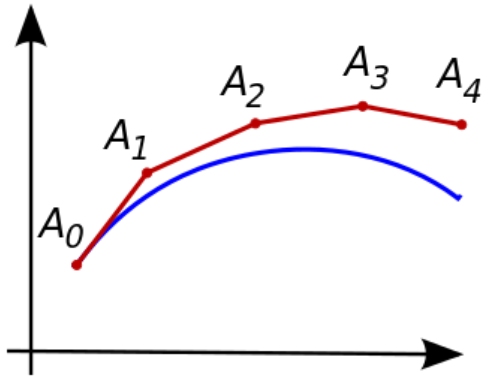
Input photo

Estimated normals

Estimated normals (needle diagram)

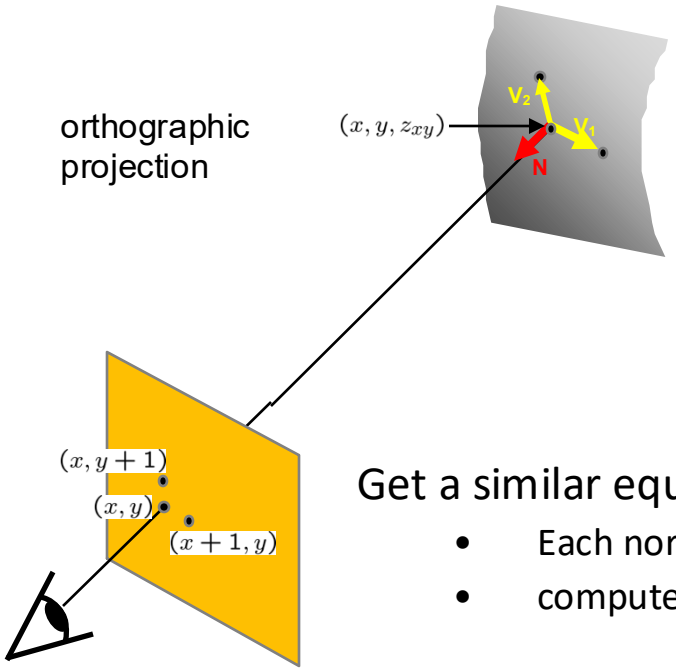
Normal Integration

- Integrating a set of derivatives is easy in 1D
  - (similar to Euler's method from diff. eq. class)
- Could integrate normals in each column / row separately
  - Wouldn't give a good surface
- Instead, we formulate as a linear system and solve for depths that *best agree with the surface normals*



Depth Map from Normal Map

- Given the surface normals we can iteratively compute depth?



Assume a smooth surface

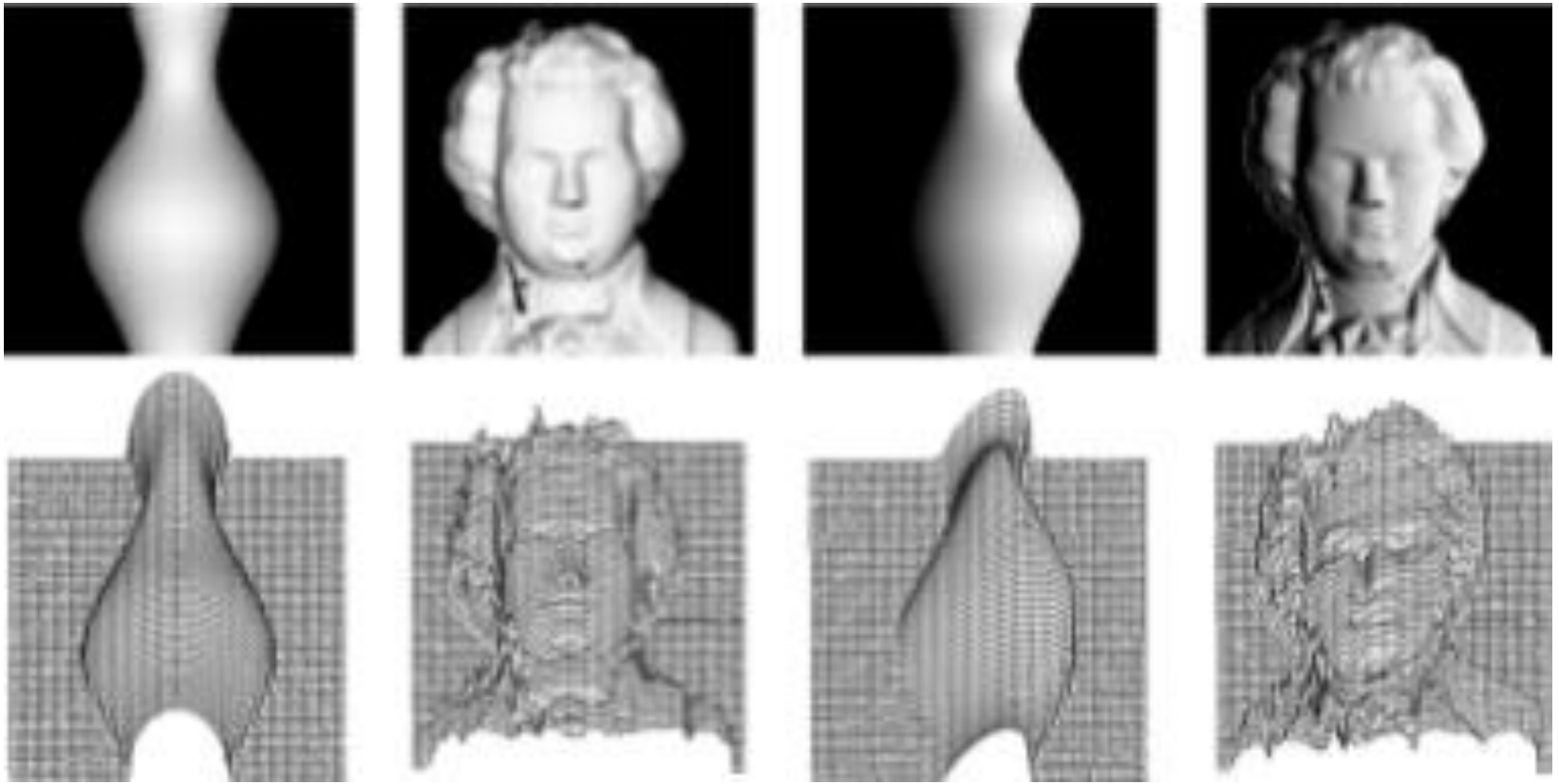
$$\begin{aligned} V_1 &= (x+1, y, z_{x+1,y}) - (x, y, z_{xy}) \\ &= (1, 0, z_{x+1,y} - z_{xy}) \end{aligned}$$

$$\begin{aligned} 0 &= N \cdot V_1 \\ &= (n_x, n_y, n_z) \cdot (1, 0, z_{x+1,y} - z_{xy}) \\ &= n_x + n_z(z_{x+1,y} - z_{xy}) \end{aligned}$$

Get a similar equation for $\mathbf{V_2}$

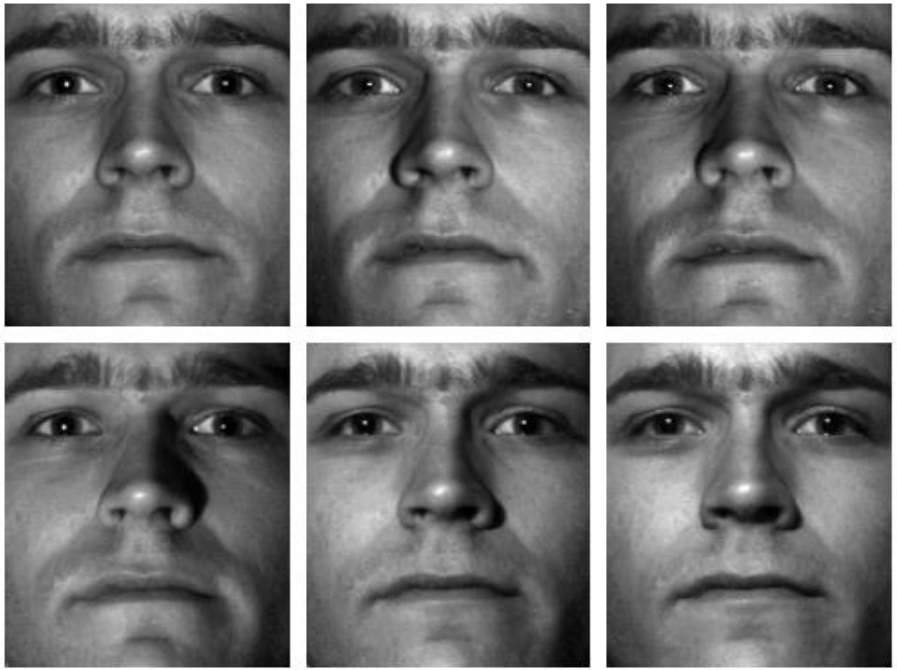
- Each normal gives us two linear constraints on z
- compute z values by solving a matrix equation

Results



[Slide credit: Andreas Geiger] [Frankot and Chellappa: A Method for enforcing integrability in shape from shading algorithms. TPAMI, 1988]

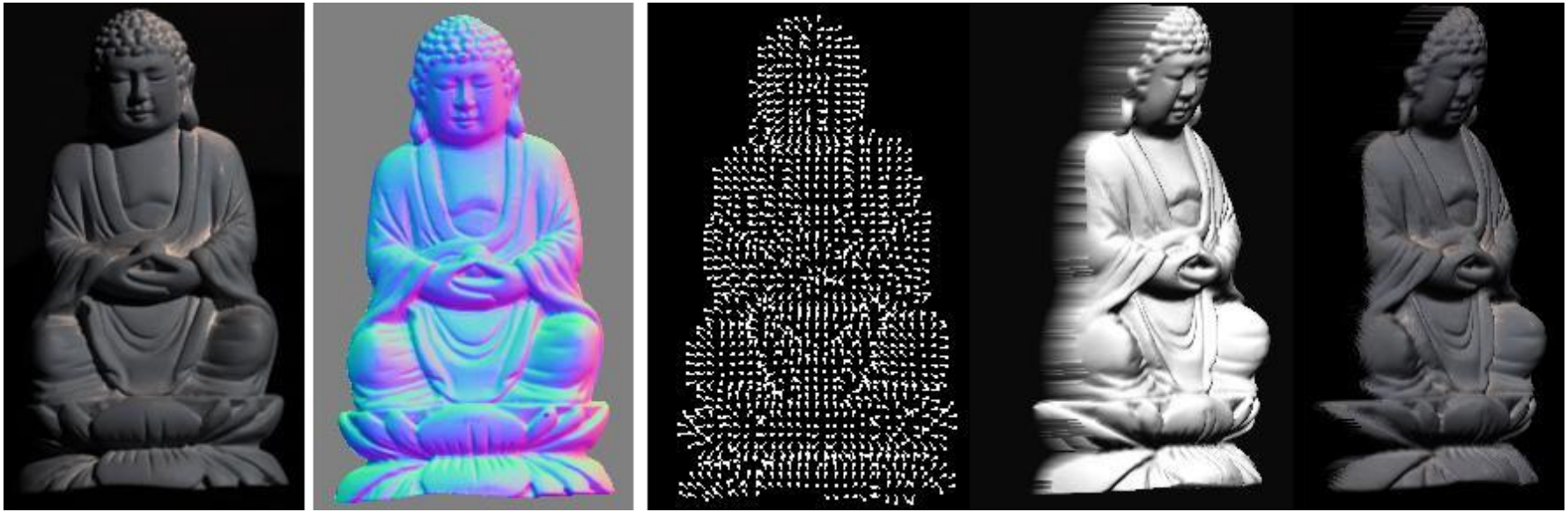
Results



[from Athos Georghiades]

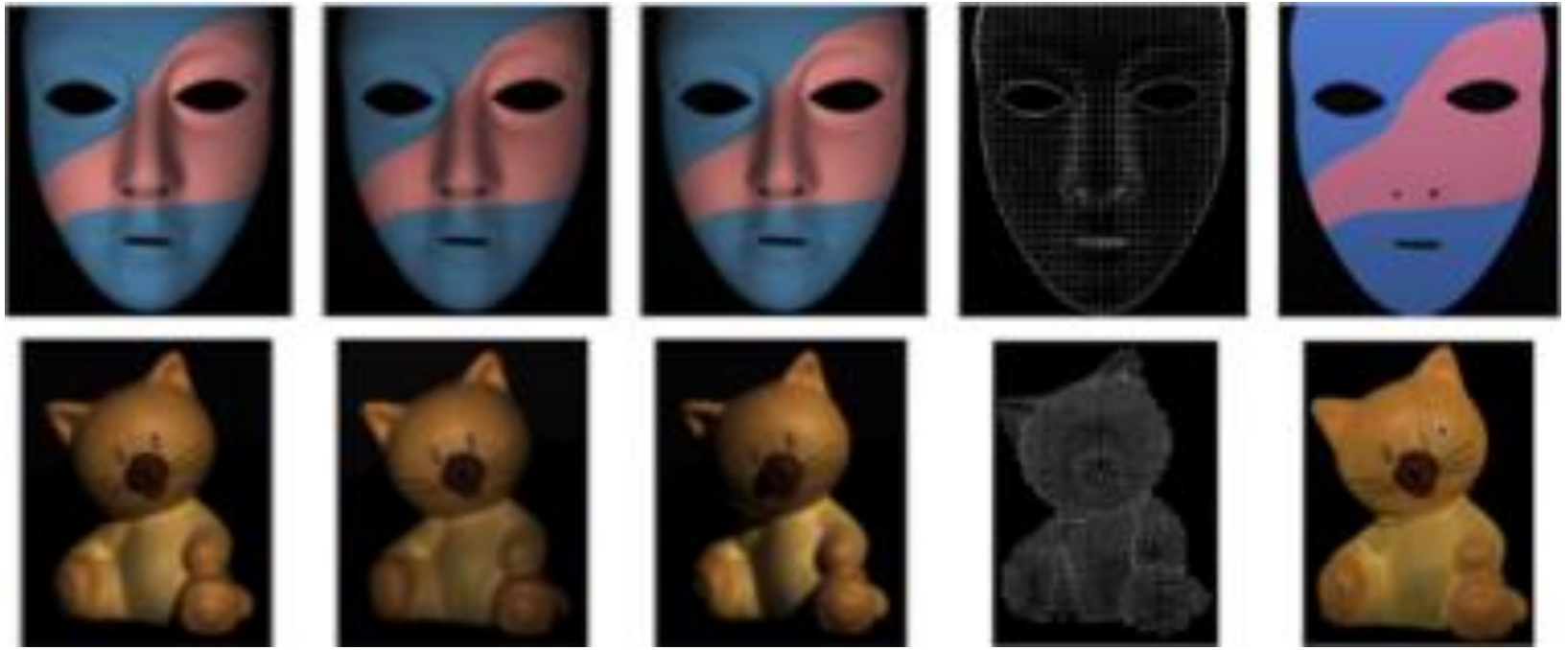
52

Results



[Woodham: Analysing images of curved surfaces. Artificial Intelligence, 1981]

Results

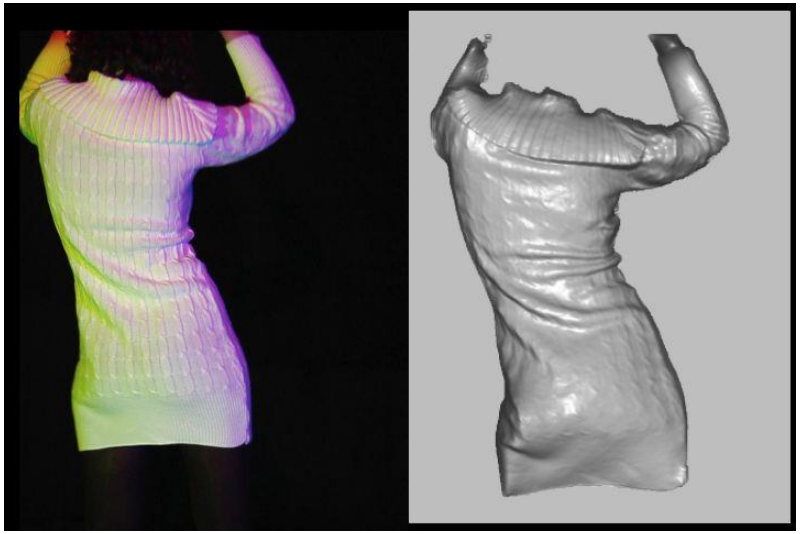


- For color images, apply PS to each channel separately to obtain color albedo.
- Deviations from Lambertian assumption and global illumination cause errors.

[slide credit: Andreas Geiger]

Extension

- Photometric Stereo from Colored Lighting



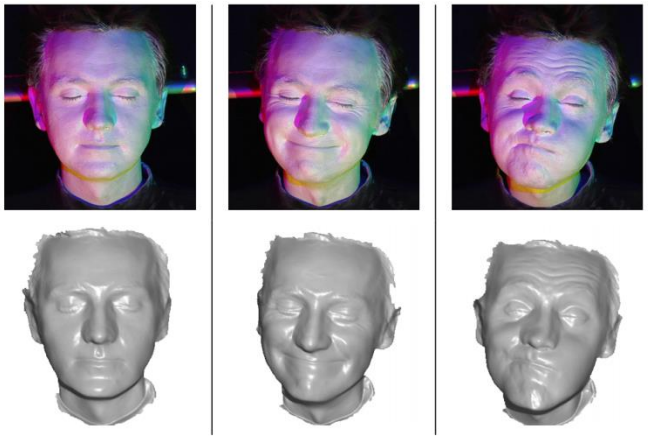
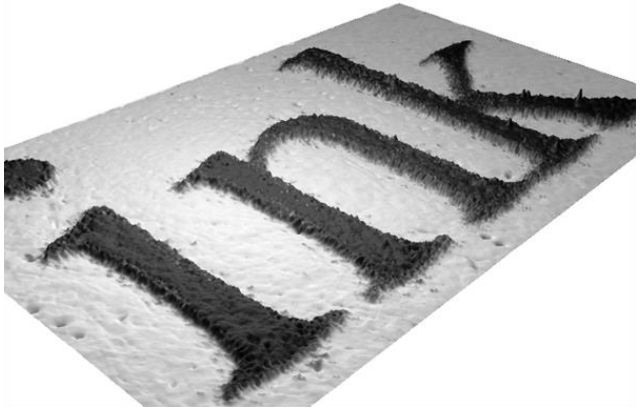


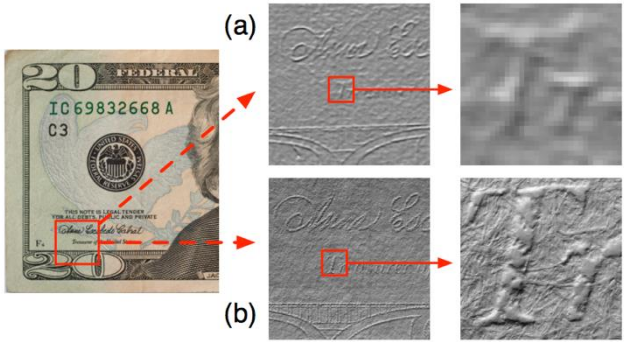
Fig. 2. **Applying the original algorithm to a face with white makeup.** Top: example input frames from video of an actor smiling and grimacing. Bottom: the resulting integrated surfaces.

Video Normals from Colored Lights
Gabriel J. Brostow, Carlos Hernández, George Vogiatzis, Björn Stenger, Roberto Cipolla
IEEE TPAMI, 2011.

Specialized High-Resolution Scanners

- Recovers extremely subtle details with high accuracy
- Photometric stereo yields unmatched performance



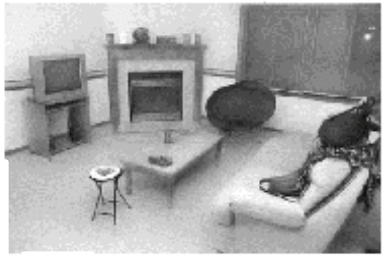


[Sensing Surfaces with GelSight, 2011]

Intrinsic Image Decomposition

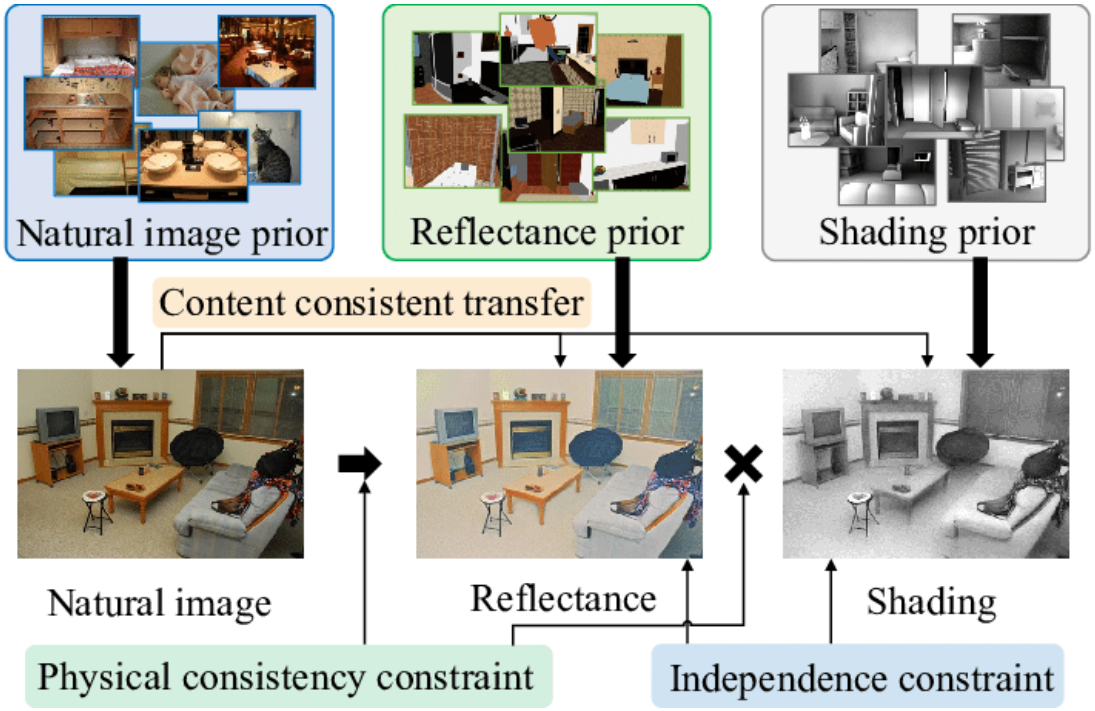
Natural image

Reflectance



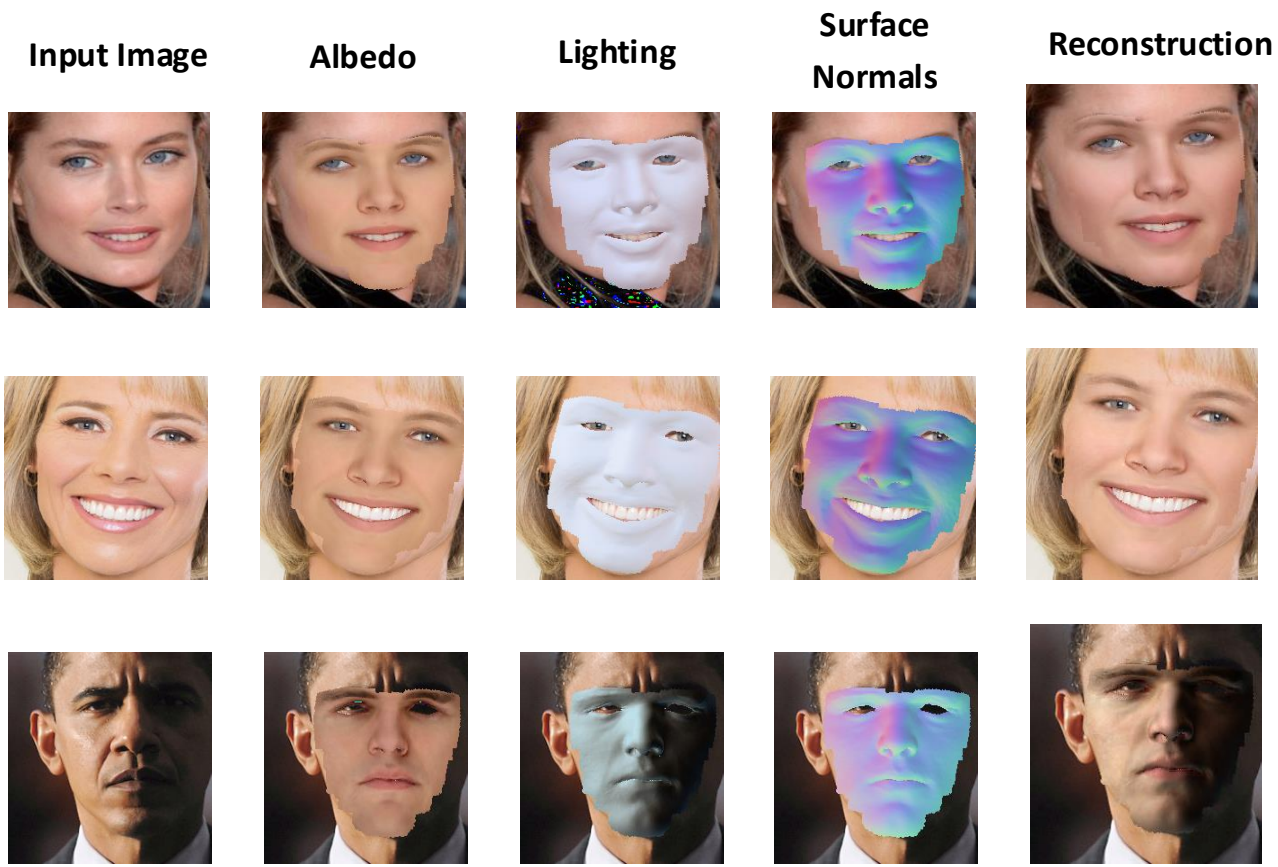
Shading

Intrinsic Image Decomposition



Unsupervised Learning for Intrinsic Image Decomposition from a Single Image
Yunfei Liu, Yu Li, Shaodi You and Feng Lu, CVPR2020

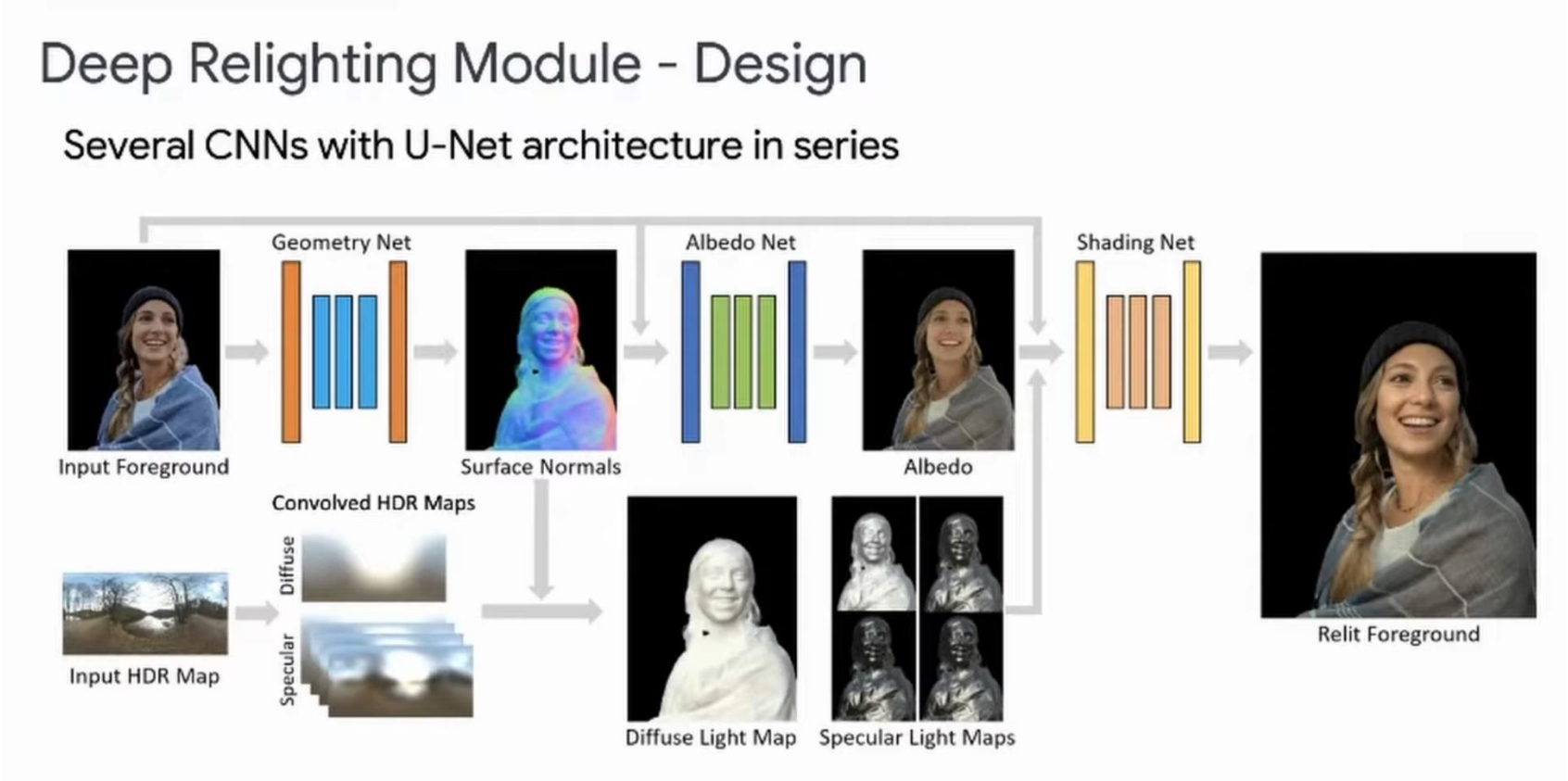
Diffuse Reflection



Total Relighting

[Pandey et al, SIGGRAPH'21, YouTube]

Total Relighting



[Pandey et al, SIGGRAPH'21, YouTube]

Image processing operations

- Point-wise operations
  - histogram operation
- Neighborhood operations
  - Spatial domain (image convolution)
  - Frequency domain (Fourier transform)
  - Binary image analysis
- Geometric operations
  - Rotation, translation, affine…

62

Histograms

How do we ‘summarize’ images?

← https://rvdboomgaard.github.io/ComputerVision_LectureNotes/LectureNotes/IP/Images/ImageHistograms.htm

https://rvdboomgaard.github.io/ComputerVision_LectureNotes/LectureNotes/IP/PointOperators/index.html →

Image Histograms

1D / Univariate Histogram

Summarize image f via ***Histogram*** of its values
(over all possible values)

$$h_f[i] = \sum_{\substack{x \in E \\ \text{for } i = 0, \dots, k-1}} [e_i \leq f(x) < e_{i+1}]$$

Iverson Bracket

$$[c] = \begin{cases} 1 & \text{for } c = \text{true} \\ 0 & \text{for } c = \text{false} \end{cases}$$

Is pixel value inside ***bin***?

$$f(x) \in [e_i, e_{i+1})$$

Bin edges

e_i for $i = 0, \dots, k$

NumPy: `histogram()`

https://numpy.org/doc/stable/reference/generated/numpy.histogram.html

How choose ***bin size/number***?

- No clear answer
- Empirically → More art than science
- Sturges' rule (assumes *Gaussian* data):

$$k = \lceil \log_2 n \rceil + 1$$

bins number

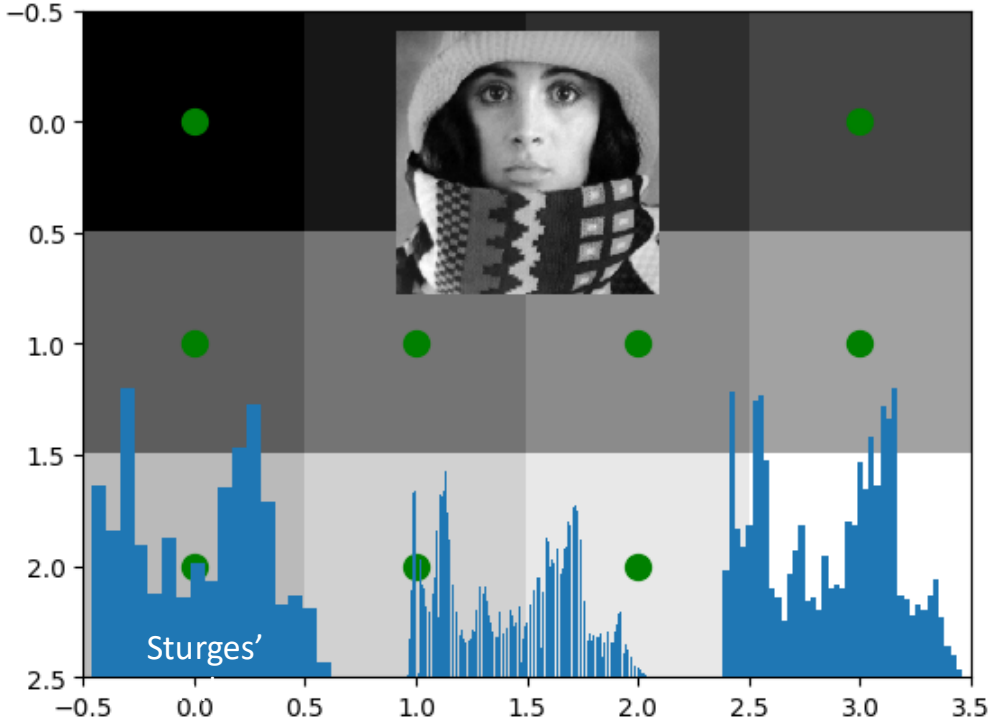
pixels number

`histogram_bin_edges()`

https://numpy.org/doc/stable/reference/generated/numpy.histogram_bin_edges.html

Image Histograms

1D / Univariate Histogram



Too many (and too small) bins?

→ Many bins will be empty

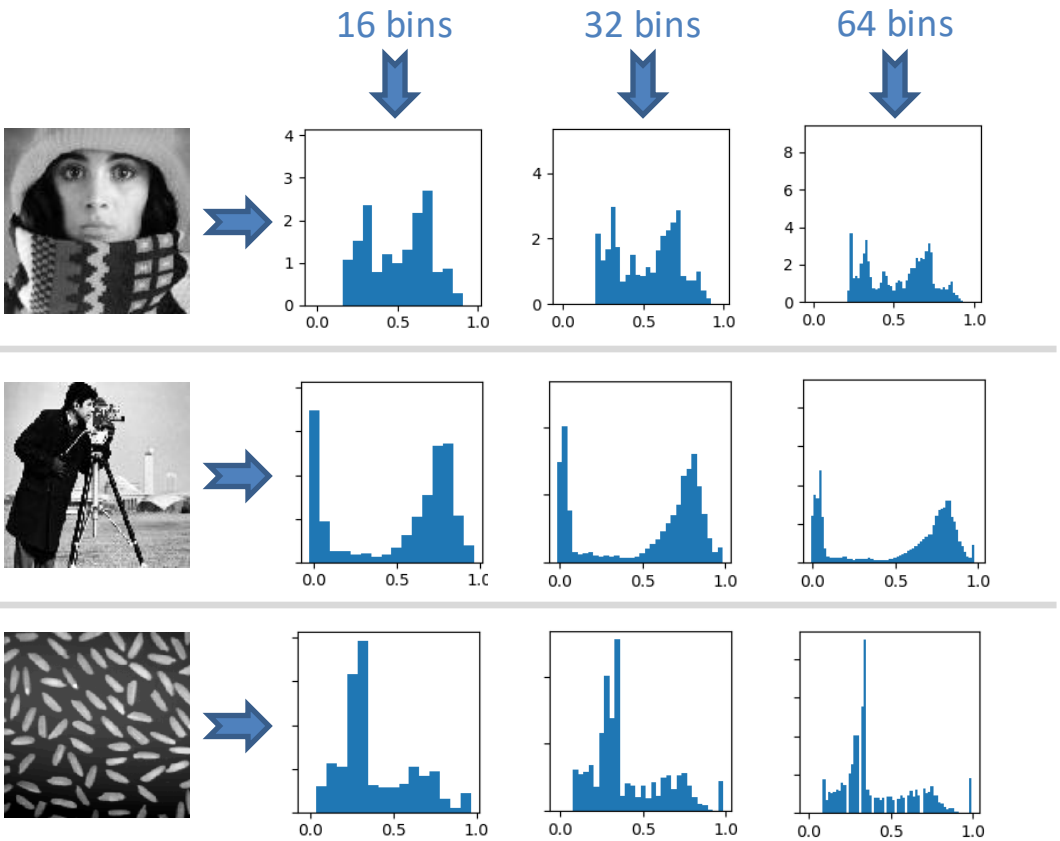
→'High-frequency' info

Too few (and too big) bins?

→ Will not encode 'well enough'

the distribution of values

Image Histograms



Too many (and too small) bins?

→ Many bins will be empty

→'High-frequency' info

Too few (and too big) bins?

→ Will not encode 'well enough'

the distribution of values

Image Histograms

Histogram → $h_f(u)$ counts pixels with value $f(x) = u$ — ***PDF*** – Probability-density function, normalized such that: $\sum h_f(u) = 1$

Cumulative Histogram → $H_f(u)$ counts pixels with value $f(x) \leq u$ — ***Cumulative distribution*** function

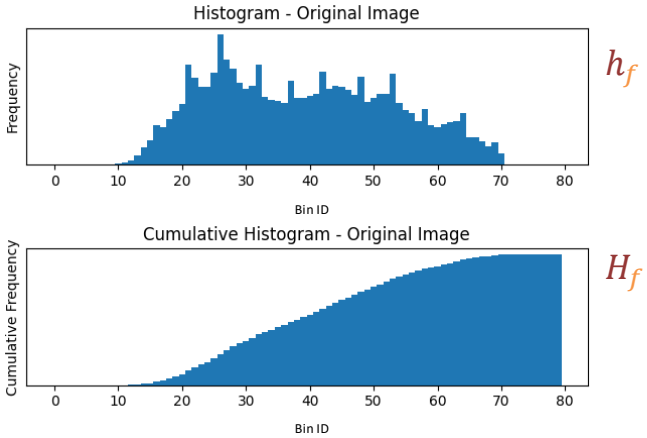
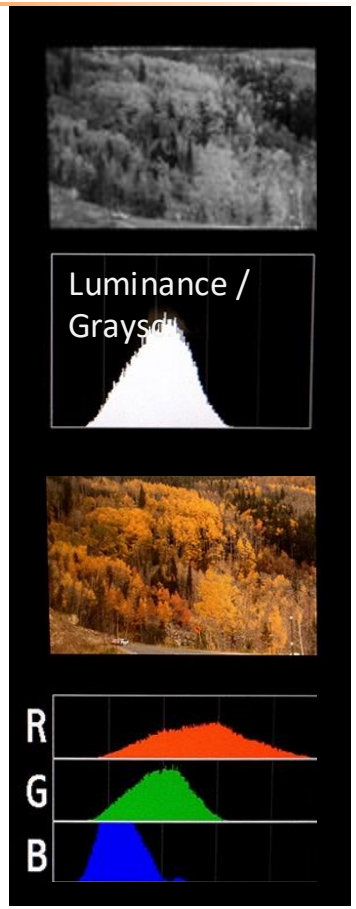


Image Histograms

3D / Multivariate Histogram

Grayscale images → ***1D Histogram***

(each pixel → Scalar value) (Univariate)



What about ***Color*** images?

(each pixel → Triplet of BGR value)

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \\ f_3(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} b(\boldsymbol{x}) \\ g(\boldsymbol{x}) \\ r(\boldsymbol{x}) \end{bmatrix}$$

Ignores color **Correlations**!

Trivial solution: ***3x separate*** 1D histograms per channel

Image Histograms

3D / Multivariate Histogram

Better solution → ***3D Histogram***

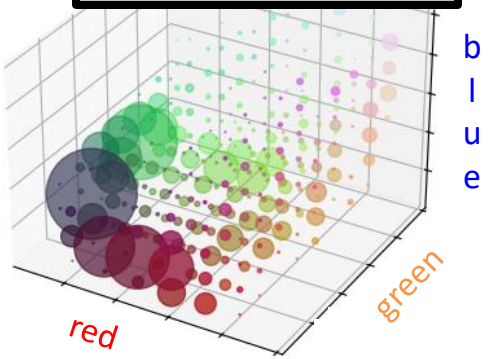
$$h_f[i,j,k] = \sum_{x \in E} [e_{1,i} \le f_1(x) < e_{1,i+1}] \; [e_{2,j} \le f_2(x) < e_{2,j+1}] \; [e_{3,k} \le f_3(x) < e_{3,k+1}]$$

Is pixel value inside

3D bin

with indices (i, j, k) ?

3D bins: 8x8x8 cubes



Bin centered @ RGB point

Sphere size == Hist. count

`histogramdd()`

https://numpy.org/doc/stable/reference/generated/numpy.histogramdd.html

Image Histograms

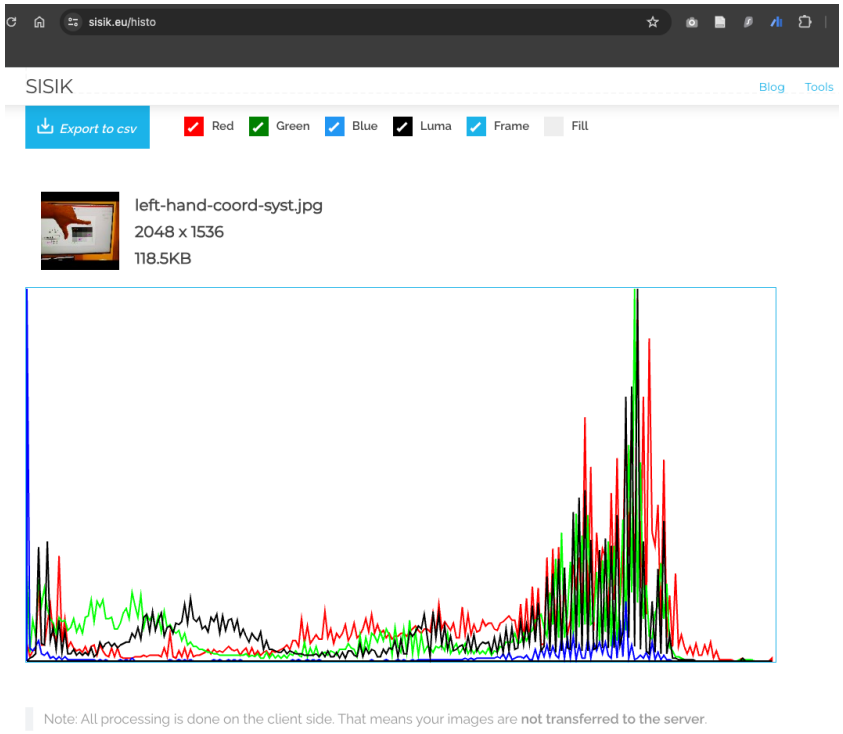
Online Demo - 1D Histograms

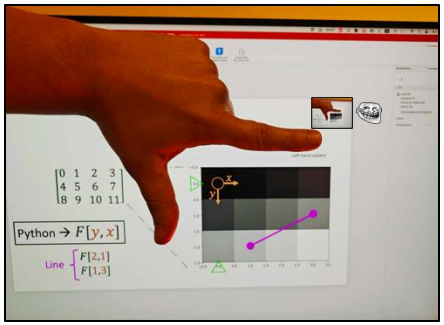
*'This tool basically creates **256 bins** for each color (red, green, blue) and greyscale (luma) intensity. The number of bins is shown on the **horizontal** axis.'*

*'The tool then loops through every image pixel and counts the **occurrence of each intensity**. The counts of occurrences in each bin are then displayed on **vertical** axis.'*

*'Counts for each pixel-intensity are **normalized** to range 0 to 255 before they are displayed on the graph.'*

'Used JavaScript in combination with WebAssmebly to create this tool.'





https://sisik.eu/histo

Point Operators

Histograms

f: $\mathcal{D} \rightarrow ([0,1] \in \mathbb{R})$

φ: $([0,1] \in \mathbb{R}) \rightarrow ([0,1] \in \mathbb{R})$

$$(\varphi f)(x) = \varphi\big(f(x)\big)$$

Monadic
'in-place'
Point Operator

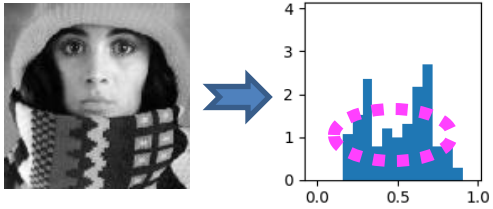
Replaces pixel value $f(x)$ ***independent*** of:

- Position x
- All other pixel values in neighborhood

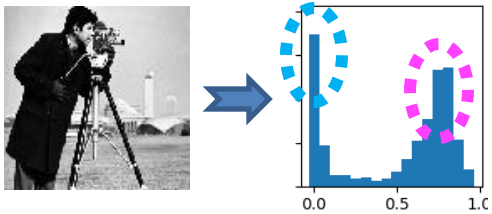
How can we 'manipulate' histograms with Monadic Point Operators?

Point Operators

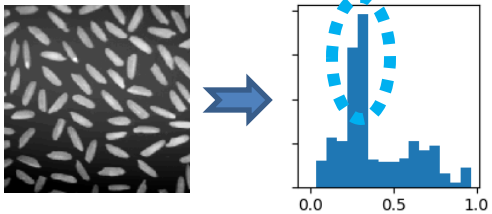
Histograms (issues)



Only *mid-range* gray
No extremes



Spikes for *bright* and *very-dark* areas



Mostly *dark-ish* areas

Not using full range
of luminance
might be sub-optimal

Image might be … :

Missing *large (bright)* values? → … too *dark*

Missing *low (dark)* values? → … too *bright*

Changing illumination conditions
for same scene → very different histograms!

Point Operators

Histogram Contrast Stretching

$f(x)$: $\mathbb{R}^2 \rightarrow [a, b] \subset [0,1] \in \mathbb{R}$

Contrast Stretching { Construct new image using full range of luminance

Monadic Operator

$$g(x) = \varphi(f(x)) = \frac{f(x) - a}{b - a}$$

'Distance' from min luminance in f

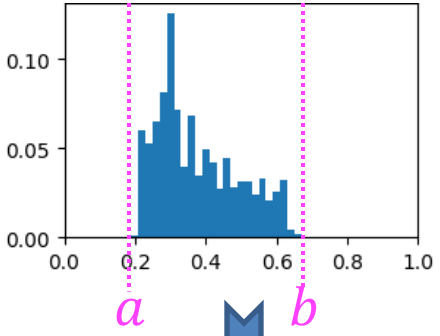
Stretches luminance range to [0,1]

$$g = \frac{f - f_{min}}{f_{max} - f_{min}}$$

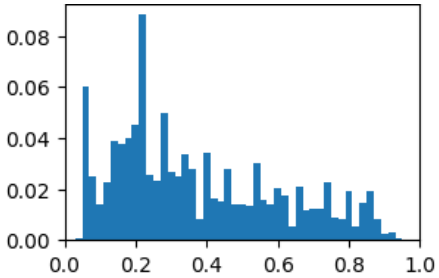
Lifted arithmetical operators

X axis: luminance values (bins)
Y axis: frequency for each value

Input: **Low Contrast**

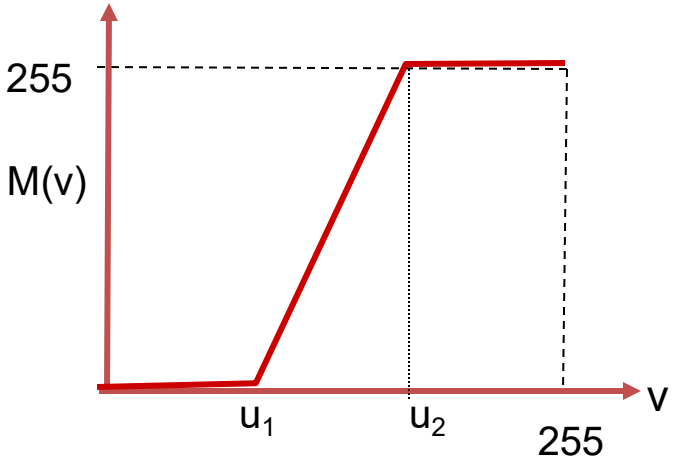


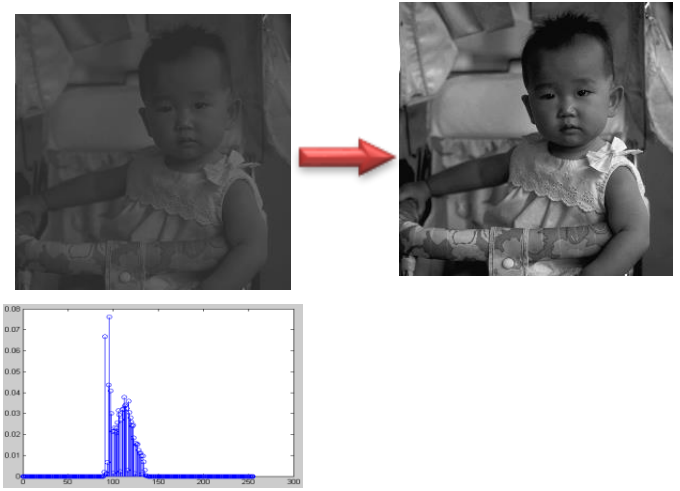
Output: **Contrast Stretched**



Contrast Enhancement

- If most of the gray-levels in the image are in $[u_1\ u_2]$, the following mapping increases the image contrast.
- The values u_1 and u_2 can be found by using the image's accumulated histogram.





74

Gamma Correction

Non-linear scaling (power-law transformation):

Output gray level $s(r) = cr^{\gamma}$

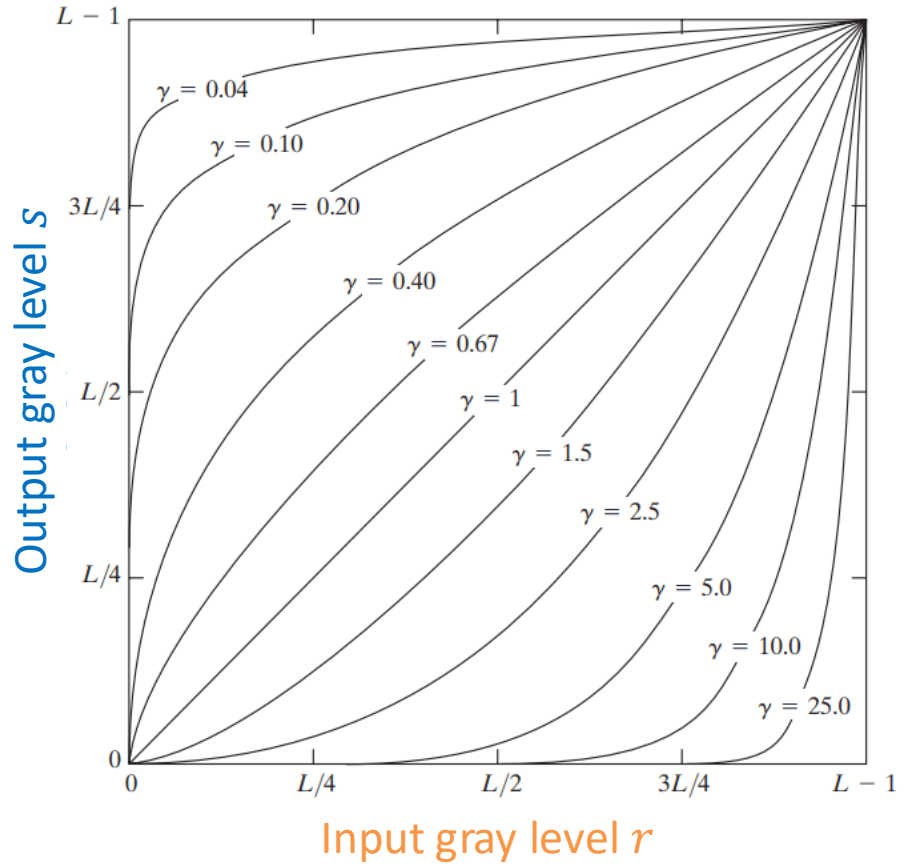
Normalization constant

Input gray level

- Note: $s(r)$ is linear for $\gamma = 1$.

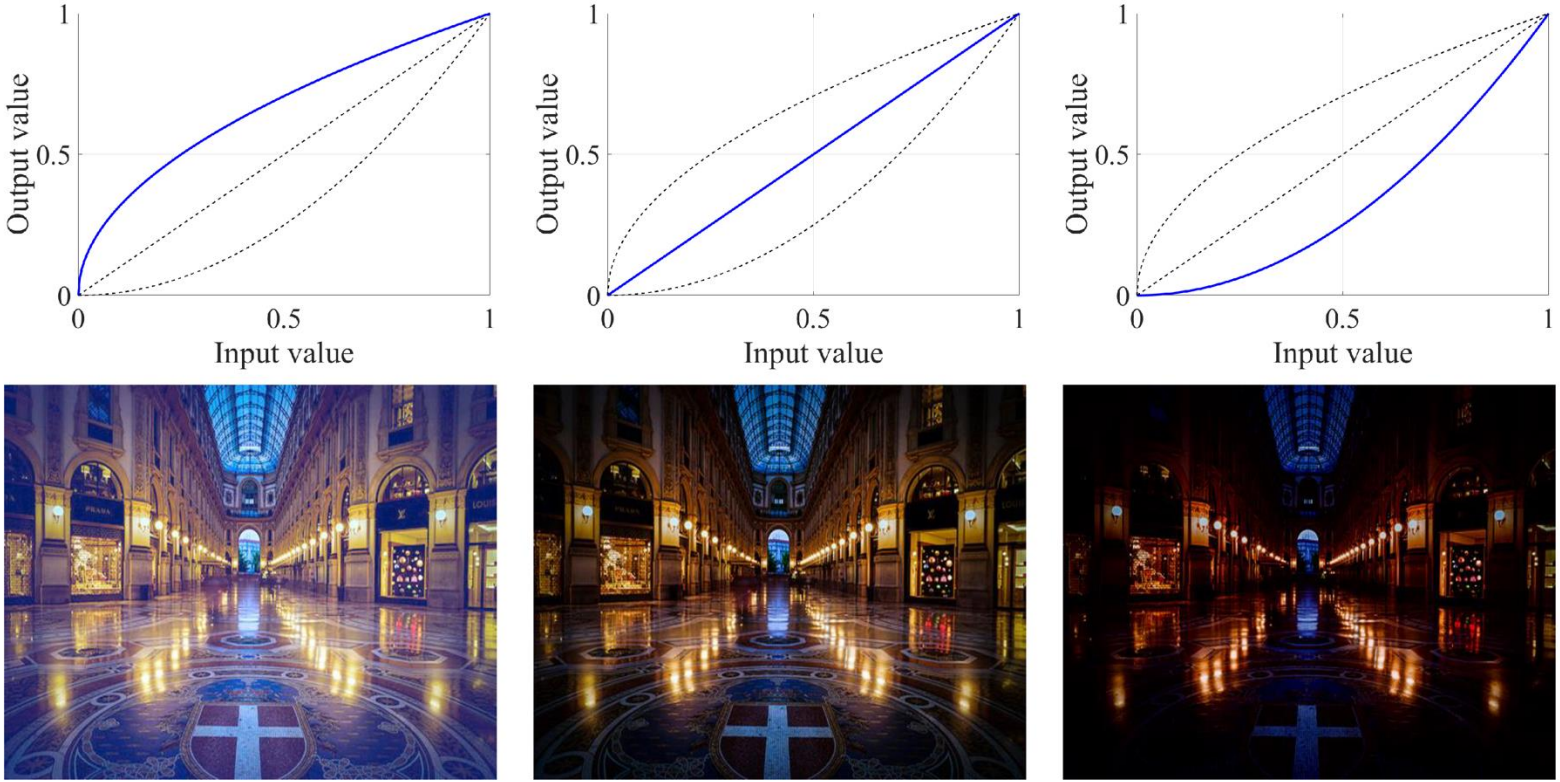
gamma=0.5 gamma=1.0 gamma=1.8 gamma=2.0 gamma=2.5

[Artlets.com]



[Yin at al., 2022]

Gamma Correction



[Ngo and Kang, Electronics, 2021]

Gamma Correction

gamma=1.0

gamma=3.0

gamma=5.0

gamma=5.0

Point Operators

Histogram Equalization (idealized)

Produce image $g = \varphi f$ with ***'constant' (flat) histogram*** h_g

All luminance values should be 'equally' probable
(as 'equal' as possible for finite number of pixels)

Useful consequence: ***cumulative histogram*** $H_g(u) = u$

Why? Think: integrate a constant

Luminance value: $u = f(x)$

Preserve ***order***: $u_1 \leq u_2 \Rightarrow \varphi(u_1) \leq \varphi(u_2)$

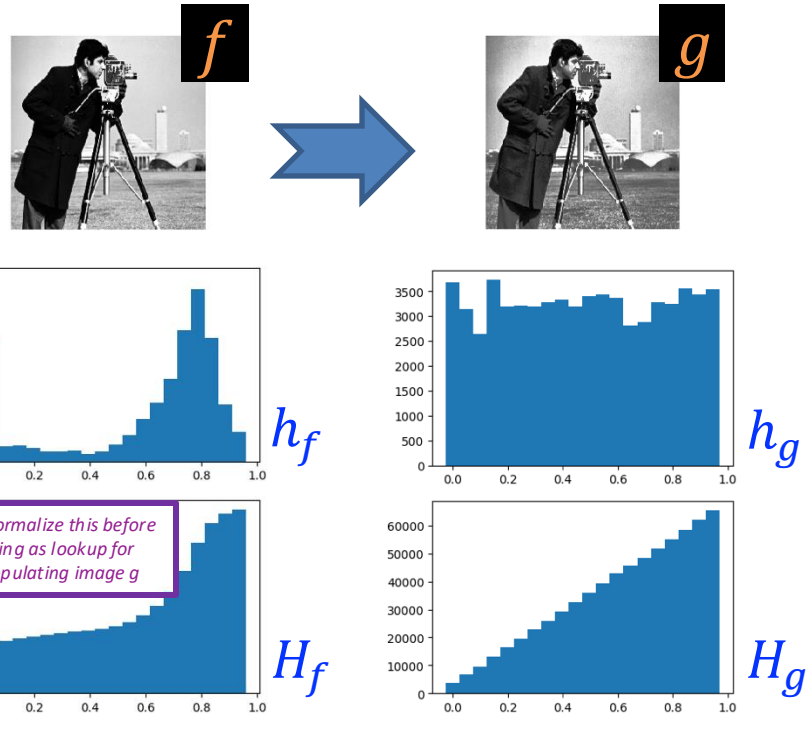
So, the **pth percentile** of f & g should be the same:

$$H_f(u) = H_g(\varphi(u)) = \varphi(u)$$

Visit each pixel of f
replace its value u with:

$$\varphi(u) = H_f^{norm}(u)$$

To populate a pixel of g:
- visit the corresponding pixel location of f
- read its value u (i.e., $f(x)$)
- use u as key to 'query' (the *normalized*) H_f as "lookup table"
- Copy value from "lookup table" as value of pixel of g



Point Operators

Histogram Thresholding

Segment foreground (objects of interests) from background

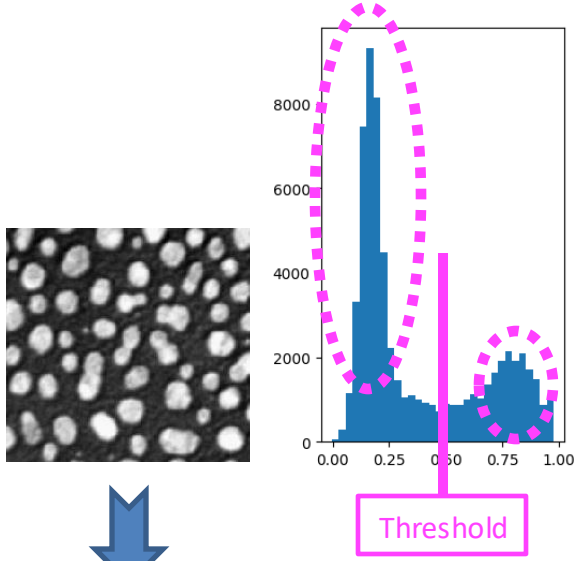
- Manually ☹ choose threshold t → looking at histogram (bimodal distribution)
- ***Automatically*** choose threshold t → ***IsoData*** thresholding

$m_L(t)$ → Mean of pixels with value: $\leq t$

$m_H(t)$ → Mean of pixels with value: $> t$

$$t = \frac{m_L(t) + m_H(t)}{2} = m(t)$$

Function of itself!



Iterative Solution

- Start with initial guess t_i for $i = 0$
- Compute $t_{i+1} = m(t_i)$
- Repeat until convergence
  (i.e. repeat as long as t_{i+1} and t_i are not too similar)

Resources for Histogram Slides

https://rvdboomgaard.github.io/ComputerVision_LectureNotes/LectureNotes/IP/Images/index.html

https://rvdboomgaard.github.io/ComputerVision_LectureNotes/LectureNotes/IP/PointOperators/index.html

Additional Resources

- Photometric Stereo
  - Shree Nayar: YouTube
  - Andreas Geiger: YouTube
- Specular and Diffuse Reflection
  - Steve Seitz: YouTube / YouTube

Disclaimer

Many of the slides used here are obtained from online resources (including many open lecture materials) without appropriate acknowledgement. They are used here for the sole purpose of classroom teaching. All the credit and all the copyrights belong to the original authors. You should not copy it, redistribute it, put it online, or use it for any other purposes than for this course.