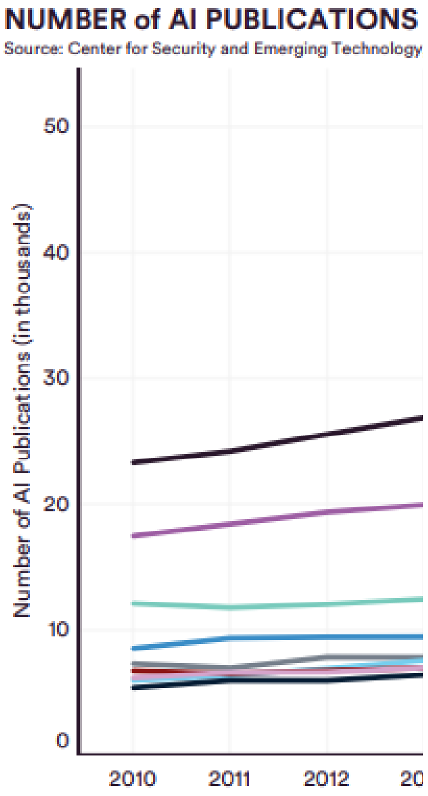
Light, Shading, Color, Pixels, and Histogram

Dr. Sander Ali Khowaja,
Assistant Professor, Department of Telecommunication Engineering
Faculty of Engineering and Technology, University of Sindh, Pakistan
Senior Member, IEEE – Member, ACM
https://sander-ali.github.io

Computer Vision & Image Processing

Computer Vision in the News

https://aiindex.stanford.edu/wp-content/uploads/2022/03/2022-AI-Index-Report_Master.pdf



Artificial Intelligence Index Report 2022

CHAPTER

Computer vision is the subfield of AI that teaches machines to understand images and videos. The
vision tasks, such as image classification, object recognition, semantic segmentation, and face de
outperform humans on a plethora of computer vision tasks. Computer vision technologies have a
applications, such as autonomous driving, crowd surveillance, sports analytics, and video-game c

DEEPFAKE DETECTION

Many AI systems can now generate fake images that are indistinguishable from real ones. A related technology involves superimposing one person's face onto another, creating a so-called "deepfake." Deepfakes are used for purposes ranging from advertising to generating misogynistic pornography and disinformation (in 2018, for example, a deepfake video of Barack Obama uttering profanities about Donald Trump was circulated online over 2 million times). In the last few years, AI researchers have sought to keep up with improving deepfake technologies by crafting stronger deepfake detection algorithms.

FaceForensics++

FaceForensics++ is a deepfake detection benchmarking dataset that contains approximately 1,000 original video sequences sourced from YouTube videos. Progress on FaceForensics++ is measured in terms of accuracy: the percentage of altered images an algorithm can correctly identify.

Although FaceForensics++ was introduced in 2019, researchers have tested previously existing deepfake detection methods on the dataset in order to track progress over time in deepfake detection (Figure 2.1.7). In the last decade, AI systems have become better and better at detecting deepfakes. In 2012, the top-performing systems could correctly identify 69.9% of deepfakes across all four FaceForenics++ datasets. In 2021, that number increased to 97.7%.[1]

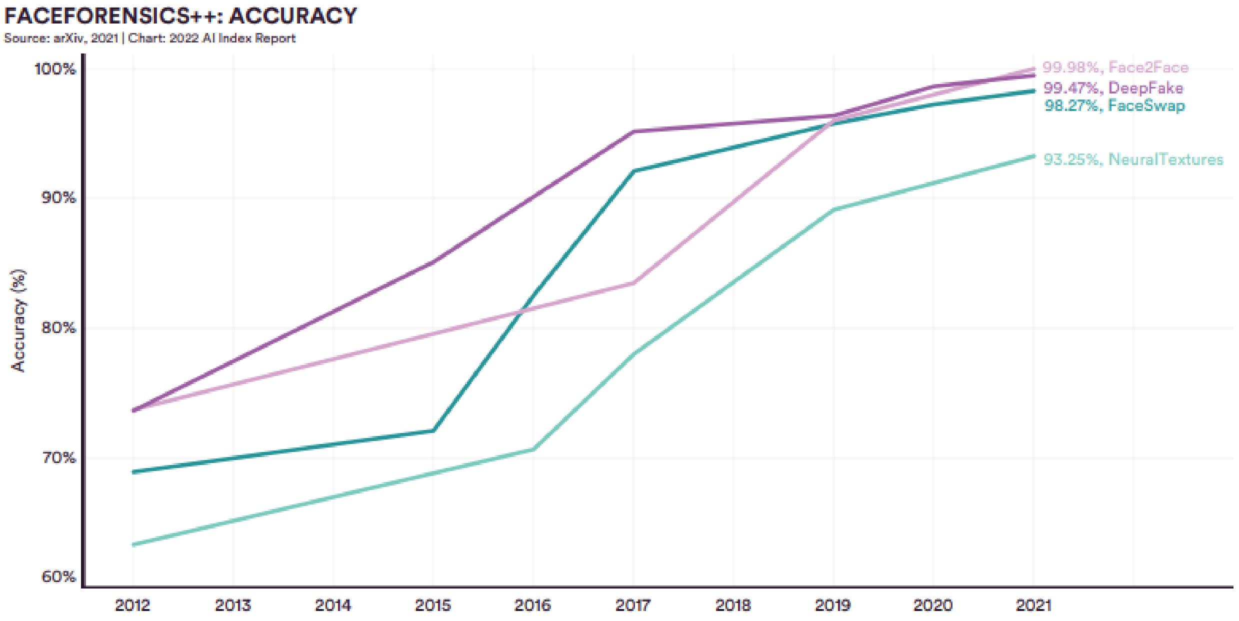
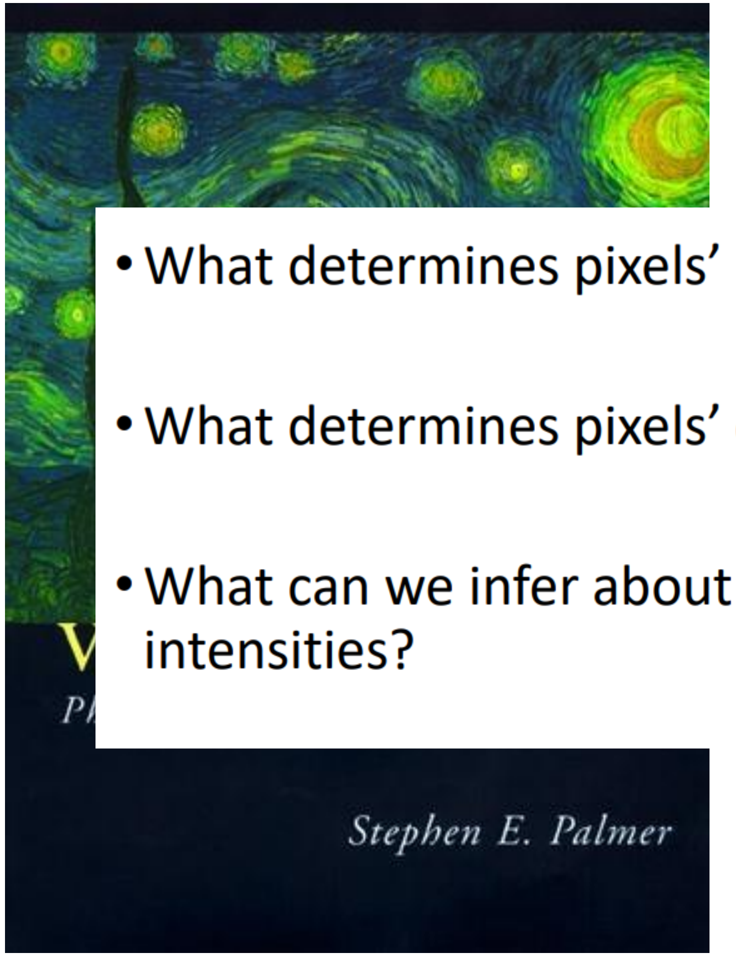


Figure 2.1.7

What is Color?

- The result of interaction between physical light in the environment and our visual system.
- A *psychological property* of our visual experiences when we look at objects and lights, *not a physical property* of those objects or lights.



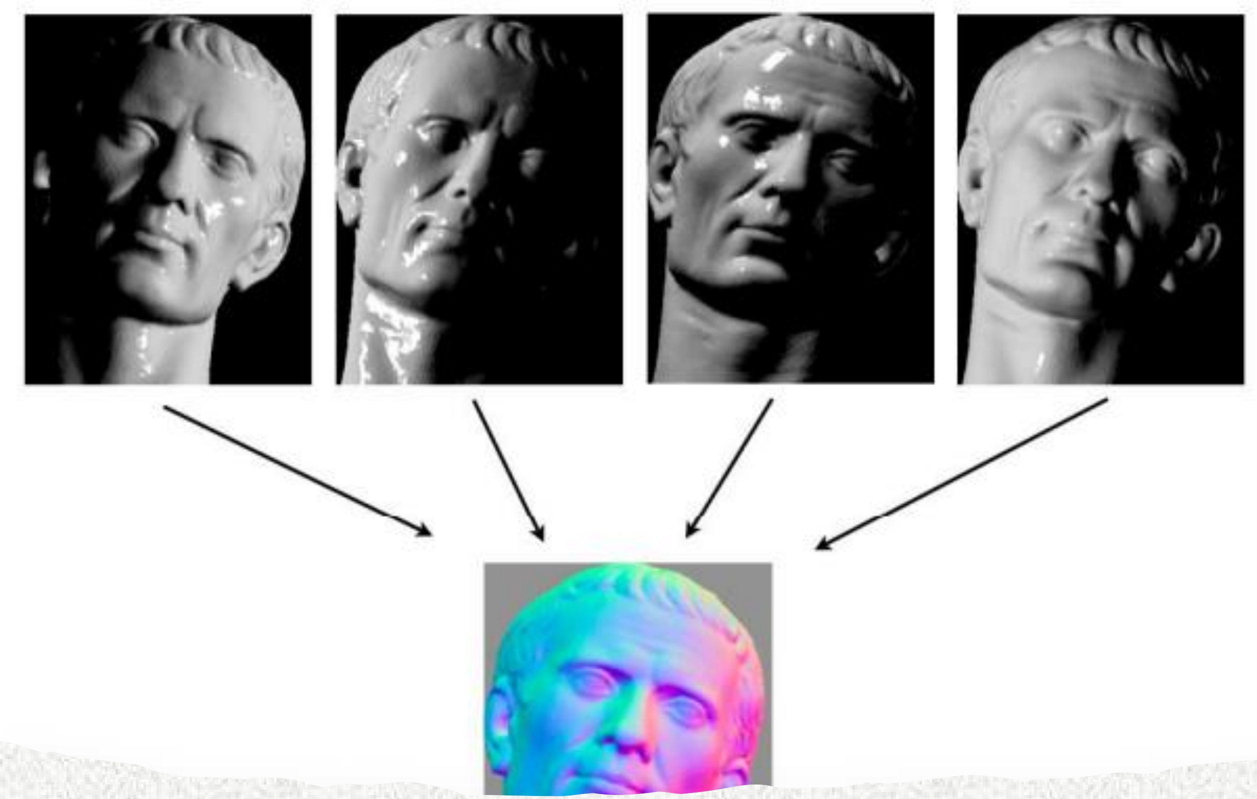
- What determines pixels' **brightness**?
- What determines pixels' **color**?
- What can we infer about the scene from pixel intensities?

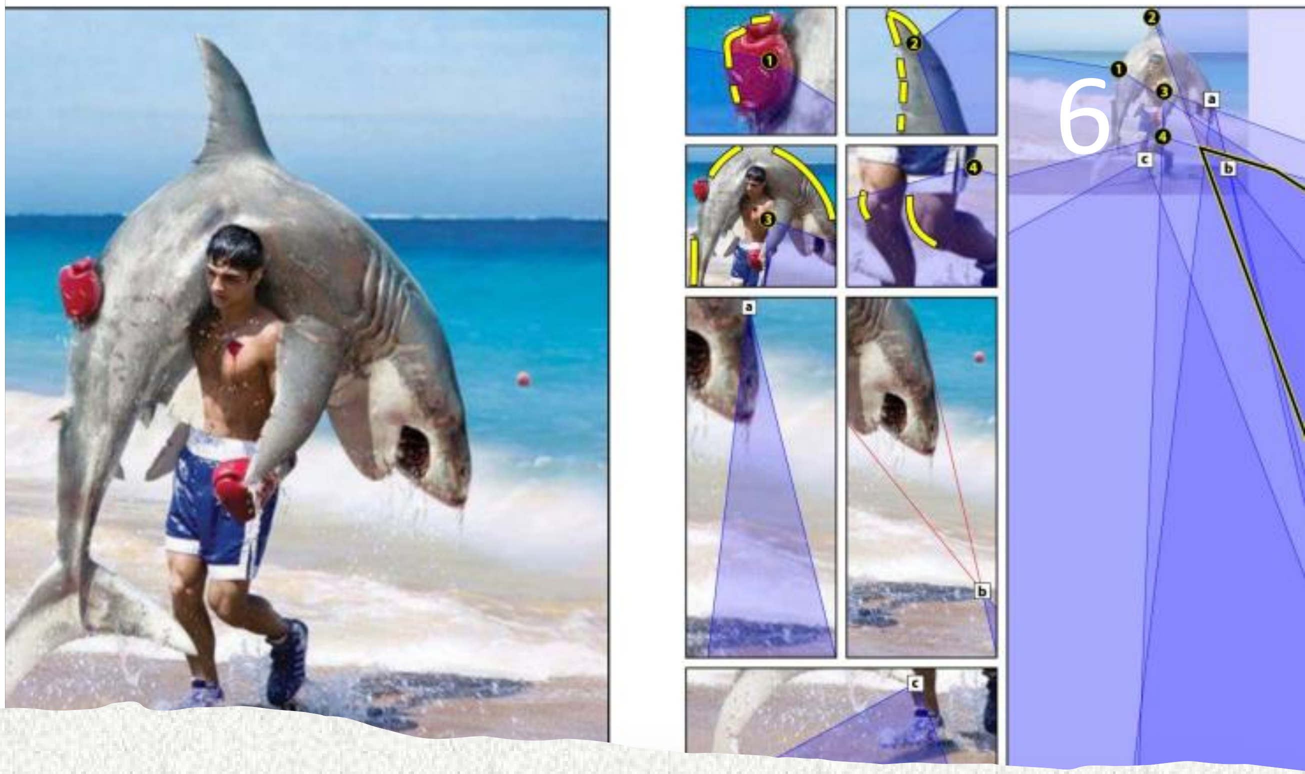
Slide credit: Lana Lazebnik

Interaction of Light and Surfaces

- What is the observed color of any surface under monochromatic light?

Interaction of Light and Surfaces

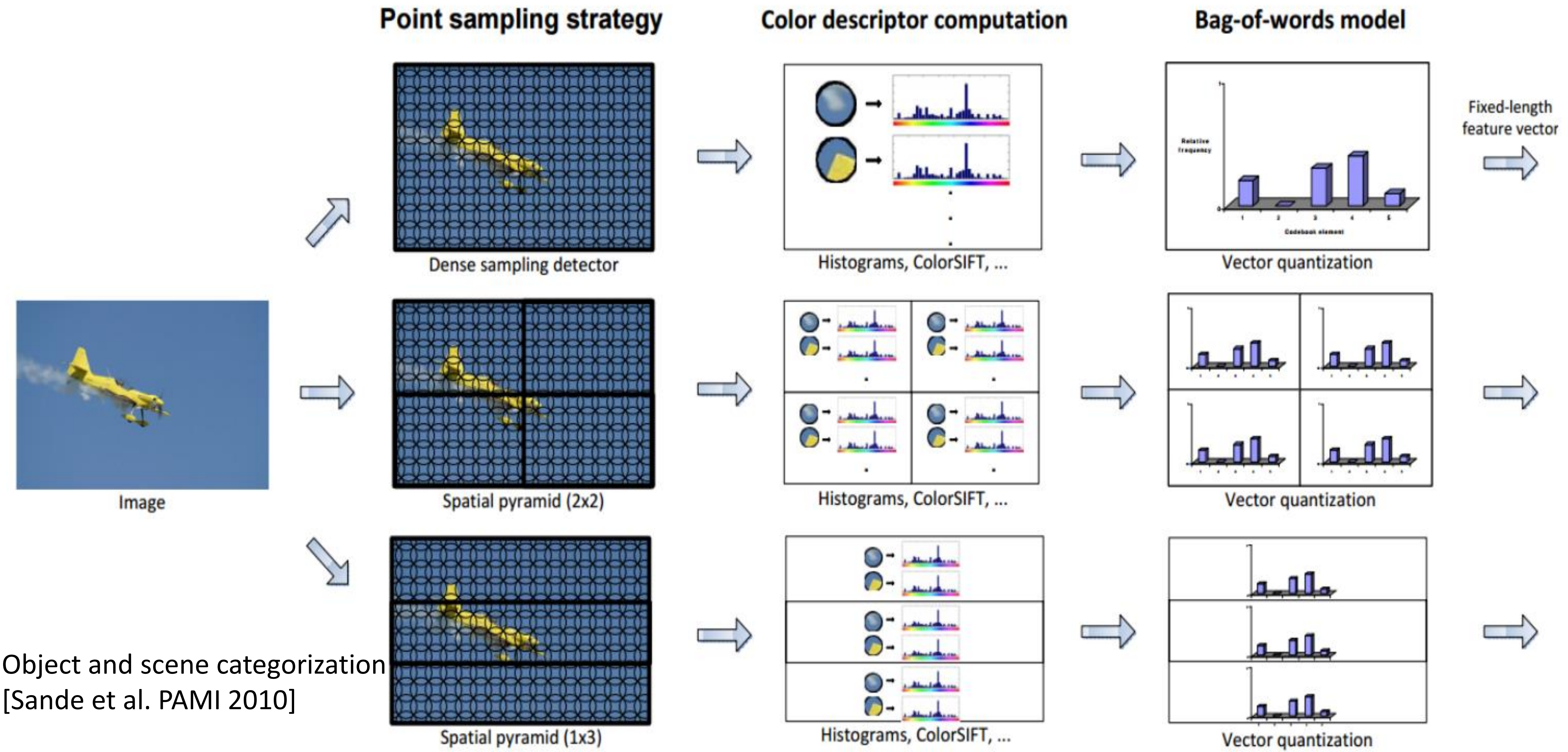




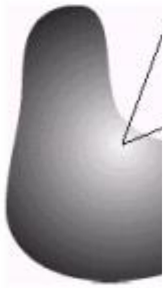
Why Should we Care?

- https://en.wikipedia.org/wiki/Photometric_stereo
- Exposing Photo Manipulation from Shading and Shadows [Kee et al. TOG 14]

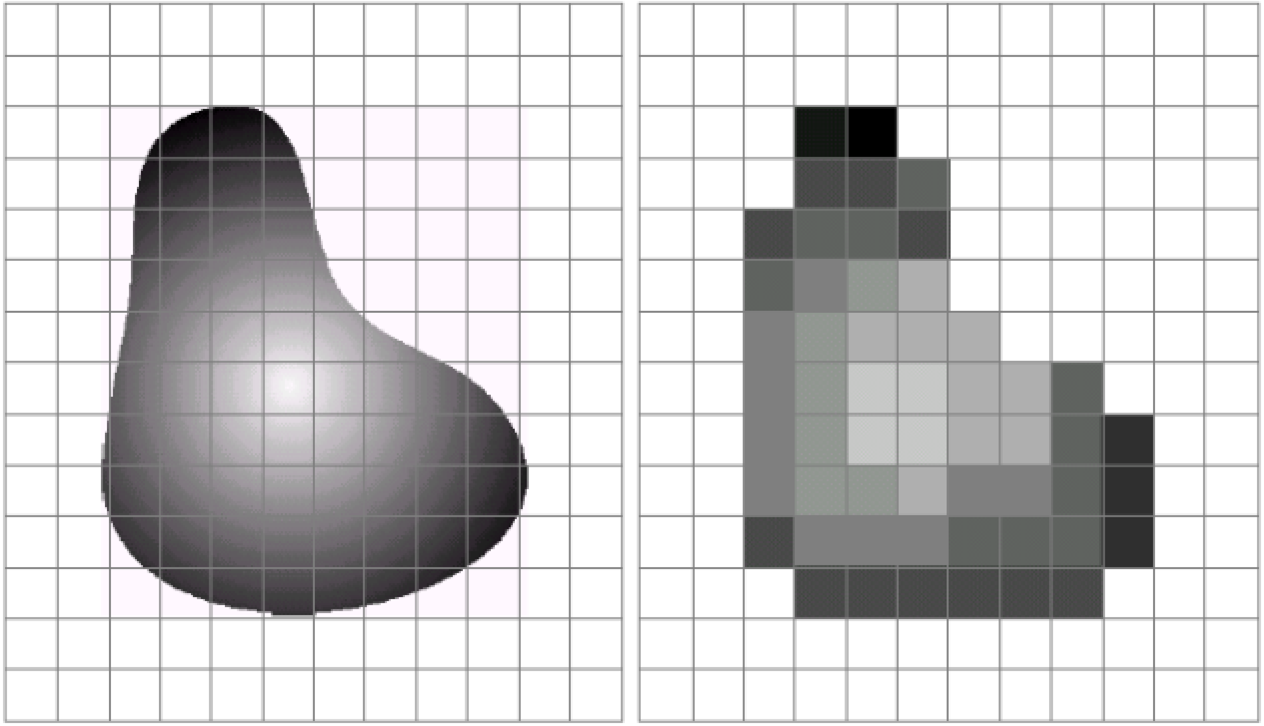
Why Should We Care?



Object and scene categorization
[Sande et al. PAMI 2010]

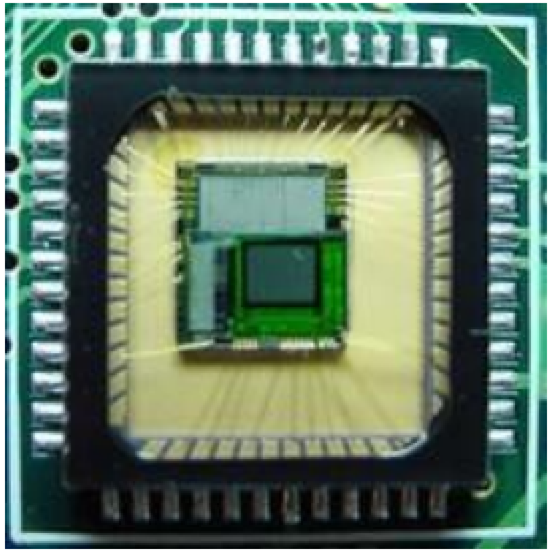


Scene elem



a b

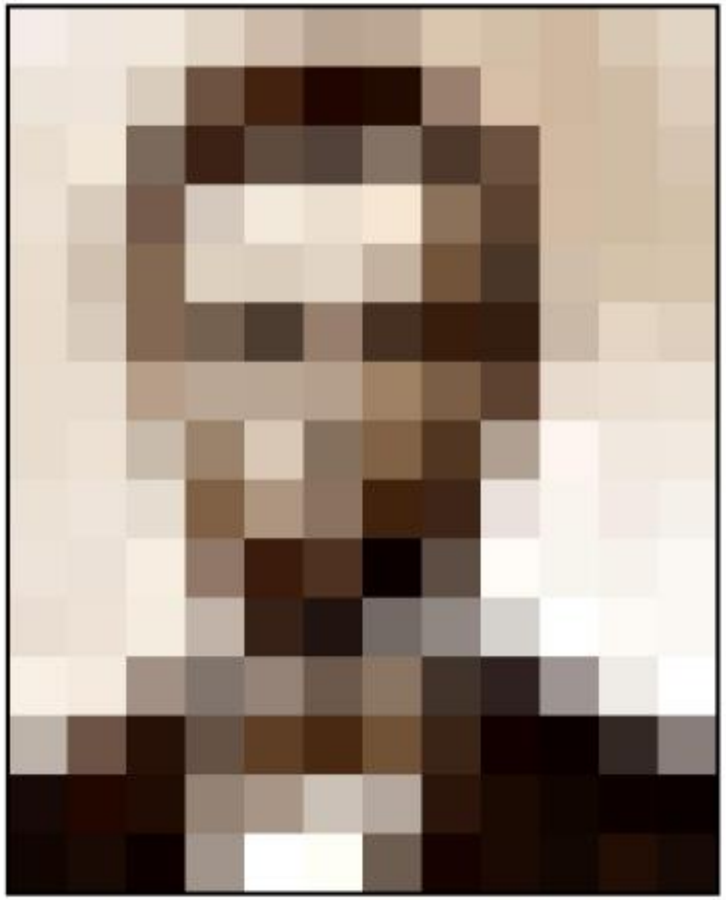
FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.



CMOS sensor

The Eye

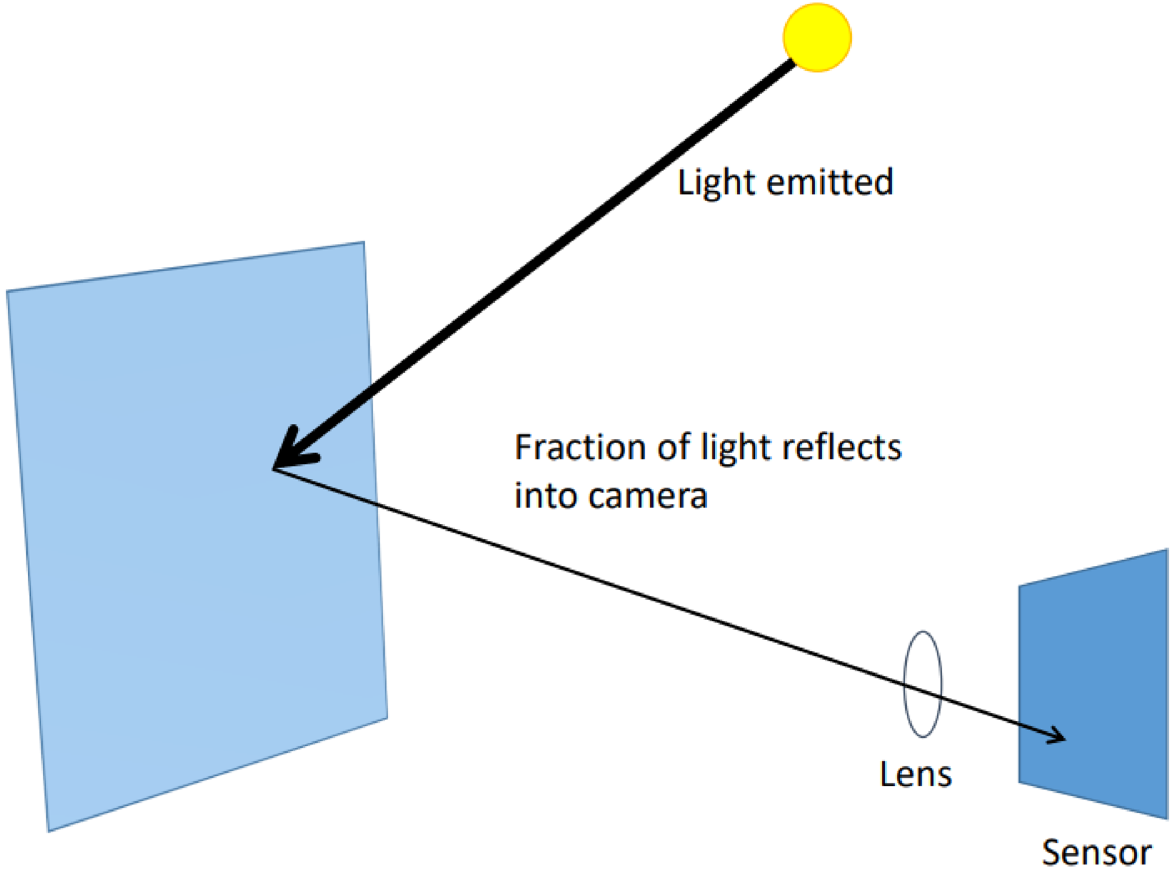
What Humans see vs What Computers See



243	239	240	225	206	185	188	218	211	206	216	225
242	239	218	110	67	33	34	152	213	206	208	221
243	242	123	58	94	82	132	77	108	208	208	215
235	217	115	212	243	236	247	139	91	209	208	211
233	208	131	222	219	226	196	114	74	208	213	214
232	217	131	116	77	150	69	56	52	201	228	223
232	232	182	186	184	179	159	123	93	232	235	235
232	236	201	154	216	133	129	81	175	252	241	240
235	238	230	128	172	138	65	63	234	249	241	245
237	236	247	143	59	78	[illegible]	94	255	248	247	251
234	237	245	193	55	33	115	144	213	255	253	251
248	245	161	128	149	109	138	65	47	156	239	255
190	107	38	102	94	73	114	58	[illegible]	[illegible]	51	137
23	32	33	148	168	203	179	43	27	32	[illegible]	[illegible]
17	26	[illegible]	160	255	255	109	[illegible]	26	10	35	24

Slide credit: Larry Zitnick

How does a pixel get its value

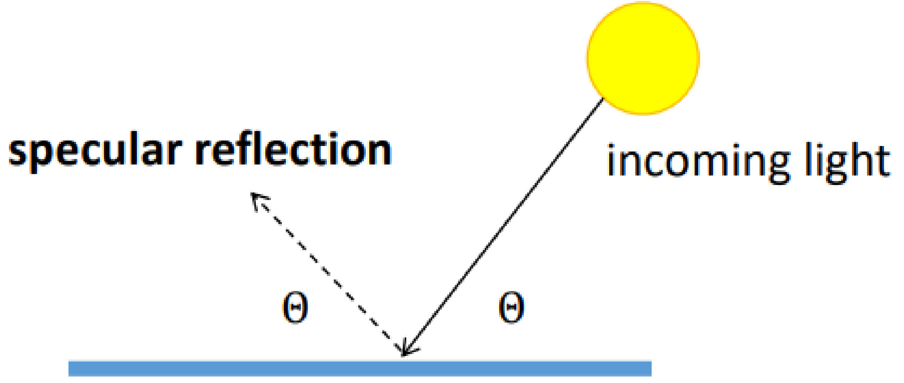


Major factors
- Illumination strength and direction
- Surface geometry
- Surface material
- Nearby surfaces
- Camera gain/exposure

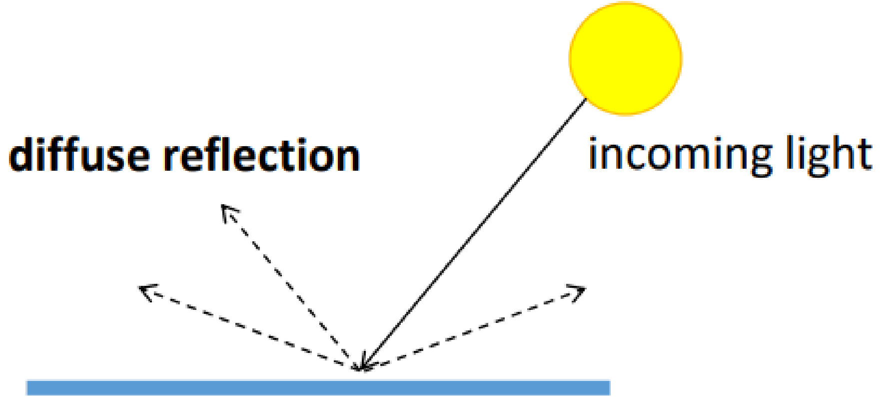
Slide credit: Derek Hoiem

Basic Models of Reflection

- Specular: light bounces off at the incident angle
    - E.g., mirror



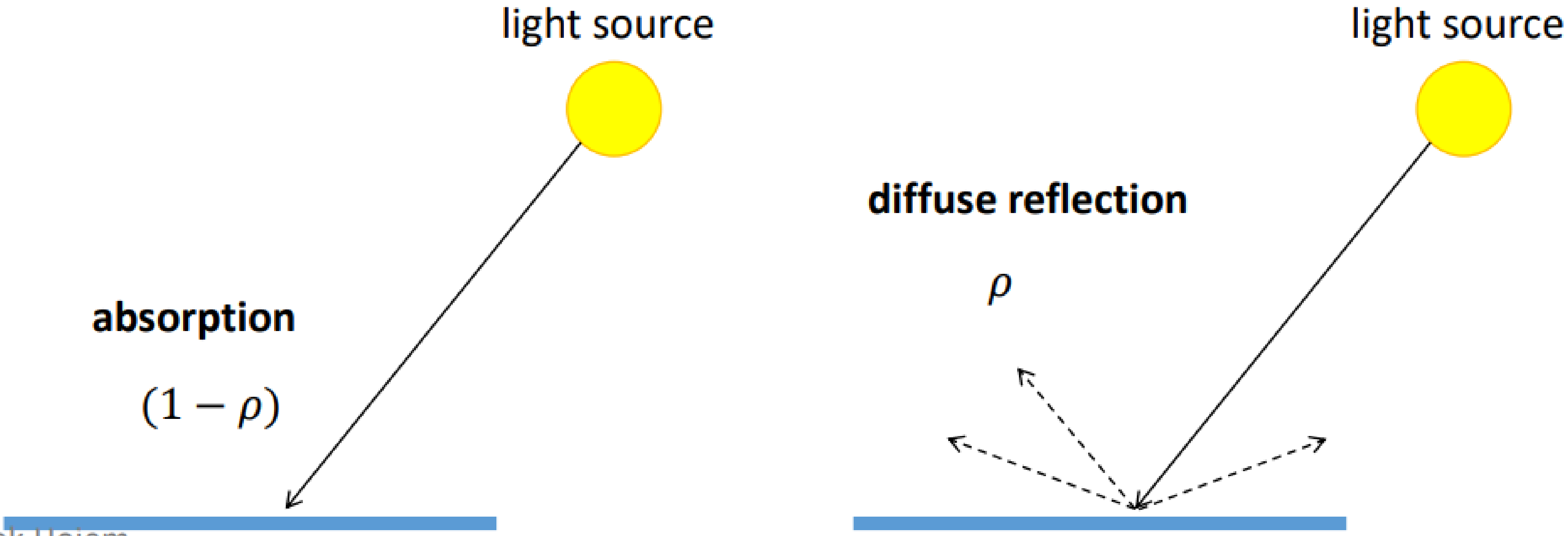
- Diffuse: light scatters in all directions
    - E.g., brick, cloth, rough wood



Slide credit: Derek Hoiem

Lambertian Reflectance Model

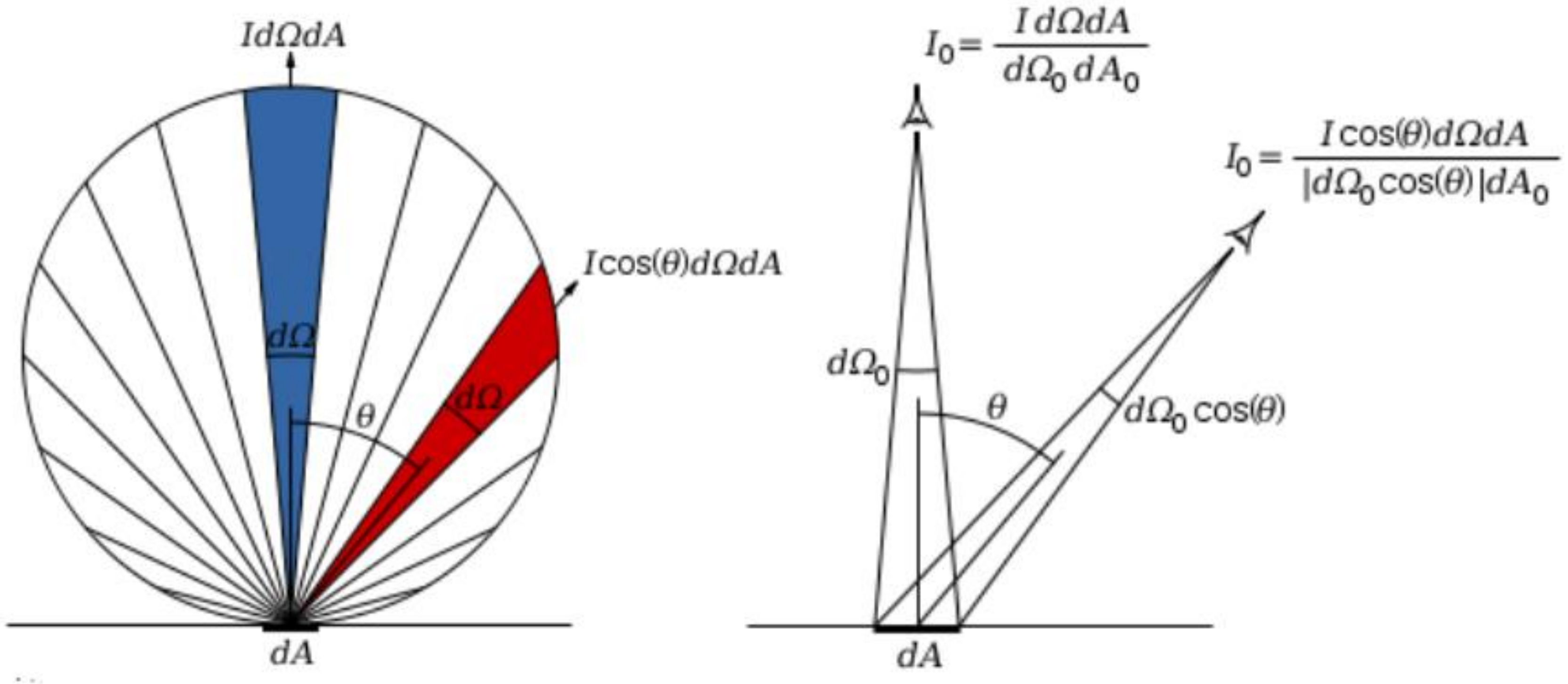
- Some light is absorbed (function of albedo ρ)
- Remaining light is scattered (diffuse reflection)
- Examples: soft cloth, concrete, matte paints



Slide credit: Derek Hoiem

Diffuse Reflection: Lambert's Cosine Law

- Intensity does not depend on viewer angle.
  - Amount of reflected light proportional to $\cos(\theta)$
  - Visible Solid angle also proportional to $\cos(\theta)$



Slide credit: Derek Hoiem

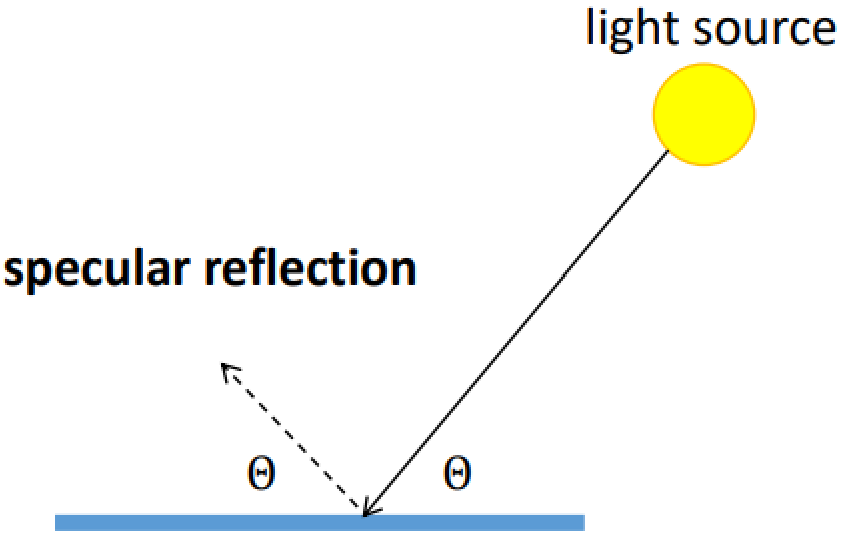
http://en.wikipedia.org/wiki/Lambert's_cosine_law

Specular Reflection

- Reflected Direction depends on light orientation and surface normal
  - E.g Mirrors are fully specular
  - Most surfaces can be modeled with a mixture of diffuse and specular components

Flickr, by suzysputnik

Flickr, by piratejohnny



Slide credit: Derek Hoiem

Most Surfaces have both Specular and Diffuse Component

- Specularity = Spot where specular reflection dominates (typically reflects light source)

Typically, specular component is small

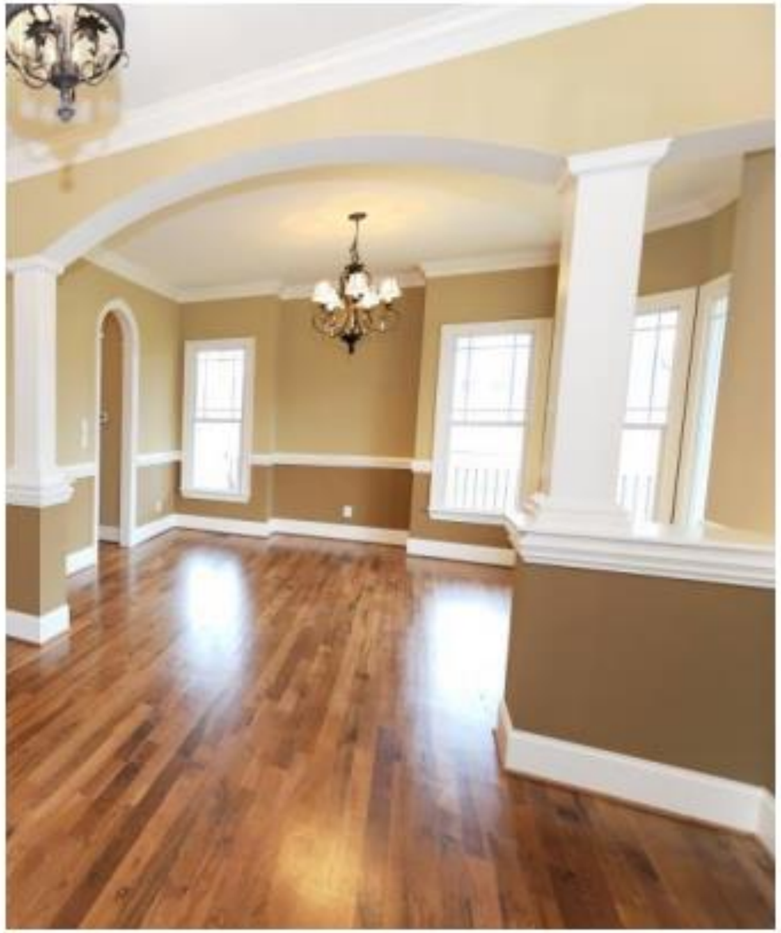


Photo: northcountryhardwoodfloors.com

Slide credit: Derek Hoiem

Intensity and Surface Orientation

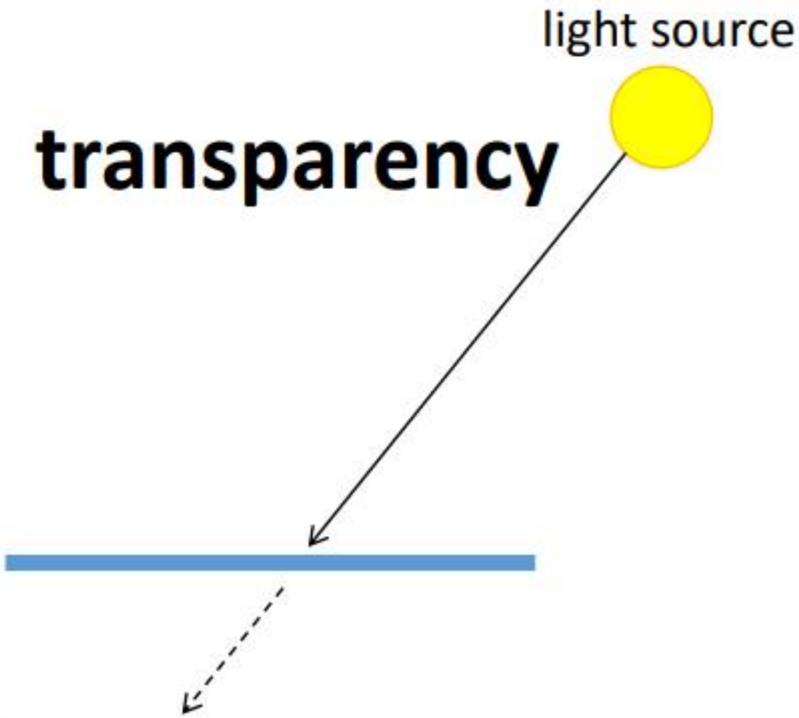
- Intensity depends on Illumination Angle. Why?
  - Less Light Comes in at Oblique Angles

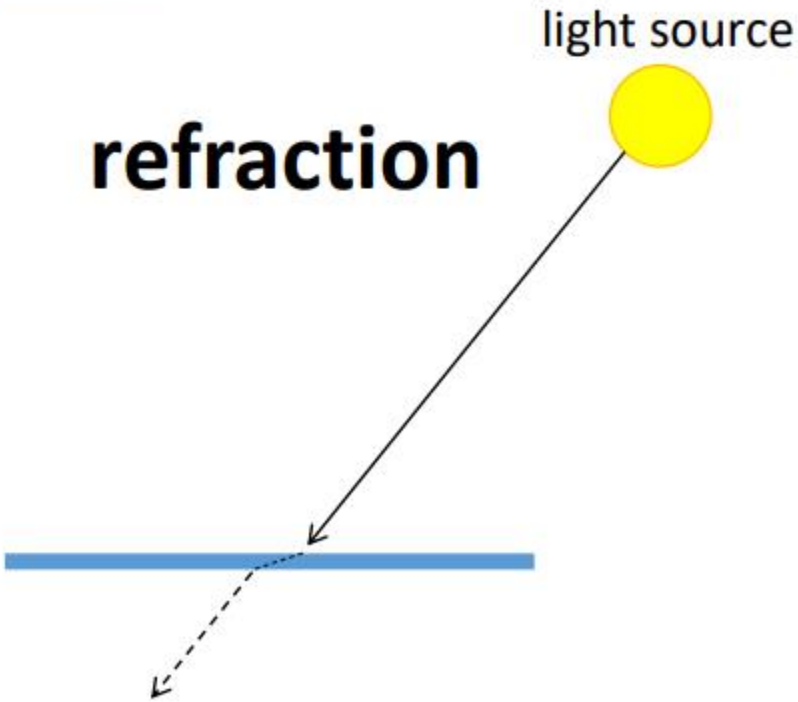
- ρ = Albedo: fraction of light that is reflected
- $\boldsymbol{S}$ = directional source
- $\boldsymbol{N}$ = surface normal
- I = reflected intensity

$$I(x) = \rho(x)(\boldsymbol{S} \cdot \boldsymbol{N}(x))$$

Slide credit: Forsyth

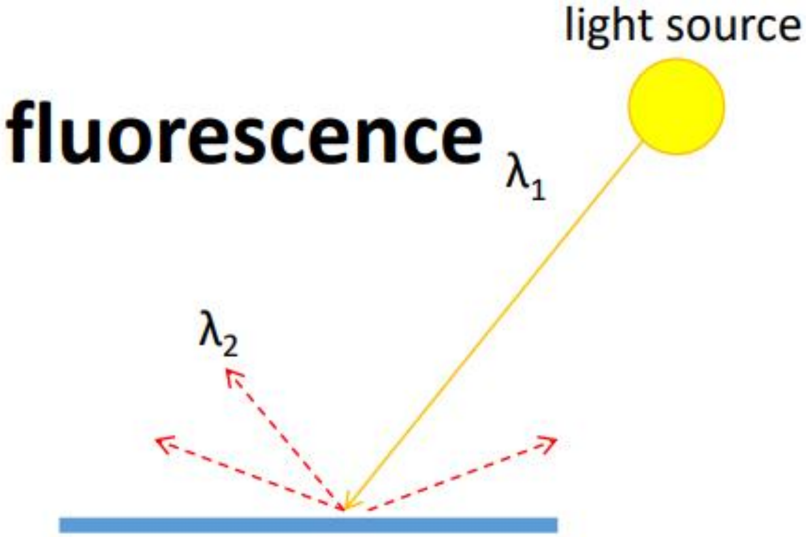
Other Possible Effects

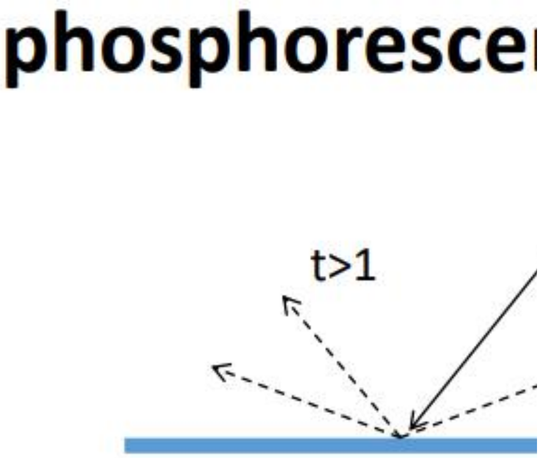


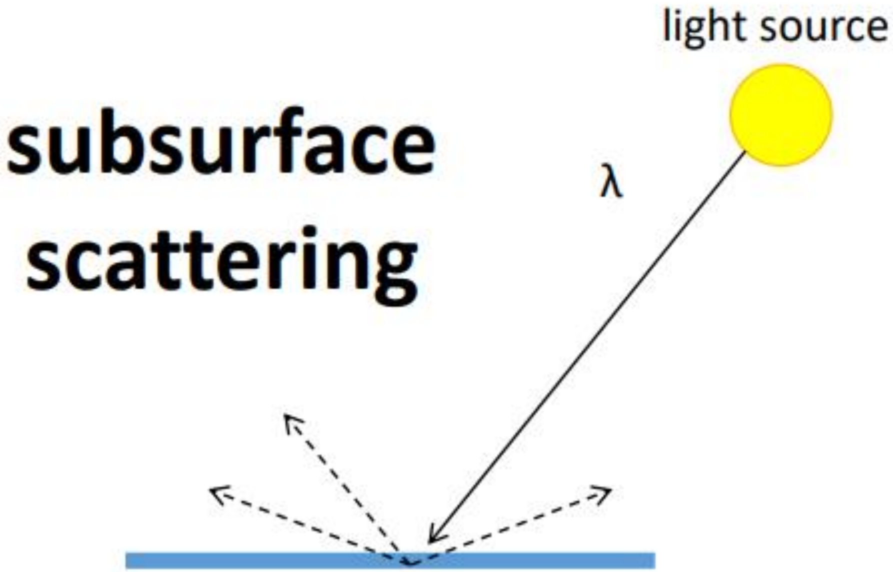


Slide credit: Derek Hoiem

Other Possible Effects



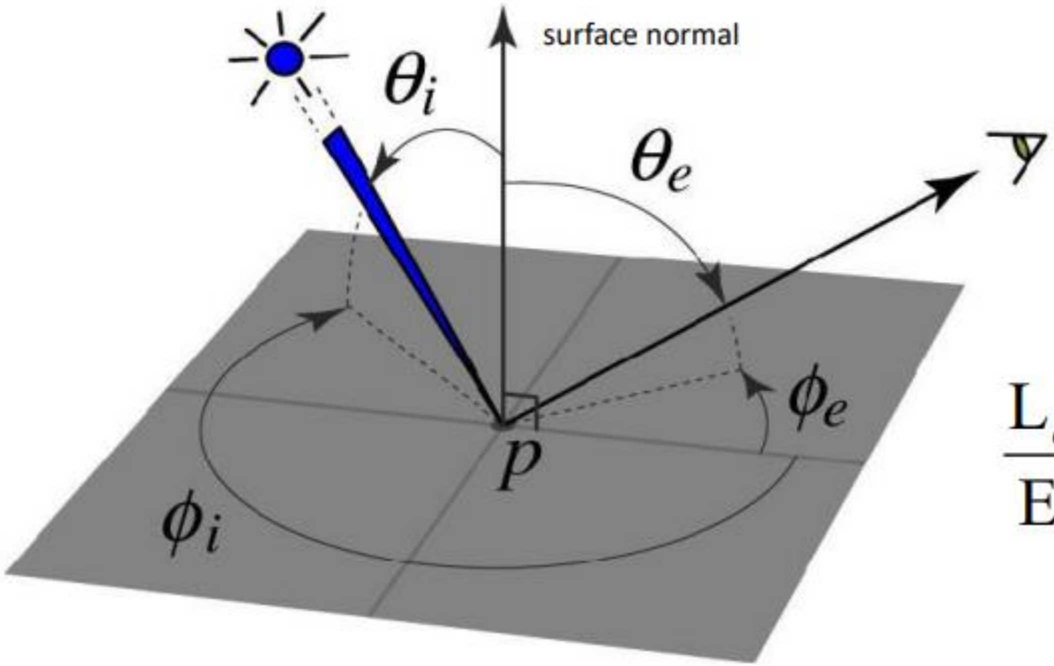




Slide credit: Derek Hoiem

BRDF: Bidirectional Reflectance Distribution Function

- Model of Local Reflection that tells how bright a surface appears when viewed from one direction when light falls on it from another

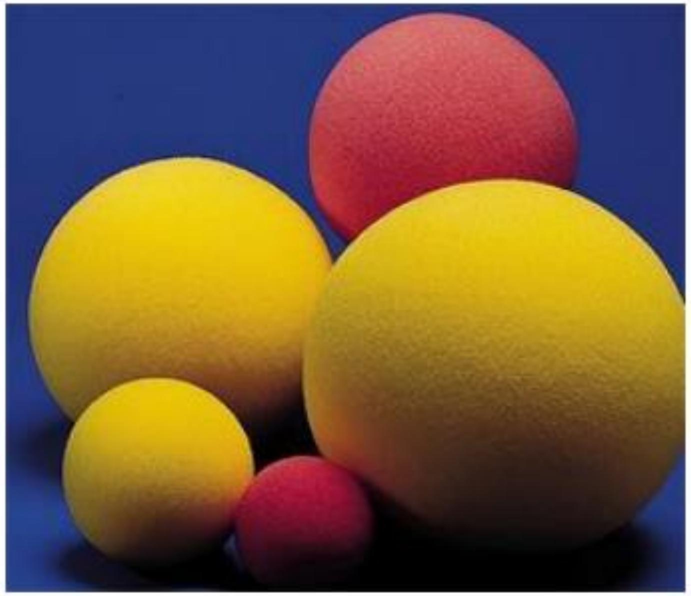


$$\rho(\theta_i, \phi_i, \theta_e, \phi_e; \lambda) =$$

$$\frac{L_e(\theta_e, \phi_e)}{E_i(\theta_i, \phi_i)} = \frac{L_e(\theta_e, \phi_e)}{L_i(\theta_i, \phi_i) \cos\theta_i d\omega}$$

Slide credit: S. Savarese

Reflection Models



Lambertian: reflection all diffuse

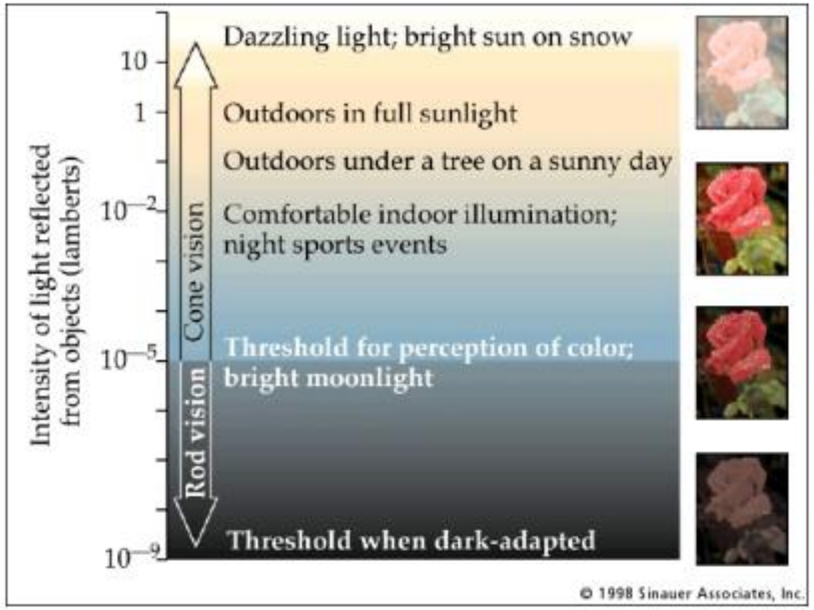
Mirrored: reflection all specular

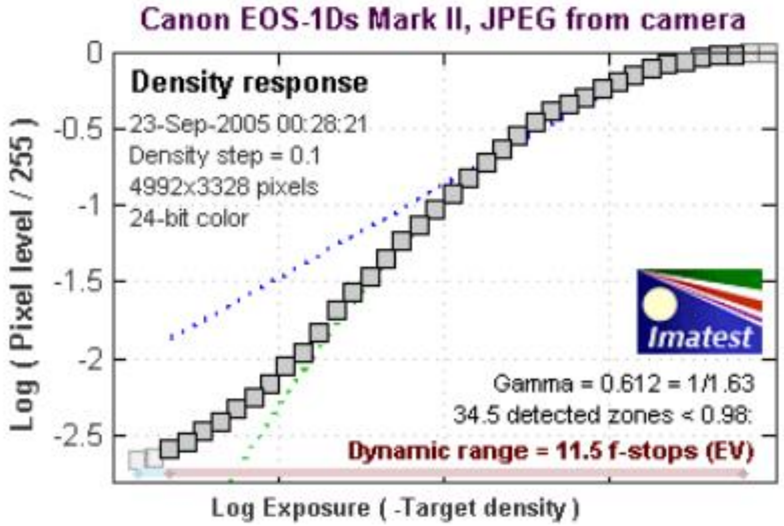
Glossy: reflection mostly diffuse, some specular

Dynamic Range and Camera Response

- Typical Scenes have a huge dynamic range

- Camera Response is roughly linear in the mid range (15 to 240) but non-linear at the extremes
  - Called saturation or under-saturation



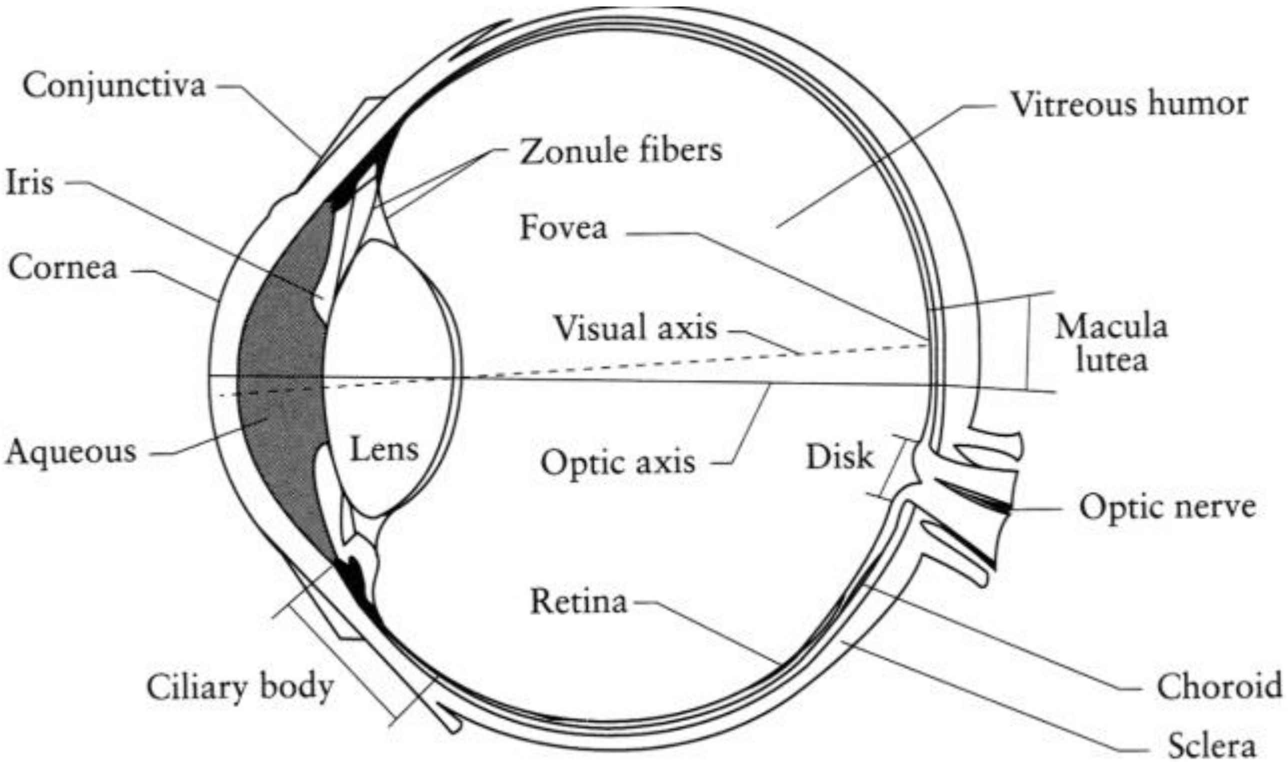


What Determines Pixels’ Color?

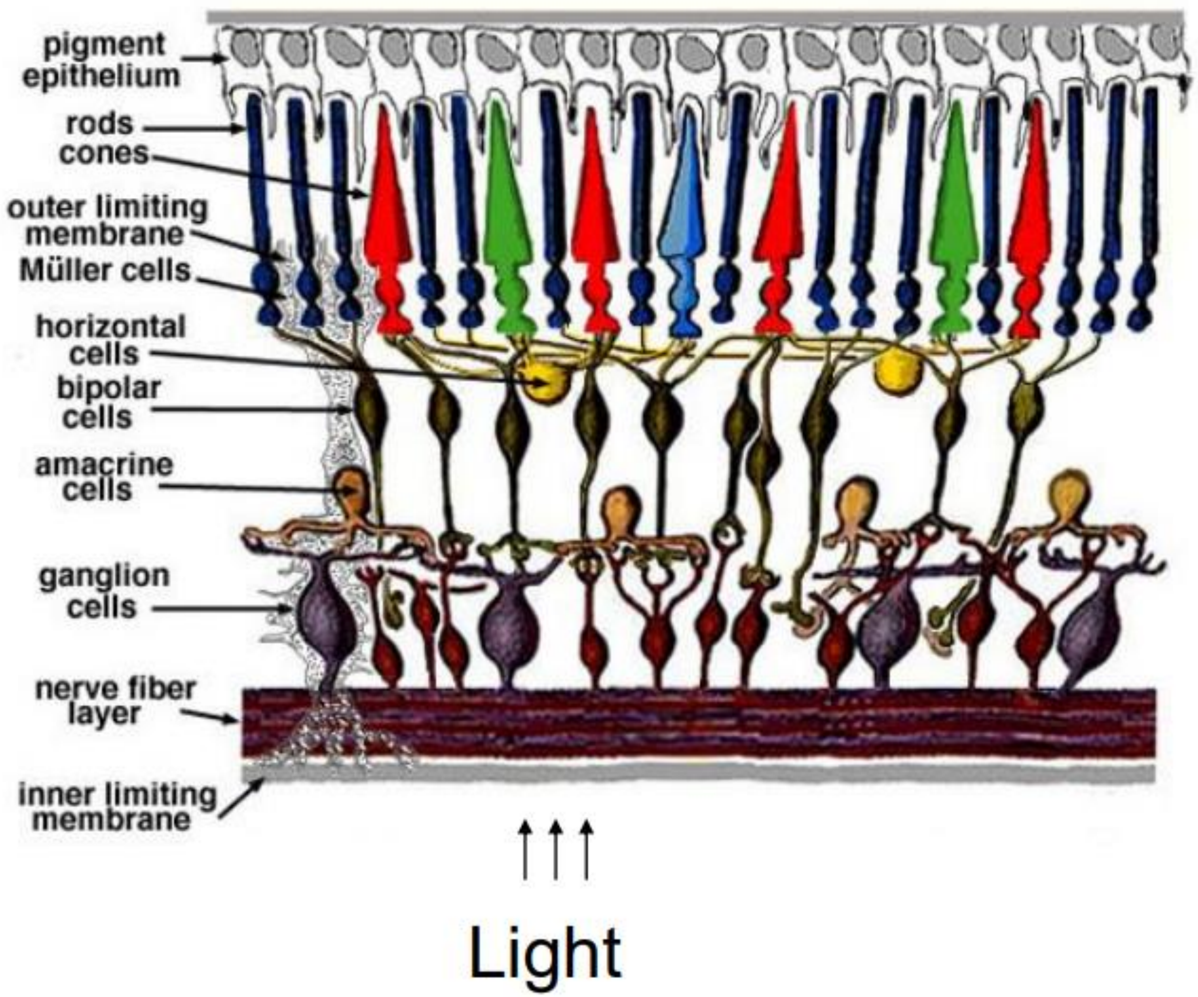
https://upload.wikimedia.org/wikipedia/commons/b/b1/Colouring_pencils.jpg

The Eye

- The Human Eye is a camera!
  - Iris – Colored annulus with radial muscles
  - Pupil – The Hole (aperture) whose size is controlled by the Iris
  - What's the "film"? Photoreceptor cells (rods and cones) in the retina



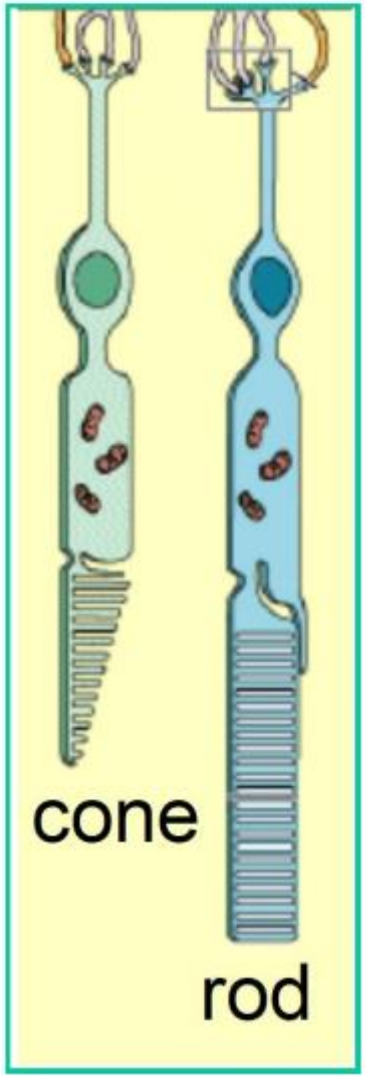
Retina Up-Close



Two types of Light-Sensitive Receptors

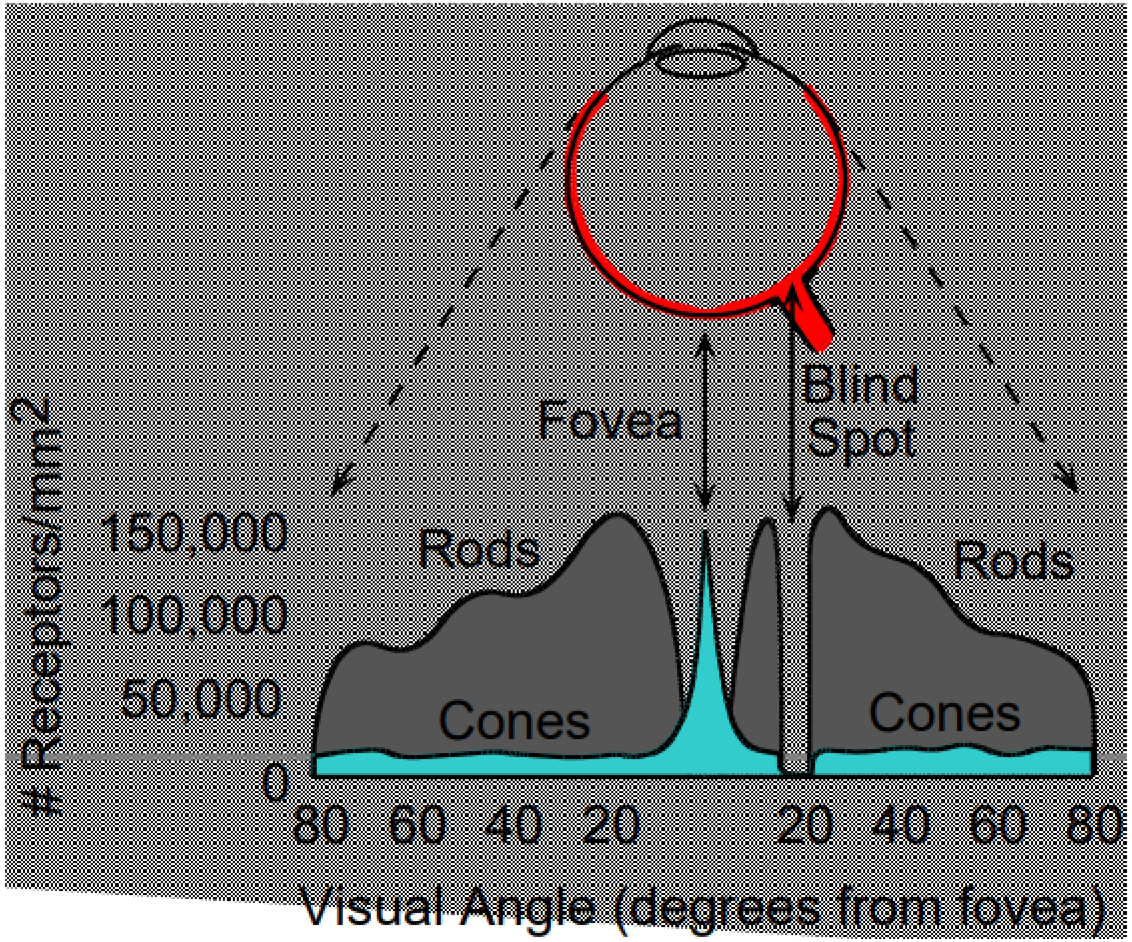
- Cones
  - Cone-shaped
  - Less Sensitive
  - Operate in high light color vision

- Rods
  - Rod-shaped
  - Highly sensitive
  - Operate at night
  - Gray-scale vision
  - Slower to respond



Slide Credit: Efros

Distribution of Rods and Cones



How to find your blind spot

Sit about a foot away from your screen.

•To find your right eye's blind spot:

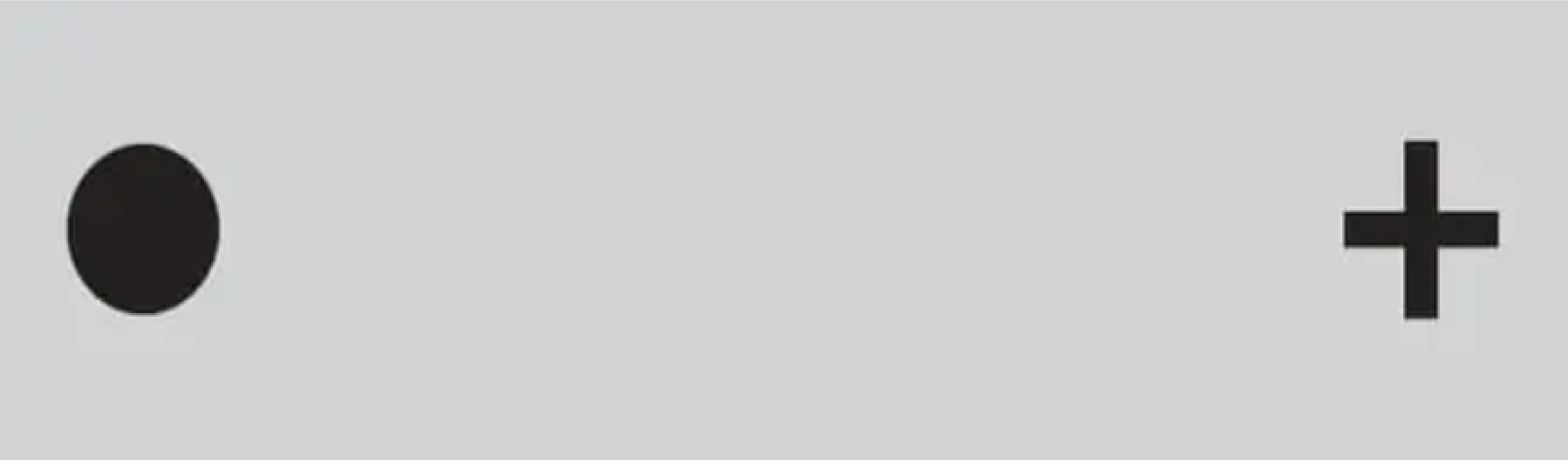
- Close your left eye.
- Stare at the circle.
- Move closer to the screen, then farther away.
- Keep doing this until the plus sign disappears.
- When it disappears, you found your right eye's blind spot.

•To find your left eye's blind spot:

- Close your right eye.
- Stare at the plus sign.
- Move closer, then farther away. Repeat.
- When the circle disappears, you found your left eye's blind spot.

Night Sky: why are there more stars off-center?

Slide credit: Efros



Blind Spot

You may not realize it, but both your eyes have a natural blind spot, or scotoma. Everyone has them. They're normal and you probably don't notice them.

Your retina, which is a thin layer of neural tissue at the back of your eye, is made up of tiny, light-detecting cells called photoreceptors. When light lands on your retina, it sends electrical bursts through your optic nerve to your brain. Your brain turns the signals into a picture.
The spot where your optic nerve connects to your retina has no light-sensitive cells, so you can't see anything there. That's your blind spot.

Why You Don't Notice It

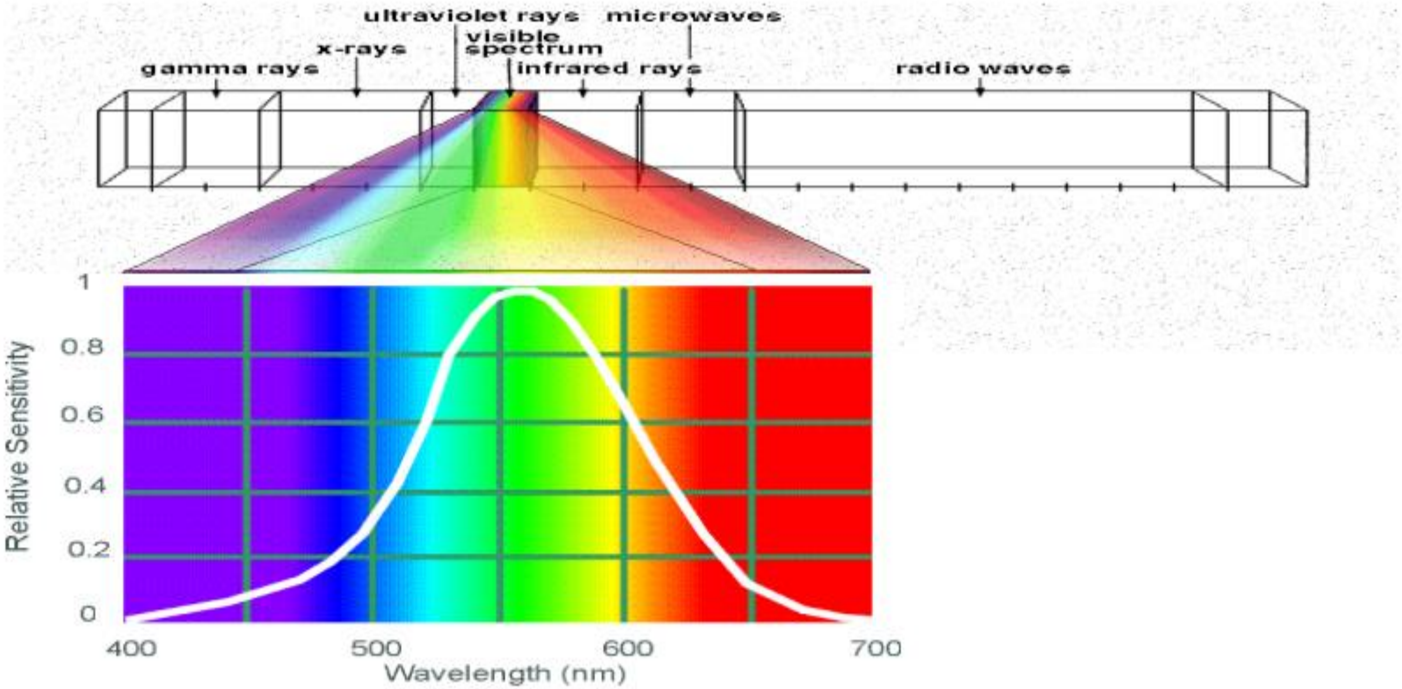
You probably don't notice your blind spot because your other eye makes up for it.
Each eye sends data to your brain on its own, so your brain fills in what's missing.
What one eye doesn't see, your other eye does.
Experts aren't sure how your brain fills in the details. They think it's a mix of processing what it thinks is missing and reusing electrical bursts around your blind spot.

Should You Worry About Your Blind Spot?

Everyone has a natural blind spot in each eye. It isn't something you need to worry about, unless you notice problems with your vision.

https://www.webmd.com/eye-health/eye-blind-spot

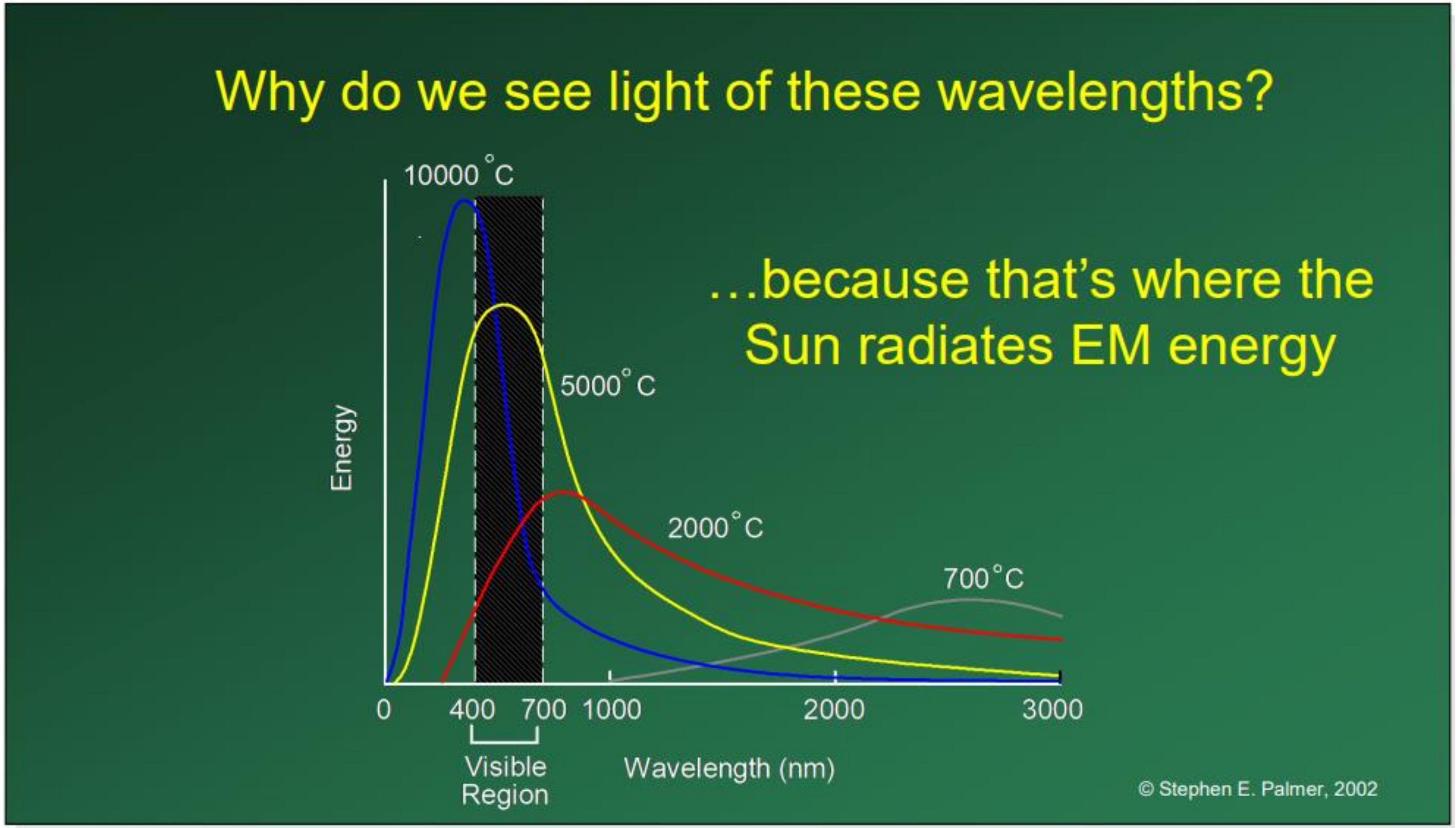
Light: Electromagnetic energy whose wavelength is between 400 nm and 700 nm. (1 nm = 10 -9 meter)



Human Luminance Sensitivity Function

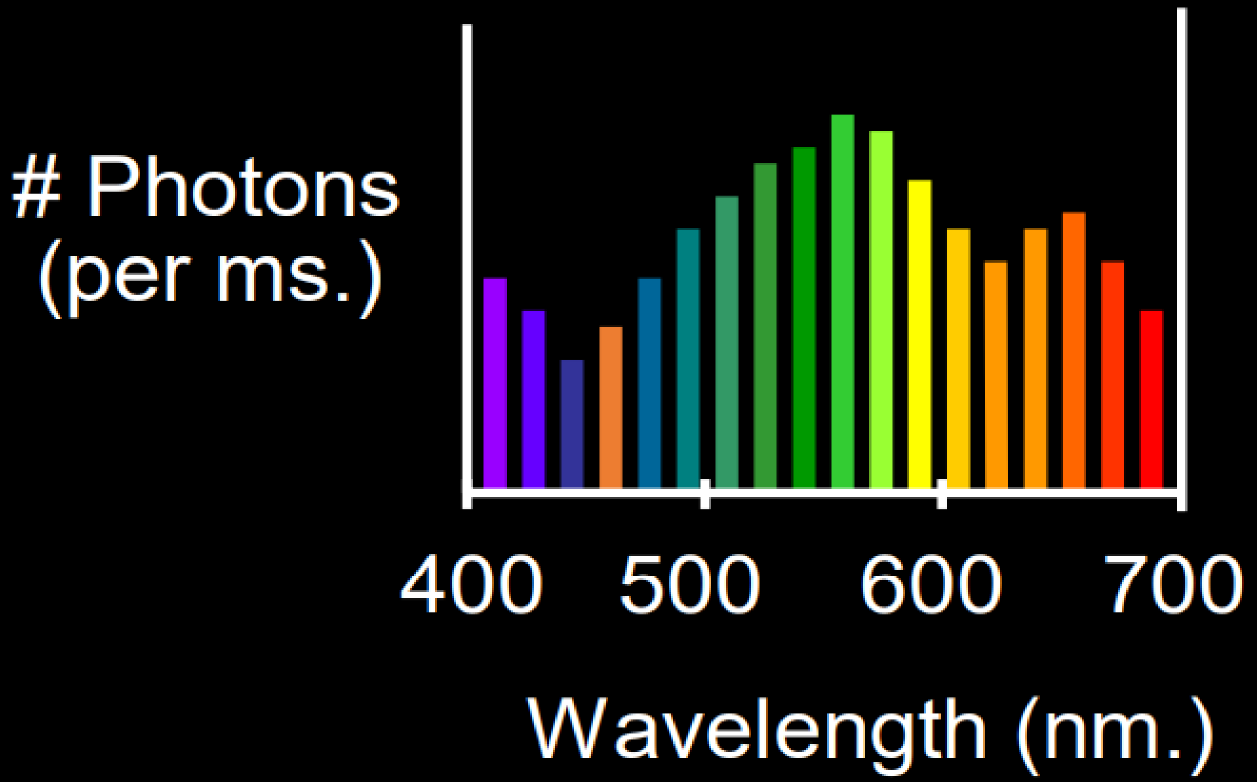
http://www.yorku.ca/eye/photopik.htm

Visible Light



The Physics of Light

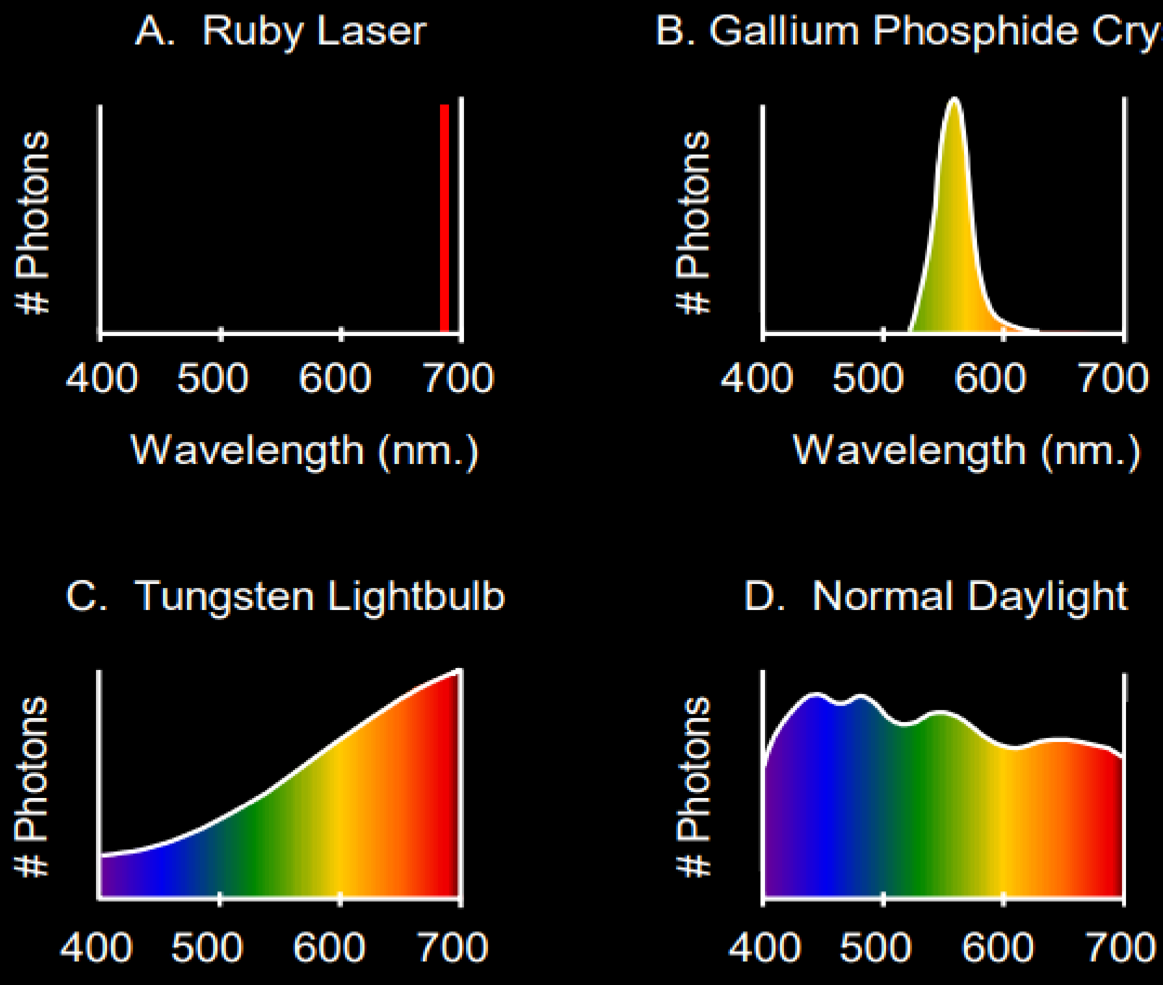
Any patch of light can be completely described physically by its spectrum: the number of photons (per time unit) at each wavelength 400 - 700 nm.



© Stephen E. Palmer, 2002

The Physics of Light

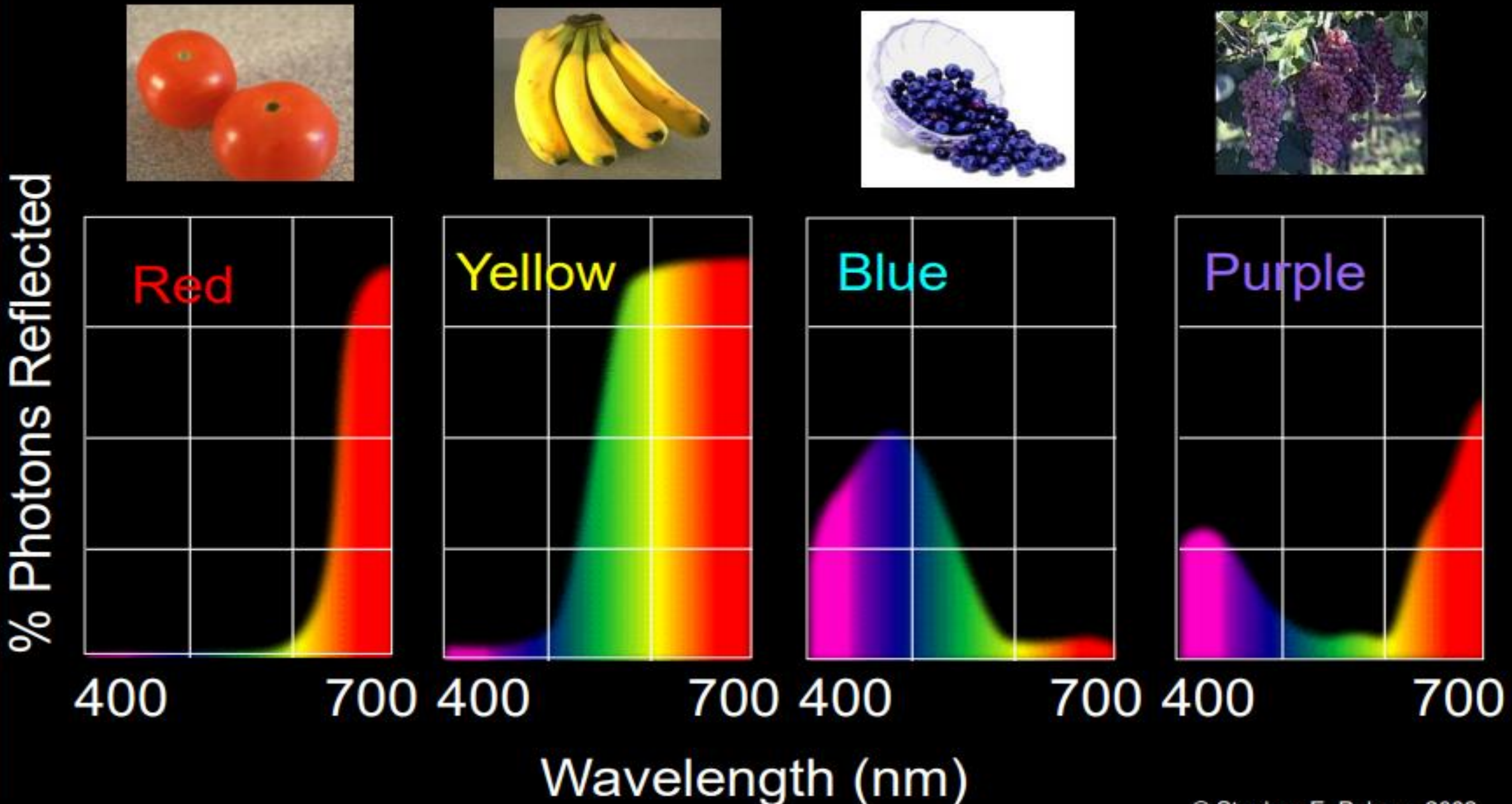
Some examples of the spectra of light sources



© Stephen E. Palmer, 2002

The Physics of Light

Some examples of the reflectance spectra of surfaces

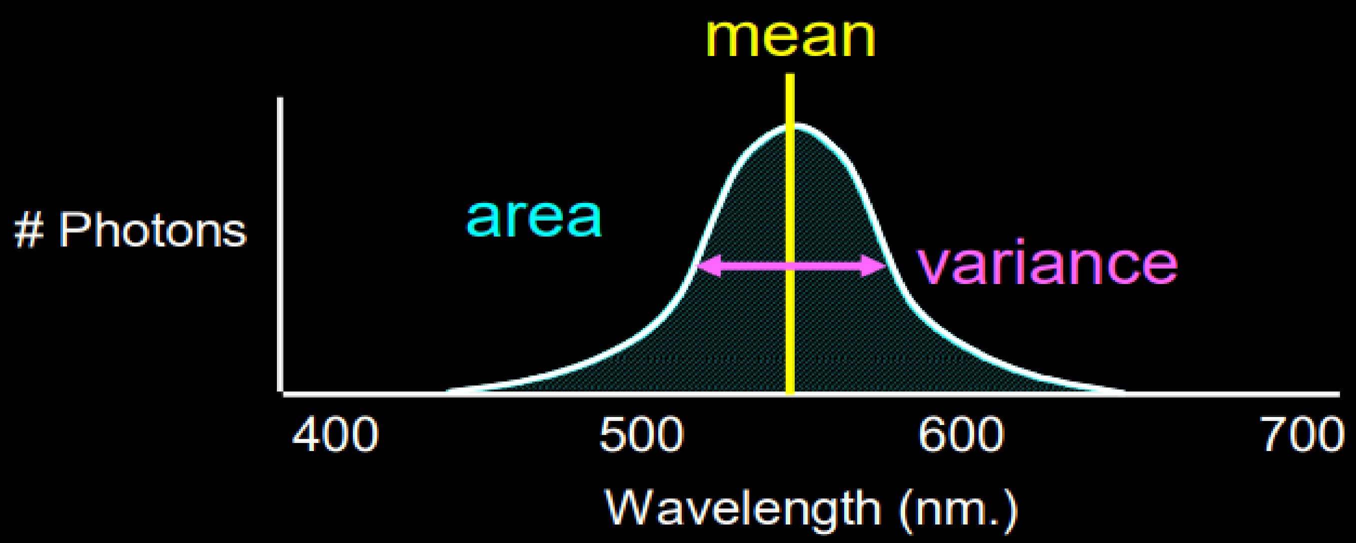


© Stephen E. Palmer, 2002

The Psychophysical Correspondence

There is no simple functional description for the perceived color of all lights under all viewing conditions, but ……

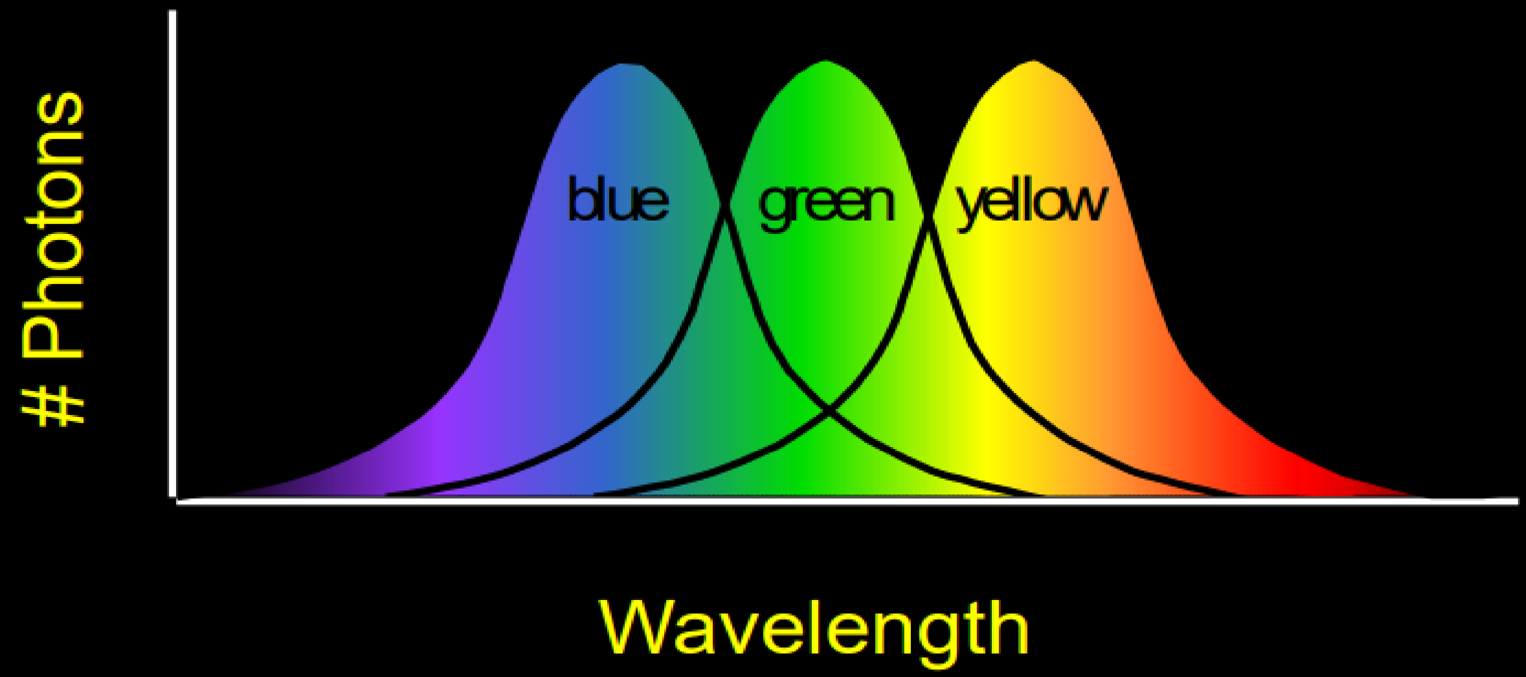
A helpful constraint:
Consider only physical spectra with normal distributions



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

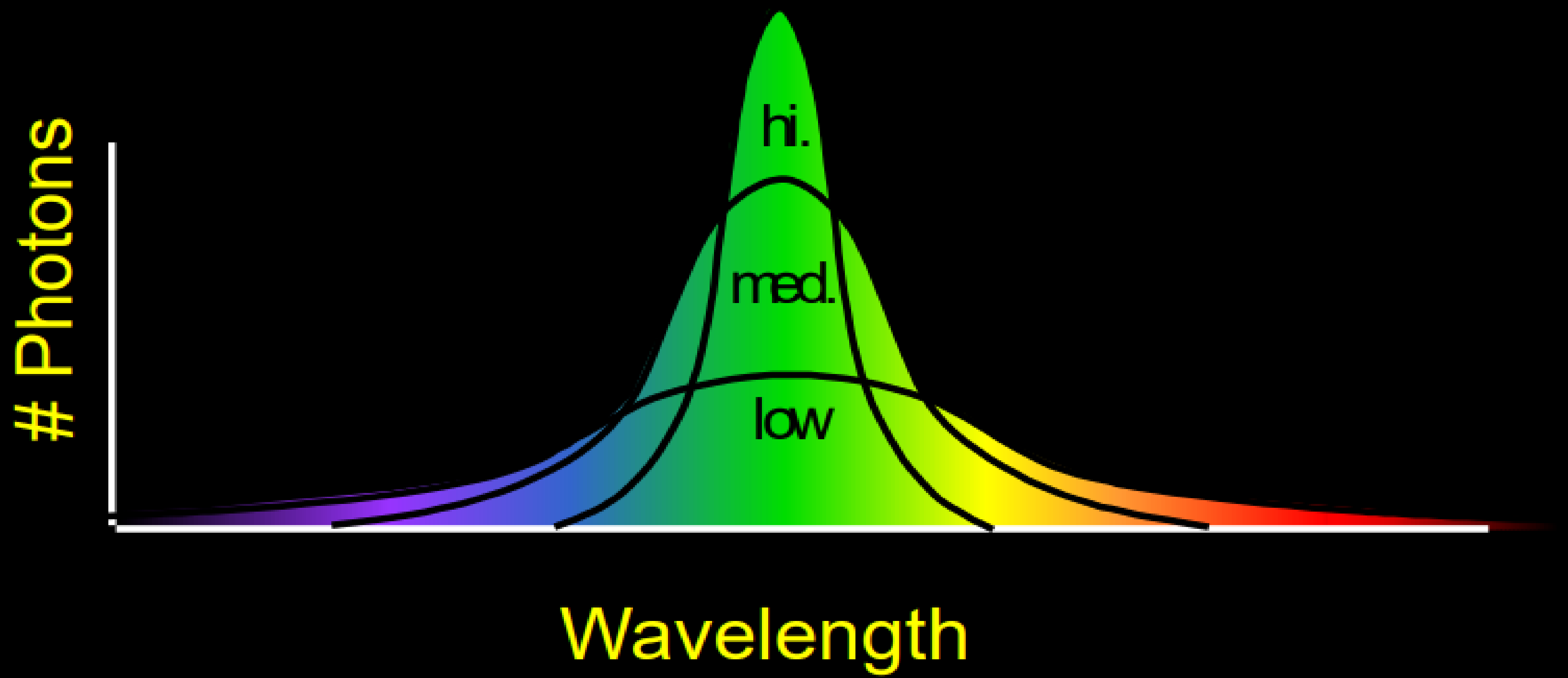
Mean ⟷ Hue



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

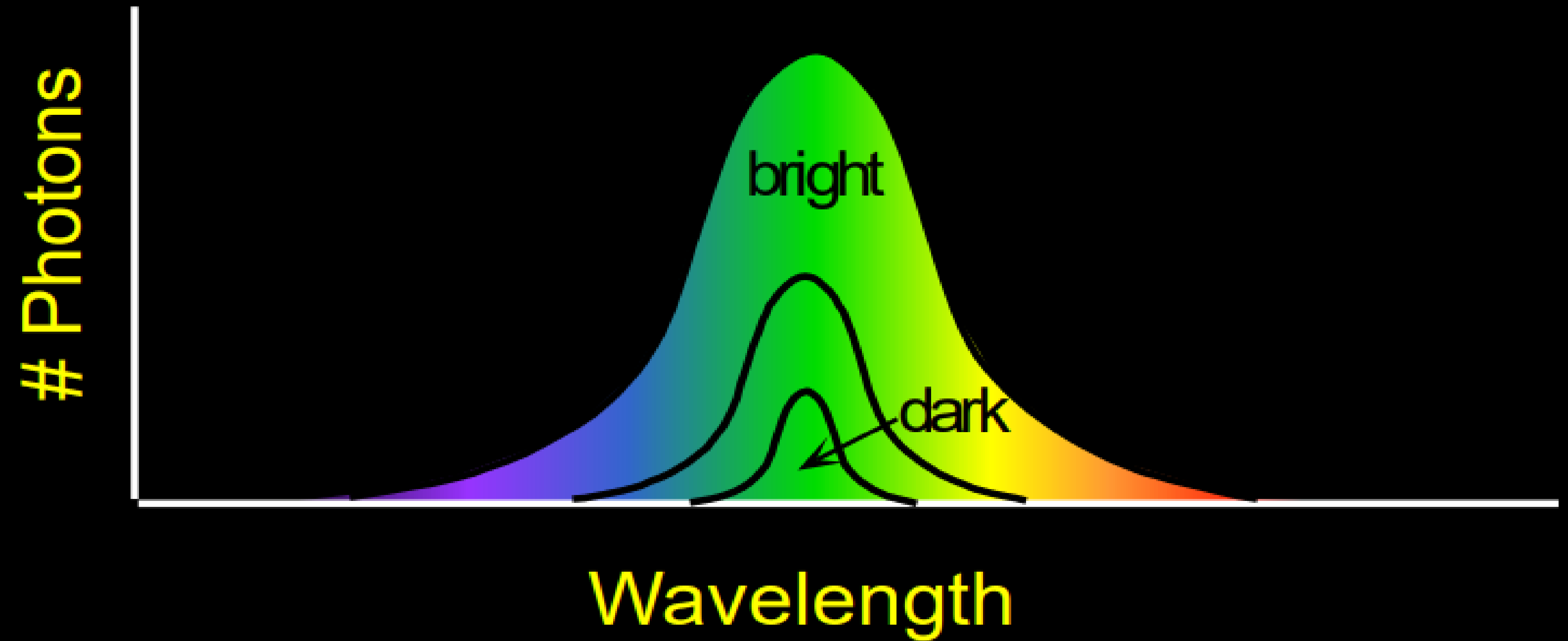
Variance ⟷ Saturation



© Stephen E. Palmer, 2002

The Psychophysical Correspondence

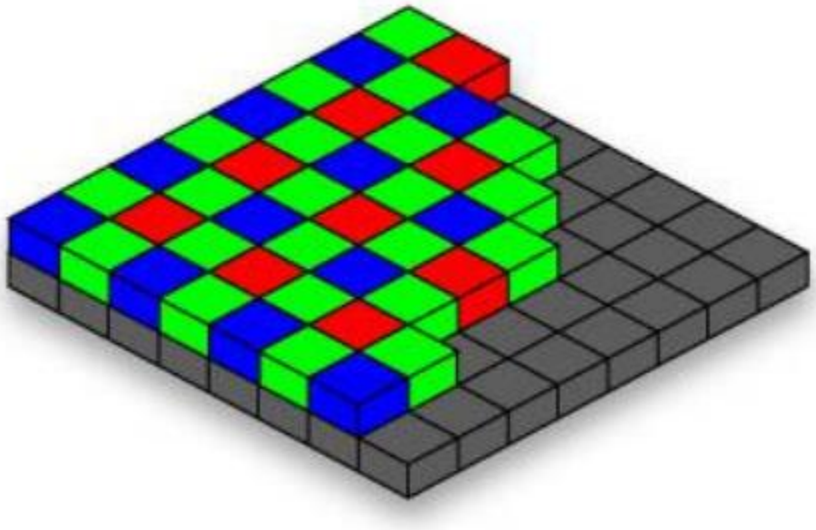
Area ⟷ Brightness

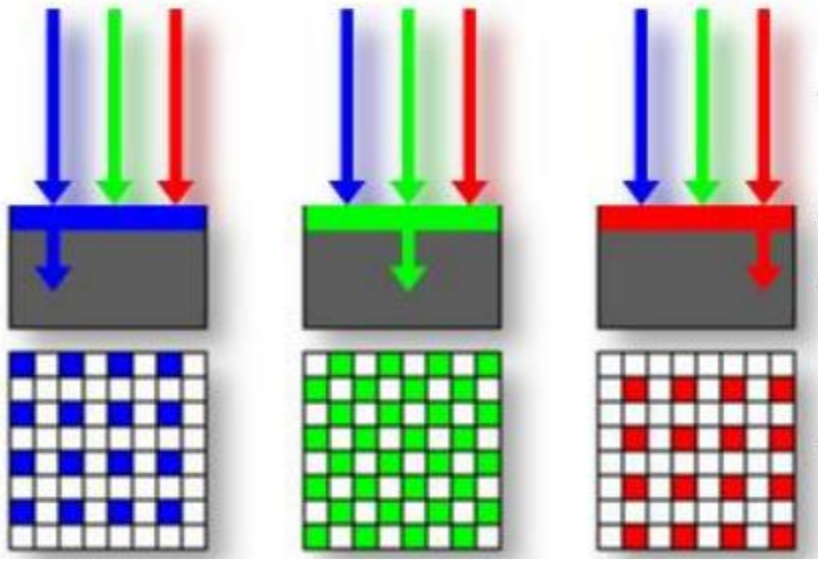


© Stephen E. Palmer, 2002

Practical Color Sensing: Bayer Grid

Estimate RGB at 'G' cells from neighboring values





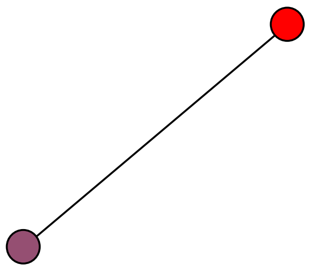
https://en.wikipedia.org/wiki/Bayer_filter

Slide by Steve Seitz

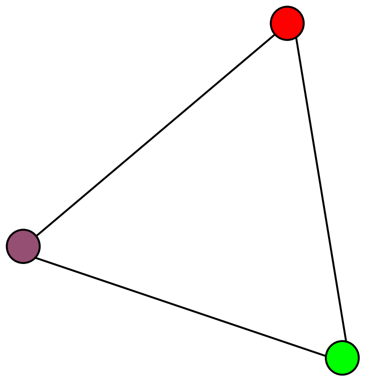
Dr. Sander Ali Khowaja

Linear Color Spaces

- Defined by a choice of three *primaries*
- The coordinates of a color are given by the weights of the primaries used to match it



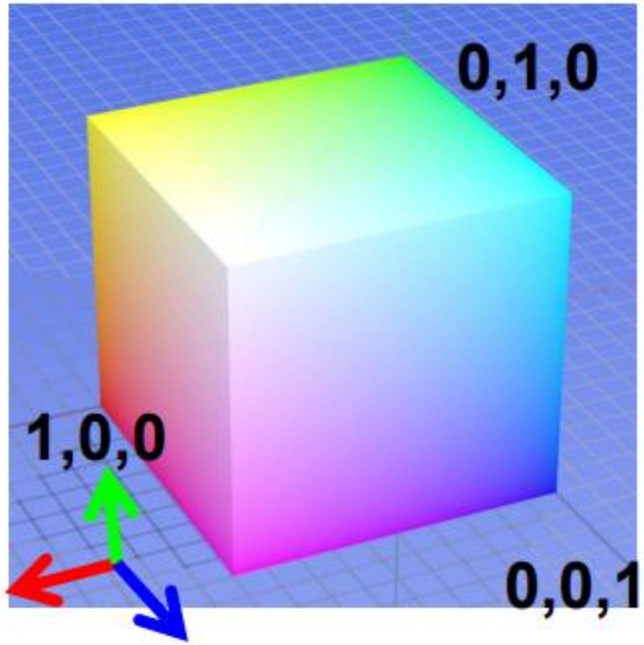
mixing two lights produces colors that lie along a straight line in color space



mixing three lights produces colors that lie within the triangle they define in color space

RGB Color Space

Default color space



lights (for mon
of phosphors)

RGB prima

R
(G=0,B=0)

G
(R=0,B=0)

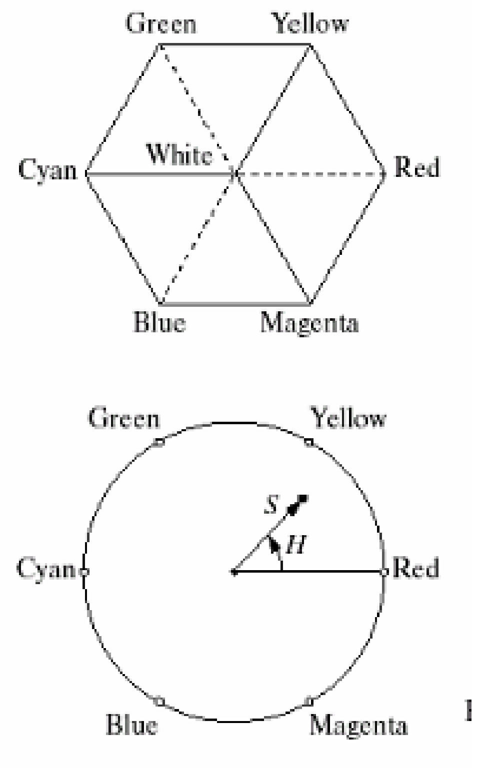
B
(R=0,G=0)

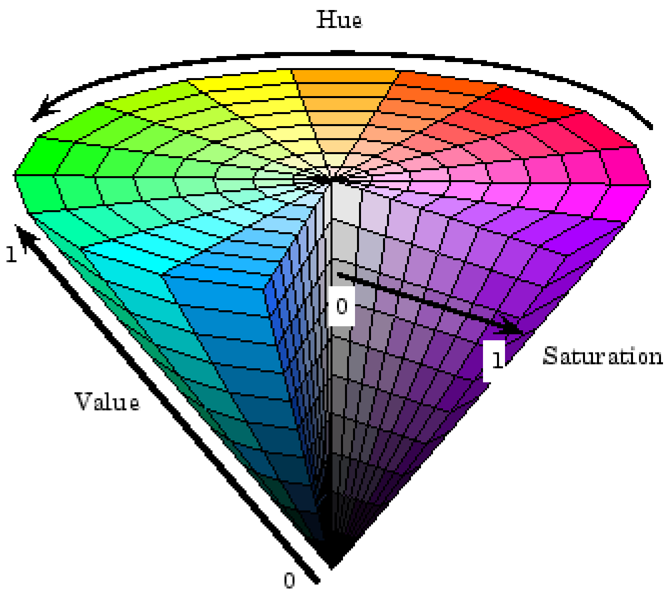
RGB cube

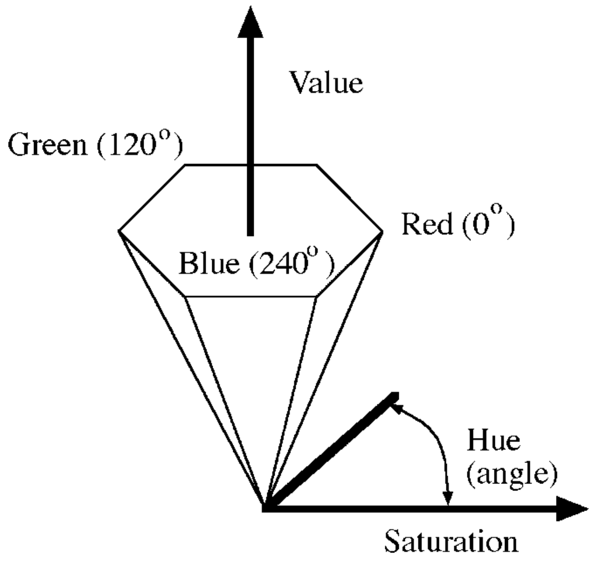
- Easy for devices
- But not perceptual
- Where do the grays live?
- Where is hue and saturation?

Color Image

Non-Linear Color Spaces: HSV







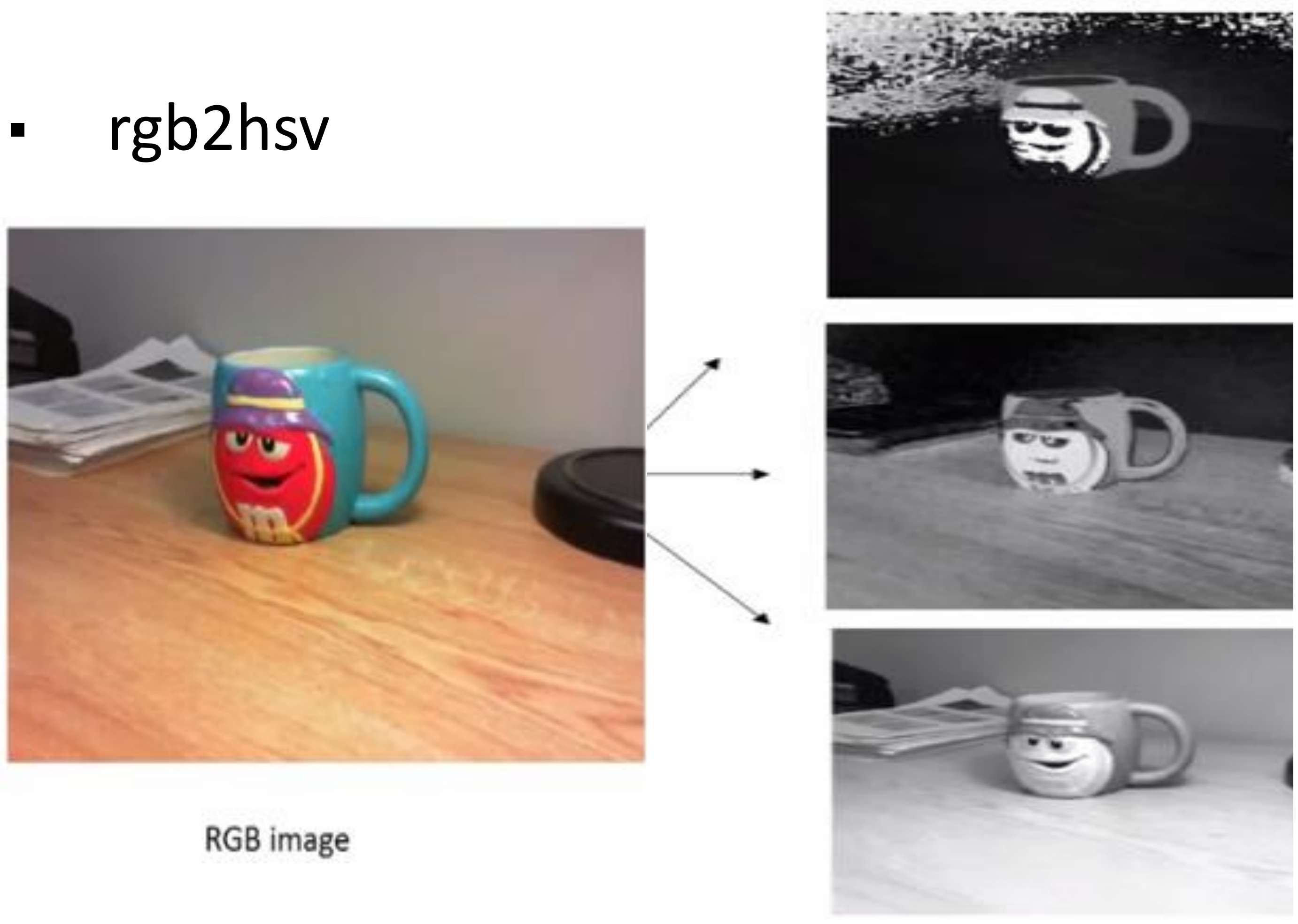
- Perceptually meaningful dimensions: Hue, Saturation, Value (Intensity)
- RGB cube on its vertex

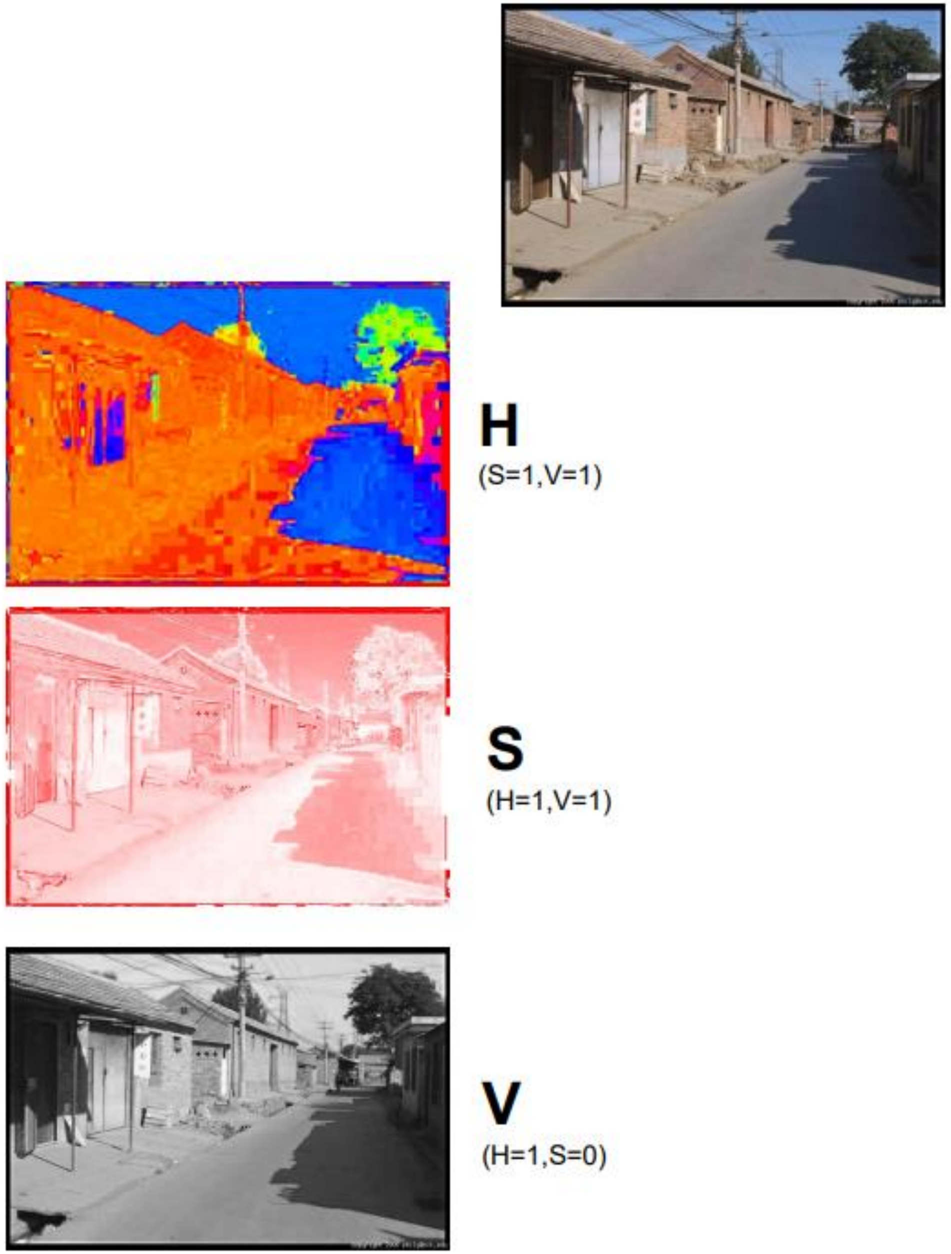
Conversion from RGB to HSV

- $V = \frac{1}{3}(R + G + B)$
- $S = 1 - \frac{3}{R+G+B}\min(R, G, B)$
- $H = \begin{cases} \theta & B \leq G \\ 360 - \theta & B > G \end{cases}$
- Where
- $Cos\ \theta = \frac{1/2[(R-G)+(R+B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}}$

MATLAB Example

- rgb2hsv

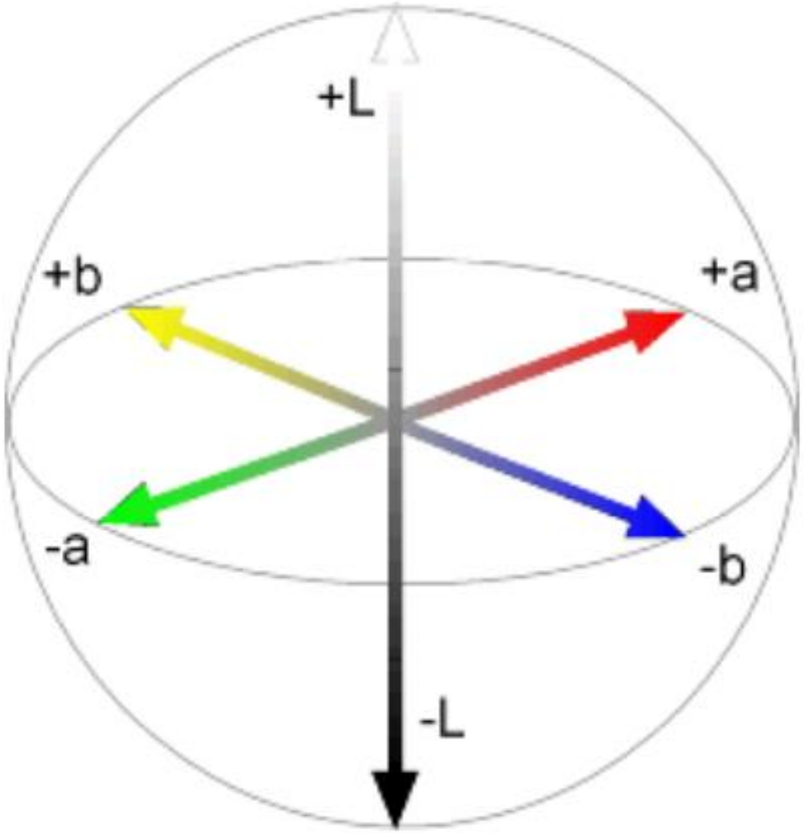


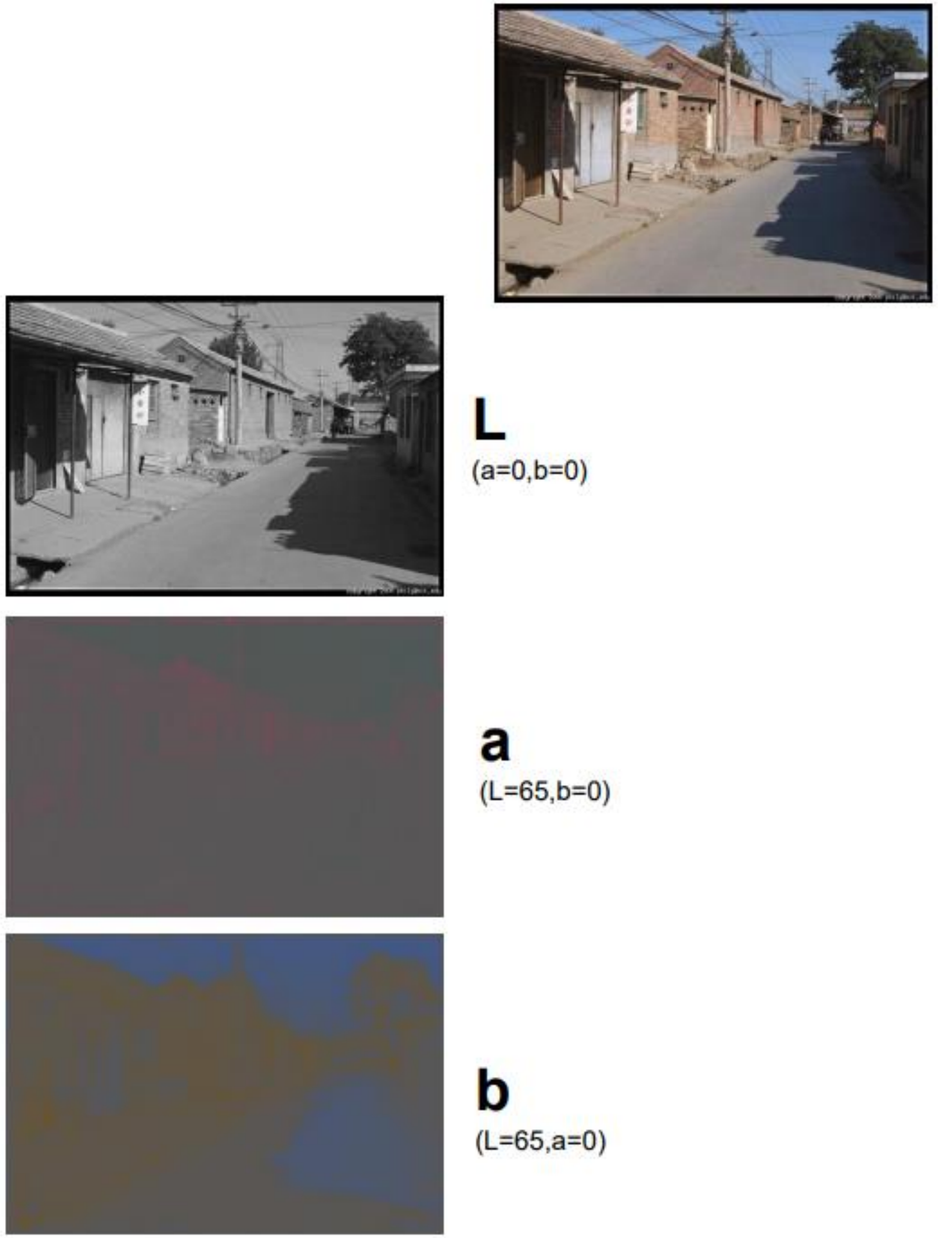


Use rgb2hsv() and hsv2rgb() in Matlab

Color Spaces: L*a*b

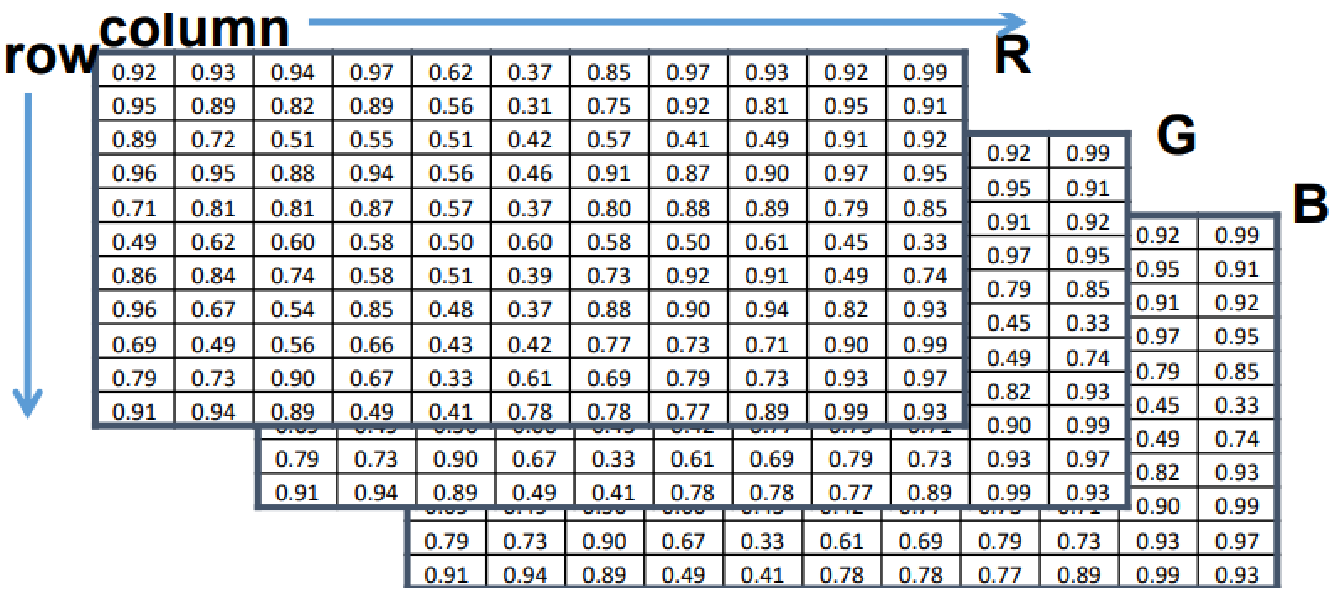
"Perceptually uniform" color space



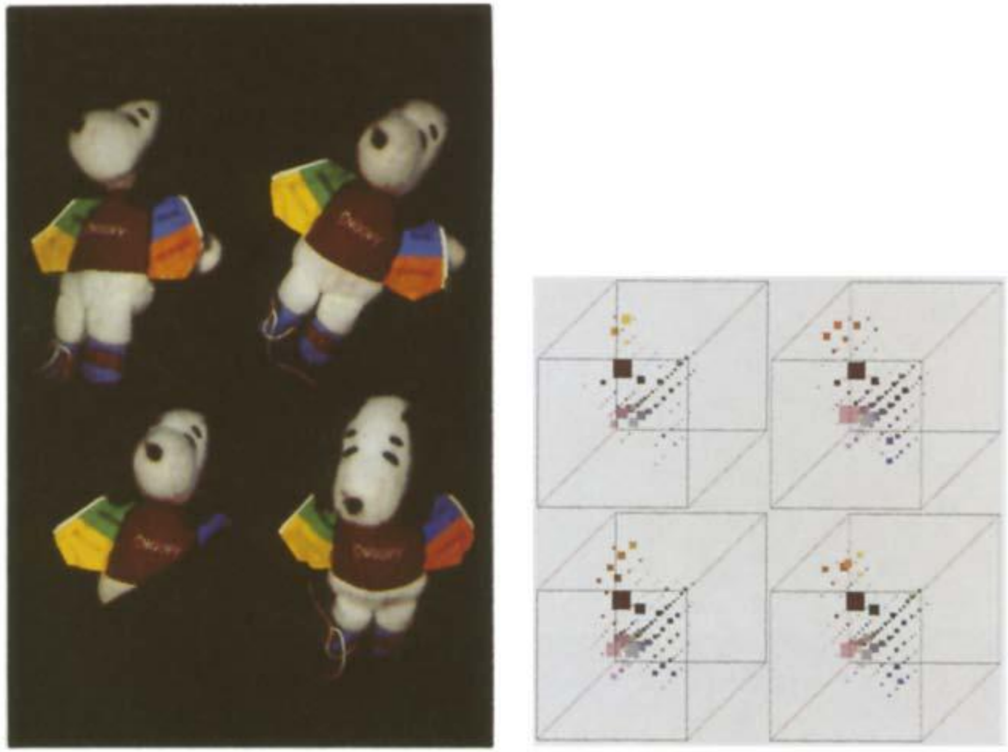


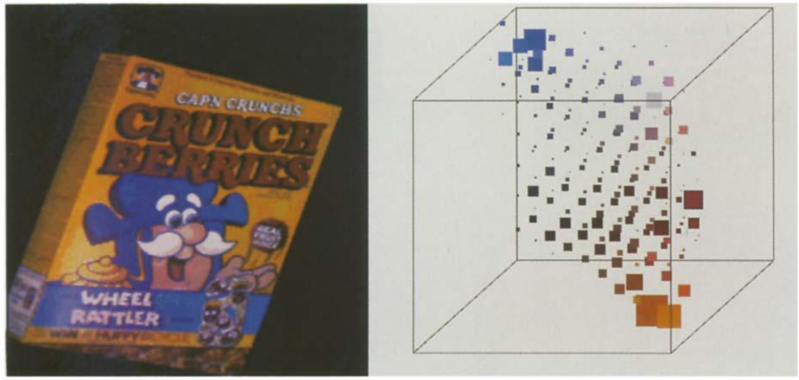
Images in MATLAB

- Images represented as a matrix
- Suppose we have a NxM RGB image called “im”
  - Im(1,1,1) = top-left pixel value in R-channel
  - Im(y,x,b) = y pixels down, x pixels to right in the b-th channel
  - Im(N,M,3) = bottom-right pixel in B-channel
- Imread(filename) returns a unit8 image (values 0 to 255)
  - Convert to double format (values 0 – 1) with im2double



Color histograms for indexing and retrieval

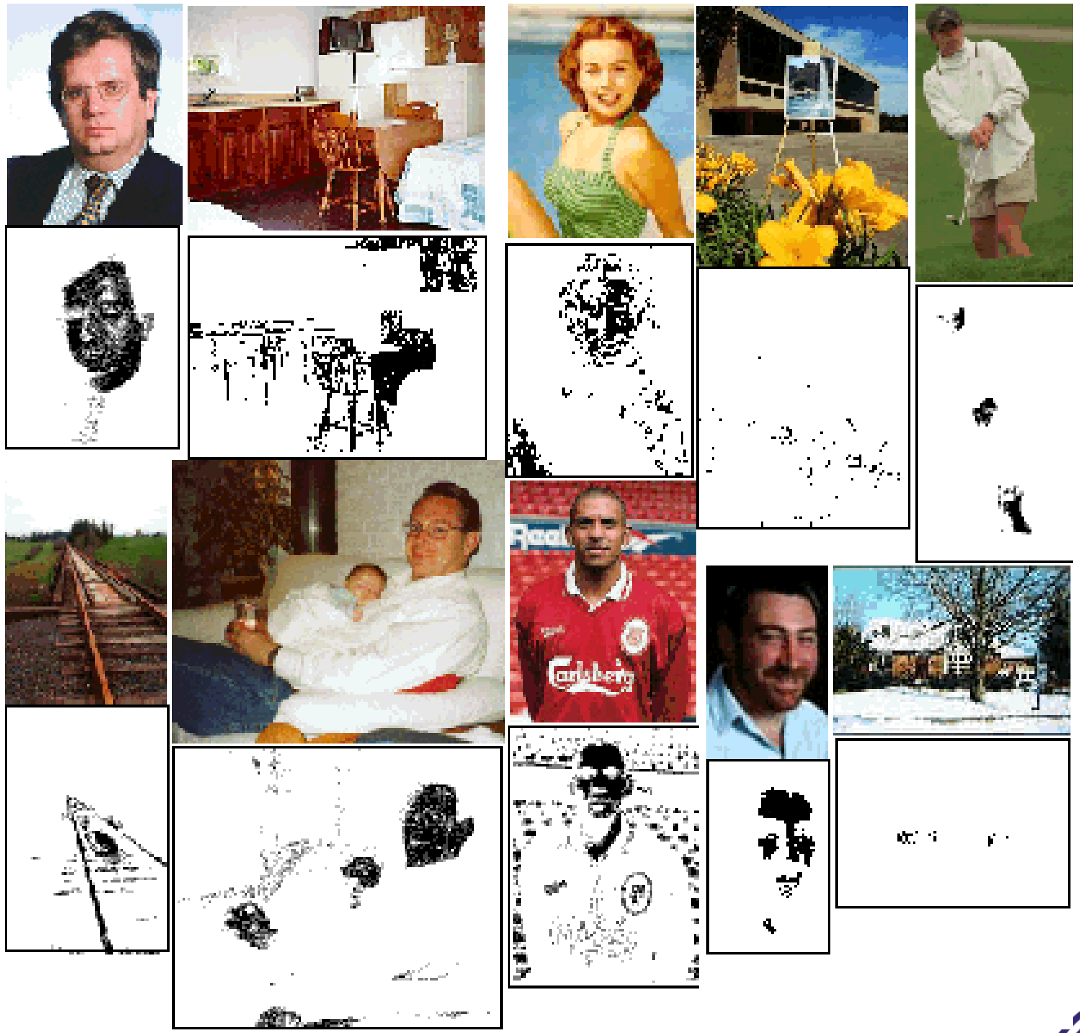




Swain and Ballard, Color Indexing, IJCV 1991.

Uses of Color in Computer Vision

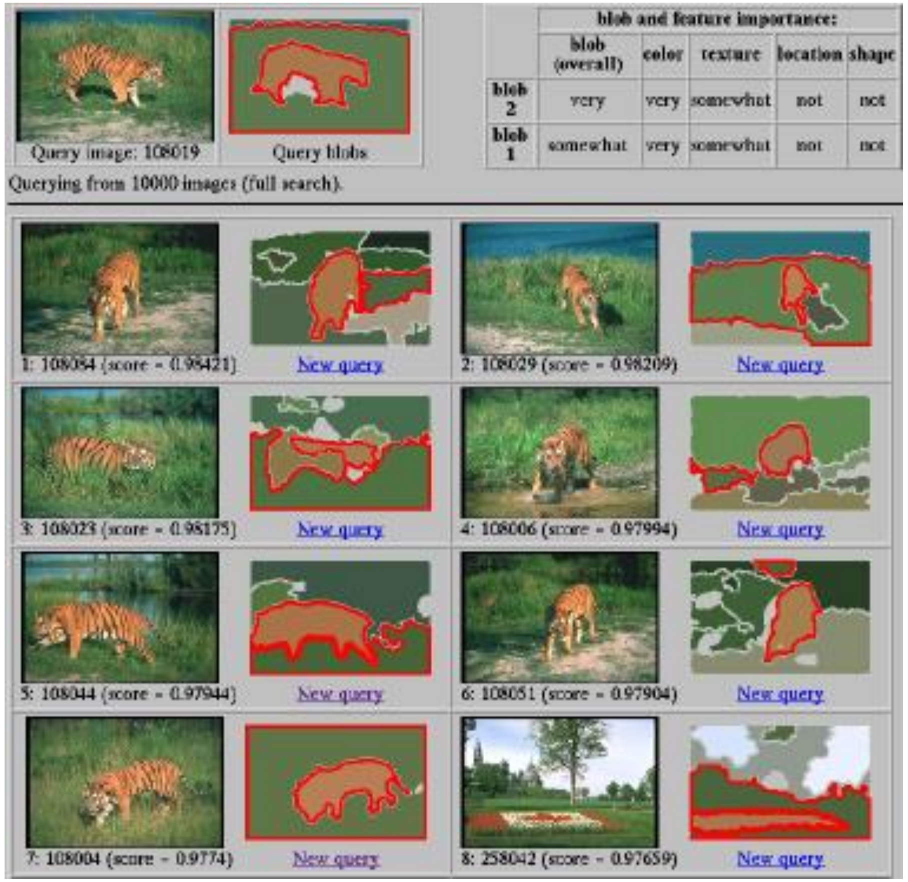
Skin detection



M. Jones and J. Rehg, [Statistical Color Models with Application to Skin Detection](), IJCV 2002.

Uses of Color in Computer Vision

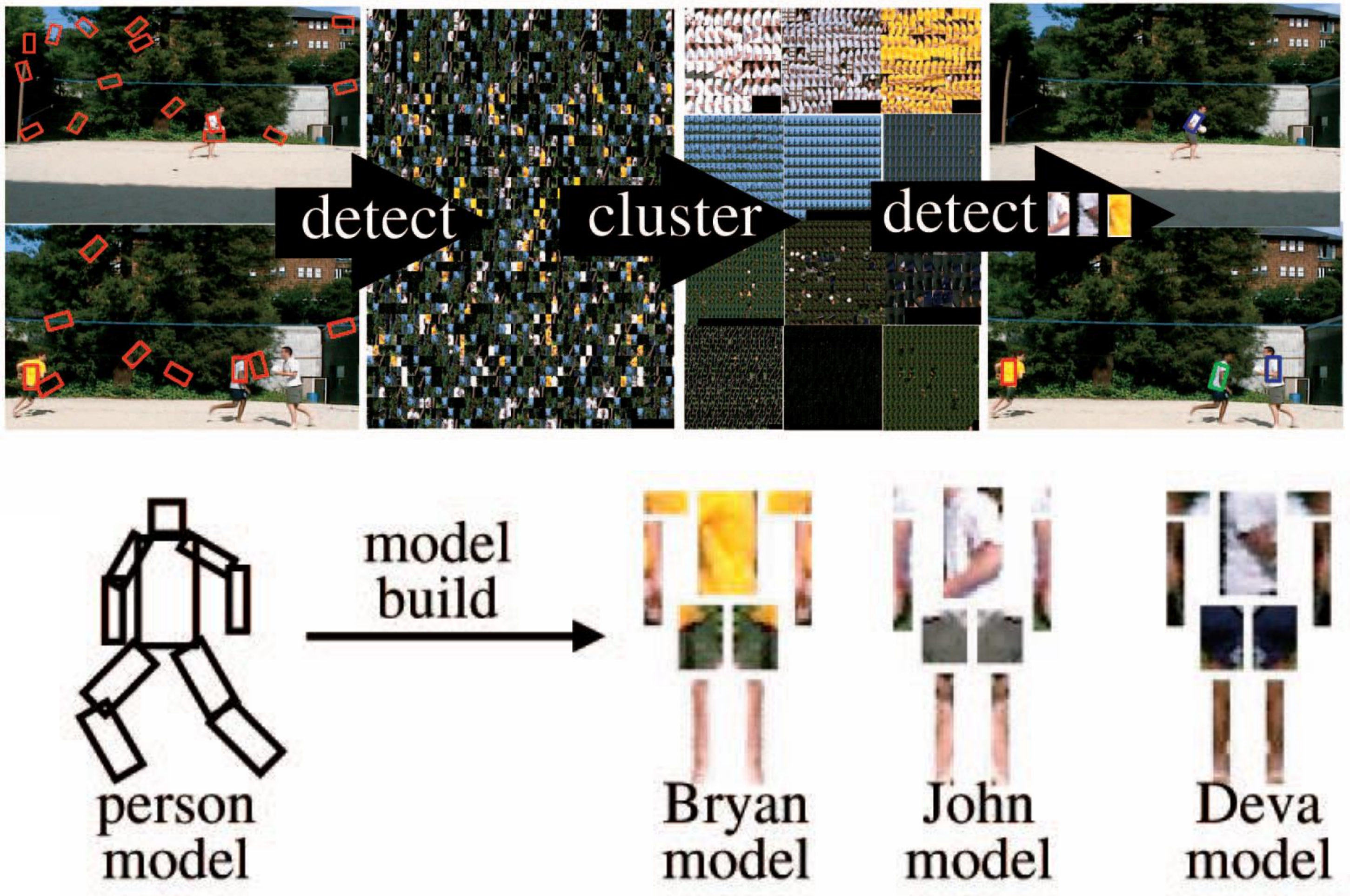
Image segmentation and retrieval



C. Carson, S. Belongie, H. Greenspan, and Ji. Malik, Blobworld: Image segmentation using Expectation-Maximization and its application to image querying, ICVIS 1999.

Uses of Color in Computer Vision

Building appearance models for tracking



D. Ramanan, D. Forsyth, and A. Zisserman. [Tracking People by Learning their Appearance](). PAMI 2007.

Common Color Representations

- Black: (0,0,0)
- White: (255,255,255)
- Red: (255,0,0)
- Green: (0,255,0)
- Blue: (0,0,255)
- Aqua: (0,255,255)
- Fuchsia: (255,0,255)
- Maroon: (128,0,0)
- Navy: (0,0,128)
- Olive: (128,128,0)
- Purple: (128,0,128)
- Teal: (0, 128,128)
- Yellow: (255,255,0)

Use the following code in Python/OpenCV

- Image = cv2.imread(“image path”)
- (b,g,r) = image[0, 0]
- Print(“Pixel at (0,0) – Red: {}, Green: {}, Blue: {}”.format(r, g, b))
- Lets manipulate the pixel value
- Image[0,0] = (0,0,255)
- (b,g,r) = image[0,0]

For Color Spaces

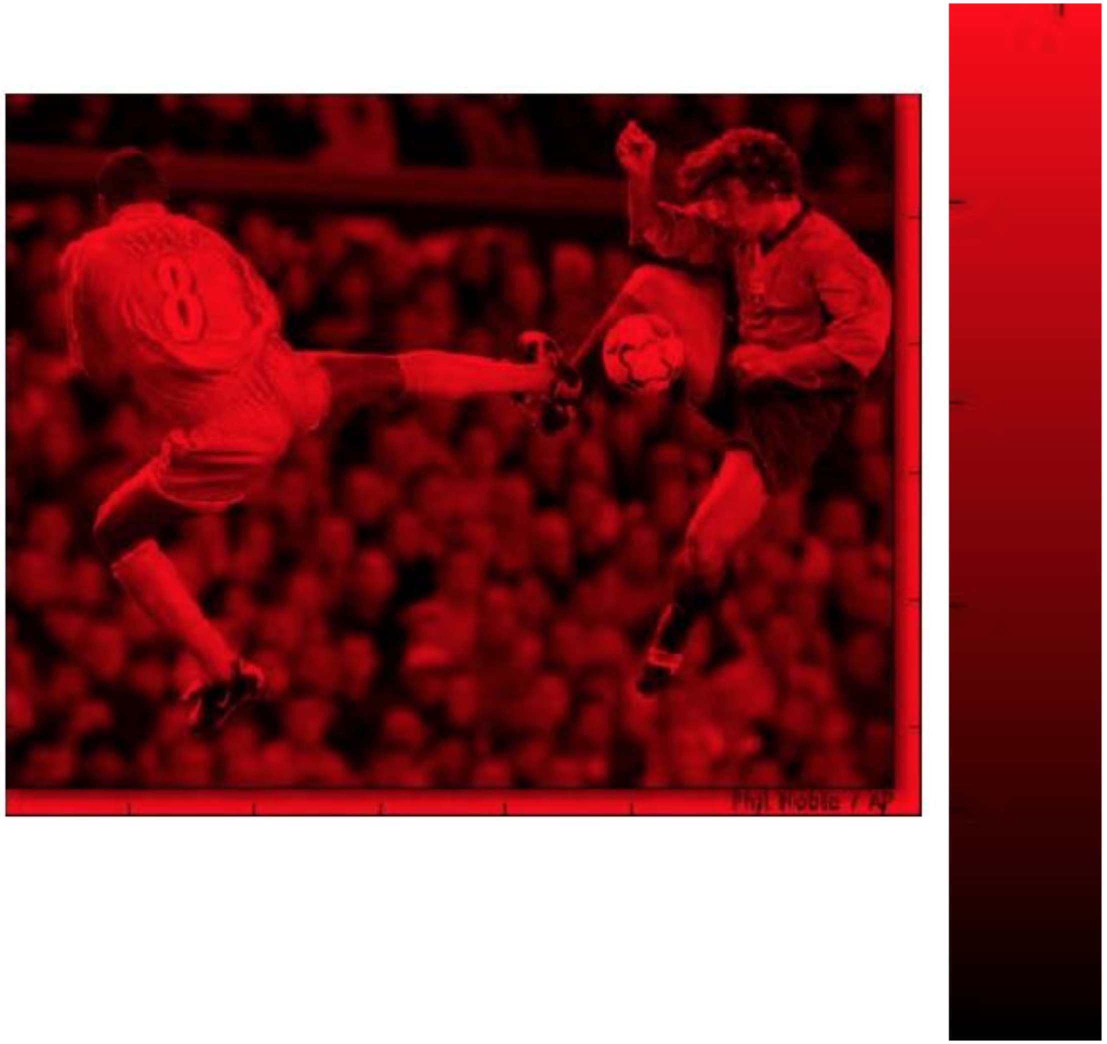
- Image= cv2.imread(“image path”)
- cv2.imshow(“Original”, Image)

- Gray = cv2.cvtColor(Image, cv2.COLOR_BGR2GRAY)
- cv2.imshow(“Gray”,Gray)

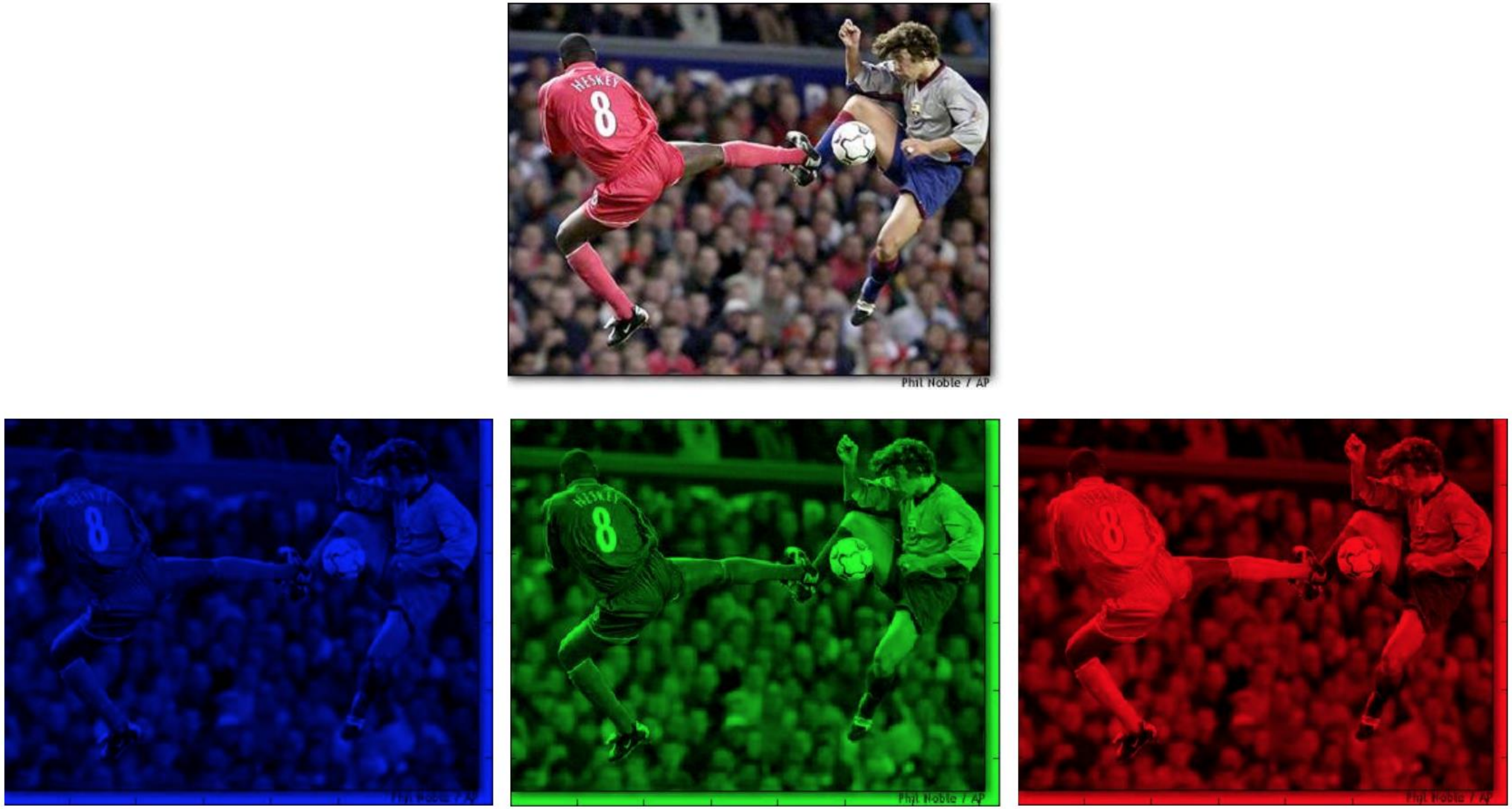
- Hsv = cv2.cvtColor(Image, cv2.COLOR_BGR2HSV)
- cv2.imshow(“HSV”,Hsv)

- Lab = cv2.cvtColor(Image, cv2.COLOR_BGR2LAB)
- cv2.imshow(“L*a*b*”, Lab)
- cv2.waitKey(0)

Color Image – One Channel

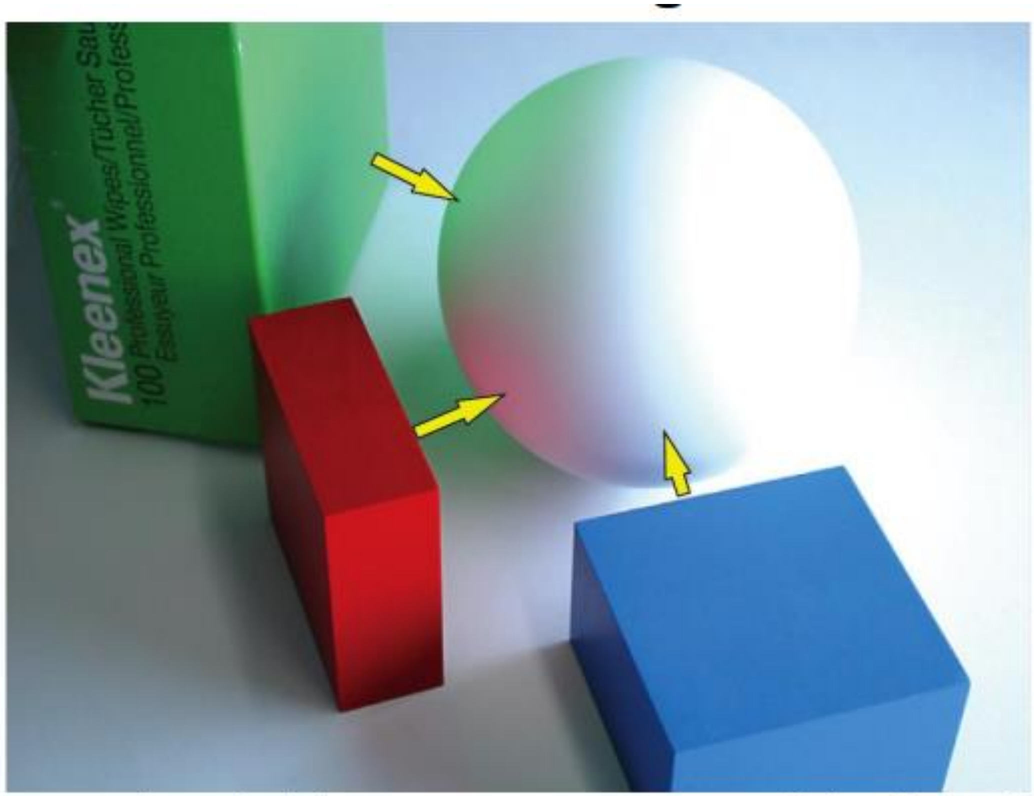


Color Image Representation



So far: Light → Surface → Camera

- Called a local illumination model
- But much light comes from surrounding surfaces



From Koenderink slides on image texture and the flow of light

Questions

A. Why is (2) brighter than (1)? Each points to the asphalt.
B. Why is (4) darker than (3)? (4) points to the marking.
C. Why is (5) brighter than (3)? Each points to the side of the wooden block.
D. Why isn't (6) black, given that there is no direct path from it to the sun?
E. Why (7) brighter than (8)? Both point to the yellow paints.
F. Why is (9) green, given that the sun light contains all visible wavelengths?

What does the intensity of a pixel tell us?

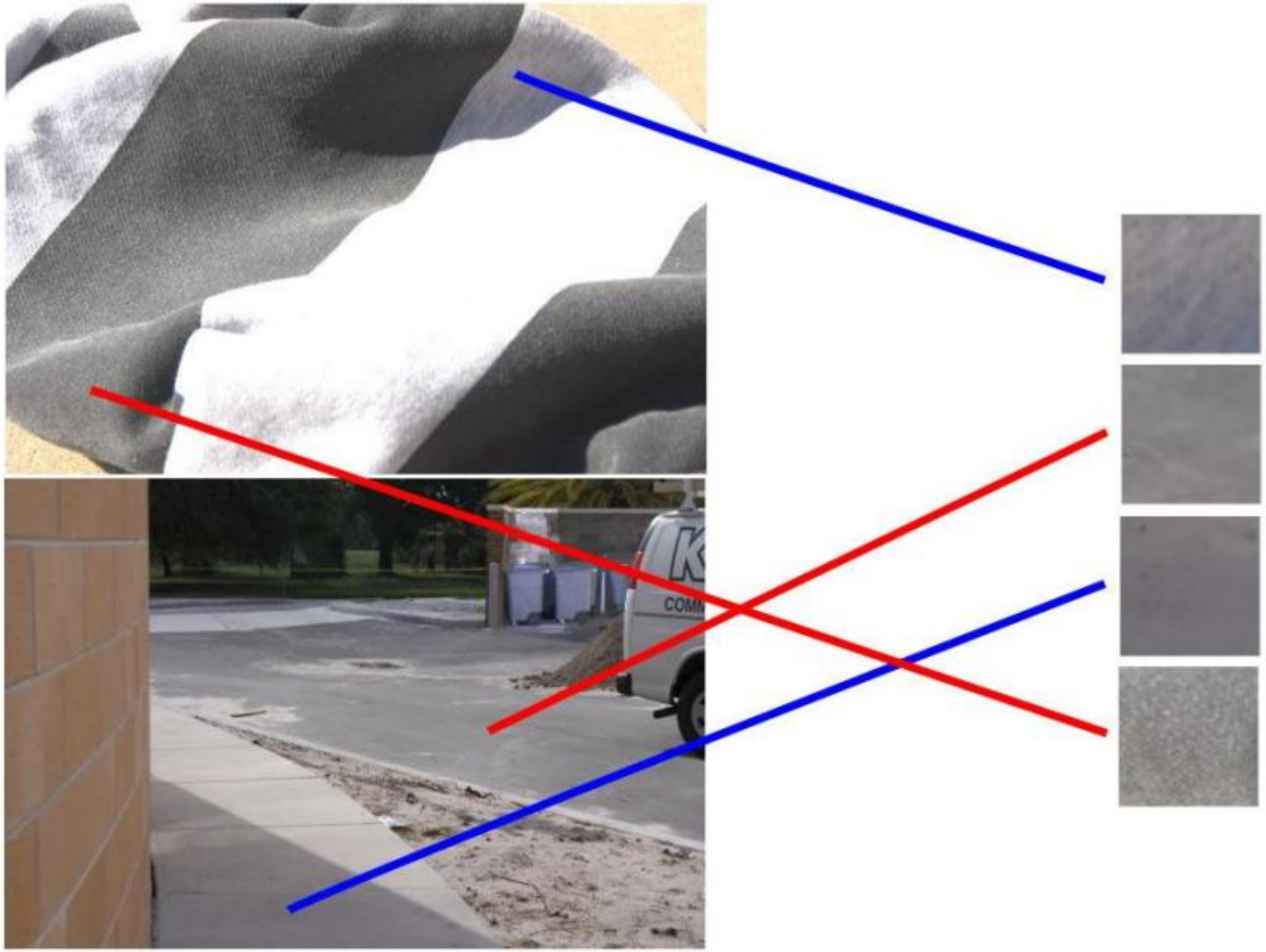
im(234, 452) = 0.58

0.92	0.93	0.94	0.97	0.62	0.37	0.85	0.97	0.93	0.92	0.99
0.95	0.89	0.82	0.89	0.56	0.31	0.75	0.92	0.81	0.95	0.91
0.89	0.72	0.51	0.55	0.51	0.42	0.57	0.41	0.49	0.91	0.92
0.96	0.95	0.88	0.94	0.56	0.46	0.91	0.87	0.90	0.97	0.95
0.71	0.81	0.81	0.87	0.57	0.37	0.80	0.88	0.89	0.79	0.85
0.49	0.62	0.60	0.58	0.50	0.60	0.58	0.50	0.61	0.45	0.33
0.86	0.84	0.74	0.58	0.51	0.39	0.73	0.92	0.91	0.49	0.74
0.96	0.67	0.54	0.85	0.48	0.37	0.88	0.90	0.94	0.82	0.93
0.69	0.49	0.56	0.66	0.43	0.42	0.77	0.73	0.71	0.90	0.99
0.79	0.73	0.90	0.67	0.33	0.61	0.69	0.79	0.73	0.93	0.97
0.91	0.94	0.89	0.49	0.41	0.78	0.78	0.77	0.89	0.99	0.93

The Plight of the poor pixel

- A Pixel's brightness is deterr
  - Light source (strength, direct
  - Surface orientation
  - Surface material and albedo
  - Reflected light and shadows
  - Gain on the sensor

- A pixel's brightness tell us n



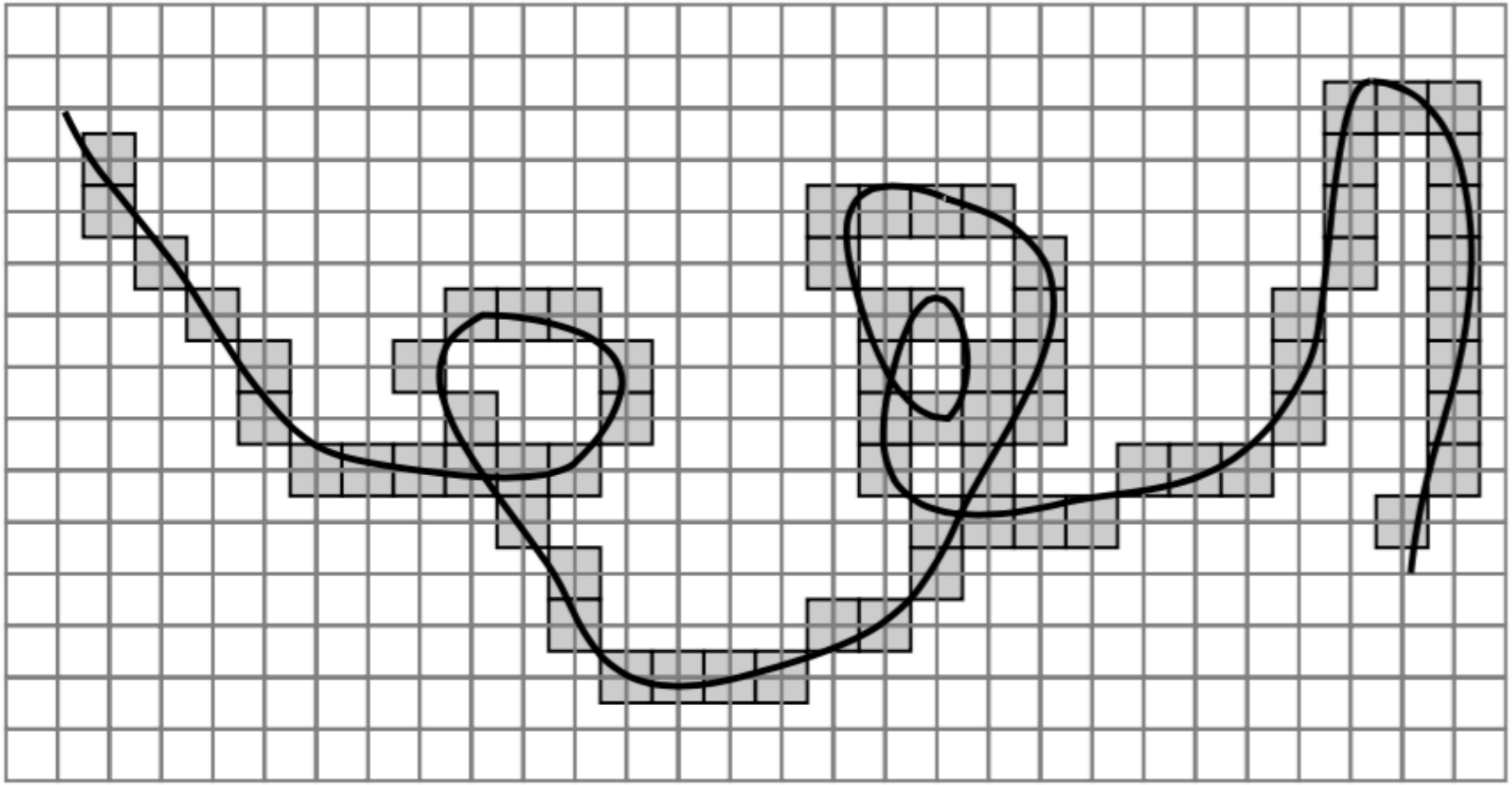
And yet we can interpret images…

- Key idea: for nearby scene points, most factors do not change much
- The information is mainly contained in *local differences* of brightness

Images are Sampled

- What happens when we zoom into the images we capture?

Errors due to sampling



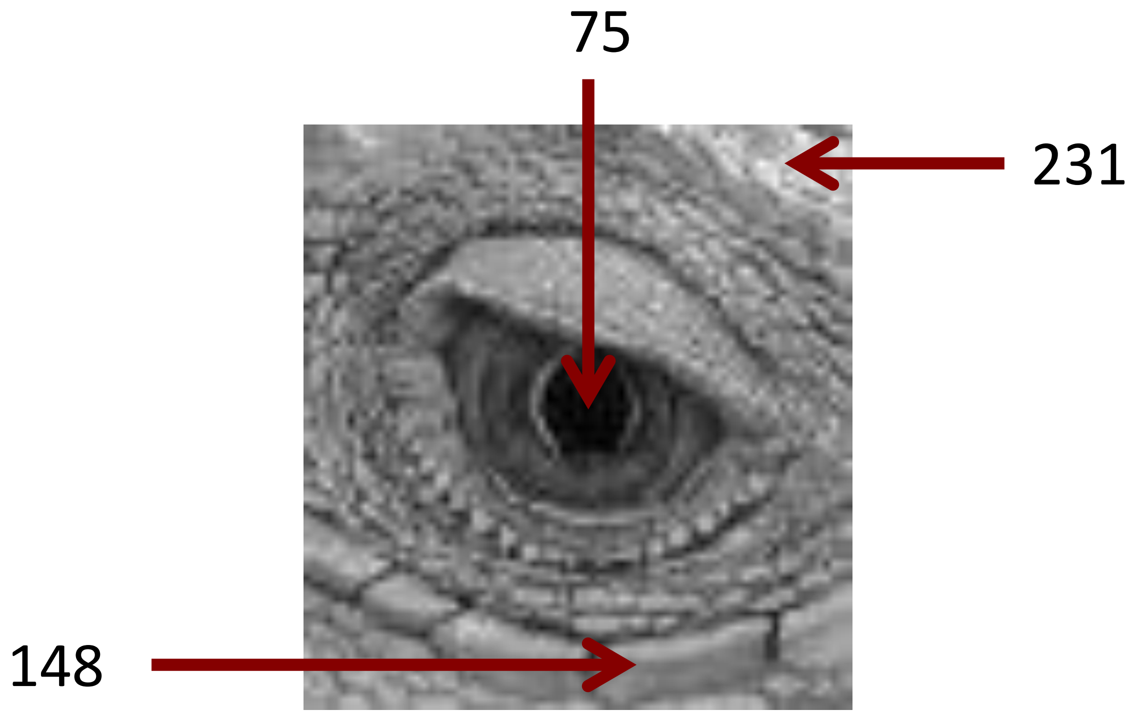
Slide credit: Ulas Bagci

Resolution

- is a **sampling** parameter, defined in dots per inch (DPI) or equivalent measures of spatial pixel density, and its standard value for recent screen technologies is 72 dpi

Images are sampled and quantized

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
    (or "intensity"): [0,255]



Images are Sampled and Quantized

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
    (or "intensity"): [0,255]
    - "color"
      - RGB: [R, G, B]
      - Lab: [L, a, b]
      - HSV: [H, S, V]

Darkness = Large Difference in Neighboring Pixels

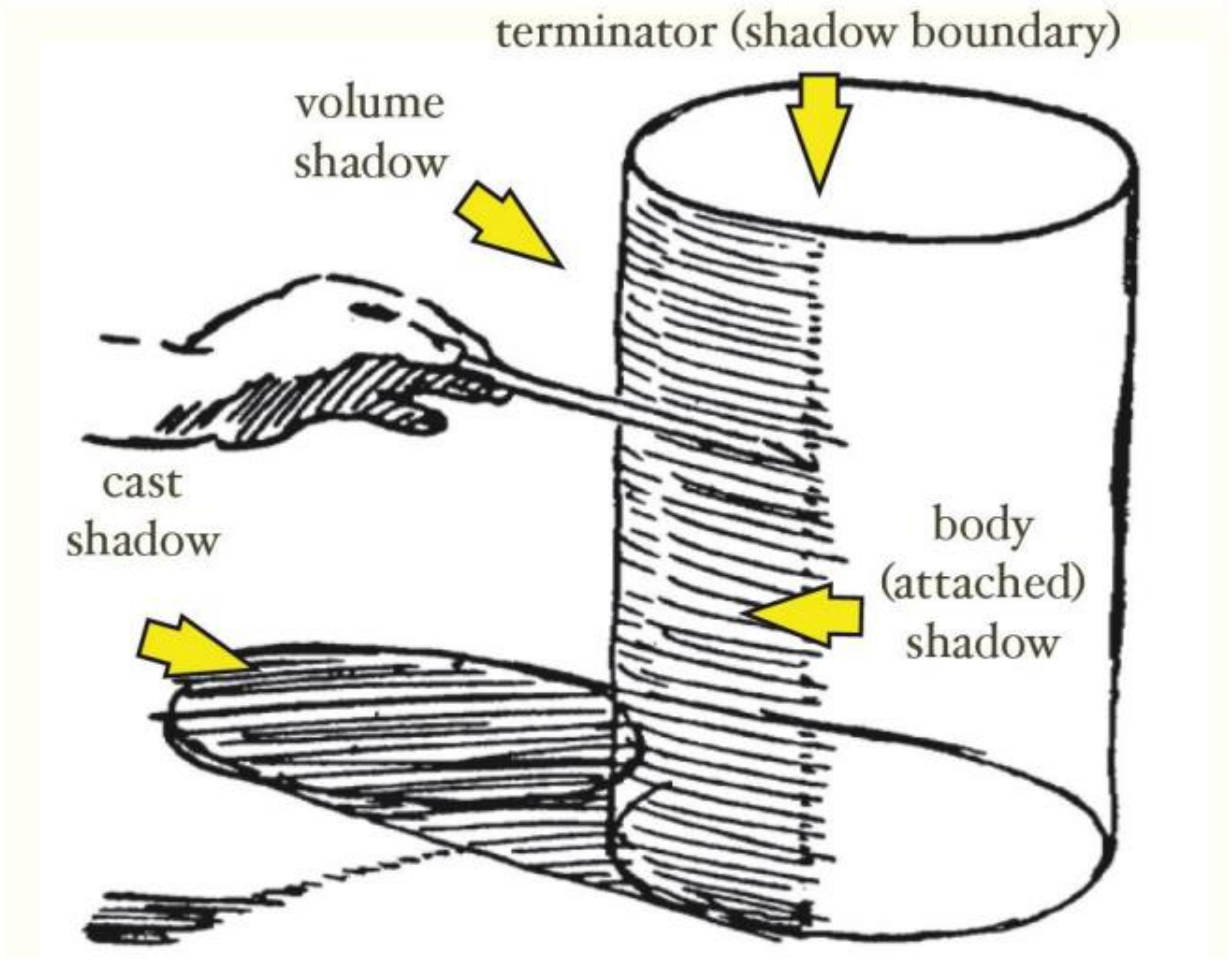
What is this?

What differences in Intensity tell us about

- Changes in surface normal
- Texture
- Proximity
- Indents and bumps
- Grooves and creases

Photos Koenderink slides on image texture and the flow of light

Shadows as cues

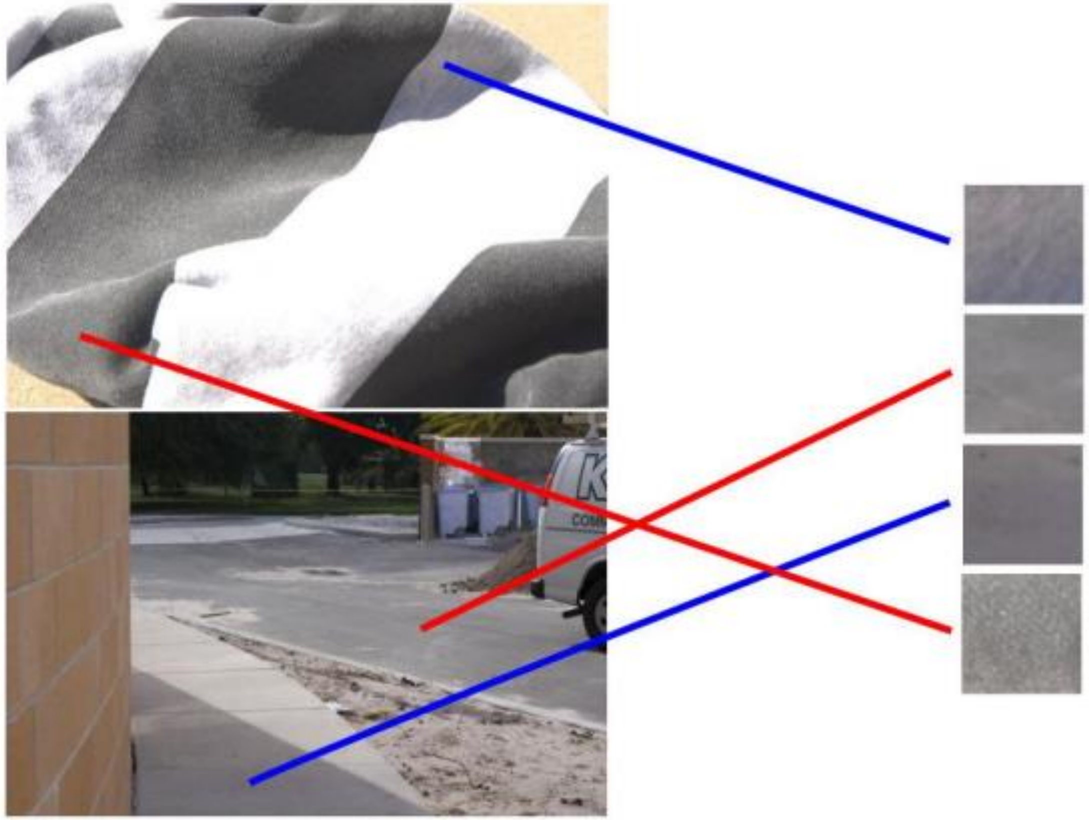


From Koenderink slides on image texture and the flow of light

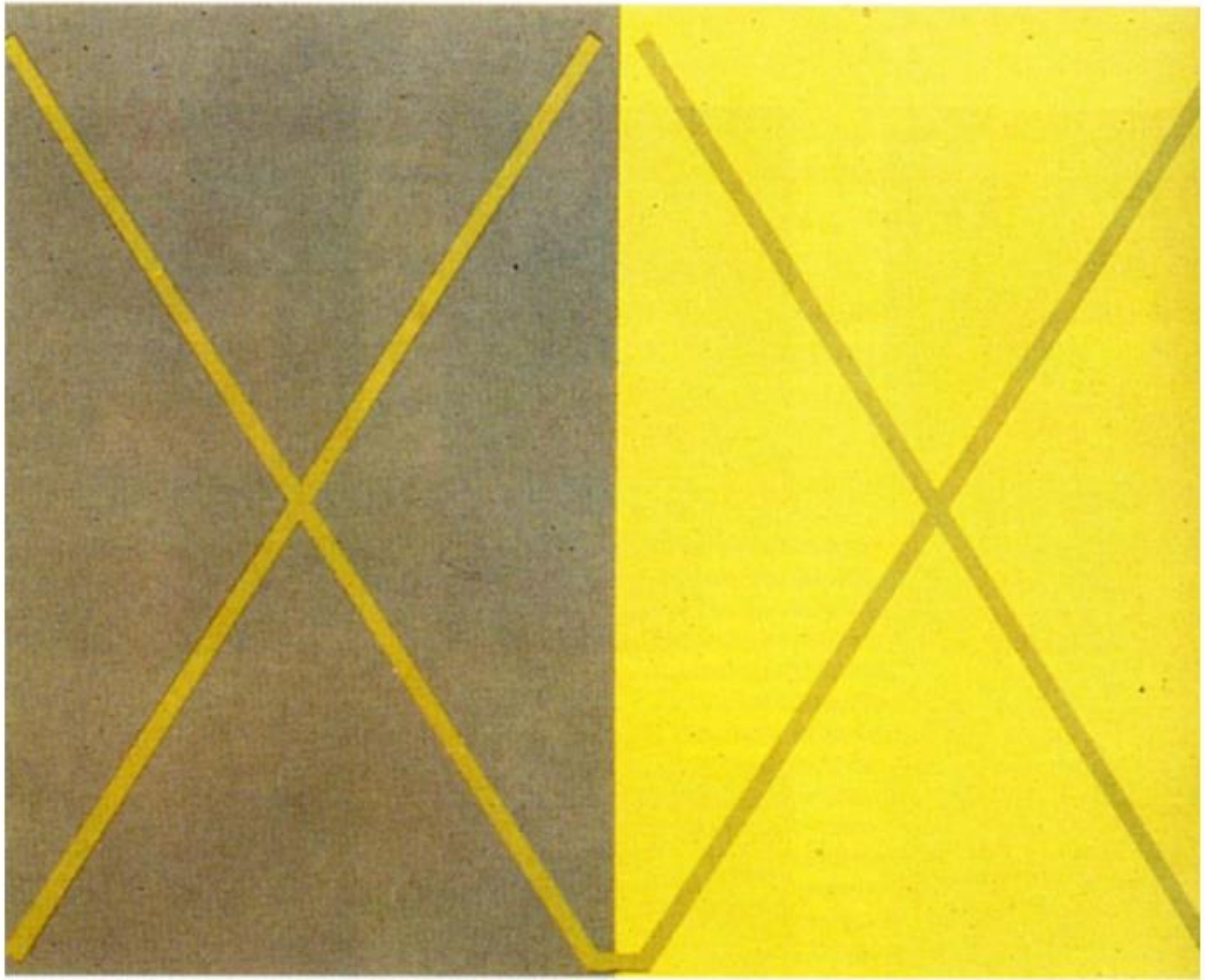
Slide: Forsyth

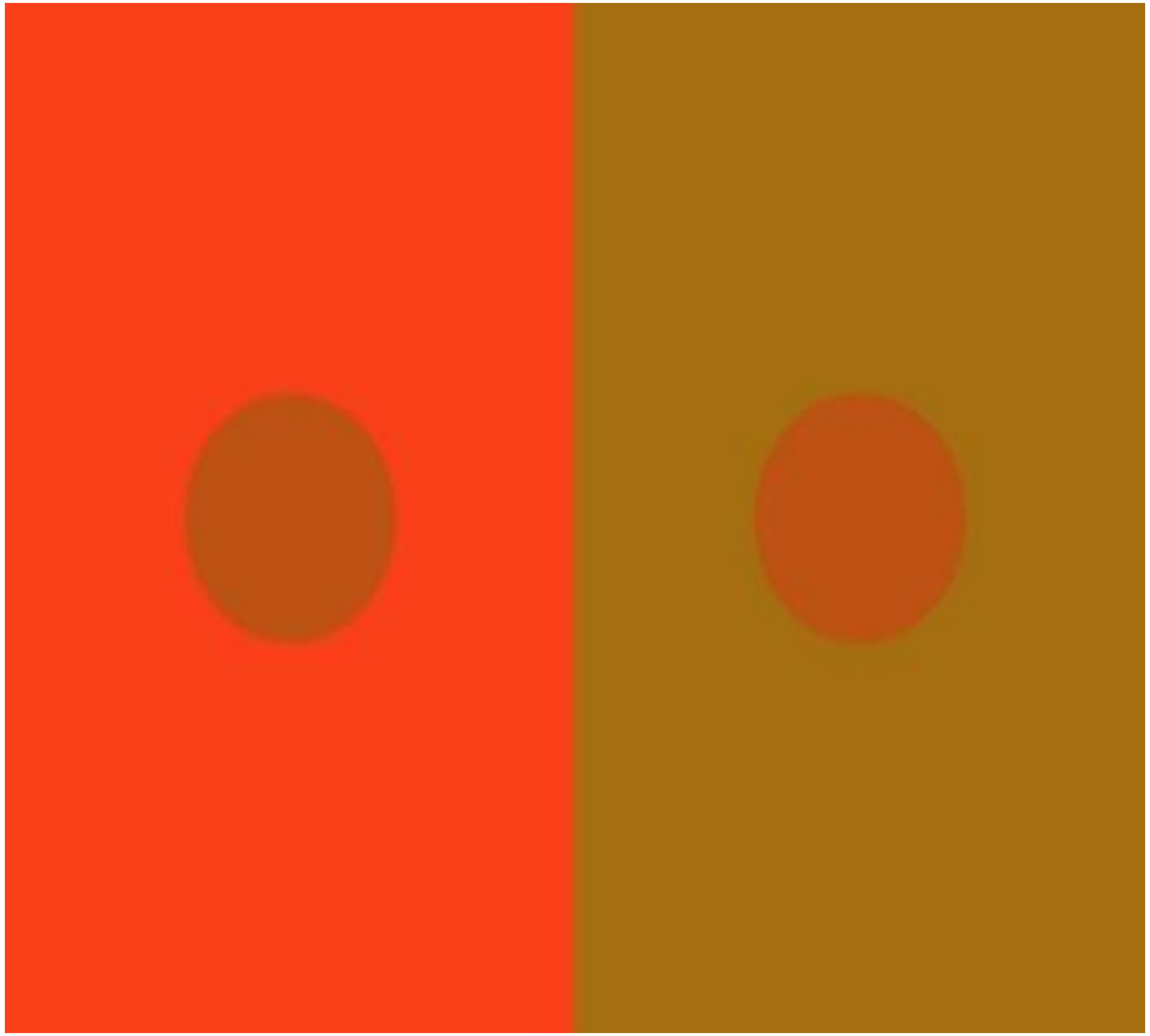
Color Constancy

- Interpret surface in terms of albedo or “true color”, rather than observed intensity
  - Humans are good at it
  - Computers are not nearly as good



One source of constancy: local comparisons





http://www.echalk.co.uk/amusements/OpticalIllusions/colourPerception/colourPerception.html

Perception of Intensity

Color Correction

- Simple Idea: Multiply R, G, and B values by separate constants

$$\begin{bmatrix} \tilde{r} \\ \tilde{g} \\ \tilde{b} \end{bmatrix} = \begin{bmatrix} \alpha_r & 0 & 0 \\ 0 & \alpha_g & 0 \\ 0 & 0 & \alpha_b \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

- How to choose the constants?
  - "White World" assumption: brightest pixel is white
    - Divide by largest value
  - "Gray World" assumption: average value should be gray
    - E.g., Multiply R channel by avg(R) / avg ((R+G+B)/3)
  - White balancing: choose a reference as the white or gray color

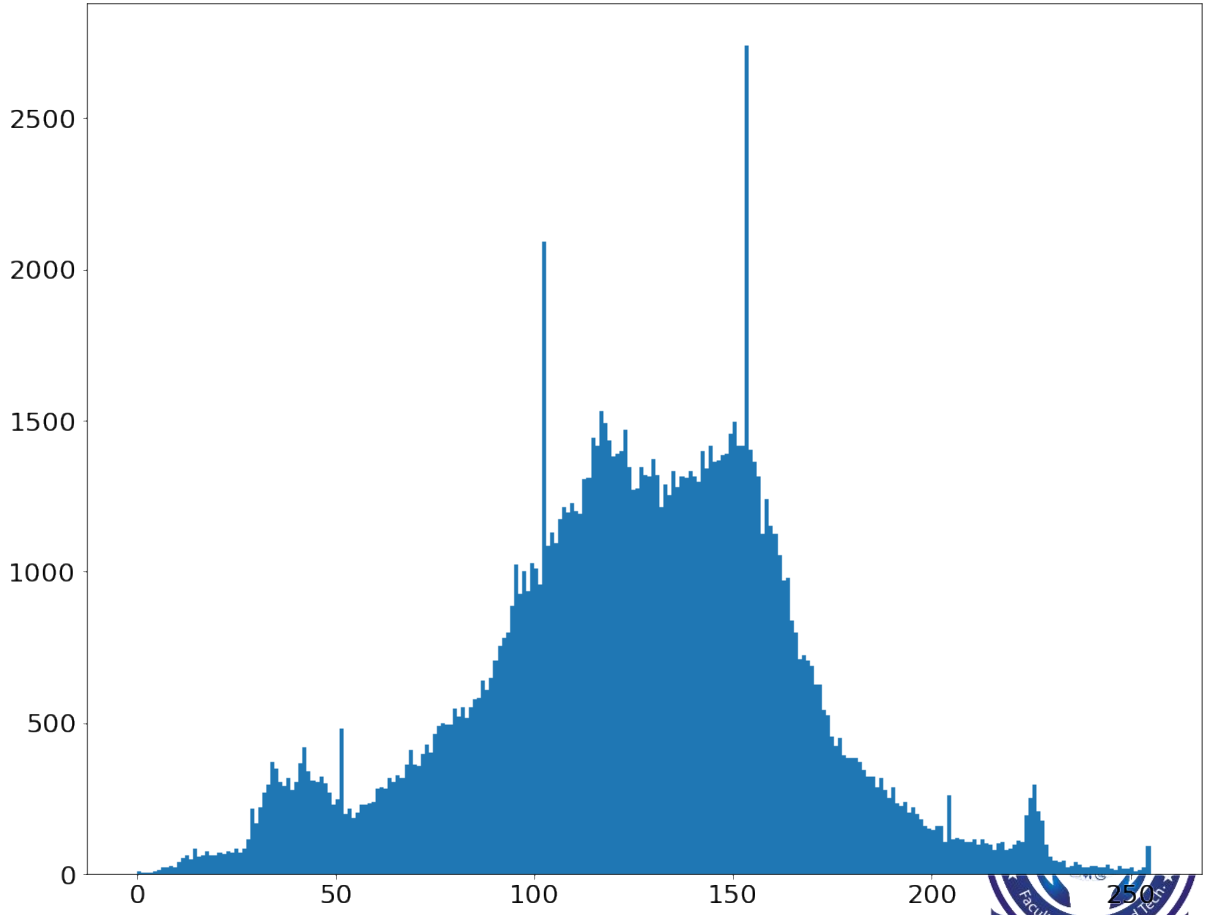
Histogram

- Histogram of an image provides the frequency of the brightness (intensity) value in the image.

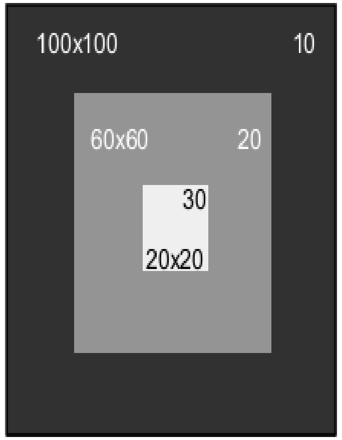
```
def histogram(im):
    h = np.zeros(255)
    for row in im.shape[0]:
        for col in im.shape[1]:
            val = im[row, col]
            h[val] += 1
```

Histogram

- Histogram captures the distribution of gray levels in the image.
- How frequently each gray level occurs in the image

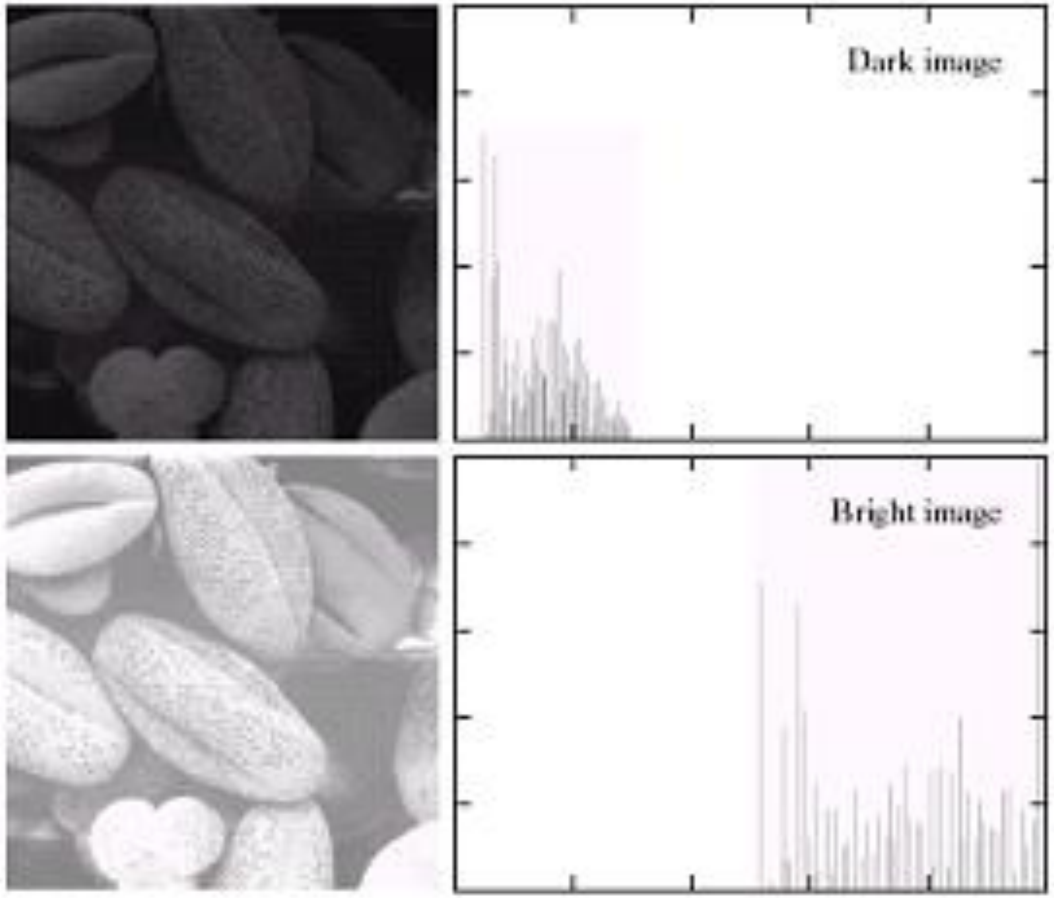


Computing Histograms



- Find Histogram
  - Find PDF
  - Find CDF
  - Find Mean
  - Find Variance

Histograms



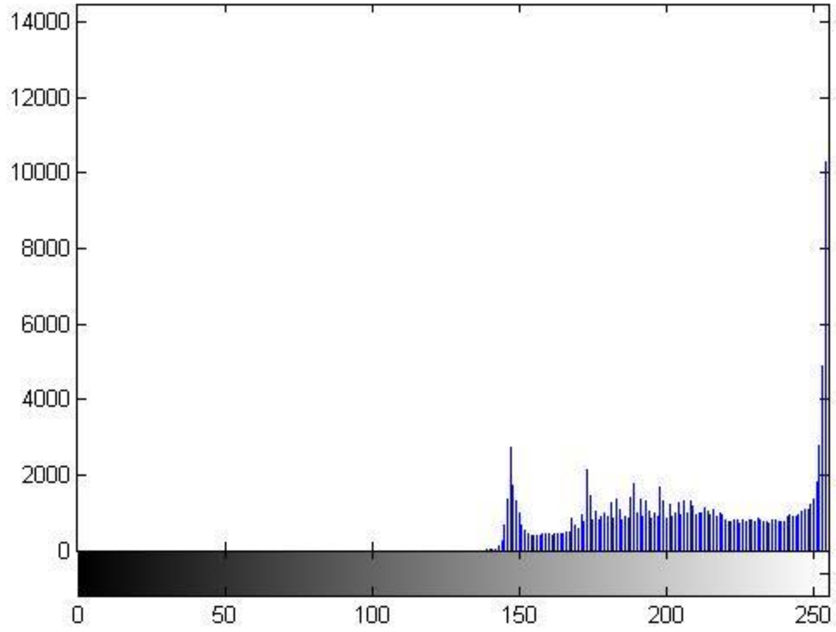
Dark image

Components of histogram are concentrated on the low side of the gray scale.

Bright image

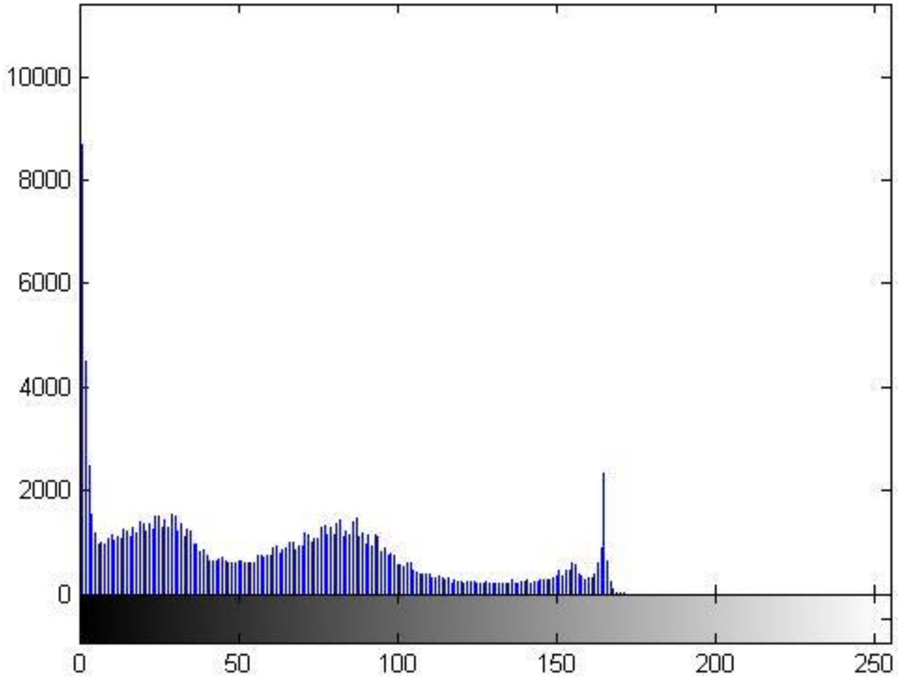
Components of histogram are concentrated on the high side of the gray scale.

Histogram Examples (Bright Image)

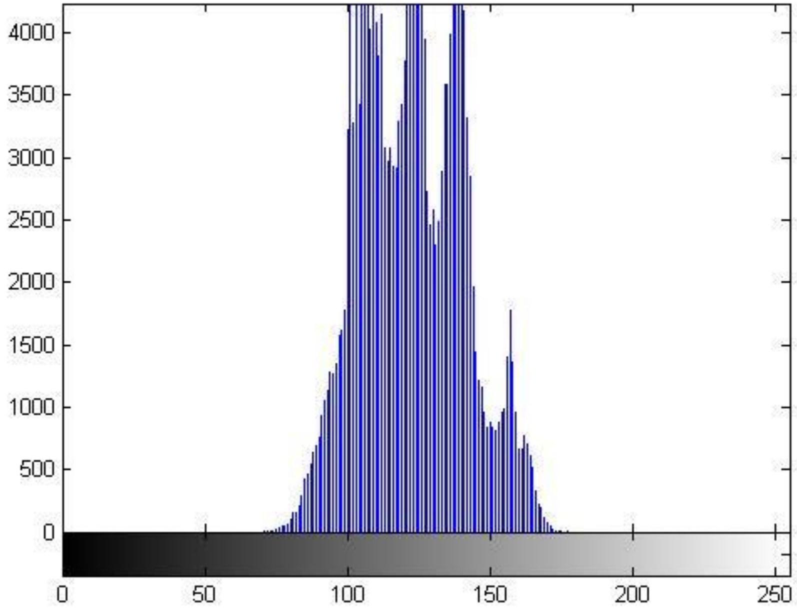


Dr. Sander Ali Khowaja

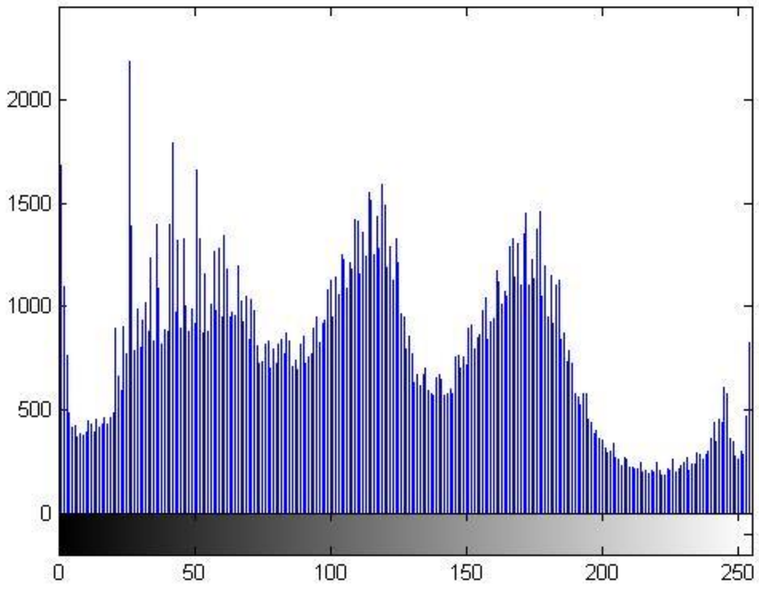
Histogram Example (Dark Image)



Histogram Example (Low Contrast Image)



Histogram Example (Contrast Equalized image)



Doing Histogram Equalization by Hand

- Get histogram of MxN input image $H_r(r)=n_r$. Gray levels range from 0…L-1
- Determine Probability Density Function (PDF)
- $P_r(r_k) = \frac{n_k}{MN}$
- Determine cumulative distribution function (CDF)
- $F_r(r_k) = \sum_{j=0}^{k} P_r(r_j)$
- Scale T(r) to desired range of output gray levels
- $T(r) = (L-1)F_r(r)$
- Apply the transformation s = T(r) to compute the output values

Doing Histogram Equalization by Hand

r	H(r)
0	0
1	0
2	0
3	10
4	20
5	40
6	20
7	10

Homework

Example: suppose that a 64*64, 8-levels image has the gray-level distribution shown in table 1.

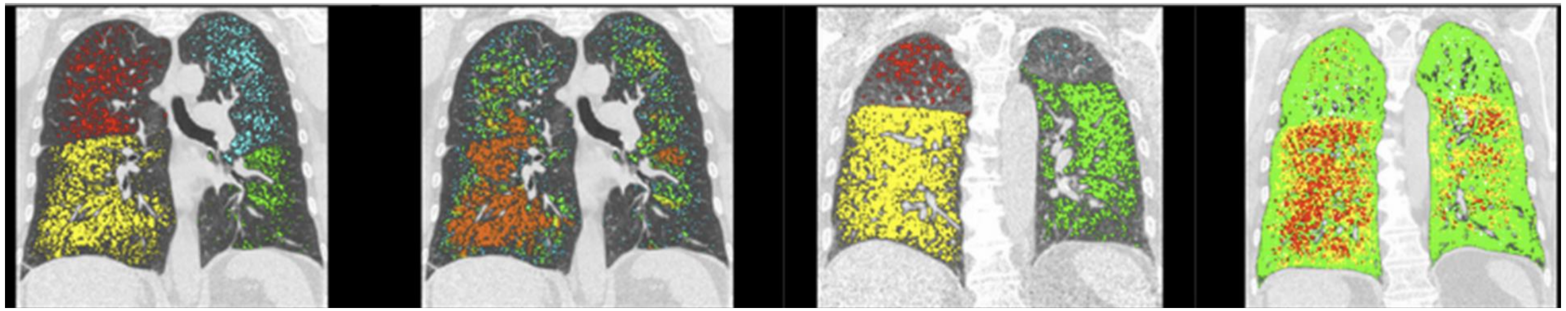
Table (1)

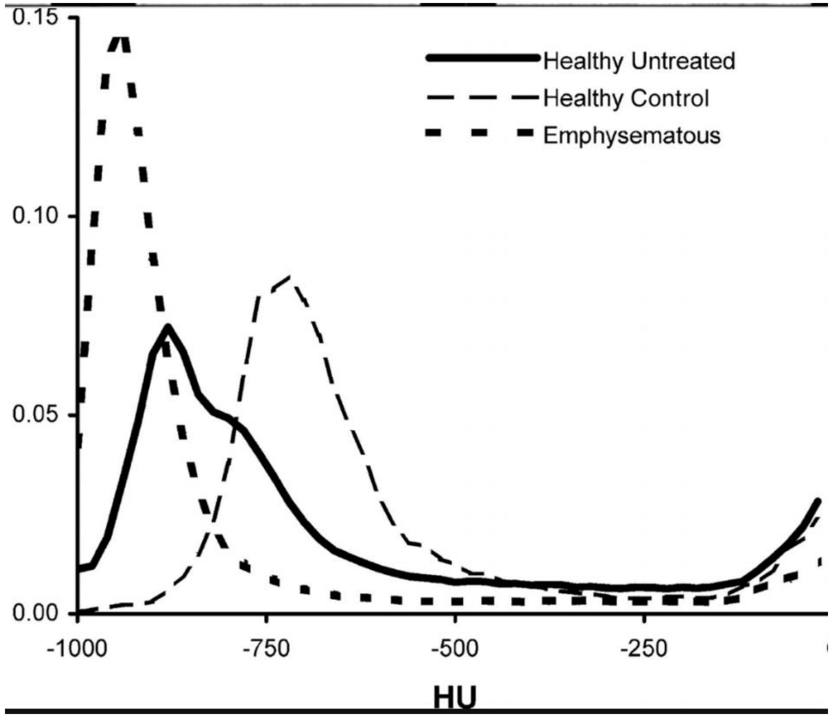
r_k	n_k	$P_r(r_k) = n_k/n$
$r_0 = 0$	790	0.19
$r_1=1/7$	1023	0.25
$r_2=2/7$	850	0.21
$r_3=3/7$	656	0.16
$r_4=4/7$	329	0.08
$r_5=5/7$	245	0.06
$r_6=6/7$	122	0.03
r7=7/7	81	0.02

Applications of Histogram

Slide credit: Dr. Mubarak Shah

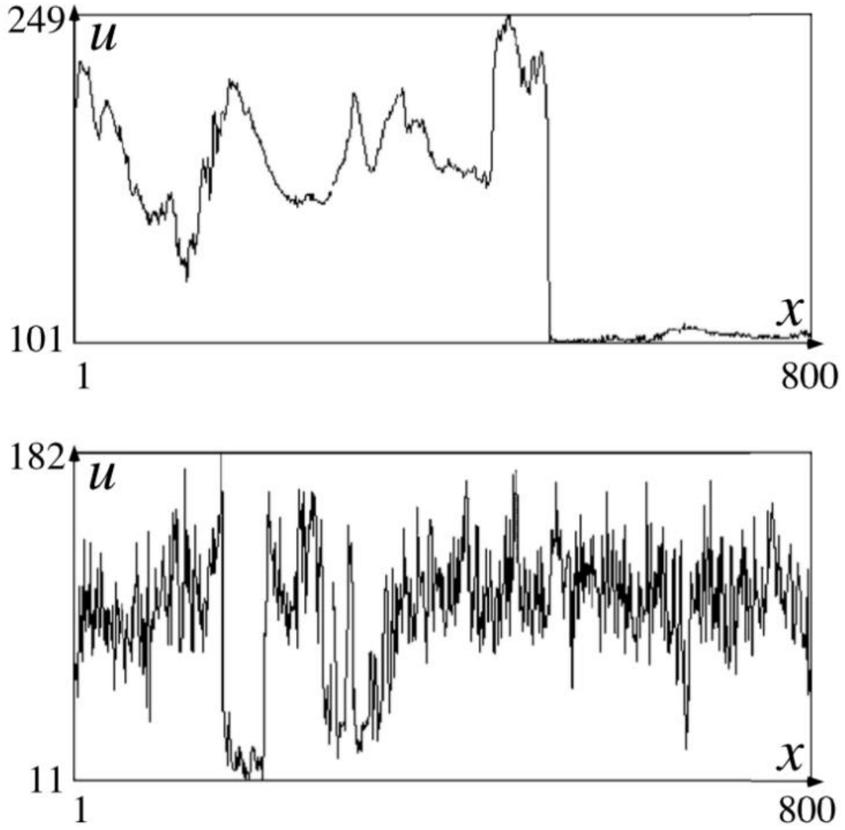
Applications of Histogram





Slide credit: Dr. Mubarak Shah

Applications of Histogram



Slide credit: Dr. Mubarak Shah

Histogram in Python

- From matplotlib import pyplot as plt
- Image = cv2.imread(“image path”)
- Image = cv2.cvtColor(Image,cv2.COLOR_BGR2GRAY)
- cv2.imshow(“Original”,Image)
- Hist = cv2.calcHist([Image],[0],None,[256],[0, 256])
- plt.figure()
- plt.title(“Grayscale Histogram”)
- plt.xlabel(“Bins”)
- plt.ylabel(“# of pixels”)
- plt.plot(Hist)
- plt.show()
- cv2.waitKey(0)

Histogram Equalization in Python

- Image = cv2.imread(“image path”)
- Image = cv2.cvtColor(Image, cv2.COLOR_BGR2GRAY)
- Eq = cv2.equalizeHist(Image)

- cv2.imshow(“Histogram Equalization”,np.hstack([Image,Eq]))
- cv2.waitKey(0)

Example from Python

Example from Python

Example from Python