DIGITAL IMAGE PROCESSING

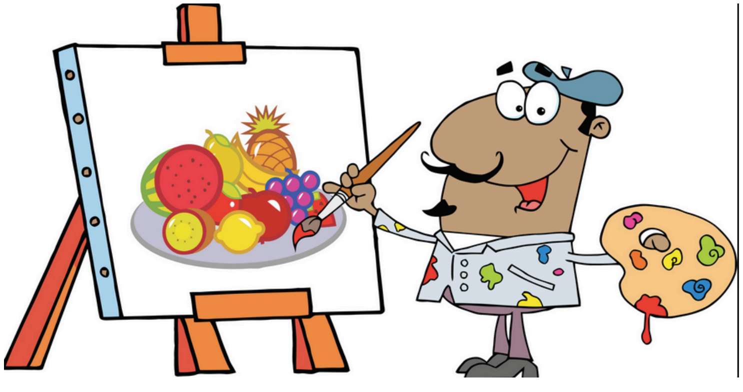
Lecture 3

Scale-space, Colors

Tammy Riklin Raviv

Electrical and Computer Engineering

Ben-Gurion University of the Negev



Last Class: Filtering

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Linear filtering is a weighted sum/difference of pixel values

- Can smooth, sharpen, translate (among many other uses)
- Filtering in Matlab, e.g. to filter image f with h

```
g = filter2( h, f );
```

h=filter   f=image

```
e.g. h = fspecial('gaussian');
```

Partial Derivatives

First order partial derivatives:

$$\frac{\partial I(x,y)}{\partial x} = I(x+1,y) - I(x,y)$$ Left Derivative

$$\frac{\partial I(x,y)}{\partial x} = I(x,y) - I(x-1,y)$$ Right Derivative

$$\frac{\partial I(x,y)}{\partial x} = \frac{I(x+1,y) - I(x-1,y)}{2}$$ Central Derivative

Partial Derivatives

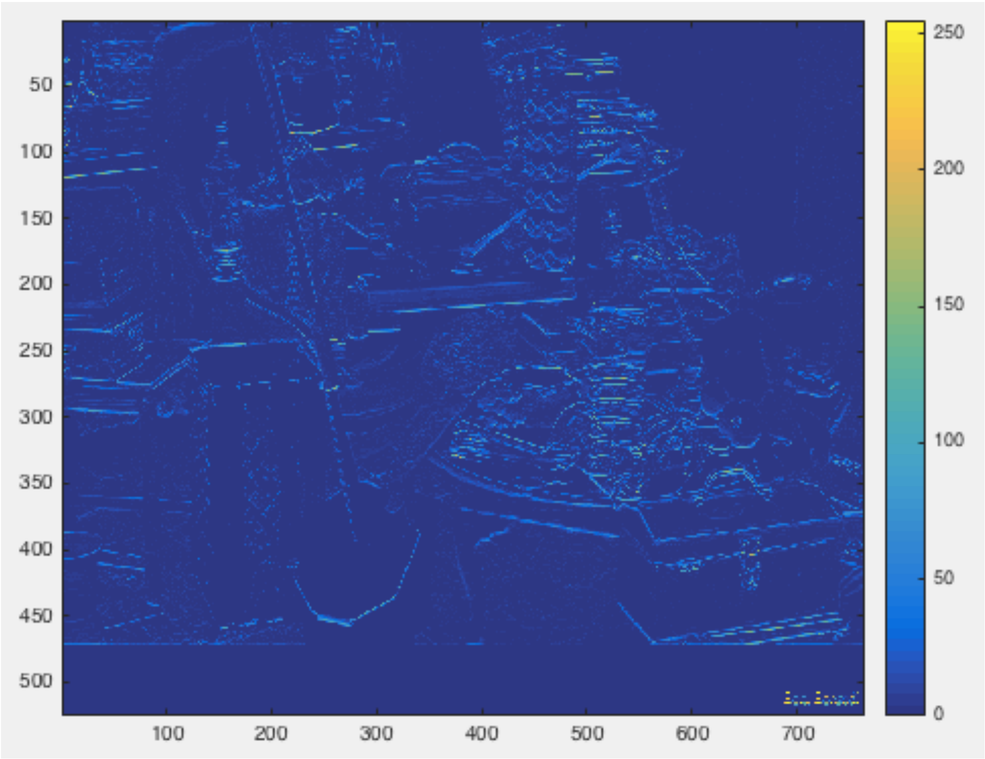
First order partial derivatives:

$$\frac{\partial I(x,y)}{\partial y} = I(x,y+1) - I(x,y)$$

$$\frac{\partial I(x,y)}{\partial y} = I(x,y) - I(x,y-1)$$

$$\frac{\partial I(x,y)}{\partial y} = \frac{I(x,y+1) - I(x,y-1)}{2}$$

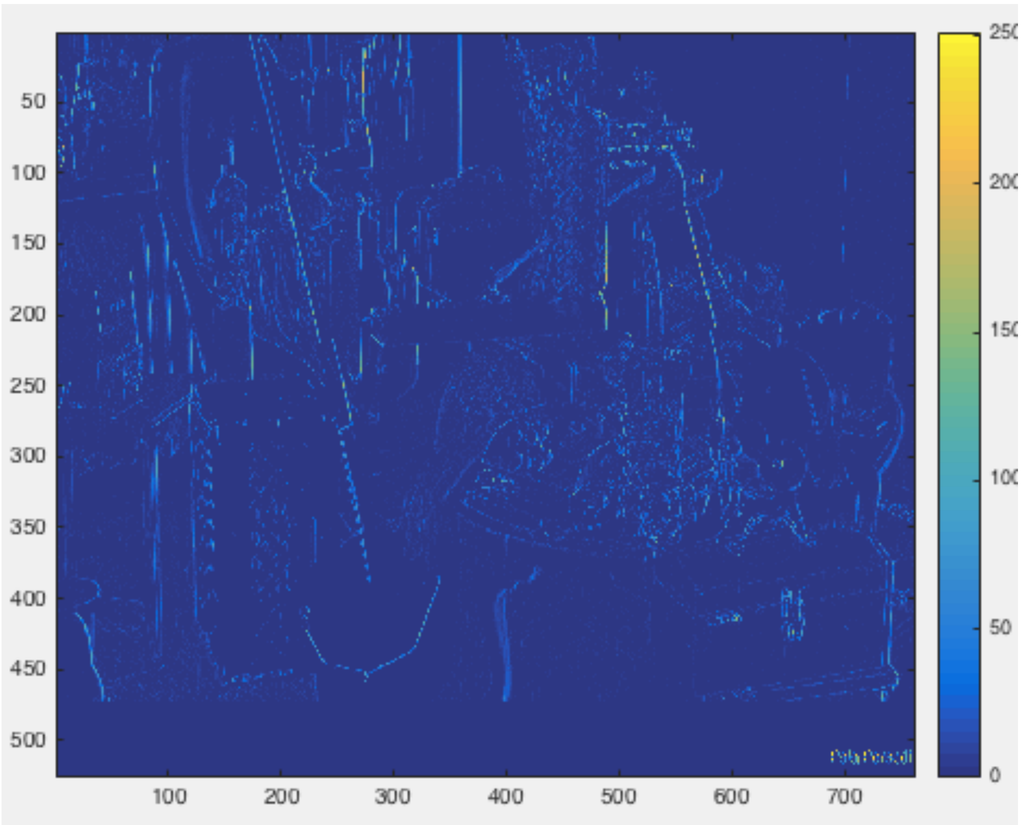
Partial Derivatives



```
>> myFirstImage = imread('someImage.png');
>> I = myFirstImage(:,:,1);
>> Ir = I(2:end,:);
>> Il = I(1:end-1,:);
>> Idx = Il-Ir;
>> figure;imagesc(Idx);colorbar
```

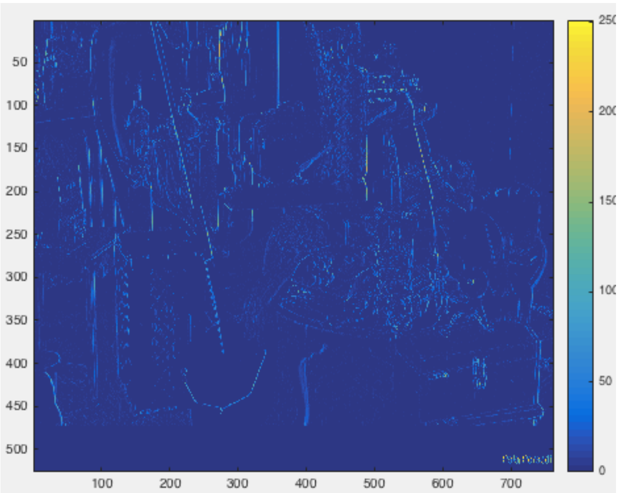
```
>> figure;imagesc(abs(Idx));colorbar
```

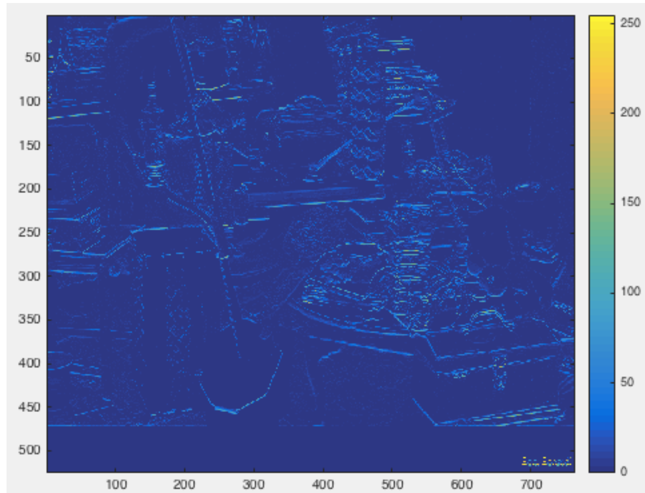
Partial Derivatives



```
>> Iu = I(:,1:end-1);
>> Id = I(:,2:end);
>> Idy = Iu-Id;
>> figure;imagesc(abs(Idy));colorbar
```

Partial Derivatives






```
>>
>> figure;imagesc(diff(I,1));colorbar
>> figure;imagesc(diff(I,2));colorbar
```

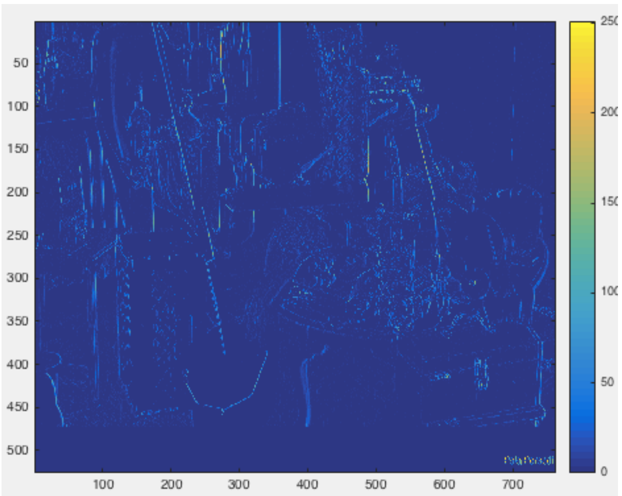
Gradients

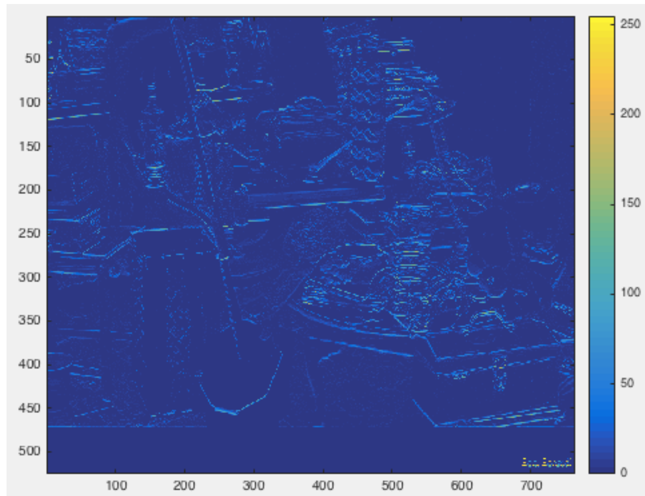
Gradients:

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$$

$$|\nabla I| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

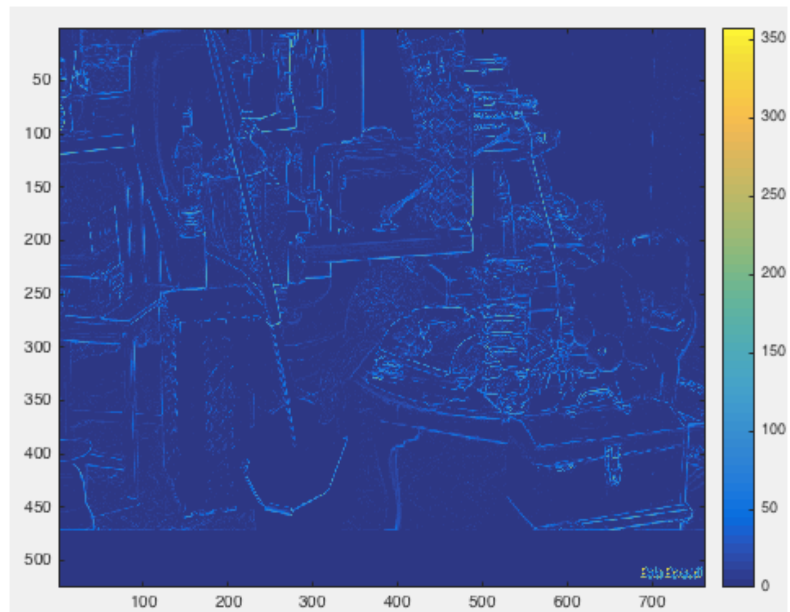
Gradients



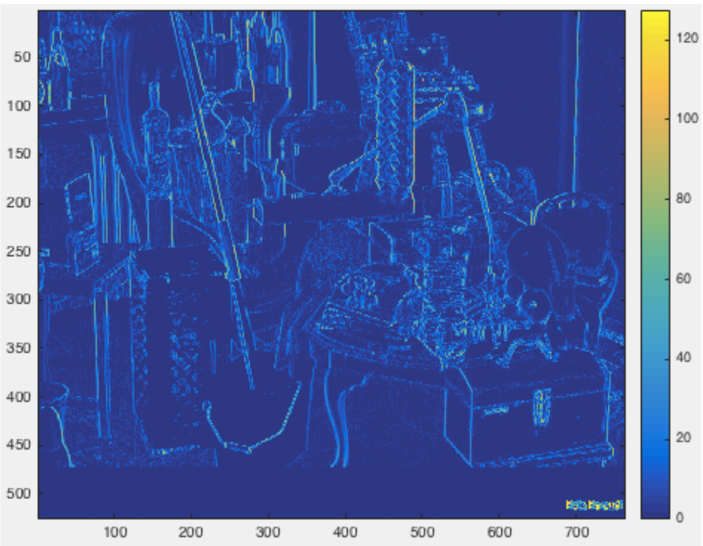


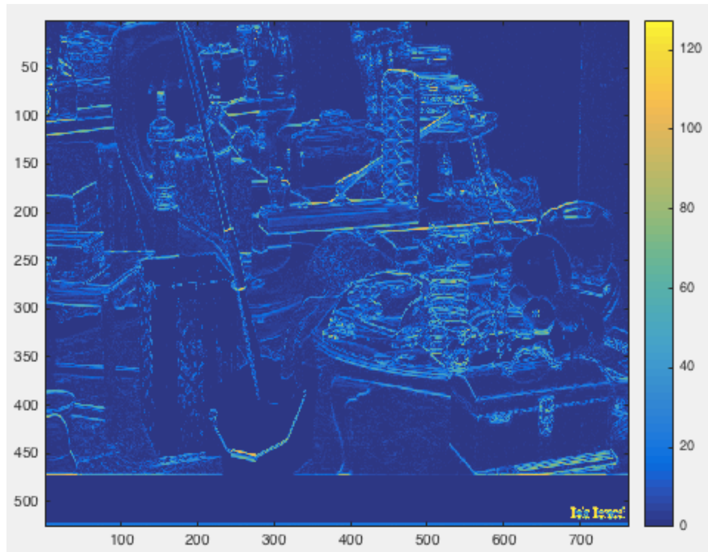

```
>>
>> figure;imagesc(diff(I,1));colorbar
>> figure;imagesc(diff(I,2));colorbar

>> G = sqrt(double(Idx).^2+double(Idy).^2);
>> figure;imagesc(G);colorbar
```



Gradients

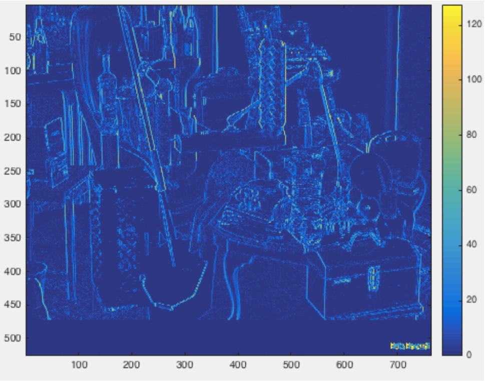


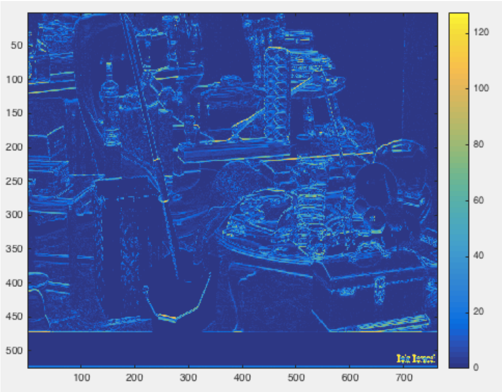


Can you explain the differences?


```
>> [Igx,Igy] = gradient(double(I));
>> figure;imagesc(abs(Igx));colorbar
>> figure;imagesc(abs(Igy));colorbar
>>
```

Gradients





```
>>
>> type gradient
```

You get long function
but here is the important part:

```
% Take forward differences on left and right edges
if n > 1
    g(:,1) = (f(:,2) - f(:,1))/(h(2)-h(1));
    g(:,n) = (f(:,n) - f(:,n-1))/(h(end)-h(end-1));
end

% Take centered differences on interior points
if n > 2
    h = h(3:n) - h(1:n-2);
    g(:,2:n-1) = bsxfun(@rdivide,(f(:,3:n)-f(:,1:n-2)),h);
end
varargout{2} = g;
```

Derivatives & the Laplacian

- Second order derivatives

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

$$\frac{\partial^2 I}{\partial x^2} = I(x+1, y) + I(x-1, y) - 2I(x, y)$$

$$\frac{\partial^2 I}{\partial y^2} = I(x, y+1) + I(x, y-1) - 2I(x, y)$$

$$\nabla^2 I = I(x+1, y) + I(x-1, y) + I(x, y+1) + I(x, y-1) - 4I(x, y)$$

Divergence

Let x, y be a 2D Cartesian coordinates

Let i, j be corresponding basis of unit vectors

The divergence of a continuously differential vector field

$$F = Ui + Vj$$

is defined as the (signed) scalar-valued function:

$$\mathrm{div} F = \nabla \cdot F = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right) \cdot (U, V) = \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y}$$

Back to Laplacian

The Laplacian of a scalar function or functional expression is the divergence of the gradient of that function or expression:

$$\Delta I = \nabla \cdot (\nabla I)$$

Therefore, you can compute the Laplacian using the divergence and gradient functions:

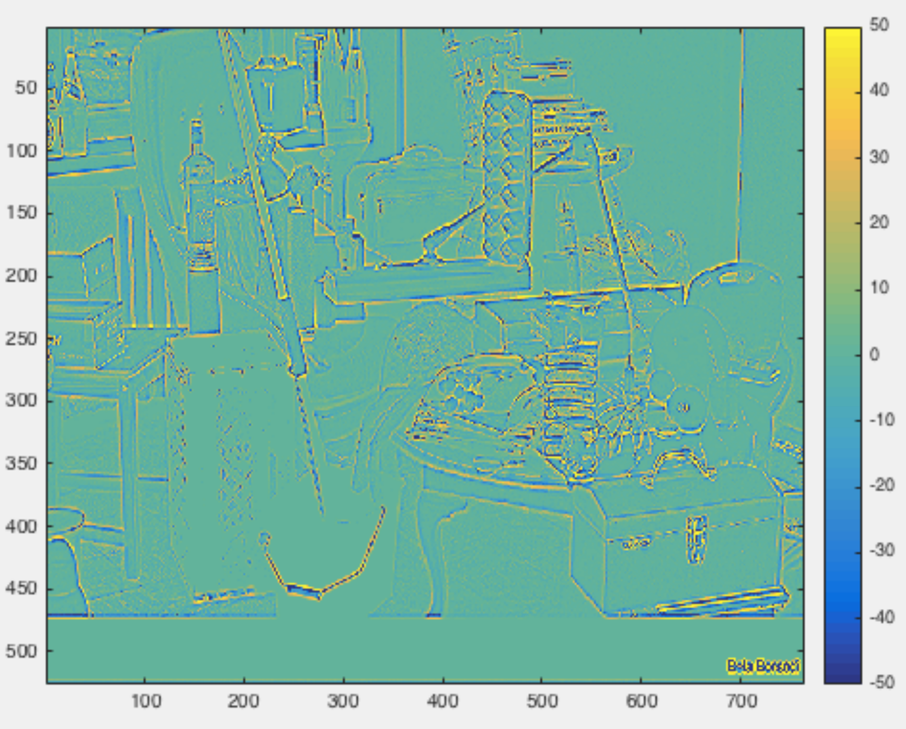
```
>> [Igx,Igy] = gradient(double(I));
>> div = divergence(Igx,Igy);
>> figure;imagesc(div);colorbar
>> figure;imagesc(abs(div));colorbar
```

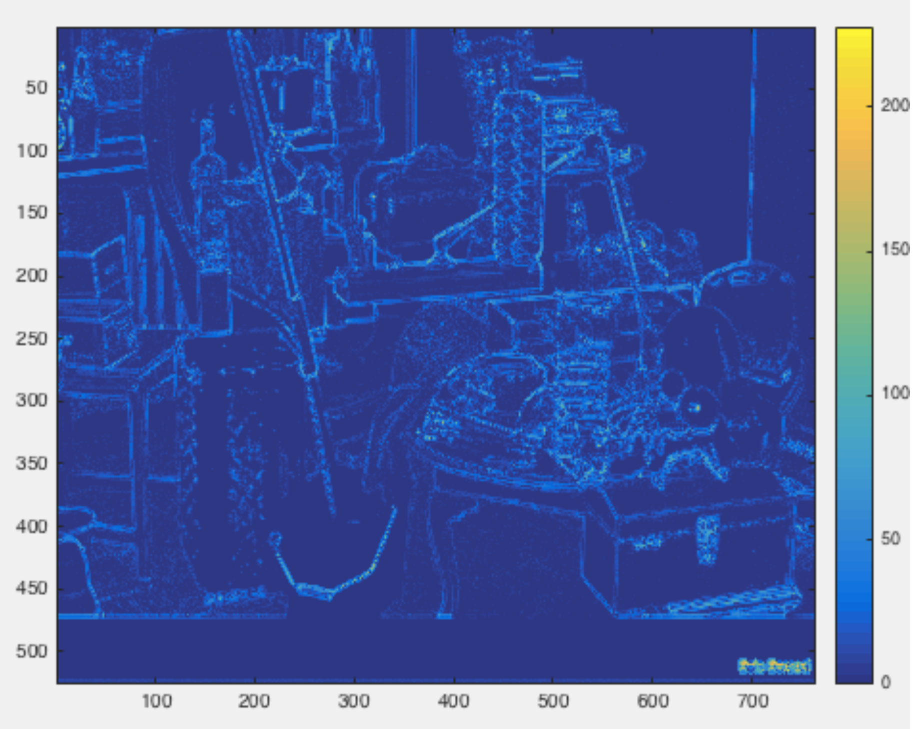
Back to Laplacian

$$\Delta I = \nabla \cdot (\nabla I)$$

```
>> [Igx,Igy] = gradient(double(I));
>> div = divergence(Igx,Igy);
>> figure;imagesc(div);colorbar
>> figure;imagesc(abs(div));colorbar
```

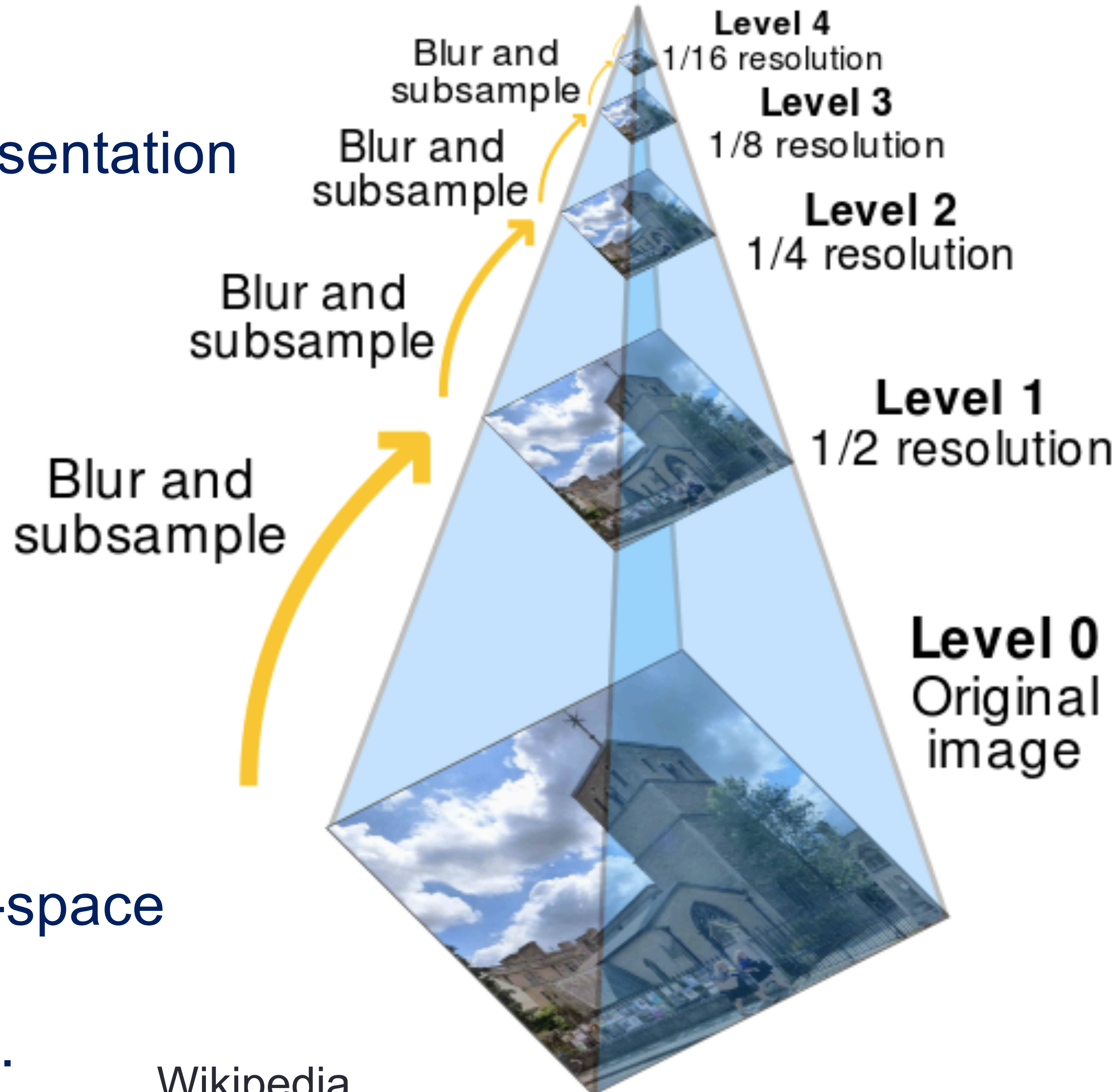
abs





Pyramids

Multi-scale signal representation

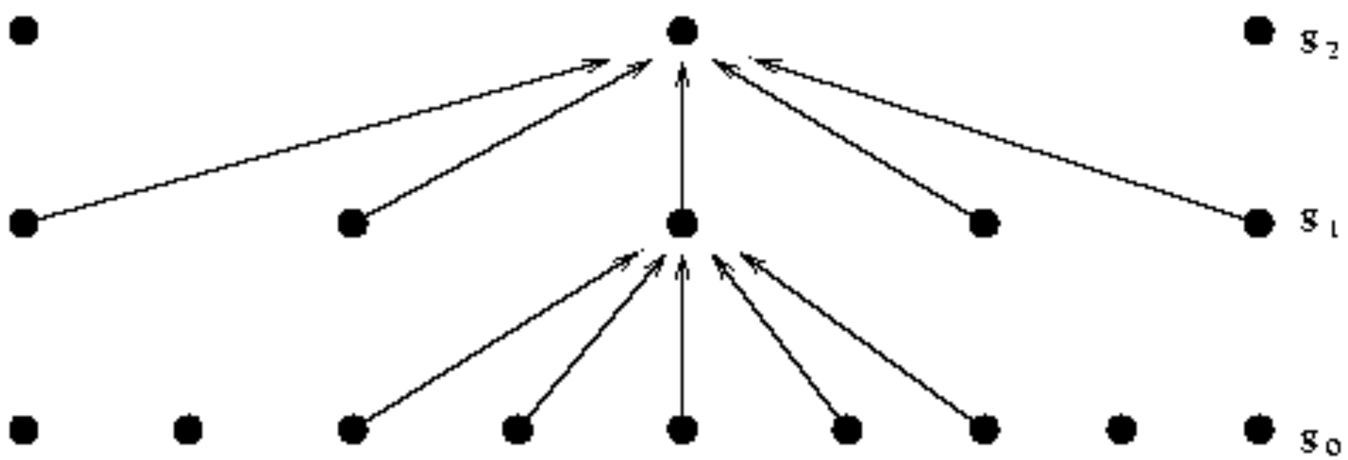


A predecessor to scale-space representation and multiresolution analysis.

Wikipedia

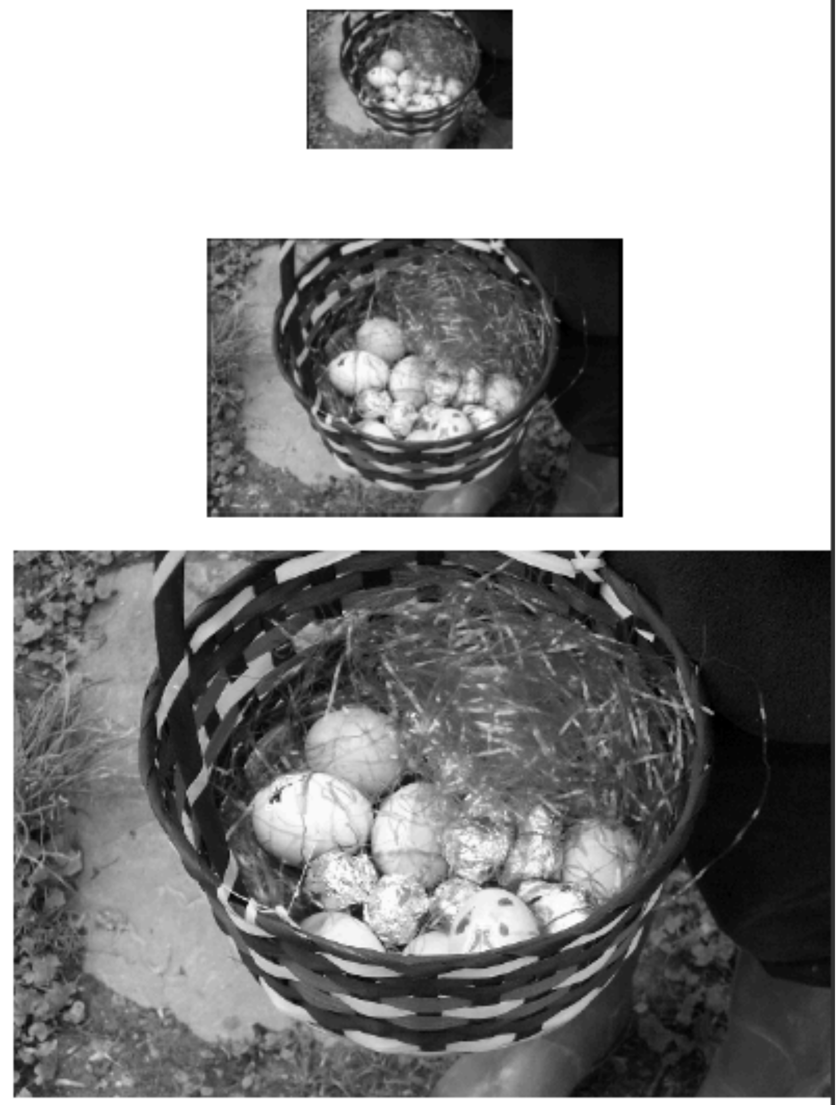
Gaussian Pyramid

The Gaussian Pyramid is a hierarchy of low-pass filtered versions of the original image, such that successive levels correspond to lower frequencies.



Gaussian Pyramids

- Algorithm:
  - 1. Filter with $\mathcal{G}(\sigma = 1)$
  - 2. Resample at every other pixel
  - 3. Repeat



Laplacian Pyramid

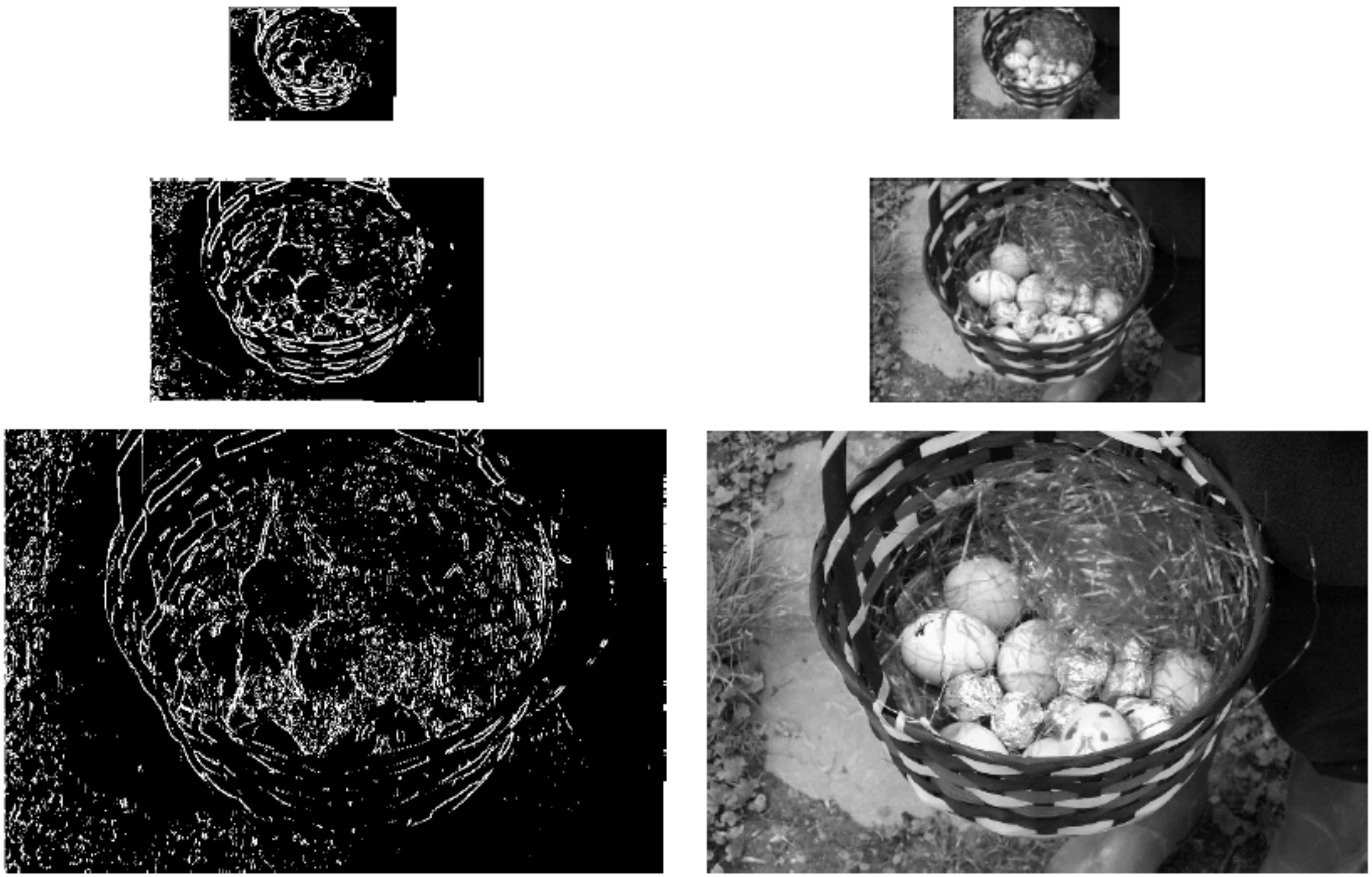
The Laplacian Pyramid is a decomposition of the original image into a hierarchy of images such that each level corresponds to a different band of image frequencies. This is done by taking the difference of levels in the Gaussian pyramid.

For image *I* the Laplacian pyramid *L*(*I*) is:

$$L_i = G_i - \mathrm{expand}(G_{i+1})$$

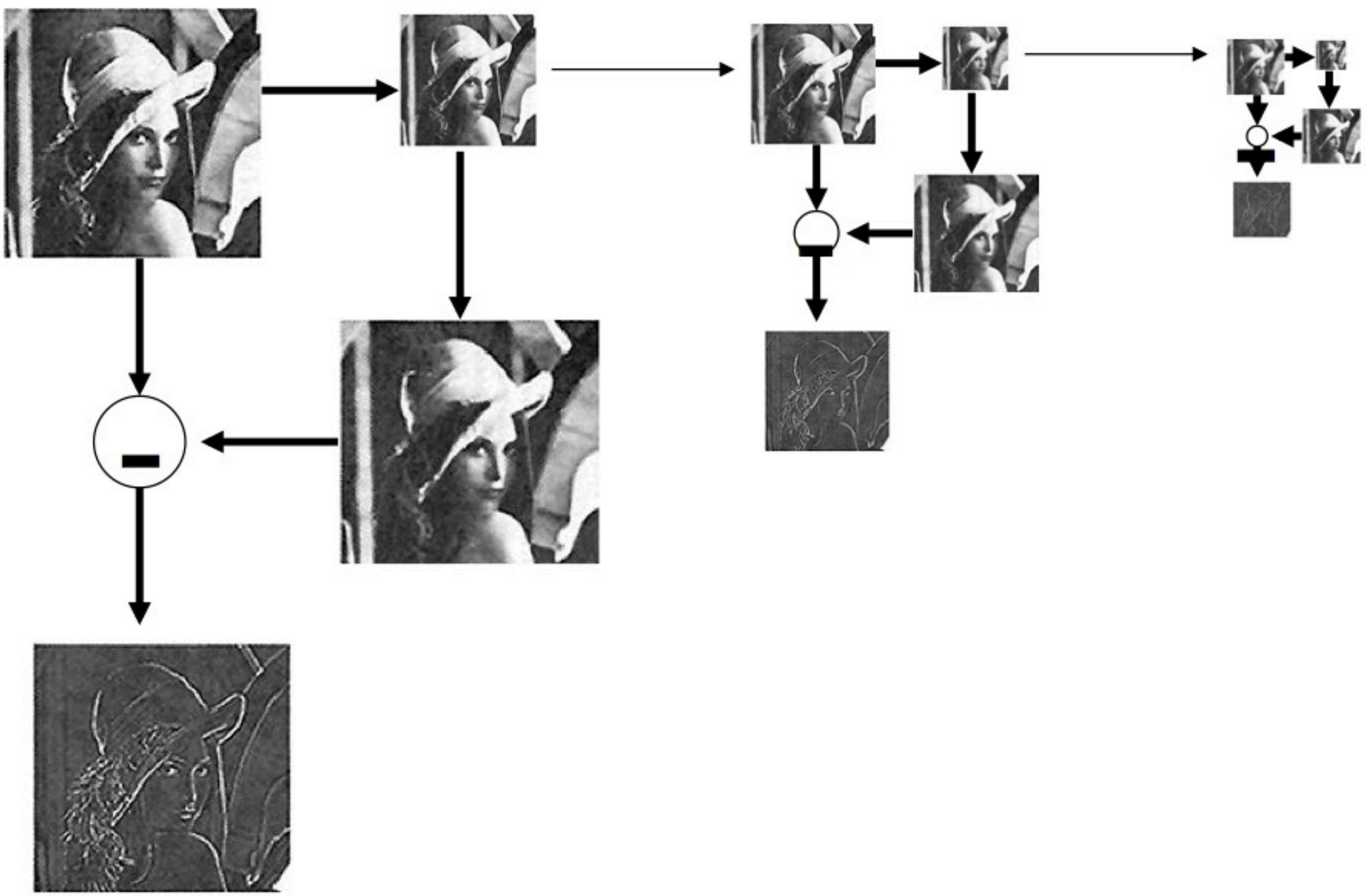
$$L_i = G_i - \mathrm{blur}(G_i)$$

Laplacian Pyramid Algorithm



Source: G Hager Slides

Pyramids Construction



Laplacian Pyramid & Laplacian

The well-known Laplacian derivative operator (isotropic second derivative) is given by

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

For Gaussian kernels, $g(x;\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2}$,

$$\begin{aligned}
\frac{dg(x;\sigma)}{dx} &= \frac{-x}{\sigma^2} g(x;\sigma) \\
\frac{d^2 g(x;\sigma)}{dx^2} &= \left(\frac{x^2}{\sigma^2} - 1\right) \frac{1}{\sigma^2} g(x;\sigma) \\
\frac{dg(x;\sigma)}{d\sigma} &= \left(\frac{x^2}{\sigma^2} - 1\right) \frac{1}{\sigma} g(x;\sigma)
\end{aligned}$$

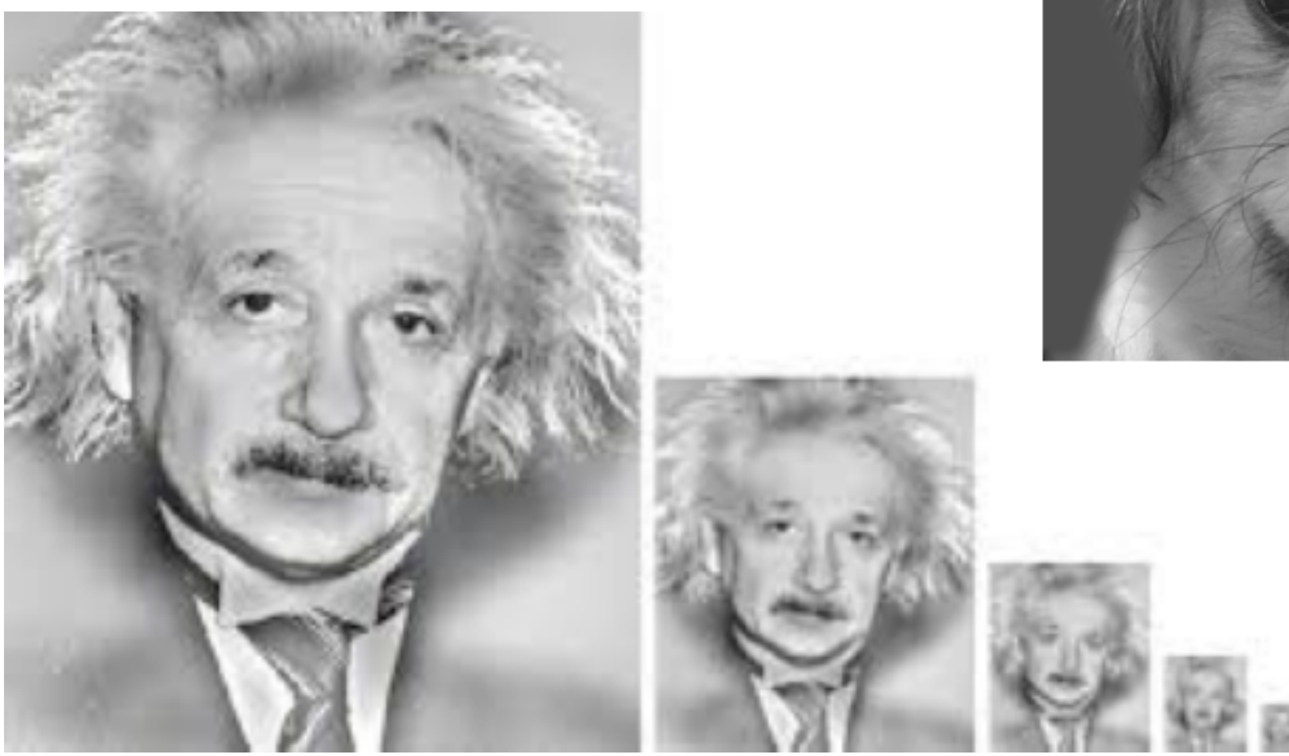
Therefore

$$\frac{d^2 g(x;\sigma)}{dx^2} = c_0(\sigma) \frac{d\, g(x;\sigma)}{d\sigma} \approx c_1(\sigma)\left(g(x;\sigma) - g(x;\sigma + \Delta\sigma)\right)$$

http://www.cs.toronto.edu/~jepson/
csc320/notes/pyramids.pdf

Hybrid Images

Original image



Isotropic Diffusion

The diffusion equation is a general case of the heat equation that describes the density changes in a material undergoing diffusion over time. Isotropic diffusion, in image processing parlance, is an instance of the heat equation as a partial differential equation (PDE), given as:

$$\frac{\partial I}{\partial t} = \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

where, I is the image and *t* is the time of evolution.

Manasi Datar

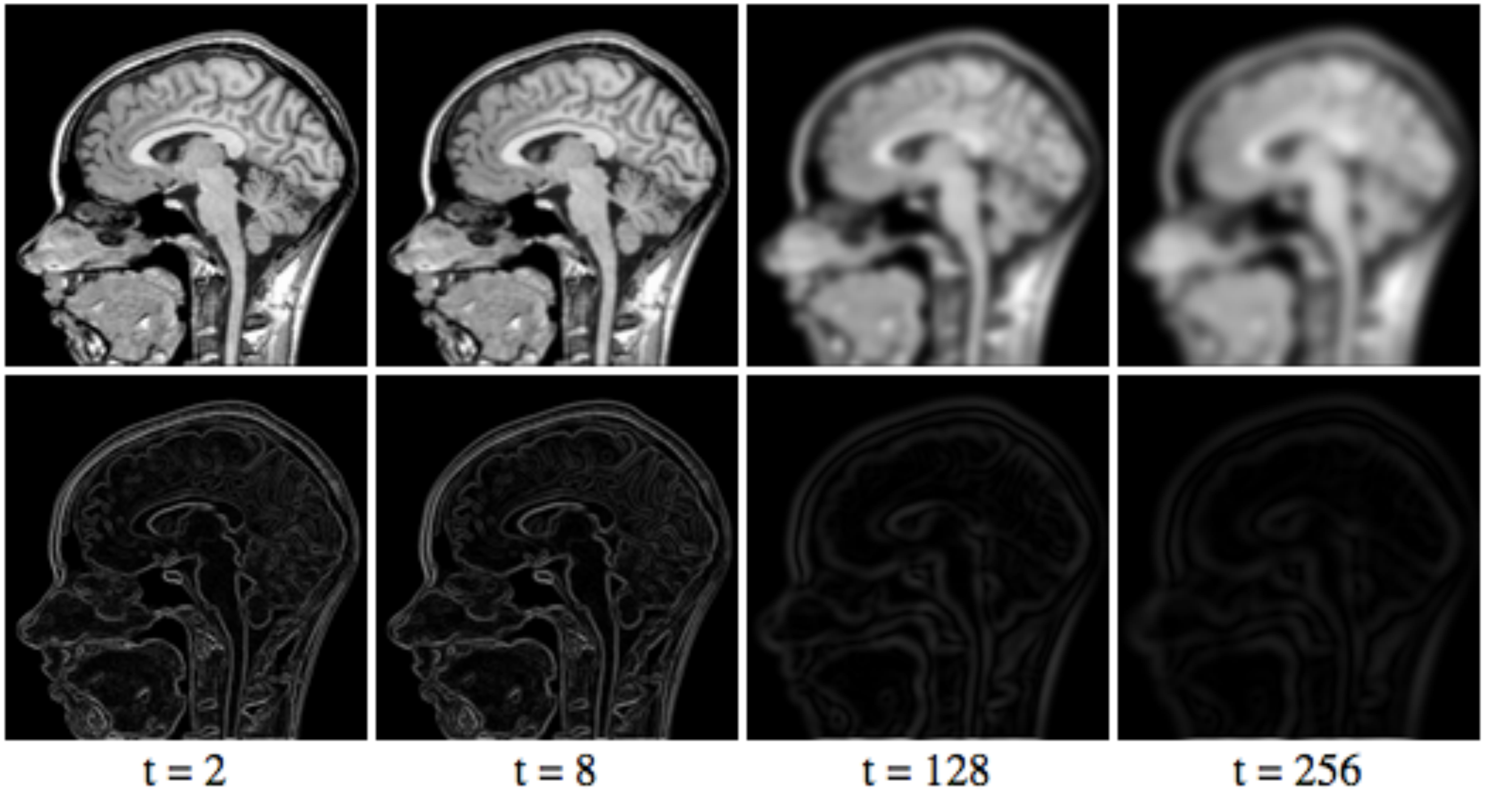
Isotropic Diffusion

Solving this for an image is equivalent to convolution with some Gaussian kernel.

In practice we iterate as follows:

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda \left[I_{i-1,j}^{t} + I_{i+1,j}^{t} + I_{i,j-1}^{t} + I_{i,j+1}^{t} - 4I_{i,j}^{t} \right]$$

Isotropic Diffusion



We can notice that while the diffusion process blurs the image considerably as the number of iterations increases, the edge information progressively degrades as well.

Anisotropic Diffusion: Perona-Malik

Perona & Malik introduce the flux function as a means to constrain the diffusion process to contiguous homogeneous regions, but not cross region boundaries. The heat equation (after appropriate expansion of terms) is thus modified to:

$$\frac{\partial I}{\partial t} = c(x, y, t)\Delta I + \nabla c \cdot \nabla I$$

where *c* is the proposed flux function which controls the rate of diffusion at any point in the image.

Anisotropic Diffusion: Perona-Malik

A choice of *c* such that it follows the gradient magnitude at the point enables us to restrain the diffusion process as we approach region boundaries. As we approach edges in the image, the flux function may trigger inverse diffusion and actually enhance the edges.

Anisotropic Diffusion: Perona-Malik

Perona & Malik suggest the following two flux functions:

$$c\left(||\nabla I||\right) = e^{-(||\nabla I||/K)^2}$$

$$c\left(||\nabla I||\right) = \frac{1}{1 + \left(\frac{||\nabla I||}{K}\right)^2}$$

Anisotropic Diffusion: Perona-Malik

The flux functions offer a trade-off between edge-preservation and blurring (smoothing) homogeneous regions. Both the functions are governed by the free parameter κ which determines the edge-strength to consider as a valid region boundary. Intuitively, a large value of κ will lead back into an isotropic-like solution. We will experiment with both the flux functions in this report.

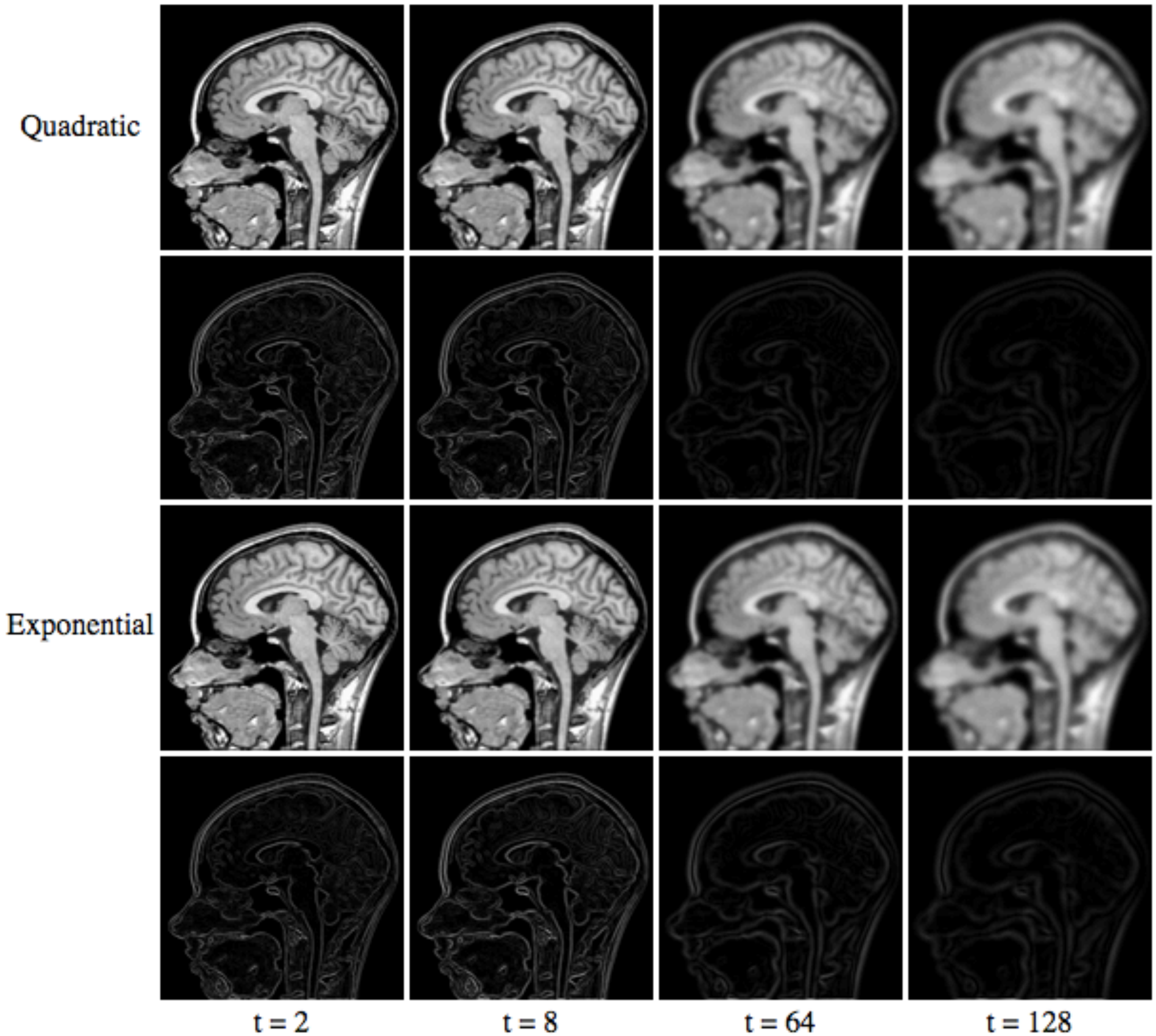
Anisotropic Diffusion: Perona-Malik

A discrete numerical solution can be derived for the anisotropic case as follows:

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda \left[c_N \cdot \nabla_N I + c_S \cdot \nabla_S I + c_E \cdot \nabla_E I + c_W \cdot \nabla_W I \right]_{i,j}^{t}$$

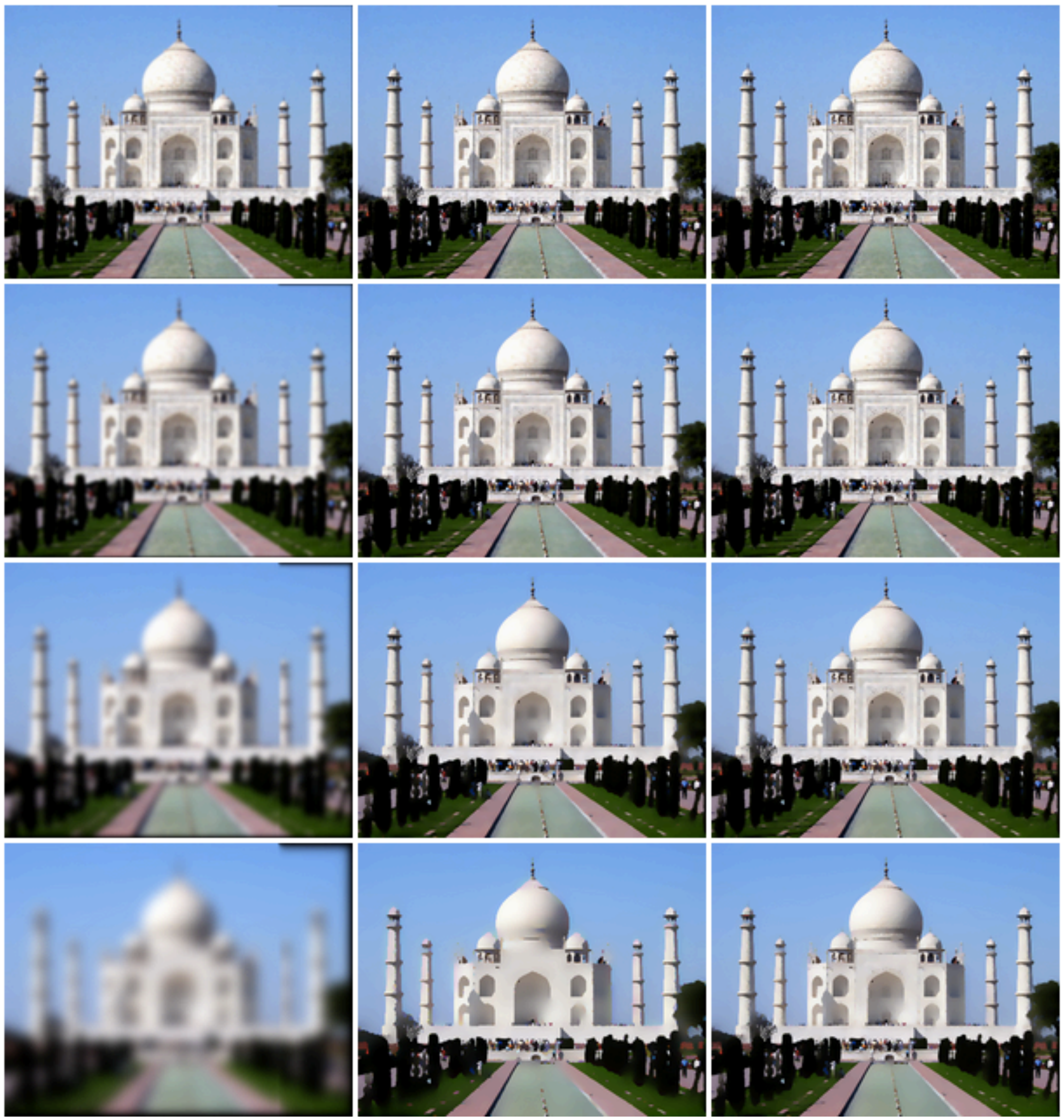
where {N,S,W,E} correspond to the pixel above, below, left and right of the pixel under consideration (i,j).

Anisotropic Diffusion: Perona-Malik



Anisotropic vs. Isotropic Diffusion

Isotropic quadratic exponent



t = 2

t = 8

t = 32

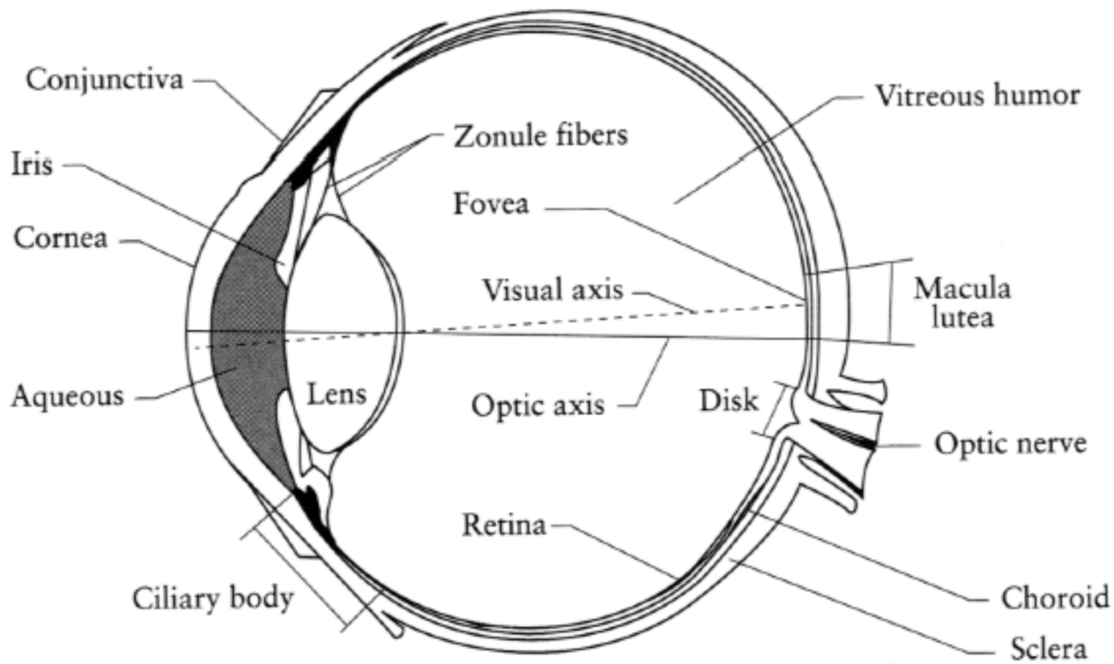
t = 128

Bonus Question: Image Enhancement

- Take an image (any image, but preferably one's that needs enhancement) and enhance it.
- Use what learned in this class to do so
- Plot the "before" and "after"
- Plot its derivatives before and after
- Matlab code is needed
- 3 Best works in class get 1 bonus point

Colors

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 sensor?
  - photoreceptor cells (rods and cones) in the **retina**

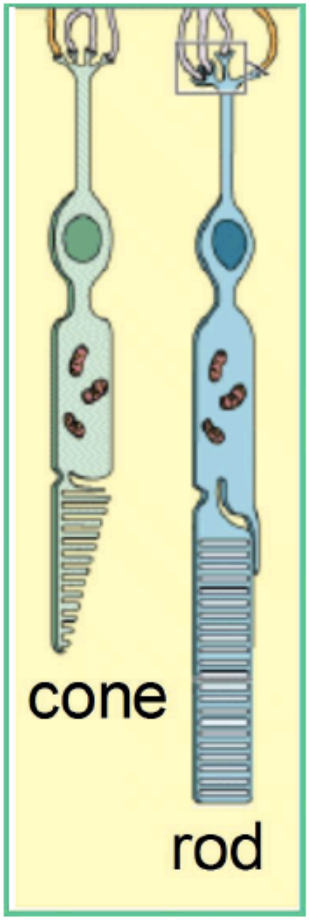
Slide by Steve Seitz

The Eye

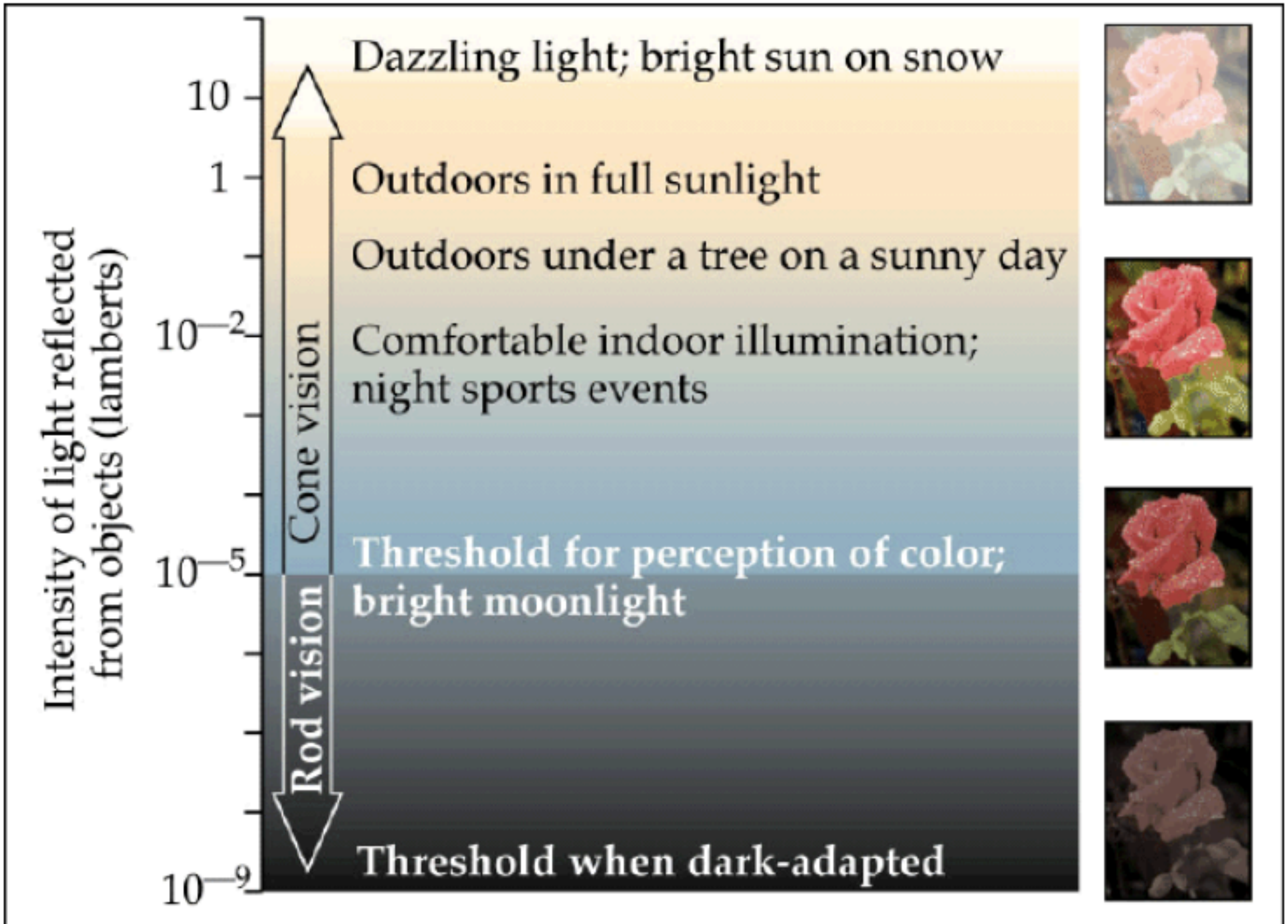
Two types of light-sensitive receptors

Cones
- cone-shaped
- less sensitive
- operate in high light
- color vision

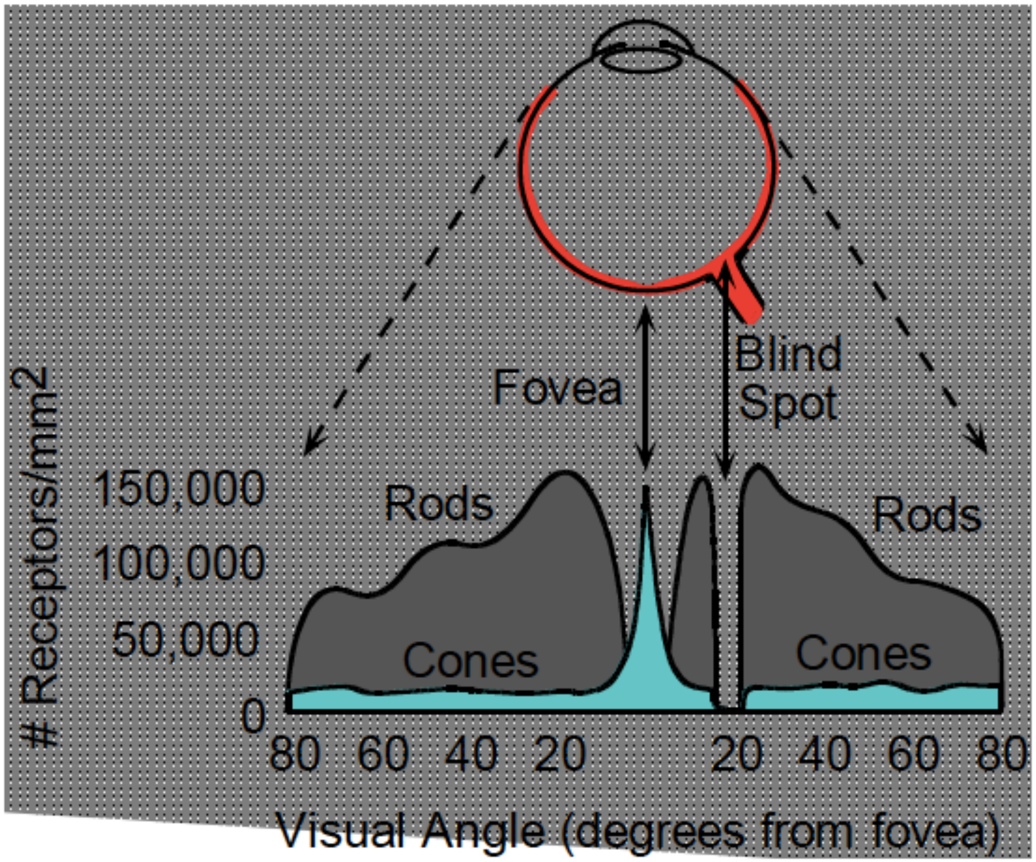
Rods
- rod-shaped
- highly sensitive
- operate at night
- gray-scale vision



Rod & Cone Sensitivity



Distribution of Rods & Cones

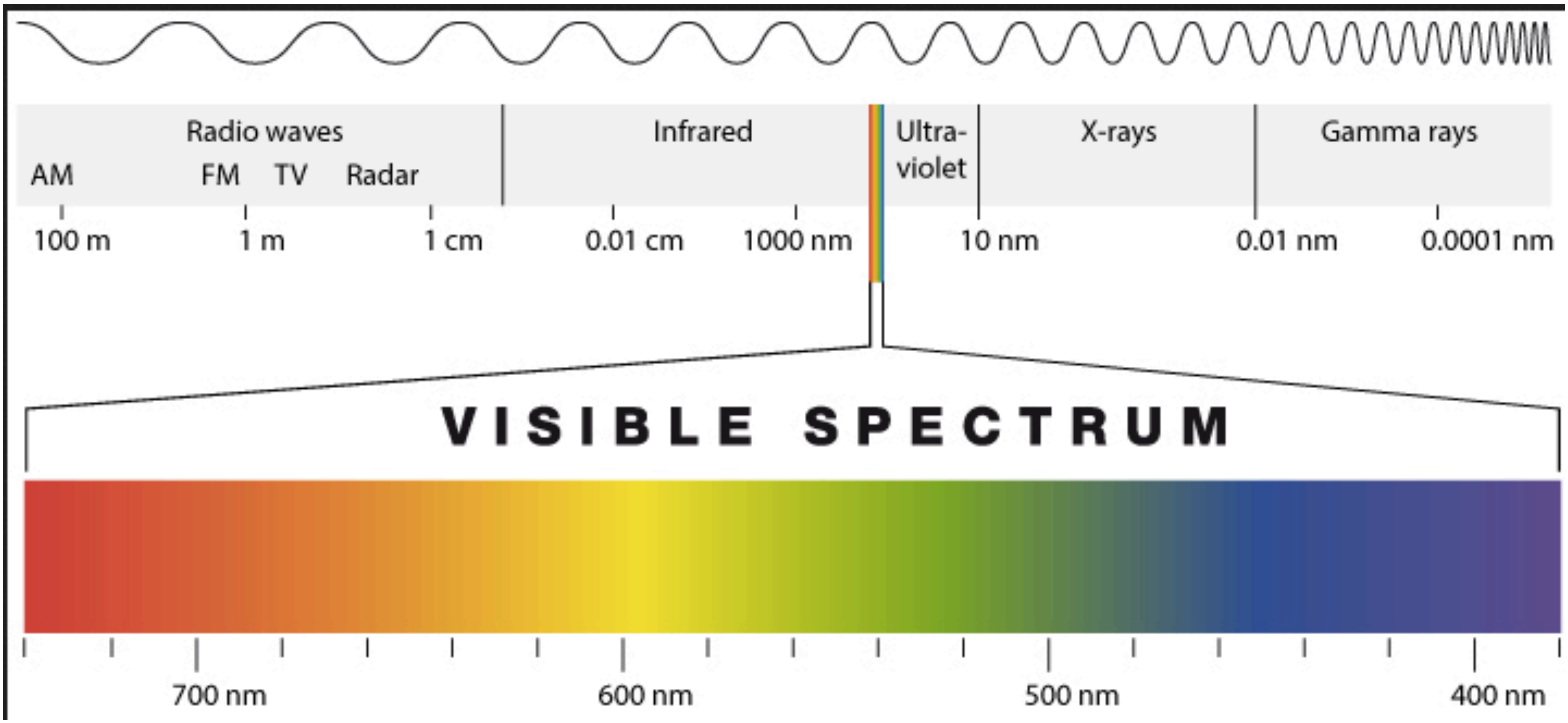


Night Sky: why are there more stars off-center?

Averted vision: http://en.wikipedia.org/wiki/Averted_vision

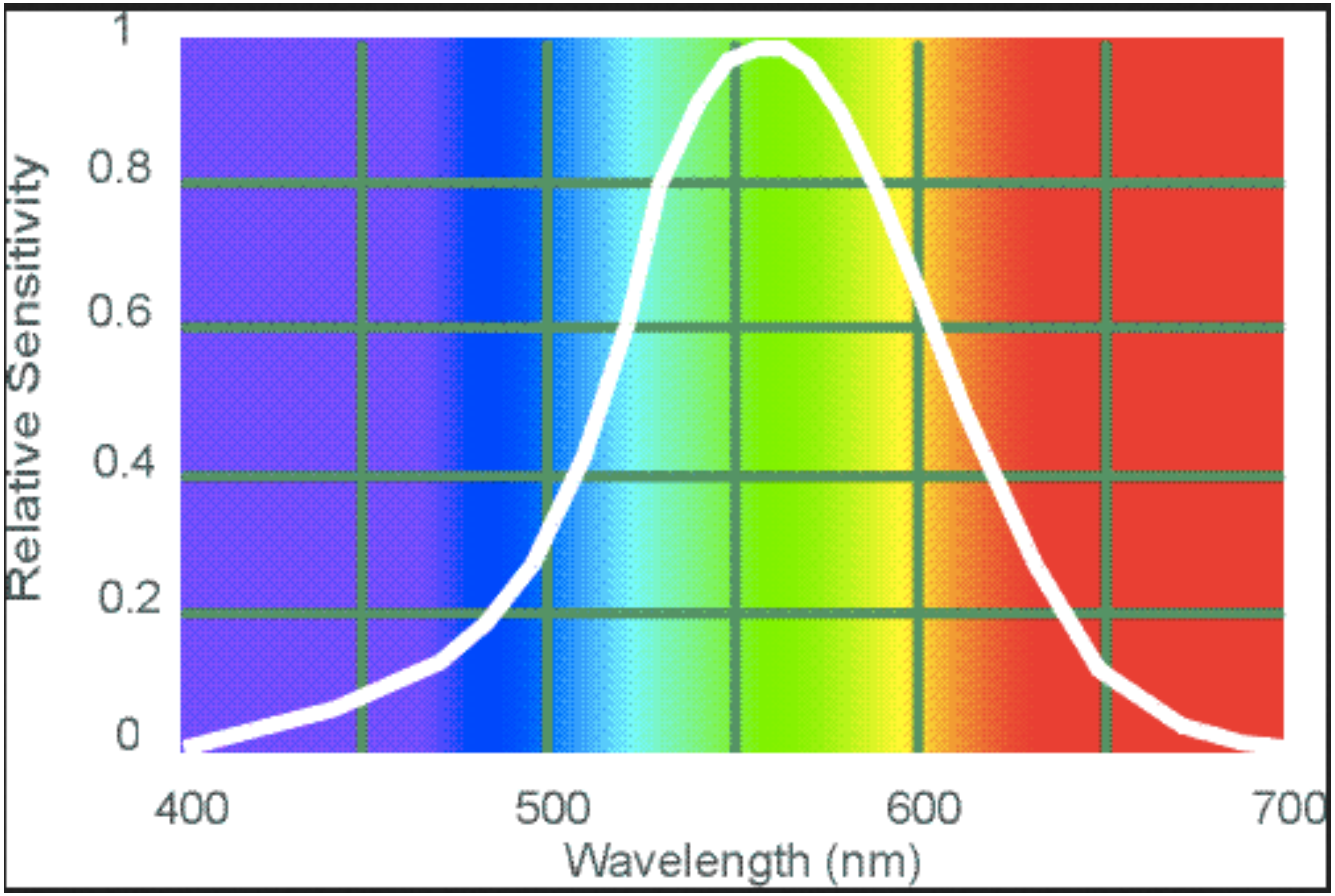
James Hays

Visible Spectrum



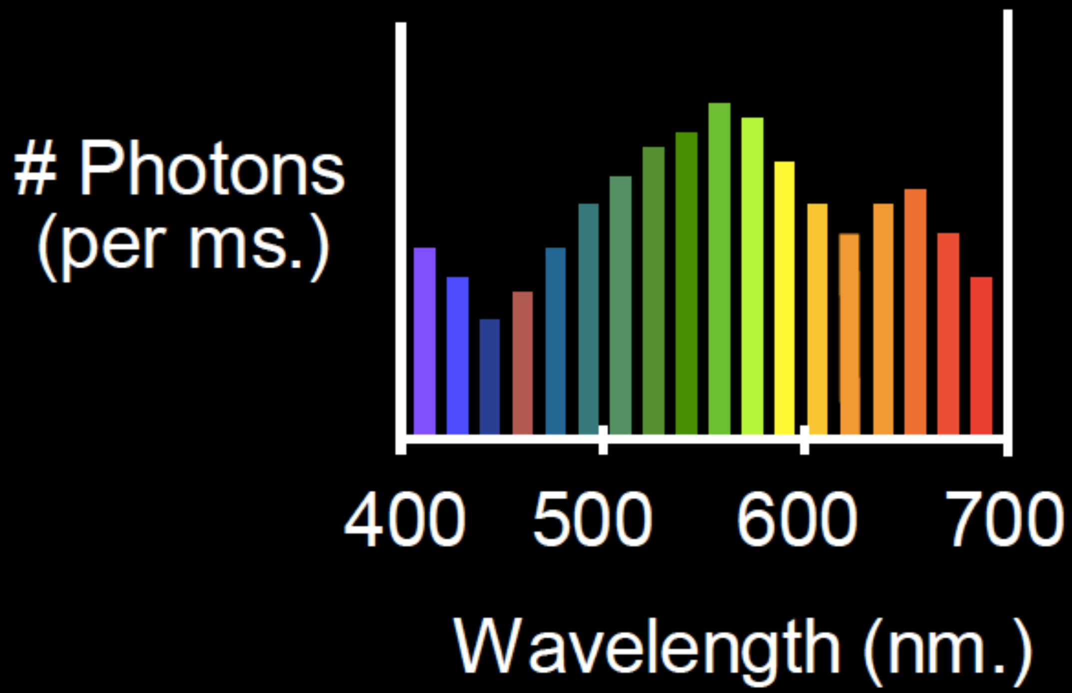
http://www.chromacademy.com/lms/sco736/images/Electromagnetic-spectrum.jpg

Visible Spectrum

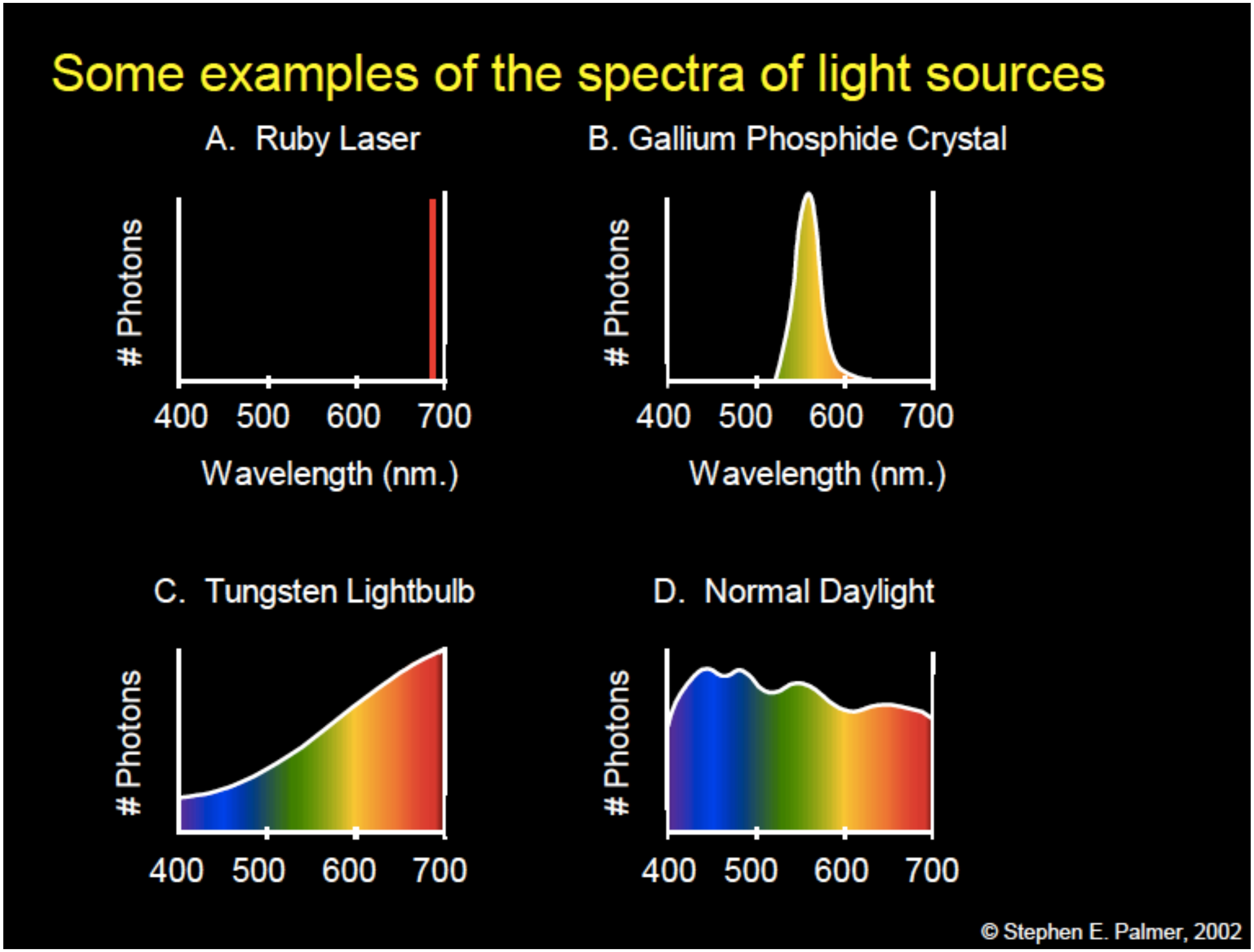


The Physics of Light

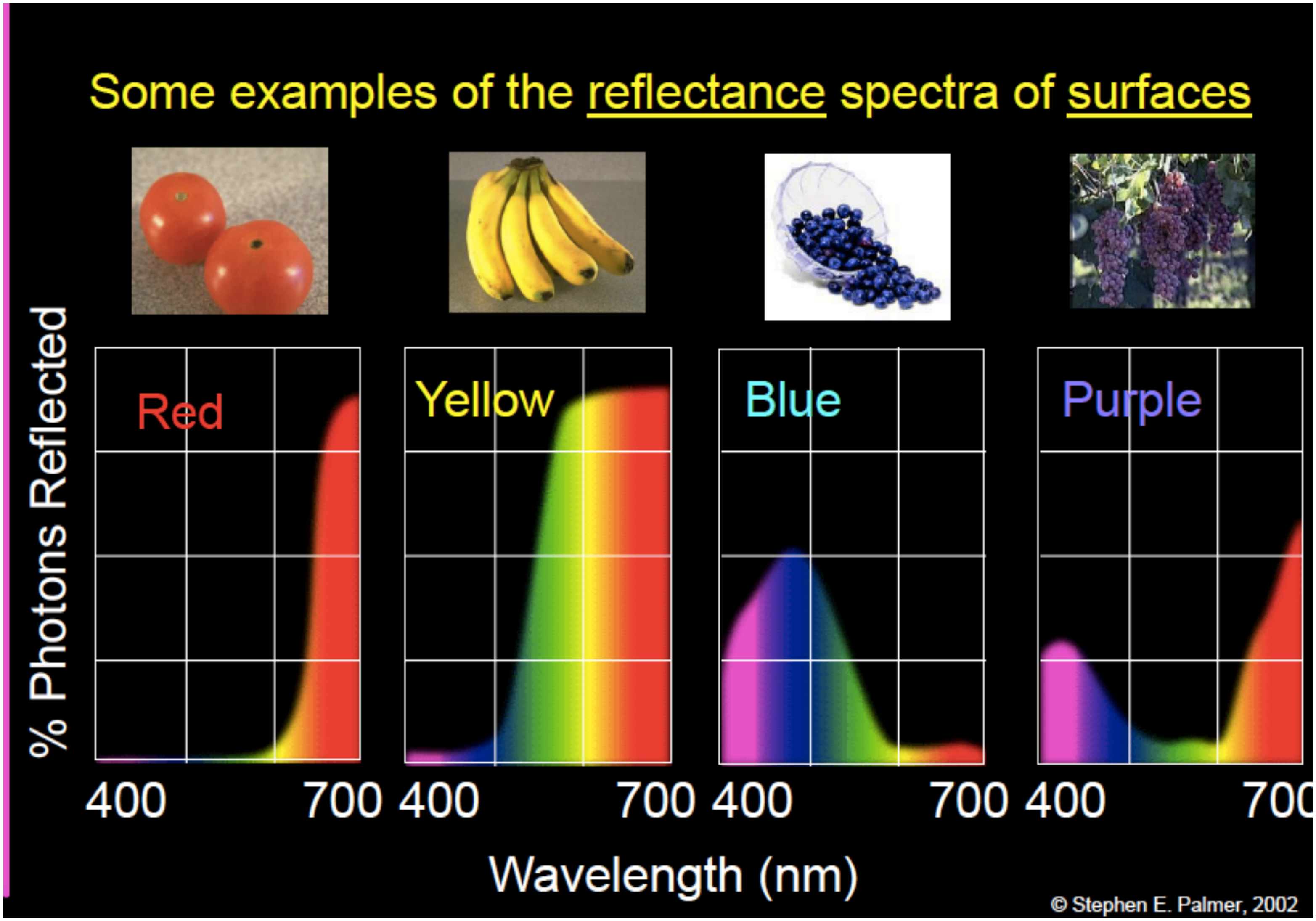
Any patch of light can be completely described physically by its spectrum: the number of photons (per time unit) at each wavelength 400 - 700 nm.



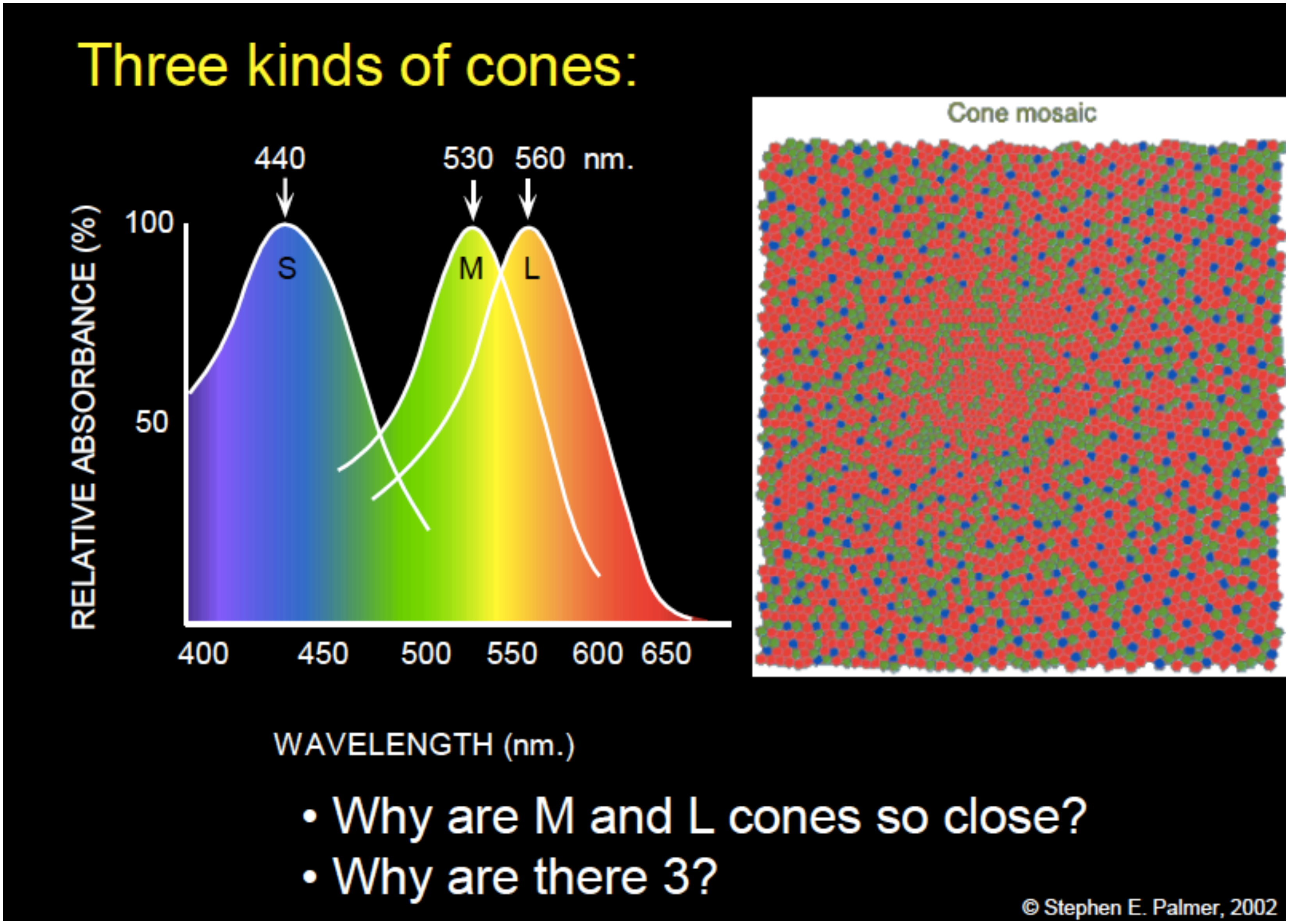
The Physics of Light



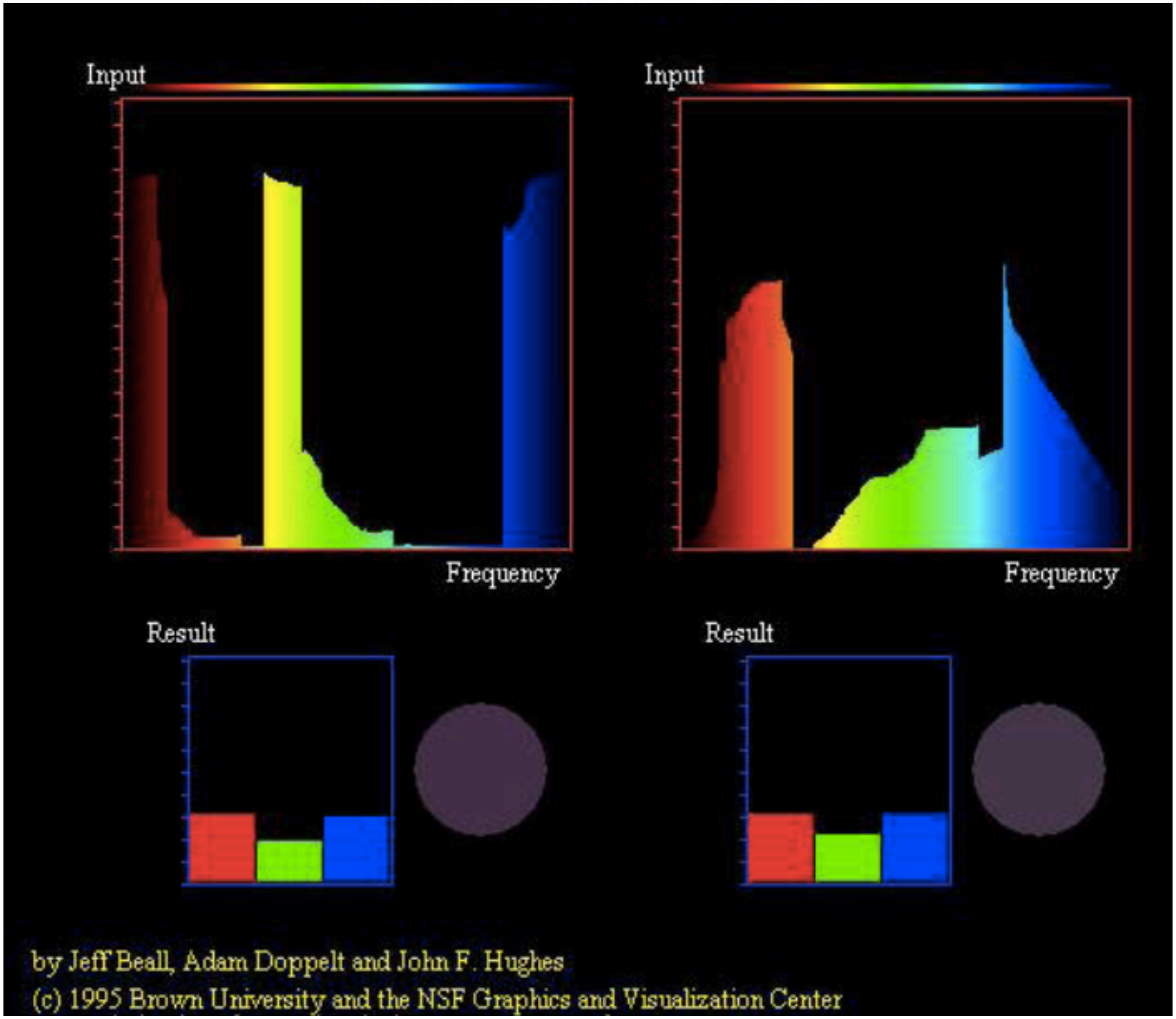
The Physics of Light



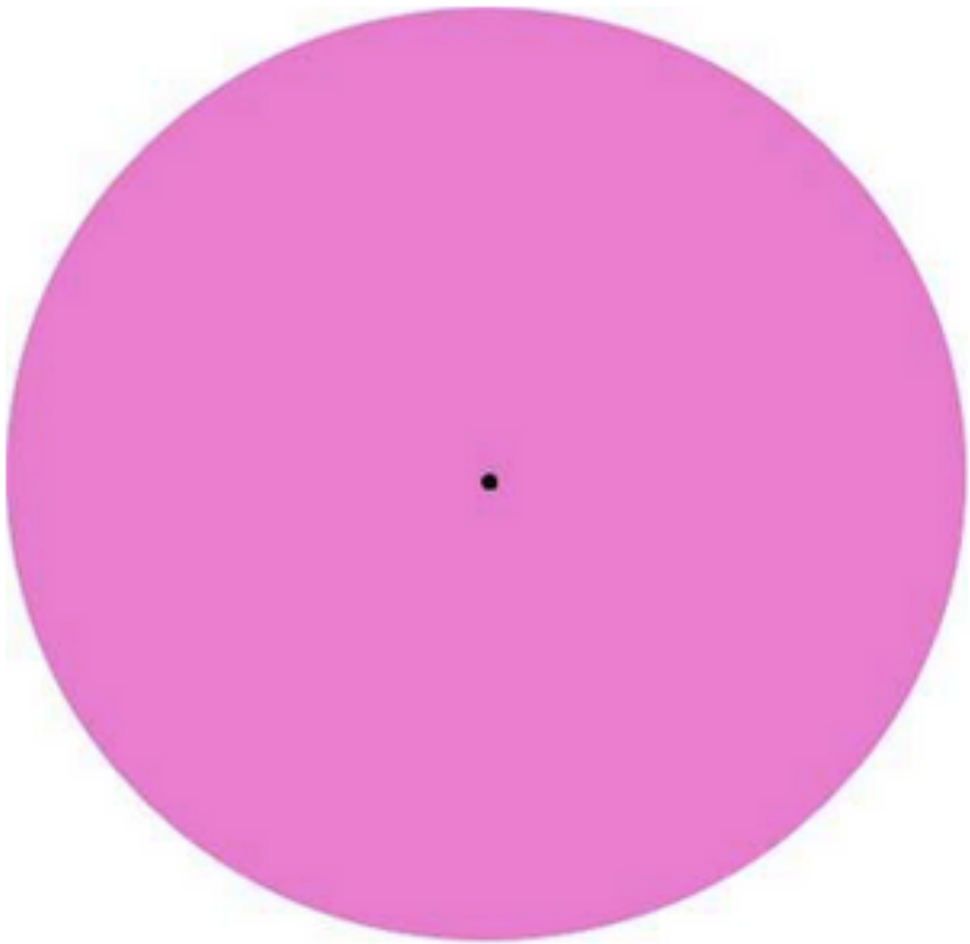
Physiology of Color Vision



Metamers

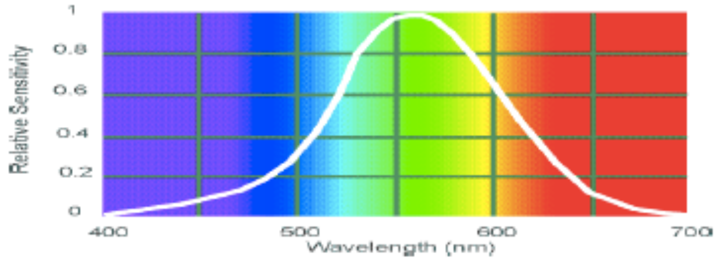

Input
Frequency
Input
Frequency
Result
Result
by Jeff Beall, Adam Doppelt and John F. Hughes
(c) 1995 Brown University and the NSF Graphics and Visualization Center

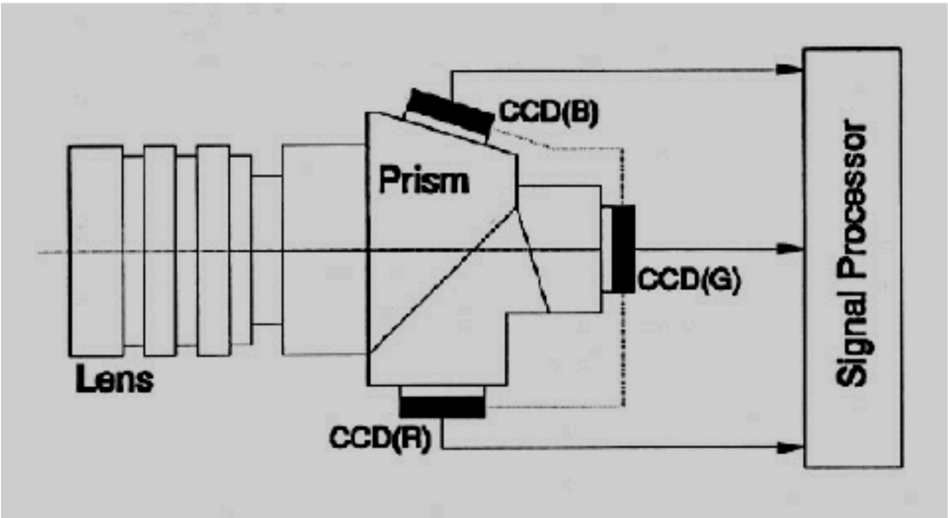
Color Perception

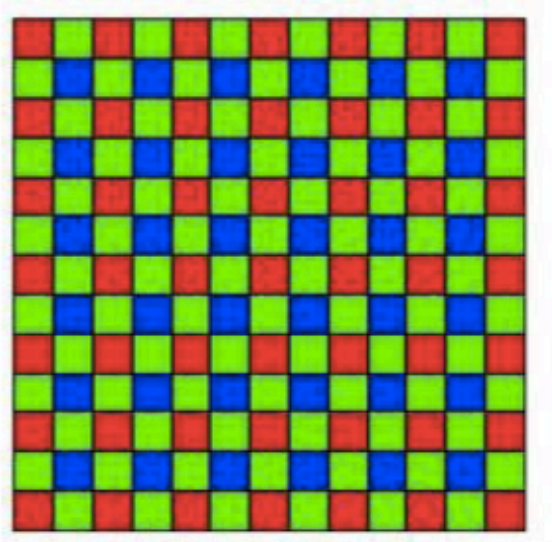


Color Sensing in Camera (RGB)

- 3-chip vs. 1-chip: quality vs. cost
- Why more green?







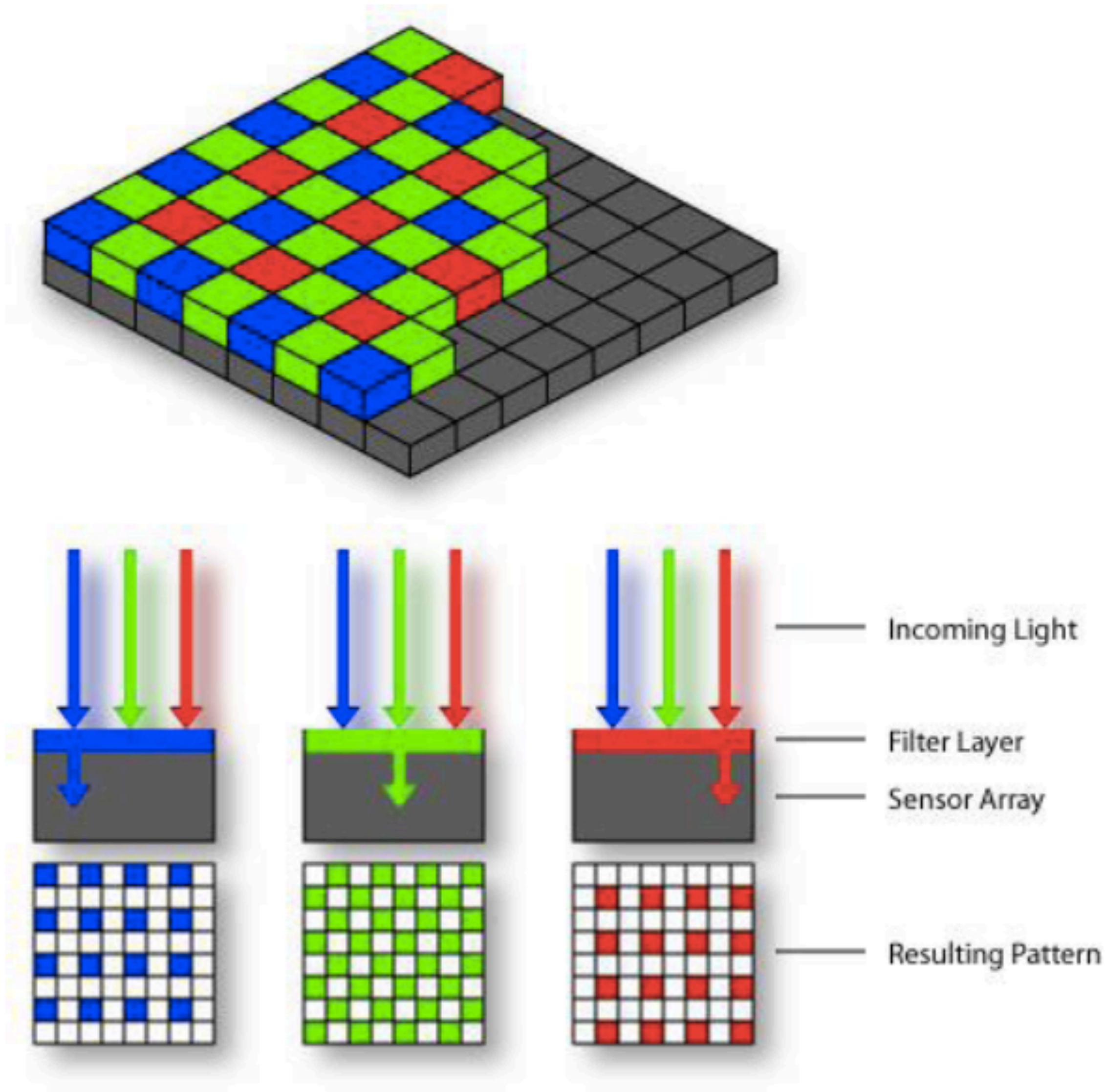
Bayer filter

Why 3 colors?

http://www.cooldictionary.com/words/Bayer-filter.wikipedia

Slide by Steve Seitz

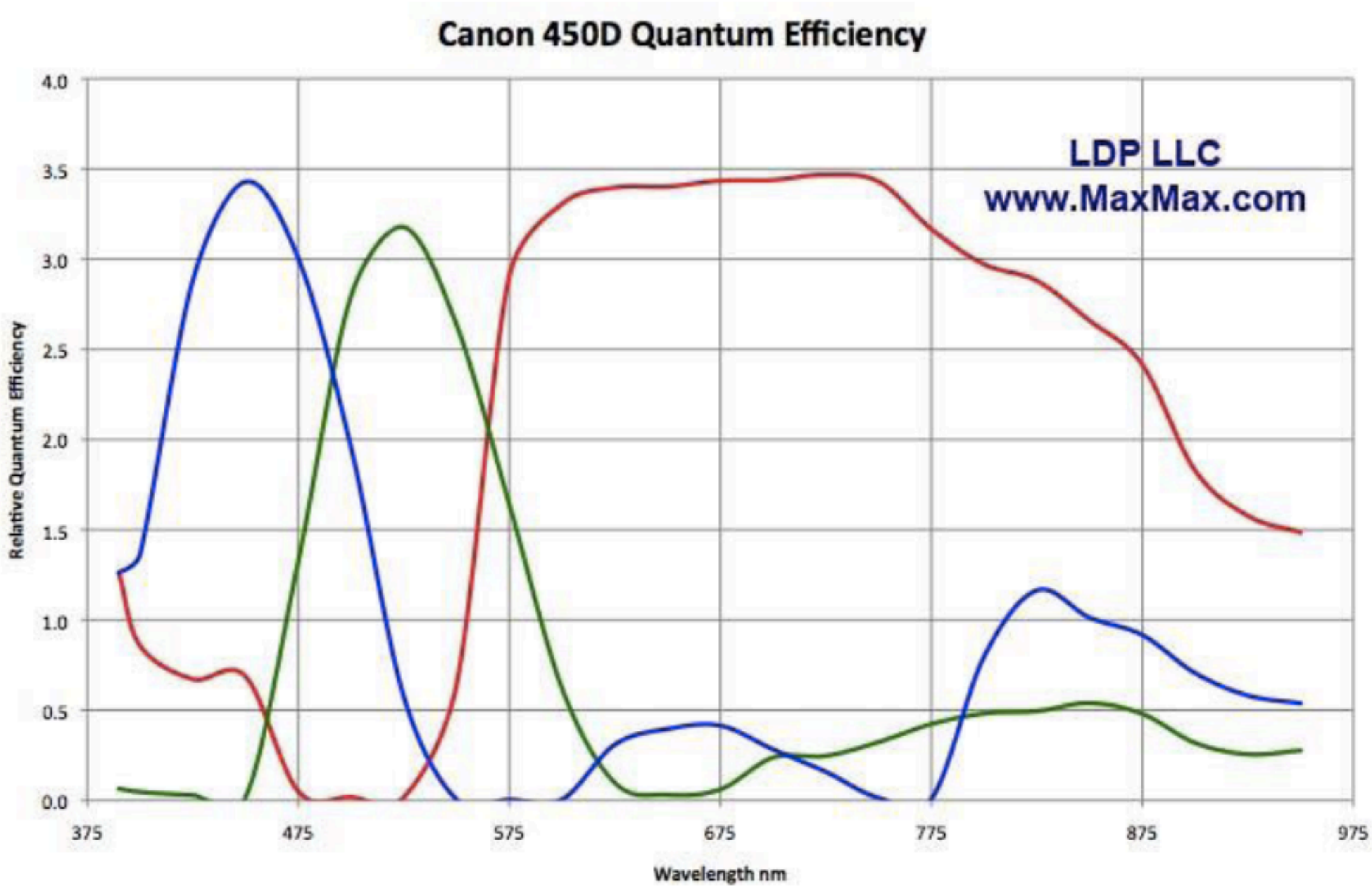
Practical Color Sensing: Bayer Grid



- Estimate RGB at ‘G’ cells from neighboring values

Slide by Steve Seitz

Camera Color Response

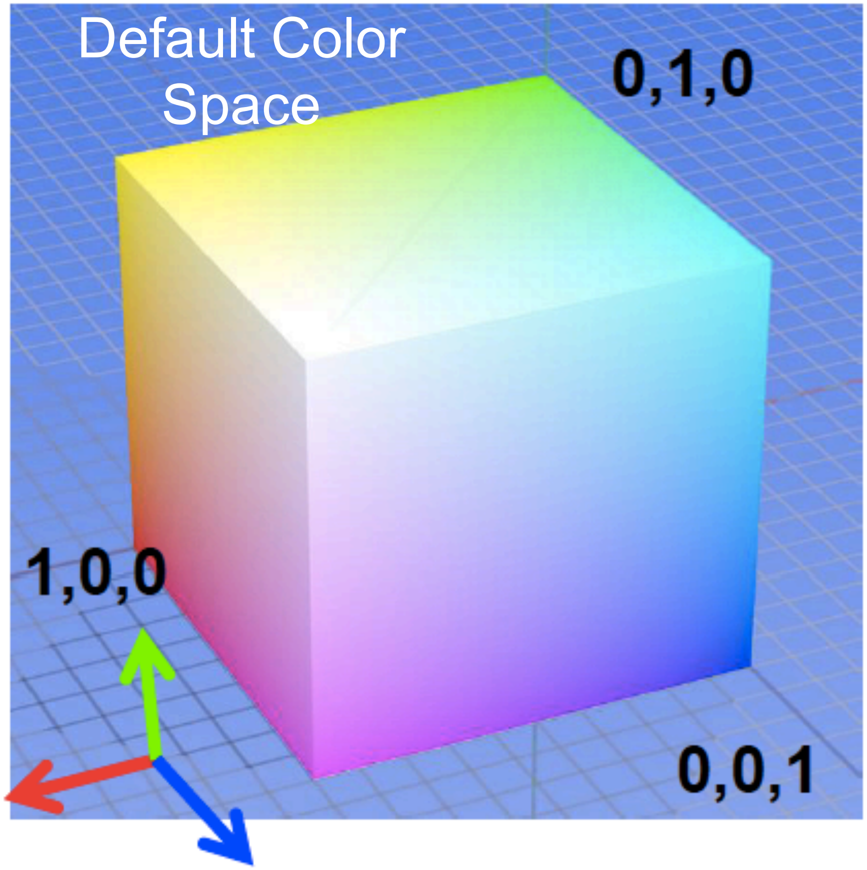


MaxMax.com

Color Space: How can we represent colors

http://en.wikipedia.org/wiki/File:RGB_illumination.jpg

Color Spaces: RGB



R = 1
(G=0,B=0)

G = 1
(R=0,B=0)

B = 1
(R=0,G=0)

Any color = r*R + g*G + b*B
- Strongly correlated channels
- Non-perceptual

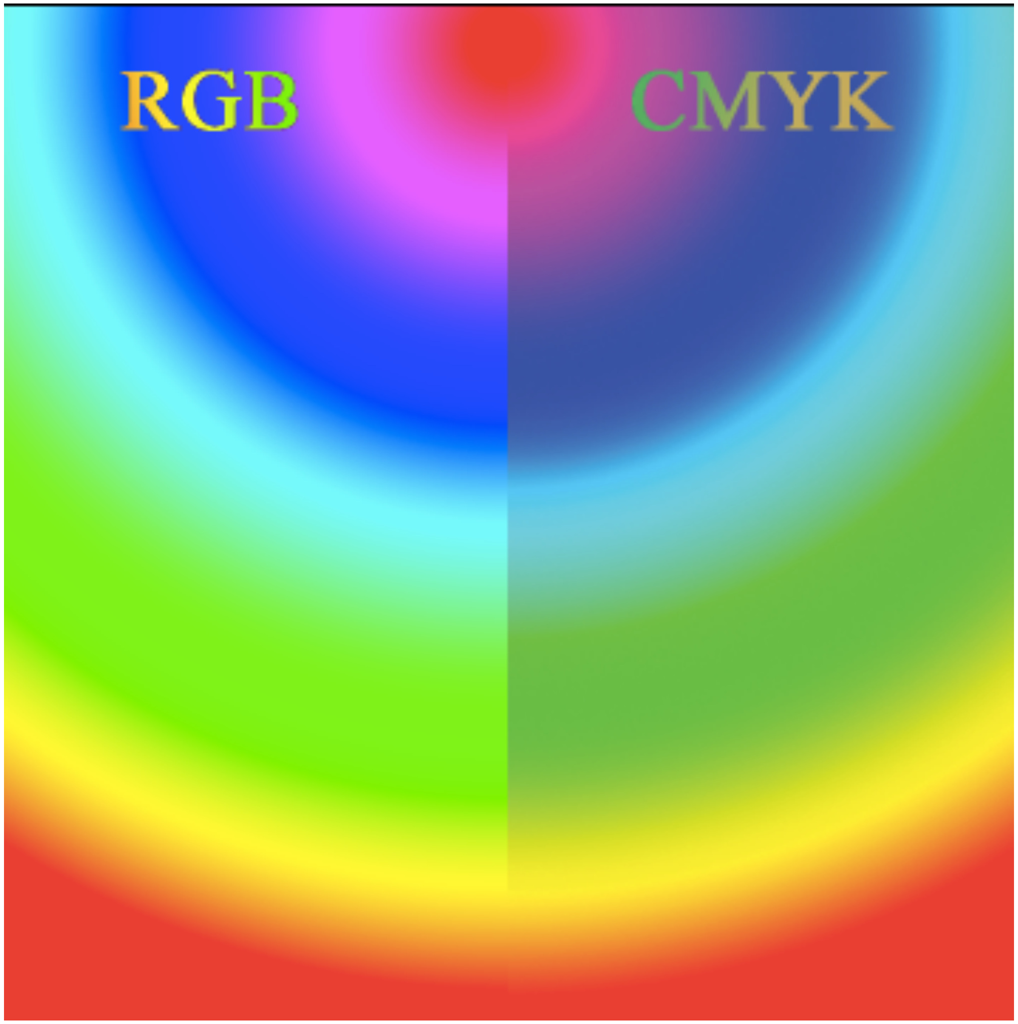
Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Color Space: CMYK

C – Cyan
M – Magenta
Y – Yellow
K -Black

Subtractive primary colors

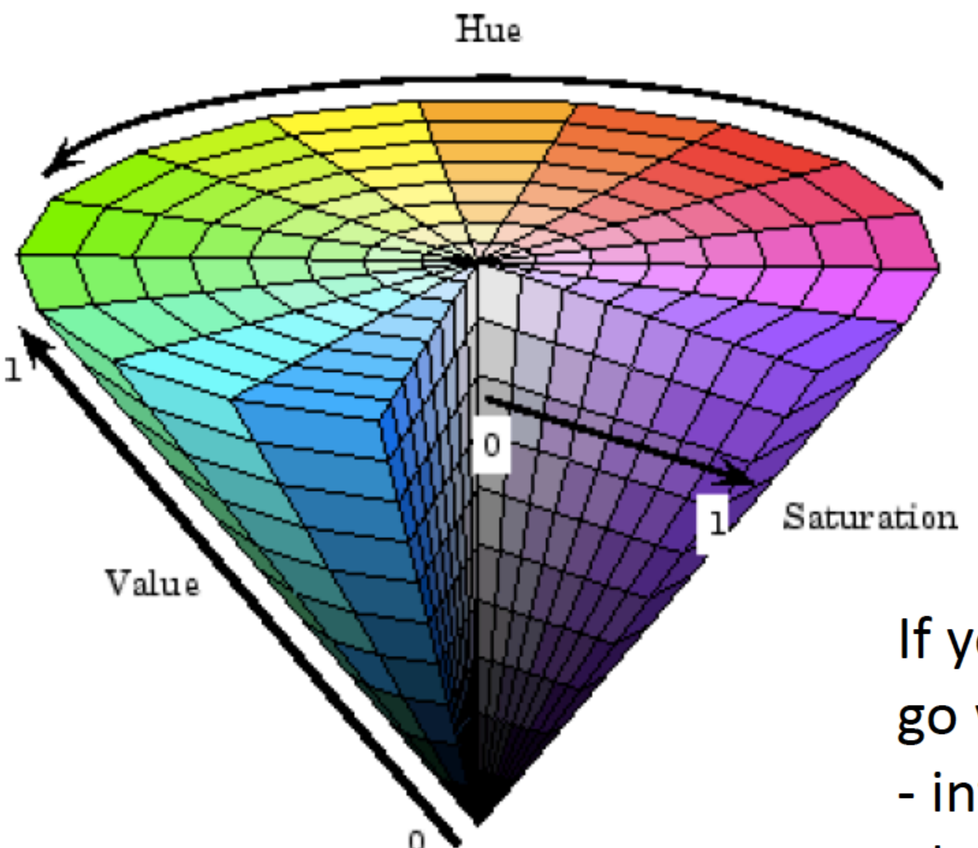
In contrast:
RGB
Additive Primary colors



Color Spaces: HSV

hue, *saturation*, and *value*

Intuitive color space

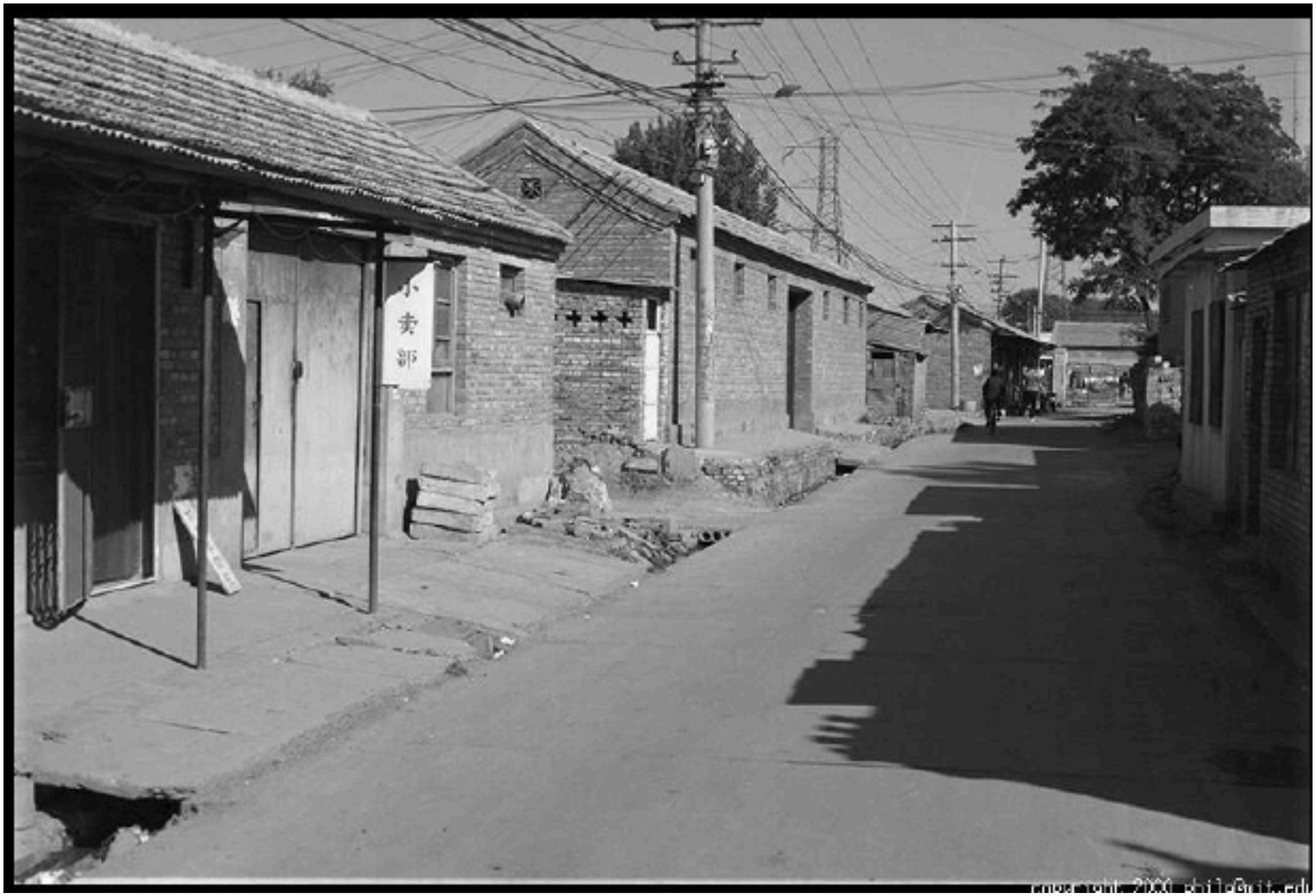


If you had to choose, would you rather go without:
- intensity ('value'), or
- hue + saturation ('chroma')?

Color Spaces: HSV

James Hays

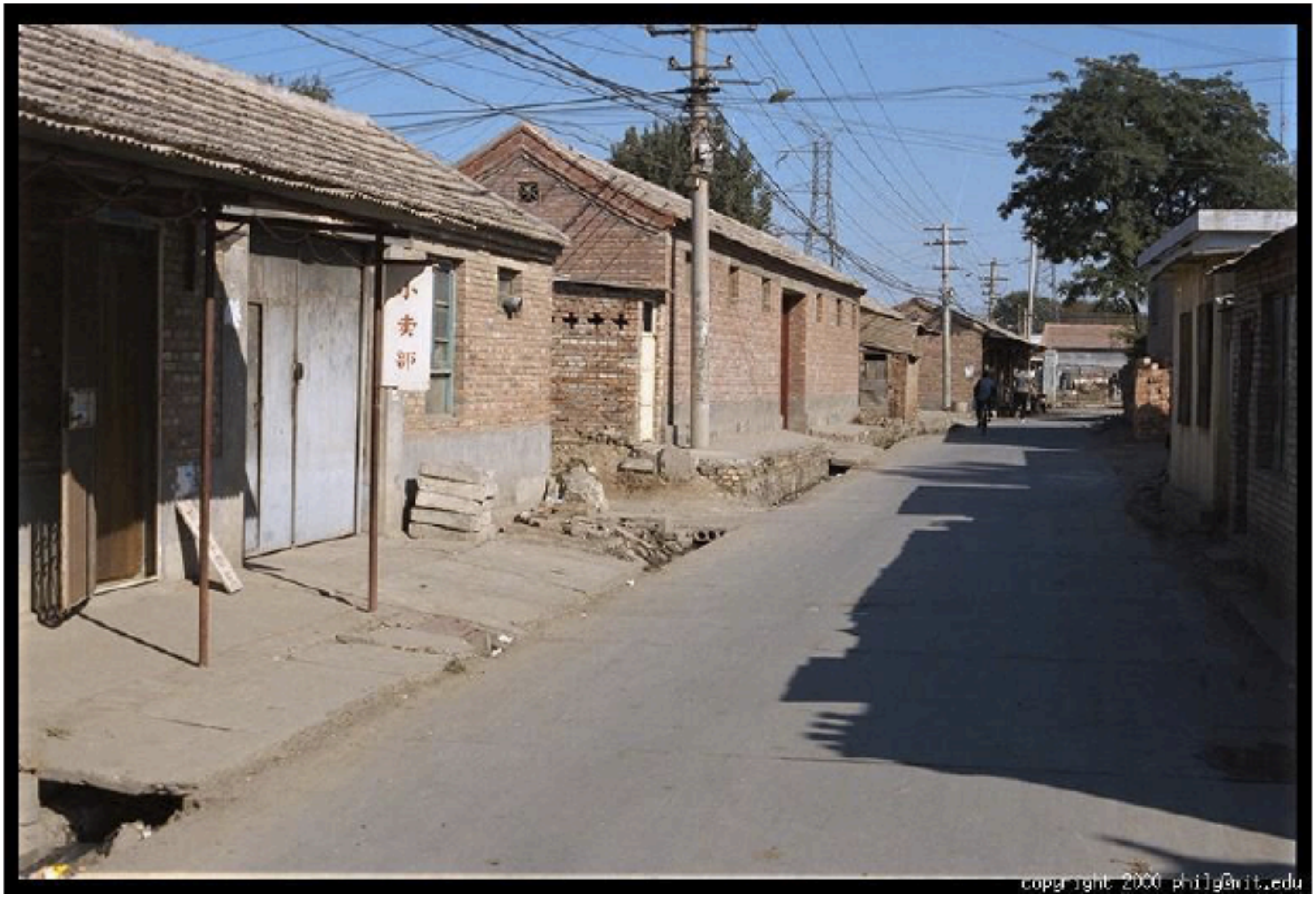
Color Spaces: HSV



Constant Color; Only Intensity

James Hays

Color Spaces: HSV

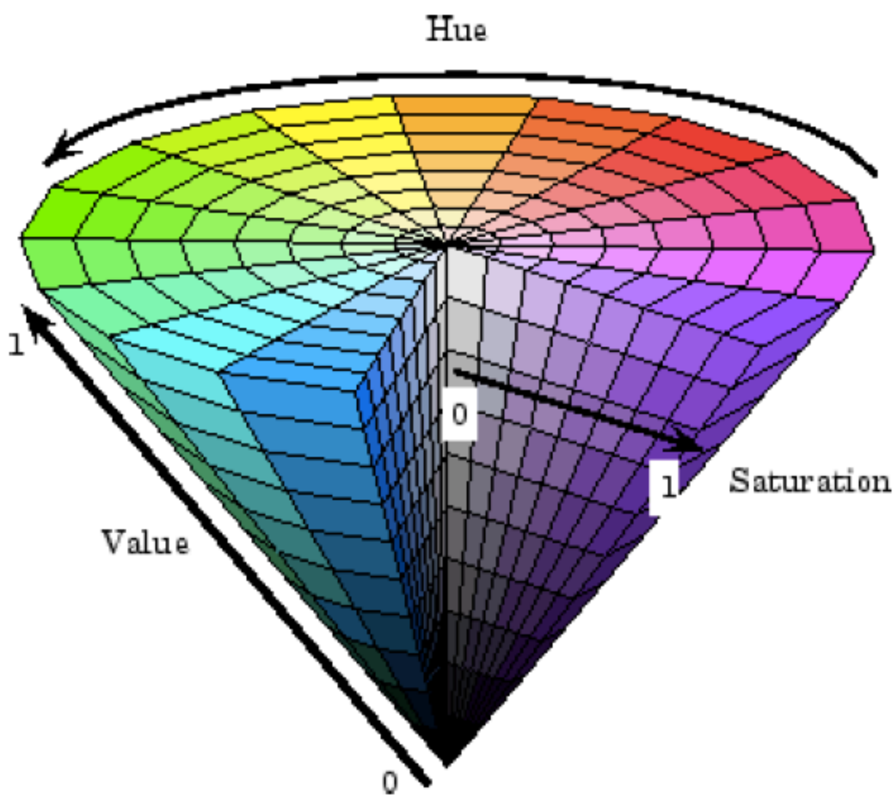


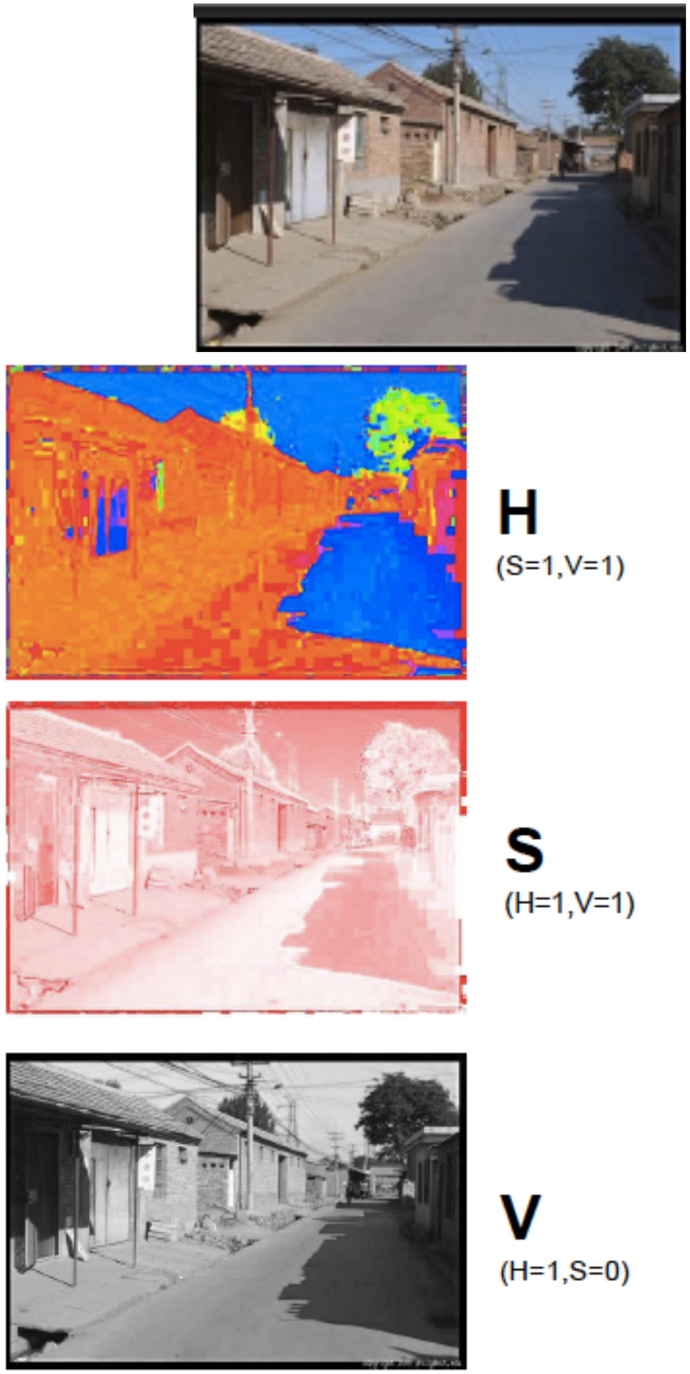
Original Image

James Hays

Color Spaces: HSV

Intuitive color space

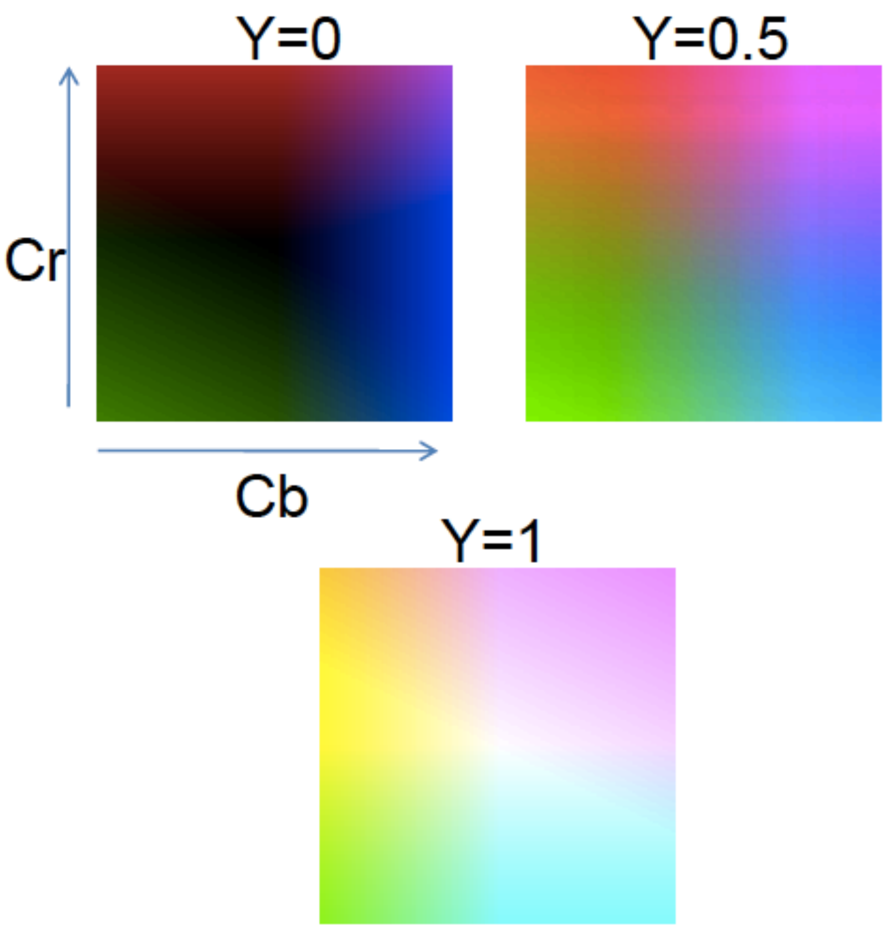


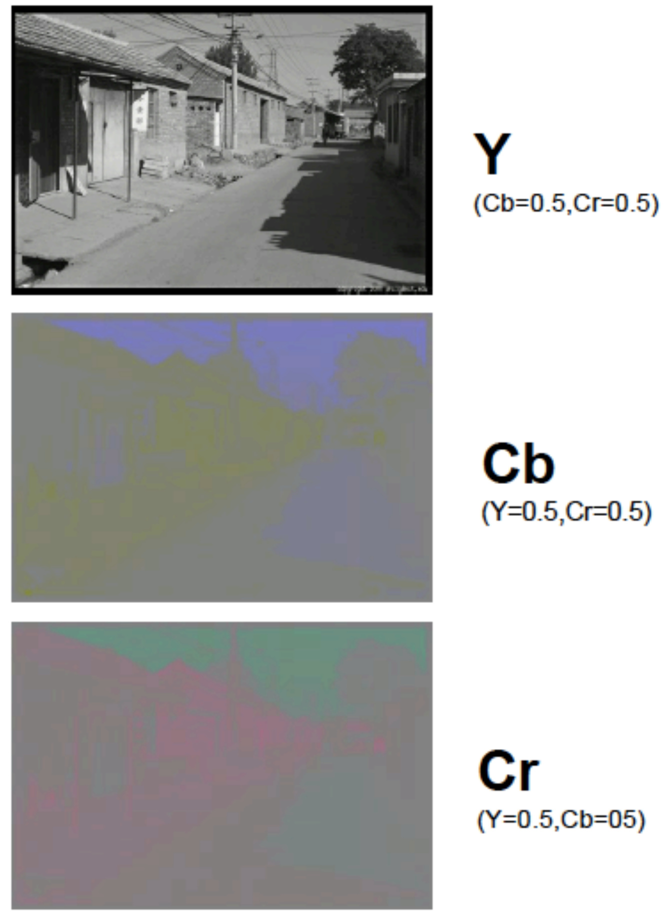


James Hays

Color Spaces: YCbCr

Fast to compute, good for compression, used by TV

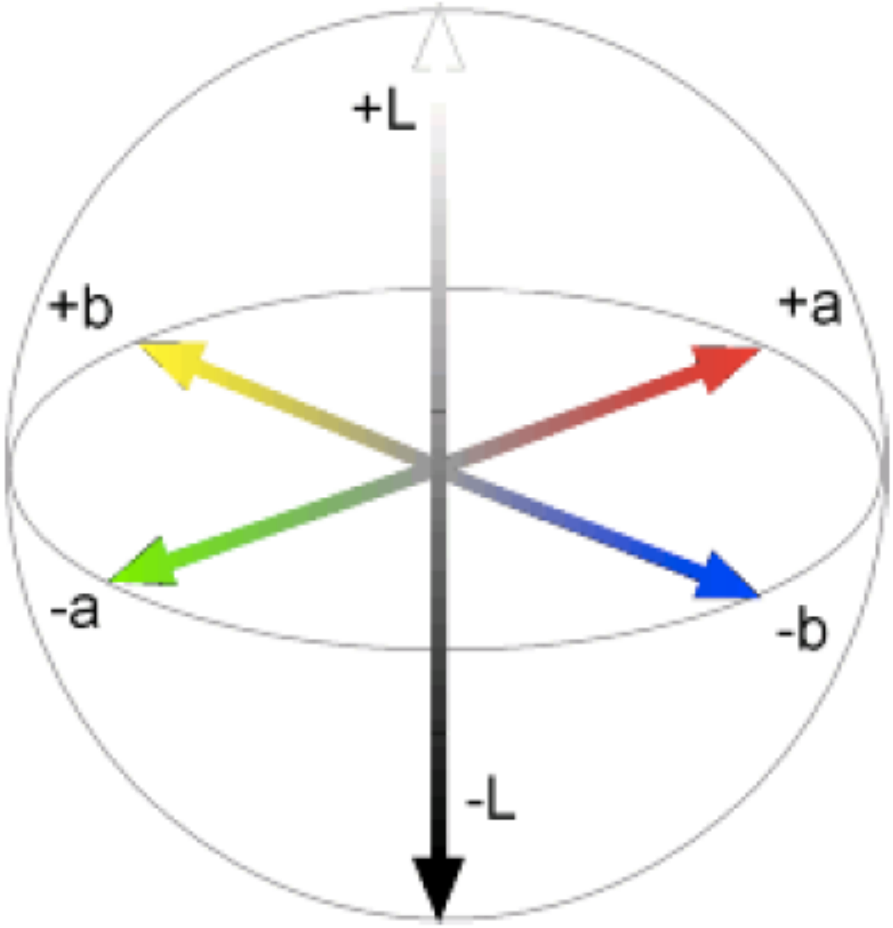




James Hays

Color Spaces: L*a*b*

“Perceptually uniform”* color space



L
(a=0,b=0)

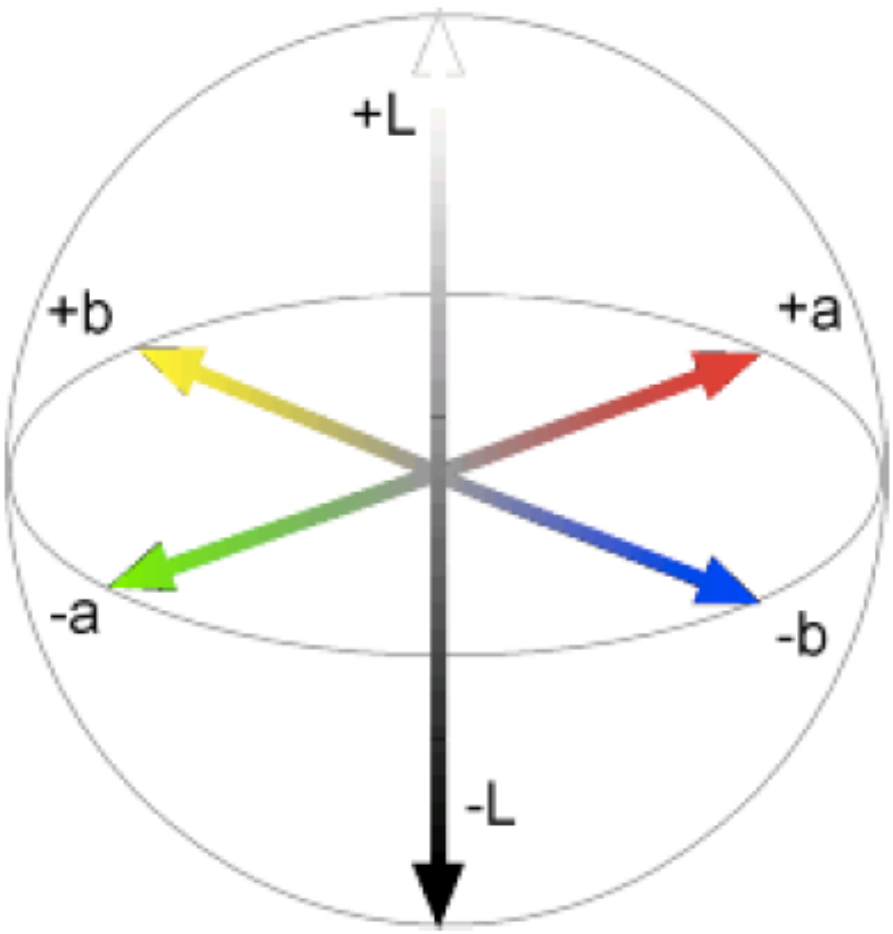
a
(L=65,b=0)

b
(L=65,a=0)

James Hays

Color Spaces: L*a*b*

"Perceptually uniform"* color space



L – Lightness

a,b color opponents

Color

Don't worry Sir, being colour-blind
is not much of a problem around here...

Next class