Computer Vision, Lecture 1
http://www.ugrad.cs.jhu.edu/~cs461

Professor Hager
http://www.cs.jhu.edu/~hager

Outline for Today

- Outline and Organization of the course
- What is Computer Vision
- Some Applications of Computer Vision

What Information is in Images?

What Information is in Images?

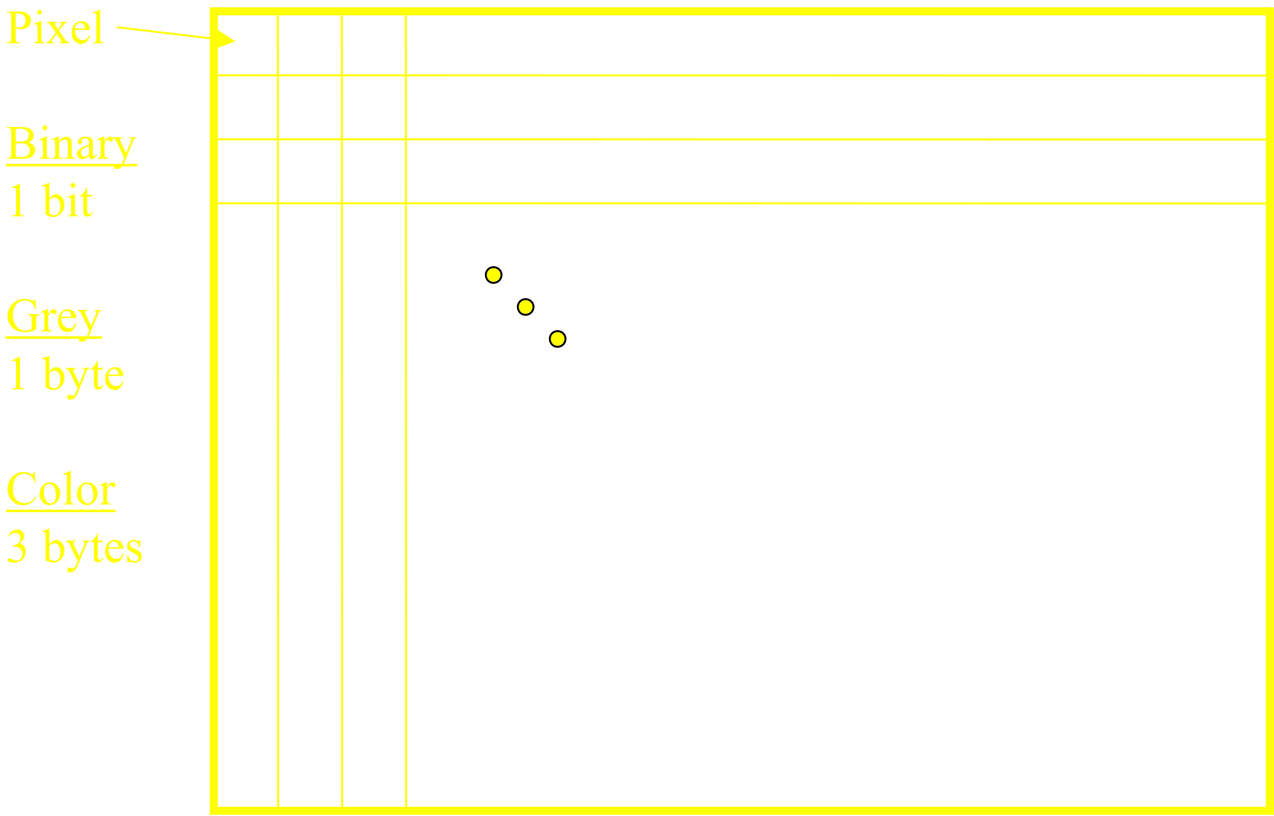
What is Computer Vision?

- **Trucco and Verri**
  - *computing properties of the 3D world from one or more digital image*
- **Stockman and Shapiro**
  - *To make useful decisions about real physical objects and scenes based on sensed images*
- **Ballard and Brown**
  - *The construction of explicit, meaningful description of physical objects from images*

Some Related Terms

- **Image Processing:** the study of the properties of operators that produce images from other images
  - we will touch on image filtering and related operators from image processing
- **Machine Vision:** a somewhat outdated term which now tends to refer to industrial vision applications where (usually) a single camera is used to solve a structured inspection task
  - the "reverse CAD" model
- **Pattern Recognition:** typically refers to the recognition of structures in 2D images (usually without reference to any underlying 3D information).
- **Photogrammetry:** the science of measurement though non-contact sensing, e.g. terrain maps from satellite images. Usually is more focused on accuracy issues than interpretation.

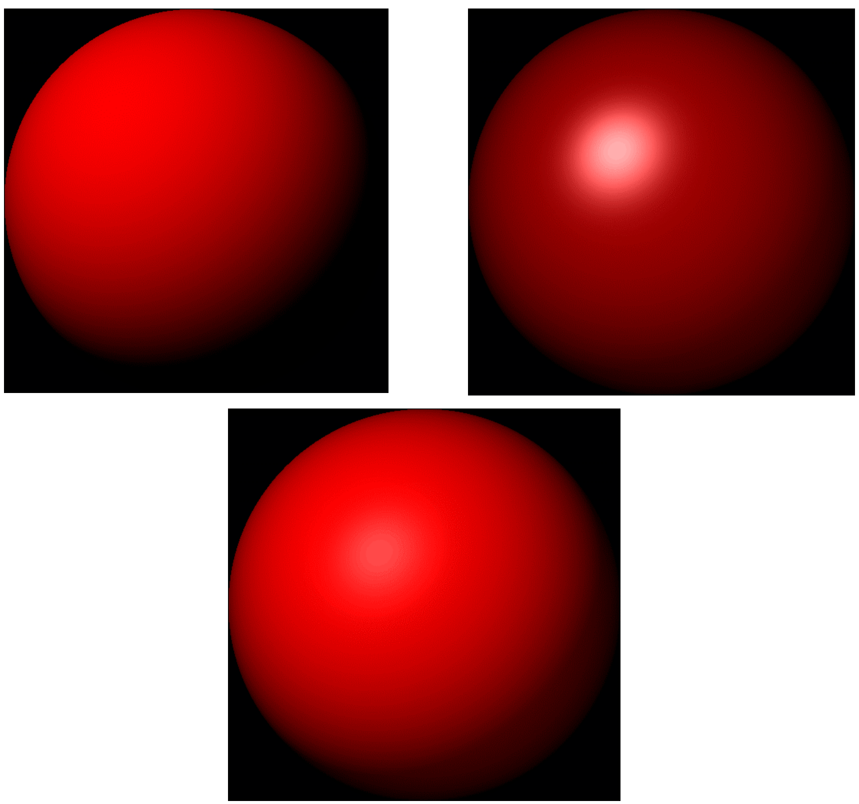
Our Data Structure

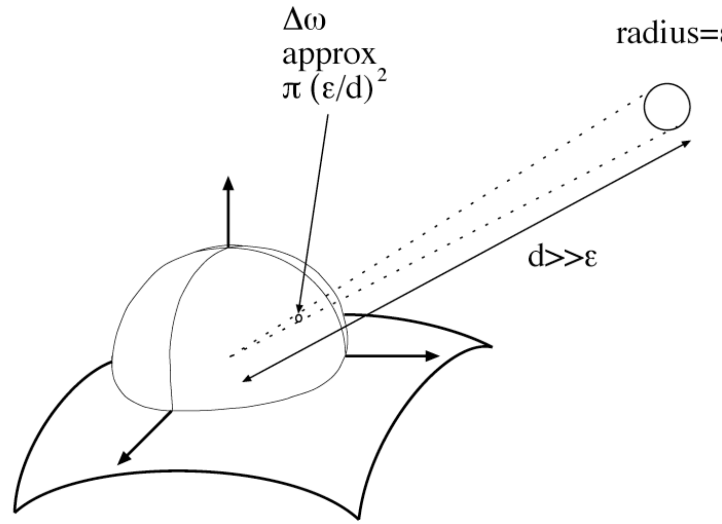


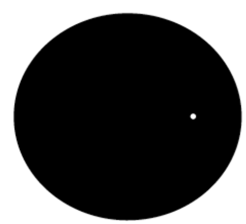
Each pixel is a measure of the brightness (intensity of light) that falls on an area of an sensor (typically a CCD chip)

Problems of Computer Vision: Modeling

What are the physical and geometric processes that govern (digital) imaging?

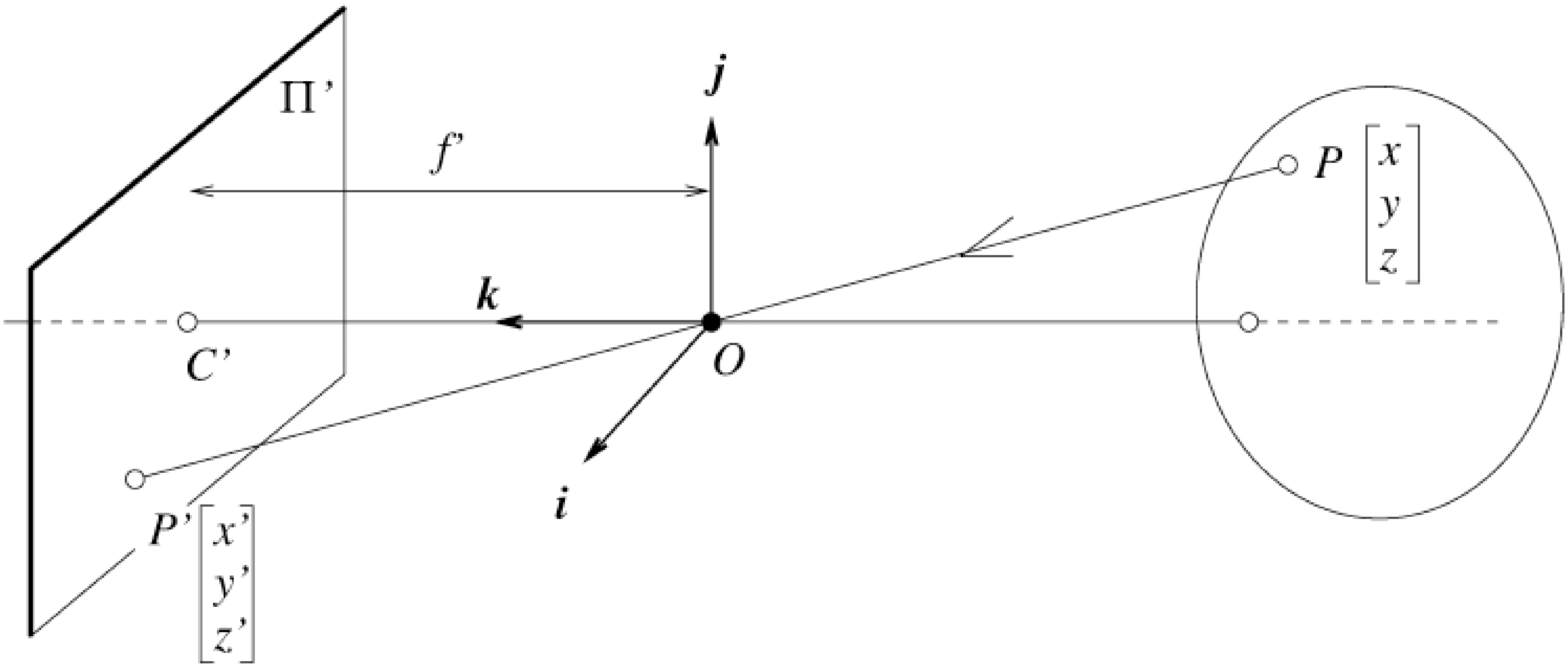






Problems of Computer Vision: Modeling

What are the physical and geometric processes that govern (digital) imaging?



General Rules

If you can't understand (i.e. model) the forward process, you will have a hard time solving the inverse!

A related point: the best way to test vision algorithms is always to implement the forward model to test the (inverse) solution.

Computer Vision vs. Graphics

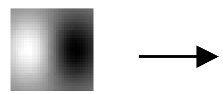
Is Vision the "Inverse" of Graphics?

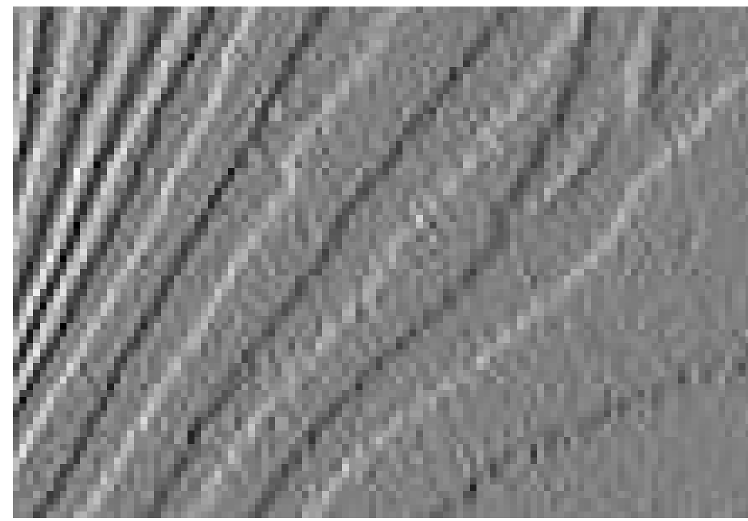
- Computer Graphics
  - Produce "plausible" images
  - You choose the models, conditions, imaging parameters, etc.

- Computer Vision
  - Given real images with noise, sampling artifacts …
  - Estimate physically quantities
  - Ill-posed ---- what is the minimum world knowledge we need?

Problems of Computer Vision: Feature Extraction

What are the “informative” areas of an image and how do we detect them?



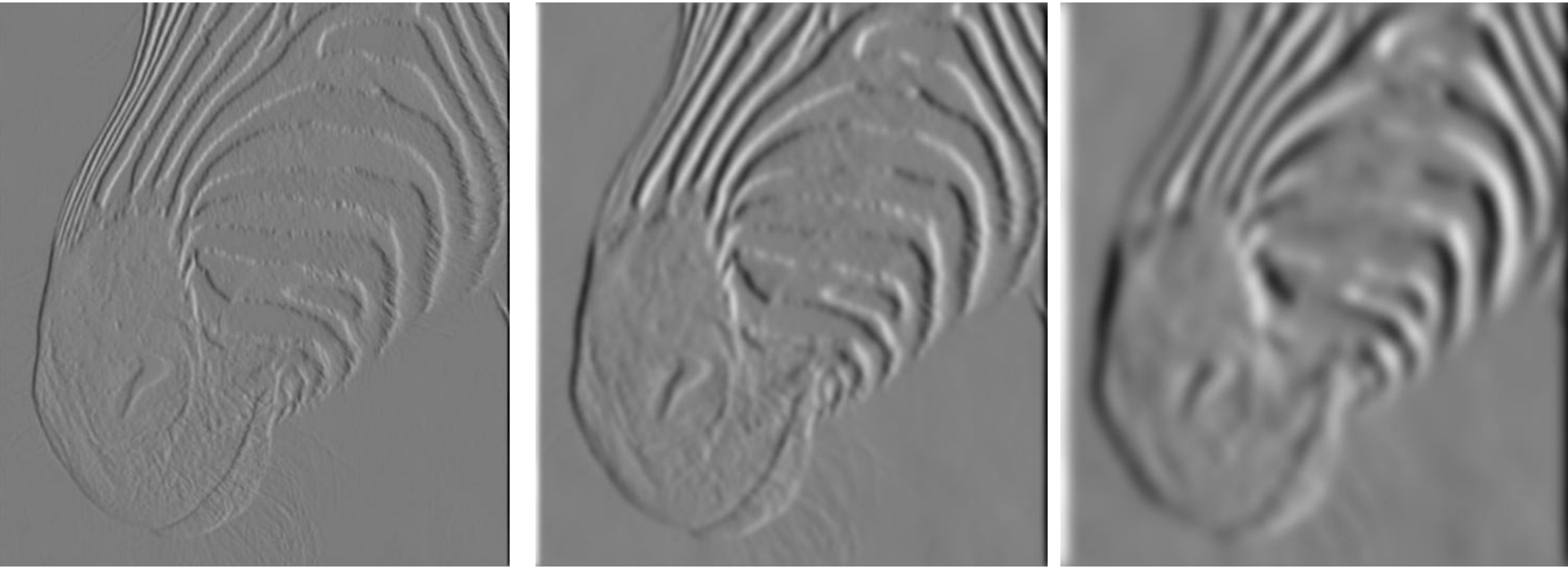


Image

Filter

Result

Problems of Computer Vision: Feature Extraction



Filter kernels that are larger see effects at coarser scales -- the filter on the left responds to the zebra's whiskers, that on the right to its stripes

Problems of Computer Vision: Feature Extraction

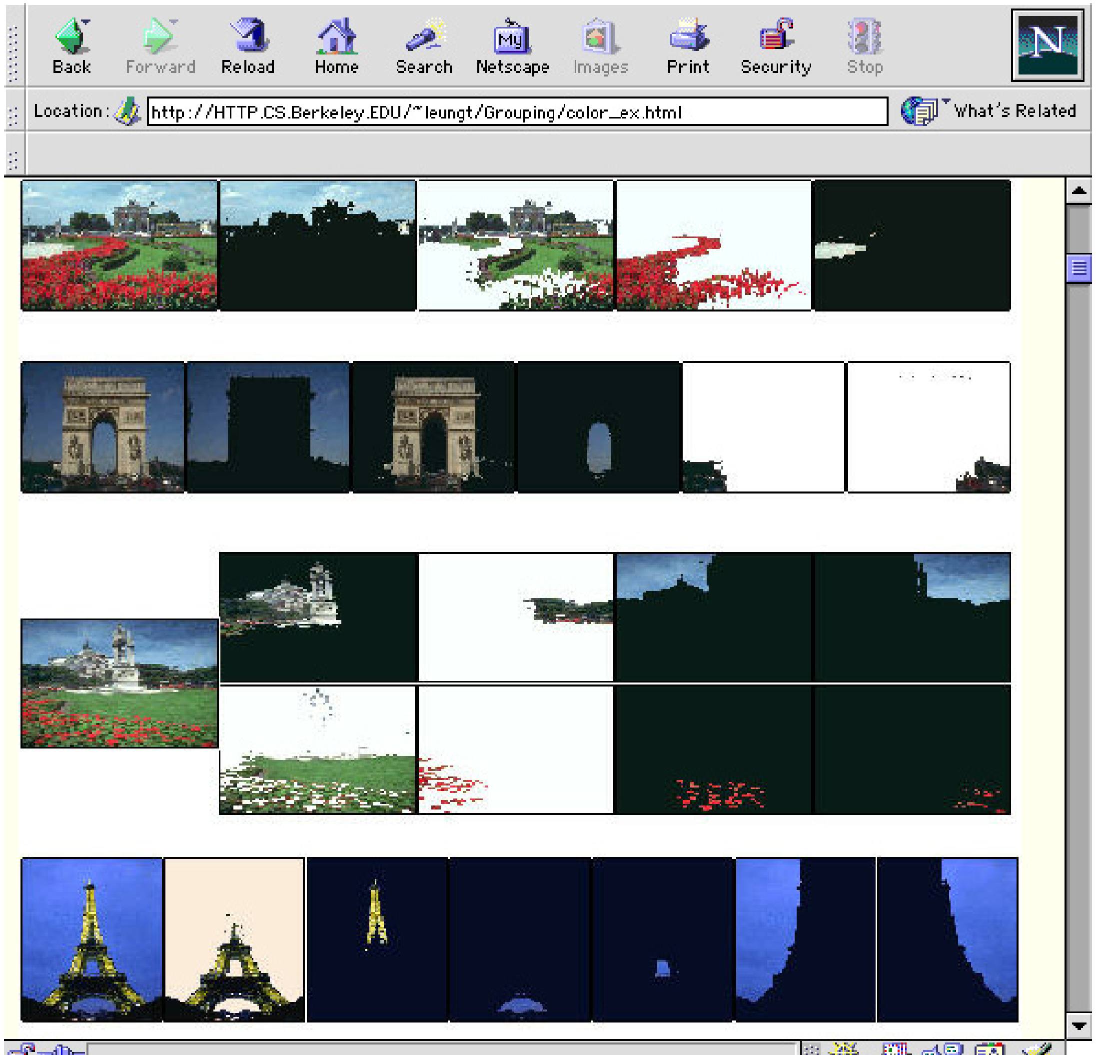
Thresholding suppresses “non-feature” areas of the image

Computer Vision vs. Image Processing

- Image Processing
  - Mostly concerned with *image-to-image* transformations
    - Filtering
    - Enhancement
    - Compression

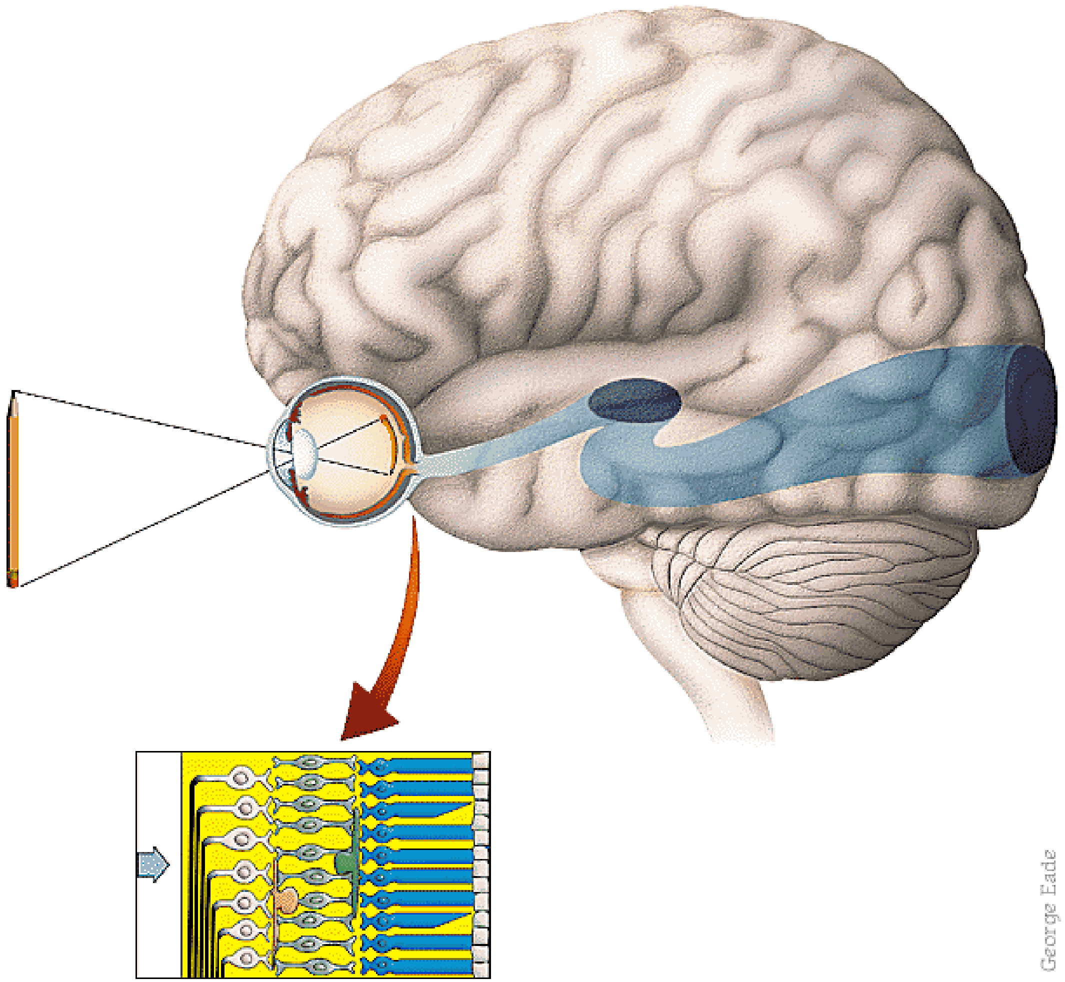
- Computer Vision
  - Concerned with how images *reflect the 3D world*
    - Filtering *for feature extraction*
    - Enhancement *for recognition/detection*
    - Compression *that preserves geometric information in images*

Problems of Computer Vision: Segmentation and Grouping



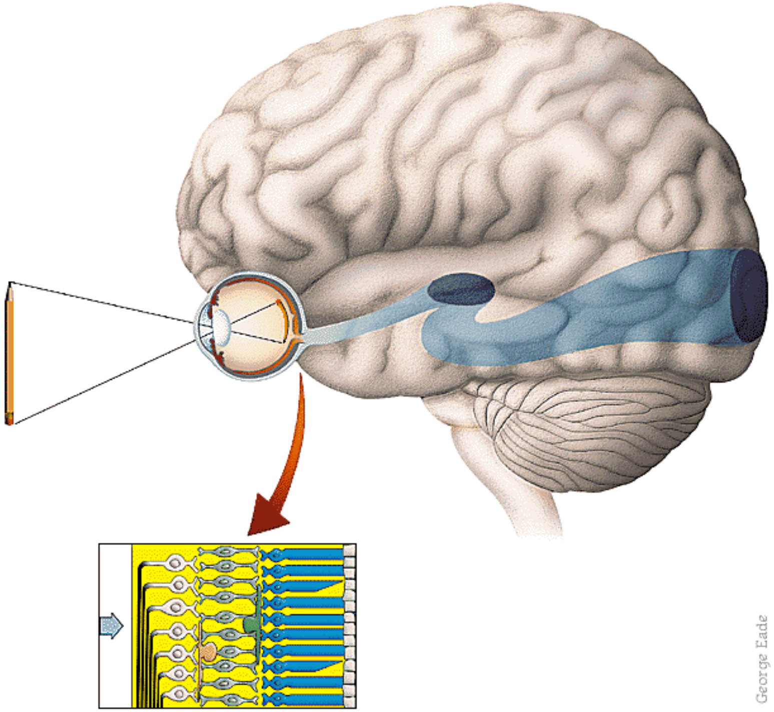
What portions of an image pertain to one another and to relevant physical phenomena?

Computer Vision vs. Human Vision



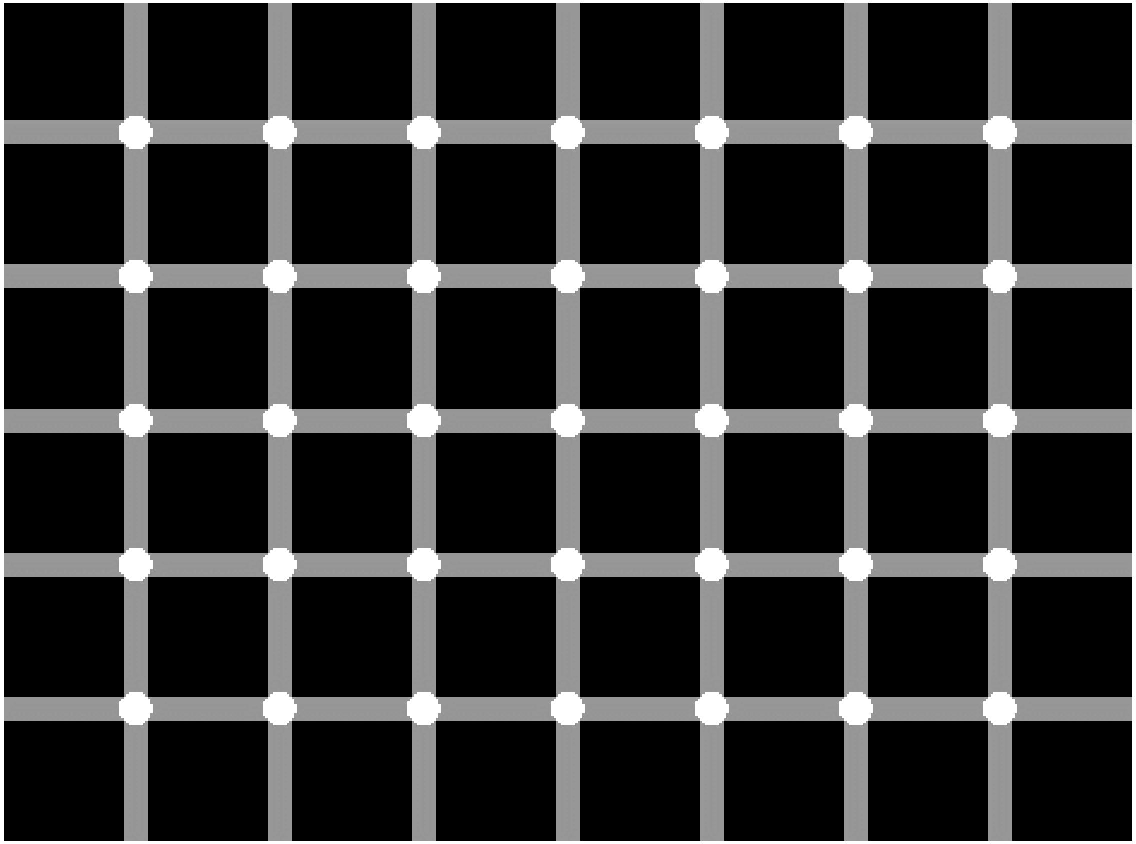
What is the right segmentation? To us it seems obvious …

Objective Reality vs. Subjective Reality

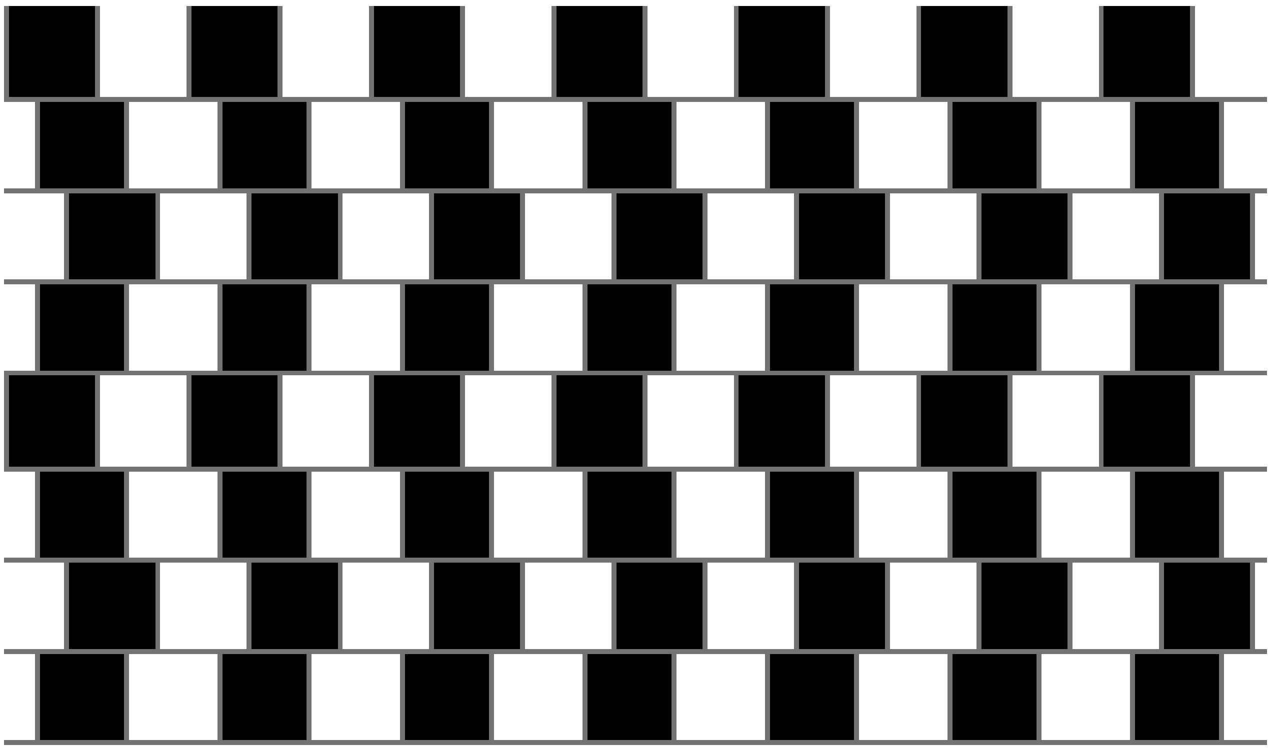


Metric Geometry vs. "Shape"
Symmetry
The color orange
......

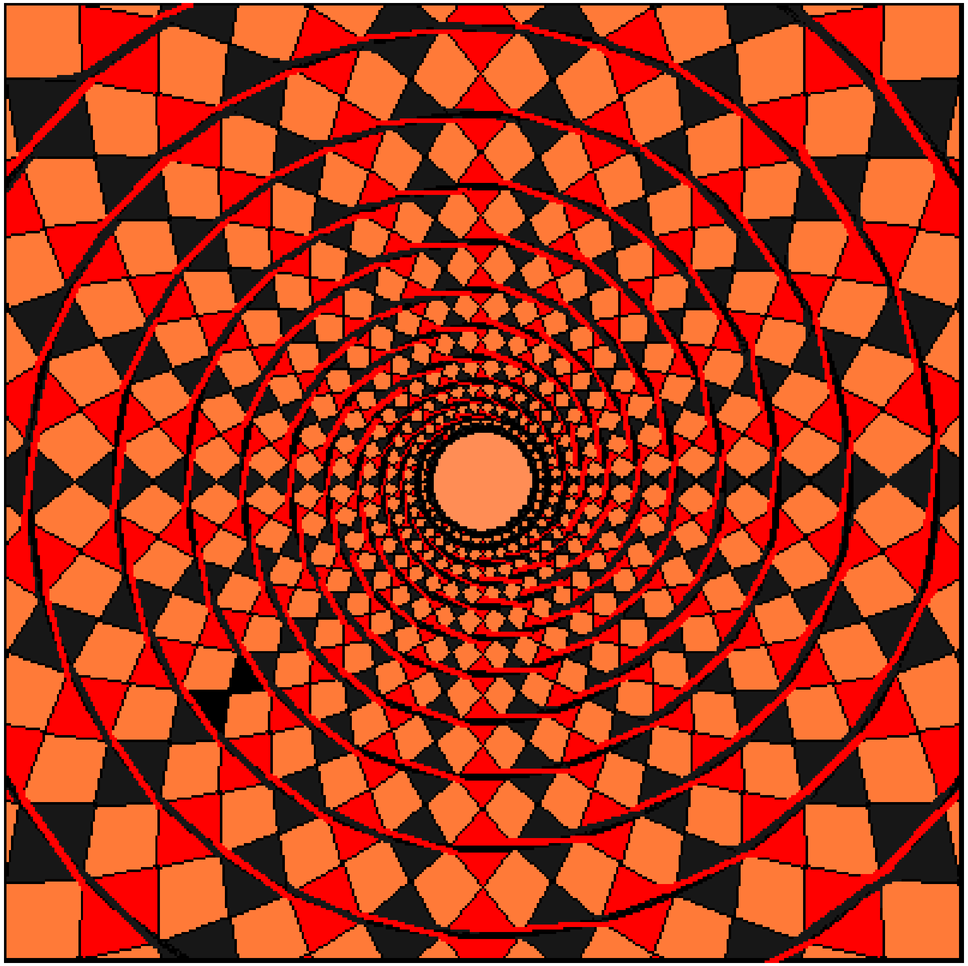
Illusions: What Do They Tell Us?



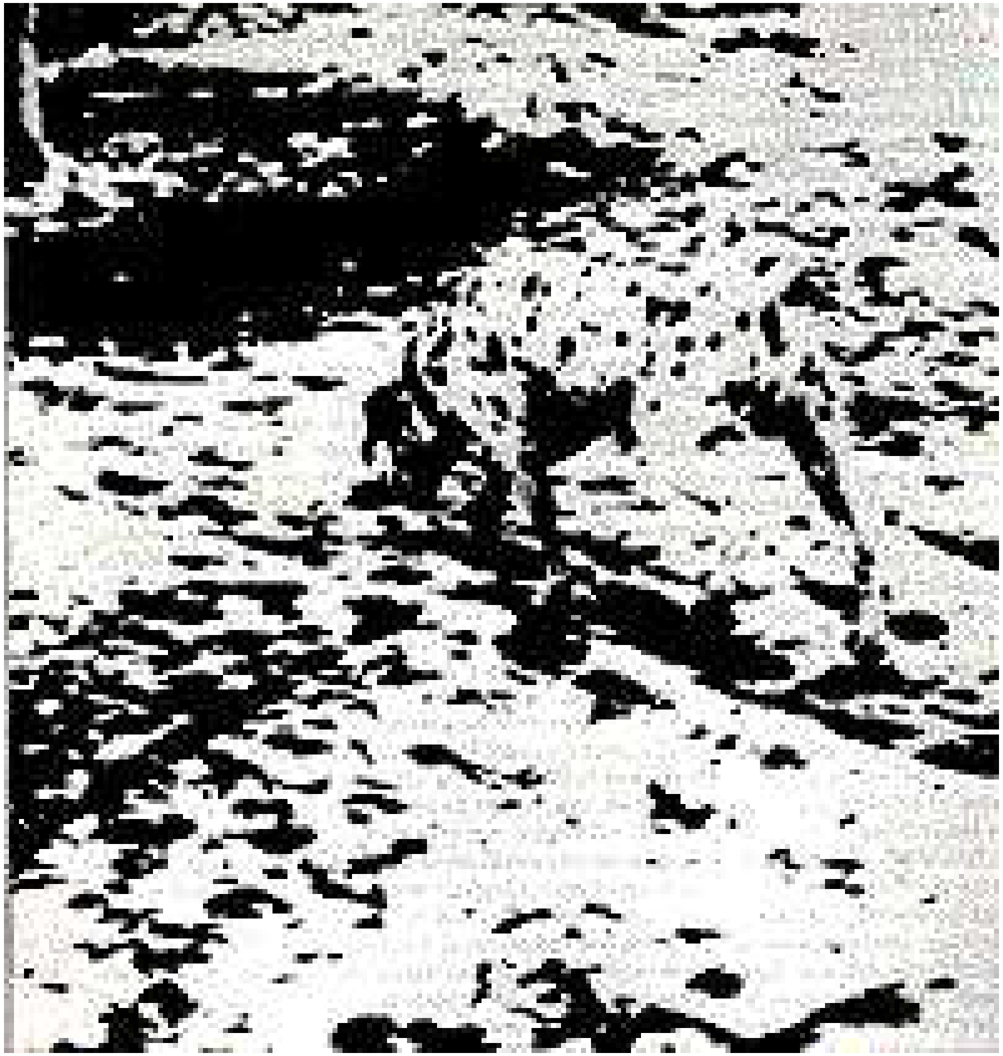
Illusions: What Do They Tell Us?



Illusions: What Do They Tell Us?

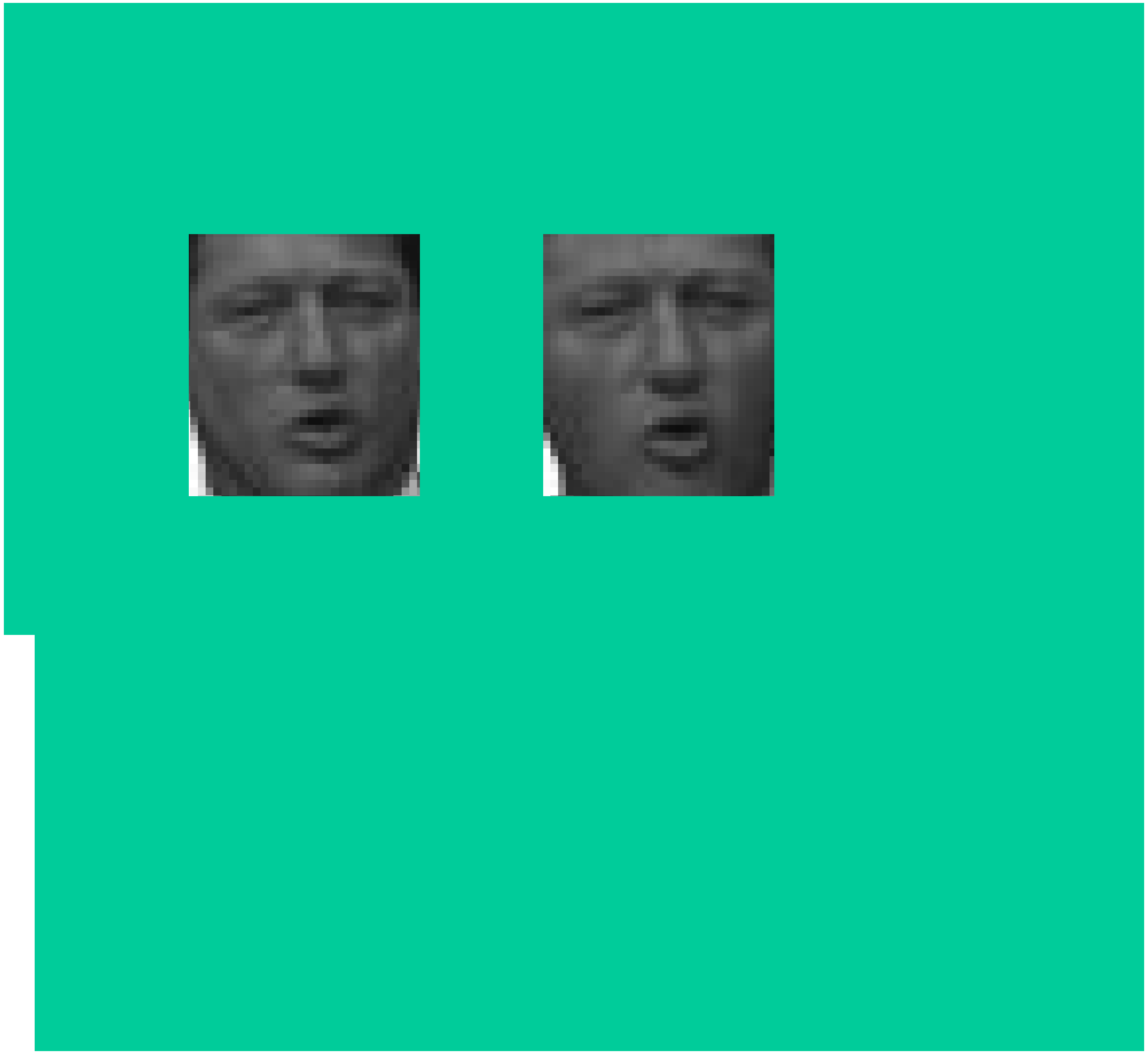


Illusions: What Do They Tell Us?



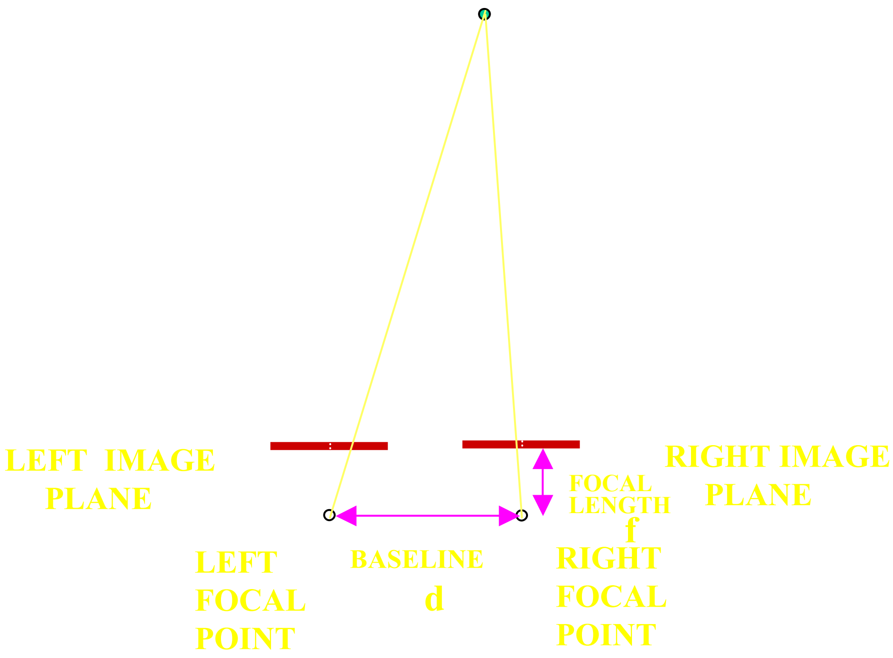
Illusions: What Do They Tell Us?

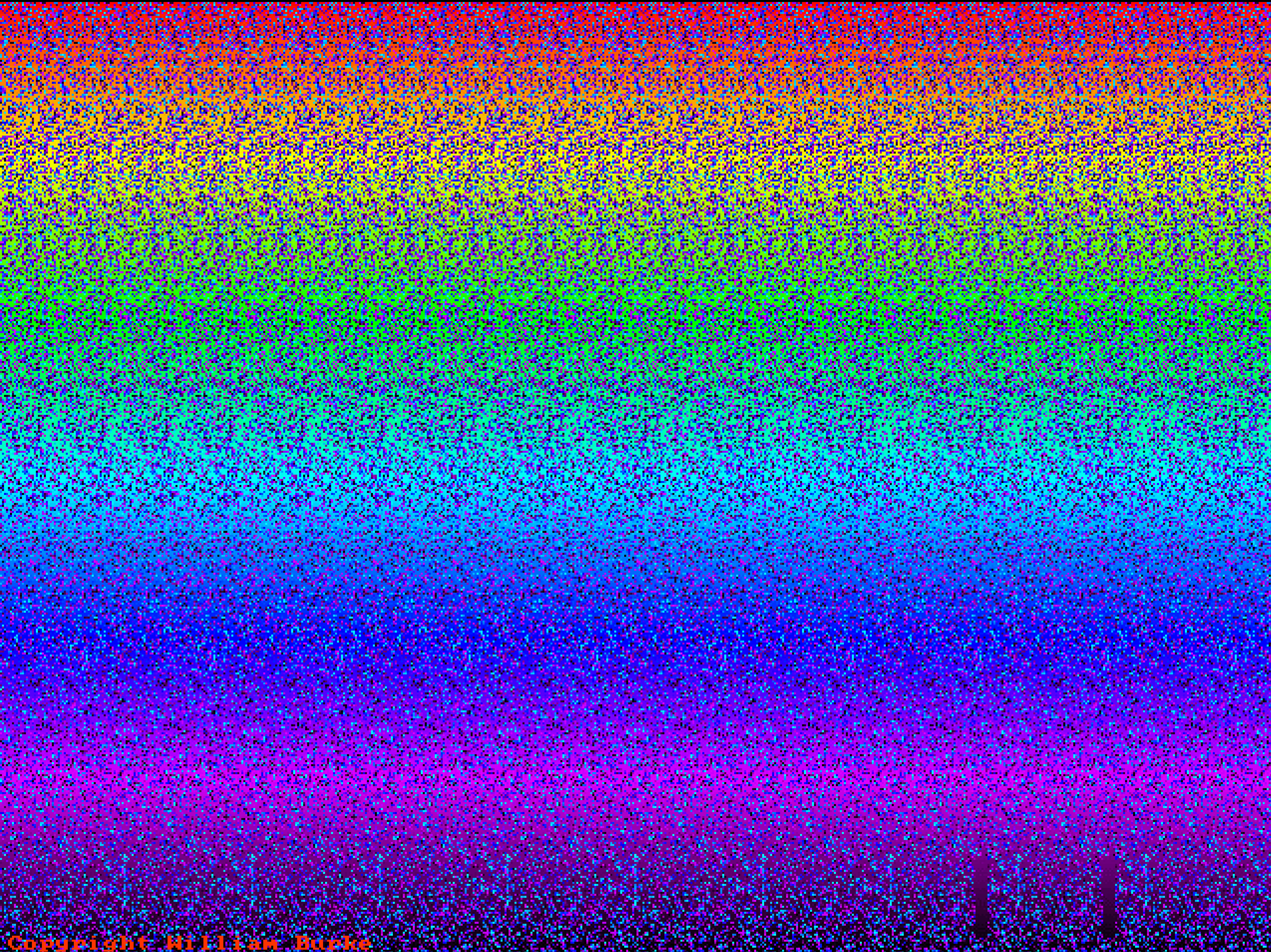
Illusions: What Do They Tell Us?



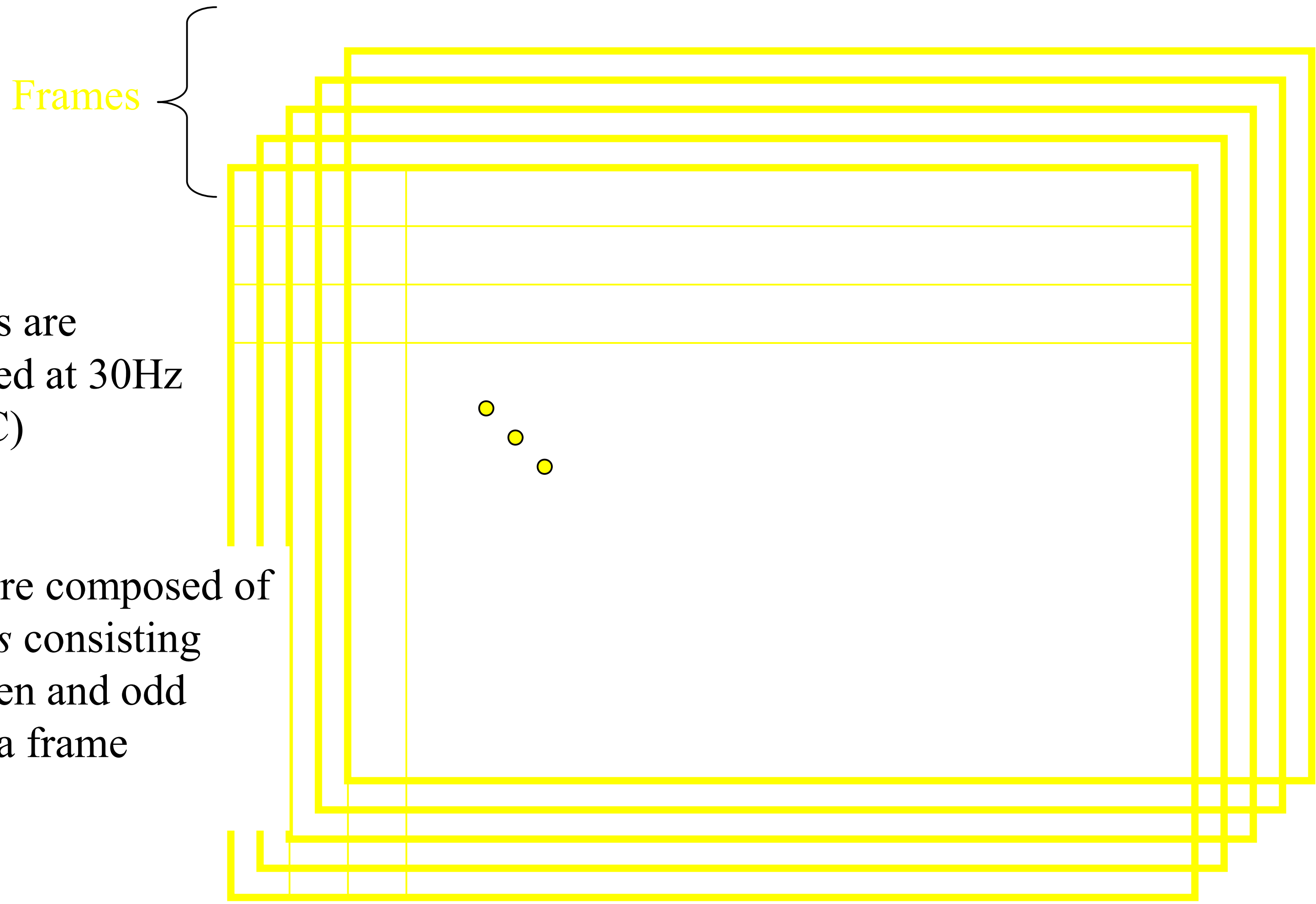
Problems of Computer Vision: Stereo Vision

From two (or more) images, determine the geometry of the scene by *matching* corresponding areas of the images




Copyright William Burke

THE ORGANIZATION OF AN IMAGE SEQUENCE

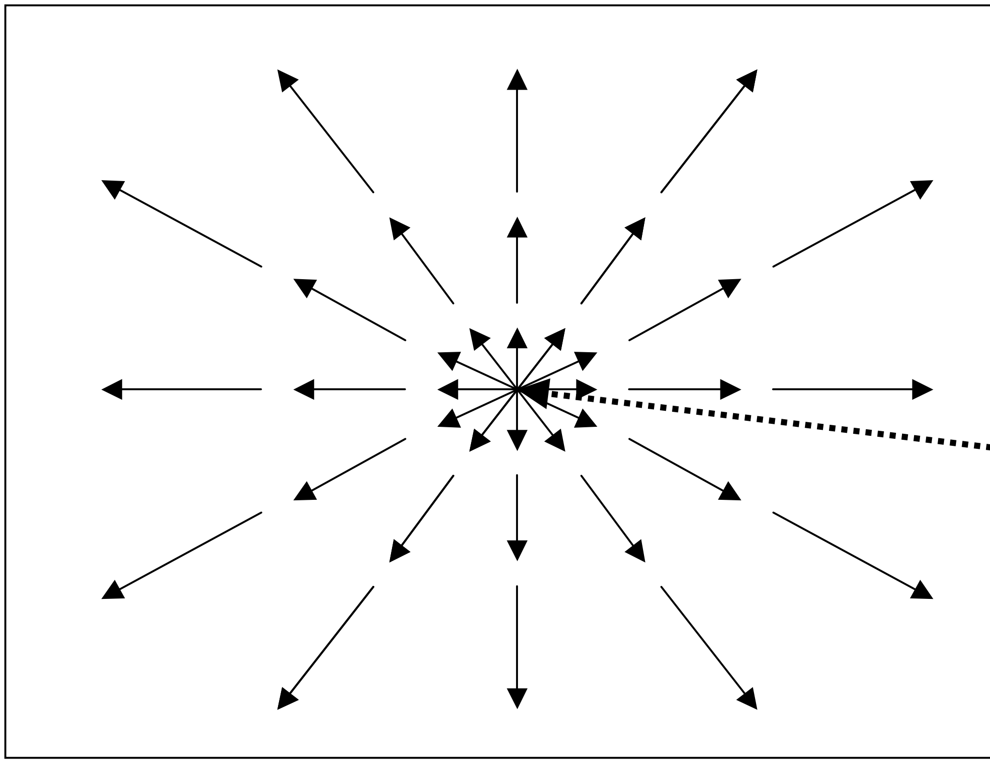


Frames are acquired at 30Hz (NTSC)

Frames are composed of two *fields* consisting of the even and odd rows of a frame

THE MOTION FIELD

The “instantaneous” velocity of points in an image



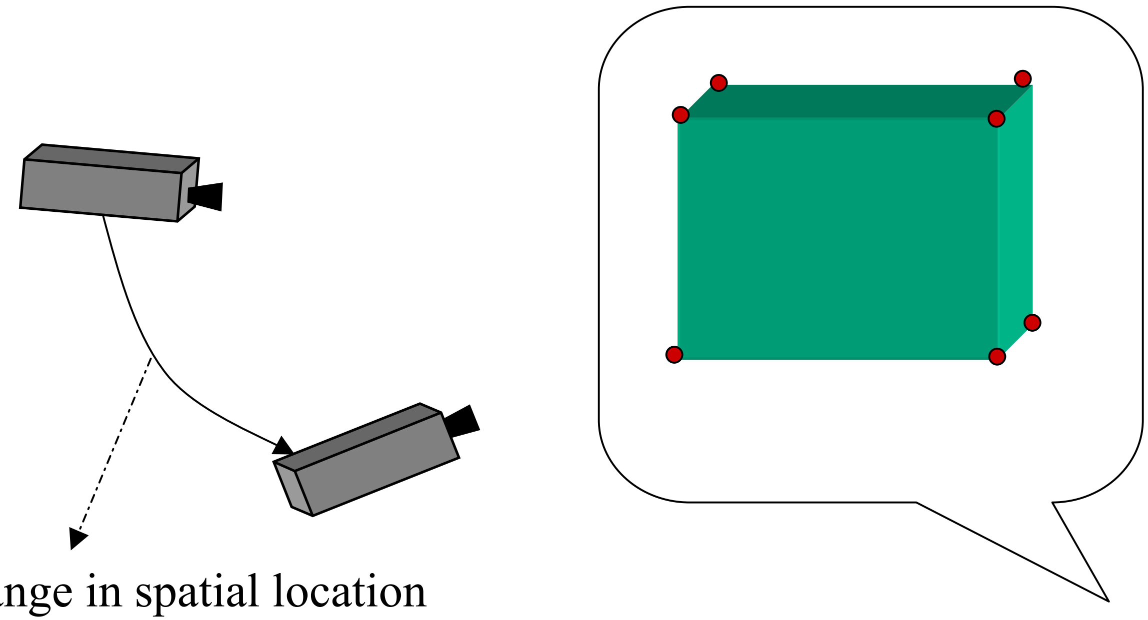
LOOMING

The *focus of expansion*

With just this information it is possible to calculate:

1. Direction of motion
2. Time to collision

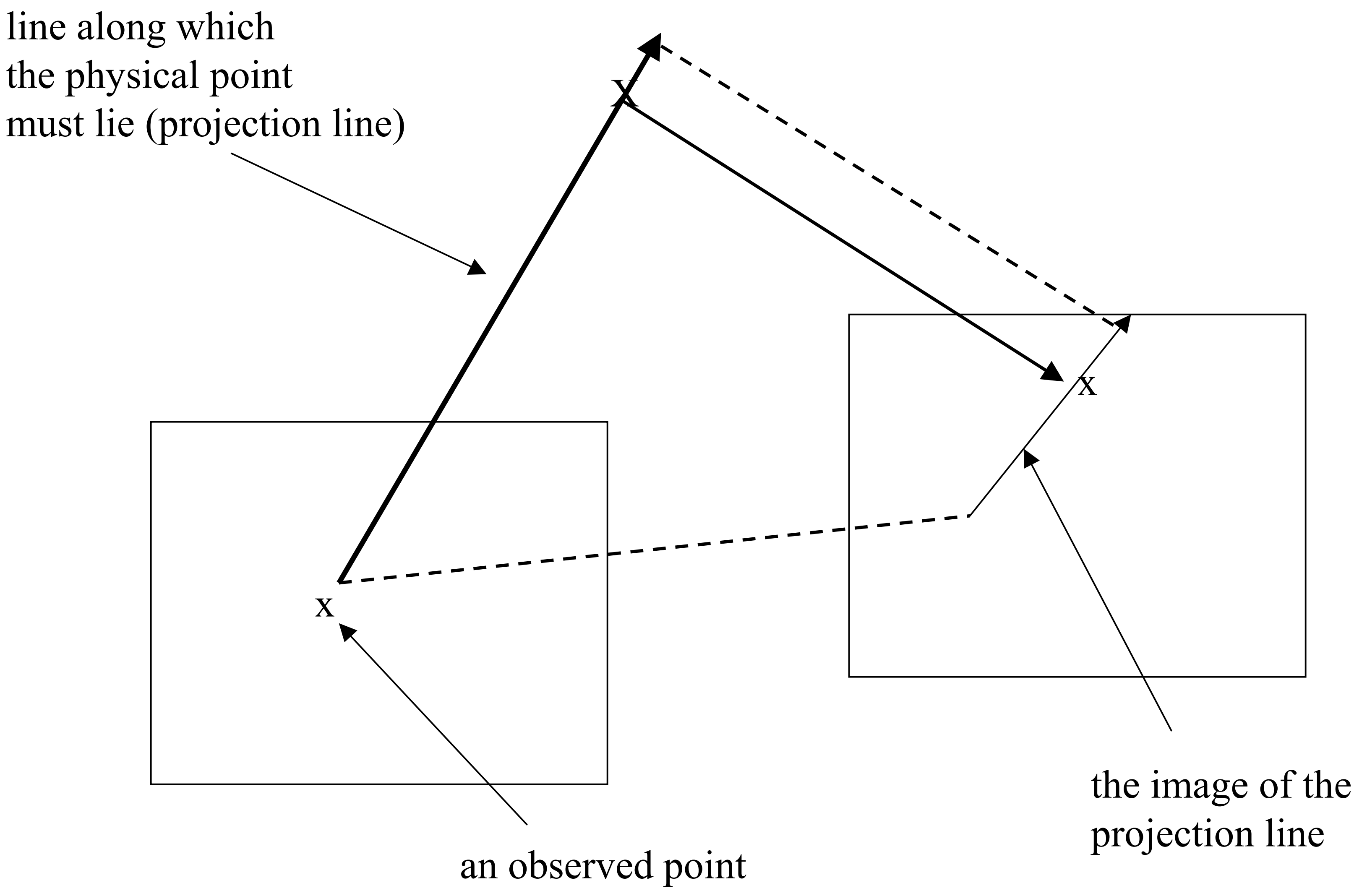
MOVING CAMERAS ARE LIKE STEREO



The change in spatial location between the two cameras (the “motion”)

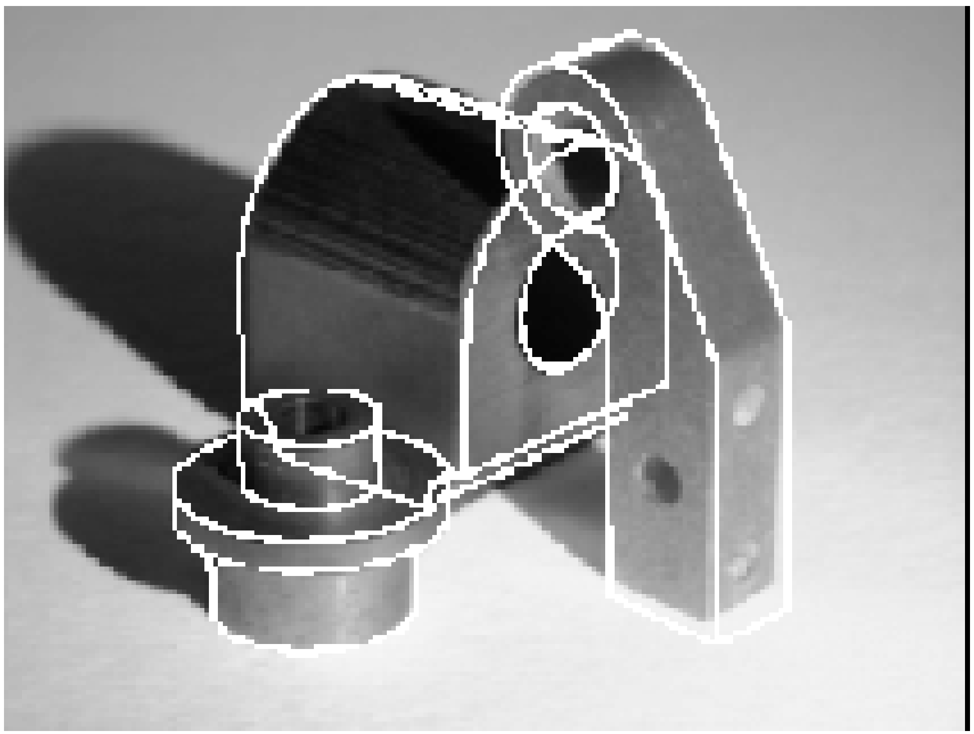
Locations of points on the object (the “structure”)

THE EPIPOLAR CONSTRAINT



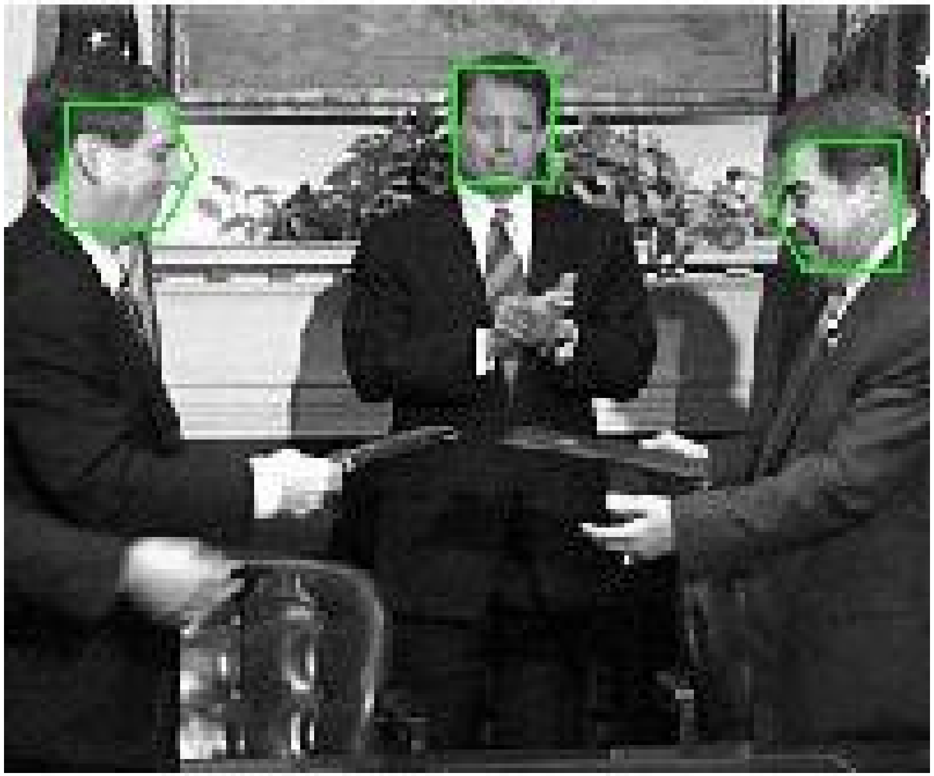
An Example
(Courtesy Carlo Tomasi)

Problems of Computer Vision: Recognition



Given a database of objects and an image determine what, if any of the objects are present in the image.

Problems of Computer Vision: Recognition

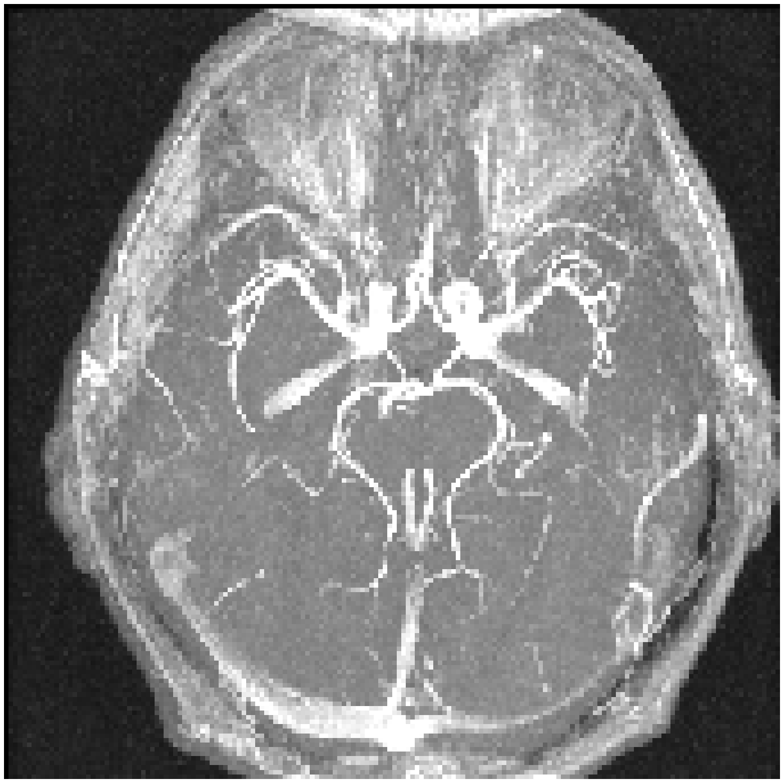


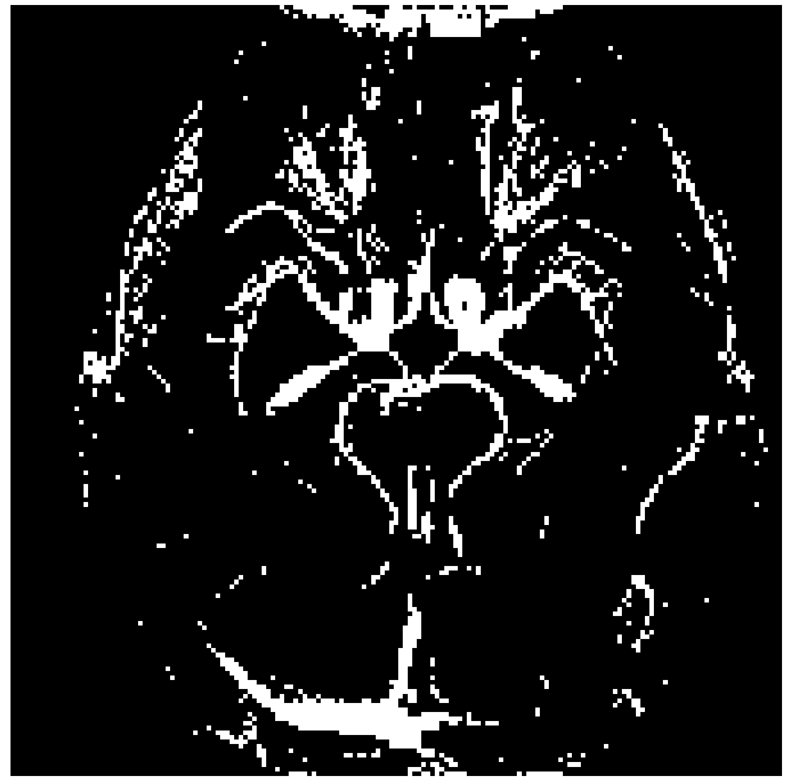
Given a database of objects and an image determine what, if any of the objects are present in the image.

Applications of Computer Vision: Biometrics

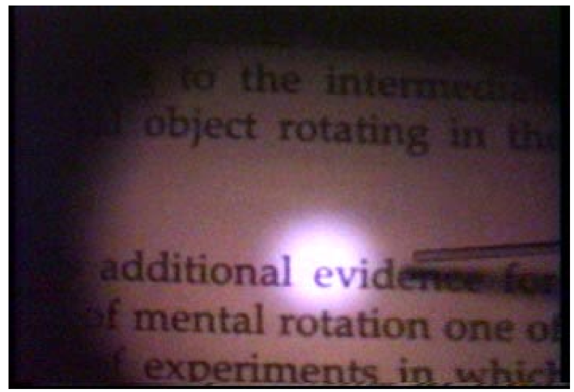
- Face recognition
- Iris scanning
- Fingerprint recognition
- Activity recognition

Applications of Computer Vision: Medical Imaging





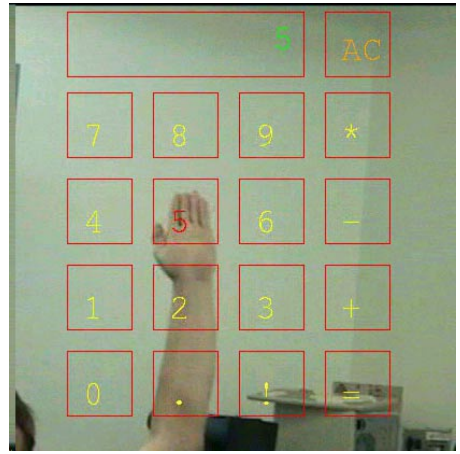
Applications of Computer Vision: Medical Imaging



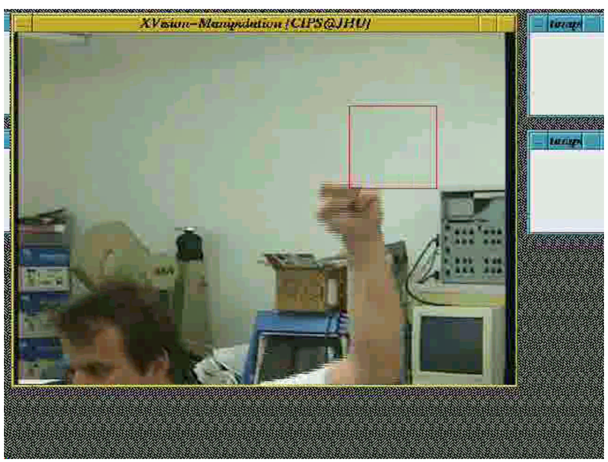
Note that these frames are rich in content. That is not true for vitreoretinal imagery.

Endoscopic Mosaic

Applications of Computer Vision: HCI

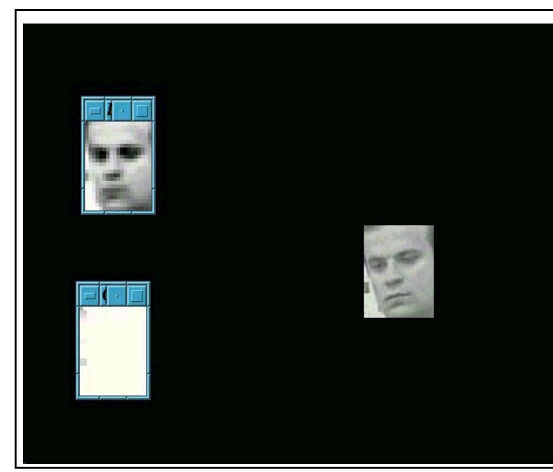


calculator



Gesture recognition

GUI



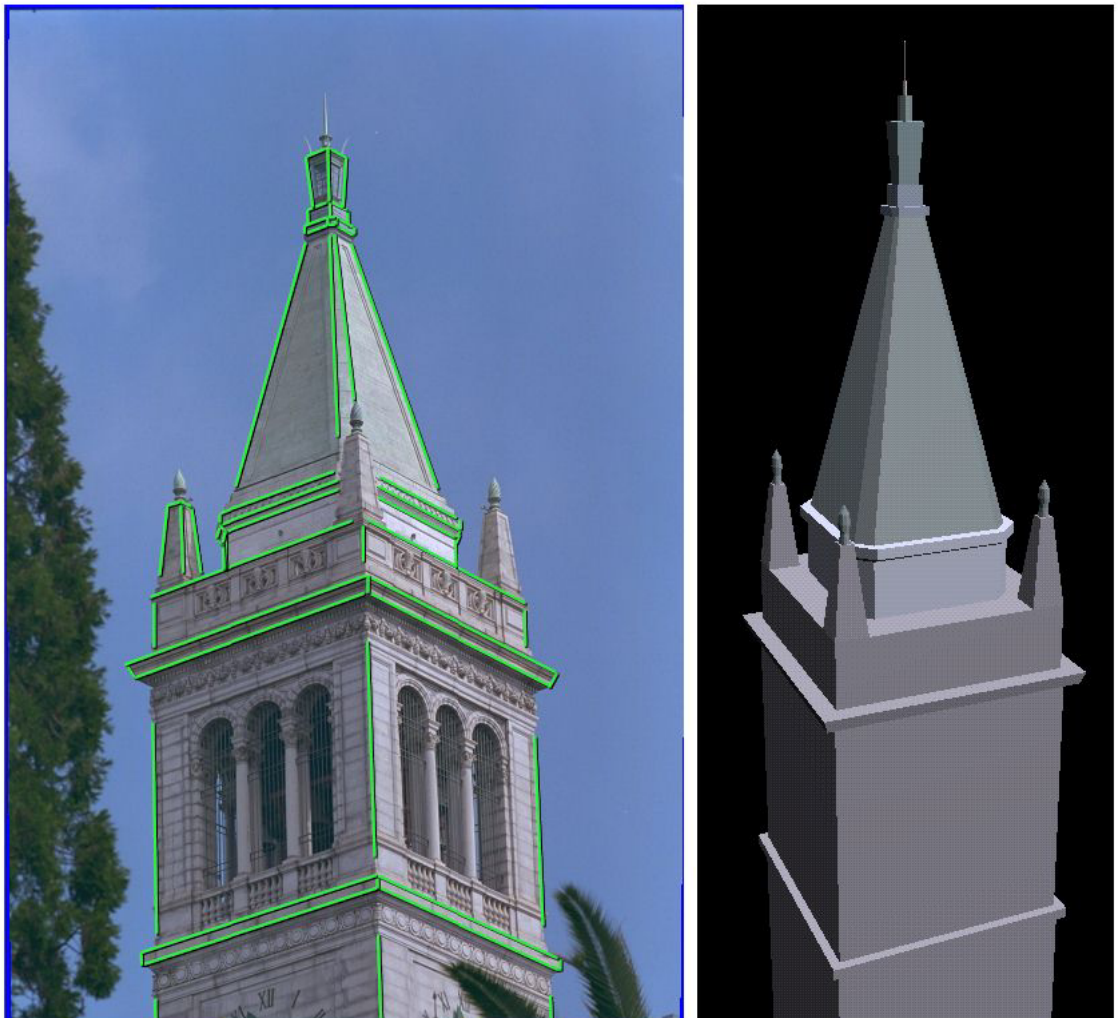
Face tracker

Applications of Computer Vision: Image Databases

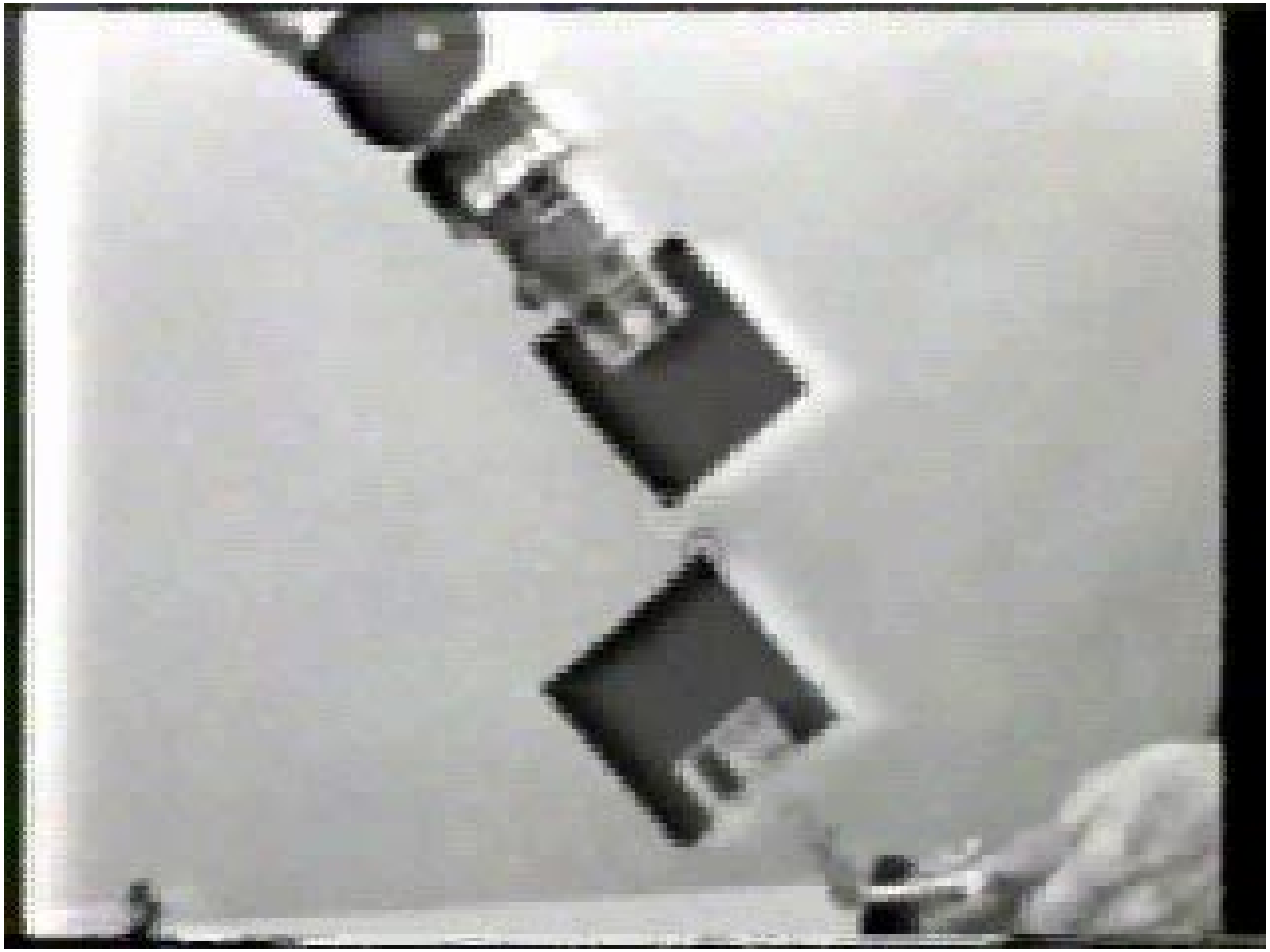
(Courtesy D. Forsyth & J. Ponce)

From a search for horse pix in 100 horse images and 1086 non-horse images

Applications of Computer Vision: Data Acquisition
(Jitendra Malik, Berkeley)



Applications of Computer Vision: Motion Control



Applications of Computer Vision: Motion Control
(CMU)

Applications of Computer Vision: Motion Control

Course Information

- Use the course WEB site

  - http://www.ugrad.cs.jhu.edu/~cs461

- For information on this course

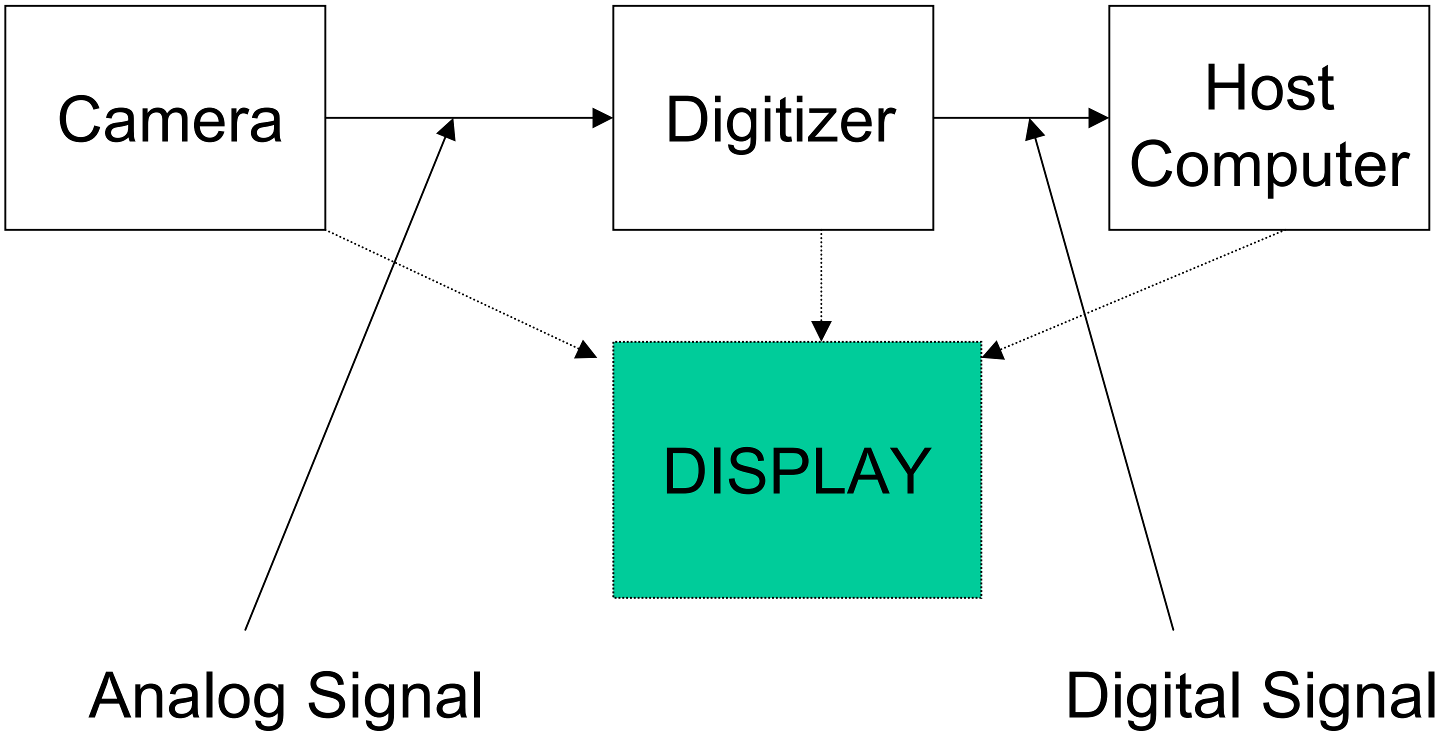
Computer Vision, Lecture 2
http://www.ugrad.cs.jhu.edu/~cs461

Professor Hager
http://www.cs.jhu.edu/~hager

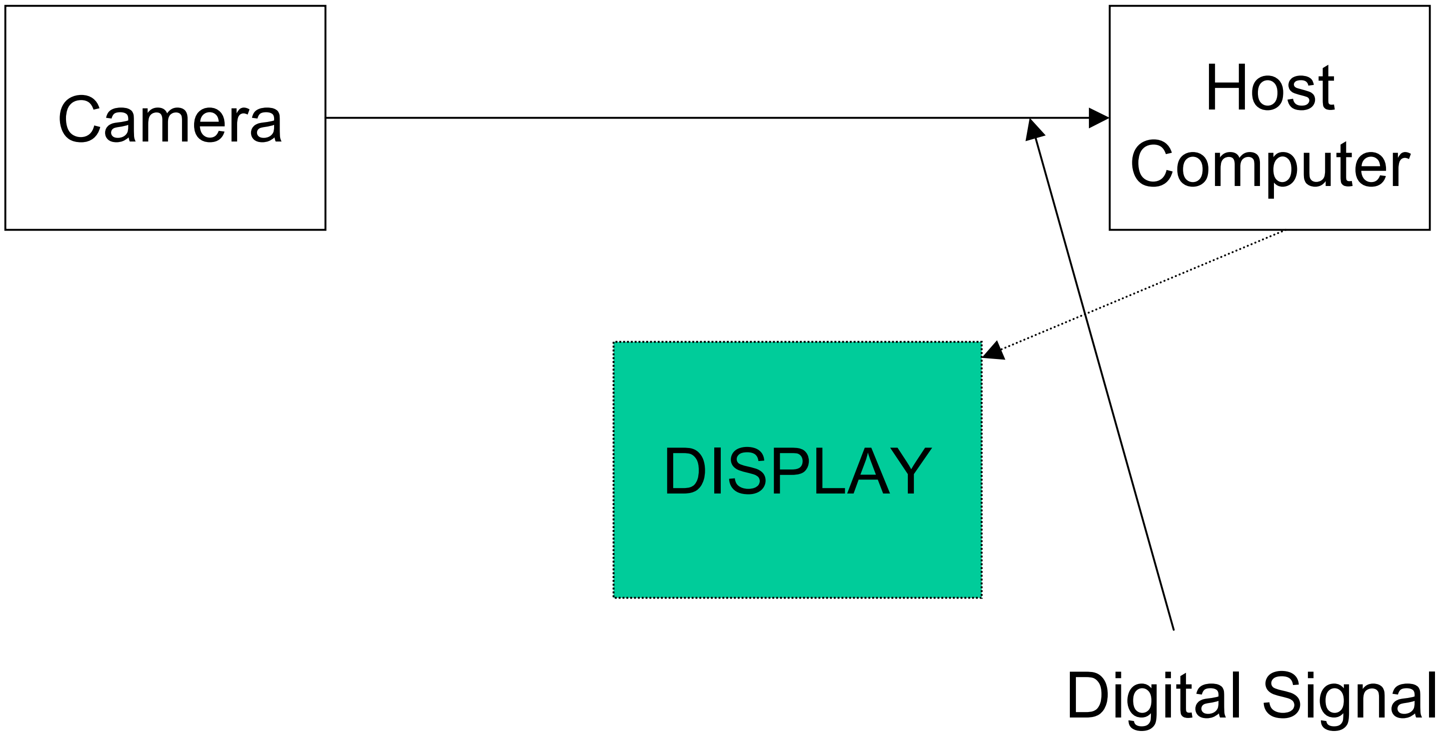
Today's Outline

- Camera imaging sensor basics
- Receptor response
- Color systems
- File formats
- Matlab

A “Traditional” Camera

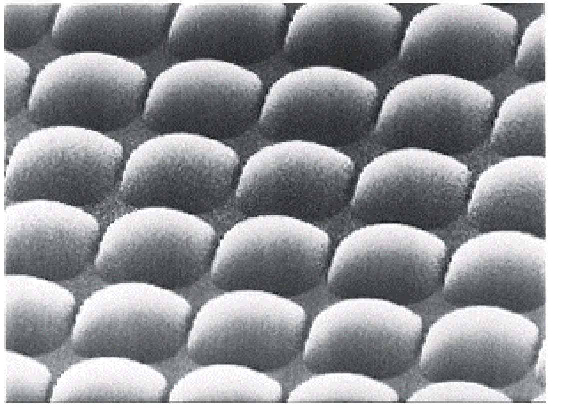


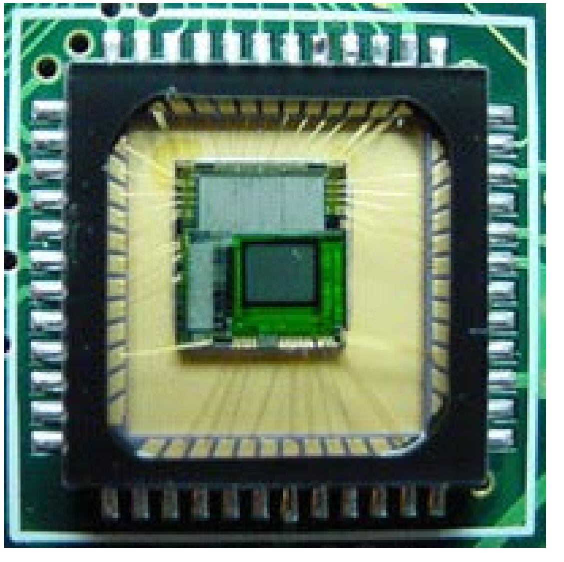
A Modern Digital Camera (Firewire)



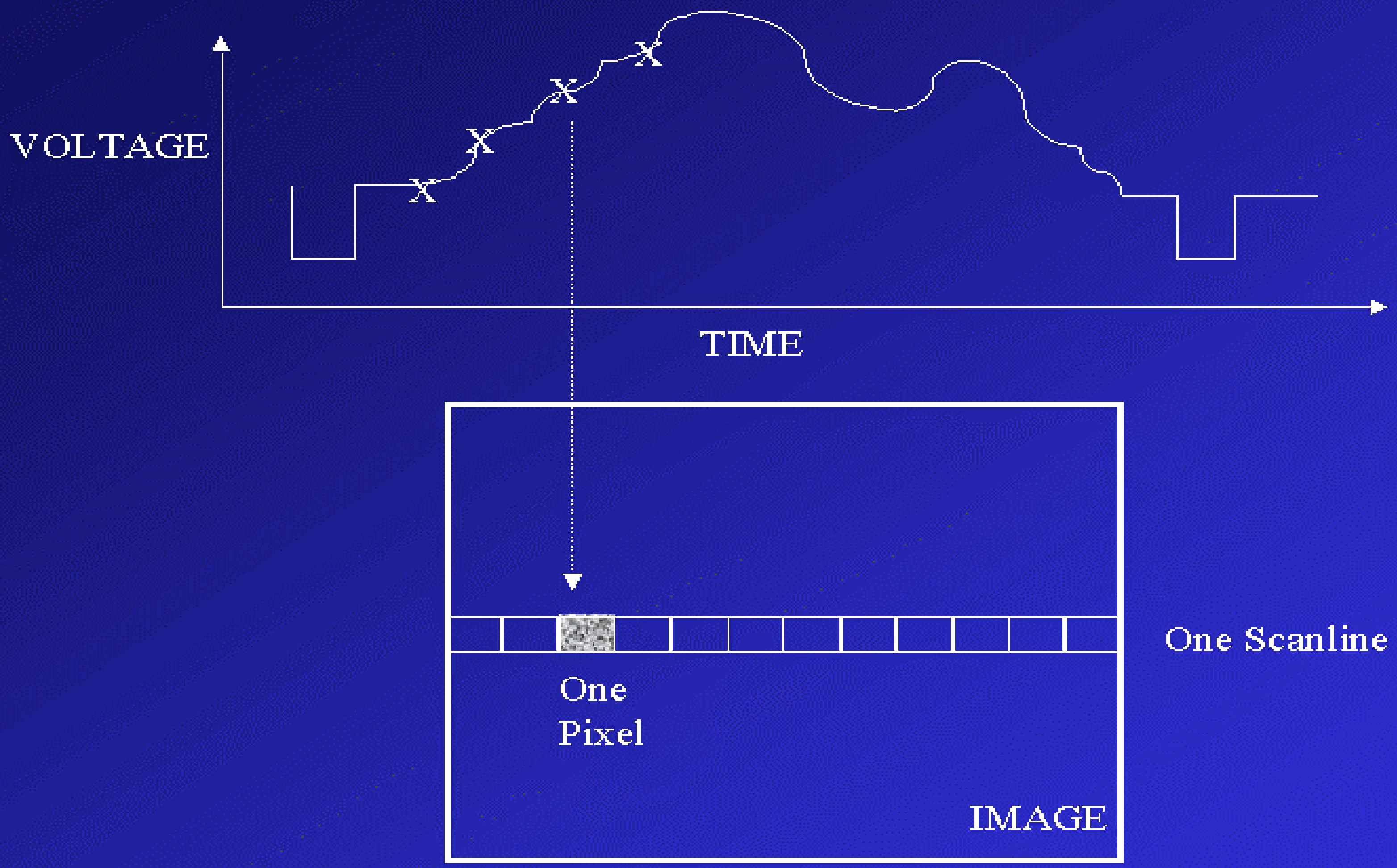
How Cameras Produce Images

- Basic process:
  - photons hit a detector
  - the detector becomes charged
  - the charge is read out as brightness
- Sensor types:
  - CCD (charge-coupled device)
    - most common
    - high sensitivity
    - high power
    - cannot be individually addressed
    - blooming
  - CMOS
    - simple to fabricate (cheap)
    - lower sensitivity, lower power
    - can be individually addressed





What's under the Hood



Digitization Effects

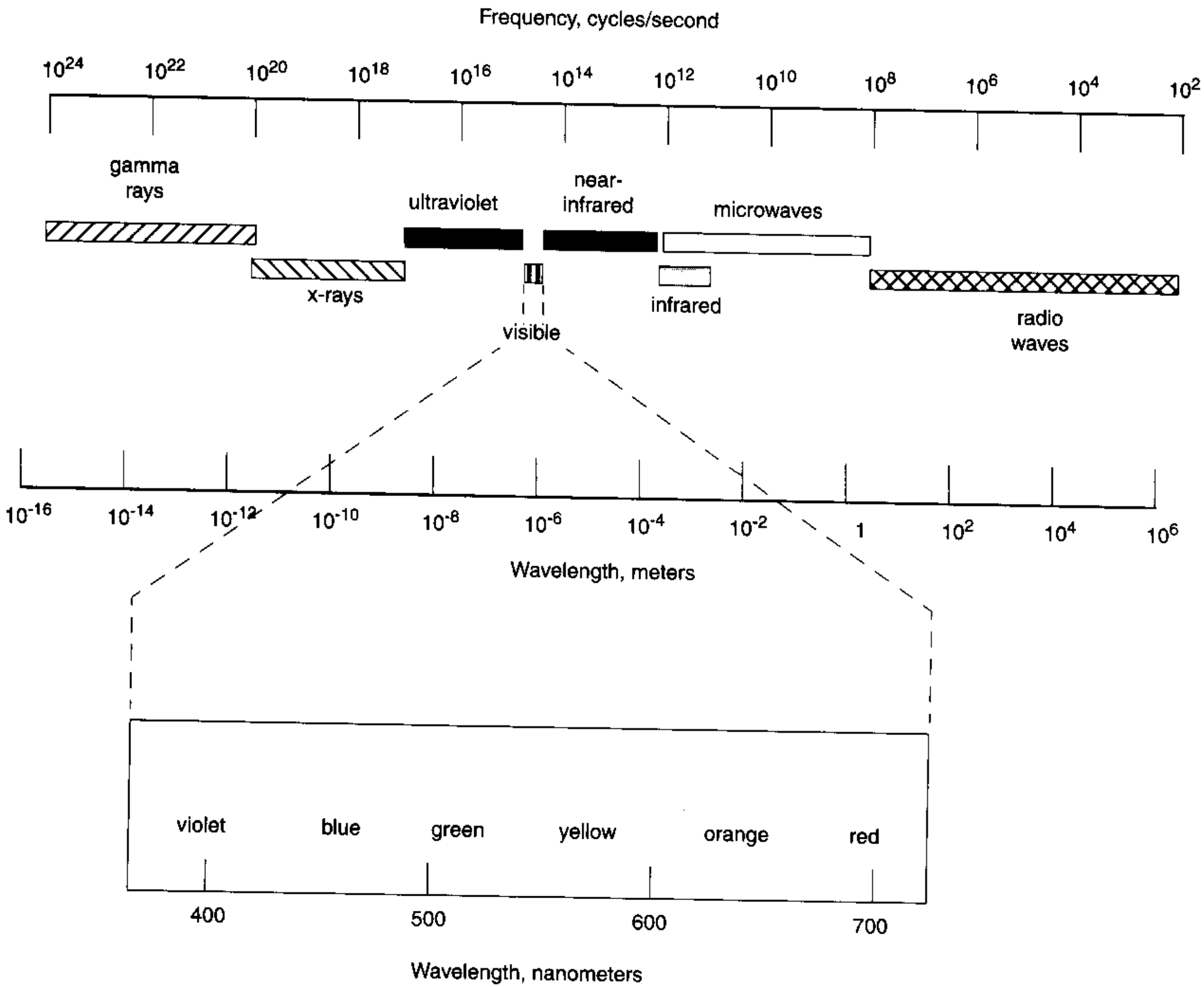
- The "diameter" d of a pixel determines the highest frequency representable in an image

$$l = 1/2d$$

- Real scenes may contain higher frequencies resulting in aliasing of the signal.
- In practice, this effect is often dominated by other digitization artifacts.
- One problem in particular is differing sampling rates between digitizer and camera readout of a row.

Color

What is Color?

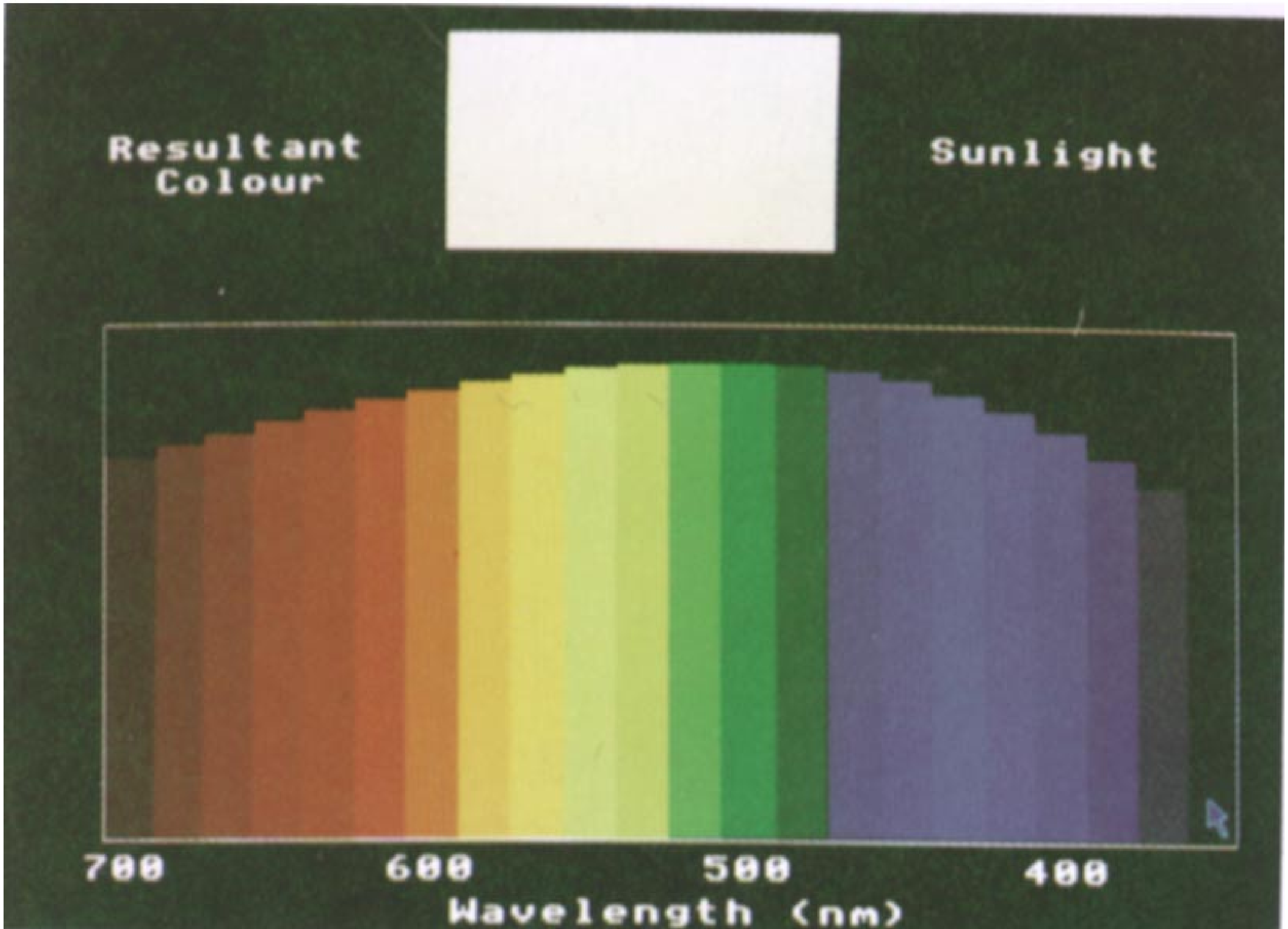


Region Tracking Video

What is Color?

- We almost never see a "pure" wavelength of light; rather a mixture of wavelengths, each with a different "power"
- Only some colors occur as pure wavelengths; many are mixtures of pure colors (e.g. white)

Sunlight

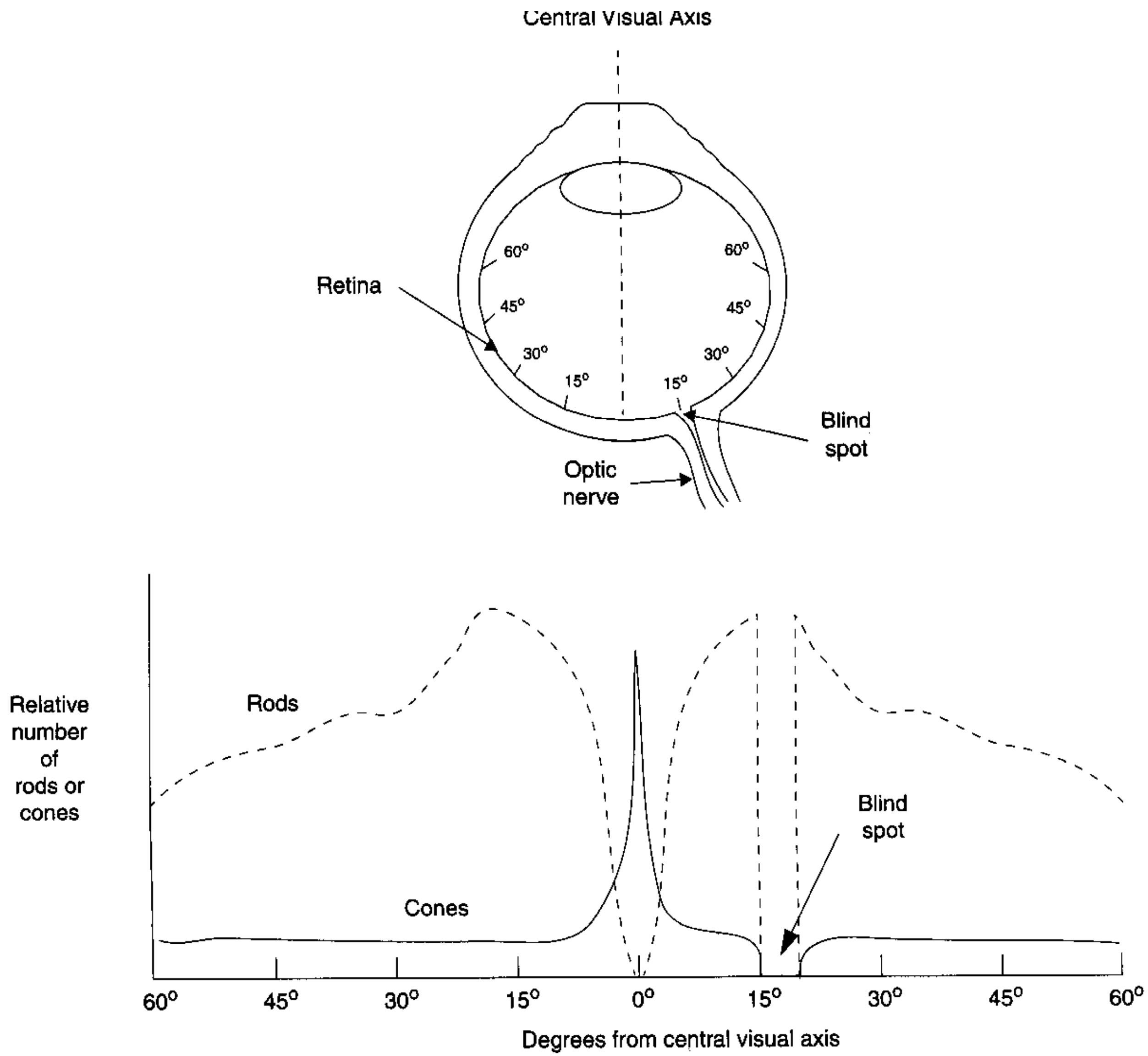


Color Measurement

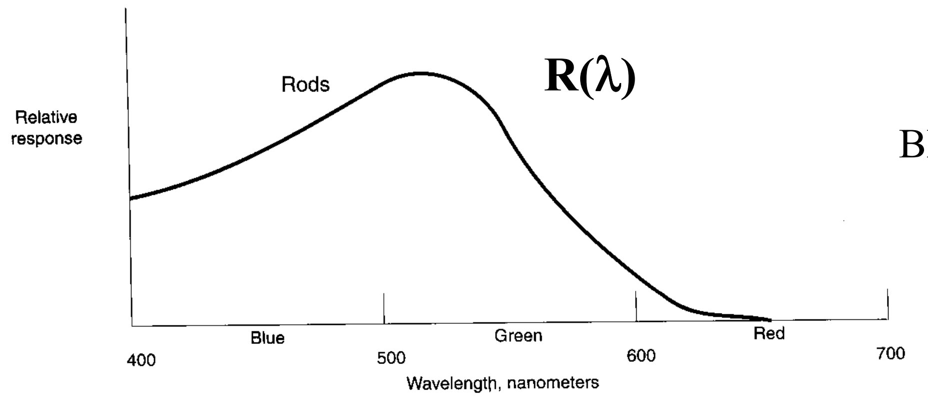
- Let λ denote wavelength
- Let $E(\lambda)$ denote the spectral power at a given wavelength (more about this later in the course)
- Let $\rho_k(\lambda)$ denote the responsiveness of a sensor k to a given wavelength of light
- Then we can compute the "response" r_k of k as

$$r_k = \int \rho(\lambda) E(\lambda) d\lambda$$

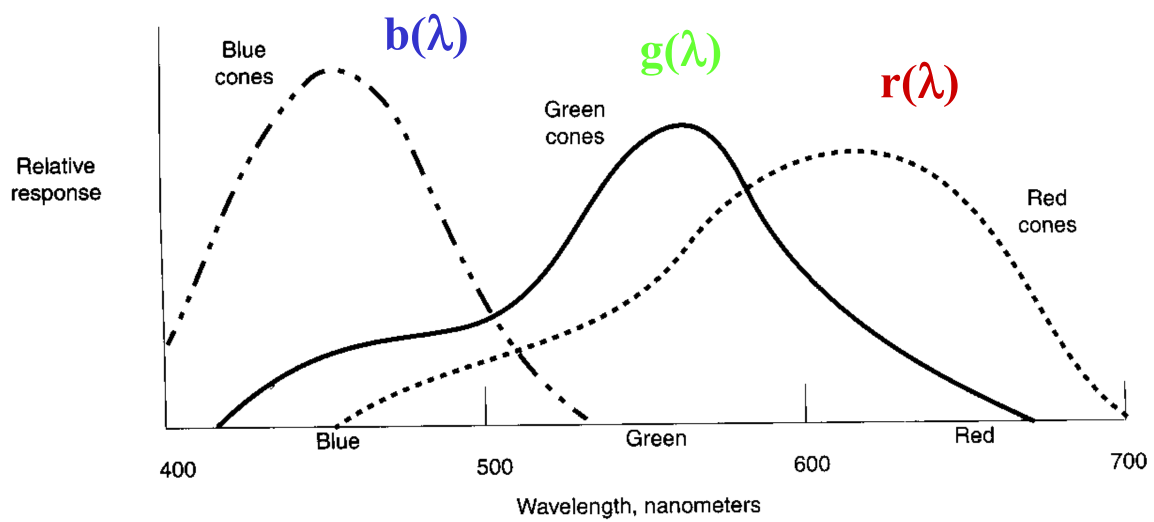
EXAMPLE: THE HUMAN EYE



THE HUMAN EYE: RESPONSE



$$\text{BRIGHTNESS} = \int_{\lambda=400nm}^{\lambda=700nm} R(\lambda) I(\lambda) d\lambda$$

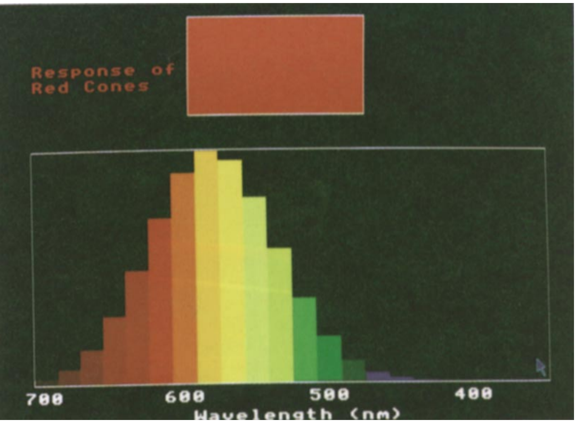


$$\text{RED} = \int_{\lambda=400nm}^{\lambda=700nm} r(\lambda) I(\lambda) d\lambda$$

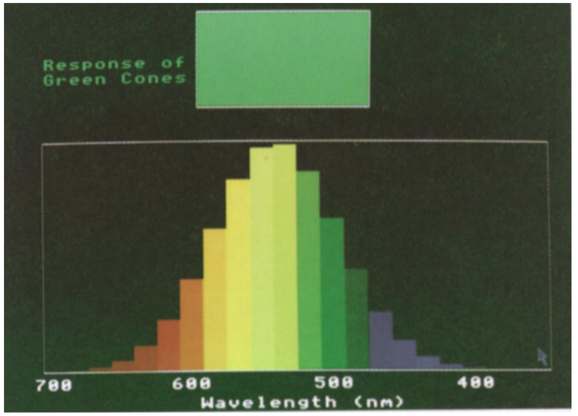
$$\text{GREEN} = \int_{\lambda=400nm}^{\lambda=700nm} g(\lambda) I(\lambda) d\lambda$$

$$\text{BLUE} = \int_{\lambda=400nm}^{\lambda=700nm} b(\lambda) I(\lambda) d\lambda$$

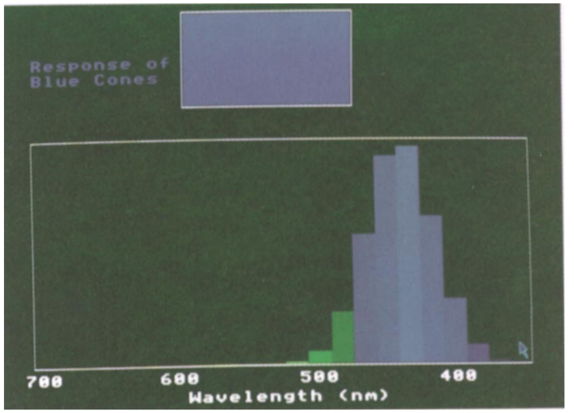
Color receptors



"Red" cone



"Green" cone

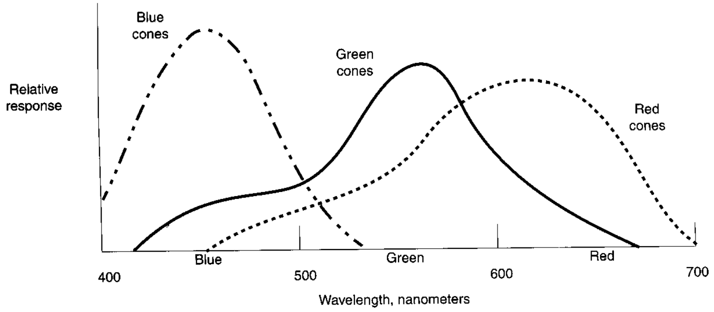


"Blue" cone

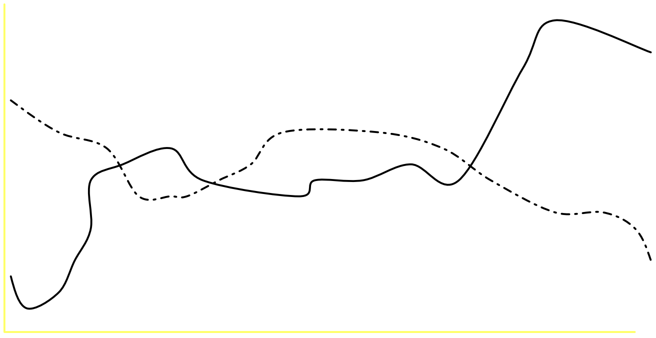
Principle of univariance: cones give the same amount of response, to different wavelengths. Output of cone is obtained by summing probability of absorption over wavelengths.

METAMERISM

- Two different Spectral Energy Distributions with the same RED, GREEN, BLUE response are termed ***metamers***.

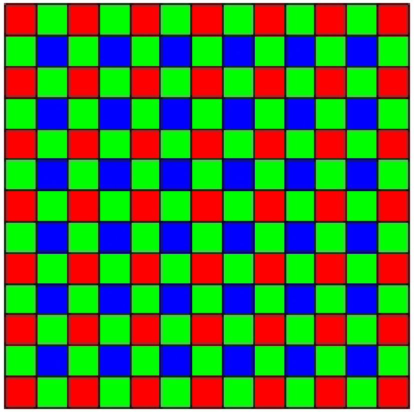


b(λ) g(λ) r(λ)



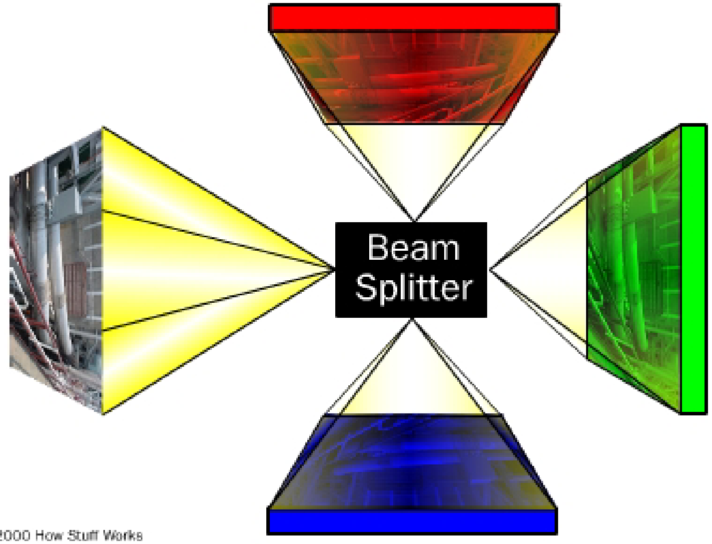
How Color Cameras Work

- 1 CCD cameras
  - A **Bayer** pattern is placed in front of the CCD
  - A **Demosaicing** process reads the pixels in a region and computes color and intensity

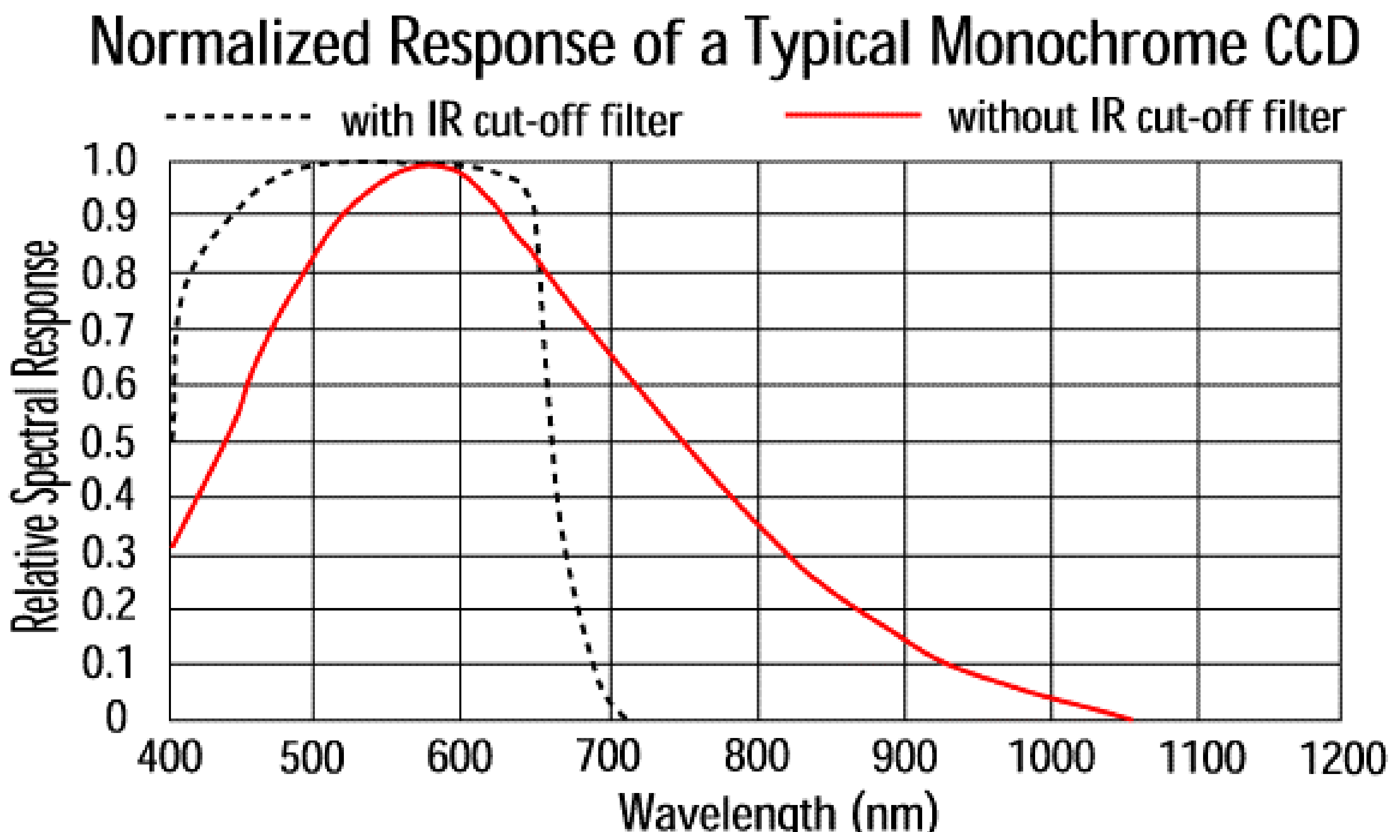


Bayer filter

- 3 CCD camera use a beam splitter and 3 separate CCDs
  - higher color fidelity
  - needs lots of light
  - requires careful alignment of ccds



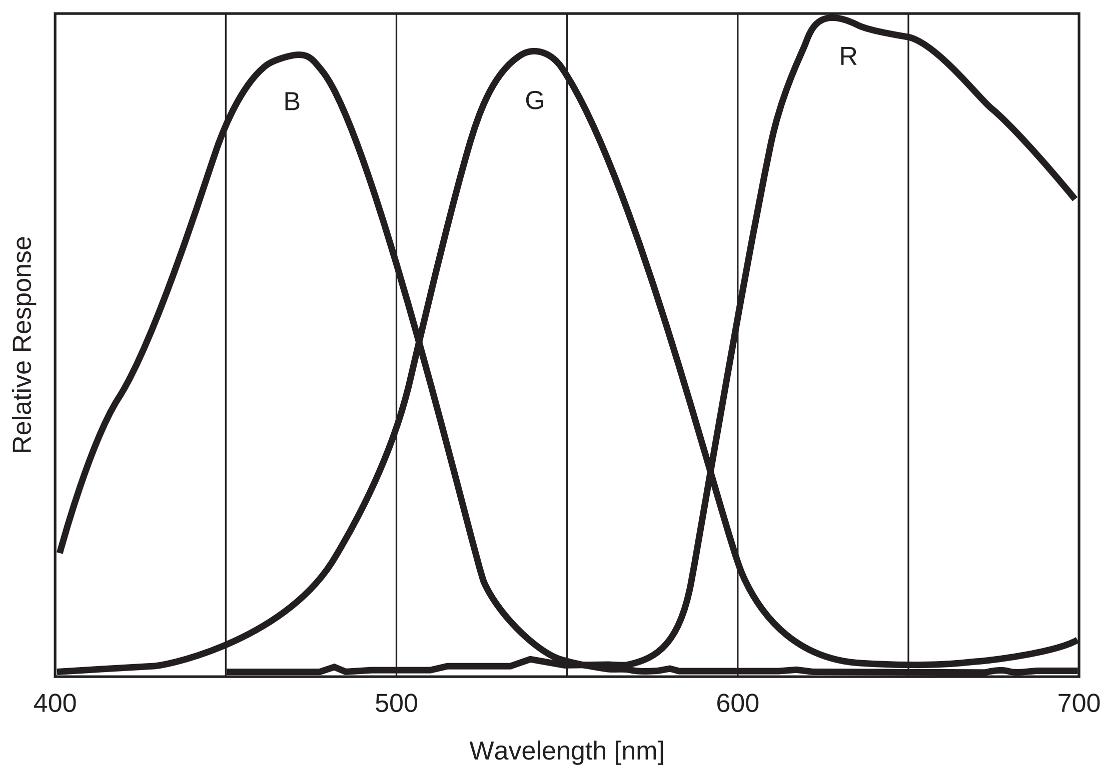
Unfiltered CCD Response



One Chip CCD Response (Sony DFW V500)

DFW-V500
DFW-VL500

SPECTRAL CHARACTERISTICS (TYPICAL VALUES)

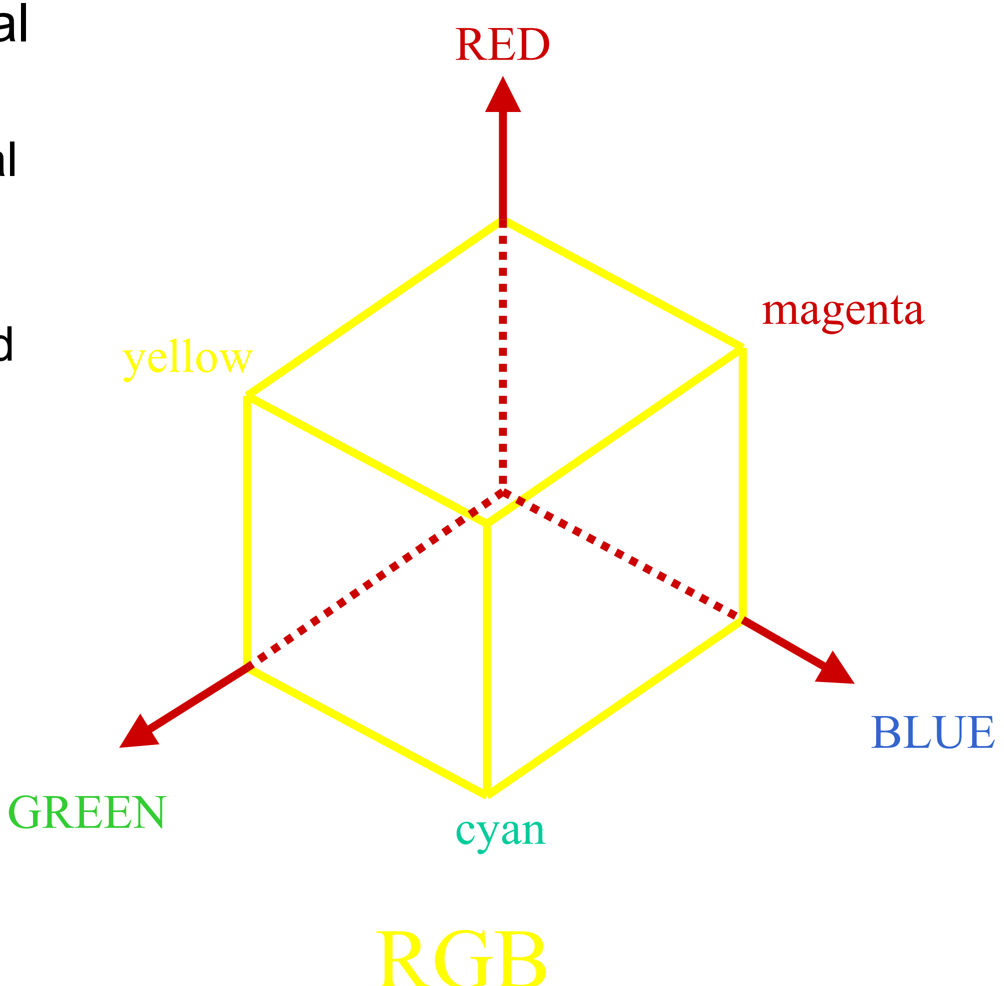


Filtering Colors

Green Red

Standard Linear Color Systems

- Several standards are used to define “color” based on spectral response functions
  - CIE (Commission International d’Eclairage) establishes standards
  - CIE XYZ is a popular standard with everywhere positive response
  - RGB requires a negative (subtractive) component in R response



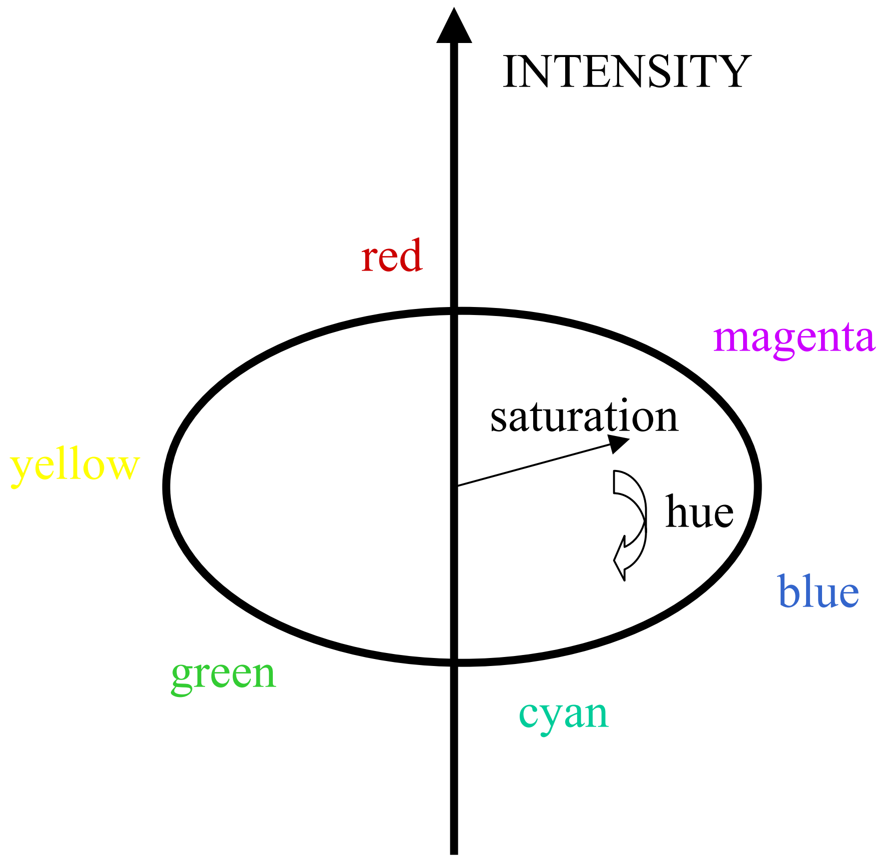
ANOTHER LINEAR SCHEME FOR REPRESENTING COLOR

- Invented for color television (NTSC)
- Backward compatible with B/W TV
- Y given higher bandwidth than I/Q

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} .3 & .59 & .11 \\ .6 & -.28 & -.32 \\ .21 & -.52 & .31 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

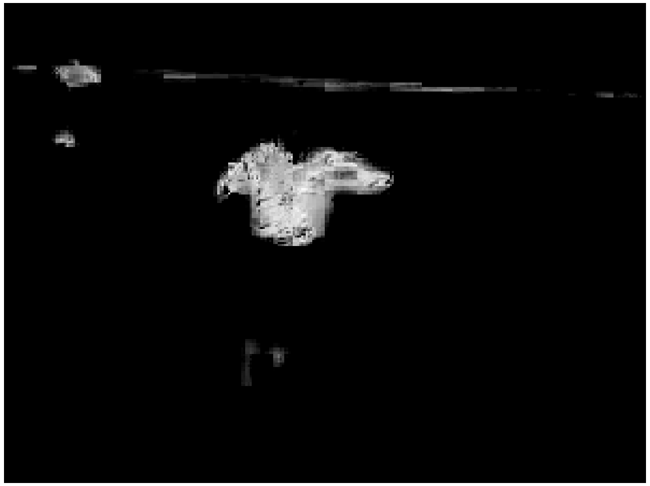
YIQ (also YUV)

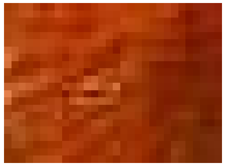
NONLINEAR SCHEMES FOR REPRESENTING COLOR



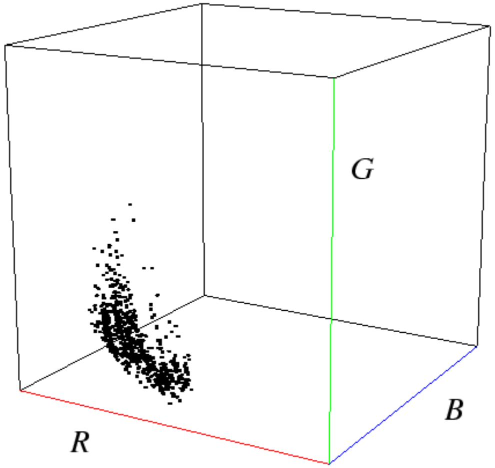
HSI

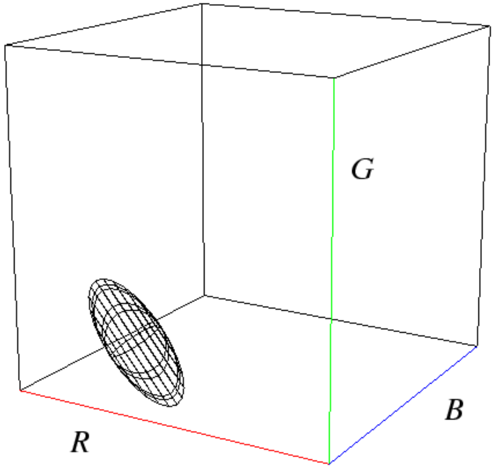
Homogeneous Region In RGB Space





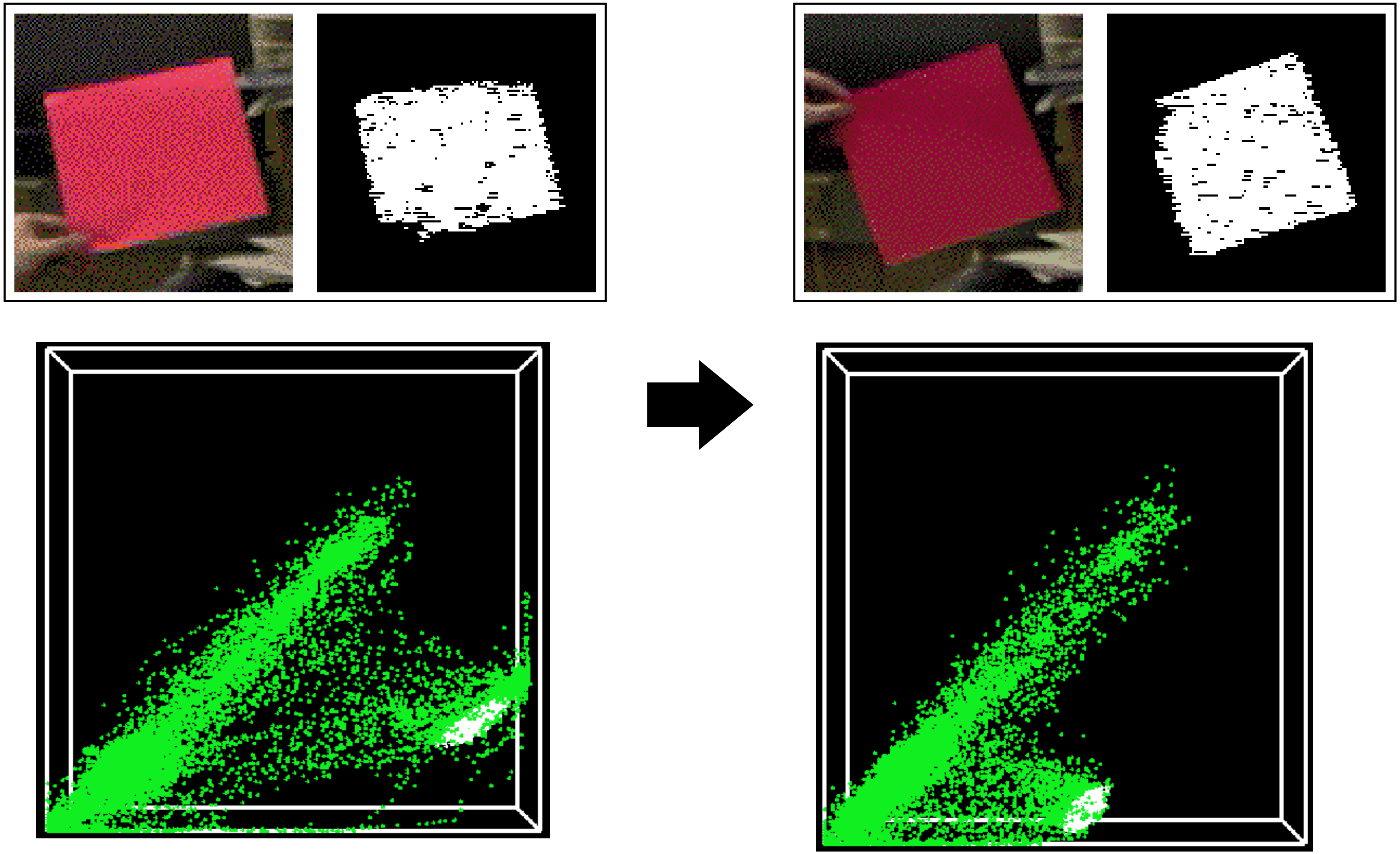
Sample



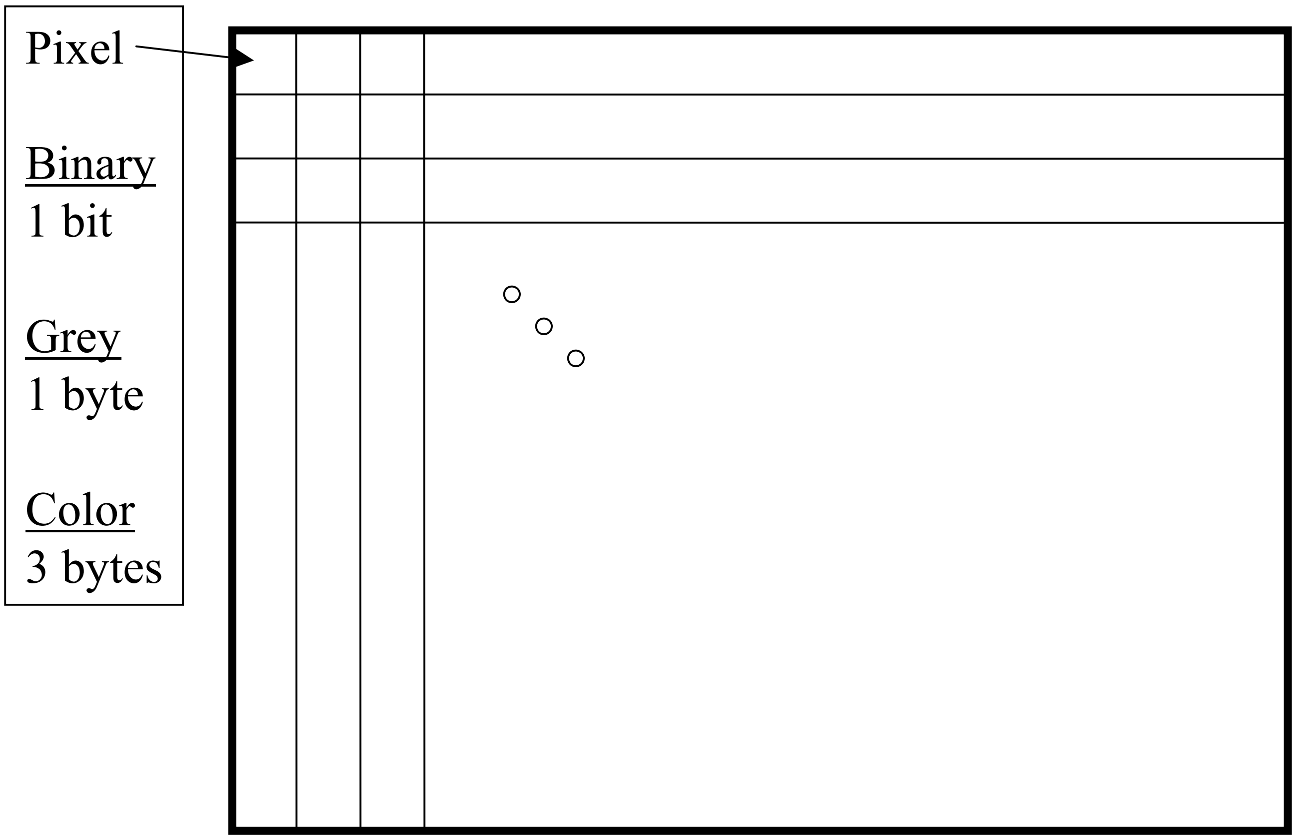


PCA-fitted ellipsoid

Homogeneous Region In RGB Space

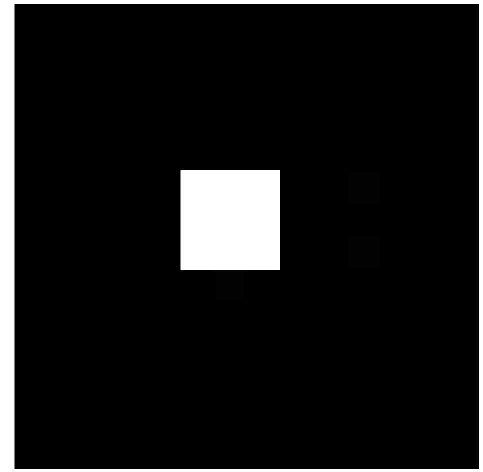


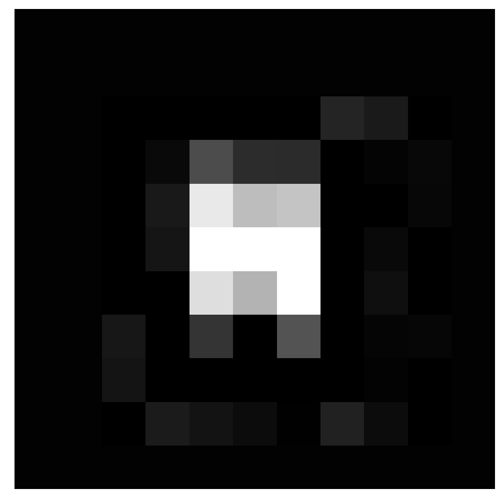
THE ORGANIZATION OF A 2D IMAGE



Storing Images

- Non-lossy schemes
  - pbm/pgm/ppm/pnm
    - code for file type, size, number of bands, and maximum brightness
  - tif (lossless and lossy versions)
  - bmp
  - gif (grayscale)





- Lossy schemes
  - jpg
    - uses Y Cb Cr color representation; subsamples the color
    - Uses DCT on result
    - Uses the fact the human system is less sensitive to color than spatial detail

GIF IMAGE FORMAT

- GIF (Graphics Interchange Format)
  - Limited to 8 bits/pixel for both color and gray-scale.

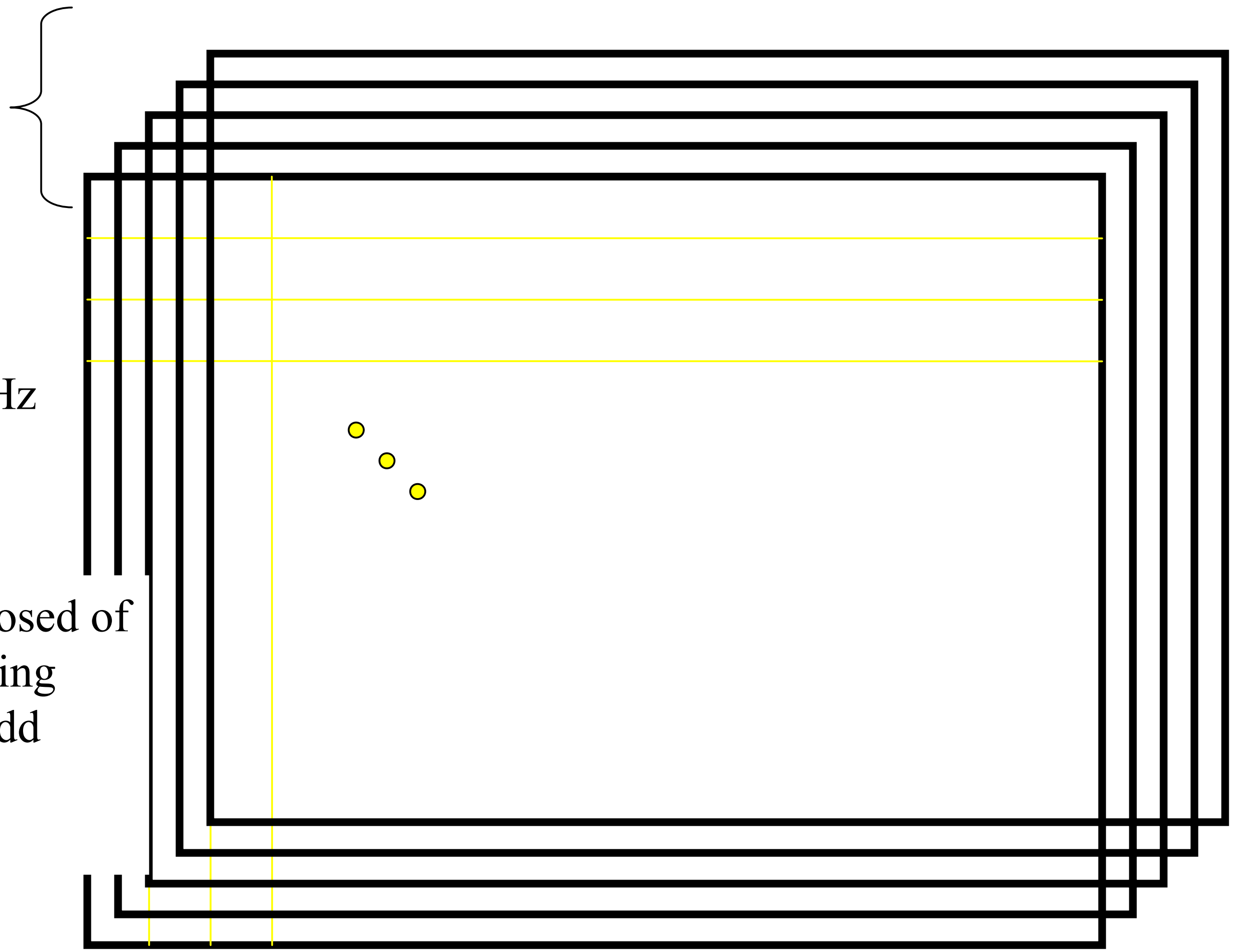
8-bit index	RED	GREEN	BLUE
0	R0	G0	B0
1	R1	G1	B1
2	R2	G2	B2
	⋮	⋮	⋮
254	R254	G254	B254
255	R255	G255	B255

TIFF IMAGE FORMAT

- TIFF (Tagged Image File Format)
  - More general than GIF
  - Allows 24 bits/pixel
  - Supports 5 types of image compression including:
    - RLE (Run length encoding)
    - LZW (Lempel-Ziv-Welch)
    - JPEG (Joint Photographic Experts Group)

THE ORGANIZATION OF AN IMAGE SEQUENCE

Frames



Frames are acquired at 30Hz (NTSC)

Frames are composed of two *fields* consisting of the even and odd rows of a frame

BANDWIDTH REQUIREMENTS

<u>Binary</u>
1 bit * 640x480 * 30 = 9.2 Mbits/second

<u>Grey</u>
1 byte * 640x480 * 30 = 9.2 Mbytes/second

<u>Color</u>
3 bytes * 640x480 * 30 = 27.6 Mbytes/second (actually about 37 mbytes/sec)

Typical operation: 3x3 convolution

9 multiplies + 9 adds ➔ 180 Mflops

Today's PC's are just getting to the point they can process images at frame rate

Matlab Basics

- Starting, stopping, help, demos, math, & variables
- Matrix definition and indexing

```
>> A = [ 1  2  3 ; 4  5  6 ; 7  8  9 ]
```

or

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

```
>> A(3,2)

>> A(3,:)

>> A(3,1:2) = [ 0  0 ]

>> A'
```

How would you set the middle row to be the first column?

```
>> A(:,:,2) = A

>> size(A)
```


See Assignment 1, part 1 for a more thorough introduction.

Matlab Built-Ins

- for, if, while, switch -- execution control
- who, whos, clear -- variable listing and removing
- save, load <file> -- saving or restoring a workspace
- diary <file> -- start recording to a file

  diary off ; diary on

- path, addpath -- display or add to search path
- close, close all, clc -- close windows, clear console
- double vs. uint8 -- data casting functions
- zeros(x,y,…) -- creates an all-zero x by y … matrix used for basic memory allocation

See the CS153 reference page for matlab links.

Images in Matlab (& Functions)

Images in Matlab are really matrices. The image toolbox is a special set of functions for representing matrices.

Images are either stored as uint (typically 0 to 255) or as doubles (typically ranging from 0 to 1). Take care to normalize images to this range (some functions do this automatically).

See Assignment 1, part 2 for a more thorough introduction.

Images in Matlab (& Functions)

Built-in functions:

A =imread(<filename>, <type>) -- pull from file

imwrite(A, <filename>, <type>) -- write to file

imagesc(A) -- display image

imshow(A) --- better way to display

Types

'tif'
'jpg'
'bmp'
'png'
'hdf'
'pcx'
'xwd'

single-quoted strings

See Assignment 1, part 2 for a more thorough introduction.

Image ⟷ Matrix

Matlab matrix A

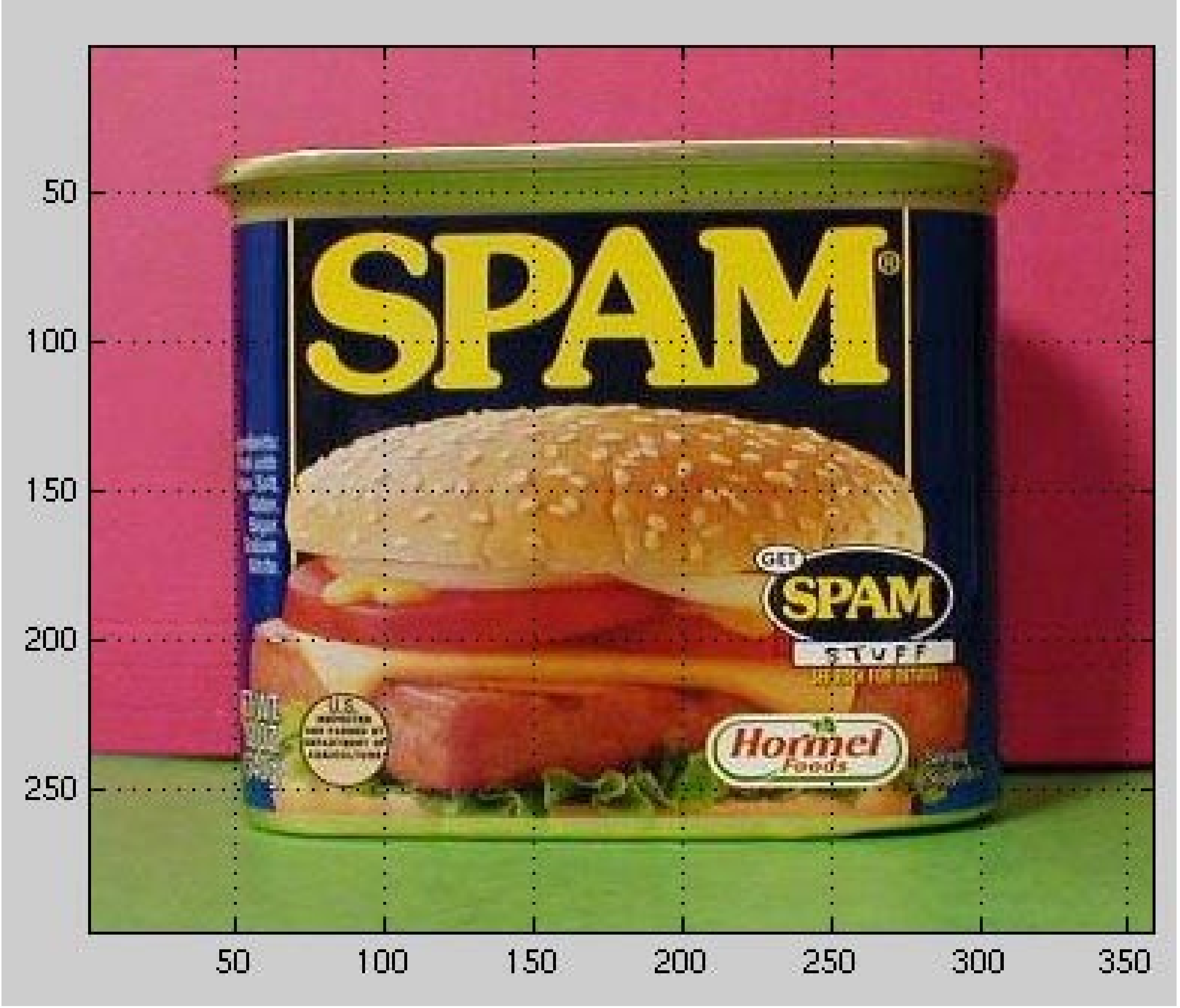
size(A)

A(1:10,1:10,:)

A(200, 50:300, 3)

The large "M"

The spam's location



Matlab Built-Ins

- F = fft2(A) -- Fourier transform: F is A's frequencies
- A = ifft2(F) -- inverse Fourier: A is F's image
- F2 = fftshift(F) -- places the coordinate system at center

- diary <file> -- start recording to a file diary off ; diary on
- keyboard -- give control back to user until return
- double vs. uint8 -- data casting functions
- zeros(x,y,…) -- creates an all-zero x by y … matrix used for basic memory allocation

See the CS153 reference page for more matlab links.