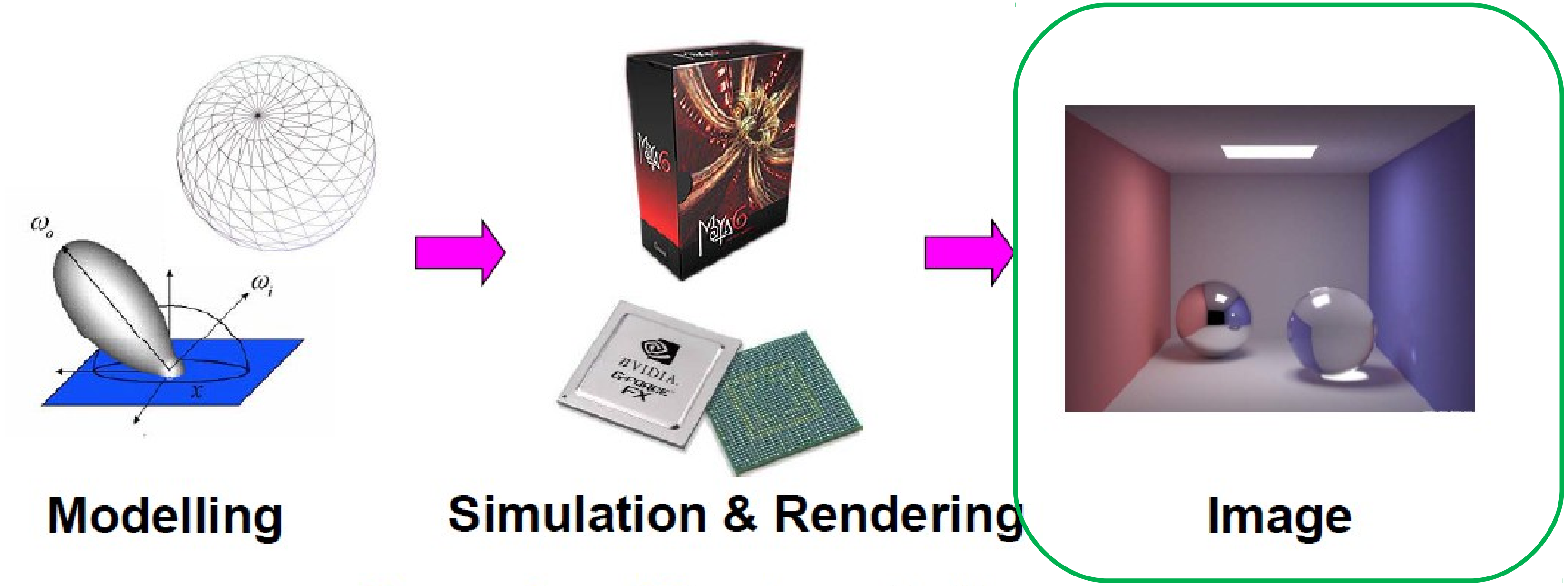
Images and Displays

Lecture 2

Introduction

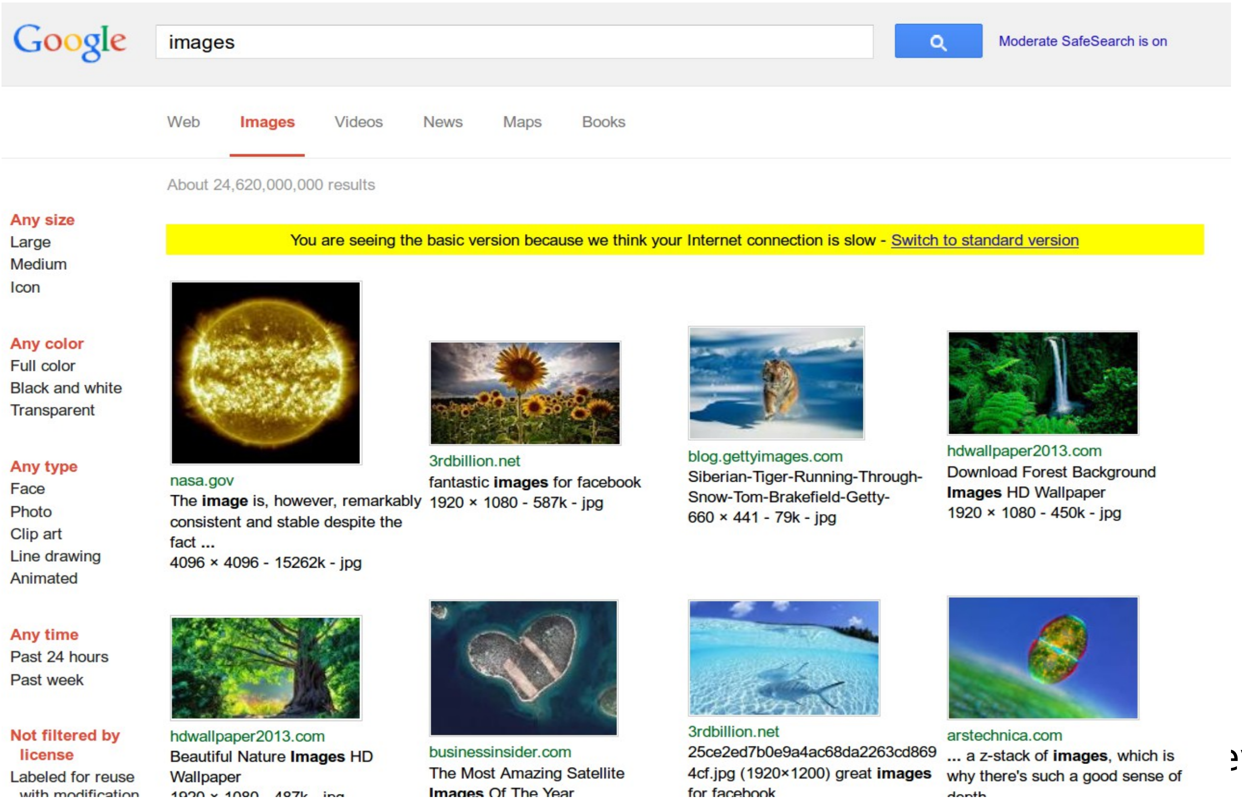
- **Computer graphics:** The study of creating, manipulating, and using visual images in the computer.



Computer vision inverts the process
Image processing deals with images

What is an image?

- A photographic print?
- Pixels?
- Some numbers in RAM?



An image is:

- A 2D distribution of intensity or color
- A function defined on a two-dimensional plane

$$I : \mathbb{R}^2 \rightarrow \ldots$$

- Note: no mention of pixels yet
- To do graphics, must:
  - represent images—encode them numerically
    - Image files such as jpg, png, svg files
  - display images—realize them as actual intensity distributions

Representative display technologies

Computer displays

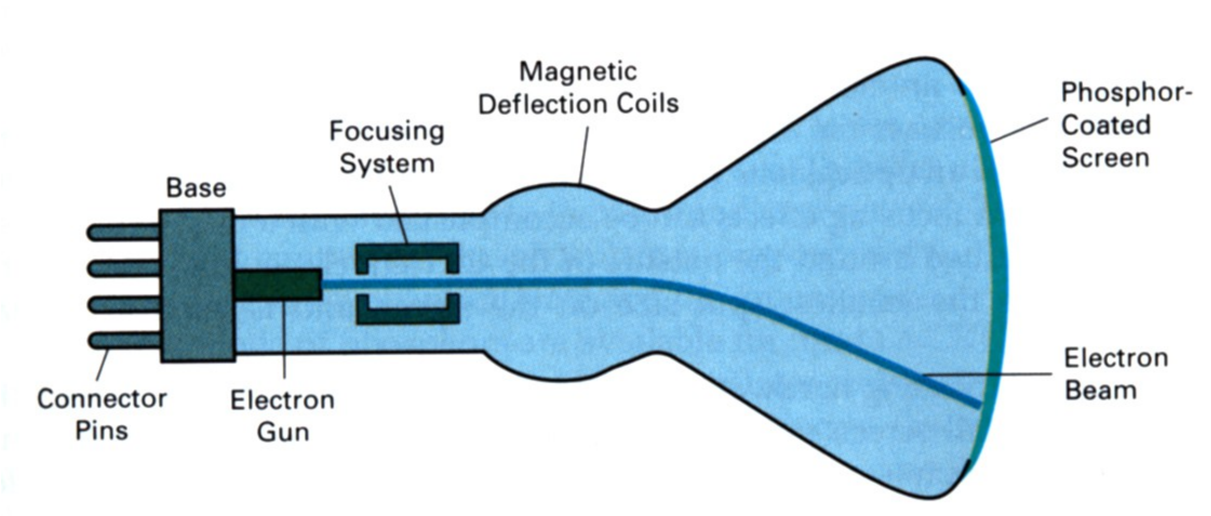
- Raster CRT display
- LCD display
- OLED display, DLP projector, …

Printers

- Laser printer
- Inkjet printer

Cathode ray tube

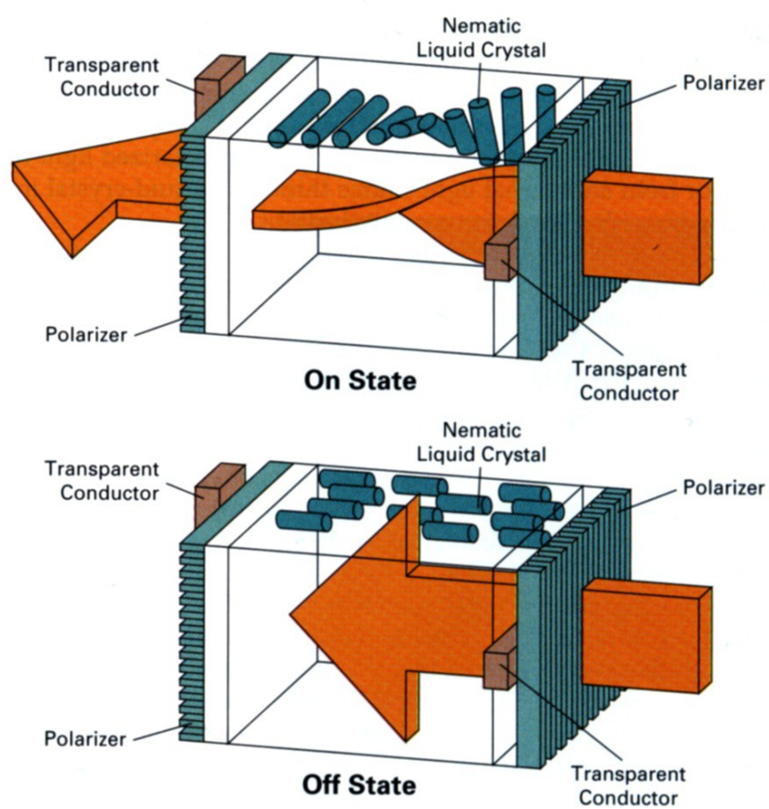
- First widely used electronic display
  - developed for TV in the 1920s–1930s



[H&B fig. 2-2]

LCD flat panel or projection display

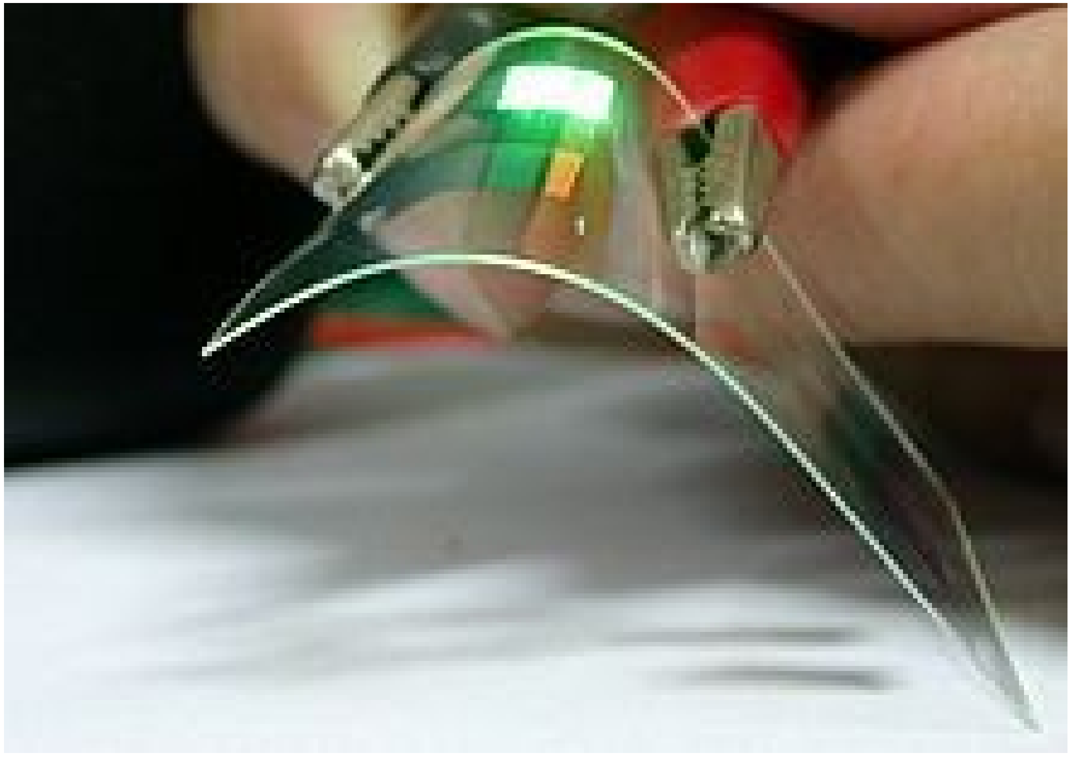
- block or transmit light by twisting its polarization
- Intermediate intensity levels possible by partial twist



[H&B fig. 2-16]

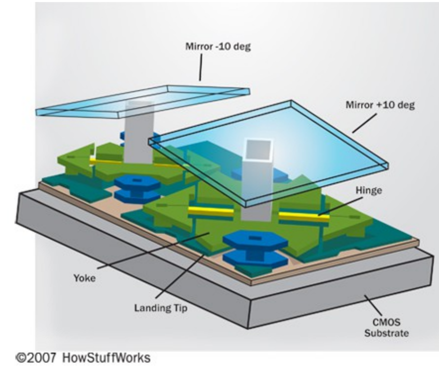
OLED display

- An OLED is a light-emitting diode
- In low ambient light conditions such as a dark room, an OLED can achieve a much higher contrast ratio than an LCD



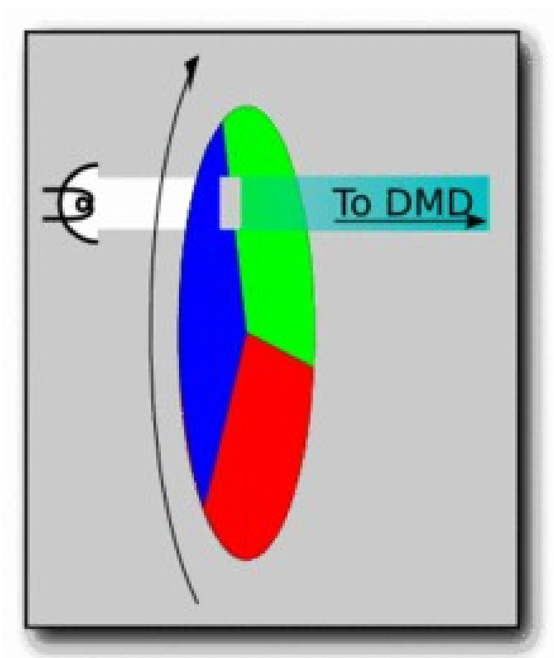
A demonstration of a flexible OLED device (from Wikipedia)

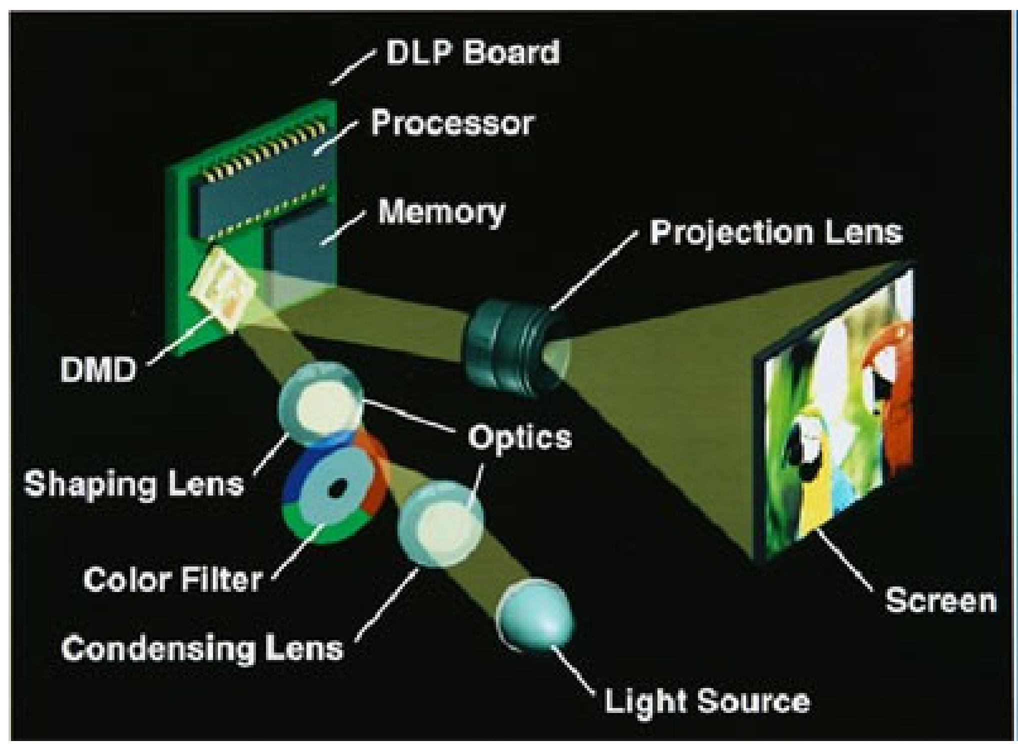
DLP projector



DMD

- Digital light processing
- The image is created by microscopically small mirrors laid out in a matrix on a semiconductor chip, known as a Digital Micromirror Device (DMD)





Color displays (light emiting devices)

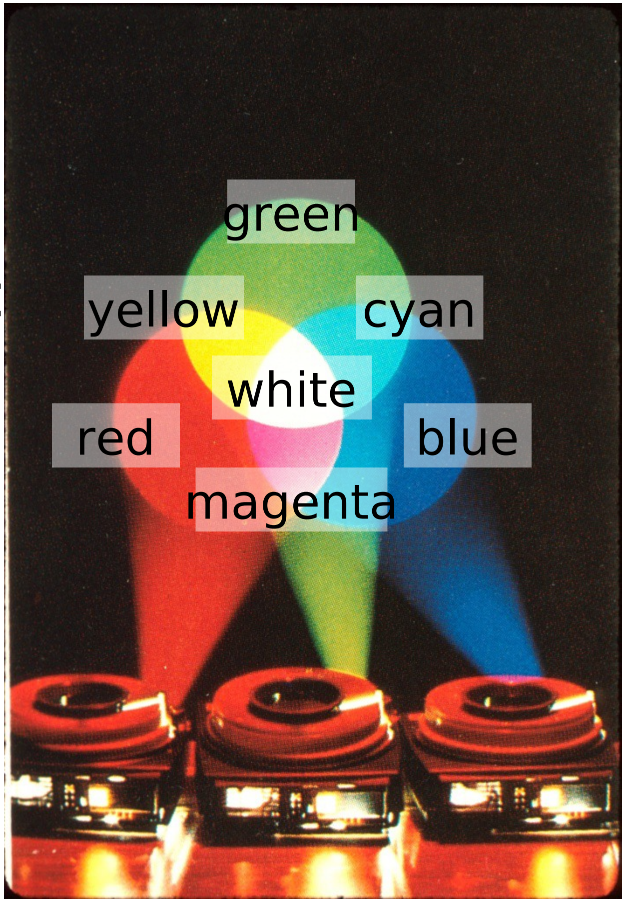
- Operating principle: humans are trichromatic
  - match any color with blend of three
  - therefore, problem reduces to producing 3 images and blending
- Additive color
  (as opposed to subtractive c
  - blend images by sum
  - e.g. overlapping projection
  - R, G, B make good primaries

R=(1,0,0)

G=(0,1,0)

B=(0,0,1)

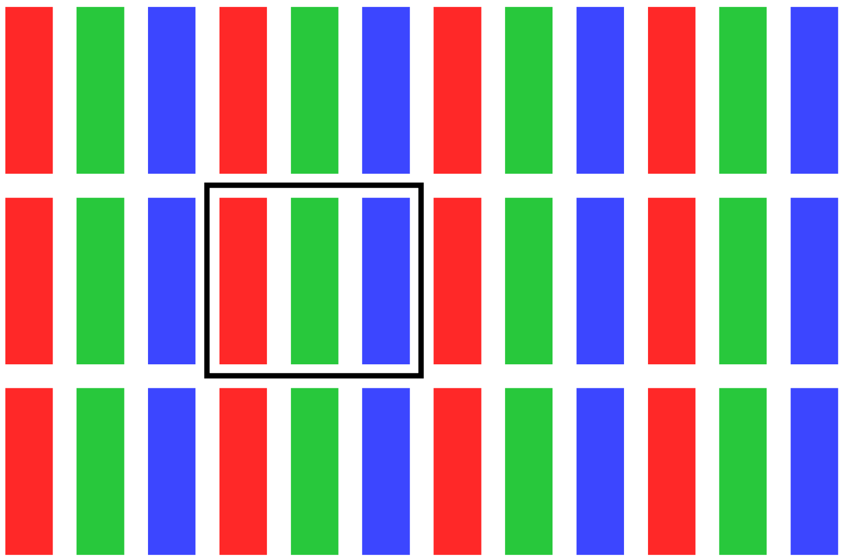
Cyan=G+B=(0,1,1)

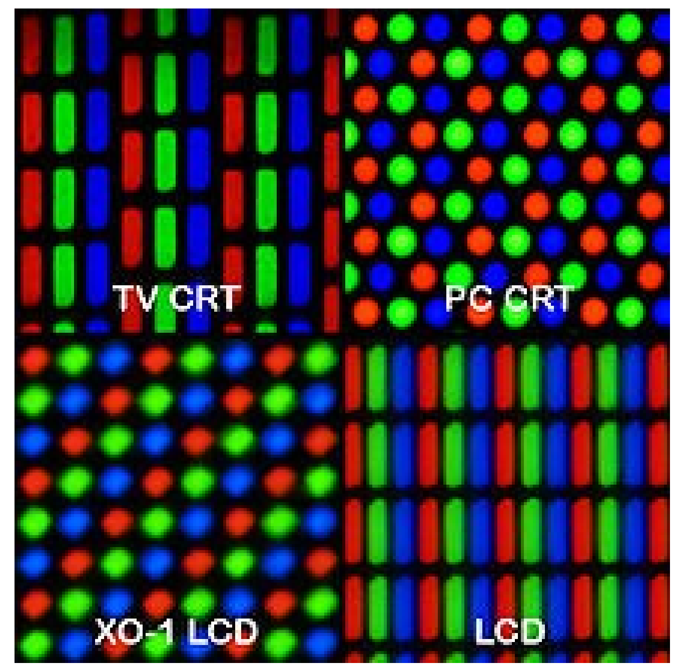


[source unknown]

Color displays

- interleaved R,G,B pixels (except DLPs)



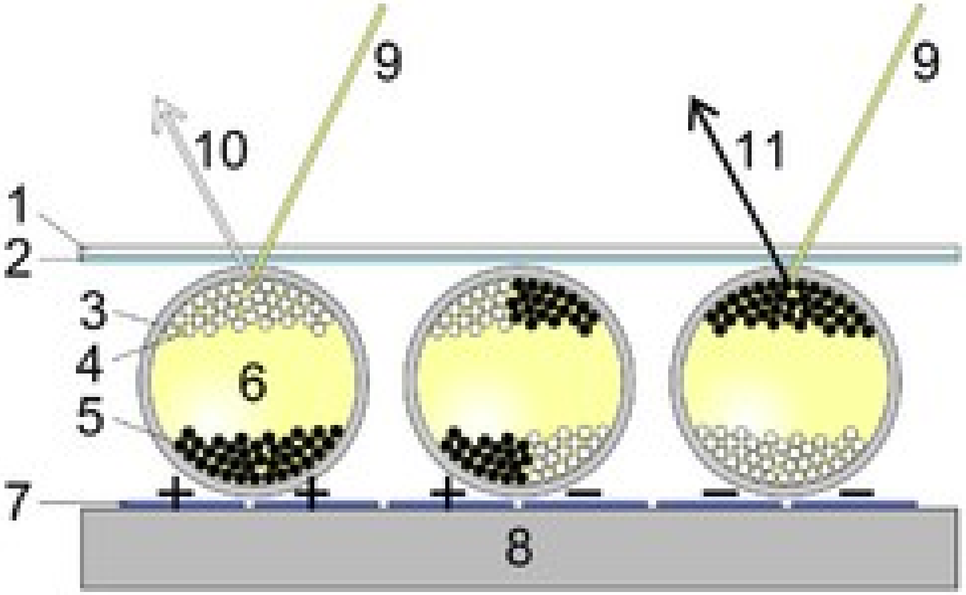


[H&B fig. 2-10]

Electronic paper (E Ink)

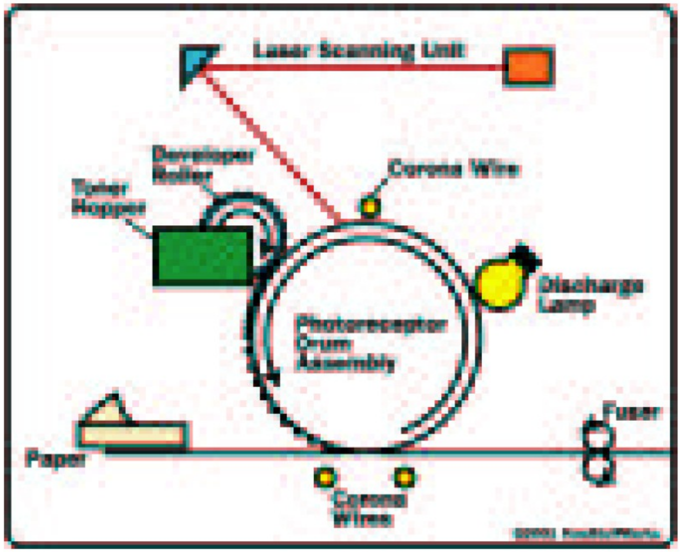
- In an extremely bright condition, E-ink outperforms all light-emitting displays

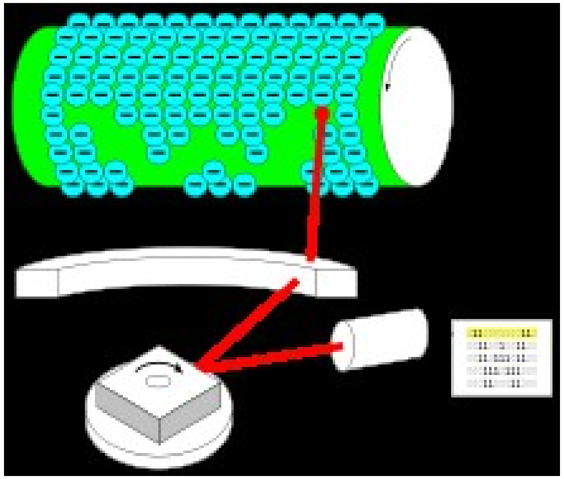
An e-paper display visible in the sunlight



Laser printer

- Like a photocopier but with laser-scanned raster as source image
- Key characteristics
  - image is binary
  - resolution is high
  - very small, isolated dots are not possible

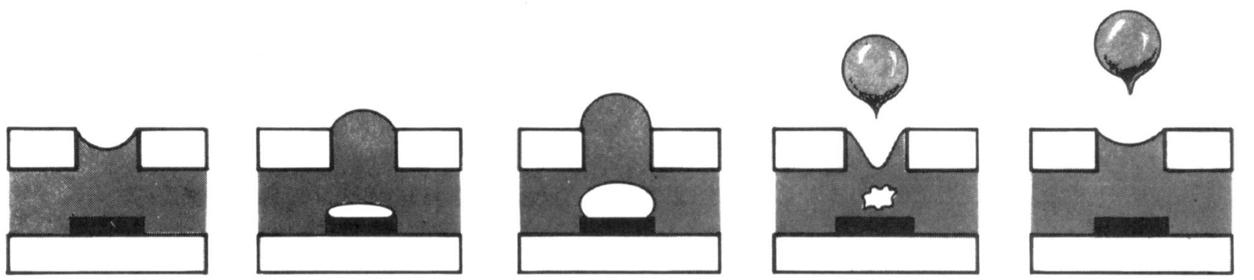




[howstuffworks.com]

Inkjet printer

- Liquid ink sprayed in small drops
- Head with many jets scans across paper
- Key characteristics:
  - image is binary (drop or no drop; no partial drops)
  - isolated dots are reproduced well

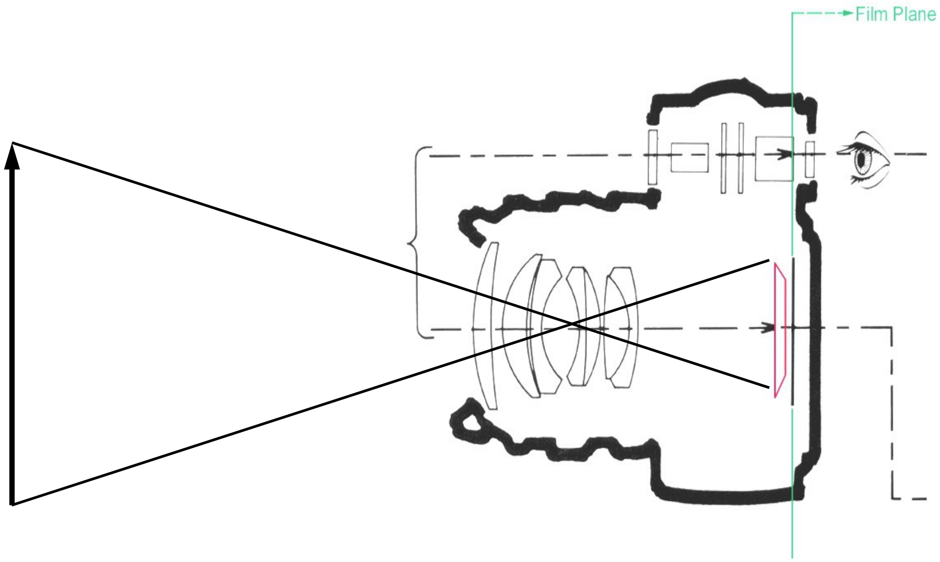


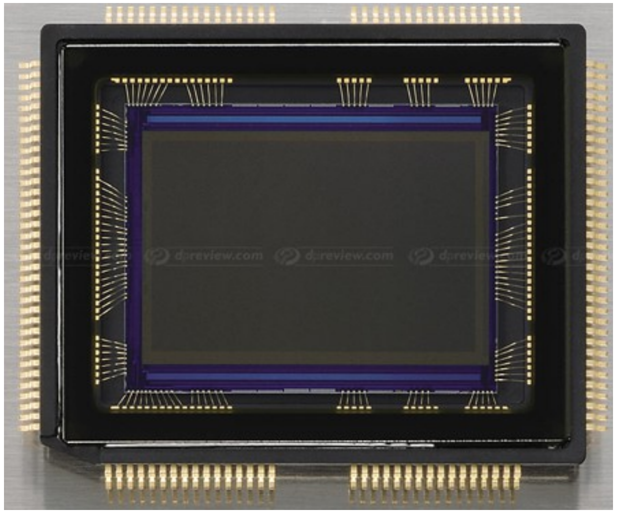
[source unknown]

Digital camera

- A raster input device
- Image sensor contains 2D array of photosensors

[CS 417 Spring 2002]

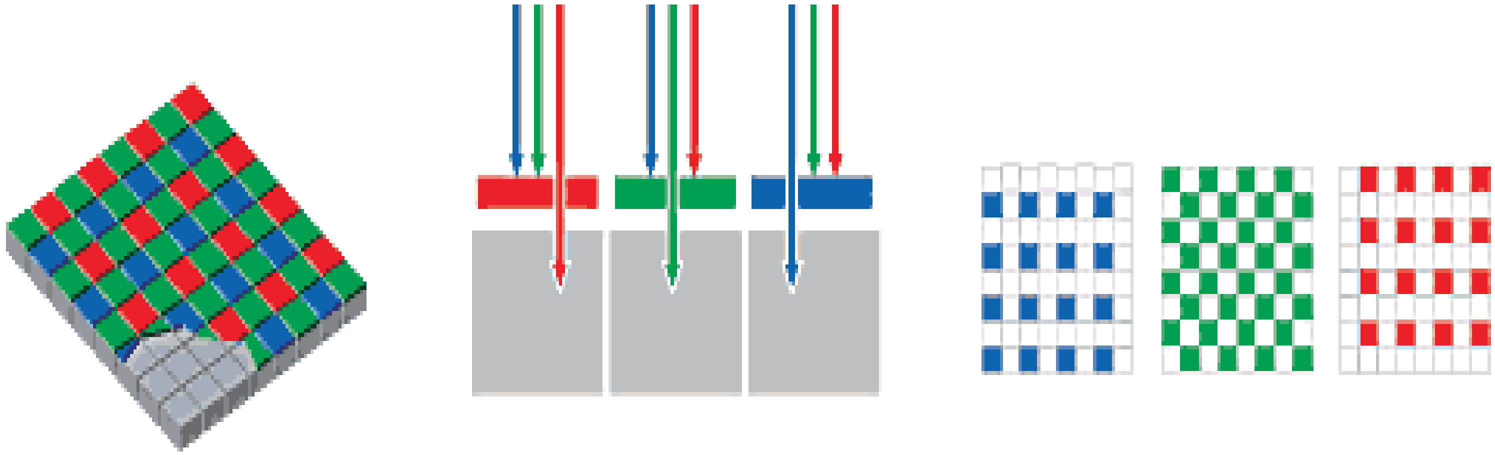




[dpreview.com]

Digital camera

- Color typically captured using color mosaic



[Foveon]

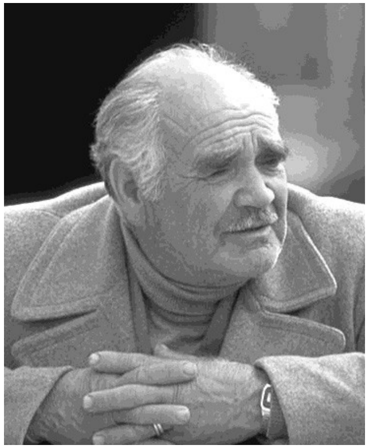
Raster image representation (Pixels as opposed to vectors)

- All these devices suggest 2D arrays of numbers (pixels)
- Big advantage: represent arbitrary images
  - approximate arbitrary functions with increasing resolution
  - works because memory is cheap (brute force approach!)

[Philip Greenspun]

Datatypes for raster images

- Bitmaps: `boolean` per pixel (1 bpp):
  - black and white
- Grayscale: integer per pixel:
  - shades of gray; e.g. black-and-white print
  - precision: usually `byte` (8 bpp); sometimes 10, 12, or 16 bpp
  - E-ink: 4bpp (16 grey levels)

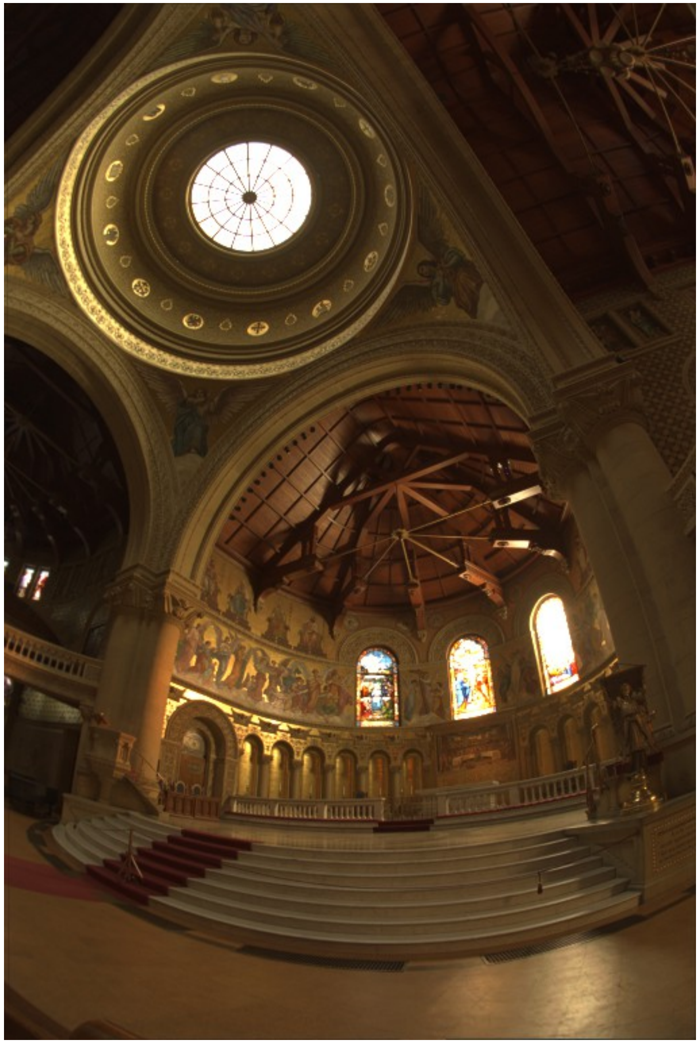


- Color: 3 integers per pixel:
  - full range of displayable color
  - precision: usually `byte[3]` (24 bpp)
  - sometimes 16 (5+6+5) or 30 or 36 or 48 bpp
  - indexed color (palletized color : usually 8bpp): a fading idea. GIF format

Datatypes for raster images

- Floating point:
  - more precision
  - provides *high dynamic range* (HDR)

24bpp image



HDR - 72bpp image

Higher bits

Lower bits

HDR images are good for post-processing (exposure compensation) - badly taken photos can be recovered

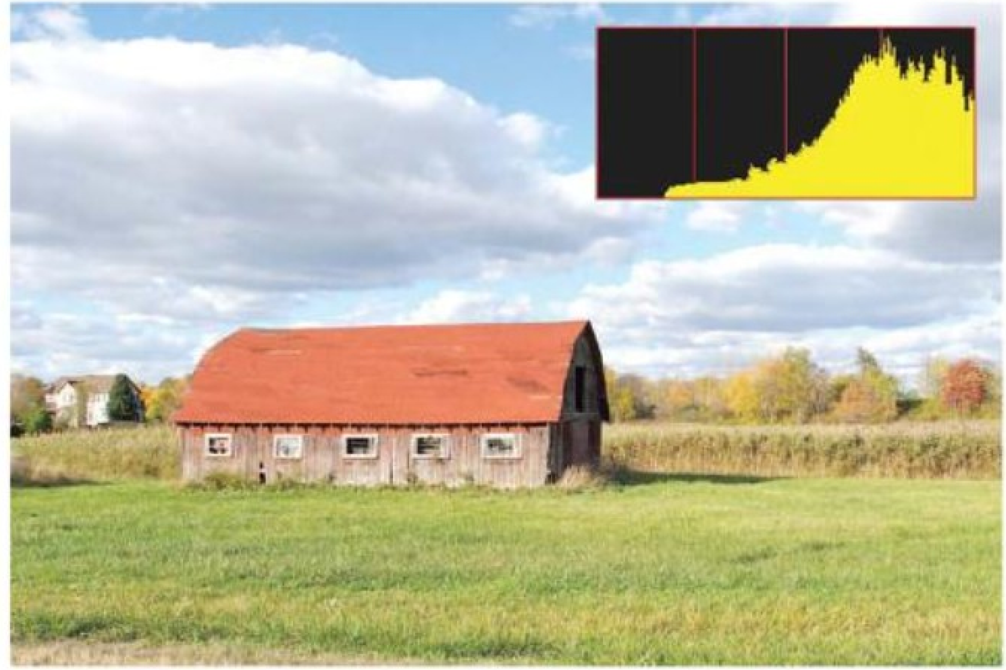


Figure 6.26
This histogram reveals that the image is overexposed.

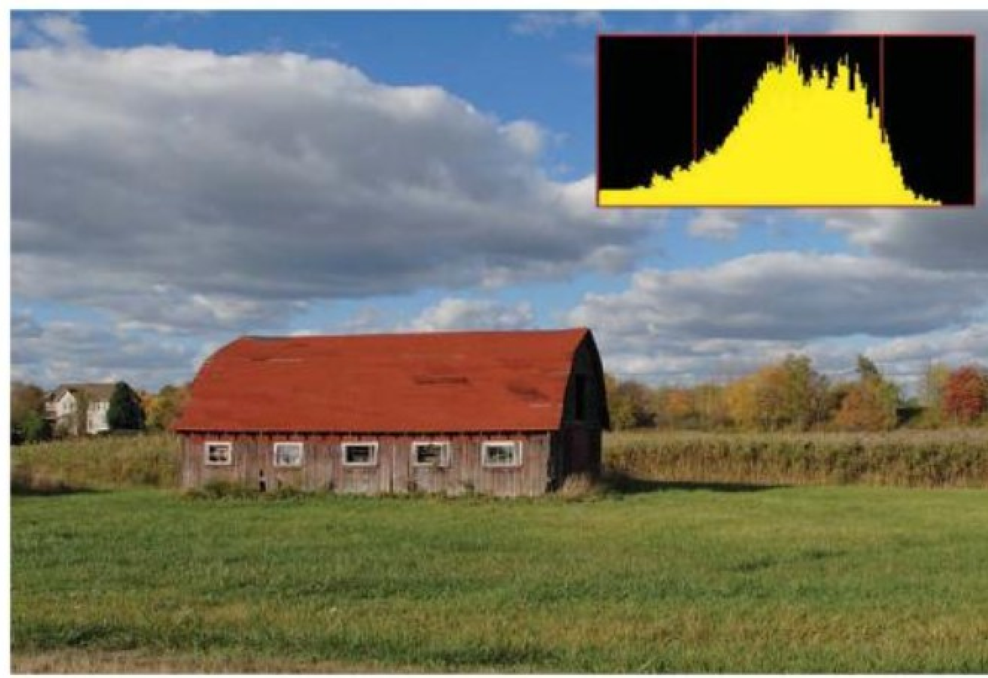


Figure 6.27
A histogram for a properly exposed image should look like this.

exposure:
-8 stops

image: Paul Debevec

exposure:
+0 stops

image: Paul Debevec

exposure:
+6 stops

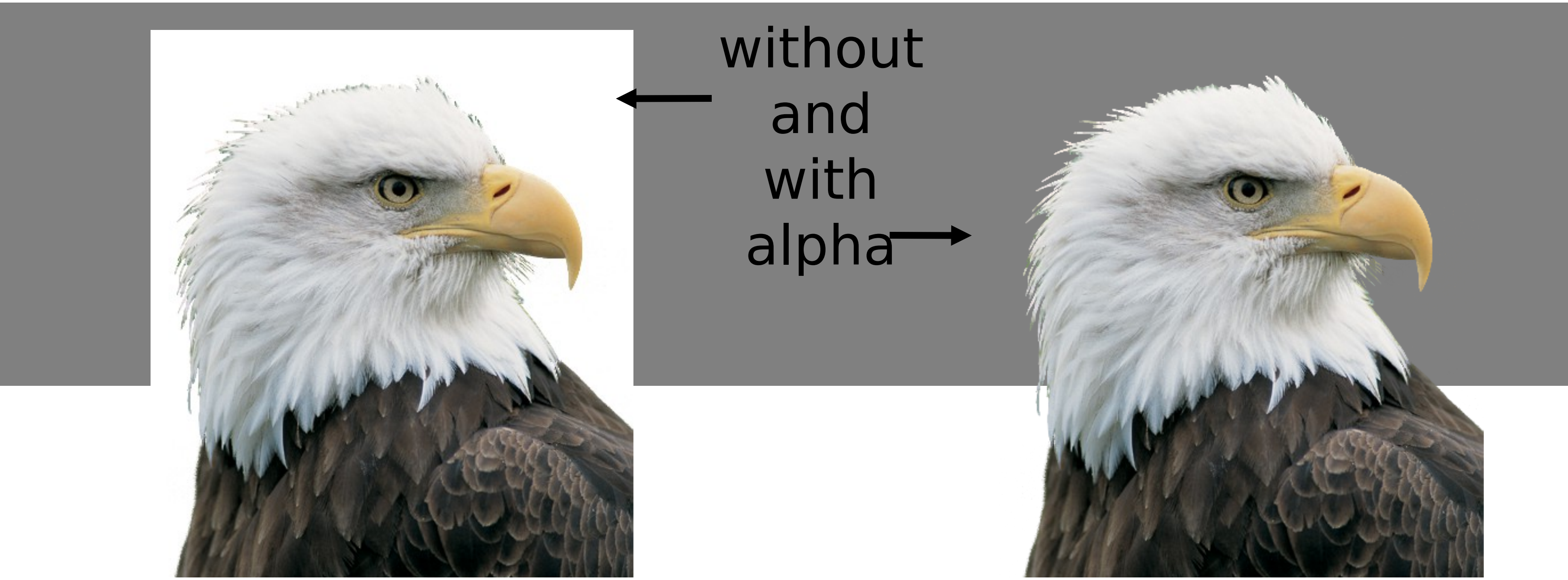
image: Paul Debevec

Datatypes for raster images

- Floating point:
  - more precision
  - provides *high dynamic range* (HDR)
  - represent real scenes independent of display
  - becoming the standard intermediate format in graphics processors

Datatypes for raster images

- For color or grayscale, sometimes add *alpha* channel
  - describes transparency of images
  - more on this in a few lectures



[Adobe Photoshop sample]

Storage requirements for images

- 1024x1024 image (1 megapixel)
  - bitmap: 128KB
  - grayscale 8bpp: 1MB
  - grayscale 16bpp: 2MB
  - color 24bpp: 3MB
  - floating-point HDR color: 12MB

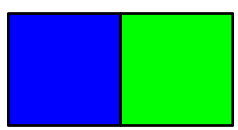
Image processing

- Color to gray
- Down pixel precision (16bpp → 4bpp)
  - Brute-force
  - Dithering
- Gamma correction

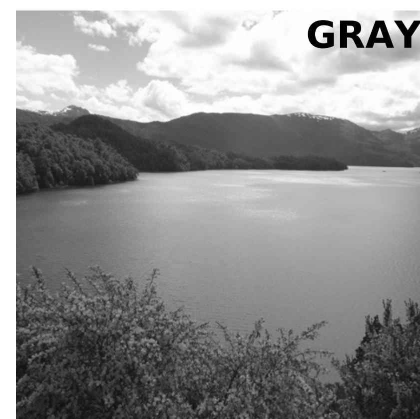
Converting pixel formats

- Color to gray
  - could take one channel (blue, say)
    - leads to odd choices of gray value
  - combination of channels is better
    - but different colors contribute differently to lightness
    - which is lighter, full blue or full green?
    - good choice: gray = 0.2 R + 0.7 G + 0.1 B
    - more on this in color, later on

Same pixel values.

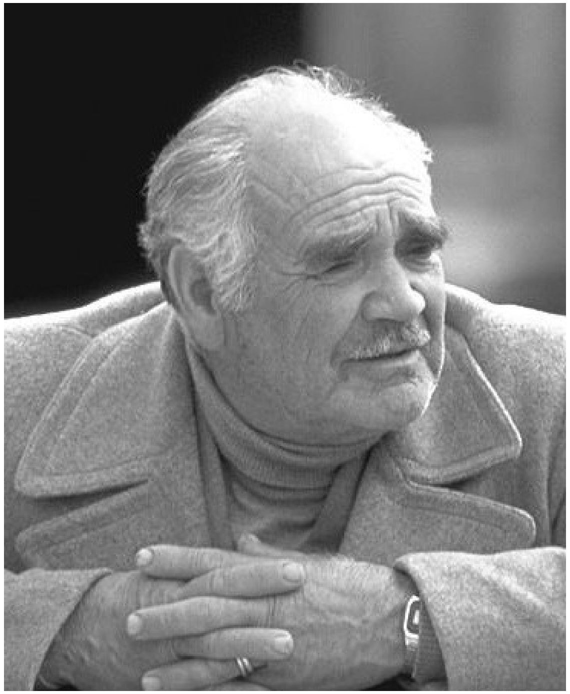


Same luminance?



Converting pixel precision

- Up is easy; down loses information—be careful

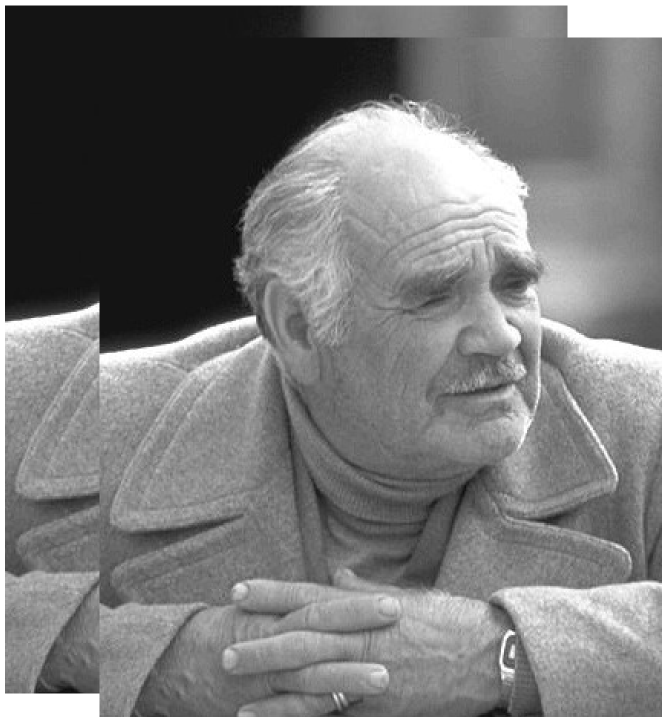


8 bpp (256 grays)

[photo: Philip Greenspun]

Converting pixel precision

- Up is easy; down loses information—be careful

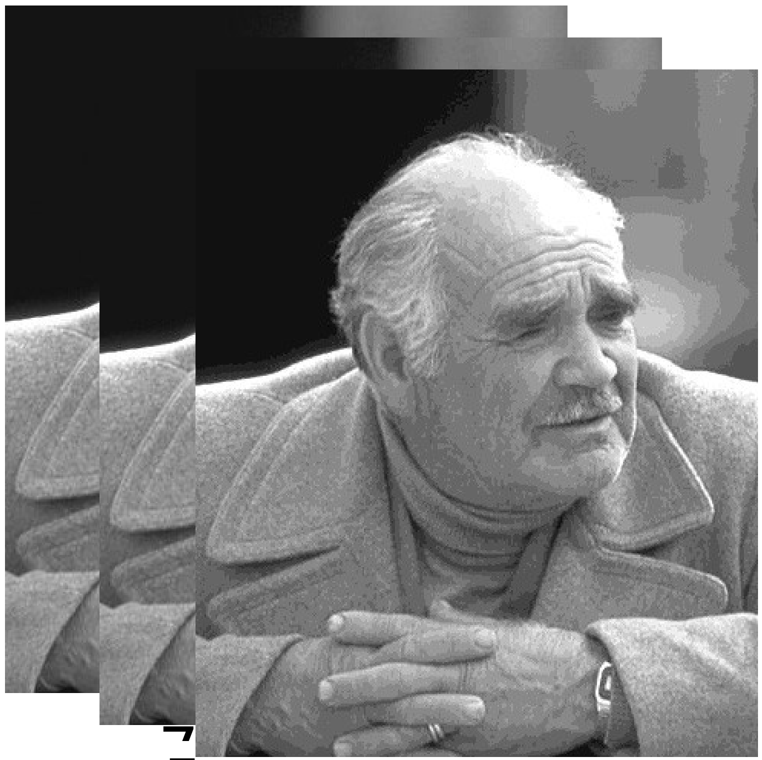


5 bpp (32 grays)

[photo: Philip Greenspun]

Converting pixel precision

- Up is easy; down loses information—be careful

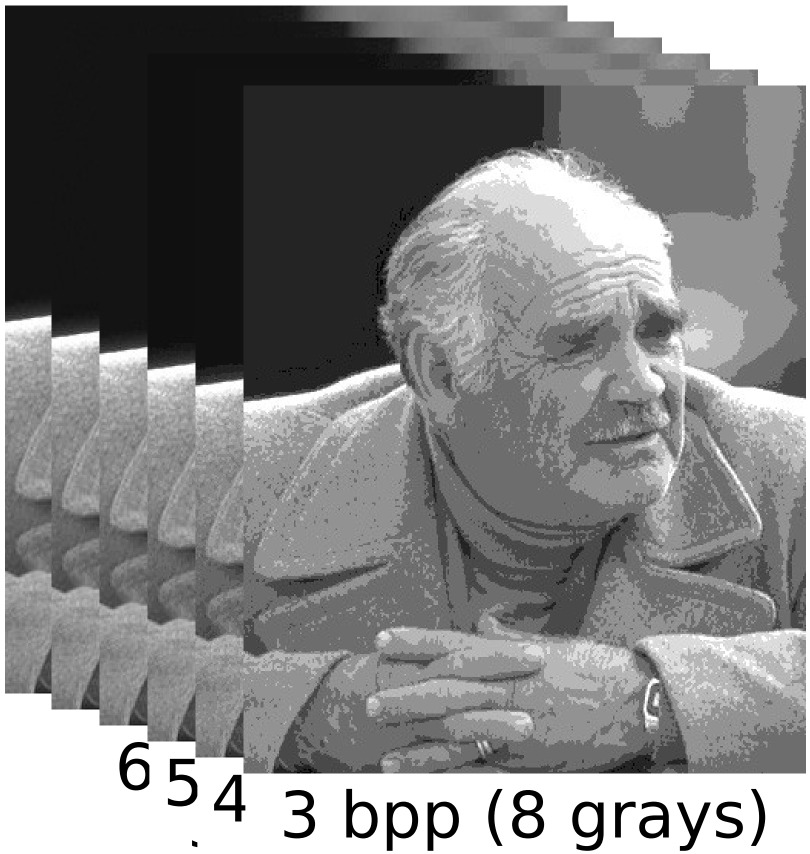


4 bpp (16 grays)

[photo: Philip Greenspun]

Converting pixel precision

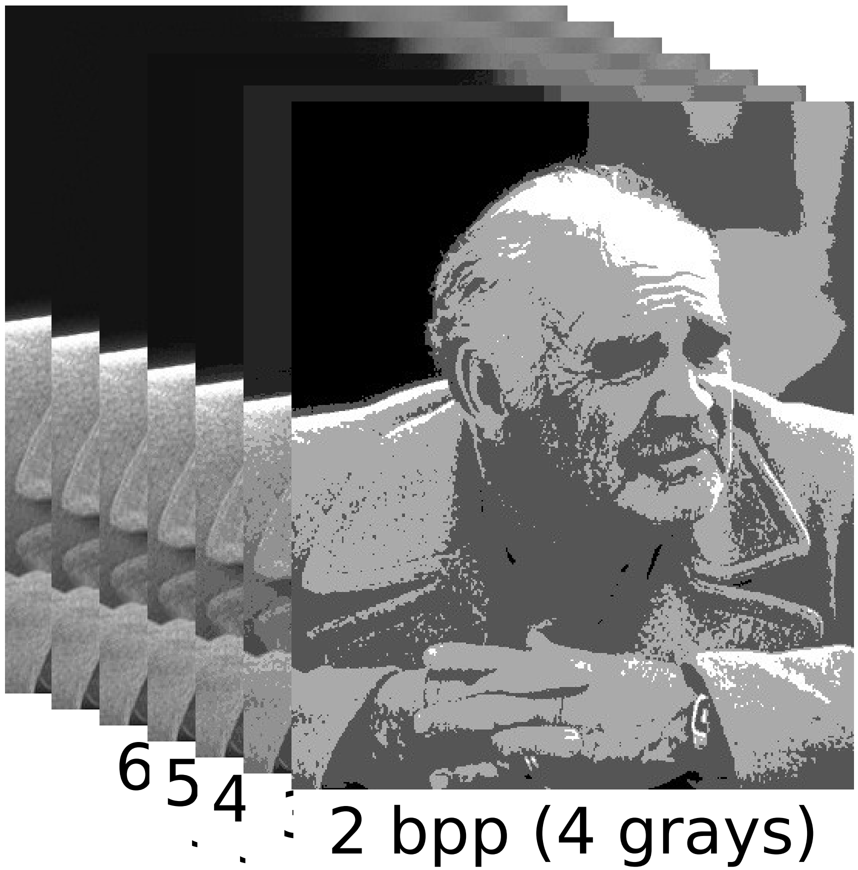
- Up is easy; down loses information—be careful



[photo: Philip Greenspun]

Converting pixel precision

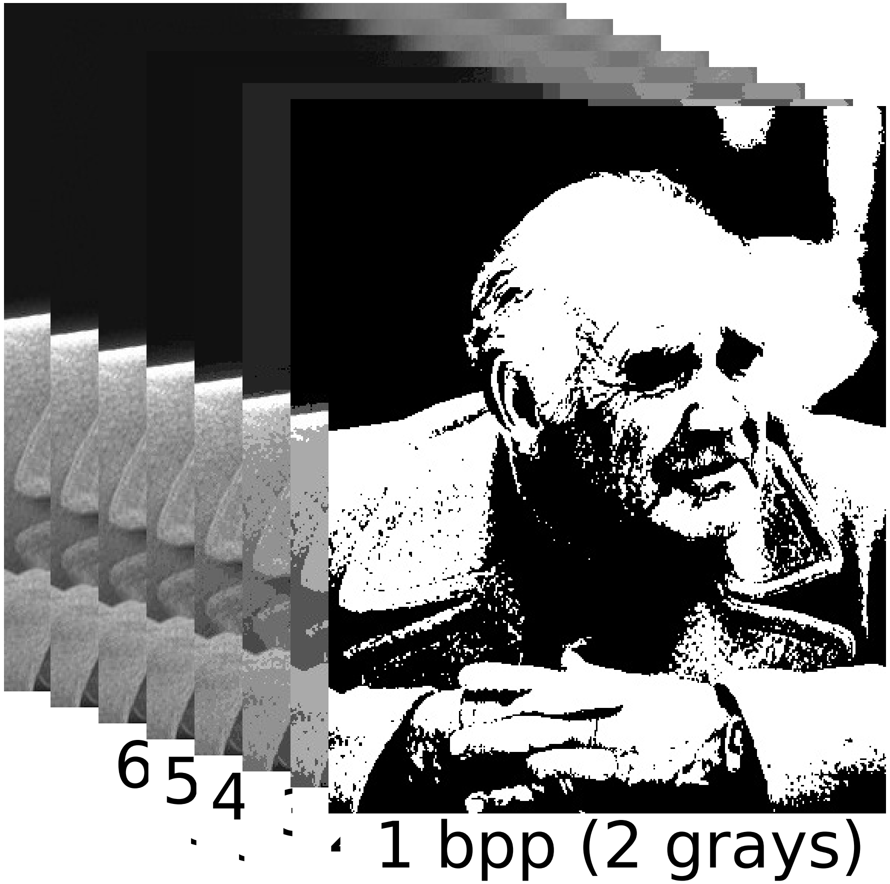
- Up is easy; down loses information—be careful



[photo: Philip Greenspun]

Converting pixel precision

- Up is easy; down loses information—be careful

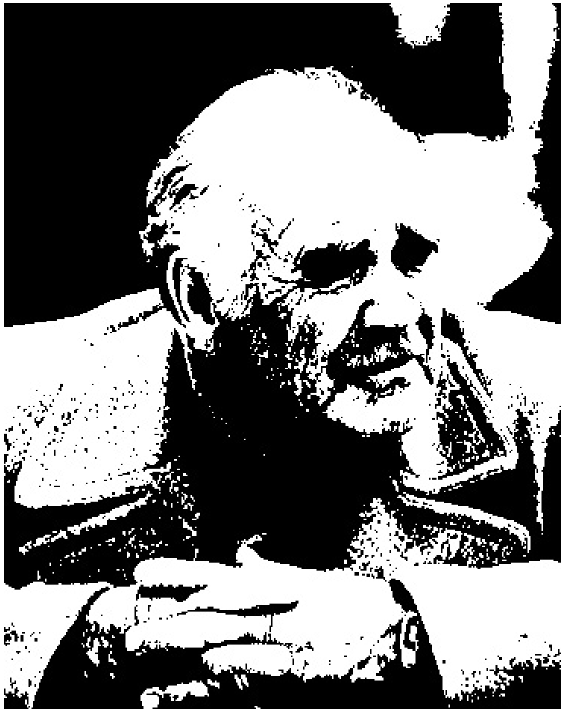


Lacks gradation

[photo: Philip Greenspun]

Dithering

- When decreasing bpp, we quantize
- Make choices consistently: banding artifacts
  - I =0 if I <0.5
  - I =1 if I >= 0.5
- Instead, be inconsistent—dither
  - turn on some pixels but not others in gray regions
  - a way of trading spatial for tonal resolution



Laser printed image WITHOUT dithering

[Philip Greenspun]

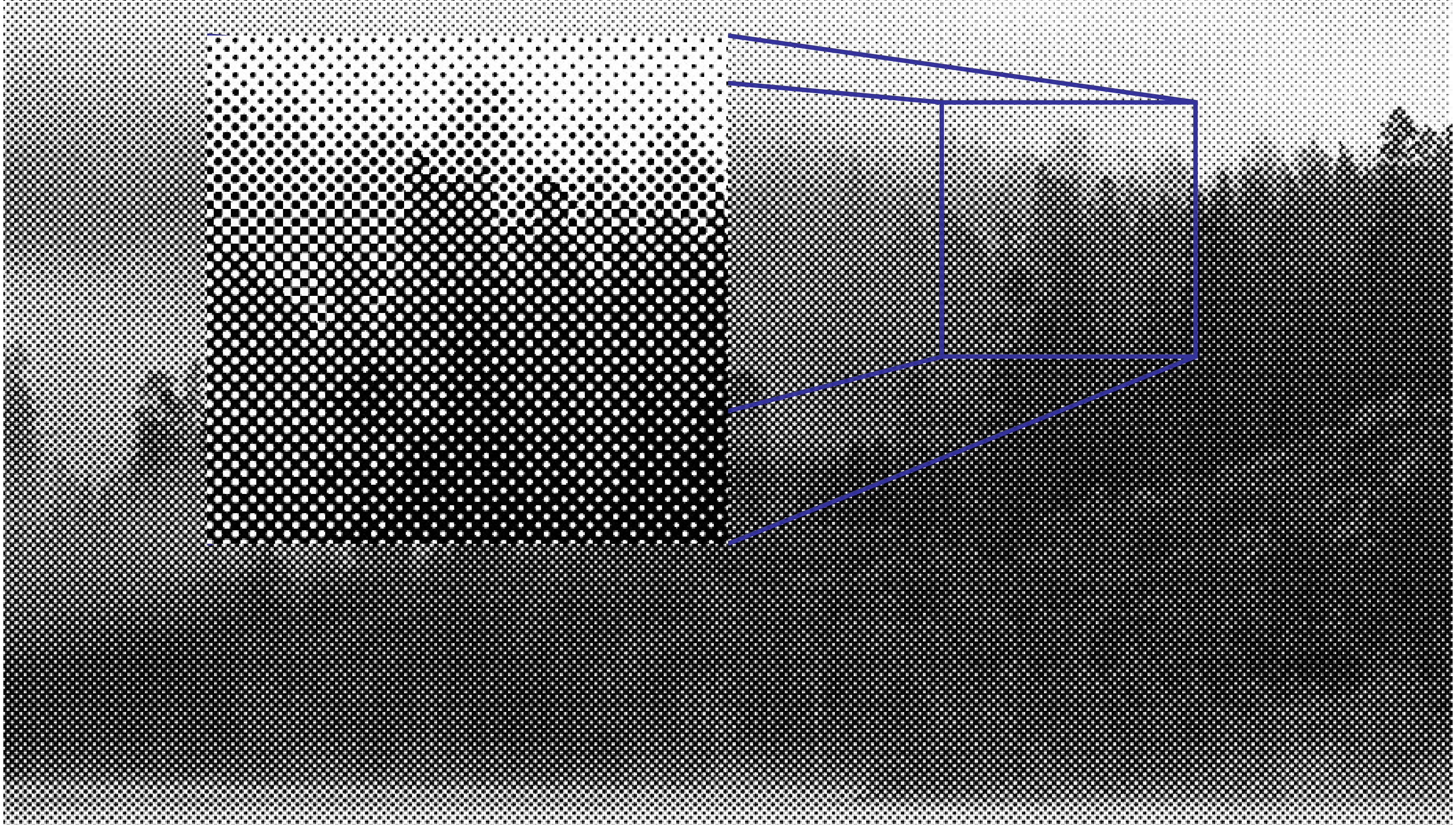
Dithering methods

- Ordered dither
  - produces larger dots
  - laser printer
- Diffusion dither
  - takes advantage of devices that can reproduce isolated dots
  - inkjet, screen

[Philip Greenspun]

Ordered Dither example

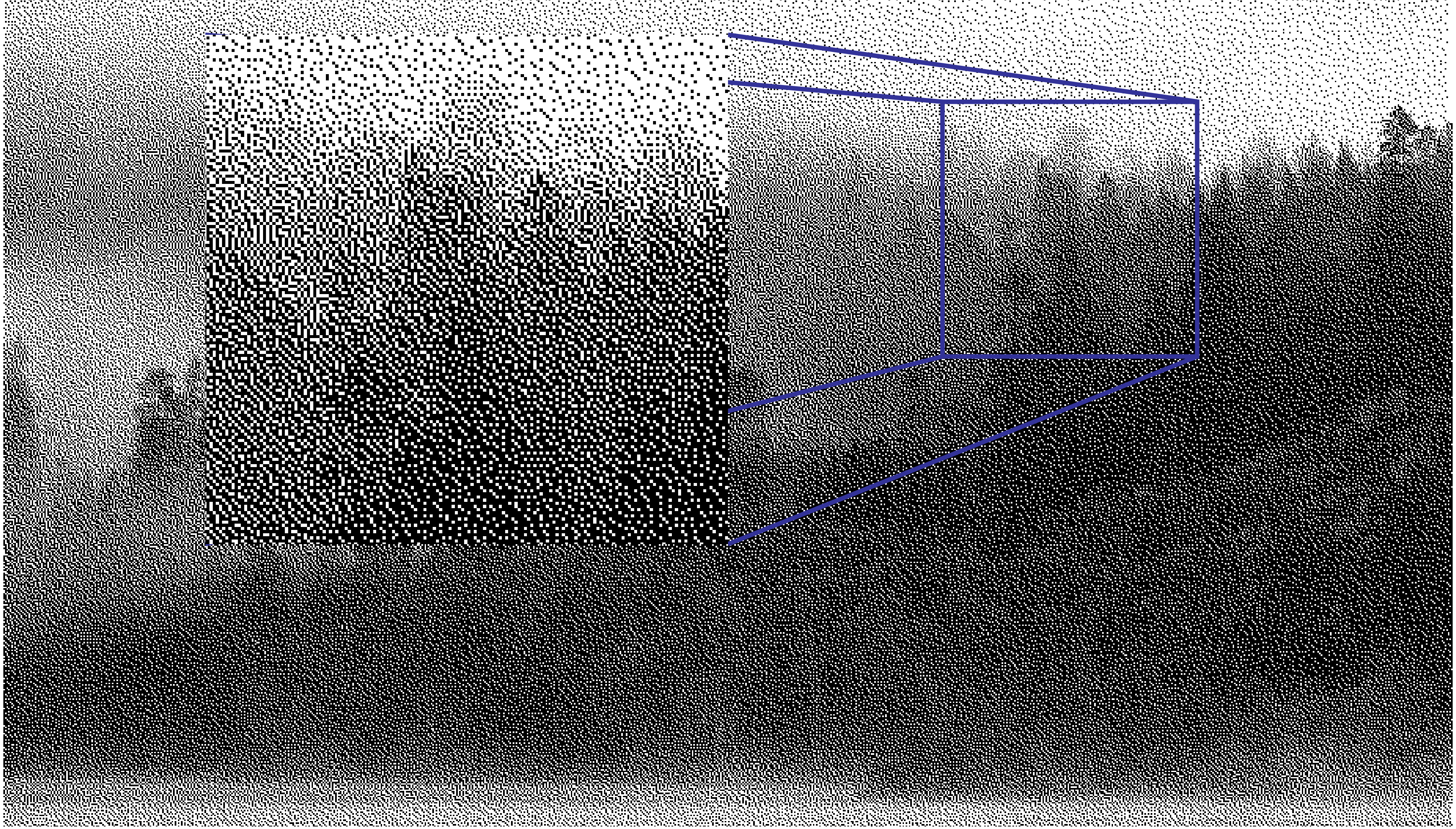
- Produces regular grid of compact dots



[photo: Philip Greenspun]

Diffusion dither

- Produces scattered dots with the right local density

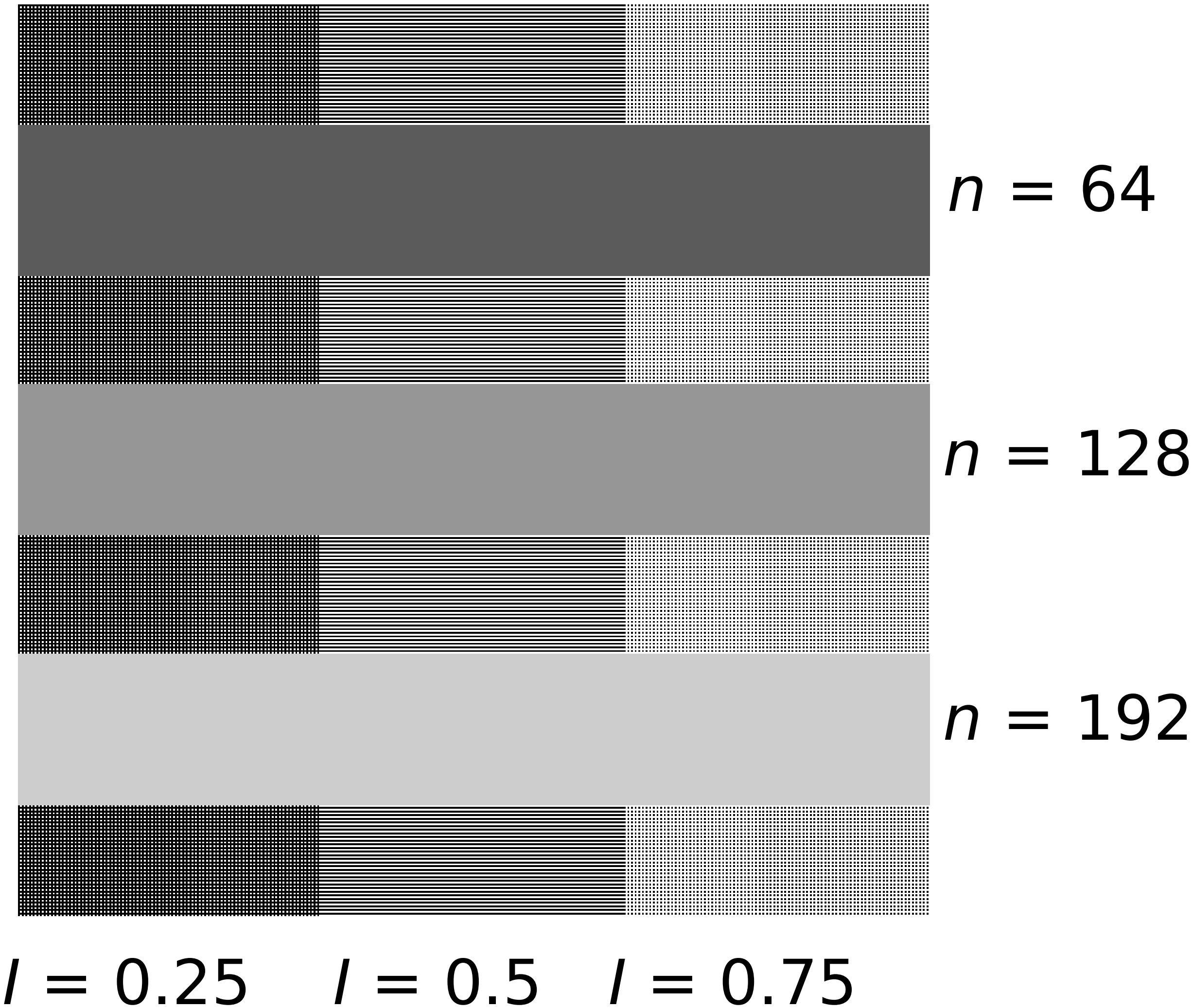


[photo: Philip Greenspun]

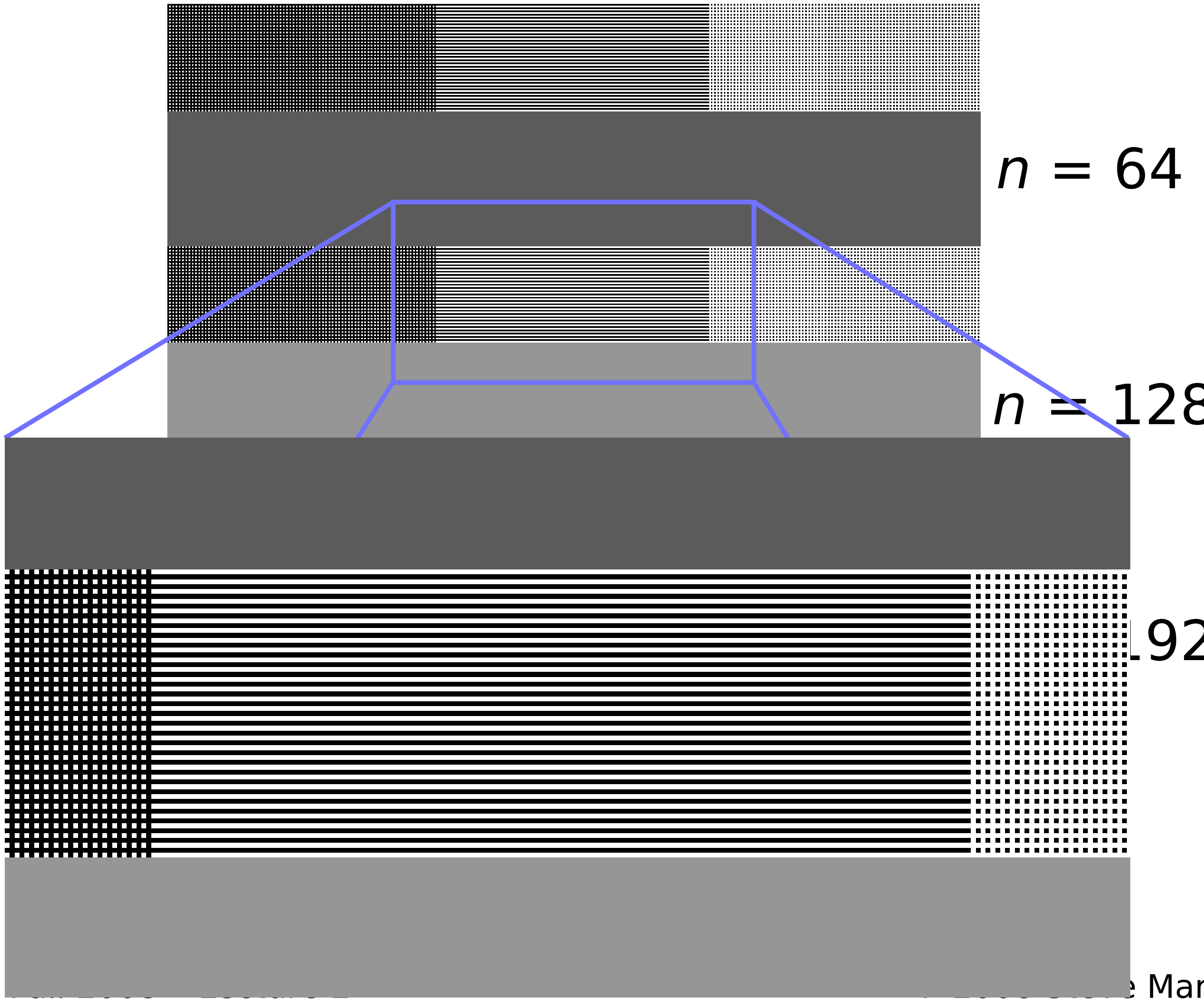
Pixel Values vs Light Intensity

- What do the numbers in images (pixel values) mean?
  - they determine how bright that pixel is
  - bigger numbers are (usually) brighter
- Is a pixel with value 200 twice as bright as another pixel with value 100?

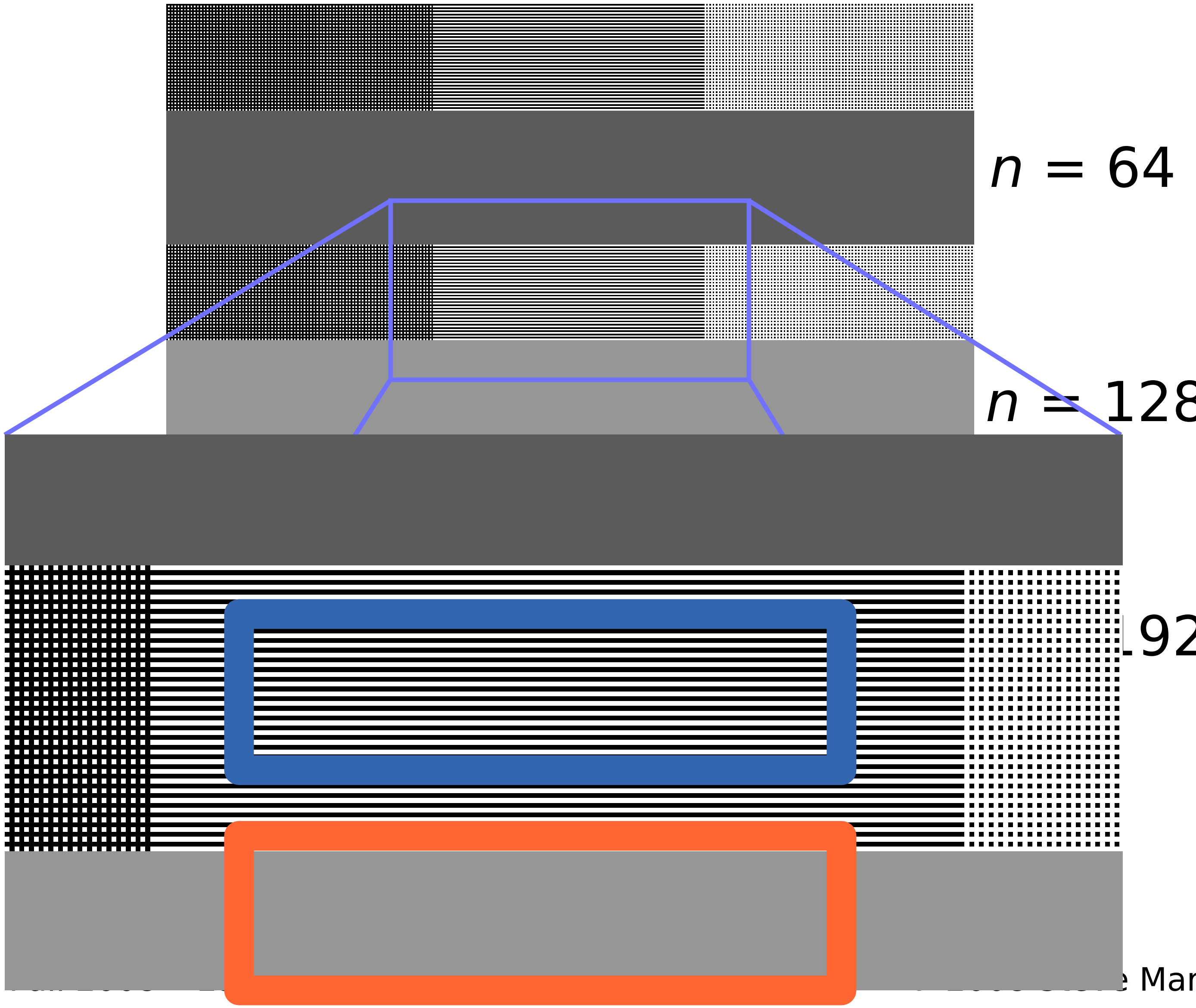
What this projector does



What this projector does

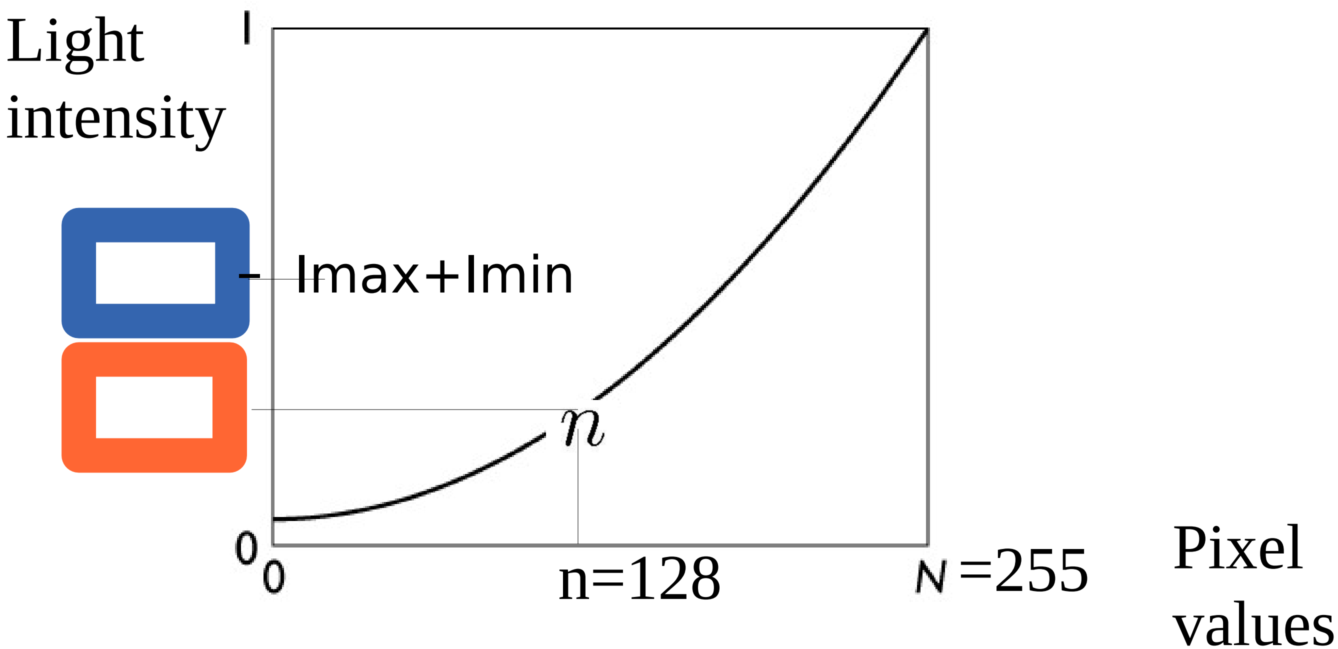


What this projector does



What this projector does

- Something like this if the blue boxed region look brighter than the red box (n=128:)
- Why?
  - Boxed region: I=0.5 (Imax+Imin)
  - I(128)<0.5 (Imax+Imin)

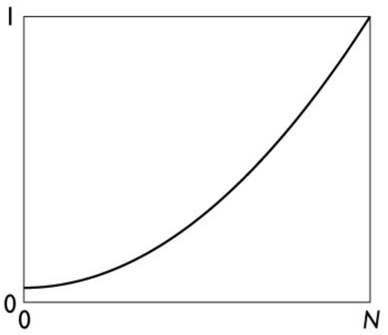


What do the numbers in images (pixel values) mean?

- they determine how bright that pixel is
  - bigger numbers are (usually) brighter
- *Transfer function*: function that maps input pixel value to luminance of displayed image

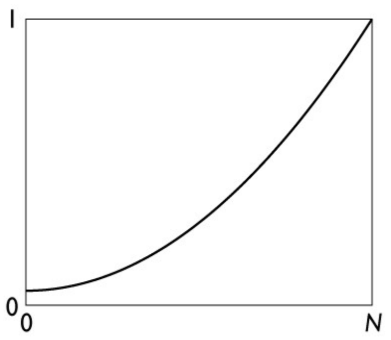
$$I = f(n) \quad f : [0, N] \to [I_{\min}, I_{\max}]$$

- What determines this function?
  - physical constraints of device or medium
  - desired visual characteristics



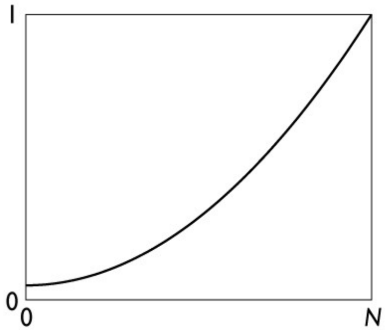
Constraints on transfer function (for light-emitting displays)

- Maximum displayable intensity, I_{max}
  - how much power can be channeled into a pixel?
    - LCD: backlight intensity, transmission efficiency (<10%)
    - projector: lamp power, efficiency of imager and optics
- Minimum displayable intensity, I_{min}
  - light emitted by the display in its “off” state
    - *e.g.* stray electron flux in CRT, polarizer quality in LCD



Viewing flare, *k*: light reflected by the display

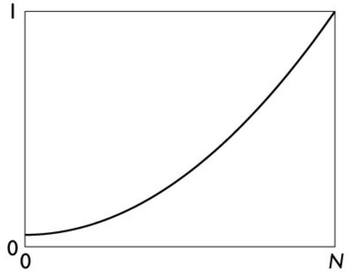
- very important factor determining image contrast in practice
  - 5% of I_{max} is typical in a normal office environment [sRGB spec]
  - much effort to make very black CRT and LCD screens
  - all-black decor in movie theaters



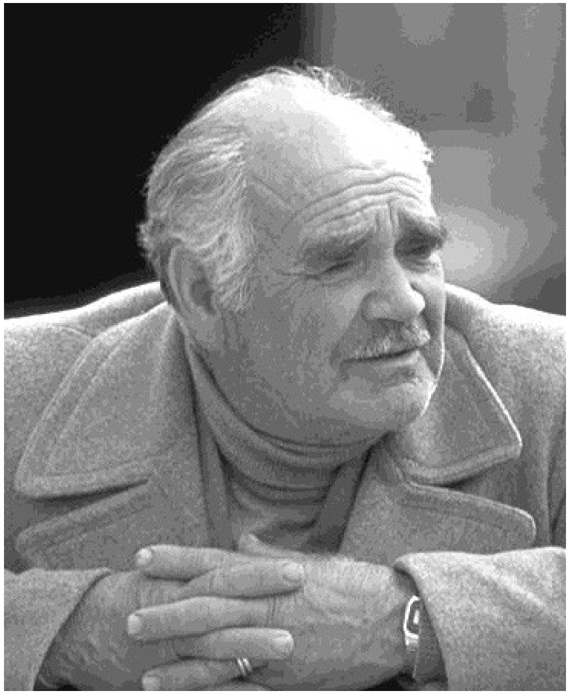
Dynamic range

- Dynamic range $R_d = I_{\text{max}} / I_{\text{min}}$, or $(I_{\text{max}} + k) / (I_{\text{min}} + k)$
  - determines the degree of image contrast that can be achieved
  - a major factor in image quality
- Ballpark values
  - Desktop display in typical conditions: 20:1
  - Photographic print: 30:1
  - Desktop display in good conditions: 100:1
  - Desktop display in a dark room: 1000:1 ~ inf:1
  - Photographic transparency (directly viewed): 1000:1
  - High dynamic range display in good conditions: 10,000:1 (very bright white)

Transfer function shape



- Desirable property: the change from one pixel value to the next highest pixel value should not produce a visible contrast
  - otherwise smooth areas of images will show visible bands
- What contrasts are visible?
  - rule of thumb: under good conditions we can notice a 2% change in intensity
  - therefore we generally need smaller quantization steps in the darker tones than in the lighter tones
  - most efficient quantization is logarithmic



an image with severe *banding*

[Philip Greenspun]

How many levels are needed?

- Ideal case : exponential quantization
  - Depends on dynamic range
  - 2% steps are most efficient:

$$0 \mapsto I_{\mathrm{min}}; 1 \mapsto 1.02 I_{\mathrm{min}}; 2 \mapsto (1.02)^2 I_{\mathrm{min}}; \ldots$$

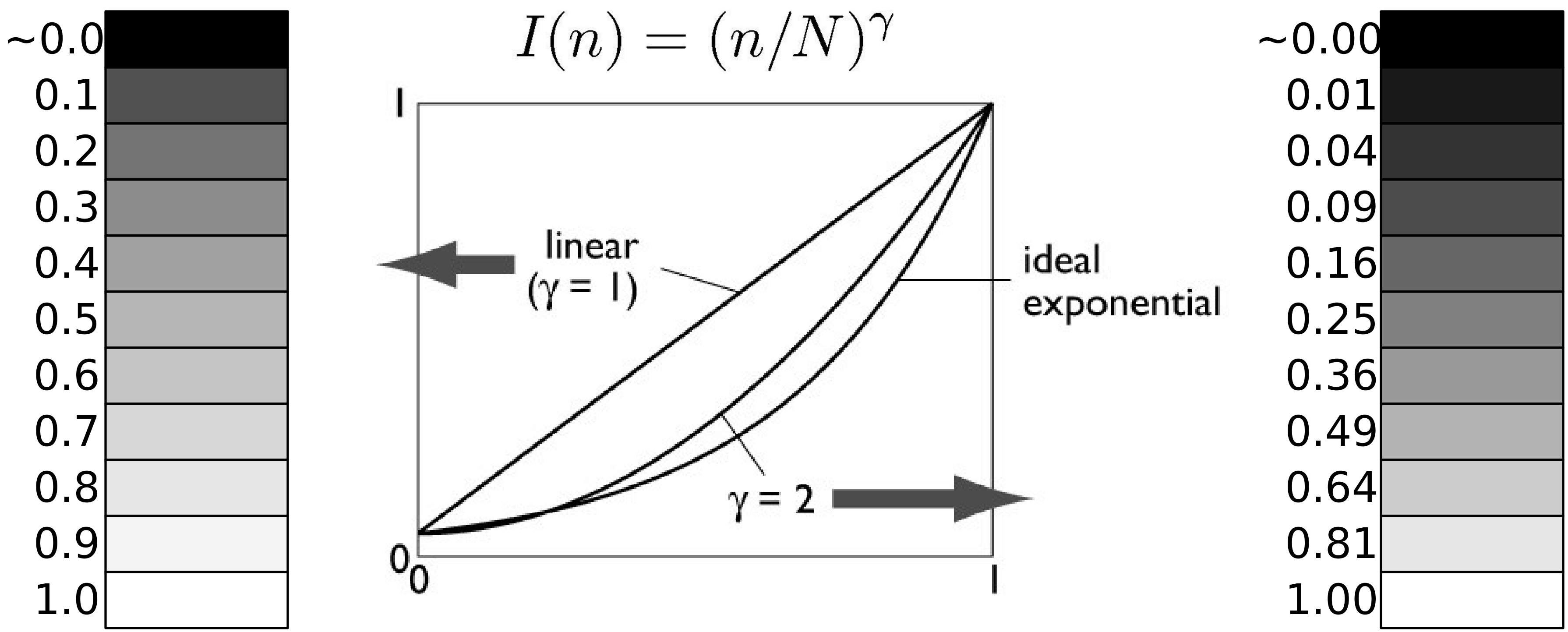
  - log 1.02 is about 1/120, so 120 steps are needed for 10 times intensity

  → 240 steps are needed for 100:1 dynamic range
- Moral: 8 bits is just barely enough for low-end applications
  - even if we are careful about quantization

Intensity quantization in practice

- Option 1: linear quantization $I(n) = (n/N)\, I_{\max}$
  - pro: simple, convenient, amenable to arithmetic
  - con: requires more steps (wastes memory)
  - need 12 bits for any useful purpose; more than 16 for HDR
- Option 2: Ideal exponential quantization
  - Too expensive $I(n) = (1.02)^n I_{min}$
- Option 3: power-law quantization $I(n) = (n/N)^\gamma\, I_{\max}$
  - pro: fairly simple, approximates ideal exponential quantization
  - con: need to linearize before doing pixel arithmetic
  - con: need to agree on exponent (standard gamma: 2.2)
  - 8 bits are OK for many applications

Option3 : Gamma quantization



- Close enough to ideal perceptually uniform exponential

Gamma correction

- Sometimes (often, in graphics) we have computed intensities *a* that we want to display linearly
- In the case of an ideal monitor with zero black level,

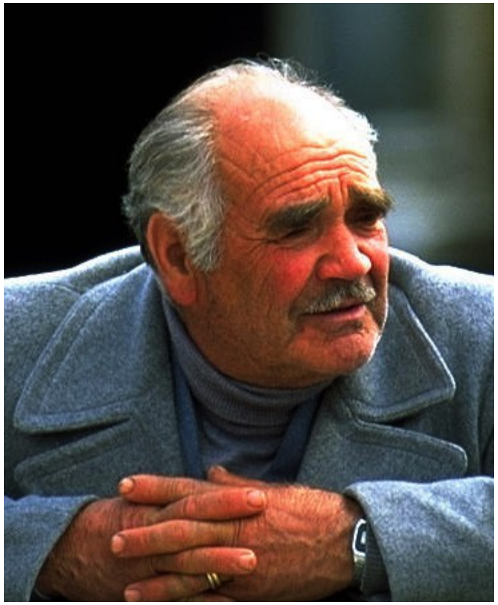
$$I(n) = (n/N)^\gamma$$

  (where N = 2^n – 1 in *n* bits). Solving for *n*:

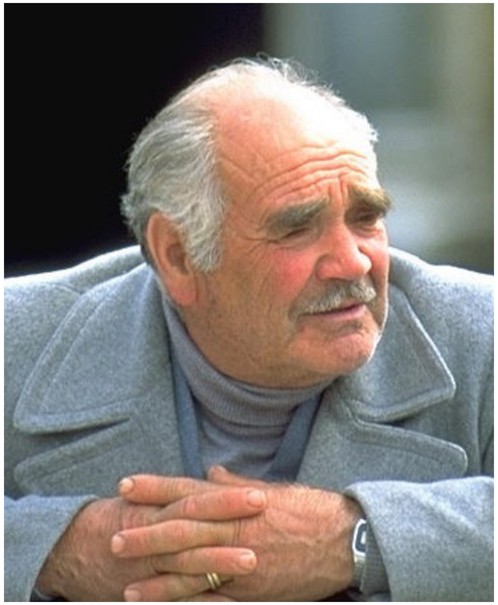
$$n = Na^{\frac{1}{\gamma}}$$

- This is the "gamma correction" recipe that has to be applied when computed values are converted to 8 bits for output
  - failing to do this (implicitly assuming gamma = 1) results in dark, oversaturated images

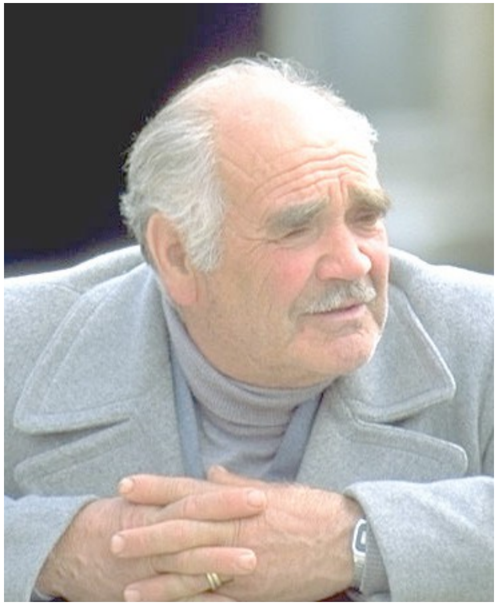
Gamma correction



corrected for
γ lower than
display



OK



corrected for
γ higher than
display

[Philip Greenspun]

