ISTA 352

Lecture 9

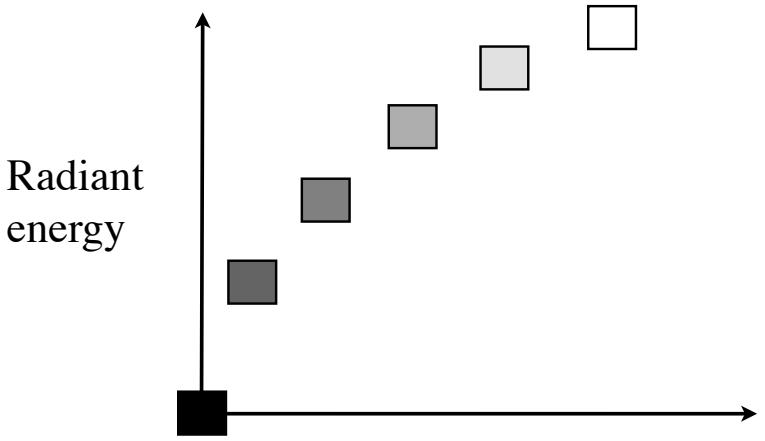
Digital representation of image data (II)

Administrivia

- HW two updated Saturday afternoon
- Material for the last required problem will be presented on Wednesday
- Linear algebra will be needed on Friday!

Summary from last time

- Images contain lots of data
  - Current memory/disk costs allows us to use 8 bits per channel, but space is not free, and reducing bandwidth is important
    - compression is still valuable
  - More that 8 bits per channel is usually not needed
  - Digitizing data implies discretization error
- Arrays will either contain the color (or gray) value for a pixel (direct color) or an index into a color map
- Three issues regarding number of bits and image fidelity
  - Color (or brightness) range
  - Resolution ("shades of gray")
  - Semantics of the numbers (be accurate where people will notice)



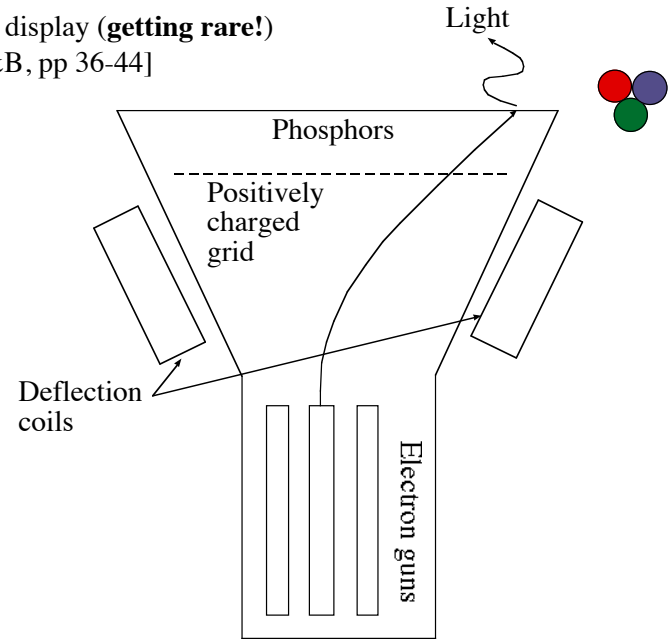
Good quiz prep exercise—explain this plot and its implications in your own words.

Gamma encoding

- Because of (a) the preceding graph and (b) discretization error, it makes sense to consider non-linear encoding of R,G, and B

- Historically, images were encoded with close to the correct non-linear function due to the "gamma" of cathodr ray oscilloscope (CRT) tubes in old style monitors (hence the name).

CRT display (**getting rare!**)
[H&B, pp 36-44]



Gamma encoding

- In the CRT, for a given input voltage, V, electrons hit the phosphors with energy E

$$E \propto V^{\gamma}, \quad \text{where } \gamma \text{ is } 5/2 \quad \text{(i.e., 2.5)}$$

- So, to drive the CRT so that the output energy is linear (recreating capture) you send it a voltage

$$V \propto E^{1/\gamma}$$

- The inverse of the CRT gamma is a pretty good answer to also deal with discretization error, and so it remains part of typical image encoding
  - Standard gamma is 2.2

Gamma encoding

- The non-linear encoding means that linear displays (now common) need to to the mapping from gamma encoded to linear
  - One way to think about it is that they have to emulate CRT monitor
  - Gamma is also becoming an image tone correction "knob" that either fixes an incorrect value, or simply makes some images look better.

- How can your mac robustly emulate a gamma of 2.2 for your monitor?
  - The OS has no idea what you have hooked up to it!
  - But it can make you turn knobs to make an image that should be linear to be linear
    - System Preferences --> Displays --> Color --> Calibrate (may need to select "expert")

Telling when an image is linear

Max OS X gamma tool demo

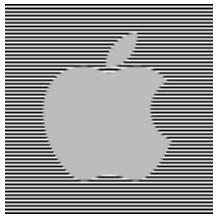
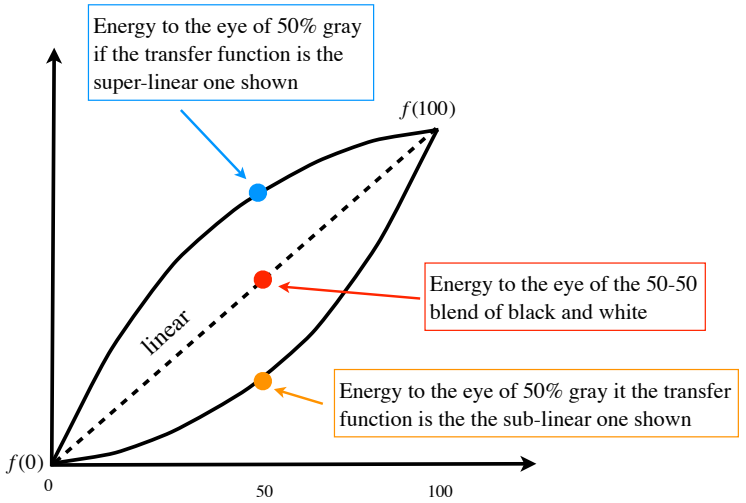


Figure for mac OS X gamma tool demo



Compression

- Lossless compression translates the data into a shorter form that contains all the information
  - The original can be reconstructed (uncompressed)
  - Common general purpose compression methods are lossless (e.g., gzip)
  - For data or computer code this is critical
- If image data loses a single low order bit, no one would notice
  - Image compression can be lossy
    - Common modern example is the jpeg format
  - Often the actual value of R,G, or B is partly noise
    - Using a close value may be just fine

Compression (II)

- Opportunities to compress images
  - Big changes (edges) are rare
    - This can be exploited in noise free (graphics) images using lossless runlength encoding
    - Example, storing a black and white checkerboard pattern
  - Many variations are ignored by the brain
    - May be able to save space by reducing fidelity in the right places
  - Color is perceived at lower resolution than intensity
    - Reducing the accuracy in color can save additional bandwidth

Analog Television (*)

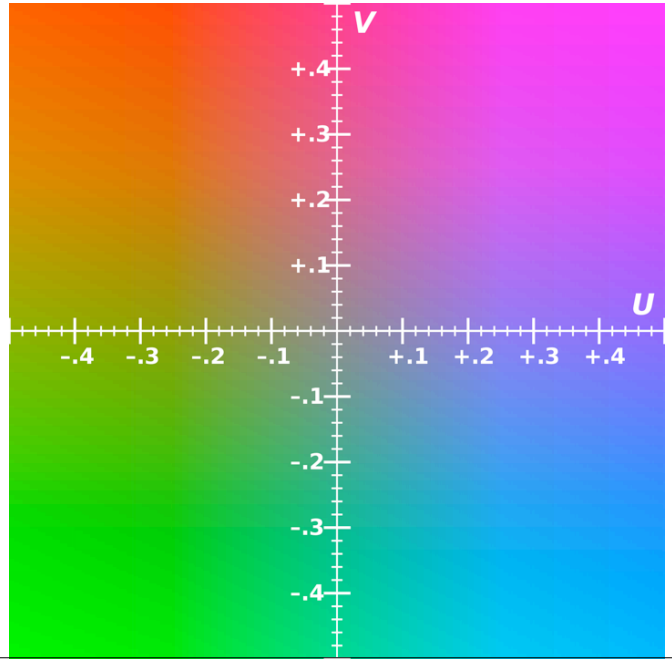
- Ideas and methods for both BW and color TV have been around since late 1800
- Broadcast TV for the masses needed cathode ray display devices, broadcast technology, and standards
- In the US the first BW transmission standard came in 1941
- Color followed in 1950

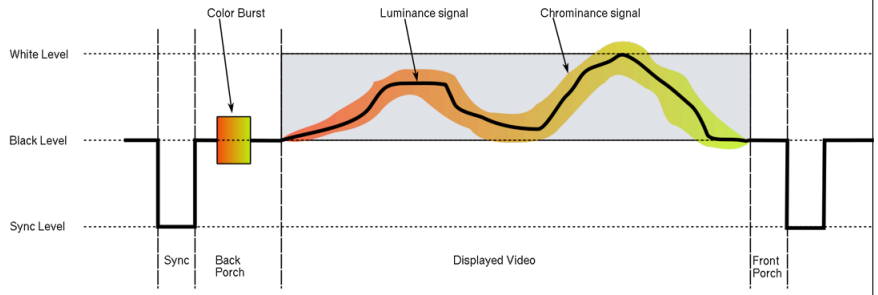
(*) Analog representation uses continuous signals (amplitude and/or frequency) instead of discretizing the values into fixed categories (digital).

Analog Television (II)

- Interesting challenges for the standards committees
  - BW was first and used most of the available signal
  - The color signal had to be backwards/forwards compatible
  - Existing BW sets needed to show color programming in BW
  - Color sets needed to show BW programming
- Our demos last time suggest that high fidelity in the color signal is not needed (the problem can be solved)
- Color signal encoding (YUV)
  - Luma (Y') encodes the brightness at high bandwidth
  - U=B'-Y'
  - V=R'-Y'

Here the primes (') mean gamma encoded.
There are additional scaling factors omitted for simplicity.





JPEG compression

- Do not confuse file format with underlying codec
  - Formats tell you where the bits go, codecs define what they mean
  - Jpeg files use JPEG coded (as do some others)
- Basic ideas
  - Color fidelity need to be the same as brightness
  - High frequency (fast changes) are often noise, and not so noticeable
  - Once lossy reduction has been done, do lossless compression (e.g., runlength) to save additional bits
- Downsides
  - Not a good method for when high frequency information is important (e.g., crisp edges in drawings and text)
    - Drawings and text are better represented in vector formats

JPEG compression (details)

- Convert to Y'CrCb (analogous to TV)
- Quantize color range by a factor of 2
- Divide images into 8x8 blocks, apply a discrete cosine transformation, and quantize higher frequency components into smaller ranges
- Apply lossless compression to the result

Discrete cosine transformation

- Consider the 8x8 black and white block as a linear combination of building blocks
  - This is the same as a "change of basis"
  - There are many way to do this
  - Simplest way is much like writing a 2D vector in terms of i-hat and j-hat.

Discrete cosine transformation

- Consider the 8x8 black and white block as a linear combination of building blocks
  - This is the same as a "change of basis"
  - There are many way to do this
  - Simplest way is much like writing a 2D vector in terms of i-hat and j-hat.
  - Illustrated using 2x2 blocks

Example, if $f(i,j) = \begin{vmatrix} 2 & 3 \\ 4 & 5 \end{vmatrix}$

$$f(i,j) = 2 \cdot \begin{vmatrix} 1 & 0 \\ 0 & 0 \end{vmatrix} + 3 \cdot \begin{vmatrix} 0 & 1 \\ 0 & 0 \end{vmatrix} + 4 \cdot \begin{vmatrix} 0 & 0 \\ 1 & 0 \end{vmatrix} + 5 \cdot \begin{vmatrix} 0 & 0 \\ 0 & 1 \end{vmatrix}$$

Refer to the basis function as box functions

Discrete cosine transformation

- In the cosine transformation we rewrite the image in terms of a different basis of 8x8 images so that some are less important
- Then we reduce the number of bits in the less important ones

Metaphorically, say

$$f(i,j) = 2 \bullet I_1 + 3 \bullet I_2 + 4 \bullet I_3 + 5 \bullet I_4$$

where each successive I is less important, then we can store higher cofficients with few bits than the earlier ones.

For example, for the last one we could use one bit for whether it is closer to 128 than -128

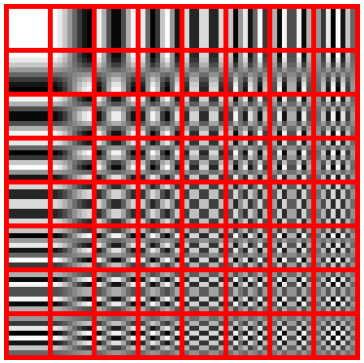
Discrete cosine transformation

- In the cosine transformation we rewrite the image in terms of a different basis of 8x8 images so that some are less important
- What do the basis images look like?

Discrete cosine transformation

- In the cosine transformation we rewrite the image in terms of a different basis of 8x8 images so that some are less important
- What do the basis images look like?

- Intuitively the most important thing about a block is its average value
- The next most important thing is whether it is brighter/darker on average to the left/right and up/down
- These are much more informative than the first three elements of the "box" basis



JPEG compression from the top

- Convert to Y'CrCb (analogous to TV)
- Quantize color range by a factor of two
- Divide images into 8x8 blocks, apply a discrete cosine transformation, and quantize higher frequency components into smaller ranges
- Apply lossless compression to the result