

Graphics = Light = Color = Memory

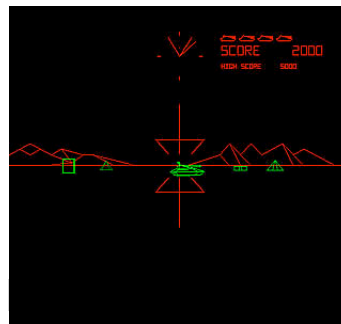
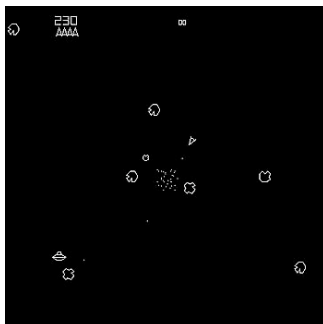
- Light-emitting media
 - CRT: cathode ray tube — phosphors excited by electrons
 - LCD: liquid crystal display — liquid crystals in a grid; current controls polarization and thus controls what colors can be seen
 - Projected LCD: LCD whose image is magnified before it reaches the viewer
 - Plasma: noble gas in between two glass sheets, also affiliated with an electrode grid; current excites the gas to emit light

 - Light-emitting media use an *additive* approach to color: add a color by adding light in that color (i.e. white = light in all colors)
- Light-reflecting media (print)
 - Ink-jet: fine spray of pigment onto display medium (most of the time, paper)
 - Thermal transfer: heat changes pigment on special paper
 - Laser: laser beam “etches” image on a drum coated with toner

 - Light-reflecting media use a *subtractive* approach to color: pigments absorb unwanted colors until only the desired color reflects back (i.e. white = no pigment)

CRT-Based Displays

- Once upon a time: vector displays
 - Based on oscilloscope technology
 - Line-based: electron gun traces lines directly from point A to point B
 - Made a lot of sense back when memory was expensive...
 - ...but what does memory have to do with graphics anyway?



CRT-Based Displays and Beyond

- Raster displays
 - Graphics display is a two-dimensional *raster* (array) of individual picture elements (*pixels*) in memory
 - a.k.a. frame buffer, graphics memory, VRAM
 - Display devices just transfer this 2D grid onto their specific type of display
 - CRT: electron gun scans the entire screen horizontally and vertically, exciting the appropriate phosphor that corresponds to its location in memory
 - Causes flicker
 - LCD/projected LCD: grid of crystals maps to a memory location
 - Plasma: ditto, but this time the pixels correspond to cells of gas
 - Note how LCD and plasma displays are inherently raster-oriented
- So, if pixels are memory, what do they hold?

Mapping Memory to Pixels

- Pixels essentially display color; the way the color is represented determines the “size” of the pixel in memory
- Linear memory is mapped to two dimensions: requires a width and height
 - mapping scheme can be linear or planar
- Display hardware does not always have square pixels: sometimes need to track *pixel ratio*
- Display hardware may not exactly match the frame buffer’s width and height, resulting in a *native resolution*

red	black	green	puce
red	black	blue	green
black	red	lime	tan
cyan	white	white	black

- 16 pixels in 4 rows and 4 columns
- How big is the buffer in bytes?
What do you need to know to answer this question?

Pixels and Color

- The *three-color model* is used to quantify color in computer graphics: colors are a triple of (*red*, *green*, and *blue*)
 - Based on the additive color approach, where every color can be represented by some combination of red, green, and blue
 - These values are a continuum of “none” to “full blast,” or representable as ranging from 0.0 to 1.0; 0 to 255 for 1-byte-per-color, and so on
 - Studies have shown that the human eye can generally perceive no more than 256 levels of a specific hue
- Quick formulas:
 - $R + G = \text{yellow}$
 - $G + B = \text{cyan}$ ← note these are the primary colors we learned way back when; they only said they were “red,” “yellow,” and “blue”
 - $R + B = \text{magenta}$
 - $R + G + B = \text{white}$

Pixels, Color, and Memory

- So if pixels map to some (R, G, B) tuple, how is this tuple stored in memory?
 - direct representation: a pixel in memory is literally the tuple
 - ex. If we have 1 byte per color, then we need 3 bytes per pixel
 - ex. If we aren’t doing colors (i.e. a pixel can only be black or white), then we need 1 bit per pixel
 - indexed representation: a pixel in memory is an index to a *color lookup table* (a.k.a. LUT or *palette*)
 - This leads to the phrase “A simultaneous i out of n possible colors.”
 - i determines the amount of memory occupied by a pixel
 - n determines the amount of memory used to represent a color
- Given this information, you should now generally be able to infer resolutions from catch phrases like “5.0 megapixels” or “128M of graphics memory”