Images

What is an image?

- Photographic print?
- Photographic negative?
- Display image?
- In-memory bits?

What is an image?

- Image: 2D distribution of colors.

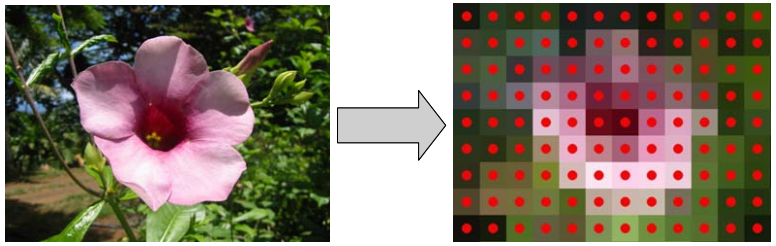
$$I : \Re^2 \rightarrow \ldots$$

What is an image?

- Digital Image: 2D regular grid of pixels
  - needed for computer manipulation

What is a pixel?

- Pixel: sample from a continuous function
  - Not a little square.

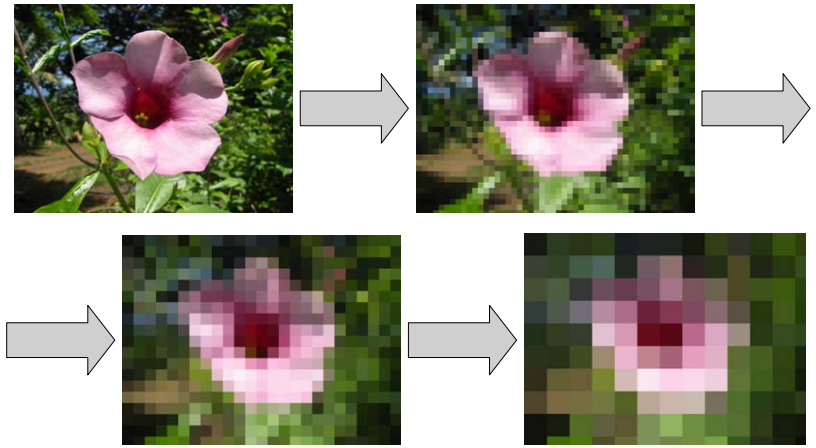


Representing images

- Sampling images introduces artifacts
  - Image Resolution
  - Color Resolution
  - Color Depth

Representing images – Image Resolution

- How many pixels?



Representing images – Color Resolution

- Color images: 3 color samples (red/green/blue)
  - Humans are trichromatic
  - match any color with mixtures of 3 primary colors
- Grayscale images: 1 sample
  - BW photography, various auxiliary maps
- Bitmap images: 1 boolean sample

Representing images – Color Depth

- How many color gradations?

Representing images – Data Structures

- Images are matrices of color arrays

```
dataType image[xres][yres]
```

- `dataType` defines type of image and color depth
- `xres` x `yres` is the image resolution

Representing images – Data types

- Bitmap: boolean per pixel $I : \Re^2 \rightarrow \{0,1\}$
  - range: black/white value only
- Grayscale: integer per pixel $I : \Re^2 \rightarrow [0,1]$
  - range: shades of gray
  - precision: 8bpp (sometimes higher)
- Color: integer[3] per pixel $I : \Re^2 \rightarrow [0,1]^3$
  - range: most reproducible colors
  - precision: 24bpp (sometimes higher)
- High-Dynamic Range: float[3] $I : \Re^2 \rightarrow \Re_+^3$
  - range: unconstrained
  - represents real lighting, not color

Dynamic Range

- Images store relative values in [0,1]
  - (0,0,0): black on display device
  - (1,1,1): white on display device
  - values relative to display device
- Light is measured in absolute value in [0,inf]
  - (0,0,0): black
  - (?,?,?): white?
  - Sun is brighter than a white wall
- High-Dynamic Range images
  - Store all values in [0,inf]
  - "Clamp" to a particular white point for display

HDR example

-8 stops exposure +8 stops

[image: Paul Debevec]

Color vs. Light … Confused?

- Color theory is very complicated!
  - Will cover some at the end of the course
- For now, think about color as RGB in [0,inf]
  - clamped to [0,1] when displayed or printed

Storage Requirements

- 1024x1024 in-memory image
  - bitmap: 128KB
  - grayscale 8bpp: 1MB
  - color 24bpp: 3MB
  - floating-point HDR color: 12MB
- Often compressed on disk
  - lossless compression (e.g. PNG)
    - data is redundant → fit in smaller space
  - lossy compression (e.g. JPG)
    - loose a bit of "non-that-visible" information

Image generation

- By acquisition
  - Scanner
  - Digital cameras
- By manipulation
  - Image operations
  - Painting/Editing
- By synthesis
  - Graphic art
  - Realistic synthesis
- Often a mixture of all

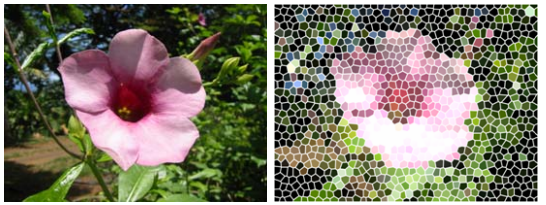
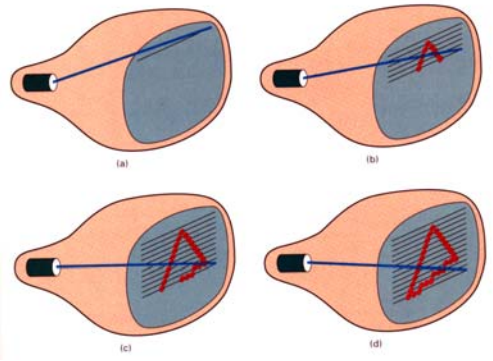


Image reproduction

- Display
  - cathode ray tube (CRT)
  - liquid crystal display (LCD)
  - plasma display
- Print
  - laser printer
  - inkjet printer

Cathode Ray Tube

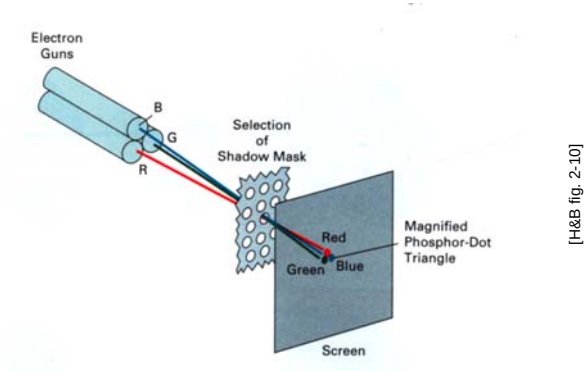
- Electron beam scans a screen
- Intensities by modulation of beam energy



[H&B fig. 2-7]

Cathode Ray Tube

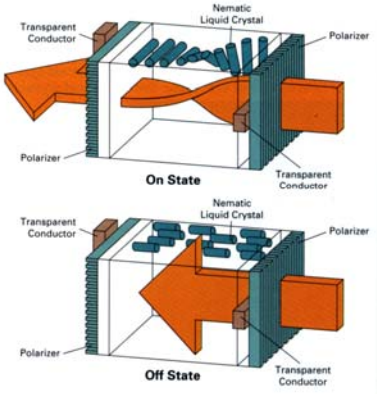
- Multiple beams for color



[H&B fig. 2-10]

Liquid Crystal Display

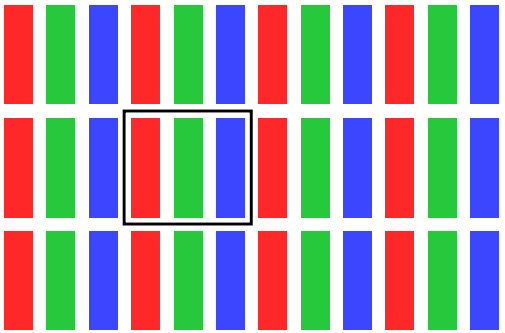
- Each pixel has a switch that can block light
- Intensities by how much light gets blocked



[H&B fig. 2-16]

Liquid Crystal Display

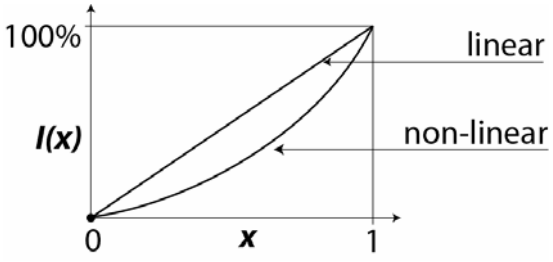
- Multiple pixels for color



[Marschner 2003]

Image units

- Display transfer function
  - display output intensity $I(x)$ for pixel value x
- Non-linear for old CRT display
  - approximately $I(x) = b + w \cdot x^{\gamma}$

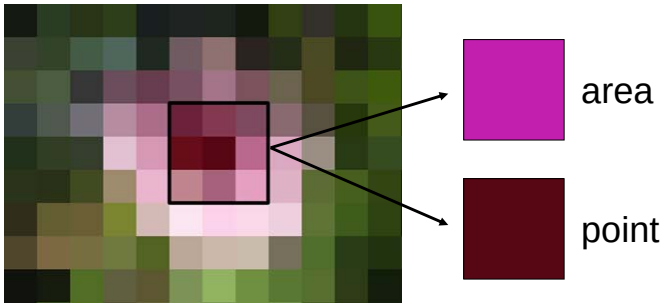


Non-linear (gamma) correction

- Old motivation – linearize display
  - CRTs are non-linear, correct them to obtain I(x)=ax
  - on most monitors, the OS performs this for us
- Current motivation – image perception
  - human visual system perceives intensities non-linearly
  - i.e. for a given intensity difference, darker intensity are perceived as larger
  - correct intensities to make them perceptually equal
  - can use a simple power law: $x' = x^{\gamma}$

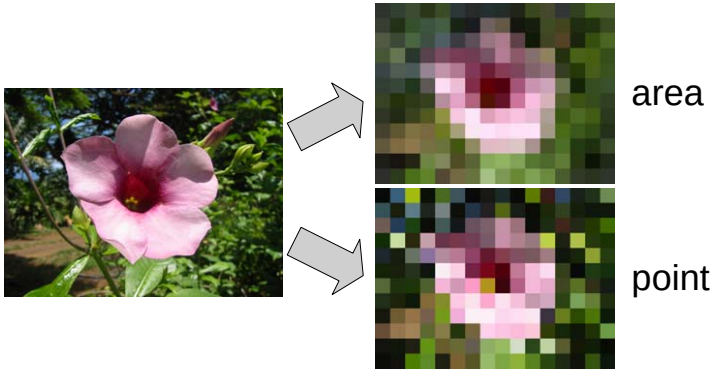
What is a pixel, really?

- Sample of Color *around* a point ("area" sample)
  - Not the color at the point (point sample)
  - Intuitive definition. Will discuss later in the course.



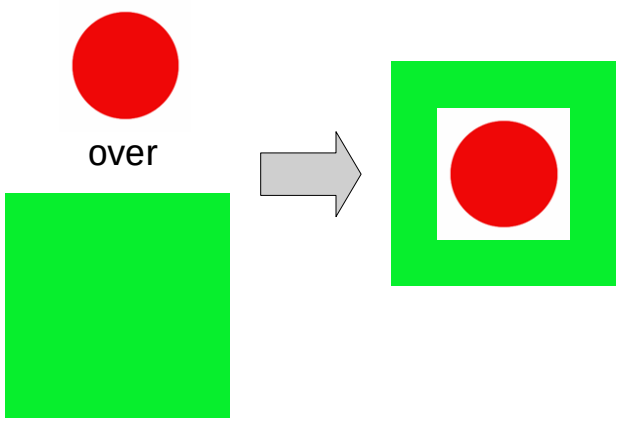
What is a pixel, really?

- Color *around* a point
  - Not the color at the point
  - Intuitive definition. Will discuss later in the course.



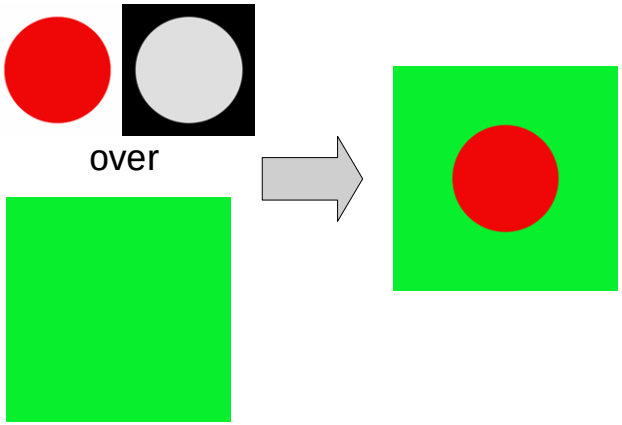
Compositing

- Combine two images by overlaying



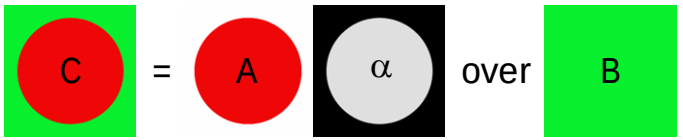
Compositing

- Encode transparency: *alpha* channel



Compositing

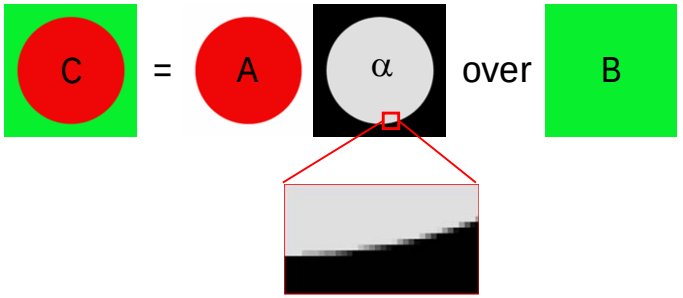
$$c_A = \begin{cases} c_A & \alpha_A \text{ is opaque} \\ c_B & \text{otherwise} \end{cases}$$



Compositing Two Layers

- Pixel on the edges are partially transparent
  - i.e. α encodes the degree of "transparency"

$C = A \textbf{ over } B$ $\quad c_C = \alpha_A \cdot c_A + (1-\alpha_A) \cdot c_B$



Compositing Two Layers

$$c_C = \alpha_A \cdot c_A + (1-\alpha_A) \cdot c_B$$ color composite

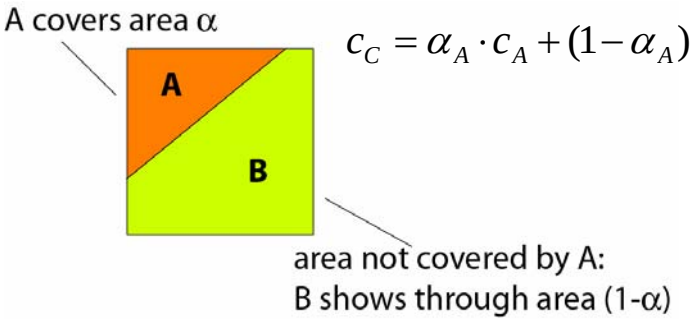
$$\Downarrow$$

$$r_C = \alpha_A \cdot r_A + (1-\alpha_A) \cdot r_B$$
$$g_C = \alpha_A \cdot g_A + (1-\alpha_A) \cdot g_B$$
$$b_C = \alpha_A \cdot b_A + (1-\alpha_A) \cdot b_B$$

composite each channel

Compositing: Definition of Coverage

- α is the fractional area covered by a fragment



$c_C = \alpha_A \cdot c_A + (1-\alpha_A) \cdot c_B$

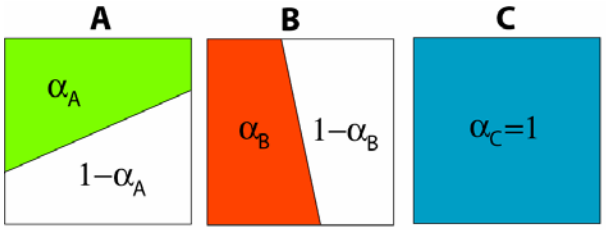
Compositing - Premultiplied colors

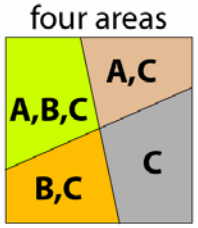
- Given the compositing equation
  $c_C = \alpha_A \cdot c_A + (1-\alpha_A) \cdot c_B$
- if we multiply the coverage by the colors
  $c' = \alpha \cdot c$
- we can rewrite the compositing result as
  $c'_C = c'_A + (1-\alpha_A) \cdot c'_B$
  - since coverage of B is 1, i.e. fully opaque
  - slightly faster, but much cleaner
    - let's see why.

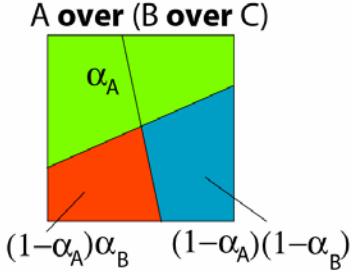
Compositing Multiple Layers

- Need to composite multiple layers
  - Special effects, Photoshop …

Compositing Multiple Layers



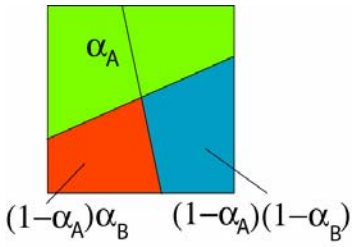




Compositing Multiple Layers

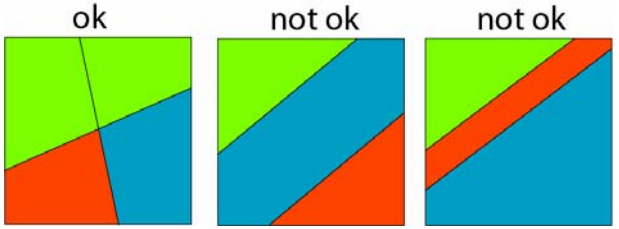
$$D = A \textbf{ over } (B \textbf{ over } C)$$

$$c_D = \alpha_A c_A + (1-\alpha_A)[\alpha_B c_B + (1-\alpha_B)\alpha_C c_C]$$
$$c'_D = c'_A + (1-\alpha_A)[c'_B + (1-\alpha_B)c'_C]$$



Compositing Multiple Layers

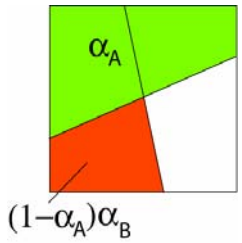
- Working assumption: independent covered areas
  - sometimes not true
  - but it works well enough and it is fast
  - so we use it all the times



Compositing Multiple Layers - Associativity

- A **over** $(B$ **over** $C) = (A$ **over** $B)$ **over** C
- need to define $\alpha_{A \textbf{ over } B}$

$$\alpha_{A \textbf{ over } B} = \alpha_A + (1-\alpha_A)\alpha_B$$



Compositing Multiple Layers - Associativity

$$D = (A \textbf{ over } B) \textbf{ over } C$$

$$c'_D = c'_{A \textbf{ over } B} + (1-\alpha_{A \textbf{ over } B})c'_C =$$
$$= c'_A + (1-\alpha_A)c'_B + [1-\alpha_A-(1-\alpha_A)\alpha_B]c'_C$$

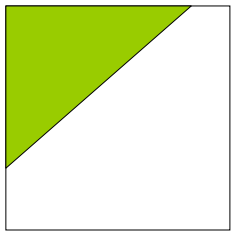
this proves that

$$A \textbf{ over } (B \textbf{ over } C) = (A \textbf{ over } B) \textbf{ over } C$$

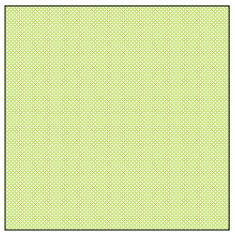
Coverage vs. Transparency

- opaque objects → compose with coverage
- transparent objects → blend with transparency
  - example: thin sheets of colored glass
  - same equations that compositing uses

opaque objects



transparent objects



Other Image Manipulations

- Convolution and Resampling
  - Resizing, Blurring, Artistic filters
- Painting
- Object removal

- Might cover later in the class