**Image Manipulation**

* Per-Pixel Manipulation
  + Individual pixels do not influence neighboring pixels
  + Possible modifications include shifts in:
    - Color
    - Brightness
    - Opacity
* Grayscale
  + RGB channels of pixel have the same value
  + Content of image expressed through color value rather than hue or sat
  + To find grayscale, set colors of pixels to (R + G + B) / 3.0
    - Average of their colors
* High Contrast
  + Increase or decrease value of RGB channels based on pixel brightness
  + Changes in value across image further emphasized
  + Threshold: kind of the amount of contrast
    - For all x in screen, if bright(x) > t, then adjust accordingly.
    - For all x in screen, if bright(x) <= t, then adjust accordingly.
* Image Kernels
  + Aka “Convolution Matrices” or “Masks”
  + Matrix used to convolve kernel values with image values
    - Square and small
    - The larger the matrix, the more data is lost.
  + Allows for “area” effects such as blur, sharpening and edge-detection
* Convolution
  + Matrix Convolution
    - 1. Multiplication of corresponding cells
    - 2. Summation of these values
* Intermediate Buffer
  + Array of pixels that matches the size of the image
  + Not displayed for users
  + Provides “safe” location for storing image data
  + Allows program to preserve original image data if necessary
  + Buffering is also a common trick to increase speed of rendering
* Box Blur
  + Pixel value is based on average of its neighborhood:
  + 1/9 \* {{1, 1, 1}, {1, 1, 1}, {1, 1, 1}}
  + Or {{.11, .11, .11}, {.11, .11, .11}, {.11, .11, .11}}
* Gaussian Blur
  + Use of Gaussian function for convolution:
  + Low-pass filter that reduces high frequency features including noise
  + Weighted average better preserves features
* Edge Detection
  + Determines sharp discontinuities in value (edges)
  + Provides information about scene:
    - Depth
    - Illumination
    - Material
  + Important filter for computer vision/feature extraction
* Sobel Operator
  + Two 3x3 kernels that approximate horizontal and vertical derivatives (changes light intensity)
  + Horizontal and vertical convolutions performed independently
  + Gradient magnitude calculated from results
  + Vertical: G\_x = {{-1, 0, 1}, {-2, 0, 2}, {-1, 0, 1}}
  + Horizontal: G\_y = {{-1, -2, -1}, {0, 0, 0}, {1, 2, 1}}
* Edge Cases
  + How to deal with the edge pixels
    - When you can’t use a NxN
  + Multiple methods:
    - Boolean – if at edge, shrink your matrix
    - Leave edges untouched
    - Fill in with 0 or 255
    - Wrap across the other side of the image
    - Mirror missing pixels from the smaller matrix you do have