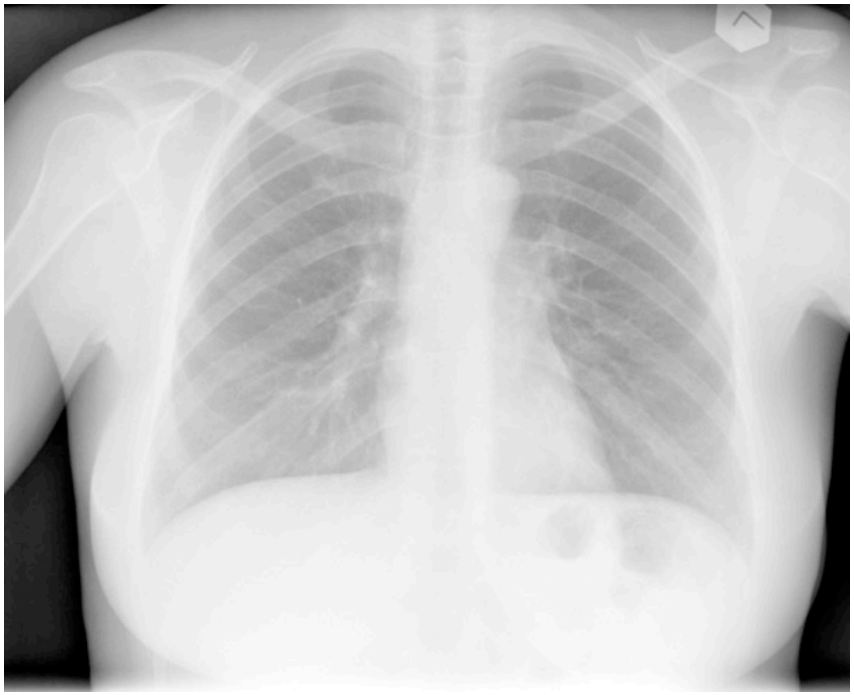
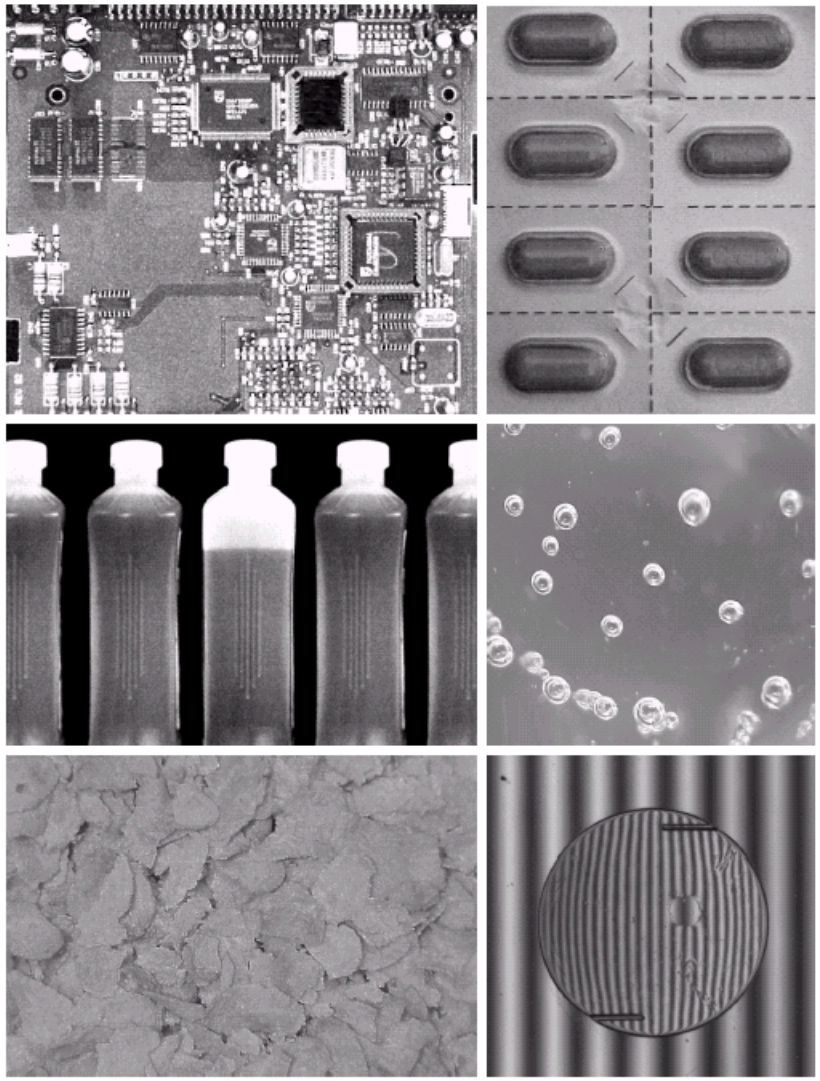
Image Processing

CS 110

Why Image Processing? Medical Images



Why Image Processing? Manufacturing



a b
c d
e f

FIGURE 1.14
Some examples of manufactured goods often checked using digital image processing. (a) A circuit board controller. (b) Packaged pills. (c) Bottles. (d) Bubbles in clear-plastic product. (e) Cereal. (f) Image of intraocular implant. (Fig. (f) courtesy of Mr. Pete Sites, Perceptics Corporation.)

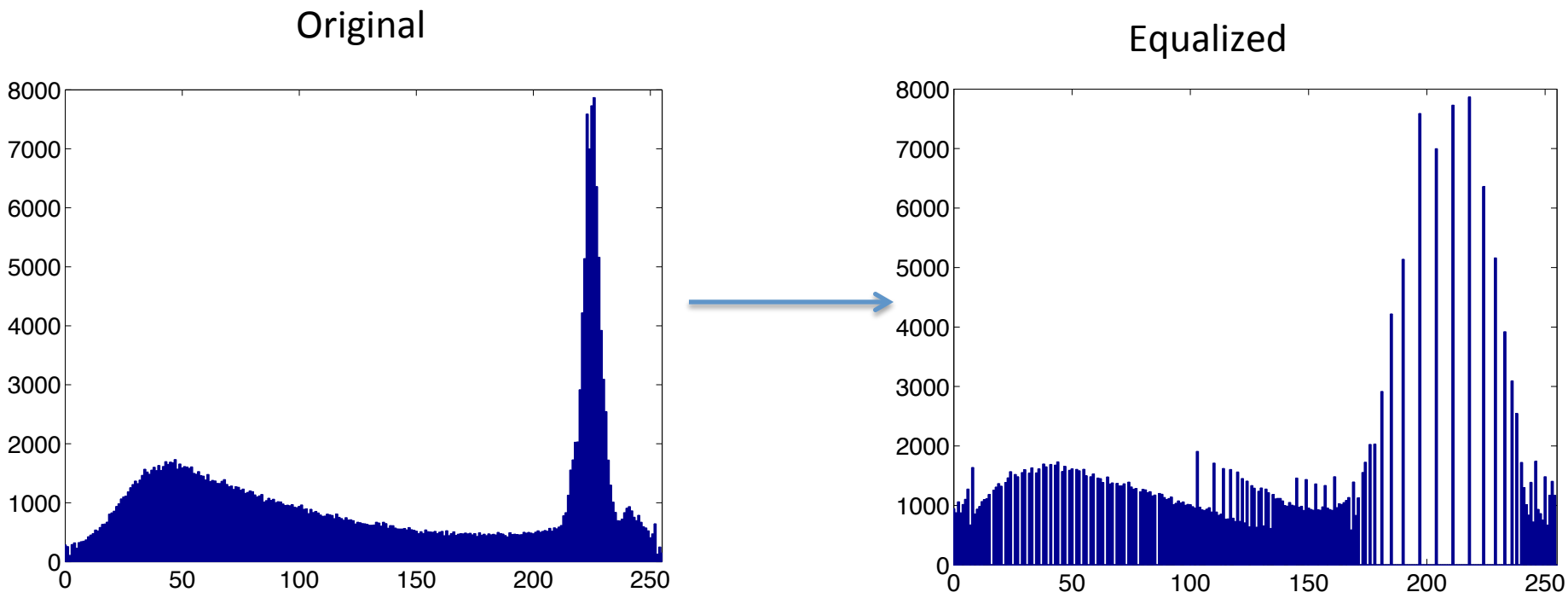
What can you do with Image Processing?

Inspect, Measure, and Count using Photos and Video
http://www.youtube.com/watch?v=KsTtNWVhpgI

Image Processing Software
http://www.youtube.com/watch?v=1WJp9mGnWSM

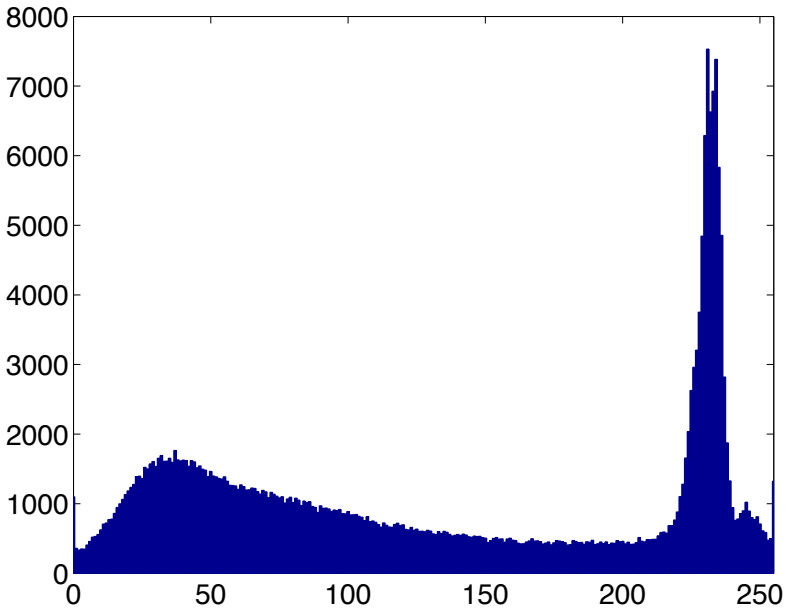
Image Enhancement: Histogram Equalization

Histogram Equalization (Red Channel)

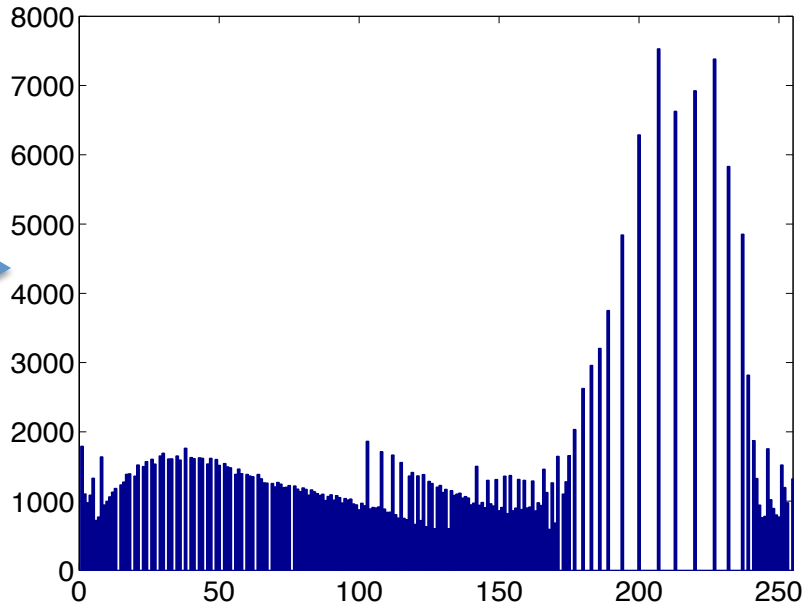


Histogram Equalization (Green Channel)

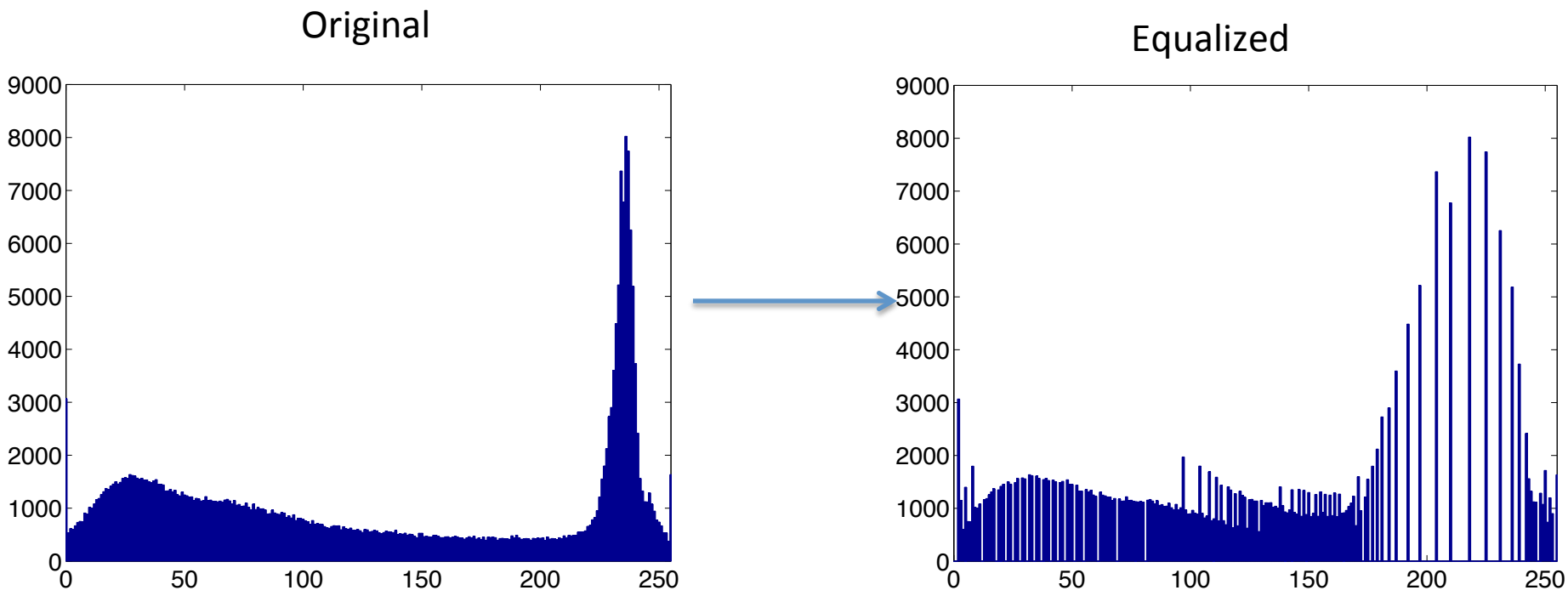
Original



Equalized



Histogram Equalization (Blue Channel)



How Do We Transform the Image

Basic idea: replace a color intensity value by it's percentile

A few sample red values and their transformed red values:

Original Red value	Percentile in Original Image	Transformed Value
52	20%	51
178	60%	153
228	90%	230

Histogram Equalization Step 1

```
void setup() {
  PImage img = loadImage("bmc3.jpg");
  size(2*img.width, img.height);
  image(img, 0, 0);

  float[] redHistogram = new float[256];
  float[] blueHistogram = new float[256];
  float[] greenHistogram = new float[256];

  // build color histograms for each channel
  img.loadPixels();
  for (int i = 0; i < img.pixels.length; i++) {
      redHistogram[(int)red(img.pixels[i])]++;
      greenHistogram[(int)green(img.pixels[i])]++;
      blueHistogram[(int)blue(img.pixels[i])]++;
  }
```

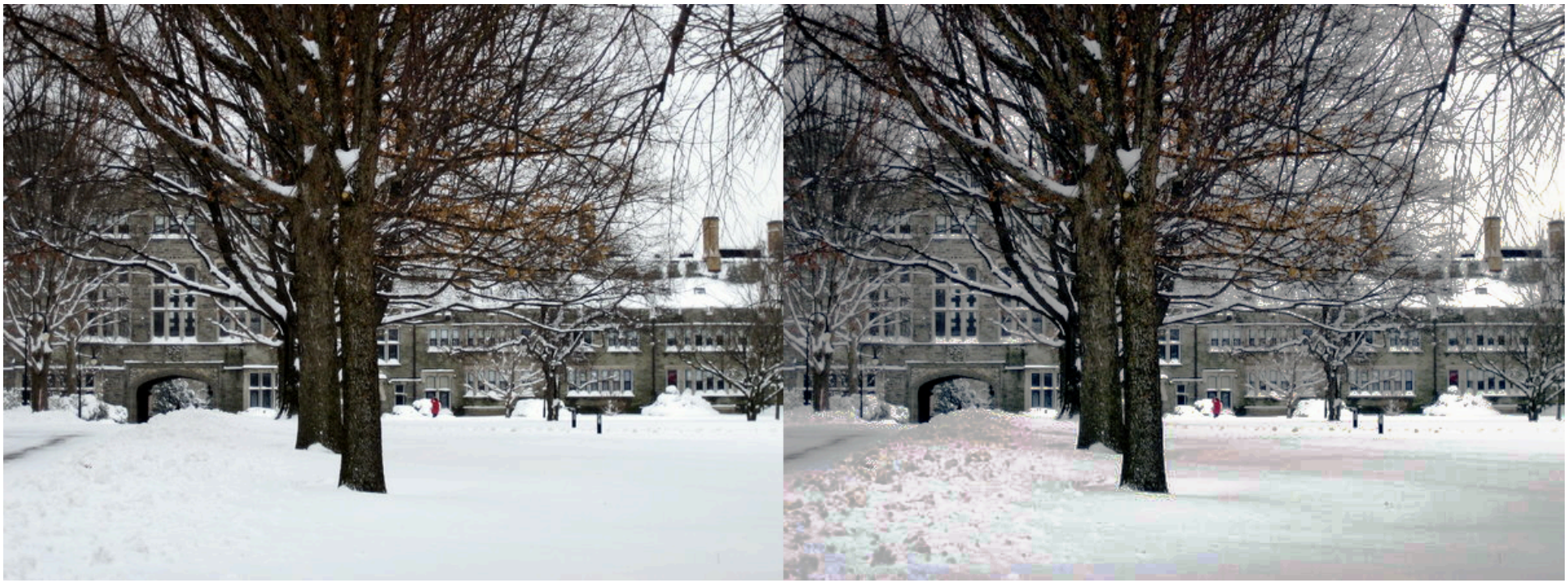
Histogram Equalization Step 2

```
// convert to quantile
float numPixels = img.width*img.height;
redHistogram[0] = redHistogram[0]/numPixels;
greenHistogram[0] = greenHistogram[0]/numPixels;
blueHistogram[0] = blueHistogram[0]/numPixels;

for (int i = 1; i < redHistogram.length; i++) {
  redHistogram[i] = redHistogram[i-1] + redHistogram[i]/numPixels;
  greenHistogram[i] = greenHistogram[i-1] + greenHistogram[i]/numPixels;
  blueHistogram[i] = blueHistogram[i-1] + blueHistogram[i]/numPixels;
}
```

Histogram Equalization Step 3

```
  // convert the values using the quantiles computed above
  for (int i = 0; i < img.height; i++) {
    for (int j = 0; j < img.width; j++) {
      int idx = j+i*img.width;
      img.pixels[idx] = color(255*redHistogram[(int)red(img.pixels[idx])],
                              255*greenHistogram[(int)green(img.pixels[idx])],
                              255*blueHistogram[(int)blue(img.pixels[idx])]);
    }
  }
  img.updatePixels();
  image(img, img.width, 0);
}
```

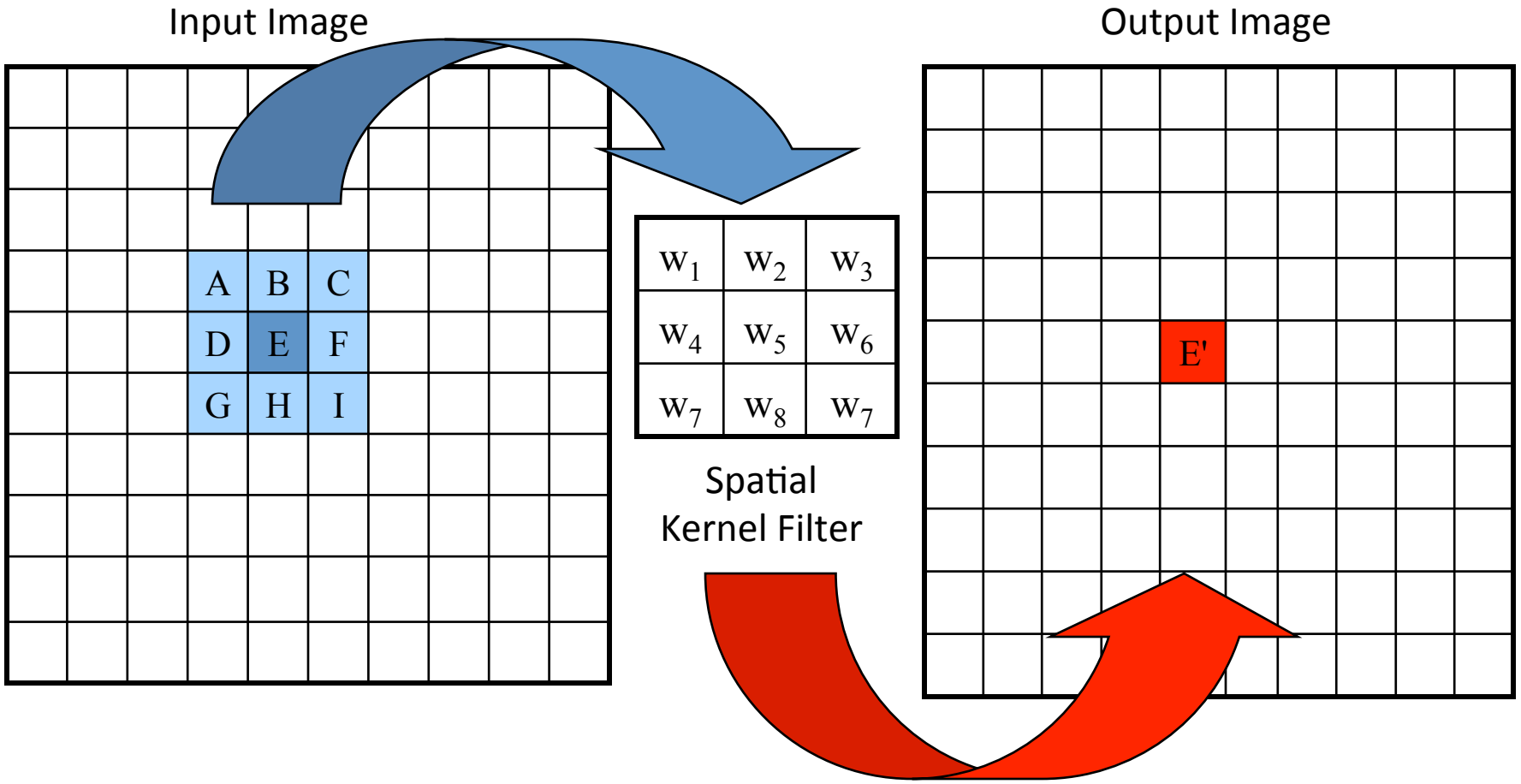
Results



Histogram Equalization in Human Vision?

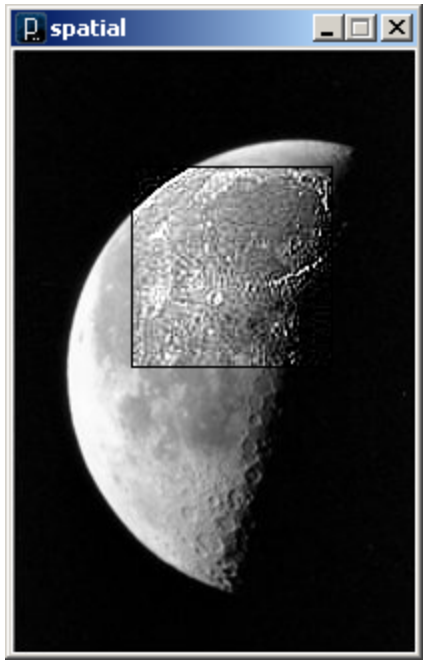
Two mechanisms to regulate the dynamic range of the eye: **Pupil dilation**, chemical changes in the retina (**Perkinje effect**)

Spatial Filtering



$$E' = w_1A+w_2B+w_3C+w_4D+w_5E+w_6F+w_7G+w_8H+w_7I$$

```
// Spatial Filtering

PImage img;
PImage filt;
int w = 100;
int msize = 3;

// Sharpen
float[][] matrix = {{ -1., -1., -1.},
                    { -1.,  9., -1.},
                    { -1., -1., -1.}};

// Laplacian Edge Detection
//float[][] matrix = {{  0.,  1.,  0. },
//                    {  1., -4.,  1. },
//                    {  0.,  1.,  0. }};

// Average
//float[][] matrix = {{ 1./9., 1./9., 1./9.},
//                    { 1./9., 1./9., 1./9.},
//                    { 1./9., 1./9., 1./9.}};

// Gaussian Blur
//float[][] matrix = {{  1./16.,  2./16.,  1./16. },
//                    {  2./16.,  4./16.,  2./16. },
//                    {  1./16.,  2./16.,  1./16. }};

void setup() {
  //img = loadImage("bmc3.jpg");
  img = loadImage("moon.jpg");
  size( img.width, img.height );
  filt = createImage(w, w, RGB);
}

void draw() {
  // Draw the image on the background
  image(img,0,0);

  // Get current filter rectangle location
  int xstart =
      constrain(mouseX-w/2,0,img.width);
  int ystart =
      constrain(mouseY-w/2,0,img.height);

  // Filter rectangle
  loadPixels();
  filt.loadPixels();

  for (int i=0; i<w; i++ ) {
    for (int j=0; j<w; j++) {
      int x = xstart + i;
      int y = ystart + j;
      color c =
          spatialFilter(x, y, matrix, msize, img);
      int loc = i+j*w;
      filt.pixels[loc] = c;
    }
  }

  filt.updatePixels();
  updatePixels();

  // Add rectangle around convolved region
  stroke(0);
  noFill();
  image(filt, xstart, ystart);
  rect(xstart, ystart, w, w);
}

// Perform spatial filtering on one pixel location
color spatialFilter(int x, int y, float[][] matrix,
                    int msize, PImage img) {
  float rtotal = 0.0;
  float gtotal = 0.0;
  float btotal = 0.0;
  int offset = msize/2;

  // Loop through filter matrix
  for (int i=0; i<msize; i++) {
    for (int j=0; j<msize; j++) {

      // What pixel are we testing
      int xloc = x+i-offset;
      int yloc = y+j-offset;
      int loc = xloc + img.width*yloc;

      // Make sure we haven't walked off
      // the edge of the pixel array
      loc = constrain(loc,0,img.pixels.length-1);

      // Calculate the filter
      rtotal += (red(img.pixels[loc]) * matrix[i][j]);
      gtotal += (green(img.pixels[loc]) * matrix[i][j]);
      btotal += (blue(img.pixels[loc]) * matrix[i][j]);
    }
  }
  // Make sure RGB is within range
  rtotal = constrain(rtotal,0,255);
  gtotal = constrain(gtotal,0,255);
  btotal = constrain(btotal,0,255);

  // return resulting color
  return color(rtotal, gtotal, btotal);
}
```



Sharpen

Edge Detection

Gaussian Blur

Image Processing in Processing

tint()	modulate individual color components
blend()	combine the pixels of two images in a given manner
filter()	apply an image processing algorithm to an image

```
// Posterize
PImage img;

void setup() {
  img = loadImage("andy-warhol2.jpg");
  size(img.width, img.height);
  image(img, 0, 0);
}

void draw() {}

void drawImg(float val)  {
  image(img, 0, 0);
  filter(POSTERIZE, val);
}

void mouseDragged() {
  float val = int(map(mouseY, 0, height, 2, 10));
  val = constrain(val, 2, 10);
  println(val);
  drawImg(val);
}
```

Let's Build Our Own Posterize

Basic Idea: reduce the number of possible values each color channel can take. (e.g. posterize level = 4)

Original Red Channel:

New Red Channel:

Original Green Channel:

New Green Channel:

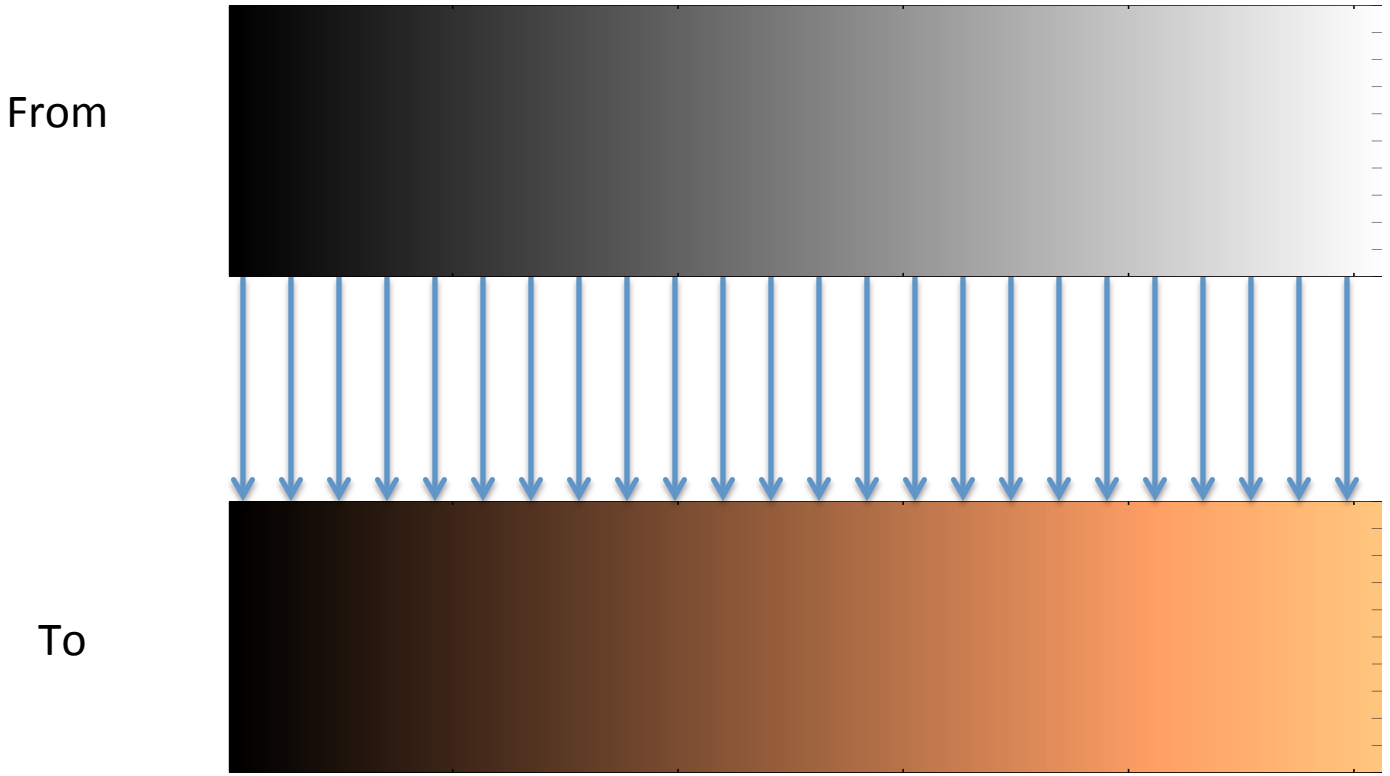
Original Blue Channel:

New Blue Channel:

Gradient Color Palettes

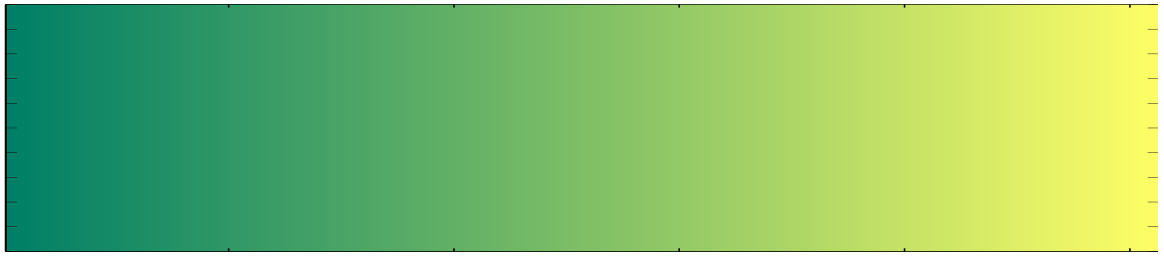
Defines a map from brightness value to a color:

For Example Copper color palette

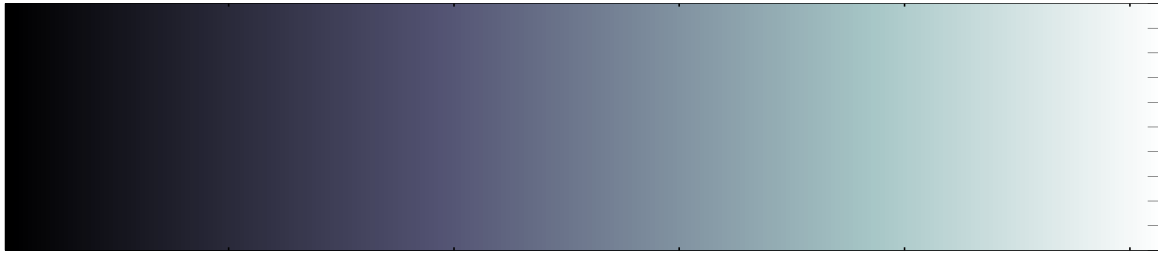


Other Examples

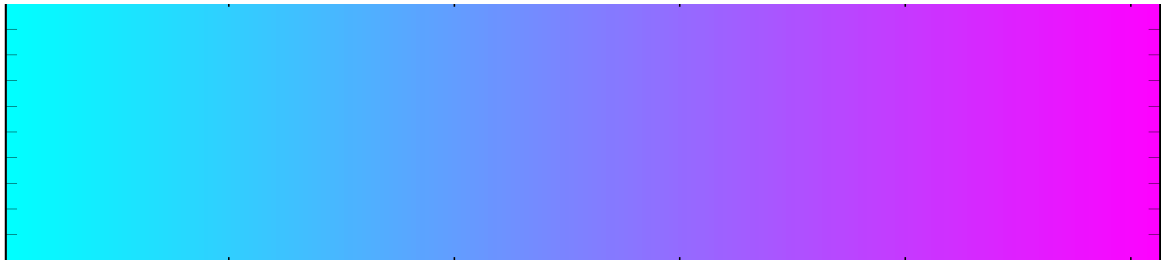
Summer



Bone



Cool



Sepia

High-level Idea

1. Convert Image to Grayscale (brightness)
2. Build color palette (maps from grayscale to RGB value)
3. Apply color palette to create the transformed image