Digital Image Processing Laboratory: Pointwise Operations and Gamma

December 11, 1998

Introduction

This laboratory will discuss various types of point operations on images. This generally refers to a remapping of pixels at one intensity to a different intensity. Point operations make up an important class of techniques for image enhancement.

In Section 1, we will introduce a graphical tool called a *histogram*, which shows the distribution of pixel intensities in an image. An enhancement technique called *histogram equalization* will also be presented here.

In Section 4, we will discuss an important parameter called *gamma* which describes the nonlinear behavior of a display device. You will determine the gamma of your computer monitor, and then apply a point operation known as *gamma correction* to compensate for this nonlinearity.

In some exercises you will be asked to display gray scale images in Matlab. If `X` is an image that takes on the values $[0, 1, \ldots, 255]$, then it may be displayed by using the following commands.

```
image(X+1);
axis('image');
graymap = [0:255; 0:255; 0:255]'/255;
colormap(graymap);
```

1 Histogram of an Image

The *histogram* of a digital image shows the distribution of its pixel intensities. Let X be a gray scale (achromatic) image. The a single pixel is denoted as X_s where $s \in S$ is a position in the lattice S. Generally, we will assume that X_s takes on the discrete values $0, \ldots, L-1$ where typically $L = 256$. The histogram of the image X is then given by

$$h(i) = \sum_{s \in S} \delta(X_s - i) \ .$$

So $h(i)$ computes the number of pixels that take on the value i. A typical histogram of an 8 bit image is shown in Figure 1.

Now we will display the histogram of two images.

Questions or comments concerning this laboratory should be directed to Prof. Charles A. Bouman, School of Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907; (765) 494-0340; bouman@ecn.purdue.edu

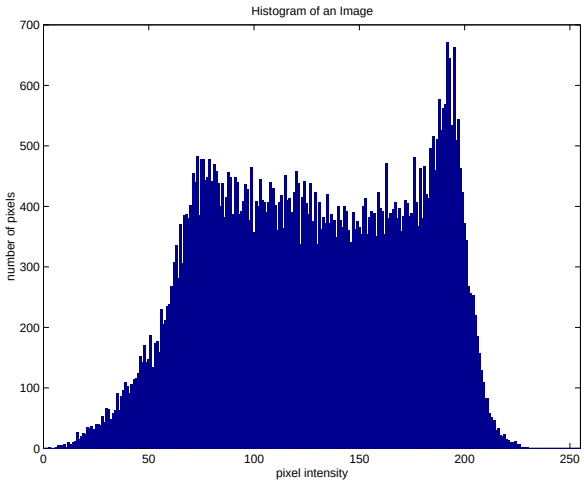

Figure 1: Histogram of an 8 bit image

1. Down load the two gray scale images *race.tif* and *kids.tif* from the lab home page.
2. Use the Matlab function *hist* to display the histograms of each image. **Note:** The *hist* command requires the image matrix be reshaped to a 1 dimensional vector to display this type of histogram. Use the command `y=reshape(x,1,M*N);` to reorder an $M \times N$ image into a $1 \times MN$ vector. Then use `hist(y,[0:255])` to create a histogram with bins centered at $0, 1, \ldots, 255$.
3. Print out each image and its labeled histogram.

Section 1 Report:
Hand in the two images and their labeled histograms.

2 Histogram Equalization

Histogram equalization is a common image enhancement technique. The objective is to transform the image so that the output histogram is uniform over the full range of gray values.

To approach this problem, think of an image as a collection of i.i.d. random variables, each with cummulative distribution and density of

$$\begin{aligned} F_x(x) &= Prob\{X \leq x\} \\ p_x(x) &= \frac{d}{dx} F_x(x) \ . \end{aligned}$$

Now consider forming the following random variable.

$$Y = F_x(X) \tag{1}$$

Here Y is the result of mapping the random variable X through its own cummulative distribution function. The cummulative distribution of Y can be easily computed.

$$\begin{aligned} F_y(y) &= Prob\{F_x(X) \leq y\} \\ &= Prob\{X \leq F_x^{-1}(y)\} \\ &= F_x(F_x^{-1}(y)) \\ &= \begin{array}{ll} 0 & \text{for } y < 0 \\ y & \text{for } 0 \leq y \leq 1 \\ 1 & \text{for } y > 1 \end{array} \end{aligned}$$

This shows that Y has a uniform distribution on the interval $[0, 1]$. Therefore, the histogram of an image can be equalized by mapping the pixels through their cummulative distribution function $F_x(x)$.

In order to implement the histogram equalization method, we must estimate the cummulative distribution function for the image. We can do this using the images histogram. Let $h(i)$ be the images histogram formed by computing the number of pixels at gray level i. Typically, the pixels take on the values $i = 0, \ldots, L-1$ where $L = 256$ is the number of discrete levels that a pixel can take on. The cummulative distribution function can then be approximated by

$$\hat{F}_x(i) = \frac{1}{h(L-1)} \sum_{j=0}^{j=i} h(j)$$

Here the normalization term assures that $F_x(L-1) = 1$. By applying the concept of (1), we equalize a pixel X_s in at the position $s \in S$ where S is the set of positions in the image.

$$Y_s = \hat{F}_x(X_s)$$

However, Y_s has a range from 0 to 1 and may not extend over the maximum number of gray levels. To correct these problems, we first compute the minimum and maximum values of Y_s.

$$\begin{aligned} Y_{max} &= \max_{s \in S} Y_s \\ Y_{min} &= \min s \in S Y_s \end{aligned}$$

And then we use these values to from Z_s, a renomalized version of Y_s.

$$Z_s = (L-1)\frac{\hat{F}_x(X_s) - Y_{min}}{Y_{max} - Y_{min}} \tag{2}$$

Histogram equalization is then the transformation from X_s to Z_s.

1. Write a Matlab function `Y=equalize(X)` that equalizes the histogram of the image X using equation (2).

2. Down load the image *kids.tif* from the lab home page.

3. Compute and plot the function $\hat{F}_x(i)$ for the image *kids.tif*.

4. Apply your *equalize* function to the *kids.tif* image. Print a hard copy of the equalized image.

5. Plot the histogram of the equalized image. Label it and print out a hard copy.

6. Compare the image and its histogram to the originals, which were generated in Section 1.

Section 2 Report:

1. Hand in the function *equalize.m*.

2. Hand in a labeled plot of $\hat{F}_x(i)$ for the image *kids.tif*.

3. Hand in a labeled plot of the of the equalized image's histogram.

4. Hand in the hard copy of the equalized image.

3 Contrast Stretching

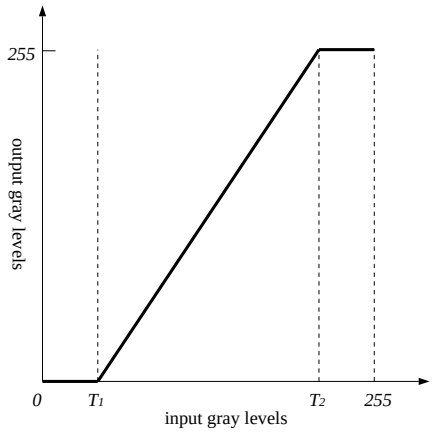

Figure 2: Contrast Stretching Transformation

Another useful image enhancement technique is *contrast stretching.* An example is depicted in Figure 2. The horizontal axis shows all possible intensities of the original image, and the vertical axis shows the intensities of the transformed image. This particular transformation maps the "darker" pixels in the range $[0, T_1]$ to a level of zero (black), and similarly maps the "lighter" pixels in $[T_2, 255]$ to white. Then the pixels in the range $[T_1, T_2]$ are "stretched out" to use the full scale of $[0, 255]$. This can have the effect of increasing the contrast in an image.

If a pixel transformation can be described by a one-to-one function, $y = f(x)$, then it can be shown that the input and output histograms are approximately related by the following:

$$h_{out}(y) \approx \left. \frac{h_{in}(x)}{|f'(x)|} \right|_{x=f^{-1}(y)} . \tag{3}$$

Since x and y need to be integers in (3), the evaluation of $x = f^{-1}(y)$ needs to be rounded to the nearest integer.

The pixel transformation shown in Figure 2 is not a one-to-one function. However, equation (3) still may be used to give insight into the effect of the transformation. Since the regions $[0, T_1]$ and $[T_2, 255]$ map to the single points 0 and 255, we might expect "spike" behavior at the points 0 and 255 in the output histogram. The region $[1, 254]$ of the output histogram will be directly related to the input histogram through equation (3).

First, notice from $x = f^{-1}(y)$ that the region $[1, 254]$ of the output is being mapped from the region $[T_1, T_2]$ of the input. Then notice that $f'(x)$ will be a constant scaling factor throughout the entire region of interest. Therefore, the output histogram should approximately be a "stretched" and rescaled version of the input histogram, with possible spikes at the endpoints.

1. Write a Matlab function that will perform the pixel transformation shown in Figure 2. The input and output images should be of type *uint8*. It should have the syntax
   `output = stretch(input, T1, T2) .`
2. Down load *kids.tif* from the lab home page and read it into Matlab. Display the image, and compute its histogram.
3. Now use your *stretch* function to increase the contrast of the image. Use the histogram to determine appropriate values for using $T1$ and $T2$. Display the new image and its histogram. Label both of them and print them out.
4. Compare the image and its histogram to the originals, which were generated in Section 1.

Section 3 Report:

1. Hand in your code for *stretch*.
2. Hand in the transformed image and its histogram.

4 Gamma (γ)

The light intensity generated by a physical device is usually a nonlinear function of the original signal. For example, a pixel that has a gray level of 200 will not be twice as bright as

a pixel with a level of 100. Almost all computer monitors have a *power law* response to their applied voltage. For a typical cathode ray tube (CRT), the brightness of the illuminated phosphors is approximately equal to the applied voltage raised to a power of 2.5. The numerical value of this exponent is known as the *gamma* (γ) of the CRT. Therefore the power law is expressed as

$$I = V^{\gamma} \tag{4}$$

where I is the pixel intensity and V is the voltage applied to the device.

If we relate equation (4) to the pixel values for an 8 bit image, we get the following relationship,

$$y = 255 \left(\frac{x}{255} \right)^{\gamma} \tag{5}$$

where x is the original pixel value, and y is the pixel intensity as it appears on the display. This relationship is illustrated in Figure 3.

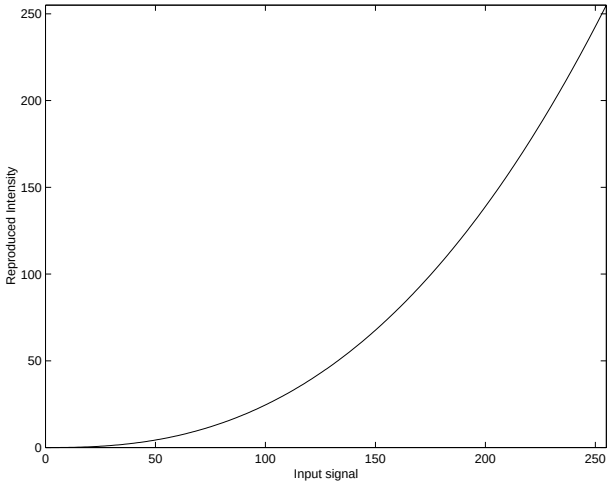

Figure 3: Nonlinear behavior of a display device having a γ of 2.2.

In order to achieve the correct reproduction of intensity, this nonlinearity must be compensated by a process known as *gamma correction*. Images that are not properly corrected usually appear too light or too dark. If the value of gamma is available, then the correction process consists of applying the inverse of equation (5). This is a straightforward pixel transformation, as we discussed in Section 3.

4.1 Setting the Black Level and Picture of Your Monitor

In a following section, you are going to determine the value of gamma for your computer monitor. In order to do this accurately, it is essential that you properly set monitor parameters called the *Black Level* and *Picture*, also respectively known as *brightness* and *contrast*. Standard symbols for these controls are shown in Figure 4.

The following is the proper procedure for setting the Black Level and Picture. See [1] for a detailed explanation of these parameters.

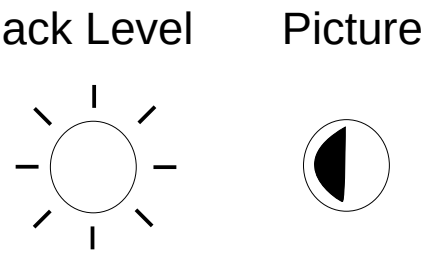

Figure 4: Video Monitor Controls

1. Set the Picture control to approximately half-way.
2. Adjust the Black Level just to the point where there is no background illumination of the phosphors in regions of the screen that should be black. In other words, the black regions should be as dark as possible, but any increase in Black Level will cause those phosphors to illuminate. This maximizes the contrast ratio of the monitor. Note: Once the Black Level is properly set, it should not need to be changed again.
3. Adjust the Picture to obtain a comfortable brightness.

4.2 Determining the Gamma of Your Computer Monitor

In this section, you will determine the gamma of your monitor by using a simple experiment that exploits properties of the human visual system. The methods works by producing visually comparing a checkerboard pattern to a fix gray level. The checkerboard pattern so produced by displaying a repeated pattern of black and white dots with the values shown in Fig. 5. When viewed from a distance, this checkboard pattern appears as a uniform gray level due to the low pass blur from the optics of the eye. Since the optics of the eye blur the intensity, the percieved intensity of the checkerboard is given by

$$I_c = (I_{255} + 0)/2$$

where I_{255} is the intensity produced by a full intensity dot of level 255. The intensity of this checkerboard is then compared to a fixed gray level produce by an input $0 \leq g \leq 255$. Assuming a standard gamma model for the behavior of the monitor, the gray level g produces a perceived intensity of

$$I_g = I_{255}(g/255)^{\gamma} \ .$$

We can then calculate the value of γ for the monitor by determining the gray level g which makes $I_g = I_c$.

In order to visually match the two gray values, you will produce a patch containing stripes of constant gray interlaced with stripes of checkerboard pattern. Such a patch is shown in Fig. fig:pattern. You will then adjust the gray level until the strip pattern disappears when viewed from a distance.

You will now determine the value of gamma for your video monitor.

1. Set the Black Point of your monitor as described in Section 4.1.

255	255	0	0
255	255	0	0
0	0	255	255
0	0	255	255

Figure 5: Fundamental Unit of Checkerboard Pattern

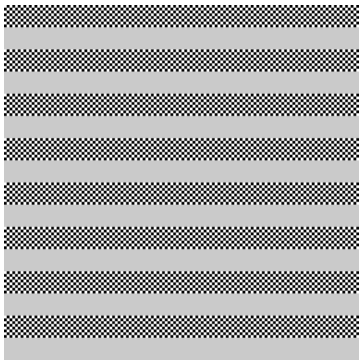

Figure 6: Array Pattern for Determining Gamma

2. Write a Matlab function that will display a 256×256 array containing horizontal stripes which alternate between a checkerboard pattern and a constant gray level. An example is shown in Figure 6.

   - Since the gray level strips must be adjusted to match the checkerboard, the gray level should be an input to your function.
   - Each stripe should be 16 pixels in height.
   - Each block of the checkerboard pattern should consist of four pixels, as shown in Figure 5.
   - Place the appropriate commands within the function to display the array.

3. Initially display the array using a gray level of 127. Then adjust the gray level until the stripes appear to have the same intensity.

4. Record the value of the matching gray level that produces the best intensity match between the stripes.

5. Print out the image corresponding the the matching gray level.

6. Derive an analytical expression for γ in terms of the matching gray level and calculate the value of γ for your monitor.

Do the regions appear to have the same intensity in the printed version?

Section 4.2 Report:

1. Hand in your print out of the image corresponding to the matching gray level.
2. Hand in a derivation of the expression which relates the matching gray level to the value of γ.
3. Hand in the values of the measured gray level and the measured γ.

4.3 Gamma Correction

As mentioned in Section 4, a process known as *gamma correction* is used to compensate for the nonlinear behavior of a display device. Most often images are already encoded in gamma corrected form, and will appear fine when displayed on most video monitors. However, if an image is stored with a linear scaling it becomes necessary to correct the image. If the value of gamma for your monitor is known, then the correction process consists of applying the inverse of equation (5).

1. Determine the value of gamma for the monitor you are currently using by following the procedure of Section 4.2.
2. Down load the image *linear.tif* from the lab home page. This image is encoded on a linear scale. Use *xv* to display the image, and print a hard copy.
3. Gamma correct *linear.tif* for your monitor.
4. Use *xv* to display the gamma corrected form of *linear.tif* and print a hard copy.

Section 4.3 Report:

1. Hand in the original and corrected images. Label them and indicate the value of gamma that was used to correct the image.
2. Hand in the formula you used to transform the original image.

Sometimes an image is encoded in gamma corrected format, but not for a gamma corresponding to your monitor. It then becomes necessary to change the gamma correction for the image.

1. Down load the image *gamma15.tif* from the lab home page. This image is already gamma corrected for a $\gamma = 1.5$.
2. Derive the procedure for transforming *gamma15.tif* so that it will be displayed properly on your monitor.

3. Use *xv* to display the image, and print a hard copy.

Section 4.3 Report:

1. Hand in the corrected image. Be sure that it is labeled.
2. Document the procedure you used to change the gamma correction of the original image.

References

[1] C. A. Poynton, "Black Level" and "Picture", 1997.