Connor McCurley

EEE 6512 **Homework 1 Due September 15, 2018** Fall 2018

Part I Textbook Questions

2.5

Since $\lambda = \frac{c}{\nu}$, then $\lambda = \frac{c}{60Hz} = \frac{2.998 \cdot 10^8 m/s}{60Hz} = 4.997 \cdot 10^8 m$ Therefore,

$$\boxed{\lambda = 4.997 \cdot 10^5 km}$$

2.10

I can imaging solving the problem presented in 1 of 2 ways (although I'm sure there are many more):

1. You could take an RGB image of the car over a spot with pure color (i.e. just the color of the car), set a threshold of intensity (close to 255 for an 8-bit image), and check each of the three channels for just the center pixel. If all three channel's pixels are above the threshold, the car's color is white, otherwise, the color should be the only channel that has an intensity value above the chosen threshold.

2. Additionally, you could just shine a light on the car and measure the wavelength of the reflected beams (since each color in the spectrum has a different wavelength).

2.16

The two subsets are adjacent with:

1. 4-adjacency: No

2. 8-adjacency: Yes

3. m-adjacency: Yes

2.23

I interpreted "one-dimensional" as meaning the images were stored in vectors. For the element-wise multiplication to make sense, both vectors must be either row vectors or column vectors with their starting an ending pixels in the same position. Alternatively, if "one-dimensional" referred to the number of channels, then orientation of the images does not matter as long as the each image's corner pixels are in corresponding locations.

2.39(b)

Given that the translation matrix is formed by $\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$ then, the inverse translation can be performed as $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -a_{13} \\ 0 & 1 & -a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$

2.39(c)

Given that the vertical and horizontal shearing matrices are formed by $\begin{bmatrix} 1 & s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $\begin{bmatrix} 1 & 0 & 0 \\ s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ respectively, then, the inverse shearing can be performed as $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{a_{12}} & 0 \\ \frac{1}{a_{21}} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$

Part II Matlab Programming

myhist

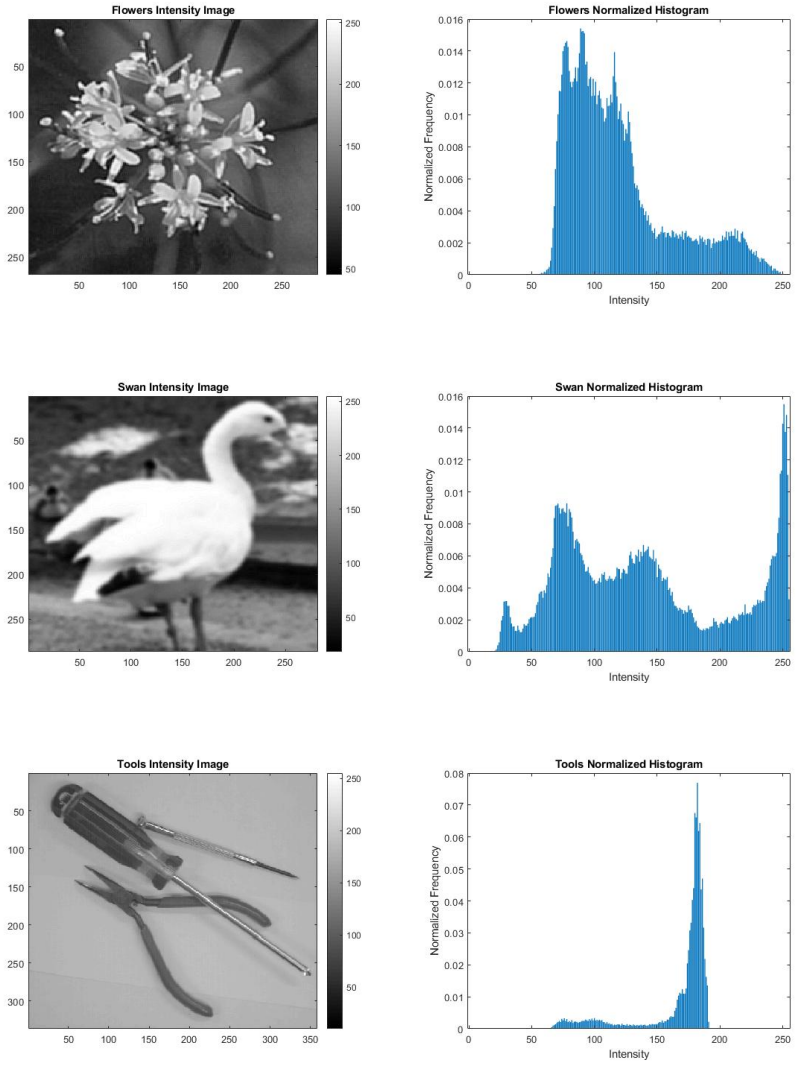

Figure 1: Left Column: Original 8-bit Intensity Images Right Column: Normalized Histogram of Image Intensities

From the image histograms, the following inferences can be made about each image:

- Flowers: Demonstrates low contrast, favors darker intensities
- Swan: Shows more high-contrast characteristics than the other images
- Tools: Exhibits low-contrast tendencies, favors lighter values

myhisteq

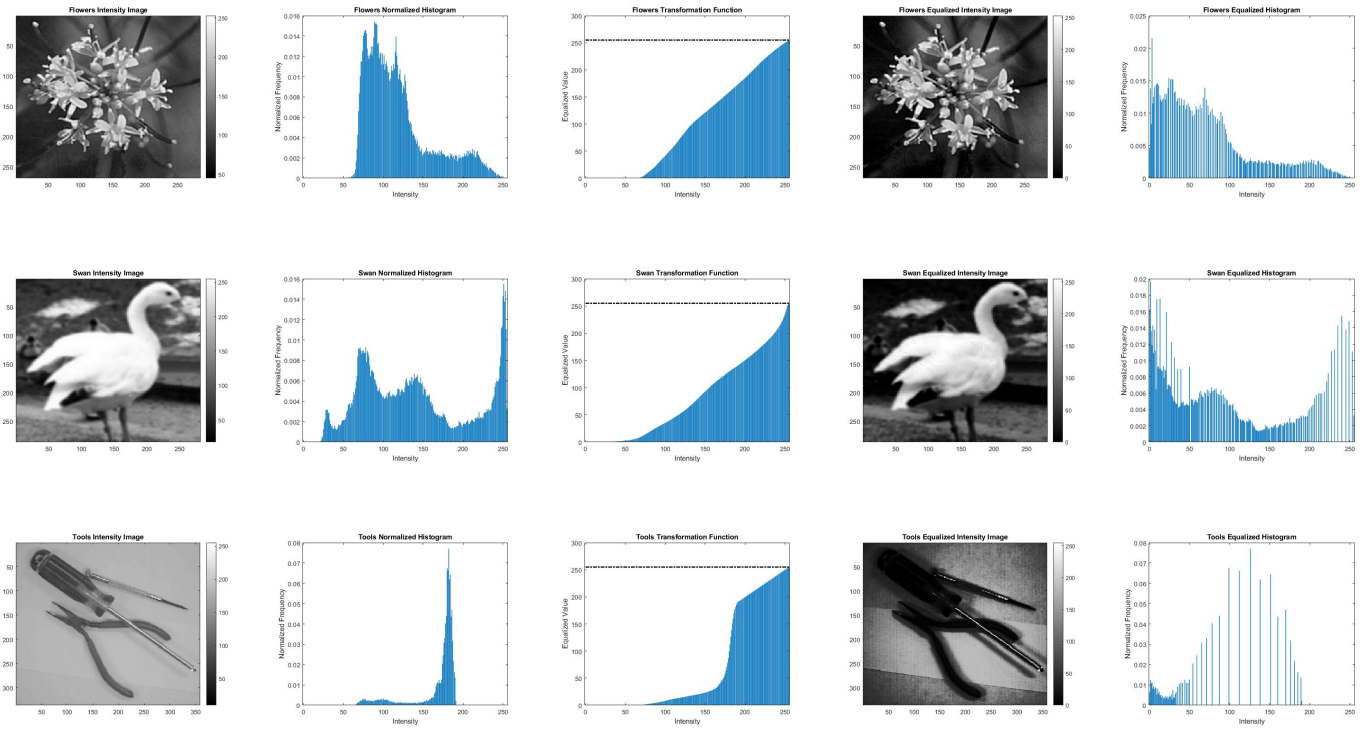

Figure 2: Left to right columns: Original 8-bit Intensity Images, Normalized Histogram of Image Intensities, Equalization Transformation Function, Equalized Intensity Image, Equalized Normalized Intensity Histogram

The above images demonstrate the outputs of the *myhisteq* function required for this assignment. The differences between the original intensity images and the equalized images transformed by the shown equalization functions can be observed visually, as well as quantitatively through the normalized image intensity histograms.

myquantize

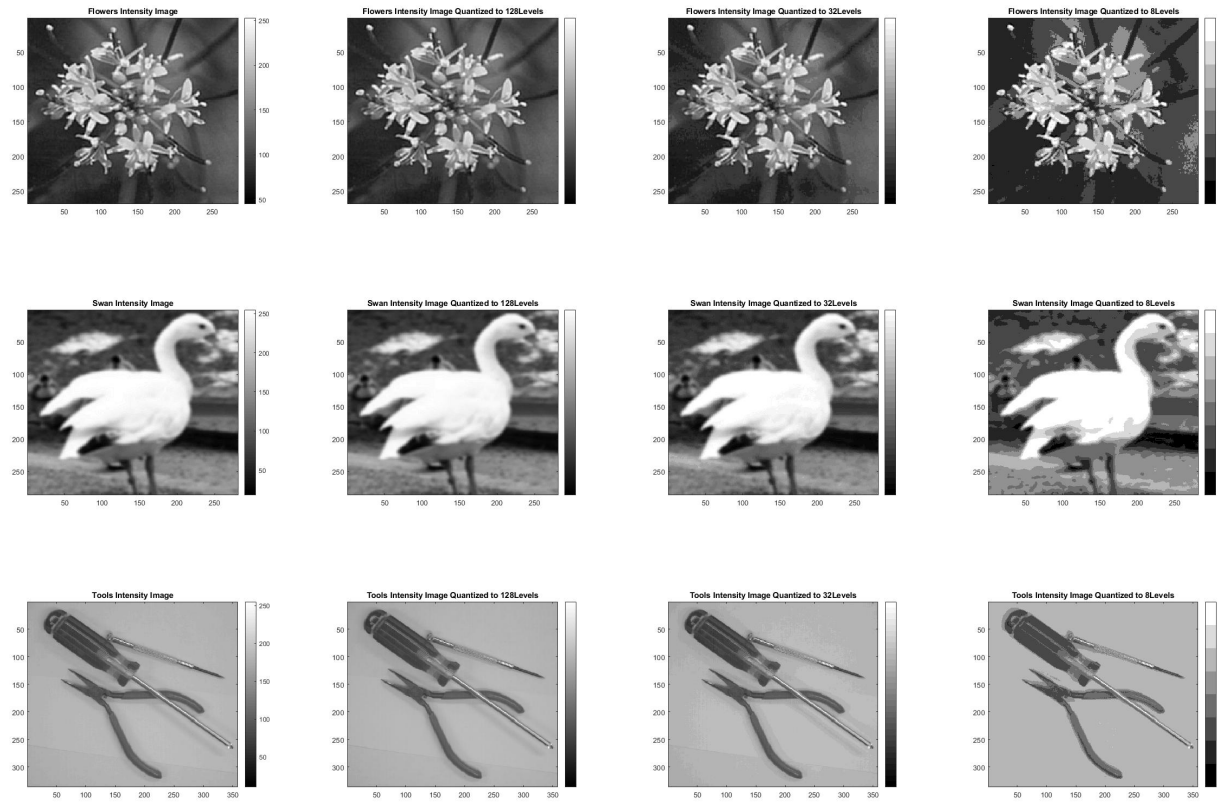

Figure 3: From left to right: Original 8-bit Intensity image, Image quantized to 128 levels, Image quantized to 32 levels, Image quantized to 8 levels

Algorithm 1 Quantization

procedure QUANTIZATION
  $quantNum \leftarrow$ number of intensity levels
  $f \leftarrow$ *pixel intensity*
  $i \leftarrow$ *pixel index*
  $Q \leftarrow$ *length of quantization intervals* $= \frac{255}{quantNum}$
  for i in numPixels **do**
    Index of quantized value, $Q_i(f_i) \leftarrow$ *floor*$(\frac{f_i}{Q})$
    Quantized value, $Q_i(f_i) \leftarrow Q_i(f_i)Q + \frac{Q}{2}$
return Quantized Image

The quantization function implemented for this assignment works as described in algorithm 1.
Accompanying code is provided in *mccurleyHW02.mat*