Michael Christensen
9/13/2013
CS 450

Homework Assignment #1

Written Exercises

1. **A standard music CD holds 74 minutes of stereo (2-channel) audio at 44,100 samples/second and 16 bits/sample/channel. How many megabytes of raw data can such a CD hold?**

74 minutes = 74 * 60 seconds = 4 440 seconds
4 440 seconds * 44 100 samples/second = 195 804 000 samples
195 804 000 samples * 16 bits/sample/channel = 3 132 864 000 bits/channel
3 132 864 000 bits/channel * 2 channels = 6 265 728 000 bits
6 265 728 000 / 8 = 783 216 000 bytes
783 216 000 bytes / 1 048 576 bytes/MB = **746.932 983 398 MB of raw data**

2. **G&W 2.5: A CCD camera chip of dimensions 7 x 7 mm, and having 1024 x 1024 elements, is focused on a square, flat area, located 0.5 m away. How many pairs per mm will this camera be able to resolve? The camera is equipped with a 35-mm lens.**

Camera distance = 500 mm
CCD width(height) = 7 mm
Focal length = ? mm
Horizontal field of view = camera distance * CCD width / focal length
= 500 * 7 / focal length

Resolution = # elements / horizontal field of view = 1024/ (500 * 7 / focal length)
= focal length mm * .2925 (elements / mm) = focal length * .2925 elements
focal length * .2925 elements / 2 = # pairs this camera will be able to resolve

3. **G&W 2.10: How many bits would it take to store a 2-hour HDTV movie**

1125 horizontal TV lines, with an aspect ratio of 16:9, means y/1125 = 16/9.
y (# vertical lines) = 16/9 * 1125 = 2000.
Thus the number of pixels is 2 000 x 1 125
2 000 x 1 125 = 2 250 000 pixels.
2 250 000 pixels * 24 bits of intensity resolution/pixel = 54 000 000 bits
Since it is interlaced, ½ of those bits are displayed at a time (1/60th of a second).
27 000 000 bits displayed every 1/60th of a second
27 000 000 * 60 = 1 620 000 000 bits/second
1 620 000 000 bits/second * 2 hours * 60 minutes/hour * 60 seconds/minute = **1.1664 x 10^13 bits**

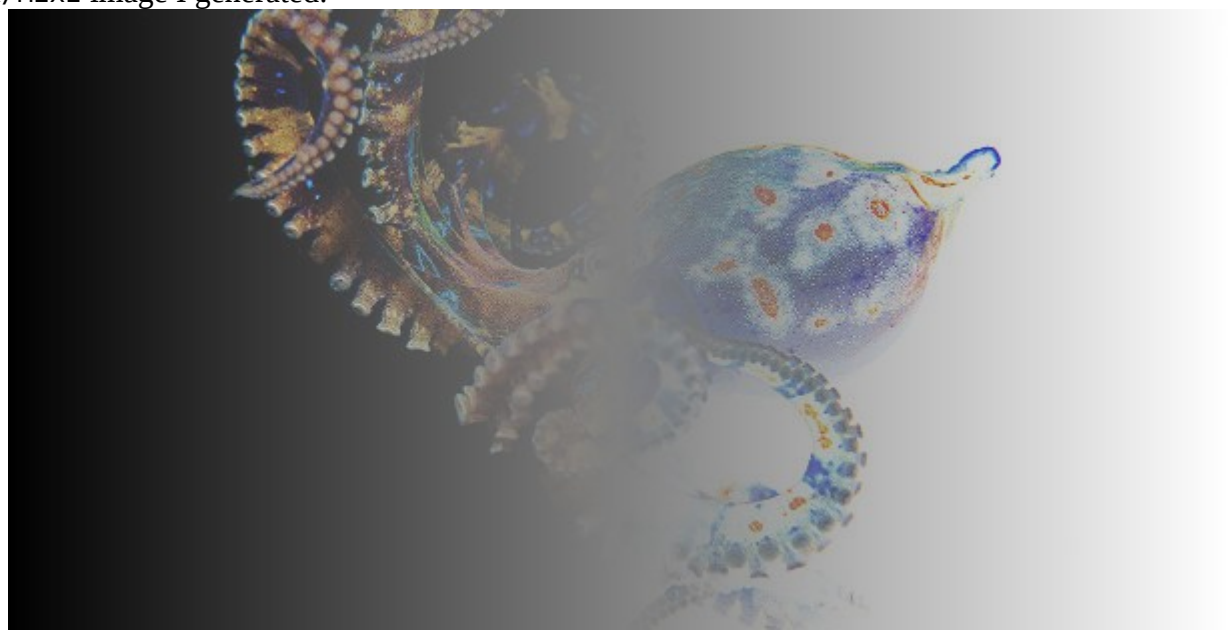
4. **G&W 2.19**: **Show that an operator that computes the median of a subimage area, *S*, is nonlinear.**

Example: Suppose there are two sequences of numbers corresponding to two rows in the subimage area, A = [2, 3, 4, 5, 6, 7], B = [1, 2, 3, 4, 1, 0], where each number represents the pixel's color. $A + B$ = [2 + 1, 3 + 2, 4 + 3, 5 + 4, 1 + 6, 7 + 0] = [3, 5, 7, 9, 7, 7,], and the median of this new sequence is 7. However, the median of A alone is 4.5, and the median of B alone is 1.5. The sum of these medians is 6. Since 6 != 7, the operator that computes the median of a subimage area, S, is nonlinear because the answers are dependent on the various orders in which the operation is performs.

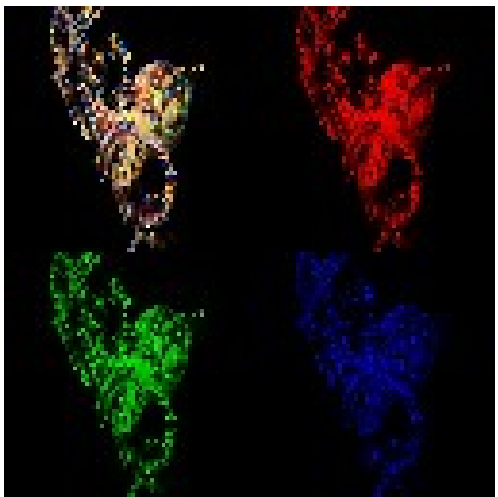
Programming Exercises

`Original` image I used (http://www-tc.pbs.org/wnet/nature/files/2008/09/610_ag_blue-ringed-octopus.jpg):

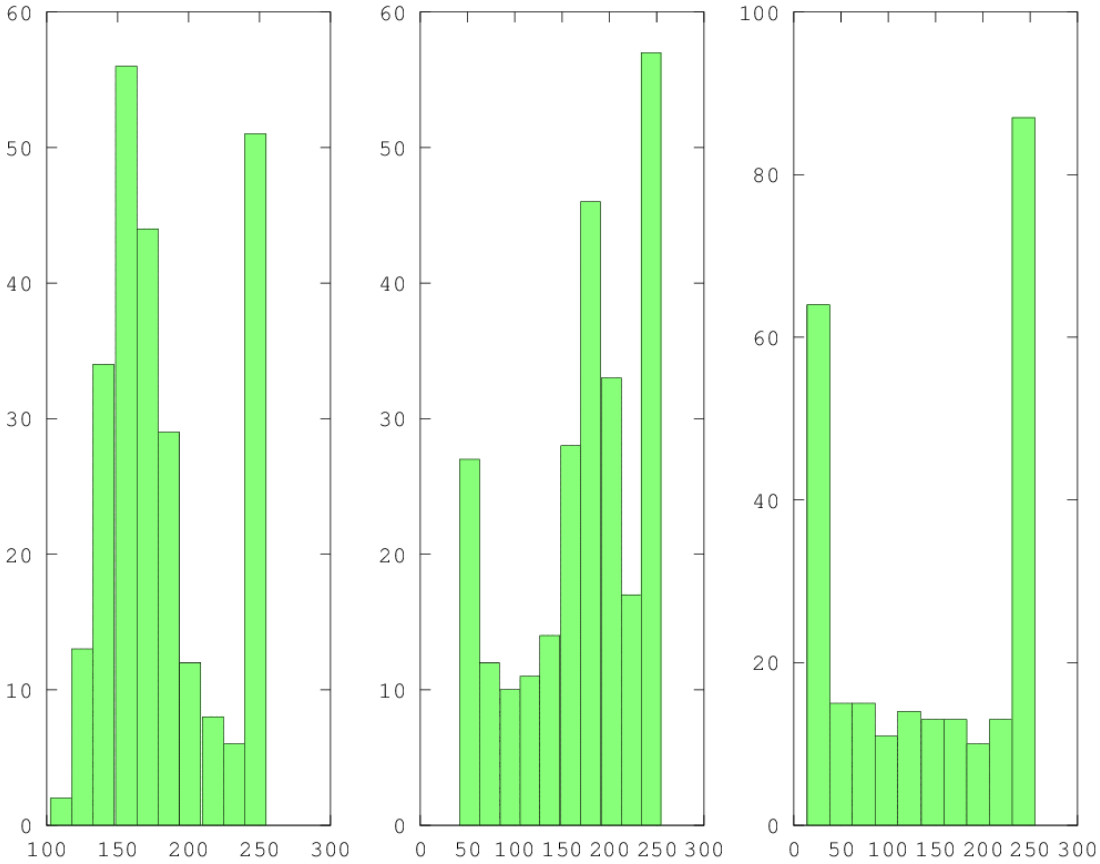
`Mix/Mix2` image I generated:



`Quad` image I generated:



Histograms of the original image:



Code used for histogram (Octave):

```
RGB = zeros(3,256,'uint8')
for r = 1:size(IMG,1)
      for c = 1:size(IMG,2)
            for colchan = 1:size(IMG,3)
                  colval = IMG(r,c,colchan)
                  RGB(colval+1,colchan)++
            end
      end
end

subplot(1,3,1)
hist(RGB(1:255,1))
subplot(1,3,2)
hist(RGB(1:255,2))
subplot(1,3,3)
hist(RGB(1:255,3))

print("histogram.png")
```

My preference between MATLAB/Octave or Python:
I preferred Matlab/Octave to Python, but only slightly. The documentation seems better (probably since Matlab, on which Octave is based, is a commercial product), and I encountered an obscure error with *imsave* in numPy, which could be because I don't have all the packages installed and Python is doing a poor job at telling me, or something else. Besides that, the differences I saw/felt between the two languages is so small that my decision to use Octave seems arbitrary at this point. I didn't find any syntactic differences between Matlab as you wrote it and the Octave code I was able to input into my interpreter, which is a plus.

Time it took for each part, problems, and suggestions:
Written Exercises: ~120 minutes (Problem 2 was pretty confusing to me (it took me the most time), while the others were very straightforward)
Matlab/Octave: 30 minutes for everything but the histograms, 30 minutes to learn how to make the histograms
numPy: 15 minutes (it was quicker since I had learned a lot with Octave by this point)

This is an Octave-specific problem, but it might help those who choose to use it. When I came to *imresize*, my (very recent) installation of Octave didn't have this package, which after some searching I found could be installed via the Octave command “pkg install <package>”. Octave's package manager didn't easily resolve dependencies (the “image” package needed the “signal” package which needed another package… etc.), so after more searching I used Ubuntu's apt-get to install the forge package that way, which worked. That took about 5 minutes to figure out.

Besides that, I felt that programming exercises were good because you gave a brief explanation of what was happening on the right without being too detailed, leaving it to us to figure out exactly why the syntax works. On the written assignment, like I mentioned already, problem 2 seemed pretty confusing, especially since I couldn't find much help in the book, besides Figure 2.3.