CSC 589 Introduction to Computer Vision

Problem set 1: Numpy and image processing tutorials.

Due date: September 12th, 1159pm

Team work: Not allowed.

Total points: 60(pt).

Introduction:

This is a short tutorial on how to use Python and a review of some small linear algebra ideas. The point of this assignment is to get you used to manipulating matrices and images in Python with NumPy. The TA Wenyan Bi will host recitation to help you with the problem set next week during the evening hours. She will send out a few time slots for you to sign up for.

NOTE: This problem set is not representative of future problem sets in terms of length or difficulty, but the logistics will be similar (submission, etc). The future problem sets will mostly composed of projects. This problem set is simply to ensure you have a sufficient understanding of the basic prerequisites for this class.

Here is a good and quick review of Numpy:
http://cs231n.github.io/python-numpy-tutorial/

Hand-in instructions:

Save your homework as Firstname_lastname_ps1.py and your report as FirstName_lastname_ps1.pdf. Please zip your code and your test images into a folder and upgrade to Blackboard. Please include all the test images you tend to run. The folder name MUST has the following structure:

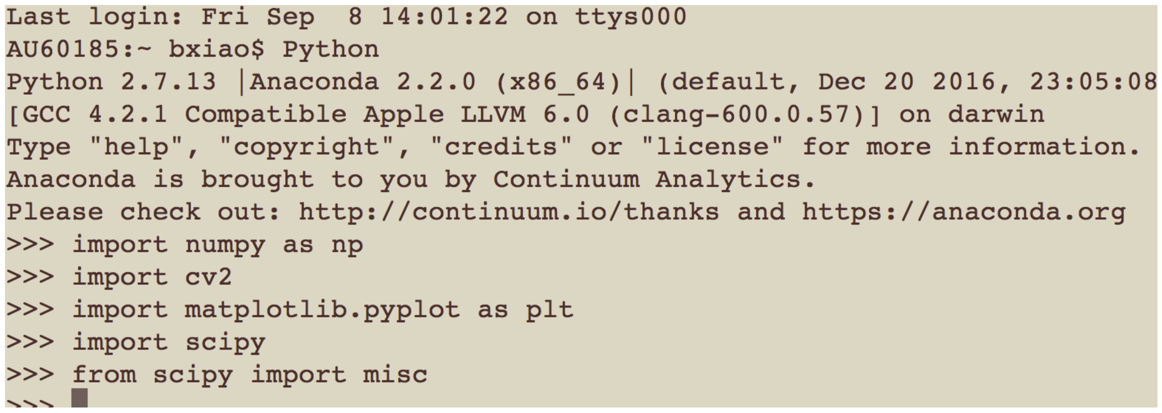
firstname_lastname_ps1

You must make sure after I download your folder, I can automatically run your code (without fussing around with image paths, etc).

You are also expected to turn in a write up document as pdf (you can use word or Latex for equations). For each problem, briefly explain what you your algorithm you have used and the answers to some of the questions asked in the problem. This report should also include the image results you generated. Please include your test images, your output images. Try to use subplot to plot multiple figures in one panel.

Make sure you use the provided ".py" files to write your Python code. You are also required to submit a pdf file of your report. You can paste your .py code and the plots in the report.

1. Set up Python library. Open the terminal and check if any of these things are happening correctly by comparing the following screen shot:



```
Last login: Fri Sep  8 14:01:22 on ttys000
AU60185:~ bxiao$ Python
Python 2.7.13 |Anaconda 2.2.0 (x86_64)| (default, Dec 20 2016, 23:05:08
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> import numpy as np
>>> import cv2
>>> import matplotlib.pyplot as plt
>>> import scipy
>>> from scipy import misc
>>> 
```

If you have errors, make sure you installed Anaconda or other Python libraries (numpy, matplotlib, scipy, and opencv).

2. Basic Matrix/Vector Manipulation (20 points)
   In Python, please calculate the following. Given matrix M and vectors a,b,c, such that:

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 0 & 2 & 2 \end{bmatrix}, a = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, b = \begin{bmatrix} -1 \\ 2 \\ 5 \end{bmatrix}, c = \begin{bmatrix} 0 \\ 2 \\ 3 \\ 2 \end{bmatrix}$$

a) Define Matrix M and Vectors a, b, c in Python. You should use Numpy.
b) Find the dot product of vectors a and b. (If you don't know what dot product means, read about dot product in any linear algebra tutorial.) Save this value to the variable aDotb and write its value in your report.
c) Find the element-wise product of a and b $[a_1 b_{1,} a_2 b_2, a_3 b_3]^T$ and write it to your report.

d) Find $\left(a^T b\right) Ma$ and write it to your report.
e) Without using a loop, multiply each row and of M element-wise by a. (Hint: the function repmat() may come in handy). Write the results in your report.
f) Without using a loop, sort all of the values of the new M from (e) in increasing ad plot them in your report.

3. Basic Image Manipulations (20 points)

a) Read in the images, image1.jpg and image2.jpg
There re many different ways to read in images. Matplotlib.image is a good one. cv2.imread() is another good one. You can also use scipy.misc.imread().

b) Convert the images to double precision and rescale them to stretch from minimum value 0 to maximum value 1.
c) Add the images together and re-normalize them to have minimum, value 0 and maximum value 1. Display this image in your report.
d) Create a new image such that the left half of the image is the left half of image1 and the right half of the image is the right half of image 2.
e) Using a for loop, create a new image such that every odd numbered row is the corresponding row from image1 and the every even row is the corresponding row from image2 (Hint: Remember that indices start at 0 and not 1 in Python). Display this image in your report.

f) Accomplish the same task as part e without using a for-loop (the functions reshape and repmat may be helpful here).

g) Convert the result from part f to a grayscale image. Display the grayscale image with a title in your report.

4. Compute the average face. (20pts).
a) Download labeled faces in the Wild dataset (google: LFW face dataset or click the link). Pick a face with at least 100 images.
b) Call numpy.zeros to create a 250 x 250 x 3 float64 tensor to hold the results.
c) Read each image with skimage.io.imread, convert to float and accumulate.
d) Write the averaged result with skimage.io.imsave. Post your resulted image in the report.