CS 231A Computer Vision (Spring 2017)
Problem Set 0
Due Date: April 12th 2017 11:59pm

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.

Make sure you use the provided ".py" files to write your Python code. Since we will be using Gradescope, the best option would be to take a screenshot of your code and put it in your pdf.

NOTE: This problem set is not representative of future problem sets in terms of length or difficulty, but the logistics will be similar (submission, Piazza, etc).

Most likely, you will find the future problem sets to be more difficult than this one. This problem set is simply to ensure you have a sufficient understanding of the basic prerequisites for this class.

Submit your published results to the class Gradescope. Feel free to ask any questions to the class Piazza forum (but use a private post if you have code or specifics to discuss).

1 Piazza and Background Poll (10 points)

Please register on the class Piazza forum and answer the Google Forms poll about your class background: `https://goo.gl/UU5Qna`.

2 Basic Matrix/Vector Manipulation (10 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. One helpful Python package that is commonly used for linear algebra related problems is Numpy.

(b) Find the dot product of vectors a and b (i.e. $a^\top b$). Save this value to the variable `aDotb` and write its value in your report.

(c) Find the element-wise product of a and b $[a_1b_1, a_2b_2, a_3b_3]^T$ and write it in your report.

(d) Find $(a^\top b)Ma$ and write it in your report.

(e) Without using a loop, multiply each row of M element-wise by a. (Hint: The function repmat() may come in handy). Write the result in your report.

(f) Without using a loop, sort all of the values of the new M from (e) in increasing order and plot them in your report.

3 Basic Image Manipulations (20 points)

(a) Read in the images, image1.jpg and image2.jpg, as color images.

(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 image2. Display this image in your report.

(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 Singular Value Decomposition (20 points)

(a) Read in image1 as a grayscale image. Take the singular value decomposition of the image.

(b) Recall from the discussion section that the best rank n approximation of a matrix is $\sum_{i=1}^{i=n} u_i \sigma_i v_i^\top$, where u_i, σ_i, and v_i are the ith left singular vector, singular value, and right singular vector respectively. Save and display the best rank 1 approximation of the (grayscale) image1 in your report.

(c) Save and display the best rank 20 approximation of the (grayscale) image1 in your report.