CS231A: Computer Vision, From 3D Perception to 3D Reconstruction and beyond Homework #0

(Spring 2024) Due: **Monday, April 8**

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

The assignment will require you to complete the provided python files and create a PDF for written answers. The instructions are bolded for parts of the problem set you should respond to with written answers. To create your PDF for the written answers, we recommend you add your answers to the latex template files on our website, but you can also create a PDF in any other way you prefer. To implement your code, make sure you modify the provided ".py" files in the code folder.

To test your code, you can choose to install the required packages and run the code yourself, or you can upload the provided PSET0.ipynb file to Google Drive and complete the code with an online interface. Here are instructions for the latter approach. In Google Drive, follow these steps:

a. Click the wheel in the top right corner and select Settings.

b. Click on the Manage Apps tab.

c. At the top, select Connect more apps which should bring up a GSuite Marketplace window.

d. Search for Colab then click Add.

e. Now, upload "PSET0.ipynb", open it, and follow the instructions inside.

There will be two assignments to submit to on Gradescope: one for coding files and one for written answers. The former will be graded by an autograder and the latter will be graded by us, so you should submit to both. To submit the python files to the autograder, create a zip file containing the ".py" files and upload this zip file to the Gradescope assignment. On to the problems!

1 Basic Matrix/Vector Manipulation (20 points)

In Python, calculate the following by filling out p1.py. 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 using Numpy.

(b) Find the dot product of vectors a and b (i.e. $a^\top b$).

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

(d) Find $(a^\top b)Ma$.

(e) Without using a loop, multiply each row of M element-wise by a. **Briefly explain the logic of your code in your written 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. **Briefly explain the logic of your code in your written report.**

2 Basic Image Manipulations (40 points)

Do the following by filling out p2.py:

(a) Read in the images, image1.jpg and image2.jpg, as color images using io.imread from the skimage package.

(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. **Save and include 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. **Save and include 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). **Save and include this image in your report.**

(f) Accomplish the same task as part e without using a for-loop (the functions reshape and tile may be helpful here). **Briefly explain the logic of your code in your written report.**

(g) Convert the result from part f to a grayscale image. **Save and include the grayscale image in your report.**

3 Singular Value Decomposition (40 points)

Do the following by filling out p3.py:

(a) After reading in image1 as a grayscale image (this is done for you in the code already) 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 include the best rank 1 approximation of the (grayscale) image1 in your report.**

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