

## Exercise 1: Image manipulation

**First import the library you might need.**

```
import matplotlib.image as mpimg
```

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
import soundfile          # you may need to install soundfile by 'pip install soundfile'
```

Part 1: Fade an image with a scalar.

Read the '**cameraman.tif**' image from a file using `mpimg.imread` function (image read function). This function will load the image in a matrix.

```
Cameraman_image = mpimg.imread('cameraman.tif',1)
```

Show the image using the following commands?

```
plt.figure()
```

```
plt.imshow(Cameraman_image)
```

What is the size of the `Cameraman_image` matrix?

Assign `fade_factor=0.5`;

Multiply the `fade_factor` scalar and the `Cameraman_image` matrix and affected to the variable `Fade_image`?

Reconstruct the image from the obtained matrix using?

```
plt.figure()
```

```
plt.imshow(Fade_image)
```

Create a new image named `First_part_image` by selecting the first 100 components of each dimension. Show this image.

Create a new image named `last_part_image` by selecting the last 100 components of each dimension. Show this image.

## Part 2 - Version 1:

Load the image '**5.1.09.tif**' which is a surface in the moon. Assign this image to the variable `Moon_image`.

Show `Moon_image`

```
plt.figure()
```

```
plt.imshow(Moon_image)
```

Fade for example *Moon\_image* by a factor of 0.8 and *Cameraman\_image* by factor of 0.2 and sum the two matrices in new matrix named *Mixte\_image*.

Show the *Mixte\_image*

```
plt.figure()
```

```
plt.imshow(Mixte_image)
```

Create a new image named *First\_part\_image\_1* by selecting the first 100 components of each dimension of image *Cameraman\_image*.

Create a new image named *last\_part\_image\_2* by selecting the last 100 components of each dimension of image *Moon\_image*.

Fade both images *last\_part\_image\_2* by a factor of 0.8 and *First\_part\_image\_1* by factor of 0.2 and sum the two matrices in new matrix named *last\_part\_Mixed\_image*.

## Part 2 - **Version 2:**

We will do the same fade and mixing process as version 1 question but this time using matrix and vector multiplication.

The first step will consist of changing both image matrices from size of 256x256 each to vector of size 56536x1; (To create a vector from matrix uses the function `reshape`?. Use the help to show you how to use `reshape` function?)

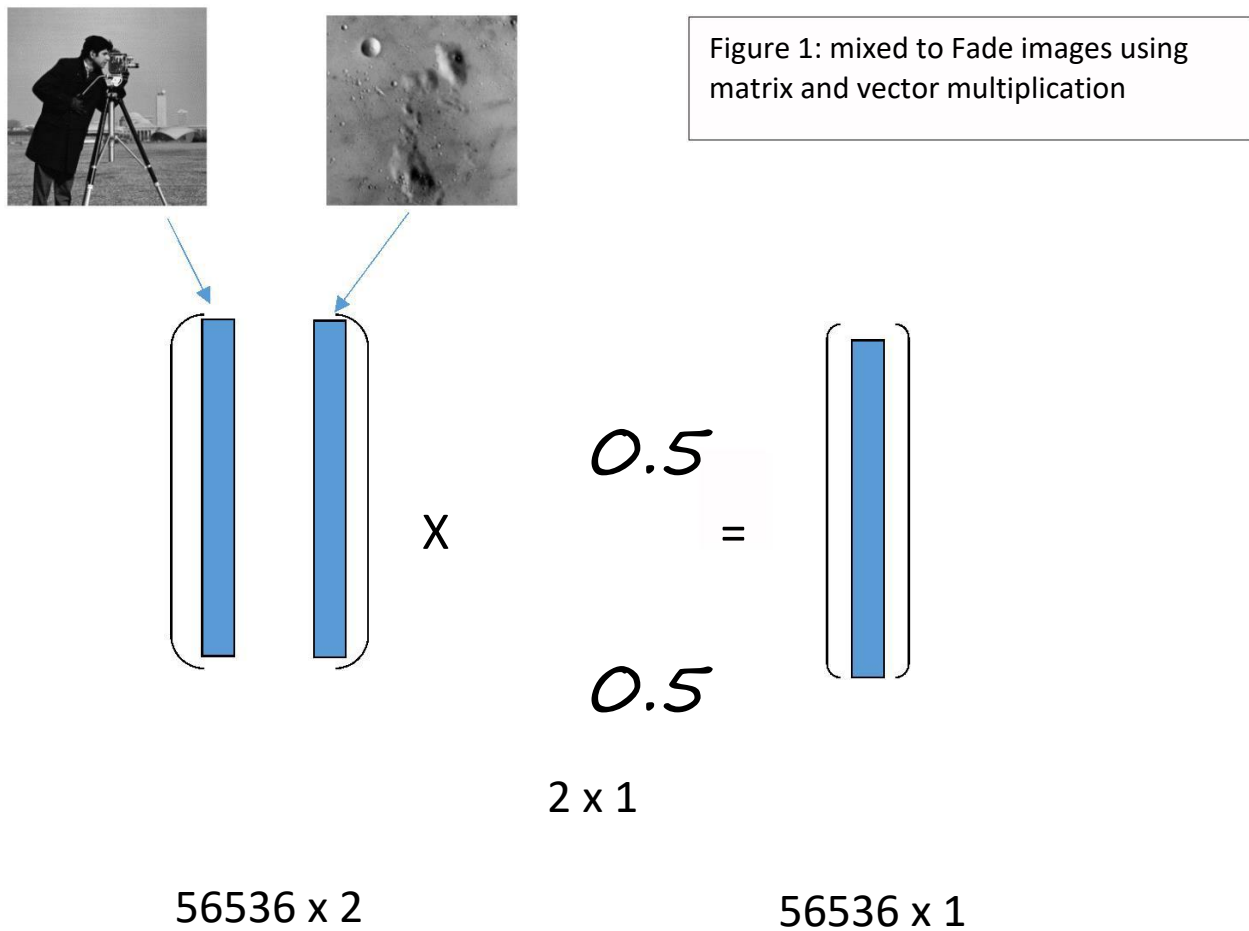
Create a new matrix (named *Both\_images*) by appending both image vectors to form a matrix of size 56536x2.

Create of vector named *Fade\_vector* of size 2x1 and containing the fade factor values (0.5,0.5) for both image.

Multiply *Both\_images* matrix and *Fade\_vector* to obtained the *mixing\_image\_vector*

Resize using the function `reshape` again the obtained vector *mixing\_image\_vector* to create *mixing\_image\_mtarix* of size (256x256)?

Please see the Figure 1 to show all the steps to solve this section?



## Exercise 2: Quantization

Load this segment of speech into Python workspace using

```
p, samplerate = soundfile.read('filename1.wav')
```

where *soundfile.read* is Python function which reads contents of a file in wave format into an array.

Change resolution from 16 bits/sample to 8 bits/sample

```
soundfile.write('filename2.wav', p, samplerate, subtype='PCM_U8')
```

```
p1, samplerate = soundfile.read('filename2.wav')
```

Change resolution from 16 bits/sample to 1bit/sample –  $\gg p2 = [(x>0)+0.0 \text{ for } x \text{ in } p1];$

Now p, p1, p2 respectively contains same speech at different resolutions 16 bits/sample,

8bits/sample and 1 bit/sample, but all at the same sampling frequency 8 kHz.

Store *p2* as filename3.wav and play all the three wav files, note the observations:

### Exercise 3: **Aliasing Effect:**

It occurs when we do not use correct sampling frequency on the signals or when we do not process the original audio signal properly (means low pass filtering)

Experiment demonstrating aliasing effect:

- 1) Create two cosine waves of frequencies 100Hz and 600Hz and plot them?
- 2) Sample them with 500Hz i.e., take 500 samples per sec at equal intervals from those signals and plot.
- 3) Compare both plots and write observations

Writing your observations in your codes (comment them) is OK.

#### Submission Requirement:

Please zip the 'Python' folder as Lab1\_YourName\_YourJHID.zip, containing the codes **py file** (or notebook, if you prefer), together with all the **output and input**; make sure we can directly run your codes and get all the figures; submit the zip file.