Q1.

a)

$$y_i = \text{softmax}(z)_i = \frac{\exp(z_i)}{\sum_{j=1}^{K} \exp(z_j)}.$$

$$\text{softmax}(z + C1) = \frac{\exp(z_i + C)}{\sum_{j=1}^{K} \exp(z_j + C)}$$

$$= \frac{\exp(C)\exp(z)_i}{\exp(C)\sum_{j=1}^{K} \exp(z)_j} = \frac{\exp(z_i)}{\sum_{j=1}^{K} \exp(z_j)} = \text{softmax}(z).$$

b) let $y_i = \text{softmax}(z)_i$

$$D_j y_i = \frac{\partial y_i}{\partial z_j} = \frac{\partial \frac{e^{(z_i)}}{\sum_{j=1}^{K} e^{(z_j)}}}{\partial z_j}$$

By applying the quotient rule & by using

for $j = i$ $\quad\Sigma$ as $\sum_{j=1}^{K} e^{z_j}$ we get

$$\partial \frac{e^{(z_i)}}{\sum_{j=1}^{K} e^{(z_j)}} = \frac{e^{z_i}\Sigma - e^{z_j}e^{z_i}}{\Sigma^2}.$$

reordering a bit.

we get $\frac{e^{z_i}}{\Sigma} \frac{\Sigma - e^{z_j}}{\Sigma}$

$$= y_i(1 - y_i)$$

for $j \neq i$

$$\frac{\partial \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_K}}}{\partial z_j} = \frac{0 - e^{z_j} e^{z_i}}{\Sigma^2}.$$

$$= -\frac{e^{z_j}}{\Sigma} \frac{e^{z_i}}{\Sigma}$$

$$= -S_j S_i$$

c) $y = \text{softmax}(z)$ , $z = W^T x$

$$y_i = \frac{e^{w^T x_i}}{\sum_{j=1}^{K} e^{w^T x_j}}$$

$$\frac{\partial y_i}{\partial (w_j)} = \sum_{K=1}^{K} \frac{\partial y_i}{\partial z_K} \cdot \frac{\partial z_K}{\partial w_j}$$

$$= \sum_{K=1}^{K} y_i(\delta_{ik} - y_k)(\delta_{jk} \cdot x) \quad \left[\frac{\partial z_K}{\partial w_j} = \delta_{jw} \cdot x\right]$$

$$= y_i (\delta_{ij} - y_j) \cdot x \quad \text{when } k = j$$

$$\frac{\partial y_i}{\partial x} = \sum_j \frac{\partial y_i}{\partial z_i} \frac{\partial z_i}{\partial x} \quad \left(z_j = w_j^T x = \frac{\partial z_j}{\partial x} = w_j^T\right)$$

$$= \sum_{j=1}^{K} (y_i(\delta_{ij} - y_j)) \cdot w_j^T$$

2) a) $V =$

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

It rotates by 45° clockwise. $\theta = 315$ anti-clockwise or $\theta = -45°$ clockwise b)

$$V = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \rightarrow V^T = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$V^{-1} = \frac{1}{\det} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \frac{1}{\frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}}} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = V^{-T}$$

$$V^T x = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

Rotates by $\theta = 45$. anticlockwise.

c)

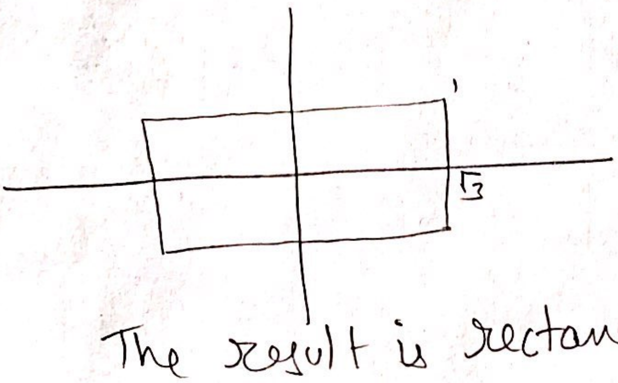
$$\Sigma V^T \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & -\frac{\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$\begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & \frac{-\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix} = \begin{bmatrix} \sqrt{3} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & \frac{-\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix} = \begin{bmatrix} -\sqrt{3} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & \frac{-\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} -\sqrt{2} \\ 0 \end{bmatrix} = \begin{bmatrix} -\sqrt{3} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & \frac{\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ -\sqrt{2} \end{bmatrix} = \begin{bmatrix} \sqrt{3} \\ -1 \end{bmatrix}.$$




The result is rectangle.

d)

$$\begin{bmatrix} \frac{1}{2} & \frac{-\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \cos 60 & -\sin 60 \\ \sin 60 & \cos 60 \end{bmatrix}$$

rotates anticlockwise as $\theta = 60$.

e)

$$U = \begin{bmatrix} \frac{1}{2} & \frac{-\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix} \qquad \Sigma V^T = \begin{bmatrix} \frac{\sqrt{3}}{\sqrt{2}} & \frac{-\sqrt{3}}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$A = U \Sigma V^T = \begin{bmatrix} 0 & \frac{-\sqrt{3}}{\sqrt{2}} \\ \sqrt{2} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

Consider B as a Singular Vector Decomposition.

$$Ax = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \begin{bmatrix} -\sqrt{3} \\ -1 \end{bmatrix} \begin{bmatrix} 0 \\ -2 \end{bmatrix} \begin{bmatrix} \sqrt{3} \\ 1 \end{bmatrix}$$

# Q3.Colorizing the Prokudin - Gorskii Photo Collection

**Project Description:**

The goal of this assignment is to learn to work with images by taking the digitized Prokudin-Gorskii glass plate images and automatically producing a color image with as few visual artifacts as possible. In order to do this, you will need to extract the three-color channel images, place them on top of each other, and align them so that they form a single RGB color image.

Given an image with 3 parts as an RGB image. The project is to extract the 3 parts from the image in a manner to align the form a colored image from the 3 RGB image portions.

*Figure1: The image above is the final product of the image aligned from the following image set.*

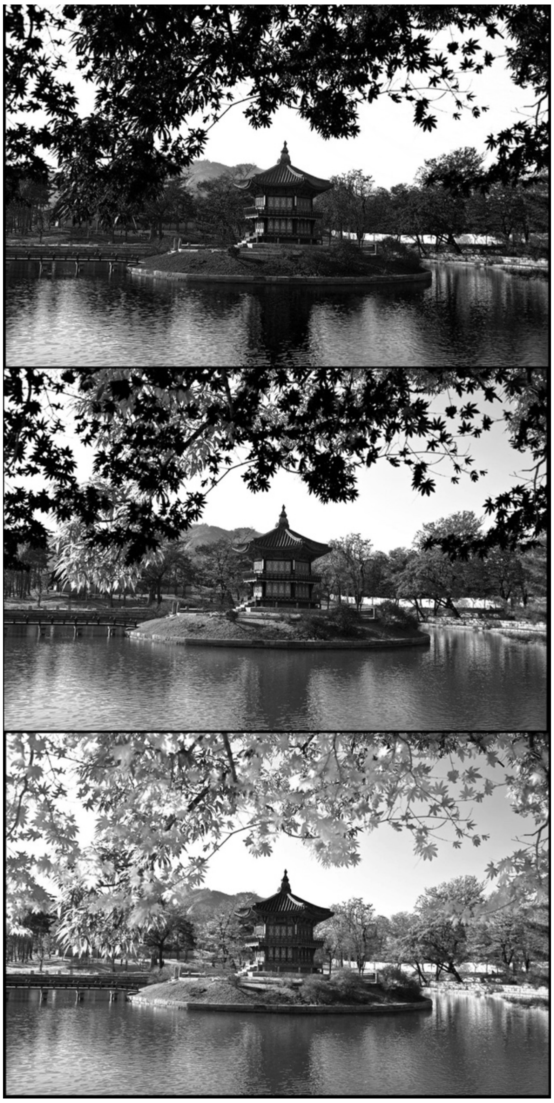


*Figure 2:Given Sample image from which the data has to extracted.*

**Part 1: Combining the image.**

In this part the job was to basically divide the image into 3 BGR parts and then save the image to stack them up together to form a colored image from a set of black-white images.

In order to produce the colored image, the computer has to first understand that these are in fact part of a colored image. The code in python divides the image as a matrix. We start by reading the whole image and then calculate the height of the image and this height is then divided into 3 parts so as to distinguish the 3 images. We then use a function to stack the image one over the other and Python then produces the colored image.

The sample data is shown below-

*Figure 3 Output produced by the code in Part 1.*

The code produces the output that look like a colored image but are not properly aligned.

This is where the second part comes in.

**Part 2: Aligning the image**

It is clear from the samples produced by the part one that we can produce a set of colored images from the given sample image.

I order to align the image blue color was taken as the anchor color. Then the offset of the other colors i.e. red and green were taken from the blue color. This offset was calculated using zero mean normalized cross correlation. We know that same object in its individual particular layer will have the similar pixel intensity. So, using this knowledge I tried to minimize the offset values.

*Figure 4:00125v.jpg Green offset: [4,0] Red offset: [13,2]*

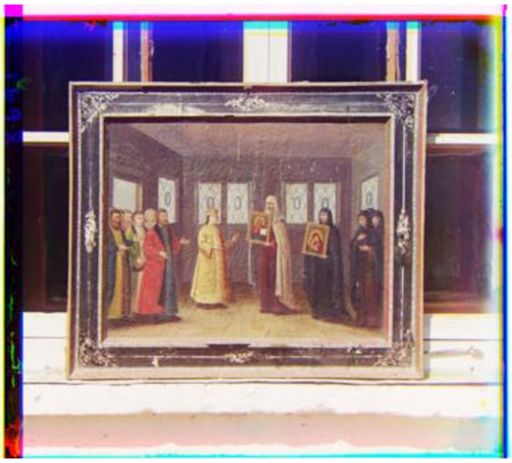


*Figure 5: 00149v.jpg Green offset: [6,2] Red offset:[8,-2]*

*Figure 6: 00153v.jpg Green offset:[16,2] Red offset:[11,3]*

*Figure 7:00351v.jpg Green offset:[4,2] Red offset:[7,2]*

*Figure 8: 00398v.jpg Green offset:[6,2] Red offset:[10,1]*

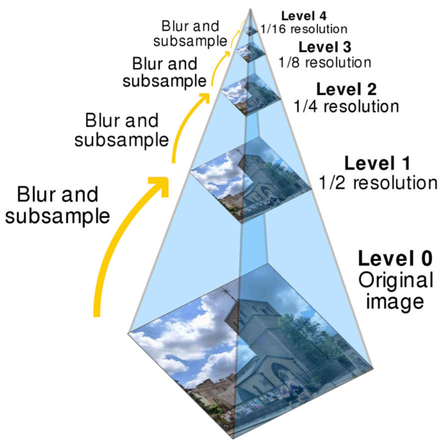
Figure 4-8 output for Code for Part 2.

**Part 3: Aligning for Big Images:**

The approach used in the part 2 is good but it might not work for big images with dimensions in range 3000 x 1600 pixels.

So, to solve the offset with the bigger images the method of pyramid is used.

Here, for a large image instead of calculating the offset for a very large value such as [-30,30] which will take a lot of time, we rescale (here decrease) the image to a factor of 2. Then the offset is calculated for the same small [-15,15] values and the image is rolled by offset*2 value and then again, the offset is calculated for a small value [-5,5] so as to produce the correct image.




In fact, there is a pyramid algorithm which uses a divide and conquer algorithm that efficiently limits the search space. An image pyramid represents the image at multiple scales (in our case a factor of 2) and the processing is done sequentially starting from the coarsest scale and going down the pyramid, updating your estimate as we go deeper. Since we can use the naive version to calculate the most coarsest scale, that limits our search space greatly

*Figure 9 vancouver_tableau.jpg Green Offset: [-10,-10] delta[-3,-4] Blue Offset:[-10,-12] delta[-4,-3]*

*Figure 10: seoul_tableau.jpg Green Offset: [2,4] delta:[0,1] Blue Offset:[2,-2] delta[0,1]*

Q3 part 4Extra Credit:

Cropping-

This part includes the cropping of the margins so as to make the picture more presentable.

*Figure 11 seoul_tableau.jpg with cropped borders*

*Figure 12 vancouver_tableau.jpg with cropped borders*