Suvam Dey (SR No: 24290)
MTech Signal Processing, Department of ECE, IISc, Bengaluru

Digital Image Processing Assignment 2

1. **Spatial Filtering and Binarisation:** Apply Gaussian blurring on the image `‘moon_noisy.png’`. Generate a spatial Gaussian filter of size 41×41 by generating a filter kernel as:

$$G_f(x, y) = \frac{1}{K} \exp\left\{\frac{-(x^2 + y^2)}{2\sigma_g^2}\right\}, \tag{1}$$

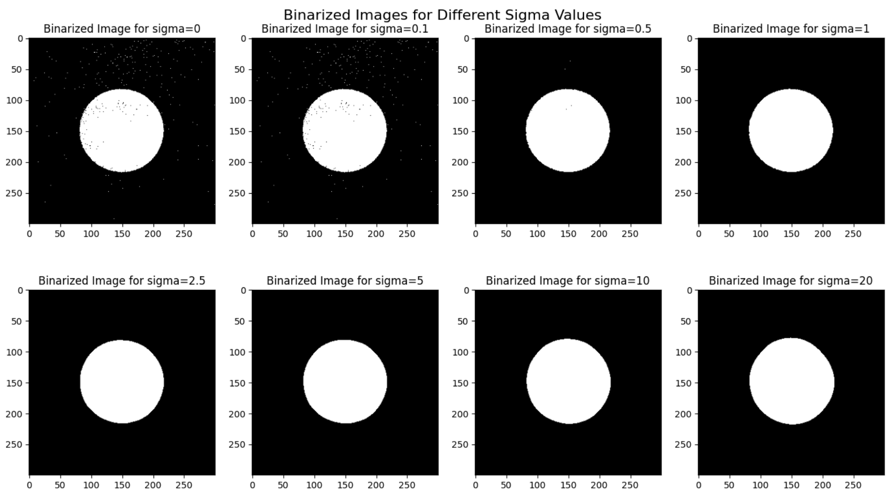
where $x, y \in \{-20, -19, \cdots, 19, 20\}$ and σ_g is the standard deviation of the Gaussian kernel. K normalizes the filter such that $\sum_x \sum_y G_f(x, y) = 1$. You can use a library function to convolve the input image with the kernel you created to obtain the Gaussian blurred image.

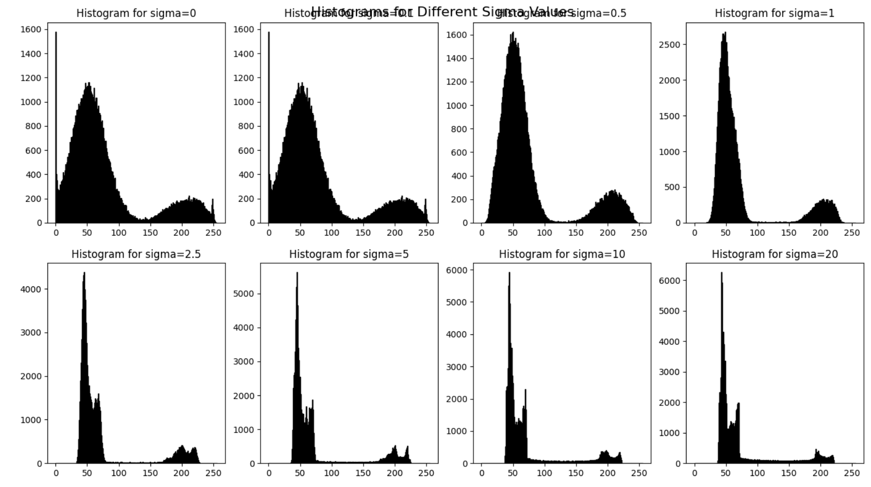
Apply Otsu's Binarization algorithm on the blurred image and note the optimal within-class variance $\sigma_w^{2\,*}$ for a given σ_g blur parameter.

Plot the histogram and the binarized image for the blurred images and find their corresponding optimal within-class variances $\sigma_w^{2\,*}$ for each $\sigma_g \in \{0, 0.1, 0.5, 1, 2.5, 5, 10, 20\}$ (0 corresponds to the input image itself). Find the optimal σ_g than minimizes $\sigma_w^{2\,*}$. Comment on your observations.

(15 Marks)

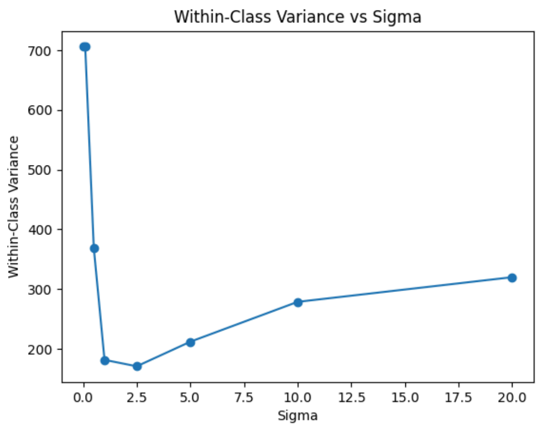
Output:

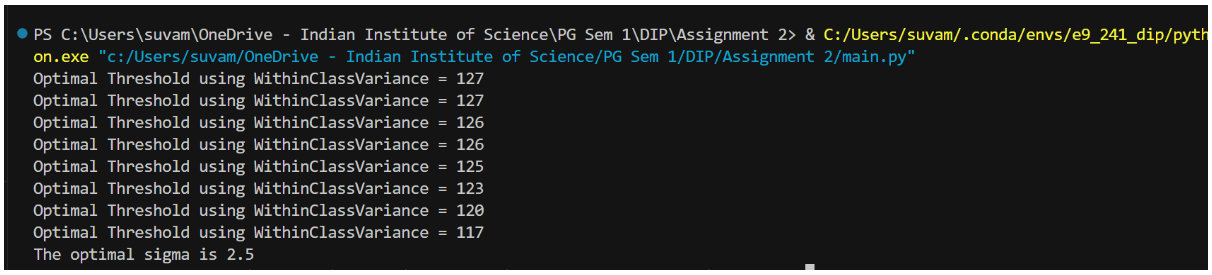

Binarized Images for Different Sigma Values
Binarized Image for sigma=0
Binarized Image for sigma=0.1
Binarized Image for sigma=0.5
Binarized Image for sigma=1
Binarized Image for sigma=2.5
Binarized Image for sigma=5
Binarized Image for sigma=10
Binarized Image for sigma=20



Histogram for Different Sigma Values

Suvam Dey (SR No: 24290)
MTech Signal Processing, Department of ECE, IISc, Bengaluru





```
● PS C:\Users\suvam\OneDrive - Indian Institute of Science\PG Sem 1\DIP\Assignment 2> & C:/Users/suvam/.conda/envs/e9_241_dip/pyth
  on.exe "c:/Users/suvam/OneDrive - Indian Institute of Science/PG Sem 1/DIP/Assignment 2/main.py"
  Optimal Threshold using WithinClassVariance = 127
  Optimal Threshold using WithinClassVariance = 127
  Optimal Threshold using WithinClassVariance = 126
  Optimal Threshold using WithinClassVariance = 126
  Optimal Threshold using WithinClassVariance = 125
  Optimal Threshold using WithinClassVariance = 123
  Optimal Threshold using WithinClassVariance = 120
  Optimal Threshold using WithinClassVariance = 117
  The optimal sigma is 2.5
```

Terminal Window Displaying the Optimal Threshold obtained using different σ_g values

Inference:

Upon generating the Gaussian Kernel for the different values of σ_G as given in the question, it was convolved with the image, giving different thresholding results as shown. The histogram is much broader, close to the original image with very less effects on the salt and pepper noise, resulting in artefacts remaining post thresholding when filtering was done with smaller values of σ_G. This is because with smaller values of σ_G, there is less blurring so the noisy pixels are not got rid of.

The within class variance value quickly reduced with increasing σ_G to the optimal value of **2.5**, followed by slight increase with minimal change seen in the histogram shape as seen in the third plot.
With higher value of σ_G, the threshold image identifies a distorted image of the foreground.
On the terminal window, the optimal threshold intensity selected has been shown.

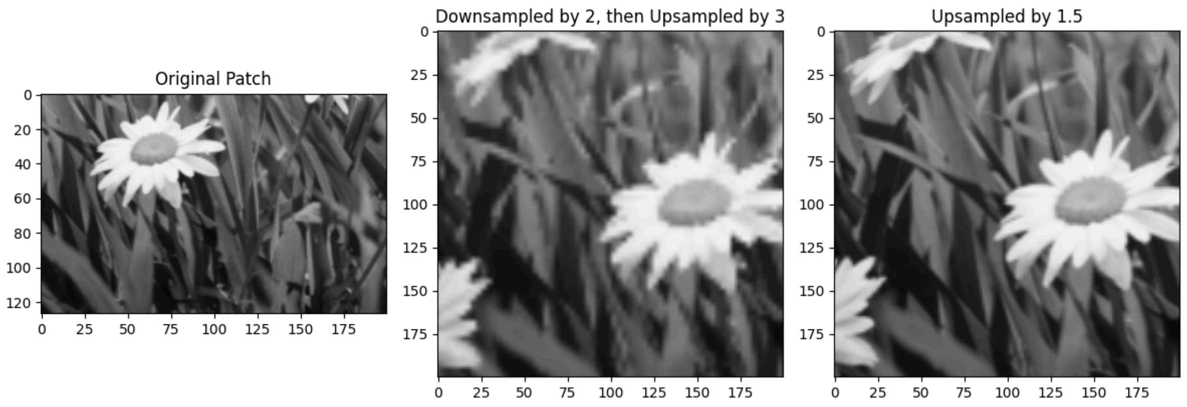
2. **Fractional Scaling with Interpolation:**
   (a) Downsample the image `'flowers.png'` by 2 and upsample the result by 3 using bilinear interpolation.
   (b) Upsample the image by $3/2 = 1.5$ using bilinear interpolation.

   Observe what is different in both results and give comments.

   Hint: Use a zoomed-in patch of the image for a better visual comparison.

   (15 Marks)

Output:



Inference:

In this question, two methods of scaling have been applied to get slightly different results. To understand the difference between the two outputs, a zoomed in patch selected at random has been shown. In the first output image, the original image was first downsampled by 2. This led to loss of information in the image, since every second pixel was got rid of. This was followed by upsampling by 3, done using bilinear interpolation, but a lot of blurry and pixelated areas remain, reducing image quality, in spite of upsampling. In the second output, the upsampling was done by a factor of 1.5, which gives a much smoother image, with almost no pixelations visible. This is because no extra information is lost in this process, and all the information available in the initial image was been used in the bilinear interpolation proces has been utilised.

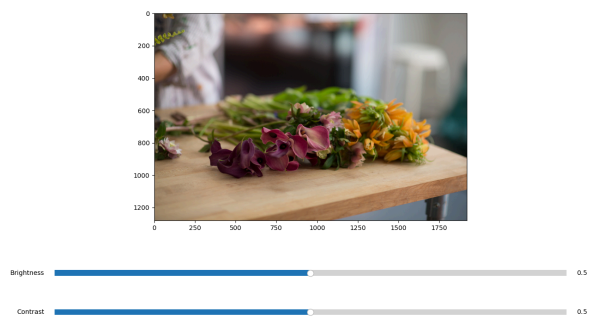
Suvam Dey (SR No: 24290)
MTech Signal Processing, Department of ECE, IISc, Bengaluru

3. **Photoshop Feature:** Watch the video explaining the Brightness/Contrast feature in Adobe Photoshop (‘`photoshop_feature.mp4`’). Implement this based on your knowledge of pointwise operations applied to images.

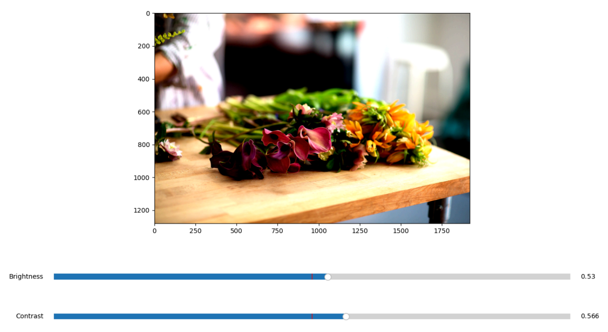
   Implement two functions `brightnessAdjust(img,p)` and `contrastAdjust(img,p)`, that output the respective adjusted images where `p` controls the respective adjustments. In particular, for `brightnessAdjust`, `p=0` should output a black image, `p=1`, a white image, and `p=0.5` should output the input image itself without any adjustment. Output for `p` values between `0.5` to `0` should gradually change from the input to a black image. Similarly, for `p` values between `0.5` to `1`, the output should gradually from the input to a white image. For `contrastAdjust`, `p=0.5` should not alter the input image, `p=0` should output a grey image, and `p=1` should output a black and white image (would look like a binarized image). For all other intermediate values of `p`, the behaviour should gradually change from the behaviour at `p=0.5` to either `p=0` or `p=1` depending on the value of `p`.

   Test your implementation on ‘`brightness_contrast.jpg`’.

Output:


Brightness 0.5
Contrast 0.5

Original Image without any Brightness/Contrast Change


Brightness 0.53
Contrast 0.566

Optimal Looking Image After Brightness and Contrast Scaling

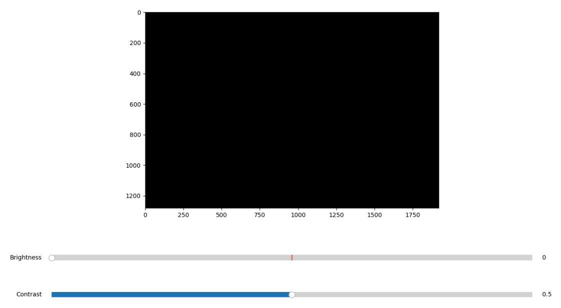

Brightness 0
Contrast 0.5

Image with Brightness Set to 0

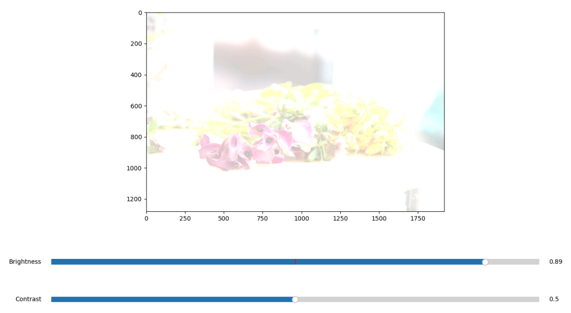

Brightness 0.89
Contrast 0.5

Image with Brightness Set to a High Value

Suvam Dey (SR No: 24290)
MTech Signal Processing, Department of ECE, IISc, Bengaluru

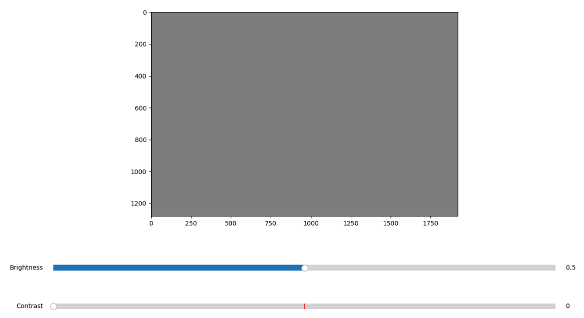


Image with Contrast Set to 0

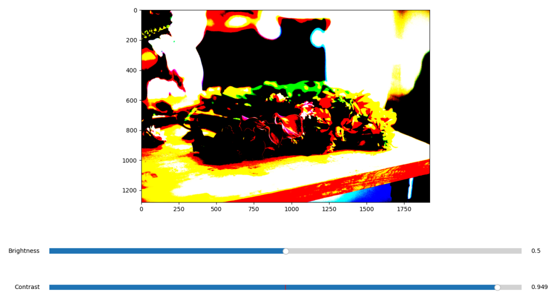


Image with Contrast Set to a High Value

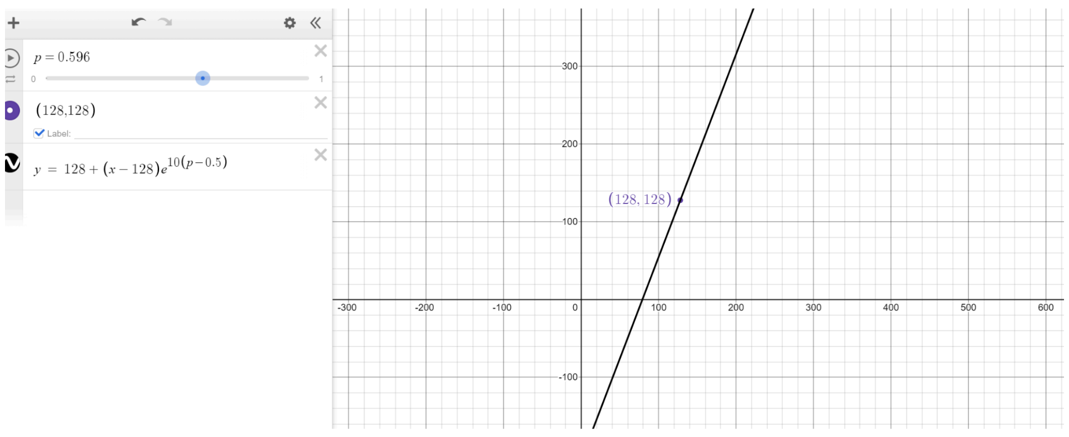
Inference:

When p = 0.5 is set in both brightnessAdjust and contrastAdjust functions, the original image is shown. Increasing p in case of brightnessAdjust causes the image brightness to increase, with the image becoming completely white when p = 1, while it becomes completely black gradually, when p value is reduced from 0.5 to 0.

Similarly, changing the value of p in contrastAdjust function towards p=1 causes the image to approach a binary thresholded image, but since it is a colour image, the thresholded image is displayed across 3 colour channels, as shown in the final picture. When p = 0, the image has no contrast, and displays a grey image across all pixels - all pixels are equalised, as shown in the second last output image.

To obtain the visually optimal image, the p values for contrast and brightness were manually set as shown in the second image.

To obtain the necessary contrast matching with the function shown, a graphing calculator was used to find which function gives a slope of +infinity when p = 1 and of 0 when p = 0, with the change happening gradually, in case of contrastAdjust, as shown below.



$p = 0.596$

$(128,128)$

$y = 128 + (x - 128)e^{10(p-0.5)}$