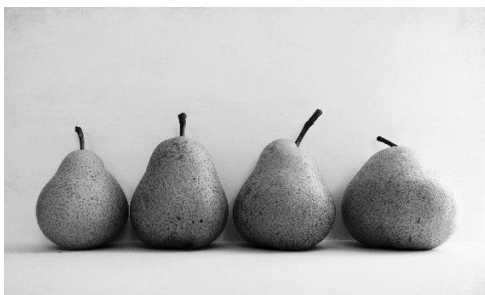


## Lab 07

### Image Processing I

#### A. Multiple Choice (20 points, 5 points each question)

- How many different colors you can have in a 16-bit RGB image?  
(a)  $256 \times 256 \times 256$  (b)  $2^{16}$   
(c)  $2^{16} \times 3$  (d)  $2^{16} \times 2^{16} \times 2^{16}$
- How many bytes are required to store a grayscale image of  $20 \times 20$  pixels having 64 different intensity levels?  
(a) 300 (b) 400  
(c) 2400 (d) 3200
- On the original gray scale image, which of the following point processing operations could have been applied to obtain the processed image?  
(a) Contrast compression (b) Image inversion  
(c) Some gray scale slicing operation (d) Extraction of the most significant bit

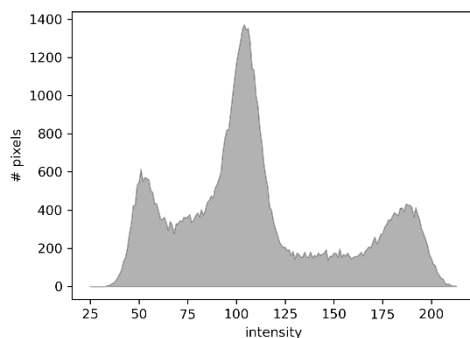


Original image

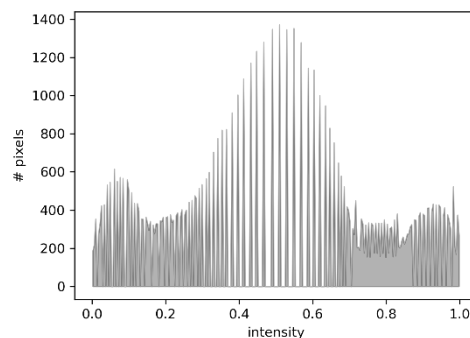


Processed image

- The figures below are histograms before and after processing a grayscale image. What is the most plausible processing applied on the original image?  
(a) Gray scale inversion (b) Binary thresholding  
(c) Histogram equalization (d) Some gray scale slicing



Histogram of the original image



Histogram of the processed image

**B. Basic Image Processing (60 points, 20 points each question)**

Note: All of the question in this part will be tested using several images, thus do NOT hard code your program to the example images. Also, you may want to make sure if your program works on different image file formats (e.g., JPG, PNG, TIFF). Find the function templates and usage in Lab07\_B.py.

1. Write a function that make the input color image gradually fade from color to grayscale as the figure below.



Original image



Processed image

2. Image matting is an important task in image and video editing. Matting refers to the process of extracting foreground object from the background in a given image. In this question, you are going to write a function that conducts image matting. You can assume that all the input images are with black (#000000) background. The background of the output image should be **transparent**.



Original image



Processed image

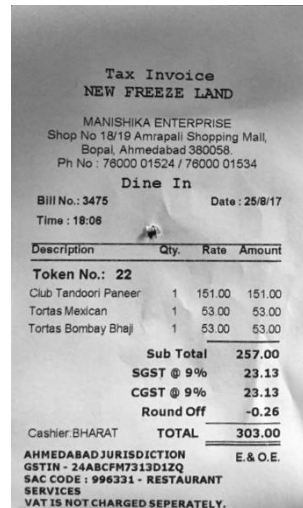
3. Resizing an image to a desired spatial dimension is a common operation when building computer vision applications based on convolutional neural networks. **Bilinear interpolation** is an intuitive algorithm for image resizing. As you might have guessed, you are going to implement your own image resizing function, `my_resize(img, height, width)`. `height` and `width` are the desired spatial dimension of the resized image. Do NOT use any of the build in functions in `skimage.transform` and try not to use `for` loop in the function. You can assume that all the input images are grayscale and square.

### C. Document Scanner (20 points)

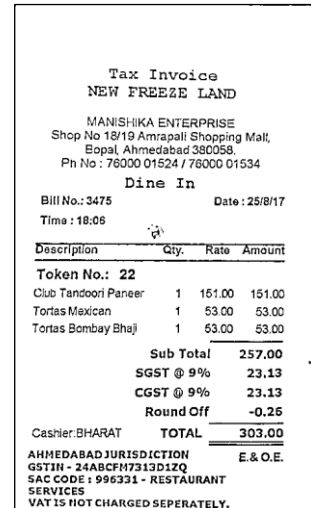
Nowadays, plenty of document scanner apps let users capture a photo of a document with their phone and convert it into a clean, rectangular PDF. In this question, you will write a program to scan *invoice.jpg*. The scanned invoice should be neatly rectified and cropped. The background of the invoice should be mostly a uniform white, with even illumination, and the foreground text should be crisp and visible with high contrast.



Original image



Intermediate image



Scanned invoice