Assignment 2
Working with Images in Spatial Domain

Please remember:

1. What you must hand in includes the assignment report (.pdf), source codes (.m) and output files (.png). Please insert each part in a different folder, and zip them all together into an archive file named according to the following template: HW2_XXXXXXXX.zip
   Where XXXXXXXX must be replaced with your student ID.
2. Your work will be evaluated mostly by the quality of your report. Don't forget to explain what you have done, and provide enough discussions which proves you have realized the subject.
3. 5 points of each homework belongs to compactness, expressiveness and neatness of your codes and report.
4. By default, we assume you implement your codes in MATLAB. If you're using Python, you have to use equivalent functions when it is asked to use specific MATLAB functions.
5. Your codes must be separated for each question, and for each part. For example, you have to create a separate .m file for part b. of question 3. Please name it like p3b.m.
6. "Keywords" will help you find useful information about the problem. They may also include some ideas for solving that problem.
7. Using built-in functions is not allowed, except for simple operations like reading, displaying, converting and saving images, or in cases it is clearly mentioned in "Allowed MATLAB Functions" section of each problem.
8. **Please upload your work in Moodle, before the end of April 13th.**
9. If there is ***any*** question, please don't hesitate to contact me through the following email address: **ali.the.special@gmail.com**
   I'd be glad to help.
10. Unfortunately, it is quite easy to detect copy-pasted or even structurally similar works, no matter being copied from another student or internet sources. Try to send me your own work, without being worried about the grade! ;)

1. Pixelation of an Image, and Then Fixing It (6 Pts.)

Keywords: *Pixelation, Image Smoothing, Image Interpolation*

Pixelation is an undesired effect in image processing, caused by displaying an image at such a large size that pixels are visible. "Pixelated" image is a very low-resolution image with visible pixels, as figure 1 shows.
Here, you are going to apply pixelation on an image, and try to see if it is possible to fix it.

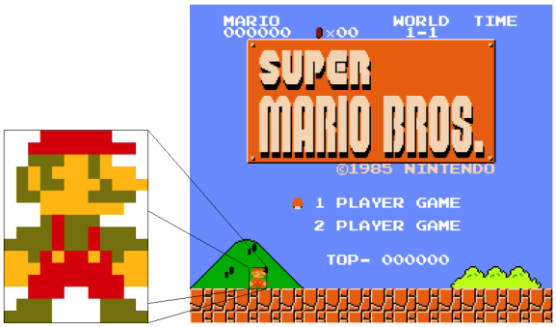

Figure 1 Images of the elements inside the old consoles games were usually pixelated

a. Implement a function to pixelate an image with a desired resolution. Your function should take an image and a size as input, and return the pixelated version of the input in the required size. Read image "modern_mario.jpg" and apply your function with an arbitrary size on it.
b. Find a way to fix a pixelated image. Read image "vintage_mario.jpg", and apply your method upon it.
Note: You are free to change input image pixel resolution (i.e. the number of pixels).

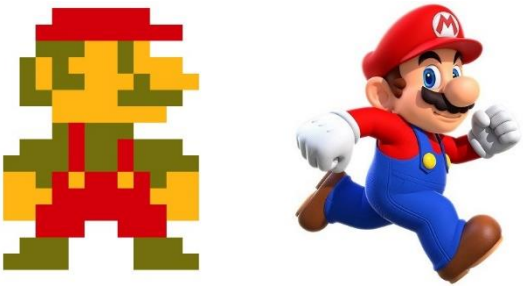

Figure 2 Vintage and modern Mario, which on them you have to pixelate (right) and fix pixelation (left)

2. Image Censorship using MATLAB! (6 Pts.)

Keywords: *Pixelization, Image Smoothing*

Sometimes the aim of applying image processing techniques is not to increase the details, but to decrease them. It is usually in the cases when some information must be removed from the images because the law requires it. The act of blurring parts of an image by displaying them at a lower resolution is called **Pixelization**.
A familiar example can be found in Street View mode of Google Maps, where vehicle license plates and faces of the pedestrians are blurred. Pixelization is primarily used for censorship.

a. Read image "trump-pence.jpg". As you can see, in this picture Donald Trump is trying to kiss his vice president, so it is inappropriate and needs to be censored! Implement a function which can apply a smoothing filter in a rectangular part of an image and with a specific kernel size. Use it on the input image in order to blur Trump's face. Find a good value for kernel size by trial and error.

Figure 3 Trump is trying to kiss Pence!

Hint 1: In MATLAB, `imtool()` could help you find Trump's face area.
Hint 2: Your function's header could be similar to this:

```
censored censor_by_smoothing(img, point1, point2, k_size)
```

Where `point1=[x1,y1]` and `point2=[x2,y2]` are the positions of the upper-left and lower-right corners of the rectangular area respectively, and `k_size` determines the size of the smoothing kernel.

b. Let's try another method of pixelization. Read image "larry.jpg". Larry Nassar (figure 2) is the former USA Gymnastics national team doctor who was accused of molesting about 250 girls, and therefore sentenced to 175 years in prison this February. In this problem, you have to censor his face in the given image. Implement a function which can reduce resolution of a rectangular part of an image with a specific size. Use it on the input image in order to blur Larry's face. Find a proper size for the area resolution.

Figure 4 Larry Nassar in the court

Hint 1: This would be similar to problem 1, except being done in a specific part of an image.
Hint 2: Your function's header could be similar to this:

```
censored censor_by_downsampling(img, point1, point2, desired_res)
```

Where `point1=[x1,y1]` and `point2=[x2,y2]` are the positions of the upper-left and lower-right corners of the rectangular area respectively, and `desired_res=[h,w]` determines the number of desired pixels in the rectangular area.

3. Applying Spatial Filtering on a Binary Image (7 Pts.)

Keywords: *Spatial Filtering, Linear Filters, Order-Statistics Filters, Kernel (Mask), Convolution*

Spatial Filtering on images is usually done using small matrices called **Kernels**. This is accomplished by applying a convolution between the kernel and the image.

In this problem, you have to find proper kernels (of any sizes) which convolving them with the input image leads to the desired outputs. Your input image in this problem is "symbols.png". Each time after finding the kernel matrix, please explain which clues helped you find your answer.

Figure 5 Input image of this problem

a. Fill empty spaces between lines
b. Remove letters in the bottom
c. Thin the letters in the bottom <u>without</u> changing the remaining items in the image
d. Change the background color to gray (intensity value 127)
e. Round the corner of the rectangular objects in the image
f. Remove the smallest three rectangles
g. Convert rectangular objects to the vertical lines

Note: Using built-in functions for convolution and filtering is unfortunately <u>not</u> allowed.
Hint 1: Choosing appropriate strategy for boundary regions is also crucial.
Hint 2: Using the below statement in MATLAB after `imshow()` function gives you a full screen display:

```
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
```

Useful Links: 1

4. Simple Blind Deconvolution using 3x3 Kernel Size Prior (7 Pts.)

Keywords: *Spatial Filtering, Linear Filters, Kernel (Mask), Derivative Filters, Sharpening Filter, Laplacian Filter*

The process of recovery of the clean image from a single or set of blurred images is called **Blind Deconvolution**. In single image blind deconvolution, usually a prior knowledge (e.g. the edges are smooth) is assumed in order to make the problem solvable.

Here, you are going to do some simple blind deconvolution, where the clean image is available as well as the blurred one. Using the clean image, and the fact that the estimated kernel size must be 3x3, try to find a kernel which produces the desired image after convolving with the main image. You are working with the image "kids.jpg". You can also find all the desired outputs in the same folder. Note that your result must be exactly the same with the desired image (You may use `isequal()` in MATLAB). You also have to explain visual clues in the desired outputs which lead you to find your answer.

Figure 6 Clean input image of dancing African kids

a. Target image is "kids_target1.jpg".
b. Target image is "kids_target2.jpg".
c. Target image is "kids_target3.jpg".
d. Target image is "kids_target4.jpg".
e. Target image is "kids_target5.jpg".
f. Target image is "kids_target6.jpg".
g. Target image is "kids_target7.jpg".

Figure 7 From left to right, expected outputs of part a. to part b.

5. Histogram Equalization and Its Applications (10 Pts.)

Keywords: *Contrast Enhancement, Histogram Equalization, Fingerprint Recognition, Face Recognition, Car Plate Recognition, Digital Radiography*

If the intensity values of pixels of an image is concentrated on a specific range, e.g. a very dark or bright image, the information may be lost in the areas where the intensity values are concentrated. The aim of **Contrast Enhancement** techniques is to optimize the contrast of an image so as all the information in the input image is very well represented.

In this problem, you are going to get familiar with **Histogram Equalization**; a classic, yet very effective method of contrast enhancement. You will also realize the importance of this method as a preprocessing step in many machine vision tasks.

a. Read image "leo.jpg", and display the histogram associated with it. Explain your observation.

b. Apply histogram equalization on it, using a built-in function (in MATLAB, `histeq()`). Display the result and corresponding histogram.
c. Implement your own function for histogram equalization. Apply it on the input image, and compare the result with part b.
d. Read image "faces.jpg", and apply histogram equalization on it. Comment on how this process could help in **Face Recognition** and **Face Detection** areas.
e. Read image "fingerprint.jpg", and apply histogram equalization on it. Comment on how this process could help in **Fingerprint Recognition** area.

Figure 8 Low contrast input for histogram equalization

Figure 9 Applying histogram equalization in different applications (from left to right): Face recognition/detection, Fingerprint recognition, Car plate recognition, Digital radiography

f. Read image "car_plate.jpg", and apply histogram equalization on it. Comment on how this process could help in **Car Plate Recognition** area.
g. Read image "chest_x-ray.jpg", and apply histogram equalization on it. Comment on how this process could help in **Digital Radiography** area.

Allowed MATLAB functions: `imhist()`, `imfilter()`.

6. Adaptive Histogram Equalization: When Histogram Equalization Fails (8 Pts.)

Keywords: *Adaptive Histogram Equalization*

Although **Histogram Equalization** shows impressive results in most cases, it won't work effectively when the image contains regions that are significantly lighter or darker than most of the image. **Adaptive Histogram Equalization**, however, solves this problem by considering only small regions of the image and using their local CDF in the process of contrast enhancement of those regions.

a. Read images "beach.jpg" and "car.jpg". Apply histogram equalization on both, and explain your observations.
b. Now apply adaptive histogram equalization on the inputs. For doing so, divide the image into small blocks of the size m x m, then histogram equalize each of these blocks as usual. Find a good value for m. Compare the results with part a.

Figure 10 Images of beach and car, which global histogram equalization might handle them differently

Allowed MATLAB functions: `imhist()`

7. Basic Gray-level Transformation Functions (8 Pts.)

Keywords: *Gray-level Transformation Functions, Linear Transformations, Negative Transformation, Logarithm Transformation, Power-law Functions, Gamma Correction*

One effective way to enhance image visual appearance is to use a transformation function which maps each value of image intensity to another value. Although it seems very simple, one can find it useful by applying **Gray-level Transformation Functions** such as **Negative**, **Logarithmic** and **Power-Law** on the input image, based on the application and the required characteristics of the output.
In this problem, you are going to work with some of these transformations and compare their results.

a. Read images "azadi_tower.jpg" and "tehran.jpg", and apply the logarithm operator upon it. Discuss whether it improves their contrast or not. You may use their histogram information as well.
b. Use the previous part inputs and do gamma corrections on them, using a proper value for gamma. Comment on the results.
c. Read image "cat_selfie.jpg" and find its complement in RGB space.
d. Read image "surprise.jpg", and make it washed-out.

Figure 11 Two different inputs for logarithm operator and gamma correction

Allowed MATLAB Functions: `imhist()`

Figure 12 Inputs of part c. and part d.

8. More Histogram Operations and Their Effects (9 Pts.)

Keywords: *Histogram Operations, Histogram Sliding (Normalization), Histogram Matching, Intensity Windowing*

Image **Histogram Operations** is an important class of point operations, which is done by manipulation of an image histogram or a region histogram. You have already practiced **Histogram Equalization**. With this problem, we are going to go one step further and get familiar with more histogram operations like **Histogram Sliding** and **Histogram Matching**.

a. Read image "hobbiton.jpg" and display histogram associated with it. Now try to slide it to the first and last 25% of the intensity levels. What are the effect of these operations?

Figure 13 You are going to brighten and darken this image by applying histogram sliding upon it

Figure 14 Left image should be histogram matched with the right one

b. Read images “aut1.jpg” and “aut2.jpg”, and display histograms associated with them. You are going to adjust the second image histogram using the first image histogram. After applying histogram matching, display the result and the corresponding histogram.

c. **Intensity Windowing** is a special case of clamp operation, which tries to stretch the image intensities to fill the full possible range. Read the image ”serious_cat.jpg”, and apply intensity windowing upon it. Set the required parameters so that the black cat and slippers distinguish clearly from the background, and display the result and the corresponding histogram.

Figure 16 Input of the part d.

d. (Additional point) Implement a function which applies histogram equalization on color images. Read image “kid_with_cat.jpg” and apply your function on it, then display the result and the corresponding histogram.

Allowed MATLAB Functions: `imhist()`

Figure 15 Black objects (cat and slippers) must be clearly separated from the background after applying intensity windowing

9. Noise Reduction by Simple Linear Operations (8 Pts.)

Keywords: *Image Denoising, Image Averaging, Image Smoothing, Signal-to-Noise Ratio (SNR), Peak Signal-to-Noise Ratio (PSNR)*

Removing unwanted noise in order to restore the original image and preserving useful information (features) in it is desirable both for human vision and machine perception, making **Image Denoising** an important pre-processing step for image analysis.

To deal with noise problem, many approaches have been proposed during the past four decades. In most of them the ultimate goal is to improve signal-to-noise ratio, i.e. SNR, or peak signal-to-noise ratio, i.e. PSNR.

In this problem, you are going to do noise reduction using simple linear operations.

a. Read images “tiny_trump-1.jpg” to “tiny_trump-20.jpg”. Find and display averages of first 2, 5, 10 and 20 images of this set. Explain your observations.

b. Compute SNR measure for “tiny_trump-1.jpg”. Then add the remaining images one by one, and each time after finding the average image, compute the corresponding SNR. Display SNR values of every 20 steps in a graph.
Note: You don’t have to display the averaged images.

Figure 17 Clean image of the Tiny Trump

c. Use Gaussian filter in order to reduce “tiny_trump-1.jpg” noises. Find the best value for sigma parameter by trial and error. Display the result, and compute the corresponding SNR.
d. Use Smoothing filter in order to reduce “tiny_trump-1.jpg” noises. Find a good value for kernel size by trial and error. Display the result, and compute the corresponding SNR.
e. After comparing their results, explain the pros and cons of each of three methods in noise reduction.

Allowed MATLAB Functions: `imgaussfilt()`, `imfilter()`.

10. Enhancing Scanned Books Pages by Thresholding (10 Pts.)

Keywords: *Image Thresholding, Global Thresholding, Local Adaptive Thresholding*

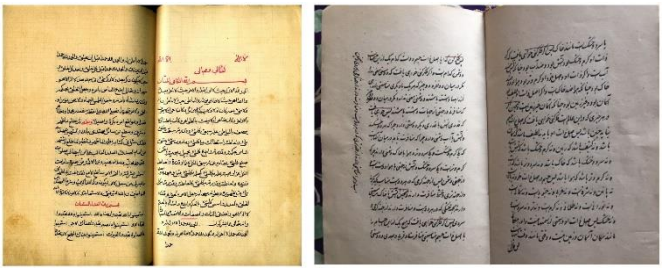

Figure 18 Uneven Illumination in the process of scanning old books pages

You may have come across old books in online services like Google Books or Gutenberg Project. In these services, first the book pages are scanned. Then, the scanned images of pages are binarized and processed through an **Optical Character Recognition**, i.e. **OCR**, engine. Although it is possible to remove the book spine in order to increase scanning quality, for old books, however, destroying the original binding of the book is often undesirable. If the process of scanning is done with book spine left intact, the curvature of the pages causes uneven illumination in the scanned images, as can be seen In Figure 18.
In this problem, you are going to deal with this issue using image thresholding. Input images are “scanned1.jpg” and “scanned2.jpg”.

a. Convert both images to grayscale, and perform global thresholding using a proper threshold value found by histograms of the grayscale images. Comment on the quality of the resulted binary images.
b. This time, perform locally adaptive thresholding on the images. Comment on the quality of the binary image and compare it with the results from part a.

11. Finding and Counting Coins using Template Matching (10 Pts.)

Keywords: *Template Matching*, *Mean Squared Error, Sum of Squared Differences (SSD), Normalized Cross-Correlation (NCC)*

Template Matching is the process of finding small part(s) of an image using another image, which is called template. Although a very simple method, template matching could be very useful in many applications like quality control, robot navigation and motion tracking. Early works in Face Recognition were also based on template matching [1].
Template matching is usually done by minimizing an objective function. The definition of this objective function is based on the chosen matching method. Here, we assume the simplest case, i.e. mean-squared error (MSE) between reference image and template image.

a. Read image "collection.jpg". Use image "collection_coin1.jpg" as the template, and find its location in the reference image. Display MSE values of image pixels as an image.
b. Do the same using "collection_coin2.jpg" as the template. Note that the search space is becoming wider this time, as the template is not aligned with its matched area in the reference image.
c. There is another coin similar to "collection_coin1.jpg" in the reference image. Can you find it by template matching?

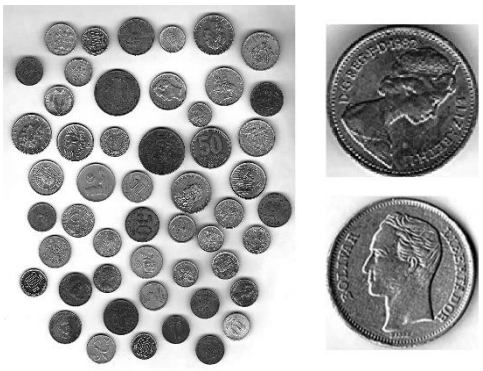

Figure 19 Collection of coins and two templates of coin images

Figure 20 Counting the number of coins Super Mario is about to achieve using coin template

d. Let's do template matching in RGB space. Read image "mario_bonus.jpg". Using image "mario_coin.jpg" as the template, find the number of coins in the reference image.
e. (Additional point) Apply template matching on part a. by considering three more matching methods.

Allowed MATLAB functions: `imrotate()`
Useful Links: 1, 2.

12. Some Explanatory Questions (6 Pts.)

Please answer the following questions as clear as possible:

a. How does image averaging (problem 9) helps reducing noise? Explain.
b. What does a second pass of histogram equalization produce? Why?
c. Is it necessary that the sum of the elements in a kernel equals to one? Explain.
d. What is the weak point of adaptive histogram equalization? Is there a way to overcome this problem?
e. Histogram equalization is a special case of histogram matching. Can you explain it?
f. The resultant image in part c. of problem 7 has a name and is used in some applications. Please mention them briefly as well as the name.

Good Luck
& Happy New Year!
Ali Abbasi