Practicum Computational Vision

Practicum 2: Image smoothing and simple geometric operations in Matlab (Part II)

In this session, we are going to illustrate the smoothing image processing toolbox in MATLAB.

- IMPORTANT: Save the solution of the exercises in separate files. Use the next nomenclature "ej_x." Ex: For exercise 2.1, generate file "ej_21.m" The Matlab code should be properly commented. The comments should include:

    - Description of Methods.
    - Results obtained and conclusions about them.
    - Observations and difficulties found during the procedure.

1.1 Creating images of 3 channels (color images)

The RGB images are formed by 3 matrices, commonly called channels.
a) Create the 3 images in gray scale as shown in Figure 1.
b) Combine the 3 obtained to construct the color image shown in Figure 1(right).
c) Save the image as 3channels.jpg.

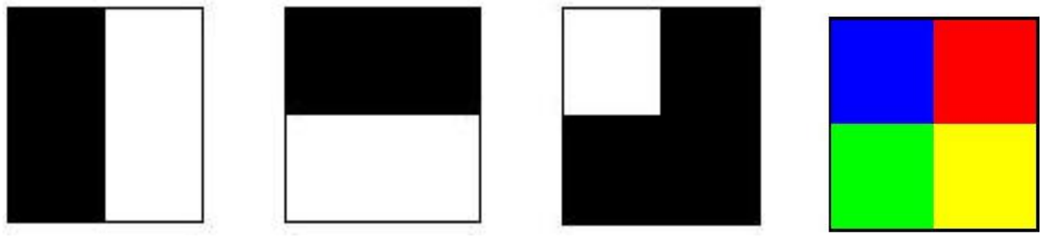


Figure 1 Constructing color images from black and white masks

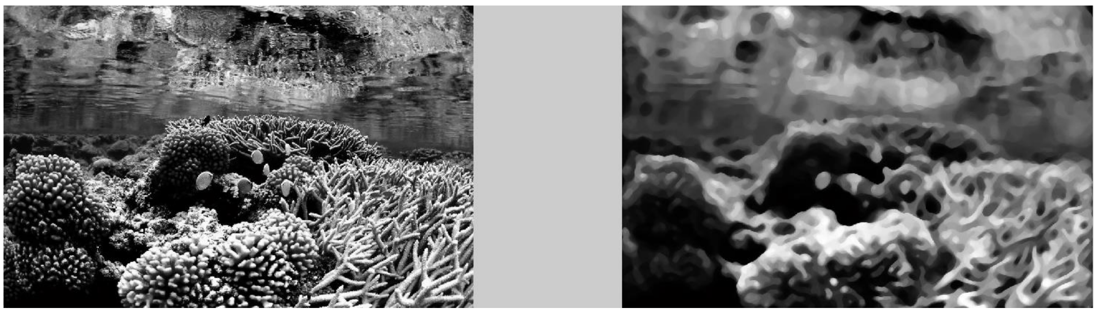
1.2 Displaying color images

a) Read the image sillas.jpg. Display different channels and explain the difference and similarities in pixel values. What would happen if we interchange the channels? What would happen if we multiply one of them by 0?

1.3 Managing different size and filters

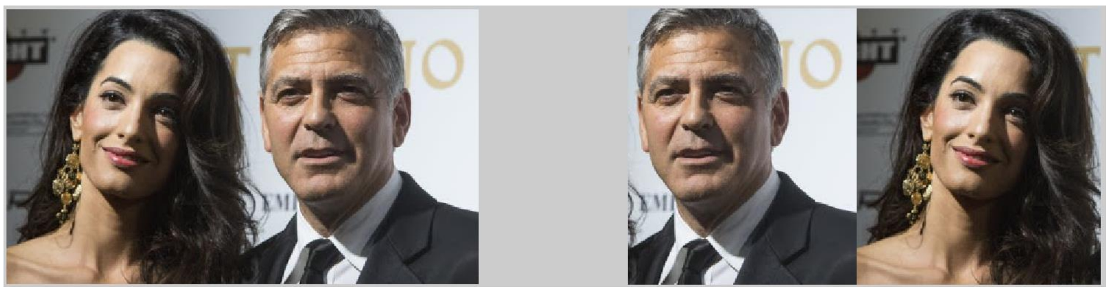
a) Read one of the images in the images.rar.
b) Show how details of the image disappear when rescale the size of the image (Help: imresize). Does the histogram change of both images (the original and the rescaled one)? Return back the smaller image to the original size. Compare to the original one.

c) As an alternative of removing image details, apply different smoothing filters (by user-defined mask as a vector e.g. [1 1 1] vs. Gaussian filter). Discuss how the size of the filter or mask affect the final outcome.
Can you apply the filter on the rgb image? What type should be the image before convolving and why?
What dimensions should be the mask? What is the difference using the following three masks: a) [1 1 1 1 1], [1;1;1;1;1] and [[1 1 1 1 1]; [1 1 1 1 1]; [1 1 1 1 1]; [1 1 1 1 1]; [1 1 1 1 1]]? Do we need to normalize the mask (divide by the sum of all its numbers)? Apply several times in order to observe effect.
d) Apply the median filter and compare the results with the previous filtering (c). Up to you, what is the optimal size of the median filter in order to smooth the image? (Help: medfilt2).
Note 1: Repeat these experiments with a couple of other images. If you have implemented all exercises as functions it should be immediate.
Note 2: You can subtract the original and smoothed images in order to illustrate the difference between them. Use subplot in order to show the original smooth and the difference image in the same figure.



1.4 Simple geometric operations

Read the image clooney.jpg and change the place of both figures so that George Clooney stands on the left. The function should have 2 parameters: the original image and a number being the column number of the cut. It should return a new image with the persons interchanged. (You can use imcrop to find the optimal cut column.)



1.5 Image binarization

The binarization BI (x, y) of an image I (x, y) from a threshold (Th) consists in turning the image into a binary image (of 0s and 1s) that will depend on whether the level of the pixel intensity of the original image is above or below a threshold.

$$B_I(x,y) = \begin{Bmatrix} 0, & \forall(x,y): I(x,y) < Th \\ 1, & \forall(x,y): I(x,y) \geq Th \end{Bmatrix}$$

..

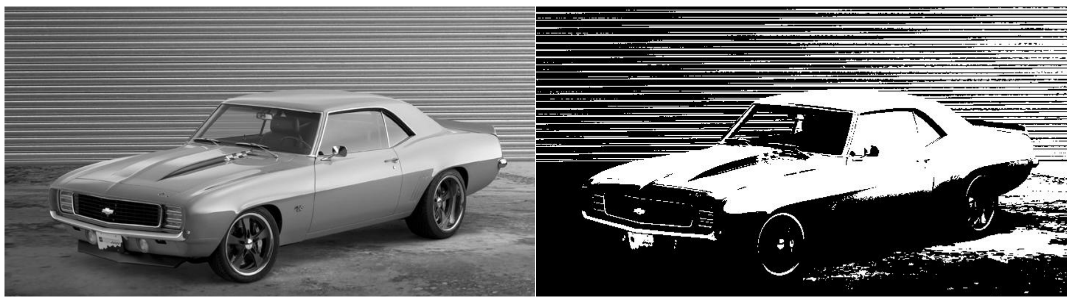

(a) (b)

Figure 2: Thresholding: b) represents the binarization of a) using Th=128

Given the image car_gray.jpg, create the function thresholdImg() which implements the following points:

a) Create the binary version of the original image by a threshold value of 20. What does it happen if we use different threshold values (30, 150, 255)? Why?

b) Visualize and save the image of threshold 150 as hand_binary.jpg.

c) What will happen if you multiply the original image by the binary image?

d) What will happen if you multiply the original image with the inverted binary image? Help: im2bw(), imtool()

1.6 Treating color images

Given the images hand.jpg (Figure 3(a)) and mapfre.jpg (Figure 3(b)), create the function fuseImg(), which implements the following points:

a) Open hand.jpg and convert it in gray scale image.

b) Perform a binarization to obtain a binary image of 2 regions: the hand (called foreground) and the rest (called background). Create the inverse binary image changing the areas of foreground and background.

c) use the binary matrices created in b) to merge the images hand and mapfre (Fig. 3(c))

e) Save the image as hand_mapfre_3C.jpg.

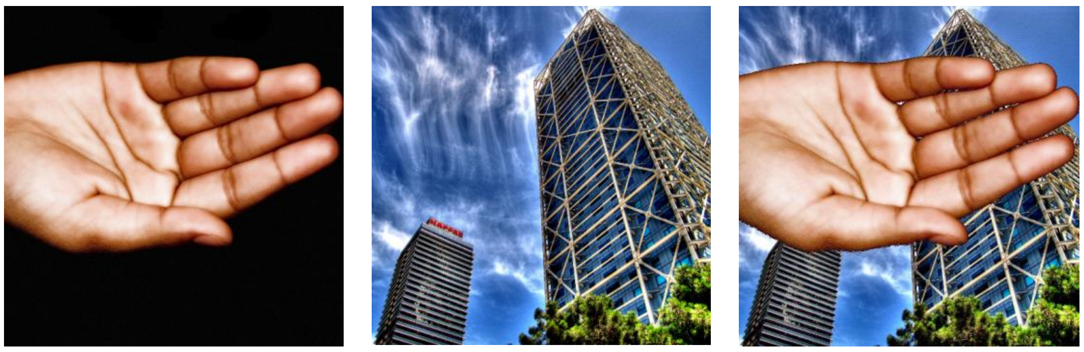

(a) (b) (c)

Figure 3 Overlapping images

Practicum submission

Deadline: Friday, 30 of October, 23:59h by Campus Virtual.

Material to submit: file “name_CV1.zip” containing: FilesXX.m of each exercise where code should be commented.