Introduction to Computer Vision

Homework exercise #2:

Image Pyramids

Due Date: Wednesday, Dec. 13, 2017
Submission: in pairs

Introduction

In this exercise you will practice working with image pyramids and get some first-hand experience with two of their applications:

- Image mosaicing: Stitching images together in a visually-appealing way.
- Image focusing: Using two or more images taken with different planes of focus in order to get a multi-focus image.

The exercise involves coding with MATLAB.

General Instructions

- You should create a script file called hw2sol_run.m that runs all the tests in this exercise, displays the results on the screen and saves them to the disk.
- The programs should be written in MATLAB. Remember that MATLAB is efficient when the code is vectorized, so try to avoid using loops.
- You will find the following MATLAB functions useful: imread, imshow, imwrite, imfilter. Matlab's help is very useful, use it!
- The program code should be readable and have a decent documentation.
- Please submit your solutions electronically to the following link: https://www.dropbox.com/request/zyACKLHqDECubssHzqen
- Your solutions should be submitted in the following procedure:
  1. Create a directory: hw2sol-<lastname1>-<lastname2> with your last names. There you should put all your code and output, together with the examples and the code supplied to you.
  2. Provide your written answers and image outputs in a PDF file. You can do this any way you want: LATEX, Word or a similar program. Please name this document: hw2sol-<lastname1>-<lastname2>.pdf
  3. When done, create an archive .zip file of the entire directory (please make sure it contains the directory node itself, not just the files), and put it in the link.

Part 1: Image pyramid generation (40 points)

In this part you will implement basic image pyramid functions. In order to do that you will need to read parts from the article **"The Laplacian Pyramid as a Compact Image Code"** by Burt and Adelson, 1983 (included in this exercise).

All of the tests will be run with depth=4 and kernel parameter a=0.6

a) Write a MATLAB function: ***h = G_Kernel(a)*** which builds the generating kernel, depending on the parameter 'a', as specified in page 533 in the article.
b) Write a MATLAB function: ***I_out = GREDUCE(I, h)*** which implements equation (1) (see page 533 in the article). Instead of using loops, your implementation should use the function *imfilter*, which performs convolution of an image with a filter.
Use this syntax: *imfilter(I,h,'replicate')*
c) Using (a) and (b), write a MATLAB function: ***G = G_Pyramid(I, a, depth)*** that receives as input an image I, the parameter 'a' of the generating kernel and the desired pyramid depth, and returns the Gaussian pyramid stored in a cell-array G such that G{i} stores level i-1 of the pyramid.
d) Using (c), write a MATLAB function: ***L = L_Pyramid(I, a, depth)*** that returns a Laplacian pyramid of I, similarly to what you did in (c).
Note: You are supplied with a function: *I_out = GEXPAND(I, h)* that performs the EXPAND operation on an image that was reduced using the filter h.
e) Write a MATLAB function: ***I_out = L_pyramid_decode(L,a)*** that takes a Laplacian pyramid L and the parameter 'a' that was used to generate it, and returns the decoded image *I_out*.
f) Test your implementation of *L_Pyramid* and *L_pyramid_decode* on the image orange.jpg in ImageMosaicing\1. Generate a Laplacian pyramid, then decode it to get the image back. Display your output on the screen in the format *dispImg = [I_before, I_after];* and include it in your submitted document.

Part 2: Image focusing (30 points)

In this part you will experiment with image focusing using pyramids. The goal is to take two pictures that were shot with different focal planes and use them to generate a multi-focus image.

Open the article **"Pyramid Methods in Image Processing"** by Burt, Adelson et al. (1984) which included in this exercise. The errata in the article was corrected for you and the corrections are marked in red.

- Note: The article addresses gray level images. When focusing RGB images, the correct way is to determine the maximum by the intensity image i.e. use rgb2gray() of the image (The final result should be in RGB).

Read the section marked in blue in page 38, and answer these questions:

a) What is the main idea that underlies this method of image focusing? Why should we expect it to work? A brief and concise explanation should suffice.

b) Write a MATLAB function: ***I_out = image_focus(I1, I2, depth)*** that takes two images I1,I2 and uses the method described above to create a multi-focus image *I_out* using a Laplacian pyramid with the given depth.

c) Test your function on the 3 image pairs in the ImageFocusing folder and display the output in the format: *dispImg = [I1, I2, I_out];*

Part 3: Image mosaicing (30 points)

In this part you will get to know a surprising application of image pyramids. It turns out that we can stitch images together in a visually-appealing way if we make the transition separately at each level of the Laplacian pyramid.

First, read about image mosaicing in pages 39-40 (marked in red) on the article **"Pyramid Methods in Image Processing"** by Burt, Adelson et al. (1984).

a) What is the main idea of using the Laplacian pyramid for performing image mosaicing? Why should this work better than using a simple smooth transition of the form: $I_{out} = (1 - m)I_1 + mI_2$ where m is a 2d mask with values between 0 and 1 (generated by smoothing a binary mask)?

b) Write a MATLAB function: ***I_out = image_mosaic(I1, I2, m, depth)*** that takes two images I1,I2 and stitches them together using the method described above. 'm' is a 2d mask, not necessarily binary (that is, may contain any value between 0 and 1). 'depth' is the depth of the pyramid.
Note: The article uses a binary mask in order to choose the value at each point in the pyramid. However, in order to avoid sharp transitions and get appealing results, we need to smooth the mask a bit at each pyramid level. You can use the command:
mSmooth = imfilter(m, ones(5)/25, 'replicate');

c) Try your implementation on the 3 image pairs and masks in the ImageMosaicing folder. For pairs 2 and 3 test also the result for the inverse mask (1-mask). Note: in total you test 5 different configurations.
Display your output on the screen in the format *[I1, I2; I_out, m]* and add it to the document, similarly to part 3.

d) In order to visualize the superiority of this method over a simple smooth transition (as mentioned in question (a)), implement image mosaicing using a smooth transition. Test it on the orange/apple example.
Note: This shouldn't take you extra work. Simply use your implementation of image_mosaic with depth=0.

Part 4: Bonus – add your own image examples (5 points)

You are encouraged to try your implemented functions (image focusing or mosaicing) on images of your choice. Be imaginative!
Add your two input images, your mask (in case of mosaicing) and your output image to the document.

Notes:

    - Make sure the size of your images is divisible by 2 to the power of pyramid depth.
    - Remember that for the output to look good, the images need to be well-aligned.
    - You do not need to create many examples. One or two is enough.
    - In the case of image focusing:
        - Make sure your images are well-aligned by displaying and flickering between them. Running this method on a misaligned image pair leads to artifacts in the output image.
- Should any problems arise, please don't hesitate to contact us.

Good Luck!