CSC 476/676: Computer Vision Prof. Xiao

Lecture 2 In-class lab exercises.

These exercises do not require submitting. But the instructor needs to see that you have competed them.

1. 1-D convolution.

One way to think about 1D convolution is to slide the kernel f over the input image. For each position of the kernel, we multiply the overlapping values of the kernel and image together, and add up the results. This sum of products will be the value of the output image at the point in the input image where the kernel is centered.

For example a 1D image is:

$$f = \boxed{10 \mid 50 \mid 60 \mid 10 \mid 20 \mid 40 \mid 30}$$

10	50	60	10	20	40	30

And our kernel is:

$$g = \boxed{1/3 \mid 1/3 \mid 1/3}$$

1/3	1/3	1/3

Let ‘s call the output image h. What is the value of h(3)? To compute this, we slide the kernel so that it is centered around f(3):

10	50	60	10	20	40	30
	1/3	1/3	1/3			

For now, we’ll assume that the value of the input image and kernel is 0 everywhere outside the boundary, so we could rewrite this as:

10	50	60	10	20	30	40
0	1/3	1/3	1/3	0	0	0

We now multiply the corresponding (lined-up) values of f and g, then add up the products. Most of these products will be 0, except for at the three non-zero entries of the kernel. So the product is:

$$\frac{1}{3}50 + \frac{1}{3}60 + \frac{1}{3}10 = \frac{50}{3} + \frac{60}{3} + \frac{10}{3} = \frac{120}{3} = 40$$

Can you finish the steps and write down the value of h?

2. Image convolution

First, you need to be fluent to use either cv2 or any other image to open an image and show it.

If you don't know how, you can go through this tutorial:

https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_image_display/py_image_display.html

We will first start to use the package scipy.ndimage to do the 2D convolution. This is more hands-on then Opencv.

https://docs.scipy.org/doc/scipy-0.16.1/reference/generated/scipy.ndimage.filters.convolve.html

You can, for example,
impot cv2
myimage= cv2.imaread('yourimage.jpg',0) # as grayscale.

1) Mean filter

from scipy import ndimage
ndimage.convolve(image,kernel, mode="constant")

You should first read in your favorite image on the disk:

Again you can use either cv2.imread or scipy.ndimage.imread() to read your image

Now, create a kernel as a small matrix. For example, you can use a 3x3 mean filter:

g1 = [1 1 1 ; 1 1 1 ; 1 1 1] / 9;

Convolve your image with g1. What happens?

2) Now try to create some new kernels. Try creating g2, a filter that multiplies the image intensities by 2, and g3, a filter that sharpens the image (you should define g3 in terms of g1 and g2). Try applying both of these to an image, and take a look at the results:

3) Try this filter:

g4 = [-1 -1 0 ; -1 3 0 ; 0 0 0];

Try applying this to an image (the otter image works better in this case). When you use cv2. imshow to view the result, the image will look mostly black—this is because the entries of this kernel sum to 0, instead of 1. To fix this, add 0.5 to the resulting image, e.g.

4) Now look at filter g1 again. What if we want to blur more? We could blur the image with g1 once, then blur the result again with g1. This will give a “twice-blurred” image. This is equivalent to the operation: $((f * g1) * g1)$. Try this out, and see what the result looks like.