Homework 2 Questions

Instructions

- 4 questions.
- Write code where appropriate.
- Feel free to include images or equations.
- Please make this document anonymous.
- **Please use only the space provided and keep the page breaks.** Please do not make new pages, nor remove pages. The document is a template to help grading.
- If you really need extra space, please use new pages at the end of the document and refer us to it in your answers.

Questions

Q1: Explicitly describe image convolution: the input, the transformation, and the output. Why is it useful for computer vision?

A1: In mathematical view, image convolution uses 2D matrix to effect the image. To achieve image convolution, we should receive an image, and the filter, usually has smaller size compared to the image. In transformation part, flipping the filter horizontally and vertically at first is necessary. Then locate the filter's first element at the pixel of the image. Do the elementwise multiplication, and sum up the result of the calculation, the output image's pixel at the same position of the center element of the filter would have that value. Do this for every pixel of the image. When filter exceeds the image, we need paddings for image to calculate the result properly. The padding could be 0, or the reflection of the image. After all the jobs are completed, the output image will come out, the image applied filter to the original image.

We can extract the important information easily with the convolution.

Q2: What is the difference between convolution and correlation? Construct a scenario which produces a different output between both operations.

Please use $imfilter$ *to experiment! Look at the 'options' parameter in MATLAB Help to learn how to switch the underlying operation from correlation to convolution.*

A2: Convolution does flipping filter horizontally and vertically before doing the elementwise operation while correlation doesn't. By doing this, convolution allows image to preserve the order of occurred changes. So the difference occurs when the filter is not symmetric. For example, using the matrix below as a kernel:

$$\begin{bmatrix} -15 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 16 \end{bmatrix}$$

Using convolution and correlation generates the images below.

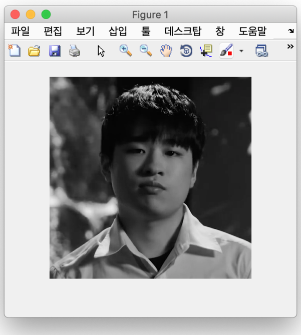

Figure 1: Original image

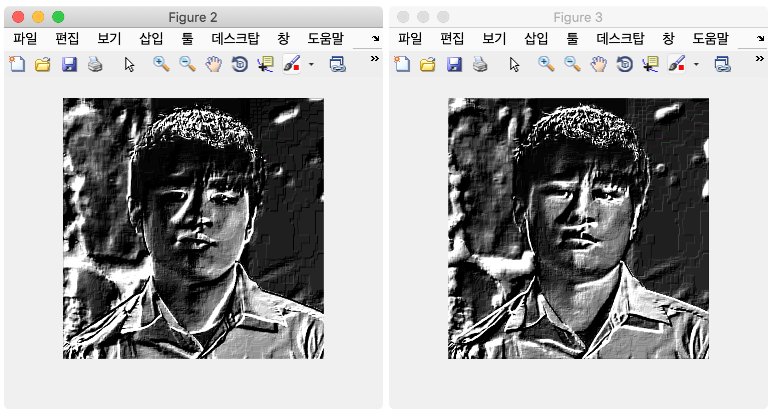
Figure 2
Figure 3
파일 편집 보기 삽입 툴 데스크탑 창 도움말
파일 편집 보기 삽입 툴 데스크탑 창 도움말

Figure 2: *Left:* Image generated with convolution *Right:* Image generated with correlation

Q3: What is the difference between a high pass filter and a low pass filter in how they are constructed, and what they do to the image? Please provide example kernels and output images.

A3: High pass filter is used for sharpening the image. For example, high pass filter uses kernels like

$$\begin{bmatrix} 0 & -1/4 & 0 \\ -1/4 & 2 & -1/4 \\ 0 & -1/4 & 0 \end{bmatrix}$$

this kernel emphasizes the difference between nearby pixels.

In opposite, low pass filter is used for blurring the image. It looks like

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

This makes the pixels' values similar to nearby pixels.

And here are the images applied two filters.

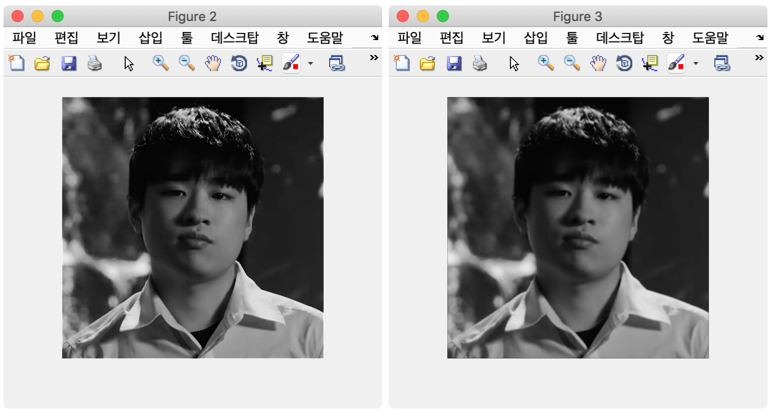
Figure 2
파일 편집 보기 삽입 툴 데스크탑 창 도움말
Figure 3
파일 편집 보기 삽입 툴 데스크탑 창 도움말

Figure 3: *Left:* High pass filter *Right:* Low pass filter

Q4: How does computation time vary with filter sizes from 3×3 to 15×15 (for all odd and square sizes), and with image sizes from 0.25 MPix to 8 MPix (choose your own intervals)? Measure both using *imfilter* to produce a matrix of values. Use the *imresize* function to vary the size of an image. Use an appropriate charting function to plot your matrix of results, such as *scatter3* or *surf*.

Do the results match your expectation given the number of multiply and add operations in convolution?

See RISDance.jpg in the attached file.

A4: If kernel size is [2m x 2n], it is expected to spend 4x time than the [m x n] kernel because it needs 4x multiplication operations. Also, if pixel number increases from N to 2N, it would take 2x time because we need to take 2x addition operations. However, here is the result:

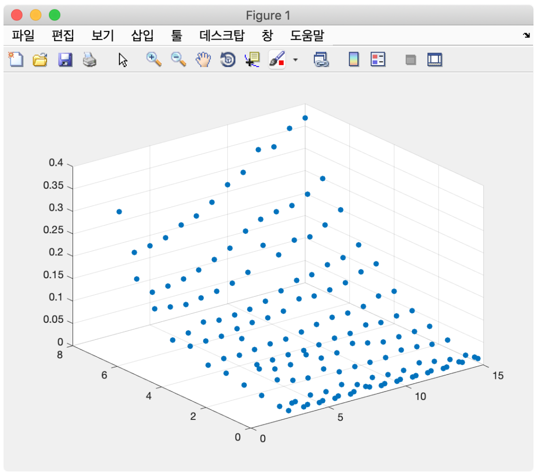

Figure 4: Result of elapsed time

x-axis means kernel size, y-axis means pixel numbers, z-axis means elapsed time. It seems that kernel size is linear with elapsed time, and (pixel numbers)^2 is linear with elapsed time. It is diffrent with expectation.