Homework 2 Questions

Instructions

- 4 questions.
- Write code where appropriate.
- Feel free to include images or equations.
- **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: The input is original image and filter for convolution. In transformation, the local neighborhood pixel values of image get multiplied with corresponding filter value and summed up to get output of the specific pixel. By image convolution, we can enhance image like denoising, resizing, increasing contrast, extract infromation from images like texture, edges, distinctive points, or detect patterns which makes it useful for computer vision.

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

A2: Both convolution and correlation multiplies neighboring pixel values with corresponding filter value. The difference comes from how to match the corresponding filter value. In correlation, the filter values get multiplied by the pixel values by the same order so that correlation: $(f \star g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t+\tau)d\tau$ However, in convolution, the filter values get multiplied in flipped order so that correlation: $(f*g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$. Correlation and convolution are identical when the filter is symmetric. So to produce a different output between both operations, the filter should be assymetric. for example, in 2D correlation and 2D convolution, filter $F = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ will produce a different output.

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: Applying low pass filter to the image smoothes it, making it blurry. It can be constructed by averaging filter or Gaussian filter like $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$. The output image becomes blurry. High pass filter is used to get the edges of an image, as high frequency get preserved. It can be constructed by accenuating the pixel value with local averages. For example, filter $F = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ can be used as kernel of the high pass filter. The output image shows the edges of image.

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 $cv2.filter2D()$ to produce a matrix of values. Use the $cv2.resize()$ function to vary the size of an image. Use an appropriate 3D charting function to plot your matrix of results, such as $plot\_surface()$ or $contour3D$.

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

See RISDance.jpg in the attached file.

A4: The plot is as below. The computation time grows linearly as image size increases, as I expected. I expected the computation time for filter size to be $N \times N$ is $N \log N$. However, it did now grow as fast when the filter size is small as seen in the plot, and the computation time even decreased when filter size is big. More precisely, the computation time when image size is 8 MPix and filter size is 15×15 was less then when the filter size is 13×13 in some runs, with other conditions the same.