Problem 1: Mechanics of Convolutions [60 points]

Suppose we have two filters:

$$\text{Filter A: } \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \text{Filter B: } \begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

(a) (10 points) In class, we described a simple and intuitive version of an x-gradient filter: $[-1\ \ 1]$. When applied, this filter computes the finite difference gradient in the x-direction, essentially solving for $\partial f/\partial x \approx \Delta f/\Delta x$, where $\Delta x = 1$ and pixels are one unit distance from their neighbors. Filter A, by contrast, is used to compute what is known as the central difference x-gradient. Although it cannot be normalized in the usual way, since its values sum to zero, it is usually multiplied by a coefficient of 1/2. Why?

(b) (10 points) Normalize B. What effect will this normalized filter have when applied to an image?

(c) (10 points) Compute $A * B$, using A and B from the **original** problem statement, i.e., **without** using the scale factors described in (a) and (b). You can treat B as the filter kernel and assume that A is zero outside of its support. You do not need to show your work. [Aside: convolution is commutative ($A * B = B * A$), so you would get the same answer by using A as the filter kernel. But, you would have to remember to “flip” the kernel to get $\tilde{A}[i, j] = A[-i, -j]$. We've asked you instead to use B as the filter kernel, but since B is symmetric, i.e., $\tilde{B}[i, j] = B[-i, -j] = B[i, j]$, you don't need to worry about flipping.]

(d) (10 points) Compute $A * B$, now using A and B after scaling them according to (a) and (b).

(e) (20 points) If we apply the result of (c) or (d) to an image f, we are computing $(A*B)*f$. Convolution is associative, so we could get the same results as computing $A*(B*f)$. In other words, we're filtering the image with B, and then filtering the result with A.

i. Why would it be desirable to apply B before computing the gradient (as opposed to not applying B at all)?

ii. Why might applying B be better than applying a filter B' that is filled with a full 3×3 set of positive coefficients (e.g., changing the 0's to 1's in the B filter given above), rather than just a single column of coefficients?

Problem 2: Filter Design [40 points]

In this problem, you will consider several convolution filtering operations and their behaviors. You do not need to worry about flipping filters before sliding them across images; i.e., assume filters are pre-flipped. In addition, assume that the y-axis points up, the x-axis points to the right, and the lower left corner of the image is at (0,0). For each sub-problem, justify your answer.

(a) (20 points) The image you're editing is too dark and noisy, and you decide you need to blur the image a little with a 3×3 filter to reduce noise, and amplify the overall brightness of the image by a factor of 3. Suggest a single convolution filter that performs this task when applied to the image. (Technically, pixel values could go out of range, i.e., brighter than 255 in one or more color channels at a pixel; assume that any needed clamping will be taken care of later, after filtering).

(b) (20 points) Suppose now you wanted to create the effect of averaging an image with a vertically shifted copy of itself, where the amount of the shift is 4 pixels in the y-direction. More, specifically, you will take the original image and shift it down by two pixels. Then you will take the original image and shift it up by two pixels. Then you will average these two shifted images together. Suggest a single convolution filter that will accomplish this.