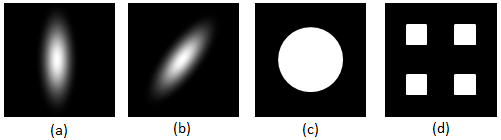
**Written Assignment 1**

1. Run GaussianBlurImage and SeparableGaussianBlurImage with sigma = 2, 4, 8 on “Seattle.jpg”. How many seconds does it take to run each function? How long do you think it would take to run each with sigma = 32?

For sigmas 2, 4, and 8, my computer takes approximately 1.3, 3.6, and 10.5 seconds. Internally, a square kernel is created based on sigma, so I'd expect there to be an n2 relationship here. Dividing 1.3 by 22 yields 0.325, 3.6 by 42 yields 0.225, and 10.5 by 82 yields 0.164. The results from the smaller times are less likely to be reliable, given the small time window, but the larger time of 10.5 is more likely to be useful. As such, I'd expect there to be a 0.16-0.18 result dividing the time by 322, which means the approximate time would be between 163.84 and 184.32 seconds. Just for fun, I ran it with sigma 32, and it took 177 seconds, which is in my expected range!

All times were taken with the executable compiled in Release mode.

2. Which of the following filters are separable, i.e. can be computed from a combination of 1D horizontal and vertical filters? Why?



Filters (a) and (d) are separable, while the other two are not.

Filter (a) is separable, because it was simply compressed about its vertical axis, but otherwise represents a standard Gaussian kernel. To achieve this, the horizontal filter would simply need to be made to converge towards zero faster as it approached the edges of the filter matrix.

Filter (b) is not separable, because it was compressed about the diagonal axis. Horizontal/vertical filters cannot correctly represent this type of mutation.

Filter (c) is not separable, because it is a circular shape with a strong edge. Circular shapes are okay (for instance, Gaussian kernels), and strong edges are fine (square filter), but not both at the same time. If they could both presumably exist at the same time, then the two filters would some how need to work together to decide whether the resulting pixel be either solid black, or solid white, but this "collaboration" is not possible.

Filter (d) is separable, because it has strong, straight edges. Given the following horizontal filter [ 0, 1, 0, 1, 0 ], it's easy to see that this would produce two vertical bars and three black bars. With a similar vertical filter, when combined, they will only intersect in the four squares that are present above.

2. What is the best amount of blur to apply when downsampling Moire.gif by 8x (pressing “Half Size” 3 times)? Does downsampling “Seattle.jpg” require the same amount of blur?

Cutting Moire.gif in half three times produces some pretty strong aliasing patterns without any blurring. If I apply a Gaussian filter with sigma=6, then the aliasing is almost completely eliminated. Applying Gaussian with sigma=8 seems to give the best result, while still allowing some of the prior details of the image to be maintained, like the hollow center and grey middle "rings".

Considering Seattle.jpg, the aliasing is not nearly as evident, because Moire.gif had regular patterns present that were very susceptible to aliasing. Seattle.jpg does not contain patterns nearly as strong as Moire.gif's, and as a result, the downsampled image looks better. However, some sharp edges are still visible (space needle looks jagged, buildings have very sharp edges), so blurring prior to downsampling helps here too, although a sigma lower than 8 can be used; I would consider sigmas as small as 4 to be acceptable for Seattle.jpg.

3. What is the best bilateral input values (sigmaS and sigmaI) for removing the jpg artifacts in “Seattle.jpg” without blurring the image’s details?

*(skipped, question is optional if BilateralImage is not implemented)*

4. Can you find an edge in “TightRope.png” that is visible to the human eye, but does not have a strong response from the Sobel edge detector?

Sobel seems to catch almost all edges in this photo. However, near the very bottom of the photo (and just to the right of center) is a small oval filled with a different shade of green trees, and Sobel does not strongly detect this. It simply shows the typical edge patterns that the trees across the image present. These trees are visibly different, but Sobel does not highlight this, likely because the difference in greens is rather subtle, compared to many of the other edges in the picture.

5. If you rotate the image 20 times by 2 degrees, does it produce the same result as rotating the image by 40 degrees? If not, why?

No. The first time an image is rotated by two degrees, a small amount of data is lost, because it has moved off screen. The second time, another small amount of data is lost, but perhaps more importantly, a small amount of the data that was lost in the first rotation should reappear on the screen, had it not been lost. This is the data along the original edges of the image, and because it is unable to scroll back into view, the edge of the image appears to start to curve by the time the 20th two-degree rotation is completed.

6. If you apply blur before applying FindPeaksImage you can remove many noisy edges. What is the best amount of blur to apply to Gogh.png to find the “cleanest” edges? Does using BilateralImage to blur the image before applying FindPeaksImage produce better edges? In addition to answering these questions, please turn in your best peak edge image called “GoghEdge.png”.

I found that applying a Gaussian blur with sigma=4, and then looking for peaks with threshold=30, yielded the best results.

*(didn't implement BilateralImage, so skipping that response)*