Project 1: Convolutions Fourier Resizing

Siddharth Patel[1]

I. P 1.1.1 Some Simple Filters

A. What effect do each of the filters have on the provided image? Describe them (in one or two sentences each). Name the filter if we discussed it in class.

According to the lecture notes provided by Dr. Stein, I can clearly conclude that F$_a$ is a Box/Mean filter. Considering that the box filter smooths the image, it is possibly the simplest operation.

The second filter, F$_b$, shifts the pixels to the right and then to the left while summing it up and taking the average with the original image causing a horizontal blur in the center of the image.

The F$_c$ filter is an inverted version of the original image. The difference is that rather than taking the sum of the pixels, we are finding the difference of the pixels and averaging it with the original image. Therefore, we are computing the first derivative of the original image.

Finally, F$_d$ filter shifts the pixels towards top-left and bottom-right followed by summation and average.

B. Which filters are separable? For the separable filters, write their components parts.

Because F$_a$ is a box filter, it is separable. The components of the box filter can be written as:

$$(1/9)\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix} = (1/3)\begin{bmatrix}1&1&1\end{bmatrix} * (1/3)\begin{bmatrix}1\\1\\1\end{bmatrix}$$

F$_b$ can be considered a separable filter as well. You can break down the F$_b$ function the following way:

$$(1/3)\begin{bmatrix}0&0&0\\1&1&1\\0&0&0\end{bmatrix} = (1/3)\begin{bmatrix}0\\1\\0\end{bmatrix} \text{ and } \begin{bmatrix}0&0&0\end{bmatrix}.$$

Once again, F$_c$ is a separable filter. The components of the separated F$_c$ filter can be mentioned as followed:

$$(1/6)\begin{bmatrix}-1&0&1\\-1&0&1\\-1&0&1\end{bmatrix} = (1/6)\begin{bmatrix}-1\\-1\\-1\end{bmatrix} \text{ and } \begin{bmatrix}-1&0&1\end{bmatrix}$$

Unfortunately, F$_d$ is not a separable filter because it does not equal to 1.

II. P 1.1.2 Image Derivatives

A. Using a finite-difference method, derive a filter for the image Laplacian and include the definition of your Laplacian filter in your write-up.

As discussed in class, a laplacian filter is an edge detector used to compute the second derivatives of an image, measuring the rate at which the first derivatives change.
Here is the 3x3 kernel matrix that corresponds to applying the laplacian operation to an image: $\begin{bmatrix}0&1&0\\1&-4&1\\0&1&0\end{bmatrix}$. The above matrix has been derived from $\nabla^2 I = \frac{d^2 I}{dx^2} + \frac{d^2 I}{dy^2}$ which can further be expanded as *f(x + 1) + f(x - 1) - 2f(x) + f(y + 1) + f(y - 1) -2f(y).*

III. P 1.1.3 Gaussian Filtering

A. What would happen if the filter width were too small compared to σ? You may answer in words, though a figure would also be acceptable.

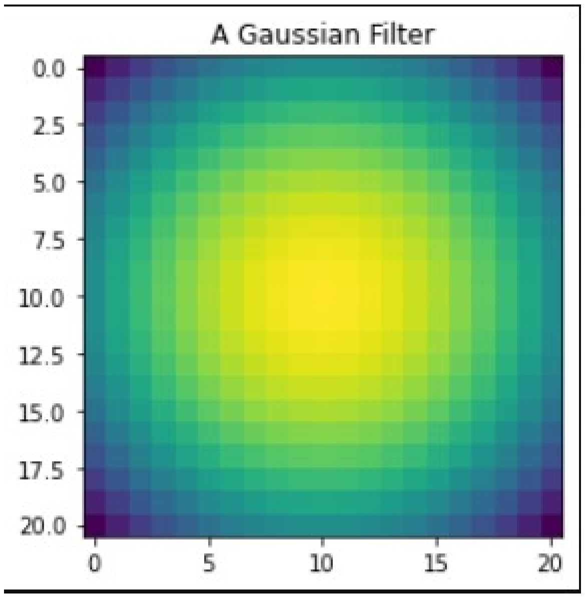

Fig. 1. An Image of Gaussian

If the filter width is too small compared to sigma, your image will be zoomed in to the center of the filtered image, as you can see in the attached image as well as in the slides. Eventually if your width is too small and your sigma is too large, the image will look pixelated as I have shown in the above attachment. Moreover, there comes a point when you reach a sigma that will not result in further changes to the image.

B. What is the σ of this filter?

After searching for an approximate σ value, I was able to get σ = 0.85, approximately, using the get_gaussian_filter. As a result, my kernel matrix could be found by using the Gaussian filter formula:

$$g(x,y) = exp(-\frac{x^2+y^2}{2\sigma^2}) \quad (1)$$

Let's solve for σ by computing for x and y as 1 and 1. g(1,1) = 4/16 or 0.25. If you now solve for σ your answer would be $\sqrt{\frac{-1}{\ln 0.25}}$

IV. P 1.2.2 Upsampling with Fourier Transforms

A. Using the ideas discussed in class (the Nyquist Limit and the Wagon Wheel Effect) explain (in words) why the four corners of the frequency-space image are where most of the intensity is located. Hint: for a discrete Fourier Transform, there are no explicitly "negative frequencies". How are frequencies above the Nyquist Limit related to the "negative frequencies" we would expect to see when taking a continuous Fourier Transform?

According to the slides, edges are high frequency regions along with noise in an image. Before I begin my explanation, note that a computer visualizes a plot/ sin wave as grid points with x and y coordinates. Because edges are blurred or carry noises due to the regions that meet at that location, it can be concluded that edges carry variations in intensity. We want to use this high-frequency regions more precisely and Fourier transforms give us a language for doing that.

How does the wagon wheel and Nyquist limit play its role in it? In order to generate a sampling rate, we need to be able to find and mark the sampling rate with dots. Wagon wheel theorem explains that without the dots, it is hard to guess the direction of the sampling rate. However, without poor sampling rate, you will end up with aliasing problems meaning you will mistake your signal for another signal. To avoid aliasing, we assume our sampling rate to be twice the max frequency. This will ensure a point at the top and bottom (where the high frequencies are located) of each cycle.

B. In 1-2 sentences, explain why the FFT upsampled image appears to have oscillating patterns? Hint: think about the higher frequency components that the upsampled image does not have. What frequency components are required (and missing) to make a sharp edge?

Edges correspond to higher frequency components in a sampling rate. Referring back to fourier transform functions, as you add more and more sin waves to a function, your edges will start to sharpen, essentially forming vertical lines/bar chart on a 2d graph. We have oscillating pattern because we are building on to the default sin wave or the sampling rate through the application of multiple sin waves. This will sharpen the edges.

C. In 3-6 sentences, describe how the frequency-space representations of the different upsampled images are different from one another. How is the FFT upsampled image particular different from the others? Is this behavior expected?

In upsampled image, consider that you have a small image, you will increase the row and column of the image. However, what ends up happening is as you are increasing the size of the image, you also have pixel gaps in the image. To restore the pixel gaps, we use different types of interpolation methods by implementing additional pixels. In the nearest neighbor frequency, your image will initial be represented as spikes in a frequency-space representation. After applying the NN interpolation, those spikes will be widen to fill the pixel gaps however, it will generate a graph that will look like bar chart. What you want is a graph that looks like sine waves to as it can be said as a smooth graph without any sharp edges. As you move towards linear interpolation and bicubic interpolation, you will proceed to get closer and closer to the ideal sine wave. It looks like the fft upsampling image has further been interpolated (one step after cubic interpolation). Indeed this behavior is expected if you are further interpolating an image past the bicubic interpolation.