1 Image Processing and Convolution

1.1 Preparation: Understanding Convolution

Think of convolution as a way to find patterns in an image by looking through a small window (kernel). As we slide this window across the image, we:

1. Look at a small neighborhood of pixels
2. Multiply each pixel by the corresponding value in our pattern-matching window (kernel)
3. Sum up these products to get a single number
4. Move the window and repeat

For example, if we want to detect vertical edges, we might use this kernel:

$$K = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix}$$

Let's see how it works on a small image region (3x3 pixels):

$$\begin{bmatrix} 10 & 80 & 10 \\ 10 & 80 & 10 \\ 10 & 80 & 10 \end{bmatrix}$$

We multiply each pixel by the corresponding kernel value and sum: $(10 \times -1 + 80 \times 1 + 10 \times 0) \times 3 = 210$. The high positive value (210) indicates a strong vertical edge was detected. Here are some common kernel patterns:

Vertical edge detection:

$$\begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix} \quad \text{Looks for } \xrightarrow{\text{dark to bright}}$$

Horizontal edge detection:

$$\begin{bmatrix} -1 & -1 & -1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{Looks for } \downarrow \text{ dark to bright}$$

Blur/Smoothing:

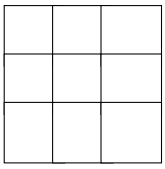
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{Averages all neighboring pixels}$$

1.2 Image Processing and Convolution

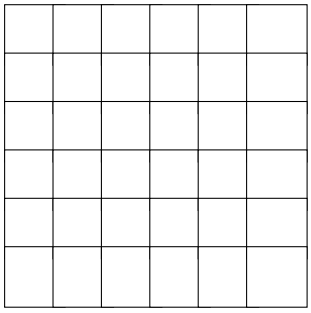
Consider a 6x6 grayscale image showing a diagonal line pattern:

80	10	10	10	10	10
10	80	10	10	10	10
10	10	80	10	10	10
10	10	10	80	10	10
10	10	10	10	80	10
10	10	10	10	10	80

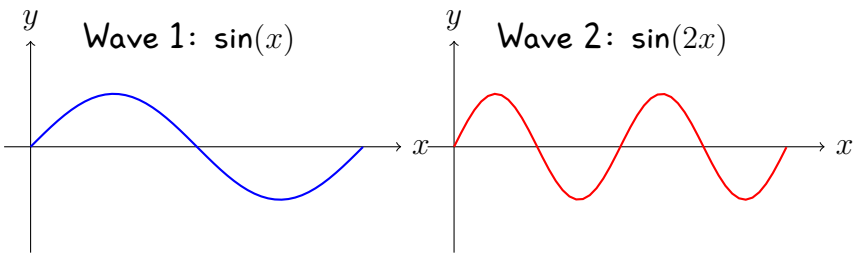
1. If we want to detect diagonal edges, which kernel of size 3x3 would be most appropriate? The kernel should have the values between -1 and 1.



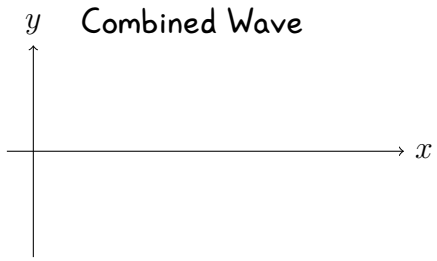
2. Apply your kernel to compute the convoluted image. No need to calculate the value of each pixel exactly but show your estimate by shading the pixels. For the boundary pixels, leave them blank since the kernel exceeds the boundary of the image.



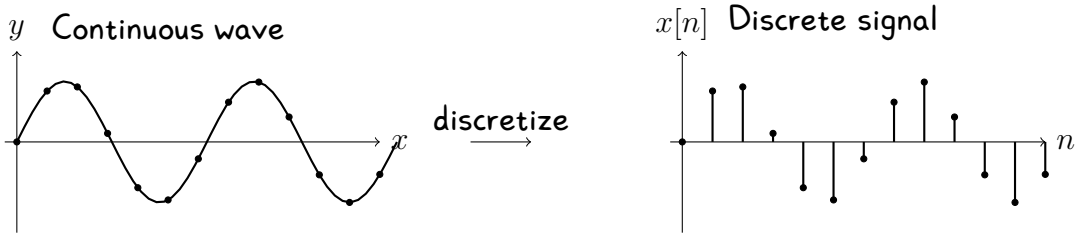
3. Now, let's learn how JPEG compression works. Consider this waves:



We combine these two waves by weighting the first wave by 1.5 and the second wave by 0.5. Combined Wave $= 1.5 \cdot \sin(x) + 0.5 \cdot \sin(2x)$. Draw the combined waves.



4. The Fourier transform is a reverse operation: it decomposes, not combines, waves into basic waves. The waves are continuous functions. But we can discretize them for computation as follows:



This results in a vector of values $[10, 80, 10, 80, 10, 80, 10, 80]$. Now, let's create a discretized mixed wave Z from X and Y as follows

$$Z = X + Y = [10, 90, 30, 90, 10, 70, -10, 70, 10] \tag{1}$$

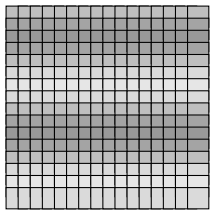
where

$$X = [10, 80, 10, 80, 10, 80, 10, 80, 10], \quad Y = [0, 10, 20, 10, 0, -10, -20, -10, 0] \tag{2}$$

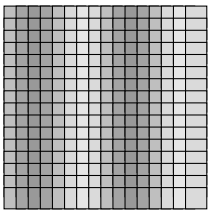
(a) If we apply a kernel $K = [-1, 1, -1]$ to this signal, what will be the resulting signal? What kind of frequencies will this kernel emphasize?

(c) If we apply a kernel $K = [1, 1, 1]$ to this signal, what will be the resulting signal? What kind of frequencies will this kernel emphasize?

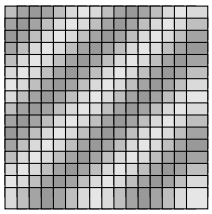
5. Just as 1D signals can be decomposed into sine waves, 2D images can be decomposed into 2D waves as follows. The Fourier transform can be applied to 2D images to decompose them into a sum of 2D waves.



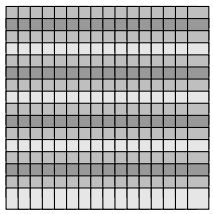
Horizontal (Low freq.)



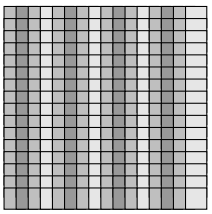
Vertical (Low freq.)



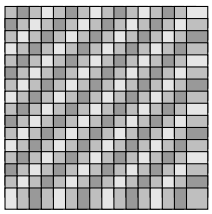
Diagonal (Low freq.)



Horizontal (High freq.)



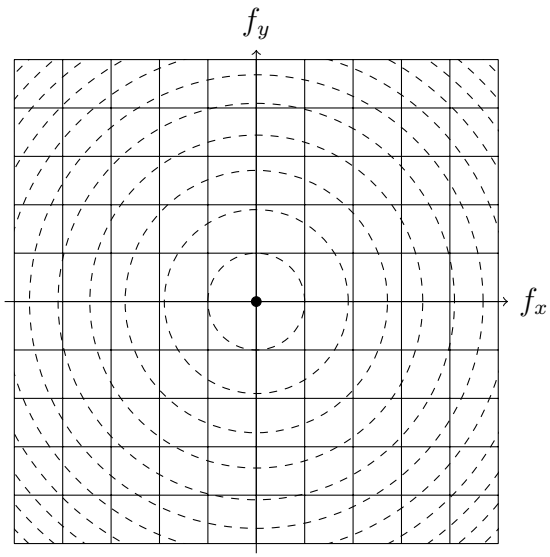
Vertical (High freq.)



Diagonal (High freq.)

Now, consider this checkerboard pattern on the left. Mark where you expect the highest magnitudes in the Fourier transform grid. The dashed circles represent the basis 2D waves in the Fourier domain.

80	10	80	10	80	10
10	80	10	80	10	80
80	10	80	10	80	10
10	80	10	80	10	80
80	10	80	10	80	10
10	80	10	80	10	80



Fourier Transform Grid

6. The image can be mapped to the Fourier transform grid (called frequency domain). We can also map it back to the original image domain (called spatial domain). Thus, we can manipulate the image in the frequency domain to remove some waves from the original image. If we want to keep only the low-frequency components of the checkerboard pattern, what regions of the Fourier transform grid should we set to zero?