Computer Vision CS-GY 6643 - Homework 1

Anushk Pandey, ap7151@nyu.edu

1 Connectivity: Connected Component Labeling

Jordan's Theorem: Every simple closed curve in a plane separates the plane into two connected nonempty sets: an interior region and an exterior region.

1.1 Applying d4 connectivity

d4 connectivity is defined as the connectivity of a pixel to the neighbors on its vertical and horizontal axes (Akin to a rook in chess).

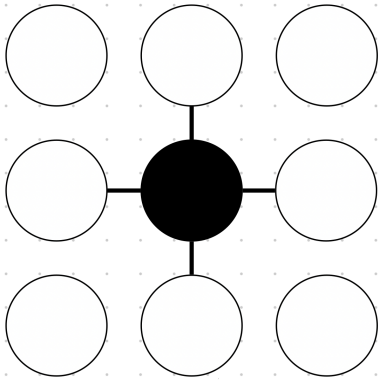

Figure 1: An example of d_4 connectivity

On computing the d4 connectivity of the given image, we get the following regions:

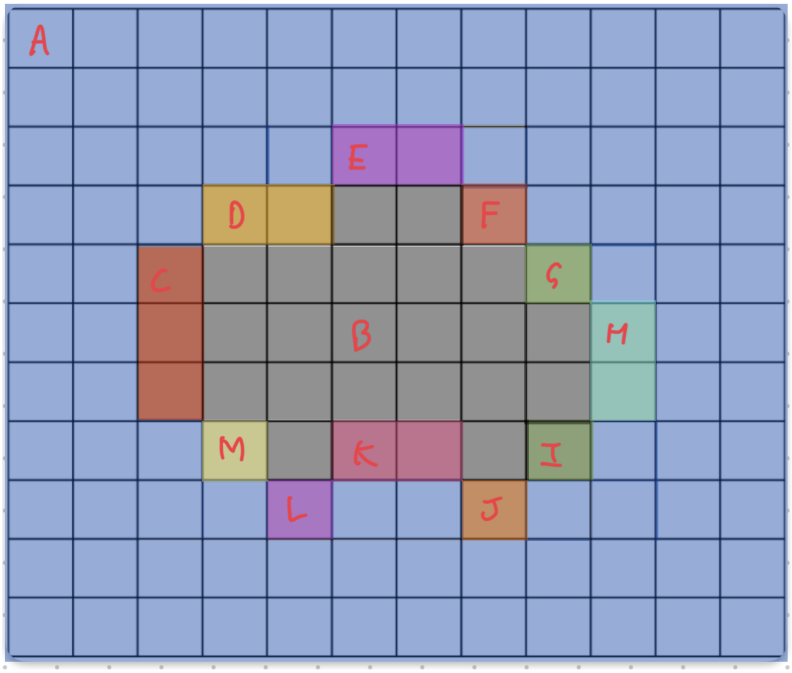

Figure 2: d_4 connectivity applied on the given image

- There are a total of 13 4-connected structures
  - Two white structures (interior and the exterior)

- Eleven gray structures

From Figure we can see that the longest gray structure has an area of 3 pixels

This approach does not follow Jordan's curve theorem as 4-connectivity of the pixels does not make a simple closed curve.

1.2 Applying d8 connectivity

d8 connectivity is defined as the connectivity of a pixel to the neighbors on its vertical, horizontal and diagonal axes (Akin to a queen in chess).

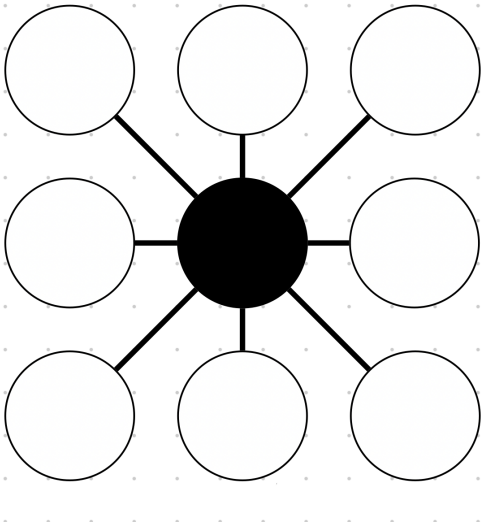

Figure 3: An example of d_8 connectivity

On computing the d8 connectivity of the given image, we get the following regions

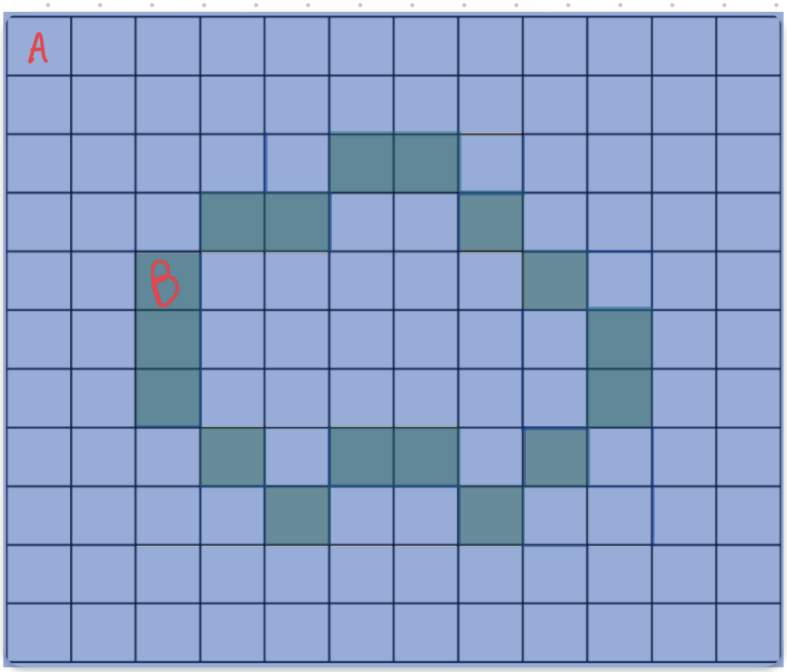

Figure 4: d_8 connectivity applied on the given image

- There are a total of 3 8-connected structures
  - One white structures

- One gray structure (curve)

From Figure we can see that the longest gray structure has an area of 17 pixels.

No, this approach does not follow Jordan's curve theorem as d8 connectivity can make the gray pixels into a simple closed curve, but the white pixels cannot be divided into an interior and an exterior.

1.3 Applying 6-C connectivity

6-C connectivity is defined as the connectivity of a pixel to the neighbors on its vertical, horizontal and **only one** diagonal axes.

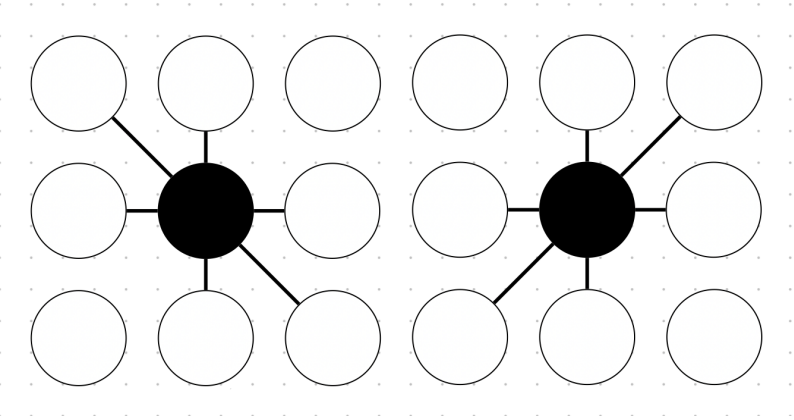

Figure 5: Examples of 6-*c* connectivity

On computing the 6-C connectivity of the given image, we get the following regions

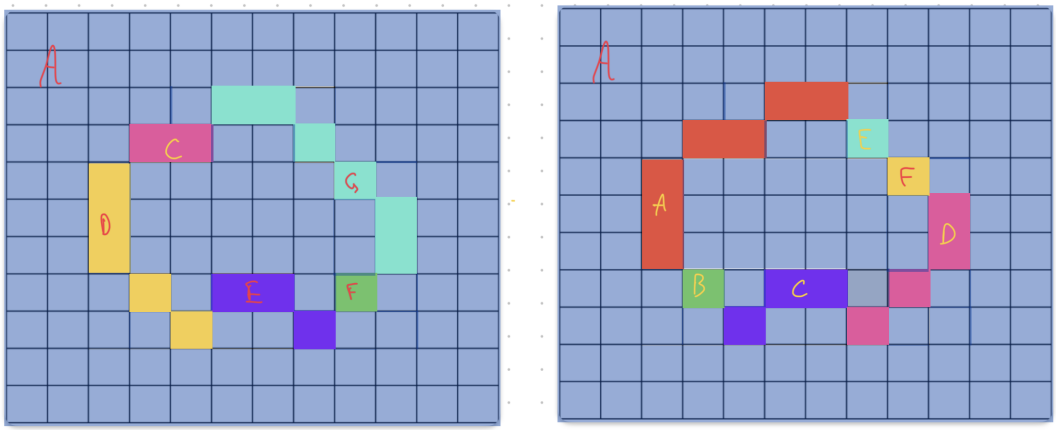

Figure 6: 6C Connectivity applied on the given image

- On using the first definition of 6-C connectivity, we get a total of 7 structures
  - One white structure
  - Five gray structures
- On using the alternative definition of 6-C connectivity, we get a total of 7 structures
  - One white structures
  - Six gray structures

From Figure we can see that the longest gray structure has an area of 6 pixels in the first case and 7 pixels in the second case.
This approach will not follow Jordan's curve theorem in either case as 6-C connectedness of the pixels does not make a simple closed curve in either case.

2 1 Dimensional Filters

2.1 Box Filter Application

The box filter of size 3 on a signal can be defined as:

$$y[i] = \frac{1}{3}\sum_{k=-1}^{1} x[i+k]$$

where $y[i]$ is the transformed signal at time i and $x[i]$ is the input signal at time i.
Using this formula, we get:

$y[1] = \frac{1}{3}\sum_{k=-1}^{1} x[1+k] = \frac{1}{3}(0+1+1) = \frac{2}{3} \Rightarrow y[1] = 0.67$

$y[2] = \frac{1}{3}\sum_{k=-1}^{1} x[2+k] = \frac{1}{3}(1+1+1) = \frac{3}{3} \Rightarrow y[2] = 1$

$y[3] = \frac{1}{3}\sum_{k=-1}^{1} x[3+k] = \frac{1}{3}(1+1+5) = \frac{7}{3} \Rightarrow y[3] = 2.34$

$y[4] = \frac{1}{3}\sum_{k=-1}^{1} x[4+k] = \frac{1}{3}(1+5+1) = \frac{7}{3} \Rightarrow y[4] = 2.34$

$y[5] = \frac{1}{3}\sum_{k=-1}^{1} x[5+k] = \frac{1}{3}(5+1+1) = \frac{7}{3} \Rightarrow y[5] = 2.34$

$y[6] = \frac{1}{3}\sum_{k=-1}^{1} x[6+k] = \frac{1}{3}(1+1+1) = \frac{3}{3} \Rightarrow y[6] = 1$

$y[7] = \frac{1}{3}\sum_{k=-1}^{1} x[7+k] = \frac{1}{3}(1+1+5) = \frac{7}{3} \Rightarrow y[7] = 2.34$

$y[8] = \frac{1}{3}\sum_{k=-1}^{1} x[8+k] = \frac{1}{3}(1+5+5) = \frac{11}{3} \Rightarrow y[8] = 3.67$

$y[9] = \frac{1}{3}\sum_{k=-1}^{1} x[9+k] = \frac{1}{3}(5+5+5) = \frac{15}{3} \Rightarrow y[9] = 5$

$y[10] = \frac{1}{3}\sum_{k=-1}^{1} x[10+k] = \frac{1}{3}(5+5+5) = \frac{15}{3} \Rightarrow y[10] = 5$

$y[11] = \frac{1}{3}\sum_{k=-1}^{1} x[11+k] = \frac{1}{3}(5+5+5) = \frac{15}{3} \Rightarrow y[11] = 5$

$y[12] = \frac{1}{3}\sum_{k=-1}^{1} x[12+k] = \frac{1}{3}(5+5+4) = \frac{14}{3} \Rightarrow y[12] = 4.67$

$y[13] = \frac{1}{3}\sum_{k=-1}^{1} x[13+k] = \frac{1}{3}(5+4+3) = \frac{12}{3} \Rightarrow y[13] = 4$

$y[14] = \frac{1}{3}\sum_{k=-1}^{1} x[14+k] = \frac{1}{3}(4+3+2) = \frac{9}{3} \Rightarrow y[14] = 3$

$y[15] = \frac{1}{3}\sum_{k=-1}^{1} x[15+k] = \frac{1}{3}(3+2+1) = \frac{6}{3} \Rightarrow y[15] = 2$

$y[16] = \frac{1}{3}\sum_{k=-1}^{1} x[16+k] = \frac{1}{3}(2+1+1) = \frac{4}{3} \Rightarrow y[16] = 1.34$

$y[17] = \frac{1}{3}\sum_{k=-1}^{1} x[17+k] = \frac{1}{3}(1+1+0) = \frac{2}{3} \Rightarrow y[17] = 0.67$

Therefore, the attenuated signal becomes:

0	0.67	1	2.34	2.34	2.34	1	2.34	3.67	5	5	5	4.67	4	3	2	1.34	0.67	0

2.2 Attenuation

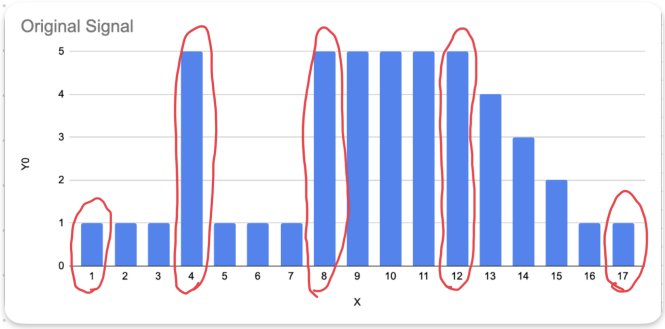

Figure 7: Original signal plotted with the attenuation in the signal encircled in red

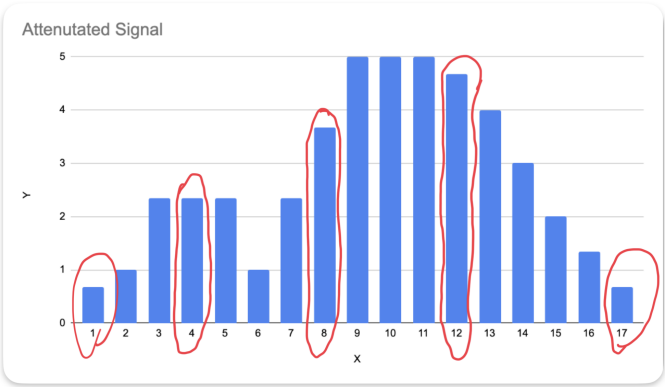

Figure 8: (Transformed signal plotted with the attenuation in the signal encircled in red

From Figures 7 and 8, we can see that the signal has been attenuated at times $i = 1, 4, 8, 12, 17$

2.2.1 At $i = 1$ and $i = 17$

Due to zero-padding along the edges of the signal, the signal value of 1 at times $i = 1$ and $i = 17$ has been attenuated to 0.67

2.2.2 At $i = 4$ and $i = 8$

The frequency of signal $x[i]$ suddenly jumps to 5 at $i = 4$ and $i = 8$. Due to the smoothening effect of the box filter, the filtered value of the signal gets attenuated.

2.2.3 At $i = 12$

The frequency of the signal x[i] starts reducing at $i = 12$ from $x[12] = 5$ to $x[16] = 1$, The box filter smoothens the frequency at $x[12]$ using the values around it ($x[11]$ and $x[13]$) attenuating the signal.

2.3 Linearity of Box Filters

The box filter is defined as:

$$T(x)[n] = \frac{1}{2k+1} \sum_{i=-k}^{k} x[n+i] \tag{1}$$

Using this definition, putting $x = ax_1 + bx_2$, the equation becomes:

$$T(ax_1 + bx_2)[n] = \frac{1}{2k+1}\sum_{i=-k}^{k}(ax_1 + bx_2)[n+i]$$

$$\Rightarrow T(ax_1 + bx_2)[n] = \frac{1}{2k+1}\sum_{i=-k}^{k}(ax_1[n+i] + bx_2[n+i])$$

$$\Rightarrow T(ax_1 + bx_2)[n] = \frac{1}{2k+1}(\sum_{i=-k}^{k} ax_1[n+i] + \sum_{i=-k}^{k} bx_2[n+i])$$

$$\Rightarrow T(ax_1 + bx_2)[n] = \frac{1}{2k+1}(\sum_{i=-k}^{k} ax_1[n+i] + \sum_{i=-k}^{k} bx_2[n+i])$$

$$\Rightarrow T(ax_1 + bx_2)[n] = \frac{1}{2k+1}\sum_{i=-k}^{k} ax_1[n+i] + \frac{1}{2k+1}\sum_{i=-k}^{k} bx_2[n+i]$$

$$\Rightarrow T(ax_1 + bx_2)[n] = \frac{1}{2k+1}a\sum_{i=-k}^{k} x_1[n+i] + \frac{1}{2k+1}b\sum_{i=-k}^{k} x_2[n+i]$$

$$\Rightarrow T(ax_1 + bx_2)[n] = aT(x_1)[n] + bT(x_2)[n]$$

Hence, we can say that the average box filter is indeed a **linear filter.**

2.4 Shift Invariance of global mean subtraction filter

A filter $y[t] = T(x[t])$ on signal $x[t]$ is defined as Shift Invariant if $y[t-\tau] = T(x[t-\tau])$ where τ is a temporal shift (change in time).

We define a transformation $y[t] = V(x[t])$ where $V(x[t]) = x[t] - \mu$, where μ is defined as the global average mean of the signal. ($\mu = \int_0^T x[t]dt$).

For a finite signal the value of μ will be constant.

To show that $y[t-\tau] = V(x[t-\tau])$:

Solving LHS:

$$y[t] = x[t] - \mu$$

For $t = t - \tau$ (Shifting the transformed signal by τ)

$$y[t-\tau] = x[t-\tau] - \mu \tag{2}$$

Solving RHS:

$$V(x[t]) = x[t] - \mu$$

For $t = t - \tau$ (Transforming the shifted signal)

$$V(x[t-\tau]) = x[t-\tau] - \mu \tag{3}$$

From Equations 2 and 3 we can say that $y[t-\tau] = V(x[t-\tau])$, which proves the linearity of the given filter $V(x[t]) = x[t] - \mu$.

2.5 Linearity of Median filter

A convolution is defined as a linear filter which is shift invariant (**??**). A linear filter $g(I)$ is a filter which satisfies the property

$$g(\alpha I_1 + \beta I_2) = \alpha g(I_1) + \beta g(I_2)$$

To prove the median filter is not a linear filter, we assume a median filter of box size 5, two constants $\alpha = 3$ and $\beta = 2$, and two 1D signals as follows

$$I_1 = [1, 4, 6, 10, 13]$$

$$I_2 = [7, 3, 21, 47, 23]$$

Computing $g(\alpha I_1 + \beta I_2)$:

$$\begin{aligned} g(\alpha I_1 + \beta I_2) &= g(3 * [1, 4, 6, 10, 13] + 2 * [7, 3, 21, 47, 23]) \\ g(\alpha I_1 + \beta I_2) &= g([3, 12, 18, 30, 39] + [14, 6, 42, 94, 46]) \\ g(\alpha I_1 + \beta I_2) &= g([17, 18, 60, 124, 85]) \\ g(\alpha I_1 + \beta I_2) &= 60 \end{aligned} \tag{4}$$

Computing $\alpha g(I_1) + \beta g(I_2)$:

$$\begin{aligned} \alpha g(I_1) + \beta g(I_2) &= 3 * g([1, 4, 6, 10, 13]) + 2 * g([14, 6, 42, 94, 46]) \\ \alpha g(I_1) + \beta g(I_2) &= 3 * 6 + 2 * g([14, 6, 42, 94, 46]) \\ \alpha g(I_1) + \beta g(I_2) &= 3 * 6 + 2 * 42 \\ \alpha g(I_1) + \beta g(I_2) &= 18 + 84 \\ \Rightarrow \alpha g(I_1) + \beta g(I_2) &= 102 \end{aligned} \tag{5}$$

From equations 4 and 5, we can see that $60 \neq 102$, which violates the linearity property.
$\therefore$ The median filter cannot be called convolution, as it violates the property of linearity.

3 2D Filters and Fourier Transforms

3.1 Sobel Filter Complexity

A horizontal edge sobel filter can be defined as follows:

$$\frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Its decomposition can be defined as follows:

$$\frac{1}{2}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Assuming a $n \times n$ image with zero padding (making the size of the image $n + 2 \times n + 2$, α as a constant computation cost of non-zero multiplications, and β as the constant computation cost of non-zero addition, the number of operations required to apply the non-decomposed filter to the entire image will be as

following (Assuming multiplication by 0,1 and -1 as taking no effective computation time)
Assuming a following pixel

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

The Convolution becomes:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \otimes \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\Rightarrow \frac{1}{8}\begin{bmatrix} -1.a + -2.b + -1.c \\ 0.d + 0.e + 0.f \\ 1.g + 2.h + 1.i \end{bmatrix}$$

$$\Rightarrow \frac{1}{8}\begin{bmatrix} -1.a + -2.b + -1.c \\ 0.d + 0.e + 0.f \\ 1.g + 2.h + 1.i \end{bmatrix}$$

$$\Rightarrow \frac{1}{8}\begin{bmatrix} -a + -2b + -c + 0 + g + 2h + i \end{bmatrix}$$

This operation takes 3 multiplication (multiplication by 2 and normalization) steps and 4 addition steps per pixel. Therefore, for $n + 2 \times n + 2$ pixels, the computation time defined as

$$C_{non\_seperable} = (3\alpha + 4\beta)n^2 \tag{6}$$

Now, for the linearly decomposed filter, passing the same pixel through the filter and calculating the computation cost.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} a.1 + 0.d - 1.g & b.1 + 0.e - 1.h & c.1 + 0.f - 1.i \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} a - g & b - h & c - i \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} a - g & b - h & c - i \end{bmatrix} \otimes \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \times \frac{1}{2}$$

$$\Rightarrow \frac{1}{2}\begin{bmatrix} a - g + 2b - 2h + c + i \end{bmatrix}$$

This operation takes 3 multiplication steps (multiplication by 2 and normalization) and 8 addition steps per pixel. Therefore, for $n + 2 \times n + 2$ pixels, the computation cost for a linearly separable filter is defined as

$$C_{separable} = (2\alpha + 8\beta)n^2 \tag{7}$$

For the usage of linearly separable filter, the inequality $C_{non\_seperable} > C_{separable}$ must hold. From Equations (6) and (7):

$$C_{non\_seperable} > C_{separable}$$

$$(3\alpha + 4\beta)n^2 > (2\alpha + 8\beta)n^2$$

$$(\alpha - 4\beta)n^2 > 0$$

As $\alpha - 4\beta$ is a constant, assuming it as C

$$Cn^2 > 0$$

$$n^2 > 0$$

As $n^2 > 0$ is true $\forall n \in \mathbb{N}$, we can say that it is feasible to use the linearly separable filter for any size of image. Therefore the minimum image size where the separable filter becomes feasible is 1×1.

3.2 Fast Fourier Transform

The Discrete Fourier Transform for each point in a frequency domain k, l is defined as

$$F[k,l] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-j2\pi(\frac{k}{M}m + \frac{l}{N}n)}$$

$$\Rightarrow F[k,l] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-2j\pi.m\frac{k}{M}} . e^{-2j\pi.n\frac{l}{N}}$$

$$\Rightarrow F[k,l] = \frac{1}{M} \sum_{m=0}^{M-1} (\frac{1}{N} \sum_{n=0}^{N-1} f[m,n]\ e^{-2j\pi.n\frac{l}{N}})\ e^{-2j\pi.m\frac{k}{M}}$$

The term:

$$\frac{1}{N} \sum_{n=0}^{N-1} f[m,n]\ e^{-2j\pi.n\frac{l}{N}}$$

in the above equation is defined as a Discrete Fourier Transform in one dimension. The Fast Fourier Transform algorithm of a 1-dimensional signal is defined as:

Algorithm 1 1D Fast Fourier Transform

```
function FFT([a_0, ...a_N − 1, ω, N)
    if N = 0 then return [a_0]
    F_even ← FFT([a_0, a_2, ..., a_{N−2}], ω^2, N/2)
    F_odd ← FFT([a_1, a_3, ..., a_{N−1}], ω^2, N/2)
    F ← new vector of length N
    x ← 1
    for j = 0 to N − 1 do
        F[j] ← F_even[j mod (N/2)] + x.F_odd[j mod (N/2)]
        x ← x · ω
    return F
```

From Algorithm (1), we can say that the FFT Algorithm calls itself recursively on half the input size $(N/2)$ and performs N operations in combining the result.
The Master Theorem is a formula which can be used to define the runtime of recursive algorithms, and is defined as:

$$T(n) = aT(\frac{n}{b}) + g(n)\ \forall\ a \geq 1,\ b > 1 \tag{8}$$

From algorithm 1 and equation (8), we can say that the Fast Fourier Transform can be represented in the form of the master theorem when $a = 2$ (two sub problems) and $b = 2$ (each subproblem is a size of $n/2$). Since the for-loop in FFT takes $O(n)$ time, the asymptotic bound of $T(n)$ becomes

$$g(n) = \Theta(n^{\log_b a)}$$

$$g(n) = \Theta(n^{\log_2 2)}$$

In this case, the value of $T(n)$ becomes $\Theta(n^{\log_b a} \log n)$. And (since $log_2 2 = 1$), we can say that the run time of a one-dimensional Fourier transform is $\Theta(n \log n)$

Since a two-dimensional fourier transform is a one dimensional fourier transform transformed in the frequency range of the other dimension's signal. Hence we defined the inner transform with the frequency range of N and the outer transform with the frequency range of M, the overall run time is multiplied by a factor of M.
Therefore the final runtime of applying a Fast Fourier Transform to a two dimensional signal becomes:

$$\Theta(MN \log N) \tag{9}$$

4 References

3blue1brown (n.d.). *Fourier Transform: A visual introduction*. URL: https://www.youtube.com/watch?v=spUNpyF58BY.

Chat with Gemini About Linearity of Box Filters (n.d.). URL: https://g.co/gemini/share/0de345f8e8d8.

Computer Science, D. S. C. S. of (n.d.). *15-451/651: Lecture 24, Design and Analysis of Algorithms*. URL: https://www.cs.cmu.edu/~15451-s23/lectures/lec24-fft.pdf#page=5.40.

Computer Vision, F. P. of (n.d.). *Segmenting Binary Images*. URL: youtube.com/watch?feature=shared&t=315&v=2ckNxEwF5YU.

ImageProcessingPlace (n.d.). *Countour Tracing*. URL: https://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/connectivity.html.

Szeliski, R. (2022). *Computer Vision: Algorithms and Applications*. Springer.

University, J. S. S. (n.d.). *CS 161: Solving Recurrences*. URL: https://web.stanford.edu/class/archive/cs/cs161/cs161.1168/lecture3.pdf.