Anirudh Rao
BE21B004

DA5401 – Data Analytics Lab

Assignment 1

Real world system

- The first step was to create a line drawing of an object.
- I drew a simple diagram of a **bacterium** in my notebook and clicked a picture to convert it into a .jpg file.

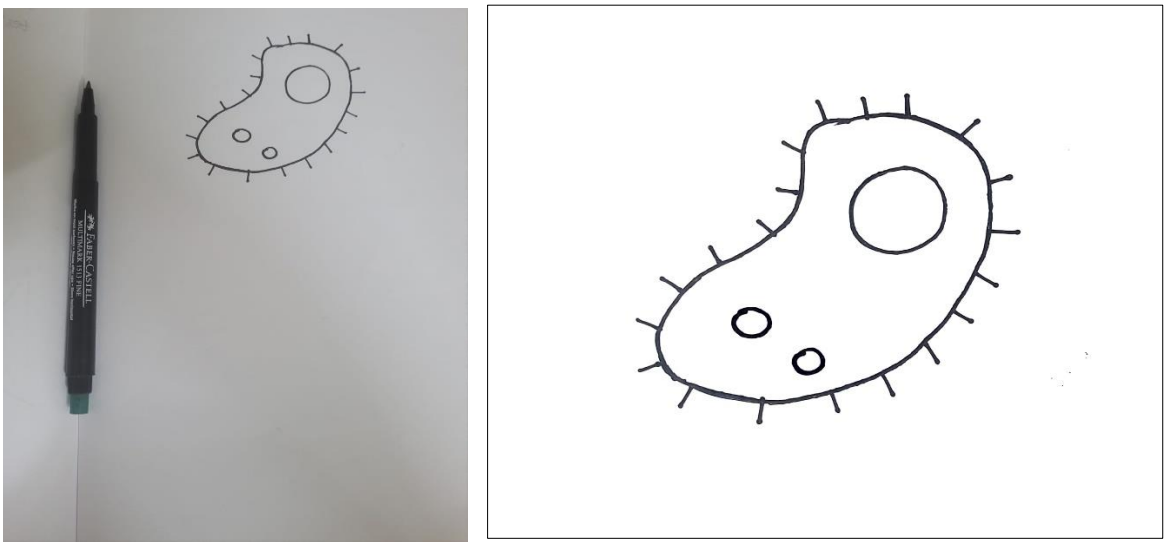

Figure 1: The original drawing and the uncropped .jpg file created from it

- For mathematical ease, the image was cropped into a square using Microsoft PowerPoint.
- The final dimensions of the image were 567 x 567 pixels.

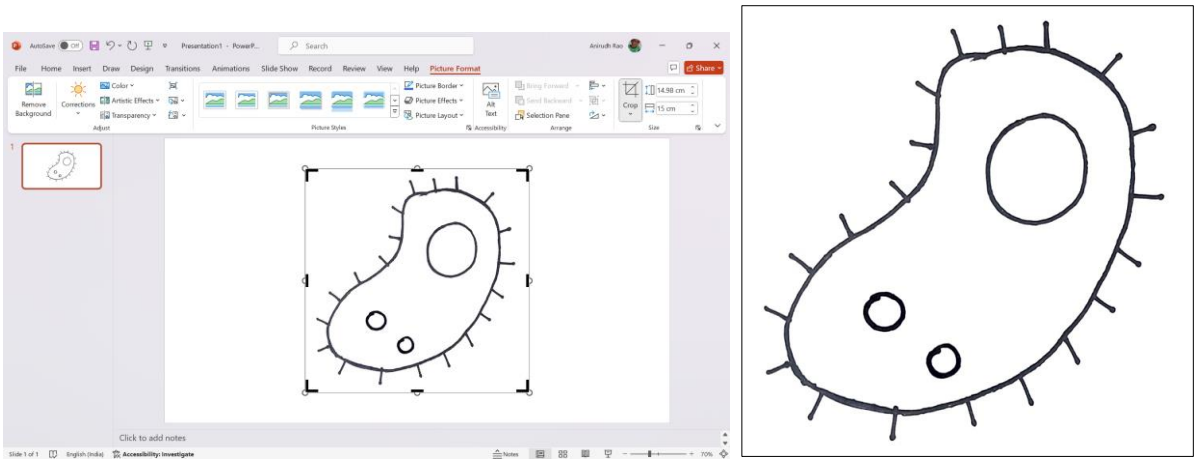

Figure 2: Cropping the image on PowerPoint and the final cropped image

Data acquisition

- The (x, y) coordinates of the points that make up the lines / curves in the image were obtained using https://wpd.starrydata2.org/.
- The cropped .jpg file was uploaded and the plot type was selected as '2D (X-Y) Plot".
- Four points were chosen to align the axes. These were $x_1 = 0, x_2 = 1000, y_1 = 0, y_2 = 1000$.
- Next, automatic extraction was performed after setting the foreground colour to black and averaging window as 3 pixels.
- The coordinates were obtained under 'View Data' and downloaded as a .csv file.

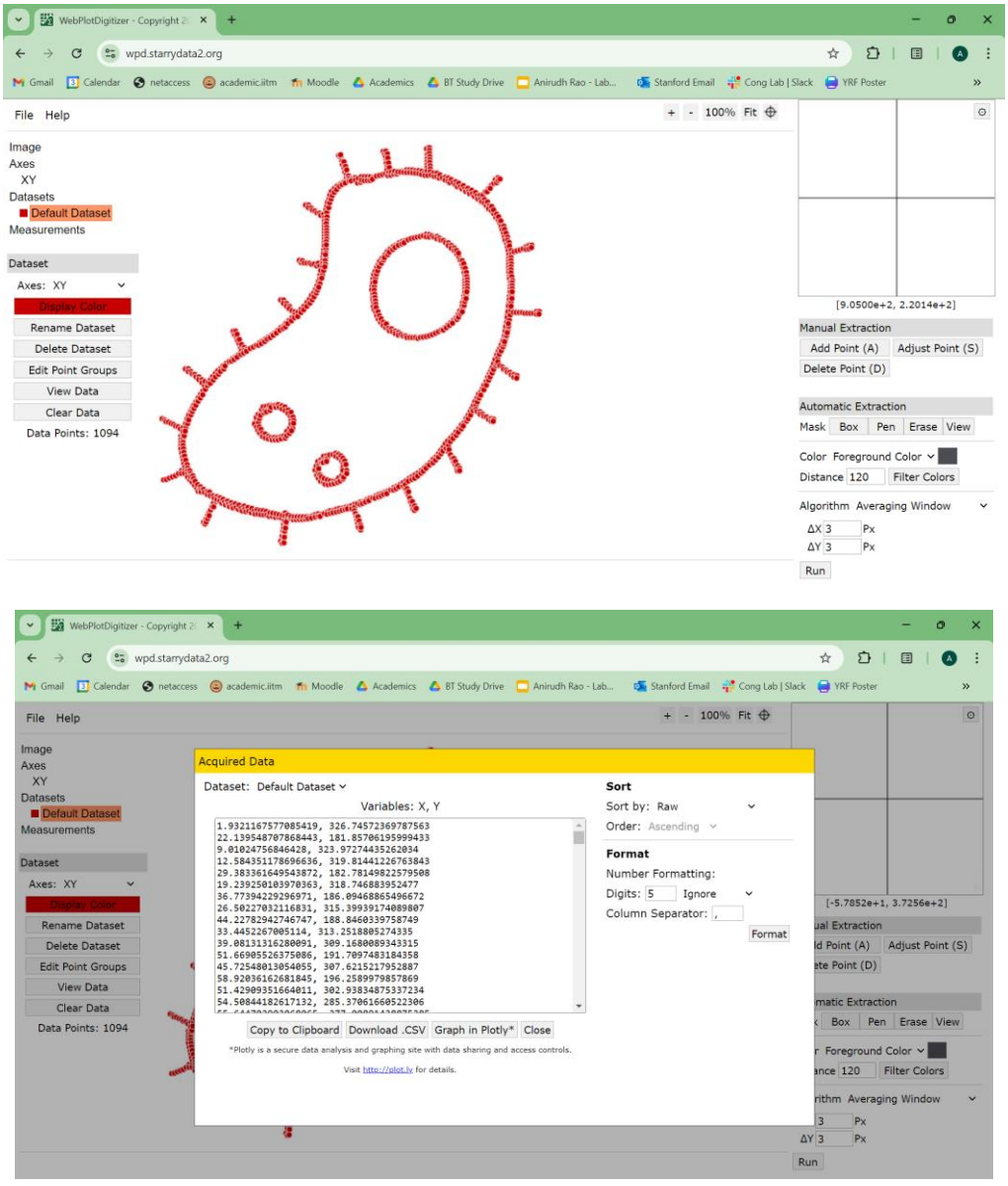

Figure 3: Extracting the coordinates of the points in the image

Data loading

- The .csv file was read into a Pandas DataFrame.
- All the coordinates were rounded off to the nearest integer. This essentially discretizes the coordinates and has them take integer values between 0 and 1000.
- A Boolean matrix of size 1000×1000 was used to represent the image. The $(i,j)^{\text{th}}$ element of the matrix was set to 1 if and only if (i,j) was the coordinate of a point after the discretization procedure. Else, the value was set to 0.

Transformation and visualisation

- A function was constructed to convert the Boolean matrix representation into (x,y) coordinates. These were then used to visualise the coordinates using a Matplotlib scatterplot (with the axes hidden).

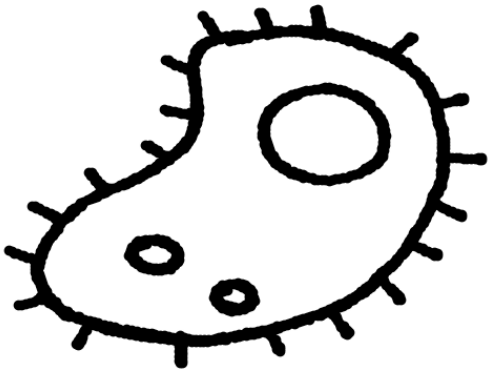

Figure 4: Matplotlib visualisation of the Boolean matrix representation of the original image

- Rotating an image clockwise by 90° requires us to perform a matrix operation that resembles:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{bmatrix}$$

- This is essentially the same as taking the transpose of the original matrix and reversing the order of its columns.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{bmatrix}$$

- We perform this sequence of operations using in-built Numpy functions `.T` and `flip()`.

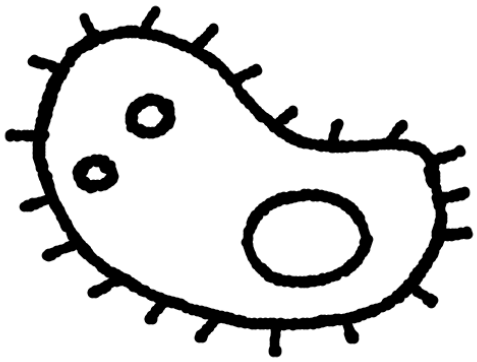

Figure 5: Matplotlib visualisation of the Boolean matrix representation of the rotated image

- Flipping an image as shown in the butterfly example requires us to perform a matrix operation that resembles:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{bmatrix}$$

- This is essentially the same as reversing the order of the rows of the original matrix.
- We perform this sequence of operations using in-built Numpy function `flip()`.

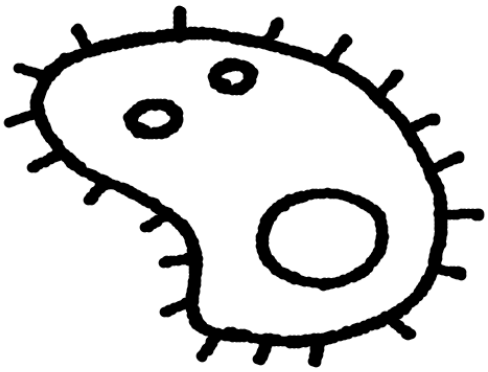

Figure 6: Matplotlib visualisation of the Boolean matrix representation of the flipped image

- Thus, the overall image transformation process looks like:

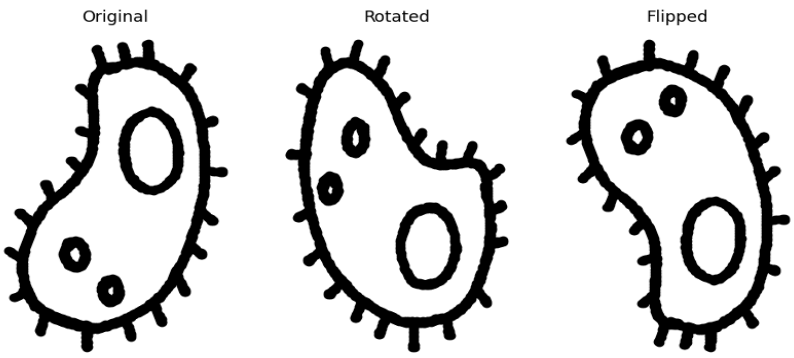

Figure 7: Overall output

Code and data availability

The Python code used for this assignment has been attached along with this report in a .zip folder. The original coordinates for the image have been attached as a .csv file in the same folder.