

DA5401 Data Analytics Lab - Assignment 1

Nandhakishore C S DA24M011

03 August 2024

1 Problem Statement

Acquire data from a hand drawn image and using algebraic operations (like matrix multiplication) to rotate and flip the images.

2 Prerequisites

The programming language used to implement this assignment is **Python** and no **image processing libraries** are used.

For math operations **Scientific Python (SciPy)** and **Numerical Python (NumPy)** are used. For visualising and saving plots, **Matplotlib** is used.

3 Data Acquisition

Data acquisition is done using the tool *WebPlotDigitizer*. The original camera image used for acquiring the 2D X-Y coordinate dataset can be found in Figure 1.

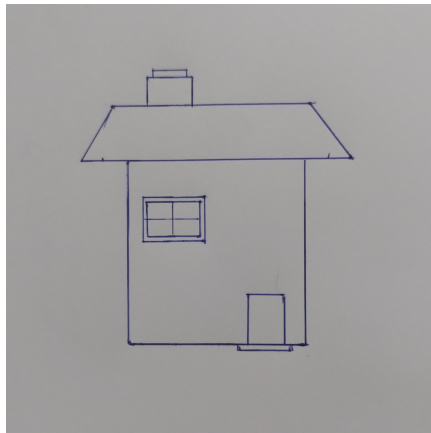


Figure 1: Hand drawn image with blue color pen

Once the fiducial points are fed to the *WebPlottdigitizer* tool, the hand drawn image is converted into a 2D dataset. We get the hand drawn image traced with points downloaded as a .csv file (Refer figures 2 and 3).

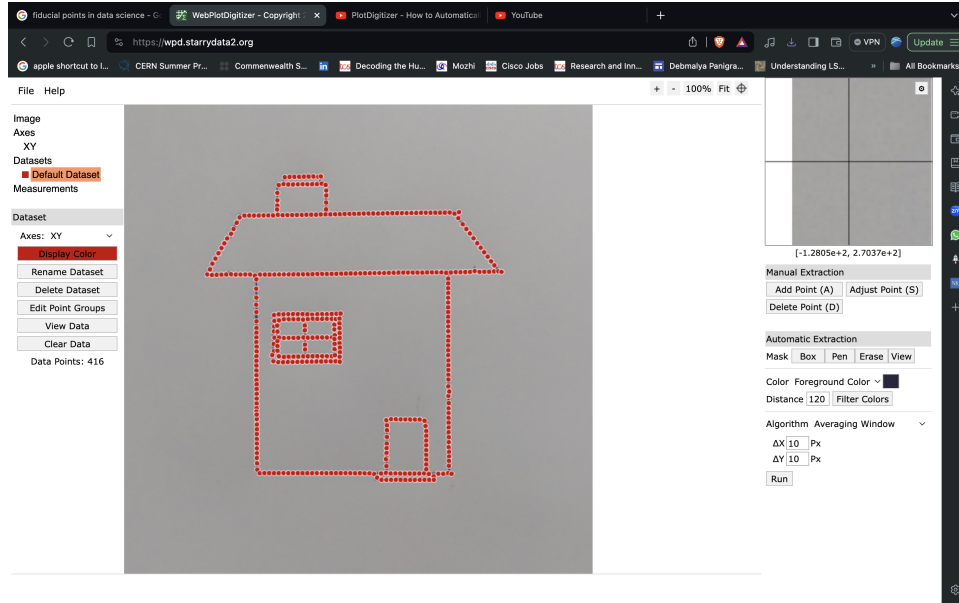


Figure 2: Hand drawn image with data points traced by the digitize tool

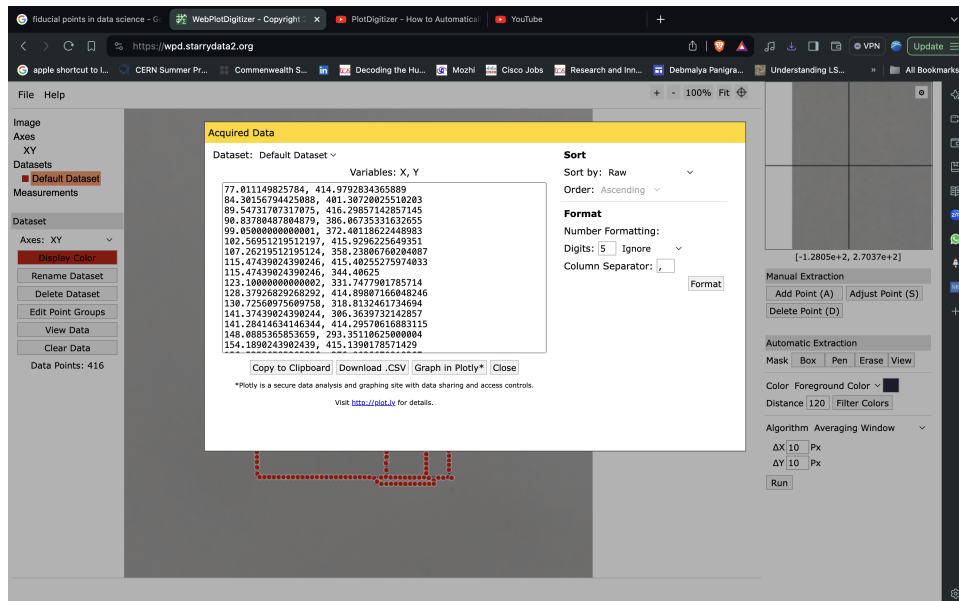
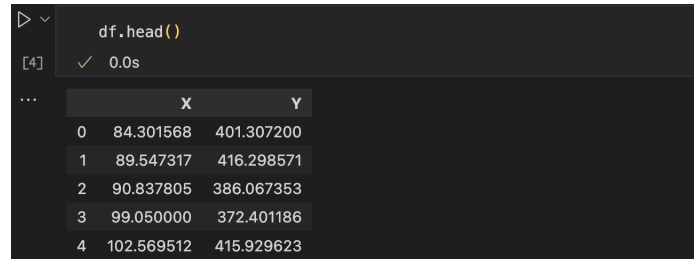


Figure 3: (X,Y) Coordinates as 2D dataset taken from the input image

4 Data Loading & Cleaning

The dataset (i.e.) (the *.csv* file) is then loaded into a Pandas dataframe. The dataset is searched for null/ missing/ NaN values. The dataset did not have any abnormal values and thus it is used as it is.

The dataset contains 415 coordinates (X,Y) pairs thus it results in a 415×415 matrix.



```
df.head()
[4] ✓ 0.0s
```

	X	Y
0	84.301568	401.307200
1	89.547317	416.298571
2	90.837805	386.067353
3	99.050000	372.401186
4	102.569512	415.929623

Figure 4: First five rows of the dataset after pre processing

The downloaded dataset has columns named with random names, and it was changed to 'X' and 'Y' for plotting the picture conveniently.

As the data downloaded from the *WebPlotdigitizer* tool has values stored as ascending values, the loaded dataset's image is inverted, (i.e.) the image is upside down. The loaded image as figure 5.

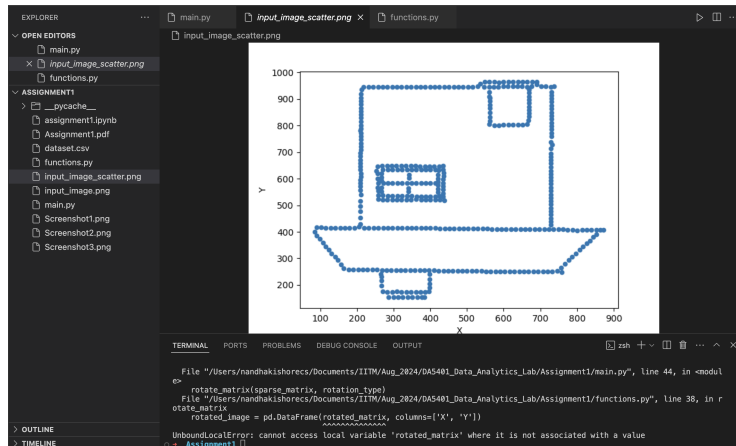


Figure 5: The scatter plot of the dataset.

(Note: The coding files saves this plot as *.png* file.)

5 Transformation - Matrix Multiplication

The core concept behind doing rotation, flipping operation on images in from computer graphics, where rotation matrices are multiplied with the image in

matrix which is in need of rotation. The general rotation matrix in 2D is as follows. (By convention the rotation matrix is denoted by R)

$$R_{2 \times 2} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$

where $R \in \mathbb{R}^2$

For rotation, it can be done in two ways: Clockwise (or) Counter Clockwise. Thus for rotation, the value of the angle (θ) can be either $+90^\circ$ or -90° .

$$R_{2 \times 2} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

rotation matrix for clockwise rotation.

$$R_{2 \times 2} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

rotation matrix for counter clockwise rotation.

For flipping the image, the angle is taken as $+180^\circ$ or -180° (i.e.) we rotate the images two times in the same direction.

$$R_{2 \times 2} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$$

rotation matrix for flipping.

As the dataframe only contains the only the coordinates of the points from the picture, but we are representing the image as a matrix. Thus the matrix would sparse in nature and we are converting the normal image into a sparse matrix. This is achieved by using the *sparse* module from *SciPy*.

As we have a 415×415 matrix, the shape of the sparse matrix is 415×2 . As the rotation matrix is of size, 2×2 , the resulting matrix is again of the same size 415×2 .

SciPy's *sparse* has two functions which convert a normal dense matrix into a sparse matrix by two functions, using compressed row and compressed column methods. This assignment incorporates compressed row method. *NumPy* library is used to perform sparse matrix multiplication using @ operator.

Once the sparse matrix is multiplied with the correct rotation matrix concerned, the sparse matrix is then converted to a dense matrix and it is plotted to see the rotation / flipping.

6 Visualization

The transformed images (rotated and flipped) are saved as *.png* files. Using *matplotlib* library and *Pandas* scatter plot function, the plots are visualised as follows:

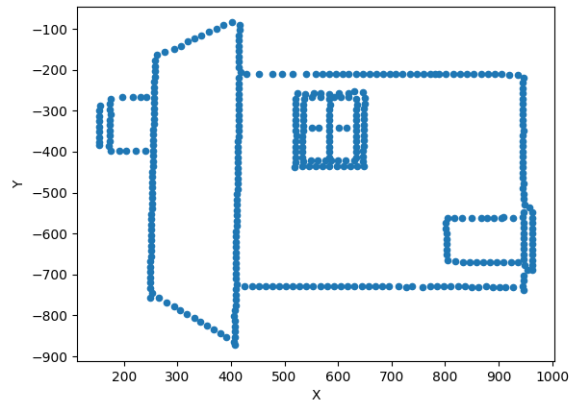


Figure 6: Clockwise Rotation of the given image

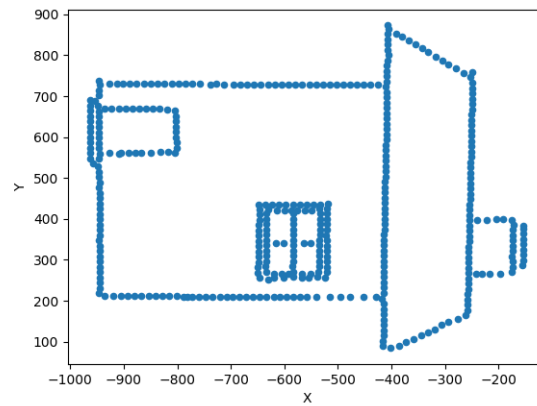


Figure 7: Counter- clockwise Rotation of the given image

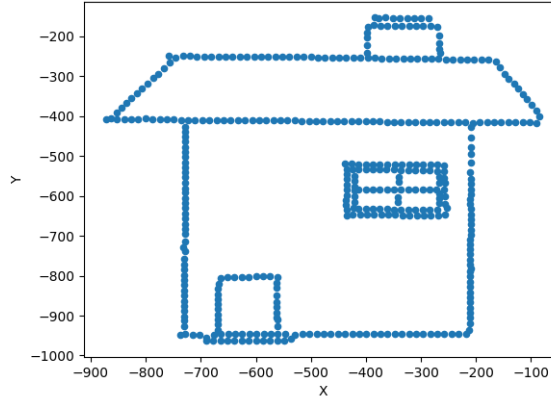


Figure 7: Flip Rotation of the given image

7 Instructions for the code file:

The *.zip* file contains two main python files - *main.py* and *functions.py*. The *main.py* contains code which will read the dataset and does basic data cleaning. The *functions.py* file has user-defined functions which do data cleaning and image rotation. Apart from the given python files, the dataset is also present in the *.zip* file as a *.csv* file.

To execute the file, make sure that the *main.py*, *functions.py* and *dataset.csv* are in the same folder. In terminal use the command ***python3 main.py*** to execute the files. A total of four images will be saved in the directory where the above files are stored. The saved images are the input image, clockwise rotated image, counter clockwise rotated image and the flipped image.

8 References

1. MIT Open Courseware 18.02 - Multi Variable Calculus, Rotational Matrices, Lecture #3
2. Sparse Matrix Multiplication in Python3