TDS Archive

Pre-Processing in OCR!!!

A basic explanation of the most widely used preprocessing techniques by the OCR system.

Susmith Reddy Follow 7 min read · Mar 25, 2019

296 7

Welcome to ***part II,*** in the series about **working of an OCR system.** In the *previous post*, we briefly discussed the different phases of an OCR system.

Among all the phases of OCR, ***Preprocessing*** and ***Segmentation*** are the most important phases, as the accuracy of the OCR system highly depends upon how well *Preprocessing* and *Segmentation* are performed. So, here we are going to learn some of the most basic and commonly used preprocessing techniques on an image.

Let's go…

The main objective of the *Preprocessing* phase is ***To make as easy as possible*** for the OCR system to distinguish a character/word from the background.

Some of the most basic and important ***Preprocessing*** techniques are:-

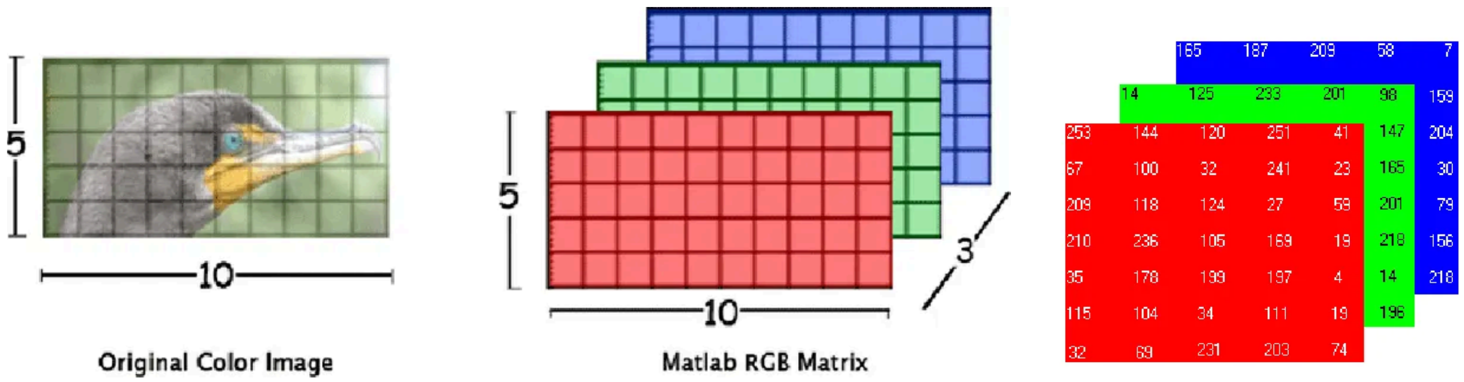
1) Binarization

2) Skew Correction

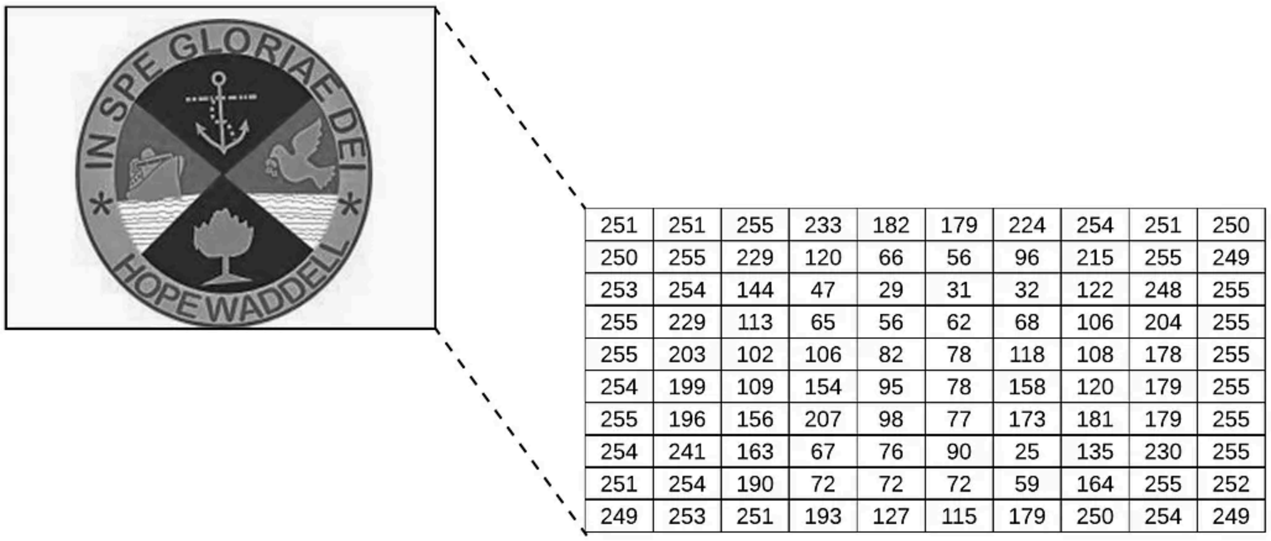
3) Noise Removal

4) Thinning and Skeletonization

Before discussing these techniques, let’s understand how an OCR system comprehends an image. For an OCR system, an *Image* is a multidimensional array (2D array if the image is grayscale (or) binary, 3D array if the image is coloured). Each cell in the matrix is called a pixel and it can store 8-bit integer which means the pixel range is 0–255.



Internal Representation of RGB image with Red, Green and Blue Channels. **Source: left image from semantics scholar, right image from researchgate.**

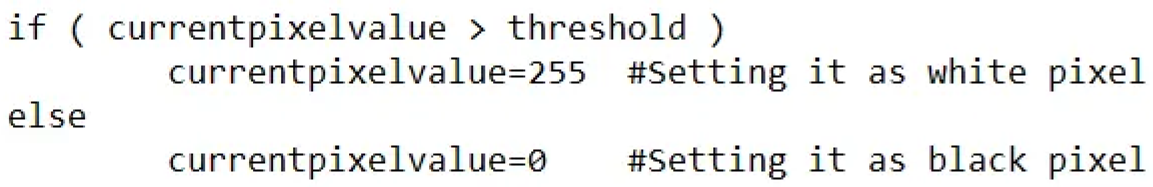


251	251	255	233	182	179	224	254	251	250
250	255	229	120	66	56	96	215	255	249
253	254	144	47	29	31	32	122	248	255
255	229	113	65	56	62	68	106	204	255
255	203	102	106	82	78	118	108	178	255
254	199	109	154	95	78	158	120	179	255
255	196	156	207	98	77	173	181	179	255
254	241	163	67	76	90	25	135	230	255
251	254	190	72	72	72	59	164	255	252
249	253	251	193	127	115	179	250	254	249

Internal Representation of Grayscale image. It has only one channel. **Source: ekababisong.org**

Let's go through each preprocessing technique mentioned above one-by-one

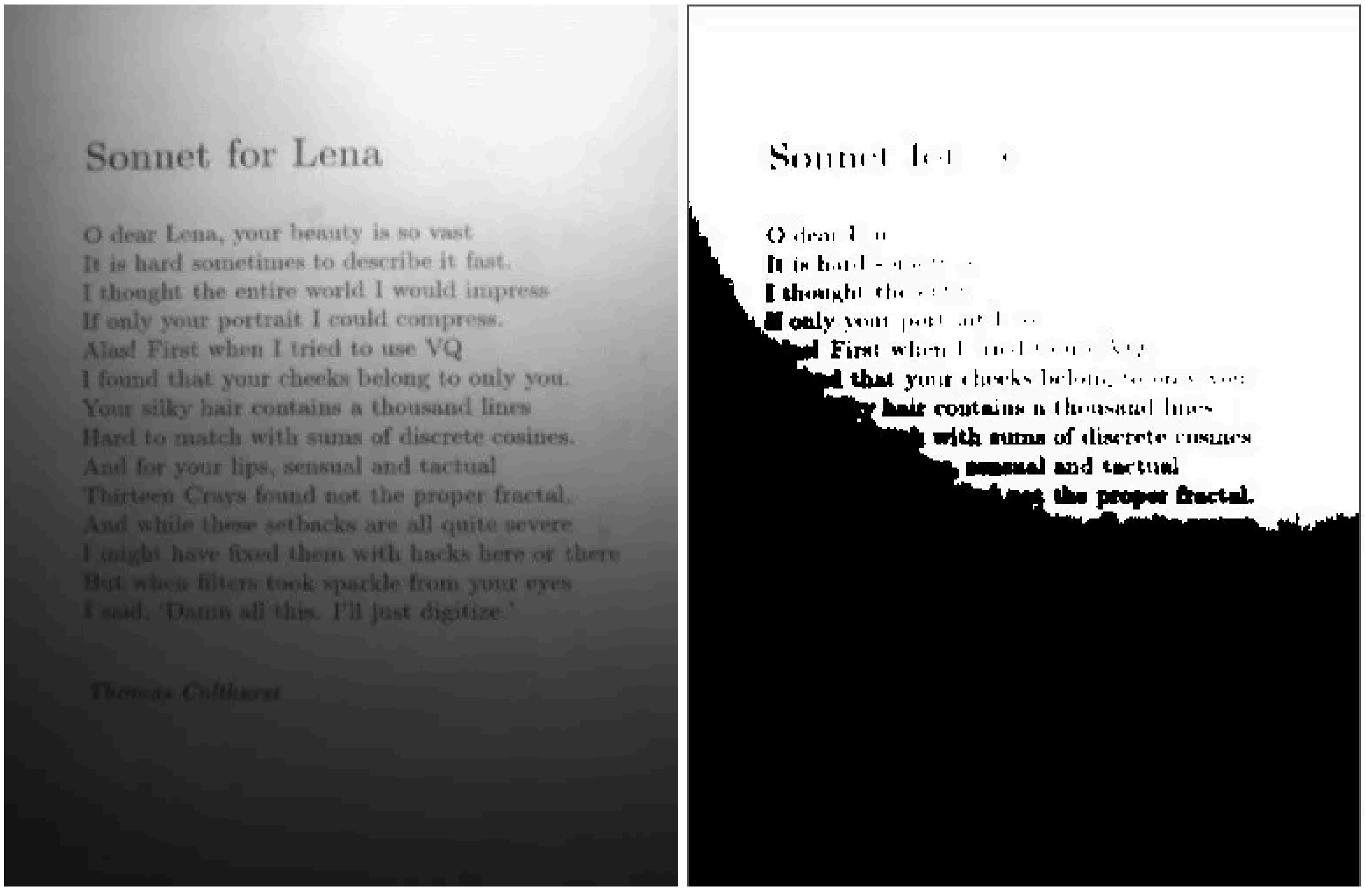
1. ***Binarization:*** In layman's terms *Binarization* means converting a coloured image into an image which consists of only black and white pixels (Black pixel value=0 and White pixel value=255). As a basic rule, this can be done by fixing a *threshold* (normally threshold=127, as it is exactly half of the pixel range 0–255). If the pixel value is greater than the threshold, it is considered as a white pixel, else considered as a black pixel.



```
if ( currentpixelvalue > threshold )
        currentpixelvalue=255  #Setting it as white pixel
else
        currentpixelvalue=0    #Setting it as black pixel
```

Binarization conditions. **Source: Image by author**

But this strategy may not always give us desired results. In the cases where lighting conditions are not uniform in the image, this method fails.



Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst

Binarization using a threshold on the image captured under non-uniform lighting. **Source: left image from this post and right image binarised by author.**

So, the crucial part of binarization is determining the ***threshold***. This can be done by using various techniques.

→ *Local Maxima and Minima Method* :

$$C(i,j) = \frac{I_{max} - I_{min}}{I_{\max} - I_{min} + \mathcal{E}}$$

Imax= Maximum pixel value in the image, Imin= Minimum pixel value in the image, E = Constant value **Source: Reference [2]**

C(i,j) is the *threshold* for a ***defined size*** of locality in the image (like a 10x10 size part). Using this strategy we'll have *different threshold values for different parts of the image,* depending on the surrounding lighting conditions but the transition is not that smooth.

→ *Otsu's Binarization*: This method gives a *threshold for the whole image* considering the various characteristics of the whole image (like lighting conditions, contrast, sharpness etc) and that threshold is used for Binarizing image.
This can be accomplished using OpenCV python in the following way:

```
ret, imgf = cv2.threshold(img, 0,
255,cv2.THRESH_BINARY,cv2.THRESH_OTSU) #imgf contains Binary image
```

-> *Adaptive Thresholding*: This method gives a threshold for a small part of the image depending on the characteristics of its locality and neighbours i.e there is no single fixed threshold for the whole image but every small part of the image has a different threshold depending upon the locality and also gives smooth transition.

```
imgf =
cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRE
SH_BINARY,11,2) #imgf contains Binary image
```

2. ***Skew Correction:*** While scanning a document, it might be slightly skewed (image aligned at a certain angle with horizontal) sometimes. While extracting the information from the scanned image, detecting & correcting the skew is crucial.
Several techniques are used for skew correction.

→ Projection profile method
→ Hough transformation method
→ Topline method
→ Scanline method

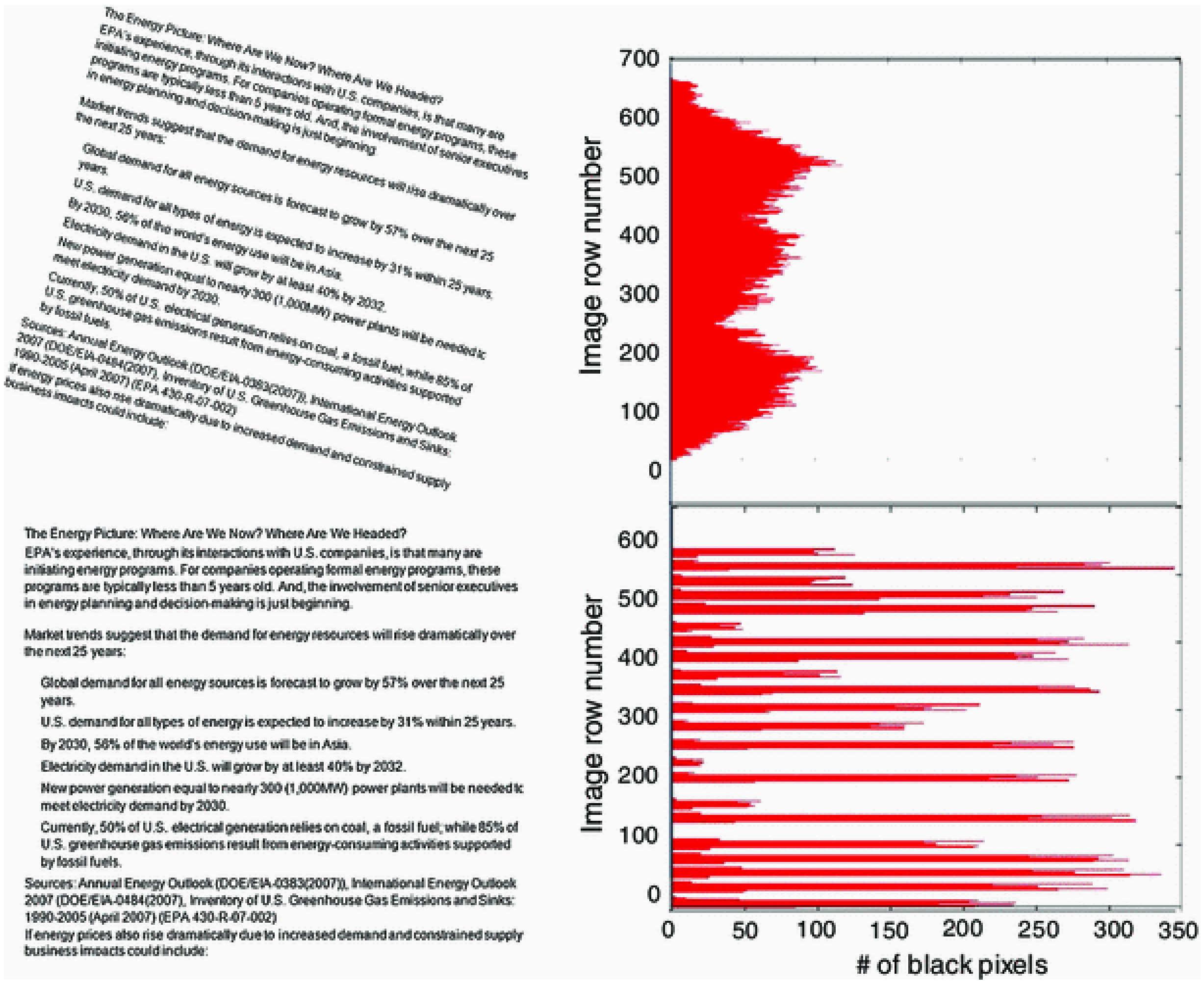
However, the *projection profile* method is the simplest, easiest and most widely used way to determine skew in documents.

In this method, First, we'll take the binary image, then

- project it horizontally (taking the sum of pixels along rows of the image matrix) to get a histogram of pixels along the height of the image i.e count of foreground pixels for every row.

- Now the image is rotated at various angles (at a small interval of angles called *Delta*) and the difference between the peaks will be calculated (*Variance* can also be used as one of the metrics). The angle at which the **maximum** difference between peaks (or *Variance*) is found, that corresponding angle will be the *Skew angle* for the image.

- After finding the Skew angle, we can correct the skewness by rotating the image through an angle equal to the skew angle in the ***opposite direction*** of skew.



The Energy Picture: Where Are We Now? Where Are We Headed?
EPA's experience, through its interactions with U.S. companies, is that many are initiating energy programs. For companies operating formal energy programs, these programs are typically less than 5 years old. And, the involvement of senior executives in energy planning and decision-making is just beginning.
Market trends suggest that the demand for energy resources will rise dramatically over the next 25 years:
Global demand for all energy sources is forecast to grow by 57% over the next 25 years.
U.S. demand for all types of energy is expected to increase by 31% within 25 years.
By 2030, 56% of the world's energy use will be in Asia.
Electricity demand in the U.S. will grow by at least 40% by 2032.
New power generation equal to nearly 300 (1,000MW) power plants will be needed tc meet electricity demand by 2030.
Currently, 50% of U.S. electrical generation relies on coal, a fossil fuel, while 85% of U.S. greenhouse gas emissions result from energy-consuming activities supported by fossil fuels.
Sources: Annual Energy Outlook (DOE/EIA-0383(2007)), International Energy Outlook 2007 (DOE/EIA-0484(2007)), Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990-2005 (April 2007) (EPA 430-R-07-002)
If energy prices also rise dramatically due to increased demand and constrained supply business impacts could include:

The Energy Picture: Where Are We Now? Where Are We Headed?
EPA's experience, through its interactions with U.S. companies, is that many are initiating energy programs. For companies operating formal energy programs, these programs are typically less than 5 years old. And, the involvement of senior executives in energy planning and decision-making is just beginning.

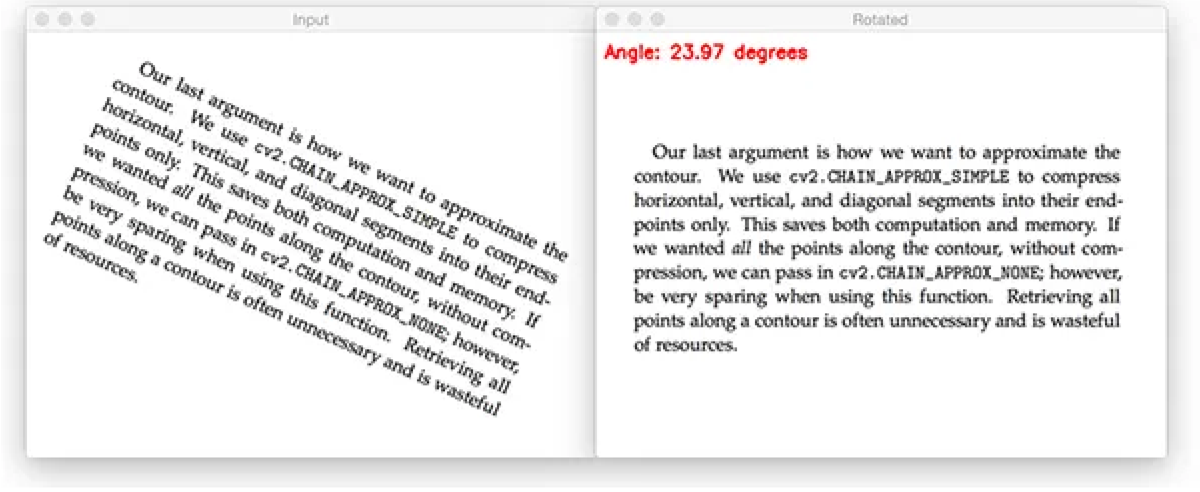
Market trends suggest that the demand for energy resources will rise dramatically over the next 25 years:

Global demand for all energy sources is forecast to grow by 57% over the next 25 years.
U.S. demand for all types of energy is expected to increase by 31% within 25 years.
By 2030, 56% of the world's energy use will be in Asia.
Electricity demand in the U.S. will grow by at least 40% by 2032.
New power generation equal to nearly 300 (1,000MW) power plants will be needed tc meet electricity demand by 2030.
Currently, 50% of U.S. electrical generation relies on coal, a fossil fuel; while 85% of U.S. greenhouse gas emissions result from energy-consuming activities supported by fossil fuels.

Sources: Annual Energy Outlook (DOE/EIA-0383(2007)), International Energy Outlook 2007 (DOE/EIA-0484(2007)), Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990-2005 (April 2007) (EPA 430-R-07-002)
If energy prices also rise dramatically due to increased demand and constrained supply business impacts could include:

Correcting skew using the Projection Profile method. **Source: Reference[1]**

```
import sys
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image as im
from scipy.ndimage import interpolation as inter

input_file = sys.argv[1]

img = im.open(input_file)

# convert to binary
wd, ht = img.size
pix = np.array(img.convert('1').getdata(), np.uint8)
bin_img = 1 - (pix.reshape((ht, wd)) / 255.0)
plt.imshow(bin_img, cmap='gray')
plt.savefig('binary.png')

def find_score(arr, angle):
    data = inter.rotate(arr, angle, reshape=False, order=0)
    hist = np.sum(data, axis=1)
    score = np.sum((hist[1:] - hist[:-1]) ** 2)
    return hist, score

delta = 1
limit = 5
angles = np.arange(-limit, limit+delta, delta)
scores = []
for angle in angles:
    hist, score = find_score(bin_img, angle)
    scores.append(score)

best_score = max(scores)
best_angle = angles[scores.index(best_score)]
print('Best angle: {}'.formate(best_angle))

# correct skew
data = inter.rotate(bin_img, best_angle, reshape=False, order=0)
img = im.fromarray((255 * data).astype("uint8")).convert("RGB")
img.save('skew_corrected.png')
```

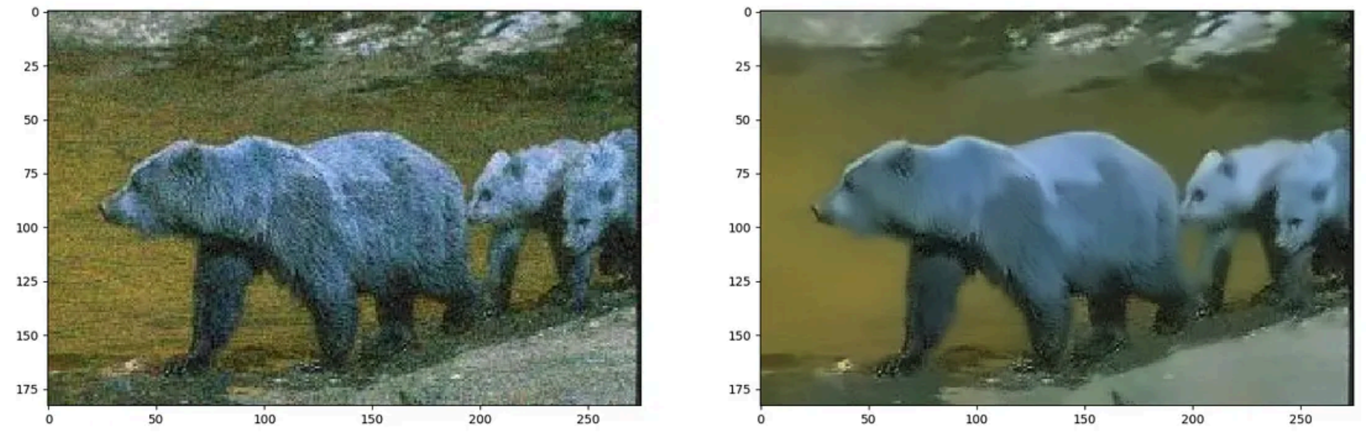


Our last argument is how we want to approximate the contour. We use cv2.CHAIN_APPROX_SIMPLE to compress horizontal, vertical, and diagonal segments into their end-points only. This saves both computation and memory. If we wanted *all* the points along the contour, without compression, we can pass in cv2.CHAIN_APPROX_NONE; however, be very sparing when using this function. Retrieving all points along a contour is often unnecessary and is wasteful of resources.

Skew Correction. **Source: pyimagesearch.com by Adrian Rosebrock**

3. ***Noise Removal:*** The main objective of the *Noise removal* stage is to smoothen the image by removing small dots/patches which have high intensity than the rest of the image. Noise removal can be performed for both *Coloured* and *Binary images*.
One way of performing Noise removal by using OpenCV *fastNlMeansDenoisingColored* function.

```
import numpy as np
import cv2
from matplotlib import pyplot as plt
# Reading image from folder where it is stored
img = cv2.imread('bear.png')
# denoising of image saving it into dst image
dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 15)
# Plotting of source and destination image
plt.subplot(121), plt.imshow(img)
plt.subplot(122), plt.imshow(dst)
plt.show()
```



Smoothening and Denoising of image. **Source: Reference [4]**

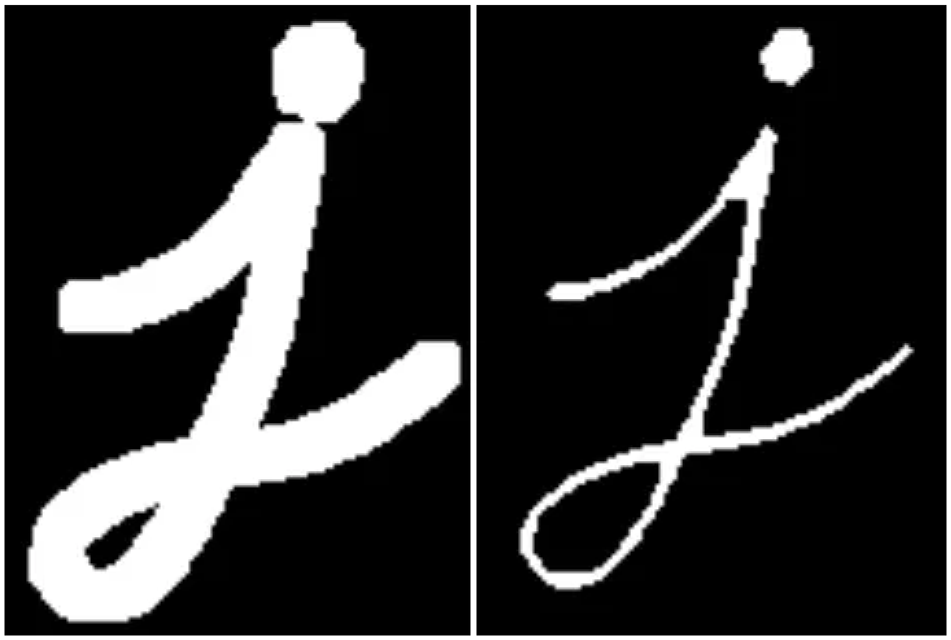
More about *Noise removal & Image smoothening* techniques can be found in ***this*** wonderful article

4. ***Thinning and Skeletonization***: This is an optional preprocessing task which depends on the context in which the OCR is being used.
→ If we are using the OCR system for the printed text, No need of performing this task because the printed text always has a uniform stroke width.
→ If we are using the OCR system for handwritten text, this task has to be performed *since different writers have a different style of writing and hence different stroke width*. So to make the width of strokes uniform, we have to perform *Thinning and Skeletonization*.

This can be performed using OpenCV in the following way

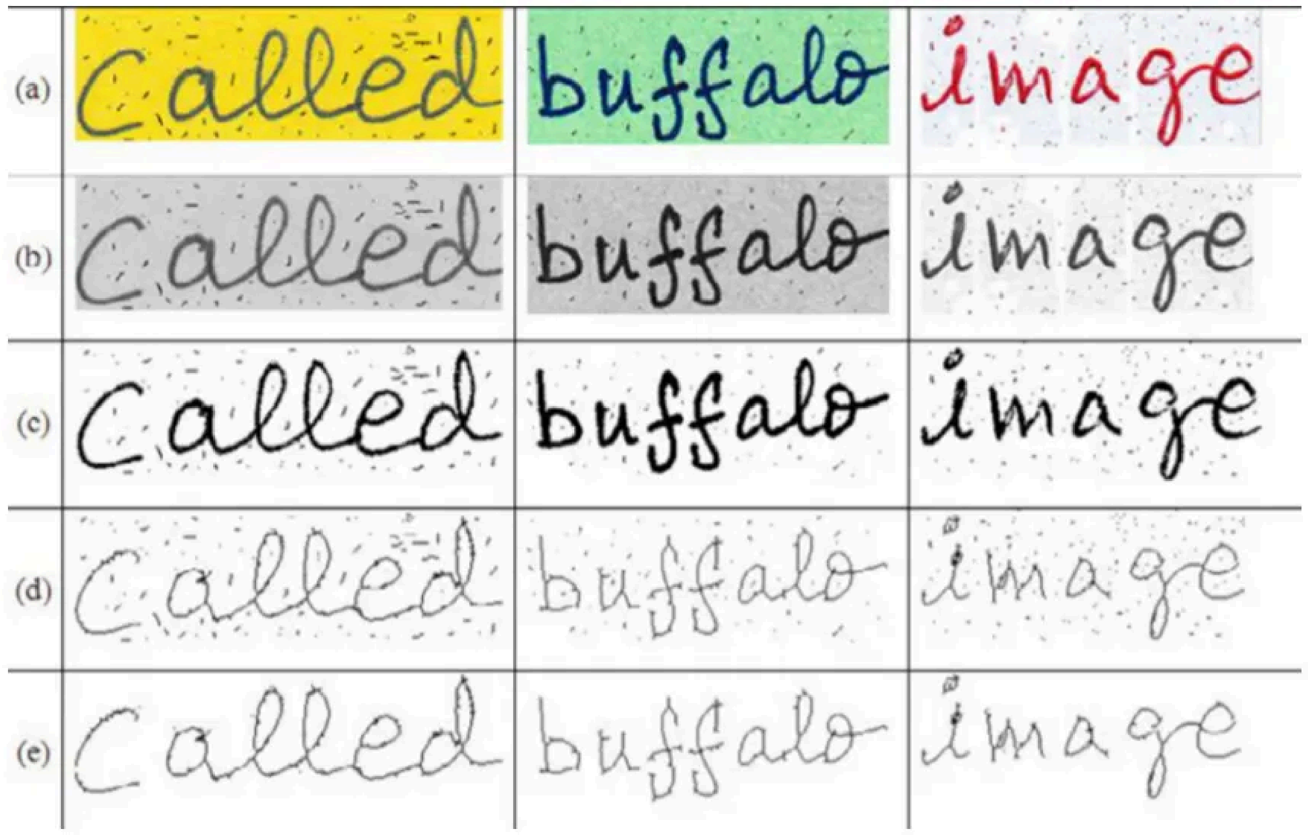
```
import cv2
import numpy as np

img = cv2.imread('j.png',0)
kernel = np.ones((5,5),np.uint8)
erosion = cv2.erode(img,kernel,iterations = 1)
```

In the above code, *Thinning* of the image depends upon kernel size and no.of iterations.



Before and After Thinning and Skeletonization. **Source: datacamp**

In this article, we have seen some of the basic and most widely used ***Preprocessing*** techniques which gives us a basic idea of what's happening inside the OCR system. An example of ***preprocessing*** workflow can be seen in the below image.



(a) Original Image. (b) Converted to Grayscale. ©Binarized image. (d) Thinning and Skeletonization are done. (e) Noise Removed

Source: Reference [5]

I hope you got an essence of how *Preprocessing* is performed in the OCR.

Further Reading:

In *part-III*, we'll see the *Segmentation* techniques used by the OCR system.

Happy Learning !!!!

Any doubts, Suggestions & Corrections are Welcome. 😃

References:

[1] Shafii, M., Sid-Ahmed, M. Skew detection and correction based on an axes-parallel bounding box. *IJDAR* **18,** 59–71 (2015).

https://doi.org/10.1007/s10032-014-0230-y

[2] Jyotsna, S. Chauhan, E. Sharma and A. Doegar, “Binarization techniques for degraded document images — A review,” *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, 2016, pp. 163–166, doi: 10.1109/ICRITO.2016.7784945.

[3] A. Papandreou and B. Gatos, “A Novel Skew Detection Technique Based on Vertical Projections,” *2011 International Conference on Document Analysis and Recognition*, Beijing, 2011, pp. 384–388, doi: 10.1109/ICDAR.2011.85.

(CVPRW), Long Beach, CA, USA, 2019, pp. 1717–1721, doi: 10.1109/CVPRW.2019.00221.

[5] Choudhary, Amit & Rishi, Rahul & Savita, Ahlawat. (2013). A New Character Segmentation Approach for Off-Line Cursive Handwritten Words. Procedia Computer Science. 17. 88–95. 10.1016/j.procs.2013.05.013.

Machine Learning Pre Processing Ocr Image Processing