

Image Segmentation Report

CS3330: Image and Video Processing Coursework: Document Image Binarization

Damon Tran

Student Number: 160111190

**1.Background**

Aim of document image binarization

Document image binarization is a method used for taking an image of a text document and separating different objects within it generally foreground (text) from background, using various approaches via thresholding on a grayscale image, to separate the text from the rest of the picture to make it more readable [1].

Typical documents you would use this on, would be where the image is affected by various variables that make the document hard to read as displayed in (figure 1). The text although visible, it is difficult to see due to dark patches and stains. However, through applying image binarization you can solve this issue.

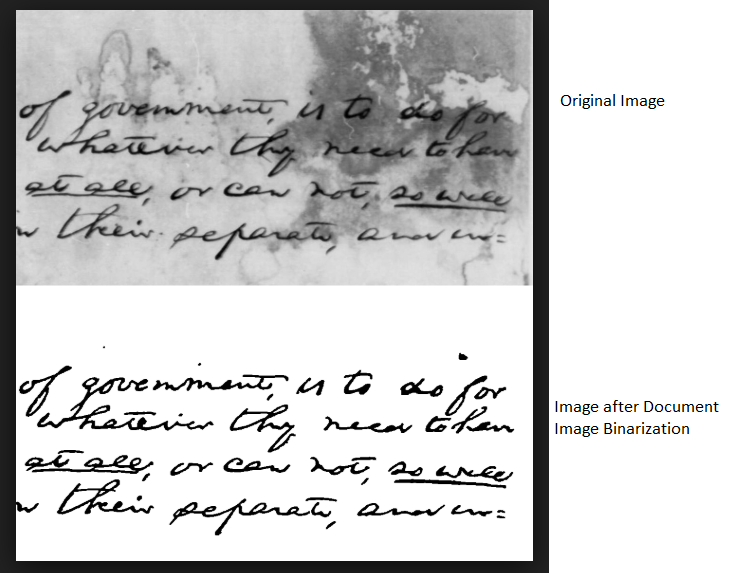


Figure 1 - reference [2]

Why is binarizing a document image useful?

It is useful because mid-level processes take an image as an input, analyses and processes the image to obtain the relevant attributes which are then passed to a high-level processing technique to apply to said attributes [3]. Acting as a phase in the processing pipeline, setting the image up for other steps to achieve a specific outcome. Meaning that we can apply any high-level technique we wish to the output we have generated from the mid-level process. Depending on our goal the flexibility of what we can do is quite varied.

For example, if the goal were to restore historical documents the degradation of the paper could obstruct people from reading it. it is possible to apply image binarization to solve this issue, then apply any high-level technique depending on what the aim is.

Existing approaches to document image binarization

There are many approaches but the three which have been researched are Parker, Eikvil and Sauvola.

Parker’s algorithm is based off the idea, that it is impossible to choose one single threshold value that can effectively binarize the whole image [4]. Instead, uses multiple thresholds which are derived from each pixel and applied to sub-regions to solve the problem of segmenting an image which is affected by degradation [4]. Results show when compared to other binarizing techniques that it performs better given the image has poor or close to none illumination [4].

Eikvil works by creating sub-images of the image to binarize, and then identifying and classifying pixels to separate text from background [5]. Specifically, optimal for grayscale images where there are large discrepancies in the grey levels of the image (foreground and background) as well as where intensities of pixels within the locality of an area differs [5]. Results from research indicate that it performs well in comparison to simple binarization techniques and is fast computationally [5].

Sauvola is designed with the goal of binarizing a general range of text document images that are affected by different types of image defects [6], with the concept that each part of the picture (text, background and picture) are intrinsic to the whole image itself [6]. Solving issues that many document image binarization algorithms face such as, noise and illumination by splitting up the image into equal size neighborhoods, deriving a threshold value for each one [6]. Sauvola performs well and operates to a similar level when compared to Niblack’s and Eikvil’s. Contrasting with Parker’s, which did poorly showing the algorithm suffers where illumination varies seen in figure 2 [6].

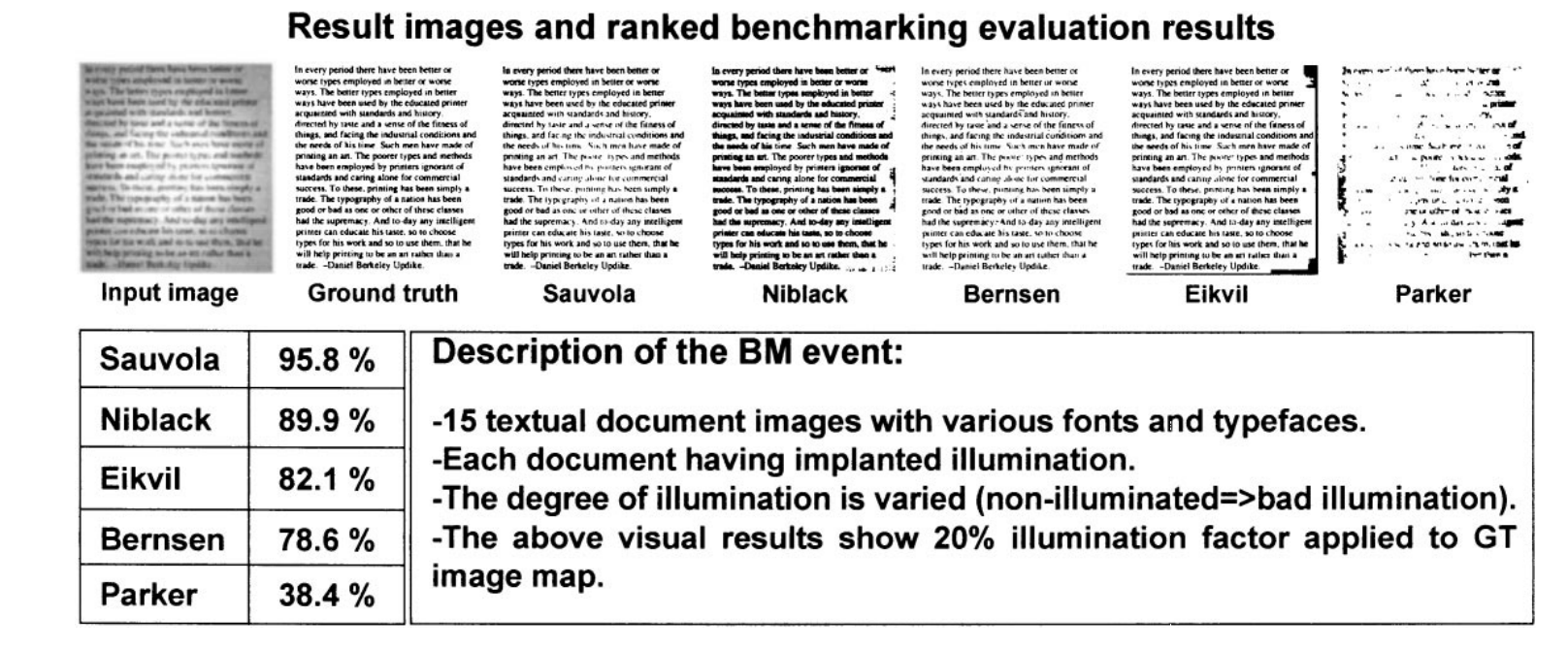


Figure 2 - Visual and numeric results on the comparison algorithms applied to illuminated, textual images. [6]

The performance of algorithms differs in respect to their aims, they try to binarize text document images with a problem in mind, thus have been designed to cater to that problem. Each approach uses the same logic of thresholding but their methods of creating the thresholds vary which in turn will produce varying results as seen in the aforementioned figure 2 [6]. Depending on degradation type or condition of the document image, each algorithm can perform better or worse than the others.

**2. Algorithm**

Motivation

Sauvola is suitable for document image binarization because it can binarize images, reduce noise and illumination producing consistently good results [6]. It has been compared to algorithms well known for performance and even outperforming them [6]. Because of the algorithms adaptive nature, it can perform well across a variation of document types with different degradations proving it to be quite flexible [6].

Drawbacks

Despite how well the algorithm performs, there are a few drawbacks that have been indicated in research papers. Sauvola only performs well given the text document is affected by noise or illumination [6]. But suffers when the image is clear i.e. when there is a small contrast between foreground and background [8] because there is little variation in pixels, it is harder to obtain precise threshold values to differentiate whether the pixels are text or background [7] which can be seen in figure 3. Cases like this, performance of the algorithm degrades. Another disadvantage is that Sauvola is requires a lot of computational power [7] because it needs lots of resources to binarize a text document image.

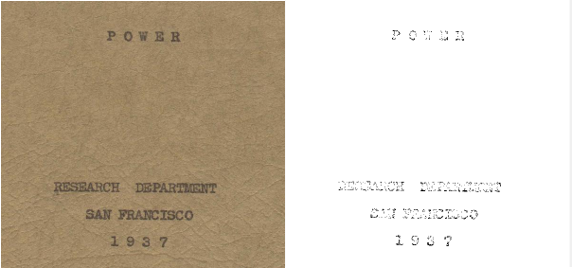


Figure 3 - Original Image (Left) Sauvola applied to Image (Right)

Sauvola takes a grayscale image and outputs a binary image which represents the segmentation, this is the separation of text from background.

It does this using a local adaptive approach to obtain the threshold values. First taking the grayscale image and separates it into equal sized neighborhoods n x n (n being the window size which is determined manually [8])

It finds the mean values of each neighborhood by using a filter on the greyscale image which outputs a new matrix consisting of all the average values within each neighborhood. Then finds the standard deviation of all pixels from the original greyscale image to our mean values, which is then used in the thresholding formula. Lastly, use the threshold value to determine whether pixels in the original greyscale Image are text or background and output it.

Pseudocode

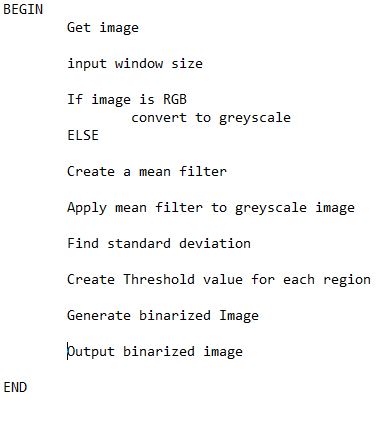


Figure 4- High-Level Pseudocode



Figure 5 Thresholding formula used in Sauvola [8]

**3. Results**

Methodology

A subjective analysis of the Sauvola algorithm will be made when applied to the provided test images describing and indicating levels of detail of binarization with descriptions. Also, comparing how well the algorithm has done in relation to the ground truth images as well, which contain a “correct” level of segmentation and evaluating Sauvola from that.

Testing and Results

|  |  |  |
| --- | --- | --- |
| Test N0 | Test Images | Result |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |

Results:

Test 1-

Sauvola has segmented the image from text and background quite clearly making it more readable for the text on the right that was covered in a dark shade of noise. The output shows the dark shade is still there due the weakness stated earlier—that the algorithm tends to suffer where there is little variation in pixel intensity [7] thus making it harder for it to segment the text from the dark shaded background in this image. Despite this, it has still managed to do it. Overall It has segmented the original image well and achieved its intended aim.

Test 2 –

The original image has a brown background and black ink text, suggesting that Sauvola should be able to segment this well due to the large variation in intensities which has been indicated to be a strength of the algorithm [6].

The result shows the original image completely binarized, text separated from the background making it clearly readable, even where the original image had a dark shade of bleeding coming through which made the writing obscure. Although there are still remnants of the bleeding effect in the output, it is very minimal. Some of the writing is faint, this is shown in figure 6. Due to the weakness of Sauvola it led to poor binarization in the region containing this specific word. Overall, Sauvola has performed well.

https://i.gyazo.com/c865d94b8eaf71ada386dcf2d88613db.png

Figure 6- Thin pen stroke from original Image(left) Thin pen stroke on binarized Image (right)

Test 3 –

The result is very good, as it has managed to separate the text from background including areas where text from the page below were made faintly visible on the front page (see figure 7). In the output these areas have been completely erased and all is left is the text, which in fact is very sharp and vivid as well clear separation from the background.

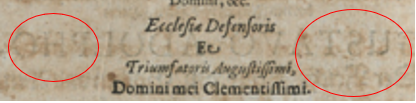


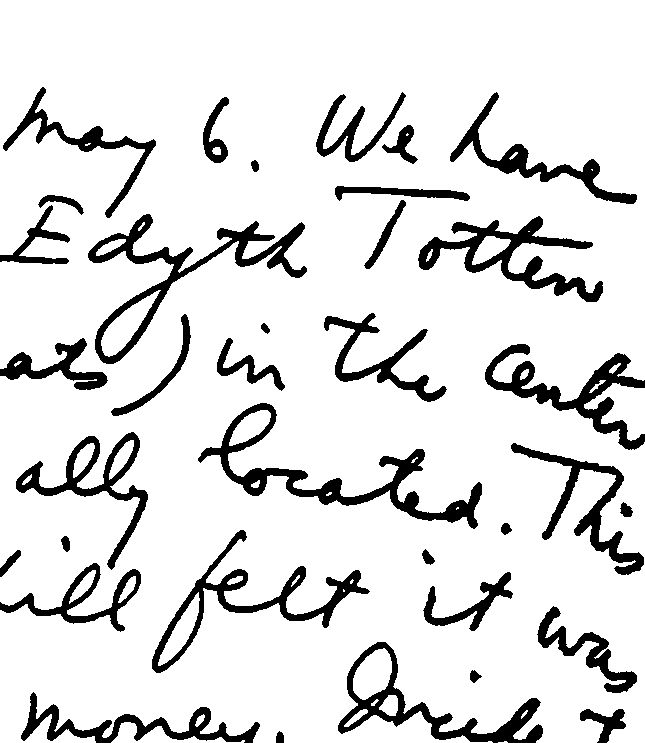
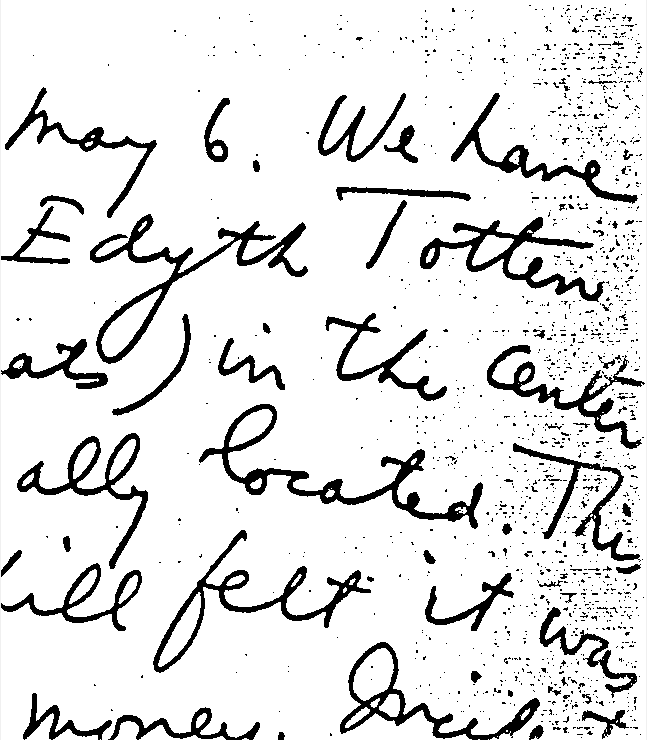
Figure 7- Original Image showing areas where text from another page is visible

Test4 –

Sauvola has clearly performed the worst on this image than the others. The output text is very faint, and in some cases words are not readable. This is largely due to the weaknesses that the Sauvola algorithm has which have been mentioned earlier [7]. The original document image having very fine, thin writing as well as the text colour being a light grey or black makes it hard to differentiate between foreground and background which ultimately led to this poor binarization.

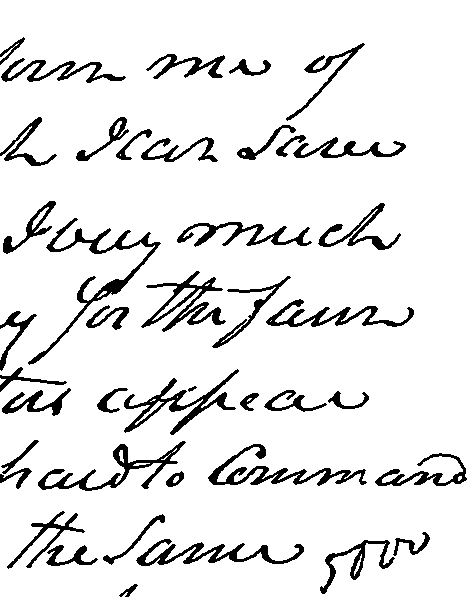
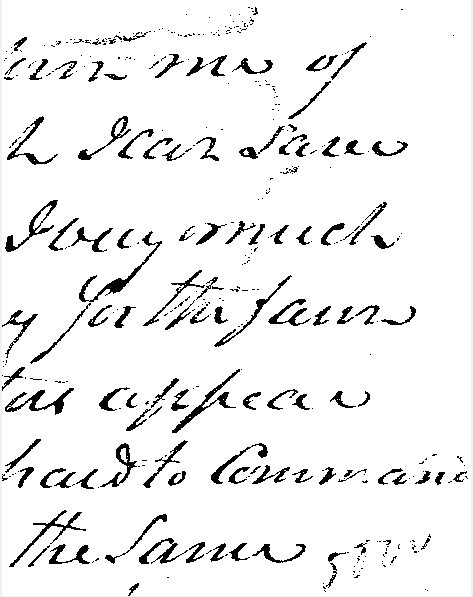
Comparisons to ground truth images:

Ground Truth images (left) Sauvola images (Right)



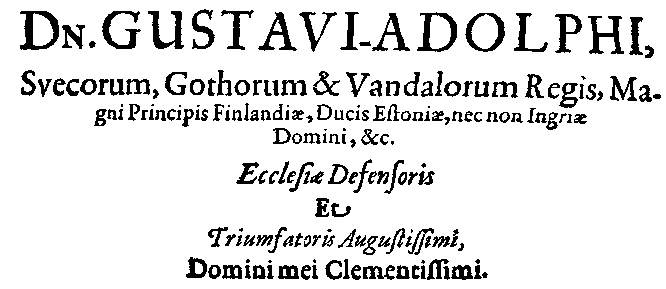
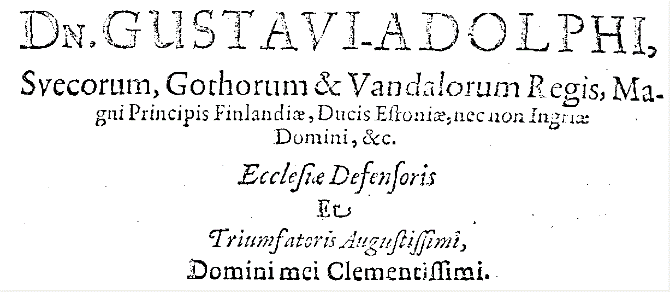
Comparison 1 - HW1.png

Sauvola has performed well when compared with a “correct” version of the segmented document image. All text is readable and clearly defined, only area lacking in the result is some minor noise on the right-hand side even still, the document image has been binarized and is quite close to the ground truth image.



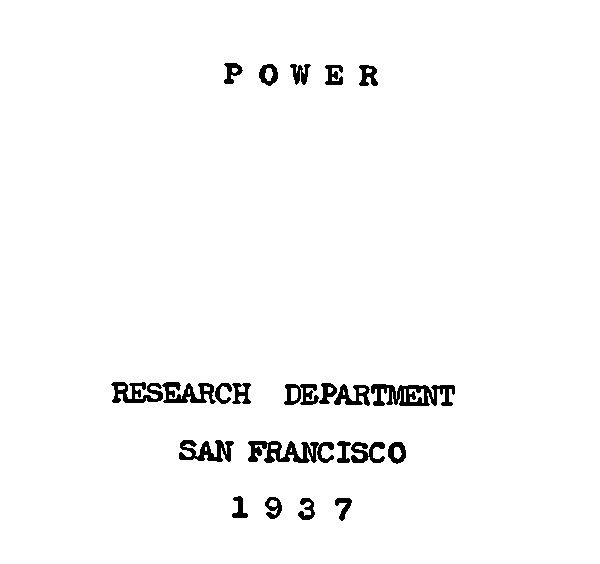
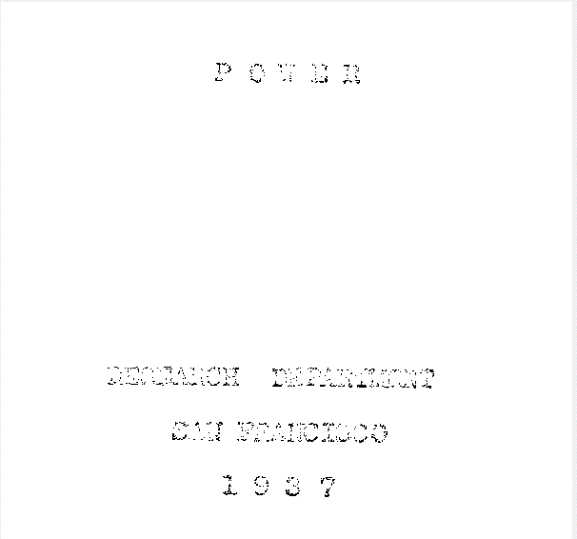
Comparison 2 - HW4.png

We can see that Sauvola has successfully segmented it. Text is readable, background clearly separated from the words. Only areas where it does not do as well, is where there are small amounts of noise from where the bleeding of the paper occurs and the word that is very faint on the bottom right. But overall Sauvola has managed well and binarized it sufficiently.



Comparison 3 - PR4.png

The output from Sauvola on this document image is good. All words and text are visible and easily decipherable from the background. Only part that suffers is, simply how faint the text comes out. Although, the text from the Sauvola image is still readable it varies and pales in comparison to the ground truth image where all words are clearly bold and black. Despite this, Sauvola has achieved a good result and the goal of binarizing the document image has been met.



Comparison 4 - PR7.png

On the last image, Sauvola is insignificant to the ground truth image. Text barely visible and most words are not readable. It is a very poor binarization, however it has been segmented and achieved its aim.

From testing the algorithm on the test images with varying window (width) sizes for the neighborhood. I have found that the algorithm can improve binarization performance given the size of the neighborhood you choose to use. Take test 4 for example, the binarization of that document image was done with a neighborhood of 20x20 as were all the test images. But test 4 gave a poor binarization. Upon increasing the width size, you can improve it (figure 8) showing a significant difference.

However, if the neighborhood is too big it can decrease performance see figure 9. Suggesting that there is an optimal size for the neighborhood when binarizing different document images.

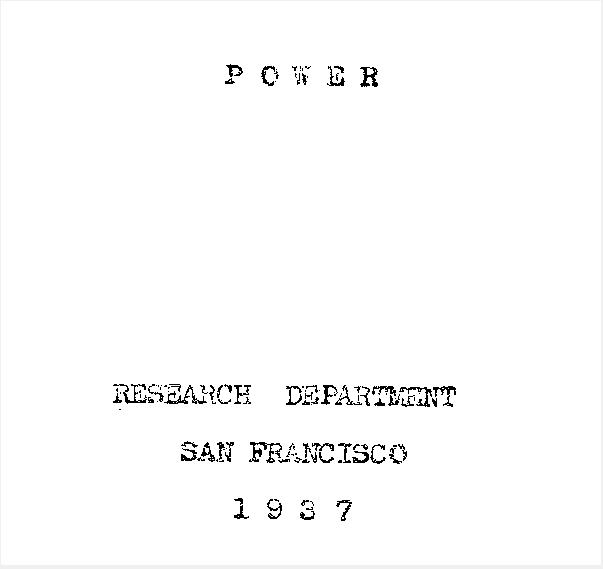


Figure 8 - Sauvola applied to test 4 image with 100x100 Neighbourhood

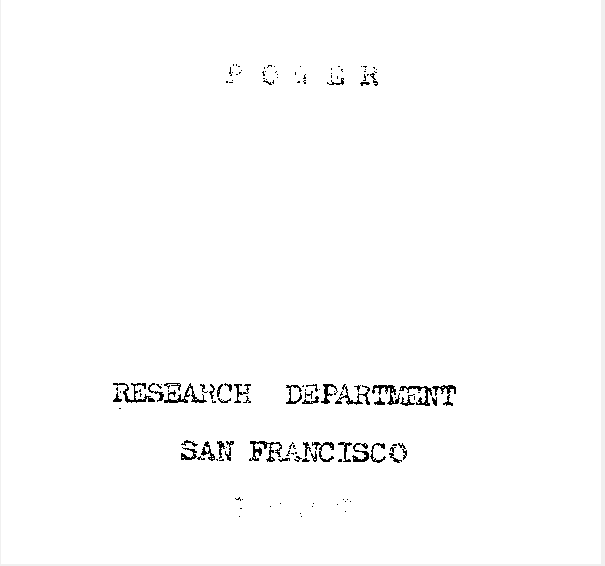


Figure 9 - Sauvola applied to test 4 image with 200x200 Neighbourhood

**4. Conclusion**

Summary

Sauvola as a document image binarization algorithm is excellent due to its flexibility and robust nature, handling different document and degradation types. However, suffers when the text is very small and has a pixel intensity too similar to the background. But, overall is a very good method for document image binarization.

Performance of the algorithm can be improved depending on size of neighbourhood used, suggesting that there is an optimal neighbourhood size that can be found to achieve best binarization results.

Limitations

Main limitations, I did not use as many references that I could have. The implication of this is, there may be essential parts of Sauvola I did not take into account and therefore have a limited grasp or knowledge of the concept in its entirety which may affect the report.

Another limitation is the small sample of test images that I had to work with, because of such a small sample I did not get to apply Sauvola to a wide variety of document image types affected by different degradations. Which could change the outcome of my results and thus the conclusion I have drawn on the algorithm.

Including the fact that the algorithm results were not objectively compared to the ground truths. They were subjectively analyzed and because of this the conclusion may not be entirely accurate, there could be important information that could be extrapolated from data to describe differences that I may have missed. Ultimately affecting my conclusion and the perception of how well the algorithm truly performs as a document image binarization technique.

Word count: 1998

**References**

1. Karthika M, Ajay James , A Novel Approach for Document Image Binarization using Bit-Plane Slicing, Romania , Tirgu-Mures, 2014.
2. https://webee.technion.ac.il/labs/cgm/Computer-Graphics-Multimedia/Software/Software.html, 26th November 2018.
3. Dr. Harry Goldingay, Image and Video Processing CS3330 Introduction, Aston University.
4. J. R. Parker, C. Jennings, Thresholding Using an Illumination Model, ICDAR ’93, Japan, 1993.
5. Line Eikvil, Torfinn Taxt and Knut Moen, A Fast Adaptive Method for Binarization of Document Images, France, 1991.
6. J. Sauvola, M. PietikaKinen Adaptive document image binarization, Machine Vision and Media Processing Group, Infotech Oulu, University of Oulu, P.O. BOX 4500, FIN-90401 Oulu, Finland, 21 January 1999.
7. Mahua Nandy (Pal) , Satadal Saha, An Analytical Study of different Document Image Binarization Methods, Department of CSE, MCKV Institute of Engineering, Howrah, India.
8. Rashmi Saini, Document Image Binarization Techniques, Developments and Related Issues: A Review, Assistant Professor, Govind Ballabh Pant Engineering College Pauri Garhwal, 246001, Uttarakhand, India.

**Appendix A**

Functions:

Sauvola.m

Scripts:

generate\_results.m

**Function – Sauvola.m**

function biImg = Sauvola(img,width)

%SAUVOLA Binarizes an image by using local average thresholding

%The method takes two parameters an image that is in the local workspace.

%if you wish to run this method load an rgb image in (MAKE SURE IT IS

%NOT ALREADY grayscale).

%it also takes a width value that you can set for yourself to determine

%size of the local neighbourhood. i.e. if you use 3 as the value then the

%local neighbourhood will be a size of 3x3.

%The image is then turned into a grayscale image and converted to a double

%matrix so that we can perform calculations to it later.

%r is a variable that is used in the thresholding formula which is a

%set to a fixed value of 128

%k is a value in the thresholding formula which takes on positive values

%and is set to 0.5

%Creates a meanFilter to get the average value for every neighbourhood of

%size wxw or width x width.

%The meanFilter is then applied to our grayscale image that we want to

%binarize to gain the average value in every neighbourhood.

%We used the standard deviation formula sqrt((x-mean). ^2) x representing our

%grey\_img, and mean being our mean\_img so we can find how far each pixel in

%the original image is from the average within each neighbourhood.

%Lastly, we use the Thresholding formula to obtain the threshold value at

%every pixel

%Then assign our output "biImg" to (grey\_img >threshold) which sets every

%pixel value in bi\_img to 0 or 1's corresponding to whether the pixel

%value exceeds the threshold or not.

grey\_img = rgb2gray(img);

grey\_img = double(grey\_img);

r =128

k= 0.5

global w;

w = width;

meanFilter = fspecial('average',w);

mean\_img = filter2(meanFilter,grey\_img);

deviation = (mean\_img - grey\_img)

deviation = deviation.^2

deviation = deviation.^0.5

threshold = mean\_img.\*(1-k \*(1-deviation/128))

biImg = (grey\_img > threshold);

imshow(biImg);

end

**Script – generate\_results.m**

%Test 1

img = imread("HW1.png")

figure('Name', 'Test 1'), Sauvola(img,20);

%Test 2

img = imread("HW4.png")

figure('Name', 'Test 2'), Sauvola(img,20);

%Test 3

img = imread("PR4.png")

figure('Name', 'Test 3'), Sauvola(img,20);

%Test 4

img = imread("PR7.png")

figure('Name', 'Test 4'), Sauvola(img,20);

%Test 4 with increase neighbourhood size improved performance

img = imread("PR7.png")

figure('Name', 'Test 4- With Increased Neighbourhood size improved performance'), Sauvola(img,100);

%Test 4 with increase neighbourhood size degrading performance

img = imread("PR7.png")

figure('Name', 'Test 4- With Increased Neighbourhood size degrading performance'), Sauvola(img,200);