A Binarization Threshold Method for Images of Filled-out Forms

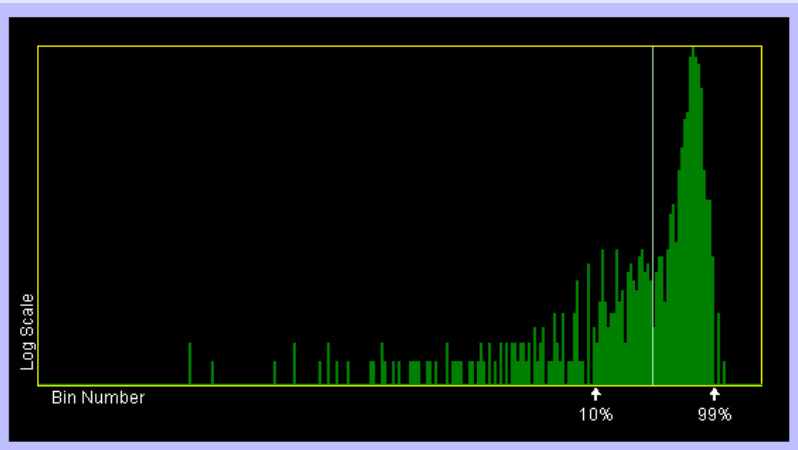
By
Randall Christensen

Image Assumptions

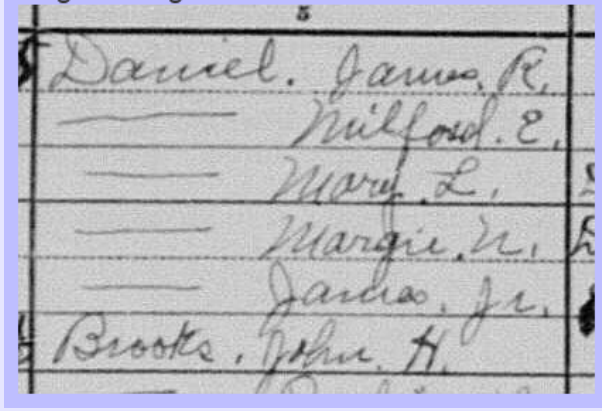
- Image consists of “white”, “black”, and “gray”
- Most of the pixels are “white” background
- Some of the pixels are “black” / ”gray”
- Little illumination gradient is present
- Sufficient contrast exists

The Binarization Problem

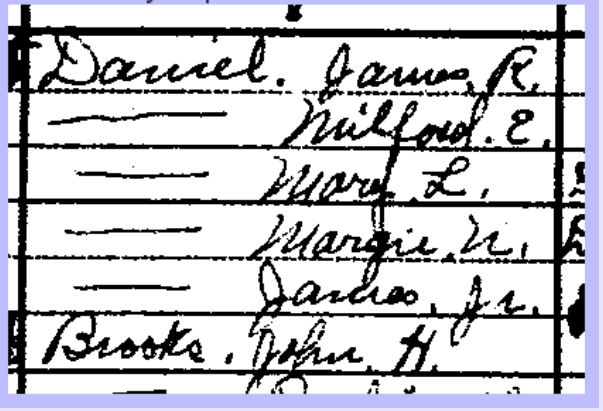
- Black, White, and Gray
- Separate Whites from Blacks and Grays



Original Image



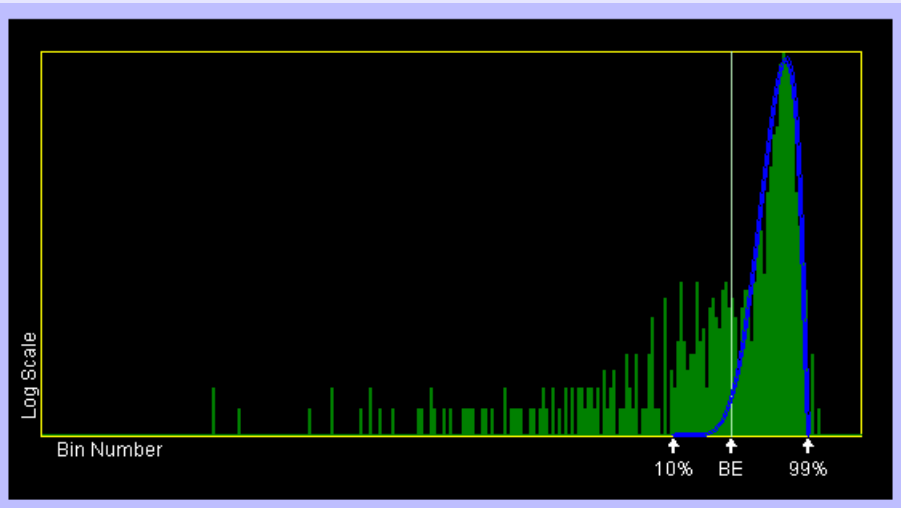
Binarized by Inspection



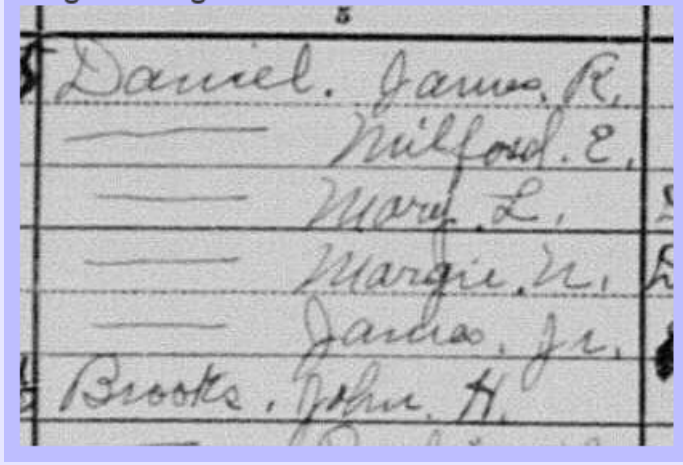
A Solution

- Fit a Kumaraswamy Distribution to White pixels
- Select a Confidence Level
- Find Corresponding Pixel Level

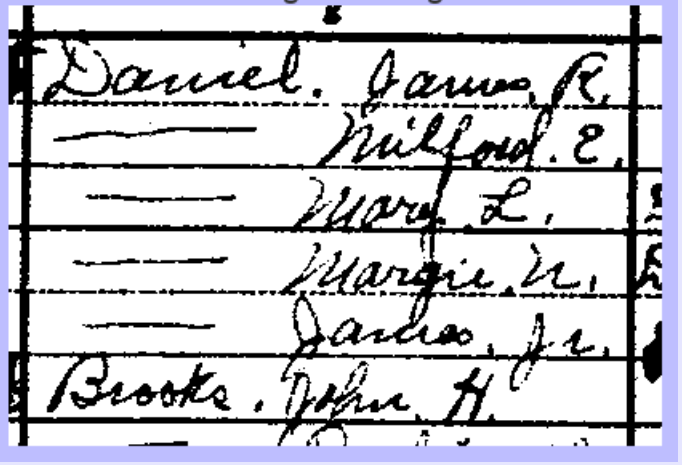
Sample 1



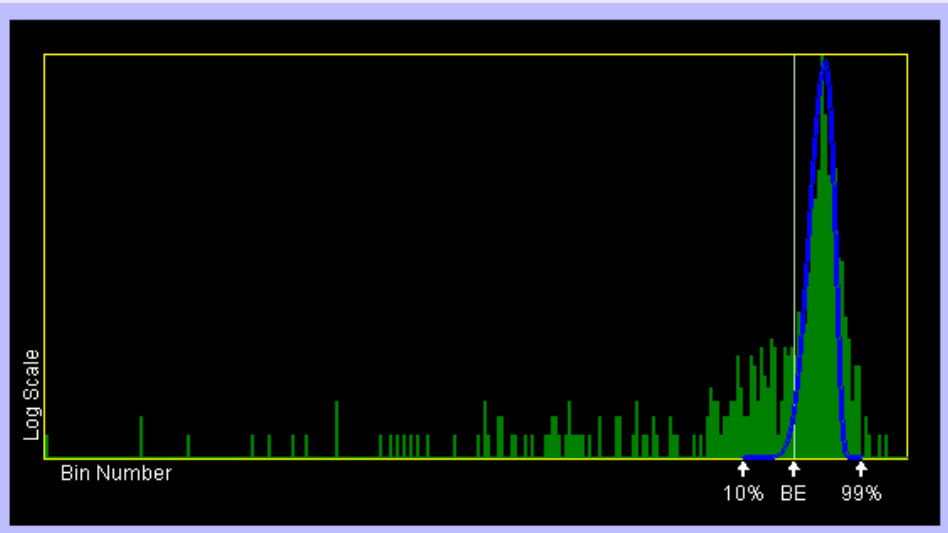
Original Image



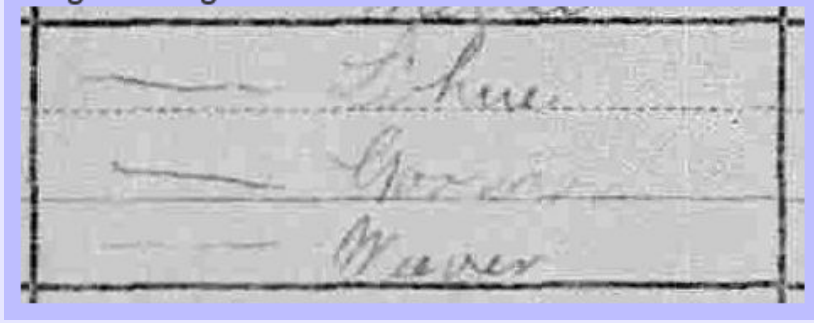
Binarized with Background Edge



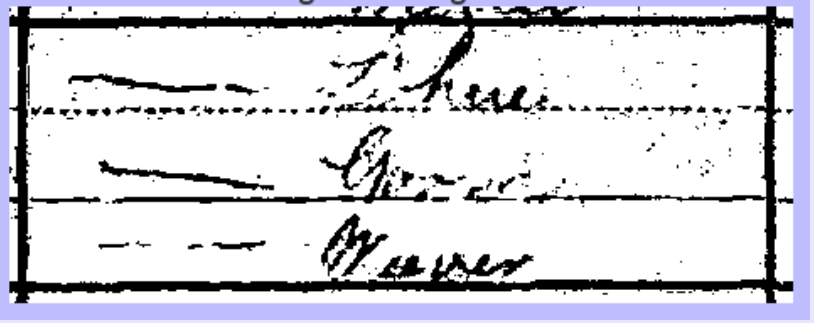
Sample 2



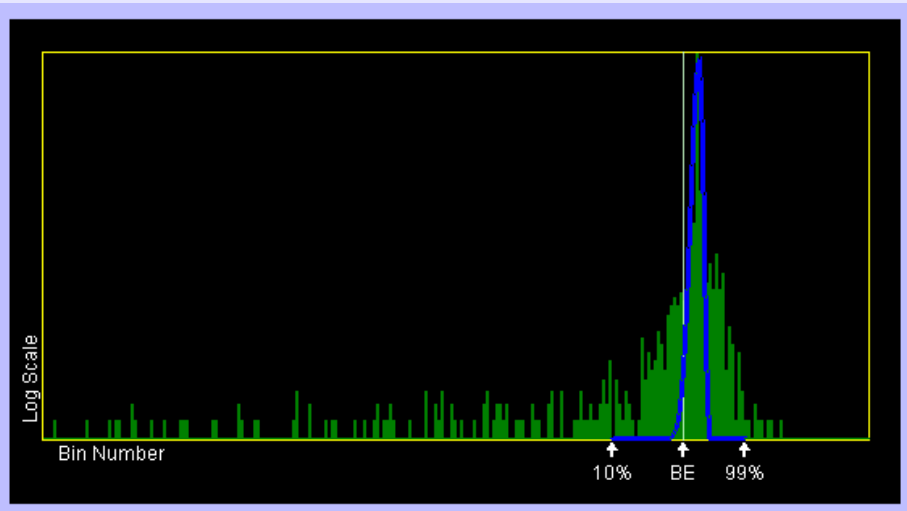
Original Image



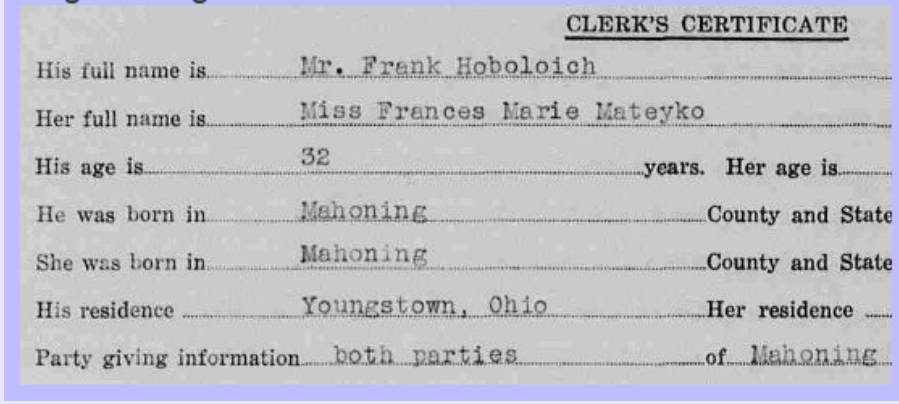
Binarized with Background Edge



Sample 3



Original Image



CLERK'S CERTIFICATE

His full name is Mr. Frank Hoboloich

Her full name is Miss Frances Marie Mateyko

His age is 32 years. Her age is

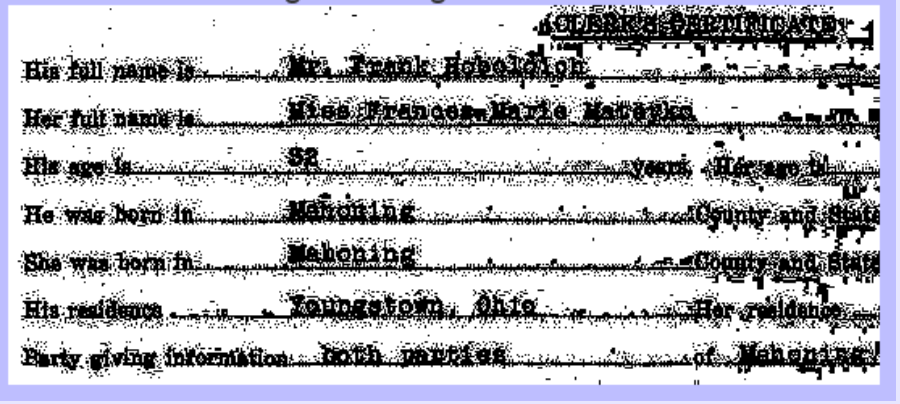
He was born in Mahoning County and State

She was born in Mahoning County and State

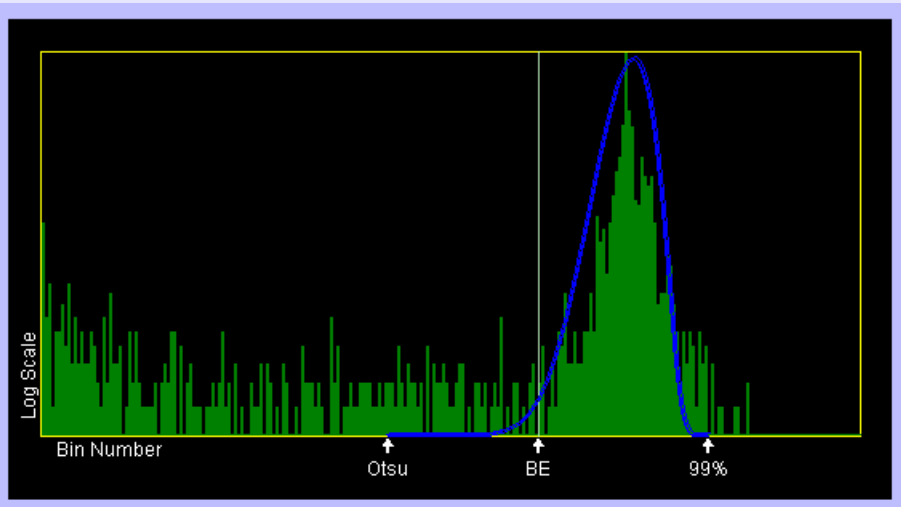
His residence Youngstown, Ohio Her residence

Party giving information both parties of Mahoning

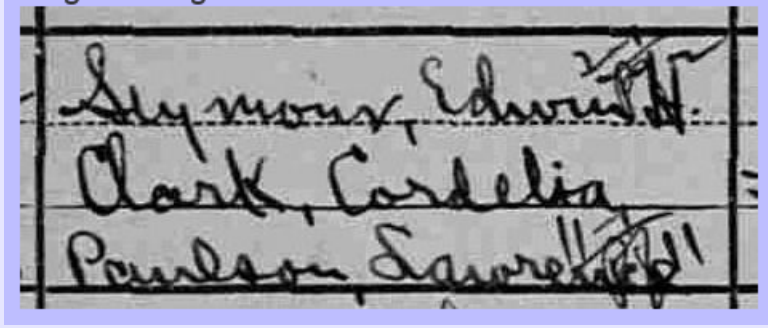
Binarized with Background Edge



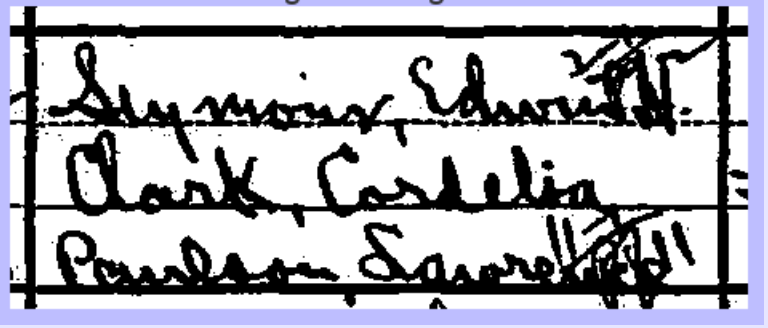
Sample 4



Original Image



Binarized with Background Edge



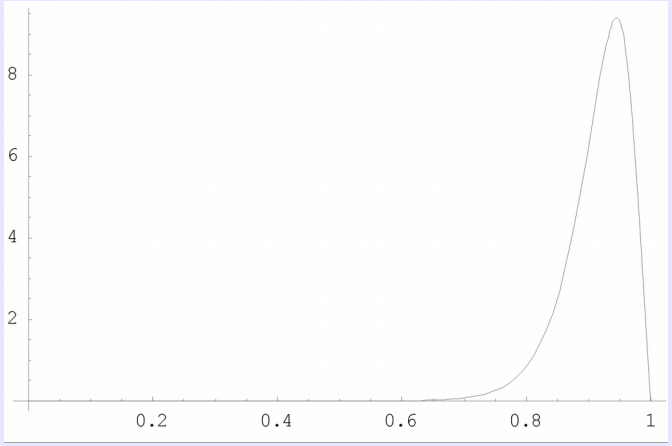
Why the Kumaraswamy Distribution?

- It is finite
- It is computationally easy
- It looks like the actual data

The Kumaraswamy Distribution

- Cumulative Distribution Function $1-\left(1-x^{a}\right)^{b}$
- Probability Density Function $abx^{(a-1)}\left(1-x^{a}\right)^{(b-1)}$
- Inverse CDF $1-\left(1-y^{(1/b)}\right)^{(1/a)}$
- Constraints $x\in[0,1],a\succ 0,b\succ 0$

PDF of a=20, b=3



Fitting to Obtain the Parameters

- Start with the quartile values 0 < q1 < q2 < q3 < 1
- Get the initial approximations

$$a_0 = \frac{\log\left(1 - 1/2^{2/7}\right) - \log\left(1 - 3^{1/7}/2^{2/7}\right)}{\log\left(q3/q1\right)} = \frac{1.5}{\log\left(q3/q1\right)}$$

$$b_0 = \frac{\log(2)}{\log\left(1/\left(1 - q2^{a_0}\right)\right)}$$

- Iterate new values until close enough

$$a_{i+1} = \frac{\log\left(1 - (1/4)^{1/b_i}\right) - \log\left(1 - (3/4)^{1/b_i}\right)}{\log\left(q3/q1\right)}$$

$$b_{i+1} = \frac{\log(2)}{\log\left(1/\left(1 - q2^{a_{i+1}}\right)\right)}$$

Conclusion

- The background edge threshold method seems to work quite well.
- It is easy to implement
- It may be useful for you

A Binarization Threshold Method for Images of Filled-out Forms

By
Randall Christensen

There is nothing really new in this presentation. The only thing that is unusual is that we are going to talk about a probability distribution that you have probably never heard of before, the Kumaraswamy Distribution. The reason I want to talk about it is that, having used it in conjunction with many thousands of images from the U.S. 1930 Census, I have found that it works very well for finding the lower edge of the background pixel values.

Image Assumptions

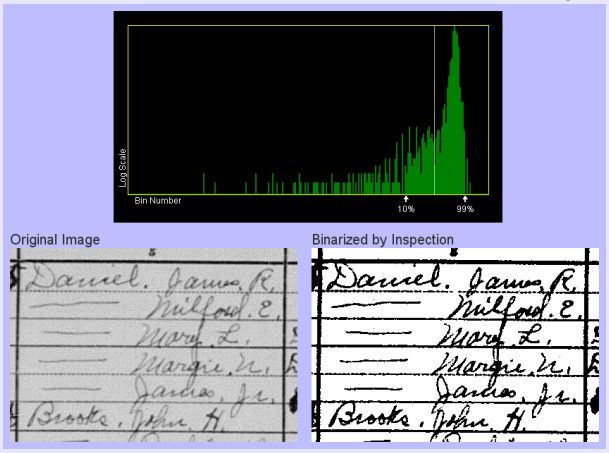
- Image consists of “white”, “black”, and “gray”
- Most of the pixels are “white” background
- Some of the pixels are “black” / ”gray”
- Little illumination gradient is present
- Sufficient contrast exists

By a filled-out form I mean that the image consists of mostly “white” background, has some printed lines, and some hand-written / typed information in it. The “white” background needs to have similar properties across the entire image. There needs to be enough contrast so that discreet pixel values are not too clumped.

The reason for doing binarization at all, is that when we are searching for structures in an image, like a line, the computations are often less ambiguous if the input data represents an estimate of whether a pixel is part of the structure or not. Hence the value of using the binarization process.

The Binarization Problem

- Black, White, and Gray
- Separate Whites from Blacks and Grays



The original images have a mixture of “white”, “black”, and “gray” pixels. We would like to find a pixel intensity value to use as a threshold so that the image can be represented in just pure white and pure black pixels.

In this example we start with an image, create the histogram of the image’s intensity values, choose a threshold at the lower edge of the big background lump by inspection. Here it is represented by the white line on the histogram. Then we create a binarized image by dividing the original image at the threshold.

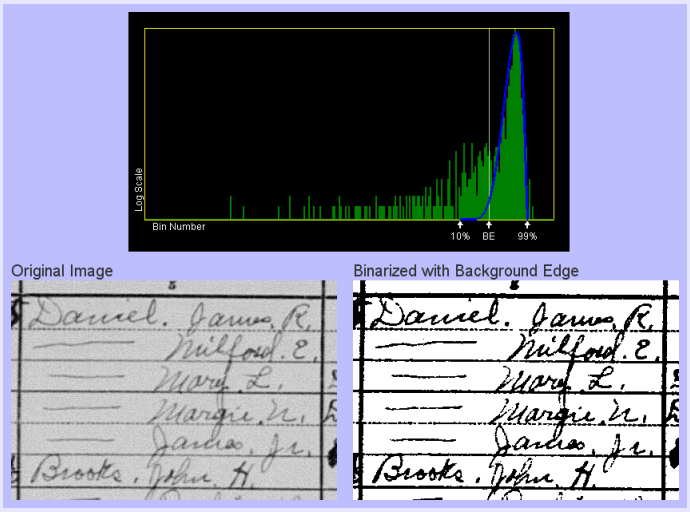
A Solution

- Fit a Kumaraswamy Distribution to White pixels
- Select a Confidence Level
- Find Corresponding Pixel Level

But how can we find the lower edge of the background automatically?

We can fit an analytic function, in this case the Kumaraswamy Distribution, to the background data. This means that we transform some histogram bins to the zero to one range required for the distribution, and then find control parameters of a Kumaraswamy distribution which produce the best fit to the range of intensities in the “white” background. Then, having a specific distribution, we find the fitted distribution value which is less than 99% of the “white” background pixel intensities. This value, after suitable transformation back to pixel intensities, becomes the desired background edge threshold. I will describe the Kumaraswamy Distribution later. Now let’s see how well it works.

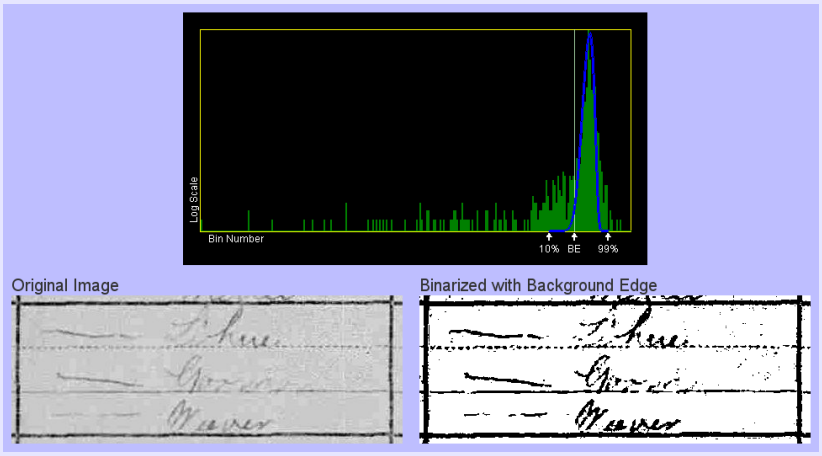
Sample 1



Here is the example I just showed with the computed Kumaraswamy distribution superimposed. For many images there are less than 10% of the pixels that are “black” or “gray.” So I use the 10 percentile and the 99 percentile values from the histogram to map to the zero to one range of the distribution.

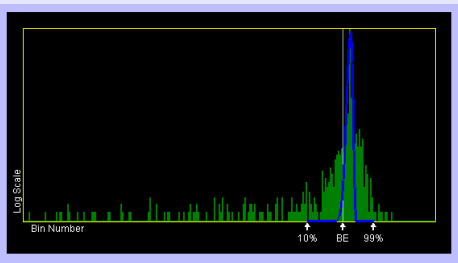
The 1 percentile value of the distribution maps to a value very close to the threshold which was manually chosen earlier. The binarized results look very similar.

Sample 2

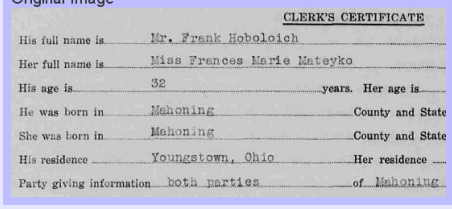


This is a low contrast handwriting example. Once again the 10 percentile and the 99 percentile histogram values are used to map to the zero to one range. The resulting binarized image is sort of OK.

Sample 3



Original Image



CLERK'S CERTIFICATE

His full name is Mr. Frank Hoboloich

Her full name is Miss Frances Marie Mateyko

His age is 32 years. Her age is

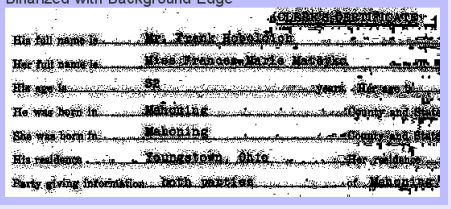
He was born in Mahoning County and State

She was born in Mahoning County and State

His residence Youngstown, Ohio Her residence

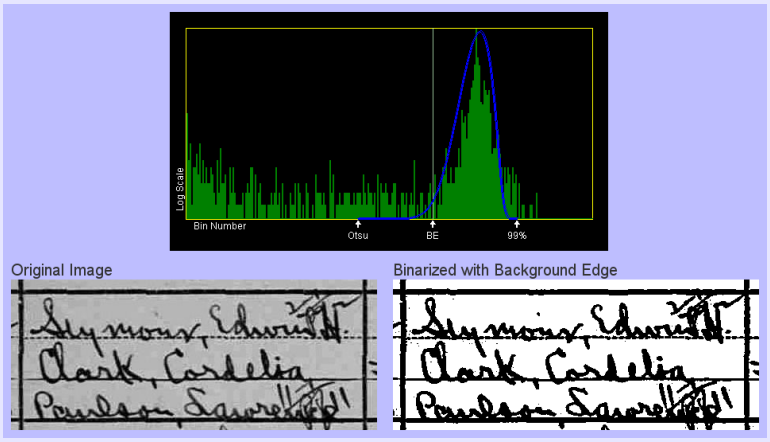
Party giving information both parties of Mahoning

Binarized with Background Edge



This is an example where the white values in the histogram do not match up with a Kumaraswamy distribution shape. Too many of the pixels have exactly the same value. The resulting binarized image leaves something to be desired. The background edge method only works well when the assumptions are valid.

Sample 4



This is an example where there is a high percentage of black/gray pixels. In this case the 10 percentile value does not work for the lower limit of the mapping, so I use the Otsu threshold as the lower bound. In the previous examples the Otsu threshold was lower than the 10 percentile point, so I use whichever is higher.

The resulting binarized image matches quite well, but a couple of loop centers got blackened.

Why the Kumaraswamy Distribution?

- It is finite
- It is computationally easy
- It looks like the actual data

There are a great many possible functions that could be used to fit the background data. Why did I choose the Kumaraswamy Distribution.

Unlike almost all probability distributions, the Kumaraswamy distribution is defined over the finite interval zero to one. There are no infinities getting in the way.

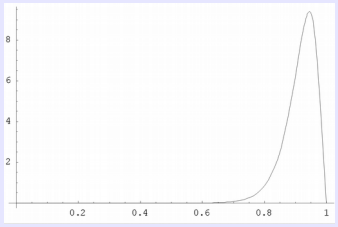
The frequently cited example of another finite interval distribution is the Beta Distribution. It is defined in terms of Gamma functions and everything about it is a computational nightmare. As shown later the Kumaraswamy distribution is easy to compute.

When one looks at histograms of these type of images, the big “white” peak looks like the Kumaraswamy distribution Probability Density Function.

The Kumaraswamy Distribution

- Cumulative Distribution Function $1-\left(1-x^{a}\right)^{b}$
- Probability Density Function $abx^{(a-1)}\left(1-x^{a}\right)^{(b-1)}$
- Inverse CDF $1-\left(1-y^{(1/b)}\right)^{(1/a)}$
- Constraints $x\in[0,1],a>0,b>0$

PDF of a=20, b=3



So what is the Kumaraswamy Distribution? The Kumaraswamy distribution is defined based on a cumulative distribution function and has two parameters “a” and “b” which must be greater than zero. The derivative of the CDF produces the probability density function. The CDF is easy to invert and this inverse is the basis for determining the background edge threshold. The range of permissible values for “x” or for “y” is zero to one.

Fitting to Obtain the Parameters

- Start with the quartile values 0 < q1 < q2 < q3 < 1
- Get the initial approximations

$$a_0 = \frac{\log\left(1 - 1/2^{2/7}\right) - \log\left(1 - 3^{1/7}/2^{2/7}\right)}{\log\left(q3/q1\right)} = \frac{1.5}{\log\left(q3/q1\right)}$$

$$b_0 = \frac{\log(2)}{\log\left(1/\left(1 - q2^{a_0}\right)\right)}$$

- Iterate new values until close enough

$$a_{i+1} = \frac{\log\left(1 - (1/4)^{1/b_i}\right) - \log\left(1 - (3/4)^{1/b_i}\right)}{\log\left(q3/q1\right)}$$

$$b_{i+1} = \frac{\log(2)}{\log\left(1/\left(1 - q2^{a_{i+1}}\right)\right)}$$

How are the “a” and “b” parameters determined so that the resulting function best fits some data. Starting with a histogram of the intensity values which occurred in an image, the first step is to trim off those “black” values that are definitely not part of the “white” background, and then transform the results to the range zero to one. Then the quartile values are determined from what is left. These values are then used iteratively to obtain progressively better estimates of the “a” and “b” parameters. This process seems to converge to within about 1% within 3 to 5 passes through the equations.

Conclusion

- The background edge threshold method seems to work quite well.
- It is easy to implement
- It may be useful for you

Having used this method of finding the background edge threshold on many thousands of images from the US 1930 Census, the background edge method described here seems to do a good job in at least 98% of the cases.

It is not difficult to program nor costly to compute.

Because it has been useful for me, I present it for your consideration.