Otsu Method & Adaptive Thresholding Implementation in MATLAB

Christos Panourgias

Contents

1 Introduction

Goal : Set a threshold, $T \in \{0, 1, \ldots, 255\}$, of which each pixel, $I(n, m)$, of a graylevel image, I, will be converted to binary through the transformation below

$$I_{binary}(n, m) = \begin{cases} 255 & I(n, m) \geq \mathrm{T} \\ 0 & I(n, m) < \mathrm{T} \end{cases} \quad \forall\ (n, m) \in N \times M$$

Where N is the number of lines and M is the number of columns in the image I

Thus a **binary** image, I_{binary}, is created with the same dimensions of image I.

2 Otsu Method

2.1 Description of the method

The threshold T is splitting the image to two classes, namely:

$$C_0 = \{(n, m) \mid I(n, m) \in \{0, 2, ..., T - 1\}\ \}$$

$$C_1 = \{(n, m) \mid I(n, m) \in \{T, T + 1, ..., 255\}\ \}$$

According to Otsu's method the threshold, T, will be equal to a value $t \in \{1, 2, ..., 255\}$ which will separate, as much as possible, the distribution of the classes C_0, C_1, therefore t will minimize the within-variance, σ_W^2, of the two classes.Equivalently, it suffices to find $t \in \{1, 2, ..., 255\}$ which maximizes the variance between the two classes, σ_B^2
It is known that

$$\sigma_W^2 = \omega_0(t)\omega_1(t)(\mu_0(t) - \mu_1(t))^2$$

Where $\omega_0(t) = \sum_{i=0}^{t-1} p_i, \quad \omega_1(t) = \sum_{i=t}^{255} p_i$, are the probabilities of classes C_0 and C_1 respectively and $\mu_0(t) = \frac{1}{\omega_0(t)} \sum_{i=1}^{t-1} ip_i$, $\mu_1(t) = \frac{1}{\omega_1(t)} \sum_{i=t}^{255} ip_i$ is the mean value of the distribution of class C_0 and C_1 respectively. Let $p_i = \frac{h(i)}{|N \times M|}$ be the probability that a pixel has the graylevel value $i \in \{0, 1, ..., 255\}$
According to the above the threshold is given by the following expression

$$T = \underset{t \in \{0,1,\ldots,255\}}{\operatorname{argmax}} (\sigma_B(t))$$

2.2 Implementation in MATLAB

2.2.1 Functions

- **h(rows, columns, image, i) = num_pix**
  This function takes as inputs the image columns, the image rows, the image, and a graylevel i. Returns the number of pixels in the image that have the graylevel value i.

  1. Initialize variable num_pix to 0
  2. Using two nested for-loops all pixels in the image are traversed.
  3. A if expression checks if the current element is equal to i
  4. In case $I(n, m) = i$, 1 is added to the variable num_pix
  5. Returns the variable num_pix

- **p(rows, columns, image, i) = prob**
  This function takes as inputs the image columns, the image rows, the image, and a graylevel i. Returns the probability that a pixel located in the image has the graylevel value i.

  1. Calculates the product of the columns and rows of the image and stores it in the variable N
  2. The division $\frac{\text{h(rows, columns, image, i)}}{\text{N}}$ is calculated and stored in the variable prob
  3. The variable prob is returned

- **image_hist(image, image_name, T) = hist**
  This function takes as inputs the image, the filename of the image, and the threshold, T. Returns the histogram of the image with the threshold drawn as a single vertical red line.

  1. The list hist is initialized and the variables rows and columns store the dimensions of the image.
  2. Two nested for loops are created where the first loops through the values $0, 1, 2, ..., 255$ in the variable gl and the second loops through the values $1, 2, ..., h(gl)$ in the variable k. Finally for these iterations (which have a count of h(gl)) the gl value is stored in the hist list
  3. Using the histogram function of MATLAB calculates the histogram of the hist list and then adds the axis names and the chart name.
  4. The threshold is stored in the variable txt with its value in string, the list y is initialized and with a for loop the values h(i) for each i
  5. Adds string "T = t" to the graph using the text() function of MATLAB where t is the threshold value found by the method by Otsu
  6. Using the line() function of MATLAB a vertical line is added to the graph at the position where the threshold T has been calculated

- **binarize(image, T) = bi**
  This function takes as inputs the image and the threshold. Returns the binary image generated according to the transformation in the algorithm description.

  1. The image is stored in the binary_image variable. The rows and columns variables store the dimensions of the image
  2. Traversing the pixels of the image with two nested loops in the x, y variables
  3. If the pixel at coordinates (x,y) has graylevel less than or equal to T then it is given the value 0. If it has graylevel greater than T then it is given the value 255
  4. Store the binary image in the variable bi and return the variable bi

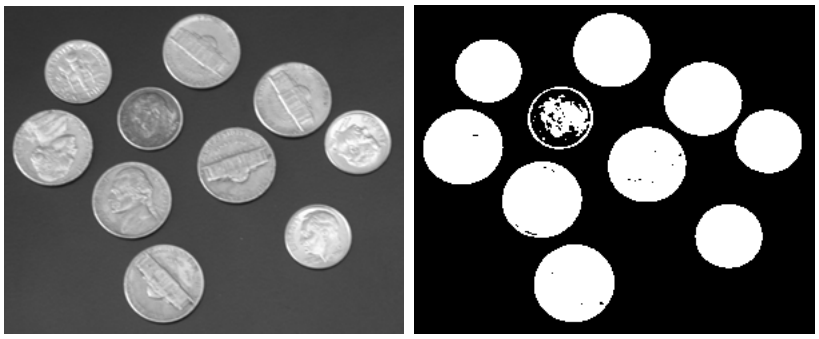
2.2.2 Main Program

1. Initializes the path from which to retrieve the original image, the name to save the binary image to, and the name to display in charts.
2. The image is read with the imread() function of MATLAB and the dimensions of the image are stored in the rows and columns variables using the size( )
3. The lists prob_list and i_pi_list are initialized where the values of the probabilities, p_i and the probabilities multiplied by the graylevel value will be stored textlatini, ip_i. The list sigma2_b_list is also initialized, which will contain the values of $\sigma_B^2(t)$ for each t
4. A for loop is created that loops through the values $0, 1, 2, ..., 255$ in the i variable. then the p_i and ip_i values are calculated and stored in the prob_list and i_pi_list lists respectively.
5. A for loop is created which runs through the values $0, 1, ..., 255$ in the variable t. Then $\omega_0(t), \omega_1(t)$, $mu_0(t), \mu_1(t)$ and
   are stored in the variables w0, w1, m0, m1 respectively.
6. Using the above $\sigma_B^2(t)$ is calculated for each t and stored in the list sigma2_b_list
7. Using the max function of MATLAB calculates the maximum value of the list sigma2_b_list and then using the find() function calculates t to which corresponds the maximum value of the list sigma2_b_list. This t is stored in the variable T
8. Using binarize() stores the binary image in the binary_image variable and saves it as an image using the imwrite() function

9. Three figure snapshots are created and the images and histogram are displayed in them, using the imshow() and image_hist() functions.

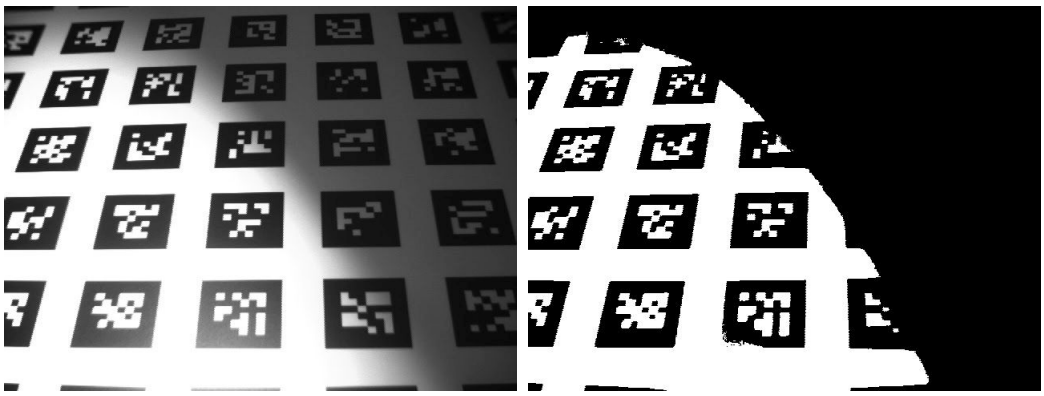
2.3 Results

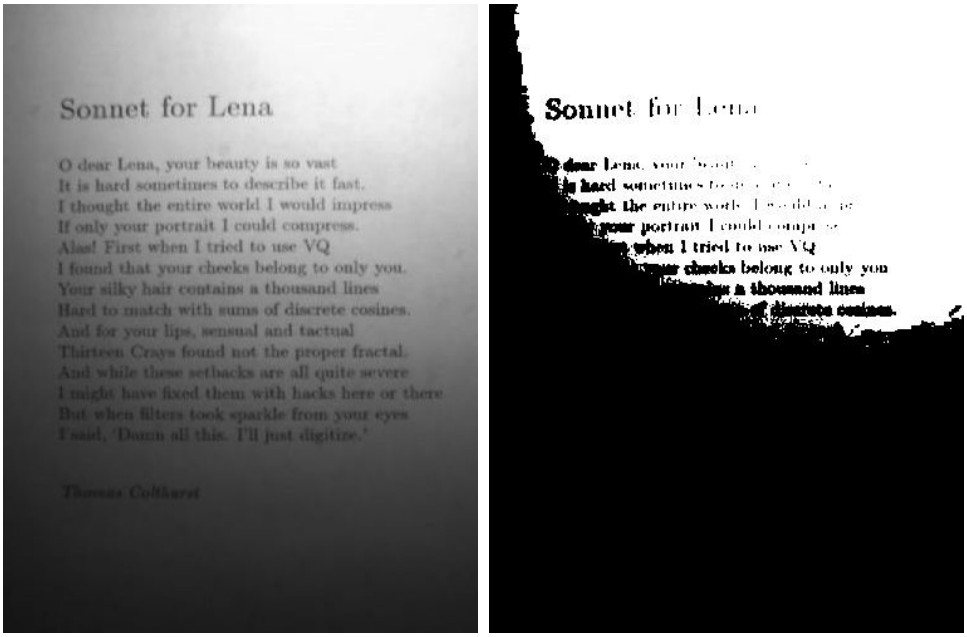
In the images above, Otsu's method separates the original image information well. This happens because the shadows in the specific images are distributed so as not to change the gradation of brightness in the information points that are to be extracted.

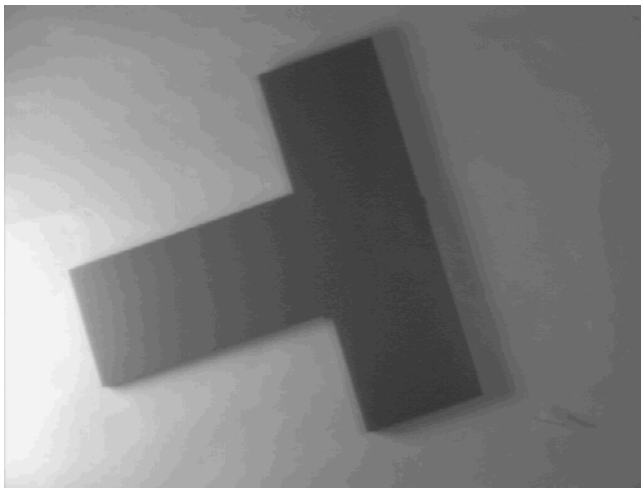


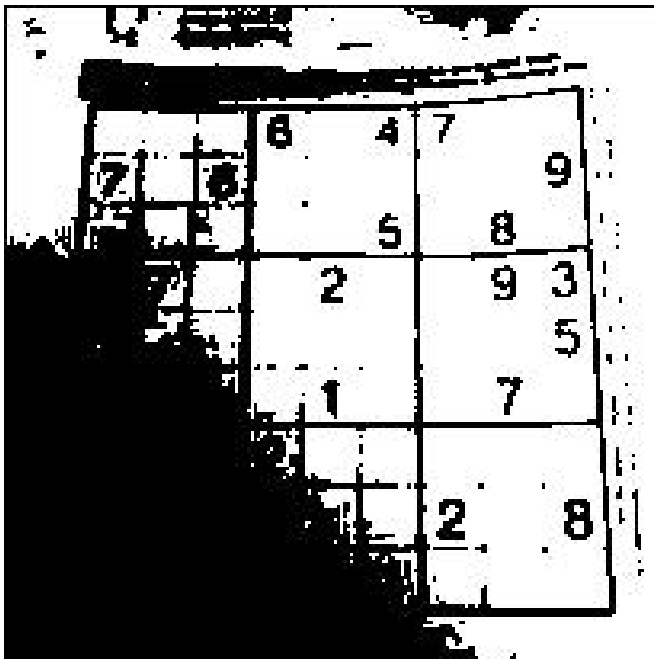
It is observed that Otsu's method succeeds in separating all the coins in the image except one, which has a darker color than the rest in the original image.

In the following images it is observed that in the cases where the information is obscured so that the gradation in brightness changes, the method of Otsu yields moderate results.




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





3 Adaptive Thresholding Method

3.1 Method Description

In this method, a mask, with radius R, is used on the image which calculates the threshold of the area to which it is applied depending on the technique chosen. In this part, three techniques will be used, namely

Mean value : $t_i = mean(R_i) - C$
Median: $t_i = median(R_i) - C$
Intermediate value: $t_i = \frac{min(R_i)+max(R_i)}{2} - C$

Where R_i is the set of graylevel values of the pixel of the given window and C is a transposition constant of the threshold so that the brightness is appropriately adjusted.

3.2 Implementation in MATLAB

3.2.1 Functions

- **binarize(image, T_list) = b** This function takes as inputs the image and the list of all thresholds from each window. Returns the binary image produced by the above method.

  1. Saves the image to the b variable. Initializes the counter counter_T_list to be used as a pointer to the list of thresholds. Image dimensions are stored in rows and columns variables
  2. Two nested for loops are created which iterate through the columns and rows of the image in the variables i, j respectively.
  3. Checks if the pixel at the current coordinates has a smaller or larger graylevel than its corresponding threshold. In
     case it is smaller then it is replaced by graylevel 0 otherwise it is replaced by graylevel 255
  4. returns the binary image b.

3.2.2 Basic Program

1. The path from which the image will be used in the image_file variable is selected. Saves the name of the image in the variable image_name
2. The image is read with the function imread(). The
   dimensions are stored in the rows and columns variables. The total number of pixels is calculated and stored in the variable N. Initialize the list t_list which will contain all the thresholds.
3. The side of the mask, R, and the constant C are selected.

4. Rounding the variable R and using
   padarray() adds padding to all sides of the image which is the same size as R and is a copy of the internal columns or rows of the image (this is achieved with the 'replicate' parameter.

5. Two nested for loops are created which iterate through the rows and columns of the image in the variables x and y respectively. Then the empty list win is created.

6. In the image with padding the new center of the pixel to be checked has coordinates (x+R, y+R) thus creating two nested for loops which span the window on the x axis and the y axis (starting at x and stopping at x + 2R and starting at y and stopping at y + 2R

7. The graylevel values of the coordinates in the win list are stored. These are the window values, so the threshold will be determined by their average value.

8. The win window is converted from an array to a vector using the reshape() function.

9. The mean/median/median value is calculated, the constant C is subtracted from it, and the result (which is the current threshold) is stored in the list t_list

10. binarize() is called which takes the image and the list of thresholds and stores the return of the binary image in the variable binary_image

11. Using imshowpair() of MATLAB the images are shown.

12. Images are saved using imwrite() of MATLAB

3.3 Results

Mean

Median

Intermediate

The values for the R,C parameters in each
case turned out to have the following values :

Mean : $R = 50$, $C = 15$
Median : $R = 80$, $C = 11$
Intermediate : $R = 80$, $C = 20$

The best technique for this image is the average value because it shows more details (eg the photographer's glove). Second best is the intermediate price technique and last is the average technique. It is observed that for the average value technique, which also has the best result, it was enough to use a mask with side $R = 50$, while in the other techniques a larger mask side was needed so that better results were achieved, which implies a greater time cost .

Mean

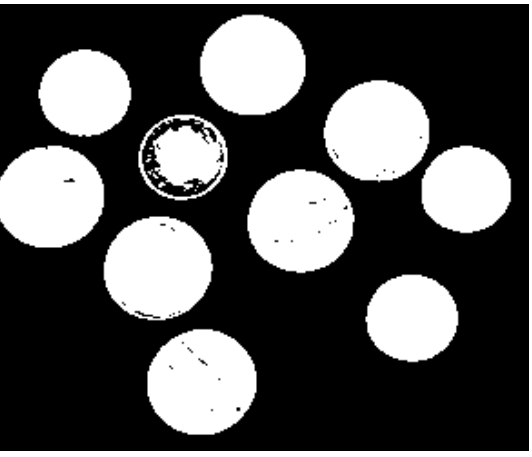
Median

Intermediate

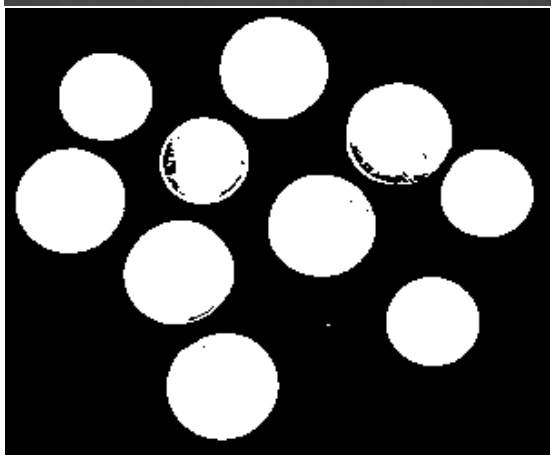
The values for the R,C parameters in each
case turned out to have the following values :

Mean : $R = 40$, $C = 5$
Median : $R = 40$, $C = 5$
Intermediate : $R = 50$, $C = 0$

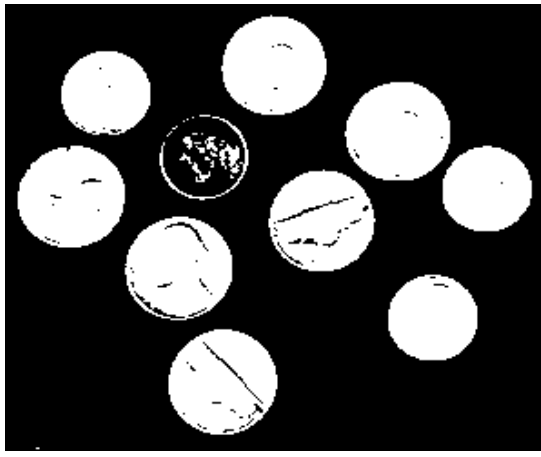
The best technique is the intermediate value, because it separates background better than the others. However, the other techniques also give satisfactory results in terms of face separation. For the mean and median value, exactly the same mask sizes and constant C were used, while for the intermediate value, a larger mask size and zero constant were used. Due to the larger size of the mask, a higher time cost is found.



Mean



Median

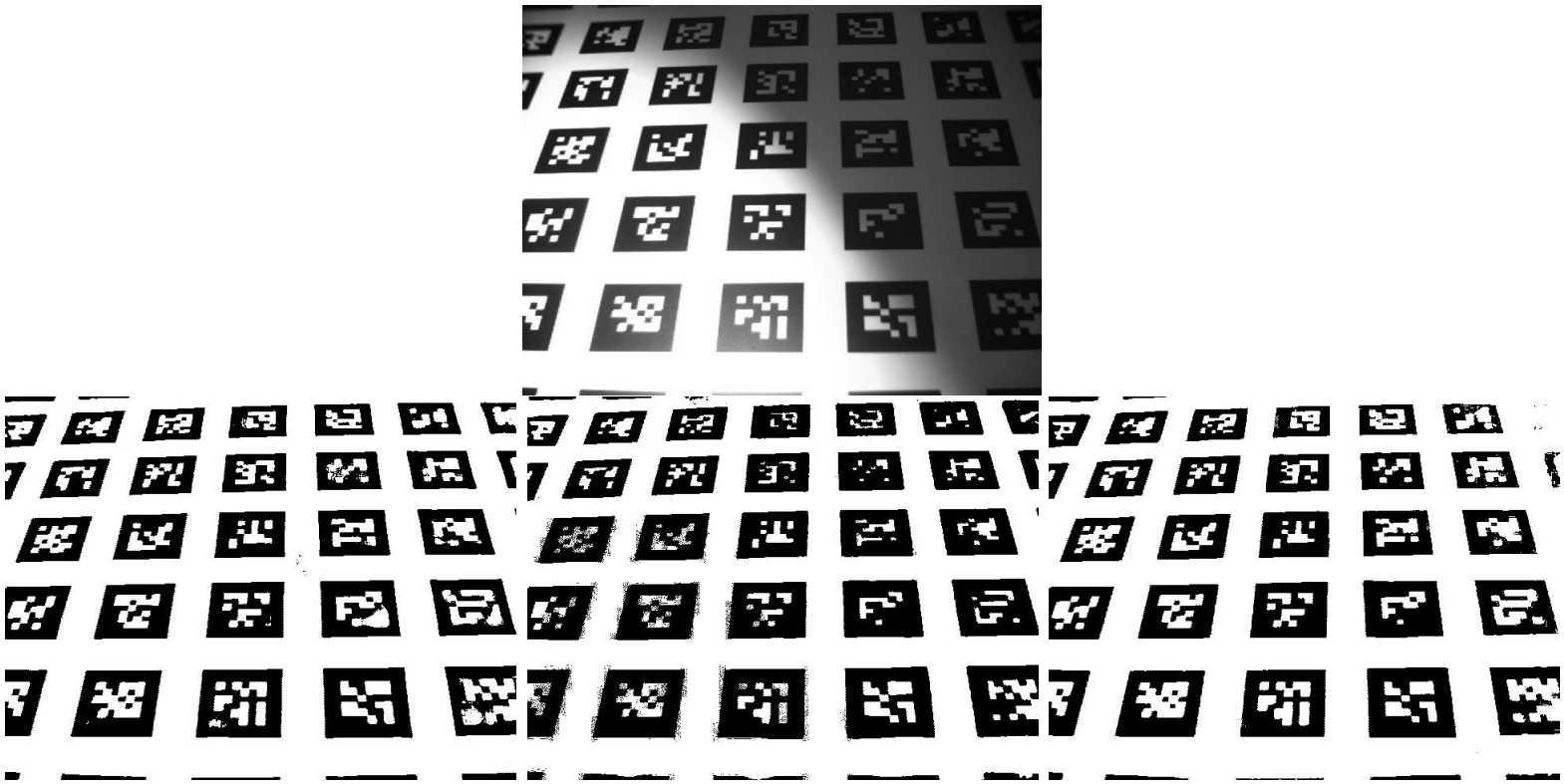


Intermediate

The values for the R,C parameters in each
case turned out to have the following values :

Mean : $R = 40$, $C = -10$
Median : $R = 45$, $C = -6$
Intermediate : $R = 40$, $C = -10$

The best technique is thresholding with the medium, because it makes even the darkest coin completely white. Then thresholding with the mean value seems to distinguish the dark coin, but not as completely white. Thresholding with the intermediate price hardly separates the dark coin. It is observed that despite the better performance of the median technique, the mean and median techniques have the same mask size and the same C constant value. The intermediate value despite larger mask technique did not perform well.

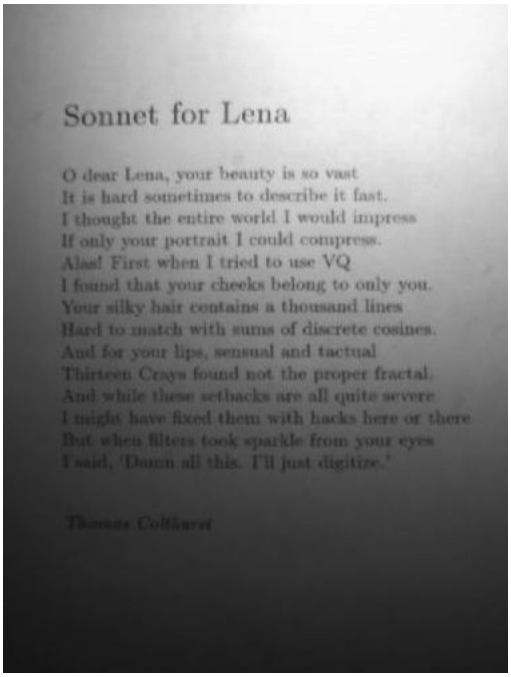


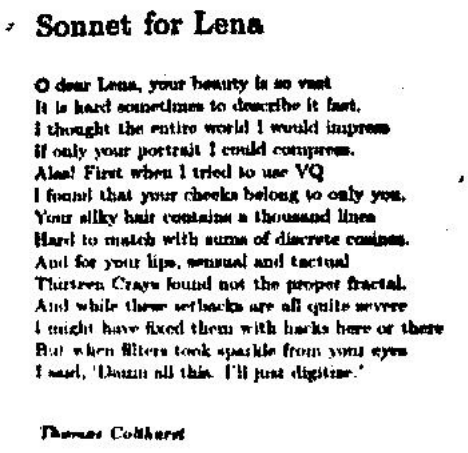
Mean **Median** **Intermediate**

The values for the R,C parameters in each
case turned out to have the following values :

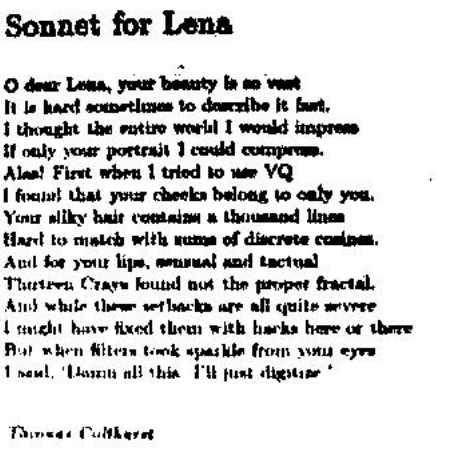
Mean : $R = 15$, $C = 7$
Median : $R = 60$, $C = 40$
Intermediate : $R = 25$, $C = 20$

The intermediate value technique produces the best result because it better analyzes the shapes inside the squares (eg lower right square). The mean value technique produces coarser shapes within the squares, but gives fairly good results. The average technique analyzes the image but does not perform well in analyzing the shapes within the squares. It is observed that the mean value technique has the smallest mask size and yet has high accuracy in its performance. The intermediate value technique also has a relatively small mask size but produces darker results than the mean value technique and thus the C constant takes on a larger value. Finally the medium technique requires a large mask size to return good results, it also returns quite a dark image and thus needs a high C value

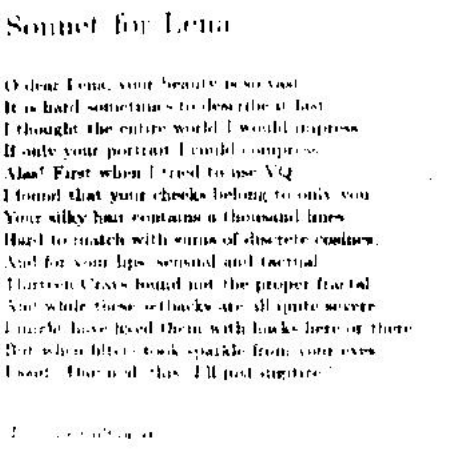




Mean



Median

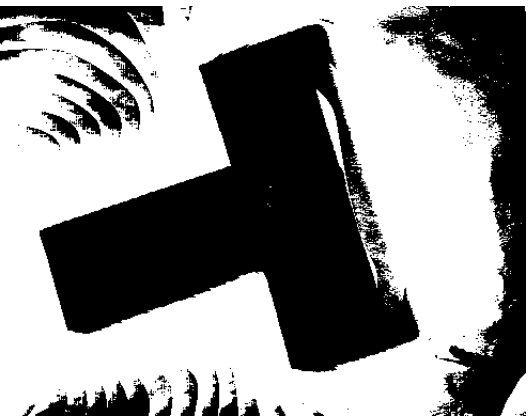


Intermediate

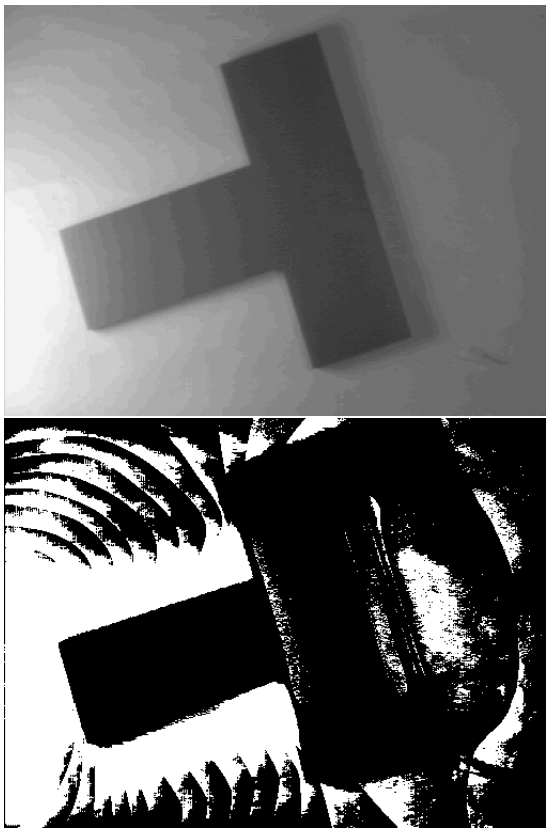
The values for the R,C parameters in each
case turned out to have the following values :

Mean : $R = 18$, $C = 7$
Median : $R = 18$, $C = 7$
Intermediate : $R = 25$, $C = 12$

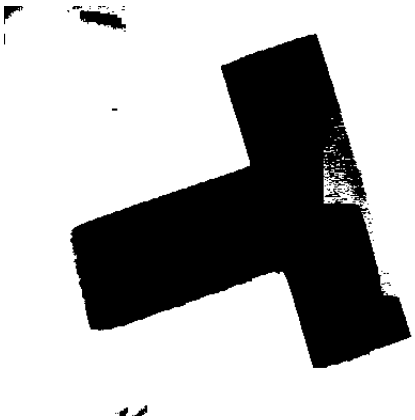
In this image the mean value technique performed best
having the same mask size and C value as the mean technique. The medium technique returns less bold letters, but the result is quite satisfactory. The intermediate value technique requires a larger mask size to return appreciable results, The brightness on the letters cannot be corrected by decreasing the C value because it then returns a darker background.



Mean



Median



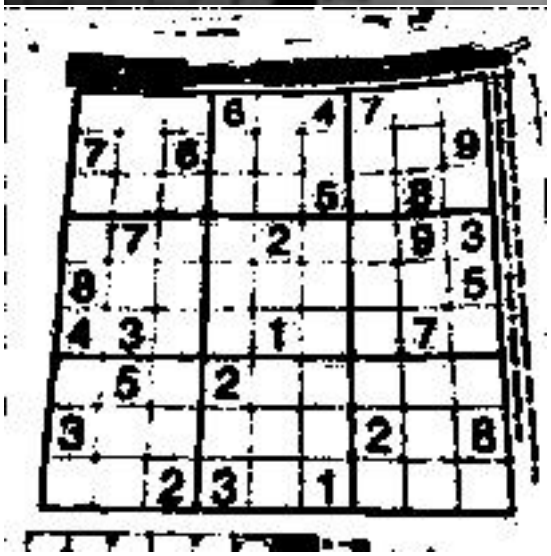
Intermediate

The values for the R,C parameters in each
case turned out to have the following values :

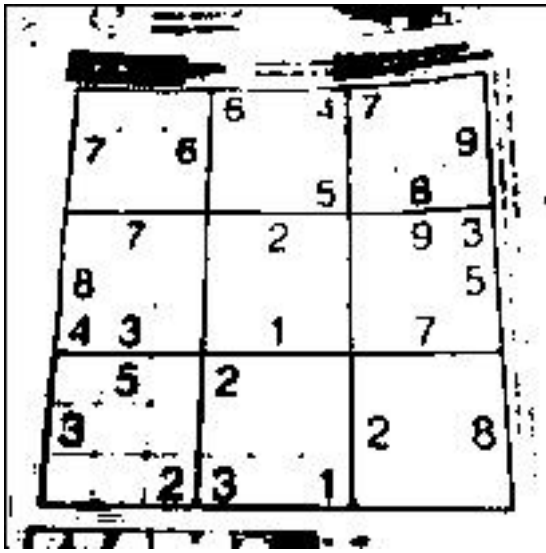
Mean : $R = 70$, $C = 0$
Median : $R = 70$, $C = 0$
Intermediate : $R = 70$, $C = 2$

The intermediate value technique performs the best in the results by a wide margin because it separates the tool from the background better than the others. The averaging technique separates the tool but there is shadow interference in the image. The medium technique presents difficulty in separating the tool from the background. All techniques have the same mask size and the constant C is zero in the mean and median techniques while in the intermediate value technique we have a constant $C = 2$. It is observed that this image is more sensitive to changes of the C value for it and it is difficult to correct the shadow interference in the binary image.

Mean



Median



Intermediate

The values for the R,C parameters in each case turned out to have the following values :

Mean : $R = 50$, $C = 12$
Median : $R = 18$, $C = 7$
Intermediate : $R = 50$, $C = 7$

The average technique produces the best result because it separates the lines between the numbers, while the other two techniques do not produce the same result. It is observed that the average technique uses a smaller size mask than the other techniques so as to produce this result. mean and median techniques have the same mask size and little difference in the C constant.

Adaptive thresholding produces very satisfactory results but due to the nested for loops the larger the mask used the greater the time cost. Therefore there are cases where it is more beneficial to use the method of Otsu (e.g. the photo cameraman or lena) which will produce quite satisfactory results and with little time cost.