California State University
Channel Islands

COMP 510 - Advanced Image Processing & Analysis

Image Segmentation Methods

Report Compiled by
Rihan Stephen Pereira, Niranjan Pawar, Vasanthi Lingamdinne
MSCS Graduate 2018-2019

December 16, 2018

Contents

1 | Abstract

In this report, we introduce an exciting topic of Image Segmentation, discuss its role in imaging systems and articulate different approaches to segmentation beyond global thresholding. We briefly elaborate on different techniques towards segmentation followed by testing those approaches on a set of given inclass images and additional images chosen by our team which are diverse and as challenging as the inclass images. We strive not to compose a subjective implementation of algorithm which is not reusable on diverse set of segmentation tasks. Finally, we showcase the results of running those algorithm and conclude on evaluating segmentation methods.

2 | Introduction & Background

Image Segmentation is an interesting, important and active area of research and development. It has serious applications in medicine for e.g screening a human brain to detect an abnormality in early stages. The objective of segmentation is to divide an image into meaning regions. In other words, the task is to group pixels together based on certain characteristic for e.g color, shape, size, etc, such is a problem of segmentation. Human beings perceive image as a whole. For us, inferring information behind that image is a natural and intuitive task. On other hand, for machines, image is a matrix composed of numbers with varying intensities and therefore a machine tasked with segmentation is a difficult problem, mainly because of ambiguity in data. In this report, we describe basic segmentation techniques which are based on **intensity discontinuity** useful for edge detection and also **similarity of intensity values**. The above figure describes a typical imaging system which incorporates Image Seg-

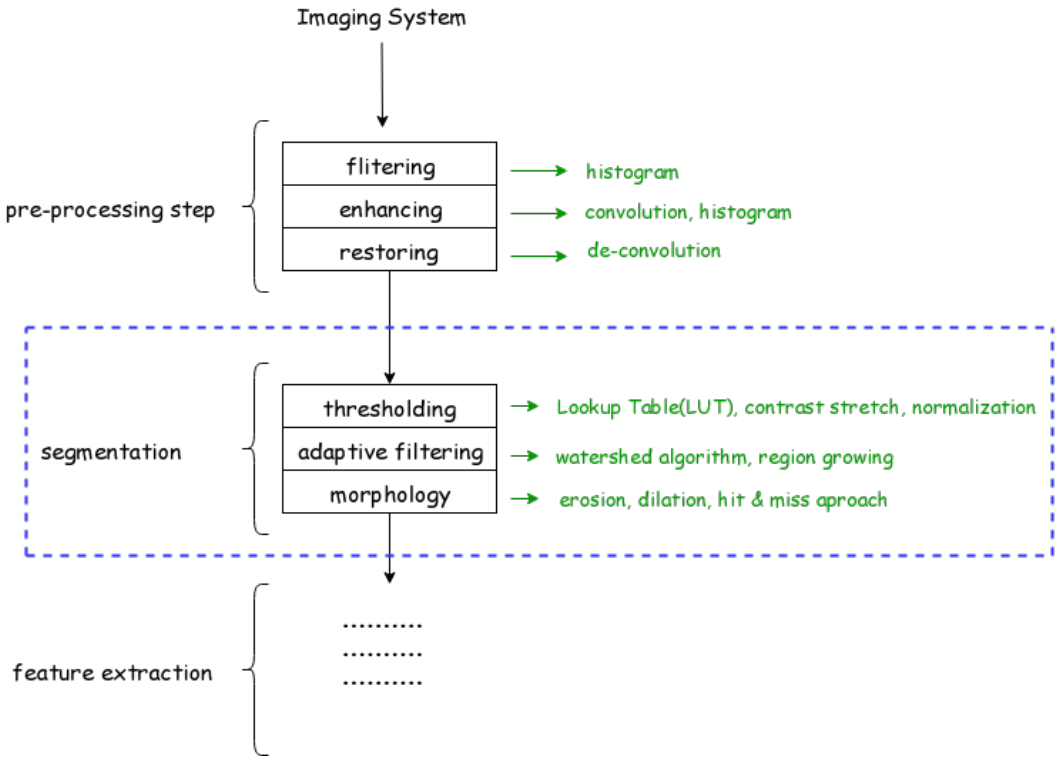


Figure 2.1: The role of Image Segmentation as a pre-requisite step in feature extraction

mentation component in its pipeline. It receives input feed as a pre-processed image, performs segmentation with ultimate goal as classification, image compression, restoring image or simply monitoring task. In this case, the next layer is feature extraction which makes it evident that system is trying to solve classification problem. As you

can see, in similar way, segmentation can be used as necessary addon based on overall purpose of the application.

3 | Segmentation methods based on Intensity Similarity

3.1 Otsus Segmentation

In computer vision and image processing, **Otsu's** method, named after **Nobuyuki Otsu**, is used to automatically perform clustering-based image thresholding, the reduction of a graylevel image to a binary image. The algorithm assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), it then calculates the optimum threshold separating the two classes so that their combined spread (intra-class variance) is minimal, or equivalently (because the sum of pairwise squared distances is constant), so that their inter-class variance is maximal. Consequently, Otsu's method is roughly a one-dimensional, discrete analog of Fisher's Discriminant Analysis. Otsu's method is also directly related to the Jenks optimization method.

Otsu's Algorithm:

1. Compute histogram and probabilities of each intensity level
2. Set up initial $w_i(0)$ and $u_i(0)$.
3. Step through all possible thresholds *t=1,....* maximum intensity.
   (a) Update w_i and u_i.
   (b) Compute $\sigma_b^2(\mathrm{t})$
4. Desired threshold corresponds to the maximum σ_b^2 (t).

Otsu's method exhibits the relatively good performance if the histogram can be assumed to have bimodal distribution and assumed to possess a deep and sharp valley between two peaks. But if the object area is small compared with the background area, the histogram no longer exhibits bimodality. And if the variances of the object and the background intensities are large compared to the mean difference, or the image is severely corrupted by additive noise, the sharp valley of the gray level histogram is degraded. Then the possibly incorrect threshold determined by Otsu's method results in the segmentation error. (Here we define the object size to be the ratio of the object area to the entire image area and the mean difference to be the difference of the average intensities of the object and the background).

3.2 Inclass Images

3.2.1 Image 10_27a

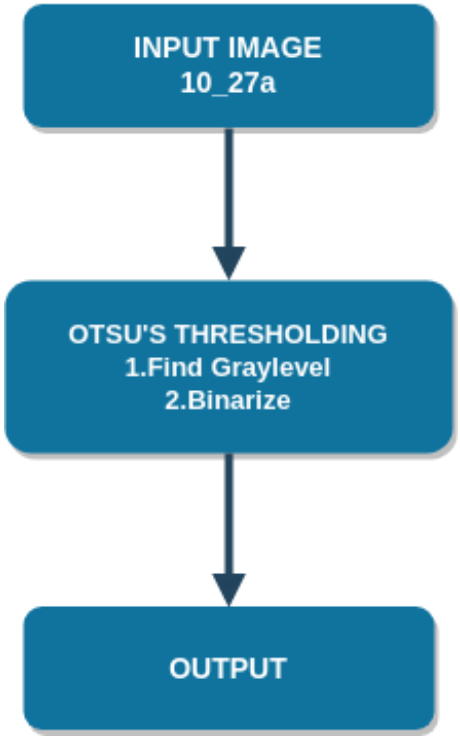


Figure 3.1: Image 10_27a workflow

This image was pretty straight forward and did not require any pre-processing. We used Otsu's Segmentation to obtain the output image.

3.2.2 Image 10_27d

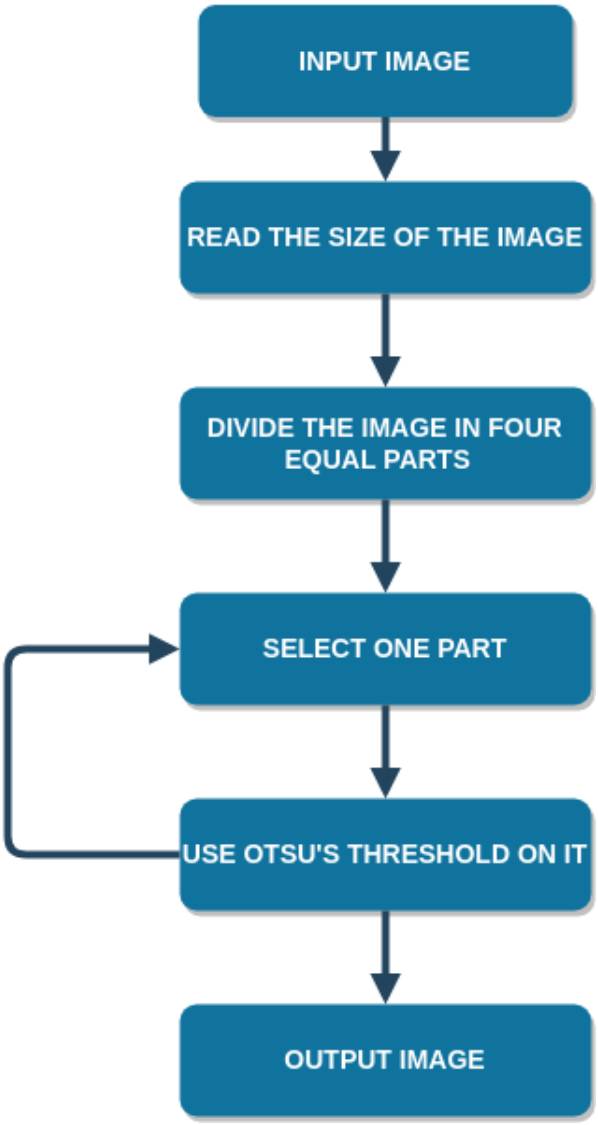


Figure 3.2: Image 10_27d workflow

The image had non-uniform background as well as foreground and it was positioned in such a way that applying Otsu's Segmentation would not produce the deserved image. The gray level of one side of the object was same as the gray level of the background on the other side. Thus the gray level threshold obtained from Otsu's Segmentation would not do justice with the image. It was not able to segment object from the background.

To overcome this problem, we strategize to break the image to four equal halves. The gray level difference in background and foreground of each of these blocks was sufficient enough to use Otsu's Segmentation on it.

This method gave us a good result with very less error.

3.3 Extra Images

3.3.1 Image group of coins

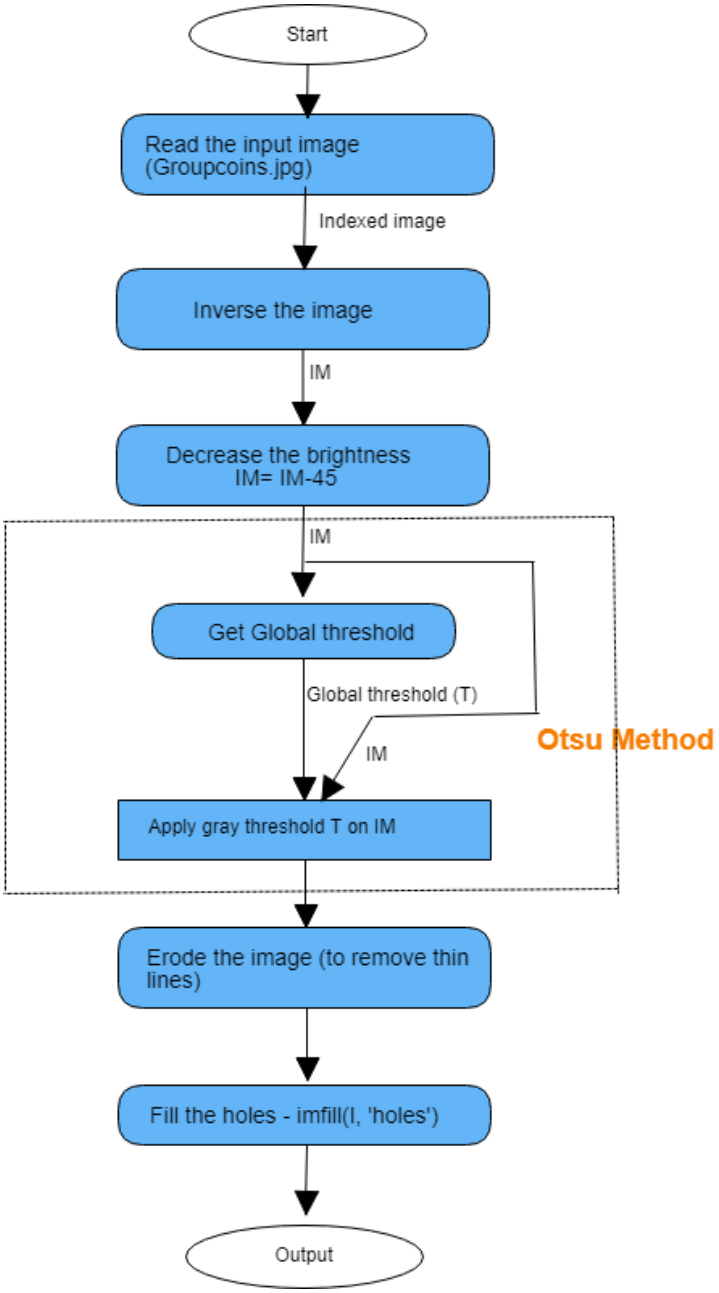


Figure 3.3: Image grp_of_coins workflow

For the image `group_of_coins.png`, our goal was to segment the coins from the background, In the image as the background is white, but we wanted objects in white and background in black, we applied inverse of the image and processed the resultant image with the further steps. As the inversed image is very bright and that might affect the output so we have decreased the brightness of image first and then we used Otsus method for image segmentation and were able to segment the coins from the background, but there was some extra lines on coins, so we eroded the image to remove

thin lines and finally we filled the holes if there are any as the coins are closed objects. With Otsus segmentation and eroding the image, we successfully segmented the coins from the background.

3.3.2 Image `jet_fighter_swarms`

This image has the objects in gray color and background is light color so we have first inversed the image as our goal was to have the objects segmented from the background with objects as white in color and background as black color. Then we decreased the brightness as some of the background pixels are bright after inverse. We then applied Otsus thresholding method and achieved the objects segmented successfully.

3.4 Adaptive Thresholding

Thresholding is the simplest way to segment objects from a background. If that background is relatively uniform, then you can use a global threshold value to binarize the image by pixel-intensity. If there's large variation in the background intensity, however, **adaptive thresholding** (a.k.a. local or dynamic thresholding) may produce better results.
Here, we binarize an image using the threshold adaptive function, which calculates thresholds in regions of size block size surrounding each pixel (i.e. local neighborhoods). Each threshold value is the weighted mean of the local neighborhood minus an offset value.

Adaptive thresholding is useful for segmenting variable background images. It changes the image threshold dynamically, as opposed to applying a monotonous threshold to all the pixels in the image. Each pixel is considered to have a local neighborhood of n x n pixels, the mean or median of which is used to calculate the local threshold value. The pixel that we are taking into consideration is set to black or white depending on whether it falls below or above this local threshold value. The technique can be improved by employing an additive constant, which can be subtracted from the calculated mean or median of the `n x n` neighborhood. The technique is successful at adapting for variations in the background if the `n x n` neighborhood is of an optimal size.

3.4.1 Image `son1.gif`

Our Goal is to segment the text from the background from the image son1.gif. But the image son1.gif has a varying background, a single threshold is unsuitable for the entire image field for separating the text from the background. Instead, it is preferable to implement some form of per-pixel thresholding, in which a different threshold is computed for every pixel (i.e. a "threshold image"), so we have choose adaptive thresholding method for this image segmentation.

As we found that, a single threshold (Otsus method) fails segmenting the text for this source image. Instead of using a single number or global threshold, we established the threshold for each pixel by taking mean average of the brightness values in its neighborhood (minus a constant value of `0.025`). For this image we have taken `(11x11)` neighborhood pixels to calculate average mean threshold. The average threshold of

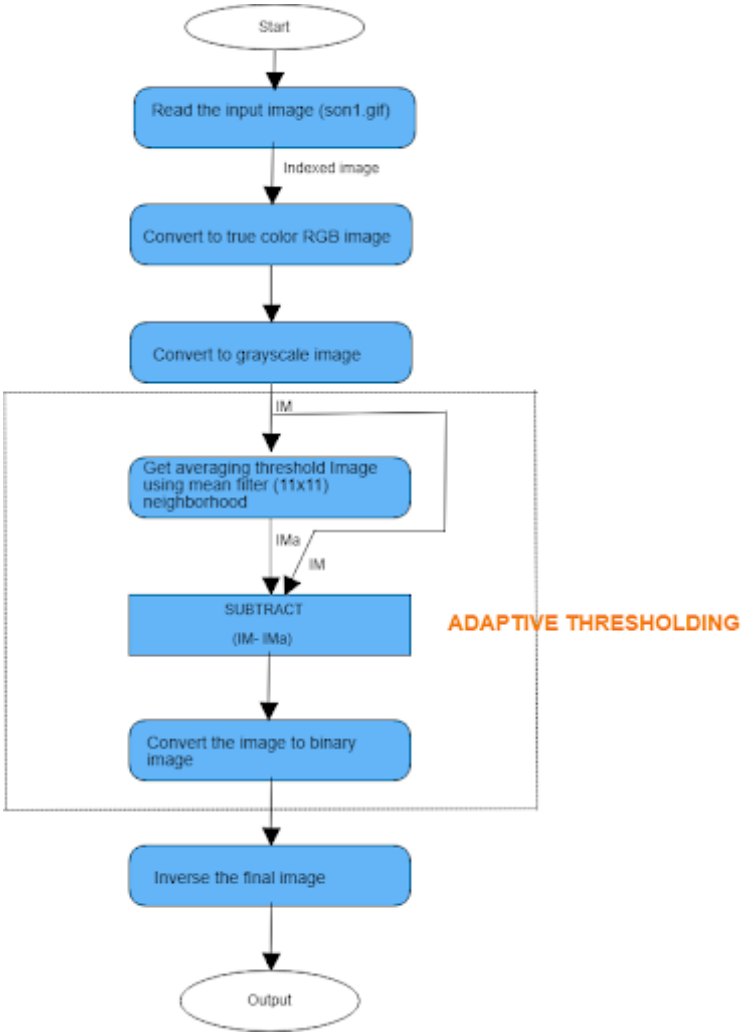


Figure 3.4: Image son1 workflow

the image is then subtracted from the input gray level image and final segmented image is achieved. With this method we were able to successfully segment the text from the background.

3.4.2 Image `papir`

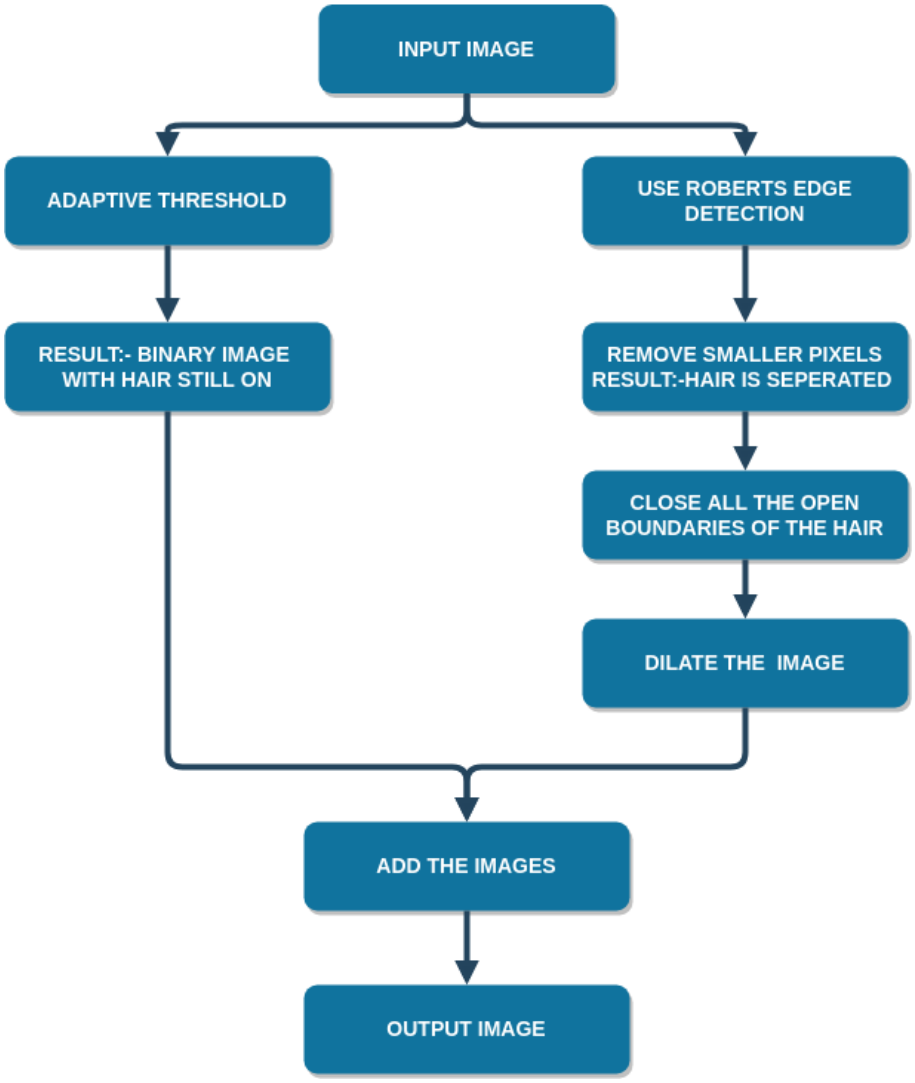


Figure 3.5: Image papir workflow

Objective for this image was

1. To remove the hair from the image.
2. To alter the uneven illumination in the background.

We started the process of hair removal using Roberts edge detection. We then removed the the small pixels to isolate the hair in the image. Then we performed bwclose on the result image to close any open boundaries. After which we dilated the image which was basically done to thicken the size of the hair. Now we had image with black background and clear thick hair in white. We solved the uneven illumination problem by using adaptive thresholding, which produced a binary image with with white background and text, hair in black.
We added both the image producing a binary image without the hair.

4 | Segmentation methods based on Intensity Discontinuity(edge detection)

Segmentation techniques based on Intensity Discontinuity are used to detect the boundary of the system and clearly if the boundary is closed then that boundary defines a region i.e the pixels that are inside the boundary.

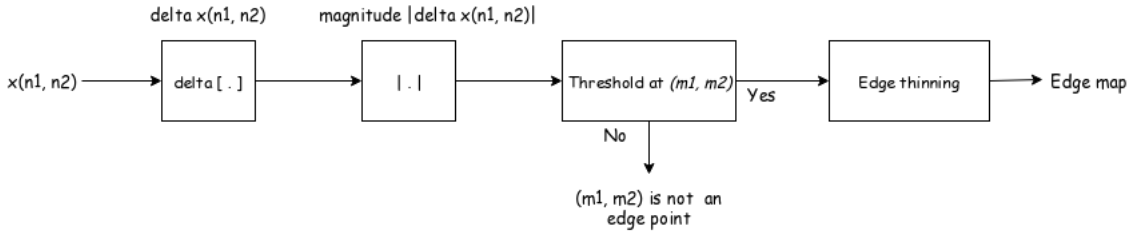


Figure 4.1: edge detection block diagram

Above figure is basic building block of edge detection algorithm. $x(n_1, n_2)$ is an input image of which the edges we need to find. The gradient for this image is what needs to be found, so at this point we have available a 2D array and each entry of the array is a vector. After this step the magnitude of the gradient is found and exactly at this point we discard the directionality of the edges and just keep its strength so now we have the length of the vector and each entry in the 2D array is a scalar. Next, the numbers in the array are compared against the threshold. If the number at location for e.g (m_1, m_2) is smaller than the threshold, then no edge exists at that point. If, however, the value's greater than the threshold, then $(m1, m2)$ pixel belongs to an edge. Since identifying edge pixels is based on this thresholding operation, very often it ends up with thick edges. Edges, multiple pixels belonging to an edge and therefore typically an edge thinning algorithm. As the name implies, the width of those edges are thinned, thus they becomes smaller, maybe a couple of pixels and finallly we have the edge map of the original image $x(n_1, n_2)$.

4.1 Sobel Operator

Sobel operator is a gradient operator which has emphasis in edge detection in images. It is basically a convolution filter of mask size `2 x 2`, `3 x 3` which is used to compute image derivative. It has a separable properties. Sobel operator is good at detecting stronger edges but the weaker edges are faint and disappear after thresholding.

4.1.1 Image `group_of_coins`

We executed Sobel operator on this image. The process starts by turning the color image into graylevel image followed by smoothening image using a gaussian filter with standard deviation of 1.0. After this, we extract the threshold value of the image used by sobel operator. Then we multiply the initial threshold value by `fudge_fx` which holds float value and run the sobel operator on that image again which indeed gives us the best of what sobel operator has to offer. The end result give us just 2 complete boundary of coins while the rest are incomplete boundaries. We discuss a remedy for solving this problem in the summary section.

4.1.2 Image `another_group_of_coins`

For this image we applied the same step as the previous image. The end result is also incomplete boundaries which we address the same in the summary section.

4.1.3 Image `pillars_of_hercules`

For this image, the only change is to adjust `fudge_fx` set to 0.7 to get the desired outcome.

4.2 Canny Edge Detector

Canny Edge detector does better **edge linking** compared to sobel operator. The four fundamental steps of canny edge detector are - **Canny Edge Algorithm**

1. Smooth the image using gaussian filter σ.
2. compute the gradient magnitude and angle images.
3. apply non-maximal suppression to the gradient magnitude image.
   (a) let d_1,d_2,d_3,d_4 be four basic edge directions(horizontal, -45 degrees, +45 degress, vertical).
   (b) At every point (n_1, n_2) in $\alpha(n_1, n_2)$:
      i. find the direction d_k that is closest to $\alpha(n_1, n_2)$
      ii. if the value of $M(n_1, n_2)$ is less then at least one of its neighbors along d_k, ignore/supress it or else keep it.
4. finally use double thresholding (T_H, T_L).
   (a) Pixels with values greater than T_H are strong edge pixels
   (b) Pixels with value smaller than t_L are weak edge pixels
5. perform edge linking
   (a) Locate a pixel p belonging to a strong edge
   (b) Find weak pixels that are connected to p
   (c) append these weak pixels to p

4.2.1 Image `group_of_coins`

The process remains the same as discussed previously. The only difference is we use canny edge algorithm in place of sobel and set the fudge value to 0.6. The end result is much improved than sobel result.

4.2.2 Image `pillars_of_hercules`

For this image we have set the fudge value to .9. There are still some incomplete edge lines which need attention. The solution is discussed in summary section of the report.

5 | Morphological Operation

Morphological Operation is another technique for image segmentation. It consist of **erosion**, **dilation**, **opening**, **closing** operations, also, but important, **skeletonization**.

5.0.1 Image `baby_elephant`

First I binarize the image and then invert it. Then I make use of **erosion** to erode the surroundings, then we use `getsinglepixel` function to eliminate miniature isolated pixels. The process is still not complete. Futher we use `bpropagation` to allow constraint dilation. Futher, using `bskeleton` through the use of *looseendsaway* we isolate regions of the image and then use branch pixels to get the main object. The final result is elimination of background pixel thus focusing on foreground object. This is best we can get using morphology.

6 | Edge detection using maximum filter

Maximum filter can also be used for edge detection outcome. Like erosion in morphology operation, maximum filter erodes shapes on the image.

6.0.1 Image `group_of_coins`

For this image, I apply unsharpening mask and then perform image subtraction from the result of maximum filter and result of unsharpen mask

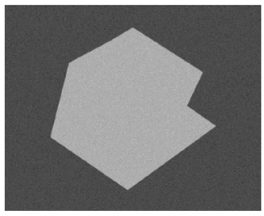
6.0.2 Image `another_coins`

The process is exactly the same as the maximum filter used for group of coins

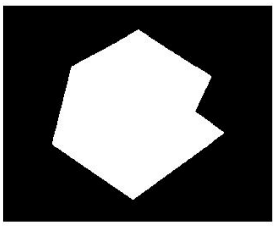
7 | Segmentation results

7.1 Inclass

7.1.1 Image `10_27a`

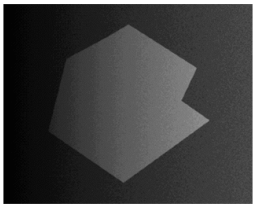


(a) Original input `10_27a` image

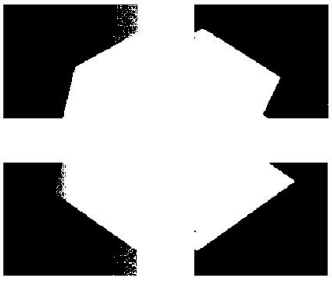


(b) Image after Otsu's Segmentation

7.1.2 Image `10_27d`

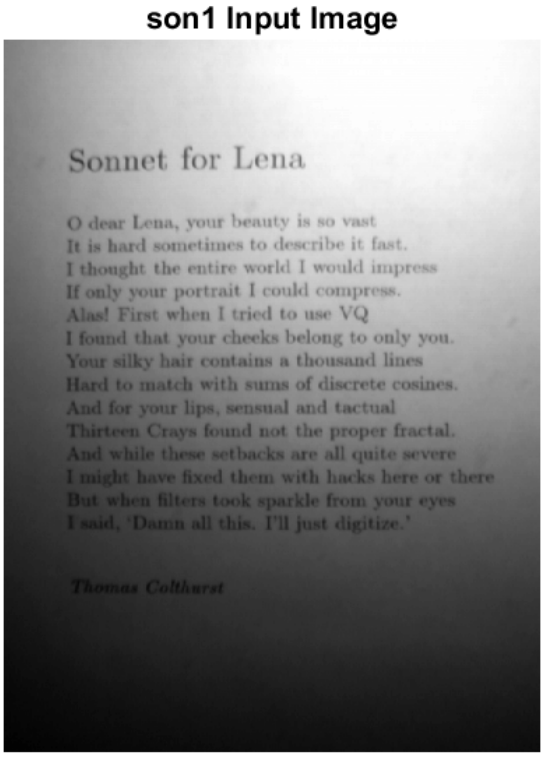


(a) Original input `10_27d` image



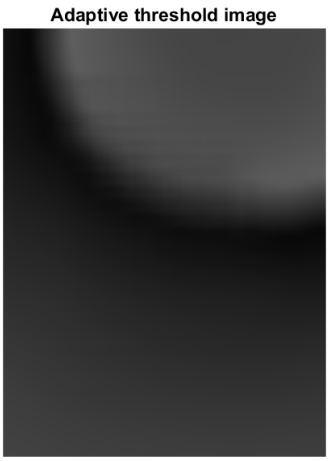
(b) Image after using divide and segment strategy

7.1.3 Image `son1`

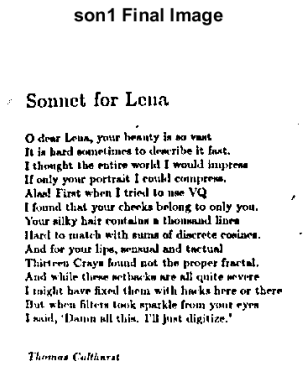


.6

Figure 7.3: Original input `son1` image

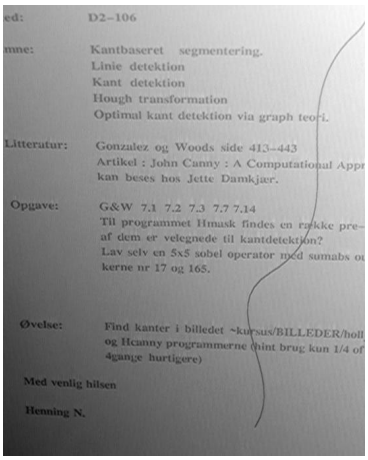


(a) `son1 - in process`

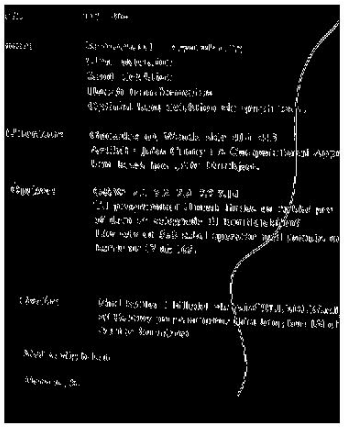


(b) `son1` - final outcome

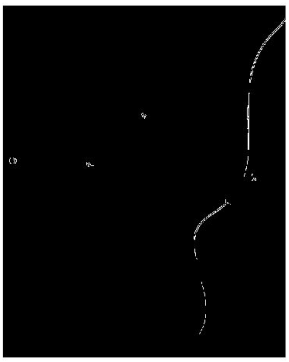
7.1.4 Image `papir`



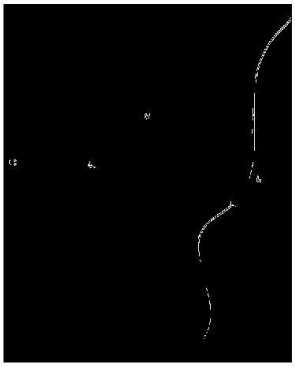
(a) `papir` - original input image



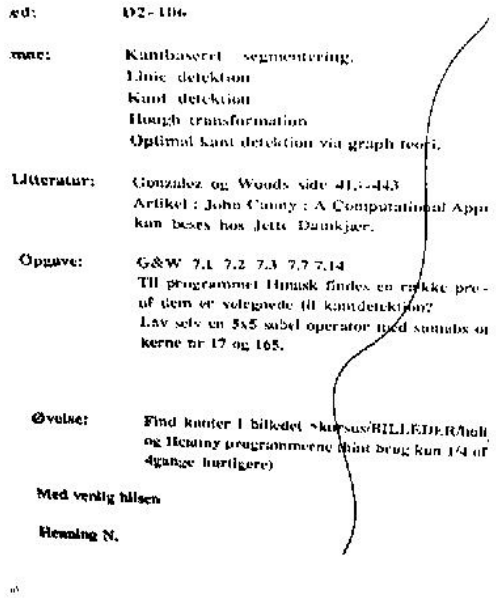
(b) `papir` - after roberts edge detection



(a) `papir` - remove small pixels

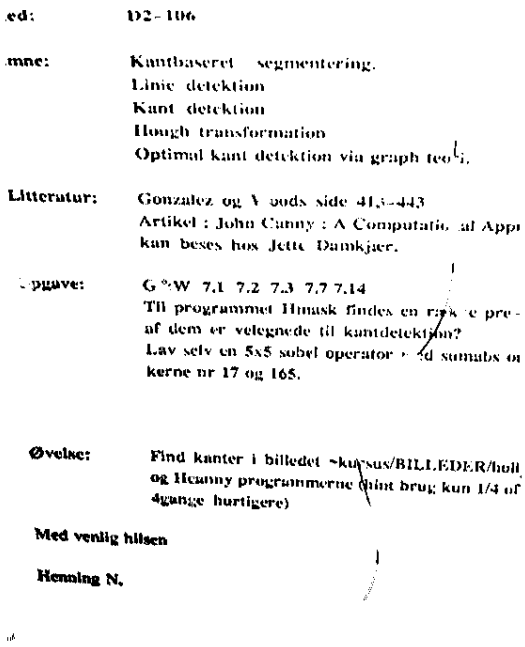


(b) `papir` - dilation to thicken hair



.6

Figure 7.7: adaptive threshold on the image to obtain binary image



.6

Figure 7.8: final outcome - addition of binary image dilated image

7.2 Extras

7.2.1 Image `grp of coins`

Figure 7.9: Original input `grp_of_coins` image

Figure 7.10: segmented image using otsu's method

7.2.2 Image `jet_fighter_swarm`

Figure 7.11: segmented image

7.2.3 Image - `grp of coins`(sobel)

Figure 7.12: Original input `grp_of_coins` image

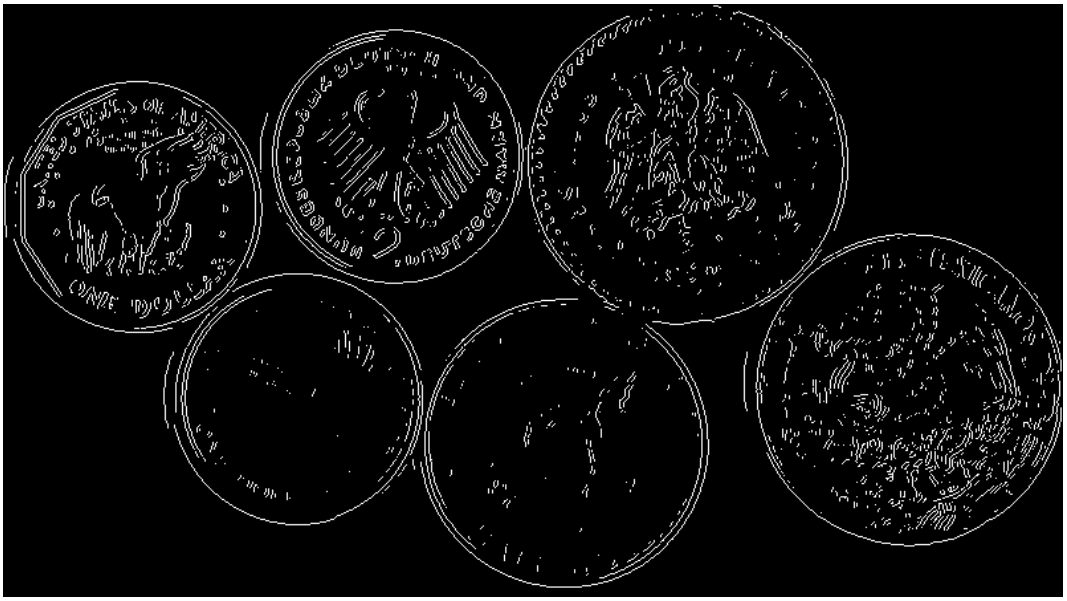


Figure 7.13: group of coins sobel operator

7.2.4 Image - another coins(sobel)

Figure 7.14: Original input another_coins image

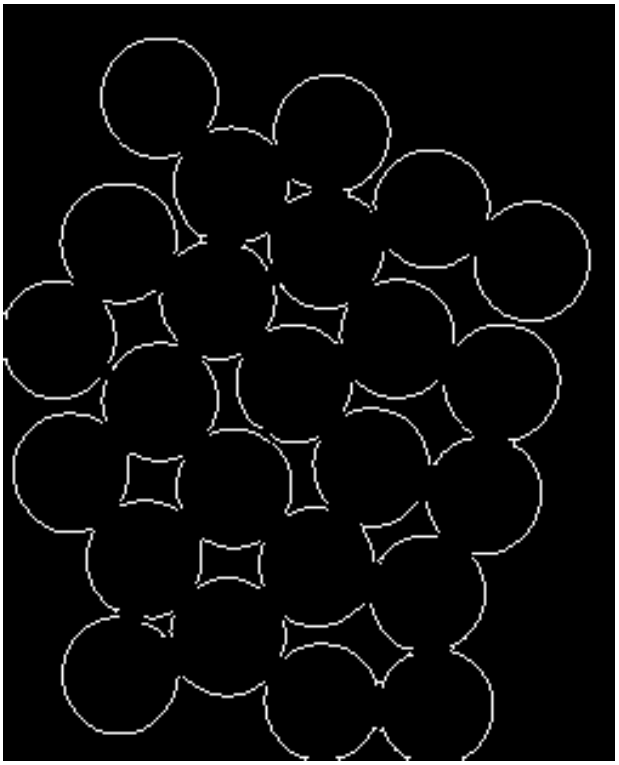


Figure 7.15: another coins sobel operator

7.2.5 Image - `pillars of hercules`(sobel)

Figure 7.16: Original input image

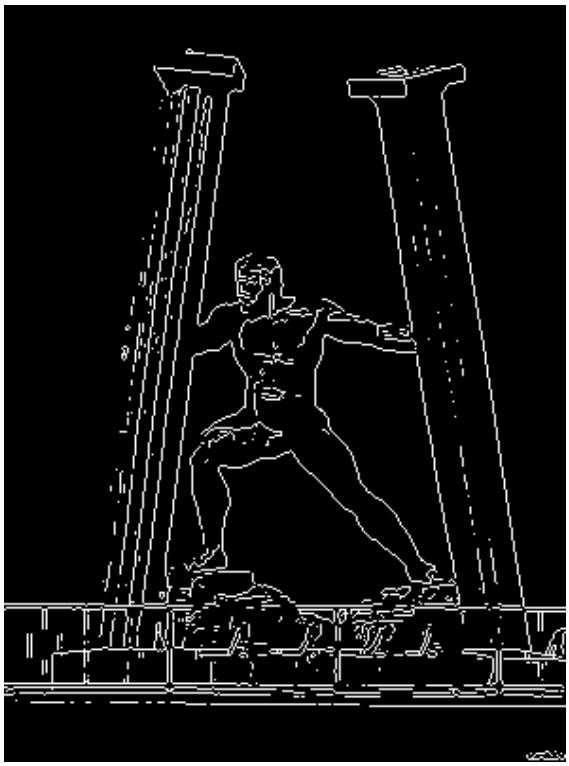


Figure 7.17: outcome using sobel operator

7.2.6 Image - `group of coins`(canny)

Figure 7.18: Original input image

Figure 7.19: outcome using canny edge detector

7.2.7 Image - `another coins`(canny)

Figure 7.20: Original input image

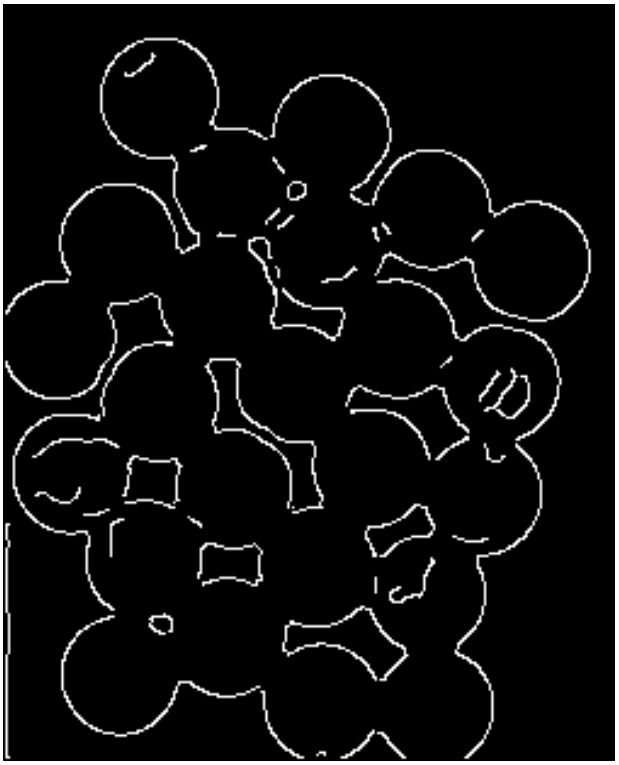


Figure 7.21: outcome using canny edge detector

7.2.8 Image - `pillars of hercules`(canny)

Figure 7.22: Original input image

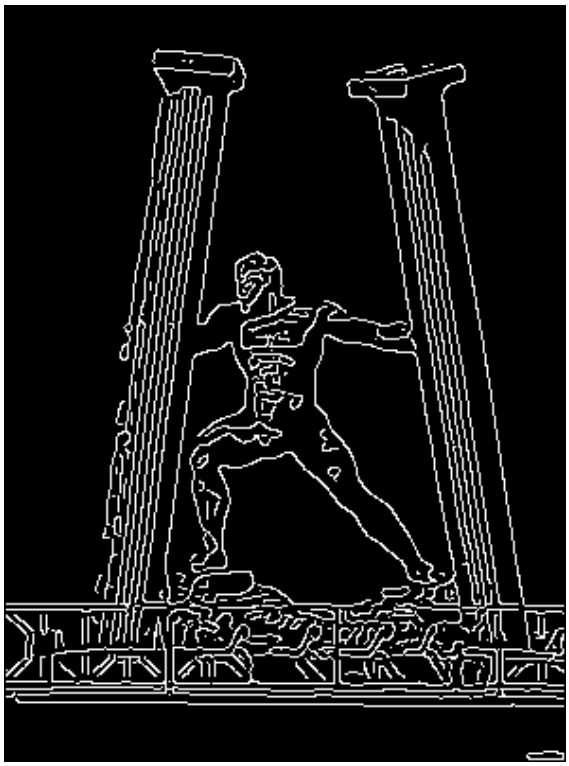


Figure 7.23: outcome using canny edge detector

7.2.9 Image - `baby elephant`(morphology)

Figure 7.24: Original input image

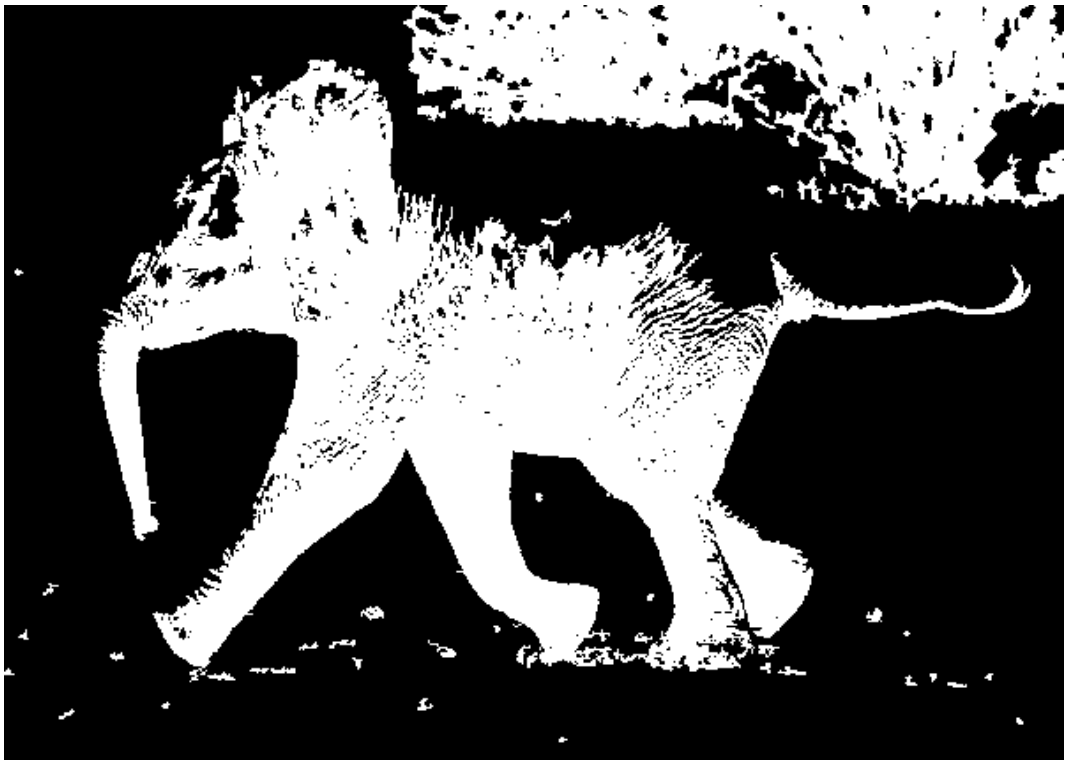


Figure 7.25: background elimination using berosion

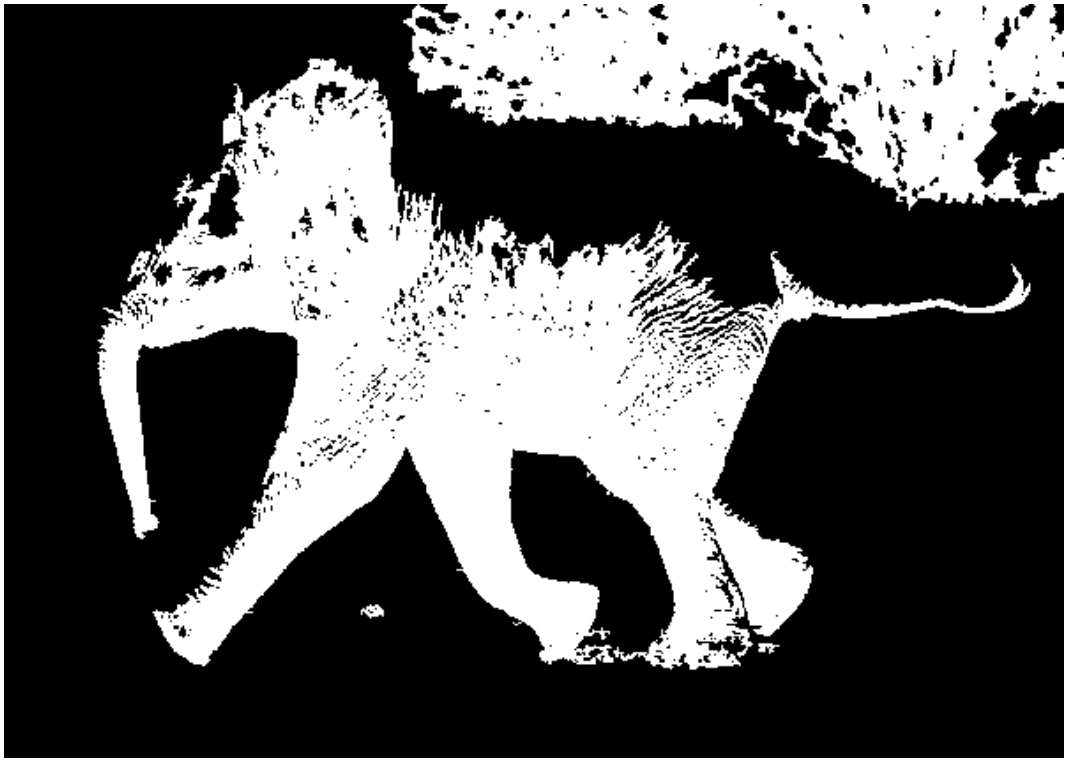


Figure 7.26: final result, best of our ability

7.2.10 Image - `baby elephant` (maximum filter)

Figure 7.27: outcome of group of coins image

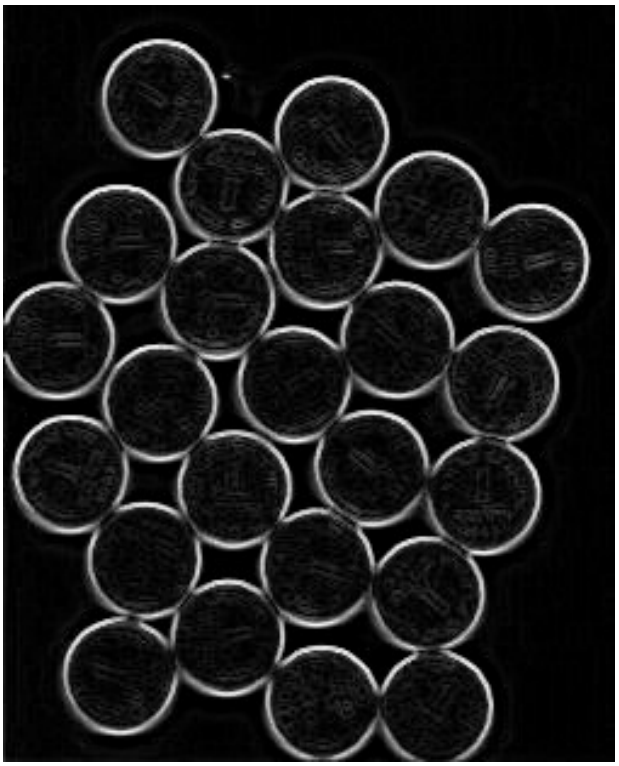


Figure 7.28: outcome of another coins image

8 | Summary

Figure `10_27a`: The method we used to solve this problem was good and use of any other algorithm would have been the same.

Figure `10_27d`: It was a good strategy to divide the image into blocks and use Otsu's segmentation on it. We did have a small amount of error which could have been avoided using moving window method.

Figure `group_of_coins`: For this image we used Otsus method and the we eroded the image to get the coins segmented properly. The resultant output image has the all the coins segmented from the background, but we observed that still some coins are attached at edges and this could be avoided or coins could be detached with **watershed** algorithm using **distance transform**.

Figure `jet_fighter_swarm`: This is pretty straight forward as we were able to segment the objects from background using Otsus thresholding method.

Figure `son1.gif`: For the image, we used adaptive thresholding and we were able to control the neighboring size directly. With this technique we were successful in segmenting the text from the background.

Figure `papir`: We were successful in removing 80% of the hair without loss of any text. In out method, 100% removal of hair would have resulted in loss text.

8.1 Beyond Sobel and Canny

8.1.1 The Hough Transform

The Hough Transform addresses the incompleteness of the edges. The incomplete edges in **Figure** `group_of_coins`, **Figure** `pillars_of_hercules`, and **Figure** `another_coins` can be solved by going one step ahead by applying **The Hough Transform**.

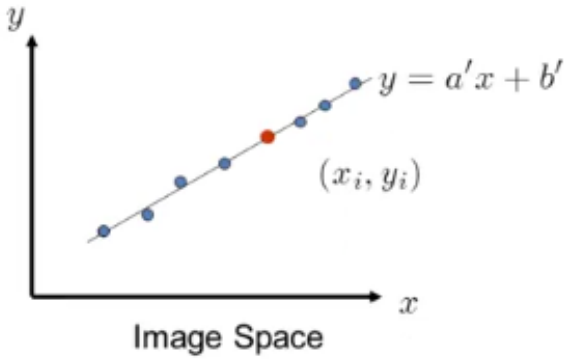


Figure 8.1: set of points belonging to a line

The basic procedure of Hough Transform is:

1. Obtain a binary edge using any edge detection techiniques discussed so far in the report.
2. Quantify $\rho\theta-$ plane.
3. Examine the number of intersects at each cell in the $\rho\theta-$ plane.
4. Bridge the gap based on continuity.

8.1.2 Active Contours

Figure 8.2: Active contour in action

Active Contours(a.k.a **snakes**) is an advanced edge-detection techniques. It is a controlled continuity contour like a elastic rubber band which encloses a target object by locking its edges. The initial contour is assigned velocities than move towards the target object boundary of interest.

In the future, we would like to work on applying active contour to all the edge-detection problems. Also, we would apply the hough transform to complete incomplete boundaries of images.