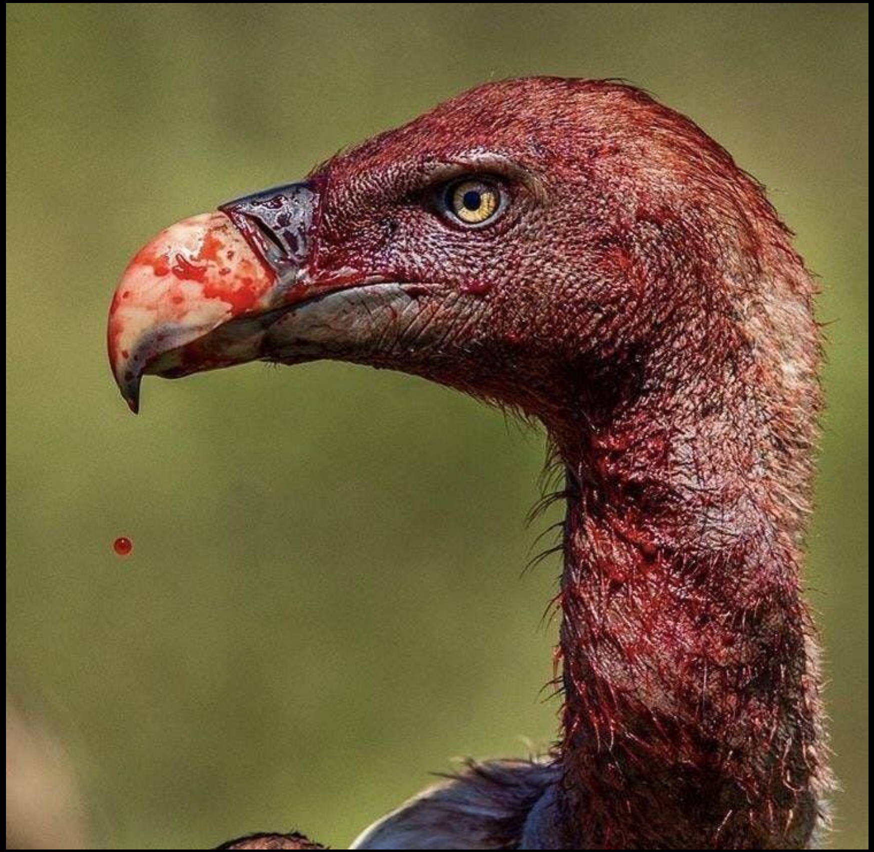
IMAGE SEGMENTATION

ACTIVITY 7

Krishna Lyn Delima

2014-64503

Original Image



A Vulture

Segmentation using various methods :

- Thresholding
- Parametric segmentation
- Non-Parametric segmentation

Done using:

- Jupyter Notebook (Python)
  - Packages:

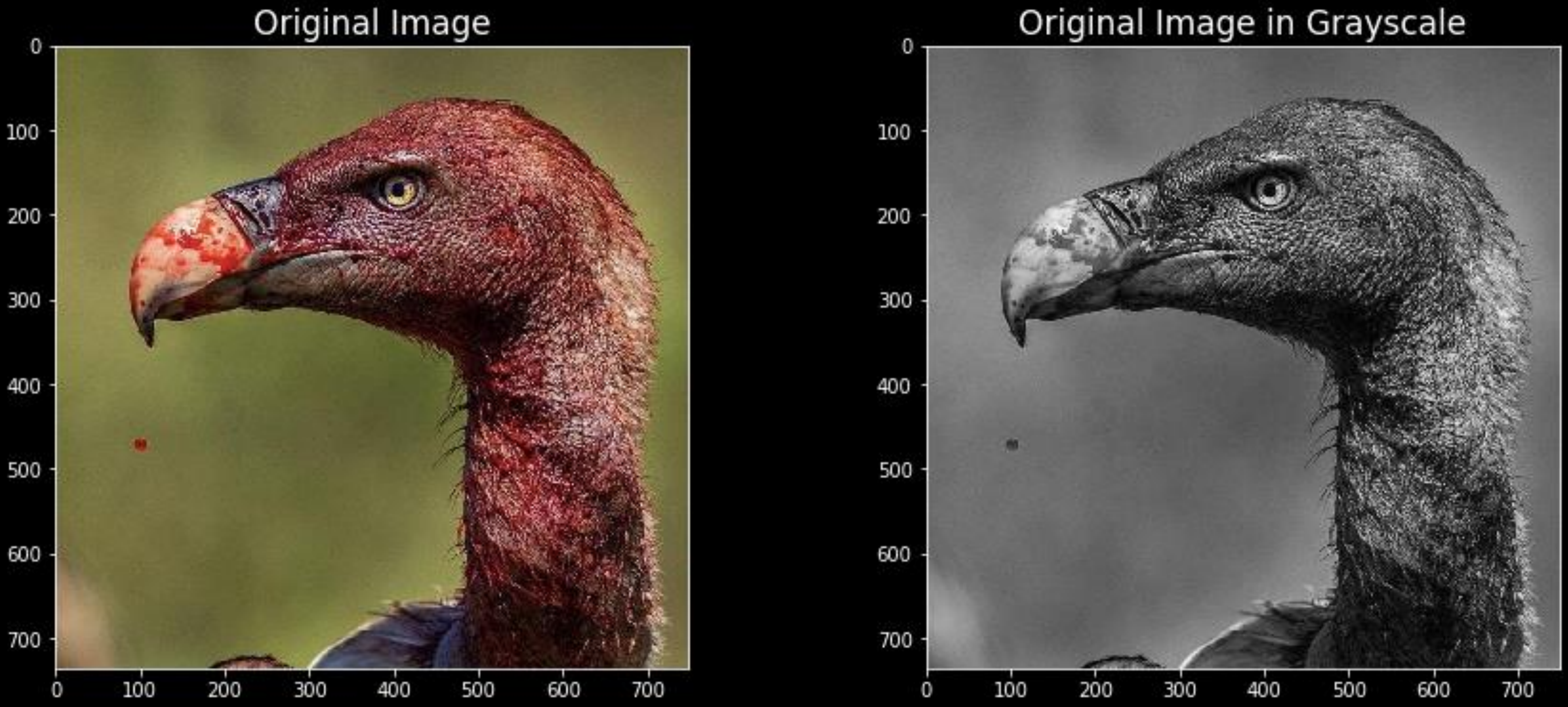
```
import matplotlib.pyplot as plt
import numpy as np
import cv2
from scipy.signal import find_peaks
from skimage import img_as_float
```

* Photos taken from Google Images and Pinterest

Thresholding

1

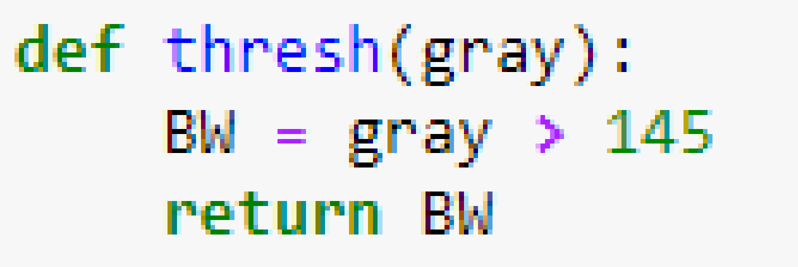
Step 1 To Grayscale



Original Image

Original Image in Grayscale

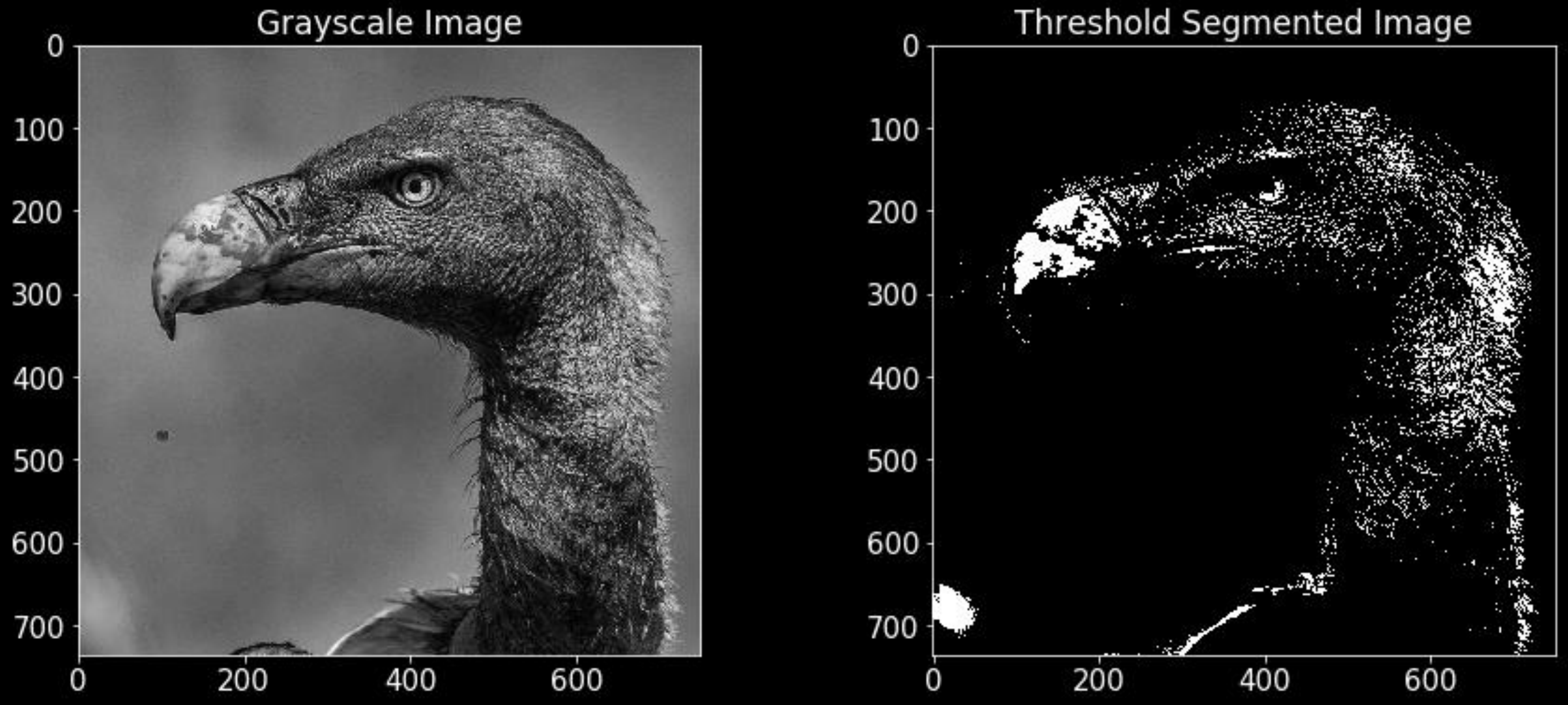
Step 2 Thresholding



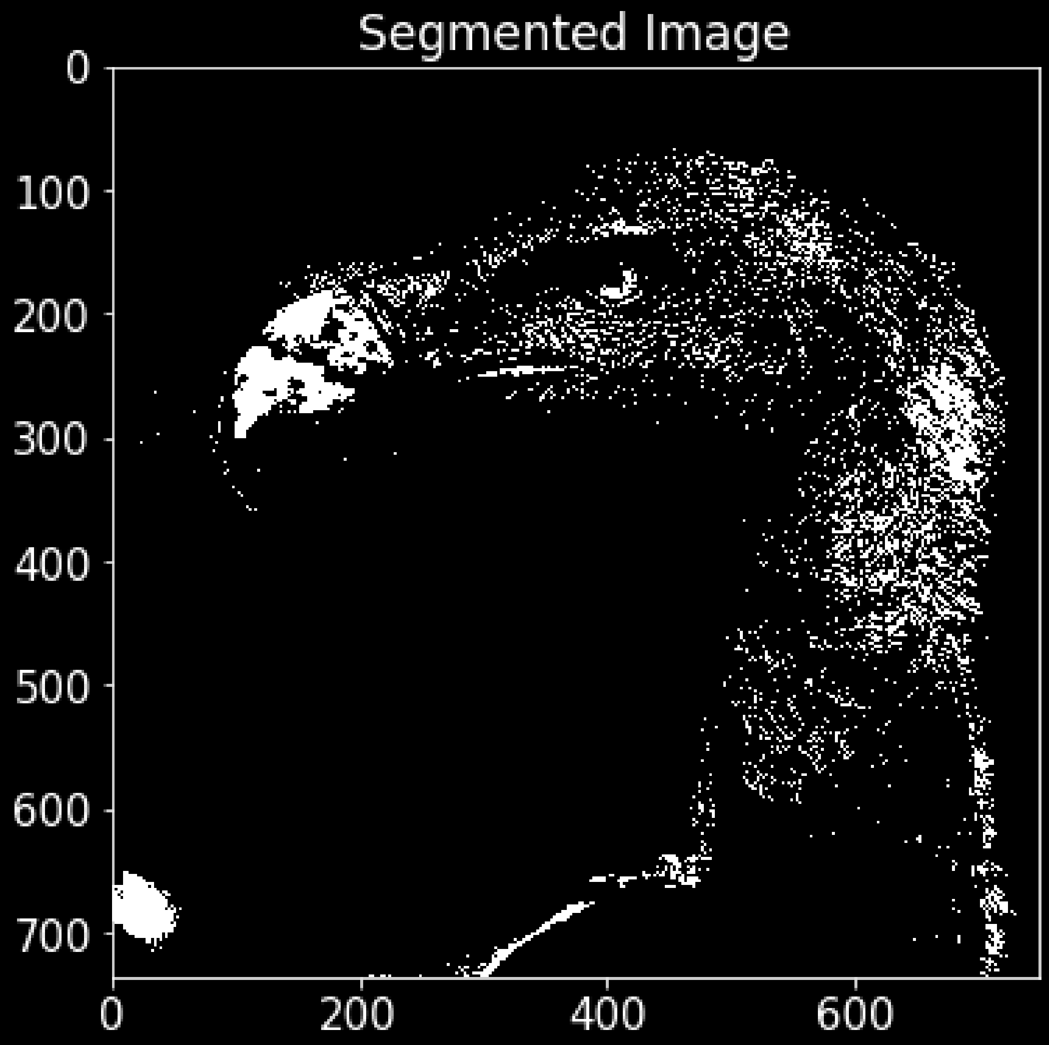
```
def thresh(gray):
    BW = gray > 145
    return BW
```

- ‘gray’ is the grayscale of the original image
- A threshold is set at 145
- The threshold value is adjustable in order to attain the best segmentation

Step 2 Thresholding (Result)



Step 2 Thresholding (Result)



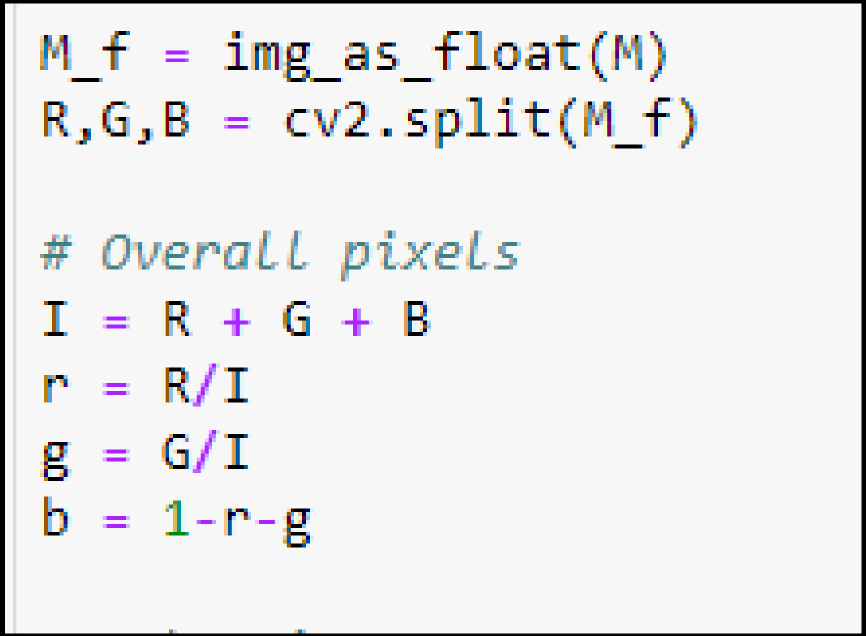
- ‘This segmentation method was able to produce the outline of the Vulture.
- However, these lines aren’t definite. After all, this method has its limit since the threshold value may account for other parts of the picture like the white pigments outside the Vulture.

Parametric Segmentation

2

Step 1 NCC coordinates

- With M as the original image, the NCC coordinates were taken by using these equations on the left.
- ‘blue’ is now dependent on r and g.
- The image was converted to float for the calculation to avoid errors such as ‘True divide by zero’.

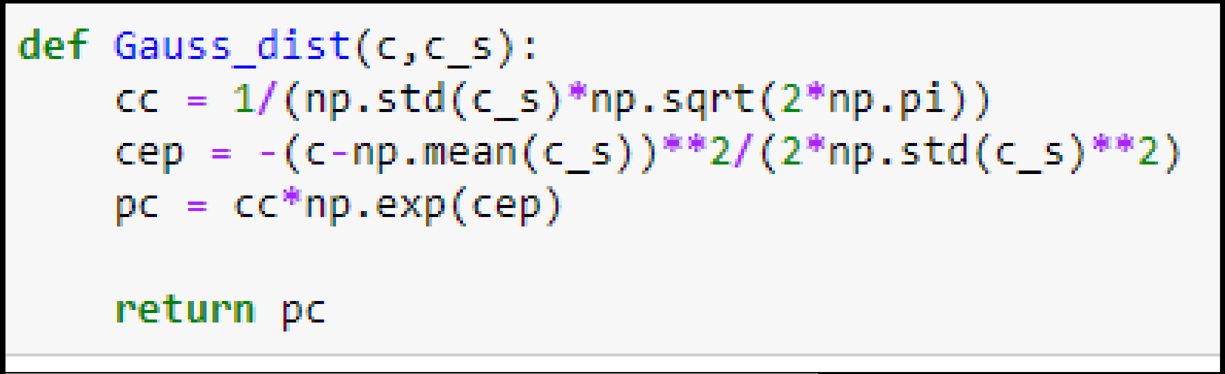


```
M_f = img_as_float(M)
R,G,B = cv2.split(M_f)

# Overall pixels
I = R + G + B
r = R/I
g = G/I
b = 1-r-g
```

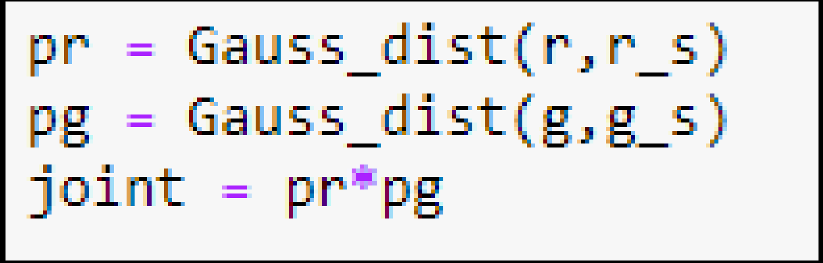
Step 2 Parametric Segmentation

- Gaussian distribution was taken for both r and g.
- ‘joint’ is the product of the r and g’s respective Gaussian distribution.
- The resulting image now has the ‘joint’ histogram.



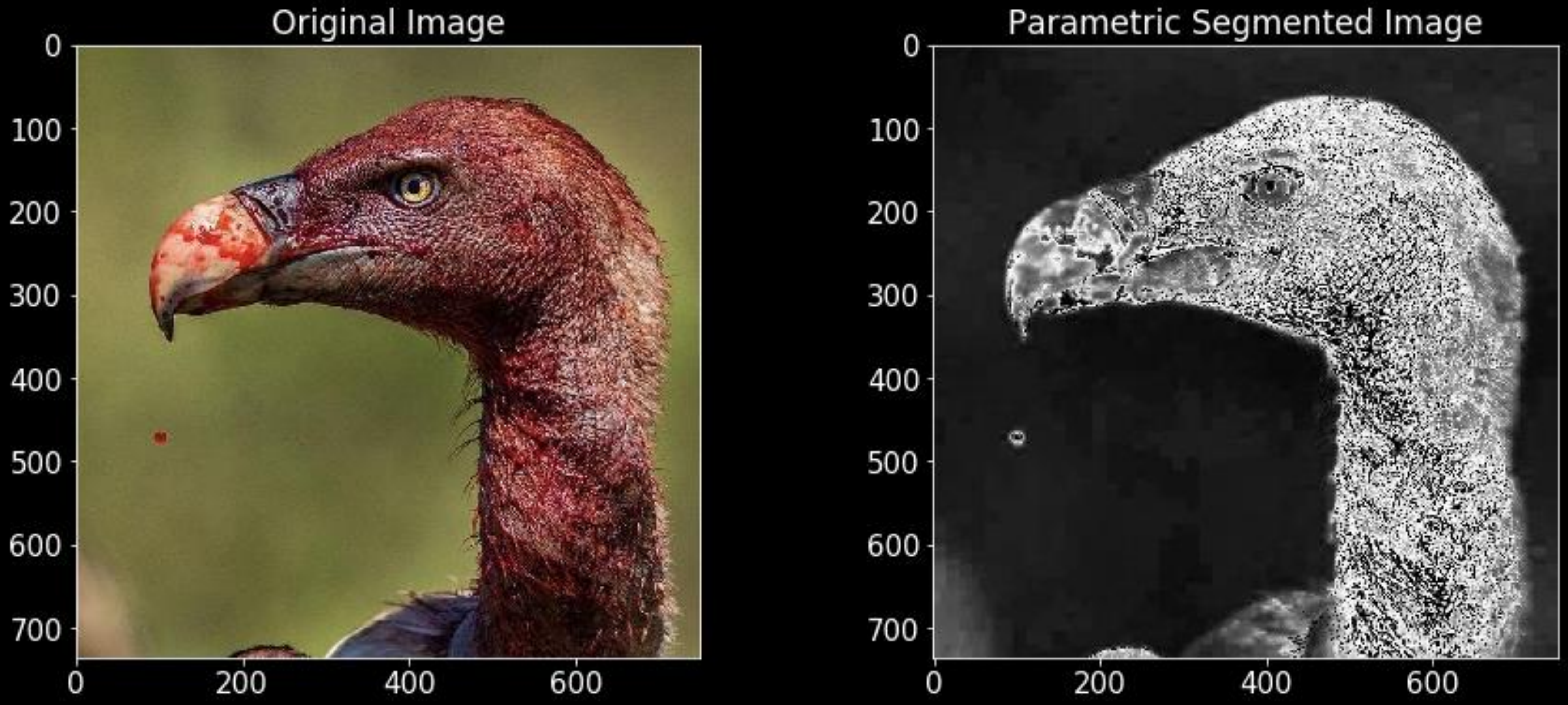
```
def Gauss_dist(c,c_s):
    cc = 1/(np.std(c_s)*np.sqrt(2*np.pi))
    cep = -(c-np.mean(c_s))**2/(2*np.std(c_s)**2)
    pc = cc*np.exp(cep)

    return pc
```



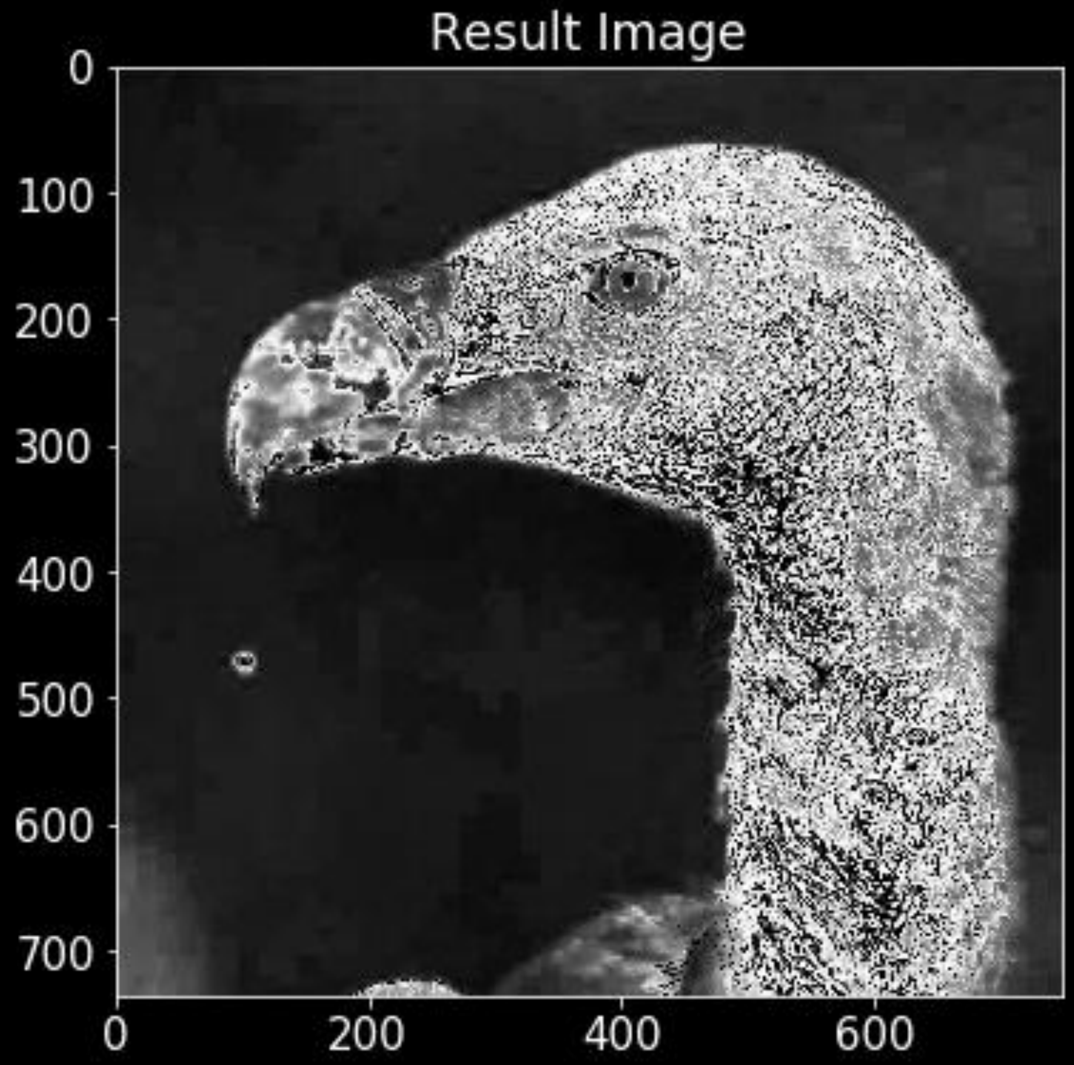
```
pr = Gauss_dist(r,r_s)
pg = Gauss_dist(g,g_s)
joint = pr*pg
```

Step 2 Parametric Segmentation (Results)



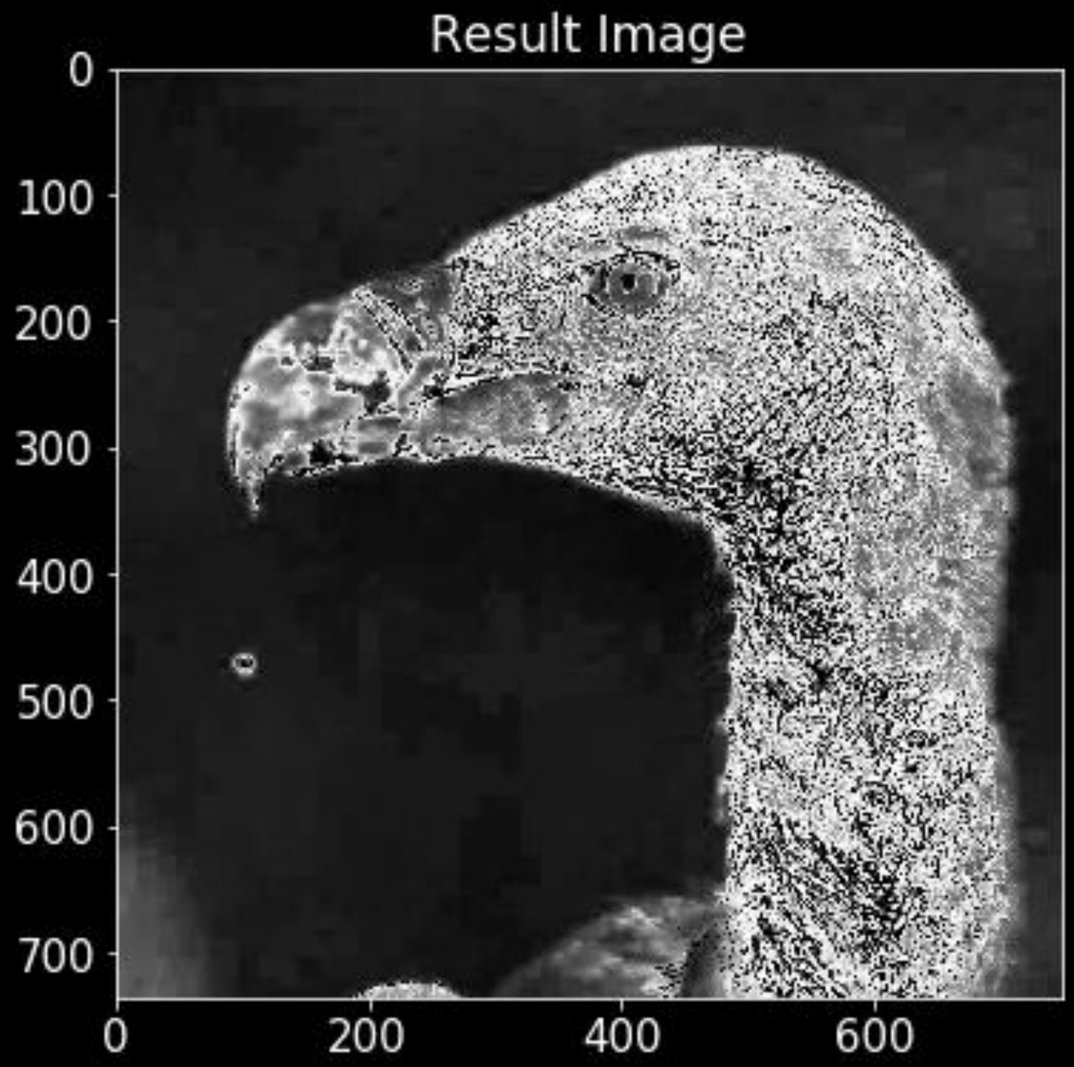
Step 2 Parametric Segmentation (Results)

- ‘This segmentation method was able to produce the outline of the Vulture accurately.
- This method also captured the blood droplet on the left side of the image.

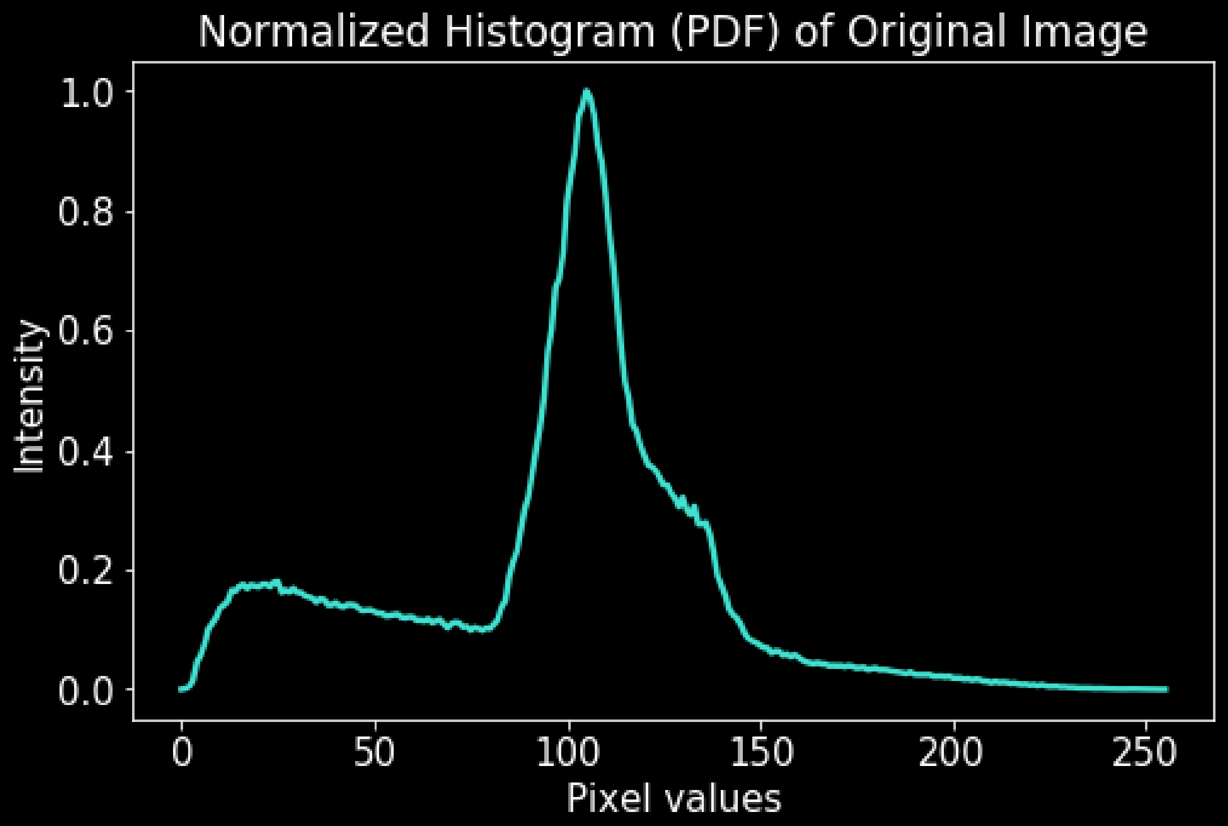


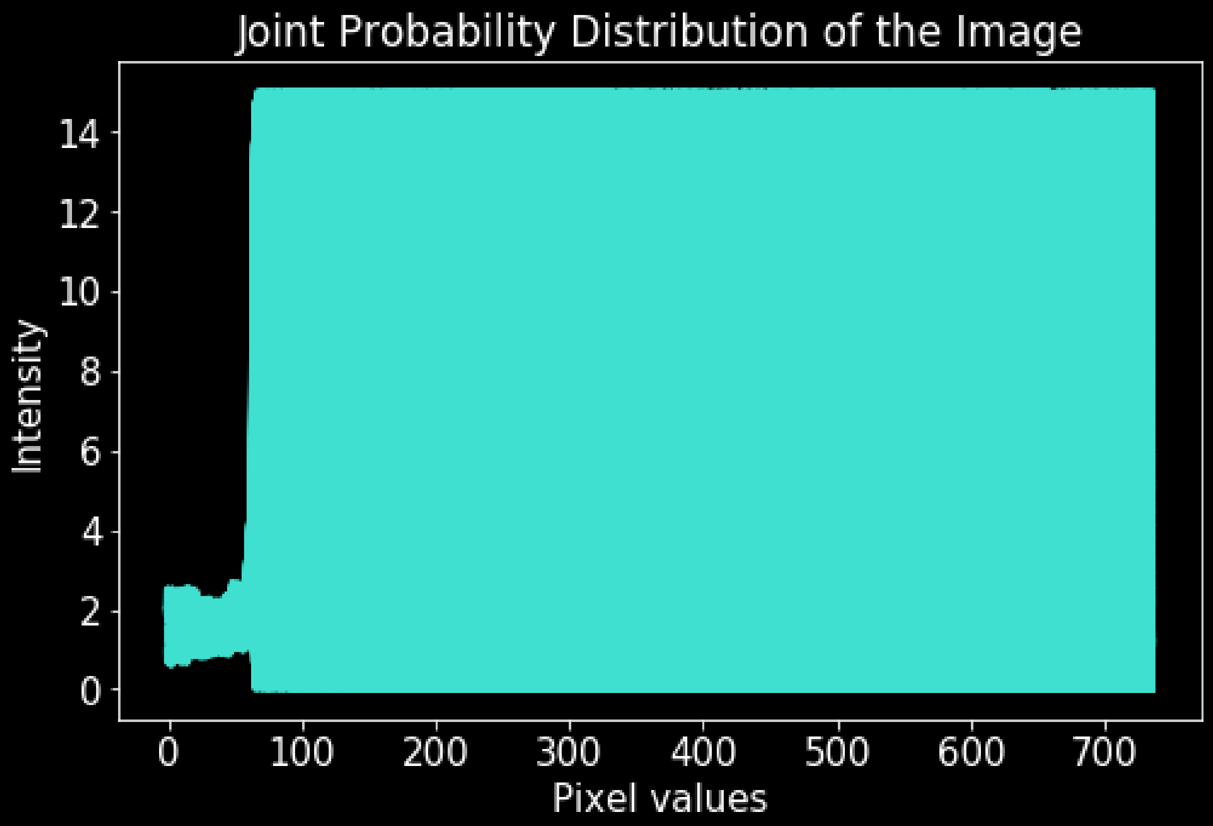
Step 2 Parametric Segmentation (Results)

- ‘This segmentation method was able to produce the outline of the Vulture accurately.
- This method also captured the blood droplet on the left side of the image.



Step 2 Parametric Segmentation (Results)



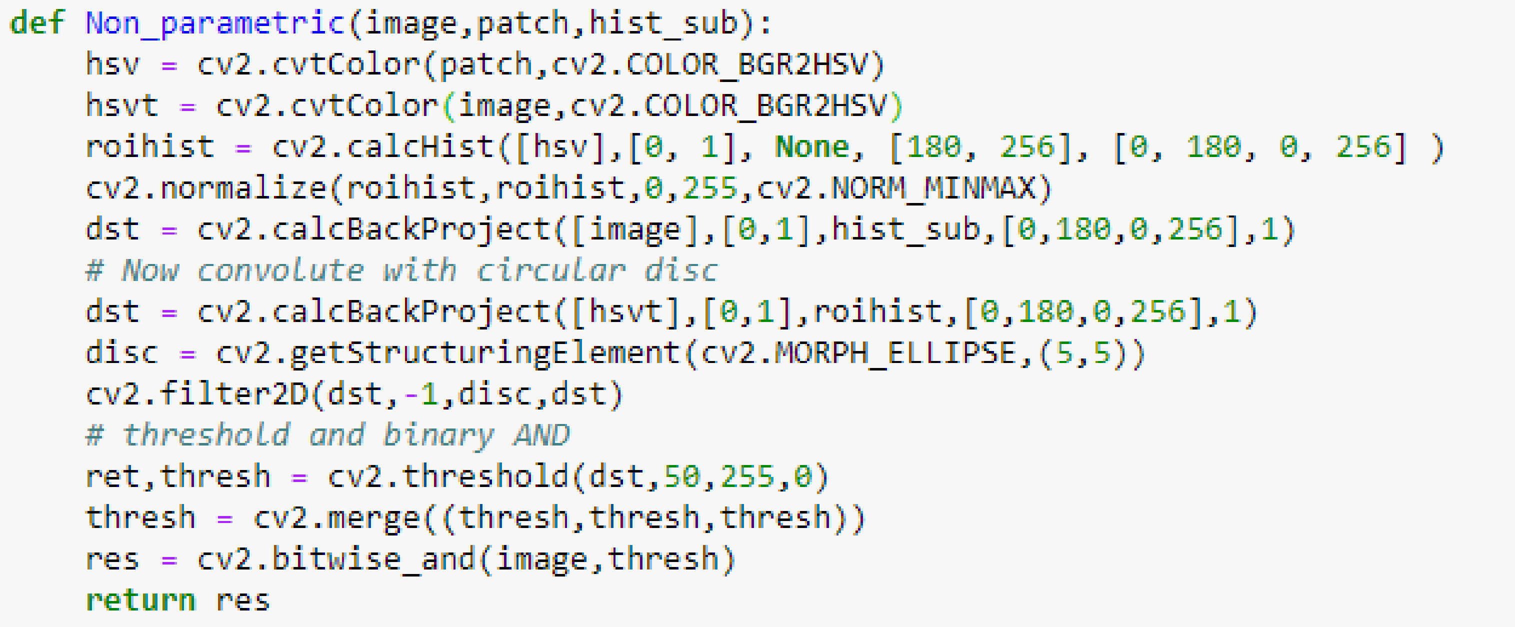


- The difference in apparent in these histograms
- Most of the pixel values had the same max peak values
- Unknowingly, the x-axis reached 700 px. This prompted a new shape for the image

Non-Parametric Segmentation

3

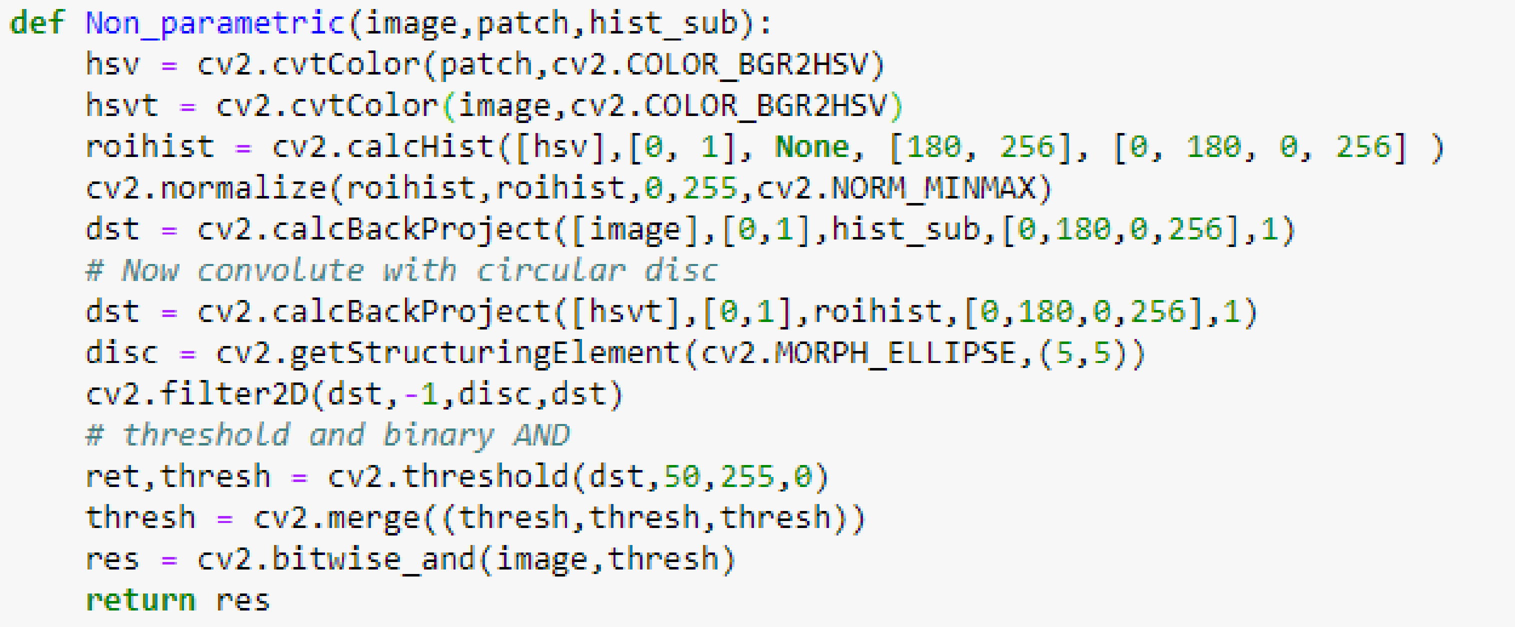
Step 1 Non-Parametric Segmentation



```
def Non_parametric(image,patch,hist_sub):
    hsv = cv2.cvtColor(patch,cv2.COLOR_BGR2HSV)
    hsvt = cv2.cvtColor(image,cv2.COLOR_BGR2HSV)
    roihist = cv2.calcHist([hsv],[0, 1], None, [180, 256], [0, 180, 0, 256] )
    cv2.normalize(roihist,roihist,0,255,cv2.NORM_MINMAX)
    dst = cv2.calcBackProject([image],[0,1],hist_sub,[0,180,0,256],1)
    # Now convolute with circular disc
    dst = cv2.calcBackProject([hsvt],[0,1],roihist,[0,180,0,256],1)
    disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
    cv2.filter2D(dst,-1,disc,dst)
    # threshold and binary AND
    ret,thresh = cv2.threshold(dst,50,255,0)
    thresh = cv2.merge((thresh,thresh,thresh))
    res = cv2.bitwise_and(image,thresh)
    return res
```

- This was a code I found online : https://docs.opencv.org/master/dc/df6/tutorial_py_histogram_backprojection.html?fbclid=IwAR1p80GPxCOmRmZoeJIdTYYyu1xuq47Ia4o98vnb6qofh36EfWbo2pfiaTg
- I wasn’t able to do the indexing algorithm for this.

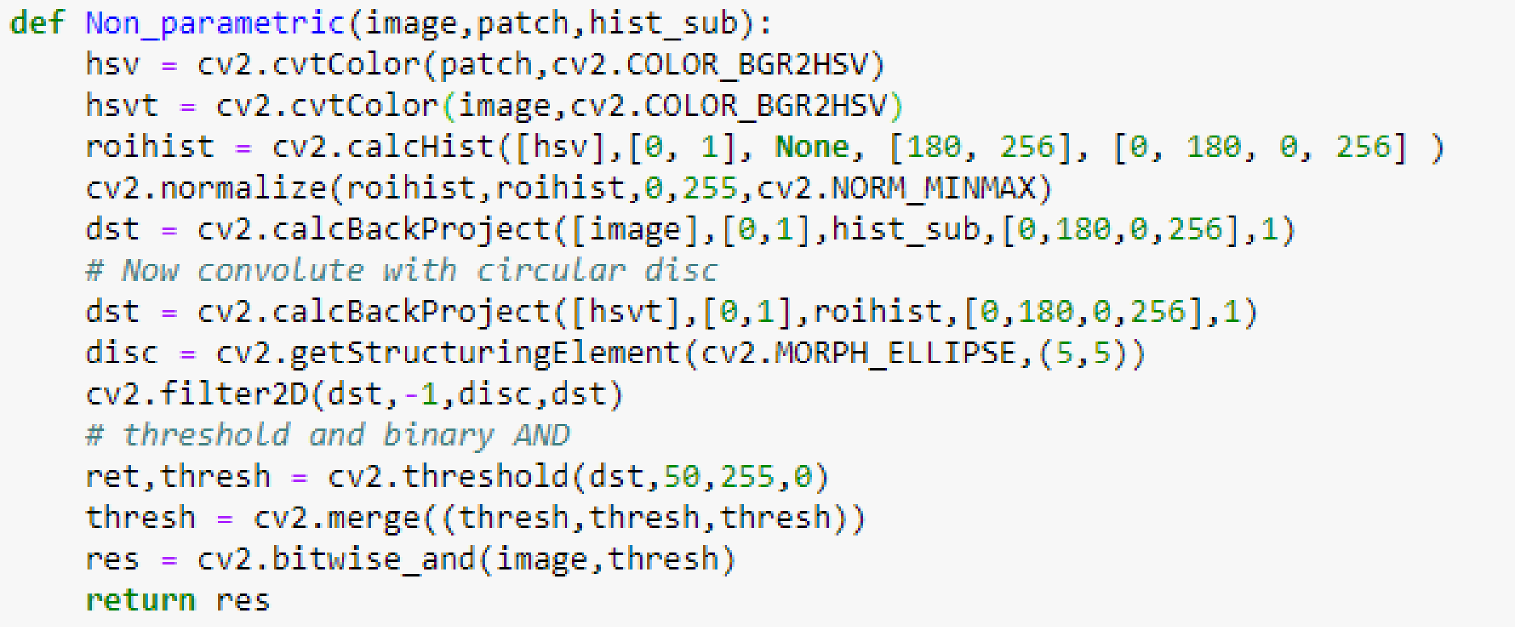
Step 1 Non-Parametric Segmentation



```
def Non_parametric(image,patch,hist_sub):
    hsv = cv2.cvtColor(patch,cv2.COLOR_BGR2HSV)
    hsvt = cv2.cvtColor(image,cv2.COLOR_BGR2HSV)
    roihist = cv2.calcHist([hsv],[0, 1], None, [180, 256], [0, 180, 0, 256] )
    cv2.normalize(roihist,roihist,0,255,cv2.NORM_MINMAX)
    dst = cv2.calcBackProject([image],[0,1],hist_sub,[0,180,0,256],1)
    # Now convolute with circular disc
    dst = cv2.calcBackProject([hsvt],[0,1],roihist,[0,180,0,256],1)
    disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
    cv2.filter2D(dst,-1,disc,dst)
    # threshold and binary AND
    ret,thresh = cv2.threshold(dst,50,255,0)
    thresh = cv2.merge((thresh,thresh,thresh))
    res = cv2.bitwise_and(image,thresh)
    return res
```

- This code uses a lot of packages from cv2.
- The histograms for the image and patch was produced using calcHist in cv2
- A disc was used as a filter mask
- A threshold was set in order to produce a fully contrasting image

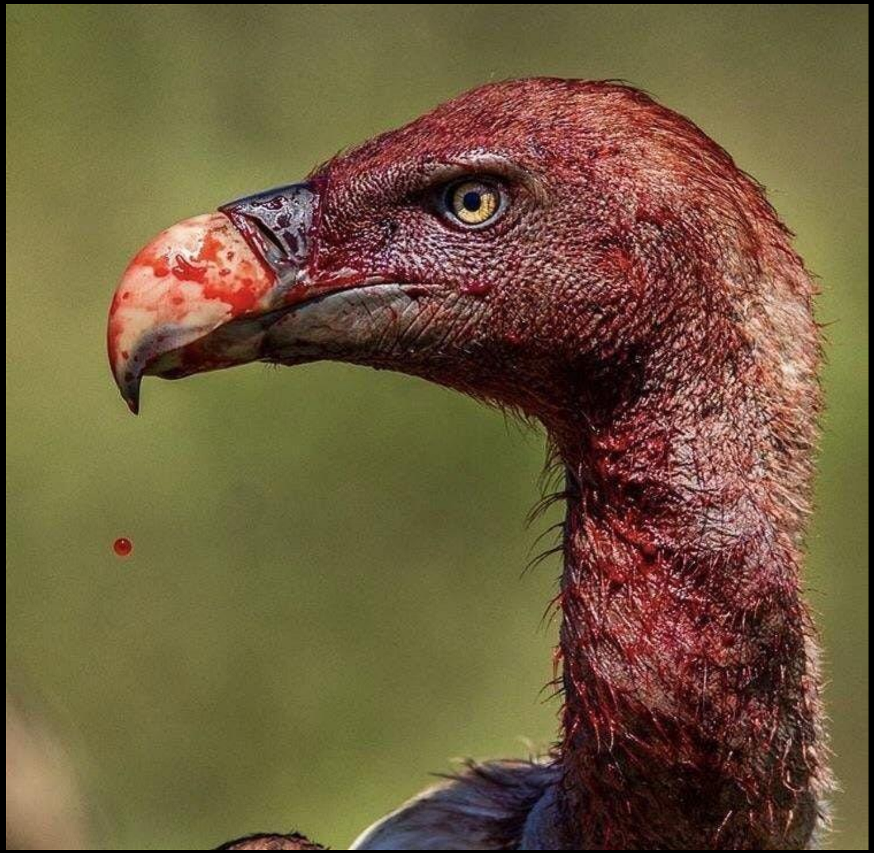
Step 1 Non-Parametric Segmentation

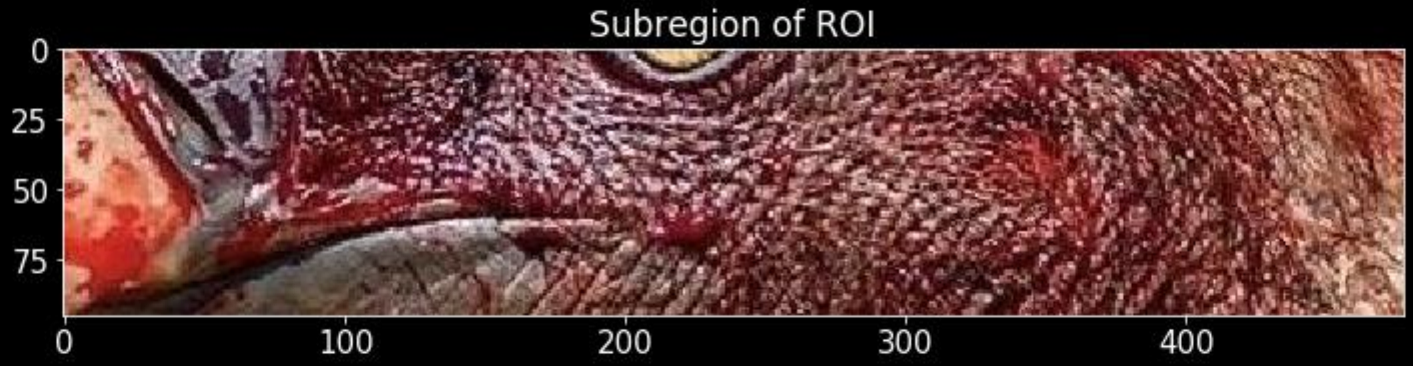


```
def Non_parametric(image,patch,hist_sub):
    hsv = cv2.cvtColor(patch,cv2.COLOR_BGR2HSV)
    hsvt = cv2.cvtColor(image,cv2.COLOR_BGR2HSV)
    roihist = cv2.calcHist([hsv],[0, 1], None, [180, 256], [0, 180, 0, 256] )
    cv2.normalize(roihist,roihist,0,255,cv2.NORM_MINMAX)
    dst = cv2.calcBackProject([image],[0,1],hist_sub,[0,180,0,256],1)
    # Now convolute with circular disc
    dst = cv2.calcBackProject([hsvt],[0,1],roihist,[0,180,0,256],1)
    disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
    cv2.filter2D(dst,-1,disc,dst)
    # threshold and binary AND
    ret,thresh = cv2.threshold(dst,50,255,0)
    thresh = cv2.merge((thresh,thresh,thresh))
    res = cv2.bitwise_and(image,thresh)
    return res
```

- The images were converted to HSV in order to produce a colored segmented image

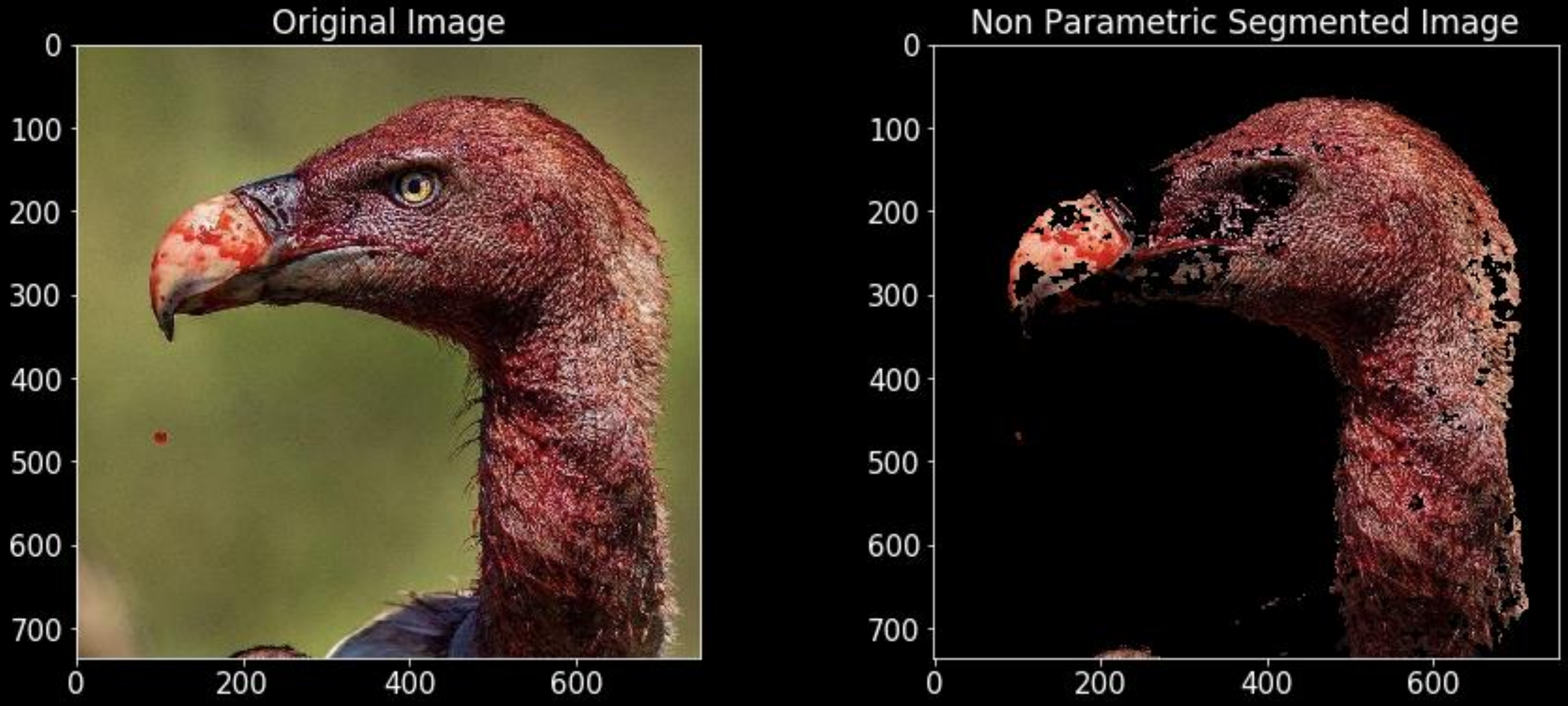
Step 2 Non-Parametric Segmentation



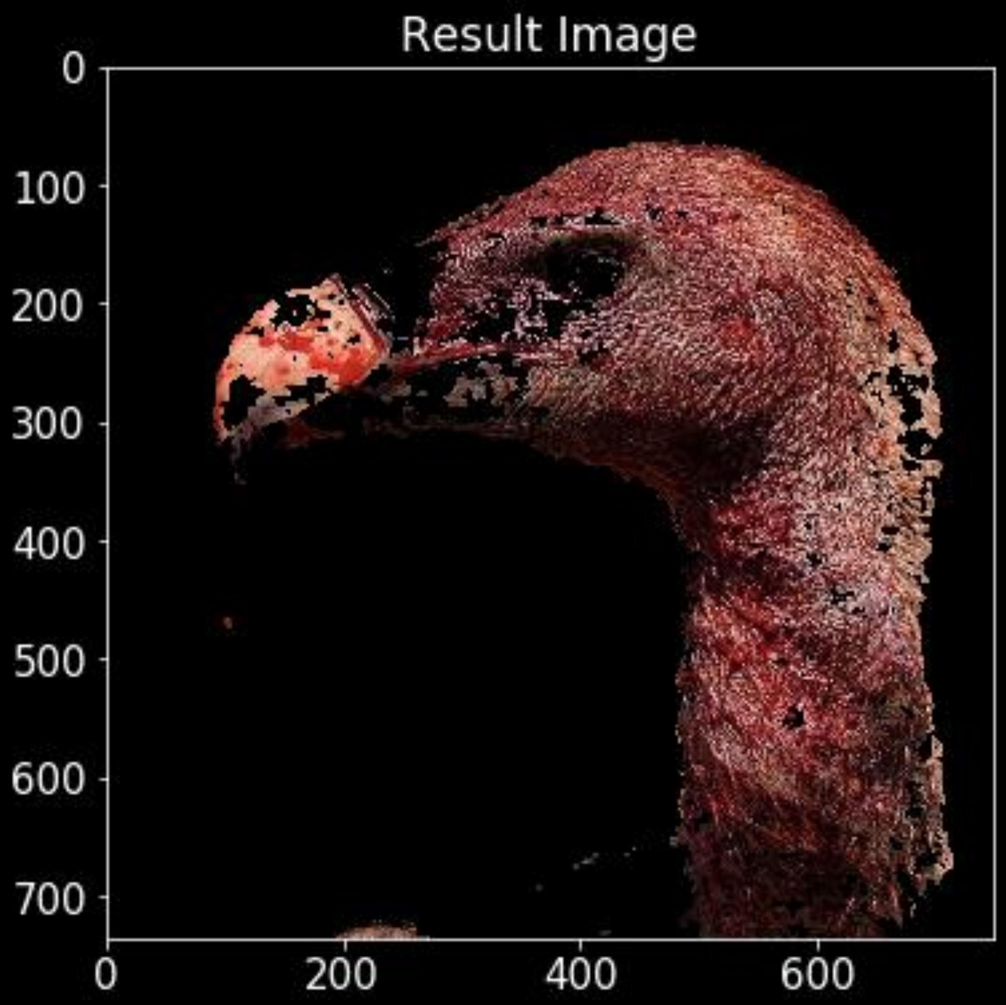


- This was the patch or subregion used in order to segment the Vulture

Step 2 Non-Parametric Results

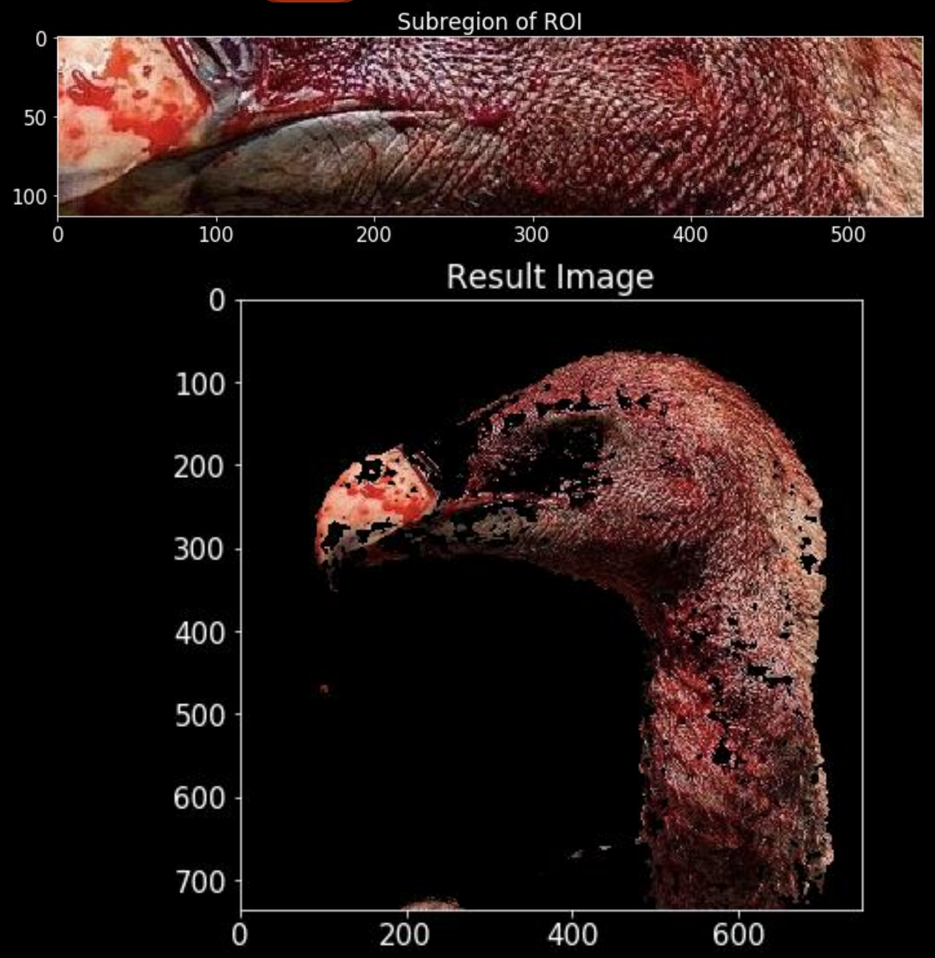


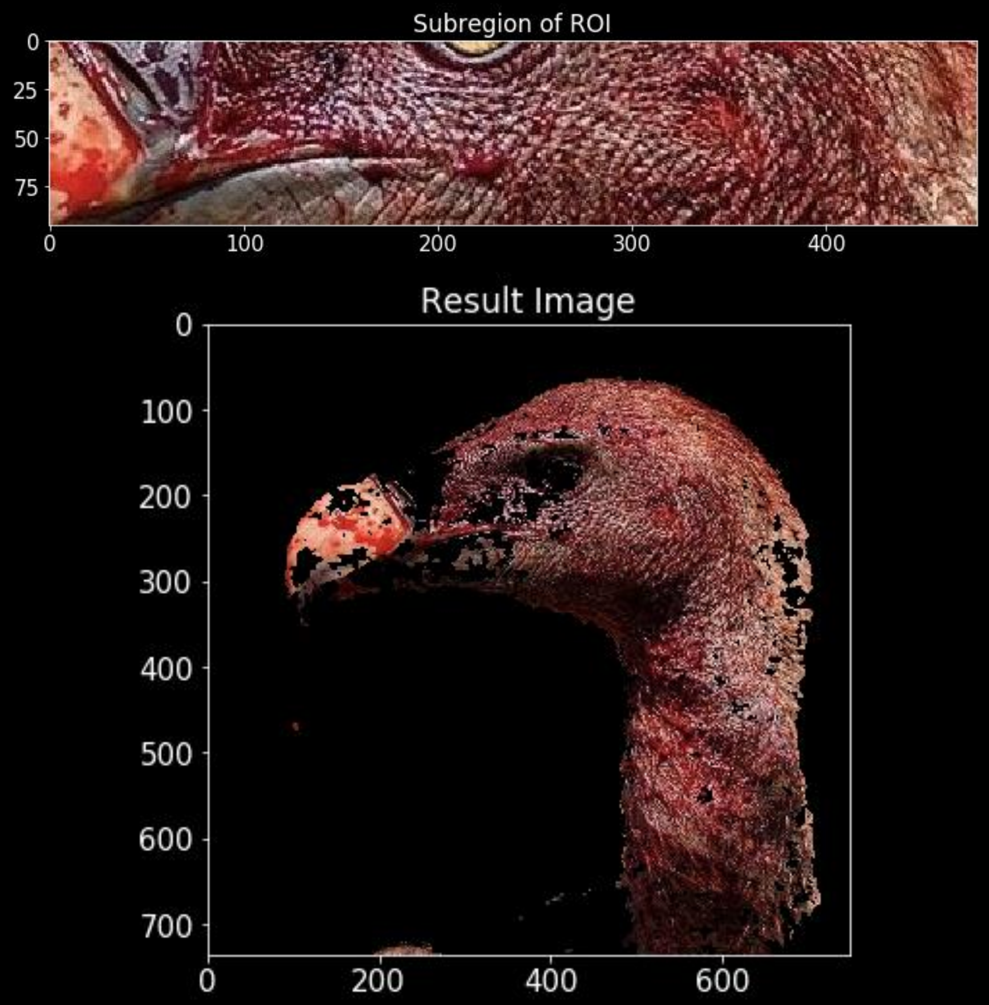
Step 2 Non-Parametric Results



- ‘This segmentation method was able to produce the outline of the Vulture
- However, there are missing parts on the Vulture’s features that weren’t captured. This may be due to the patch used

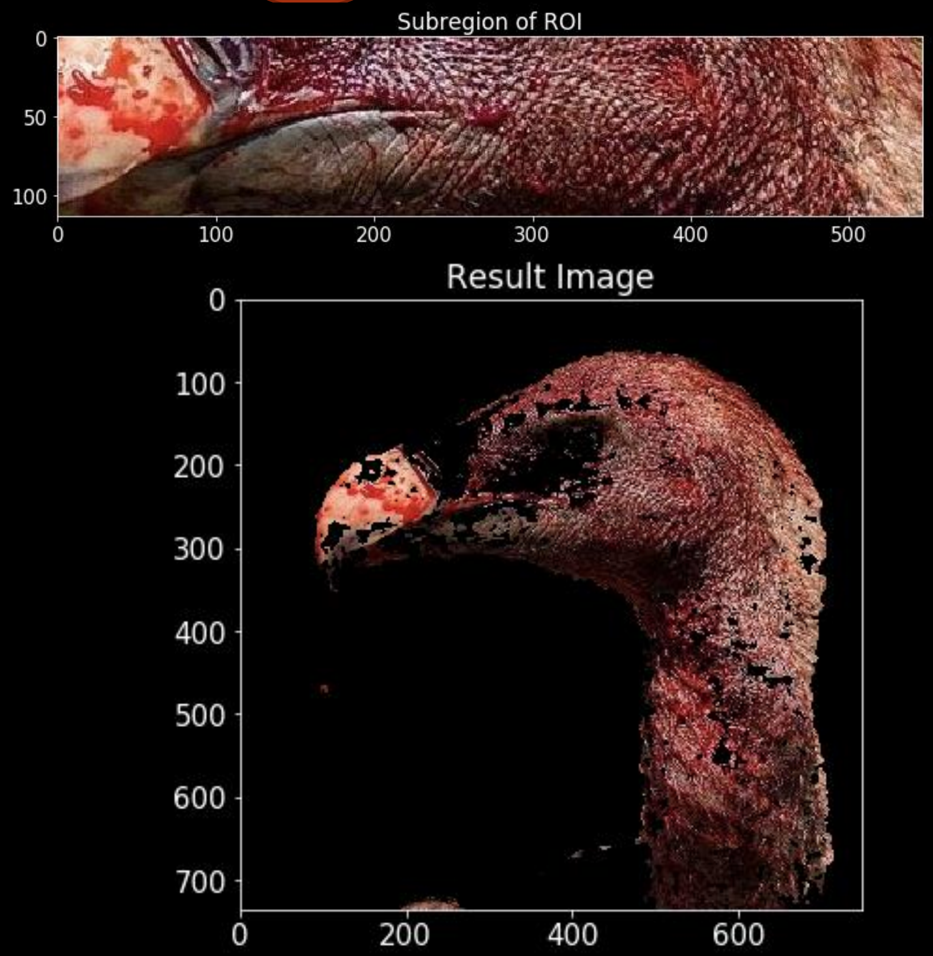
Step 2 Non-Parametric Results

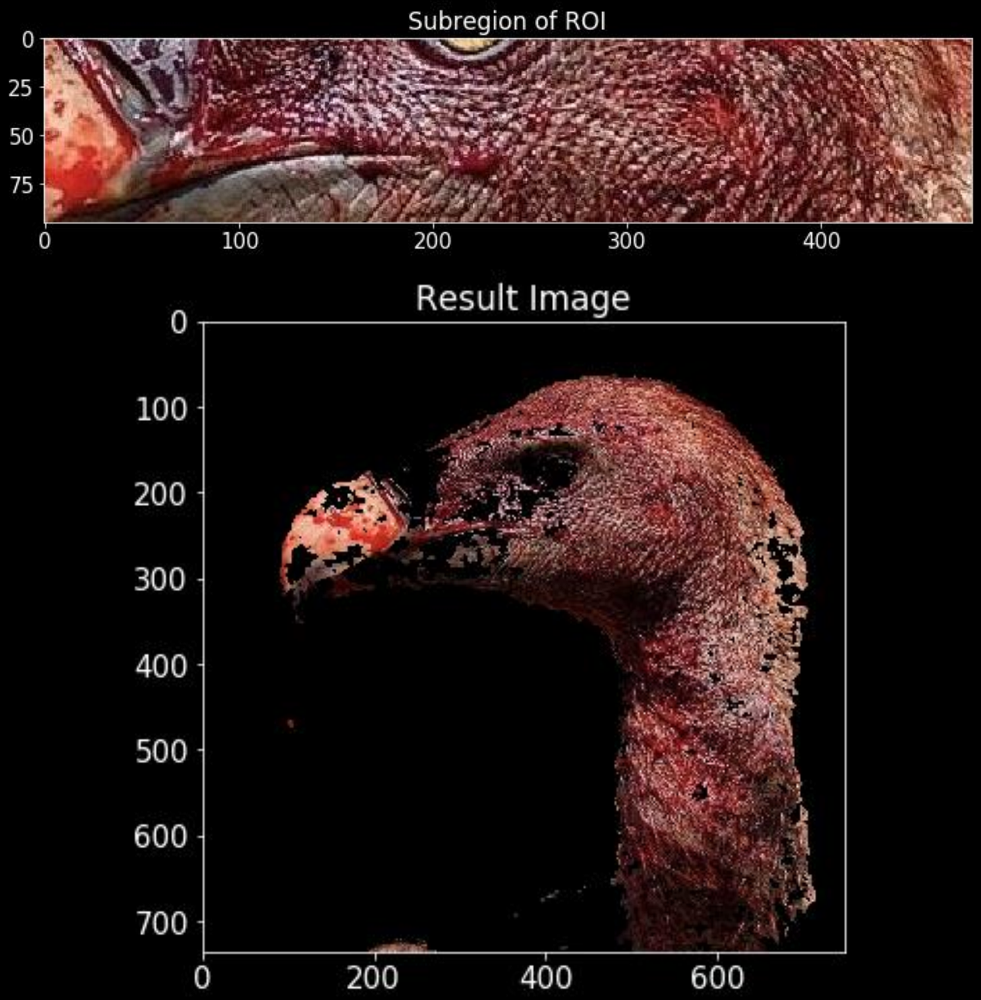




- The difference between these two patches was the vertical shift

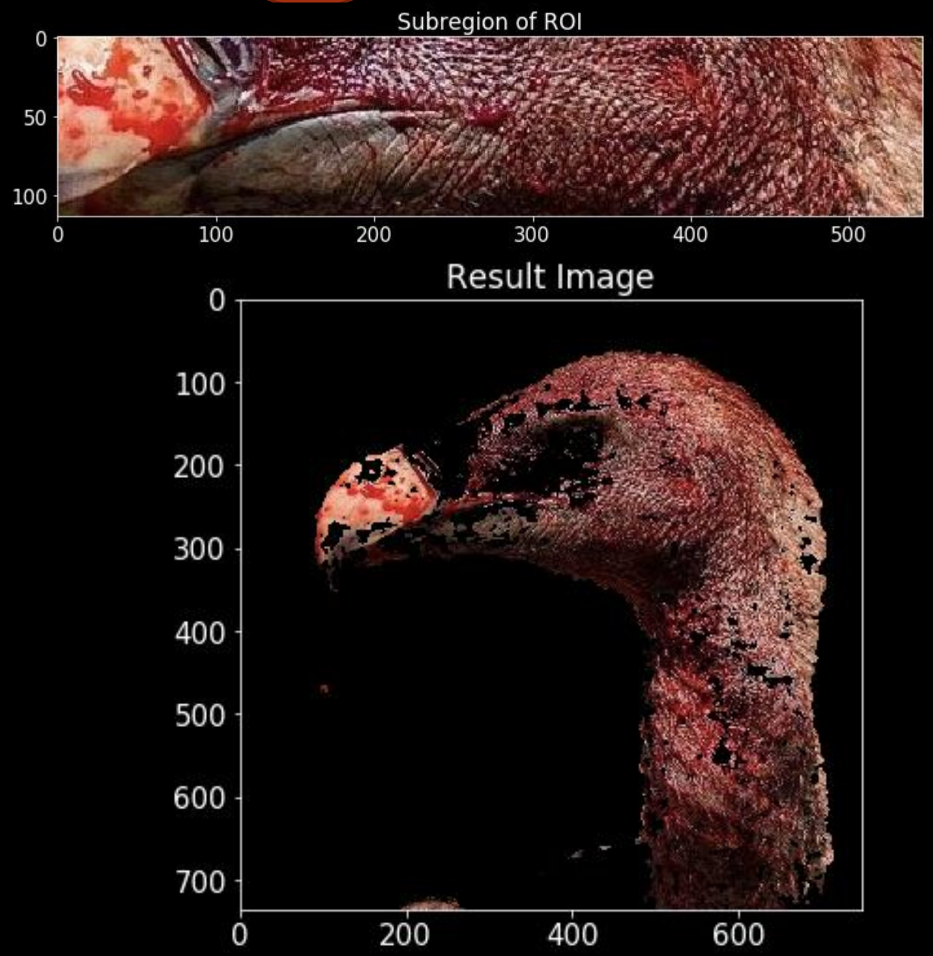
Step 2 Non-Parametric Results

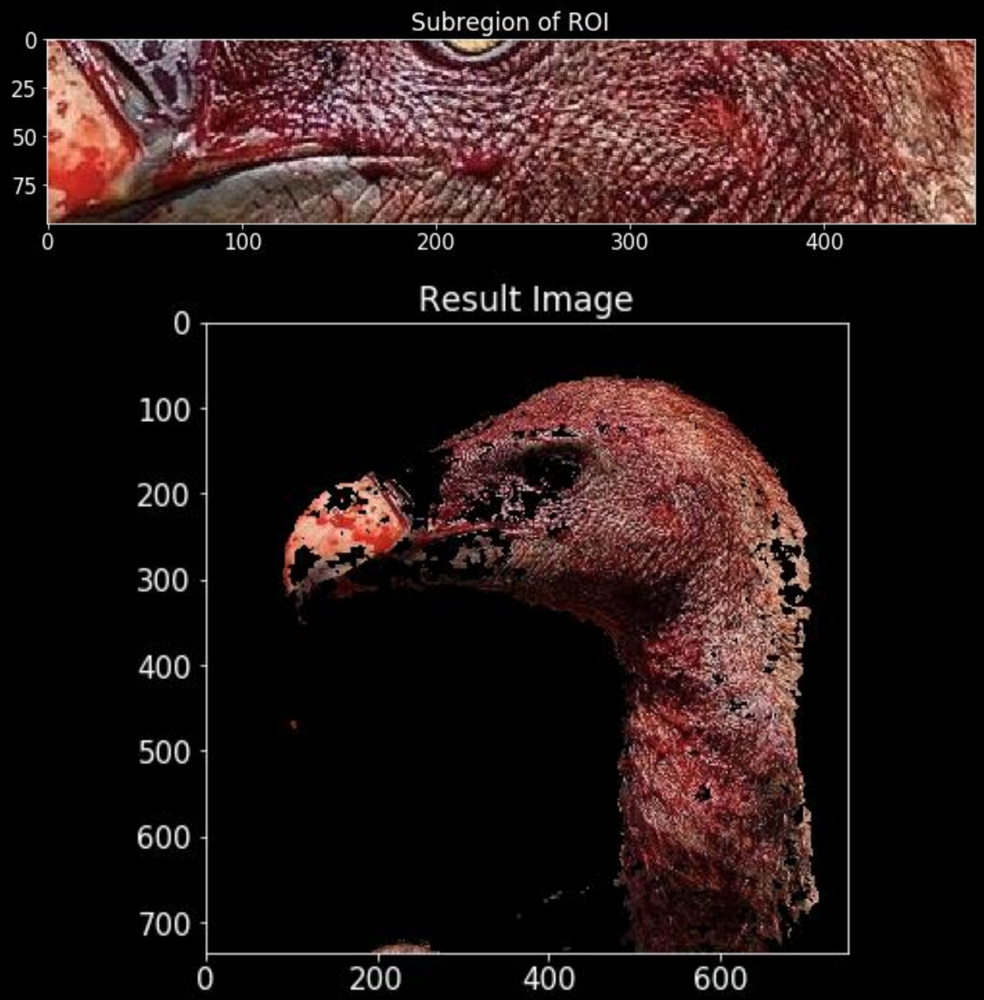




- There's a difference in the resulting image. The Vulture had lesser features on the left since the bottom of the eye was excluded in the patch

Step 2 Non-Parametric Results



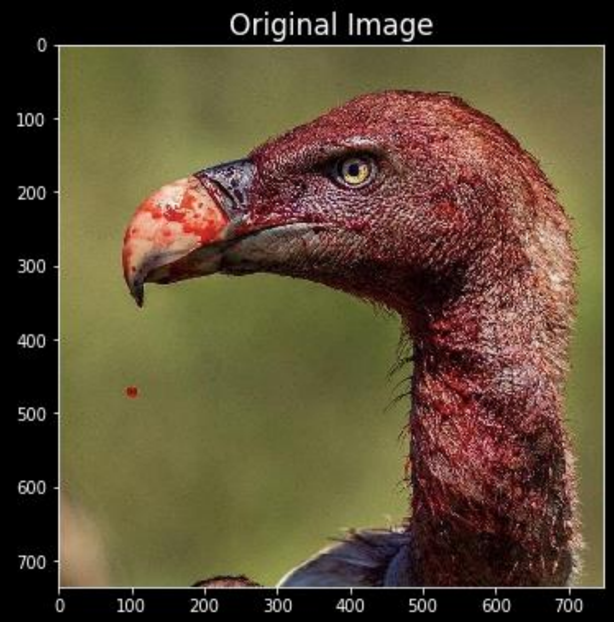


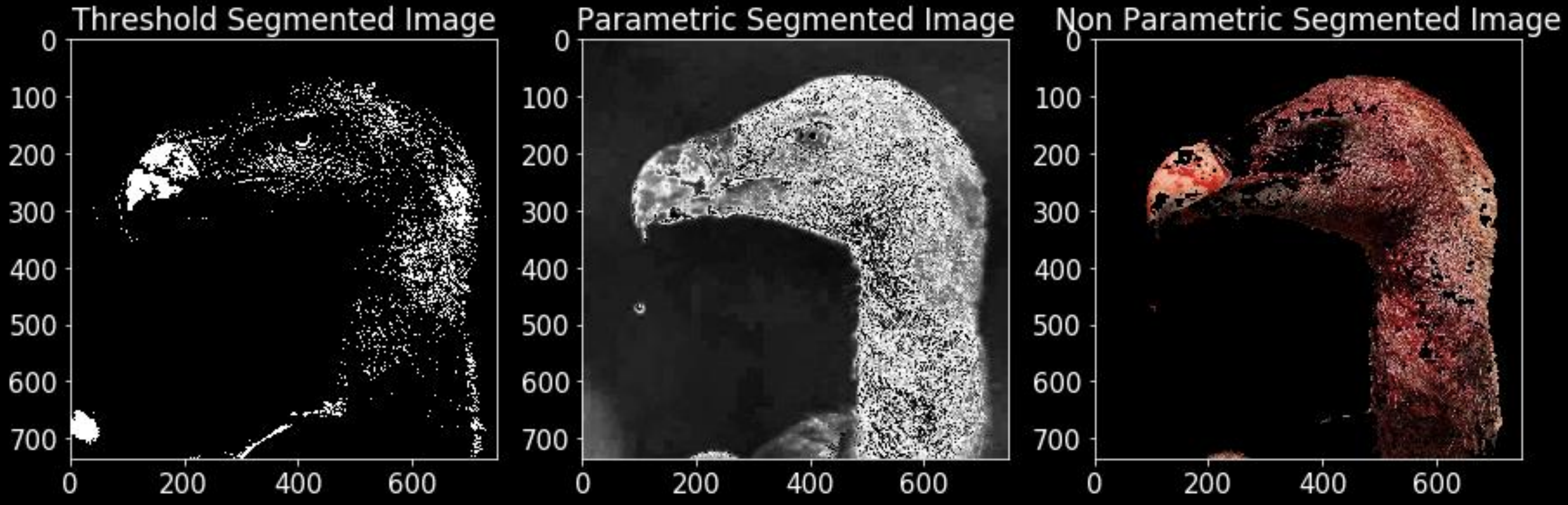
- This shows that the Non-parametric segmentation relies on the patch used- which can be a weakness and a strength. However, in this case, it is its weakness.

* Comparison

- Best segmented image: Parametric

* This may not be the case for other images. Let us try other colorful images as well.

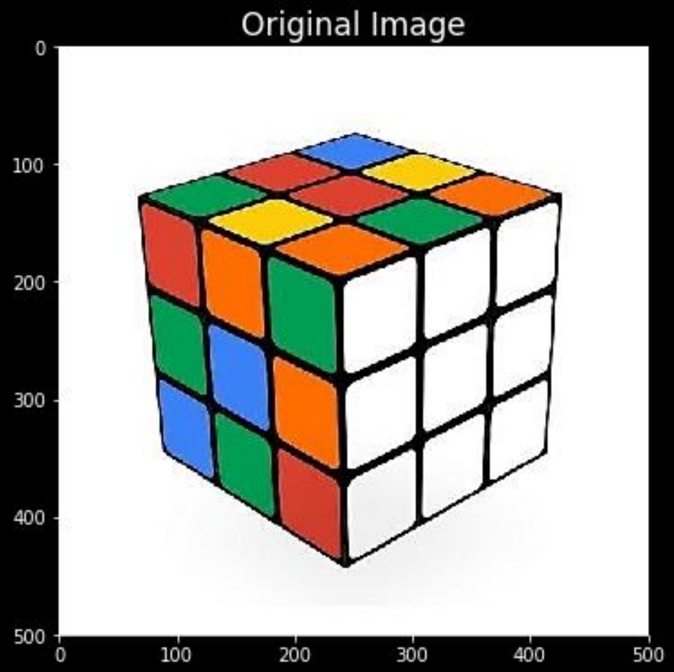



Threshold Segmented Image
Parametric Segmented Image
Non Parametric Segmented Image
0
100
200
300
400
500
600
700
0
200
400
600

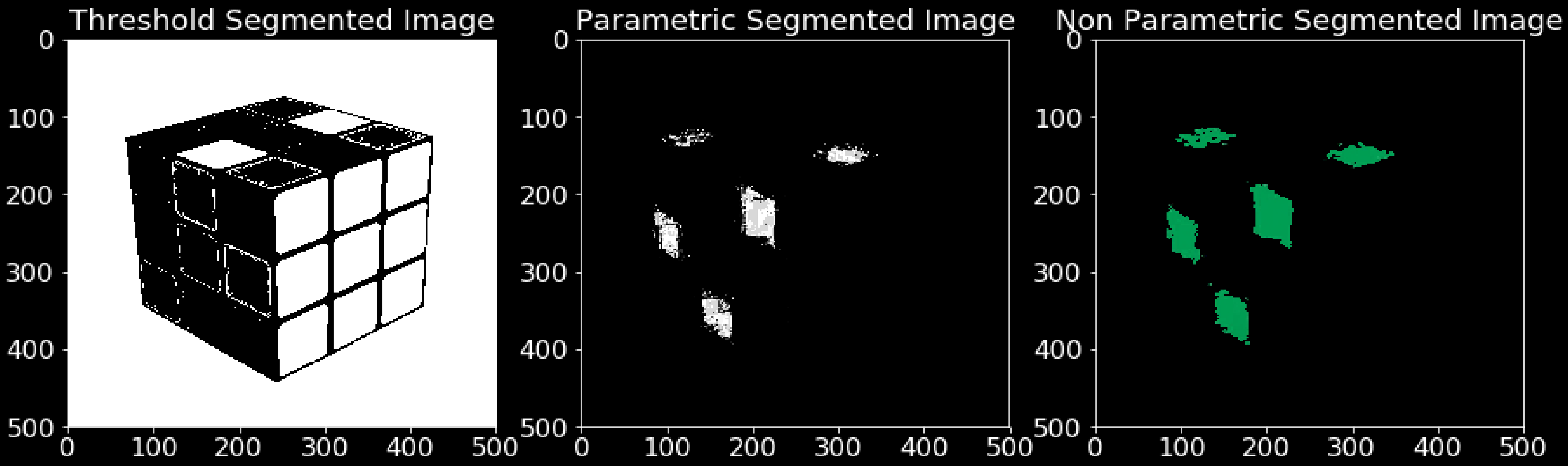
* Comparison

- Green cells:
- Best segmented image: Non - parametric

* Threshold can’t be used for this since it only works for grayscale



Original Image



Threshold Segmented Image

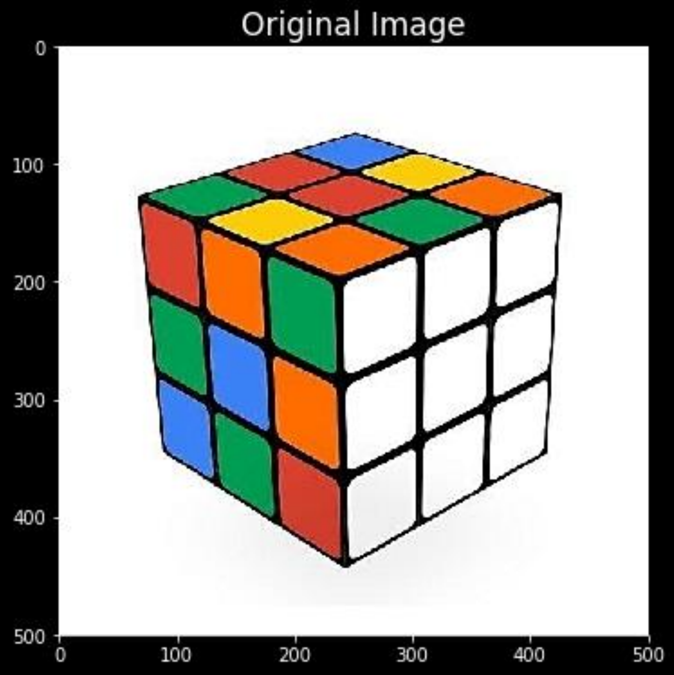
Parametric Segmented Image

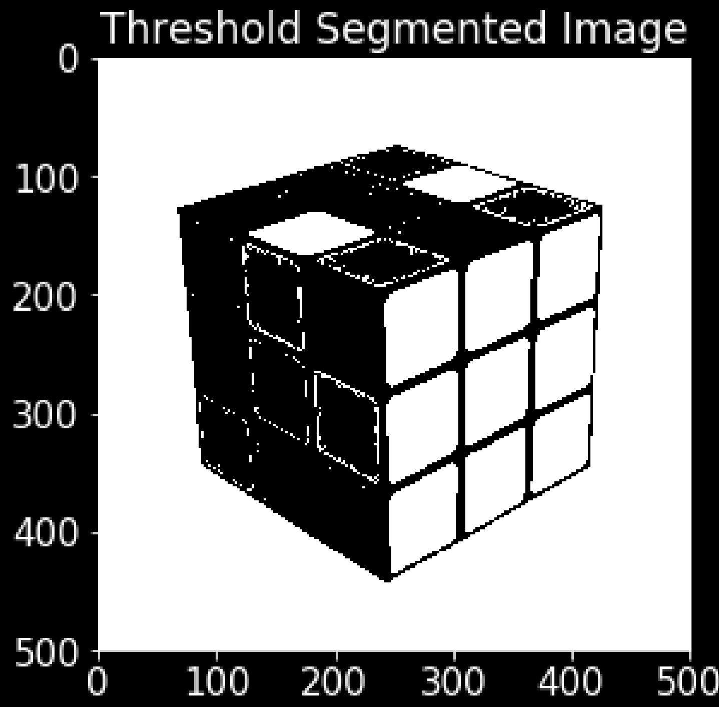
Non Parametric Segmented Image

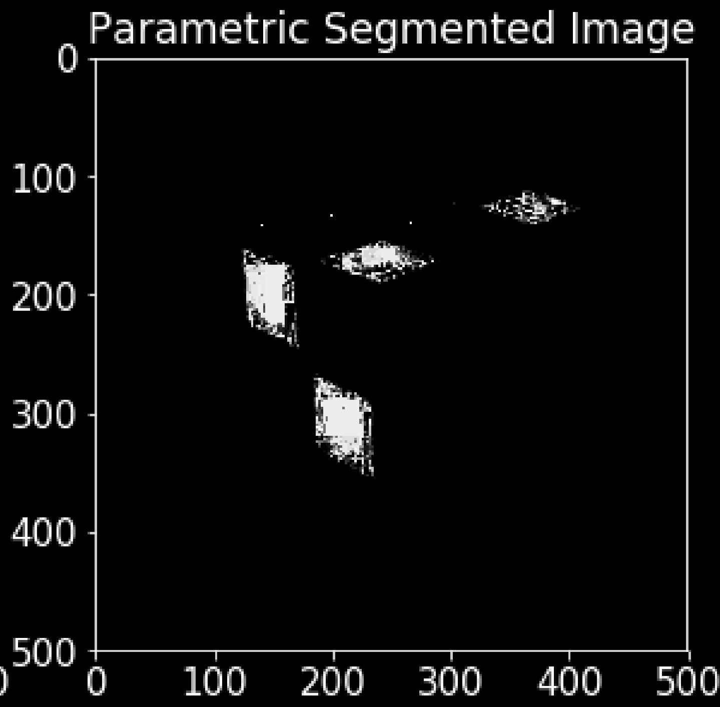
* Comparison

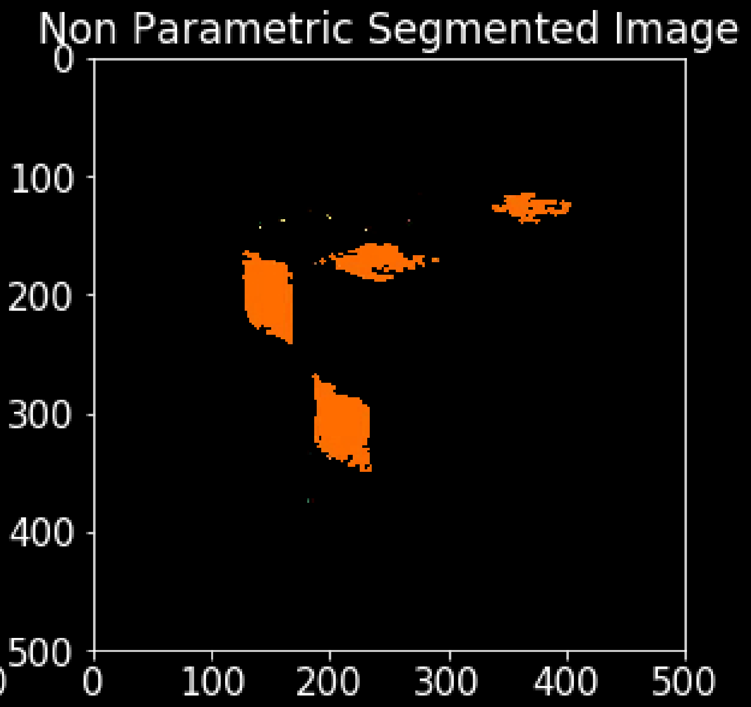
- Orange cells:
- Best segmented image: Non - parametric

* Threshold can’t be used for this since it only works for grayscale


Original Image


Threshold Segmented Image

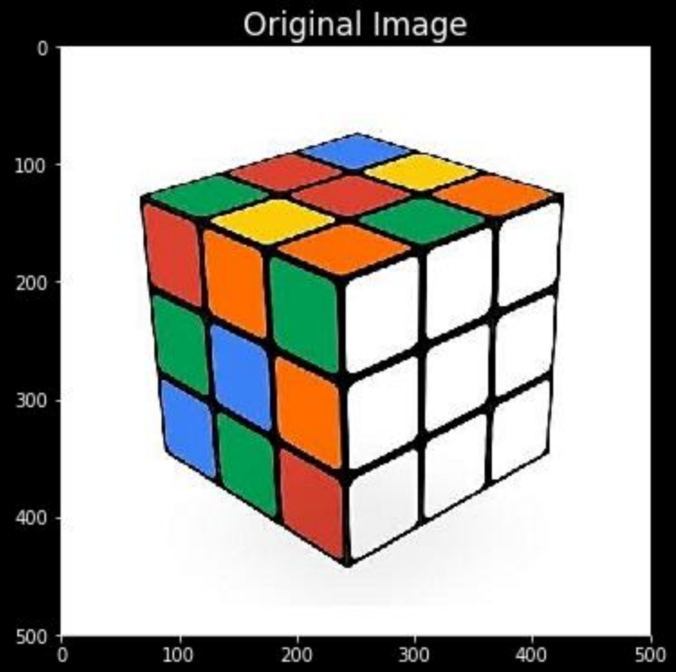

Parametric Segmented Image

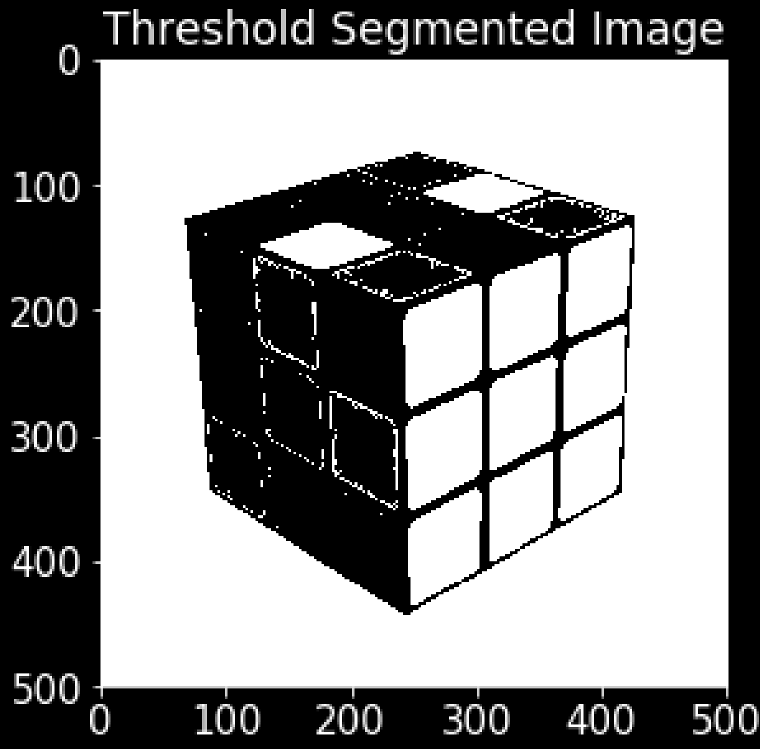

Non Parametric Segmented Image

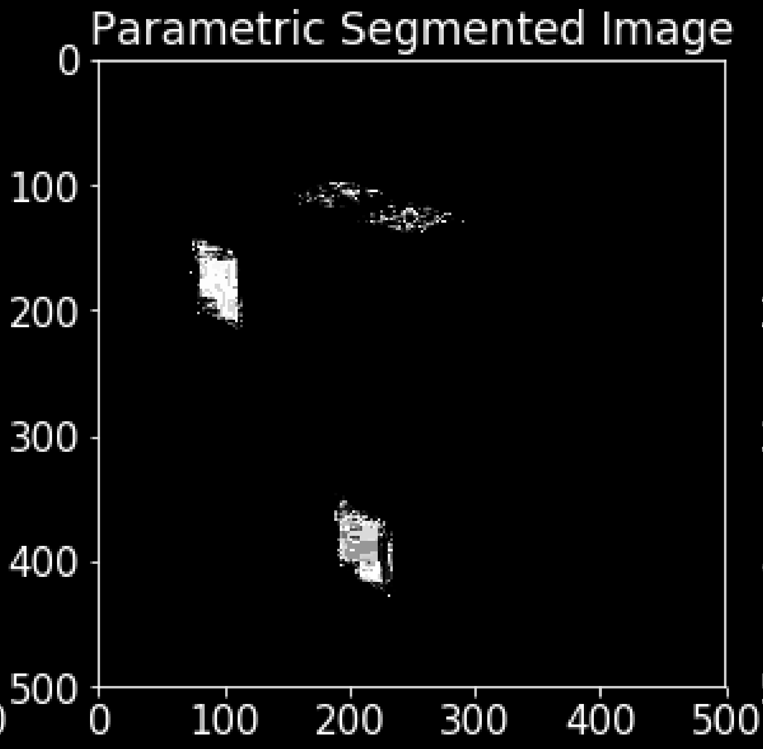
Comparison

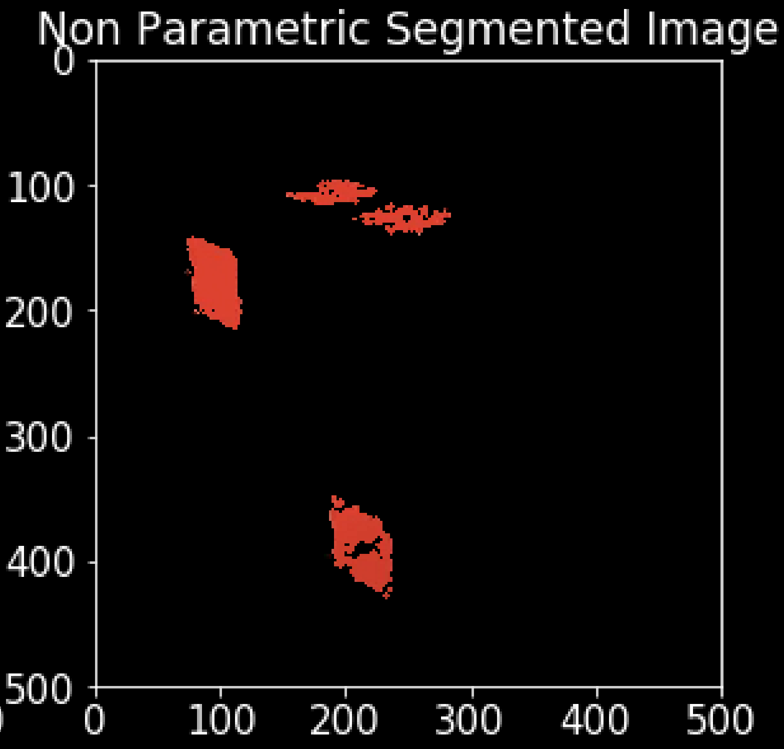
- Red cells:
- Best segmented image: Non - parametric

* Threshold can’t be used for this since it only works for grayscale


Original Image

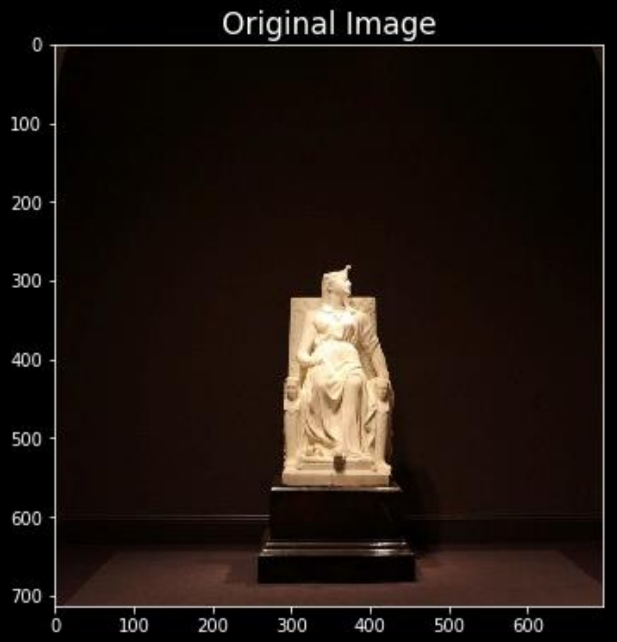

Threshold Segmented Image

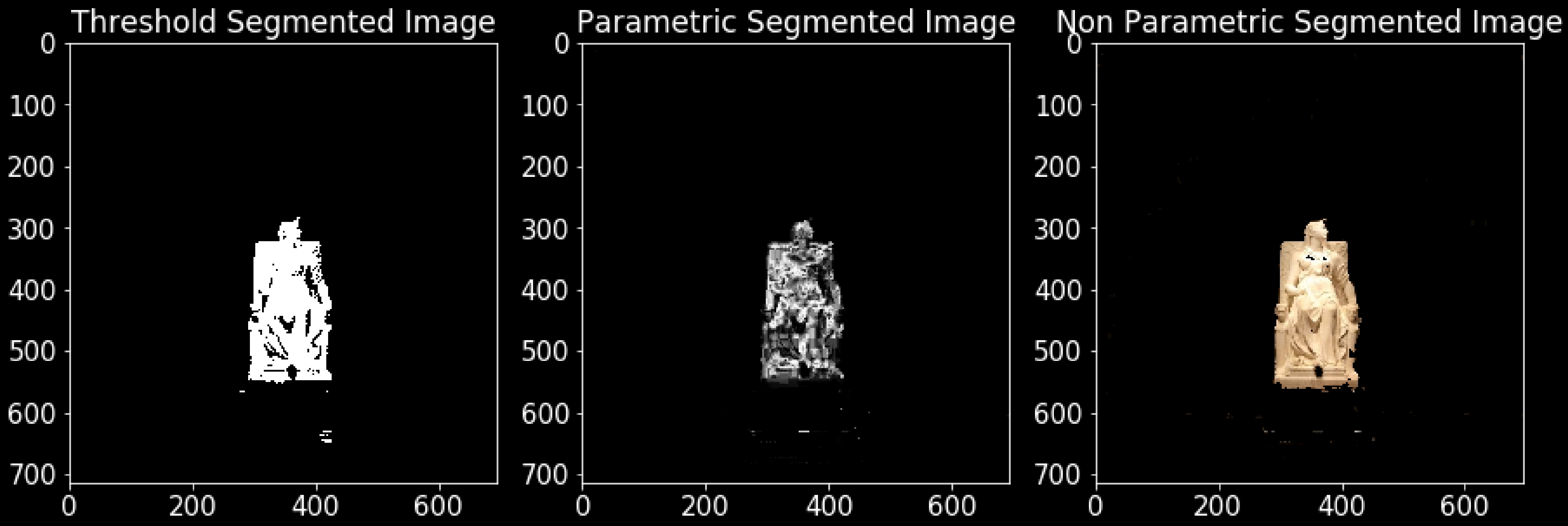

Parametric Segmented Image


Non Parametric Segmented Image

Comparison

Best segmented image: Non - parametric




Threshold Segmented Image
Parametric Segmented Image
Non Parametric Segmented Image
0
100
200
300
400
500
600
700
0
200
400
600

* Other pictures

- Concluding from other pictures used, the preferred segmentation method differ depending on the original image.
- Preference lies on the colors or pixel values the ROI has.
- Threshold works best for grayscale images (with contrasting backgrounds
- Parametric works best for detailed images whose histogram is similar to a Gaussian distribution
- Non-Parametric works better for segmenting certain colored object because of the extraction of desired patch.

* Pointssss

- TC : 5
- QP : 4
- IN : 1.....??

- This was so much fun ☺! Thank you!