INF 250 2022
Mandatory Exercise 2

Image Analysis

Submitted by: Sujan Devkota
Date: 4th November, 2022

Norwegian University of Life Sciences (NMBU)
Autumn Semester 2022

Mandatory2_Final

November 4, 2022

0.1 Introduction:

An image of chocolates has been provided. The image contains two types of chocolates which are M & m's and the Non stops. The task is to distinguish the Non stop chocolates from the M& m's or vice versa. M & m's are eliptical and slightly elongated chocolates whereas Non stops are round and circular. This knowledge has been utilized to compare the results after they have been distinguished from each other using various image processing techniques.

0.2 Theory and Background:

Various image processing techniques have been applied throughout this task. The image is converted to gray scale and the contrast of the image is improved by histogram equalization. Histogram equalization is the process of improving the contrast of the image by spreading out the most frequent intensity values (Sudhakar,2017). Histogram equalization increases the global contrast of the image where there are close contrast values. The image is then converted to binary image using thresholding. Thershold is calculated using various methods among which yen threshold gave the best results in this task. Morphological operations are carried out in the image which is the process of modifying the shape of an object using local filter operations like eroding, dilating, opening, and closing. Opening feature is used in this task. Opening is a morphological feature which performs an erosion operation followed by dilation. This operation makes the object smaller thus removing small objects (eroding) followed by making the object bigger(dilating).

0.3 Methods:

2 methods were used to distinguish the two kinds of chocolates; Non stops and M & m's which are both demonstrated in this report. At first, Fiji application was used to separate the two types of chocolates. And then, python was used to distinguish the two types of chocolates. The results from both the methods are compared in the Results section of the report. Fiji application and python was used to perform all the methods. Different image processing techniques were applied to analyze the chocolates.

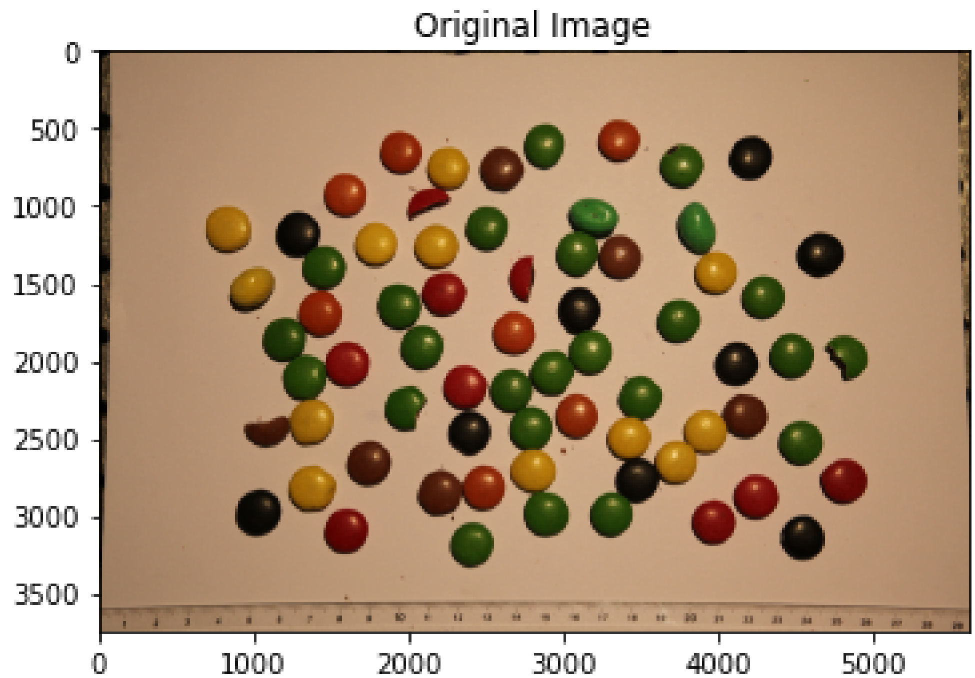
0.3.1 Using Fiji

Different processes was used using Fiji at first to differentiate the two types of chocolates; M & Ms and the nonstops. The images after each step were saved and are displayed in the following steps:

Loading the image The image was loaded in Fiji application.

[1]:
```python
import numpy as np
import pandas as pd
from skimage import io
import matplotlib.pyplot as plt
image = io.imread('IMG_2754_nonstop_alltogether.JPG')
plt.imshow(image)
plt.title('Original Image')
```

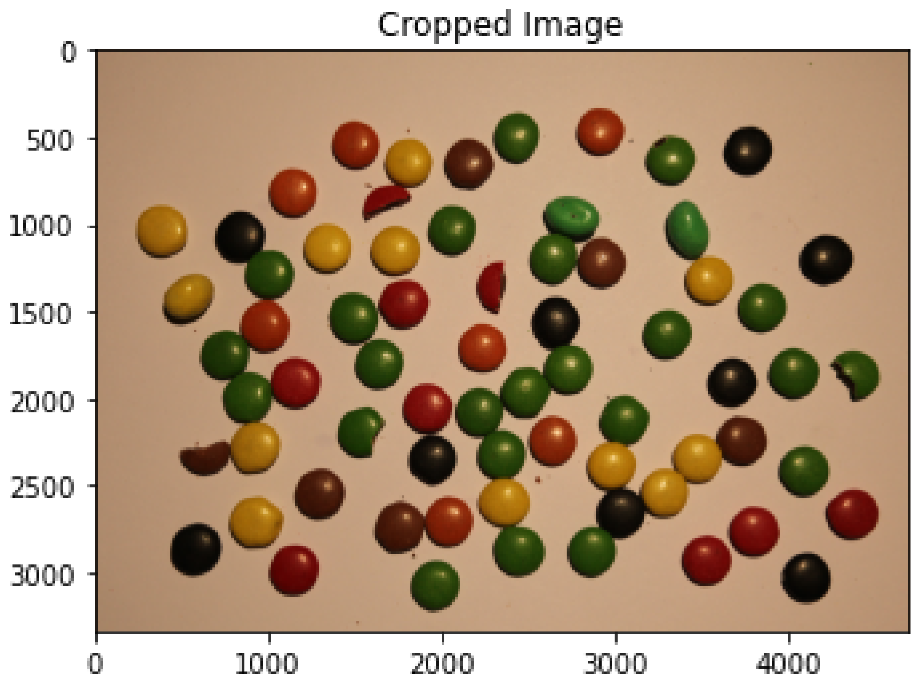
[1]: Text(0.5, 1.0, 'Original Image')



Cropping the Image The image is then cropped so that the unnecessary elements at the bottom of the image and the side of the image is eliminated so that it would be easier to analysize the image in the further steps. The cropped image is displayed below:

[2]:
```python
cropped_image = io.imread('cropped_image.tif')
plt.imshow(cropped_image)
plt.title('Cropped Image')
```

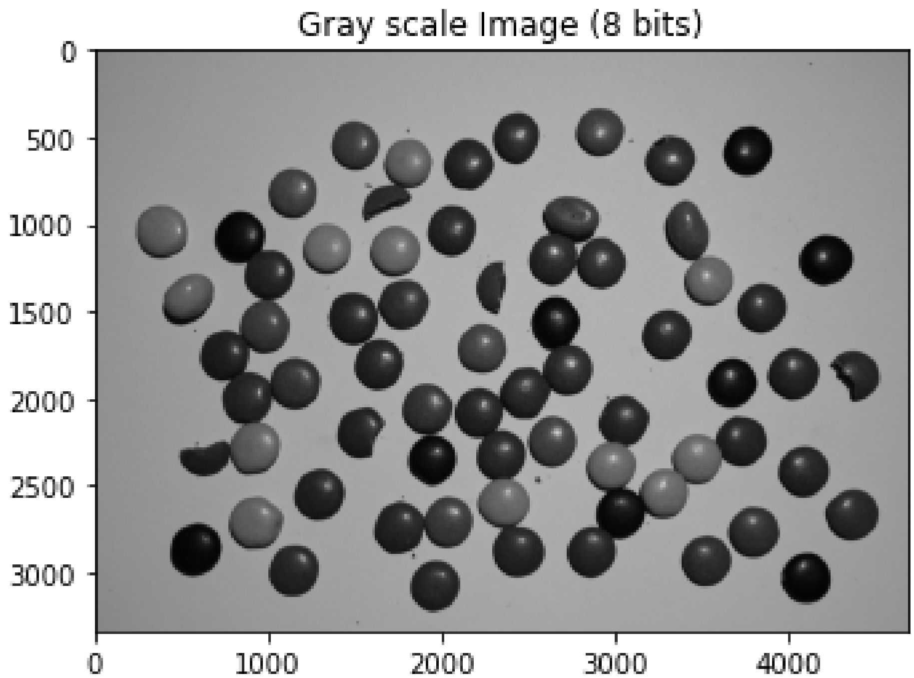
[2]: Text(0.5, 1.0, 'Cropped Image')



Converting the Image in 8 bits Gray scale Image The image is then converted into an 8 bits image in Fiji. The option is inside the type in the image section. The result is a 8 bits gray scale image with intensity range from 0 to 255. 0 represents black, 255 represents white and the other values in between are the different shades of grays. The image is converted into 8 bits to reduce the computational complexities that come with handling color image.

[3]:
```
grayscale_image = io.imread('8bits_image.tif')
plt.imshow(grayscale_image, cmap = 'gray')
plt.title('Gray scale Image (8 bits)')
```

[3]:
```
Text(0.5, 1.0, 'Gray scale Image (8 bits)')
```

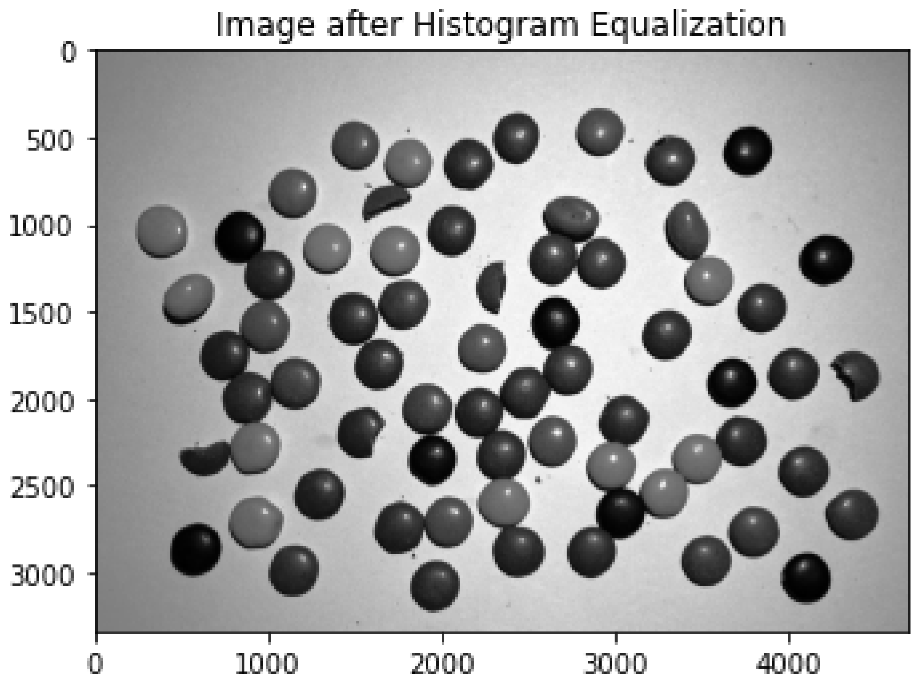


Histogram equalization Histogram equalization is the process of improving the contrast of the image by spreading out the most frequent intensity values (Sudhakar,2017). Histogram equalization increases the global contrast of the image where there are close contrast values.

The image is then histogram equalized using Fiji. This can also be done in python which is represented in the codes below. It is an important step in image processing as it can increase the contrast of the areas where there is lower local contrast.

[4]:
```
histequalized_image = io.imread('histequalized_image.tif')
plt.imshow(histequalized_image, cmap = 'gray')
plt.title('Image after Histogram Equalization')
```

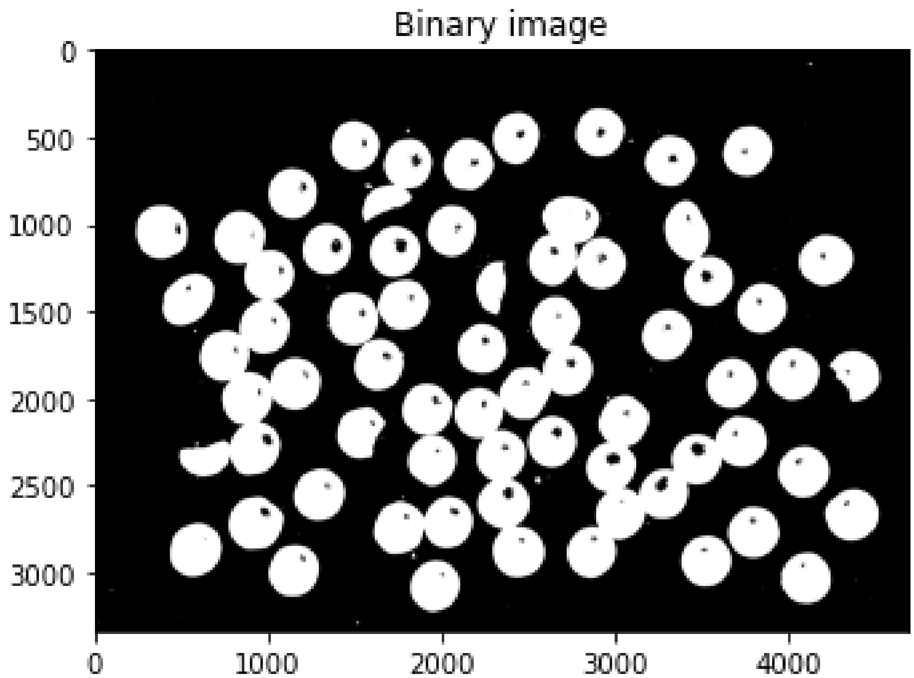
[4]:
```
Text(0.5, 1.0, 'Image after Histogram Equalization')
```



Converting the image to binary Image The image after histogram equalization is then converted into the binary image using Fiji. Binary image is an image which only has only 2 pixel values stored as a single bit which is 0 for black and 1 for white. The result is the image with black and white. The chocolates are represented as black and the background is white. It can also be done in python using the threshold value. In python, the best results is obtained using the yen thresholding.

[5]:
```
binary_image = io.imread('binary_image.tif')
plt.imshow(binary_image,cmap = 'gray')
plt.title('Binary image')
```

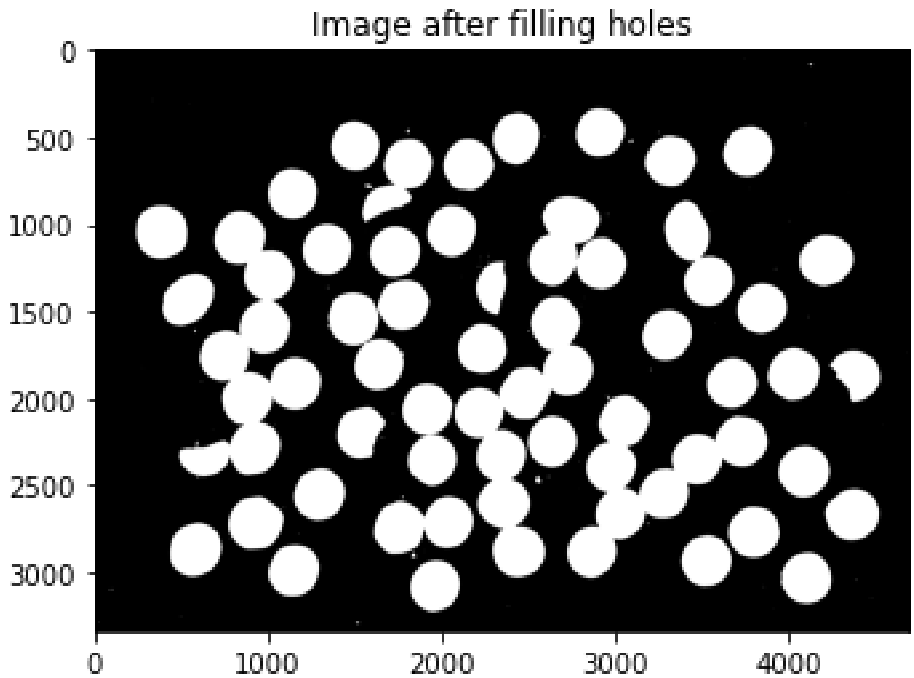
[5]: `Text(0.5, 1.0, 'Binary image')`



Filling Holes The binary image is seen to have small holes inside the chocolates. These holes must be filled before further processes. It is done in Fiji by going to the binary inside the process and doing fill holes.

[6]:
```
filled_holes_image = io.imread('filled_holes_image.tif')
plt.imshow(filled_holes_image, cmap = 'gray')
plt.title('Image after filling holes')
```

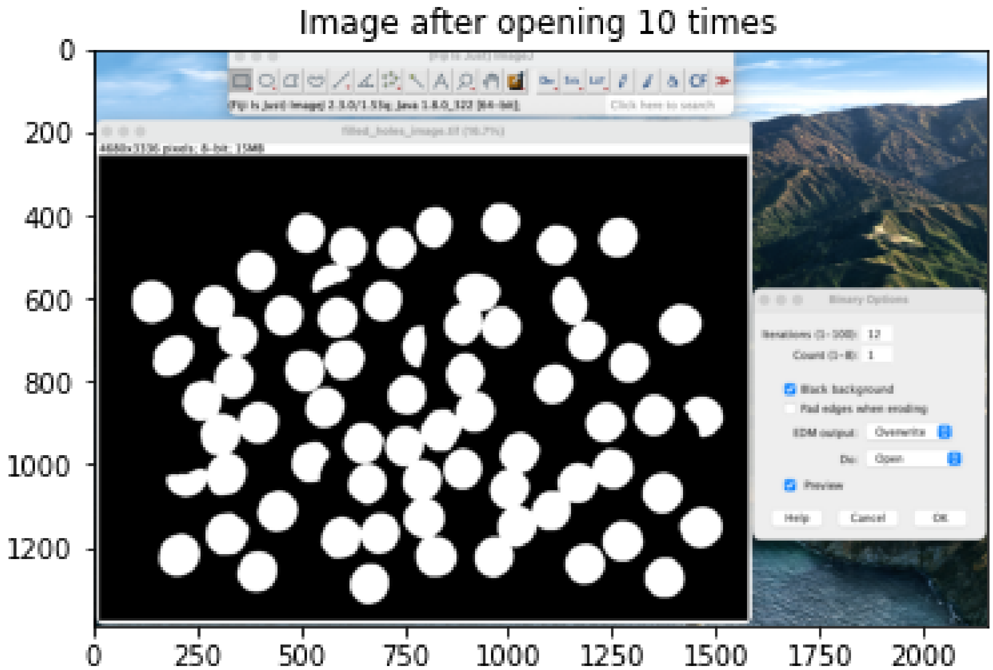
[6]:
```
Text(0.5, 1.0, 'Image after filling holes')
```



Using Opening feature of morphology The image after filling holes has small isolated objects or pixels. These objects must be removed as these objects are unnecessary in our analysis. Opening is a morphological feature which performs an erosion operation followed by dilation. The operation was performed ten times as all of the pixels was removed after doing the opening procedure ten times. The result of opening feature removed small pixels from the image.

[7]:
```
open_image = io.imread('open10times.jpg')
plt.imshow(open_image)
plt.title('Image after opening 10 times')
```

[7]:
```
Text(0.5, 1.0, 'Image after opening 10 times')
```

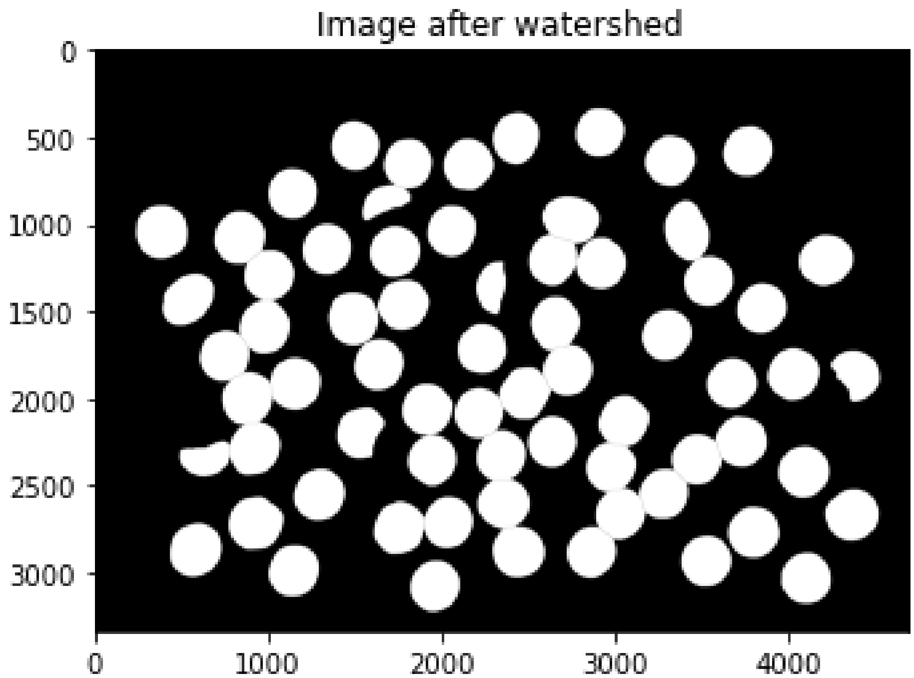


Using Watershed to separate the joined chocolates Watershed segmentation is a technique of image morphology for separating joined objects. In watershed segmentation, the bright pixels in the surface of the image is considered as the ridges and the darker pixels are considered as the valleys(Bernhard and Botha,111-175). The water is then filled from a minimum point and sheds are built so the water doesnt flood in from other areas. Water basins and floodlines are created from this process which is then used to separate the objects.

In Fiji, this can be done to the binary image. This process separated the chocolates that were connected previously. The chocolates after this process can be analyzed for different properties and can be used to distinguish the two types of chocolates.

[8]:
```
final_image = io.imread('watershed_image.tif')
plt.imshow(final_image, cmap = 'gray')
plt.title('Image after watershed')
```

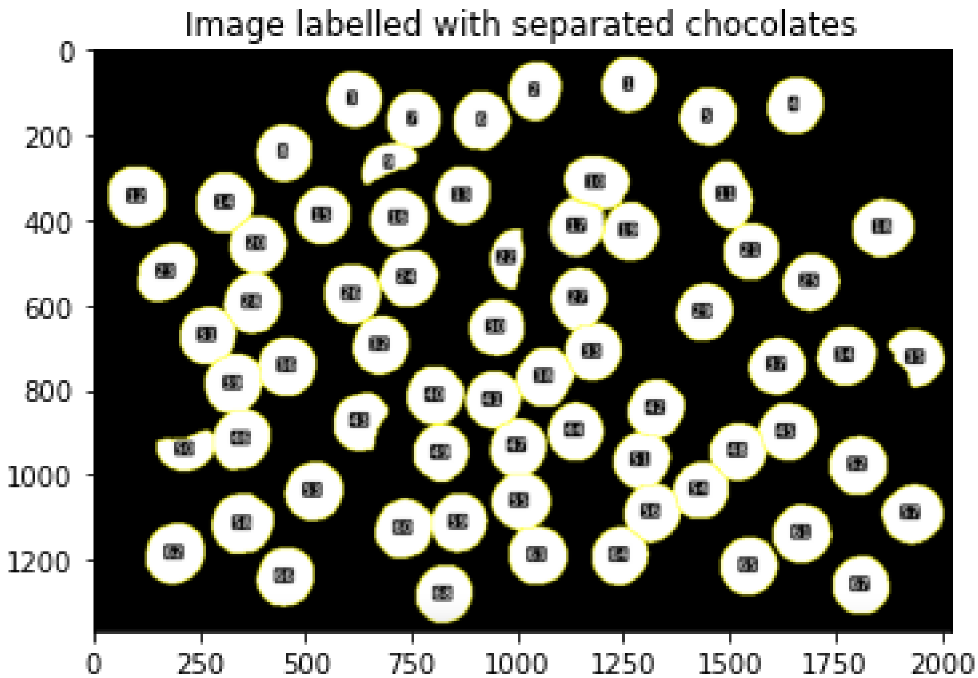
[8]:
```
Text(0.5, 1.0, 'Image after watershed')
```



Analyze particles The image is then analyzed using different parameters like roundness, circularity, area, perimeter, solidity. These parameters can be extracted from Fiji and the results are discussed in the Result section.

[9]:
```
final_labelled_image = io.imread('labelled_image.png')
plt.imshow(final_labelled_image,cmap = 'gray')
plt.title('Image labelled with separated chocolates')
```

[9]:
```
Text(0.5, 1.0, 'Image labelled with separated chocolates')
```



0.3.2 Using Python:

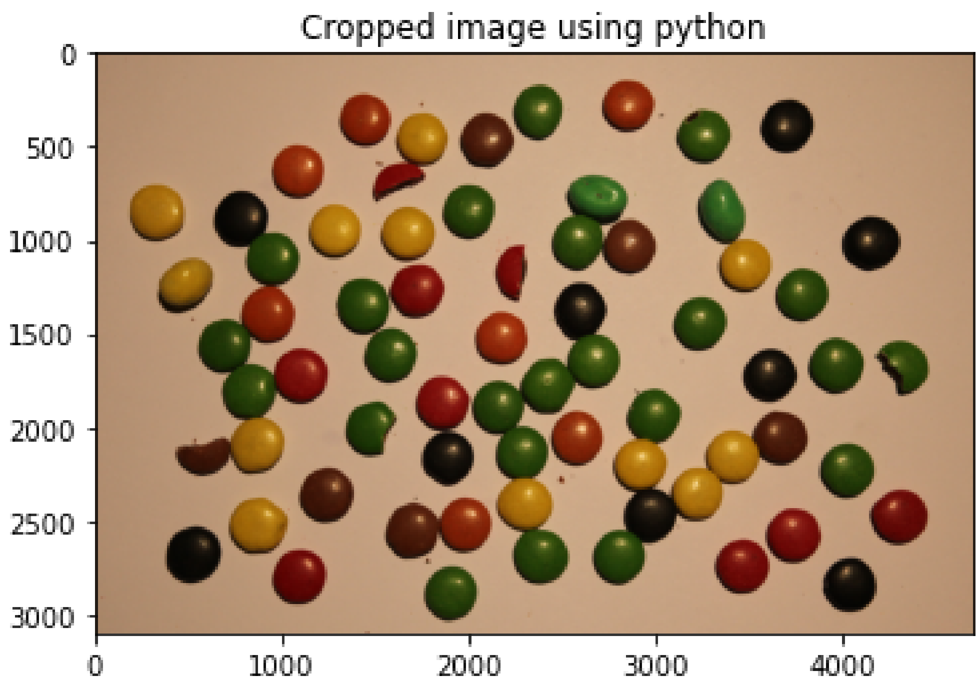
Python was used to distinguish the chocolates following similar methods applied on Fiji. The threshold value was calculated using the yen thresholding and the value was found to be 79. But setting the threshold value to 75 gave the best results. Therefore, 75 was used as the threshold value for converting the image to binary.

Importing necessary libraries, loading the image and cropping the image All the libraries necessary for the codes are imported, the image was read and was cropped to remove unnecessary elements in the image.

[10]:
```python
# Importing necessary libraries

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
pd.set_option('display.precision',3)
from skimage import io
from skimage.feature import peak_local_max
from scipy import ndimage as ndi
import skimage.morphology
import skimage.segmentation
import skimage.filters
from skimage import measure
import matplotlib.patches as mpatches
```

```
# Loading the image

image = io.imread('IMG_2754_nonstop_alltogether.JPG')

# Cropping the image
croppedimage = image[300:3400,500:5200]
plt.imshow(croppedimage)
plt.title('Cropped image using python')
```

[10]: Text(0.5, 1.0, 'Cropped image using python')



Doing the histogram equalization The image was first converted to grayscale image using imagemean and then a histogram equalization was done on the image. The python code for histogram equalization is displayed below:

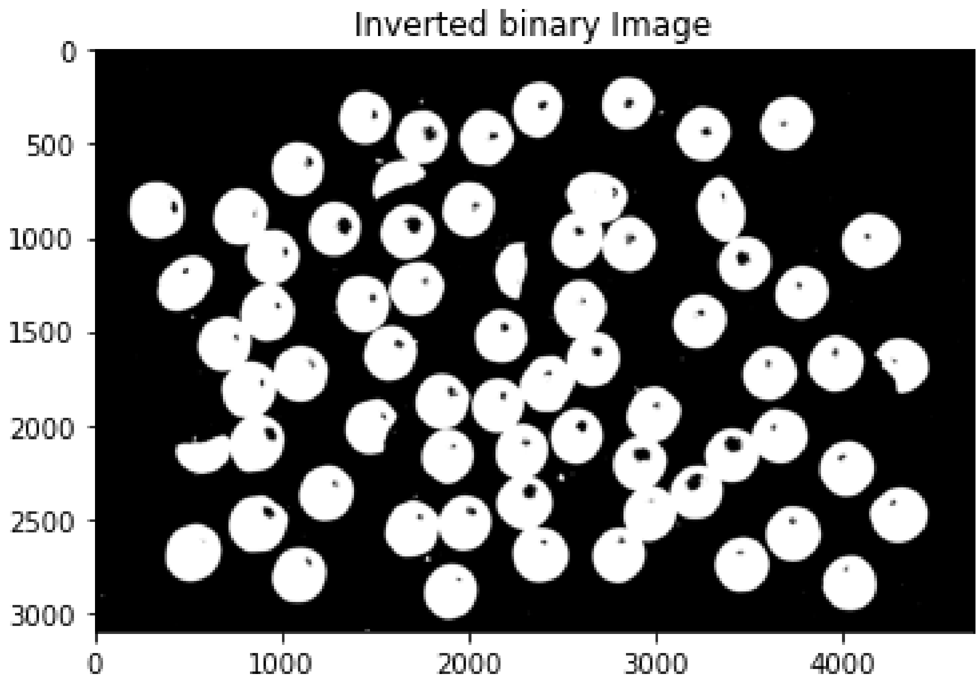
[11]:
```
# Histogram Equalization
imagemean = croppedimage.mean(axis = 2)
shape = np.shape(imagemean)
K = 256
M = shape [0]*shape[1]
histogram = np.zeros(256)

for i in range(shape[0]):
```

```
    for j in range(shape[1]):
        pixval = int(imagemean[i,j])
        histogram[pixval] +=1

cumhist = np.zeros(256)
cumhist[0] = histogram[0]

for i in range(255):
    cumhist[i+1] = cumhist[i] + histogram[i+1]

for i in range(shape[0]):
    for j in range(shape[1]):
        a = int(imagemean[i,j])
        b = cumhist[a]*(K-1)/M
        imagemean[i,j] = b
```

Converting the image into Binary image and inverting the image The grayscale image after histogram equalization is converted to binary image using the threshold value as 75. The yen threshold gave the value as 79.18 but setting the value to 75 gave the best results.

[12]:
```
# Making the image binary using thresholding
automatic_threshold = skimage.filters.threshold_yen(imagemean)
# threshold calculated using yen thersholding (79.18) but setting the threshold␣
 ↪to 75 gave the best results
imagebinary = imagemean > 75

#Inverting the binary image
invert_imagebinary = (imagebinary == False)
plt.imshow(invert_imagebinary, cmap = 'gray')
plt.title('Inverted binary Image')
```

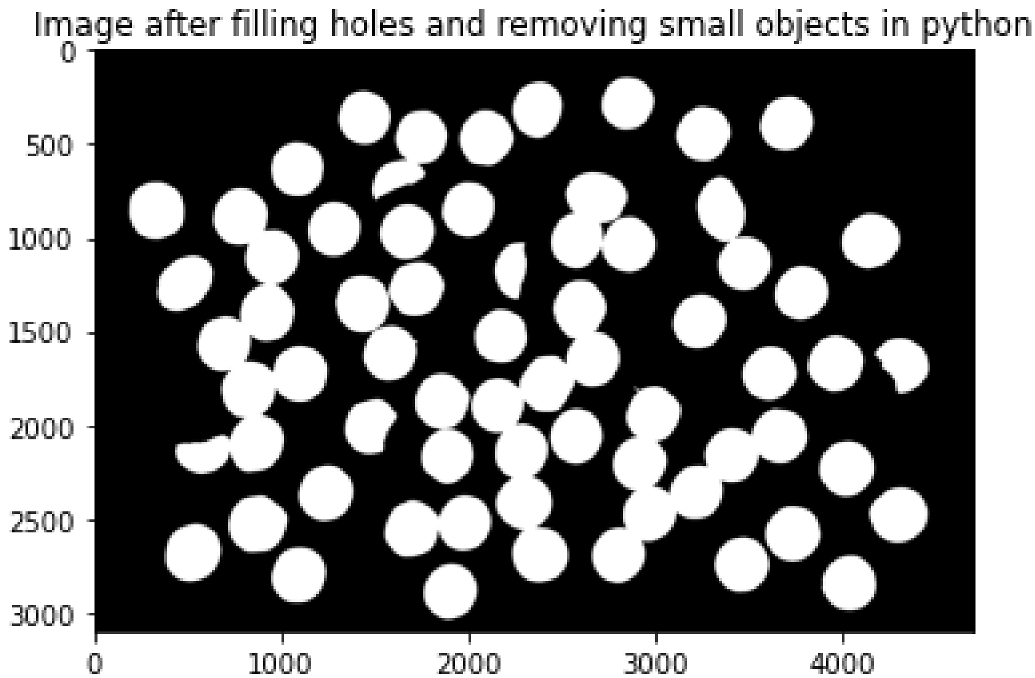
[12]:
```
Text(0.5, 1.0, 'Inverted binary Image')
```



Filling the holes using Skimage morphological features The binary image is seen to have holes and small objects. The holes were filled using the skimage morphology module called remove_small_holes. Similarly, the small pieces of chocolates are removed using skimage morphology module called remove_small_objects.

[13]:
```
# Filling the holes and removing small pixels/objects

filledholes_image = skimage.morphology.remove_small_holes(invert_imagebinary,␣
 ↪area_threshold = 8000)
finalimage = skimage.morphology.remove_small_objects(filledholes_image, 1000)
plt.title('Image after filling holes and removing small objects in python')
plt.imshow(finalimage,cmap = 'gray')
```

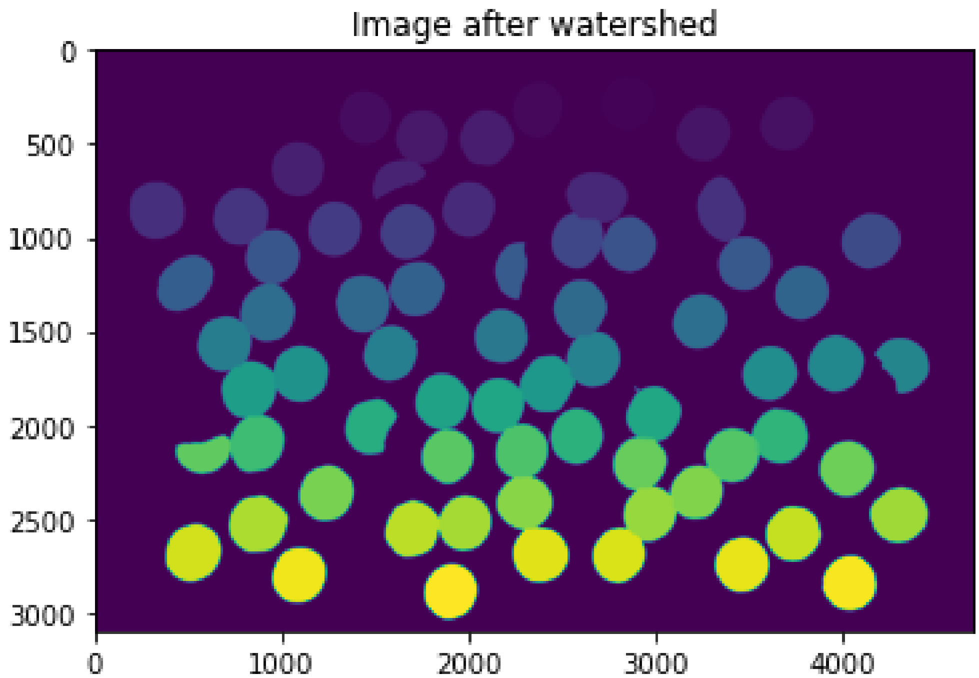
[13]:
```
<matplotlib.image.AxesImage at 0x298557eb0>
```



Doing the watershed on the image using skimage segmentation module called watershed

[14]:
```
# Watershed to separate joined objects
distance = ndi.distance_transform_edt(finalimage)
# min_distance set to 100
max_coords = peak_local_max(distance,min_distance = 100, labels =␣
↪finalimage,footprint = np.ones((3,3)))
local_maxima = np.zeros_like(finalimage,dtype = bool)
local_maxima[tuple(max_coords.T)]= True

markers = ndi.label(local_maxima)[0]
labels = skimage.segmentation.watershed(-distance,markers, mask = finalimage)

# Showing the plot after watershed
plt.title('Image after watershed')
plt.imshow(labels)
```

[14]: <matplotlib.image.AxesImage at 0x2af143df0>



Calculating properties for each region:

[15]:
```
# Calculating properties
properties = measure.regionprops(labels)
props = measure.regionprops(labels, finalimage)
results = {}
label = []
area = []
perimeter = []
circularity =[]
roundness = []
solidity = []
eccentricity = []
centroid = []

for prop in properties:
    if prop.area > 3000:
        label.append(prop.label)
        area.append(prop.area)
        perimeter.append(prop.perimeter)
        circularity.append((4*3.14*prop.area)/(prop.perimeter**2))
        roundness.append(4*(prop.area/(3.14*prop.axis_major_length**2)))
        solidity.append(prop.solidity)
        eccentricity.append(prop.eccentricity)
        centroid.append(prop.centroid)
```

```
# Adding the properties in the dictionary
results ={'labels': label,
            'area':area, 'perimeter': perimeter,
             'circularity': circularity,
                 'roundness':roundness,'solidity':solidity, 'eccentricity':
 ↪eccentricity,
                     'centroid': centroid}
```

0.4 Results :

The results from both the methods using Fiji and Python are extracted and are converted to pandas dataframe to analyze the chocolates.

0.4.1 Results from Fiji

The results obtained from analyzing particles in the last step using Fiji was extraced as a csv file which was then converted into a pandas dataframe as shown below.

0.4.2 Converting the results obtained from fiji into pandas dataframe

[16]:
```
results_from_fiji = pd.read_csv('finalResults.csv')
results_from_fiji
```

[16]:
```
                                   Label   Area  Mean          X         Y  \
0    1  IMG_2754_nonstop_alltogether.JPG  60398   255  2834.530   263.263
1    2  IMG_2754_nonstop_alltogether.JPG  61146   255  2353.619   296.370
2    3  IMG_2754_nonstop_alltogether.JPG  60814   255  1430.270   340.055
3    4  IMG_2754_nonstop_alltogether.JPG  63275   255  3683.786   371.702
4    5  IMG_2754_nonstop_alltogether.JPG  64347   255  3238.792   426.198
..  ..                               …      …    …         …         …
63  64  IMG_2754_nonstop_alltogether.JPG  63627   255  2785.931  2670.850
64  65  IMG_2754_nonstop_alltogether.JPG  65623   255  3444.534  2720.339
65  66  IMG_2754_nonstop_alltogether.JPG  67168   255  1079.166  2775.717
66  67  IMG_2754_nonstop_alltogether.JPG  65440   255  4018.846  2819.207
67  68  IMG_2754_nonstop_alltogether.JPG  64826   255  1888.646  2863.236

          XM        YM   Perim.   Circ.     AR  Round  Solidity
0   2834.530   263.263  915.862  0.905  1.038  0.964     0.993
1   2353.619   296.370  927.519  0.893  1.143  0.875     0.993
2   1430.270   340.055  922.548  0.898  1.031  0.970     0.993
3   3683.786   371.702  942.105  0.896  1.081  0.925     0.993
4   3238.792   426.198  950.975  0.894  1.054  0.949     0.993
..       …         …       …      …      …     …         …
63  2785.931  2670.850  943.217  0.899  1.049  0.953     0.992
64  3444.534  2720.339  955.318  0.904  1.023  0.978     0.994
65  1079.166  2775.717  968.933  0.899  1.074  0.931     0.994
```

```
66  4018.846  2819.207  950.833  0.910  1.021  0.980     0.993
67  1888.646  2863.236  949.904  0.903  1.063  0.941     0.994

[68 rows x 13 columns]
```

0.4.3 Extracting chocolates which have roundness less than 0.87 using the results from Fiji

It can be seen from the table that there are 68 chocolates which contains M & m's and non stops chocolates. Some of these chocolates are half eaten so the parameters like roundness and circularity is used to differentiate between all the chocolates. At first, the chocolates which have the roundness less than 0.87 was extracted as below.The result obtained from setting that parameter is 8 chocolates. Out of these 8 chocolates, some are half eaten and the remaining are M & m's chocolate. The chocolates which have the roundness greater than 0.87 are considered to be round and are non stop chocolates. To differentiate between the half eaten and the M & m's chocolate, a different paramater must be set.

[17]:
```
results_from_fiji1 = results_from_fiji[results_from_fiji['Round'] < 0.87]
results_from_fiji1
```

[17]:
```
                                         Label   Area  Mean         X         Y  \
8    9  IMG_2754_nonstop_alltogether.JPG  34794   255  1599.695   654.144
9   10  IMG_2754_nonstop_alltogether.JPG  67601   255  2670.182   761.341
10  11  IMG_2754_nonstop_alltogether.JPG  66097   255  3334.224   830.458
21  22  IMG_2754_nonstop_alltogether.JPG  37752   255  2213.844  1150.108
22  23  IMG_2754_nonstop_alltogether.JPG  66235   255   471.088  1221.405
34  35  IMG_2754_nonstop_alltogether.JPG  55329   255  4310.791  1647.664
42  43  IMG_2754_nonstop_alltogether.JPG  58829   255  1455.858  1981.441
49  50  IMG_2754_nonstop_alltogether.JPG  42371   255   574.621  2136.050

          XM        YM   Perim.  Circ.     AR  Round  Solidity
8   1599.695   654.144  804.080  0.676  1.995  0.501     0.944
9   2670.182   761.341  995.075  0.858  1.271  0.787     0.986
10  3334.224   830.458  987.845  0.851  1.400  0.715     0.992
21  2213.844  1150.108  810.299  0.723  1.841  0.543     0.976
22   471.088  1221.405  976.975  0.872  1.346  0.743     0.992
34  4310.791  1647.664  934.874  0.796  1.291  0.775     0.954
42  1455.858  1981.441  943.619  0.830  1.254  0.797     0.963
49   574.621  2136.050  844.825  0.746  1.678  0.596     0.953
```

0.4.4 Extracting chocolates which have circularity greating than 0.85 using the results from Fiji

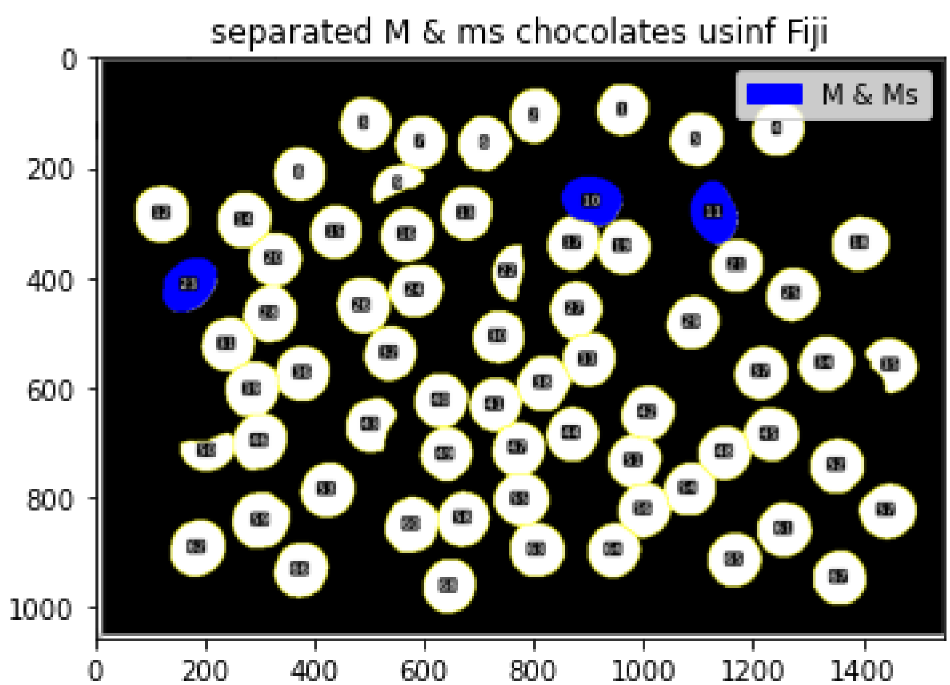
In the next step, circularity was used to eliminate or distinguish the half eaten chocolates from M & M's. The chocolates which had the circularity less than 0.85 were considered half eaten and therefore the chocolates which have the circularity greater than 0.85 among the 8 chocolates are the M & m's and the rest are Non stop chocolates. The result obtained is 3 chocolates after this parameter which are the M & m's. This result can also be verified from observing the image very closely as these chocolates are slightly elongated. The remaining 5 are half eaten chocolates and the remaining chocolates are Non stops chocolate.

[18]:
```
results_from_fiji2 = results_from_fiji1[results_from_fiji1['Circ.'] > 0.85]
results_from_fiji2
```

[18]:
```
                              Label   Area  Mean         X         Y  \
9   10  IMG_2754_nonstop_alltogether.JPG  67601   255  2670.182   761.341
10  11  IMG_2754_nonstop_alltogether.JPG  66097   255  3334.224   830.458
22  23  IMG_2754_nonstop_alltogether.JPG  66235   255   471.088  1221.405

          XM        YM   Perim.  Circ.     AR  Round  Solidity
9   2670.182   761.341  995.075  0.858  1.271  0.787     0.986
10  3334.224   830.458  987.845  0.851  1.400  0.715     0.992
22   471.088  1221.405  976.975  0.872  1.346  0.743     0.992
```

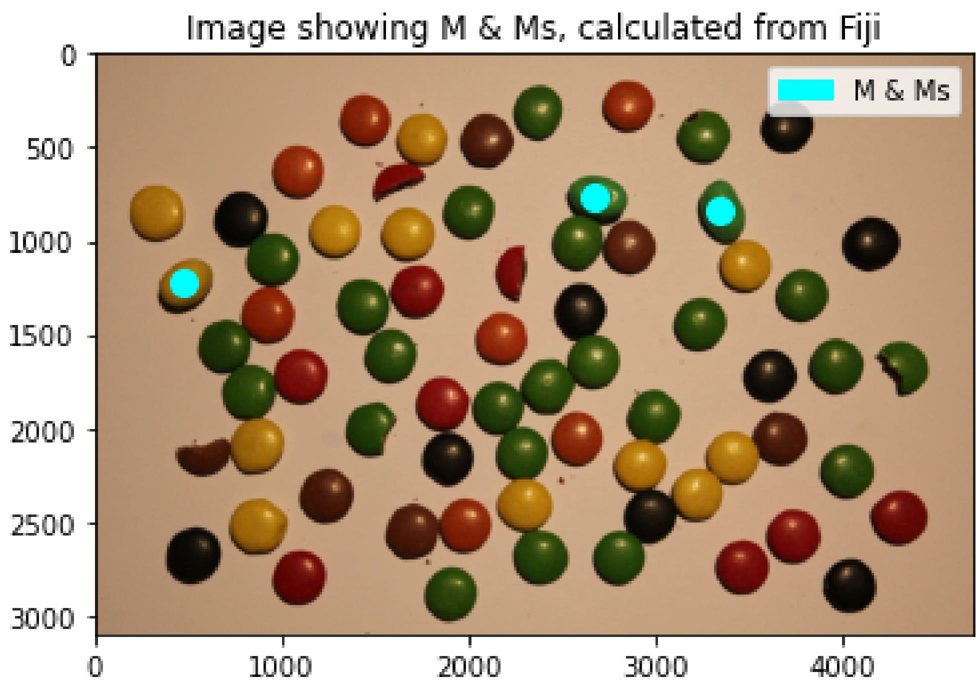
[19]:
```
separated_image = io.imread('separated_image.png')
plt.title('separated M & ms chocolates usinf Fiji')
plt.imshow(separated_image)
plt.title = ('Image with highlighted M & Ms using Fiji')
cyan_patch = mpatches.Patch(color = 'blue', label = 'M & Ms')
plt.legend(handles = [cyan_patch],loc = 'upper right')
```

[19]: <matplotlib.legend.Legend at 0x2af1914e0>



[22]:
```
plt.imshow(croppedimage)
plt.scatter(2670.182,761.341, s=100, c='cyan', marker='o')
plt.scatter(3334.224,830.458, s=100, c='cyan', marker='o')
plt.scatter(471.088,1221.405, s=100, c='cyan', marker='o')

cyan_patch = mpatches.Patch(color = 'cyan', label = 'M & Ms')
plt.legend(handles = [cyan_patch],loc = 'upper right')
plt.title('Image showing M & Ms, calculated from Fiji')
```

[22]:
```
Text(0.5, 1.0, 'Image showing M & Ms, calculated from Fiji')
```



0.4.5 Results from python:

The properties obtained from python was converted to the pandas dataframe. In python, the circularity and roundness was calculated by using the formulas which are

circularity = 4 * pi * Area / perimeter ^2 Roundness = 4 * Area / pi * (major axis length)^2

and based on the same criteria we used using Fiji, the chocolates are separated. The two criterias used are

Roundness $<$ 0.87 and Circularity $>$ 0.85

0.4.6 Converting the results obtained from python into pandas dataframe

[23]:
```
results_from_python = pd.DataFrame(results)
results_from_python
```

[23]:
```
    labels   area  perimeter  circularity  roundness  solidity  eccentricity  \
0        1  60457    918.105        0.901      0.964     0.995         0.267
1        2  61193    928.448        0.892      0.875     0.995         0.484
2        3  60902    924.791        0.894      0.970     0.995         0.244
3        4  63361    943.519        0.894      0.925     0.995         0.380
4        5  64409    953.318        0.890      0.949     0.995         0.315
..     …      …         …           …          …        …             …
63      86  68871    983.862        0.894      0.971     0.991         0.237
64      87  65717    959.561        0.896      0.978     0.995         0.211
65      88  67215    968.690        0.900      0.931     0.995         0.365
66      89  65543    957.318        0.898      0.981     0.994         0.196
67      90  64899    952.146        0.899      0.941     0.995         0.339

                                   centroid
0   (286.73025456109303, 2844.0775757976744)
1     (319.8706387985554, 2363.148709819751)
2   (363.60723785754163, 1439.7866408328134)
3    (395.2430675020912, 3693.2939347548177)
4   (449.70678010836997, 3248.2855191044728)
..                                        …
63  (2690.9218103410726, 2376.8402811052547)
64  (2743.8364654503403, 3454.0150341616322)
65   (2799.182652681693, 1088.6764561481812)
66  (2842.7306653647224, 4028.2600430251896)
67   (2886.711983235489, 1898.1619439436663)

[68 rows x 8 columns]
```

0.4.7 Extracting chocolates which have roundness less than 0.87 using the results from python

[24]:
```
results_from_python1 = results_from_python[results_from_python['roundness']< 0.
↪87]
results_from_python1
```

[24]:
```
    labels   area  perimeter  circularity  roundness  solidity  eccentricity  \
8        9  34990    825.335        0.645      0.470     0.936         0.866
9       10  67748   1000.004        0.851      0.783     0.978         0.617
12      13  66180    992.087        0.845      0.712     0.993         0.700
21      26  37937    825.411        0.699      0.531     0.972         0.840
22      27  66359    983.703        0.861      0.743     0.994         0.670
33      41  55501    949.016        0.774      0.751     0.954         0.632
```

```
42      56  58981     958.525         0.806       0.783      0.966          0.602
49      68  42630     877.566         0.695       0.569      0.953          0.802

                                    centroid
8    (678.0979994284081, 1609.1311803372391)
9     (784.8431392808644, 2679.244553344748)
12     (853.969734058628, 3343.703566032034)
21  (1173.5777736774126, 2223.5939847642144)
22   (1244.922346629696, 480.60323392456183)
33    (1671.160087205636, 4319.980360714221)
42     (2005.0063579796883, 1465.5461080687)
49   (2159.2878489326763, 584.0422472437251)
```

0.4.8 Extracting chocolates which have circularity greating than 0.85 using the results from python

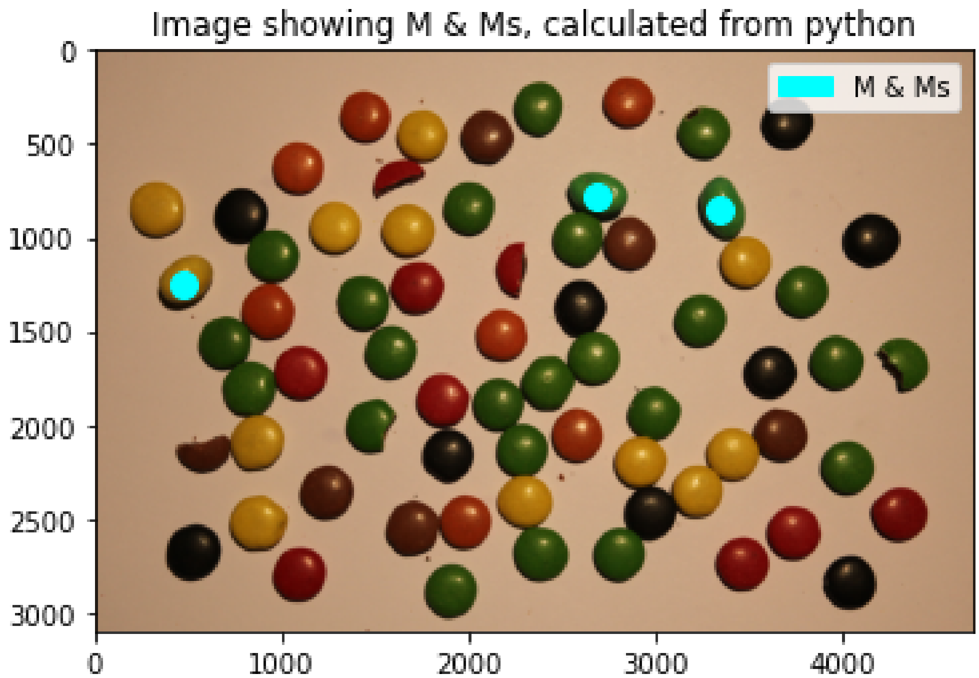
[25]:
```python
results_from_python2 =␣
↪results_from_python1[results_from_python1['circularity']>0.84]
results_from_python2
```

[25]:
```
    labels   area  perimeter  circularity  roundness  solidity  eccentricity  \
9       10  67748   1000.004        0.851      0.783     0.978         0.617
12      13  66180    992.087        0.845      0.712     0.993         0.700
22      27  66359    983.703        0.861      0.743     0.994         0.670

                                   centroid
9    (784.8431392808644, 2679.244553344748)
12    (853.969734058628, 3343.703566032034)
22  (1244.922346629696, 480.60323392456183)
```

0.4.9 Creating Marker using the centroid points obtained from python

[26]:
```python
plt.imshow(croppedimage)
plt.scatter(2679.24,784.84, s=100, c='cyan', marker='o')
plt.scatter(3343.70,853.96, s=100, c='cyan', marker='o')
plt.scatter(480.6,1244.92, s=100, c='cyan', marker='o')

cyan_patch = mpatches.Patch(color = 'cyan', label = 'M & Ms')
plt.legend(handles = [cyan_patch],loc = 'upper right')
plt.title('Image showing M & Ms, calculated from python')
```

[26]:
```
Text(0.5, 1.0, 'Image showing M & Ms, calculated from python')
```



0.4.10 Comparing Results from Fiji and Python

The results was obtained to be the same from both of the methods. Slight difference in the values of the all of the parameter was seen from the two methods like the area, and perimeter but the final result was the same. Therefore, M & Ms are distinguished from Non Stops. The 3 chocolates as showm in the image are classified as M & M's, the other 5 are the half eaten ones and the remaining chocolates are the Non Stops.

The 3 separated M & M's have similar properties from both of the methods as shown below. The roundness and circularity values differ only slightly and therefore, the identified chocolates are the same from both the methods.

3 M & M's from Fiji

[27]:
```
results_from_fiji2
```

[27]:
```
                                       Label   Area  Mean         X         Y  \
9   10  IMG_2754_nonstop_alltogether.JPG  67601   255  2670.182   761.341
10  11  IMG_2754_nonstop_alltogether.JPG  66097   255  3334.224   830.458
22  23  IMG_2754_nonstop_alltogether.JPG  66235   255   471.088  1221.405

          XM        YM   Perim.   Circ.     AR  Round  Solidity
9   2670.182   761.341  995.075  0.858  1.271  0.787     0.986
10  3334.224   830.458  987.845  0.851  1.400  0.715     0.992
22   471.088  1221.405  976.975  0.872  1.346  0.743     0.992
```

3 M & M's from python

[28]:
```
results_from_python2
```

[28]:
```
    labels   area  perimeter  circularity  roundness  solidity  eccentricity  \
9       10  67748   1000.004        0.851      0.783     0.978         0.617
12      13  66180    992.087        0.845      0.712     0.993         0.700
22      27  66359    983.703        0.861      0.743     0.994         0.670

                                   centroid
9    (784.8431392808644, 2679.244553344748)
12    (853.969734058628, 3343.703566032034)
22  (1244.922346629696, 480.60323392456183)
```

0.5 Conclusion:

Different image processing techniques can be used to classify objects. The results from both the methods were compared and was found to be similar.

0.6 References:

Sudhakar, S. (2017, July 10). Histogram Equalization. Towardsdatascience. https://towardsdatascience.com/histogram-equalization-5d1013626e64

Preim, Bernhard, and Charl Botha. "Watershed Segmentation." Watershed Segmentation - an Overview | ScienceDirect Topics, 15 Nov. 2013, https://www.sciencedirect.com/topics/computer-science/watershed-segmentation.

[ ]:
```

```