**AIM**

Calculate the Percentage of Dead part in a Leaf

**Step 1: The Problem is that our image Background is not uniform. So, First we convert the original image to a new image (say IMG1) such that the background would be uniform.**

**Step 2: Now We will Make another image in which the dead part of the Leaf is removed (say IMG2).**

**Step 3**: **We calculate the Percentage of Leaf part in the whole Image IMG1 (say Percent\_1 ) and Percentage of the Leaf part (*note that the dead part are now removed*) in the whole Image IMG2 (say Percent\_2 ).**

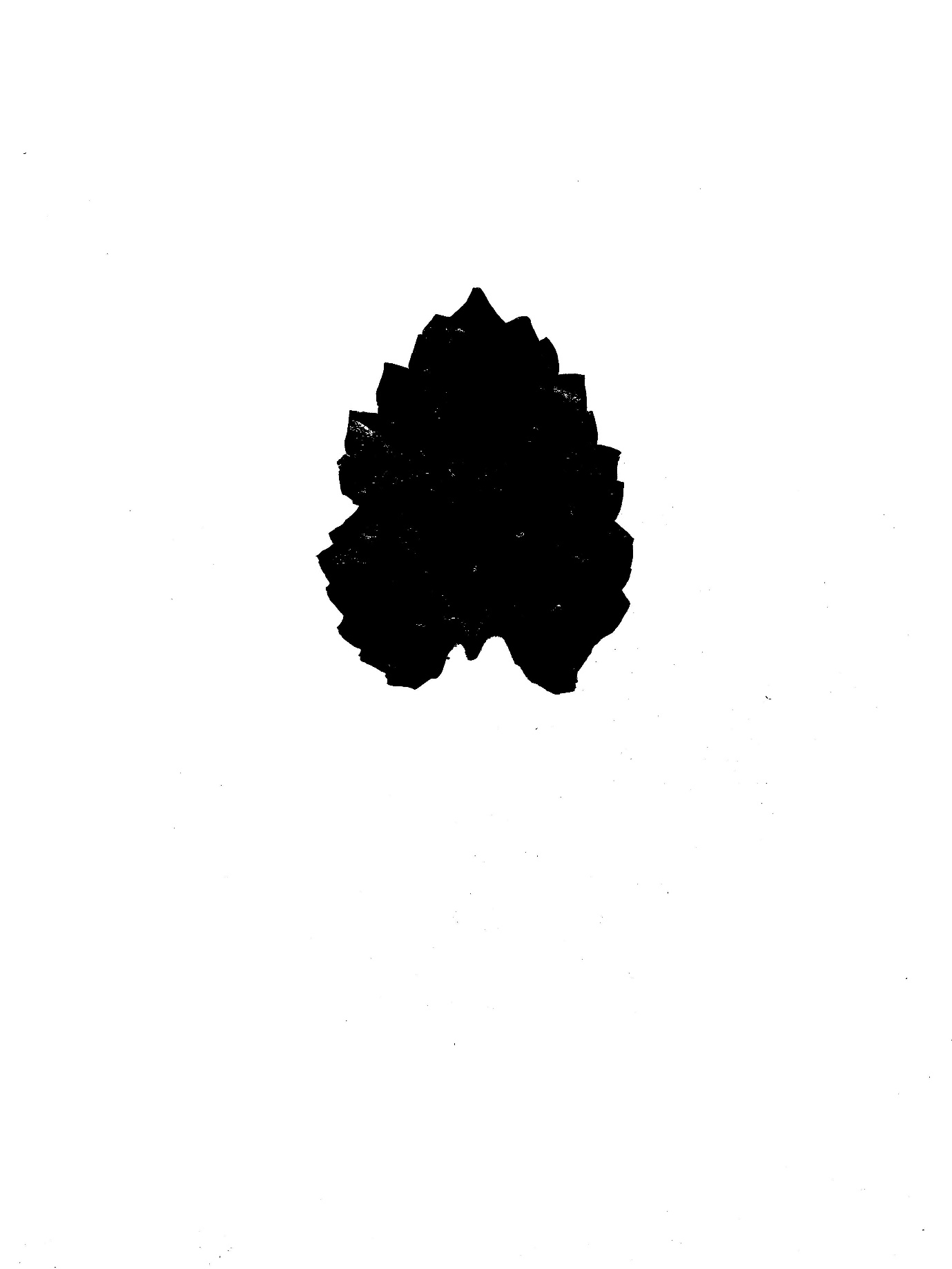
**Step 4: Now Just divide the Percent\_1 with Percent\_2**

* Libraries: -
  + import cv2
  + import numpy as np
  + import matplotlib.pyplot as plt
* Taking input of Image file :-
  + Image\_name = 'Leafspot3.jpg'
* Opencv loads the image in BGR, convert it to RGB.
  + img = cv2.cvtColor(cv2.imread(Image\_name),

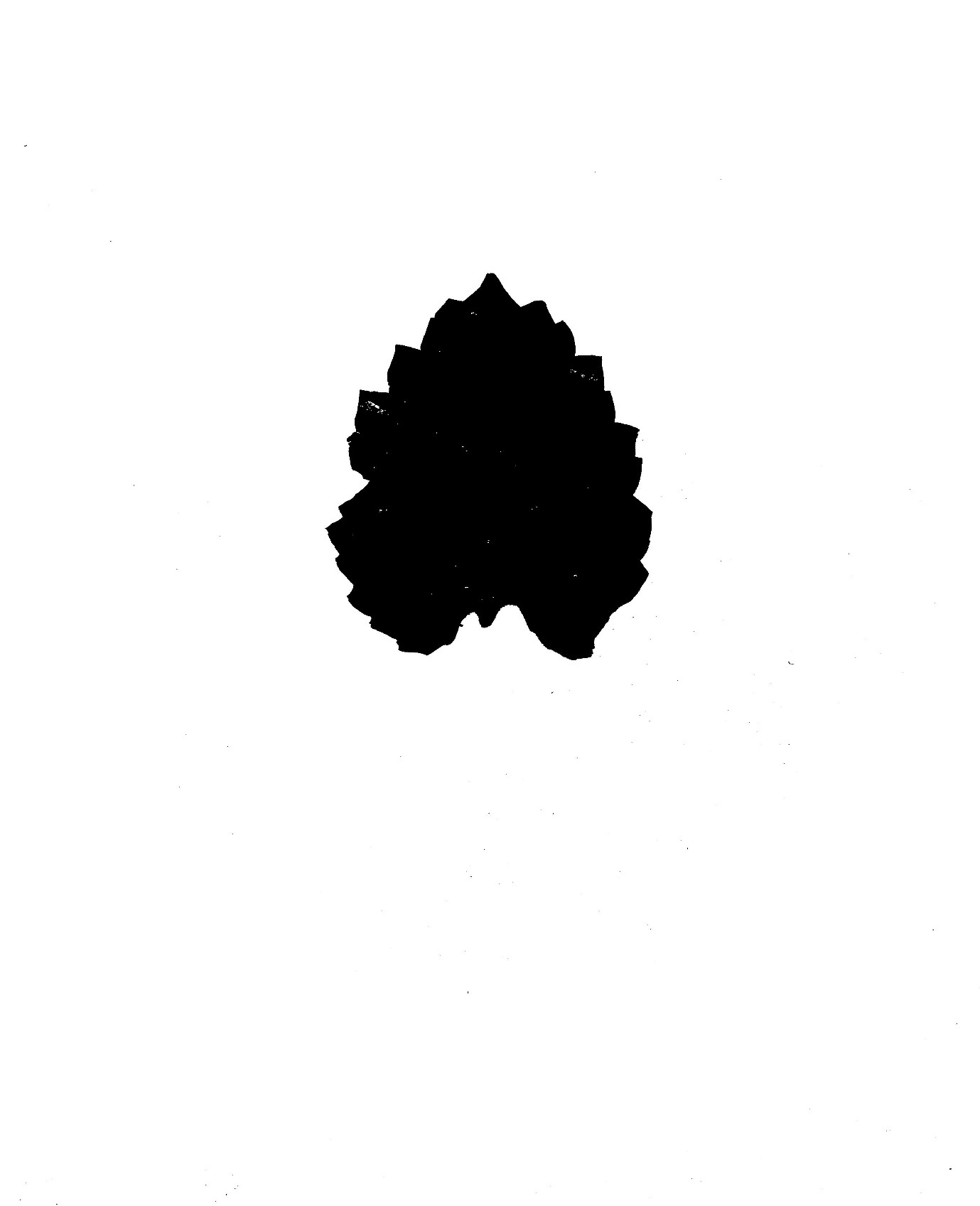
                  cv2.COLOR\_BGR2RGB)



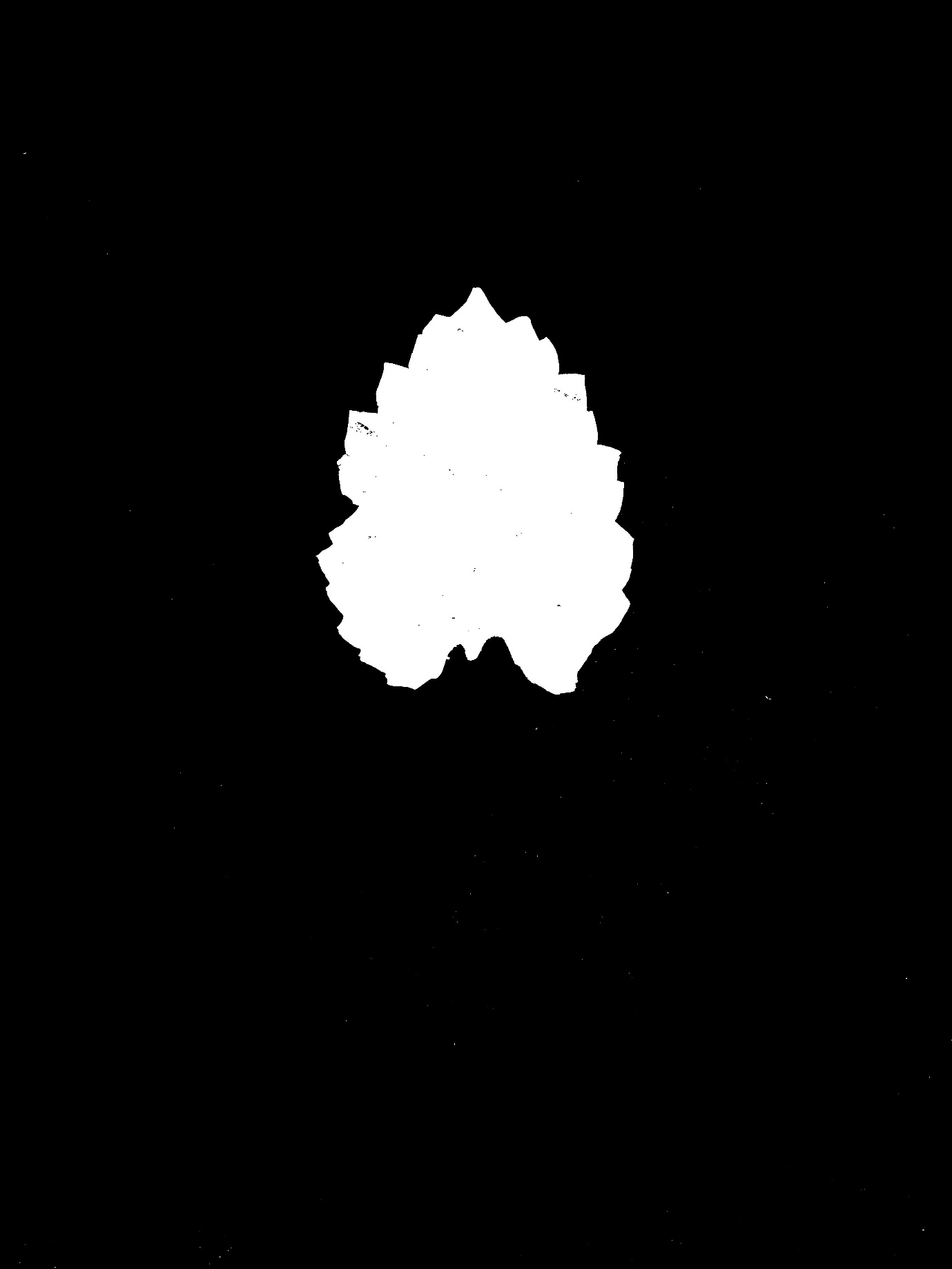
* Range of green color in numpy array
  + lower\_green = np.array([0, 120, 0], dtype=np.uint8)
  + upper\_green = np.array([255, 255, 255], dtype=np.uint8)
* Thresolding the Image to get only Green color from it.
  + mask = cv2.inRange(img, lower\_green, upper\_green)



* So, after the above step we get a black and white image stored in a variable ‘mask’. In which the green color is replaced by Black color and the rest becomes white.
* Now the problem is to remove the noise from an image which is in form of some random white spots in our ‘mask’ image.
* The below code will help us to remove the noise.
  + mask = cv2.morphologyEx(mask, cv2.MORPH\_OPEN, cv2.getStructuringElement(cv2.MORPH\_ELLIPSE, (3, 3)))



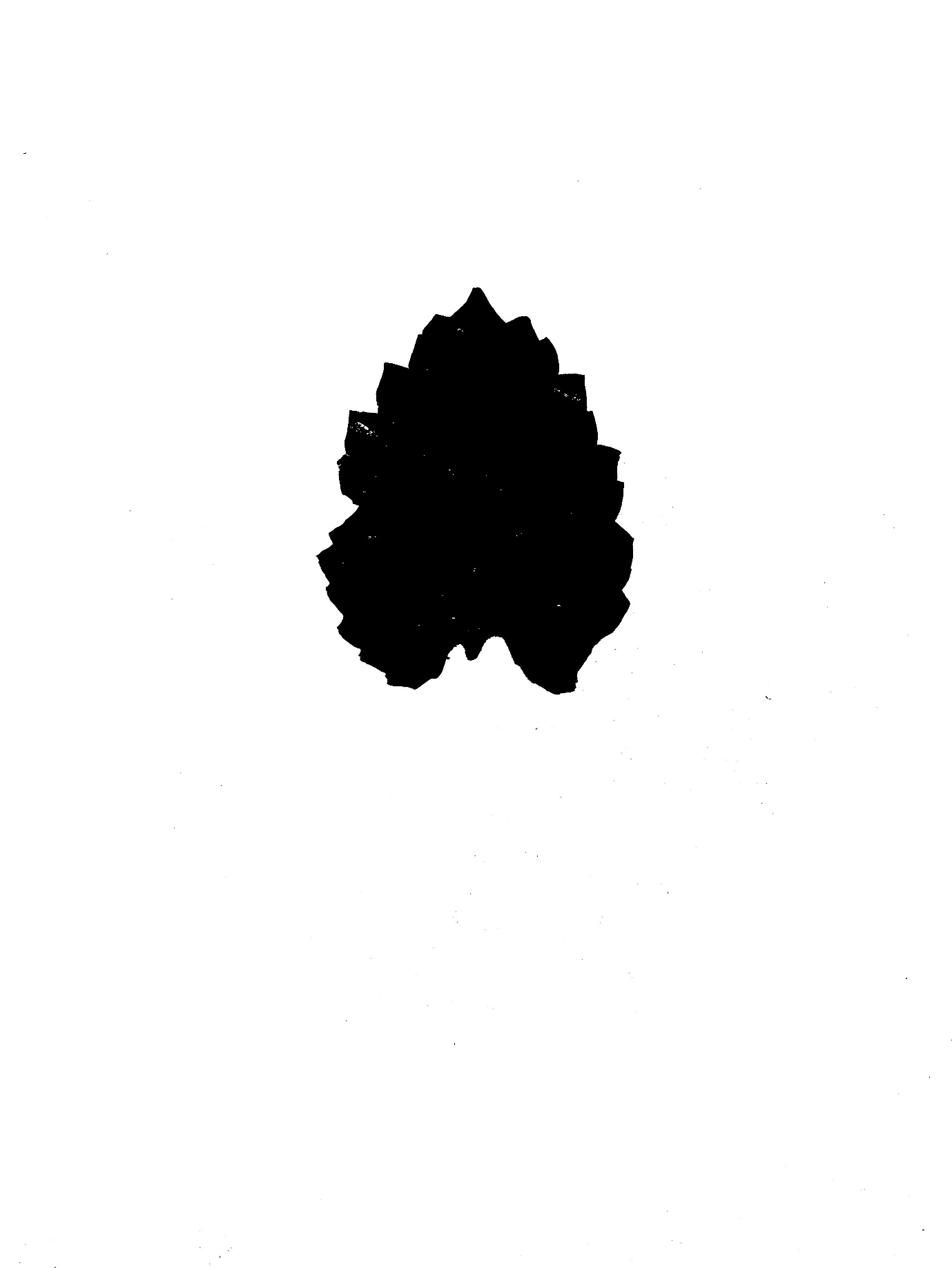
* **Inverting the mask** => Now We will convert our background to Black color and the image to white color, for future applications
  + mask = cv2.bitwise\_not(mask)



* **Foreground Mask** => Now we merge the original image ‘img’ to the black and white image ‘mask’ ( note that white is our leaf image and black is our background ) by below code :-
  + fg\_masked = cv2.bitwise\_and(img, img, mask=mask)



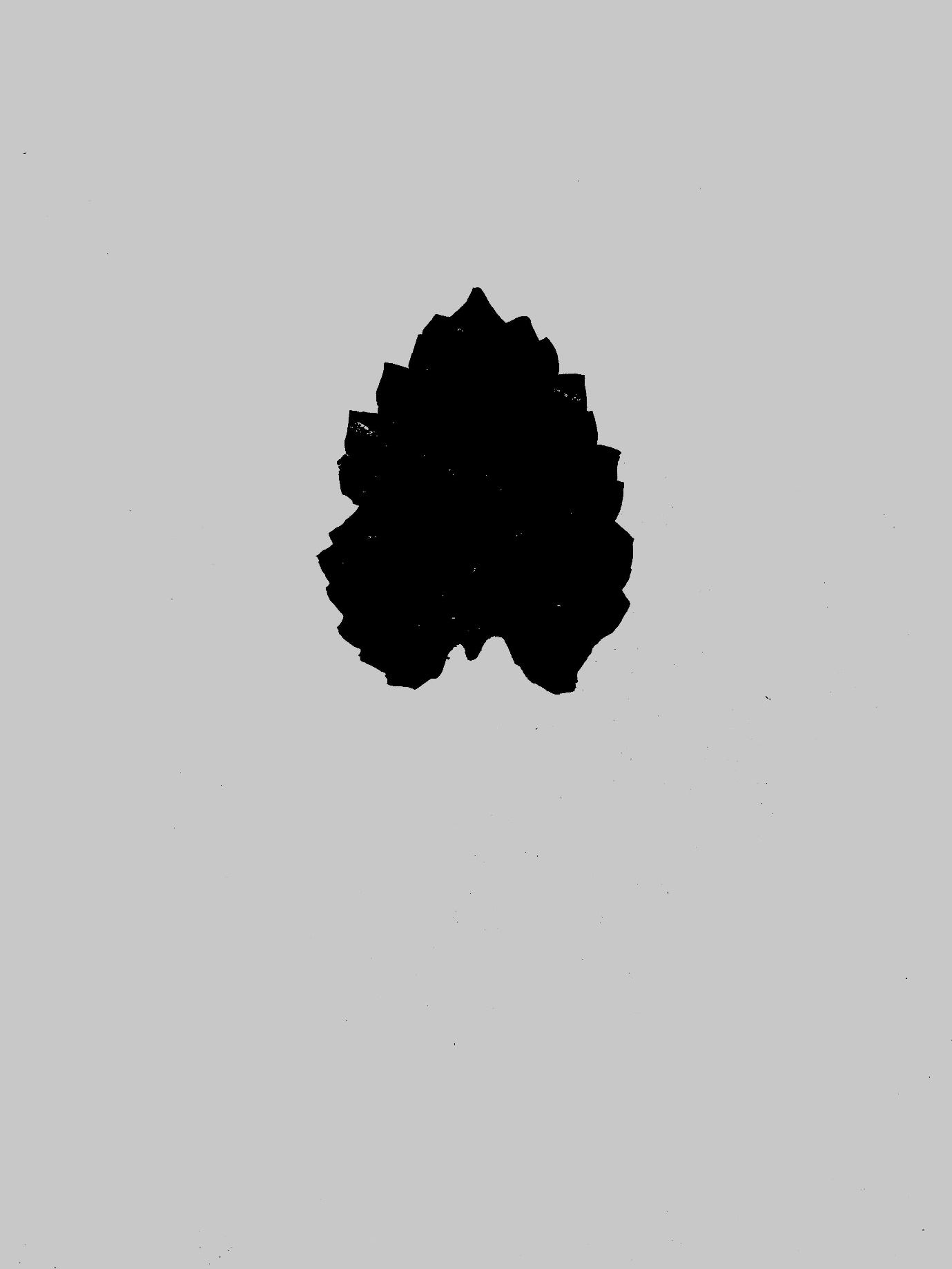
* Getting the Original Mask by inverting the Inverted Mask to get back again it, by below code
  + mask = cv2.bitwise\_not(mask)



* Now we will generate any background image of any color But of same shape of our original image (means same size i.e. length and width) by below code :-
  + bk = np.full(img.shape, 200, dtype=np.uint8)



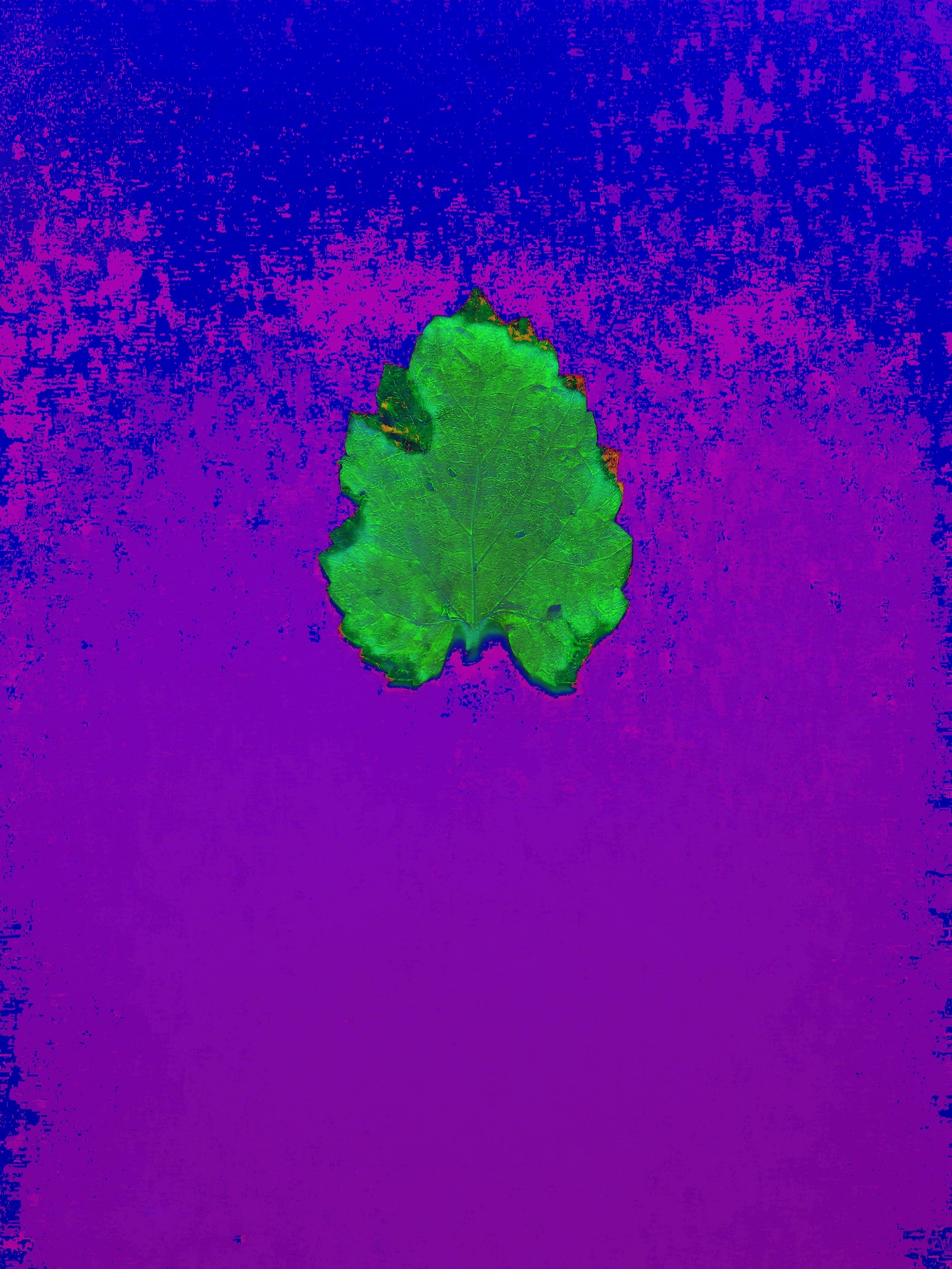
* **Background Masked** => Now we merge the Above background to the Original Mask by below code
  + bk\_masked = cv2.bitwise\_and(bk, bk, mask=mask)



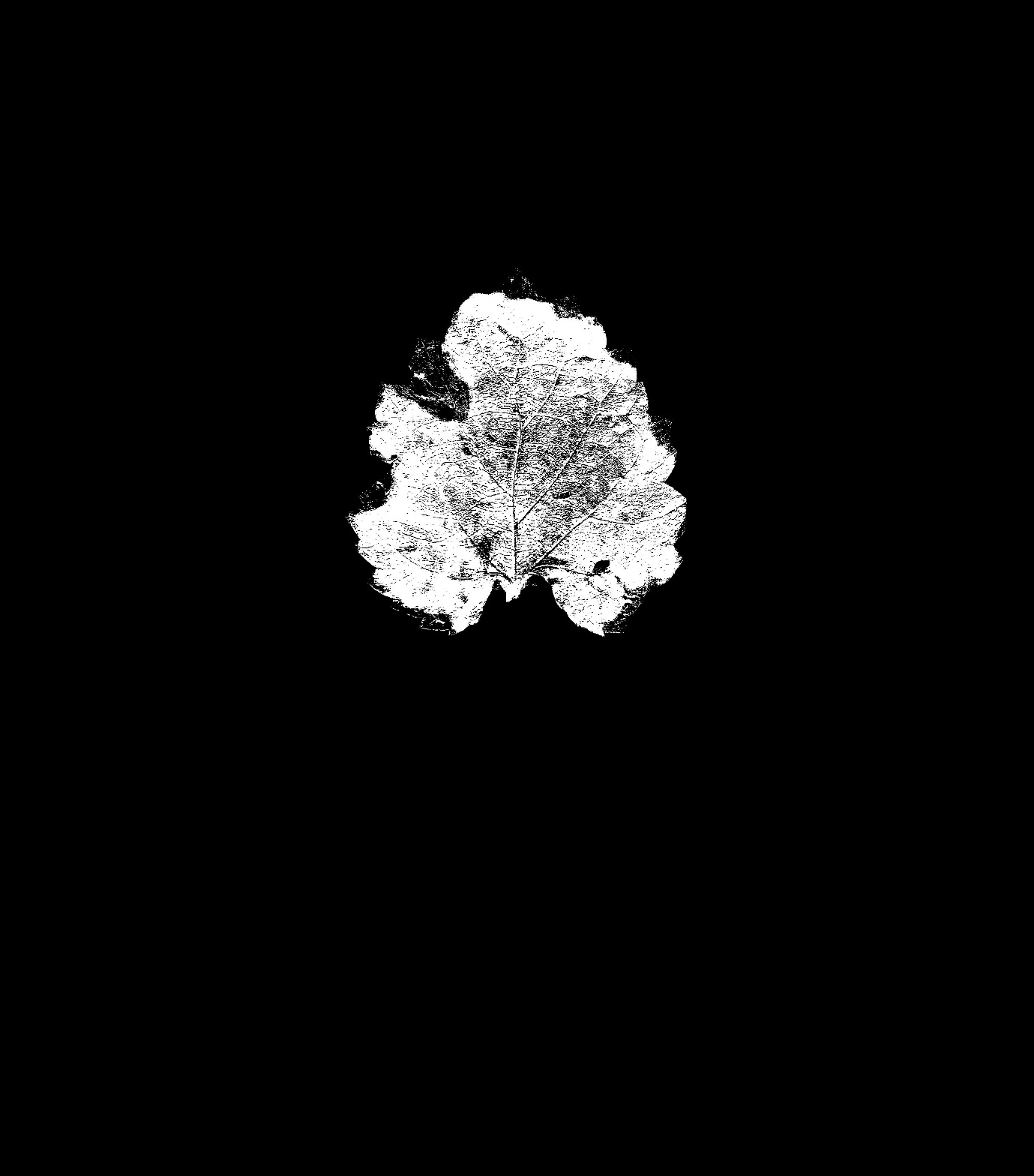
* Forming the New Image which is IDENTICAL to the First Original Image => Now we add the FOREGROUND IMAGE and BACKGROUND IMAGE by below code
  + final = cv2.bitwise\_or(fg\_masked, bk\_masked)



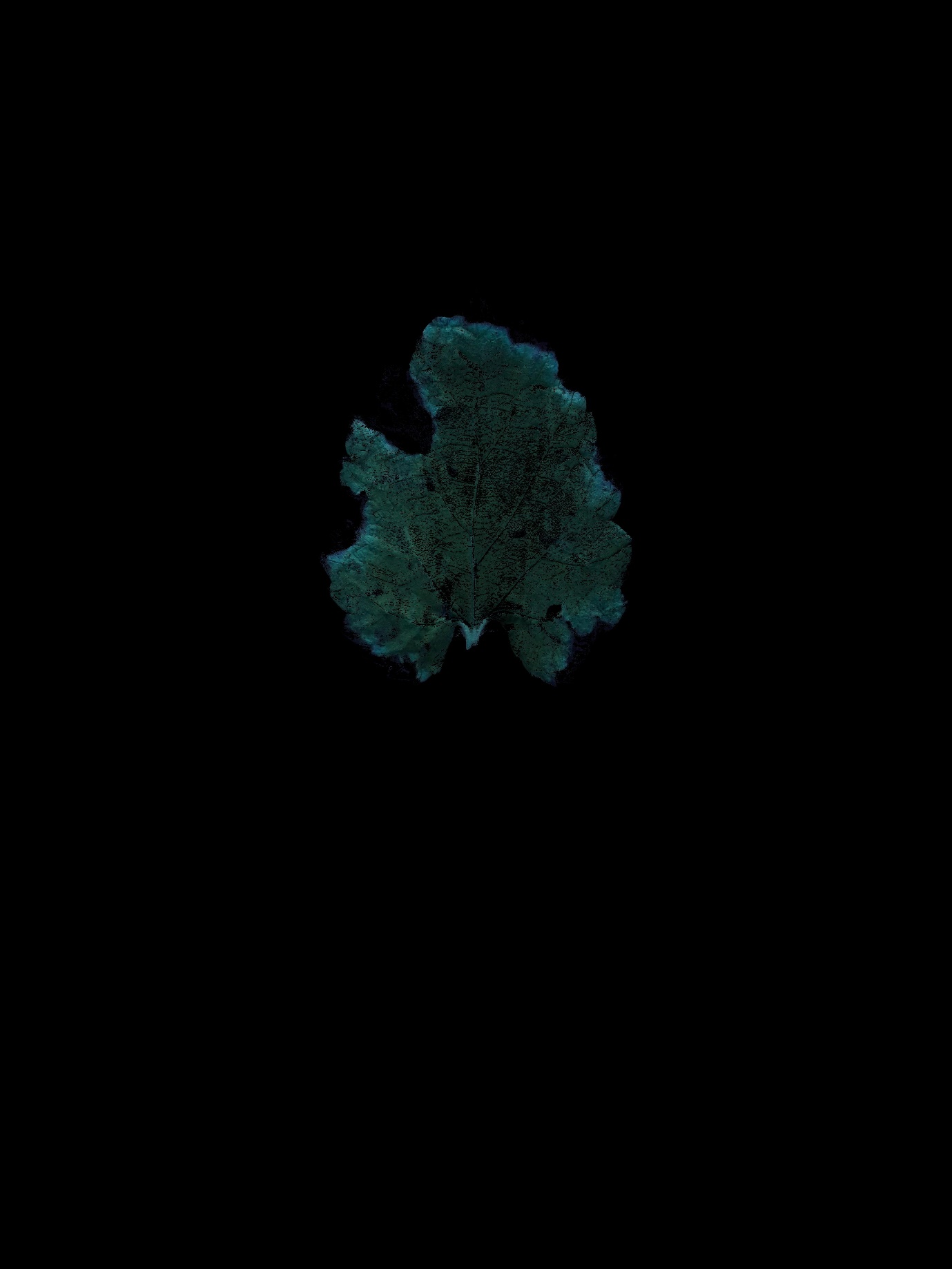
* Convert the Original BGR Image into HSV Image by below code
  + hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)



* Range of a green color which is stored in numpy arrays by below code
  + upper\_green = np.array([75,255,255])
  + lower\_green = np.array([20,100,50])
* Mask of the original image in which the green part is replaced by white color and rest with black color which helps us to remove the dead part from the leaf image by help of below code
  + mask = cv2.inRange(hsv, lower\_green, upper\_green)



* Merging the original image into the above mask by below code
  + res = cv2.bitwise\_and(frame,frame, mask= mask)



* Making kernel to smoothened the above image or Basically this is our filter which helps us to get the better image
  + kernel = np.ones((15,15),np.float32)/225
  + smoothed = cv2.filter2D(res,-1,kernel)



* Opening the image and storing in ‘image’ variable
  + image = Image.open("removed\_dead\_im.jpg")
* Getting the leftmost corner pixel of the background of an image which is unique and identical to every pixel of the background image ( so whatever pixels are there in an image other than leaf pixel )
  + bg = image.getpixel((0,0))

=> (0,0,0) or may be different (depend on the background color of an image)

* Storing the width and height of an image in the respective variables
  + width, height = image.size => width = 3096, height = 4128
* So, total pixels in an image are = 3096x4128 = 12,780,288
* Now, Calculating the number of pixels which represent the black color or may be a unique color (depend on the background color of an image) of our background image
  + bg\_count = next(n for n,c in image.getcolors(width\*height) if c==bg)
* 11709229 are the number of background color pixels
* Now, Calculating what is the number of pixels that my leaf image contain in the whole image
  + img\_count = width\*height - bg\_count
* 12,780,288 – 11,709,229 = 1,071,059 are the number of leaf pixels in a whole image.
* Percentage of the leaf part present in the whole image =
  + Percentage = ( ( img\_count ) / ( widthxheight ) ) x 100
* So, Using above formula we calculate the percentage of leaf part in IMG1(Original image) say Percent\_1
* Last step to divide first image percent with second

Percentage\_dead\_in\_leaf = Percent\_2 / Percent\_1

* Similarly for IMG2(removed dead part of leaf image) say Percent\_2