Question 3:Scale space blob detection

Steps:

1. Generate a laplacian of guassian filter
   ->The scale normalized laplacian of guassian is given by

$$(x^2 + y^2 - 2\sigma^2)\, e^{-(x^2+y^2)/2\sigma^2}$$

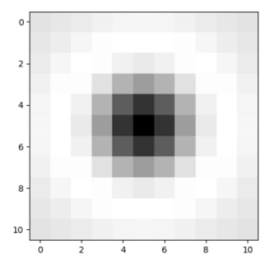
where sigma is the scale , x and y are the coordinates

->Here the coordinates x,y should be taken as the center coordinates should be 0,0 and the remaining accordingly.

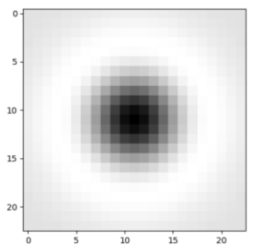
->Here is the code snippet for laplacian of guassian filter

```
def LaplacianofGuassian(sigma,log):
    for i in range(0,size):
        for j in range(0,size):
            y=int(size/2)-i
            x=j-int(size/2)

            log[i][j]=((x*x)+(y*y)-(2*sigma*sigma))*(math.exp(-
((x*x)+(y*y))/(2*sigma*sigma)))
```

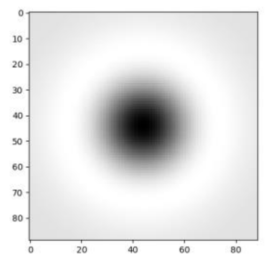
->The plots of laplacian of guassian filter with sigma 2 and 15



Sigma=2



Sigma=4



Sigma=15

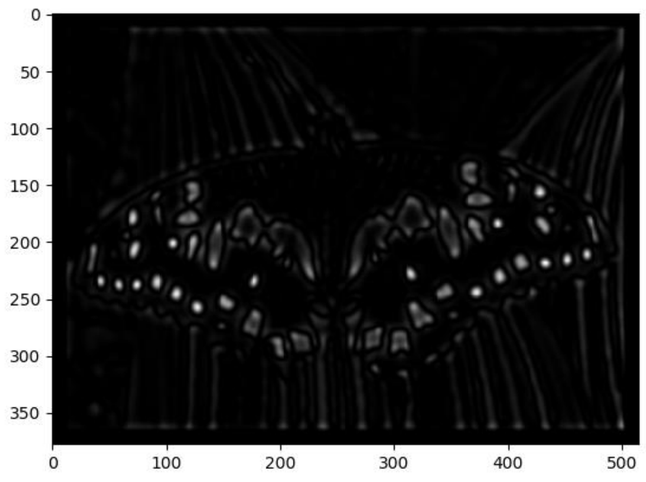
2.Build a laplacian scale space

->First apply the filter to the image with a scale.

-First perform zero padding to the image and perform convolution to the image using the generated laplacian filter.

```
#zero padding
 for i in range(0,m1):
        for j in range(0,n1):
            if(i<s1 or i>=m+s1 or j<s1 or j>=n+s1):
                im_z[i][j]=0

            else:
                im_z[i][j]=gray[i-s1][j-s1]
#convolution of each pixel
def convolution(image,filter1,size):
    sum=0
    for i in range(0,size):
        for j in range(0,size):
            sum=sum+image[i][j]*filter1[i][j]

    return sum*sum
```

->Then take the square of laplacian of response for the present scale space i.e., sigma=4 here.



Sigma=4

->Increase the scale space by a factor k(1.5 here) and again repeat the operations

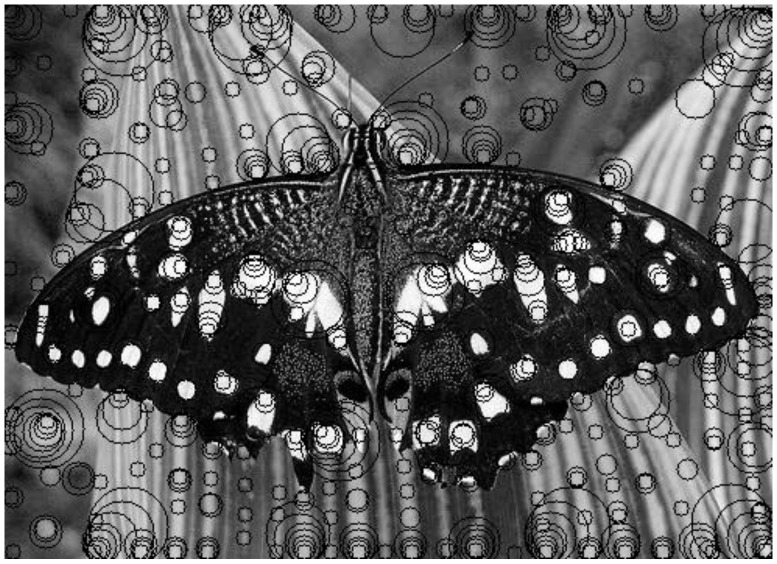
3. Now perform non maxima suppression

```
    for i in range(s1,m1-s1):
        for j in range(s1,n1-s1):
            sample=[]
            for p in range(i-s1,i+s1+1):
                for q in range(j-s1,j+s1+1):
                    sample.append(images_sigma[k][p][q])
            a=max(sample)

            if(k==0):
                if(images_sigma[k][i][j]==a):
                    coordinates_list[k].append([i-s1,j-s1])

            else:
                print(k)
                if(images_sigma[k][i][j]==a):
                    for b in range(0,k):
                        if(coordinates_list[b].count([i-s1,j-s1])>0):
                            coordinates_list[b].remove([i-s1,j-s1])


                    coordinates_list[k].append([i-s1,j-s1])
```
->Check whether the maximum of pixels of filter size with current pixel as origin and current pixel values are same. If same consider that coordinates. Perform that operation for the same coordinates with all the scales and consider the coordinate with highest sigma.

4.Display resulting circles

After you collect all the coordinates, draw blobs with that coordinates as center with radius sigma*square root(2).

OUTPUT:



Above is the output for 8 iterations of sigma starting with 4 and multiplied with a factor 1.25.