Question 1
For this assignment below picture was chosen.

Part A : (Degrading the image and adding noise)

- **Motion blurred filter:**

For motion blur first, the color channel is split, second apply the transferred channels to the Fourier domain. Third, by multiplying each channel by *H*, a horizontal and vertical periodic noise is added to each channel which constant α adjusts the horizontal and constant β adjusts the vertical noise.

$$H(u,v) = \text{sinc}(\ \alpha \cdot u + \beta \cdot v\ ) \cdot \exp(-j\pi\ (\ \alpha \cdot u + \beta \cdot v\ ))$$

Below the result of motion-blurred filter e=with alpha nd beta value, 15 are presented.

- **Gaussian noise**

Gaussian noise is signal noise with a normal distribution. In this assignment by using inbuild function `random_noise(blurry, 'gaussian', mean=0, var=0.03)` an array of random number with normal distribution, mean = 0 and variance = 0.03 has been applied to the picture with blur image. which the result is shown below

Part B : Removing Noise

- **direct inverse filtering Blur**

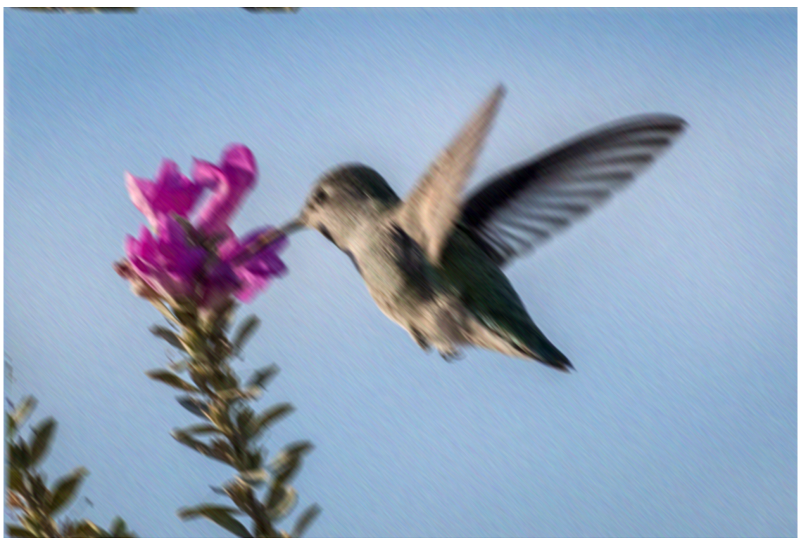
By dividing each channel (in the frequency domain) of blurred image by *H* we can remove the blur added to the image. Below the result is presented.

- **direct inverse filtering to the blur and additive noise image**

In this assignment the same procedure in the previous task was applied, the only difference is that there is also additive noise in the image. So After calculating the *F(u,v)* we can put this in below formula get the noise back. But the problem is the noise are not completely gon and there is still some light noise in the image.

$$F^{*} = F(u, v) + \frac{N(u,v)}{H(u,v)}$$

In below the result of this experiment is presented.



- **MMSE filter HW for additive noise Image**

By multiplying the H_W to the transfer domain of each chanel , denoise image is obtained.

In below H_W formula and it's property are presented.

$$H_w(u, v) = \frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + S_n(u,v)/S_f(u,v)}$$

$H_{(u,v)} = degradation\ function$

$H^{*}_{(u,v)} = complex\ conjugate$

$|H(u, v)|^2 = H(u, v)\, H^{*}(u, v)$

$S_n(u, v) \ = \ |N(u, v)|^2 = \ Noise\ power\ spectrum$

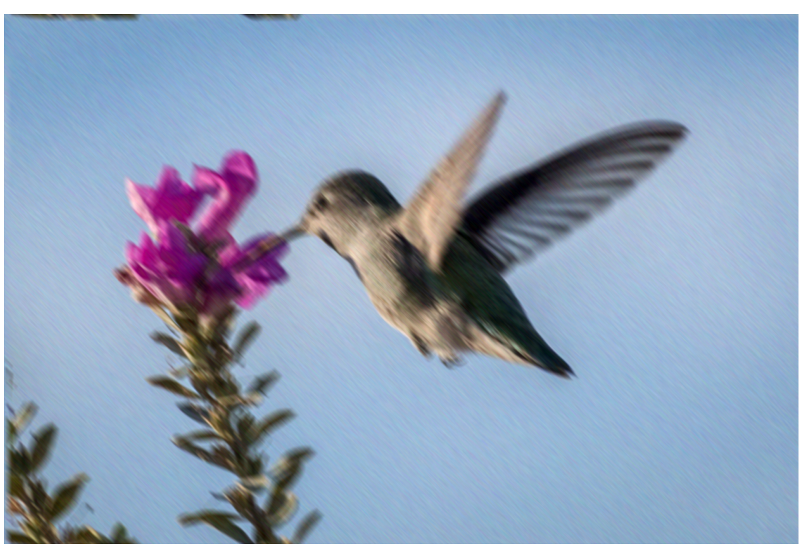
$S_f(u, v) \ = \ |F(u, v)|^2 = \ Original\ power\ spectrum$

Since in this task the MMSE filter have to applied to the image with **additive noise**, degradation function considers 1 since the the applied noise should be removed. In below the result for this task is presented.

- **MMSE filter HW for additive noise and motion blure Image**

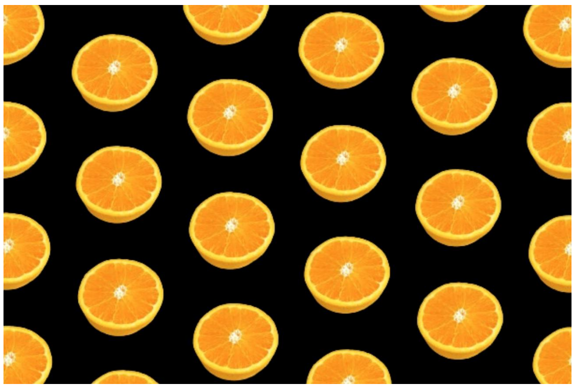
In this task since MMSE filter suppose to applied on image with motion blur and additive noise *H(u,v)* cannot consider as 1 anymore so H(u,v) can obtain by dividing the *G(u,v)* in *F(u,v).* The power spectrum ratio of applied noise to the original image also can obtain by taking the mean of $S_n(u, v)/S_f(u, v)$

The following image present the result after removing noise with MMSE filter from the image that has additive noise and motion blur.

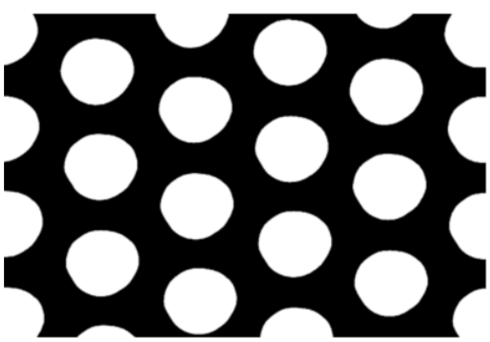


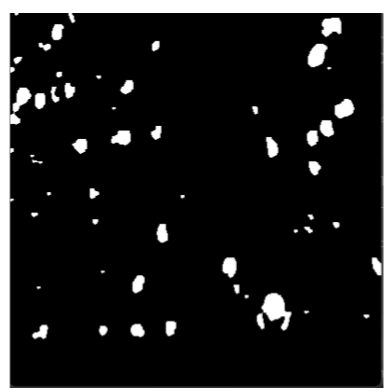
Question 3 - Count Orange

Task 3.1:
For this task below images were chosen.



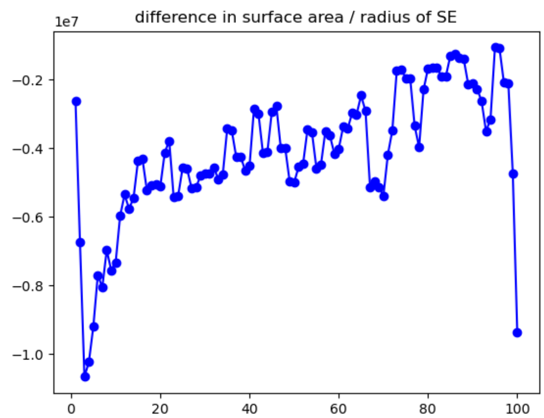
The first step is to convert the image into black and white(round objects are white and the rest is black) by the threshold. So it’s a binary function with getting threshold it adjusts the colors 0 or 255 if they are higher or lower than the threshold. With this black and white image, the white regions are labeled by finding the 255 value. The area and shape of the oranges can be found in these labels by checking the different sizes of the circles based on the position of each label. Below you can see the black and white version of each image. From below it is observed that in the orange image, the shape of these labels is more obvious it can be concluded that the number of the orange is also more accurate in comparison to the other image since in the other one the circles are not clear.

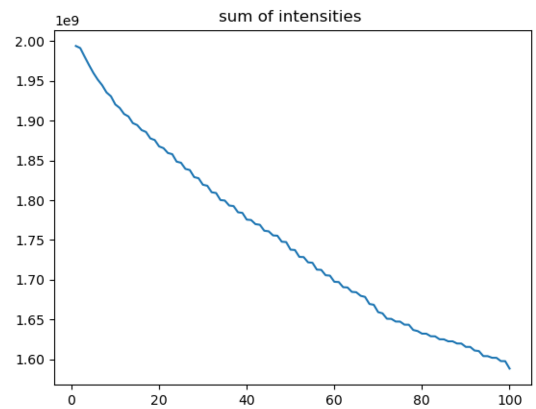




```
the number of the oranges in orange image is:  23
the number of the oranges in orange tree image is:  16
loding 0 : 100
```

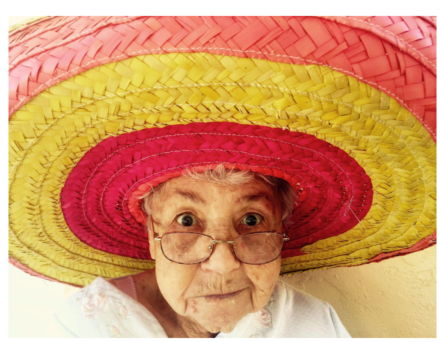
Task 3.2:
For this assignment, some steps of the previous task were used, So the image was converted to the gray intensity and after that by using threshold, the black and white image was obtained then by checking the different sizes of the ellipse which this action happened by the help of opening function which gives us the information about the intensity and size of elements, so the elements with lower radius have high intensity and vice versa. By using this information and getting derivative of this information the change ratio of these lights to remove duplicated lights. The Below graph represents the frequency for different sizes of lights.





Question 4 - PCA Recognition

In this exercise, PCA recognition has been done by using `cv2.PCACompute` inbuild function. This function itself is a linear dimensional reduction technique that extracts information from higher dimensional space by projecting it into a lower-dimensional subspace.
For this assignment, 4 different variations were chosen for three different data.
So this function returns mean and eigenvector.
Below these three images were presented.



Task 4.1 Finding eigenfaces

For each image, a matrix consisting of rows of flattened variation Was initialized. the purpose of PCAComute function is to calculate the covariance function, for this goal first mean of the image for each axis was calculated after that by creating another matrix M which is subtracting the mean from every data point, the covariance of the matrix was calculated by the follow formola.

$$C = MM^T$$

by passing this matrix to `cv2.PCACompute` we can get mean and eigenvector. So mean is matrix of subtraction of the mean of image from each data point. Below figures shows the combination of each variations with preserving the essential parts that have more variation of the data and removing non-essentioal parts with fewer variation regarding to thair covariance.

Task 4.2 Finding eigenfaces

For finding the eigenface, weights of images were calculated which is M matrix times its eigenvector, and the results were shown below.


Task 4.3 Reconstruct each face using 'wrong' PCA weights

In figure 4.3.1, weights were initialized with the eigenvector of image 1 and the mean of image 2 and in figure 4.3.2, wights are based on the eigenvector of image 2 and the mean of the first picture.
As you can see in the images, manipulation of the mean for each image does not make any difference, and the eigenvector has more influence in comparison to the mean.

Figure 4.3.2

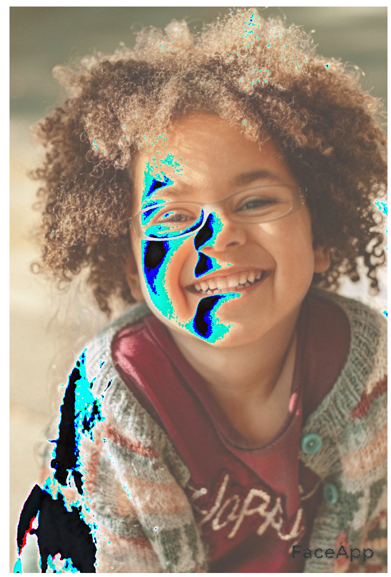


Figure 4.3.2