Computer Vision

Project 1

Submitted to:

Dr. Ahmed Badawy

Eng. Ayman Anwar

Eng. Laila Abbas

Submitted by:

Esraa Sayed

Irini Adel

Amany Bahaa Eldin

Alzahraa Eid

In this report, we represent screen shots for our project results.

1-Add noise to an image:

1-Salt and pepper noise

This is a randomly appearing white point (salt) or black point(pepper), which assumes an equal probability of pepper (0) or salt (255).

2-Gaussian noise

In this type of noise, the probability density function obeys a Gaussian(normal) distribution.

3-Uniform noise

Uniform noise means the different values of noise have equal probability, which means that the noise contains random values from a uniform distribution.

2- Filter the image(denoising)

1-Gaussian filter

This is a 2D convolution operator that’s used to blur an image and remove the noise. The used kernel follows the Gaussian shape.

2-Average filter

Uses a linear method to average the pixel values in the entire window range, but it doesn’t remove the noise points well and doesn’t protect the image details so make the image blurred.

3-Median filter (non-linear filter)

Replace the pixel value by the median value of the neighboring pixel.

3- Edge Detection:

- It mainly depends on the *convolution* of a kernel with an image.
- Edge detection is enhanced by *blurring the image first* (This is achieved through *Gaussian filter*).
- 4 filters were implemented:

a) Sobel filter:

$$Sx = \begin{matrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{matrix}, \quad Sy = \begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix}$$

b) Prewitt filter:

$$Px = \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}, \quad Py = \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix}$$

c) Roberts filter:

$$Rx = \begin{matrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{matrix}, \quad Ry = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{matrix}$$

Each of the matrices gets convolved with the image to get horizontal and vertical edges. The final output is the ***gradient***.

$$\sqrt{Sx^2 + Sy^2}$$

d) Canny:

1- Gaussian filter is applied

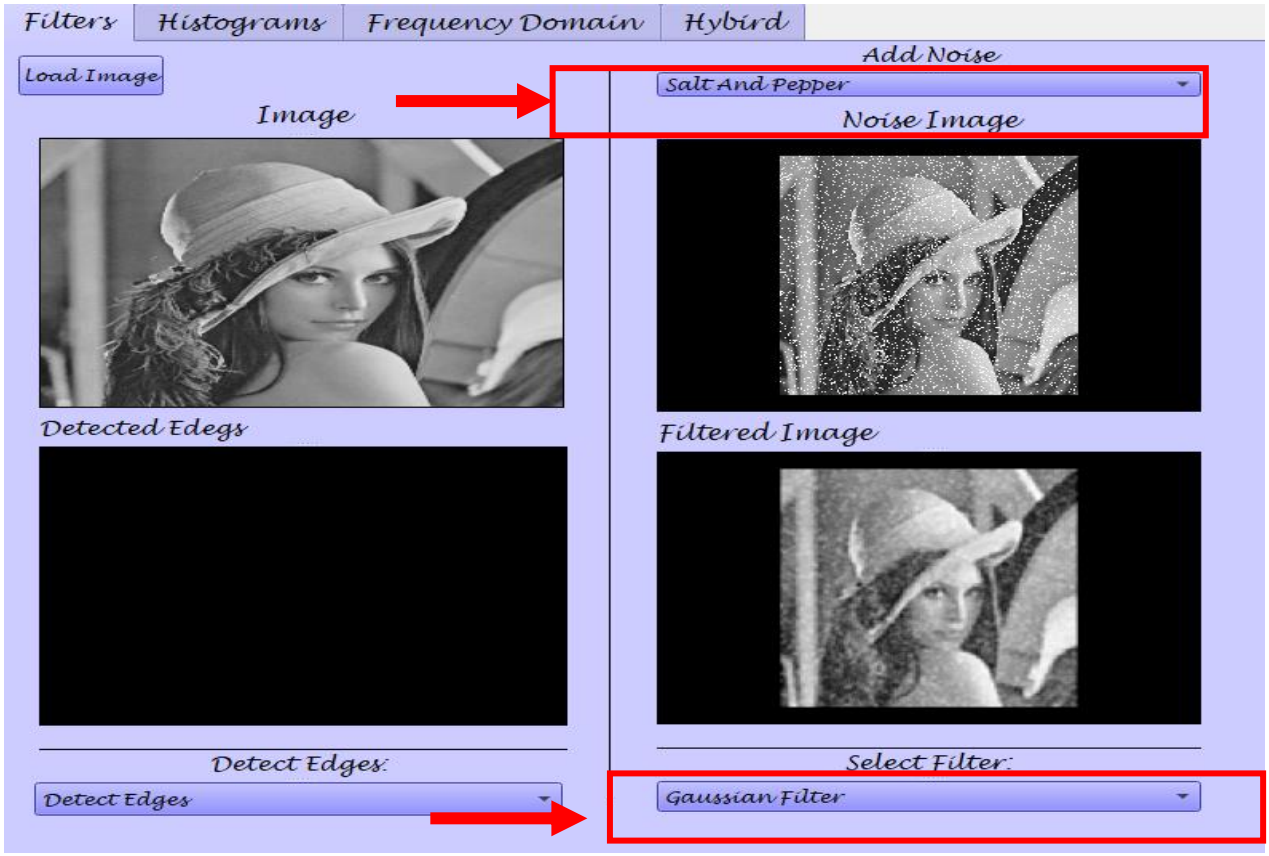
2- Edges are detected by Sobel

3- Non maximum suppression for edge thinning

4- Hysteresis Thresholding

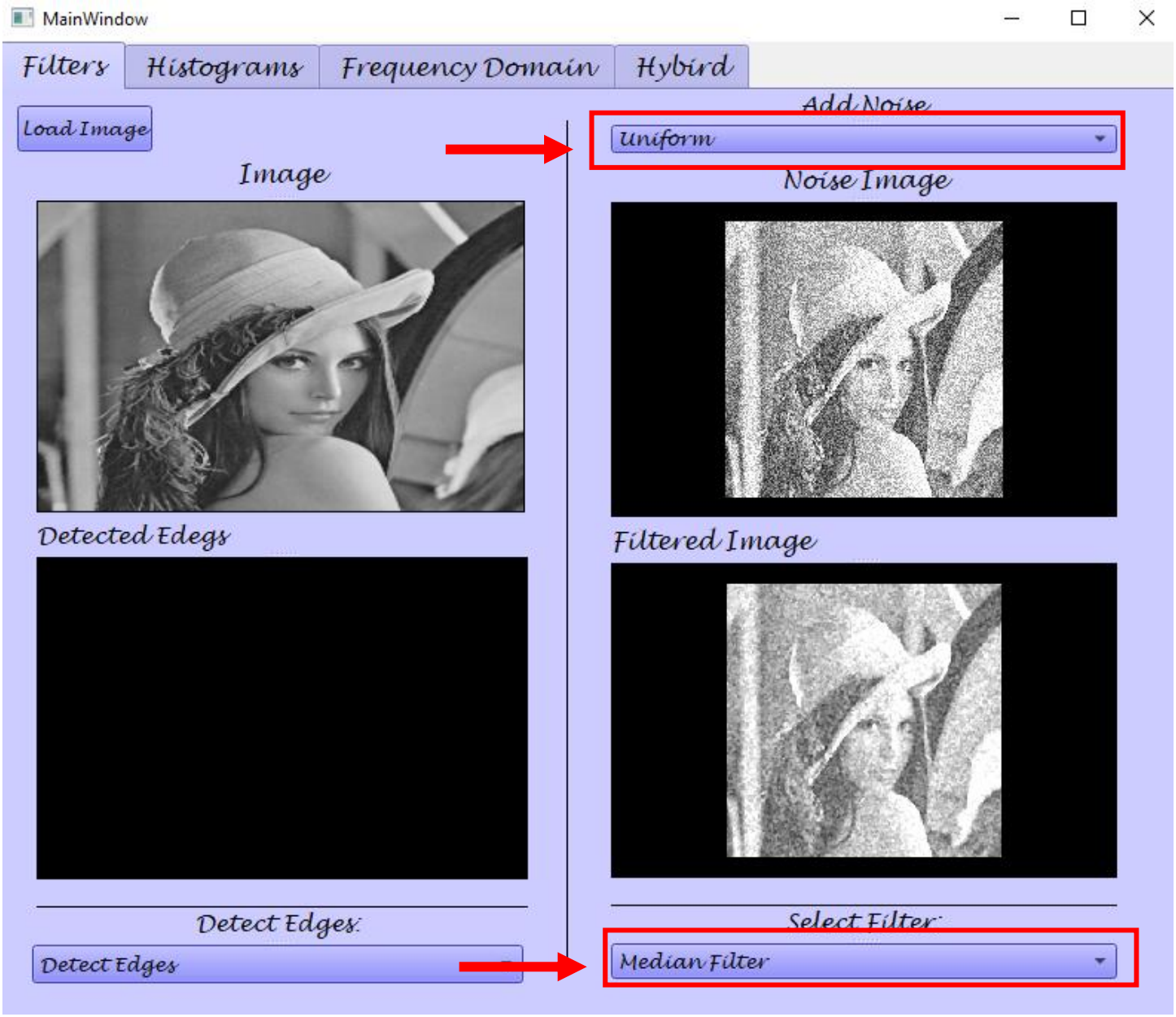
Results for adding noise, filtering noise and edge detection:

1)adding *salt and pepper* noise and filtering with *Gaussian* filter

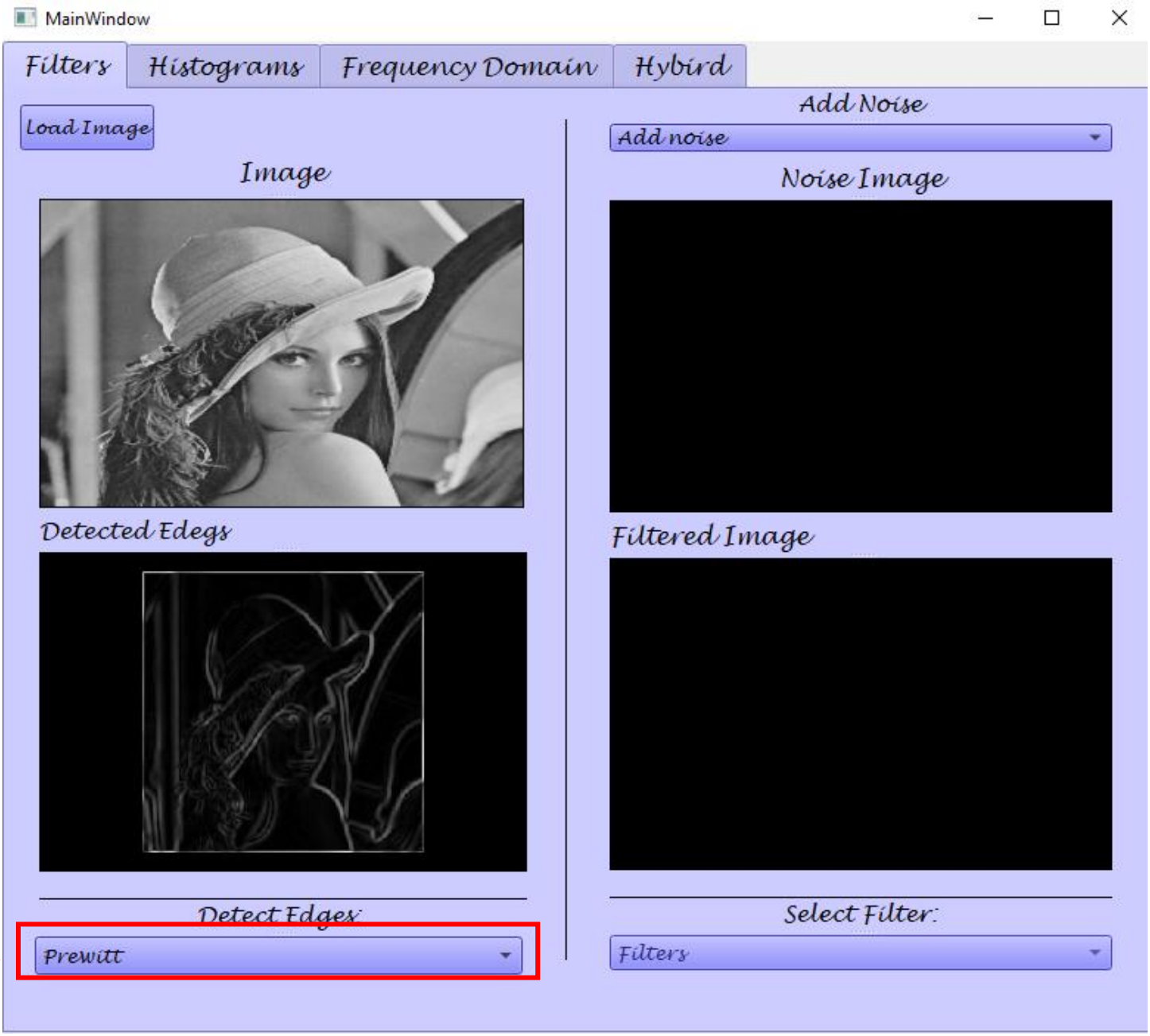


2) adding *Gaussian* noise and using *average filter*

3) adding *Uniform* noise and using Median *filter*



4) results for edge detection



Histogram

If the input image is:

Gray scale image

Plotting the histogram of the gray scale image

Options:

- **Equalization**

  1 -> 1 transformation maps each pixel value to another value through few steps implemented in the code

- **Normalization**

  Mapping the range of the image intensities to the range of 0 -> 1 by:
  pixel's intensity =
  pixel's intensity / max pixel intensity

- **Global Threshold**

  Specifying a threshold and compare each pixel's intensity with this threshold:
  if the intensity < threshold, the pixel is *Black,*
  if the intensity > threshold, the pixel is *white*
  resulting a binary image.

- **Local Threshold**

  Divides the image to many blocks and apply on each block thresholding with different value, by increasing number of blocks:
  blurring increases

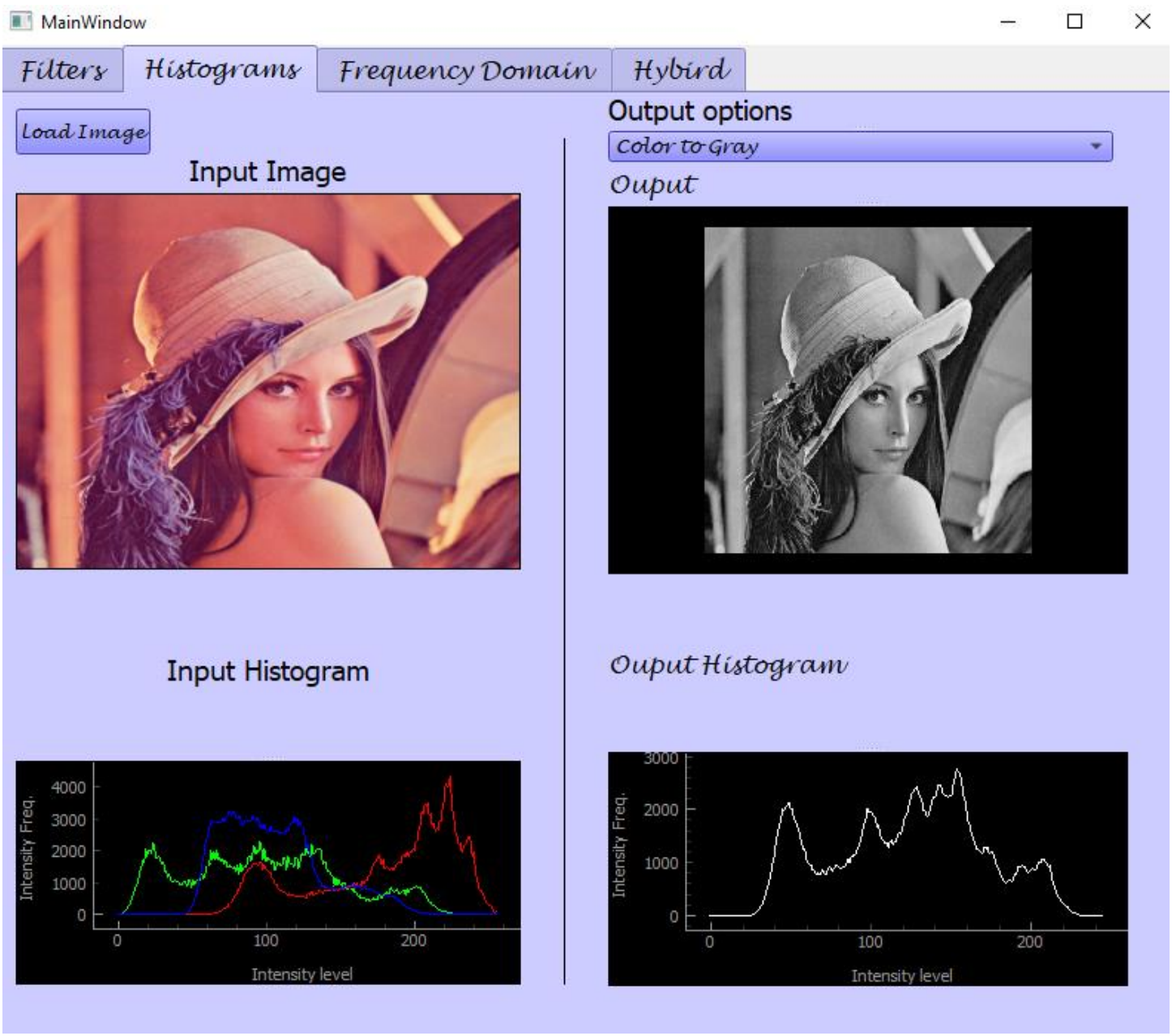
Color image

Plotting the histogram of each channel (color image histogram)

Options:

- Color to gray
  Converts the color image to gray scale image and plots the histogram of the resulted image.

4) results for converting to gray scale color



MainWindow

Filters | Histograms | Frequency Domain | Hybird

Load Image

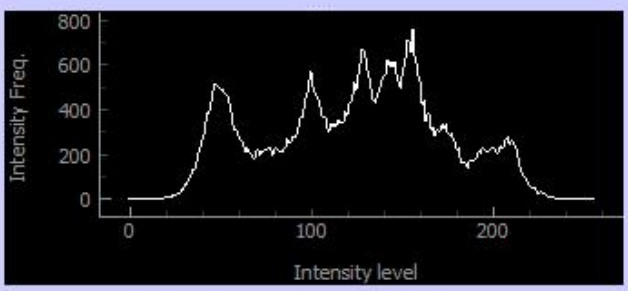
Input Image

Output options

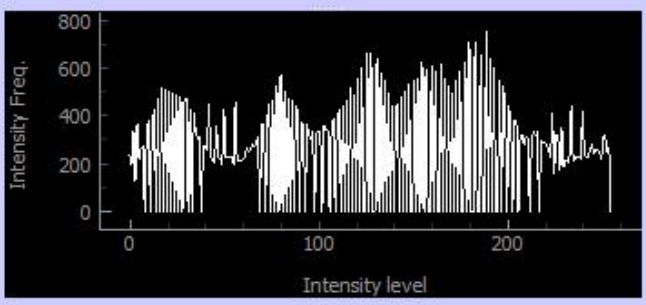
Equalization

Ouput

Input Histogram



Ouput Histogram



MainWindow

Filters | Histograms | Frequency Domain | Hybird

Load Image

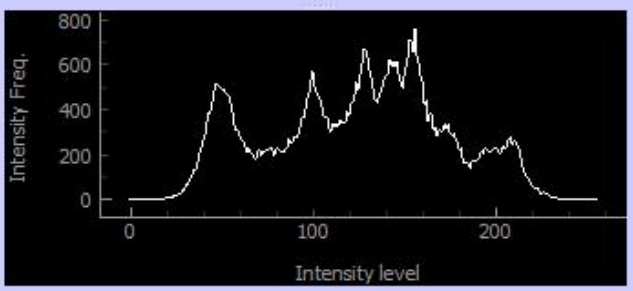
Input Image

Output options

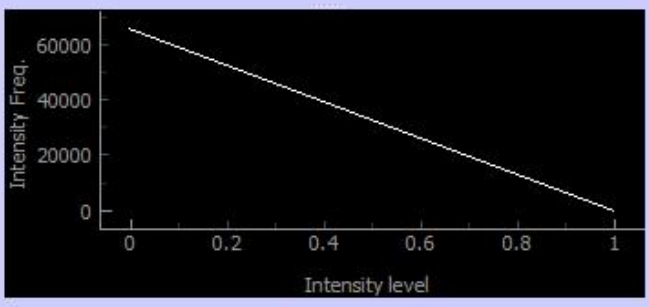
Normalization

Ouput

Input Histogram



Ouput Histogram



MainWindow

Filters | Histograms | Frequency Domain | Hybird

Load Image

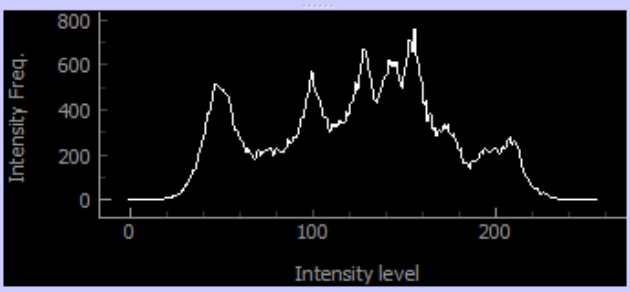
Input Image

Output options

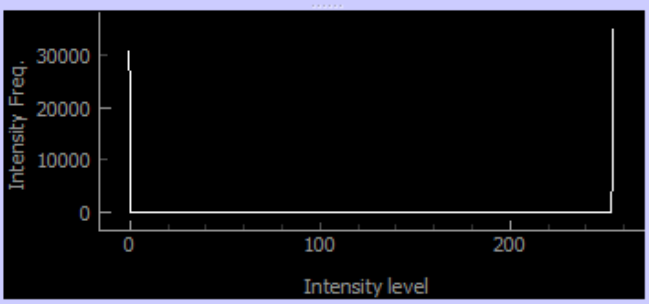
Global Threshold

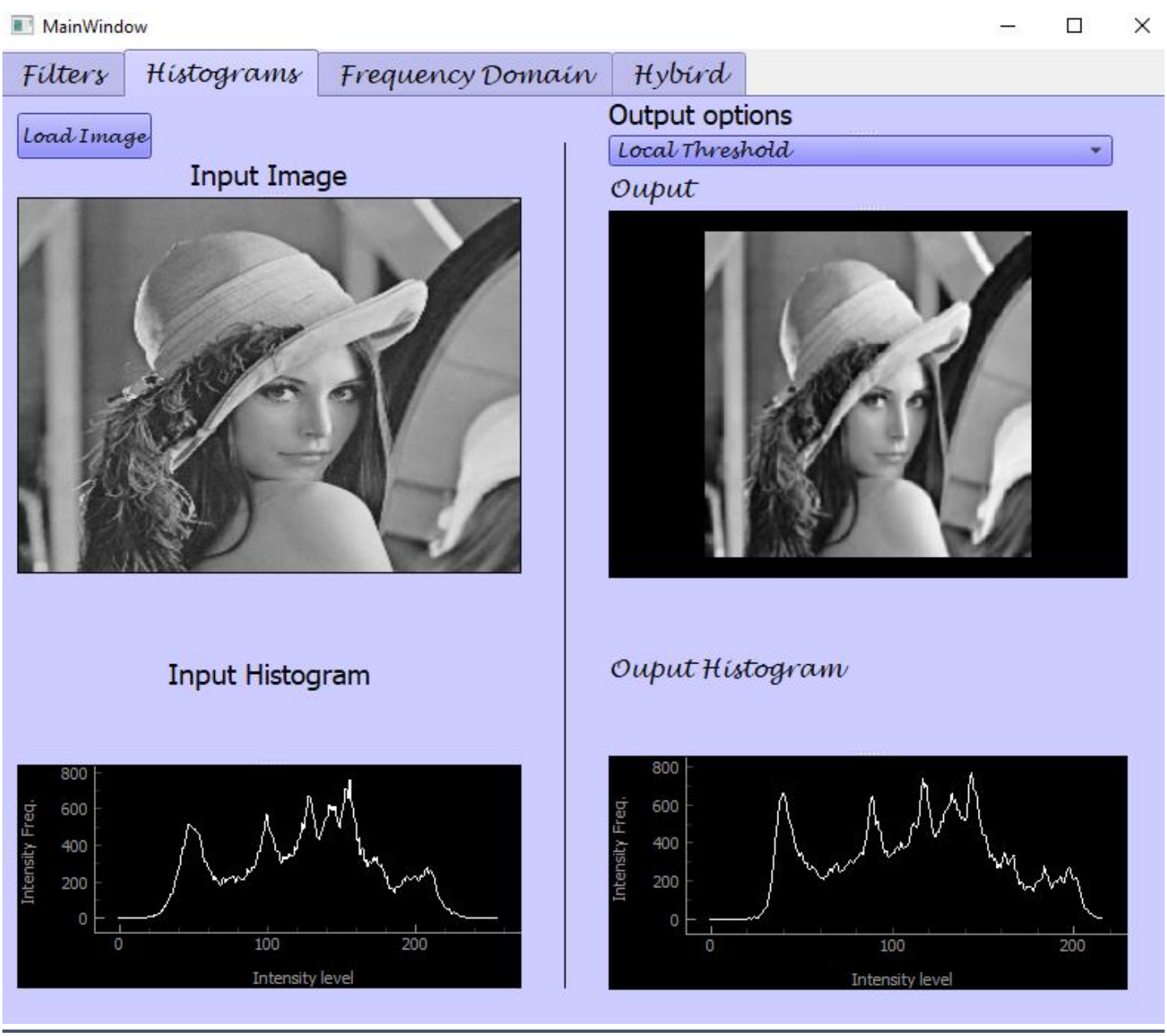
Ouput

Input Histogram



Ouput Histogram





4-Frequency Domain Filters:

a- We get the fourier transform of the grayscale image
b- Shifting is applied so that dc frequency components are in the middle. (instead of being at the 4 corners)
c- We get the log of the magnitude of the complex output of fourier.

d- We create a mask:

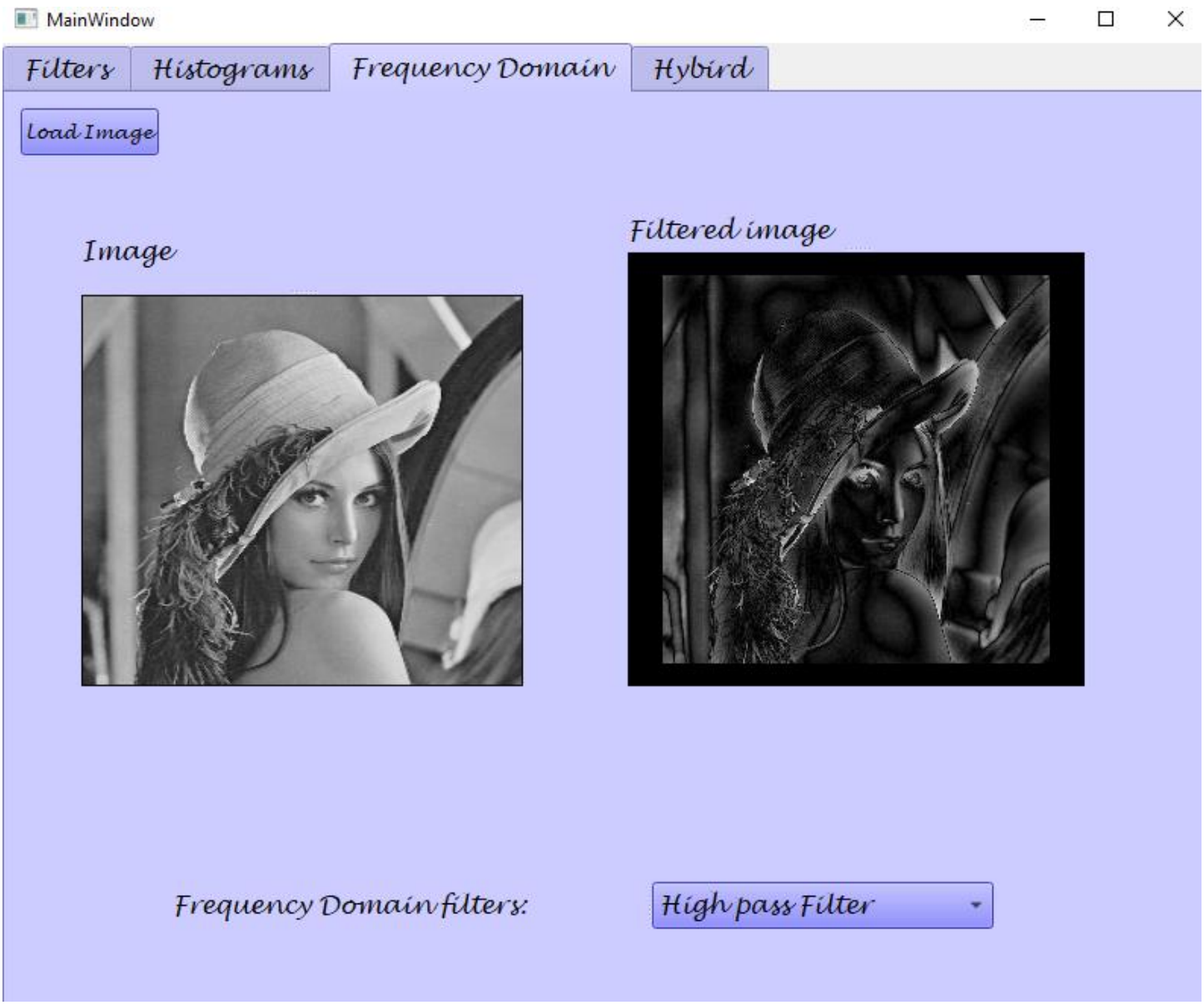
1) High Pass filter:

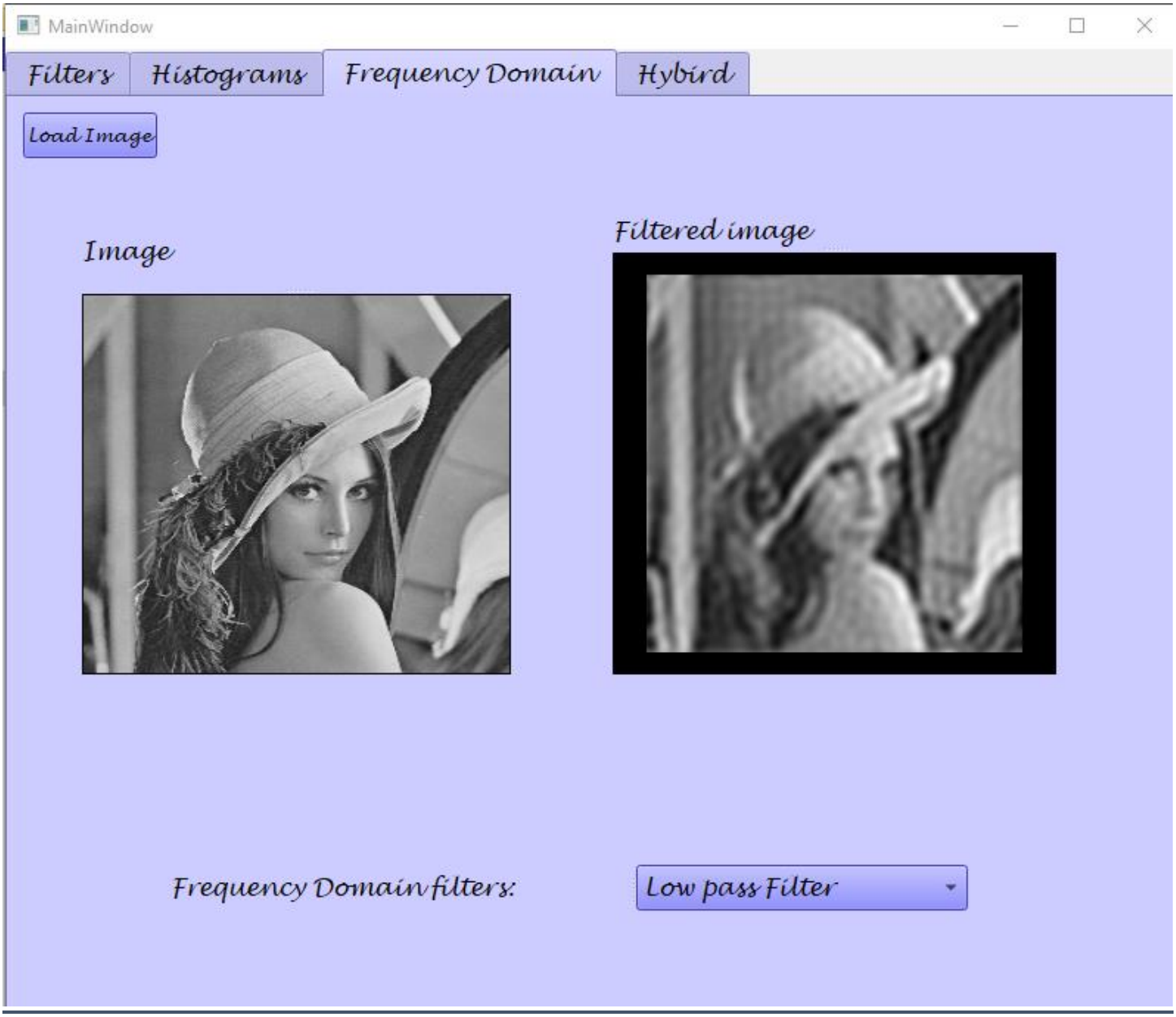
   A circular mask is created. It is centered at a matrix of the size of the image. All the dc components get multiplied by
   0 and the higher frequencies get multiplied by 1.

2) Low pass filter:

   All the dc components get multiplied by 1, and the higher frequencies get multiplied by 0.

The radius of the mask controls the details appearing in the image.





5-Hybrid Images:

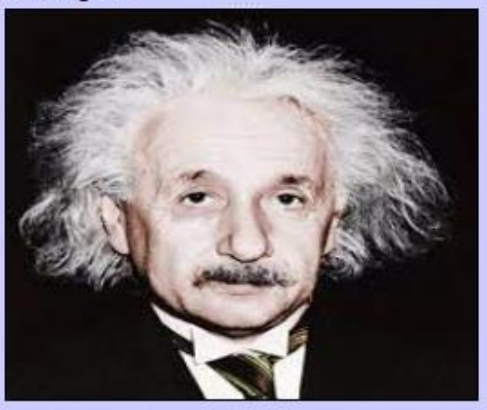
Low pass filter is applied on an image, and high pass filter is applied on another image and then they get added together.

MainWindow

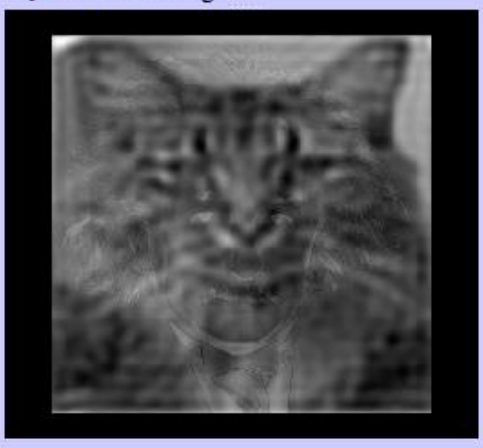
Filters | Histograms | Frequency Domain | Hybird

Load Image1

Image1



Hybird Image



Load Image2

Make Hybrid

Image2