

# Digital Image Processing

Alex Peyrard

December 25, 2014

## 1 Introduction

All of the programs are written in python 3. The shebangs are included and thus they should work if called using the ”./program.py” notation. In case this causes a problem, please try to call them using ”python3 program.py” notation.

I will provide further help on how to call each program.

### 1.1 libraries

I used numpy in all of the programs and scipy in some of them. I will detail when scipy is used.

## 2 Exercise 1

The program called ex1.py computes the histogram of a greyscale image, and enhances the image using histogram equalization. it displays the enhanced image, the histograms of the default and enhanced images, and the transformation function.

In this exercise, the matplotlib library is used to plot histograms and functions.

### 2.1 Examples

#### 2.1.1 Fig1.jpg

All of the results are obtained using the program with the call ”./ex1.py Fig1.jpg”



Figure 1: Original image

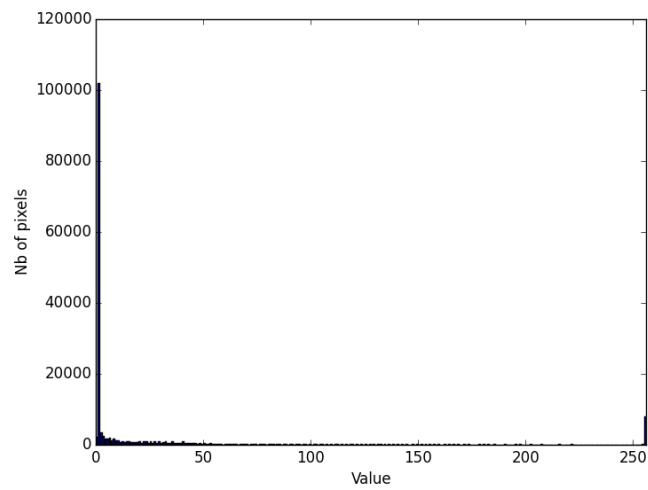


Figure 2: Original image's histogram

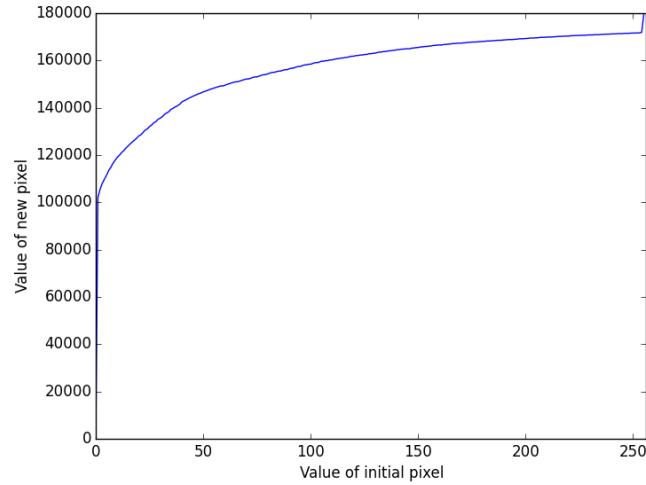


Figure 3: Enhancement function

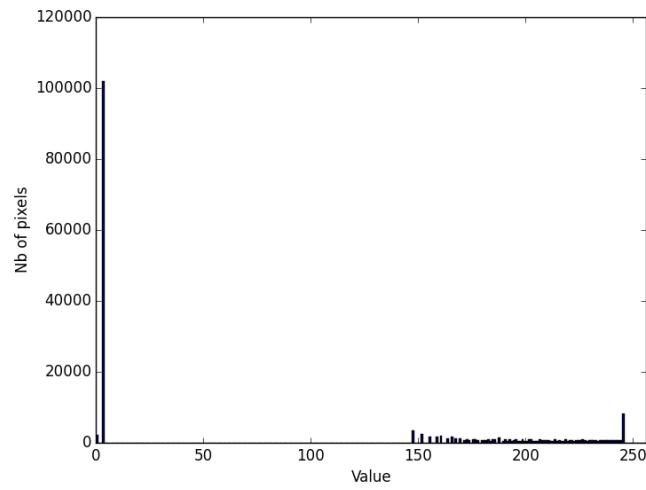


Figure 4: Enhanced image histogram



Figure 5: Enhanced image

### 2.1.2 Fig2.jpg

All of the results are obtained using the program with the call ”./ex1.py Fig2.jpg”

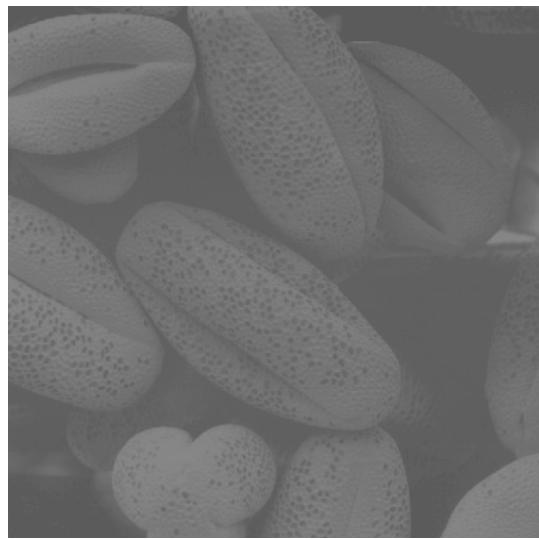


Figure 6: Original image

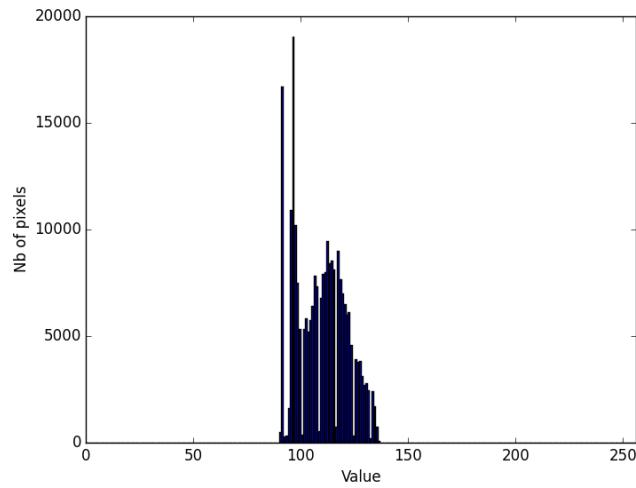


Figure 7: Original image's histogram

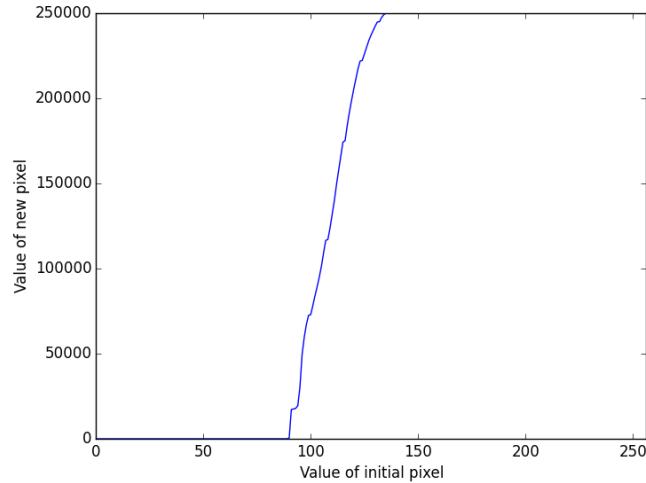


Figure 8: Enhancement function

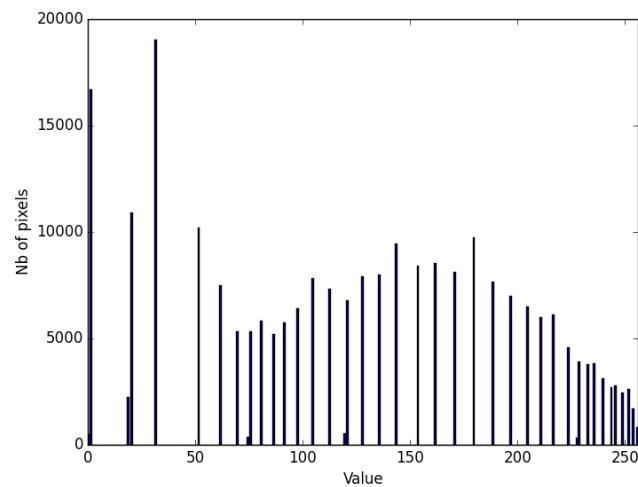


Figure 9: Enhanced image histogram

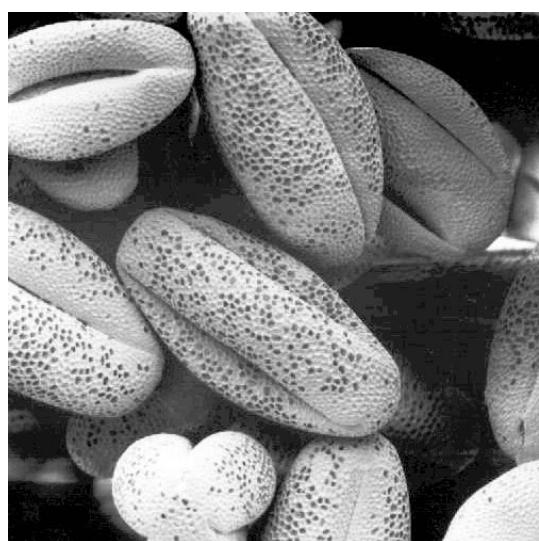


Figure 10: Enhanced image

### 3 Exercise 2

The program called ex2.py performs several spatial enhancement techniques on a given greyscale image.

#### 3.1 Example on skeleton\_orig.tif

All of the results are obtained using the program with the call ”./ex2.py skeleton\_orig.tif”

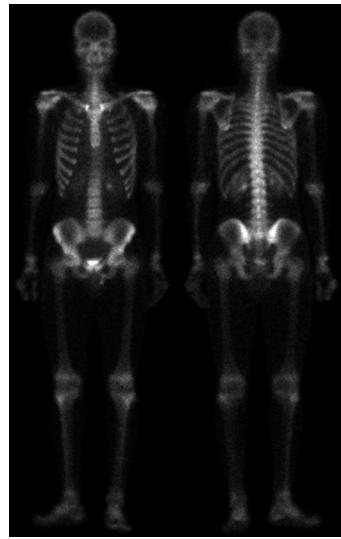


Figure 11: Original image



Figure 12: Rescaled laplacian of image

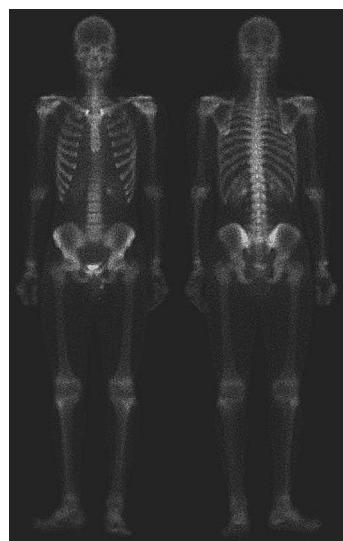


Figure 13: Sum of laplacian and original image

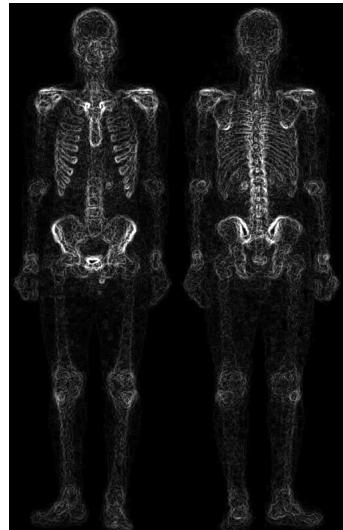


Figure 14: Sobel gradient of original image

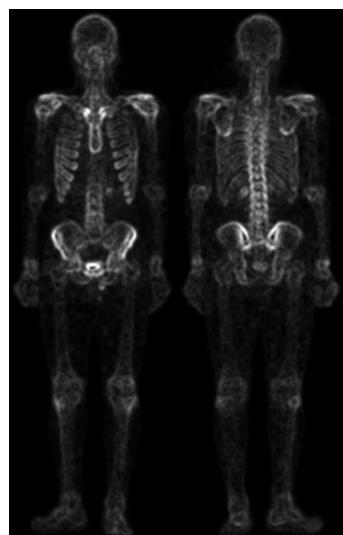


Figure 15: Smoothed Sobel gradient of original image

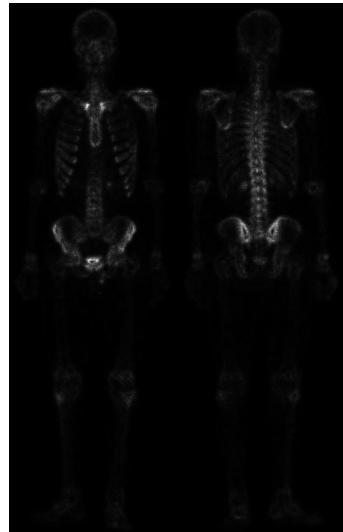


Figure 16: Product of smoothed Sobel gradient and of the previous sum of laplacian and original

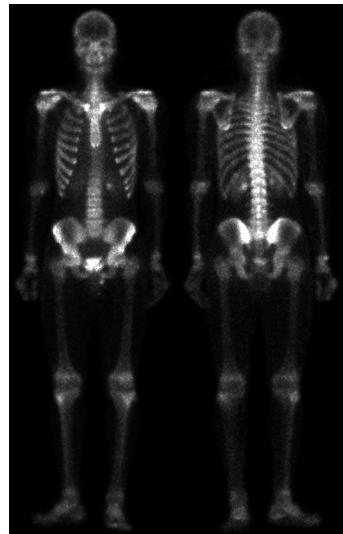


Figure 17: Sum of original image and previous image

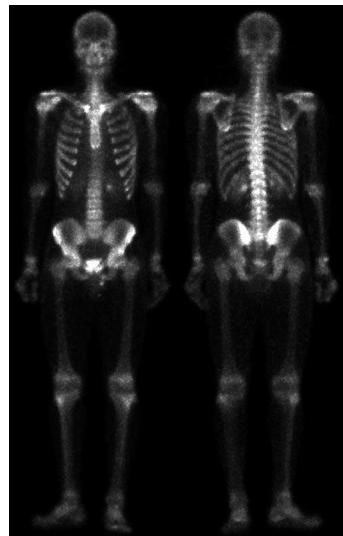


Figure 18: Power law of original image, gamma = 0.5, c = 1

## 4 Exercise 3

The program called ex3.py performs several frequency domain enhancement techniques on a given greyscale image.

### 4.1 Examples

#### 4.1.1 Ideal filter

Those images are obtained using the call ”./ex3 –ideal –highpass cutoff image” for highpass images, and ”./ex3 –ideal –lowpass cutoff image” for lowpass images.

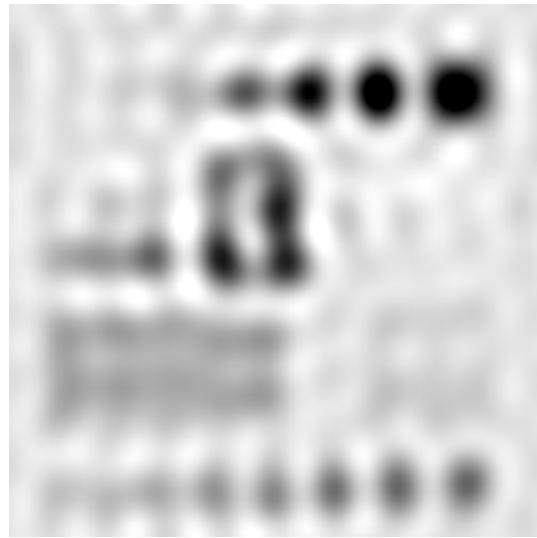


Figure 19: Cutoff 10 Ideal lowpass filter



Figure 20: Cutoff 30 Ideal lowpass filter

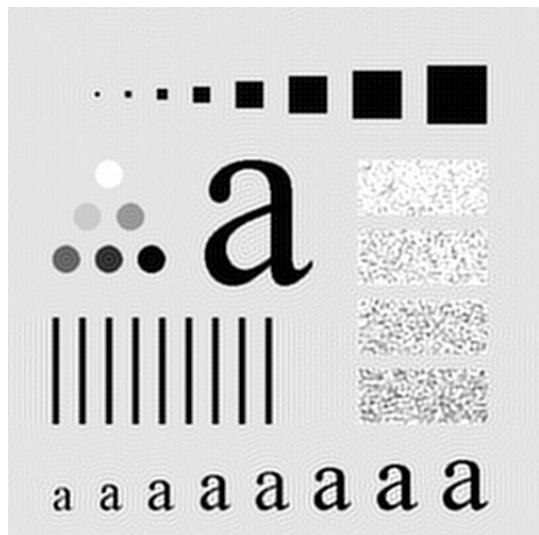


Figure 21: Cutoff 100 Ideal lowpass filter

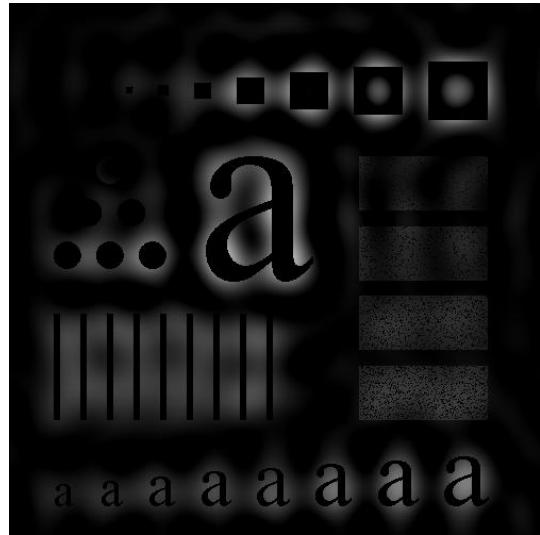


Figure 22: Cutoff 10 Ideal highpass filter

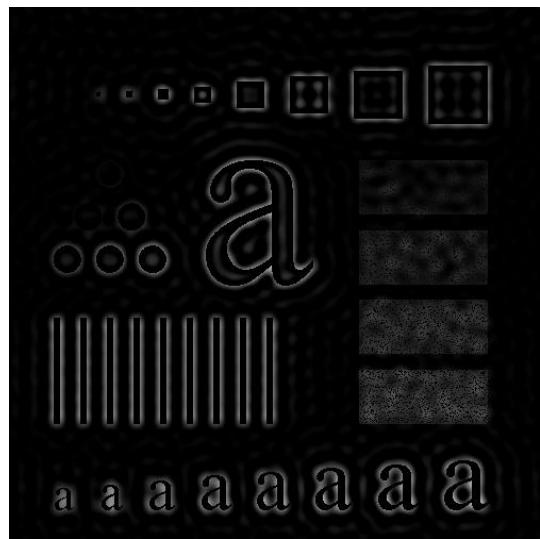


Figure 23: Cutoff 30 Ideal highpass filter

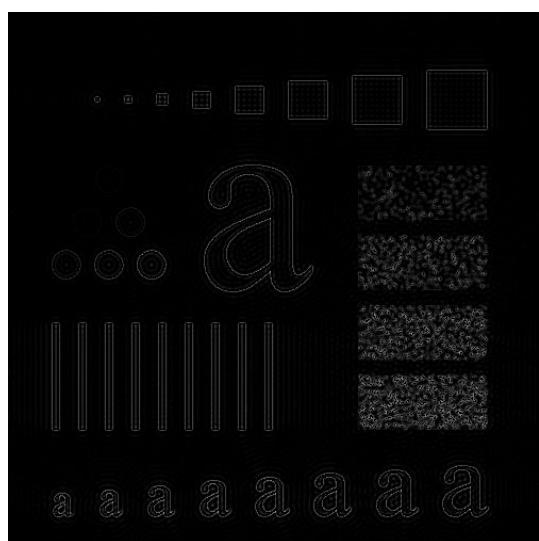


Figure 24: Cutoff 100 Ideal highpass filter

#### 4.1.2 Gaussian filter

Those images are obtained using the call ”./ex3 –gaussian –highpass cutoff image” for highpass images, and ”./ex3 –gaussian –lowpass cutoff image” for low-pass images.

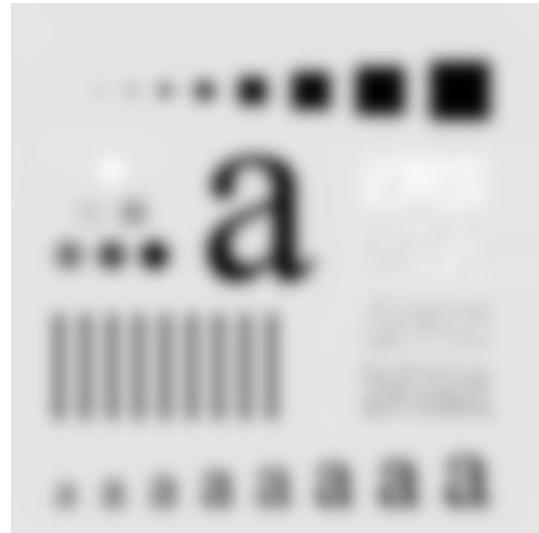


Figure 25: Cutoff 10 Gaussian lowpass filter



Figure 26: Cutoff 30 Gaussian lowpass filter

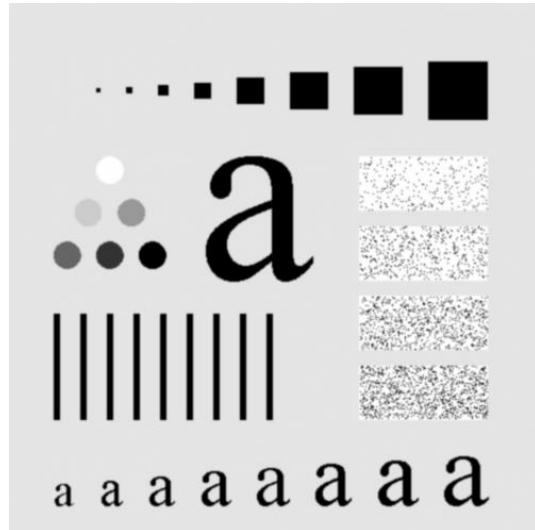


Figure 27: Cutoff 100 Gaussian lowpass filter

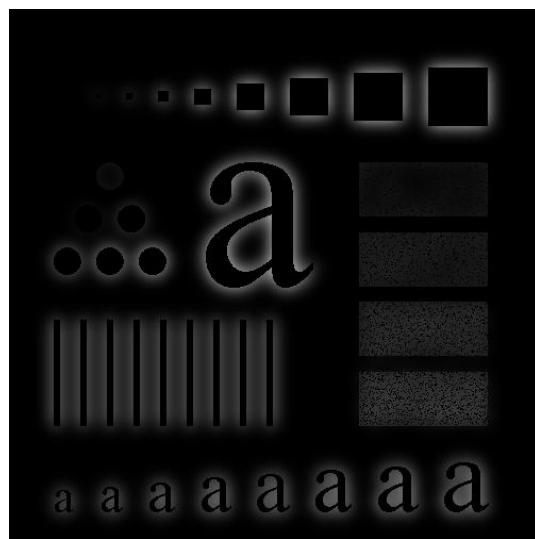


Figure 28: Cutoff 10 Gaussian highpass filter

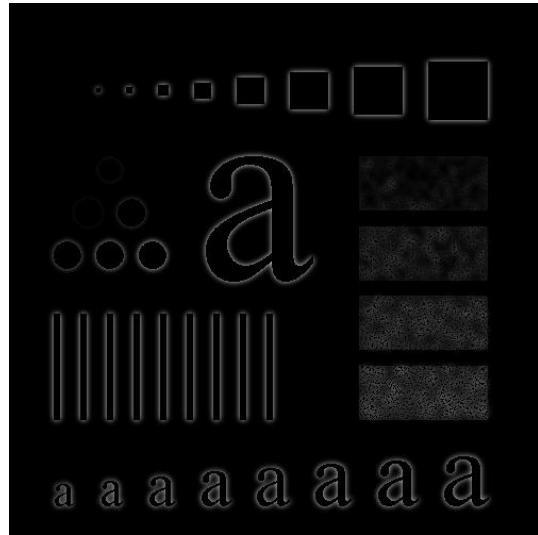


Figure 29: Cutoff 30 gaussian highpass filter

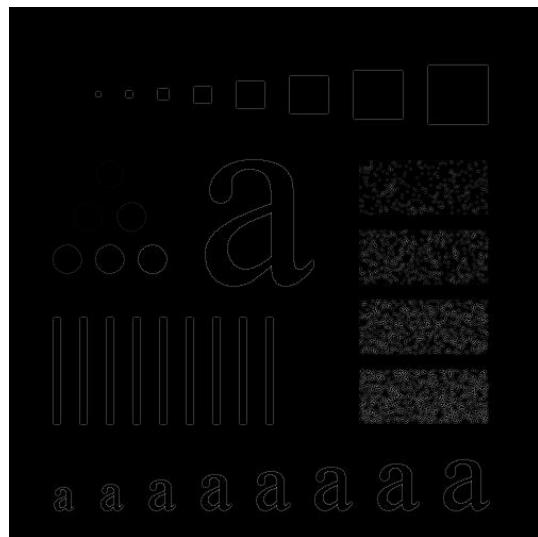


Figure 30: Cutoff 100 Gaussian highpass filter

#### 4.1.3 Butterworth filter

Those images are obtained using the call ”./ex3 –butterworth –highpass –order order cutoff image” for highpass images, and ”./ex3 –butterworth –lowpass –order order cutoff image” for lowpass images.

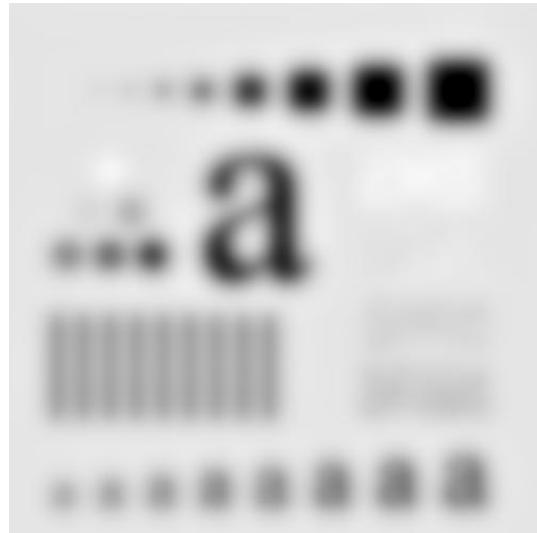


Figure 31: Cutoff 10 order 2 Butterworth lowpass filter



Figure 32: Cutoff 30 order 2 Butterworth lowpass filter

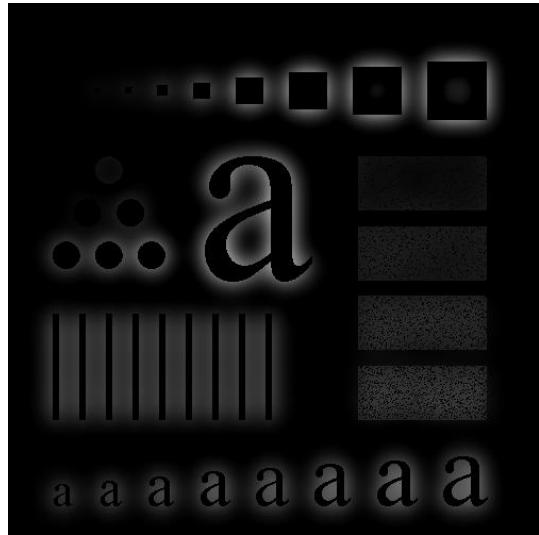


Figure 33: Cutoff 10 order 2 Butterworth highpass filter

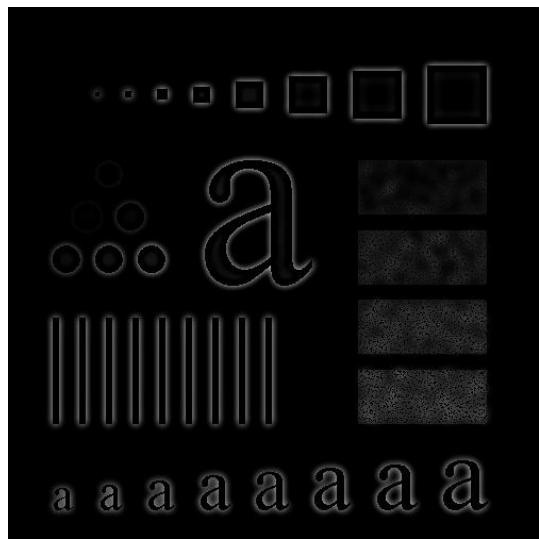


Figure 34: Cutoff 30 order 2 Butterworth highpass filter

## 5 Exercise 4

For exercise 4, the program gaussNoise.py adds gaussian noise of desired mean and variance  $\sigma$  to an image. The program uniNoise.py adds uniform noise of desired lower and upper bound to an image. The program filtering.py uses different means of filtering an image.

### 5.1 Examples

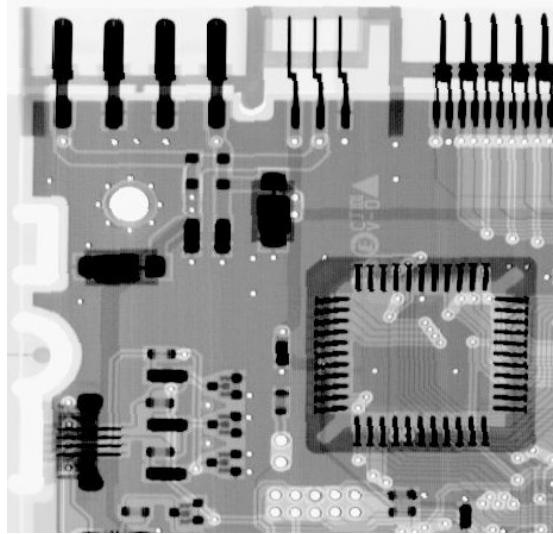


Figure 35: Original image

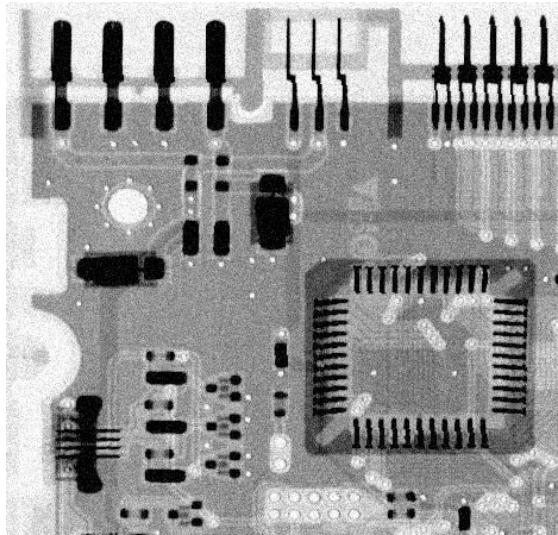


Figure 36: Original image with gaussian noise of mean 0 and variance 260

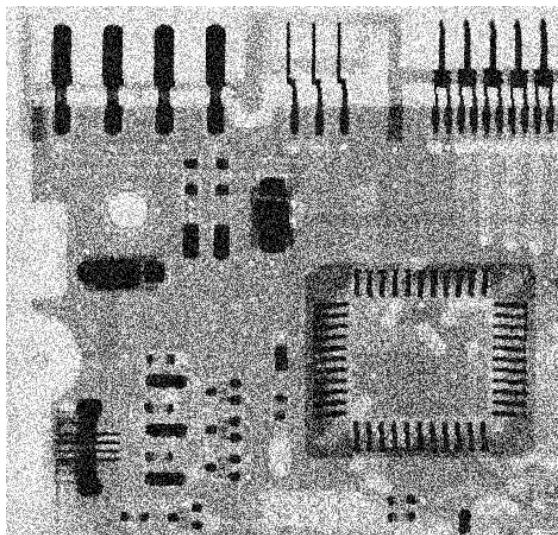


Figure 37: Original image with uniform noise in the [-100, 100] range

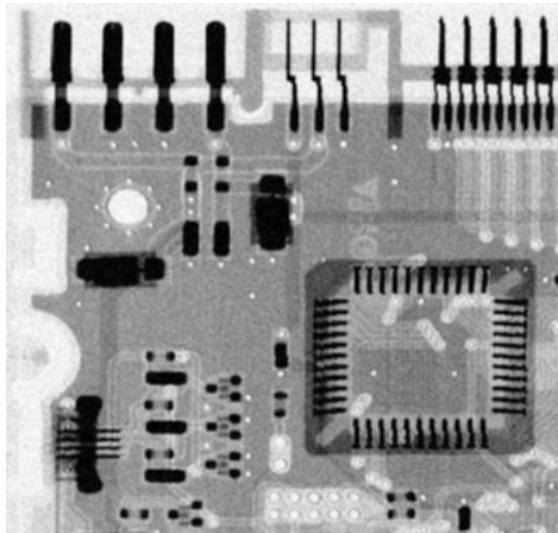


Figure 38: Image with gaussian noise enhanced with arithmetic mean

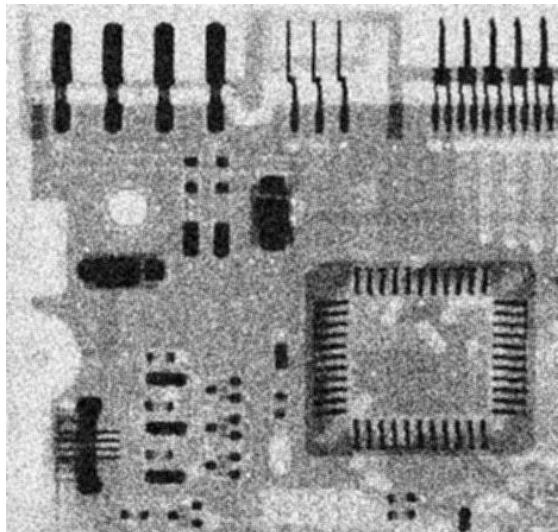


Figure 39: Image with uniform noise enhanced with arithmetic mean

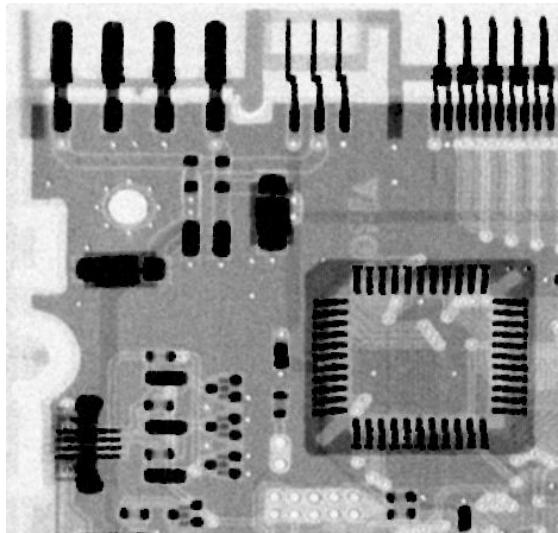


Figure 40: Image with gaussian noise enhanced with geometric mean

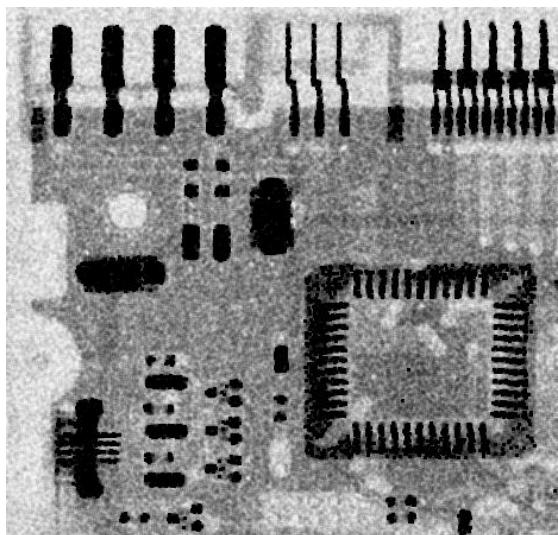


Figure 41: Image with uniform noise enhanced with geometric mean

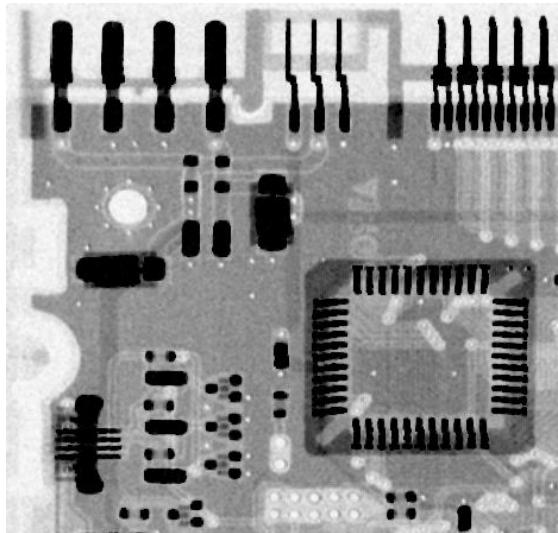


Figure 42: Image with gaussian noise enhanced with harmonic filter

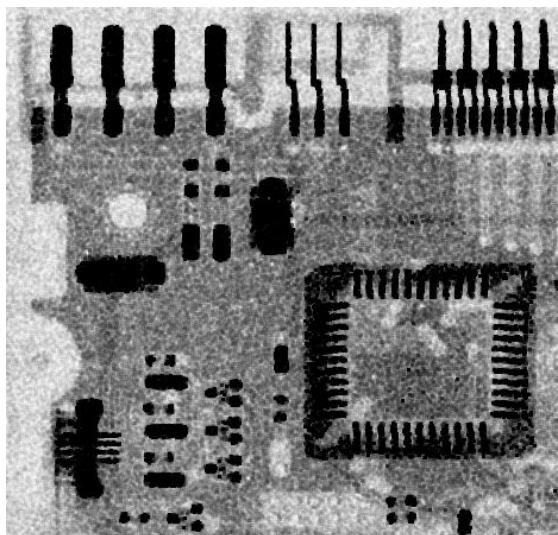


Figure 43: Image with uniform noise enhanced with harmonic filter

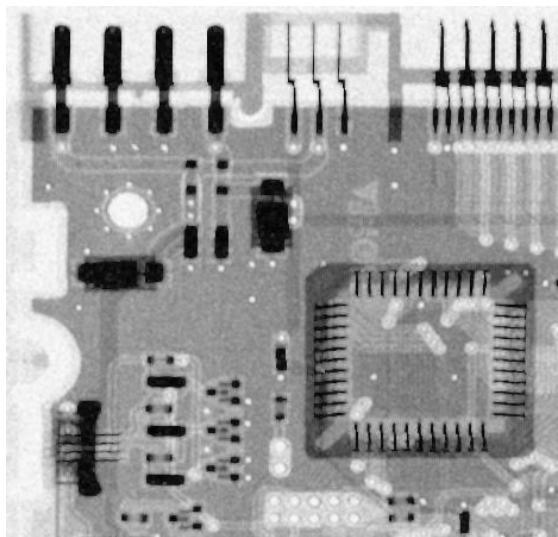


Figure 44: Image with gaussian noise enhanced with contra harmonic filter of order 1.5

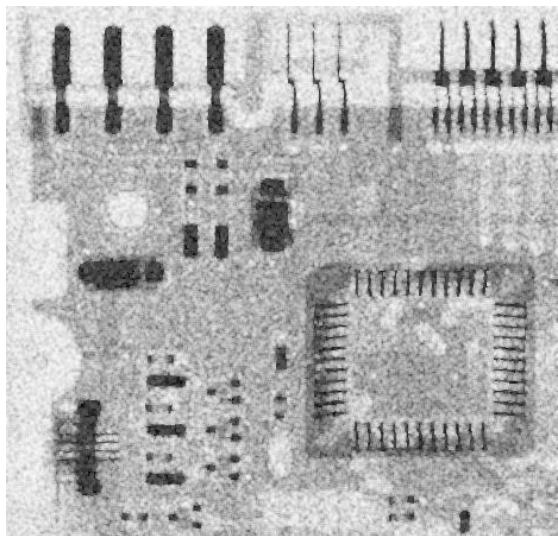


Figure 45: Image with uniform noise enhanced with contra harmonic filter of order 1.5

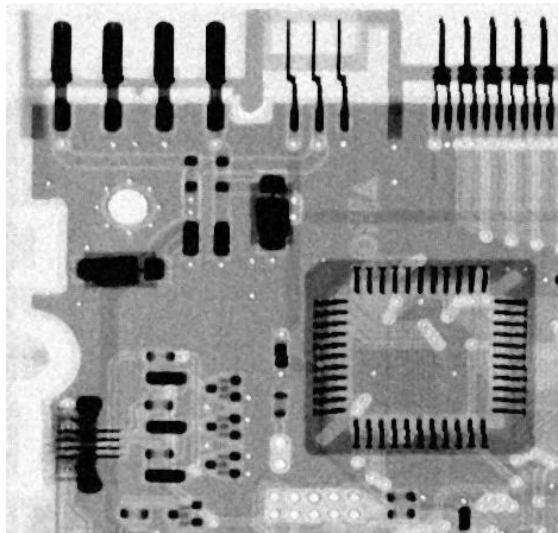


Figure 46: Image with gaussian noise enhanced with median filter

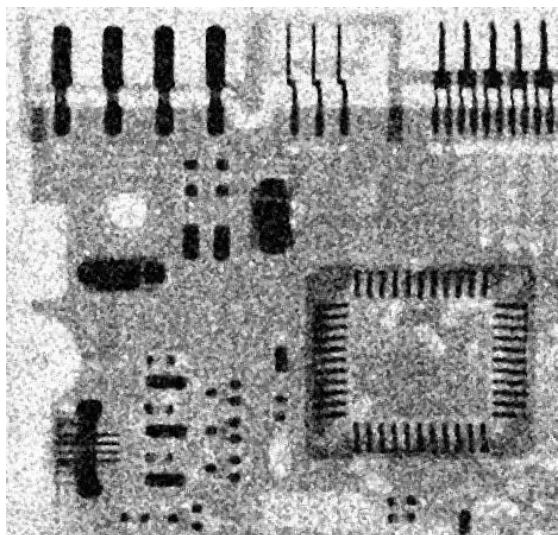


Figure 47: Image with uniform noise enhanced with median filter

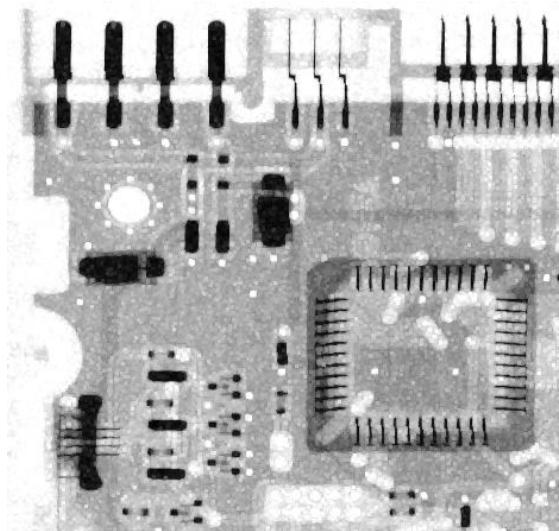


Figure 48: Image with gaussian noise enhanced with max filter

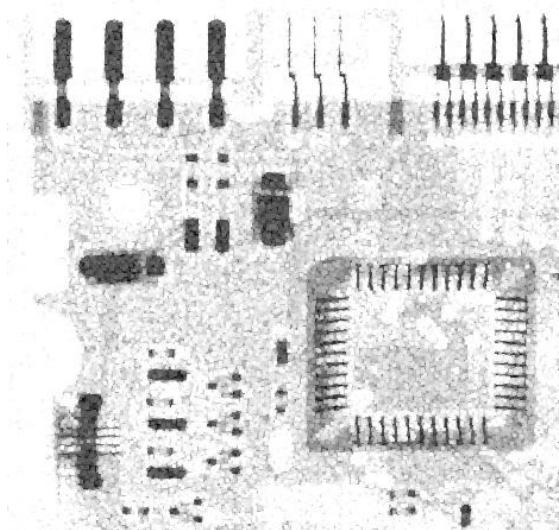


Figure 49: Image with uniform noise enhanced with max filter

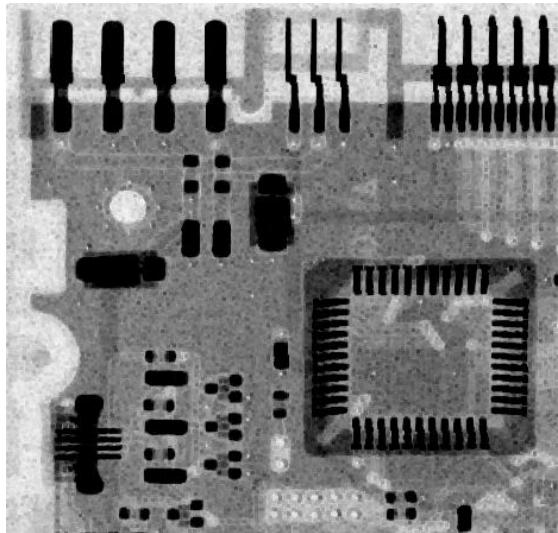


Figure 50: Image with gaussian noise enhanced with min filter

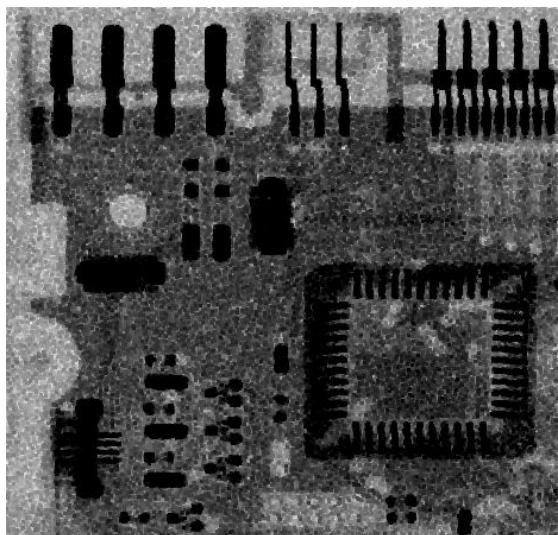


Figure 51: Image with uniform noise enhanced with min filter

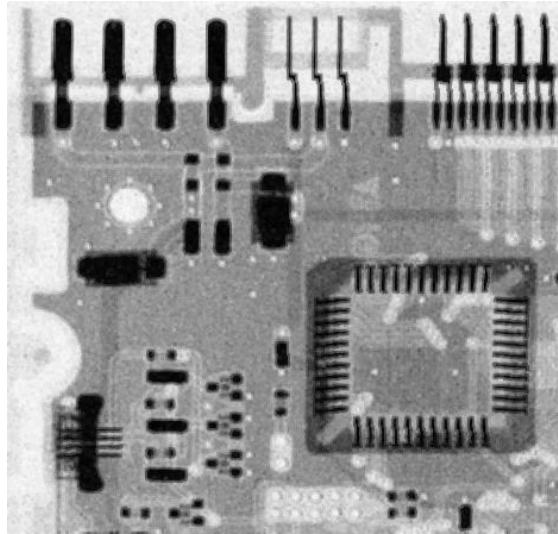


Figure 52: Image with gaussian noise enhanced with midpoint filter

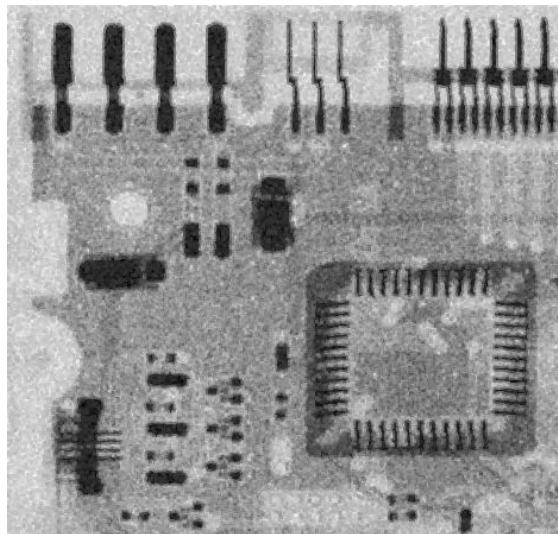


Figure 53: Image with uniform noise enhanced with midpoint filter

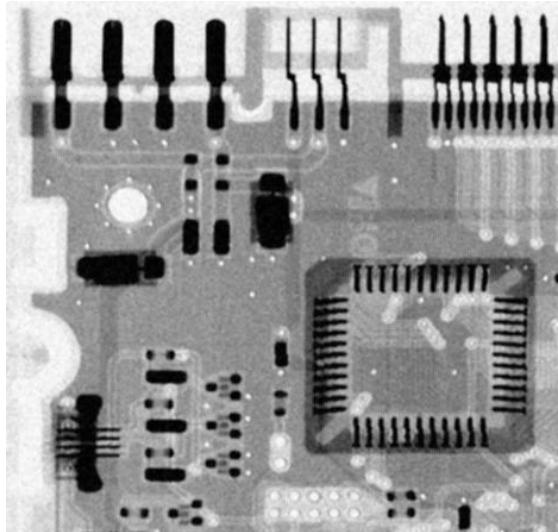


Figure 54: Image with gaussian noise enhanced with alpha trimmed filter of order 2

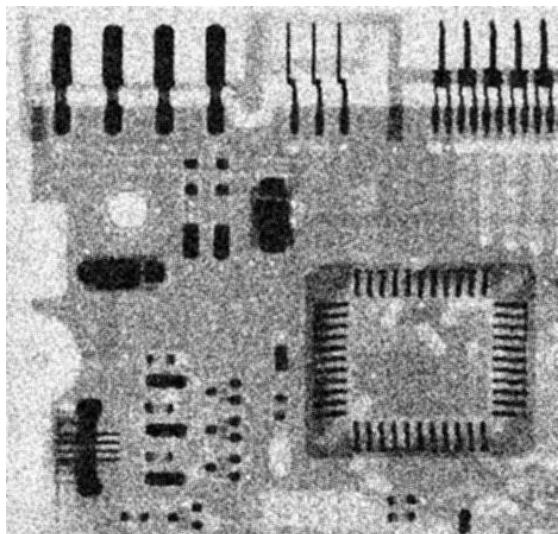


Figure 55: Image with uniform noise enhanced with alpha trimmed filter of order 2

## 6 Exercice 5

In exercise 5 we analyze blurring degradation.

Using "./ex5 blur image output" The program blurs and saves an image.

By using "./ex5 filter image output", the program enhances a blurred image using the inverse filter.

## 7 Example

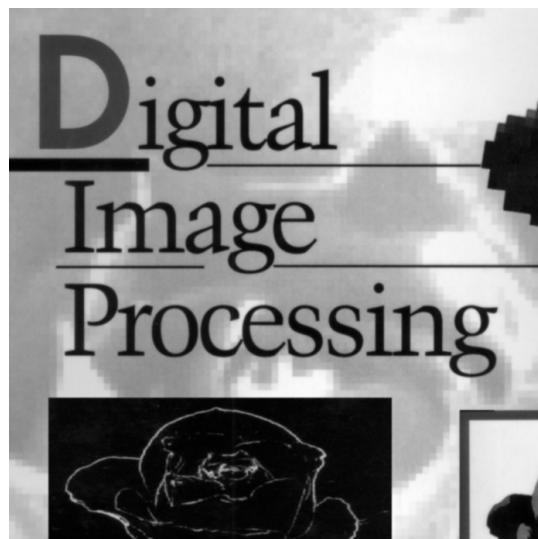


Figure 56: Original image



Figure 57: Blurred image



Figure 58: Blurred image with gaussian noise of mean 0 variance 650



Figure 59: Enhanced image of the blurred book

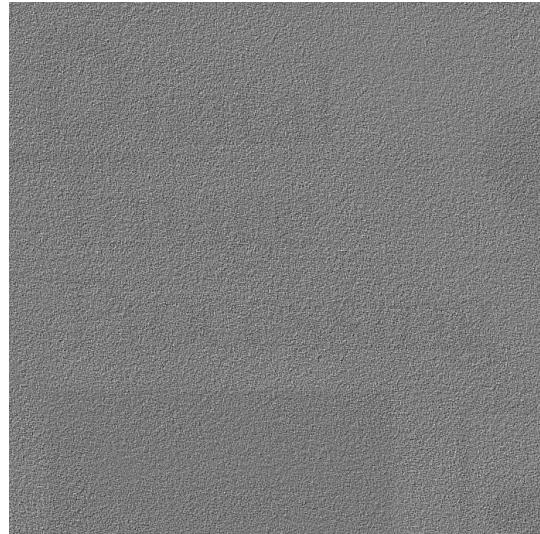


Figure 60: Enhanced image of the blurred book with noise

We can notice that the enhancement with inverse filtering isn't perfect, and is useless with noise.

I apologize for not being able to do the rest of the exercise.

## 8 Exercise 6

This program is capable of performing rotations, translations and rescaling on an image, using either the nearest neighbor or bilinear interpolation.

Usage :

- neighbor for nearest neighbor
- bilinear for bilinear interpolation

-rotate, -rescale, -translate to choose transform

### 8.1 Examples

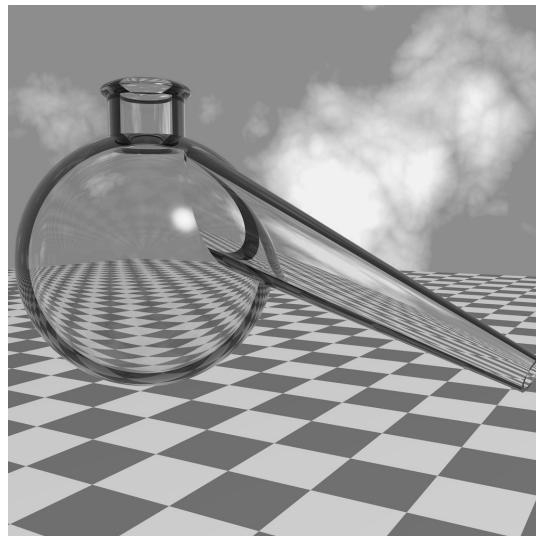


Figure 61: Original image

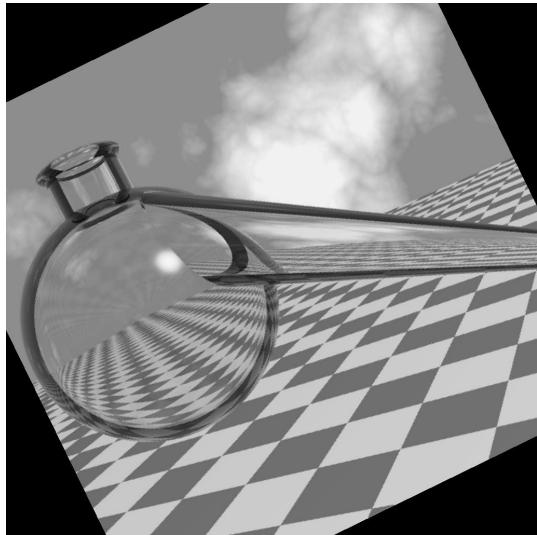


Figure 62: Image rotated by 26 degrees with nearest neighbor. Command line : ./ex6 -neighbor -rotate image 26

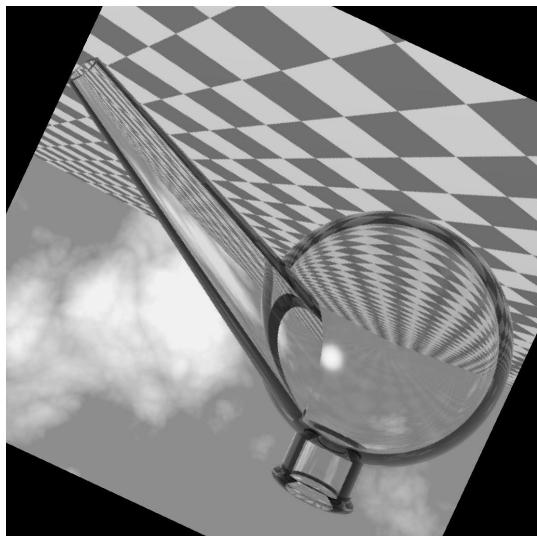


Figure 63: Image rotated by 155 degrees with bilinear interpolation. Command line : ./ex6 -bilinear -rotate image 155

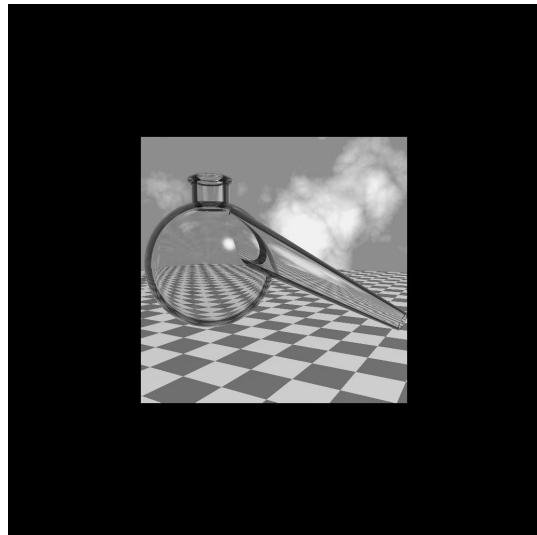


Figure 64: Image rescaled by 0.5 with nearest neighbor. Command line : ./ex6  
-neighbor -rescale image 0.5

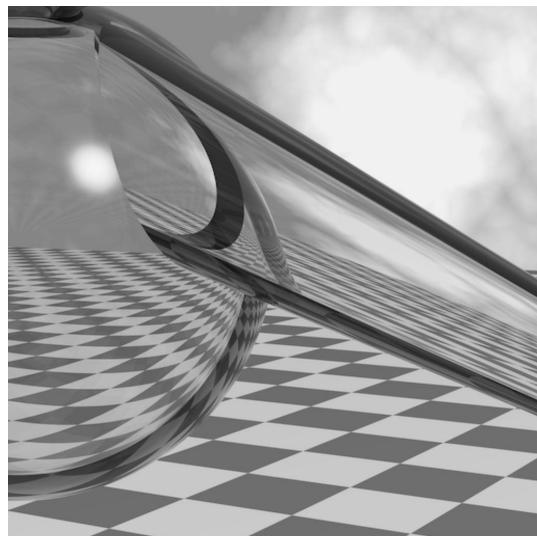


Figure 65: Image rescaled by 2 with bilinear interpolation. Command line :  
./ex6 -bilinear -rescale image 2

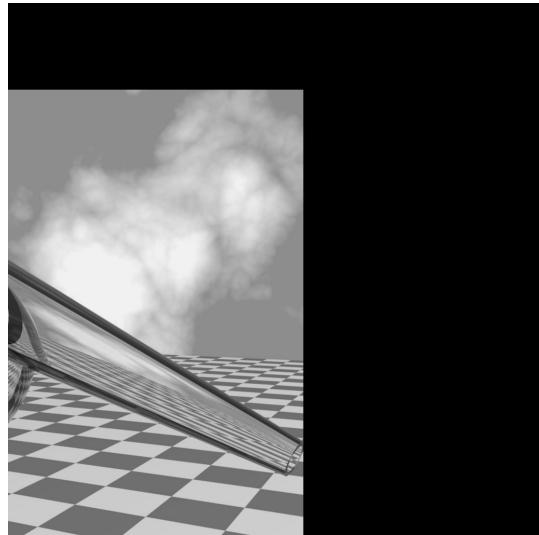


Figure 66: Image translated by 166.5 -455.3 with nearest neighbor. Command line : ./ex6 –neighbor –translate image 166.5 -455.3

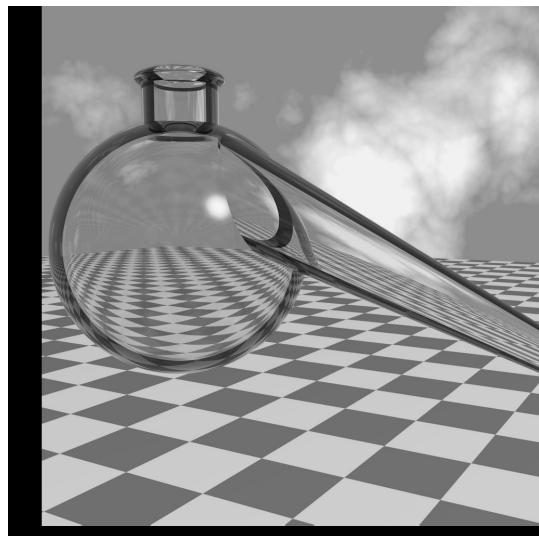


Figure 67: Image translated by -25.12 65 with bilinear interpolation. Command line : ./ex6 –bilinear –translate image -25.12 65

## 9 Exercise 7

The exercise 7 program can compress an image using the discrete cosine transform and a mask to discard coefficients.

Usage :  
-dct to perform the dct  
-showdiff to show the differences between the original image and the compressed image  
-threshold to use the threshold mask instead of the zonal mask.

### 9.1 Examples



Figure 68: Original image



Figure 69: Image compressed using the zonal mask

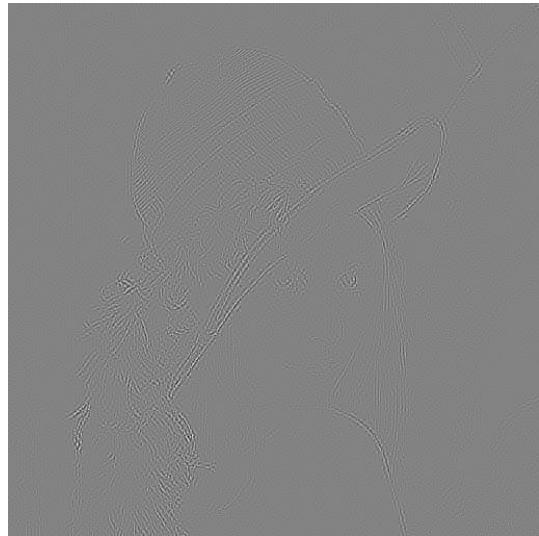


Figure 70: Differences between original and zonal compressed images



Figure 71: Image compressed using the zonal mask

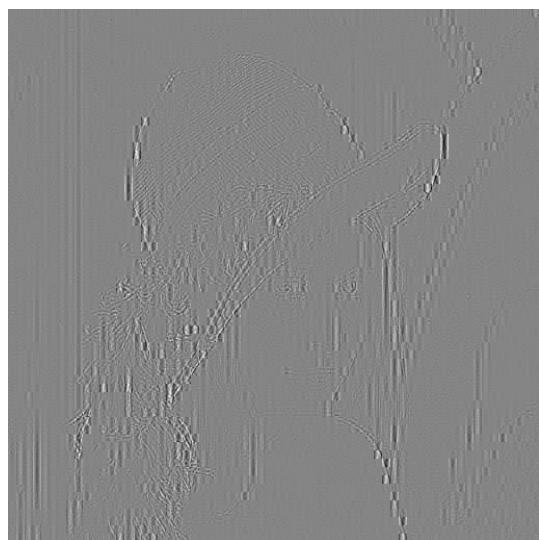


Figure 72: Differences between original and zonal compressed images

I wasn't able to do the part of the exercise with wavelets.

## 10 Exercise 8

The exercise 8 program can perform morphological processing on an image.

Usage :

- dilate to use dilation
  - erode to use erosion
  - open to use opening
  - close to use closing
- 
- boundary to use boundary extraction
  - filling to use hole filling
  - extraction for feature extraction

### 10.1 Examples



Figure 73: Original image



Figure 74: Dilated image. Command ./ex8 -dilate image



Figure 75: Eroded image. Command ./ex8 -erode image



Figure 76: Opened image. Command ./ex8 -open image



Figure 77: Closed image. Command ./ex8 -close image

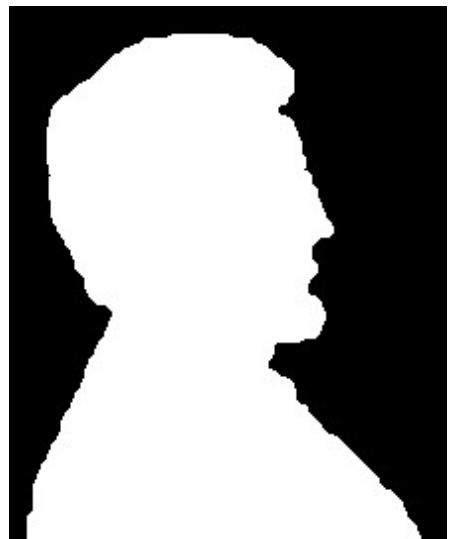


Figure 78: Original image



Figure 79: Boundary of image. Command ./ex8 -boundary image

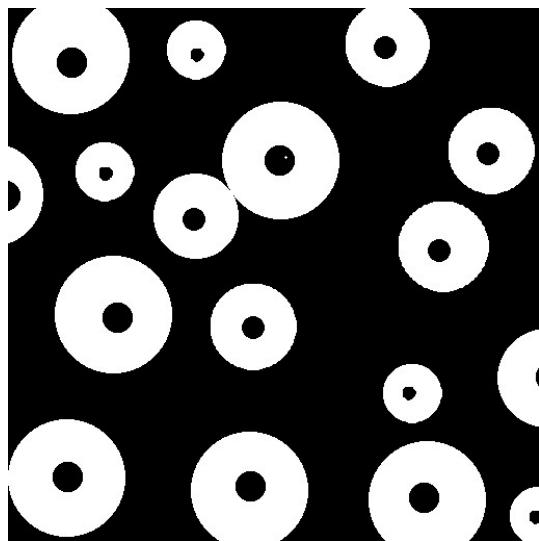


Figure 80: Original image

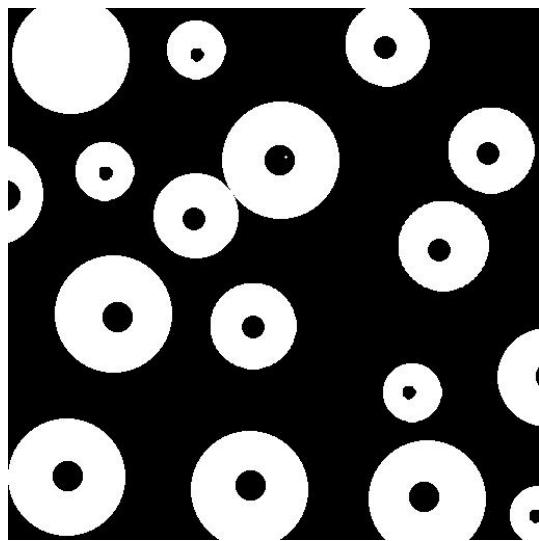


Figure 81: Image filled at position 50 50. Command ./ex8 -filling image 50 50

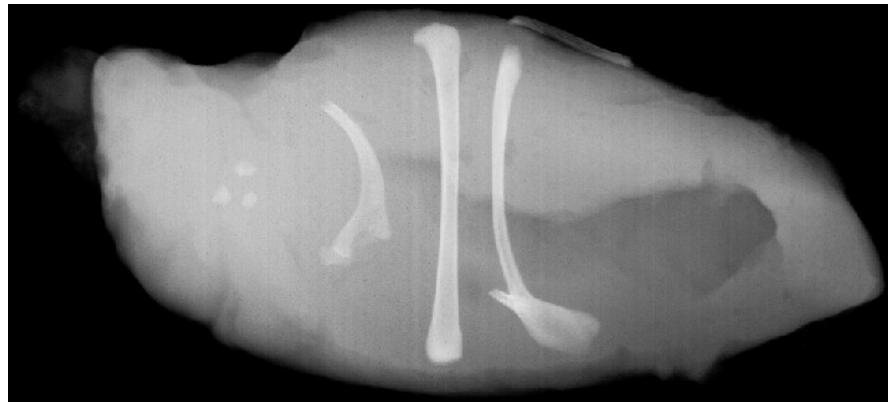


Figure 82: Original image



Figure 83: Image thresholded, eroded, and then extracted at position 100 356.  
Command ./ex8 -extraction image 100 356

The program also gives the number of pixels of the extracted component. In this case : 1356 pixels

## 11 Exercise 9

The exercise 9 program can perform image segmentation on an image using edge detectors or thresholding segmentation methods.

Usage :

- roberts to use the Roberts edge detectors
  - prewitt to use the Prewitt edge detector
  - sobel to use the Sobel edge detector
  - mh to use the Marr hildreth edge detector
  - canny to use the canny edge detector
- otsu to use Otsu's threshold segmentation.  
-threshold to use the global thresholding method

### 11.1 Examples



Figure 84: Original image



Figure 85: Image obtained using Roberts edge detector. Command : ./ex9  
-roberts image



Figure 86: Image obtained using Prewitt edge detector. Command : ./ex9  
-prewitt image



Figure 87: Image obtained using Sobel edge detector. Command : ./ex9 -sobel image



Figure 88: Image obtained using Marr Hildreth edge detector. Command : ./ex9 -mh image



Figure 89: Image obtained using Canny edge detector. Command : ./ex9 – canny image

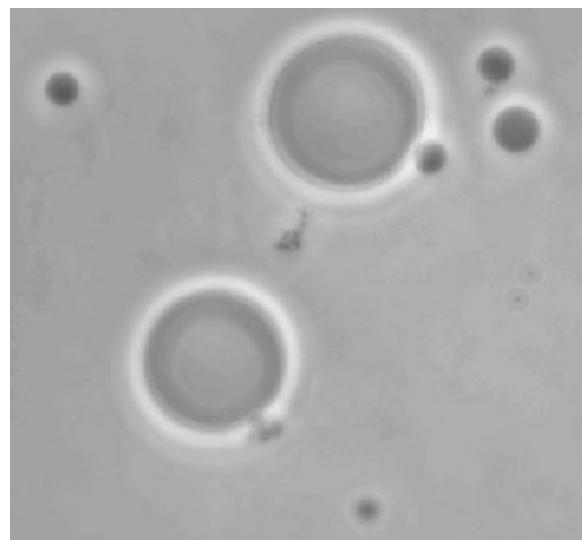


Figure 90: Original image

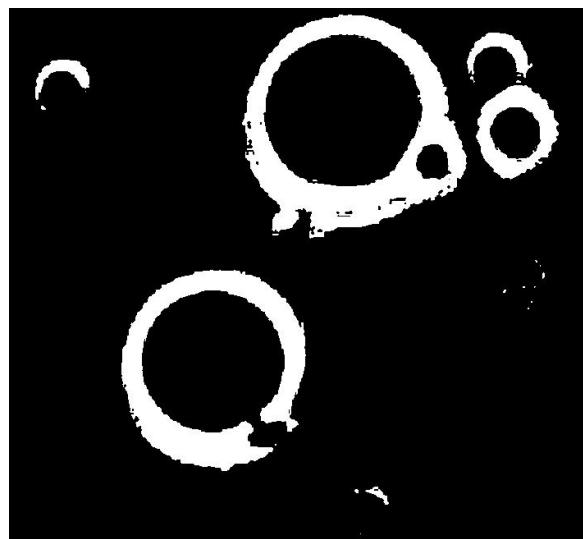


Figure 91: Image obtained using Otsu's method. Command : ./ex9 –otsu image.  
Obtained  $\eta$  : 0.48



Figure 92: Image obtained using global thresholding method method. Com-  
mand : ./ex9 –threshold image.

We can see that Otsu's method is better at determining the best threshold value./

## 12 Exercise 10

The exercise 10 program can perform boundary following, resampling, and compute the chain code and the difference chain code of a grayscale image.

The pca program can use principal component analysis to compress an image and reconstruct it from fewer components.

Usage of ex10:

- boundary to use boundary following
- resampling to use boundary following and resampling
- linking to link the resampled points
- chain to compute the chain code
- chaindiff to compute the difference chain code

Usage of pca: calling pca automatically reconstructs the 6 images from washington DC from 2 components and also prints the difference images.

### 12.1 Examples

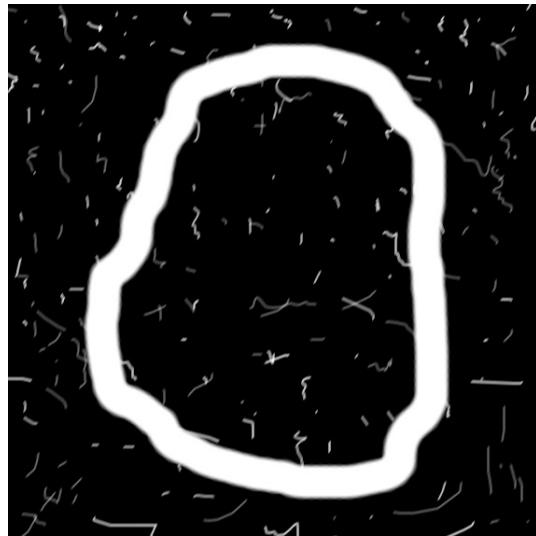


Figure 93: Original image

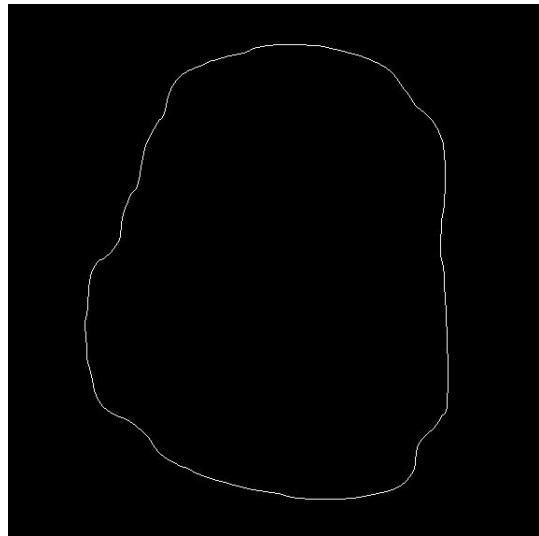


Figure 94: Boundary of the image. Command : ./ex10 -boundary image

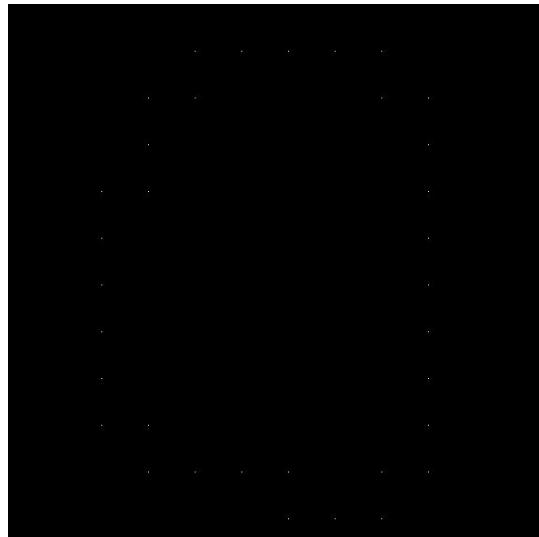


Figure 95: Resampling of the image's boundary. Command : ./ex10 -resampling image

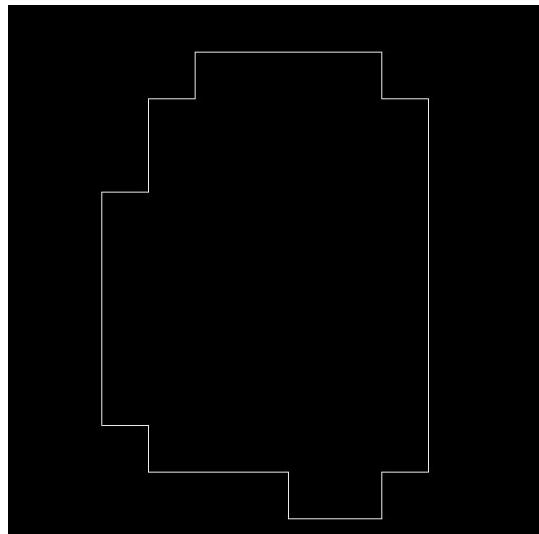


Figure 96: Linking of the previous image. Command : ./ex10 -linking image

The chain code, obtained with ./ex10 -chain image, is 00006066666664644244424222202202

The first difference chain code, obtained with ./ex10 -chaindiff image, is 000626000000062606200626000062062

Images used for pca :



Figure 97: Original image 1

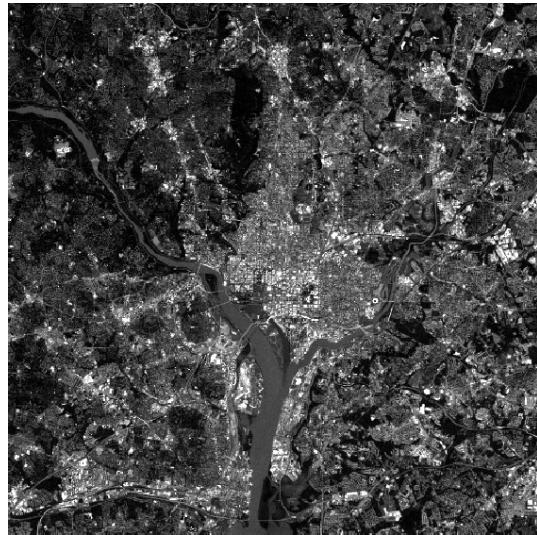


Figure 98: Original image 2



Figure 99: Original image 3

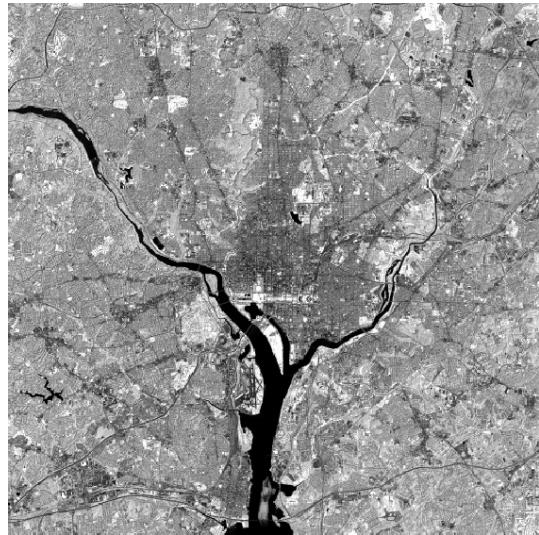


Figure 100: Original image 4

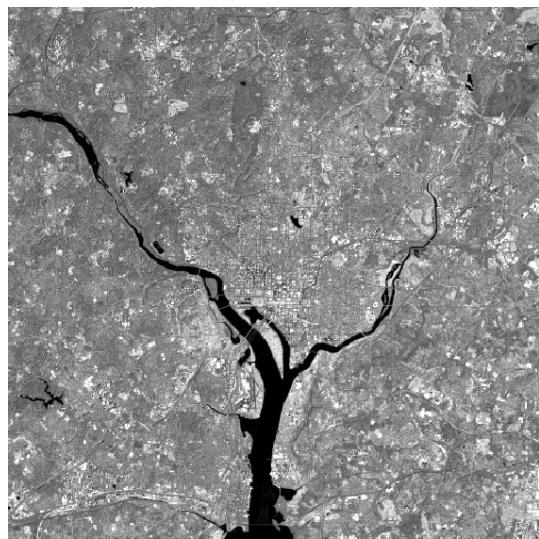


Figure 101: Original image 5

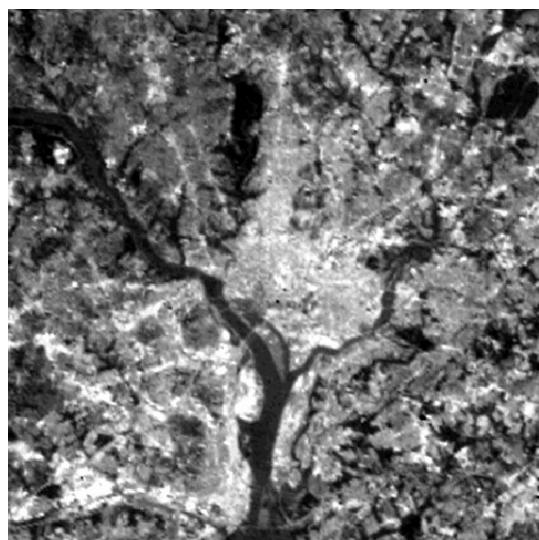


Figure 102: Original image 6

Images obtained with 2 remaining components :

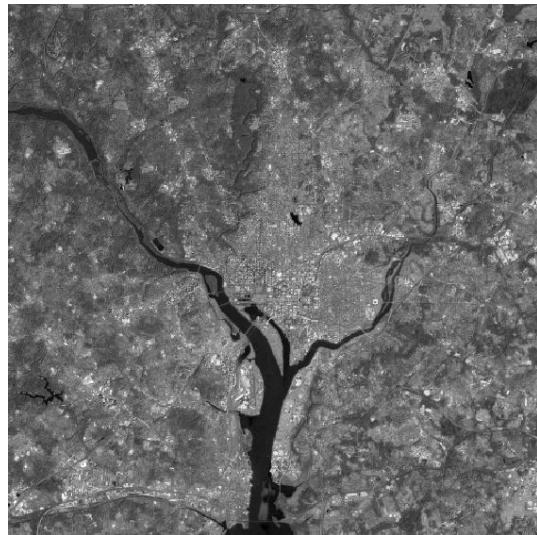


Figure 103: Image 1

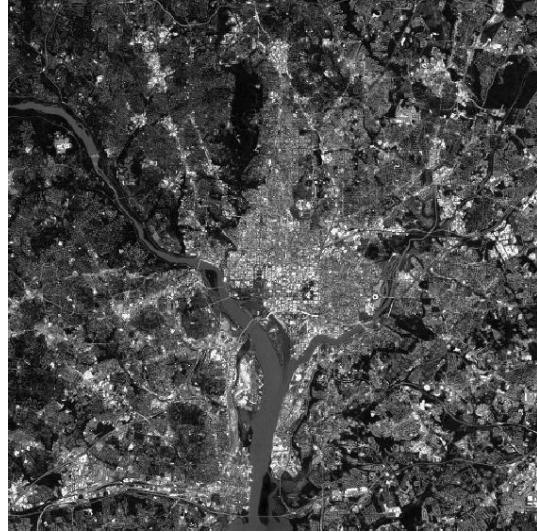


Figure 104: Image 2

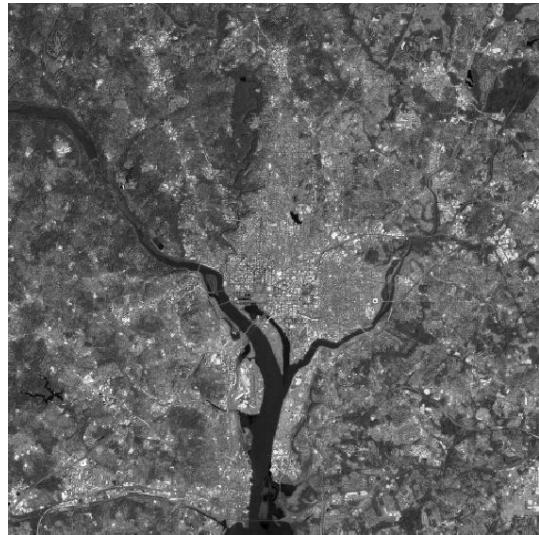


Figure 105: Image 3

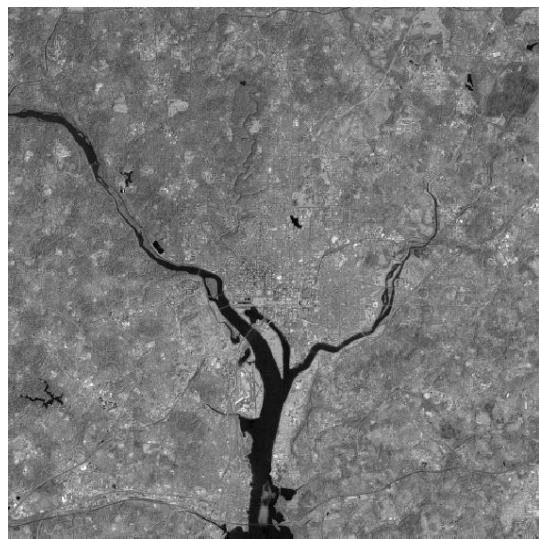


Figure 106: Image 4

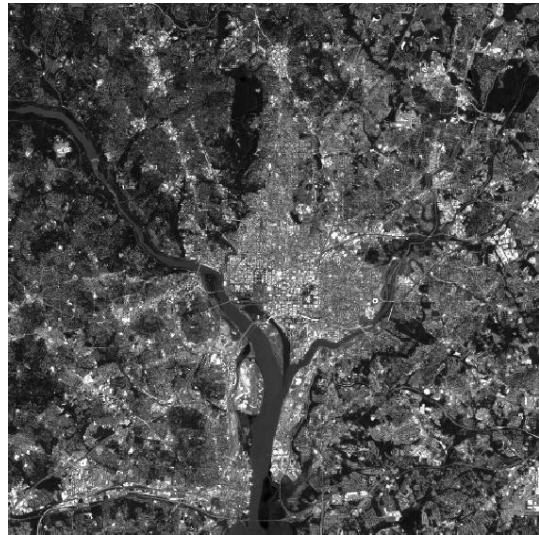


Figure 107: Image 5

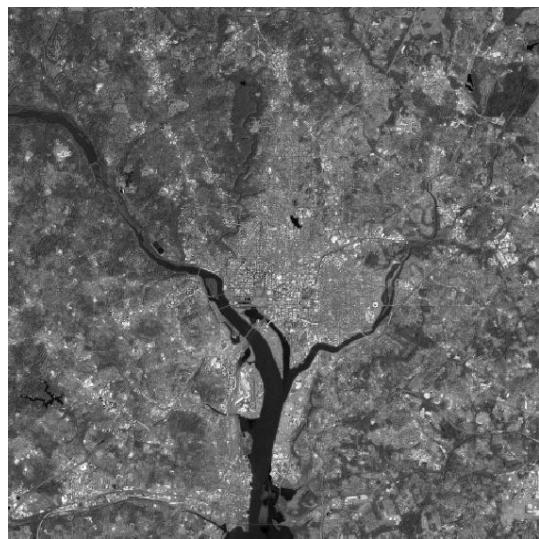


Figure 108: Image 6

Stretched differences images:

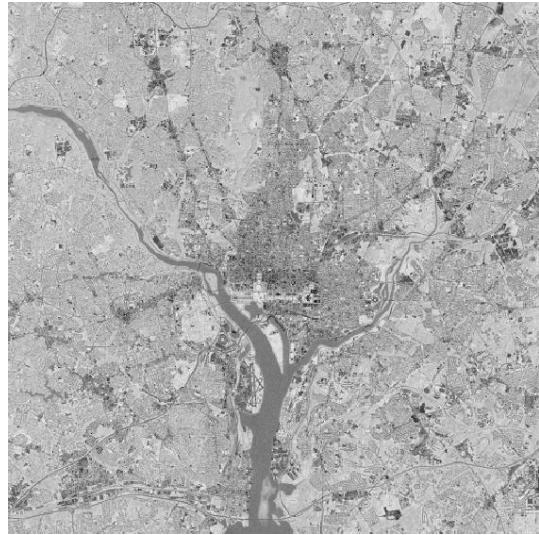


Figure 109: Difference image 1

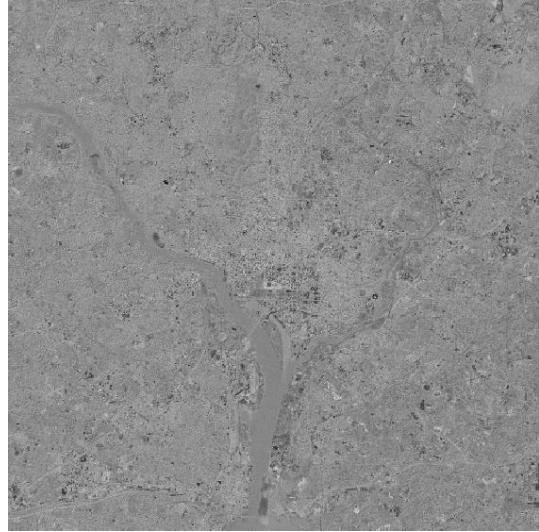


Figure 110: Difference image 2

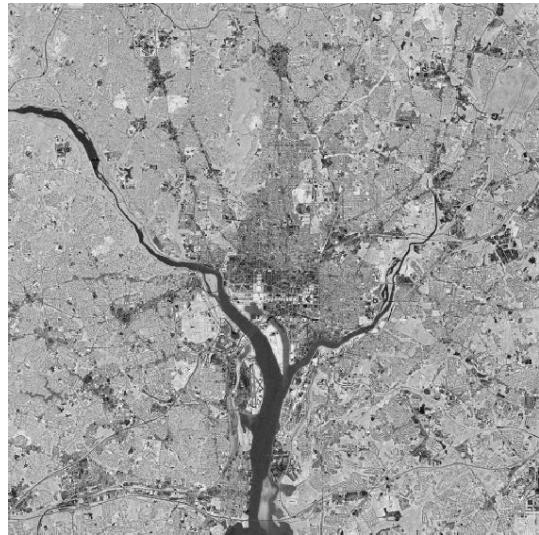


Figure 111: Difference image 3

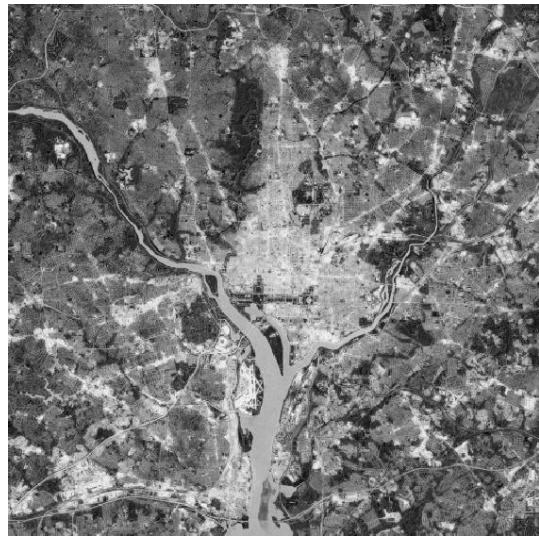


Figure 112: Difference image 4



Figure 113: Difference image 5

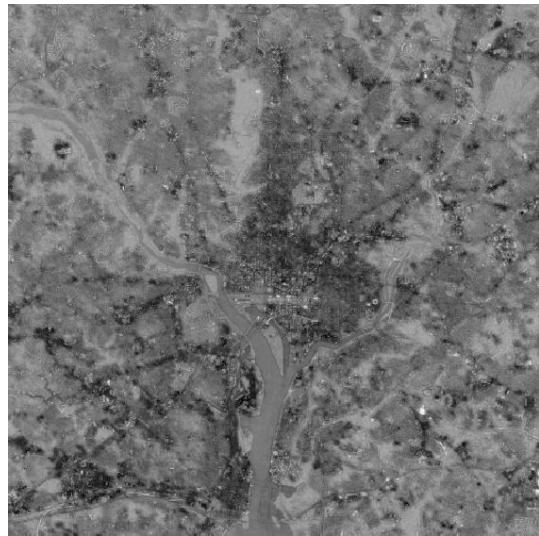


Figure 114: Difference image 6