Week #6 Assignment – Satish Ramachandran

Problem #1

```
'''
Week6 - Problem #1
2D Convolution example
'''

from scipy import signal as sg
import tensorflow as tf

'''
Convolution using SciPy
'''
def convolution_scipy():
    image = [[97, 52, 99, 62, 69, 45, 70],
             [99, 14, 60, 50, 74, 45, 22],
             [59, 72, 74, 14, 74, 100, 28],
             [28, 8, 47, 85, 2, 88, 77],
             [74, 6, 30, 87, 49, 22, 43],
             [86, 87, 4, 53, 36, 10, 46],
             [54, 7, 67, 23, 29,26, 15]]

    gaussian_filter = [[1, 2, 1],
                       [2, 4, 2],
                       [1, 2, 1]]

    conv_full_result = sg.convolve(image, gaussian_filter)
    conv_valid_result = sg.convolve(image, gaussian_filter,"valid")

    print("Full Convolution result:")
    print(conv_full_result)

    print("Valid Convolution result:")
    print(conv_valid_result)
```

```
'''
Convolution usinf Tensorflow
'''
def convolution_tf():
   image_list_tf = tf.constant([[97., 52., 99., 62., 69., 45., 70.],
                           [99., 14., 60., 50., 74., 45., 22.],
                           [59., 72., 74., 14., 74., 100., 28.],
                           [28., 8., 47., 85., 2., 88., 77.],
                           [74., 6., 30., 87., 49., 22., 43.],
                           [86., 87., 4., 53., 36., 10., 46.],
                           [54., 7., 67., 23., 29., 26., 15.]])
   gaussian_filter_list_tf = tf.constant([[1., 2., 1.],
                                  [2., 4., 2.],
                                  [1., 2., 1.]])
   image_tf = tf.reshape(image_list_tf, [1,7,7,1])
   gaussian_filter_tf = tf.reshape(gaussian_filter_list_tf, [3, 3, 1, 1])

   conv_full_result = tf.nn.conv2d(image_tf, gaussian_filter_tf, strides=[1,1,1,1],
padding='SAME')
   conv_valid_result = tf.nn.conv2d(image_tf, gaussian_filter_tf, strides=[1,1,1,1],
padding='VALID')

   with tf.Session() as sess:
      print(conv_full_result.eval())
      print(conv_valid_result.eval())


convolution_scipy()
convolution_tf()
```

The output from both SciPy and Tensorflow are the same as seen below:

```
Full Convolution result:
[[  97  246  300  312  292  245  229  185   70]
 [ 293  704  787  808  818  733  644  459  162]
 [ 354  860  951  914  936  993  903  519  142]
 [ 245  656  832  839  805  944 1045  643  155]
 [ 189  472  575  761  867  823  948  748  225]
 [ 262  631  587  641  871  726  629  560  209]
 [ 300  787  779  613  687  584  436  368  150]
 [ 194  489  534  476  430  349  294  214   76]
 [  54  115  135  164  142  107   96   56   15]]
Valid Convolution result:
[[ 951  914  936  993  903]
 [ 832  839  805  944 1045]
 [ 575  761  867  823  948]
 [ 587  641  871  726  629]
 [ 779  613  687  584  436]]
```

```
[[[[ 704.]
   [ 787.]
   [ 808.]
   [ 818.]
   [ 733.]
   [ 644.]
   [ 459.]]

  [[ 860.]
   [ 951.]
   [ 914.]
   [ 936.]
   [ 993.]
   [ 903.]
   [ 519.]]

  [[ 656.]
   [ 832.]
   [ 839.]
   [ 805.]
   [ 944.]
   [1045.]
   [ 643.]]

  [[ 472.]
   [ 575.]
   [ 761.]
   [ 867.]
   [ 823.]
   [ 948.]
   [ 748.]]
```

```
[[[[ 704.]
   [ 787.]
   [ 808.]
   [ 818.]
   [ 733.]
   [ 644.]
   [ 459.]]

  [[ 860.]
   [ 951.]
   [ 914.]
   [ 936.]
   [ 993.]
   [ 903.]
   [ 519.]]

  [[ 656.]
   [ 832.]
   [ 839.]
   [ 805.]
   [ 944.]
   [1045.]
   [ 643.]]

  [[ 472.]
   [ 575.]
   [ 761.]
   [ 867.]
   [ 823.]
   [ 948.]
   [ 748.]]
```

```
  [[ 631.]
   [ 587.]
   [ 641.]
   [ 871.]
   [ 726.]
   [ 629.]
   [ 560.]]

  [[ 787.]
   [ 779.]
   [ 613.]
   [ 687.]
   [ 584.]
   [ 436.]
   [ 368.]]

  [[ 489.]
   [ 534.]
   [ 476.]
   [ 430.]
   [ 349.]
   [ 294.]
   [ 214.]]]]
[[[[ 951.]
   [ 914.]
   [ 936.]
   [ 993.]
   [ 903.]]

  [[ 832.]
   [ 839.]
   [ 805.]
   [ 944.]
   [1045.]]
```

```
  [[ 575.]
   [ 761.]
   [ 867.]
   [ 823.]
   [ 948.]]

  [[ 587.]
   [ 641.]
   [ 871.]
   [ 726.]
   [ 629.]]

  [[ 779.]
   [ 613.]
   [ 687.]
   [ 584.]
   [ 436.]]]]
```

Problem 2:

```
'''
Week6 - Problem 2
'''

import numpy as np
from scipy import signal
import matplotlib.pyplot as plt
from PIL import Image


#Read the image
image = Image.open('02Rice.png')
fig, aux = plt.subplots(figsize=(8,8), num='Original figure')
aux.imshow(image, cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Convert to just the Luminance values
image_lum = image.convert('L')

#Get the image as an array
image_arr = np.asarray(image_lum)

#Laplacian filter #1
lap_filter_1 = np.array([[0, 1, 0],
                         [1, -4, 1],
                         [0, 1, 0]])
conv_image_1 = signal.convolve2d(image_arr, lap_filter_1, mode='same',
boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 1')
aux.imshow(np.absolute(conv_image_1), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()
```

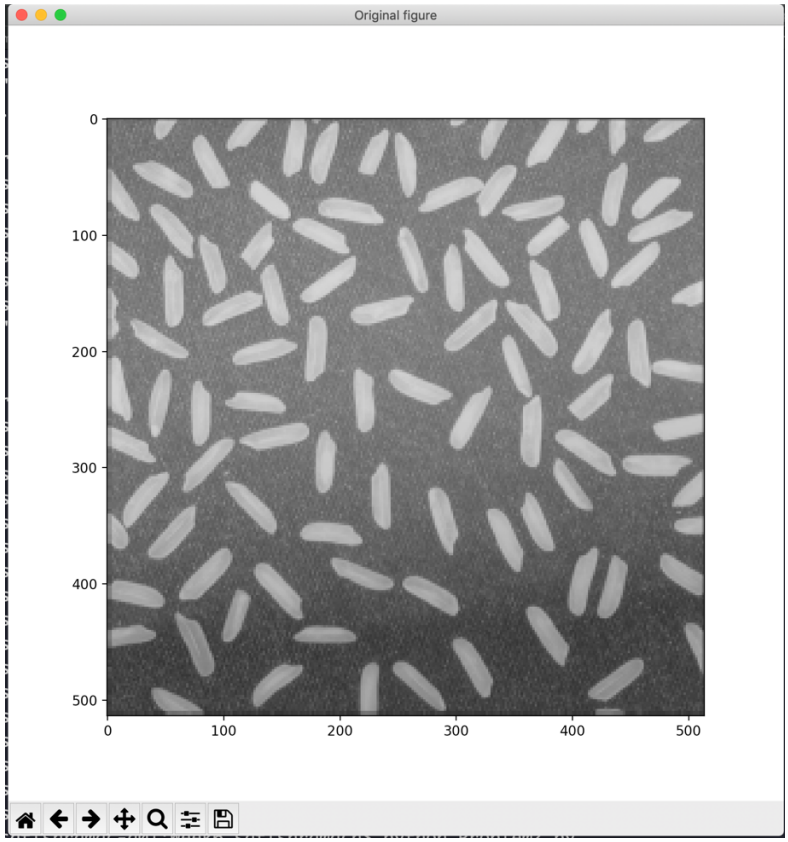
```
#Laplacian filter #2
lap_filter_2 = np.array([[1, 1, 1],
                         [1, -8, 1],
                         [1, 1, 1]])
conv_image_2 = signal.convolve2d(image_arr, lap_filter_2, mode='same',
boundary='symm')
#conv_image_2 = signal.convolve2d(conv_image_1, lap_filter_2, mode='same',
boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 2')
aux.imshow(np.absolute(conv_image_2), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter #3
lap_filter_3 = np.array([[0, 1, 0],
                         [1, -5, 1],
                         [0, 1, 0]])
conv_image_3 = signal.convolve2d(image_arr, lap_filter_3, mode='same',
boundary='symm')
#conv_image_3 = signal.convolve2d(conv_image_2, lap_filter_3, mode='same',
boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 3')
aux.imshow(np.absolute(conv_image_3), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter #4
lap_filter_4 = np.array([[1, 1, 1],
                         [1, -9, 1],
                         [1, 1, 1]])
conv_image_4 = signal.convolve2d(image_arr, lap_filter_4, mode='same',
boundary='symm')
#conv_image_4 = signal.convolve2d(conv_image_3, lap_filter_4, mode='same',
boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 4')
```

```
aux.imshow(np.absolute(conv_image_4), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter #1 and #2
conv_image_5 = signal.convolve2d(image_arr, lap_filter_1 + lap_filter_2,
mode='same', boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 1 + 2')
aux.imshow(np.absolute(conv_image_5), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter #3 and #4
conv_image_6 = signal.convolve2d(image_arr, lap_filter_3 + lap_filter_4,
mode='same', boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 3 + 4')
aux.imshow(np.absolute(conv_image_6), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter #1, #2, #3 and #4
conv_image_7 = signal.convolve2d(image_arr, lap_filter_1 + lap_filter_2 +
lap_filter_3 + lap_filter_4, mode='same', boundary='symm')
fig, aux = plt.subplots(figsize=(8,8), num='After applying filter 1 + 2 + 3 + 4')
aux.imshow(np.absolute(conv_image_7), cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()
```

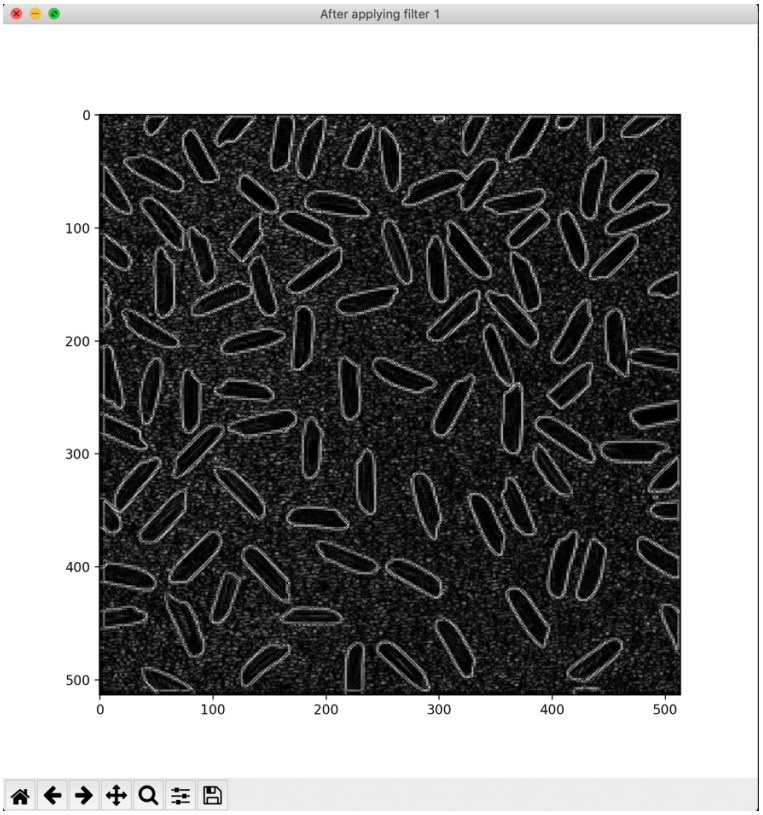
Attaching output applying filters independently, and then Filter1+Filter2, Filter3+Filter4, and then Filter1+Filter2+Filter3+Filter4

The title of the figure says the filter applied.

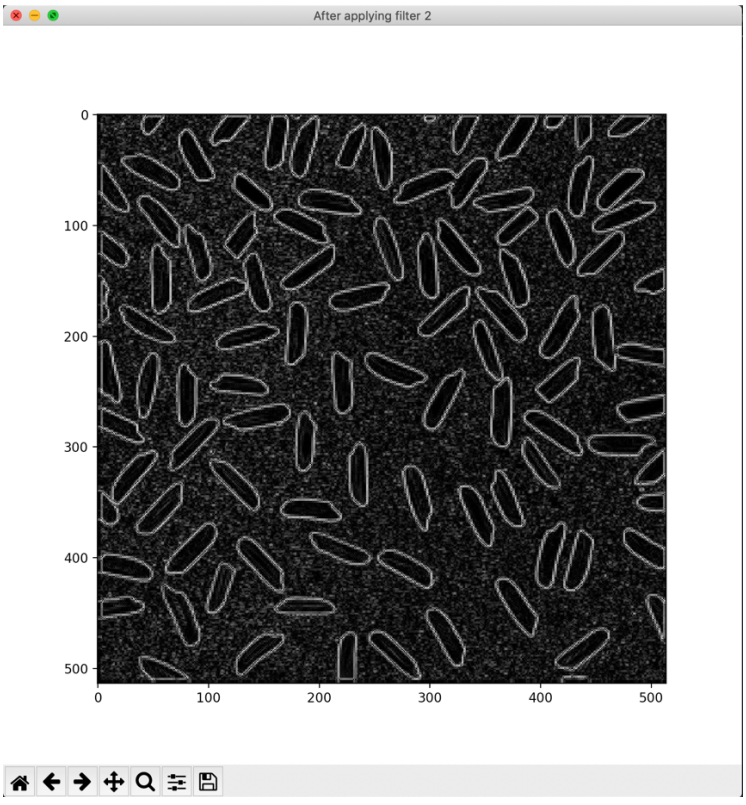
Original image



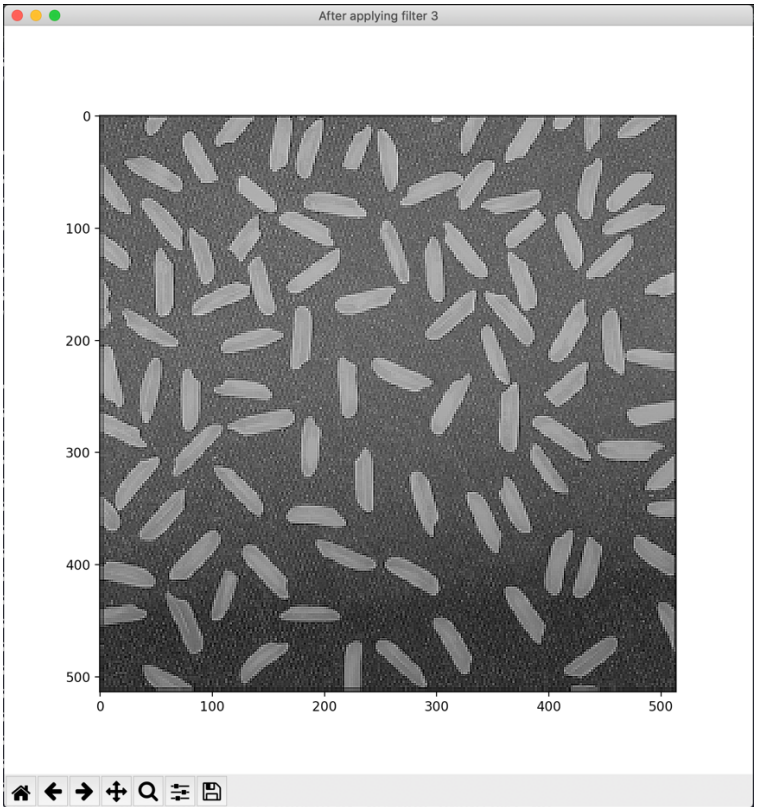
Filter #1



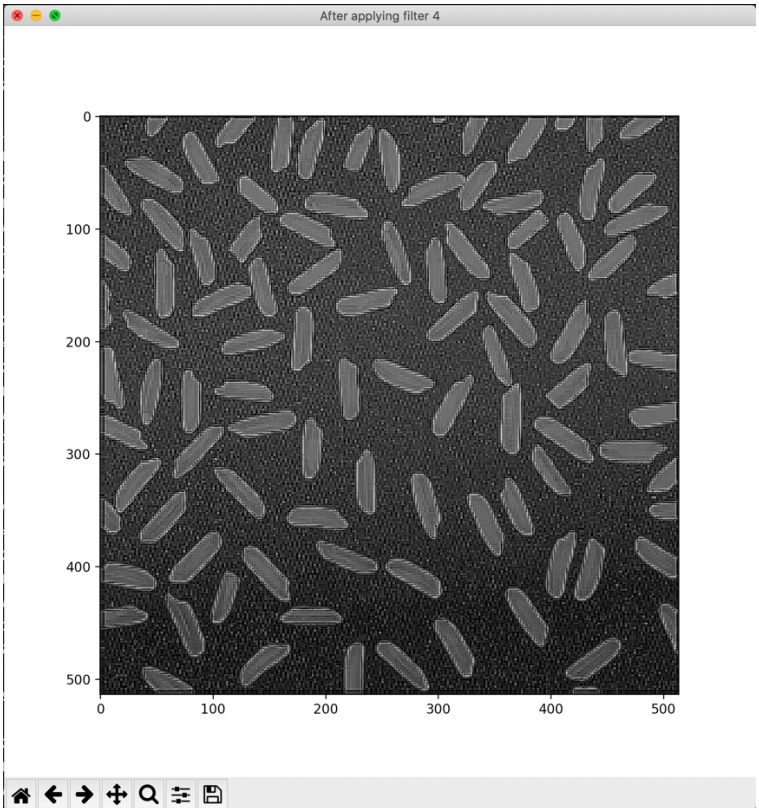
Filter #2



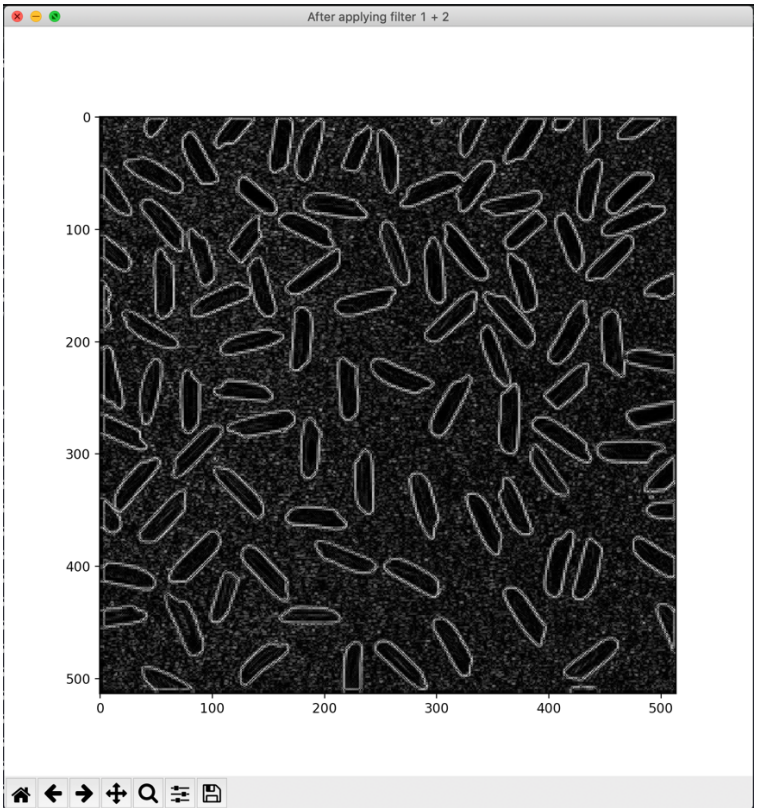
Filter #3



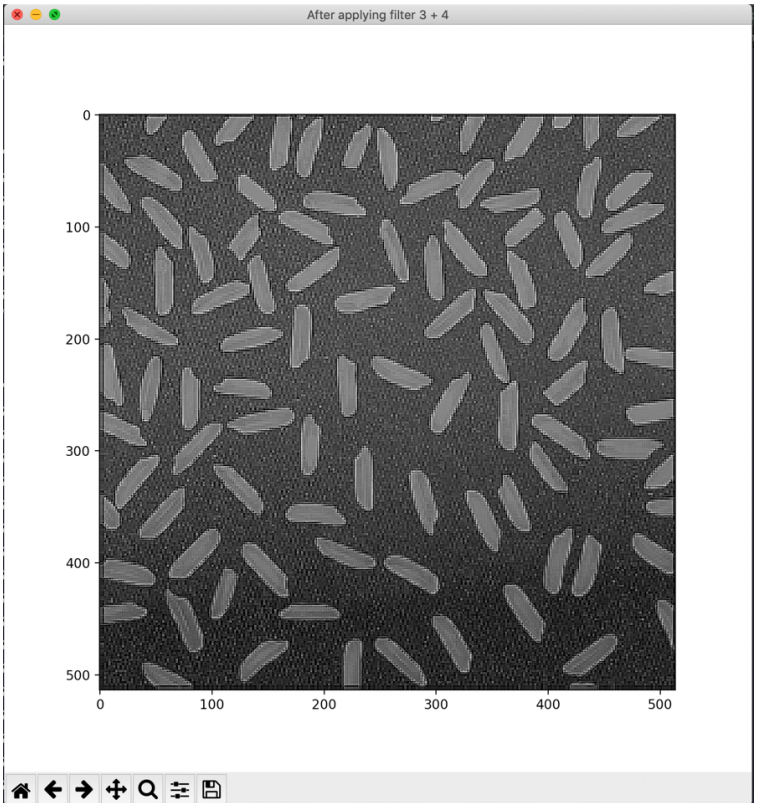
Filter #4



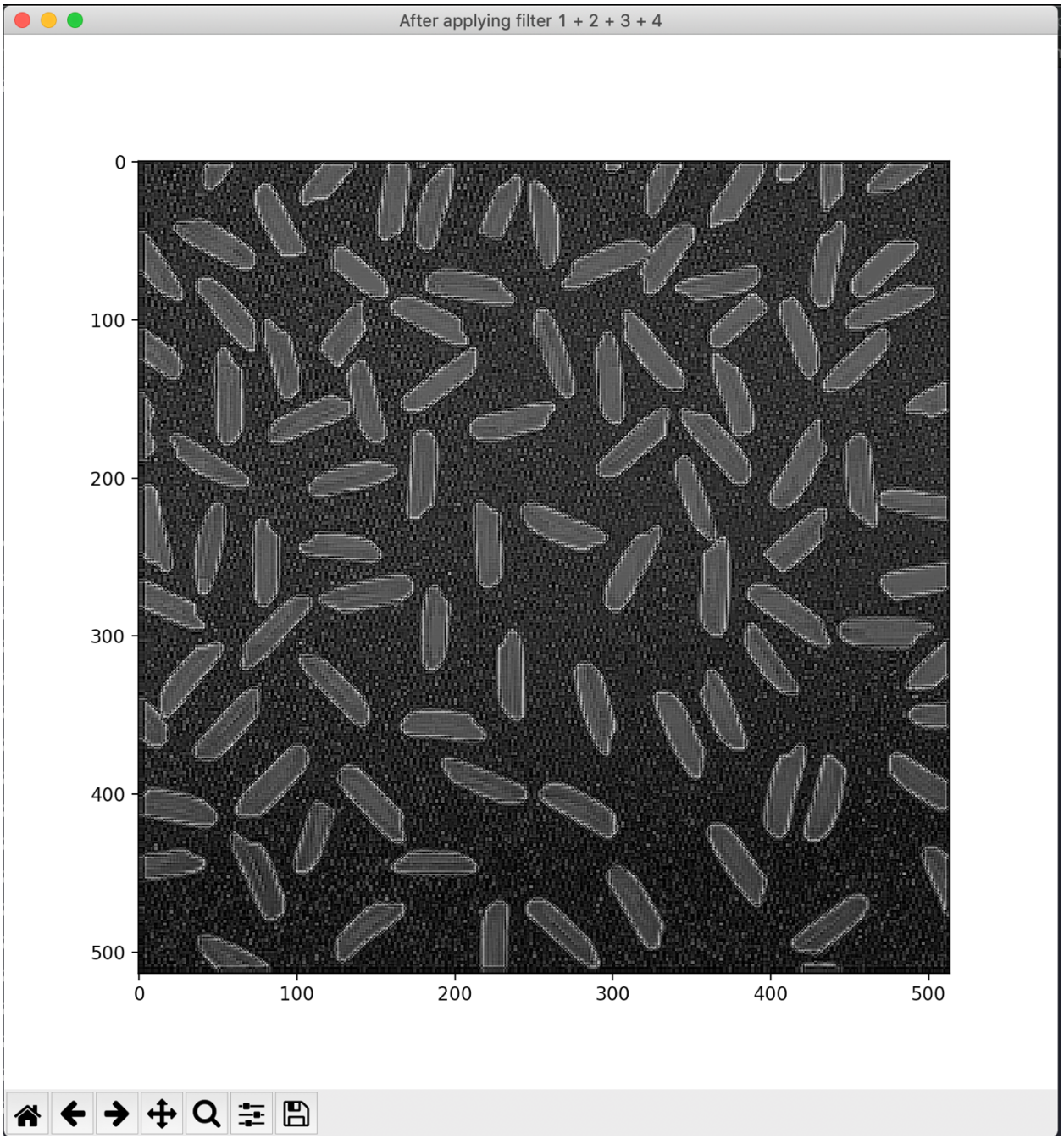
Applying Filter1 + Filter2



Applying Filter3 + Filter4



Applying Filter1 + Filter2 + Filter3 + Filter4

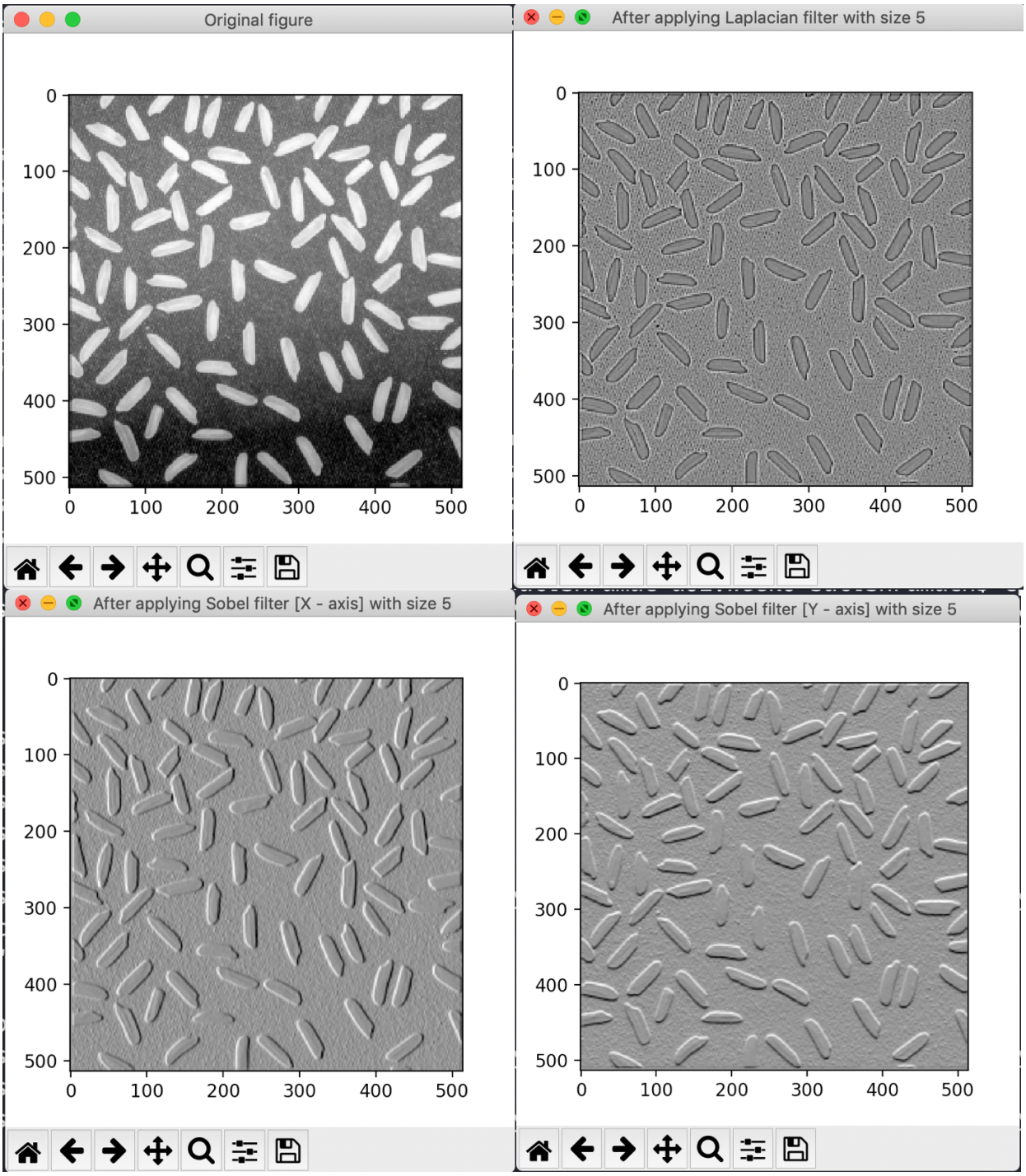


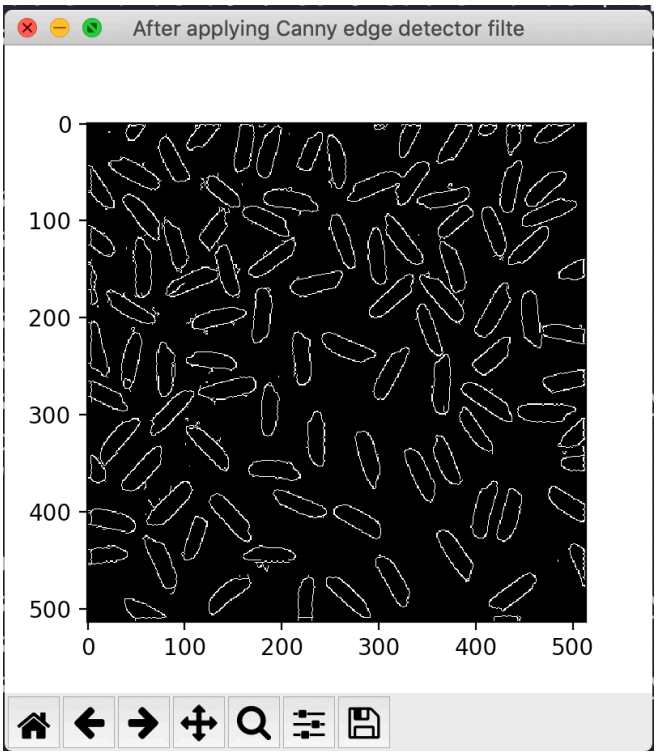
Problem #3

```
'''
Week6 - Problem 3
Applying Laplacian, Sobel and Canny filters on an image
'''

import numpy as np
import matplotlib.pyplot as plt
import cv2


#Read the image
image = cv2.imread('02Rice.png', 0)
fig, aux = plt.subplots(figsize=(4,4), num='Original figure')
aux.imshow(image, cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Laplacian filter
laplacian = cv2.Laplacian(image, cv2.CV_64F, ksize=5)
fig, aux = plt.subplots(figsize=(4,4), num='After applying Laplacian filter with size
5')
aux.imshow(laplacian, cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Sobel-x filter
sobelx = cv2.Sobel(image, cv2.CV_64F, 1, 0, ksize=5)
fig, aux = plt.subplots(figsize=(4,4), num='After applying Sobel filter [X - axis] with
size 5')
aux.imshow(sobelx, cmap='gray')
plt.show()
plt.waitforbuttonpress()
```

```
plt.close()

#Sobel-y filter
sobely = cv2.Sobel(image, cv2.CV_64F, 0, 1, ksize=5)
fig, aux = plt.subplots(figsize=(4,4), num='After applying Sobel filter [Y - axis] with
size 5')
aux.imshow(sobely, cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()

#Canny edge filter
canny = cv2.Canny(image, 50,200)
fig, aux = plt.subplots(figsize=(4,4), num='After applying Canny edge detector
filte')
aux.imshow(canny, cmap='gray')
plt.show()
plt.waitforbuttonpress()
plt.close()
```

The output is captured in the images. The title of the image shows the filter that has been applied.


Original figure
After applying Laplacian filter with size 5
After applying Sobel filter [X - axis] with size 5
After applying Sobel filter [Y - axis] with size 5
0
100
200
300
400
500


After applying Canny edge detector filte
0
100
200
300
400
500
0
100
200
300
400
500