**Practical No: 6**

Page| 13

**Aim:** Write a program to implement linear and nonlinear noise smoothing on suitable image or

Sound signal.

**A:** Smoothing with Linear

Linear smoothing filters remove high-frequency components, and the sharp detail in the image is lost. For example, step changes will be blurred into gradual changes, and the ability to accurately localize a change will be sacri- ficed.

**Program code:**

import pylab

import numpy as np

from PIL import Image, ImageFilter

from skimage import filters

i = 1

pylab.figure(figsize=(10,25))

for prop\_noise in np.linspace(0.05,0.3,3):

im = Image.open(r'C:\Users\LJP\_IT\_LAB\Desktop\Ravi Singh\Image Processing\Docs\nature.jpg')

def plot\_image(im,title=''):pylab.title(title, size=7)

n = int(im.width \* im.height \* prop\_noise)

x, y = np.random.randint(0, im.width, n), np.random.randint(0, im.height, n)

for (x,y) in zip(x,y):

im.putpixel((x, y), ((0,0,0) if np.random.rand() < 0.5 else (255,255,255)))

im.save(r'C:\Users\LJP\_IT\_LAB\Desktop\Ravi Singh\Image Processing\Docs\output' + str(prop\_noise) + '.jpg')

pylab.subplot(6,2,i), plot\_image(im, 'Original Image with ' +

str(int(100\*prop\_noise)) + '% added noise')

pylab.imshow(im)

i += 1

im1 = im.filter(ImageFilter.BLUR)

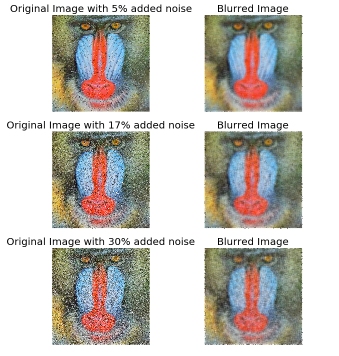
pylab.subplot(6,2,i), plot\_image(im1, 'Blurred Image')

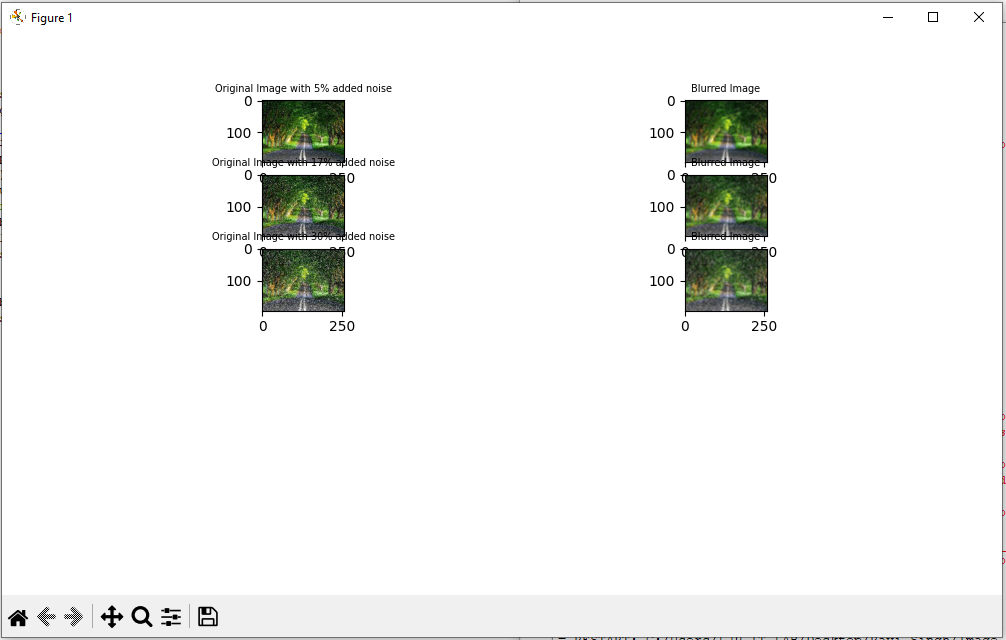
pylab.imshow(im1)

i += 1

pylab.show()

**Output:**





**B:** Smoothing with Non-linear

Page| 14

Non-linear smoothing filters are a powerful weapon that should be in the arsenal of every image processing practitioner. They find application in several fields, from computer vision to astronomy, medical imaging, geology, digital art, photography and many more.

**Program Code:**

import pylab

import numpy as np

from PIL import Image, ImageFilter

from skimage import filters

i = 1

pylab.figure(figsize=(25,35))

for prop\_noise in np.linspace(0.05,0.3,3):

im = Image.open(r'C:\Users\LJP\_IT\_LAB\Desktop\Ravi Singh\Image Processing\Docs\nature.jpg')

def plot\_image(im,title=''):pylab.title(title, size=7)

n = int(im.width \* im.height \* prop\_noise)

x, y = np.random.randint(0, im.width, n), np.random.randint(0, im.height, n)

for (x,y) in zip(x,y):

im.putpixel((x, y), ((0,0,0) if np.random.rand() < 0.5 else (255,255,255)))

im.save(r'C:\Users\LJP\_IT\_LAB\Desktop\Ravi Singh\Image Processing\Docs\output' + str(prop\_noise) + '.jpg')

pylab.subplot(6,4,i)

pylab.imshow(im)

plot\_image(im, 'Original Image with ' + str(int(100\*prop\_noise)) + '%added noise')

i += 1

for sz in [3,7,11]:

im1 = im.filter(ImageFilter.MedianFilter(size=sz))

pylab.subplot(6,4,i), plot\_image(im1, 'Output (Median Filter size=' + str(sz) + ')')

pylab.imshow(im1)

i += 1

pylab.show()

**Output:**

