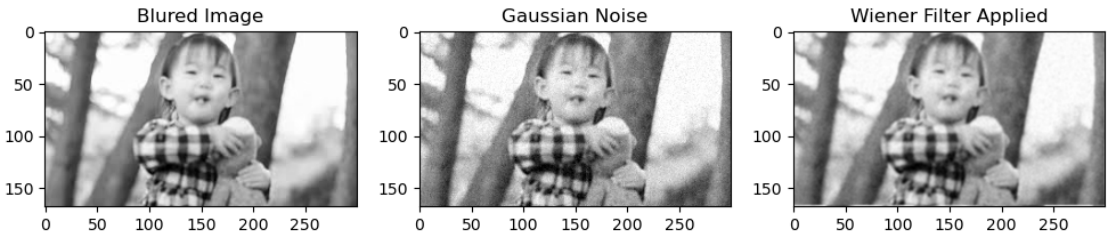
c2011050-195150mer-yurtalan-462hw4

April 17, 2024

[9]:
```
import os
import numpy as np
from numpy.fft import fft2, ifft2
from scipy.signal import gaussian
import matplotlib.pyplot as plt

def addGaussianNoise(img,sigma):
    gauss=np.random.normal(0,sigma,img.shape)
    noisy=img+gauss

    noisy[noisy<0]=0
    noisy[noisy>255]=255

    return noisy

def wienerFilter(img,kernel,K):
    kernel/=np.sum(kernel)
    temp=np.copy(img)
    temp=fft2(temp)

    kernel=fft2(kernel, s=img.shape)
    kernel=np.conj(kernel)/(np.abs(kernel)**2+K)

    temp=temp*kernel
    temp=np.abs(ifft2(temp))

    return temp

def gaussianKernel(kernelSize=3):
    a=gaussian(kernelSize,kernelSize/3).reshape(kernelSize,1)
    a=np.dot(a,a.transpose())
    a/=np.sum(a)

    return a

def rgb2gray(rgb):
```

```
    return np.dot(rgb[..., :3], [0.2989, 0.5870, 0.1140])

image=os.path.join('C:/Users/dogan/Pictures/Screenshots/ö.jpeg')
img=rgb2gray(plt.imread(image)) # convert gray
noisy_img=addGaussianNoise(img,sigma=10) # gauss noise function calling,␣
 ↪sigma(for gauss noise rate)
kernel=gaussianKernel(3) # create kernel (for wiener filter)

filtered_img=wienerFilter(noisy_img,kernel,K=30) # wiener filter function␣
 ↪calling

display=[img, noisy_img, filtered_img] # Display images
label=['Blured Image','Gaussian Noise','Wiener Filter Applied']  # photo names

fig=plt.figure(figsize=(12, 10))

for i in range(len(display)):                #
    fig.add_subplot(1, 3, i+1)               #
    plt.imshow(display[i],cmap='gray')      #  for display photo
    plt.title(label[i])                      #

plt.show()
```



[ ]:

462 Homework 4

1.Introduction:

Motion blur is a problem caused by the quick capture or panning movement when taking a photograph. In this report, we will endeavor to overcome this challenge.

2. Methodology:

I have developed a method to address the issue of blurred images. This method incorporates Gaussian noise addition and Wiener filtering techniques. Below, you will find a general description and mathematical formulations:

General Description of the Method:

A blurred image is first added. Then, noise is introduced. Following this, Wiener filtering is applied to refine the image. Finally, a clean image is produced.

Pseudo-code

Take input image: img

Add Gaussian noise:

noisy_img = add_gaussian_noise(img, sigma)

Create Gaussian kernel for Wiener filtering:

kernel = gaussian_kernel(kernel_size)

Apply Wiener filter:

filtered_img = wiener_filter(noisy_img, kernel, K)

Return the cleaned image: filtered_img

3. Implementation:

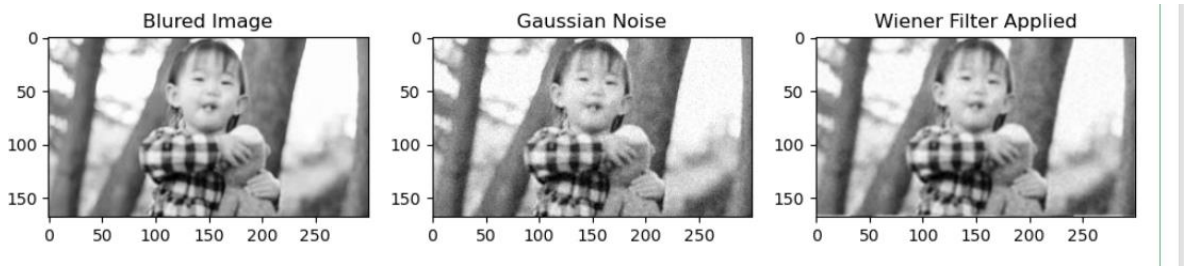
Blurry image is obtained by applying a blurring filter to the original image. Then, Gaussian noise is added to the blurred image. Finally, the Wiener filter is applied to filter out the noise and restore the image.

addGaussianNoise function: Adds noise to generate a noisy image.

WienerFilter function: Applies the Wiener filter.

gaussianKernel function: Generates a Gaussian kernel.

rgb2gray function: Converts a color image to grayscale.



4. Results:

Although the method may not perfectly restore the image to its original state, it allows us to achieve a value close to it. In fact, areas that appear to be in motion tend to look more stable, which is visually pleasing. This could be a reason why this method is worth applying.

5 . References:

https://scikit-image.org/docs/stable/auto_examples/filters/plot_deconvolution.html

Deconvolution | Image Processing II (youtube.com)

https://www.youtube.com/watch?v=mBpaz5bk0XI