assignment11

December 13, 2018

1 This is assignment11

1.1 Name : Jaehyun Lim

1.2 Student ID : 20145450

1.3 Obtaining u to minimize by lamda

$$\parallel f - u \parallel_2^2 + \lambda(\parallel \frac{\mathrm{d}u}{\mathrm{d}x} \parallel^2 + \parallel \frac{\mathrm{d}u}{\mathrm{d}y} \parallel^2)$$

1.4 import packages

```
In [14]: import numpy as np
         import matplotlib.pyplot as plt
         %matplotlib inline
         from scipy import signal
         from skimage import io, color
         from skimage import exposure
         from skimage.io import imread
         from skimage.color import rgb2gray
```

1.5 Denoise function

```
In [15]: def denoise(img, lamda=0.1, eps=1e-3, num_iter_max=200):
             u = np.zeros_like(img)
             px = np.zeros_like(img)
             py = np.zeros_like(img)

             nm = np.prod(img.shape[:2])
             tau = 0.125

             i = 0
             while i < num_iter_max:
                 u_old = u

                 # x and y components of u's gradient
                 ux = np.roll(u, -1, axis=1) - u
```

```
        uy = np.roll(u, -1, axis=0) - u

        # update the dual variable
        px_new = px + (tau / lamda) * ux
        py_new = py + (tau / lamda) * uy
        norm_new = np.maximum(1, np.sqrt(px_new **2 + py_new ** 2))
        px = px_new / norm_new
        py = py_new / norm_new

        # calculate divergence
        rx = np.roll(px, 1, axis=1)
        ry = np.roll(py, 1, axis=0)
        div_p = (px - rx) + (py - ry)

        # update image
        u = img + lamda * div_p

        # calculate error
        error = np.linalg.norm(u - u_old) / np.sqrt(nm)

        if i == 0:
            err_init = error
            err_prev = error
        else:
            # break if error small enough
            if np.abs(err_prev - error) < eps * err_init:
                break
            else:
                e_prev = error

        # don't forget to update iterator
        i += 1

    return u
```

1.6 Load image and Convert to grayscale

```
In [16]: def rgb2gray(img):
             grayImage = np.zeros(img.shape)
             R = np.array(img[:, :, 0])
             G = np.array(img[:, :, 1])
             B = np.array(img[:, :, 2])

             R = (R *.299)
             G = (G *.587)
             B = (B *.114)

             Avg = (R+G+B)
```

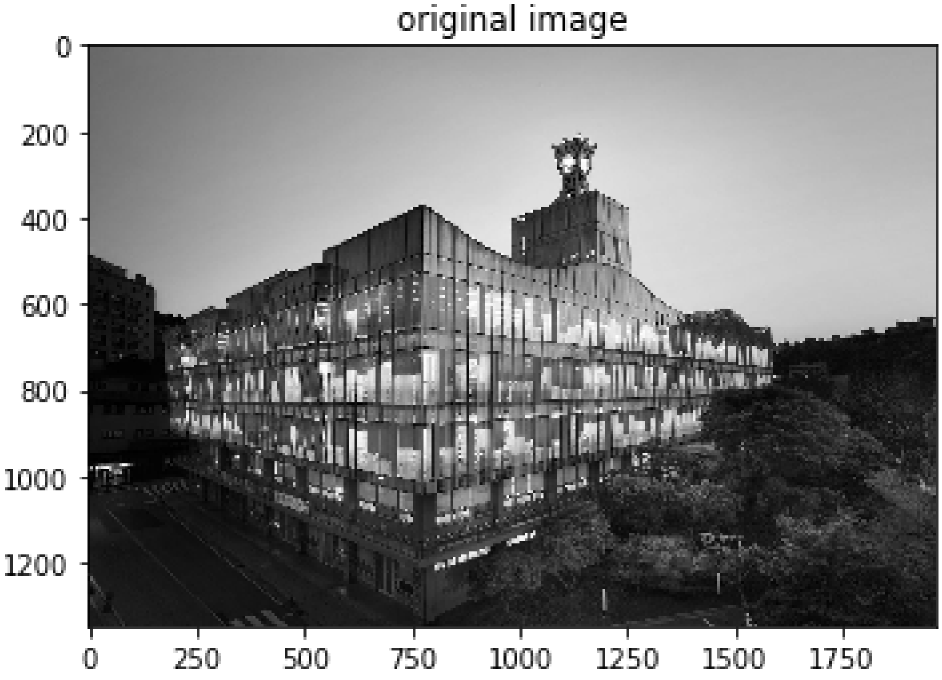
```
        grayImage = img

        for i in range(3):
           grayImage[:,:,i] = Avg

        return grayImage

file_image        = 'cau.jpg'
im_color         = io.imread(file_image)
img2          = rgb2gray(im_color)
plt.title('original image')
plt.imshow(img2.astype(np.uint8))
```

```
Out[16]: <matplotlib.image.AxesImage at 0x1c14c6f5f8>
```

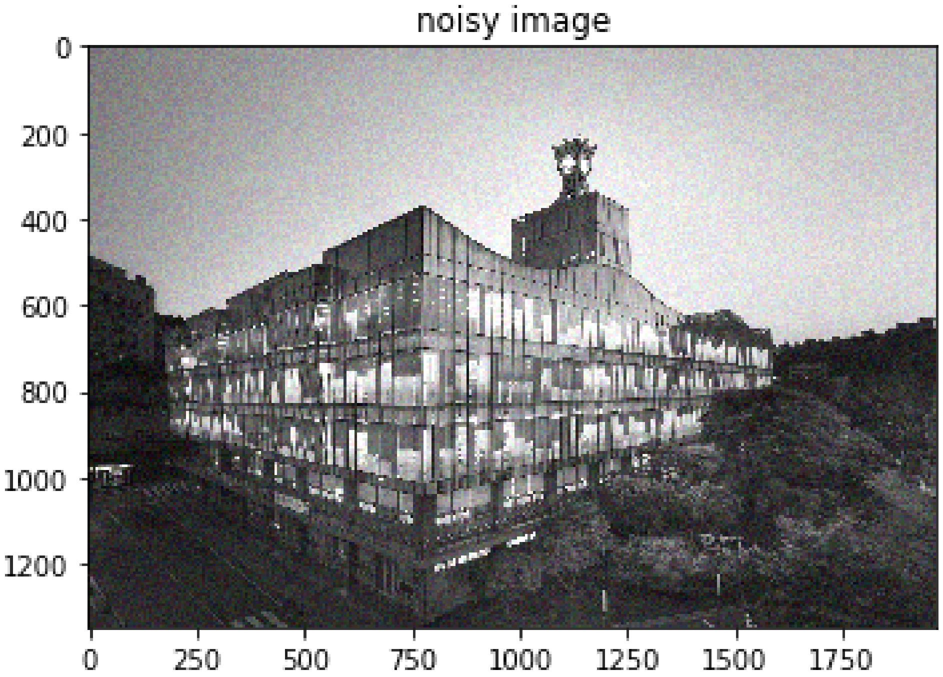


```
In [17]: ## noisy image
```

```
In [18]: noisy = img2 + 0.5 * img2.std() * np.random.random(img2.shape)
         noisy = np.clip(noisy, 0, 255)

         plt.title('noisy image')
         plt.imshow(noisy.astype(np.uint8))
```

```
Out[18]: <matplotlib.image.AxesImage at 0x1c152f8438>
```

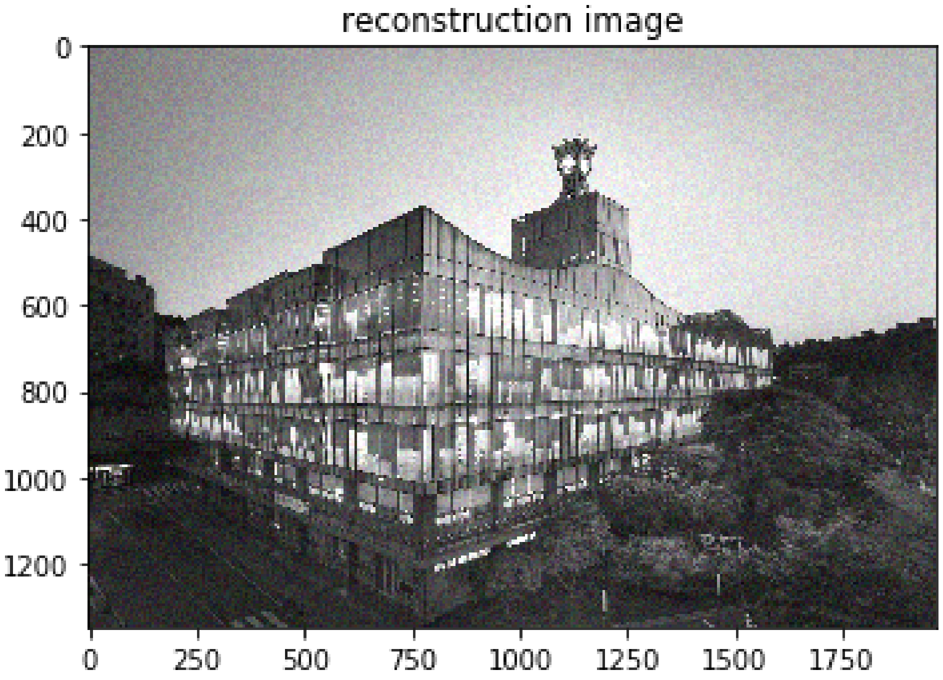


1.7 reconstruction image with varying regularization parameter

$$\lambda = 2^{-3}, 2^{-1}, 2^{0}, 2^{3}, 2^{7}$$

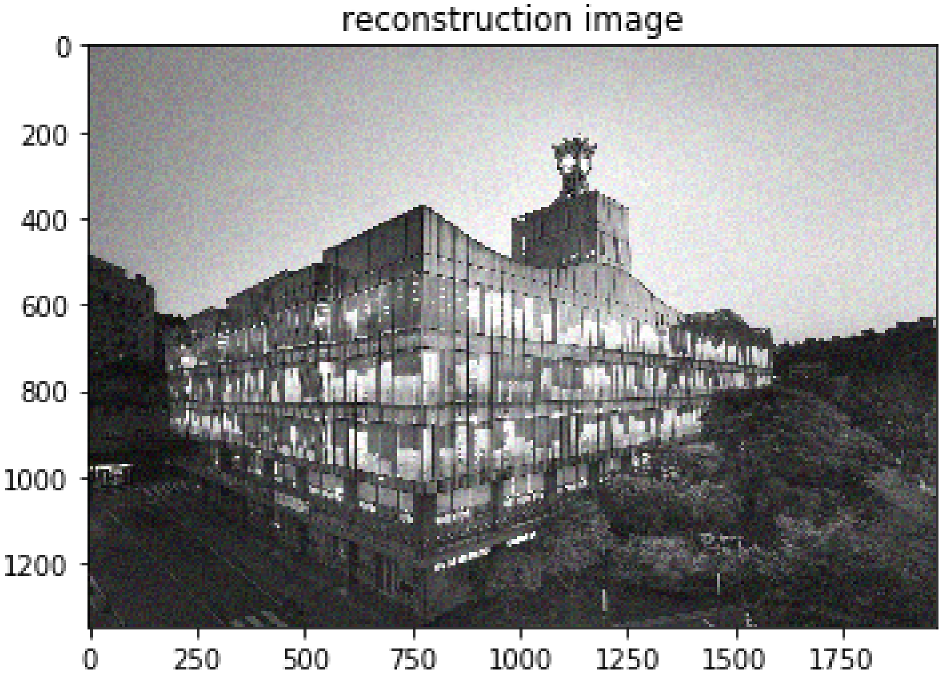
```
In [19]: plt.title('reconstruction image')
         plt.imshow(denoise(noisy, lamda=1/8).astype(np.uint8), cmap='viridis')
```

```
Out[19]: <matplotlib.image.AxesImage at 0x1c1546d0f0>
```



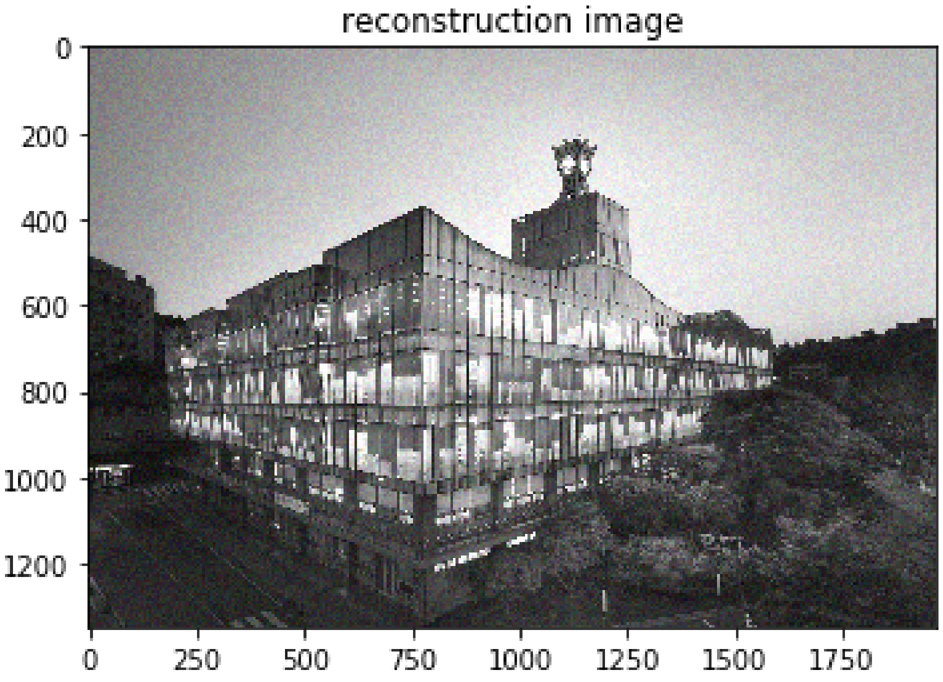
```
In [20]: plt.title('reconstruction image')
         plt.imshow(denoise(noisy, lamda=1/2).astype(np.uint8), cmap='viridis')

Out[20]: <matplotlib.image.AxesImage at 0x1c14d9ba20>
```



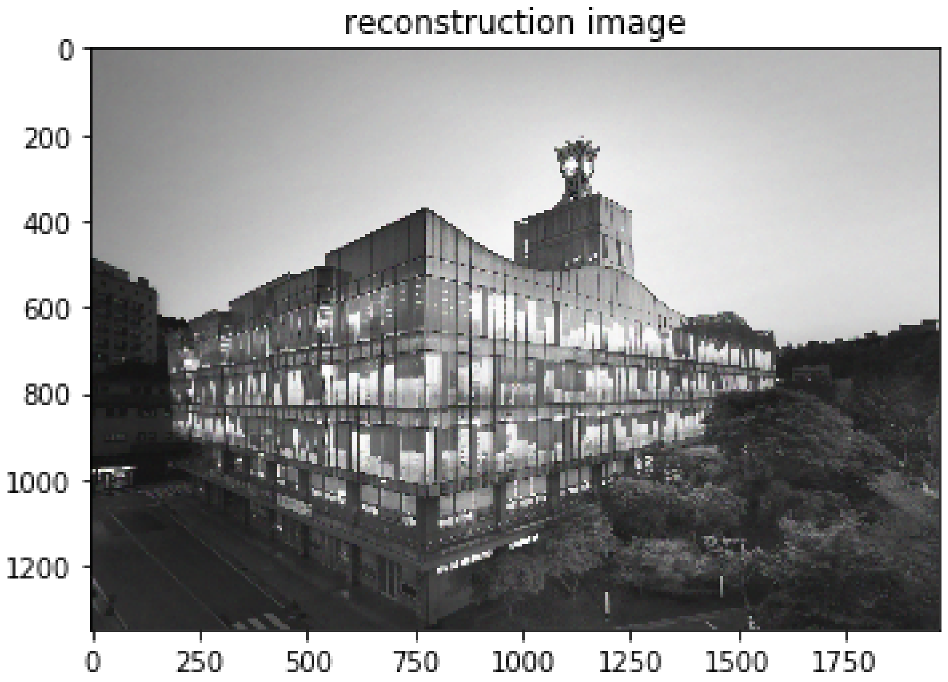
```
In [21]: plt.title('reconstruction image')
         plt.imshow(denoise(noisy, lamda=1).astype(np.uint8), cmap='viridis')

Out[21]: <matplotlib.image.AxesImage at 0x1c14e62710>
```



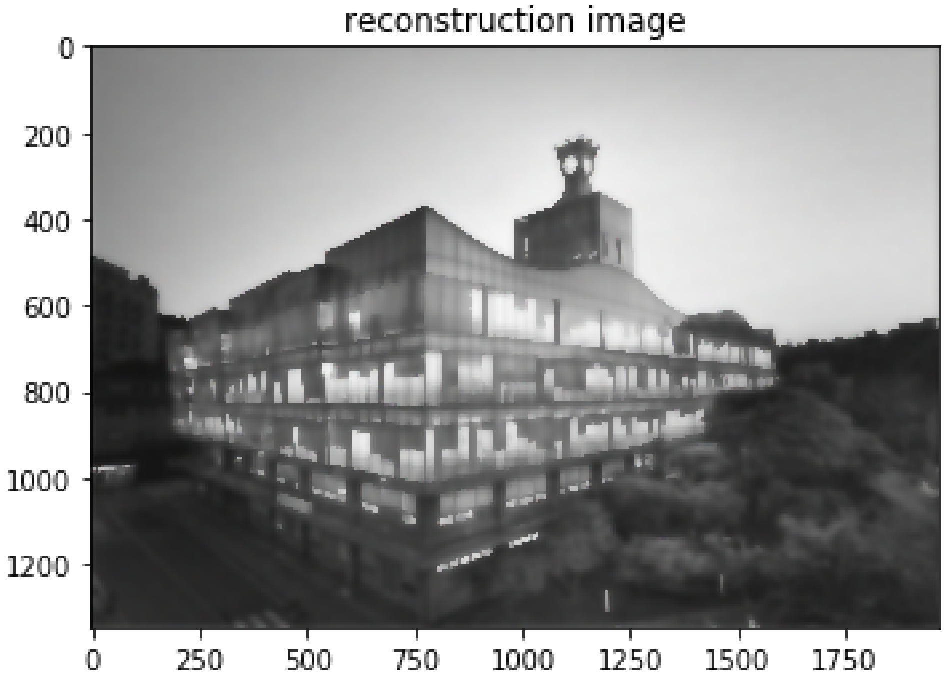
```
In [22]: plt.title('reconstruction image')
         plt.imshow(denoise(noisy, lamda=8).astype(np.uint8), cmap='viridis')

Out[22]: <matplotlib.image.AxesImage at 0x1c14ddf6a0>
```



```
In [23]: plt.title('reconstruction image')
         plt.imshow(denoise(noisy, lamda=128).astype(np.uint8), cmap='viridis')

Out[23]: <matplotlib.image.AxesImage at 0x1c14f44630>
```



1.8 The link to the github

https://github.com/JaeHyunLim/assignment.git