Image Processing HW01 - Spatial Image Enhancement CSIE 4B 408410120 鍾博丞

Date Information

- Due: 2022.11.08
- Last Modified: 2022.11.08

Environment Requirement

- python 3.0 or newer - for f-strings `f"Something {variable}"`.
- Another requirements are written in `requirements.txt`, just type `pip install -r requirements.txt` in the terminal.
    - matplotlib
    - numpy
    - opencv-python == 4.5.5.62 (for auto-complete working on pycharm)
        - https://stackoverflow.com/questions/73174194/opencv-autocomplete-not-working-on-pycharm

Execution

The main python code is `main.py`, type the following command and then you can run the program.

```
python main.py
```

There are some parameters in `main.py`.

Note that the program will always save all images in `Img`.

Warning: The program will always delete existing folder `Img` and recreate it.

- `is_show`: Tell the program if you want to show the result on the screen or not.
    - `True`: Show the result on the screen.
    - `False`: Don't show. Save it only.
- `save_eps`: Tell the program if you want to save the *histograms* by vector (.eps) or bitmap (.png) images.
    - `True`: Vector images (.eps)
    - `False`: Bitmap images (.png), default resolution is 600 dpi.
- `dpi`: The resolution for plotting histogram, default is 600 dpi.

All the methods is implemented in `spatial_image_enhancement.py` and import in main as `sie`.

Technical Description

Histogram

We set the bins to 256 because the grayscale intensity is [0, 255].

Gamma Correction

The power-law transformations, or Gamma correction, have the basic form:

$$s = cr^{\gamma}, \text{ where } c \text{ and } \gamma \text{ are positive constants.}$$

Note that *s* and *r* are normalized to [0, 1].

Therefore, we first divide all value by 255, then do the gamma correction, and then transfer back to [0, 255] by multiplying all value by 255.

Histogram Equalization

1. Count the intensity value for every pixel by using `np.bincount`.
2. Normalize the intensity value to get the probability mass function.
3. Calculate the cumulative distribution function by using `np.cumsum`.
4. Create the mapping table by the CDF transfer back to [0, 255].
5. Map all pixels to create the new image.

2D Convolution

First of all, we have to apply the **cross correlation** to the kernel, which is **flipping the kernel matrix horizontally and vertically**.

Then, we need to know the final size of image.

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

, where

- n_{in} is the number of input feature.
- n_{out} is the number of output feature.
- k is the kernel size.
- p is the padding size. We perform zero-padding here.
- s is the stride step.

Next, we replace the inner portion of the padded image with the actual image

```
imagePadded[int(padding):int(-1 * padding), int(padding):int(-1 * padding)] = image
```

If `padding = 1`, then the effect will be like this.

```
0 0 0 0 0 0 0 0
0 I I I I I I 0
0 I I I I I I 0
0 I I I I I I 0
0 I I I I I I 0
0 I I I I I I 0
0 I I I I I I 0
0 0 0 0 0 0 0 0
```

Finally, we iterate through the entire image, Let A be (x, y) and the filter size is 3 * 3, then

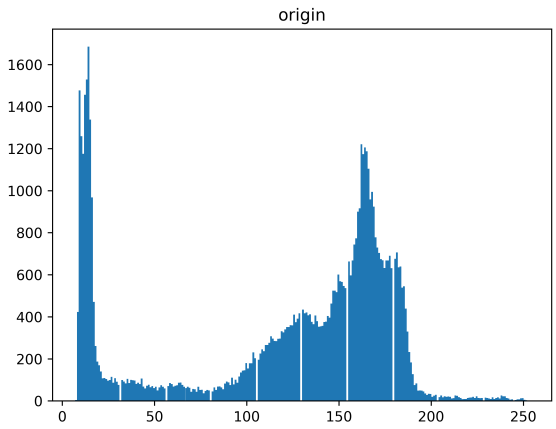
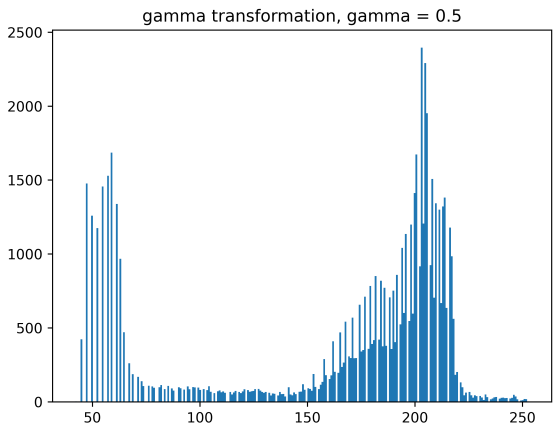
```
A - + 0 0 0 0 0       0 0 0 0 0 0 0 0 0
| x | I I I I 0       0 I I I I I I 0
+ - + I I I I 0       0 I I I I I I 0
0 I I I I I I 0  -->  0 I I I I I I 0
0 I I I I I I 0       0 I I I I I I 0
0 I I I I I I 0       0 I I I A - +
0 I I I I I I 0       0 I I I I | x |
0 0 0 0 0 0 0 0       0 0 0 0 0 + - +
```

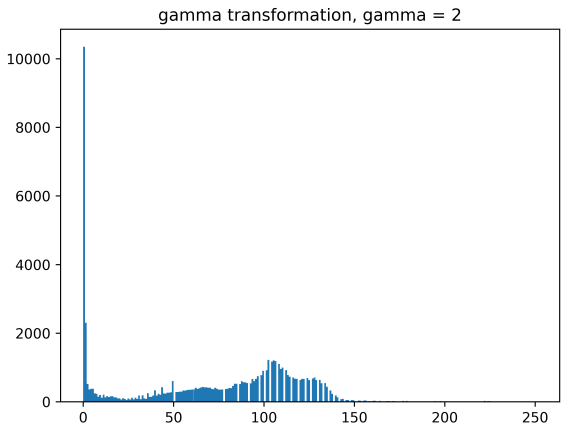
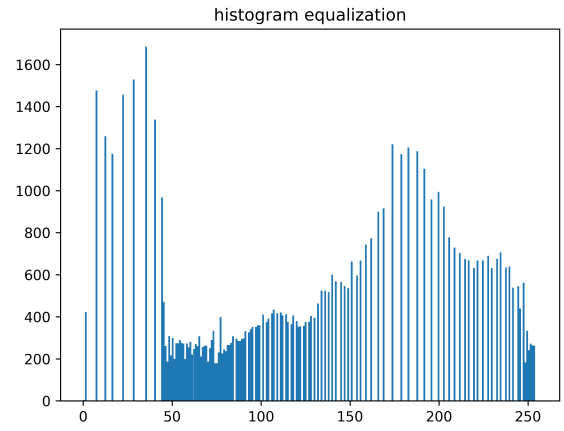
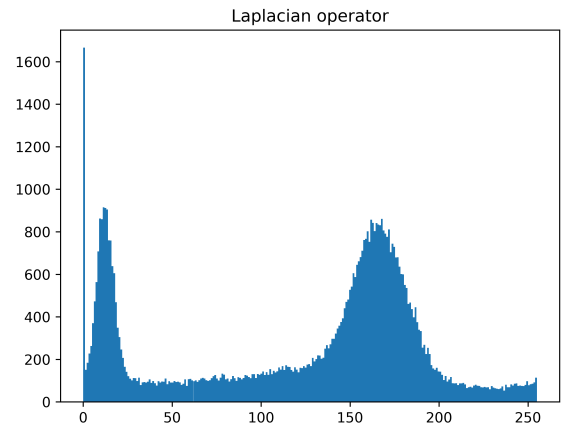
We only do convolution for specified *strides*

```
if x % strides == 0 and y % strides == 0
```

Experimental results

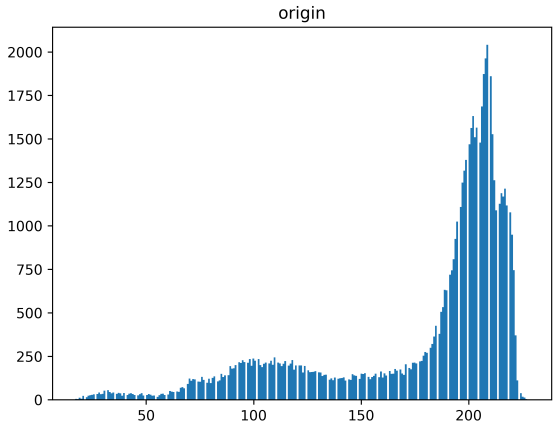
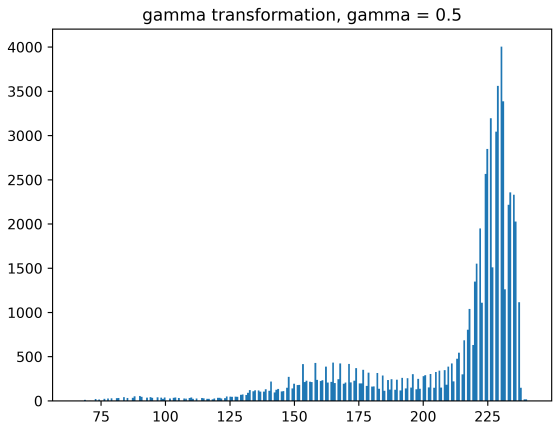
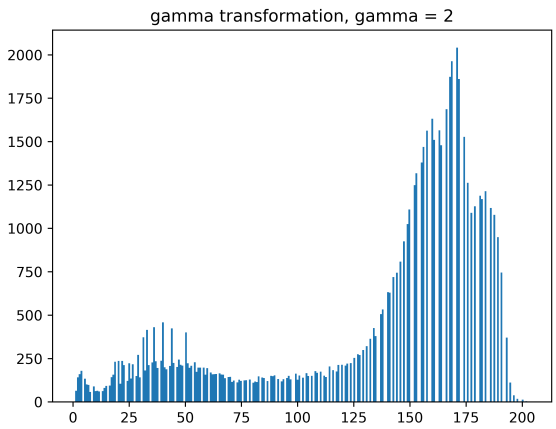
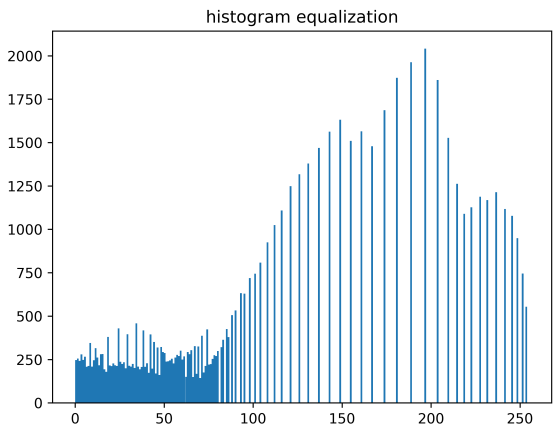
Cameraman

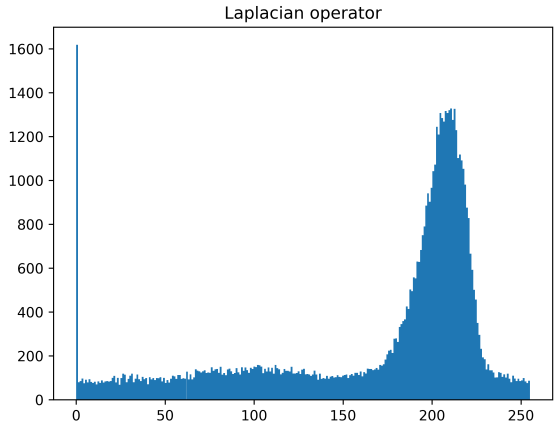
Method	Image	Histogram
original		
gamma = 0.5		

Method	Image	Histogram
gamma = 2		
histeq		
Laplacian		

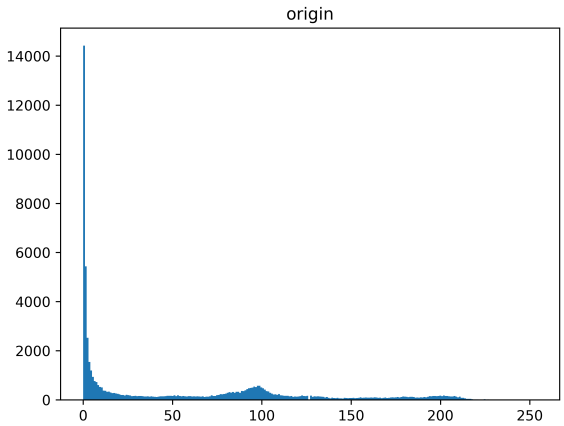
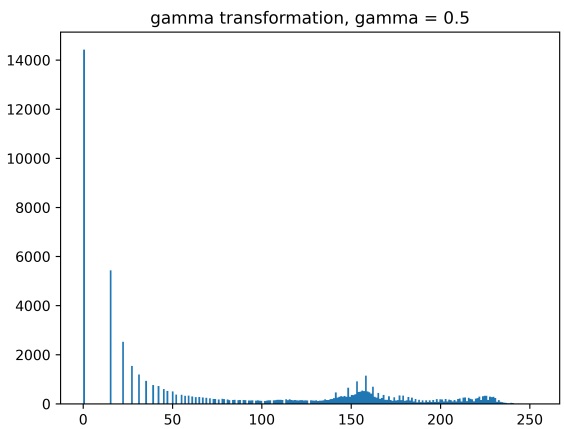
Jetplane

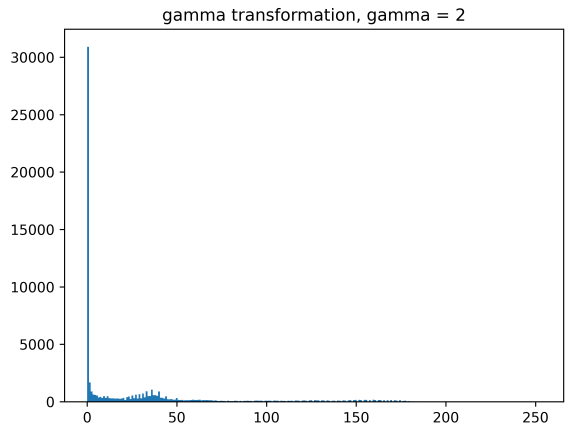
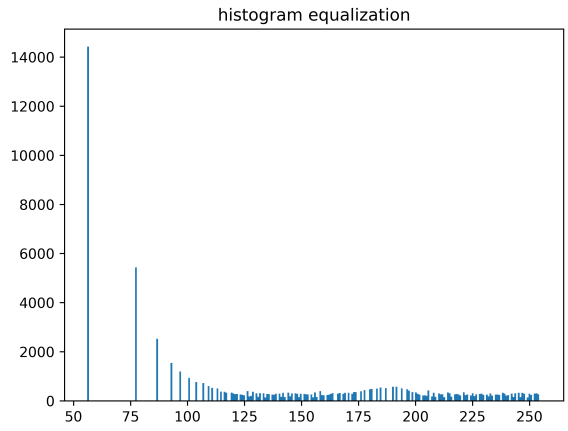
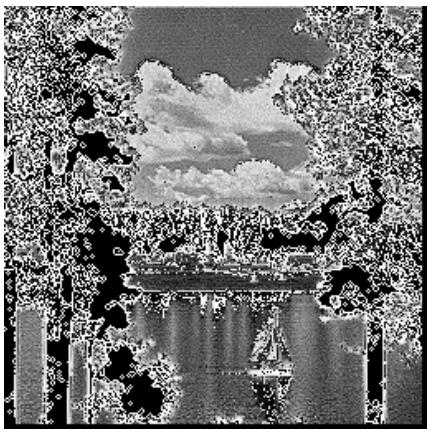
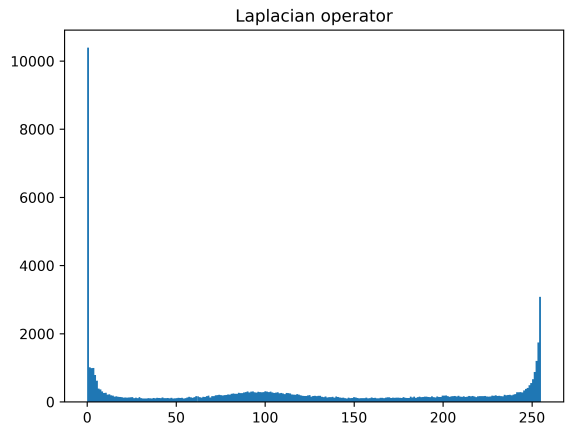
Method	Image	Histogram

Method	Image	Histogram
original		
gamma = 0.5		
gamma = 2		
histeq		

Method	Image	Histogram
Laplacian		

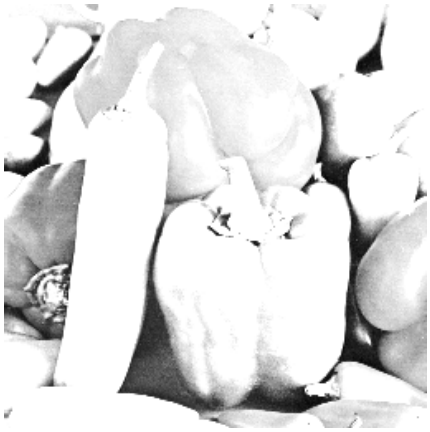
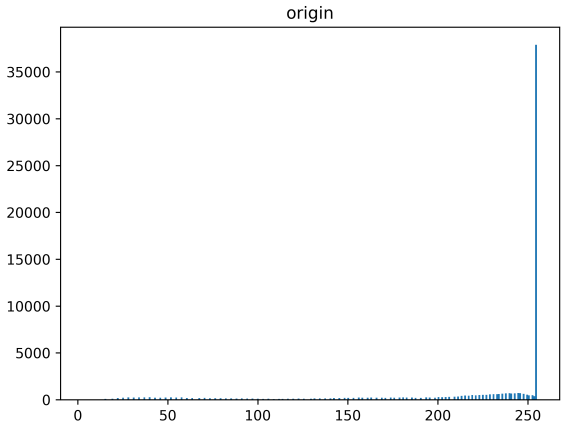
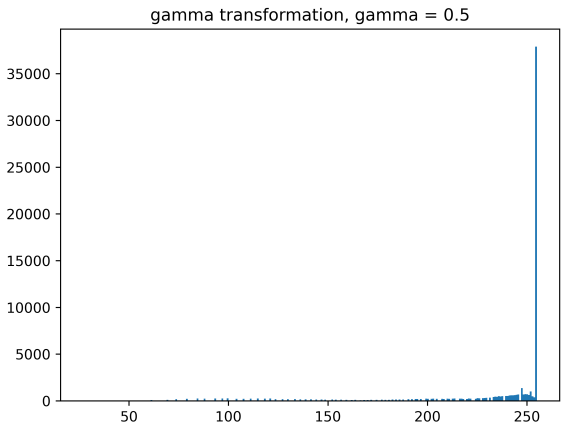
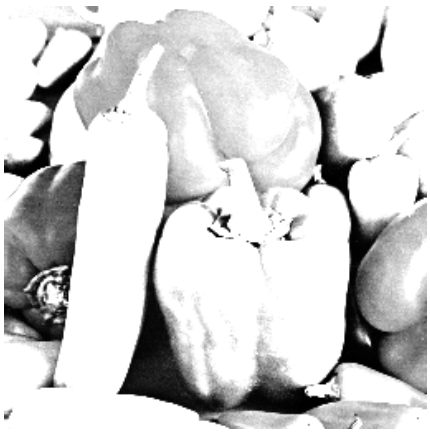
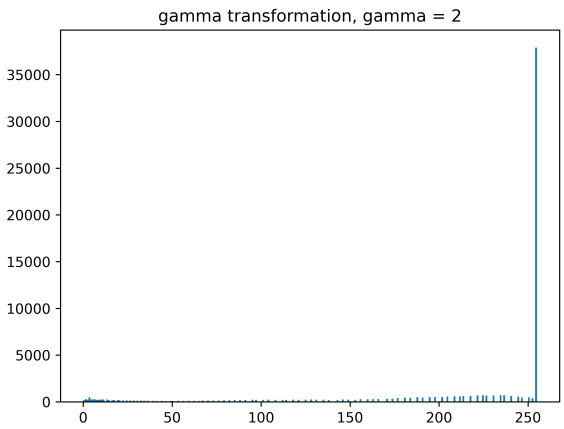
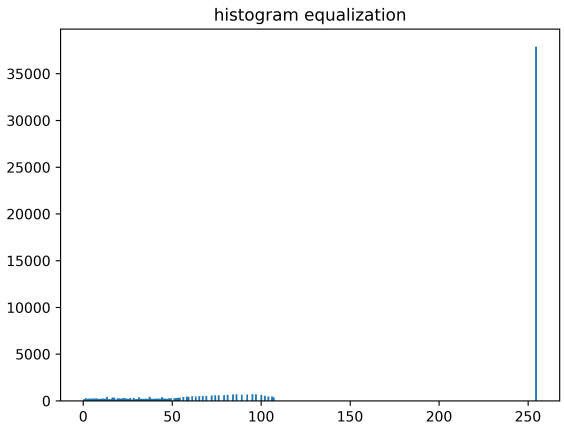
Lake

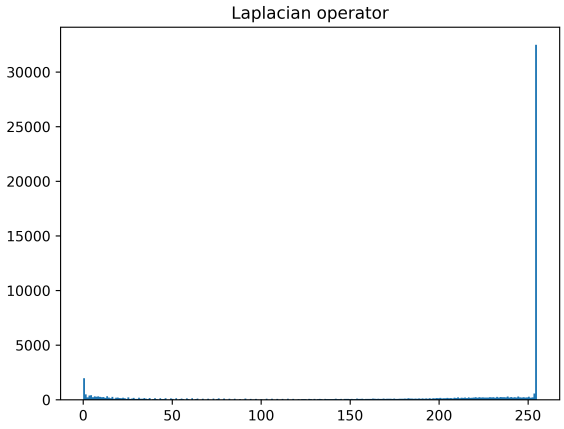
Method	Image	Histogram
original		
gamma = 0.5		

Method	Image	Histogram
gamma = 2		
histeq		
Laplacian		

Peppers

Method	Image	Histogram

Method	Image	Histogram
original		
gamma = 0.5		
gamma = 2		
histeq		

Method	Image	Histogram
Laplacian		

Discussions

Take Peppers for example, the picture have many white pixels and thus the histogram equalization performs worse than other pictures.

The Laplacian operator tends to detect too many edge points and thus the performance is worse.

Reference

- https://geek-docs.com/opencv/opencv-examples/gamma-correction.html
- https://levelup.gitconnected.com/introduction-to-histogram-equalization-for-digital-image-enhancement-420696db9e43
- https://medium.com/analytics-vidhya/2d-convolution-using-python-numpy-43442ff5f381