Introduction to Image Processing – Lab02 Report

107062230 張力仁

Proj03-01: Image Enhancement Using Intensity Transformations

(1) **Log Transformation:**

Step1. $input = input \times 255$ (rescale from [0, 1] to [0, 255])

Step2. $output = c \times \log(1 + input)$

Step3. $output = output \div (c * \log 256)$ (rescale from [0, 255] to [0, 1])

Power-law Transformation:

Step1. $input = input \times 255$ (rescale from [0, 1] to [0, 255])

Step2. $output = c \times (input\, .\hat{}\, \gamma)$ (element-wise exponential)

Step3. $output = output \div (c\ *\ 255\hat{}\gamma)$ (rescale from [0, 255] to [0, 1])

(2) Results

Fig3.8 (a)

Log Transformation

Power-law Transformation:

$\gamma = 0.3$

$\gamma = 0.4$

$\gamma = 0.6$

$\gamma = 3.0$

$\gamma = 5.0$

(3) Analysis of the major difference between them:

Log transformation expands values of dark pixels and compress high-level values. Power-law transformation with $\gamma < 1$ maps a narrow range of dark values into a wider range. When $\gamma > 1$, power-law transformation maps a narrow range of bright values into a wider range.

Proj03-02: Histogram Equalization

(1) **imageHist:**

Use a nested for-loop to calculate the histogram of an image.

```
for i = 1 : size(input, 1)
    for j = 1 : size(input, 2)
        histVector(input(i, j) + 1) = histVector(input(i, j) + 1) + 1
```

histEqualization:

Step1. Obtain a row vector containing the histogram of an image using *imageHist*.

Step2. Use a for-loop to calculate a row vector of transformation function.

```
for i = 1 : 256
    if i == 1
        T(i) = 255 / (size(input, 1) * size(input, 2)) * histVector(i)
    else
        T(i) = T(i - 1) + 255 / (size(input, 1) * size(input, 2)) * histVector(i)
```

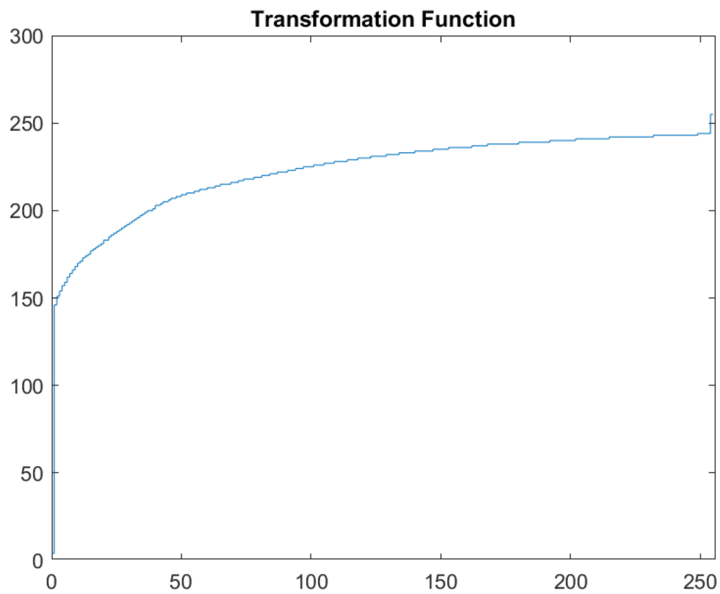
Step3. Apply transformation function to each element of the image.

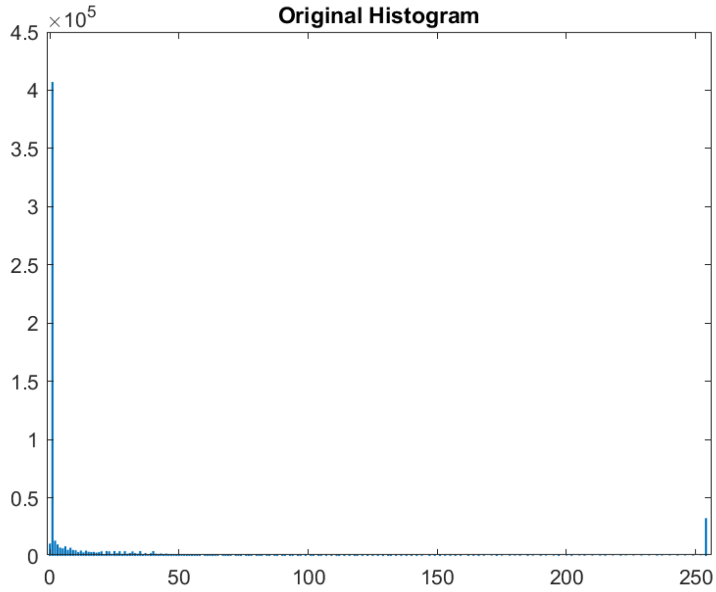
```
for i = 1 : size(input, 1)
    for j = 1 : size(input, 2)
        output(i, j) = T(input(i, j) + 1)
```

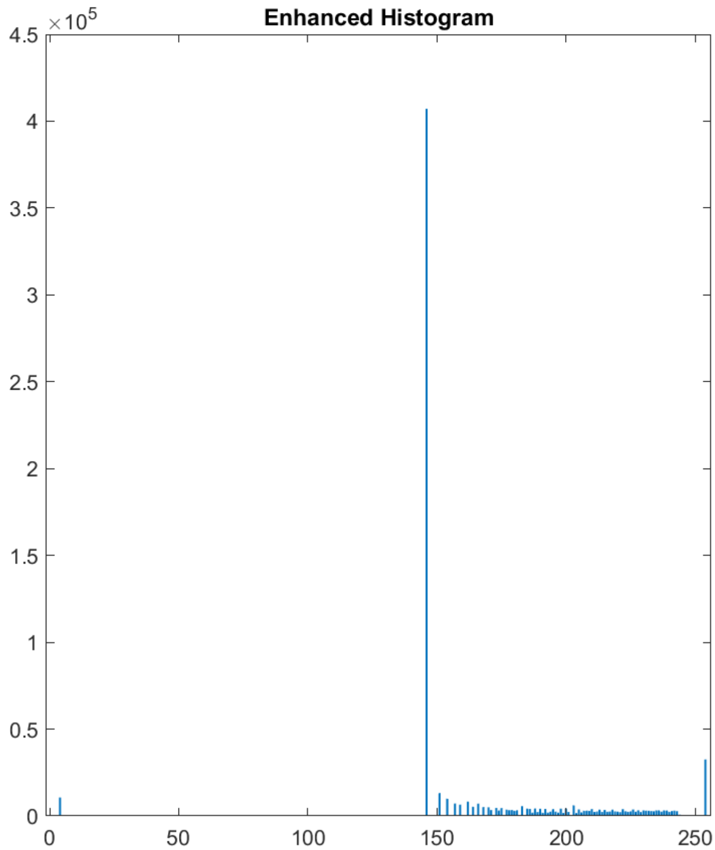
(2) Results

Fig3.8 (a)

Enhanced Image







(3) Discussion

The histogram of the enhanced image is more similar to uniform distribution. However, since there's an extremely dominant dark pixel value in the original image, the effect of histogram equalization is not visually good.

Proj03-03~04: Spatial Filtering & Enhancement Using the Laplacian

(1) **spatialFiltering:**

Step1. Zero padding

```
pad_size = floor(mask_size / 2)
padded_input = padarray(input, [pad_size pad_size])
```

Step2. Reverse the mask and stack its rows into a column vector

```
vec_mask = reshape(flip(flip(mask, 1), 2)', [], 1)
```

Step3. Convolution

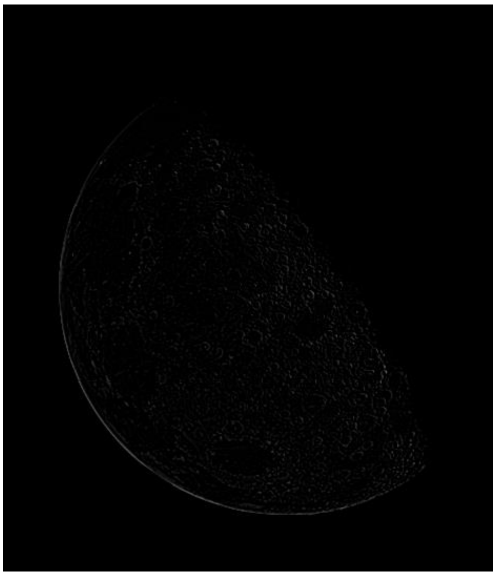
```
for i = 1 : size(input, 1)
    for j = 1 : size(input, 2)
        % Extract the region and stack its rows into a row vector
        vec_padded_input = reshape(
            padded_input(i : i + mask_size - 1, j : j + mask_size - 1)', 1, [])
        output(i, j) = vec_padded_input * vec_mask
```

laplacianFiltering:

```
scaledLaplacian = scale * spatialFiltering(input, laplacianMask)
output = input + scaledLaplacian
```

(2) Reproduced Results:

4/e Fig 3.46 (a) 4/e Fig 3.46 (b)



4/e Fig 3.46 (c) 4/e Fig 3.46 (d)

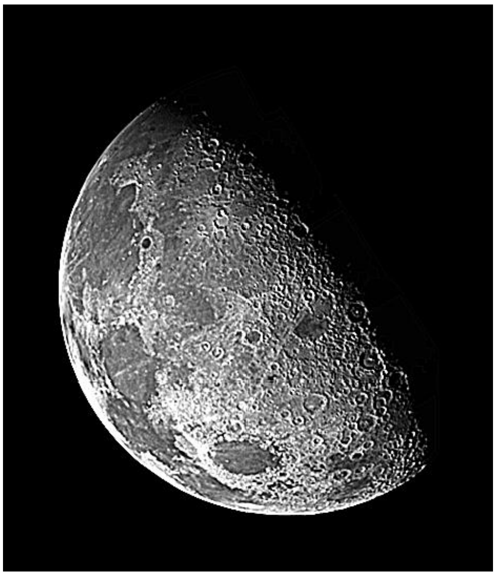
(3) Analysis of different scales and mask sizes

Different mask sizes:

Mask size = 3 x 3

$$kernel = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Mask size = 5 x 5

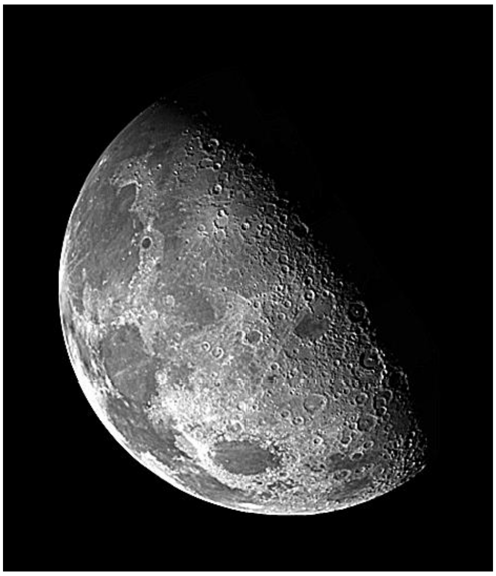


$$kernel = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Different scales:

$scale = -1$ (Fig 3.46 c)

$scale = -3$



From the above results, we can observe that both the larger mask size and the larger absolute-valued scale may achieve more enhancement.

Moreover, since the borders of the original image are all zero-valued pixels, using different padding methods in spatial filtering will get the same result.