HW4

*Date: 2021. 10. 01*

*Student ID: 21600635*

*Name: Bomoon JUNG*

**1. Homework 4**

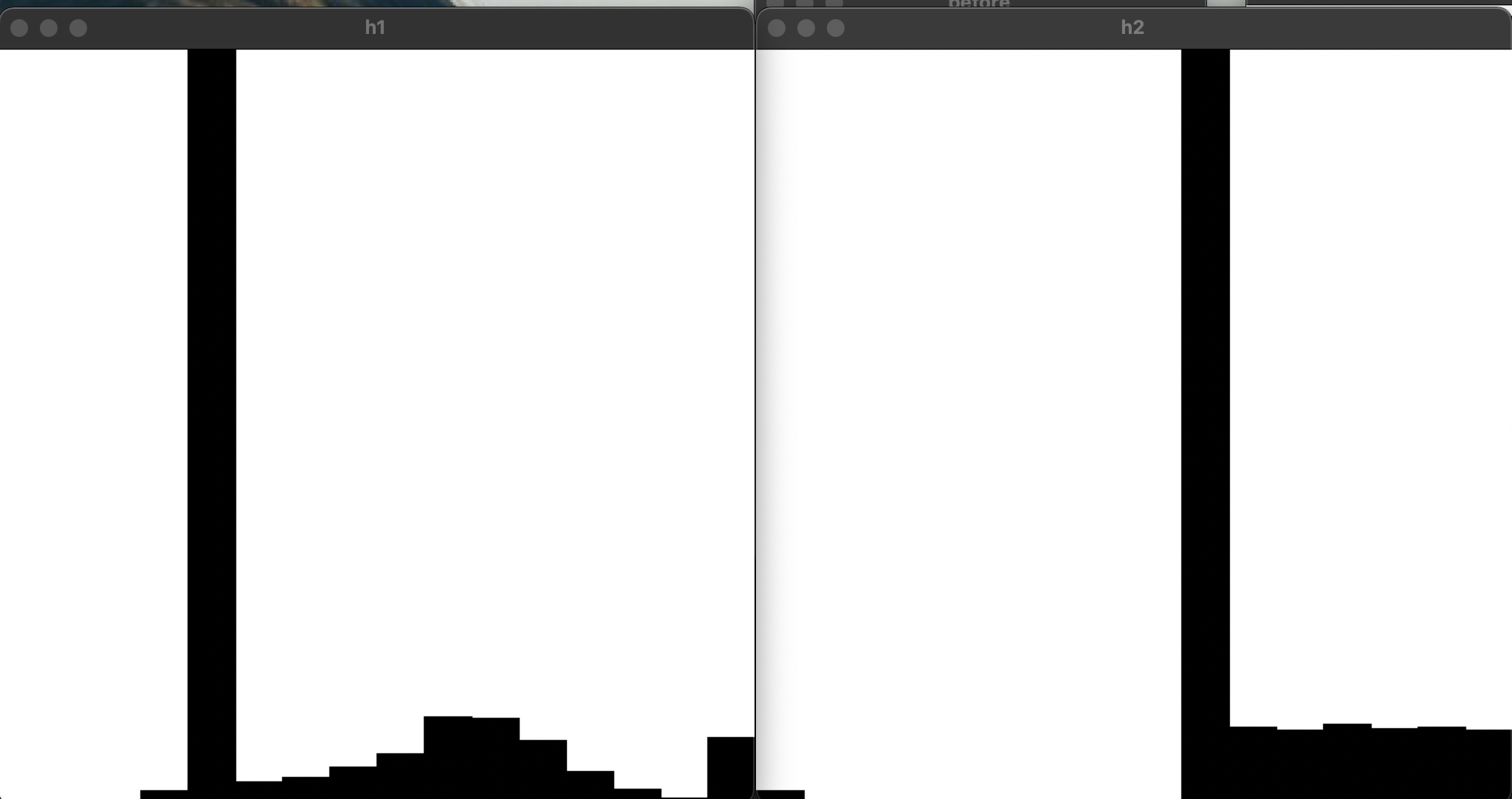
1) Please follow the instructions below.

* Read an image “moon.png” as gray scal image
* Perform histogram equalization on the input image
* Display each image with the window name as “before”, “after”
* Display each histogram of the input and the result image
* Set the number of bins to 16
* Ste the matrix size for displaying histogram ass width: 512, height:512
* Display each image with the window name as “h1”, “h2”
* Compute. The value of each component of a normalized histogram of the input image; write all values on the input image; and display the result
* Set the number of bins to 8
* You can arbitrarily set the. Font, color, and position of the text (but consecutively)

실내, 다른, 진드기이(가) 표시된 사진

자동 생성된 설명

**Figure 1. results of “before” and “after”**

****

**Figure 2. results of “h1” and “h2”**

2) explanation

First, read ‘moon.png’ and then equalize the average of the entire image through the equalizeHist() function. If the colors are evenly distributed, the contrast is ensured through equalization. However, as shown in figure 2, since the input image is dominated by data in a specific range, the average rises when equalization is performed, and the overall brightness rises. Also, it can be confirmed that data in a specific range still appears dominant.

3) Source code

*#include* <opencv2/opencv.hpp>

*#include* <iostream>

*#include* <string>

using *namespace* cv ;

using *namespace* std ;

Mat

drawHistogram (Mat src, *int* histSize) {

*int* width = 512 ;

*int* height = 512 ;

*float* range[] = { 0, 256 } ;

const *float*\* hist\_Range = { range } ;

*// bin interval*

*int* bin\_w = cvRound((*double*)width / histSize) ;

Mat histImage = Mat(width, height, CV\_8UC3, Scalar(255, 255, 255)) ;

Mat hist ;

calcHist(&src, 1, 0, Mat(), hist, 1, &histSize, &hist\_Range) ;

normalize(hist, hist, 0, histImage.rows, NORM\_MINMAX, -1, Mat()) ;

*for* ( *int* i = 0 ; i < histSize ; i++ ) {

rectangle(histImage, Point(bin\_w \* i, height), Point(bin\_w \* i + width / histSize, height - cvRound(hist.at<*float*>(i))), Scalar(0, 0, 0), -1) ;

*// putText(histImage, to\_string(cvRound(hist.at<float>(i))), Point(bin\_w \* i + bin\_w / 6, height), 1, 0.8, Scalar(100, 150, 200), 1, 8) ;*

}

*return* histImage ;

}

Mat

drawText (Mat src, *int* histSize) {

*float* range[] = { 0, 256 } ;

const *float*\* hist\_Range = { range } ;

Mat hist ;

calcHist(&src, 1, 0, Mat(), hist, 1, &histSize, &hist\_Range) ;

normalize(hist, hist, 0, 512, NORM\_MINMAX, -1, Mat()) ;

*for* ( *int* i = 0 ; i < histSize ; i++ ) {

putText(src, format("bin %d: %d", i + 1, cvRound(hist.at<*float*>(i))), Point(5, i \* 20 + 20), 1, 0.8, Scalar(255, 255, 255), 1, 8) ;

}

*return* src ;

}

*int*

main() {

*// 1. read an image*

Mat moon = imread("./resources/moon.png", 0) ;

*if* ( moon.empty() ) {

cout << "no such file" << endl ;

*return* 0 ;

}

*// 2. Perform histogram equalization on the input image*

Mat before = moon.clone() ;

Mat after ;

equalizeHist(before, after) ;

*// // display each image with the window name as 'before', 'after'*

*// imshow("before", before) ;*

*// imshow("after", after) ;*

*// 3. display each histogram of the input and result image*

Mat before\_hist\_16 = drawHistogram(before, 16) ;

Mat after\_hist\_16 = drawHistogram(after, 16) ;

*// display each image with the window name as 'h1', 'h2'*

imshow("h1", before\_hist\_16) ;

imshow("h2", after\_hist\_16) ;

*// 4. compute the value of each component of a normalized histogram of the input image;*

*// write all values on the input image;*

*// and display the result*

*// Mat before\_hist\_8 = drawHistogram(before, 8) ;*

*// Mat after\_hist\_8 = drawHistogram(after, 8) ;*

before = drawText(before, 8) ;

after = drawText(after, 8) ;

imshow("before", before) ;

imshow("after", after) ;

*// imshow("before histogram(bin 8)", before\_hist\_8) ;*

*// imshow("after histogram(bin 8)", after\_hist\_8) ;*

waitKey(0) ;

*return* 0 ;

}