2021/10/25 陳嘉政 r10922171 HW3
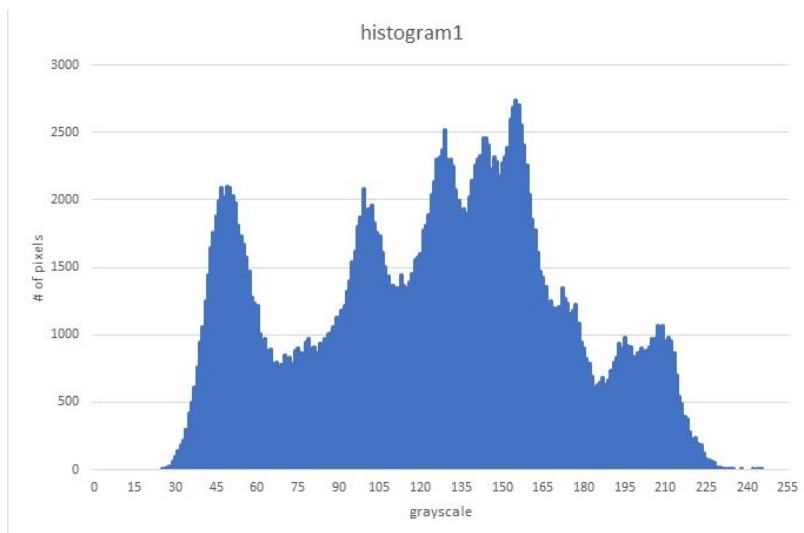
1. (1)原始 lena

與上禮拜作業重疊，所以就不貼 code 了

(2) histogram
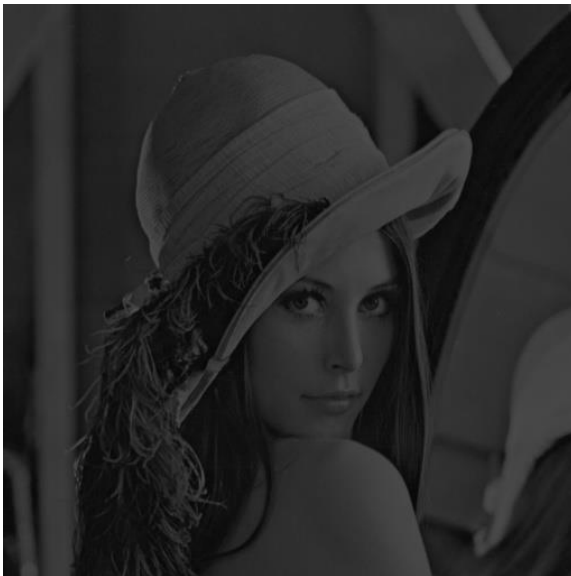

2. (1)將 lena 亮度調成 1/3

```
/*灰階值除3並輸出影像*/
for (int i = 0; i < img2.rows; i++) {
    for (int j = 0; j < img2.cols; j++) {
        img2.at<uchar>(i, j) /= 3;
    }
}
imshow("img2", img2);
waitKey(0);
imwrite("part2.jpg", img2);
```
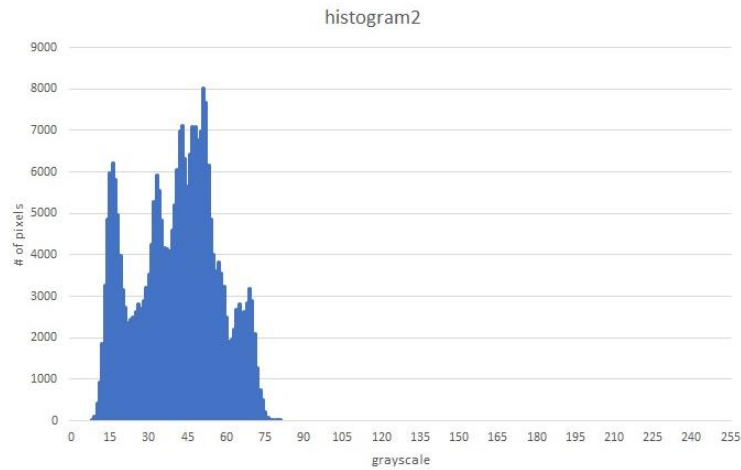

(2)histogram

與上次作業相同，先計算每個灰階的個數，然後以 **CSV**

檔輸出到 **excel**，再用 **excel** 圖表做出直方圖

```
/*count pixels*/
int histogram2[256] = { 0 };
for (int i = 0; i < img2.rows; i++) {
    for (int j = 0; j < img2.cols; j++) {
        histogram2[img2.at<uchar>(i, j)] += 1;
    }
}

/*以CSV檔輸出*/
fstream file2("part2.csv", ios::out);
for (int i = 0; i < 256; i++) {
    file2 << histogram2[i] << endl;
}
file2.close();
```

histogram2

3.(1)將 part2 的 lena 做直方圖均衡化後輸出

```
/*count pixels*/
int grayscale[256] = { 0 };
for (int i = 0; i < img3.rows; i++) {
    for (int j = 0; j < img3.cols; j++) {
        grayscale[img3.at<uchar>(i, j)] += 1;
    }
}
```

```
/*equalization*/
/*count cdf*/
int cdf[256] = { 0 };
int count = 0;
for (int i = 0; i < 256; i++) {
    if (grayscale[i] != 0) {
        count += grayscale[i];
        cdf[i] = count;
    }
}


/*find max_cdf and min_cdf*/
int max = 0, min = 0;
for (int i = 0; i < 256; i++) {
    if (cdf[i] != 0) {
        min = cdf[i];
        break;
    }
}

for (int i = 255; i >= 0; i--) {
    if (cdf[i] != 0) {
        max = cdf[i];
        break;
    }
}
```

```
/*計算新的灰值*/
double h[256] = { 0 };
for (int i = 0; i < 256; i++) {
    if (cdf[i] != 0) {
        h[i] = round((double)((cdf[i] - min) * 255 / (max - min)));
    }
}

/*out put lena*/
for (int i = 0; i < img3.rows; i++) {
    for (int j = 0; j < img3.cols; j++) {
        for (int k = 0; k < 256; k++) {
            if (img3.at<uchar>(i, j) == k) {
                img3.at<uchar>(i, j) = h[k];
                break;
            }
        }
    }
}

imshow("img3", img3);
waitKey(0);
imwrite("part3.jpg", img3);
```


(2)histogram

直方圖的 code 與之前皆相同，就不重複貼了