#include <iostream>

#include <C:\opencv\sources\modules\highgui\include\opencv2\highgui.hpp>

#include <C:\opencv\sources\modules\imgproc\include\opencv2\imgproc.hpp>

using std::cout;

using std::cin;

using std::endl;

void imhist(Mat image, int histogram[])

{

// initialize all intensity values to 0

for (int i = 0; i < 256; i++)

{

histogram[i] = 0;

}

// calculate the no of pixels for each intensity values

for (int y = 0; y < image.rows; y++)

for (int x = 0; x < image.cols; x++)

histogram[(int)image.at<uchar>(y, x)]++;

}

void cumhist(int histogram[], int cumhistogram[])

{

cumhistogram[0] = histogram[0];

for (int i = 1; i < 256; i++)

{

cumhistogram[i] = histogram[i] + cumhistogram[i - 1];

}

}

void histDisplay(int histogram[], const char\* name)

{

int hist[256];

for (int i = 0; i < 256; i++)

{

hist[i] = histogram[i];

}

// draw the histograms

int hist\_w = 512; int hist\_h = 400;

int bin\_w = cvRound((double)hist\_w / 256);

Mat histImage(hist\_h, hist\_w, CV\_8UC1, Scalar(255, 255, 255));

// find the maximum intensity element from histogram

int max = hist[0];

for (int i = 1; i < 256; i++) {

if (max < hist[i]) {

max = hist[i];

}

}

// normalize the histogram between 0 and histImage.rows

for (int i = 0; i < 256; i++) {

hist[i] = ((double)hist[i] / max) \* histImage.rows;

}

// draw the intensity line for histogram

for (int i = 0; i < 256; i++)

{

line(histImage, Point(bin\_w \* (i), hist\_h),

Point(bin\_w \* (i), hist\_h - hist[i]),

Scalar(0, 0, 0), 1, 8, 0);

}

// display histogram

namedWindow(name, CV\_WINDOW\_AUTOSIZE);

imshow(name, histImage);

}

int main()

{

// Load the image

Mat image = imread("hello.jpg", CV\_LOAD\_IMAGE\_GRAYSCALE);

// Generate the histogram

int histogram[256];

imhist(image, histogram);

// Caluculate the size of image

int size = image.rows \* image.cols;

float alpha = 255.0 / size;

// Calculate the probability of each intensity

float PrRk[256];

for (int i = 0; i < 256; i++)

{

PrRk[i] = (double)histogram[i] / size;

}

// Generate cumulative frequency histogram

int cumhistogram[256];

cumhist(histogram, cumhistogram);

// Scale the histogram

int Sk[256];

for (int i = 0; i < 256; i++)

{

Sk[i] = cvRound((double)cumhistogram[i] \* alpha);

}

// Generate the equlized histogram

float PsSk[256];

for (int i = 0; i < 256; i++)

{

PsSk[i] = 0;

}

for (int i = 0; i < 256; i++)

{

PsSk[Sk[i]] += PrRk[i];

}

int final[256];

for (int i = 0; i < 256; i++)

final[i] = cvRound(PsSk[i] \* 255);

// Generate the equlized image

Mat new\_image = image.clone();

for (int y = 0; y < image.rows; y++)

for (int x = 0; x < image.cols; x++)

new\_image.at<uchar>(y, x) = saturate\_cast<uchar>(Sk[image.at<uchar>(y, x)]);

// Display the original Image

namedWindow("Original Image");

imshow("Original Image", image);

// Display the original Histogram

histDisplay(histogram, "Original Histogram");

// Display equilized image

namedWindow("Equilized Image");

imshow("Equilized Image", new\_image);

// Display the equilzed histogram

histDisplay(final, "Equilized Histogram");

waitKey();

return 0;

}