浙江大学

数字图像处理实验报告 3

课程名称：	数字图像处理
姓　　名：	毛雨帆
学　　院：	信息与电子工程学院
专　　业：	电子科学与技术
学　　号：	3180104584
指导教师：	李东晓
日　　期：	2021 年 5 月 2 日

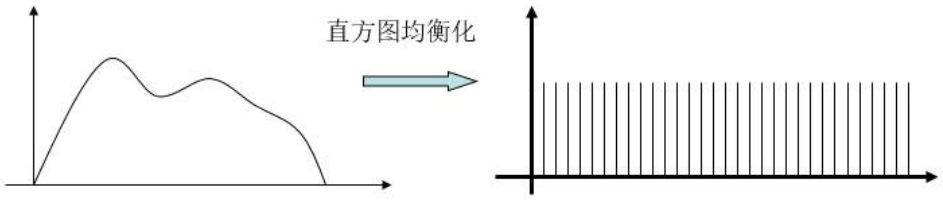
一：实验任务

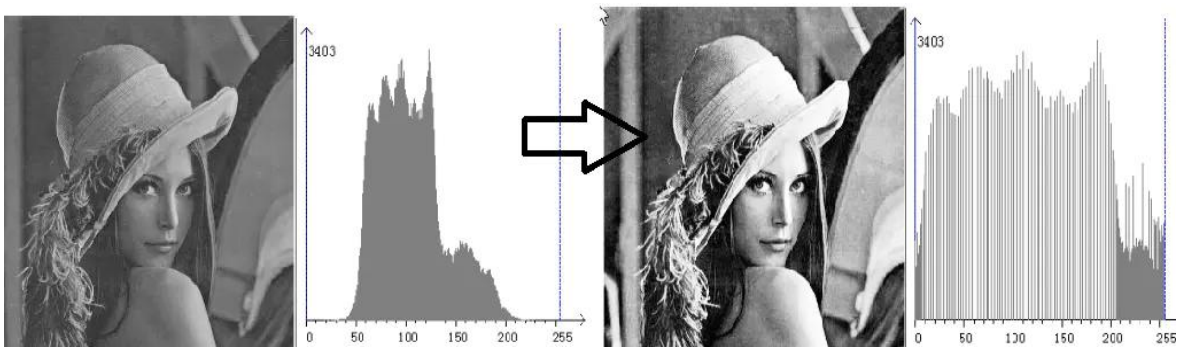
Color Image Enhancement by Histogram Processing
彩色图像通过直方图均衡进行图像增强。
(a) Download the dark-stream color picture in Fig. 6.35 from the book web site. Convert the image to RGB (see comments at the beginning of Project 06-01). Histogram-equalize the R, G, and B images separately using the histogram-equalization program from Project 03-02 and convert the image back to tif format.
下载图中的暗流彩色图片。6.35.来自图书网站。将图像转换为 RGB（请参见项目 06-01 开始时的注释）。直柱方图使用项目 03-02 的柱方图均衡程序分别均衡 R、G 和 B 图像，并将图像转换回 tif 格式。
(b) Form an average histogram from the three histograms in (a) and use it as the basis to obtain a single histogram equalization intensity transformation function. Apply this function to the R, G, and B components individually, and convert the results to jpg. Compare and explain the differences in the tif images in (a) and (b).
从(a)中的三个直方图中形成一个平均直方图，并以此为基础，得到单个直方图均衡强度变换函数。将此功能分别应用于 R、G 和 B 组件，并将结果转换为 jpg。比较并解释(a)和(b)中的 tif 图像中的差异。

二：算法原理与设计

（1）灰度图像直方图均衡回顾

直方图均衡化是将原图像的直方图通过变换函数变为均匀的直方图，然后按均匀直方图修改原图像，从而获得一幅灰度分布均匀的新图像。

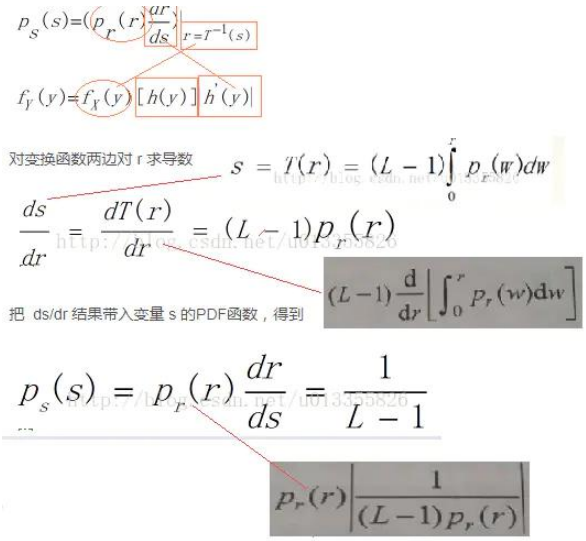




直方图均衡算法推导如下：

首先考虑连续的灰度值，用变量 r 表示输入图像的灰度，用 s 表示输出图像的灰度。r 范围是 [0, L-1]，则函数： s = T(r), r 的范围：[0 , L-1] （0 为全黑，L-1 为全白）。
T(r)满足下列两个条件：
1.T(r)在区间 0≤r≤1 中为 严格单调递增函数，保证原图各灰度级在变换后仍保持从黑到白
2.当 0≤r≤1 时,0≤T(r) ≤1，保证变换前后灰度值动态范围的一致性。

根据概率论以及微积分的知识，



最终得到重要的变换函数：

$$s = T(\mathrm{r}) = (L-1)\int_0^r p_r(w)dw$$

其中 $\int_0^r p_r(w)dw$ 为累积分布函数CDF

对于离散值，我们处理其概率（直方图值）与求和来替代处理概率密度函数与积分。因此，一幅数字图像中灰度级出现的概率（近似）和离散变换形式如下：

$$p_r(r) = \frac{n_k}{MN}$$

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j, \quad \mathrm{k}=0, 1, 2.\ldots.\mathrm{L}-1$$

综上，直方图均衡化步骤为：
1.统计图像中每个灰度级出现的次数，计算图像中每个灰度级出现的概率；
2.根据变换公式得到直方图均衡化的变换函数；
3.根据变换函数映射到每个像素点；
4.输出映射后的图像；

（2）彩色图像直方图均衡处理

直方图均衡自动确定一个转换，试图产生强度值均匀直方图的图像。然而在 RGB 图像中，由于具有三个分量，如果直方图独立地均衡彩色图像的分量图像通常是不明智的，这将导致错误的颜色。

一种更合乎逻辑的方法是均匀地传播颜色强度，使颜色本身（例如，色调）保持不变。可以在 HSI 空间开展，对 I 进行直方图均衡，保持 H 和 S 保持不变。

三：代码实现

本程序分别实现对 RGB 图像的三个分量分别进行直方图均衡以及仅对图像亮度（先转化为 HSI 图像）进行直方图均衡。其中 rgb2hsi 函数实现将 RGB 图像转化为 HSI 图像，而 hsi2rgb 函数实现将 HSI 图像转化为 RGB 图像。为方便起见，直方图均衡的功能用系统自带函数 histeq 实现，不再引用 3.02 中的代码。（实际也无大区别）

```
%%
close all;
clc;
clear all;
%%
img = imread('Fig0635.tif');
figure
subplot(1,3,1);
imshow(img);
title('original image');

%% 对 RGB3 个通道的灰度值分别做直方图均衡化，然后再合为一幅新的图像
R = img(:, :, 1);
G = img(:, :, 2);
B = img(:, :, 3);

A = histeq(R);
B = histeq(G);
C = histeq(B);

img1 = cat(3, A, B, C);

subplot(1,3,2);
```

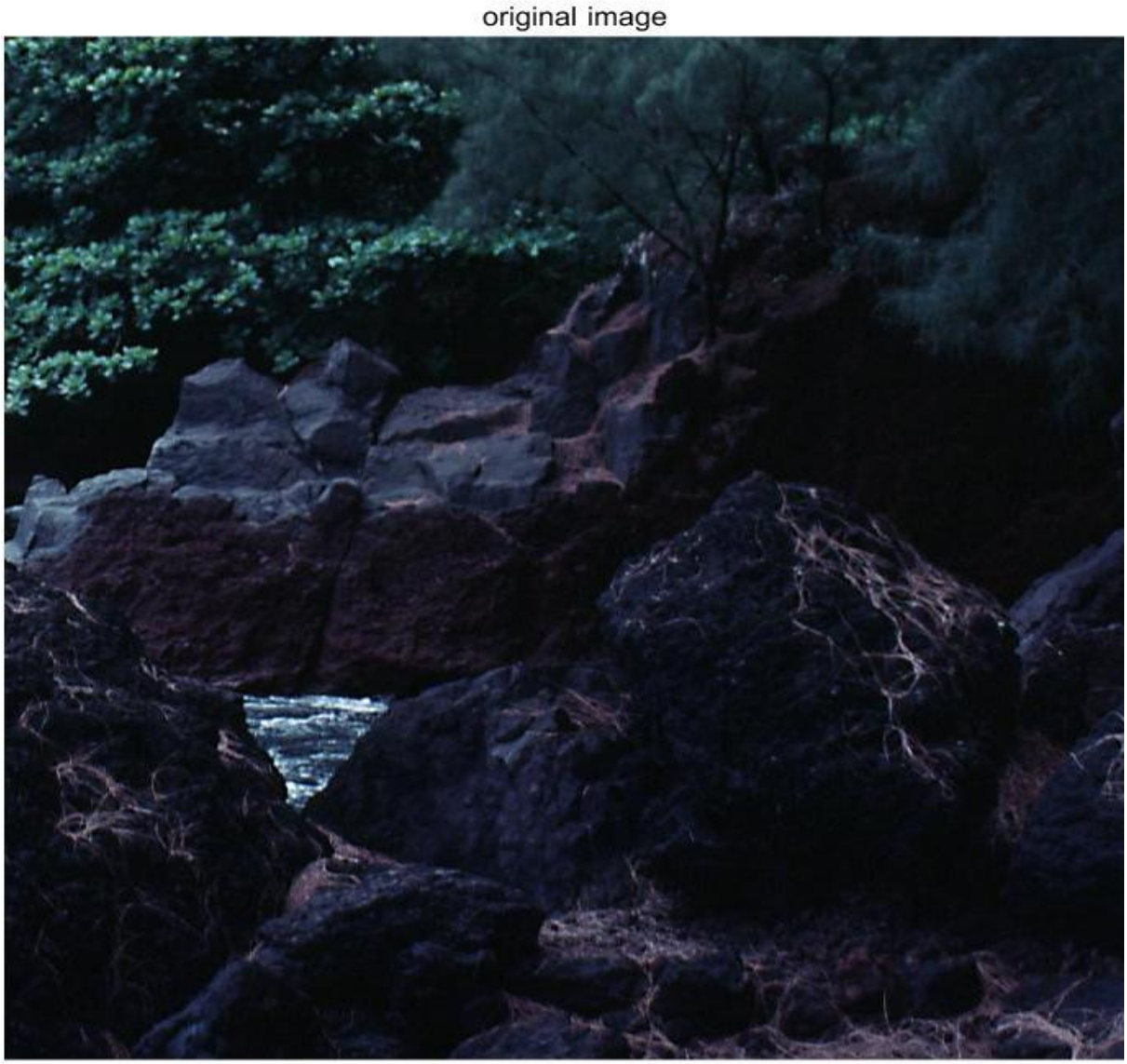
```
imshow(img1);
title('histogram-equalization 1');
%% 先将 RGB 格式的图像转换为 HSI 格式的图像，然后再对亮度 I 做直方图均衡化，紧接着转换成 RGB 格式的图像

img_hsi = rgb2hsi(img);
img_hsi_i = img_hsi(:, :, 3);
img_hsi_I = histeq(img_hsi_i);
img_hsi(:, :, 3) = img_hsi_I;
img2 = hsi2rgb(img_hsi);

subplot(1,3,3);
imshow(img2);
title('histogram-equalization 2');

function rgb = hsi2rgb(hsi)
%HSI2RGB Converts an HSI image to RGB.
%   RGB = HSI2RGB(HSI) converts an HSI image to RGB, where HSI is
%   assumed to be of class double with:
%     hsi(:, :, 1) = hue image, assumed to be in the range
%                    [0, 1] by having been divided by 2*pi.
%     hsi(:, :, 2) = saturation image, in the range [0, 1].
%     hsi(:, :, 3) = intensity image, in the range [0, 1].
%
%   The components of the output image are:
%     rgb(:, :, 1) = red.
%     rgb(:, :, 2) = green.
%     rgb(:, :, 3) = blue.

%   Copyright 2002-2004 R. C. Gonzalez, R. E. Woods, & S. L. Eddins
%   Digital Image Processing Using MATLAB, Prentice-Hall, 2004
%   $Revision: 1.5 $  $Date: 2003/10/13 01:01:06 $

% Extract the individual HSI component images.
H = hsi(:, :, 1) * 2 * pi;
S = hsi(:, :, 2);
I = hsi(:, :, 3);

% Implement the conversion equations.
R = zeros(size(hsi, 1), size(hsi, 2));
G = zeros(size(hsi, 1), size(hsi, 2));
B = zeros(size(hsi, 1), size(hsi, 2));

% RG sector (0 <= H < 2*pi/3).
```

```
idx = find( (0 <= H) & (H < 2*pi/3));
B(idx) = I(idx) .* (1 - S(idx));
R(idx) = I(idx) .* (1 + S(idx) .* cos(H(idx)) ./ ...
                                    cos(pi/3 - H(idx)));
G(idx) = 3*I(idx) - (R(idx) + B(idx));

% BG sector (2*pi/3 <= H < 4*pi/3).
idx = find( (2*pi/3 <= H) & (H < 4*pi/3) );
R(idx) = I(idx) .* (1 - S(idx));
G(idx) = I(idx) .* (1 + S(idx) .* cos(H(idx) - 2*pi/3) ./ ...
                 cos(pi - H(idx)));
B(idx) = 3*I(idx) - (R(idx) + G(idx));

% BR sector.
idx = find( (4*pi/3 <= H) & (H <= 2*pi));
G(idx) = I(idx) .* (1 - S(idx));
B(idx) = I(idx) .* (1 + S(idx) .* cos(H(idx) - 4*pi/3) ./ ...
                                     cos(5*pi/3 - H(idx)));
R(idx) = 3*I(idx) - (G(idx) + B(idx));

% Combine all three results into an RGB image.  Clip to [0, 1] to
% compensate for floating-point arithmetic rounding effects.
rgb = cat(3, R, G, B);
rgb = max(min(rgb, 1), 0);
end

function hsi = rgb2hsi(rgb)
%RGB2HSI Converts an RGB image to HSI.
%   HSI = RGB2HSI(RGB) converts an RGB image to HSI. The input image
%   is assumed to be of size M-by-N-by-3, where the third dimension
%   accounts for three image planes: red, green, and blue, in that
%   order. If all RGB component images are equal, the HSI conversion
%   is undefined. The input image can be of class double (with values
%   in the range [0, 1]), uint8, or uint16.
%
%   The output image, HSI, is of class double, where:
%     hsi(:, :, 1) = hue image normalized to the range [0, 1] by
%                    dividing all angle values by 2*pi.
%     hsi(:, :, 2) = saturation image, in the range [0, 1].
%     hsi(:, :, 3) = intensity image, in the range [0, 1].

%   Copyright 2002-2004 R. C. Gonzalez, R. E. Woods, & S. L. Eddins
%   Digital Image Processing Using MATLAB, Prentice-Hall, 2004
%   $Revision: 1.5 $  $Date: 2005/01/18 13:44:59 $
```

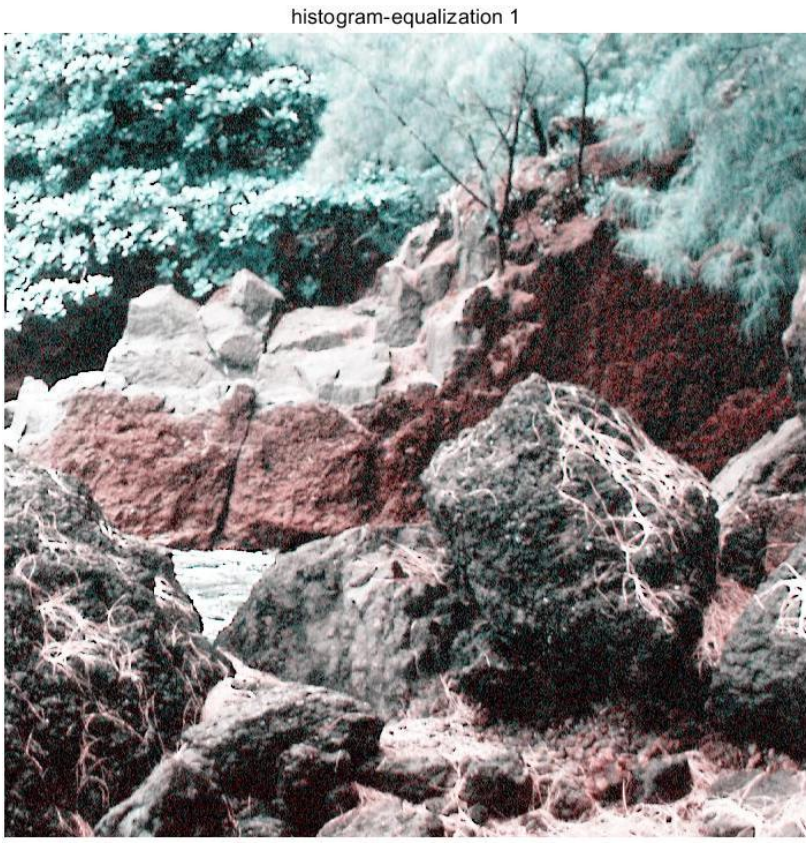
```
% Extract the individual component images.
rgb = im2double(rgb);
r = rgb(:, :, 1);
g = rgb(:, :, 2);
b = rgb(:, :, 3);

% Implement the conversion equations.
num = 0.5*((r - g) + (r - b));
den = sqrt((r - g).^2 + (r - b).*(g - b));
theta = acos(num./(den + eps));

H = theta;
H(b > g) = 2*pi - H(b > g);
H = H/(2*pi);

num = min(min(r, g), b);
den = r + g + b;
den(den == 0) = eps;
S = 1 - 3.* num./den;

H(S == 0) = 0;

I = (r + g + b)/3;

% Combine all three results into an hsi image.
hsi = cat(3, H, S, I);
end
```

四：实验结果

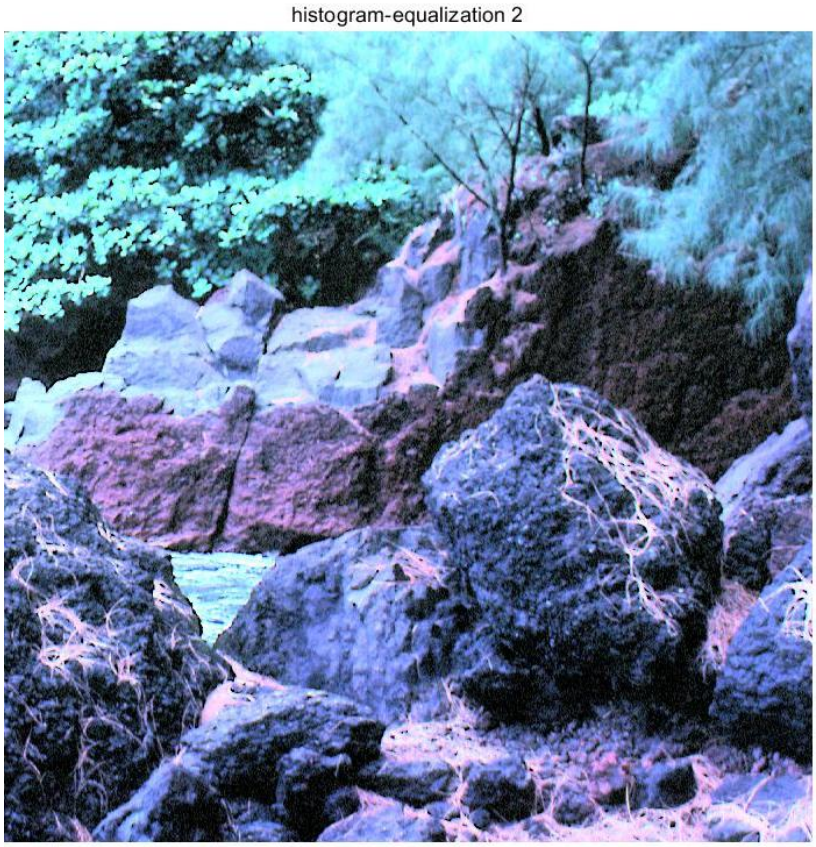
运行上述程序



此为原图



Histogram-equalization1:对 RGB 分量分别进行直方图均衡



Histogram-equalization2:转化为 HSI 图像，对 I 进行直方图均衡，再转化为 RGB 图像

五：总结

（1）实验结果分分析：不难看出，采用第一种直方图均衡的方法，虽然增加了亮度和对比度，一定程度上起到了图像增强的作用，但与原图相比较，各部分的颜色都已经发生了改变，使得图像失去了它原有的意义，因此这种方法不可取。而采用第二种直方图均衡的方法，在 HSI 空间中进行处理，没有改变色调和饱和度，使得颜色保持了正确，但亮度和对比度得到了提升，因此更为合适。

（2）本次实验的代码实现相对较为简单，只需按照设计好的算法流程进行代码的编写即可。值得提出的一点是实验中注意到的 matlab 中图像的编码方式。

为了节省存储空间，matlab 为图像提供了特殊的数据类型 uint8(8 位无符号整数），以此方式存储的图像称作 8 位图像。imread 把灰度图像存入一个 8 位矩阵，当为 RGB 图像时，就存入 8 位 RGB 矩阵中。因此，matlab 读入图像的数据是 uint8，而 matlab 中数值一般采用 double 型（64 位）存储和运算。所以要先将图像转为 double 格式的才能运算。

im2uint8 和 uint8 都是将图像数据转化为 uint8 ,前者要求被转化的数据必须是符合图像数据标准(如：double [0 1]) ，而 uint8 则不必，它会自动截断数据。

im2double 和 double 。double 就是将一个数据的类型转化为 double ，但是数值不变；im2double 将输入的 uint8 或 uint16 归一化到[0 1]区间 ，如果输入是 double，则不进行归一化。