专业：_自动化（电气）_
姓名：_____潘盛琪_____
学号：___3170105737___
日期：_____3.21_____
地点：**生工食品学院机房**

浙江大学实验报告

课程名称：**计算机图像处理与机器视觉** 指导老师：**饶秀勤** 成绩：_____

实验名称：**伪彩色变换** 实验类型：**设计型**

一、实验目的和要求

将灰度图像送入具有不同变换特性的红、绿、蓝 3 个变换器(变换函数)，同一灰度由 3 个变换函数对其实施不同变换，并重新合成某种色彩

二、计算机配置与软件处理平台

硬件：

Windows 版本

Windows 10 家庭中文版

© 2018 Microsoft Corporation。保留所有权利。

系统

处理器:	AMD Ryzen 7 PRO 2700U w/ Radeon Vega Mobile Gfx 2.20 GHz
已安装的内存(RAM):	8.00 GB (6.93 GB 可用)
系统类型:	64 位操作系统，基于 x64 的处理器
笔和触控:	没有可用于此显示器的笔或触控输入

软件：基于 matlab

三、算法描述

1. 查表进行伪彩色变换

4、灰度级和使用彩色的对应关系是：{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}对应于{黑、深蓝、深绿、深红、深灰、品红、浅蓝、棕色、浅绿、浅红、浅灰、浅蓝绿、黄色、白色、深蓝绿、浅紫}，可应用Qbcolor(n)根据新的灰度值来确定颜色

5、显示彩色图像

QBColor(n)	颜色	对应RGB颜色值
0	黑	RGB(0, 0, 0)
1	蓝	RGB(0, 0, 191)
2	绿	RGB(0, 191, 0)
3	青	RGB(0, 191, 191)
4	红	RGB(191, 0, 0)
5	洋红	RGB(191, 0, 191)
6	黄	RGB(191, 191, 0)
7	白	RGB(191, 191, 191)
8	灰	RGB(64, 64, 64)
9	亮蓝	RGB(0, 0, 255)
10	亮绿	RGB(0, 255, 0)
11	亮青	RGB(0, 255, 255)
12	亮红	RGB(255, 0, 0)
13	亮洋红	RGB(255, 0, 255)
14	亮黄	RGB(255, 255, 0)
15	亮白	RGB(255, 255, 255)

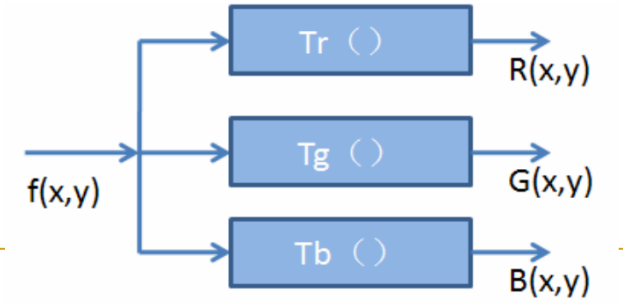
装 订 线

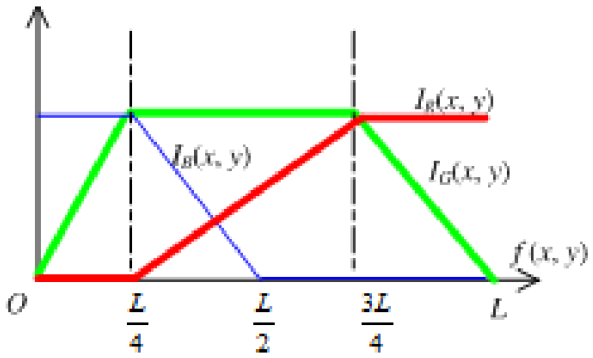
2. 灰度级彩色变换

这种伪彩色处理技术（在遥感技术中常称为假彩色合成方法），可以将灰度图像变为具有多种颜色渐变的连续彩色图像， 实现图像的连续伪彩色变换。

其变换过程为：将灰度图像送入具有不同变换特性的红、绿、蓝3个变换器(变换函数)，同一灰度由3个变换函数对其实施不同变换，并重新合成某种色彩。

可见，不同大小灰度级一定可以合成不同色彩。





若$f(x,y)=0$，则$I_B(x,y)=L, I_R(x,y)=I_G(x,y)=0$，显示蓝色。

若$f(x,y)=L/2$，则$I_G(x,y)=L$，$I_R(x,y)=I_G(x,y)=0$，显示为绿色。

若$f(x,y)=L$, 则$I_R(x,y)=L$，$I_B(x,y)=I_G(x,y)=0$，从而显示红色。

因此不难理解，若灰度图像$f(x,y)$灰度级在$0\sim L$之间变化， I_R 、I_B 、I_G 会有不同输出，从而合成不同的彩色图像。

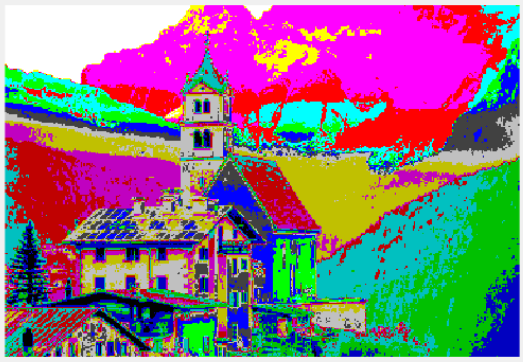
四、结果与讨论

4.1 输入图像

4.2 输出图像

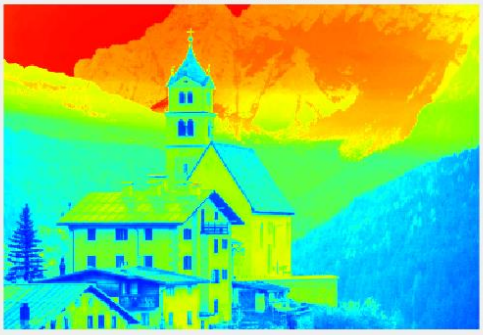
原图像

伪彩色变换图像



原图像

伪彩色变换图像



4.3 讨论

可以看到通过 QBcolour 查表进行伪彩色变换的结果明显是分成一块一块的，而灰度级伪彩色变换的颜色变化则基本连续，看起来比较舒适。

五、结论

灰度级伪彩色变换能对图像进行连续伪彩色变换，效果较好，可以用于遥感、医学、红外处理等多种领域。

六、源程序

1. 查表进行伪彩色变换

```
%%%离散（查表）伪彩色变换
img = imread('test.png');
img = rgb2gray(img);
[row, col] = size(img);        %确定图像大小
imgout = zeros(row, col, 3);   %初始化输出图像
max_grayscale = max(img(:));  %确定最大灰度值的大小
step = double(max_grayscale / 16);    %分为 16 档
for i = 1 : row
    for j = 1 : col
        scale = ceil(double(img(i, j))/step);
        imgout(i, j, 1:3) = QBcolor(scale);
    end
end
%%%显示图像
subplot(2, 1, 1);
imshow(img);
title('原图像');
subplot(2, 1, 2);
imshow(uint8(imgout));
title('伪彩色变换图像');
%%%输入灰度级数，输出对应的 RGB 值
function y = QBcolor(n)
    switch n
        case 1
            y = [1, 1, 1];
        case 2
            y = [1, 1, 192];
        case 3
            y = [1, 192, 1];
        case 4
            y = [1, 192, 192];
        case 5
            y = [192, 1, 1];
        case 6
```

```
            y = [192, 1 ,192];
        case 7
            y = [192, 192, 1];
        case 8
            y = [192, 192, 192];
        case 9
            y = [65, 65, 65];
        case 10
            y = [1, 1, 256];
        case 11
            y = [1, 256 ,1];
        case 12
            y = [1, 256, 256];
        case 13
            y = [256, 1, 1];
        case 14
            y = [256, 1, 256];
        case 15
            y = [256, 256, 1];
        case 16
            y = [256, 256, 256];
    end
end
```

2. 灰度级伪彩色变换

```
%%%连续（分段函数）伪彩色变换
img = imread('test.png');
img = rgb2gray(img);
img = double(img);            %必须转为 double 后再处理，否则会导致计算时数据溢出
[row, col] = size(img);       %确定图像大小
imgout = zeros(row, col, 3);   %初始化输出图像
max_grayscale = max(img(:));  %确定最大灰度值的大小
for i = 1:row
        for j = 1:col
                grayscale = img(i, j);
        [R, G, B] = gray2rgb(grayscale, max_grayscale);
        imgout(i, j, 1:3) = [R, G, B];
        end
end
%%%显示图像
subplot(2, 1, 1);
imshow(uint8(img));
title('原图像');
subplot(2, 1, 2);
```

```
imshow(uint8(imgout));
title('伪彩色变换图像');
function [R, G, B] = gray2rgb(n, L)
        if n < L/4
                R = 1;
                G = 256*4/L*n;
                B = 256;
        elseif n < L/2
                R = 256*2/L*(n-L/4);
                G = 256;
                B = 256-256*4/L*(n-L/4);
        elseif n < 3*L/4
                R = 256*2/L*(n-L/4);
                G = 256;
                B = 1;
        else
                R = 256;
                G = 256-256*4/L*(n-(3*L)/4);
                B = 1;
        end
end
```