**实验报告**

装 订 线

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**

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**

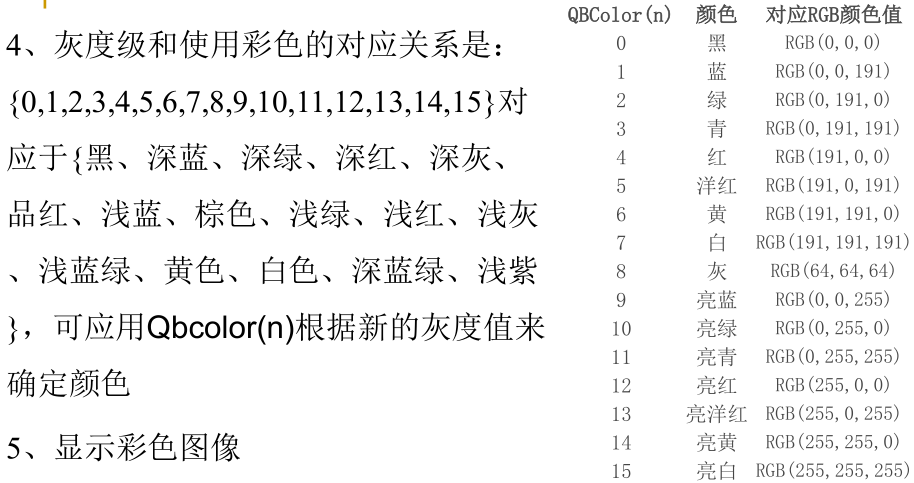
硬件：



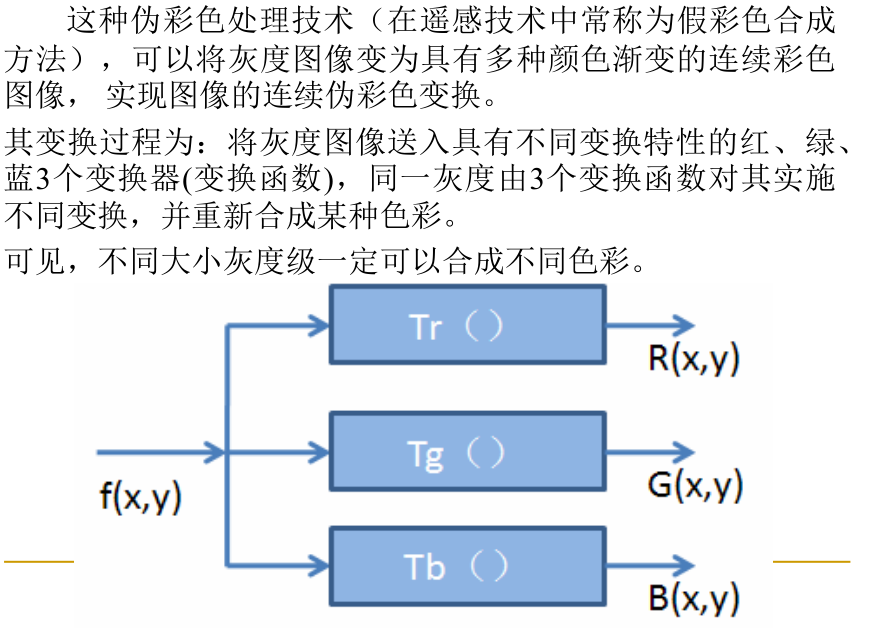
软件：基于matlab

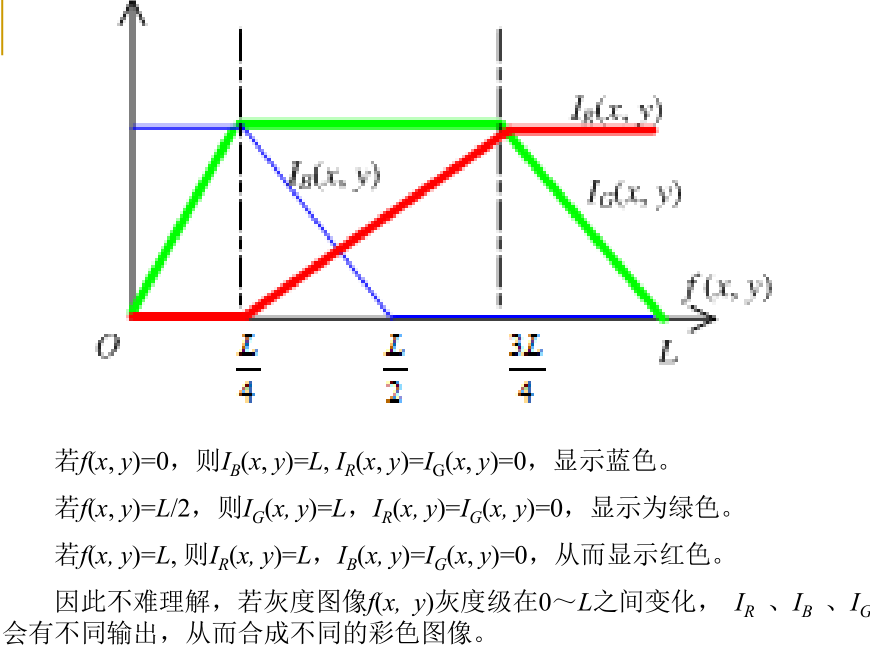
**三、算法描述**

1. 查表进行伪彩色变换



1. 灰度级彩色变换



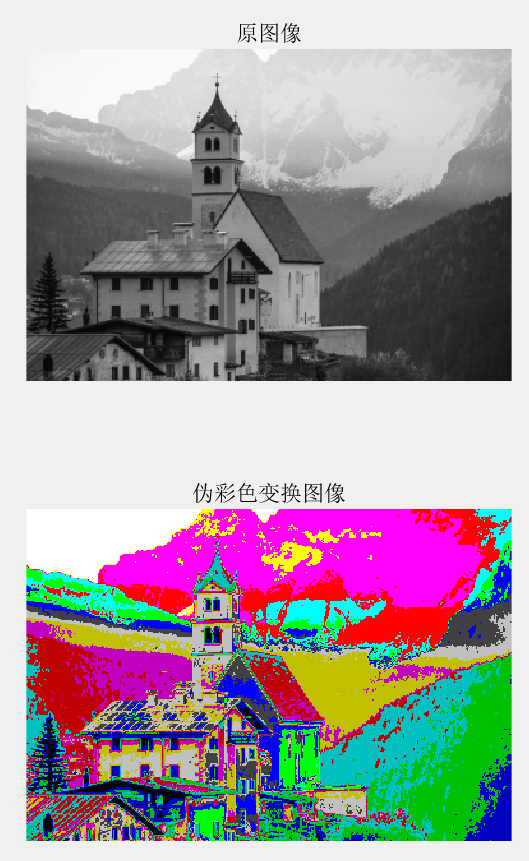


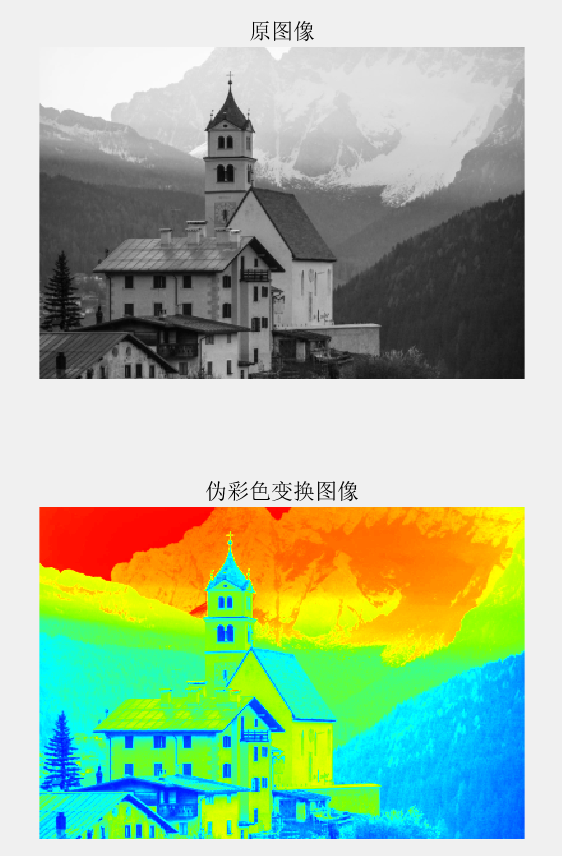
**四、结果与讨论**

**4.1 输入图像**



**4.2 输出图像**





**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

1. 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