《数字图像处理》实验报告

实验名称　　：实验 7 彩色图像处理技术

实验日期　　：2022.10.14

姓　　名　　：傅康

学　　号　　：084520126

班　　级　　：医信 20

成　　绩　　：

信息技术学院

南京中医药大学

实验目的：

1. 通过应用 MATLAB 语言编程实现对图像的处理，进一步熟悉 MATLAB 软件的编程及应用。
2. 熟悉及掌握在 MATLAB 中如何实现彩色图像、灰度图像、索引图像、二值图像的转换。
3. 理解伪彩色映射算法思想，并自主设计分段函数实现伪彩色增强算法。
4. 设计 GUI 完成转换要求。

实验内容和要求

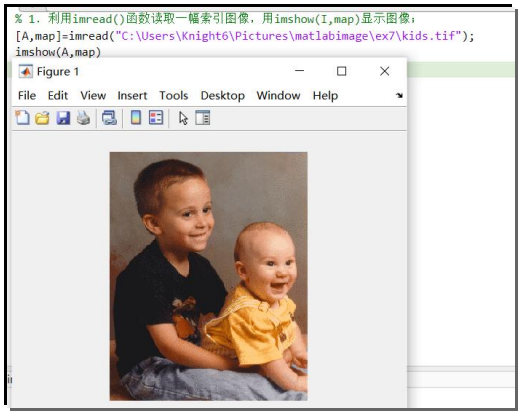
建立一个名为“xxxxx 实验 7”的解决方案（xxxxx 为自己的学号）

1. 利用 imread()函数读取一幅索引图像，用 imshow(I,map)显示图像；
2. 使用 ind2rgb、rgb2ind 命令实现彩色图像与索引图像的转换操作；
3. 使用 grayslice、gray2ind,ind2gray 命令实现索引图像与灰度图像的转换操作。
4. **参考课件，自定义彩色映射矩阵，自主编程实现灰度图像转换为索引图像**
5. 实现图像抖动效果的添加
6. 查看不同颜色模型各个分量显示效果
7. 参考教材，实现彩色图像的平滑处理
8. 参考教材，实现彩色图像的锐化处理
9. 参考教材，实现彩色图像的边缘检测及分割处理
10. **学习 ice 函数的使用，通过阅读 ice 函数的脚本尝试自主开发曲线调节函数。**

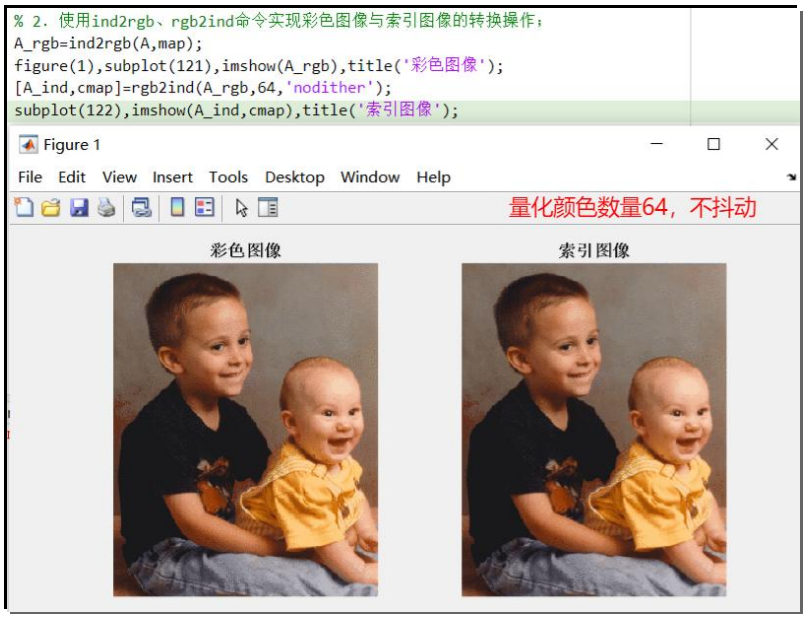
运行结果（写清题号）

描述实验的基本步骤，用数据和图片给出各个步骤中取得的实验结果和源代码，并进行必要的讨论，必须包括原始图像及其计算/处理后的图像。

1. 利用 imread()函数读取一幅索引图像，用 imshow(I,map)显示图像；

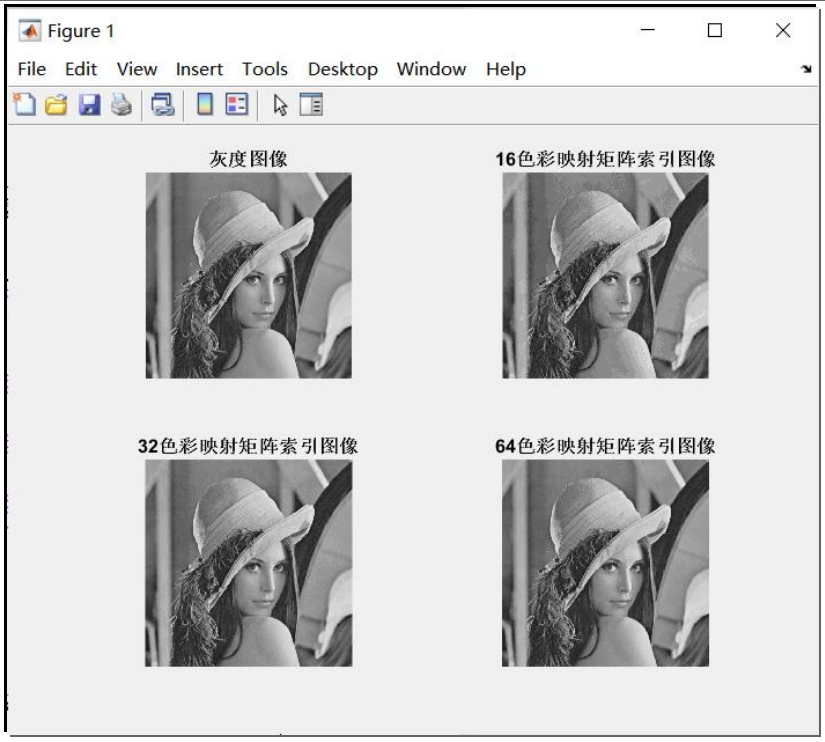


2. 使用 ind2rgb、rgb2ind 命令实现彩色图像与索引图像的转换操作；

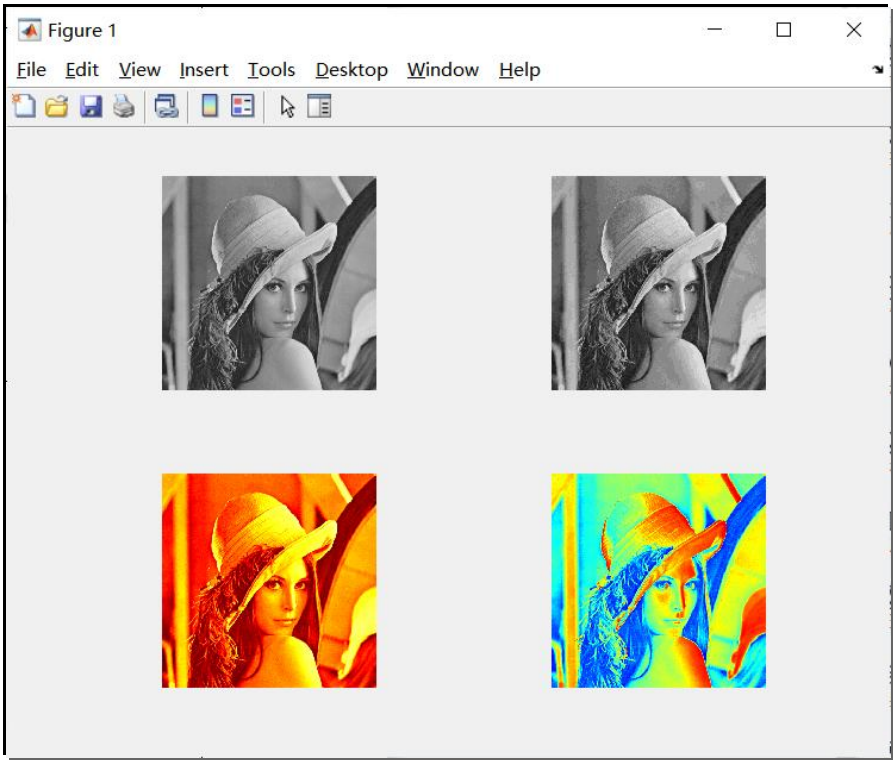


3. 使用 grayslice、gray2ind,ind2gray 命令实现索引图像与灰度图像的转换操作。

```
% gray2ind
A=imread("C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp");%灰度图像
figure(1),subplot(221),imshow(A),title('灰度图像');
[A_ind,map]=gray2ind(A,16);
subplot(222),imshow(A_ind,map),title('16色彩映射矩阵索引图像');
[A_ind,map]=gray2ind(A,32);
subplot(223),imshow(A_ind,map),title('32色彩映射矩阵索引图像');
[A_ind,map]=gray2ind(A,64);
subplot(224),imshow(A_ind,map),title('64色彩映射矩阵索引图像');
```




```
% grayslice
A=imread("C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp");%灰度图像
subplot(221),imshow(A);
A_sli16=grayslice(A,16);
subplot(222),imshow(A_sli16,gray(16));
A_sli32=grayslice(A,32);
subplot(223),imshow(A_sli32,hot(32));%hot颜色图数组
A_sli64=grayslice(A,64);
subplot(224),imshow(A_sli64,jet(64));%jet颜色图数组
```




```
% ind2gray
[A,map]=imread("C:\Users\Knight6\Pictures\matlabimage\ex7\kids.tif");
figure(2),subplot(121),imshow(A,map),title('索引图像');
A_gray=ind2gray(A,map);
figure(2),subplot(122),imshow(A_gray),title('灰度图像');
```

4. **参考课件，自定义彩色映射矩阵，自主编程实现灰度图像转换为索引图像**

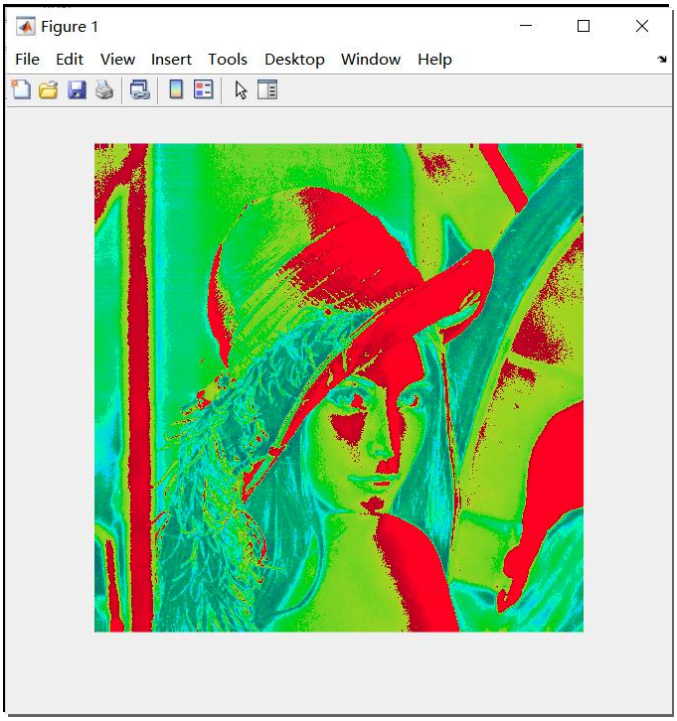
```
% 4. 参考课件，自定义彩色映射矩阵，自主编程实现灰度图像转换为索引图像
A=imread("C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp");
[hang,lie]=size(A);
B(:,:,1)=zeros(size(A));
B(:,:,2)=zeros(size(A));
B(:,:,3)=zeros(size(A));
for i=1:hang
    for j=1:lie
%         [R,G,B]=myfun(uint16(A(i,j)));%每行每行处理
    [B(i,j,1),B(i,j,2),B(i,j,3)]=myfun(uint16(A(i,j)));%每行每行处理
    end
end
%归一化处理，否则会很奇怪
[B(:,:,1),ps1]=mapminmax(B(:,:,1),0,1);
[B(:,:,2),ps2]=mapminmax(B(:,:,2),0,1);
[B(:,:,3),ps3]=mapminmax(B(:,:,3),0,1);
imshow(B);
```

```
function [R,G,B]=myfun(n)
R=0;
G=0;
B=0;
if n<64
    R=30;
    G=2*n+40;
    B=115;
elseif n<90
    R=30;
    G=2*n+40;
    B=510-4*n;
elseif n<128
    R=30;
    G=180;
    B=510-4*n;
elseif n<160
    R=2*n-150;
    G=180;
    B=36;
elseif n<192
    R=2*n-150;
    G=0;
    B=36;
else
    R=234;
    G=0;
    B=36;
end
```

$$R(x,y)=\begin{cases}30 & X(x,y)<128\\ 2*X(x,y)-150 & 128\le X(x,y)<192\\ 234 & X(x,y)\ge 192\end{cases}$$

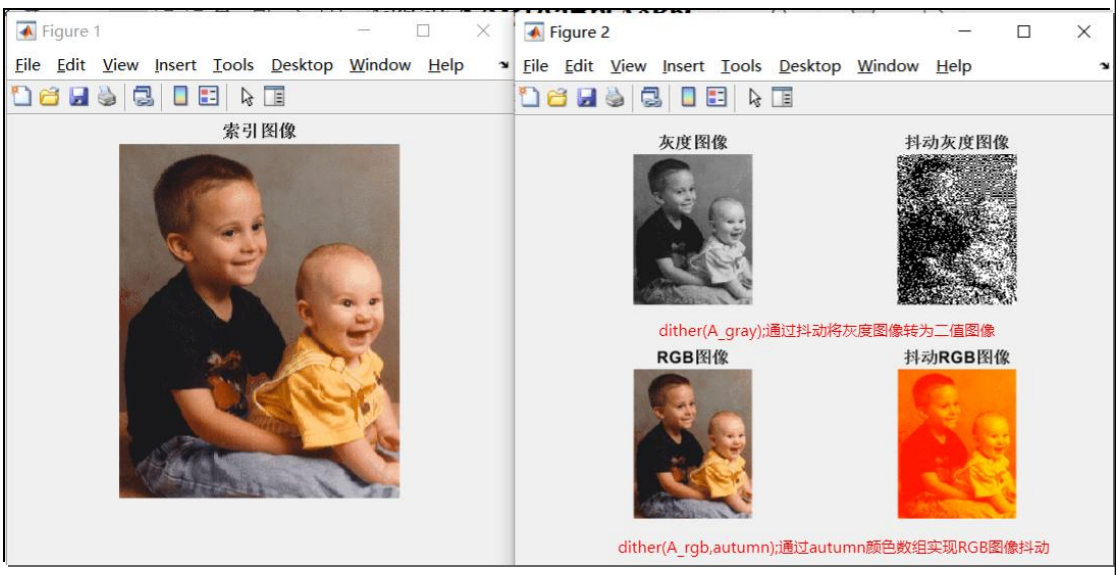
$$G(x,y)=\begin{cases}2*X(x,y)+40 & X(x,y)<90\\ 180 & 90\le X(x,y)<160\\ 0 & X(x,y)\ge 160\end{cases}$$

$$B(x,y)=\begin{cases}115 & X(x,y)<64\\ 510-4*X(x,y) & 64\le X(x,y)<128\\ 36 & X(x,y)\ge 128\end{cases}$$



5. 实现图像抖动效果的添加

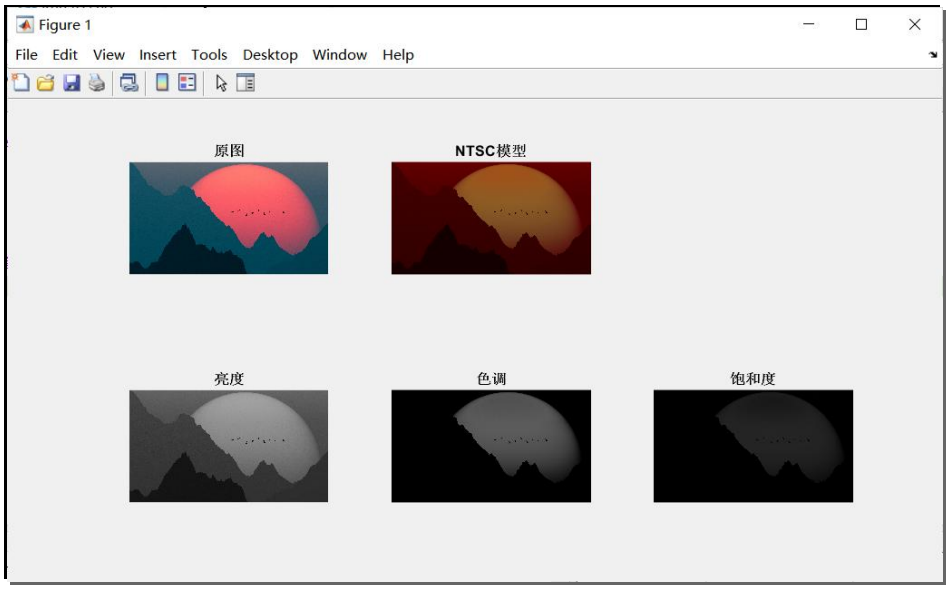
```
% 5. 实现图像抖动效果的添加
[A,map]=imread("C:\Users\Knight6\Pictures\matlabimage\ex7\kids.tif");
figure(1),imshow(A,map),title('索引图像');
A_gray=ind2gray(A,map);
figure(2),subplot(221),imshow(A_gray),title('灰度图像');
A_dithergray=dither(A_gray);
subplot(222),imshow(A_dithergray),title('抖动灰度图像');

A_rgb=ind2rgb(A,map);
subplot(223),imshow(A_rgb),title('RGB图像');
A_ditherrgb=dither(A_rgb,autumn);
subplot(224),imshow(A_ditherrgb,autumn),title('抖动RGB图像');
```



6. 查看不同颜色模型各个分量显示效果

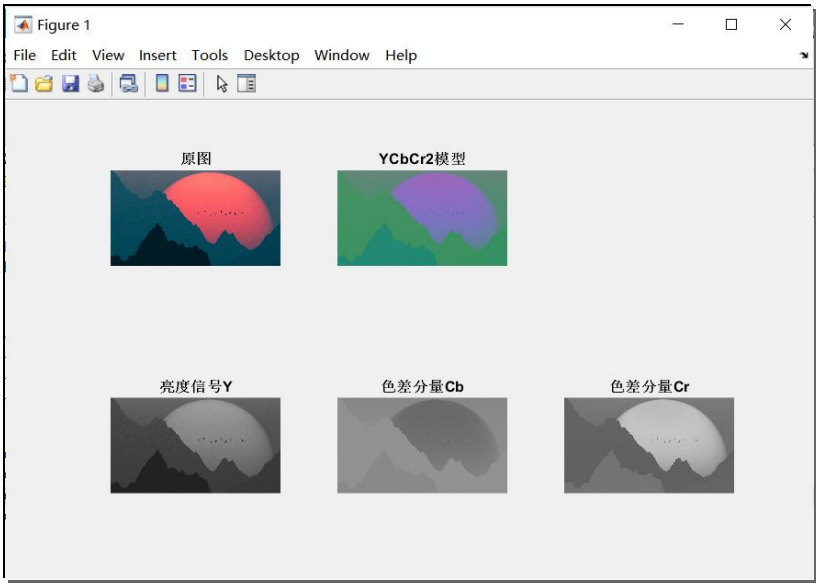
NTSC 模型

```
% 6. 查看不同颜色模型各个分量显示效果
A=imread("C:\Users\Knight6\Pictures\matlabimage\iTab-20220723.png");
figure,subplot(231),imshow(A),title('原图');
A_ntsc=rgb2ntsc(A);
subplot(232),imshow(A_ntsc),title('NTSC模型');
subplot(234),imshow(A_ntsc(:,:,1)),title('亮度');
subplot(235),imshow(A_ntsc(:,:,2)),title('色调');
subplot(236),imshow(A_ntsc(:,:,3)),title('饱和度');
```



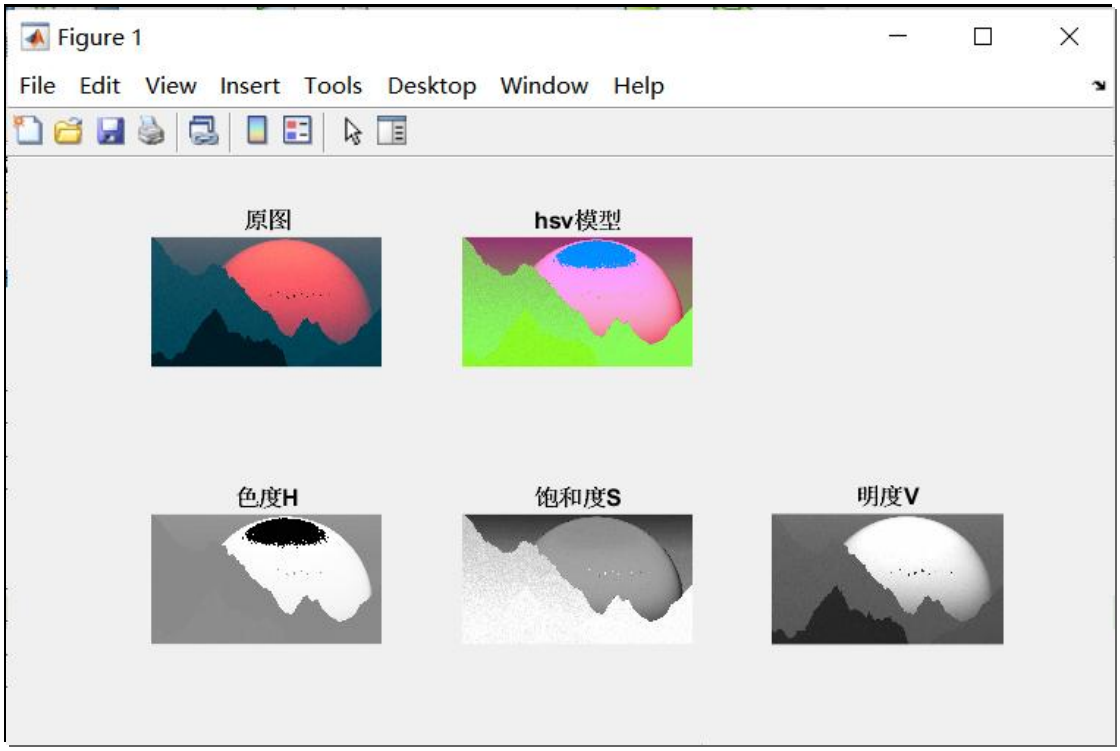
YCbCr2 模型

```
A_ycbcr=rgb2ycbcr(A);
subplot(232),imshow(A_ycbcr),title('YCbCr2模型');
subplot(234),imshow(A_ycbcr(:,:,1)),title('亮度信号Y');
subplot(235),imshow(A_ycbcr(:,:,2)),title('色差分量Cb');
subplot(236),imshow(A_ycbcr(:,:,3)),title('色差分量Cr');
```



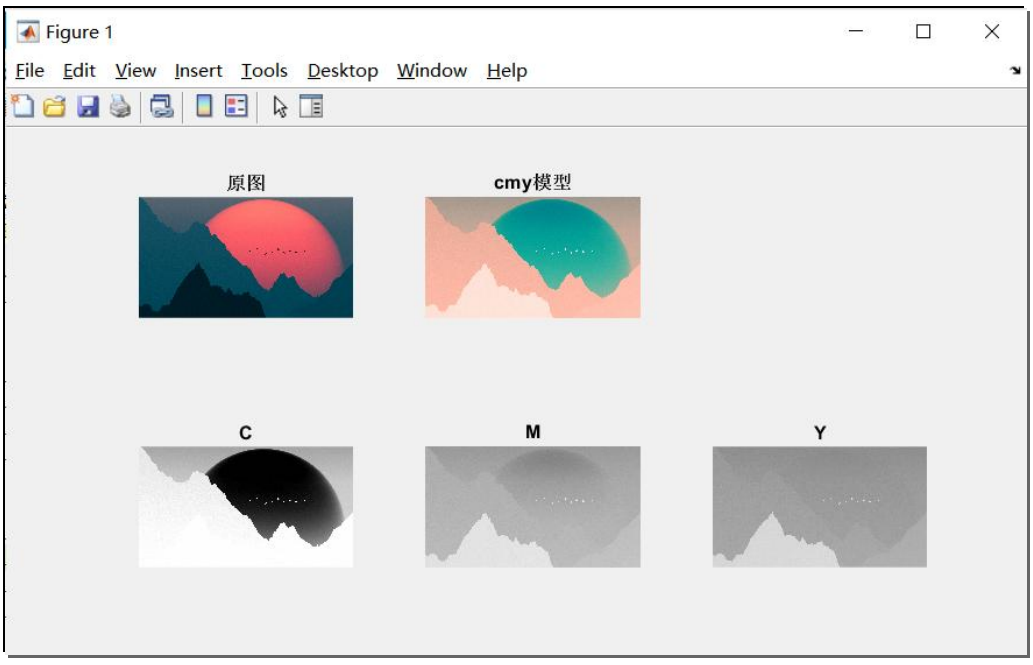
HSV 模型

```
%HSV
A_hsv=rgb2hsv(A);
subplot(232),imshow(A_hsv),title('hsv模型');
subplot(234),imshow(A_hsv(:,:,1)),title('色度H');
subplot(235),imshow(A_hsv(:,:,2)),title('饱和度S');
subplot(236),imshow(A_hsv(:,:,3)),title('明度V');
```



CMY 模型

```
%CMY
A_cmy=imcomplement(A);%计算图像A的补码，并以A_cmy为单位返回结果。
subplot(232),imshow(A_cmy),title('cmy模型');
subplot(234),imshow(A_cmy(:,:,1)),title('C');
subplot(235),imshow(A_cmy(:,:,2)),title('M');
subplot(236),imshow(A_cmy(:,:,3)),title('Y');
```

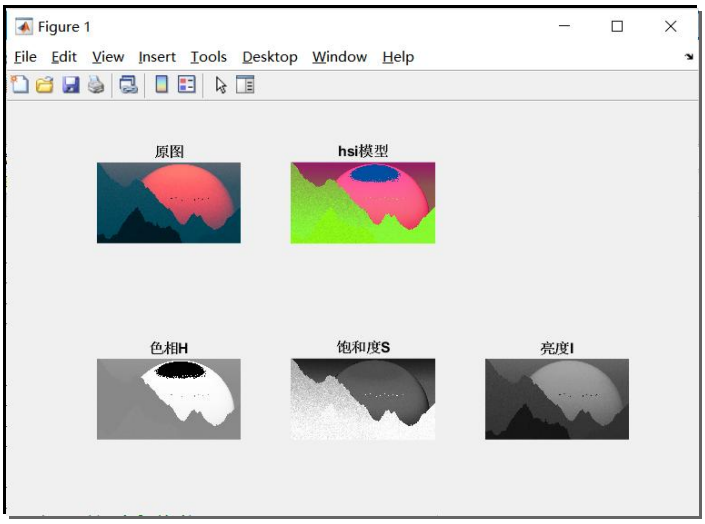


HSI 模型

函数 rgb2hsi 在新版的 matlab 中已被移除，我通过教材 P131 之前函数的代码复现了该函数。

```
%HSI
A_hsi=rgb2hsi(A);
subplot(232),imshow(A_hsi),title('hsi模型');
subplot(234),imshow(A_hsi(:,:,1)),title('色相H');
subplot(235),imshow(A_hsi(:,:,2)),title('饱和度S');
subplot(236),imshow(A_hsi(:,:,3)),title('亮度I');
```

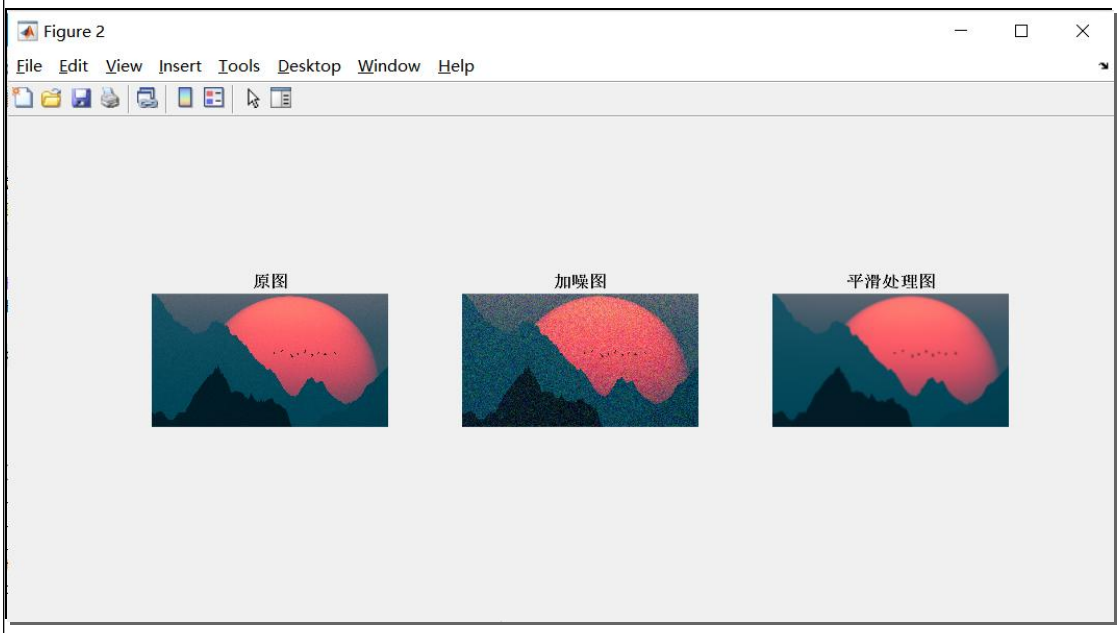
```
function hsi=rgb2hsi(rgb)
rgb=im2double(rgb);
r=rgb(:,:,1);
g=rgb(:,:,2);
b=rgb(:,:,3);

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;

hsi=cat(3,H,S,I);
```



7. 参考教材，实现彩色图像的平滑处理

先对图像加噪，再进行平滑进行效果比较。

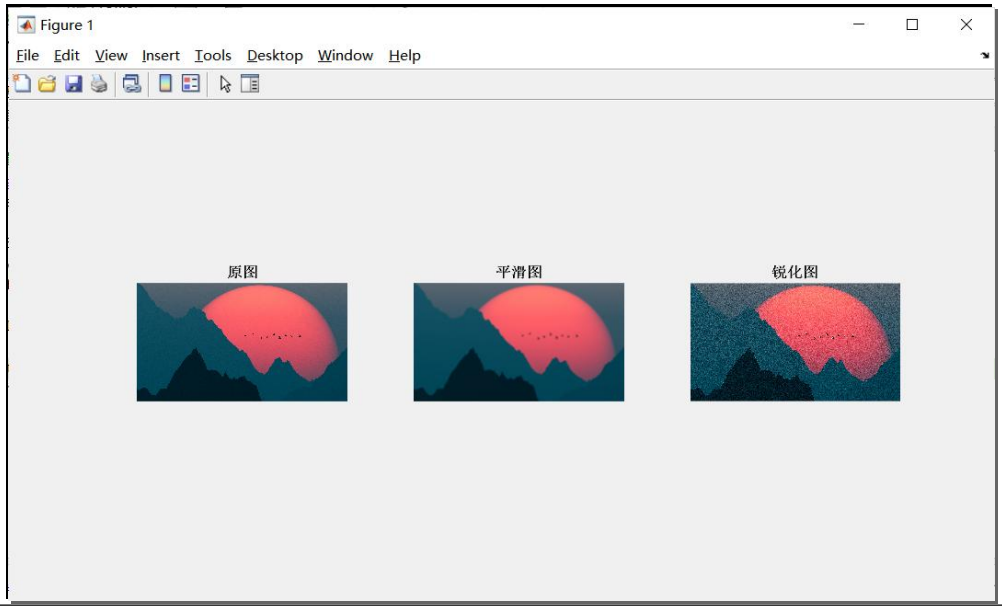
```
% 7. 参考教材，实现彩色图像的平滑处理
A=imread("C:\Users\Knight6\Pictures\matlabimage\iTab-20220723.png");
figure,subplot(131),imshow(A),title('原图');
A_noise=imnoise(A);
subplot(132),imshow(A_noise),title('加噪图');

fR=A(:,:,1);
fG=A(:,:,2);
fB=A(:,:,3);
w=fspecial('average',25);%均值平滑滤波器
fR_filtered=imfilter(fR,w,'replicate');
fG_filtered=imfilter(fG,w,'replicate');
fB_filtered=imfilter(fB,w,'replicate');
A_filtered=cat(3,fR_filtered,fG_filtered,fB_filtered);
subplot(133),imshow(A_filtered),title('平滑处理图');
```



8. 参考教材，实现彩色图像的锐化处理

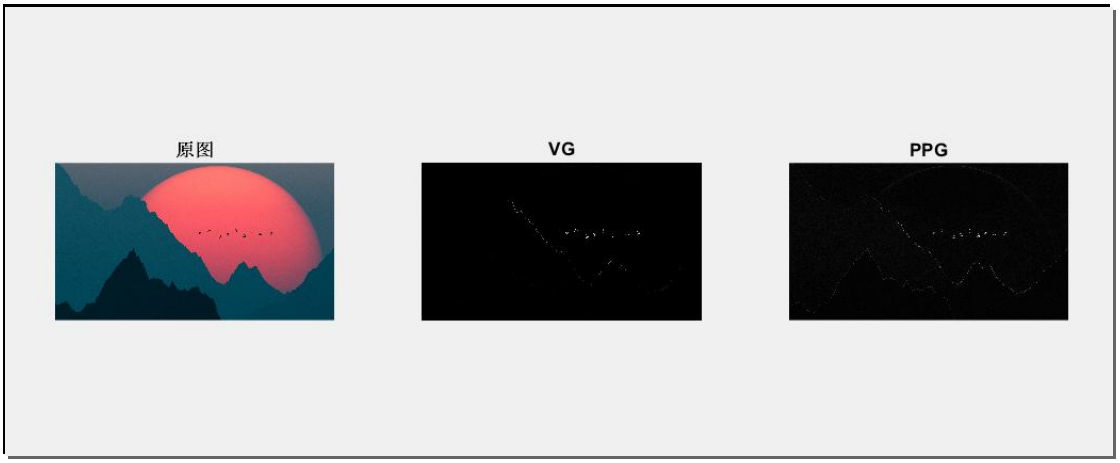
锐化和平滑相似，也是先对三个通道锐化后再 cat，此处先对图像平滑处理后，再锐化以便凸显锐化的效果。由结果可看出太阳及山峰的轮廓更加清晰明显了。

```
% 8. 参考教材，实现彩色图像的锐化处理
A=imread("C:\Users\Knight6\Pictures\matlabimage\iTab-20220723.png");
figure,subplot(131),imshow(A),title('原图');

w=fspecial('average',25);
A_filtered=imfilter(A,w,'replicate');
subplot(132),imshow(A_filtered),title('平滑图');

lapmask=[1 1 1;1 -8 1;1 1 1];%拉普拉斯算子
A=double(A);
A_lap=A-imfilter(A,lapmask,'replicate');
subplot(133),imshow(uint8(A_lap)),title('锐化图');
```



9. 参考教材，实现彩色图像的边缘检测及分割处理

彩色边缘检测自定义函数 colorgrad 在教材 P337。

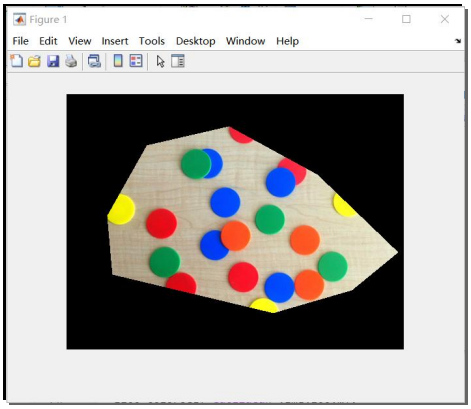


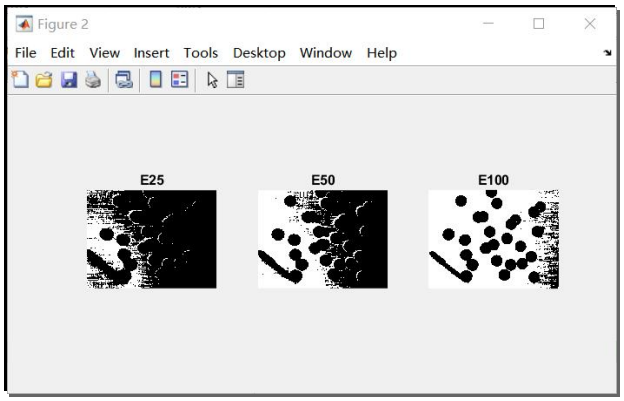

```
% 9. 参考教材，实现彩色图像的边缘检测及分割处理
img=imread("C:\Users\Knight6\Pictures\matlabimage\iTab-20220723.png");
figure,subplot(131),imshow(img),title('原图');
[VG,A,PPG]=colorgrad(img);
subplot(132),imshow(VG),title('VG');
subplot(133),imshow(PPG),title('PPG');
```

```
function [VG,A,PPG]=colorgrad(f,T)
if (ndims(f)~=3)||(size(f,3)~=3)
    error('Input image must be RGB.');
end

sh=fspecial('sobel');
sv=sh';
Rx=imfilter(double(f(:,:,1)),sh,'replicate');
Ry=imfilter(double(f(:,:,1)),sv,'replicate');
Gx=imfilter(double(f(:,:,2)),sh,'replicate');
Gy=imfilter(double(f(:,:,2)),sv,'replicate');
Bx=imfilter(double(f(:,:,3)),sh,'replicate');
By=imfilter(double(f(:,:,3)),sv,'replicate');

gxx=Rx.^2+Gx.^2+Bx.^2;
gyy=Ry.^2+Gy.^2+By.^2;
gxy=Rx.*Ry+Gx.*Gy+Bx.*By;
A=0.5*(atan(2*gxy./(gxx-gyy+eps)));
G1=0.5*((gxx+gyy)+(gxx-gyy).*cos(2*A)+2*gxy.*sin(2*A));

A=A+pi/2;
G2=0.5*((gxx+gyy)+(gxx-gyy).*cos(2*A)+2*gxy.*sin(2*A));
G1=G1.*0.5;
G2=G2.*0.5;
VG=mat2gray(max(G1,G2));

RG=sqrt(Rx.^2+Ry.^2);
GG=sqrt(Gx.^2+Gy.^2);
BG=sqrt(Bx.^2+By.^2);
PPG=mat2gray(RG+GG+BG);

if nargin==2
    VG=(VG>T).*VG;
    PPG=(PPG>T).*PPG;
end
```

分割处理（掩模需手动截取）

covmatrix 位于教材 P286，colorseg 位于 P338，imstack2vectors 位于 P356。Colorseg 阈值取不同值，分割效果也不同，当取到 100 时，产生了明显的过分割。

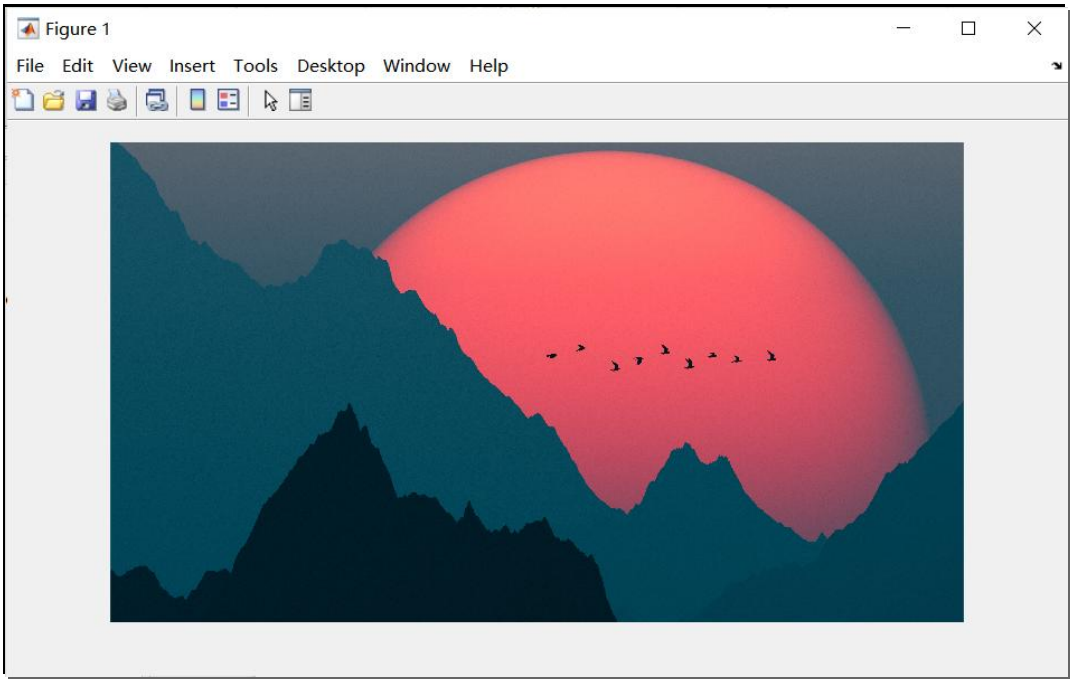


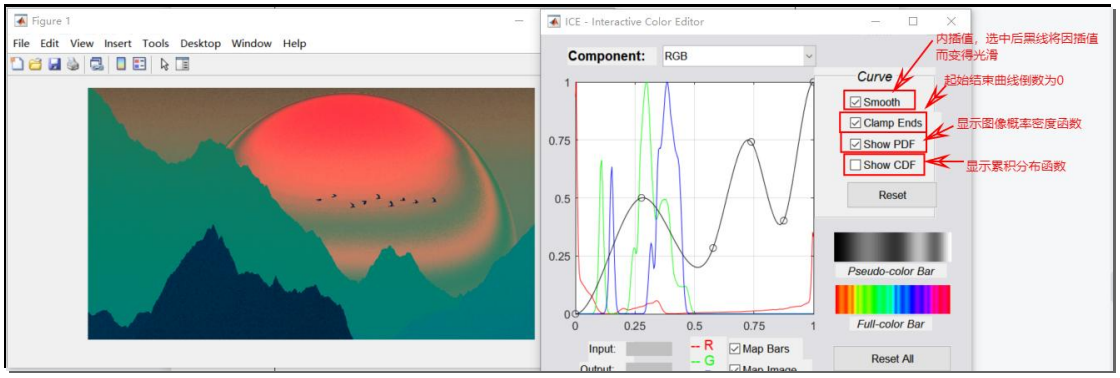



```
%分割处理
mask=roipoly(img);
red=immultiply(mask,img(:,:,1));
green=immultiply(mask,img(:,:,2));
blue=immultiply(mask,img(:,:,3));
g=cat(3,red,green,blue);
imshow(g);

[M,N,K]=size(g);
I=reshape(g,M*N,3);
idx=find(mask);
I=double(I(idx,1:3));
[C,m]=covmatrix(I);
d=diag(C);
sd=sqrt(d);
E25=colorseg('euclidean',img,25,m);
E50=colorseg('euclidean',img,50,m);
E100=colorseg('euclidean',img,100,m);
figure,subplot(131),imshow(E25),title('E25');
subplot(132),imshow(E50),title('E50');
subplot(133),imshow(E100),title('E100');
```

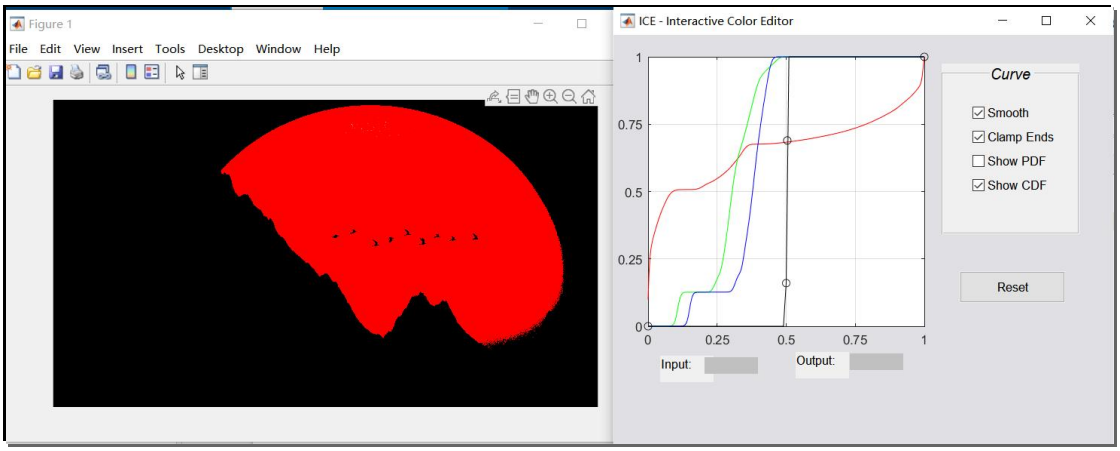
10. **学习 ice 函数的使用，通过阅读 ice 函数的脚本尝试自主开发曲线调节函数。**

Ice 函数：





自主开发曲线调节函数（由于时间问题，只是在阅读源码的基础上进行了简单的修改优化）



讨论：

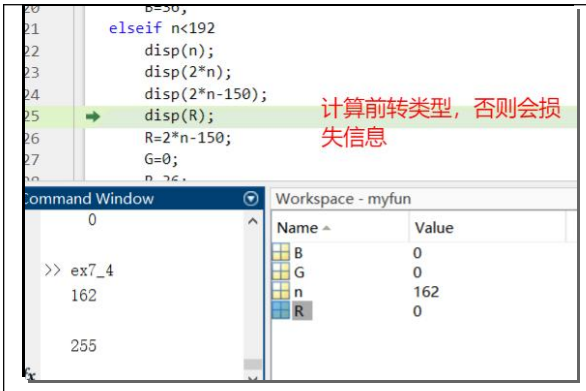
①第 4 题开始设计时想用 .* 直接对矩阵成行操作，而忽视了函数内部的比较方式：逐个元素比较。如果传矩阵过去，则是用矩阵的大小比较，与设计初衷相悖。这也解释了为什么一开始 R,G,B 各自传回来都是一个值的原因。后改成逐个元素传参。

```
map(:,:,3)=zeros(size(A));
for i=1:hang
    [map(i,:,1),map(i,:,2),map(i,:,3)]=myfun(A(i,:));%每行每行处理
end
```

```
function [R,G,B]=myfun(n)
R=zeros(size(n));
G=zeros(size(n));
B=zeros(size(n));
if n<64
    R=30;
    G=2.*n+40;
    B=115;
elseif n<90
    R=30;
    G=2.*n+40;
    B=510-4.*n;
```

```
ans =

  Columns 1 through 20

   234   234   234   234   234   234   234   234   234   234   234   234   234   234   234   234

  Columns 21 through 40

   234   234   234   234   234   234   234   234   234   234   234   234   234   234   234   234
```

②别忘了转类型（uint16/double）



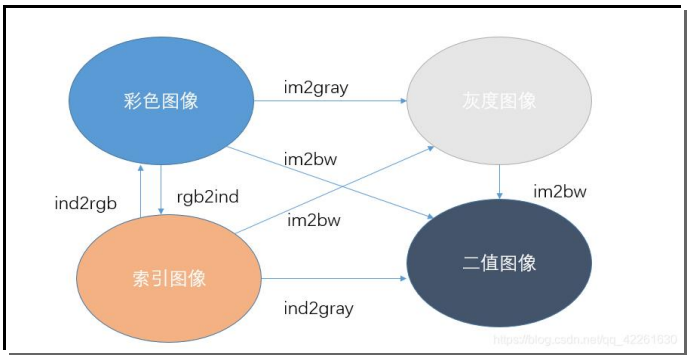
实验的体会与思考题

1. 彩色图像、灰度图像、索引图像各自的存储特点？

彩色图像：每个像素由 R、G、B 三个分量表示，每个通道取值范围 0~255。数据类型一般为 8 位无符号整形。

灰度图像：每个像素只有一个采样颜色的图像，单通道，通常显示为从最暗黑色到最亮的白色的灰度。

索引图像：文件结构比较复杂，除了存放图像的二维矩阵外，还包括一个称之为颜色索引矩阵 MAP 的二维数组。类似于查字典，主要是为了解决彩色图像消耗空间大的问题，一般应用于色彩构成比较简单的场景。



2. 讨论真彩色增强、伪彩色增强的异同。

伪彩色增强是从灰度到彩色的映射，是为了将人眼难以区分的灰度差异变得更加明显、易于区分。而真彩色增强中原图像本身有颜色，只不过是对原始图像本身的颜色调节，是彩色到彩色的映射过程。