R05631027_楊皓文 _影像處理 _作業 #5

編譯環境 : Windows 10 \ Matlab 2016

Part 1: (40%) Color Model Conversion

程式碼 :

```
function varargout = HW01(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @HW01_OpeningFcn, ...
                   'gui_OutputFcn',  @HW01_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


function HW01_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;
guidata(hObject, handles);


function varargout = HW01_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


% Å¤¨ú¹Ï¤ù
function pushbutton1_Callback(hObject, eventdata, handles)

filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image
Files'});
handles.I = imread(filename);
```

```
handles.RGB_R = handles.I(:, :, 1);
handles.RGB_G = handles.I(:, :, 2);
handles.RGB_B = handles.I(:, :, 3);

axes(handles.axes1); imshow(handles.I);

handles.output = hObject;
guidata(hObject, handles);


% RGB
function pushbutton2_Callback(hObject, eventdata, handles)

handles.I_RGB = cat(3, handles.RGB_R, handles.RGB_G, handles.RGB_B);
axes(handles.axes2); imshow(handles.I_RGB);
axes(handles.axes3); imshow(handles.RGB_R);
axes(handles.axes4); imshow(handles.RGB_G);
axes(handles.axes5); imshow(handles.RGB_B);

handles.output = hObject;
guidata(hObject, handles);


% CMY
function pushbutton3_Callback(hObject, eventdata, handles)

handles.CMY_C = 255 - handles.RGB_R;
handles.CMY_M = 255 - handles.RGB_G;
handles.CMY_Y = 255 - handles.RGB_B;
handles.I_CMY = cat(3, handles.CMY_C, handles.CMY_M, handles.CMY_Y);

axes(handles.axes2); imshow(handles.I_CMY);
axes(handles.axes3); imshow(handles.CMY_C);
axes(handles.axes4); imshow(handles.CMY_M);
axes(handles.axes5); imshow(handles.CMY_Y);

handles.output = hObject;
guidata(hObject, handles);


% HSI
function pushbutton4_Callback(hObject, eventdata, handles)
```

```
[row, col, ~] = size(handles.I);

for i = 1:row;
    for j = 1:col;
        r = double(handles.RGB_R(i, j))/255;
        g = double(handles.RGB_G(i, j))/255;
        b = double(handles.RGB_B(i, j))/255;
        theta = acosd( (1/2)*( (r-g) + (r-b) )/( (r-g)^2 + (r-b)*(g-
b) )^(1/2) );

        if b <= g
            H(i, j) = theta;
        else
            H(i, j) = 360 - theta;
        end

        S(i, j) = 1 - 3/(r + g + b)*(min([r, g, b]));
        I(i, j) = 1/3*(r + g + b);
    end
end

% Hue¤£¬O¦Ç¶¥¡A¬O¨¤«×¡A¤£¥Î-¼¥H255
H = uint8(H);
S = uint8(S*255);
I = uint8(I*255);

I_HSI = cat(3, H, S, I);

axes(handles.axes2); imshow(I_HSI);
axes(handles.axes3); imshow(H);
axes(handles.axes4); imshow(S);
axes(handles.axes5); imshow(I);

handles.output = hObject;
guidata(hObject, handles);



% XYZ
function pushbutton5_Callback(hObject, eventdata, handles)

handles.XYZ_X = 0.412453 * handles.RGB_R + 0.357580 * handles.RGB_G +
```

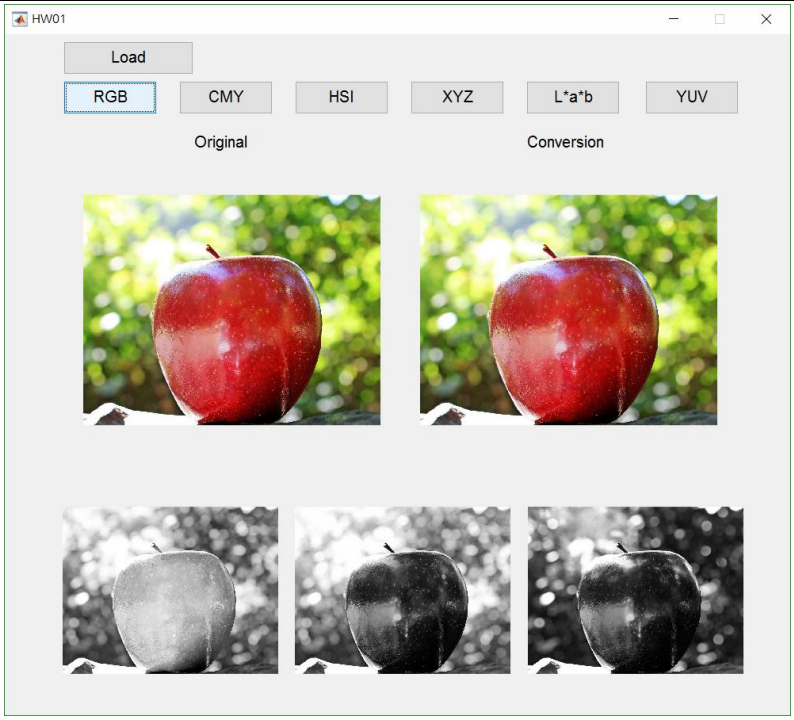
```
0.180423 * handles.RGB_B;
handles.XYZ_Y = 0.212671 * handles.RGB_R + 0.715160 * handles.RGB_G +
0.072169 * handles.RGB_B;
handles.XYZ_Z = 0.019334 * handles.RGB_R + 0.119193 * handles.RGB_G +
0.950227 * handles.RGB_B;

handles.I_XYZ = cat(3, handles.XYZ_X, handles.XYZ_Y, handles.XYZ_Z);
axes(handles.axes2); imshow(handles.I_XYZ);
axes(handles.axes3); imshow(handles.XYZ_X);
axes(handles.axes4); imshow(handles.XYZ_Y);
axes(handles.axes5); imshow(handles.XYZ_Z);

handles.output = hObject;
guidata(hObject, handles);


% L*a*b
function pushbutton6_Callback(hObject, eventdata, handles)

X = double(0.412453 * handles.RGB_R + 0.357580 * handles.RGB_G +
0.180423 * handles.RGB_B);
Y = double(0.212671 * handles.RGB_R + 0.715160 * handles.RGB_G +
0.072169 * handles.RGB_B);
Z = double(0.019334 * handles.RGB_R + 0.119193 * handles.RGB_G +
0.950227 * handles.RGB_B);

[row, col, ~] = size(handles.I);

xn = 0.9515;
yn = 1;
zn = 1.0886;

for i = 1:row
    for j = 1:col
        if Y(i,j)/yn > 0.008856;
            L(i,j) = 116*((Y(i,j)/yn)^(1/3)) - 16;
        else
            L(i,j) = 903.3*Y(i,j)/yn;
        end

        if X(i,j)/xn > 0.008856 ;
            fx = (X(i,j)/xn)^(1/3);
```

```
        else
            fx = 7.787*(X(i,j)/xn) + 16/116;
        end

        if Y(i,j)/yn > 0.008856 ;
            fy=(Y(i,j)/yn)^(1/3);
        else
            fy = 7.787*(Y(i,j)/yn) + 16/116;
        end

        if Z(i,j)/zn > 0.008856 ;
            fz = (Z(i,j)/zn)^(1/3);
        else
            fz = 7.787*(Z(i,j)/zn) + 16/116;
        end

        a(i,j) = 500*(fx-fy);
        b(i,j) = 200*(fy-fz);
    end
end

I_LAB = cat(3, L, a, b);
axes(handles.axes2); imshow(I_LAB);
axes(handles.axes3); imshow(L);
axes(handles.axes4); imshow(a);
axes(handles.axes5); imshow(b);

handles.output = hObject;
guidata(hObject, handles);


% YUV
function pushbutton7_Callback(hObject, eventdata, handles)

handles.YUV_Y = 0.299 * handles.RGB_R + 0.587 * handles.RGB_G + 0.114
* handles.RGB_B;
handles.YUV_U = 128 - 0.168736 * handles.RGB_R - 0.331264 *
handles.RGB_G + 0.5 * handles.RGB_B;
handles.YUV_V = 128 + 0.5 * handles.RGB_R - 0.418688 * handles.RGB_G
- 0.081312 * handles.RGB_B;
handles.I_YUV = cat(3, handles.YUV_Y, handles.YUV_U, handles.YUV_V);
```

```
axes(handles.axes2); imshow(handles.I_YUV);
axes(handles.axes3); imshow(handles.YUV_Y);
axes(handles.axes4); imshow(handles.YUV_U);
axes(handles.axes5); imshow(handles.YUV_V);

handles.output = hObject;
guidata(hObject, handles);
```

結果呈現與討論 ：

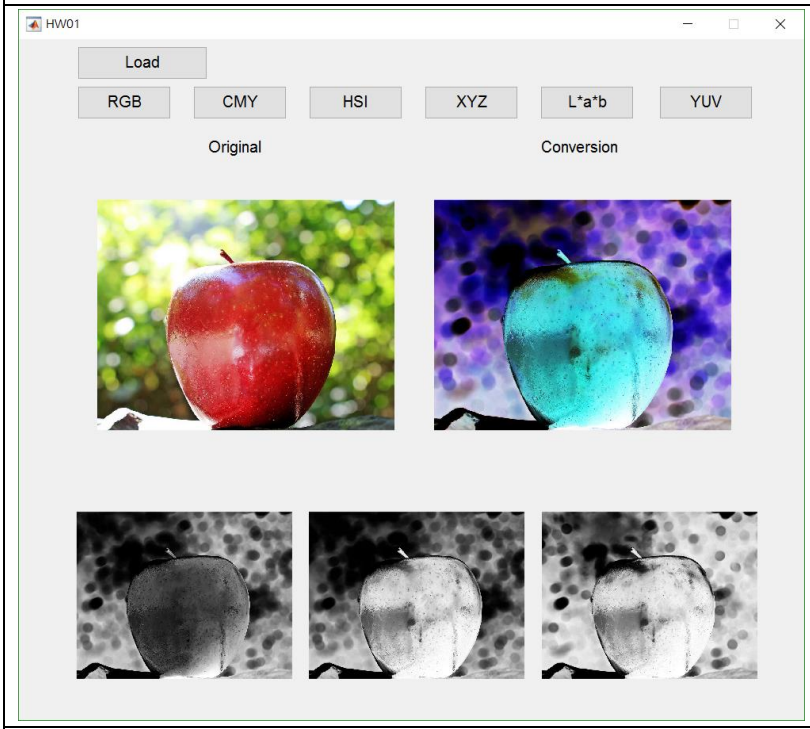
RGB



結果討論:

可以看出 RGB 出來的 R、G、B 影像，還是可以明顯看出蘋果的形狀與光澤。另一方面，也可以看出色彩分布的強度，像是 R 影像在蘋果的部分就是白色，而 G 和 B 影像在蘋果的部分則較黑，代表該區域主要屬於紅色。

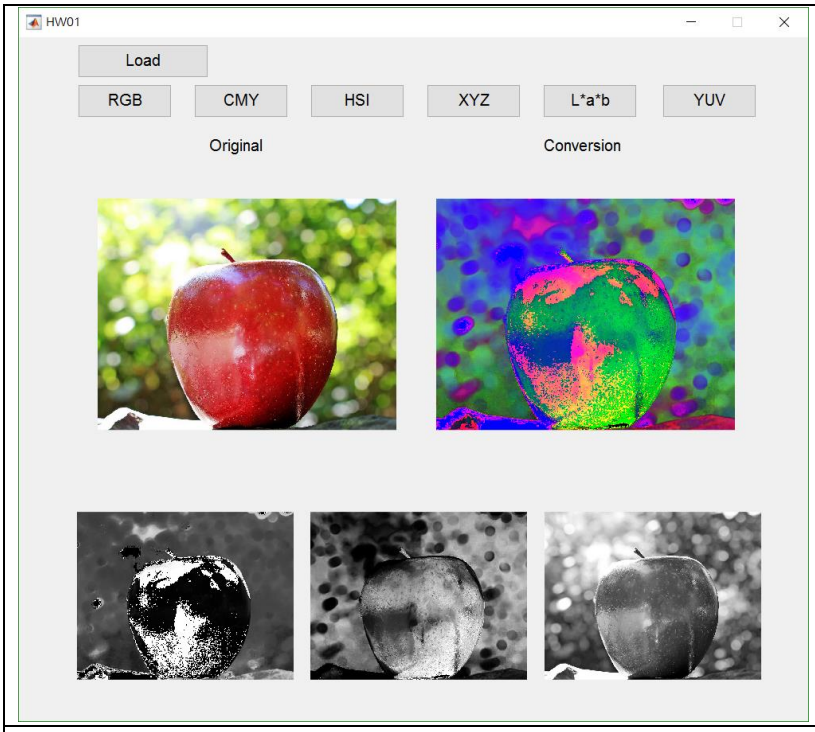
CMY



結果討論:

CMY 為 RGB 的互補色彩，因此所呈現出來的效果會有負片的感覺。另一方面，也可以看出色彩分布的強度，像是 R 影像在蘋果的部分就是黑色，而 G 和 B 影像在蘋果的部分則較白，代表該區域主要屬於紅色。

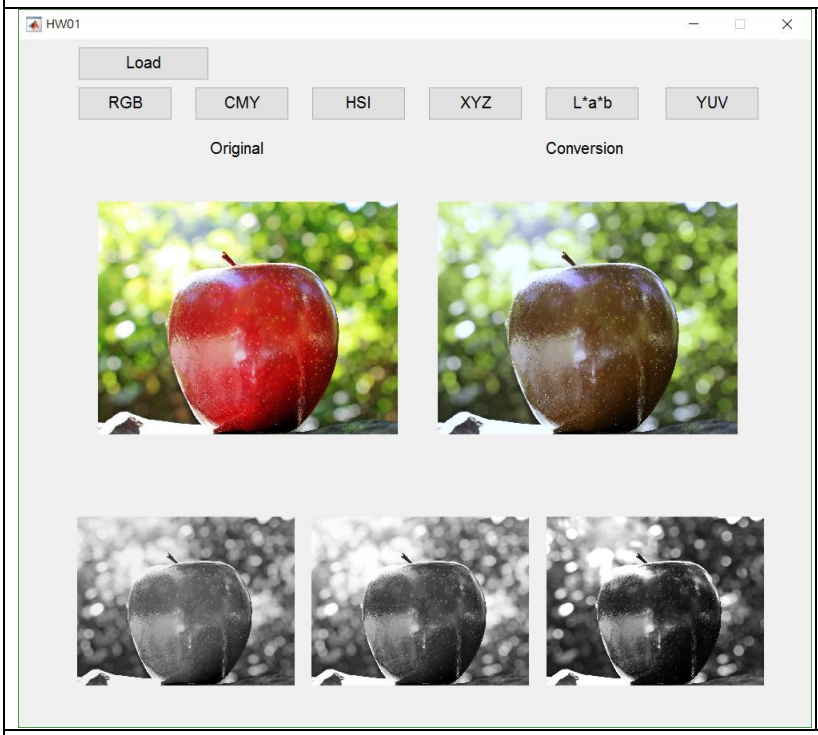
HSI



結果討論:

HSI 色彩空間是從人的視覺系統出發，用色調(Hue)、色飽和度(Saturation)和亮度(Intensity)來描述色彩。因此由左圖的結果可觀察，H 描述色調，單位為角度從紅色開始，因此蘋果紅色部分為黑色。S 飽和度可以看到像在後面的亮光，由於較不飽和，因此呈現黑色區塊。最後 I 亮度就有點類似灰階的效果。

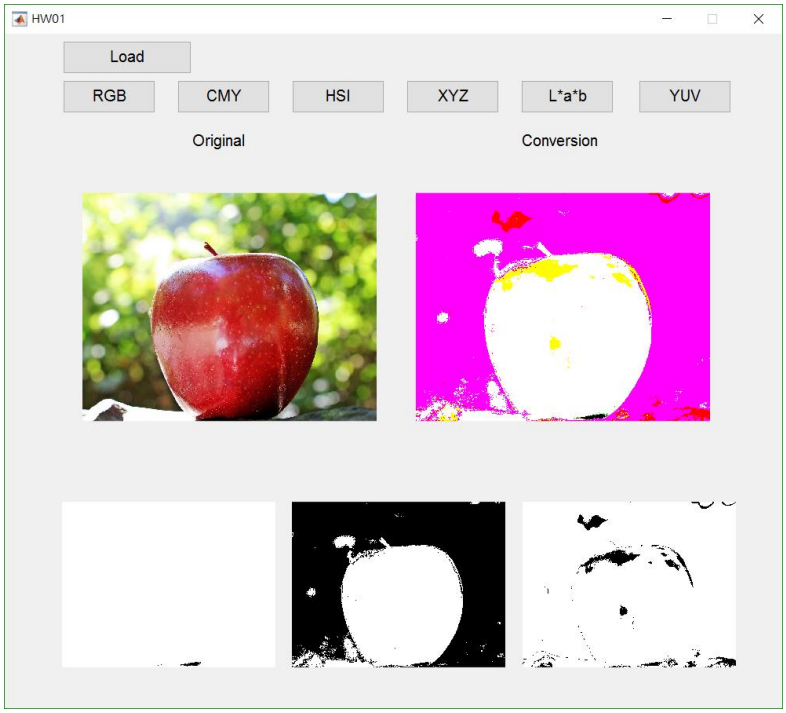
XYZ



結果討論:

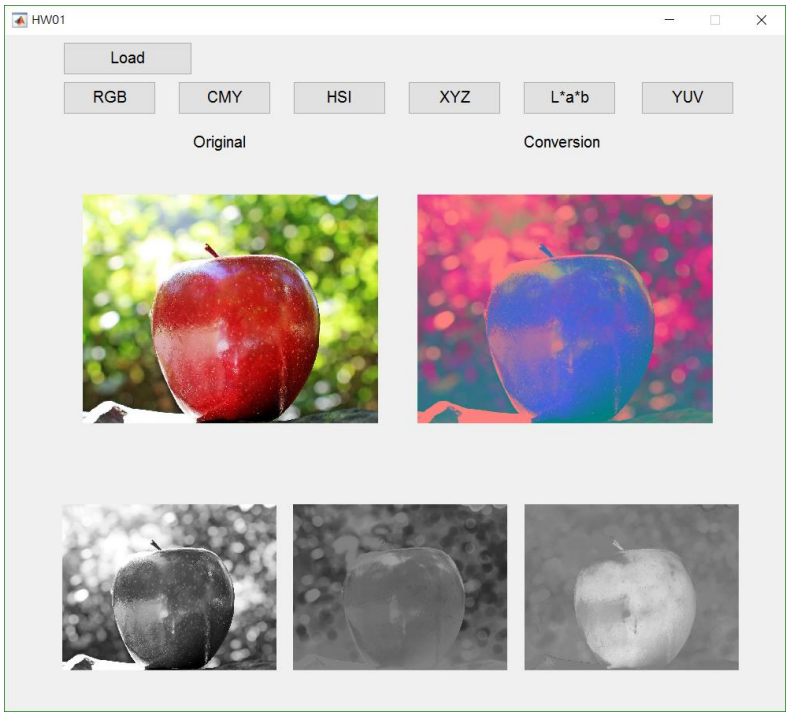
XYZ 顏色空間具有獨特的特性，可以表達人類眼睛所能看到的所有顏色。由轉換公式也可以了解，為一個線性轉換矩陣，因此 XYZ 都是 RGB 的線性組合，所呈現的效果也和 RGB 類似，但其色彩可以表示的幅度更廣。

L*a*b



結果討論:
L*a*b 這種模式是以數字化方式來描述人的視覺感應，因此由結果可以看出漢人眼第一眼看到影像的輪廓類似，藉由這個特性可以推演出此顏色空間或許很適合進行前景分離。

YUV



結果討論:
採用 YUV 色彩空間的重要性是它的亮度信號 Y 和色度信號 U、V 是分離的。如果只有 Y 信號分量而沒有 U、V 分量， 那麼這樣表示的圖就是黑白灰度圖，從結果圖可以觀察到此特性。彩色電視採用 YUV 空間正是為了用亮度信號 Y 解決彩色電視機與黑白電視機 的兼容問題，使黑白電視機也能接收彩色信號。

Part 2: (30%) Pseudo-color Image

程式碼 ：

```
function varargout = HW02(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @HW02_OpeningFcn, ...
                   'gui_OutputFcn',  @HW02_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
```

```
                'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


function HW02_OpeningFcn(hObject, eventdata, handles, varargin)

% ³]©wªì©lªº¦â½Õ¤@¦â½Õ¤G
set(handles.edit1, 'String', 255);
set(handles.edit2, 'String', 255);
set(handles.edit4, 'String', 0);
set(handles.edit5, 'String', 0);
set(handles.edit6, 'String', 0);
set(handles.edit7, 'String', 255);

handles.output = hObject;
guidata(hObject, handles);


function varargout = HW02_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


function pushbutton1_Callback(hObject, eventdata, handles)

% Åª¨ú¹ÏÀɤù
filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image
Files'});
handles.I = imread(filename);

% Âà¦¨¦Ç¶¥
[handles.row, handles.col, handles.lay] = size(handles.I);

if handles.lay == 1
```

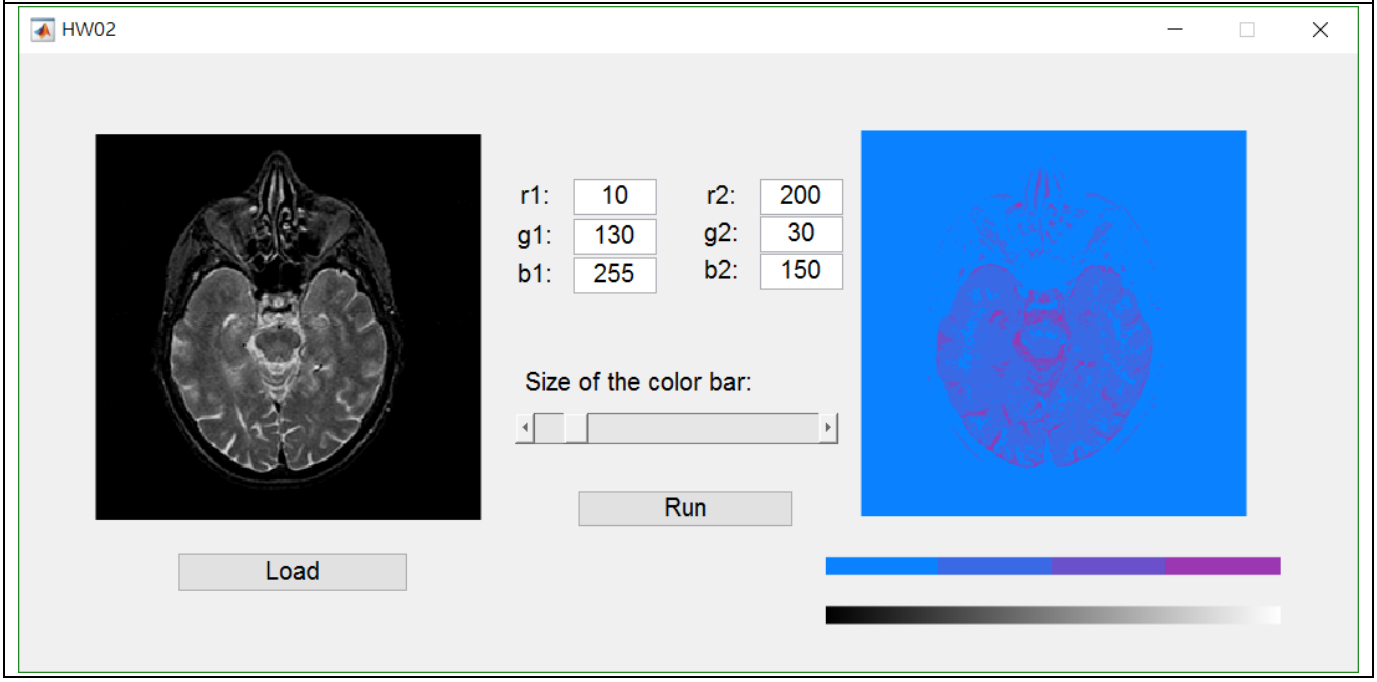
```
    handles.I = handles.I;
    fprintf('1 layers image\n');
elseif handles.lay == 3
    handles.R = uint8(handles.I(:, :, 1));
    handles.G = uint8(handles.I(:, :, 2));
    handles.B = uint8(handles.I(:, :, 3));
    handles.I = 0.299*handles.R + 0.587*handles.G + 0.114*handles.B;
    fprintf('3 layers image\n');
end

axes(handles.axes1); imshow(handles.I);

handles.output = hObject;
guidata(hObject, handles);


function edit1_Callback(hObject, eventdata, handles)

handles.output = hObject;
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit2_Callback(hObject, eventdata, handles)


function edit2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

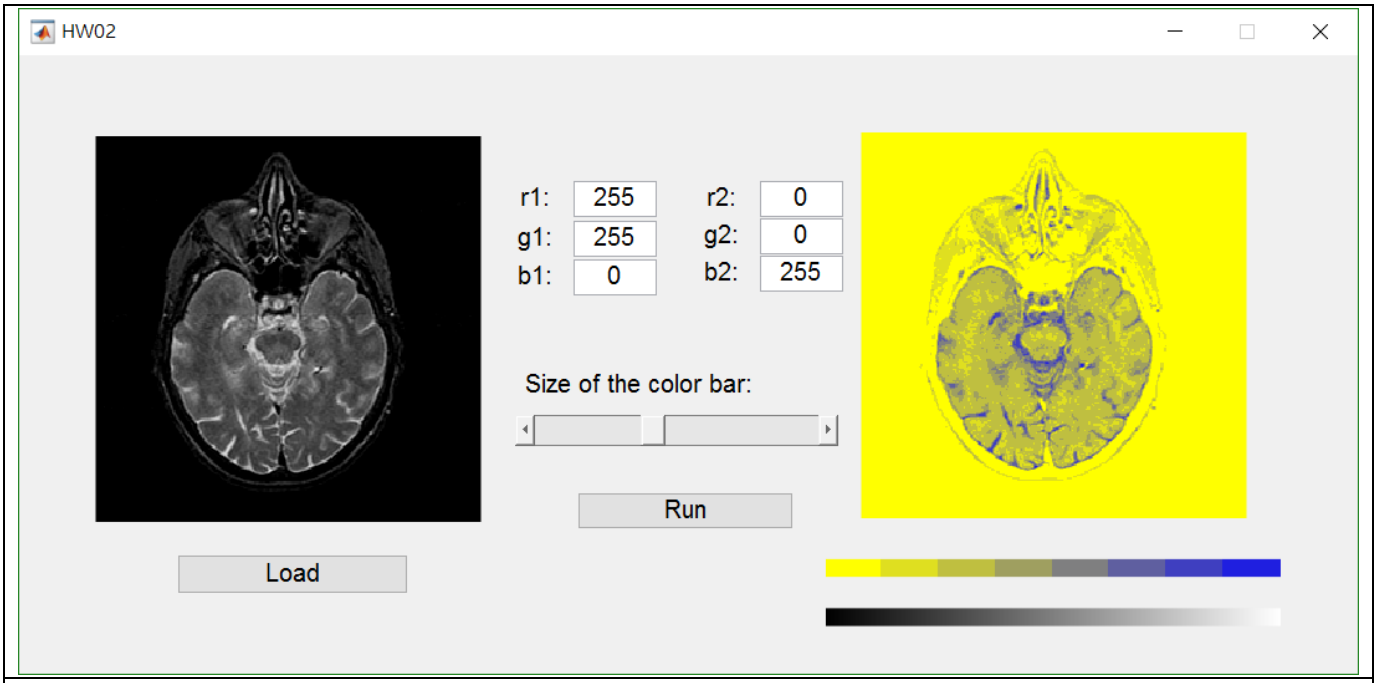
```
function edit4_Callback(hObject, eventdata, handles)


function edit4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit5_Callback(hObject, eventdata, handles)


function edit5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit6_Callback(hObject, eventdata, handles)


function edit6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit7_Callback(hObject, eventdata, handles)


function edit7_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


function slider1_Callback(hObject, eventdata, handles)


function slider1_CreateFcn(hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


function pushbutton2_Callback(hObject, eventdata, handles)

r1 = str2num( get(handles.edit1, 'String') );
g1 = str2num( get(handles.edit2, 'String') );
b1 = str2num( get(handles.edit4, 'String') );
r2 = str2num( get(handles.edit5, 'String') );
g2 = str2num( get(handles.edit6, 'String') );
b2 = str2num( get(handles.edit7, 'String') );
num = uint8(get(handles.slider1, 'Value'));

num = double(2^num);
fprintf('num = %d\n', num);

I = handles.I;
[row, col, ~]=size(I);
% num = 2;
bar = zeros(10, 256);
bar2 = zeros(10, 256);
for i = 1:256
    bar2(:,i) = i-1;
end
% r1 = 255; g1 = 0; b1 = 255;
% r2 = 0; g2 = 255; b2 = 255;
% rgbª°®t-È
R = round((r2 - r1)/num);
G = round((g2 - g1)/num);
B = round((b2 - b1)/num);
```

```
for n = 1:num
    bound = n*round(255/num);
    fprintf('bound = %d\n', bound);
    for i = 1:row
        for j = 1:col
            % bound:intensityªº¤WÃä¬É¡Around(255/num):intensity¶¡ªº¶ZÂ÷
            if I(i, j) >= bound-round(255/num) && I(i, j) <= bound
                % ¦â½Õ¤@³vº¥¥[¤Wrgbªº®t-È¡AÅÜ¦¨â½Õ¤G
                x(i, j, 1) = r1 + (n-1)*R;
                x(i, j, 2) = g1 + (n-1)*G;
                x(i, j, 3) = b1 + (n-1)*B;
            end
        end
    end
    bar(:, bound-255/num:bound+1, 1) = r1 + (n-1)*R;
    bar(:, bound-255/num:bound+1, 2) = g1 + (n-1)*G;
    bar(:, bound-255/num:bound+1, 3) = b1 + (n-1)*B;
end

axes(handles.axes2); imshow(uint8(x));
axes(handles.axes3); imshow(uint8(bar));
axes(handles.axes4); imshow(uint8(bar2));
```

結果呈現與討論 ：





結果討論:

由此GUI可以自己設定colorbar的初始和最終顏色，並藉由內插法將顏色漸層化出，對比到灰階的colorbar。此外，也可以看到上圖下使用黃藍互補色產生的pseudo-color可以將大腦的部分顯現較好， 上圖上使用藍色和紫色因為色調接近，所以較難觀察大腦部分。另一方面，當bar的大小改變時，會增加pseudo-color的數目，使得色彩漸層較均勻。

Part 3: (30%) Color Segmentation

程式碼 ：

```
function varargout = HW03(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @HW03_OpeningFcn, ...
                   'gui_OutputFcn',  @HW03_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

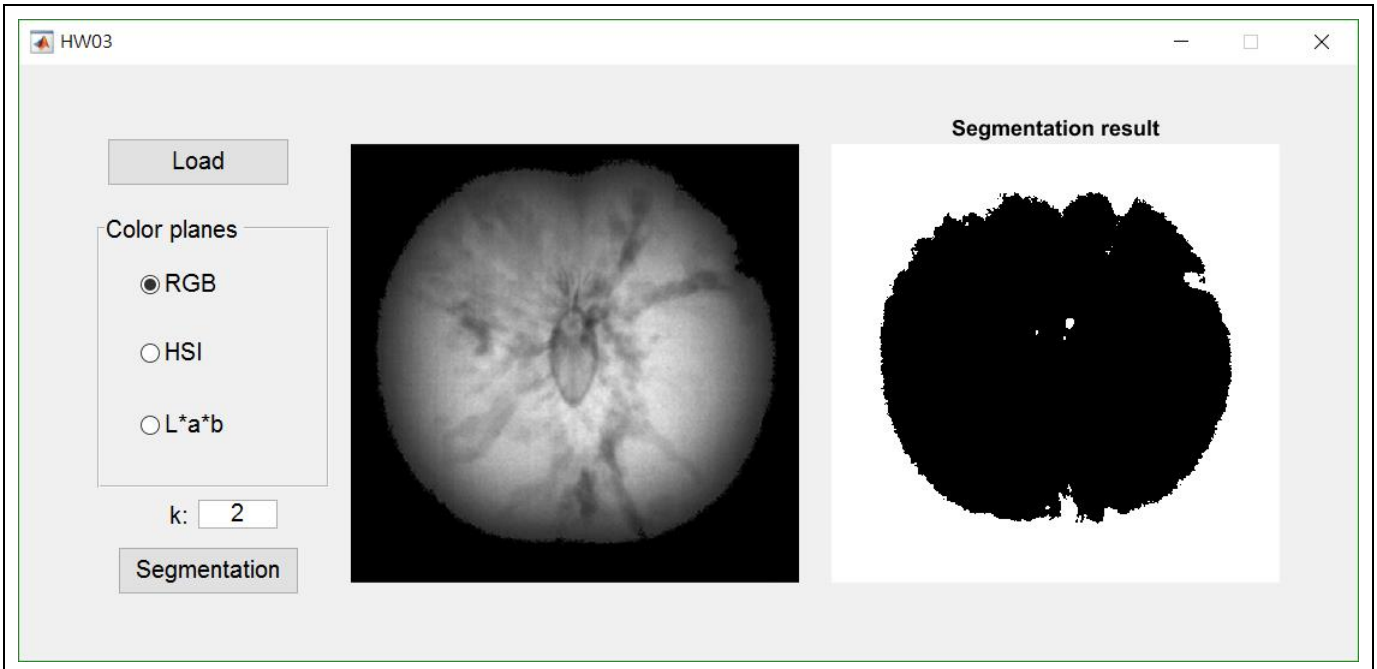
```
function HW03_OpeningFcn(hObject, eventdata, handles, varargin)

set(handles.edit1, 'String', 2);

handles.output = hObject;
guidata(hObject, handles);


function varargout = HW03_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


% Åª¨ú¹Ï¤ù
function pushbutton1_Callback(hObject, eventdata, handles)

filename = uigetfile({'*.jpg;*.tif;*.png;*.gif;*.bmp','All Image
Files'});
handles.I = imread(filename);
handles.RGB_R = handles.I(:, :, 1);
handles.RGB_G = handles.I(:, :, 2);
handles.RGB_B = handles.I(:, :, 3);

axes(handles.axes1); imshow(handles.I);

handles.output = hObject;
guidata(hObject, handles);

% RGB
function radiobutton1_Callback(hObject, eventdata, handles)

handles.color = cat(3, handles.I(:, :, 1), handles.I(:, :, 2),
handles.I(:, :, 3));

handles.output = hObject;
guidata(hObject, handles);


% HSI
function radiobutton2_Callback(hObject, eventdata, handles)

[row, col, ~] = size(handles.I);
```

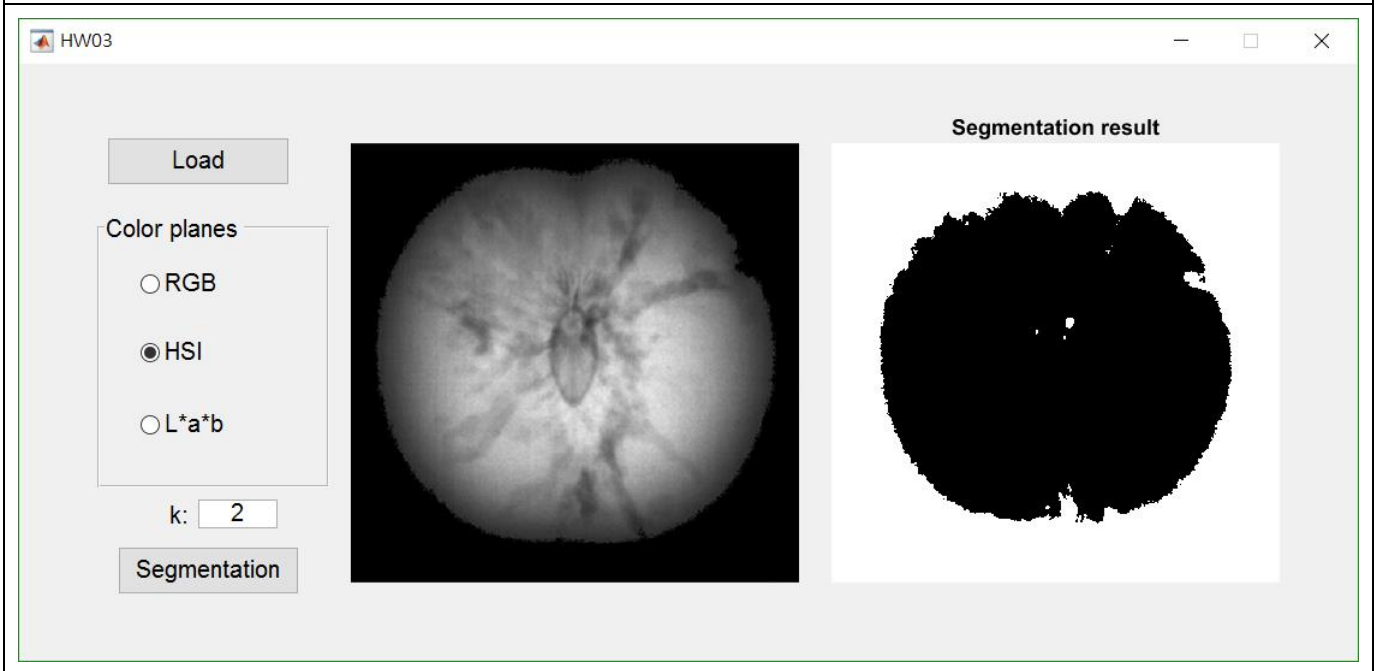
```
for i = 1:row;
    for j = 1:col;
        r = double(handles.RGB_R(i, j))/255;
        g = double(handles.RGB_G(i, j))/255;
        b = double(handles.RGB_B(i, j))/255;
        theta = acosd( (1/2)*( (r-g) + (r-b) )/( (r-g)^2 + (r-b)*(g-
b) )^(1/2) );

        if b <= g
            H(i, j) = theta;
        else
            H(i, j) = 360 - theta;
        end

        S(i, j) = 1 - 3/(r + g + b)*(min([r, g, b]));
        I(i, j) = 1/3*(r + g + b);
    end
end
H = uint8(H);
S = uint8(S*255);
I = uint8(I*255);

handles.color = cat(3, H, S, I);

handles.output = hObject;
guidata(hObject, handles);



% L*a*b
function radiobutton3_Callback(hObject, eventdata, handles)

X = double(0.412453 * handles.RGB_R + 0.357580 * handles.RGB_G +
0.180423 * handles.RGB_B);
Y = double(0.212671 * handles.RGB_R + 0.715160 * handles.RGB_G +
0.072169 * handles.RGB_B);
Z = double(0.019334 * handles.RGB_R + 0.119193 * handles.RGB_G +
0.950227 * handles.RGB_B);

[row, col, ~] = size(handles.I);

xn = 0.9515;
```

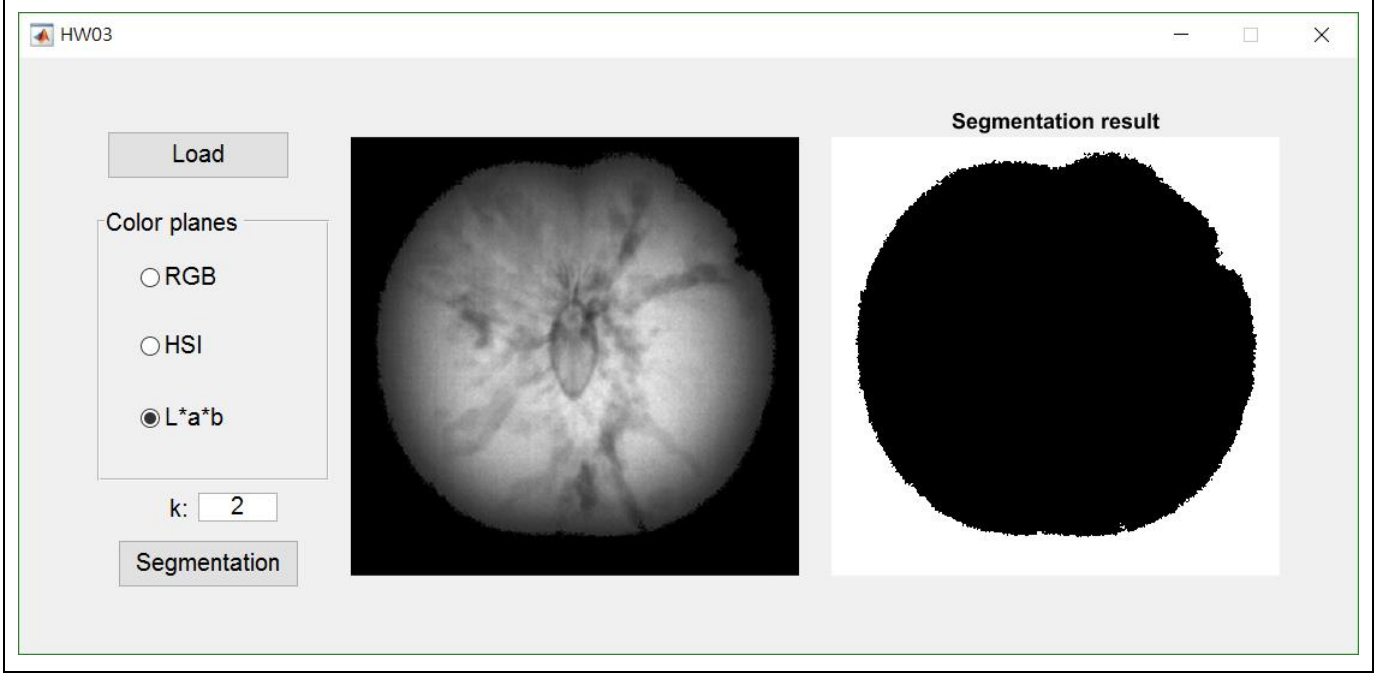
```
yn = 1;
zn = 1.0886;

for i = 1:row
    for j = 1:col
        if Y(i,j)/yn > 0.008856;
            L(i,j) = 116*((Y(i,j)/yn)^(1/3)) - 16;
        else
            L(i,j) = 903.3*Y(i,j)/yn;
        end

        if X(i,j)/xn > 0.008856 ;
            fx = (X(i,j)/xn)^(1/3);
        else
            fx = 7.787*(X(i,j)/xn) + 16/116;
        end

        if Y(i,j)/yn > 0.008856 ;
            fy=(Y(i,j)/yn)^(1/3);
        else
            fy = 7.787*(Y(i,j)/yn) + 16/116;
        end

        if Z(i,j)/zn > 0.008856 ;
            fz = (Z(i,j)/zn)^(1/3);
        else
            fz = 7.787*(Z(i,j)/zn) + 16/116;
        end

        a(i,j) = 500*(fx-fy);
        b(i,j) = 200*(fy-fz);
    end
end

handles.color = cat(3, L, a, b);

handles.output = hObject;
guidata(hObject, handles);


% Kmeans
function pushbutton2_Callback(hObject, eventdata, handles)
```

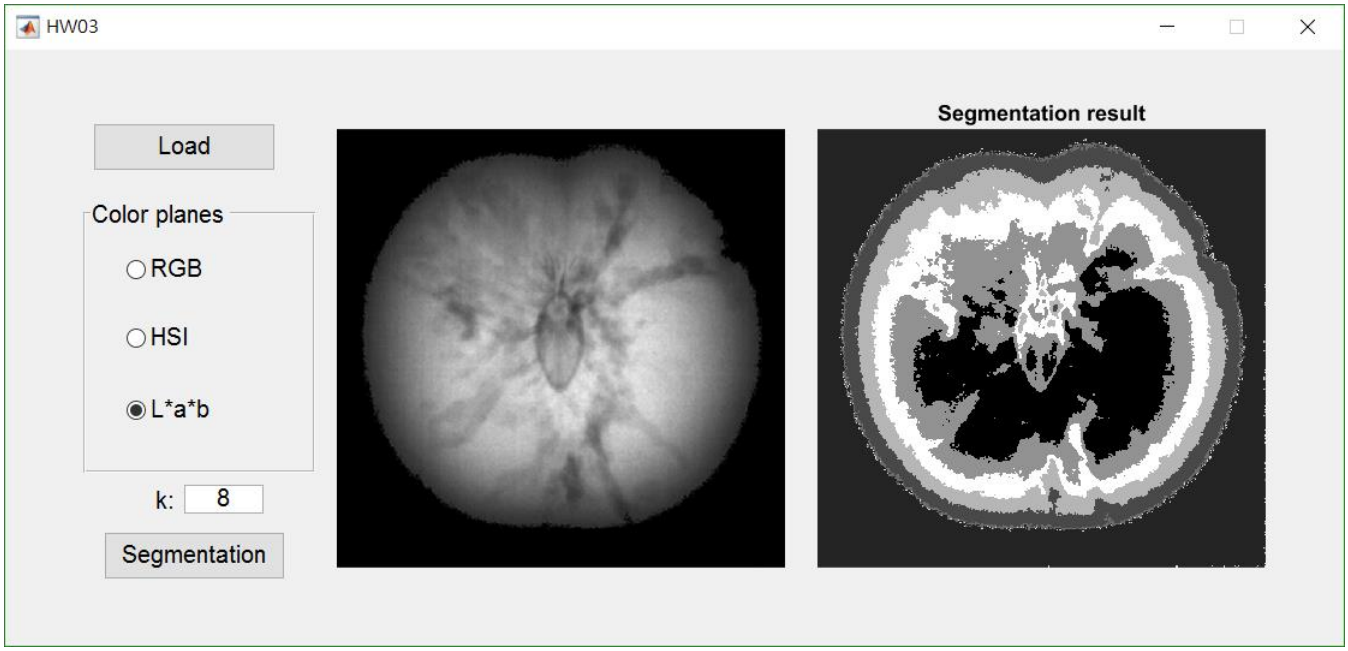
```
fprintf('Processing...');

X = double(handles.color(:,:,1:3));
[row, col, ~] = size(X);
X = reshape(X, row*col, 3);

n = str2num( get(handles.edit1, 'String') );
% -«½Æ¤T¦¸¤ÀÃþ¡AÁ×§K§½³¡³Ì¤p-È
[cluster_idx, ~] = kmeans(X, n, 'distance', 'sqEuclidean',
'Replicates',3);

pixel_labels = reshape(cluster_idx, row, col);
axes(handles.axes2);
imshow(pixel_labels, []), title('Segmentation result');

fprintf('Done\n');

handles.output = hObject;
guidata(hObject, handles);


function edit1_Callback(hObject, eventdata, handles)


function edit1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

結果呈現與討論 ：


HW03
Load
Color planes
RGB
HSI
L*a*b
k:
2
Segmentation
Segmentation result


HW03
Load
Color planes
RGB
HSI
L*a*b
k:
2
Segmentation
Segmentation result


HW03
Load
Color planes
RGB
HSI
L*a*b
k:
2
Segmentation
Segmentation result



結果討論:

由上圖可以觀察出，L*a*b使用kmeans產生的分群效果較RGB和HIS好，可能是因為L*a*b這種模式是以數字化方式來描述人的視覺感應，因此對於人眼上看得區別會有較佳的分類效果。
當k=2到k=8時，由分類兩群變成分類三群，可以看到影像中間的紋路也有部分被區分出來。