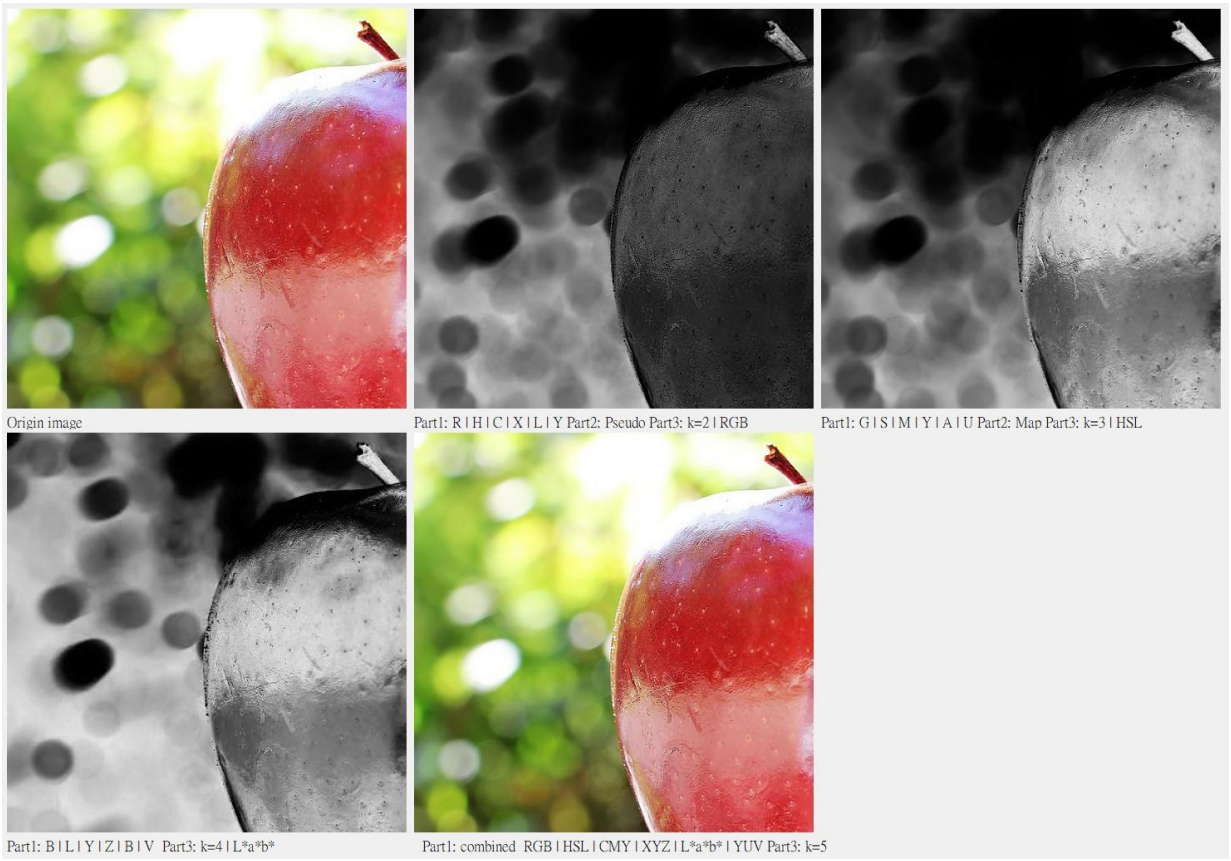
# Principles and Applications of Digital Image Processing

## Homework 5 R12631055 林東甫

Part1

1.要求將影像轉換成各種 color model ,首先是預設的 RGB, 除了將預設的 BGR 轉成 RGB 以外不須做任何轉換:




左上:原影像 中上:R Channel 右上:G Channel 左下:B Channel 中下:RGB 影像

2.HSL

根據 weblink 中所給定的演算法;須注意 HSV 與 HSL 略有不同,因此需要略微修正:

$$H_L = H_V$$

$$L = V\left(1 - \frac{S_V}{2}\right)$$

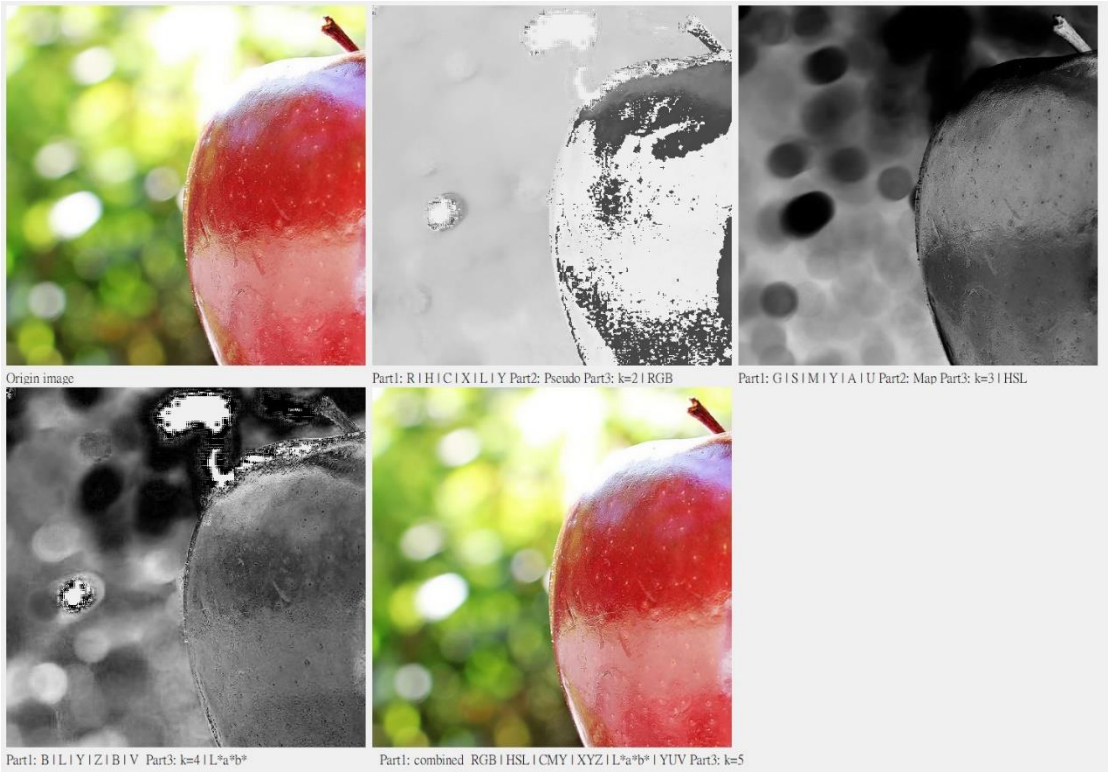
$$S_L = \begin{cases} 0 & \text{if } L = 0 \text{ or } L = 1 \\ \frac{V-L}{\min(L,1-L)} & \text{otherwise} \end{cases}$$

```
void RGBtoHSV( float r, float g, float b, float *h, float *s, float *v )
{
    float min0, max0, delta;
    min0 = min({ r, g, b });
    max0 = max({ r, g, b });
    *v = max0;              // v
    delta = max0 - min0;

    if( max0 != 0 )
        *s = delta / max0;      // s
    else {
        // r = g = b = 0        // s = 0, v is undefined
        *s = 0;
        *h = -1;
        return;
    }

    if( r == max0 )
        *h = ( g - b ) / delta;     // between yellow & magenta
    else if( g == max0 )
        *h = 2 + ( b - r ) / delta; // between cyan & yellow
    else
        *h = 4 + ( r - g ) / delta; // between magenta & cyan

    *h *= 60;               // degrees
    if( *h < 0 )
        *h += 360;
}
```




左上:原影像 中上:H Channel 右上:S Channel 左下:L Channel 中下:合成影像
以下影像皆同.

3.CMY

By 演算法:

```
int rgb2cmyk( cv::Mat &image,cv::Mat &cmyk)
{
    if(!image.data){
        cout<<"Miss Data"<<endl;
        return -1;
    }
    int nl = image.rows;
    int nc = image.cols;
    if(image.isContinuous()){
        nc = nc*nl;
        nl = 1;
    }

    for(int i=0;i<nl;i++){
        uchar *data = image.ptr<uchar>(i);
        uchar *dataCMYK = cmyk.ptr<uchar>(i);
        for(int j = 0;j < nc;j++){
            uchar c = 255 - data[3*j+2];
            uchar m = 255 - data[3*j+1];
            uchar y = 255 - data[3*j];
            uchar k = min(min(c,m),y);

            dataCMYK[4*j] = c ;
            dataCMYK[4*j+1] = m ;
            dataCMYK[4*j+2] = y  ;
            dataCMYK[4*j+3] = k;
        }
    }
    return 0;
}
```

Origin image

Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB

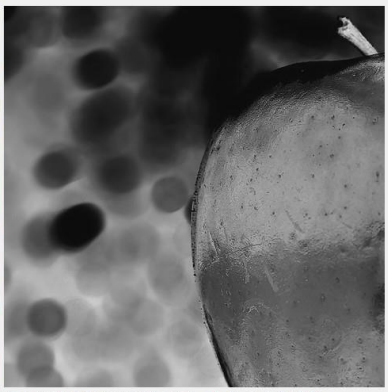
Part1: G | S | M | Y | A | U Part2: Map Part3: k=3 | HSL

Part1: B | L | Y | Z | B | V Part3: k=4 | L*a*b*

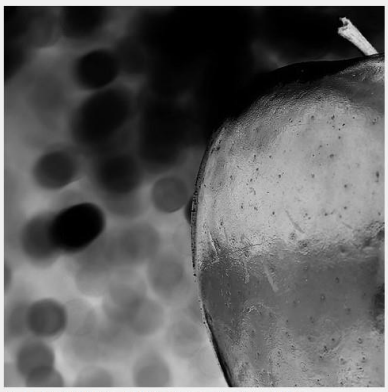
Part1: combined RGB | HSL | CMY | XYZ | L*a*b* | YUV Part3: k=5

## 4.XYZ

Origin image



Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB



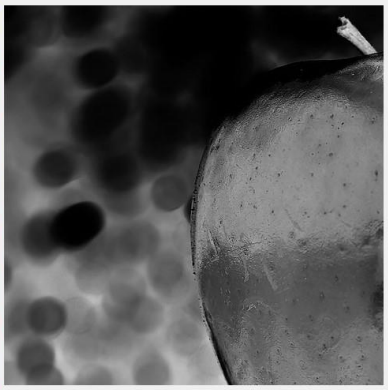
Part1: G | S | M | Y | A | U Part2: Map Part3: k=3 | HSL

Part1: B | L | Y | Z | B | V Part3: k=4 | L*a*b*

Part1: combined RGB | HSL | CMY | XYZ | L*a*b* | YUV Part3: k=5

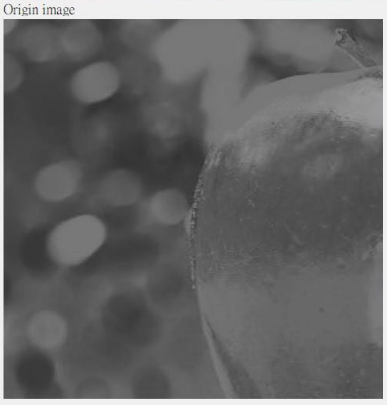
## 5.L*a*b*

Origin image



Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB

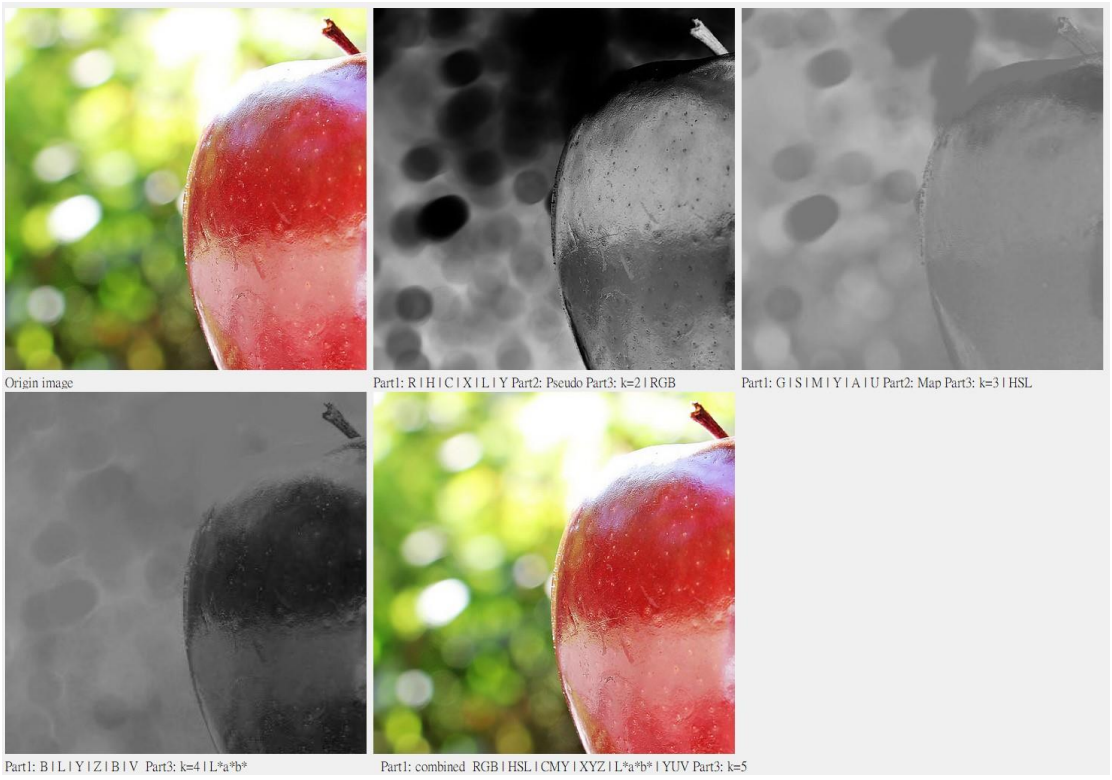
Part1: G | S | M | Y | A | U Part2: Map Part3: k=3 | HSL



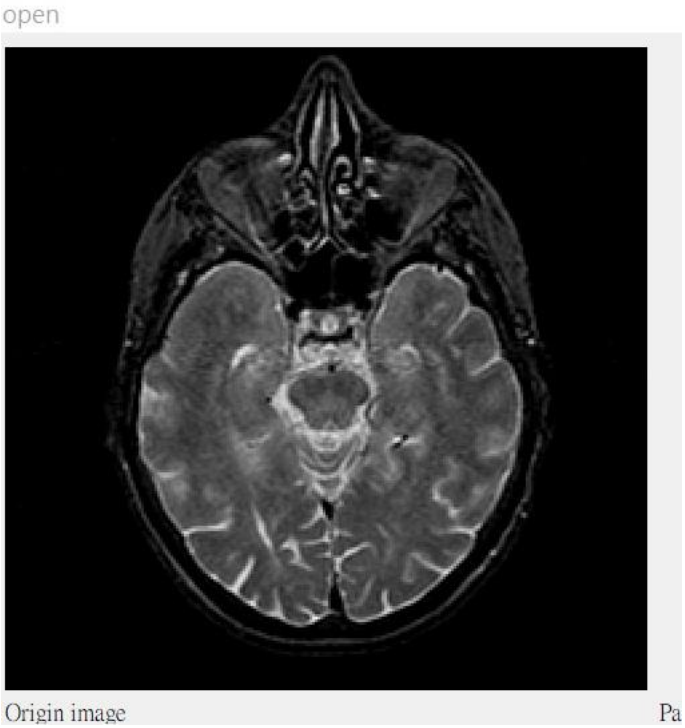
Part1: B | L | Y | Z | B | V Part3: k=4 | L*a*b*

Part1: combined RGB | HSL | CMY | XYZ | L*a*b* | YUV Part3: k=5

6.YUV



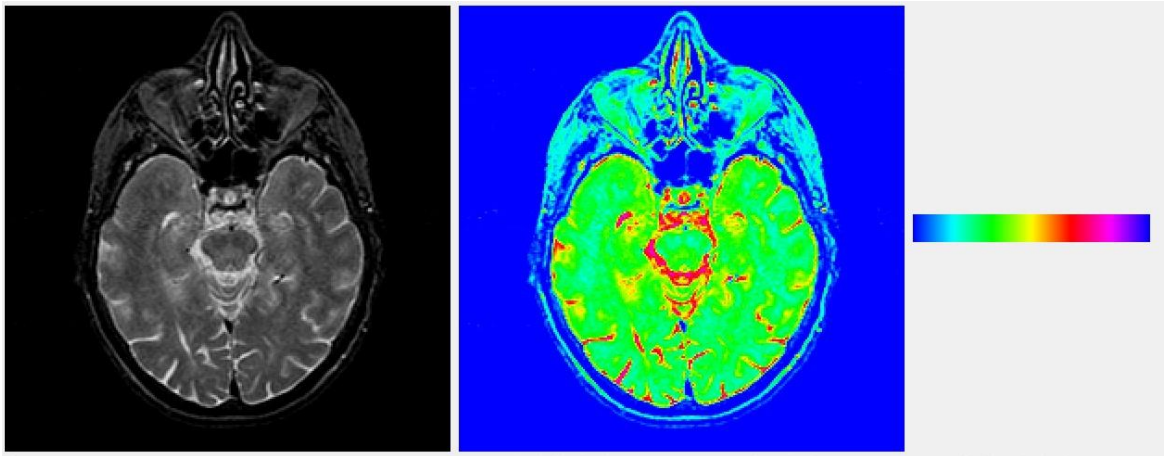

Part2




先從左上角重新載入灰階圖片:

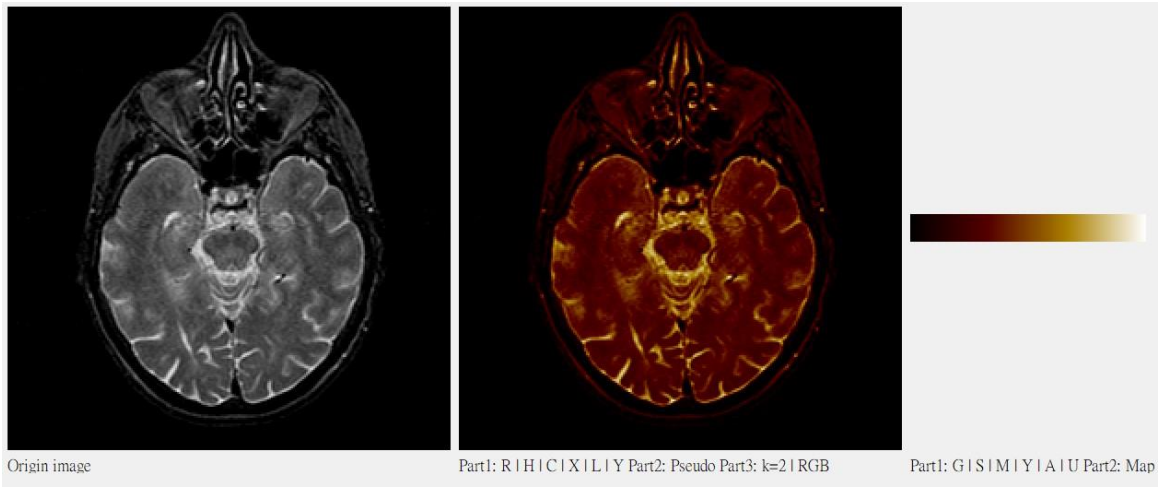
再至左下角選擇想使用的 color table 


選擇完成後即會出現 pseudo-color image 和對應的 color bar(0~255)

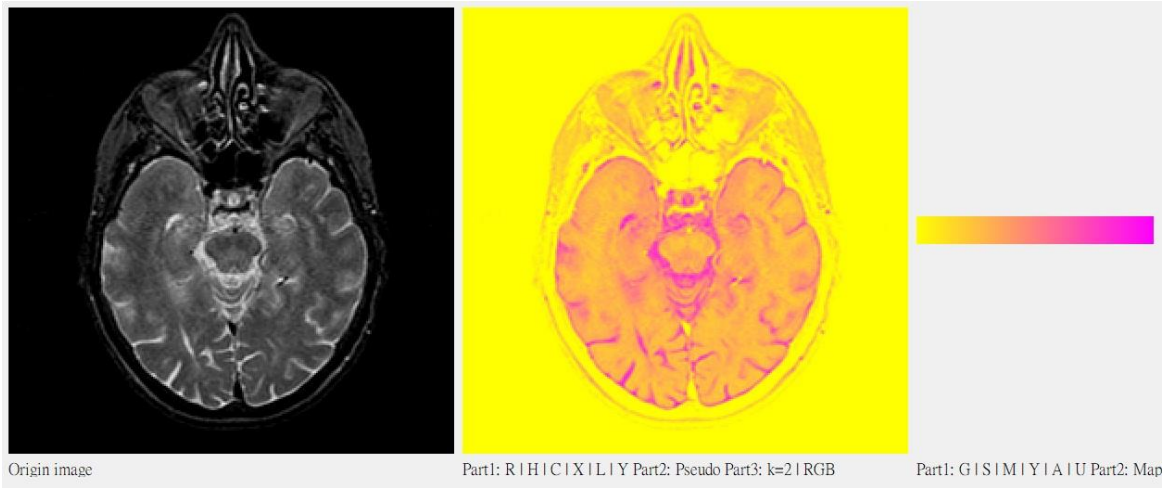


Origin image　　Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB　　Part1: G | S | M | Y | A | U Part2: Map

另提供多種色彩可以自由替換:



Origin image　　Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB　　Part1: G | S | M | Y | A | U Part2: Map



Origin image　　Part1: R | H | C | X | L | Y Part2: Pseudo Part3: k=2 | RGB　　Part1: G | S | M | Y | A | U Part2: Map

Part3

Origin image

一樣先從左上重新載入色彩影像

再按下 Part3: Comparison by k value 即可得到本題目標(image segmentation by color clustering using the k-means algorithm with different k value)

1. k-means algorithm. 2. Using OpenCV kmeans function for clustering.

```
cv::Scalar colorTab[] = {
    Scalar(0,0,255),
    Scalar(0,255,0),
    Scalar(255,0,0),
    Scalar(0,255,255),
    Scalar(255,0,255)
};

int width = img.cols;
int height = img.rows;
int dims = img.channels();
int dims2 = img2.channels();
int dims3 = img3.channels();


int sampleCount= width*height;
int clusterCount = 2;
Mat points(sampleCount,dims,CV_32F,Scalar(10));
Mat labels;
Mat centers(clusterCount,1,points.type());
```

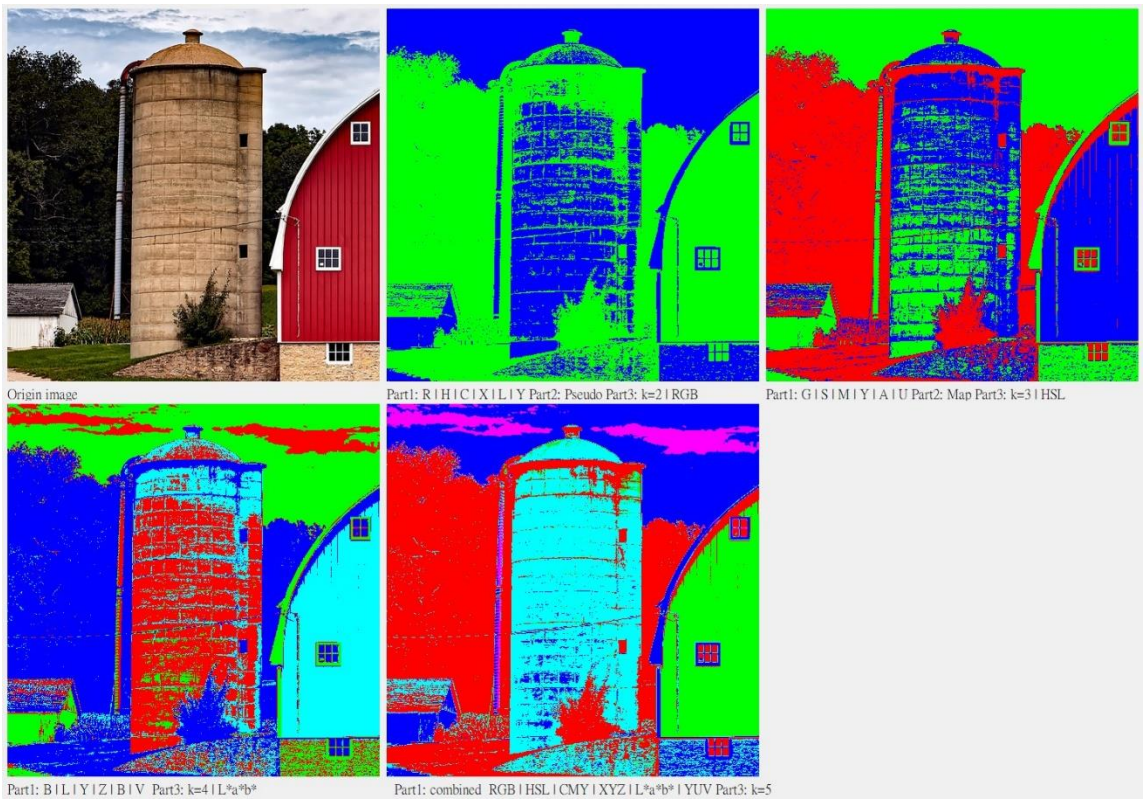
```
for(int row = 0;row<height;row++){
    for(int col = 0;col<width;col++){
        index = row*width+col;
        Vec3b bgr = img3.at<Vec3b>(row,col);
        points3.at<float>(index,0) = static_cast<int>(bgr[0]);
        points3.at<float>(index,1) = static_cast<int>(bgr[1]);
        points3.at<float>(index,2) = static_cast<int>(bgr[2]);
    }
}


TermCriteria criteria = TermCriteria(TermCriteria::EPS+TermCriteria::COUNT,10,0.1);
kmeans(points,clusterCount,labels,criteria,3,KMEANS_PP_CENTERS,centers);
```

```
Mat result = Mat::zeros(img.size(),img.type());
for(int row=0;row<height;row++){
    for(int col=0;col<width;col++){
        index  = row*width+col;
        int label = labels.at<int>(index,0);
        result.at<Vec3b>(row,col)[0] = colorTab[label][0];
        result.at<Vec3b>(row,col)[1] = colorTab[label][1];
        result.at<Vec3b>(row,col)[2] = colorTab[label][2];
    }
}
```

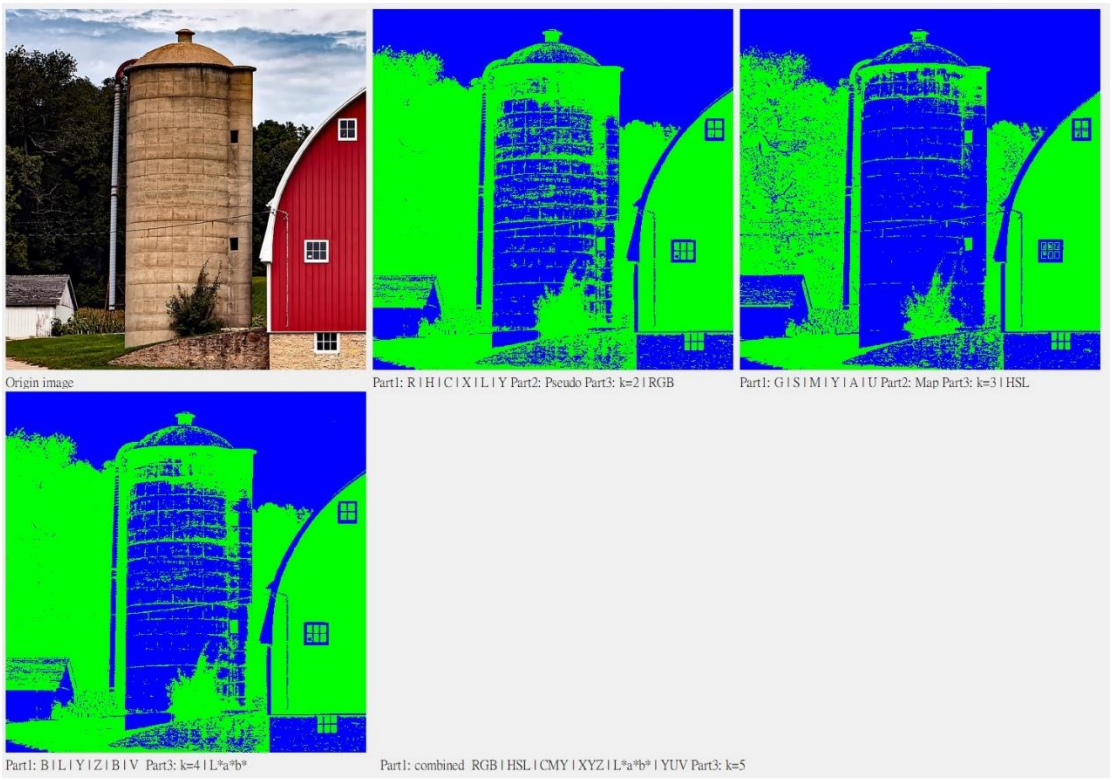
3. Test your program with the accompanied images with various levels of complexity. Compare and discuss color segmentation results using different k values of the k-means algorithm.

運算大概需要花數秒時間,可以感覺到圖片越複雜,分群所需要的時間也越多,另外,從 k=2 分兩色相比,k=4or5 已經分成更多不相似的顏色,也展現出更多的細節.




左上:原影像 中上:k=2 右上:k=3 左下:k=4 中下:k=5

4. Compare the color segmentation results using RGB, HSI, and L*a*b* color planes (using k = 2). ( Part3: Comparison by color model )




左上:原影像 中上: RGB 右上: HSI 左下: L*a*b*