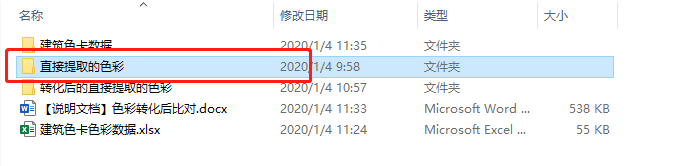
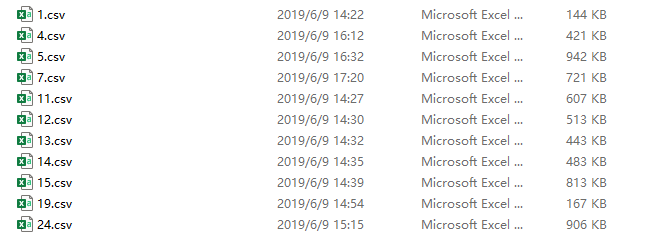
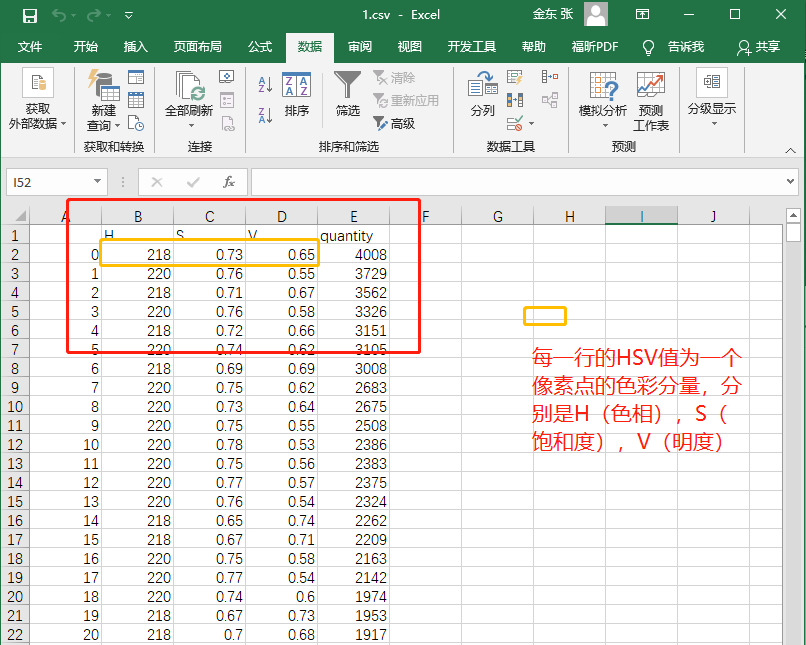
1、将“直接提取的色彩”文件夹中每个CSV文件中的 HSV 值转化为 HVC值。

Ps:先转化为RGB值，再转化为HVC值。



部分截图





HSV 转 RGB色彩 python代码

/// 将HSV色彩模型转换为RGB模型

/// </summary>

/// <param name="H">色调 0-360</param>

/// <param name="S">饱和度 0-1</param>

/// <param name="V">亮度 0-1</param>

/// <returns></returns>

private Color HSVToRGB(int H,double S,double V)

{

if ((H >= 0 && H <= 360) && (S >= 0 && S <= 1) && (V >= 0 && V <= 1))

{

int R=0, G=0, B = 0;

double \_R=0, \_G=0, \_B = 0;

var C = V \* S;

var X = C \* (1 - Math.Abs((H / 60)%2-1));

var m = V - C;

if (H >= 0 && H < 60)

{

\_R = C;

\_G = X;

\_B = 0;

}

else if(H >= 60 && H < 120)

{

\_R = X;

\_G = C;

\_B = 0;

}

else if(H > 120 && H <180)

{

\_R = 0;

\_G = C;

\_B = X;

}

else if(H >= 180 && H < 240)

{

\_R = 0;

\_G = X;

\_B = C;

}

else if(H >= 240 && H < 300)

{

\_R = X;

\_G = 0;

\_B = C;

}

else if (H >= 300 && H < 360)

{

\_R = C;

\_G = 0;

\_B = X;

}

R = Convert.ToInt32((\_R + m) \* 255);

G = Convert.ToInt32((\_G + m) \* 255);

B = Convert.ToInt32((\_B + m) \* 255);

return Color.FromArgb(R, G, B);

}

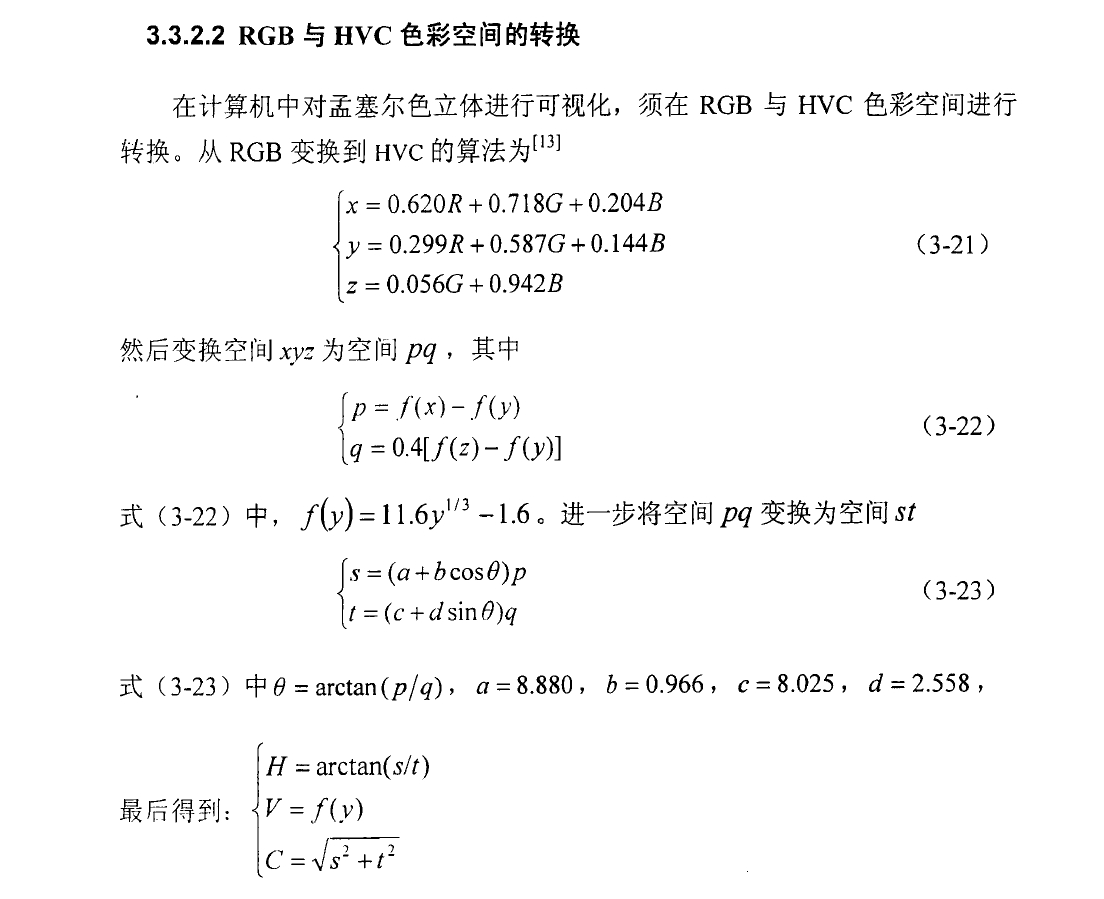
else

{

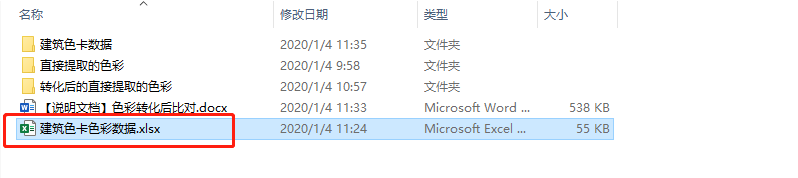
return Color.FromArgb(0, 0, 0);

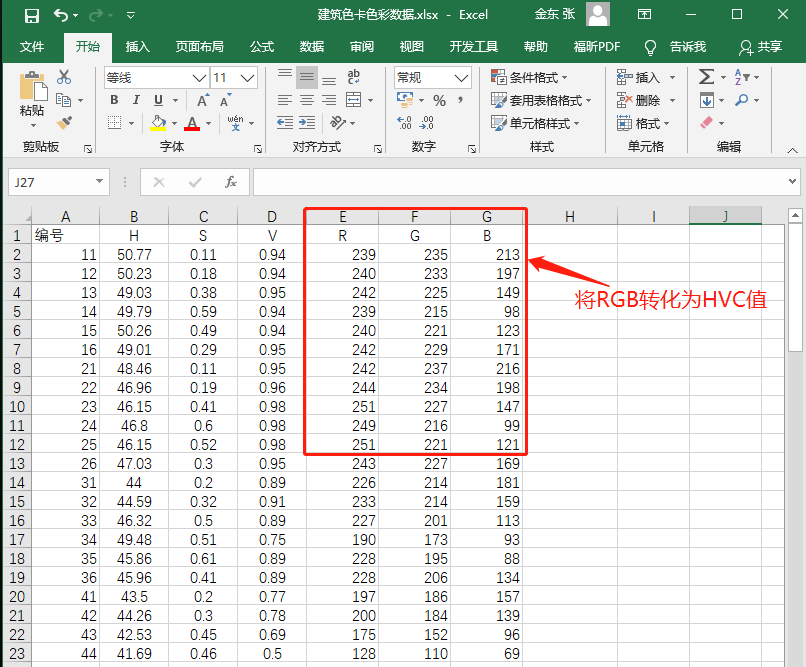
}

}

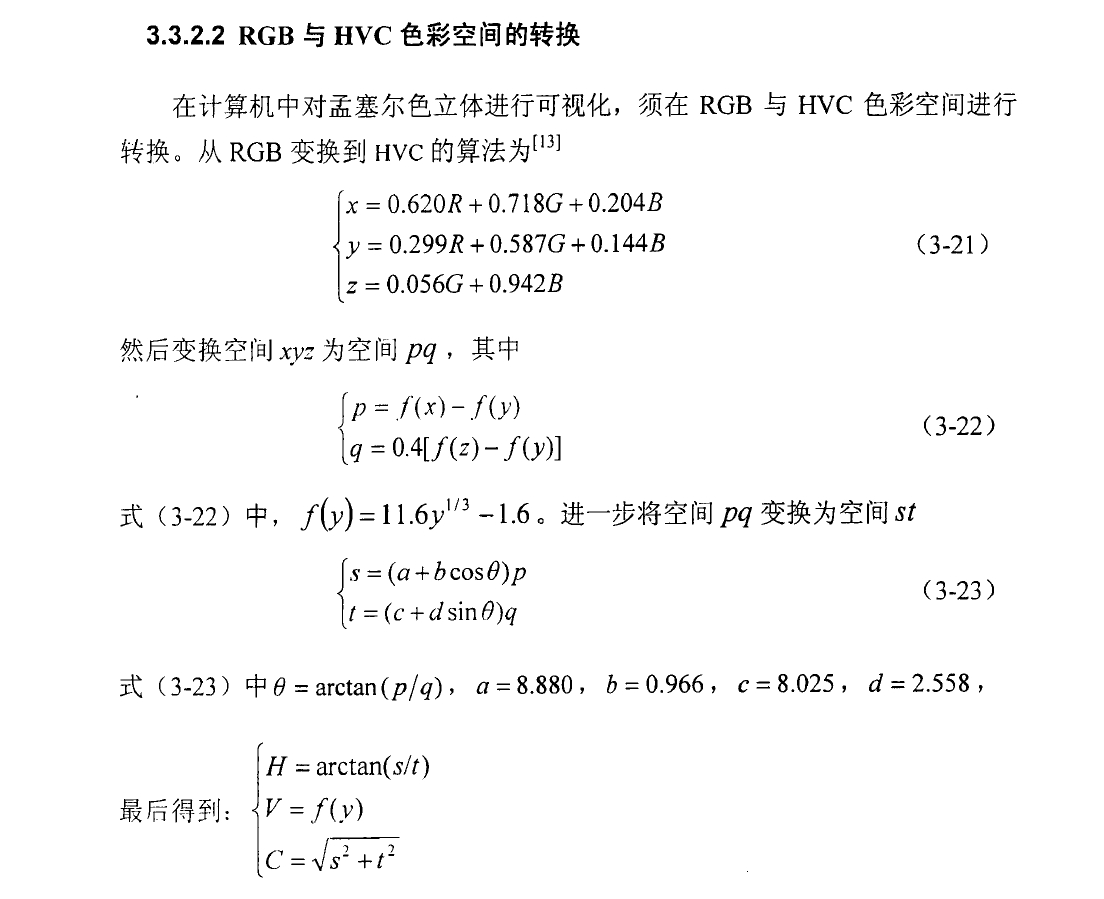


第二步：将“建筑色卡色彩数据.xlsx”中的RGB色彩转化为HVC值





**转化公式**



其中f(x)和f(z)和f(y)公式类似， f(x)=11.6-1.6 ,f(z)=11.6-1.6

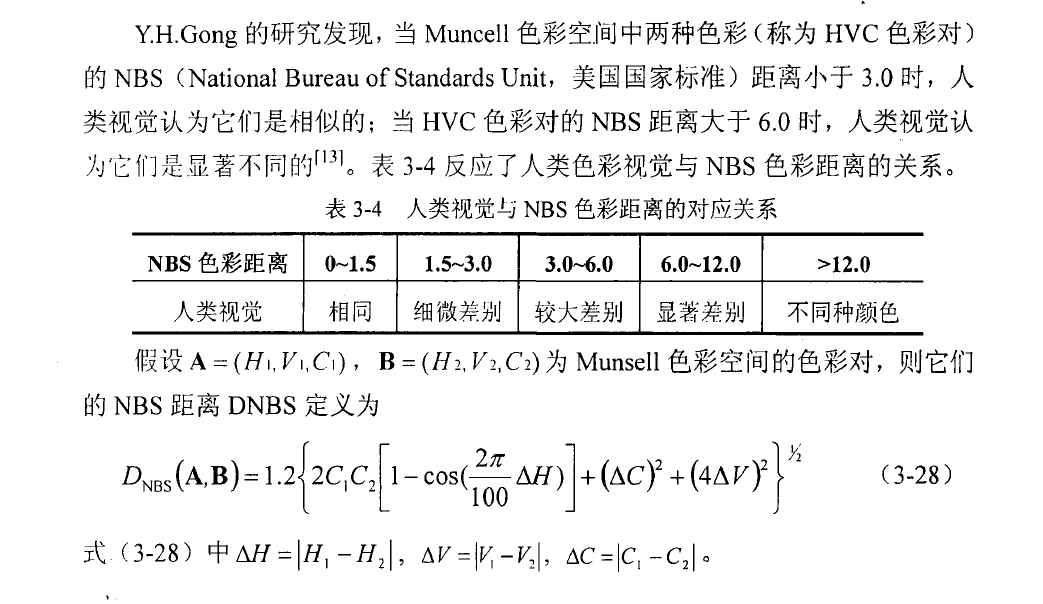
第三步：两两HVC值之间相似性计算

3.1

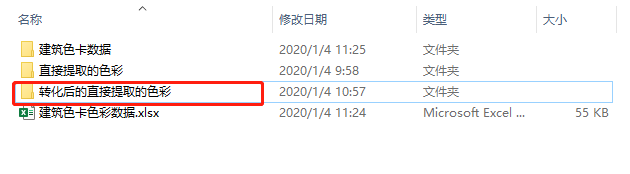
将计算得到的“直接提取的色彩”文件夹中每个CSV文件的每个HVC值与“建筑色卡色彩数据.xlsx”中转化得到的每个HVC值基于NBS比对，若的值小于1.5，则将该“直接提取的色彩”HSV值与HVC值及quantity提取出来，放到一个新的csv文件中，命名为“new\_x”,其中“x”为原始CSV文件的文件名，如“new\_1;new\_2”等；并放到一个命名为“转化后的直接提取的色彩”文件夹中。若“直接提取的色彩”文件夹中每个CSV文件的每个HVC值存在与几个“建筑色卡色彩数据.xlsx”中转化得到的HVC值都小于1.5，则归类到值最小的那个“建筑色卡色彩数据.xlsx”数据中。

3.2

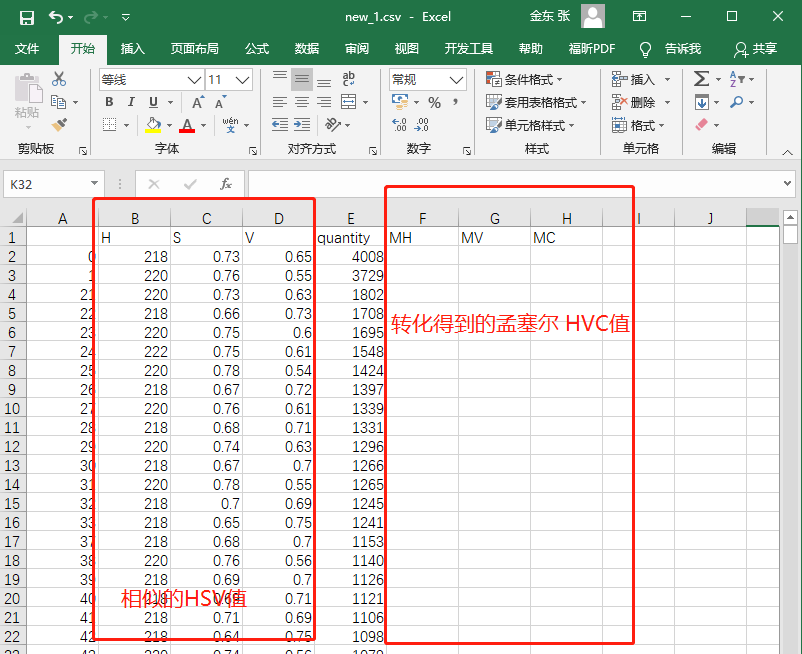
相似的部分的将quantity数量加和，放到一个新的excel表“x-建筑色卡聚类”中，新的excel表放到到“建筑色卡数据”文件夹目录下。Ps：“x-建筑色卡聚类”中的“x”值为原始csv文件的文件名称。例如：“1\_建筑色卡聚类；2\_建筑色卡聚类”等。



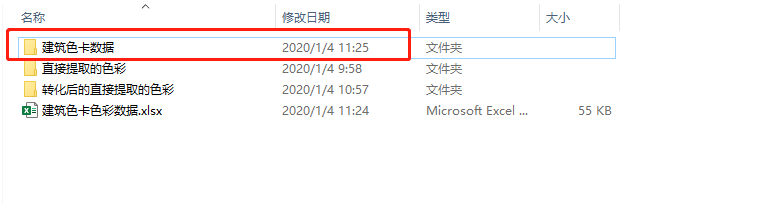
示例3.1：

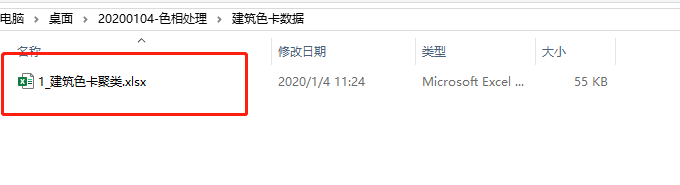


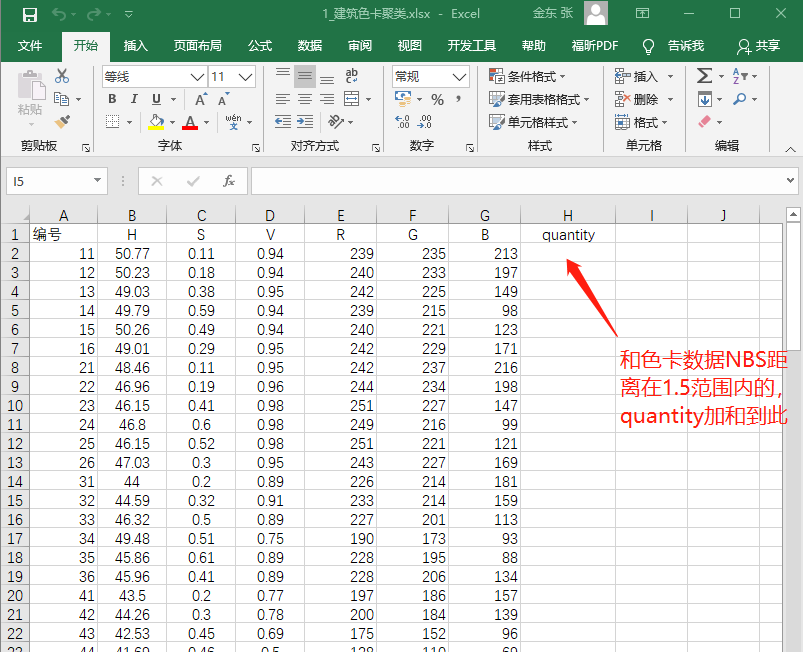




示例3.2







输出结果

