# 颜色

## 色彩平衡

typedef struct {

bool preserve\_luminosity;

double cyan\_red[3];

double magenta\_green[3];

double yellow\_blue[3];

} color\_balance\_config;

该调整算法分别修改颜色的RGB分量，针对不同的强度范围（该强度只是相应RGB分量的强度），给予不同程度的调整。

三个强度范围分别是阴影、中等深浅、高亮，对应数组cyan\_red、magenta\_green、yellow\_blue的0、1、2下标。

调整量在[-0.5, 0.5]之间，负值表示减少该颜色分量，正值表示增加该颜色分量。

算法不影响alpha通道。

设置preserve\_luminosity为true保持图像强度不变。

## 色调-饱和度

typedef struct \_hue\_saturation\_config {

double hue[7];

double saturation[7];

double lightness[7];

double overlap;

} hue\_saturation\_config;

该调整算法针对不同色调的颜色分别调整其色调、饱和度、强度。可以分别配置红、黄、绿、青、蓝、品6个色系的修改量，对应数组下标1-7。六个色系对应的色调范围为（这里假设整个色调范围为6，在原来色调基础上乘以6得到）：[0.0, 0.5]U[5.5, 6.0], [0.5, 1.5], [1.5, 2.5], [2.5, 3.5], [3.5, 4.5], [4.5, 5.5]。主值影响所有色系，对应数组下标0 。

如果指定了覆盖范围（overlap > 0，overlap <= 1），那么每个色系的范围向两端分别延伸overlap/2，当要修改的颜色色调处于相邻色系的重叠区域时，颜色整个调整效果是对这两个色系调整效果的加权平均值。越靠近的某个色系，对该色系调整效果占比重越大。

对色调的修改是将色调加上调整量，需要注意的是色调是循环的，即x与x+1.0和x-1.0是同一个色调。调整量是该色系的色调调整量与主值的色调调整量的平均值。配置值hue的取值范围都是[-1.0, 1.0]，因此可以将颜色调整到整个色调区间内的任何值。

对强度的修改是：如果减少强度，是将原来的强度乘以一个系数，范围[0.0, 1.0]，对应lightness的取值为[-1.0, 0.0]，如果是增加强度，是将原来的强度与最亮值1.0的差值减少，差值被缩小到原来的[0.0, 1.0]倍，对应lightness的取值也是[0.0, 1.0]。

对饱和度的修改是将原来的饱和度乘以一个系数，配置值stauration的取值范围是[-1.0, 1.0]，对应的乘积系数为[-1.0, 3.0]。（因为对单独色系的修改量与主值修改直接相加，没有做平均，所以系数范围是配置取值范围的2倍，系数为负数时，等同于0）。

该调整算法不影响alpha通道。

## 单色化

typedef struct {

double hue;

double saturation;

double lightness;

} colorize\_config;

图像将被修改为唯一的色调值和饱和度，0-360度的色调映射为hue取值的[0.0, 1.0]，0-100的饱和度映射为saturation取值的[0.0, 1.0]。

强度值被增量调整，调整量在[-1.0, 1.0]之间，负值表示减少强度，正值表示增加强度。

该调整算法不影响alpha通道。

## 亮度-对比度

typedef struct {

double brightness;

double contrast;

} brightness\_contrast\_config;

调整亮度，对比度，取值范围都是[-1.0, 1.0]。

## 阈值

typedef struct {

double low;

double high;

} threshold\_config;

计算每个像素的最大rgb分量，如果该值处于[low, high]范围内，结果像素被置成白色，否则置成黑色。low、high的取值范围都是[0.0, 1.0]，且low <= high。

不影响alpha通道。

## 色阶

typedef struct {

double gamma[5];

double low\_input[5];

double high\_input[5];

double low\_output[5];

double high\_output[5];

} levels\_config;

算法可以调整五个通道：强度、红色、绿色、蓝色、Alpha的颜色分量，分别对应各数组的0-4下标，调整强度通道等效于相同操作同时作用于红色、绿色、蓝色通道。如果图像没有Alpha通道，Alpha通道不可编辑。如果同时编辑了强度通道和红色、绿色、蓝色通道，则先作用于红色、绿色、蓝色通道，生成一个中间值，再用强度通道定义的调整方式进行调整。

对每个通道的调整由相应的色阶控制，控制分为输入色阶、输出色阶两个控制阶段。色阶实际上是一种曲线映射。

输入色阶由low\_input、high\_input、gamma控制，将值low\_input映射为0，值high\_input映射为1.0（整数值是255），其他值被线性拉伸。 low\_input、high\_input取值范围都是[0.0, 1.0]，high\_input应该大于low\_input，否则颜色会翻转。

gamma对颜色值进行指数非线性映射，，如果gamma大于1，颜色向白色靠近，如果gamma值小于1，颜色向黑色靠近。

输出色阶由low\_ouput、high\_output控制。将值0映射为low\_output，值1.0（整数值是255）映射为high\_output，其他值被线性压缩。low\_ouput、high\_ouput取值范围都是[0.0, 1.0]，high\_ouput应该大于low\_ouput，否则颜色会翻转。

## 曲线

typedef struct {

Curve \*curve[5];

} curves\_config;

Curve定义一个曲线，有两种类型的曲线：平滑曲线、手绘曲线。两种曲线类型内部都通过一组x均匀分布的样本点表示，样本点之间的值通过线性插值计算。平滑曲线由一组控制点控制形状，由控制点通过bezier插值计算出样本点。

采用一下步骤生成平滑曲线：

1. 初始化曲线结构

curve\_init(curve);

1. 设置控制点数目和样本点数目

curve\_set\_n\_points(curve, num);

curve\_set\_n\_samples(curve, num);

1. 设置每个控制点

curve\_set\_point(curve, 0, x0, y0);

curve\_set\_point(curve, 1, x1, y1);

……

1. 插值计算样本点

curve\_dirty(curve);

采用以下步骤生成手绘曲线：

1. 初始化曲线结构

curve\_init(curve);

1. 设置曲线类型

curve\_set\_curve\_type(curve, CURVE\_FREE);

1. 设置样本点数目

curve\_set\_n\_samples(curve, num);

1. 设置样本点

curve\_set\_curve(curve, x0, y0);

curve\_set\_curve(curve, x1, y1);

……

实际编辑中将原来的值按曲线定义的映射关系修改为映射目标值。可以修改的通道有强度、红色、绿色、蓝色、Alpha五个通道，其中强度通道通过同时修改红色、绿色、蓝色通道实现。如果图像没有Alpha通道，Alpha通道不可编辑。

如果同时选择了强度和红色、绿色、蓝色的组合，则对红色、绿色、蓝色分量的修改效果是二者重叠作用的效果，先分别用红色、绿色、蓝色曲线映射，生成的中间值再通过强度曲线映射，得到最终值。

## 海报效果

typedef struct {

double levels;

} posterize\_config;

采用聚类方式调整RGB三个颜色分量。具体操作是将[0.0, 1.0]的颜色范围划分成若干组，每组的长度是1/levels，所以位于某个组的颜色分量调整为该组范围的中间值。