**像素转换算法**

**一、算法目的：**

终端提供了填充背景像素的相关操作，但需要用户一个个像素进行填充，对于一张图像，需要它在终端上显示，我们需要对每个像素的颜色进行手动输入说明，这将会产生庞大的工作量，此算法的主要目的就是帮助用户以固定格式输出每个像素的颜色编号，减少工作量

终端虽然提供了颜色填充操作，使编译结果有了彩色图像的可能，但是它所支持的颜色数量有限，一共包括16种，具体支持的颜色种类如表1-1，于是需要一种转换方法，将包括了多种颜色的图像转换成由这16种颜色构成的图像。

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 颜色 | RGA值 | 颜色示例 |
| 0 | 黑色 | （0,0,0） |  |
| 1 | 深蓝色 | （0,0,170） |  |
| 2 | 绿色 | （0,170,0） |  |
| 3 | 蓝绿色 | （0,170,170） |  |
| 4 | 红色 | （170,0,0） |  |
| 5 | 品红色 | （170,0,170） |  |
| 6 | 浅棕色 | （170,85,0） |  |
| 7 | 浅灰色 | （170,170,170） |  |
| 8 | 深灰色 | （85,85,85） |  |
| 9 | 蓝紫色 | （85,85,255） |  |
| 10 | 浅绿色 | （85,255,85） |  |
| 11 | 浅蓝色 | （85,255,255） |  |
| 12 | 浅红色 | （255,85,85） |  |
| 13 | 浅洋红 | （255,85,255） |  |
| 14 | 黄色 | (255,255,85) |  |
| 15 | 白色 | (255,255,255) |  |
| 16 | 透明 | A = 0 | 无 |

表1-1. 终端支持的颜色种类

**二、算法思路**：

对于待处理的图像，首先要获取到每个像素的RGBA的值，将对图像的分析转换到对单个像素的RGBA的分析上，对于透明度A我们暂不考虑，因为A的取值只有0或1（0代表不透明，1代表透明），对颜色不会产生影响；我们只需考虑RGB三个颜色值的组合模式对颜色变化的影响，接下来我们主要讨论RGB的几种组合模式，假设我们读取到的某个像素的RGB=（r, g, b）。

如果r，g，b中的某个值要明显大于其他两个，我们认为这个值对颜色起主要作用，例如r = 200，g=b=20，200明显大于20，则这个像素的颜色一定属于红色系（如图2.1），剩余两个值只会对颜色的深浅起到一定作用；

   

图2.1 RGB=(200,20,20) 图2.2 RGB=(200,200,20) 图2.3 RGB=(85,85,85) 图2.4 RGB=(170,170,170)

如果r，g，b中的两个值要明显大于余下的一个值，我们认为这两个值对颜色起主要作用，例如r=g=200，b=20，则这个像素一定介于红色与绿色之间，属于黄色系,如图2.2；

如果r，g，b三个值相等，则这个像素的颜色一定属于黑白色系，例如r=g=b=0，则这个像素的颜色是黑色；r=g=b=85，则颜色是浅灰色；r=g=b=170，则颜色是深灰色；r=g=b=255，则颜色为黑色；示例如图2.3、2.4

另外，算法允许用户输入一个值value，r、g、b之间的差值若小于value则认为相等，如r-b<value，则r=b，以此可以调节转换后图像的颜色数目，value越大数目越少，最大可到255（颜色范围0-255），此时转换出来的图像的颜色只属于黑白色系，即灰度图，示例如图2.5。

  

图2.5 从左到右依次为原图，value=50，value=255

最后和可选颜色进行模式对比，如果匹配到目标（目标可能匹配到多个），则计算待处理颜色值与各个目标的距离，设一个目标颜色RAB=（r’，g’，b’），则距离distance= （r-r’）²+（g-g’）²+（b-b’）² 选择距离最小的目标颜色即最相近的颜色，输出像素编号；如果没有匹配到则直接计算待处理颜色与各个可选颜色的距离，选取最小值输出编号；

**算法流程图**：

保存颜色信息

计算最相近颜色

开始

结束

输出颜色编号输出图像

可选的几个颜色中是否存在

对图像的每个像素点进行处理分析与相关计算

接受图像信息