

基于 RGB 颜色空间的马赛克瓷砖选色问题

摘 要

随着人们生活水平的提高,对审美的更高追求,采用马赛克瓷砖进行装饰时,设计人员要赋予它们更多的功能和情感^[8]。目前,马赛克瓷砖,因其符合人们的几何审美逻辑和优良的自身特性,作为一种基本的建材装饰材料,广泛用于各种装饰性图案的拼接。但由于工艺和成本的限制,马赛克瓷砖在还原图像方面仍具有局限性。

本文通过对 RGB 颜色空间的矢量-角度距离色差公式的应用,由改良后的两种颜色之间的色差度量公式,引入描述两幅图像相差程度的平均色差和色彩还原度两个量纲的概念,实现了通过量化图像颜色和已有瓷砖颜色的相差度匹配最相近颜色,并基于实验数据和假想模型,引入人眼阈值梯度简化模型,合理分析求解了瓷砖增色的最优解,最后通过定量分析比较还原度收益对瓷砖增色进行了评价和建议。

首先,通过相关论文查阅,我们基于同时考虑 RGB 颜色间空间距离和矢量角度值的矢量-角度距离色差公式,通过出于普适性提高的优化和改良后,建立模型一。模型一通过优化后的色差度量公式,将图像颜色和现有瓷砖颜色之间的色彩差距离量化为色差,通过色差之间的对比,对不同的瓷砖选择进行单一变量的评价,并匹配与图像颜色色差最小的瓷砖颜色,实现了对马赛克瓷砖选砖问题的解决(见附录 9.1.2)。模型一的普适性较强,评价较为客观,但美中不足的是对于颜色这一受环境影响较大的对象,模型只以两种不同颜色的 RGB 编码值作为变量,环境中其它变因(如光线等)会对其产生较大干扰。

然后,我们将两种颜色之间的色差概念推广到两幅图像之间的色差度量,构建了平均色差这一概念,建立模型二,用于求出新增瓷砖颜色的最优解。出于普适性的考虑,我们假想了一幅含有所有颜色 $255 \times 255 \times 255$ (1658 万)种的图像,马赛克瓷砖颜色对这一假想样本颜色还原后的平均色差将较为客观地反应已有瓷砖颜色的还原能力,通过定量分析不同新增颜色对瓷砖还原能力的提高程度,就能筛选出增加不同数目的 RGB 颜色的最优解。虽然这样模型二的优化结果较为客观,但处理的数据量过于庞大,需要牺牲大量时间,效率极低。于是我们设计并进行了一个关于人眼对于 RGB 颜色敏感度阈值的实验,经过对实验结果的分析和相关文献的查证,我们设计了一个针对人眼敏感度的 RGB 三原色编码值的阈值梯度,我们认为 R、B 基色编码值的阈值梯度约为 43,而人眼相对敏感的 G 基色编码值的阈值梯度设置为 32,经过简化后,我们挑选出 $7 \times 9 \times 7$ (441)种颜色作为样本,能够较为客观地代表原始模型样本,将这一样本作为新增颜色来源,并将颜色具有相对合理梯度的附件中的图像一作为待还原的图像样本,通过求解平均色差,求得新增不同颜色数目时的最优解。

基于上述两种模型,我们将平均色差这一量纲依次进行正向化和归一化处理,构建了还原度这一概念,范围定义在 $[0,1]$ 。还原度能较为直观地表现出瓷砖拼接图像对于原始图像的还原程度。在对颜色增加数目和还原度、颜色增加数目和还原度增量分别进行拟合和回归分析后,我们得到了增加颜色数目和还原度的收益之间的粗略关系。基于以上分析,我们建议增加 4 种瓷砖颜色,RGB 编码值分别为 $(86,32,215)$, $(86,224,129)$, $(129,64,43)$ 和 $(129,64,215)$ 。

本文的特色为基于翔实丰富的资料,通过改良矢量-角度距离色差公式,并对建立模型过程中的问题大胆设想,提出假设,小心求证,在借鉴大量论据经验的条件下,提出了一种开创性的马赛克瓷砖拼接图像颜色匹配与选砖的方案,具有适用性广、稳定性好、灵敏度高等特点,在深入分析后给出了具有可观可行性的建议。

关键词：RGB 颜色空间 色差 还原度 非均匀量化 马赛克瓷砖 人眼阈值梯度

一、 问题重述

1.1 问题背景

马赛克瓷砖，因其尺寸小，便于铺设的特性，常用于各种装饰性图案的拼接。但由于工艺和成本的限制，瓷砖的颜色的种类是有限的。用户根据理想效果拼接图案时，需根据原图的颜色，在已有的颜色中找到最相似的进行匹配。

1.2 问题提出

某马赛克瓷砖生产厂仅能生产 22 种颜色的马赛克瓷砖。该厂希望开发一个软件，能根据原始颜色，自动找出与之颜色最相似的瓷砖，并得到相应的算法，达到减少人工的工作量的目的。此外，工厂提供了三个附件，附件 1(该厂可以生产的马赛克瓷砖的颜色)、附件 2(图像 1 中的 216 种颜色)和附件 3(图像 2 中的 200 种颜色)。

基于上述背景和附件信息，我们需要建立数学模型解决以下问题：

(1)从该厂可以生产的瓷砖颜色中，找出与图像 1 和图像 2 中每个颜色分别相似的瓷砖颜色，并输出结果；

(2)若研发新颜色的瓷砖，在仅考虑拼接图像的表现力的情况下，如何选择需优先增加的瓷砖颜色。并根据增加的颜色数量，给出对应颜色的 RGB 编码值；

(3)在研发一种新颜色的成本相同的情况下，综合考虑成本及表现效果，应新增哪几种颜色；

二、 问题分析

2.1 问题一的分析

问题一的目标是根据已知图像的颜色 RGB 编码列表，匹配最相近的瓷砖颜色。基于这个目的，我们定义色差作为指标，反应两种颜色的相差程度。

首先，该问题基于 RGB 颜色空间，RGB 颜色空间是基于 RGB 颜色模型的任何附加颜色空间，常以 RGB 立方体呈现(见图 1)。^[1] 定义在 RGB 颜色空间中任意两点间的距离：

$$D(x_i, x_j) = \sqrt{(r_i - r_j)^2 + (g_i - g_j)^2 + (b_i - b_j)^2} \quad (1)$$

为色差，反映它们之间的相似度。公式(1)是建立在 RGB 颜色空间是均匀的情况下，但由实验得到，人眼对 RGB 三个分量的敏感度存在差异，人眼对 RGB 三原色的敏感度大小依次为：绿色>红色>蓝色，导致 RGB 色彩空间是不均匀的^[2]，故考虑对三个分量分别赋予权重。现有的文献中，根据所研究图像的不同，研究人员在实验中对 RGB 分量赋予了不同的权重，但是经实验证明简单加权并不具有普适性。于是，D.Androutsos 等人提出 RGB 角距离色差公式，加入了角度差对颜色差之间的影响。^{[4][5]}

其次，鉴于现有的公式对于 RGB 空间色差的度量效果均不理想，杨振亚等人提出了一种 RGB 颜色空间色差计算公式，将固定权值转变为可依据情况改变的动态权值，一定程度上补偿 RGB 空间的不均匀性^[3]。考虑到 RGB 三个分量之间的关系并非简单的比例关系，故我们认为杨振亚等提出的模型中，分别赋予 RGB 分量 1, 2, 1 的权值是不够完善的，需对权值进行修改。

并且，我们发现杨振亚等提出的 RGB 颜色空间色差计算公式有局限的部分，当存在某种颜色要与黑色(RGB 值为(0, 0, 0))进行色差比较的时候，公式(7)将会因为 0 向量与任意向量夹角无法比较的问题，从而无法计算出 θ 的数值，所以这个时候我们需要加入对特殊情况的考虑，使公式更具普适性。如前文所述，人眼无法分辨色差过小的两种颜色，因此，我们认为 RGB 编码值为(0,0,0)的颜色可以近似等效为 RGB 编码值为(1,1,1)的颜色(见表 1)，这个等效所产生的误差在可接受范围内。经过优化后，模型具有更高的普适性。

当两种待比较的颜色的 RGB 某一个或两个分量数值都为 0 时，例如：(0, 20, 36)，(0, 20, 54)这两种颜色时，二者的 R 值皆为 0，会导致建立在三维坐标系统的公式(9)、(10)、(11)无意义，从而无法计算。经过不断的分析和探讨，我们认为在比较色差的过程中，如果两种颜色的 RGB 三基色分量都为 0，应该将三维坐标系统降维成二维坐标系统或是一维坐标系统，所以这种情况下我们在模型建立的过程中加入了新的计算公式作为完善与补充。

基于上述分析，我们对杨振亚等^[3]提出的模型提出优化模型。由实验，得到人眼对三原色敏感度的粗粒度的结论：在黑色背景下，人眼对颜色的敏感度为：绿色>红色>蓝色，故将 RGB 三个分量的敏感度权重分别赋为 1.35/4、1.35/3 和 1.35/6.5，^[2]并结合原模型，得到优化后的模型。

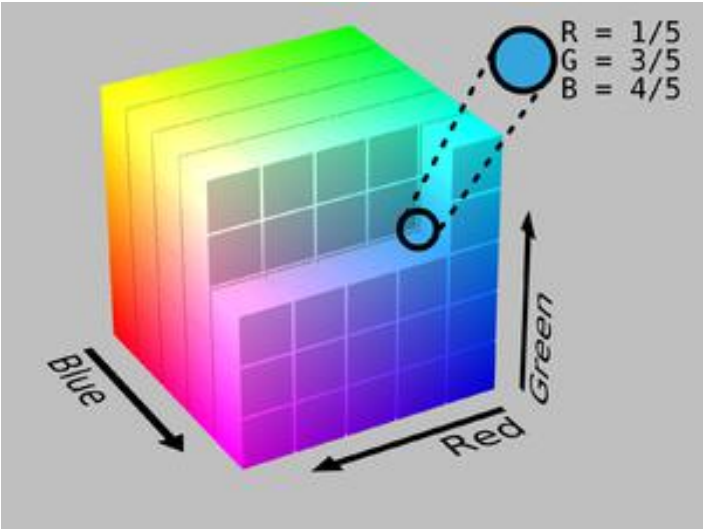


图 1 RGB 立方体^[1]

(R,G,B)	color
(0, 0, 0)	
(1, 1, 1)	

表 1 RGB 编码(0,0,0)与(1,1,1)颜色对比

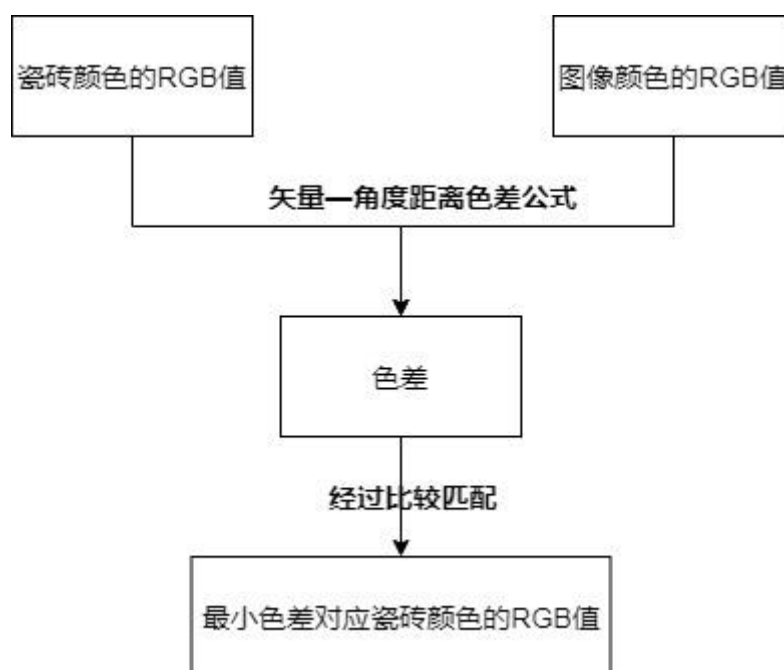


图 2 问题一的分析流程图

2.2 问题二的分析

问题二要求我们在忽略成本的基础上，只考虑拼接图像的表现力。在只增加 1 种颜色的条件下，如何最大限度的提高它的表现力，然后再考虑同时增加 2 种颜色至 10 种颜色的情况。

我们认为题述拼接图像的表现力就是指拼接图像对原始图像的还原程度，而基于问题一的分析，我们定义的色差能够反映两种颜色的相差程度，于是我们认为，将拼接时所有匹配瓷砖颜色与原始图像颜色的色差求和、求平均，得到的值能直接反映两幅图像的颜色相差程度，从而侧面反映拼接图像对原始图像的还原程度。

考虑生活中图像的多样性以及对模型普适性的追求，我们提出了一个假设模型，假设存在一幅包含所有 RGB 颜色，即 $255 \times 255 \times 255$ (1658 万) 种颜色的原始图像，根据问题一中计算色差的方法和现有的 22 种颜色，分别找到与 1658 万种颜色最接近的瓷砖颜色并累加它们的色差值，计算得出整幅画的色差值，并求平均，可以得到平均色差。将所有颜色 (1658 万种) 依次作为新增颜色加入到瓷砖颜色中，计算并比较每一种颜色加入后拼接图像与原始图像的平均色差，使平均色差最小的那一种新增颜色即为我们的选择。但在代码模拟的过程中，我们发现这样的模型计算量过于庞大，耗时过长，效率过低，所以我们希望能对原始模型进行简化。

首先我们进行了关于人眼阈值的一个实验，即探究人眼能辨别出颜色变化的粗略范围。我们采用计算器模拟调色的方式，生成了 RGB 数值为 (100, 0, 0) 的颜色，然后接连生成 RGB 值由 (100, 0, 1) 至 (100, 0, 10) 的颜色 (见表 2)。

(R,G,B)	color
(100, 0, 0)	
(100, 0, 1)	
(100, 0, 2)	
(100, 0, 3)	
(100, 0, 4)	
(100, 0, 5)	


(100, 0, 6)	
(100, 0, 7)	
(100, 0, 8)	
(100, 0, 9)	
(100, 0, 10)	

表 2 RGB(100,0,0)~RGB(100,0,10)

观察对比后发现，当颜色 RGB 的某一分量以较小幅度变化时，肉眼几乎无法分辨出颜色的变化。所以我们扩大倍数，分别生成了(100, 0, 0)至(100, 0, 80)的颜色(见表 3)。


(R,G,B)	color
(100, 0, 0)	
(100, 0, 10)	
(100, 0, 20)	
(100, 0, 30)	
(100, 0, 40)	
(100, 0, 50)	
(100, 0, 60)	
(100, 0, 70)	
(100, 0, 80)	

表 3 RGB(100,0,0)~(100,0,80)

通过实验，我们发现人眼对颜色变化的辨别能力有限，以至于从(100, 0, 0)一直增长到(100, 0, 30)，须通过对图像进行放大才能看出细微的变化，直到(100, 0, 40)的时候变化才较为明显。

因此，我们得出结论：只有当颜色的 RGB 分量变化大到 40 左右的时候，颜色的变化才会比较明显，为了避免实验存在偶然性和不同颜色分量的影响，以及人眼对于绿色较为敏感，我们又进行了在 R、B 分量保持不变的情况下，改变 G 分量的实验，对我们的结论进行验证，举例如下：

分别生成 RGB 编码为(0, 10, 0), (0, 20, 0), (0, 30, 0), (0, 40, 0)的颜色(见表 4)。


(R,G,B)	color
(100, 0, 0)	
(100, 10, 0)	
(100, 20, 0)	
(100, 30, 0)	
(100, 40, 0)	
(100, 50, 0)	
(100, 60, 0)	
(100, 70, 0)	
(100, 80, 0)	
(100, 90, 0)	
(100, 100, 0)	

表 4 RGB(100,0,0)~RGB(100,100,0)

我们发现对于人眼更敏感的绿色分量在变化 30 左右的时候，才会有较为明显的变化，虽然绿色分量的变化比其它两种颜色更快，但仍需产生较大变化时，人眼才能

分辨。然后，我们又进行了大量的实验发现红色对于蓝色的变色快慢相似，绿色最快。

最终得出结论：人眼对于颜色变化的识别能力比预期更弱，当 R 分量和 B 分量变化 40 左右，G 分量变化 30 左右的时候，人眼才能对颜色的改变做出一个较为准确的判断。

基于上述实验结论，我们对原有建立的模型进行优化，因为人眼的识别能力不足以区分 RGB 中 1658 万种颜色的细微差别，以及 1658 万种颜色计算量过于庞大，我们将模型进行简化。我们将 R 分量和 B 分量的阈值梯度范围选取为{0, 43, 86, 129, 172, 215, 255}，G 分量则选取为{0, 32, 64, 96, 128, 160, 192, 224, 255}，将原有的 1658 万种颜色简化为 $7 \times 7 \times 9(441)$ 种颜色，然后从中进行挑选最能提高还原度的那一种颜色。我们发现附件 1 中所给的 216 种颜色呈一定规律排布，其范围几乎涵盖了生活中人眼所能观察到的所有颜色。所以我们将待还原的原始图像由 1658 万种颜色简化为还原附件 1 所给的图像，筛选出最能提高还原度的那一种颜色。

我们发现人眼的识别范围有限，对于颜色及其细微的区别，人眼几乎不能做出判断，提出的 441 种颜色已经基本满足我们人眼对于现实生活中所有颜色的辨别。并且，将整体从 1658 万种简化为了 441 种，可以简化计算过程，提高效率。另一方面，我们提出的简化后的颜色范围也符合人视觉识别上的基本需要，具有一定的合理性。同时，根据视觉注意理论^{[6][7]}，仅当像素值的变化达到一定幅度时，人眼才能察觉到其变化，即人眼对于相近颜色不容易察觉它们之间的差别。因此，对于一幅彩色图像，进行颜色的量化处理，在一定范围内合并相似颜色，对于人眼区分显著目标的影响可以忽略不计的。且在一定程度上，对我们提出的优化起到支撑作用。

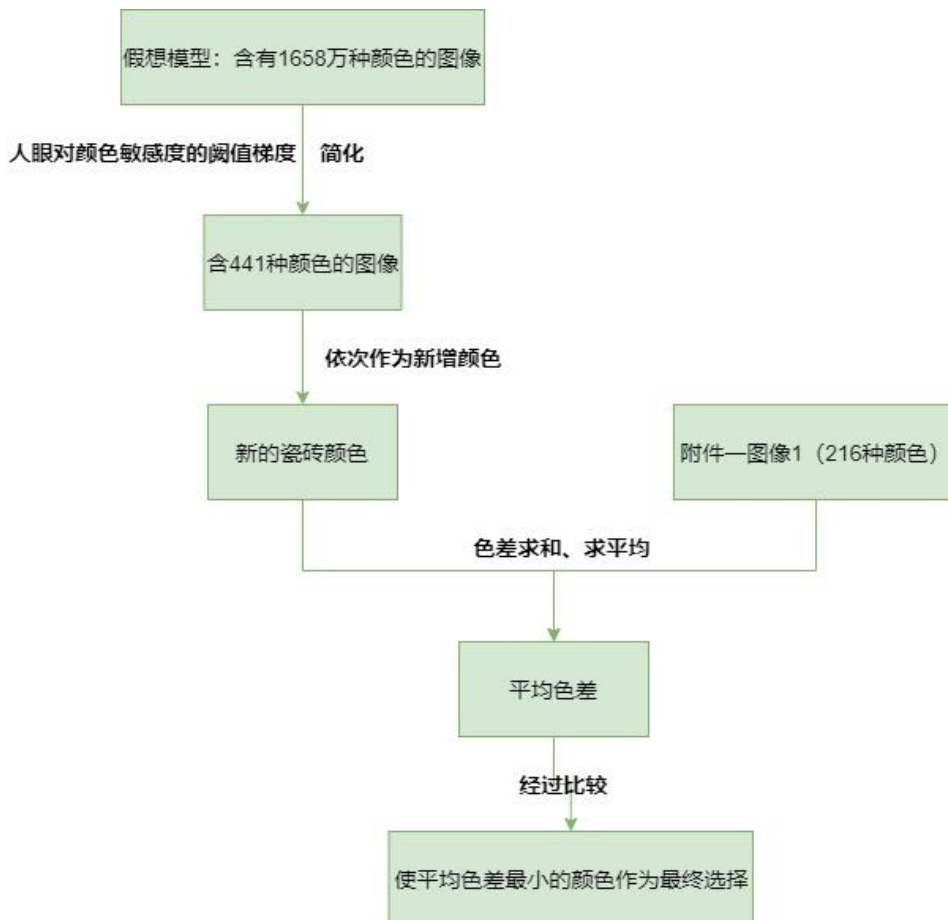


图 3 问题二的分析流程图

2.3 问题三的分析

问题三要求我们在兼顾成本和表现效果的前提下，如何添加尽量少的颜色，达到最好的表现效果。经过分析，考虑问题二和问题三所研究的问题有实质性的相似之处，其中最大的差别就是是否考虑到成本因素。

基于问题二的分析，我们先定义色彩还原度。首先，对反应图像相差程度的指标平均色差进行正向化处理(用色差最大值，即黑(0,0,0)白(255,255,255)色差的大小0.8123 减去平均色差和)，然后标准化处理(除以最大色差 0.8123)，得到范围为[0,1]的还原度，记作 ρ 。

我们希望在用尽量少的颜色的前提下，提高还原原图的效果，即还原度。于是，我们希望得到新增瓷砖颜色数目和还原度之间的关系。绘制散点图(见图 6)，并用曲线拟合，我们发现随着颜色种数的增加，新添加颜色后还原度整体上呈下降趋势，但曲线斜率减小，下降趋势趋缓。

为了进一步体现二者关系，我们用曲线拟合了新增瓷砖数目和还原度增量之间的关系。我们希望通过这些可视化数据，确定一个新增颜色种类阈值，在这个阈值我们以最少的成本得到最高的还原度收益。

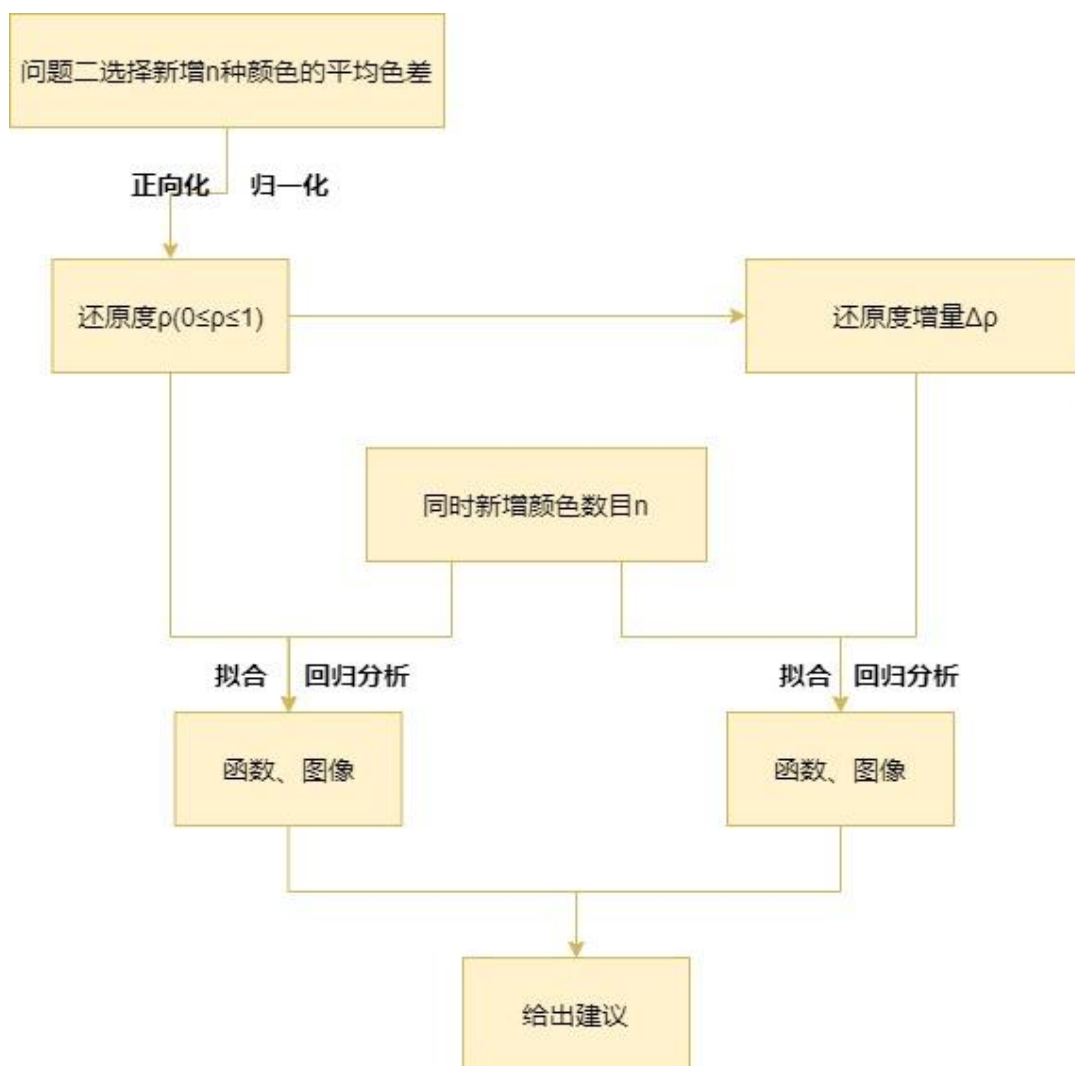


图 4 问题三的分析流程图

三、 模型假设

- 1) 假设原始图像的 R、G、B 三个分量的颜色均为 8 位；
- 2) 假设同时添加几种颜色和逐个添加对总体色差的影响相同；
- 3) 假设开发每一种颜色的成本和研发难度是相同的；
- 4) 假设 RGB 值为(0, 0, 0)的黑色与 RGB 值为(1, 1, 1)的黑色近似相等；
- 5) 假设以阶梯状排布的 441 种颜色在人眼识别范围内等同于 RGB 颜色表的 1658 万种颜色；
- 6) 假设 RGB 颜色分量在变化很小的情况下人眼无法辨别；
- 7) 假设附件 1 中的图像已经几乎包含了所有人眼可识别的颜色。

四、 符号说明

符号	说明
$D(x_i, x_j)$	x_i 和 x_j 在RGB颜色空间中的距离
ω	加权系数
d	色差
ρ	还原度
ξ	色差和
$\bar{\xi}$	平均色差
R^2	拟合优度

表 5 符号说明

五、 模型的建立与求解

5.1 问题一模型的建立与求解

5.1.1 模型的建立

对于 RGB 颜色空间中的色差问题，由于 RGB 不同分量之间对色差起的作用不尽相同，一般情况下，G 分量起到主导作用，为补偿 RGB 颜色空间的不均匀性，提高匹配马赛克瓷砖颜色时的准确性，本文采用基于杨振亚等提出的 RGB 色彩空间色差计算公式^[3]的优化模型，对问题进行解决，模型如下：

$$d = \sqrt{\frac{S_r^2 \cdot w_r \cdot (r_1 - r_2)^2 + S_g^2 \cdot w_g \cdot (g_1 - g_2)^2 + S_b^2 \cdot w_b \cdot (b_1 - b_2)^2}{(w_r + w_g + w_b) \cdot 255^2}} + S_\theta \cdot S_{\text{ratio}} \cdot \theta^2 \quad (2)$$

上式中各系数的意义和计算公式如下：

d 表示两个颜色之间的色差， ω_r 、 ω_g 、 ω_b 是人眼对红、绿、蓝三分量变化的敏感程度加权系数，此处 $(\omega_r, \omega_g, \omega_b)$ 取值为(1.35/4, 1.35/3, 1.35/6.5)。 S_r 、 S_g 、 S_b 表

示各个分量的重要性程度：

$$S_r = \min\left(\frac{r_1 + r_2}{\sum(r, g, b)}, 1\right) \quad (3)$$

$$S_g = \min\left(\frac{g_1 + g_2}{\sum(r, g, b)}, 1\right) \quad (4)$$

$$S_b = \min\left(\frac{b_1 + b_2}{\sum(r, g, b)}, 1\right) \quad (5)$$

$$\Sigma(r, g, b) = \frac{1}{3} (r_1 + r_2 + g_1 + g_2 + b_1 + b_2) \quad (6)$$

θ 表示两个待比较的颜色在 RGB 空间的矢量角度的归一化。

$$\theta = \frac{2}{\pi} \arccos\left(\frac{r_1 \cdot r_2 + g_1 \cdot g_2 + b_1 \cdot b_2}{\sqrt{(r_1^2 + g_1^2 + b_1^2) \cdot (r_2^2 + g_2^2 + b_2^2)}}\right) \quad (7)$$

当存在 RGB 值为(0, 0, 0)的颜色，即黑色时，公式(7)无法运算，所以我们用(1,1,1)近似替代(0,0,0)进行色差计算。

S_θ 用来考查红、绿、蓝三分量变化对两个待比较颜色在 RGB 空间的矢量角度的贡献，依据各个分量重要性程度来调整角度 θ 。

$$S_\theta = S_{\theta r} + S_{\theta g} + S_{\theta b} \quad (8)$$

$$S_{\theta r} = \frac{\frac{|r_1 - r_2|}{r_1 + r_2}}{\frac{|r_1 - r_2|}{r_1 + r_2} + \frac{|g_1 - g_2|}{g_1 + g_2} + \frac{|b_1 - b_2|}{b_1 + b_2}} \cdot S_r^2$$

$$(9) S_{\theta g} = \frac{\frac{|g_1 - g_2|}{g_1 + g_2}}{\frac{|r_1 - r_2|}{r_1 + r_2} + \frac{|g_1 - g_2|}{g_1 + g_2} + \frac{|b_1 - b_2|}{b_1 + b_2}} \cdot S_g^2$$

$$(10) S_{\theta b} = \frac{\frac{|b_1 - b_2|}{b_1 + b_2}}{\frac{|r_1 - r_2|}{r_1 + r_2} + \frac{|g_1 - g_2|}{g_1 + g_2} + \frac{|b_1 - b_2|}{b_1 + b_2}} \cdot S_b^2$$

(11)

当 RGB 颜色有为 0 值时，如果 $r_1=r_2=0$ 时，原三维空间系统降为二维空间系统，此时：

$$S_\theta = S_{\theta g} + S_{\theta b} \quad (12)$$

$$S_{\theta g} = \frac{\frac{|g_1 - g_2|}{g_1 + g_2}}{\frac{|g_1 - g_2|}{g_1 + g_2} + \frac{|b_1 - b_2|}{b_1 + b_2}} \cdot S_g^2 \quad (13)$$

$$S_{\theta b} = \frac{\frac{|b_1 - b_2|}{b_1 + b_2}}{\frac{|g_1 - g_2|}{g_1 + g_2} + \frac{|b_1 - b_2|}{b_1 + b_2}} \cdot S_b^2 \quad (14)$$

当 G 值或 B 值等于 0 时也同理如此。而当有两种颜色分量都为 0 时，系统则降为一维坐标系统，例如当 R 值和 G 值都为 0 时，所用公式应为：

$$S_{\theta} = S_{\theta b} \quad (15)$$

$$S_{\theta b} = S_b^2 \quad (16)$$

另外情况当 R 值和 B 值都为 0 或 G 值和 B 值都为 0 时也同理如此。此时我们所分析的情况就可以解决所有情况。

S_{ratio} 用于调整系数防止在 RGB 空间底部 θ 过大。

$$S_{ratio} = \frac{\max(r_1, r_2, g_1, g_2, b_1, b_2)}{255}$$

(17)

5.1.2 模型的求解

该问模型利用 Matlab 进行求解，步骤如下：

- 1) 输入已有的 22 种颜色数据，得到一个 22*3 的矩阵 color(见附录 9.1.2);
- 2) 定义 getdist 函数，将图像中的每一个颜色与已有的 22 个颜色计算色差 d ，(源代码见附录 9.1.1);
- 3) 定义 Result 函数，通过 for 循环的嵌套，输出待求解颜色和对应已知颜色一一对应的编号(源代码见附录 9.1.1)。

5.1.3 模型的结果

根据 5.1.2 我们计算出图像 1 和图像 2 中颜色在已知颜色中的相似结果，输出在 result1.txt 和 result2.txt(见附录 9.1.2 表)

5.2 问题二模型的建立与求解

5.2.1 模型的建立

基于 2.2 中对问题二的分析，我们认为色差和越大，说明我们还原这张图的效果越不理想，即拼接图像的表现力越差。并且建立模型时，因为同时添加的计算量过于庞大，故我们假设同时添加几种颜色和逐个添加对总体色差的影响相同，仍采用 5.1.1 中的公式(2)计算色差，并采用如下模型：

定义色差和为已有颜色对于待还原的颜色的，记作 ξ

$$\xi = \sum_{i=1}^{num} d_i$$

(18)

公式 18 中, d_i 为已有颜色与待还原图像中第 i 种颜色的色差, num 为待还原图像中的颜色种数。为了减小误差, 对色差和 ξ 取平均值, 得平均色差记作 $\bar{\xi}$;

$$\bar{\xi} = \frac{\xi}{num} \quad (19)$$

基于 2.2 中的分析和模型假设, 我们将原题目中要求同时增加多种的颜色的要求转化为逐步增加进行计算, 结果见 5.2.3。

5.2.2 模型的求解

该问模型利用 Matlab 进行求解(具体代码见附录 9.2.1), 步骤如下:

- 1) 根据预设的 RGB 三基色的人眼阈值梯度, 利用循环构造简化后的 441 种颜色的 RGB 编码矩阵, 包含 441 种颜色中部分颜色的编码矩阵(见表 6-8);

R	G	B	color
0	0	0	
0	0	43	
0	0	86	
0	0	129	
0	0	172	
0	0	215	
0	0	255	

表 6 部分颜色的 RGB 值

R	G	B	color
0	160	0	
0	160	43	
0	160	86	
0	160	129	

表 7 部分颜色的 RGB 值

R	G	B	color
215	0	0	
215	0	43	
215	0	86	
215	0	129	
215	0	172	
215	0	215	

表 8 部分颜色的 RGB 值

- 2) 遍历 441 种颜色的 RGB 编码值, 分别将其中一种颜色加入已有瓷砖颜色种类中作为新增的瓷砖颜色, 利用 Result 函数将新的瓷砖颜色矩阵与样本图像进行匹配, 得到所有匹配结果的色差 d ;
- 3) 对这一匹配结果的色差值进行累加, 求平均, 可得到瓷砖拼接结果和原图像

- 的平均色差 $\bar{\xi}$ (平均色差越大, 色彩还原度和表现力越低);
- 4) 循环 441 次后, 通过比较所有颜色选择对应求得平均色差, 筛选出使得平均色差最小的新增颜色(样本实验结果为 RGB 编码(86,32,215)), 将其 RGB 编码值作为新增项加入原瓷砖颜色 RGB 矩阵;
 - 5) 以新的瓷砖颜色矩阵作为已有颜色, 加入下一次循环, 重复步骤(2)(3)(4);
 - 6) 将最终结果放入 result(见表 9);

数量	R	G	B	平均色差
1	86	32	215	0.116575315
2	86	224	129	0.111142968
3	129	64	43	0.105777979
4	129	64	215	0.101152229
5	86	32	129	0.098272724
6	43	224	86	0.095483954
7	129	64	86	0.092822983
8	43	192	129	0.090207316
9	86	96	255	0.087701361
10	86	192	255	0.085381952
11	129	96	129	0.083251326
12	172	32	43	0.081146779
13	86	64	255	0.079144186
14	129	128	43	0.077256082
15	129	160	129	0.075508396
16	43	224	43	0.073795858
17	129	192	172	0.07216544
18	129	192	86	0.070570856
19	129	160	255	0.069141868
20	172	32	215	0.067766145

表 9 添加的颜色和平均色差

对表 9 作出如下说明:

若需新增 i 种颜色, 表格中第 1 行至第 i 行即为新增的 i 种颜色;

- 7) 用 excel 做出增加颜色的数量和平均色差和之间的散点图(见图 5);

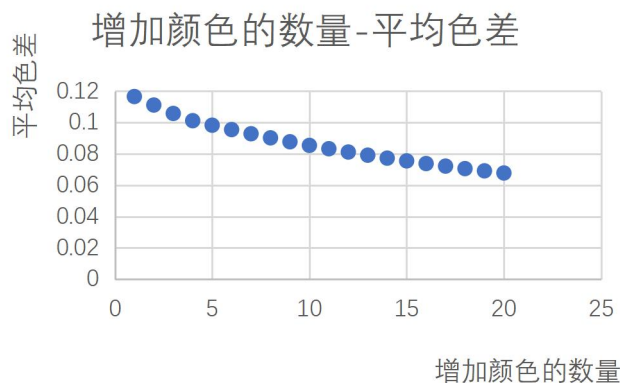


图 5 增加颜色的数量和平均色差的散点图

5.2.3 模型的结果

根据 5.2.2, 我们计算出同时添加一个至十个颜色时, 若要使拼接图像的表现力增

强，应添加的颜色和添加后对于待还原图像的平均色差。

我们由图 2 大致观察出，随增加瓷砖颜色种类的增多，拼接图像的色差在减小，且减小的速率越来越慢，曲线趋于平稳。故我们可以利用 MATLAB 的 cftool 得出如下拟合曲线：

拟合曲线	R^2
$y = (8.385e - 05)\chi^2 - 0.004153\chi + 0.1183$	0.9957
$y = (-5.267e - 06)\chi^3 + 0.0002498\chi^2 - 0.005581\chi + 0.1211$	0.9988

表 10 几种拟合曲线及对应的拟合优度

由表 10 可得，拟合优度 R^2 接近于 1，说明拟合效果好。

5.3 问题三模型的建立与求解

5.3.1 模型的建立

基于 2.3 中对问题三的分析，我们对新增加的颜色还原度增量做出比较，将还原度做量化处理，进而计算和比较出还原度增量，从而计算出的是已有颜色对于还原图像的平均色差，模型如下：

在 RGB 颜色空间中，黑色(0, 0, 0)和白色(255, 255, 255)的色差是最大的，经过计算得到 $d_{\max} = 0.8123$ 。

首先，定义已有颜色对于原图像的还原度为 ρ ，将 0 定为零还原，将 1 定为完全还原， ρ 的取值范围是 [0,1]。

已知用已有颜色还原图像产生的平均色差越小，表示它的还原度越大，为了方便计算，且更直观地反映还原图像的还原度，将平均色差这一变量正向化并归一化。

还原度 ρ 表达式如下：

$$\rho = \frac{d_{\max} - \bar{\xi}}{d_{\max}} \quad (20)$$

公式 20 中， d_{\max} 表示黑色与白色之间的色差，恒等于 0.8123； $\bar{\xi}$ 表示平均色差。

将新增加的 1 种颜色所产生的还原度增量，记作 $\Delta\rho$ 。

5.3.2 模型的求解

该问利用 Excel 辅助我们对数据进行分析 and 处理。步骤如下：

- 1) 利用 5.3.1 中提出的模型，分别计算出增加一种颜色至二十种颜色的还原度，并计算出其增量；
- 2) 将增加的颜色种数和还原度增量的数据输入 Excel(见表 12)；
- 3) 在 Excel 中做出散点图，并用平滑曲线连接各点(见图 6)；

数量	R	G	B	平均色差 $\bar{\xi}$	还原度 ρ
1	86	32	215	0.116575315	0.856487363

2	86	224	129	0.111142968	0.863174975
3	129	64	43	0.105777979	0.869779664
4	129	64	215	0.101152229	0.875474296
5	86	32	129	0.098272724	0.879019176
6	43	224	86	0.095483954	0.882452353
7	129	64	86	0.092822983	0.885728201
8	43	192	129	0.090207316	0.888948275
9	86	96	255	0.087701361	0.892033287
10	86	192	255	0.085381952	0.894888647
11	129	96	129	0.083251326	0.897511602
12	172	32	43	0.081146779	0.900102451
13	86	64	255	0.079144186	0.902567788
14	129	128	43	0.077256082	0.904892181
15	129	160	129	0.075508396	0.907043708
16	43	224	43	0.073795858	0.909151966
17	129	192	172	0.07216544	0.911159129
18	129	192	86	0.070570856	0.913122176
19	129	160	255	0.069141868	0.914881364
20	172	32	215	0.067766145	0.916574979

表 11 增加颜色的种数和平均色差、还原度之间的关系

增加的种数	还原度增量
1	0.014960945
2	0.006687612
3	0.006604689
4	0.005694632
5	0.00354488
6	0.003433177
7	0.003275848
8	0.003220075
9	0.003085011
10	0.00285536
11	0.002622955
12	0.002590849
13	0.002465337
14	0.002324393
15	0.002151527
16	0.002108258
17	0.002007162
18	0.001963047
19	0.001759188
20	0.001693615

表 12 增加颜色的种数和还原度增量的关系

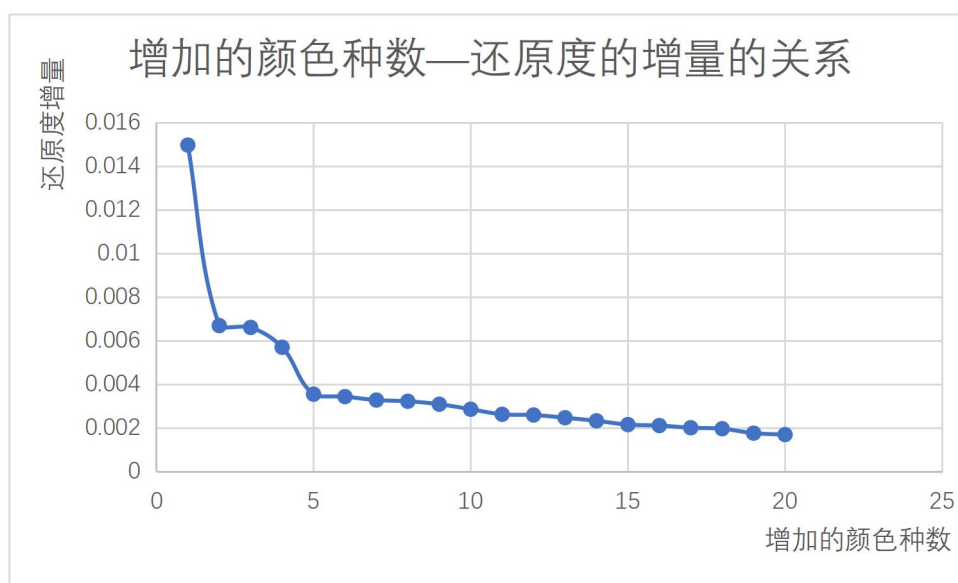


图 6 增加的颜色种数—还原度的增量的关系

5.3.3 模型的结果

根据表 11 和图 6 的综合分析后,我们发现当新增 1 种颜色,还原度有了明显提升,在增加第二种颜色的时候,其还原度增量虽然骤降但相对而言依然较高,在增加第四种颜色的时候,还原度增量发生了骤降,之后则呈慢下降的趋势,所以综合研发成本和表现效果,我们认为四种颜色就是新增颜色阈值,从新增第五种颜色开始,还原度的收益大大降低,所以应该增加四种颜色,RGB 编码分别为(86,32,215), (86,224,129), (129,64,43)和(129,64,215)。

六、模型的分析与检验

6.1 模型的分析

6.1.1 稳定性分析

基于杨振亚等人提出的色差计算公式,其最开始提出人眼对于 RGB 三种不同颜色的敏感度不同,但其增加的权重比例不够严谨导致公式的稳定性很低。在进一步优化的过程中,根据苑亚会研究的基于 RGB 视觉敏感度的显著性检测方法,设计人眼对于 RGB 三原色的敏感度测试实验,通过大量实验数据将其颜色量化,在黑色背景下,采用 RGB 敏感度的加权颜色距离的计算方法和基于 RGB 敏感度的加权区域距离的计算方法,提出并完善了人眼对于 RGB 三原色的权重比。引入到色差公式中,使其稳定性和严谨性大幅度提高可以计算不同颜色的色差。而在实际生活中,我们会面对除黑色外的其他颜色背景,通过实验后发现证明,根据不同背景颜色,我们可以确立 RGB 不同的权重比,进而使最后计算的结果更加具有稳定性。

6.1.2 误差分析

1) 黑色近似后计算色差的误差

由于本实验中所涉及的与 RGB 值为(0, 0, 0)黑色的计算都近似为 RGB 值(1, 1, 1),所以在计算色差过程中会产生误差,但是因为人眼所识别的能力范围内, (0, 0, 0)与(1, 1, 1)两者的颜色几乎无法被区分,所以该误差可以近似忽略;

2) 拟合增加颜色数量-平均色差散点图所产生的误差

用 Excel 拟合增加颜色数量-平均色差散点图的时候,为保证拟合出来一条平滑的曲线并选择函数,所以模型计算出的平均色差会存在不能和曲线完全拟合的情况,导致模型产生误差。不过由拟合图像效果来看,二者的拟合程度几乎相等,其产生的误差可以忽略不计。

七、 模型的评价

7.1 模型的优点

- 1) 问题一中,我们的模型基于人眼对 RGB 颜色分量不同的敏感度,分别赋予 RGB 三个颜色分量不同的权重,在一定程度上补偿了由 RGB 颜色空间的不均匀性带来的影响;
- 2) 问题二中,我们在简单分析人眼辨识颜色变化的基本规律,对模型进行了合理的简化,相对于原始模型提升了运算效率,且为后续进行色差等相关计算准备了条件;
- 3) 问题二中,我们对增加颜色的种数和平均色差之间的关系进行了曲线拟合,可预测它们之间的变化趋势,具有较强的普适性;
- 4) 本文原创性高,富有创思维,文章中大部分模型都是自行完成推导到建立的过程。

7.2 模型的缺点

- 1) 问题一中,我们用于得到权重的实验^[2]是在黑色背景下进行的,在实际生活中应该根据不同颜色背景,对 RGB 颜色分量权重重新进行实验测定;
- 2) 对模型的部分参数通过实际情况进行了估计,对结果产生一定误差。

7.3 模型的改进

- 1) 研究人眼在不同颜色背景下对 RGB 颜色分量的敏感度,进而调整 RGB 颜色分量的权重。

7.4 模型的推广

- 1) 对于生活中,基于 RGB 颜色空间的选色问题,本文所提出的模型也具有一定的适用性和参考价值。

八、 参考文献

- [1] 维基百科.Wikipedia explanation of RGB color space [EB/OL].互联网,2021.2,
https://en.jinzhaowiki/wiki/RGB_color_space
- [2] 苑亚会.基于 RGB 视觉敏感度的显著性检测方法研究[D].山东大学,2016.
- [3] 杨振亚,王勇,杨振东,王成道.RGB 颜色空间的矢量-角度距离色差公式[J].计算机工程与应用,2010,46(06):154-156.
- [4] Androutsos D, Plataniotis K N, Venetsanopoulos A N. A novel vector based approach to color image retrieval using a vector angular based distance measure[J]. Computer Vision and Image Understanding, 1999, 75(1): 46—58.
- [5] Androutsos D, Plataniotis K N, Venetsanopoulos A N. Vector angular distance measure for indexing and retrieval of color[C]//SPIE Proceedings of Storage and Retrieval for

- Image and Video Databases VII, San Jose, CA, 1999, 3656: 604—613.
- [6] 维基百科.Wikipedia explanation of luminosity function[EB/OL].互联网,2016.02 ,
https://en.wikipedia.org/wiki/ Luminosity function.
- [7] Robert Desimone, John Duncan, Neural mechanisms of selective visual
attention[J].Annual review of neuroscience.1995,vol.18,no.1,pp.193-222
- [8] 2014 年格子马赛克瓷砖引领时尚[J].陶瓷,2014(10):68.

九、 附录

9.1 问题一代码及数据

9.1.1 代码

Matlab 代码:

```
1) main.m
format='%d %*d %d %d %d %*d';
[x,r,g,b]=textread('image_1.txt',format,'headerlines',1,'delimiter',',()'); %#ok<DTX
TRD>
%x 存放图像中的颜色编号， r， g， b 存放 RGB 编码
color=[0,0,0;255,255,255;255,0,0;246,232,9;72,176,64;27,115,186;53,118,84;244,
181,208;255,145,0;177,125,85;92,59,144;11,222,222;
228,0,130;255,218,32;118,238,0;17,168,226;255,110,0;201,202,202;255,249,177;
179,226,242;249,225,214;186,149,195];
%color 存放马赛克瓷砖厂生产的瓷砖的现有颜色的 RGB 编码
result=Result(x,r,g,b,color);
%result 存放根据算法匹配的瓷砖编号和色差
id=fopen('result1.txt','wt');
fprintf(id,'序号， 瓷砖颜色编号\n');

%第二问
for i=1:size(x,1)
    fprintf(id,'%1d,%d\n',x(i),result(i));
end
fclose(id);
% 将结果输出到文件
% 选择优先增加的瓷砖颜色
dis=0;%re 存放用现有瓷砖还原图像各个颜色的色差和
for i=1:216
    dis=dis+result(i,2);
    %累加存放在 result 第二列的所有匹配的瓷砖与图像颜色的色差
end
Dist=dis/216;
%将色差和求平均
color_dist=select(x,r,g,b,Dist);
```

%select 函数返回一个 20*4 的矩阵，存放增加的颜色对应 RGB 编码和增加后对图像还原的平均色差和（反应还原程度）

```
color_dist_reduc=ones(20,5);
for i=1:20
    color_dist_reduc(i,1)=color_dist(i,1);
    color_dist_reduc(i,2)=color_dist(i,2);
    color_dist_reduc(i,3)=color_dist(i,3);
    color_dist_reduc(i,4)=color_dist(i,4);
    color_dist_reduc(i,5)=dist_reduc(color_dist(i,4));
end
%将平均色差和经过正向化和归一化得到还原度 reduc
```

2) getdist.m(计算色差)

```
function[dist]=getdist(r1,r2,g1,g2,b1,b2)
wr=1.35/4;wg=1.35/3;wb=1.35/6.5;
%人眼对红绿蓝三分量变化的敏感程度加权系数
if r1==0&&g1==0&&b1==0
    r1=1;
    g1=1;
    b1=1;
end
if r2==0&&g2==0&&b2==0
    r2=1;
    g2=1;
    b2=1;
end
%用(1,1,1)替代(0,0,0)
Sr=min((r1+r2)/(3*(r1+r2+g1+g2+b1+b2)),1);
Sg=min(g1+g2/(3*(r1+r2+g1+g2+b1+b2)),1);
Sb=min(b1+b2/(3*(r1+r2+g1+g2+b1+b2)),1);
%各个分量的重要性程度
Sratio=max([r1,r2,g1,g2,b1,b2])/255;
% 用于调整系数防止在 RGB 空间底部 Theta 过大
Theta=2/pi*acos((r1*r2+g1*g2+b1*b2)/sqrt((r1^2+g1^2+b1^2)*(r2^2+g2^2+b2^2)));
%代表两个待比较颜色在 RGB 空间的矢量角度的贡献
if r1==0&&r2==0
    r1=1;
    r2=1;
end
if g1==0&&g2==0
    g1=1;
    g2=1;
end
if b1==0&&b2==0
```

```

        b1=1;
        b2=1;
    end
    %若基色参数值存在同时为零的情况，将对应参数置1 将三维公式展开为二维公式
    Sthetar= Sr^2*abs(r1-r2)/(r1+r2)/(abs(r1-r2)/(r1+r2)+abs(g1-g2)/(g1+g2)+abs(b1-b2)/(b1+b2));
    Sthetag= Sg^2*abs(g1-g2)/(g1+g2)/(abs(r1-r2)/(r1+r2)+abs(g1-g2)/(g1+g2)+abs(b1-b2)/(b1+b2));
    Sthetab= Sb^2*abs(b1-b2)/(b1+b2)/(abs(r1-r2)/(r1+r2)+abs(g1-g2)/(g1+g2)+abs(b1-b2)/(b1+b2));
    Stheta=Sthetar+Sthetag+Sthetab;
    %考察红绿蓝三分量变化对两个待比较颜色在 RGB 空间的矢量角度的贡献
    dist=sqrt((Sr^2*wr*(r1-r2)^2+Sg^2*wg*(g1-g2)^2+Sb^2*wb*(b1-b2)^2)/((wr+wg+wb)*255^2)+Stheta*Sratio*Theta^2);
    end
3) result.m(输出与代求颜色相似的颜色)
function[result]=Result(x,r,g,b,color)
s=size(x,1);
result=ones(s,2);
for i=1:s
    result(i,2)=0.8123;%黑白色差(最大色差)
    for j=1:size(color,1)
        dist=getdist(r(i),color(j,1),g(i),color(j,2),b(i),color(j,3));
        %getdist 函数获取两种颜色的色差
        if dist<=result(i,2)
            result(i,2)=dist;
            result(i,1)=j;
            %经过比较将和第 i 种颜色色差最小的瓷砖编号和 RGB 值放入
        end
    end
end
end
end
end

```

9.1.2 数据

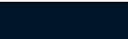

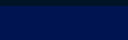















































1) 矩阵 Color(存储已有的 22 种颜色):
















































































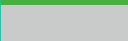








R	G	B
0	0	0
255	255	255
255	0	0
246	232	9
72	176	64
27	115	186

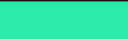























































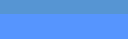































53	118	84
244	181	208
255	145	0
177	125	85
92	59	144
11	222	222
228	0	130
255	218	32
118	238	0
17	168	226
255	110	0
201	202	202
255	249	177
179	226	242
249	225	214
186	149	195

表 13

2) result1.excel(用已知 22 种颜色找出图像 1 中相似的颜色):

序号	R	G	B	color	color	编号	R	G	B
1	0	20	39			1	0	0	0
2	0	20	82			11	92	59	144
3	0	20	125			11	92	59	144
4	0	20	168			11	92	59	144
5	0	20	211			11	92	59	144
6	0	20	254			11	92	59	144
7	0	63	39			7	53	118	84
8	0	63	82			7	53	118	84
9	0	63	125			11	92	59	144
10	0	63	168			11	92	59	144
11	0	63	211			6	27	115	186
12	0	63	254			6	27	115	186
13	0	106	39			7	53	118	84
14	0	106	82			7	53	118	84
15	0	106	125			7	53	118	84
16	0	106	168			6	27	115	186
17	0	106	211			6	27	115	186
18	0	106	254			6	27	115	186
19	0	149	39			5	72	176	64
20	0	149	82			7	53	118	84
21	0	149	125			7	53	118	84
22	0	149	168			6	27	115	186
23	0	149	211			16	17	168	226
24	0	149	254			16	17	168	226
25	0	192	39			5	72	176	64

26	0	192	82			5	72	176	64
27	0	192	125			5	72	176	64
28	0	192	168			12	11	222	222
29	0	192	211			16	17	168	226
30	0	192	254			16	17	168	226
31	0	235	39			5	72	176	64
32	0	235	82			5	72	176	64
33	0	235	125			12	11	222	222
34	0	235	168			12	11	222	222
35	0	235	211			12	11	222	222
36	0	235	254			12	11	222	222
37	43	20	39			1	0	0	0
38	43	20	82			11	92	59	144
39	43	20	125			11	92	59	144
40	43	20	168			11	92	59	144
41	43	20	211			11	92	59	144
42	43	20	254			11	92	59	144
43	43	63	39			7	53	118	84
44	43	63	82			11	92	59	144
45	43	63	125			11	92	59	144
46	43	63	168			11	92	59	144
47	43	63	211			11	92	59	144
48	43	63	254			6	27	115	186
49	43	106	39			7	53	118	84
50	43	106	82			7	53	118	84
51	43	106	125			7	53	118	84
52	43	106	168			6	27	115	186
53	43	106	211			6	27	115	186
54	43	106	254			6	27	115	186
55	43	149	39			5	72	176	64
56	43	149	82			7	53	118	84
57	43	149	125			7	53	118	84
58	43	149	168			22	186	149	195
59	43	149	211			16	17	168	226
60	43	149	254			16	17	168	226
61	43	192	39			5	72	176	64
62	43	192	82			5	72	176	64
63	43	192	125			5	72	176	64
64	43	192	168			18	201	202	202
65	43	192	211			16	17	168	226
66	43	192	254			16	17	168	226
67	43	235	39			15	118	238	0
68	43	235	82			5	72	176	64
69	43	235	125			19	255	249	177

70	43	235	168			19	255	249	177
71	43	235	211			12	11	222	222
72	43	235	254			12	11	222	222
73	86	20	39			1	0	0	0
74	86	20	82			13	228	0	130
75	86	20	125			11	92	59	144
76	86	20	168			11	92	59	144
77	86	20	211			11	92	59	144
78	86	20	254			11	92	59	144
79	86	63	39			10	177	125	85
80	86	63	82			11	92	59	144
81	86	63	125			11	92	59	144
82	86	63	168			11	92	59	144
83	86	63	211			11	92	59	144
84	86	63	254			6	27	115	186
85	86	106	39			10	177	125	85
86	86	106	82			7	53	118	84
87	86	106	125			7	53	118	84
88	86	106	168			6	27	115	186
89	86	106	211			6	27	115	186
90	86	106	254			6	27	115	186
91	86	149	39			5	72	176	64
92	86	149	82			7	53	118	84
93	86	149	125			7	53	118	84
94	86	149	168			22	186	149	195
95	86	149	211			22	186	149	195
96	86	149	254			16	17	168	226
97	86	192	39			5	72	176	64
98	86	192	82			5	72	176	64
99	86	192	125			5	72	176	64
100	86	192	168			18	201	202	202
101	86	192	211			18	201	202	202
102	86	192	254			16	17	168	226
103	86	235	39			15	118	238	0
104	86	235	82			5	72	176	64
105	86	235	125			19	255	249	177
106	86	235	168			19	255	249	177
107	86	235	211			12	11	222	222
108	86	235	254			20	179	226	242
109	129	20	39			3	255	0	0
110	129	20	82			13	228	0	130
111	129	20	125			13	228	0	130
112	129	20	168			11	92	59	144
113	129	20	211			11	92	59	144

114	129	20	254			11	92	59	144
115	129	63	39			10	177	125	85
116	129	63	82			10	177	125	85
117	129	63	125			11	92	59	144
118	129	63	168			11	92	59	144
119	129	63	211			11	92	59	144
120	129	63	254			11	92	59	144
121	129	106	39			10	177	125	85
122	129	106	82			10	177	125	85
123	129	106	125			7	53	118	84
124	129	106	168			6	27	115	186
125	129	106	211			6	27	115	186
126	129	106	254			6	27	115	186
127	129	149	39			5	72	176	64
128	129	149	82			10	177	125	85
129	129	149	125			7	53	118	84
130	129	149	168			22	186	149	195
131	129	149	211			22	186	149	195
132	129	149	254			16	17	168	226
133	129	192	39			5	72	176	64
134	129	192	82			5	72	176	64
135	129	192	125			5	72	176	64
136	129	192	168			18	201	202	202
137	129	192	211			18	201	202	202
138	129	192	254			20	179	226	242
139	129	235	39			15	118	238	0
140	129	235	82			5	72	176	64
141	129	235	125			19	255	249	177
142	129	235	168			19	255	249	177
143	129	235	211			21	249	225	214
144	129	235	254			20	179	226	242
145	172	20	39			3	255	0	0
146	172	20	82			13	228	0	130
147	172	20	125			13	228	0	130
148	172	20	168			13	228	0	130
149	172	20	211			11	92	59	144
150	172	20	254			11	92	59	144
151	172	63	39			17	255	110	0
152	172	63	82			10	177	125	85
153	172	63	125			11	92	59	144
154	172	63	168			11	92	59	144
155	172	63	211			11	92	59	144
156	172	63	254			11	92	59	144
157	172	106	39			10	177	125	85

158	172	106	82			10	177	125	85
159	172	106	125			10	177	125	85
160	172	106	168			6	27	115	186
161	172	106	211			6	27	115	186
162	172	106	254			6	27	115	186
163	172	149	39			10	177	125	85
164	172	149	82			10	177	125	85
165	172	149	125			10	177	125	85
166	172	149	168			22	186	149	195
167	172	149	211			22	186	149	195
168	172	149	254			16	17	168	226
169	172	192	39			14	255	218	32
170	172	192	82			5	72	176	64
171	172	192	125			18	201	202	202
172	172	192	168			18	201	202	202
173	172	192	211			18	201	202	202
174	172	192	254			20	179	226	242
175	172	235	39			14	255	218	32
176	172	235	82			5	72	176	64
177	172	235	125			19	255	249	177
178	172	235	168			19	255	249	177
179	172	235	211			21	249	225	214
180	172	235	254			20	179	226	242
181	215	20	39			3	255	0	0
182	215	20	82			13	228	0	130
183	215	20	125			13	228	0	130
184	215	20	168			13	228	0	130
185	215	20	211			13	228	0	130
186	215	20	254			11	92	59	144
187	215	63	39			17	255	110	0
188	215	63	82			10	177	125	85
189	215	63	125			11	92	59	144
190	215	63	168			11	92	59	144
191	215	63	211			11	92	59	144
192	215	63	254			11	92	59	144
193	215	106	39			17	255	110	0
194	215	106	82			10	177	125	85
195	215	106	125			10	177	125	85
196	215	106	168			6	27	115	186
197	215	106	211			6	27	115	186
198	215	106	254			22	186	149	195
199	215	149	39			9	255	145	0
200	215	149	82			10	177	125	85
201	215	149	125			10	177	125	85























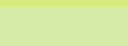

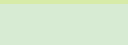
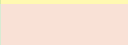
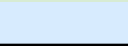
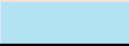
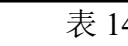

202	215	149	168			22	186	149	195
203	215	149	211			22	186	149	195
204	215	149	254			22	186	149	195
205	215	192	39			14	255	218	32
206	215	192	82			5	72	176	64
207	215	192	125			19	255	249	177
208	215	192	168			18	201	202	202
209	215	192	211			18	201	202	202
210	215	192	254			20	179	226	242
211	215	235	39			14	255	218	32
212	215	235	82			14	255	218	32
213	215	235	125			19	255	249	177
214	215	235	168			19	255	249	177
215	215	235	211			21	249	225	214
216	215	235	254			20	179	226	242

表 14 result1

3) result1.txt(图像 1)

序号	瓷砖颜色编号
1	1
2	11
3	11
4	11
5	11
6	11
7	7
8	7
9	11
10	11
11	6
12	6
13	7
14	7
15	7
16	6
17	6
18	6
19	5
20	7
21	7
22	6
23	16
24	16
25	5

26	5
27	5
28	12
29	16
30	16
31	5
32	5
33	12
34	12
35	12
36	12
37	1
38	11
39	11
40	11
41	11
42	11
43	7
44	11
45	11
46	11
47	11
48	6
49	7
50	7
51	7
52	6
53	6
54	6
55	5
56	7
57	7
58	22
59	16
60	16
61	5
62	5
63	5
64	18
65	16
66	16
67	15
68	5
69	19

70	19
71	12
72	12
73	1
74	13
75	11
76	11
77	11
78	11
79	10
80	11
81	11
82	11
83	11
84	6
85	10
86	7
87	7
88	6
89	6
90	6
91	5
92	7
93	7
94	22
95	22
96	16
97	5
98	5
99	5
100	18
101	18
102	16
103	15
104	5
105	19
106	19
107	12
108	20
109	3
110	13
111	13
112	11
113	11

114	11
115	10
116	10
117	11
118	11
119	11
120	11
121	10
122	10
123	7
124	6
125	6
126	6
127	5
128	10
129	7
130	22
131	22
132	16
133	5
134	5
135	5
136	18
137	18
138	20
139	15
140	5
141	19
142	19
143	21
144	20
145	3
146	13
147	13
148	13
149	11
150	11
151	17
152	10
153	11
154	11
155	11
156	11
157	10

158	10
159	10
160	6
161	6
162	6
163	10
164	10
165	10
166	22
167	22
168	16
169	14
170	5
171	18
172	18
173	18
174	20
175	14
176	5
177	19
178	19
179	21
180	20
181	3
182	13
183	13
184	13
185	13
186	11
187	17
188	10
189	11
190	11
191	11
192	11
193	17
194	10
195	10
196	6
197	6
198	22
199	9
200	10
201	10

202	22
203	22
204	22
205	14
206	5
207	19
208	18
209	18
210	20
211	14
212	14
213	19
214	19
215	21
216	20

表 15 result1

4) Result2(图像 2):

序号	瓷砖颜色编号
1	12
2	5
3	6
4	6
5	5
6	12
7	11
8	7
9	6
10	16
11	6
12	7
13	16
14	19
15	6
16	5
17	12
18	5
19	12
20	5
21	11
22	7
23	5
24	7
25	6

26	5
27	11
28	6
29	7
30	5
31	12
32	5
33	11
34	7
35	7
36	5
37	18
38	18
39	11
40	7
41	7
42	15
43	5
44	11
45	11
46	7
47	7
48	6
49	11
50	6
51	16
52	16
53	16
54	11
55	11
56	7
57	11
58	16
59	11
60	11
61	6
62	15
63	6
64	18
65	6
66	5
67	18
68	10
69	5

70	18
71	3
72	11
73	18
74	11
75	6
76	6
77	9
78	22
79	13
80	11
81	11
82	11
83	18
84	5
85	19
86	7
87	11
88	11
89	16
90	18
91	5
92	11
93	11
94	10
95	11
96	14
97	13
98	19
99	22
100	10
101	11
102	6
103	14
104	5
105	11
106	6
107	10
108	10
109	10
110	14
111	19
112	8
113	9

114	12
115	18
116	11
117	9
118	8
119	13
120	19
121	15
122	19
123	13
124	22
125	21
126	19
127	8
128	22
129	5
130	10
131	14
132	22
133	11
134	11
135	10
136	22
137	11
138	22
139	3
140	10
141	9
142	18
143	19
144	10
145	2
146	13
147	10
148	22
149	18
150	19
151	10
152	14
153	11
154	17
155	10
156	19
157	11

158	19
159	10
160	13
161	8
162	3
163	11
164	6
165	11
166	9
167	11
168	10
169	10
170	22
171	13
172	17
173	8
174	14
175	22
176	19
177	19
178	19
179	11
180	22
181	21
182	11
183	10
184	19
185	8
186	9
187	8
188	2
189	14
190	4
191	22
192	10
193	17
194	17
195	14
196	8
197	22
198	13
199	13
200	9

表 16 result2

9.2 问题二代码

9.2.1 代码

```
1) main.m
format='%d %*d %d %d %d %*d';
[x,r,g,b]=textread('image_1.txt',format,'headerlines',1,'delimiter',',()'); %#ok<DTX
TRD>
%x 存放图像中的颜色编号， r， g， b 存放 RGB 编码
color=[0,0,0;255,255,255;255,0,0;246,232,9;72,176,64;27,115,186;53,118,84;244,
181,208;255,145,0;177,125,85;92,59,144;11,222,222;
228,0,130;255,218,32;118,238,0;17,168,226;255,110,0;201,202,202;255,249,177;
179,226,242;249,225,214;186,149,195];
%color 存放马赛克瓷砖厂生产的瓷砖的现有颜色的 RGB 编码
result=Result(x,r,g,b,color);
%result 存放根据算法匹配的瓷砖编号和色差
id=fopen('result1.txt','wt');
fprintf(id,'序号， 瓷砖颜色编号\n');

%第二问
for i=1:size(x,1)
    fprintf(id,'%1d,%d\n',x(i),result(i));
end
fclose(id);
% 将结果输出到文件
% 选择优先增加的瓷砖颜色
dis=0;%re 存放用现有瓷砖还原图像各个颜色的色差和
for i=1:216
    dis=dis+result(i,2);
    %累加存放在 result 第二列的所有匹配的瓷砖与图像颜色的色差
end
Dist=dis/216;
%将色差和求平均
color_dist=select(x,r,g,b,Dist);
%select 函数返回一个 20*4 的矩阵，存放增加的颜色对应 RGB 编码和增加后
对图像还原的平均色差和（反应还原程度）
color_dist_reduc=ones(20,5);
for i=1:20
    color_dist_reduc(i,1)=color_dist(i,1);
    color_dist_reduc(i,2)=color_dist(i,2);
    color_dist_reduc(i,3)=color_dist(i,3);
    color_dist_reduc(i,4)=color_dist(i,4);
    color_dist_reduc(i,5)=dist_reduc(color_dist(i,4));
end
%将平均色差和经过正向化和归一化得到还原度 reduc
```

```

2) select.m
function[color_dist]=select(x,r,g,b,Dist)
color=[0,0,0;255,255,255;255,0,0;246,232,9;72,176,64;27,115,186;53,118,84;244,
181,208;255,145,0;177,125,85;92,59,144;11,222,222;
228,0,130;255,218,32;118,238,0;17,168,226;255,110,0;201,202,202;255,249,177;
179,226,242;249,225,214;186,149,195];
R=ones(441,1);
G=ones(441,1);
B=ones(441,1);
grad1=[0,43,86,129,172,215,255];%预设的红、蓝基色梯度
grad2=[0,32,64,96,128,160,192,224,255];%预设的绿基色梯度
all=1;%作为行标计数，循环自增
for i=1:7
    for j=1:9
        for l=1:7
            R(all,1)=grad1(1,i);
            G(all,1)=grad2(1,j);
            B(all,1)=grad1(1,l);
            all=all+1;
        end
    end
end
%生成 441 种颜色的 RGB 编码矩阵
Dist_add=Dist;
%增加颜色后的平均色差和，初始值为原始平均色差和
color_dist=ones(20,4);
for num=1:20
    dis_add=0;
    add=0;
    %add 存放使平均色差和最小的新增颜色的 RGB 编码下标
    for i=1:441
        color_add=[color;R(i),G(i),B(i)];
        %增加颜色后的瓷砖颜色 RGB 方阵
        result=Result(x,r,g,b,color_add);
        for j=1:216
            dis_add=dis_add+result(j,2);
        end
        if Dist_add>dis_add/216
            Dist_add=dis_add/216;
            add=i;
        end
        %筛选出使平均色差和最小的新增颜色选择
        dis_add=0;
        %置零进入下一轮筛选
    end
end

```



```

end
color_add=[color;R(add),G(add),B(add)];
color=color_add;
%将选择完成后的颜色放入 color 中，作为下一轮循环的已有颜色
color_dist(num,1)=R(add);
color_dist(num,2)=G(add);
color_dist(num,3)=B(add);
color_dist(num,4)=Dist_add;
fprintf('%d %d %d\n',[R(add),G(add),B(add)]);
%在屏幕上显示筛选出的颜色选择最优解
end
end
3) dist_reduc.m(完成平均色差与还原度之间的转换)
function[reduc]=dist_reduc(dist)
reduc=(0.8123-dist)/0.8123;
%将平均色差转化为还原度
end

```