2021.11.28周报

纪新龙

本周计划任务

调试丁守鸿两个算法的代码，重点是提升效果。

具体完成情况

完成

下周计划任务

转入阅读论文，至少读三篇相关的论文。

具体完成情况

- [x] 发现了上周报告里金字塔聚类效果不理想的原因，修改后达到效果理想
- [x] 算法2的其他能量项都改成log后的能量项
- [x] 重新规划算法2能量项的参数，提升了效果

金字塔聚类的改进

Algorithm 1 Irregular pyramid structure building

Inputs: image I, the level number of the pyramid T
$l = 0$
Compute W^l and L^l according to Eq.(6)
Compute pixel feature $f^l_{i.}, i = 1, 2, \cdots, n$
while $l < T$ **do**
    Select representative nodes B
    Compute interpolation matrix C^l
    Accumulate region properties $f^{l+1}_{.s}$ using Eq.(9)
    $Q^{l+1}_{mn} = \exp(-(\alpha_p D^p_{mn} + \alpha_q D^q_{mn} + \alpha_c D^c_{mn}))$
    Compute the coarse couplings W^{l+1} using Eq.(8)
    Compute the Laplacian matrix of W^{l+1} L^{l+1}
    $l = l + 1$
end while
//Incorporate high-level information

15: For any $Q_{mn}^T > t_h$, set $w_{mn}^T = 1$

影响金字塔聚类因素：

1. 代表点集B的形成方式。

总的选择方式是从旧的B里边按照**volumes**给所有点降序排序，第一个点无条件放入新的B，然后从第二个点开始计算该点和新的B中的所有点的边的最大权值maxW、该点和旧的B中所有点的权值之和allW。如果$maxW < allW * 0.15$，就说明当前这个点和新选的B中的点没什么关联，也就是说当前点是新的代表点集B中所缺少的、没法表示的样本。该点可以加入新的B。当加入B的点数量达到一半时，结束。

这个过程主要是*volumes*是什么论文没有说清楚，我把它理解为点与其他点的权值之和，达到了较好的效果。我在测试时尝试将降序排列改成升序排列，结果导致聚类的结果点都聚集在图像的边缘，原因是算法中的权值矩阵W本身就是非常稀疏的，经过一遍筛选会导致新的W成为全0矩阵。这样的话就会使旧的B中少数几十个点是有非零*volumes*的，而其他的都是0*volumes*。如果升序排序就会使得优先选取这些*volumes*为0点到B中，而边缘的点*volumes*一般都为0，这就导致聚类的结果成了一堆堆叠在一起的边缘点。所以相比之下还是按照降序，有限筛选有正权值的点进入B比较好。

上述的**0.15**和每次选取**一半**的点作为新的B，这些是作者明确的做法。但我在尝试改进的效果的时候都尝试过调整并观察调整之后的影响。我在测试的时候发现作者选取的这两个参数是比较恰当的。

2. 特征f的初始计算

特征主要是位置p、亮度q、色度s。位置就是归一化之化的x、y。主要是亮度和色度，没有明确说怎么求、要不要归一化、各个特征在初始和计算$Q = exp(-D_p - D_q - D_s)$时的权重。上周周末，我对特征f的上述四个方面做了测试，测试方法是控制变量，其他参数不变的情况下改变其中一个部分观察影响，**最后发现特征f的设定没有对聚类的结果产生绝对性的影响。**

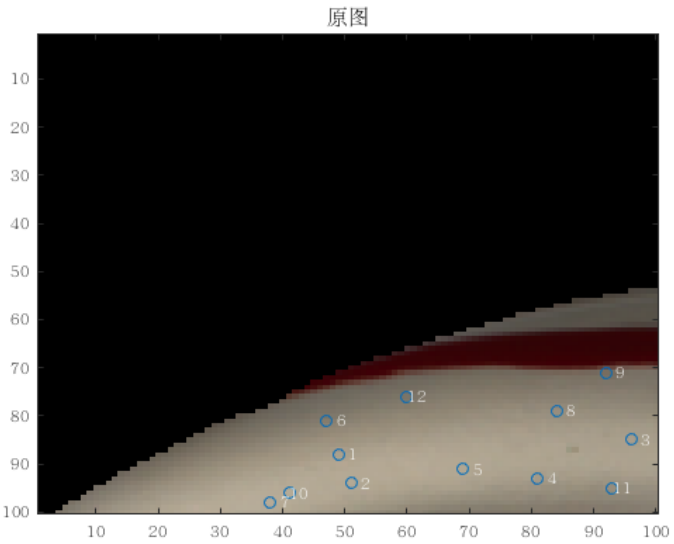
3. 初始W的计算

上周对上述两个影响因素的调整没能得出理想的金字塔聚类效果。本周初我对初始W的计算进行观察。作者是设定了色度阈值，计算相邻点之间的色度差距，如果差异小于一个阈值就表示两点相似。本周我尝试着在构建金字塔之前对图像的三通道进行平均，将其变成三通道值相等的灰图像。又由于色度是归一化的色度，这就导致所有像素点的色度之差为0，即不存在色度上的偏差，所有像素点之间的相似度都为1。意外发现，这样聚类的效果很好，每个区域最后都有一个代表点。

把色度的差异都消除，聚类结果却很好，原因是金字塔构建过程会综合考虑两点之间的权值w_{ij}和两点之间的特征相似度Q_{ij}来算出新的权值。色度的差异都消除，金字塔的底层所有w_{ij}都变成了1，但是底层的初始特征f仍然起到了区分不同区域的作用。虽然上一部分证明了特征f的初始计算不是绝对影响因素，但是当把色度的差异都消除，*初始W的计算*这个决定性的影响因素失效之后，初始特征f由次要变为主要，发挥了聚类的作用。

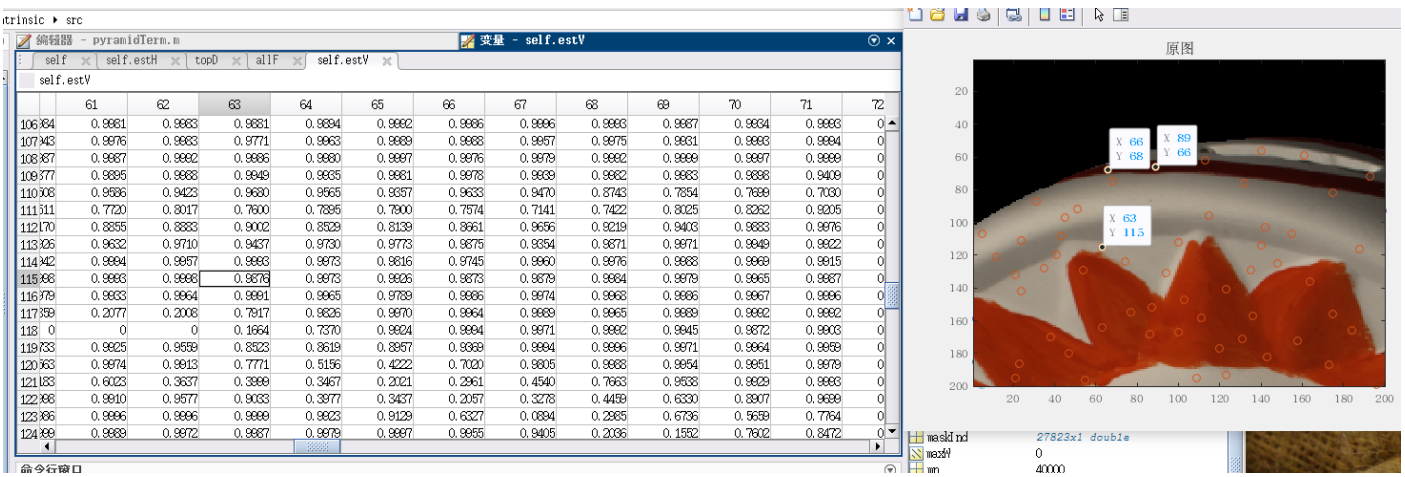
但是金字塔聚类的原意是需要初始W的正确计算的。既然把色度取消后效果很好，那就说明色

度的计算是导致聚类不好的原因。我之前计算色度的方法是直接继承自第一个算法的主要参考文献*Recovering Intrinsic Images with a Global Sparsity Prior on Reflectance*，它是把三通道取log之后再计算色度和亮度的。不同之处是，它的色度只用来和阈值进行比较，判断是否满足color retinex的色彩相近要求，而真正计算权值时使用的是log之后的亮度。而丁守鸿的第二个算法在聚类时直接用色度算出权值。再加上我之前的聚类结果中，最后代表点总时落在白色区域。换了一幅红色区域和白色区域差不多大的图，最后结果仍然聚集在白色区域。

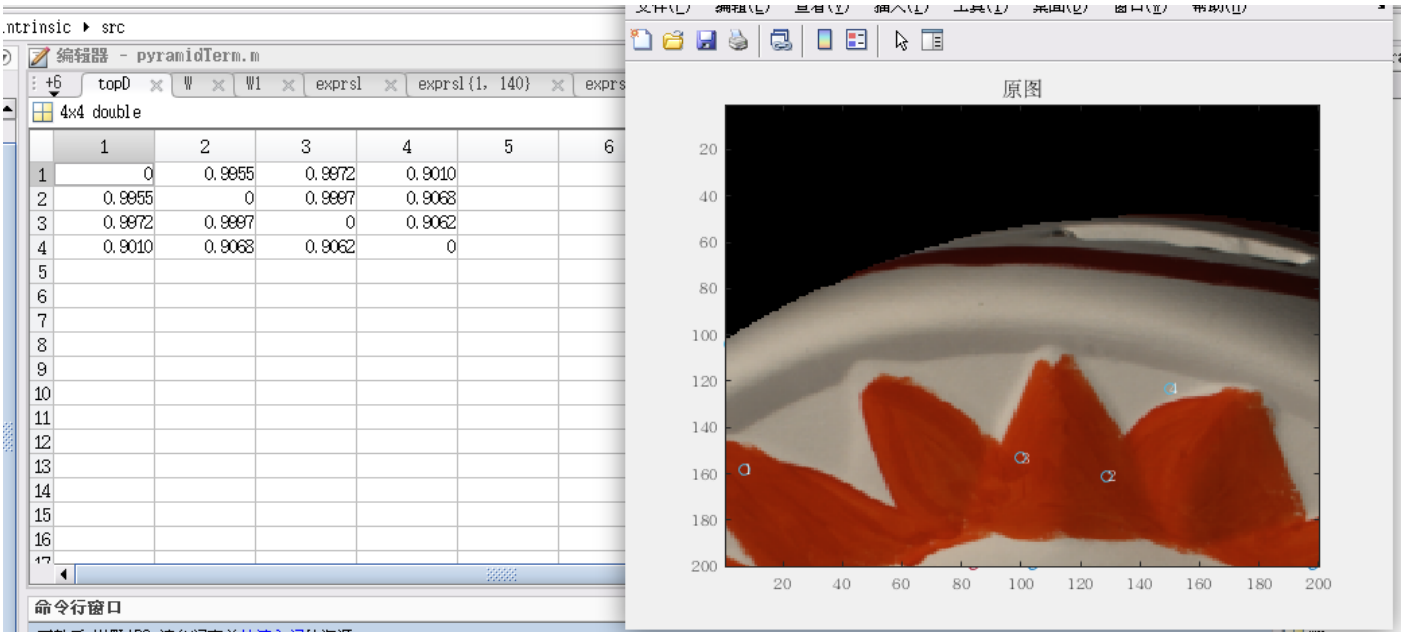


** 思考之后，我认为是盲目沿用了log之后的三通道计算色度，导致我的色度空间不均匀：假如一块区域的三通道都接近1，那么同一片白色区域三通道log之后计算出的色度不会差别很大；但如果是某一种色彩很明显的颜色，比如红色、蓝色、绿色这种，在RGB空间是有某一维度是接近0的，那么先对三通道log再求色度就会产生非常大的“色度”偏差。一片区域看起来都是红色，也理应聚类到一块，但是有些点的G和B通道稍微和其他点有点误差，就会导致log之后这种偏差无限放大，这种方法计算出来的“色度”也就无法体现真正的色度关系了。**

我取消了log，直接根据原图的三通道求色度，聚类结果就变成理想了。下面这幅图是修改色度计算方法之后，金子塔某一层的代表点以及金字塔底层使用的初始权值（V代表是根据垂直方向色度差计算出来的相似度权值）。我在原图中标记了各个代表点的坐标，该坐标和左边的矩阵的横纵坐标一致。可以清楚的看到原图的(63,115)点下方就是红色的边界，而左边矩阵中的(63,115)号数据的下方数据出现明显的几个值非常小的数据，对应了原图中的边界。



下图是金字塔顶层的四个代表点，以及点与点之间的特征相似度。原图中点的序号对应左方矩阵的序号。我们可以看到，前三号点落在红色区域，在矩阵中他们之间的相似度确实比较大，大于0.95，而第四个点落在白色区域，在矩阵中它和前三个点的相似度也确实都小于0.95。

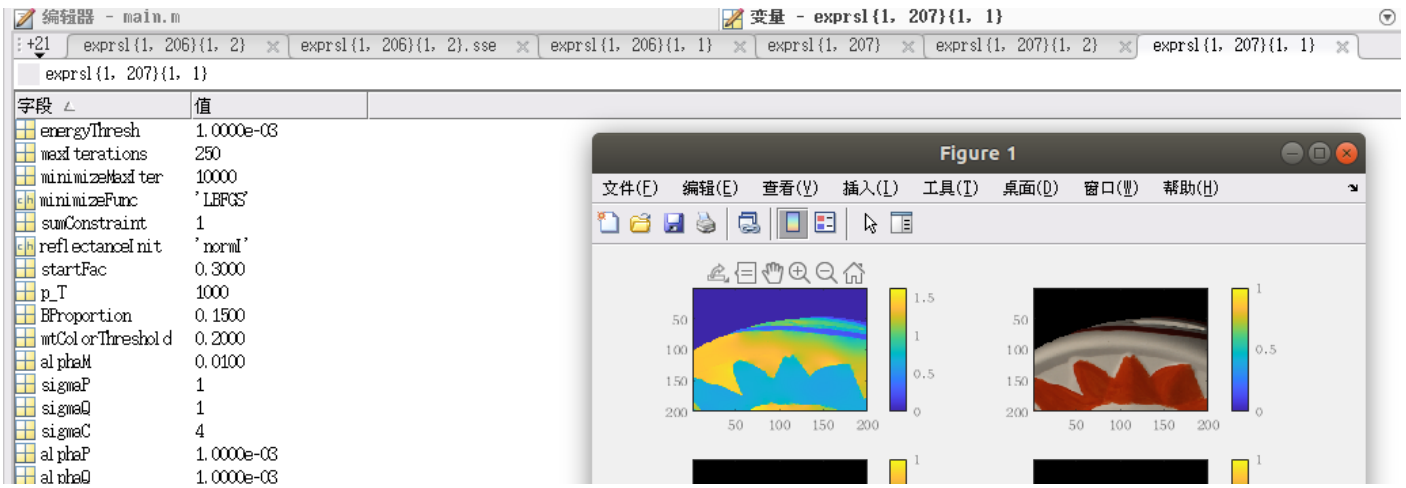


题外话，为什么聚类时候把“色度”当成唯一标准，而不是像彩色retinex计算时综合考虑亮度和色度。我觉得这是因为有一个“色彩全局稀疏性”的假设在，以及单色光照的假设。有了两个假设，下面便是成立的： - 能够将距离较远但是颜色相同的区域归为一类： 这就意味着，如果图像上两块区域归一化后的色度相同，但是亮度不同，比如 亮红色归一化之前(0.9,0,0) 和暗红色归一化之前(0.4,0,0),归一化之后都是(1,0,0)那么我们一般就认为这两块区域其实是同一种颜色，他们归一化之前的差异正好归咎于光照的明暗。 - 能够将不同距离相近，灰度相近，看起来很相似但是应该属于两类的区域分割： 灰度更多是被光照影响的，色度更多影响反射率。假如两个点 **(0.5,0,5,0.5)和(0.9,0.5,0.1)**因为光照、形状的原因看起来很相似，灰度计算（三通道求平均）后也是一样，这个时候就比较有迷惑性，如果依靠灰度来聚类就会把它们归纳成同类，但他们从色度上看明显是两种颜色。这就是不能用灰度的原因。

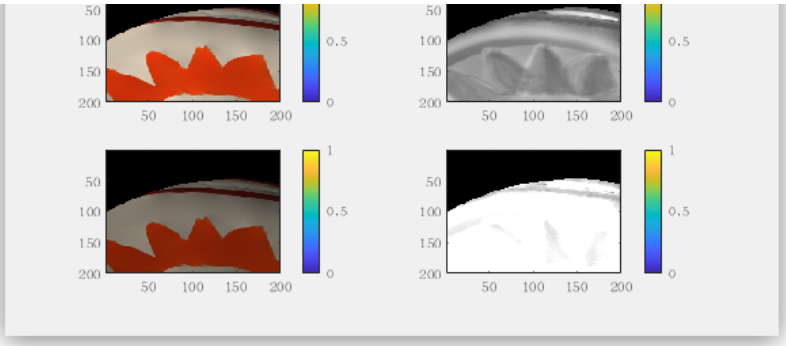
算法2的效果：

同样一种算法面对不同的不同图像，计算出来的LMSE是不同的。

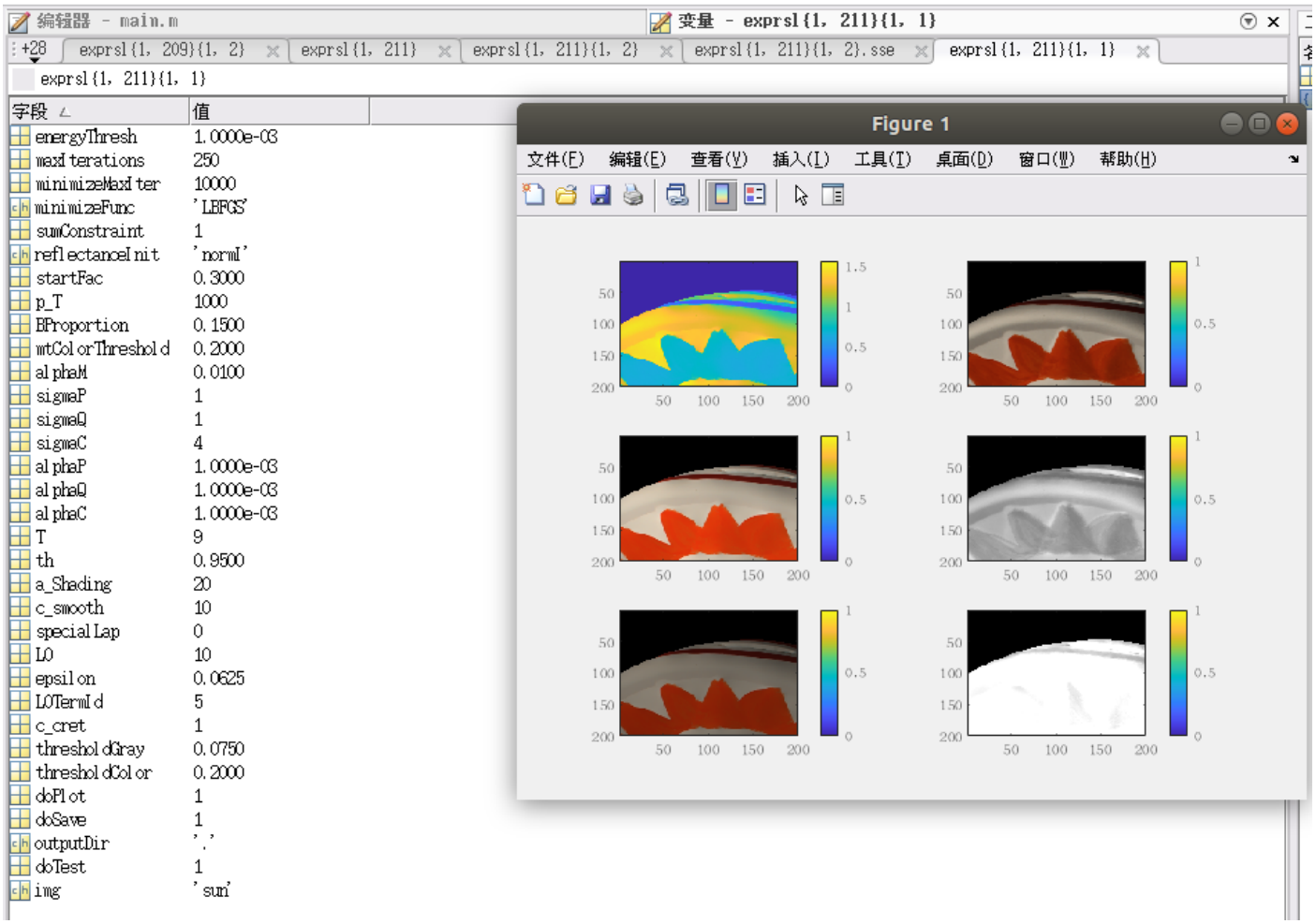
不断修改算法中的参数，得到在下图上的最小LMSE=0.0042及其参数选择、效果：



alphaC	1.0000e-03
T	9
th	0.9500
a_Shading	20
c_smooth	200
specialLap	1
L0	10
epsilon	0.0625
LOTermId	5
c_cret	1
thresholdGray	0.0750
thresholdColor	0.2000
doPlot	1
doSave	1
outputDir	'.'
doTest	1
img	'sun'

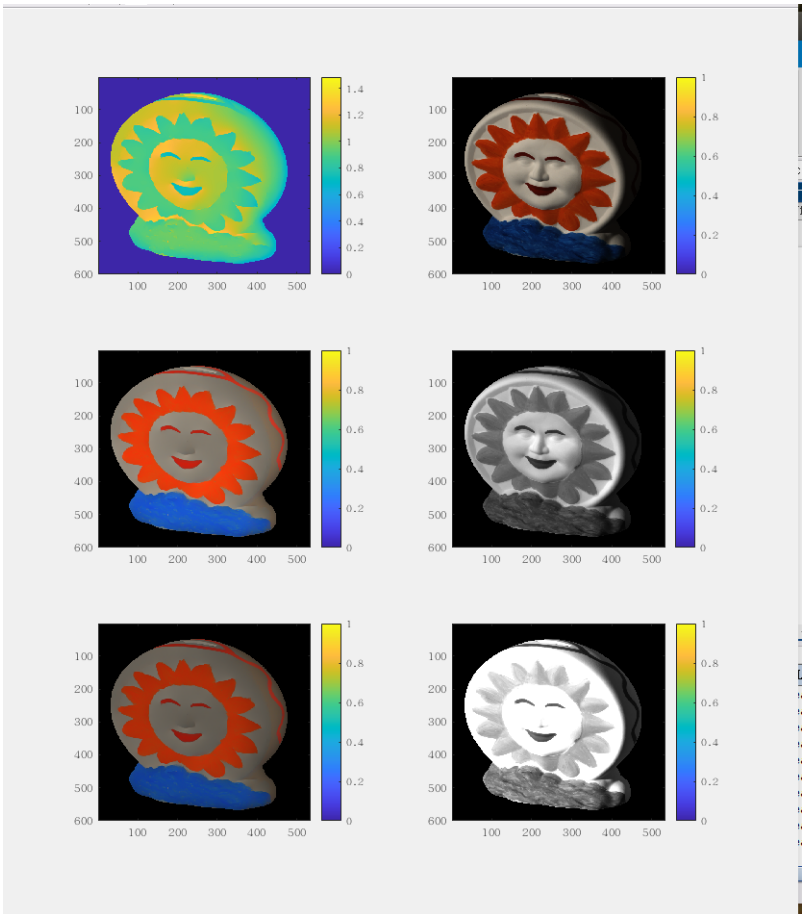


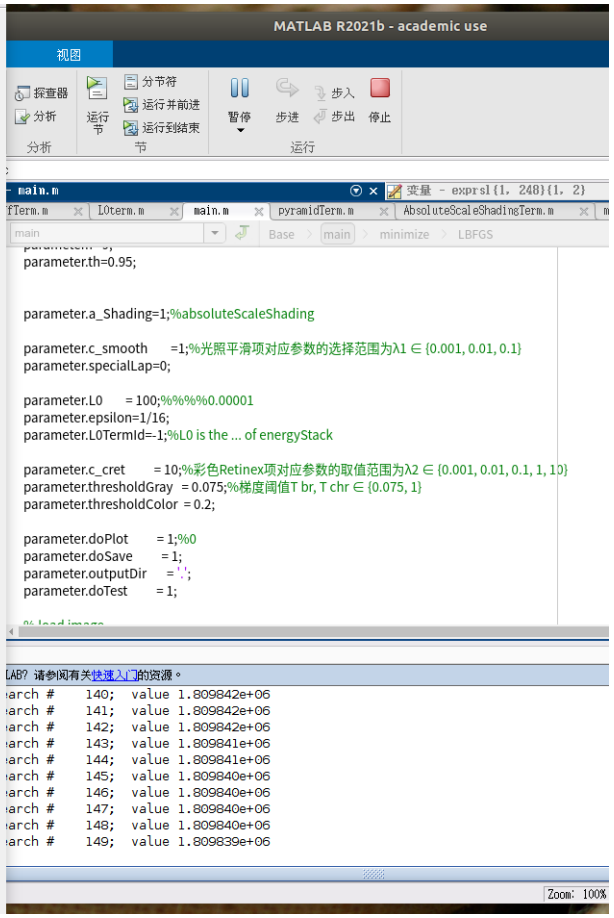
参数里的specialLap是我自己实现的时候加的。表示光照平滑项采用特殊的拉普拉斯算子。丁守鸿使用的是普通的拉普拉斯算子，即计算所有像素和其附近点的差。而我认为在光照图上，微小色度变化是允许的，这样有助于保证光照图上的缓慢的光细节变化。在反射率图上被允许的大色度变化（象征着边界）反而是在光照图上应当被克制的。所以我设计了一个和彩色retinex项中相反的拉普拉斯算子用于光照平滑，即反射率层上色度差异小于阈值时，我们设置其关系值为100；在光照中反着来，色度大于阈值时，像素点关系设置为100。这样以期待达到将大的色度变化显示在反射率图中，将小的色度变化体现在光照层中。灵感来自于之前读过的一篇分解视频中直接光照和间接光照的论文。如果使用丁守鸿的普通拉普拉斯算子，LMSE最低可以达到0.0050这样。效果图和参数如下：（参数只要是specialLap和c_smooth变了）



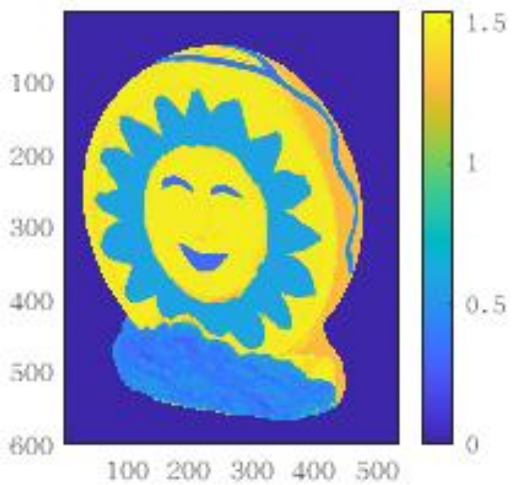
六个小图，第二行是正规化后的反射率层和光照层分解结果。第一行左侧是反射率的值呈现出的图，右边是原图。第三行是未经正规化的反射率和光照结果。虽然使用绝对光照约束项了，但还是

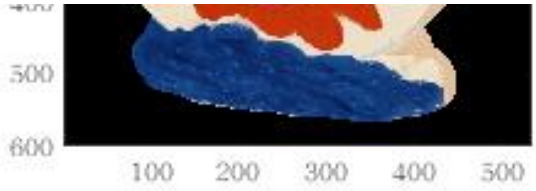
没法约束光照中最亮的像素点值为1。算法2总的效果：

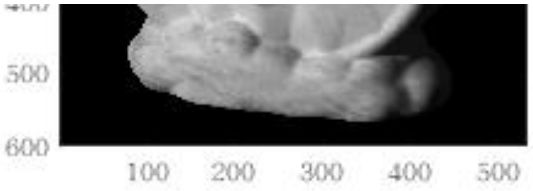




算法2分解出的光照中残留属于反射率层的色彩，但是反射率层比较符合色彩稀疏性假设。 #### 算法1的效果







算法1的结果很接近真实的反射率和光照情况了，瑕疵是反射率层的边界处残留有棱角的影响。