阴影检测与去除 实验报告

14307130078 张博洋

一、完成情况

实现了“The Shadow Meets the Mask: Pyramid-Based Shadow Removal”一文中方法的**简化版**。

原文中的方法只需要用户手动选择一个阴影内部的点；而简化版的方法还需要手动调节两个参数。

经实验表明，该方法能够去除简单场景（如单一场景且阴影不跨越不同材质）下的阴影；对于复杂的场景该算法无能为力。

二、算法概述

1. 阴影检测

阴影检测分为四个步骤：

（1）由用户选择的点来扩展出一个相似颜色的区域（“阴影种子”）

此步骤类似于 PhotoShop 中的“魔法棒工具”。在用户选择一个点后，程序会计算出连续且颜色十分相近的部分作为接下来操作的“种子”。

（2）由阴影种子计算出与阴影在同一平面上的所有像素（“平面”）

原文假定，阴影区域所在平面的每个像素点的 RGB 向量，不论在阴影内还是阴影外，都是共线的。因此原文定义两个颜色 A、B 之间的距离为 1-abs(cos<A,B>)。（然而，原文这样假定不一定是正确的。在很多情况下，同一平面阴影内和阴影外的颜色并不是共线的。这样就会导致求出的平面发生严重的错误。）

用“种子”中像素颜色的中位数作为标准，计算图片中每一个像素的距离（与阴影平面的相似度），设置一定的阈值（用户可调）后即可得到与阴影在同一平面上的像素。

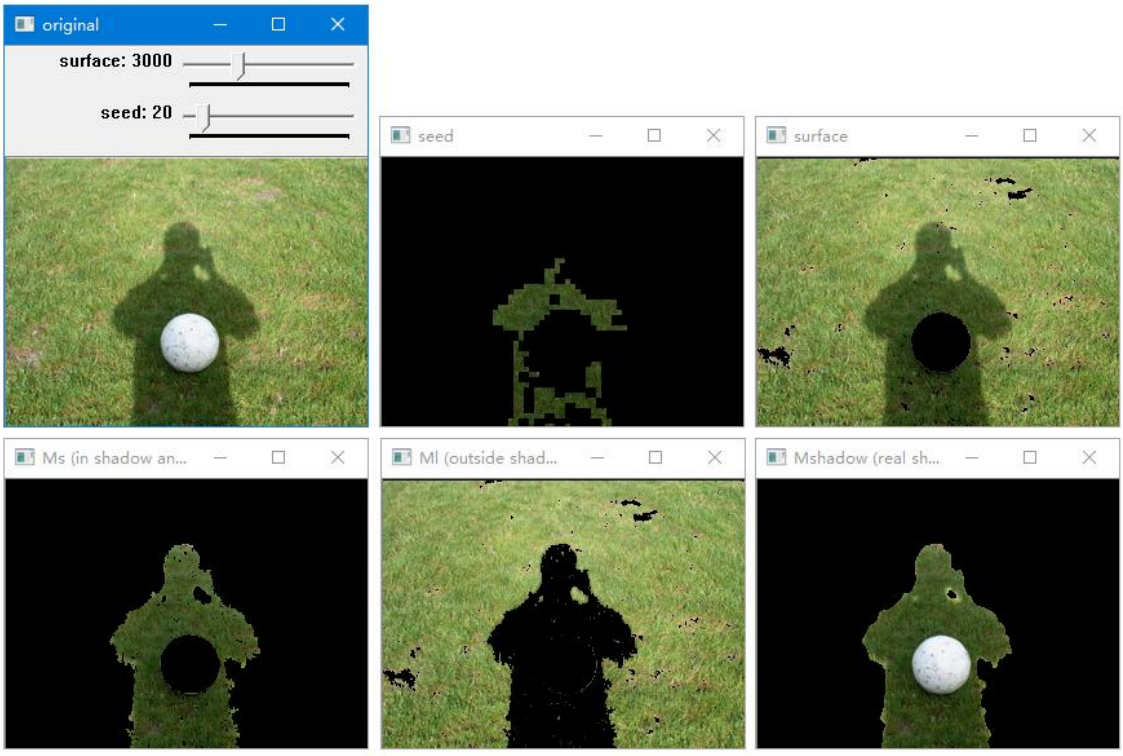
在确定下与阴影在同一平面上的像素之后，后续的所有操作均在此平面所含像素内执行。

（3）由“阴影种子”计算出一定在阴影内部的像素（“Ms”）和一定不在阴影内部的像素（“Ml”）

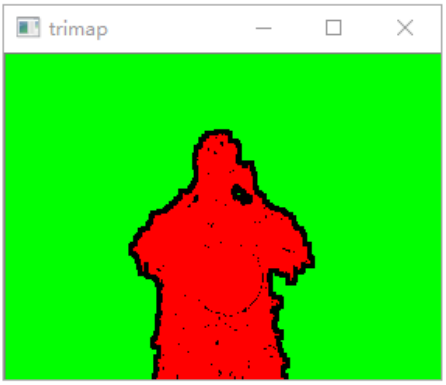
将 Ms 初值设为“阴影种子”，Ml 初值设为“平面减去种子”。每次迭代都将 Ml 中的一部分像素移入 Ms 中。在此过程中 Ms 颜色的标准差会慢慢增大，而 Ml 的颜色的标准差则会慢慢减小。重复此过程直到 Ms 的标准差超过 Ml 的标准差的前一刻。此即求出了 Ms 和 Ml。（然而在某些情况下，标准差并不像预期的那样变化。这样就会导致求出的 Ms 和 Ml 发生严重的错误。）

（4）通过“平面”、“Ms”、“Ml”计算阴影区域“Mshadow”

由于阴影中可能含有不属于阴影平面的部分（例如草地上的足球），因此取 Ms 和被 Ms 所完全包裹的像素作为作为“一定是阴影区域的部分”（记为 A）、“将 A 扩张一定范围后取反”作为“一定不是阴影区域的部分”、将剩余部分作为“未知部分”形成一个 trimap。再通过这个 trimap 计算出最终的阴影区域“Mshadow”。原文中使用了一种专门的 alpha matting 算法，而我使用的算法是：未知区域中的点，距哪个区域的距离较近就算作哪个区域的一部分。



图：算法各阶段所产生的图像



图：trimap

红色区域是“一定是阴影区域的部分”；

绿色区域是“一定不是阴影区域的部分”；

黑色区域是“未知部分”。

我实现的方法与原文中方法不同之处在于：

（1）计算种子区域时所采用的阈值需要用户手动指定；

（2）区域扩张算法我使用的是只考虑 4 邻居的简单 BFS，效果较差；

（3）计算同一表面的所采用的阈值需要用户手动指定；

（4）trimap 处理采用了一种简单的算法，效果较差。

2. 阴影去除

（1）简单阴影去除

原文中提出了一种光照还原模型：

$$I^{lit} = alpha + gamma * I^{shadow}$$

其中 I^{shadow} 表示含有阴影的图像，I^{lit} 表示去除阴影后的图像。

参数可以由下面的公式求得：

$$alpha = Mean(M_l) - Mean(M_s)$$

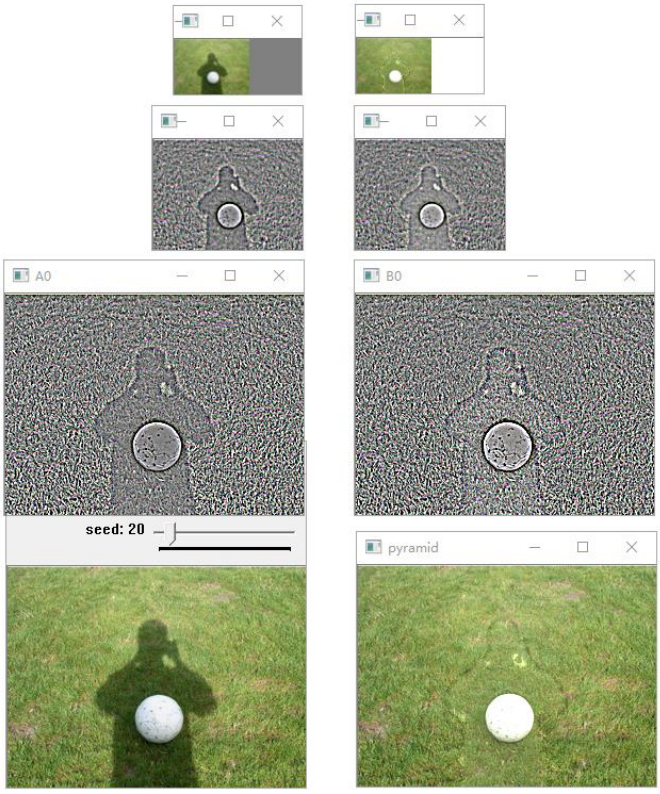
$$gamma = StdDev(M_l) / StdDev(M_s)$$

因此只要按照文章中的算法实现即可。

（2）基于图像金字塔的阴影去除

文中分析了简单方法的缺点（例如对比度不一致等），并提出了一种基于图像金字塔的去除阴影的方法，可以克服这些缺点。

具体做法是，先生成原图像的拉普拉斯金字塔，然后对于金字塔的每一级都应用简单阴影去除方法，最后再把整个金字塔还原成一张图片。



图：拉普拉斯金字塔

（左侧为原金字塔；右侧为经过普通方法处理后的金字塔）

经实验表明，在大部分情况下，采用金字塔方式得到的结果要优于普通方法的到的结果。然而也有一小部分情况下，普通方法更好一些。

（3）非均匀阴影的处理、边缘的处理

原文还提到了对于非均匀阴影的处理方法（分为多带分别处理）和阴影边缘的处理方法（用周围的点来填补）。由于这两个步骤比较繁琐，我没有实现它们。

三、性能分析

由于算法需要人工介入，因此从数据集中选出一些**场景简单**的图片，然后进行人工测试。根据结果分为效果较好、效果较差、失败三种。

下文图片从左到右排列为：trimap，金字塔法去阴影结果，原图，平面。

1. 效果较好的案例

效果较好的案例一般具有特点：

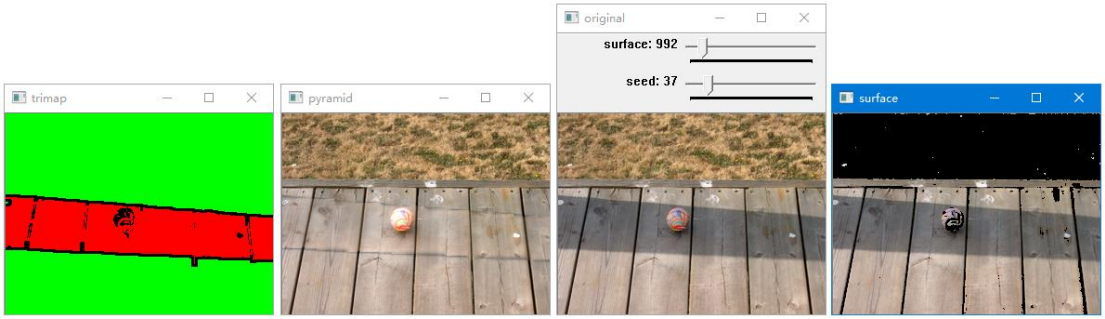
（1）图片场景单一；

（2）阴影颜色均一；

（3）阴影颜色与非阴影部分相差较大；

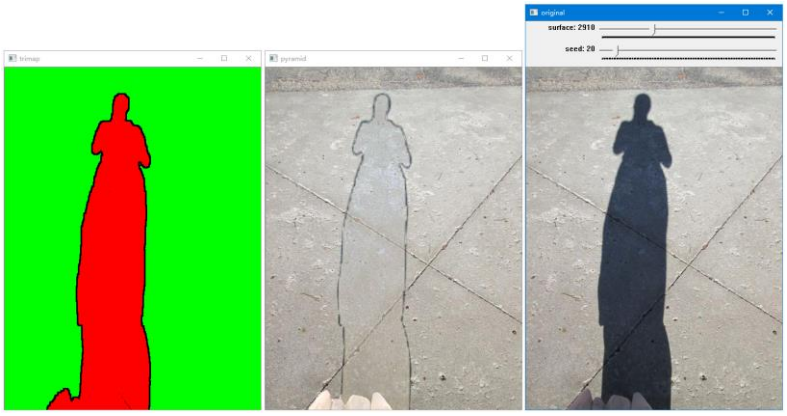
（4）光源颜色接近白色。

（1）ball.png



（2）lssd125.jpg

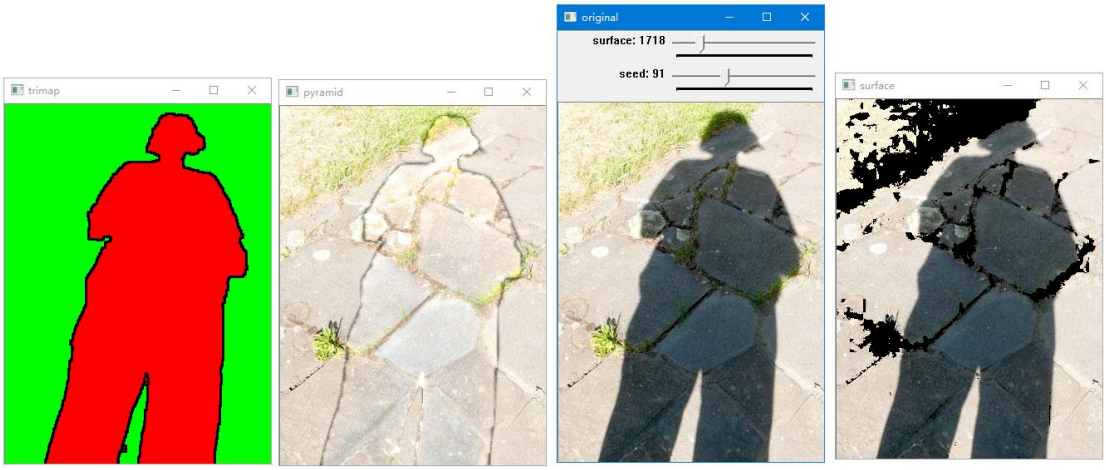
（3）lssd3548.jpg



2. 效果较差的案例

（1）lssd566.jpg

该图片中阴影颜色不均一，靠近头部的阴影相对较浅，靠近脚部的阴影相对较深，因此去除阴影后出现头部较亮，脚部较深得情况。



（2）lssd667.jpg

该图片中阴影跨越了两种不同的材质，算法的颜色模型无法处理这种情况，两种材质的颜色都向平均色发生了偏移，出现了偏色现象。

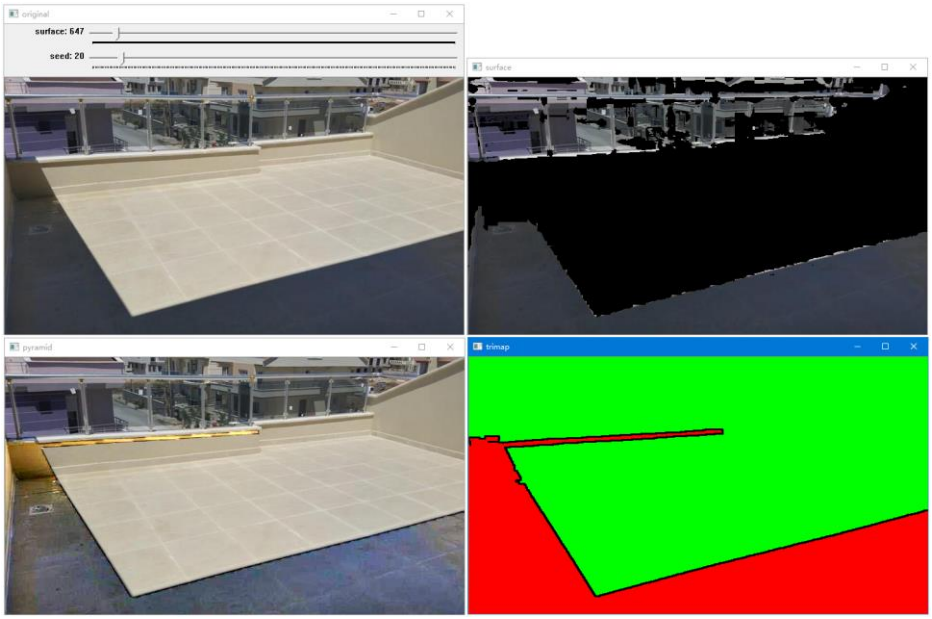
（3）lssd9.jpg

在阴影的头部，阴影内外的颜色差距不大，由于使用了简单的基于 4 邻居的 BFS 区域扩展算法，导致阴影头部左侧的一部分区域也被当做阴影了。

3. 失败的案例

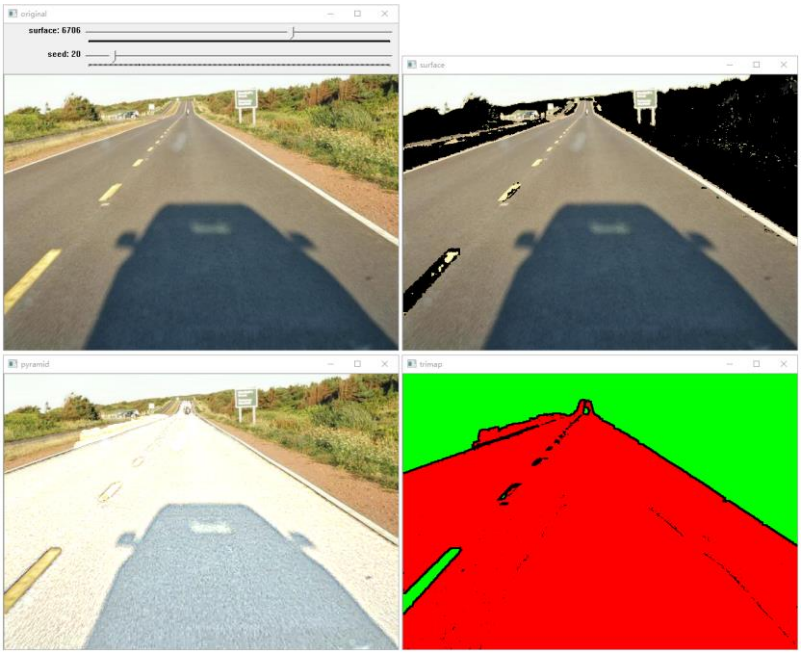
(1) lssd281.jpg

由于同一平面的阴影部分和非阴影部分的 RGB 向量不共线导致平面检测算法失败，因此后续的颜色模型完全失效，从而颜色出现巨大偏差。



(2) lssd15.jpg

由于 RGB 向量不共线导致在计算 Ms 和 Ml 的过程中，Ml 方差很大，Ms 需要添加许多像素才能弥补，于是整个路面都被当做阴影进行了处理。



（3）lssd102.jpg

由于影子较浅，在计算 Ms 和 Ml 的过程中，方差没有按照预期变化，导致整个图片被识别成阴影，后续无法进行处理。

四、总结

该算法的亮点在于采用拉普拉斯金字塔进行多级阴影去除，能够增强阴影区域内纹理的处理效果。在实践中，大部分情况下使用金字塔的效果要比普通方法强。

该算法的缺陷在于仅能对简单场景的图片进行阴影去除，而且该算法严重依赖于“同一平面无论是否在阴影内 RGB 向量一定共线”，“在 Ml 中像素移入 Ms 过程中 Ms 方差增大且 Ml 方差减小”，一旦图片不满足这两个条件，阴影去除的结果可能会很差甚至根本无法得到结果。另外，该算法在某些情况下也不能很好完成任务：例如阴影跨过多个材质等。此外，简化版的算法不仅受到上述限制，而且缺少不均匀阴影的处理和边界的处理。