数据可视化第一次作业

凌华 17210180082

1 Design a similarity metric or distance metric for student information data formatted as follows:

Data: <V1, V2, V3, V4>, where

V1:  height of the student (unit: m)

V2:  Weight of the student (unit: kg)

V3:  Place of Birth (province/city)

V4:  Grade (grade 1-5)

X1 = (V1- V1\_mean)/V1\_std

X2 = (V2- V2\_mean)/V2\_std

X3 = 1 if 两人V3相同 else 0

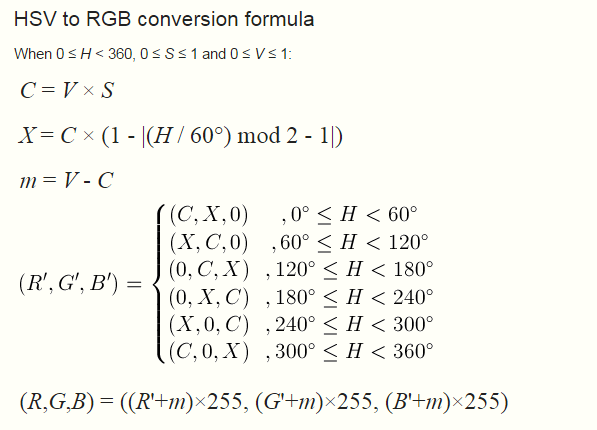
X4 = V4

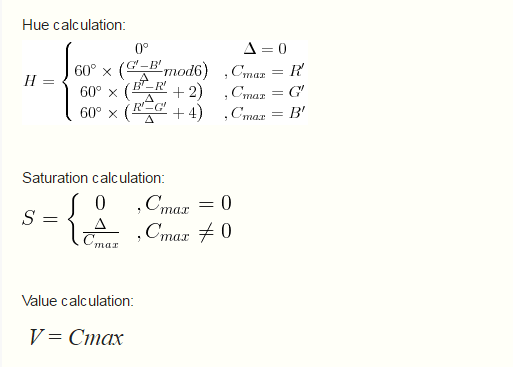
Distance of (student i and student j) =

2 Textbook Exercise Page39： 2.1

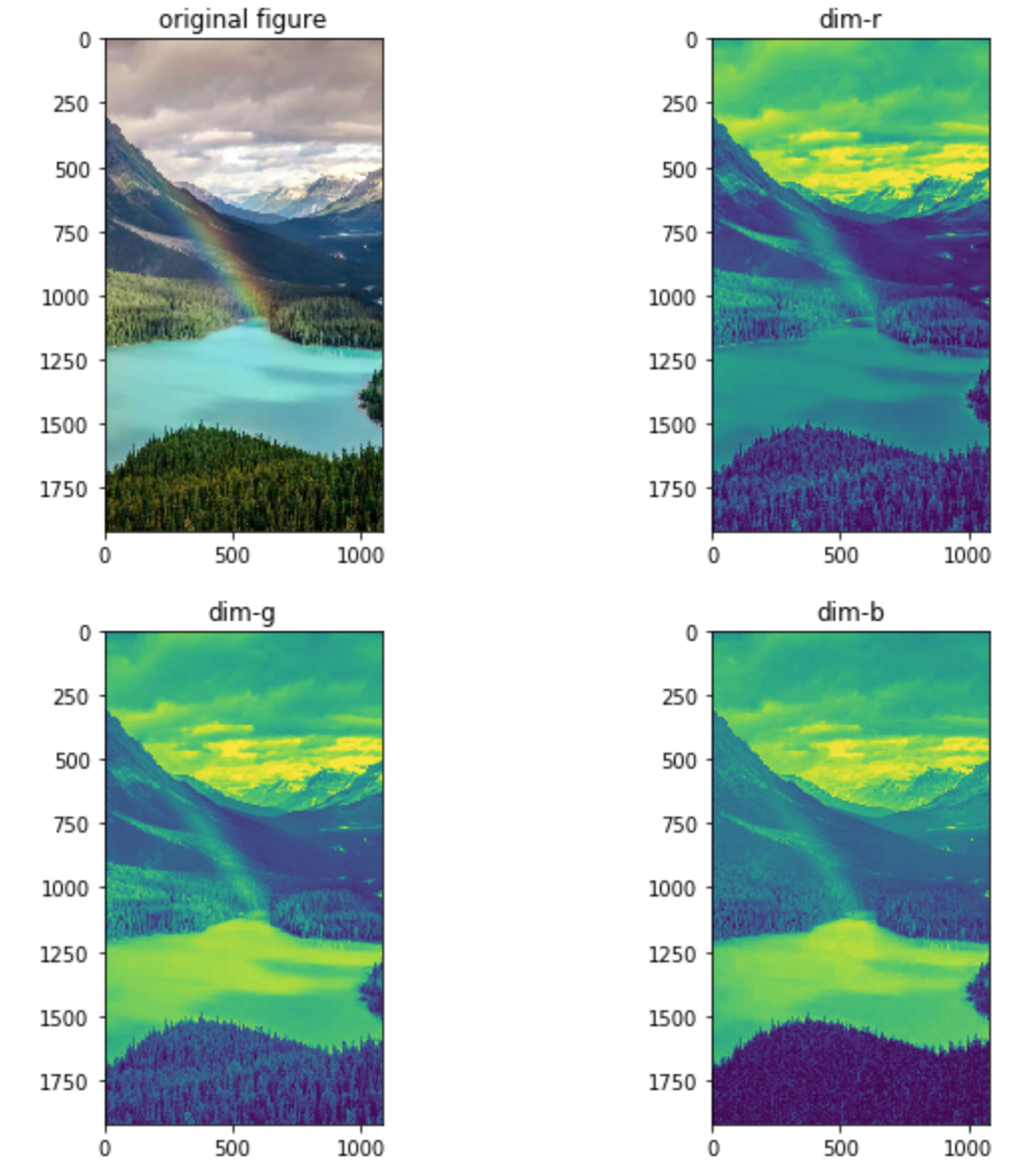
在进行颜色的计算时通常需要在不同的颜色空间中进行数值转换，请用编程语言（建议C/C++或Python）实现2个常用的颜色空间的相互转换算法。请做好代码的注释。

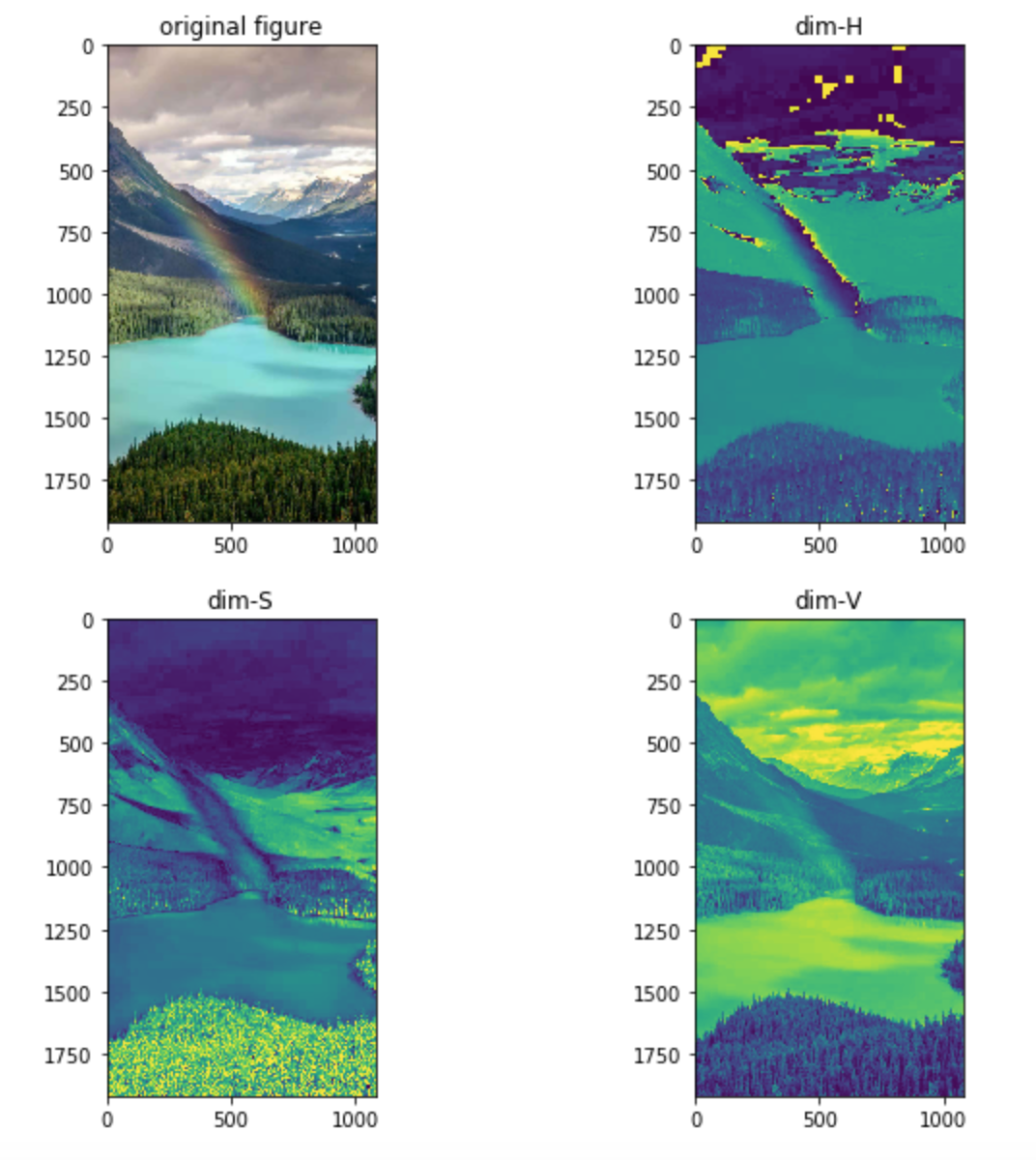
 RGB与HSV互换



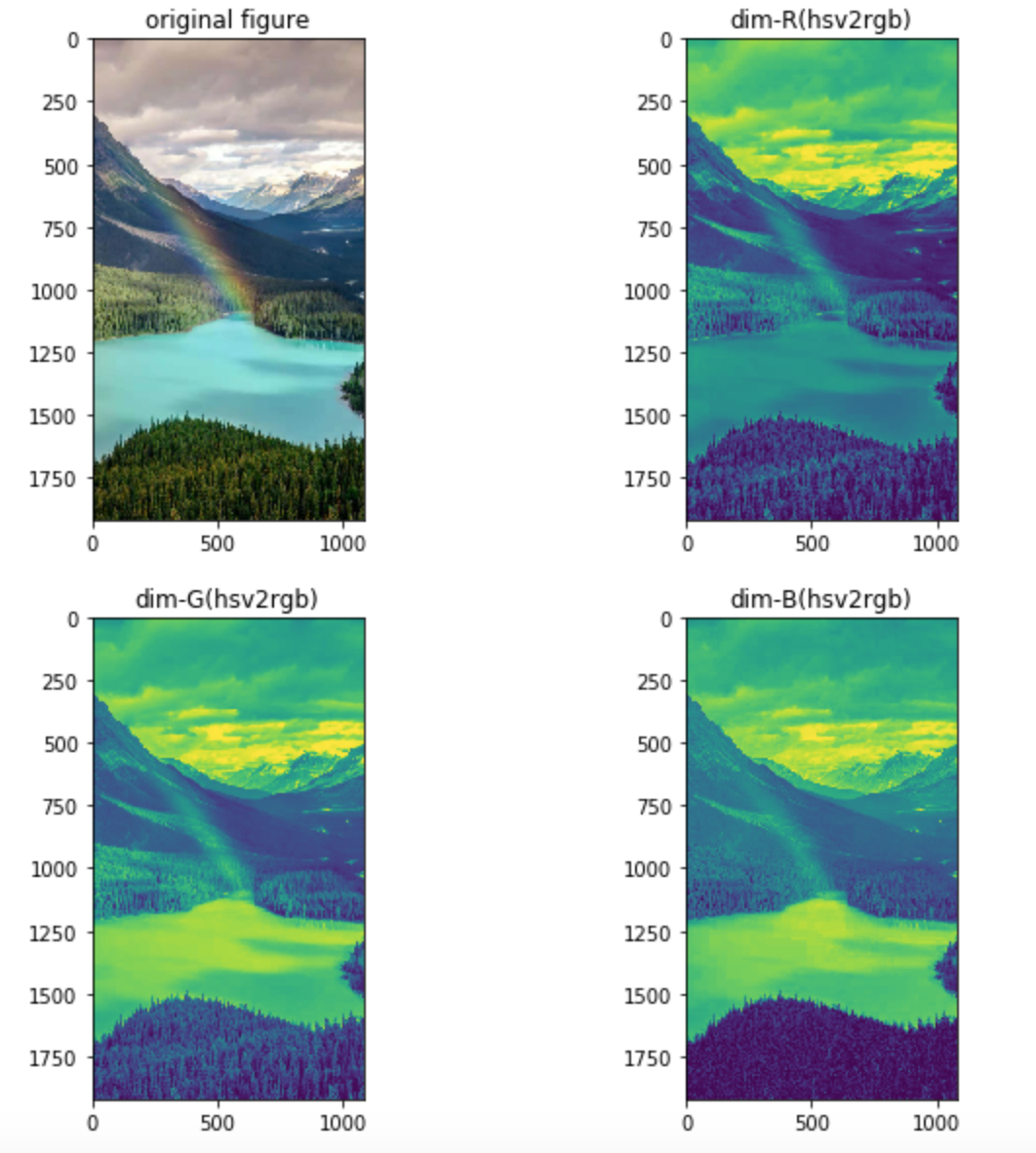


展示原图，以及R,G,B三个通道的图像。



展示RGB转化成HSV后原图和H,S,V三通道的图像。

展示HSV转化成RGB后原图和R,G,B三通道的图像。

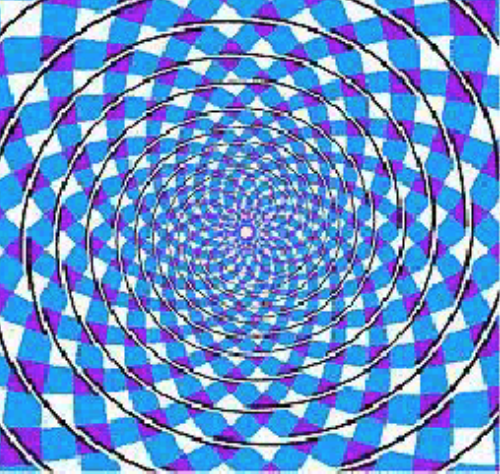


3 Textbook Exercise Page39： 2.2

（1）视觉感知的相对判断原则在网络上常被用于制作一些包含错觉的图片，请查找两幅这样的图片，并说明所理解的其背后用到的视觉感知的原理。或（2）寻找设计利用相对判断，增强视觉效果的例子。

 （1）

褐色直线是平行的。这种现象被称为左氏错觉，它是一种空间错觉。理论解释认为，当两个轮廓彼此接近时，他们在视网膜上的投影彼此接近。由于视网膜上的神经细胞兴奋产生视觉信息时存在侧抑制现象，由轮廓所刺激的视细胞的活动被改变，使得神经兴奋的中心发生了相对的位移，进而引起几何图形形状和方向的错觉。在上例中，本来是平行的褐色直线在视网膜上投影也是平行的，在加上了不同方向的直线后，他们共同产生的轮廓在视网膜上的投影由于侧抑制的影响而发生位移，因此产生了这种假平行的错觉现象。



弗雷泽螺旋：黑色的一圈圈的弧看起来是一个螺旋，其实他们是由一组同心圆构成。但是如果遮住图的上半部分，这种幻觉不复存在，这意味着直觉上的特性必然产生此种效应。

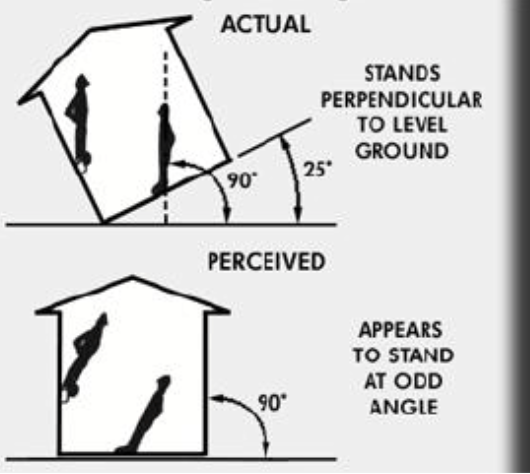
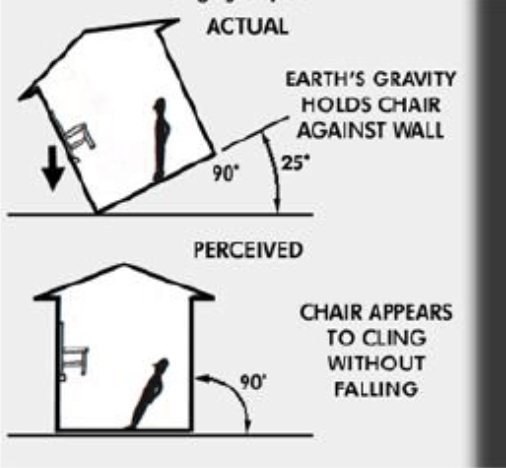
弗雷泽螺旋错觉是最复杂的盘旋绳索错觉，许多因素导致了这种视觉上的错觉。因此，即使这些同心圆本身的轨迹暴露了，背景上的每一个带有方向性的小单元格使之产生螺旋上升的知觉。这种错觉的形成是因为多变的背景。你会发现这住上半部分后错觉不是很明显了，知识因为背景改变了，但它确实还存在。这些带有方向性的小单元格分组聚合，使螺旋路径明显。表明放生在视网膜和大脑皮层细胞在简单图形的加工过程中的影响。这种螺旋效应可能由这些区域的方位敏感性细胞造成。例如：连续的视觉效果时V1视皮层上“相似”细胞之间的水平连接。成对细胞间交叉相联的模式并非完全固定不变的，随着环境的变化而稍微改变。细胞间相互影响，使视网膜上形成的简单的连续的线由于方向性单元格而倾斜，造成错觉。

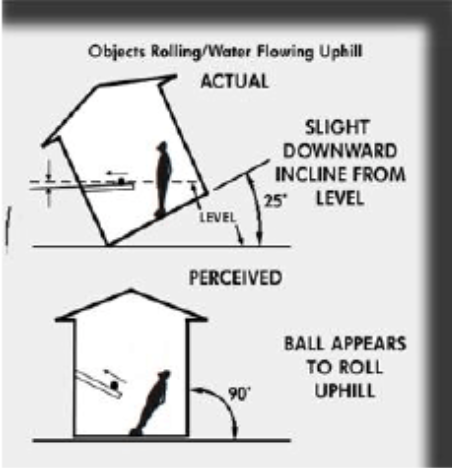
（2）

颠倒屋

当进入这个房屋时，你会注意到他有一种强烈的倾斜性。所有相对于真实水平线的参照物都从你的视野中移走，这样无论你在房屋里还是在房屋外一切都总是真实的。例如，把所有相对于真实水平线的明显对照物移走后房屋的周围看起来总是有一个木制的篱笆。

斜站在墙上的效果，你会看到房屋相对于真实水平米娜来说确实是倾斜的。任何一种情况下人们都是垂直于真实的水平面。他们没法达到真实的水平面，只好根据房屋中人为创造的水平面来判断他们周围的环境。这样就使得人们在参考框架上有了一种内部的变化，这种变化又使得人们看起来像真的一样斜站在墙上。



4 (PG optional) Design an algorithm which returns the n-th ranked number from great to small, from an array of un-sorted numbers, submit your code (in python or c/c++) and test data.

Method1: 采用局部淘汰法。选取前n个元素，并排序，记为序列s。然后一次扫描剩余的元素lst[n:]，与排好序的n个元素中最小的元素比，如果比这个最小的要大，那么把这个最小的元素删除，并把x利用插入排序的思想，如果k较小，则插入后对序列s快速排序，否则利用二分法将x插入到序列s中。依次循环，知道扫描了所有的元素。复杂度为O(len(lst)\*k)。

Method2: 用含有n个元素的最小堆完成。

Method3:基于快速排序算法或堆排序对所有数据进行排序，复杂度为len（lst）\*log(len(lst))

现随机生成1亿个服从标准正态分布的数据，提取top100, 比较三种算法的耗时:

|  |  |
| --- | --- |
| 算法 | 耗时 |
| Method1 二分法插入 | 7秒 |
| Method2 堆排序 | 20秒 |
| Method3 快速全排列 | 163秒 |