FreeSandal

樹莓派, 樹莓派之學習, 樹莓派之教育

GOPIGO 小汽車：格點圖像算術《色彩空間》灰階・丙

2017-06-27 | 懸鉤子 | 發表迴響

『眼睛』並非單純的『物理量』度量儀表，『感官知覺』雖有相當之『共性』，『差異』還是存在的。因此明白『完美的黑』 — 入眼光譜為零 — 與『理想的白』— 所有可見光輻射強度均等 — 之所以為重要『色覺參考』思過半矣！或可『讀』『解』公孫龍之『白馬非馬論』乎？

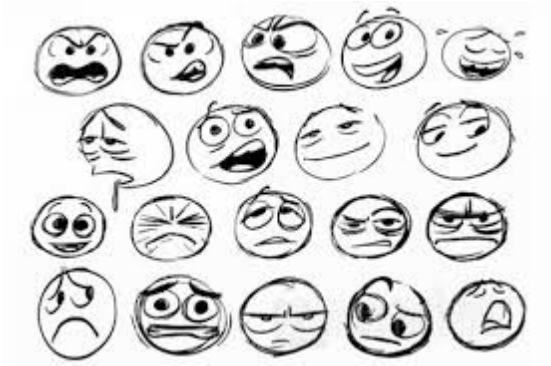
[/wc_column][/wc_row]

『**白**』色是各種叫**白色**的**東西**，所**共有**的**顏色**。有這種顏色嗎？然而『**白馬**』的白、『**白羊**』的白和『**白狐**』的白，雖然在**其類中**都是稱之為**白**者，**彼此類間**確又**有所不同**，當然也與『**共相**』**抽象**的『**白**』色有所不同。比方說把『白羊』之『白』用之於馬，『**那馬**』—— 羊白之馬 ——**或**叫做『**黃馬**』，為什麼呢？因為『白羊』之白**近之於**『**馬類**』之**黃**，**雖**然它**是**『**羊類**』之**白**。

許多**抽象**的**概念**，從『**具體**』的物來看，不過是『該物』**屬性之一**。說不定還不是『**物理**』上的屬性，就像『**光**』與『**色**』的不同，更不要說世界上尚有『**色弱**』之人，若是彼二人爭論『**物之顏色**』的**異同**，恐怕最好求之於物理『**量測儀器**』較好。更不要說它還可以是『**模糊**』的，舉例說『**禿頭**』一詞：設想一個『**濃髮**』之人『**掉一根**』頭髮，不會**就變成**了『**禿頭**』的吧！那一根一根的掉呢？掉到『**哪一根**』才**讓他成為**『**禿頭**』的人了？果真能說是那一根的『**錯**』？冬雪陽春，不如踏雪尋梅吧！！

馬『**形**』是看起來『**象**』馬的東西，『**馬**』字的構創之來歷。但是『形』能回答『**馬是什麼嗎？**』，也許只因人們心中有著**馬**的『概念』，它不是**牛**的『概念』，也不同於**羊**的『**概念**』，由於馬牛羊外『**形**』**不同**，正**足**以『**區別**』這些不同的動物。想當日創生『馬』『牛』字之時，卻遇著了『**牛頭馬面**』來訪，這兩個字會『寫的』和今天不一樣嗎？

過去有一個『**瞎子摸象**』的故事，因為『**見不著**』象，於是**各說各話**，所以不能說用『形』來取象沒有大用。那就說『**同一張**』『**臉**』又怎麼會因為『**幾條線**』的不同，就『**變臉**』不象『**那個人**』了呢？



夏日炎炎，何不行到水窮處，坐看雲起時，見著『形如白

馬』之雲，正奔騰！！

—— **弗雷格的理念**就是想讓『**表達**』和『**論證**』**清晰明白** ——

——— 摘自《白馬非馬論》

如是者當聽聞『三原色』說：

原色

原色是指不能透過其他顏色的混合調配而得出的「基本色」。

以不同比例將原色混合，可以產生出其他的新顏色。以數學的向量空間來解釋色彩系統，則原色在空間內可作為一組基底向量，並且能組合出一個「色彩空間」。由於人類肉眼有三種不同顏色的感光體，因此所見的色彩空間通常可以由三種基本色所表達，這三種顏色被稱為「**三原色**」。一般來說疊加型的三原色是紅色、綠色、藍色（又稱三基色，用於電視機、投影儀等顯示設備）；而消減型的三原色是洋紅色、黃色、青色（用於書本、雜誌等的印刷）。

以及

格拉斯曼定律 (色彩)

格拉斯曼定律是一個關於光學理論的經驗法測，他說明了人類對色彩的感知（大約）是線性的。這個定律是由格拉斯曼所發現的。

敘述

若兩單色光組合成一測試色光，則觀測者感知到的三原色數值為兩單色光分別被單獨觀測的三原色數值之和。換句話說，如果光束一及光束二為單色光，而 (R_1, G_1, B_1)與 (R_2, G_2, B_2)分別為觀測者對光束一及光束二的感知三原色數值，當此二光束合併時，觀測者感知的三原色數值為 (R, G, B)，其中：

$$R = R_1 + R_2$$

$$G = G_1 + G_2$$

$$B = B_1 + B_2$$

更一般的來說，格拉斯曼定律說明了任一光束的三原色座標為

$$R = \int_0^\infty I(\lambda)\,\bar{r}(\lambda)\,d\lambda$$

$$G = \int_0^\infty I(\lambda)\,\bar{g}(\lambda)\,d\lambda$$

$$B = \int_0^\infty I(\lambda)\,\bar{b}(\lambda)\,d\lambda$$

$I(\lambda)$為該光束對波長的強度分布；$\bar{r}(\lambda)$，$\bar{g}(\lambda)$，$\bar{b}(\lambda)$則分別為人眼中三種錐狀細胞對不同波長的反應強度。

自能曉得任選『自然』或『科技』中『可調變』之『紅』、『綠』、『藍』，比方講用

磷光體

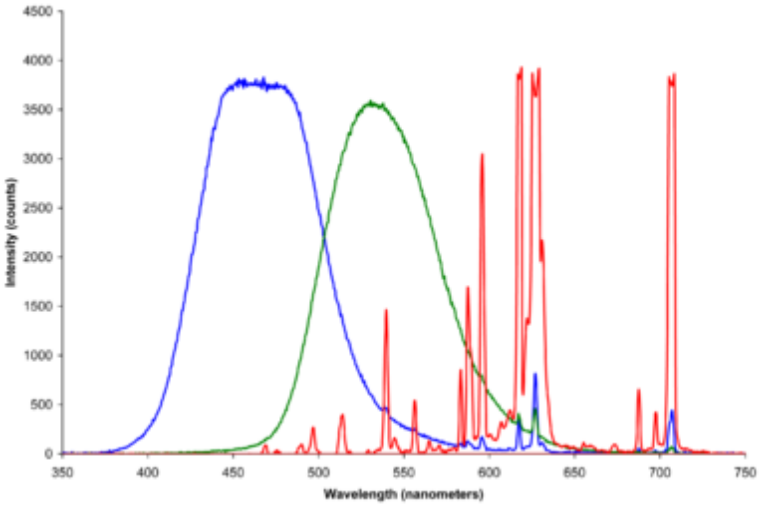
磷光體(Phosphor)是產生冷發光現象的物質，包括亮度衰減緩慢的(>1ms)磷光材料和發光衰減在幾十奈秒的螢光材料。磷光材料在雷達螢幕及夜光玩具上用得較多，螢光材料在CRT和電漿顯示器、傳感器和發光二極體中更為常見。

磷光體一般是各種過渡金屬化合物或者稀土金屬化合物。磷光體長用在陰極射線管顯示器和螢光燈中。CRT磷光體在第二次世界大戰時標準化，表示方法為P加數字。

元素磷的發光原理是化學發光而不是磷光[1]，所以磷不是磷光體。

製造

Cathode ray tubes



Spectra of constituent blue, green and red phosphors in a common cathode ray tube.
Cathode ray tubes produce signal-generated light patterns in a (typically) round or rectangular format. Bulky CRTs were used in the black-and-white household television (“TV”) sets that became popular in the 1950s, as well as first-generation, tube-based color TVs, and most earlier computer monitors. CRTs have also been widely used in scientific and engineering instrumentation, such as oscilloscopes, usually with a single phosphor color, typically green. Phosphors for such applications may have long afterglow, for increased image persistence.

The phosphors can be deposited as either thin film, or as discrete particles, a powder bound to the surface. Thin films have better lifetime and better resolution, but provide less bright and less efficient image than powder ones. This is caused by multiple internal reflections in the thin film, scattering the emitted light.

White (in black-and-white): The mix of zinc cadmium sulfide and zinc sulfide silver, the ZnS:Ag+(Zn,Cd)S:Ag is the white **P4** phosphor used in black and white television CRTs. Mixes of yellow and blue phosphors are usual. Mixes of red, green and blue, or a single white phosphor, can also be encountered.

Red: Yttrium oxide–sulfide activated with europium is used as the red phosphor in color CRTs. The development of color TV took a long time due to the search for a red phosphor. The first red emitting rare earth phosphor, $YVO_4:Eu^{3+}$, was introduced by Levine and Palilla as a primary color in television in 1964.[21] In single crystal form, it was used as an excellent polarizer and laser material.[22]

Yellow: When mixed with cadmium sulfide, the resulting **zinc cadmium sulfide** (Zn,Cd)S:Ag, provides strong yellow light.

Green: Combination of zinc sulfide with copper, the **P31** phosphor or ZnS:Cu, provides green light peaking at 531 nm, with long glow.

Blue: Combination of zinc sulfide with few ppm of silver, the ZnS:Ag, when excited by electrons, provides strong blue glow with maximum at 450 nm, with short afterglow with 200 nanosecond duration. It is known as the **P22B** phosphor. This material, **zinc sulfide silver**, is still one of the most efficient phosphors in cathode ray tubes. It is used as a blue phosphor in color CRTs.

The phosphors are usually poor electrical conductors. This may lead to deposition of residual charge on the screen, effectively decreasing the energy of the impacting electrons due to electrostatic repulsion (an effect known as “sticking”). To eliminate this, a thin layer of aluminium (about 100 nm) is deposited over the phosphors, usually by vacuum evaporation, and connected to the conductive layer inside the tube. This layer also reflects the phosphor light to the desired direction, and protects the phosphor from ion bombardment resulting from an imperfect vacuum.

To reduce the image degradation by reflection of ambient light, contrast can be increased by several methods. In addition to black masking of unused areas of screen, the phosphor particles in color screens are coated with pigments of matching color. For example, the red phosphors are coated with ferric oxide (replacing earlier Cd(S,Se) due to cadmium toxicity), blue phosphors can be coated with marine blue (CoO·*n*Al2O3) or ultramarine (Na8Al6Si6O24S2). Green phosphors based on ZnS:Cu do not have to be coated due to their own yellowish color.[2]

，只要一條『白光方程式』：

【colormodels.py】

```
# sRGB (ITU-R BT.709) standard phosphor chromaticities
SRGB_Red = xyz_color (0.640, 0.330)
SRGB_Green = xyz_color (0.300, 0.600)
SRGB_Blue = xyz_color (0.150, 0.060)
SRGB_White = xyz_color (0.3127, 0.3290) # D65


def init (
    phosphor_red   = SRGB_Red,
    phosphor_green = SRGB_Green,
    phosphor_blue  = SRGB_Blue,
    white_point    = SRGB_White):
……
    global PhosphorRed, PhosphorGreen, PhosphorBlue, PhosphorWhite
    PhosphorRed   = phosphor_red
    PhosphorGreen = phosphor_green
    PhosphorBlue  = phosphor_blue
    PhosphorWhite = white_point
    global xyz_from_rgb_matrix, rgb_from_xyz_matrix
    phosphor_matrix = numpy.column_stack ((phosphor_red, phosphor_green, phosph$
    # normalize white point to Y=1.0
    normalized_white = white_point.copy()
    xyz_normalize_Y1 (normalized_white)
    # Determine intensities of each phosphor by solving:
    #     phosphor_matrix * intensity_vector = white_point
    intensities = numpy.linalg.solve (phosphor_matrix, normalized_white)
    # construct xyz_from_rgb matrix from the results
    # construct xyz_from_rgb matrix from the results
    xyz_from_rgb_matrix = numpy.column_stack (
        (phosphor_red   * intensities [0],
         phosphor_green * intensities [1],
         phosphor_blue  * intensities [2]))
    # invert to get rgb_from_xyz matrix
    rgb_from_xyz_matrix = numpy.linalg.inv (xyz_from_rgb_matrix)
```

，就可形成一個『顯色空間』耶！！？？

```
pi@raspberrypi:~ $ ipython3 --pylab
Python 3.4.2 (default, Oct 19 2014, 13:31:11)
Type "copyright", "credits" or "license" for more information.

IPython 2.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
Using matplotlib backend: TkAgg

In [1]: import colorpy.colormodels

In [2]: 磷光紅 = colorpy.colormodels.SRGB_Red

In [3]: 磷光綠 = colorpy.colormodels.SRGB_Green

In [4]: 磷光藍 = colorpy.colormodels.SRGB_Blue

In [5]: 參考白 = colorpy.colormodels.SRGB_White

In [6]: 磷光紅
Out[6]: array([ 0.64,  0.33,  0.03])

In [7]: 磷光綠
Out[7]: array([ 0.3,  0.6,  0.1])

In [8]: 磷光藍
Out[8]: array([ 0.15,  0.06,  0.79])

In [9]: 參考白
Out[9]: array([ 0.3127,  0.329 ,  0.3583])

In [10]: 磷光矩陣 = numpy.column_stack ((磷光紅, 磷光綠, 磷光藍))

In [11]: 磷光矩陣
Out[11]:
array([[ 0.64,  0.3 ,  0.15],
       [ 0.33,  0.6 ,  0.06],
       [ 0.03,  0.1 ,  0.79]])

In [12]: 光度歸一白 = 參考白.copy()

In [13]: 光度歸一白
Out[13]: array([ 0.3127,  0.329 ,  0.3583])

In [14]: colorpy.colormodels.xyz_normalize_Y1(光度歸一白)
Out[14]: array([ 0.95045593,  1.        ,  1.08905775])

In [15]: 光度歸一白
Out[15]: array([ 0.95045593,  1.        ,  1.08905775])

In [16]: 白光分量強度 = numpy.linalg.solve(磷光矩陣, 光度歸一白)

In [17]: 白光分量強度
```

```
Out[17]: array([ 0.64436062,  1.1919478 ,  1.20320526])

In [18]: 由RGB轉XYZ矩陣 = numpy.column_stack (
   ....:       (磷光紅 * 白光分量強度[0],
   ....:        磷光綠 * 白光分量強度[1],
   ....:        磷光藍 * 白光分量強度[2]))

In [19]: 由RGB轉XYZ矩陣
Out[19]:
array([[ 0.4123908 ,  0.35758434,  0.18048079],
       [ 0.21263901,  0.71516868,  0.07219232],
       [ 0.01933082,  0.11919478,  0.95053215]])

In [20]: 由XYZ轉RGB矩陣 = numpy.linalg.inv(由RGB轉XYZ矩陣)

In [21]: 由XYZ轉RGB矩陣
Out[21]:
array([[ 3.24096994, -1.53738318, -0.49861076],
       [-0.96924364,  1.8759675 ,  0.04155506],
       [ 0.05563008, -0.20397696,  1.05697151]])

In [22]:

```