

Aquisição de Imagens

Aquisição de Imagens

- Amostragem:
 - Função bi-dimensional $I(x,y)$

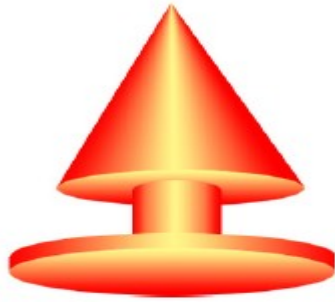
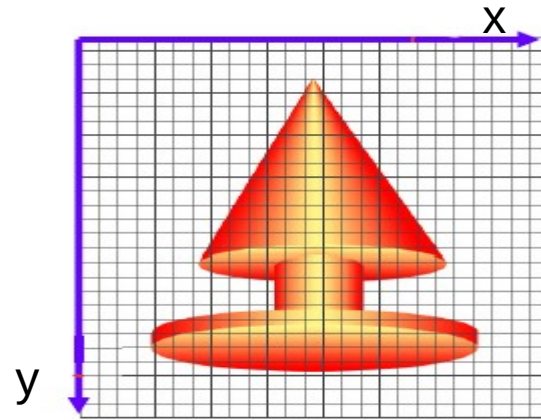


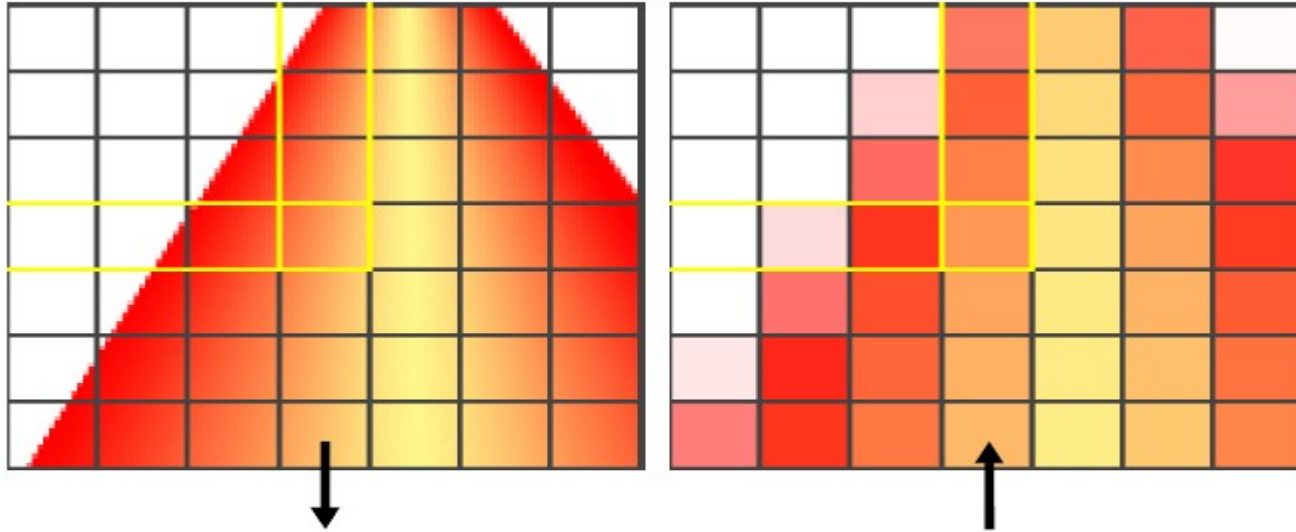
Imagem Real



Grid de pixels

Aquisição de Imagens

- Quantização



Aquisição de Imagens

- Resultado da amostragem

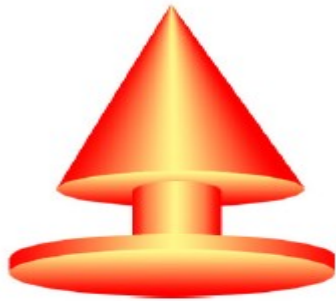


Imagem Real

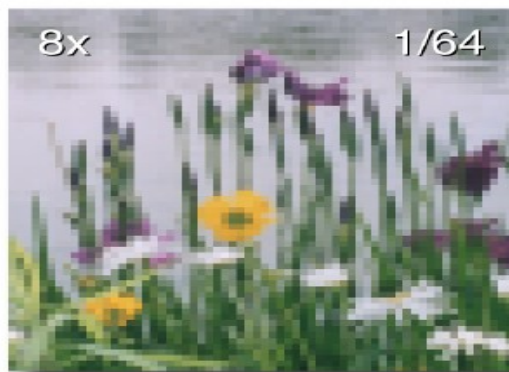
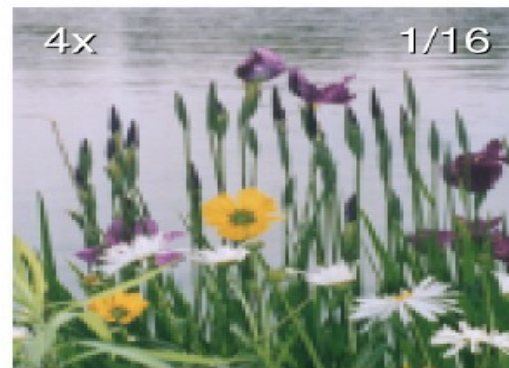
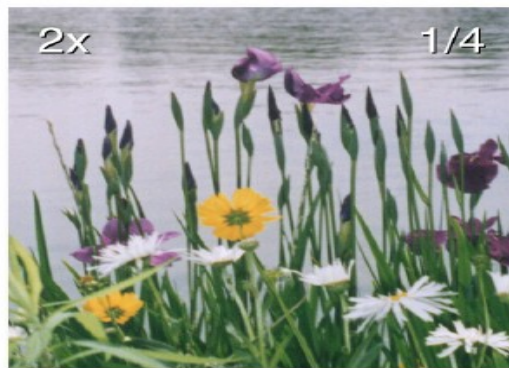


Imagem amostrada

Aquisição de Imagens

- Resolução

Pixelization of
Color Images:
All Bands Equal




Exemplo

trinket Python3 Run Share

main.py

```
1 from PIL import Image
2 import urllib.request
3
4 urllib.request.urlretrieve('https://www.rioverde.com.br/wp-content/uploads/2018/04/UNIP-02.jpg', "unip.png")
5 img = Image.open("unip.png")
6
7 w,h = img.size
8 img.save("unip.png")
9
```

Powered by trinket



unip.png

The image shows a screenshot of a Trinket Python3 IDE. The code in the editor downloads an image from a URL and saves it as 'unip.png'. To the right, the resulting image is displayed, showing an aerial view of a large, modern university building with a prominent 'UNIP' sign, a large parking lot, and surrounding greenery. The image is labeled 'unip.png' with a small blue icon.

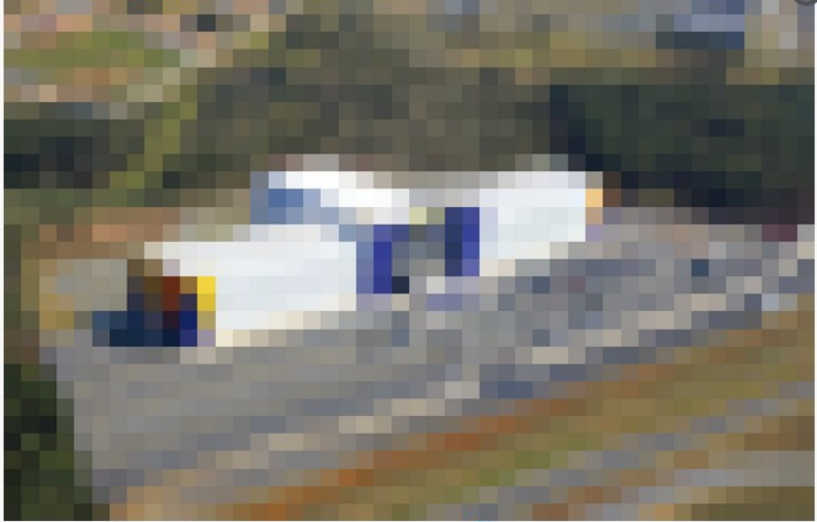
Exemplo

trinket Python3 Run Share

main.py

```
1 from PIL import Image
2 import urllib.request
3
4 urllib.request.urlretrieve('https://www.rioverde.com.br/wp-content/uploads/2018/04/UNIP-02.jpg', "unip.png")
5 img = Image.open("unip.png")
6
7 w,h = img.size
8 img = img.resize((int(w/16),int(h/16)))
9 img = img.resize((w,h),Image.NEAREST) # Image.BILINEAR
10
11 img.save("unip.png")
12
```

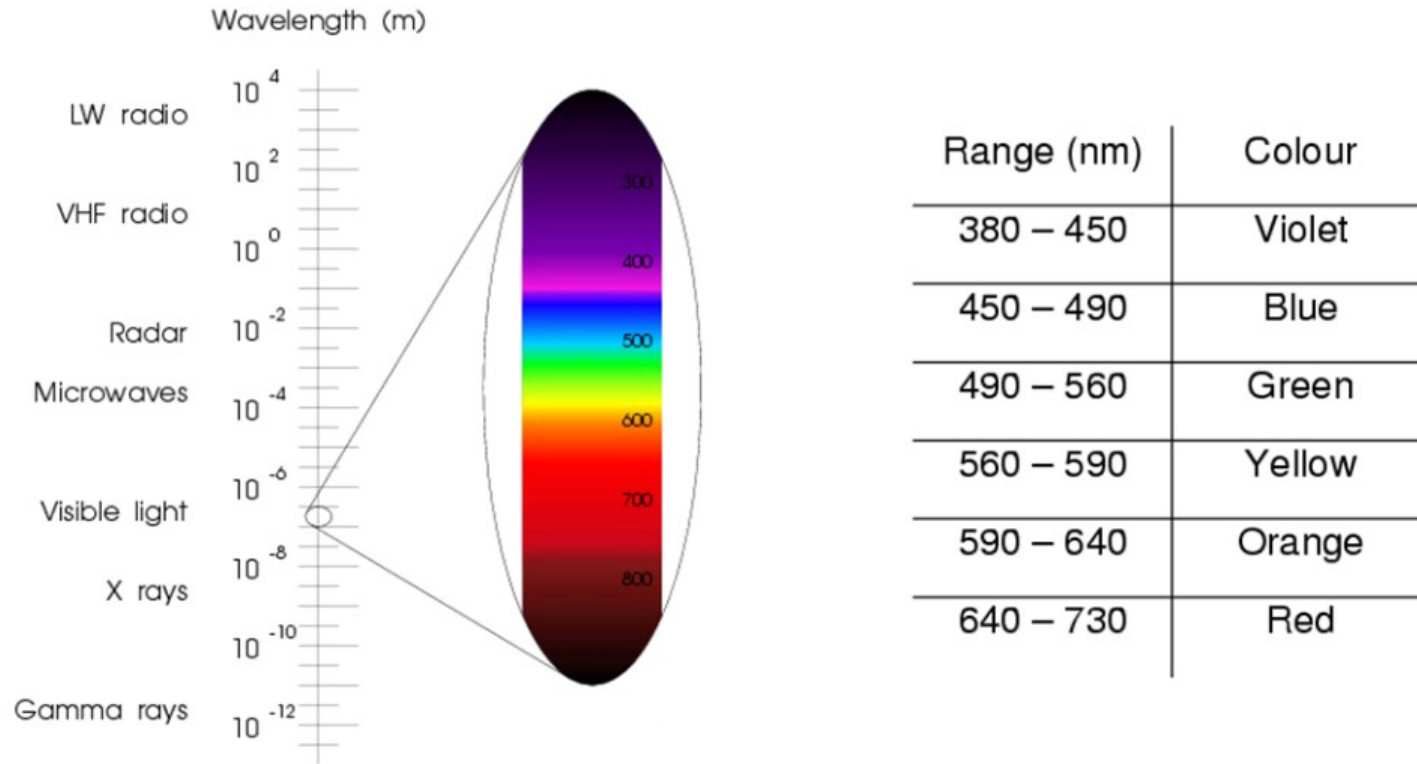
Powered by trinket



unip.png

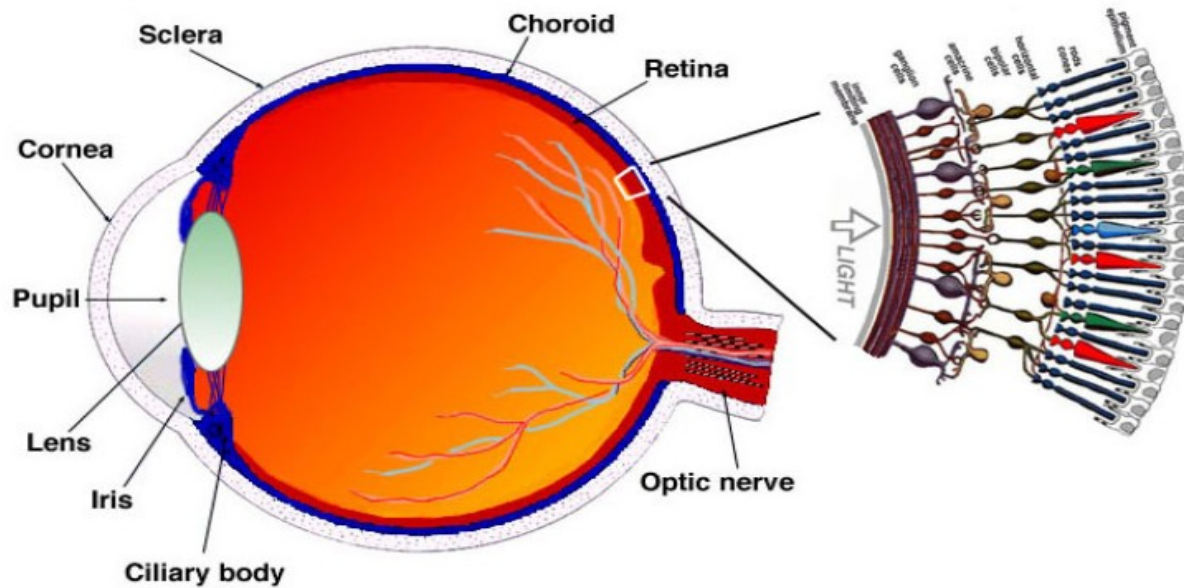
Modelo de Cores

Espectro visível



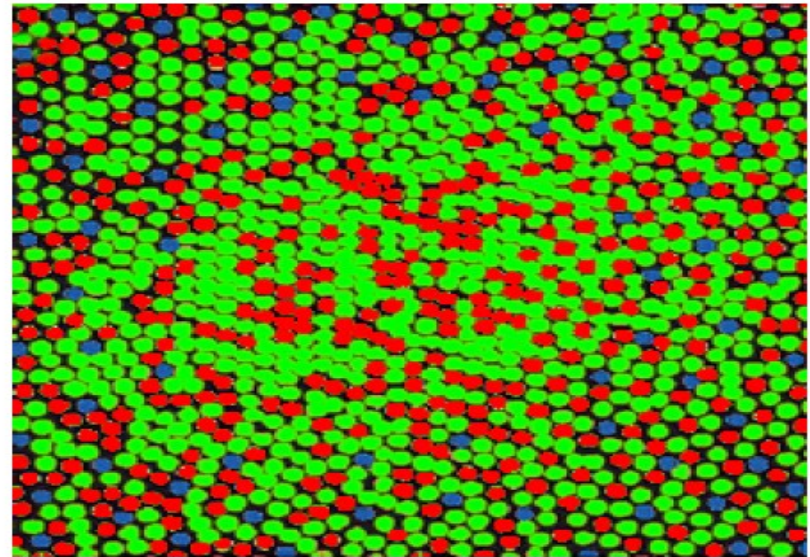
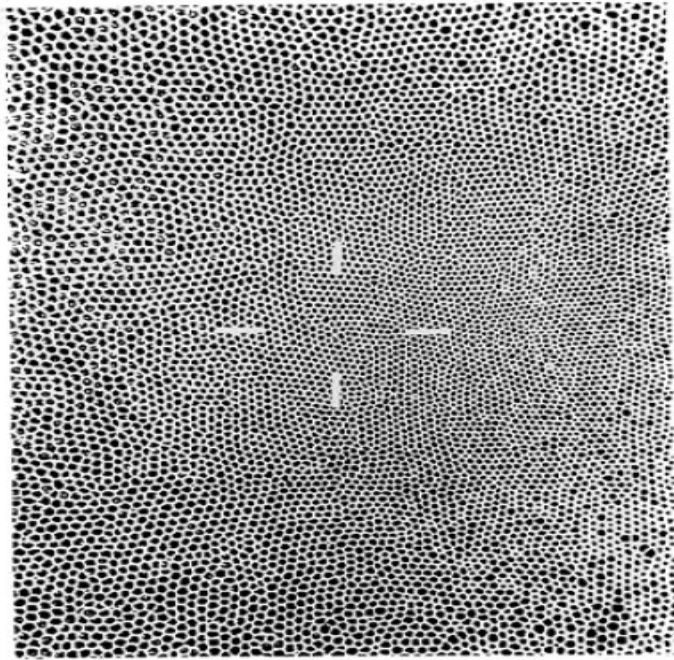
Modelo de Cores

- Percepção humana



Modelo de Cores

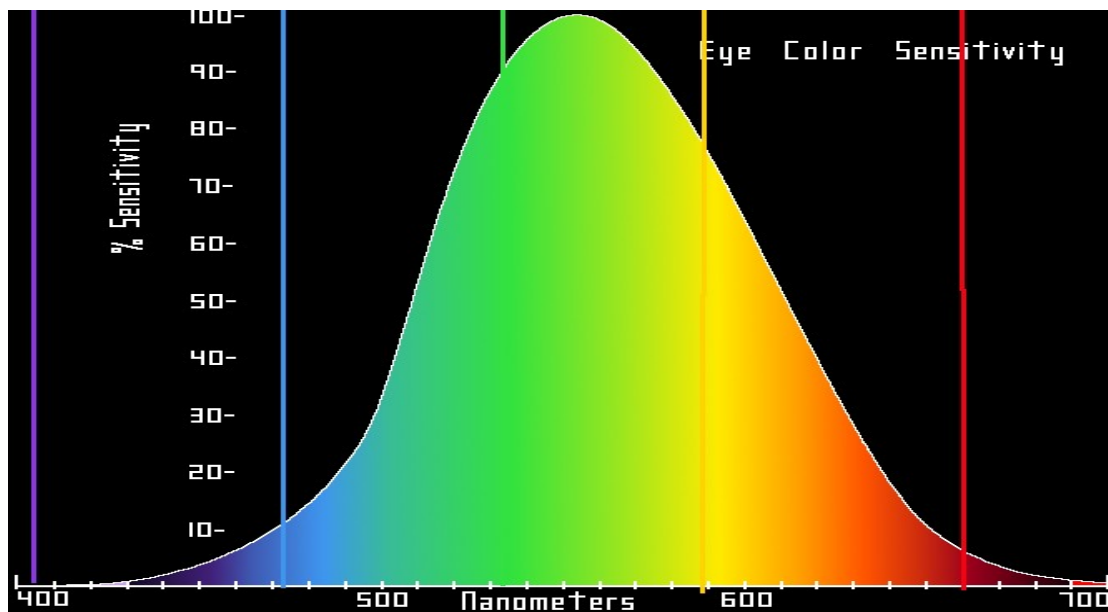
- Mosaico da retina



Cepko, Connie, "Giving in to the blues",
Nature Genetics, 24, 99 - 100 (2000)
cepko@genetics.med.harvard.edu

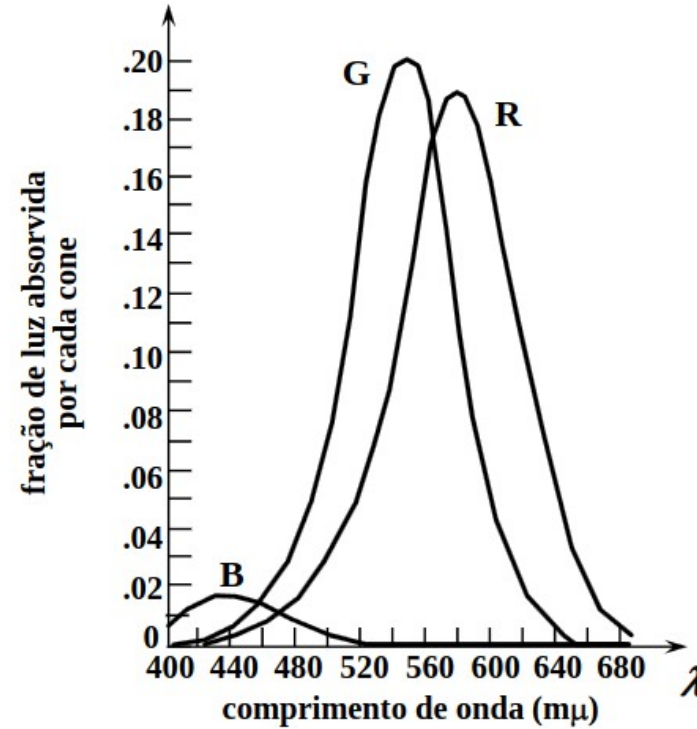
Modelo de Cores

- Sensibilidade



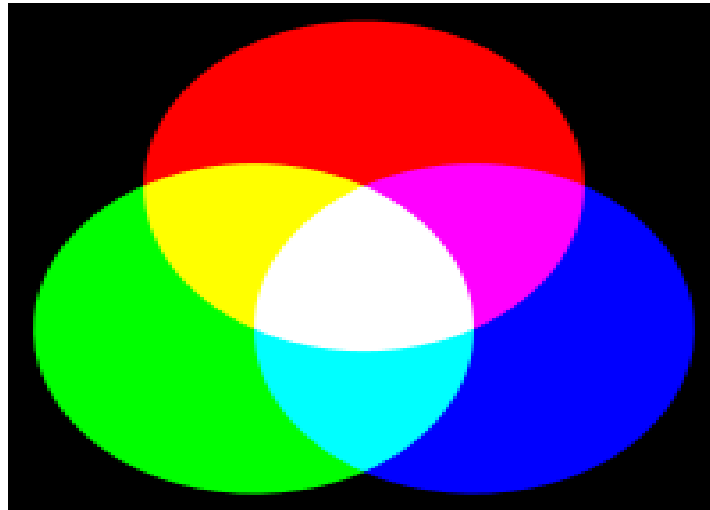
Modelo de Cores

- Sensibilidade



RGB

- Modelo de cores aditivas
 - reprodução de cores em dispositivos eletrônicos



RGB

16× Pixelization
of Color Images:
R, G, & B Bands



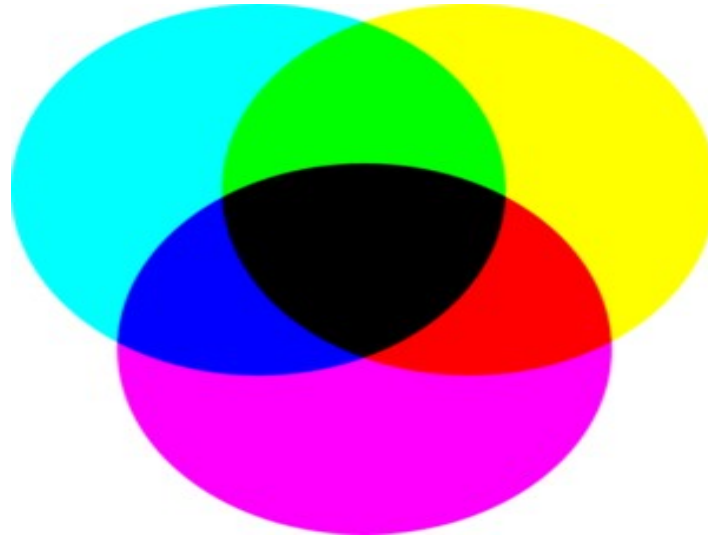
Exercício

- Reproduzir o resultado anterior
 - `r, g, b = img.split()`
 - `img = Image.merge('RGB', (r, g, b))`



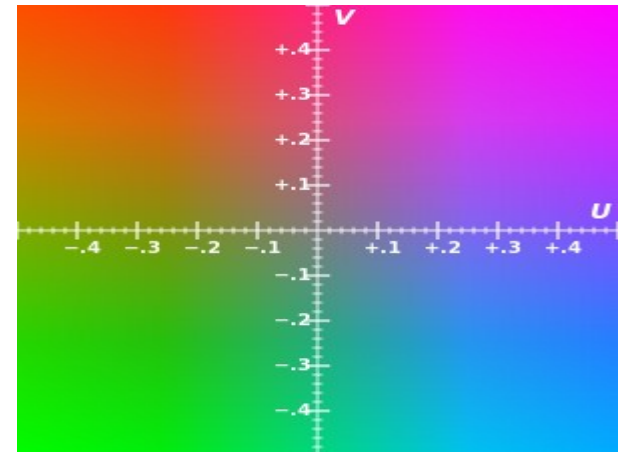
CMYK

- Modelo de cores subtrativas
 - C: Ciano, M: magenta, Y: amarelo, K: Preto
 - empregado por impressoras, impressoras e fotocopiadoras



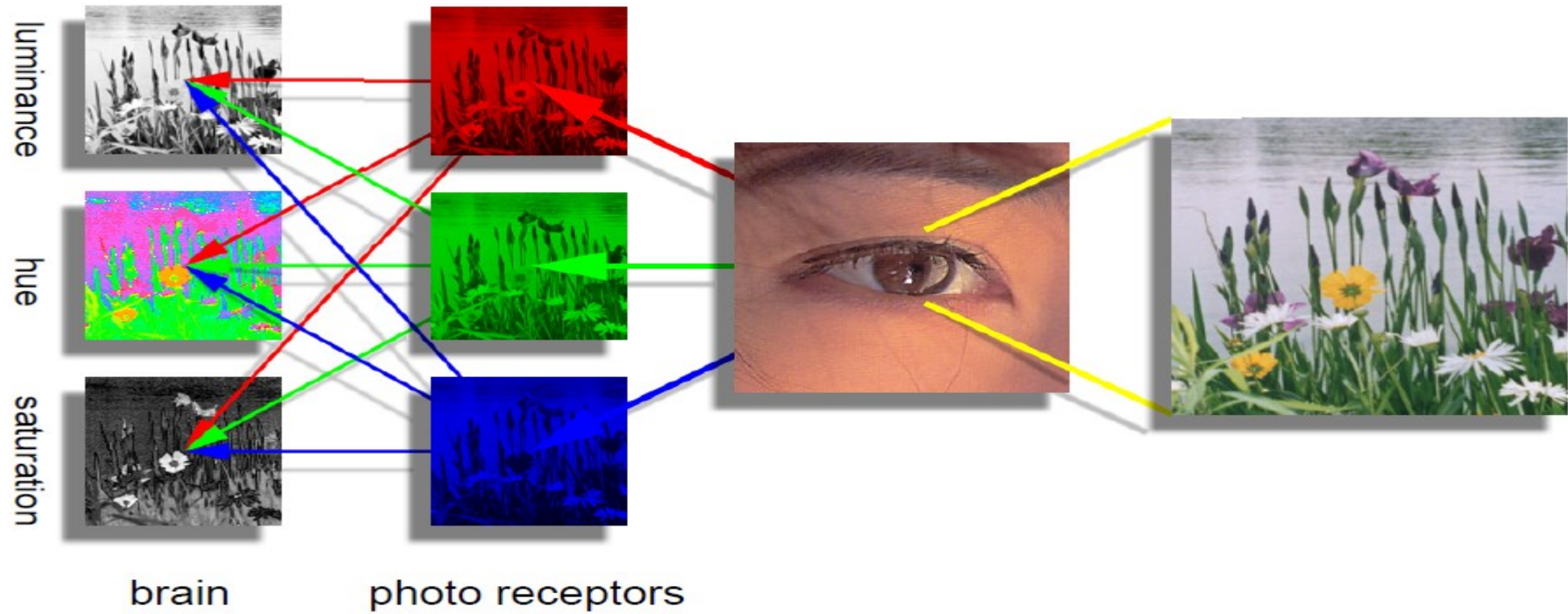
YUV

- Codifica imagens coloridas levando mais em consideração a percepção humana
- Aplicação de Visão computacional
 - Deve ser mais robusta a alterações em iluminação.
 - Cor e iluminação são representados separadamente
- Y → Luminância
- UV → Crominância



YUV

- Percepção



YUV



YUV

- Modelo YUV
 - Conversões entre modelos

$$W_R = 0.299$$

$$W_B = 0.114$$

$$W_G = 1 - W_R - W_B = 0.587$$

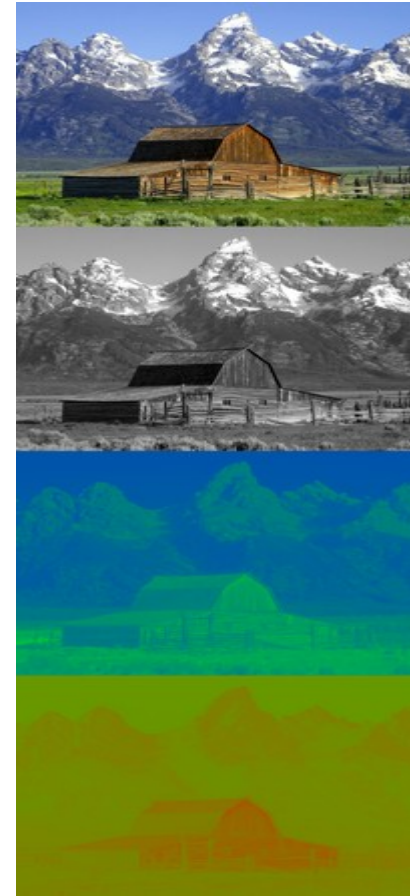
$$U_{Max} = 0.436$$

$$V_{Max} = 0.615$$

$$Y' = W_R R + W_G G + W_B B$$

$$U = U_{Max} \frac{B - Y'}{1 - W_B} \approx 0.492(B - Y')$$

$$V = V_{Max} \frac{R - Y'}{1 - W_R} \approx 0.877(R - Y')$$



YUV

- Modelo YUV
 - Forma matricial

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix}$$

Exercício

- Repetir o exercício anterior para YUV
 - `img_yuv = img.convert('YcbCr')`
 - `y, u, v = img_yuv.split()`
 - `img = Image.merge('YCbCr', (y, u, v))`

