Master: Ciencia de Datos e Ingeniería de Computadores

Curso: Visión por Computador

Rosa Mª Rodríguez Sánchez

Dpto. Ciencias de la Computación e I. A.

E.T.S. de Ingenierías Informática y de Telecomunicaciones

Universidad de Granada

Modelos de color. El histograma

Índice de contenido

1. Introducción

Un modelo de color no es más que un sistema que permite definir un color en función de una serie de parámetros. Estos parámetros pueden ser, por ejemplo, colores básicos tal que mezclándolos se obtienen nuevos colores. Matlab permite trabajar con distintos espacios de color, entre los más populares están RGB, YIQ, HSV, CMY o HSI. Para convertir entre ellos disponemos de varias funciones.

1.1. El modelo YIQ

Las componentes de este modelo de color son la luminancia (Y), que contiene información sobre el nivel de gris (la intensidad de luz reflejada por la imagen), y otras dos componentes que codifican la información de color: la fase (I) y la cuadratura (Q). Este modelo se utilizaba por el sistema de televisión NTSC. La transformación entre YIQ y RGB se define como:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

En Matlab, para convertir una imagen RGB a YIQ se utiliza la función rgb2ntcs:

```
>> imYIQ = rgb2ntcs(imRGB)
```

La clase de la matriz que tiene la imagen RGB puede ser uint8, uint16 o double y el resultado será double en cualquier caso.

Para convertir de YIQ a RGB se usa ntsc2rgb:

```
>> imRGB = ntcs2rgb(imYIQ)
```

Tanto la imagen YIQ como la RGB serán de clase double.

La ventaja de este modelo frente al RGB es que es más fácil de interpretar por una persona al separar la luminancia del color.

1.2. El modelo YcbCr

Este espacio es muy parecido al YUV y se utiliza en los estándares MPEG-2 y JPEG. Y es la componente de luminancia y Cb y Cr son las componentes cromáticas.

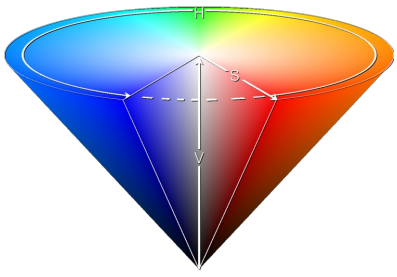
$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Matlab usa las funciones rgb2ycbcr y ycbcr2rgb para convertir a o desde RGB:

```
>> imYCbCr = rgb2ycbcr(imRGB)
>> imRGB = ycbcr2rgb(imYCbCr)
```

La imagen que se va a convertir puede ser de clase uint8, uint16 o double y el resultado es del mismo tipo.

1.3. El modelo HSV



También es conocido como HSB. Las tres componentes de este modelo son:

Tono (H): Indica el color (rojo, azul, verde, ...). La gama de colores se dispone en forma de círculo de manera que este valor es un ángulo entre 0 y 360º, aunque para trabajar con él está normalizado al rango [0,1].

Saturación (S): Indica la pureza del color. Este valor varía entre 0% y 100%. Cuanto más cercano al 0%, el color será más

grisáceo (hasta llegar al blanco) y cuando más cercano al 100% el color será más vivo o puro. También se normaliza al intervalo [0,1].

Valor o brillo (V): El rango de valores también está entre el 0% y el 100% (normalizados al intervalo [0,1]). Indica la intensidad del color: si la intensidad es baja el color será más oscuro.

Matlab permite pasar del espacio RGB al HSV mediante la función rgb2hsv:

```
>> imHSV = rgb2hsv(imRGB)
```

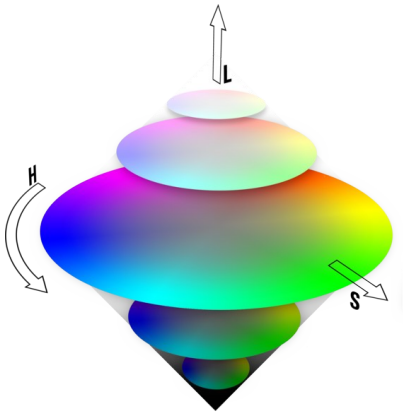
donde imRGB puede ser de clase uint8, uint16 o double y el resultado es de clase double. La conversión inversa se hace con hsv2rgb:

```
>> imRGB = hsv2rgb(imHSV)
```

en donde ambas imágenes son de clase double.

1.4. El modelo HSI

También se conoce como HSL. Sus tres componentes son el tono la saturación y el brillo (o intensidad). Es muy similar al espacio HSV aunque refleja mejor las nociones de saturación y luminosidad.



Matlab no tiene funciones de conversión entre RGB y HSI. La formulación la puedes encontrar en la bibliografía y en internet, por ejemplo en:

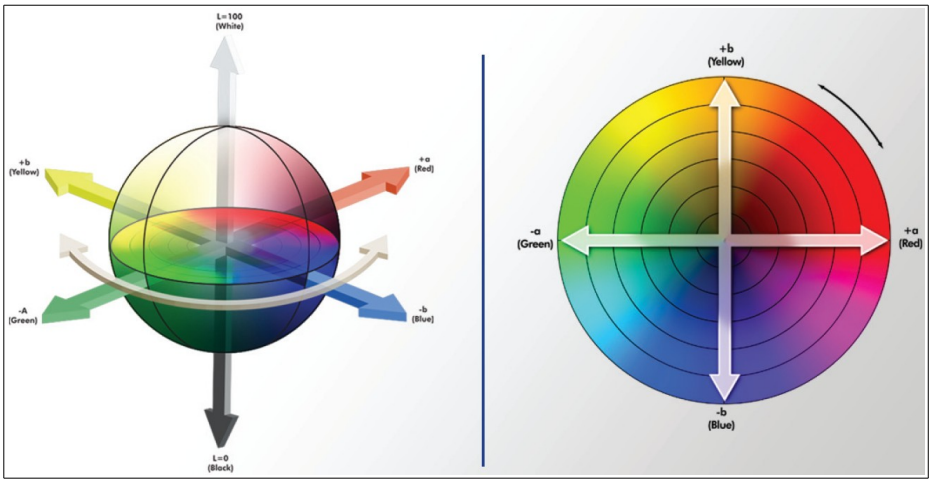
http://en.wikipedia.org/wiki/HSL_color_space

En el libro de Gonzalez, Woods y Eddins hay una formulación alternativa aunque menos eficiente ya que usa operaciones computacionalmente más costosas como la raíz cuadrada o el coseno.

1.5. El modelo CIELAB (CIE 1976 L*a*b*)

Este modelo de color, derivado del espacio CIE 1931 XYZ, tiene dos propiedades importantes:

- Independencia del dispositivo. Al contrario que los otros de modelos de color, este es independiente del dispositivo.
- Linealidad perceptual. En este modelo, la distancia euclídea entre dos colores está altamente correlada con la diferencia perceptual que percibiría un observador humano.



Este modelo tiene dos componentes cromáticas (a y b) y una componente de luminancia. En concreto, la componente a define la posición entre el verde y el rojo mientras que la componente b define la posición entre el azul y el amarillo.

En Matlab disponemos de dos funciones que nos permiten pasar de este a otros espacios de color y viceversa. La función makecform, crea una estructura que permite convertir entre espacios de color. La función applycform aplica esa transformación a una imagen.

Algunas de las transformaciones que podemos crear con makecform son las siguientes:

- srgb2lab: para convertir de RGB a CIELab.
- lab2srgb: para convertir de CIELab a RGB.

Por ejemplo, dada una imagen X en el espacio RGB, la podemos convertir a CIELab así:

```
>> ff = makecform('srgb2lab');
>> result = applycform(X,ff);
```

Para convertir de CIELab a RGB:

```
>> ff = makecform('lab2srgb');
>> result = applycform(X,ff);
```

En Matlab, los valores Lab pueden ser de tipo uint8, uint16 y double. Para convertir el tipo, disponemos de funciones como lab2uint8 o lab2double. El rango de valores, dependiendo del tipo de almacenamiento es el siguiente:

	double	**uint8**	**uint16**
L	0.0 – 100.0	0 - 255	0 - 65280
a	-128.0 – 127.0	0 - 255	0 - 65280
b	-128.0 – 127.0	0 - 255	0 - 65280

2. El histograma

El histograma de una imagen de niveles de gris representa la frecuencia de aparición de cada nivel de gris. Cuando se trata de imágenes en color se obtiene un histograma por cada banda de color de la imagen.

Para ver el histograma de una imagen de niveles de gris en matlab usaremos la función imhist:

```
>> a=imread('alhambra.jpg');
>> b=rgb2gray(a);
>> imhist(b)
```

En este ejemplo visualizaremos el histograma de la figura anterior (el asociado a la imagen en niveles de gris). Podemos agrupar los niveles de grises en intervalos de igual tamaño para hacer el histograma. Por ejemplo:

```
>> imhist(b,4)
```

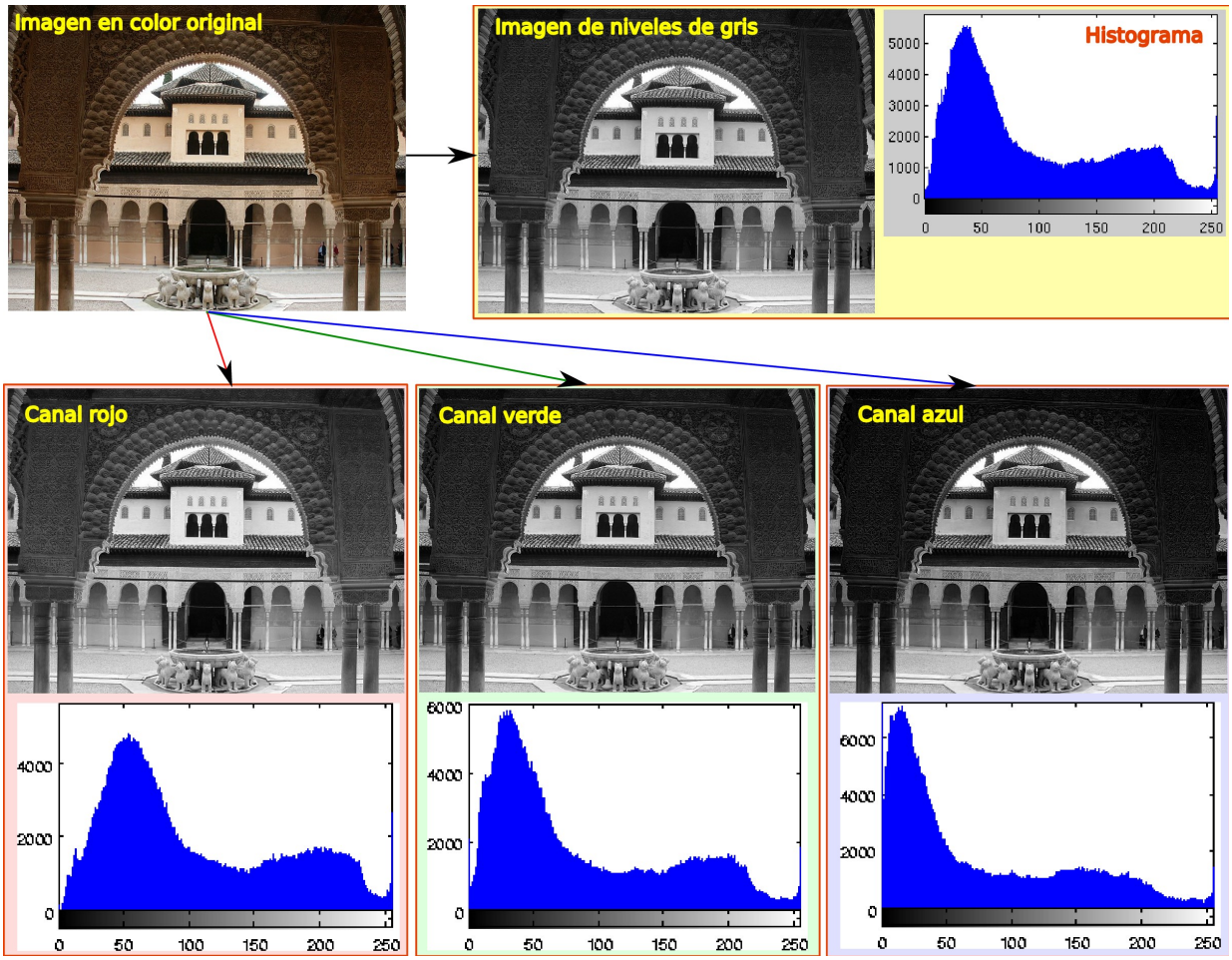
toma los 256 posibles valores de gris y los agrupa en cuatro intervalos de manera que el histograma representa la frecuencia de aparición de los niveles de gris dentro de cada intervalo. Los intervalos, en este ejemplo, serían [0,64[, [64, 128[, [128,192[, [192, 256[.

Si en lugar de visualizar el histograma lo que queremos es almacenar los valores para hacer algún cálculo con ellos, llamaremos a la función de la forma:

```
>> [y x] = imhist(b)
```

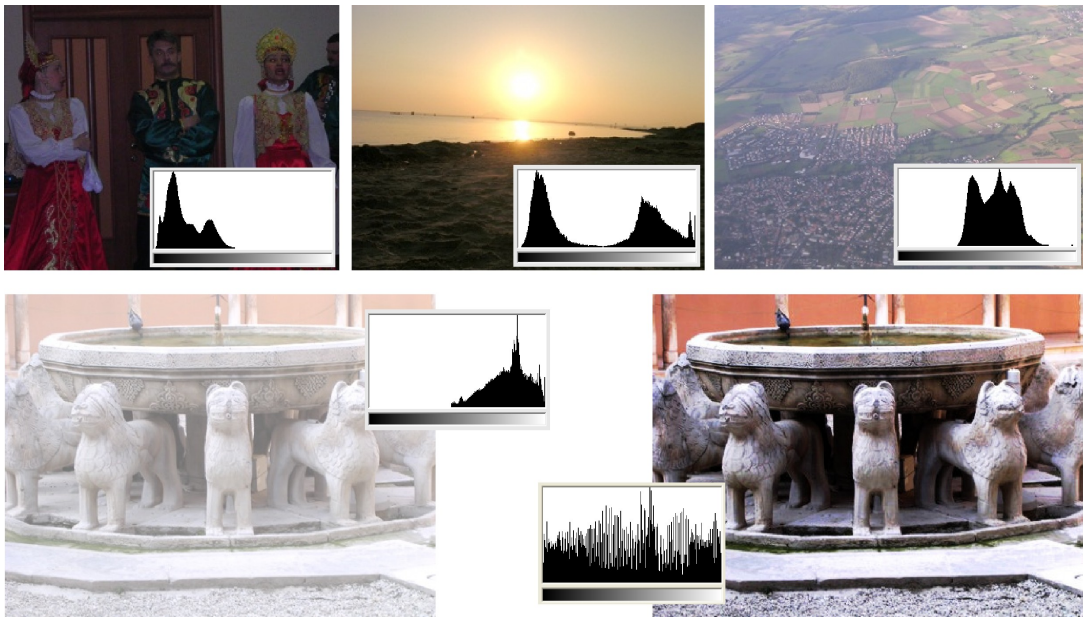
donde x representa la clase de conteo e y la frecuencia de aparición.

También es frecuente hacer uso del histograma acumulado, que se define, para cada clase de conteo X, como la cuenta de todos los valores menores o iguales que los de la clase X.



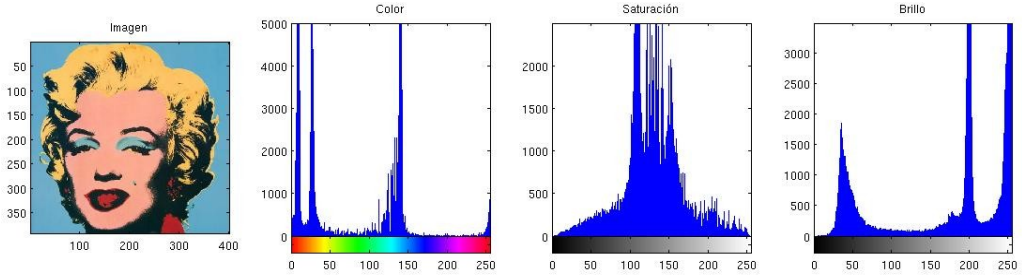
Generalmente, una imagen bien contrastada suele usar, más o menos por igual, todos los niveles de gris. Es decir, el número de ocurrencias de cada nivel es aproximadamente el mismo. Cuando en un histograma observamos que hay ciertos grupos de niveles de gris que no se usan, o bien otros que se usan de manera muy desigual, posiblemente la imagen no tenga buena calidad.

En la siguiente figura vemos cuatro imágenes mal contrastadas, con sus correspondientes histogramas, y una (la última) bien contrastada.



3. Ejercicios

1. Cargar una imagen RGB y convertirla a HSV. Para ello implementa una función que reciba como entrada la imagen en formato RGB y que muestre una ventana con la imagen y los histogramas de sus tres componentes HSV.



   Para la componente de Color, muestra en la barra inferior el color con que se corresponde cada valor mostrado (usa el parámetro map de la función imhist y la función hsv para obtener el mapa de color de este modelo).

2. Haz otra función que, de forma análoga, muestre los histogramas de las componentes RGB. Esta vez no hay que mostrar ningún mapa de color específico para ninguna de ellas.

3. Implementa alguna modificación sobre la componente H de la imagen. Por ejemplo, suma un valor constante a dicho componente y visualiza la imagen resultante.

4. Chroma Key. Haz una función que dada una imagen con un color de fondo casi uniforme (en el ejemplo primer plano) inserte esta imagen en otra (en el ejemplo segundo

Primer Plano

Segundo Plano

Chroma Key

   plano). Para ello el fondo de color uniforme en la imagen primer plano se sustituye por los valores de la imagen segundo plano. La función debe dar la posibilidad de insertar la imagen primer plano en diferentes posiciones de la imagen segundo plano.

5. Crea una animación en la que, dada una imagen, se pueda ver como van ciclando los colores. Para ello, cada fotograma se obtendrá sumando un pequeño incremento a la componente H del fotograma anterior. Para ello puedes usar las funciones im2frame y movie.