PROCESAMIENTO DE IMAGENES

Programa de teoría

1. Adquisición y representación de imágenes.
2. Procesamiento global de imágenes.
3. Filtros y transformaciones locales.
4. Transformaciones geométricas.
5. **Espacios de color y el dominio frecuencial.**
6. Análisis de imágenes.

Tema 5. Espacios de color y el dominio frecuencial.

5.1. Qué es el color.

5.2. Modelos y espacios de color.

5.3. El dominio frecuencial.

5.4. Otras transformaciones lineales.

A.5. Color y transf. lineales en OpenCV.

5.1. Qué es el color.

- Hasta ahora hemos considerado el color como una simple **tupla de tres números (R, G, B)**... pero esto es únicamente un modelo de color concreto, y no el único.
- ¿Qué es el color? ¿Cuál es su naturaleza?
  - Los **objetos** tienen color. El color es una propiedad de los objetos.

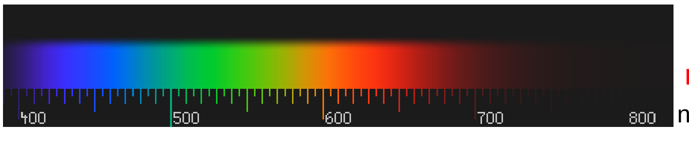
    FALSO. Los objetos reflejan o absorben ciertas frecuencias
  - La **luz** tiene color. El color es una propiedad de la luz.

    FALSO. La luz es una radiación electromagnética en cierto rango de frecuencias
- Según los expertos, el **color** es una **sensación humana**, derivada de la capacidad del ojo de captar los niveles de radiación en 3 frecuencias diferentes.
- Por extensión, hablamos del color de la luz y de los objetos.

5.1. Qué es el color.

- La **luz** es una **radiación electromagnética** en el rango de frecuencias de los 405 a los 790 THz. (740 a 380 nm).
- Este rango de frecuencias es lo que se conoce como la **radiación visible**.

Ultra-violetas



Infra-rrojos

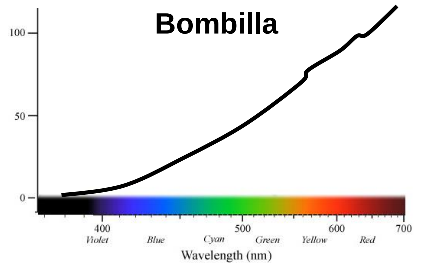
nm

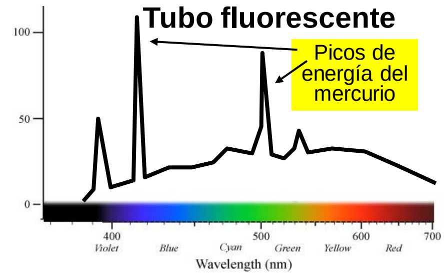
http://en.wikipedia.org/wiki/Color

- Pero la luz no es (normalmente) un simple punto en este rango, sino que se forma **combinando un poco de cada frecuencia**.
- La descomposición de una fuente de luz en sus componentes frecuenciales es lo que se conoce como el **espectro** (o **descomposición espectral**) de esa luz.

5.1. Qué es el color.

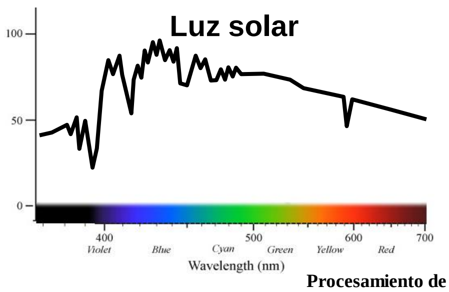
- **Ejemplo**. Comparación de los espectros de luz solar, lámpara incandescente (una bombilla) y un tubo fluorescente.





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

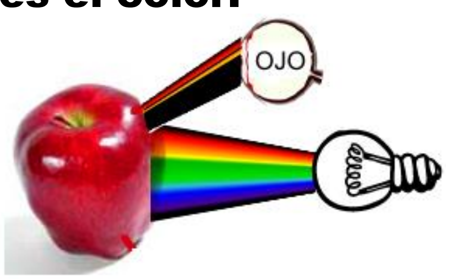
http://astroneu.com/plasma-redshift-1/



Los tres espectros son muy distintos. Y, sin embargo, las tres luces nos parecen blancas...

5.1. Qué es el color.

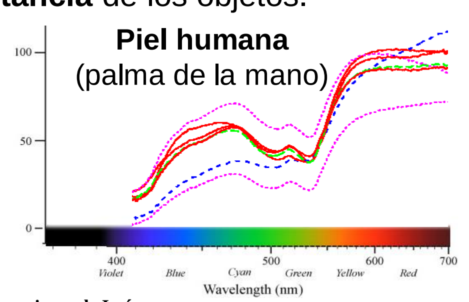
- En los **objetos**, la sensación de color se produce por la **absorción de frecuencias** incidentes.



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

- De manera análoga a la luz, tenemos los **espectros de absorción** y de **reflectancia** de los objetos.

- **Ejemplo**. Espectro de reflectancia de la piel humana.



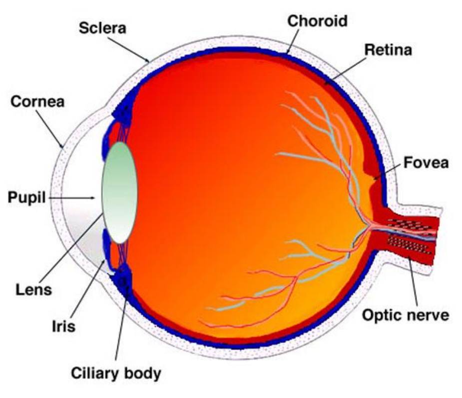
http://www.cis.upenn.edu/~elli/tech-report.skin.pdf

5.1. Qué es el color.

- Un **modelo preciso del "color"** debería definir una especie de tabla: **color: array [400,...,700] de real**.
- Sin embargo, sólo usamos tres enteros (R, G, B). ¿Por qué?
- Porque lo que se modela es la **sensación humana** de color, que es mucho más limitada que el espectro de frecuencias.

Anatomía de la retina del ojo humano

http://webvision.med.utah.edu/spanish/indice.html

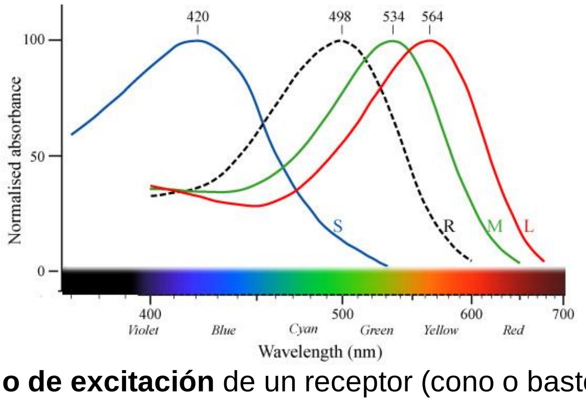


- Existen 2 tipos de receptores: **bastones** (*rod*) y **conos** (*cone*).
- Grosso modo, podemos decir que los bastones captan **intensidad**, y los conos (3 tipos distintos) captan **color**.
- Hay muchos menos conos que bastones.

5.1. Qué es el color.

- Los conos se clasifican según la longitud de onda de la radiación que captan en: bajos (**S**), medios (**M**) y altos (**L**).

Absorción espectral de los conos (S,M,L) y los bastones (R)

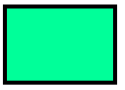


http://en.wikipedia.org/wiki/Color

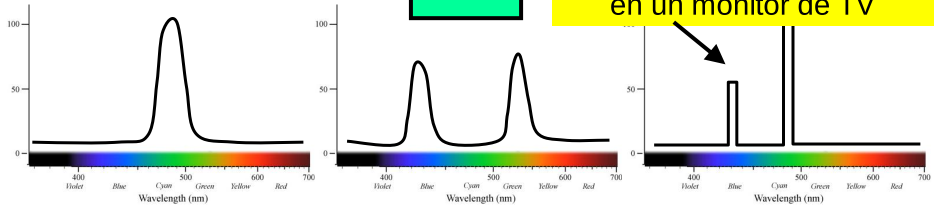
- El **grado de excitación** de un receptor (cono o bastón) sería una suma, para cada frecuencia, del producto de la absorción por la intensidad de luz entrante.

5.1. Qué es el color.

- Los **conos M** y **L** están muy correlacionados entre sí. Los **M** son más próximos al **verde**, y los **L** al **rojo**.
- Los **conos S** responden principalmente al **azul**. Existen muchos menos. Pero el cerebro se encarga de compensarlo.
- Así que el ojo humano es capaz de detectar menos colores que combinaciones espectrales existen.
- **Ejemplo**. Todos estos espectros de luz son percibidos como si fueran el mismo color.

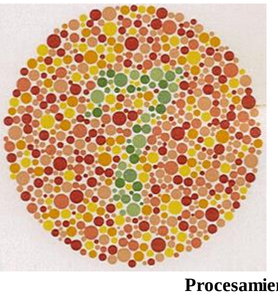


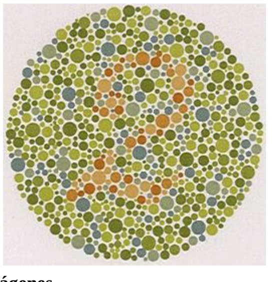
Esto sería lo más parecido a cómo se genera el color en un monitor de TV



5.1. Qué es el color.

- Formalmente, se puede decir que un **color** es la **variedad de espectros** (en principio, infinita) que dan lugar a la misma excitación de los conos y bastones del ojo humano.
  - El color es fruto de una **sensación humana**, no de la naturaleza intrínseca de la luz.
  - Un modelo de color completo debería tener por lo menos **tres dimensiones**.
- ... un ojo humano medio, porque hay ojos y ojos...



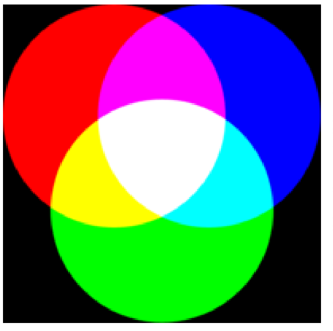


http://en.wikipedia.org/wiki/Color_blindness

5.1. Qué es el color.

- El significado diferente del color en **la luz** (espectro emitido) y en **los objetos** (espectro reflejado) da lugar a dos modos de ver el proceso de mezcla de colores: modelo **aditivo** y **substractivo**.

Modelo aditivo mezcla:

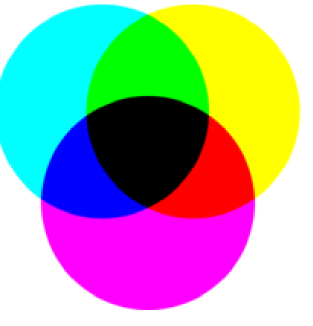


http://en.wikipedia.org/wiki/Color_space

- Corresponde a los colores **luz**.
- La mezcla de dos colores se obtiene **sumando** los espectros asociados a ambos colores.
- Podemos seleccionar un nº reducido de **colores primarios** y obtener los demás con ellos.
- Por ejemplo, si tomamos como **primarios rojo**, **verde** y **azul**, los **secundarios** son **amarillo**, **magenta** y **cian**.

5.1. Qué es el color.

Modelo substractivo de mezcla:

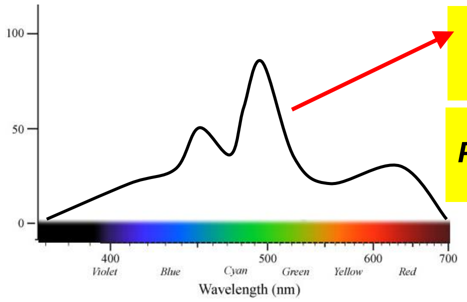


http://en.wikipedia.org/wiki/Color_space

- Corresponde a los colores **pigmento**.
- La mezcla de dos colores se obtiene tomando el **producto** de los espectros asociados a ambos colores.
- Si tomamos como **colores primarios** cian, magenta y amarillo, los **colores secundarios** son rojo, azul y verde.
- **Colores terciarios**: los que surgen de combinar un primario con un secundario.
- **Ojo**: tanto en uno como en otro, la elección de los colores primarios no es única.

5.2. Modelos y espacios de color.

- **Definición**: un **modelo de color** es un modelo matemático abstracto, que describe la forma en que se representan los colores mediante tuplas de números (normalmente 3 ó 4).
- El **conjunto de colores** posibles que surgen de estas tuplas es conocido como el **espacio de color**.
- El **modelo matemático** diría: dado un espectro de luz, ¿cómo se obtiene la tupla de color correspondiente?



Ejemplo, en el modelo **RGB** esto es: (14, 26, 17)

En el modelo ***Pepito*** es: (29, 14, 73)

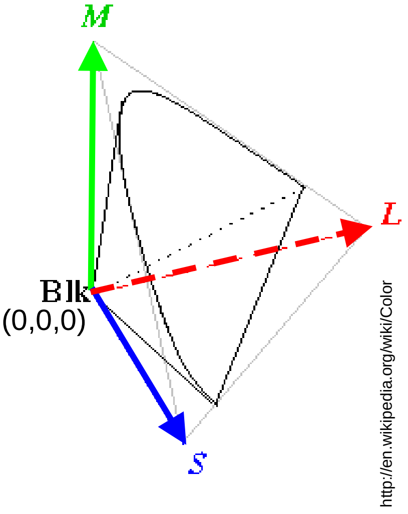
5.2. Modelos y espacios de color.

- El estudio de los modelos de color es importante, porque el modelo condiciona cómo se **captura**, **almacena**, **procesa**, **transmite** y **genera** el color.
- Existen muchos modelos de color. Algunos son mejores para ciertas aplicaciones. **No todos son completos**.
  - RGB es el que más se ajusta al modo de captura (filtros de color) y de generación (píxeles del monitor) en imagen digital.
  - CMYK se relaciona con la generación de color en impresoras.
  - YIQ y YUV separan crominancia (color) y luminancia (brillo).
  - XYZ está relacionado con la sensación humana de color.
- Las **operaciones** estudiadas en los temas anteriores se pueden aplicar usando **diferentes espacios**:
  - Transformaciones globales: aritméticas (binarias o unarias), lógicas, de comparación (diferencia entre imágenes), etc.
  - Transformaciones locales: convoluciones, morfológicos, rellenado de regiones, etc.

5.2. Modelos y espacios de color.

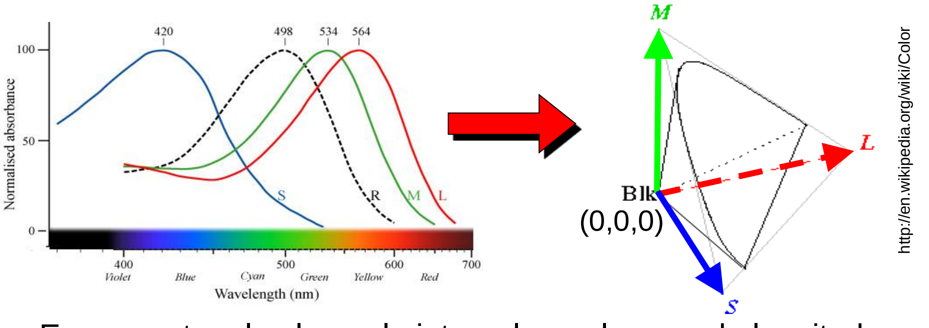
El modelo triestímulo:

- Es el modelo más próximo a la percepción humana del color.
- El **modelo triestímulo** es un espacio 3D, donde cada dimensión corresponde al nivel de excitación de cada tipo de cono (S, M, L).
- Es un **modelo completo**: cualquier sensación humana de color caerá en un punto de este espacio.
- El espacio es ilimitado.



5.2. Modelos y espacios de color.

- El valor en cada eje se obtiene usando las **funciones de transferencia** asociadas a cada tipo de **cono**.

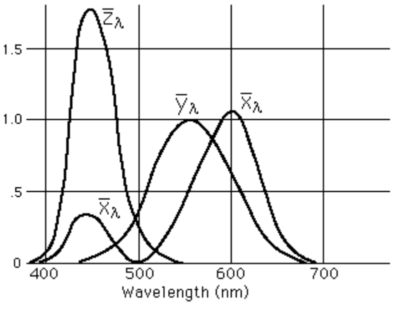


- En concreto, el valor es la integral, en el rango de longitudes de onda, del producto de la función de transferencia por el espectro de la luz analizada.
- La típica **forma de herradura** surge por la correlación entre las funciones de S, M y L.

5.2.1. El modelo CIE XYZ.

El modelo C.I.E. XYZ:

- La **C.I.E.** (*Commission Internationale d'Eclairage*) es el organismo encargado de los **estándares de color**.
- El modelo CIE XYZ fue uno de los primeros que creó (1931).
- Intuitivamente, es una **proyección** del modelo triestímulo que separa por un lado el componente de **luminancia** (**Y**) y por otro la **crominancia** o **color** (**x**, **y**).
- CIE XYZ define unas funciones de transferencia para cada parámetro (**X**, **Y**, **Z**), que se asemejan, a las de (**L**, **M**, **S**), respectivamente.



http://hyperphysics.phy-astr.gsu.edu/hbase/vision/cieprim.html

5.2.1. El modelo CIE XYZ.

- Se definen también los valores **x** e **y**:

  x= X/(X+Y+Z)

  y= Y/(X+Y+Z)

  Valores normalizados en intensidad

- La representación del plano (**x, y**) da lugar al **diagrama cromático CIE**.

Modelo triestímulo

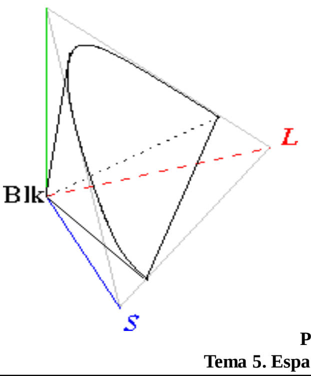


Diagrama cromático CIE

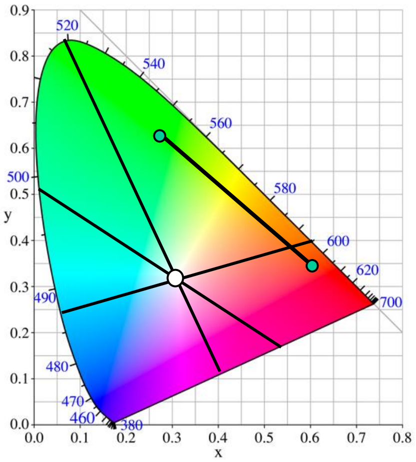
Estos son (x,y), falta el eje Y, la luminosidad

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

5.2.1. El modelo CIE XYZ.

- **Propiedades** del diagrama cromático:
  - La curva exterior son los **colores espectrales**. El resto son colores **no espectrales** (o colores compuestos).
  - La **suma** de dos colores se encuentra en la línea que los une.
  - El **blanco** se encuentra en x= 1/3, y= 1/3. La línea que une dos colores **complementarios** pasa por ese punto.
  - El diagrama es **completo**, contiene todos los colores visibles por los humanos.

Diagrama cromático CIE

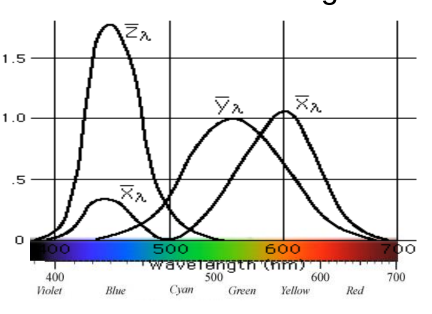


http://en.wikipedia.org/wiki/Color

5.2.1. El modelo CIE XYZ.

- **Ejemplo**. Descomposición en canales XYZ de una imagen.

Imagen de entrada



5.2.2. El modelo RGB.

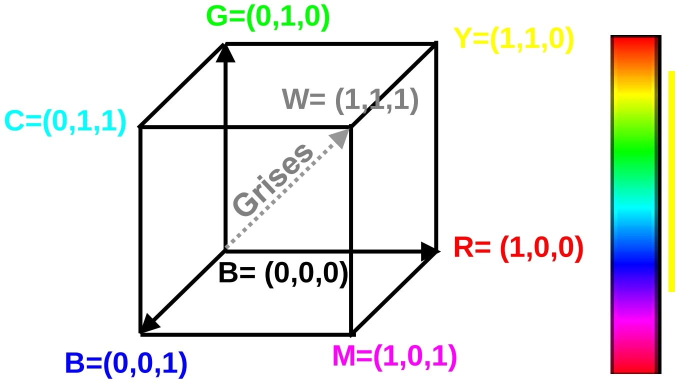
El modelo RGB:

- Los modelos anteriores son poco prácticos en aplicaciones como adquisición y generación de color en TV, impresoras...
- Se usa más el **RGB**, basado en un **modelo aditivo de mezcla**, con 3 **colores primarios**: **R-rojo**, **G-verde**, **B-azul**.
- La combinación aditiva de estos colores primarios produce todo el rango de colores representables en RGB.

(R,G,B)	Long. de onda	Frecuencia
(255,0,0)	~ 625-740 nm	~ 480-405 THz
(255,128,0)	~ 590-625 nm	~ 510-480 THz
(255,255,0)	~ 565-590 nm	~ 530-510 THz
(0,255,0)	~ 500-565 nm	~ 600-530 THz
(0,255,255)	~ 485-500 nm	~ 620-600 THz
(0,0,255)	~ 440-485 nm	~ 680-620 THz
(0,139,255)	~ 380-440 nm	~ 790-680 THz

5.2.2. El modelo RGB.

- El **espacio RGB** tiene forma de **cubo** de lado 1.
- El punto (R=0,G=0,B=0) es el **negro**, y el (R=1,G=1,B=1) es el **blanco**.

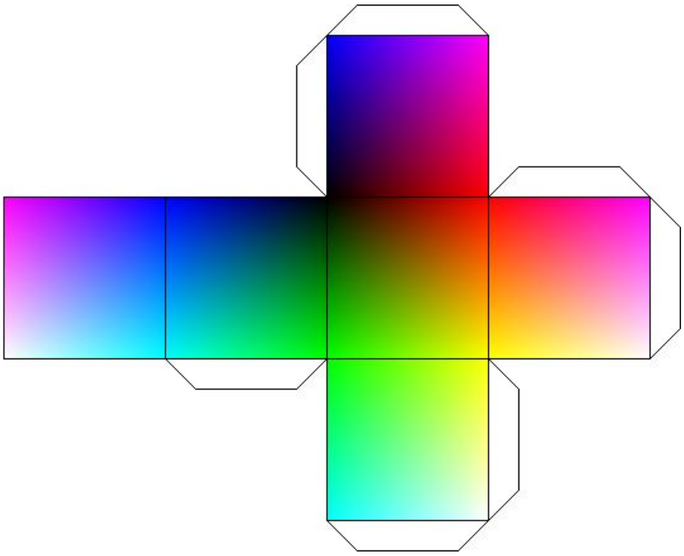


Recorriendo las aristas exteriores del cubo sale algo “parecido” a un espectro

- Surgen tres **colores secundarios**: cian, magenta y amarillo.

5.2.2. El modelo RGB.

Desarrollo del cubo RGB

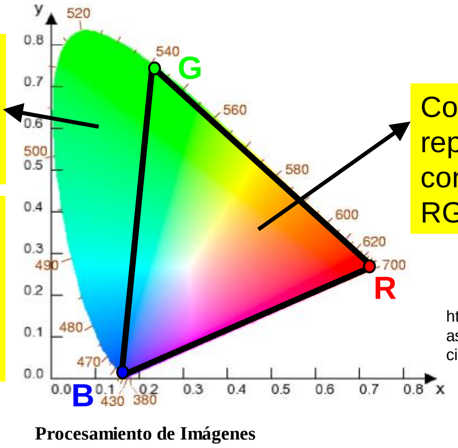


5.2.2. El modelo RGB.

- El espacio RGB es el más utilizado en la práctica.
- Pero **no es completo**: existen colores que no se pueden obtener con la combinación de R, G y B.
- Se puede comprobar en el diagrama cromático CIE.

Colores visibles por los humanos pero no representables en el modelo RGB

De hecho no podrás verlos aquí, porque estas transparencias usan el modelo RGB...



Colores representables con el modelo RGB

http://hyperphysics.phy-astr.gsu.edu/hbase/vision/cie.html

5.2.2. El modelo RGB.

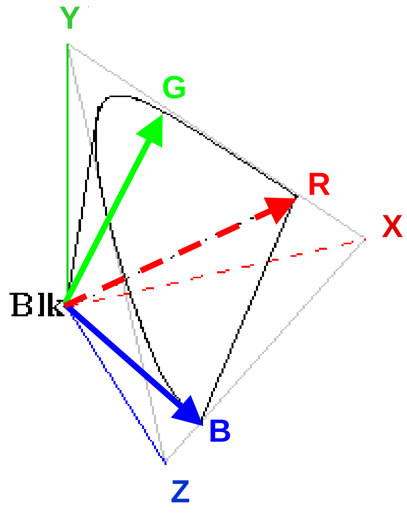
- El espacio RGB se relaciona de manera **lineal** con el CIE XYZ. Se puede entender como una rotación del espacio XYZ.

- Transformación **XYZ a RGB**:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3,24 & -1,5 & -0,5 \\ -0,9 & 1,88 & 0,04 \\ 0,06 & -0,2 & 1,05 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

- Transformación **RGB a XYZ**:

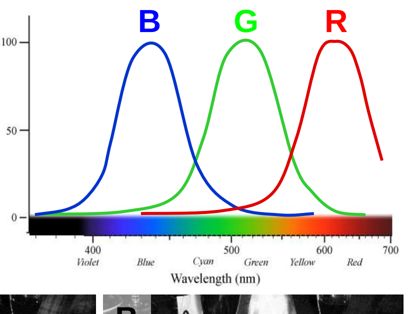
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0,41 & 0,36 & 0,18 \\ 0,21 & 0,72 & 0,07 \\ 0,02 & 0,12 & 0,95 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$



5.2.2. El modelo RGB.

- **Ejemplo**. Descomposición en canales RGB de una imagen.

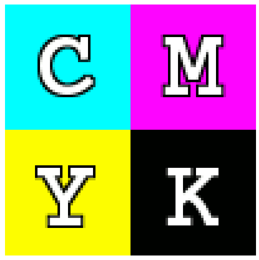
Imagen de entrada



5.2.3. El modelo CMY.

El modelo CMY:

- En ciertas aplicaciones, como por ejemplo **impresión de imágenes**, se utiliza más el modelo **CMY** (o **CMYK**).
- **CMY** está basado en un **modelo sustractivo de mezcla**, con 3 **colores primarios**: **C-cian**, **M-magenta**, **Y-amarillo**.
- La combinación sustractiva (**tintas de color**) de estos colores primarios produce todo el rango de colores representables en CMY.
- En la práctica, la mezcla de **C**, **M** e **Y** no llega a producir **negro**, sino una especie de **gris marengo**.
- El modelo **CMYK** soluciona el problema, añadiendo el **negro** como color primario.

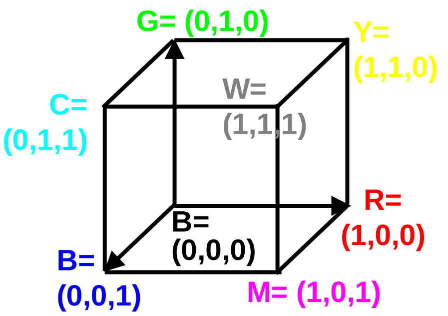


http://en.wikipedia.org/wiki/CMYK

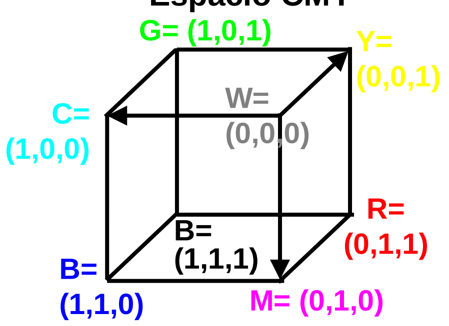
5.2.3. El modelo CMY.

- El **espacio CMY** es el mismo que el RGB, solo que viendo el cubo “desde el lado opuesto”.

Espacio RGB



Espacio CMY

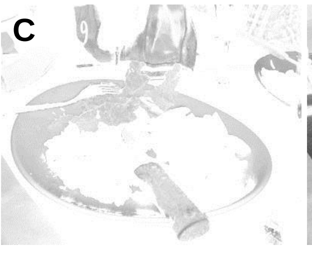


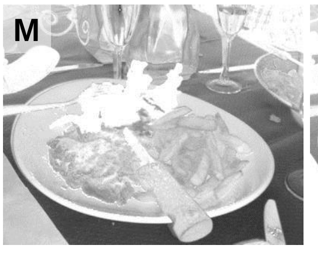
- **Conversión RGB → CMY**:
  C:= 1 - R
  M:= 1 - G
  Y:= 1 - B
- **Conversión CMY → RGB**:
  R:= 1 - C
  G:= 1 - M
  B:= 1 - Y

5.2.3. El modelo CMY.

- **Ejemplo**. Descomposición en canales CMYK de la imagen.

Imagen de entrada



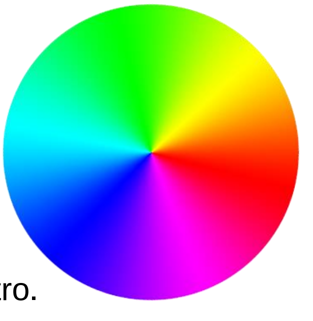


5.2.4. Modelos HLS y HSV.

Los modelos HLS y HSV:

- Los modelos HLS (o HSL) y HSV están pensados para ser fácilmente **interpretables y legibles** por un humano, usan términos más familiares cuando hablamos de color.
- **Luminosidad o intensidad de un color**: cualidad de ser más claro o más oscuro.
- **Saturación**: diferencia del color respecto a un gris con la misma intensidad. Cuanto más diferente, más saturado.
- **Matiz de un color**: su ángulo dentro de la rueda cromática.
- También, se puede definir como la **frecuencia dominante** del espectro.

Rueda cromática

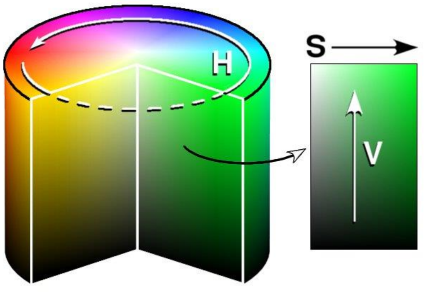


http://en.wikipedia.org/wiki/HSV_color_space

5.2.4. Modelos HLS y HSV.

- **HSV** consta de los componentes: **H**-matiz (*hue*), **S**-saturación, **V**-valor de intensidad.
- **HLS** consta de: **H**-matiz, **L**-luminosidad, **S**-saturación.
- Ambos son transformaciones no lineales del RGB.
- La definición de **H** es igual en ambos. La diferencia se encuentra en la forma de calcular la saturación, **S**, y la intensidad, **V** o **L**.

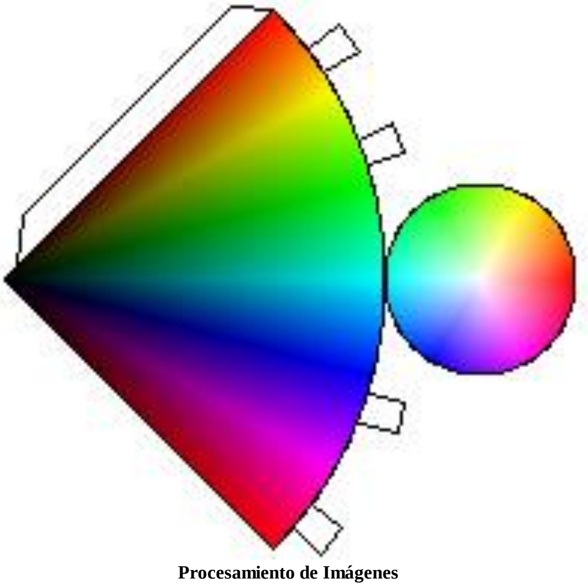
- El **espacio HSV** se suele representar como un **cono**.
- O como un **cilindro**.



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

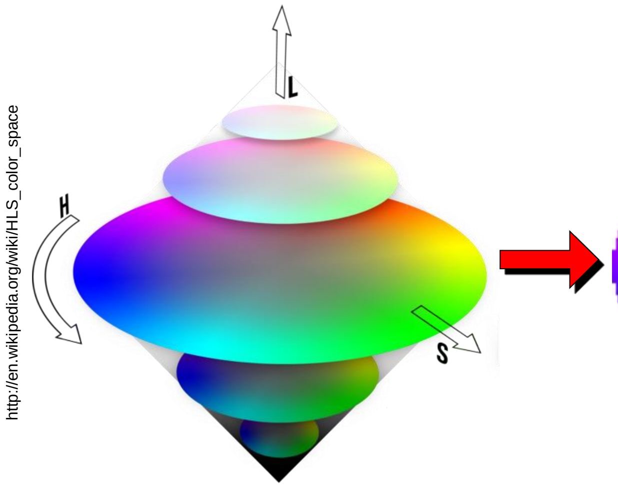
5.2.4. Modelos HLS y HSV.

Desarrollo del cono HSV



5.2.4. Modelos HLS y HSV.

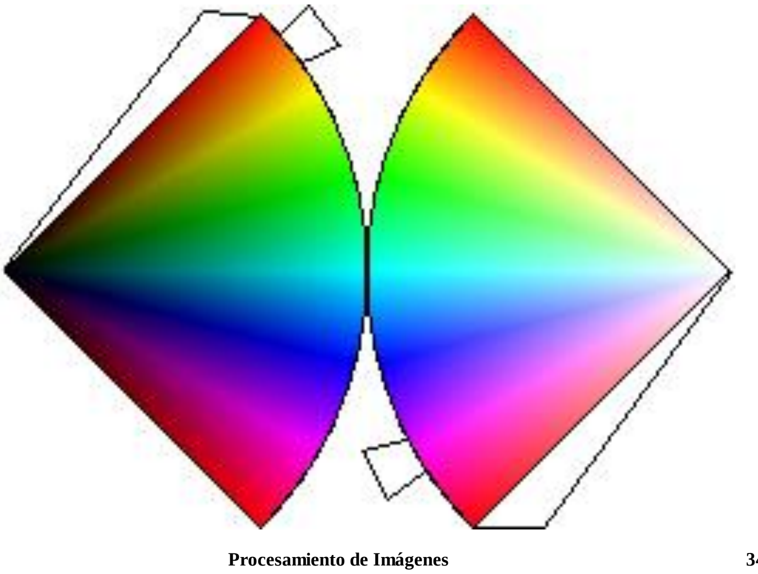
- Por el contrario, el **espacio HLS** se suele representar como un **doble cono**.



Variación de H y S, con L=0,5

5.2.4. Modelos HLS y HSV.

Desarrollo del doble cono HLS



5.2.4. Modelos HLS y HSV.

- **Conversión RGB a HSV y HLS**:
- Sea MAX:= max{R, G, B} y MIN:= min{R, G, B}
- El valor de **H** se calcula según el "cuadrante" en RGB respecto a la línea de grises:
- $$H := \begin{cases} (G-B)*60/(MAX-MIN) & \text{si } R = MAX \\ (B-R)*60/(MAX-MIN)+120 & \text{si } G = MAX \\ (R-G)*60/(MAX-MIN)+240 & \text{si } B = MAX \end{cases}$$
- **En HSV**:
  - S:= (MAX-MIN)/MAX
  - V:= MAX
- **En HLS**:
  - S:= MAX-MIN
  - L:= (MAX+MIN)/2
- **Ojo**: si R=G=B (color gris), el H no está definido. Es más, conforme disminuye la saturación, el cálculo de H es más inestable.

5.2.4. Modelos HLS y HSV.

- **Ejemplo 1**.

Canales S y V de HSV

Canal H

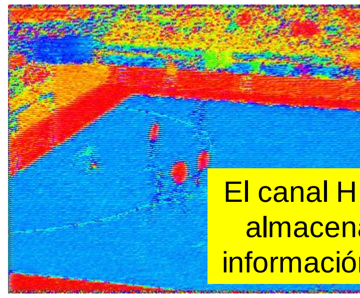
Canales de S y L de HLS

5.2.4. Modelos HLS y HSV.

- **Ejemplo 2**.

Imagen de entrada

Sólo con el canal H



El canal H es el que almacena mayor información de color

Pero cuidado, puede ser engañoso si la saturación es baja

5.2.5. Otros modelos de color.

Otros modelos de color:

- Existen **otros muchos modelos de color**, algunos de ellos creados para aplicaciones específicas: YIQ, YUV, YCrCb, YCC, CIE Lab, CIE LUV, y otros muchos.
- Por ejemplo, **YIQ** y **YUV** se crearon para transmisión de vídeo (TV analógica); **YIQ** en el estándar americano (**NTSC**), y **YUV** en el europeo (**PAL**).
- **YUV** se puede ver como una rotación del **YIQ** en 33º.
- La mayoría de estos espacios se basan en **separar** por un lado el canal de **luminosidad** o **brillo**, **Y**, y por otro dos canales de **color** o **crominancia**.
- El ojo humano es mucho más sensible al brillo que al color, por lo que **Y** es más prioritario (necesita más resolución).

5.2.5. Otros modelos de color.

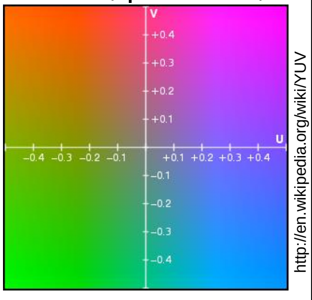
- **Modelo YUV**. Se define cómo una transf. lineal del RGB.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0{,}30 & 0{,}59 & 0{,}11 \\ -0{,}15 & -0{,}29 & 0{,}44 \\ 0{,}62 & -0{,}52 & -0{,}10 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

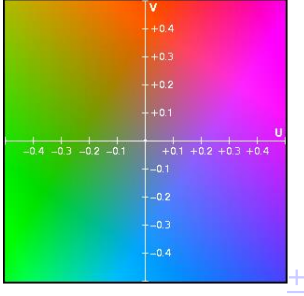
- El modelo YUV se usa también en compresión JPEG y en MPEG.
- **Modelo YIQ**. Parecido a YUV, pero con coeficientes distintos.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0{,}30 & 0{,}59 & 0{,}11 \\ 0{,}60 & -0{,}27 & -0{,}32 \\ 0{,}21 & -0{,}52 & 0{,}31 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Plano UV, para Y=0,5



Plano IQ, para Y=0,5

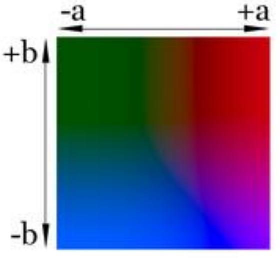


http://en.wikipedia.org/wiki/YUV

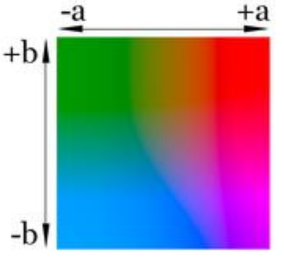
5.2.5. Otros modelos de color.

- **Modelo CIE Lab**. Es otro espacio de color definido por la CIE (en 1976), intentando linealizar las diferencias perceptibles por el ojo humano.
- Se define mediante transformaciones no lineales a partir del CIE XYZ. Separa luminosidad (**L**) y color (**a**: rojo/verde, **b**: azul/amarillo).

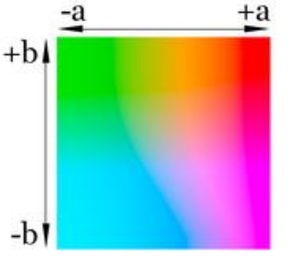
Plano ab, para L=0,25



Plano ab, L=0,5



Plano ab, L=0,75

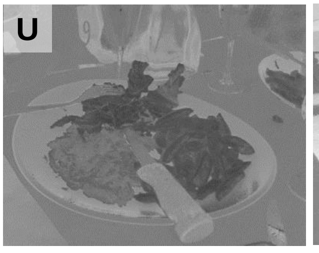


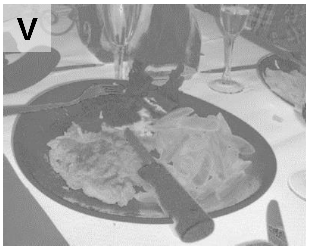
http://en.wikipedia.org/wiki/Lab_color_space

5.2.5. Otros modelos de color.

- **Ejemplo**.

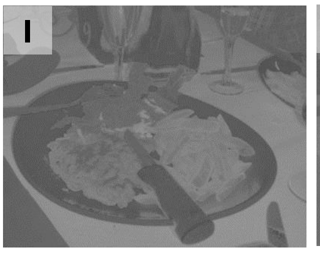
Canales U y V de YUV





Canal Y

Canales de I y Q de YIQ



5.2.6. Operaciones con color.

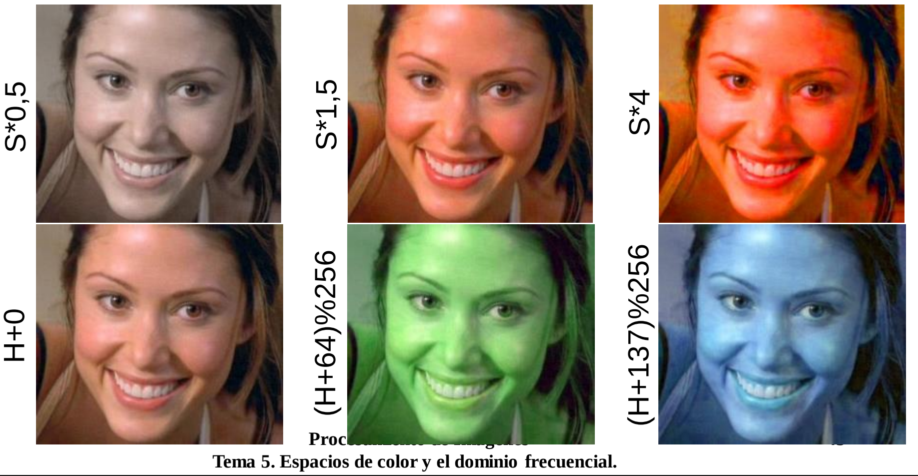
Operaciones globales en diferentes espacios

- En el espacio RGB, las **operaciones globales** (suma, producto) con **valores distintos** en cada canal tienen el sentido de dar a la imagen cierto tono de color.
- ¿Qué ocurre en los otros espacios?
- **Espacios HSV o HLS**: las operaciones tienen un significado diferente en cada canal.
  - **H**: cambiar los tonos de color de la imagen.
  - **S**: cambiar la saturación, color más brillante o apagado (gris).
  - **V**, **L**: cambiar la luminosidad, manteniendo el color.
- **Espacios Y*XX***: ajuste separado de la luminosidad (**Y**) y el color (**XX**).
  - **Y**: cambiar la luminosidad, manteniendo el color.
  - **XX**: cambiar el tono de color, de manera progresiva. Se pueden usar estos espacios para hacer "balance de blancos".

5.2.6. Operaciones con color.

- **Ajuste del matiz, saturación y luminosidad (con HLS):**

1) Convertir la imagen RGB al espacio HLS. En OpenCV %180
2) Multiplicar los canales S y L por un valor dado.
3) Sumar (módulo 256) al canal H un valor (cambio de matiz).
4) Transformar la imagen de HLS al espacio RGB.



5.2.6. Operaciones con color.

- **Balance de blancos (usando YUV):**

1) Convertir la imagen RGB al espacio YUV.
2) Calcular la media de los canales U y V.
3) Modificar U y V (suma) de manera que la media sea 128.
4) Transformar la imagen YUV al espacio RGB.

Imagen de entrada 1

Imagen de entrada 2

5.2.6. Operaciones con color.

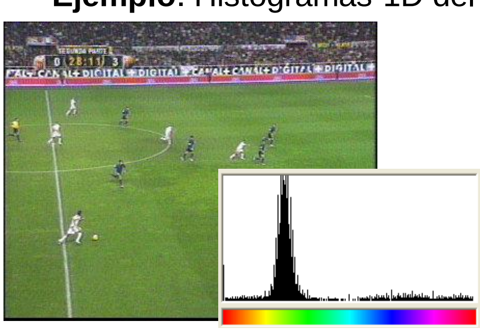
Im 1. Medias: U= 161, V= 102 **Im 2**. Medias: U= 88, V= 166

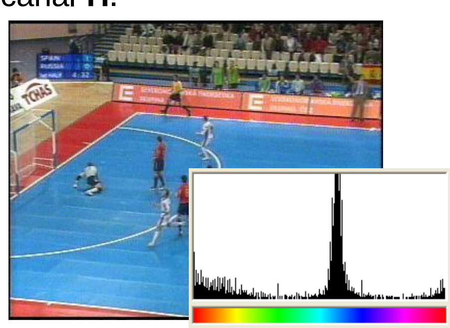
(Y, U-33, V+26) (Y, U+40, V-38)

5.2.6. Operaciones con color.

Histogramas de color

- Los histogramas pueden ser útiles para analizar los colores que aparecen con **más frecuencia** en una imagen.
- Se pueden usar histogramas 1D, 2D o 3D.
- Un **histograma 1D** puede suponer pérdida de información.
- El significado depende del canal estudiado.
- **Ejemplo**. Histogramas 1D del canal **H**.

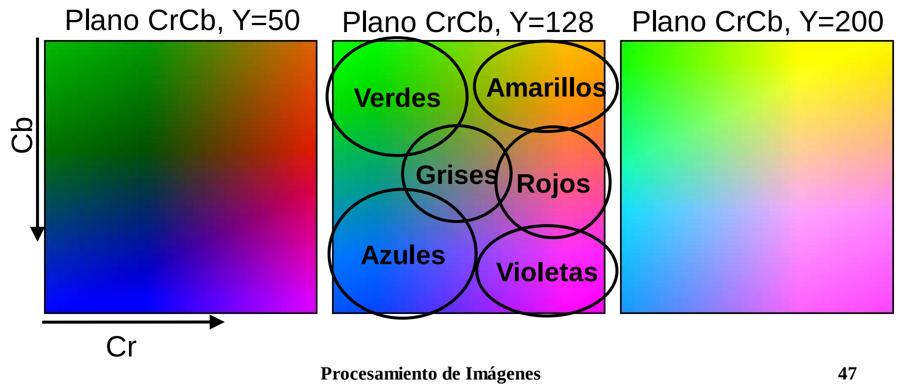




5.2.6. Operaciones con color.

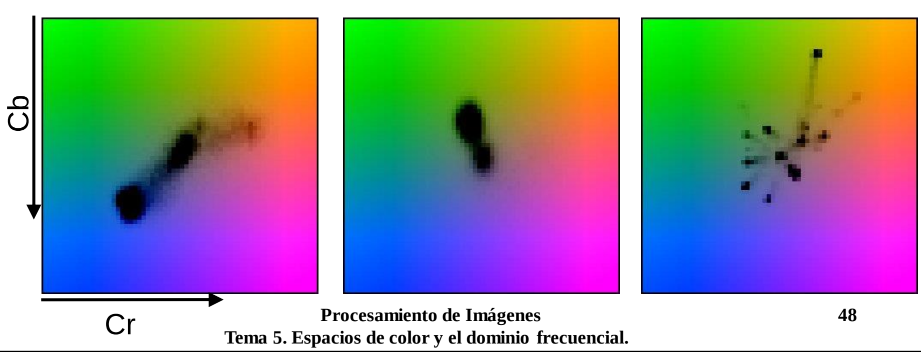
- En el caso de los **histogramas 2D**, puede ser interesante usar espacios estilo ***YXX***, y analizar los ***XX***.
- **Ejemplo**. Histograma 2D usando el espacio **YCrCb**. **Y**= intensidad, **Cr**= canal R normalizado, **Cb**= canal B normaliz.

  Aplicamos el histograma a los canales Cr y Cb.



5.2.6. Operaciones con color.

- **Ejemplo**. Histograma 2D de los canales (Cr, Cb).



5.2.6. Operaciones con color.

Rellenado de color

- El rellenado de una región se basa en una medida de diferencia entre colores.
- **Algoritmo**. **Rellenar una región** a partir de un píxel (x_0,y_0): Para todo píxel (x,y) adyacente a los rellenados, si la diferencia es menor que cierto umbral, rellenar también (x,y).
- **Dos modos**:
  - Diferencia de (x, y) respecto al punto inicial (x_0, y_0).
  - Diferencia de (x, y) respecto al píxel adyacente más cercano.
- Se pueden usar diferentes medidas de distancia y en distintos espacios de color.
- **Ejemplo**. Diferencia entre dos píxeles (r_1,g_1,b_1) y (r_2,g_2,b_2).
- **D1**:= $|r_1-r_2| + |g_1-g_2| + |b_1-b_2|$; **D2**:= $\max(|r_1-r_2|, |g_1-g_2|, |b_1-b_2|)$
- **D3**:= $\mathrm{sqrt}((r_1-r_2)^2 + (g_1-g_2)^2 + (b_1-b_2)^2)$

5.2.6. Operaciones con color.

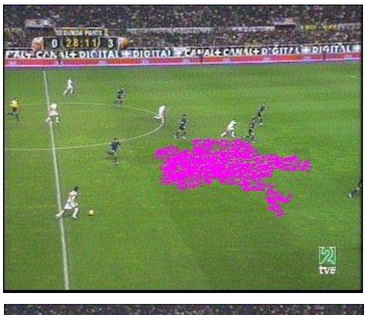
- **Ejemplo**. Rellenado de color.

  Modo 1: rango fijo.

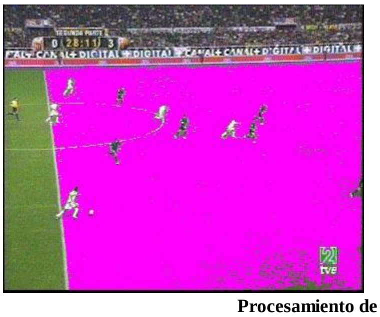
  Se rellena a partir del píxel central, con cierto umbral inferior y superior.

  Usando el espacio RGB y la distancia D2.

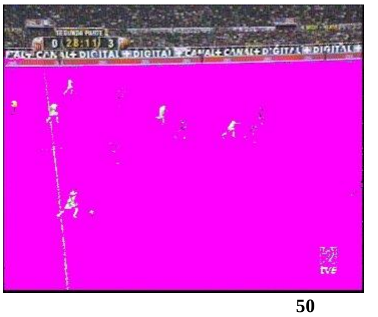
Rango = (-20, +20)



Rango = (-50, +50)



Rango = (-90, +90)



5.2.6. Operaciones con color.

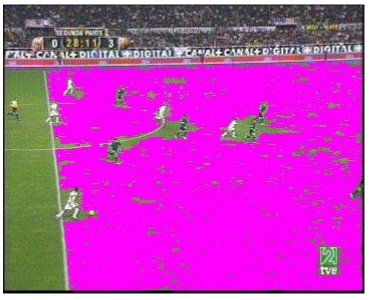
- **Ejemplo**. Rellenado de color.

  Modo 2: rango "flotante".

  Se rellena a partir del píxel central, con cierto umbral inferior y superior.

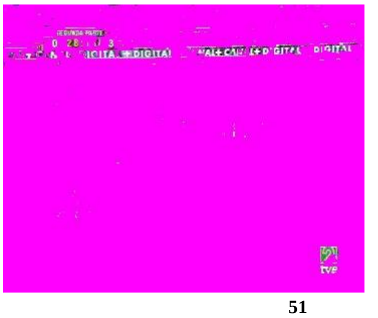
  También usando RGB y D2.

Rango = (-16, +16)



Rango = (-26, +26)

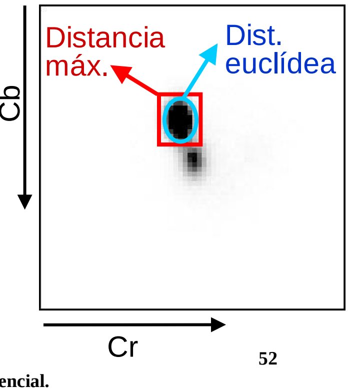
Rango = (-50, +50)



5.2.6. Operaciones con color.

- En general:
  - El modo fijo es más **sensible** a la elección del **punto inicial**.
  - El modo flotante es problemático si hay **gradientes suaves**.
  - Los **umbrales** en el modo flotante deben ser menores.
- Encontrar el modo de rellenado, el espacio de color, la medida de distancia y los umbrales adecuados no es sencillo.
- Menos aun si hay que hacerlo automáticamente...
- Los **histogramas** pueden ser útiles para decidir los umbrales y la medida de distancia.

Histograma de (Cr,Cb)



5.2. Modelos y espacios de color.

Conclusiones:

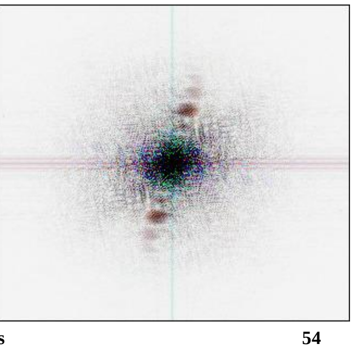
- El color es un fenómeno ligado a la **percepción humana**.
- Existen muchos **modelos de color**, algunos creados con fines específicos.
- Aunque **externamente** (entrada/salida) trabajemos normalmente con el **modelo RGB**, para realizar ciertas operaciones puede ser adecuado usar **otros modelos** de color:
  - Transformar de RGB al otro modelo.
  - Operar en el otro modelo.
  - Transformar el resultado en el otro modelo a RGB.
- Estudiar, decidir y utilizar el modelo más adecuado a cada aplicación.

5.3. El dominio frecuencial.

- Hasta ahora siempre hemos visto las imágenes en el **dominio espacial**: las distancias entre píxeles se traducen en distancias dentro del espacio.
- Pero también se pueden visualizar y manipular en el **dominio frecuencial**: las distancias entre píxeles se traducen en diferencias dentro de la frecuencia.

Imagen A en **dominio espacial**

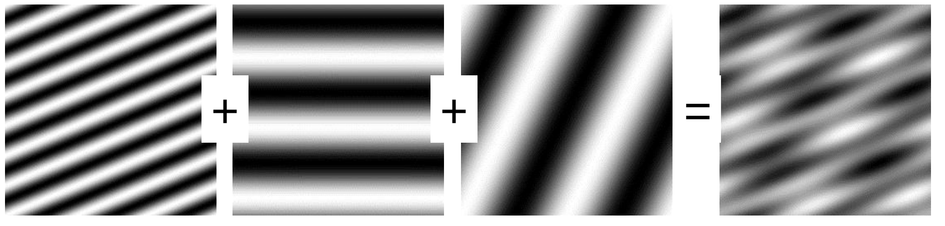
Imagen A en **dominio frecuencial**



5.3. El dominio frecuencial.

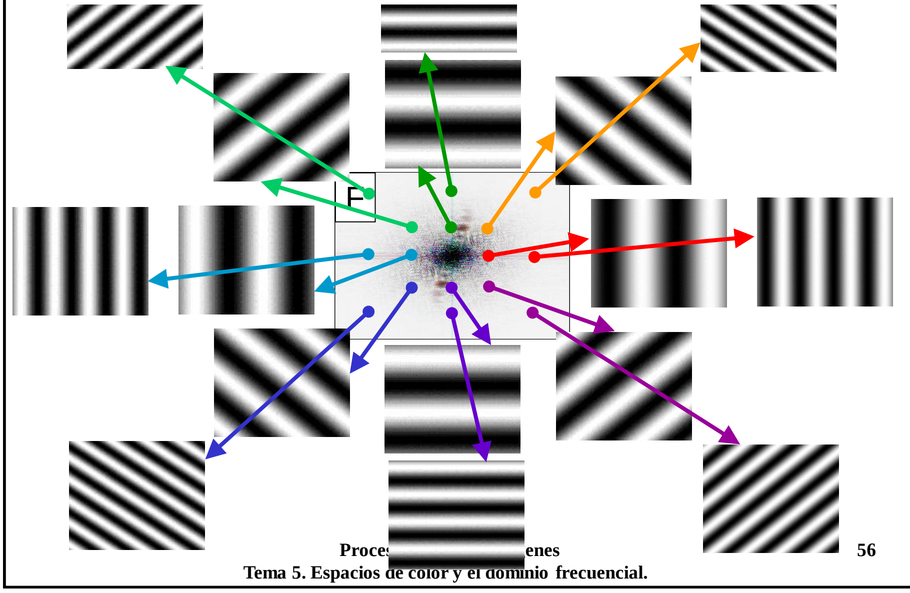
- Ambos dominios, espacial y frecuencial, son **duales**:
  - Los dos contienen la misma **cantidad de información**.
  - La **transformación** de uno a otro es **unívoca**.
- Pero, ¿cuál es el **significado del dominio frecuencial**?
- Recordar el **principio de Fourier**: cualquier señal se puede expresar como una suma de señales sinusoidales.
- También en imágenes, pero con señales sinusoidales 2D.
- **Ejemplo**. Imagen como suma de componentes frecuenciales.

Señales sinusoidales (componentes frecuenciales) Imagen resultante



5.3. El dominio frecuencial.

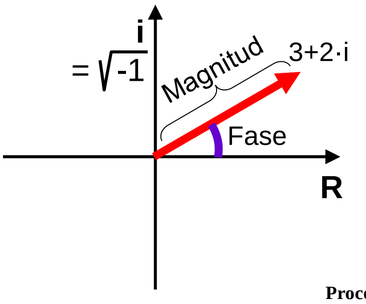
- Si **F** es una imagen en el dominio frecuencial, el valor del píxel **F(x,y)** indica cómo es el c. frecuencial asociado a **(x,y)**.



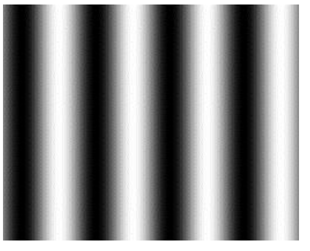
5.3. El dominio frecuencial.

- En concreto, el valor de cada píxel indica la **magnitud** y la **fase** del componente frecuencial correspondiente.
  - **Magnitud**: mayor o menor fuerza (peso) del componente.
  - **Fase**: ángulo en el punto 0.
- Por lo tanto, cada píxel (x, y) se puede expresar como un vector 2D. → Se usan **números complejos**: parte **real** y parte **imaginaria**.

Píxel F(x, y)



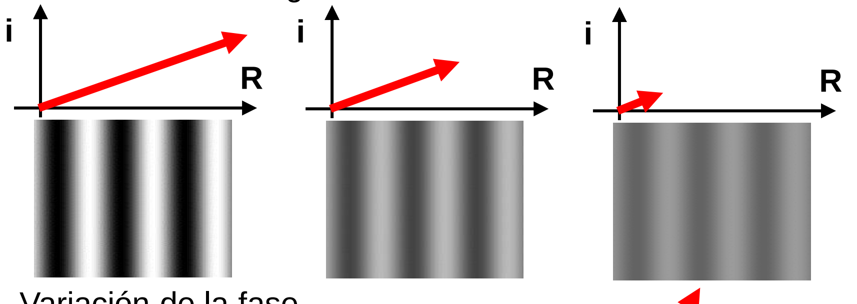
Componente frec. asociado a F(x, y)



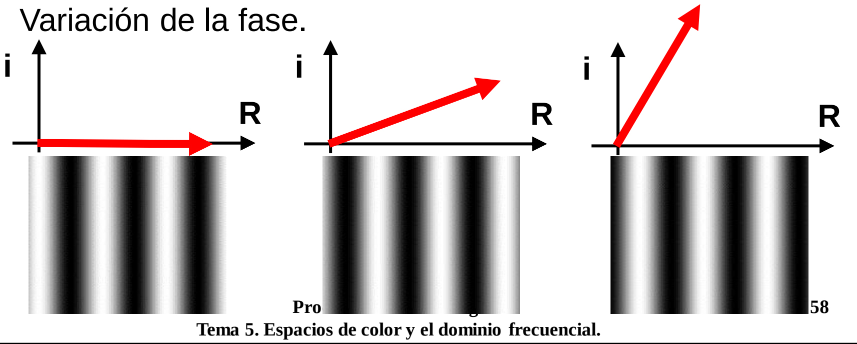
Recordar, una imagen es la suma de muchos de estos componentes

5.3. El dominio frecuencial.

- Variación de la magnitud.



- Variación de la fase.



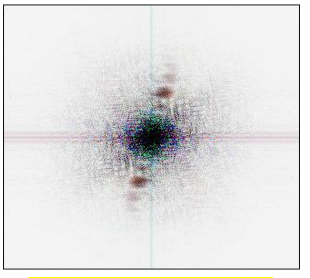
5.3. El dominio frecuencial.

- En definitiva, la imagen se descompone como una **suma de muchos componentes frecuenciales**.
  - La posición (x, y) del píxel, indica la frecuencia en X y en Y.
  - El valor del píxel indica el peso (magnitud) y la fase del comp.

Imagen A en el **dominio espacial**

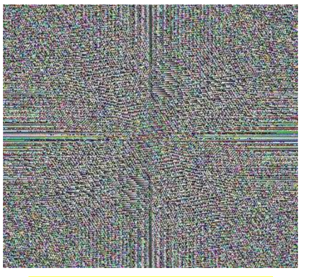
Imagen A en el **dominio frecuencial**

Magnitud



Negro = mayor magnitud

Fase



Negro = 0º
Blanco = 359º

5.3.1. La transformada de Fourier.

- El paso de una imagen desde el dominio espacial al dominio frecuencial es la llamada **transformada de Fourier**.
- En nuestro caso, usamos la **Tr. Discreta de Fourier** (**DFT**).
- **Fórmula**: **DFT**. Sea **A** una imagen de tamaño WxH. La DFT es otra imagen (compleja, de WxH) **F**, dada por:

$$F(a, b) := \sum_{x=0..W-1} \sum_{y=0..H-1} A(x, y)\cdot e^{(-2\pi i\cdot x\cdot a/W)}\cdot e^{(-2\pi i\cdot y\cdot b/H)}$$

Denotamos:
F:= DFT(A)

- Recordatorio: $e^{ik} = \cos k + i\cdot \text{sen } k$; $i = \sqrt{-1}$
- La transformación se puede **invertir**: dada F calcular A.
- **Fórmula**: **Tr. Inversa de Fourier (IDFT)**. Sea **F** una imagen compleja en el d. frec., la imagen **A** en el dom. espacial es:

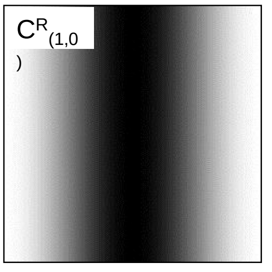
$$A(x, y) := 1/WH \sum_{a=0..W-1} \sum_{b=0..H-1} F(a, b)\cdot e^{(2\pi i\cdot x\cdot a/W)}\cdot e^{(2\pi i\cdot y\cdot b/H)}$$

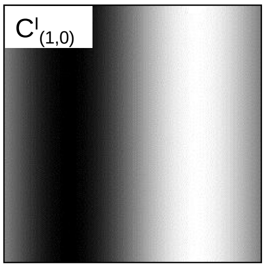
A:= IDFT(F)

- Ver que: IDFT(DFT(A)) = A, DFT(IDFT(F)) = F.

5.3.1. La transformada de Fourier.

- **Significado intuitivo de la DFT**.
  - El píxel F(0,0) contiene la suma de todos los píxeles. No tiene parte imaginaria.
  - Píxel F(1,0): la parte real sería como una convolución de la imagen, con una imagen con un coseno en X, de 1 ciclo. La parte imaginaria sería un seno.

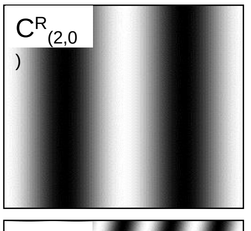


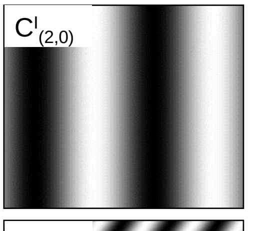


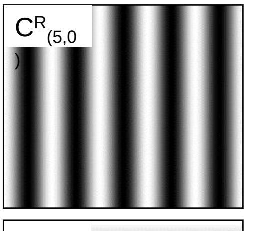
$$F(1,0) = \sum_{\forall x,y} A(x,y) \cdot C^R_{(1,0)}(x,y) + i \cdot \sum_{\forall x,y} A(x,y) \cdot C^I_{(1,0)}(x,y)$$

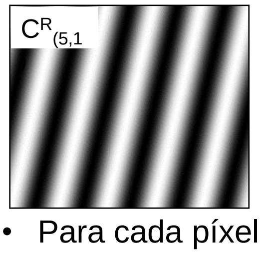
5.3.1. La transformada de Fourier.

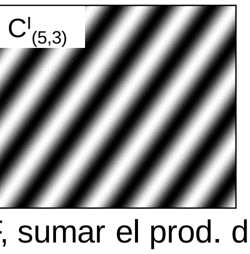
  - Y así para todos los píxeles.

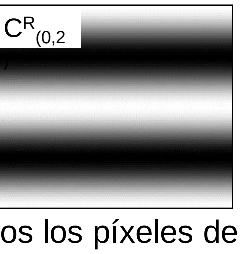








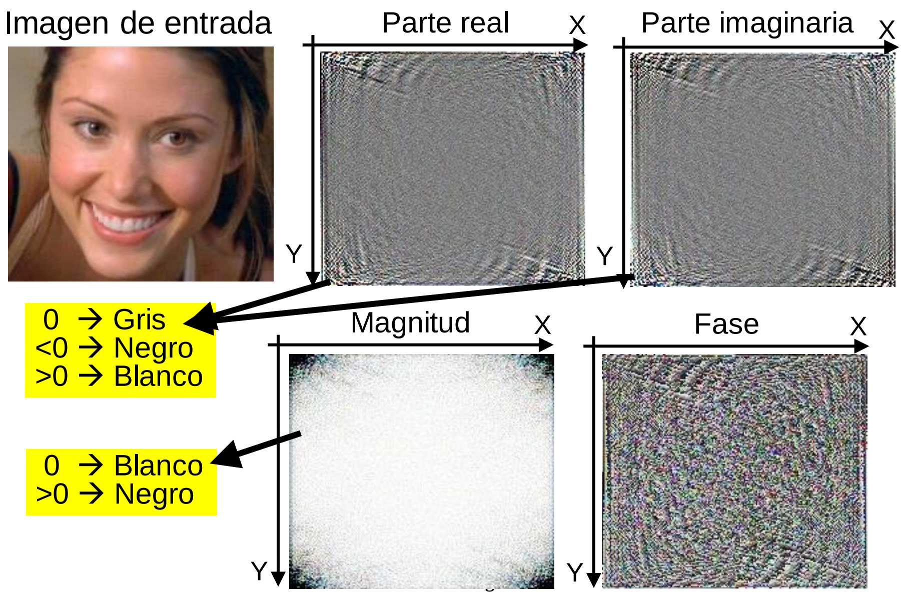




- Para cada píxel de F, sumar el prod. de todos los píxeles de A y C. Esto requiere un O($\mathbf{n^2}$), con **n** el nº de píxeles.
- La **Transf. Rápida de Fourier** (**FFT**), es una optimización del cálculo para obtener la DFT en O(**n**·**log n**).

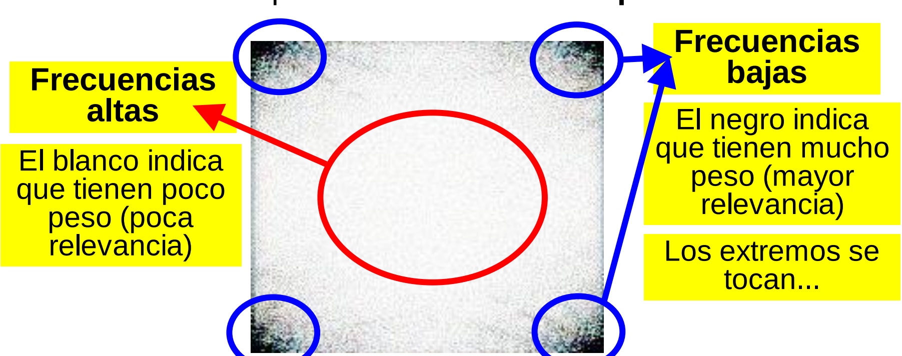
5.3.1. La transformada de Fourier.

- **Ejemplo**.



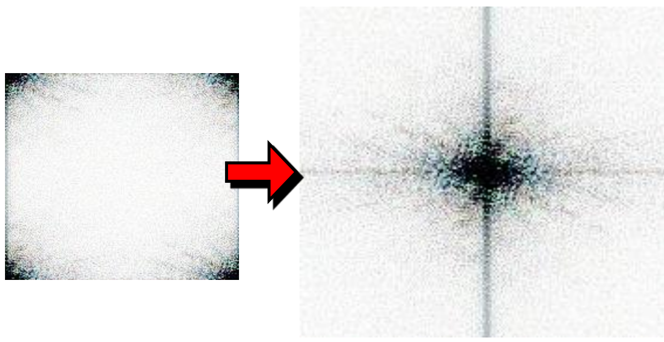
5.3.1. La transformada de Fourier.

- ¿Cómo **interpretar** la transformada de Fourier?
- Normalmente se visualiza la **magnitud de la DFT**.
- Las **esquinas** representan las **frecuencias bajas**. → Características de la imagen que varían **lentamente**.
- La zona **interior** son las **frecuencias altas**. → Características que varían con mucha **rapidez**.



5.3.1. La transformada de Fourier.

- Los extremos se tocan...
- La DFT se suele representar **centrada**: desplazar la imagen para colocar el píxel (0,0) en el centro de la imagen.



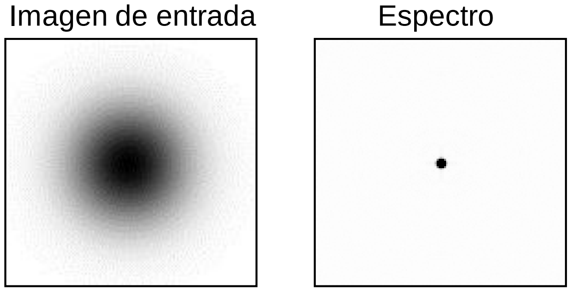
Esta representación (magnitud de la DFT, centrada) se suele denominar el **espectro de la imagen**

- Por la forma de calcularla, la DFT es **semi-simétrica**: el cuadrante 1 es simétrico al 3, y el 2 al 4.
- En concreto, el F(a,b) es conjugado del F(W-a, H-b).

5.3.2. Propiedades del dominio frecuencial.

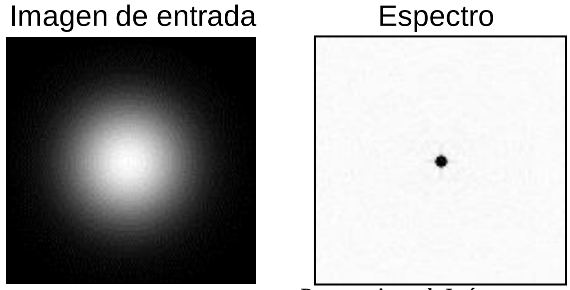
- **Ejemplos 1**.

Imagen de entrada | Espectro



La transformada de una **gaussiana** es también una gaussiana

Imagen de entrada | Espectro



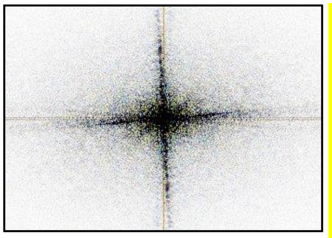
La transformada de la **inversa** de una imagen no cambia en la magnitud (se invierte el ángulo)

5.3.2. Propiedades del dominio frecuencial.

- **Ejemplos 2**.

Imagen de entrada

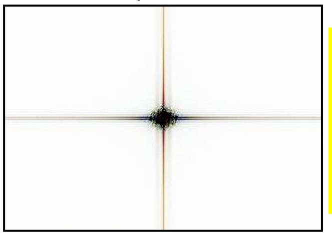
Espectro



La **acumulación** de valores altos en una dirección, indica características destacadas en la imagen en cierto ángulo y frecuencia

Imagen suavizada

Espectro



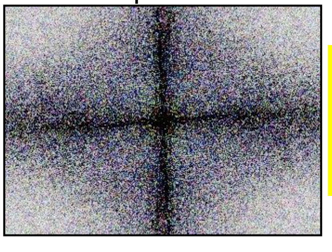
La DFT de una imagen **suavizada** elimina (pone a 0) las frec. altas y respeta las bajas

5.3.2. Propiedades del dominio frecuencial.

- **Ejemplos 3**.

Imagen perfilada

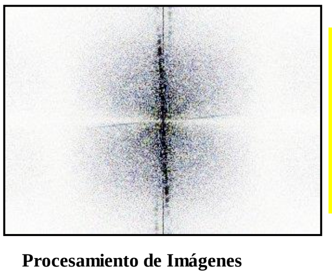
Espectro



Por el contrario, el **perfilado** aumenta las frecuencias altas, respetando las bajas

Imagen gradiente Y

Espectro



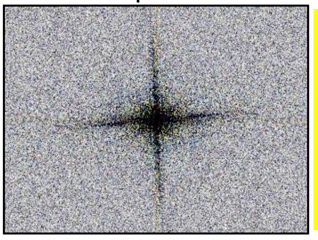
La transformada de la **derivada** extrae las frecuencias en determinada dirección

5.3.2. Propiedades del dominio frecuencial.

- **Ejemplos 4**.

Imagen con ruido

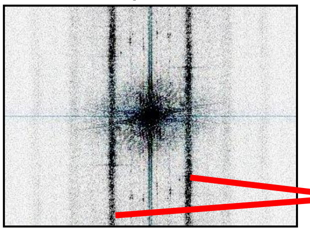
Espectro



El ruido desestructurado (**ruido blanco**) afecta por igual a todas las frecuencias, las aumenta

Imagen de entrada

Espectro



Pero normalmente el ruido afecta sólo a ciertas **frecuencias** características

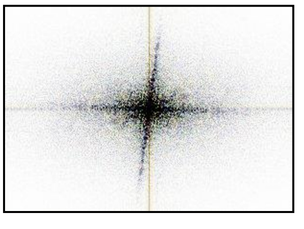
Bandas de ruido

5.3.2. Propiedades del dominio frecuencial.

- **Ejemplos 5**.

Imagen rotada

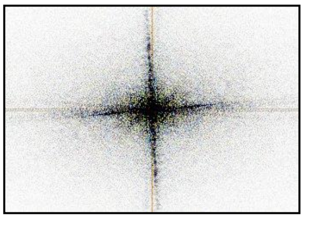
Espectro



El espectro de una imagen **rotada** aparece rotado en la misma cantidad

Imagen trasladada

Espectro



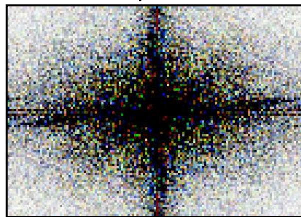
El **desplazamiento** no afecta a la DFT, siempre que aparezcan visibles los mismos trozos (sólo cambia la fase)

5.3.2. Propiedades del dominio frecuencial.

- **Ejemplos 6**.

Imagen reducida 1/4

Espectro



El espectro de una **imagen reducida** en cierta cantidad K consiste en coger del espectro orig. la **zona central** de tamaño K. Las frecuencias altas se pierden

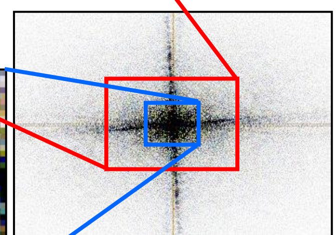
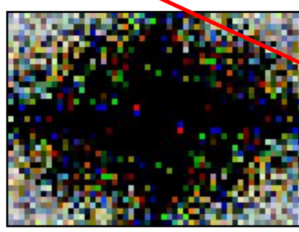


Imagen reducida 1/6

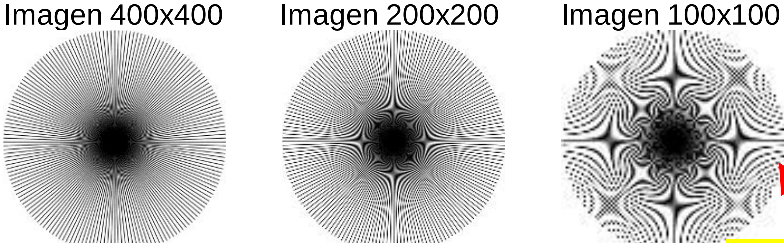
Espectro

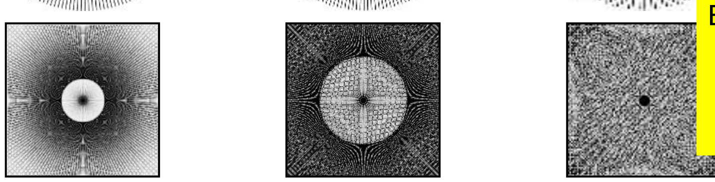


5.3.2. Propiedades del dominio frecuencial.

- Esto está relacionado con el **teorema de muestreo**: la resolución de la imagen debe ser por lo menos el doble de la frecuencia más alta de interés (el detalle más pequeño).

Imagen 400x400 | Imagen 200x200 | Imagen 100x100





Este tipo de aliasing se denomina **moaré** (o muaré)

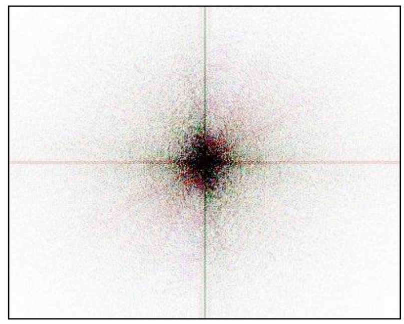
- Al reducir resolución, se recortan las frecuencias más altas.

5.3.2. Propiedades del dominio frecuencial.

- En todos los ejemplos anteriores, se aplica la **DFT en los tres canales** (R, G, B) de forma **independiente**.
- En la salida, cada canal es la DFT de ese canal. El resultado debería ser en color. ¿Por qué aparecen siempre en gris?

Imagen de entrada

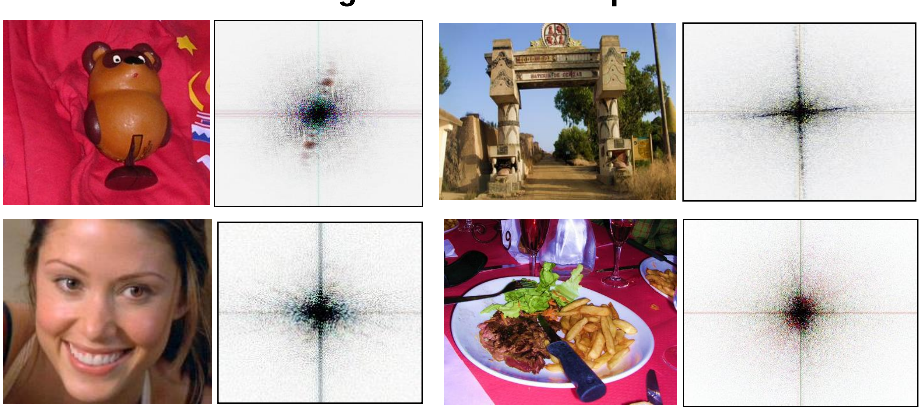
Espectro



- **Resultado**: normalmente, todos los canales de color aportan la misma o parecida información frecuencial.

5.3.2. Propiedades del dominio frecuencial.

- Otra idea interesante es que en casi todas las imágenes los **valores altos** de **magnitud** están en la **parte central**.



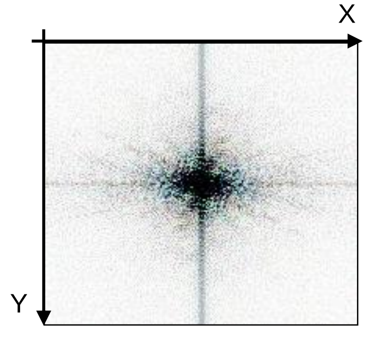
- ¿Qué significa esto?
- ¿Por qué ocurre así?
- ¿Para qué puede servir?

5.3.2. Propiedades del dominio frecuencial.

¿Qué significa?

- Recordar, la **parte central** del espectro indica los **comp. frecuenciales bajos**, los que varían lentamente.
- Este fenómeno nos indica que normalmente los componentes **bajos** son los **importantes**, mientras que los **altos** son **irrelevantes**, su contribución es muy pequeña (magnitud reducida).
- También son relevantes los componentes altos de X para frecuencia 0 de Y, y viceversa.

¿A qué es debido?



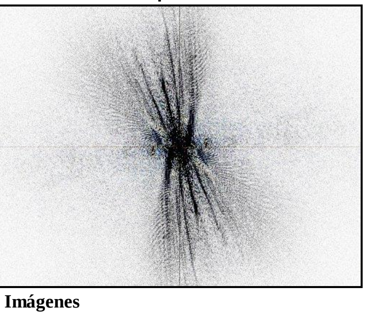
5.3.2. Propiedades del dominio frecuencial.

¿Por qué ocurre?

- Recordar la relación de **vecindad entre píxeles**: en una imagen "natural" se espera que dos píxeles próximos tengan valores parecidos.
- Las frec. altas significan variaciones **rápidas** o **abruptas** en las imágenes. Pero estas son menos comunes. Lo normal es encontrar variaciones **suaves** y zonas **uniformes**.

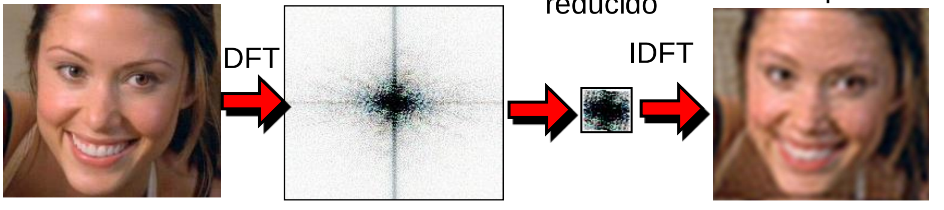
Imagen de entrada

Espectro



5.3.3. Aplicaciones de la DFT.

- Este resultado tiene una aplicación directa en **compresión de imágenes**.
  - **Compresión**: calcular la DFT y eliminar los componentes frecuenciales con magnitud baja.
  - **Descompresión**: rellenar los componentes eliminados con valor 0 y calcular la DFT inversa.



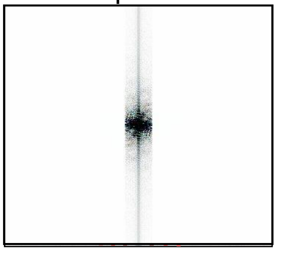
- Esta es la base de la técnica de **compresión JPEG**, aunque como vimos se aplica otra transformación (DCT) y algunas fases adicionales.

5.3.3. Aplicaciones de la DFT.

- **Ejemplo**. Compresión con simple eliminación de frec. altas.

Imagen original

Espectro

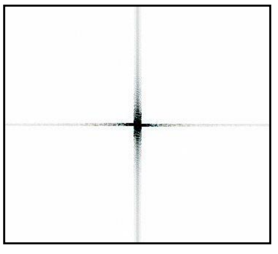


Eliminación del 70% | Eliminación del 80% | Eliminación del 90%

5.3.3. Aplicaciones de la DFT.

- También es importante decidir **qué parte** es la que se elimina del espectro.

Eliminación del 96% Eliminación del 99%



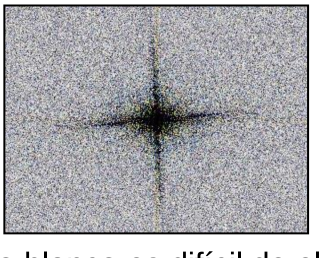
De todos los pasos de compresión de JPEG, el más importante es este

Realmente no se debería eliminar una "región" fija del espectro, sino coger los componentes frecuenciales de mayor magnitud

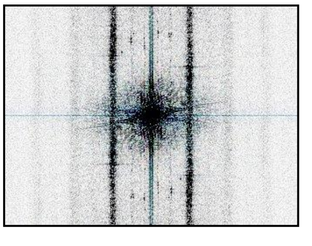
5.3.3. Aplicaciones de la DFT.

- Otra aplicación interesante del dominio frecuencial es el **análisis** del **tipo y nivel de ruido** y su **eliminación**.
- Podemos distinguir dos tipos de ruido:
  - **Ruido blanco**. Afecta por igual a todas las frecuencias.
  - **Ruido repetitivo**. Afecta a ciertas frecuencias concretas.

Ejemplo, espectro de imagen con **ruido blanco**



Ejemplo, espectro de imagen con **ruido repetitivo**



- El ruido blanco es difícil de eliminar sin degradar la calidad de la imagen.
- Eliminación del ruido repetitivo: eliminar (poner a 0) las bandas de ruido.

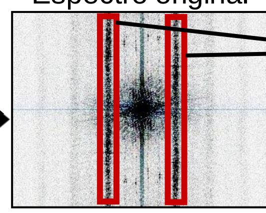
5.3.3. Aplicaciones de la DFT.

- **Proceso de eliminación del ruido repetitivo**:
  - **Calcular la DFT** de la imagen.
  - Localizar las **bandas de ruido** en la DFT.
  - **Poner a 0** las bandas de ruido.
  - Calcular la **DFT inversa** de la imagen retocada.

Imagen original

DFT

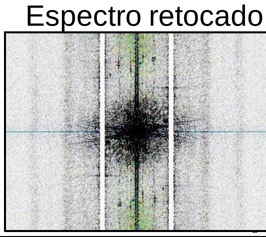
Espectro original



Bandas de ruido

¿Cómo localizarlas de forma automática?

Espectro retocado



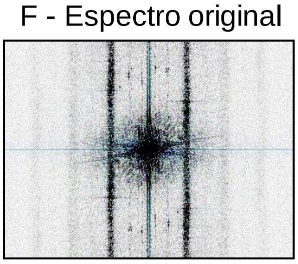
IDFT

Imagen restaurada

5.3.3. Aplicaciones de la DFT.

- La eliminación de las bandas de ruido es un **producto de dos imágenes** (píxel a píxel) **en el dominio frecuencial**.
  - **A**: imagen de entrada (dom. espacial)
  - **F**: imagen A en el dominio frecuencial (F = DFT(A))
  - **G**: máscara de eliminación de ruido.

F - Espectro original

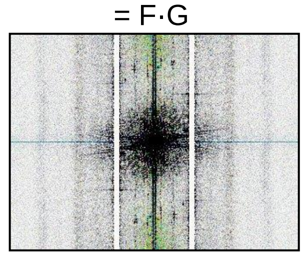


·

G - Máscara de ruido

=

F' - Espectro retocado = F·G



1 = Blanco
0 = Negro

5.3.3. Aplicaciones de la DFT.

- **Ejemplo 1**. Eliminación de ruido. Se ha aplicado un zoom, perfilado y ajuste del contraste, para apreciar mejor el ruido.

Imagen original

Imagen restaurada

- R= IDFT(F·G)
- También podemos usar la máscara opuesta (es decir, 1-G), para quedarnos sólo con el ruido.

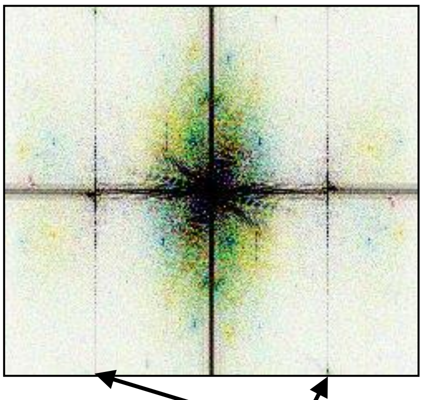
Imagen del ruido

5.3.3. Aplicaciones de la DFT.

- **Ejemplo 2**. Eliminación de ruido.

Imagen original

Espectro original



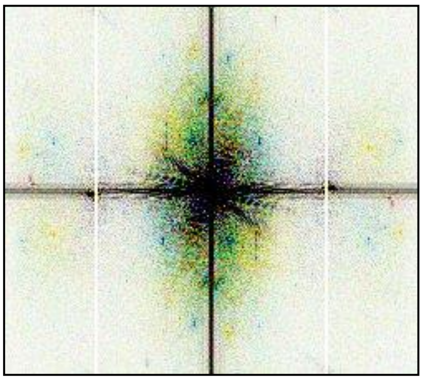
En este caso, el ruido está mucho más localizado

5.3.3. Aplicaciones de la DFT.

- **Ejemplo 2**. Eliminación de ruido.

Imagen reconstruida

Espectro retocado



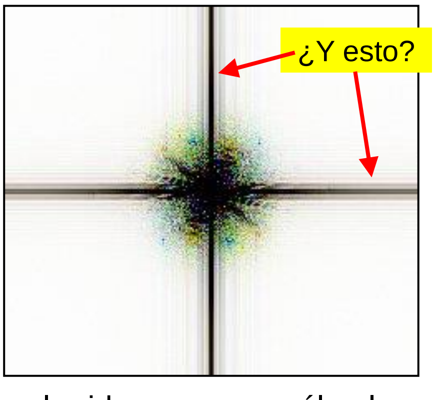
- Podemos intentar aplicar otro paso más de eliminación de ruido, aplicando un suavizado gaussiano.

5.3.3. Aplicaciones de la DFT.

- **Ejemplo 2**. Aplicando suavizado gaussiano.

Imagen reconstruida y suavizada

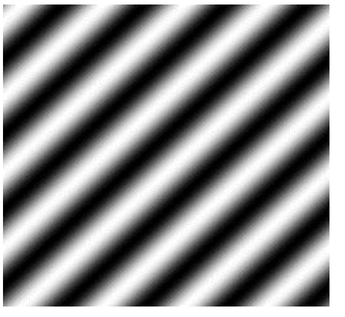
Espectro resultante



- Con esto hemos eliminado todo el ruido... pero no sólo el ruido, también las frecuencias altas. Hemos perdido información.

5.3.3. Aplicaciones de la DFT.

- La existencia de valores grandes para frecuencias altas en una dimensión y bajas en la otra, se debe a que la DFT supone que la imagen **se repite infinitamente** en el plano.

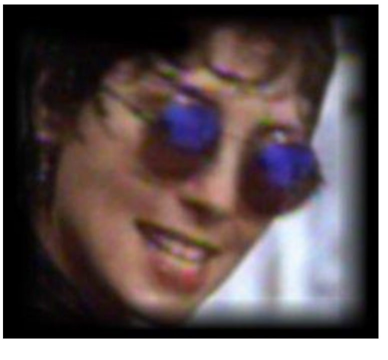


- Este fenómeno produce **discontinuidades abruptas** en los bordes, horizontales y verticales, que se reflejan en la DFT.
- Para evitarlo se pueden usar funciones de **enventanado**.

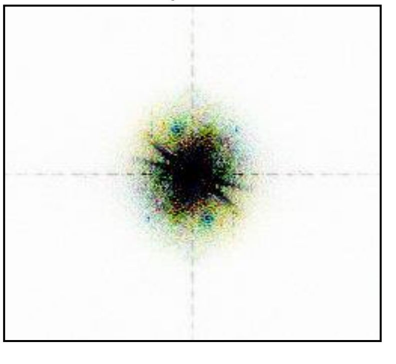
5.3.3. Aplicaciones de la DFT.

- **Enventanado** (*windowing*): modificar los bordes, para que la imagen se pueda *plegar* suavemente (sin discontinuidades).

Imagen suavizada y enventanada



Espectro



- El enventanado tiene sentido en **análisis** y **restauración** de imágenes. Podemos eliminar sin problemas todas las frecuencias altas en un eje (aunque sean bajas en el otro).

5.3.3. Aplicaciones de la DFT.

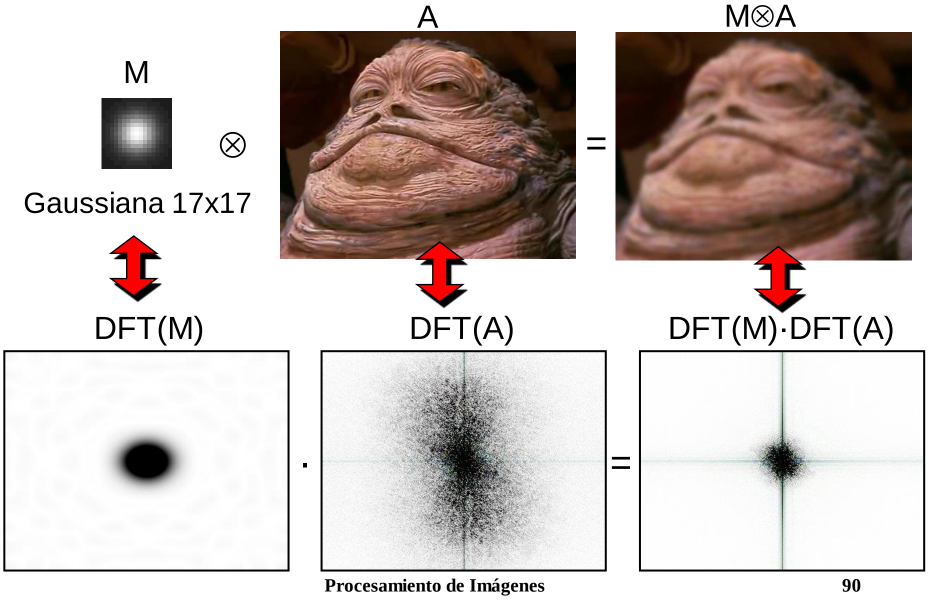
- Otra **propiedad fundamental** de la DFT es la **relación con las convoluciones**: la convolución de una imagen en el dominio espacial es equivalente a un producto en el dominio frecuencial.
- Sea **A** una imagen y **M** una máscara de convolución:

$$\mathrm{DFT}(M \otimes A) = \mathrm{DFT}(M) \cdot \mathrm{DFT}(A)$$

- Donde "⊗" es la operación de convolución, "·" es el producto de dos imágenes píxel a píxel (global), y DFT(M) es la DFT de la másc. de convolución (suponiéndola como una imagen del mismo tamaño que A).
- **Consecuencia**: en lugar de aplicar convoluciones en el dom. espacial, podemos aplicar productos en el frecuencial:
  $M \otimes A = \mathrm{IDFT}(\mathrm{DFT}(M \otimes A)) = \mathrm{IDFT}(\mathrm{DFT}(M) \cdot \mathrm{DFT}(A))$

5.3.3. Aplicaciones de la DFT.

- **Ejemplo**. Convolución con máscara gaussiana.



5.3.3. Aplicaciones de la DFT.

- Pero el verdadero interés de esta propiedad son las operaciones denominadas de **deconvolución**.
- **Deconvolución**: dada una imagen, **A**, a la cual se le ha aplicado una convolución, aplicarle otra convolución (**convolución inversa**) para obtener la imagen original.
- Las **deformaciones** por desenfoque, movimiento, perturbación atmosférica, etc., se pueden modelar como convoluciones, **M**, de formas conocidas.
- La imágenes resultantes salen borrosas debido a estas convoluciones: **B**= M⊗A
- **Objetivo**: encontrar la **convolución inversa**, **N**, para recuperar la imagen original: **A**= N⊗B
- Pero, ¿cómo podemos calcular **N** a partir de **M**?

5.3.3. Aplicaciones de la DFT.

- El problema se simplifica en el **dominio frecuencial**.
  - Sea **F** = DFT(B), la imagen que tenemos (la deformada),
  - Sea **H** = DFT(M), la deformación de tipo conocido,
  - Sea **G** = DFT(A), la imagen que queremos reconstruir.
- Tenemos:

B = M⊗A ⇒ DFT(B)= DFT(M⊗A) ⇒ DFT(B) = DFT(M)·DFT(A)

- Luego: F = H·G ⇒ G = F/H
- Siendo “/” la división, píxel a píxel.
- ¡Si sabemos la deformación producida por un desenfoque o un movimiento, podemos **reconstruir la imagen original** con IDFT(G)! Es decir:

  IDFT(DFT(B)/DFT(M))

- Esto es lo que se llama una **restauración mediante deconvolución**.

5.3.3. Aplicaciones de la DFT.

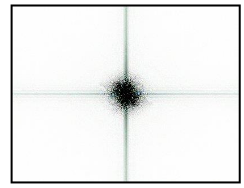
- **Proceso de deconvolución**.

1) Calcular la DFT de la imagen de entrada, F.
2) Averiguar la forma de la deformación, y calcular su DFT, H.
3) Calcular G= F/H.
4) Calcular la IDFT de G.

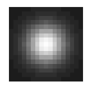
Ojo, esto es una división de números complejos

B

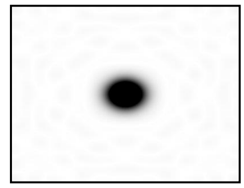
F= DFT(B)



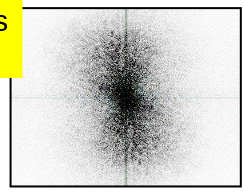
M



H = DFT (M)



G= F/H



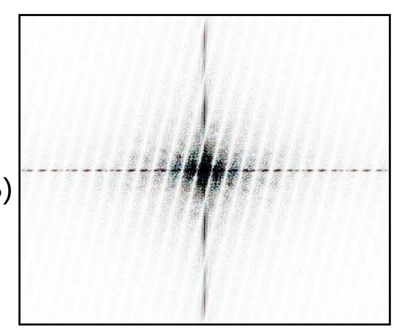
IDFT (G)

5.3.3. Aplicaciones de la DFT.

- **Ejemplo**. Una imagen sufre una deformación de tipo "eco". La imagen se repite desplazada 25 píxeles en X y 5 píxeles en Y. Restaurar la imagen original.

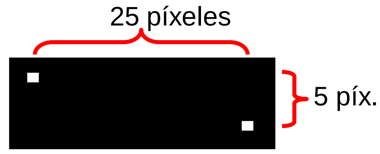
B

F = DFT(B)



- ¿Cuál es la máscara de convolución asociada a esta deformación?

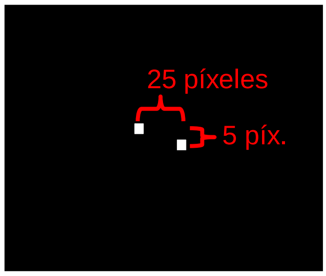
M



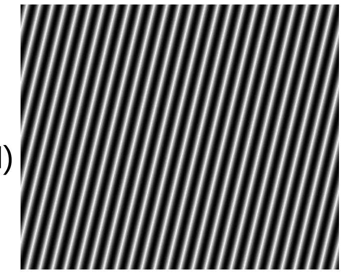
5.3.3. Aplicaciones de la DFT.

- Aplicamos la deconvolución, usando la máscara **M**.

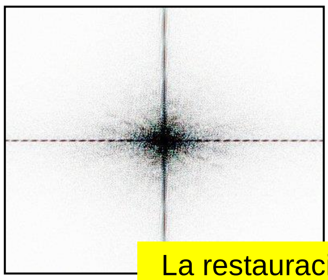
M



H = DFT(M)



G = F/H



A = IDFT (G)

La restauración es perfecta... porque la deformación es “artificial” y conocida

5.3.3. Aplicaciones de la DFT.

- Pero, en la práctica, este método tiene muchos **problemas**:
  - No se puede aplicar la división donde H(u,v) = 0.
  - Es más, si H(u,v) es pequeño (próximo a 0), la operación es muy inestable y sensible.
  - Si hay ruido en la imagen, sólo se potencia el ruido.
- Para solucionarlo se usan los **filtros de Wiener**.

- **Filtro de Wiener**: sea **F** la DFT de la imagen deformada y **H** la DFT de la deformación, la imagen reconstruida es:

$$G = H^{*}\cdot F/(|H|^2 + k)$$

es decir, $G(u,v) = H^{*}(u,v)\cdot F(u,v)/(|H(u,v)|^2 + k), \ \forall\ u, v$

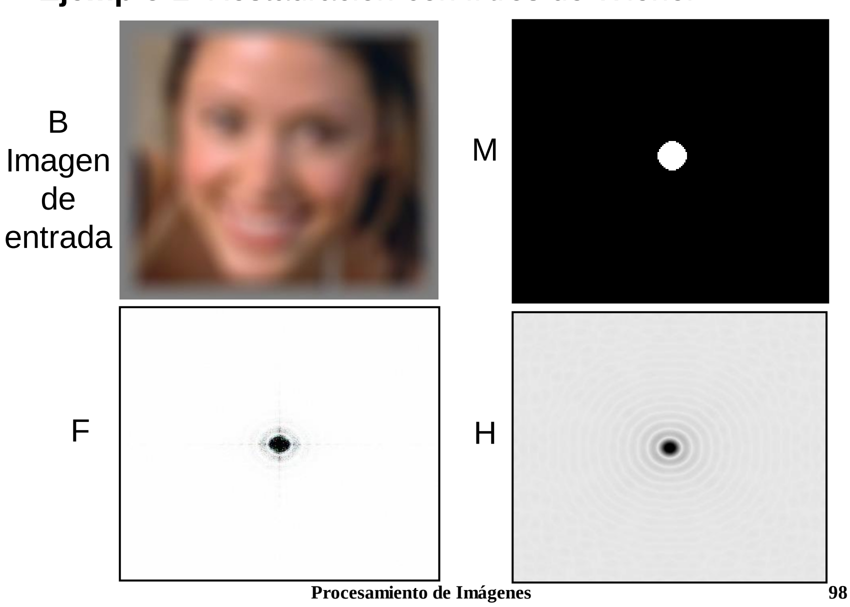
- Siendo **H*** el complejo conjugado de **H** (e.d., a + bi → a – bi), **|H|** el módulo del nº complejo ($|a + bi|^2 = a^2 + b^2$), y **k** una constante que se establece según el nivel de ruido.

5.3.3. Aplicaciones de la DFT.

- Observar que si **k**=0, tenemos el filtro de deconv.: G= F/H.
- **Cuestión clave**: encontrar la máscara de convolución asociada a la deformación que ha ocurrido.
- Por ejemplo, el **desenfoque** de una cámara se modela con una máscara de media con forma redonda.

5.3.3. Aplicaciones de la DFT.

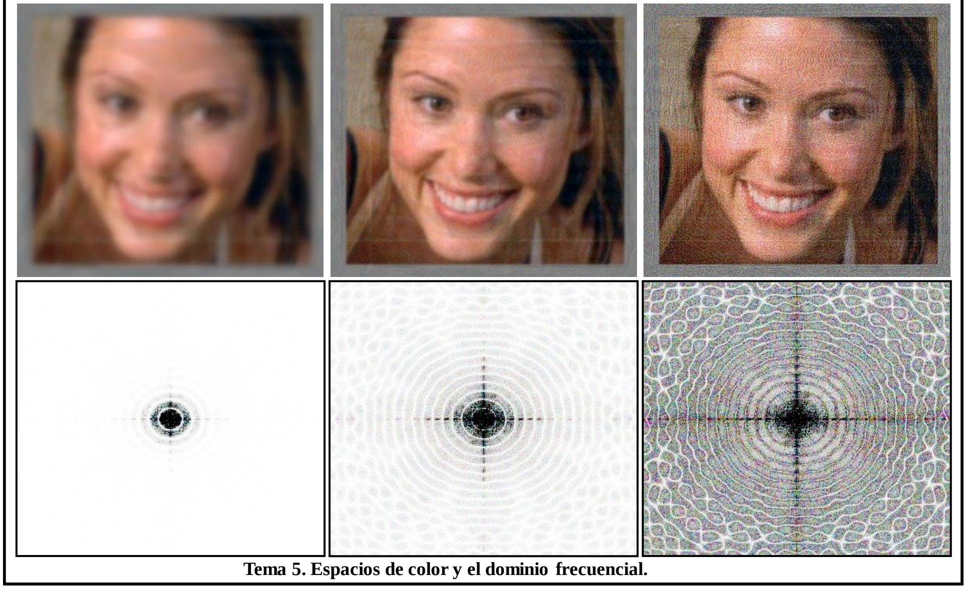
- **Ejemplo 1**. Restauración con filtros de Wiener.



5.3.3. Aplicaciones de la DFT.

- **Ejemplo 1**. Restauración con filtros de Wiener.

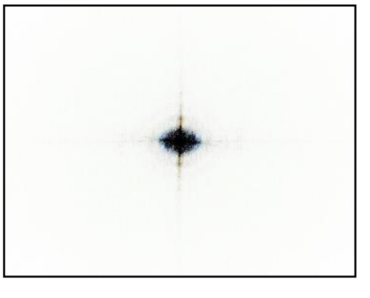
k = 0.0156 **k** = 0.000625 **k** = 0.000025



5.3.3. Aplicaciones de la DFT.

- En estos ejemplos va muy bien. Pero, ¿qué pasa cuando lo aplicamos en imágenes con deformaciones reales?
- En la práctica, conseguir una **buena restauración** es un proceso muy costoso de prueba y error.

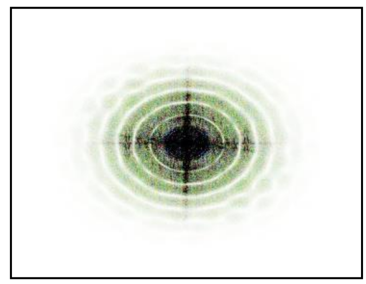
¿Qué pone en ese titular?



Tamaño: 552x424

5.3.3. Aplicaciones de la DFT.

- **Ejemplo 2**. Tras muchas pruebas, se encuentra que la deformación es un desenfoque de ~10 píxeles de radio.
- Aplicando un filtro de Wiener, se puede leer el titular, aunque la calidad de la imagen dista mucho de ser buena...



5.4. Otras transformaciones lineales.

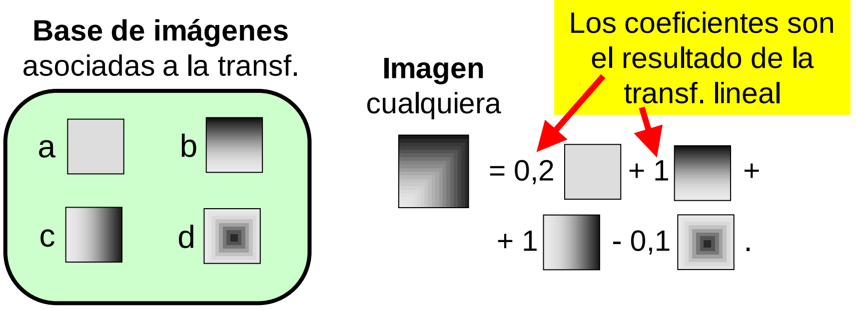
- En general, una **transformación lineal**, **T**, es una operación donde el valor de cada píxel (a, b) resultante se obtiene como una **combinación lineal** de todos los píxeles de la imagen de entrada, **A**.

$$T[A](a, b) := \sum_{x=0..W-1} \sum_{y=0..H-1} A(x, y)\cdot C(a, b, x, y)$$

- Los coeficientes de la combinación, **C**, dependen de cada posición (a, b) del resultado.
- **Ejemplo**. La transformada de Fourier es una de las transf. lineales más importantes, dada por coeficientes de la forma: $C(a, b, x, y) := e^{(-2\pi i\cdot x\cdot a/W)}\cdot e^{(-2\pi i\cdot y\cdot b/H)}$
- Pero existen otras transformaciones lineales interesantes: transformada del **coseno**, **wavelets**, **Haar**, etc.

5.4. Otras transformaciones lineales.

- **Otra perspectiva de las transf. lineales**: se puede interpretar que la transf. lineal “descompone” una imagen cualquiera, como una suma de componentes básicos.
- La DFT usa los componentes frecuenciales estudiados. En general, tendremos una **base** cualquiera **de imágenes**.



- La transf. es **invertible** si las imágenes de la base son ortogonales entre sí, y hay tantos compon. como píxeles.

5.4.1. Transformada del coseno.

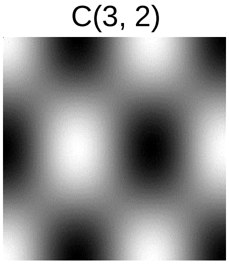
Transformada discreta del coseno (DCT)

- Es parecida a la DFT, pero sólo usa números reales.
- Los coeficientes de la transformación son de la forma:

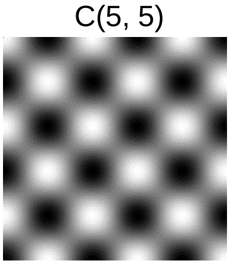
  $C(a, b, x, y) = \cos((2x+1)a\pi/2W) \cdot \cos((2y+1)b\pi/2H)$

- Esto da lugar a una base de imágenes con **forma de hueveras**.

C(3, 2)



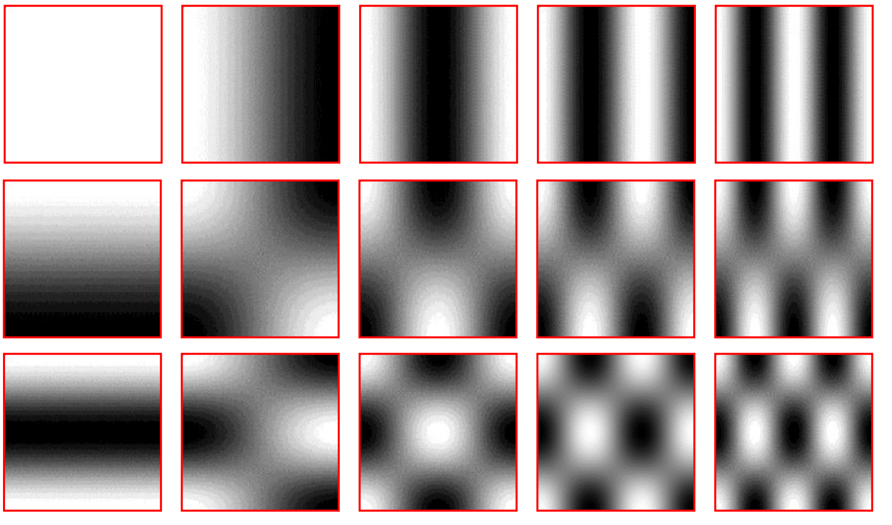
C(5, 5)



- El valor de **a** indica el número de huevos en cada fila (eje X), y el de **b** el número de huevos en cada columna (eje Y).

5.4.1. Transformada del coseno.

- Primeros componentes de la DCT, C(a, b, x, y).

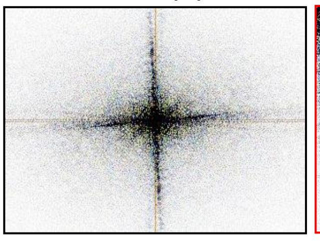


5.4.1. Transformada del coseno.

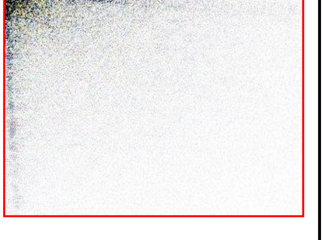
- El resultado de la DCT se asemeja a un **cuadrante** de la DFT. Magnitud ≡ valor absoluto del número.

Imagen de entrada, A

DFT(A)



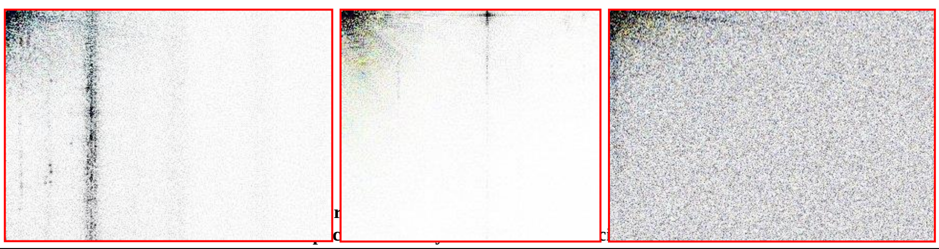
DCT(A)



- La DCT tiene **propiedades parecidas** a la DFT (comportamiento frente a rotación, escala, suavizado, etc.).
- Las **aplicaciones** de la DCT son las mismas que para la DFT: eliminación de ruido, compresión de imágenes y análisis frecuencial de las imágenes.

5.4.1. Transformada del coseno.

- Además, la DCT tiene dos **ventajas**: sólo usa números reales y no tiene problemas de discontinuidades (los que obligan al uso de enventanado en DFT).
- **Ejemplo**. Uso de la DCT en eliminación de ruido.



5.4.1. Transformada del coseno.

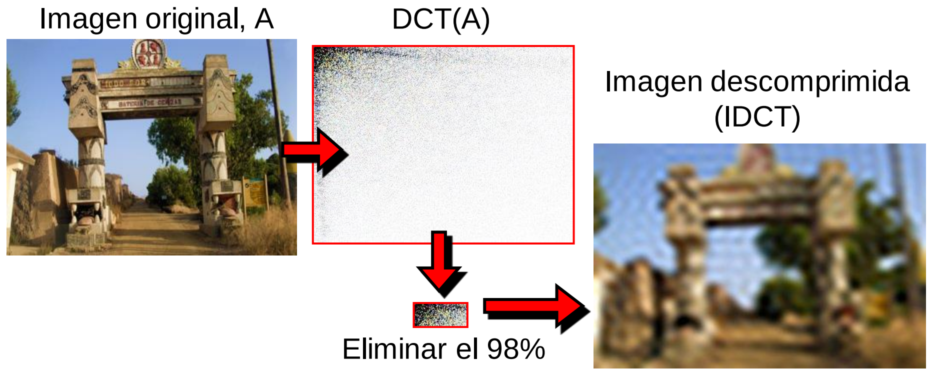
- **Ejemplo**. Igual que con DFT, se ponen a 0 las frecuencias de ruido y luego se calcula la DCT inversa (IDCT).

DCT retocadas (eliminación de bandas de ruido)

Transformada inversa de las DCT retocadas

5.4.1. Transformada del coseno.

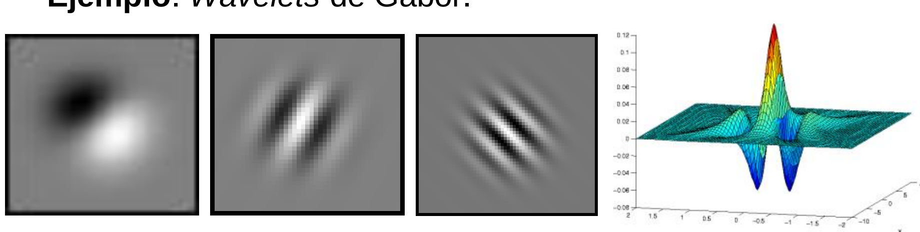
- **Ejemplo**. Uso de la DCT en compresión de imágenes.
- La idea es igual, quedarse sólo con los componentes mayores (en valor absoluto).



- La DCT es la que realmente se usa en **JPEG** y en **MPEG**. No se aplica en toda la imagen, sino en grupos de 8x8.

5.4.2. Transformadas wavelet.

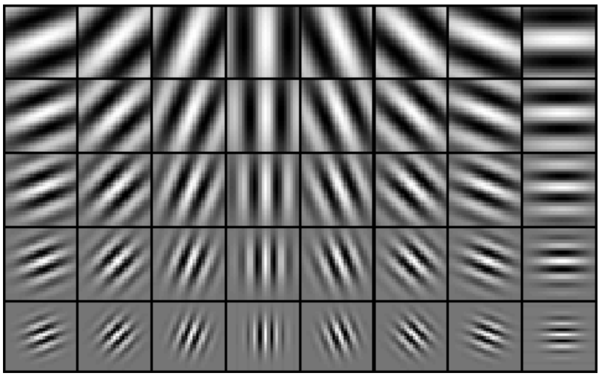
- Existen muchas otras transformaciones lineales, aún en fases de estudio e investigación.
- Un tipo importante son las **transformadas basadas en wavelets**.
- **Idea**: los componentes de la DFT y la DCT son ondas (*waves*) que se extienden indefinidamente. Pero normalmente en las imágenes sólo aparecen trozos de ondas (*wavelets*) de extensión espacial limitada.
- **Ejemplo**. *Wavelets* de Gabor.



5.4.2. Transformadas wavelet.

- Existen muchos tipos de wavelets, muchas variantes.
- Por ejemplo, los **wavelets de Gabor** se definen como un producto de una función sinusoidal por una gaussiana.
- Una vez con la forma del wavelet, se define la **base de imágenes** asociada a la transformación.

Ejemplo. Base de imágenes de una transformada con wavelets de Gabor.

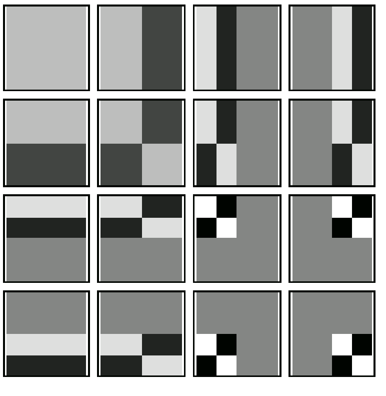


http://www.hig.no/~erikh/papers/scia99/node6.html

5.4.2. Transformadas wavelet.

- Los **wavelet de Haar** se definen usando func. escalonadas.
- La base de imágenes cambia en el tamaño del escalón y su situación dentro de la imagen.

Ejemplo. Base de imágenes de una transformada con wavelets de Haar.



http://www.echoescan.com/expr2.htm

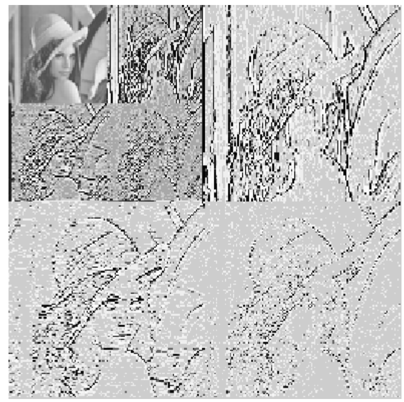
- Esta transf. se usa mucho en compresión y análisis de imágenes.

5.4.2. Transformadas wavelet.

- **Ejemplo**. Transf. wavelet usando funciones de Gabor.

Imagen de entrada

Transf. Gabor wavelet



- Lo interesante es que gran parte de los coeficientes son 0 → se pueden descartar.
- Además, la transformación es invertible (ojo, no todas lo son).

5.4.2. Transformadas wavelet.

- Muchos coeficientes 0 y transf. invertible → Se puede utilizar para **compresión** de imágenes.
- De hecho, los wavelets son la base del estándar **JPEG2000**.

JPEG2000, ratio 1:155

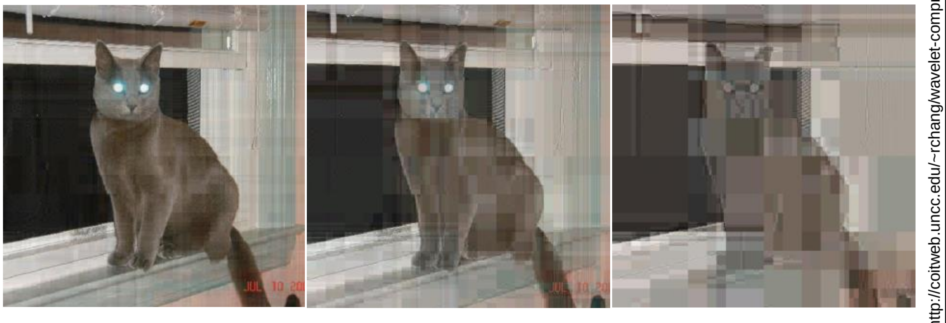
JPEG, ratio 1:155

- Generan menos artificios con igual nivel de compresión, aunque el cálculo es algo más costoso.

5.4.2. Transformadas wavelet.

- La misma idea se puede aplicar a la transformada de Haar.
- Intuitivamente, la compresión se podría ver como una detección de **regiones cuadradas** uniformes.
- **Ejemplo**. Compresión mediante transformadas de Haar: aplicar la transformación, y eliminar los componentes frecuenciales menores.

Compresión al 80% Compresión al 95% Compresión al 99%



http://coitweb.uncc.edu/~rchang/wavelet-compression.html

5.4. Otras transformaciones lineales.

Conclusiones:

- Las **transformaciones lineales** ofrecen otro modo de ver y manipular las imágenes.
- Las transformaciones DFT y DCT descomponen una imagen como suma de **componentes frecuenciales**.
- El dominio frecuencial es dual al espacial. De hecho, muchos expertos trabajan preferiblemente en el frecuencial.
- **Propiedad fundamental**: la convolución en el dominio espacial es equivalente a una multiplicación en el dominio frecuencial.
- **Aplicaciones** de las transf. lineales: compresión, eliminación de ruido, restauración (filtros inversos), análisis de imágenes, etc.

Anexo A.5.
Color y transformaciones lineales en OpenCV.

- Espacios de color
- Otras operaciones con color
- Transformaciones lineales DFT y DCT

A.5. Color y transf. lineales en OpenCV.

Tipos de operaciones:

- **Espacios de color**:
  - cvtColor
- **Otras operaciones con color**:
  - floodFill, mixChannels
- **Transformaciones lineales**:
  - dft, dct

A.5. Color y transf. lineales en OpenCV.

Espacios de color:

- OpenCV incluye operaciones para cambiar el **modelo de color** de una imagen:
  - RGB (BGR), XYZ, HSV, HLS, Lab, Luv, YCrCb, GRAY
- En OpenCV, las imágenes por defecto se almacenan en el **orden BGR** (no en RGB).
- El control del color no es riguroso. Si convertimos una imagen, por ejemplo, de BGR a HLS y mostramos la imagen convertida, entonces:
  - En el canal B vemos la H.
  - En el canal G vemos la L.
  - En el canal R vemos la S.
- Recordar también las funciones relacionadas con histogramas (anexo A.2): calcHist.

A.5. Color y transf. lineales en OpenCV.

- El **modo normal de trabajo** será:
  - Transformar la imagen de BGR al espacio ???.
  - Operar en el espacio ???.
  - Transformar el resultado de ??? a BGR.
- Opcionalmente, se pueden **separar los canales** para trabajar con ellos independientemente (split) y luego **juntarlos** (merge).
- Todas las transformaciones entre espacios son siempre **a través de BGR (RGB)**. Por ejemplo, para convertir de HSV a XYZ habrá que hacer: convertir de HSV a RGB + convertir de RGB a XYZ.
- **Ojo**, cada vez que se convierte el espacio se puede perder un poco de calidad (por los redondeos de los cálculos).
- **Recordar**: no se almacena en RGB sino en BGR.

A.5. Color y transf. lineales en OpenCV.

- **Conversión de color en OpenCV**:

  void **cvtColor** (Mat src, Mat dst, int code);

  – Transforma la imagen **src** en **dst**, según la operación dada en **code**.

  – Deben tener el mismo tamaño y nº de canales adecuado.

  – **code**: CV_BGR2XYZ, CV_RGB2XYZ, CV_XYZ2BGR, CV_XYZ2RGB, CV_BGR2YCrCb, CV_RGB2YCrCb, CV_YCrCb2BGR, CV_YCrCb2RGB, CV_BGR2HSV, CV_RGB2HSV, CV_HSV2BGR, CV_HSV2RGB, CV_BGR2HLS, CV_RGB2HLS, CV_HLS2BGR, CV_HLS2RGB, CV_BGR2Lab, CV_RGB2Lab, CV_Lab2BGR, CV_Lab2RGB, CV_BGR2Luv, CV_RGB2Luv, CV_Luv2BGR, CV_Luv2RGB, CV_BGR2GRAY, CV_GRAY2BGR, CV_BGR2YUV, CV_YUV2BGR…

  – Hay que indicar correctamente el **orden de los canales**. Normalmente el RGB se almacena como BGR → En lugar de usar CV_RGB2???, usar CV_BGR2???.

A.5. Color y transf. lineales en OpenCV.

- **Rellenado de color**:

  void **floodFill** (Mat img, Point seed, Scalar newVal,
  Rect *rect=NULL, Scalar lo=0, Scalar up=0, int flags=4);

  – Rellenar la imagen **img** con el color **newVal**, partiendo del punto **seed**.

  – **rect** (parámetro opcional) contiene información de la región rellenada, el rectángulo contenedor. Se puede descartar pasando en valor NULL.

  – **flags** indica el modo de rellenado. Cada grupo de bits indica una cosa (combinar con OR).

    - Los 3 menos significativos indican la conectividad de la vecindad: 4 u 8.
    - Los más significativos se activan sumando la constante: **CV_FLOODFILL_FIXED_RANGE**

A.5. Color y transf. lineales en OpenCV.

- **Rellenado de color** (segunda variante):

  void **floodFill** (Mat img, Mat mask, Point seed, Scalar newVal, Rect *rect=NULL, Scalar lo=0, Scalar up=0, int flags=4);

  – En esta variante se pasa una máscara **mask** como parámetro, que debe ser una imagen de 2 píxeles más de ancho y 2 más de alto que img, y de 1 solo canal 8U. (La máscara “envuelve” a img.)

  – La máscara se puede usar como **entrada**: si existe una máscara, solo se rellenan píxeles donde la máscara valga 0.

  – Y también se puede usar como **salida**: en el modo “mask only”, usando el flag:

    CV_FLOODFILL_MASK_ONLY

A.5. Color y transf. lineales en OpenCV.

- **Opciones de rellenado de color**:
  – **CV_FLOODFILL_MASK_ONLY**: solo máscara. Si se activa, solo se modifica la máscara, **mask**, pero no la imagen. En la máscara, se ponen a 255 los píxeles rellenados.
  – **CV_FLOODFILL_FIXED_RANGE**: rango fijo. La diferencia es respecto al punto **seed**. Si no se pone, el rango es flotante (diferencia respecto al píxel adyacente al rellenado).
  – **Medida de distancia (en rango fijo)**: si el píxel **seed** vale (r, g, b), el píxel rellenado debe estar en el rango: (r-lo.r...r+up.r ; g-lo.g...g+up.g ; b-lo.b...b+up.b)
  – Igual en modo **rango flotante**.
  – Ver el **programa de ejemplo**: samples\cpp\ffilldemo.cpp
- **Mezcla de canales: mixChannels**
  – Sirve para mezclar (reordenar) los canales de una o varias imágenes (por ejemplo, intercambiar dos canales).

A.5. Color y transf. lineales en OpenCV.

Otras operaciones con color:

- Para hacer modificaciones de color, se pueden usar las operaciones globales y locales (ver anexos A.2 y A.3) de forma independiente por canal.
- **Ejemplo**. Colorear *img* con “un poco” de color rojo.

  img= CV_RGB(45, 0, 0) + img;
- Otra operación interesante: **mean**, obtener la media de todos los píxeles de una imagen, por canales (por ejemplo, para hacer balance de blancos).
- Ojo a la **suma módulo 180** de la página 43. Hay que:

1. Convertir la imagen a 16S con escala: 255.0/180.0
2. Hacer la suma en 16S
3. Hacer un AND con 0xFF (equivalente a calcular módulo 256)
4. Convertir a 8U con escala: 180.0/255.0

A.5. Color y transf. lineales en OpenCV.

Transformaciones lineales:

- OpenCV incluye dos tipos de transformaciones lineales y sus correspondientes inversas: **DFT** y **DCT**. Las funciones de OpenCV son **restrictivas** en cuanto a profundidad y número de canales.
- Funciones de **OpenCV**: dft (idft), dct (idct).
- Las imágenes resultantes pueden estar **escaladas** o no (divididas por el número de píxeles de la imagen). Mejor no escalar.
- Recordar que la DFT usa **números complejos** y es **semi-simétrica**. Esto da lugar a un formato especial de almacenamiento, conocido como el **formato empaquetado RCPack2D**.

A.5. Color y transf. lineales en OpenCV.

- **Formato empaquetado RCPack2D**: se usa para las imágenes en el dominio de DFT. Una **Mat Y** de **N**x**M** almacena una imagen compleja de la siguiente forma:

Re $Y_{0,0}$	Re $Y_{0,1}$	Im $Y_{0,1}$	Re $Y_{0,2}$	Im $Y_{0,2}$	...	Re $Y_{0,N/2-1}$	Im $Y_{0,N/2-1}$	Re $Y_{0,N/2}$
Re $Y_{1,0}$	Re $Y_{1,1}$	Im $Y_{1,1}$	Re $Y_{1,2}$	Im $Y_{1,2}$	...	Re $Y_{1,N/2-1}$	Im $Y_{1,N/2-1}$	Re $Y_{1,N/2}$
Im $Y_{2,0}$	Re $Y_{2,1}$	Im $Y_{2,1}$	Re $Y_{2,2}$	Im $Y_{2,2}$	...	Re $Y_{2,N/2-1}$	Im $Y_{2,N/2-1}$	Im $Y_{2,N/2}$
....	....	....	....	....	....	....	....	....
Re $Y_{M/2-1,0}$	Re $Y_{M-3,1}$	Im $Y_{M-3,1}$	Re $Y_{M-3,2}$	Im $Y_{M-3,2}$	...	Re $Y_{M-3,N/2-1}$	Im $Y_{M-3,N/2-1}$	Re $Y_{M-3,N/2}$
Im $Y_{M/2-1,0}$	Re $Y_{M-2,1}$	Im $Y_{M-2,1}$	Re $Y_{M-2,2}$	Im $Y_{M-2,2}$	...	Re $Y_{M-2,N/2-1}$	Im $Y_{M-2,N/2-1}$	Im $Y_{M-2,N/2}$
Re $Y_{M/2,0}$	Re $Y_{M-1,1}$	Im $Y_{M-1,1}$	Re $Y_{M-1,2}$	Im $Y_{M-1,2}$	...	Re $Y_{M-1,N/2-1}$	Im $Y_{M-1,N/2-1}$	Im $Y_{M-1,N/2}$

- **Observar**:
  - Algunos píxeles sólo tienen parte real, por ejemplo, el (0,0).
  - La última columna solo se usa si **N** es par, y la última fila si **M** es par.
  - Las filas solo llegan hasta **N**/2 porque la imagen es semi-simétrica: Y(a,b) es el conjugado de Y(W-a, H-b).

A.5. Color y transf. lineales en OpenCV.

- Es conveniente definirse funciones para **empaquetar/ desempaquetar** imágenes en este formato. Por ejemplo, de formato **RCPack2D** a formato **Fase/Magnitud**.

Usar esta constante en la llamada

- **Transformada Discreta de Fourier**:
  void **dft** (Mat src, Mat dst, CV_DXT_FORWARD)
  - Transformar la imagen **src** (en el dominio espacial) en la **dst** (dominio frecuencial).
  - Usar imágenes de 1 canal y profundidad real, es decir, 32F o 64F. También se puede usar profundidad *compleja*.
  - El resultado estará en formato RCPack2D.
  - La llamada usa el algoritmo O(n·log n) de **FFT**.

A.5. Color y transf. lineales en OpenCV.

- **Transformada Discreta Inversa de Fourier**:

  void **idft** (Mat src, Mat dst);
  - Transformar la imagen **src** (en el dominio frecuencial) en la **dst** (dominio espacial), es decir, la transformación inversa a la DFT.
  - Igual que antes, la función **idft** es restrictiva en los tipos y profundidades.

- **Ejemplo**. Dada una imagen **img** (de 8 bits y 1 canal), la imagen **imgIDFT** resultante del siguiente código será la misma.

```
Mat imgDFT, imgIDFT;
dft(img, imgDFT);
idft(imgDFT, imgIDFT);
```


A.5. Color y transf. lineales en OpenCV.

- La transformada del coseno, **dct**, no tiene problemas de representación, porque solo usa números reales.
- **Transformada Discreta del Coseno**:

  void **dct** (Mat src, Mat dst);
  - Dada la imagen **src** (en el dominio espacial) obtiene la transformada del coseno en **dst**.
  - La función **dct** es restrictiva en los tipos y profundidades admitidos.
  - Para hallar la magnitud, podemos usar la función **abs** (valor absoluto de una imagen).

A.5. Color y transf. lineales en OpenCV.

- **Transformada Discreta Inversa del Coseno**:

  void **idct** (Mat src, Mat dst);

  – Dada la imagen **src** obtienen la transformada inversa del coseno en **dst**.

- **Ejemplo**. Ver la DCT de una imagen en grises:

```
Mat imagen= imread(nombre, 0);
Mat conv, imgDCT;
imagen.convertTo(conv, CV_32FC1, 1.0/256);
dct(conv, imgDCT);
imshow("Imagen", imgDCT);
```


A.5. Color y transf. lineales en OpenCV.

- **Ejemplo 1**. Dada una imagen **img**, calcular el espectro (magnitud de la DFT, sin centrar) en la imagen **res**.

```
Mat img= imread(nombre.toStdString(), 0);
Mat escala;
img.convertTo(escala, CV_32FC1, 1.0/255);
Mat imagenDTF1(img.size(), CV_32FC1);
dft(escala, imagenDTF1);
Mat imagenDFT2(img.size(), CV_32FC1);
ggmComplexToMagnitud(imagenDTF1, imagenDFT2);
Mat res;
imagenDFT2.convertTo(res, CV_8U, -1, 255);
namedWindow("Magnitud", 0);
imshow("Magnitud", res);
```

- Donde tenemos definidas las funciones de las siguientes páginas…

A.5. Color y transf. lineales en OpenCV.

```
void ggmReimToMag (Mat ent, int x1, int y1, int x2, int y2,
              Mat sal, int x, int y, int xc= -1, int yc= -1)
{
   float pf1= ent.at<float>(y1, x1);
   float pf2= x2>=0 ? ent.at<float>(y2, x2) : 0;
   for (int i= 0; i<3; i++)
      pf1= sqrt(pf1*pf1+pf2*pf2);
   sal.at<float>(y, x)= pf1;
   if (xc>=0) sal.at<float>(yc, xc)= pf1;
}
```


A.5. Color y transf. lineales en OpenCV.

```
void ggmComplexToMagnitud (Mat ent, Mat sal)
{
   int ancho= ent.size().width, ancho2= (ancho%2 ? ancho-1 : ancho-2);
   int alto= ent.size().height, alto2= (alto%2 ? alto-1 : alto-2);
   int x, y;
   ggmReimToMag(ent, 0, 0, -1, -1, sal, 0, 0);
   for (y= 1; y<alto; y++) for (x= 1; x<ancho2; x+=2)
      ggmReimToMag(ent, x, y, x+1, y, sal, (x-1)/2+1, y, ancho-(x-1)/2-1, alto-y);
   for (x= 1; x<ancho2; x+=2)
      ggmReimToMag(ent, x, 0, x+1, 0, sal, (x-1)/2+1, 0, ancho-(x-1)/2-1, 0);
   for (y= 1; y<alto2; y+=2)
      ggmReimToMag(ent, 0, y, 0, y+1, sal, 0, (y-1)/2+1, 0, alto-(y-1)/2-1);
   if (alto%2==0) ggmReimToMag(ent, 0, alto-1, -1, -1, sal, 0, alto/2);
   if (ancho%2==0) {
      ggmReimToMag(ent, ancho-1, 0, -1, -1, sal, ancho/2, 0);
      for (y= 1; y<alto2; y+=2)
         ggmReimToMag(ent, ancho-1, y, ancho-1, y+1,
                     sal, ancho/2, (y-1)/2+1, ancho/2, alto-(y-1)/2-1);
      if (alto%2==0)
         ggmReimToMag(ent, ancho-1, alto-1, -1, -1, sal, ancho/2, alto/2);
   }
}
```

A.5. Color y transf. lineales en OpenCV.

- Otra opción es usar **el parámetro**: **DFT_COMPLEX_OUTPUT**

  void **dct** (Mat src, Mat dst, DFT_COMPLEX_OUTPUT);

  Con esa opción, no se usa el formato "empaquetado" para el resultado de la DFT, sino que devuelve una matriz dst de números complejos (que tiene la simetría de complejos conjugados propia de la DFT).

- **Ejemplo 2**. Hacer que el color medio de una imagen **img** sea un **color** pasado como parámetro.

```
Scalar color= CV_RGB(250, 50, 50);  // Este es el color objetivo
Scalar media= mean(img);
Scalar suma= color-media;
img= img+suma;
```