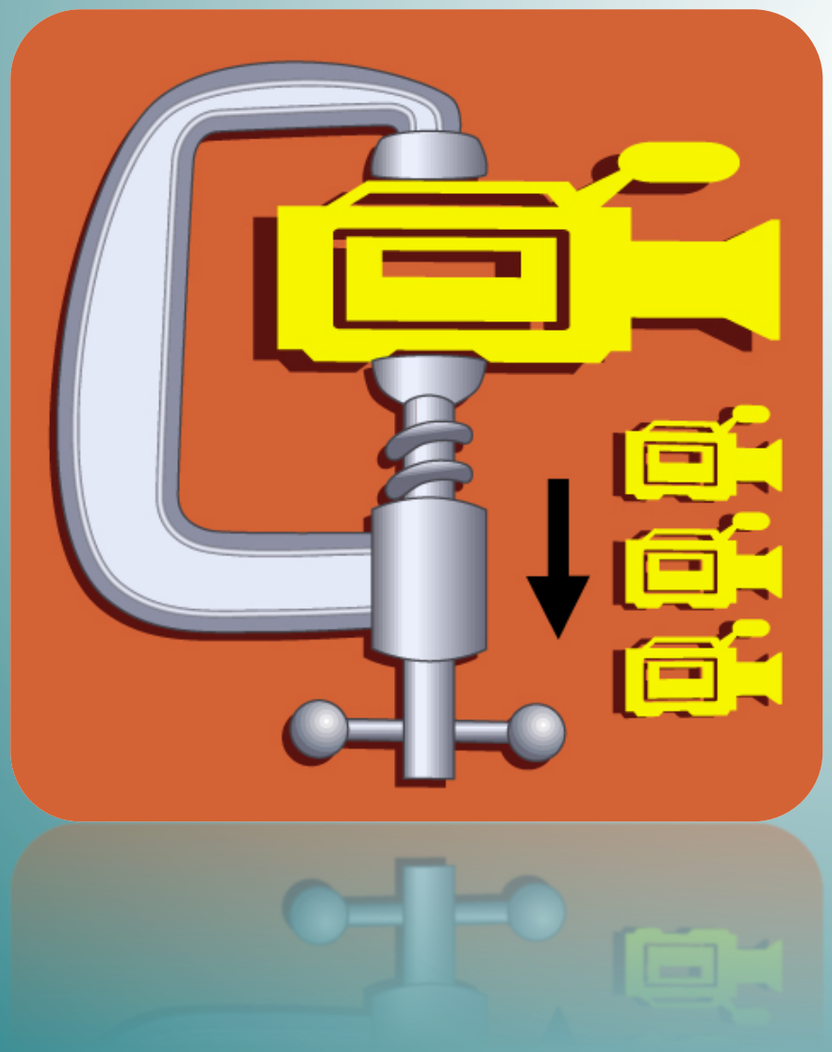
Compresión de Vídeo

Tema 1.2. Vídeo digital



Juan A. Michell Martín
Gustavo A. Ruiz Robredo

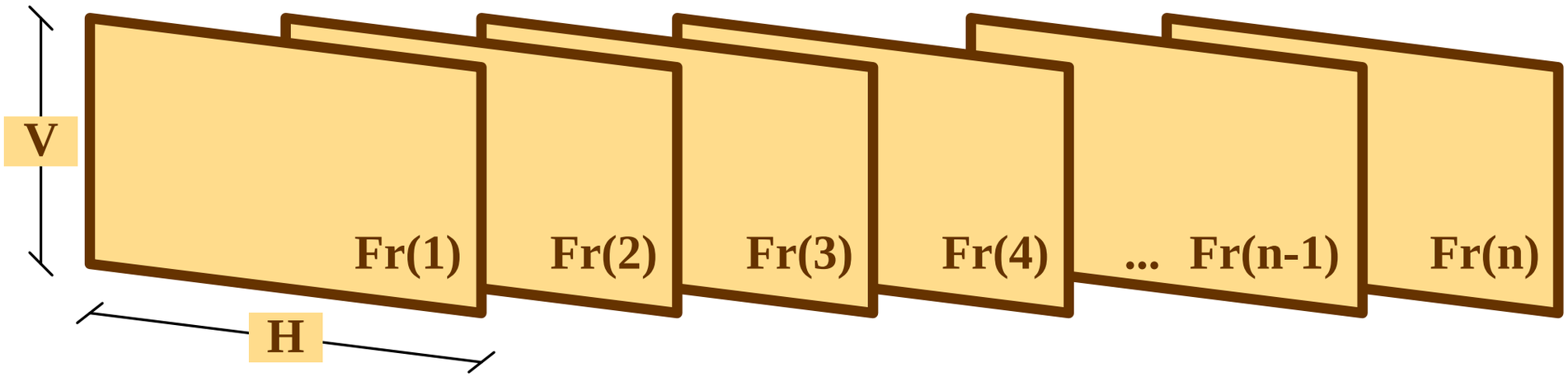
Departamento de Electrónica y Computadores

Este tema se publica bajo Licencia:
Creative Commons BY-NC-SA 4.0

VIDEO DIGITAL

- **Representación digital de una escena continua, adecuada para su almacenamiento y transmisión.**

- **Está formado por imágenes digitales (fotogramas o *frames*), tomadas a intervalos regulares.**



- **Características Básicas:**
  - **Longitud: número de fotogramas (*n*).**
  - **Resolución: tamaño de los fotogramas ($V \times H$).**
  - **Velocidad de muestreo (*frame rate*): número de fotogramas tomados cada segundo.**
  - **Profundidad de bits (bit depth): 8, 10 12 y 16 bits.**

ESPACIO DE COLOR

- **Monocromática**

  Imagen B&W. Cada píxel se representa por un número.

- **RGB**

  Cada píxel se representa por tres números que indican las proporciones relativas de los colores primarios, Rojo (R), Verde (G) y Azul (B). Las tres colores están muy correlacionadas. Adecuado para dispositivos de visualización.

- **YCbCr (YUV)**

  Componentes: brillo Y (*luminancia* o *luma*) y dos diferencias de color (*crominancias* o *cromas*) Cb y Cr. Apropiado para compresión de imagen y vídeo, ya que el sistema visual humano (*HVS*) es menos sensible al color que al brillo.

Clip de Vídeo RGB: 36 Fotogramas 352×240

Fotograma RGB

Componente R

Componente G

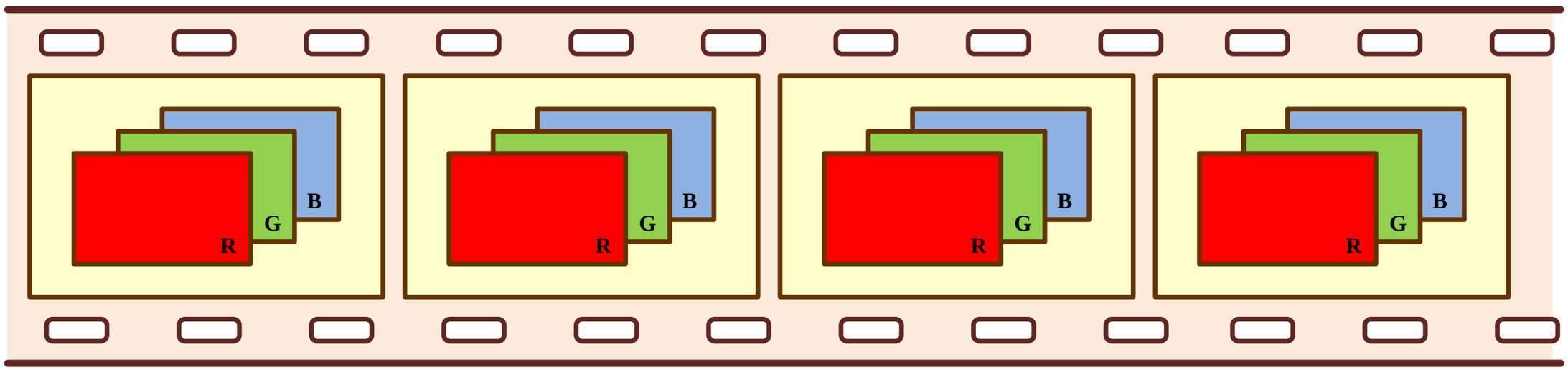
Componente B

Almacenamiento en Matlab

Mapa de color (1 monocroma, 3 RGB)

Matlab 4D: $I(1{:}V, 1{:}H, 1{:}3, n)$

Tamaño frame VxH — **Número de frame**

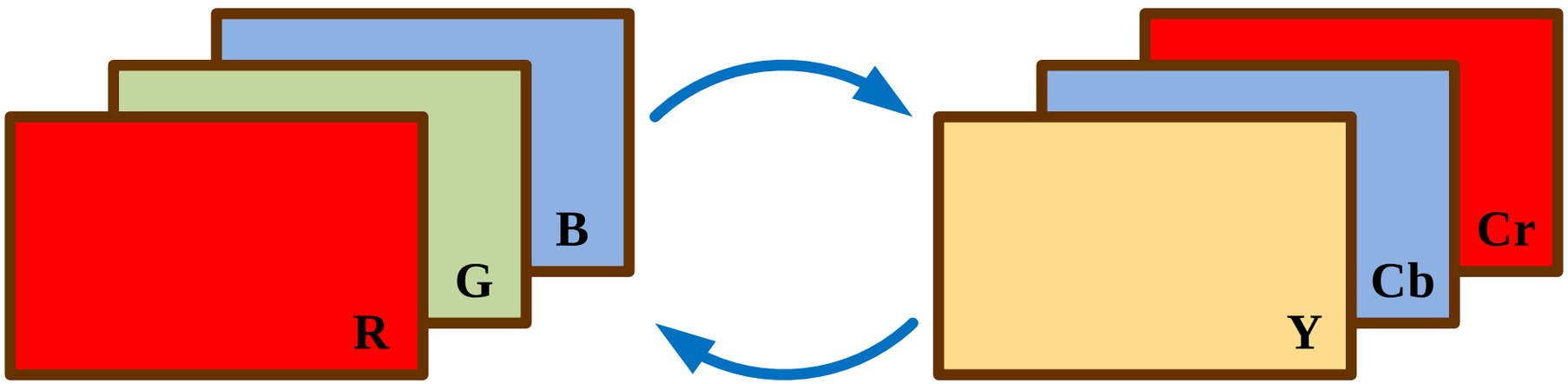


Frame 1	Frame 2	Frame 3	Frame 4
I(1:V,1:H,1:3,1)	I(1:V,1:H,1:3,2)	I(1:V,1:H,1:3,3)	I(1:V,1:H,1:3,4)
R: I(1:V,1:H,1,1)	R: I(1:V,1:H,1,2)	R: I(1:V,1:H,1,3)	R: I(1:V,1:H,1,4)
G: I(1:V,1:H,2,1)	G: I(1:V,1:H,2,2)	G: I(1:V,1:H,2,3)	G: I(1:V,1:H,2,4)
B: I(1:V,1:H,3,1)	B: I(1:V,1:H,3,2)	B: I(1:V,1:H,3,3)	B: I(1:V,1:H,3,4)

CONVERSIÓN RGB-YCbCr

YCbCr: Se adapta más a las características de percepción humana.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168 & -0.331 & 0.499 \\ 0.499 & -0.418 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

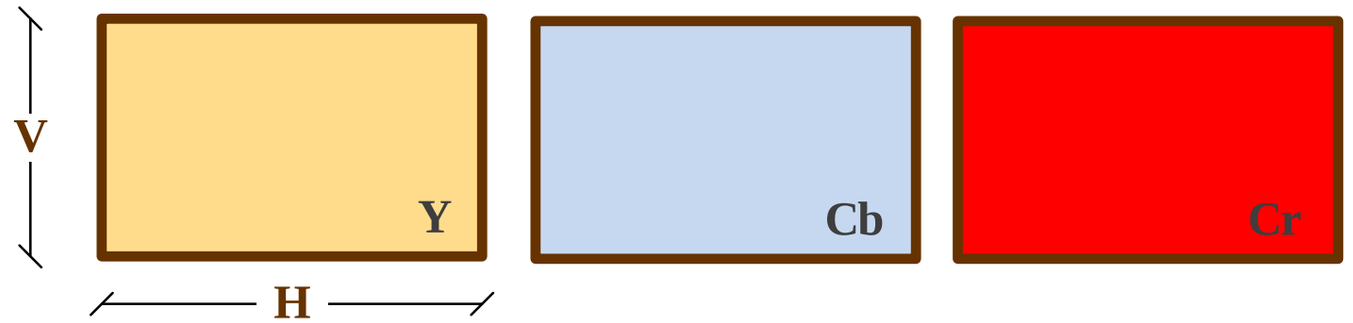


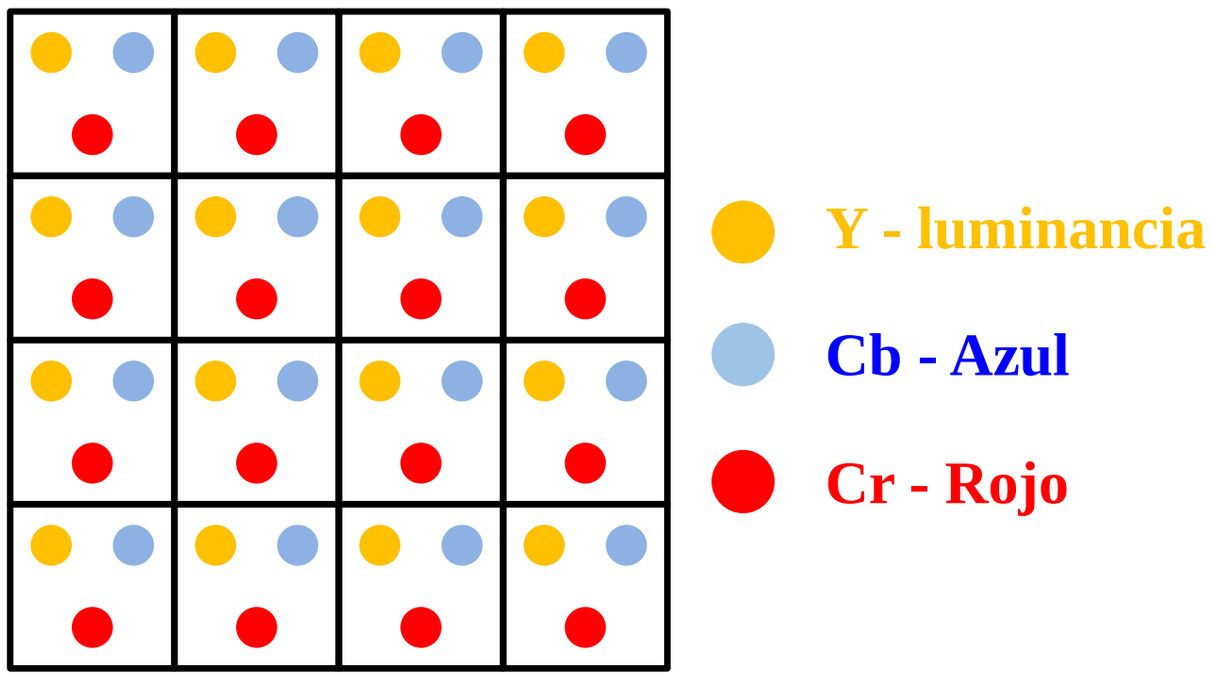
Array 3D: Imagen RGB **Array 3D: Imagen YCbCr**

➢ Formato “4:4:4” : *luma* y *cromas* de igual tamaño

RGB (24 bits/pix) ↔ Y_{444} (24 bits/pix)

Usado, por ejemplo, en los formatos de vídeo profesionales como el HDCAM SR.

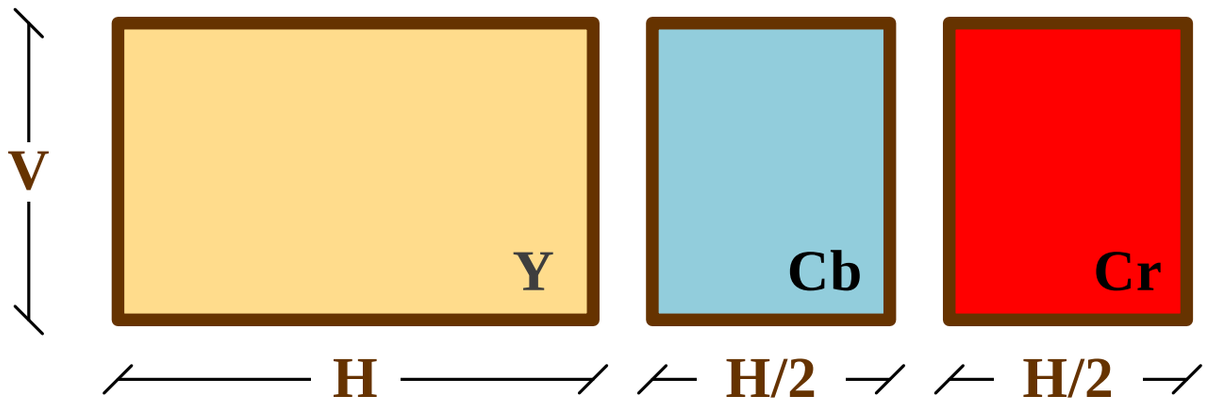


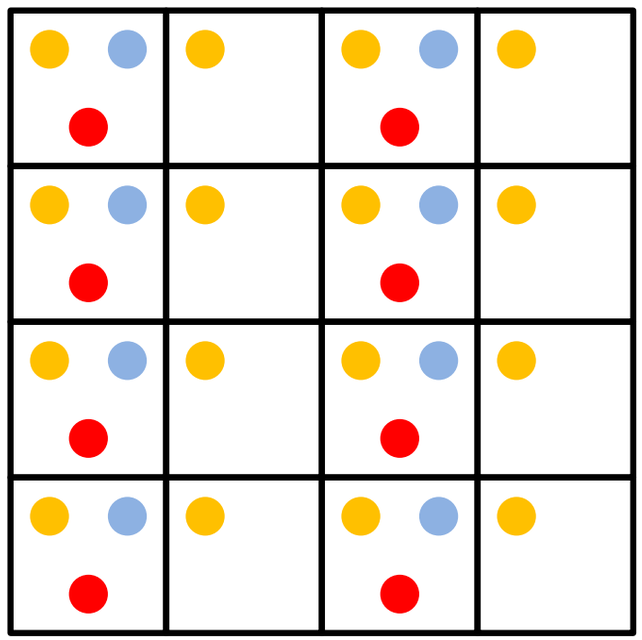


➢ Formato “4:2:2” : reducción de *cromas* a ½

RGB (24 bits/pix) ↔ Y_{422} (16 bits/pix)

Estudios de televisión, sistemas profesionales DV50 y en el formato MPEG-2, recomendado en el documento UIT-R BT.601-7.





Y - luminancia

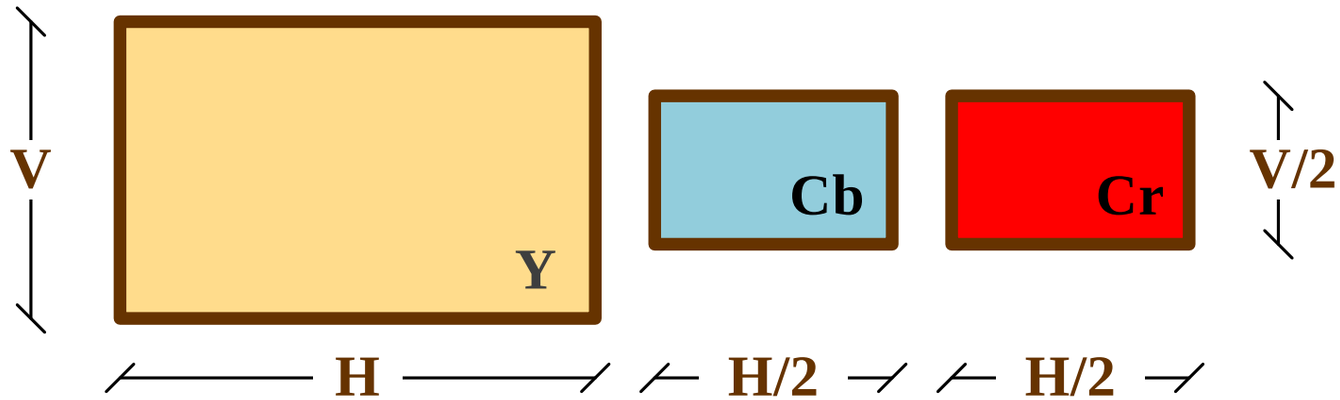
Cb - Azul

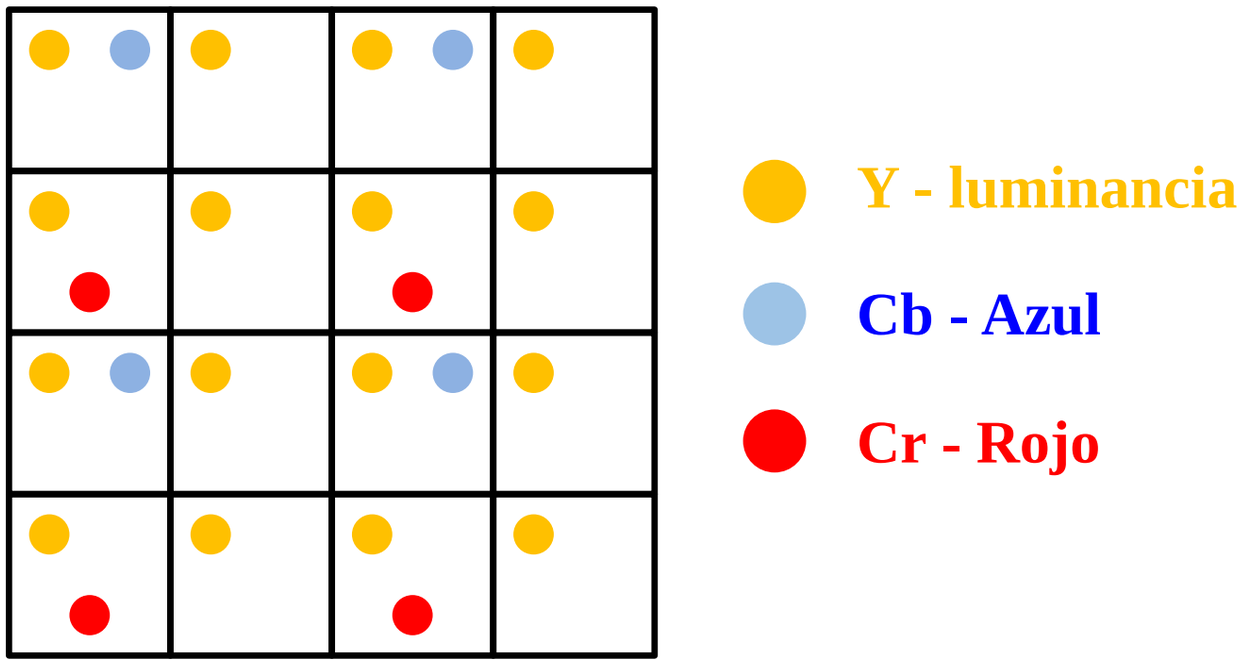
Cr - Rojo

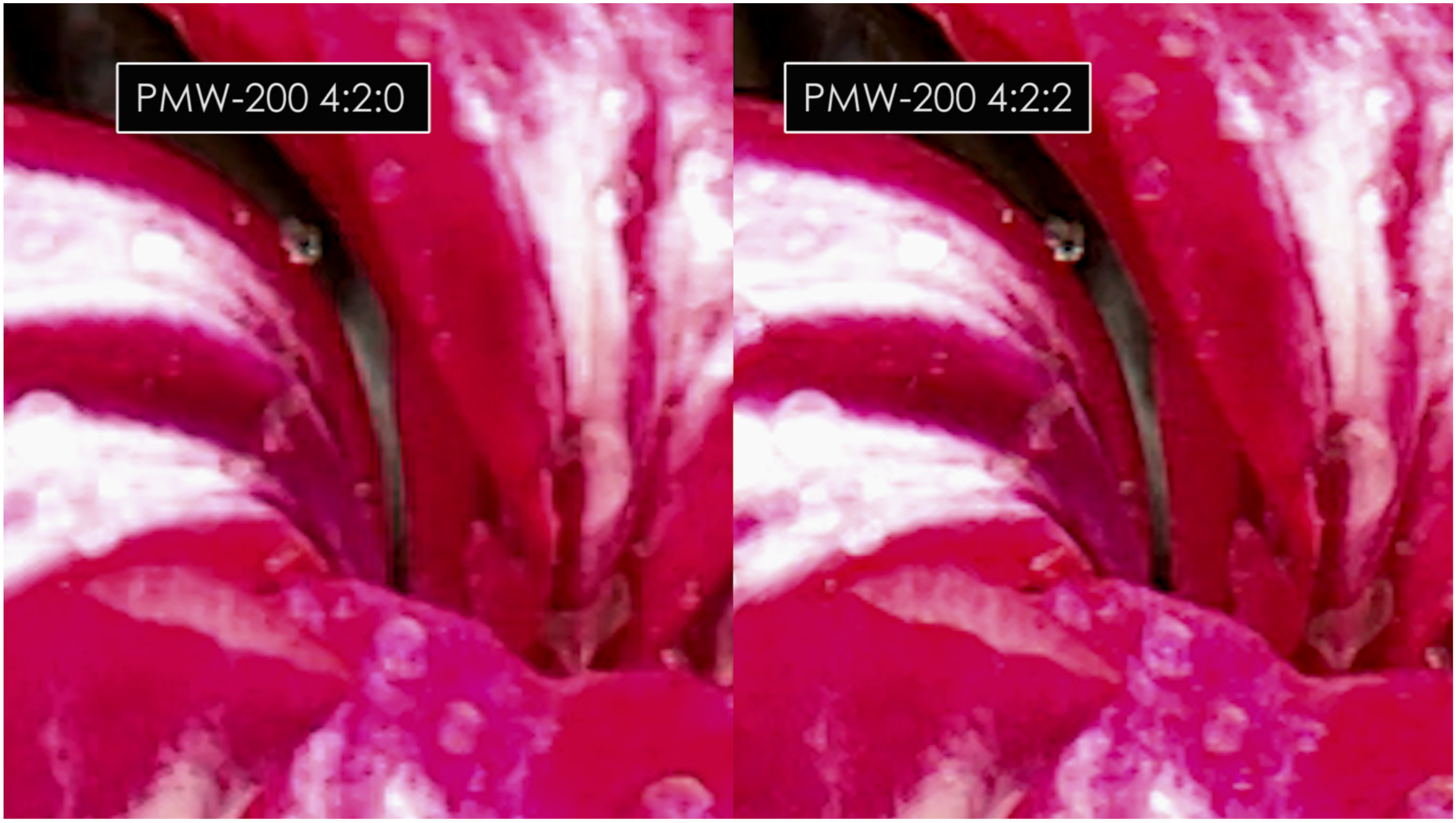
➢ Formato “4:2:0” : reducción de *cromas* a 1/4

RGB (24 bits/pix) ↔ Y_{420} (12 bits/pix)

Formato JPEG a nivel de imagen, video de baja/media calidad, DV25 y algunas variantes del formato MPEG-2/MPEG-4. Es el más utilizado.





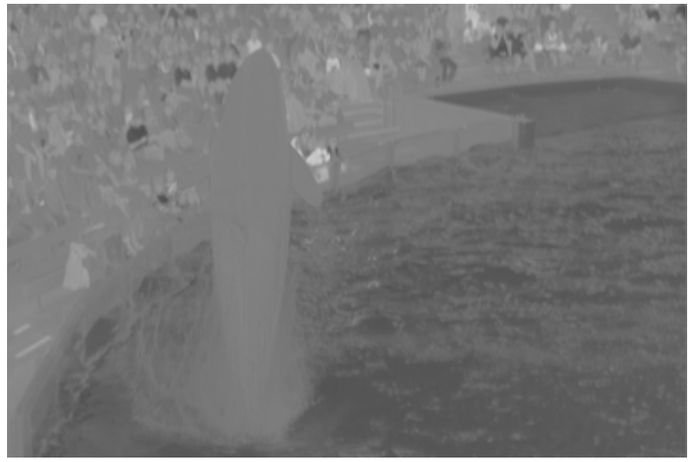


http://www.eoshd.com/comments/topic/2125-exploring-nikon-d5200-hdmi-output-review-update/?page=2

Fotograma YCbCr 4:2:0

Componente Y
720×480

Componente Cb
360×240



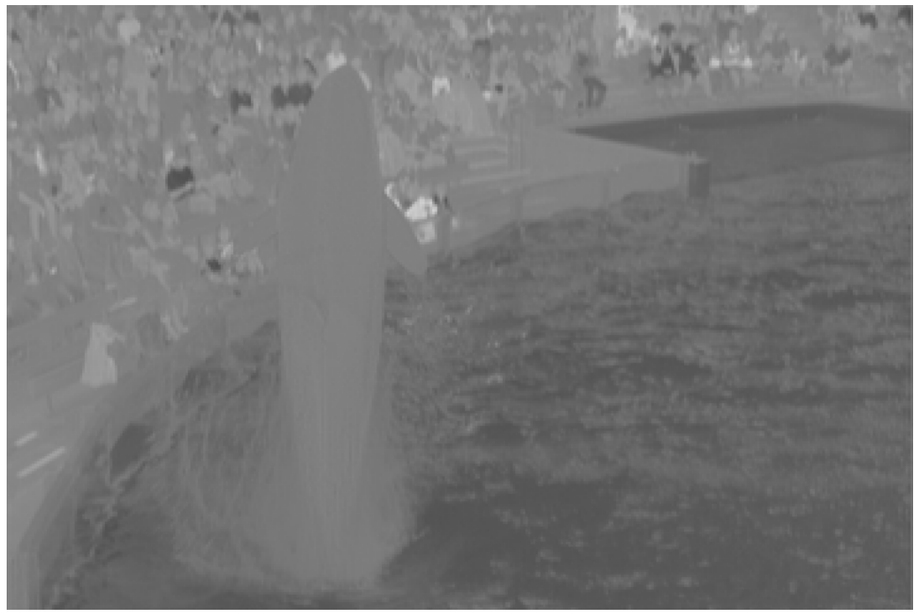
Componente Cr
360×240

Interpolación de las *Cromas*

Cb 360×240

Cb 720×480

Cr 360×240

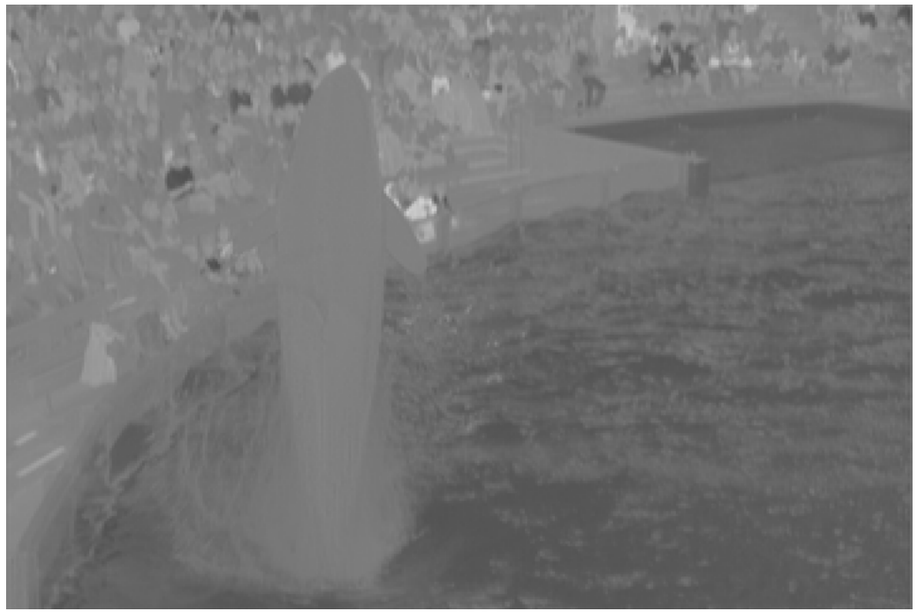


Cr 720×480

Fotograma YCbCr

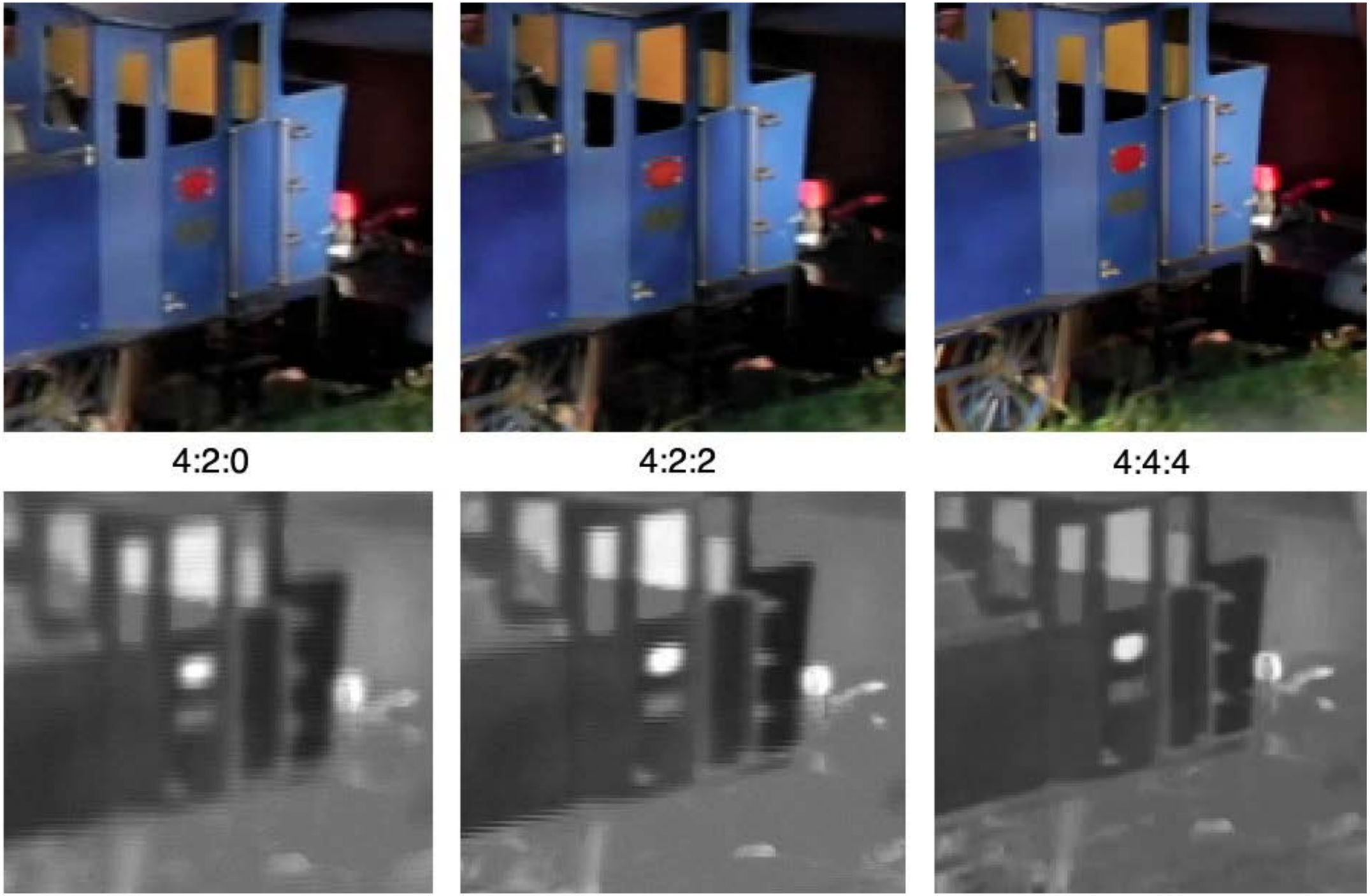
Componente Y

Componente Cb Reconstruida



Componente Cr Reconstruida

YCbCr 4:2:0 4:2:2 4:4:4



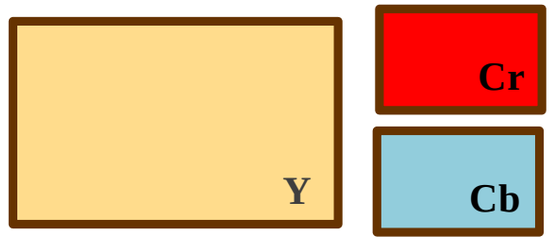
https://en.wikipedia.org/wiki/Chroma_subsampling

Conversión YCbCr → RGB

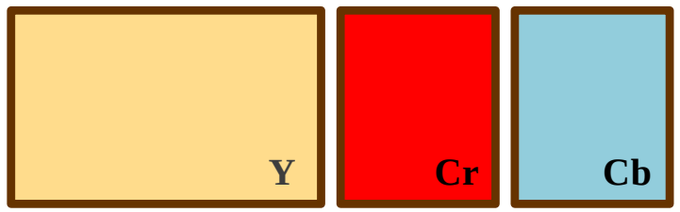
Y **Cb** **Cr**

R **G** **B**

Conversión YCbCr → RGB

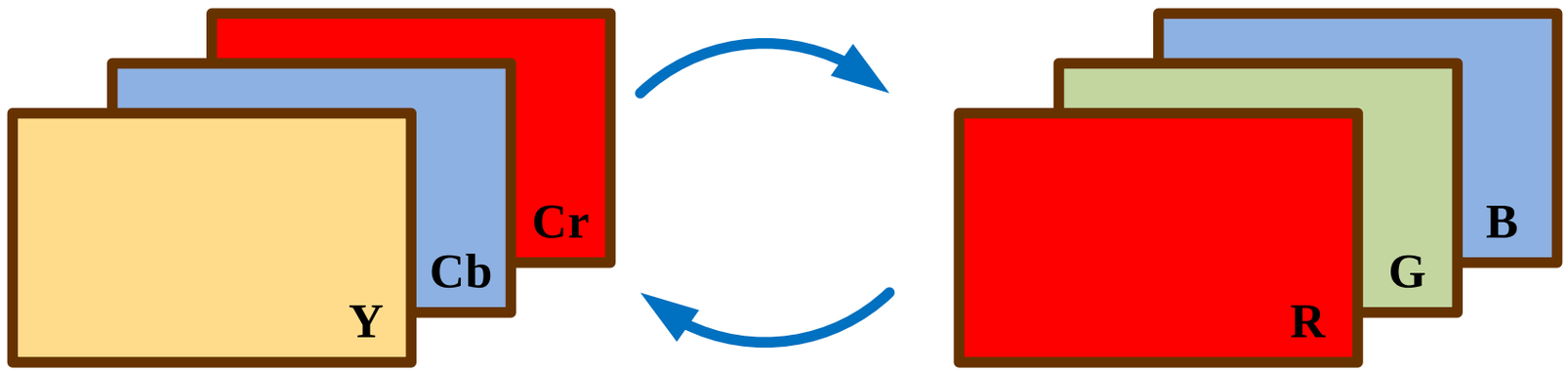


Array 3D: Imagen YCbCr 4:2:0



Array 3D: Imagen YCbCr 4:2:2

Conversión 4:2:0 o 4:2:2 a 4:4:4



Array 3D: Imagen YCbCr 4:4:4

Array 3D: Imagen RGB

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.402 \\ 1 & -0.344 & -0.714 \\ 1 & 1.772 & 0 \end{bmatrix} \begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix}$$

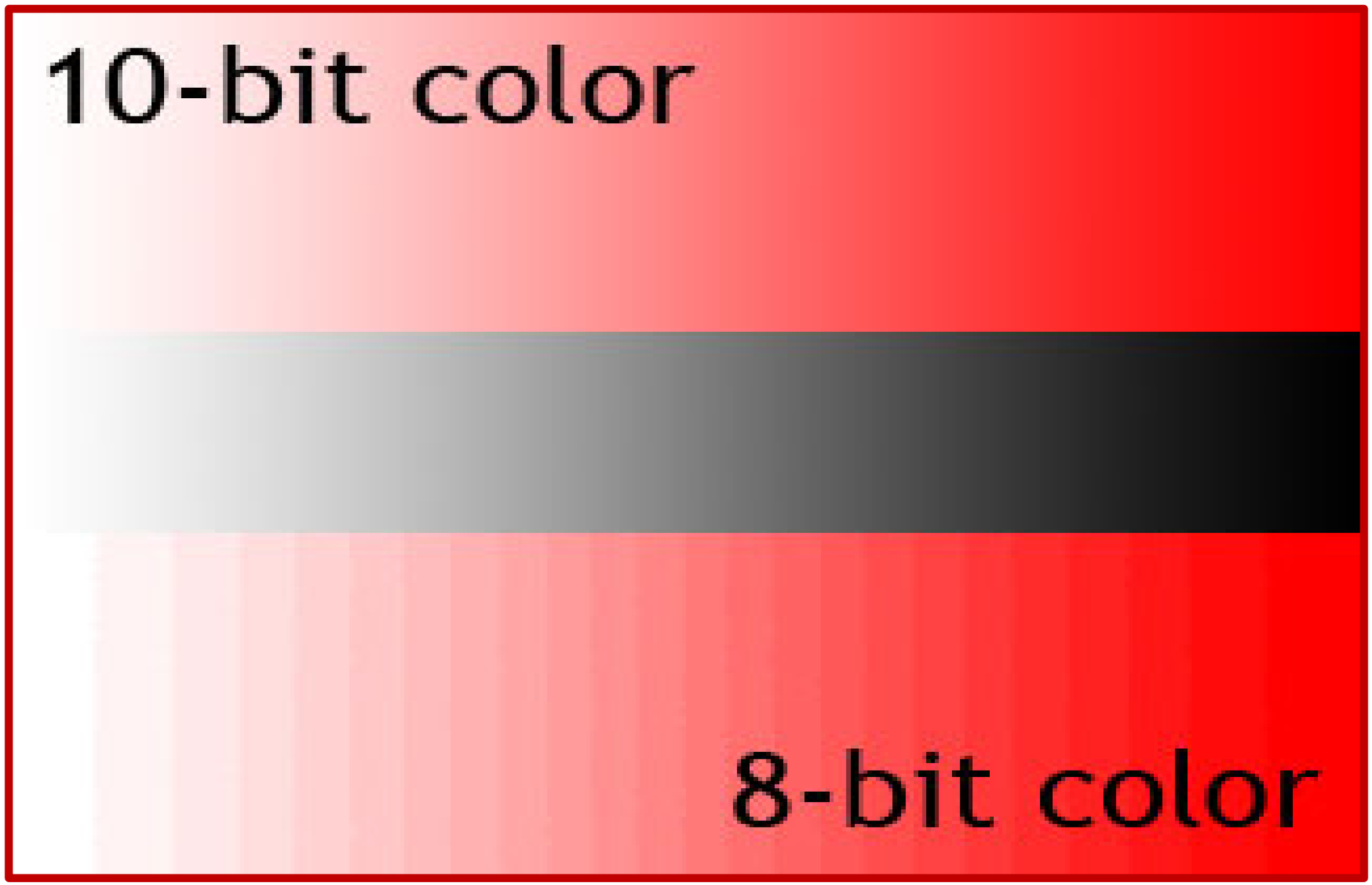
Fotograma RGB 720×480

YCbCr 4:2:0 versus RGB

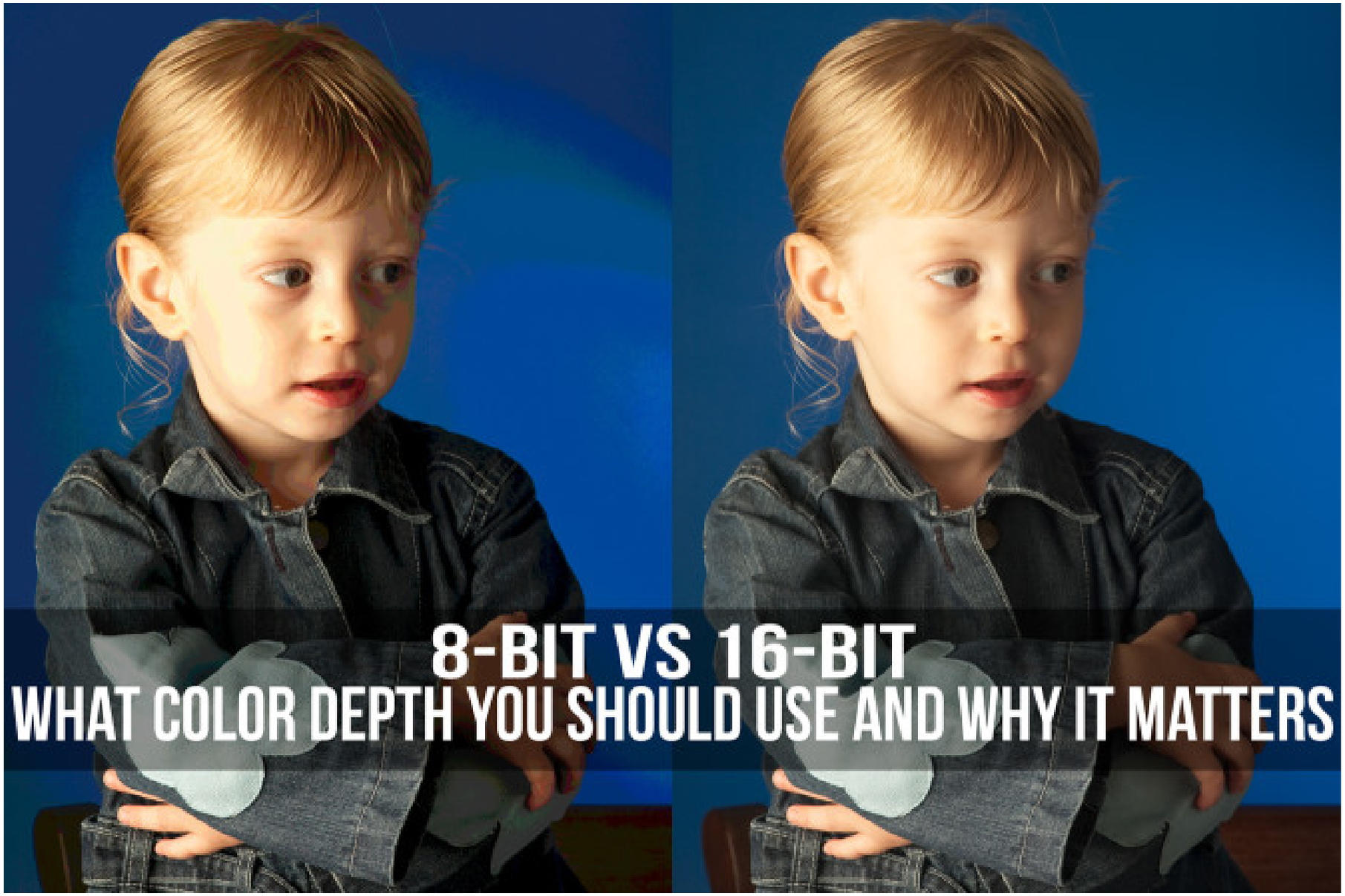
https://www.youtube.com/watch?v=_m1_j9neuQA

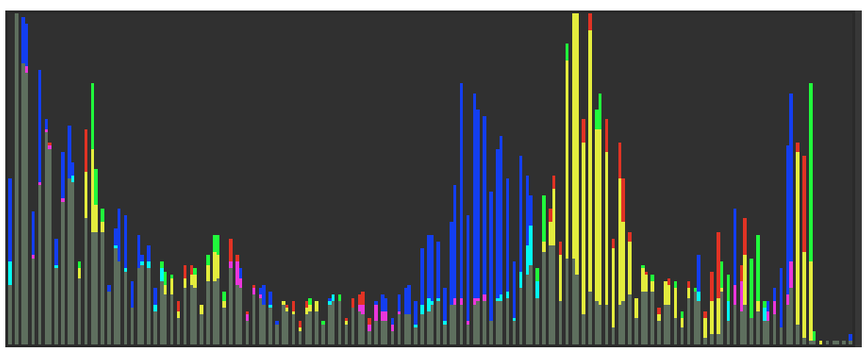
EFECTO DE PROFUNDIDAD DE BIT

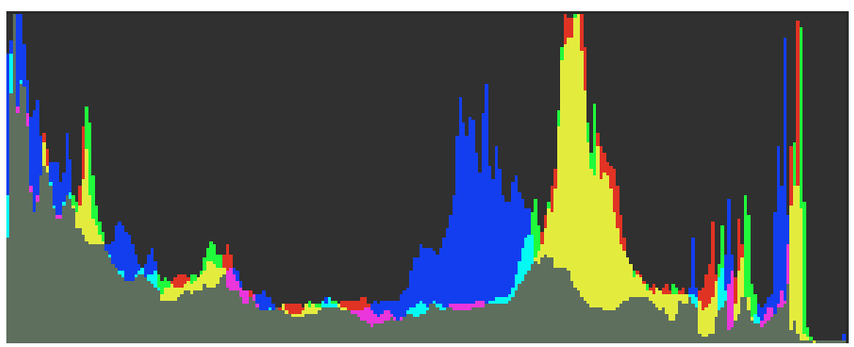
8 bit = $2^8 \cdot 2^8 \cdot 2^8$= 16.7M ⇔ 10 bit = $2^{10} \cdot 2^{10} \cdot 2^{10}$= 1261M



https://forum.videohelp.com/threads/326820-8-bit-color-RGB-versus-10-bit-color-4-2-2







https://www.diyphotography.net/8-bit-vs-16-bit-color-depth-use-matters/

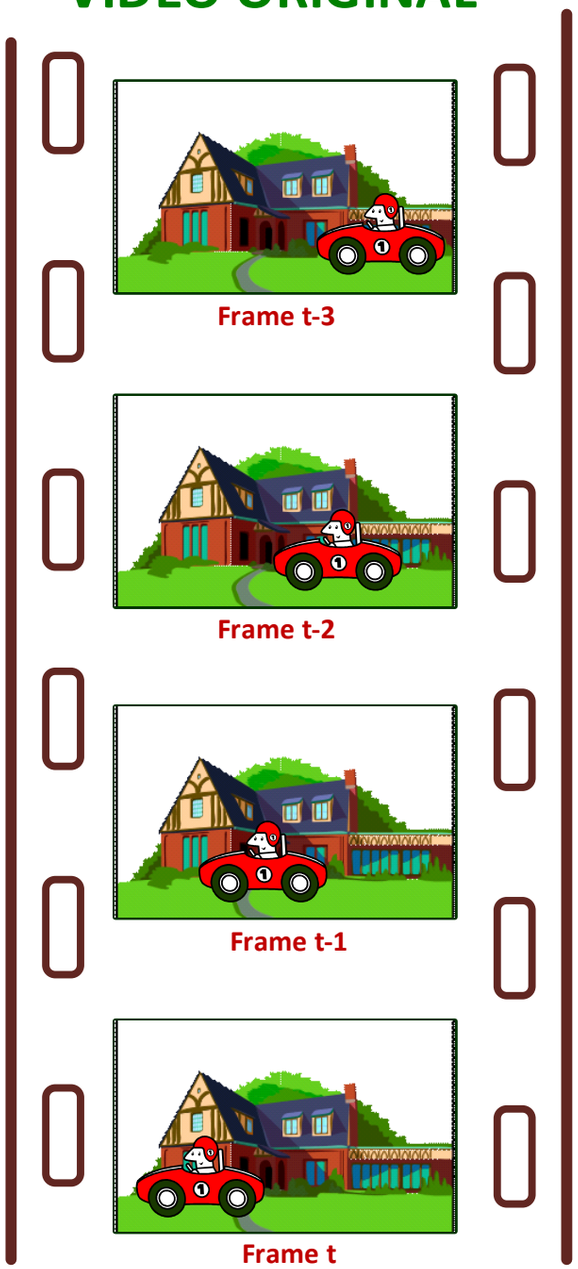
https://www.youtube.com/watch?v=bU2vpx-_j7I

EFECTO DE FRAME-RATE

https://www.youtube.com/watch?v=j9x7y4ZpmoY

VIDEO PROGRESIVO Y ENTRELAZADO

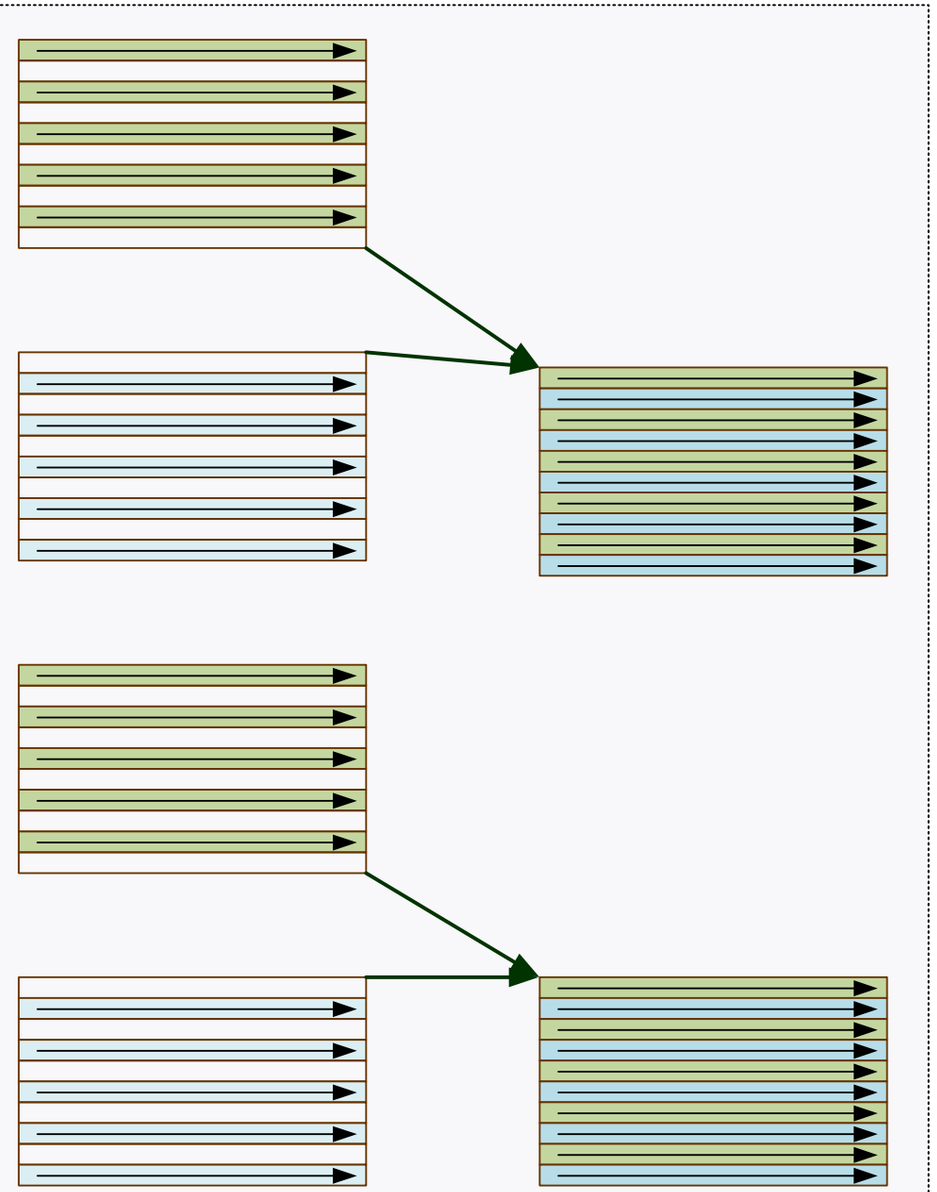
VIDEO ORIGINAL



PROGRESIVO

CAPTURA
Y
REPRODUCCIÓN
Frame Rate x1

ENTRELAZADO



CAPTURA
Frame Rate x2

REPRODUCCIÓN
Frame Rate x1

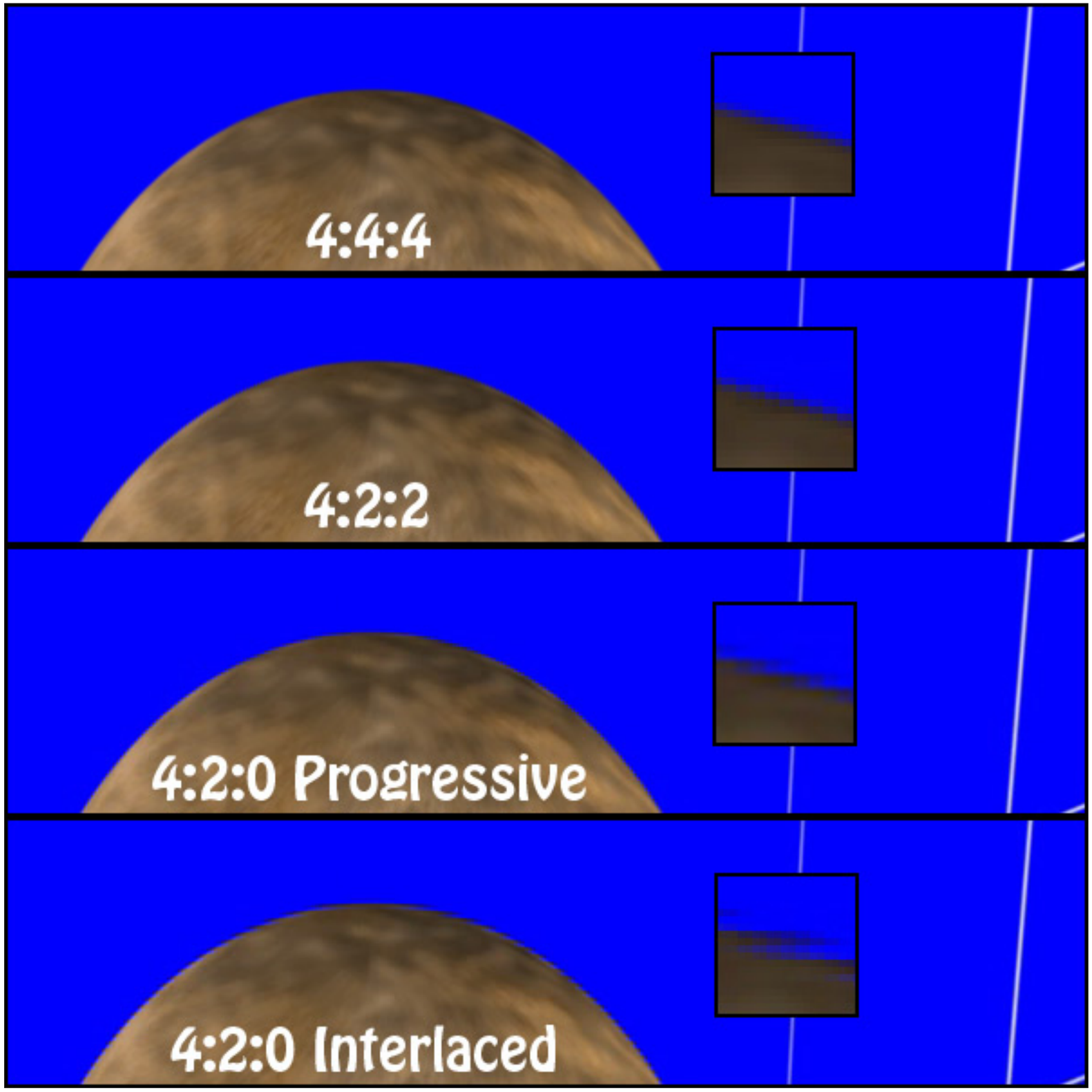
- **El video entrelazado (interlaced video) es una técnica que permite duplicar la percepción del frame rate sin consumir ancho de banda (bandwidth) extra. Contiene dos campos capturados en dos tiempos diferentes.**

- **Como ventaja, logra mejorar la perfección del receptor especialmente en las imágenes con mucho movimiento. No obstante, necesitan de un proceso de deinterlacing para eliminar efectos de aliasing.**

deinterlacing

Video entrelazado versus progresivo

https://www.youtube.com/watch?v=LV8UDBsf45Q



http://community.avid.com/forums/p/37441/209742.aspx#209742

FORMATOS DE VIDEO INTERMEDIOS

Formato	Resolución de la *Luma* (H×V)	Bits por *Frame* (4:2:0 y 8 *bps*)
Sub-QCIF	128 × 96	147456
QCIF (1/4 CIF)	176 × 144	304128
CIF (*)	352 × 288	1216512
4CIF	704 × 576	4866048

(*) CIF: *Common Intermediate Format*

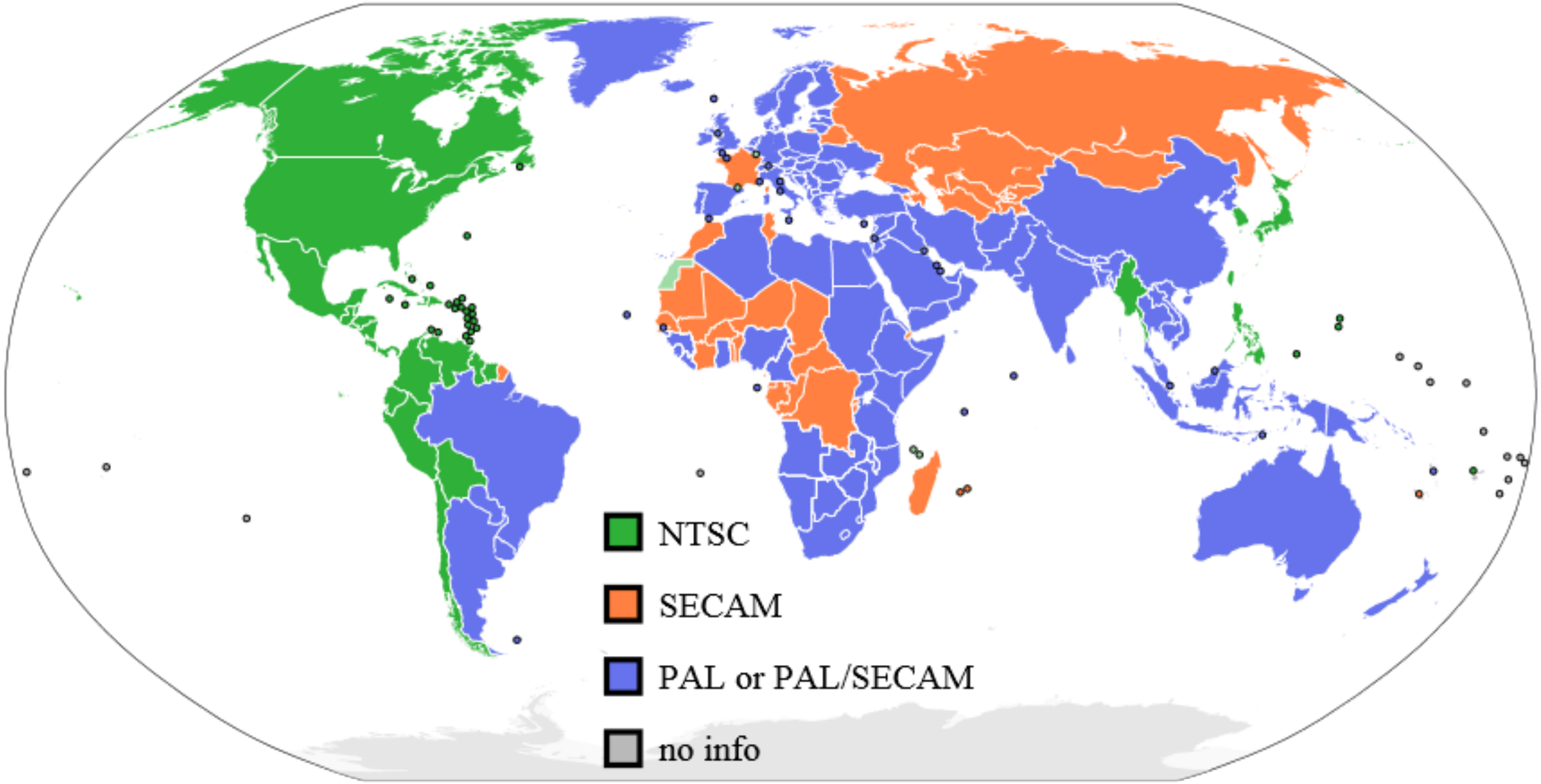
ESTANDARES DE VIDEO ANALÓGICO

- **NTSC** (***National Television System Committee***).
  Relación 4:3. 525 líneas. Frame rate 59.94.

- **PAL** (***Phase Alternating Line***).
  Relación 4:3. 625 líneas. Frame rate 50.

- **SECAM** (***Séquentiel Coulour à mémorie***).
  Relación 4:3. 625 líneas. Frame rate 50.

- **HDTV** (***Tamaño es > 720 pixels***).
  Más común 1280x720 (720p60 o 720p) y 1920x1080 (1080i30 o 1080i). Relación 16:9.

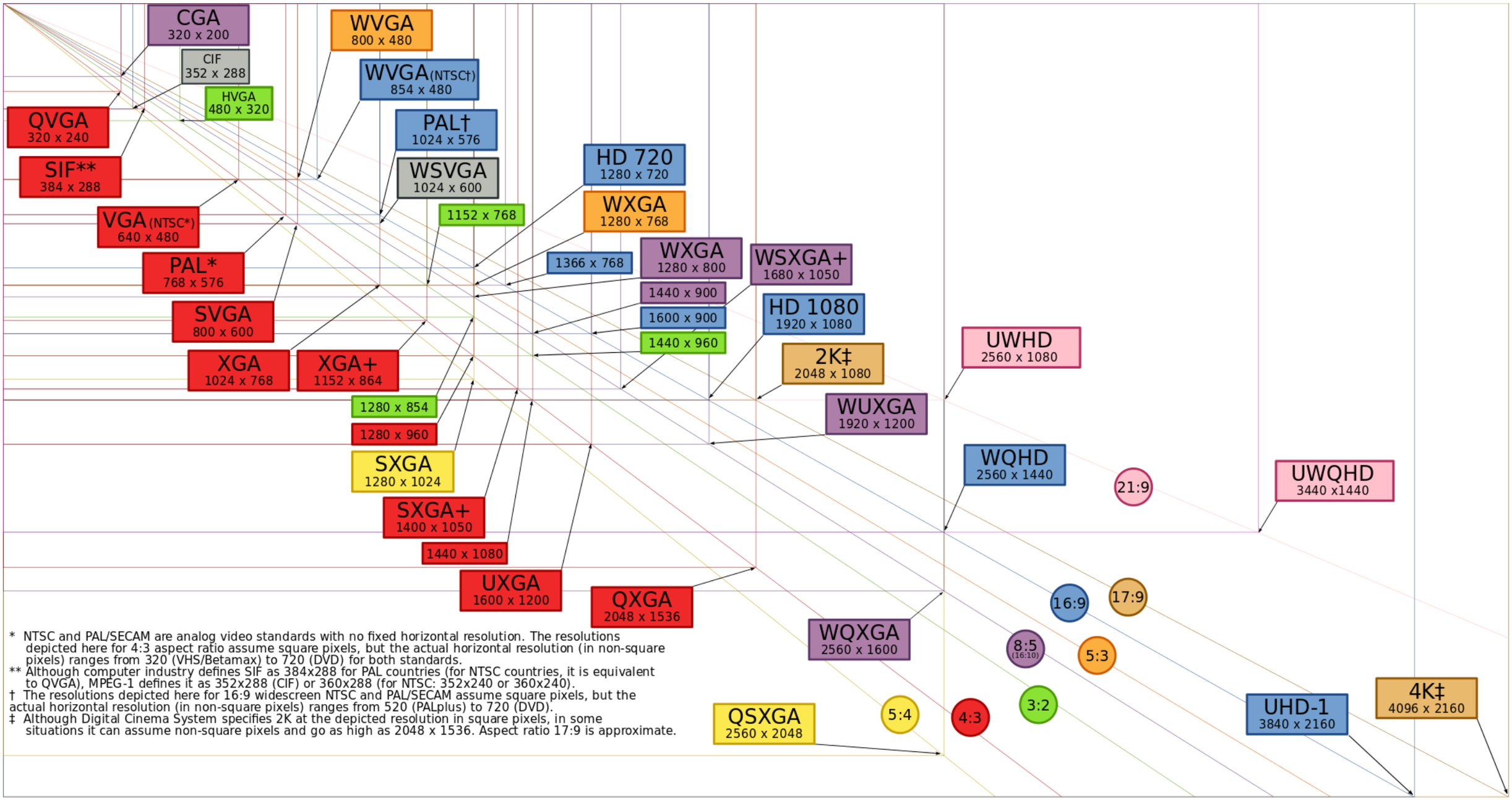


https://en.wikipedia.org/wiki/PAL

FORMATOS DIGITALES DE VIDEO PARA TV

- **Definición**: estándar (SD) o alta definición (HD).
- **Muestreo**: progresivo (p) o entrelazado (i).

Formato	*Luma* (H × V)	Rel. Aspecto
SD (30 Hz)	720 × 480	4 / 3
SD (25 Hz)	720 × 576	4 / 3
HD 720p	1280 × 720	16 / 9
HD 1080i	1920 × 1080	16 / 9
HD 1080p	1920 × 1080	16 / 9



* NTSC and PAL/SECAM are analog video standards with no fixed horizontal resolution. The resolutions depicted here for 4:3 aspect ratio assume square pixels, but the actual horizontal resolution (in non-square pixels) ranges from 320 (VHS/Betamax) to 720 (DVD) for both standards.
** Although computer industry defines SIF as 384x288 for PAL countries (for NTSC countries, it is equivalent to QVGA), MPEG-1 defines it as 352x288 (CIF) or 360x288 (for NTSC: 352x240 or 360x240).
† The resolutions depicted here for 16:9 widescreen NTSC and PAL/SECAM assume square pixels, but the actual horizontal resolution (in non-square pixels) ranges from 520 (PALplus) to 720 (DVD).
‡ Although Digital Cinema System specifies 2K at the depicted resolution in square pixels, in some situations it can assume non-square pixels and go as high as 2048 x 1536. Aspect ratio 17:9 is approximate.

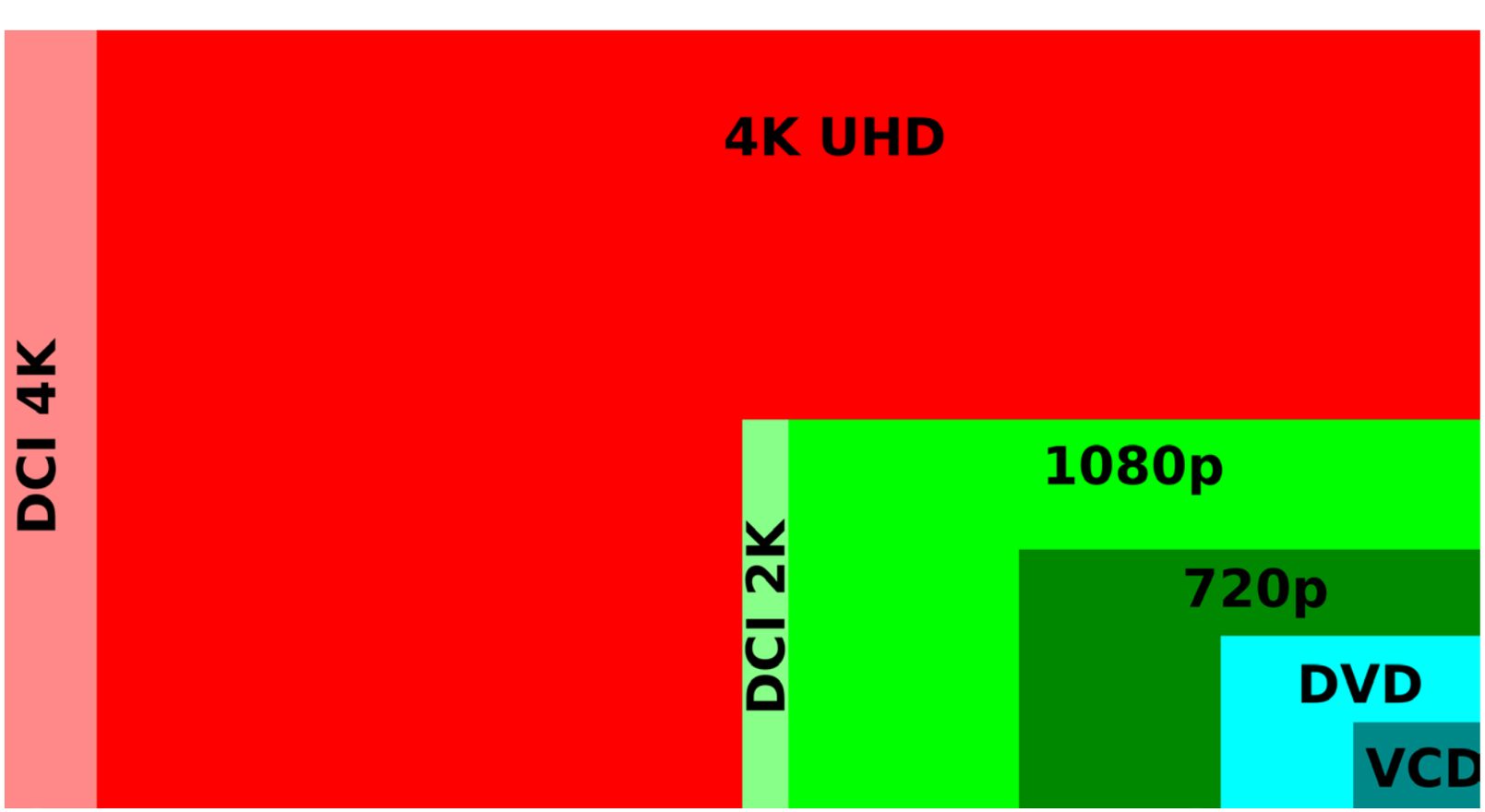
https://en.wikipedia.org/wiki/Display_resolution

RESOLUCIÓN 4K

Los dos tipos de resoluciones 4K:

- **DCI 4K: 4096 × 2160 (unos 8.8 megapíxeles) . Pantallas de los cines e infografía. Ratio: 17:9. 24 fps. Profundidad de color 8 bits.**

- **TV UHD (Ultra Alta Definición): 3840 x 2160 (unos 8.3 megapíxeles). Pantallas de TV. Ratio:16:9. 50 o 60 fps. Profundidad de color hasta 12 bits.**



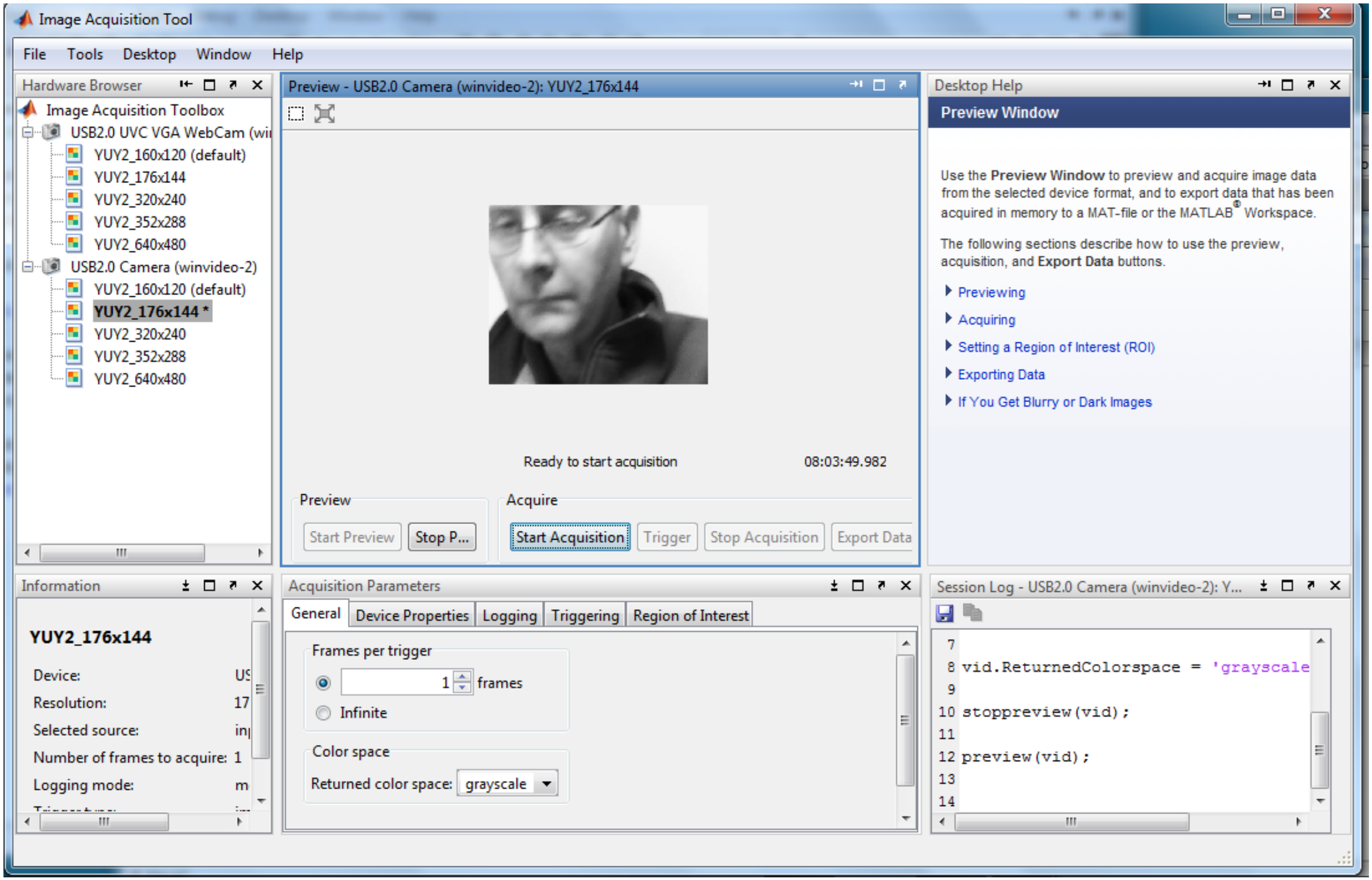
SD
480p

Full HD
1080p

4k Ultra HD
2160p

https://technologyformedia.com/2013/01/10/how-to-get-your-photography-and-videography-ready-for-ultra-hd-4k/

“IMAGE ACQUISITION TOOL” DE MATLAB (imaqtool)



PROCESAMIENTO DE VÍDEO CON MATLAB

- **Operaciones básicas**
- **Videoclips de duración reducida**
- **Número limitado de formatos de vídeo**

➢ Lectura de Ficheros de Vídeo

`VideoReader`: crea un objeto para leer ficheros de video, cuyos campos contienen sus características; soporta diversos formatos multimedia.

`read(objeto)`: crea un *array* de vídeo 4D.

➢ Manejo de Fotogramas:

`frame2im`: convierte un *frame* en imagen.
`im2frame`: convierte una imagen en un *frame*.

➢ Reproducción de Ficheros de Vídeo:

`movie`: reproductor de video primitivo.
`implay`: reproductor de video e imagen.

➢ Escritura de Ficheros de Vídeo:

`VideoWriter`: junto con `open` y `close`, permite crear ficheros de vídeo; soporta diversos formatos.
`writeVideo`: escribe datos de vídeo en un fichero.

➢ Lectura de Información de Ficheros de Vídeo

- **Creación de un objeto con información de vídeo.**

```
fichero_in='HummBird.avi';
infovideo=VideoReader(fichero_in);
```

- **Visualización de la información de los campos.**

```
infovideo.Height
infovideo.Width
infovideo.FrameRate
infovideo.Duration
infovideo.VideoFormat
```

- **Número de Fotogramas.**

```
L=infovideo.Duration*...
            infovideo.FrameRate
```

➢ Lectura de los Datos de un Fichero de Vídeo

- Carga de los datos de la película en una *array* 4D.

```
fotogramas=read(infovideo);
```

- Creación de una estructura *movie*.

```
sDriv=struct('cdata',...
             zeros(M,N,Z,'uint8'),...
             'colormap',[]);
```

- Carga de un *frame* en una estructura *movie*.

```
sDriv(k).cdata=read(infovideo,k);
```

➢ Visualización de Vídeos

- Frames Individuales

```
imshow(sDriv(k).cdata)
```

- Película Completa

```
implay(sDriv) o implay(fotogramas)
```

- **Procesado Individual de Fotogramas**

```
nDriv=sDriv;
for k=1:L
   fra=frame2im(nDriv(k));
   nfr=imcomplement(fra);
   nDriv(k)=im2frame(nfr);
end
```

- **Creación de una Estructura de Vídeo con immovie**

```
nfot = uint8(zeros(M,N,Z,L));
for i = 1:L
   temp = fotogramas(:,:,:,i);
   n_fot(:,:,:,i) =imcomplement(temp);
end
n_Mov = immovie(n_fot);
implay(n_Mov)
```

➢ Escritura de un Fichero de Vídeo AVI

```
fich_out='N_HummBird.avi';
n_info=VideoWriter(fich_out);
open(n_info);
writeVideo(n_info,n_Mov);
close(n_info);
```

Lectura y Visualización

```
n_HB=read(VideoReader(fich_out));
implay(n_HB)
```

➢ Lectura y Visualización de Vídeo MPEG-4

```
idriv=VideoReader('Driving.mp4');
L=idriv.Duration*idriv.FrameRate;
driv_4D=read(idriv);
implay(driv_4D)
```

➢ Lectura de Vídeo YUV

```
file='Whale_Show.yuv';
format='NTSC';
nfs=50; % número de fotogramas
[mov,frms]= readYUV(file,nfs,format);
```