La vidéo numérique en pratique

Du pixel à l’écran

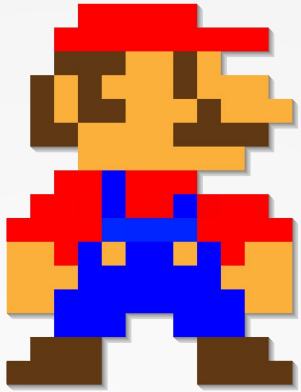
Description d’un pixel
Structure d’une image vidéo
Cadences
Entrelacement
Affichage

La vidéo numérique en pratique

Description d’un pixel

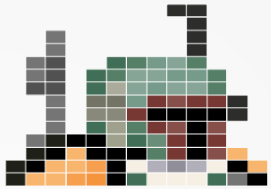
Pixel

- Élément d’une image
- Mais encore?
  - Format
  - Profondeur (Depth)
  - Espace de Couleur (Color Space)



Pixel : Aspect Ratio

- Monde PC : PAR = 1:1
  - Pixels carrés
- Monde vidéo : PAR >= 1
  - Pixels rectangulaires
- Pourquoi ?
  - Analogique : pas de résolution
  - Numérique : résolutions partout
  - Besoin de correspondance numérique ↔ analogique
  - **PAR x DIM = DAR**



Pixel : Color Space

- Générateurs de graphismes RGB
  - ordis, smartphones, tablettes, OSDs
  - ARGB, ABGR,
  - BGRA, RGBA...
- Vidéo : YUV
  - Images et séquences animées
  - Y : luminance
  - U, V : Chrominance (aka Cb, Cr)
  - Raisons historiques

Exemple RGB

R G B

Exemple YUV

Faible dynamique

U

Y

Faible dynamique

V

Pixel: formats

- Format linéaire:
  - Toute l’info est dans le pixel
  - Profondeur: 8..16 bits / composante
  - 8 bpc : « True Color ». ARGB, ARGB, ARGB...
  - Exemple : #FF00FF00 ?
  - Exotiques : High Color : 15 / 16 bits : 555, 565, 1555

La vidéo numérique en pratique

True Color,

Considérations système

- Que faut-il dans un système ?
- Horloge Système
- CPU
- RAM
  - Efficacité
  - Bande passante (BW)
- BUS DMA
    - Taille
    - BW
- Modules hardware
  - Accélérateurs (GPU)
  - Autres modules d’affichage

CCC Full HD True Color ?

- Image fixe RGB 8bpc @1080p60
- BW affichage : 1920 x 1080 x 60 x 24 = 2,98 Gb/s
- Pour un système:
  - Bus clock : 200 MHz
  - DRAM : DDR3 12800 (8 bytes / burst, classiquement)
    - 8 bursts / clock
    - ⇒ 64 bytes / clock
    - 80 % d'efficacité
    - ⇒ BW RAM = 200M x 64 x 8 x 0,8 = 81,92 Gb/s
  - DMA : 32 bits / clock
    - ⇒ BW bus = 200M x 32 = 6,4 Gb/s
  - ⇒ **~3,88 % BW ram**
  - ⇒ **~46 % BW bus**
  - **Juste pour afficher !**
  - **x2 : NOK. Pas d'animation possible**

CCC Full HD True Color DMA++ ?

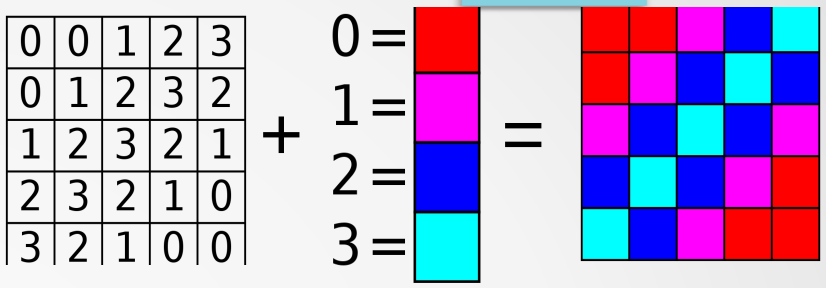
- Image fixe RGB 8bpc @1080p60
- BW affichage : 1920 x 1080 x 60 x 24 = 2,98 Gb/s
- Pour un système:
  - Bus clock : 200 MHz
  - DRAM : DDR3 12800 (8 bytes / burst, classiquement)
    - 8 bursts / clock
    - ⇒ 64 bytes / clock
    - 80 % d'efficacité
    - ⇒ BW RAM = 200M x 64 x 8 x 0,8 = 81,92 Gb/s
  - **DMA : 64 bits / clock**
    - ⇒ **BW bus = 200M x 64 = 12,8 Gb/s**
  - ⇒ **~3,88 % BW ram**
  - ⇒ **~23 % BW bus**
  - ⇒ **Image animée jouable**
  - ⇒ **Coûteux**

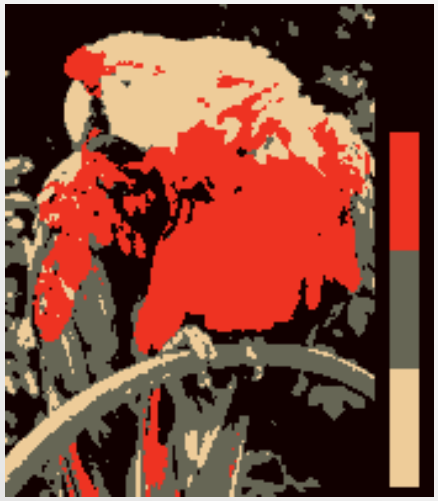
CCC 4K ?

- Image fixe RGB 10bpc @2160p60
- BW affichage : 3840 x 2160 x 60 x 30 = 14,92 Gb/s
- ~5x + !
- Que change-t-on ?
  - La taille du DMA ?
  - La clock système ?
    - **...donc la RAM?**
      - **...donc le contrôleur ?**

Pixel: formats

- Format palettisé :
  - Palette de couleur prédéfinie
  - Pixel = index couleur palette
  - Profondeurs : 1, 2, 4, 8 bits / pixel





2 bpp

4 bpp

8 bpp

CCC Full HD palettisé ?

- Image palettisée **256** couleurs @1080p60
- Index : **8** bits
- BW affichage : 1920 x 1080 x 60 x 8 = ~1 Gb/s
- **3x moins qu'en RGB interleavé**
- **Seulement 256 couleurs**
- **Parfait pour OSD**
- **Et sous-titres**

La vidéo numérique en pratique

Structure d’une image vidéo

Colorspace YUV

Faible dynamique

U

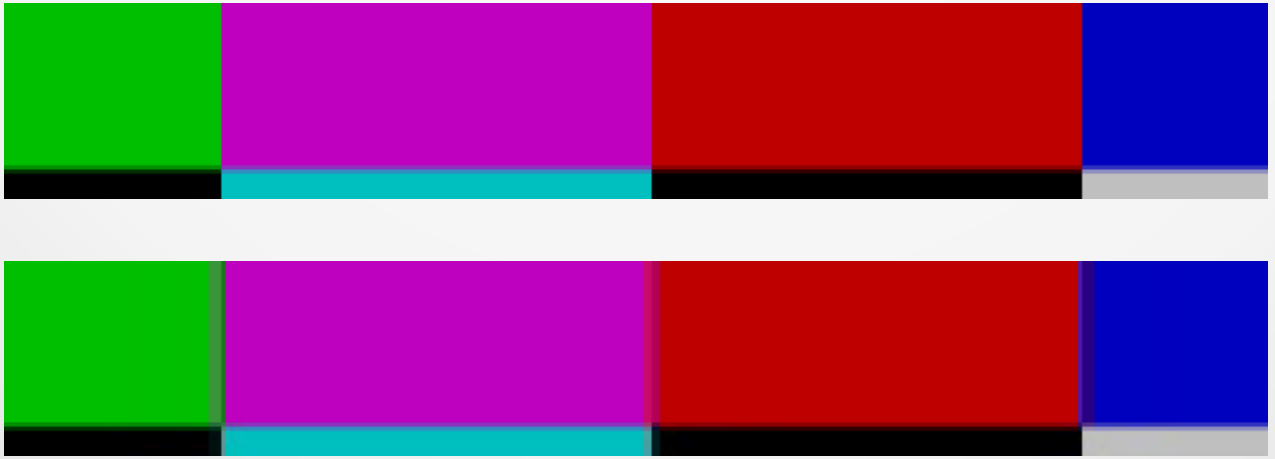
Y

Faible dynamique

V

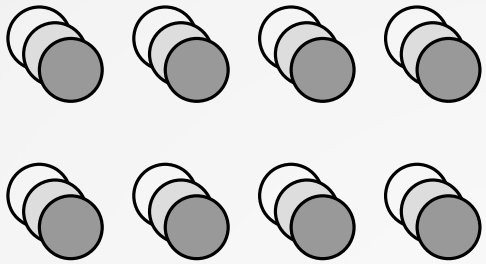
Sampling Mode

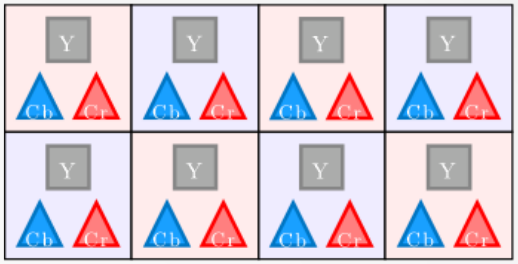
- Sampling Mode =
  Sous-échantillonnage de la chrominance
- Seulement en color space YUV
- Oeil moins sensible à U V
- Exemple :



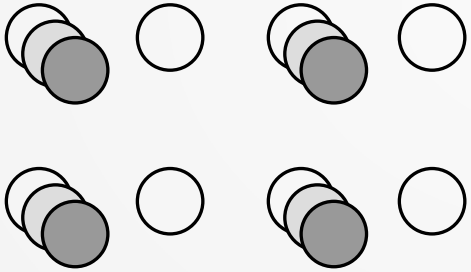
Sampling Mode

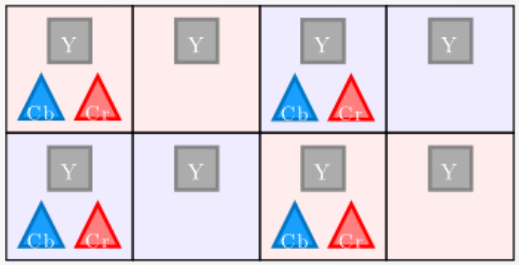
- 4:4:4



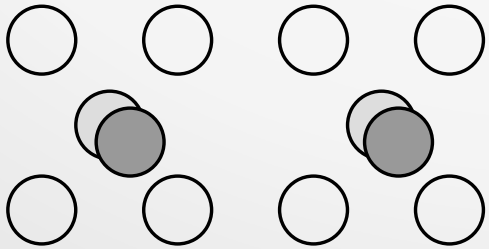


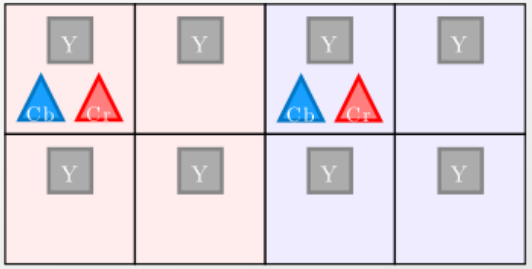
- 4:2:2





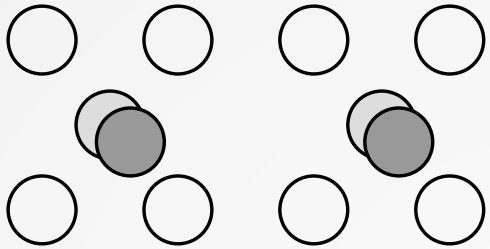
- 4:2:0 MPEG-1

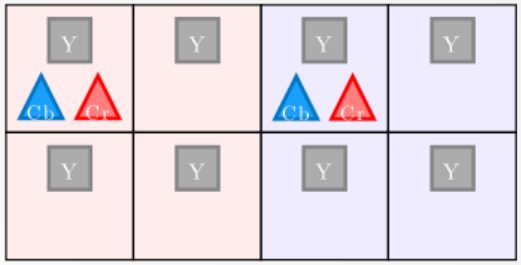




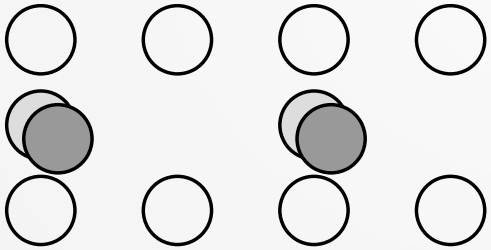
Sampling Mode

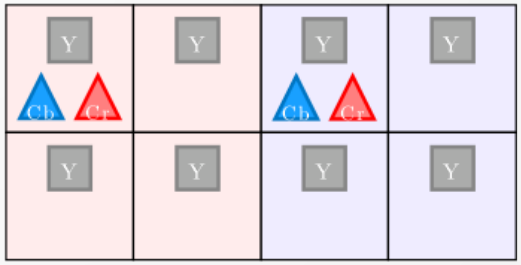
- 4:2:0 MPEG-1



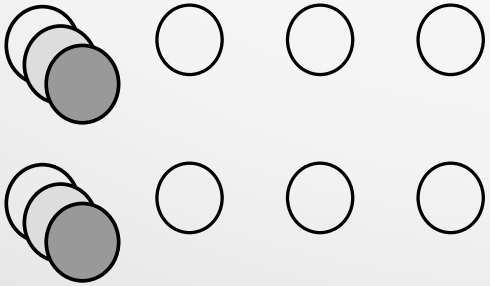


- 4:2:0 MPEG-2





- 4:1:1



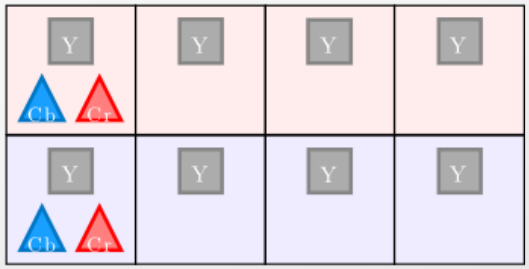


Image planaire

- Héritage TV : Y avant UV
- PC : Images interleavées (ARGB, ARGB…)
- Vidéo: Images planaires:
  - Buffers composantes séparés
  - Luma : Y,Y,Y...
  - Chroma : UV,UV,UV...
  - Découplage hardware / composantes

La vidéo numérique en pratique

Une image planaire,

CCC Full HD Planar 4:2:2 ?

- 1080p60 YUV 4:2:2 8 bits :
  - Luma : 1920 x 1080 x 60 x 8 = **~ 1 Gb/s**
  - Chroma : 1920x1080 x 60 x 16 **/ 2** = **~ 1 Gb/s**
- Pour un système:
  - Bus clock : 200 MHz
  - DRAM : DDR3 12800 (8 bytes / burst, classiquement)
    - 8 bursts / clock
    - ⇒ 64 bytes / clock
    - 80 % d'efficacité
    - ⇒ BW RAM = 200M x 64 x 8 x 0,8 = 81,92 Gb/s
  - **Deux DMA deux fois plus petits**: 16 bits / clock
    - ⇒ BW = 200M x 16 = **3,2 Gb/s/canal**
  - **⇒ ~2,5 % BW ram**
  - **⇒ ~31 % BW par canal**
  - **⇒ x2 OK : animation possible**

CCC Full HD Planar 4:2:0 ?

- 1080p60 YUV 4:2:0 8 bits : >90 % des fichiers vidéo (TNT, SAT, YT...)
  - Luma : 1920 x 1080 x 60 x 8 = ~ 1 Gb/s
  - Chroma : 1920x1080 x 60 x 16 **/ 4** = **~ 0,5 Gb/s**
- Pour un système:
  - Bus clock : 200 MHz
  - DRAM : DDR3 12800 (8 bytes / burst, classiquement)
    - 8 bursts / clock
    - ⇒ 64 bytes / clock
    - 80 % d'efficacité
    - ⇒ BW RAM = 200M x 64 x 8 x 0,8 = 81,92 Gb/s
  - Deux DMA deux fois plus petits: 16 bits / clock
    - ⇒ BW = 200M x 16 = 3,2 Gb/s/canal
  - ⇒ ~2,5 % BW ram
  - **⇒ ~31 % BW Y, 15% BW UV**

CCC 4K Planar 4:2:0 10 bits ?

- 2160p60 YUV 4:2:0 10 bits : Netflix
  - Luma : 3840 x 2160 x 60 x 10 = ~ 4,97 Gb/s
  - Chroma : 3840 x 2160 x 60 x 20 **/ 4** = **~ 2,49 Gb/s**
- Pour un système:
  - Bus clock : 200MHz
  - DRAM : DDR3 12800 (8 bytes / burst, classiquement)
    - 8 bursts / clock
    - ⇒ 64 bytes / clock
    - 80 % d'efficacité
    - ⇒ BW RAM = 200M x 64 x 8 x 0,8 = 81,92 Gb/s
  - **Deux DMA 64 bits / clock**
  - **Hardware à 2 px / clock**
    - ⇒ **BW = 200M x 64 = 12,8 Gb/s/canal**
  - ⇒ **~9 % BW ram**
  - ⇒ **~39 % BW Y, ~20 % BW UV**
  - **Autre choix : BUS 2x rapide**
  - ⇒ **Plus cher**

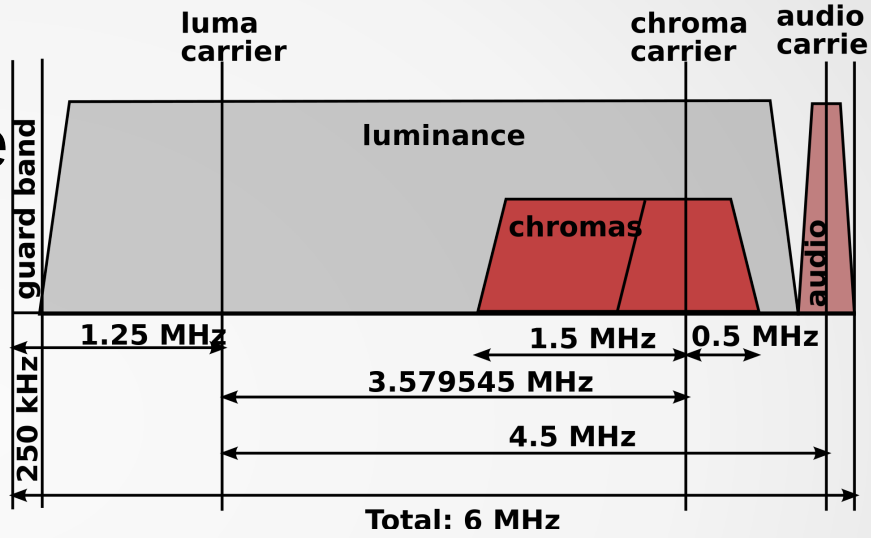
La vidéo numérique en pratique

Entrelacement
Désentrelacement

70 ans
Toutes ses dents

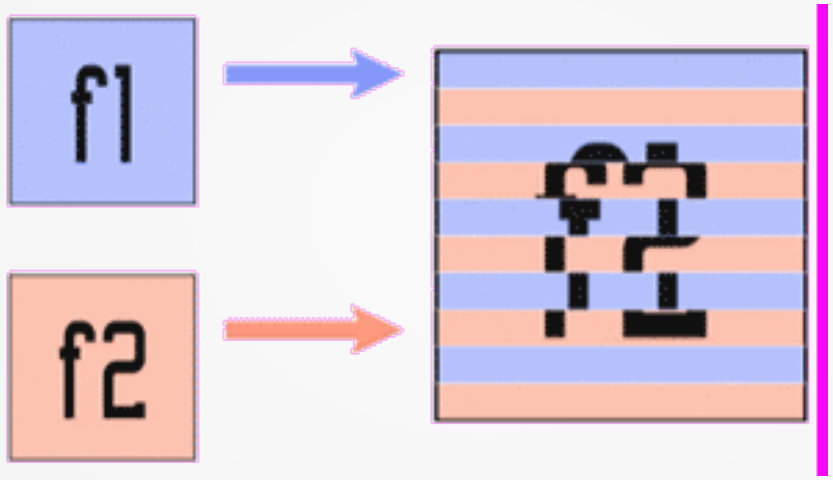
Un peu d’histoire

- 1941: standard NTSC
  - 6 MHz de bande passante
  - 15730 lignes/s
  - 525 lignes/image
  - 15730 / 525 = 30 ips
    - OK pour films (24 ips)
    - NOK pour du direct

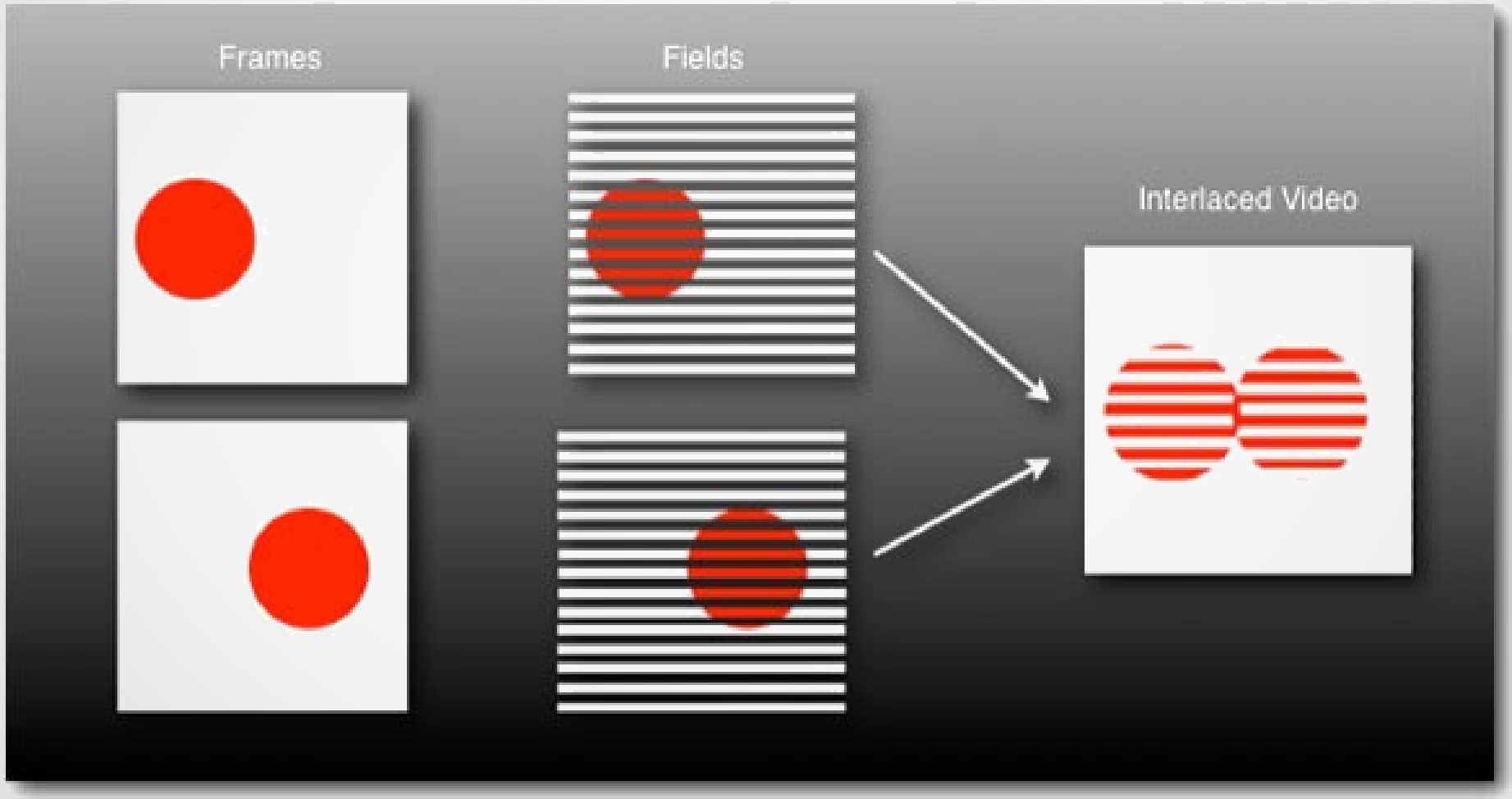


Une idée simple

- 1 frame : 2 fields : pair/impair
- capturés à des instants différents



Exemples



La vidéo numérique en pratique

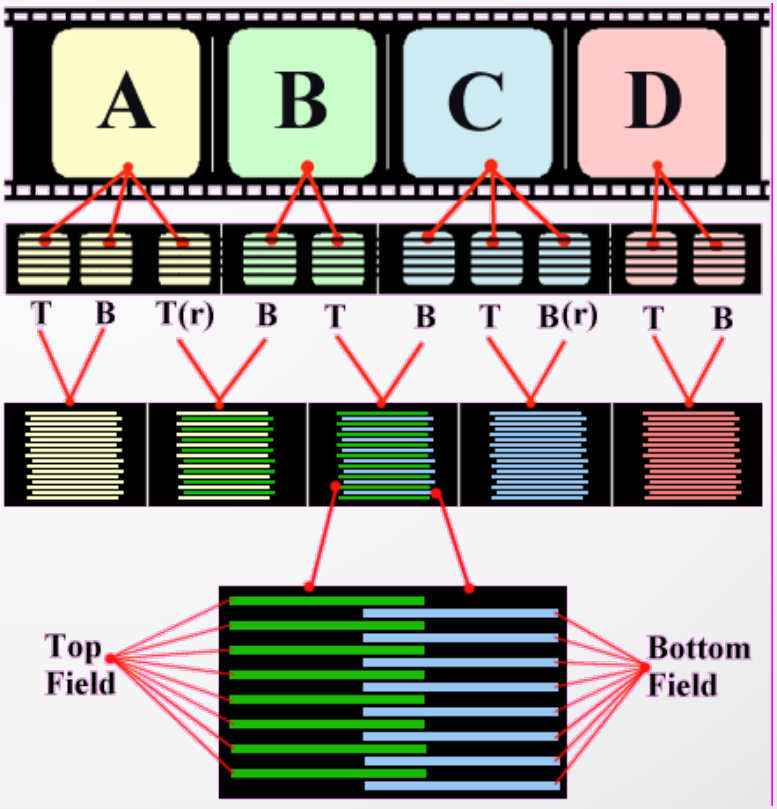
Exemples d’entrelacement
Vidéos illustratives

Exemples

- Avantages
  - Image fixe: résolution conservée
  - Image mouvante: 2x plus fluide
  - BW inchangée
  - Parfait pour tube cathodique
- Inconvénients
  - Résolution H / 2 si mouvement
  - Signaliser l’ordre des fields
  - Scintillement
  - Le futur devra faire avec...

Film et entrelacement

- Film = 24P, NTSC = 60I
- Comment passer un film à la télé ?
  - 3:2 pulldown
  - 4 frames, 10 fields
  - A : TFF +RFF
  - B : BFF
  - C : BFF + RFF
  - D : TFF

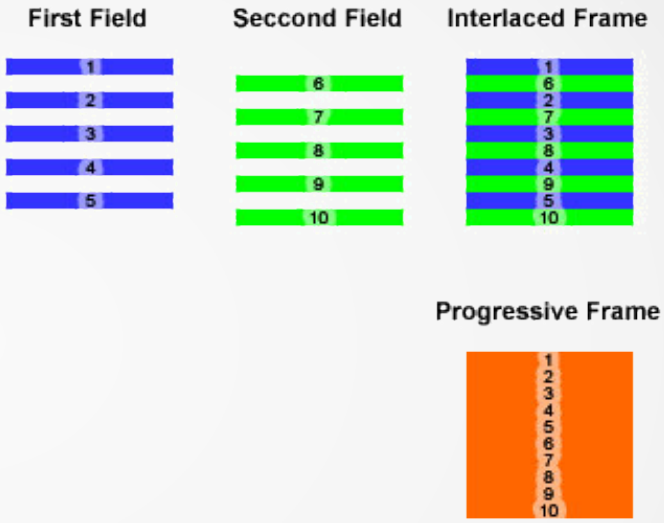


La vidéo numérique en pratique

Désentrelacement

Désentrelacement

- Reconstruire les lignes manquantes
- Plusieurs approches
- Toutes inexactes
- Complexité variable
- En pratique : efficacité / coût

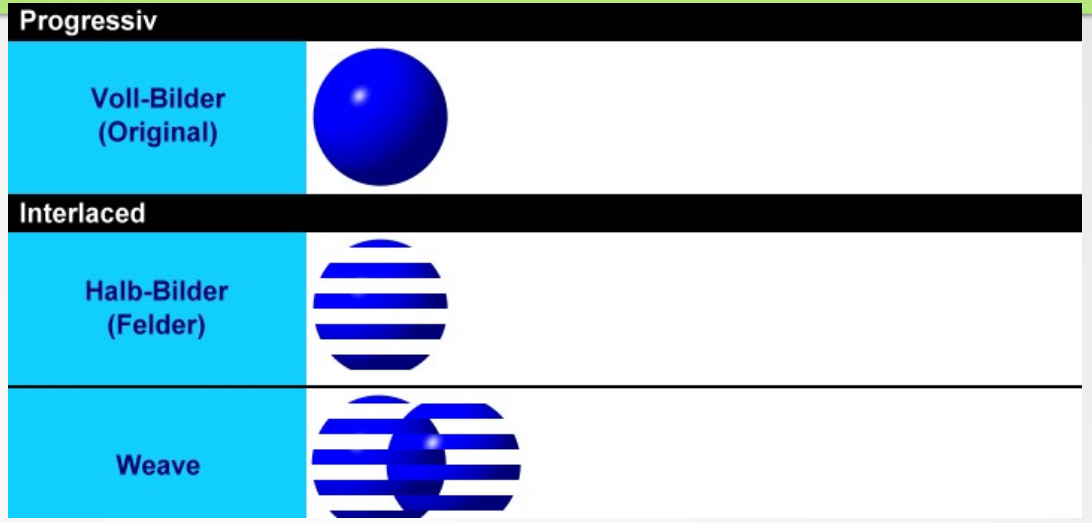


La vidéo numérique en pratique

Désentrelacer, comment ça marche,

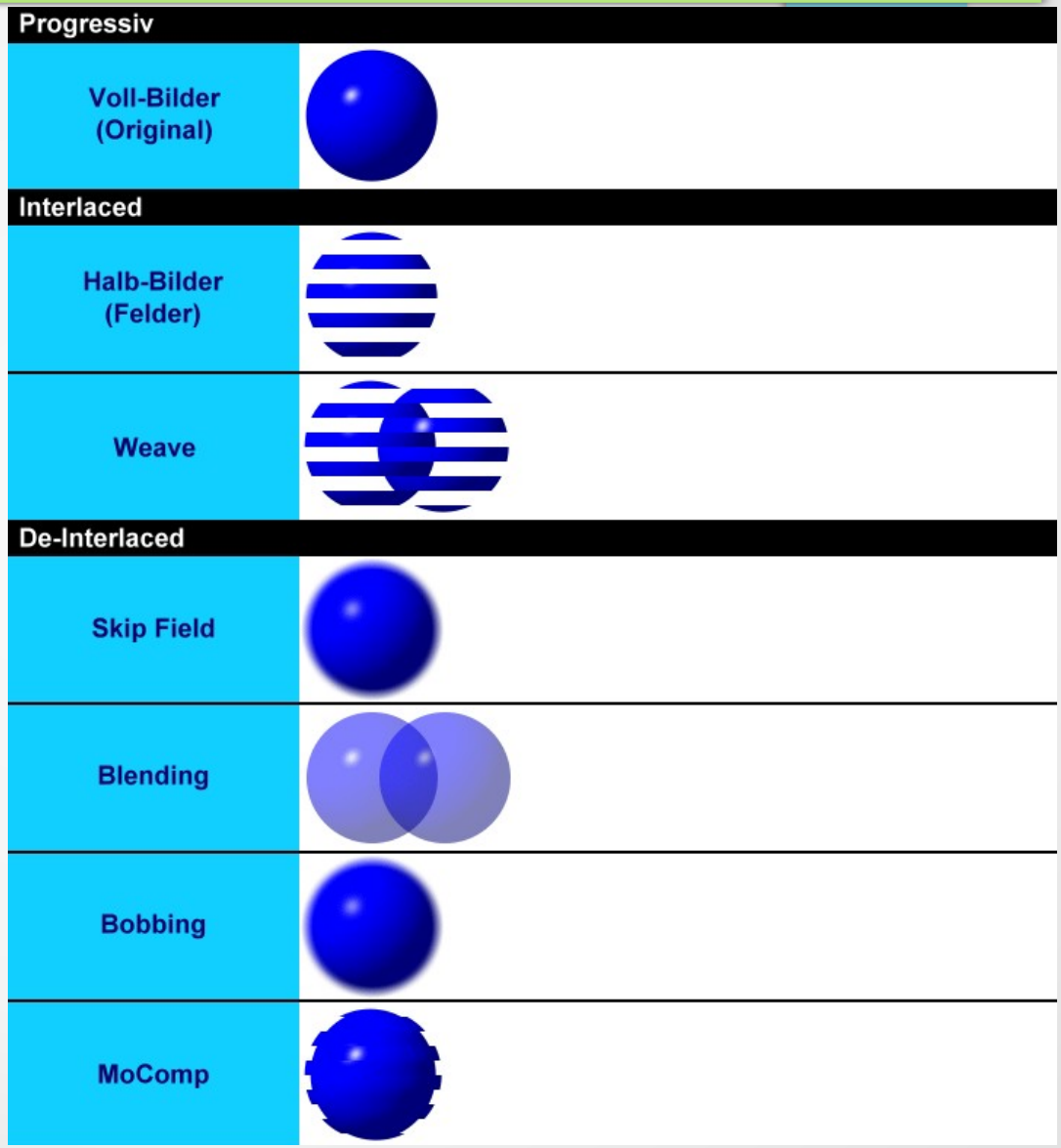
CCC désentrelacer?

- **Weave** :
  - Ne rien faire
  - **$** = 1:1
  - **Inadmissible**



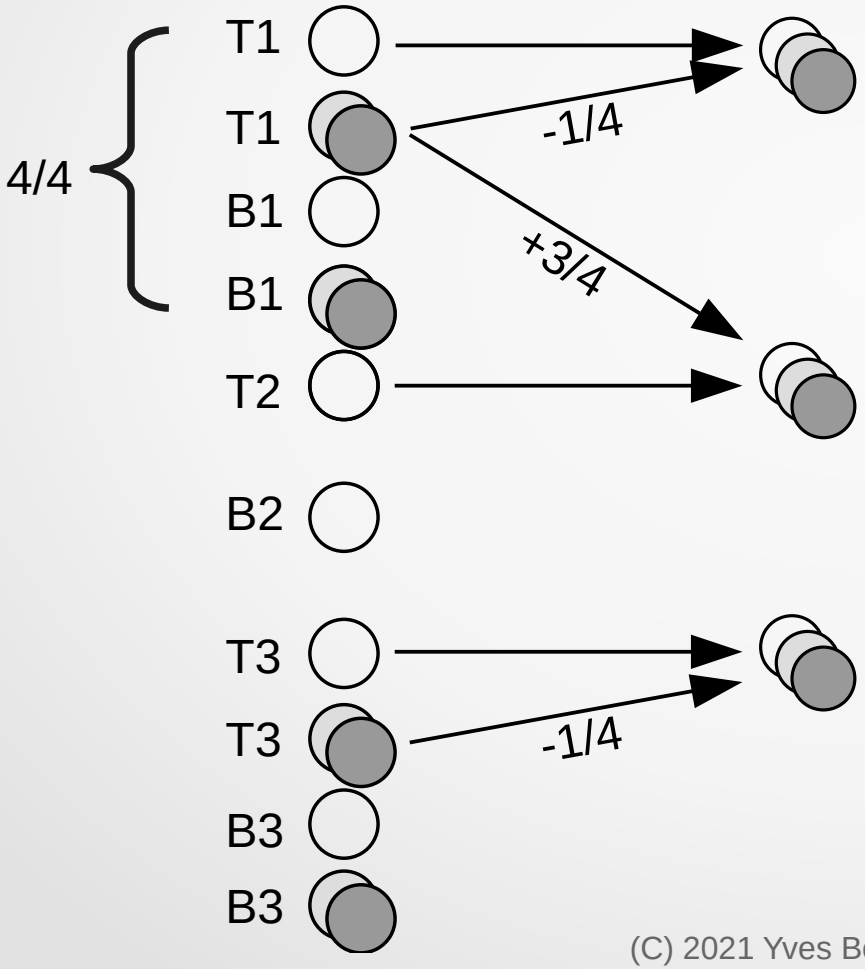
CCC désentrelacer?

- **Weave** :
  - Lire une frame (a)
  - $ = 1:1 (a)
- **Skip field :**
  - Lire 2x le même field (a)
  - Si field bottom → aligner frame (b)
  - Upscaler verticalement x2 (c)
  - $ = 1:1 (a) + filtrage (b) + upscale (c)
- **Bobbing** :
  - Lire chaque field 1/1 (a)
  - Si field bottom → aligner frame (b)
  - Upscaler verticalement x2 (c)
  - $ = 1:1 (a) + filtrage (b) + upscale (c)
- **Blending (Microsoft) :**
  - Lire une paire de fields (a)
  - Aligner fields bottom (b)
  - Upscaler verticalement x2 (c)
  - Mélanger les frames (d)
  - $ = 2:1 (a) + 2x filtrage (b) + 2x upscale (c) + mixage (d)
- **Adaptative (Spatial, MoComp)**



Alignement

- **Image entrelacée 4:2:0 MPEG-2**
- Conversion field **TOP 4:2:0** ➡ **TOP 4:4:4**



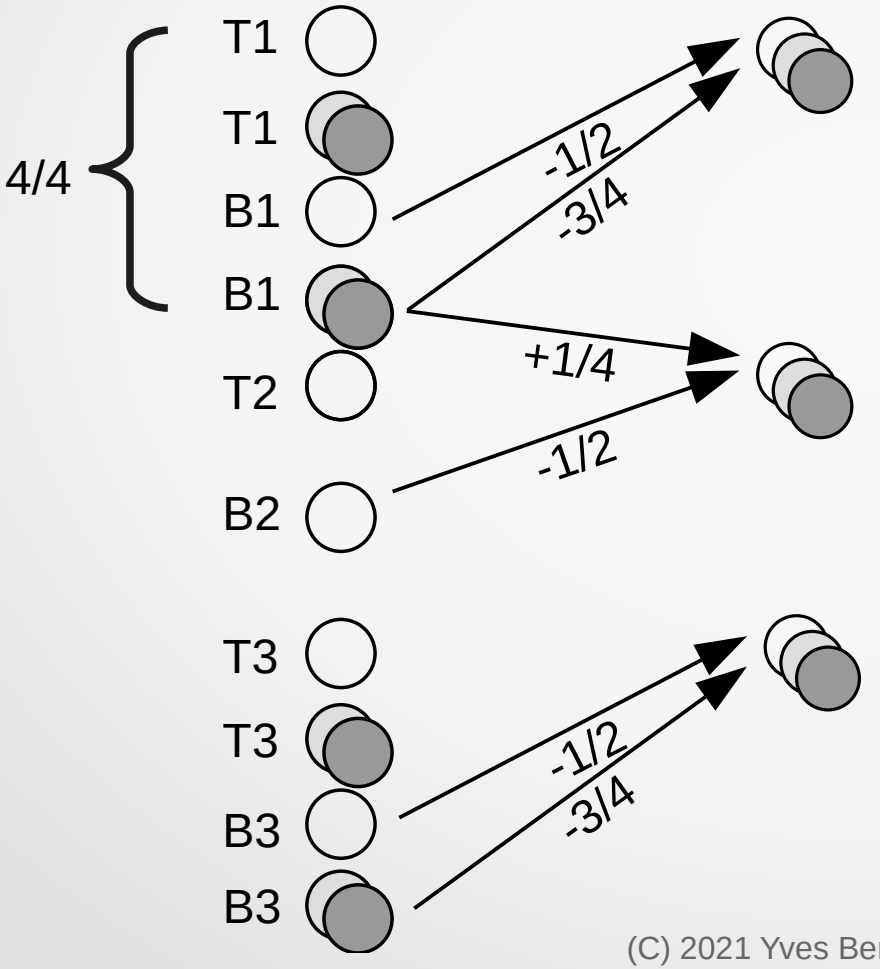
Ligne TOP avec Chroma:
Déphasage Luma : 0
Déphasage Chroma: -1/4

Ligne TOP SANS Chroma:
Déphasage Luma : 0
Déphasage Chroma: +3/4

Etc.

Alignement

- **Image entrelacée 4:2:0 MPEG-2**
- Conversion field **BOT 4:2:0** ➡ **TOP 4:4:4**



Ligne BOT avec Chroma:
Déphasage Luma : -1/2
Déphasage Chroma: -3/4

Ligne BOT SANS Chroma:
Déphasage Luma : -1/2
Déphasage Chroma: +1/4

Etc.

Upscaling

- **Agrandir une image source à faible résolution**
- ⇒ Créer des pixels (suréchantillonnage 2D)
- Algorithmes variés
- Tous imparfaits
- Compromis qualité / prix (HW, CPU, RAM)

Upscaling NN

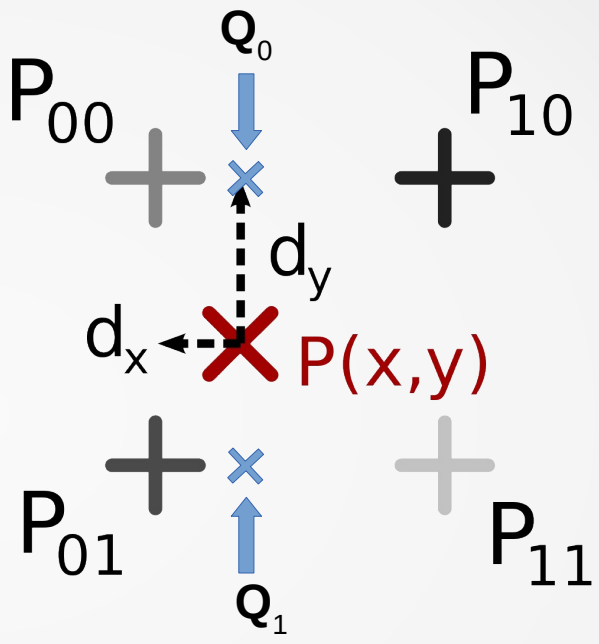
- **Nearest Neighbour / Plus Proche Voisin**
- Répéter le pixel... voisin

- Rapide
- Gratuit
- **Moche**

Upscaling BF

- **Bilinear Filtering**
- Interpolation linéaire 2D
- Trois interpolations 1D :
  - entre **P**$_{00}$ et **P**$_{10}$: **Q**$_0$
  - entre **P**$_{01}$ et **P**$_{11}$: **Q**$_1$
  - entre **Q**$_0$ et **Q**$_1$: **P**

$$Q_0 = (1 - d_x)P_{00} + d_x P_{10}$$
$$Q_1 = (1 - d_x)P_{01} + d_x P_{11}$$
$$P(x,y) = (1 - d_y)Q_0 + d_y Q_1$$



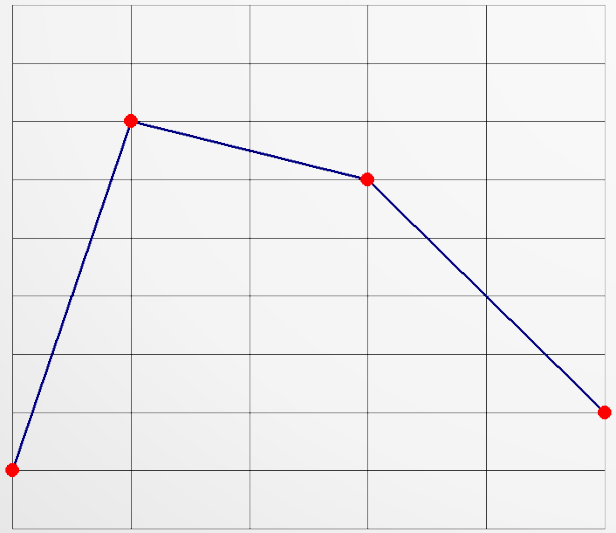
Upscaling BF

- **Bilinear Filtering**
- Interpolation linéaire 2D

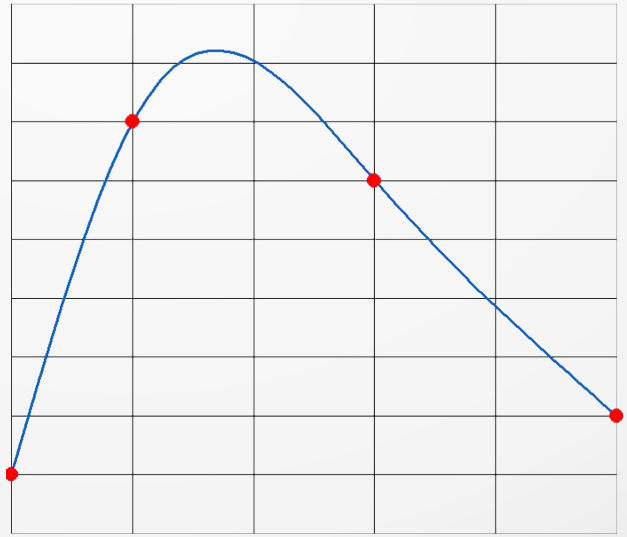
- Acceptable
- Pas cher
- Pas pour les gros ratios (SD ⇒ 4K)

Upscaling B-Spline

- **B-Spline interpolation**
- Spline : fonction définie par la jonction de plusieurs polynômes
- 2D : Points de passage: pixels
- B-Spline : Contrainte de continuité entre les polynômes



B-Spline d'ordre 1



B-Spline d'ordre 3

Upscaling B-Spline

- **B-Spline Filtering**
- Très souvent : B-Splines cubiques

- Bons résultats
- **SD → 4K : OK**
- **Cher**
- **Risqué (ringing)**

CCC désentrelacer?

- **Adaptatif spatial pur:**
  - Plusieurs fields : B0, **T1**, B1
  - Découpés en zones de pixels : Zi
  - Pour chaque Zi :
    - « différence » entre B1 et B0 (parité)
    - Ei ~= Zi(B1) – Zi(B0)
    - Ei > seuil : Zi « en mouvement »
      - ⇒ pixels Zi(**T1**) bobbés
    - Sinon : Zi « statique »
      - ⇒ pixels Zi(**T1**) weavés avec Zi(B0)
    - Avantages :
      - Si mouvement : Zi(**T1**) désentrelacée
      - Sinon: Zi(**T1**) pleine résolution
      - Rendu acceptable
    - Inconvénients :
      - Certains mouvements indétectables (translations, recouvrements)
  - **$** = 3:1 + différenciateur + bob + multiplexeur

CCC désentrelacer?

- **Amélioration :**
  - 4 fields : **T0,** B0, **T1**, B1
  - Ei = max(Zi(B1) – Zi(B0), + **Zi(T1) - Zi(T0)**)
  - Meilleure détection de « mouvement »
  - **$** = 4:1 + 2x différenciateur + bob + multiplexeur
- **Adaptatif temporel:**
  - Spatial + estimateurs de mouvement convolutifs
  - FIR / Turbo
  - Cuisine secrète brevetée (Faroudja / STMicro...)
  - Plus beau
  - **BEAUCOUP PLUS CHER**

La vidéo numérique en pratique

Cadence image

1001 vitesses

Cadence Image

- Image animée : quelle fréquence choisir?
- Cinéma :
  - Muet: 16 ips
  - Parlant : 24 ips
- TV :
  - Synchroniser caméras et TVs
  - « Horloge commune » ?
  - ⇒ Fréquence secteur !
  - US, JP : 60 ips (NTSC, 1941)
  - EMEA : 50 ips (bien avant PAL/SECAM)

Cadence Image

- US : 1953 : NTSC en **couleur**
- Interférences image / son à 60 ips
- Solution : changer la fréquence image
- **X 1000/1001 !**
  - 60 ips ⇒ 59,94 ips
  - 30 ips ⇒ 29,97 ips
  - 24 ips ⇒ 23,978 ips
- Transparent,économique, **pénible**
- « Modes TV », « Modes PC »

Cadences US

- US : 60 ips
- 60 ⇒ 30 : sauter 1 image sur 2
- 30 ⇒ 60 : répéter 1 image sur 2
- 24 ⇒ 30 : répéter 1 image sur 5
- 59,97 ⇒ 60 : répéter 1 image sur 1000
- 60 ⇒ 59,97 : sauter 1 image sur 1000
- 29,97 ⇒ 60 : répéter 1 image sur 2 **et 1 fois sur 1000**
- 60 ⇒ 29,97 : sauter 1 image sur 2 **sauf 1 fois sur 1000**

Cadences EU

- EU : 50 ips
- 60 ⇒ 50 : sauter 1 image sur 6
- 50 ⇒ 60 : répéter 1 image sur 5
- 24 ⇒ 25 : répéter 1 image sur 24 ?
  - NON ! ⏩ accélérer 25 / 24 : +4 %
  - -2m20s / h
  - +1 demi-ton
- 30 ⇒ 50 : rapport 5 / 3. Répéter 2x fois la 3ème image ?
  - 1 2 3 **3** **3**...
  - **Saccadé.** Mieux ?
  - 1 **1** 2 **2** 3
  - Plus homogène ⇒ perception plus fluide

Cadences en pratique

- Tout est entier :
  - PTS : temps image source
  - STC : temps horloge affichage
- Résolution d'incrément : TIR
  - TIR(PTS) = durée d'une seconde dans le flux vidéo
  - TIR(STC) = durée d'une seconde à l'affichage
- Si TIR(PTS) non % TIR(STC),
  problème de **fraction continue !**

Cadences en pratique

- Exemple :
  - TIR PTS = 90000 = 1 seconde
- Pour un flux vidéo à 50 ips :
  - ΔPTS = temps image = 90000 / 50 = **1800** (TIR % fps)
- Pour un flux vidéo à 60 ips :
  - ΔPTS= 90000 / 60 = **1500** (TIR % fps)
- Pour un flux vidéo à... 59,94 ips :
  - ΔPTS = 90000 x **1001** / (**1000** x 60) = **1501,5**
- **Fraction continue : 1501,5 = 1501 + 1 / 2**
- S'en suivent des images avec les incréments suivants :
- | 1501 | **1502** | 1501 | **1502** | 1501 | **1502**...
- **La FC fait apparaître de la gigue aka JITTER**

Cadences en pratique

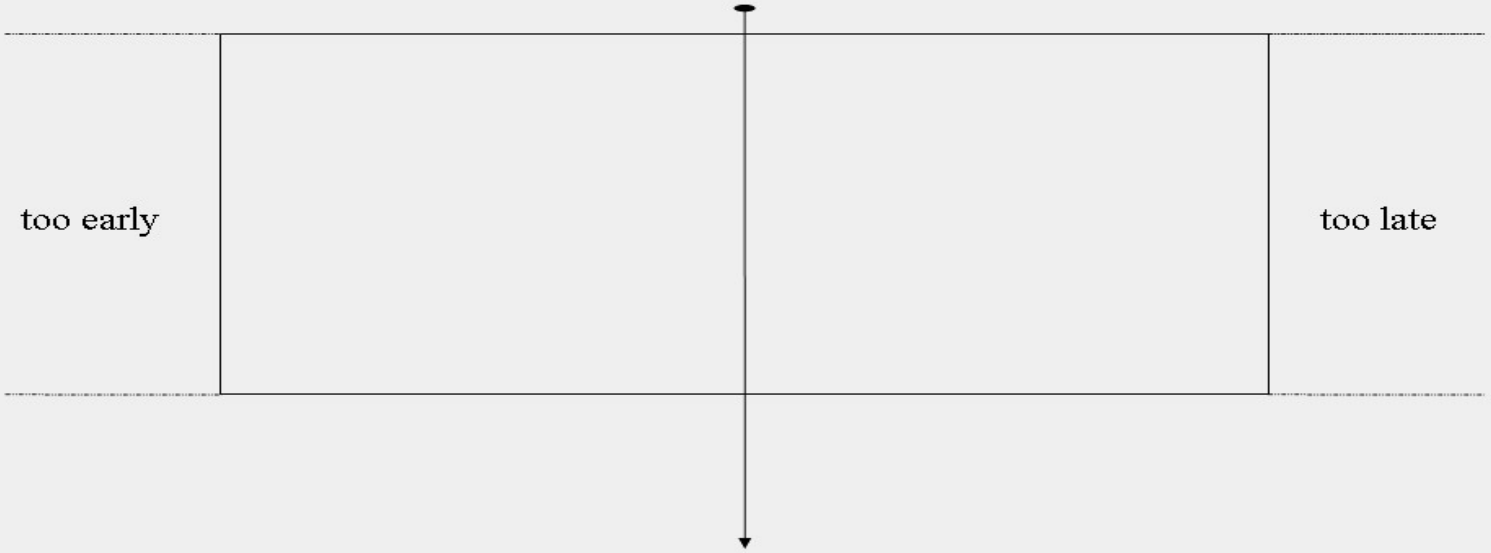
- Supposons STC = timer hardware à 5 KHz:
  - TIR(STC)= 5000
- Pour un affichage à 50 fps :
  - ΔSTC = 5000 / 50 = **100** (TIR % fps)
- Pour un affichage à 60 fps :
  - ΔSTC = 5000 / 60 = **83,33 = 83 + 1/3**
  - ***Fraction continue!***
- S'en suivent les valeurs de STC avec les incréments suivants :
- | 83 | 83 | **84** | 83 | 83 | **84**...
- **FC ⇒ JITTER aussi dans la STC**
- **Cumulée avec le JITTER de PTS**

Cadences en pratique

- Comment comparer STC avec TIR(STC) = 5000 vs PTS avec TIR(PTS) = 90000 ?
  - ⇒ produit en croix
  - **STC'** = STC x TIR(PTS) / TIR(STC)
  - (on espère que le ratio des TIRs est entier !)
  - Ici, **STC'** = STC x 90K / 5K = STC x 18
  - ⇒ **STC'** maintenant comparable avec PTS
  - Mais jitter de **STC'** multiplié par 18 ...
- Problèmes d'adaptation de cadence à cause des jitter de PTS et jitter de STC ou **STC'**

Exemple :

- Adaptation source 59,97 ips → affichage 60 ips
  - Théoriquement : adaptation par répétition 1 image sur 1000
- En pratique : jitter PTS + jitter STC
  - Tremblement du critère PTS – STC
  - Mauvais choix d'adaptation : saute / répète / saute / répète…



La vidéo numérique en pratique

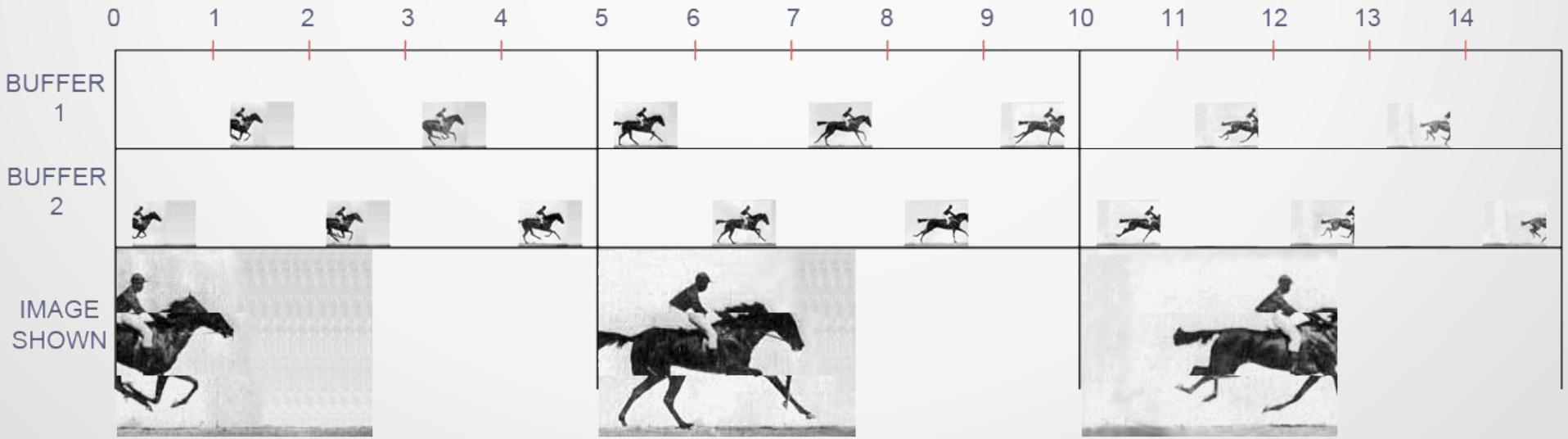
Buffering
Choisir l’image à afficher

Bufferisation

- Envoyer une image à l’afficheur
- En vidéo : celle « à l’heure » (PTS vs STC)
- En CGI realtime (jeux) : comment ?
- ⇒Bufferisation
  - Double non synchronisée
  - Double synchronisée
  - Triple synchronisée

Bufferisation non VSYNC

- Envoyer le backbuffer suivant dès qu'il est prêt
- Quel que soit l'état d'affichage du frontbuffer
- Avantages :
  - Un seul backbuffer
  - Rapide
- Inconvénient :
  - Tearing back/front

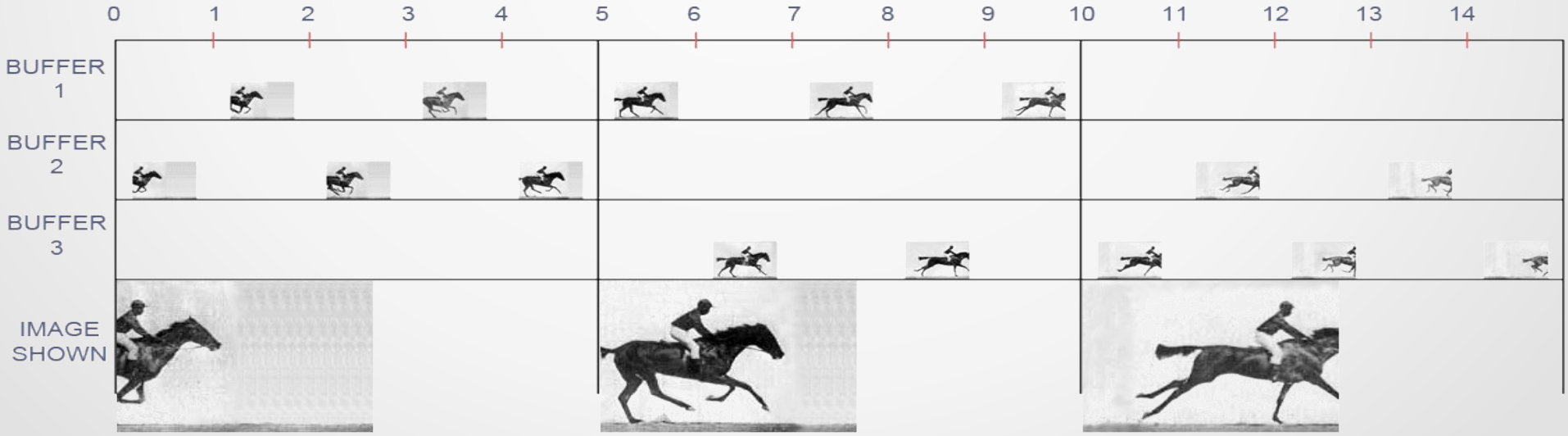


Bufferisation VSYNC

- Permuter frontbuffer et backbuffer au VSYNC
- Avantages :
  - Pas de tearing
- Inconvénient :
  - Producteur aussi lent que l'afficheur

Bufferisation triple + VSYNC

- Deux backbuffers composés en alternance
- Au VSYNC, envoyer le backbuffer prêt en frontbuffer
- Avantages :
  - Pas de tearing
  - Découplage cadence production vs affichage
- Inconvénients :
  - Deux backbuffers
  - CPU/GPU à fond



La vidéo numérique en pratique

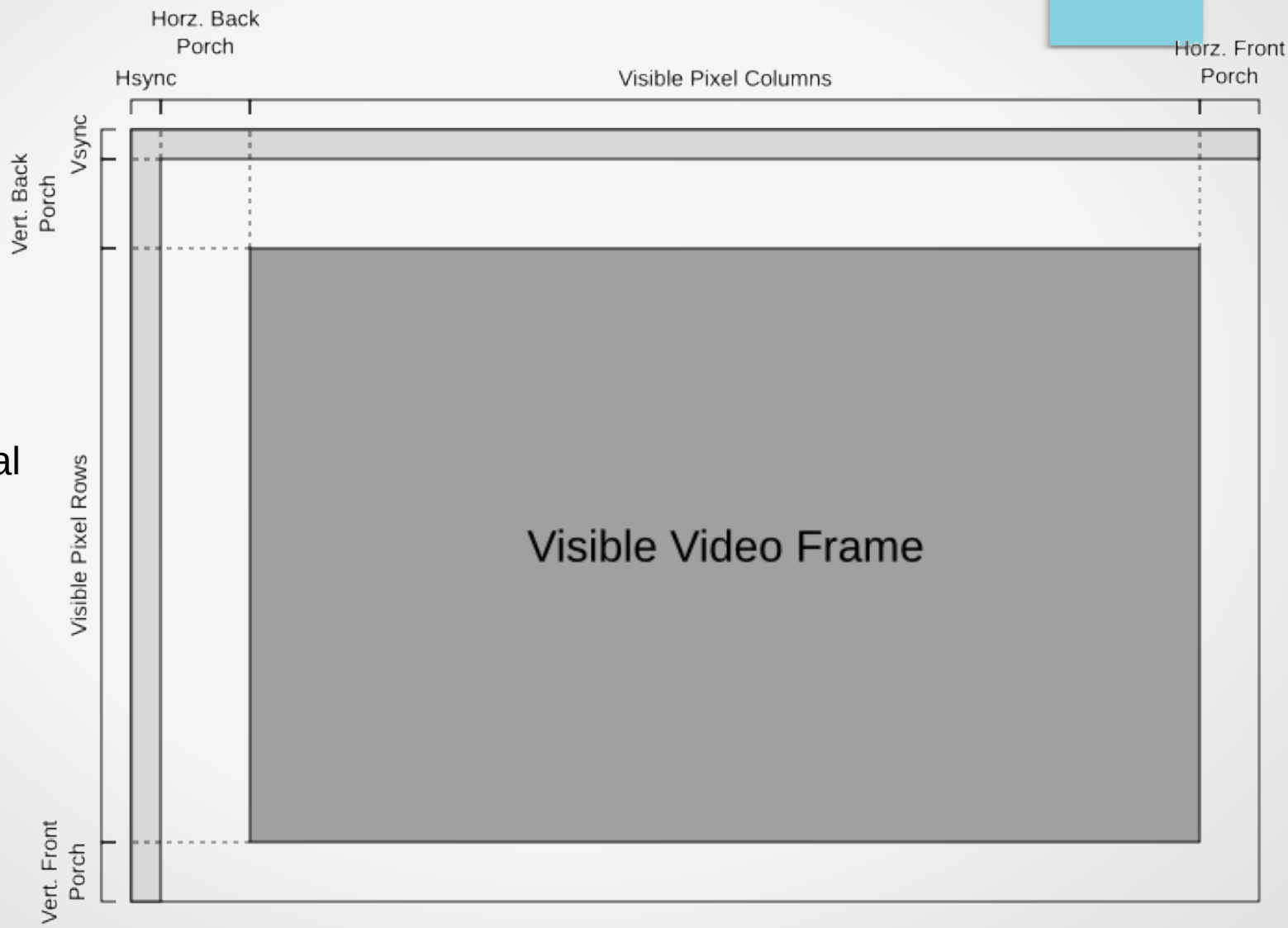
De l’image à l’écran
Comment afficher

De l’image à l’écran

- Comment cadrer l’image dans l’écran ?
  - En fréquence
  - En phase
- En fréquence :
  - Pulses verticaux : VSYNC
  - Pulses horizontaux : HSYNC
- En phase :
  - Palliers avant / arrière
- Pulses et palliers normalisés
- **VGA, DVI, HDMI : D**isplay **D**ata **C**hannel ⇒ **E**xtended **D**isplay **I**dentification **D**ata
- **Xorg** : « Modelines »

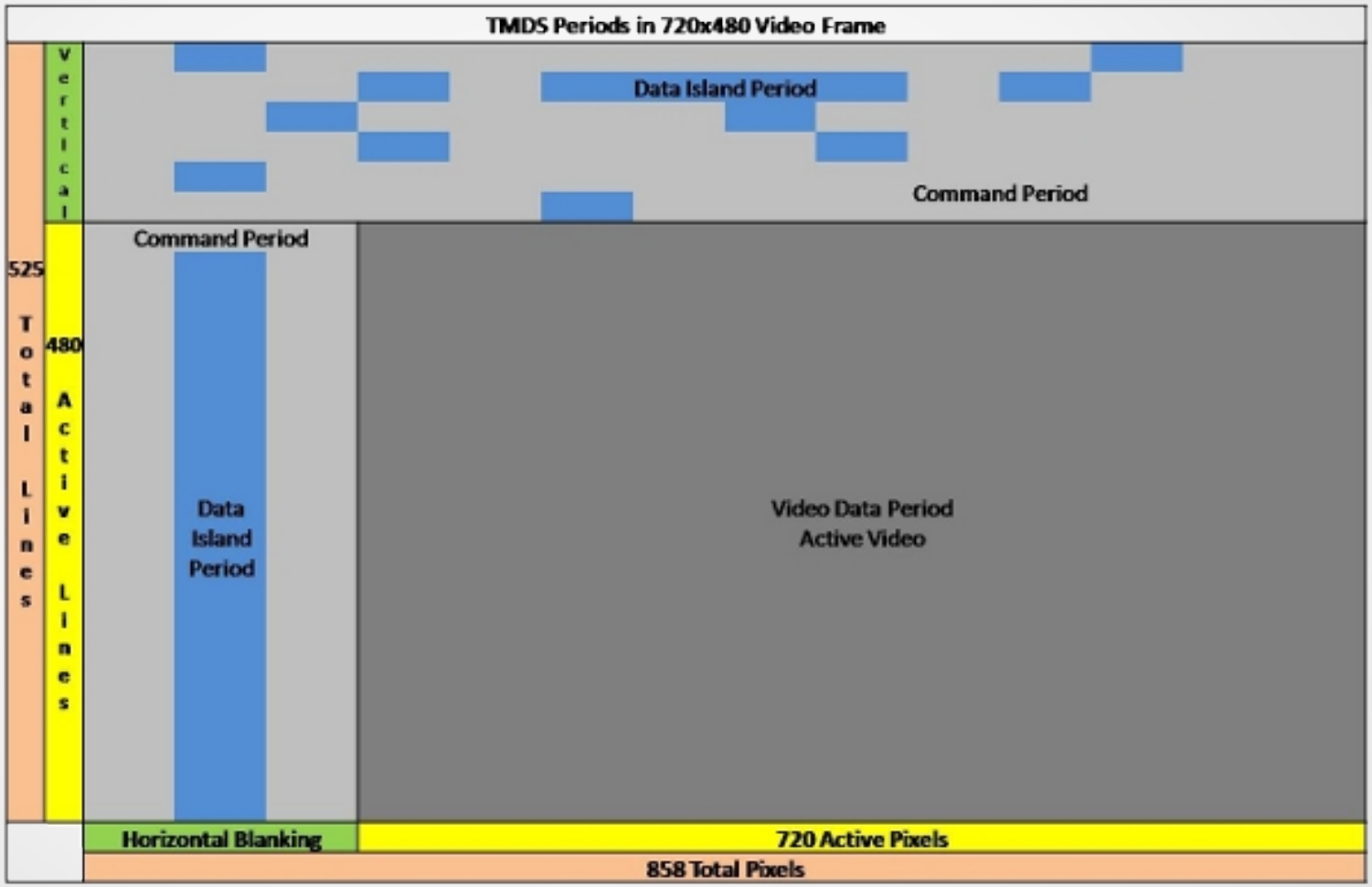
De l’image à l’écran

- Vertical Blanking
- Horizontal Blanking



DVI / HDMI

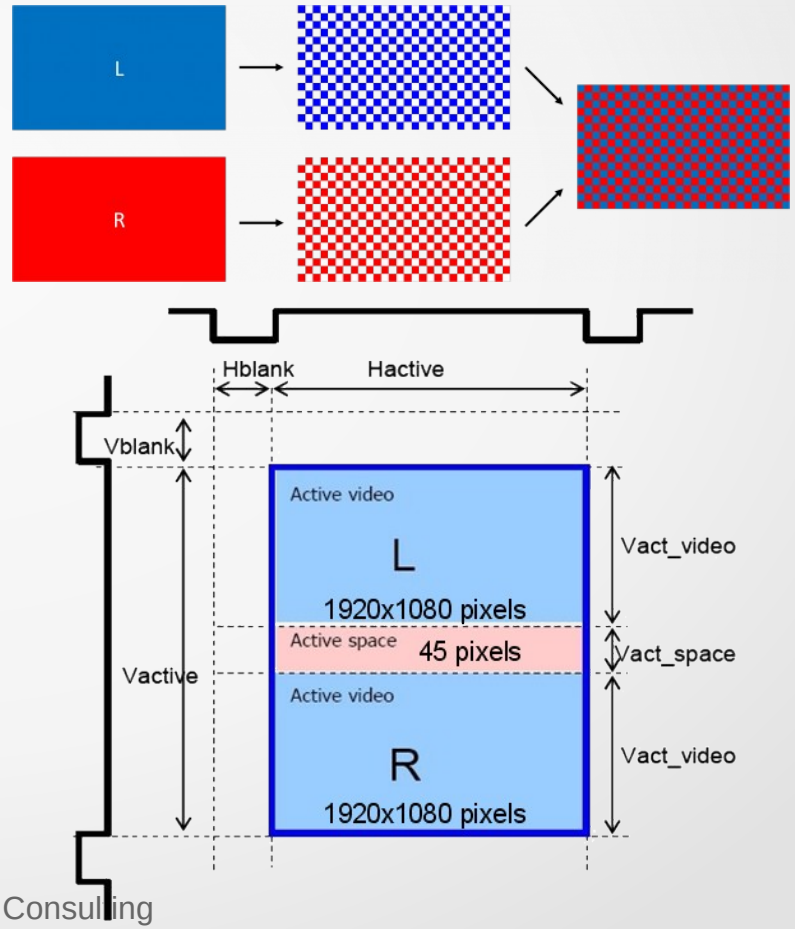
- Command Period
- Data Island Period



HDMI 3D

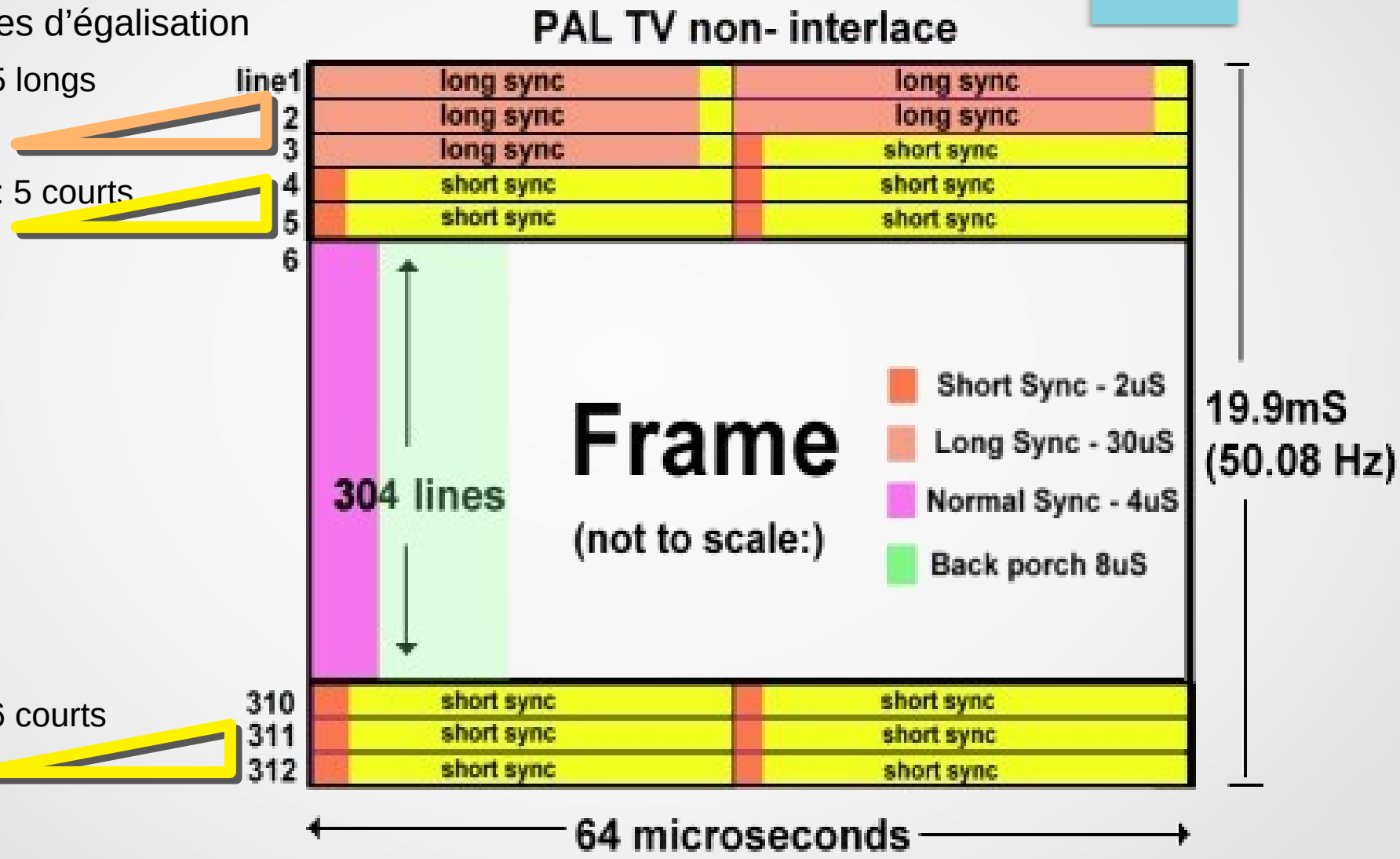
- Comment afficher des images 3D ?
- Plusieurs formats 3D numériques
- Dans tous les cas, pixel clock x2
- CheckerBoard (NVIDIA) :
  - VBlank + VSync
  - HSyncs + Lignes de pixels OG/OD en quinconce
  - Lignes deux fois plus larges
- Frame Pack (HDMI 1.4A) :
  - VBlank + VSync

    HSyncs + Lignes vue OG
  - HSyncs + Lignes vue OD
  - i.e: OG puis OD sans VBlank



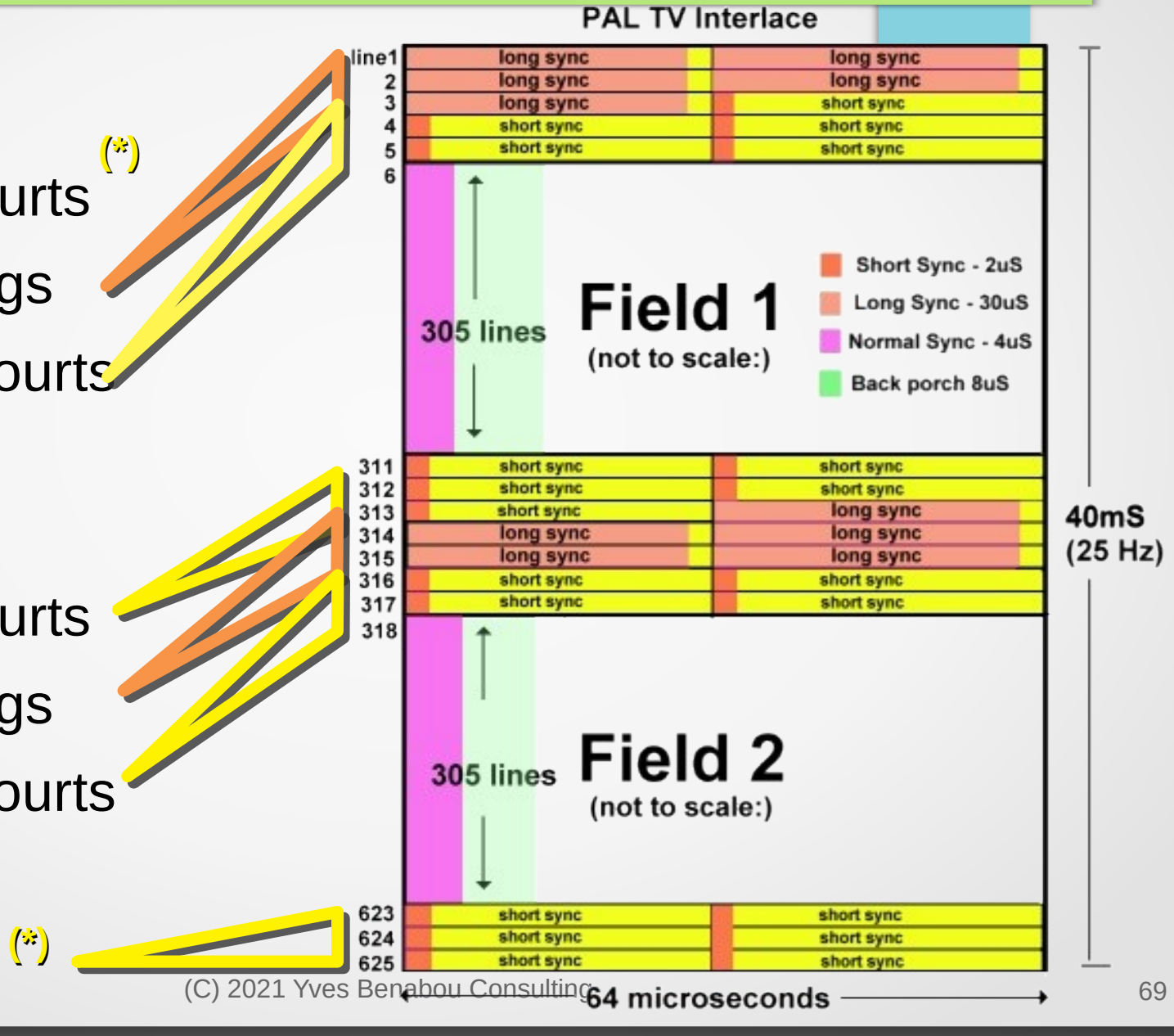
Analogique

- Pulses d'égalisation
- Eq : 5 longs
- Post : 5 courts
- Pré : 6 courts



Analogique entrelacé

- Field 1
  - Pré : 6 courts (*)
  - Eq : 5 longs
  - Post : 5 courts
- Field 2
  - Pré : 5 courts
  - Eq : 5 longs
  - Post : 4 courts



(*)

La vidéo numérique en pratique

Merci !
Des questions ?