



L'ULTIME PACK

pour comprendre l'étalonnage

KIT ÉTALONNAGE

*Décoder ton rush avant de l'étalonner —
6 leçons pour comprendre ton fichier vidéo.*



videomontage.fr

KIT DE DÉMARRAGE

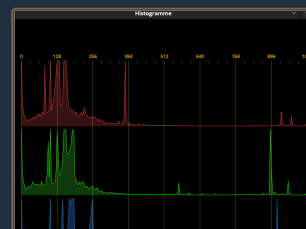
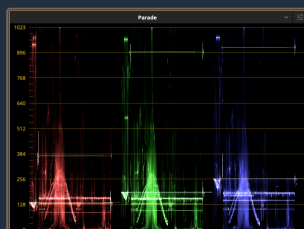
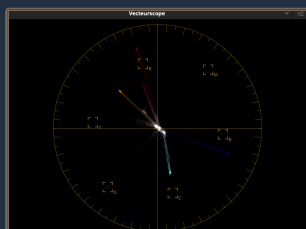
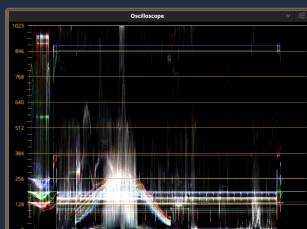
Étalonnage ▶



01:23:45:12

● REC 24p

L'étalonnage commence ici. Pas dans les courbes.



SOMMAIRE

INTRODUCTION

- Ce qu'il faut comprendre p.04

A — CONNAÎTRE TON FICHIER

- 1 Le format d'image p.06
- 2 Le bitrate p.08

B — COMPRENDRE LE SIGNAL

- 3 Le signal vidéo p.10
- 4 Le chroma subsampling p.12

C — MAÎTRISER LA COMPRESSION

- 5 La compression p.14
- 6 Les codecs et les conteneurs p.16

CONCLUSION

- Lire un rush en 30 secondes p.18
- Glossaire — 25 termes essentiels p.20
- Pour aller plus loin p.21

L'étalonnage commence avant la première courbe.

Avant même d'ouvrir ton logiciel de montage, ton rush a déjà tout dit. Sa résolution, son débit, la façon dont il code la couleur, dont il compresse le mouvement — tout ça décide à l'avance de ce que tu pourras faire ou pas en post-production. Si tu connais ton fichier, tu sais où tu peux pousser et où ça va casser. Si tu ne le connais pas, tu fais ce que la moitié des gens font : tu tournes les boutons au hasard, tu vois un dégradé qui bande sur un ciel, une sélection peau qui foire, une timeline qui rame, et tu ne comprends pas pourquoi.

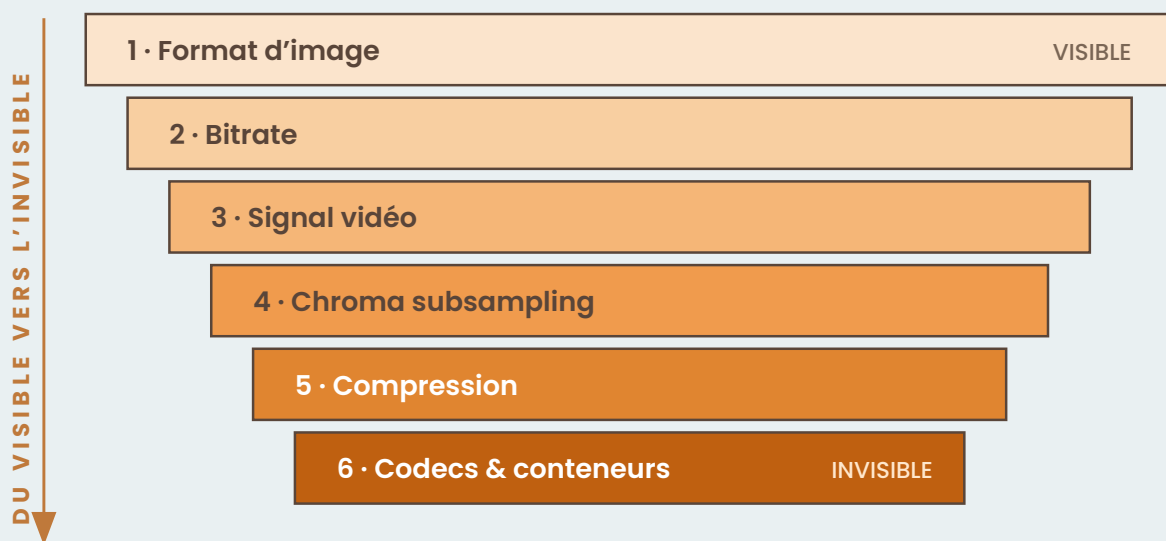
Ce guide a un seul but : te donner les six clés qui te permettent de regarder n'importe quel rush et de savoir tout de suite ce qu'il a dans le ventre. On va remonter du visible vers l'invisible. On commence par ce que tu vois en premier — la taille de l'image — et on descend couche après couche jusqu'à ce qui se passe réellement dans le fichier au niveau des bits.

Format d'image, débit, signal, échantillonnage couleur, compression, codec : **six niveaux d'analyse**, six couches qui se conditionnent les unes les autres.

À la fin de ces 6 leçons, tu n'auras plus besoin de mémoriser des recettes mystérieuses. Tu pourras ouvrir un .mp4 sorti d'un smartphone et savoir, en 30 secondes, jusqu'où tu peux aller à l'étalonnage et à partir d'où il faudra marcher sur des œufs. Tu verras ton workflow autrement : moins de bricolage, plus de décisions claires.

Ce que tu vas apprendre ici reste vrai dans dix ans. Les caméras vont changer, les versions logicielles aussi, mais les fondamentaux de l'image numérique — ils sont là pour durer.

Allez, on attaque. Première couche : la carte d'identité de ton fichier.



Six couches d'analyse qui se conditionnent les unes les autres.

Allez, c'est parti, suis le guide !

1

SECTION A — CONNAÎTRE TON FICHIER

Le format d'image

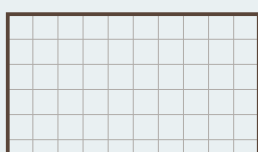
Avant de toucher à ton image, il faut savoir ce que tu as entre les mains. Le format d'image, c'est sa carte d'identité.

Quand tu regardes ton rush dans le navigateur de ton logiciel, trois informations te sautent aux yeux : sa taille, sa forme, et le type de fichier dans lequel il est rangé. Ces trois infos forment ce que j'appelle la carte d'identité du fichier. Et comme une vraie carte d'identité, elle te dit immédiatement à qui tu as affaire — et ce que tu peux faire avec.

La résolution, c'est le nombre de pixels qui composent ton image. Plus il y en a, plus ton image est définie. Les standards modernes sont parfaitement homothétiques entre eux : la HD pleine fait 1920 par 1080 pixels, l'UHD en fait exactement le double avec 3840 par 2160 — soit 2×1920 et 2×1080 . Et la 8K (UHD2) vient encore doubler : 7680 par 4320. Tout est calé sur le même rapport, ce qui simplifie énormément les conversions entre formats.

Petite confusion à éliminer tout de suite : une télévision 4K, ça n'existe pas. Ce qu'on te vend comme « 4K » dans ton salon, c'est de l'UHD (3840×2160). La vraie 4K, c'est du cinéma : 4096 pixels de large, avec un ratio variable selon le film. La différence n'est pas anodine — la télé a une forme figée une fois achetée, alors qu'au cinéma il suffit d'écarter un peu les rideaux pour changer le cadre. Ta caméra grand public ne fait quasiment jamais de la « vraie 4K » non plus, elle fait de l'UHD.

RÉSOLUTION



1920 × 1080

nombre de pixels

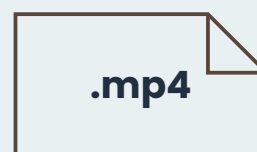
RATIO



largeur ÷ hauteur

forme du cadre

CONTENEUR



enveloppe du fichier

.mov · .mxf · .mts...

La carte d'identité = résolution + ratio + conteneur.

Le ratio, c'est la forme du cadre. Tu prends la largeur, tu la divises par la hauteur, et tu obtiens un chiffre : 1.78 (le 16:9 de ta télé), 1.85 (le format cinéma standard), 2.39 (les westerns, les films cinémascope). Plus le chiffre est grand, plus l'image est allongée. C'est aussi simple que ça. Et selon où ta vidéo va être vue, le ratio compte : 16:9 remplit une télé, mais coupe ou laisse des bandes noires sur d'autres écrans.

Le conteneur, c'est la « boîte » qui range tout ça : .mp4, .mov, .mxf, .mts... On reviendra dessus en détail au chapitre 6, parce que le conteneur n'est pas le fichier vidéo lui-même — c'est juste l'enveloppe qui le transporte. Mais sache déjà qu'il fait partie de la carte d'identité : il conditionne la lisibilité de ton rush sur les différents logiciels et plateformes.

**REMARQUE
#1**

« 4K » sur ta télé = UHD (3840×2160). La vraie 4K, c'est 4096 px de large, et c'est cinéma. Confusion la plus fréquente.

POUR TON ÉTALONNAGE

La carte d'identité, c'est la première chose que tu lis avant d'attaquer. La résolution change la vitesse de prévisualisation de ta timeline (plus c'est gros, plus ton ordinateur souffre). Le ratio détermine ton recadrage final. Et le conteneur peut tout simplement empêcher ton rush de s'ouvrir si la combinaison n'est pas reconnue. Avant même de toucher à une courbe, vérifie que la carte d'identité de ton rush est compatible avec le format de livraison que tu vises. Sinon tu vas perdre du temps pour rien.

À RETENIR

- ▶ La carte d'identité = **résolution + ratio + conteneur**.
- ▶ HD = 1920×1080. UHD = 3840×2160 (2× la HD). 8K = 7680×4320 (2× l'UHD).
- ▶ « 4K » télé n'existe pas — c'est de l'UHD. La vraie 4K (4096 px) = cinéma.
- ▶ Ratio = largeur ÷ hauteur. 1.78 = 16:9 · 1.85 = ciné · 2.39 = cinémascope.

Maintenant qu'on sait lire la carte d'identité, on plonge un cran plus bas — vers la quantité d'information qui circule réellement dans ton fichier : son débit.

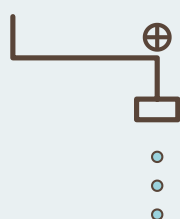
🔍 Fiche mémo associée : **M0_F1 — Le format d'image** (incluse dans ton Kit).

Le bitrate

Tu as une carte d'identité, mais ça ne te dit pas combien de matière vit dans le fichier. Pour ça, il faut regarder le débit.

Imagine un robinet. Si le débit est faible, l'eau coule au compte-goutte — ton fichier « encaisse » peu d'information par seconde, il est maigre. Si le débit est gros, l'eau coule fort — ton fichier est plein de matière, riche, et tes étalonnages tiennent. Le bitrate, c'est exactement ce robinet : la quantité de données qu'on enregistre chaque seconde. On le mesure en mégabits par seconde (Mbps).

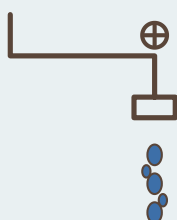
Plus le bitrate est élevé, plus ton fichier contient d'informations détaillées sur chaque image. Et plus tu peux te permettre de pousser ton image en post-production sans qu'elle craque. À l'inverse, un bitrate famélique te donne une image qui paraît correcte au premier coup d'œil mais qui s'effondre dès que tu touches un curseur.



10 Mbps

débit faible · upload YouTube

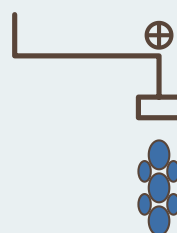
MAIGRE



50 Mbps

débit moyen · caméra 4K grand public

CORRECT



200 Mbps

débit pro · codec intermédiaire

RICHE

Le bitrate, c'est le robinet : combien d'informations par seconde dans ton fichier.

Quelques ordres de grandeur pour te repérer. La télévision en France diffuse autour de 4 Mbps. Quand tu envoies une vidéo sur YouTube, tu uploades à environ 10 Mbps — donc tes propres exports ont déjà plus de matière que ce que la télé fait passer en diffusion. Une caméra grand public en 4K va plutôt enregistrer entre 50 et 100 Mbps. Et en postproduction professionnelle, un codec intermédiaire peut grimper à 200, 400 Mbps ou plus, parce qu'on veut conserver le maximum de matière pour travailler.

Le bitrate dépend aussi du contenu : un plan fixe a besoin de beaucoup moins de débit qu'un plan plein de mouvement, parce qu'il y a moins de différence d'image en image. C'est pour ça que ton fichier peut avoir un débit « variable » (VBR, variable bitrate) qui s'adapte à ce qui se passe à l'écran, ou un débit « constant » (CBR) qui maintient le même flux quoi qu'il arrive.

REMARQUE
#2

Ton export YouTube (~10 Mbps) a déjà plus de matière que ce que la télé française fait passer en diffusion (~4 Mbps).

POUR TON ÉTALONNAGE

C'est ici que ça se joue. Si ton bitrate est trop bas, tes dégradés vont craquer dès que tu pousses une courbe. Le ciel va apparaître par strates au lieu de couler doucement, les peaux vont prendre des marches d'escalier dans les ombres — c'est ce qu'on appelle **le banding**, et c'est le symptôme classique d'un fichier trop maigre. La règle est simple : si tu sais que tu vas étalonner sérieusement, assure-toi que ton fichier a un débit suffisant. Sinon, transcoder ton rush vers un codec intermédiaire haute qualité avant de l'attaquer règle 90 % des soucis.

À RETENIR

- ▶ Le bitrate, c'est le robinet : combien d'informations par seconde (en Mbps).
- ▶ Télé française = ~4 Mbps. Upload YouTube = ~10 Mbps. Caméra 4K grand public = 50–100 Mbps.
- ▶ Bitrate trop bas → **banding** sur les dégradés (ciels, peaux).
- ▶ VBR = débit qui s'adapte au contenu. CBR = débit constant.

Maintenant qu'on sait combien de matière circule, il faut comprendre comment cette matière est organisée. C'est là qu'on entre dans le signal.

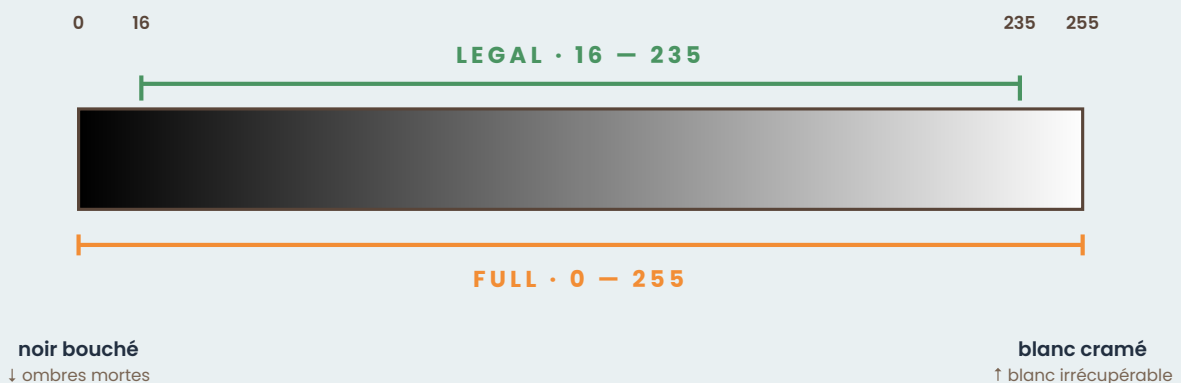
🔍 Fiche mémo associée : **M0_F2 — Le bitrate** (incluse dans ton Kit).

Le signal vidéo

Le débit te dit combien il y a de matière. Le signal te dit dans quelle plage cette matière a le droit d'exister.

Pense à un thermostat. Il a un minimum, un maximum, et toute la matière utilisable se trouve entre les deux. Si tu sors du minimum par le bas, tu coupes le chauffage. Si tu sors du maximum par le haut, tu fais sauter le compteur. Le signal vidéo, c'est exactement pareil : il y a une plage où l'information de ton image est valide. Si tu sors par le haut, ça crame (blanc irrécupérable). Si tu sors par le bas, ça noir-bouché (ombres mortes). Dans les deux cas, ce qui est perdu est perdu pour toujours.

Mais avant de parler de la plage, il faut comprendre combien de niveaux ton fichier peut représenter dedans. C'est ce qu'on appelle la quantification, et elle se mesure en bits par canal de couleur. En 8 bits, chaque canal peut prendre 256 valeurs. En 10 bits, on passe à 1 024 valeurs par canal — ce n'est pas 4 fois plus, c'est 64 fois plus de combinaisons couleur au total. En 12 bits (4 096 valeurs par canal), on entre dans le territoire du cinéma numérique haut de gamme.



Le signal a une plage où il a le droit d'exister. Sortir = perte définitive.

Une caméra te propose souvent plusieurs codecs : certains en 8 bits, d'autres en 10 bits. Le choix dépend de ce que tu vas faire. Une interview que tu vas livrer dans la foulée ? Le 8 bits suffit largement. Une scène que tu vas étalonner sérieusement ? Passe en 10 bits si tu peux — tu vas remercier ton toi du futur.

Sache aussi qu'un logiciel comme DaVinci Resolve travaille en interne en 32 bits (en virgule flottante), bien au-delà de tout ça. Mais s'il n'a que 256 niveaux en entrée, il fait juste tourner ces 256 niveaux dans un grand bocal — il ne peut pas inventer de la matière qui n'existe pas dans la source.

Maintenant, la plage du thermostat. Le signal numérique peut être codé sur deux échelles. Le Full Range utilise toute l'amplitude disponible (0 à 255 en 8 bits) — c'est le standard ordinateur. Le Legal Range (ou Video Range) garde des marges de sécurité : 16 à 235 en 8 bits — c'est le standard broadcast.

La confusion entre les deux est l'erreur classique du débutant. Tu importes un fichier en Full Range dans une timeline configurée Legal (ou l'inverse) — et ton image part en cacahuète sans que tu comprennes pourquoi. Tes noirs deviennent gris pâteux, tes blancs partent en vrille, tes contrastes sont faussés.

REMARQUE
#3

10 bits, ce n'est pas 4 fois plus que 8 bits — c'est 64 fois plus de combinaisons couleur au total.

POUR TON ÉTALONNAGE

Vérifier dans quelle plage tu travailles, c'est la première chose à faire en ouvrant un projet. Tes scopes (waveform, parade RGB) sont tes meilleurs alliés : ils t'affichent en temps réel où ton signal se situe. Si tu vois ton waveform écrasé en bas ou collé en haut sans détail, tu as probablement un souci d'interprétation Full / Legal. Et si tes dégradés bandent malgré un bon bitrate, regarde du côté de ta profondeur de bits : tu travailles peut-être sur du 8 bits là où il aurait fallu du 10.

À RETENIR

- ▶ Le signal a une plage où il a le droit d'exister. Sortir en haut = **cramé**. Sortir en bas = **bouché**.
- ▶ **8 bits** = 256 niveaux (télé, web). **10 bits** = 1024 niveaux (postprod sérieuse). **12 bits** = 4096 (cinéma).
- ▶ **Full Range** = 0-255. **Legal Range** = 16-235. Mélanger les deux = catastrophe.
- ▶ Un logiciel travaille en 32 bits en interne, mais ne peut pas inventer la matière manquante de la source.

Tu sais maintenant lire la plage et la précision de ton signal. Reste à voir comment la couleur, elle, est rangée à l'intérieur de tout ça.

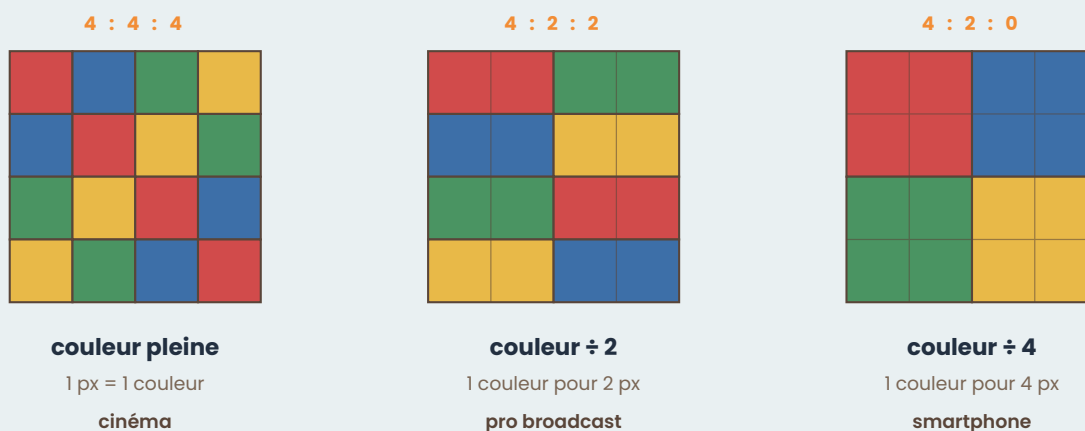
🔍 Fiche mémo associée : **M0_F3 — Le signal vidéo** (incluse dans ton Kit).

Le chroma subsampling

Tous les pixels de ton image n'ont pas la même précision en couleur. Et c'est volontaire.

L'œil humain est plus sensible à la lumière qu'à la couleur. Si tu fais l'expérience d'éteindre les lumières de ton salon, tu remarqueras qu'avant de ne plus rien voir du tout, tu cesses d'abord de distinguer les couleurs — tu ne vois plus que des nuances de gris. Cette particularité de la vision a donné une idée géniale aux ingénieurs : on peut enlever une grande partie de l'information de couleur sans que l'œil voie la différence, du moment qu'on garde toute la luminosité.

C'est exactement ce que fait le sous-échantillonnage couleur, ou chroma subsampling. Pour comprendre comment ça marche, il faut savoir qu'avant ce sous-échantillonnage, ton signal est transformé. Au départ ta caméra capte du RVB pur. Ensuite elle convertit ce signal en Y CrCb : Y c'est la luminance (l'image en noir et blanc), Cr la chrominance rouge, Cb la chrominance bleue. Et le vert ? Il n'est pas stocké. Il est recalculé à partir des trois autres composantes.



La luminance Y reste à pleine résolution. C'est la chrominance qui est sous-échantillonnée.

Une fois qu'on a séparé luminance et chrominance, on peut décider de garder plus ou moins de chrominance.

4:4:4 — chaque pixel a sa propre information couleur complète. Aucune compression couleur. C'est le territoire du cinéma numérique haut de gamme. Aucune caméra grand public n'enregistre en 4:4:4 — c'est un marqueur « pro / cinéma ».

4:2:2 — il y a une information couleur pour deux pixels. Ton œil ne voit aucune différence avec du 4:4:4 en visionnage. C'est le standard des caméras vidéo professionnelles et de la postproduction broadcast haut de gamme.

4:2:0 — il y a une information couleur pour quatre pixels. C'est le standard des smartphones, des hybrides photo-vidéo, des caméras grand public.

Concrètement : imagine la dame de la météo devant son écran vert. En 4:4:4, le contour de ses cheveux est net au pixel près. En 4:2:0, tu n'as plus qu'un pixel sur quatre défini en couleur précisément — et là, quand tu enlèves le fond vert, les contours commencent à être un peu approximatifs. Le 4:2:0 **n'est pas un défaut** — c'est un compromis intelligent qui économise jusqu'à 50 % du stockage avec une perte quasi imperceptible en visionnage normal.

REMARQUE
#4

En 4:2:0, la perte de précision se concentre sur le **bleu** et le **rouge**. Le vert reste relativement bien défini.

POUR TON ÉTALONNAGE

Si tu sais que ton rush est en 4:2:0 (smartphone, hybride photo, caméra reportage), tu vas devoir ajuster ta façon de travailler. Les sélections fines de zones colorées (isoler une couleur de peau, une teinte de ciel) vont galérer — tu auras des bords crénelés. La parade : compenser avec des **power windows** (masques manuels) plutôt que de s'appuyer uniquement sur les qualifieurs couleur. Et si tu veux travailler du keying proprement, oriente-toi vers du 4:2:2 dès la captation — ça change tout au moment du détourage.

À RETENIR

- ▶ L'œil voit mieux la lumière que la couleur → on compresse la couleur sans que tu le voies.
- ▶ Y = luminance · Cr = chrominance rouge · Cb = chrominance bleue. Le vert est recalculé.
- ▶ **4:4:4** = cinéma · **4:2:2** = pro broadcast · **4:2:0** = smartphone, grand public.
- ▶ 4:2:0 n'est pas un défaut, c'est un compromis. Mais ça complique les sélections fines et le keying.

Tu sais maintenant comment la couleur est rangée. Reste à voir comment tout ce signal est compacté dans le fichier — la couche la plus invisible mais la plus déterminante pour ton confort de travail.

🔍 Fiche mémo associée : **M0_F4 — Le chroma subsampling** (incluse dans ton Kit).

La compression

Compresser une vidéo, c'est résumer un livre. Tu peux le résumer page par page, ou en ne notant que ce qui change d'un chapitre à l'autre. Les deux marchent, mais pas pareil.

Compresser une vidéo, c'est trouver le moyen de stocker plus d'informations dans moins d'espace. En vidéo, ces deux philosophies portent un nom.

Compression intra-image (All-I) — chaque image est compressée séparément, comme une pile de photos indépendantes. Lourd en stockage, mais chaque frame est complète et autonome. Tu peux étalonner image par image sans que ton ordinateur ait à reconstituer quoi que ce soit. Codecs typiques : **ProRes, DNxHR, Cineform**.

Compression inter-image (Long-GOP) — une **image-clé** complète toutes les ~12 à 30 frames, et entre ces images-clés, on ne stocke que les *différences*. C'est très léger en stockage, mais ton processeur doit reconstruire les images intermédiaires en temps réel. Codecs typiques : **H.264, HEVC (H.265), AV1**.

INTRA-IMAGE (ALL-I)

6 images, chacune complète et autonome

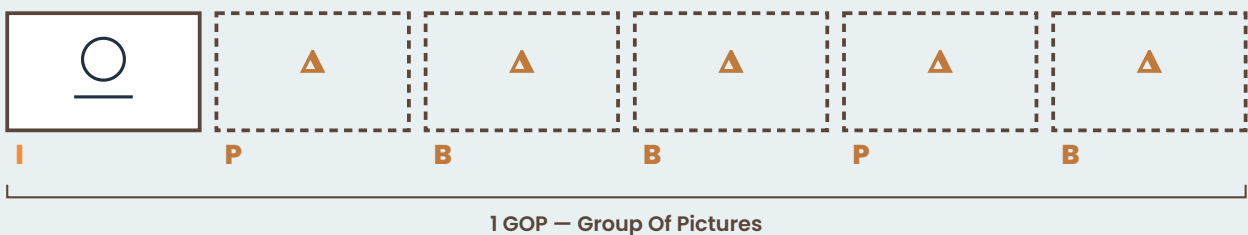


+ Confort de timeline. – Lourd à stocker.

ProRes · DNxHR · Cineform

INTER-IMAGE (LONG-GOP)

1 image-clé + 5 deltas (différences uniquement)



+ Léger à stocker. – Dur à monter.

H.264 · HEVC · AV1

Deux philosophies de compression — l'une stocke chaque image, l'autre les différences.

Le terme technique pour le groupe d'images intermédiaires entre deux images-clés s'appelle un **GOP (Group Of Pictures)**. Historiquement, le MPEG-2 utilisait des GOP de 12 images. Aujourd'hui, les codecs modernes comme HEVC peuvent espacer les images-clés de plusieurs minutes si rien ne bouge à l'écran — c'est ce qui les rend si performants en compression, et si exigeants en décodage.

Au-delà de cette distinction intra/inter, il faut savoir qu'il existe en réalité **trois niveaux de compression** qui s'empilent dans ton fichier :

- 1. La compression spatiale** — ce qui se passe dans une seule image. On regroupe les pixels qui se ressemblent en blocs, on simplifie ce que l'œil ne voit pas. À perte.
- 2. La compression temporelle** — ce qui se passe entre les images. C'est l'inter-frame dont on vient de parler. Aussi à perte.
- 3. La compression informatique (binaire)** — la dernière étape. C'est purement statistique : on repère les motifs qui se répètent dans la suite de 0 et de 1, et on les compacte. Exactement comme si tu disais « j'en veux 5 » plutôt que « je veux une pomme de terre, une pomme de terre, une pomme de terre... ». Cette compression-là, **elle ne perd aucune information**. C'est aussi la raison pour laquelle ça ne sert à rien de zipper une vidéo.

REMARQUE
#5

Zipper une vidéo ne sert à rien. La compression binaire a déjà été faite par le codec — il n'y a plus de motif évident à exploiter.

POUR TON ÉTALONNAGE

Si tu galères avec un fichier H.264 long-GOP qui rame sur ta timeline, ce n'est pas ta machine qui est nulle — c'est ton processeur qui s'épuise à reconstruire en permanence les images intermédiaires. Réflexe simple : **transcoder en codec intermédiaire** (ProRes Proxy ou DNxHR LB selon Mac ou Windows) avant de monter et d'étalonner. Tu ne gagnes pas en qualité — tu ne peux pas inventer ce qui n'est pas là — mais tu gagnes en *fluidité*. C'est exactement à ça que servent ces codecs : être agréables à manipuler pour le travail, pas légers pour le transport.

À RETENIR

- ▶ **Intra-image (All-I)** = chaque image autonome. Lourd à stocker, agréable à monter.
- ▶ **Inter-image (Long-GOP)** = images-clés + différences. Léger à stocker, dur à monter.
- ▶ 3 couches de compression : **spatiale** (à perte) · **temporelle** (à perte) · **binaire** (sans perte).
- ▶ Si Resolve rame sur du H.264 → transcoder en ProRes ou DNxHR règle le souci.

Reste un dernier mystère : derrière ces noms de codecs (H.264, HEVC, ProRes, DNx), il y a quoi exactement, et pourquoi il y en a autant ?

🔍 Fiche mémo associée : **M0_F5 — La compression** (incluse dans ton Kit).

Les codecs et les conteneurs

Si le conteneur est la boîte qui range le fichier, le codec est la langue dans laquelle il est écrit. Et toutes les langues ne se parlent pas pareil.

Quand tu importes un fichier vidéo, deux notions se cachent derrière chaque rush, et il faut absolument les distinguer.

Le conteneur, c'est la « capsule » qui range tous les ingrédients de ta vidéo. Il porte une extension visible : .mp4, .mov, .mxf, .mts, .avi... À l'intérieur, il range plusieurs flux : la vidéo, l'audio, le timecode, parfois des sous-titres, des métadonnées. C'est l'enveloppe, le contenant.

Le codec, c'est la « langue » dans laquelle ta vidéo est encodée à l'intérieur du conteneur. C'est l'algorithme de compression qu'on a vu au chapitre précédent. H.264, HEVC, ProRes, DNxHR — voilà les langues principales que tu vas croiser.

La distinction est cruciale parce que le même conteneur peut renfermer différents codecs. Un fichier .mp4 peut contenir du H.264 ou du HEVC. Un fichier .mov peut contenir du ProRes, du H.264, ou plein d'autres codecs. C'est le codec qui dit ce que ton ordinateur doit faire pour décoder l'image — pas le conteneur.

CODEC	FAMILLE	USAGE	CONFORT TIMELINE
H.264 (AVC)	Inter-image (Long-GOP)	Livraison · streaming	Lourd à décoder
HEVC (H.265)	Inter-image (Long-GOP)	Livraison UHD · smartphones récents	Très lourd à décoder
ProRes	Intra-image (All-I)	Postprod — écosystème Apple	Ultra fluide ✓
DNxHR · DNxHD	Intra-image (All-I)	Postprod — écosystème Avid/PC	Ultra fluide ✓

Les 4 codecs que tu rencontreras 95 % du temps.

Tous ces codecs n'ont pas le même usage. H.264 et HEVC sont conçus pour la **livraison** (légers à transporter, mais coûteux à décoder). ProRes et DNx sont conçus pour la **postproduction** (lourds à transporter, mais ultra fluides à manipuler). Ce sont ce qu'on appelle des *codecs intermédiaires de postproduction* — leur job, c'est d'être agréables à travailler.

Côté conteneurs, un mot sur les principaux :

.mp4 — la capsule la plus universelle. Vieille, pauvre en métadonnées, mais lue par absolument tout. Si tu veux qu'un fichier soit lisible partout sans réfléchir, mp4 + H.264 = le duo gagnant.

.mov — la capsule Apple, robuste, supporte les codecs pro (ProRes notamment). Pas toujours idéale pour la lecture sur PC anciens.

.mxl — la capsule pro broadcast, très évoluée, transporte beaucoup de métadonnées. C'est ce que tu trouves sur les caméras pro Sony et Panasonic.

.mts / .m2ts (AVCHD) — capsule « transport stream » conçue pour le flux (broadcast en direct), pas pour les applications créatives.

**REMARQUE
#6**

Sur une carte mémoire, **garde toute l'arborescence d'origine**. Ne copie jamais les .mts ou .mxl seuls — tu casses la structure qui lie audio, vidéo et timecode.

POUR TON ÉTALONNAGE

Le bon codec n'est pas « le plus performant dans l'absolu », c'est « celui qui matche ton workflow ». Pour étalonner sereinement, reste sur les codecs intermédiaires (ProRes ou DNx). Pour livrer au client final, exporte en H.264 ou HEVC selon la cible. Et si ton ordinateur souffre dès l'import, pose-toi la question : « Est-ce que je devrais transcoder mes rushs avant de monter ? » Cinq minutes de transcodage en début de projet te sauveront des heures de frustration ensuite.

À RETENIR

- ▶ **Codec** = la langue. **Conteneur** = la boîte. Ne pas confondre.
- ▶ H.264 / HEVC = livraison, légers, coûteux à décoder. ProRes / DNx = postprod, lourds, fluides.
- ▶ mp4 + H.264 = duo universel pour la livraison. ProRes ou DNx en mov pour la postprod.
- ▶ Sur les cartes mémoire : ne casse **JAMAIS** l'arborescence d'origine. Tout ou rien.

Tu as maintenant les six clés. Reste à mettre tout ça ensemble dans la vraie vie.

🔍 Fiche mémo associée : **M0_F6 — Les codecs** (incluse dans ton Kit).

Lire un rush en 30 secondes.

Imagine : un client te livre un rush. Tu cliques droit sur le fichier, tu regardes ses propriétés, tu l'ouvres dans ton logiciel. Ce que tu vois :



FICHER À ÉTALONNER

.mp4 · UHD 3840 × 2160 · 50 Mbps
8 bits · 4:2:0 · H.264

Voilà ce que ton œil entraîné lit immédiatement, sur les six couches :

- 1 Carte d'identité.** UHD 16:9, conteneur mp4. Côté résolution et forme, tu as ce qu'il faut pour livrer en télé ou web. Le mp4 te garantit la compatibilité.
- 2 Bitrate.** 50 Mbps, honorable pour du 4K grand public. Pas du niveau pro broadcast (200+), mais suffisant pour un étalonnage léger. Pousser fort sur les ciels ou les peaux → banding probable.
- 3 Signal.** 8 bits, 256 niveaux. Marge mince. Une remontée brutale du contraste ou une bascule de balance des blancs créera des marches d'escalier. Vérifier Full / Legal en ouvrant.

4

Chroma. 4:2:0. La couleur est définie sur 1 pixel sur 4. Tes sélections fines (qualifiers peau, ciel) vont être bruitées sur les bords. Pour keying ou étalonnage zone par zone précis, compenser avec power windows manuels.

5

Compression. H.264 = Long-GOP (inter-image). Sur une timeline UHD, ton processeur va peiner à reconstruire les images intermédiaires en temps réel. Lecture saccadée garantie, sauf si tu transcodes.

6

Codec / conteneur. H.264 + mp4 = format de *livraison*, pas de *création*. Réflexe pro : transcoder en ProRes 422 (Mac) ou DNxHR HQ (Windows) avant d'attaquer. Tu ne gagnes pas en qualité — tu gagnes en fluidité de travail et en stabilité du signal à l'export.

LA STRATÉGIE EN UNE PHRASE

Ce rush est viable pour un étalonnage modéré, à condition de le transcoder en codec intermédiaire avant de toucher quoi que ce soit, et d'éviter les mouvements brutaux sur les courbes pour ne pas faire ressortir le banding.

C'est ça, lire un rush en 30 secondes. Tu n'as plus besoin de mémoriser des recettes mystérieuses ni d'essayer au hasard. Tu connais ta matière première, tu sais où elle est forte, où elle est fragile, et tu adaptes ton approche en conséquence. L'étalonnage commence avant la première courbe — il commence à l'instant où tu poses les yeux sur ton fichier.

À très vite !

Ludovic

Glossaire

25 termes essentiels — à garder sous le coude.

AVCHD

Format conteneur (.mts / .m2ts) utilisé par certaines caméras grand public et hybrides.

Banding

Marches d'escalier visibles dans les dégradés, dues à une quantification ou un bitrate insuffisants.

Bitrate

Débit binaire, quantité d'information par seconde dans le fichier (en Mbps).

CBR

Constant Bitrate, débit constant tout au long du fichier.

Chroma subsampling

Sous-échantillonnage de la couleur par rapport à la luminance (ex : 4:2:0).

Chrominance (Cr / Cb)

Composantes couleur séparées du signal (rouge et bleu).

Codec

Algorithme de compression / décompression d'un flux numérique.

Conteneur

Enveloppe logicielle (.mp4, .mov, .mxf) qui regroupe vidéo + audio + métadonnées.

DNxHR

Codec intermédiaire de postproduction (équivalent Avid de ProRes), idéal sous Windows.

Full Range

Codage utilisant l'intégralité de l'échelle (0-255 en 8 bits).

GOP

Group of Pictures, groupe d'images entre deux images-clés en compression inter-image.

HEVC (H.265)

Codec de compression haute efficacité, successeur du H.264.

HD / UHD / 4K

HD = 1920×1080. UHD = 3840×2160 (la « 4K » télé). 4K = 4096 px de large (cinéma).

Inter-image (Long-GOP)

Compression temporelle stockant uniquement les différences entre images.

Intra-image (All-I)

Compression où chaque image est encodée de manière autonome.

Legal Range

Plage de signal restreinte (16-235) pour la diffusion broadcast.

Luminance (Y)

Composante de luminosité du signal, à laquelle l'œil est le plus sensible.

MXF

Conteneur professionnel broadcast, riche en métadonnées.

Pixel

Plus petite unité de l'image numérique.

ProRes

Codec intermédiaire de postproduction (référence Apple), idéal sous Mac.

Quantification

Précision avec laquelle le signal est mesuré, exprimée en bits par canal.

Ratio

Forme du cadre, largeur ÷ hauteur (ex : 1.78 pour 16:9).

Transcoder

Convertir un fichier d'un codec à un autre, sans modifier la résolution.

VBR

Variable Bitrate, débit qui s'adapte au contenu (plus dense sur les scènes complexes).

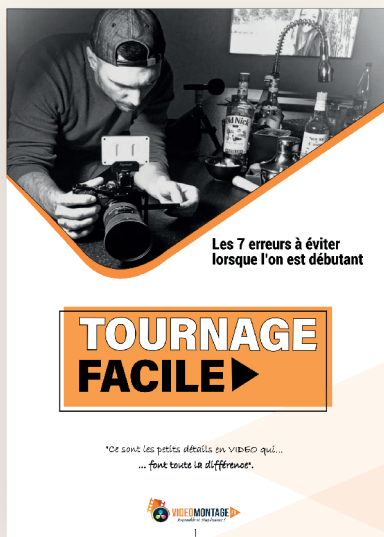
YCrCb

Espace couleur séparant la luminance et les deux composantes de chrominance.

POUR ALLER PLUS LOIN

D'autres guides gratuits sur videomontage.fr

Si tu as aimé ce guide, tu vas adorer mes deux autres.

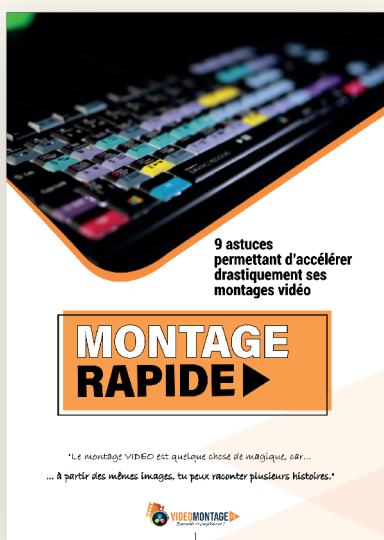


GUIDE GRATUIT

Tournage facile

Le guide complet pour bien filmer avant de monter — composition, éclairage, son, équipement.

↓ Télécharger gratuitement



GUIDE GRATUIT

Montage rapide

Le guide complet pour organiser ton workflow et monter efficacement avec DaVinci Resolve.

↓ Télécharger gratuitement

À très bientôt sur videomontage.fr !



videomontage.fr

Ce livre numérique est protégé par le droit d'auteur. Tous les droits sont exclusivement réservés à Ludovic Voigt et aucune partie de cet ouvrage ne peut être republiée, sous quelque forme que ce soit, sans le consentement écrit de l'auteur.

Vous n'avez aucun des droits de revente, ni de diffusion, ni d'utilisation de cet ouvrage sans accord préalable de l'auteur.

Vous ne disposez d'aucun droit de label privé. Toute violation de ces termes entraînerait des poursuites à votre égard.

Copyright 2026 — Ludovic Voigt · videomontage.fr · Tous droits réservés.

Ensemble et simplement !