# Principes et normes de compression vidéo

Pourquoi la compression ?

La taille est trop importante. La vidéo brute n’est pas viable, on est obligé de réduire la taille. La majorité des médias sont limités en bande passante.

**5.** Même information brute mais en quantité d’information, il y a une différence.

* L’image 3 est la plus facile à compresser car il y a plus de redondance.
* La 1 est plus difficile car elle parait plus aléatoire
* Information : Données utiles non redondante

**6.** L’œil humain voit mieux les zones lumineuses. Si on veut dégrader, il faut dégrader les zones sombres.

Il existe des techniques pour tromper l’œil humain

**11.** Avec combien de pixels on représente la couleur

8 bits/pixels est le top, à partir de 7, l’œil humain peut voir la différence.

En gros, quand on compresse, si on représente un pixel par 8 bits, on voit parfaitement. Par exemple, pour 5 bits, on voit la différence, il y a des « plaques », qui montrent qu’il y a moins de nuances de couleurs.

**12.**Quand on compresse avec perte et qu’il y a la moindre perte, cela influe sur tout le fichier.

Avec perte : Ce n’est pas grave de perdre des informations mais il faut que ces pertes soient significatives pour l’œil humain et pour l’objet sur lequel il l’utilise.

**14.** Quand on a des couleurs uniformes, on a des basses fréquences. Dès qu’il y a un changement de couleurs, on a les hautes fréquences.

**16.** 8 bits par composant (R,G,B) 🡪 24 bits / pixels

Chrominance : information sur la couleur

Pour compresser, on passe à 12 bits/pixels

**17.** YUV : format qui représente le plus de lumières

**18.** L’oeil humain est sensible à 4 millions de couleurs donc 24 bit/pixel c’est beaucoup trop

La verte contient le plus d’informations car c’est celle qui a attrapé le plus de lumières mais on voit que la lumière est sur les 3 images et donc ce n’est pas bien.

**19.** YUV permet de représenter la luminosité

**20.** La composante Y (intensité) capte toute la lumière donc l’information est plus importante

**22 à 54.** Les détails de l’image sont dans les hautes fréquences.

La valeur la plus importante est la composante DC.

La DCT sépare les basses fréquences et les hautes fréquences.

Il existe plusieurs quantifications.

L’inconvénient de la quantification fixe (diviser toute la matrice par le même facteur) c’est que cela réduit les HF mais aussi les BF.

Les tables de quantification sont mieux car elles permettent de mieux diminuer.

**À retenir :**

**Comment produire un flux VBR ?**

**Qu’est-ce qu’un flux VBR ?**

**53.** L’idée de la compression MJPEG est une compression image par image. Cela est commun pour les vidéos car cela ne nécessite pas de traitement vidéo. On ne décrypte que la trame sur laquelle on travaille.

**56**. Compression MPEG : utilise un codage temporel.

Je code une trame en JPEG et la trame suivante est codé par rapport à la trame précédente. On fait la différence avec l’image d’avant. Avec 30 image/seconde, il y a beaucoup de redondance et donc les différences sont presque nulles.

Il faut toujours des images intra car si on change de chaines et qu’on arrive sur des trames temporelles, on n’a pas la trame intra de référence. Il faut donc en insérer une toutes les 1 à 2 secondes pour toujours avoir une trame de référence.

**57.** La dernière image (différence entre image) contient moins d’informations donc la compression est beaucoup plus efficace

**58.** On fait des différences par bloc. On calcul la DCT de la matrice d’erreur (différence entre deux images), il y aura beaucoup plus de 0 et donc la compression sera plus efficace.

Au lieu de transmettre l’image S2, on transmet la matrice d’erreur

**61.** On fait la différence avec un autre bloc, qui n’est pas forcement celui d’origine. On cherche le bloc qui ressemble le plus et on fait la comparaison avec ce bloc-là.

Il faut définir la zone où chercher pour que la comparaison soit rentable

**62. L**a matrice de luminance car c’est celle qui contient le plus d’informations

**66.** Codage bidirectionnel : au lieu de coder l’erreur avec l’image d’avant, on peut coder l’image avec celle d’après si on obtient une erreur moins importante.

**67.**