



Institut Supérieur du Numérique
Département Communication Numérique et Multimédia



NCO

Numérisation et Codage des Objets Multimédia

PLAN

Chapitre 1 : Généralités et définitions

Chapitre 2 : Numérisation des données Multimédia

Chapitre 3 : Théorie de l'Information et Codage

Chapitre 4 : Méthodes de Compression de Données Multimédia

Dr. EL BENANY Mohamed Mahmoud

1. Introduction



- ❑ En **multimédia**, l'information peut prendre plusieurs formes comme le **texte, le son, l'image ou la vidéo**. Selon la nature de l'information, les données peuvent avoir des tailles importantes.
- ❑ Le **stockage** ou la **transmission** de données volumineuses pose problème soit vu l'espace mémoire nécessaire pour le stockage soit vu le débit et le temps demandés pour la transmission.
- ❑ La **compression** est la solution pour ce problème. Elle consiste à transformer les données vers une **forme moins gourmande en espace**.

2. Définition et intérêt de la compression



- ❑ Action de transformer des données en une représentation **plus compacte** :
 - Occupent **moins d'espace** qu'avant compression
 - Il n'existe pas de compresseur absolu
- ❑ La compression est devenue indispensable pour deux raisons majeures :
 - Les unités de stockage ont des **capacités limitées** ; ces ressources doivent être gérées de sorte qu'elles soient capables d'enregistrer une quantité d'information qui théoriquement dépasse leurs capacités.
 - La transmission d'information est d'autant **plus performante** si la quantité de données à transmettre est faible.



3. Mesures de compression



Plusieurs **critères** sont adoptés **pour mesurer** la **performance** d'une méthode de compression à savoir :

- **Le taux de compression** : Il est défini par le rapport de la taille du fichier compressé sur la taille du fichier initial et exprimé en pourcentage.

$$Taux = \frac{\text{taille après compression}}{\text{taille avant compression}}$$

- **Quotient de compression** : Il est l'inverse du taux de compression.
- **Gain de compression** : Le gain représente le gain en espace après compression.

Il est exprimé par :

$$\text{Gain} = 1 - \frac{\text{taille après compression}}{\text{taille avant compression}}$$

3. La compression sans perte



- La **compression sans perte** utilise des algorithmes qui compressent les données **sans les dégrader**. Autrement dit, après la décompression des données, celles-ci sont **identiques** à celles utilisées pour la compression.
- Ces algorithmes de compactage sont utilisés pour compresser tous types de données : des données textuelles, des images, du son, des vidéos, etc.
 - Le codage de **Huffman**
 - Le codage de **Shannon-Fano**
 - Le Run-length encoding ou **RLE**
 - La famille des Lempel-Ziv (**LZ77, LZ78, LZW**)

3.1 Le Codage de Shannon-fanon



- Antérieur au codage d'Huffman,
- l'arbre est construit en partant du sommet c'est à dire du MSB pour le codage.
- Il est basé sur la recherche de groupes de probabilités les plus proches possibles.
- Ordonner les symboles de la source en probabilités décroissantes
- Séparer les symboles en deux groupes de probabilités les plus proches possibles.
- Le MSB du groupe supérieur sera "0" et celui du groupe inférieur "1".
- Recommencer l'opération dans chaque sous-groupe pour affecter les bits suivants et ce jusqu'à séparation de tous les symboles de la source.

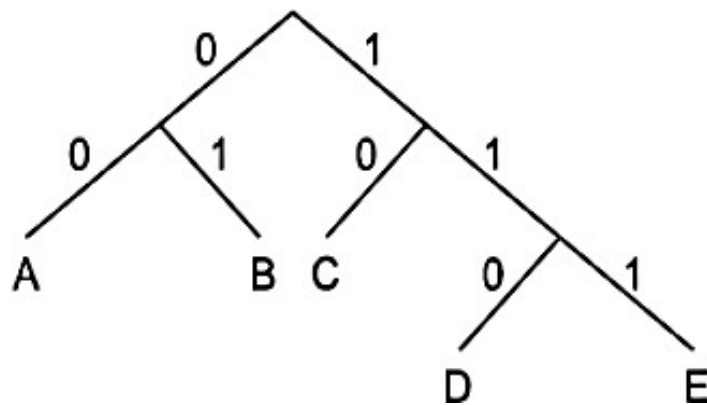
3.1 Le Codage de Shannon-fanon



Exemple

Soit la figure suivante qui représente la fréquence d'apparition des symboles d'un **Alphabet = A, B, C, D, E**. Les statistiques sont extraites a partir d'un texte :

Symbole	A	B	C	D	E
Fréquence	15 / 39	7 / 39	6 / 39	6 / 39	5 / 39



Symbole	Mot de code
A	00
B	01
C	10
D	110
E	111

3.1 Le Codage de Shannon-fanon



Avantages :

- Produit des codes préfix.
- Reduction importante dans la taille du fichier à compresser.

Inconvénients :

- la taille des codes augmente avec le nombre de symboles qui existent dans l'alphabet utilisé.

3.2 Le Run-length encoding ou RLE



- ❑ L'encodage **RLE** consiste à remplacer toutes les suites de caractères ou bits pareils par un nombre représentant le nombre de répétitions du caractère ou bits suivit de celui-ci.

Exemple 1

Soit la séquence de caractères suivante : **AAAAAAAAARRRRRROOOO** qui est codée sur **15 octets**. En appliquant la fonction de codage RLE on obtient la séquence suivante : **7A5R3O** qui code sur **6 octets**, de ce fait on obtient **un gain de 9 octets**.

3.2 Le Run-length encoding ou RLE



Exemple 2

Soit un fichier PGM :



```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 3 3 3 3 0 0 7 7 7 7 0 0 11 11 11 11 0 0 15 15 15 15 0
0 3 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0 0 15 0 0 15 0
0 3 3 3 0 0 0 7 7 7 0 0 0 11 11 11 0 0 0 15 15 15 15 0
0 3 0 0 0 0 0 7 0 0 0 0 0 11 0 0 0 0 0 15 0 0 0 0
0 3 0 0 0 0 0 7 7 7 7 0 0 11 11 11 11 0 0 15 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

168 valeurs

On obtient :

```
25 0 4 3 2 0 4 7 2 0 4 11 2 0 4 15 2 0 1 3 5 0 1 7 5 0 1 11 5 1 1
15 2 0 1 15 2 0 3 3 3 0 3 7 3 0 3 11 3 0 4 15 2 0 1 3 5 0 1 7 5 0
1 11 5 0 1 15 5 0 1 3 5 0 4 7 2 0 4 11 2 0 1 15 26 0
```

86 valeurs

3.2 Le Run-length encoding ou RLE



Avantages :

- Le principal avantage de l'encodage RLE est son immense simplicité.
- Il peut de plus être utilisé sur n'importe quel type de fichier ou après d'autres algorithmes de compression plus évolués pour gagner quelques octets de plus.

Inconvénients :

- L'encodage RLE possède un désavantage de taille. Le fichier ou texte à compresser doit contenir plusieurs chaînes de caractères ou bits répétés pour être d'une quelconque utilité.

"REELLEMENT" ➡ "1R2E2L1E1M1E1N1T" gain négatif

3.3 La famille des Lempel-Ziv



- ❑ Cette famille est un ensemble de techniques de compression sans perte qui utilisent **un dictionnaire** et où les codes signifient les indices des séquences d'octets placées dans le dictionnaire.
- ❑ Ce sont Abraham Lempel et Jakob Ziv qui ont inventé, en 1977, le premier algorithme de compression de cette famille sous le nom de **Lempel-Ziv-77** ou **LZ77**. LZ77 connut rapidement une nouvelle version : **le LZ78**. Dans les années 80, Terry Welch modifia les LZ* pour créer le **Lempel-Ziv-Welch** ou **LZW**.

3.3 La famille des Lempel-Ziv (LZ77)



3.3.1 LZ77 :

- L'algorithme LZ77 est un algorithme de compression sans perte utilisant une fenêtre glissante.
- l'algorithme LZ77 utilise une fenêtre glissante comme l'a majorité des algorithmes de la famille LZ. Cette fenêtre glissante est coupée en deux parties, une partie qui est le tampon et une autre qui le dictionnaire. Le tampon est la partie lecture et le dictionnaire est la partie recherche.

H E L L O H E L L O

3.3 La famille des Lempel-Ziv (LZ77)

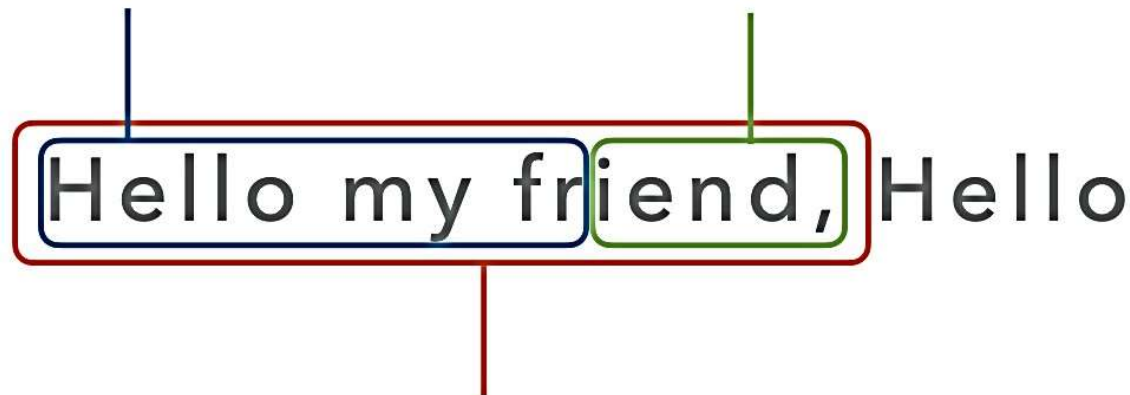


Le **tampon** est la partie lecture où se trouve le motif à encodé.

Le **dictionnaire** est la partie recherche.

Dictionnaire ou search buffer

Tampon ou look ahead buffer



Fenêtre glissante ou sliding window

Le but de cet algorithme est d'encoder une récurrence qui se trouve dans le tampon et le dictionnaire par la combinaison **position/taille**.

3.3 La famille des Lempel-Ziv (LZ77)



Original Hello friends, Hello world

Encoded Hello friends, (15,6)world

15
Hello friends, Hello world
6

- Le fonctionnement est simple, trouver le plus grand motif du tampon dans le dictionnaire.

Dans le tampon = "AABBCC"

Dans le dictionnaire = "RRYYIIAANNMMXX"

On commence à chercher:

1. AABBCC NOT FOUND
2. AABBC NOT FOUND
3. AABB NOT FOUND
4. AAB NOT FOUND
5. AA FOUND RRYYYIIAANNMMXX

➤ On calcule la distance du motif trouvé jusqu'au tampon de lecture.

“RRYYII**AANNMMXX**” = 8

3.3 La famille des Lempel-Ziv (LZ77)



- La longueur du motif trouvé est 2 (**AA**) et le prochain caractère du motif dans le tampon est B alors pour l'encodage on obtient **(8, 2, B)**.
- Si aucun motif est trouvé, l'encodage sera comme ça **(0, 0, [premier caractère du tampon])**, alors si aucun motif est trouvé dans l'exemple précédent, l'encodage sera **(0, 0, A)**.
- Pour résumer, si on trouve un **motif ([distance], [taille], [prochain caractère])** et si aucun motif est **trouvé (0, 0, [premier caractère du tampon])**.

3.3 La famille des Lempel-Ziv (LZ78)



3.3.2 LZ78 :

- Le principe de **LZ78** consiste à parcourir la source en apprenant les suites de symboles les plus fréquentes. Ces suites de symboles, appelées préfixes sont stockées dans un dictionnaire, construit au fur et à mesure que l'on parcourt la source. Plus des suites de symboles sont fréquentes, plus elles donneront lieu à des longs préfixes dans le dictionnaire.
- Le dictionnaire grandit au fur et à mesure qu'on parcourt la source. Pour des sources très longues, cela peut devenir coûteux en mémoire et temps de recherche, c'est pourquoi souvent on fixe une taille maximale, au delà de laquelle on arrête d'y ajouter des symboles.

3.3 La famille des Lempel-Ziv (LZ78)



Exemple

Séquence : wabba□wabba□wabba□wabba□woo□woo□woo

□ désigne l'espace.

Résultat d'encodage	Dictionnaire Indice	Entrée	Résultat d'encodage	Dictionnaire Indice	Entrée
<0, c(w)>	1	w	<4, c(□)>	10	ba□
<0, c(a)>	2	a	<9, c(b)>	11	wabb
<0, c(b)>	3	b	<8, c(w)>	12	a□w
<3, c(a)>	4	ba	<0, c(o)>	13	o
<0, c(□)>	5	□	<13, c(□)>	14	o□
<1, c(a)>	6	wa	<1, c(o)>	15	wo
<3, c(b)>	7	bb	<14, c(w)>	16	o□w
<2, c(□)>	8	a□	<13, c(o)>	17	oo
<6, c(b)>	9	wab			

Il faut stopper la croissance du dictionnaire à un certain point

3.3 La famille des Lempel-Ziv-Welsh (LZW)



3.3.3 LZW :

- **LZW** est une autre amélioration de l'**algorithme LZ78**, celui-ci a éliminé l'utilisation des tuples, il utilise seulement les index des séquences dans le dictionnaire.
- Utilisation d'un dictionnaire
- Remplacement des facteurs répétés par leur indice dans le dictionnaire
- Le dictionnaire est clos par préfixe : chaque préfixe d'un mot du dictionnaire est aussi présent dans le dictionnaire
- Le dictionnaire est initialisé avec tous les caractères de l'alphabet

3.3 La famille des Lempel-Ziv-Welsh (LZW)



Exemple

Séquence: wabba□wabba□wabba□wabba□woo□woo□woo

- Alphabet $A = \{\square, a, b, o, w\}$.

Dictionnaire Indice	Entrée
1	□
2	a
3	b
4	o
5	w
6	wa
7	ab
8	bb
9	ba

Dictionnaire Indice	Entrée
10	a□
11	□w
12	wab
13	bba
14	a□w
15	wabb
16	ba□
17	□wa
18	abb

Dictionnaire Indice	Entrée
19	ba□w
20	wo
21	oo
22	o□
23	□wo
24	oo□
25	□woo

Code : 5 2 3 3 2 1 6 8 10 12 9 7 9 11 7 16 5 4 4 11 21 23 4

4. La compression avec pertes



- **La compression avec pertes** utilise des algorithmes qui compressent les données en les dégradant.
- Les algorithmes de compression destructrice sont utilisés pour compresser des **données graphiques, audio et vidéo**. Ils ne pourraient être appliqués sur des données textuelles ou sur celles d'un programme au risque de les rendre illisibles ou inexécutables.
 - **La transformée en cosinus discrète (DCT)**
 - **La compression par ondelettes**
 - **La compression fractale** (uniquement appliquée aux images)

4. La compression avec pertes (DCT)



4.1 La transformée en cosinus discrète (DCT) :

- La transformée en cosinus discrète est une fonction mathématique complexe dont il existe plusieurs variantes. Elle est utilisée pour la compression destructrice des données, notamment pour les sons aux formats Vorbis, WMA et MP3 et pour les images et les vidéos suivant les normes JPEG et MPEG.
- Lors de l'opération de compression des données dans ces différents formats, les fréquences qui ne correspondent pas à des données pertinentes pour l'oeil ou l'oreille humaine sont éliminées par l'algorithme.

4. La compression avec pertes (DCT)



- La transformation est réalisée, non pas sur l'image entière, mais sur des **blocs de 8 x 8 pixels**. En effet, les matrices doivent être carrées et une taille plus grande serait trop coûteuse en temps de calcul. A partir des valeurs **$n(i,j)$** du bloc considéré de $M \times M$ pixels, la transformation calcule les coefficients **$N(u,v)$** par :

$$N(u, v) = \frac{2}{M} c(u) c(v) \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} \cos \left[\frac{\pi}{M} u(i + 1/2) \right] \cos \left[\frac{\pi}{M} v(j + 1/2) \right] n(i, j)$$

avec :

$$c(0) = \frac{1}{\sqrt{2}} \quad \text{et} \quad \forall \alpha \neq 0, \quad c(\alpha) = 1 \quad \text{et} \quad (u, v) \in [0, M-1] \times [0, M-1]$$

4. La compression avec pertes (DCT)



La transformation inverse s'écrit :

$$n(i, j) = \frac{2}{M} \sum_{u=0}^{M-1} \sum_{v=0}^{M-1} c(u)c(v) \cos \left[\frac{\pi}{M} u(i + 1/2) \right] \cos \left[\frac{\pi}{M} v(j + 1/2) \right] N(u, v)$$

Le calcul de la DCT se réalise aisément en s'écrivant sous une forme matricielle

Considérons la matrice S des valeurs des pixels :

$$S = \begin{pmatrix} n(0, 0) & n(0, 7) \\ n(7, 0) & n(7, 7) \end{pmatrix}$$

On considère la matrice A , formée par les éléments a_{ij} :

4. La compression avec pertes (DCT)



$$a_{ij} = c(u) \sqrt{\frac{2}{M}} \cos\left(\frac{j\pi(i + 1/2)}{M}\right)$$

Cette matrice A s'écrit :

$$\begin{pmatrix} \frac{1}{\sqrt{8}} & \frac{1}{2} \cos\left(\frac{\pi}{16}\right) & \frac{1}{2} \cos\left(\frac{2\pi}{16}\right) & \dots & \frac{1}{2} \cos\left(\frac{7\pi}{16}\right) \\ \frac{1}{\sqrt{8}} & \frac{1}{2} \cos\left(\frac{3\pi}{16}\right) & \frac{1}{2} \cos\left(\frac{3 * 2\pi}{16}\right) & \dots & \frac{1}{2} \cos\left(\frac{3 * 7\pi}{16}\right) \\ \vdots & & & & \\ \frac{1}{\sqrt{8}} & \frac{1}{2} \cos\left(\frac{15\pi}{16}\right) & & & \frac{1}{2} \cos\left(\frac{15 * 7\pi}{16}\right) \end{pmatrix}$$

La transformation s'obtient par :

$$\mathbf{M} = \mathbf{A}^t \mathbf{x} \mathbf{S} \mathbf{x} \mathbf{A}$$

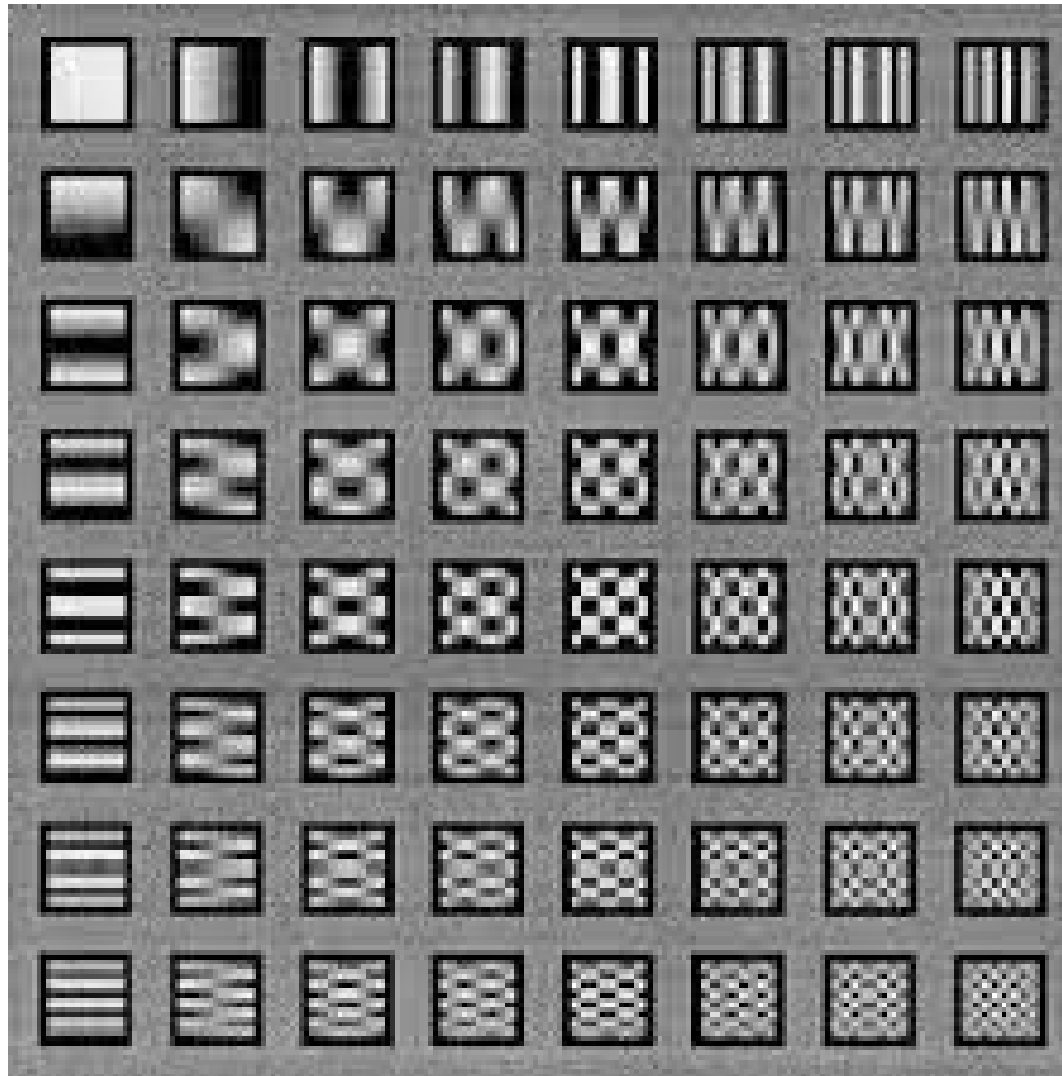
Les factorisations et développements spécifiques au cas 8x8 permettent d'effectuer la transformation en 40 multiplications et 56 additions (le calcul direct donne 512 multiplications et autant d'additions).

4. La compression avec pertes (DCT)



- La **DCT** est donc effectuée sur chaque matrice 8×8 de valeurs de pixels, et elle donne une matrice 8×8 de coefficients significatifs de fréquences spatiales : l'élément $(0,0)$ est représentatif de la valeur moyenne du bloc et est appelé coefficient DC.
- La matrice C transformée par DCT présente la propriété de regrouper les valeurs les plus élevées dans le coin supérieur gauche de la matrice (les valeurs devenant d'autant plus faibles que l'on s'approche du coin inférieur droite, donc que l'on se situe dans les " hautes fréquences "). Ainsi le maximum d'information sur l'image se trouve concentré sur la partie supérieure gauche de la matrice et la compression de l'image sera facilitée.

4. La compression avec pertes (DCT)



5. Normes de compression des images



- **JPEG (Joint Photographic Expert Group)** : norme ISO de compression d'images fixes avec pertes, standardisée en 1992.
- JPEG prend en compte les failles du système de perception humaine, de manière à réduire la quantité d'information de l'image qui sera codée.
- L'algorithme de compression repose principalement sur :
 - Une transformation mathématique de l'image la **DCT** (Discrete Cosine Transform)
 - Un codage à longueur variable le **codage de Huffman**

5. Normes de compression des images



- **JPEG2000 :**
 - Repose sur une analyse multi-résolution de l'image, appelée **la transformée en ondelettes**.
 - Meilleurs taux de compression.
 - Compression JPEG2000 pour les images fixes

6. Normes de compression vidéo



Qu'est-ce qu'une vidéo numérique ?

Vidéo = Son + Images

- **Caractéristiques importantes :**
 - Nombre d'images par secondes
 - Nombre de pixels par image
 - Codage de la couleur
 - Fréquence d'échantillonnage
 - Niveaux de quantification

➡ Ces caractéristiques influent sur le poids de la vidéo.

6. Normes de compression vidéo



- **Le poids d'une vidéo numérique non compressée est donné par :**

Poids des images = 3 couleurs × Nombre de pixels par image ×

Nombre d'images par seconde × Nombre de bits × Durée

Poids du son = Nombre de pistes × Fréquence d'échantillonnage ×

Nombre de bits × Durée

Poids total = Poids des images + Poids du son

- **Quelques données numériques :**

- **Nombre d'images par secondes :**

- Cinéma : 24 im/s
- Télévision : 25 im/s ou 30 im/s (dépend du format)

6. Normes de compression vidéo



- **Quantification**

- Image : 8/10 bits
- Son : 16/24 bits

- **Fréquence d'échantillonnage**

- Son : 44.1 KHz

❑ **La compression MJPEG :**

- **Moving JPEG (MJPEG)** n'est pas une norme !! C'est un algorithme propriétaire, donc inadaptable d'un appareil à l'autre.
- **MJPEG** consiste en la compression des images successives d'une vidéo de façon indépendante.

6. Normes de compression vidéo



□ La compression MPEG :

- La compression **MPEG** repose sur la **transformation DCT** de l'image, comme dans JPEG et MJPEG.
- L'algorithme part du principe que deux images successives dans une vidéo se ressemblent : on peut alors transmettre uniquement les informations qui ont changé entre les deux images.

■ La norme MPEG 1

- CD-ROM, disques optiques
- VCD (magnétoscope numérique en Chine)
- Utilisée par la TV numérique aux USA en attendant la mise au point du MPEG2

6. Normes de compression vidéo



- **La norme MPEG 2**

- La norme **MPEG2** a été mise au point pour le stockage et la diffusion de la vidéo numérique. C'est la norme de compression utilisée dans la télévision numérique terrestre (TNT ou DVB-T).

- **La norme MPEG 4**

- La TV numérique (compression audio/vidéo naturelle)
- Les applications graphiques 2D/3D (synthèse d'images)
- Le Web (interactivité, accès, transmission)
- Nouveauté majeure : découpage des scènes en « objets »

7. Normes de compression audio



- La compression audio **MPEG** exploite les limites de l'audition humaine.
- A une fréquence donnée, notre oreille ne peut percevoir un son qu'à condition que son niveau de pression acoustique soit supérieur à un seuil.

Le seuil est différent pour chaque fréquence de la bande audible.

- **La compression audio MPEG 1**

- Fréquences d'échantillonnage : 32 à 48 kHz
- Débits : 32 à 384 kbit/s

- **La compression audio MPEG 2**

- Fréquences d'échantillonnage : 16 à 24 kHz
- Débits qui peuvent descendre jusqu'à 8 kbits/s

7. Normes de compression audio



- **La compression audio MPEG 4**

- Débits de 2 à 64 kbit/s
- La norme n'a pas pu proposer 1 seul algorithme pour tous les types de sons (musique, parole, etc. ...)
- Elle inclut plusieurs standards de codage, adapté à chaque type de signal sonore