Informatique multimédia

✓ Compression des documents du multimédia
Formats de diffusion

Benoît Piranda
Équipe SISAR
Université de Marne La Vallée
piranda@univ-mlv.fr
http://www-igm.univ-mlv.fr/~piranda

Les algorithmes de compression

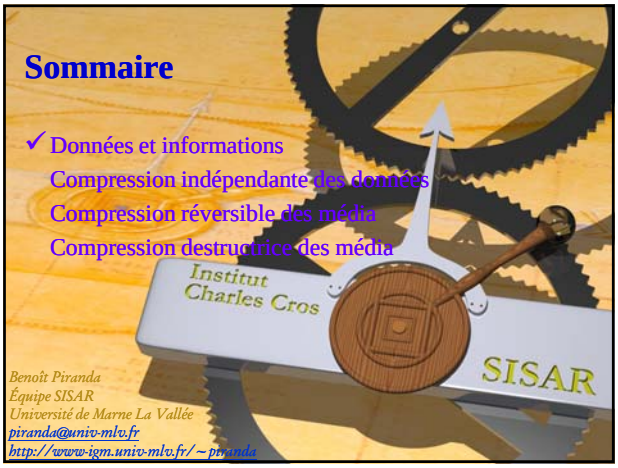
- **Compression de données quelconques**
  - **indépendante du type de données**
    - **compression sans perte**
    - **codage des événements plutôt que des données**
  - **exploitation des corrélations dans les données**
- **Compression de média**
  - **exploitation de la cohérence dans les données**
    - **son : cohérence temporelle**
    - **image : cohérence spatiale**
    - **animation : cohérence spatio-temporelle**
  - **compression sans perte**
    - **méthodes de codage de la cohérence**
  - **compression avec perte**
    - **suppression des informations peu significatives**

Principes de compression

- **Compression de données sans perte**
  - **Réduire la taille des données en supprimant des redondances**
  - ☑ **Processus réversible,**
  - ☑ **Valable pour tout type de données,**
  - ☒ **Gain théoriquement assez faible.**
- **Compression dégradante**
  - **Suppression des informations « peu importantes »**
  - ☒ **Compression non réversible**
  - ☑ **Gain de compression très grand**

Sommaire

✓ Données et informations
Compression indépendante des données
Compression réversible des média
Compression destructrice des média



Benoît Piranda
Équipe SISAR
Université de Marne La Vallée
piranda@univ-mlv.fr
http://www-igm.univ-mlv.fr/~piranda

La compression de données

- **Évaluation de la compression**
  - **quotient de compression *Q***

$$Q = \frac{\text{Taille initiale}}{\text{Taille compressée}}$$

  - **taux de compression T (exprimé en pourcentage)**

$$T = \frac{1}{Q}$$

  - **gain de compression G**

$$G = 1 - T$$

Information et signal

- **Source ou signal**
  - **tout processus générant de l'information**
  - **séquence finie ou infinie d'événements**
- **Les événements**
  - **sont codés sur un alphabet (liste des valeurs possibles)**
- **Différents types de source**
  - **texte :**
    - **symbole = les lettres de l'alphabet latin, grec…**
  - **signal numérique**
    - **symbole = bits ou octets**
  - **son, image numérique**
    - **symbole = échantillons**

Source Markovienne

- **Source markovienne (théorique)**
  - **source aléatoire**
  - **pas de cohérences entre les symboles**
  - **ordre de la source markovienne : nombre de caractères minimum formant des symboles non corrélés.**

abacbbabbbacab... Alphabet : {a,b,c}

14 caractères

abacbbabbbacab... Alphabet : {ab,bb,ac}

7 caractères

L'information dans les données

- **L'information réside dans la succession des données**
  - **signal, chaîne ou séquence de symboles**
- **Quantité d'information**
  - **liée à l'entropie du signal**
    - **entropie : incertitude entre deux messages successifs**
  - **données constantes**
    - **entropie nulle**
    - **information très réduite**

Un exemple : le scrabble

- **Jeu consistant à écrire des mots qui rapportent des points en sommant les valeurs associées à chacune des lettres qui le compose.**
- **Plus une lettre est rare dans la langue considérée plus le nombre de points associé est grand**
  - **lettres rares de la langue françaises (K W X Y Z) : 10 points,**
  - **lettres courantes : 1 points**

Probabilité et occurrence

- **Considérons une source finie de type texte**

`Ecole nationale des ponts et chaussées`

  - **alphabet : 16 symboles**
- **Occurrence et probabilité de chacun des symboles**

Lettre	occurrence	probabilité	Lettre	occurrence	probabilité
a	3	7,89%	o	3	7,89%
c	2	5,26%	p	1	2,63%
d	1	2,63%	s	5	13,16%
e	5	13,16%	t	3	7,89%
h	1	2,63%	u	1	2,63%
i	1	2,63%	é	1	2,63%
l	2	5,26%	_	5	13,16%
n	3	7,89%	E	1	2,63%

Sommaire

Données et informations

✓ Compression indépendante des données

Compression réversible des média

Compression destructrice des média

Benoît Piranda
Équipe SISAR
Université de Marne La Vallée
piranda@univ-mlv.fr
http://www-igm.univ-mlv.fr/~piranda

Compression indépendante des données

- **Codage entropique ou Code à longueur variable (VLC)**
- **Idée : codage statistique à longueur variable**
  - **coder les données fréquentes par un codage plus court**
- **Compression réversible**
- **Utilisation courante en complément d'autres méthodes**
  - **PhotoCD,**
  - **JPEG,**
  - **MPEG Audio**
  - **MPEG**

Codage entropique

- **Méthode**
  - codage de l'alphabet par des données de longueurs variables
  - décodage capable de retrouver la séparation entre les symboles de codage
    - caractère spécial entre les symboles
    - structure des codes permettant de retrouver leur début
- **Méthode de Shanon-Fano**
- **Méthode de Huffman**
- **Codage arithmétique**

Algorithme du codage de Huffman

- **Optimisation de la méthode de Shanon-Fano**
- **Etude statistique des données**
  - Ordonner dans une tables, les symboles par ordre d'occurrence
- **Construction de l'arbre**
  - Définir un arbre binaire contenant les symboles
  - Plus les symboles sont rares plus ils sont placés profondément dans l'arbre
- **Construction du code**
  - À chaque symbole on associe un code binaire dont la taille dépend de la profondeur
  - Code permettant le décodage des symboles (frontière entre les codes)

Algorithme de Huffman par l'exemple

- **Codage d'une chaîne ADN**
  - Alphabet A={A,C,G,T} codables sur 2 bits
  - Séquence à compresser : 48 lettres soit 96 bits

TTTCATTGTACCTGTTATTAGATTAGTCTAGATTTAGTACTATATCTT

- **Etude des données**
  - Fréquence d'apparition de chaque lettre :

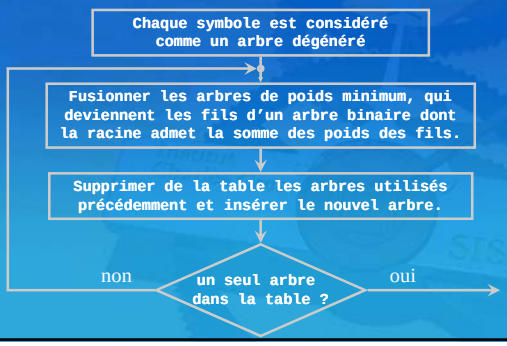
$$p_A = \frac{12}{48}, p_C = \frac{6}{48}, p_G = \frac{6}{48}, p_T = \frac{24}{48}$$

  - Table des symboles ordonnés (et nombre d'occurrences) :

C(6)	G(6)	A(12)	T(24)

Algorithme de Huffman par l'exemple

- **Construction de l'arbre**



Algorithme de Huffman par l'exemple

C(6)	G(6)	A(12)	T(24)

Étape 1 :

Étape 2 :

Étape 3 :

Étape 4 :

Algorithme de Huffman par l'exemple

- **Construction des codes**
  - association des valeurs binaires
    - 0 aux brins de gauche
    - 1 aux brins de droite
  - le code associé au symbole est obtenu en mémorisant les valeurs des brins parcourus pour aller de la racine de l'arbre à la feuille contenant le symbole.

T → 0
A → 10
C → 110
G → 111

Algorithme de Huffman par l'exemple

- **Codage d'une chaîne ADN**
  - **Alphabet A={A,C,G,T} codables sur 2 bits**
  - **Séquence à compresser : 48 lettres soit 96 bits**

TTTCATTGTACCTGTTATTAGATTAGTCTAGATTTAGTACTATATCTT

- **Résultat**

T →0 A →10 C →110 G →111

0 0 0 1 1 0 1 0 0 0 1 1 1 0 1 0 1 1 0 1 1 0 0 0 1 1 1 0 0 1 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1 0 1 0 1 1 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1 0 1 0 1 1 0 0 1 0 0 1 0 0 1 1 0 0 0

Résultat compressé : 84 bits soit un gain de 12,5 %

Algorithme de décodage de Huffman

- **Reconstruction de la source d'origine**
- **Besoin**
  - **Codage de chaque symbole**
  - **Données compressées**
- **Algorithme de décodage**
  - **Lecture de gauche à droite des données compressées**
  - **Identification unique du code de chaque symbole compressé lors de la lecture**
  - **Reconstruction itérative de la source décompressée**

Décodage de Huffman sur un exemple

0→T 10→ A 110→ C 111→G

0 1 1 0 1 0 0 0 . . .

0→T T

0 1 1 0 1 0 0 0 . . .

1→? ?→C TC

: : :

Codage arithmétique

- **Principe**
  - **Étude statistique de chaque symbole**
  - **Chaque symbole s_i est associé à un intervalle**
    - **Compris dans [0,1[**
    - **De longueur $p(s_i)$**
  - **Le codage d'une séquence**
    - **Générer une suite d'intervalles encadrant un réel codant la séquence de façon unique**
- **Utilisation**
  - **Remplace le codage de Huffman dans les algorithmes récents**
    - **JPEG 2000**

Codage arithmétique

- **Calcul du réel codant une séquence**

$$V_{s_i} = \sum_{l=1}^{i-1} p(S_l)$$

$$V_{S_iS_j} = \sum_{l=1}^{i-1} p(S_l) + p(S_i)\sum_{m=1}^{j-1} p(s_m)$$

$$V_{S_iS_jS_k} = \sum_{l=1}^{i-1} p(S_l) + p(S_i)\left(\sum_{m=1}^{j-1} p(s_m) + p(S_k)\sum_{n=1}^{k-1} p(s_n)\right)$$

Codage arithmétique

- **Exemple de stockage par intervalle**
  - **séquence S={ABACBACA}**
  - **p(A)=0.5, p(B)=0.25, p(C)=0.25**

Symbole	Binf	Bsup	Emission
init	0	1	
A	0	0.5	
B	0.25	0.375	
A	0.25	0.3125	
C	0.296875	0.3125	
B	0.30046875	0.30859375	30
décalage	0.46875	0.859375	
A	0.46875	0.664602	
C	0.615774	0.664602	6
décalage	0.15774	0.64602	
A	0.15774	EOF	15774

Code final émis : 30615774

Décodage arithmétique

- Symétrique au codage
  - Utilisation de la formule de codage du réel codant la séquence

$$V_{S_i S_j S_k} = \sum_{l=1}^{i-1} p(S_l) + p(S_i)\left(\sum_{m=1}^{j-1} p(s_m) + p(S_k)\sum_{n=1}^{k-1} p(s_n)\right)$$

  - besoin de connaître la fréquence de chaque symbole

```
  0 < 0,30615774 < 0.5  → A → (0,30615774 - 0  ) / 0.5
0.5 < 0,61231548 < 0.75 → B → (0,61231548 - 0.5) / 0.25
  0 < 0,44926192 < 0.5  → A → (0,44926192 - 0  ) / 0.5
0.75< 0,89852384 < 1.0  → C → (0,89852384 - 0.75)/ 0.25
0.5 < 0,59409536 < 0.75 → B → (0,59409536 - 0.5) / 0.25
  0 < 0,37638144 < 0.5  → A → (0,37638144 - 0  ) / 0.5
0.75< 0,75276288 < 1.0  → C → (0,75276288 - 0.75)/ 0.25
```

  - possibilité de définir un algorithme n'utilisant que des entiers

Méthodes à base de dictionnaires

- Principe
  - Recherche des corrélations dans la source
  - Codage des ensembles de symboles corrélés
- Utilisation de dictionnaires
  - statiques
    - connaissance à priori des corrélations
    - statistique sur des sources du même type
  - dynamiques ou adaptatifs
    - acquisition du dictionnaire au cours de la lecture de la source

Compression à base de dictionnaires

- Codage par dictionnaires statiques
  - Connaissances préalables
    - ensemble des échantillons des données
    - statistique à priori sur les données
- Exemple : compresser un texte en français
  - statistique sur la fréquence d'apparition des lettres
    - ' e ' très probable
  - statistique sur la fréquence de succession des lettres
    - ' q ' suivit de ' u ' très probable

Compression à base de dictionnaires

- Codage par dictionnaire dynamique
  - Méthode de Lempel-Ziv-Welch (LZ77, LZ78, LZW)
    - dictionnaire dynamique construit au cours de la lecture du fichier
    - une seule lecture est nécessaire
  - utilisation
    - compression de fichiers gzip, Pkzip, Arj...,
    - sauvegarde MS Backup, Norton Backup…
    - compresseur temps réel : stacker, Dblspace,…
    - compression d'images : option du format TIFF

Méthode Lempel-Ziv 77

- Principe
  - utilisation d'une fenêtre sur les données
    - zone des données utilisée pour rechercher des corrélations
    - visualisant N symboles, divisée en deux parties
      - ✓ B_R la partie recherche N_R (~4096)
      - ✓ B_E la partie encodage N_E (~128)
  - rechercher dans B_R la plus longue sous chaîne égale à la plus longue suite de symbole de B_E
  - constitue le triplet <a,l,c>
    - a = adresse du début
    - l = longueur de la séquence identique
    - c = nouveau caractère ajouté au dictionnaire
  - résultat : liste des triplets <a,l,c>

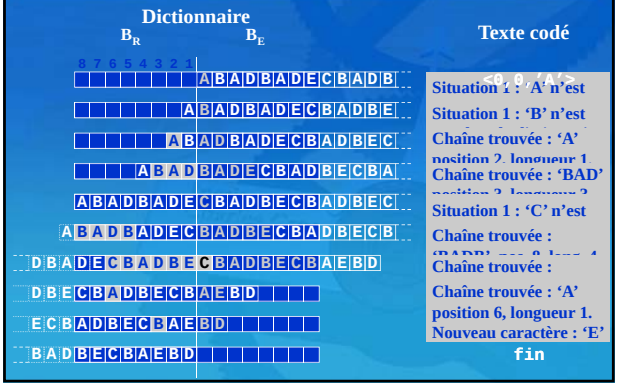
Méthode Lempel-Ziv 77 par l'exemple

- Exemple de compression d'une chaîne de caractères
  - S={ABADBADECBADBECBADBECBAEDB}
  - avec N_R=8 et N_E=8
- Quantité de données
  - source : 1 octet par caractère
  - triplet de compression : 14 bits
    - adresse : 3bits
    - longueur : 3 bits
    - caractère suivant : 1 octet

Algorithme Lempel-Ziv 77

- Lecture du buffer B_E
- Trois situations
  - Pas de chaîne commune
    - par exemple à l'initialisation : buffer vide
    - caractère codé par le triplet <0,0,'A'> :
      - ✓ position à droite de B_R (0),
      - ✓ longueur de la chaîne trouvée = 0
      - ✓ puis le caractère lui même
  - Plusieurs chaînes de B_R correspondent à B_E
    - on choisit la plus longue placée la plus à gauche de B_R
  - La chaîne la plus longue déborde sur B_E
    - cas critique
- Décalage de la fenêtre de l+1

Méthode Lempel-Ziv 77 par l'exemple

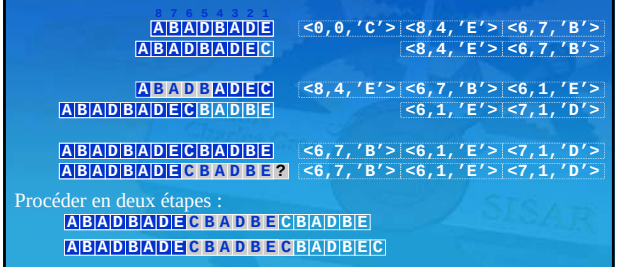


Méthode Lempel-Ziv 77

- Bilan de la compression précédente
  - 9 triplets de 14 bits soit 126 bits
  - source de 26 octets soit 208 bits
  - gain de 39,4 %
- Avantages et inconvénients
  - pas besoin de données supplémentaires
  - mémoire courte :
    - ce qui sort de la fenêtre de recherche est oublié
    - fenêtre de taille réduite, définie à priori
  - décodage simple et rapide

Décodage de Lempel-Ziv 77

- Le dictionnaire est reconstruit de gauche à droite à partir des triplets



Méthode Lempel-Ziv 78

- Principe
  - Ajout d'un dictionnaire explicite à LZ77
  - Le dictionnaire contient tous les motifs trouvés
- Algorithme
  - On construit les couples <a,c>
    - a : adresse du plus grand motif du dictionnaire
    - c : nouveau caractère
    - plus besoin de coder la longueur
  - on ajoute le nouveau motif au dictionnaire
    - motif trouvé + nouveau caractère

Méthode Lempel-Ziv 78 par l'exemple

Exemple de compression d'une chaîne de caractères

A B A D B A D E C B A D B E C B A D B E C B A D B E C D

Entrée		Dictionnaire		Sortie
motif	Car. suiv.	index	Motif	couple
	A	1	A	<0,'A'>
	B	2	B	<0,'B'>
A	D	3	AD	<1,'D'>
B	A	4	BA	<2,'A'>
	D	5	D	<0,'D'>
BA	D	6	BAD	<4,'D'>
	E	7	E	<0,'E'>
	C	8	C	<0,'C'>
BAD	B	9	BADB	<6,'B'>
E	C	10	EC	<7,'C'>
BADB	E	11	BADBE	<9,'E'>
C	B	12	CB	<8,'B'>
AD	B	13	ADB	<3,'B'>
EC	D	14	ECD	<10,'D'>

Décodage Lempel-Ziv 78

- **Décodage symétrique au codage**
  - **données**
    - **Couples**
    - **dictionnaire**

A <0,'A'>,<0,'B'>...
A B <0,'B'>,<1,'D'>...
A B A D <1,'D'>,<2,'A'>...
A B A D B A <2,'A'>,<0,'D'>...

Couple
<0,'A'>
<0,'B'>
<1,'D'>
<2,'A'>
<0,'D'>
<4,'D'>
<0,'E'>
<0,'C'>
<6,'B'>
<7,'C'>
<9,'E'>
<8,'B'>
<3,'B'>
<10,'D'>

Dictionnaire	
1	A
2	B
3	AD
4	BA
5	D
6	BAD
7	E
8	C
9	BADB
10	EC
11	BADBE
12	CB
13	ADB
14	ECD

Sortie	
	A
	B
A	D
B	A
	D
BA	D
	E
	C
BAD	B
E	C
BADB	E
C	B
AD	B
EC	D

Méthode de Lempel-Ziv-Welch

- **Variante de LZ78**
  - **Dictionnaire prédéfini initialisé avec tous les symboles de l'alphabet**
    - **partie statique**
    - **nouveau symbole = index du dictionnaire**
  - **résultat : liste d'adresses dans le dictionnaire**
- **Fait l'objet d 'un brevet**

Méthode LZW par l'exemple

- **Exemple de compression d'une chaîne de caractères**

A B A D B A D E C B A D B E C B A D B E C B A D B E C D

Partie statique du dictionnaire

index	0	1	2	3	4	5
symbole	∅	A	B	C	D	E

Entrée	
motif	Car. suiv.
∅	A
A	B
B	A
A	D
D	B
B	A
BA	D
D	E
E	C
C	B
:	:

Dictionnaire	
Index	Motif
6	AB
7	BA
8	AD
9	DB
10	BAD
11	DE
12	EC
13	CB
:	:

Sortie
adresse
1
2
1
4
7
4
5
2
:

Sommaire

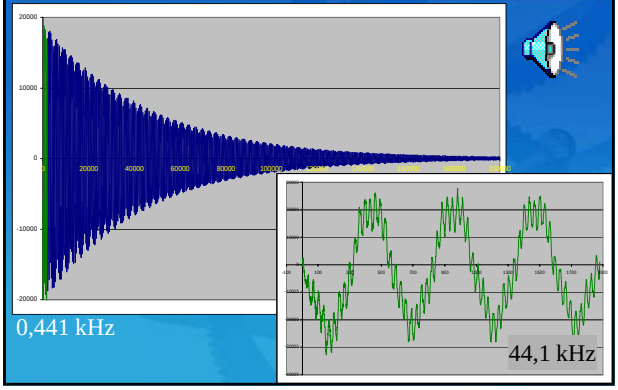
Données et informations
Compression indépendante des données
✓ Compression réversible des média
Compression destructrice des média

Benoît Piranda
Équipe SISAR
Université de Marne La Vallée
piranda@univ-mlv.fr
http://www-igm.univ-mlv.fr/~piranda

Compression de média

- **Compression sans perte du son**
- **Méthode des différences**
  - **Exploitation de la cohérence temporelle**
  - **Perte liée à cette méthode**

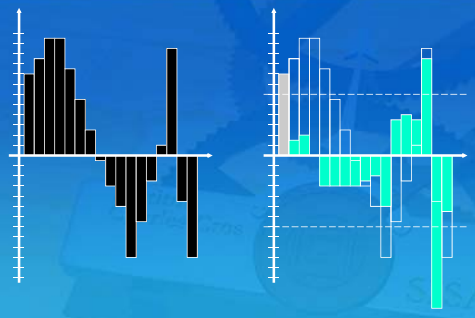
Sons de synthèse



Méthodes différentielles

- Codage de la différence entre deux échantillons successifs
  - grande probabilité de valeurs proches de 0
  - échantillons sur *n* octets sont codés sur *n*/2 octets
    - gain théorique 50%
    - perte des signaux hautes fréquences
  - propagation d'une erreur
  - besoin d'échantillons de référence placés régulièrement dans le fichier compressé.
- Couplé à un codage VLC

Méthode différentielle



échantillons différences

Perte d'informations dans la méthode

- Différences codée sur n/2 octets
  - perte des variations très fortes de l'amplitude
  - correspond à des hautes fréquences
- Décalage du signal lors de la décompression
  - besoin d'échantillons « clés » pour recaler

Compression de média

- Compression sans perte de l'image
- Méthode Run Length Encoding (RLE ou RLC)
  - choix d'une direction de parcours de l'image
  - exploitation de la cohérence spatiale le long de ce parcours

Codage par répétition

- Run Length Encoding (RLE ou RLC)
  - *n* octets successifs identiques de valeur *V* codés par *nV*
  - Performances moyennes mais implémentation très simple
    - bien adapté aux images sur fond uniforme (dessins)
  - utilisation
    - formats PCX, BMP (option RLE)

3 2 1 1 1 1 4 9 =14 octets = 112 bits

Optimisation de la méthode RLE

- Codage du nombre d'octets successifs sur 7 bits
  - valeurs de 2 à 129 (stocke n-2)
- Premier bit indique un pixel de couleur différent

3 2 1 1 1 1 4 9 =14 octets = 112 bits

00000001 00000000 1 1 1 00000010 00000111 = 91 bits

Exemple de compression

Dessin du clown 400x390 pixels en couleurs indexées sur une palette de 256 couleurs (1 octet), soit environ 152 Ko.
Taille du fichier compressé : 21 Ko. (Gain de 86%)

Sur les aplats, on code jusqu'à 128 pixels consécutifs identiques par 2 octets.

Inconvénient de la méthode

- **Cas défavorable :**
  - **fichier « compressé » plus lourd que le fichier brut**

Sommaire

Données et informations
Compression indépendante des données
Compression réversible des média
✓Compression destructrice des média

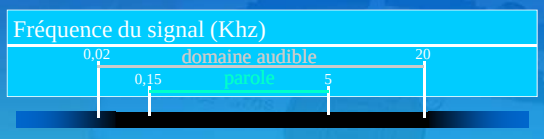
Benoît Piranda
Équipe SISAR
Université de Marne La Vallée
piranda@univ-mlv.fr
http://www-igm.univ-mlv.fr/~piranda

Compression du son avec perte

- **Méthode**
  - **suppression des données peu significatives**
- **Constatation**
  - **oreille sensible à une fenêtre de fréquences réduites**
- **Compression**
  - **suppression des fréquences inaudibles**
- **La méthode MPEG-Audio ou MP3**

Codage informatique du son

- **Fréquences audibles**

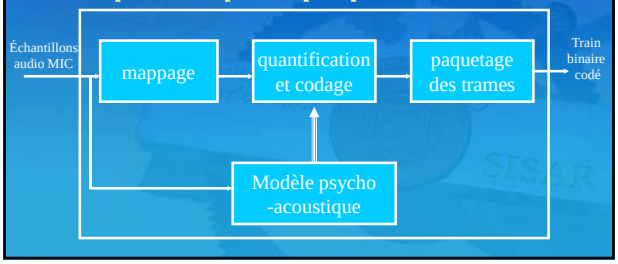


Compression du son

- **Principe**
  - **conserver uniquement les données pertinentes du son : les fréquences audibles**
  - **étapes**
    - **numérisation**
      - ✓ **échantillons sur 16 bits à 32, 44.1 ou 48 kHz**
    - **calcul du spectre fréquentiel (FFT)**
    - **conservation des fréquences audibles**
      - ✓ **entre 20 Hz et 20 kHz**
    - **codage**
- **Utilisation**
  - **MPEG Audio,**
  - **MUSICAM,**
  - **DOLBY AC3**

Algorithme MPEG Audio (MP3)

- **Algorithme Musicam (Masking pattern adapted Universal Subband Integrated Coding And Multiplexing)**
  - **algorithme non normalisé**
  - **prise en compte de la perception de l'oreille humaine**

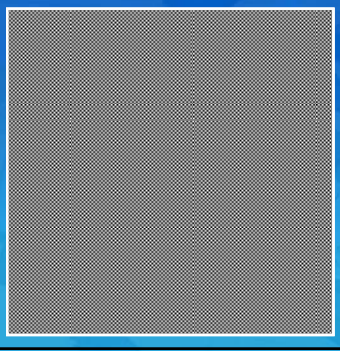


Compression d'image avec perte

- **Notion de fréquence dans une image**
  - **information codée : amplitude de luminance en fonction de la position le long de deux axes**
  - **fréquence des variations de l'amplitude suivant une direction**
- **Image discrétisée en pixels**
  - **variation de l'amplitude = différence entre deux échantillons successifs**
  - **utilisation : recherche de contours,**
  - **fréquence maximum : succession de deux pixels de valeurs extremum**
- **Passage au mode fréquentielle**
  - **Transformée en cosinus**

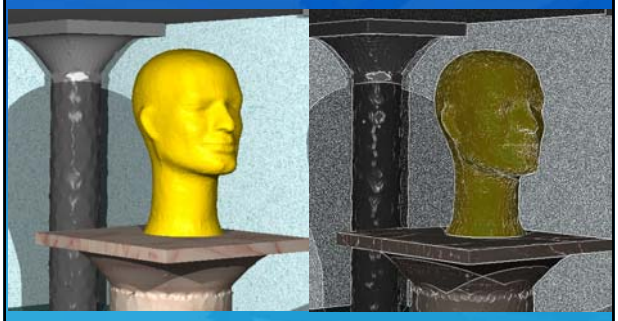
Illustration sur les fréquences

L'œil est un filtre passe bas

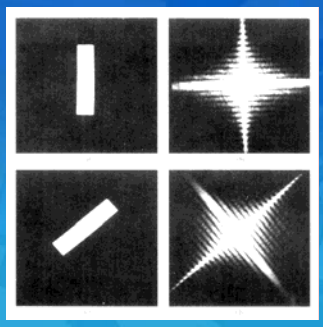


Hautes fréquences dans une image

Hautes fréquences dans une image



Exemple d'analyse spectrale d'une image



Transformée de Fourier