Réseaux Multimédia

© 2002 Damien Magoni

Toutes les illustrations
© 2001 Pearson Education Limited – Fred Halsall

Contenu

- Représentation des informations multimédia
  - Numérisation
  - Structure d’un encodeur
  - Structure d’un décodeur
  - Représentation du texte, des images et de l’audio/vidéo
- Compression du texte et des images
- Compression audio et vidéo
- Standards pour les communications multimédia

Partie 2

Compression du texte et des images

Principes de compression

- Encodeur source et décodeur destination peuvent être matériels ou logiciels
- Compression sans perte (réversible) : utilisé pour du texte ou des données
- Compression avec pertes : utilisé pour des images numériques et des flux audio ou vidéo
- Le codage entropique est sans perte et indépendant du type d'information qui est à compresser :
  - Encodage *run-length* (pour du binaire, on code le nombre d'occurrences)
  - Encodage statistique (on code des mots fréquents avec des petites valeurs)

Entropie de la source

- **Théorème de Shannon** : Le nombre moyen théorique minimal de bits requis pour transmettre un flot source est nommé **entropie** de la source et vaut :

$$H = -\sum_{i=1}^{n} P_i \times \log_2 P_i$$

- Ou *n* est le nombre de symboles différents et Pi est la probabilité d'apparition du symbole i.
- L'efficacité d'un encodage est calculé sous forme de ratio de l'entropie de la source sur le nombre moyen de bits par mot du code (= $\Sigma N_i P_i$)

Encodage par la source

- Encodage différentiel :
  - On ne code que les différences d'amplitude entre chaque valeurs successives du signal ce qui nécessite moins de bits par échantillons
  - Peut être avec ou sans perte
- Encodage par transformée :
  - On transforme le signal source en une forme plus facile à compresser
  - En général, pas de perte d'information dans la transformation
  - Utilisé pour les images et la vidéo
  - Notion de fréquence spatiale : une *Discrete Cosine Transform* (DCT) donne un spectre spatial d'une image, les HF peuvent être éliminées

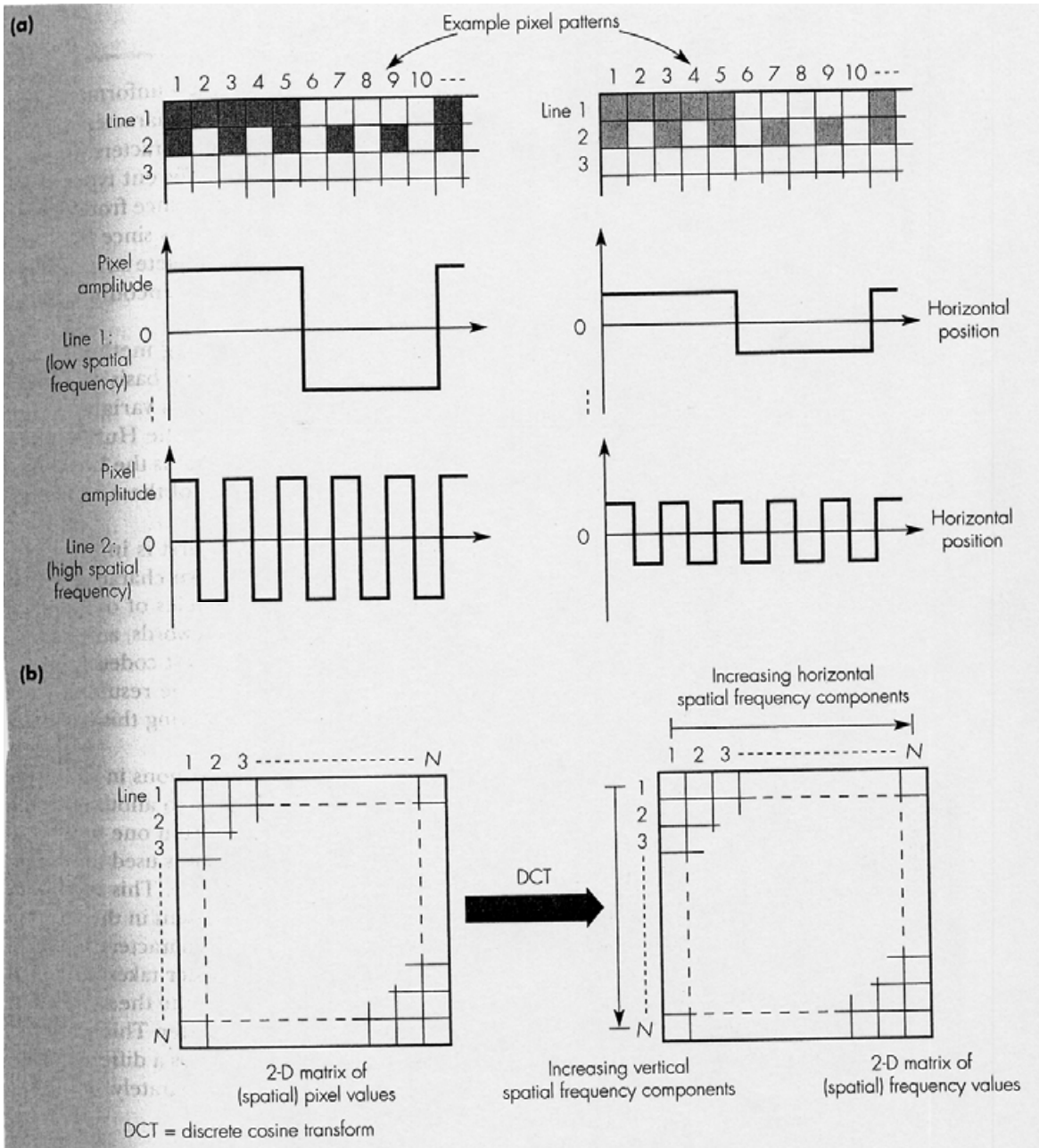


Figure 3.2 Transform coding: (a) example pixel patterns; (b) DCT transform principles.

Compression du texte

- Doit être sans perte (la modification d'un caractère peut changer le sens d'une phrase)
- Utilisation d'un codage entropique statistique avec des mots de code :
  - De caractères simples pour **Huffman**
  - De chaînes de caractères pour **Lempel-Ziv**
- Lorsqu'on connaît la fréquence d'apparition des mots du code dans un contexte donné, on peut faire du codage statique
- Dans le cas contraire, on réalise un codage dynamique ou adaptatif

Codage de Huffman statique

- Création d’un arbre (de codes) de **Huffman** qui définit le codage binaire de chaque symbole

- Un arbre de **Huffman** est un arbre binaire déséquilibré

- Aucun code n’est le préfixe d’un autre code plus long

- On attribue les codes binaires les plus courts aux symboles les plus fréquents

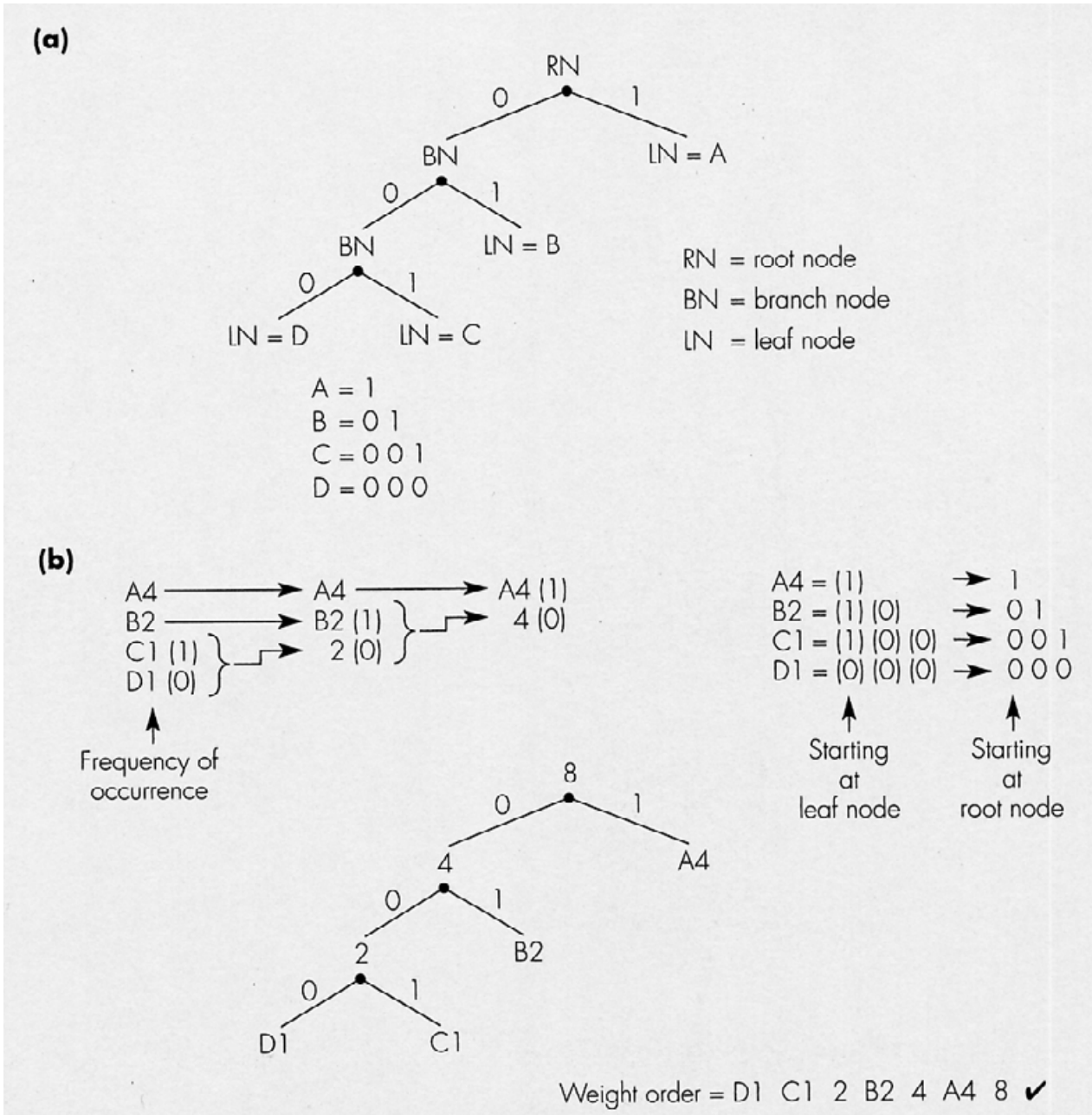


Figure 3.3 Huffman code tree construction: (a) final tree with codes; (b) tree derivation.

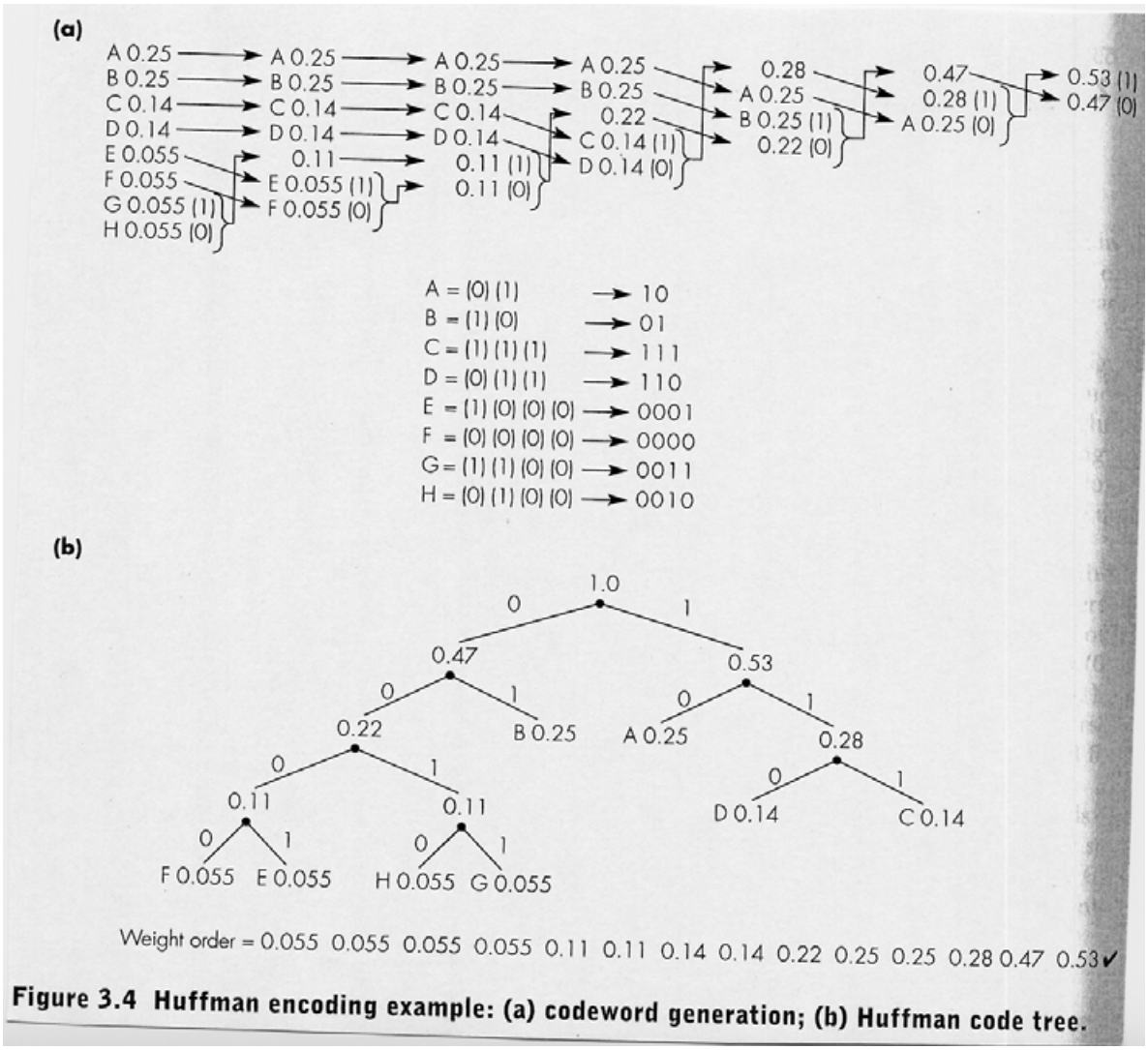


Figure 3.4 Huffman encoding example: (a) codeword generation; (b) Huffman code tree.

Décodage de Huffman statique

- Décodage orienté bit

- Possible grâce à la propriété du préfixe : aucun code binaire n'est le préfixe d'un autre code binaire plus long

- Le récepteur doit connaître les mots de code utilisés :
  - Soit ils sont transmis avant les données
  - Soit le récepteur et l'émetteur se mettent d'accord au préalable sur le jeu de codes à utiliser

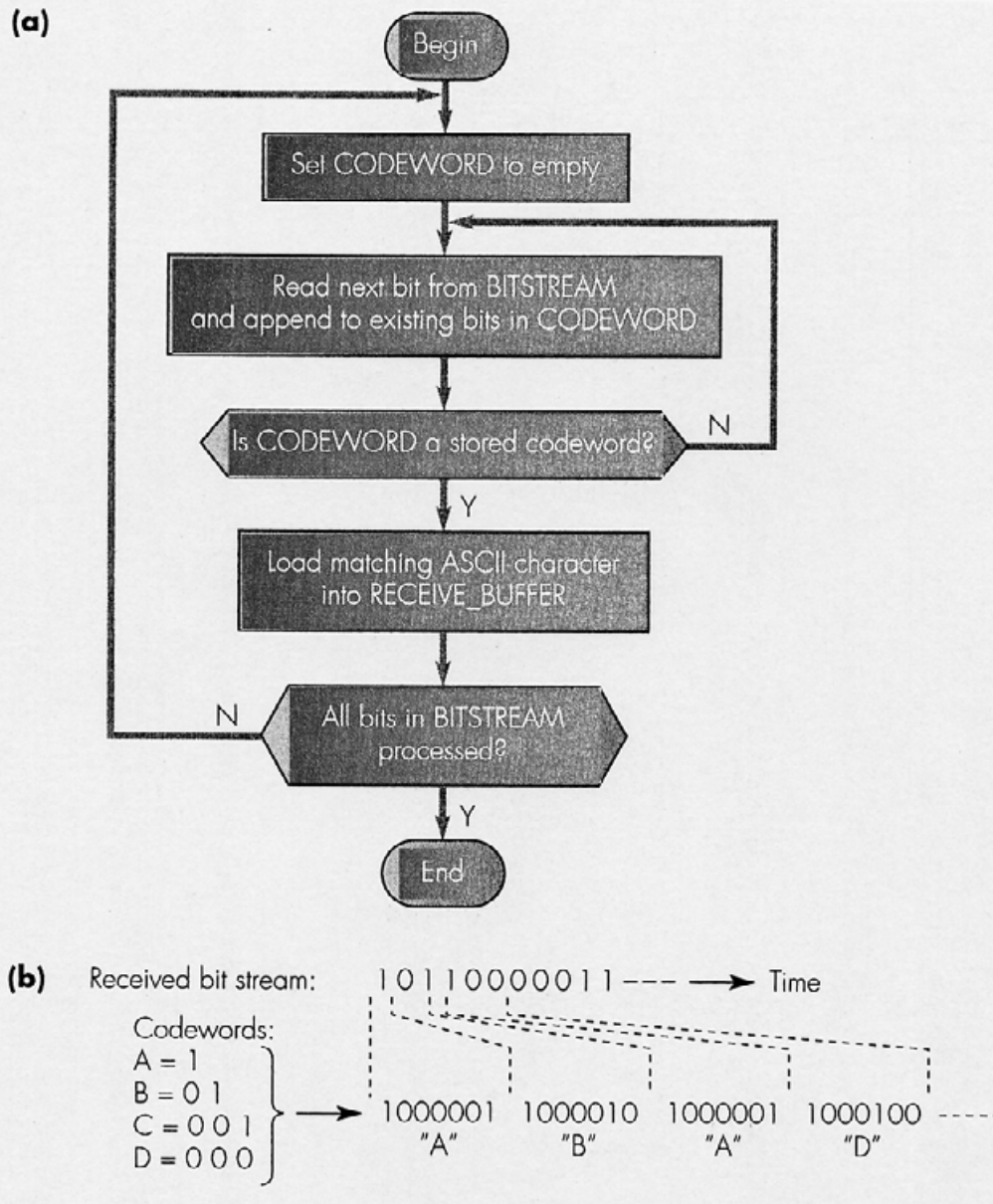


Figure 3.5 Decoding of a received bitstream assuming codewords derived in Figure 3.3: (a) decoding algorithm; (b) example.

Codage de Huffman dynamique

- L’encodeur et le décodeur construisent l’arbre de **Huffman** dynamiquement
- Si le caractère courant est présent dans l’arbre, son mot de code est envoyé, sinon il est transmis sous forme non compressée et l’encodeur met son arbre à jour
- Un caractère non compressé est précédé par le code de la feuille vide courante (e0)
- Le nombre d’occurrences des caractères est stocké afin de réorganiser les mots de code de façon optimale à chaque modification de l’arbre

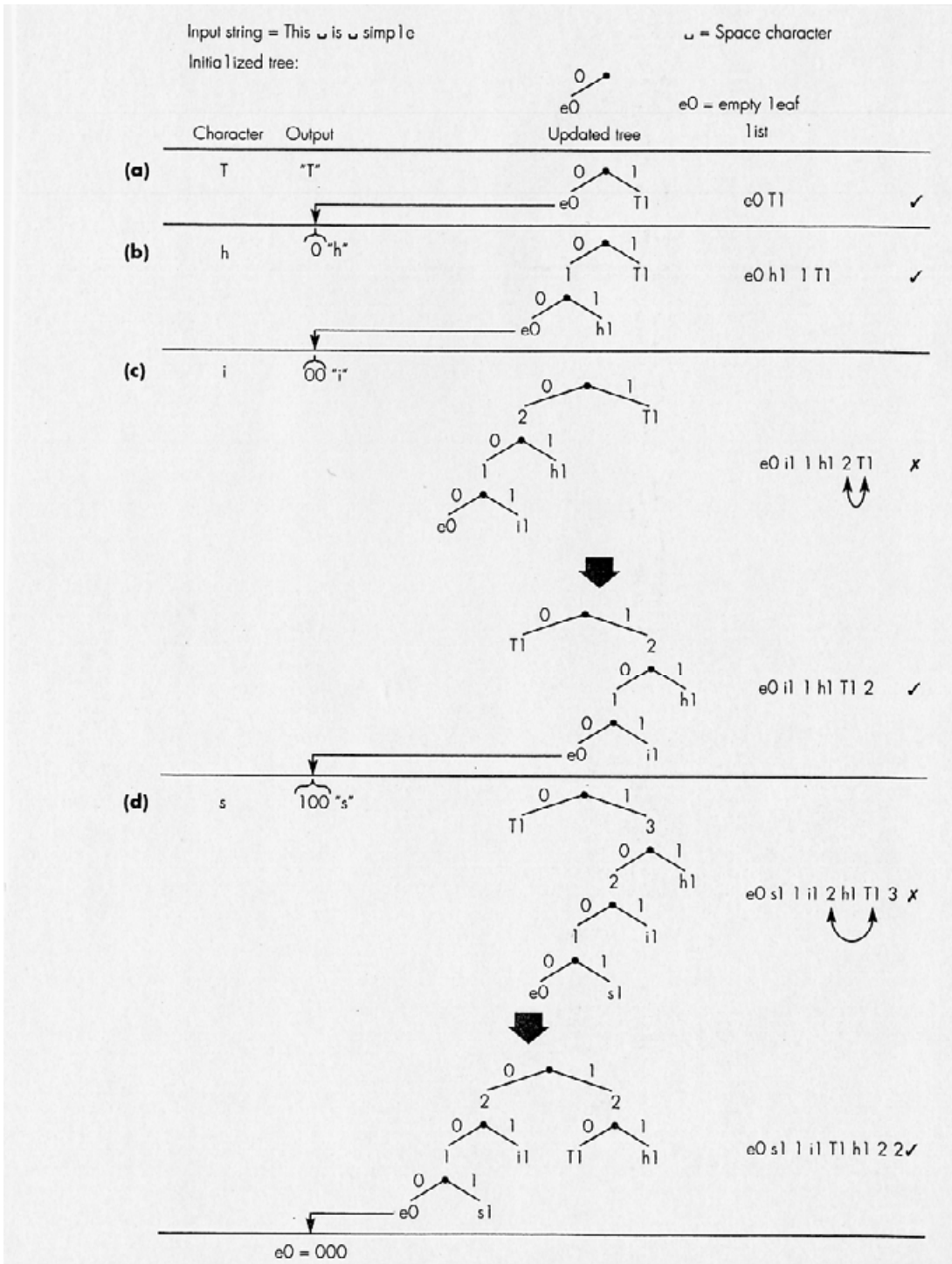


Figure 3.6 Dynamic Huffman encoding algorithm.

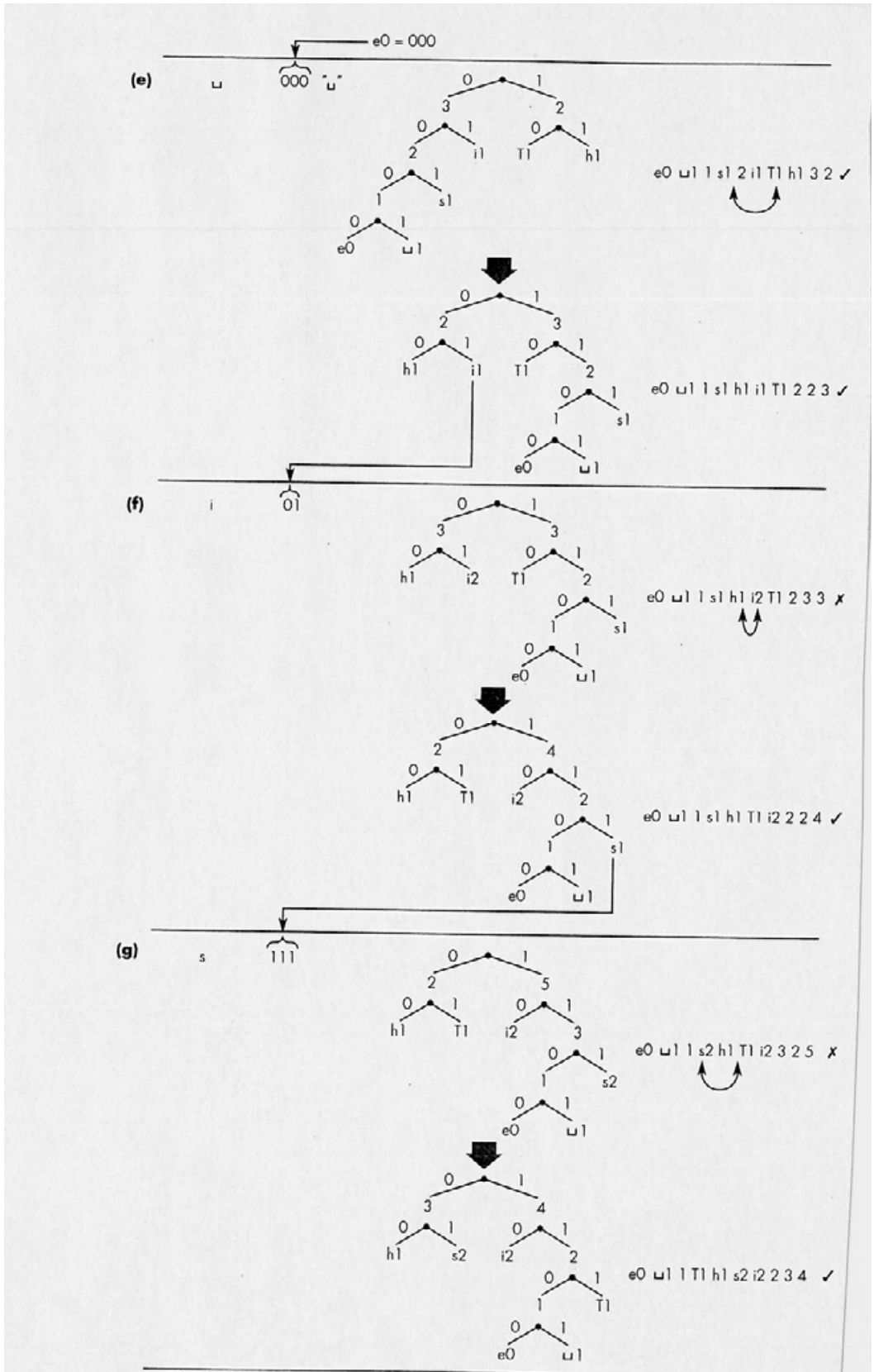


Figure 3.6 Continued

Codage de Lempel-Ziv

- Utilise des chaînes de caractères comme unité de compression plutôt que des caractères simples
- Chaque mot (l'espace est utilisé comme séparateur de chaîne) est contenu dans une table et on ne transmet que l'index du mot dans cette table
- L'encodeur et le décodeur doivent avoir cette table
- C'est un algorithme de compression par **dictionnaire**
- Typiquement 25000 mots, soit un index de 15 bits de long

Codage de Lempel-Ziv-Welsh

- Le contenu du dictionnaire est construit dynamiquement par l’encodeur et le décodeur
- Initialement le dictionnaire de chacun ne contient que le jeu de caractères (e.g. ASCII) utilisé pour créer le texte
- Les mots sont ajoutés jusqu’à atteindre la taille maximale du dictionnaire

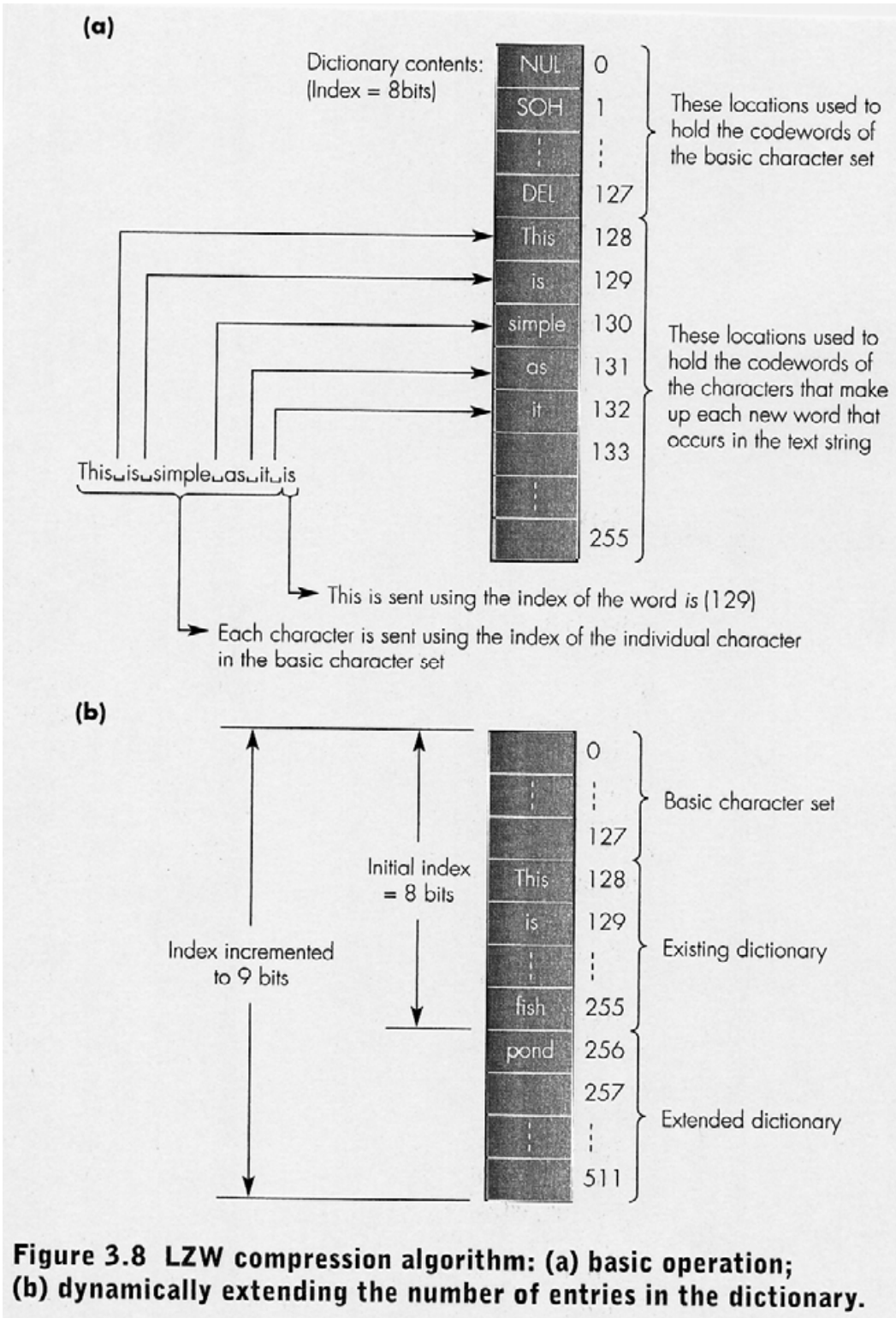


Figure 3.8 LZW compression algorithm: (a) basic operation; (b) dynamically extending the number of entries in the dictionary.

Compression des images

- Deux types d'images :
  - Images générées par ordinateur (graphiques)
  - Images numérisées de documents ou de photos
- Compression sans perte pour le dessin vectoriel (graphiques)
- Compression avec ou sans perte pour les *bitmaps*
  - Sans perte : compression *run-length* + statistique
  - Avec perte : transformée + différentielle + *run-length*

Format GIF

- *Graphics Interchange Format* (GIF)
- Supporte des images couleurs de 24 bits par pixel (8 bits pour R, G et B)
- Réduit le nombre de couleurs utilisées en choisissant 256 couleurs du jeu original de 2^{24} couleurs
- La table de couleur stocke donc 256 couleurs de 24 bits
- L'index sur 8 bits est utilisé pour coder chaque pixel (compression 3:1)
- La table des couleurs peut être globale ou locale à une portion de l'image

Format GIF (suite)

- L’algorithme LZW est utilisé pour ajouter dynamiquement des chaînes de couleurs fréquentes supplémentaires à la table des couleurs (en ajoutant 1 ou plusieurs bits d’indexation)
- L’occurrence de chaînes fréquentes ayant la même couleur et 3 pixels de long est représentée par une entrée dans la partie étendue de la table des couleurs
- GIF autorise le stockage et le transfert d’une image en mode entrelacé : l’image compressée est divisée en 4 groupes de 1/8, 1/8, ¼ et ½ transférable l’un après l’autre

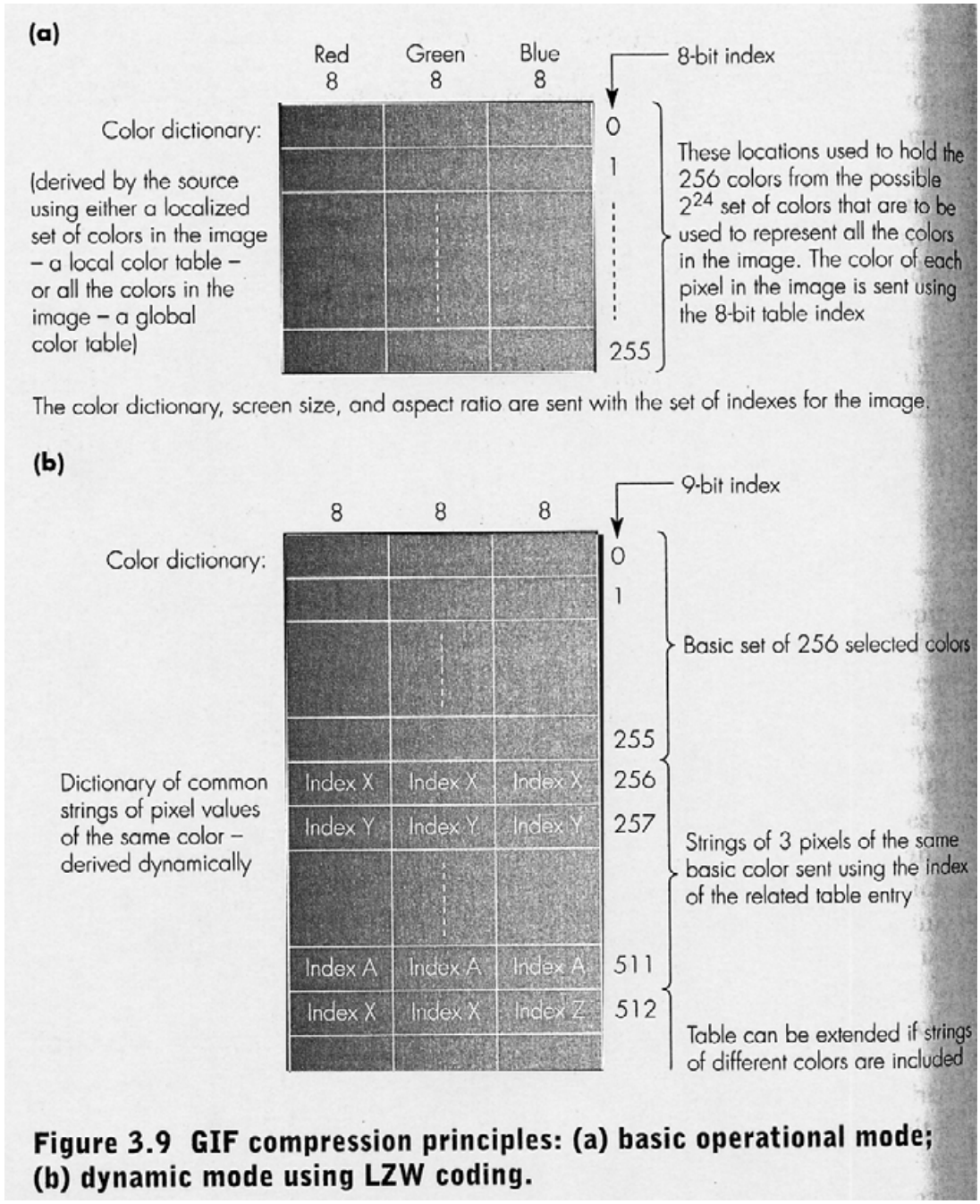


Figure 3.9 GIF compression principles: (a) basic operational mode; (b) dynamic mode using LZW coding.

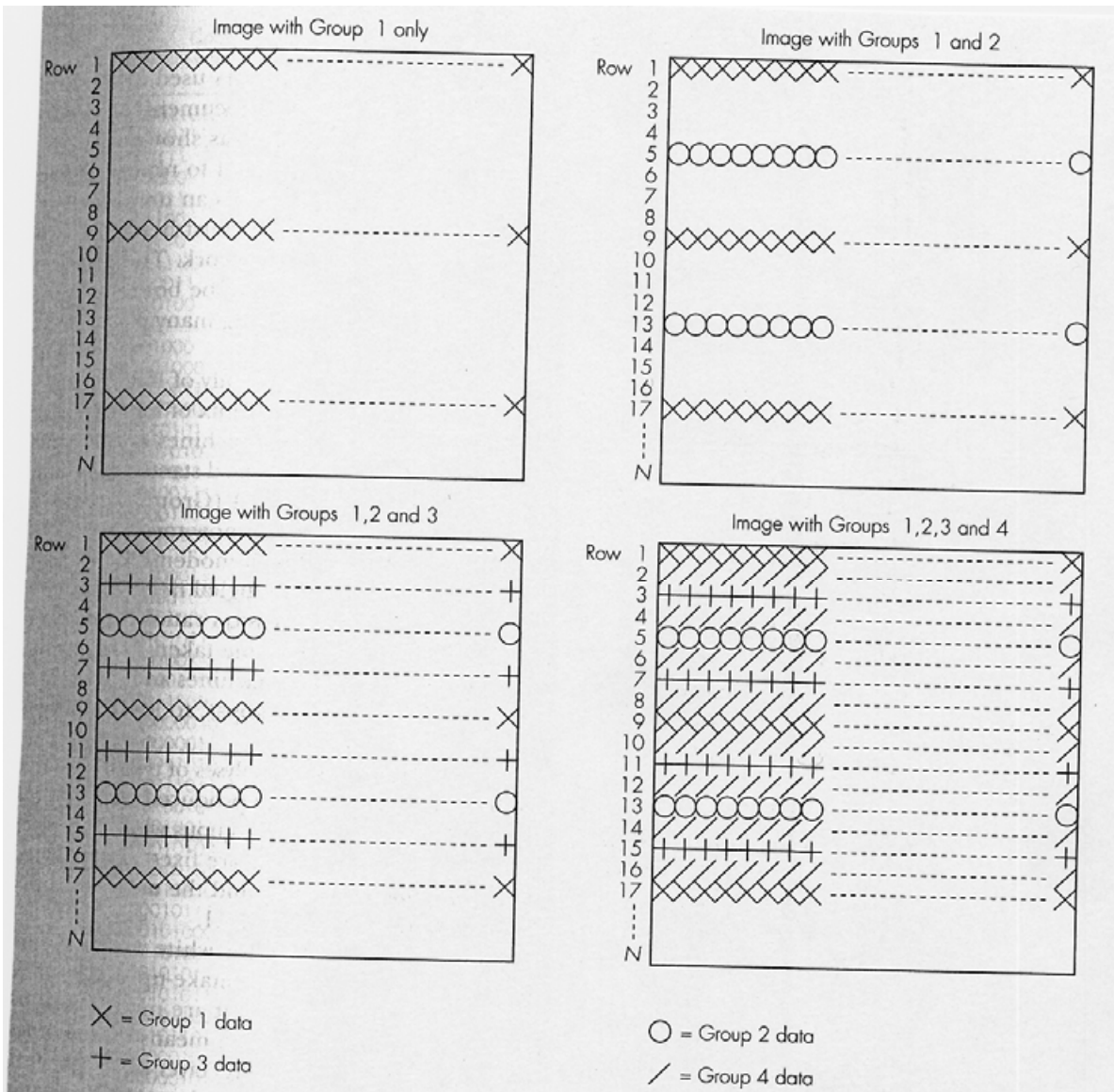


Figure 3.10 GIF interlaced mode.

Format TIFF

- *Tagged Image File Format* (TIFF)
- Supporte une résolution couleur jusqu'à 48 bits (16 bits par RGB)
- L'image peut être stockée sous divers formats :
  - Code 1 : non compressé
  - Code 2 à 4 : compression de type fax pour documents numérisés
  - Code 5 : compression LZW (idem que pour GIF, la table peut atteindre 4096 entrées)

Documents numérisés

- Standards de compression pour les machines facsimilé
- Définis par l'ITU-T (CCITT) : groupes 1 à 4, codages T2 à 4 et T6
- Les machines des groupes 3 et 4 opèrent en numérique :
  - Les machines du groupe 3 utilisent des modems sur le RTC analogique
  - Les machines du groupe 4 utilisent des modems numériques sur des lignes RNIS
- Les documents sont en noir et blanc d'où un codage de 1 bit par pixel

ITU-T groupes 3 et 4, T4

- Définition de tables de mots de code
- Codage en fonction de la longueur d'une chaîne noire ou blanche
- Deux types de codes :
  - Codes de terminaison (longueur de 0 à 63)
  - Codes de make-up : pour gérer les grands nombres (64, 128, 192, etc.)
- Exemple : 5 blancs = 1100
  5 noirs = 0011
  128 blancs = 10010
  128 noirs = 000011001000
- Le codage de 133 blancs se décompose en 128 + 5 =10010+1100

ITU-T groupes 3 et 4, T4 (suite)

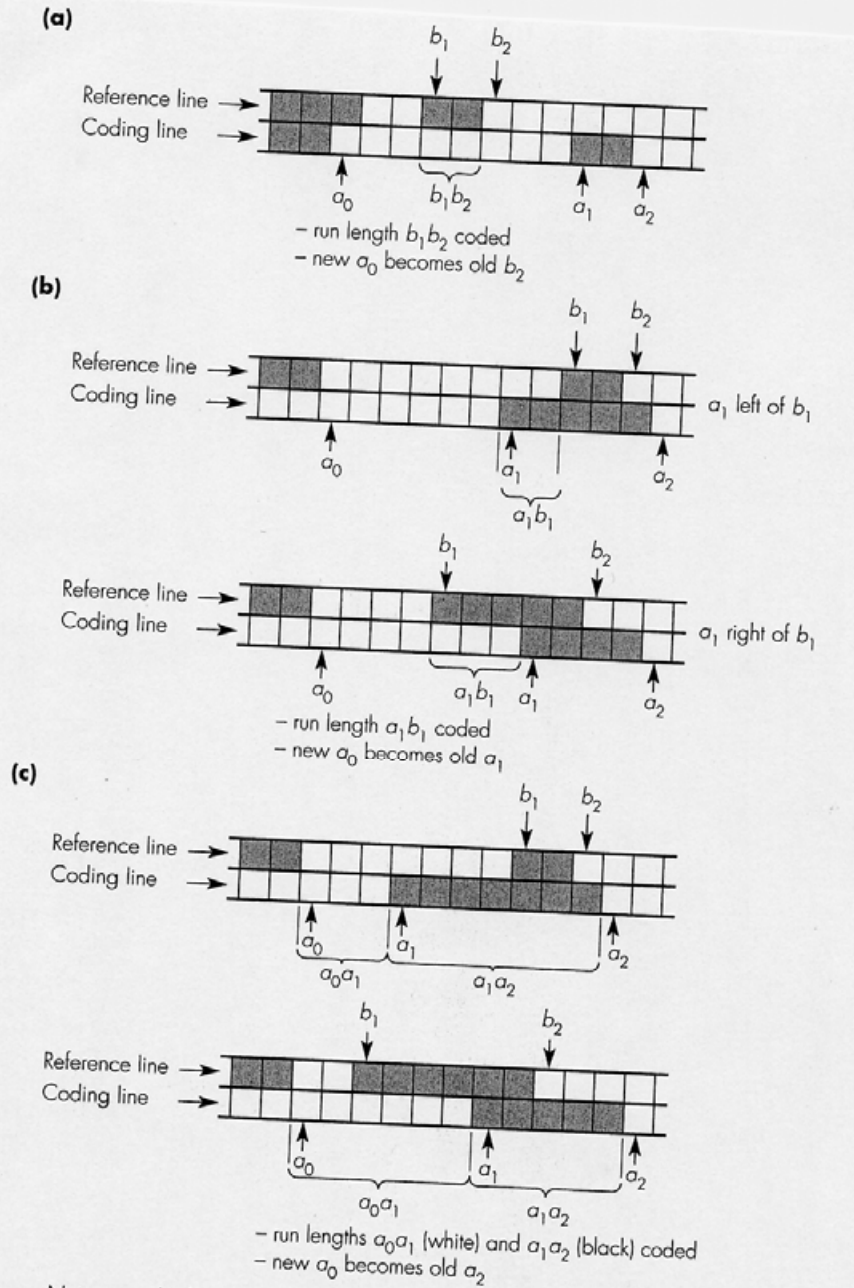
- *Overscanning* : toute ligne commence par un pixel blanc puis il y a alternance noir/blanc
- Deux jeux de codes d'où le nom de codes de **Huffman** modifiés
- Pas de correction d'erreur dans le groupe 3 : pour éviter qu'une erreur ne perturbe tout le décodage, chaque ligne scannée se termine par un code de fin de ligne (*EOL*)
- Un *EOL* précède le codage de chaque page et une chaîne de 6 *EOL* indique la fin de chaque page
- Codage mono dimensionnel
- Fonctionne mal avec les photos

ITU-T groupes 3 et 4, T6

- Optionnel dans le groupe 3, obligatoire dans le groupe 4
- Codage bidimensionnel *modified-modified READ* (MMR) car il identifie les longueurs blanc/noir en comparant les lignes adjacentes
- *READ* signifie *relative element address designate*
- Le codage MMR exploite le fait que la plupart des lignes ne diffèrent des précédentes que par quelques pixels
- Le codage d'une ligne (*coding line*) se fait relativement à la ligne précédente (*reference line*)
- La 1ère ligne est imaginaire blanche

Modes

- *Pass mode* : le *run-length* (bits identiques) b_1b_2 dans la RL est strictement à la gauche du prochain *run-length* a_1a_2 dans la CL. Le *run-length* b_1b_2 est codé par les *modified huffman codes*
- *Vertical mode* : le *run-length* b_1b_2 dans la RL recouvre le prochain *run-length* a_1a_2 dans la CL par un maximum de + ou – 3 pixels. Seul le *run-length* différentiel a_1b_1 est codé
- *Horizontal mode* : le *run-length* b_1b_2 dans la RL recouvre le prochain *run-length* a_1a_2 dans la CL par plus de + ou – 3 pixels. Les 2 *run-length* a_0a_1 et a_1a_2 sont codés.



Note: a_0 is the first pel of a new codeword and can be black or white
a_1 is the first pel to the right of a_0 with a different color
b_1 is the first pel on the reference line to the right of a_0 with a different color
b_2 is the first pel on the reference line to the right of b_1 with a different color

Figure 3.12 Some example run-length possibilities: (a) pass mode; (b) vertical mode; (c) horizontal mode.

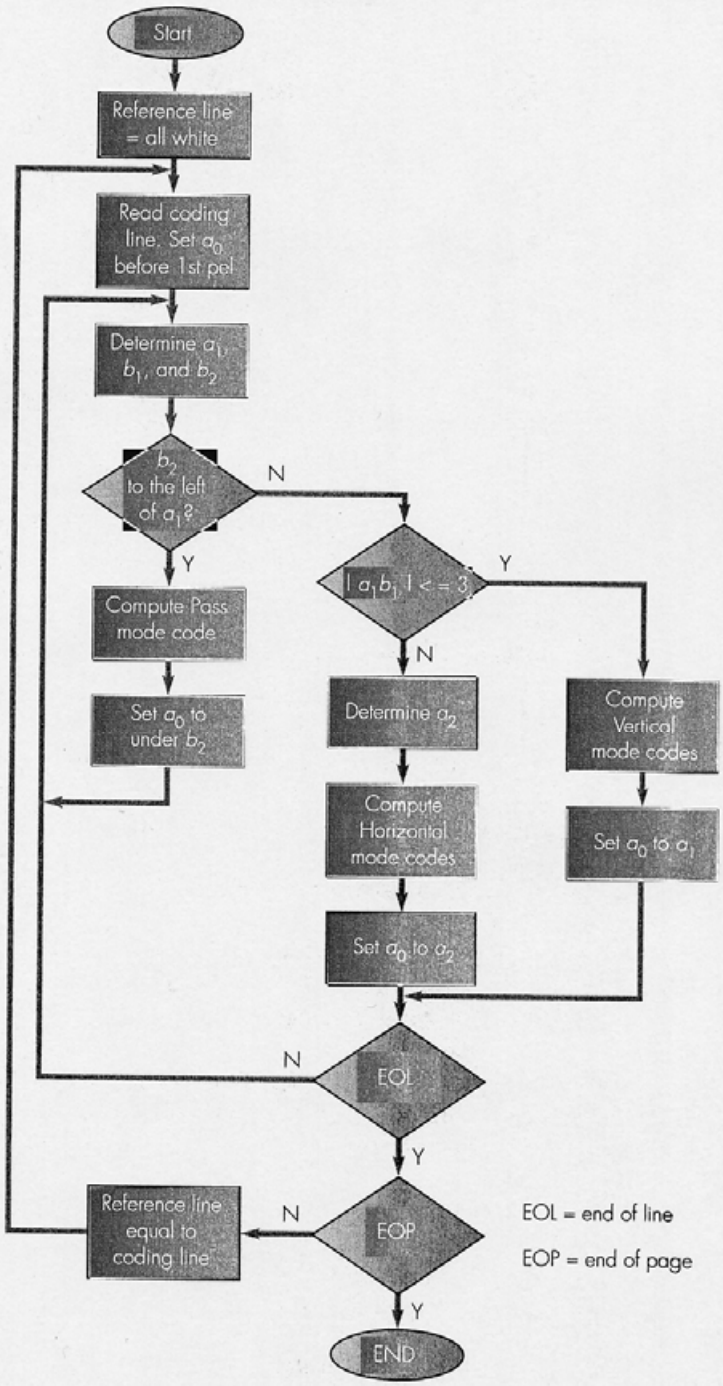


Figure 3.13 Modified-modified READ coding procedure.

Format JPEG

- Joint Photographic Experts Group (JPEG)
- Standard ISO/ITU/IEC IS10918
- Présentation du mode séquentiel avec perte (mode de base) utilisé pour compresser des images numérisées couleurs ou monochromatiques
- 5 étapes associées à ce mode :
  - Préparation des blocs
  - DCT
  - Quantification
  - Codage entropique
  - Construction du cadre

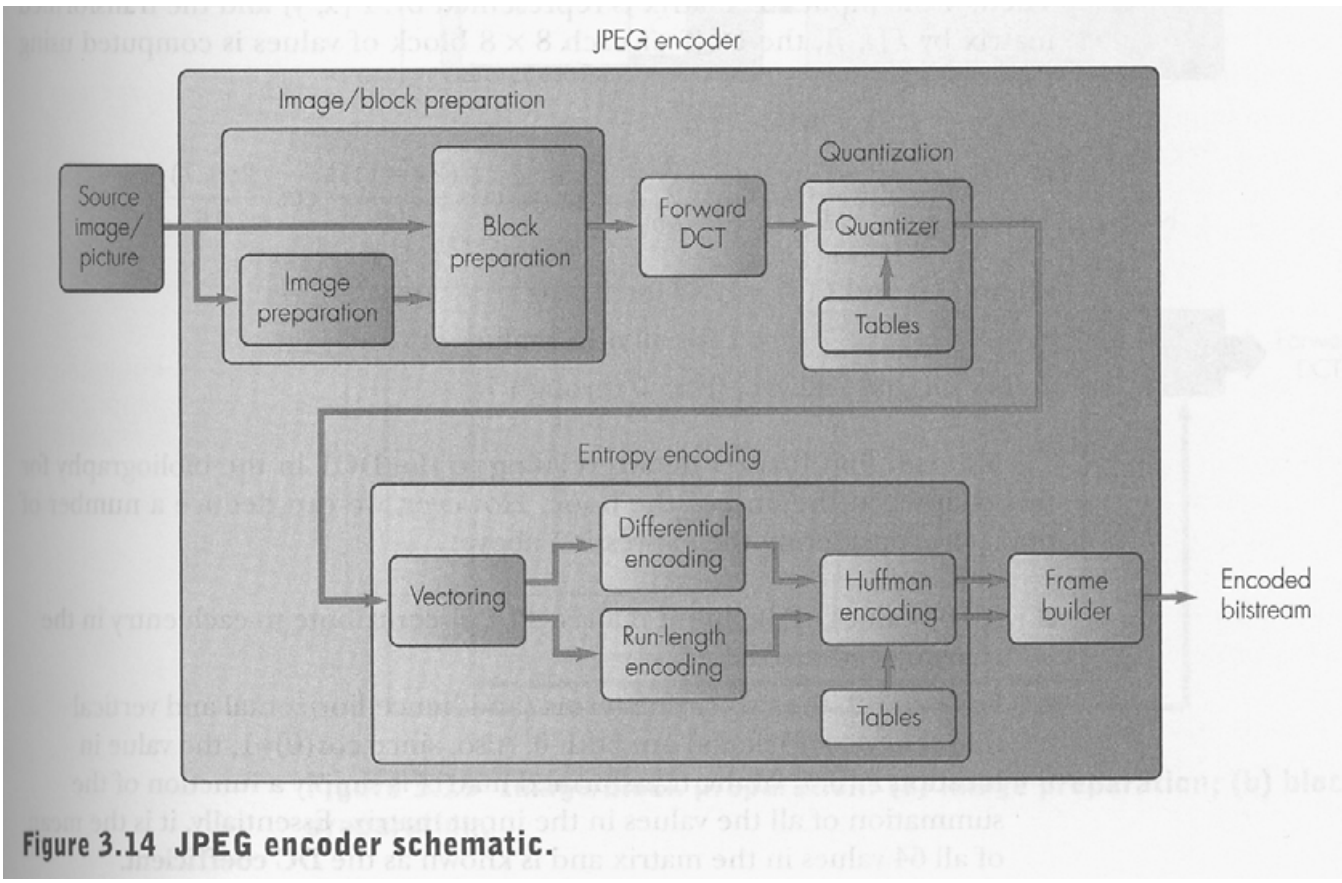


Figure 3.14 JPEG encoder schematic.

Préparation de l’image et des blocs

- Représentation par une ou plusieurs matrices 2D :
  - 1 en niveaux de gris (8 bits par pixel)
  - 1 avec une Color Look Up Table
  - 3 en RGB
  - 1 + 2 de taille ¼ en YC_bC_r
- L’image est découpée en un ensemble de blocs / sous-matrices de 8×8 pixels

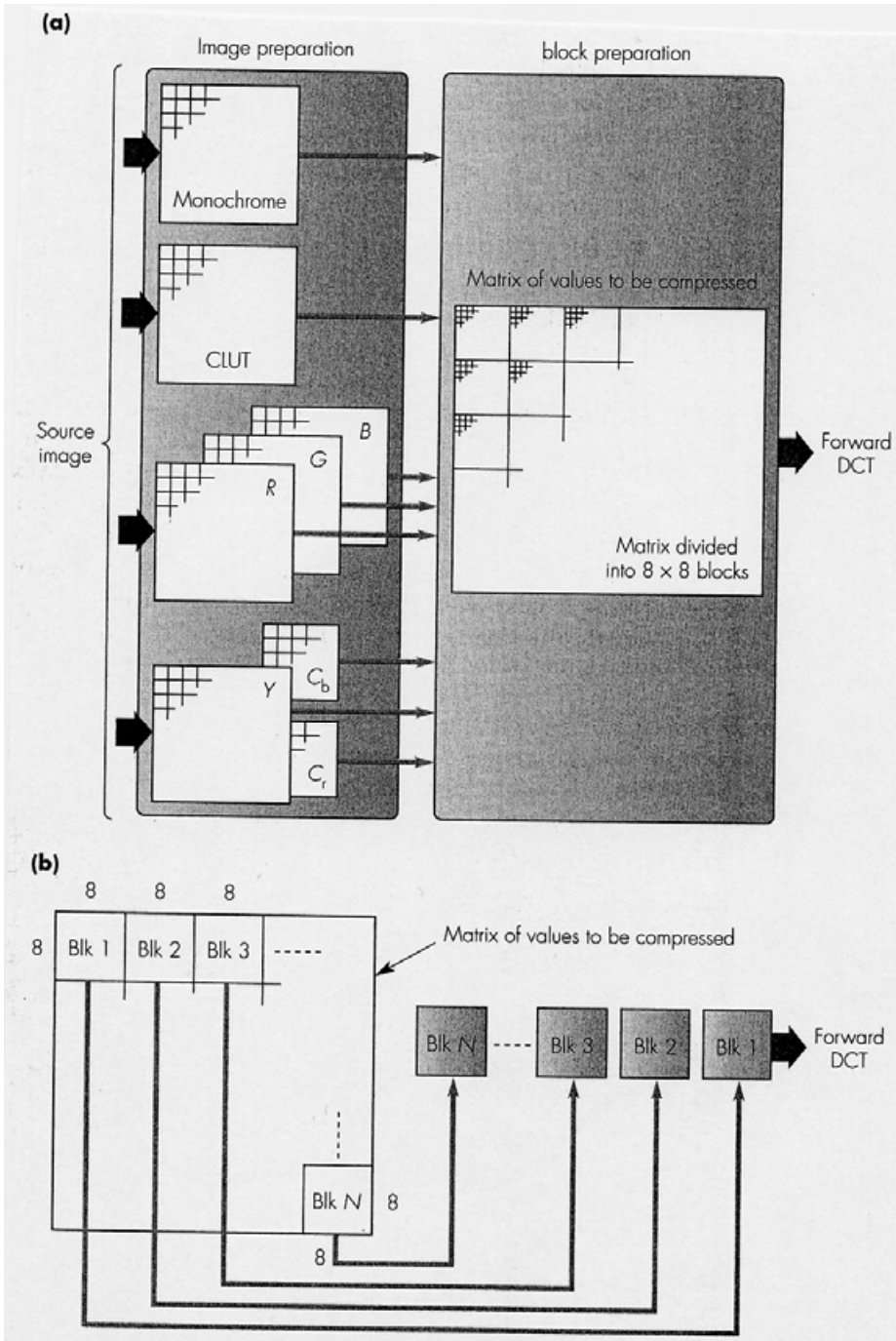


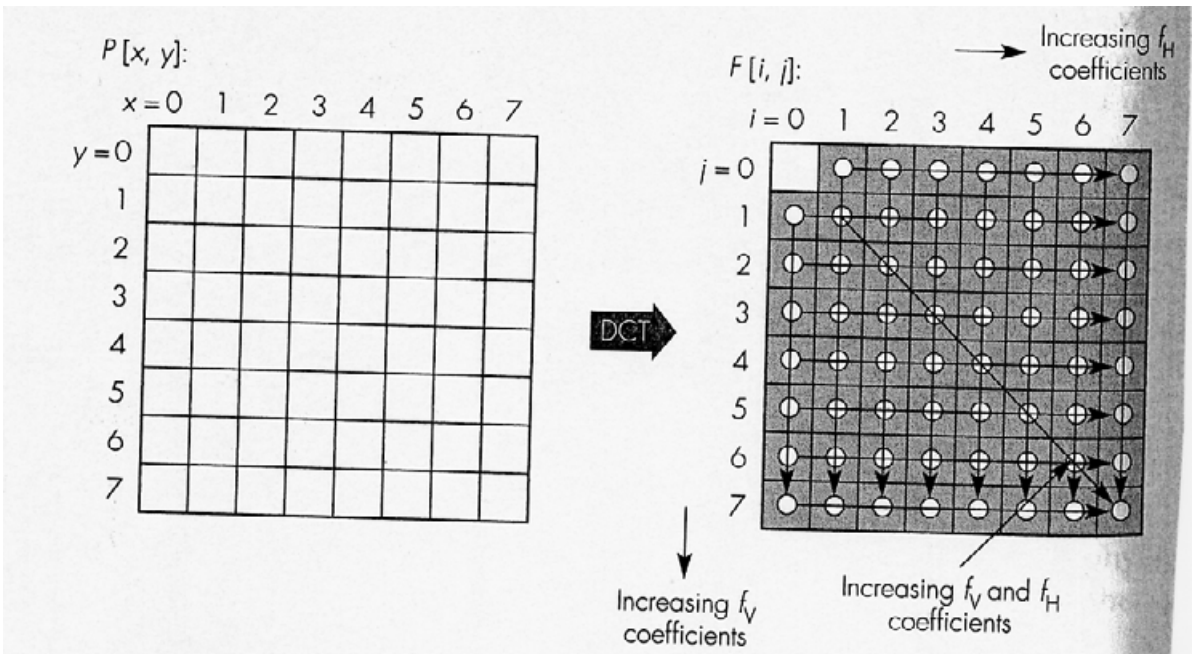
Figure 3.15 Image/block preparation: (a) image preparation; (b) block preparation.

Transformée en cosinus discrète

- Notion de fréquence spatiale : une *Discrete Cosine Transform* (DCT) donne un spectre spatial d'une image, les HF peuvent être éliminées
- Les valeurs de couleur des pixels sont codées sur 8 bits et centrées à 0 (– 128 à +127)
- Si la matrice d'entrée est P[*x,y*] et celle de sortie est F[*i,j*] alors :

$$F[i,j]=\frac{1}{4}C(i)C(j)\sum_{x=0}^{7}\sum_{y=0}^{7}P[x,y]\cos\frac{(2x+1)i\pi}{16}\cos\frac{(2y+1)j\pi}{16}$$

- C(*i*) = C(*j*) = 1/√2 pour *i, j* = 0
  = 1 pour toutes autres valeurs de *i* et *j*



$P[x, y]$ = 8 × 8 matrix of pixel values

$F[i, j]$ = 8 × 8 matrix of transformed values/spatial frequency coefficients

In $F[i, j]$: □ = DC coefficient ▒ = AC coefficients

f_H = horizontal spatial frequency coefficient

f_V = vertical spatial frequency coefficient

Figure 3.16 DCT computation features.

Quantification

- La valeur en 0,0 est égale à la moyenne des 64 valeurs du bloc et est nommé coefficient DC, les autres valeurs sont les coefficients AC

- L’œil est sensible au DC et aux basses fréquences spatiales, si l’amplitude d’un signal haute fréquence est sous un certain seuil, elle va passer inaperçue

- Les coefficients de la DCT sont divisés par les valeurs de la table de quantification

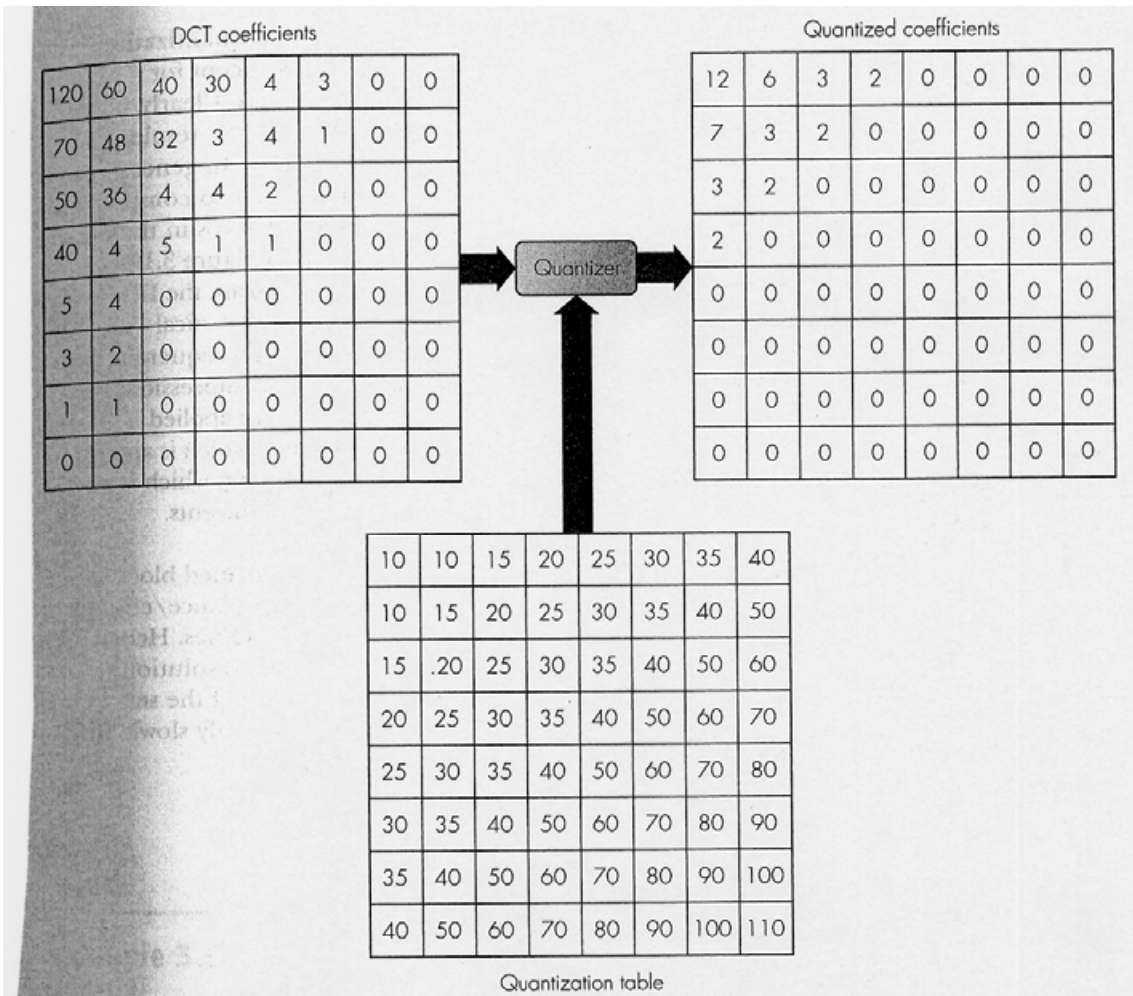


Figure 3.17 Example computation of a set of quantized DCT coefficients.

Codage entropique

- Le codage entropique se compose de 4 étapes :
  - *Vectoring*
  - Codage différentiel
  - Codage *run-length*
  - Codage de Huffman
- *Vectoring* : linéarisation de la matrice en vecteur par un chemin en zig-zag pour créer des chaînes longues
- Codage différentiel : seule la différence d'amplitude entre les DC des blocs est codée sous la forme (**SSS**, **valeur**) où **SSS** indique le nb de bits pour coder la **valeur** (la 1ère valeur est codée par rapport à 0)

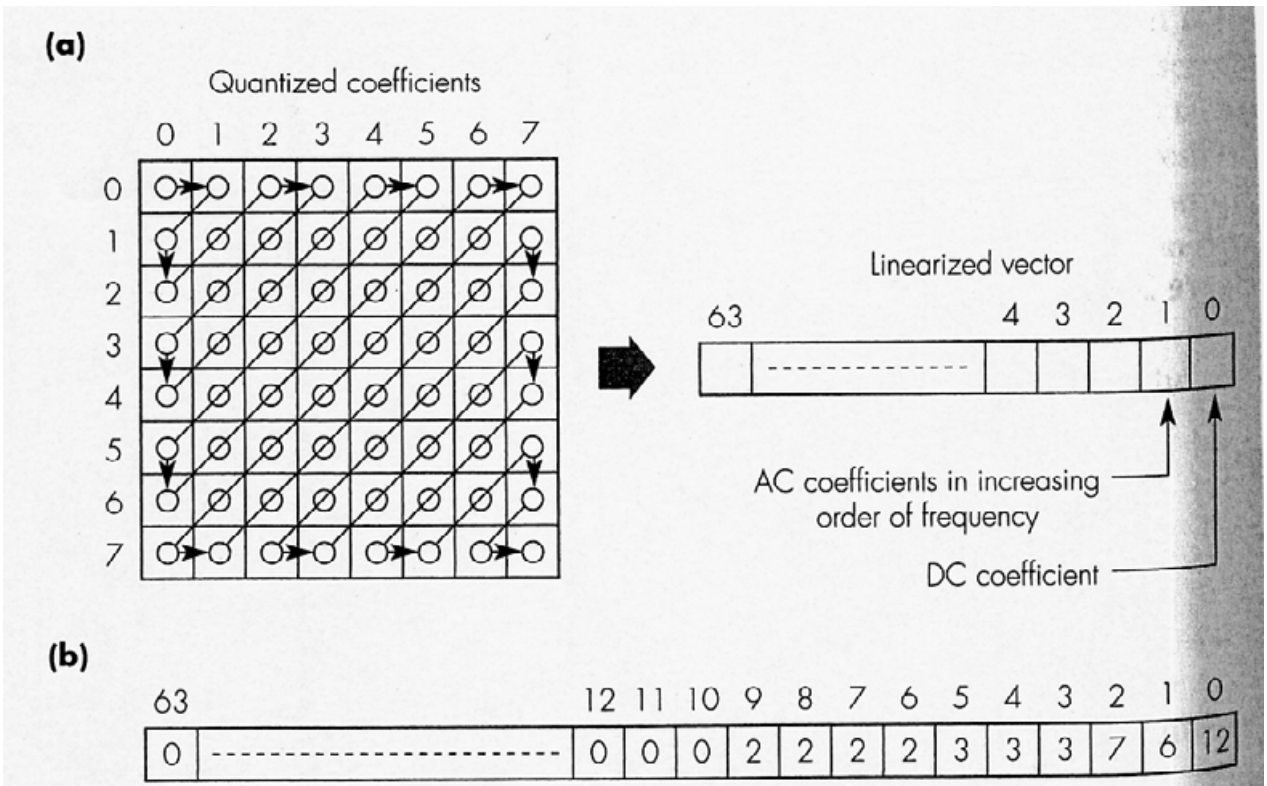


Figure 3.18 Vectoring using a zig-zag scan: (a) principle; (b) vector for example shown in Figure 3.17.

Codage entropique (suite)

- Codage *run-length* : les 63 coefficients AC sont codés sous forme de paires (**saut**, **valeur**) où **saut** est le nombre de 0 dans le *run* et **valeur** le prochain coefficient non nul. La paire (0,0) indique la fin de la chaîne pour ce bloc et que les coefficients restants sont nuls. Le champ **valeur** est codé sous la forme (**SSS**, **valeur**)
- Codage de Huffman : pour les coefficients DC codés en différentiel, les bits du champ **SSS** sont codés avec **Huffman** (les bits du champ **SSS** ont la propriété du préfixe)

Construction du cadre

- Structure de l’image compressée
- Le constructeur de cadre encapsule toute l’information relative à l’image compressée dans le format choisi
- L’entête de cadre contient :
  - La largeur et la hauteur de l’image
  - Le nombre et le type de composantes utilisées pour représenter l’image (CLUT, RGB, YC_bC_r)
  - Le format de numérisation (4:2:2,4:2:0, etc.)
- L’entête de *scan* contient :
  - L’identité de la composante (RGB, etc.)
  - Le nb de bits utilisé pour chaque composante
  - La table de quantification utilisée

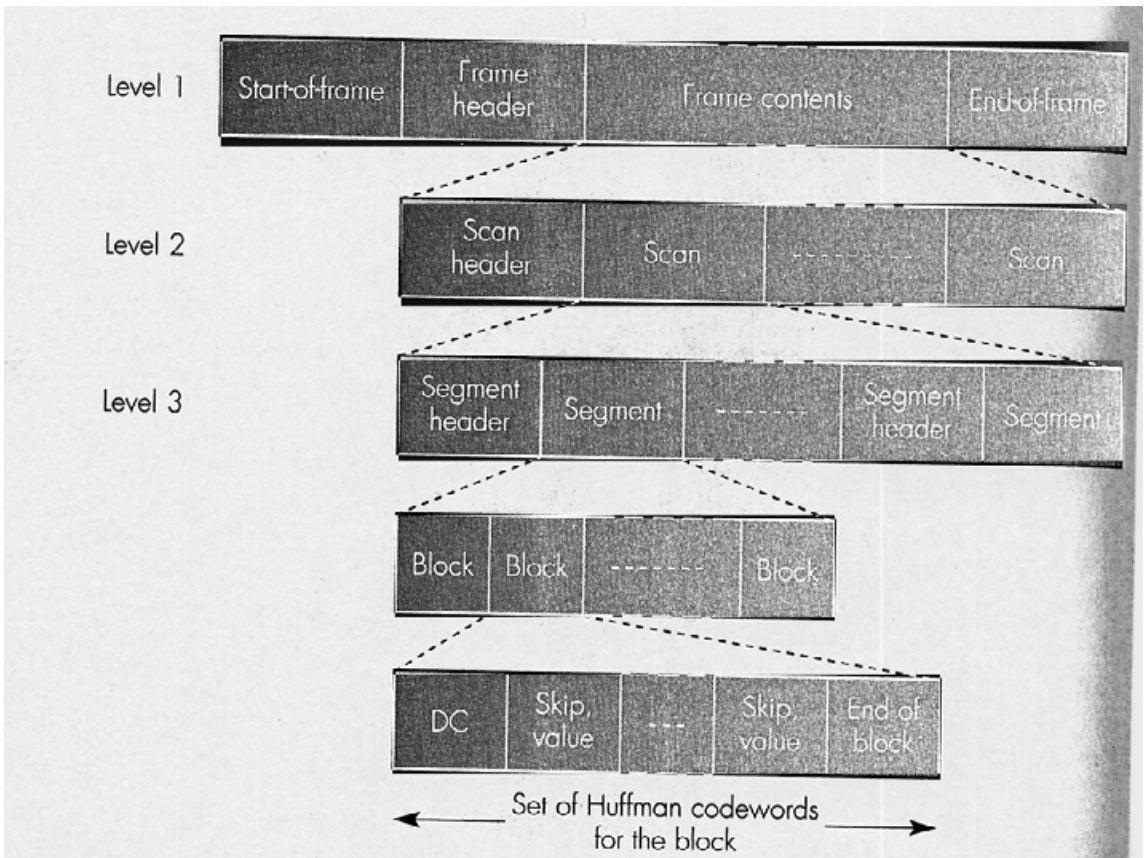


Figure 3.20 JPEG encoder output bitstream format.

Décodeur JPEG

- Le décodeur de cadre identifie les informations de contrôle et les tables
- Le flot de bits est décompressé puis déquantifié (par multiplication)
- La DCT inverse est calculée pour chaque bloc puis l'image est reconstruite
- Possibilité de reconstruire l'image progressivement :
  - Mode progressif : les DC et les coefficients BF de chaque bloc sont envoyés avant les autres
  - Mode hiérarchique : l'image est envoyée en basse résolution (e.g. 320×240) puis en haute résolution (e.g. 640×480)

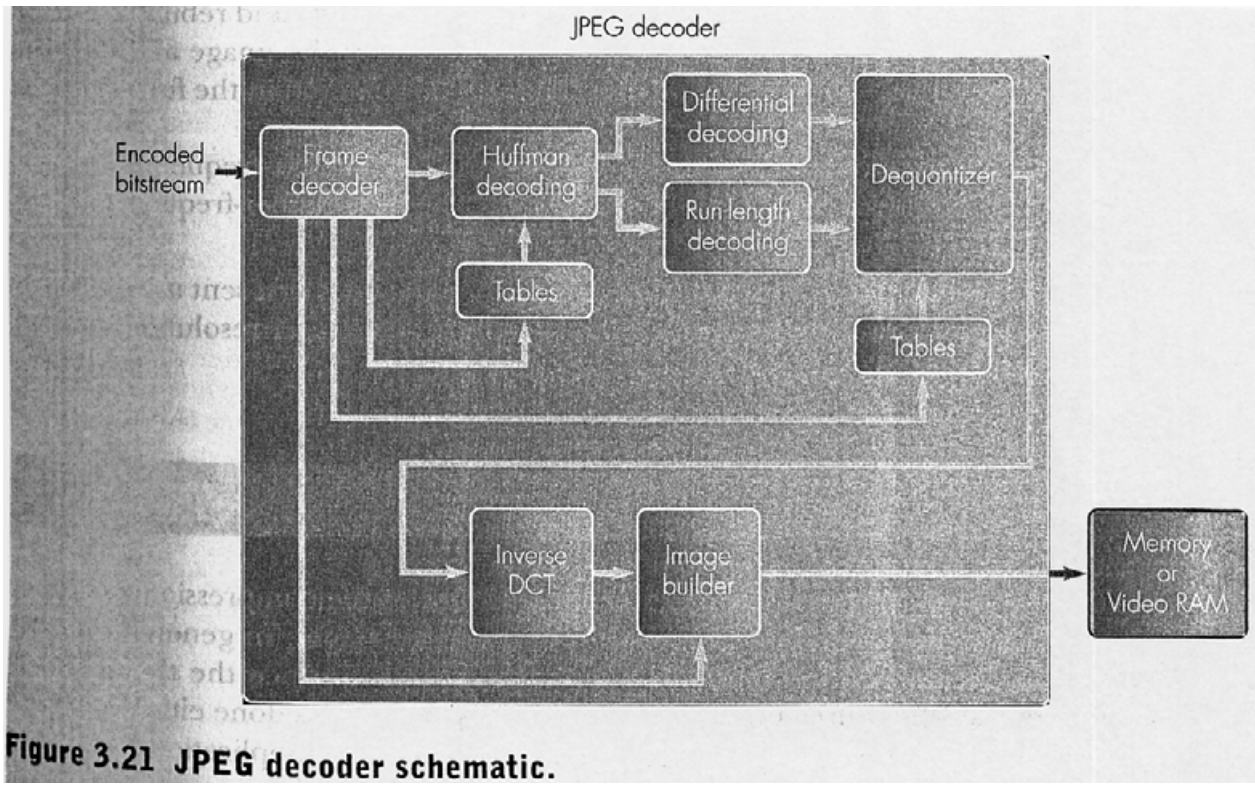


Figure 3.21 JPEG decoder schematic.