Initiation au traitement d'images

Franck Luthon

INITIATION AU TRAITEMENT D'IMAGES

Franck Luthon

Franck.Luthon@univ-pau.fr

Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour EA3000
DPT G.I.M., IUT 2 Allée du Parc de Montaury, 64600 Anglet, France

http ://www.iutbayonne.univ-pau.fr/∼luthon/img0.pdf [1]

Table des matières

Chapitre 1

Traitement d'image

1.1 Introduction

La vision est la **perception** du monde extérieur. L'image est la **représentation** bidimensionnelle d'une scène 3-D. C'est l'information issue d'un capteur de vision (caméra, œil). Le traitement d'image comporte différentes tâches : capture, analyse et traitement (bas niveau), interprétation et décision (haut niveau). On distingue différents traitements :

- amélioration, restauration et correction d'image : augmentation du contraste, correction des distorsions optiques, filtrage du bruit
- analyse [2] : détection et localisation d'objets, segmentation, reconnaissance de formes, estimation et mesures
- codage pour la compression, l'encryptage et la transmission numérique.

Le traitement d'image couvre plusieurs domaines [3] :

- l'**électronique** (capteur, optique, acquisition)
- le **traitement de signal** (détection, segmentation, estimation, décision)
- l'**informatique** et l'intelligence artificielle (reconnaissance, interprétation, guidage robotique)

On ne s'intéresse ici qu'aux **images numériques**, c'est-à-dire échantillonnées et quantifiées. Une image numérique est définie par un tableau de pixels de taille $L \times C$ (typ. 256×256). Chaque pixel, portant l'information d'intensité lumineuse perçue en ce point, est codé sur n bits. Typ. $n = 8$ pour une image à 256 niveaux de gris (NdG) et $n = 24$ pour une image couleur RVB (8 bits par composante couleur).

On définit dans l'image un repère $0xy$ et une notion de distance et de voisinage (Fig. 1.1).

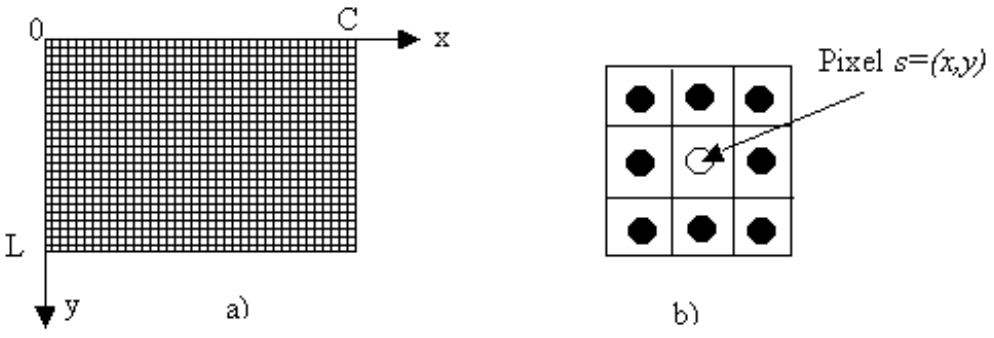

FIGURE 1.1 – a) Repère relatif à l'image ; b) 8-voisinage d'un pixel.

1.2 Rappels de théorie de l'information

1.2.1 Entropie d'une source discrète

L'entropie H (exprimée en bits) d'une source discrète d'observations à valeurs dans $\{0...M-1\}$ (typ. $M = 256$) est définie par [4] :

$$H_{bit} = -\sum_{i=0}^{M-1} p_i \log_2 p_i \tag{1.1}$$

où p_i représente la probabilité pour que l'observation en le site $s = (x, y) \in S$ vaille : $o_s = i$ (S étant le support des observations, en l'occurrence ici une image de taille $L \times C$).

Rappelons que le choix de la base du logarithme est arbitraire et correspond au choix de l'unité de mesure (bit si log binaire, nat si log népérien) ; et l'on a : $H_{nat} = H_{bit} \cdot \log_e 2 \approx 0.7 \cdot H_{bit}$. En effet, le passage d'une base a à une base b requiert simplement une multiplication par la constante $K = \log_b a$.

Comme l'information élémentaire apportée par un événement de probabilité p_i vaut par définition : $I_i = \log \frac{1}{p_i}$, on voit que l'entropie est une mesure de l'**information moyenne** débitée par une source : $H = \sum p_i I_i$.

Shannon a démontré que l'entropie (exprimée en nats) d'une source gaussienne centrée d'écart-type σ vaut $H_{nat} = \log(\sigma \cdot \sqrt{2\pi e})$ et qu'à σ fixé, la source donnant l'entropie maximale est la distribution gaussienne.

Pour une source quelconque d'entropie H donnée, Shannon définit la notion de "puissance entropique" N :

$$N = \frac{1}{2\pi e} \exp(2H) \tag{1.2}$$

C'est la puissance de la distribution gaussienne équivalente à la distribution d'entropie H. Pour un bruit gaussien centré d'écart-type σ, on retrouve bien sûr : $N = \sigma^2$. Comme le bruit blanc gaussien possède l'entropie maximale à puissance fixée, la puissance entropique d'un bruit quelconque est toujours inférieure ou égale à sa puissance effective. Toujours selon Shannon, un bruit blanc gaussien a la propriété d'absorber tout autre signal qui lui est ajouté. La puissance entropique résultante est à peu près égale à la somme de la puissance du bruit blanc et de la puissance du signal (supposé centré), à condition que la puissance du signal soit faible, "dans un certain sens", comparée au bruit [4].

En supposant le bruit additif gaussien et majoritaire sur le support, on peut donc estimer l'écart-type entropique σ_e équivalent en calculant l'entropie H des observations [5] :

$$\sigma_e = \sqrt{N} = \frac{\exp(H)}{\sqrt{2\pi e}} = \frac{2^{H_{bit}}}{\sqrt{2\pi e}} \tag{1.3}$$

En prenant une mesure de seuil θ_e estimée par :

$$\theta_e = 4\sigma_e \approx 2^{H_{bit}} \tag{1.4}$$

on obtient une technique adaptative automatique de détection du mouvement dans les séquences d'images [6].

L'entropie est une notion fondamentale car elle renseigne sur la redondance spatiale présente dans une image et permet de mettre en œuvre les outils de codage définis par Shannon pour réaliser une compression sans perte de l'information contenue dans l'image (économie de débit pour la transmission : codage de Huffman, de Shannon-Fano). La redondance d'une source X est définie par :

$$R = 1 - \frac{H(X)}{H_{max}} \tag{1.5}$$

Pour une source discrète à M événements, l'entropie maximale est atteinte quand les événements sont équiprobables et elle vaut : $H_{max} = \log_2 M$.

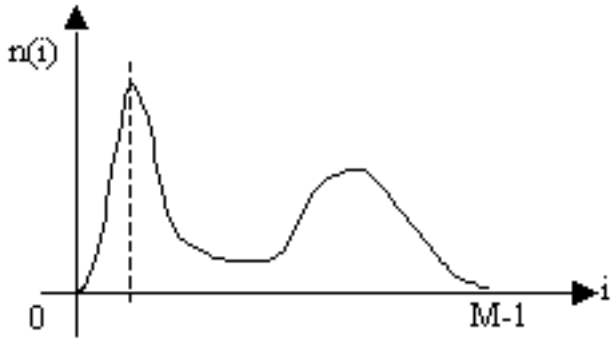

FIGURE 1.2 – Exemple d'histogramme comportant deux modes.

1.2.2 Histogramme

L'histogramme est la courbe $n(i)$ où n est le nombre de pixels d'intensité i. Il peut comporter plusieurs pics ou modes, ou bien être uniforme (Fig. 1.2).

Si l'on note $N = L \times C$ le nombre de pixels de l'image, et M le nombre de niveaux de gris possibles ($i = 0 \cdots M-1$), la probablité d'un niveau de gris vaut $p_i = \frac{n_i}{N}$ et la fonction de densité de probabilité (PDF) s'exprime : $F(i) = \sum_{k=0}^{i} p_k$ avec $F(0) = p_0$ et $F(M-1) = 1$.

Les deux principales corrections d'histogramme sont l'égalisation et l'étalement. L'**égalisation** consiste à redistribuer les niveaux de gris en remplaçant le niveau i par le niveau l_i tel que :

$$l_i = (M-1)F(i) = \frac{M-1}{L \times C} \sum_{k=0}^{i} n_k \tag{1.6}$$

L'**étalement** consiste à utiliser toute la dynamique des NdG :

$$l_i = (M-1)\frac{i - i_{min}}{i_{max} - i_{min}} \tag{1.7}$$

L'intérêt de ces techniques est d'être complètement automatiques, donc facilement programmables.

Dans le cas général, une correction d'histogramme correspond à appliquer une transformation non linéaire T sur les niveaux de gris :

$$l_i = T(i) \tag{1.8}$$

T doit être une fonction monotone croissante et bornée dans l'intervalle $[0, M-1]$.

Cela permet amélioration ou **trucage** en appliquant un facteur d'amplification non linéaire sur l'échelle de gris (Fig. 1.3).

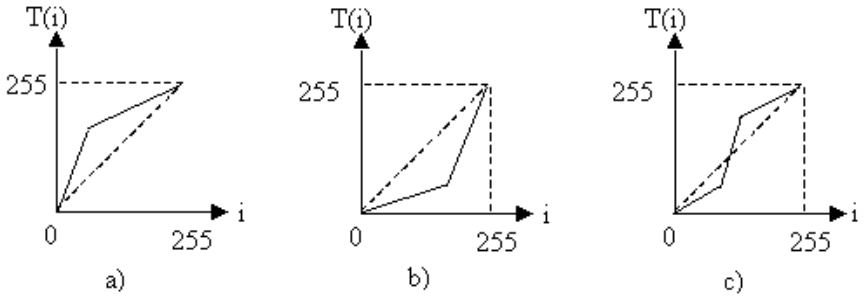

FIGURE 1.3 – Exemple de correction d'histogramme par rehaussement : a) du sombre b) du clair c) central.

1.3 Seuillage

Le seuillage (sur les NdG, sur leurs dérivées, sur l'histogramme, sur l'entropie, sur les transformées ...) est une opération très fréquente (pour ne pas dire systématique) en traitement d'image (nombreuses applications en détection de contours, détection de mouvement ...) Il permet de passer d'une image en NdG à une image N&B (étiquetage binaire $e(s)$) :

$$\forall s=(x,y)\in S,\quad e(s)=\begin{cases}\text{“1”} & \text{si} \quad I(s)>\theta \\ \text{“0”} & \text{sinon}\end{cases} \tag{1.9}$$

La difficulté est évidemment le choix du seuil adéquat θ. Il existe de très nombreuses méthodes et techniques adaptatives, semi-automatiques, ... de sélection du seuil [6]. L'intérêt est de pouvoir ensuite appliquer des outils spécifiques aux images binaires (morphologie mathématique binaire, suivi de contours binaires).

1.4 Contraste

Il existe différentes définitions du contraste dans une image.

1.4.1 Contraste de Michelson

- c'est à l'origine une mesure de visibilité de franges d'interférence
- il considère le cas d'une luminance sinusoïdale
- l'estimateur est : $C_M=\frac{L_{Max}-L_{min}}{L_{Max}+L_{min}}$
- il est utilisé avec succès en psychophysique.
- variantes : $C_M=\frac{L_{Max}-L}{L_{Max}+L}$ et $C_M=\frac{L-L_{min}}{L+L_{min}}$

1.4.2 Contraste de Weber

- il considère le cas d'un fond uniforme
- l'estimateur est : $C_W=\frac{\Delta L}{L}=\frac{L_0-L_{moy}}{L_{moy}}$
- il est utilisé en psychophysique (CIE, météo)

NB : La courbe expérimentale de seuil de visibilité de Weber est largement utilisée.

1.4.3 Contraste de Gordon

- ref Applied Optics 23(4),540 :564,1984
- il considère le cas d'un stimulus complexe (image naturelle)
- c'est un estimateur local basé sur le calcul du NdG moyen de deux régions : $C_G=\frac{L_{moy_1}-L_{moy_2}}{L_{moy_1}+L_{moy_2}}$
- c'est un estimateur signé.

1.4.4 Contraste de Beghdadi

- ref CVGIP 1989, V. 46, pp.162-174
- c'est une mesure entre les NdG moyens des contours estimés sur 2 régions
- $C_B=\frac{C_{moy_1}-C_{moy_2}}{C_{moy_1}+C_{moy_2}}$

1.4.5 Contraste de Peli

- ref J. Optical Soc. America, JOSA 7(10),2032 :2040,1990
- c'est un estimateur local à bande limitée (gamme de fréquences spatiales ω_x,ω_y). Il est basé sur les propriétés du HSV (*Human Visual System)*
- inconvénient : il a un coût de calcul élévé

1.4.6 Contraste de Köhler

- ref bib : GMIP 15,319 :338, 1981
- soit un seuil s tel que : $\min(f(x), f(x_1)) \leq s \leq \max(f(x), f(x_1))$
- c'est un estimateur ponctuel du contraste : $C_{xx_1}(s) = \min(|s - f(x)|, |s - f(x_1)|)$
  ou $C_{xx_1}(s) = \max(|s - f(x)|, |s - f(x_1)|)$
- variantes possibles :
  $C_{xx_1}(s) = \min\left(\frac{|s-f(x)|}{s+f(x)}, \frac{|s-f(x_1)|}{s+f(x_1)}\right)$
  ou
  $C_{xx_1}(s) = \min\left(\frac{|s-f(x)|}{\max(s,f(x))}, \frac{|s-f(x_1)|}{\max(s,f(x_1))}\right)$

1.5 Filtrage linéaire

Le filtrage linéaire est une opération classique pour le pré-traitement des images avec deux applications principales : la réduction de bruit (filtrage passe-bas) et le rehaussement pour la détection de contours (filtrage passe-bande ou passe-haut). Les outils mathématiques utilisés sont la transformée de Fourier, la convolution et la transformée en Z.

1.5.1 Transformée de Fourier Discrète

Cet outil permet de passer du domaine spatial au domaine fréquentiel. Le filtrage fréquentiel s'obtient en multipliant la TF de l'image par la fonction de transfert du filtre (gabarit). L'inconvénient principal est le coût de calcul élévé, bien qu'il existe un algorithme rapide appelé FFT. On réserve donc le filtrage fréquentiel aux cas où la précision est le facteur important.

La transformée de Fourier discrète (TFD) 2-D d'une image s'exprime par :

$$F(u,v) = \frac{1}{\sqrt{LC}} \sum_{x=0}^{C-1} \sum_{y=0}^{L-1} I(x,y) e^{-i2\pi(\frac{u.x}{C} + \frac{v.y}{L})} \tag{1.10}$$

On définit de même la transformée de Fourier inverse, qui permet de remonter à l'image originale $I(x,y)$ connaissant sa transformée $F(u,v)$:

$$I(x,y) = \frac{1}{\sqrt{LC}} \sum_{u=0}^{C-1} \sum_{v=0}^{L-1} F(u,v) e^{+i2\pi(\frac{u.x}{C} + \frac{v.y}{L})} \tag{1.11}$$

On appelle spectre d'amplitude (resp. phase) le module $|F(u,v)|$ (resp. l'argument $\Phi(u,v)$) de la transformée de Fourier, qui est une grandeur complexe. Si la dynamique du spectre est importante, on travaille sur le module en dB : $o_s = 20\log_{10}|F(u,v)|$. Les valeurs de module étant des réels, on peut se ramener au cas d'une source discrète d'observations en quantifiant les modules pour obtenir un ensemble de M valeurs discrètes (*e.g.*, quantum de 0.5dB pour $M = 256$).

L'interprétation physique importante de la TF repose sur l'examen du lieu de phase nulle. Dans l'espace normalisé $\left(X = \frac{x}{C}, Y = \frac{y}{L}\right)$, ce lieu correspondant à : $u.X + v.Y = k \Leftrightarrow Y = -\frac{u}{v}X + \frac{k}{v}$. C'est une série de droites parallèles distantes de $d = \frac{1}{\sqrt{u^2+v^2}}$ et de direction perpendiculaire à la droite de pente $\tan\theta = \frac{v}{u}$. Cette relation montre que les hautes fréquences correspondent à des lignes de phase nulle rapprochées. En terme d'image, un contour se traduit en fréquence par une ligne située à distance d de l'origine. Plus le contour est net, plus d est grande (présence d'énergie en hautes fréquences). Plus le contour est flou, plus d est petite (présence d'énergie en basses fréquences). Donc les hautes fréquences traduisent des contours nets, et les basses fréquences traduisent des contours flous ou des régions uniformes. L'orientation des fréquences spatiales renseigne sur l'orientation des contours dans l'image. L'utilisation de la TF permet donc de pratiquer des filtrages fréquentiels sur l'image.

La Fig. 1.4 montre l'allure typique du spectre d'une image réelle. Notons que l'on trouve pour cette image une entropie $H = 6.5\text{bits} < 8\text{bits}$, ce qui est cohérent pour des observations codées sur 8 bits.

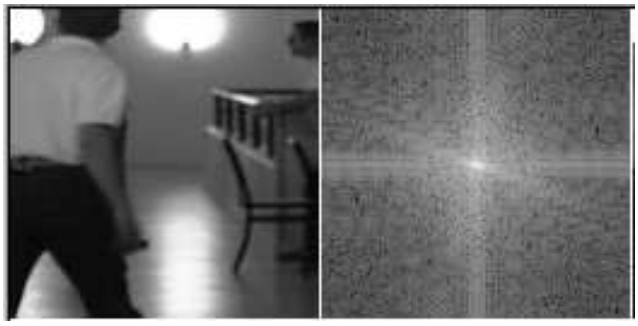

FIGURE 1.4 – Spectre d'une image réelle (**Couloir**).

Repliement de spectre

On sait que l'échantillonnage se traduit par une périodisation fréquentielle. Pour éviter le phénomène de repliement de spectre, le théorème de Shannon stipule que l'on doit respecter la condition suivante entre fréquence d'échantillonnage et fréquence maximale du signal : $F_e \geq 2F_{max}$.

Un filtrage passe-bas permet de respecter cette contrainte : ce filtrage est souvent inhérent au système de capture pour des images réelles. Par contre, on devra se méfier particulièrement de ce problème quand on génère des images synthétiques pour tester des algorithmes de traitement.

Passage de la TF à la TFD

Partant de la TF continue :

$$TF: \quad F(\nu) = \int_{-\infty}^{+\infty} f(x) e^{-i2\pi\nu x} dx \tag{1.12}$$

$$TF^{-1}: \quad f(x) = \int_{-\infty}^{+\infty} F(\nu) e^{i2\pi\nu x} d\nu \tag{1.13}$$

la discrétisation temporelle (cas des signaux fonction du temps) ou spatiale (cas des images fonction de l'espace) donne : $x = kT_e = \frac{k}{F_e}$ (échantillonnage spatial à la période T_e).

Sans perte de généralité, on pose souvent $T_e = 1$ ou bien on utilise la notation de **fréquence réduite** $u = \frac{\nu}{F_e}$, ce qui implique $u_{max} \leq \frac{1}{2}$ car d'après Shannon : $F_e \geq 2\nu_{max}$.

On obtient ainsi la TF discrète dans le temps (TFDT) calculée sur une durée d'observation évidemment limitée : $T = NT_e$.

$$TFDT: \quad F(u) = \sum_{k=0}^{N-1} f(k) e^{-i2\pi uk} \tag{1.14}$$

Si l'on y ajoute l'échantillonnage fréquentiel avec un pas $\Delta_e = \frac{F_e}{N}$ où N est le nombre d'échantillons, alors $\nu = n\Delta_e \Leftrightarrow u = \frac{n}{N}$ et l'on obtient la TFD :

$$TFD: \quad F(n) = \sum f(k) e^{-i2\pi\frac{nk}{N}} \tag{1.15}$$

$$TFD^{-1}: \quad f(k) = \sum F(n) e^{i2\pi n\Delta_e k} = \sum F(n) e^{i2\pi\frac{nk}{N}} \tag{1.16}$$

Si dans la dernière équation on remplace k par x et Δ_e par f_0 où f_0 est le fondamental du signal (ou bien si l'on considère une périodisation arbitraire sur la durée d'observation T), on retrouve le lien avec le DSF classique :

$$f(x) = \sum c_n \exp(i\frac{2\pi nx}{T}) \quad \text{avec } c_n = F(n) = \frac{1}{T}\int_{[T]} f(x)\exp(-i\frac{2\pi nx}{T})dx \tag{1.17}$$

1.5.2 Convolution spatiale

Dans le cas 1-D continu, l'opération (commutative) de convolution s'exprime :

$$f(x) * h(x) = \int_{-\infty}^{+\infty} f(\chi) h(x-\chi) d\chi \tag{1.18}$$

Le filtrage peut être réalisé directement dans le domaine spatial en convoluant l'image $I(x,y)$ par la réponse impulsionnelle $h(x,y)$ du filtre. La convolution discrète est commutative et s'exprime par :

$$J(x,y) = I(x,y) * h(x,y) = \sum_i \sum_j I(x-i, y-j) h(i,j) = \sum_i \sum_j I(i,j) h(x-i, y-j) \tag{1.19}$$

Si $h(i,j)$ est à support borné symétrique, on la représente sous forme d'un tableau de coefficients $H_{m\times n}$ appelé masque de convolution que l'on applique sur la fenêtre image correspondante pour chaque pixel. Par exemple, pour $m = 3$ et $n = 5$, on aura :

$$H_{3\times 5} = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ h_{31} & h_{32} & h_{33} & h_{34} & h_{35} \end{bmatrix} \tag{1.20}$$

La convolution de deux masques 1-D donne un masque 2-D (propriété de séparabilité) :

$$\begin{bmatrix} a & b & c \end{bmatrix} * \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} a\alpha & b\alpha & c\alpha \\ a\beta & b\beta & c\beta \\ a\gamma & b\gamma & c\gamma \end{bmatrix} \tag{1.21}$$

1.5.3 Typologie des filtres

Un filtre peut avoir de nombreuses propriétés [7, 8, 9, 10] : invariance, causalité, anti-causalité, support borné, symétrie, séparabilité, RIF ou RII. Un point particulier à résoudre concerne les effets de bords et la représentation numérique du résultat de filtrage (problème de visualisation de valeurs négatives).

La séparabilité est un critère important pour réduire le temps de calcul.

Les filtres RIF (réponse impulsionnelle finie) sont les plus courants et les masques de taille 3×3, opérant sur les 8 plus proches voisins d'un pixel, sont très utilisés pour tous les traitements élémentaires sur les images :

- moyenneur : $H = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ séparable : $\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} * \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$
- binomial gaussien : $H = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ séparable : $\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$
- laplacien : $H = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
- Sobel horizontal : $H = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ séparable : $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$
- gradient oblique : $H = \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$
- rehausseur de contraste : $H = \begin{bmatrix} 1 & -3 & 1 \\ -3 & 9 & -3 \\ 1 & -3 & 1 \end{bmatrix}$ séparable : $\begin{bmatrix} -1 & 3 & -1 \end{bmatrix} * \begin{bmatrix} -1 \\ 3 \\ -1 \end{bmatrix}$

La mise en cascade de filtres (associativité de la convolution) permet de générer des filtres de grande taille à moindre coût de calculs :

$$H_{3\times3} * H_{3\times3} = \frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix} * \frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix} = \frac{1}{256}\begin{bmatrix}1&4&6&4&1\\4&16&24&16&4\\6&24&36&24&6\\4&16&24&16&4\\1&4&6&4&1\end{bmatrix} = H_{5\times5} \tag{1.22}$$

Les filtres RII (réponse impulsionnelle infinie) sont caractérisés par leur fonction de transfert en Z ou leur équation aux différences :

$$H(z) = \frac{S(z)}{E(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}} \tag{1.23}$$

$$s_k = b_0 e_k + b_1 e_{k-1} + \cdots + b_m e_{k-m} - a_1 s_{k-1} - \cdots - a_n s_{k-n} \tag{1.24}$$

où s_k et e_k sont les échantillons resp. de sortie et d'entrée courants. Si les coefficients a_i sont tous nuls, on retrouve un filtre RIF. La stabilité du filtre est liée aux pôles de la fonction de transfert. Un exemple type de filtre RII est le filtre de Canny-Deriche.

1.5.4 Transformée en Z

La transformée en Z est l'outil classique pour l'étude des signaux échantillonnés. Pour un signal 1-D $f(x)$, elle est définie par :

$$F(z) = \sum_{k=-\infty}^{+\infty} f(k) z^{-k} \tag{1.25}$$

z est un nombre complexe lié à la variable symbolique de Laplace par l'équation :

$$z = \exp pT_e = \exp(j2\pi\nu T_e) = \exp(j2\pi u) \tag{1.26}$$

où T_e est la période d'échantillonnage. z^{-1} est donc l'opérateur retard pur.

1.5.5 Fonction de transfert

Par définition, la fonction de transfert est la TF de la réponse impulsionnelle. Dans le cas d'un filtre 1-D, la TF de $h(x)$ vaut :

$$H(u) = \sum_k h(k) e^{-i2\pi u.k} \tag{1.27}$$

où $h(k)$ sont les coefficients de la réponse impulsionnelle. Pour le filtre moyenneur symétrique $H = \frac{1}{3}[1 \quad 1 \quad 1]$, on obtient :

$$H(u) = \frac{1}{3}(e^{-i2\pi u} + 1 + e^{+i2\pi u}) = \frac{1}{3}(1 + 2\cos 2\pi u) \tag{1.28}$$

On peut aussi utiliser la TZ. Considérons par exemple le filtre binomial centré : $H = [1 \quad 2 \quad 1]$. La convolution du signal $f(x)$ par le filtre $h(x)$ donne : $g(x) = f(x) * h(x) = f(x-1) + 2f(x) + f(x+1)$.

Soit, par transformée en Z : $G(z) = (z^{-1} + 2 + z)F(z)$. D'où la fonction de transfert en Z :

$$H(z) = \frac{G(z)}{F(z)} = z^{-1} + 2 + z \tag{1.29}$$

Pour une implantation matérielle de ce filtre, on décale la réponse impulsionnelle pour la rendre causale en multipliant la fonction de transfert par l'opérateur retard z^{-1}. D'où $H^-(z) = z^{-2} + 2z^{-1} + 1 = (1 + z^{-1})^2$. On en déduit ainsi deux schémas fonctionnels possibles pour réaliser ce filtre numérique (Fig. 1.5).

Ceci est le domaine de la synthèse des filtres numériques en traitement d'images [11, 12, 13, 14].

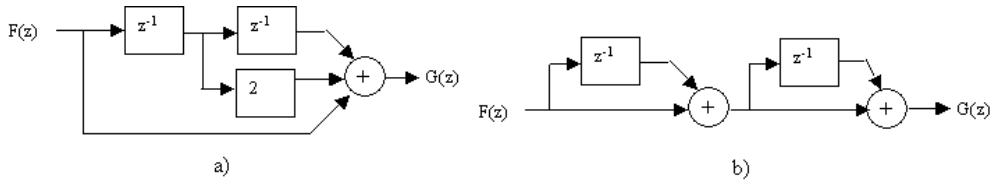

FIGURE 1.5 – Schémas fonctionnels du filtre binomial 1-D : a) direct b) version pipeline.

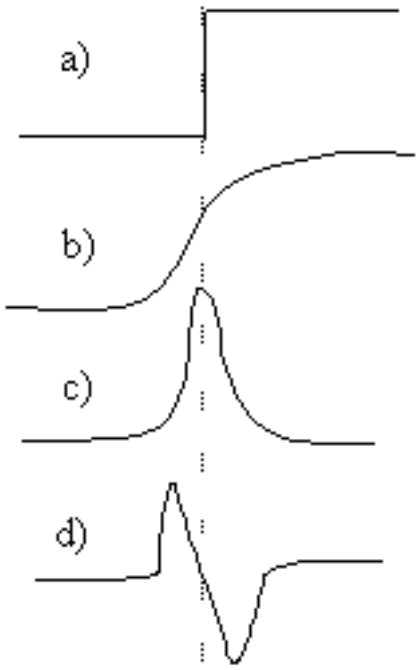

FIGURE 1.6 – a) Contour 1D idéalisé ; b) Lissage ; c) Dérivée première ; d) Dérivée seconde.

1.6 Détection de contours

On distingue deux méthodes de détection de points de contours [15] (Fig. 1.6) :
– la recherche des extrêma de la dérivée première : vecteur gradient défini par

$$G(x,y) = \nabla I(x,y) = \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{bmatrix}$$

– la recherche des passages par zéro de la dérivée seconde : laplacien défini par

$$L(x,y) = \nabla^2 I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2}$$

Marr propose dans son schéma de principe brut du processus de vision (*raw primal sketch* ou esquisse primitive fondamentale) d'utiliser un filtre passe-bande de type laplacien d'une gaussienne $\nabla^2 G$, que l'on peut correctement approximer en pratique avec un filtre *DOG* différence de deux gaussiennes [16].

1.6.1 Détecteur de Roberts

Il est basé sur deux masques qui calculent le gradient spatial dans deux directions 45° et 135° :

$H_{135} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ $\qquad$ $H_{45} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

En convoluant l'image avec ces deux masques, on obtient deux images filtrées I_1 et I_2 que l'on

combine pour calculer le module et la direction du contraste :

$$M(i,j) = \sqrt{I_1(i,j)^2 + I_2(i,j)^2} \tag{1.30}$$

$$\Phi(i,j) = \arctan \frac{I_2(i,j)}{I_1(i,j)} + \frac{\pi}{4} \tag{1.31}$$

On applique ensuite une technique de seuillage du module pour ne garder que les points contours. Se pose ensuite le problème de chaînage et d'approximation polygonale pour exhiber des contours fermés.

Ce détecteur est sensible au bruit haute fréquence, que l'on peut réduire par un filtrage passe-bas initial.

1.6.2 Détecteur de Sobel

Il utilise également deux masques, l'un horizontal H_0, l'autre vertical H_{90} :

Sobel vertical : $H_{90} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ séparable en lissage et dérivation : $\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$

La procédure est ensuite la même pour extraire les contours, mis à part la suppression du terme additif $\frac{\pi}{4}$ dans l'Eq. (1.31).

NB : Incluant un pré-filtrage passe-bas, Sobel est moins sensible au bruit que Roberts.

1.6.3 Détecteur de Prewitt

Il consiste en deux masques élémentaires :

Prewitt horizontal : $H_0 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ et Prewitt vertical : $H_{90} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

1.6.4 Détecteur de Canny-Deriche

Il fait partie des extracteurs par optimisation de critères (sensibilité, localisation et détection optimales étant donné un modèle de contour). Sa réponse impulsionnelle 1-D s'exprime par :

$$h(x) = cxe^{-\alpha|x|} \tag{1.32}$$

α est un coefficient de lissage et c un coefficient de normalisation déterminé en maximisant l'amplitude de la réponse à un échelon unité : $c = -\frac{(1-e^{-\alpha})^2}{e^{-\alpha}}$.

Il se décompose en deux parties, dont on peut calculer les fonctions de transfert en Z :

- causale : $h^+(x) = cxe^{-\alpha x}$ pour $x \geq 0$ d'où $H^+(z) = c\sum_{k=0}^{+\infty} kz^{-k}e^{-\alpha k} = c\frac{e^{-\alpha}z^{-1}}{(1-e^{-\alpha}z^{-1})^2}$
- anti-causale : $h^-(x) = cxe^{\alpha x}$ pour $x \leq 0$ d'où $H^-(z) = c\sum_{k=0}^{+\infty} -kz^{k}e^{-\alpha k} = c\frac{-e^{-\alpha}z}{(1-e^{-\alpha}z)^2}$

On en déduit aisément les équations de récurrence du filtre.

1.6.5 Détecteur de Shen-Castan

Le détecteur de Shen et Castan est une version de filtre RII plus simple que le filtre de Canny-Deriche : un unique paramètre permet de régler la force du filtre, le second paramètre étant le choix du seuil pour ne conserver que les points ayant les plus fortes valeurs de dérivées. Il a été utilisé avec succès en contrôle qualité automatique pour mesurer la mouillabilité de polymères traités par plasma [17, 18].

1.6.6 Algorithme de Rosenfeld & Kak

Contrairement aux filtres précédents qui opèrent sur des images à NdG, il s'agit ici d'un extracteur de contours opérant sur des images binaires (N&B).

Définitions

On appelle *direction courante* du contour de l'objet en un pixel appartenant au contour, le vecteur de translation qui joint ce pixel au pixel du contour le précédant dans le sens trigonométrique.

Partant d'un pixel donné dans l'image, les 8 premiers voisins sont définis par leur direction par rapport au pixel de départ et par les coordonnées des vecteurs de translation permettant de passer d'un pixel au suivant connaissant la direction courante (code de Freeman, cf. Fig .1.7).

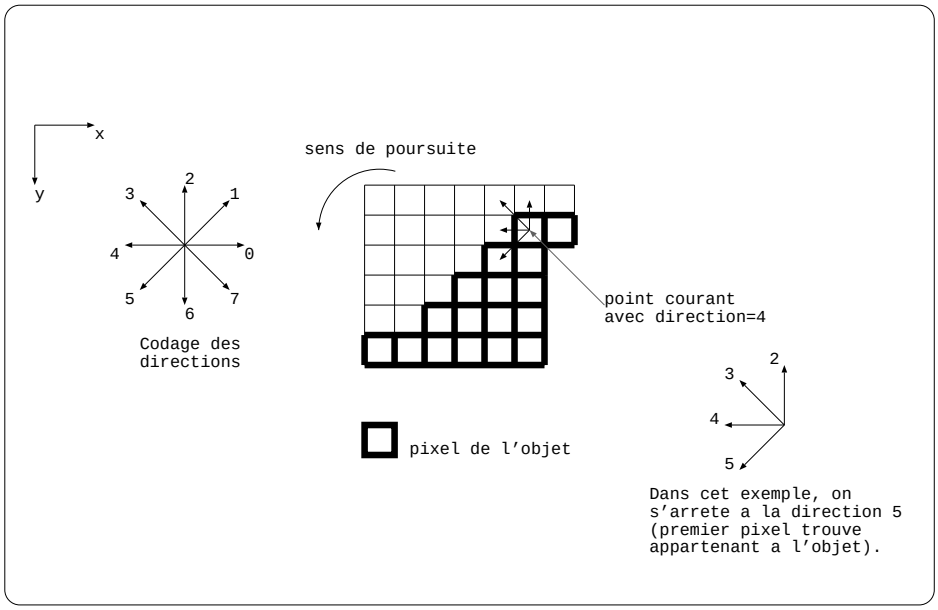

FIGURE 1.7 – Poursuite de contours dans une image binaire.

Principe de l'algorithme

Duda et Hart suggèrent un algorithme très rapide pour extraire les frontières d'un objet [19]. L'inconvénient de leur algorithme provient du fait que le suiveur de contour ne prend que 4 directions (directions 0, 2, 4, 6, du code de Freeman). Autrement dit, il explore 4 voisins du pixel de départ. Ainsi les pixels reliés à l'objet selon les directions diagonales sont ignorés. On lui préfère donc l'algorithme de Rosenfeld et Kak, qui utilise une exploration à 8 voisins dans le sens trigonométrique [20].

Le principe est le suivant : supposons que l'on connaisse un point de contour et la direction courante du contour en ce point. La recherche du point suivant consiste en un balayage des 8 voisins du point courant. Le balayage est effectué de l'extérieur vers l'intérieur de l'objet. Le premier voisin qui appartient à l'objet devient le nouveau point courant du contour et la direction courante est celle qui joint les deux points contours. En examinant tous les cas de figure possibles, on constate qu'il suffit d'explorer un secteur limité à *direction courante* -2 jusqu'à *direction courante* +4.

Pour accélérer l'exécution du programme de suivi de contours, les directions à tester et les composantes de vecteurs de translation (qui permettent de passer d'un pixel au suivant, connaissant la direction courante) sont regroupées dans des tableaux (Tab. 1.1).

Un exemple d'application pour l'extraction des lèvres d'un visage est montré Fig. 1.8.

Choix du point initial pour la pousuite des contours La détermination du point de départ pour la poursuite des contours est simple quand on n'a qu'un seul objet binaire connexe. Mais un masque de lèvres est constitué pratiquement de deux objets : les lèvres et l'intérieur de la bouche. Par conséquent deux points initiaux, au moins, sont nécessaires : l'un pour détecter le contour externe, et l'autre pour détecter le contour de l'intérieur de la bouche (contour interne des lèvres).

On effectue la détection de contour externe en partant du point initial (x_{moy}, y_1). Ce point est le point de contour externe de la lèvre supérieure situé sur l'axe vertical médian des lèvres. Partant de y_{min} dans le sens croissant des y, ce point est rapidement localisé : il s'agit du premier point appartenant au masque des lèvres sur l'axe de celles-ci (Fig. 1.9). La direction initiale D_0 de contour en ce point de

Direction courante	0	1	2	3	4	5	6	7
v_x	1	1	0	-1	-1	-1	0	1
v_y	0	-1	-1	-1	0	1	1	1
Directions à explorer	6	7	0	1	2	3	4	5
	7	0	1	2	3	4	5	6
	0	1	2	3	4	5	6	7
	1	2	3	4	5	6	7	0
	2	3	4	5	6	7	0	1
	3	4	5	6	7	0	1	2
	4	5	6	7	0	1	2	3

TABLE 1.1 – Coordonnées (v_x, v_y) des vecteurs de translation et Directions à explorer en fonction de la direction courante

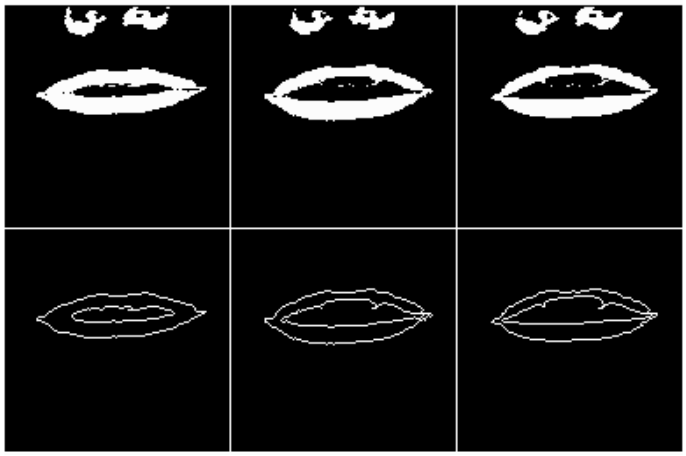

FIGURE 1.8 – Détection des contours des lèvres par l'algorithme de Rosenfeld et Kak avec deux points initiaux. En haut : masques binaires de lèvres ; en bas : contours extraits.

départ est fixée à 6 (sens de recherche du point initial) pour assurer le démarrage de la poursuite dans le sens trigonométrique.

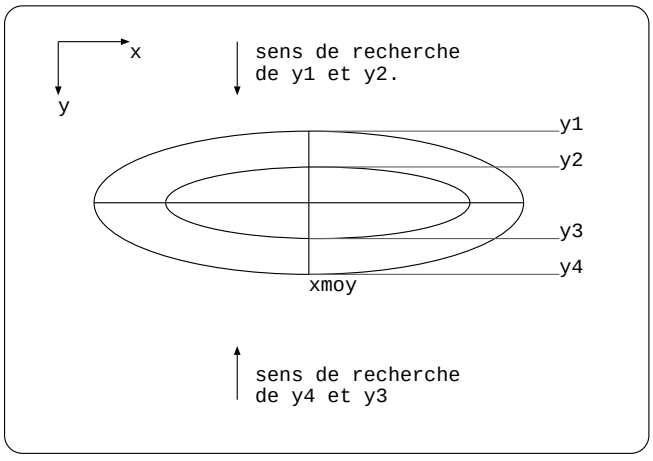

FIGURE 1.9 – Les 4 points initiaux candidats

On cherche de même un point initial du contour interne de la lèvre supérieure situé sur l'axe vertical médian. L'algorithme d'initialisation recherche, en partant de y_1 dans le sens croissant des y (vers l'intérieur de la bouche), le point limite du masque comme point initial. On obtient alors le point (x_{moy}, y_2), (Fig. 1.9). Dans ce cas, la direction initiale est fixée à 2.

NB : On peut aussi bien utiliser le couple de points initiaux y_3 et y_4.

1.7 Morphologie mathématique

1.7.1 Introduction

La morphologique mathématique concerne des transformations géométriques non linéaires (filtrage non linéaire). Elle est basée sur la théorie des ensembles. On applique à l'image un élément structurant E. On peut faire une analogie avec le filtrage linéaire : E correspond au masque de coefficients d'un filtre. L'opération morphologique correspond à une convolution. Les opérations de base sont l'érosion, la dilatation, l'ouverture et la fermeture. On distingue la morphologie binaire, s'appliquant aux images en N&B, et la morphologie pour images à NdG.

Le principe général consiste à comparer les objets d'une image avec un objet de référence de forme et de taille données qu'on appelle élément structurant. Ce type de filtrage est recommandé dans le cas d'un bruit important (bords irréguliers, trous à l'intérieur des objets), car il permet de boucher les trous et adoucir les angles. L'élément structurant E translaté en chaque site s de l'image est noté E_s. La Fig. 1.10 montre des éléments structurants classiques.

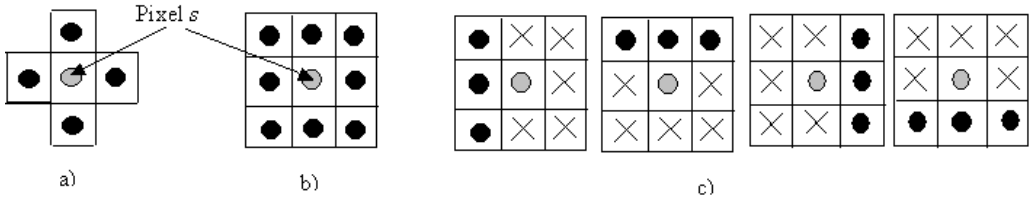

FIGURE 1.10 – Eléments structurants : a) croix 4-connexe b) carré 8-connexe c) traits pivotés (×=indifférent)

Tout l'enjeu de la morphologie mathématique repose évidemment sur le bon choix de l'élément structurant, en fonction du problème à résoudre.

1.7.2 Opérations binaires (images N&B)

L'érosion $\ominus$ s'obtient en considérant l'"intersection" (inclusion) de l'objet X avec l'élément structurant E :

$$X \ominus E = X \cap E = \{s | E_s \subseteq X\} \tag{1.33}$$

La dilatation $\oplus$ s'obtient en considérant l'"union" de l'objet avec l'élément structurant :

$$X \oplus E = X \cup E = \{s | E_s \cap X \neq \emptyset\} \tag{1.34}$$

L'ouverture $\circ$ est la succession érosion puis dilatation ("roll inside") :

$$X \circ E = (X \ominus E) \oplus E \tag{1.35}$$

La fermeture $\bullet$ est la succession dilatation puis érosion ('roll outside') cf. Fig. 1.11 :

$$X \bullet E = (X \oplus E) \ominus E \tag{1.36}$$

1.7.3 Propriétés

La dilatation est commutative et associative.

L'érosion et la dilatation sont invariantes en translation :

$$X_s \oplus E = (X \oplus E)_s \tag{1.37}$$

$$X_s \ominus E = (X \ominus E)_s \tag{1.38}$$

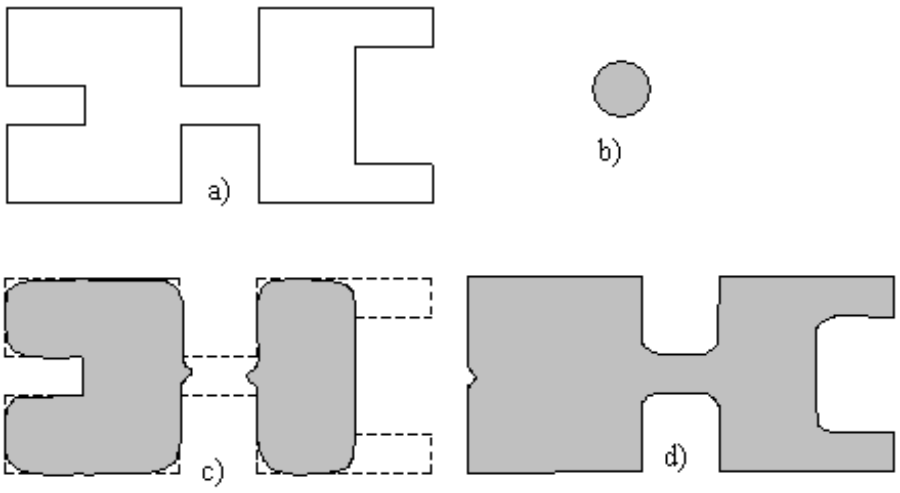

FIGURE 1.11 – a) objet b) élément structurant c) ouverture d) fermeture

Croissance :

$$X \subseteq Y \Rightarrow \begin{cases} X \oplus E \subseteq Y \oplus E \\ X \ominus E \subseteq Y \ominus E \\ X \bullet E \subset Y \bullet E \\ X \circ E \subset Y \circ E \end{cases} \tag{1.39}$$

Distributivité :

$$(X \cup Y) \oplus E = (X \oplus E) \cup (Y \oplus E) \tag{1.40}$$
$$X \oplus (D \cup E) = (X \oplus D) \cup (X \oplus E) \tag{1.41}$$
$$X \ominus (D \cup E) = (X \ominus D) \cap (X \ominus E) \tag{1.42}$$
$$(X \cap Y) \ominus E = (X \ominus E) \cap (Y \ominus E) \tag{1.43}$$
$$\tag{1.44}$$

Itérativité :

$$(X \oplus D) \oplus E = X \oplus (D \oplus E) \tag{1.45}$$
$$(X \ominus D) \ominus E = X \ominus (D \oplus E) \tag{1.46}$$

Complémentarité (Dualité entre fond et forme) :
le complémentaire de l'ensemble X étant défini par : $X^c = \{s | s \notin X\}$, on a :

$$(X \ominus E)^c = X^c \oplus E \tag{1.47}$$
$$(X \bullet E)^c = X^c \circ E \tag{1.48}$$
$$(X \circ E)^c = X^c \bullet E \tag{1.49}$$

Extensivité et anti-extensivité :

$$X \circ E \subset X \subset X \bullet E \tag{1.50}$$

Idempotence :

$$(X \bullet E) \bullet E = X \bullet E \tag{1.51}$$
$$(X \circ E) \circ E = X \circ E \tag{1.52}$$

1.7.4 Autres opérations

L'opération Tout-ou-Rien (*Hit or Miss*) $\oslash$ s'obtient à l'aide d'un élément structurant plus général $E = (E_1, E_2)$ composé d'une forme E_1 et d'un fond E_2.

$$X \oslash E = (X \ominus E_1) \cap (X^c \ominus E_2) = (X \ominus E_1) - (X \oplus E_2) \tag{1.53}$$

Cet ensemble représente tous les points où, simultanément, E_1 'colle' dans X et E_2 'colle' dans X^c.

L'extraction de frontières $\beta(X)$ s'obtient par différence entre X et son érodée par l'élément structurant carré de la Fig. 1.10 b.

$$\beta(X) = X - (X \ominus E) \tag{1.54}$$

Le remplissage de région s'obtient à partir de dilatations itérées :

$$X_k = (X_{k-1} \oplus E) \cap X^c \qquad k = 1, 2, 3, ... \tag{1.55}$$

où X est la frontière initiale et $X_0 = \{p\}$ un point initial intérieur à la frontière. L'union de X et des X_k représente le résultat du remplissage. L'élément structurant habituellement utilisé est la croix (Fig. 1.10 a).

L'extraction de composants connexes s'obtient par :

$$X_k = (X_{k-1} \oplus E) \cap X \qquad k = 1, 2, 3, ... \tag{1.56}$$

où X est l'objet initial et $X_0 = \{p\}$ un point de connexion initial connu.

La coque convexe C d'un objet (*convex hull*, Fig. 1.12) s'obtient par itérations de tout-ou-rien, à l'aide de quatre éléments structurants pivotés E_i avec $i = 1, 2, 3, 4$ (Fig. 1.10 c) :

$$X_k^i = (X_{k-1}^i \oslash E_i) \cup X \quad i = 1, 2, 3, 4 \quad k = 1, 2, 3, ... \quad X_0^i = X \tag{1.57}$$

$$C = \bigcup_{i=1}^{4} X_{conv}^i \tag{1.58}$$

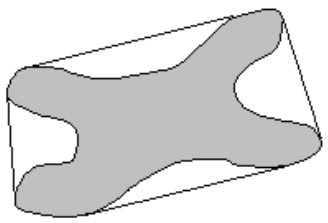

FIGURE 1.12 – Coque convexe

L'amincissement $\otimes$ enlève des points à la frontière de X :

$$X \otimes E = X - (X \oslash E) = X \cap (X \oslash E)^c \tag{1.59}$$

L'épaississement $\odot$ ajoute à X des points de la frontière de X^c :

$$X \odot E = X \cup (X \oslash E) \tag{1.60}$$

La squelettisation s'obtient par amincissements répétés :

$$S = X \otimes \{E\} = ...((((X \otimes E_1) \otimes E_2) \otimes E_3) \otimes E_4) \otimes E_1) \otimes E_2)... \tag{1.61}$$

Il existe des opérations plus complexes comme la taille ('pruning') qui utilise des érosions successives avec des éléments structurants de plus en plus fins (tamis) permettant de classifier les régions.

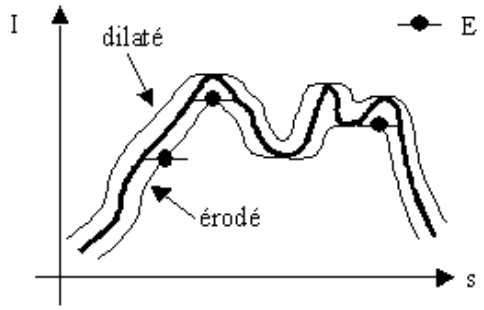

FIGURE 1.13 – Erosion et dilatation en NdG

1.7.5 Images NdG

L'érosion $\ominus$ donne la plus petite valeur prise par I à l'intérieur de l'élément structurant E_s centré en le pixel s (Fig. 1.13) :

$$\forall s, (I \ominus E)(s) = Inf\{I(p); p \in E_s\} \tag{1.62}$$

La dilatation $\oplus$ donne la plus grande valeur prise par I sur le support structurant E_s :

$$\forall s, (I \oplus E)(s) = Sup\{I(p); p \in E_s\} \tag{1.63}$$

L'ouverture $\circ$ et la fermeture $\bullet$ sont définies comme en morphologie binaire. Intuitivement, l'ouverture consiste à suivre le profil par le dessous en montant l'élément structurant le plus haut possible ; cela revient à écrêter les pics sans toucher aux vallées. De même, la fermeture consiste à suivre le profil par le dessus en descendant E le plus bas possible ; cela revient à combler les vallées sans toucher aux pics.

L'opération chapeau haut de forme (*top hat*) $H = I - I \circ E$ permet d'exhiber les pics de I. De même, on peut extraire les creux en soustrayant $I - I \bullet E$.

Le lissage peut s'obtenir par ouverture puis fermeture : $L = (I \circ E) \bullet E$.

Le gradient morphologique correspond à la soustraction entre dilatée et érodée : $G = (I \oplus E) - (I \ominus E)$

1.8 Segmentation couleur

1.8.1 Vision des couleurs

L'intérêt d'utiliser la couleur en traitement d'image repose sur plusieurs constats. D'abord, la couleur est un descripteur puissant car l'œil humain peut discerner des milliers de couleurs, alors qu'il ne peut distinguer que quelques dizaines de niveaux de gris. Par ailleurs, la teinte est une grandeur peu sensible aux variations d'éclairage (ombres), contrairement à la luminance.

Les principaux constituants de l'œil sont présentés Fig 1.14 :

- la cornée : protection, filtre
- l'iris : diaphragme (variation d'un facteur 10 en surface)
- le cristallin : indice optique variable, focus (déformable)
- la rétine : couche de capteurs photo-sensibles (120 millions de bâtonnets et 6 millions de cônes). Les cônes sont sensibles aux trois couleurs RVB (vision chromatique) et les bâtonnets à l'intensité lumineuse I (vision achromatique). La fovéa est le centre de la rétine. La zone aveugle correspond au rattachement du nerf optique.
- le nerf optique : transport de l'information (100000 neurones)

La vision humaine n'a pas une sensibilité égale dans tout le spectre visible : l'énergie lumineuse perçue par l'œil est fonction de la longueur d'onde comprise entre 400 et 700nm (Fig. 1.15).

1.8.2 Trichromie

Alors que les images achromatiques (en NdG) sont caractérisées par une seule valeur (intensité lumineuse ou *luminance* I), les images couleurs sont caractérisées par trois composantes [21].

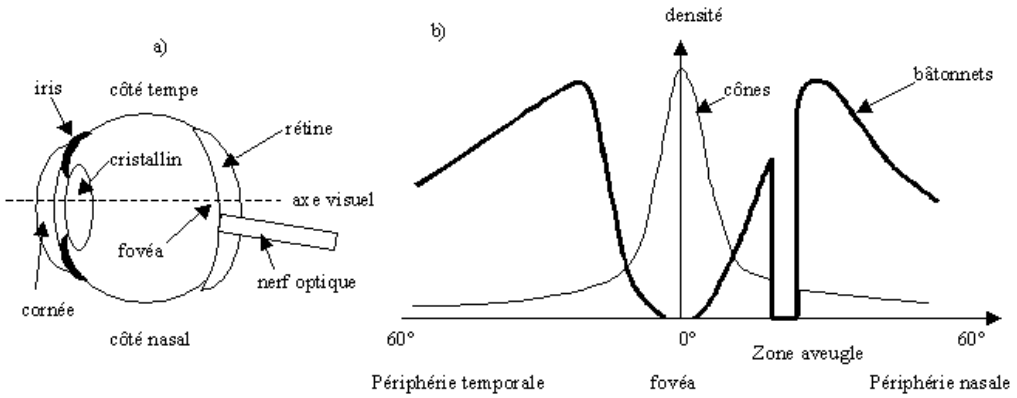

FIGURE 1.14 – a) Coupe horizontale de l'œil ; b) Répartition des photo-récepteurs sur la rétine.

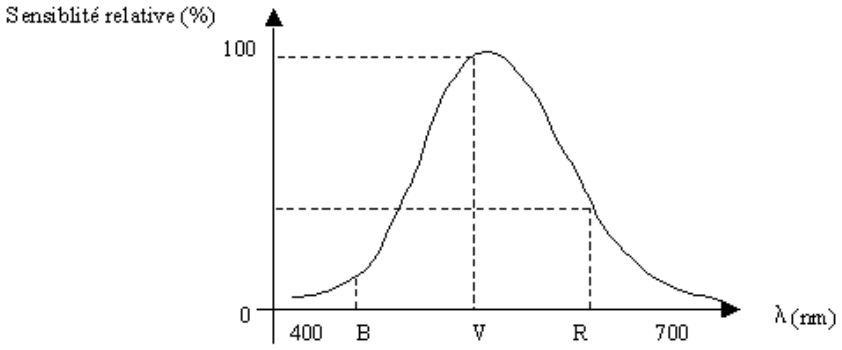

FIGURE 1.15 – Courbe de sensibilité de l'œil en fonction de la longueur d'onde.

On peut utiliser les trois couleurs primaires : rouge, vert et bleu (RVB), correspondant à trois longueurs d'onde du spectre visible : $\lambda_R = 610nm$, $\lambda_V = 535nm$, $\lambda_B = 470nm$ (Fig. 1.15). Les autres couleurs visibles s'obtiennent par combinaison des couleurs primaires (synthèse additive, Fig. 1.16a).

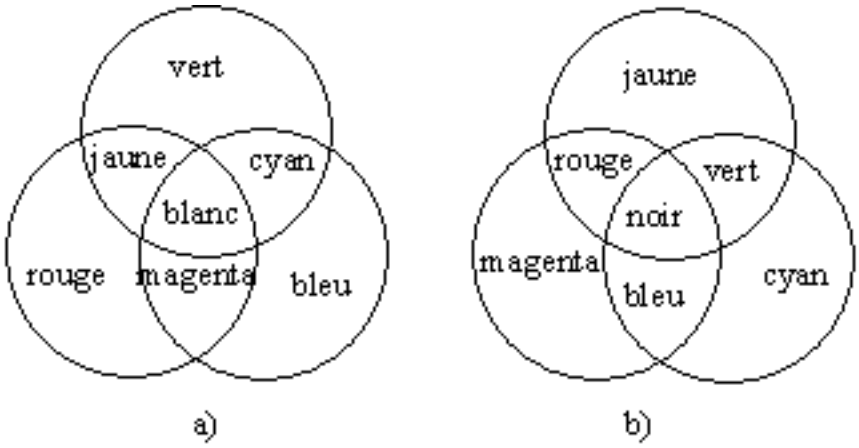

FIGURE 1.16 – Synthèse des couleurs : a) additive ; b) soustractive

Le modèle RVB est typiquement utilisé pour la réalisation des moniteurs et caméras couleurs. On représente l'espace RVB par un cube (Fig. 1.17a) :

Les couleurs secondaires cyan, magenta et jaune (CMY) s'obtiennent par additivité des couleurs primaires :

$$C = V + B \tag{1.64}$$

$$M = R + B \tag{1.65}$$

$$Y = R + V \tag{1.66}$$

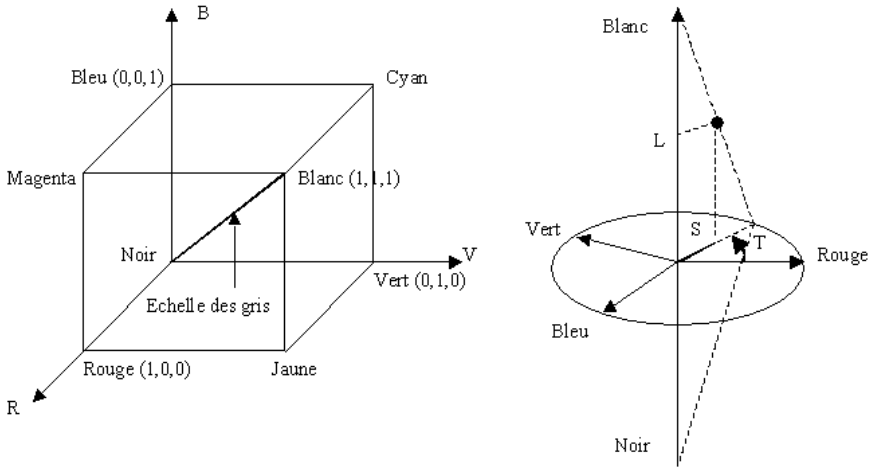

FIGURE 1.17 – a) Cube couleur RVB ; b) Pyramide couleur TLS

Le modèle CMY est typiquement utilisé pour les imprimantes couleurs (synthèse soustractive, Fig. 1.16b).

Un troisième modèle nommmé YC_rC_b est utilisé comme standard en TV couleur, assurant la compatibilité avec la TV N&B. Y représente la luminance. C_r, C_b sont deux composantes codant la chrominance (par différences de couleurs). Leur définition (et leur appellation) varie selon le format : YUV pour les formats européens (PAL, Secam), YIQ pour le format nord-américain (NTSC).

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.3 & 0.59 & 0.11 \\ -1.33 & 1.12 & 0.21 \\ -0.45 & -0.88 & 1.33 \end{bmatrix} . \begin{bmatrix} R \\ V \\ B \end{bmatrix} \tag{1.67}$$

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.3 & 0.59 & 0.11 \\ 0.6 & -0.28 & -0.32 \\ 0.21 & -0.52 & 0.31 \end{bmatrix} . \begin{bmatrix} R \\ V \\ B \end{bmatrix} \tag{1.68}$$

D'un point de vue **perception**, les caractéristiques pertinentes pour distinguer les couleurs sont la teinte, la luminance et la saturation. On définit ainsi le modèle TLS (ou HSI en anglais 'hue, saturation, intensity').

$$T = \theta = \arccos \frac{1}{2} \frac{(R-V)+(R-B)}{\sqrt{(R-V)^2+(R-B)(R-V)}} \quad \text{si } B < V \tag{1.69}$$

$$= 2\pi - \theta \quad \text{si } B > V \tag{1.70}$$

$$L = \frac{R+V+B}{3} \tag{1.71}$$

$$S = 1 - \frac{\min(R,V,B)}{L} \tag{1.72}$$

A la place de la saturation, on peut utiliser la pureté $P = S.L$, et adopter un calcul de teinte basé sur l'arctangente (espace TLP) :

$$\begin{aligned} T &= \frac{\pi}{2} - \arctan\left(\frac{2R-V-B}{\sqrt{3}(V-B)}\right) + k \\ L &= \frac{R+V+B}{3} \\ P &= \frac{R+V+B}{3} - \min(R,V,B) \end{aligned} \tag{1.73}$$

où : $k = 0$ si $V > B$ et $k = \pi$ sinon.

Il existe donc de nombreux espaces pour modéliser l'information de couleur. Un bon espace correspond à un choix de trois composantes bien décorrélées : c'est tout l'enjeu des différentes transformées couleurs proposées. Un autre problème essentiel est la sensibilité au bruit du calcul de la teinte, reposant sur un calcul de rapport de différences.

Pour palier ce problème, certaines approches s'inspirent de constatations concernant le système visuel humain :

- le premier traitement réalisé par l'œil est une compression logarithmique de l'information qu'il perçoit
- les capteurs majoritaires au centre de la fovéa sont les cônes sensibles au R et au V.
- la courbe de sensibilté de l'œil est maximale pour le vert.

Ces remarques conduisent à proposer des espaces couleurs qui offrent des alternatives intéressantes pour faire de la segmentation couleur robuste, que ce soit pour des applications en robotique, en visiophonie ou en visioconférence. Le système 1.74 donne l'expression de la transformée logarithmique LUX [22, 23, 24].

$$\begin{aligned} L &= (R+1)^{0.3}(G+1)^{0.6}(B+1)^{0.1} - 1 \\ U &= \begin{cases} \frac{M}{2}\left(\frac{R+1}{L+1}\right) & \text{if } R < L \\ M - \frac{M}{2}\left(\frac{L+1}{R+1}\right) & \text{otherwise} \end{cases} \\ X &= \begin{cases} \frac{M}{2}\left(\frac{B+1}{L+1}\right) & \text{if } B < L \\ M - \frac{M}{2}\left(\frac{L+1}{B+1}\right) & \text{otherwise} \end{cases} \end{aligned} \tag{1.74}$$

L'effet principal de cette transformée logarithmique est d'amplifier le contraste tout en s'affranchissant le plus possible de l'éclairage. La figure 1.18 montre que l'espace LUX augmente le contraste des composantes chromatiques tout en gardant la cohérence des teintes (rouge, bleu).

FIGURE 1.18 – Transformées chromatiques sur une image de visage. *En haut :* Les composantes Y, C_r et C_b ; *En bas :* Les composantes L, U et X.

1.8.3 Segmentation couleur

Projection sur un axe

Pour réduire la quantité d'information apportée par la chrominance, on peut ne s'intéresser qu'à la projection sur un seul des axes. Un axe privilégié est l'axe luminance car :

- c'est la projection intrinsèque faite par une caméra vidéo N&B
- elle correspond à la composante L de l'espace TLS
- elle se calcule facilement en faisant la moyenne des quantités RVB

Pour une application spécifique (suivi de teinte de visage par exemple), toute autre projection est possible (par exemple sur l'axe rouge).

Analyse en composantes principales

Une image couleur peut être considérée comme un ensemble d'échantillons statistiques $I(i,j) = \begin{bmatrix} R(i,j) \\ V(i,j) \\ B(i,j) \end{bmatrix}$ caractérisé par ses moments d'ordre 1 (vecteur moyenne $M = \begin{bmatrix} M_R \\ M_V \\ M_B \end{bmatrix}$) et d'ordre 2 (matrice de covariance symétrique $C = \begin{bmatrix} C_{RR} & C_{RV} & C_{RB} \\ C_{RV} & C_{VV} & C_{VB} \\ C_{RB} & C_{VB} & C_{BB} \end{bmatrix}$).

Dans le cas discret, ces grandeurs se calculent par :

$$M_X = E[X] = \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N} X(i,j) \tag{1.75}$$

$$C_{XY} = E[(X-M_X)(Y-M_Y)] = \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}(X(i,j)-M_X)(Y(i,j)-M_Y) \tag{1.76}$$

L'analyse en composantes principales (ACP) permet de définir une projection plus intelligente. Elle consiste à diagonaliser la matrice de covariance (changement de base) et à ne conserver que la composante correspondant au coefficient maximal, c'est-à-dire celle apportant le plus d'information. Elle est basée sur le calcul des valeurs propres et vecteurs propres de C.

Segmentation par seuillage de la teinte

Un histogramme de couleur fournit une densité de probabilité permettant par seuillage ou filtrage, de trouver les pixels d'une région spécifique, par exemple la peau humaine.

Pour détecter les visages, la teinte à dominante rouge est un discriminateur satisfaisant quelle que soit la couleur effective de la peau (la différence de couleur de peau provient d'une différence de saturation, c'est-à-dire de quantité de couleur présente dans le visage).

L'espace logarithmique LUX peut être radicalement simplifié si l'application ne concerne que l'analyse de visage :

$$I = \frac{R+V+B}{3} \tag{1.77}$$

$$H = \begin{cases} 256 \cdot \frac{V}{R} & \text{si} \quad 0 \le V < R \\ 255 & \text{sinon} \end{cases} \tag{1.78}$$

On utilise uniquement les plans couleur R et V pour deux raisons principales : tout d'abord le visage est principalement rouge et contient peu de bleu ; par ailleurs, le vert est souvent utilisé en traitement vidéo comme une bonne approximation de la luminance. Cette expression a déjà sous d'autres formes fait ses preuves en robotique et vision par ordinateur.

Le visage, principalement rouge, correspond donc à une teinte moyenne de 128 par cette transformée logarithmique. L'équation 1.81 donne la justification de cette affirmation :

$$I = \frac{R+V+B}{3} \quad \text{avec} \quad B \sim 0 \text{ et } I \sim V \tag{1.79}$$

$$\text{d'où} \quad R \sim 2I \sim 2V \tag{1.80}$$

$$\text{soit} \quad H = 256\frac{V}{R} \sim 256\frac{V}{2V} \sim 128 \tag{1.81}$$

La figure 1.19 montre la distribution typique de la teinte sur une image de visage centrée sur les lèvres. Pour éliminer le mode dû au fond, prépondérant dans des conditions de prise de vue télévisées, il suffit de filtrer la distribution de teinte par un filtre centré sur 128 de pente suffisamment douce pour laisser une relative souplesse à l'estimation (filtre gaussien de centre 128, d'écart type 128).

Pour extraire le mode des lèvres, qui se trouve la plupart du temps confondu avec le mode de teinte de la peau du visage, on procède de façon hiérarchique par itération du filtrage : on applique là aussi un filtre gaussien de centre 128 mais d'écart type plus réduit, 64.

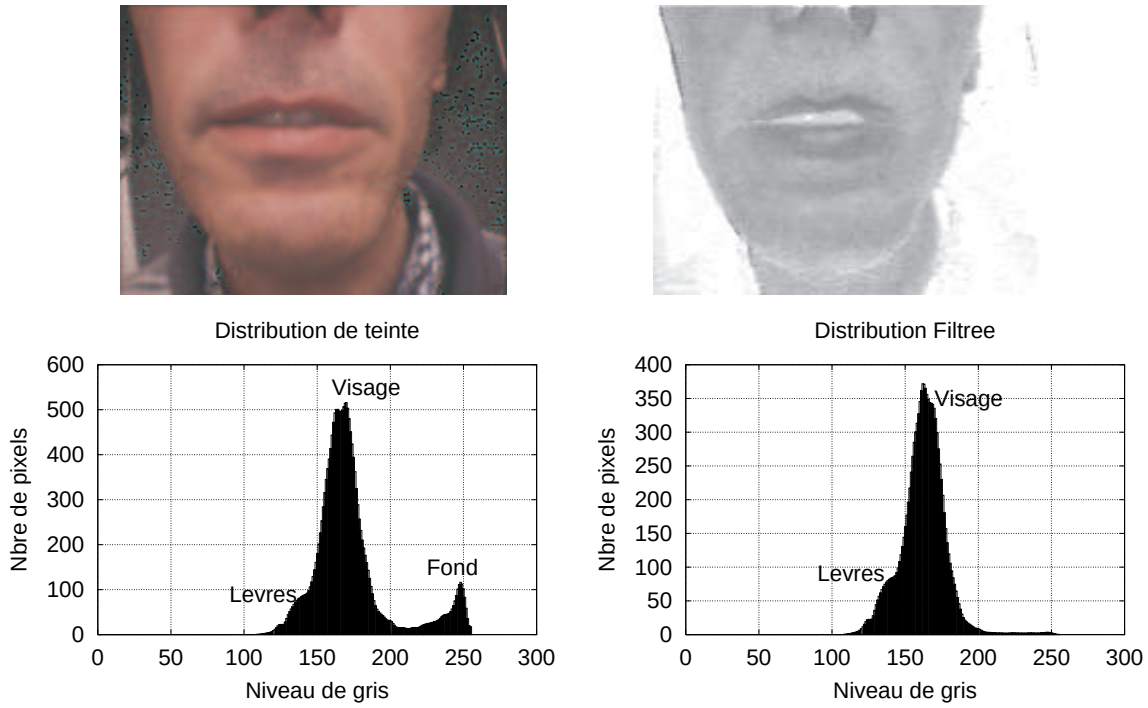

FIGURE 1.19 – En haut : image de visage (couleur), plan teinte logarithmique correspondant. En bas : distribution de teinte, distribution filtrée.

Classification de teinte par approche crédibiliste

La théorie de l'évidence, appélée aussi théorie des fonctions de croyances, est une alternative à la théorie probabiliste bayésienne classique. Elle permet de traiter des situations ambigues, dans le cas de données incomplètes ou bruités, ce qui est souvent la situation rencontrée dans le traitement d'images réelles.

Nous ne la détaillons pas ici, mais on peut consulter les références suivantes qui illustrent son application pour la détection et le suivi de visage basé sur l'information de teinte chair, et où l'on trouvera une bibliographie sur le sujet [25, 26, 27]. Un audioslide est également visionnable en ligne sur youtube [28].

Chapitre 2

Traitement de séquences d'images

2.1 Détection de mouvement

2.1.1 Introduction

L'analyse du mouvement dans les séquences d'images est un domaine de recherche actif, en raison de son importance dans de nombreuses applications : télésurveillance, compression pour les télécommunications ou l'archivage, diagnostic médical, météorologie, contôle non destructif, robotique mobile, etc. On distingue habituellement quatre phases en analyse de mouvement : la *détection* (des zones mobiles), l'*estimation* (des vecteurs-vitesses, en chaque pixel ou pour chaque objet), la *segmentation* (en zones cohérentes au sens du mouvement) et l'*interprétation* de haut niveau (reconnaissance de formes faisant appel à l'intelligence artificielle). Ces quatre étapes ne sont en aucun cas indépendantes ni forcément séquentielles, mais au contraire fortement interdépendantes (notamment en ce qui concerne les deux phases estimation-segmentation). Nous nous intéressons ici à la phase de détection du mouvement des objets mobiles dans le cas d'une caméra fixe par rapport à la scène observée.

2.1.2 Principe

La détection de mouvement consiste à attribuer à chaque pixel ou *site* $s = (x, y, t)$ des images d'une séquence un attribut, qu'on appelle *étiquette* e_s, indiquant si le pixel appartient à un objet mobile ou au fond fixe de la scène observée :

$$e_s = e(x, y, t) = \begin{cases} a = \text{"1"} & \text{si le pixel appartient à une zone mobile,} \\ b = \text{"0"} & \text{si le pixel appartient au fond fixe.} \end{cases} \tag{2.1}$$

Etiqueter chaque pixel s de l'image à l'instant t permet ainsi d'obtenir une carte binaire des changements temporels (Fig. 2.1).

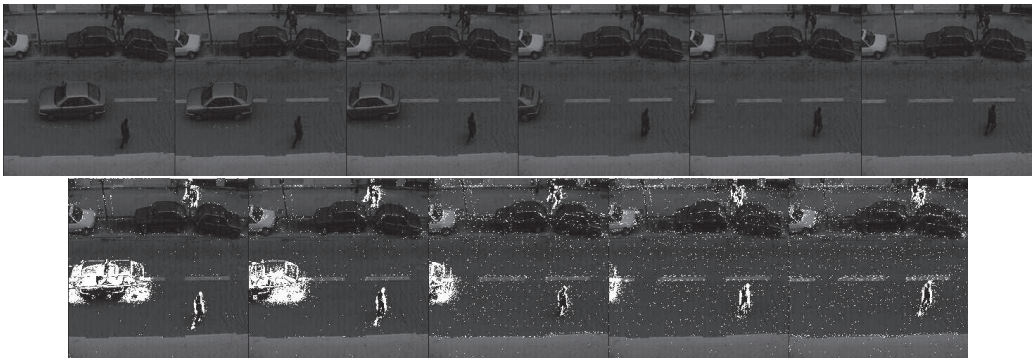

FIGURE 2.1 – En haut : 6 images d'une séquence **Rue** ; En bas : Cartes binaires obtenues par seuillage entropique (pixels mobiles en blanc, $\theta = 4 \cdot \sigma_e$)

En faisant l'hypothèse d'une caméra fixe et d'un éclairement quasi-constant de la scène observée, on peut extraire en chaque pixel une information de bas niveau, qu'on appelle *observation* o_s, portant

sur la variation temporelle de l'intensité lumineuse I du pixel. On utilise comme observation la valeur absolue de la différence temporelle d'intensité lumineuse entre deux instants :

$$o_s = o(x, y, t) = |I_t(s) - I_{t-1}(s)| = |I(x, y, t) - I(x, y, t-1)| \tag{2.2}$$

Dans le cas idéal, cette variation temporelle entre deux instants d'acquisition $t-1$ et t est nulle pour un pixel du fond fixe, non nulle s'il y a eu mouvement d'un objet d'intensité non parfaitement uniforme. Cependant, cette observation de bas niveau fournit une information qui est bruitée et peu robuste (bruit d'acquisition de la caméra, quantification, etc.). Elle ne donne qu'une information partielle sur les objets en mouvement et nécessite un seuillage adéquat. Se pose alors le problème du choix automatique de la valeur du seuil θ [29, 30, 31].

Un simple seuillage ne permet pas d'extraire les objets mobiles. En effet, on distingue typiquement 4 zones dans le champ d'observations (Fig. 2.2) :

- le fond fixe,
- la zone d'écho (zone de fond découverte par l'objet à l'instant t),
- la zone de glissement de l'objet sur lui-même,
- la zone de recouvrement (zone de fond recouverte par l'objet à l'instant t).

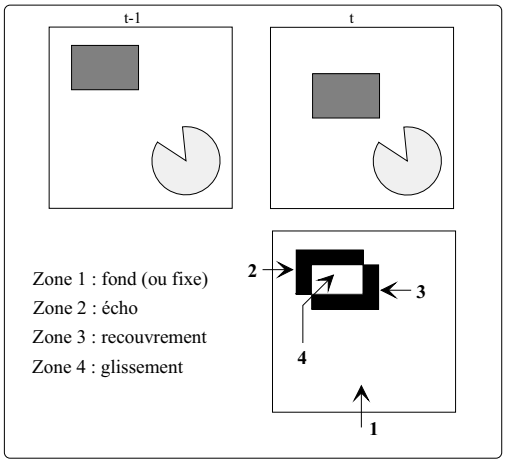

FIGURE 2.2 – Carte binaire des changements temporels entre les instants $t-1$ et t. Cas d'un rectangle mobile et d'un camembert fixe.

Il faut d'une part éliminer l'écho, d'autre part reconstruire l'intérieur des objets mobiles (zone de glissement). Différentes techniques, très simples dans leur principe, mais relativement peu robustes, permettent d'obtenir les zones mobiles :

- la technique de simple différence par rapport à une image de référence censée ne contenir que le fond fixe de la scène : le problème est bien sûr l'obtention de cette image de référence, qu'il faut éventuellement mettre à jour régulièrement en fonction de l'évolution de la scène,
- la technique du ET logique entre cartes binaires de changements temporels, qui ne donne de bons résultats que si le mouvement est assez rapide pour qu'il n'y ait pas de recouvrement entre deux positions successives de l'objet, ce qui est assez restrictif.

C'est pourquoi on fait appel à une approche statistique probabiliste pour régulariser le problème qui est initialement mal posé (sous-déterminé). Le principe consiste à introduire une modélisation a priori du champ des étiquettes, à l'aide de la théorie des champs aléatoires de Markov (*Markov Random Fiels* ou MRF en anglais) [32]. On contraint alors la solution vers une configuration la plus probable du champ des étiquettes étant donné les observations et le modèle *a priori* [33]. On modélise classiquement le lien statistique entre observation et étiquette en introduisant une fonction Ψ d'attache aux données : $o(s) = \Psi(e_s) + g_s$, où g_s représente un bruit gaussien centré de variance σ^2. La fonction déterministe Ψ correspond à l'information pertinente et peut être définie de plusieurs façons [33, 34].

2.2 Approche markovienne - Régularisation statistique

2.2.1 Fonctions d'énergie

Adoptons les notations suivantes :
- S l'image à l'instant courant t,
- s un pixel (*site*) quelconque de S,
- η_s un voisinage spatio-temporel de s, par exemple celui défini par la Fig. 2.3,
- r n'importe quel voisin de s (spatial ou temporel),
- C l'ensemble des cliques binaires $c = (s, r)$ constituant les voisinages η_s,
- $O = \{O_s, s \in S\}$ le champ aléatoire des observations,
- $E = \{E_s, s \in S\}$ le champ aléatoire des étiquettes,
- $o = \{o_s, s \in S\}$ une réalisation particulière du champ d'observations O à l'instant t,
- $e = \{e_s, s \in S\}$ une réalisation particulière du champ d'étiquettes E à l'instant t,
- R l'ensemble des configurations possibles e du champ aléatoire E.

A chaque fois que ce sera nécessaire, on ajoutera un indice temporel pour préciser l'instant considéré : t, $t-1$, etc.

Une clique binaire est une paire de sites mutuellement voisins. Les interactions spatiales et tem-

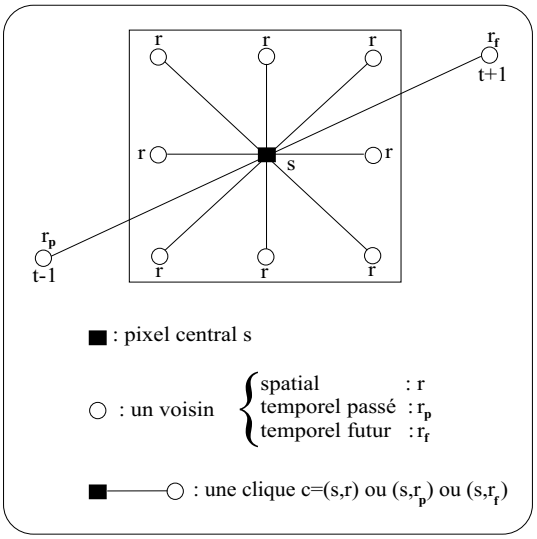

FIGURE 2.3 – Voisinage spatio-temporel et cliques binaires associées.

porelles entre étiquettes sont modélisées par un MRF, qui constitue le modèle a priori du champ des étiquettes. La propriété essentielle d'un MRF relativement à un voisinage est le caractère local des interactions : la probabilité d'avoir en un pixel s une étiquette e_s ne dépend que des étiquettes de ses voisins, et non pas de toute l'image.

1. $\forall e \in R, Pr[E = e] > 0$
2. $Pr\left[E_s = e_s \,/\, E_r = e_r, r \neq s, r \in S\right] = Pr\left[E_s = e_s \,/\, E_r = e_r, r \in \eta_s\right]$

Cette propriéte est cruciale car elle implique des calculs purement locaux et fortement parallélisables, d'où les possibilités de mise en œuvre temps réel [35], [36].

Un MRF étant équivalent à une distribution de Gibbs, on peut exprimer la probabilité a priori du champ d'étiquettes à l'aide d'une fonction d'énergie :

$$Pr[E = e] = \frac{1}{Z}\, exp(-U_m(e)) \tag{2.3}$$

où Z est une constante de normalisation (appelée *fonction de partition*) et U_m une fonction d'énergie associée au modèle a priori. Le champ d'étiquettes le plus probable relativement aux observations est obtenu par le critère du Maximum A Posteriori (MAP). A l'aide du théorème de Bayes et de l'équation (2.3), on montre que ce critère équivaut à la minimisation d'une fonction d'énergie totale U [37] :

$$\max_e Pr[E = e/O = o] \Longleftrightarrow \min_e U \quad \text{où} \quad U = U_m(e) + U_a(o, e) \tag{2.4}$$

$U_m(e)$ est un terme d'énergie qui assure la régularisation de la solution. Il s'exprime comme une somme de potentiels énergétiques élémentaires sur les cliques :

$$U_m(e) = \sum_{c \in C} V_c(e_s, e_r) \tag{2.5}$$

Ces potentiels élémentaires sont définis en fonction du problème à traiter. On peut par exemple prendre des potentiels à niveaux du type :

$$V_c(e_s, e_r) = \begin{cases} -\beta & \text{si } e_s = e_r \\ +\beta & \text{si } e_s \neq e_r \end{cases} \tag{2.6}$$

ou des potentiels quadratiques du type :

$$V_c(e_s, e_r) = \beta(e_s - e_r)^2 \tag{2.7}$$

où $\beta > 0$ prend une des trois valeurs β_s, β_p ou β_f selon la clique considérée (spatiale, passée ou future). Ces potentiels énergétiques favorisent un étiquetage homogène puisque des étiquettes semblables sur des pixels voisins induisent une moindre contribution énergétique, donc une configuration plus favorable. En pratique, on prend $\beta_f > \beta_p$, pour bien éliminer les zones d'écho du mouvement.

$U_a(o, e)$ est l'énergie d'adéquation, qui assure une bonne attache aux données. Elle traduit le lien entre observations et étiquettes et s'exprime à l'aide d'une fonction Ψ censée modéliser les observations :

$$U_a(o, e) = \frac{1}{2\sigma^2} \sum_{s \in S} [o_s - \Psi(e_s)]^2 \tag{2.8}$$

où σ^2 est la variance des observations et Ψ est une fonction d'attache aux observations définie en fonction du problème à traiter. On peut par exemple prendre une fonction simple du type :

$$\Psi(e_s) = \begin{cases} 0 & \text{si } e_s = b \\ \alpha > 0 & \text{sinon} \end{cases} \tag{2.9}$$

où α correspond alors à la valeur moyenne des observations non nulles.

2.2.2 Estimation des paramètres

Tous les paramètres qui interviennent dans la modélisation (α, β, σ) peuvent soit être déterminés de façon empirique après des tests sur des séquences typiques correspondant à l'application envisagée, soit être estimés automatiquement grâce à des algorithmes du type EM (*Expectation-Maximisation*) ou SEM (*Stochastic Expectation Maximisation*). Voir par exemple les travaux de Pieczynski [38] [39].

2.2.3 Algorithmes de relaxation

Pour calculer la configuration d'étiquettes donnant l'énergie minimale, différents algorithmes, dits de relaxation, peuvent être utilisés.

- Les algorithmes de relaxation stochastiques, du type recuit simulé, consistent à explorer l'espace des configurations possibles avec une loi de descente en température adéquate (i.e. suffisamment lente) qui assure en principe de converger vers le minimum global de la fonction d'énergie, mais au prix d'un coût de calcul prohibitif.
- C'est pourquoi on leur préfère souvent les algorithmes de relaxation déterministes, du type ICM (*Iterated Conditional Modes*) [40], beaucoup moins lourds en calculs mais qui ne garantissent pas de converger vers le mimimum global d'énergie. Ils risquent de rester piégés dans un minimum local, car ils n'autorisent aucune remontée en température pour sortir d'un minimum local. C'est pourquoi ce type d'algorithme requiert une bonne configuration initiale du champ d'étiquettes (Fig. 2.4).

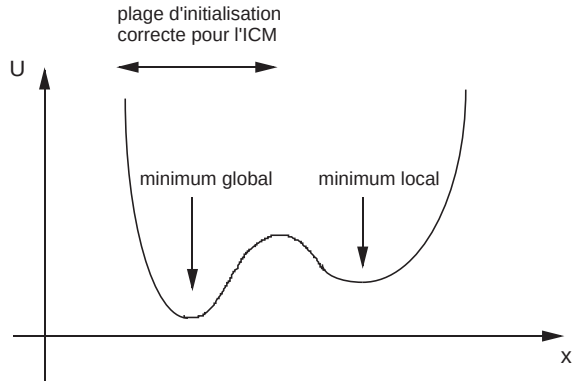

FIGURE 2.4 – Influence de l'initialisation sur le résultat de l'ICM : évolution de la fonction d'énergie U en fonction de la configuration X du champ d'étiquettes.

Ces algorithmes déterministes sont récursifs et itératifs : on visite chaque pixel de l'image et on calcule, pour chaque étiquette qu'on peut lui attribuer, la contribution énergétique sur son voisinage. On retient l'étiquette qui donne l'énergie minimale, puis on passe au pixel suivant et ainsi de suite sur toute l'image. On réitère la visite des sites de l'image jusqu'à la convergence de la fonction d'énergie totale U vers une valeur qui n'évolue plus, ou très peu, au cours des itérations. Différentes politiques de visite de sites sont envisageables :

- visite séquentielle classique (*line scanning*) (ligne par ligne de gauche à droite et image par image de haut en bas),
- visite aléatoire,
- visite chaînée (ligne par ligne alternativement de gauche à droite puis de droite à gauche et image par image alternativement de haut en bas puis de bas en haut) pour bénéficier au mieux des derniers voisinages mis à jour.

Différentes politiques de mise à jour des sites sont aussi envisageables, le choix dépendant notamment du type de matériel utilisé pour la mise en œuvre :

- pixel-récursif : on met à jour chaque pixel au fur et à mesure de la visite,
- ligne-récursif : on attend d'avoir visité tous les pixels d'une ligne avant de mettre à jour leurs étiquettes,
- image-récursif : on attend d'avoir visité toute une image avant de faire une mise à jour simultanée de toutes les étiquettes de l'image.

De même, différents critères d'arrêt des itérations peuvent être utilisés :

- nombre de pixels instables inférieur à un seuil prédéterminé,
- variation relative d'énergie totale inférieure à un seuil prédéterminé,
- nombre fixe d'itérations (en pratique un faible nombre d'itérations, typiquement 5 à 10 itérations par image, est suffisant). Ce critère est évidemment le moins coûteux pour une mise en œuvre matérielle.

2.2.4 Synoptique d'un algorithme de détection de mouvement

Le synoptique d'un algorithme typique est donné Fig. 2.5. Sur ce schéma-bloc, la notation $\hat{e}$ représente un champ d'étiquettes initial grossièrement estimé par simple binarisation du champ correspondant d'observations. Pour réaliser la binarisation, on peut soit faire un simple seuillage, soit utiliser des techniques plus robustes de maximum de vraisemblance, avec modèle de luminance (constant, linéaire ou quadratique) sur un voisinage des pixels [31]. Etant donné le voisinage spatio-temporel utilisé pour détecter le mouvement, voisinage qui inclut un voisin futur et un voisin passé pour chaque pixel (Fig. 2.3), le résultat de traitement est obtenu avec un retard d'une image par rapport à l'acquisition, puisqu'il faut disposer de l'image en $t+1$ pour estimer le champ d'étiquettes en t.

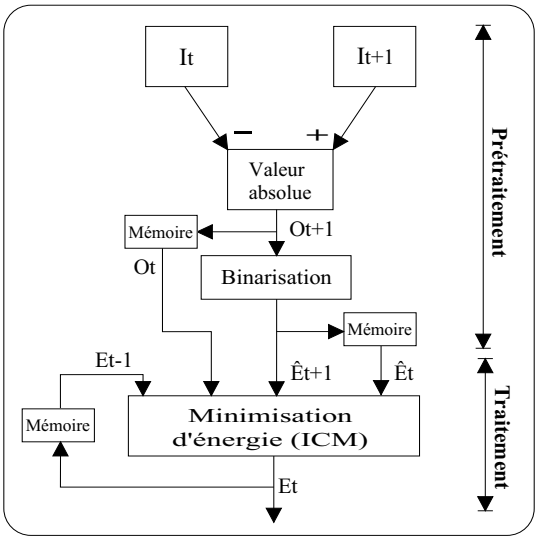

FIGURE 2.5 – Synoptique de l'algorithme de détection de mouvement.

2.3 Multirésolution spatio-temporelle

Pour améliorer les performances de l'algorithme dans le cas du mouvement d'objets très lents (mouvement sub-pixel), ou de gros objets peu texturés, on peut envisager d'utiliser une technique du type multirésolution. On présente ici quelques résultats obtenus avec une technique de multirésolution spatio-temporelle qui consiste à filtrer passe-bas et sous-échantillonner la séquence d'images à la fois en espace et en temps selon le schéma de la Fig. 2.6, où est aussi représenté le noyau du filtre 3-D appliqué à la séquence.

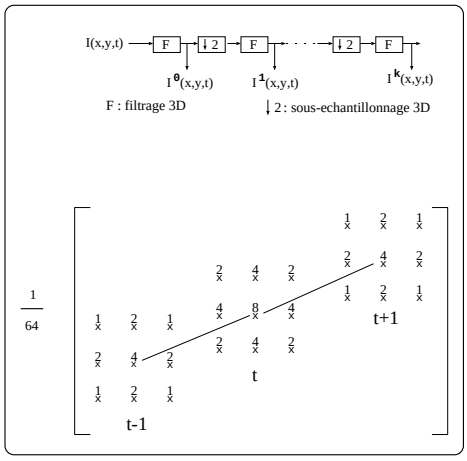

FIGURE 2.6 – Principe de construction de la pyramide et noyau du filtre spatio-temporel 3-D.

Un exemple de pyramide spatio-temporelle est donné Fig. 2.7. Le nombre de niveaux dans la pyramide est fonction notamment de l'amplitude des mouvements présents dans la scène.

Le filtrage spatial permet de renforcer les observations dans le cas de gros objets uniformes (Fig. 2.8).

Quant au filtrage temporel, il permet d'intégrer l'information de mouvement de plusieurs images successives, ce qui est indispensable pour détecter des objets au mouvement très lent (Fig. 2.9).

2.4 Voisinage 3-D spatio-temporel

On peut envisager une structure de voisinage 3-D comme présenté Fig. 2.10 [41]. Si on note $\delta_x, \delta_y, \delta_t$ les coordonnées du vecteur représentatif de chaque clique dans l'espace 3-D (x, y, t) centré en le pixel courant s, on définit :

FIGURE 2.7 – Pyramide spatio-temporelle à trois niveaux sur la séquence *Trevor* (de haut en bas, $k = 0, 1, 2$).

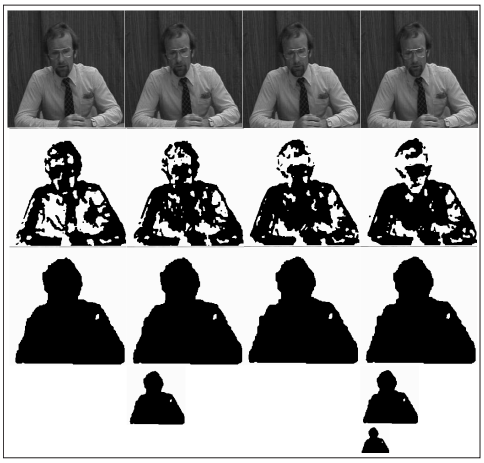

FIGURE 2.8 – Détection de mouvement multirésolution : de haut en bas : 1) séquence *Trevor* ; 2) masques monorésolution ; 3) masques multirésolution (3 niveaux d'analyse $k = 0, 1, 2$).

- 4 cliques spatiales horizontales et verticales (δ_x ou $\delta_y = \pm 1$, $\delta_t = 0$) ;
- 4 cliques spatiales diagonales (δ_x et $\delta_y = \pm 1$, $\delta_t = 0$) ;
- 2 cliques temporelles (δ_x et $\delta_y = 0$, $\delta_t = \pm 1$) ;
- 8 cliques spatio-temporelles horizontales et verticales (δ_x ou $\delta_y = \pm 1$, $\delta_t = \pm 1$) ;
- 8 cliques spatio-temporelles diagonales (δ_x et $\delta_y = \pm 1$, $\delta_t = \pm 1$).

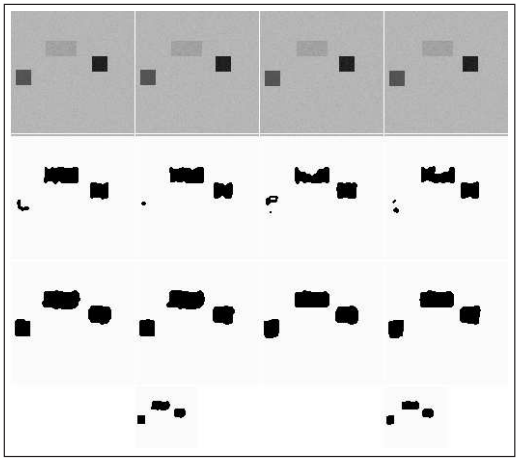

FIGURE 2.9 – Détection de mouvement sous-pixel : de haut en bas : 1) séquence synthétique avec 3 objets mobiles dont l'un a un mouvement sub-pixel ; 2) masques monorésolution ; 3) masques multirésolution (2 niveaux d'analyse $k = 0, 1$).

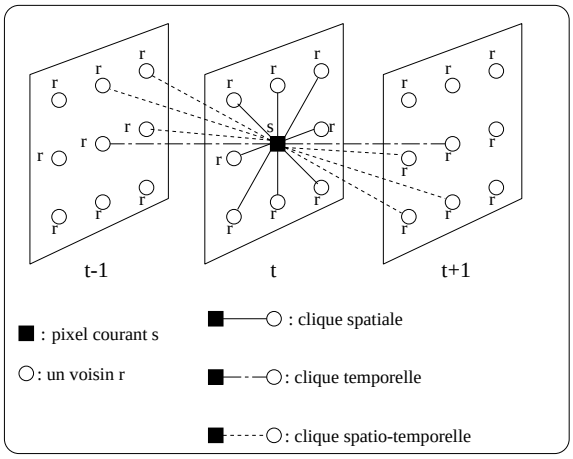

FIGURE 2.10 – Voisinage spatio-temporel 3-D, cliques binaires associées (pour des raisons de clarté, toutes les cliques possibles n'ont pas été représentées).

Il faut alors modéliser les interactions spatio-temporelles sur toutes les cliques de ce voisinage 3-D par des potentiels $\beta(s,r)$ qui sont fonction du type de clique $c=(s,r)$ considérée :

$$\beta(s,r) = \frac{1}{d^2(s,r)\ (\frac{\delta_x(s,r)^2}{\beta_s} + \frac{\delta_y(s,r)^2}{\beta_s} + \frac{\delta_t(s,r)^2}{\beta_t})} \tag{2.10}$$

où $d(s,r) = \sqrt{\delta_x^2 + \delta_y^2 + \delta_t^2}$ est la distance euclidienne entre le pixel courant s et le voisin considéré r. Cette formule donne :

- $\beta(s,r) = \beta_s$ pour les cliques spatiales horizontales ou verticales $(d(s,r)=1)$;
- $\beta(s,r) = \frac{\beta_s}{4}$ pour les cliques spatiales diagonales $(d(s,r)=\sqrt{2})$;
- $\beta(s,r) = \beta_t$ pour les cliques temporelles $(d(s,r)=1)$;
- $\beta(s,r) = \frac{\beta_s\beta_t}{2(\beta_s+\beta_t)}$ pour les cliques spatio-temporelles horizontales ou verticales $(d(s,r)=\sqrt{2})$;
- $\beta(s,r) = \frac{\beta_s\beta_t}{3(\beta_s+2\beta_t)}$ pour les cliques spatio-temporelles diagonales $(d(s,r)=\sqrt{3})$.

Notons que cette définition de $\beta(s,r)$ ne fait intervenir que deux paramètres β_s et β_t.

Cette variante de l'algorithme permet d'améliorer les performances dans le cas de séquences bruitées comme illustré sur la Fig. 2.11.

2.5 Mises en œuvre matérielles

Un algorithme de détection de mouvement se décompose typiquement en deux étapes principales (cf. Fig. 2.5) :

- un prétraitement qui calcule les observations et les champs initiaux d'étiquettes,
- un traitement proprement dit qui calcule le minimum d'énergie sur les voisinages spatio-temporels.

Le prétraitement qui consiste simplement en des calculs de différences d'images, de valeurs absolues et des binarisations par seuillage ne pose pas de problème particulier d'implantation temps réel. Par contre, la minimisation d'énergie est un processus itératif gourmand en calculs et en mémoire et doit donc faire l'objet d'une mise en œuvre sur une architecture adaptée. Une évaluation grossière montre en effet que la relaxation markovienne compte pour 90% de la charge totale de calcul. On a alors affaire à un problème d'adéquation algorithme-achitecture. Plusieurs mises en œuvre sont envisageables [42] :

- implantation logicielle en C : problème d'optimisation des pointeurs et des types (integer ou float...) pour minimiser les transferts et les calculs (utilisation de LUT ou *Look-Up-Table*)
- une première solution matérielle, qui prend essentiellement en compte le caractère parallèle des calculs identiques effectués en chaque pixel, consiste à implanter l'algorithme sur une machine parallèle SIMD ou MIMD. L'inconvénient principal de ce type de mise en œuvre est l'encombrement, le coût et les transferts de données.
- une seconde solution, qui prend essentiellement en compte le caractère local des calculs sur un petit voisinage spatio-temporel, consiste à s'inspirer du fonctionnement des réseaux de neurones

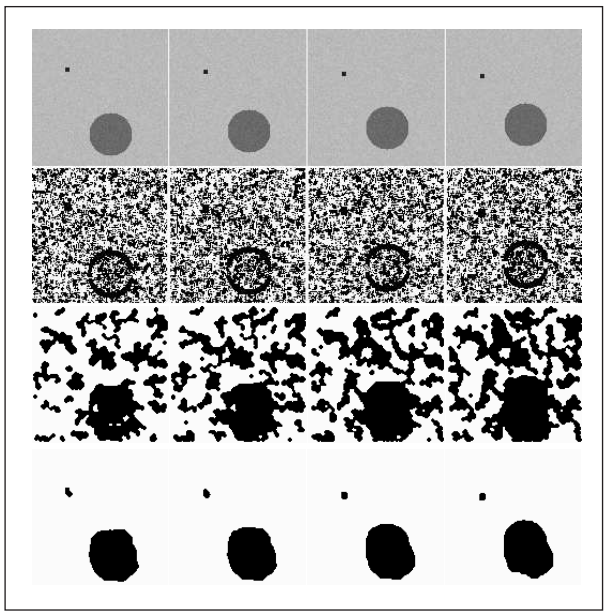

FIGURE 2.11 – Comparaison des deux algorithmes : de haut en bas : 1) séquence synthétique bruitée ; 2) initialisation binaire ; 3) masques avec l'algorithme spatial ; 4) masques avec l'algorithme spatio-temporel.

pour implanter l'algorithme sur un circuit résistif analogique en technologie VLSI. Il faut alors développer une analogie électrique du modèle markovien [43]. Inconvénients : coût de développement. Avantages : taille réduite du circuit.

– une troisième solution alternative, développée par certains laboratoires de recherche (comme le LIS-Grenoble [44, 45]), consiste à implanter l'algorithme sur une carte au format PC à base de DSP et de circuits logiques programmables FPGA (Fig. 2.12). Avantages : coût réduit, encombrement réduit. Inconvénient : manque de flexibilité.

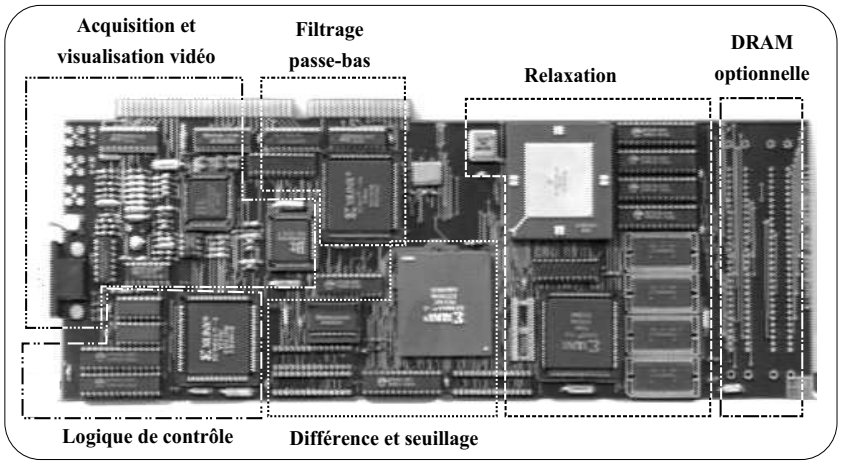

FIGURE 2.12 – Photo d'une carte d'acquisition et traitement vidéo.

2.6 Exemples d'applications

2.6.1 Télésurveillance

Une application typique de détection de mouvement est le contrôle du trafic routier [46] ou la télésurveillance à l'entrée d'un site grâce à une caméra "intelligente". La Fig. 2.13 présente un exemple de résultat de détection automatique du mouvement d'un piéton sur un trottoir, obtenu grâce à la carte à DSP décrite ci-dessus. Notons que l'algorithme détecte également l'ombre du piéton sur le capot de

la voiture en stationnement (il ne s'agit pas d'une fausse détection).

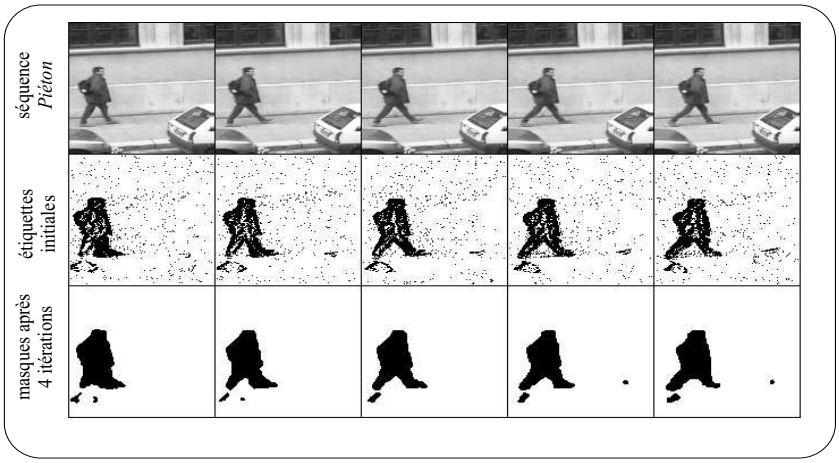

FIGURE 2.13 – Séquence acquise à la cadence de 25 images/seconde : le mouvement entre deux images est peu important ; la convergence est atteinte après 4 itérations.

2.6.2 Analyse du mouvement des lèvres d'un locuteur

Il est bien connu que l'homme s'aide d'informations visuelles pour améliorer sa reconnaissance de la parole, notamment dans un environnement bruité. En complément du signal auditif, la vision du mouvement des lèvres du locuteur est donc une donnée précieuse, qui peut être exploitée pour réaliser un système automatique de reconnaissance de la parole, ou de synthèse de visages parlants. Dans ce but, on peut développer un algorithme de détection automatique du mouvement des lèvres d'un locuteur [47]. Un projet qui s'inscrit dans le cadre de la compression audio-visuelle consiste à équiper le locuteur d'un casque léger qui comporte, en plus du microphone, une micro-caméra couleur solidaire du crâne et dirigée vers la zone des lèvres. On reste bien dans le cas d'une caméra fixe par rapport au locuteur et on peut donc appliquer les principes de détection exposés précédemment, en adaptant les observations et le modèle à l'application envisagée.

Les grandes lignes du traitement sont les suivantes : on acquière une séquence couleur RVB (rouge-vert-bleu) du mouvement de la bouche, la région d'intérêt allant de la base du nez au menton. On transforme l'espace RVB en l'espace HIP (teinte ou *hue* en anglais, luminance ou intensité, pureté), également dénommé espace TLP (cf. Eq. 1.73), qui s'avère mieux adapté au problème.

En effet :
- la teinte rouge, qui est prédominante sur les lèvres, est une observation spatiale pertinente,
- la pureté permet de s'affranchir des zones d'ombre (car elle est voisine de zéro dans les zones d'ombre),
- enfin, les variations temporelles de luminance donnent l'information de mouvement.

On utilise deux types d'information de bas niveau :
- une information spatiale sur la teinte rouge (*hue*) : $h(s)$,
- une information temporelle sur la différence d'images (*frame difference*) : $fd(s)$.

$$fd(s) = I_t(s) - I_{t-1}(s) \tag{2.11}$$

$$h(s) = \left[256 - \left(\frac{H(s) - H_m}{\sigma}\right)^2\right] \times 1_{P(s)>\delta} \times 1_{|H(s)-H_m|\leq 16.\sigma} \tag{2.12}$$

H_m représente la valeur moyenne de la teinte des lèvres (determinée au préalable sur la première image), σ est l'écart-type de la teinte des lèvres (valeur empirique, typ. $4 \leq \sigma \leq 9$) et δ est un seuil appliqué sur la pureté (typ. $50 \leq \delta \leq 100$). La notation $1_{condition}$ dénote une fonction binaire qui vaut 1 si la condition est vraie, 0 sinon.

En pratique, l'algorithme utilise trois observations pour chaque site : $h_{t-1}(s)$, $h_t(s)$ et $fd(s)$. Les Fig. 2.14 et 2.15 illustrent ces champs d'observation sur une séquence typique.

FIGURE 2.14 – *De haut en bas* : séquence des luminances ; séquence des observations spatiales h_t à 5 instants t (régions à dominante rouge en blanc, $\delta = 0$).

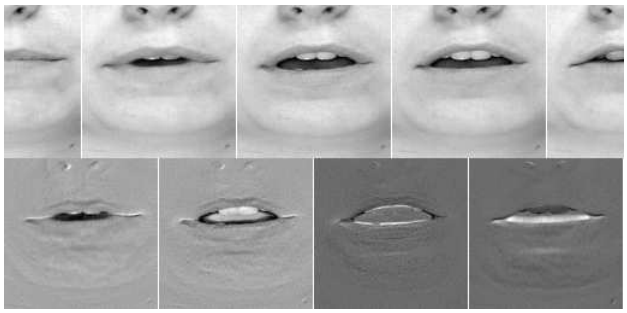

FIGURE 2.15 – *De haut en bas* : même séquence de luminances ; séquence des observations temporelles fd (valeurs positives en blanc, négatives en noir, nulles en gris).

La combinaison de ces trois observations permet de définir un jeu de douze étiquettes (Table 2.1). L'étiquette c_1 par exemple correspond à la présence de la teinte rouge des lèvres en t, à l'absence de teinte rouge en $t-1$, avec une observation de mouvement détecté positive. On définit alors un

TABLE 2.1 – Observations, codage bas niveau et les 12 étiquettes initiales correspondantes (typ. $\theta = 10$, $\gamma = 100$).

observations			détection initiale en s		codage			étiquettes
$h_t(s)$	$h_{t-1}(s)$	$fd(s)$	teinte	mvt	$r_t(s)$	$r_{t-1}(s)$	$m(s)$	$e(s)$
$< \gamma$	$< \gamma$	$\lvert . \rvert < \theta$	$\emptyset$	$\emptyset$	0	0	0	a_0
		$> \theta$		$+$			1	a_1
		$< -\theta$		$-$			2	a_2
	$> \gamma$	$\lvert . \rvert < \theta$	$t-1$	$\emptyset$		1	0	b_0
		$> \theta$		$+$			1	b_1
		$< -\theta$		$-$			2	b_2
$> \gamma$	$< \gamma$	$\lvert . \rvert < \theta$	t	$\emptyset$	1	0	0	c_0
		$> \theta$		$+$			1	c_1
		$< -\theta$		$-$			2	c_2
	$> \gamma$	$\lvert . \rvert < \theta$	t & $t-1$	$\emptyset$		1	0	d_0
		$> \theta$		$+$			1	d_1
		$< -\theta$		$-$			2	d_2

modèle énergétique du problème à traiter [47], et on minimise une fonction d'énergie totale faite de cinq termes : trois termes d'attache aux trois observations, un terme d'interaction spatiale et un terme d'interaction temporelle :

$$U = \sum_{s \in S} \left[\lambda [U_{h_t}(s) + U_{h_{t-1}}(s)] + U_{fd}(s) + U_{sp}(s) + U_{tp}(s) \right] \tag{2.13}$$

où λ est un coefficient de pondération sur les énergies d'attache aux observations spatiales. La mini-

misation de cette fonction d'énergie permet finalement d'extraire de la région d'intérêt les lèvres du locuteur, comme illustré sur la Fig. 2.16.

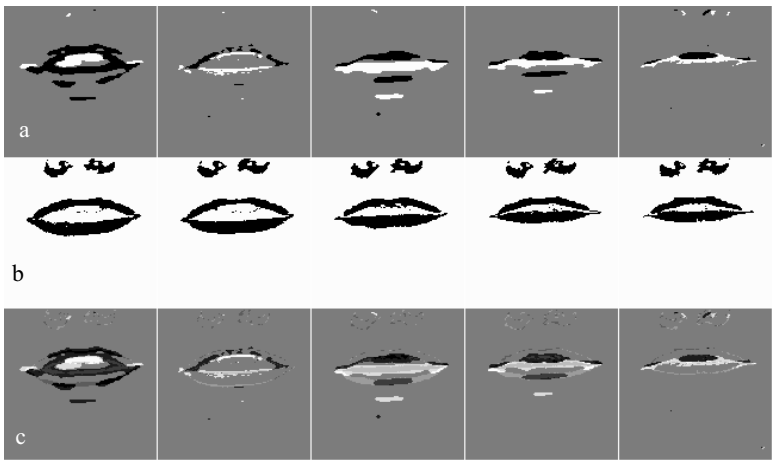

FIGURE 2.16 – Champs après relaxation. *a) Champs de mouvement M ; b) Champs de teinte rouge R_t avec $\delta = 100$; c) Champs E_t des 12 étiquettes représentées en niveaux de gris.*

Améliorations de la robustesse

- Pour s'affranchir de la contrainte forte du casque, on peut mettre en œuvre une boucle de rétrocation (inspirée de l'automatique) entre modèle 3D et contours actifs [48].
- Pour améliorer la classification de teinte, on peut travaillet dans l'espace couleur LUX [23].

2.6.3 Conclusion

On a passé en revue un certain nombre de techniques applicables en détection de mouvement 2-D dans les séquences d'images. Mais les outils de traitement présentés ici sont de portée très générale et applicables dans tout problème mal posé portant sur des signaux à deux ou trois dimensions, quelle que soit la nature de ces dimensions, et où il est nécessaire de régulariser la solution en introduisant des contraintes ou des connaissances *a priori*.

2.7 Estimation de mouvement

L'estimation de mouvement consiste à calculer le déplacement de chaque pixel de l'image ou des régions en mouvement. On obtient alors un champ dense ou épars de vecteurs-vitesse, qu'on appelle flux optique. On distingue trois classes de méthodes d'estimation :

- les méthodes différentielles correspondent à une approche physique (optique) ;
- les méthodes fréquentielles sont inspirées de la biologie ;
- les méthodes de mise en correspondance sont une approche purement informatique du problème.

Une bonne revue de ces méthodes est présentée par Barron et al. [49], avec un site pour télécharger le code source :

`ftp.csd.uwo.ca/pub/vision.`

Notons que des approches moins classiques permettent également d'estimer le mouvement à partir de la couleur [50].

2.7.1 Méthodes différentielles

On fait les hypothèses suivantes :

- déplacements petits par rapport à la taille des objets en mouvement (typ. 1 pixel/trame)
- invariance de l'intensité lumineuse des objets au cours du temps.

La deuxième hypothèse s'exprime classiquement avec la DFD (*Displaced Frame Difference*) :

$$DFD(p,d) = I(x+dx, y+dy, t+dt) - I(x,y,t) = 0 \tag{2.14}$$

En pratique, on cherche à minimiser la DFD. Pour cela, on considère le développement en série de Taylor de la DFD au premier ordre, qui aboutit à l'équation de contrainte du mouvement exprimant la relation entre les gradients spatiaux et temporels de l'intensité lumineuse :

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{2.15}$$

Comme on a une seule équation pour deux inconnues $u = \frac{dx}{dt}$ et $v = \frac{dy}{dt}$, le problème est sous-déterminé. Ceci constitue le fameux problème d'ouverture (*aperture problem* en anglais) : si l'on observe la scène à travers une toute petite ouverture, on ne perçoit que la composante orthogonale au contour (Fig. 2.17).

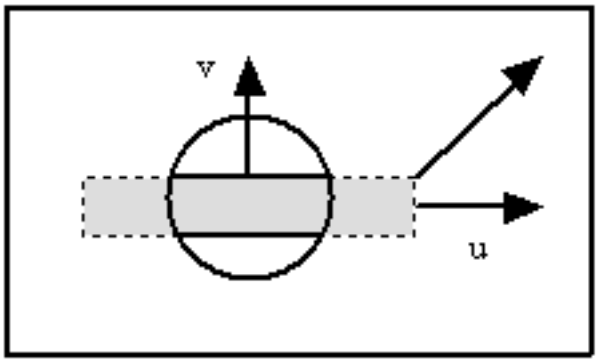

FIGURE 2.17 – Le problème d'ouverture

Il faut donc introduire une containte supplémentaire, dite contrainte de lissage, pour lever l'indétermination. On ajoute donc une hypoythèse d'homogénéité du champ de vitesses dans un voisinage (domaine D).

Horn et Schunck adoptent un terme de lissage portant sur la somme des carrés des modules des gradients des composantes de vitesse, conduisant à la minimisation de la formule [51] :

$$\iint_D \left(\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}\right)^2 + \alpha^2\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right) dx\,dy \tag{2.16}$$

où α est un coefficient de pondération du terme de lissage.

Un algorithme itératif permet de minimiser l'intégrale :

$$u^{k+1} = \overline{u}^k - \frac{I_x[I_x\overline{u}^k + I_y\overline{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{2.17}$$

$$v^{k+1} = \overline{v}^k - \frac{I_y[I_x\overline{u}^k + I_y\overline{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{2.18}$$

où k représente l'indice d'itération (conditions initiales u^0 et v^0 nulles), I_x, I_y, I_t les composantes du gradient selon x, y, t et $\overline{u}, \overline{v}$ les valeurs moyennes calculées sur un voisinage 3×3 pondéré (Fig. 2.18b).

Nagel propose une contrainte de lissage orientée qui tient compte de l'orientation du gradient, pour éviter de lisser sur les frontières des objets en mouvement [52, 53]. Le terme de lissage est plus compliqué car il repose sur les dérivées secondes :

$$\frac{\alpha^2}{I_x^2 + I_y^2 + 2\delta}\left[(u_xI_y - u_yI_x)^2 + (v_xI_y - v_yI_x)^2 + \delta(u_x^2 + u_y^2 + v_x^2 + v_y^2)\right] \tag{2.19}$$

où δ est un paramètre de conditionnement (qui empêche l'annulation du dénominateur). Un algorithme itératif permet là aussi d'obtenir la solution correspondant au minimum de la formule.

Les méthodes différentielles reposant sur des calculs de dérivées partielles, la qualité de l'estimation est fortement dépendante des formules choisies pour ces calculs. Comme un calcul de dérivée est sensible au bruit, un préfiltrage passe-bas spatio-temporel est souvent nécessaire. Horn et Schunck utilisent un masque $[1 \quad -1]$ avec moyennage dans le cube de 8 pixels $x, x+1, y, y+1, t, t+1$ (Fig. 2.18a). Barron

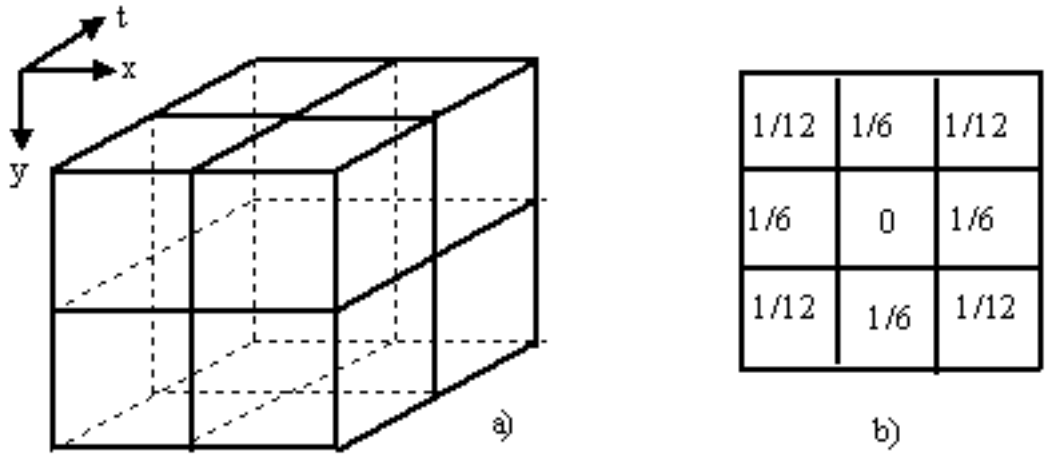

FIGURE 2.18 – a) Voisinage cubique b) Masque de calcul des moyennes

et al. proposent un masque $\frac{1}{12}[-1 \quad 8 \quad 0 \quad -8 \quad 1]$ avec un préfiltrage spatio-temporel gaussien.

2.7.2 Méthodes fréquentielles

Basées sur l'énergie

Considérons le cas d'un mouvement mono-dimensionnel. La fréquence spatiale ω_x d'une sinusoïde en mouvement s'exprime en cycles/pixel alors que le fréquence temporelle ω_t s'exprime en cycles/unité de temps, c'est-à-dire en cycles/trame, puisque l'unité de temps en vidéo est l'intervalle temporel séparant deux trames consécutives de la séquence. La vélocité ou vitesse, qui est la distance sur le temps, s'exprime donc en pixels/trame et vaut $u = \frac{\omega_t}{\omega_x}$ d'où l'on a la relation : $\omega_t = u\omega_x$.

De façon analogue, une texture 2-D translatant dans une image occupe un plan dans le domaine des fréquences spatio-temporelles :

$$u\omega_x + v\omega_y = \omega_t \tag{2.20}$$

C'est l'équation de contrainte du mouvement dans le domaine fréquentiel. En estimant l'orientation de ce plan, on peut en déduire les composantes u, v de la vitesse. La technique consiste donc à paver le domaine fréquentiel avec une batterie de filtres passe-bande (Fig. 2.19). Là où il y a de l'énergie, la réponse du filtre sera forte, ce qui permet de localiser le plan et d'estimer son orientation, d'où les vitesses.

Heeger propose d'utiliser des filtres de Gabor 3-D [54] :

$$g(x,y,t) = \frac{1}{\sqrt{2}\pi^{3/2}\sigma_x\sigma_y\sigma_t} \exp\left[-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2} + \frac{t^2}{2\sigma_t^2}\right)\right] \times \sin\left(2\pi\omega_{x_0}x + 2\pi\omega_{y_0}y + 2\pi\omega_{t_0}t\right) \tag{2.21}$$

où $\omega_{x_0}, \omega_{y_0}, \omega_{t_0}$ est la fréquence centrale du filtre et $\sigma_x, \sigma_y, \sigma_t$ l'extension de la fenêtre gaussienne spatio-temporelle.

L'inconvénient de cette technique est son coût de calculs, lié au nombre de filtres nécessaires (12) et à la taille des masques de convolution (taille 23 pour les filtres spatiaux, et taille 7 en temproel). Mais on peut utiliser la séparabilité des filtres 3-D pour réduire le coût des convolutions.

Basées sur la phase

La règle de translation de la TF s'exprime par :

$$TF[f(x)] = F(\nu) \Rightarrow TF[f(x-x_0)] = F(\nu)\exp\left(-i2\pi\nu x_0\right) \tag{2.22}$$

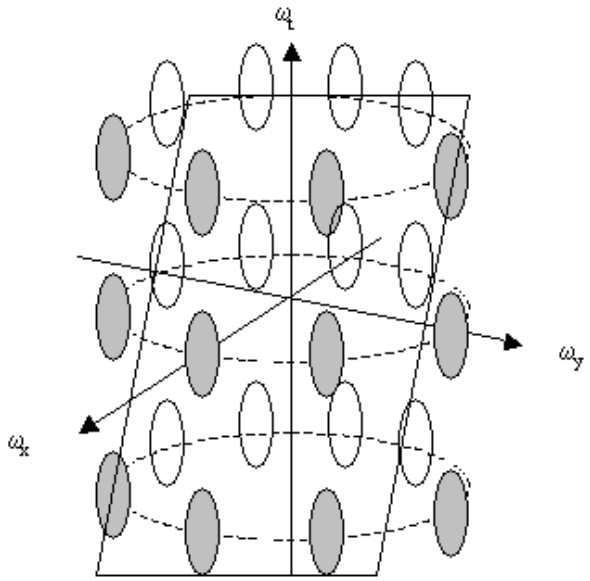

FIGURE 2.19 – Plan fréquentiel

Cette règle peut être exploitée pour estimer un mouvement de translation 1-D dans une image à partir de l'étude de la phase. Hélas, dans le cas 2-D, la mesure du déphasage donne une information sur la somme du déplacement en x et en y. Pour désimbriquer l'information sur la translation horizontale et verticale, on peut utiliser une approche basée sur les nombres hypercomplexes (espace des quaternions)

2.7.3 Méthodes de Mise en correspondance

La mise en correspondance de régions consiste à rechercher entre deux images successives la meilleure correspondance en maximisant une mesure de similarité comme l'intercorrélation, ou en minimisant une mesure de distance comme la SSD (somme des différences au carré) entre deux blocs :

$$SSD = \sum_{j=-n}^{n} \sum_{i=-n}^{n} W(i,j) \times [I_1(x+i, y+j) - I_2(x+i+dx, y+j+dy)]^2 \tag{2.23}$$

où W est une fonction de pondération et $d = (d_x, d_y)$ le déplacement estimé (Fig. 2.20).

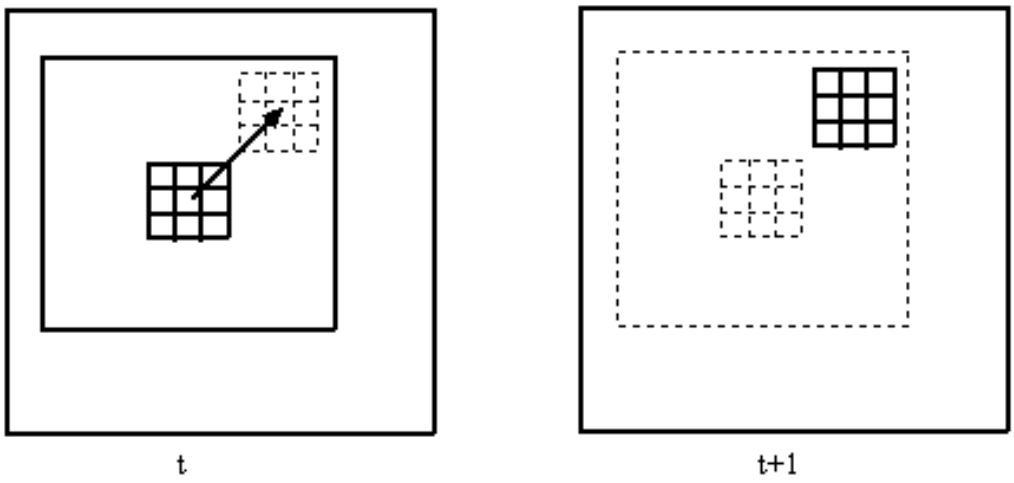

FIGURE 2.20 – Mise en correspondance d'un bloc 3×3

A la place de la SSD, on peut utiliser la SAD (somme des différences absolues) moins coûteuse en calcul.

On associe souvent ce type de technique avec une approche pyramidale *coarse-to-fine*, car une recherche exhaustive à pleine résolution est très coûteuse. Anandan [55] utilise par exemple une pyramide laplacienne avec 2 ou 3 niveaux, ce qui permet le calcul de grands déplacements en commençant par une mise en correspondance au niveau le plus grossier de la pyramide. Cela permet aussi de rehausser les structures importantes (contours) dans l'image.

La mise en correspondance de bloc est très utilisée pour le codage vidéo (compression MPEG). L'artéfact classique est celui des effets de blocs perceptibles à la restitution. Ceci est dû aux hypothèses limitatives de cette approche : le découpage en blocs et leur taille (typ. 8×8 ou 16×16) est arbitraire, les mouvements complexes (autres qu'une simple translation), et les grands déplacements sont mal pris en compte. C'est pourquoi des variantes (approches hiérarchiques ou par transformées spatiales) sont proposées dans la littérature [56].

2.7.4 Modèles paramétriques de mouvement

Au lieu d'une contrainte de lissage, on peut utiliser un modèle explicite de mouvement. On peut baser l'estimation de mouvement sur une modélisation polynomiale du déplacement. Les paramètres du modèle sont déterminés par une technique de type moindres carrés ou moindres carrés robustes. Il existe une hiérarchie de modèles de complexité croissante :

- modèle translationnel : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 \\ a_4 \end{bmatrix}$
- translation + rotation : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 + a_3 y \\ a_4 - a_3 x \end{bmatrix}$
- translation + rotation + divergence : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 - a_3 x + a_2 y \end{bmatrix}$
- modèle affine complet : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{bmatrix}$
- modèle quadratique : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y + a_7 x^2 + a_8 xy \\ a_4 + a_5 x + a_6 y + a_7 xy + a_8 y^2 \end{bmatrix}$
- modèle panoramique : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a_1 + a_1 x^2 + a_4 xy \\ a_4 + a_1 xy + a_4 y^2 \end{bmatrix}$

Le modèle affine est un bon compromis entre représentativité et complexité. On recherche le vecteur des paramètres $\Theta = (a_1, a_2, a_3, a_4, a_5, a_6)$ qui minimise un critère défini par une fonction de coût :

$$\Theta = \arg\min_{\theta} \sum \rho(r) \tag{2.24}$$

où r est le résiduel (typ. $r = DFD$) et où ρ est un estimateur qui peut être quadratique ($\rho(r) = r^2$, auquel cas on retombe sur une estimation classique aux moindres carrés), ou un estimateur dit robuste (du type M-estimateur). En effet, une bonne fonction ρ doit être définie positive mais ne pas croître indéfiniment lorsque $r \to \infty$, pour limiter l'influence des points aberrants. Un exemple classique est l'estimateur "biweight" de Tukey :

$$\rho(r) = \begin{cases} \frac{r^6}{6} - \frac{C^2 r^4}{2} + \frac{C^4 r^2}{2} \text{ si : } |r| < C \\ \frac{C^6}{6} \text{ sinon} \end{cases} \tag{2.25}$$

2.8 Compensation de mouvement

2.8.1 Introduction

Les méthodes de compression de séquences d'images pour la transmission à bas débit reposent sur une prédiction par compensation de mouvement. Nous décrivons ici l'approche classique d'estimation de mouvement proposée par Netravali [57] et améliorée par Walker-Rao [58] et Biemond [59].

L'algorithme d'estimation, basé sur une méthode différentielle, est pixel-récursif et itératif. L'hypothèse de travail étant l'invariance de la luminance, le champ de déplacement est estimé en minimisant la DFD (*Displaced Frame Difference*) en chaque pixel ou sur un voisinage causal constitué de N pixels incluant le pixel courant, selon une technique du type descente de gradient ou estimateur de Wiener. Ceci donne une image prédite à partir de l'image passée et du champ de déplacement estimé.

2.8.2 Estimation de mouvement pel-récursive

Les techniques pel-récursives d'estimation de mouvement développées pour le codage ont pour objectif la reconstruction de l'image. En ce sens, elles ne visent pas forcément la meilleure estimation du déplacement réel des objets mais la minimisation d'une erreur de prédiction. La méthode introduite par Netravali *et al.* [57], pour le codage de séquences télévisuelles par compensation de mouvement, consiste à estimer en chaque point p de l'image le déplacement $\hat{d}$ qui minimise le critère de prédiction donné par l'équation (2.26) où la DFD est la différence d'image déplacée définie par l'équation (2.27) :

$$\hat{d} = \arg\min_{d}\{DFD^2(p,d)\} \tag{2.26}$$

$$\text{où} \qquad DFD(p,d) = I(p,t) - I(p-d,t-1) \tag{2.27}$$

La solution proposée par Netravali utilise la technique numérique et itérative de descente de gradient :

$$\hat{d}^i = \hat{d}^{i-1} - T_c \tag{2.28}$$

$$\text{où} \qquad T_c = \varepsilon.DFD(p,\hat{d}^{i-1}).\nabla I(p-\hat{d}^{i-1},t-1) \tag{2.29}$$

T_c est le terme de correction appliqué pour calculer $\hat{d}^i$, déplacement estimé à la ième itération, ε est un gain constant positif réglant la vitesse de convergence et $\nabla I(p-\hat{d}^{i-1},t-1)$ est le gradient spatial. Pour améliorer les performances de l'estimateur, Walker *et al.* proposent d'utiliser un gain adaptatif [58] :

$$\varepsilon = \frac{1}{2.|\nabla I(p-\hat{d}^{i-1},t-1)|^2} \tag{2.30}$$

A chaque itération, plusieurs tests sont effectués :
- test de nullité du gradient, auquel cas on ne fait pas d'itération,
- test de limitation du terme de correction T_c (bornes supérieure et inférieure),
- test d'arrêt par compensation (seuil minimum atteint sur la DFD),
- test d'arrêt par dépassement du nombre maximal d'itérations.

Pour améliorer la convergence de l'algorithme, on peut choisir pour valeur initiale $\hat{d}^0$ du déplacement en le pixel courant p, la meilleure valeur parmi celles estimées précédemment dans un voisinage causal du pixel, c'est-à-dire celle qui minimise la DFD en le pixel p. Un voisinage couramment utilisé est celui de la figure 2.21(b).

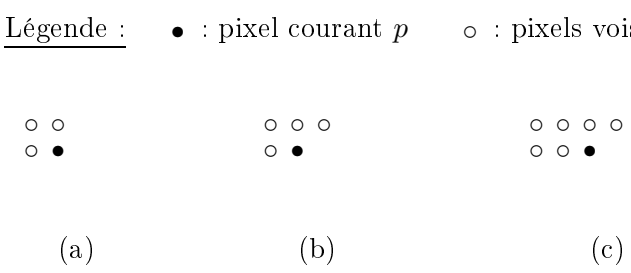

FIGURE 2.21 – (a,b,c) : Différents voisinages spatiaux causaux

Les pixels dans l'image sont parcourus de gauche à droite et de haut en bas (balayage télévision). Pour reconstruire l'image à l'instant t, on ajoute à chaque pixel de l'image à l'instant $t-1$ le vecteur déplacement estimé lui correspondant. Le critère classiquement utilisé pour évaluer la qualité de la reconstruction est le rapport signal à bruit crête (*Peak SNR*) défini par :

$$PSNR_{dB} = 10.log_{10}\left(\frac{(I_{max}-I_{min})^2}{EQM}\right) \tag{2.31}$$

$$\text{avec} \qquad EQM = \frac{1}{LC}\sum_{p}\left(I(p,t)-\hat{I}(p,t)\right)^2 \tag{2.32}$$

où $I_{max} - I_{min} = 255$ pour des images codées sur 8 bits et où $\hat{I}(p,t)$ est l'image prédite par compensation, L et C étant le nombre de lignes et colonnes de l'image.

Pour des séquences bruitées, l'algorithme de Walker-Rao est peu robuste. Une estimation du déplacement tenant compte d'un voisinage du pixel courant améliore la robustesse. C'est la méthode proposée par Biemond *et al.* [59]. Pour N voisins, l'équation (2.27) développée en série de Taylor conduit à un système matriciel :

$$\begin{bmatrix} DFD(p_1, d^{i-1}) \\ \vdots \\ DFD(p_N, d^{i-1}) \end{bmatrix} = - \begin{bmatrix} g_x^1 & g_y^1 \\ \vdots & \vdots \\ g_x^N & g_y^N \end{bmatrix} .[d - d^{i-1}] + \begin{bmatrix} \nu(p_1, d^{i-1}) \\ \vdots \\ \nu(p_N, d^{i-1}) \end{bmatrix} \tag{2.33}$$

où g_x^j et g_y^j sont les coordonnées du gradient spatial $\nabla I(p_j - \hat{d}^{i-1}, t-1)$ en le pixel $p_j - d^{i-1}$ à l'instant $t-1$ et où $\nu(p_j, d^{i-1})$ est l'erreur de développement.

Avec les notations de [59], (2.33) peut s'écrire : $z = G.u + \nu$. On cherche alors une estimation linéaire de u à partir de z, ce qui revient à trouver la matrice pseudo-inverse L telle que : $\hat{u} = L.z$ avec $E\left\{\|u - \hat{u}\|^2\right\}$ minimum. La solution, obtenue par estimateur de Wiener, est établie par Biemond, moyennant quelques hypothèses simplificatrices :

$$d^i = d^{i-1} - \begin{bmatrix} \sum_{j=1}^N g_x^2(j) + \mu & \sum_{j=1}^N g_x(j).g_y(j) \\ \sum_{j=1}^N g_x(j).g_y(j) & \sum_{j=1}^N g_y^2(j) + \mu \end{bmatrix}^{-1} \cdot \begin{bmatrix} \sum_{j=1}^N g_x(j).DFD(p_j, d^{i-1}) \\ \sum_{j=1}^N g_y(j).DFD(p_j, d^{i-1}) \end{bmatrix} \tag{2.34}$$

où $\mu = \sigma_\nu^2/\sigma_u^2$ est le rapport des variances de u et ν. Un μ faible a un effet de lissage du champ de vecteurs estimé et assure la stabilité numérique (dénominateur non nul).

Plusieurs choix sont possibles pour le voisinage à N pixels, à condition qu'il reste causal (Fig . 2.21). Choisir $N = 1$ ramène à une formule proche de celle de Walker-Rao. L'algorithme de Biemond converge plus vite que celui de Walker-Rao. Il donne de meilleurs résultats lorsque l'hypothèse de mouvement uniforme est vérifiée pour tous les points du voisinage (typiquement à l'intérieur d'un objet mobile), mais devient moins efficace dans les zones de discontinuité de mouvement (frontière d'un objet mobile) où l'hypothèse de mouvement uniforme dans le voisinage n'est plus respectée. Pour résoudre ce handicap, Baaziz [60] propose une version combinée des deux algorithmes : l'idée consiste à faire itérer l'algorithme de Walker-Rao dans les cas où l'algorithme de Biemond n'a pas convergé. On bénéficie ainsi des avantages respectifs des deux algorithmes : robustesse au bruit pour Biemond, calcul ponctuel adapté aux frontières pour Walker. Cette démarche revient en fait à introduire un voisinage adaptatif constitué de N voisins pour l'algorithme de Biemond, ou de 1 voisin pour l'algorithme de Walker-Rao. Mais quel que soit l'algorithme choisi, seuls de petits déplacements peuvent être estimés. Dans le cas de déplacements importants, il est nécessaire d'envisager une technique multirésolution avec une stratégie de type *coarse to fine*. La transformée en ondelettes est un des choix possibles.

2.8.3 Principe du codage vidéo

La Fig. 2.22 présente le schéma de principe du codage vidéo basé sur l'estimation de mouvement.

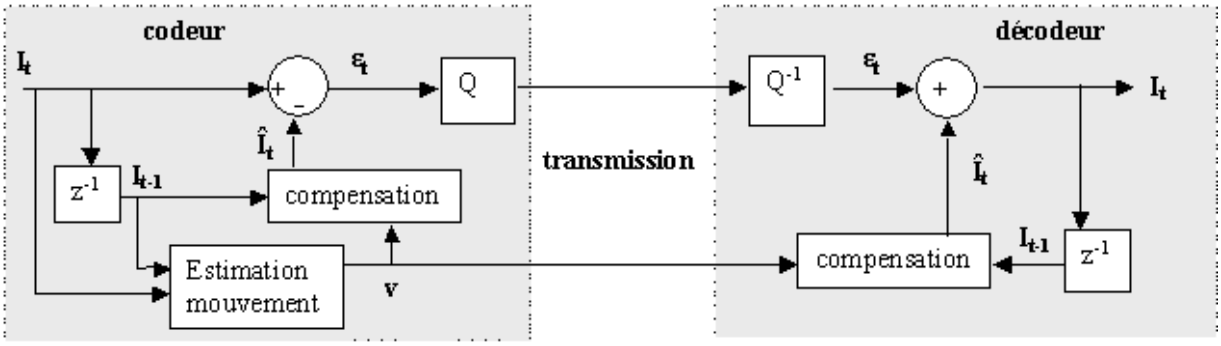

FIGURE 2.22 – Schéma de principe d'un codec vidéo avec estimation de mouvement.

2.9 Méthodes hybrides de compression vidéo

Les méthodes hybrides de compression de séquences d'images, qui reposent conjointement sur une prédiction par compensation de mouvement et sur l'utilisation d'une transformée (par exemple une transformée en ondelettes des images), présentent un grand intérêt pour la transmission à bas débit.

Pour estimer des déplacements importants et/ou accélerer la convergence de l'algorithme, il est intéressant d'intégrer cette estimation de mouvement dans une approche multi-résolution : à un niveau de résolution spatiale grossière, on estime les déplacements importants, puis on propage l'information vers des niveaux de résolution plus fine. Baaziz par exemple [60] utilise les ondelettes dyadiques de Daubechies à 4 coefficients [61], en s'arrêtant au premier niveau de décomposition (4 sous-bandes).

2.9.1 Transformée en ondelettes

La transformée en ondelettes appliquée aux images présente plusieurs avantages : analyse multi-résolution, bonne localisation spatio-fréquentielle, non redondance entre les sous-bandes [62]. Elle offre une structure hiérarchique utile pour l'interprétation de l'image. Dans le cas de l'estimation de mouvement, la stratégie *coarse to fine* est intéressante pour estimer correctement les déplacements importants. Si l'on considère une suite dyadique de résolutions, alors l'amplitude du déplacement sera divisée par 2 à chaque niveau. Or l'estimation de petits déplacements est plus aisée. De plus, on peut utiliser l'estimation à un niveau de résolution grossier comme initialisation du niveau de résolution plus fin, ce qui permet d'accélérer la convergence des algorithmes. Notons m l'indice correspondant au niveau de résolution, $m = 0$ correspondant à l'image initiale pleine résolution, m croissant quand la résolution décroît. L'analyse par transformée en ondelettes permet d'approximer un signal f de $L^2(R)$ à différents niveaux de résolution (par sa tendance ou approximation au niveau m que l'on notera $A_m \bullet f$), et d'extraire l'information de différence contenue entre 2 niveaux (les fluctuations). Notons V_m le sous-espace vectoriel contenant l'ensemble de toutes les approximations possibles de fonctions au niveau m et W_m son complémentaire orthogonal dans l'espace englobant V_{m-1}. On a : $V_{m-1} = V_m \oplus W_m$. A_m est l'opérateur de projection orthogonale sur V_m. Il a été montré qu'il existe une unique fonction $\phi(x)$, appelée fonction d'échelle, telle que les fonctions $\phi_{m,n} = \sqrt{2^{-m}}\phi(2^{-m}x - n)_{n\in Z}$ forment une base orthonormée de V_m. L'analyse consiste à décomposer le signal sur une base orthonormée d'ondelettes $\psi_{m,n}$ de $L^2(R)$, obtenues par translation et dilatation d'une unique fonction $\psi(x)$ appellée ondelette mère :

$$\psi_{m,n} = \sqrt{2^{-m}}\psi(2^{-m}x - n)_{n\in Z} \tag{2.35}$$

On peut alors exprimer la transformation en ondelettes comme une projection du signal f sur ces différents sous-espaces :

$$A_{m-1} \bullet f = A_m \bullet f + \sum_{n\in Z} \langle f, \psi_{m,n}\rangle \, \psi_{m,n} \tag{2.36}$$

Cette décomposition peut s'obtenir simplement par des convolutions du signal avec des filtres miroirs en quadrature (QMF), comme l'a montré Daubechies [61]. A chaque niveau, on filtre le signal par des filtres passe-haut et passe-bas et on sous-échantillonne. Tendance et fluctuations au niveau m ont donc un support deux fois moindre qu'au niveau $m-1$, ce qui permet de ne pas augmenter le volume des données. Pour nos tests, nous avons utilisé les filtres de Daubechies à 4 coefficients :

$$c_0 = \frac{1+\sqrt{3}}{4\sqrt{2}} \quad c_1 = \frac{3+\sqrt{3}}{4\sqrt{2}} \quad c_2 = \frac{3-\sqrt{3}}{4\sqrt{2}} \quad c_3 = \frac{1-\sqrt{3}}{4\sqrt{2}} \tag{2.37}$$

La réponse impulsionnelle du filtre passe-bas h donnant la tendance est composée des coefficients c_0, c_1, c_2, c_3, tandis que celle du passe-haut g donnant la fluctuation est composée des coefficients c_3, $-c_2$, c_1, $-c_0$, puisqu'on a la relation : $g_n = (-1)^n h_{-n+1}$ [61]. Les ondelettes de Daubechies ont l'avantage d'avoir un support compact, ce qui évite des calculs trop lourds.

Une image étant représentée par une matrice à 2 dimensions, la décomposition en tendance et fluctuations se fait selon 2 axes. On obtient 4 matrices chacune de taille 4 fois plus petite, que l'on appelle BB, BH, HB et HH, H correspondant au filtrage passe-haut et B au passe-bas appliqués de façon séparable sur les lignes et les colonnes. La première lettre correspond au filtrage en x (sur les colonnes), et la seconde au filtrage en y (sur les lignes).

La figure 2.23 schématise la décomposition d'une image en 4 sous-bandes, et montre aussi les pixels intervenant dans le voisinage multirésolution proposé par Baaziz pour appliquer l'algorithme d'estimation de mouvement de Biemond.

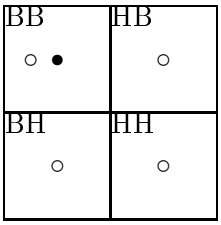

FIGURE 2.23 – Voisinage multirésolution sur 4 sous-bandes proposé dans [60].

La sous-bande BB étant celle qui contient le plus d'information (entropie la plus élevée), Baaziz applique l'algorithme itératif d'estimation uniquement dans cette bande, mais en utilisant l'information de voisinage des autres bandes. Le champ de déplacement obtenu sert à compenser chacune des bandes de fréquence, c'est-à-dire que les coefficients des 4 sous-bandes de l'instant $t-1$ sont tous translatés par le vecteur déplacement leur correspondant. A partir de ces 4 sous-bandes, on effectue une transformation en ondelettes inverse pour obtenir l'estimée de l'image à l'instant t. Cette méthode est donc fondée sur la cohérence du mouvement dans toutes les sous-bandes.

2.9.2 Critique de la méthode de Baaziz

La spécificité de la décomposition en ondelettes doit être prise en compte pour une utilisation correcte dans un schéma de compensation de mouvement. Ceci ne nous semble pas être le cas dans la mise en oeuvre proposée par Baaziz, basée sur l'hypothèse de cohérence du mouvement dans toutes les sous-bandes. Nous montrons, à l'aide de quelques simulations, que cette hypothèse est à reconsidérer : on ne peut compenser toutes les sous-bandes de façon simple avec le même champ de déplacement. En effet, pour une pyramide d'ondelettes discrètes dyadiques, une translation de l'objet dans l'image initiale (niveau de résolution $m=0$) ne correspond à une translation des coefficients de fluctuation dans la sous-bande de niveau de résolution m que si le déplacement dans l'image initiale est un multiple de 2^m. Dans le cas contraire, il n'y a pas identité entre coefficients passés translatés et coefficients présents.

La figure 2.24 donne un exemple de décomposition en ondelettes sur une séquence d'images synthétiques comportant un carré sombre se déplaçant horizontalement vers la droite de 1 pixel/image. On constate évidemment que les coefficients d'ondelettes sont non nuls au voisinage des contours d'objet (le fond gris foncé correspond à des coefficients nuls, le noir à des coefficients négatifs et le blanc à des coefficients positifs). Mais il est surtout intéressant de noter l'évolution du signe des coefficients sur cet exemple simple : alors que les coefficients des contours horizontaux gardent le même signe (contours toujours noirs ou toujours blancs), les coefficients des contours verticaux, qui sont perpendiculaires au sens du mouvement, alternent de signe à chaque image : noir, blanc, noir, blanc... Ce phénomène nous amène à reconsidérer l'hypothèse faite par Baaziz de cohérence du mouvement dans les sous-bandes.

Voyons dans quel cas une translation de Δx dans le signal original discret correspond à une translation de $\Delta n \in Z$ dans les coefficients, c'est-à-dire pour quelles valeurs de Δx l'égalité (2.38) est vérifiée :

$$\langle f(x+\Delta x), \psi_{m,n}(x)\rangle = \langle f(x), \psi_{m,n+\Delta n}(x)\rangle \tag{2.38}$$

$$\text{On a : } \int_{-\infty}^{+\infty} f(x+\Delta x)\psi_{m,n}(x)dx = \int_{-\infty}^{+\infty} f(x+\Delta x)2^{-\frac{m}{2}}\psi(2^{-m}x-n)dx$$

$$= \int_{-\infty}^{+\infty} f(y)2^{-\frac{m}{2}}\psi(2^{-m}(y-\Delta x)-n)dy = \int_{-\infty}^{+\infty} f(y)2^{-\frac{m}{2}}\psi\left(2^{-m}y-(2^{-m}\Delta x+n)\right)dy$$

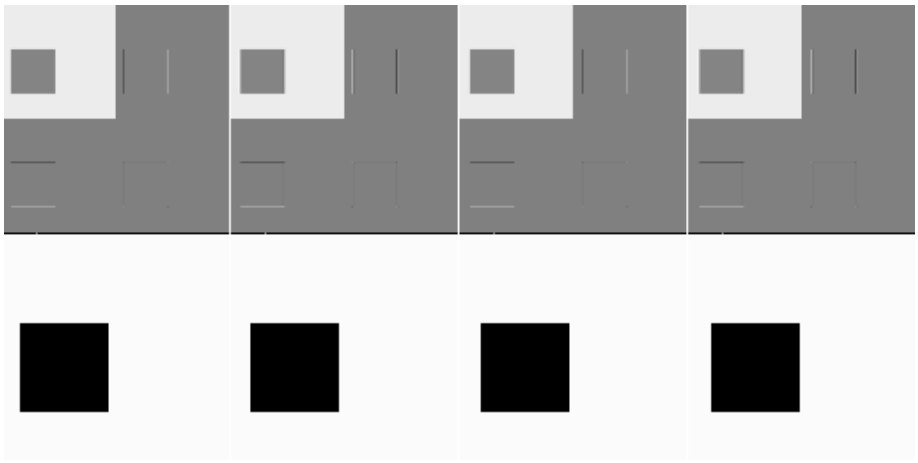

FIGURE 2.24 – (haut) Transformée en ondelette, (bas) Séquence synthétique

Donc la relation (2.38) n'est vérifiée que si le déplacement de l'objet est un multiple de 2^m. En effet, il faut avoir : $2^{-m}\Delta x = \Delta n \in Z \Leftrightarrow \Delta x = \Delta n.2^m$. Sinon, un déplacement dans le signal ne se traduit pas par un simple déplacement dans les coefficients. Ce résultat correspond en fait à la non invariance par translation que Mallat avait déjà signalée dans [62]. L'hypothèse de cohérence du mouvement dans toutes les sous-bandes est donc à revoir dans le cas général. Le problème résulte de la discrétisation des ondelettes ($\Delta n \in Z$).

D'autre part, l'algorithme de Biemond, employé avec le voisinage multirésolution de Baaziz, nécessite le calcul des gradients spatiaux dans les 4 sous-bandes. Or la différence importante de dynamique des coefficients d'ondelettes dans les différentes sous-bandes (tendance et fluctuations correspondant respectivement aux composantes BF et HF de l'image) peut poser problème pour ce calcul de gradient.

De plus, la nature des coefficients est différente : alors que les coefficients d'ondelettes sont des réels signés, ceux de la tendance sont du même type que l'image originale, c'est-à-dire des entiers non-signés. Une "normalisation" est donc nécessaire si l'on veut mêler dans un même calcul des "voisins" multirésolution si différents.

Le voisinage multirésolution proposé par Baaziz pose donc un problème d'inhomogénéité lié à la nature différente des voisins pris en compte : intensités lumineuses ou coefficients d'ondelettes HF ou BF.

La figure 2.25 illustre ces remarques sur un cas d'école : on a simulé un créneau lissé (par 2 convolutions avec un filtre binominal) et on a calculé tendance et fluctuation pour différentes positions du créneau (le déplacement étant un multiple du pixel). On constate bien la différence de dynamique entre le signal, la tendance et les fluctuations, ainsi qu'un phénomène "stroboscopique" typique de l'influence de l'échantillonnage sur les fluctuations : on ne retrouve le motif de fluctuation identique au cas du signal non déplacé que pour un déplacement multiple de 2, alors que pour un déplacement impair (1 ou 3), le motif est notablement différent, ce qui est évidemment gênant pour un calcul ultérieur de gradient.

2.9.3 Discussion

On conclut de cette étude que l'hypothèse de cohérence du mouvement entre les 4 sous-bandes suppose le respect de conditions restrictives à la fois sur la nature des discontinuités spatiales dans l'image (amplitude des contrastes, absence de fronts raides pour respecter le théorème de Shannon) mais aussi sur l'amplitude du mouvement des objets dans l'image, conditions rarement maîtrisables en pratique.

Par ailleurs, la spécificité des coefficients d'ondelettes pourrait être exploitée de façon intéressante. Notamment, on propose d'utiliser l'information de direction propre à chaque sous-bande (contours horizontaux, verticaux, diagonaux) pour obtenir des caractéristiques fines sur le mouvement (amplitude et sens), en considérant l'évolution des fluctuations au cours du temps. La figure 2.26 illustre le principe pour extraire uniquement les contours en mouvement de l'ensemble des contours (que l'on peut obtenir par simple binarisation des coefficients d'ondelettes ou par détection de passages par zéro). D'autre

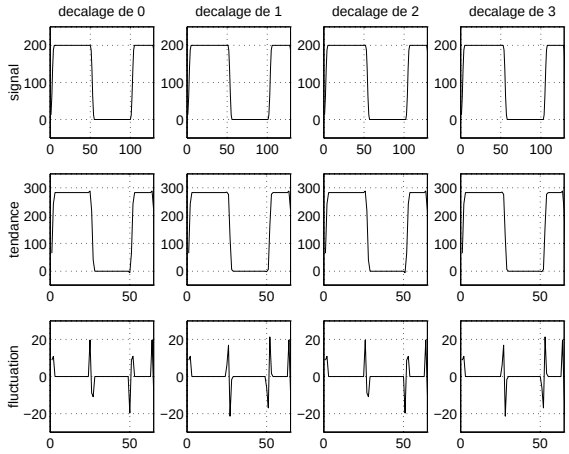

FIGURE 2.25 – Simulation du mouvement d'un créneau dans un signal 1D

part, on sait qu'il est facile d'estimer un déplacement perpendiculaire à un contour. On peut donc estimer, dans chaque sous-bande, uniquement le déplacement dans la direction perpendiculaire aux contours mobiles, puis fusionner toutes ces informations, précises mais parcellaires, pour reconstituer le mouvement de l'objet global.

Une autre proposition consiste à utiliser une décomposition en ondelettes sur le signal à 3 dimensions que constitue la séquence d'images, ce qui reviendrait en fait à une méthode proche de celle proposée par Heeger [54].

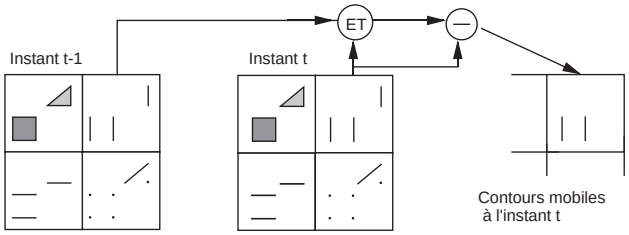

FIGURE 2.26 – Principe d'extraction des contours mobiles (cas d'un carré mobile et d'un triangle fixe)

Chapitre 3

Normes et standards du multimédia

3.1 Introduction

Le terme multmédia naît dans les années 50 pour désigner un système de présentation avec projecteur de diapositives couplé à un magnétophone déclenchant l'avance des diapos.

Etymologiquement, Communication Multimédia signifie : "plusieurs moyens" pour faire passer un message. Elle résulte de la synthèse entre trois métiers : informatique (micro-ordinateur, bases de données, internet), télécommunications (téléphonie, réseaux haut débit) et audiovisuel.

Les objets manipulés sont l'image, la vidéo, le son, la parole, les données, le texte, le graphique, le modèle 3-D. Dans le futur, d'autres objets tactiles, olfactifs ou gustatifs entreront en jeu.

Les applications multimédia sont les programmes qui manipulent ces contenus : transmission TV numérique, site web, jeu vidéo, visioconférence, visiophone, serveur vidéo interactif, banque d'images, borne interactive, catalogue vidéo, identification d'individu, encyclopédie, commerce électronique.

La numérisation est la clé technologique de la communication multimédia.

La synchronisation entre médias est cruciale : un son décalé de quelques dixièmes de seconde par rapport à la vidéo est intolérable.

La standardisation est indispensable pour l'interopérabilité, la pérennité et la rentabilité des applications de communication multimédia : "Standard is volume", d'où l'importance des normes et formats [63]. La standardisation influence le marché (e.g. Windows qui n'offre pas de produit capable de télécharger son système d'exploitation est concurrencé par Java qui permet cette fonctionnalité). Le processus de normalisation implique 4 acteurs : industriels (telecom), développeurs (informatique), créateurs de contenu (audiovisuel), consommateurs. Une norme met 5 à 10 ans à s'imposer.

3.2 Manipulation d'objets multimédia

- Les manipulations en entrée (création) concernent : la saisie, la numérisation, le codage, la compression, la protection des contenus.
- Les manipulations intermédiaires concernent : le stockage, l'archivage, la transmission, la distribution.
- Les manipulations en sortie (exploitation) sont : la sélection, la décompression, le décodage, la restitution analogique, la reproduction.

3.2.1 Saisie

- Capture d'une image naturelle (appareil photo, caméra, scanner)
- ou création d'image de synthèse (VRML).

Physiologiquement, une image couleur est décrite par 3 grandeurs : la teinte (information de couleur reconnue et mémorisée par le cerveau), la luminosité (ou intensité) et la saturation (ou pureté) indiquant le pourcentage de blanc mélangé à la couleur (cf propriétés de l'œil).

L'œil discerne 360000 nuances (120 couleurs pures $\times$ 30 niveaux de saturation $\times$ 100 intensités).

Pouvoir séparateur de l'œil : 0.3 milliradian (3mm à 10m).

3.2.2 Numérisation

pixel = élément d'image discret (*picture element*)

La quantification est conditionnée par le pouvoir séparateur de l'œil : 1/100. Donc 8 bits suffisent pour coder la luminance. Un pixel couleur est alors codé sur 24 bits. Avec moins de 7 bits (128 niveaux), on voit apparaître le phénomène des bandes de Mach sur un dégradé continu. C'est le cas pour un codage couleur sur 16 bits : 5 bits (32 niveaux) par couleur, plus un bit de masquage pour l'incrustation (overlay).

- La résolution d'une image s'exprime en ppm (points par millimètre) ou en dpi (dot per inch ; 1 pouce=25mm).
- Image imprimée : typ. r=300 à 600 dpi (imprimante laser).
- Pellicule photo, diapo : typ. r=100ppm.
- La définition est le produit de la résolution par la dimension physique de l'image : $d = r \times dim$.

Différents formats caractérisent différents types d'images associés à une définition spécifique (Tab. 3.1).

TABLE 3.1 – Définition des images

type	définition	nb points	taille fichier
QCIF	176×144	25000	50ko à 16 bpp (32000 couleurs)
CIF	352×288	100000	200ko à 16 bpp (32000 couleurs)
TV CCIR	720×576	400000	1.2Mo à 24 bpp (16 millions de couleurs)
TVHD	1440×1152	1600000	4.8Mo à 24 bpp (16 millions de couleurs)
TVHD 16/9	1920×1152	2211840	6.6Mo à 24 bpp (16 millions de couleurs)
VGA	640×480	300000	300ko à 8 bpp (256 couleurs)
SVGA	800×600	480000	480ko à 8 bpp 1Mo à 16 bpp 2 Mo à 32 bpp
XGA	1024×768 1280×1024	768ko 1.3Mo	1.5Mo à 16 bpp 4Mo à 24 bpp
A4 N&B	21×29.7cm	8640000	8.6Mo à 8 bpp (256 NdG)
diapo 24×36	2000×3000	6000000	18Mo à 24 bpp (16 millions de couleur)

- Le format CIF correspond au quart de l'image de TV classique. La qualité CIF équivaut à celle d'un magnétoscope VHS.
- Le format CCIR est le seul format normalisé de présentation d'image.

Les formats informatiques :

- non-entrelacé,
- $f > 60Hz$ pour éliminer le papillotement (*flicker*)
- *pitch*=taille du point élémentaire de l'écran (0.3mm)
- attention aux formats trop grands (21 pouces) : fatigue visuelle.

3.2.3 Codage

"La nuit, tous les chats sont gris"

codage YCrCb : $Y = 0.3R + 0.59G + 0.11B$

codage 4 : 2 : 2 : 4 valeurs de Y pour 2 valeurs de Cr et 2 valeurs de Cb.

3.2.4 Compression

Sans perte (lossless) GIF : taux de compression 1 :2 ($t = 2$)

Avec perte (lossy) JPEG : $t = 10$ sans altération visuelle

3.2.5 Stockage

Intérêt des formats multi-échelle.

3.2.6 Protection et Identification du contenu

Techniques d'aquamarquage :
- soit sur l'amplitude de certains points-image,
- soit au niveau du spectre fréquentiel,
- soit au niveau du code compressé.

L'opération de contrôle des aquamarques s'appelle le *monitoring*.

Pour l'identification des objets numériques : 64 bit suffisent.

3.2.7 Transmission

protocole de transmission : modèle OSI en 7 couches.

"An image is worth a thousand words" : rapport de taille de fichier de 1/1000 entre une page d'écriture et une image.

besoin : pour une borne interactive, le temps d'attente maximal est de 9 secondes.

La Fig. 3.1 donne pour un taux de compression $t = 20$, les temps de transmission d'image (typ. transfert à 10s/img pour image CIF transmise par GSM).

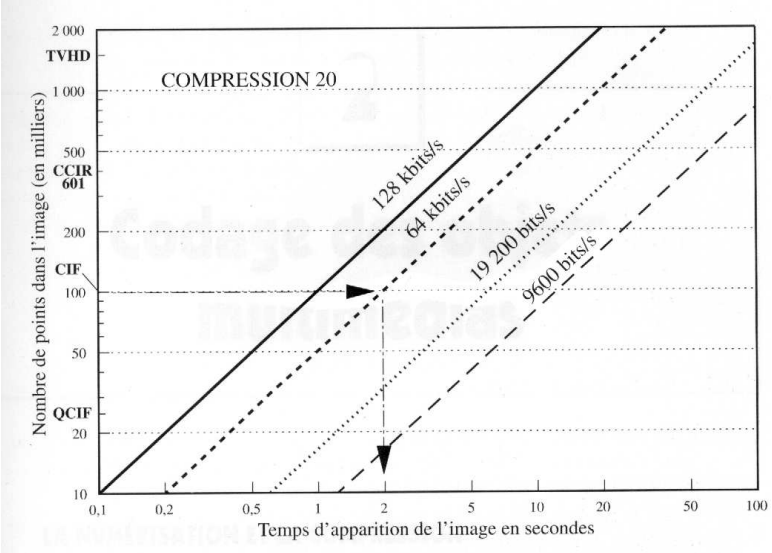

FIGURE 3.1 – Transmission d'images fixes compressées [63].

3.2.8 Restitution

- synthèse additive : écrans cathodiques. La distance optimale d'observation est fonction du *pitch* et de l'œil.
- synthèse soustractive : imprimantes
- calibrage des couleurs : table de référence Q60A.

3.3 Codage et compression d'objets multimédia

Une minute de vidéo de définition 640×480 en 16 millions de couleurs représente 1.4Go. Sa transmission temps-réel nécessite un débit de 185Mbit/s. D'où le besoin de compression.

La compression sans perte est utile pour le médical, ou pour la reconnaissance des formes.

La compression avec perte utilise trois types de procédés : DCT, ondelettes ou fractales.

3.3.1 JPEG : exemple de DCT

La compression comporte 5 étapes :

1. découpage en blocs 8×8

2. transposition amplitude-fréquence : centrer les échantillons ; appliquer la DCT (nombre d'opérations proportionnel au carré du nombre de points).

$$S_{uv} = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} S_{yx} \cos \frac{(2x+1)\pi u}{16} \cos \frac{(2y+1)\pi v}{16} \tag{3.1}$$

où $C_u = 1$ (resp. C_v) sauf pour $u = 0$ (resp. v) auquel cas $C_u = 1/\sqrt{2}$ (resp.C_v).

3. filtrage : matrice de quantification résulte d'essais psychovisuels. L'œil est un filtre passe-bas (Fig.)
4. réorganisation en zig-zag
5. codage entropique (Huffman) : les valeurs les plus fréquentes sont codées avec les mots les plus courts.

La décompression correspond aux opérations inverses.

Performance atteinte pour une image couleur 24 bits/pixel (24bpp) :
- bonne qualité à 0.75bpp ($t = 30$).
- visuellement identique à l'original à 2.25bpp ($t = 10$).

3.3.2 JPEG2000 : exemple d'ondelettes

Principes de la méthode (Fig. 3.2) : multi-échelle, filtres localisés spatialement, bandes fréquentielles orientées spatialement [64, 65].

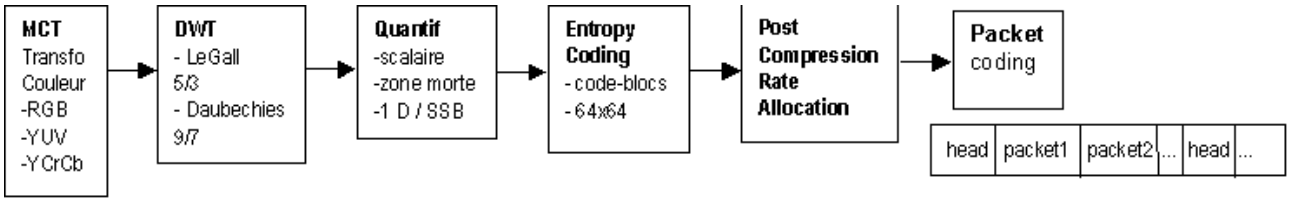

FIGURE 3.2 – Schéma-bloc de JPEG2000.

Complexité : o(n) opérations.

Performance : 0.6bpp

Sites web :

```
http://linuxj2k.org/ (version système linux)
http://www.jpeg.org (information générale)
http://www.migrator2000.org
http://jura.jpeg.org
http://jj2000.epfl.ch (publications disponibles)
http://jpeg2000.epfl.ch (implantation JAVA avec interface graphique)
http://www.ece.uvic.ca/~mdadams/jasper/ (implantation en C)
```

Limites de JPEG :
- compression avec perte limitée en pratique à 0.25 bpp ;
- pas de sans perte (excepté JPEG-LS)
- convient surtout pour des images naturelles (problème des documents composés avec texte)
- décodage progressif limité
- peu robuste aux erreurs

Avantages de JPEG2000 par rapport à JPEG :
- nouvelles fonctionnalités : ROI, accès aléatoire, flexibilité
- robustesse aux erreurs : BER=10^{-5} à 10^{-4} (voire 10^{-3} pour liaison radio, sans fil).
- gain en SNR : 3 à 6dB
- applications : spatial, médical

La norme comporte 11 parties :

1. Part1 : format de base JP2 (décembre 2000)
2. Part2 : extension pour applications spécifiques (e.g. hyperspectral) : format JPX (équivaut à du PDF normé international)
3. Part3 : Motion JPEG2000 : MJP
4. Part4 : conformance
5. Part5 : logiciel de référence
6. Part6 : images composites (multi-couche) : JPM
7. Part7 : VIDE (était prévu pour les caméras numériques)
8. Part8 : sécurité (authentification, droits, accès) : JPSEC
9. Part9 : interactivité et protocole JPIP
10. Part10 : volumétrique : JP3D
11. Part11 : sans fil (téléphone, vidéoprojecteur) : JPWL

(a) JPEG (compression 1 :90) (b) JPEG2000 (compresion 1 :90 ;YUV ; DWT9/7)

FIGURE 3.3 – JPEG2000 versus JPEG : gain $\times 2$ en compression.

3.3.3 Fractales

Principes de la méthode : géométrie fractale (autosimilarité), théorème du point fixe (attracteur), itération d'une transformée affine (IFS).

3.3.4 MPEG

MPEG est un format non entrelacé.

MPEG2 permet un débit de 20Mbit/s. Il offre plusieurs profils : le plus courant MP@ML (*Main Profil at Main Level*) correspond à 720×576 à 25 img/s.

3 types d'images : I,P et B.

3.3.5 H261-H263

Formats pour le visiophone.

3.3.6 MP3

Couche 3 de la norme audio de MPEG2.
retard minimal codage/décodage : 59 ms.
Stockage (intérêt pour autoradio) : 10 heures de musique sur un CD.
Son "qualité CD" (44kHz, 16bits) : taux de compression $t = 10$, débit=64kbit/s

3.3.7 Formats de fichiers images fixes

Il existe deux bibles des formats de fichiers [66, 67]. Un fichier image est constitué de :

- en-tête : mode d'emploi du contenu, magic number de 2 octets (SOI pour JPEG), dimensions de l'image (typ. 512×512 en robotique)
- données (brutes ou compressées) : codes binaires
- métadonnées (sous forme d'étiquettes ou tags) : identifiant de transport, de trame intra, sous-titres, protection (copyright), colorimétrie, réglage de caméra, scénario, synchronisation, time-code, positionnement GPS, prise de vue, panoramique, mot-clé pour catalogage et moteur de recherche ;

Pour images fixes : nombreux formats propriétaires.

Nouveau système de compression : JPEG2000.

Formats moins courants : TGA, PCX (PaintBrush), RAS (Sun Raster), PICT (Macintosh).

Les formats utilisant une table de couleurs (palette) ne présentent plus guère d'intérêt (carte vidéo sans limite de mémoire à 32 bit/pixel).

BMP

- Format bitmap de Windows
- Entête de taille fixe : 54 octets
- Magic number : BM (2 mots)
- taille fichier (4 mots)
- reservé pour extension : 2x2 octets
- offset de début (4 octets) : 36H=54
- adr.14 : taille en-tête (1 mot) : 28H=40 octets
- largeur image (4 octets) : ici 01A4H=420
- hauteur image (4 octets) : ici 00FFH=255
- nb de plans (4 octets) : ici 1
- nb de bits par pixel (18H=24)
- compression : 4 octets nuls = RLC
- taille des données image (4 octets)
- résolutions horizontale et verticale (pixel/mètre) : 2×4 octets
- adr.46 : nb de couleurs utilisées (4 octets)
- couleurs importantes (4 octets)

TIFF

Le plus répandu (Adobe Photoshop).

Précurseur des formats avec métadonnées.

Structure en tags (Tagged Image File Format).

GIF et PNG

- adapté aux (petites) images de synthèse (bouton, logo).
- restitution progressive
- compression sans perte (Lempel-Ziv Welch).
- Créé par Compuserve.
- NB : le format PNG est une norme ISO (équivalent à GIF sans brevet).

Formats issus de la norme JPEG

JFIF C'est un format à la norme JPEG extrêment réducteur.

Codage des couleurs : selon le CCIR 4 :2 :2.

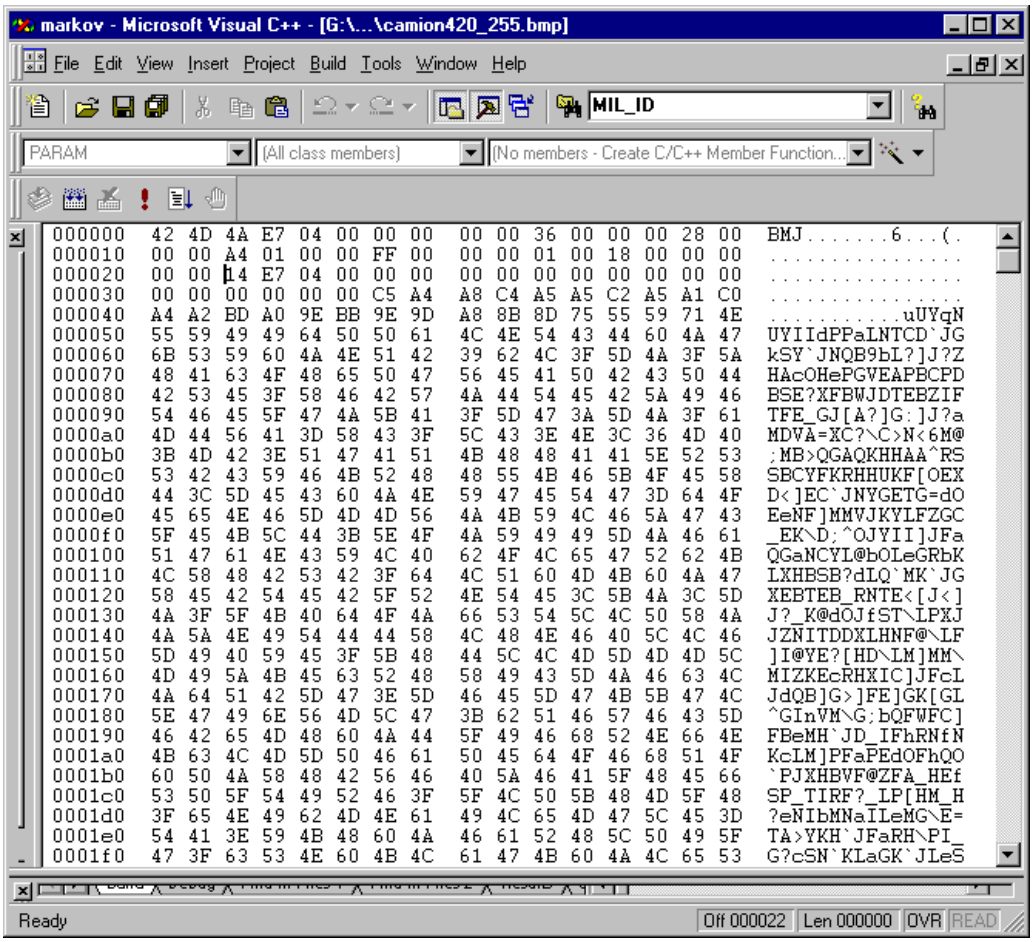

FIGURE 3.4 – Fichier BMP.

SPIFF SPIFF (Still Picture Interchange file format) est le seul format du domaine public (norme JPEG, partie 3).

Nouveau format ISO pour garantir l'interopérabilité.

JTIP JTIP (JPEG Tiled Image Pyramid) est le premier format pyramidal de JPEG.

Il répond à la tendance actuelle (pour gérer les grandes images) : représentation pyramidale et découpage en tuiles.

Taille de tuiles variable (par défaut 720×576).

FlashPix

Format propriétaire multirésolution (pyramide et tuiles).

Taille de tuiles fixe : 64×64.

Formats liés à la photo numérique

- **PhotoCD** : Format pyramidal mono-fichier, codage chroma 4 :2 :0 (discutable). Sur un CD : stockage de 100 images 2000×3000. Kodak.
- **EXIF** : EXchangeable Image file format for Digital Still Camera. Consortium japonais de fabricants d'appareils photo numériques. Le fichier contient 2 images : vignette + image complète.
- **CIFF** : format JFIF simple + informations relatives à appareil photo.

3.3.8 Représentation 3-D

Norme VRML (Virtual Reality Modeling Language).

Origine : Né en 1994, issu du langage Open Inventor de Silicon Graphics

Objectifs : pour la description des mondes 3-D et la prise en compte des hyperliens.
Evolution de VRML vers internet : Web3D.
Principe : Un objet est codé par des sommets (vertex) $V(x, y, z)$ et des faces $F(V_1, V_2, V_3)$. Le maillage repose sur une information géométrique (position) et une information topologique (connectivité). On appelle valence d'un sommet le nombre de ses voisins. Le coût de codage naif (3F 6V) vaut (avec du int32) : 192bit/sommet. Un codage adapté donne : $6Vlog_2(V) = 100b/s$. La borne inférieure (demontrée par Tutte en 1964) est : 3.24b/s. On peut donc compresser la représentation 3-D.

3.4 Les classes d'applications multimédia

Paramètres régissant la complexité d'une application :
- mono/multiposte
- temps réel ou temps différé : au sens informatique, capacité à réagir instantanément à un événement.
- réseau local ou longue distance

Exemples d'application locale (off-line) :
- CD-Rom (600Mo) et DVD (4 à 17 Go) pour résoudre le stockage (1 heure de vidéo MPEG1=675Mo), la portabilité (galette de 12cm), le piratage (découpage du monde en zones).
- réseau local : partage de ressources, interaction entre utilisateurs.

Exemples d'application distante (en ligne) :
- Internet : protocole simple IP (Internet Protocol).
  Couche de contrôle : TCP (Transmission Control Protocol).
  Internet pose problème dans la gestion du temps (pb de synchronisme).
  retard de 300ms maxi acceptable pour la conversation.
  Besoin de resynchronisation des paquets.
  Solution : applets Java ; protocole supplémentaire RTP et RTCP (Real Time Control Protocol) qui gèrent la priorité des paquets.
- Intranet : réseau d'entreprise.
  Photothèque numérique de Renault (30000 images ; 50Go).
  Photothèque Yves Rocher (20000 visuels, multilingue).
- Téléphone d'accès à Internet : Miniweb dont l'ancêtre est le Minitel (datant de 1980).
  Interface RNIS : 64 ou 128kbit/s
  Interface xDSL : 1 Mbit/s
  Téléphones internet : WebTouch (Alcatel) et Tel@phone (Matra).
- Terminaux d'accès Internet pour TV : WebTV et DomoTV apparus en 1997.
  Boîtier d'interface avec modem : décodeur appelé *set-top box*.

3.5 Codage des applications multimédia

Objectifs [68] : pérennisation de la communication, interopérabilité, indépendance vis-à-vis des plates-formes, portabilité.
On distingue :
- les normes de codage (du contenu, de la structure) : Un document électronique comporte un contenu, une structure logique, des éléments de présentation. La structure logique du document est décrite à l'aide de balises.
- les normes d'échanges (du contenu, du contenant) : gestion de l'interactivité. Les normes MPEG4 et MPEG7 rapprochent les notions de document et d'application (en incluant dans les documents MPEG4 des éléments d'interactivité).

3.5.1 HTML

- HyperText Markup Language.
- Langage de balisage très simple et portable.
- La DTD (Définition de Type de Document) fournit la syntaxe des balises.

– document HTML comporte 3 parties : ligne de la version, entête déclarative, corps.
Exemple :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"
"http://www.w3.org/TR/REC-html40/strict.dtd">
<HTML>
<HEAD>
<TITLE> Titre du document html </TITLE>
</HEAD>
<BODY>
<P> Ceci est le corps du document
</BODY>
</HTML>
```

3.5.2 XML

Extensible Markup Language
Langage de balise de document avec information structurée, qui devrait succéder à HTML.
Il est moins limité que HTML, mais moins lourd à implanter que le SGML (Standard Generalized Markup Language).

3.5.3 WAP et WML

Le WAP (Wireless Application Protocol) est destiné aux applications mobiles sans fil. Il est basé sur XML et IP.
La téléphonie mobile a connu une évolution rapide résumée dans le lexique ci-dessous :

– **GSM (Global System for Mobile communications)** : adoptée en Europe à partir de 1992, cette norme dite de deuxième génération (2G) a remplacé l'analogique. La plupart des réseaux de téléphonie mobile fonctionnent encore sous ce sytème, très lent pour la transmission de données, à l'exception des SMS (minimessages écrits).
– **WAP (Wireless Application Protocole)** : ce protocole permet de naviguer sur Internet à partir du réseau GSM. Introduit en 1999, le WAP, trop lent, a été un échec commercial.
– **i-mode** : ce standard permet de naviguer sur Internet. Conçu par l'opérateur japonais NTT DoCoMo, il est développé sous licence en Europe par le français Bouygues Télécom, l'espagnol Telefonica Moviles et le néerlandais KPN.
– **GPRS (General Packet Radio Service)** : lancée en 2002, cette technologie dite de "deuxième génération et demie" (2,5G) améliore la transmission sur le réseau GSM grâce à une commutation de données par paquets, optimisant ses capacités. Elle permet une transmission de données multimédia 5 à 6 fois plus rapide que sur le réseau GSM classique.
– **UMTS (Universal Mobile Telecommunications System)** : norme dite de troisième génération (3G), elle est mise en place très progressivement en Europe. En offrant un débit beaucoup plus important que le GPRS (5 à 8 fois plus rapide), l'UMTS permet la transmission de données plus riches, notamment la vidéo.

3.5.4 MHEG

Multimedia & Hypermedia Expert Group.
Les moteurs MHEG-5 utilisent une machine virtuelle Java.

3.5.5 Applications distribuées

Corba

Common Object Request Broker Architecture.

COM

Le standard COM (Component Objects Model) de Microsoft utilise les liens OLE.

Java

Ressemble au C++
Avantages : portabilité, sécurité, gestion automatique de mémoire, modèle de pointeur sûr.
Inconvénient : lenteur
Machine virtuelle JVM
Applet (*application object*) : élément de marquage qui assure le lien entre page HTML et code Java.

MPEG-4 et MPEG-7

- MPEG4 est un standard orienté objet, c'est-à-dire qu'il code le contenu après une reconnaissance des objets qui composent l'image.
  MPEG-J : intégration de Java à MPEG-4 (pour décrire des scènes intercatives et gérer des contrôleurs de média).
  Objectifs de MPEG-4 : haut et bas débit, interactivité, compressions fortes (5 à 64 kbit/s en téléphonie mobile, 2Mbit/s pour les films).
- MPEG7 code la description du contenu. Norme de description standardisée des données permettant une interprétation informatique (métadonnées).
  Utile pour moteur de recherche et indexation, reconnaissance des images par leur contenu.
  Les outils de description du contenu Visuel sont groupés en diverses catégories : couleur, texture, forme, contour, mouvement.
  Les outils de description Audio sont de 3 types : effets sonores, instruments, reconnaissance vocale.
  MPEG4 et 7 sont complémentaires (cf MPEG-21). Ils ne définissent rien au niveau de la segmentation d'image.

3.6 Compression de Séquences Vidéo

3.6.1 Principe du Codage Hybride

Le schéma de principe d'un codage hybride (estimation de mouvement et transformée fréquentielle) est donné sur la Fig. 3.5 [69].

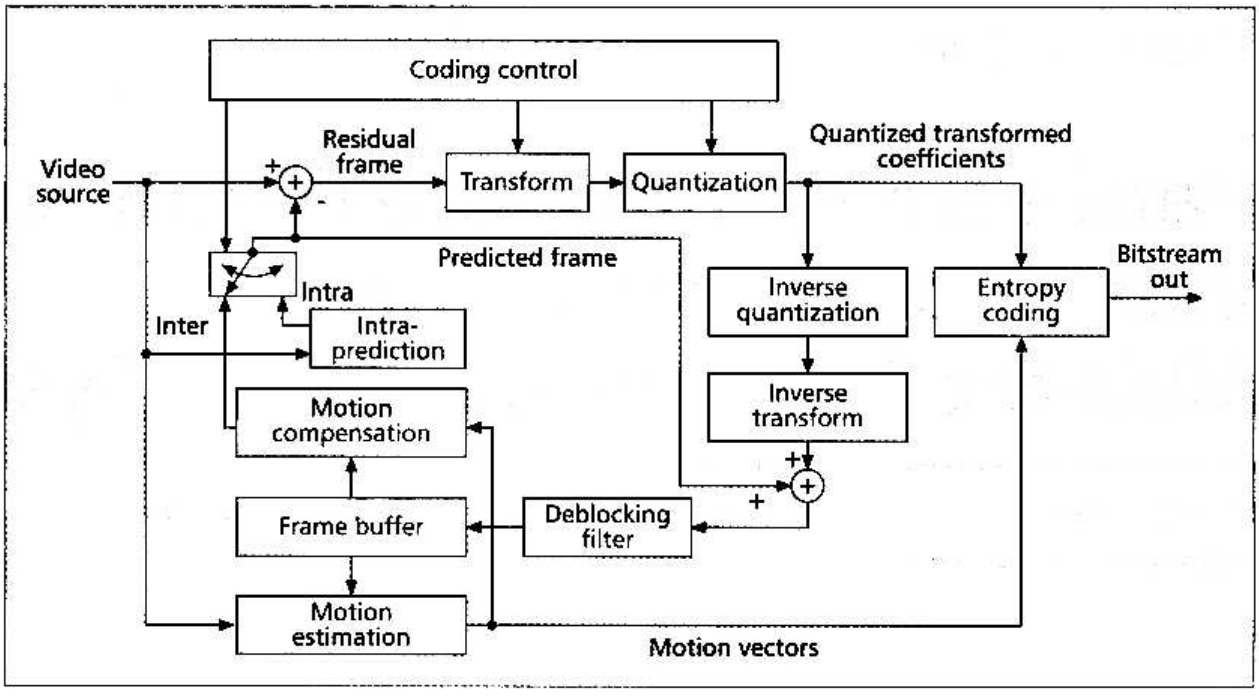

FIGURE 3.5 – Schéma-Block d'un Codeur Hybride.

- Le **buffer de trames** stocke une ou plusieurs images de référence.
- L'**estimation de mouvement** utilise la mise en correspondance de blocs entre image courante et images de référence pour calculer les déplacements (vecteurs de mouvement). C'est le module le plus important et le plus coûteux.
- La **compensation de mouvement** peut alors reconstruire l'image courante à partir des vecteurs de mouvement et de la référence (image prédite dans le domaine temporel : interprédiction)
- L'**intraprédiction** réalise une autre forme de prédiction reposant uniquement sur l'image courante.
- La **différence** entre l'image courante et l'image prédite (inter ou intra) constitue l'erreur résiduelle à transmettre.
- La **transformée fréquentielle** et la quantification réduisent la redondance spatiale : en transformant les données spatiales en données fréquentielles, la représentation devient plus compacte (plus facile à compresser).
- La **quantification** des coefficients fréquentiels repose sur le SVH (système visuel humain, cf. masquage). Les coefficients quantifiés ont une meilleure distribution statistique en vue de la compression.
- Le **codage entropique** supprime la redondance statistique et encode les coefficients transformés quantifiés et les vecteurs de mouvement sous forme d'un flux de bits (bitstream) en sortie
- Par ailleurs, les coefficients sont aussi utilisés après **inversions** pour reconstruire les trames stockées dans le buffer.
- Parfois, un **filtre anti-blocs** est appliqué avant le buffer de trames, pour réduire les effets de blocs et améliorer la qualité subjective. C'est un filtre passe-bas (cf. visiophone H.261) qui coûte 33% des calculs du décodeur H.264.

3.6.2 Normes MPEG-4 et H.264

Parmi les algorithmes de compression vidéo, les standards MPEG-4 (standard ISO 1999) et H.264 (standard ITU 2003) sont les plus performants mais complexes et très coûteux en calculs, d'où le besoin d'architecture matérielle et de circuits SoC pour encoder de la vidéo 720×480 à 30Hz [69].

Pour les systèmes de communication multimédia temps-réel, plusieurs technologies de pointe sont requises :

- réseau large bande (pour augmenter la bande passante),
- compression (pour réduire le bitrate, la vidéo étant la plus gourmande),
- technologie VLSI pour fabriquer des codecs hardware (la capacité des puces double tous les 18 mois suivant la loi de Moore).

MPEG-4 a trois objectifs : interactivité (requiert le codage basé objet), compression, accès universel (scalability, robustesse aux erreurs). Plusieurs profils et niveaux existent, dont le profil simple (SP) et le profil simple avancé (ASP).

H.264, aussi appelé Joint Video Team (JVT) ou MPEG-4 Advanced Video Coding (AVC), est destiné à la diffusion, à la haute définition de DVD, au streaming, au stockage et à la surveillance vidéo. Il a démarré en 1999 (projet long terme H26L) et a été finalisé en 2003 sous le nom MPEG-4 Part 10 AVC. Il offre un gain de 60 % par rapport à MPEG-2 et un gain de 40% par rapport à MPEG4-ASP. H.264 a quatre profils : basique (téléphonie et mobiles), étendu (internet streaming), principal (video broadcasting, jeux) et haut (HDTV).

3.6.3 Caractéristiques des Nouveaux Standards

- compensation de mouvement : précision au quart de pixel
- code à longueur variable adaptatif au contexte (CAVLC) : choix de la LUT en fonction du voisinage du bloc
- taille de blocs variable (41 SAD à calculer par bloc candidat)
- taille de la fenêtre de recherche : $[-64, +63]$ en horizontal et $[-32, 31]$ en vertical pour la première image de référence, $[-32, 32]$ et $[-16, 15]$ pour les autres.
- images de référence multiples

– Intrapédiction : 4 modes possibles pour chaque bloc 16×16, 9 modes pour chaque bloc 4×4. Elle peut s'implanter sur une architecture à bus de données reconfigurable.

Notons qu'entre MPEG-4 standardisé en 1999, et H.264 standardisé en 2003, la complexité de calcul (donc le besoin en puissance de calcul) a été multipliée par plus de 10 (plus rapide que la loi de Moore qui ne donne que $\times 6.36$ sur 4 ans). Ceci justifie le besoin d'accélérations matérielles.

Le profilage d'instructions à l'encodeur indique les modules critiques à optimiser (implantation matérielle parallèle, instructions spéciales MMX) : l'estimation de mouvement pèse 95% du coût total (la recherche exhaustive jouant le rôle majeur). Pour MPEG-4, un pipeline à 2 étages sur les macro-blocs suffit. Pour H.264, un pipeline à 4 étages est nécessaire. Un codeur H.264 sur puce en technologie UMC $0.18\mu m$ qui mesure $8 \times 4\ mm^2$, comporte 1000K portes logiques, 35ko de mémoire et fonctionne à 81 MHz, permet l'encodage en temps-réel (NB : 30 images/s en CIF profil de base requièrt 300 GIPS).

Chapitre 4

Exercices

4.1 Débit d'information

Calculer le débit d'information (en bits/seconde) apporté par un émetteur TV noir et blanc dans les deux cas suivants :

1. contraste poussé à fond (pixel soit noir, soit blanc)
2. 256 niveaux de gris dans l'image
3. quelle conséquence peut-on en déduire en terme de besoin ?
4. étude de cas d'une transmission sans fil sur réseau GSM (9600 bauds). Extrapoler avec l'évolution technologique : TV couleur, réseau GPRS, UMTS, compression JPEG, MPEG etc.
5. évaluer entropie et redondance de la source TV N& B dans les 2 cas étudiés

N.B. : Standard en Europe : 25 img/sec ; 520000 pixels/img

4.2 Modification d'histogramme

Sur la Fig. 4.1 sont représentées 8 images dont l'originale en haut à gauche. A droite de chaque image est représenté l'histogramme correspondant.

Déterminer l'allure de la correction opérée (par exemple par des LUTs) pour obtenir les différentes images à partir de l'image originale.

A titre d'exemple, la correction neutre est représentée sur le premier histogramme.

4.3 Principe de l'égalisation d'histogramme

Le principe de l'égalisation d'histogramme dans le cas continu repose sur les équations suivantes :

$$s = T(r) = \int_0^r p_r(u)du \tag{4.1}$$

$$p_s(s) = p_r(r)\frac{dr}{ds} \tag{4.2}$$

où $r \in [0;1]$ représente le niveau de gris de l'image en entrée, $s \in [0;1]$ le niveau après égalisation, et $p_r(r)$ et $p_s(s)$ les densités de probabilité correspondantes.

On considère un histogramme continu défini par :

$$p_r(r) = -2r + 2 \text{ pour } 0 \leq r \leq 1 \tag{4.3}$$

$$p_r(r) = 0 \text{ ailleurs} \tag{4.4}$$

1. Représenter l'histogramme initial
2. Calculer $s = T(r)$ et représenter l'allure de la courbe correspondante
3. Exprimer r et $\frac{dr}{ds}$ en fonction de s et en déduire l'expression de $p_s(s)$
4. Tracer la courbe $p_s(s)$ en fonction de s et conclure sur l'effet d'égalisation d'histogramme.

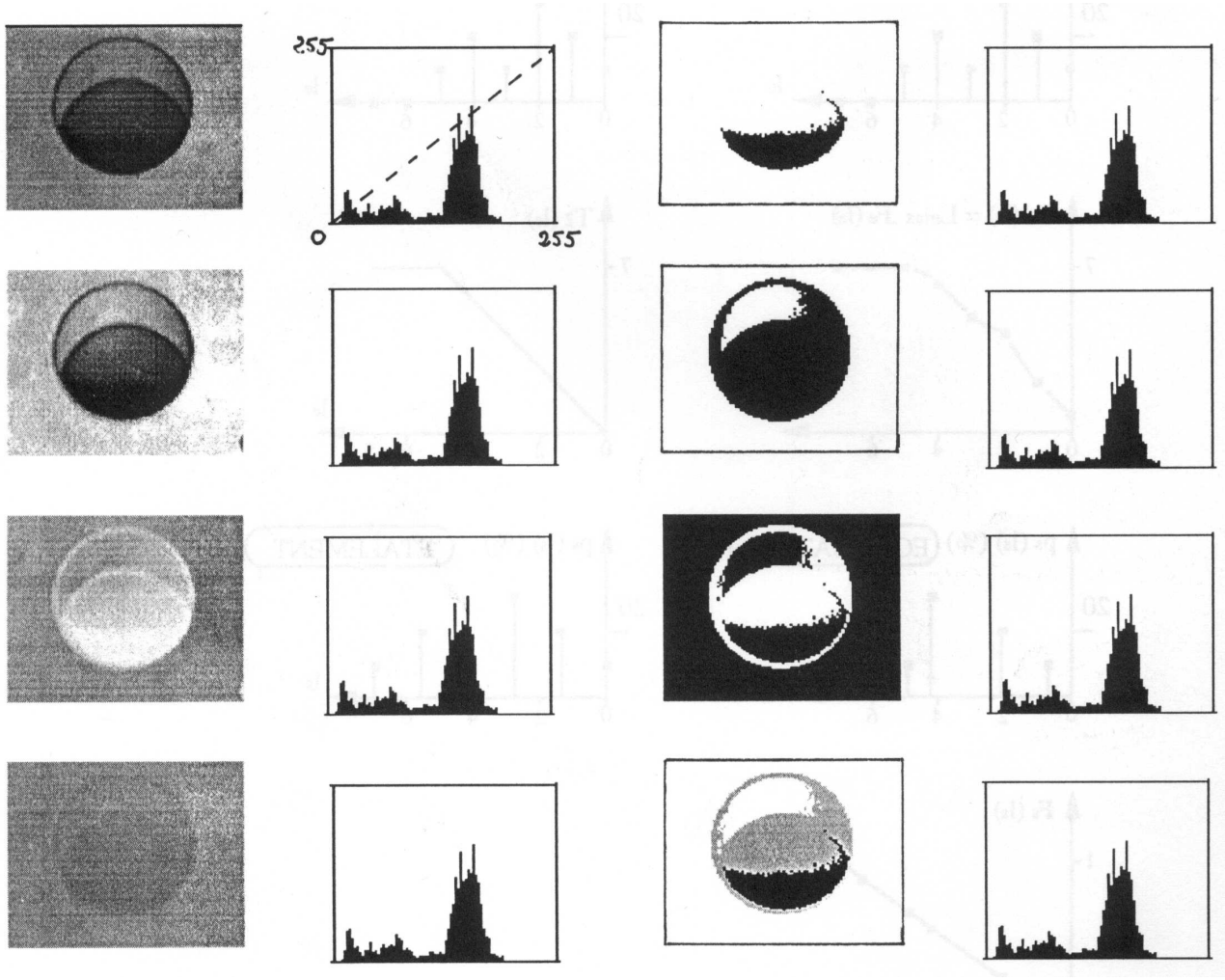

FIGURE 4.1 – Images et histogrammes [10].

4.4 Egalisation et étalement

Soit une imagette de taille 10×10 à 8 niveaux de gris dont les probablilités sont données dans le Tab. 4.1.

TABLE 4.1 – Histogramme

niveau i	probabilité $p_i(\%)$	$F(i)$	niveau j	étalement $T(i)$
0	10			
1	20			
2	30			
3	10			
4	20			
5	10			
6	0			
7	0			

1. réaliser l'égalisation de l'histogramme (on complétera le tableau avec les valeurs de la fonction de répartition $F(i)$ et des niveaux de sortie j)
2. réaliser l'étalement de l'histogramme et comparer à l'égalisation.

4.5 Détection de contours

Les opérateurs $M_0 = [-1\ \ 1]$ et $M_{90} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ sont les approximations discrètes du gradient. On considère l'image $I(i,j)$ de taille 8×8 de la Fig. 4.2.

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	4	4	4	1	1	1
1	1	4	4	4	1	1	1
1	1	4	4	4	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

FIGURE 4.2 – Image originale.

1. Calculer les images filtrées I_0 et I_{90} par les deux masques M_0 et M_{90}. Pour gérer les effets de bords, on dupliquera à l'extérieur de l'image les valeurs des pixels périphériques.
2. Calculer l'amplitude du gradient $A(i,j)$
3. Calculer la direction du gradient $\Phi(i,j)$. Pour ce calcul, on utilisera la fonction arctan $\frac{y}{x}$ qui donne l'angle entre $-\pi$ et $+\pi$
4. Calculer l'image binaire $E(i,j)$ correspondant aux extrêma du gradient.
5. Faire une liste (chaîne) des points de contraste (extrêma).
6. Coder le contour avec l'algorithme de Rosenfeld & Kak en partant du point correspondant à x_{min}, y_{min}.
7. En déduire la compression obtenue par ce codage.

4.6 Calcul de Laplacien

Calculer le laplacien de l'image de la Fig. 4.3.

0	0	0	0	0	0	0	0
0	0	10	0	0	0	0	0
0	0	10	10	0	0	0	0
0	0	10	10	10	0	0	0
0	0	10	10	10	10	0	0
0	0	10	10	10	10	10	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

FIGURE 4.3 – Image originale.

4.7 Filtrage linéaire

On veut comparer deux filtres mono-dimensionnels classiquement utilisés en traitement d'image : le filtre moyenneur centré de masque $H = \frac{1}{3}[1\ \ 1\ \ 1]$ et le filtre binomial centré de masque $G = \frac{1}{4}[1\ \ 2\ \ 1]$.

1. Rappeler l'expression de la fonction de transfert $H(u)$, où u représente la fréquence spatiale normalisée ($0 \leq u \leq 0.5$).
2. Calculer la fonction de transfert $G(u)$ du filtre binomial
3. Tracer sur le graphe de la Fig. 4.4 les courbes de module des deux fonctions de transfert.

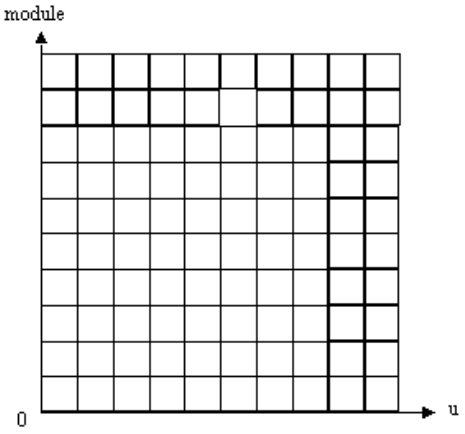

FIGURE 4.4 – Courbes de module de spectre.

4. De quel type de filtre s'agit-il ? Commenter les courbes et comparer la qualité de ces deux filtres.

On considère maintenant le filtre dérivateur : $H = \frac{1}{2}[-1 \quad 0 \quad 1]$. Etudier sa fonction de transfert. De quel type de filtre s'agit-il ?

4.8 Effet de Moiré par repliement de spectre

On considère une image I_0 formée de traits horizontaux de période 4/300ème de pouce (Fig. 4.5).

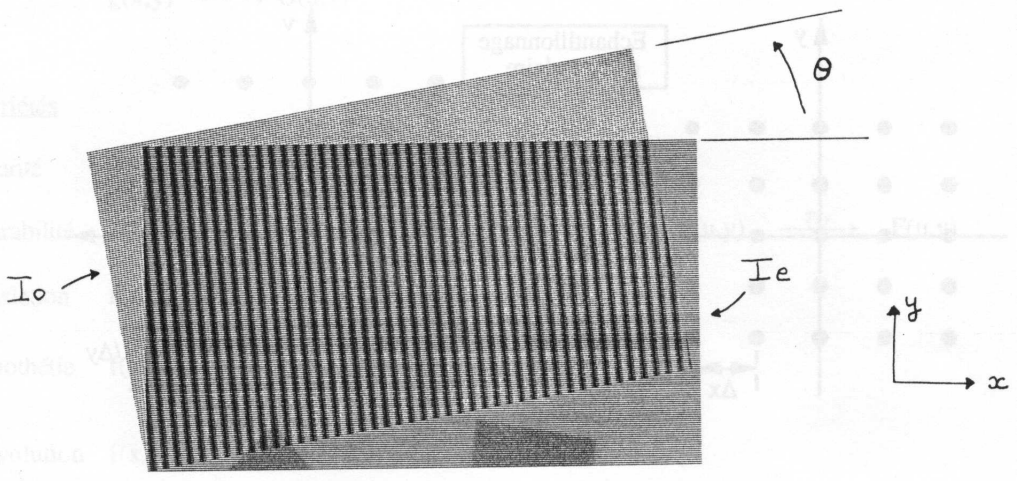

FIGURE 4.5 – Image originale et grille d'échantillonnage [10].

L'orientation de I_0 est donnée par l'angle $\theta = 10°$. Cette image est échantillonnée par I_e, identique à I_0, mais avec $\theta = 0$.

1. Représenter l'allure du spectre de I_0 (en ne tenant compte que des deux premières composantes fréquentielles)
2. Représenter le spectre de I_s, échantillonnée de I_0 par I_e.
3. En déduire les paramètres du moiré f_m (fréquence spatiale apparente) et α (angle apparent)
4. Vérifier ces résultats par mesure directe sur l'image.

NB : 1 pouce = 25.4 mm.

4.9 Transformée de Fourier

1. Démontrer que le lieu de phase nulle de la TF 2-D correspond dans le plan fréquentiel à une série de droites parallèles distantes de $d = \frac{1}{\sqrt{u^2+v^2}}$ et de direction perpendiculaire à la droite de pente $\tan\theta = \frac{v}{u}$.
2. Associer le bon spectre à chacune des images de la Fig. 4.6.
3. Calculer la transformée de Fourier d'une porte centrée $\Pi_{A,T}(x)$ de largeur T et d'amplitude A. Commenter l'importance de ce résultat.
4. Etablir le lien entre TFD et DSF dans le cas d'un signal 1-D.

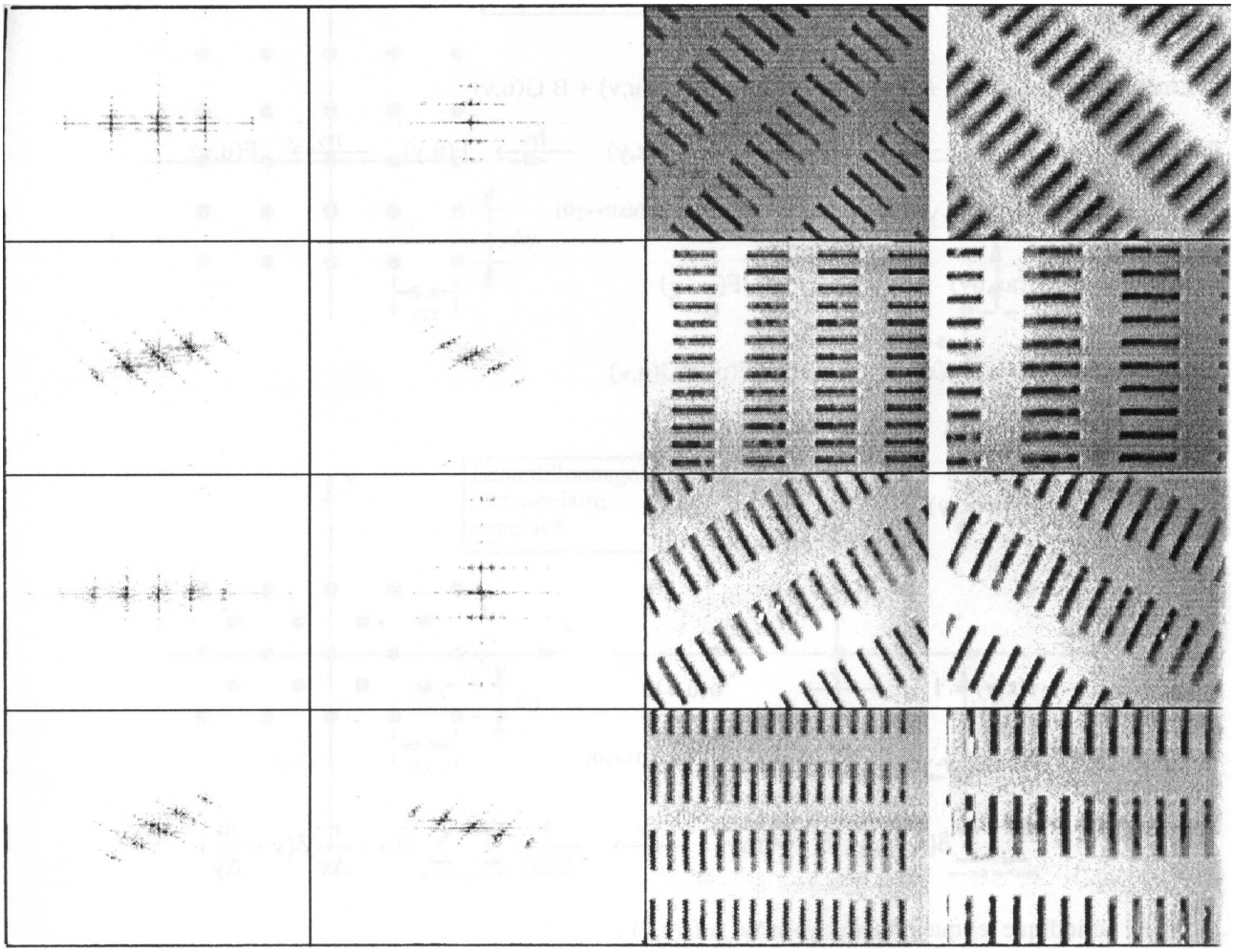

FIGURE 4.6 – A chacun sa transformée [10].

4.10 Morphologie mathématique

On considère l'image binaire de la Fig. 4.7.

Calculer les images filtrées correspondant à :

- une dilatation 4-connexe
- une dilatation 8-connexe
- une érosion 4-connexe
- une érosion 8-connexe

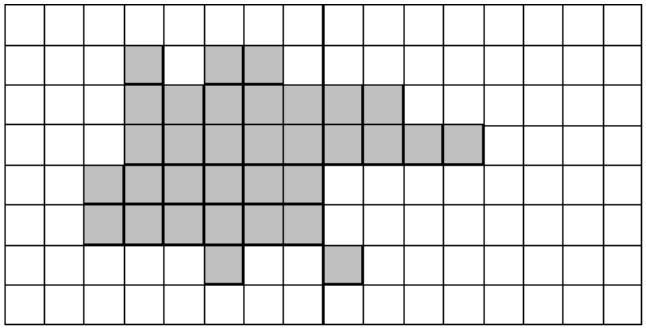

FIGURE 4.7 – Image binaire.

4.11 Poursuite de contour binaire “-2+4"

1. Coder selon le code de Freeman le contour de l'objet binaire de la Fig. 4.7, en utilisant l'algorithme de Rosenfeld & Kak, et en partant du point initial de coordonnées (4,2) (origine en haut à gauche).
2. Calculer le taux de compression obtenu grâce à ce codage.

4.12 Ouverture et fermeture

Soit la forme binaire X et l'élément structurant circulaire E présentés Fig. 4.8.

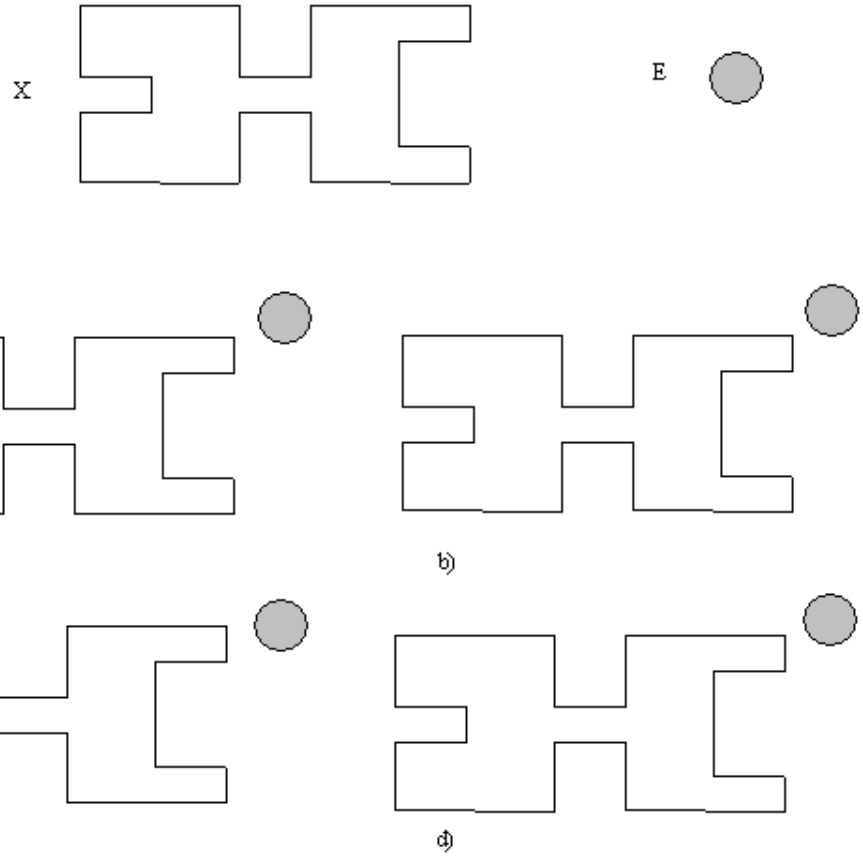

FIGURE 4.8 – Forme X et élement structurant E.

Repésenter les résultats :
– de l'érosion en a),
– de l'ouverture en b),
– de la dilatation en c),
– de la fermeture en d).

4.13 Lissage morphologique

Le lissage morphologique s'obtient par une ouverture suivie d'une fermeture. Soit la forme bruitée X et l'élément structurant E de la Fig. 4.9.

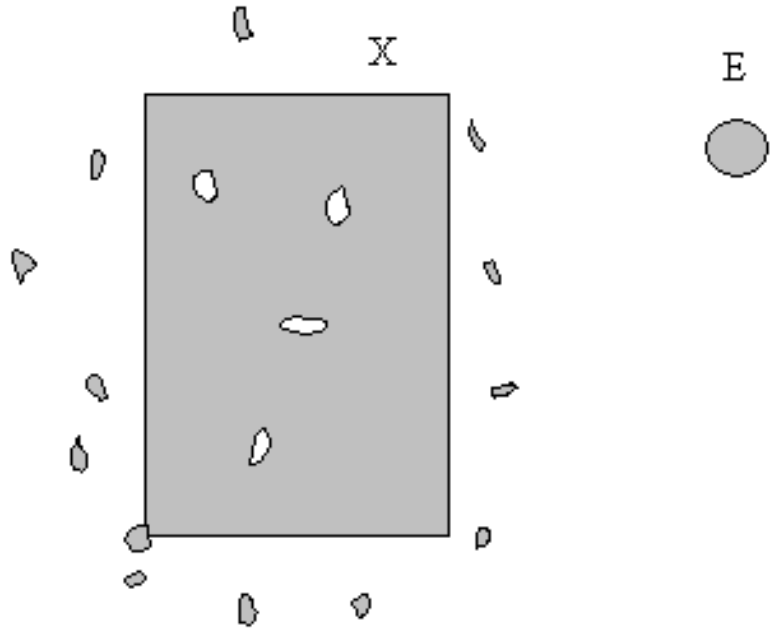

FIGURE 4.9 – Forme X d'une image binaire bruitée et élement structurant E.

Repésenter le résultat du lissage et les résultats intermédiaires (érosion, puis dilatation, puis dilatation, puis érosion).

4.14 Morphologie mathématique

On veut amincir l'objet binaire A de la Fig. 4.10 pour en extraire son squelette. Pour cela, on

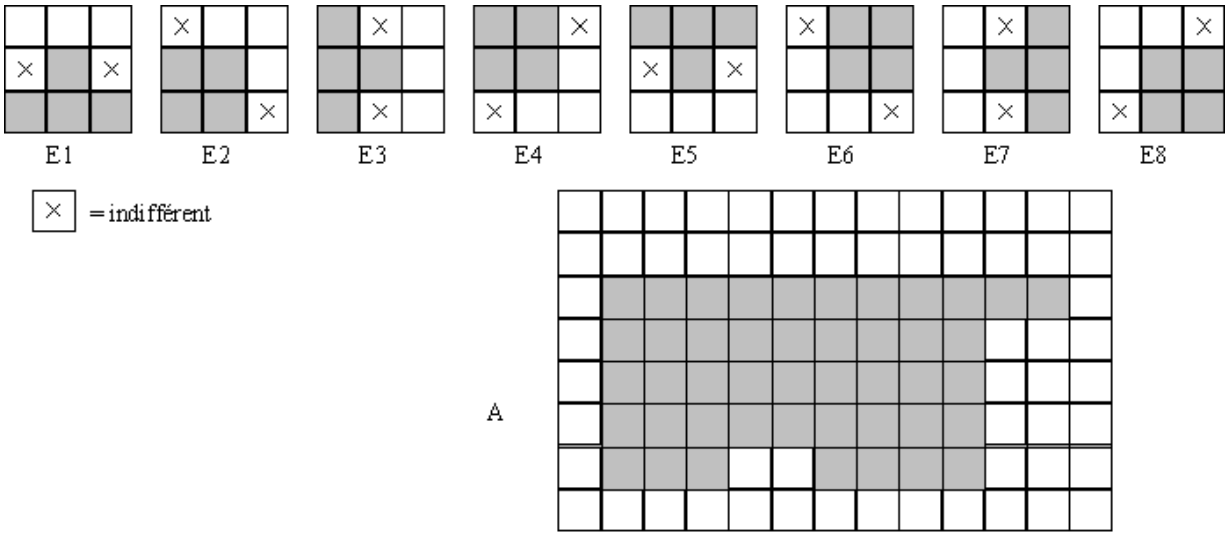

FIGURE 4.10 – Eléments structurants et objet à traiter

utilise l'ensemble d'éléments structurants $E = \{E_1, E_2, E_3, E_4, E_5, E_6, E_7, E_8\}$ constitué de versions pivotées à 45° du premier élément E_1, et on réitére jusqu'à convergence (i.e., $k = K$) la séquence

d'amincissements suivante :

$$A \otimes \{E\} = (((\cdots(((A \otimes E1) \otimes E2) \otimes E3)\cdots) \otimes E7) \otimes E8) \quad (4.5)$$
$$A_0 = A \quad (4.6)$$
$$A_k = A_{k-1} \otimes \{E\} \quad k = 1, 2, \cdots K \quad (4.7)$$

Représenter le résultat de cette opération, en détaillant les résultats intermédiaires de la procédure, sur les imagettes jointes en annexe.
On rappelle que l'amincissement $\otimes$ d'une forme A par un élément structurant B est défini par :
$A \otimes B = A - (A \oslash B)$ où $\oslash$ représente l'opération Tout-ou-Rien.
Rappel sur le tout-ou-rien $\oslash$: soit $B = (B_1, B_2)$ l'élélment structurant constitué d'une forme B_1 et d'un fond B_2, $A \oslash B = (A \ominus B_1) \cap (A^C \ominus B_2)$, c'est-à-dire que le tout-ou-rien donne l'ensemble des pixels où, simultanément, B_1 colle dans A et B_2 colle dans le complémentaire de A.

4.15 Zone aveugle

La zone aveugle de l'œil correspond au rattachement du nerf optique sur la rétine. Le test suivant permet de s'en convaincre (Fig. 4.11) :

Masquer un œil ; avec l'autre, fixer la lettre ad-hoc (D pour œil droit...). L'autre lettre disparaît pour une distance feuille-œil de l'ordre de 25 cm, correspondant à un angle d'environ $15 - 20°$ de l'axe optique.

FIGURE 4.11 – Zone aveugle.

4.16 TV couleur Secam

Le standard français Secam ('Séquentiel Couleur A Mémoire') utilise les signaux YC_rC_b définis par :

$$Y = 0.3R + 0.59V + 0.11B \quad (4.8)$$
$$C_r = -1.9(R - Y) \quad (4.9)$$
$$C_b = +1.5(B - Y) \quad (4.10)$$

Pour assurer la compatibilité avec la TV N&B, on a par ailleurs :

$$R = V = B \quad \text{pour toute la gamme des gris} \quad (4.11)$$
$$R = V = B = 1 \quad \text{pour le blanc (100\%)} \quad (4.12)$$
$$R = V = B = 0 \quad \text{pour le noir (0\%)} \quad (4.13)$$

Un générateur de mire normalisée délivre un signal constitué de barres verticales : blanc, jaune, turquoise (cyan), vert, mauve (magenta), rouge, bleu, noir. Le blanc est saturé à 100%, les autres couleurs à 75%. Ce type de mire est notamment utilisé pour le réglage des téléviseurs.

1. remplir un tableau donnant les composantes R,V,B de chaque barre de la mire
2. calculer les amplitudes des luminances de chaque couleur délivrée par le générateur de mire
3. sachant que la luminance est représentée par une tension (volts), et que l'écran est balayé ligne par ligne, dessiner l'allure temporelle du signal vidéo qui commande la cathode du tube-image
4. donner les formules de matriçage permettant de reconstituer les 3 composantes RVB à partir des signaux transmis YC_rC_b

4.17 ACP couleur

On considère une image couleur RVB caractérisée par son vecteur moyenne M et sa matrice de covariance C :

$$M = \begin{bmatrix} 69 \\ 74 \\ 76 \end{bmatrix} \tag{4.14}$$

$$C = \begin{bmatrix} 613 & 170 & 387 \\ 170 & 477 & 458 \\ 387 & 458 & 796 \end{bmatrix} \tag{4.15}$$

Appliquer l'ACP.

4.18 Détection de mouvement

On considère trois images successives d'une séquence (Fig. 4.12).

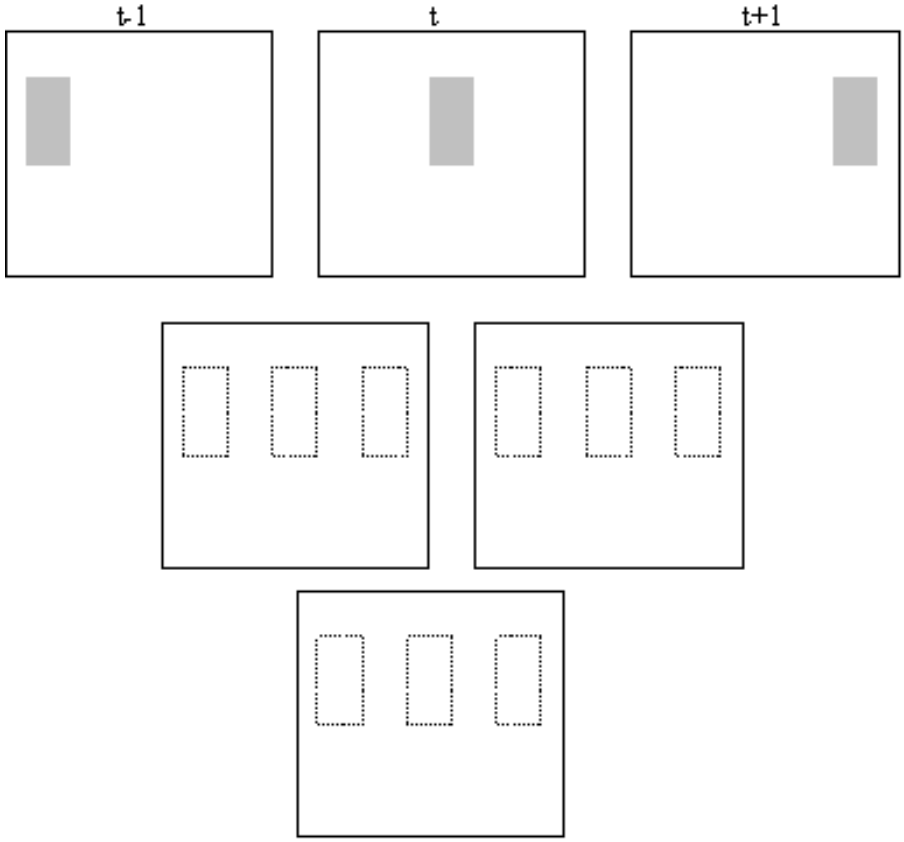

FIGURE 4.12 – Localisation par opération logique.

1. Donner les cartes des changements temporels et le résultat du ET logique
2. Que se passe-t-il en cas de recouvrement partiel ?

4.19 Détection de mouvement par étiquetage statistique contextuel

On considère un pixel courant s et son voisinage spatio-temporel formé des 8 plus proches voisins spatiaux et de 2 voisins temporels (passé et futur). Pour détecter le mouvement, on dispose des observations $o(t)$ et on suppose que les voisins sont déjà étiquetés comme indiqué sur la Fig. 4.13b.

L'étiquetage est binaire (fixe ou mobile).

Le critère de décision repose sur le calcul de l'énergie minimale.

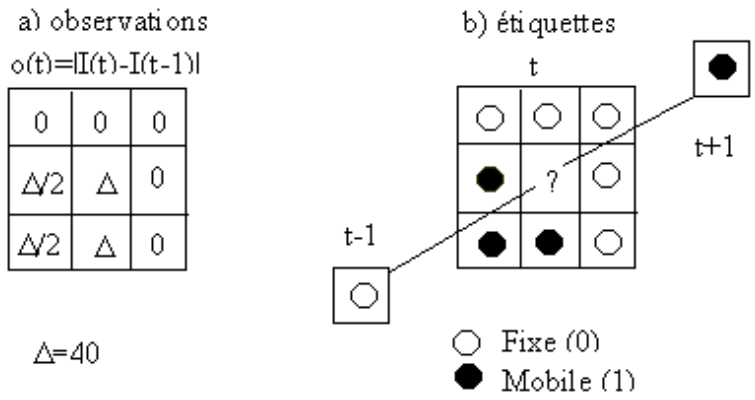

FIGURE 4.13 – a) observations b) étiquettes.

1. Calculer les deux termes d'énergie (attache aux données et modèle a priori) pour $e_s = 0$
2. Idem pour $e_s = 1$
3. En déduire quelle est la meilleure étiquette à attribuer au pixel s
4. Si l'on avait considéré uniquement la valeur de l'observation ponctuelle et opéré un seuillage binaire avec un seuil $\theta = 10$, quelle étiquette aurait-on obtenue ?
5. Conclure sur l'effet de la régularisation statistique par MRF.

Rappels sur les énergies de la modélisation markovienne :

$$U = u_m + \lambda u_a \tag{4.16}$$

où :

$$u_m = \sum_{c \in C} V_c(e_s, e_r) \tag{4.17}$$

$$u_a = \frac{1}{2\sigma^2}[o_s - \Psi(e_s)]^2 \tag{4.18}$$

avec :

$$V_c(e_s, e_r) = \begin{cases} -\beta & \text{si } e_s = e_r \\ +\beta & \text{si } e_s \neq e_r \end{cases} \tag{4.19}$$

où $\beta > 0$ prend une des trois valeurs β_s, β_p ou β_f selon la clique $c = (s, r)$ considérée (spatiale, passée ou future).

$$\Psi(e_s) = \begin{cases} 0 & \text{si } e_s = 0 \\ \alpha > 0 & \text{sinon} \end{cases} \tag{4.20}$$

Les paramètres du modèle sont :
$\beta_s = 20$, $\beta_p = 10$, $\beta_f = 30$, $\lambda = 6$.
$\alpha = 30$ est la valeur moyenne des observations non nulles.
$\sigma^2 = 300$ est la variance des observations dans le voisinage spatial.

4.20 Equation de contrainte du mouvement

L'équation de contrainte du mouvement relie les gradients spatiaux et temporels de l'image I_x, I_y, I_t aux composantes du vecteur vitesse u, v :

$$I_x u + I_y v + I_t = 0 \tag{4.21}$$

Traduire cette équation en adoptant la notation vectorielle :

$$\vec{G} = \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix} \tag{4.22}$$

$$\vec{v} = \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.23}$$

En déduire l'interprétation vectorielle de cette équation (produit scalaire).

4.21 Equation fréquentielle de contrainte du mouvement

Démontrer que l'équation ECM fréquentielle :

$$u\omega_x + v\omega_y = -\omega_t \tag{4.24}$$

s'obtient par simple transformée de Fourier de l'ECM différentielle :

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{4.25}$$

Rappel de la propriété de dérivation de la TF : si $TF[f(x)] = F(\omega)$ alors $TF[df/dx] = i\omega F(\omega)$.

4.22 Algorithme de Horn & Schunck

Pour l'implantation de l'algorithme itératif de Horn et Scunck (cf. Fig. 2.18), donner l'expression :
- des dérivées spatiales et temporelles I_x, I_y, I_t,
- des valeurs moyennes des composantes de vitesse $\overline{u}, \overline{v}$

Quelles limitations peut-on en déduire concernant l'estimation des déplacements de zones mobiles (vitesse maximale, occlusion) ?

4.23 Estimation par mise en correspondance de bloc

On considère les deux imagettes successives d'une séquence bruitée à NdG représentées sur la Fig. 4.14. On veut estimer le mouvement du bloc central 3×3 (c.f., zone mobile non nulle sur fond

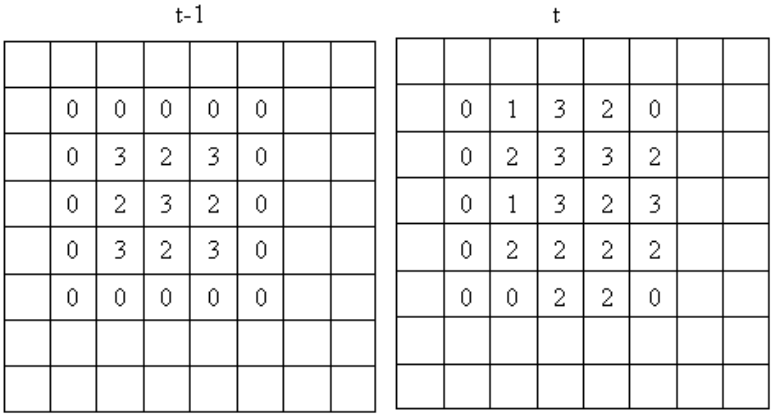

FIGURE 4.14 – Deux images successives d'une séquence bruitée.

nul à l'instant $t-1$) en utilisant comme critère d'erreur la SAD (somme des différences absolues) et en faisant une recherche exhaustive à l'intérieur d'une fenêtre de recherche centrée de taille 5×5.

1. Combien y-a-t'il de vecteurs de déplacement possibles ?
2. Calculer, pour chaque vecteur de déplacement possible, la valeur correspondante de la SAD
3. En déduire le vecteur de déplacement estimé par la méthode
4. Indiquer les limites, avantages et inconvénients de cette méthode.

4.24 Estimation d'un modèle de mouvement

On dispose des deux imagettes successives de la Fig. 4.15. On suppose qu'elles correspondent à une séquence sans bruit.

t-1

0	0	0	0	0	0		
0	0	3	2	3	0		
0	0	4	5	6	0		
0	0	3	8	3	0		
0	0	0	0	0	0		

t

0	0	0	0	0	0		
0	4	3	2	0	0		
0	3	5	3	0	0		
0	8	3	6	0	0		
0	0	0	0	0	0		

FIGURE 4.15 – Deux images successives d'une séquence.

On veut estimer le vecteur déplacement en adoptant un modèle de mouvement à trois paramètres a, b, c : $\begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} a + cy \\ b - cx \end{bmatrix}$

1. Quel est le nombre minimal de pixels à prendre en compte pour pouvoir estimer les paramètres ?
2. Calculer les paramètres de mouvement par estimation directe. On prendra l'origine de l'image au centre de l'objet (pixel $(4, 4)$).
3. Qualifier le type de mouvement subi par l'objet mobile entre les deux instants

4.25 Estimateur Robuste

L'estimateur de Geman et MacLure est défini par :

$$\rho(r) = \frac{r^2}{\sigma + r^2} \tag{4.26}$$

où σ est un coefficient de robustesse.

Tracer la courbe $\rho(r)$ et la comparer à l'estimateur quadratique.

Expliquer l'intérêt de ce type d'estimateur.

4.26 Filtre de Canny-Deriche

La réponse impulsionnelle du filtre de Canny-Deriche monodimensionnel se décompose en une partie causale et une partie anti-causale :

$$h(x) = h^+(x) + h^-(x) \tag{4.27}$$
$$h^+(x) = cxe^{-\alpha x} \text{ pour } x \geq 0 \tag{4.28}$$
$$h^-(x) = cxe^{\alpha x} \text{ pour } x \leq 0 \tag{4.29}$$

où α est un coefficient de lissage et c un coefficient de normalisation.

On montre alors que la fonction de transfert en Z s'exprime par :

$$H(z) = H^+(z) + H^-(z) \tag{4.30}$$
$$H^+(z) = c\frac{e^{-\alpha} z^{-1}}{(1 - e^{-\alpha} z^{-1})^2} \tag{4.31}$$
$$H^-(z) = c\frac{-e^{-\alpha} z}{(1 - e^{-\alpha} z)^2} \tag{4.32}$$

1. Dessiner l'allure de $h(x)$ pour $\alpha = 0.5$
2. A partir des fonctions de transfert, retrouver les équations aux différences qui servent à l'implantation numérique du filtre.
3. De quel type de filtre s'agit-il ? A quoi sert-il en traitement d'image ?

4.27 Transformée Couleur Logarithmique

La transformée logarithmique fait passer des couleurs primaires (R, G, B) aux composantes logarithmiques (L_y, L_r, L_b) définies par les équations suivantes (où $M = 256$) :

$$L_y = (R+1)^{0.3}(G+1)^{0.6}(B+1)^{0.1} - 1 \tag{4.33}$$

$$L_r = \frac{M}{2}\Big(1 + \frac{R+1}{L_y+1} - \frac{L_y+1}{R+1}\Big) \tag{4.34}$$

$$L_b = \frac{M}{2}\Big(1 + \frac{B+1}{L_y+1} - \frac{L_y+1}{B+1}\Big) \tag{4.35}$$

Cette transformation présente un intérêt en segmentation d'image mais aussi en compression. Il faut alors disposer de la transformée inverse pour décompresser l'image.

1. Préciser la dynamique des composantes (R, G, B) pour une image en vraies couleurs (codée sur 24 bits)
2. Donner les équations de la transformée inverse permettant de retrouver (R, G, B) à partir de (L_y, L_r, L_b)
3. Vérifier sur un triplet particulier (R, G, B) la validité des équations inverses
4. Donner au moins un argument en faveur de cette transformée, en rapport avec la perception des couleurs par le système visuel humain.

4.28 Filtrage linéaire

Soit le filtre symétrique défini par le masque H, qui se décompose de la manière suivante :

$$H = \begin{bmatrix} -1 & -2 & -1 \\ -2 & 12 & -2 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{bmatrix} + 2 \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{4.36}$$

1. Considérant que le filtre $L = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ est séparable sous forme $\begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 & 2 & -1 \end{bmatrix}$, montrer que sa fonction de transfert 2-D vaut :

$$L(u, v) = 4 - 2\cos 2\pi u - 2\cos 2\pi v \tag{4.37}$$

   où (u, v) sont les fréquences spatiales réduites ($\in [0; 1/2[$).
2. Représentation graphique : Tracer le module $|L(u, 0)|$ correspondant à la coupe en $v = 0$.
3. Calculer la fonction de transfert 2-D $H(u, v)$.
   N.B. Rappel : la forme générique d'une fonction de transfert 2-D est :

$$H(u, v) = \sum_k \sum_l h(k, l) e^{-i2\pi(u.k + v.l)}$$

4. Interprétation : De quel type de filtre s'agit-il ? A quoi sert-il ?

4.29 Transformée Couleur Non-Linéaire

On considère une transformée qui fait passer des couleurs primaires RGB aux composantes dites logarithmiques (L_y, L_r, L_b) définies par les équations ci-dessous (où $A = M^{0.3}$ et $D = M^{0.1}$ avec $M = 256$) :

$$L_y = (R+1)^{0.3}(G+1)^{0.6}(B+1)^{0.1} - 1 \quad (4.38)$$
$$L_r = A(R+1)^{0.7}(G+1)^{-0.6}(B+1)^{-0.1} - 1 \quad (4.39)$$
$$L_b = D(R+1)^{-0.3}(G+1)^{-0.6}(B+1)^{0.9} - 1 \quad (4.40)$$

Cette transformation présente un intérêt en segmentation d'image mais aussi en compression. Il faut alors disposer de la transformée inverse pour décompresser l'image.

1. Calculer la dynamique (i.e. valeurs mini et maxi) des composantes (L_y, L_r, L_b) pour une image RGB en vraies couleurs (codée sur 24 bits). Quel est le rôle des constantes A et D ?
2. Montrer que la transformée inverse permettant de retrouver (R, G, B) à partir de (L_y, L_r, L_b) est donnée par les équations :

$$R+1 = (L_y+1)^{a_{11}}(\frac{L_r+1}{A})^{a_{12}} \quad (4.41)$$
$$G+1 = (L_y+1)^{a_{21}}(\frac{L_r+1}{A})^{a_{22}}(\frac{L_b+1}{D})^{a_{23}} \quad (4.42)$$
$$B+1 = (L_y+1)^{a_{31}}(\frac{L_b+1}{D})^{a_{33}} \quad (4.43)$$

   On déterminera la valeur des coefficients a_{ij}.
3. Pour le triplet particulier $RGB = (200, 100, 50)$, calculer les valeurs transformées correspondantes : (L_y, L_r, L_b). Puis vérifier la validité des équations inverses trouvées ci-dessus.
4. Donner au moins un argument en faveur de cette transformée, en rapport avec la perception des couleurs par le système visuel humain.
5. Donner au moins un inconvénient de cette transformation non linéaire, par rapport aux transformées linéaires usuelles du type YUV.

4.30 Compensation de Mouvement

Soit les deux portions d'images successives d'une séquence bruitée, représentées sur la Fig. 4.16.

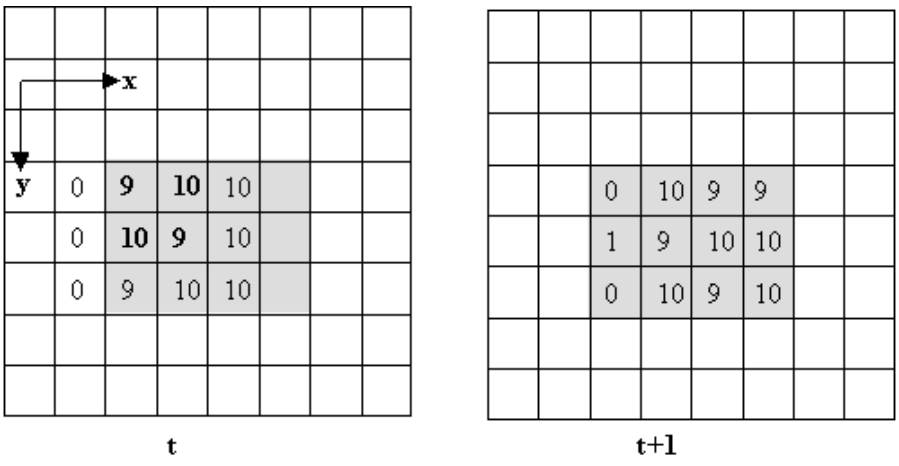

FIGURE 4.16 – Deux imagettes successives.

1. Calculer les gradients spatio-temporels moyennés des 4 pixels marqués en gras (masque $[1 \quad -1]$ et moyennage des dérivées dans le cube $2 \times 2 \times 2$, comme pour l'algorithme de Horn & Schunck).

2. Appliquer l'ECM (hypothèse de la DFD) à ces 4 pixels. En déduire la vitesse estimée.
3. Réaliser la compensation de mouvement de l'imagette grisée entre les instants t et $t+1$.
4. Calculer l'imagette d'erreur à transmettre en $t+1$.
5. Conclure sur l'intérêt de cette technique dans un codec.

4.31 Teinte du visage

Pour le suivi automatique de locuteur, on utilise souvent l'information de teinte afin de segmenter l'image en 2 régions : le visage et le fond de la scène.

On acquière alors une image couleur (*i.e.* 3 plans R, V, B chacun codé sur 8 bits). La luminance et la teinte sont définies par :

$$L = \frac{R+V+B}{3} \tag{4.44}$$

$$T = E\left[256\frac{V}{R}\right] \text{ si } R > V \tag{4.45}$$

$$T = 255 \text{ si } R \leq V \tag{4.46}$$

où $E[x]$ représente la partie entière de x.

1. Quelle est la dynamique de la teinte ?
2. Sachant qu'un visage contient très peu de bleu ($B \approx 0$), et que le plan vert est très proche du plan luminance ($V \approx L$), donner la valeur approchée de la teinte moyenne T_m caractérisant le visage.
3. Proposer une méthode simple de seuillage pour extraire de l'image la région du visage.
4. En se basant sur la courbe de visibilité de l'œil, donner un argument en faveur de l'approximation : $V \approx L$.

4.32 Transformée de Fourier

Le spectre d'une image (module au carré de la transformée de Fourier) se représente avec l'origine des fréquences spatiales au centre, les fréquences ω_x étant en abscisses et les fréquences ω_y en ordonnées. Six spectres sont représentés sur la Fig.4.18.

1. Associer le bon spectre à chacune des images de la Fig. 4.17.
2. Justifier les associations.
3. idem avec la Fig. 4.19.

4.33 Application industrielle

Proposer une technique pour restaurer un schéma électrique propre des pistes d'un circuit imprimé à partir d'une copie dégradée (Fig. 4.20).

4.34 Résolution et Contraste

La résolution R (exprimée en pixels) d'un capteur d'images et le contraste C dans une image sont définis par les équations :

$$R = 2 \times \frac{FoV}{SF} \text{ (unité : le pixel)} \tag{4.47}$$

$$C = \frac{I_B - I_D}{I_B + I_D} \tag{4.48}$$

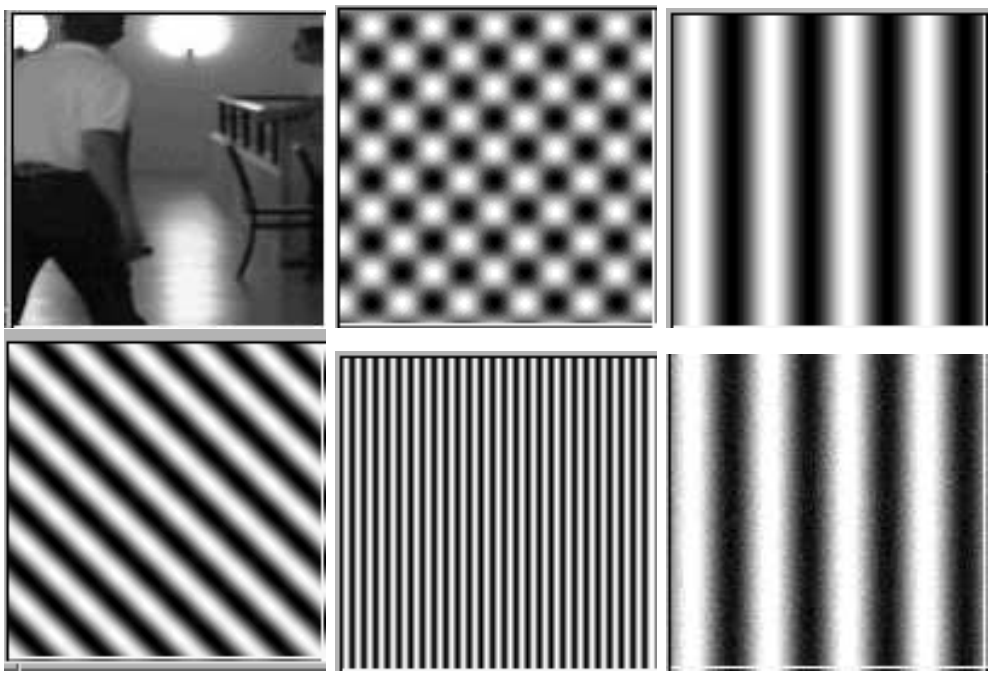

FIGURE 4.17 – 6 images.

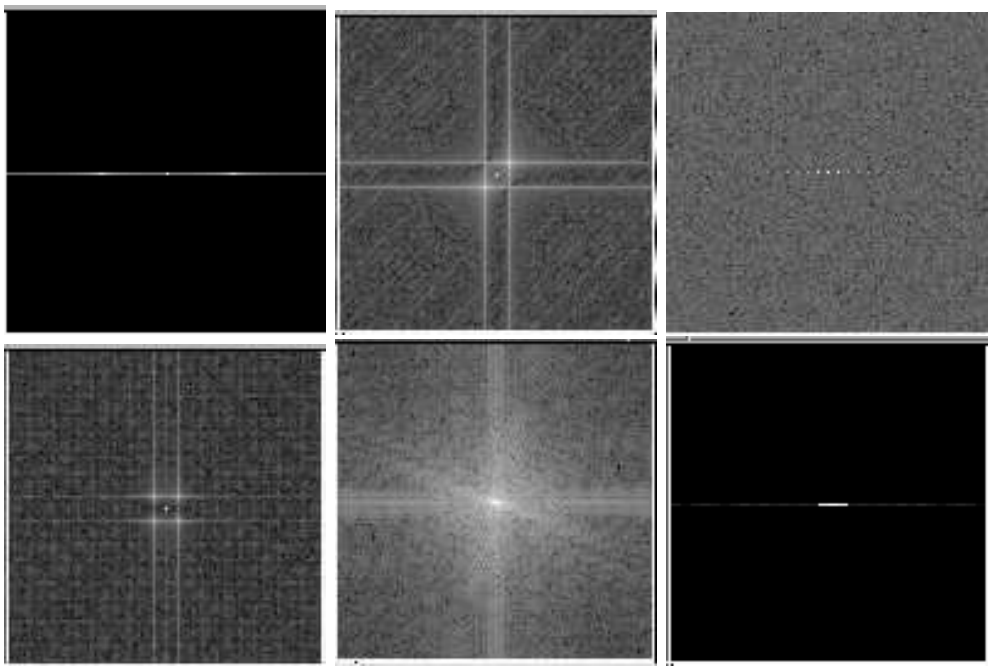

FIGURE 4.18 – 6 spectres d'images.

où FoV est le champ de vision (*Field of View*), SF la taille de la plus petite structure des objets à observer (*Smallest Feature*), et où I_B et I_D sont les intensités maximale (*Brightest*) et minimale (*Darkest*) dans l'image.

On considère les imagettes en NdG codées sur 8 bits et de taille 5×5 représentées sur la Fig. 4.21.

1. Calculer pour chaque imagette le contraste correspondant.
2. Quelles techniques d'histogrammes peut-on utiliser pour améliorer le contraste d'une image ?
3. Choisir une technique et la mettre en œuvre sur l'image la moins contrastée des deux.
   NB : Pour les calculs, arrondir les résultats à l'entier le plus proche.
4. Recalculer la valeur du contraste pour l'image ainsi améliorée. Comparer à l'image originale.
5. Conclure sur l'utilité du paramètre C. Quel est son domaine de variation ?

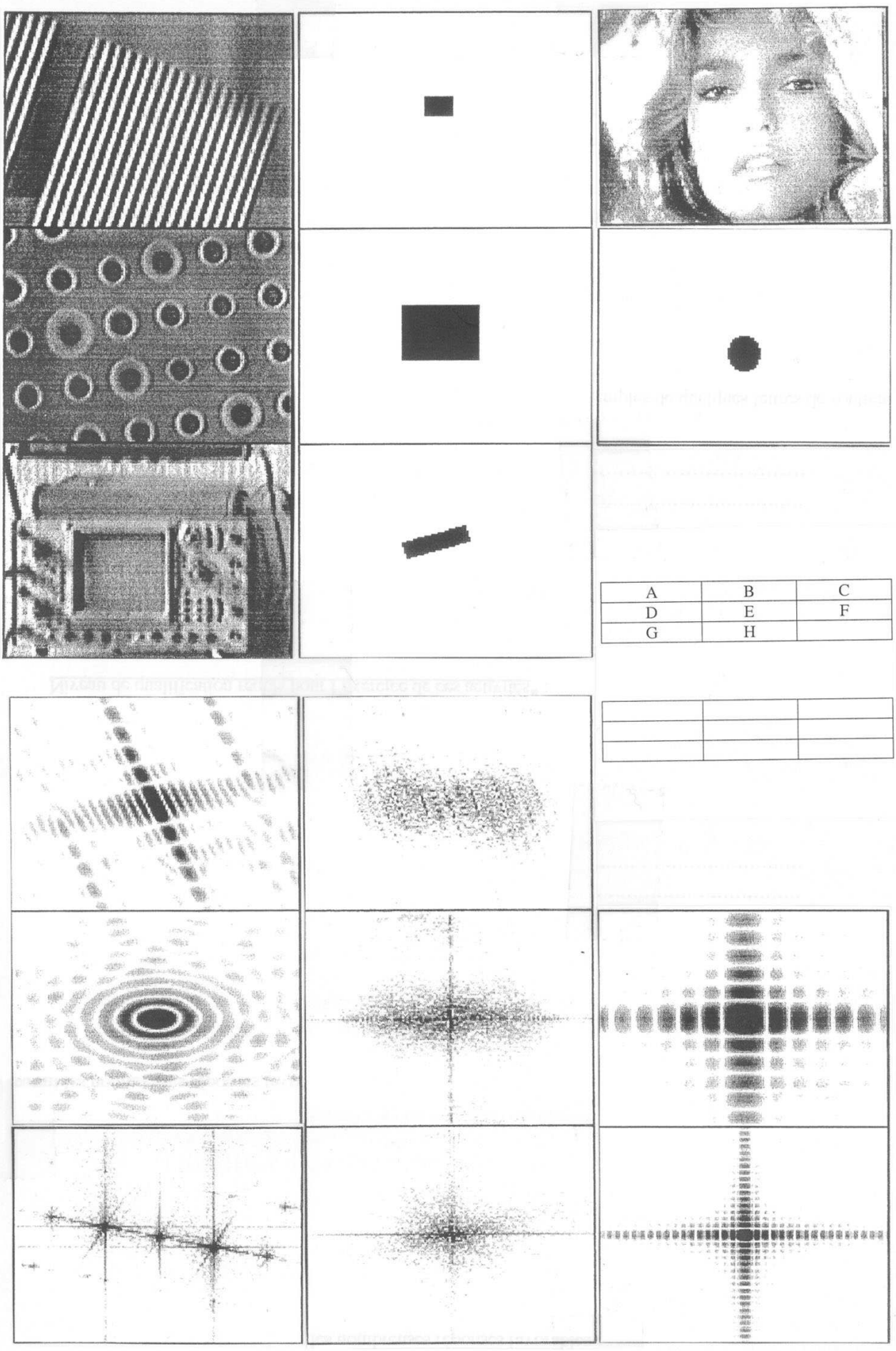

FIGURE 4.19 – 8 images, 8 spectres [10].

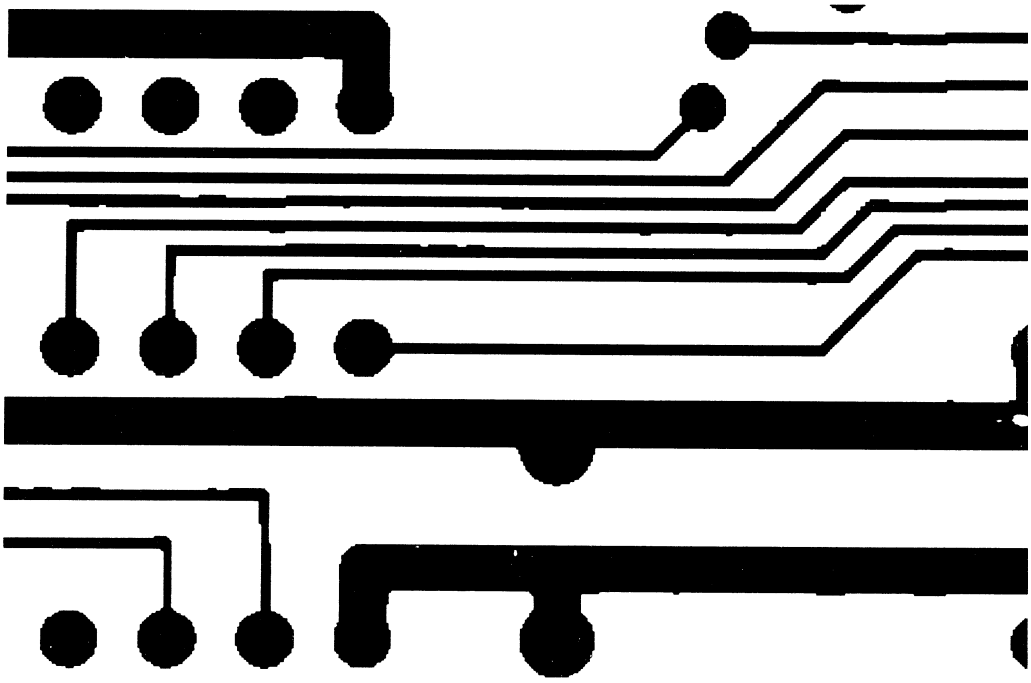

FIGURE 4.20 – Photocopie de schéma électrique.

122	122	120	125	135
122	122	110	132	132
130	120	120	125	132
132	130	120	125	130
132	130	120	122	130

123	180	193	221	247
186	183	149	173	185
255	208	89	143	136
250	210	11	78	102
246	205	15	101	66

FIGURE 4.21 – a) Imagette 1 ; b) Imagette 2.

6. Si l'on veut acquérir et traiter des images de codes-barres (étiquettes sur les produits du commerce) de $3,5cm$ de long et dont les barres verticales sont d'épaisseur minimale $0,75mm$, quelle résolution minimale doit-on respecter pour le choix du capteur ?
7. Si l'on veut inspecter des pommes de largeur maximale $8cm$ et de hauteur maximale $10cm$, ceci avec une précision de $1mm$, quelles sont les résolutions minimales en lignes et colonnes à respecter dans le choix du capteur ?
8. Si l'on dispose d'un capteur 128×128, quelles précisions (*i.e.* valeurs de SF en mm) obtient-on dans les deux applications ci-dessus 6 et 7 ?

4.35 API JMF-RTP

Etablir un dessin synoptique de la chaîne vidéo complète (acquisition, traitement, transmission, réception, restitution) implantée en TP à l'aide de l'API JMF du langage Java.

Pour chaque opération, indiquer les objets à utiliser ainsi qu'une brève description de leurs fonctionnalités respectives.

Les liens entre les objets devront être précisés afin d'avoir l'enchaînement complet permettant de mettre en œuvre une telle chaîne.

Toute autre information est bienvenue (protocoles, etc.).

4.36 Phénomène d'Aliasing

L'image analogique texturée de la Fig. 4.22a est décrite par la fonction continue de l'espace 2D :

$$s(x,y) = 1 + \cos\left[2\pi(u_0 x + v_0 y)\right],$$

avec $u_0 = 0.2$, $v_0 = 0.8$. Elle est échantillonnée avec un pas d'échantillonnage unitaire en x et en y ($\Rightarrow Fe_x = Fe_y = 1$, cf. résultat Fig. 4.22b pour taille 100×100), puis elle est filtrée (cf. résultat Fig. d) par un filtre passe-bas idéal de fréquences de coupure $u_c = v_c = \frac{1}{2}$.

1. Quelle est l'orientation des structures visibles dans l'image initiale Fig. 4.22a ? On précisera la valeur de la pente.
2. Etudier, puis dessiner sommairement, les périodicités en x et y de cette texture en calculant les périodes horizontale et verticale (en pixels/cycle). Pour cela, considérer les deux cas de figure typiques : cas $x = x_0 = cte$ d'une part, et cas $y = y_0 = cte$ d'autre part.
3. Dessiner la représentation fréquentielle (spectre) de l'image initiale dans le plan des fréquences réduites $(u,v) \in [-1;+1] \times [-1;+1]$
4. Dessiner ensuite le spectre de l'image échantillonnée correspondante Fig. 4.22b.
5. Quelle est l'orientation des structures visibles dans l'image Fig. 4.22b ? Préciser la pente.
6. Déterminer le couple de fréquences de l'image filtrée passe-bas Fig. 4.22d.
7. Comparer les textures apparentes dans les images a) et d) de la Fig. 4.22.
8. Comparer d'autre part l'image basse-fréquence analogique Fig. 4.22c et l'échantillonnée correspondante Fig. 4.22d. Commentaire de ce cas BF ?
9. Enoncer le théorème qui explique le phénomène observé.

4.37 QCM3 (1 à 4 réponses VRAI par question)

1. Soit une imagette à 4 niveaux de gris $X = \{I_1; I_2; I_3; I_4\}$ de probabilités respectives $\{p_1 = \frac{1}{2}\ ; p_2 = \frac{1}{4};\ p_3 = \frac{1}{8};\ p_4 = \frac{1}{8}\}$. Son entropie $H(X)$ exprimée en bits/NdG vaut :
   ☐ -1.75 ☐ 1.75 ☐ 2= H_{max} ☐ 4
2. La cadence vidéo en Europe vaut :
   ☐ 25 img/s ☐ 30 img/s ☐ 33 img/s ☐ 40 img/s
3. L'idempotence est vérifiée par les fonctions :
   ☐ érosion ☐ dilatation ☐ ouverture ☐ fermeture
4. Dans une image, le gradient local est un :
   ☐ vecteur parallèle au contour ☐ vecteur orthogonal au contour ☐ scalaire entier positif ☐ scalaire réel
5. Le code de Freeman est un :
   ☐ algorithme de compression d'image ☐ codage de directions ☐ format vidéo ☐ droit à l'image
6. La fonction de transfert $H(u)$ du filtre de masque spatial $M = [-1 \quad 3 \quad 1]$ est :
   ☐ $\frac{1}{3}(1 + 2\cos 2\pi u)$ ☐ $\frac{1}{2}(1 + \cos 2\pi u)$ ☐ $\exp(-\alpha u)$ ☐ une grandeur complexe
7. Si l'on transforme une image en son négatif, cela change son :
   ☐ entropie ☐ contraste ☐ histogramme ☐ spectre
8. Le masque $H = \begin{bmatrix} 1 & -3 & 1 \\ -3 & 9 & -3 \\ 1 & -3 & 1 \end{bmatrix}$ correspond à un filtre :
   ☐ dérivateur ☐ intégrateur ☐ passe-bas ☐ passe-haut
9. Dans cette liste, trouver l'intrus :
   ☐ tout ou rien ☐ chapeau haut-de-forme ☐ Sobel ☐ lissage morphologique

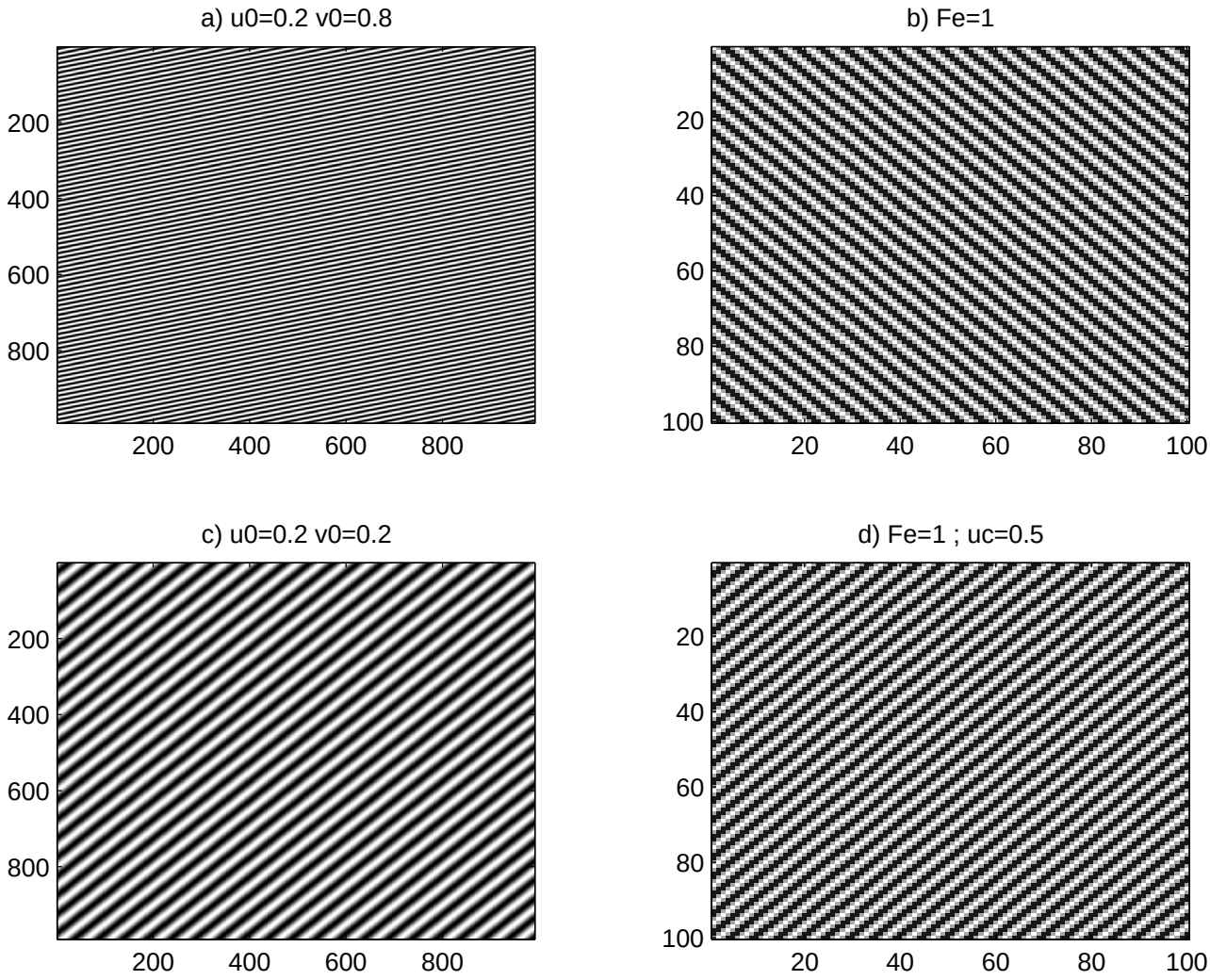

FIGURE 4.22 – a) Image HF originale ; b) Image HF échantillonnée ; c) Image BF analogique ; d) Image HF filtrée.

10. L'estimation de mouvement est un problème :
    ☐ bien posé ☐ sous-déterminé ☐ sur-déterminé ☐ d'ouverture
11. Pour détecter les contours, on peut utiliser un filtre :
    ☐ laplacien ☐ binomial-gaussien ☐ gradient ☐ moyenneur
12. Le filtre 1D de réponse impulsionnelle $h(k) = a^{|k|}$ avec $0 < a < 1$ est un filtre :
    ☐ R.I.F. ☐ R.I.I. ☐ non-linéaire ☐ instable
13. Lorsqu'on utilise la DCT pour la compression d'image au format JPEG, on traite l'image selon une approche :
    ☐ statistique ☐ spatiale ☐ fréquentielle ☐ ensembliste

4.38 QCM1 (une ou plusieurs bonnes réponses à cocher) [1]

1. Dans le secteur de la production industrielle, les 2 principaux clients du traitement d'image sont :
   (a) l'industrie métallurgique
   (b) l'industrie électrique et des semi-conducteurs
   (c) l'industrie automobile et équipementiers
   (d) l'industrie pharmaceutique, cosmétique et médicale
   (e) l'industrie agroalimentaire

2. La croissance du chiffre d'affaires de l'industrie européenne du traitement d'image est due à :
   (a) la demande en capteurs de vision et caméras
   (b) la demande en systèmes spécifiques dédiés
   (c) la demande en produits standards bon marché et faciles à utiliser
   (d) la demande de l'Amérique
   (e) la demande de la Chine
3. Pour la reconnaissance d'objets par vision industrielle, le facteur le plus limitant est actuellement :
   (a) le temps de traitement en secondes
   (b) la sensibilité du capteur
   (c) la flexibilité du système
   (d) la précision de la mesure
   (e) la qualité de l'IHM
4. Le progrès technologique le plus important pour un opérateur utilisant le traitement d'image est :
   (a) l'intégration et la compacité du système matériel, p.ex. sans PC
   (b) le pilotage intuitif de l'IHM et la convivialité du logiciel
   (c) la technologie multicore augmentant la capacité de calculs
   (d) les caméras et les capteurs performants
   (e) les techniques de communication avec l'outil de production
5. Pour la pérennité d'un sous-traitant, l'application de traitement d'image déterminante est :
   (a) la mesure sans contact
   (b) le contrôle de la qualité pour l'assurance qualité
   (c) le stockage automatique de pièces
   (d) la saisie automatique de pièces
   (e) la surveillance, par génération automatique de signaux d'arrêt
6. Pour le contrôle qualité, le traitement d'image est intéressant car il permet :
   (a) le zéro défaut
   (b) le contrôle à 100%
   (c) un mode opératoire sans contact
   (d) un contrôle très rapide
   (e) une solution très flexible
7. La morphologie mathématique est un outil du traitement d'image qui ne permet pas de :
   (a) numériser et capturer une image
   (b) faire l'analyse géométrique d'un objet
   (c) repérer, c'est-à-dire détecter, un objet
   (d) localiser un objet
   (e) piloter un robot

4.39 Filtrage médian

On donne l'imagette I_0 de taille 10×10 de la Fig. 4.23.a) dont les niveaux de gris vont de 0 à 15, et on considère le filtre médian 4-connexe (c.à.d. en forme de croix centrée sur le pixel courant Fig. 4.23.b).

NB : tous les pixels dessinés en blanc sont d'intensité égale à zéro.

Rappel : le filtrage médian consiste à trier par valeurs croissantes les intensités des pixels du voisinage considéré (ici la croix), à les ranger dans un tableau 1D (ici de taille 1×5) et à retenir la valeur médiane (c'est-à-dire celle qui est classée au milieu du tableau 1D).

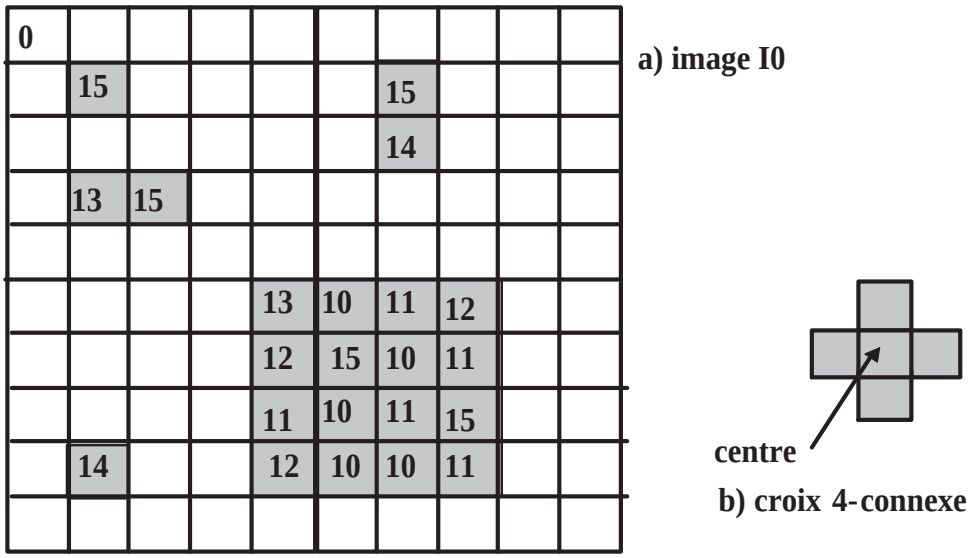

FIGURE 4.23 – a) Imagette I_0 ; b) croix 4-connexe.

1. Filtrer l'image par ce filtre médian. On appellera I_M le résultat.
   NB : pour gérer les effets de bords, on supposera que les pixels manquants sont d'intensité nulle.
2. Que deviennent les points isolés et les petits groupes de 2 pixels ?
3. Que devient l'objet carré de plus grande taille ?
4. Commenter les effets et l'intérêt de ce type de filtre, en extrapolant au cas d'une image bruitée de grande taille contenant divers objets.
5. A quelle famille de filtres appartient-il ?

4.40 Filtrage linéaire

On recupère l'image I_M résultant du filtrage médian précédent. On lui applique un filtre linéaire de masque $H = [-1 \quad 0 \quad 1]$ horizontalement.

1. De quel type de filtre s'agit-il ? Quel est son rôle ?
2. Dessiner le résultat du filtrage de I_M par H, qu'on appellera I_H.
3. On applique le même filtre sur l'image I_M, mais dans la direction verticale : c.à.d. avec le masque $V = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$. Donner le résultat du filtrage de I_M par V, qu'on appellera I_V.
4. On calcule maintenant l'image $I_G = \sqrt{I_H^2 + I_V^2}$. Dessiner le résultat.
   NB : on arrondira les racines carrées à l'entier le plus proche (cf. Tab. 4.2 en annexe).
5. On va appliquer un seuil θ (bien choisi) sur I_G. On obtiendra ainsi une image binaire qu'on notera I_B. Dessiner le résultat du seuillage après avoir justifié le choix d'une bonne valeur du seuil.
6. Commenter l'intérêt de ce traitement, en extrapolant au cas d'une image bruitée de grande taille contenant des objets également de grande taille et d'intensité assez uniforme.
7. Sans faire de calculs, comparer à ce qu'on aurait obtenu si l'on avait directement appliqué les filtres H et V sur l'image initiale I_0.

4.41 Codage

On considère désormais l'image binaire I_B résultant du seuillage précédent.

1. Indiquer le nombre N_1 de bits informatiques a priori nécessaire à son stockage sur ordinateur.

2. On veut appliquer à la forme binaire obtenue l'algorithme de codage directionnel de Freeman, en partant du pixel le plus en haut et à gauche appartenant à la forme détectée. Préciser les coordonnées de ce pixel initial, l'origine de l'image étant prise en haut à gauche. Combien de bits sont nécessaires pour coder l'information sur la position de ce pixel dans l'image ?
3. Donner ensuite le code complet obtenu (point initial + codage des directions successives).
4. Calculer le nombre N_2 de bits nécessaire à la transmission de cette information.
5. Comparer N_2 à N_1 et conclure sur l'intérêt de ce codage.
6. Calculer l'entropie $H(B)$ (en bits/pixel) de la source d'information que constitue l'image I_B, ainsi que sa redondance définie par : $R = 1 - \frac{H(B)}{H_{max}}$.
7. Si l'on mettait en œuvre un codage entropique de la source du type Shannon-Fano ou Huffman, combien de bits d'information pourraient suffire à la transmission de I_B ?
8. On note N_3 ce nombre de bits. Comparer N_3 à N_2. Quel est le codage le plus intéressant de cette image I_B : codage directionnel ou codage entropique ?

TABLE 4.2 – Annexe : tableau de valeurs numériques

$\sqrt{265} \approx 16$	$\sqrt{221} \approx 15$	$\sqrt{125} \approx 11$	$\sqrt{101} \approx 10$
$\log_2(2) = 1$	$\log_2(3) = 1.58$	$\log_2(5) = 2.32$	$\log_2(7) = 2.80$

4.42 Résolution d'une caméra linéaire

Pour inspecter un matériau plan en défilement continu (production de tissu, papier, tôles, Fig. 4.24), on utilise une caméra monochrome linéaire, c'est-à-dire constituée d'une seule rangée de photosites ou barrette de pixels. Le nombre de pixels de la barrette définit la résolution transversale de la caméra. Une grande résolution permet des contrôles dimensionnels précis. Pour déterminer la résolution transversale R, on divise la dimension du champ de visée Fov (*Field of View*) par la moitié de la taille x du plus petit détail à détecter (car pour une bonne perception visuelle, il faut que le plus petit détail soit vu par au moins deux pixels voisins).

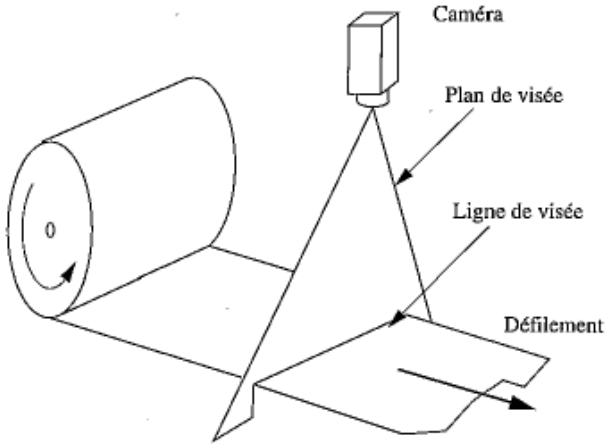

FIGURE 4.24 – Inspection d'un matériau plan en défilement [13]

La résolution longitudinale est fonction de la vitesse de déplacement relative entre l'objet observé et la caméra, et de la fréquence d'acquisition des lignes successives. Pour déterminer la fréquence d'acquisition F_a, on divise la vitesse relative V du déplacement de l'objet par rapport à la caméra par la moitié de la dimension longitudinale y du plus petit détail à détecter (car il est conseillé que le plus petit détail soit au moins vu sur 2 lignes successives acquises par la caméra linéaire). Evidemment, plus cette fréquence est élevée, plus le temps d'intégration est réduit et donc plus l'éclairage de la scène doit être intense.

Enfin, si l'objet est suffisamment éloigné de la caméra (d'une distance D), la connaissance de la largeur du champ de visée et de la taille l du capteur permet de déterminer la distance focale f de l'objectif, appelée optique de la caméra :

$$R = \frac{Fov}{x/2} \quad ; \quad F_a = \frac{V}{y/2} \quad ; \quad f = \frac{l.D}{Fov}$$

Application : on désire détecter des points de corrosion de taille supérieure ou égale à 1×1mm sur une tôle de un mètre de large défilant sous une caméra CCD linéaire de contrôle à la vitesse de 10 mètres par minute. La distance entre tôle et caméra vaut $D = 1m50$ et la longueur de la barrette CCD est $l = 10mm$.

Déterminer les caractéristiques de la caméra à utiliser :

1. la résolution transversale minimale (en pixels)
2. la fréquence d'acquisition minimale (en Hz)
3. la focale de l'objectif adaptée aux conditions de prise de vue (en mm).
4. la taille d'un photosite élémentaire du capteur.

4.43 Aberration chromatique d'un capteur mono-CCD

Dans une caméra mono-CCD, les couleurs sont recueillies par la technique du filtrage en mosaïque : des filtres colorés sont placés sur la surface sensible du capteur de telle sorte que chaque pixel ne perçoit que l'une des trois composantes primaires Rouge, Vert ou Bleu (Fig. 4.25). Il est donc impossible d'obtenir simultanément les valeurs des 3 composantes colorimétriques en chaque pixel : les réponses de 9 pixels voisins sont nécessaires pour connaître la couleur d'un point de l'image, ce qui entraîne une perte de résolution et des aberrations chromatiques lors de transitions brusques.

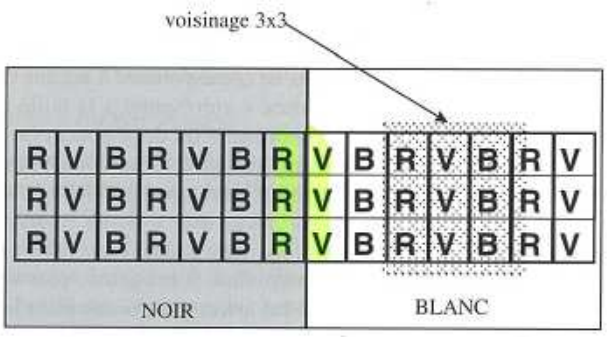

FIGURE 4.25 – Structure d'un capteur mono-CDD

On considère la zone d'image avec la transition verticale Noir/Blanc de la Fig. 4.25. Sachant que la couleur de chaque pixel est évaluée par analyse de son voisinage 3×3, déterminer les couleurs perçues au niveau de la transition :

1. sur la colonne à gauche de la transition Noir/Blanc
2. sur la colonne à droite de la transition Noir/Blanc

4.44 Seuillage d'image

On considère l'image de micrographie de cellule de la Fig. 4.26a. Son histogramme est donné en b), avec deux seuils positionnés à 180 et 200.

1. Les images c) et d) sont des versions binarisées obtenues par seuillage de l'histogramme. Indiquer quel seuil donne quelle image.
2. Si vous deviez choisir un seuil pour un rendu optimal des cellules, où positionneriez-vous le seuil ? Justifier.
3. Si l'on ne veut recupérer que la cellule noire, quelle valeur de seuil choisir ?

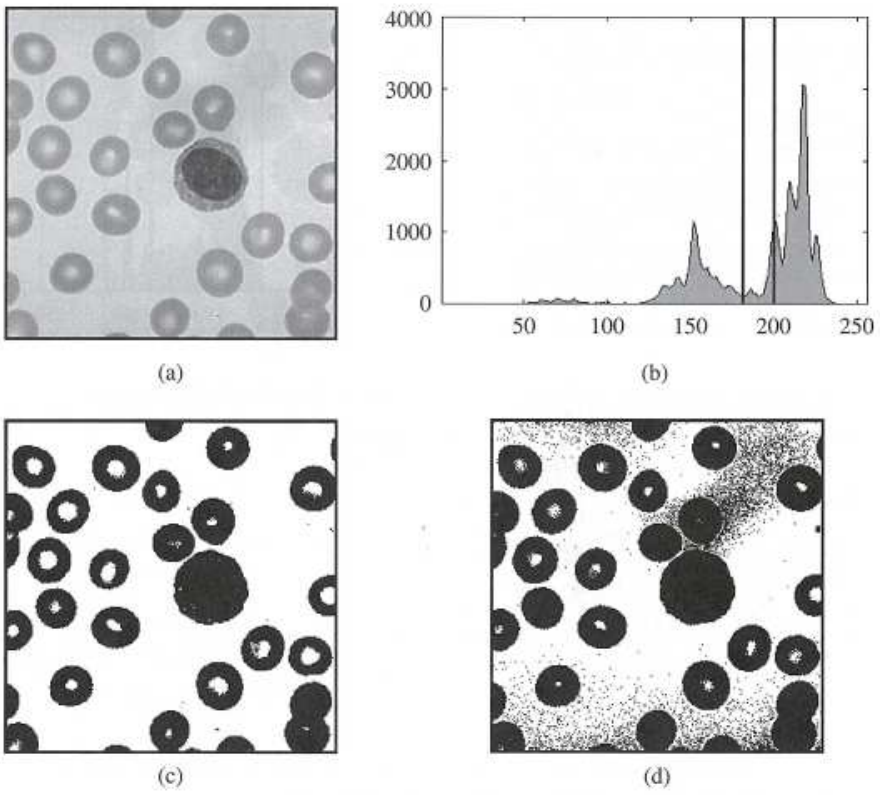

FIGURE 4.26 – Micrographie de cellules : a) image originale ; b) histogramme montrant deux seuils (à 180 et 200) ; c) et d) résultats de binarisation [70]

4.45 Codage de chaîne

Le codage de chaîne est une représentation efficace d'images binaires contenant des contours.

1. En utilisant l'algorithme de Rosenfeld et Kak avec le code de Freeman (à 8 directions), coder le contour présent dans l'imagette de taille 11×11 représentée sur la Fig. 4.27.
2. Combien de bits sont nécessaires à ce codage ? Justifier le résultat.
3. Combien de bits seraient nécessaires si l'on voulait transmettre toutes les coordonnées de tous les points-contours ?
4. En déduire le taux de compression obtenu grâce au codage chaîné.
5. Dans le cas d'une image de dimension 512×512, calculer le gain en taille mémoire qu'on obtient grâce au codage chaîné des contours, par rapport à la représentation basique qui consisterait à stocker les coordonnées de chaque point de contour.

4.46 Binarisation par Seuillage Entropique

Pour binariser les observations issues d'une source discrète (typiquement une image) entachée d'un bruit blanc additif majoritaire sur le support, on peut utiliser la technique du seuillage entropique [6] qui consiste à calculer l'entropie H et à choisir un seuil θ défini par :

$$H = -\sum_{i=1}^{M} p_i \log_2 p_i \tag{4.49}$$

$$\theta = 2^H \tag{4.50}$$

où :

- M est le nombre de valeurs distinctes prises par les observations
- p_i est la probabilité pour qu'une observation o_s prenne la valeur i en n'importe quel site-pixel s.

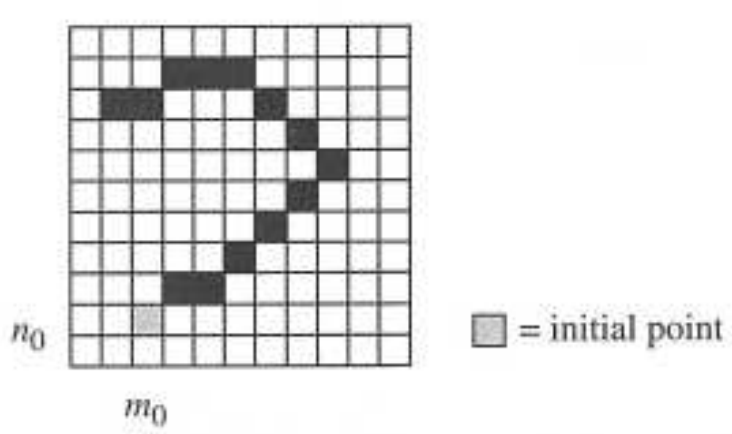

FIGURE 4.27 – Contour avec point initial de coordonnées (n_0, m_0)

– enfin $\log_2(x)$ dénote le logarithme binaire. On rappelle que $\log_2(x) = \frac{\ln x}{\ln 2}$ et que $\log_2(2^n) = n$. On considère l'imagette de la Fig. 4.28 comportant $M = 5$ NdG distincts (qui pourrait représenter par exemple une différence temporelle d'images en valeur absolue, comme en détection de mouvement, ou un module de gradient spatial, comme en détection de contours) :

0	1	0	1	1	0	0	0
0	0	3	2	2	1	0	0
0	1	4	4	4	1	0	0
0	0	4	3	4	1	0	0
0	1	4	4	4	1	0	0
0	1	3	2	3	0	0	0
0	0	1	1	2	1	0	0
0	1	1	0	1	0	0	0

FIGURE 4.28 – Imagette d'observations bruitée.

1. Donner les probabilités des observations et dessiner l'histogramme.
2. Calculer l'entropie H.
3. **Question facultative :** Comparer à l'entropie maximale H_{max} qu'on peut obtenir avec 5 NdG. En déduire la redondance R.
4. Calculer le seuil entropique θ correspondant à H.
5. Appliquer le seuillage à l'imagette (en prenant $o_s > \theta$). Commenter le résultat de la binarisation et son intérêt en détection (de mouvement ou de contour).

4.47 Filtrage linéaire

On considère les contours verticaux en échelon et en rampe de la Fig. 4.29-a) où : $b > a$ et $c = \frac{a+b}{2}$. On notera $h = b - a$.

1. Soit le masque de filtrage $M_1 = [-1 \quad 1]$ (coefficient du pixel courant à droite). Dessiner le résultat du filtrage d'une ligne pour les deux types de contours (échelon et rampe).

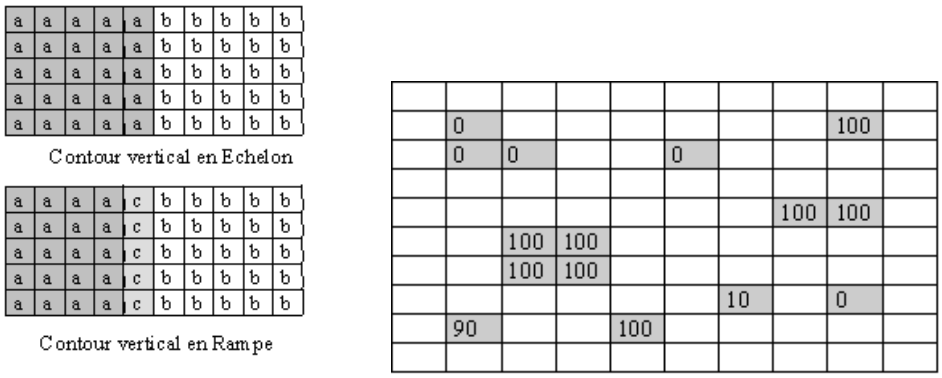

FIGURE 4.29 – a) Contours verticaux ; b) Imagette.

2. Même question pour le masque de filtrage $M_2 = [-1 \quad 0 \quad 1]$ (coefficient du pixel courant au centre).
3. Comparer ces deux filtres (avantages et inconvénients) sur les deux types de contours : localisation, sensibilité au bruit ...

On considère l'imagette de la Fig. 4.29-b), où tous les pixels du fond (en blanc) valent 50. On la filtre avec le masque de filtrage $M = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ et on ne retient que les pixels dont le résultat R du filtrage vérifie : $|R| > \theta$ où $\theta = 45$ est un seuil fixé.

1. Représenter le résultat du filtrage (on ne traitera pas les bords de l'imagette).
2. Représenter le résultat du seuillage (dessiner en noir les pixels retenus par la procédure).
3. Quel est le rôle de ce filtre (effet sur les zones uniformes, les groupes de pixels, les pixels isolés) ?

4.48 Morphologie mathématique

On considère la forme binaire A et l'élément structurant E de la Fig. 4.30. Les pixels du fond valent

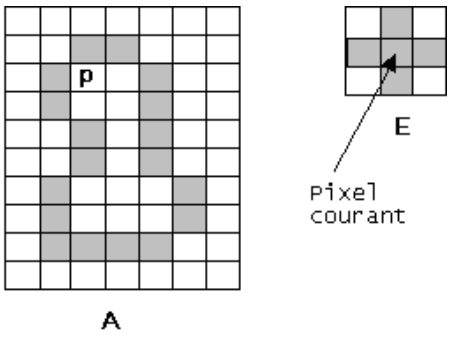

FIGURE 4.30 – a) Forme binaire (en gris) correspondant par exemple à la version scannée d'un "a" manuscrit ; b) Elément stucturant (croix centrée).

0 et ceux de la forme A valent 1. Soit un point initial intérieur à la forme (point p sur la figure). On lui attribue la valeur 1 et on applique la procédure suivante de dilatations itérées :

$$X_k = (X_{k-1} \oplus E) \cap A^c \qquad k = 1, 2, 3, ... \tag{4.51}$$

où A^c représente le complémentaire de A et $X_0 = \{p\}$.

1. Dessiner d'abord le complémentaire de A.
2. Représenter les résultats X_k à chaque itération jusqu'à convergence.

3. Quel est le nombre N d'itérations jusqu'à la convergence (quand $X_{k+1} = X_k$, $\forall k \geq N$) ?
4. Le résultat est l'ensemble constitué de l'union de X_N et de A. Quel est le rôle de ce traitement ?

4.49 Analyse de mouvement

Soit une séquence d'images en niveaux de gris $I(t)$ et une image de référence I_{ref}. On suppose que la séquence comporte un objet mobile (carré de taille 5×5) d'intensité lumineuse plus forte que le fond, et se déplaçant dans la direction sud-est à la vitesse $v = (v_x, v_y)$=(2 pix/img, 1 pix/img). L'image de référence est la première image de la séquence correspondant à l'instant initial $t_0 = 0$ (on suppose qu'elle contient tout l'objet mobile, cf. Fig 4.31). L'origine spatiale de l'image est en haut à gauche.

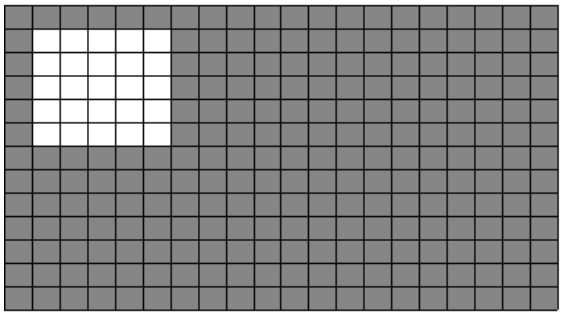

FIGURE 4.31 – Image de référence I_{ref} (à $t_0 = 0$).

L'image des différences positives accumulées (PADI en anglais) est définie à l'instant t en chaque site-pixel $s = (x, y)$ de la façon suivante :

$$PADI(s,t) = PADI(s,t-1) + d(s,t) \quad (4.52)$$

$$\text{où } d(s,t) = \begin{cases} +1 & \text{si } I_{ref}(s) - I(s,t) > \theta > 0 \\ 0 & \text{sinon} \end{cases} \quad (4.53)$$

En chaque pixel, un compteur est donc incrémenté à chaque fois qu'une **différence positive** est détectée entre la référence et le pixel courant (θ symbolisant la valeur de seuil utilisée). Au départ ($t_0 = 0$), on a : $\forall s$, $PADI(s, t_0) = 0$

1. Représenter les 5 imagettes PADI correspondant aux 5 instants successifs $t = 1$ à $t = 5$.
2. Commenter ce qui se passe à partir de $t = 3$.
3. Expliquer l'intérêt de ce traitement (en détection d'objet mobile et estimation de vitesse) ?

4.50 Filtre Médian

On donne ci-dessous l'algorithme dit du filtrage médian, qui lui-même repose sur un algorithme de tri (classement des intensités des pixels dans un voisinage donné).

```
Balayage vidéo de l'image (x,y) (sauf les bords)
  k=0 (indice du tableau monodimensionnel intermédiaire tab)
  Balayage vidéo du voisinage 3x3: Coordonnées Relatives (i,j) où i,j varient dans {-1;0;1}
    - calcul des coordonnées réelles du pixel voisin (x+i,y+j)
    - tab[k]=I(x+i,y+j) /*entrée des valeurs dans un tableau*/
    - k++
  Classement des valeurs du tableau tab (algo du tri à bulles)
  Med(x,y)=tab[5] /*valeur médiane du tableau*/
```

On rappelle le principe du tri à bulles à l'aide de l'algorithme ci-après :

```
Balayer le tableau, en considérant l'élément courant et l'élement suivant
  Si l'élement courant est supérieur à l'élément suivant
    -transposer l'élément courant et l'élément suivant
Réitérer le balayage s'il y a une transposition au cours du balayage précédent.
```

On considère d'autre part l'imagette en NdG de la Fig. 4.32 comportant quelques objets de forme différente (rectangulaire, linéaire, ponctuelle).

0	0						
0	100		10	10	10	10	
		0	10	0	10	10	
		0	10	10	10	10	
	50	0					
	20	20	20	20	20		

FIGURE 4.32 – Imagette en NdG (les pixels non renseignés valent zéro par défaut).

1. Préciser la taille du tableau 1D intermédiaire qui servira au tri ?
2. Dérouler l'algorithme du filtre médian sur l'imagette fournie et dessiner l'imagette résultat du filtrage.
3. Interpréter le rôle du filtre en détection (action sur les trous, les points isolés, les traits horizontaux, les angles etc.)
4. Que donnerait ce filtrage sur un trait vertical ?
5. Quelle est la taille des trous et des taches éliminables dans une image bruitée ? De quoi est-elle fonction ?
6. A quelle famille appartient ce type de filtre ?

4.51 Codage

Soit l'imagette bruitée de taille 10×10 de la Fig. 4.33 comportant 8 niveaux de gris (NdG) du noir « = 0 » au blanc « = 7 ».

1. Combien de bits informatiques sont nécessaires pour coder un pixel ? Quelle est la taille sur disque de l'image ?
2. Calculer les probabilités p_i de chaque niveau de gris (en laissant les p_i sous forme fractionnaire). En déduire l'entropie H de la source d'information correspondant à cette image.

   NB : en l'absence de calculatrice, on se réfèrera au Tab. 4.2 p.84 du polycopié pour les valeurs des logarithmes binaires, ainsi qu'aux propriétés usuelles du logarithme : $\log(x/y) = \log(x) - \log(y)$; $\log(x^a) = a \log(x)$ etc.
3. Quelle serait l'entropie maximale H_{max} de cette source ? En déduire la redondance R.
4. Tracer l'histogramme de l'image. Proposer une valeur de seuil θ pour binariser cette image en 2 niveaux : Noir et Blanc.
5. Quelle sera alors la taille sur disque nécessaire pour stocker l'image binaire ?

1	0	0	0	0	0	0	1	0	1
0	0	1	0	1	1	0	0	1	0
1	1	6	0	0	0	1	0	0	0
0	2	7	5	0	0	0	2	0	1
1	0	5	5	6	1	0	0	0	0
0	0	5	4	4	5	1	2	1	0
0	1	6	4	3	4	5	0	0	1
1	0	5	6	5	6	5	5	1	0
0	0	0	1	0	0	2	1	0	0
1	0	1	0	1	0	0	0	0	1

FIGURE 4.33 – Imagette bruitée.

6. Appliquer l'algorithme de Rosenfeld et Kak pour suivre le contour binaire obtenu (en partant du pixel de l'objet le plus en haut et à gauche).
7. Calculer le nombre total de bits nécessaires au codage du contour, et donc la taille mémoire sur disque pour stocker l'image binaire.
8. En comparant à l'image initiale, en déduire le taux de compression obtenu grâce à ce codage.

4.52 Filtrage Linéaire : Filtre de Sobel

On filtre l'imagette présentée Fig. 4.33 avec le filtre de masque :

$$M = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

1. Exprimer ce filtre sous forme séparable en 2 filtres 1-D selon les lignes et colonnes.
2. Calculer et tracer les fonctions de transfert des 2 filtres 1-D qui le constituent (en module uniquement, et en fonction de la fréquence réduite u).
3. Commenter le type et le rôle de ces 2 filtres. Quel est l'intérêt de leur association dans le filtre M ?
4. Calculer l'imagette filtrée par le filtre de masque M.
   NB : Ne pas traiter les rangées de pixels sur les bords de l'image (mettre simplement à zéro).
5. Proposer une valeur de seuil pour binariser le résultat. Que détecte-t-on ?
6. Obtient-on le contour horizontal ? Sinon, proposer une solution pour détecter aussi ce contour.

4.53 Morphologie Mathématique : Gradient Morphologique

On veut comparer le filtre linéaire du § 4.52 au filtre non-linéaire correspondant au gradient morphologique avec l'élément structurant fait d'un segment horizontal centré de largeur 3 pixels.

1. Rappeler l'expression du gradient morphologique $\overrightarrow{G}$ pour images en NdG.
   NB : dans la suite, on ne traite que la composante horizontale G_x du gradient.
2. Calculer la dilatée, l'érodée et enfin le résultat du gradient morphologique de l'imagette Fig. 4.33.
3. Proposer une valeur de seuil pour binariser le résultat.
4. Comparer au résultat du filtre linéaire et commenter.

4.54 Codage entropique de Huffman

Soit l'image de taille 9×11 de la Fig. 4.34 codée sur 4 niveaux de gris seulement (blanc, gris clair, gris foncé, noir). NB : Elle contient 99 pixels, mais pour simplifier les calculs, on pourra arrondir à 100 le nombre de pixels.

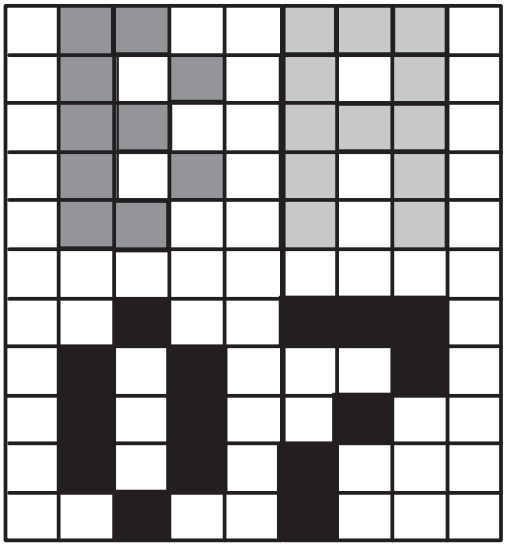

FIGURE 4.34 – Image affichant 4 caractères ASCII.

1. A priori, que vaut l'entropie maximale H_{max} de cette source d'information (en bits/pixel) ?
2. Calculer l'entropie H_0 et la redondance R_0 dans l'image initiale (avant codage entropique).
3. Réaliser le codage entropique (de Huffman ou Shannon-Fano) de cette image. Pour cela, construire un tableau contenant les 4 messages M_i (*i.e.* 4 niveaux de gris), leurs probabilités P_i (approximées par leur fréquence d'apparition dans l'image), les mots-codes C_i et les longueurs de mots correspondantes L_i.
4. Calculer la longueur moyenne L_{moy} d'un mot-code. En déduire le taux de compression τ obtenu par ce codage entropique à longueur variable.
5. Calculer la probabilité des symboles "0" et "1" en sortie du codeur
6. En déduire l'entropie H_c et la redondance R_c après codage. Comparer aux valeurs avant codage et commenter le résultat obtenu.
7. Décoder le message véhiculé par cette image source d'information visuelle qui affiche quatre caractères parmi des chiffres et des lettres. Calculer la quantité d'information moyenne (entropie) de ce message textuel, en supposant équiprobables tous les caractères (les 26 lettres de l'alphabet et les 10 chiffres). Commentaire ? (comparer avec les premières questions sur H_{max} et H_0).

NB : Voir le tableau des logarithmes binaires fourni par ailleurs (Tab. 4.3) si l'on veut travailler sans calculatrice.

Annexe

On donne le tableau des logarithmes binaires (pour ceux qui n'auraient pas de calculatrice) :

TABLE 4.3 – Table de Logarithmes à Base 2

x	0	1	2	3	4	5	6	7	8	9	10	11	12
$\log_2(x)$	$-\infty$	0	1	1.585	2	2.322	2.585	2.807	3	3.17	3.322	3.459	3.585

4.55 Effet 2D d'un filtre RIF 1D

Soit le filtre RIF centré de masque $H = \frac{1}{5}[1\ \ 1\ \ 1\ \ 1\ \ 1]$ et les images de la Fig. 4.35.

0														0
			0	0	0	0	0	0	0	0	0			
			0	100	100	100	100	100	100	100	0			
			0	100	100	100	100	100	100	100	0			
			0	100	100	100	100	100	100	100	0			
			0	100	100	100	100	100	100	100	0			
			0	0	0	0	0	0	0	0	0	0		
0														0

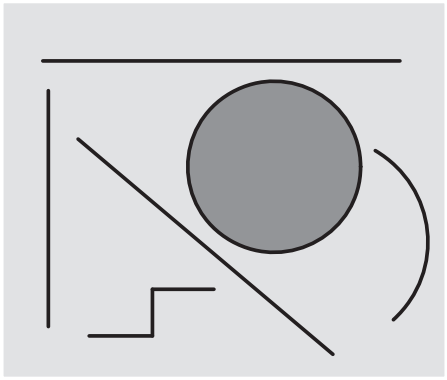

FIGURE 4.35 – Image a) à gauche (tous les pixels non spécifiés valent zéro). Image b) à droite

1. Représenter le résultat du filtrage de l'image de la Fig. 4.35a par ce filtre.
2. Calculer la fonction de transfert $H(u)$ du filtre, où $u = \nu/Fe$ est la fréquence réduite telle que $0 \leq u \leq 0.5$ si l'on ne considère que les féquences positives respectant le théorème de Shannon.
3. Rappeler à cette occasion ce que stipule ce théorème et quels phénomènes apparaissent si on ne le respecte pas.
4. Représenter graphiquement l'allure de la courbe de module $|H(u)|$ en fonction de u.
5. De quel type de filtre s'agit-il : passe-bas, passe-haut, passe-bande, coupe-bande, réjecteur, passe-tout... ? Commenter ses qualités et défauts.
6. Quel est le nom de l'effet principal produit par ce filtre : flou isotrope, détecteur de contour, rehaussement de contraste, lissage de bruit, bougé horizontal, bougé vertical... ?
7. Que donnera le filtre sur l'image de la Fig. 4.35b ? Dessiner le résultat.

4.56 Filtrage Non-Linéaire

On considère l'imagette binaire de la Fig. 4.36 obtenue par un algorithme de seuillage sur la teinte d'un visage. NB : L'origine de l'image est située en haut à gauche.

On lui applique le filtre non-linéaire suivant :

$$M(i) = \begin{cases} 1 + \sum\limits_{j \in V(i)} a(j) M(j) & \text{si } U(i) \in \text{“}gris\text{”} \\ 0 & \text{sinon} \end{cases} \tag{4.54}$$

où i est l'indice du pixel courant, $U(i)$ sa valeur de NdG (*“blanc”*=fond, *“gris”*=forme), où M est l'image résultat (initialisée à 0) et où les $a(j)$, avec $j \in V(i)$, sont les 4 coefficients rangés dans la matrice $A = \begin{bmatrix} 1 & 1 & 1 \\ 5 & i & \end{bmatrix}$ pondérant les 4 voisins j du pixel courant i restreints au voisinage causal $V(i)$ comme indiqué dans la matrice (*i.e.* les 4 plus proches voisins précédant le pixel courant).

1. Calculer l'imagette filtrée en remplissant la Fig. 4.36 avec les valeurs calculées.
   NB : feuille à joindre à votre copie sans apposer de signe identifiable.
2. Indiquer les coordonnées (ligne, colonne) du point P_{max} portant la valeur maximum après filtrage.
3. Appliquer l'algorithme de suivi de contour de formes binaires de Rosenfeld et Kak en prenant comme point initial le point P_{max} trouvé ci-dessus et comme direction initiale l'Est ($d_0 = 0$). Donner le code obtenu.
4. Quel contour détecte-t-on ainsi dans l'imagette ? En déduire le rôle de ce filtre.

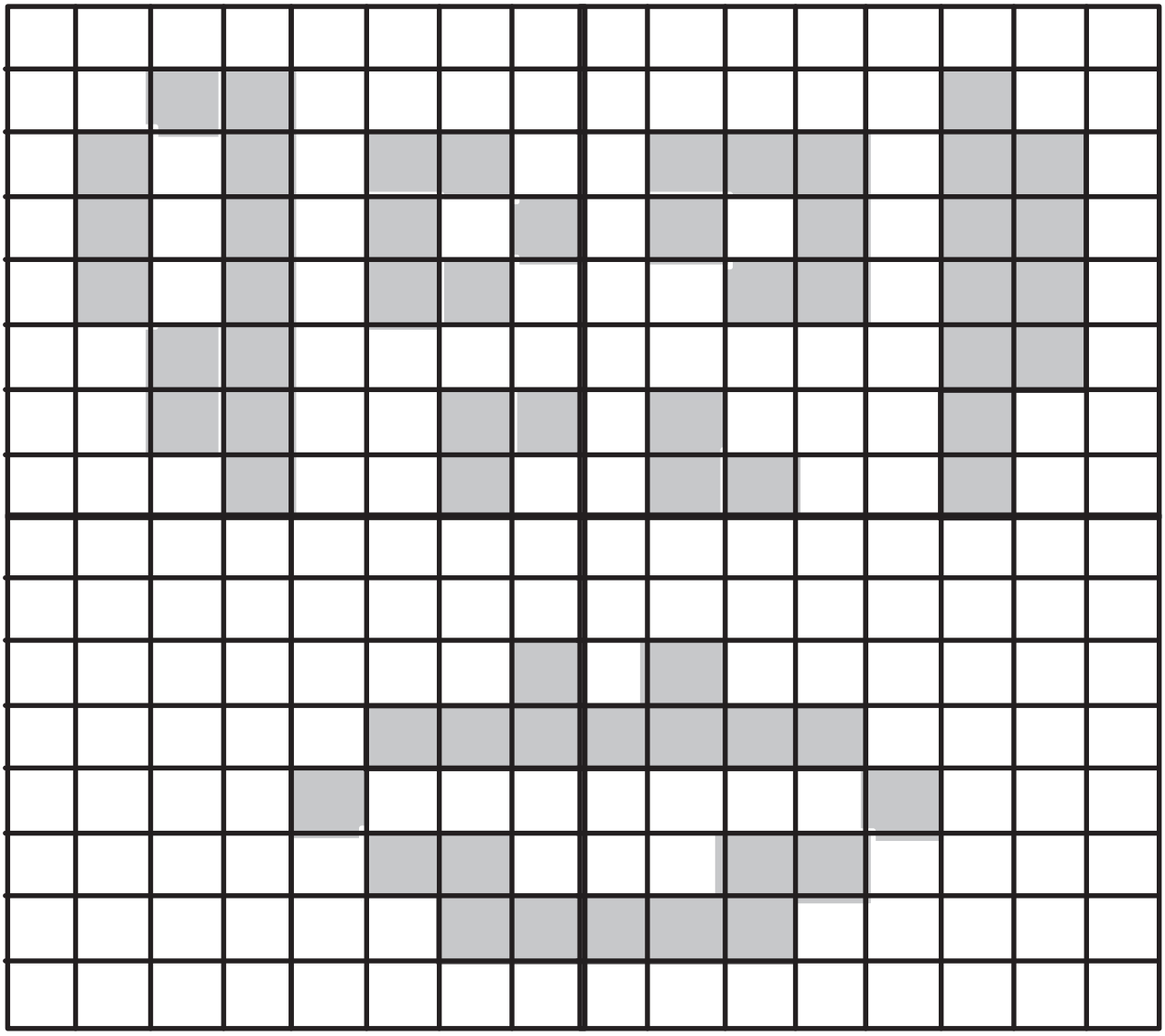

FIGURE 4.36 – Image binaire : fond en blanc, formes en gris.

5. Calculer le taux de compression obtenu grâce au codage par l'algorithme de R&K, ceci **uniquement sur la demi-image** (de taille 8×16) contenant la forme détectée.
6. Mesurer la compacité C de chacune des formes présentes dans l'image sachant que : $C = 4\pi\frac{S}{P^2}$ où S est la surface occupée par un objet (exprimée en nombre de pixels) et P est le périmètre de la forme (également exprimé en nombre de pixels).
7. La compacité est une mesure indiquant si une forme est compacte ou non. La compacité maximale théorique est obtenue pour le disque (maximum de surface entourée par un périmètre minimum). Comparer la compacité de la forme détectée par l'algorithme de R&K avec celle d'un d'œil.

4.57 Etiquetage en Composantes Connexes

On se propose d'étudier un algorithme qui, à partir d'une image binaire I, génère une image d'étiquettes E.

L'image binaire I de départ peut, par exemple, résulter d'une détection de mouvement ou de teinte.

Les pixels p de l'image I valent donc : $I(p) =$ « 0 » pour le FOND noir, ou bien $I(p) =$ « 1 » pour les OBJETS en blanc.

A l'issue du traitement par l'algorithme, chaque pixel p se verra affecté une étiquette $E(p)$.

Travail demandé

1. **Dérouler l'algorithme** sur l'imagette binaire fournie (Fig. 4.39) et, **simultanément**, remplir **au fur et à mesure la table** d'équivalence des étiquettes (Fig.4.37) : chaque colonne de cette table correspond aux mises à jour successives des équivalences d'une étiquette. La première ligne correspond à l'état initial de la table.
2. Représenter sur l'une des grilles vierges de la Fig.4.39, l'image des étiquettes résultant du **premier balayage**.
3. Quel est le **nombre maximal** d'étiquettes générées par l'algorithme ?
4. Représenter, sur une autre grille, l'image finale des étiquettes après le **second balayage**.
5. Conclure sur la **fonction** réalisée par cet algorithme et son intérêt pratique en traitement d'image.
6. Si l'on suppose que chaque étiquette utile est associée à une couleur distincte (par exemple Rouge, Vert, Bleu, Jaune, Turquoise, Mauve, Blanc, Noir, Marron, Gris, Orange, etc...), illustrer (en coloriant) ce que donnera l'algorithme sur les **trois images** de la Fig. 4.38.

NB : Les grilles vierges supplémentaires de la Fig.4.39 pourront servir de brouillon lors du déroulement de l'algorithme.

n° étiquette :	k	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	...
Initialisation :	T(k)	0	1	2	3	4	5	6	7	8	9	10	11	12	...
Mises à jour successives de T(k)		0													
		0													
		0													
		0													
		0													
		0													

FIGURE 4.37 – Table d'équivalence T

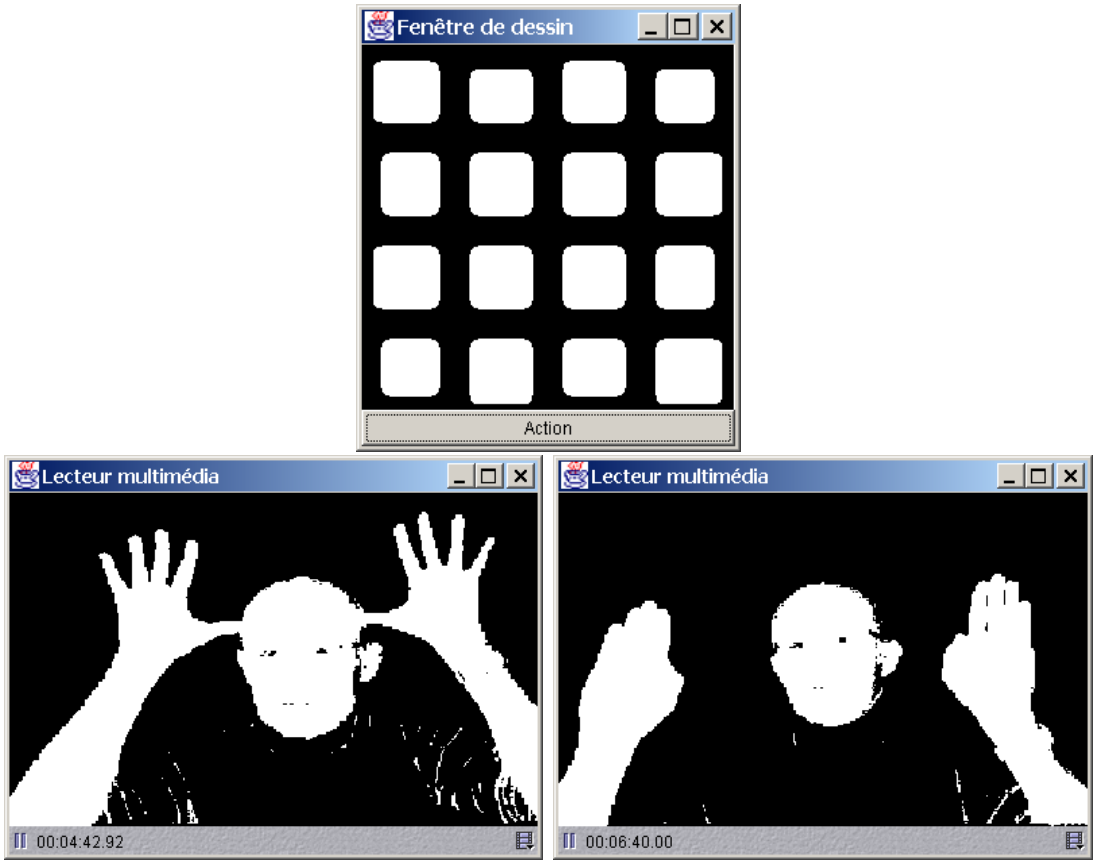

FIGURE 4.38 – Capture d'images réelles

Algorithme

L'algorithme, décrit ci-après, procède en **2 balayages séquentiels** (ligne par ligne de haut en bas et de gauche à droite) et gère simultanément une table **T** d'équivalence entre étiquettes, table qui évolue au cours du balayage.

On considère pour chaque pixel p **uniquement ses 2 voisins causaux** v : c'est-à-dire le voisin **nord** et le voisin **ouest** ($v = n$ ou $v = o$).

Initialisation :

- Les **bords extérieurs** de l'image (i.e. les voisins v manquants) sont supposés appartenir au FOND (I(v)=0).
- L'image E des étiquettes est initialisée à la valeur « 0 » qui représente par convention le FOND : pour tout p, E(p)= 0 .
- k=1 : **numéro initial** de nouvelle étiquette
- La table d'équivalence entre étiquettes est initialisée à **l'identité** : pour tout k , **T[k]=k**

1er balayage (sur l'image) :
Récurrence sur les **pixels de l'objet uniquement** :

POUR tout pixel p appartenant à un objet (I(p)= « 1 ») **FAIRE** :
  SI ses 2 voisins ∈ **FOND** (I(v)=0) **ALORS**
    E(p)=k (nouvelle étiquette)
    k=k+1
  SINON
    SI ses 2 voisins ont 1 **étiquette identique** : ∀v, E(v)=e (c'est-à-dire : E(n)=E(o)=e) **ALORS**
      E(p)=e
    SINON

```
e = minimum{ T[E(n)] , T[E(o) ] }        (où E(n) et E(o) sont les étiquettes des 2 voisins)
SI E(o)= 0  ALORS  e = T[E(n)]             (gestion de cas particulier où un voisin est à 0)
SI E(n)= 0  ALORS  e = T[E(o)]
E(p)=e
POUR chaque voisin non nul (nord et ouest) d'étiquette b  telle que  T[b] ≠e  FAIRE :
  TANT QUE  T[b] ≠e  FAIRE :
    tmp=T[b]
    T[b]=e        (mise à jour de la table d'équivalence)
    b=tmp
   FIN tant que
```

NB : A l'issue de ce 1er balayage, chaque pixel est affecté d'une étiquette **provisoire**.

Actualisation de la table d'équivalence (à l'issue du 1er balayage) :

```
POUR  k=1 jusqu'à Nb maximum d'étiquettes générées par le premier balayage  FAIRE :
  m=k
  TANT QUE : T[m]≠m   FAIRE :
    m=T[m]
  FIN tant que
  T[k]=m
```

2e balayage (sur l'image des étiquettes) :

On affecte à chaque pixel p, d'étiquette provisoire E(p)=k, une étiquette **définitive** correspondant à celle lue dans la table actualisée :

```
POUR  tout pixel p  d'étiquette  E(p)=k  FAIRE :
  E(p)=T[k]
```

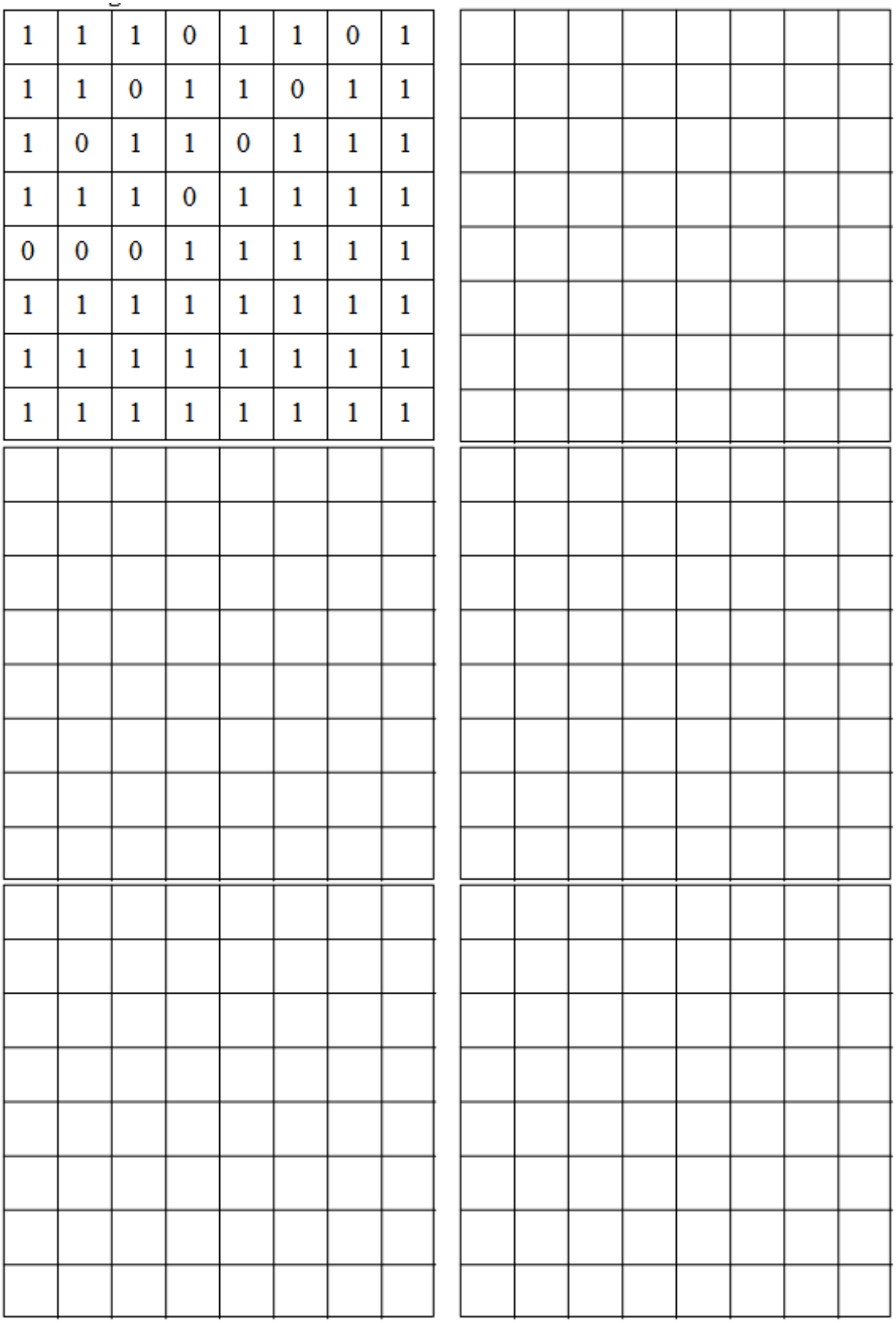

FIGURE 4.39 – Imagette binaire à traiter

4.58 Programmation OpenCV

Soit le code C ci-dessous utilisant la structure image d'OpenCV.

```
void traitement(IplImage* img) {
```

```
    register int l,c,i;
    int haut=img->height; //nb de lignes dans l'image
    int larg=img->width; //nb de pixels par ligne
    int plan=img->nChannels; //nb de canaux couleur par pixel
    int step=img->widthStep; //nb d'octets constituant une ligne
    double pix;

    unsigned char *pin=(unsigned char *)(img->imageData); //pointeur sur la 1ère ligne

    for(l=1;l<haut;l++){
       for(c=1;c<larg;c++){
          for(i=0;i<plan;i++){
              pix=pin[(l-1)*step+c*plan+i]+pin[l*step+(c-1)*plan+i]
                   -4*pin[l*step+c*plan+i]+pin[l*step+(c+1)*plan+i]
                   +pin[(l+1)*step+c*plan+i];
              pin[l*step+c*plan+i]=(unsigned char)pix;
          }
       }
    }
}
```

1. Quel traitement d'image réalise cette fonction ?
2. A quoi sert-il ?
3. Citer quels sont les 2 ou 3 problèmes les plus classiques que l'on risque de rencontrer quand on programme des algorithmes de traitement d'image ?
4. Y a-t'il dans le code source fourni des erreurs ou problèmes ?
5. Si oui, expliquer lesquels et les corriger.

4.59 Détection de contour

On considère l'imagette de la Fig. 4.40 et le filtre de masque $[-1 \quad 1]$, approximation discrète de la dérivée. (NB : le cœfficient de droite est affecté au pixel courant).

0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	2	3	2	0	0
0	0	2	3	4	3	0	0
0	0	5	4	4	3	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

FIGURE 4.40 – Imagette en NdG.

1. Calculer les deux composantes du gradient (Gx, Gy), son module $|G|$ et sa phase Φ pour chaque pixel de l'imagette $(-\pi \leq \Phi \leq \pi)$.

2. Choisir un seuil θ adéquat pour obtenir les contours.
3. Représenter le résultat du seuillage du module du gradient $|G| > \theta$ et commenter le résultat.

4.60 Morphologie mathématique

Soit l'objet binaire X et les éléments structurants E et F de la Fig. 4.41.

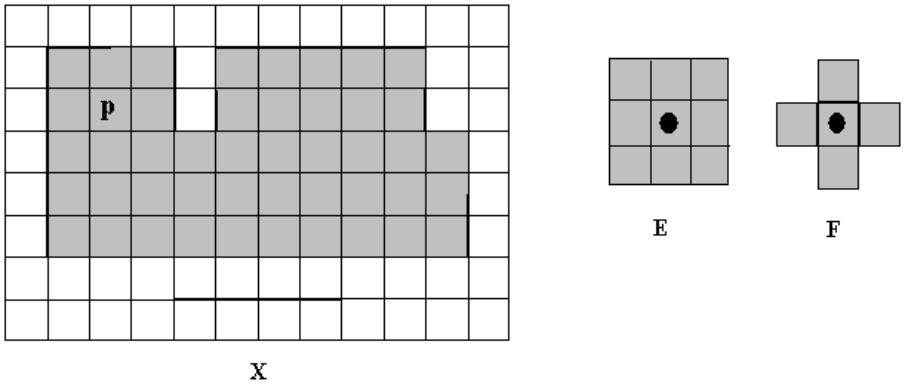

FIGURE 4.41 – Imagette binaire et élements structurants.

1. Donner le résultat Y de l'opération $\beta(X) = X - (X \ominus E)$, où $\ominus$ représente l'érosion.
2. De quel opérateur s'agit-il ?
3. Appliquer au résultat Y obtenu l'opération itérée avec l'élément structuant F :

$$Y_k = (Y_{k-1} \oplus F) \cap Y^c \qquad k = 1, 2, 3, ..., K \tag{4.55}$$

où $\oplus$ représente la dilatation, Y^c le complémentaire, et en partant d'un ensemble initial $Y_0 = \{p\}$ où p est un point quelconque intérieur à Y (p.ex. celui proposé sur la Fig. 4.41).

4. Combien d'itérations K faut-il pour converger ?
5. Que représente l'ensemble $Y_K \cup Y$? Quel est le rôle de ce deuxième opérateur ?

4.61 Codage

Soit une image numérique à 6 niveaux de gris. On considère cette image comme une source de messages $X = \{m_1 \cdots m_6\}$ dont les probabilités $P = \{p_1 \cdots p_6\}$ valent respectivement :

$$p_1 = \frac{3}{8} = 0.375 \quad p_2 = \frac{1}{6} \approx 0.167 \quad p_3 = p_4 = p_5 = \frac{1}{8} = 0.125 \quad p_6 = \frac{1}{12} \approx 0.083$$

1. A priori, combien faut-il de bits informatiques pour coder chaque message émis par cette source d'information (i.e. chaque niveau de gris de l'image) ?
2. Réaliser le codage de Huffman (ou de Shannon-Fano) de cette source.
3. Calculer la longueur moyenne d'un mot-code.
4. Evaluer le taux de compression ainsi obtenu.
5. S'agit-il d'une compression avec perte ou sans perte ?
6. Quelle est la valeur de H_{max}, entropie maximale de la source X ?
7. Calculer l'entropie et la redondance de X avant codage.
8. Recalculer ces deux grandeurs après le codage de Huffman. Conclusions ?
   Indication : pour ce faire, on pourra considérer la source codée comme une nouvelle source émettant 2 messages (les deux symboles "0" et "1") dont on calculera au préalable les probabilités respectives.

NB : Pour ceux qui n'ont pas de calculatrice, on donne le tableau des log binaires : cf. Tab4.4.

TABLE 4.4 – Table de Logarithmes à Base 2

x	1	2	3	4	5	6	7	8	9	10	11	12
$\log_2(x)$	0	1	1.585	2	2.322	2.585	2.807	3	3.17	3.322	3.459	3.585

Chapitre 5

Travaux Pratiques

5.1 Filtre numérique RII : Implantation en C

5.1.1 Filtre de lissage

On veut implanter le filtre numérique de réponse impulsionnelle $h(k)$:

$$h(k) = c.e^{-\alpha|k|} \tag{5.1}$$

Le paramètre de lissage α est un réel positif. La constante de normalisation c est telle que le gain du filtre vaut 1 pour une entrée constante unitaire. On montre que cette condition implique : $c = \frac{\alpha}{2}$. Les équations de récurrence pour l'implantation en cascade sont :

$$y^+(k) = x(k) + e^{-\alpha}(y^+(k-1) - x(k)) \tag{5.2}$$
$$y(k) = y^+(k) + e^{-\alpha}(y(k+1) - y^+(k)) \tag{5.3}$$

Le filtre est donc représenté par deux équations :
- une équation causale qui ne fait intervenir que les valeurs présentes et passées de l'entrée et de la sortie (Eq. (5.2))
- une équation non causale qui fait intervenir les valeurs futures de la sortie (Eq. (5.3))

Le filtrage d'une image consiste simplement à effectuer un premier filtrage 1-D pour chaque ligne de l'image, puis un deuxième filtrage 1-D pour toutes les colonnes (propriété de séparabilité).

1. **Implantation** :
   Ecrire une fonction **USERx_lissage()** qui réalise le lissage d'une image chargée en mémoire.
   On utilisera la structure **image** définie ci-dessous (cf. fichier **bibima.h**) pour stocker les données relatives à une image :

```
/******************/
typedef unsigned char Pixtyp; /*definition du type de donnees (8 bits non signes)*/
/******************/
typedef struct image {
  int lin; /*nombre de lignes*/
  int col; /*nombre de colonnes*/
  Pixtyp **pix;  /*pointeur des pixels*/
} Image, *PImage;
/******************/
```

   On fournit le fichier **bibima.c** qui contient les routines d'allocation etdésallocation de cette structure image.
   On écrira la fonction USERx_lissage() selon le squelette ci-dessous :

```
/*************************/
PImage USERx_lissage(PImage im, double alpha)
{
```

```
  PImage out;
  ...
  b=exp(-alpha);
  for(l=0;l<nbli;l++) /*pour chaque ligne...*/
  { /*lissage*/
    for(c=0;c<nbco;c++) {...}
    for(c=nbco-1;c>=0;c--) {...}
  }
  for(c=0;c<nbco;c++) /*pour chaque colonne...*/
  { /*lissage*/
    for(l=0;l<nbli;l++){...}
    for(l=nbli-1;l>=0;l--){...out->pix[l][c]=...}
  }
  ...
  return out;
}
/*************************/
```

où **out->pix[l][c]** représente le pixel destination de coordonnées (l,c) dans l'image résultat.
NB : Bien penser à la gestion des effets de bords et du type des données.

2. **Tests** :
   lancer l'exécutable et tester le filtrage sur les images de la bibliothèque. Commenter l'influence de α.

5.1.2 Filtre de dérivation

La détection de contour utilise souvent un lissage de l'image dans une direction suivi d'une dérivation dans l'autre direction. La réponse impulsionnelle du filtre de dérivation est alors la dérivée de la réponse impulsionnelle du filtre de lissage. Ainsi, la dérivée de la réponse impulsionnelle précédente peut s'écrire comme la somme d'un terme causal et d'un terme anti-causal :

$$g(k) = g^+(k) + g^-(k) \quad \text{avec} \tag{5.4}$$
$$g^+(k) = -c\alpha e^{-\alpha k} \quad \text{pour } k > 0 \tag{5.5}$$
$$g^-(k) = c\alpha e^{\alpha k} \quad \text{pour } k < 0 \tag{5.6}$$

ce qui conduit tout naturellement à une implantation parallèle. Après normalisation, les équations de récurrence sont :

$$y^+(k) = -x(k) + e^{-\alpha}(y^+(k-1) + x(k)) \tag{5.7}$$
$$y^-(k) = x(k) + e^{-\alpha}(y^-(k+1) - x(k)) \tag{5.8}$$
$$y(k) = y^+(k) + y^-(k) \tag{5.9}$$

1. Implantation : écrire une fonction **USERx_derivation()** pour calculer une composante du gradient.
2. Tester sur les images de la bibliothèque. Commenter l'influence de α.

5.1.3 Détecteur de contours

En appliquant le filtre précédent dans les deux directions, on peut mettre en œuvre une détection de contours. Pour cela, on calculera d'abord le module et l'argument du gradient. On fera ensuite un seuillage pour exhiber les maxima du module.

1. écrire une fonction qui calcule le module et la phase du gradient
2. écrire une fonction de seuillage
3. tester ce détecteur sur les images de la bibliothèque. Commenter l'influence du seuil θ.

N.B. : Implantation logicielle

On pourra au choix :

- Insérer les routines de traitement dans l'application **markov** disponible sur le serveur (informations fournies en TP). Ceci permet de disposer d'une interface graphique pour la gestion et l'affichage des images (fichiers au format propriétaire TRF).
- Construire une application autonome qui intègre les routines de traitement et les entrées/sorties fichier. Il faudra alors écrire un programme principal en C appelant trois fonctions :

  1. lecture d'un fichier image (chargement en mémoire) au format PGM,
  2. traitement de l'image chargée en mémoire,
  3. sauvegarde du résultat dans un fichier image PGM.

  On utilisera dans ce cas l'application **xv** (commande **xv&**) pour la visualisation des fichiers images au format PGM.

5.2 Estimateur de mouvement : Implantation en C

5.2.1 Algorithme de Horn et Schunck

On rappelle que l'algorithme itératif est défini par :

$$u^{k+1} = \overline{u}^k - \frac{I_x[I_x\overline{u}^k + I_y\overline{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{5.10}$$

$$v^{k+1} = \overline{v}^k - \frac{I_y[I_x\overline{u}^k + I_y\overline{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2} \tag{5.11}$$

où :

- I_x, I_y, I_t sont les dérivées partielles par rapport à x, y, t calculées dans le cube $2 \times 2 \times 2$ (cf. cours Fig. 2.18a),
- α est un coefficient de lissage (typ. $\alpha = 40$),
- k est l'indice d'itération (typ. $k_{max} = 50$),
- u^k, v^k sont les estimées des composantes de vitesse u, v à l'itération k (condition initiale : $u^0 = 0, v^0 = 0$)
- $\overline{u}, \overline{v}$ sont des moyennes pondérées sur un voisinage 3×3 (cf. cours Fig. 2.18b).

Le critère d'arrêt portera sur la variation relative $\left|\frac{\Delta u}{u}\right| + \left|\frac{\Delta v}{v}\right|$ comparée à un seuil de précision (typ. $p = 0.01\%$).

NB : En pratique, un lèger filtrage passe-bas initial de la séquence est souvent souhaitable pour réduire l'influence du bruit sur les calculs de dérivées (on pourra utiliser pour cela le programme du TP précédent).

1. Donner les équations pour le calcul de I_x, I_y, I_t et $\overline{u}, \overline{v}$
2. Donner l'organigramme de l'algorithme complet, en séparant le prétraitement (filtrage passe-bas optionnel et calcul des gradients), le traitement principal (calcul itératif des vitesses) et l'affichage des résultats (images de gradients et vecteurs-vitesses).

5.2.2 Calcul des gradients

Ecrire une routine qui calcule les gradients (3 tableaux de *float*) à partir de deux images successives (pixel de type *unsigned char*).

Pour pouvoir visualiser les résultats grâce à l'interface graphique, on utilisera les structures prédéfinies *image, sequence, flot et seqflot* et les routines associées : ***_create(), *_destroy(), *_display()** (déclarées dans **bibima.h** et définies dans **bibima.c**).

NB : On dispose d'une routine **float2ima** pour transférer un tableau de flottants dans une structure image en vue de l'affichage des gradients :

```
void float2ima(pin,img)
float **pin; /*pointeur memoire 2D flottants en entree*/
PImage img; /*structure image normalisee en sortie*/
{...}
```

5.2.3 Estimation itérative des vitesses

Ecrire une routine qui calcule les vitesses (paramètres d'entrée : α, k_{max}, p). Les composantes u, v seront stockées dans une structure pointée par **seqflot** (pour affichage automatique via le menu display de l'interface).

5.2.4 Test

On testera l'algorithme sur quelques séquences de la bibliothèque (au format TRF). Tester d'abord sur la séquence **carre2.trf** (carré en translation diagonale) et interpréter les résultats. Puis sur **cercle.trf** contenant un objet se déplaçant à la vitesse de 3 pixels/image. Commentaire ?

NB : Utilisation de l'interface

L'utilisateur **USERx** dispose d'un makefile (lire les informations en entête) et d'un point d'entrée sous forme d'un fichier **USERx_interface.c** qui crée une fenêtre avec deux boutons (apply et reset) que l'on affectera aux deux procédures à écrire (USERx_gradient et USERx_vitesse). On pourra si besoin ajouter d'autres objets graphiques (bouton, entrée numérique $\cdots$).

NB : Les données de la séquence sont chargées dans une structure de type *sequence* pointée par **seq**.

Les informations sur l'utilisation des menus de l'interface graphique (load, processing, display) seront données en TP.

5.3 Correction d'histogramme

Programmation d'une LUT de correction d'histogramme.

Une LUT est un tableau de transfomation 1-D : les adresses sont les niveaux de gris en entrée, les données pointées par les adresses sont les niveaux de gris en sortie.

Le Tab. 5.1 montre le cas particulier de l'inversion des niveaux de gris, qui permet d'obtenir le négatif d'une image.

TABLE 5.1 – Principe de la LUT

adresse i	donnée $T(i)$	Ex : négatif
0	$T(0)$	255
1	$T(1)$	254
.	.	.
254	$T(254)$	1
255	$T(255)$	0

1. Programmer différentes LUT linéaires par morceaux pour modifier l'histogramme d'une image et tester l'effet sur les images de la bibliothèque.
2. Programmer l'égalisation d'histogramme.
3. Programmer l'étalement d'histogramme.

5.4 Transformation de Fourier

Implantation de l'algorithme de FFT et analyse fréquentielle.

On donne ci-dessous l'algorithme en FORTRAN de la FFT 1-D (1024 points) de Cooley-Tukey.

Implanter en C la FFT 2-D pour afficher le spectre d'une image (en module et en phase).

```
      SUBROUTINE FFT(F,LN)
      COMPLEX F(1024),U,W,T,CMPLX
      PI=3.141593
      N=2**LN
      NV2=N/2
      NM1=N-1
      J=1
      DO 3 I=1,NM1
        IF(I.GE.J) GO TO 1
        T=F(J)
        F(J)=F(I)
        F(I)=T
1     K=NV2
2     IF(K.GE.J) GO TO 3
        J=J-K
        K=K/2
        GO TO 2
3     J=J+K
      DO 5 L=1,LN
        LE=2**L
        LE1=LE/2
        U=(1.0,0.0)
        W=CMPLX(COS(PI/LE1),-SIN(PI/LE1))
        DO 5 J=1,LE1
          DO 4 I=J,N,LE
```

```
        IP=I+LE1
        T=F(IP)*U
        F(IP)=F(I)-T
4     F(I)=F(I)+T
5 U=U*W
  DO 6 I=1,N
6 F(I)=F(I)/FLOAT(N)
  RETURN
  END
```

5.5 Filtres Flou et Détecteur de Contours : Implantation Java

5.5.1 Présentation

On fournit :
- quelques images-test JPEG
- deux fichiers d'exemple `ChangeCouleurFond.Java` et `DiminueTaille.Java` utilisant le PixelGrabber pour accéder au tableau de pixels d'une image.
- les sources (`filtr_tp.c`) correspondant à l'implantation en C des filtres exponentiels classiques de Shen et Castan (cf énoncé TP 5.1).

5.5.2 Travail demandé

1. Implanter le filtre de lissage exponentiel.
   Il requiert un paramètre réglant la force du filtrage : $\alpha > 0$ (typ. $\alpha \in 0...1$).
2. Tester le filtre sur les différentes images. Commenter l'influence de α.
3. Implanter le filtre de dérivation pour l'extraction des contours.
   Cela requiert deux paramètres : α et le seuil θ appliqué sur le module du gradient.
4. Interface : ajouter deux entrées numériques sur l'interface graphique permettant à l'utilisateur d'ajuster les 2 paramètres.
5. Tester l'extracteur de contours notamment sur l'image-test damier_00.jpg pour voir le phénomène visuel perçu.

5.6 Détecteur de mouvement : Implantation Java

5.6.1 Présentation

On fournit :
- une séquence d'images-test au format JPEG : bleroJPG.zip
- un fichier d'exemple VisioSeqJCV.java et les classes nécessaires pour réaliser une visionneuse de séquence d'images avec réglage du timer (en ms).

5.6.2 Travail demandé

1. Calculer et visualiser les observations de mouvement (simples différences d'images).
2. Implanter un seuillage pour obtenir la séquence binaire des changements temporels.
3. Ajouter un post-traitement de morphologie mathématique pour améliorer la détection et visualiser les masques mobiles.
4. Remplacer finalement le fond fixe de la scène par un autre fond obtenu avec l'image plage.jpg.

5.7 Traitement d'images avec OpenCV : Implantation Java dans Eclipse

5.7.1 Présentation

On dispose de :
- 3 bibliothèques (préalablement téléchargées) placées à la racine `C:\` de chaque PC :
  opencv : 275 Mo (l'exe est disponible sur opencv.org)
  javacv-bin : 13 Mo
  javacv-cppjars : 89 Mo (les zip sont disponibles sur code.google.com)
- une documentation Learning OpenCV dans le parcours Traitement d'Images du webcampus (rubrique Part III-Travail proposé).
- des video-tutoriels (youtube) sur le site en ligne :
  `http://engineervisions.blogspot.in/p/javacv.html`
- un site expliquant l'installation de JavaCV et la configuration de l'environnement Eclipse :
  `http://opencvlover.blogspot.fr/2012/04/javacv-setup-with-eclipse-on-windows-7.html`

5.7.2 Travail demandé

1. créer un nouveau projet :
   `File>New JavaProject>ProjectName 'MonProj'>Next>Add project 'MonProj' to build path;`
   et importer les 3 archives JAR dans l'environnement de programmation :
   `Project>Properties>Java Build Path>Add external JARS > Order&Export > Select All`
2. Importer un premier programme pour tester le bon fonctionnement :
   `File Import>General>File System > 'CheminRepertoire'`
   choisir les sources 'JavaCV1.java' et les mettre dans 'MonProj/src' ; choisir les fichiers de données (img, vidéo) et les mettre dans répertoire 'MonProj/'
3. En vous aidant de TP3.java et des tutoriels en ligne, implanter un traitement parmi les suivants sur un fichier vidéo ou webcam :
   - Histogramme et seuillage
   - Détection teinte Chair
   - espace couleur LUX
   - estimation de mouvement (Horn et Schunk)
   - transformée de Fourier (DCT)
   - ROI et détection de contours (algo Canny)
   - pyramide et détection de contours (algo Sobel)
   - Morphologie Mathématique
   - Lissage (flou)
   - Seuillage entropique
   - Rehaussemet de contraste par égalisation ou étalement d'histogramme
   - ...

5.8 Projet : Interface de Traitement d'Images

Réaliser une interface graphique en Java qui offre les fonctions suivantes :
- application d'un flou sur une séquence d'images JPEG
- détection de contours et visualisation dynamique
- détection de mouvement
- changement du fond de la séquence (incrustation du mobile dans une nouvelle scène)

L'utilisateur doit pouvoir :
- choisir la séquence d'images à traiter
- régler les différents paramètres de traitement (seuil, caractéristique du filtre...)
- choisir le fond sur lequel il souhaite incruster l'objet en mouvement (une image plage.JPG est fournie à titre d'exemple).

5.8.1 Conseils sur le travail demandé

- Penser à travailler en objets (définir les classes ad hoc).
- Rendre un document avec toutes les explications utiles et les listings.

5.8.2 Classes utiles au projet image

cf Fichiers programmes : `ChangeCouleurFond.java` et `VisioSeq.java`

```
ImageIcon
 .getIconWidth()
 .getIconHeight()
 .getImage()
Image
 .createImage()
 .drawImage(img)
 .prepareImage()
MemoryImageSource
PixelGrabber
 .grabPixels()
Color
 .getRGB()
 .getGreen()
 .equals()
Visionneuse            //classe maison (auteur M.D)
 .ajouterMedia(img)
 .automatique(int tps) //affichage avec interval tps (en ms)
```

Pour ceux qui connaissent le C : cf. Fichier source `filtr_tp.c` (auteur F.L)

5.9 Projet : Acquisition, Traitement et Transmission Vidéo

Objectifs

Réaliser une application qui offre les fonctionnalités suivantes :
- acquisition et restitution vidéo (webcam)
- traitement d'images en temps-réel (*live*) ou différé (*batch*)
- transmission vers un poste distant (via Internet)

Cahier des charges

Une interface graphique conviviale permettra à l'utilisateur de :
- choisir la source de données : flux vidéo capté par la webcam, séquence AVI, ou images JPEG stockées sur disque
- régler les paramètres d'acquisition, transfert et restitution (choix URL, port, adresse IP)
- choisir les paramètres de traitement et filtrage (via des entrées numériques ou *slider*)
- visualiser simultanément la source vidéo et le résultat du traitement
- sauvegarder des données sur disque (séquence acquise, image résultat...)

Consignes de travail

- On utilisera le langage Java et son API JMF (Java Media Framework).
- Penser à travailler en objets : définir classes, interfaces et méthodes ad hoc, donner le diagramme de classes.
- Rendre un document avec toutes les explications utiles, organigrammes, illustrations de résultats, objets graphiques et fonctionnalités de l'interface,
- NB. listings inutiles : Envoyer les fichiers sources java,
- Tester et caractériser l'application : performances, limitations, bugs, rapidité, efficacité, qualités et défauts, perspectives d'évolution.
- On travaillera en binôme et l'on implantera l'un des traitements suivants.

Liste des traitements proposés

1. Application d'un **flou** (de force réglable) sur les NdG ou sur la couleur
2. **Détection de contours** sur les NdG et incrustation sur la couleur
3. **Détection de présence** basée sur la détection de mouvement et déclenchement d'une alarme visuelle ou sonore sur le poste distant
4. **Rehaussement automatique de contraste** par égalisation ou étalement de l'histogramme des NdG
5. **Détection de teinte chair** (algo fourni)
6. **Rehaussement de détails** (algo fourni)
7. Apprentissage et **classification d'objets** par reconnaissance de forme et/ou de teinte (algo fourni).
8. **Trucage d'image** par manipulation d'histogramme : affichage d'histogramme et choix par l'utilisateur d'une transformation non-linéaire (linéaire par morceaux).
9. **Filtre morphologique adaptatif** (algo fourni)
10. Seuillage entropique pour détection de mouvement
11. Etiquetage en composantes connexes (algo fourni)
12. Implantation de régularisation par MRF pour détection de mouvement
13. Estimation de mouvement (algo Horn et Schunck)
14. Autre algo morpho math : squelettisation
    Pgm canevas : TPM2.java 2006

5.10 TP MatLab 1 - Séance d'Initiation

Initiation au traitement d'image

Démos matlab

La toolbox Traitement d'Image contient de nombreuses démonstrations dans le répertoire :

`<Matlab>\toolbox\images\imdemos\`

Lancer quelques démos, par exemple **imadjdemo, edgedemo, firdemo, dctdemo, nrfiltdemo**

Lecture-Ecrirture, Affichage et Manipulation d'Image

Ecrire un programme qui lit et affiche une image et vérifier la taille mémoire.

`help: imread(), imshow(), image(), whos`

Ajouter divers manipulations telles que : découpage, re-échantillonnage, rotation,

`help: imcrop(), imrotate(), imresize()`

Sauvegarder le résultat d'un de vos traitements dans un fichier au format JPEG et afficher ses infos

`help: imwrite(), imfinfo,`

Modification d'Histogramme et Seuillage

Afficher et égaliser l'histogramme d'une image à faible contraste (p.ex. l'image **auto_00.bmp** ou l'image **pout.tif** fournie dans matlab)

`help: imhist(), histeq()`

Ecrire une fonction d'étalement d'histogramme et la tester dans votre programme principal. Comparer à la fonction prédéfinie dans Matlab (... trouver laquelle !)

Compléter le programme en ajoutant un seuillage des NdG de l'histogramme pour binariser l'image. Le seuil sera soit choisi par l'utilisateur, soit déterminé automatiquement.

`help: im2bw(), graythresh()`

Filtrage Linéaire : Détection de Contours, Gradient, Laplacien, Lissage

Tester un détecteur de contour proposé dans matlab.

Implanter par ailleurs votre propre détecteur de contour avec le masque bien connu $[-1 \quad 1]$. Implanter aussi un Laplacien et un Lisseur 5×5. Visualiser les images résultats.

`help: edge(),filter2(), imfilter(), im2double(), unint8(), sqrt(), ones(), colorbar`

Transformée de Fourier et Fonction de Transfert

Calculer et afficher le spectre d'une image. Visualiser la fonction de transfert d'un filtre gaussien

`help: fft2(), fftshift(), log(), abs(), fspecial(), freqz2()`

5.11 TP MatLab 2 - Traitements d'image statique

5.11.1 Filtrage Linéaire

Tracer les fonctions de transfert des filtres vus en TD. Les appliquer sur quelques images et visualiser les résultats.

`help: freqz2(), imfilter()`

5.11.2 Détection de teinte chair

On fournit le principe d'un algorithme de détection de teinte chair pour le suivi de visage (voir [71]). Il travaille sur les composantes normalisées $r = \frac{R}{R+G+B}$ et $g = \frac{G}{R+G+B}$ et recherche les pixels dont le couple (r, g) est compris à l"intérieur d'un croissant de lune du plan (r, g) limité par deux paraboles dont les coefficients sont fournis au verso. Un simple test d'appartenance au croissant de lune fournit les pixels candidats pour la teinte chair.

Programmer cet algorithme sous Matlab et le tester sur les images de visages fournies

5.11.3 Morphologie Mathématique

Tester quelques-uns des opérateurs de Morphologie Mathématique disponibles dans Matlab, à la fois sur des images en NdG :

`help: strel(), imerode(), imdilate(), imclose(), imopen(), imtophat(),`

ou sur des images en Noir et Blanc (obtenues après un seuillage p.exemple) :

`bwhitmiss(), bwmorph()`

Bibliographie

[1] V. Lebugle. Une vision industrielle du traitement de l'image. *Le journal de la production*, (91) :24–30, septembre/octobre 2008.

[2] J. P. Coquerez and S. Philipp. *Analyse d'images : filtrage et segmentation.* Masson, Paris,, 1995.

[3] A. Belaïd and Y. Belaïd. *Reconnaissance des formes : Méthodes et applications.* InterEditions, Paris, 1992.

[4] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27 :379–423,623–656, July, October 1948.

[5] F. Luthon and M. Liévin. Entropy power for thresholding technique in image processing. In *XI European Signal Processing Conf. (EUSIPCO'02)*, pages 605–608, Toulouse, France, Sept. 3-6 2002. Vol. I.

[6] F. Luthon, M. Liévin, and F. Faux. On the use of entropy power for threshold selection. *Signal Processing*, 84 :1789–1804, 2004.

[7] R.C. Gonzalez and R.E. Woods. *Digital image processing.* Addison-Wesley Publishing Company, Reading, MA., 1993.

[8] W.K. Pratt. *Digital Image Processing.* John Wiley, New York, 2nd edition, June 1991. 698p.

[9] A. Chéhikian, P. Y. Coulon, and F. Luthon, editors. *Ecole des Techniques Avancées en Signal-Image-Parole (ETASIP'97)*, Grenoble, 9-12 Septembre 1997. LTIRF-INPG. Session 2, 486 pages.

[10] P.Y. Coulon. *Traitement des Images.* Reprographie ENSERG, INPG, Grenoble, 1998.

[11] M. Najim. *Synthèse de filtres numériques en traitement du signal et des images.* Hermes, Lavoisier, Paris, 2004.

[12] A. Marion. *Introduction aux techniques de traitement d'images.* Eyrolles, Paris, 1987.

[13] J.M. Jolion. *Les systèmes de vision, Traitement du Signal et de l'Image.* Hermès Sciences Europe, Paris, 2001.

[14] C.G. Relf. *Image Acquisition and Processing with LabVIEW.* CRC Press, Boca Raton, 2004.

[15] V. Torre and T. A. Poggio. On edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(2) :147–163, March 1986.

[16] D. Marr. *Vision.* Freeman, 1980.

[17] F. Luthon and F. Clément. Macroscopic quality measurement of plasma treated polystyrene through computer vision. In *Int. Conf. on Automation, Quality & Testing, Robotics, AQTR'06*, pages 321–326, Cluj-Napoca, Romania, May 25-28 2006. IEEE-TTTC. Vol. II.

[18] J. Larrieu, F. Clément, B. Held, N. Soulem, F. Luthon, C. Guimon, and H. Martinez. Analysis of microscopic modifications and macroscopic surface properties of polystyrene thin films treated under DC pulsed discharge conditions. *Surface and Interface Analysis*, 37 :544–554, 2005.

[19] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York, 1973.

[20] A. Rosenfeld and A. Kak. *Digital Picture Processing.* Academic Press, 1976.

[21] A. Trémeau, C. Fernandez-Maloigne, and P. Bonton. *Image Numérique Couleur. De l'acquisition au traitement.* Sciences Sup. Dunod, Paris, 2004.

[22] M. Lievin. *Analyse entropico-logarithmique de séquences vidéo couleur. Application à la segmentation markovienne et au suivi de visages parlants.* PhD thesis, National Polytechnic Institute, Grenoble, France, September 2000.

[23] M. Liévin and F. Luthon. Nonlinear color space and spatiotemporal MRF for hierarchical segmentation of face features in video. *IEEE Trans. on Image Processing*, 13(1) :63–71, January 2004.

[24] F. Luthon, B. Beaumesnil, and N.Dubois. LUX color transform for mosaic image rendering. In *17th IEEE Int. Conf. on Automation, Quality and Testing, Robotics (AQTR 2010)*, pages 93–98, Cluj-Napoca, Romania, May 28-30 2010. Vol. III.

[25] F. Faux and F. Luthon. Théorie de l'évidence pour suivi de visage. *Traitement du Signal*, 28(5) :515–545, Sept-Oct. 2011.

[26] F. Faux and F. Luthon. Theory of evidence for face detection and tracking. *Int. Journal of Approximate Reasoning*, 53(5) :728–746, July 2012.

[27] F. Luthon. Face detection using the theory of evidence. In F. Dornaika, editor, *Advances in Face Image Analysis : Theory and Applications*, chapter 9, pages 169–200. Bentham Science Publishers, 2015. ISBN : 978-1-68108-111-3.

[28] F. Luthon. Audioslide ScienceDirect. `https://www.youtube.com/watch?v=AaV51gz1GBU`, 2013.

[29] T. Aach, A. Kaup, and R. Mester. Statistical model-based change detection in moving video. *Signal Processing*, 31(2) :165–180, March 1993.

[30] K. Skifstad and R. Jain. Illumination independent change detection for real world image sequences. *Computer Vision, Graphics, and Image Processing*, 46 :387–399, 1989.

[31] Y. Z. Hsu, H. H. Nagel, and G. Reckers. New likelihood test methods for change detection in image sequences. *Computer Vision, Graphics, and Image Processing*, 26 :73–106, 1984.

[32] R. Chellappa and A. Jain, editors. *Markov Random Fields : Theory and Application.* Academic Press, Inc., San Diego, 1993.

[33] P. Bouthémy and P. Lalande. Recovery of moving object masks in an image sequence using local spatiotemporal contextual information. *Optical Engineering*, 32(6) :1205–1212, June 1993.

[34] F. Luthon, A. Caplier, and M. Liévin. Spatiotemporal MRF approach to video segmentation : Application to motion detection and lip segmentation. *Signal Processing*, 76(1) :61–80, July 1999.

[35] A. Caplier. *Modèles markoviens de détection de mouvement dans les séquences d'images : Approche spatio-temporelle et mises en œuvre temps réel.* PhD thesis, Institut National Polytechnique, Grenoble, France, December 1995.

[36] C. Dumontier. *Etude et mise en œuvre temps réel d'un algorithme de détection de mouvement par approche markovienne.* PhD thesis, Institut National Polytechnique, Grenoble, November 1996.

[37] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6(6) :721–741, November 1984.

[38] B. Benmiloud and W. Pieczynski. Estimation des paramètres dans les chaînes de Markov cachées et segmentation d'images. *Traitement du Signal*, 12(5) :433–454, 1995.

[39] W. Pieczynski. Champs de Markov cachés et estimation conditionnelle itérative. *Traitement du Signal*, 11(2) :141–153, 1994.

[40] J. Besag. On the statistical analysis of dirty pictures. *J. R. Statist. Soc. B*, 48(3) :259–302, 1986.

[41] A. Caplier and F. Luthon. Approche spatio-temporelle pour l'analyse de séquences d'images. Application en détection de mouvement. *Traitement du Signal*, 14(2) :195–208, 1997.

[42] A. Caplier, F. Luthon, and C. Dumontier. Real-time implementations of an MRF-based motion detection algorithm. *Real-Time Imaging*, 4(1) :41–54, February 1998.

[43] F. Luthon and D. Dragomirescu. A cellular analog network for MRF-based video motion detection. *IEEE Trans. on Circuits and Systems-I*, 46(2) :281–293, February 1999.

[44] C. Dumontier, F. Luthon, and J. P. Charras. Real-time DSP implementation for MRF-based video motion detection. *IEEE Trans. on Image Processing*, 8(10) :1341–1347, October 1999.

[45] A. Caplier, C. Dumontier, F. Luthon, and P. Y. Coulon. Algorithme de détection de mouvement par modélisation markovienne. Mise en oeuvre sur DSP. *Traitement du Signal*, 13(2) :177–190, 1996.

[46] F. Luthon and B. Beaumesnil. Color and R.O.I. with JPEG2000 for wireless videosurveillance. In *IEEE Int. Conf. on Image Processing (ICIP'04)*, pages 3205–3208, Singapore, October 24-27 2004. Vol. 5.

[47] F. Luthon and M. Liévin. Lip motion automatic detection. In *10th Scandinavian Conference on Image Analysis (SCIA'97)*, pages 253–260, Lappeenranta, Finland, June 9-11, 1997.

[48] F. Luthon and B. Beaumesnil. Real-time liptracking for synthetic face animation with feedback loop. In *International Conference on Computer Vision Theory and Applications (VISAPP'06)*, pages 402–407, Setubal, Portugal, 25-28 Feb 2006. Vol.2.

[49] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow technics. *International Journal of Computer Vision*, 12(1) :43–77, 1994.

[50] P. Golland and A.M. Bruckstein. Motion from color. *Computer Vision and Image Understanding*, 68(3) :346–362, December 1997.

[51] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17 :185–203, 1981.

[52] H. H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(5) :565–593, September 1986.

[53] H. H. Nagel. On the estimation of optical flow : relations between different approaches and some new results. *Artificial Intelligence*, 33 :299–324, 1987.

[54] D. J. Heeger. Model for the extraction of image flow. *J. Opt. Soc. Am. A*, 4(8) :1455–1471, August 1987.

[55] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2 :283–310, 1989.

[56] F. Lopes and M. Ghanbari. Hierarchical motion estimation with spatial transforms. In *7th IEEE Int. Conf. Image Processing (ICIP'2000)*, Vancouver, Canada, 2000.

[57] A. N. Netravali and J. D. Robbins. Motion-compensated television coding : Part 1. *Bell System Technical Journal*, 58(3) :631–670, March 1979.

[58] D. R. Walker and K. R. Rao. Improved pel-recursive motion compensation. *IEEE Trans. on Communications*, 32(10) :1128–1134, October 1984.

[59] J. Biemond, L. Looijenga, D.E. Boekee, and R.H.J.M. Plompen. A pel-recursive Wiener-based displacement estimation algorithm. *Signal Processing*, 13 :399–412, 1987.

[60] N. Baaziz. *Approches d'estimation et de compensation de mouvement multirésolutions pour le codage de séquences d'images.* PhD thesis, Université de Rennes 1, October 1991.

[61] I. Daubechies. *Ten lectures on wavelets*, volume 61 of *CBMS-NSF regional conference series in Applied Mathematics.* SIAM, Philadelphia, Pennsylvania, 1992.

[62] S. G. Mallat. Multifrequency channel decompositions of images and wavelet models. *IEEE Trans. on ASSP*, 37(12) :2091–2110, December 1989.

[63] D. Lecomte, D. Cohen, P. de Bellefonds, and J. Barda. *Les normes et les standards du multimédia.* Dunod, Paris, 2nd edition, 2000.

[64] D.S. Taubman and M.W. Marcellin. *JPEG 2000 Image Compression Fundamentals, Standards and Practice.* Kluwer Academic Publishers, The Netherlands, 2001.

[65] M. Rabbani and R. Joshi. An overview of the JPEG 2000 still image compression standard. *Signal Processing : Image Communication*, 17 :3–48, 2002. www.jpeg.org/JPEG2000.htm.

[66] J. Murray and W. Van Ryper. *Encyclopedia of Graphics File Formats.* O'Reilly, 1994.

[67] W. Brown and B. Shepherd. *Graphics File Formats reference & guide*. Manning Publications Co., Greenwich, ? ?

[68] A.C. Luther. *Audio et vidéo numériques. Principes et applications*. Eyrolles, Paris, 2001.

[69] S.Y. Chien, Y.W. Huang, C.Y. Chen, H.H. Chen, and L.G. Chen. Hardware architecture design of video compression for multimedia communication systems. *IEEE Communications Magazine*, 43(8) :123–131, August 2005.

[70] A. Bovik, editor. *Handbook of Image & Video Processing*. Elsevier, USA, 2nd edition, 2005.

[71] M. Soriano, B. Martinkauppi, S. Huovinen, and M. Lakksonen. Using the skin locus to cope with changing illumination conditions in color-based face tracking. In *Proc. IEEE Nordic Signal Processing Symposium (NORSIG'2000)*, Kolmaarden, Sweden, June 2000.