TABLES DES MATIERES

NOTATIONS

1. Minuscules Latines

$col_{imp}(M)$	Colonnes impairs de la matrice M
$dist[m, n]$	Distance par rapport à l'axe des gris
$f_{filtrée}[m, n]$	Opération de filtrage proposée par Sangwine
$g_\sigma(i, j)$	Noyau gaussien
h(i, j)	Masque d'un filtre
h(t)	Réponse impulsionnelle du filtre
$lig_{imp}(M)$	Lignes impairs de la matrice M
q	Quaternion
$\overline{q}$	Conjugué de q
q^{-1}	Inverse de q
q_{proj}	Projection de q par rapport à un vecteur
q_{refl}	Réflexion de q par rapport à un vecteur
q_{rej} :	Réjection de q par rapport à un vecteur
$rej[m, n]$	Réjection
v_d	Vecteur propre à droite d'une matrice
v_g	Vecteur propre à gauche d'une matrice
x(t)	Signal
$x_{filtré}(t)$	Transformée de Fourier inverse du spectre

2. Majuscules Latines

(C_1,C_2, C_3)	Espace couleur
$\mathbb{C}$	Ensemble des nombres complexes
COV(x)	Covariance de la matrice x
$(\mathbb{H}, +, x)$	Corps non commutatif des quaternions
H(f)	Transformée de Fourier du filtre h(t)
I (i, j)	Pixel de l'image sur la colonne i et sur la ligne j
$\mathbf{I}(n,m)$	Image numérique couleur comportant N lignes et M colonnes
I_b , $I_b(I, j)$	Image bruitée
$\mathbf{I}_{R,V,B}(n,m)$	Images des trois canaux Rouge, Vert et Bleu
$M_4(\mathbb{R})$	Sous-algèbre des matrices
$\overrightarrow{OC}$	Vecteur couleur
PSNR	Mesure de distorsion utilisée en image numérique
Q'(x, y)	Filtre quaternionique
RVB	Espace de couleurs (Rouge, Vert et Bleu)
$\|S\|_k$	Somme des k-premiers termes des « eigen-images »
S(q)	Partie scalaire d'un quaternion
TSL	Espace colorimétrique (Teinte, Saturation, Luminance)
U	Matrice orthogonal m×m
V	Matrice orthogonal n×n
V(q)	Partie vectorielle d'un quaternion
W	Matrice unitaire contenant les vecteurs propres
X(f)	Transformée de Fourier du signal x(t) à filtrer
$X_{filtré}(f)$	Spectre

3. Minuscules grecques

λ_g	Valeur propre gauche d'une matrice
λ_d	Valeur propre droite d'une matrice
μ_{gris}	Axe de niveau gris
σ	Ecart-type
$\sigma_1, \sigma_2, \ldots, \sigma_n$	Valeurs singulières d'une matrice

4. Majuscules grecques

Φ	Angle principal du quaternion
Σ	Matrice $m \times n$ contenant les valeurs singulières
Ω	Matrice diagonal des valeurs propres

ABBREVIATIONS

3D	Trois Dimensions
ACP	Analyse en Composantes Principales
ACPQ	Analyse en Composantes Principales Quaternionique
EQM	Erreur Quadratique Moyenne
JPEG	Joint Photographic Experts Group
KLT	Karhunen-Loeve Transformation
KLTQ	« Karhunen-Loeve Transformation » Quaternionique
PAO	Publication Assistée par Ordinateur
PSNR	Peak Signal to Noise Ratio
RVB	Rouge, Vert et Bleu
SVD	Singular Value Decomposition
SVDQ	« Singular Value Decomposition » Quaternionique
SVH	Système Visuel Humain
TCD	Transformation Cosinus Discrète
TSL	Teinte, Saturation, Luminance
TWH	Transformation de Walsh-Hadamard

INTRODUCTION

Le domaine de l'imagerie est l'un des axes de recherche le plus répandu actuellement. Il a connu ces derniers temps des évolutions majeures. Ces évolutions sont surtout liées tant à la multiplicité des techniques qu'à l'exploitation des images produites.

L'imagerie est utilisée dans plusieurs applications comme :

- l'analyse et la recherche d'information : imagerie médicale, télédétection, …
- la présentation d'information : télévision, publicité, …
- l'amélioration du transfert d'information appliquée à la vidéoconférence ou aux sites web.

Dans la modélisation mathématique, une image numérique couleur est représentée par un tableau bidimensionnel de pixels dont chacun contient une information couleur. Ce dernier doit être représenté par trois composantes à valeurs entières positives. Ainsi, tous les traitements d'images numériques couleur doivent prendre en compte un à un ces trois composantes et de les consolider.

En 1996, S.J. Sangwine et S.C. Pei [15] ont proposé une nouvelle modélisation qui consiste à représenter une couleur par un nombre quaternion. Basée sur l'algèbre quaternionique, elle permet de généraliser tous les traitements d'image en niveau de gris et de les présenter en niveau de couleur.

Ce mémoire répond à des questions sur la nécessité de la modélisation quaternionique dans la démarche usuelle de traitement d'images couleur.

Le travail est divisé en trois chapitres. En premier lieu nous allons voir le concept de l'imagerie numérique, l'ensemble des quaternions et leurs modélisations en couleur. En deuxième chapitre, nous allons développer la détection de contours en utilisant les quaternions. Enfin, nous développerons l'Analyse en Composantes Principales Quaternionique - ACPQ qui est une généralisation de l'Analyse en Composantes Principales - ACP dans l'ensemble des quaternions et l'appliquerons à la compression d'image et à l'amélioration de l'image.

CHAPITRE 1

REGLE DE L'ART

Afin de faciliter la compréhension de notre recherche, les notions de base tels que la représentation de la couleur, le principe de filtrage et la modélisation des images couleur par les quaternions sont tout d'abord développées.

1.1 Représentation de la couleur dans les images numériques

Afin de pouvoir coder des images en couleur, nous avons besoin d'étudier au préalable comment on exprime de manière générale la couleur numériquement. Nous voulons aussi effectuer des opérations sur des images en utilisant différents espaces couleurs, cependant nous analyserons comprendre comment s'opèrent les transitions numériques entre ces différents espaces.

1.1.1 Principe

Dans le cas général, la couleur d'un pixel va être représentée par trois composantes notées C_1, C_2 et C_3, et à ces trois composantes nous faisons correspondre respectivement trois vecteurs directeurs normés $\overrightarrow{OC_1}$, $\overrightarrow{OC_2}$ et $\overrightarrow{OC_3}$ qui forment le repère d'un espace vectoriel d'origine O appelé espace couleur. Dans cet espace, chaque couleur est ainsi représentée par un point C, qui définit le vecteur couleur $\overrightarrow{OC}$ et dont les coordonnées sont les valeurs de composantes C_1, C_2 et C_3. La figure 1.1 illustre ce propos. Une image sera donc représentée dans l'espace couleur (C_1, C_2, C_3) par un nuage de points (cf. figure 1.2).

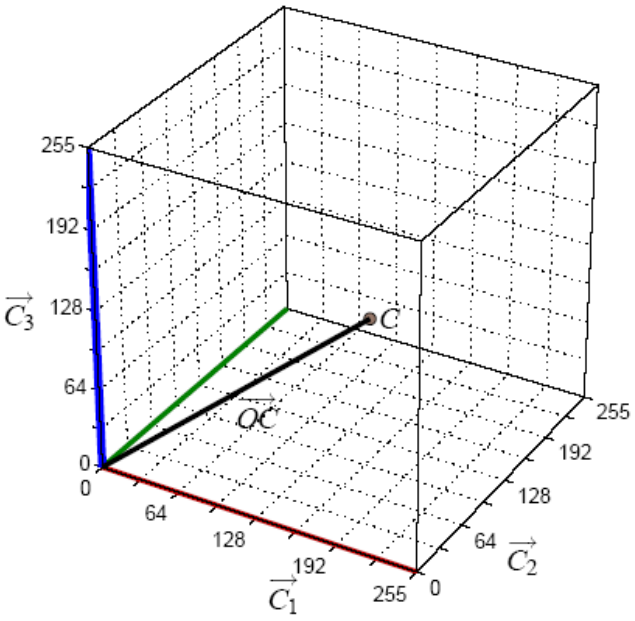


Figure 1.1 Le pixel encadré en bleu dont les composantes couleurs sont C_1, C_2 et C_3 donne naissance à un point C dans l'espace (C_1, C_2, C_3).

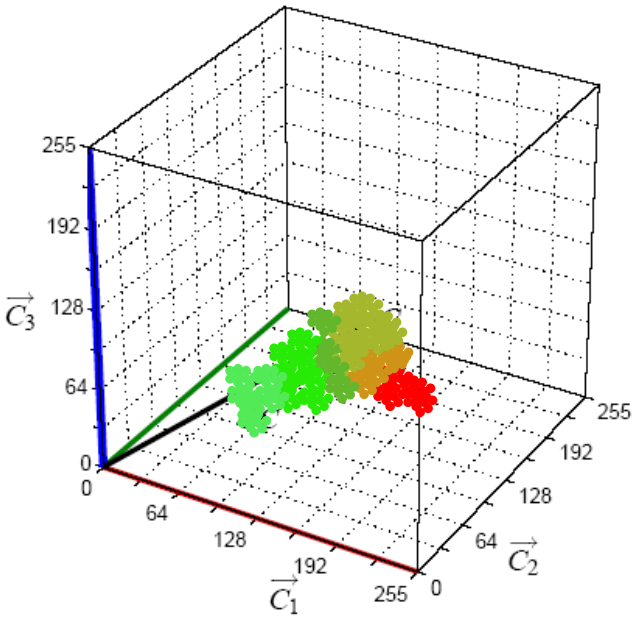


Figure 1.2 Les pixels de l'image donnent naissance à un nuage de points dans l'espace (C_1,C_2,C_3).

1.1.2 L'espace RVB

L'espace de couleurs RVB[1] (Rouge, Vert et Bleu) demeure le plus répandu. En effet, il est implémenté dans la plupart des outils matériels de visualisation : écran, vidéoprojecteur, imprimante, …. Dans cet espace, un pixel est codé par trois composantes Rouge, Vert et Bleu, à valeurs à l'intérieur d'un cube (cf. figure 1.3.a). Cet espace a été développé en fonction des connaissances liées à la vision humaine, les cônes étant plus sensibles à ces trois couleurs. Ce modèle est additif, ce qui signifie que toutes les couleurs sont déduites à partir du noir (R=V=B=0) en ajoutant plus ou moins certaines composantes (cf. figure 1.3.b). Dans cet espace, chaque composante est donc définie par une valeur entre 0 et 255. De ce fait elles doivent être normalisées de la même façon, ce qui est quelquefois contraignant. Le principal inconvénient de ce modèle réside dans la manipulation même des couleurs. En effet, si on veut augmenter la luminosité d'une couleur, il faut incrémenter proportionnellement chaque composante étant donnée la corrélation entre les plans R, V et B. Ces contraintes font du modèle additif des couleurs (RVB), un espace de couleurs peu approprié à la représentation d'images multicomposantes sous la forme d'une image de couleurs.

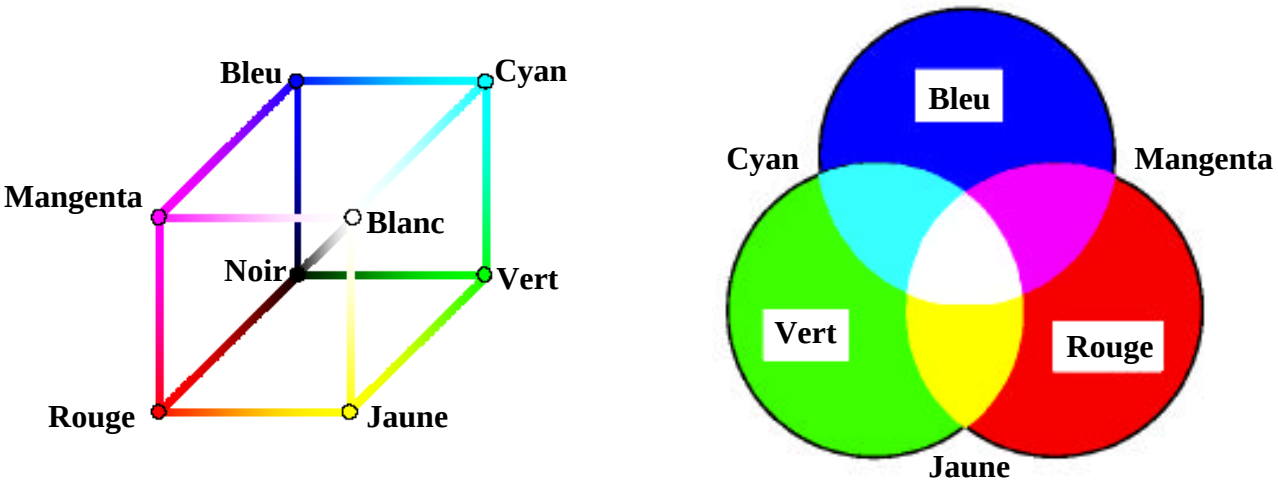


Figure 1.3 (a) : cube RVB (b) **:** composition additive des couleurs

[1] RGB (Red, Green and Blue) en anglais

1.1.3 L'espace TSL

L'espace colorimétrique TSL (Teinte, Saturation, Luminance) a été développé pour offrir une manipulation intuitive des couleurs et permettre une sélection manuelle facile dans les applications interactives de type PAO (Publication Assistée par Ordinateur). Il permet de décomposer une couleur en trois critères physiologiques :

- la teinte qui correspond à la perception de la couleur, $0 \leq T \leq 360$;

- la saturation qui correspond à la pureté de la couleur (vif ou terne), $0 \leq S \leq 1$;

- la luminance correspondant à la quantité de lumière de la couleur (clair ou sombre), $0 \leq L \leq 1$.

Le modèle de couleurs TSL est utilisé pour la manipulation de la teinte et de la saturation car il permet de modifier directement ces valeurs (contrairement au modèle RVB). Les trois composantes TSL définissent un cône représenté dans la figure 1.4.a où l'ensemble des couleurs représentables y est également synthétisé (cf figure 1.4.b). Le principal avantage de cet espace est que chacune de ses composantes est reliée à une grandeur physique facilement interprétable visuellement. Ainsi augmenter la luminosité d'une couleur se fera uniquement en augmentant la composante L. Cette propriété permet de s'affranchir de la corrélation entre la teinte, la luminance et la saturation et offre ainsi un contrôle plus souple dans la manipulation des couleurs.

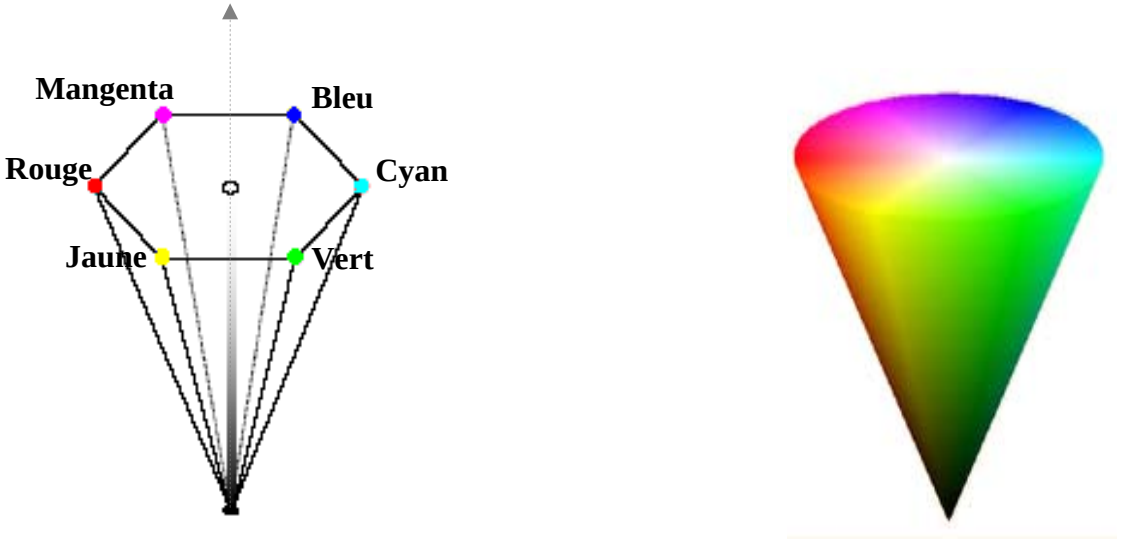


Figure 1.4 (a) : cône TSL (b) **:** ensemble de couleur TSL

La figure 1.4 nous montre :

- L'axe vertical représente la **luminance**, plus elle est grande, plus la couleur est lumineuse ;
- L'écart par rapport à l'axe central représente la **saturation**, le taux de pureté de la couleur ;
- Le choix de l'angle et de la couleur représente la teinte.

1.2 Filtrage d'images numériques

Dans le cas général, on parle du filtrage d'un signal (cf. annexe), Mais dans ce paragraphe, nous allons parler le cas du filtrage des images.

1.2.1 Principes généraux

Dans l'imagerie, le filtrage est utilisé :

- pour réduire le bruit dans l'image ;
- pour détecter les contours d'une image (voir chapitre 2) ;
- pour faire l'opération de voisinage qui effectue une combinaison linéaire (ou non) de pixels de l'image I, produisant une nouvelle image I'

Par définition, un bruit est un phénomène parasite aléatoire suivant une distribution de probabilité connue ou non dont les origines sont diverses : capteur, acquisition, lumière, ... Dans le cas du filtrage linéaire, on considère que le bruit est additif, c'est-à-dire si I_b est l'image bruité alors on peut l'écrire sous la forme :

$$I_b(i, j) = I(i, j) + b(i, j) \tag{1.1}$$

Avec I l'image originale et b le bruit.

1.2.2 Filtrage spatial linéaire

1.2.2.1 Produit de convolution

Le produit de convolution d'une image I (i, j) par un filtre h(i, j) est donné par la formule :

$$I'(i,j) = (I \times h)(i,j) = \sum_{n=1}^{N} \sum_{m=1}^{M} I(i+n, j+m)h(n,m) \tag{1.2}$$

En général, h est un masque carré de taille d impaire, et on a alors :

$$I'(i,j) = (I \times h)(i,j) = \sum_{n=-\frac{d-1}{2}}^{\frac{d-1}{2}} \sum_{m=-\frac{d-1}{2}}^{\frac{d-1}{2}} I(i+n, j+m)h(n,m) \tag{1.3}$$

1.2.2.2 Lissage par moyennage

Pour le lissage par moyennage, la valeur d'un pixel de l'image est relativement similaire à celle de ses voisins. Dans le cas où l'image contient un bruit et que la propriété précédente est préservée, un moyennage local peut atténuer ce bruit. Cette opération est appelée lissage ou « smoothing ».

Pour effectuer un moyennage dans un bloc voisinage de taille d x d, on obtient la sortie I'

$$I'(i,j) = \frac{1}{d^2} \sum_{n=-\frac{d-1}{2}}^{\frac{d-1}{2}} \sum_{m=-\frac{d-1}{2}}^{\frac{d-1}{2}} I(i+n, j+m) \tag{1.4}$$

1.2.2.3 Lissage gaussien

Le noyau gaussien centré et d'écart-type σ est défini par :

$$g_\sigma(i,j) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{i^2+j^2}{2\sigma^2}} \tag{1.5}$$

C'est un lissage par moyennage pondéré de l'image en fonction de la distance du pixel voisin. Il a les propriétés suivantes :

- Si σ est plus petit qu'un pixel, le lissage n'a presque pas d'effet ;

- Plus σ est grand, plus on réduit le bruit, mais plus l'image filtrée est floue ;

- Si σ est choisi trop grand, tous les détails de l'image sont perdus.

On doit alors trouver un compromis entre la quantité de bruit à enlever et la qualité de l'image en sortie.

1.2.3 Filtrage fréquentiel

1.2.3.1 Définition et principe

Le filtrage fréquentiel a pour but de garder ou de supprimer des fréquences de l'image à l'aide d'un filtre. C'est le produit entre les spectres de l'image et du filtre[2].

Le principe général du filtrage fréquentiel suit les étapes suivantes :

- Calculer la transformée de Fourier X(f) du signal x(t) à filtrer ;

- Calculer la transformée de Fourier F(f) du filtre f (t) ;

- Multiplier les spectres $X_{filtré}(f) = X(f)H(f)$;

- Calculer la transformée de Fourier inverse du spectre obtenu pour obtenir le signal filtré $x_{filtré}(t)$

Il y a trois familles dans le filtrage fréquentiel : le filtrage passe-bas, le filtrage passe-haut et le filtrage passe-bande.

---

[2] Dans le domaine spatial, on calcule le produit de convolution entre l'image et le filtre

1.2.3.2 Filtrage passe-bas

Un filtre passe-bas idéal est un système linéaire ne modifiant pas ou peu les basses fréquences de l'image d'entrée.

Ainsi les basses fréquences et la fréquence fondamentale sont conservées c'est-à-dire que l'information d'intensité est restituée lors de la reconstruction de l'image. Les hautes fréquences sont par contre éliminées : les changements brusques d'intensité, tels les bruits ou les frontières, sont atténués voire éliminés ; on parle d'étalement des frontières dans ce cas. On obtient alors une image reconstruite qui présente du flou sur le contour.

1.2.3.3 Filtrage passe-haut

Un filtre passe-haut est un système linéaire ne modifiant pas ou peu les hautes fréquences de l'image d'entrée.

Dans ce cas, les basses fréquences et la fréquence fondamentale sont éliminées car l'information d'intensité est enlevée lors de la reconstruction de l'image. Les hautes fréquences quant à elles, sont préservées c'est à dire que les changements brusques d'intensité sont mis en évidence. Dans ce cas, l'image reconstruite n'a plus ses couleurs mais le contour est net.

1.2.3.4 Filtre passe-bande

Un filtre passe-bande est complémentaire d'un filtre passe-bas et d'un filtre passe-haut. C'est un système linéaire qui préserve une plage de fréquences. L'image reconstruite est alors une combinaison d'un nombre réduit d'images de base.

1.3 Corps non commutatif des quaternions ($\mathbb{H}$, +, x)

Après avoir parlé les concepts de base de l'imagerie et de couleur, nous allons développer dans ce paragraphe l'ensemble des quaternions ou hypercomplexes $\mathbb{H}$.

1.3.1 Définition de l'ensemble $\mathbb{H}$

Les quaternions sont un système de nombres hypercomplexes de dimension 4. Ils forment un corps non commutatif, noté $\mathbb{H}$, découvert par Sir W.R. Hamilton [05]. Un quaternion est composé d'une partie réelle et de trois parties imaginaires :

$$q = a + ib + jc + kd \tag{1.6}$$

Où a, *b, c, d* $\in \mathbb{R}$ et les nombres imaginaires i, j et k sont reliés de la façon suivante :

$$\begin{aligned} i^2 = j^2 = k^2 = ijk = -1 \\ ij = k = -ji \\ jk = i = -kj \\ ki = j = -ik \end{aligned} \tag{1.7}$$

Si a= 0, q est un *quaternion pur*.

1.3.2 Opérations sur $\mathbb{H}$

Soient les quaternions suivants :

$$q = a + ib + jc + kd\ ;$$

$$q_1 = a_1 + ib_1 + jc_1 + kd_1\ ;$$

$$q_2 = a_2 + ib_2 + jc_2 + kd_2$$

1.3.2.1 Addition

On définit la somme de deux quaternions q_1 et q_2 par

$$q_1 + q_2 = (a_1 + a_2) + i(b_1 + b_2) + j(c_1 + c_2) + k(d_1 + d_2) \tag{1.8}$$

1.3.2.2 Multiplication

On définit également le produit $q_1 \times q_2$ de deux quaternions q_1 et q_2 par l'expression

$$\begin{aligned} q_1 \times q_2 = {} & (a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2) + i(a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2) \\ & + j(a_1c_2 + c_1a_2 - b_1d_2 + d_1b_2) + k(d_1a_2 + a_1d_2 + b_1c_2 - c_1b_2) \end{aligned} \tag{1.9}$$

Cette expression nous montre que le produit de deux quaternions n'est pas commutatif.

1.3.2.3 Conjugaison

Le conjugué de q est donné par :

$$\bar{q} = a - ib - jc - kd \tag{1.10}$$

1.3.2.4 Module

Son module est

$$|q| = \sqrt{a^2 + b^2 + c^2 + d^2} \tag{1.11}$$

Si $|q| = 1$, q est un *quaternion unitaire*.

1.3.2.5 Inverse

L'inverse d'un quaternion q non nul est donné par

$$q^{-1} = \frac{\bar{q}}{|q|^2} \tag{1.12}$$

1.3.3 Représentation des quarternions

Il existe différentes façons de représenter les quaternions.

1.3.3.1 Représentation cartésienne

Le quaternion $q \in \mathbb{H}$ est représenté sous forme cartésienne dans l'équation suivante (avec $a,b,c,d \in \mathbb{R}$) :

$$q = a + ib + jc + kd \tag{1.6}$$

1.3.3.2 Représentation scalaire-vecteur

Il est possible de séparer un quaternion en sa partie scalaire et sa partie vectorielle (ou imaginaire) :

$$q = S(q) + V(q) \tag{1.13}$$

avec $S(q) = a = \frac{1}{2}(q + \bar{q})$ et $V(q) = ib + jc + kd = \frac{1}{2}(q - \bar{q})$

1.3.3.3 Représentation polaire

Comme dans le cas des nombres complexes, il est possible de définir une notation polaire pour les quaternions. On peut l'appréhender en considérant la formule d'Euler, qui reste valable pour les quaternions, donnée par :

$$q = |q| (\cos\Phi + \mu \sin\Phi), \tag{1.14}$$

où μ est un quaternion pur et unitaire ($|\mu| = 1$ et $S(\mu)=0$).

Il est possible de réécrire l'expression d'Euler pour un quaternion comme :

$$q=|q|e^{\mu\Phi}, \tag{1.15}$$

Les expressions des diverses composantes de cette représentation sont données par :

$$\mu = \frac{V(q)}{|V(q)|}, \text{ qui est appelé l'}\textit{axe principal}\text{ de q ;}$$
$$\Phi = \tan^{-1}\frac{|V(q)|}{S(q)}, \text{ appelé } \textit{l'angle principal}\text{ du quaternion.} \tag{1.16}$$

Cette formulation des quaternions est proche de celle des complexes.

1.3.3.4 Représentation de Cayley-Dickson

Pour tout $q = a + ib + jc + kd \in \mathbb{H}$, on peut écrire q sous sa forme de Cayley-Dickson :

$$q = z_1 + z_2 j \tag{1.17}$$

avec $z_1, z_2 \in \mathbb{C}$ tels que $z_1 = a + ib$ et $z_2 = c + id$

1.3.3.5 Représentation symplectique

Cette représentation est la généralisation de la notation de Cayley-Dickson. En effet, tout quaternion exprimé dans la base d'origine (1,i,j,k) peut être représenté dans une autre base orthonormée $(1, \mu_1, \mu_2, \mu_3)$ avec des quaternions pures tel que $\mu_1^2 = \mu_2^2 = \mu_3^2 = -1$, $|\mu_1| = |\mu_2| = |\mu_3| = 1$ et $\mu_1 \mu_2 = \mu_3 = -\mu_2 \mu_1$. Avec ces quaternions μ_1 et μ_2 tels $\mu_1 \perp \mu_2$ on peut donc représenter un quaternion q sous une forme complexe généralisée appelée *représentation symplectique* :

$$q = q_1 + q_2 \mu_2 \tag{1.18}$$

avec q_1, q_2 tels que $q_1 = a' + b'\mu_1$ et $q_2 = c' + d'\mu_1$

Donc :

$$q = (a' + b'\mu_1) + (c' + d'\mu_1)\mu_2 \tag{1.19}$$

On appelle souvent q_1 la partie *simplexe* et q_2 la partie *perplexe*.

1.3.3.6 Représentation matricielle des quaternions

Tandis que l'ensemble $\mathbb{H}$ possède une structure d'algèbre sur $\mathbb{R}$, c'est en particulier un espace vectoriel sur $\mathbb{R}$, de dimension 4, dont on a explicité une base, la famille (1, i, j, k). Tout quaternion s''ecrit alors sous la forme d'une combinaison linéaire des éléments de cette base, c'est-à-dire comme une somme $a \cdot 1 + b \cdot i + c \cdot j + d \cdot k$, avec a, b, c, d des réels.

De par la structure d'algèbre de $\mathbb{H}$, la multiplication (à gauche) par un quaternion q donné, soit l'application $p \rightarrow q \cdot p$, est une application linéaire sur $\mathbb{H}$.

Si q s'écrit a + bi + cj + dk, cette application a pour matrice M, dans la base 1, i, j, k [04] :

$$M = \begin{pmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{pmatrix} \qquad (1.20)$$

En fait $\mathbb{H}$ est isomorphe à l'ensemble des matrices de cette forme. Si l'on note, pour $p \in \mathbb{H}$, M(p) la matrice qui est associée à la multiplication par p à gauche, on a par composition que $M(p_1).M(p_2)$ est la matrice associée à la multiplication par $p_1.p_2$.

$\mathbb{H}$ s'identifie donc à la sous-algèbre de $M_4(\mathbb{R})$ engendrée par des matrices de la forme ci-dessous:

$$1 = I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad i = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad j = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix} \quad k = \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad (1.21)$$

1.3.4 Quaternion et rotation dans l'espace

Soit $q = iq_1 + jq_2 + kq_3$ un quaternion pur non nul. Alors, l'application :

$$q \rightarrow \mu q \bar{\mu} \tag{1.22}$$

est une *rotation* de q d'un angle 2φ autour de μ, où $\mu = e^{\mu\varphi}$ est un quaternion unitaire ($|\mu|=1$)

1.3.5 Autres transformations géométriques

Les quaternions peuvent être utilisés pour décrire des vecteurs de $\mathbb{R}^3$, en utilisant uniquement leur partie vectorielle (ou de manière équivalente leur partie imaginaire). Il apparaît que des transformations géométriques simples sur des vecteurs de $\mathbb{R}^3$ peuvent être exprimées en utilisant que des additions et des multiplications de quaternions. Ces transformations sont illustrées dans la figure 1.5

- **Réflexion** $q_{refl} = -\mu q \mu$ est la *réflexion* de q par rapport à μ ; (1.23)
- **Projection** $q_{proj} = \frac{1}{2}(q - \mu q \mu)$ est la *projection* de q par rapport à μ ; (1.24)
- **Réjection** $q_{rej} = \frac{1}{2}(q + \mu q \mu)$ est la *réjection* de q par rapport à μ ; (1.25)

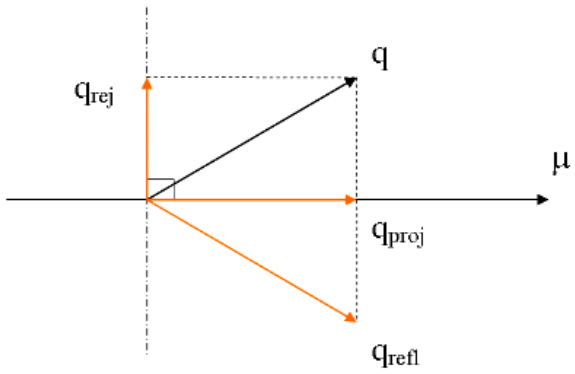


Figure 1.5 Les différentes transformations géométriques : q est un quaternion pur représentant un vecteur couleur ; μ représente un autre vecteur couleur.

1.3.6 Matrice quaternionique

1.3.6.1 Définition

Une matrice I de dimension NxM est quaternionique lorsque les éléments de la matrice sont constitués de quaternions :

$I \in \mathbb{H}^{NxM}$, c'est-à-dire, pour tout i,j tels que $1 \leq i \leq N$ et $1 \leq j \leq M$ on a $I(i,j) \in \mathbb{H}$ (1.26)

1.3.6.2 Multiplication de deux matrices

$\mathbb{H}$, corps des quaternions, étant un corps gauche, c'est-à-dire non-commutatif, il existe deux manières de multiplier les matrices quaternioniques : le produit hamiltonien, qui respecte l'ordre des facteurs, et le produit octonionique, qui ne le respecte pas.

Etant données les deux matrices $U = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}$ et $V = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{pmatrix}$

Le produit hamiltonien respecte l'ordre des facteurs

$$UV = \begin{pmatrix} u_{11}v_{11} + u_{12}v_{21} & u_{11}v_{12} + u_{12}v_{22} \\ u_{21}v_{11} + u_{22}v_{21} & u_{21}v_{12} + u_{22}v_{22} \end{pmatrix} \quad (1.27)$$

Le produit octonionique ne respecte pas l'ordre des facteurs : sur la diagonale principale, il y a commutation des deuxièmes produits et sur la deuxième diagonale il y a commutation des premiers produits

$$UV = \begin{pmatrix} u_{11}v_{11} + v_{21}u_{12} & v_{12}u_{11} + u_{12}v_{22} \\ v_{11}u_{21} + u_{22}v_{21} & u_{21}v_{12} + v_{22}u_{22} \end{pmatrix} \quad (1.28)$$

Tout au long de ce mémoire, nous utilisons le produit hamiltonien.

1.3.7 Domaines d'application des quaternions

Les quaternions sont utilisés dans de nombreuses branches de la science, dont le traitement du signal, le traitement des images, la mécanique quantique, ...

En traitement d'images plus particulièrement, les quaternions peuvent s'appliquer dans la détection des contours, le tatouage, la transformation géométrique trois dimensions (3D) et la compression.

1.4 Codage quaternionique des images couleurs

L'utilisation des quaternions pour modéliser et traiter les images numériques couleur a été initialement proposée en 1996, simultanément par S.J. Sangwine et S.C. Pei [15]. La définition d'une image numérique couleur est vue comme une image dont les pixels ont des valeurs dans $\mathbb{H}$. La nature cartésienne d'un repère dans $\mathbb{H}$ a conduit à choisir l'expression d'une image dans l'espace RGB pour le modèle quaternionique des images couleurs.

1.4.1 Definition

Une image numérique couleur $\mathbf{I}(n,m)$ *comportant N lignes et M colonnes (*$n = 1, ...,N$ *et* $m=1,...,M$*) est une image dont les pixels sont des quaternions purs* [15] *tels que :*

$$\mathbf{I}(n,m) = \mathbf{I}_R(n,m)\mathrm{i} + \mathbf{I}_V(n,m)\mathrm{j} + \mathbf{I}_B(n,m)\mathrm{k} \tag{1.29}$$

où $\mathbf{I}_{R,V,B}(n,m)$ *sont les images des trois canaux Rouge, Vert et Bleu.*

Une image numérique couleur de *NxM* pixels (*N* lignes et *M* colonnes) est donc une matrice dont les éléments sont des quaternions *purs*. Un pixel couleur est représenté comme un quaternion à partie réelle nulle.

Ce modèle d'images couleur à valeurs dans $\mathbb{H}$ s'inscrit dans un cadre plus global de modélisation des images multidimensionnelles et multispectrales à l'aide des nombres hypercomplexes.

1.4.2 Séparation en partie simplexe et partie perplexe

La décomposition sympléctique (cf section 1.3.3.5) a été utilisée sur les pixels couleurs des images codées avec des quaternions afin de séparer l'information de couleur en une partie de luminosité et une partie de chromaticité. Deux quaternions purs sont utilisés pour cette décomposition : le premier μ_1 est l'axe des niveaux de gris ($\mu_1 = \mu_{gris} = \frac{i+j+k}{\sqrt{3}}$) ; le second est l'axe perpendiculaire à μ_1 dans la direction de la couleur rouge ($\mu_2 = \sqrt{\frac{2}{3}}(i, -\frac{j}{2}, -\frac{k}{2})$).

Lorsqu'on décompose une image couleur en parties parallèle et perpendiculaire à μ_1, il apparait que cette décomposition permet de séparer l'information de luminosité sur la partie parallèle et l'information chromatique sur la partie perpendiculaire à μ_1.

En effet, la partie parallèle à μ_1 (un réel codé sur la partie imaginaire de la partie simplexe) correspond à la projection de tous les pixels couleur sur l'axe des niveaux de gris : c'est une information d'intensité lumineuse. La partie perpendiculaire (nécessitant deux réels, respectivement associé à μ_2 et $\mu_1\mu_2$, pour coder les coefficients de la partie perplexe) correspond à la projection de tous les pixels de l'image sur le plan perpendiculaire à μ_1 et qui passe sur le vecteur μ_2 (on passe par un changement de base pour revenir dans la base i,j,k qui correspond à l'espace RVB). Cette seconde partie est donc considérée comme de l'information chromatique. Si on somme les deux parties simplexe f_1 et perplexe f_2, on obtient de nouveau l'image originale f. Cette décomposition symplectique est utilisée dans l'analyse des images couleur par Ell et Sangwine [15] pour séparer l'information spectrale obtenue après transformée de Fourier en parties chromatique et achromatique.

La figure 1.6 nous montre des exemples de décomposition symplectique des images couleur.

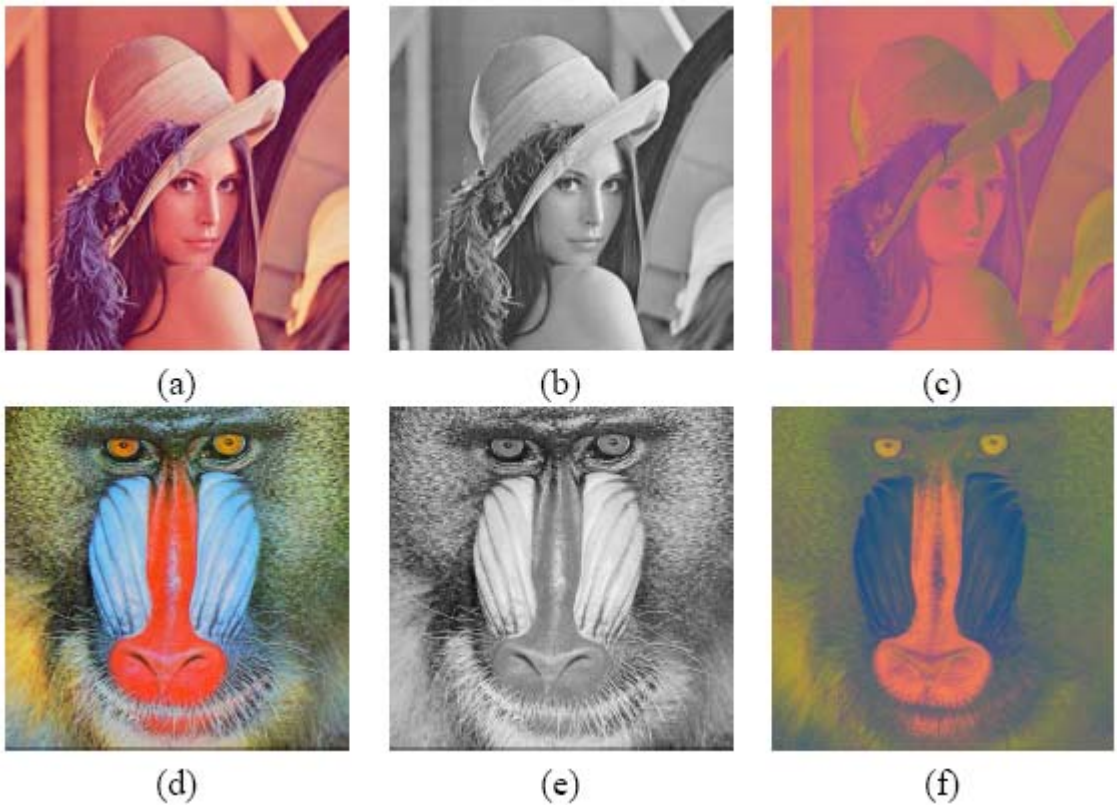


Figure 1.6 Décomposition symplectique sur des images couleur : première colonne images originales ; seconde colonne : parties parallèles ; dernière colonne : parties perpendiculaires

1.4.3 Transformations couleur

Pour chaque couleur décrite par un quaternion dans l'espace RVB on peut faire correspondre son équivalent dans un espace couleur de teinte, saturation et intensité. Nous considérons que cette dernière information est représentée par la norme de la projection du vecteur couleur q sur l'axe des niveaux de gris $\mu_{gris} = \frac{i+j+k}{\sqrt{3}}$. La saturation et la teinte peuvent être représentées par le plan orthogonal à μ_{gris} à l'intersection de cet axe et de l'extrémité du vecteur projeté de q sur μ_{gris}. La saturation est la distance entre l'extrémité du vecteur couleur q et l'axe μ_{gris}. La teinte correspond à l'angle entre la réjection du vecteur q par rapport à μ_{gris} et un vecteur v (vecteur de référence pour une teinte d'angle nul) pris sur le plan orthogonal à μ_{gris} (cf figure 1.7).

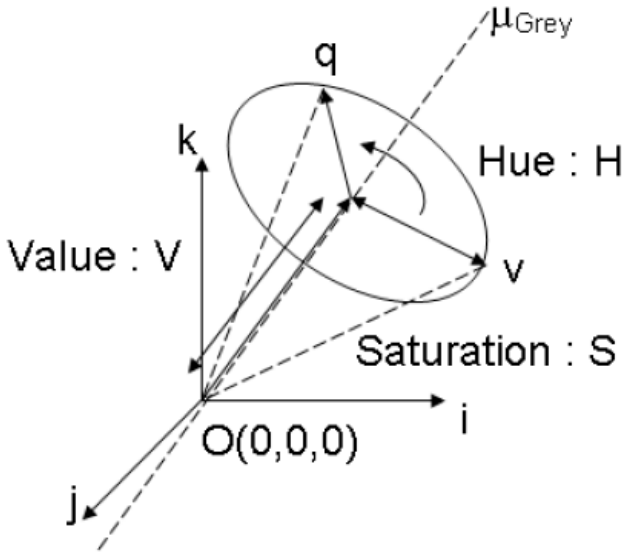


Figure 1.7 Teinte (Hue), Saturation et Clarté (Value) obtenues avec μ_{gris} et T(v) = 0

La transformation couleur en TSL se résume par les équations suivantes :

$$\begin{cases} T = \tan^{-1} \dfrac{|q - \mu v q v \mu|}{|q - v q v|} \\ L = \left| \dfrac{1}{2}(q - \mu q \mu \right| \\ S = \left| \dfrac{1}{2}(q + \mu q \mu \right| \end{cases} \tag{1.30}$$

CHAPITRE 2

DETECTION DE CONTOUR PAR ALGEBRE QUATERNIONIQUE

De nombreuses applications dans l'analyse d'images, telles l'imagerie satellitaire ou encore l'imagerie médicale, dépendent de la détection de contour. Ce chapitre démontre la détection de contour par l'utilisation de l'algèbre quaternionique.

2.1 Définition

La détection de contour est l'étude de la détection des discontinuités qui est une problématique fondamentale du traitement des images en général

Les contours constituent en effet des indices riches, au même titre que les points d'intérêts, pour toute interprétation ultérieure de l'image. Les contours dans une image proviennent des :

- discontinuités de la fonction de réflectance (texture, ombre),
- discontinuités de profondeur (bords de l'objet).

Ils sont caractérisés par des discontinuités de la fonction d'intensité dans les images.

Le principe de la détection de contours repose donc sur l'étude des dérivées de la fonction d'intensité dans l'image : les extréma locaux du gradient de la fonction d'intensité et les passages par zéro du laplacien. La difficulté réside dans la présence de bruit dans les images. Les modèles de contours utilisés sont ceux de contours idéalisés, comme ceux présentés sur la figure 2.1 [10].

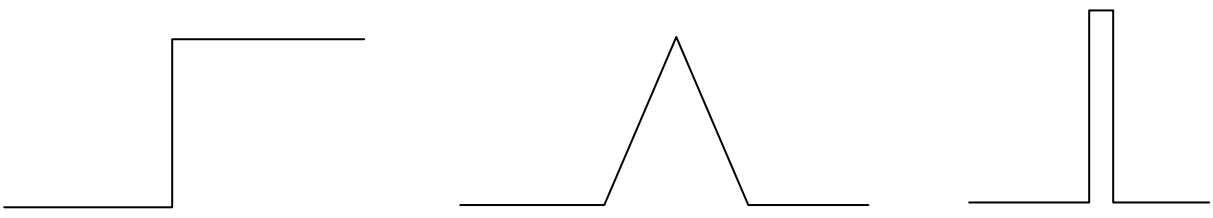


Figure 2.1 Quelques modèles de contours : marche d'escalier, toit et pointe. Le plus utilisé est celui en marche d'escalier

2.2 Les approches classiques

Cette section présente un ensemble de méthodes qui ont eu historiquement une grande importance sur le traitement des images.

2.2.1 Méthodes marginales

Les méthodes marginales reposent sur l'extension des traitements sur les images en niveaux de gris en effectuant séparément sur chaque canal couleur un traitement. Elles bénéficient donc d'une implémentation relativement simple car il suffit d'adapter les traitements déjà existants sur chacune des matrices de pixels correspondant aux composantes couleurs. Cependant, comme elles ne prennent pas l'information couleur dans sa globalité, vu qu'elles ne considèrent pas les corrélations (interactions) entre les composantes, les résultats font souvent apparaître des incohérences en termes de perception (fausses couleurs).

(a_1)

(b_1)

(c_1)

(d_1)

(e_1)

(f_1)

(a_2) **(b_2)** **(c_2)**

(d_2) **(e_2)** **(f_2)**

Figure 2.2 Méthode marginal : (a_i) images originales, (b_i) moyennes des contours R V et B, (c_i) maximum des contours R V et B, (d_i) contours couleur Rouge, (e_i) contours couleur Vert, (f_i) contours couleur Bleu

Les images dans la figure 2.2 nous montrent les contours sur les trois canaux séparés de couleur : Rouge, Vert et Bleu ; et de les consolider par la moyenne et par le maximum. Les formules ci-après nous donnent les modes de calcul.

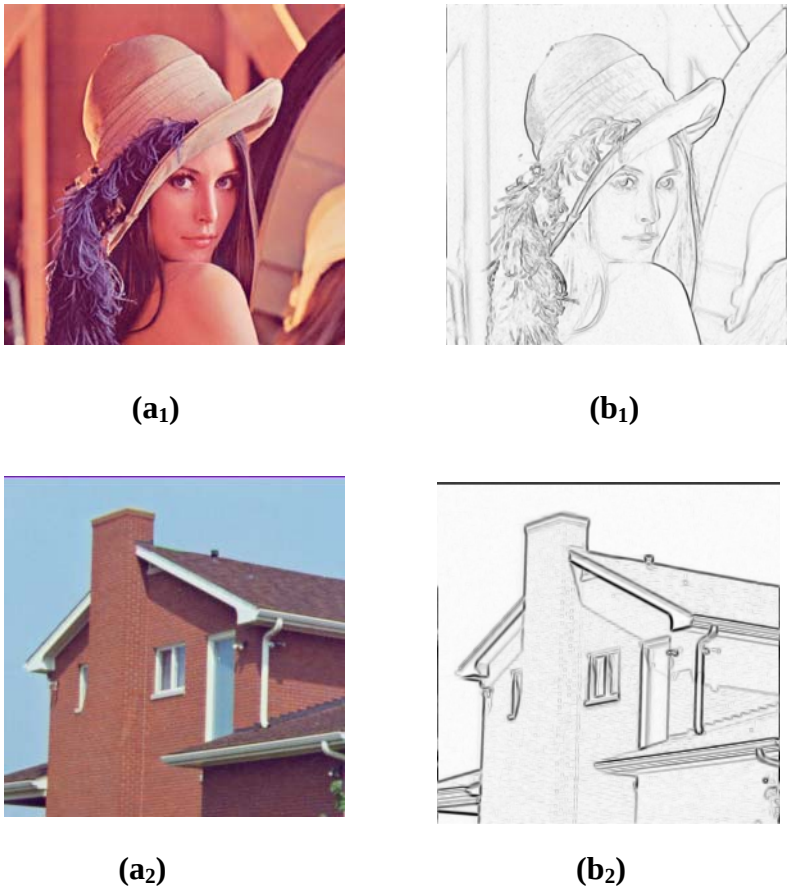
Pour la moyenne : $C_{moyenne} = \frac{1}{3} \sum_{Coul=R,V,B} Countour_{Coul}$ (2.1)

Et pour le maximum : $C_{max} = \frac{1}{3} \max_{Coul=R,V,B} (Countour_{Coul})$ (2.2)

2.2.2 Méthodes vectorielles

Pour éviter les inconvénients des méthodes marginales, les méthodes vectorielles considèrent les pixels des images comme des vecteurs couleurs. Les traitements sont alors effectués de manière globale sur la couleur. Le problème de ce type de méthode repose à la fois sur le choix de l'espace de codage, pour faire correspondre l'information couleur à une information vectorielle et surtout sur la modélisation et l'exploitation mathématiques des données.

Celui-ci, associé à un champ vectoriel, permet de rechercher les variations locales de l'image, autrement dit les contours. La plus grande valeur propre du tenseur correspond alors à la norme de ce gradient vectoriel qui détecte les contours couleur. Comparées aux méthodes marginales, les méthodes vectorielles obtiennent de meilleurs résultats perceptuels mais au prix d'une plus grande complexité.



(a_1) **(b_1)**

(a_2) **(b_2)**

Figure 2.3 Méthode vectoriel : (a_i) images originales, (b_i) contours en utilisant les distances vectorielles

La formule mathématique de cette méthode s'écrit :

$$C_{vect} = \sqrt{\sum_{Coul=R,V,B} (pixel_1 - pixel_0)^2_{Coul}} \quad (2.3)$$

Dont les $pixel_1$ et $pixel_0$ sont les pixels à comparer et la couleur est considérée comme un vecteur de trois composantes : rouge, vert et bleu.

2.2.3 Méthodes perceptuelles

Les méthodes perceptuelles sont basées sur des caractéristiques du système visuel humain (SVH). Deux méthodes sont proposées, la première considère qu'à faible niveau de saturation, la teinte n'est pas pertinente, elle n'est alors pas utilisée. Cependant lorsque la saturation est faible, le gradient proposé rajoute l'information de teinte.

La deuxième méthode privilégie l'information de teinte, les informations de saturation et de luminance ne sont prises en compte que lorsque celle-ci n'est pas pertinente pour séparer les couleurs en terme de perception humaine. En effet, à teinte égale, deux couleurs auront l'air de présenter une plus grande différence lorsqu'elles sont fortement saturées que lorsqu'elles le sont moins.

2.3 Les approches quaternioniques

2.3.1 Filtrage quaternionique

2.3.1.1 Quaternion convolution

Le plus utilisé en filtrant les images est la convolution de l'image par un masque (cf. section 1.2.2.1). Dans le cas des quaternions, un masque d'un quaternion peut être défini comme celle d'un masque réel. Etant donné que le produit quaternionique n'est pas commutatif, il est possible de définir plusieurs produits de convolution, le produit de convolution à droite est défini ici pour un signal en deux dimensions.

Généralement, nous pouvons définir un filtre quaternionique par [16] :

$$Q'(x,y) = \sum_{s=-r}^{r} \sum_{t=-r}^{r} h_L(s,t) \times Q(x+s, y+t) \times h_R(s,t) \tag{2.4}$$

avec Q'(x,y) est le pixel filtré et Q(x,y) est un pixel dans l'image origine, $h_L(s,t)$ et $h_R(s,t)$ sont les masques gauche et droite respectivement, les deux ont une dimension de (2r+1)x(2r+1). L'opération « × » indique la multiplication entre deux nombres quaternioniques.

2.3.1.2 Lissage des images couleur

Evans et Sangwine ont proposé une méthode de filtrage pour lisser des composantes particulières dans une image couleur. Supposons que nous voulons lisser une image à une direction parallèle à une couleur C qui est un quaternion pur, mais en respectant la direction orthogonal à cette couleur.

L'opération de lissage d'une matrice M d'une image peut être obtenue à partir d'un masque de 3x3 défini par [17] :

$$\text{Uc} = \frac{1}{9} \begin{pmatrix} \hat{c} & \hat{c} & \hat{c} \\ \hat{c} & \hat{c} & \hat{c} \\ \hat{c} & \hat{c} & \hat{c} \end{pmatrix} \tag{2.5}$$

Avec $\hat{c}$, un quaternion unitaire pur de même direction que C.

2.3.2 L'approche quaternionique de Sangwine

Les travaux de Sangwine et Al. [16] sur le filtrage spatial proposent de généraliser les filtres linéaires classiques à l'utilisation des quaternions pour les images couleur. Une fois un filtre linéaire quaternionique défini, il suffit de le convoluer avec l'image couleur pour obtenir l'image couleur filtrée.

Sangwine a proposé un détecteur de contours basé sur la rotation des couleurs des pixels de l'image d'un angle π autour de l'axe des gris (cf section 1.3.4). Le vecteur couleur résultat de cette rotation est comparé aux vecteurs couleurs des pixels voisins (cf figure 2.2).

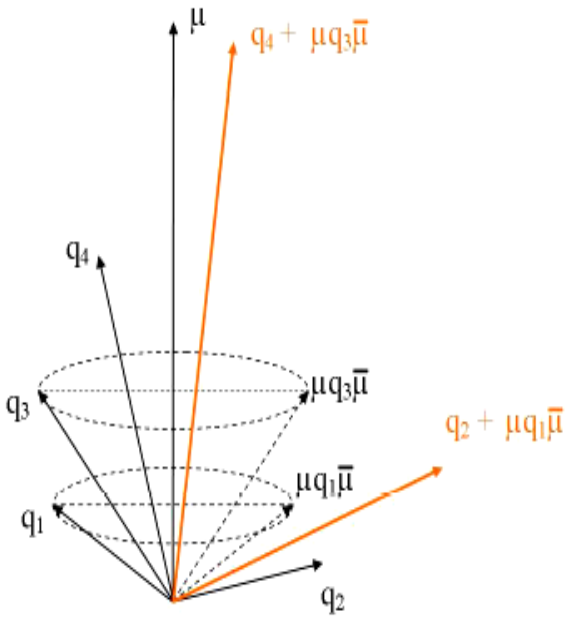


Figure 2.4 Schéma du détecteur de contours de Sangwine.

Dans cette figure, μ est l'axe des gris ; $\mu q_1 \mu$ (resp. $\mu q_3 \mu$) est la rotation du vecteur q_1 (resp. q_3) autour de μ et d'angle π ; le vecteur de comparaison entre q_1 et q_2 (resp. entre q_3 et q_4) est donné par $q_2 + \mu q_1 \mu$ (resp. $q_4 + \mu q_3 \mu$) ; $q_4 + \mu q_3 \mu$ est proche de l'axe des gris et $q_2 + \mu q_1 \mu$ en est plus éloigné donc le détecteur de Sangwine peut détecter un contour avec ce vecteur qui est plus coloré que le précédent.

L'opération de filtrage proposée par Sangwine [16] est la suivante :

$$f_{filtrée}[m, n] = (h_1 \, x \, f \, x \, h_2)[m, n] \tag{2.6}$$

où les filtres h1 et h2 sont une paire de filtres conjugués avec Q = $e^{\mu\frac{\pi}{2}}$, μ = $\mu_{gris} = \frac{i+j+k}{\sqrt{3}}$ l'axe des gris et :

$$h_1 = \frac{1}{6}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ Q & Q & Q \end{pmatrix} \text{ et } h_2 = \frac{1}{6}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ \overline{Q} & \overline{Q} & \overline{Q} \end{pmatrix} \tag{2.7}$$

A première vue, l'image résultat (cf. figure 2.2) ne donne pas l'impression d'être pertinente pour détecter les contours. En effet, il semble qu'elle soit constituée uniquement d'information en termes de niveaux de gris. Cependant ce n'est pas le cas, car si on y regarde de plus près, seules les zones de couleurs homogènes de l'image d'origine apparaissent en niveaux de gris. En effet, le résultat affiche le vecteur de comparaison qui est un quaternion pur représentant des couleurs RVB. Ce vecteur est le résultat de l'opération géométrique qui consiste à ajouter deux vecteurs couleurs entre eux. Le premier terme de la somme est le pixel couleur à analyser. Le second terme est calculé par la rotation du vecteur correspondant à la moyenne des pixels voisins du pixel analysé d'un angle de π par rapport à l'axe des gris. Lorsque le pixel analysé est situé dans une zone homogène de l'image d'origine, comme c'est le cas des vecteurs q_3 et q_4 dans l'exemple, le vecteur résultat, $q_4 + \mu q_3 \mu$ sur l'exemple, est un vecteur couleur très proche de l'axe des niveaux de gris, autrement dit, il a une faible saturation. Le résultat apparaîtra comme une couleur proche d'un niveau de gris. En revanche, si le pixel analysé se situe sur une zone de contour, autrement dit, ses pixels voisins sont en opposition de couleur (comme q_1 et q_2 sur l'exemple), le vecteur couleur de comparaison ($q_2 + \mu q_1 \mu$) sera éloigné de l'axe des niveaux de gris. Autrement dit, ce vecteur aura une forte saturation et par conséquent les contours apparaîtront colorés du fait de cette plus grande distance.

2.3.3 Détection de contours par maximum de distance couleur

En reprenant le principe du détecteur de contours de Sangwine [16] [17] nous proposons de créer un gradient quaternionique. Pour chaque pixel, par exemple q_1 (resp. q_3), nous avons un vecteur de comparaison avec son voisin q_2 (resp. q_4) obtenu par la méthode de Sangwine. Nous avons vu que plus la différence de couleur était grande entre pixels voisins, plus le vecteur de comparaison résultant était éloigné de l'axe des niveaux de gris. Nous proposons donc de calculer la distance q_{dist} entre ce vecteur de comparaison et l'axe des niveaux de gris μ ($q_{sum} = q_2 + \mu q_1 \mu$ ou $q_{sum} = q_4 + \mu q_3 \mu$). Cette distance peut être calculée au moyen d'opérations quaternioniques élémentaires (cf. section 1.3.5) car cette distance est la norme de la réjection du vecteur de comparaison par rapport à μ.

Cette opération de filtrage est appliquée en utilisant des filtres horizontaux, verticaux et dans les deux directions diagonales. On sélectionne ensuite la distance maximale de saturation obtenue

pour chacune de ces directions pour obtenir le gradient couleur final. Nous remarquons que ces opérations sont linéaires mais que le résultat final ne l'est pas.

2.4 Mis en algorithme de la détection de contour quaternionique

2.4.1 Les étapes de l'algorithme

Pour détecter le contour par les quaternions, nous avons suivi les étapes suivantes :

- Choix de l'image ;

- Initialisation des vecteurs : μ_{gris} et $\overline{\mu}_{gris} = -\ \mu_{gris}$ avec $\mu_{gris} = \frac{i+j+k}{\sqrt{3}}$;

- Conversion de l'image en matrice quaternionique ;

- Application du filtrage de Sangwine pixel par pixel, c'est-à-dire, pour chaque élément de la matrice :

  - Calcul de $f_{filtrée}[m, n] = (h_1 \, x \, f \, x \, h_2)[m, n]$ ($Q = e^{\mu\frac{\pi}{2}}$ et $\mu = \mu_{gris}$)

    avec $h_1 = \frac{1}{6}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ Q & Q & Q \end{pmatrix}$ et $h_2 = \frac{1}{6}\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ \overline{Q} & \overline{Q} & \overline{Q} \end{pmatrix}$ pour le filtrage horizontal

    ou $h_1 = \frac{1}{6}\begin{pmatrix} 1 & 0 & Q \\ 1 & 0 & Q \\ 1 & 0 & Q \end{pmatrix}$ et $h_2 = \frac{1}{6}\begin{pmatrix} 1 & 0 & \overline{Q} \\ 1 & 0 & \overline{Q} \\ 1 & 0 & \overline{Q} \end{pmatrix}$ pour le filtrage vertical

  - Calcul de la réjection $rej[m, n] = f_{filtrée}[m, n] + (\mu * f_{filtrée}[m, n] * \mu)$

  - Calcul de la distance $dist[m, n] = abs(rej[m, n])$

  - Après avoir obtenu les distances verticales et horizontales, on prend les maximums des distances par pixel ;

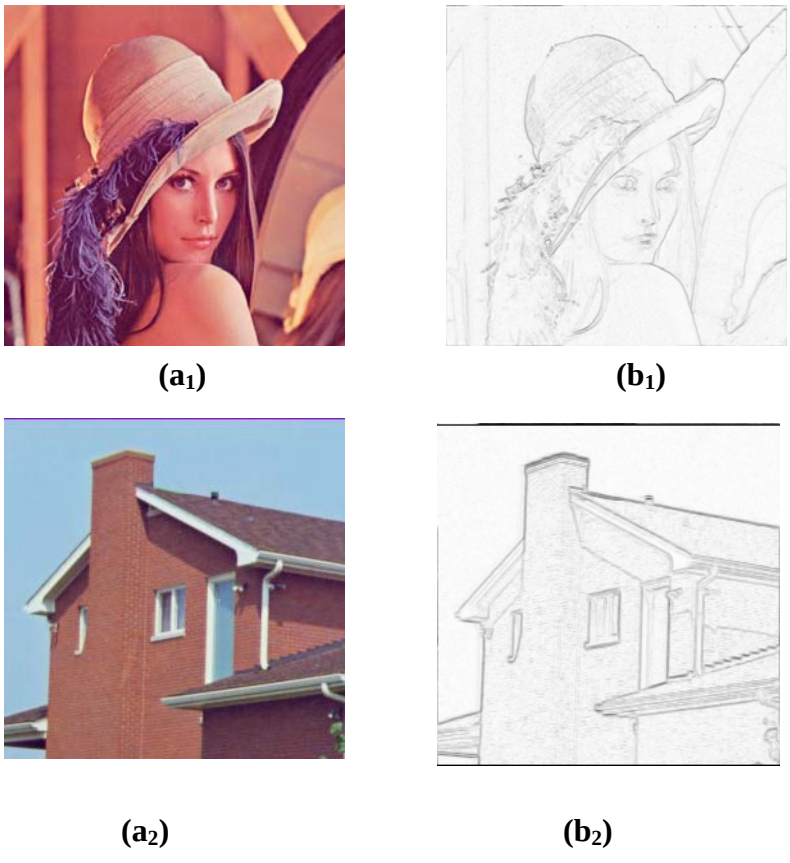
- Pour chaque pixel, la distance calculée définit un contour ou non en comparant cette valeur avec un seuil de contour ;

- Conversion de la matrice de contours en image contour.

2.4.2 Mise en œuvre sur MATLAB

Durant cette recherche sur les quaternions, nous avons utilisé un « toolbox quaternion » développé par Steve Sangwine et Nicolas Le Bihan en mars 2005 [13]. Nous avons utilisé les opérations élémentaires et les fonctions sur les points, les vecteurs et les matrices quaternioniques.

Le code MATLAB se trouve en annexe et la figure suivante montre les résultats :



(a_1) **(b_1)**

(a_2) **(b_2)**

Figure 2.5 Méthode de Sangwine : (a_i) images originales, (b_i) images de contours obtenues

2.5 Amélioration de la détection de contours en utilisant un repère contenant U_{gris}

2.5.1 Preuve sur la performance

Comme l'axe achromatique appartient au repère contenant μ_{gris}, nous avons proposé durant cette étude de faire un changement de repère de RVB en ce nouveau repère. Puis, on applique le même principe de comparaison des vecteurs quaternioniques. Ce procédé permet, pour un même résultat, de gagner énormément en performances pour la détection de contours.

En effet, les opérations sur les produits quaternioniques dans le repère RVB sont plus complexes car le vecteur μ_{gris} (axe des gris) est de la forme $\mu_{gris}/RVB = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$; tandis que dans le nouveau repère $\Re'$, ce dernier devient $\mu_{gris}/\Re' = (1,0,0)$.

2.5.2 Description de l'algorithme correspondant

Nous pouvons voir ci-dessous les étapes de notre proposition :

- Choix de l'image
- Conversion de l'image en matrice quaternionique
- Changement de repère, nous avons utilisé les trois vecteurs u_1, u_2 et u_3 comme nouveau repère avec :

  - $u1 = \frac{1}{\sqrt{3}} (i + j + k)$
  - $u2 = \frac{1}{\sqrt{6}} (2i - j - k)$ (2.8)
  - $u3 = \frac{1}{\sqrt{2}} (\quad j - k)$

- Application de filtrage pixel par pixel, c'est-à-dire, pour chaque élément de la matrice :

  - Calcul de $f_{filtrée}[m, n] = (h_1 \times f \times h_2)[m, n]$ ($Q = e^{\mu\frac{\pi}{2}}$ et $\mu = \mu_{gris(1,0,0)}$)

Dans notre proposition nous avons utilisé le filtre de Canny [10] défini par les deux masques

$$M_1=\begin{pmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{pmatrix} \text{ et } M_2=\begin{pmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{pmatrix} \qquad (2.9)$$

dont il y a 8 filtres issus ces 2 matrices

On a pour les deux premières matrice :

$$h_1 = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2i & 1 \\ i & i & i \end{pmatrix} \text{ et } h_2 = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & -2i & 1 \\ -i & -i & -i \end{pmatrix} \text{ car dans ce cas } Q = i$$

$$\text{et } \overline{Q} = -Q = -i$$

- Calcul de la réjection $rej[m, n] = f_{filtrée}[m, n] + (\mu * f_{filtrée}[m, n] * \mu)$

  $= 2 * f_{filtrée}[m, n]$

- Calcul de la distance $dist[m, n] = abs(rej[m, n])$

- Après avoir obtenu les distances des 8 cas possibles, on prend les maximums des distances par pixel ;

- Pour chaque pixel, la distance calculée définit un contour ou non en comparant cette valeur avec un seuil de contour ;

- Conversion de la matrice de contours en image contour.

La figure ci-dessous nous montre les résultats des détections de contour sur « Lena.tiff » et « House.tiff »

Figure 2.6 (a) Originale, (b) Valeur de la projection (c) Contour en utilisant le filtre de Canny

CHAPITRE 3

ANALYSE EN COMPOSANTES PRINCIPALES QUATERNIONIQUE

L'utilisation intensive de l'Analyse en Composantes Principales (ACP) en traitement du signal et des images traduit l'intérêt de trouver une représentation compacte, c'est-à-dire, de dimension la plus petite possible, du signal ou de l'image acquise. Les applications de l'ACP vont ensuite de la compression à l'analyse.

3.1 Définition

Une image couleur de N lignes et M colonnes peut être modélisée comme une matrice S à valeurs dans $\mathbb{H}^{NxM}$. Il est alors possible de faire une ACP sur ce jeu de données. L'ACP sur $\mathbb{H}$ consiste donc à diagonaliser la matrice de covariance des observations, c'est-à-dire, à en effectuer la décomposition en valeurs propres. La spécificité du modèle quaternionique intervient dans la décomposition en valeurs singulières d'une matrice de quaternions.

3.2 Décomposition en Valeurs Singulières quaternionique

La décomposition en valeurs singulières est une technique importante dans l'algèbre linéaire. Elle joue un rôle intéressant et fondamental dans différentes applications, plus particulièrement, dans le traitement d'images.

La décomposition en valeurs singulières d'une matrice A(m×n) est de la forme :

$$A = U\Sigma V^T \tag{3.1}$$

Avec U une matrice orthogonal m×m ; V une matrice orthogonal n×n et Σ une matrice m×n contenant les valeurs singulières de A avec $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$ sur le diagonal.

3.2.1 Théorème sur l'existence d'un SVD d'une matrice quaternionique

Afin de faire une ACP spatiale sur une image de N lignes et M colonnes, il est nécessaire de calculer la décomposition en valeurs singulières de celle-ci en la considérant comme une matrice de dimension N× M.

Toute matrice $S \in \mathbb{H}^{NxM}$ admet une décomposition en valeurs singulières donnée par [7] [8] :

$$S = U\begin{pmatrix} \sum_r & 0 \\ 0 & 0 \end{pmatrix} V^T \tag{3.2}$$

Avec

- $U \in \mathbb{H}^{NxN}$ et $V \in \mathbb{H}^{PxP}$, matrices carrées unitaires
- $\sum_r \in \mathbb{R}^{rxr}$ matrice diagonale et r le rang de la matrice S (nombre des valeurs singulières non nulles). Les valeurs sur le diagonal, λ_n ($1 \leq n \leq r$), sont des valeurs singulières de la matrice de quaternion et ordonnées de façon décroissante sur la diagonale.

3.2.2 Equivalence d'une matrice quaternionique en matrice complexe

Soit une matrice A de $\mathbb{H}^{NxM}$ dont l'expression est (cf 1.3.3.4 Représentation Cayley Dickson) :

$$A = A_1 + A_2 j, \tag{3.3}$$

où A_1 et A_2 sont des matrices à valeurs complexes.

La matrice complexe adjointe de A, notée χ_A, est la matrice complexe de $\mathbb{C}^{2Nx2M}$:

$$\chi_A = \begin{pmatrix} A_1 & A_2 \\ -\overline{A_2} & \overline{A_1} \end{pmatrix}, \tag{3.4}$$

Cette matrice, entre autres, vérifie les propriétés suivantes :

- $\chi_{AB} = \chi_A \chi_B$
- $\chi_{A+B} = \chi_A + \chi_B$
- $\chi_{\bar{A}} = \overline{\chi_A}$
- χ_A est de rang 2r si A est de rang r

(3.5)

3.2.3 SVD d'une matrice complexe

Toute matrice A de $\mathbb{C}^{NxP}$ admet une décomposition en valeurs singulières donnée par :

$$A = U\Sigma V^T \tag{3.6}$$

Avec

- $U \in \mathbb{C}^{NxN}$ et $V \in \mathbb{C}^{PxP}$, matrices carrées unitaires
- $\Sigma = \text{diag}(\sigma 1, \sigma 2, \ldots, \sigma_{min}(N,P))$, matrice pseudo-diagonale, dont les éléments sont positifs et ordonnés de façon décroissante sur la diagonale.

Les σ_i sont les valeurs singulières de A et les colonnes de U (respectivement de V) sont les vecteurs singuliers gauches (respectivement droits).

3.2.4 Théorème sur la relation entre les matrices SVDQ et SVD complexe équivalent

Dans le cas où le SVD d'une matrice quaternionique et son équivalent en matrice complexe sont respectivement

$$Q_q = U_q \Sigma V_q^T \text{ et } C_c = U_c \Sigma' V_c^T$$

alors

1) $\Sigma = lig_{imp}\,(col_{imp}(\Sigma'))$ et

2) Si $U_c = \begin{bmatrix} [U_c^1]_{nx2n} \\ [U_c^2]_{nx2n} \end{bmatrix}_{2nx2n} = \begin{bmatrix} A_c \\ B_c \end{bmatrix}$, alors $\begin{aligned} U_q &= col_{imp}(U_c^1) + col_{imp}(-\overline{U}_c^2).j \\ &= col_{imp}(A_c) + col_{imp}(-\overline{B}_c).j \end{aligned}$ (3.7)

$lig_{imp}(M)$ et $col_{imp}(M)$ sont les lignes impairs et les colonnes impairs de la matrice M

3.3 Transformation Karhunen-Loeve Quaternionique

La transformation Karhunen-Loeve est une technique plus utilisée dans l'image et le traitement de signal. Elle est basée sur la diagonalisation de la matrice de covariance de lignes (ou colonnes) de l'image couleur [14].

3.3.1 Diagonalisation de matrices de quaternions

La spécificité de la diagonalisation de matrices de quaternions réside en l'existence, du fait du non commutativité du produit sur $\mathbb{H}$, de deux types de valeurs propres : les valeurs propres gauches λg et les valeurs propres droites λ_d. Elles sont définies comme suit :

Soit une matrice de quaternions A dans $\mathbb{H}^{N\times N}$. Ses valeurs propres droites λ_d et gauches λ_g sont définies comme :

$$\begin{aligned} A.v_d &= v_d\lambda_d \\ A.v_g &= \lambda_g v_g \end{aligned} \quad (3.8)$$

avec v_d et v_g les vecteurs propres droits et gauches de A.

La décomposition de la matrice A est :

$$A = W\Omega W^{\Delta} \tag{3.9}$$

Avec

- $W \in \mathbb{H}^{N\times N}$ matrice unitaire contenant les vecteurs propres
- $\Omega \in \mathbb{C}^{NxN}$ matrice diagonal des valeurs propres

Si A est *hermitienne*, c'est-à-dire $A = A^{\Delta}$, alors $\Omega \in \mathbb{R}^{NxN}$

3.3.2 Diagonalisation de la matrice de convariance

Etant donnée une image S de dimension N × M, on construit une matrice Γ_i à partir de s_i, la i-ème colonne de S par :

$$\Gamma i = \tilde{s}_i \tilde{s}_i^{\Delta} \tag{3.10}$$

avec $\tilde{s}_i = s_i - m[s_i]$ la i-ème colonne centrée

Et la matrice de covariance est donnée par la formule suivante :

$$\Gamma = \frac{1}{M}\sum_{i=0}^{M-1}\Gamma_i \tag{3.11}$$

En appliquant la diagonalisation des matrices carrés quaternioniques sur Γ, on a la décomposition suivante :

$$\Gamma = W\Omega W\Delta \tag{3.12}$$

Par construction, Γ est hermitienne quaternionique d'où les valeurs dans la matrice Ω diagonale sont des réelles.

3.3.3 Projection de l'image sur les vecteurs propres

Après avoir diagonalisé la matrice de covariance, chaque colonne de l'original est projeté un après l'autre par :

$$y_i = W^T s_i \tag{3.13}$$

En répétant cette opération pour toutes les colonnes, on obtient une nouvelle image Y dont les colonnes sont les y_i.

3.4 Interprétations et applications

3.4.1 Interprétations géométriques

Il est possible de voir l'ACP sur le corps des quaternions comme la recherche des axes principaux non plus d'un nuage de points mais d'un nuage de vecteurs tridimensionnels.

L'ACP Quaternionique donnera donc comme première composante l'axe de dispersion maximale des vecteurs, et en même temps la direction dominante de tous les vecteurs du nuage. Dans le cas des images couleurs, cette information est liée à la répartition des fréquences spatiales ainsi qu'à l'information de couleur.

L'ACPQ permet donc une extension directe de l'ACP pour les images en niveau de gris au cas des images couleur.

3.4.2 Décomposition de l'image « Eigen-image »

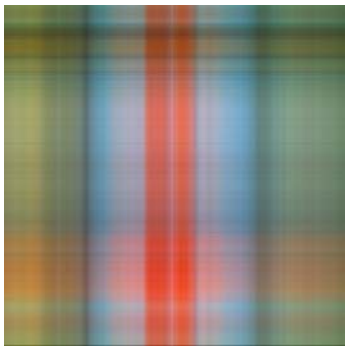
Similaire à la décomposition des images en niveau de gris, le SVD d'une image couleur peut se décomposer en produit des vecteurs [11] :

$$S = U\begin{pmatrix} \sum_r & 0 \\ 0 & 0 \end{pmatrix} V^{\Delta} = \sum_{i=1}^{r} \lambda_i . u_i . v_i^{\Delta} \tag{3.14}$$

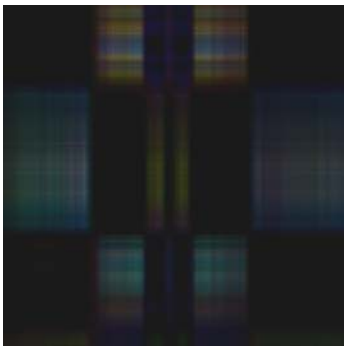
Avec u_i et v_i sont des vecteurs colonnes de U et V respectivement, λ_i est les valeurs dans le diagonal du matrice réelle, et r le rang de la matrice S.

Ainsi, l'image couleur S peut être considérée comme la combinaison linéaire de r-images propres couleur. Cette décomposition nous permet d'avoir une compression d'image sans perte.

(a_1)



(b_1)



(c_1)

(d_1)

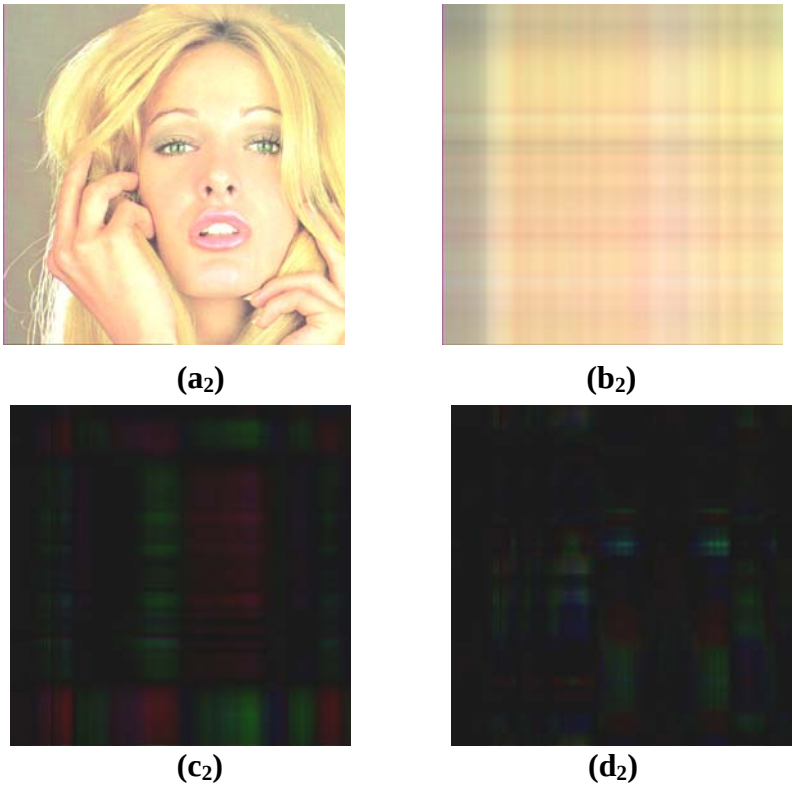


Figure 3.1 « Eigen-images » sélectionnés de Mandrill et Tiffany. (a) : images originales ; (b) : 1-image propre ; (c) : 2-image propre ; (d) : 4-image propre

3.4.3 Compression d'image

3.4.3.1 Définition et objectif

L'objectif de la compression d'image est de minimiser autant que possible la quantité d'information nécessaire à une représentation fidèle de l'image originale.

Ceci est d'autant plus réalisable que la corrélation spatiale des pixels de l'image est importante.

Les méthodes de compression de l'information peuvent être classées en deux catégories, les méthodes de codage pixel et les méthodes de codage global. Ces méthodes peuvent être qualifiées de réversible (sans perte) ou d'irréversible (avec perte). Les techniques réversibles produisent une reproduction exacte de l'image originale, tandis que les techniques irréversibles ne restituent qu'une approximation des données initiales. On notera cependant, que toute méthode irréversible peut devenir réversible après transmission de l'erreur de quantification.

3.4.3.2 Les méthodes de codage pixel

Les méthodes de codage pixel, dites syntaxiques, analysent l'information pixel par pixel. Elles consistent à rechercher la redondance de l'information contenue dans l'image et à la coder de façon réduite, en faisant appel à des concepts de la théorie de l'information. Cette famille de codage comprend :

- Les méthodes de codage statistique : Huffman, Shannon-Fano, arithmétique… Ces méthodes tiennent compte de la fréquence d'apparition d'un symbole dans l'ensemble du message afin de coder, sur un nombre réduit de bits, les symboles les plus probables, et sur des structures binaires plus longues, les symboles les moins fréquents.

- Les méthodes de codage spatial : par plages, par blocs et prédictives. Ces méthodes tiennent uniquement compte de la corrélation spatiale des éléments à coder.

- Les méthodes de codage par transformation orthogonale : Transformation de Walsh-Hadamard (TWH), transformation cosinus discrète (TCD, base de la norme JPEG)… Une transformation orthogonale réalise une rotation de l'espace de représentation. Les données de l'image passent alors d'un espace où elles sont hautement corrélées, dans un espace où cette corrélation est minimisée. L'élimination d'un certain nombre de coefficients et la quantification de ceux conservés permettent alors une réduction du nombre de bits nécessaire pour représenter l'image.

- Les méthodes hiérarchiques : par arborescence, par interpolation et par transformation en S. Ces méthodes décomposent une image de manière itérative en différentes images sous échantillonnées, et sont ainsi adaptées à la transmission progressive des images.

3.4.3.3 Les méthodes de codage global

Les méthodes de codage global, dites sémantiques, extraient l'information contenue dans l'image non plus pixel par pixel mais selon les caractéristiques de l'image. L'image est modélisée sous la forme de deux composantes additives : l'une comporte les brusques ruptures (information contour), l'autre les variations régulières (information texture). Cette catégorie comprend :

- Les méthodes hiérarchiques : décomposition en sous bandes, en sous images directionnelles et pyramide laplacienne.

- Les méthodes de codage par segmentation et séparation en deux composantes : Les premiers travaux utilisant cette méthode remontent à ces dix dernières années seulement. Cette méthode permet d'aboutir à des taux de compression proches de 7:1.

3.4.3.4 La reconstruction de l'image par SVDQ

La représentation approximative d'une image couleur peut se former par la somme des k-premiers termes des « eigen-images » [11] :

$$|S|_k = \sum_{i=1}^{k} \lambda_i . u_i . v_i^{\Delta} \quad (k \leq r) \tag{3.14}$$

Dans le cas où k=r, la reconstruction de l'image est exacte; tandis que si k<r la compression présente des pertes (cf section 3.5.3.1).

L'espace requis de l'image approximative $|S|_k$ diminue de $3n^2$ à (4*2n+1)k.. En effet, on a k valeurs singulières et 2n*k vecteurs de quaternion utilisés et chaque quaternion contient 4 réels.

En se basant sur la formule précédente, prenons par exemple k = 40. L'espace utilisé par l'image réduit de 3*512*512 = 786.10^3 à (4*2*512+1)*40 = $164*10^3$, c'est-à-dire, une réduction en taille de 79,2%.

K	**1**	**2**	**4**	**10**	**20**	**40**	**60**	**80**	**100**
Réduction	99,5%	99,0%	97,9%	94,8%	89,6%	79,2%	68,7%	58,3%	47,9%

Tableau 3.1 Pourcentage de gains en espace selon la valeur de k

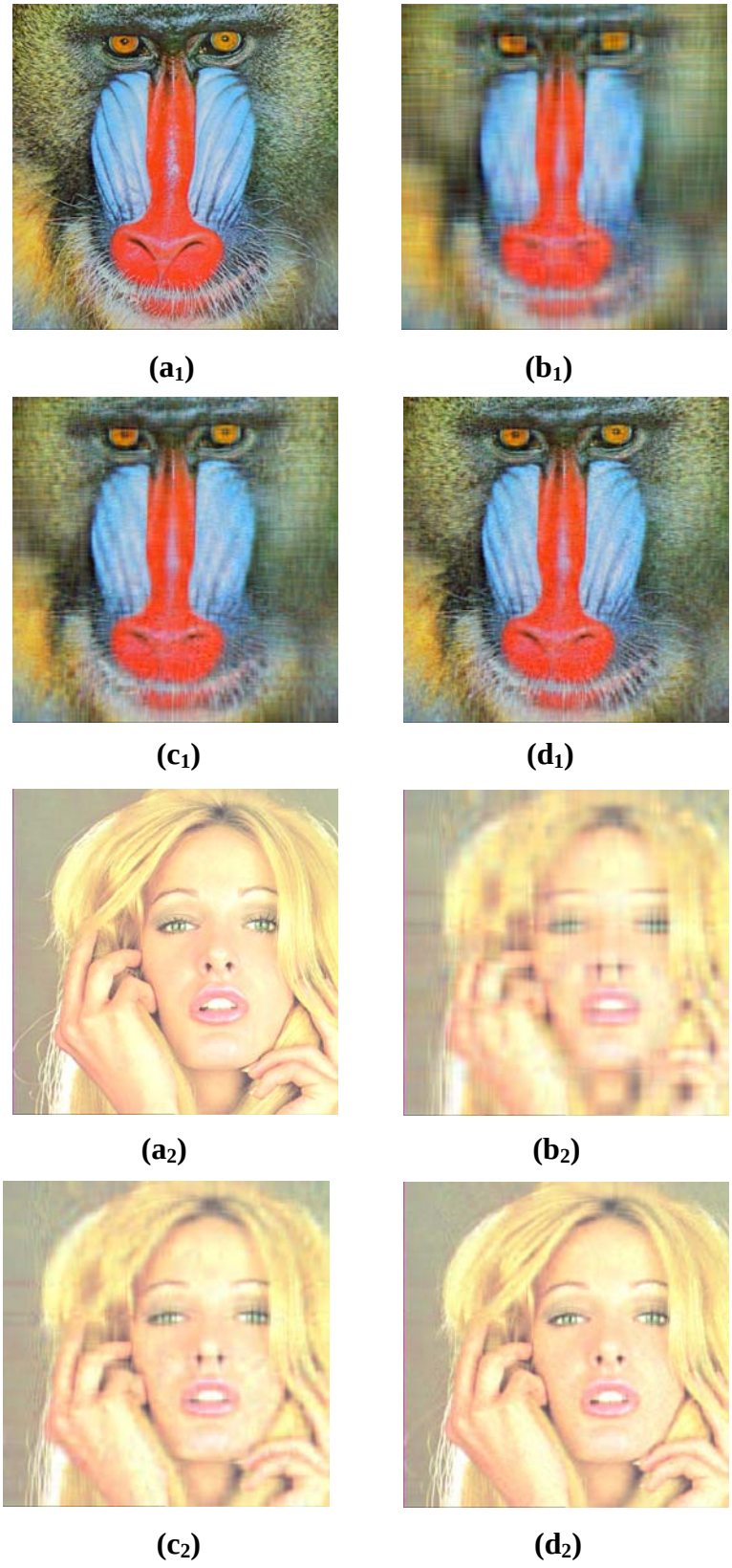


Figure 3.2 Les reconstructions des images de Mandrill et Tiffany par la somme des « eigen-image »: (a) originales, (b) k=10, (c) k=20, (d) k=40

3.4.3.5 L'analyse des images reconstruites

Toutefois, malgré le gain en taille, il est aussi important de savoir les pertes de qualités sur les images. Dans ce cas, nous proposons d'étudier la courbe des valeurs singulières en fonction de son index et de faire le calcul des PSNR des images reconstruites.

La figure nous montre les valeurs singulières de Mandrill et Tiffany. En analysant les deux courbes, Tiffany est plus rapidement reconstruite que Mandrill.

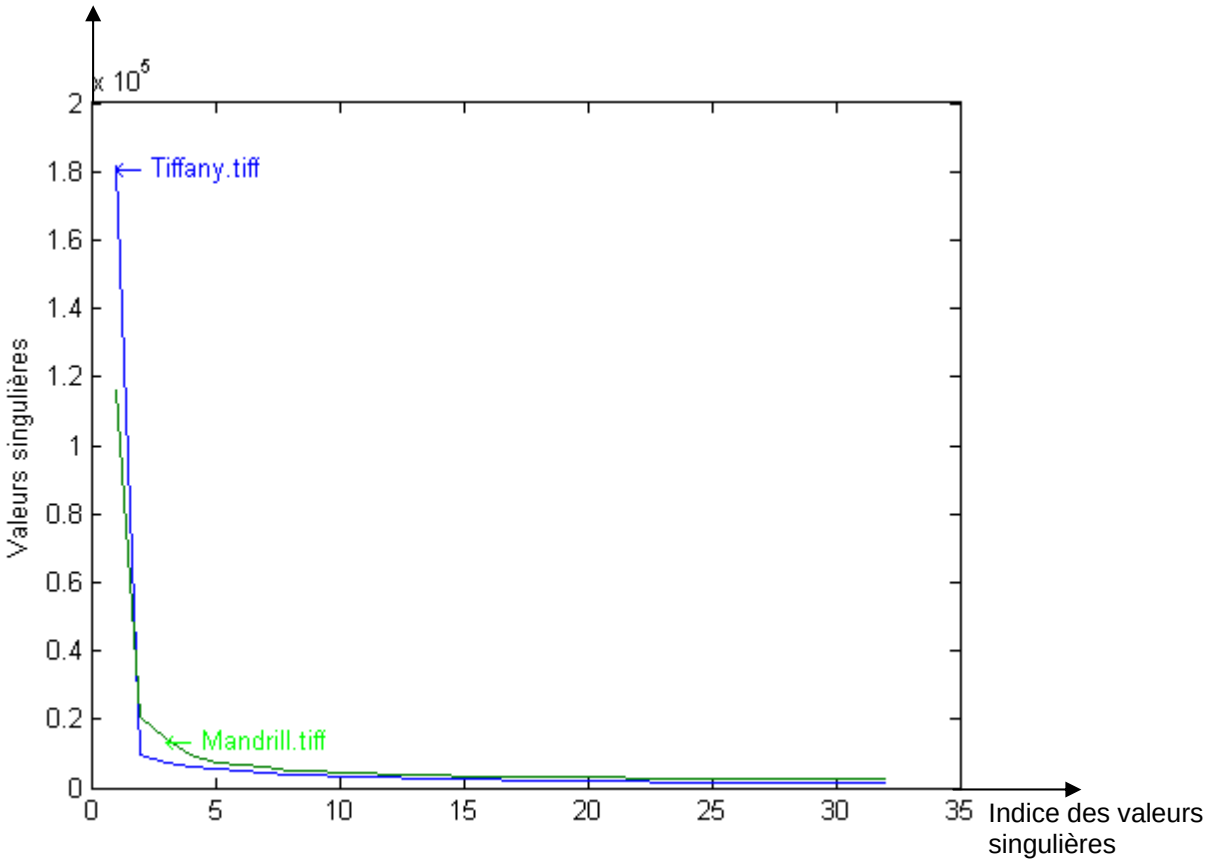


Figure 3.3 Les valeurs singulières de Tiffany et Mandrill

L'étude de PSNR nous aide aussi à qualifier les images reconstruites.

Par définition, PSNR (sigle de Peak Signal to Noise Ratio) est une mesure de distorsion utilisée en image numérique, tout particulièrement en compression d'image. Il s'agit de quantifier la

performance des codeurs en mesurant la qualité de reconstruction de l'image compressée par rapport à l'image originale.

Il est calculé par :

$$PSNR = 10 \cdot \log_{10}\left(\frac{d^2}{EQM}\right) \tag{3.15}$$

où d est la dynamique du signal. Dans le cas standard d'une image où les composantes d'un pixel sont codées sur 8 bits, d = 255.

EQM est l'erreur quadratique moyenne et est définie pour 2 images I_0 et I_k de taille m×n comme:

$$EQM = \frac{1}{3mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{Coul=R,V,B}\left|I_0^{Coul}(i,j) - I_k^{Coul}(i,j)\right|^2 \tag{3.16}$$

Les valeurs typiques de PSNR pour des images de bonne qualité varient entre 30 et 40 dB.

k	**10**	**20**	**30**	**40**	**50**	**60**
Tiffany	27,3	30,7	33,1	35	36,6	38,1
Mandrill	21,9	23,4	24,6	25,7	26,8	27,8

Tableau 3.2 PSNR des images couleur reconstruites, Tiffany et Mandrill, avec plusieurs valeurs de k .

3.4.4 Amélioration d'image « Image enhancement »

A partir de la décomposition en valeur singulières de l'image donnée par la formule (3.14), on obtient :

- les valeurs singulières λ_i
- les vecteurs colonnes u_i et v_i

La pondération de ces valeurs singulières obtenues permet d'améliorer les images reconstruites. Il y a deux types de pondération : la pondération linéaire et la pondération non linéaire [11].

La pondération linéaire est donnée par :

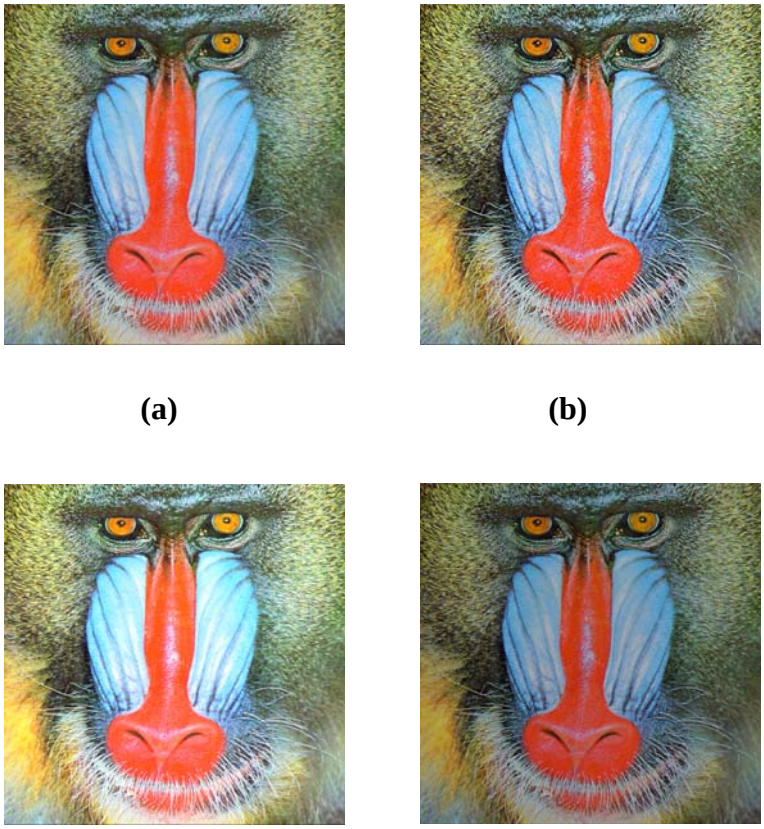
$$[S]_m = \sum_{i=1}^{r} (1 + m.i)\lambda_i . u_i . v_i^{\Delta} \; (m \geq 0) \tag{3.17}$$

Cette pondération fait apparaitre les composants à haute fréquence de l'image, il est équivalent au filtre à passe haut.

Tandis que la pondération non linéaire est représentée par:

$$[S]_{\alpha} = \sum_{i=1}^{r} \lambda_i^{\alpha} . u_i . v_i^{\Delta} \tag{3.18}$$

Si $\alpha > 1$, la pondération non linéaire est un filtre passe-haut. Par contre, si i $\alpha < 1$, elle devient un filtre passe-bas.



(a) **(b)**

(c) **(d)**

Figure 3.4 (a) Image originale

(b) Image obtenue à partir de la pondération linéaire m = 0.005

(c) Image obtenue à partir de la pondération non linéaire α =1.02

(d) Image obtenue à partir de la pondération non linéaire α = 0.98

CONCLUSION

Le travail réalisé montre que la modélisation quaternionique des images couleur est un outil adapté au traitement d'image. Puisque qu'il est possible de représenter trois composantes dans un quaternion, cela nous donne la possibilité de généraliser toutes les techniques standards appliquées aux images au niveau de gris. Basées sur l'algèbre quaternionique, cette méthodologie utilise surtout les opérations ponctuelles et matricielles.

En effet, nous avons utilisé trois modélisations :

- Filtrage quaternionique (appliqué surtout dans la détection de contour) ;
- Analyse en composantes principales quaternioniques (utilisée dans la compression d'images) ;
- Décomposition en valeurs singulières quaternioniques (dans l'amélioration de la qualité des images).

Dans le traitement d'image il s'avère indispensable de prendre en compte les deux paramètres : qualité de produit, performance de traitement. Cependant, vue la complexité de la modélisation quaternionique, surtout les opérations matricielles, il serait plus méthodique de suivre les concepts suivants :

- Choix du meilleur algorithme (complexité de traitement) ;
- Choix du référentiel (pour mieux se situer dans les opérations matricielles).

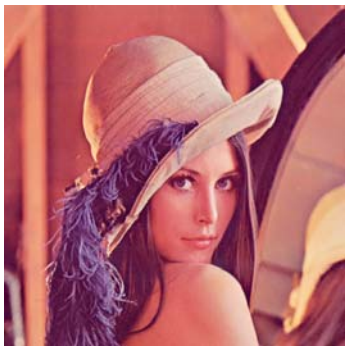
Ce travail donne des bases à de futurs travaux de recherche sur les traitements des images numériques couleur afin de faire évoluer les concepts comme :

- la transformée en ondelettes quaternionique appliquée à la compressions d'image ;
- la transformée de Fourrier Quaternionique ;
- le tatouage des images numériques couleur ;
- la segmentation des images numériques couleur.

ANNEXE

ANNEXE 1 LES IMAGES UTILISEES

(a)



(b)

(c)

(d)

Figure a.1 Les images utilisées :

(a) Mandrill Baboon 512x512

(b) Lena 512x512

(c) House 256x256

(d) Tiffany 512x512

ANNEXE 2 TRANSFORMEE DE FOURIER (TF) D'UNE IMAGE

Soit f(x,y) une fonction à deux variables représentant l'intensité d'une image au point d'abscisse x et d'ordonnée y. La transformée de Fourier de cette image permet de passer d'une représentation spatiale à la représentation de l'image dans le domaine fréquentiel. Elle est donnée par :

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\, e^{-j2\pi(ux+vy)}\, dx\, dy \qquad \text{(A2.1)}$$

Où F(u,v) est la transformée de fourrier de la matrice *f(m,n)*. Les deux variables u et v représentent les fréquences spatiales de l'image selon les directions *Ox* et *Oy* respectivement.

Or, l'image ayant un nombre de pixels fini, l'intensité lumineuse des pixels est donc un signal à support borné. D'où l'utilisation de la TF discrète donnée par :

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi(ux/M+vy/N)} \qquad \text{(A2.2)}$$

Nous voyons d'après l'expression de la TF que *F(u,v)* est en général un nombre complexe, même si *f(m,n)* est un nombre réel. *F(u,v)* possède donc une amplitude et une phase.

La transformée inverse se calcule par :

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v)\, e^{j2\pi(ux/M+vy/N)} \qquad \text{(A2.3)}$$

Dans une image, on s'intéresse à la fréquence spatiale, qui est en fait le taux auqel l'intensité varie à travers l'image (taux de changements de tons de gris dans l'image).

ANNEXE 3 LE FILTRAGE NUMERIQUE ET LE PRODUIT DE CONVOLUTION

La convolution est l'opération de traitement de signale la plus fondamentale. Elle indique que la valeur du signal de sortie à l'instant t est obtenue par la sommation (intégrale) pondérée des valeurs passées du signal d'excitation x(t). La fonction de pondération est précisément la réponse impulsionnelle f(t). L'équation générale de la convolution est une somme de réponse impulsionnelle pour les réponses. La convolution est commutative.

$$y(t) = x * f(t) = \int_{-\infty}^{\infty} x(t-\tau)\ f(\tau)\, d\tau = f * x(t) \int_{-\infty}^{\infty} x(\tau)\ f(t-\tau) d\tau \qquad \text{(A3.1)}$$

Le produit de convolution 1D d'un signal x(n) avec un filtre h(n) est donné par :

$$(x * h)(n) = \sum_{k=-\infty}^{+\infty} x(k)h(n-k) \qquad \text{(A3.2)}$$

Cette opération s'appelle aussi un filtrage linéaire spatial.

Dans le cas 2D, c'est la notion de filtrage d'image qui est un signal 2D (cf. 1.2).

ANNEXE 4 QUATERNION TOOLBOX FOR MATLAB®

Matlab® is a proprietary software system for calculating with matrices of real and complex numbers, developed and sold by The MathWorks.

Quaternions are hypercomplex numbers (that is generalizations of the complex numbers to higher dimensions than two).

Quaternion toolbox for Matlab® extends Matlab® to allow calculation with matrices of quaternions in almost the same way that one calculates with matrices of complex numbers. This is achieved by defining a private type to represent quaternion matrices and overloadings of many standard Matlab® functions. The toolbox supports real and complex quaternions (that is quaternions with four real or complex components).

License

The toolbox is licensed under the GNU General Public License.

Prerequisites

The toolbox requires MATLAB® Version 7.4 (R2007a) or later.

http://qtfm.sourceforge.net/

ANNEXE 5 CODES EN MATLAB®

A5.1 Détection de contours par la méthode de Sangwine

```
%Lire l'image vers un matrice des quaternions
A = imreadq('house.tiff') ;
% Taille de l'image ;
n = size(A,1) ;

u = quaternion(1,1,1)/sqrt(3) ; % axe des niveaux gris
cu=conj(u);

%Horizontal
HorizBas   = [A(n, :) ; A(1:n-1, :)] ;
HorizHaut  = [A(2:n, :) ; A(1, :)] ;

%Vertical
VertDroit  = [A(: ,n) A(: ,1:n-1)];
VertGauche = [A(:, 2:n)  A(:, 1)];

Horiz   = cast(HorizBas, 'double') +  cast(A, 'double') + cast(HorizHaut, 'double');
HorizU  = u * Horiz * cu;

Vert    = cast(VertDroit, 'double') +  cast(A, 'double') + cast(VertGauche, 'double');
VertU   = u * Vert * cu;

HorizFinal  = [Vert(n, :); Vert(1:n-1, :)] +  [VertU(2:n, :); VertU(1, :)];
```

```
HorizFinal  = HorizFinal/3.0;

VertFinal   = [Horiz(:, n) Horiz(: ,1:n-1)] +  [HorizU(:, 2:n) HorizU(: , 1)];
VertFinal   = VertFinal/3.0;

HorizRej    = HorizFinal + (u*HorizFinal*u);
VertRej     = VertFinal + (u*VertFinal*u);

HorizDist = abs (HorizRej);
distH = cast (HorizDist, 'uint16');

VertDist = abs (VertRej);
distV = cast (VertDist, 'uint16');

maxi = ((distH + distV) + abs(distH - distV))/2;

mini = cast(255*ones(n,n),'uint16')  - maxi ;

contourA = cast(quaternion (mini, mini, mini), 'uint8') ;

imwrite (contourA, 'houseContourSangwine.tiff');
image(contourA);
```

A.5.2 Décomposition en valeurs singulières

```
% Lire l'image vers un matrice des réels (ligne * colonne * 3)
I = imread('Mandrill.tiff') ;
%  Taille de l'image ;
n = size(A,1) ;

%  Matrice complexe équivalent à la matrice quaternionique de l'image
% ROUGE
Ri1= complex(zeros(n, n, 'double'), cast(I(:,:,1),'double')) ;
Ri2 = conj (Ri1);
%  VERT et BLEU
 VBi1 = complex(cast(I(:,:,2),'double'), cast(I(:,:,3),'double')) ;
 VBi2 = -conj(VBi1);
%  Matrice complexe equivalence
Iqc = [Ri1 VBi1 ; VBi2 Ri2] ;

% SVD Complexe
[U,S,V] = svd(Iqc);

U1 = U(1:n, :) ; U2 = U(n+1:2*n, :) ;
V1 = V(1:n, :) ; V2 = V(n+1:2*n, :) ;

Uq = quaternion (real(U1), imag(U1), -real(U2), imag(U2));
Vq = quaternion (real(V1), imag(V1), -real(V2), imag(V2));
```

A.5.3 Reconstruction des images

```
% Eigen image 1
Q = S(1,1) * Uq(:, 1)* (Vq(:, 1))'  ;
% Partial sum 10
for k = 1:9
    Q = Q + S( 1 + 2*k , 1 + 2*k ) * Uq(:, 1 + 2*k )* Vq(:, 1 + 2*k)';
end
J = cast(v(Q),'uint8');
imwrite (J, 'Mandrill_s10.tiff');

% Partial sum 20
for k = 10:19
    Q = Q + S( 1 + 2*k , 1 + 2*k ) * Uq(:, 1 + 2*k )* Vq(:, 1 + 2*k)';
end
J = cast(v(Q),'uint8');
imwrite (J, 'Mandrill_s20.tiff');

% Partial sum 40
for k = 20:39
    Q = Q + S( 1 + 2*k , 1 + 2*k ) * Uq(:, 1 + 2*k )* Vq(:, 1 + 2*k)';
end
J = cast(v(Q),'uint8');
imwrite (J, 'Mandrill_s40.tiff');
image(J);
```

A.5.4 Amélioration des images

```
% Eigen enhancement linear weighting
m = 0.005;
Q = (1+m)*S(1,1) * Uq(:, 1)* (Vq(:, 1))'  ;
% Filtrage linéaire
for k = 1:511
    Q = Q + (1+m*(k+1))*S( 1 + 2*k , 1 + 2*k ) * Uq(:, 1 + 2*k )* Vq(:, 1 + 2*k)';
end
J = cast(v(Q),'uint8');
imwrite (J, 'Mandrill_lin_weig.tiff');

% Eigen enhancement nonlinear weighting
alpha = 0.98 ; % alpha = 1.02

Q = (S(1,1)^alpha) * Uq(:, 1)* (Vq(:, 1))'  ;
for k = 1:511
    Q = Q + (S( 1 + 2*k , 1 + 2*k )^alpha) * Uq(:, 1 + 2*k )* Vq(:, 1 + 2*k)';
end
J = cast(v(Q),'uint8');
imwrite (J, 'Mandrill_nonlin_0_98.tiff');
image(J);
```

BIBLIOGRAPHIE

[01] Alleysson D., Sûsstrunk S. « *Saptio-chromatic PCA of a mosaices color image* », Université Pierre-Medes et EPFL

[02] Bloch I., Gousseau Y., Maître H., Matignon D., Pesquet-Popescu B. , Schmitt F., Sigelle M., Tupin F. « *Le traitement des images – Tome 1* » Cours ANIM Département TSI - Télécom-Paris version 5.0, Décembre 2005

[03] Bloch I., Gousseau Y., Maître H., Matignon D., Pesquet-Popescu B. , Schmitt F., Sigelle M., Tupin F. « *Le traitement des images – Tome 2* » Cours ANIM Département TSI - Télécom-Paris version 5.0, Décembre 2005

[04] Canonne C. «*T.I.P.E. : Nos amis les Quaternions* », Juin 2007

[05] Hamilton W.R. . « *On quaternions* ». Proceeding of the Royal Irish Academy, 1843.

[06] Kyrchei I. « *Determinantal representations of the Moore-Pensore inverse over the quaternion skew field and corresponding Cramer's rules* », May 2010

[07] Le Bihan N. « *Traitement algébrique des signaux vectoriels, application en séparation d'ondes sismiques* ». Thèse de doctorat, INPG, 2001.

[08] Le Bihan N. and Sangwine S. J., « *Quaternion singular value decomposition based on bidiagonalization to a real or complex matrix using quaternion Householder transformations* », Applied Mathematics and Computation, 12pp., DOI :10.1016/j.amc.2006.04.032, in press, available online June 2006.

[09] Le Bihan N. and Sangwine, S. J. « *Computing the SVD of a quaternion matrix* », University of Essex, INPG Grenoble

[10] Maître H. « *La détection de contours dans les images* », Cours, Télécom ParisTech

[11] Pei S. C., Chang J. H., and Ding J. J., « *Quaternion matrix singular value decomposition and its applications for color image processing* » in Proc. Int. Conf. Image Process., Sep. 2003, vol. 1, pp. 805–808.

[12] Petrenand M., Louys M., Collet Ch., Flitti F. « *Représentation couleur d'images multispectrales astronomiques* » Université Strasbourg 1, Juin2004

[13] Sangwine S. J. and Le Bihan N. . « *Quaternion Toolbox for Matlab* ® ». URL http://qtfm.sourceforge.net/ .Software library, licensed under the GNU General Public License, 2005.

[14] Sangwine S. J. and Le Bihan N.. « *Quaternion principal component analysis of color images* », In IEEE International Conference on Image Processing (ICIP), volume 1, pages 809–812, 2003.

[15] Sangwine, S. J. and Le Bihan, N. «*Analyse de signaux vectoriels base sur modèle quaternionique* », University of Essex, INPG Grenoble

[16] Sangwine S.J : « Colour image edge detector based on quaternion convolution» Electronics Letters, 34, 10:969–971, May 1998.

[17] Shi L. « *Exploration in quaternion colour* », Thesis Master of Science, Simon Fraser University, Summer 2005

RENSEIGNEMENTS SUR L'AUTEUR

Nom : RAMINOSON
Prénoms : Tsiriniaina
Tél. : +261 34 00 162 84
E-mail : raminoson@gmail.com

Titre du mémoire :

Traitement des images numériques couleur par les quaternions

Nombre de pages : 62
Nombre de tableaux : 02
Nombre de figures : 19

Mots clés : Quaternions, images numériques couleur, Détection de contour, Matrices de quaternions, Analyse en Composantes Principales Quaternionique, Décomposition en Valeurs Singulières quaternionique.

Directeur de mémoire : Monsieur RAZAFINDRADINA Henri Bruno
Tél. : +261 32 174 56

RESUME

Le modèle quaternionique est un outil mathématique qui pourrait être utilisé dans le traitement des images numériques couleur. En effet, en modélisant les images numériques couleur comme des images dont chaque pixel est un quaternion, il est possible de développer d'autres démarches de traitement de couleur. Au préalable, il faut redéfinir les outils de traitement d'image connus pour les modélisations en niveaux de gris (sur $\mathbb{R}$ ou $\mathbb{C}$). Ces outils seront par la suite généralisés à l'aide du corps des quaternions ($\mathbb{H}$). Ceci convertit la modélisation en niveau de gris en niveau de couleurs. Dans ce travail, nous voudrions présenter les aspects théoriques de la modélisation quaternionique et leurs applications sur des traitements des images numériques couleur. Ces aspects s'articulent autour des principes théoriques suivants : le filtrage quaternionique appliqué à la détection des contours, l'Analyse en Composantes Principales Quaternionique (ACPQ) et la Décomposition en Valeurs Singulières quaternionique (SVDQ).

ABSTRACT

Quaternionic model is a mathematical tool that can be used in colour image processing. In fact, by modelling colour image into quaternion defined pixel, it is possible to develop other colour image processing method. First, grey scale models (in $\mathbb{R}$ or $\mathbb{C}$) should be defined. After, they should be generalized with quaternion($\mathbb{H}$). That will convert grey into colour scale, a very wider scale for image processing. This article is presenting theorical aspects of quaternion model and its use in image processing with the following steps: quaternionic filter applied in edge detector, quaternionic principal components analysis (QPCA) and quaternionic singular value decomposition (QSVD).