République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique
Département d'ELECTRONIQUE
Année scolaire : 2011 - 2012

Projet de fin d'études
en vue de l'obtention du diplôme
d'ingénieur d'état en ELECTRONIQUE

Thème

Modèles de représentation de l'information couleur à une dimension

Proposé et dirigé par :

Pr. L. HAMAMI
Dr. D. AOUADA

Réalisé par :

M. ZEBOUDJ Abdennour

Promotion : Juin 2012
Ecole nationale polytechnique, 10, Avenue Hassen Badi, BP 182 El-Harrach, Alger, Algérie

Dédicace

A mes parents
A mes frères et soeur
A tous mes amis
Je dédie ce modeste travail

Remerciements

J'adresses mes plus vifs remerciements à ma promotrice Pr. L. HAMAMI ainsi qu'à ma co-promotrice Dr. D. AOUADA pour leurs précieux conseils, pour les connaissances qu'elles m'ont octroyées et surtout pour leur assistance malgré leurs nombreuses occupations.

Je tiens à remercie :

Monsieur BOUSBIA , Maitre de conférence à l'école nationale polytechnique, qui nous a fait l'honneur de présider le jury.

Monsieur BOUSSEKSSOU pour avoir accepté d'examiner ce modeste travail.

Mes parents qui me soutiennent et m'aident toujours.

Je tiens à remercier également l'ensemble des enseignants qui ont contribué à ma formation.

Je voudrais enfin remercier tous mes camarades ainsi que tous ceux qui, de près ou de loin, ont fait que l'aboutissement de ce travail soit possible.

ملخص:
أغلب نماذج تمثيل الألوان تستعمل ثلاث أبعاد لذلك التمثيل، مثل النموذج RVB.
يكمن هذا العمل في اقتراح نموذج من نوع آخر مشتق من نموذج تم اقتراحه مؤخرا، من أجل تمثيل معلومة الألوان في بعد واحد فقط. النموذج المقترح يعطي نتائج أفضل بكثير من النموذج الأصلي الذي لم يعط النتائج المرغوب فيها.
التصديق على النتائج تم بواسطة البرنامج MATLAB . قاعدة البيانات المستخدمة تتكون من 110250 صورة ملونة.
كلمات مفتاحية: نموذج ألوان، RVB،HCL ، نموذج CA ،قياس التشابه، حلزونية spring curve، حلزونية أرخميدس.

Résumé :

La majorité des espaces couleurs existant représentent l'information couleur dans un espace 3D, comme dans le système RVB. Un modèle couleur 1D a été récemment proposé pour stocker toute l'information couleur de l'espace HCL dans seulement une dimension, mais ce modèle ne préserve pas totalement l'information couleur originale. Ce travail consiste à développer d'autres modèles 1D dérivés du modèle 1D original, et qui préservent efficacement les propriétés perceptuelles de l'espace HCL. L'implémentation de ces modèles 1D ainsi que la validation des résultats ont été faites par le logiciel Matlab. La base de données utilisée contient 110250 images couleurs prises sous différentes conditions.

Mots-clés : Modèle couleur, HCL, RVB, Modèle CA, Mesure de similarité, Spirale spring curve, Spirale d'Archimède.

Abstract :

The majority of existing colour spaces represent colour information in a 3D space, as RGB system doses. A 1D colour model has been recently proposed to stock all HCL colour information in only 1D, but this model doses not preserve totally original colour information. This work consists of developing others 1D models derived from the original 1D colour model, and which preserve efficiently perceptual properties of HCL colour space. Both Implementation and validation of the results were made by the Matlab software. The database used is made up of 110250 colour images taken in different conditions.

Keywords : Colour Model, HCL, RGB, CA Model, Similarity measure, Spring curve spiral, Archimedean spiral.

Table des matières

Table des figures

Liste des Abréviations :

RVB	:	Rouge, Vert, Bleu
HCL	:	Hue, Chroma, Luminance
CA	:	Cumulatif Angle
DCT	:	Discret Cosine Transform
RLE	:	Run Length Encoding
LZW	:	Lempel-Ziv-Welch
JPEG	:	Joint Photographic Experts Group
SSIM	:	Structural SIMilarity
CIE	:	Commission Internationale de l'Eclairage
S-CIELAB	:	Spatial CIELAB
CAs	:	Cumulatif Angle spring curve
CAL	:	Cumulatif Angle Luminance
ALOI	:	Amsterdam Library of Object Images

Introduction générale

Les images numériques couleurs occupent une grande place dans notre vie quotidienne, les appareils photos et les applications internet traitent des images couleurs d'une grande qualité. La majorité des systèmes (surtout les systèmes temps réel) matériels associés à ces applications sont limités par le temps de traitement et par la capacité de stockage des données, c'est pourquoi plusieurs algorithmes de traitements d'images utilisent les images en niveaux de gris comme étant des données d'entrée, au lieu d'utiliser les images couleurs contenant toute l'information couleur originale, ce qui donnerait des résultats plus précis. Une image en niveaux de gris est définie comme étant une combinaison du rouge, vert et bleu représentés dans l'espace RVB. Or, il résulte de cette combinaison une représentation non unique des couleurs naturelles, ce qui peut induire à des problèmes sérieux comme la perte de certaines propriétés.
La plupart des modèles couleurs existant (comme RVB) représentent l'information couleur en utilisant trois composantes (3 Dimensions). Chaque espace est défini suivant des critères particuliers souvent liés aux propriétés perceptuelles de l'oeil. Parmi les espaces perceptuels, il y a l'espace couleur HCL, qui utilise trois composantes (H (Hue), C (Chroma), L (Luminance)) pour représenter l'information couleur selon les propriétés visuelles de l'oeil. Ceci est très important dans les applications de reconnaissances de formes.
Pour rendre plus léger la représentation 3D de l'espace HCL, un modèle couleur à une dimension nommé le modèle CA (Cumulatif Angle) a été récemment défini, son objectif est de concentrer toute l'information couleur contenue dans l'espace HCL en seulement une dimension, sans perdre ses propriétés perceptuelles. Un seul paramètre noté ζ représente cette unique dimension. Les auteurs de ce modèle 1D avaient testé ses performances, et ce en calculant l'erreur RMSE entre 100 images de référence et leurs versions reconstruites à partir du modèle CA [1]. Les résultats montrent que le modèle CA proposé présente certains inconvénients vis-à-vis du rendu des images reconstruites. En effet, l'information couleur originale s'avère perdue dans certains cas de figures, suivant la façon dont on quantifie les composantes C et L. Par exemple, dans le cas où C et L sont représentées sur 8 bits, la couleur dans l'image reconstruite est délavée. Ces artéfacts posent un problème majeur si l'on veut utiliser le paramètre ζ comme étant une entrée pour des algorithmes de traitement, car ce paramètre ne préserve pas totalement l'information couleur de l'espace HCL.
Le domaine dans lequel se situe ce travail est celui de la modélisation des espaces couleurs. On proposera d'autres modèles 1D dérivés du modèle CA original. Ces modèles dérivés ont pour objectif d'améliorer les performances du modèle CA original. La vérification et la validation des résultats théoriques vis-à-vis des performances de ces modèles ont été

faites sur le logiciel de simulation MATLAB.

Ce travail sera reparti en cinq chapitres :

- Dans le chapitre 1, on présentera quelques techniques de compression des images numériques, ce qui constitue une introduction à une problématique non traitée encore, qui concerne la compression du paramètre ζ de l'espace couleur 1D. Ce chapitre permet aussi de différrencie entre le principe de réduction de dimensionnalité et la compression proprement dite.

- À travers le chapitre 2, on introduira les notions d'espace couleur et de calcul de distance dans un espace couleur. On présentera quelques espaces couleurs dont le HCL, et on abordera des mesures de similarité dont la SSIM et la S-CIELAB qui seront utilisées pour l'évaluation et la comparaison entre le modèle CA original et ses versions dérivées proposées.

- Le chapitre 3 introduit le modèle couleur original à une dimension (CA). On présentera d'abord très brièvement un état de l'art sur des modèles 1D existants. Ensuite, on détaillera les étapes de construction du modèle CA original, et on donnera les relations de passage de ce modèle vers l'espace HCL.

- Le chapitre 4 présentera les versions dérivées du modèle CA original proposées dans notre PFE. On expliquera le principe de chaque modèle dérivé, et on donnera les formules de passage à partir de chaque modèle vers l'espace HCL.

- Dans le chapitre 5 on présentera d'abord la base de données utilisée. Le reste du chapitre sera consacré au résultats et expériences réalisées sur MATLAB. On expliquera alors chaque expérience et on interprétera les résultats correspondants, ce qui aboutira à une comparaison subjective et objective entre les différents modèles proposés ainsi que le modèle CA original.

Nous terminerons ce travail par une conclusion générale qui passera en revue tout ce qui a été abordé dans ce mémoire, et on abordera les perspectives possibles pour la suite de ce travail.

Chapitre 1

Quelques techniques de compression des images numériques

1.1 Introduction

La compression des données est une activité ancienne : l'utilisation d'abréviation en est une preuve. Les langues elles-mêmes utilisent des mots de longueurs variées, les plus fréquents étant les plus courts, afin de réduire la taille des phrases. Cependant, la compression des images numériques est une application plus visible que les autres. C'est elle qui a permis la diffusion des images sur Internet ou encore la démocratisation des appareils photos numériques. Elle constitue également la base de la compression vidéo. L'objectif de ce chapitre est de présenter certaines techniques de compressions d'images les plus utilisées dont la DCT. [2]

1.2 Notions générales

1.2.1 Image numérique

Une image peut être définie comme étant une fonction de deux dimensions f(x,y), où x et y sont des coordonnées spatiales, et l'amplitude de f pour chaque paire de coordonnées (x,y) représente l'intensité ou encore le niveau de gris de l'image en ce point (figure 1.1) [3]. Une image numérique est composée d'un nombre fini d'éléments, chacun d'entre eux a une valeur et location particulières. Ces éléments sont appelés éléments d'image ou pixels. Le terme pixel est le plus petit élément d'une image numérique.

1.2.2 Image en niveaux de gris

Dans une image en niveaux de gris, la couleur d'un pixel peut prendre des valeurs allant du noir au blanc, en passant par un nombre fini de niveaux intermédiaires (figure 1.2). En général, les images en niveaux de gris renferment 256 teintes de gris (chaque pixel

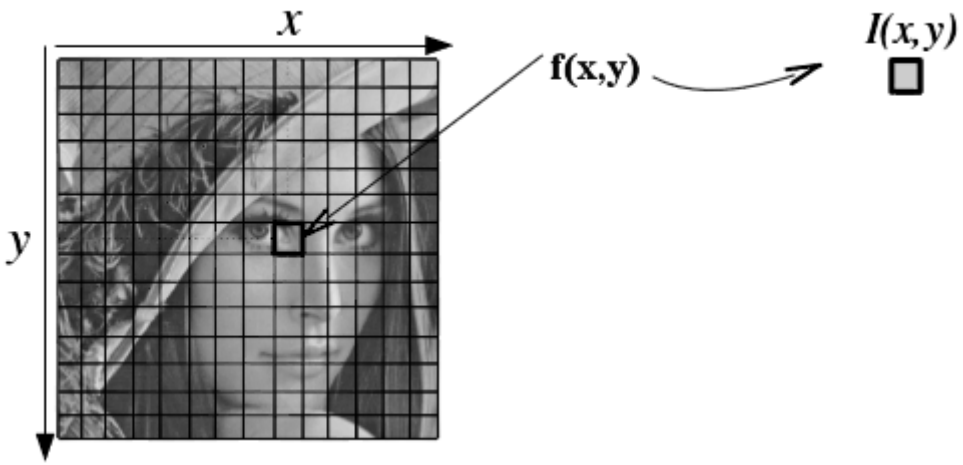


Fig. 1.1 – *Image numérique. [4]*

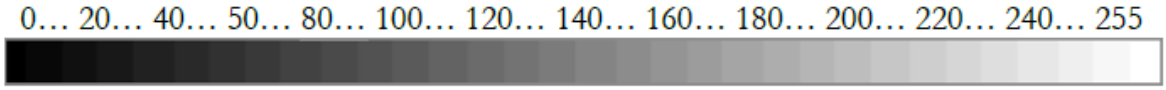


Fig. 1.2 – *Valeurs des niveaux de gris et teintes correspondantes*

étant représenté sur un octet). Par convention la valeur zéro représente le noir (intensité lumineuse nulle) et la valeur 255 le blanc (intensité lumineuse maximale). Chaque pixel est représenté par un octet. Le nombre de niveaux de gris dépend du nombre de bits utilisés pour décrire la couleur de chaque pixel de l'image. Plus ce nombre est important, plus les niveaux possibles sont nombreux. [5]

1.2.3 Image en couleurs

Une image en couleurs est censée représenter le mieux possible la réalité. La représentation des couleurs s'effectue de la même manière que les images en niveaux de gris mais avec quelques particularités. En effet, il faut tout d'abord choisir un modèle de représentation, on peut représenter les couleurs à l'aide de leurs composantes primaires. Les systèmes émettant de la lumière (écrans d'ordinateurs,...) sont basés sur le principe de la synthèse additive (voir chapitre 2) : les couleurs sont composées d'un mélange de rouge, vert et bleu (modèle R.V.B.). [6]

1.2.4 Échantillonnage et quantification [7]

Echantillonnage

L'échantillonnage est une étape fondamentale qui doit tenir compte du contenu informationnel pertinent de l'image à analyser. Lors de la numérisation d'une image, l'image est découpée en une matrice de pixels (on dit aussi échantillonnage spatial). Pixel est l'abréviation de picture element, c'est un élément du quadrillage de l'image numérique.

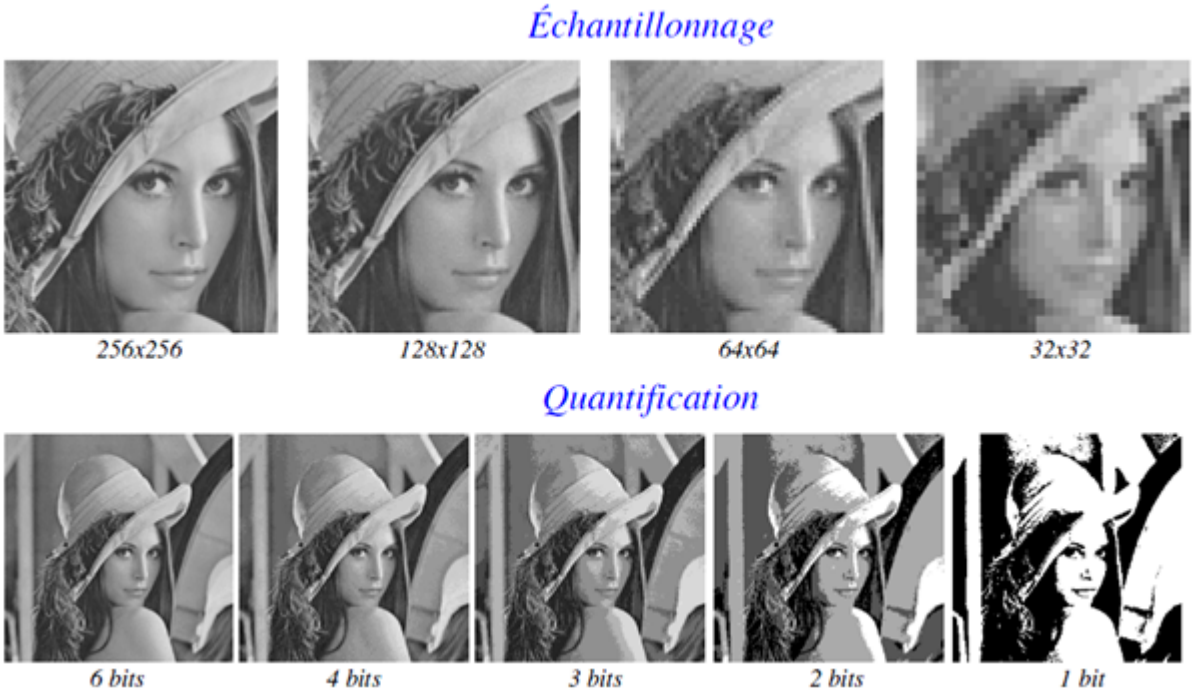


Fig. 1.3 – *Effets de quantification et d'échantillonnage sur une image numérique.*

Le pixel représente le plus petit point que l'on peut distinguer, et c'est la juxtaposition des différents pixels qui produit une image.

Quantification
C'est une technique pour coder l'information couleur sur un certain nombre de bits, il existe des règles pour déterminer la bonne quantification (le bon nombre de bits) pour coder les images numériques :
-L'une dépend du capteur, et de sa capacité effective à observer des signaux de valeurs différentes.

-L'autre concerne le nombre de bits réellement nécessaires pour coder une image, il varie d'une image à l'autre, en fonction de leur contenu informationnel. Ce nombre dépend de l'entropie [8].

La figure (1.3) montre l'effet de l'échantillonnage et de la quantification sur une image.

1.2.5 Résolution

Le nombre total de pixels d'une image numérique définit la résolution de l'image. Elle est exprimée en nombre de pixels par unité de longueur (souvent en ppp : pixel par pouce en français ou dpi en anglais). Cela aura une influence sur la qualité du zoom si on étire l'image.

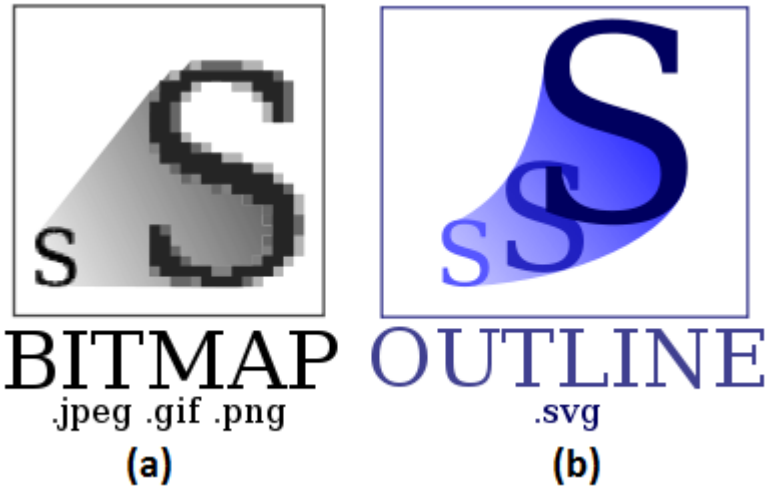


Fig. 1.4 – *(a) Image matricielle. (b) Image vectorielle.*

1.3 Images matricielles et vectorielles

1.3.1 Images matricielles

Une image matricielle (bitmap) est un ensemble de points (pixels), caractérisée par sa discrétisation et sa quantification (figure 1.4(a)).

Il existe différents formats de fichiers d'images matricielles :

- Formats limités à 256 couleurs (GIF, PCX, PGM, ...),
- Formats acceptant différentes quantifications (BMP, TIFF, TGA, PNG, ...),
- Formats limités à 16 millions de couleurs (JPEG, JPEG 2000, ...).

Avantages des images bitmap

- Les images bitmaps peuvent facilement être créées et stockées dans un tableau de pixels représentant l'image,
- La lecture/écriture d'un pixel est aisée,
- Les images bitmap peuvent facilement être affichées sur un écran ou être imprimées.

Inconvénients des images bitmap

- Les fichiers peuvent être très gros (nécessité de compression),
- Problème de changement d'échelle (apparition d'effets de marches, d'escalier ou de flou avec interpolation),
- Les dimensions de l'image doivent être prévues pour la résolution de l'interface de sortie (écran, imprimante).

1.3.2 Images vectorielles

Elles ne stockent pas le résultat du dessin sous la forme de pixels (bitmap), mais elles stockent la façon de dessiner par un ensemble d'objets géométriques (lignes, cercles, polygones, courbes de Bézier, texte, ...) définis par différents attributs (coordonnées, couleur, épaisseur de trait, remplissage, ...) [7]

La figure (1.4) montre l'exemple d'une image vectorielle et sa version matricielle.

Avantages des images vectorielles
- Elles sont adaptées au stockage d'images composées de formes géométriques,
- Elles peuvent aisément être redimensionnées ((figure 1.4(b)),
- Elles prennent moins de place qu'une image bitmap.

Inconvénients des images vectorielles
- Elles peuvent difficilement stocker des images complexes comme des photographies,
- L'affichage d'une image vectorielle peut prendre plus de temps que l'affichage d'une image bitmap de complexité égale.

Vectorisation
Il est facile de convertir une image vectorielle en image bitmap (il suffit d'en faire le rendu). La transformation inverse, appelée vectorisation, est moins triviale. L'algorithme doit analyser l'image bitmap pour en déduire un ensemble de zones qu'il décrira avec des primitives géométriques. L'opération s'accompagne d'une perte de détails et de nuances de couleurs.

1.4 Compression

Compresser une image revient à réduire la taille de son fichier en remplaçant la série de ses données par une autre plus réduite. Le processus inverse est appelé décompression (par exemple pour afficher l'image). La compression concerne essentiellement les images bitmap.
Pourquoi compresser une image ? :
- Pour réduire sa place sur le disque.
- Pour accélérer sa transmission sur un réseau.

On définit le taux de compression par : taux de compression=$\frac{\text{taille originale}}{\text{taille compressée}}$.

Exemple de taux de compression :
Soit une image originale de 100 Ko, compressée à 20 Ko, le taux de compression est 100/20=5. Le taux de compression est variable, il dépend de la nature de l'image à compresser et de l'algorithme de compression utilisé.
Deux types de compression existent :
- Réversible, sans pertes (lossless),
- Non réversible, avec pertes (lossy).

1.5 Compression sans pertes

Ce type de compression ne modifie pas l'image, mais change uniquement la façon dont elle est codée sur le disque. L'image décompressée est identique à l'originale. Il existe plusieurs algorithmes de compression sans pertes, les plus utilisés sont :
- Méthodes à base de redondances (RLE), utilisées par les formats d'images PCX, TGA, BMP, ...

- Méthodes statistiques (Huffman, ...), utilisées par les formats d'images JPEG (en partie),
- Méthodes à base de dictionnaires (LZW, ...) utilisées par les formats d'images GIF, TIFF,... .

1.5.1 La compression RLE

La méthode de compression RLE (Run Length Encoding), parfois notée RLC pour (Run Length Coding) est utilisée par de nombreux formats d'images (BMP, PCX, TIFF). Elle est basée sur la répétition d'éléments consécutifs.
Le principe de base consiste à coder un premier élément donnant le nombre de répétitions d'une valeur puis le compléter par la valeur à répéter. Ainsi, selon ce principe la chaîne "AAAAAHHHHHHHHHHHHHH" compressée donne "5A14H". Le taux de compression est ainsi 19/5=3.8. En contrepartie pour la chaîne "REELLEMENT", dans laquelle la redondance des caractères est faible, le résultat de la compression donne "1R2E2L1E1M1E1N1T" ; la compression s'avère ici très coûteuse, avec un taux < 1, soit 10/16=0.625.
En réalité la compression RLE est régie par des règles particulières permettant de compresser lorsque cela est nécessaire et de laisser la chaîne telle quelle lorsque la compression induit à un gaspillage. Ce type de compression est surtout efficace pour des images à 256 couleurs. Mais elle s'avère inefficace pour des images en 16 millions de couleurs (trop de nuances différentes pour qu'il y ait de longues séquences de pixels identiques).

1.5.2 Compression de Huffman

Le code de Huffman (1952) est un code de longueur variable optimale, c'est-à-dire tel que la longueur moyenne d'un texte codé soit minimale. On observe ainsi des réductions de taille de l'ordre de 20 à 90%. Le principe de l'algorithme de Huffman consiste à recoder les octets rencontrés dans un ensemble de données source avec des valeurs de longueur binaire variable. L'unité de traitement est ramenée au bit. Huffman propose de recoder les données qui ont une occurrence très faible sur une longueur binaire supérieure à la moyenne, et recoder les données très fréquentes sur une longueur binaire très courte. Ainsi, pour les données rares, nous perdons quelques bits regagnés pour les données répétitives. Par exemple, dans un fichier ASCII le "w" apparaissant 10 fois aura un code très long : 010100001000. Ici la perte est de 40 bits (10 x 4 bits), car sans compression, il serait codé sur 8 bits au lieu de 12. Par contre, le caractère le plus fréquent comme le "e" avec 200 apparitions sera codé par 1. Le gain sera de 1400 bits (7 x 200 bits). On comprend l'intérêt d'une telle méthode. De plus, le codage de Huffman a une propriété de préfixe : une séquence binaire ne peut jamais être à la fois représentative d'un élément codé et constituer le début du code d'un autre élément. Si un caractère est représenté par la combinaison binaire 100 alors la combinaison 10001 ne peut être le code d'aucune autre information. Dans ce cas, l'algorithme de décodage interpréterait les 5 bits comme une succession du caractère codé 100 puis du caractère codé 01. Cette caractéristique du codage de Huffman permet une codification unique à l'aide d'une structure d'arbre binaire. [7]

Algorithme de décompression
On transmet la bibliothèque de codage, puis on associe les caractères à leur code.

1.5.3 Codage LZW (Lempel-Ziv-Welch)

Ce codage est réalisé selon les étape suivantes :
- Trouver des séquences de pixels qui se répètent,
- Construire un dictionnaire associant un code unique à chaque séquence,

- Remplacer la séquence de pixels par le code correspondant. Cet algorithme est utilisé par les formats GIF, TIFF, ... , Il est intéressant lorsqu'il y a peu de couleurs.

1.6 Compression avec pertes

Principe
Elle consiste à supprimer les informations les moins indispensables pour l'oeil humain. Elle est caractérisée par :
- Une réduction du nombre des données,
- Un taux de compression plus élevé que dans le cas de compression sans pertes,
- Plus le taux de compression est élevé, plus le niveau de pertes est important et plus la qualité d'image est dégradée.

1.6.1 Compression par DCT : (JPEG) [7]

principe
le principe de la compression en utilisant la transformée en DCT est illustré sur le schéma de la figure (1.5).

1) Transformation par DCT
- L'image RVB est d'abord transformée en image YIQ (Luminance + Chrominance),
- La luminance et la chrominance sont compressées séparément,
- L'analyse de l'image se fait par blocs de 8x8 pixels (tableaux de 64 valeurs).

-Application d'une DCT (Transformée en Cosinus Discrète) au tableau de 64 valeurs, le résultat est un tableau de 64 fréquences (opération complètement réversible).

$$DCT(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} pixel(x,y) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right), \tag{1.1}$$

Exemple :

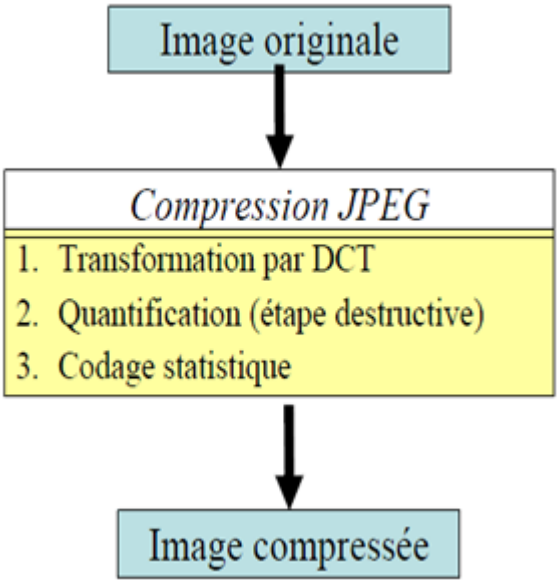


Fig. 1.5 – *Etapes de compression par DCT.*

$$\begin{bmatrix} 100 & 155 & 131 & 116 & 151 & 135 & 131 & 211 \\ 120 & 135 & 127 & 88 & 155 & 131 & 155 & 179 \\ 120 & 135 & 151 & 100 & 179 & 116 & 155 & 167 \\ 120 & 155 & 151 & 108 & 191 & 112 & 155 & 179 \\ 135 & 151 & 135 & 210 & 197 & 112 & 179 & 179 \\ 120 & 151 & 155 & 151 & 151 & 116 & 179 & 179 \\ 135 & 151 & 167 & 167 & 151 & 151 & 167 & 171 \\ 120 & 151 & 179 & 151 & 151 & 131 & 155 & 167 \end{bmatrix} \Rightarrow \begin{bmatrix} 145 & -84 & 34 & -69 & 4 & -66 & -35 & 72 \\ -45 & -28 & 28 & 19 & 10 & -54 & 5 & 15 \\ 0 & -2 & -8 & -15 & -9 & 0 & 30 & -41 \\ 9 & -14 & 15 & -11 & 5 & 8 & -12 & -32 \\ 1 & 1 & 3 & -11 & 7 & -23 & -4 & 0 \\ 18 & 4 & -17 & -10 & 4 & -10 & 7 & -10 \\ -5 & 1 & -7 & -20 & 1 & -1 & -3 & 5 \\ 3 & 1 & 1 & 9 & 2 & 7 & 2 & -2 \end{bmatrix}$$

- Basses fréquences ⇒ plage de couleurs uniforme.
- Hautes fréquences ⇒ variations brusques de couleur d'un pixel à l'autre.

On observe que les basses fréquences ont des coefficients plus élevés (donc plus significatifs) que les hautes fréquences. Les fréquences augmentent lorsque l'on se déplace à droite vers le bas sur la matrice résultante aprés l'application de la DCT.

Idée principale de la compression JPEG

Retirer des détails pour diminuer la taille du fichier image en supprimant les hautes fréquences. Le taux de compression va dépendre du seuil de l'écrêtage : plus on conserve des composantes à amplitude faible, meilleure sera la qualité (car plus il y aura de détails fins), mais plus faible sera le taux de compression.

2) Quantification

Cette étape consiste à diviser chaque valeur de la matrice de fréquences par une matrice de quantification. On obtient une matrice quantifiée dans laquelle beaucoup de coefficients sont nuls.

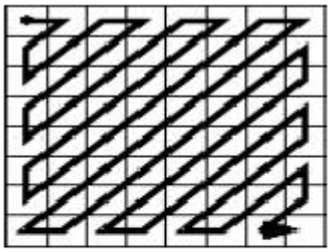


Fig. 1.6 – *Parcours en zigzag de l'image quantifiée.*

$$\begin{bmatrix} 100 & 155 & 131 & 116 & 151 & 135 & 131 & 211 \\ 120 & 135 & 127 & 88 & 155 & 131 & 155 & 179 \\ 120 & 135 & 151 & 100 & 179 & 116 & 155 & 167 \\ 120 & 155 & 151 & 108 & 191 & 112 & 155 & 179 \\ 135 & 151 & 135 & 210 & 197 & 112 & 179 & 179 \\ 120 & 151 & 155 & 151 & 151 & 116 & 179 & 179 \\ 135 & 151 & 167 & 167 & 151 & 151 & 167 & 171 \\ 120 & 151 & 179 & 151 & 151 & 131 & 155 & 167 \end{bmatrix} / \begin{bmatrix} 6 & 11 & 16 & 21 & 26 & 31 & 36 & 41 \\ 11 & 16 & 21 & 26 & 31 & 36 & 41 & 46 \\ 16 & 21 & 26 & 31 & 36 & 41 & 46 & 51 \\ 21 & 26 & 31 & 36 & 41 & 46 & 51 & 56 \\ 26 & 31 & 36 & 41 & 46 & 51 & 56 & 61 \\ 31 & 36 & 41 & 46 & 51 & 56 & 61 & 66 \\ 36 & 41 & 46 & 51 & 56 & 61 & 65 & 71 \\ 41 & 46 & 51 & 56 & 61 & 66 & 71 & 76 \end{bmatrix} =$$

$$\begin{bmatrix} 24 & -7 & 2 & -3 & 0 & -2 & 0 & 1 \\ -4 & -1 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

La matrice de quantification est définie à partir de critères psychovisuels et du taux de compression souhaité.

3)Codage statistique

La matrice quantifiée contient beaucoup de 0. Les coefficients de la matrice sont parcourus selon une séquence en zigzag (figure 1.6), ce qui génère de longues suites de 0 consécutifs, on applique ensuite la compression RLE. Les coefficients non nuls sont compressés avec la méthode statistique de Huffman qui donne des taux de compression très élevés.

Algorithme de décompression

C'est le procédé inverse de celui de la compression. Pour chaque bloc de 8x8 pixels on effectue :

- Une décompression par la méthode de Huffman,
- Une multiplication de la matrice obtenue par la matrice de quantification,

– Une application de la DCT inverse pour retrouver une image plus ou moins dégradée par rapport à l'image initiale.
la DCT inverse est donnée par :

$$pixel(x,y) = \frac{1}{\sqrt{2N}} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} C(i)C(j)DCT(i,j) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right),$$

(1.2)

avec $C(x) = \frac{1}{\sqrt{2}}$ *si* x vaut 0, et 1 si $x > 0$,

Artefacts de la compression JPEG
- La compression avec pertes entraîne une baisse de qualité de l'image,
- Apparition de blocs (car traités séparément), d'autant plus visibles que le taux de compression augmente.

1.6.2 Compression par ondelettes (Wavelets)

Évolution du format JPEG en JPEG 2000, utilisant une compression par ondelettes. C'est une méthode aussi utilisée par le format ECW (Enhanced Compression Wavelet). La compression par ondelettes est caractérisée par :
- Qualité d'image plus élevée à fort taux de compression,
- Taux de compression plus élevé que JPEG,
- Possibilité de ne charger (décompresser) qu'une partie de l'image,
- Niveaux de détail : possibilité de décompresser des versions plus petites de l'image,
- Mode optionnel de compression sans perte,
- Méthode globale portant sur toute l'image, donc pas d'apparition de blocs comme dans le cas de la compression JPEG.

Artefacts de compression : image plus floue.

1.6.3 Compression fractale [7]

Principe
Une image peut être décrite à partir d'un ensemble de motifs identiques en nombre limité, transformés par translations, rotations, symétries et agrandissements ou réductions. Pour coder l'image, il suffit de décrire les motifs originaux et les transformations utilisées. Le temps de compression est très élevé, mais le temps de décompression est semblable au JPEG. La compression fractale permet d'obtenir des taux de compression très élevés, elle fait encore l'objet de nombreuses recherches.

1.7 Conclusion

On a présenté dans ce chapitre certaines méthodes très utilisées pour la compression des images numériques. La compression sans pertes peut être utilisée sous certaines conditions, sinon elle s'avérerait plus coûteuse . La compression avec pertes permet d'avoir des taux de compression très élevés par rapport à la compression sans pertes. Ce chapitre est une introduction à la problématique qui consiste à compresser avec pertes les modèles couleurs 1D introduits dans les chapitres 3 et4.

Chapitre 2

Espaces couleurs et mesures de similarités

2.1 Introduction

Un espace de couleurs est une représentation des couleurs dans un espace permettant de les spécifier au moyen de trois composantes au plus, et dont les valeurs numériques définissent une couleur spécifique. Un espace couleur est généralement associé à une mesure de similarité pour mesurer la distance entre les couleurs, ce qui permet d'apprécier le degré de similitude entre deux images données. Les premiers espaces proposés pour spécifier des couleurs étaient basés sur des expériences sur la perception humaine pour construire une classification des couleurs. Selon leurs caractéristiques, les espaces de couleurs sont classés en quatre familles [9] :

- Les espaces primaires basés sur la théorie de trichromacité, dont le principe est qu'il est possible de reproduire n'importe qu'elle couleur en combinant des proportions appropriées des trois couleurs primaires (R,V,B),
- Les espaces de chrominance-luminance, dans lesquels un composant représente la luminosité et les deux autres composants représentent la chromaticité,
- Les espaces perceptuels, dont l'objectif est d'essayer de quantifier la perception humaine subjective des couleurs en utilisant les trois composantes d'intensité, de teinte et de saturation,
- Les espaces à axes indépendants, qui résultent des différentes méthodes statistiques permettant d'avoir le moins possible de corrélation entre les composantes.

Il existe plusieurs espaces couleurs que l'on pourrait classer suivant ces 4 familles. Dans ce chapitre, on se limitera à certains de ces espaces couleurs (RVB, XYZ, HCL et CIELAB). On abordera pour les mesures de similarités les mesures SSIM et CIELAB.

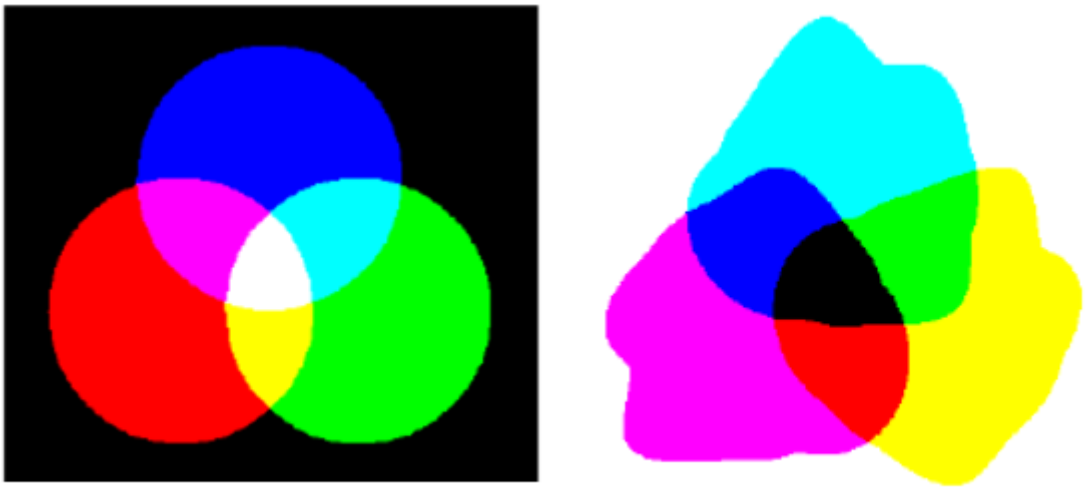


Fig. 2.1 – *Synthèse des couleurs. (a) Synthèse additive. (b) Synthèse soustractive.*

2.2 Synthèse des couleurs

La synthèse des couleurs consiste à reproduire l'ensemble des couleurs visibles à partir d'un petit nombre de couleurs, appelées couleurs primaires. Le but est de créer un rayonnement lumineux produisant la même couleur perçue que la couleur d'origine, sans reconstruire son spectre complet. On utilise soit des sources lumineuses, soit des pigments colorés. [10]

Synthèse additive
La synthèse additive est la construction des couleurs par addition des sources lumineuses, (exemples : écrans, projecteurs cinéma et vidéo, éclairages colorés,...). Plus on ajoute des composantes lumineuses, plus la couleur obtenue est claire (figure 2.1 (a)). Les 3 couleurs primaires de la synthèse additive sont le rouge (R), le vert (V) et le bleu (B). L'absence de lumière (R=V=B=0) donne le noir. La somme des 3 couleurs primaires R+V+B donne le blanc.
Les couleurs secondaires sont définies par addition de 2 couleurs primaires :
Rouge + Vert = Jaune,
Vert + Bleu = Cyan,
Bleu + Rouge = Magenta.
Les autres couleurs sont obtenues en faisant varier les intensités respectives des 3 couleurs primaires.

Synthèse soustractive
La synthèse soustractive est la construction des couleurs à partir de pigments colorés . (exemples : peinture, imprimantes, reprographie).

Les 3 couleurs primaires de la synthèse soustractive sont le Cyan (C), le Magenta (M) et le Jaune (J). Chacune absorbe une des couleurs primaires de la lumière : Le Cyan absorbe le Rouge , le Magenta absorbe le Vert et le Jaune absorbe le Bleu. L'absence

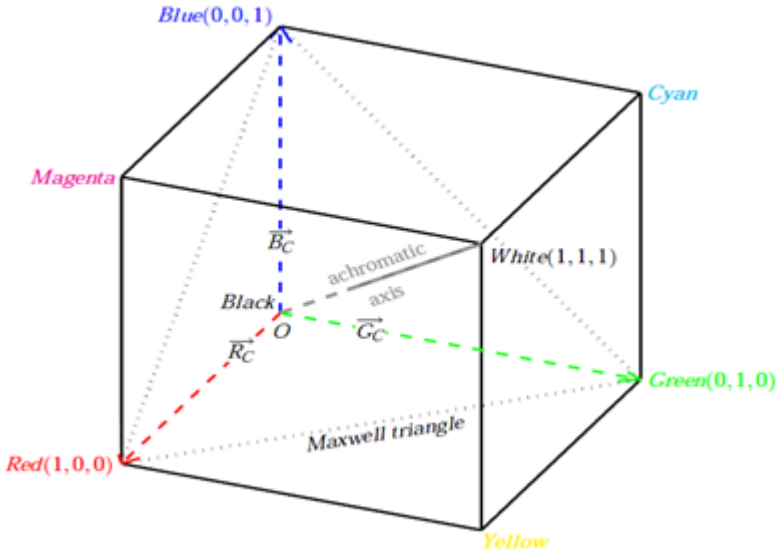


Fig. 2.2 – *Espace RVB.*

de pigment donne le blanc. La somme des 3 couleurs primaires C+M+J donne le noir (figure 2.1(b)).

2.3 Gamut

Le choix exact des couleurs primaires (longueur d'onde, spectre d'émission ou d'absorption) est important. En effet, il n'existe pas 3 couleurs primaires (additives ou soustractives) qui permettent de synthétiser toutes les couleurs visibles. Selon le choix des primaires effectué, on obtient un ensemble de couleurs différent. Le gamut est l'ensemble des couleurs qui sont réalisables à partir de 3 couleurs primaires, ou synthétisables par un matériel ou représentables dans un espace de couleurs. Les couleurs hors-gamme sont celles qui ne sont pas dans le gamut.

2.4 Espace RVB

Dans cet espace, les composantes primaires sont les trois couleurs monochromatiques, rouge, vert et bleu, de longueurs d'ondes 700 nm, 546.1 nm et 435.8 nm respectivement, et la couleur blanche de référence est égale à l'énérgie de luminance (CIE, 1986). Cet espace définit une réponse spectrale de l'oeil d'un observateur standard qui approxime la moyenne des réponses spectrales d'un ensemble d'obseravateurs humains. [9]
Pour les trois composantes primaires R, V et B, correspondent 3 vecteurs normalisés Rc, Gc et Vc respectivement. Ces vecteurs constituent la référence de cet espace vectoriel dont l'origine est le point O (figure 2.2), il correspond à la couleur noir (R=V=B=0), tandis que la référence blanche est définie par le mélange a quantités égales des trois primaires telles que R=V=B=1.

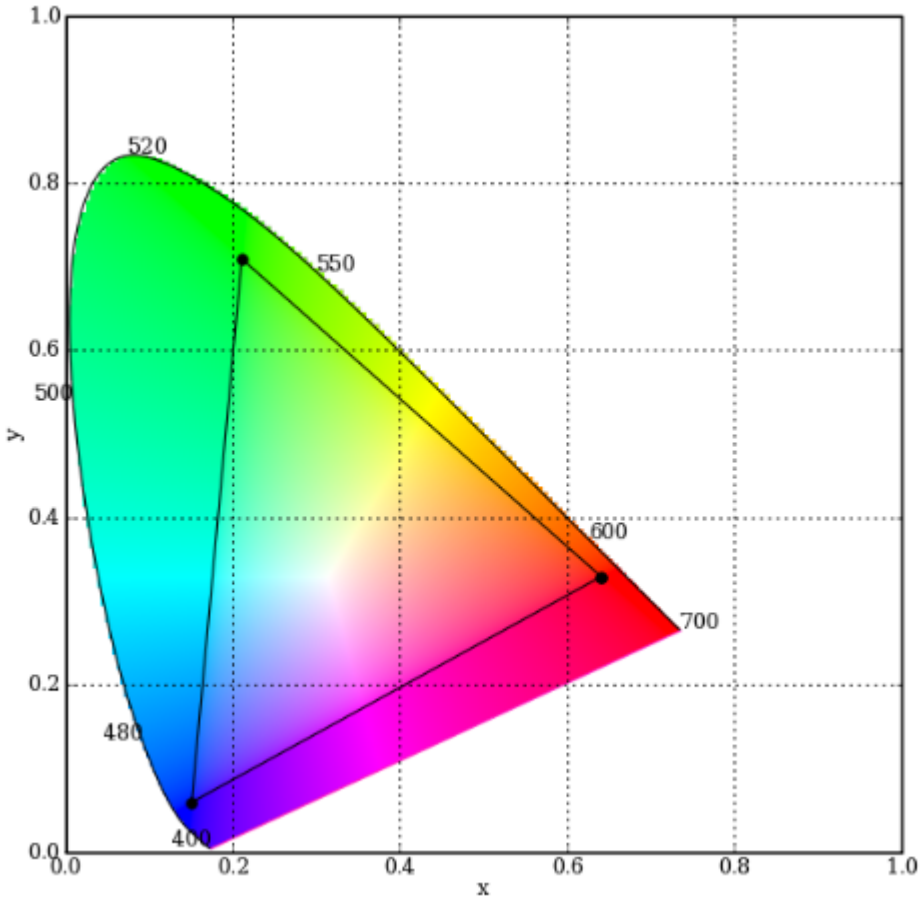


Fig. 2.3 – *Gamuts RGB et XYZ.*

Inconvénient de l'espace RVB
Théoriquement, l'espace RVB ne peut représenter des couleurs situées en dehors de son gamut (triangle RVB sur la figure 2.3). Car dans ce cas, au moins une des composantes est négative. Or, les écrans ne peuvent effectuer de synthèse soustractive.

2.5 Espace CIE-XYZ

L'espace CIE-XYZ a été créé en 1931 par la CIE, à partir des mesures sur de nombreuses personnes (l'oeil moyen). Les 3 primaires X, Y et Z sont définies comme une combinaison linéaire des 3 primaires monochromatiques R,V et B normalisées par la CIE. Elles ont été choisies pour avoir les propriétés suivantes :

- Toutes les couleurs visibles peuvent s'exprimer comme l'addition des 3 composantes positives X,Y et Z. Ainsi toutes les couleurs visibles sont à l'intérieur du cube unité,
- Y ne contient que l'information de luminance perçue, on additionne R,V et B avec des proportions de 30%, 59%, 11% respectivement qui tiennent compte de la sensibilité de loeil [11],
- Les gris correspondent aux points tels que X=Y=Z.

Le passage de l'espace RVB vers l'espace XYZ se fait par une matrice de transformation comme suit :

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.618 & 0.177 & 0.205 \\ 0.299 & 0.587 & 0.114 \\ 0 & 0.056 & 0.944 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{2.1}$$

Etalons fictifs

Les primaires X, Y et Z sont purement théoriques : elles ne sont pas réalisables physiquement. C'est pourquoi on les appelle étalons fictifs ou irréels. On ne peut donc pas construire de procédé technique (écran par exemple) basé sur ce système. Pour caractériser la chrominance, c'est-à-dire la couleur indépendamment de sa luminance, on utilise les coordonnées x, y et z telles que x + y + z = 1 et définies par :

x = X / (X +Y + Z) ; y = Y / (X + Y + Z) ; z=1-x-y .

Gamut

Dans le plan (x,y), on visualise le gamut pour les couleurs visibles et réalisables à partir des composantes x, y et z (figure 2.3). Les couleurs formant le contour du diagramme sont des couleurs pures. Le gamut XYZ est à comparer avec le triangle RVB sur la figure (2.3).

2.6 Distance Euclidienne

Les espaces RVB et XYZ utilisent la distance euclidienne pour mesurer la distance entre les couleurs. Considérons deux vecteurs $X = (x_1, x_2, ..., x_N)$ et $Y = (y_1, y_2, ..., y_N)$, la distance euclidienne ΔE entre ces deux vecteurs est donnée par :

$$\Delta E(x, y) = \sqrt{\sum_{i=1}^{N} |x_i - y_i|^2}, \tag{2.2}$$

Dans le cas des espaces RVB et XYZ : N=3, et les composantes des vecteurs x et y correspondent aux trois composantes primaires x,y et z (ou R,V et B).

2.7 L'Espace couleur HCL (Hue, Chroma et Luminance)

2.7.1 De l'espace RVB vers l'espace HCL [12]

L'espace HCL est un espace qui représente les couleurs selon les propriétés percepuelles de l'oeil. Selon la loi de proportionnalité de Weber [13], la luminance L peut être exprimée par $Q.Y$, où Y représente la luminosité capturée par un photorécepteur. On définit la luminance L par combinaison linéaire de $Max(R, V, B)$ et $Min(R, V, B)$ comme suit :

$$L = \frac{Q.Max(R, V, B) + (1 - Q).Min(R, V, B)}{2} \tag{2.3}$$

Avec $Q = e^{\alpha\gamma}$ un paramètre permettant d'ajuster la variation de luminosité entre une couleur saturée en H (Hue) et une couleur dont la teinte H contient une grande quantité

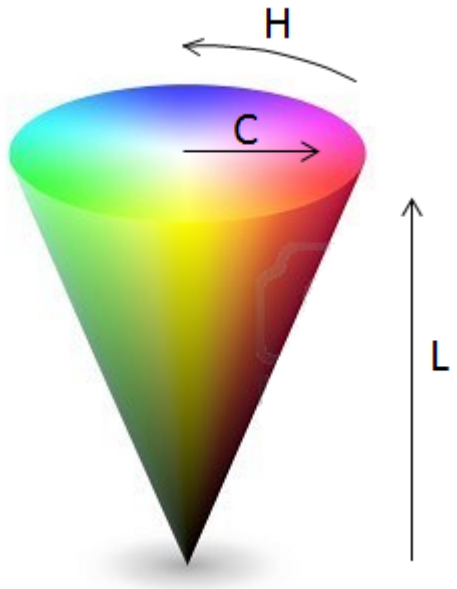


Fig. 2.4 – *Espace HCL.*

de blanc, avec $\alpha = \frac{Min(R,V,B)}{Max(R,V,B)} \cdot \frac{1}{Y_0}$ et $Y_0 = 100$. γ est un facteur de corrélation dont la valeur vaut 3. Il y a lieu de noter que lorsque $Min(R,V,B) = 0$ et $Max(R,V,B)$ varie entre 0 et 255, la luminance L prend ses valeurs entre 0 (noir) et 128. De la même façon, on définit la chroma comme suivant : $C = Q.C_n$, avec C_n représentant une mixture des trois combinaisons différentes de $R, V et B(R-V, G-V, B-R)$. la Chroma C est obtenue par :

$$C = \frac{Q.(|R-V| + |V-B| + |B-R|)}{3} \tag{2.4}$$

la teinte H peut être calculée par la formule suivante :

$$H = \arctan\left(\frac{V-B}{R-V}\right), \tag{2.5}$$

Or, la formule précedente permet d'avoir des valeurs de la teinte H variant seulement entre -90° et $+90^{\circ}$. Afin de permettre à H de varier dans une plage plus large (0 à 360°), les formules alternatives suivantes sont proposées :

$$H = \begin{cases} H = H + 180 \ \text{ si } \ ((R-G) < 0 \ \text{ et } \ (G-B) \geq 0), \\ H = H - 180 \ \text{ si } \ ((R-G) < 0 \ \text{ et } \ (G-B) < 0), \end{cases} \tag{2.6}$$

La figure (2.4) illustre la représentation géométrique de l'espace couleur HCL (forme conique).

2.7.2 De l'espace HCL vers l'espace RVB

Pour le passage de l'espace HCL vers l'espace RVB, on calcule d'abord une valeur intermédiaire $x = C.\left(1 - \left|\left(\frac{3}{\pi}H\right) \mod 2 - 1\right|\right)$ qu'on applique au système d'équations suivant [14] :

$$(R', V', B') = \begin{cases} (0,0,0) & \text{si H est non défini,} \\ (C,x,0) & \text{si } 0 \leq H < \frac{\pi}{3}, \\ (x,C,0) & \text{si } \frac{\pi}{3} \leq H < \frac{2\pi}{3}, \\ (0,C,x) & \text{si } \frac{2\pi}{3} \leq H < \pi, \\ (0,x,C) & \text{si } \pi \leq H < \frac{4\pi}{3}, \\ (x,0,C) & \text{si } \frac{4\pi}{3} \leq H < \frac{5\pi}{3}, \\ (C,0,x) & \text{si } \frac{5\pi}{3} \leq H < 2\pi, \end{cases} \tag{2.7}$$

et finalement pour avoir l'espace de départ (R, V, B) à partir de (R', V', B'), on translate les coordonnées (R', V', B') par une distance minimale $m = Min(R, V, B)$ dans les directions R,V et B comme suit :

$$(R, G, B) = \begin{cases} R = R' + m, \\ V = V' + m, \\ B = B' + m, \end{cases} \tag{2.8}$$

Les valeurs de m peuvent être aussi obtenues par la formule : $m = \left(L - \frac{1}{2}C\right)$.

2.7.3 Calcul de distance dans l'espace HCL

Pour les mesures de distances, une distance adaptée à l'espace HCL notée D_{HCL} a été définie, elle est basée sur le modèle cylindrique avec comme paramètres A_L et A_H. La distance D_{HCL} est calculée par la formule suivante :

$$D_{HCL} = \sqrt{(A_L \Delta L)^2 + A_H(C_1^2 + C_2^2 - 2C_1C_2cos(\Delta H))}, \tag{2.9}$$

Le paramètre A_L est une constante de linéarisation pour la luminance, pour le passage du modèle conique vers le modèle cylindrique. A_H est un paramètre permettant de réduire la distance entre des couleurs ayant la même valeur de H que la couleur de référence.

Remarque : Des expériences ont montré qu'en utilisant la distance D_{HCL} appliquée au modèle conique HCL, on obtient de résultats meilleurs que ceux obtenus par rapport à ceux obtenus en utilisant d'autres distances appliquées à d'autres modèles.

2.8 Similarité structurale (SSIM)

2.8.1 Contexte

Les images numériques subissent une variété de distorsions lors de l'acquisition, traitement, compression, stockage, transmission et reproduction. Une des méthodes pour apprécier le niveau de ces dégradations est la méthode subjective, qui consiste à faire une comparaison entre les images reconstruites et celles de référence en se basant sur la vision humaine, mais une telle approche présente souvent des inconvénients, comme le temps qu'elle nécessite et aussi le coût. On introduit dans cette section une mesure de similarité objective notée SSIM. Un exemple d'illustration est le suivant : soit une image originale altérée par différentes distorsions, chacune d'elles a été ajustée de telle manière à avoir approximativement la même valeur de MSE (Mean Square Error [15]), qui donne la sensibilité en erreur. Néanmoins, on peut remarquer que ces images possèdent des qualités perceptuelles différentes, par conséquent, il est difficile d'expliquer pourquoi l'image à contraste étendu par exemple est de bonne qualité (car elle présente une valeur élevée de MSE). En l'occurrence, il est facile d'expliquer ce dernier constat en utilisant la philosophie de SSIM, car presque toute l'information structurelle est perçue (valeur élevée de SSIM). [16]

2.8.2 Rappels Statistiques

Moyenne

On considère un vecteur X représentant une distribution de n données :
$X = [X1, X2, ..., Xi, ..., Xn]$, La moyenne $\overline{X}$ de cette distribution s'écrit :

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i, \tag{2.10}$$

Ecart-type

L'écart-type est une mesure de la dispersion d'un ensemble de données. D'un point de vue qualitatif, l'écart-type caractérise la largeur d'une distribution de données en mesurant la dispersion autour de la moyenne. La formule de l'écart-type σ est :

$$\sigma = \sqrt{\frac{\Sigma_{i=1}^{n}(X_i - \overline{X})^2}{n}}, \tag{2.11}$$

Variance

La variance est une autre mesure de la dispersion d'un ensemble de données. Il s'agit tout simplement du carré de l'écart-type. La formule de la variance σ^2 est donc :

$$\sigma^2 = \frac{\Sigma_{i=1}^{n}(X_i - \overline{X})^2}{n}, \tag{2.12}$$

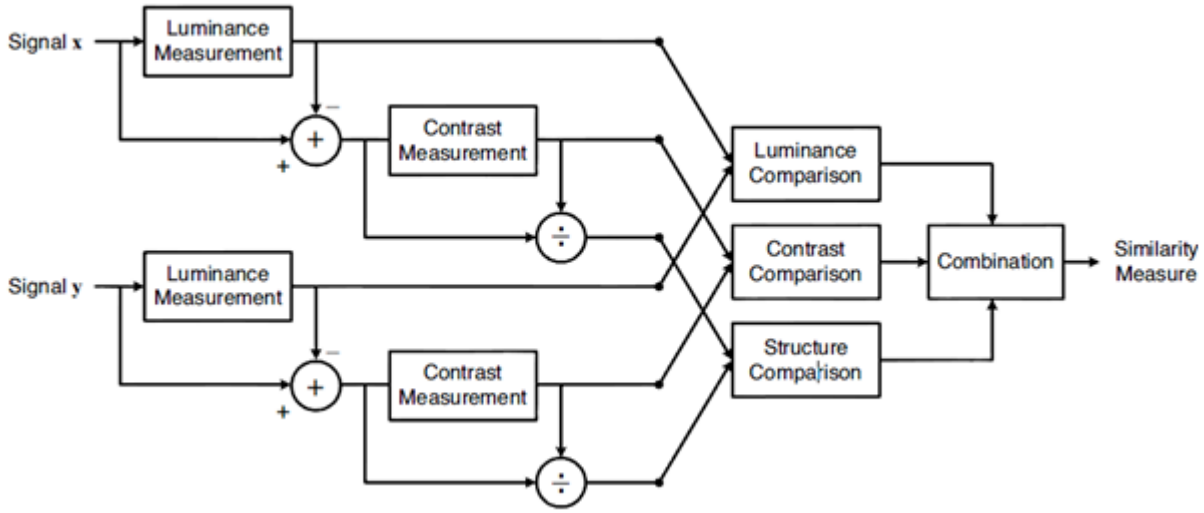


Fig. 2.5 – *Structure de la SSIM. [16]*

Covariance

La covariance est une extension de la notion de variance. Elle mesure la variation simultanée de deux variables et donne une matrice carrée. La formule de covariance entre deux variables X et Y s'écrit :

$$Cov(X,Y) = \frac{\Sigma_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{n}, \tag{2.13}$$

Le signe de la covariance entre les deux variables X et Y indique la façon avec laquelle chaque variable varie en fonction de l'autre, c'est-à-dire :

- $Cov(X,Y) > 0$: X varie proportionnellement à la variation de Y,
- $Cov(X,Y) < 0$: X varie en inversement proportionnel avec la variation de Y,
- $Cov(X,Y) = 0$: La variable X est indépendante de la variable Y.

2.8.3 Calcul de SSIM [16]

La structure de la SSIM est illustrée sur la figure (2.5). Le système effectue trois comparaisons pour calculer la valeur de SSIM : La luminance, le contraste et la structure. D'abord, la luminance de chaque signal est comparée en déterminant sa moyenne. La moyenne de l'intensité lumineuse est donnée par :

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{2.14}$$

La fonction de comparaison de luminance est donc fonction de μ_x et μ_y, où x et y représentent deux signaux images. La prochaine étape consiste à retrancher la valeur moyenne de chaque signal : $x - \mu_x$, ce qui est équivalent à une projection du vecteur x

dans l'hyperplan défini par :

$$\sum_{i=1}^{N} x_i = 0, \tag{2.15}$$

Pour la comparaison du contraste, on utilise la déviation standard comme estimateur du signal de contraste, cet estimateur est non biaisé et est donné par la formule :

$$\sigma_x = \left(\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_x)^2 \right)^2, \tag{2.16}$$

Le signal est ensuite normalisé (divisé) par sa propre déviation standard, donc les deux signaux à comparer vont avoir une déviation standard égale à l'unité. La structure de comparaison $s(x, y)$ est appliquée sur ces deux signaux normalisés $(x - \mu x)/\sigma_x$ et $(y - \mu_y)/\sigma_y$.

La fonction permettant de calculer la SSIM sera donc fonction de trois composantes :

$$S(x, y) = f(l(x, y), c(x, y), s(x, y)), \tag{2.17}$$

Par conséquent, il faut définir les trois fonctions $l(x, y)$, $c(x, y)$ et $s(x, y)$ ainsi que la fonction $f(.)$. La fonction de comparaison de luminance est définie comme suivant :

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{2.18}$$

C_1 étant une constante pour éviter une instabilité dans le cas où $\mu_x^2 + \mu_y^2$ soit proche de zéro. La fonction de comparaison du contraste possède une forme similaire définie par :

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_x\sigma_y^2 + C_2}, \tag{2.19}$$

C_2 étant une constante jouant le même rôle que C_1. La fonction de structure de comparaison est définie par :

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}, \tag{2.20}$$

μ_{xy} étant la covariance des deux signaux x et y, et C_3 une constante ayant le même rôle que C_2 et C_1. finalement, on combine les trois comparaisons définies dans (2.18), (2.19) et (2.20). La fonction globale $f(.)$ possède la forme suivante :

$$\mathrm{SSIM}(x, y) = [l(x, y)]^{\alpha} \cdot [c(x, y)]^{\beta} \cdot [s(x, y)]^{\gamma}, \tag{2.21}$$

Or, la fonction SSIM doit vérifier les conditions suivantes :
1- Symétrie : $S(x, y) = S(y, x)$.
2- Bornée : $S(x, y) \leq 1$.
3- Un seul maximum : $S_{max} = 1$, qui est atteint si et seulement si $x = y$.

On peut vérifier qu'en choisissant : $\alpha = \beta = \gamma = 1$ et $C_3 = C_2/2$, les trois propriétés citées ci-dessus seront vérifiées. D'où la formule finale de SSIM :

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{2.22}$$

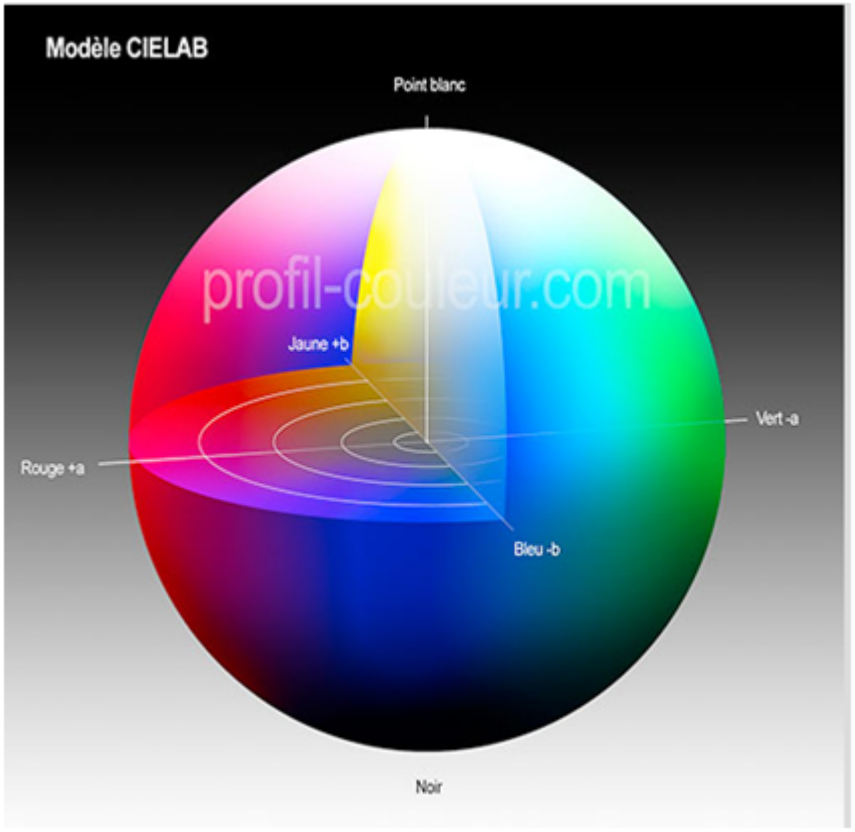


Fig. 2.6 – *Espace L*a*b*.*

2.9 Le modèle L*a*b*

2.9.1 Description du modèle L*a*b*

CIELAB (plus précisément L*a*b*) est un modèle de représentation des couleurs, il caractérise une couleur à l'aide d'un paramètre d'intensité correspondant à la luminance et de deux paramètres de chrominance qui décrivent la couleur. Il a été spécialement étudié pour que les distances calculées entre les couleurs correspondent aux différences perçues par l'oeil humain. Ce système vise à uniformiser la perception des différences de couleurs. Les relations non-linéaires pour L*, a* et b* ont pour but d'imiter la réponse logarithmique de l'oeil (dans l'espace L*a*b l'oeil détecte 1 point de variation de a* ou b* pour 5 points de L*). Contrairement au modèle RVB, l'espace L*a*b* possède la propriété d'uniformité perceptuelle, ce qui signifie que la même quantité de variation de la couleur produit pratiquement la même variation du point de vue perception humaine. En plus, la représentation de sa composante L coïncide avec la perception humaine de la luminance.

Dans ce modèle, les informations de chromaticité s'appuient sur deux axes de couleurs perpendiculaires. L'axe jaune / bleu et l'axe vert / rouge. L'opposition des couleurs est matérialisée par un point neutre central qui prend la valeur 0 (figure 2.6).

2.9.2 De l'espace RVB vers l'espace L*a*b*

La première étape consiste à passer des composantes RVB aux composantes XYZ. On utilise pour cela la transformation donnée par (2.1).

Ensuite, il s'agit de passer de l'espace XYZ vers l'espace L*a*b*. On utilise alors les formules de transformation suivantes :

$$\begin{cases} L = 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 & \text{pour} \quad \frac{Y}{Y_n} > 0.008856 \\ L = 903.3\frac{Y}{Y_n} & \text{pour} \quad \frac{Y}{Y_n} \leq 0.008856 \end{cases} \tag{2.23}$$

$$a = 500(f(\frac{X}{X_n}) - f(\frac{Y}{Y_n})), \tag{2.24}$$

$$b = 200(f(\frac{Y}{Y_n}) - f(\frac{Z}{Z_n})), \tag{2.25}$$

avec :
pour $t > 0.008856 \;\; f(t) = \sqrt[3]{t}$
pour $t \leq 0.008856 \;\; f(t) = 7.7787t + \frac{16}{116}$

X_n, Y_n et Z_n correspondent au blanc décrit dans l'espace XYZ. On les obtient pour $(R, G, B) = (255, 255, 255)$.

2.9.3 De l'espace L*a*b* vers l'espace RVB

On convertit les composantes L*a*b* vers l'espace XYZ. Pour cela, on inverse les formules précédentes :

$$\begin{cases} Y = Y_n.\frac{L}{903.3} & \text{pour} \quad L \leq 8, \\ Y = Y_n.\left(\frac{L+16}{116}\right)^3 & \text{pour} \quad L > 8, \end{cases} \tag{2.26}$$

$$X = X_n.f(\frac{a}{500} + \frac{L+16}{116}), \tag{2.27}$$

$$Z = Z_n.f(\frac{L+16}{116} - \frac{b}{200}), \tag{2.28}$$

avec :

pour $t > 0.207 \; f(t) = t^3$,
pour $t \leq 0.207 \; f(t) = \frac{116.t-16}{903.3}$,

Ensuite, par application de la matrice inverse (XYZ $\rightarrow$ RVB), on obtient les composantes RVB :

$$\begin{pmatrix} R \\ V \\ B \end{pmatrix} = \begin{pmatrix} 1.867 & -0.533 & -0.343 \\ -0.967 & 1.998 & -0.031 \\ 0.057 & -0.118 & 1.061 \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{2.29}$$

2.9.4 Inconvénients du modèle CIELAB

Ce modèle est loin d'être parfait. Bien qu'il ait été créé à l'origine pour fournir un modèle uniforme, il est loin de l'être totalement. Les écarts les plus visibles se situent dans la région des bleus saturés qui présentent une dérive vers le violet lors des conversions induisant à une perte. La luminosité L*, bien qu'elle soit proche de la réponse logarithmique de l'oeil à la luminance [17] , n'est qu'une grossière approximation de la véritable distribution tonale que l'oeil est capable d'adopter.
En plus, une des difficultés majeures de ce système est qu'il utilise un système mixte de repérage des points de couleur. La saturation est mesurée de manière cartésienne, alors que la teinte et la luminosité sont mesurèes de manière angulaire. [18]

2.10 La mesure S-CIELAB

2.10.1 Pourqoi la mesure S-CIELAB et pas la mesure CIELAB ?

Le système CIELAB a été créé dans une époque où la plupart des applications traitaient des images contenant de grandes surfaces uniformes colorées, et par conséquent la mesure CIELAB de ce système avait été adaptée et testée avec ce types d'images. Or, plusieurs études psychophysiques montrent que la discrimination et l'apparence des petites surfaces diffèrent des mêmes mesures faites sur de grandes surfaces uniformes. Par conséquent, l'application de la mesure CIELAB pour la reproduction des erreurs des couleurs locales ne donne pas des résultats précis. D'où la nécessité d'introduire une autre mesure de distance pour les erreurs dans les images colorées reproduites, il s'agit de l'extension de CIELAB notée S-CEILAB (Spatial CIELAB). L'utilisation d'un filtrage spatial fait la différence essentielle entre les mesures S-CIELAB et CIELAB. En effet, le filtrage spatial est appliqué sur les données de l'image afin de simuler le brouillard spatial du système visuel humain [19]. La figure (2.7) montre les étapes de calcul de la mesure S-CIELAB, ces étapes assurent à ce que l'extention S-CIELAB soit cohérente avec la mesure CIELAB dans le cas des grandes surfaces uniformes. Par conséquent, la représentation S-CIELAB obtenue par le schéma de la figure(2.7), comporte le traitement de la mesure CIELAB et le filtrage spatial.

D'abord, l'image est représentée dans l'espace couleur CIE XYZ. En suite elle est transformée vers un espace couleur (dit d'adversaire). Chaque image ainsi transformée est convoluée avec une fonction de kernel dont l'allure de sa réponse est déterminée par la sensibilité visuelle spatiale de la dimension de la couleur. A présent, on passe à l'étape de filtrage spatial, l'image filtrée est ensuite transformée à nouveau vers la représentation CIE XYZ. On utilise la transformation de séparation de couleurs pour deux raisons :

- Une telle transformation est facile à calculer,
- Les expériences psychophysiques préconisent que la représentation visuelle humaine des modèles colorés simples est la même que le modèle couleur séparable.

On note ici que les filtres spaciaux ont été estimés en se basant sur ces mesures psychophysiques.

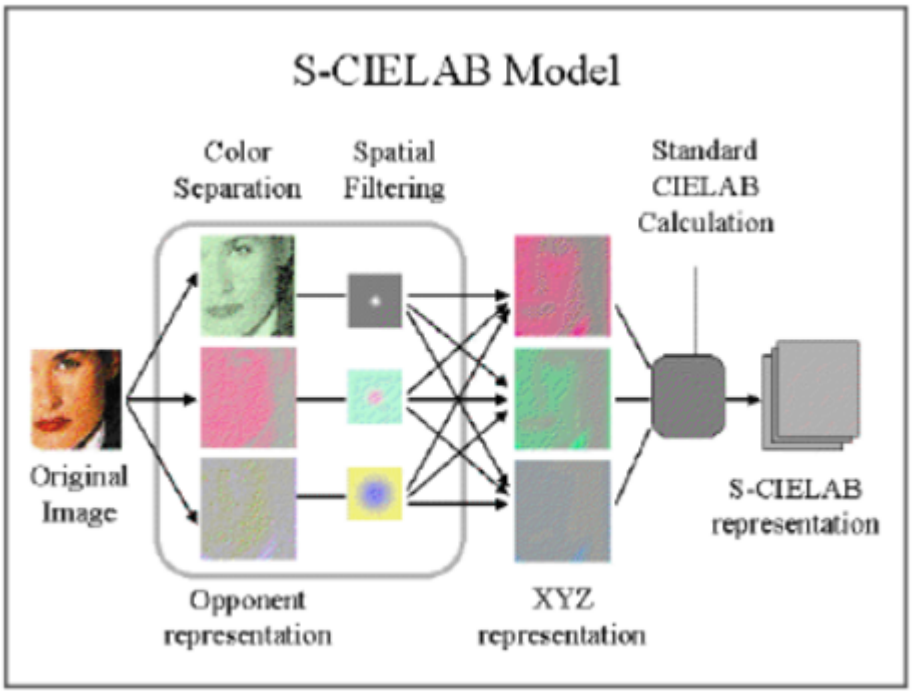


Fig. 2.7 – *Structure de la S-CIELAB. [16]*

2.10.2 Calcul de l'erreur de reproduction

Elle est notée ΔE_s et calculée dans la représentation S-CIELAB, et ce exactement comme la distance ΔE conventionnelle dans l'espace CIELAB. [19]

2.11 conclusion

On a introduit à travers ce chapitre quelques espaces couleurs et mesures de similarités. Les espaces HCL et CIELAB possèdent la propriété d'être perceptuels vis-à-vis de la vision humaine. Les mesures de similarités SSIM et S-CIELAB sont des outils d'appréciation des degrés de similtude entre les images, elles représentent une approche objective remplaçant l'approche d'évaluation subjective.

Chapitre 3

Modèle couleur à une dimension

3.1 Introduction

Le traitement d'images couleurs rigoureux nécessite l'utilisation de l'information couleur réelle et ainsi d'utiliser trois composantes, ce qui assure des résultats précis. Or la majorité des systèmes sont restreints par le temps de traitement et par la capacité mémoire, surtout quand le temps réel est requis ; par conséquent la manipulation de trois composantes est désavantagée par ces restrictions. Les propriétés perceptuelles très intéressantes de l'espace HCL ne peuvent être exploitées si on convertit les images couleurs en niveaux de gris, car la représentation en niveaux de gris ne préserve pas toute l'information couleur. On introduit dans ce chapitre un modèle couleur qui réduit la dimensionnalité de l'espace HCL à 1, tout en préservant ses propriétés perceptuelles dans une seule dimension. [1]

3.2 Modèles couleurs 1D

Le modèle couleur 1D présenté dans ce chapitre est un modèle original, il n'existe pas de travaux similaires pour réduire la dimension de l'espace HCL à 1. Néamoins il existe d'autres travaux ayant été réalisés sur d'autres espaces couleurs, et dont l'objectif est de réduire la dimensionnalité (pas forcément à 1). Mais la plupart de ces modèles sont basés sur des statistiques, et proposent de procéder par quantification vectorielle pour représenter deux (ou plus) composantes par une seule valeur quantifiée appartenant à un dictionnaire donné [20], [21], or ces méthodes sont plutôt des méthodes de compression avec pertes. Ce qui fait que les modèles de réduction basée sur le principe de paramétrisation (comme c'est le cas pour le modèle 1D présenté dans ce chpitre) sont rares. On propose de présenter très brièvement un modèle existant déjà, qui exploite le principe de paramétrisation pour représenter l'information couleur de l'espace CIELAB par une seule dimension.

Principe Le modèle en question utilise une courbe paramétrique 3D, qui traverse des

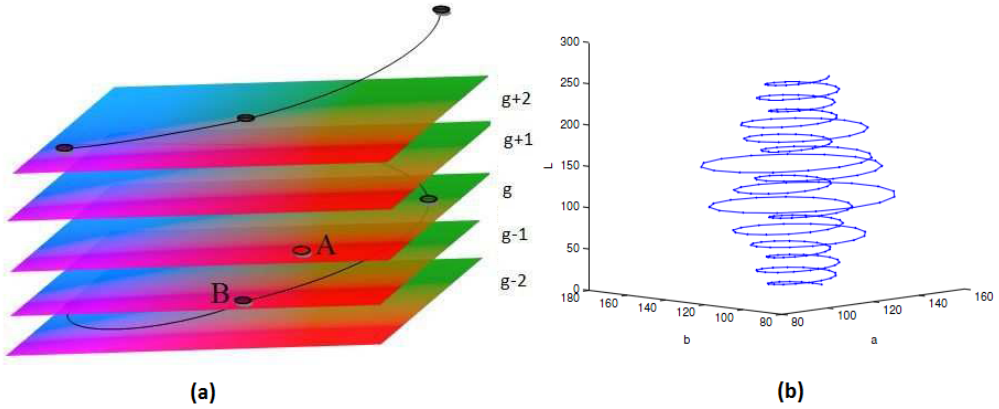


Fig. 3.1 – *(a) Modèle couleur avec paramétrisation. (b) Courbe 3D.*

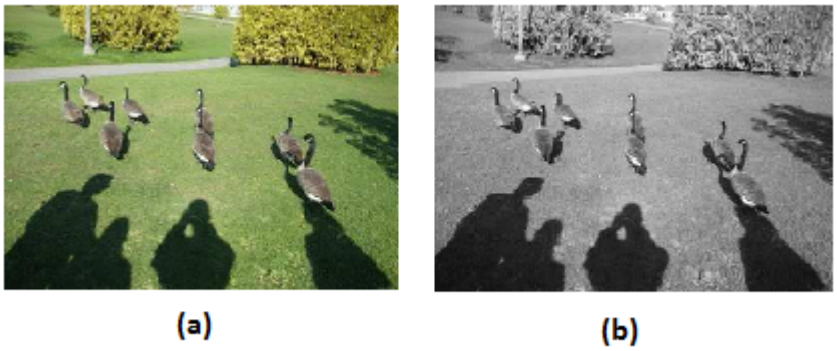


Fig. 3.2 – *Images reconstruites. (a) Image couleur. (b) Image à N.G.*

plans horizontaux de l'espace CIELAB (figure 3.1(a)). La courbe choisie pour ce modèle est illustrée sur la figure(3.1(b)). Cette courbe est choisie telle que le paramètre qui la représente correspond à la composante de luminance de l'espace CIELAB. Par conséquent, les images représentées par ce modèle 1D sont des images en niveaux de gris (figure 3.2(b)) [22].

3.3 Principe de la réduction 1D

Le principe de base est de définir un seul paramètre permettant de relier les trois composantes de l'espace HCL entre elles, et ainsi pouvoir écrire chaque composante en fonction de ce paramètre. Ainsi, il suffit de disposer de ces équations paramétriques et du paramètre en question pour retrouver les trois composantes originales h, c et l. Ce problème de paramétrisation sera traité en deux parties complémentaires : on s'intéressera d'abord à la réduction de dimentionnalité de 3D à 2D, puis on exploite ce travail pour réduire la dimensionnalité à 1D.

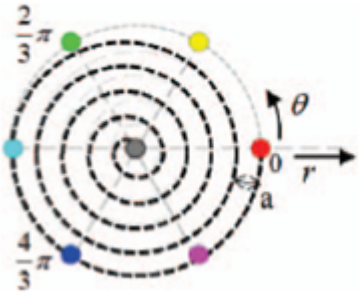


Fig. 3.3 – *Spirale d'Archimède [14].*

3.4 Réduction 3D vers 2D

3.4.1 Principe

L'objectif est d'utiliser seulement deux composantes au lieu de trois, pour représenter toute l'information couleur de l'espace HCL. L'idée est d'approximer la base circulaire du cône solide HCL par une spirale. Cette spirale étant décrite par un seul paramètre polaire θ, permettant d'écrire la chroma c en fonction de la teinte h, ce qui signifie que les composantes c et h seront représentées uniquement par le paramètre θ. Ce nouveau paramètre est alors combiné avec la luminance pour représenter toute l'information couleur en seulement 2D. [14]

3.4.2 Spirale d'Archimède

Pour ce modèle 2D, la spirale proposée est une spirale d'Archimède. Celle-ci possède la propriété de conserver un pas constant entre ses tours successifs, donc l'échantillonnage du disque de chromaticité est de type uniforme.

Construction géométrique
La spirale d'Archimède est la courbe décrite par un point M en déplacement uniforme sur une droite en rotation uniforme sur elle-même autour d'un point [23](figure(3.3)). Le sillon des disques vinyles est une spirale d'Archimède.

La spirale d'Archimède peut être aussi décrite par l'équation polaire suivante :

$$r = \frac{\mathrm{a} \cdot \theta}{2\pi}, \tag{3.1}$$

où 'a' est une constante représentant la distance entre deux tours successifs. θ est l'angle polaire de la spire, avec $\theta \in [0; 2\pi K_C]$, K_C étant le nombre total de tours. La figure(3.3) illustre la représentation géométrique de cette spirale.

Cette spirale est définie pour des angles positifs. La spirale d'équation $r = \frac{\text{-a}\cdot\theta}{2\pi}$, définie pour des angles négatifs possède la même forme mais tournerait dans le sens contraire.

3.4.3 Application : réduction 2D

La spirale d'Archimède présentée dans la section précédente est utlisée pour le modèle de réduction 3D vers 2D. Le disque de chromaticité est approximé par une spirale comme suit [14] :

$$h(\theta) = \theta - 2\pi k, \tag{3.2}$$

où $k \in [0, 1., K_C]$, K_C étant le nombre de tours, et l'axe C est uniformément échantillonné en $(K_C + 1)$ valeurs c_k, avec un pas égal à a (figure (3.3)). c_k dépend de h, ou bien encore de l'angle θ, on définit $c_k(\theta)$ alors ainsi :

$$c_k(\theta) = r(h(\theta)) + \text{a}.k, \tag{3.3}$$

on affecte aux deux points d'extrémités de la spire les valeurs : $(0, 0)$ pour le point de départ (c_0, h_0), et $(1, 0)$ pour le point de la fin(c_{max}, h_{max}). En remplaçant ces valeurs dans (3.2) et (3.1), on trouve a $= \frac{1}{K}$. Dans le cas continu où $K \to \infty$, on peut écrire :

$$c = r(\theta) = \frac{\theta}{2\pi K} \Rightarrow \theta = 2\pi K c, \tag{3.4}$$

En remplaçant (3.4) dans (3.2), on trouve :

$$k = \text{round}\left(K.c \; - \; \frac{h}{2\pi}\right), \tag{3.5}$$

Avec round(.) est la fonction d'arrondi qui attribue le nombre entier le plus proche à k. On peut à présent définir la transformation de (c, h) vers θ comme suit :

$$\theta = h + 2\pi.\text{round}\;\left(K.c - \frac{h}{2\pi}\right), \tag{3.6}$$

la transformation inverse de θ vers (c, h) est entièrement définie par (3.2) et (3.3) avec :

$$k = \text{round}\left(\frac{\theta - \theta \mod (2\pi)}{2\pi}\right), \tag{3.7}$$

3.5 Réduction 2D vers 1D

3.5.1 Principe

On se base sur le travail effectué précédemment (réduction de 3D vers 2D), pour réduire la dimensionnalité à 1. Pour ce faire, il faut un nouveau paramètre en fonction du quel s'exprimeront θ et l. Ce nouveau paramètre va alors représenter toute l'information couleur contenue initialement dans les composantes h, c et l. Pour inclure la composante de luminance l dans ce nouveau paramètre, on échantillonne uniformément l'axe de luminance, ce qui donne comme résultat n niveaux de luminance donnés par la relation suivante :

$$l_n = \frac{n}{K_L}, \text{ où } n \in [0, 1, ..., K_L], \tag{3.8}$$

pour garder une précision uniforme pour les n disques de chromaticité ainsi définis, on impose un plus grand nombre de tours de spires pour les sections les plus larges du cône . Donc à chaque niveau de luminance l_n correspond un rayon spiral défini par :
$r(\theta_n) = \text{a}.\theta_n$ avec $\theta_n \in [0,\ 2\pi n]$. L'étape suivante consiste à relier tous les K_L spirales à ce même paramètre. Pour cela, pour un point appartenant à la spirale de niveau l_n, on introduit l'angle cumulatif (CA) noté ζ [1] :

$$\zeta = \zeta_{n-1} + \theta, \tag{3.9}$$

avec :

$$\zeta_{n-1} = \sum_{j=0}^{n-1} 2\pi \cdot j = \pi \cdot n(n-1) \tag{3.10}$$

le paramètre ζ définit par (3.9) modélise la structure géométrique du cône solide HCL par une seule dimension, le modèle 1D correspondant est nommé "angle cummulatif"' (CA Model). ζ_{n-1} représente l'angle cumulatif (CA) des spires au niveau l_{n-1}.
On note ici que ce modèle 1D est reversible de et vers l'espace HCL, et par conséquent vers tous les espaces convertissables de et vers HCL, comme l'espace RVB.

3.5.2 Transformation inverse : de 1D vers 3D

Afin de revenir vers la représentation HCL, on utilise le paramètre ζ et on procède aux étapes suivantes :

- D'abord on récupère la composante de luminance l et on détermine son approximation l_n,
- On retrouve en suite la paire (h, c) correspondant au niveau de luminance l_n.

Or, suivant la définition du modèle (CA), on trouve :

$$\zeta_{n-1} \leq \zeta < \zeta_n \Longrightarrow l \approx l_n \text{ et } \theta = \zeta - \zeta_{n-1} \tag{3.11}$$

En résolvant les deux inégalités précédentes, on détermine une expression analytique de n nous évitant une recherche exhaustive entre les intervalles $[\zeta, \zeta_{n-1}[$, n est alors déterminé en résolvant les deux inégalités suivantes obtenues comme étape intermédiaire :

$$\Big(\frac{-1+\sqrt{1+\frac{4}{\pi}\zeta}}{2}\Big) < n \leq \Big(\frac{1+\sqrt{1+\frac{4}{\pi}\zeta}}{2}\Big) \tag{3.12}$$

Or $n \in \mathbb{N}$, on obtient alors :

$$n = \left\lfloor \frac{1}{2}\sqrt{1+\frac{4}{\pi}\zeta + \frac{1}{2}} \right\rfloor \tag{3.13}$$

Et finalement, on tire l_n de (3.8), et pour avoir la paire (h, c), on suit les mêmes étapes présentées dans la section précédente (3D vers 2D) en utilisant l'angle θ de (3.6), ce qui aboutit finalement aux expressions suivantes :

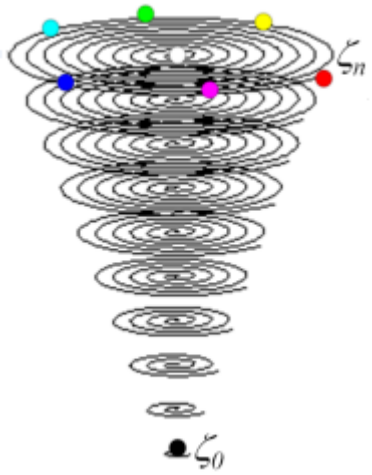


Fig. 3.4 – *Modèle CA [1].*

$$\begin{cases} h \equiv \zeta \bmod 2\pi, \\ c = \frac{1}{2\pi K_C}\left(\zeta - \left\lfloor \frac{1}{2}\sqrt{1+\frac{4}{\pi}\zeta} + \frac{1}{2} \right\rfloor \left\lfloor \frac{1}{2}\sqrt{1+\frac{4}{\pi}\zeta} - \frac{1}{2} \right\rfloor\right), \\ l = \frac{1}{K_L}\left\lfloor \frac{1}{2}\sqrt{1+\frac{4}{\pi}\zeta} + \frac{1}{2} \right\rfloor, \end{cases} \tag{3.14}$$

La figure(3.4) montre la représentation spatiale du modèle CA présenté.

3.5.3 Distance

La distance pour le modèle d_{CA} a été définie en s'inspirant de la distance cylindrique usuelle d_{cyl} comme suit :

$$d_{CA}(\zeta_p, \zeta_q) = \sqrt{(\Delta l)^2 + (\Delta c)^2 + 4 \cdot c_p \cdot c_q \cdot \sin^2(\frac{\Delta h}{2})}, \tag{3.15}$$

Avec c_p et c_q des valeurs de la chrominance correspondant à ζ_p et ζ_q respectivement. Pour le modèle CA, cette formule a été simplifiée en considérant une valeur normalisée de $\Delta\zeta$ à la place du premier terme Δl. Le facteur de normalisation a_1 est tel que l'on obtienne une distance totale de 1 entre les deux couleurs de référence noir et blanc. On trouve finalement :

$$d_{CA}(\zeta_p, \zeta_q) = a_1 \cdot \ |\Delta\zeta| + |\Delta c| + 2 \cdot \sqrt{c_p \cdot c_q} \ \left|\sin(\frac{\Delta\zeta}{2})\right| \tag{3.16}$$

Malgré que cette expression est plus simple que l'originale, elle reste néanmoins complexe du fait qu'elle fasse intervenir des valeurs de chrominance qui sont calculées lourdement à partir de (1.19). C'est une première étape pour définir une meilleure distance moins complexe pour évaluer le modèle (CA).

3.6 Conclusion

On a présenté dans ce chapitre les étapes de construction du modèle couleur 1D. Ce modèle a pour but de faciliter le traitement des images couleurs, et ce en manipulant uniquement une seule dimension au lieu de trois, cette dimension est représentée par le paramètre ζ qui contient toute l'information couleur de l'espace couleur HCL. L'évaluation des performances de ce modèle est détaillée dans le chapitre 5.

Chapitre 4

Modèles proposés dérivés du modèle CA

4.1 Introduction

Le modèle CA original présenté dans le chapitre précédent est une première étape pour modéliser le cône solide HCL par un seul paramètre préservant toutes ses propriétés perceptuelles, car comme illustré dans le chapitre suivant, l'information couleur de l'espace HCL n'est pas totalement préservée dans certains cas (comme pour $K_C > K_L$) par le modèle CA, ce qui signifie que le modèle CA original présente certaines limitations. On propose dans ce chapitre d'autres modèles 1D dérivés du modèle CA et reposant sur le même principe de réduction. Ces modèles ont pour objectif d'améliorer les performances du modèle CA original, en éliminant ses points faibles et en renforçant ses points forts. Ce chapitre est organisé comme suit : - D'abord on abordera l'échantillonnage de l'axe de chrominance C, et on proposera alors un modèle dérivé du modèle CA nommé CAs, dont l'échantillonnage de l'axe C est non uniforme . - On proposera en suite un autre modèle dérivé du modèle CA dont l'intérêt est de montrer les propriétés intéressantes du modèle CAs.

- On s'intéressera par la suite à l'échantillonnage de l'axe de luminance. On proposera alors un dernier modèle dérivé du modèle CA, qui combine les propriétés du modèle CAs avec un échantillonnage non uniforme de l'axe de luminance. - Et à la fin du chapitre, on proposera une méthode basée sur la DCT et la mesure S-CIELAB pour la détermination des valeurs optimales des paramètres K_C et K_L, on introduira ensuite un algorithme simple pour un codage efficace des composantes C et L.

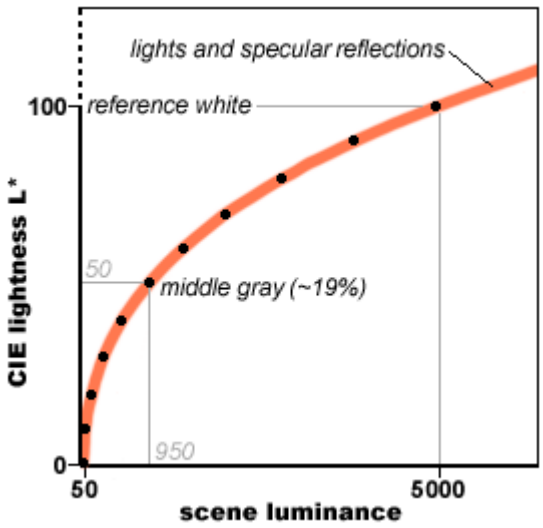


Fig. 4.1 – *Réponse de l'oeil en fonction de l'intensité lumineuse. [17]*

4.2 Modèle CAs proposé

4.2.1 Principe de codage efficace de la chrominance

La luminance L n'est pas en rapport avec le fonctionnement perceptif de l'oeil. En effet, un doublement de la valeur de L n'aboutit pas à un doublement de la sensation de luminosité. Une loi (strictement expérimentale) connue sous le nom de loi de Weber-Fechner dit que la sensibilité de l'oeil pour un niveau de luminance donné est du type : $dL/L = 0.01$. Autrement dit, la capacité de l'oeil à distinguer deux niveaux différents dépend de la luminance courante, et deux niveaux, pour être distincts, doivent différer d'au moins 1%. La loi gouvernant la sensation de luminosité n'est donc absolument pas linéaire, mais plutôt logarithmique [13] (figure(4.1)).

Notre idée est d'exploiter les propriétés de la réponse de l'oeil en fonction de la luminance, pour déduire ensuite la réponse de l'oeil en fonction de la saturation des couleurs. On sait qu'une couleur saturée à 100% contient 0% de luminance [12], et une couleur délavée contient plus de luminance qu'une couleur saturée. Ceci signifie que pour les grandes valeurs de la chroma c (couleurs saturées), le niveau de luminance L correspondant est faible, par conséquent on se retrouve dans la région où l'oeil possède plus de sensibilité pour la luminance, ceci signifie qu'une petite variation de chrominance pour ces valeurs de c engendre une variation de L perçue par l'oeil plus importante qu'une même variation pour de petites valeurs de c. Donc la Chroma c est plus sensible aux variations lorsque ses valeurs sont importantes. On se propose alors de tirer profit de notre raisonnement, et ainsi échantillonner l'axe de chrominance de telle façon à ce que l'on donnera plus de précision pour les grandes valeurs de la chroma c. Plusieurs allures sont possible pour la fonction d'échantillonnage répondant à ces critères. On propose dans la section qui suit une spirale de type spring curve [24], dont la distance entre les tours successifs représente une fonction d'échantillonnage qui satisfait les critères ci-dessus.

4.2.2 Application au modèle CA

La spirale d'Archimède utilisée dans le modèle original CA présente un pas constant entre deux tours successifs. Or nous avons vu dans la section précédente qu'une forme de codage de la chroma c autre que la forme uniforme donnerait de meilleurs résultats, en permettant de compresser plus tout en gardant une précision et une qualité acceptables de l'image, ceci est justifié par le fait que les grandes valeurs de c sont plus sensibles aux erreurs de quantification que les petites valeurs. On se propose d'utiliser une spirale de type spring curve [24], dont les caractéristiques répondent à notre objectif. Cette spirale présente des pas décroissants en fonction de l'accroissement des valeurs de la chroma (i.e. donner plus de précision pour des valeurs plus grandes de c) . Le rayon de la spirale (spring curve) dépend de l'angle polaire θ suivant la relation :

$$r(\theta) = \frac{\mathrm{a}}{1 + e^{m\theta}}, \tag{4.1}$$

Où 'a' et $'m'$ sont des paramètres à ajuster pour adapter cette spirale au disque de chromaticité du cône solide HCL. En traçant cette fonction dans le plan polaire, on obtient une courbe partant d'un cercle asymptote et aboutissant à un point au centre du cercle, comme le montre la figure(4.2(b)). Or, si on applique cette spirale à notre modèle spiral, les valeurs discrètes de la chroma auront une variation inversement proportionnelle avec θ, ce qui correspond à l'inverse de la variation cherchée pour notre modèle. Pour résoudre ce problème, on modifie l'équation polaire $r(\theta)$, et ce en posant $1/\theta$ à la place de θ . On obtient alors la nouvelle équation polaire suivante (figure(4.2(a)) :

$$r(\theta) = \frac{\mathrm{a}}{1 + e^{\frac{m}{\theta}}} \tag{4.2}$$

La teinte h dépend de l'angle θ suivant la relation (3.2).
La chroma c dépend de h ainsi :

$$c(\theta) = r(h(\theta) + 2\pi k), \tag{4.3}$$

k étant le même paramètre utilisé dans le cas de la spirale d'Archimède. ce qui donne :

$$c(\theta) = \frac{\mathrm{a}}{1 + e^{\frac{m}{(h(\theta)+2\pi k)}}}, \tag{4.4}$$

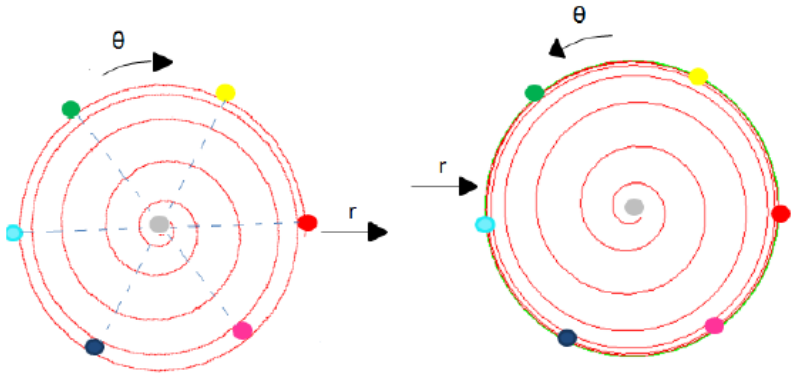
pour la détermination de a, on utilise le point particulier de la spirale $(c_{max}, h_{max}) = (1, 0)$ en remplaçant ces valeurs dans (4.2) et (4.3), il vient :

$$\mathrm{a} = 1 + e^{\frac{m}{2\pi . K_C}}, \tag{4.5}$$

dans le cas continu ($K \longrightarrow \infty$), on aura :

$$c(\theta) = r(\theta) = \frac{\mathrm{a}}{1 + e^{\frac{m}{\theta}}} \Longrightarrow \theta = \frac{m}{\ln\left(\frac{\mathrm{a}-c}{c}\right)}, \tag{4.6}$$

On fait remarquer ici que $(\frac{\mathrm{a}-c}{c})$ est toujours positif et supérieur à 1, donc la fonction logarithme ici est définie pour toutes les valeurs de la chroma c. En remplaçant (4.6) dans (4.3), il vient :



(a) rayon divergent (b) rayon convergent

Fig. 4.2 – *Spirales spring curve.*

$$k = \text{round}\Big(\frac{1}{2\pi}\Big(\frac{m}{\ln(\frac{a-c}{c})}\Big) - h\Big), \tag{4.7}$$

Et en remplaçant cette valeur de k dans (4.3), on obtient alors θ en fonction de c et h comme suit :

$$\theta = h + 2\pi.\text{round}\Big(\frac{1}{2\pi}\Big(\frac{m}{\ln(\frac{a-c}{c})}\Big) - h\Big), \tag{4.8}$$

En utilisant cette nouvelle expression de θ , on définit alors le nouveau paramètre ζ_s réduisant la dimensionalité à 1 comme suit :

$$\zeta_s = \zeta_{s_{n-1}} + \theta, \tag{4.9}$$

le modèle correspondant sera nommé CAs. $\zeta_{s_{n-1}}$ a la même forme que ζ_{n-1} dans (3.10).

La figure (4.3) illustre une représentation spatiale du modèle CAs.

En suivant les mêmes étapes que dans le cas de la spirale d'Archimède, on obtient les relations de passage du modèle (CAs) vers l'espace HCL :

$$\begin{cases} h \equiv \zeta_s \bmod 2\pi, \\ c = \dfrac{1}{\big(1 + e^{\frac{m}{2\pi K_C}}\big) \cdot \big(1 + e^{\frac{m}{(\zeta_s - n.(n-1).\pi)}}\big)}, \\ \\ l = \frac{1}{K_L}\left\lfloor \frac{1}{2}\sqrt{1 + \frac{4}{\pi}\zeta_s} + \frac{1}{2} \right\rfloor, \end{cases} \tag{4.10}$$

avec : $n = \left\lfloor \frac{1}{2}\sqrt{1 + \frac{4}{\pi}\zeta_s} + \frac{1}{2} \right\rfloor$,

Les résultats d'implémentation de ce modèle, permettent de fixer m à 30.

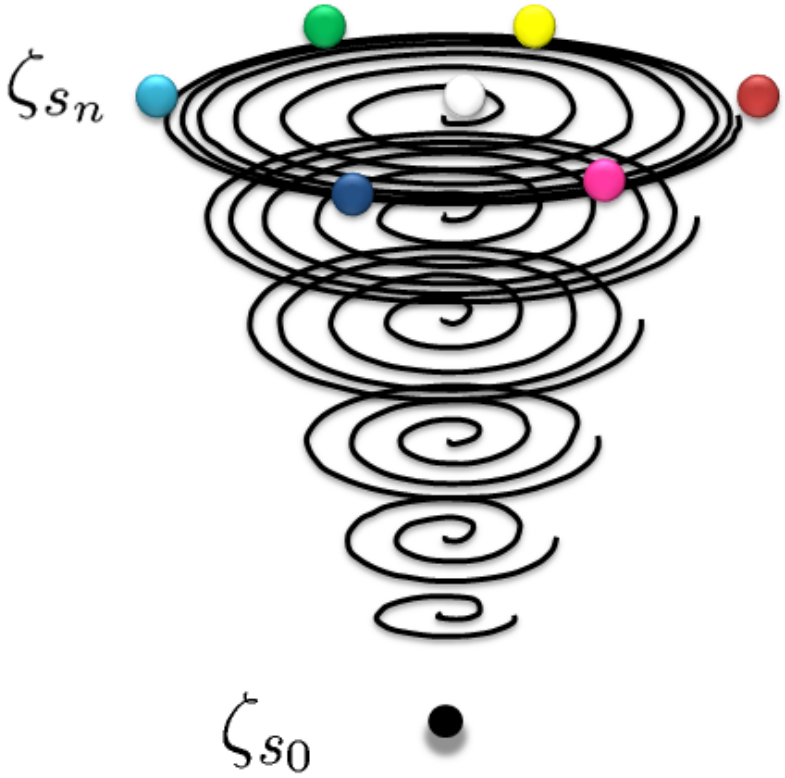


Fig. 4.3 – *Modèle CAs proposé.*

4.3 Modèle CA Utilisant une spirale logarithmique

Dans le but d'illustrer l'importance d'avoir choisi des pas décroissants entre deux tours successifs avec l'accroissement de c, on se propose d'utiliser une spirale logarithmique qui présente des caractéristiques opposées à celles de la spirale (spring curve), et donc présentant des pas croissants avec l'accroissement de c. On s'attend alors théoriquement à ce que la spirale logarithmique donne des résultats médiocres. Le rayon d'une spirale logarithmique est lié à l'angle polaire θ par la relation :

$$r(\theta) = \mathrm{a} \cdot e^{\alpha \cdot \theta}, \tag{4.11}$$

la forme géométrique d'une spirale logarithmique est illustrée par la figure(4.4).

Où a et α sont des paramètres à déterminer dans le but d'adapter la spirale au disque de chromaticité. Dans ce cas, la chroma c dépend de h suivant la relation (4.3). pour la détermination de a, on utilise le point particulier de la spirale $(c_{max}, h_{max}) = (1, 0)$ comme dans le cas de la spirale (spring curve), on obtient alors :

$$\mathrm{a} = e^{-2\pi K_C \cdot \alpha}, \tag{4.12}$$

Et dans le cas continu :

$$c(\theta) = r(\theta) = \mathrm{a} \cdot e^{\alpha \cdot \theta} \implies \theta = \frac{1}{\alpha} \cdot \ln\left(\frac{c}{\mathrm{a}}\right), \tag{4.13}$$

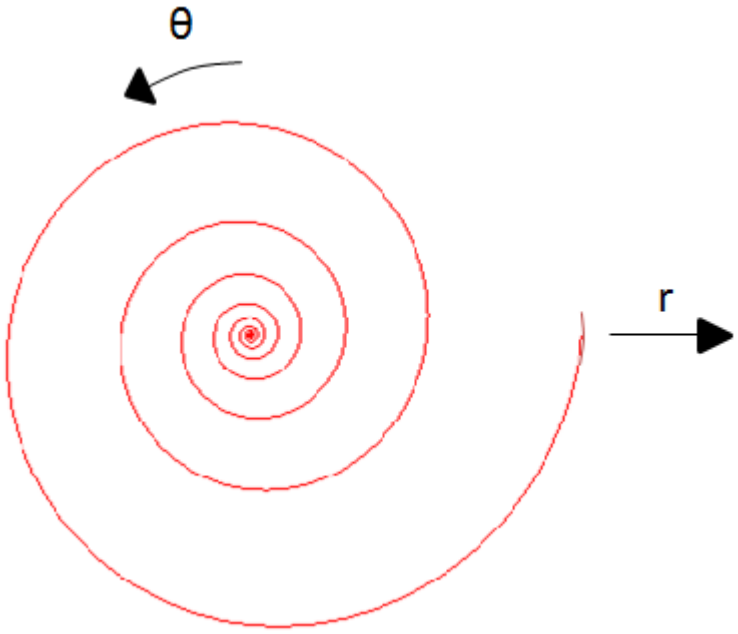


Fig. 4.4 – *Spirale logarithmique.*

La valeur optimale de α déterminée empiriquement vaut : 0.04.
On remplace (4.13) dans (4.3), on obtient ;

$$k = \text{round}\left(\frac{1}{2\pi}\left(\frac{1}{\alpha}\ln\left(\frac{c}{\text{a}}\right) - h\right)\right), \tag{4.14}$$

d'où la nouvelle expression de θ :

$$\theta = h + 2\pi.\text{round}\left(\frac{1}{2\pi}\left(\frac{1}{\alpha}\ln\left(\frac{c}{\text{a}}\right) - h\right)\right), \tag{4.15}$$

Comme précédemment, on détermine les équations de passage du modèle (CA) (utilisant une spirale logarithmique) vers l'espace HCL :

$$\begin{cases} h \equiv \zeta \mod 2\pi, \\ c = e^{-2\pi K_C \cdot \alpha}.\, e\zeta - n(n-1).\pi, \\ l = \frac{1}{K_L}\left\lfloor \frac{1}{2}\sqrt{1 + \frac{4}{\pi}\zeta} + \frac{1}{2}\right\rfloor, \end{cases} \tag{4.16}$$

avec : $n = \left\lfloor \frac{1}{2}\sqrt{1 + \frac{4}{\pi}\zeta} + \frac{1}{2}\right\rfloor$,

4.4 Modèle CA_L Proposé

4.4.1 Contexte

Notre idée principale est d'exploiter la réponse de l'oeil en fonction de la luminance selon la loi de Weber [13]. Cette loi montre que l'oeil est moins sensible aux variations pour

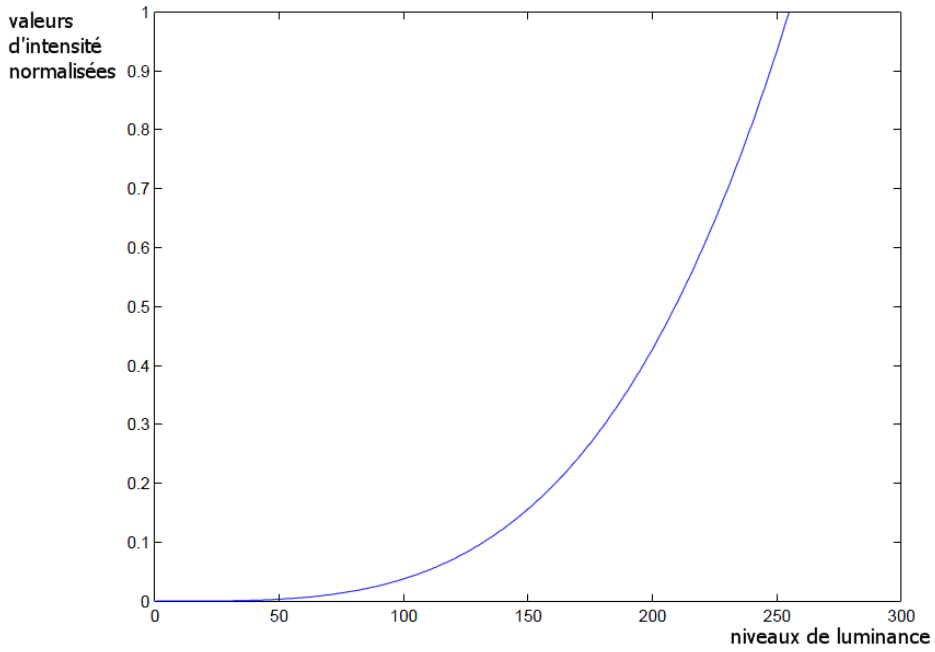


Fig. 4.5 – *Fonction d'échantilloannage de L proposée.*

les grands niveaux de luminance que pour les petits niveaux. Par conséquent la fonction d'échantillonnage qui s'adapte le mieux à la réponse logarithmique de l'oeil est la fonction exponentielle, qui attribue plus de précision aux bas niveaux qu'aux niveaux élevés . On propose dans cette section une fonction plus simple que la fonction exponontielle et ayant pratiquement la même allure pour l'ordre de grandeur des niveaux de luminance considérés. La fonction d'échantillonnage proposée est de la forme :

$$\varphi(n) = \left(\frac{n}{K_L}\right)^{\alpha}, \tag{4.17}$$

avec α un nombre réel > 1, sinon l'allure serait de forme logarithmique. Les résultats expérimentaux permettent de déterminer la valeur empirique optimale de α qui vaut 3.5. L'allure de notre fonction d'échantillonnage pour cette valeur de α est illustrée sur la figure(4.5).

L'évaluation des performances de la fonction d'échantillonnage proposée est détaillée dans le chapitre suivant.

4.4.2 Application au modèle CAs

On propose d'améliorer encore les performances du modèle CA original, en appliquant sur sa version dérivée (modèle CAs) un échantillonnage non uniforme de l'axe de luminance, on utilise pour cela la fonction d'échantillonnage proposée dans la section précédente.
Le niveau de luminance l_n aura donc comme expression :

$$l_n = \left(\frac{n}{K_L}\right)^{\alpha}, \tag{4.18}$$

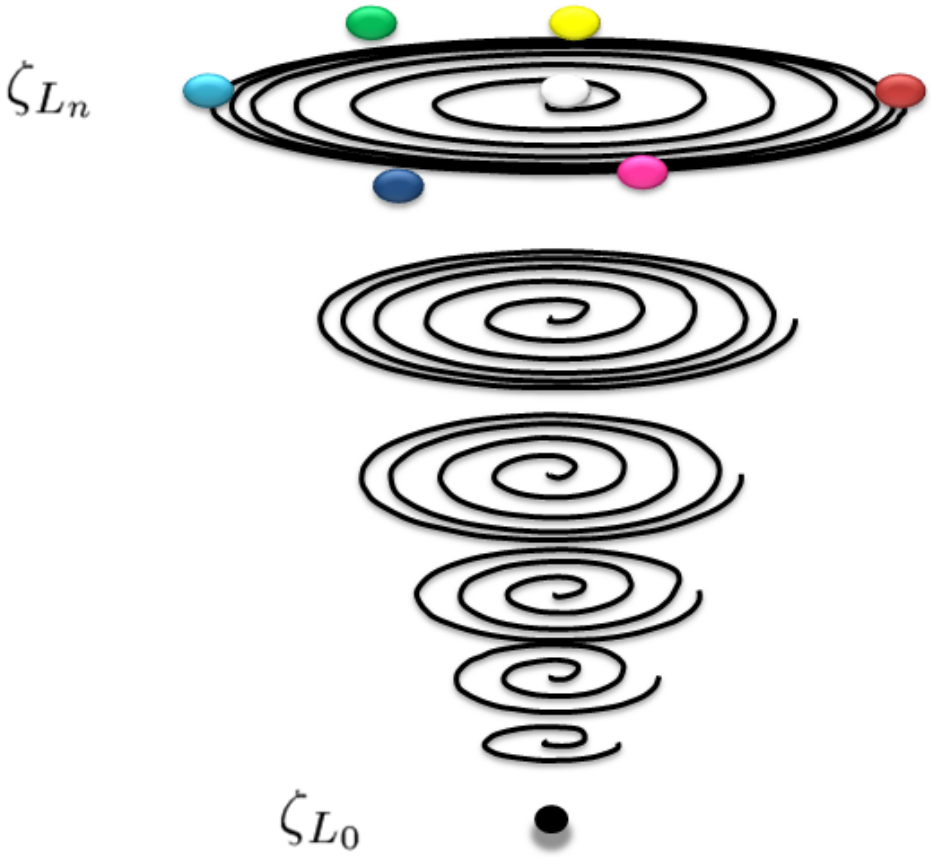


Fig. 4.6 – *Modèle CA_L proposé.*

le nouveau paramètre réduisant l'espace HCL à une dimension noté ζ_L s'écrit alors :

$$\zeta_L = \zeta_{L_{n-1}} + \theta, \tag{4.19}$$

θ étant calculé comme dans (4.8). Le modèle défini par (4.19) sera noté CA_L. En suivant les mêmes étapes illustrées dans le chapitre précédent pour l'élaboration du modèle CA, on obtient alors les équations de passage de l'espace HCL vers le modèle CA_L :

$$\begin{cases} h \equiv \zeta_L \bmod 2\pi, \\ c = \dfrac{1}{\left(1 + e^{\frac{m}{2\pi K_C}}\right) \cdot \left(1 + e^{\frac{m}{(\zeta_L - n.(n-1).\pi)}}\right)}, \\ \\ l = \left(\frac{1}{K_L}\left\lfloor \frac{1}{2}\sqrt{1 + \frac{4}{\pi}\zeta_L} + \frac{1}{2} \right\rfloor\right)^{\alpha}, \end{cases} \tag{4.20}$$

La figure (4.6) montre une représentation spatiale du modèle CA_L proposé.

Les résultats expérimentaux illustrés au chapitre suivant montrent qu'à partir de $K_L \geq 70$, les deux types d'échantillonnages uniforme et non uniforme donnent des résultats similaires. Or vu la lègère complexité en plus que présente l'échantillonnage non uniforme sur l'échantillonnage uniforme (calcul de puissance en plus), on propose de combiner les deux types d'échantillonnage comme le montre le schéma de la figure (4.7).

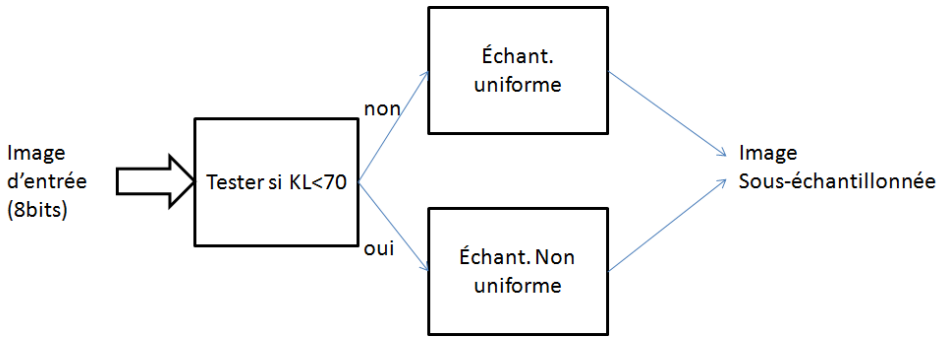


Fig. 4.7 – *Combinaison des deux formes d'échantillonnages.*

4.4.3 Applications

1) Economie de l'espace mémoire

On propose d'utiliser notre modèle CA_L pour représenter les images couleurs avant de les stocker dans une mémoire ; l'utilisateur doit donc disposer des formules de la modélisation CA_L. Ceci permet de diviser la taille des fichiers par un rapport de 3, soit un gain de 66.67% en terme de capacité de stockage. Pour les applications temps réel, l'utilisation du modèle CA_L entraîne des temps de calculs supplémentaires, on propose alors comme perspective d'implémenter le modèle CA_L proposé sur carte FPGA pour l'adapter aux applications temps réel.

3) Augmentation du débit de transmission

Dans une application où on doit transmettre des images couleurs, on propose d'utiliser la représentation CA_L pour augmenter le débit de transmission par un facteur qui vaut approximativement 3. Dans ce cas, on aura à transmettre 1 composante (paramètre CA_L) au lieu de trois composantes. Puisque la plupart des applications de transmissions nécessitent du temps réel, on propose le schéma de la figure (4.8) utilisant un circuit FPGA.

4.5 Méthode proposée pour optimiser le sous-échantillonnage des axes C et L

4.5.1 Principe

La figure (4.9) illustre le synoptique de base pour la méthode proposée.

- On utilise des images d'entrée RVB reconstruites en appliquant l'un des modèles (CA) présentés (on prend la meilleure sortie possible), donc les valeurs de K_C et K_L déterminées correspondront au modèle CA choisi.
- L'étape suivante consiste à appliquer l'algorithme de compression DCT sur ces images d'entrée, le taux de compression sera fonction de la valeur s-cielab fixée. Cette valeur de s-cielab permet à l'utilisateur de fixer le niveau de précision adapté à son application, donc les valeurs de K_C et K_L déterminées reflèteront cette pré-

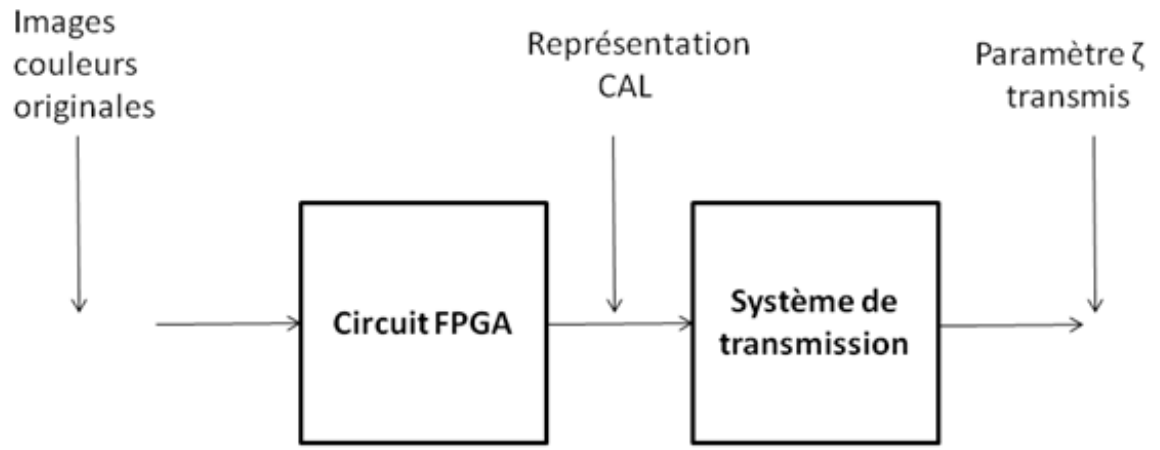


Fig. 4.8 – *Augmentation du débit en utilisant le modèle* CA_L

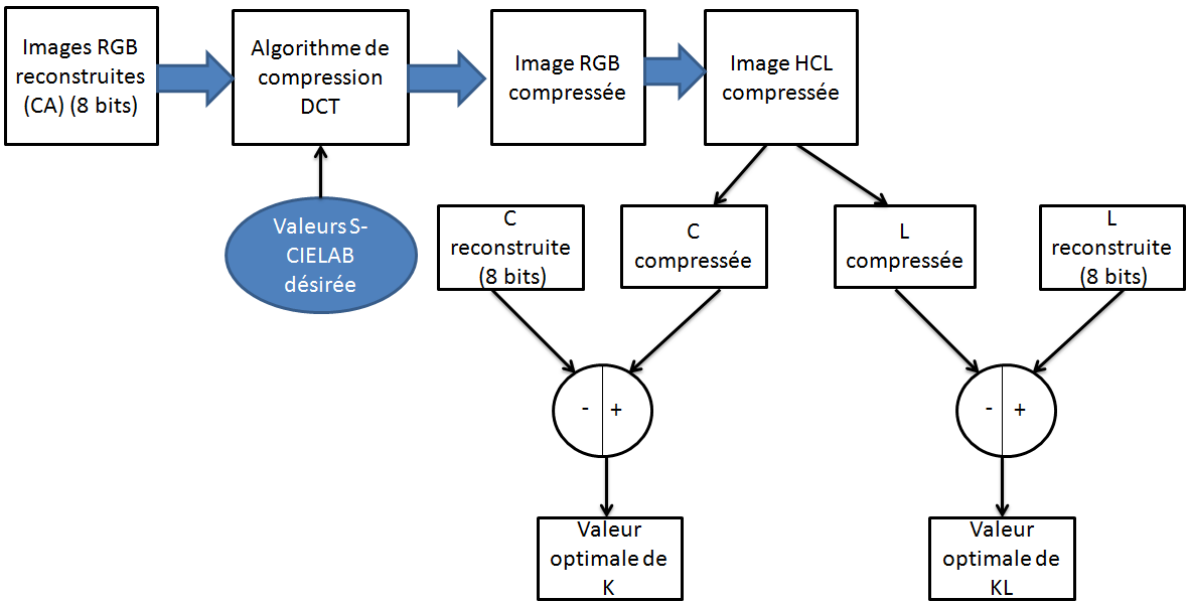


Fig. 4.9 – *Méthode proposée pour optimiser le sous-échantillonnage des axes* C *et* L.

cision.
- Les images compressées sont converties vers leurs versions HCL correspondantes. Les composantes c et l compressées sont alors extraites.
- La dernière étape consiste à faire une certaine opération de différence entre les composantes ainsi compressées et celles reconstruites (8 bits : $K_C = K_L$=255), cette opération est expliquée en détail ci-dessous. Le résultat de cette opération est les valeurs de K_C et K_L optimales pour le type d'images choisi.
- Le choix d'un algorithme de compression est justifié par le fait que l'image compressée peut être considérée comme étant une représentation sur un nombre de bits réel, et qui ne s'écrit donc pas forcément comme une puissance de 2, ceci permet d'avoir des valeurs de K_C et K_L variant dans toute la gamme [1 , 255]. L'algorithme DCT a été choisi car il permet facilement de fixer le taux de compression désiré.
- La mesure S-CIELAB a été adoptée pour son caractère universel vis-à-vis du type d'images considéré (voir chapitre 2), de telle sorte que le synoptique ci-dessus fournisse des valeurs optimales de K_C et K_L pour chaque type d'images.

L'idée principale est d'exploiter le caractère adaptatif de la mesure S-CIELAB (voir chapitre 2) vis-à-vis du type de distorsion qu'une image peut subir, ceci signifie que l'effet de distorsion du à un taux de compression réglé pour une valeur s-cielab (M unités) donnée, peut être traduit en un effet de distorsion du à un sous-échantillonnage générant la même valeur s-cielab (M unités). Pour ce faire, on fait l'analogie avec l'example de sous-échantillonnage uniforme suivant :

Exemple

Soit une image 8 bits que l'on veut représenter sur 6 bits ($K_C = K_L$=63). Pour cela, on élargit le pas de quantification par : $Q = 2^{(8-6)} = 4$, ceci signifie que :
0,1,2,3 seront représentées par 0 et 4,5,6,7 seront représentées par 4 et ainsi de suite. La nouvelle valeur du pas de quantification 4 peut être retrouvée par la formule :

$$Q = Max(I - Is) + 1, \tag{4.21}$$

où I et Is sont les images 8 bits et sous-quantifiée respectivement. Pour l'exemple précédent : $Max(I - Is) = 3 - 0 = 3 \Rightarrow Q = 3 + 1 = 4$. Et la valeur de K_C (ou K_L) correspondante est donnée par la formule suivante :

$$K = 2^{(8-\frac{\ln(Q)}{\ln 2})} - 1 \tag{4.22}$$

En suivant le même raisonnement pour le cas des images compressées, et en considérant $Max(I - Is) = Moy(I - Ic) = Qc$, avec Ic l'image compressée, et Qc le pas de quantification permettant de déterminer la valeur de K_C (ou K_L) optimale suggérée par notre algorithme. Il faut remarquer que contrairement à Q, Qc peut être un nombre réel ou entier. On obtient la valeur optimale K_{op} de K_C (ou K_L) correspondante par la formule :

$$K_{opt} = 2^{(8-\frac{\ln(Q_c)}{\ln 2})} - 1, \quad K_{opt} \in [1, 255]. \tag{4.23}$$

4.5.2 Application

Pour une application donnée, l'algorithme proposé peut être utilisé en procédant selon les étapes suivantes :

- 1-Choisir le type d'images pour l'application considérée,
- 2-Tracer la courbe moyenne correspondante ($[K_C, K_L]$=f(s-cielab)),
- 3- Fixer un niveau de distorsion acceptable pour cette application,
- 4-Tirer les valeurs optimales de K_C et K_L correspondant au niveau de distorsion considéré,
- 5- Représenter les données des images sur un nombre réel (float) de bits en utilisant les valeurs optimales de K_C et K_L retrouvées. (voir section suivante).

Des exemples d'illustration sont détaillés dans le chapitre suivant, lors de l'évaluation de la méthode proposée.

4.6 Méthode proposée pour le codage de c et l

Soient K_C et K_L deux valeurs optimales retrouvées par la méthode présentée dans la section précédente. Les nombres de bits N_C et N_L correspondant aux valeurs de K_C et K_L respectivement sont donnés par les formules suivantes :

$$\begin{cases} N_L = \frac{\ln(K_L+1)}{\ln 2}, \\ N_C = \frac{\ln(K_C+1)}{\ln 2}, \end{cases} \tag{4.24}$$

Ces valeurs optimales de K_C et K_L ne sont pas forcément des puissances de 2, par conséquent les nombres de bits correspondants ne sont pas forcément des entiers, donc les images reconstruites doivent être représentées sur un nombre réél (float) de bits pour optimiser la représentation, sinon les valeurs optimales de K_C et K_L ne seraient que théoriques.
On propose dans cette section un algorithme simple pour le codage sans perte des données des images reconstruites utilisant les valeurs de K_C et K_L optimales. On supposera que tous les niveaux des pixels possèdent la même probabilité d'apparition, puis on discutera le cas où l'équiprobabilité d'apparition n'est pas vérifiée.
Soit V une valeur optimale de K_C (ou de K_L) dont on veut représenter les données des images reconstruites, et soit N le nombre de bits correspondant, on a alors : $N = \ln(V+1)/\ln(2)$. Soient Vs l'entier le plus proche supérieur à V et Vi l'entier le plus proche inférieur à V. On utilise les fonctions de MATLAB (floor) et (ceil) pour la détermination de Vi et Vs respectivement, on a alors :

$$Vs = \text{ceil}(V), \quad Vi = \text{floor}(V), \tag{4.25}$$

L'idée principale est de représenter une certaine proportion de pixels sur Vs bits et le reste sur Vi bits. Soient Ns et Ni les proportions des données qui seront représentées sur Vs et Vi bits respectivement. Ns et Ni doivent alors vérifier $Ns + Ni = K_C$(ou K_L).
On calcule ensuite la différence en valeur absolue entre V et Vi :
$D = V - Vi$,
Si $D > 0.5$, ceci signifie que V est plus proche de Vs que de Vi, on en déduit qu'il y aura plus de niveaux à représenter sur le plus grand nombre de bits Vs que sur Vi. D représente la proportion des valeurs qui seront représentées sur le nombre le plus grand de bits Vs. Ns et Ni sont calculés comme suit :

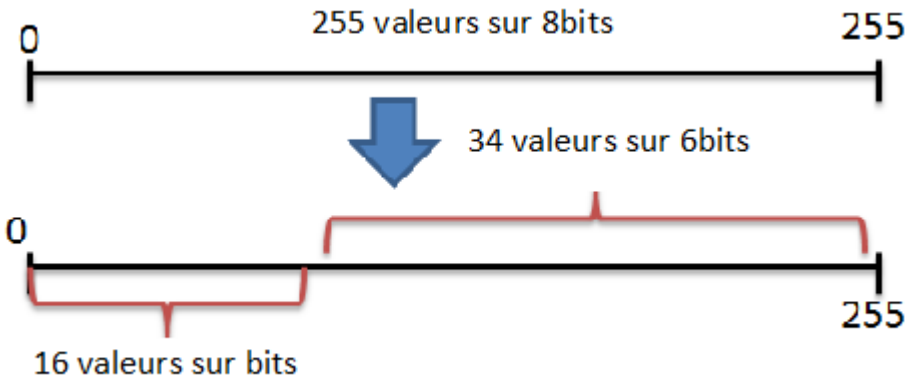


Fig. 4.10 – *Exemple de représentation sur 5.67 bits.*

$$\begin{cases} Ns = \text{round}(D.V), \\ Ni = K_C - Ns, \ \ (\text{car} \ \ Ns + Ni = K_C \ (\text{ou} \ \ K_L)), \end{cases} \tag{4.27}$$

Dans ce cas, on a : $Ns > Ni$.
Si D<0.5, ceci signifie que V est plus proche de Vi que de Vs, on en déduit alors qu'il y aura plus de niveaux à représenter sur le plus petit nombre de bits Vi que sur Vs. D représente toujours la proportion des valeurs qui seront représentées sur le nombre le plus grand de bits Vs. Ns et Ni sont calculés comme précédemment, et on a dans ce cas : $Ns < Ni$.
L'intervalle [0 , 255] contiendra K_C (ou K_L)valeurs (niveaux) et sera alors partagé en deux parties :

- Ns niveaux représentables sur Vs bits,
- Ni niveaux représentables sur Vi bits.

On peut vérifier facilement par (4.28) que cette représentation à longueur variable de bits assure que les données des images soient représentées sur le nombre réel de bits suggéré par les valeurs optimales de K_C et K_L , (ceci n'est vrai que lorsque tous les niveaux possèdent la même probabilité d'apparition, ce qui est pratiquement le cas pour certaines applications vidéos.)

$$V = \frac{Ns.Vs}{Ns + Ni} + \frac{Ni.Vi}{Ns + Ni} \ , \tag{4.28}$$

L'exemple ci-dessous illustre les étapes de l'algorithme proposé.

Exemple

$K = 50 \Rightarrow R = \frac{\ln(50+1)}{\ln 2} = 5.67$ bits , on aura alors :
$Vs = 6$bits,
$Vi = 5$bits $\Rightarrow$ $D = 5.67 - 5 = 0.67$,
N1=round(D.K)= 34 $\Rightarrow$ 34 valeurs représentées sur 6 bits,
$N2 = K - N1 = 16 \Rightarrow$ 16 valeurs représentées sur 5 bits.
La figure (4.10) illustre cette représentation à longueur variable.

il faut noter que le décodeur a besoin uniquement de la valeur de K utilisée pour retrouver les étapes de codage citées ci-dessus, et donc décoder les données des images reconstruites ainsi codées.
On note aussi que l'algorithme de codage proposé n'a pas pour objet de compresser l'information des images, mais d'optimiser la représentation en s'adaptant à l'aspect aléatoire des valeurs optimales de K_C et K_L, donc il est sans pertes. Notre représentation à longueur variable n'est donc qu'une étape intermédiaire qui pourrait servir à des fins de compression avec des taux importants.

Remarque

Si la condition sur l'équiprobabilité d'apparition des niveaux de pixels n'est pas vérifiée, alors dans ce cas il suffit qu'au moins deux niveaux de pixels parmi tous les niveaux appartiennent chacun à une région différente parmi les deux régions (parties) citées précédemment, dans ce cas on est sûre d'avoir un nombre moyen de bits inférieur à l'arrondi supérieur Vs, et plus ce cas de figure se présente plus la représentation sera optimale. Une évaluation de la méthode de codification proposée est détaillée dans le chapitre suivant.

4.7 Conclusion

On a proposé dans ce chapitre différents modèles dérivés du modèle CA original, l'idée principale était d'exploiter l'échantillonnage non uniforme des axes C et L. Ces modèles dérivés présentent de meilleurs performances (voir chapitre 5) et préservent les propriétés perceptuelles de l'espace HCL, ils pourront donc être utilisés pour des tests plus poussés sur le modèle CA, tel que l'aspect de compression du pramètre ζ_L. Les méthodes proposées pour le sous-échantillonnage et le coadge de C et L sont alors un travail a priori (représentation optimale) avant d'aborder la question de compression avec pertes.

Chapitre 5

Simulations sur MATLAB et résultats expérimentaux

5.1 But et description de la simulation

Dans ce chapitre, on va présenter les expériences ainsi que les résultats des simulations sur MATLAB qui ont été réalisées sur les modèles et méthodes proposées. Le but de ce chapitre est essentiellement la validation et la vérification des résultats théoriques. On commence d'abord par la description de la base de données utilisée (ALOI) [25], ensuite on expliquera le code MATLAB ayant permis l'implémentation du modèle CA original et de ses versions dérivées proposées. Le reste du chapitre est consacré aux expériences réalisées pour améliorer les performances du modèle CA original. On présentera d'abord une comparaison entre les performances des différentes spirales présentées dans les chapitres précédents. On abordera ensuite les expériences et résultats concernant l'échantillonnage non uniforme de l'axe de luminance, et on évaluera alors les performances du modèle CA_L proposé. Et dans la dernière partie, on proposera d'évaluer les performances des méthodes de codage et du sous-échantillonnage proposées.

5.2 Présentation de la base de données utilisée (ALOI)

La base de données ALOI (Amsterdam Library of Object Images) est une collection de mille images couleurs de petits objets, établie pour des fins scientifiques. Afin de capturer la variation sensorielle dans l'enregistrement des images, les angles de vue et d'illumination ainsi que la couleur d'illumination ont été variées systématiquement. Environs 100 images ont été prises pour chaque objet, ce qui fait un total de 110250 images pour la collection [25] Comme exemple illustratif, La figure (5.1) montre les 24 configurations pour la direction d'illumination prises pour chaque objet.

l1 à l8 représentent 8 différents angles d'illumination. c1 à c3 représentent 3 caméras permettant de varier l'angle d'illumination en tournant l'objet chaque fois vers une ca-

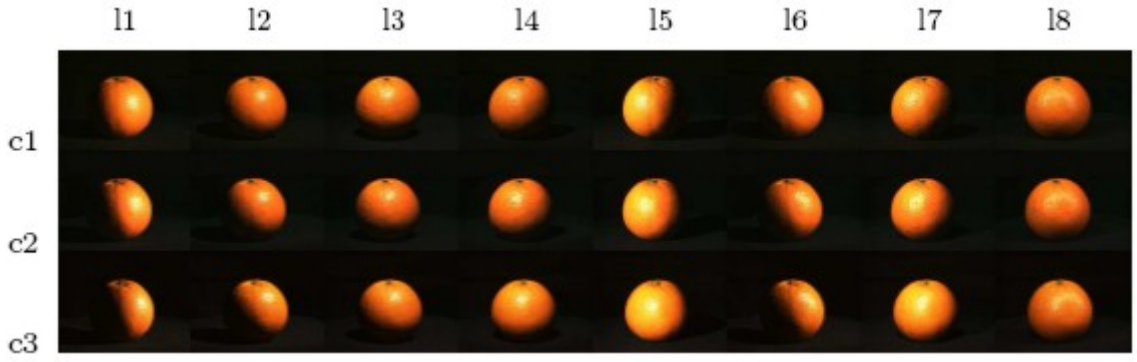


Fig. 5.1 – *Objet sous 24 angles d'illumination différents. [25]*

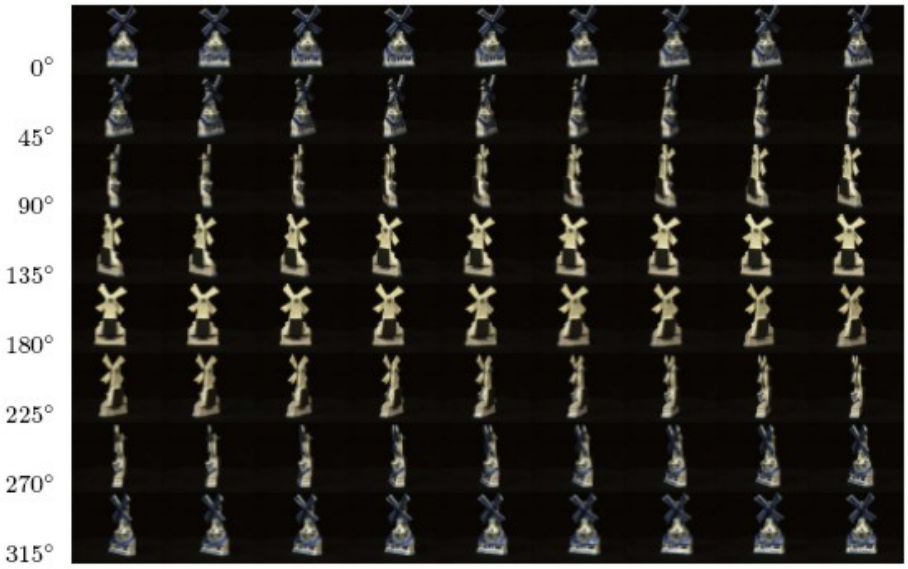


Fig. 5.2 – *Objet sous 72 positions différentes.*

méra donnée, ce qui donne finalement 24 angles d'illumination. La position de la caméra a été changée aussi en 72 positions (angles) différentes (figure(5.2)).

5.3 Implémentation du modèle CA

L'implémentation sur MATLAB du modèle CA (1D) a été faite en utilisant les différentes étapes et équations illustrées dans le chapitre sur le modèle CA. Elle a été faite d'une façon modulaire en utilisant différentes fonctions '.m' dont la fonction principale **RGB2rgb.m** faisant appel aux fonctions suivantes élaborées :

Rgb2hcl.m : cette fonction effectue la transformation de l'espace RVB vers l'espace HCL (car les images sont représentées initialement dans l'espace RVB), elle prend comme argument une image RVB et retourne en sortie l'image HCL correspondante,
Hcl2ca.m : cette fonction effectue la transformation de l'espace HCL vers le modèle CA, et ce en exploitant les étapes et équations traitées dans le chapitre sur le modèle CA,

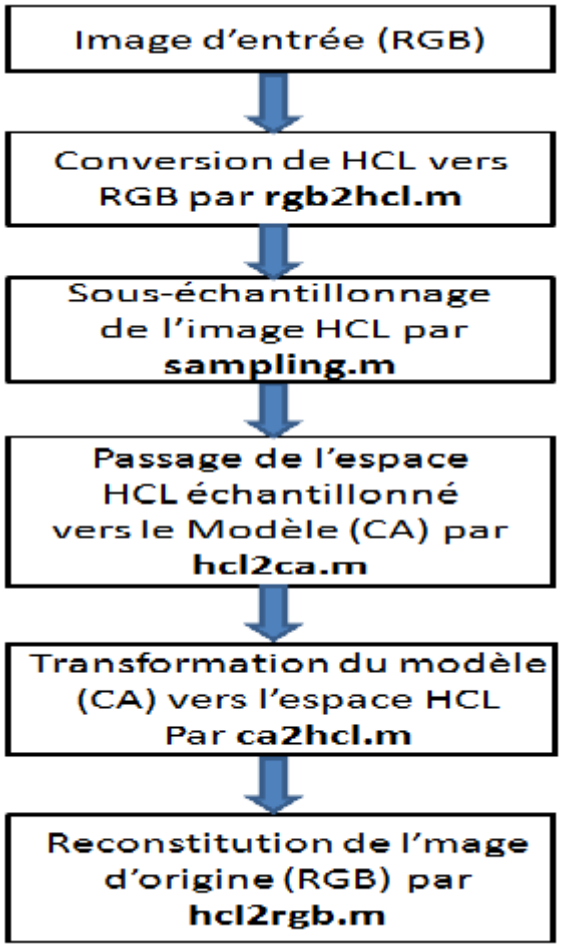


Fig. 5.3 – *Organigramme de l'implémentation du modèle CA.*

Ca2hcl.m : cette fonction permet la reconstitution du modèle HCL à partir du paramètre ζ,
Hcl2rgb.m : cette fonction permet de revenir à l'espace d'origine (RVB), à partir de l'espace HCL reconstitué,
Sampling.m : c'est la fonction qui réalise le sous-échantillonnage uniforme du cône solide HCL, ce qui permet de faire varier les paramètres K_C et K_L. Elle prend comme arguments l'image à échantillonner plus les paramètres K_C ou K_L, et renvoit en sortie l'image sous-échantillonnée correspondante.
L'exécution des fonctions citées ci-dessus se fait selon les étapes illustrées par l'organigramme de la figure(5.3).

Remarque l'implémentation des modèles CA utilisant une spirale spring curve et une spirale logarithmique respectivement a été faite de la même façon, ce qui change sont les équations de passage du modèle CA considéré vers l'espace HCL et vice-versa.

5.4 Comparaison des performances des trois spirales

5.4.1 Comparaison subjective

On propose de faire une évaluation subjective pour les différents modèles (CA) utilisant chacun une spirale différente ; pour cela on choisit des échantillons d'images de la base de données (ALOI) et on les transforme aux différents modèles CA, et ce pour différentes valeurs de K_C et K_L, puis on compare les images reconstruites pour chaque modèle. Les résultats obtenus sont montrés sur la figure (5.4).

Interprétation des résultats

L'image de la pomme (figure(5.4)) a été choisie, la première colonne à partir de la gauche corespond à l'image originale, le reste des colonnes corespondent chacune à l'une des trois spirales utilisées. (voir figure 5.4).

- La première ligne montre l'image originale et les images reconstruites pour chaque spirale, et ce pour le cas : K_C=K_L=255, ceci correspond à une représentation 8 bits (sans sous-échantillonnage), donc les images reconstruites devraient être très proches de l'image originale (a), ceci étant le cas seulement pour la spirale spring curve (modèle CAs) dont la sortie coïncide exactement avec l'image d'origine.

Pour la spirale d'Archimède , image(c), la couleur reconstruite est délavée (presque perdue) et est clairement différente de l'originale, ceci constitue l'une des faiblesses du modèle CA original, qui échoue dans le cas full 8 bits représentation.

Pour la spirale logarithmique, la couleur est tatalement absente dans l'image reconstruite. Ceci est presque le cas pour toutes les autres images reconstuites en utilisant la spirale logarithmique (sauf dans le cas : $K_C << K_L$), la couleur est alors partiellement préservée. Ceci a été prédit par l'étude théorique (voir chapitre 4), car la spirale logarithmique ne sert ici que de témoin en comparaison aux caractéristiques intéressantes que présente la spirale spring curve.

Les observations précédentes sont aussi valables pour le cas : K_L=63 et $K_C = 127$, ce qui constitue une deuxième contrainte pour la spirale d'Archimède, c'est le fait que la couleur résultante dans l'image reconstruite est délavée dans le cas :$K_C > K_L$ (voir perdue dans le cas $K_C >> K_L$), le cas d'égalité est aussi illustré par l'image (o).

Dans le cas où $K_C < K_L$, le modèle CA d'origine (utilisant une spirale d'Archimède) génère des images reconstruites très proches de l'originale (images (g) et (w)), ceci est aussi le cas pour le modèle CAs (utilisant une spirale spring curve). Ceci représente le premier avantage du modèle CAs sur le Modèle CA, c'est le fait que la condition $K_C < K_L$ n'infflue pas sur le rendu du modèle CAs, ceci constitue une contrainte en moins pour le modèle CAs. Le seul cas où la couleur est perdue partiellement par le modèle CAs est lorsque K_L=15 (image (r)), or ceci est aussi le cas pour les autres spirales. Cette valeur de K_L étant faible, ceci explique ce résultat. D'autre part, la modification apportée au modèle CAs affecte seulement la façon dont le codage de la Chroma c est effectué, la luminance étant calculée de la même façon que dans le cas du modèle CA original.

On fait juste remarquer ici que le modèle CA_L affecte le codage des deux axes C et L, donc le rendu devrait être meilleur (voir section(5.5)).

Cette évaluation subjective a montré que les performances de la spirale spring curve sont bien meilleurs que celles des deux autres spirales, et ce dans tous les cas de figures. Les

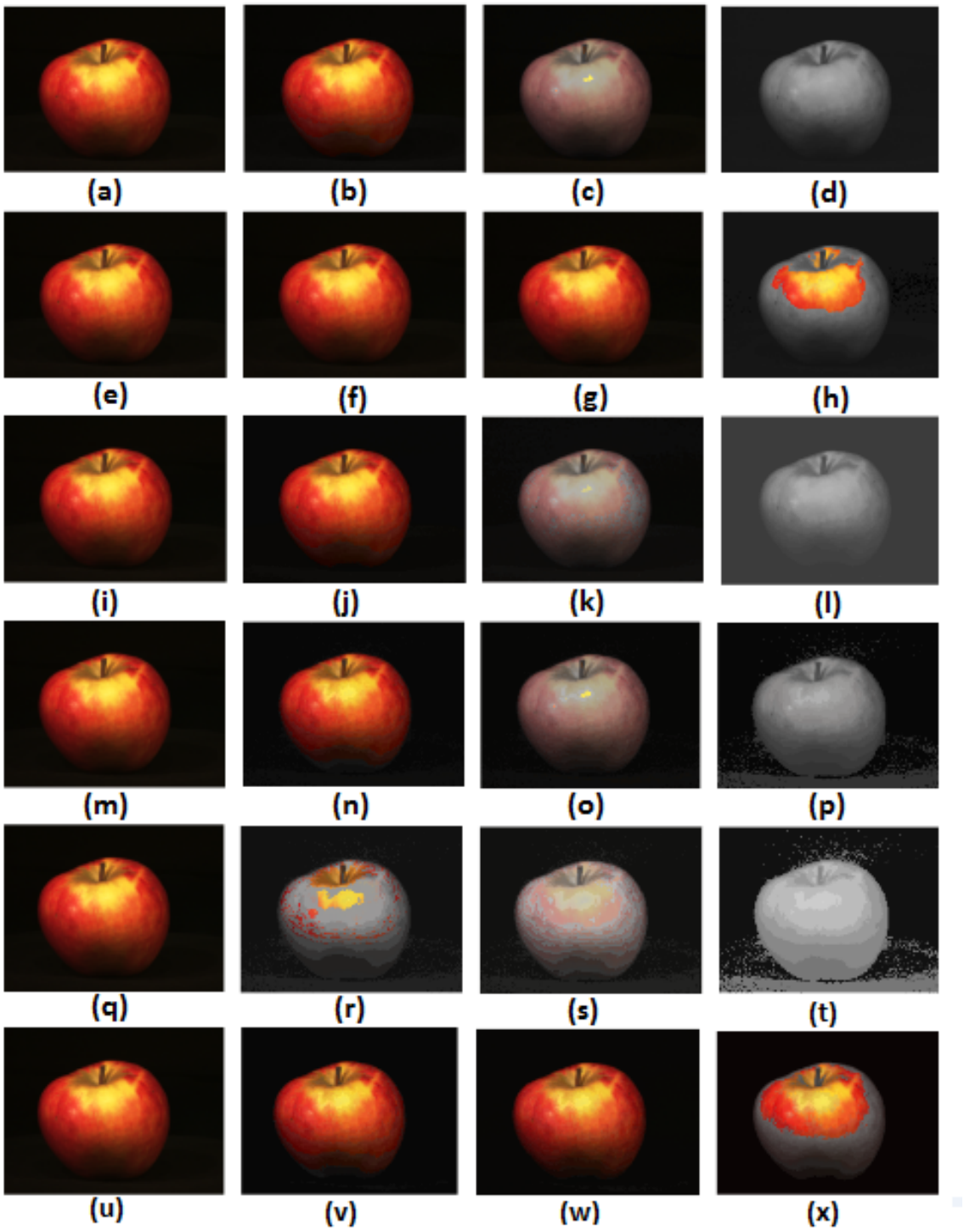


Fig. 5.4 – *Comparaison entre les performances des trois spirales pour les images reconstruites correspondantes.* 1^{re} *col. : image originale.* 2^{me} *col. : images reconstruites utilisant la spirale spring curve.* 3^{me} *col. : images reconstruites utilisant la spirale d'Archimède.* 4^{me} *col. : images reconstruites utilisant la spirale logarithmique.* 1^{re} *ligne :* $K_C = K_L = 255$. 2^{me} *ligne :* $K_C = 63, K_L = 127$. 3^{me} *ligne :* $K_C = 127, K_L = 63$. 4^{me} *ligne :* $K_C = K_L = 35$. 5^{me} *ligne :* $K_C = 35, K_L = 15$. 6^{me} *ligne :* $K_C = 15, K_L = 35$.

résultats du modèle CAs sont très proches des images originales sauf pour de petites valeurs de K_L ce qui est logique.

5.4.2 Comparaison par RMSE

On procède par évaluation globale des performances des différentes spirales utilisées pour chaque modèle CA. Pour cela on calcule l'erreur RMSE entre 100 images différentes de la base de données (ALOI) et leurs versions reconstruites à partir de chaque modèle CA, et ce pour des valeurs de K_C et K_L variant entre 1 et 255 (8bits). Les figures (5.5),(5.6) et (5.7) montrent les courbes RMSE obtenues pour une spirale d'Archimède, une spirale spring curve et une spirale logarithmique respectivement.

Interprétation des résultats
Pour la spirale d'Archimède, on remarque bien sur la figure(5.5) qu'il y a deux régions à distinguer dans le graphe RMSE correspondant :

- Pour $K_C < K_L$: l'erreur s'annule pratiquement, ce qui correspond aux résultats de l'évaluation subjective,
- Pour $K_C \geq K_L$: l'erreur maintient une valeur importante relativement constante, ce qui explique les résultats médiocres constatés lors de l'évaluation subjective lorsque $K_C \geq K_L$, ce qui fait la contrainte que la spirale d'Archimède présente.

Le graphe RMSE correspondant à la spirale logarithmique (figure(5.7)) montre que pratiquement quelque soit les valeurs de K_C et K_L l'erreur reste toujours importante, à l'exception d'une petite région correspondant aux cas ($K_C << K_L$), ce qui explique les résultats médiocres constatés lors de l'évaluation subjective. On rappelle que ce résultat sert juste comme témoin vis-à-vis des caractéristique de la spirale spring curve.

Pour la spirale spring curve, le graphe RMSE correspondant (figure(5.6)) présente une surface uniforme (erreurs négligeables) pour toutes les valeurs de K_C et K_L sauf pour des valeurs avoisinant le zéro ce qui est logique. Ce résultat important montre que la spirale spring curve ne présente pas la contrainte que la spirale d'Archimède présente, ce qui a été bien constaté aussi par l'évaluation subjective.

5.4.3 Comparaison par la mesure S-CIELAB

On propose d'illustrer les niveaux d'erreurs entre les images originales et celles reconstruites en utilisant les deux modèles CA et CAs (avec la spirale d'Archimède et la spirale spring curve respectivement). Pour cela, on utilise la mesure S-CIELAB (voir chapitre (2)) pour apprécier le degré de ressemblance entre deux pixels de l'image originale et celle reconstruite. La mesure S-CIELAB renvoie une matrice S dont la taille est égale à la taille (en nombre de pixels) des images utilisées. La valeur de chaque élément $S_{n,m}$ (appartenant à la ligne n et à la colonne m) de la matrice S représente l'erreur (mesurée en unités s-cielab) entre les deux pixels de rang m, n des deux images reconstruite et originale respectivement. On utilise 100 images de la base de données (ALOI) et on trace la distribution du nombre de pixels en fonction des valeurs s-cielab et du paramètre K_C variant de 1 jusqu'à 255. K_L étant fixé à 100 de telle façon à ce que les deux

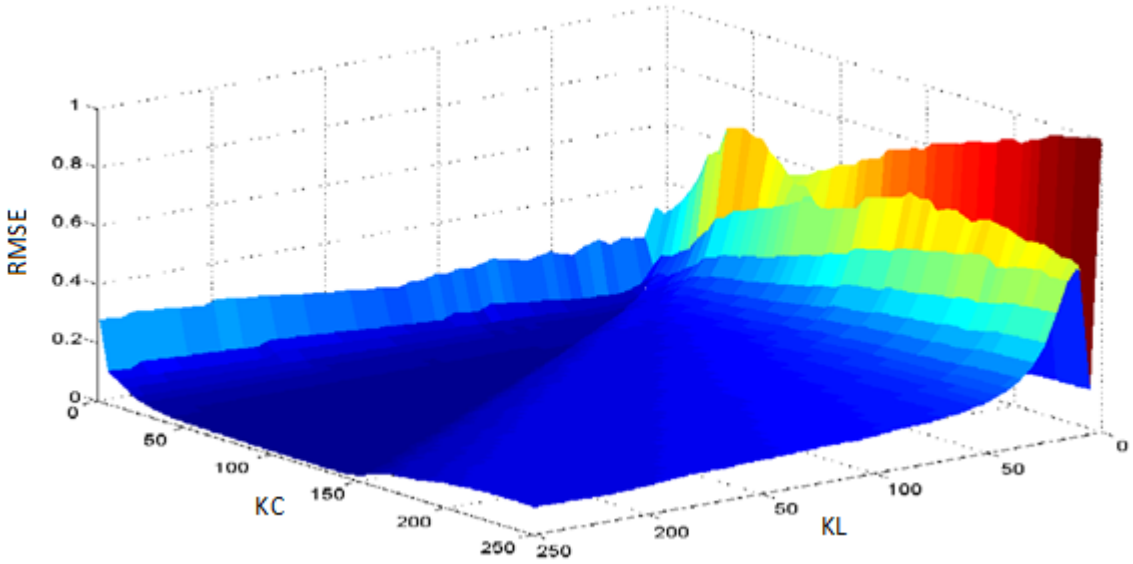


Fig. 5.5 – *Graphe RMSE=f(K_C, K_L) pour la spirale d'Archimède.*

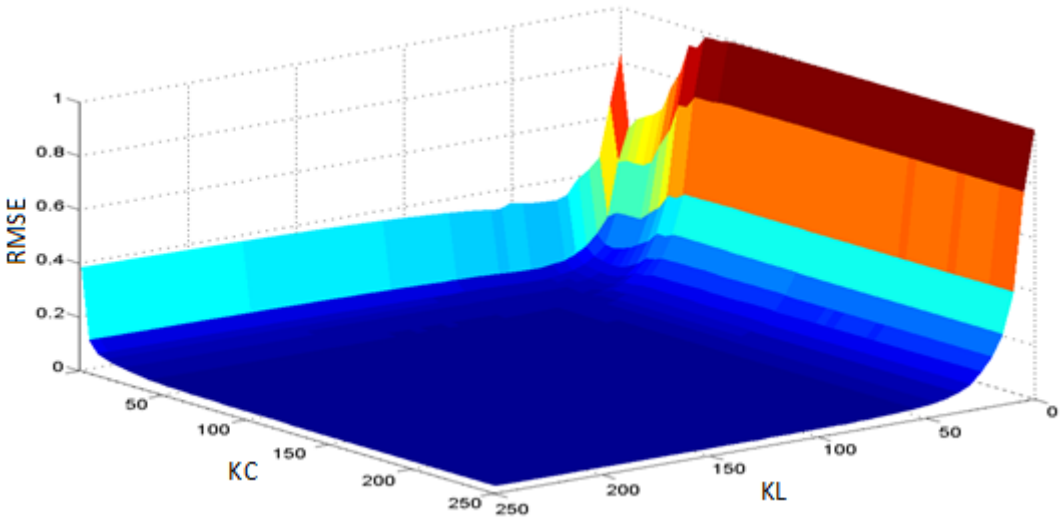


Fig. 5.6 – *Graphe RMSE=f(K_C, K_L) pour la spirale spring curve.*

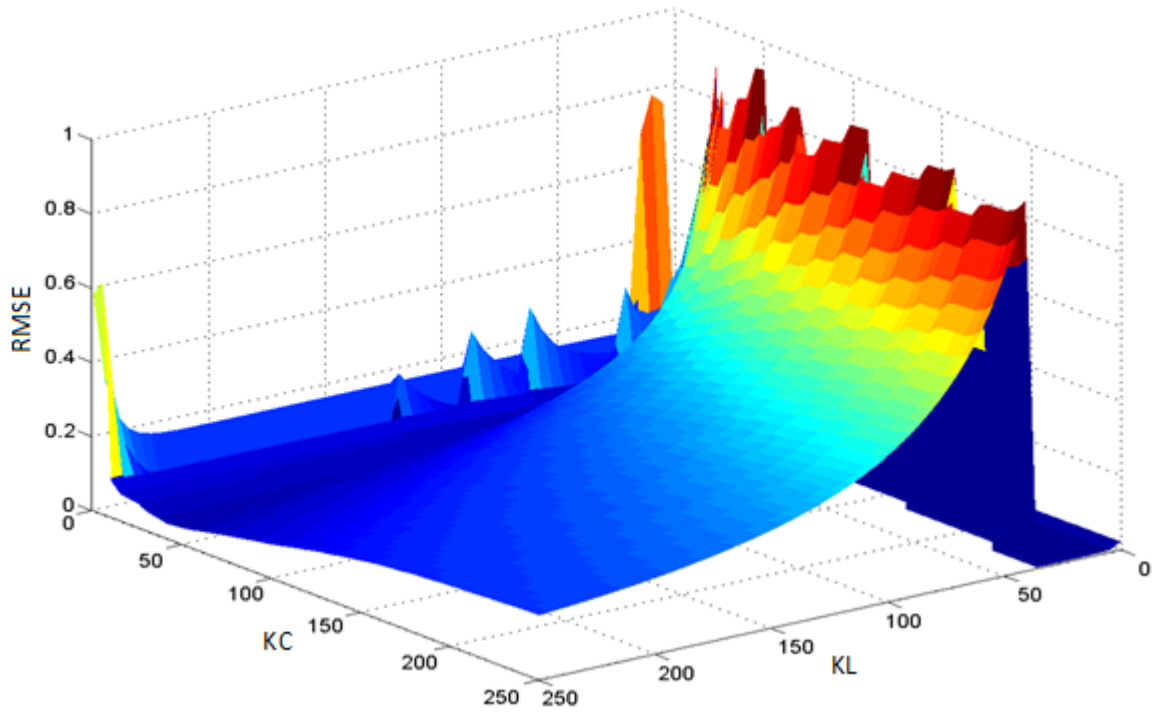


Fig. 5.7 – *Graphe RMSE=f(K_C, K_L) pour la spirale logarithmique.*

cas ($K_C < K_L$et$K_C \geq KL$) se présentent lorsque K_C varie de 1 jusqu'à 255. On fait remarquer ici que seulement K_C qui est variable pour cette évaluation, car la différence entre les spirales d'Archimède et spring curve est due à la façon dont on échantillonne le disque de chromaticité du cône solide HCL, or cet échantillonnage est réalisé suivant le paramètre K_C.
On propose aussi de tracer la distribution du nombre de pixels en fonction des valeurs s-cielab seulement, et ce pour le cas particulier où $K_C = K_L$ =255 (full 8 bits représentation figure(5.8)).

Résultats et interprétations

Les graphes obtenus pour la spirale d'Archimède et la spirale spring curve sont montrés sur les figures (5.9) et (5.10) respectivement. On constate sur la figure (5.9) que la majorité des données de l'image présentent des erreurs concentrées autour de la valeur 10 unités s-cielab, et ce pour différentes valeurs de K_C, ce qui donne une appréciation sur les niveaux d'erreurs engendrées dans le cas de la spirale d'Archimède. Pour la spirale spring curve (figure(5.10)), la majorité des données de l'image présentent des erreurs concentrées autour d'une valeur entre 5 à 6 unités s-cielab pour les différentes valeurs de K_C considérées. Ces ordres de grandeurs de la mesure S-CIELAB pour les deux spirales permettent de bien apprécier les niveaux d'erreurs qui sont clairement plus faibles dans le cas de la spirale spring curve.
La courbe obtenue pour le cas particulier ($K_C = K_L = 255$) (figure(5.8)) reflète les mêmes constats. En effet, presque 50% des données des images reconstruites présentent

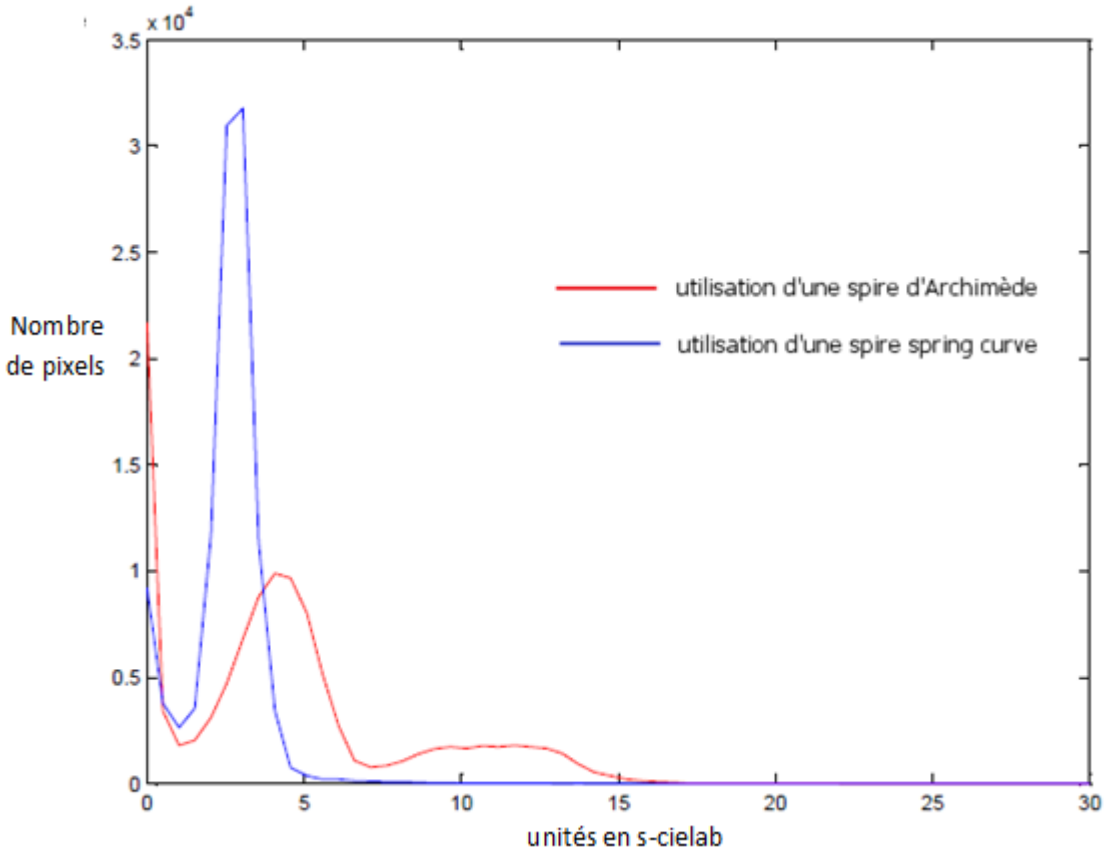


Fig. 5.8 – *Nombre de pixels en fonction des valeurs scielab pour les deux spirales d'Archimède et spring curve (pour K_C=K_L=255.*

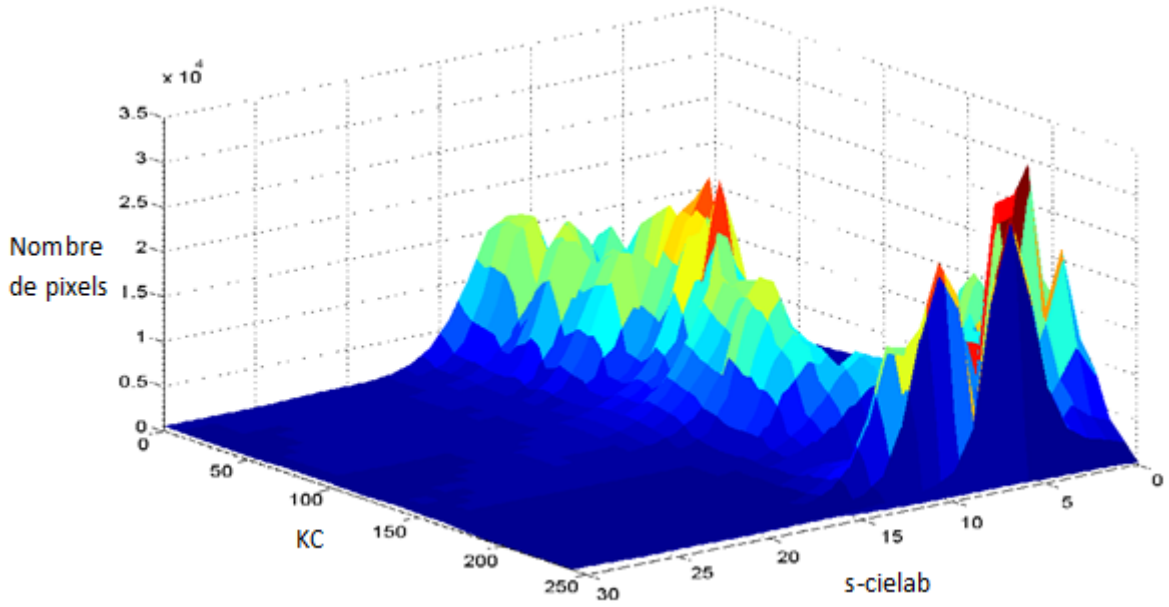


Fig. 5.9 – *Nombre de pixels en fonction des valeurs de K_C et scielab pour la spirale d'Archimède.*

des erreurs supérieures à 5 unités s-cielab dans le cas de la spirale d'Archimède. Pour la spirale spring curve, presque 100% des données des images reconstruites présentent des erreurs inférieures à 5 unités s-cielab. Ces résultats sont en corrélation avec les observations subjectives de la section (5.4.1).

Les résultats des sections (5.4.1), (5.4.2) et (5.4.3) permettent de conclure que la spirale spring curve présente deux avantages majeurs sur la spirale d'Archimède :

- D'abord la spirale spring curve ne présente pas de contraintes du genre ($K_C \geq K_L$ ou $K_C < K_L$),
- La spirale spring curve donne des résultats avec des erreurs clairement plus faibles que celles engendrées dans le cas de la spirale d'Archimède.

5.5 Evaluation du modèle CA_L (échantillonnage non uniforme de l'axe L)

5.5.1 Evaluation subjective

On utilise les modèles CA_L et CAs présentés dans le chapitre précédent. La figure (5.11) illustre des images originales et reconstruites à partir de ces modèles. Les images reconstruites ont été obtenues pour différentes combinaisons des paramètres K_C et K_L, (les images originales ont été tirées de la base de données ALOI).

Interprétation des résultats

On remarque sur la figure(5.11), que pour les valeurs moyennes et grandes de K_L, les deux modèles proposés donnent les mêmes résultats qui sont très proches de l'image ori-

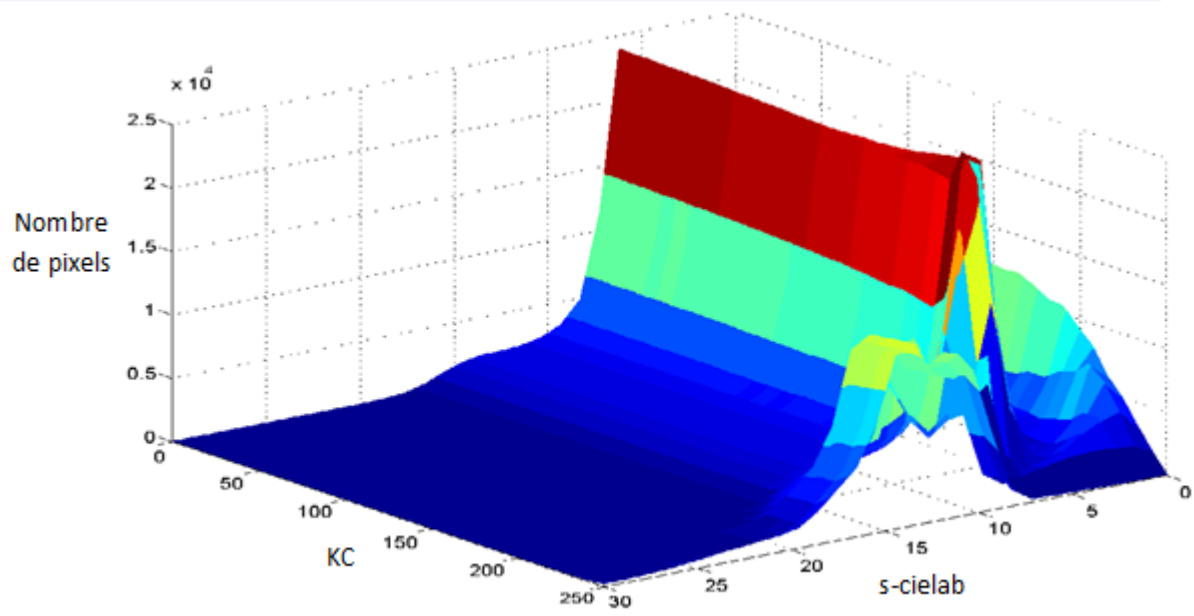


Fig. 5.10 – *Nombre de pixels en fonction des valeurs de K_C et scielab pour la spirale spring curve.*

ginale (1^{re} colonne).
Pour de petites valeurs de K_L, et plus précisémment pour les cas ($K_L = 15$, images (k) et (l)) et ($K_L = 10$, images (q) et (r)), le modèle CAs perd pratiquement l'information couleur, alors que le modèle CA_L utilisant un échantillonnage non uniforme de l'axe L préserve toujours l'information couleur originale malgré les distorsions apparaissant, ce qui est logique vu l'ordre de grandeur des valeurs K_L correspondantes.

Cette évaluation subjective du modèle CA_L montre le grand avantage que présente l'échantillonnage non uniforme de l'axe L, qui est la préservation des propriétés perceptuelles de l'espace HCL et ce, même pour de petites valeurs de K_L.
Les évaluations subjectives (présentes et précédentes) permettent de conclure que le modèle CA_L est le modèle le plus performant, car il préserve le mieux toute l'information couleur contenue dans les images originales (RVB).

5.5.2 Evaluation par la mesure S-CIELAB

On propose de procéder par une évaluation globale de notre modèle CA_L (utilisant un écht. non uniforme des axes C et L) . On le compare avec l'autre modèle proposé CAs utilisant un échantillonnage uniforme de l'axe C uniquement. En d'autres termes, il s'agit d'une comparaison objective des deux formes d'échantillonnage uniforme et non uniforme de l'axe L.
Pour cela, on trace les graphes donnant la variation du nombre de pixels en fonction des valeurs de K_L et s-cielab (K_C étant fixé à 127). les résultats sont montrés sur les figures (5.12) et (5.13).

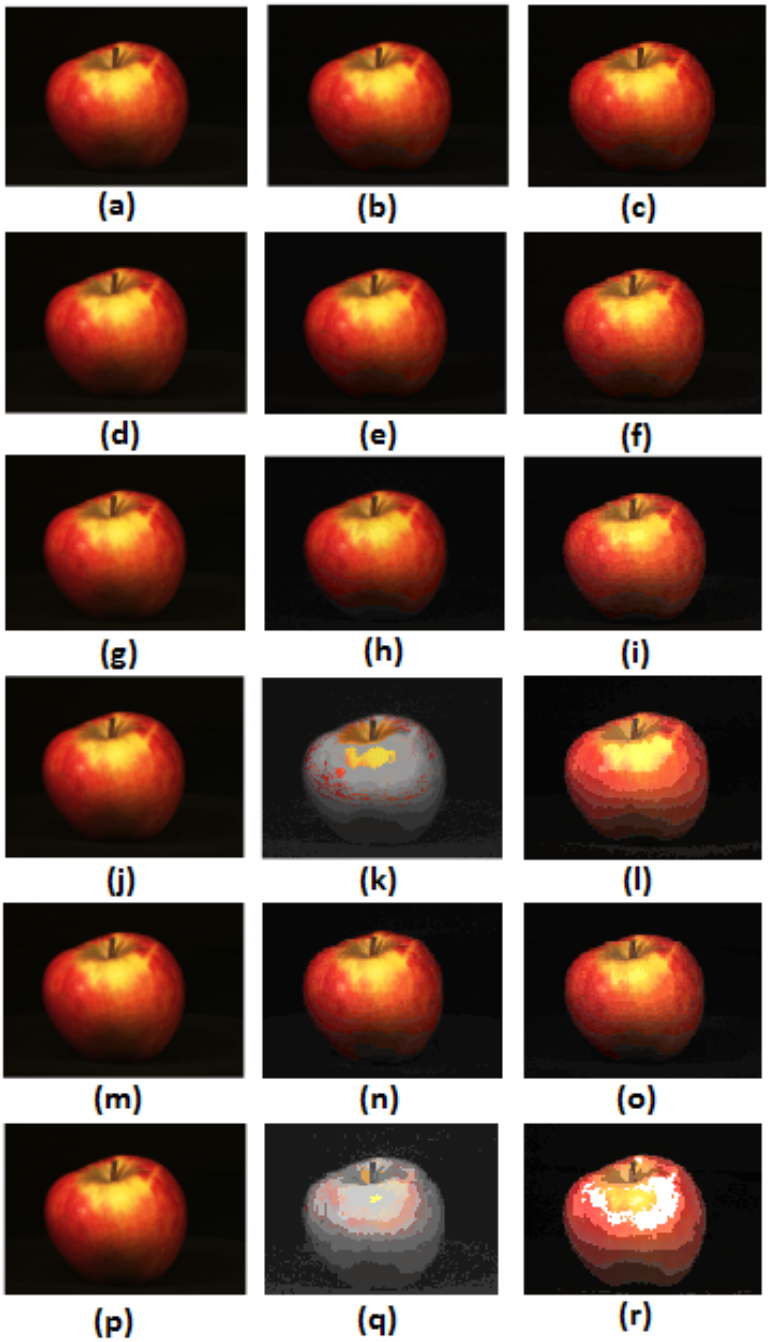


Fig. 5.11 – *Images reconstruites à partir des modèles CAs et* CA_L . 1^{re} *col. : image originale.* 2^{me} *col. : images reconstruites par le modèle CAs.* 3^{me} *col. : images reconstruites par le modèle* CA_L. 1^{re} *ligne :* $K_C = K_L = 255$. 2^{me} *ligne :* $K_C = 127, K_L = 63$. 3^{me} *ligne :* $K_C = K_L = 35$. 4^{me} *ligne :* $K_C = 35, K_L = 15$. 5^{me} *ligne :* $K_C = 15, K_L = 35$. 6^{me} *ligne :* $K_C = 10, K_L = 35$.

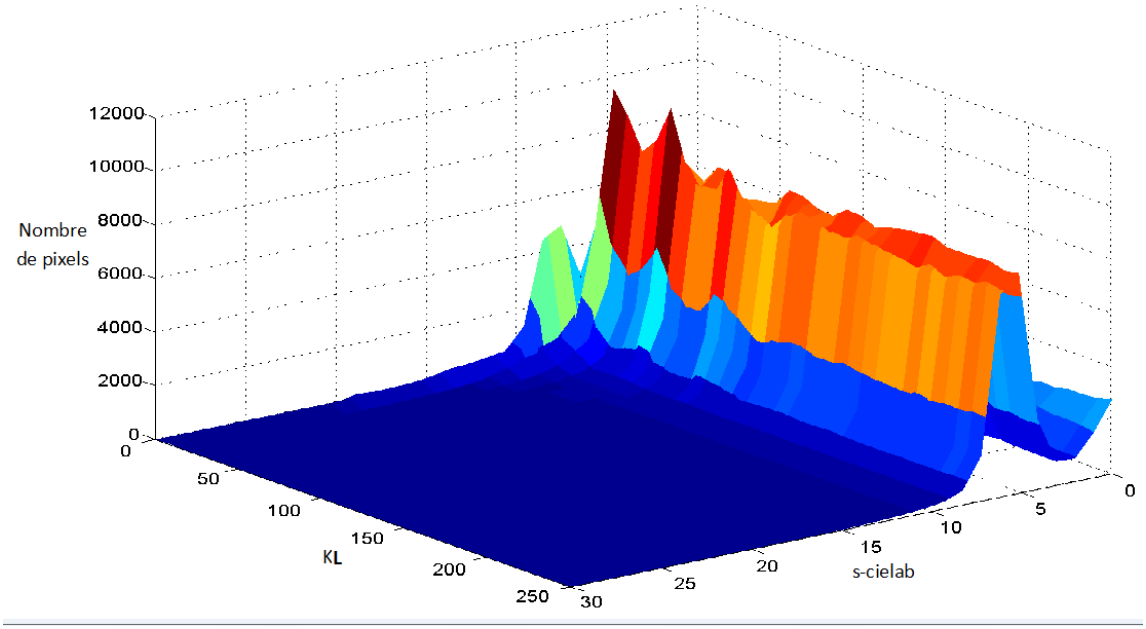


Fig. 5.12 – *Nombre de pixels en fonction des valeurs de K_L et scielab pour le modèle CA_L.*

Interprétation des résultats
On constate que pour des valeurs de K_L >80 approximativement, les deux formes d'échantillonnages présentent les mêmes distributions, ce qui signifie des niveaux d'erreurs pratiquement similaires pour ces valeurs de K_L, ce qui confirme les résultats de l'évaluation subjective.
En revanche, on remarque que pour des valeurs de $K_L < 80$, le graphe correspondant au cas d'échantillonnage uniforme présente des endroits dont les valeurs s-cielab dépassent les 10 unités, et plus K_L diminue plus les valeurs de s-cielab deviennent plus importantes. Ce constat n'est pas observé dans le cas d'échantillonnage non uniforme (figure(5.12)).
Ces résultats objectifs, montrent que l'échantilonnage non uniforme présente des niveaux d'erreurs nettement inférieurs pour des valeurs de K_L <80, mais ce chiffre peut varier selon le type d'images considérées.

On trace aussi les courbes donnant la variation du nombre de pixels en fonction des valeurs s-cielab uniquement, et ce pour quelques valeurs particulières de K_L (figure 5.14). On constate alors que l'allure de ces courbes reflète les résultats de l'évaluation globale suivant les valeurs de K_L.

5.6 Evaluation de la méthode proposée pour le sous-échantillonnage des axes C et L

On prend des échantillons d'images de la base de données (ALOI), et on en trace la courbe moyenne donnant les variations des valeurs optimales K_C et K_L (trouvées par

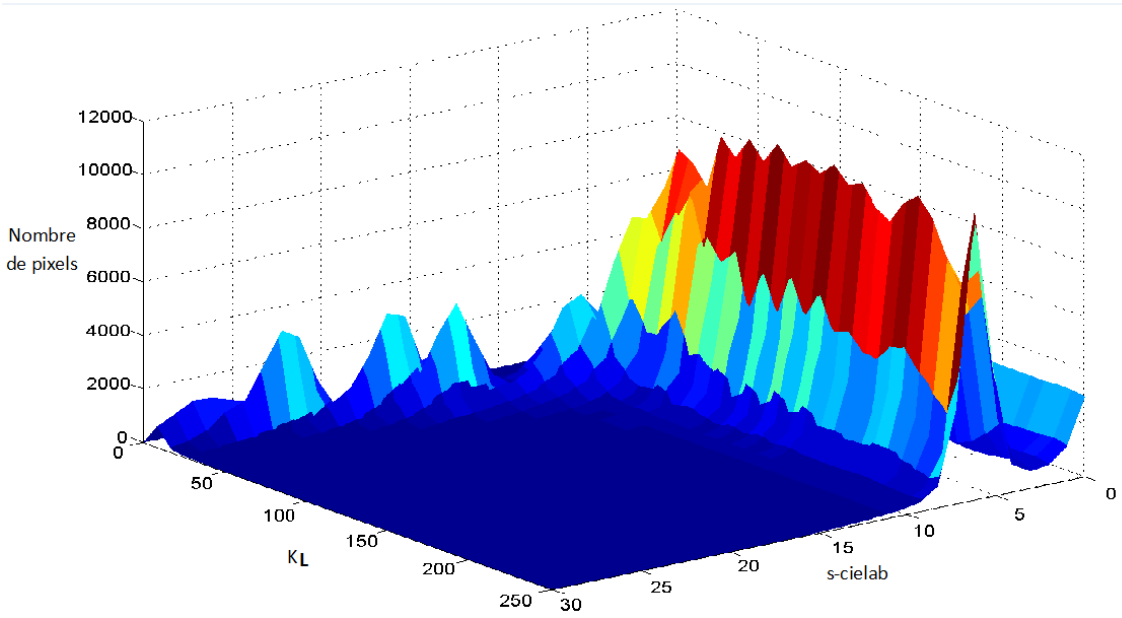


Fig. 5.13 – *Nombre de pixels en fonction des valeurs de K_L et s-cielab pour le modèle CAs.*

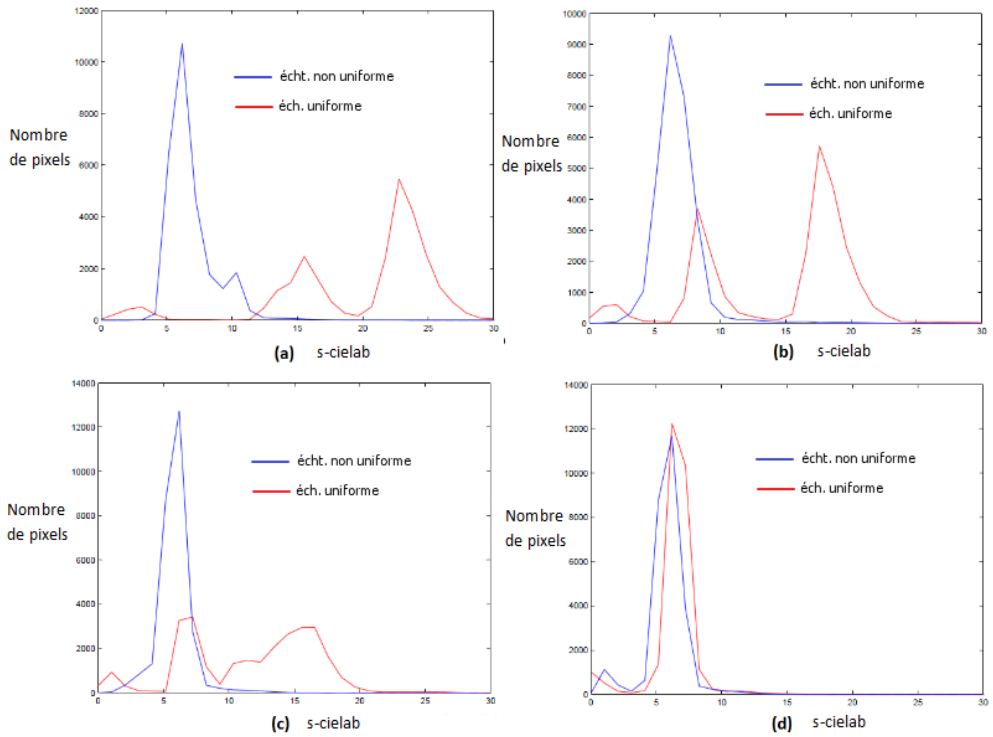


Fig. 5.14 – *Nombre de pixels en fonction des valeurs s-cielab. (a) : pour $K_L = 20$. (b) pour K_L=30. (c) pour $K_L = 40$. (d) pour $K_L = 100$.*

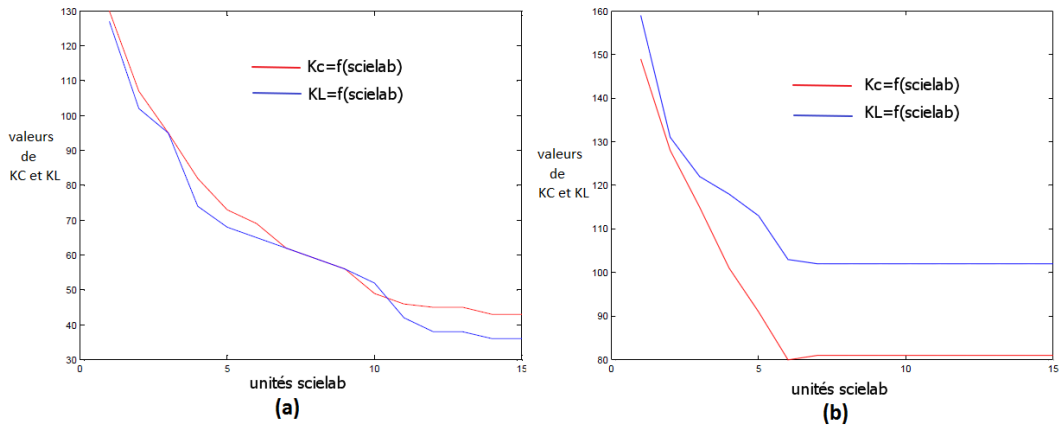


Fig. 5.15 – *Valeurs optimales K_C et K_L en fonction des valeurs scielab. (a) pour des images naturelles. (b) pour des images sombres.*

notre algorithme) en fonction des valeurs s-cielab. On trace cette courbe pour deux types d'images :

– Images naturelles (figure 5.14(a)),

– Images sombres (figure 5.14(b)).

On constate que pour chaque type d'images, la courbe correspondante possède une allure bien particulière, ceci signifie que les valeurs optimales de K_C et K_L dépendent du type d'images considéré.

Pour tester ces résultats, on prend deux images appartenant chacune à l'un des deux types d'images mentionnées précédemment, et on obtient leurs versions reconstruites à partir du modèle CA_L. On choisit deux valeurs scielab qui décideront des niveaux de dégradations :

- scielab= 5 unités : (c'est l'ordre de grandeur des erreurs engendrées par le modèle CAs, et qui sont acceptables),
- scielab=10 unités, (c'est l'ordre de grandeur des erreurs engendrées par le modèle CA, et qui sont plus importantes que dans le cas CAs).

La figure (5.16) illustre les résultats obtenus.

On remarque que pour le cas (s-cielab=10 unités), les images reconstruites obtenues présentent des artéfacts dus au sous-échantillonnage (images (b) et (e) sur la figure (5.16)), alors que pour le cas (s-cielab=5 unités), ces artéfacts disparaissent (images (c) et (e) sur la figure (5.16)), ce qui signifie que les courbes obtenues donnent des valeurs optimales ($K_C = K_L = 75$) pour les images naturelles (figure 5.15(a)), et des valeurs (KC=100, etK_L=130) pour les images sombres (figure 5.15(b)).
Ces résultats permettent d'avoir une première évaluation de la méthode proposée, on propose dans la section suivante d'évaluer notre méthode de codage proposée pour ces valeurs optimales de K_C et K_L.

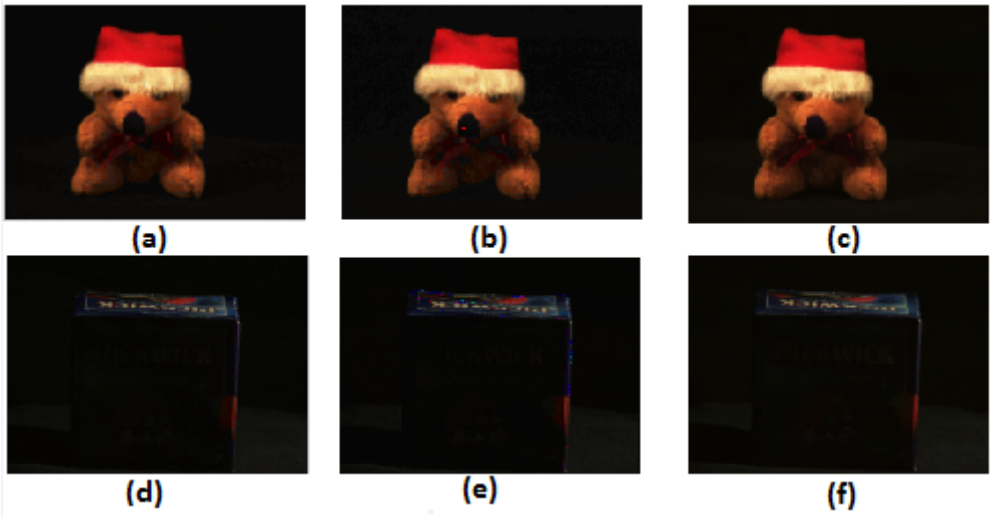


Fig. 5.16 – *Images reconstruites à partir du modèle CA_L . 1^{re} col. : images originales. 2^{me} col. : images reconstruites pour scielab=10 unités. 3^{me} col. : images reconstruites pour scielab=5 unités. 1^{re} ligne : images naturelles. 2^{me} ligne : images sombres.*

5.7 Evaluation de la méthode de codage à longueur variable

Pour tester notre algorithme de codage dans le cas où la condition d'équiprobabilité n'est pas vérifiée, on se propose de prendre des échantillons d'images de la base de données ALOI (50 images) et d'en tracer la courbe moyenne donnant le nombre de bits moyen résultant de notre algorithme en fonction des valeurs optimales de K_C et K_L. La figure (5.17) montre les résultats obtenus (l'allure de la courbe est la même pour C et L). On remarque que par exemple pour $K_{opt} \in [65, 100]$, l'algorithme de codage proposé permet une présentation sur un nombre de bits moyen $<$ 7bits. Or, si on avait représenté les données des images sur une longueur constante, toutes ces données auraient été représentées carrément sur 7 bits, ce qui enlève l'efficacité des valeurs K_C et K_L optimales. Ces résultats montrent que l'algorithme de codage proposé peut être utilisé pour optimiser la représentation et ce, même dans le cas où la condition d'équiprobabilité n'est pas vérifiée.

D'autre part, on propose de tester l'efficacité de notre algorithme pour des images satisfaisant pratiquement la condition sur l'équiprobabilité. la figure(5.18) montre une image vérifiant ce critère. On trace la courbe (Nbre de bits =f(K_{opt})) pour cette image, les résultats obtenus sont montrés sur la figure (5.19).

On remarque bien que le résultat obtenu par notre algorithme de codage est plus intéressant dans ce cas. Par exemple, pour le même intervalle de K_{opt} ([65,100]), les données sont représentées en grande partie sur 6 bits et le reste sur un nombre $<$ 6.5 bits. Or, $2^{6.5} \approx 90$, ce qui est bien reflété par la courbe de la figure(5.19), car on ne devrait pas atteindre 6.5 bits, pour K_{opt}<90.

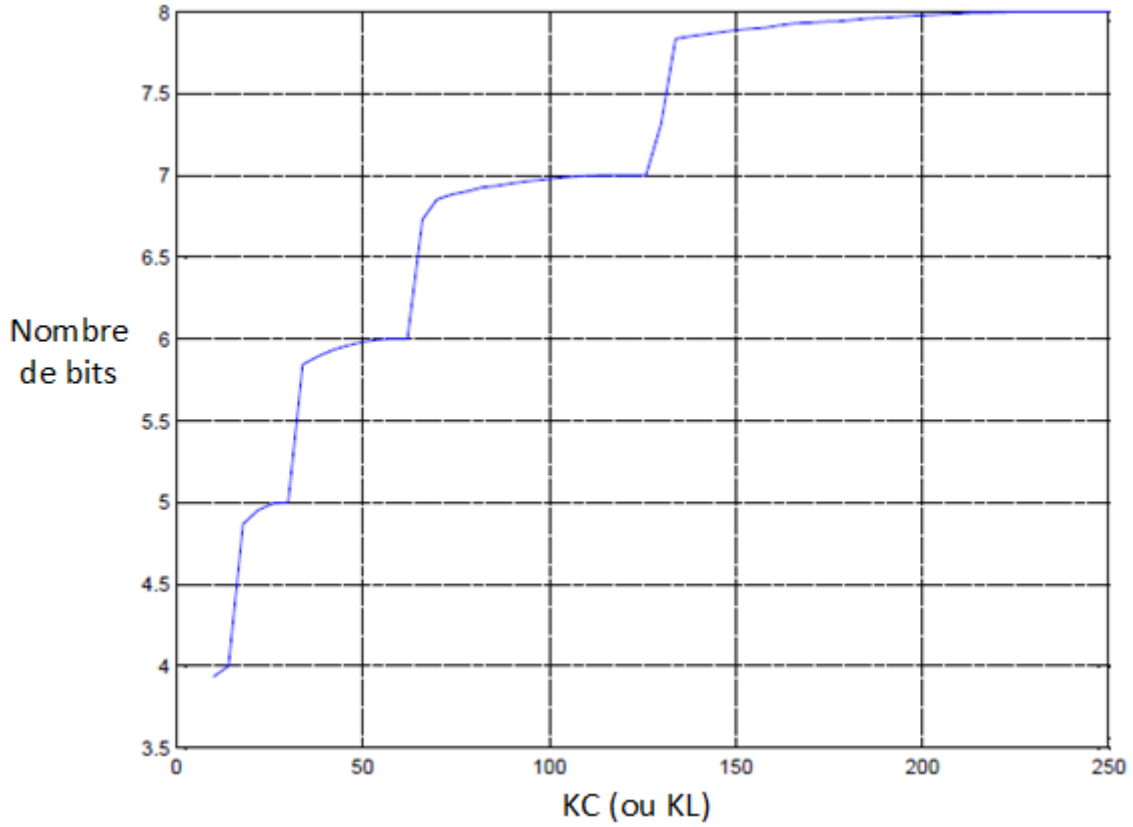


Fig. 5.17 – *Evaluation de l'algorithme de codage proposé pour des images ne satisfaisant pas la condition d'équiprobabilité.*

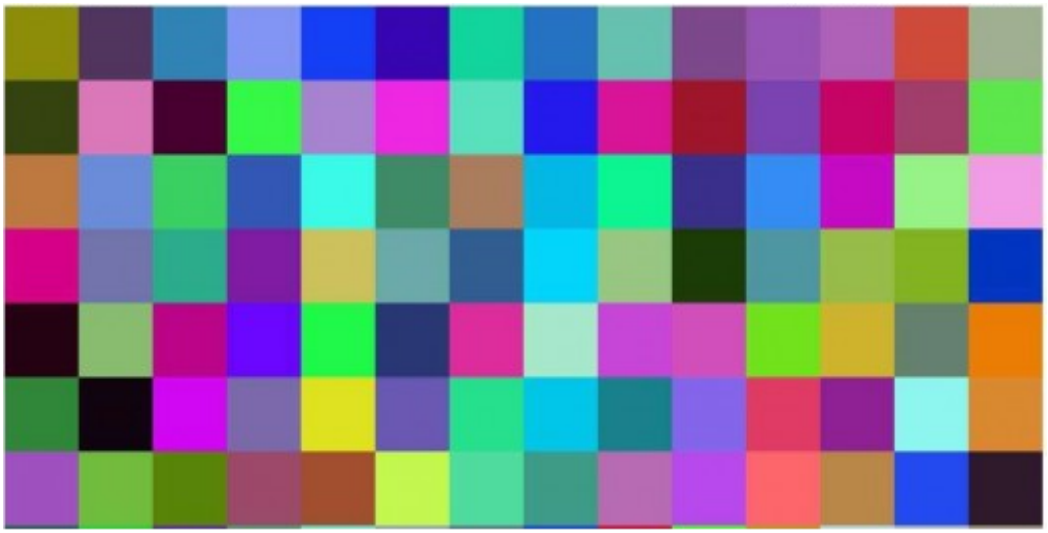


Fig. 5.18 – *Exemple d'une image satisfaisant la condition d'équiprobabilité.*

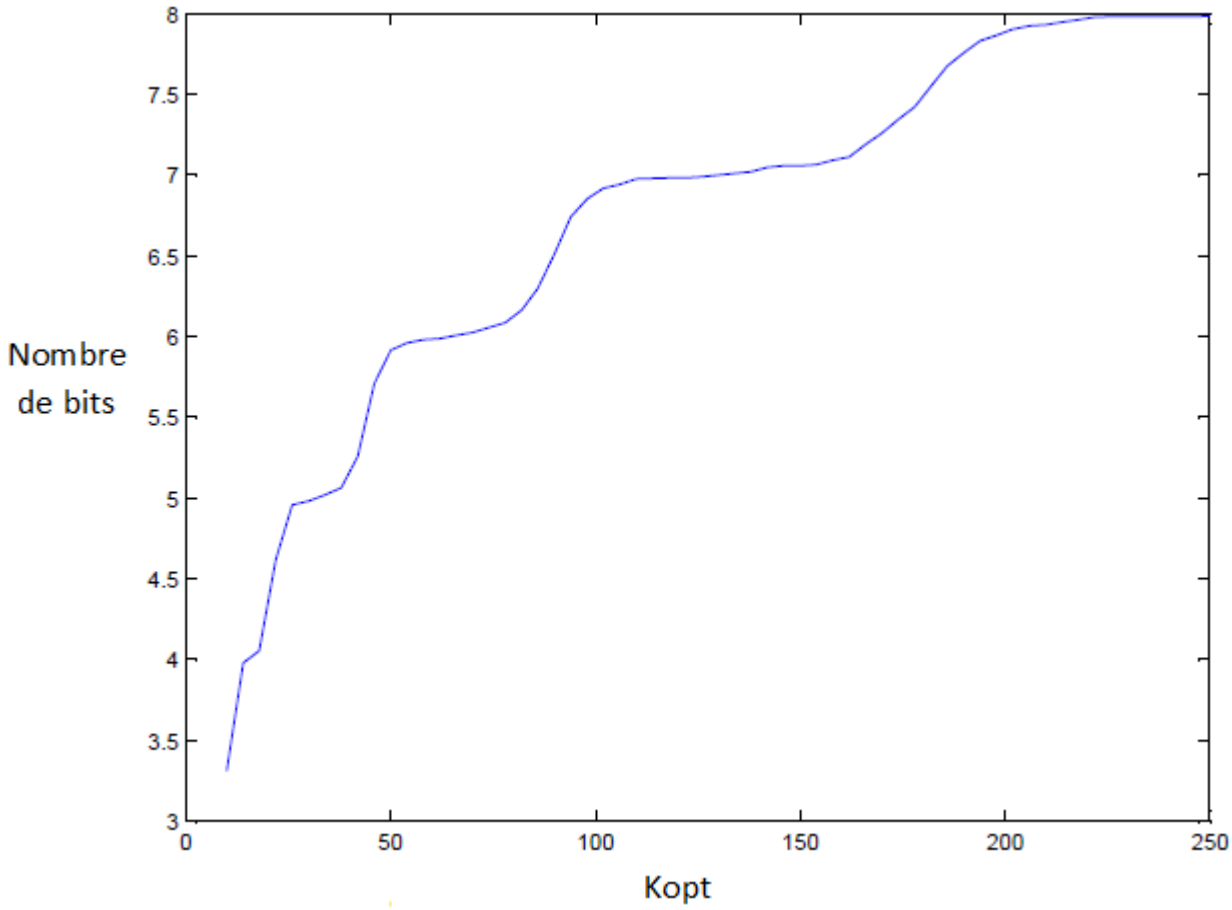


Fig. 5.19 – *Evaluation de l'algorithme de codage proposé pour des images satisfaisant la condition d'équiprobabilité.*

On fait remarquer que notre premier test (condition d'équiprobabilité non vérifiée) a été effectué sur des images de la base de données ALOI, or ces images sont des images d'objets ne présentant pas une large gamme de variation des couleurs pour une image donnée, donc on s'attend à ce que les résultats obtenus pour ces images soient meilleurs pour d'autres types d'images.

5.8 Conclusion

On a présenté dans ce chapitre les expériences et résultats des simulations réalisées dans notre PFE. Ces résultats expérimentaux ont permis de déterminer le modèle le plus performant parmi les modèles proposés (y compris le modèle CA original), il s'agit du modèle CA_L. Ce modèle présente des propriétés très intéressantes, en exploitant l'échantillonnage non uniforme des axes de chrominance C et de luminance L, ce qui permet de mieux préserver les propriétés perceptuelles du modèle HCL. L'évaluation des méthodes proposées pour le codage et le sous-échantillonnage des axes C et L a permis d'avoir une première estimation sur les niveaux de compression sans pertes pour le modèle CA_L, ce qui pourrait être une étape intermédiaire avant d'aborder la question de compression avec pertes.

Conclusion générale et perspectives

L'objectif principal de notre travail de PFE a été l'amélioration des performances du modèle CA original à une dimension. On a proposé pour cela d'autres modèles dérivés du modèle CA 1D. L'idée principale était d'adapter un échantillonnage non uniforme aux axes de chrominance C et de luminance L. Ce qui a abouti finalement au modèle CA_L qui présente des résultats nettement plus performants que ceux du modèle CA.
L'élaboration du modèle CA_L a été faite en deux étapes, d'abord on s'est intéressé à l'échantillonnage non uniforme de l'axe de chrominance C, pour cela on a proposé une spirale de type spring curve adaptée au disque de chromaticité, ce qui a donné le modèle dérivé CAs, pour lequel l'information de chrominance est mieux conservée. La deuxième étape consiste à appliquer un échantillonnage non uniforme à l'axe de luminance L, la combinaison de cette forme d'échantillonnage non linéaire avec le modèle CAs nous a permis d'élaborer le modèle CA_L.
L'analyse des résultats obtenus par simulations sur MATLAB, montre que le modèle CA_L préserve beaucoup mieux l'information couleur contenue dans l'espace couleur HCL.

Ce travail de PFE représente une étape importante, qui prépare aux prochains travaux, comme la définition d'une meilleure distance moins complexe pour le modèle CA_L, ainsi que d'aborder le problème de compression avec pertes du paramètre ζ_L. D'autre part, on propose comme perspectives d'implémenter d'abord sur carte FPGA le modèle CA_L, puis l'appliquer pour augmenter le débit de transmission et aussi économiser l'espace de stockage des données images en couelur.

Références

[1] Frederic GARCIA, Djamila AOUADA, Bruno MIRBACH, Bjorn OTTERSTEN. A new 1-d colour model and its application to image filtering. 7th International Symposium on Image and Signal Processing and Analysis (ISPA 2011). september 2011.

[2] Erwan LE PENNEC. Compression d'image. Academic Press, 2nd ed. october 2006.

[3] R. C. GONZALEZ and R. E. Woods. *Digital image processin*, 2nd ed. Prentice Hall. 2002.

[4] Florence TUPIN, Isabelle BLOCH. Traitement et reconnaissance d'images. 2009.

[5] BOULFANI Yasmine, DOUMANDJI Samah. Implémentation sur dsp tms320c5000 de filtres optimaux appliqués aux images et introduction de réseaux neuronaux. PFE, ENP, 2004.

[6] AISSA BRAHIM, Salim KADDOUR Chakib. Généralités sur le traitement d'images. http: //www.kaddour.com/chap1/chap1.htm.

[7] Sébastien THON. Imagerie numérique, représentation et codage des images, codage et compression des images. 2010.

[8] Robert M. GRAY. *Entropy and Information Theory*. Springer 2nd ed. Department of Electrical Engineering Stanford University. july 2009.

[9] Nicolas VANDENBROUCKE, Laurent BUSIN and Ludovic MACAIRE. Color spaces and image segmentation. Edition JCR Science. 2007.

[10] C. TERRIER. *Les couleurs. http://www.*cterrier.*com*

[11] http ://www.dede75007.com/alacouleur/14/14alacouleur.htm.

[12] Rokia MISSAOUI M, SARIFUDDIN. A new perceptually uniform color space with associated color similarity measure for contentbased image and video retrieval. 2005.

[13] Hans-W ERNER SINN. Weber's law and the biological evolution of risk. september 2002.

[14] Frederic GARCIA, Djamila AOUADA, Bruno MIRBACH, Bjorn OTTERSTEN. Spiral colour model : Reduction from 3D to 2D. International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2010.

[15] http ://en.wikipedia.org/wiki/Mean_squared_error

[16] Zhou WANG, Member IEEE, Alan Conrad BOVIK, Fellow IEEE, Hamid Rahim SHEIKH, Student Member IEEE and Eero P. SIMONCELLI, Senior Member IEEE. Image quality assessment : From error visibility to structural similarity. 2002.

[17] Bruce MACEVOY. *Color theory*. 2004.

[18] Daniel METZ. Espaces colorimétriques. Art 2nd *Edition.* décembre 2005.

[19] Xuemei ZHANG and Brian A. WANDELL. Spatial extension of cielab for digital color image reproduction. Department of Psychology, Stanford University Stanford, CA 94305 .1996.

[20] M. BARTKOWIAK and M. DOMANSKI. Scalar chrominance and its applications in color representation. 2001.

[21] M. BARTKOWIAK and M. DOMANSKI. Color video compression based on chrominance vector quantization. 1996.

[22] Arash VAHDAT and Mark S. DREW. Colour from grey by optimized colour. 2010.

[23] André STOLL. *Les spirales.* avril 2000.

[24] Robert FERREOL. Encyclopédie des formes remarquables courbes, surface, fractales. 2008.

[25] http ://sta_.science.uva.nl/aloi.