République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou

Faculté de Génie Électrique Et d'Informatique
Département d'Automatique

THÈSE

pour l'obtention du diplôme de

Doctorat ES Science en Automatique

présentée et soutenue publiquement le 27 \02 \2025 par

Zohra HALICHE

Matrices aura des couleurs floues : Classification et segmentation d'images couleur texturées.

Jury

M. Salah HADDAB	Professeur	UMMTO	Président
M. Kamal HAMMOUCHE	Professeur	UMMTO	Rapporteur
M. Ludovic MACAIRE	Professeur	U/Lille de France	Co-Rapporteur
M. Khaled HARRAR	Professeur	UMB/Boumerdès	Examinateur
M. Reda KASMI	Professeur	UAM/Béjaia	Examinateur
M. Mahmoud BELHOCINE	Directeur de recherche	CDTA Alger	Examinateur
M. Idir FILALI	Professeur	UMMTO	Examinateur

Remerciements

Les travaux de recherche présentés dans cette thèse ont été réalisés au sein du Laboratoire Vision Artificielle et Automatique des Systèmes (LVAAS) de l'Université Mouloud Mammeri de Tizi-Ouzou, Algérie, en collaboration avec le Centre de Recherche en Informatique, Signal et Automatique de Lille (CRISTAL), France.

J'exprime mes sincères remerciements à mon directeur de thèse, Mr. Kamal HAMMOUCHE, Professeur à l'UMMTO, pour m'avoir accueilli au sein de son équipe de recherche. Je le remercie pour sa disponibilité, sa confiance, et son soutien constant, qui se sont manifestés dès mon parcours de Magister.

J'adresse également mes sincères remerciements à mon co-directeur de thèse, Mr. Ludovic MACAIRE, Professeur au Centre de Recherche en Informatique, Signal et Automatique de Lille (CRISTAL), pour m'avoir accueilli au sein de son équipe de recherche. Cette expérience a été marquée par un encadrement attentif, qui a largement contribué à mon épanouissement personnel et professionnel. Je lui suis profondément reconnaissante pour ses précieux conseils, son expertise scientifique et ses encouragements constants.

Je tiens à exprimer ma gratitude à Mr. Olivier LOSSON, Maitre de conférences à l'Université de Lille. Je le remercie pour ses précieuses remarques et conseils constructifs.

Je remercie vivement Mr. Salah HADDAB, Professeur à l'UMMTO, pour l'honneur qu'il me fait en acceptant de présider le jury de cette thèse.

J'adresse mes plus vifs remerciements à Mr. Khaled HARRAR, Professeur à l'Université de M'hamed Bougara de Boumerdès, pour avoir accepté d'examiner ce travail. C'est avec un énorme plaisir que je le vois participer au jury.

Je tiens aussi à remercier Mr. Reda KASMI, professeur à l'Université de Béjaïa, pour l'honneur qu'il me fait en participant au jury, ainsi que pour avoir pris la peine d'examiner ce travail.

Mes remerciements vont également à Mr. Mahmoud BELHOCINE, Directeur de recherche au Centre de Développement des Technologies Avancées (CDTA), pour l'intérêt qu'il manifeste à l'égard de ce travail et pour l'honneur qu'il me fait en acceptant de faire partie du jury.

Que Mr. Idir FILALI, Professeur à l'UMMTO, veuille trouver ici ma sincère reconnaissance pour avoir accepté d'examiner ce travail et faire partie du jury.

J'adresse mes remerciements les plus sincères à tous les membres du laboratoire LVAAS, et tout particulièrement à Abderezak SALMI, dont le soutien constant et les conseils précieux ont grandement contribué à la réussite de mon travail. Sa disponibilité et son accompagnement de chaque instant ont été d'une aide inestimable. Un grand merci également à Farid, Kahina, Damia et Dahmane pour leurs encouragements et surtout pour tous les bons moments partagés au sein du laboratoire.

Je tiens à adresser mes plus affectueux remerciements tout particulier à mes parents, mes frères et sœurs, ainsi qu'à mes neveux et nièces, pour leur présence, leurs encouragements et leur bienveillance tout au long de ce parcours. Enfin, je souhaite exprimer ma profonde gratitude à mon mari pour son soutien inestimable et sa patience, sans lesquels ce travail n'aurait pas été possible.

Zohra HALICHE

Table des matières

Liste des figures

Liste des tableaux

Abréviations

GLAM	Gray Level Aura Matrix
CLAM	Color Level Aura Matrix
MCLAM	Marginal Color Level Aura Matrix
OCLAM	Opponent Color Level Aura Matrix
CAM	Color Aura Matrix
FGLAM	Fuzzy Gray Level Aura Matrix
FCLAM	Fuzzy Color Level Aura Matrix
MFCLAM	Marginal Fuzzy Color Level Aura Matrix
OFCLAM	Opponent Fuzzy Color Level Aura Matrix
FCAM	Fuzzy Color Aura Matrix
GLCM	Gray Level Co-occurrence Matrix
CCM	Color Co-occurrence Matrix
ICM	Integrative Co-occurrence Matrices
FGLCM	Fuzzy Gray Level Co-occurrence Matrix
FCCM	Fuzzy Color Co-occurrence Matrix
FCLCM	Fuzzy Color Level Co-occurrence Matrices
LBP	Local Binary Patterns
CLBP	Completed Local Binary Patterns
LBPC	Local Binary Patterns Color
ELBP	Extended Local Binary patterns
ILBP	Improved Local Binary patterns
TS	Texture Spectrum
FLBP	Fuzzy Local Binary Patterns
FuzEnC_{2D}	Fuzzy Entropy Color 2D
MCLBP	Multiple Channels Local Binary Patterns

OCLBP	Opponent Color Local Binary Patterns
IOCLBP	Improved Opponent Color Local Binary Patterns
MOCLBP	Mixed Order Color Local Binary Patterns
SWOBP	Spatially Weighted Order Binary Patterns
SLIC	Simple Linear Iterative Clustering
SCALP	Superpixels with Contour Adherence using Linear Path
NNSC	Nearest Neighbor-based Superpixel Clustering
TASP	Texture-Aware SuperPixel
PPV	Plus Proche Voisin (PPV)
SVM	Support Vector Machine
MLP	Multi Layer Perceptron
ELM	Extreme Learning Machine
CNN	Convolutional Neural Network

Notations

$\mathbf{I}$	Image numérique
$N_{\mathbf{I}}$	Taille de l'image $\mathbf{I}$
I^k	Image d'une composante couleur k, $k \in \{C_1,C_2,C_3\}$
$\mathbf{I}(\mathbf{s})$	Couleur du pixel $\mathbf{s}$
$\mathtt{S}$	Grille finie
$\mathbf{s}$	Pixel
$\mathscr{N}$	Système de voisinage
$\mathtt{N}_{\mathbf{s}}$	Ensemble des pixels voisins du pixel $\mathbf{s}$
S_g	Ensemble des pixels ayant le niveau de gris g
S_g^k	Ensemble des pixels ayant le niveau g dans la composante couleur k
$\mathtt{A}_{S_{g'}}(S_g)$	Ensemble aura des niveaux de gris
$m_l\,(S_g,S_{g'})$	Mesure aura locale des niveaux de gris
$m_c\,(S_g,S_{g'})$	Mesure aura cardinale des niveaux de gris
M_l	Matrice aura locale
M_c	Matrice aura cardinale
$\mathtt{A}_{S_{g'}^{k'}}(S_g^k)$	Ensemble aura des niveaux des composantes couleur k et k'
$m_l^{k,k'}\,(g,g')$	Mesure aura locale des niveaux des composantes couleur k et k'
$M_l^{k,k'}$	Matrice aura locale des niveaux des composantes couleur k et k'
$m_c^{k,k'}\,(g,g')$	Mesure aura cardinale des niveaux des composantes couleur k et k'
$M_c^{k,k'}$	Matrice aura cardinale des niveaux des composantes couleur k et k'
$\tilde{g}$	Niveau de gris flou
$\mu_{\tilde{g}}$	Degré d'appartenance d'un niveau de gris flou $\tilde{g}$
μ_{S_g}	Degré d'appartenance de chaque pixel $\mathbf{s} \in \mathtt{S}$ à S_g
$\mu_{\mathtt{A}_{S_{g'}}(S_g)}$	Degré d'appartenance de l'ensemble aura des niveaux de gris flous
$\tilde{m}_l(g,g')$	Mesure aura locale floue des niveaux de gris
$\tilde{M}_l$	Matrice aura locale floue

$\tilde{m}_c(g,g')$	Mesure aura cardinale floue des niveaux de gris
$\tilde{M}_c$	Matrice aura cardinale floue
$\mu_{S_g^k}(\mathbf{s})$	Degré d'appartenance de chaque pixel $\mathbf{s} \in \mathtt{S}$ à S_g^k
$\mu_{\mathtt{A}_{S^{k'}_{g'}}(S_g^k)}$	Degré d'appartenance de l'ensemble aura des niveaux des composantes couleur flous
$\tilde{m}_l^{k,k'}(g,g')$	Mesure aura locale des niveaux de composante couleur flous
$M_l^{k,k'}$	Matrice aura locale des niveaux des composantes couleur flous
$\tilde{m}_c^{k,k'}(g,g')$	Mesure aura cardinale des niveaux des composantes couleur flous
$M_c^{k,k'}$	Matrice aura cardinale des niveaux des composantes couleur flous
$\mathbf{x}$	Couleur donnée dans l'espace RGB
$\mathtt{S}_{\mathbf{x}}$	Ensemble des pixels de couleur $\mathbf{x}$
$\mathtt{N}_{\mathbf{s},\mathtt{S}_{\mathbf{x}}}$	Ensemble des pixels voisins de chaque pixel $\mathbf{s} \in \mathtt{S}_{\mathbf{x}}$
$\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_{\mathbf{x}})$	Ensemble aura couleur de $\mathtt{S}_{\mathbf{x}}$ par rapport à $\mathtt{S}_{\mathbf{x}'}$
$\mathtt{N}_{\mathbf{s}}$	Ensemble des pixels voisins du pixel $\mathbf{s}$
$m_l\,(\mathbf{x},\mathbf{x}')$	Mesure aura locale des couleurs $\mathbf{x}$ et $\mathbf{x}'$
$m_c(\mathbf{x},\mathbf{x}')$	Mesure aura cardinale des couleurs $\mathbf{x}$ et $\mathbf{x}'$
C^k	Centre des intervalles pour chaque composante couleur
L^k	Largeur de chaque intervalle
C	Nombre de couleurs retenues parmi les 256^3 couleurs possibles
$\mathscr{C}$	Sous-ensemble de C couleurs
$\mathbf{c} \in \mathscr{C}$	Couleur nette
$\tilde{\mathbf{c}}$	Couleur floue
$\mu_{\tilde{\mathbf{c}}} : \mathrm{RGB} \longrightarrow [0,1]$	Fonction d'appartenance
$\mu_{\tilde{\mathbf{c}}}(\mathbf{x})$	Degré d'appartenance de toute couleur $\mathbf{x} \in \mathrm{RGB}$ à la couleur floue $\tilde{\mathbf{c}}$
α , β et ζ	Constantes réelles positives utilisées pour contrôler l'étendue du flou
$K_{\mathbf{I}}$	Nombre de classes de textures présentes dans l'image $\mathbf{I}$
T	Nombre de prototypes par classe
t	numéro du prototype
$\mathbf{W}^t$	Patch de taille $(2\omega+1)^2$ centré sur chaque pixel prototype
N	Nombre de données d'apprentissages
$\mathbf{x}_j$	Vecteur d'attribut de chaque pixel prototype
X	Ensemble des prototypes
$\mathbf{X}$	Vecteur d'attributs correspondant à l'ensemble X
$\mathbf{Y}$	Vecteur des étiquettes correspond à chaque texture de la base d'apprentissage

P	Nombre de super-pixels
L	Nombre de neurones dans la couche cachée
$\mathtt{P}^p$	Superpixel p
$f_{\mathtt{R}}$	Attribut régional
$\mathtt{N}^p_{\mathbf{s},\mathtt{S}_\mathbf{c}}$	Ensemble des pixels voisins de chaque site $\mathbf{s} \in \mathtt{S}_\mathbf{c}$ situés dans le superpixel $\mathtt{P}^p$
$\mathtt{A}^p_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})$	Ensemble aura relatif au superpixel $\mathtt{P}^p$
$m^p(c,c')$	Mesure aura couleur cardinale relative au superpixel $\mathtt{P}^p$
$\mu_{\mathtt{A}^p_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})}(\mathbf{r})$	Ensemble aura des couleurs floues relatif au superpixel $\mathtt{P}^p$
$n^p(\mathbf{r},\mathbf{s})$	Fonction de voisinage qui exprime le degré d'appartenance du pixel $\mathbf{r}$ dans le voisinage du pixel $\mathbf{s}$ dans un superpixel $\mathtt{P}^p$
$\tilde{m}^p(\mathbf{c},\mathbf{c}')$	Mesure aura cardinale des couleurs floues pour un superpixel $\mathtt{P}^p$
$\bar{M}^p$	matrice aura cardinale des couleurs floues pour un superpixel $\mathtt{P}^p$
$\mathbf{x}^p$	Vecteur d'attributs d'un superpixel $\mathtt{P}^p$
$\hat{\mathbf{y}}_p$	Vecteur de sortie de dimension $K_\mathbf{I}$ pour le superpixel $\mathtt{P}^p$

Introduction Générale

1 Contexte

L'analyse d'images est une discipline de l'informatique et de la vision par ordinateur en constante évolution et qui consiste à extraire des informations significatives à partir d'images numériques en vue de segmenter, classer ou comprendre les contenus visuels. L'analyse d'images est de plus en plus appliquée dans des domaines très variés tels que la médecine, la télédétection, l'industrie agroalimentaire, l'industrie automobile (véhicules électriques ou autonomes), la robotique, les applications multimédia, etc. Ce gain d'intérêt est dû au développement des ordinateurs, à la miniaturisation et à la baisse des coûts des appareils d'acquisition d'images, tels que les cameras couleur, les scanners, les radars, les téléphones portables, les drones, ainsi qu'au développement des méthodes de l'intelligence artificielle englobant l'apprentissage automatique (machine learning) et l'apprentissage profond (deep learning).

Concrètement l'analyse d'images consiste à caractériser localement chaque pixel de l'image ou globalement l'image entière par un ensemble d'attributs. Dans le premier cas, ces attributs servent à partitionner l'image en régions homogènes. Cette opération, dite segmentation, a pour but de décrire ou de détecter les objets contenus dans l'image. Dans le second cas, les attributs sont utilisés pour identifier ou reconnaitre une image ou un objet parmi un ensemble d'images groupées en classes. On parle alors de classification des images. L'analyse d'images s'appuie essentiellement sur deux attributs visuels : la texture et la couleur, car celles-ci sont omniprésentes dans la quasi-totalité des images naturelles et permettent dans le cas de la segmentation de discerner des régions ayant chacune une texture couleur spécifique et de discriminer des objets différents, dans le cas de la classification des images.

Ces attributs peuvent être des descripteurs couleurs comme les composantes trichromatiques Rouge (R), Vert (G) et Bleu (B) de l'image couleur, ou de texture comme ceux dérivés des fameuses matrices de co-occurrence des niveaux de gris (Gray Level Cooccurence Matrices ou GLCMs) [1], ou une combinaison des deux types d'attributs (texture couleur).

Notre intérêt dans cette thèse concerne les méthodes d'extraction d'attributs de texture couleur. Un grand nombre de ces méthodes sont proposées dans la littérature et qu'on peut regrouper en deux catégories : les méthodes traditionnelles (handcrafted methods) et les méthodes basées sur l'apprentissage qui englobent les modèles de réseaux de neurones convolutifs (Convolutional Neural Networks, CNNs) [2, 3]. Les méthodes de la première catégorie se divisent en quatre approches en fonction de la manière de combiner les informations de couleur et de texture. Dans l'approche parallèle, les attributs de texture extraits de l'image luminance sont simplement combinés avec les attributs couleur. Dans l'approche marginale les attributs de texture sont calculés sur chaque composante couleur puis concaténés dans un seul vecteur d'attributs. Dans l'approche opposée, les attributs de texture couleur sont obtenus en considérant les relations inter (et intra) composantes. Dans cette approche on trouve la méthode basée sur les matrices de co-occurrence chromatiques [4, 5]. L'approche compacte exploite la représentation vectorielle des couleurs pour analyser les relations entre les couleurs des pixels voisins, comme dans le cas des images en niveaux de gris.

D'un autre côté, les images numériques sont plus ou moins floues en raison de l'imprécision inhérente à la discrétisation du domaine spatial (échantillonnage) et des niveaux de gris ou des couleurs (quantification). De plus la couleur et la texture sont des notions ambiguës et les frontières séparant les différentes régions de l'image ne sont pas définies avec précision. Il est alors indispensable de tenir compte de toutes ces imprécisions lors de l'analyse des images couleur texturées. Dans cette optique, des techniques d'analyse de texture basées sur la théorie des ensembles flous ont été élaborées, parmi lesquelles on peut citer celle basée sur les matrices de co-occurrences des couleurs floues (Fuzzy Color Co-occurrence Matrices, FCCMs) [6].

Nous nous sommes focalisés dans cette thèse sur la méthode d'extraction des attributs de texture basée sur les matrices aura des niveaux de gris (GLAMs: Gray level aura matrices) et leur extension aux images couleur floues. Les GLAMs, initialement proposées par Elfadel

et Picard [7], reposent sur un concept théorique appelé "ensemble aura" qui décrit la présence relative d'un ensemble de pixels ayant un certain niveau de gris dans le voisinage d'un autre ensemble de pixels ayant un niveau de gris différent. Un ensemble aura peut être caractérisé par une valeur, appelée "mesure aura", qui exprime le degré de mélange entre les deux ensembles de pixels voisins. Une GLAM regroupe toutes les mesures aura des paires d'ensembles de pixels associées à toutes les paires de niveaux de gris présents dans une image. Avec la mesure aura d'Elfadel et Picard (qualifiée par la suite de mesure locale), les GLAMs locales peuvent être considérées comme une généralisation des GLCMs. Par la suite, Hammouche et al. ont proposé une autre mesure cardinale de l'ensemble aura et les GLAMs cardinales associées ont été étendues au cas flou (FGLAMs: Fuzzy Gray level aura matrices), puis appliquées aux images couleurs selon l'approche marginale [8]. Les GLAMs et FGLAMs se sont révélées un outil efficace pour caractériser les images texturées [9, 10, 8].

2 Objectifs et contributions

Notre objectif dans cette thèse est d'étendre la notion des matrices aura (nette et floue) aux images couleur, afin d'exploiter pleinement les informations de texture et de la couleur présentes dans une image. Pour cela, trois approches sont possibles.

Dans la première approche, une matrice aura des niveaux des composantes couleur nette (MCLAM: Marginal Color Level Aura Matrix) et floue (MFCLAM: Marginal Fuzzy Color Level Aura Matrix) est déterminée pour chaque composante de l'image couleur de manière indépendante. Cette approche marginale, proposée par Hammouche et al. dans le cadre de la classification des textures couleur [8], ignore malheureusement les interactions entre les composantes couleurs.

Pour pallier cet inconvénient, nous proposons dans cette thèse une seconde approche (opposée ou conjointe) qui consiste à déterminer six matrices (inter composantes) sur des paires de composantes couleur nette (OCLAMs: Opponent Color Level Aura Matrices) et floue (OFCLAMs: Opponent Fuzzy Color Level Aura Matrices). Néanmoins, l'exploitation des éléments de ces matrices sous forme d'attributs dans le cadre de la classification ou de la

segmentation d'images couleur texturées se heurte au problème d'espace mémoire et temps de calcul.

Pour contourner ce problème, nous proposons d'exploiter la stratégie compacte pour définir une seule matrice aura des couleurs dans les deux contextes net (CAM: Color Aura Matrix) et flou (FCAM: Fuzzy Color Aura Matrix). Cette troisième approche permet de prendre en considération l'interaction spatiale entre les couleurs des pixels tout en réduisant l'espace mémoire.

Par ailleurs, nous proposons une méthode de segmentation supervisée d'images couleur texturées dans laquelle les FCAMs sont utilisées pour caractériser des superpixels obtenus par l'intermédiaire d'un algorithme SLIC (Simple Linear Iterative Clustering) modifié, spécifiquement développé dans le cadre de cette thèse. Ces superpixels sont classés par l'intermédiaire du classifieur supervisé basé sur les réseaux de neurones dénommé Extrema Learning Machine (ELM).

3 Organisation de la thèse

Le travail réalisé est rapporté dans cette thèse qui comporte principalement quatre chapitres ainsi qu'une conclusion générale.

Dans le premier chapitre, nous exposons d'abord quelques notions sur la couleur, la texture en niveaux de gris, suivi d'un état de l'art sur les différentes approches d'analyse d'images texturées. Puis, nous définissons la texture couleur et établissons un état de l'art sur les différents attributs de texture couleur à travers les différentes stratégies utilisées pour combiner la texture et la couleur. Par la suite, nous aborderons l'aspect ambigu et imprécis lié à la couleur et à la texture, et nous montrerons comment traiter ces imprécisions à l'aide de la théorie des ensembles flous. Nous décrirons alors quelques attributs de texture flous (des niveaux de gris et des couleurs). La fin de ce chapitre est réservée à la classification et à la segmentation d'images couleur texturées.

Le deuxième chapitre est consacré au concept aura. Les notions d'ensemble aura, de mesure aura et de matrice aura des niveaux de gris (GLAM) sont tout d'abord présentées. Ces

notions sont étendues dans un premier temps aux images couleur en introduisant les matrices aura des niveaux des composantes couleur (CLAMs). Dans un deuxième temps, les matrices aura des niveaux de gris flous (FGLAMs) sont exposées tout en montrant, à travers un exemple, leur avantage sur les GLAMs. Au final, l'extension des FGLAMs aux images couleur selon les deux stratégies marginale (MFCLAMs) et opposée (OFCLAMs) sera décrite.

Le troisième chapitre montre comment définir une seule matrice aura pour caractériser des textures couleur dans les deux contextes net (CAM) et flou (FCAM). La pertinence des attributs de texture couleur représentés par les éléments de la FCAM et celle des OFCLAMs est évaluée dans le cadre de la classification des textures couleur. Une étude sur la robustesse des FCAMs vis-à-vis des conditions d'acquisition comme la rotation, la résolution et les dégradations (bruit, flou) est menée. Une comparaison des performances des FCAMs et des OFCLAMs avec d'autres types d'attributs est effectuée.

Le quatrième chapitre est dédié à la méthode de segmentation d'images de textures couleur élaborée. Nous détaillons chaque étape de l'algorithme, en mettant particulièrement l'accent sur l'algorithme Superpixel régional proposé. Ensuite, nous évaluons ses performances sur deux bases de textures couleur naturelles (Prague) et satellitaires (ALI), tout en les confrontant à celles des autres méthodes de segmentation supervisée d'images couleur texturées tirées de l'état de l'art.

Nous concluons cette thèse en dressant un bilan général des contributions apportées et en proposant quelques perspectives pour ce travail.

Chapitre 1

Analyse d'images couleur texturées

1.1 Introduction

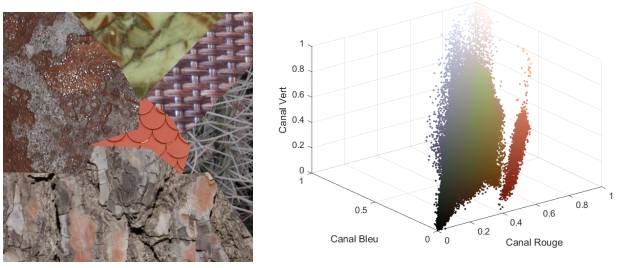
La classification et la segmentation des textures couleurs sont des procédures très utilisées dans diverses applications de vision artificielle, dans le but de reconnaissance ou d'identification d'objets. Ces deux opérations partagent souvent une phase commune qui est l'extraction d'attributs de texture couleur. Cette phase d'analyse a pour but de caractériser localement (segmentation) ou globalement (classification) les textures couleurs présentes dans l'image. Ce chapitre propose un panorama des méthodes d'extraction d'attributs de texture couleur. Une attention particulière est accordée aux méthodes d'analyse de textures couleur floues. Dans un premier temps, nous présentons quelques notions liées à la couleur, à la texture et à la texture couleur. Par la suite, nous explorons les principales techniques d'extraction d'attributs de texture en niveaux de gris et en couleur. Dans un second temps, nous focaliserons sur l'aspect incertain et vague des textures couleurs, puis nous montrerons comment traiter ces incertitudes à l'aide de la théorie des ensembles flous. Nous décrirons alors quelques méthodes de textures en niveaux de gris flous et couleur. Nous concluons ce chapitre par un état de l'art sur les méthodes de segmentation et de classification d'images couleur texturées.

1.2 Texture couleur nette (crisp)

1.2.1 Couleur

1.2.1.1 Définition

La couleur joue un rôle important dans la vision humaine car elle est le résultat de la perception visuelle de la lumière réfléchie par les objets. Cette lumière est composée de différentes longueurs d'onde, chacune associée à une couleur spécifique dans le spectre visible. Selon le principe de la trichromie, la couleur peut être reproduite par trois couleurs primaires : rouge (R), vert (G) et bleu (B). Les systèmes d'acquisition d'images couleur (caméras couleur) exploitent ce principe pour effectuer un échantillonnage et une quantification en 8 bits pour chaque composante couleur R, G et B. Par conséquent, l'image finale peut prendre jusqu'à 256^3 couleurs différentes. Les couleurs perçues peuvent être représentées par des points dans un espace couleur tridimensionnel, où les coordonnées correspondent aux niveaux des composantes couleur R, G et B de chaque pixel de l'image. La Figure 1.1 montre un exemple d'une image couleur avec sa représentation dans un espace RGB.



(a) (b)

FIG. 1.1: Représentation de l'image couleur dans l'espace RGB.

D'autres espaces de couleur ont été proposés dans la littérature et qui peuvent être regroupés en quatre familles [11] :

- *Espaces de représentation primaires :* Ils sont basés sur l'utilisation de trois couleurs fondamentales : rouge, vert et bleu, également appelées couleurs primaires. Parmi ces espaces, on peut mentionner l'espace de couleur RGB et l'espace de couleur XYZ.

- *Espaces de représentation Luminance-Chrominance :* Les espaces de représentation Luminance-Chrominance sont composés de trois composantes : la première représente l'intensité, appelée "luminance", tandis que les deux autres composantes permettent de quantifier une information de couleur, appelée "chrominance". La première composante est indépendante des deux autres, ce qui permet de traiter la composante d'intensité sans affecter l'information de couleur. Parmi les différents espaces de luminance-chrominance, on peut citer l'espace de couleur La^*b^* [11].

- *Espaces de représentation perceptuels :* Les espaces de représentation perceptuels visent à quantifier les aspects subjectifs de la perception humaine des couleurs, notamment ceux liés à la teinte, à la saturation et à la luminosité. La *teinte* (hue) qui correspond à la couleur est définie par un angle en degrés (de 0 à 360), la *saturation* mesure le degré de pureté de la couleur par rapport à un illuminant (0 pour une teinte non saturée à 1 pour le niveau de saturation maximum), et la *valeur* (value) qui mesure la luminosité de la couleur (0 pour le noir et 1 pour le blanc). Parmi les espaces de cette famille, on peut citer : HSV, HSI, et TLS [11].

- *Espaces d'axes indépendants :* Parmi les inconvénients de l'espace RGB, les composantes rouge, verte et bleue sont corrélées et dépendantes de la luminance. Pour remédier à ce problème, l'espace de couleur I_1,I_2,I_3 a été introduit en 1980 par Ohta [12]. La composante I_1 représente la luminance alors que les composantes I_2 et I_3 représentent respectivement les oppositions bleu-rouge et magenta-vert.

D'une manière générale, une image couleur numérique $\mathbf{I}$ est constituée de trois images composantes I^k telles que $k \in \{C_1,C_2,C_3\}$, chacune correspondant à l'une des composantes de l'espace de représentation de la couleur. Chaque pixel $\mathbf{s}$ dans une image couleur $\mathbf{I}$ est représenté par trois composantes couleur indépendantes.

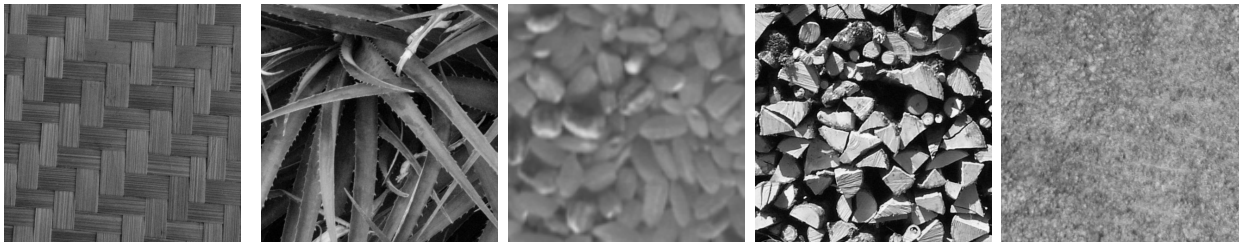
1.2.1.2 Attributs couleur

Plusieurs attributs peuvent être utilisés pour décrire les images couleur. Ces méthodes, dites spectrales, exploitent la couleur pure des pixels sans tenir compte des interactions spatiales entre ces pixels. Parmi ces attributs, on peut citer la moyenne des composantes couleur (R, G, B) ou d'autres composantes dérivées des espaces de couleurs tels que HSV, ou La^*b^* [13], la variance qui mesure la dispersion des composantes couleur autour de leur moyenne, la dissymétrie qui mesure l'asymétrie des composantes couleur par rapport à leur moyenne, les histogrammes de couleurs, la couleur dominante [14], la couleur moyenne[15], le correlogramme de couleur [16] et les moments statistiques [17].

1.2.2 Texture en niveaux de gris

1.2.2.1 Définition

Contrairement à la couleur, la texture est une caractéristique facile à identifier par le système visuel humain, mais difficile à définir. Selon le dictionnaire Robert, le mot "texture" trouve son origine dans le mot latin "textura" et désigne l'arrangement des fils dans un tissu. La texture est également définie par des termes décrivant son apparence comme fine ou grossière, lisse ou rugueuse, bosselée ou striée, régulière ou irrégulière, ridée ou granuleuse, isotrope ou anisotrope, tachetées, marbrées, autant de termes qualificatifs qui sont proches de nos concepts perceptifs (Fig. 1.2).



(a) Régulière (b) Irrégulière (c) Granuleuse (d) Grossière (e) Fine

FIG. 1.2: Différents types de textures

D'autres définitions sont proposées dans la littérature, comme celle de Haralick qui considère la texture comme un agencement spatial plus ou moins régulier d'un motif de base

ou de primitives (textons) [1]. Une primitive est une agglomération connexe de pixels ayant des propriétés similaires. Une autre définition probabiliste, fournie par Sklansky, considère la texture comme une région de l'image pour laquelle les propriétés statistiques sont constantes ou approximativement périodiques [18]. Des définitions spécifiquement adaptées à la segmentation d'images texturées sont aussi proposées. L'une d'elles considère une texture comme une région de l'image pour laquelle on peut définir une fenêtre de dimension minimale, à travers laquelle l'observation se traduit par une impression visuelle identique pour toutes translations possibles de cette fenêtre à l'intérieur de la région considérée. Pour un pixel la texture est définie à partir des informations apportées par l'ensemble des pixels appartenant à son voisinage, alors que, la texture d'une région donnée désigne les caractéristiques de la répartition spatiale des niveaux de gris des pixels dans cette région.

Toutes ces définitions mettent en évidence l'importance de prendre en considération l'arrangement spatial des niveaux de gris des pixels et l'aspect spatial qui se traduit par un voisinage spatial dont la taille dépend du type de la texture ou de la surface de son motif de base pour caractériser une texture.

1.2.2.2 Attributs de texture

Face à la difficulté de donner une définition précise de la notion de texture, diverses méthodes d'extraction d'attributs de texture ont été proposées pour caractériser les textures. Ces attributs doivent être représentatifs, pertinents et discriminants de manière à permettre la distinction entre différentes textures. Leur extraction représente un défi fondamental dans de nombreuses applications, telles que la classification des textures et la segmentation d'images de textures. De nombreuses méthodes d'analyse des textures ont été développées [19, 20]. Parmi elles on peut citer:

- Les méthodes basées sur la modélisation spatiale des textures à base des modèles autorégressifs [21], des modèles fractals [22] et des modèles Markoviens [23].
- Les méthodes basées sur des transformations comme la transformée en cosinus discrète (DCT), les filtres de Laws, la transformée de Gabor [24] et la transformée en ondelettes [25].

- Les méthodes par apprentissage, notamment celles basées sur le réseau de neurones ELM (Extrema Learning Machine) [26] et sur les modèles de réseaux de neurones convolutifs (Convolutional Neural Networks, CNNs) tels que ResNet-50, VGG 16, GoogleNet et AlexNet [2, 3]. Ces méthodes sont principalement employées dans la classification des images (voir section 1.4).
- Les méthodes statistiques telles que les matrices de co-occurrences des niveaux de gris (Gray Level Cooccurrence Matrices ou GLCMs) [1] et les motifs binaires locaux (Local Binary Patterns ou LBP) [27].

Ces dernières étant très populaires, nous donnons ci-dessous une description succincte :

Matrices de co-occurrence des niveaux de gris (GLCM)

La matrice de co-occurrence des niveaux de gris (Gray Level Co-occurrence Matrix ou GLCM), introduite par Haralick en 1973 [1], est considérée comme une référence dans le domaine d'analyse de la texture. La matrice de co-occurrence permet d'analyser les relations des niveaux de gris de pixels pris deux à deux, Elle peut être définie comme suit :

Soit I une image en niveaux de gris définie sur une grille finie $\mathtt{S}$, composée d'un ensemble de pixels $\mathbf{s}$, où chaque pixel $\mathbf{s} \in \mathtt{S}$ est défini par ses coordonnées (x,y) sur la grille et est caractérisé par son niveau de gris $I(\mathbf{s})$. Soit q le niveau de gris maximal dans l'image et g le niveau de gris du pixel $\mathbf{s}$, tel que $g \in \Lambda = \{0,1,\ldots,q-1\}$.

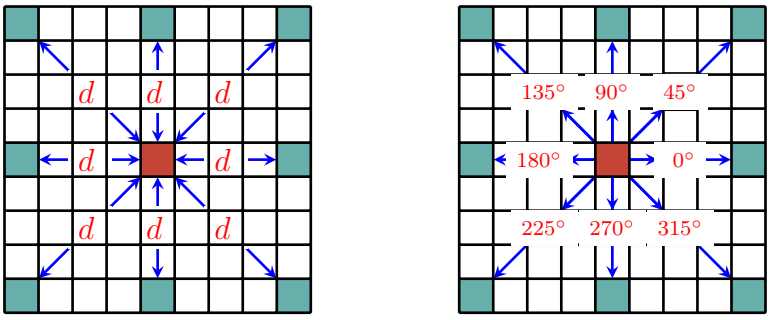
Soit $c^{\mathbf{d}}(g,g')$ le nombre de couples de pixels séparés par un vecteur de distance $\mathbf{d}$ dont l'un a un niveau de gris g et l'autre a un niveau de gris g' tel que :

$$c^{\mathbf{d}}(g,g') = \left|(\mathbf{s},\mathbf{r}) \in \mathtt{S}^2 \text{ tel que } \mathbf{s}+\mathbf{d}=\mathbf{r}, (I(\mathbf{s})=g) \wedge (I(\mathbf{r})=g')\right|. \tag{1.1}$$

Le vecteur de distance $\mathbf{d}$ est défini par une distance spatiale (norme) de Tchebychev d (en pixels) et une orientation θ entre deux pixels (Fig. 1.3).

La matrice de co-occurrence des niveaux de gris notée, $C(d,\theta)$ est une matrice de dimension $(q \times q)$ regroupant tous les couples de niveaux de gris (g,g') tels que : $C\,(d,\theta) = \left[c^{\mathbf{d}}(g,g')\right]$.

Les matrices de co-occurrence usuelles sont définies par une distance $d = 1$ ou 2 entre les deux pixels et les directions : horizontale ($\theta = 0°$), verticale ($\theta = 90°$), diagonale gauche ($\theta = 45°$) et diagonale droite ($\theta = 135°$).



(a) (b)

FIG. 1.3: Différentes distance d et orientations θ dans les GLCMs.

Quatre à huit GLCMs sont nécessaires pour caractériser une image. Ces matrice contiennent donc beaucoup d'informations et sont difficiles à utiliser dans leur ensemble. À cet effet, Haralick et al. [1] ont proposé de condenser ces informations par quatorze attributs.

Motifs binaires locaux

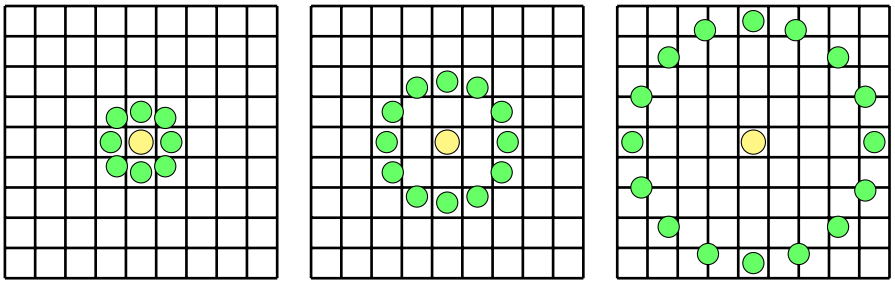
Cette méthode, connue sous le nom de LBP (Local Binary Pattern), a été proposée par Ojala et al. [27]. Elle considère un nombre de pixels voisins P et une distance R reliant un pixel donnée à ses P pixels voisins (Fig. 1.4). La méthode LBP attribue un code appelé « motif binaire local » à chaque pixel $\mathbf{s}$ de l'image en fonction du niveau de gris du pixel $\mathbf{s}$ et celle de ses P voisins. Si la valeur du pixel voisin est supérieure ou égale à celle du pixel central $\mathbf{s}$, on attribue la valeur 1, sinon 0. Les résultats de ces comparaisons sont pondérés et sommés afin d'obtenir le code binaire du pixel $\mathbf{s}$, tel que :

$$\mathrm{LPB}_{P,R}(\mathbf{s}) = \sum_{i=0}^{P-1} S(g_i - g) \cdot 2^i \tag{1.2}$$

g représente le niveau de gris du pixel $\mathbf{s}$ et g_i le niveau de gris du $i^{\text{ème}}$ pixel voisin. La fonction $S(x)$ est définie comme suit :

$$S(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{sinon} \end{cases} \tag{1.3}$$

L'opérateur $\text{LPB}_{P,R}$ contient 2^P valeurs différentes (256 dans le cas où $P = 8$).



(a) $P = 8$, $R = 1$ (b) $P = 12$, $R = 2$ (c) $P = 16$, $R = 4$

FIG. 1.4: Exemples de voisinages utilisés pour le calcul des LBP.

L'histogramme des codes LBP de tous les pixels de l'image est alors utilisé comme vecteur d'attributs.

$$h(k) = \sum_{\mathbf{s}=1}^{|\mathbf{S}|}(\text{LBP}_{P,R}(\mathbf{s}) == k), \quad k = 1, \ldots, 2^P \tag{1.4}$$

Plusieurs variantes du LBP ont été proposées, parmi lesquelles on peut citer :

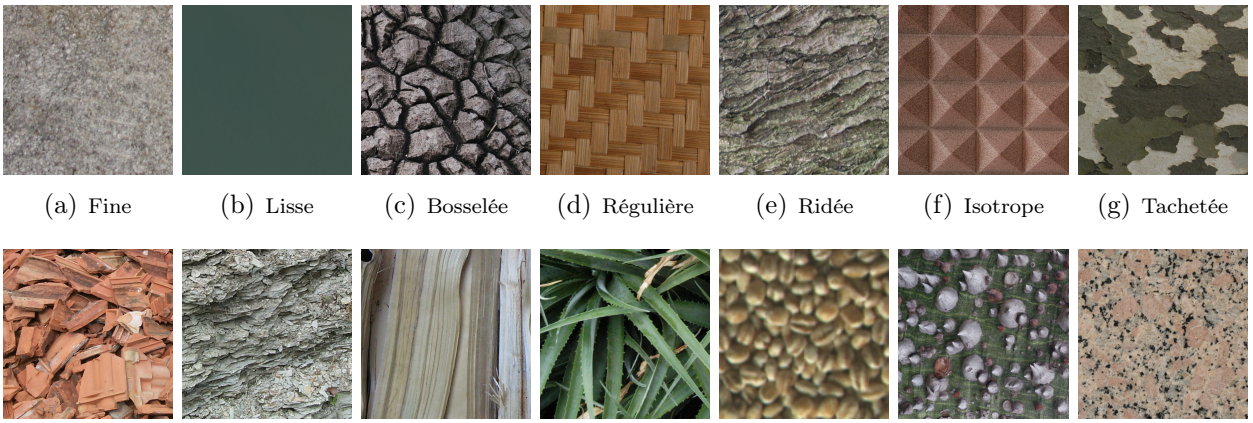
- LBP invariant par rotation ($\text{LPB}^{ri}_{P,R}$): Cette méthode a été proposée afin d'obtenir des attributs de texture invariants à la rotation [28]. Elle permet également de réduire la taille du vecteur d'attributs, puisque seulement 36 motifs LBP sont considérés au lieu de 256 dans le cas de $P = 8$.
- LBP uniforme ($\text{LPB}^{u2}_{P,R}$), proposé par Ojala et al. [28] utilise une mesure d'uniformité U, qui correspond au nombre de transitions spatiales de bits (changements 0/1) dans la chaîne binaire circulaire. Le nombre de codes différents est égal à $P(P-1)+3$, soit 59 pour $P = 8$, dont 58 sont les motifs uniformes, et un seul code pour les motifs non-uniformes.
- LBP uniforme invariant par rotation ($\text{LPB}^{r_i u_2}_{P,R}$), également proposé par Ojala et al. [28] et qui combine LBP uniforme et LBP invariants par rapport à la rotation.
- CLBP (Completed LBP) : Cette méthode, introduite par Guo et al. en 2010 [29], combine trois opérateurs complémentaires, à savoir, l'opérateur Signe $\text{CLBP\_}S_{P,R}$ (identique à celui de LBP standard), l'opérateur Magnitude $\text{CLBP\_}M_{P,R}$ (Amplitude des différences locales), l'opérateur Center $\text{CLBP\_}C_{P,R}$ (pixels centraux). Le vecteur d'attributs est obtenu par les éléments d'un histogramme joint 3D construit à partir des valeurs $\text{CLBP\_}S_{P,R}$, $\text{CLBP\_}M_{P,R}$ et $\text{CLBP\_}C_{P,R}$ de tous les pixels de l'image.

- ILBP (Improved LBP) : Cette méthode, introduite par Jin et al. [30], utilise, à la place de la valeur g du pixel central s, la moyenne de tous les pixels situés dans son voisinage.
- Spectre de texture : Similaire à la méthode LBP, le spectre de texture (Texture Spectrum : TS) introduit par Wang et He [31] consiste à attribuer une valeur 0, 1, ou 2 à chaque pixel de l'image en fonction des niveaux de gris de ses voisins situés dans un voisinage 3×3.

1.2.3 Texture couleur

1.2.3.1 Définition

Du point de vue perceptif, une texture couleur correspond à une sensation visuelle résultant de la combinaison d'un motif répétitif ou aléatoire avec des variations de teinte, saturation et luminosité. Les textures couleurs sont perçues comme une unité cohérente intégrant motif et couleur. Drimbarean et Whelan [32] définissent la texture couleur comme un motif spatio-chromatique, qui caractérise une "distribution des couleurs sur une surface ". D'un autre côté, les textures couleur peuvent être définies, comme pour les textures en niveaux de gris, par des termes linguistiques tels que : fine ou grossière, lisse ou rugueuse, bosselée ou striée, régulière ou irrégulière, ridée ou granuleuse, isotrope ou anisotrope, tachetée, ou marbrée (Fig. 1.5). Mais contrairement aux textures en niveaux de gris, qui se basent uniquement sur les variations d'intensité, les textures couleurs prennent en compte les interactions complexes entre les différentes composantes de couleur, offrant ainsi une information supplémentaire dans l'analyse d'images. En effet, comme on peut le voir sur la figure 1.6, deux objets différents ayant la même couleur et la même forme peuvent être différenciés par leurs textures. La figure 1.7 montre par contre deux objets différents ayant la même texture (en niveaux de gris) et la même forme, qui peuvent être différenciés par leurs couleurs. Ces observations mettent en évidence l'importance de prendre en compte à la fois l'aspect chromatique et spatial pour caractériser pleinement la texture couleur.



(a) Fine (b) Lisse (c) Bosselée (d) Régulière (e) Ridée (f) Isotrope (g) Tachetée

(h) Grossière (i) Rugueuse (j) Striée (k) Irrégulière (l) Granuleuse (m) Anisotrope (n) Marbrée

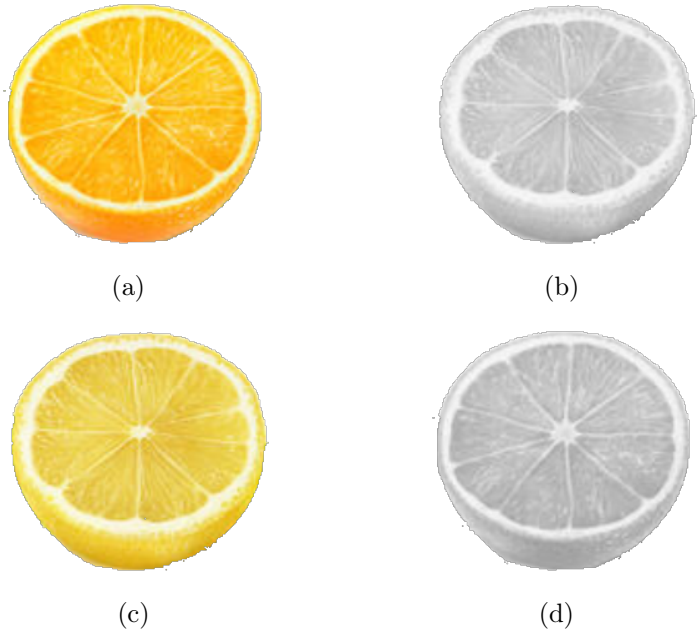
FIG. 1.5: Exemples de textures couleur.

(a) (b)

FIG. 1.6: Deux objets différents ayant la même couleur et la même forme mais des textures différentes.

1.2.3.2 Attributs de texture couleur

Plusieurs travaux, notamment ceux menés par Drimbarean [32], Palm [5], ainsi que Bianconi et Fernández [33], ont mis en évidence l'importance de l'exploitation de l'information couleur en plus de la texture dans la caractérisation des textures couleur en offrant des résultats bien plus intéressants que de prendre les informations couleur et texture séparément. On trouve dans la littérature un grand nombre de méthodes d'extraction d'attributs de texture couleur [34]. Ces méthodes se basent sur les mêmes techniques d'analyse des textures en niveaux de gris (GLCM, LBP, Transformées de Gabor, d'ondelettes, etc). Elles exploitent la couleur dans l'analyse des textures en s'appuyant sur l'hypothèse que les informations de texture et de couleur peuvent être combinées (voir la figure 1.8) soit de manière indépendante (approche parallèle) soit conjointement (approche conjointe) [35, 5, 33].



(a) (b)

(c) (d)

FIG. 1.7: Deux objets différents de même texture mais de couleurs différentes:(a) Une orange en couleur. (b) Une orange en niveaux de gris. (c) Un citron en couleur. (d) Un citron en niveaux de gris.

Approche parallèle

Dans l'approche parallèle, illustrée par la figure 1.9, les attributs de couleur et de texture sont calculés de manière distincte. Les attributs de couleur sont obtenus à partir de la distribution des couleurs dans l'espace de représentation, tandis que les attributs de texture sont calculés sur une image en niveaux de gris (luminance) en ignorant la couleur. Les attributs de couleur et de texture sont ensuite combinés dans un seul vecteur d'attributs. Dans cette approche, toutes les méthodes bien connues de l'analyse de la texture en niveaux de gris et de la couleur peuvent être appliquées directement. A titre d'exemple, Permuter et al. [15] ont présenté une méthode de cette approche parallèle où les attributs de texture, correspondent aux coefficients d'un modèle autorégressif 2D, les coefficients de la transformée en ondelettes et les coefficients de la transformée en cosinus discrète (DCT). En ce qui concerne les attributs de couleur, les auteurs ont utilisé la moyenne et la covariance, dérivées des composantes des espaces de couleur RGB et La*b*. Dans une autre étude menée par Nakyoung et al. [36], la moyenne, l'écart-type et l'asymétrie extraits à partir des composantes de chaque canal de couleur dans l'espace de couleur HSV sont utilisés comme attributs couleur, tandis que les attributs de texture correspondent aux attributs de Haralick extraits des matrices des

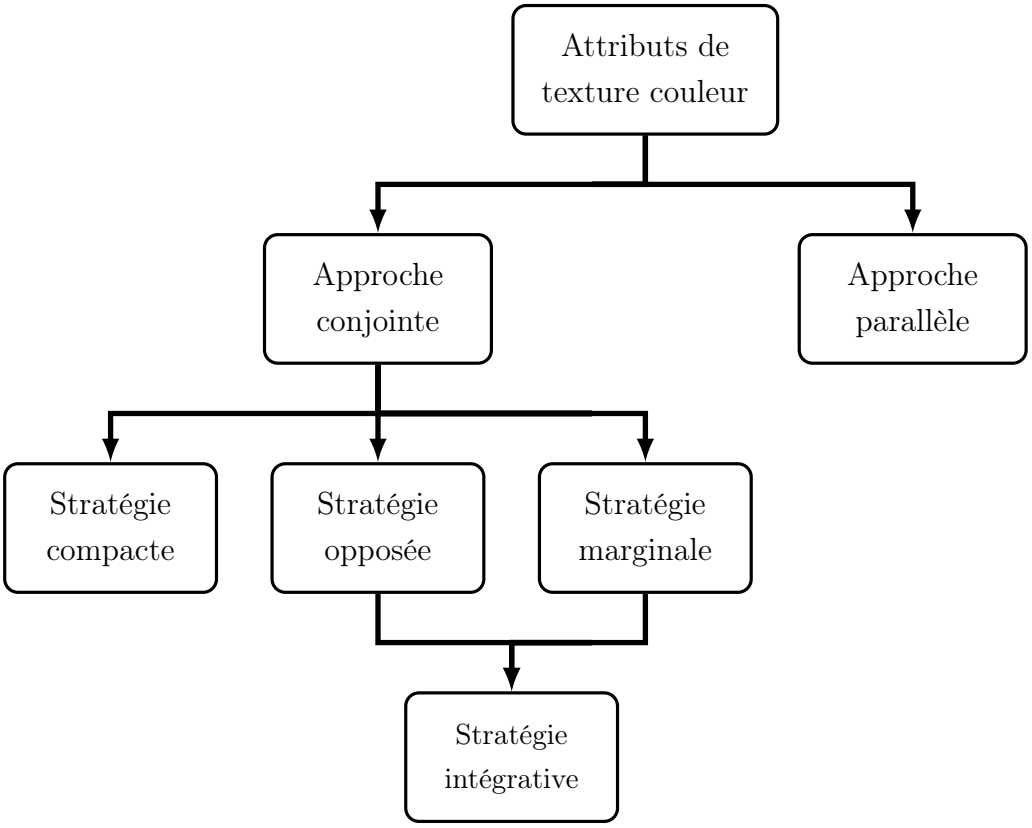


FIG. 1.8: Catégorisation des méthodes d'extraction des attributs de texture couleur.

co-occurrences des niveaux de gris. Dans [37], les auteurs explorent une méthode qui combine deux types d'attributs, à savoir l'histogramme de couleur et l'histogramme des motifs binaires locaux (LBP).

Approche conjointe

Dans l'approche conjointe, on suppose que les attributs de couleur et de texture sont mutuellement dépendants [38]. Cela permet de caractériser conjointement la distribution spatiale et la distribution de couleur. Dans cette approche, quatre stratégies différentes peuvent être envisagées pour calculer les caractéristiques de texture couleur : la stratégie marginale, la stratégie opposée, la stratégie intégrative et la stratégie compacte [5, 38, 33]. Cette approche est illustrée par la figure 1.10.

Stratégie marginale : La stratégie marginale prend uniquement en compte les relations entre les niveaux couleurs au sein d'une même composante (intra-composante) sans prendre

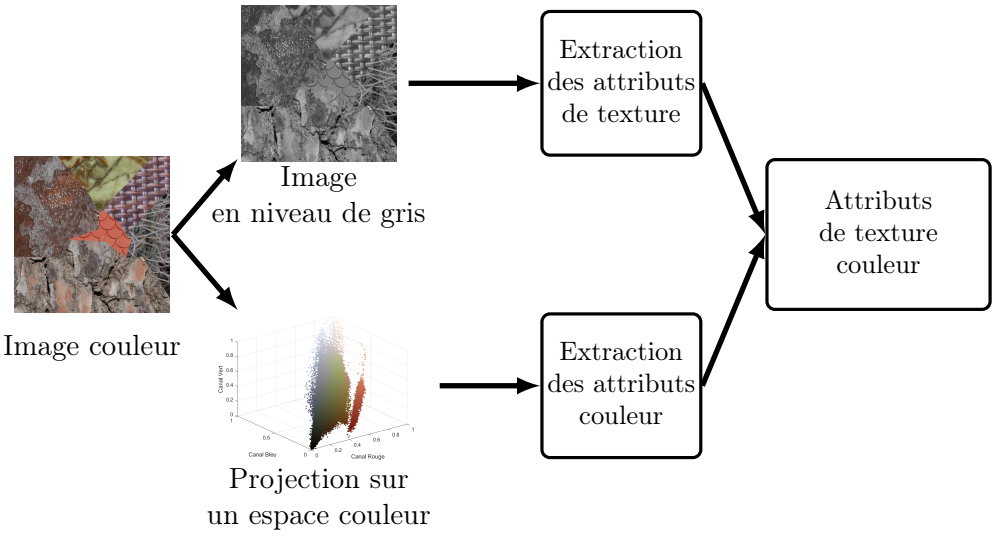


FIG. 1.9: Approche parallèle d'extraction d'attributs de texture couleur.

en considération l'interaction spatiale entre les niveaux des autres composantes de couleur. Cette approche suppose que la texture peut être décrite indépendamment dans chaque canal de couleur, où les pixels sont caractérisés par une seule composante de couleur (Fig. 1.10). Dans cette stratégie, les attributs de texture conçus pour les images en niveaux de gris sont calculés pour chaque canal de couleur, puis concaténés dans un seul vecteur d'attributs. Parmi les méthodes les plus populaires, on peut citer les histogrammes marginaux (Hist. RGB) [39], les spectres de texture (TS) [31], les matrices de co-occurrences [32, 5], les filtres de Gabor [32, 40, 33], la transformation en ondelettes (DT-CWT Dual Tree Complex Wavelet Transform) [41, 42], les motifs binaires locaux (LBP) [43] et ses variants CLBP (Completed Local Binary Patterns) [29], ELBP (Extended Local Binary Patterns) [44], et ILBP (Improved Local Binary Patterns) [30].

Stratégie opposée : La stratégie opposée prend en considération les relations inter-composantes et les attributs de texture sont obtenus en traitant conjointement des paires de canaux de couleur (Fig. 1.10). Dans [5], Palm a utilisé les matrices de co-occurrences (GLCM) à canaux multiples. Dans cette méthode, il a tenté d'incorporer l'information de corrélation entre les canaux en calculant la GLCM à partir de paires de canaux (LV, UV et LU). Dans [45], Maheswari et al. ont utilisé les attributs de Haralick extraits des matrices de co-occurrences de couleurs inter-composantes de l'espace de couleur HSI. Cette technique a été reprise dans [46].

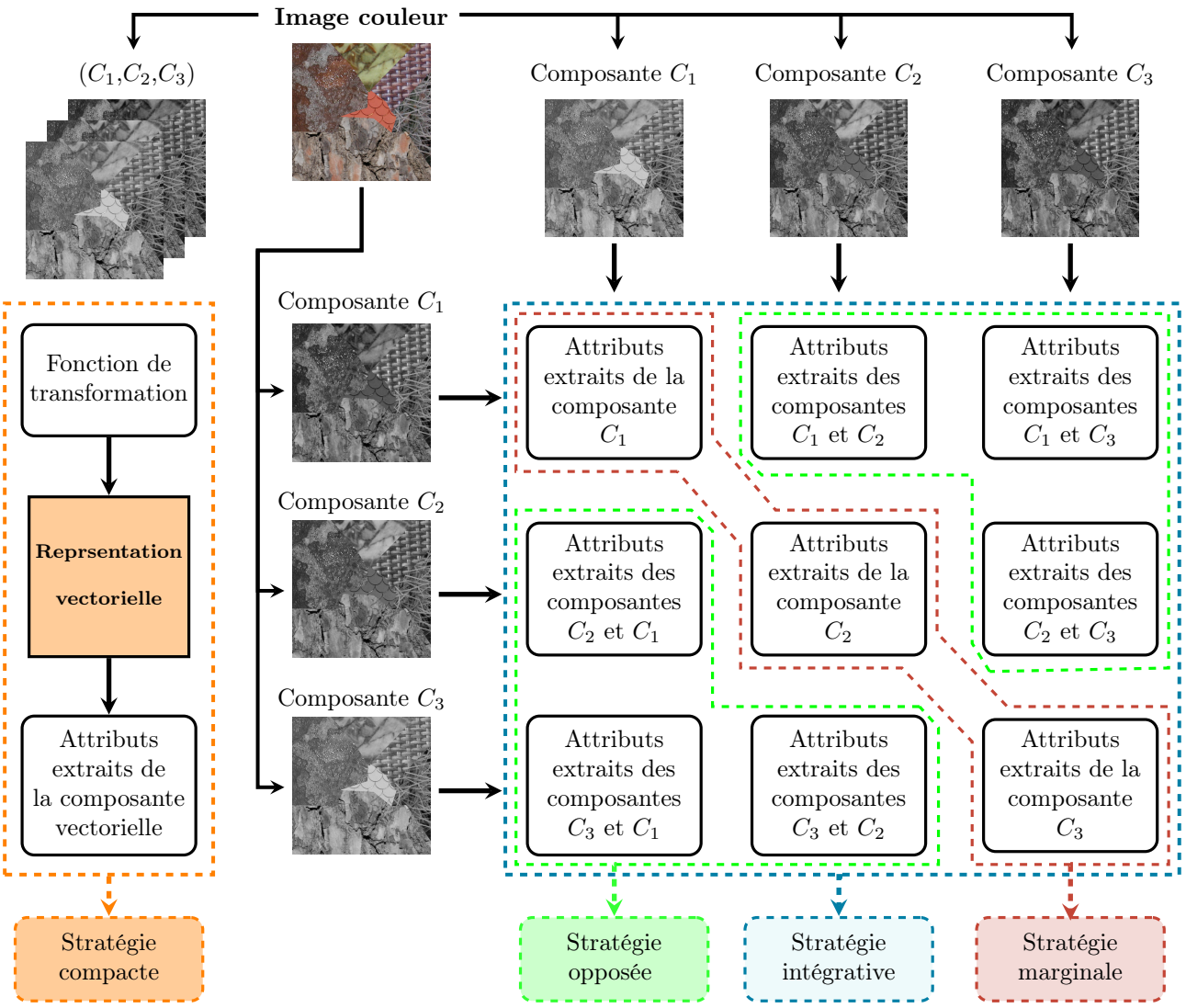


FIG. 1.10: Combinaison conjointe de la couleur et de la texture selon les stratégies: marginale (rouge), opposée (vert), intégrative (bleu) et compacte (orange).

Stratégie intégrative : La stratégie intégrative combine à la fois la stratégie marginale et la stratégie opposée. Souvent confondue avec la stratégie opposée, elle considère la distribution spatiale à la fois au sein de chaque composante couleur et également entre les différentes composantes (relations intra-composante et inter-composantes). Parmi les méthodes de cette stratégie, on peut citer le modèle MSAR (Multispectral Simultaneous Auto-Regressive model) [47], les modèles de champ aléatoire de Markov multi-spectraux proposés par Panjwani et al. [48], la méthode basée sur les filtres de Gabor (Opponent Gabor) proposée par Jain et al. [49], les matrices de co-occurrences chromatiques et les motifs locaux binaires couleurs.

Matrices de co-occurrences chromatiques : Les matrices de co-occurrences en niveaux de gris (GLCM) ont été étendues par Palm et al. aux images couleur sous l'appellation de matrices de co-occurrences intégratives (Integrative Co-occurrence Matrices, ICM) [5] et par Arvis sous le nom de matrices de co-occurrence généralisées [4]. Ces matrices de co-occurrences chromatiques sont composées de trois matrices intra-composantes (une pour chaque composante couleur) et de trois matrices inter-composantes (trois paires de composantes couleur), définies comme suit :

Soit $\mathbf{I}$ une image couleur définie sur une grille finie $\mathtt{S}$. Cette image $\mathbf{I}$ peut être décomposée en trois images composantes I^k, où $k \in \{C_1,C_2,C_3\}$. Soit $C^{k,k'}[\mathbf{I}](d,\theta)$ la matrice de co-occurrence chromatique qui mesure les interactions spatiales entre les composantes couleurs k et k' des pixels de l'image $\mathbf{I}$ se trouvant à une distance spatiale d l'un de l'autre selon la direction θ. Un élément $C^{k,k'}[\mathbf{I}](d,\theta)(g,g')$ de cette matrice, où $(g,g') \in \{0,\ldots,q-1\}^2$, contient le nombre de fois qu'un pixel $\mathbf{s}$ dont le niveau de composante couleur $I^k(\mathbf{s})$ est égal à g possède dans son voisinage un autre pixel $\mathbf{r}$ séparé d'un vecteur de translation $\mathbf{d}$ et dont le niveau de la composante couleur $I^{k'}(\mathbf{r})$ est égal à g' :

$$C^{k,k'}[\mathbf{I}](d,\theta)(g,g') = \#\{(\mathbf{s},\mathbf{r}) \in \mathtt{S}^2 \,|\, \mathbf{r} = \mathbf{s} + \mathbf{d}, I^k(\mathbf{s}) = g, I^{k'}(\mathbf{r}) = g'\} \tag{1.5}$$

Le vecteur de distance $\mathbf{d}$ étant défini par une distance spatiale d et une orientation θ.
Il en découle dans le cas des images RGB, trois matrices intra-composantes : $C^{R,R}[\mathbf{I}](d,\theta)$, $C^{G,G}[\mathbf{I}](d,\theta)$, $C^{B,B}[\mathbf{I}](d,\theta)$, et trois matrices inter-composantes $C^{R,G}[\mathbf{I}](d,\theta)$, $C^{R,B}[\mathbf{I}](d,\theta)$, $C^{G,B}[\mathbf{I}](d,\theta)$ [4].

Dans [50] Vadivel et al. ont proposé une Matrice de Co-occurrence Intégrée Couleur-Intensité (ICICM) qui regroupe à la fois les informations de couleur et d'intensité d'une image texturée. Elle est composée de quatre sous-matrices, comme suit :

$$ICICM = \begin{bmatrix} ICICMcc & ICICMcg \\ ICICMgc & ICICMgg \end{bmatrix} \tag{1.6}$$

La matrice ICICMcc contient le nombre d'occurrences des paires couleur-couleur, ICICMcg les co-occurrences des paires couleur-gris, ICICMgc les co-occurrences des paires gris-couleur,

et ICICMgg les co-occurrences des paires gris-gris. Ces matrices de co-occurrences sont déterminées en définissant la couleur $\mathbf{c}$ et le niveau de gris g d'un pixel à partir de sa représentation dans l'espace HSI.

$$c = \begin{cases} S^{r_1 * (\frac{255}{I})^{r_2}} & \text{si } I \neq 0 \\ 0 & \text{si } I = 0 \end{cases} \tag{1.7}$$

et

$$g = 1 - \mathbf{c} \tag{1.8}$$

où S représente la saturation, I l'intensité, et r_1 et r_2 sont des constantes fixées empiriquement à 0.1 et 0.85. Il est à remarquer que la teinte n'est pas utilisée.

Motifs locaux binaires couleur : La méthode LBP a été étendue aux images couleur par Mäenpää et al. [51]. Dans cette méthode, appelée Opponent color LBP (OCLBP), un code LBP couleur est attribué à chaque pixel $\mathbf{s}$ dont la valeur de la $k^{\text{ème}}$ composante couleur est g^k, en fonction de la composante couleur k' de chacun de ses P pixels voisins situés dans un voisinage circulaire de rayon R tel que :

$$\text{LPB}^{k,k'}_{(P,R)}(\mathbf{s}) = \sum_{i=0}^{P-1} S(g_i^{k'} - g^k) \cdot 2^i \tag{1.9}$$

$S(.)$ étant la fonction signe définie par l'équation 1.3.
Chaque pixel $\mathbf{s}$ d'une image couleur RGB peut être caractérisée par trois LBP intra-composantes $LBP^{R,R}$, $LBP^{G,G}$, $LBP^{B,B}$ et six LBP inter-composantes $LBP^{R,G}$, $LBP^{R,B}$, $LBP^{G,B}$, $LBP^{G,R}$, $LBP^{B,R}$ et $LBP^{B,G}$. D'autres variantes de LBP telles que ILBP et CLBP, ont été également étendues aux images couleur. Ces deux méthodes sont respectivement référencées sous les noms IOCLBP (Improved opponent colour local binary patterns) [52] et MCLBP (Multiple Channels Local Binary Patterns) [53].

Stratégie compacte : La stratégie compacte exploite la représentation vectorielle des couleurs pour analyser les relations entre les couleurs des pixels voisins, comme dans le cas pour les images en niveaux de gris. Elle est particulièrement adaptée à la caractérisation des textures couleur en raison de sa prise en compte complète des corrélations entre les couleurs des pixels voisins tout en réduisant l'espace mémoire et le temps de calcul. Malgré cela,

très peu de méthodes ont adopté cette stratégie. La plupart d'entre elles sont basées sur les LBP [54, 55, 56, 57, 58, 59]. Dans [54, 55, 56], les auteurs représentent la couleur de chaque pixel par un vecteur et comparent les vecteurs de couleur des pixels voisins avec celui du pixel central. Ils utilisent pour cela une relation d'ordre partiel sur les couleurs, basée sur la distance euclidienne, pour comparer le rang des couleurs. Dans [57], Ledoux et al. explorent d'autres relations d'ordre et proposent de combiner deux ordres de couleurs dans un seul opérateur LBP, appelé Mixed Color Order LBP (MCOLBP). Dans [58], Lan et al. ont utilisé les quaternions pour représenter chaque couleur d'un pixel par un nombre complexe intégrant simultanément toutes les composantes de couleur. La méthode LBP appliquée à ces quaternions (Quaternionic Local Binary Pattern: QLBP) permet d'aboutir à un histogramme des QLBP de dimension équivalente à celle d'un LBP en niveaux de gris. La méthode appelée SWOBP (spatially weighted order binary pattern) décompose les différences de couleur locales en modèles binaires pondérés dans l'espace et les utilise pour coder les informations relatives à l'ordre des couleurs dans un voisinage local [59].

1.3 Texture couleur floue

Toutes les méthodes d'analyse de la couleur, de la texture en niveaux de gris et des textures couleur, citées précédemment, supposent que la couleur et la texture sont définies de manière précise. Or, en réalité, ni la couleur et encore moins la texture ne sont clairement définies. En effet, ces deux notions sont vagues, ambiguës et imprécises, car elles sont définies de manière subjective. Par exemple, une surface peut être considérée comme légèrement rugueuse pour une personne, tandis qu'une autre peut la juger comme lisse. Afin de prendre en compte toutes ces incertitudes, la théorie des ensembles flous a été appliquée pour analyser d'une manière plus efficace des images couleur texturées. Dans cette section, nous rappelons quelques notions sur les ensembles flous avant d'explorer leur utilisation pour l'analyse de la texture et de la texture couleur.

1.3.1 Couleur floue

La couleur telle que nous l'avons définie à la section 1.2.1.1 est sujette à deux problèmes d'incertitude [60]. Le premier concerne la représentation numérique de la couleur qui ne correspond pas à la manière dont les humains comprennent ce concept. Ces derniers ont l'habitude de décrire les couleurs à l'aide des catégories telles que le vert, le jaune, le noir, etc., tandis que les ordinateurs utilisent les trois composantes couleur pour représenter les couleurs. De plus, la perception des couleurs est subjective, car elle peut varier d'une personne à une autre. Par exemple, la couleur des objets dans la figure 1.11 peut appartenir à la catégorie des "rouges" avec un certain degré. Un objet peut être considéré comme "rouge" à 70 % et "orange" à 30 %. Cela reflète mieux la manière dont les humains perçoivent les couleurs, où les nuances peuvent varier d'une personne à l'autre et selon les conditions d'éclairage [61, 62, 60].

FIG. 1.11: Exemples d'objets de couleur rouge.

Ensuite, les couleurs sont généralement imprécises, car il est difficile de délimiter clairement les frontières entre elles [62]. En général, la perception des couleurs peut être considérée comme un concept graduel au sein de l'ensemble des couleurs, où la frontière entre ces couleurs n'est pas nette. Prenons l'exemple de la figure 1.12, qui est composée d'un dégradé linéaire entre les couleurs. Cette figure illustre la nature ambiguë de la couleur, car il n'y a pas de frontière précise séparant les deux couleurs, mais la frontière est floue.

Pour tenir compte de ses imprécisions, Kay et al. [61] ont considéré les catégories de couleur comme des ensembles flous. Chaque ensemble flou contient une partie de l'espace de couleurs nettes avec des frontières floues, définissant ainsi un ensemble de couleur floue [63].

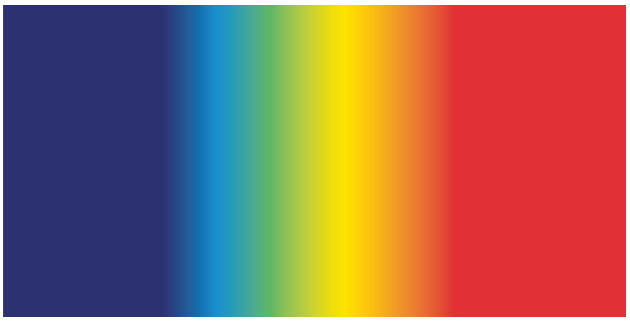


FIG. 1.12: Exemple d'un dégradé linéaire entre les couleurs.

Deux approches sont proposées pour définir des ensembles de couleur flous [64]. La première consiste à définir les ensembles flous unidimensionnels [63]. Cette catégorie permet de traiter chaque composante couleur de manière séparée. En général, ces partitions sont définies par des fonctions d'appartenance, telles que la fonction gaussienne ou triangulaire (voir section 1.3.3 du Chapitre 2). La seconde approche repose sur la définition de partitions floues tridimensionnelles dans le domaine de l'espace couleur [60]. Dans ce cas, chaque couleur est représentée par un point dans un espace à trois dimensions, où les axes correspondent aux différentes composantes couleur. Cette approche permet de prendre en compte simultanément les trois composantes. Le degré d'appartenance de toute couleur peut être calculé en utilisant la distance euclidienne entre un point représentant une couleur nette (ou référence) et un point représentant une couleur floue (voir section 3.3.1 du Chapitre 3).

Pour caractériser des couleurs floues, les histogrammes des couleurs floues sont souvent employés [65].

1.3.2 Notion de logique floue

La logique floue est une branche des mathématiques appartenant à l'algèbre abstraite, développée par Lotfi Zadeh en 1965 [66]. Cette notion permet d'étendre la logique classique, associée aux variables booléennes ne prenant que deux valeurs 0 et 1. Elle consiste à représenter mathématiquement la notion d'ensemble dans le cadre où les frontières entre les éléments d'un ensemble sont floues ou imprécises.

Définition d'un sous-ensemble flou

Soit X un espace de points, appelé ensemble de référence ou univers. Soit x un élément quelconque de X. Un sous-ensemble flou A est une partie de cet ensemble, caractérisée par une fonction d'appartenance $\mu_A(x)$, qui attribue à chaque élément x de l'ensemble X un nombre réel dans l'intervalle [0,1]. Un sous-ensemble flou A de X est défini comme suit [66] :

$$A = \{(x,\mu_A(x)) \mid x \in X\} \quad \text{avec } \mu_A : X \to [0,1], \forall x \in X \tag{1.10}$$

La valeur $\mu_A(x)$ représente le degré d'appartenance de x au sous-ensemble A.

Contrairement à la théorie des ensembles classiques (ou nets), où un élément appartient strictement soit à un ensemble, soit à son complémentaire, dans la théorie des ensembles flous, un élément peut appartenir à un ensemble avec une certaine mesure variant entre 0 et 1. Ces valeurs indiquent le degré d'appartenance de cet élément à cet ensemble. Ainsi, un élément peut appartenir fortement à un ensemble, avec une valeur proche de 1, faiblement, avec une valeur proche de 0, ou bien il peut avoir une appartenance absolue si elle est égale à 1, ou une non-appartenance absolue si elle est égale à 0. Dans ce cas, un sous-ensemble ordinaire de X, appelé aussi ensemble net, peut être considéré comme un cas particulier de ses sous-ensembles flous pour lesquels le degré d'appartenance est limité à {0,1}.

Les notions d'égalité, d'inclusion, d'union et d'intersection sont étendues à de tels ensembles, et diverses propriétés de ces notions dans le contexte des ensembles flous sont décrites dans l'annexe A.

La théorie des ensembles flous offre une alternative à la logique classique pour modéliser les phénomènes naturels complexes. Les concepts mathématiques flous ont été largement appliqués aux problèmes d'analyse d'images [67], démontrant leur supériorité en termes de performances grâce à leur capacité à traiter l'incertitude des images numériques. Des techniques de traitement d'image basées sur des concepts flous ont alors été proposées afin de prendre en compte toutes ces imprécisions [68, 69, 70].

La section suivante rappelle certaines notions sur les ensembles flous et les nombres flous, qui sont utilisés pour définir des ensembles des niveaux de gris flous d'une image.

1.3.3 Ensemble de niveau de gris flou

Soit une image en niveaux de gris I définie sur une grille $\mathtt{S}$ et quantifiée avec q niveaux, et soit $S_g \subseteq \mathtt{S}$ l'ensemble de niveau de gris g tel que, $0 \leq g \leq q-1$. Dans la théorie des ensembles flous, la grille $\mathtt{S}$ de tous les pixels est appelée ensemble de référence ou univers et les sous-ensembles ordinaires de $\mathtt{S}$ sont appelés ensembles nets. On définit alors le sous ensemble flou de niveau de gris $S_g \subseteq \mathtt{S}$ par sa fonction d'appartenance μ_{S_g}. $\mu_{S_g}(\mathbf{s})$ définit le degré d'appartenance de chaque pixel $\mathbf{s} \in \mathtt{S}$ au sous ensemble S_g, tel que $0 \leq \mu_{S_g}(\mathbf{s}) \leq 1$. Par ailleurs, nous considérons les niveaux de gris comme des valeurs floues. Un niveau de gris flou $\tilde{g}$ est caractérisé par sa fonction d'appartenance $\mu_{\tilde{g}} : [0,\ldots,q-1] \rightarrow [0,1]$. Cette fonction peut être définie de plusieurs manières [71] et prend souvent la forme $\mu_{\tilde{g}}(x) = f\left(|x-g|\right)$, où f est une fonction qui contrôle la forme du nombre flou et g est l'équivalent net du nombre flou $\tilde{g}$. Les fonctions d'appartenance les plus utilisées sont:

$$\text{Nette:} \quad \mu_{\tilde{g}}(x) = \begin{cases} 1 & \text{si } |x-g| \leq \delta, \\ 0 & \text{sinon.} \end{cases} \tag{1.11}$$

$$\text{Gaussienne:} \quad \mu_{\tilde{g}}(x) = \exp\left(-\tfrac{(x-g)^2}{2\alpha^2}\right) \tag{1.12}$$

$$\text{Triangulaire:} \quad \mu_{\tilde{g}}(x) = \max\left(0{,}1 - \tfrac{|x-g|}{\beta}\right) \tag{1.13}$$

$$\text{Bell généralisée :} \quad \mu_{\tilde{g}}(x) = \tfrac{1}{1+|\frac{x-g}{\alpha}|^{2\beta}} \tag{1.14}$$

$$\text{Trapézoïdale:} \quad \mu_{\tilde{g}}(x) = \max\left(0, \min\left(\tfrac{x-(g-a)}{b+a}, 1, \tfrac{(g+a)-x}{b-a}\right)\right) \tag{1.15}$$

Ces fonctions sont symétriques par rapport à l'équivalent net g du niveau flou $\tilde{g}$ et sont paramétrées par des constantes réelles positives α, β, δ, a et b ayant pour rôle de contrôler la portée du niveau flou. La fonction floue triangulaire a été utilisée, par exemple, par Jawahar et Ray pour la représentation des textures [72]. Son degré d'appartenance vaut 1 à $x = g$, diminue lorsque la différence entre x et g augmente, et s'annule lorsque

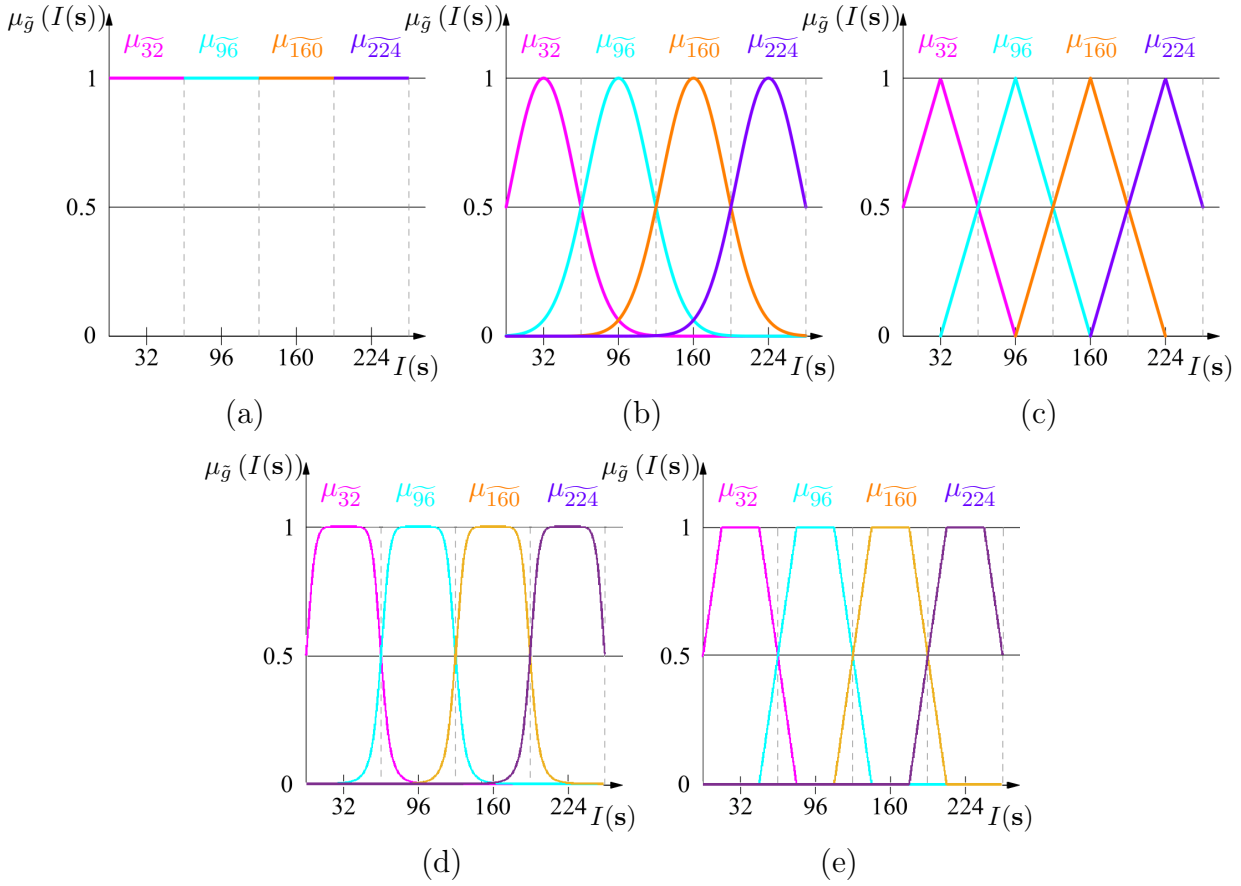


FIG. 1.13: Fonction d'appartenances $\mu_{\tilde{g}}$, $g \in \Lambda$, pour 4 intervalles lorsque $G = 256$: (a) Nette, (b) Gaussienne, (c) Triangulaire, (d) Bell généralisée et (e) Trapézoïdale.

$|x - g| \geq \beta$. La figure (Fig. 1.13) montre l'allure de quelques-unes de ces fonctions d'appartenance.

Le degré d'appartenance de chaque pixel $\mathbf{s}$ à l'ensemble flou S_g est le degré d'appartenance $\mu_{\tilde{g}}(I(\mathbf{s}))$ du niveau de gris $I(\mathbf{s})$ au nombre flou $\tilde{g}$:

$$\mu_{S_g}(\mathbf{s}) = \mu_{\tilde{g}}(I(\mathbf{s})) \tag{1.16}$$

1.3.4 Texture floue

Comme nous l'avons mentionné précédemment, la texture est par définition une caractéristique floue et subjective, et l'hypothèse selon laquelle les images de texture sont principalement représentées par des répétitions spatiales d'un motif pourrait ne plus être valable, surtout pour les textures naturelles. De plus, les images en niveaux de gris ou en couleur

peuvent présenter un certain degré de flou en raison de l'imprécision inhérente à la discrétisation à la fois du domaine spatial (échantillonnage) et au niveau des composantes de couleur (quantification). Par conséquent, les frontières qui séparent les différentes régions de l'image ne sont pas définies de manière précise, et les niveaux des pixels sont des mesures imprécises des réflexions émises par les surfaces observées par la caméra. Face à ces contraintes, des techniques d'analyse de texture basées sur la théorie des ensembles flous ont été élaborées. Il s'agit par exemple, des histogrammes des niveaux de gris flous [72], du spectre de texture floue (Fuzzy Texture Spectrum, FTS) [73], des motifs binaires flous [74, 70], des motifs locaux flous (Local Fuzzy Pattern, LFP) [75], des matrices de co-occurrences des niveaux de gris flous [72, 76], et des matrices aura des niveaux de gris flous [8]. Ces dernières sont abordées dans le prochain chapitre.

1.3.4.1 Matrice de co-occurrence des niveaux de gris flous

Les matrices de co-occurrences des niveaux de gris ont été également étendues au concept flou. Jawahar et Ray [72] ont défini la matrice de co-occurrence des niveaux de gris flous (Fuzzy Gray Level Co-occurrence Matrice, FGLCM) comme une matrice $\tilde{C}(d,\theta) = [\tilde{c}^{\mathbf{d}}(g,g')]$ de taille $(q \times q)$, dont l'élément $\tilde{c}^{\mathbf{d}}(g,g')$ est égal à la somme des minimums entre le degré d'appartenance des niveaux de gris de chaque pixel $\mathbf{s}$ au niveau de gris flou $\tilde{g}$ et le degré d'appartenance du niveau de gris de son pixel voisin au niveau de gris flou $\tilde{g}'$. Elle peut être exprimée comme suit :

$$\tilde{c}^{\mathbf{d}}(g,g') = \sum_{\mathbf{s}\in\mathbf{S}} \min\left\{\mu_{\tilde{g}}\left(I(\mathbf{s})\right), \mu_{\tilde{g}'}\left(I(\mathbf{s}+\mathbf{d})\right)\right\} \ . \tag{1.17}$$

où le vecteur distance $\mathbf{d}$ entre $\mathbf{s}$ et son voisin $\mathbf{r} = \mathbf{s} + \mathbf{d}$ est défini par une distance spatiale d et selon une orientation θ.

Munklang et al. [76] ont proposé une autre matrice de co-occurrence des niveaux de gris flous, notée, F_zCM, telle que:

$$\tilde{c}^{\mathbf{d}}(g,g') = \sum_{\mathbf{s}\in\mathbf{S}} \mu_{\tilde{g}}\left(I(\mathbf{s})\right) + \mu_{\tilde{g}'}\left(I(\mathbf{s}+\mathbf{d})\right) \ . \tag{1.18}$$

1.3.4.2 Motifs locaux binaires flous

La méthode Fuzzy Local Binary Pattern (FLBP) est une extension de la technique LBP à la logique floue. Elle a été proposée par Dimitris et Keramidas dans [74, 70]. Cette méthode suppose qu'un voisinage local peut être partiellement caractérisé par plusieurs motifs binaires en raison de l'incertitude des valeurs des niveaux de gris des pixels. Formellement, la FLBP peut être exprimée comme suit :

Soient B un ensemble des pixels voisins ayant un niveau de gris g_i supérieur ou égal à celui du pixel référence g, et A un ensemble des pixels voisins avec un niveau de gris g_i inférieur à celui du pixel référence g. Leurs équivalents flous $\tilde{B}$ et $\tilde{A}$ sont définis par leurs degrés d'appartenance des P pixels voisins $\mathbf{r}_i(i = 1,...,P)$ tels que:

$$\tilde{B} \equiv \{\langle \mathbf{r}_i, \mu_{\tilde{B}}(i)\rangle\} \quad \text{et} \quad \tilde{A} \equiv \{\langle \mathbf{r}_i, \mu_{\tilde{A}}(i)\rangle\} \tag{1.19}$$

Le degré d'appartenance $\mu_{\tilde{B}}(i)$ d'un pixel voisin $\mathbf{r}_i$ au sous-ensemble flou $\tilde{B}$ est défini par la fonction d'appartenance suivante:

$$\mu_{\tilde{B}}(i) = \begin{cases} 1 & \text{si} \quad g_i - g \geq \tau, \\ \frac{\tau + g_i - g}{2\tau} & \text{si } |g_i - g| < \tau, \quad \tau \neq 0 \\ 0 & \text{si } g_i - g \leq -\tau, \quad \tau \neq 0 \\ 0 & \text{si } g_i - g < \tau, \quad \tau = 0 \end{cases} \tag{1.20}$$

$\tau \in [0,q]$ représente un paramètre qui contrôle le degré de flou et q le niveau de gris maximal. Le degré d'appartenance $\mu_{\tilde{A}}(i)$ des pixels voisins $\mathbf{r}_i$ à l'ensemble flou $\tilde{A}$ peut être déduit de $\mu_{\tilde{B}}(i)$:

$$\mu_{\tilde{A}}(i) = 1 - \mu_{\tilde{B}}(i) \tag{1.21}$$

Le code FLBP est donné par la formule suivante :

$$FLBP(\mathbf{s}) = \prod_{i=0}^{n-1} \mu_Z(i) \tag{1.22}$$

Z est une variable, indiquant le sous-ensemble flou $\tilde{A}$ ou $\tilde{B}$, défini en fonction du signe de chaque pixel voisin $\mathbf{r}_i$ de $\mathbf{s}$ (voir l'équation (1.3), comme suit :

$$Z \equiv \begin{cases} \tilde{B} & \text{si } S(g_i - g) = 1 \\ \tilde{A} & \text{si } S(g_i - g) = 0 \end{cases} \tag{1.23}$$

1.3.5 Attributs de texture couleur floue

Les méthodes d'analyse de texture initialement conçues pour les images en niveaux de gris flous peuvent être facilement étendues aux images couleur texturées floues selon la stratégie marginale. C'est le cas des méthodes FLBP [74, 70], LFP [75], matrices aura [8]. Les matrices aura seront abordées dans le chapitre suivant. D'autres méthodes compactes comme les histogrammes des couleurs floues (Fuzzy Color Histogram) [77, 65], les correlogrammes des couleurs floues [78], les caractéristiques profondes floues (Fuzzy Deep features) [79], les matrices de co-occurrences des couleurs floues [6, 80], et l'entropie floue [81] sont proposées dans la littérature. Ces deux dernières sont décrites ci-après.

1.3.5.1 Matrices de co-occurrences des couleurs floues

Au lieu de caractériser une texture couleur par au moins trois matrices de co-occurrences floues (une par canal de couleur), Ledoux et al. [6] ont proposé d'étendre les matrices de co-occurrences des niveaux de gris (FGLCM) aux images couleur en introduisant des ensembles de couleurs flous $\mathtt{S}_{\mathbf{c}}$. Ainsi, une image couleur peut être représentée par une seule matrice de co-occurrence des couleurs floues (Fuzzy Color Co-occurrence Matrix, FCCM), qui caractérise les interactions locales entre les couleurs des pixels voisins. Cette matrice est définie comme suit:

$$\tilde{C}[\mathbf{I}](\mathbf{c},\mathbf{c}') = \sum_{\mathbf{r}\in\mathtt{S}} \sum_{\mathbf{s}\in\mathtt{S}} \min\left(\mu_{\mathtt{S}_{\mathbf{c}}}(\mathbf{s}),\mu_{\mathtt{S}_{\mathbf{c}'}}(\mathbf{r})\right) \ . \tag{1.24}$$

La matrice de co-occurrence intégrée modifiée de l'intensité et de la couleur (MICICM), proposée dans [80], est calculée en comptabilisant les co-occurrences entre les composantes de gris g et de couleur $\mathbf{c}$ de chaque pixel (Equations 1.6, 1.7 et 1.8), pondérés par des degrés

d'appartenance différents afin de prendre en considération l'aspect flou des couleurs et des niveaux de gris.

1.3.5.2 Entropie floue

Cette méthode basée sur l'entropie floue, a été proposée par Mirvana et al. [81] pour analyser les images de textures couleur. Soit $\mathbf{I}$ une image couleur de taille $H \times L$ définie par ses trois images composantes couleur I^k, $k \in \{C_1,C_2,C_3\}$. Le calcul de la mesure d'entropie floue pour chaque image composante couleur I^k se fait comme suit :
Soit $\mathtt{W}^m_{i,j,k}$ un patch de taille m^2, défini pour chaque image composante couleur k au pixel de coordonnées (i,j) avec $(1 \leq i \leq H-m$ et $1 \leq j \leq L-m)$:

$$\mathtt{W}^m_{i,j} = \begin{bmatrix} I^k(i,j) & \cdots & I^k(i,j+m-1) \\ I^k(i+1,j) & \cdots & I^k(i+1,j+m-1) \\ \cdots & \cdots & \cdots \\ I^k(i+m-1,j) & \cdots & I^k(i+m-1,j+m-1) \end{bmatrix}.$$

Le nombre total de patchs définis pour chaque composante k est donc $N_m = (L-m)(H-m)$. Soit $d^m_{ij,ab,k}$ la distance entre le patch $\mathtt{W}^m_{i,j,k}$ et ses patchs voisins $\mathtt{W}^m_{a,b,k}$ dans une même composante couleur k telle que:

$$\begin{aligned} d^m_{ij,ab,k} &= d[\mathtt{W}^m_{i,j,k},\mathtt{W}^m_{a,b,k}] \\ &= \max_{k,l\in[0,m-1]}(|I^k(i+k,j+l) - I^k(a+k,b+l)|) \end{aligned} \tag{1.25}$$

Le degré de similarité de chaque patch avec ses patchs voisins est défini par la fonction Gaussienne suivante :

$$D^m_{ij,ab,k}(r) = \mu(d^m_{ij,ab,k},r) = \exp\left(-\frac{(d^m_{ij,ab,k})^2}{r}\right). \tag{1.26}$$

r est un paramètre qui définit l'étendu du flou.
Les degrés de similarité dans chaque patch sont moyennés pour obtenir:

$$\phi^m_{i,j,k}(r) = \frac{1}{N_m - 1} \sum_{a=1,b=1}^{a=H-m,b=L-m} D^m_{ij,ab,k}(r) \tag{1.27}$$

avec $(a,b) \leq (i,j)$. Puis:

$$\phi_k^m(r) = \frac{1}{N_m} \sum_{i=1,j=1}^{i=H-m,j=L-m} \phi_{i,j,k}^m(r). \tag{1.28}$$

Finalement, l'entropie floue couleur $FuzEnC_{2D}^k$ de chaque image composante I^k est calculée comme suit:

$$FuzEnC_{2D}^k(m,r,I^k) = \ln\left(\frac{\phi_k^m(r)}{\phi_k^{m+1}(r)}\right). \tag{1.29}$$

En concaténant les trois entropies floues $FuzEnC_{2D}^k$ des composantes, nous obtenons l'entropie floue $FuzEnC_{2D}(m,r,I)$ de l'image couleur entière :

$$FuzEnC_{2D}(m,r,I) = [FuzEnC_{2D}^{C_1}, FuzEnC_{2D}^{C_2}, FuzEnC_{2D}^{C_3}]. \tag{1.30}$$

Cette méthode marginale a été reformulée suivant la stratégie compacte en considérant les patchs $\mathtt{W}_{i,j,k}^m$ et $\mathtt{W}_{i,j,k}^{m+1}$ comme des cubes d'arêtes respectives m et $m+1$. Le nombre total des cubes pouvant être générés à partir de l'image $\mathbf{I}$ est donc $N_m = (L-m)(H-m)(K-m)$. K étant le nombre de composantes couleur ($K = 3$). En suivant les mêmes étapes que $FuzEnV_{2D}$ on obtient une seule valeur d'entropie floue bidimensionnelle multi-canaux $FuzEnC_{2D}^k$ de l'image couleur :

$$FuzEnV_{2D}(m,r,I) = \ln\left(\frac{\phi^m(r)}{\phi^{m+1}(r)}\right). \tag{1.31}$$

Après avoir examiné attentivement des méthodes d'extraction des attributs de textures couleur et des textures de couleur floues, nous aborderons la manière dont la caractérisation de la texture couleur s'intègre dans les deux applications, à savoir la classification et la segmentation d'images couleur texturée. Les attributs de texture décrits précédemment peuvent être extraits pour chaque pixel ou chaque image en fonction de l'application envisagée (classification ou segmentation).

1.4 Classification des textures couleur

La classification des textures couleur est une opération qui vise à identifier ou regrouper des textures en fonction de leurs caractéristiques de texture couleur. Ce processus joue un rôle

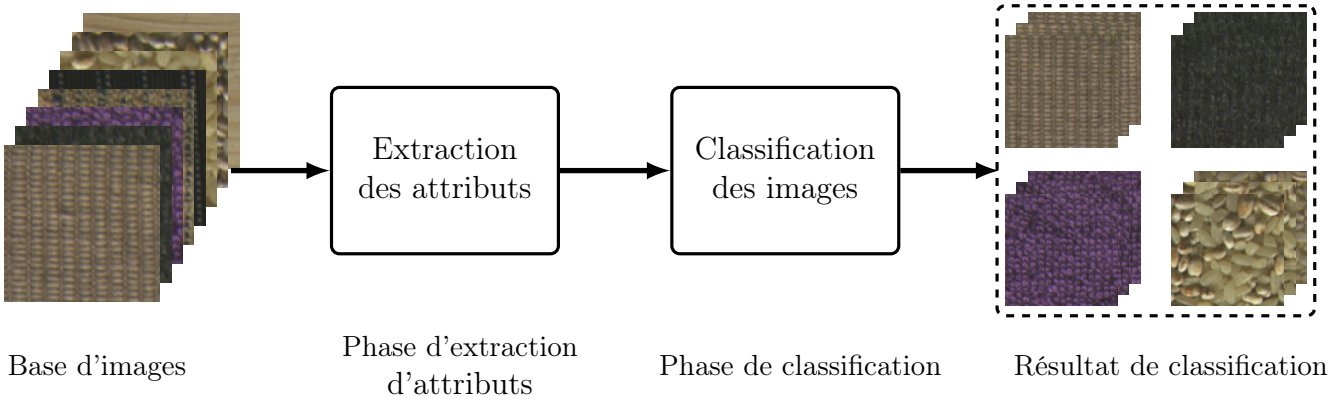


FIG. 1.14: Organigramme de la méthode de classification des textures couleur

fondamental dans des domaines très variés tels que l'imagerie biomédicale, l'inspection industrielle, l'analyse d'images satellites ou aériennes, l'analyse de documents, la reconnaissance faciale, la biométrie, et bien d'autres.

La classification des textures couleur s'appuie sur le principe d'apprentissage dans lequel une base d'images de textures couleur est créée afin d'entraîner un modèle pour chaque classe ou type de texture couleur. Elle comprend deux phases successives et distinctes (Figure 1.14). Dans la première phase, chaque image de la base est caractérisée par un ensemble d'attributs de texture couleur en utilisant les méthodes d'analyse présentées dans les sections précédentes. La seconde phase, celle de l'apprentissage, fait appel à des algorithmes de classification pour construire un modèle pour chaque classe de texture couleur à partir des attributs de texture couleur des images de la base d'apprentissage. Ces modèles seront ensuite utilisés lors de la phase de décision pour affecter une image inconnue (à identifier) à l'une des classes de texture couleur prédéfinies. Les méthodes de classification des textures couleur peuvent être scindées en deux catégories : méthodes traditionnelles ou par apprentissage classique (shallow learning) et les méthodes par apprentissage profond (deep learning).

1.4.1 Méthodes de classification traditionnelles

Dans cette catégorie, les méthodes de classification se distinguent par le contexte non supervisé, semi supervisé, ou supervisé. En classification supervisée, chaque image de la base d'apprentissage est caractérisée par un ensemble d'attributs de texture couleur, et l'appartenance de chaque image à une classe de texture est connue a priori. Ces images sont appelées

prototypes. Le processus de classification se déroule alors en deux étapes successives : l'apprentissage et la décision. Durant l'étape d'apprentissage, l'ensemble des prototypes est utilisé pour définir un modèle pour chaque classe de texture. Lors de l'étape de décision, la méthode d'extraction des attributs est appliquée à une image d'entrée inconnue. Une règle de décision liée au modèle du classifieur est ensuite appliquée pour étiqueter cette image inconnue dans l'une des classes en fonction de ses attributs. Plusieurs méthodes de classification supervisée sont proposées, les plus classiques sont les K plus proches voisins [82], les machines à vecteurs de support (SVM) [83], et les réseaux de neurones (MLP) [84]. Dans notre travail (Chapitre 3), nous avons exploité l'algorithme du plus proche voisin.

Dans un contexte non supervisé, on part du principe qu'il n'existe aucune connaissance préalable sur l'appartenance des images de la base à l'une des classes de texture couleur. Cette approche a pour but de regrouper les images en k classes, où le nombre de classes k peut être a priori connu ou inconnu. Pour atteindre cet objectif, plusieurs algorithmes peuvent être utilisés, parmi lesquels on peut citer l'algorithme K-means [85, 86].

Dans le contexte semi-supervisé ou faiblement supervisé, on considère qu'une partie des images de la base d'apprentissage est étiquetée, mais l'appartenance des autres images à des classes reste inconnue. Contrairement à la classification non supervisée, qui traite des images entièrement non étiquetées, la classification faiblement supervisée cherche à regrouper l'ensemble des données non étiquetées en k classes tout en exploitant les données étiquetées. Parmi les travaux répertoriés dans la littérature figurent la classification hiérarchique semi-supervisée, SVM transductive [87].

1.4.2 Méthodes par apprentissage profond

Dans les méthodes de classification traditionnelles, les textures couleur à classer sont caractérisées par un ensemble d'attributs par l'intermédiaire d'une technique d'extraction choisie au préalable. Les performances de la classification sont fortement dépendantes du choix des attributs. Pour surmonter cet inconvénient, des méthodes basées sur les réseaux

de neurones convolutifs (CNNs) ont été proposées ces dernières années. Elles ont l'avantage d'apprendre automatiquement les attributs discriminants durant la phase de l'entraînement. Les CNNs [88] sont composés d'une succession de plusieurs types de couches de neurones (couches convolutives, couches de sous-échantillonnages ou pooling) et en dernier des couches de neurones complètement connectées avec une couche de sortie munie d'une fonction softmax qui a pour rôle de prédire les probabilités des classes de textures. Les couches convolutives appliquent des filtres (kernels) pour extraire des motifs locaux, comme les contours, les textures fines et les variations de couleur. Les premières couches capturent des caractéristiques simples (bordures, couleurs dominantes), tandis que les couches plus profondes détectent des structures complexes. Le pooling réduit la résolution spatiale tout en conservant les informations clés. Cela aide à rendre le réseau invariant aux petites translations. Les couches complètement connectées jouent le rôle de classifieur. Les modèles de CNN les plus couramment utilisées dans la classification d'images sont: ResNet-50, VGG, GoogleNet et AlexNet [2, 3]. Les CNNs peuvent être principalement exploités de 2 façons. La première consiste à utiliser des modèles pré-entraînés (Transfert d'apprentissage) sur des bases générales, telle que ImageNet, comme extracteur d'attributs. La seconde consiste à ajuster les paramètres de ces modèles en utilisant une base d'images de textures couleur spécifique (fine tuning).

1.5 Segmentation d'images de textures couleur

La segmentation d'images couleur texturées est une procédure qui a pour but de partitionner une image numérique en régions homogènes, composées d'un ensemble de pixels connexes ayant principalement des caractéristiques communes en termes de couleur, de texture et éventuellement de taille ou de forme. Chaque région est censée représenter un objet ou une zone d'intérêt de la scène observée. La segmentation d'images couleurs texturées peut être abordée selon deux approches (contour et région). Les méthodes de type contour visent à détecter les zones de fortes discontinuités ou dissimilarités entre les caractéristiques de deux régions connexes.Tandis que l'approche région cherche à former des régions en regroupant des pixels ayant des caractéristiques similaires dans une même région. Ces deux approches se complètent bien qu'elles utilisent des formalismes différents. On s'est principalement intéressé

dans notre travail à l'approche région. D'un point de vue mathématique, la segmentation d'une image $\mathbf{I}$ en $K_{\mathbf{I}}$ régions [89] est définie par les conditions suivantes:

1. $R_i \neq \emptyset \quad \forall i$
2. $R_i \cap R_j = \emptyset \quad \forall i, j, ; i \neq j$
3. $\bigcup_{i=1}^{K_{\mathbf{I}}} R_i = \mathbf{I}$
4. R_i est une composante connexe $\quad \forall i$
5. $C(R_i) =$ Vrai $\quad \forall i$
6. $C(R_i \cup R_j) =$ Faux $\quad \forall i, j, i \neq j,$ et R_i adjacente à R_j

Ces conditions indiquent que chaque région R_i, $(i = 1,2,...,K_{\mathbf{I}})$ est définie comme un ensemble connexe de pixels partageant des caractéristiques communes évaluées par un critère d'uniformité noté $C(R_i)$. Ce critère distingue ces pixels de ceux des régions voisines en se basant sur des attributs de couleur, de texture, de texture couleur, etc.

Les approches présentées dans ce type de segmentation peuvent être regroupées en trois catégories distinctes : segmentation par analyse des caractéristiques spatiales, segmentation par classification des pixels, et segmentation d'images par apprentissage profond.

1.5.1 Segmentation par analyse des caractéristiques spatiales

Les méthodes de cette catégorie considèrent une région comme un ensemble de pixels connexes ayant des attributs de texture similaires. Dans cette approche, on distingue deux types de méthodes : par croissance de régions et par division-fusion de régions.

La segmentation par croissance de régions consiste à faire croître progressivement des régions en ajoutant successivement les pixels adjacents qui répondent à un critère ou plusieurs critères d'homogénéité. L'étape initiale commence par la sélection de "germes" qui correspondent généralement à un pixel ou plusieurs, puis les pixels connexes sont ajoutés successivement en respectant un critère de similarité pour construire les régions. Ce type d'approche a été utilisé pour la segmentation d'images de textures couleur par Chen et al. [90] et par Fondón et al. dans [91].

La segmentation par Division-Fusion de Régions se déroule en deux étapes. La première étape divise l'image en quatre régions de taille plus petite, et chaque région non homogène est subdivisée en quatre sous-régions encore plus petites. Ce processus de division est réitéré jusqu'à obtenir des régions homogènes. Dans la deuxième étape, les régions adjacentes qui ont été divisées et qui présentent des caractéristiques similaires sont fusionnées. Cette approche a été adoptée par Chen et Pavlidis en se basant principalement sur les paramètres extraits de la matrice de co-occurrence comme attributs de texture [92]. Doherty et al [93], appliquent l'algorithme division-fusion en considérant uniquement les niveaux de gris dans la première phase, puis fusionnent les régions formées dans la deuxième phase si leurs attributs de texture sont similaires. Dans le cas des images de textures couleur, le critère d'homogénéité repose sur le calcul des attributs de texture couleur. Cette approche a été appliquée dans [94, 95, 96, 97] pour réaliser la segmentation d'images de textures couleur.

1.5.2 Segmentation d'images par classification des pixels

Les méthodes de cette catégorie consistent à regrouper l'ensemble des pixels ou des régions (superpixels) de l'image, en différentes classes, en regroupant dans une même classe les pixels qui ont des caractéristiques similaires (texture, couleur, forme). Ce processus se déroule principalement en deux étapes. Dans la première étape, chaque pixel est caractérisé par un ensemble d'attributs de texture couleur. Les méthodes développées pour la classification des textures couleur sont utilisées pour cette étape. Dans la seconde étape, un algorithme de classification est appliqué pour regrouper ces pixels en régions homogènes en se basant sur les attributs de texture couleur de chaque pixel. Les méthodes de segmentation d'images par classification des pixels peuvent également être classées en trois catégories, selon le contexte supervisé, non supervisé, ou semi-supervisé [98, 99, 100]. Plusieurs méthodes de classificateurs supervisés sont utilisées pour la segmentation d'images couleur texturées [99, 101, 100]. Parmi elles, on trouve K-plus proches voisins (KNN) [102], le classifieur bayésien [103], la machine à vecteurs de support (SVM) [104], Random forest [36], les champs aléatoires de Markov [98], et les réseaux de neurones [105].

Cependant, en l'absence de connaissances préalables ou lorsque les étiquettes des classes ne sont pas connues, la segmentation non supervisée devient essentielle. Cette approche se base

uniquement sur les caractéristiques des pixels pour les regrouper. Plusieurs algorithmes sont utilisés pour effectuer ce type de segmentation, parmi lesquels l'algorithme de K-means reste le plus populaire [106, 107]. D'autres méthodes de segmentation de texture couleur reposent sur des techniques telles que la coupe de graphe (texNcut) [108], la méthode mean shift [109], l'estimation–maximization [99], le modèle aléatoire de Markov [98], les méthodes par apprentissage de dictionnaires basées sur la représentation parcimonieuse (dictionary learning and sparse representation-based classification (DLSRC)) [110], les méthodes par division fusion de régions (dynamic hierarchical classification (DHC) [97]), eCognition (eCog) [111], la méthode factorization-based texture segmentation (FSEG) [106] et le spectral clustering [112].

Quant à la segmentation d'images semi-supervisée, celle-ci est employée lorsque seule une connaissance partielle des étiquettes de classe est disponible. Elle se situe entre la classification non supervisée et la classification supervisée. Par exemple, dans [100], l'algorithme de regroupement spectral contraint a été appliqué pour la classification semi-supervisée de pixels caractérisés par des attributs de texture couleur. D'autres méthodes de segmentation semi-supervisée des images de couleur texturées sont proposées comme la version semi-supervisée de FSEG [113], la méthode weakly-supervised sparse coding geometric prior (WSSCGP) [113], et les méthodes basées sur la coupe de graphe (block-based graph cut (BGC) [114] et weakly-supervised graph cut (WSGC) [115]).

1.5.3 Segmentation d'images par apprentissage profond

La segmentation d'images basée sur l'apprentissage profond a été développée au cours de la dernière décennie [116, 117, 118]. Ces méthodes efficaces utilisent principalement les réseaux de neurones convolutionnels (CNN) spécifiques, tels que :

- Les réseaux Fully Convolutional Networks (FCN) qui remplacent les couches entièrement connectées des CNN classiques par des couches convolutionnelles afin de générer des cartes de segmentation pixel par pixel. Dans ce cas, les textures et couleurs sont prises en compte grâce à des couches convolutives profondes. Parmi ces réseaux, on trouve le supervised fully convolutional network for texture (FCNT) [116] et empirical wavelet transform-based fully convolutional network for texture (EWT-FCNT) [118].
- Le modèle U-Net [53], organisé en deux chemins: un chemin contractant (encoder) pour

extraire les caractéristiques importantes et un chemin expansif (decoder) pour reconstruire l'image segmentée.
- Les modèles deep visual model (DA) [119] et pyramid scene parsing network (PSP-Net) [120] qui utilisent des convolutions à trous (dilated convolutions) et des modules pyramidaux pour capturer des textures à différentes échelles.

1.6 Conclusion

L'analyse d'images de textures couleur est un sujet vaste et fondamental en vision par ordinateur, jouant un rôle significatif en classification et en segmentation d'images couleur texturées. Ces deux opérations interviennent souvent dans diverses applications. L'analyse des images consiste à décrire localement (cas des pixels) ou globalement (cas des images), les propriétés des textures couleurs.

Nous avons passé en revue, dans ce chapitre, les principales techniques d'extraction d'attributs couleurs et attributs de texture. Puis, nous avons montré les différentes façons de les combiner afin de caractériser les textures couleurs. Par la suite, nous avons montré que les notions de couleur et de texture sont vagues et ambiguës, d'où la nécessité de les caractériser à l'aide de la théorie des ensembles flous. Ceci nous a amené à présenter quelques notions liées aux couleurs et textures floues et exposer quelques méthodes d'extraction d'attributs de textures couleurs floues.

Chapitre 2

Matrices aura des niveaux des composantes couleur flous

2.1 Introduction

Parmi les attributs de texture décrits dans le chapitre précédent, ceux dérivant des matrices de co-occurrences des niveaux de gris sont très populaires et ont été largement utilisés en classification et segmentation d'images en niveaux de gris ou en couleur dans les deux contextes crisp (net) et flou. On s'intéressera dans ce chapitre à une autre technique d'extraction d'attributs de textures, dénommée "matrices aura des niveaux de gris" (GLAM : Gray Level aura Matrix). Nous présenterons, en premier lieu, le fondement de cette méthode incluant les notions d'ensemble aura, de mesures aura et de matrices aura des niveaux de gris et toute autre information liée au concept aura. Par la suite, nous montrerons comment étendre ces notions aux images couleur, aux niveaux de gris flous et aux couleurs floues.

2.2 Matrice aura des images en niveaux de gris

Le concept aura a été introduit par Elfadel et Picard [7] dans le cadre de l'analyse et la synthèse des textures. Il s'appuie sur la théorie des ensembles pour définir un ensemble aura, une mesure aura et une matrice aura des niveaux de gris (GLAM).

2.2.1 Ensemble aura des niveaux de gris

On considère $\mathbf{I}$ une image en niveaux de gris définie sur une grille finie $\mathtt{S}$, composée d'un ensemble de pixels $\mathbf{s}$, où chaque pixel $\mathbf{s} \in \mathtt{S}$ est défini par ses coordonnées (x,y) sur la grille et son niveau de gris $I(\mathbf{s})$. On définit sur la grille $\mathtt{S}$, un système de voisinage $\mathscr{N} = \{\mathtt{N}_\mathbf{s}, \mathbf{s} \in \mathtt{S}\}$, où $\mathtt{N}_\mathbf{s}$ est l'ensemble des pixels voisins du pixel $\mathbf{s}$. Soit q le niveau de gris maximal dans l'image et g le niveau de gris du pixel $\mathbf{s}$, tel que $g \in \Lambda = \{0,1,\ldots,q-1\}$.

Nous appelons *ensemble de niveau de gris* g, l'ensemble $S_g \subseteq \mathtt{S}$ des pixels ayant un niveau de gris donné g, $0 \leq g \leq q-1$, tel que $S_g = \{\mathbf{s} \mid \mathbf{s} \in \mathtt{S},\ I(\mathbf{s}) = g\}$. Ainsi, $\{S_g,\ 0 \leq g \leq q-1\}$ peut être considéré comme une partition de $\mathtt{S}$, telle que: $\bigcup_{g=0}^{q-1} S_g = \mathtt{S}$ et $S_g \cap S_{g'} = \emptyset$ pour $g \neq g'$.

L'aura de S_g par rapport à $S_{g'}$, notée $\mathtt{A}_{S_{g'}}(S_g)$, est définie comme suit [7]:

$$\mathtt{A}_{S_{g'}}(S_g) = \bigcup_{\mathbf{s} \in S_g} (\mathtt{N}_\mathbf{s} \cap S_{g'}), \tag{2.1}$$

L'ensemble aura $\mathtt{A}_{S_{g'}}(S_g)$ décrit la présence d'un sous-ensemble $S_{g'}$ dans le voisinage d'un autre sous-ensemble S_g.

L'ensemble aura dépend de la structure de voisinage et plus particulièrement du voisinage $\mathtt{N}_\mathbf{s}$. Ce voisinage peut être spatialement variable, c'est-à-dire différent d'un pixel à autre, symétrique ou non, de forme et de taille variables. Si $\mathtt{N}_\mathbf{s}$ est symétrique, l'ensemble aura sera alors formé par un anneau qui encercle S_g. C'est de là que le terme "aura" a été inspiré.

On peut définir $\mathtt{N}_\mathbf{s}$ par l'ensemble des pixels $\mathbf{r}$ les plus proches du pixel $\mathbf{s}$ au sens d'une distance D :

$$\mathtt{N}_\mathbf{s} = \{\mathbf{r} \in \mathtt{S} \mid 0 < D(\mathbf{s},\mathbf{r}) \leq d\}. \tag{2.2}$$

où $D(\mathbf{s},\mathbf{r})$ représente la distance entre le pixel $\mathbf{s}$ de coordonnées (x,y) et le pixel $\mathbf{r}$ de coordonnées (k,l) et d un seuil définissant l'étendue du voisinage. Plusieurs types de distances peuvent être définis. Les plus couramment utilisées sont :

Distance de Manhattan (City block) :

$$D_1(\mathbf{s},\mathbf{r}) = \|\mathbf{r}-\mathbf{s}\|_1 = |k-x| + |l-y| \tag{2.3}$$

Distance euclidienne :

$$D_2(\mathbf{s},\mathbf{r}) = \|\mathbf{r}-\mathbf{s}\|_2 = \sqrt{(k-x)^2 + (l-y)^2} \tag{2.4}$$

Distance de l'échiquier (Tchebychev) :

$$D_\infty(s,r) = \|\mathbf{r}-\mathbf{s}\|_\infty = \max(|k-x|,|l-y|) \tag{2.5}$$

Pour illustrer cette notion d'ensemble aura, on considère sur la figure 2.1(a) une image en niveaux de gris définie sur une grille finie $\mathtt{S}$ de taille 5×5 et deux ensembles de niveaux de gris S_2 (encadrés par des losanges) et S_3 (encadrés par des carrés). Pour le voisinage $\mathtt{N_s}$ défini de manière identique en tout pixel $\mathbf{s}$ représenté sur la figure 2.1(d), l'ensemble aura $\mathtt{A}_{S_3}(S_2)$ est composé des pixels de S_3 encadrés (Fig. 2.1(b)), alors que l'ensemble aura $\mathtt{A}_{S_2}(S_3)$ est composé des pixels de S_2 entourés par les losanges (Fig. 2.1(e)).

À travers cet exemple, nous constatons que l'ensemble aura n'est pas symétrique ($\mathtt{A}_{S_{g'}}(S_g) \neq \mathtt{A}_{S_g}(S_{g'})$), car $\mathtt{A}_{S_2}(S_3)$ est différent de $\mathtt{A}_{S_3}(S_2)$. D'autres propriétés de l'ensemble aura sont énoncées dans [7].

2.2.2 Description morphologique d'un ensemble aura

Comme nous l'avons évoqué précédemment, l'ensemble aura dépend de la structure de voisinage choisie sur la grille. Celle-ci peut-être différente d'un pixel à un autre. Dans le cas où le voisinage $\mathtt{N_s}$ est défini d'une manière identique en tout pixel $\mathbf{s}$, il est considéré comme une version translatée d'un élément de base notée E et qu'on appelle élément structurant en termes de morphologie mathématique. L'analyse d'images à base de la morphologie mathématique consiste à déplacer un élément structurant E sur l'image et à étudier les interactions entre les objets de l'image et E [121]. L'un des intérêts de l'ensemble aura est son lien à la morphologie mathématique et plus exactement à l'opérateur de dilatation. Rappelons que

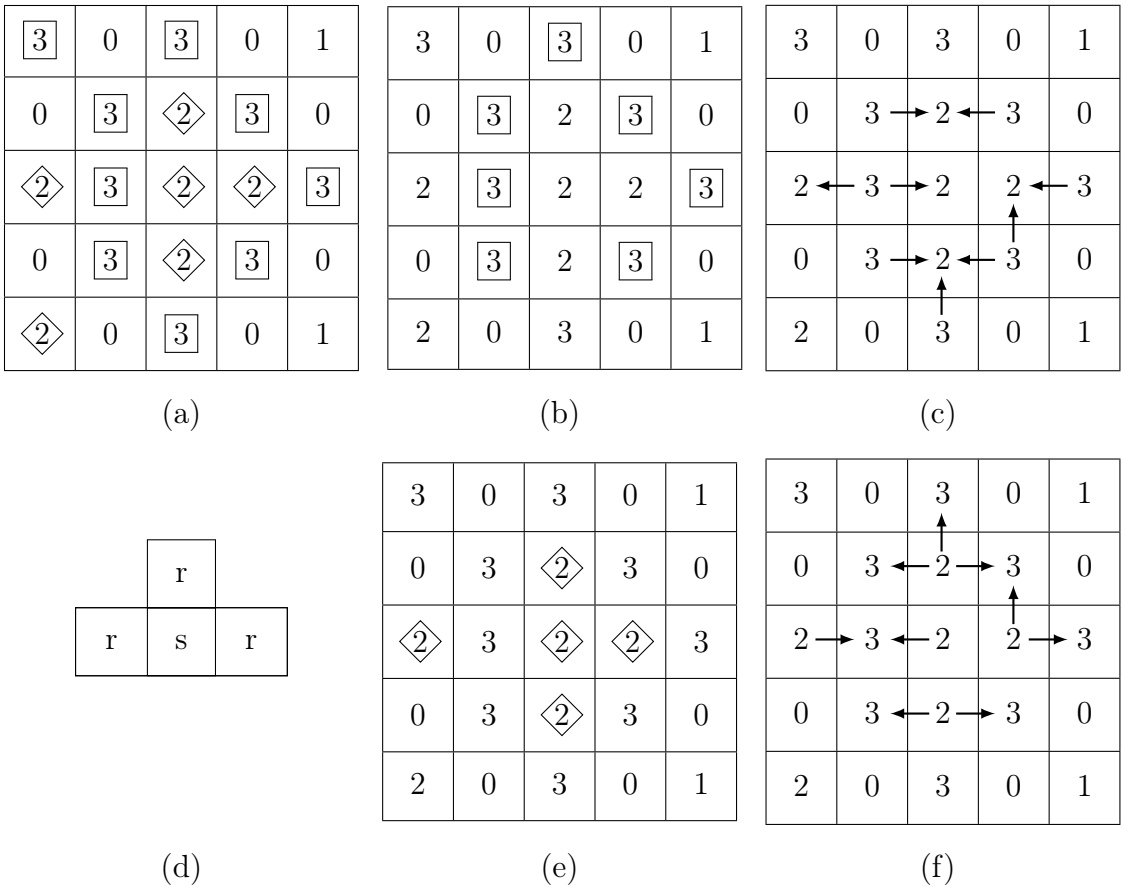


FIG. 2.1: Illustration de deux ensembles aura et de deux mesures aura: (a) Ensemble de niveaux de gris $\mathtt{S}_2$ (losanges) et $\mathtt{S}_3$ (carrés),(b) le cardinal de l'ensemble aura $\mathtt{A}_{\mathtt{S}_3}(\mathtt{S}_2) = 7$ (carrés) ,(c) Mesure aura locale $m_l\,(3{,}2) = 9$ (flèches),(d) Fonction de voisinage $\mathtt{N}_\mathbf{s}$,(e) le cardinal de l'ensemble aura $\mathtt{A}_{\mathtt{S}_2}(\mathtt{S}_3) = 4$ (losanges),(f) Mesure aura locale $m_l\,(2{,}3) = 9$ (flèches).

la dilatation d'un ensemble S_g par un élément structurant E est définie par [7]:

$$S_g \oplus E = \{\mathbf{s} \mid \mathtt{N}_\mathbf{s} \cap S_g \neq \emptyset\} = \bigcup_{t \in \mathtt{N}_\mathbf{s}} S_g^t \tag{2.6}$$

où $\mathtt{N}_\mathbf{s}$ correspond à l'élément structurant E centré sur le pixel $\mathbf{s}$. S_g^t est le translaté de l'ensemble S_g par le vecteur t.

Ainsi l'ensemble aura d'un ensemble de niveaux de gris S_g par rapport à un autre ensemble de niveaux de gris $S_{g'}$ peut être défini à l'aide de l'opérateur de dilatation morphologique

de base comme l'intersection entre $S_{g'}$ et S_g dilaté par E [7]:

$$\mathtt{A}_{S_{g'}}(S_g) = \bigcup_{\mathbf{s}\in S_g} (\mathtt{N}_\mathbf{s} \cap S_{g'}) = \left(\bigcup_{\mathbf{s}\in S_g} \mathtt{N}_\mathbf{s} \right) \cap S_{g'} = (S_g \oplus E) \cap S_{g'} \tag{2.7}$$

2.2.3 Mesure et matrice aura locale des niveaux de gris

Elfadel et Picard [7] définissent la mesure aura comme une fonction sur la grille $\mathtt{S}$ qui caractérise un ensemble aura, et la matrice aura des niveaux de gris pour englober toutes les mesures aura entre toutes les paires possibles d'ensembles de niveaux de gris dans $\mathtt{S}$.

2.2.3.1 Mesure aura locale des niveaux de gris

La mesure aura proposée par Elfadel et Picard [7] est définie comme le nombre de fois qu'un pixel $\mathbf{r}$ appartenant à $S_{g'}$ se trouve dans le voisinage d'un pixel $\mathbf{s}$ appartenant à S_g. Pour calculer cette mesure aura pour deux ensembles de niveaux de gris, S_g et $S_{g'}$, il est nécessaire d'examiner le voisinage $\mathtt{N}_\mathbf{s}$ de chaque pixel $\mathbf{s}$ appartenant à S_g, puis de compter les pixels $\mathbf{r}$ de $S_{g'}$ dans ce voisinage. La mesure aura proposée par Elfadel et Picard est ainsi qualifiée par Hammouche et al. [8] de mesure aura locale et notée $m_l\,(g,g')$. La mesure aura locale d'un ensemble S_g par rapport à un autre ensemble $S_{g'}$ pour une fonction de voisinage $\mathtt{N}_\mathbf{s}$ est définie de la manière suivante:

$$m_l\,(S_g,S_{g'}) = \sum_{\mathbf{s}\in S_g} |\mathtt{N}_\mathbf{s} \cap S_{g'}|\,. \tag{2.8}$$

$|.|$ représente le cardinal d'un ensemble.

2.2.3.2 Matrice aura locale des niveaux de gris

La matrice aura locale des niveaux de gris (GLAM_l) regroupe les mesures aura locales entre tous les couples des ensembles de niveau de gris. La GLAM_l notée M_l est de dimension $(q \times q)$, telle que $M_l = \Big[m_l\,(S_g,S_{g'})\Big]$, $0 \leq g,g' \leq q-1$. Afin de simplifier les notations, les

éléments de cette matrice seront dorénavant notés $m_l\,(g,g')$:

$$M_l = \left[m_l\,(g,g')\right]. \tag{2.9}$$

Il en sera de même pour toutes les autres matrices aura que nous définirons par la suite.

Les deux figures 2.1(c) et 2.1(f) montrent deux exemples de calcul de la mesure aura locale $m_l\,(2,3)$ et $m_l\,(3,2)$, respectivement. Les 9 flèches représentent la mesure aura locale. On peut facilement vérifier que la matrice aura locale de l'image présentée dans la figure 2.1(a), construite avec le voisinage asymétrique de la figure 2.1(d), est :

$$M_l = \begin{bmatrix} 0 & 3 & 3 & 12 \\ 3 & 0 & 0 & 0 \\ 2 & 0 & 4 & 9 \\ 13 & 0 & 9 & 2 \end{bmatrix}.$$

Les matrices aura locales dépendent du choix du voisinage utilisé. Lorsque $\mathtt{N_s}$ est symétrique, la GLAM locale sera également symétrique. Elle est désignée dans la littérature sous le nom de SGLAM (Symmetric Gray Level Aura Matrix) [122]. En revanche, si $\mathtt{N_s}$ est asymétrique, comme dans notre exemple (Fig. 2.1(d)), la GLAM est également asymétrique et appelée AGLAM (Asymmetric Gray Level Aura Matrix) [122].

2.2.3.3 Relation entre la matrice aura locale et la matrice de co-occurrence

Elfadel et Picard [7] ont montré que les GLAM$_l$ pouvaient être définies de la même manière que les GLCMs, d'où l'intérêt d'établir une relation entre la matrice de co-occurrence et la matrice aura locale. Rappelons qu'un élément d'une matrice de co-occurrence est défini par [7]:

$$c^{\mathbf{d}}(g,g') = \left|(\mathbf{s},\mathbf{r}) \in \mathtt{S}^2 \text{ tel que } \mathbf{s}+\mathbf{d}=\mathbf{r}, (I(\mathbf{s})=g) \wedge (I(\mathbf{r})=g')\right|. \tag{2.10}$$

$c^{\mathbf{d}}(g,g')$ représente le nombre de couples des pixels séparés par un vecteur de distance $\mathbf{d}$ dont l'un a un niveau de gris g et l'autre un niveau de gris g'.
La matrice de co-occurrence notée $C\,(g,g')$ de dimension $(q \times q)$ est définie par $C\,(g,g') = \left[c^{\mathbf{d}}(g,g')\right]$, où le vecteur de distance $\mathbf{d}$ représente la distance spatiale d séparant deux pixels

selon une orientation θ. Or, un élément de la matrice aura locale est défini par le nombre de pixels de niveau de gris g ayant dans leur voisinage défini par $\mathtt{N}_{\mathbf{s}}$ des pixels de niveaux de gris g'. Compte tenu de ces deux définitions, si $\mathtt{N}_{\mathbf{s}}$ contient un seul pixel tel que $\mathtt{N}_{\mathbf{s}} = \{\mathbf{s}+\mathbf{d}\}$, alors on a :

$$M_l\,(g,g') = C\,(g,g')\,, \tag{2.11}$$

En effet, lorsque le voisinage $\mathtt{N}_{\mathbf{s}} \equiv \mathtt{N}_{\mathbf{s}}^{\mathbf{d}} = \mathbf{s}+\mathbf{d}$ contient un seul voisin, une GLAM$_l$ est identique à une GLCM car $m_l(g,g')$ peut être défini comme le nombre de co-occurrences des pixels $\mathbf{s}$ ayant le niveau de gris g et des pixels voisins $\mathbf{r} = \mathbf{s}+\mathbf{d}$ ayant le niveau de gris g' [7]. Lorsque $\mathtt{N}_{\mathbf{s}}$ est composé de n voisins, il peut être décomposé en n voisinages de base disjoints $\mathtt{N}_{\mathbf{s}}^{\mathbf{d}_i}$ tels que $\mathtt{N}_{\mathbf{s}} = \bigcup_{i=1}^{n} \mathtt{N}_{\mathbf{s}}^{\mathbf{d}_i}$, où chaque $\mathtt{N}_{\mathbf{s}}^{\mathbf{d}_i}$ contient seulement un voisin donné par sa distance $\mathbf{d}_i$ à partir de $\mathbf{s}$. Ainsi,

$$m_l(g,g') = \sum_{i=1}^{n} c^{\mathbf{d}_i}(g,g')\ , \tag{2.12}$$

Une GLAM$_l$ peut donc être considérée comme une généralisation des matrices de co-occurrences des niveaux de gris, puisqu'elle est équivalente à une somme de GLCMs calculées sur ces n voisins de base. Ces GLCMs ont été dénommées par Qin et Yang comme des matrices aura des niveaux de gris de base, ou Basic Gray Level Aura Matrice (BGLAM), et utilisées dans le cadre de la synthèse des textures [123] et la classification des textures [124]. Nous avons également exploité ces BGLAMs pour segmenter des images texturées en niveaux de gris [125].

2.2.3.4 Relation entre la matrice aura locale et la matrice de covariance

La fonction de covariance ou d'auto-corrélation est utilisée pour évaluer la similarité entre une image I et sa version translatée $I^{\mathbf{d}}$ [126]. Cette fonction est fréquemment utilisée dans l'analyse de texture, où elle joue un rôle essentiel dans la caractérisation des propriétés spatiales des motifs présents dans une image. Son lien avec les matrices aura est explicité par Elfadel et Picard [7]. L'auto-corrélation est définie par la relation suivante [126] :

$$A = \sum_{\mathbf{s}\in\mathbf{S}} I(\mathbf{s}) \cdot I(\mathbf{s}+\mathbf{d}) \tag{2.13}$$

$\mathbf{d}$ étant le vecteur de décalage.

Si $\mathbf{s}+\mathbf{d}$ représente les pixels $\mathbf{r}$ voisins de $\mathbf{s}$, à une distance $\mathbf{d}$, l'auto-corrélation peut alors être réécrite sous la forme suivante :

$$A = \sum_{\mathbf{s}\in\mathtt{S}} \sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}} I(\mathbf{s}) \cdot I(\mathbf{r}) \tag{2.14}$$

Cette relation peut être reformulée comme suit :

$$A = \sum_{g\in\Lambda} \sum_{\mathbf{s}\in S_g} \sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}} g I(\mathbf{r}) \tag{2.15}$$

ou encore par :

$$A = \sum_{g\in\Lambda} \sum_{\mathbf{s}\in S_g} \sum_{g'\in\Lambda} \sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}\cap S_{g'}} gg' \tag{2.16}$$

ou simplement :

$$A = \sum_{g\in\Lambda} \sum_{g'\in\Lambda} gg' \sum_{s\in S_g} \sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}\cap S_{g'}} 1 \tag{2.17}$$

Or, $\sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}\cap S_{g'}} 1 = |\mathtt{N}_\mathbf{s} \cap S_{g'}|$. Il s'ensuit alors :

$$A = \sum_{g,g'\in\Lambda} gg' \sum_{\mathbf{s}\in S_g} |\mathtt{N}_\mathbf{s} \cap S_{g'}| \tag{2.18}$$

L'auto-corrélation peut être donc définie en fonction de la mesure aura locale comme suit :

$$A = \sum_{g,g'\in\Lambda} gg'\, m_l(g,g') \tag{2.19}$$

Cette équation révèle une relation entre les mesures aura locales et les matrices d'auto-corrélation, bien qu'elles soient distinctes. Cependant, la dimension $(q \times q)$ des matrices aura est différente de celle des matrices d'auto-corrélation qui dépend de la taille de l'image.

2.2.3.5 Relation entre la matrice aura locale et les champs aléatoires de Markov

Les champs de Markov sont largement utilisés dans le domaine de l'analyse des textures. Ils permettent de modéliser des caractéristiques globales en se basant sur des informations locales [127, 128]. Un champ de Markov est équivalent à un champ de Gibbs défini par une

distribution de probabilité définie de la manière suivante [129]:

$$P(x) = \frac{1}{Z} \exp\left(-\frac{1}{T} E(x)\right) \tag{2.20}$$

où T est « la température » du champ, Z est la constante de normalisation de la distribution de probabilité $P(x)$, et $E(x)$ est la fonction d'énergie associée à l'état x qui peut être formulée comme suit [130] :

$$E(x) = \sum_{\mathbf{s}\in\mathbf{S}} V_{\mathbf{s}}(I(\mathbf{s})) + \sum_{\mathbf{s}\in\mathbf{S}} \sum_{\mathbf{r}\in\mathbf{N_s}} V_{\mathbf{sr}}\left(I(\mathbf{s}), I(\mathbf{r})\right) \tag{2.21}$$

$V_{\mathbf{s}}$ représente le potentiel d'un pixel individuel $\mathbf{s}$ et $V_{\mathbf{sr}}$ désigne le potentiel d'interaction entre deux pixels $\mathbf{s}$ et $\mathbf{r}$. $V_{\mathbf{s}}$ et $V_{\mathbf{sr}}$ sont également connus respectivement comme le champ externe et le champ interne.

Plusieurs modèles de potentiel sont proposés. Les plus populaires sont le modèle auto-binomial et le modèle de Potts [131].

L'énergie de Gibbs d'un modèle auto-binomial est donnée par:

$$E_A(x) = -\sum_{\mathbf{s}\in\mathbf{S}} \left((\alpha_{\mathbf{s}} I(\mathbf{s}) + T \ln \binom{q-1}{I(\mathbf{s})} + \sum_{\mathbf{r}\in N_{\mathbf{s}}} \beta_{\mathbf{sr}} I(\mathbf{s}) I(\mathbf{r}) \right) \tag{2.22}$$

$\alpha_{\mathbf{s}}$ et $\beta_{\mathbf{sr}}$ sont les paramètres du modèle et $\binom{q-1}{I(\mathbf{s})} = \frac{(q-1)!}{I(\mathbf{s})!(q-1-I(\mathbf{s}))!}$ le coefficient binomial.

Dans le cas où le champ de l'image est homogène $\alpha_{\mathbf{s}}$ est indépendant de la position du pixel $\mathbf{s}$ donc $\alpha_{\mathbf{s}} = \alpha$ et $\beta_{\mathbf{sr}} = \beta_{\mathbf{r}}$. De plus, si le champ est isotrope, alors, $\beta_{\mathbf{r}} = \beta$ pour tout $\mathbf{r} \in N_{\mathbf{s}}$, et l'énergie de Gibbs d'un modèle auto binomial homogène devient :

$$E_A(x) = -\sum_{\mathbf{s}\in\mathbf{S}} \left(\alpha I(\mathbf{s}) + \sum_{\mathbf{r}\in N_{\mathbf{s}}} \beta_{\mathbf{r}} I(\mathbf{s}) I(\mathbf{r}) \right) \tag{2.23}$$

En négligeant le champ externe ($\alpha = 0$), cette énergie prend la forme suivante :

$$E_A(x) = \sum_{\mathbf{s}\in\mathbf{S}} \sum_{\mathbf{r}\in\mathbf{N_s}} \beta_{\mathbf{r}} I(\mathbf{s}) I(\mathbf{r}) \tag{2.24}$$

Dans le cas de modèle de Potts, le champ externe est nul ($V_{\mathbf{s}} = 0$) et son énergie est donnée par la formule suivante [130] :

$$E_P(x) = \sum_{\mathbf{s}\in\mathbb{S}} \sum_{\mathbf{r}\in N_{\mathbf{s}}} \beta_{\mathbf{r}} \left(2\delta_{I(\mathbf{s})I(\mathbf{r})} - 1\right) \tag{2.25}$$

avec

$$\delta_{I(\mathbf{s})I(\mathbf{r})} = \begin{cases} 1 & \text{si } I(\mathbf{s}) = I(\mathbf{r}) \\ 0 & \text{sinon} \end{cases} \tag{2.26}$$

En décomposant le système de voisinage en n sous-voisinages tels que $\mathtt{N}_{\mathbf{s}} = \bigcup_{i=1}^{n} \mathtt{N}_{\mathbf{s}}^{i}$, et en effectuant les mêmes transformations que dans le cas de l'auto-corrélation (Eq. 2.13 à 2.19), l'énergie de Gibbs devient :

Pour le modèle auto-binomial,

$$E_A(x) = \sum_{i=1}^{n} \beta_i \sum_{g,g'\in\Lambda} gg' m_l^i(g,g') \tag{2.27}$$

et pour le modèle de Potts,

$$E_P(x) = \sum_{i=1}^{n} \beta_i \sum_{g,g'\in\Lambda} (2\delta_{gg'} - 1) m_l^i(g,g') \ . \tag{2.28}$$

$m_l^i(g,g')$ étant la mesure aura locale de S_g par rapport S'_g selon le voisinage $N_{\mathbf{s}}^i$.

2.2.4 Mesure et matrice aura cardinale des niveaux de gris

2.2.4.1 Mesure aura cardinale des niveaux de gris

La mesure aura locale a été principalement adoptée par la plupart des auteurs car elle généralise les GLCMs et à cause de son lien avec d'autres outils de traitement d'images comme l'auto-correlation et les champs de Markov [123, 132, 133]. Cependant, cette mesure n'évalue pas réellement le nombre de pixels appartenant à l'ensemble aura $\mathtt{A}_{S_{g'}}(S_g)$. Hammouche et al [8] considèrent que la façon la plus simple de mesurer la taille d'un ensemble aura est son cardinal. Ils ont alors proposé une autre mesure et ont montré qu'elle caractérise mieux les

ensembles aura. Cette mesure, appelée mesure aura cardinale et notée $m_c(g,g')$, est définie comme suit :

$$m_c(g,g') = \left|\mathtt{A}_{S_{g'}}(S_g)\right| = \left|\bigcup_{\mathtt{s}\in S_g}(\mathtt{N_s}\cap S_{g'})\right|. \tag{2.29}$$

La mesure aura cardinale évalue le nombre de pixels de $S_{g'}$ qui sont voisins d'un pixel de S_g. Une valeur élevée de cette mesure indique que les deux ensembles sont mélangés selon la structure de voisinage $\mathscr{N}$. Une faible valeur indique que S_g et $S_{g'}$ sont plutôt séparés l'un de l'autre.

2.2.4.2 Matrice aura cardinale des niveaux de gris

La matrice aura cardinale des niveaux de gris ($GLAM_c$) regroupe les mesures aura cardinales entre tous les couples des ensembles des niveaux de gris. Nous noterons alors M_c la matrice aura cardinale de dimension $(q \times q)$, telle que $M_c = \left[m_c\,(g,g')\right]$, $0 \leq g,g' \leq q-1$.

Les figures 2.1(b) et 2.1(e) nous permettent de lire directement les mesures aura cardinales $m_c\,(2,3) = 8$ et $m_c\,(3,2) = 5$. La matrice aura cardinale calculée par le voisinage défini dans la figure 2.1 (d) est:

$$M_c = \begin{bmatrix} 0 & 2 & 2 & 8 \\ 3 & 0 & 0 & 0 \\ 2 & 0 & 3 & 7 \\ 8 & 0 & 5 & 2 \end{bmatrix}.$$

Notons que $m_l(g,g') \geq m_c(g,g')$ pour n'importe quelle paire de niveaux de gris (g,g') [7], mais toutes les cellules qui sont nulles dans M_c le sont également dans M_l. Il est important de noter que contrairement à une matrice aura locale, une matrice aura cardinale peut être asymétrique même si le voisinage $\mathtt{N_s}$ est symétrique.

2.3 Matrices aura des images couleur

Nous proposons dans cette section d'étendre la notion aura aux images couleur. Les images couleur possèdent des informations supplémentaires par rapport aux images en niveaux de

gris. L'utilisation du concept aura pour le traitement d'images couleur pourrait permettre de capturer les relations entre les différentes composantes de couleur et d'exploiter ces informations supplémentaires, ouvrant ainsi de nouvelles perspectives dans ce domaine.

La notion aura peut être étendue aux images couleur à travers trois stratégies distinctes, telles que détaillées dans le chapitre 1 : la stratégie marginale, la stratégie opposée, et la stratégie compacte. Cette dernière, qui constitue une contribution majeure de notre travail, sera exposée dans le prochain chapitre. La stratégie marginale suppose que la texture peut être décrite de manière indépendante dans chaque composante couleur, où les pixels sont caractérisés par un seul niveau relatif à la composante couleur. Dans ce cas, une GLAM comme celle définie sur une image en niveaux de gris, peut être définie sur chaque image composante couleur. Ces matrices aura intra-composantes ne tiennent pas compte de l'aspect vectoriel de l'image couleur, car elles ignorent les corrélations entre les différentes composantes. Pour prendre en compte les interactions spatiales entre les niveaux de composantes couleur, nous proposons, dans ce chapitre, la stratégie opposée, qui s'applique à des canaux multiples. Les matrices aura inter-composantes intègrent la corrélation entre les différentes composantes couleur, de sorte que trois matrices aura inter-composantes sont calculées pour prendre en compte les différentes paires de composantes couleur. Les matrices aura intra-composantes sont appelées matrices aura des niveaux des composantes couleur marginale (Marginal Color Level Aura Matrices), notées MCLAMs. Tandis que, les matrices aura inter-composantes sont appelées matrices aura des niveaux des composantes couleur opposée (Opponent Color Level Aura Matrices), notées OCLAMs. Ces matrices sont basées sur la notion d'ensemble de niveau de composante couleur et de mesure aura des niveaux des composantes couleurs.

2.3.1 Ensemble de niveau de composante couleur

Considérons une image couleur $\mathbf{I}$ définie sur une grille finie $\mathtt{S}$, où chaque pixel $\mathbf{s} \in \mathtt{S}$ est caractérisé par sa couleur $\mathbf{I}(\mathbf{s}) = (I^{C_1}(\mathbf{s}), I^{C_2}(\mathbf{s}), I^{C_3}(\mathbf{s}))^T$. I^k, $k \in \{C_1, C_2, C_3\}$ correspond à une image composante de l'espace de représentation de la couleur (rouge (R), verte (G) et bleue (B) dans le cas d'une image couleur RGB) .

Soit q le nombre de niveaux utilisés pour quantifier chaque composante couleur k. Nous définissons l'ensemble de niveau de composante couleur $S_g^k \subseteq \mathtt{S}$ comme l'ensemble des pixels ayant le niveau donné $g \in \{0,1,...,q-1\}$ dans la composante couleur k, c'est-à-dire $S_g^k = \{\mathbf{s} \in \mathtt{S}, I^k(\mathbf{s}) = g\}$. S_g^k étant une partition de $\mathtt{S}$ telle que : $\bigcup_{g=0}^{q-1} S_g^k = \mathtt{S}$ et $S_g^k \cap S_{g'}^k = \emptyset$ pour $g \neq g'$, $\forall k$ et $S_g^k \cap S_{g'}^{k'} = \emptyset$ pour $k \neq k'$, $\forall g,g'$.

2.3.2 Ensemble aura des niveaux des composantes couleur

Considérons S_g^k et $S_{g'}^{k'}$ deux ensembles de niveau de composante couleur k et k'. Soit $\mathtt{N}_{\mathbf{s}}^{k,k'}$ un voisinage centré sur un pixel $\mathbf{s} \in \mathtt{S}_g^k$. Ce voisinage $\mathtt{N}_{\mathbf{s}}^{k,k'}$ est constitué de pixels $\mathbf{r}$ de la composante couleur k' qui sont voisins du pixel $\mathbf{s}$ dans la composante couleur k. L'ensemble aura de niveau de composante couleur S_g^k par rapport à un autre ensemble de niveau de composante couleur $S_{g'}^{k'}$ est un ensemble composé des pixels de $S_{g'}^{k'}$ présents au voisinage d'un ensemble S_g^k. Cet ensemble aura des niveaux des composantes couleur est noté $\mathtt{A}_{S_{g'}^{k'}}(S_g^k)$ et est défini par :

$$\mathtt{A}_{S_{g'}^{k'}}(S_g^k) = \bigcup_{\mathbf{s} \in S_g^k} (\mathtt{N}_{\mathbf{s}}^{k,k'} \cap S_{g'}^{k'}) \ , \tag{2.30}$$

avec :

$$\mathtt{N}_{\mathbf{s}}^{k,k'} = \left\{ \mathbf{r} \in S_{g'}^{k'}, \left(\mathbf{s} \in S_g^k\right) \wedge \left(\mathbf{r} \in \mathtt{N}_{\mathbf{s}}\right) \right\}. \tag{2.31}$$

2.3.3 Mesures et matrices aura des niveaux des composantes couleur

La mesure aura des niveaux des composantes couleur est définie comme une fonction sur la grille $\mathtt{S}$ qui caractérise un ensemble aura des niveaux intra et inter-composantes couleur. La matrice aura des niveaux des composantes est une représentation des mesures aura pour toutes les paires d'ensembles des niveaux intra et inter-composantes couleur dans la grille $\mathtt{S}$. Comme dans le cas des images en niveaux de gris, deux mesures aura caractérisant la taille d'un ensemble aura des niveaux des composantes couleur et les matrices aura associées peuvent être définies.

2.3.3.1 Mesures et matrices aura locales des niveaux des composantes couleur

La mesure aura locale permet de définir si deux ensembles sont entrelacés, ou plus ou moins séparés. Dans le cas des niveaux de composantes couleur, nous noterons $m_l^{k,k'}(g,g')$ la mesure aura locale d'un ensemble S_g^k par rapport à $\mathtt{S}_{g'}^{k'}$. L'expression de cette mesure suivant le voisinage $\mathtt{N}_{\mathbf{s}}^{k,k'}$ est donnée par la formule suivante :

$$m_l^{k,k'}(g,g') = \sum_{\mathbf{s} \in S_g^k} \left| \mathtt{N}_{\mathbf{s}}^{k,k'} \cap \mathtt{S}_{g'}^{k'} \right|, \tag{2.32}$$

La mesure aura locale des niveaux des composantes couleur estime le nombre de fois qu'un pixel $\mathbf{s}$ de composante couleur k égale à g comporte dans son voisinage un pixel $\mathbf{r}$ dont la composante couleur k' est égale à g'.

La matrice aura locale des niveaux des composantes couleur (Color Level aura matrice Local ou $CLAM_l$) est une représentation des mesures aura locales des niveaux des composantes entre tous les couples (g,g') des niveaux des composantes couleur k et k'. La $CLAM_l$ de dimension $(q \times q)$ est notée $M_l^{k,k'}$ telle que :

$$M_l^{k,k'} = \left[m_l^{k,k'}(g,g') \right], \quad 0 \leq g,g' \leq q-1, \tag{2.33}$$

Pour une image couleur définie par les composants R, G et B, six CLAM$_l$ peuvent être calculées, trois matrices intra-composantes (MCLAMs) $M_l^{R,R}$, $M_l^{G,G}$ et $M_l^{B,B}$, et trois ou six autres inter-composantes (OCLAMs) $M_l^{R,G}$ ($M_l^{G,R}$), $M_l^{R,B}$ ($M_l^{B,R}$) et $M_l^{B,G}$ ($M_l^{G,B}$) (six dans le cas ou le voisinage $\mathtt{N}_{\mathbf{s}}$ n'est pas symétrique).

2.3.3.2 Mesures et matrices aura cardinales des niveaux des composantes couleur

La mesure aura cardinale peut être également exploitée comme mesure de l'ensemble aura des niveaux de composantes couleur. Nous l'appellerons dans ce cas mesure aura cardinale

des niveaux des composantes couleur et la désignerons par $m_c^{k,k'}(g,g')$ telle que :

$$m_c^{k,k'}(g,g') = \left|\mathtt{A}_{\mathtt{S}_{g'}^{k'}}(S_g^k)\right| = \left|\bigcup_{\mathtt{s}\in S_g^k}(\mathtt{N}_{\mathtt{s}}^{k,k'} \cap \mathtt{S}_{g'}^{k'})\right|. \tag{2.34}$$

La matrice aura cardinale des niveaux des composantes couleur (Color Level aura matrix Cardinal ou $CLAM_c$) peut être décrite comme la représentation des mesures aura cardinales des niveaux des composantes couleur entre tous les couples (g,g') des niveaux des composantes couleur k et k'. La $CLAM_c$ de dimension $(q \times q)$ peut donc s'écrire comme suit :

$$M_c^{k,k'} = \left[m_c^{k,k'}(g,g')\right]. \tag{2.35}$$

Comme pour les $CLAM_l$, on peut définir trois $CLAM_c$ intra-composantes (MCLAMs) $M_c^{R,R}$, $M_c^{G,G}$ et $M_c^{B,B}$, et six $CLAM_c$ inter-composantes (OCLAMs) $M_c^{R,G}$, $M_c^{G,R}$, $M_c^{R,B}$, $M_c^{B,R}$, $M_c^{B,G}$ et $M_c^{G,B}$.

2.3.4 Limitations des matrices aura nettes

Toutes les méthodes citées précédement supposent que les images sont nettes. Cependant, les images numériques présentent du flou dû à l'imprécision de la discrétisation spatiale et des niveaux de composantes couleur. Par conséquent, les frontières entre les différentes régions de l'image ne sont pas précisément définies et les niveaux de pixels sont des mesures imprécises. Par conséquent, les GLAMs ou les CLAMs ne peuvent pas décrire correctement des images floues. Pour étayer cette assertion, nous considérons une image en niveaux de gris I_o composée de quatre bandes verticales binaires (Fig. 2.2(a)) et sa version dégradée I_d (Fig. 2.2(b)). Toutes deux sont définies sur une grille $\mathtt{S}$ de taille 8×8 et quantifiées avec $q = 4$ niveaux.

Avec le voisinage de la figure 2.1(d), nous obtenons pour chaque image et chaque mesure aura les GLAMs suivantes :

0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3
0	0	3	3	0	0	3	3

(a)

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

(b)

FIG. 2.2: Une image de texture en niveaux de gris synthétique et sa version dégradée.

$$M_l[I_o] = \begin{bmatrix} 23 & 0 & 0 & 24 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 24 & 0 & 0 & 60 \end{bmatrix} \; M_l[I_d] = \begin{bmatrix} 37 & 8 & 16 & 0 \\ 8 & 7 & 0 & 8 \\ 16 & 0 & 14 & 16 \\ 0 & 8 & 16 & 14 \end{bmatrix}$$

et:

$$M_c[I_o] = \begin{bmatrix} 32 & 0 & 0 & 24 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 24 & 0 & 0 & 32 \end{bmatrix} \; M_c[I_d] = \begin{bmatrix} 23 & 8 & 16 & 0 \\ 8 & 7 & 0 & 8 \\ 16 & 0 & 14 & 16 \\ 0 & 8 & 16 & 14 \end{bmatrix}$$

Pour mesurer la similarité entre les deux images (I_o et I_d), nous avons employé l'intersection entre leurs GLAMs normalisées [134], définie comme suit :

$$\begin{aligned} &\text{Sim}\,(M[I_o],M[I_d]) \\ &= \sum_{g=0}^{q-1}\sum_{g'=0}^{q-1} \min\left(\frac{M[I_o](g,g')}{\sum_{a=0}^{q-1}\sum_{a'=0}^{q-1} M[I_o](a,a')}, \frac{M[I_d](g,g')}{\sum_{a=0}^{q-1}\sum_{a'=0}^{q-1} M[I_d](a,a')}\right) \end{aligned} \tag{2.36}$$

Notons qu'une GLAM est dite normalisée si $\sum_{g,g'} m(g,g') = 1$. La normalisation assure que deux GLAMs sont comparables même si la taille de leurs images est différente et/ou si la

somme de toutes les cellules dans chaque GLAM est différente, comme c'est le cas pour $M_c[I_o]$ et $M_c[I_d]$ dans cet exemple.

La similarité $\mathrm{Sim}\,(M[I_o],M[I_d])$ varie de 0 pour des images très différentes à 1 pour des images très similaires. Ainsi la similarité entre l'image originale et sa version dégradée est égale à $\mathrm{Sim}\,(M_l[I_o],M_l[I_d]) = 0.304$ ou à $\mathrm{Sim}\,(M_c[I_o],M_c[I_d]) = 0.240$, selon la mesure aura utilisée. Ces faibles valeurs montrent que les GLAMs peuvent être fortement affectées par de petits changements de certains niveaux de gris. Pour faire face à ce problème, la notion d'ensembles flous a été étendue au concept aura.

2.4 Matrice aura des images en niveaux de gris flous

La généralisation des matrices aura des niveaux de gris (GLAM) aux matrices aura des niveaux de gris flous a été proposée par Hammouche et al. [8]. Celle-ci considère les niveaux de gris comme des valeurs floues. Dans ce cas, un niveau de gris flou $\tilde{g}$ est caractérisé par sa fonction d'appartenance $\mu_{\tilde{g}}$ et l'ensemble flou S_g par sa fonction d'appartenance $\mu_{S_g}(\mathbf{s}) = \mu_{\tilde{g}}\,(I(\mathbf{s}))$, comme nous l'avons décrit à la section 1.3.3 du chapitre 1.

2.4.1 Voisinage flou

L'ensemble aura tel que proposé par Elfadel et Picard [7] dépend de la structure de voisinage choisie. Ses auteurs considèrent des voisinages spatialement invariants, définis de manière identique pour chaque pixel $\mathbf{s}$, et ces voisinages sont considérés comme des versions translatés d'un élément structurant. Dans le cadre flou, le voisinage $\mathtt{N_s}$ d'un pixel $\mathbf{s}$ est aussi considéré comme un sous-ensemble flou de $\mathtt{S}$. Le voisinage flou d'un pixel $\mathbf{s} \in \mathtt{S}$ est donc défini par une fonction d'appartenance $n_{\mathbf{s}}(\mathbf{r})$, appelée fonction de voisinage. Cette fonction prend des valeurs comprises entre 0 et 1. Elle désigne le degré d'appartenance d'un pixel $\mathbf{r}$ au voisinage du pixel $\mathbf{s}$. Plusieurs fonctions de voisinage peuvent être définies, comme celles utilisées en morphologie mathématique floue [135, 136].

2.4.2 Ensemble aura des niveaux de gris flous

Soient S_g et $S_{g'}$ deux sous-ensembles de niveau de gris flou définis par leurs fonctions d'appartenance μ_{S_g} et $\mu_{S_{g'}}$. Nous définissons l'ensemble aura flou $\mathtt{A}_{S_{g'}}(S_g)$ de S_g par rapport à $S_{g'}$ comme le sous-ensemble flou de $\mathtt{S}$ avec le degré d'appartenance à chaque pixel $\mathbf{r} \in \mathtt{S}$ suivant [8]:

$$\mu_{\mathtt{A}_{S_{g'}}(S_g)}(\mathbf{r}) = \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right)\right],\mu_{S_{g'}}(\mathbf{r})\right\} \tag{2.37}$$

Justification

L'ensemble aura net de l'équation (2.1) peut être réécrit comme suit:

$$\mathtt{A}_{S_{g'}}(S_g) = \bigcup_{\mathbf{s}\in\mathtt{S}}(\mathtt{N}_{\mathbf{s},S_g} \cap S_{g'}) \ , \tag{2.38}$$

où $\mathtt{N}_{\mathbf{s},S_g}$ est l'ensemble net contenant les pixels voisins $\mathbf{r}$ de chaque pixel $\mathbf{s} \in S_g$, qui est donné par la relation suivante :

$$\mathtt{N}_{\mathbf{s},S_g} = \{\mathbf{r} \in \mathtt{S}, (\mathbf{s} \in S_g) \wedge (\mathbf{r} \in \mathtt{N}_{\mathbf{s}})\} \, . \tag{2.39}$$

Son équivalent flou est défini par le degré d'appartenance à chaque pixel $\mathbf{r} \in \mathtt{S}$ suivant:

$$\mu_{\mathtt{N}_{\mathbf{s},S_g}}(\mathbf{r}) = \min\left(\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right) \ . \tag{2.40}$$

Dans cette expression, l'opérateur logique *et* ($\wedge$) est transcrit par l'opérateur flou *min*, et l'appartenance de $\mathbf{r}$ au voisinage de $\mathbf{s}$ est transcrite par $n_{\mathbf{s}}(\mathbf{r})$. Selon la théorie des ensembles flous, l'ensemble flou $\mathtt{N}_{\mathbf{s},S_g} \cap S_{g'}$ de l'equation (2.38) est défini par sa fonction d'appartenance donnée par:

$$\mu_{\mathtt{N}_{\mathbf{s},S_g}\cap S_{g'}}(\mathbf{r}) = \min\left[\mu_{\mathtt{N}_{\mathbf{s},S_g}}(\mathbf{r}),\mu_{S_{g'}}(\mathbf{r})\right] \ . \tag{2.41}$$

L'ensemble aura flou est défini par analogie avec l'ensemble aura net. L'opérateur d'union de l'ensemble net $\bigcup$ de l'équation (2.38) est transcrit par l'opérateur flou *sup* [64] et permet

d'obtenir le degré d'appartenance de l'ensemble aura flou à chaque pixel $\mathbf{r} \in \mathtt{S}$:

$$\begin{aligned}\mu_{\mathtt{A}_{S_{g'}}(S_g)}(\mathbf{r}) &= \sup_{\mathbf{s}\in\mathtt{S}} \left(\mu_{\mathtt{N}_{\mathbf{s},S_g}\cap S_{g'}}(\mathbf{r})\right) \\ &\overset{(2.41)}{=} \sup_{\mathbf{s}\in\mathtt{S}} \left\{\min\left[\mu_{\mathtt{N}_{\mathbf{s},S_g}}(\mathbf{r}),\mu_{S_{g'}}(\mathbf{r})\right]\right\} \\ &= \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\mu_{\mathtt{N}_{\mathbf{s},S_g}}(\mathbf{r})\right],\mu_{S_{g'}}(\mathbf{r})\right\} \end{aligned} \tag{2.42}$$

L'insertion de la définition de $\mu_{\mathtt{N}_{\mathbf{s},S_g}}(\mathbf{r})$ de l'équation (2.40) aboutit l'équation (2.37).

2.4.3 Description morphologique d'un ensemble aura flou

Comme dans le cas net (voir l'équation (2.7)), lorsque le voisinage est défini de manière identique en chaque pixel $\mathbf{s}$, l'ensemble aura flou peut également être défini en utilisant une dilatation morphologique. Dans le cadre de la morphologie mathématique floue, l'élément structurant est un ensemble flou défini par sa fonction d'appartenance n appelée *fonction structurante*. La définition la plus classique de la dilatation d'un ensemble flou S_g par n est à chaque pixel $\mathbf{s} \in \mathtt{S}$ [137]:

$$D2_v(\mu_{S_g})(\mathbf{s}) = \sup_{\mathbf{r}\in\mathtt{S}}\left[\min\left(\mu_{S_g}(\mathbf{r}),n(\mathbf{r}-\mathbf{s})\right)\right] \ . \tag{2.43}$$

La fonction structurante n dépend du vecteur de translation $\mathbf{r}-\mathbf{s}$ et correspond à la fonction de voisinage $n_{\mathbf{s}}(\mathbf{r})$ à condition que cette dernière soit invariante spatialement. L'ensemble aura flou peut alors être défini en transcrivant la formulation morphologique nette (2.7) comme suit :

$$\mu_{\mathtt{A}_{S_{g'}}(S_g)}(\mathbf{s}) = \min\left\{\sup_{\mathbf{r}\in\mathtt{S}}\left[\min\left(\mu_{S_g}(\mathbf{r}),n_{\mathbf{s}}(\mathbf{r})\right)\right],\mu_{S_{g'}}(\mathbf{s})\right\} \ . \tag{2.44}$$

En échangeant les variables $\mathbf{r}$ et $\mathbf{s}$, nous obtenons :

$$\mu_{\mathtt{A}_{S_{g'}}(S_g)}(\mathbf{r}) = \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{S_g}(\mathbf{s}),n_{\mathbf{r}}(\mathbf{s})\right)\right],\mu_{S_{g'}}(\mathbf{r})\right\} \ . \tag{2.45}$$

Cette définition morphologique d'un ensemble aura flou est totalement cohérente avec l'équation (2.37) à condition que $n_{\mathbf{r}}(\mathbf{s})$ soit identique à $n_{\mathbf{s}}(\mathbf{r})$, c'est-à-dire que la fonction de voisinage soit symétrique et spatialement invariante.

2.4.4 Exemple illustratif

Pour une meilleure compréhension du calcul des ensembles aura flous à partir de l'équation (2.37), nous présentons un exemple illustratif sur la figure 2.4. Dans ce contexte, considérons la fonction de voisinage $n_{\mathbf{s}}(\mathbf{r})$ représentée sur la figure 2.3(a) est défini par :

$$n_{\mathbf{s}}(\mathbf{r}) = \begin{cases} 1 & \text{if } (\|\mathbf{r}-\mathbf{s}\|_1 \leq 1) \wedge (\mathbf{r}-\mathbf{s} \neq (0,1)^{\intercal}), \\ 0 & \text{sinon.} \end{cases} \tag{2.46}$$

Notons que $n_{\mathbf{s}}(\mathbf{r})$ peut généralement prendre n'importe quelle valeur réelle comprise entre 0 et 1, mais dans cet exemple, pour simplifier, $n_{\mathbf{s}}(\mathbf{r})$ est pris binaire.

À chaque pixel $\mathbf{r}$, les pixels $\mathbf{s}$ pour lesquels $n_{\mathbf{s}}(\mathbf{r}) = 1$ font partie du voisinage défini par la transposition de la fonction de voisinage $\mathtt{N}_{\mathbf{s}}$, autrement dit, le voisinage (habituellement désigné par $\check{\mathtt{N}}_{\mathbf{r}}$) réfléchi par rapport au pixel d'intérêt $\mathbf{r}$. Cela est illustré dans la figure 2.3(b) pour un pixel $\mathbf{r}_0$ (les pixels $\mathbf{s}$ dont $\mathbf{r}_0$ est voisin étant représentés par des losanges), et généralisé à tout pixels $\mathbf{r}$ dans la figure 2.3(c).

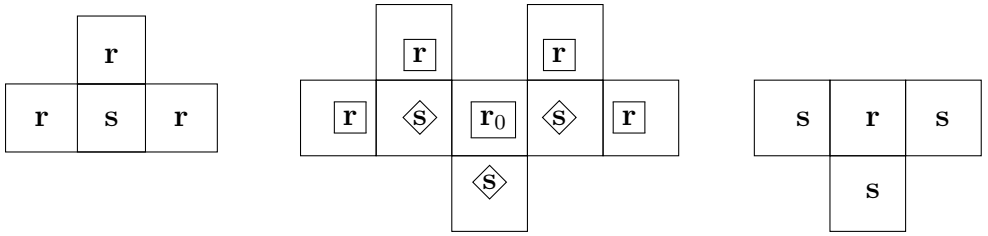


FIG. 2.3: Illustration de la transposition de la fonction de voisinage symétrique : (a) voisinage $\mathtt{N}_{\mathbf{s}}$, (b) $\check{\mathtt{N}}_{\mathbf{r}_0} = \{\mathbf{s} \in \mathtt{S}, \mathbf{r}_0 \in \mathtt{N}_{\mathbf{s}}\}$, (c) $\check{\mathtt{N}}_{\mathbf{r}}$.

La Figure 2.4 montre deux exemples d'ensembles aura flous calculés sur l'image de la figure 2.1(a) avec la fonction de voisinage affichée sur la figure 2.1(d). Les figures 2.4(b) et 2.4(d) montrent les degrés d'appartenance $\mu_{\mathtt{S}_3}$ et $\mu_{\mathtt{S}_2}$ de chaque pixel $\mathbf{s}$ aux ensembles flous S_3 et S_2. Ils sont définis à partir de la fonction d'appartenance triangulaire avec $\alpha = 2$ (équations (1.13) et (1.16). L'équation (2.42) indique que le degré d'appartenance de $\mu_{\mathtt{A}_{S_2}(S_3)}$ en chaque pixel $\mathbf{r}$ est le minimum entre $\mu_{S_2}(\mathbf{r})$ et $\sup_{\mathbf{s}\in\mathtt{S}} \left[\mu_{\mathtt{N}_{\mathbf{s}},S_g}(\mathbf{r})\right]$. La figure 2.4(c) montre $\sup_{\mathbf{s}\in\mathtt{S}} \left[\mu_{\mathtt{N}_{\mathbf{s}},S_3}(\mathbf{r})\right] = \sup_{\mathbf{s}\in\mathtt{S}} \left[\min\left(\mu_{S_3}(\mathbf{s}), n_{\mathbf{s}}(\mathbf{r})\right)\right]$ pour chaque $\mathbf{r} \in \mathtt{S}$. On considère

les deux pixels $\mathbf{r}$ encadrés dans la figure 2.4(b) qui sont voisins des quatre pixels encerclés tels que $\mathbf{s} \in \mathtt{N}_r$. À chaque pixel $\mathbf{r}$, le maximum de ces quatre valeurs est retenu comme $\sup_{\mathbf{s}\in\mathtt{S}} [\min (\mu_{S_3}(\mathbf{s}), n_{\mathbf{s}}(\mathbf{r}))]$, ce qui donne les valeurs de la figure 2.4(c). La figure 2.4(e) présente la fonction d'appartenance de l'ensemble aura flou $\mu_{\mathtt{A}_{\mathtt{S}_2}(\mathtt{S}_3)}$ calculée comme le minimum pour chaque pixel entre les valeurs représentées sur les figures 2.4(c) et 2.4(d) (voir l'équation (2.37)). La figure 2.4(f) affiche les valeurs de $\mu_{\mathtt{A}_{\mathtt{S}_3}(\mathtt{S}_3)}$.

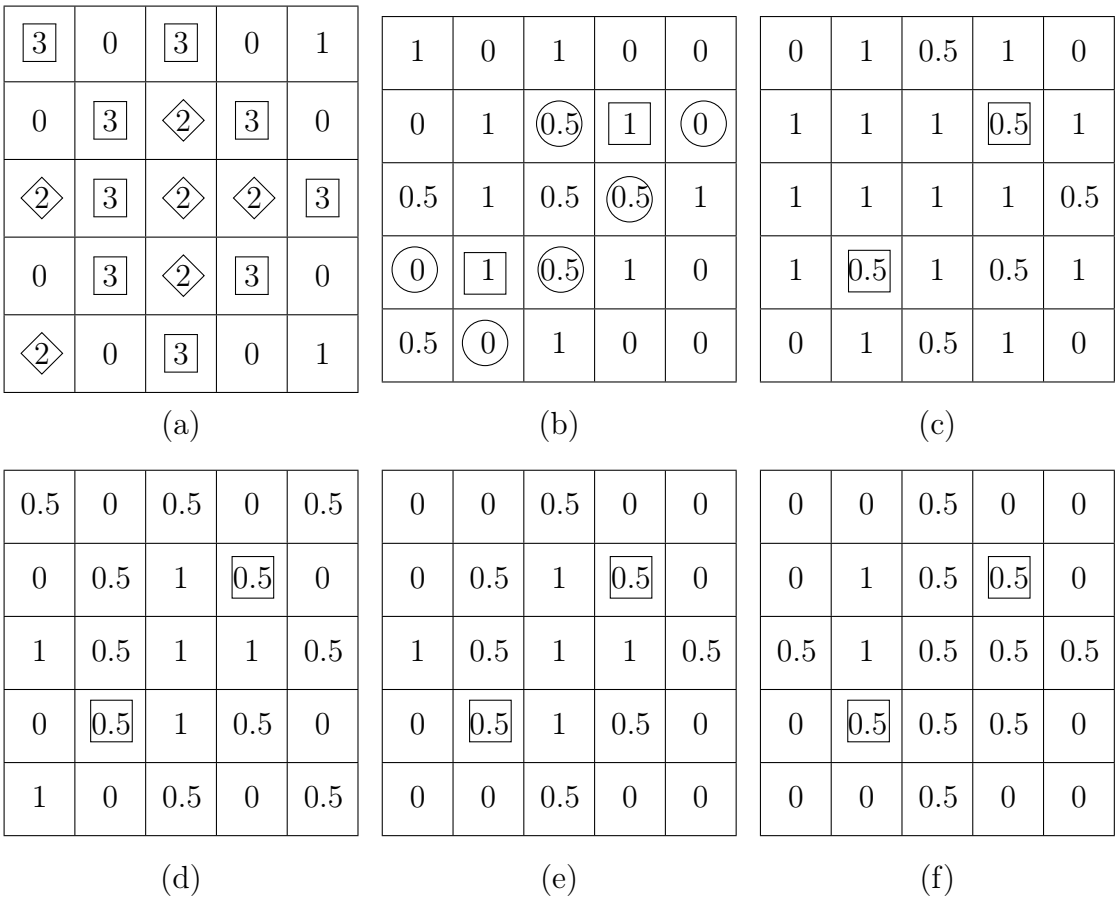


FIG. 2.4: Illustration des différentes étapes de calcul des ensembles aura flous : (a) Ensembles de niveaux de gris S_2 (losanges) et S_3 (carrés), (b) Fonction d'appartenance μ_{S_3}, (c) $\sup_{\mathbf{s}\in\mathtt{S}} [\min (\mu_{S_3}(\mathbf{s}), n_{\mathbf{s}}(\mathbf{r}))]$, (d) Fonction d'appartenance μ_{S_2}, (e) Ensemble aura flou $\mu_{\mathtt{A}_{S_2}(S_3)}$, (f) Ensemble aura flou $\mu_{\mathtt{A}_{S_3}(S_3)}$.

2.4.5 Mesures et matrices aura des niveaux de gris flous

Deux mesures aura floues d'un sous-ensemble flou S_g par rapport à un autre sous-ensemble flou $S_{g'}$ sont déduites de leurs équivalentes nettes [8] : La mesure aura locale floue et la mesure aura cardinale floue.

Deux matrices aura floues découlent de ces deux mesures. Elles seront qualifiées de matrices aura des niveaux de gris flous FGLAM locale (FGLAM$_l$) et FGLAM cardinale (FGLAM$_c$) selon la mesure aura utilisée.

2.4.5.1 Mesure et matrice aura locale floue des niveaux de gris

La mesure aura locale floue $\tilde{m}_l(g,g')$ d'un ensemble flou S_g par rapport à un ensemble flou $S_{g'}$ est définie par [8]:

$$\tilde{m}_l(g,g') = \sum_{\mathbf{r}\in\mathtt{S}} \sum_{\mathbf{s}\in\mathtt{S}} \min\left\{\min\left[\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right],\mu_{S_{g'}}(\mathbf{r})\right\} \ . \tag{2.47}$$

Justification La mesure aura locale nette de l'équation (2.8) peut être réécrite comme suit:

$$m_l\left(S_g,S_{g'}\right) = \sum_{\mathbf{s}\in S_g} \left|\mathtt{N}_{\mathbf{s},S_g} \cap S_{g'}\right|, \tag{2.48}$$

Sachant que le cardinal d'un ensemble flou $\mathtt{S}$ est $|\mathtt{S}| = \sum_{\mathbf{r}\in\mathtt{S}} \mu_{\mathtt{S}}(\mathbf{r})$ [64], la mesure aura locale floue de l'ensemble flou S_g par rapport à un ensemble $S_{g'}$ est définie par son équivalent flou comme suit :

$$\tilde{m}_l(g,g') = \sum_{\mathbf{s}\in\mathtt{S}} \sum_{\mathbf{r}\in\mathtt{S}} \mu_{\mathtt{N}_{\mathbf{s}},S_g\cap S_{g'}}(\mathbf{r}). \tag{2.49}$$

Le degré d'appartenance $\mu_{\mathtt{N}_{\mathbf{s}},S_g\cap S_{g'}}(\mathbf{r})$ est déduit à partir des l'équations (2.40) et (2.41) comme suit :

$$\mu_{\mathtt{N}_{\mathbf{s}},S_g\cap S_{g'}}(\mathbf{r}) = \min\left\{\min\left[\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right],\mu_{S_{g'}}(\mathbf{r})\right\}. \tag{2.50}$$

L'insertion de l'expression définie par l'équation (2.50) dans l'équation (2.49) conduit à l'expression de la mesure aura locale floue présentée dans l'équation (2.47).

Dans le cas simple d'une fonction de voisinage binaire (Eq.2.46), chaque paire de pixels $(\mathbf{r},\mathbf{s})$ pour laquelle $n_{\mathbf{s}}(\mathbf{r}) = 1$ contribue à $\tilde{m}_l(g,g')$ en ajoutant le minimum entre $\mu_{S_g}(\mathbf{s})$ et $\mu_{S_{g'}}(\mathbf{r})$. La mesure aura locale floue peut être alors réécrite comme suit :

$$\tilde{m}_l(g,g') = \sum_{\mathbf{s}\in\mathtt{S}} \sum_{\mathbf{r}\in\mathtt{N}_{\mathbf{s}}} \min\left(\mu_{S_g}(\mathbf{s}),\mu_{S_{g'}}(\mathbf{r})\right) \tag{2.51}$$

La FGLAM locale (FGLAM$_l$), notée $\tilde{M}_l$, est une matrice réelle de dimension $q \times q$ regroupant toutes les mesures aura locale floues entre deux couples de sous-ensembles flous S_g et $S_{g'}$ $\tilde{M}_l = [\tilde{m}_l(g,g')]$, $(g,g') \in \Lambda^2$.

Notons que, d'une manière générale, FGLAM$_l$ est asymétrique. La matrice aura locale floue, calculée sur l'image de la Figure 2.1(a) avec la fonction de voisinage indiquée sur la figure 2.1(d), est donc:

$$\tilde{M}_l = \begin{bmatrix} 3.0 & 5.5 & 10.0 & 14.0 \\ 6.0 & 7.5 & 15.5 & 14.0 \\ 10.5 & 15.5 & 14.0 & 16.5 \\ 13.5 & 14.0 & 16.5 & 13.0 \end{bmatrix} .$$

2.4.5.2 Relation entre la matrice aura locale floue et la matrice de co-occurrence floue

Rappelons qu'une Matrice de Co-occurrence des Niveaux de Gris flous (FGLCM) [72, 138] est une matrice $\tilde{C} = \left[\tilde{c}^{\mathbf{d}}(g,g')\right]$ de taille $q \times q$ où l'élément $c^{\mathbf{d}}(g,g')$ représente le nombre de paires de pixels ayant le niveau de gris g qui sont voisins des pixels ayant le niveau de gris g' séparés par un vecteur de distance $\mathbf{d}$, défini par une distance d et une orientation θ. En d'autres termes, $c^{\mathbf{d}}(g,g')$ exprime le nombre d'occurrences pour une paire de nombres flous $\tilde{g}$ et $\tilde{g}'$ de deux pixels séparés par $\mathbf{d}(d,\theta)$. Il est défini comme suit :

$$\tilde{c}^{\mathbf{d}}(g,g') = \sum_{\mathbf{s} \in \mathbf{S}} \min \left\{ \mu_{\tilde{g}} \left(I(\mathbf{s}) \right), \mu_{\tilde{g}'} \left(I(\mathbf{s}+\mathbf{d}) \right) \right\} . \tag{2.52}$$

Comme dans le cas net, lorsque la fonction de voisinage $n_{\mathbf{s}}(\mathbf{r})$ est binaire et vaut 1 pour un seul voisin seulement, une FGLAM$_l$ est identique à une FGLCM. En effet, dans ce cas, la fonction de voisinage s'écrit :

$$n_{\mathbf{s}}^{\mathbf{d}}(\mathbf{r}) = \begin{cases} 1 & \text{if } \mathbf{r} = \mathbf{s} + \mathbf{d}, \\ 0 & \text{sinon}, \end{cases} \tag{2.53}$$

et la mesure aura locale floue prend la forme suivante :

$$\begin{aligned}\tilde{m}_l(g,g') &\overset{(2.47)}{=} \sum_{\mathbf{s}\in\mathtt{S}}\sum_{\mathbf{r}\in\mathtt{S}} \min\left\{\min\left[\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}^{\mathbf{d}}(\mathbf{r})\right],\mu_{S_{g'}}(\mathbf{r})\right\}\\ &\overset{(2.53)}{=} \sum_{\mathbf{s}\in\mathtt{S}} \min\left\{\mu_{S_g}(\mathbf{s}),\mu_{S_{g'}}(\mathbf{s}+\mathbf{d})\right\}\\ &\overset{(1.16)}{=} \sum_{\mathbf{s}\in\mathtt{S}} \min\left\{\mu_{\tilde{g}}(I(\mathbf{s})),\mu_{\tilde{g}'}(I(\mathbf{s}+\mathbf{d}))\right\}\\ &\overset{(2.52)}{=} \tilde{c}^{\mathbf{d}}(g,g')\ . \end{aligned} \tag{2.54}$$

2.4.5.3 Mesure et matrice aura cardinale floue des niveaux de gris

La mesure aura cardinale floue $\tilde{m}_c(g,g')$ d'un sous-ensemble de niveau de gris flou S_g par rapport à un autre sous-ensemble flou $S_{g'}$ est définie par [8]:

$$\tilde{m}_c(g,g') = \sum_{\mathbf{r}\in\mathtt{S}} \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{S_g}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right)\right],\mu_{S_{g'}}(\mathbf{r})\right\}\ , \tag{2.55}$$

sachant que $\tilde{m}_c(g,g') = \left|\mathtt{A}_{S_{g'}}(\mathtt{S}_g)\right| = \sum_{\mathbf{r}\in\mathtt{S}} \mu_{\mathtt{A}_{S_{g'}}(S_g)}(\mathbf{r})$.

Pour une fonction de voisinage binaire (Eq. 2.46), la mesure aura cardinale floue se réduit à :

$$\tilde{m}_c(g,g') = \sum_{\mathbf{r}\in\mathtt{S}} \min\left(\sup_{\mathbf{s}\in\tilde{\mathtt{N}}_{\mathbf{r}}}\left(\mu_{S_g}(\mathbf{s})\right),\mu_{S_{g'}}(\mathbf{r})\right)\ . \tag{2.56}$$

Cette relation montre que le calcul de $\tilde{m}_c(g,g')$ est basé sur le voisinage transposé $\tilde{\mathtt{N}}_{\mathbf{r}}$ (Fig. 2.3), alors que celui de $\tilde{m}_l(g,g')$ est basé sur le voisinage $\mathtt{N}_{\mathbf{s}}$.

La FGLAM cardinale (FGLAM$_c$), notée $\tilde{M}_c$, de dimension $q \times q$, regroupe toutes les mesures aura cardinales floues entre deux couples de sous-ensembles flous S_g et $S_{g'}$, telle que $\tilde{M}_c = [\tilde{m}_c(g,g')]$, $(g,g') \in \Lambda^2$. La FGLAM$_c$ est asymétrique quelle que soit la fonction de voisinage.

La matrice aura cardinale floue, calculée sur l'image de la figure 2.1(a) avec la fonction de voisinage donnée par la figure 2.1(d), est :

$$\tilde{M}_c = \begin{bmatrix} 2.5 & 4.5 & 7.0 & 9.0 \\ 5.5 & 6.5 & 8.0 & 7.0 \\ 5.5 & 6.5 & 7.5 & 9.5 \\ 8.0 & 6.5 & 8.5 & 7.0 \end{bmatrix} .$$

2.4.6 Avantages des FGLAMs par rapport aux GLAMs

Pour démontrer l'apport du flou et l'intérêt des FGLAMs par rapport aux GLAMs, nous avons calculé les FGLAMs locale et cardinale de l'image de texture I_o et de l'image de texture dégradée I_d, présentées dans la section 2.3.4 avec la fonction de voisinage de la Figure 2.1(d). Les résultats obtenus sont :

$$\tilde{M}_l[I_o] = \begin{bmatrix} 60 & 30 & 12 & 24 \\ 30 & 30 & 12 & 12 \\ 12 & 12 & 30 & 30 \\ 24 & 12 & 30 & 60 \end{bmatrix} \tilde{M}_l[I_d] = \begin{bmatrix} 48.5 & 42 & 27.5 & 12 \\ 42 & 56.5 & 34.5 & 31 \\ 27.5 & 34.5 & 48.5 & 42 \\ 12 & 31 & 42 & 37 \end{bmatrix} ,$$

$$\tilde{M}_c[I_o] = \begin{bmatrix} 32 & 16 & 12 & 24 \\ 16 & 16 & 12 & 12 \\ 12 & 12 & 16 & 16 \\ 24 & 12 & 16 & 32 \end{bmatrix} \tilde{M}_c[I_d] = \begin{bmatrix} 27.5 & 28 & 24 & 12 \\ 20 & 27.5 & 20 & 20 \\ 20 & 20 & 27 & 24 \\ 12 & 24 & 28 & 23 \end{bmatrix} .$$

En mesurant la similarité entre les deux images de texture I_o et I_d par l'intersection de leurs FGLAMs normalisées, nous obtenons: $\text{Sim}\left(\tilde{M}_l[I_o],\tilde{M}_l[I_d]\right) = 0.793$ et $\text{Sim}\left(\tilde{M}_c[I_o],\tilde{M}_c[I_d]\right) = 0.808$. Ces valeurs sont bien plus élevées que celles mesurées entre les GLAMs normalisées (0.304 et 0.240), ce qui démontre que les FGLAMs sont moins affectées par la dégradation de l'image que les GLAMs.

2.5 Matrices aura des images couleurs floues

Les notions de mesure et matrices aura des niveaux de gris flous peuvent être également étendues aux images couleur. Comme dans le cas "net", cette extension peut se faire selon la stratégie marginale, opposée et compacte. La stratégie compacte sera abordée dans le prochain chapitre. Les deux autres stratégies seront décrites conjointement dans cette section pour définir les matrices aura floues intra et inter composantes. Ces matrices seront dénommées matrices aura des niveaux de composantes couleur flous (Fuzzy Color Level Aura Matrices : FCLAMs), comme pour les matrices aura des niveaux des composantes couleur (CLAMs) (voir section 2.3).

2.5.1 Ensemble de niveau de composante couleur flou

Soit une image de composantes couleur $\mathbf{I}$ définie par ses trois composantes couleur $\mathbf{I}^k$ ($k \in \{C_1,C_2,C_3\}$) sur une grille $\mathtt{S}$. L'ensemble de niveau de composante couleur k flou $S_g^k \subseteq \mathtt{S}$ est un sous-ensemble flou défini par le degré d'appartenance $\mu_{S_g^k}(\mathbf{s})$ de chaque pixel $\mathbf{s} \in \mathtt{S}$ à S_g^k. Pour définir la fonction d'appartenance $\mu_{S_g^k}$, nous considérons les valeurs de chaque composante couleur k comme des niveaux flous [72].

Un niveau flou $\tilde{g}$ d'une composante couleur k est caractérisé par sa fonction d'appartenance $\mu_{\tilde{g}}^k(x) : [0,\ldots,q-1] \rightarrow [0,1]$. Les fonctions les plus utilisées sont les fonctions d'appartenance gaussiennes et triangulaires définies comme $\mu_{\tilde{g}}^k(x) = \exp\left(-|x-g|^2/2\alpha^2\right)$ et $\mu_{\tilde{g}}^k(x) = \max\left(1-|x-g|/\beta,\ 0\right)$. Ces fonctions sont identiques à celle utilisée pour les ensembles des niveaux de gris flous (Eq.1.12 et 1.13). Le degré d'appartenance de chaque pixel $\mathbf{s}$ à l'ensemble flou $\mathtt{S}_g^k$ est ainsi défini par le degré d'appartenance $\mu_{\tilde{g}}^k\left(I^k(\mathbf{s})\right)$ de la valeur de la composante couleur $\mathbf{I}^k(\mathbf{s})$ du pixel $\mathbf{s}$ au niveau flou $\tilde{g}$:

$$\mu_{S_g^k}(\mathbf{s}) = \mu_{\tilde{g}}^k\left(I^k(\mathbf{s})\right). \tag{2.57}$$

2.5.2 Ensemble aura des niveaux des composantes couleur flous

Soient S_g^k et $S_{g'}^{k'}$ deux sous-ensembles de niveau de composante couleur flou définis par leurs fonctions d'appartenance $\mu_{S_g^k}$ et $\mu_{S_{g'}^{k'}}$. L'ensemble aura de S_g^k par rapport à $S_{g'}^{k'}$, dénoté par $\mu_{\mathtt{A}_{S_{g'}^{k'}}(S_g^k)}(\mathbf{r})$, est défini comme le sous-ensemble flou de $\mathtt{S}$ et se déduit de son équivalent net de l'équation (2.30) par :

$$\mu_{\mathtt{A}_{S_{g'}^{k'}}(S_g^k)}(\mathbf{r}) = \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{S_g^k}(\mathbf{s}),n_{\mathbf{s}}^{k,k'}(\mathbf{r})\right)\right],\mu_{S_{g'}^{k'}}(\mathbf{r})\right\} \tag{2.58}$$

La justification de cette relation suit les mêmes étapes que celles établies pour les images en niveaux de gris (section 2.4.2).

2.5.3 Mesures aura et matrices aura des niveaux des composantes couleur flous

Deux mesures aura d'un sous-ensemble de niveau de composante couleur k flou S_g^k par rapport à un sous-ensemble de niveau de composante couleur k' flou $S_{g'}^{k'}$ sont déduites de leurs équivalents nets comme dans le cas des images en niveaux de gris (section 2.2.3 et 2.2.4).

La mesure aura locale des niveaux des composantes couleur flous est définie comme suit :

$$\tilde{m}_l^{k,k'}(g,g') = \sum_{\mathbf{r}\in\mathtt{S}}\sum_{\mathbf{s}\in\mathtt{S}}\min\left\{\min\left[\mu_{S_g^k}(\mathbf{s}),n_{\mathbf{s}}^{k,k'}(\mathbf{r})\right],\mu_{S_{g'}^{k'}}(\mathbf{r})\right\} \tag{2.59}$$

et la mesure aura cardinale des niveaux des composantes couleur flous $\tilde{m}_c^{k,k'}(g,g')$ est définie par :

$$\tilde{m}_c^{k,k'}(g,g') = \sum_{\mathbf{r}\in\mathtt{S}}\min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{S_g^k}(\mathbf{s}),n_{\mathbf{s}}^{k,k'}(\mathbf{r})\right)\right],\mu_{S_{g'}^{k'}}(\mathbf{r})\right\}\ , \tag{2.60}$$

sachant que $\tilde{m}_c^{k,k'}(g,g') = \left|\mathtt{A}_{S_{g'}^{k'}}(S_g^k)\right| = \sum_{\mathbf{r}\in\mathtt{S}}\mu_{\mathtt{A}_{S_{g'}^{k'}}(S_g^k)}(\mathbf{r})$.

Dans le cas simple d'une fonction de voisinage binaire, chaque paire de pixels $(\mathbf{r},\mathbf{s})$ pour laquelle $n_{\mathbf{s}}^{k,k'}(\mathbf{r}) = 1$, $\tilde{m}_l^{k,k'}(g,g')$ et $\tilde{m}_c^{k,k'}(g,g')$ se réduisent à ces deux expressions :

$$\tilde{m}_l^{k,k'}(g,g') = \sum_{\mathbf{s}\in\mathtt{S}} \sum_{\mathbf{r}\in\mathtt{N}_\mathbf{s}} \min\left(\mu_{S_g^k}(\mathbf{s}),\mu_{S_{g'}^{k'}}(\mathbf{r})\right) \tag{2.61}$$

$$\tilde{m}_c^{k,k'}(g,g') = \sum_{\mathbf{r}\in\mathtt{S}} \min\left(\sup_{\mathbf{s}\in\mathtt{N}_\mathbf{r}}\left(\mu_{S_g^k}(\mathbf{s})\right),\mu_{S_{g'}^{k'}}(\mathbf{r})\right) \ . \tag{2.62}$$

Une matrice aura des niveaux des composantes couleur flous (FCLAM) est une représentation de toutes les mesures aura des niveaux des composantes couleur flous.

Dans le cas d'une mesure aura locale, la FCLAM est nommée FCLAM locale (FCLAM$_l$) et dénotée par $\tilde{M}_l^{k,k'}$, telle que $\tilde{M}_l^{k,k'} = \left[\tilde{m}_l^{k,k'}(g,g')\right]$, $(g,g') \in \Lambda^2$.

Dans le cas d'une mesure aura cardinale, elle est appelée FCLAM cardinale (FCLAM$_c$), dénotée par $\tilde{M}_c^{k,k'}$, telle que $\tilde{M}_c^{k,k'} = \left[\tilde{m}_c^{k,k'}(g,g')\right]$, $(g,g') \in \Lambda^2$.

À partir de ces deux FCLAMs, on peut définir : trois FCLAMs marginales (MFCLAMs : Marginal Fuzzy Color Level Aura Matrices) $\tilde{M}^{R,R}$, $\tilde{M}^{G,G}$ et $\tilde{M}^{B,B}$, et trois ou six FCLAMs opposées (OFCLAMs : Opponent Fuzzy Color Level Aura Matrices) $\tilde{M}^{R,G}$ ($\tilde{M}^{G,R}$), $\tilde{M}^{R,B}$ ($\tilde{M}^{B,R}$) et $\tilde{M}^{G,B}$ ($\tilde{M}^{B,G}$).

2.6 Conclusion

Dans ce chapitre, nous avons présenté en détail les notions fondamentales des ensembles aura, des mesures aura et des matrices aura (Gray Level Aura Matrices ou GLAMs) dans le contexte des images en niveaux de gris.

Puis, nous avons étendu dans un premier temps ces notions aux images couleur en introduisant particulièrement les matrices aura des niveaux des composantes couleurs (Color Level Aura Matrices ou CLAMs), suivans les deux stratégies : marginale (Marginal Color Level Aura Matrices ou MCLAMs) et opposée (Opponent Color Level Aura Matrices ou OCLAMs).

Dans un deuxième temps, nous avons étendu le concept d'ensemble, de mesure et de matrice aura au cadre flou en définissant les matrices aura des niveaux de gris flous (Fuzzy Gray Level Aura Matrices, FGLAMs). Nous avons montré, à travers un exemple, l'avantage du flou et des FGLAMs sur les GLAMs. Au final, nous avons décrit les matrices aura des niveaux des composantes couleur flous (Fuzzy Color Level aura Matrices ou FCLAMs). L'extension des matrices aura aux images couleur dans le cadre flou a été également réalisée selon deux stratégies, à savoir marginale (MFCLAMs) et opposée (OFCLAMs). Cette dernière constitue l'une des contributions de notre travail. Ces deux stratégies, regroupées en une seule, prennent en compte la nature tridimensionnelle des données couleur (intra et inter composantes), ce qui permet une meilleure représentation de la texture couleur. Néanmoins, elles présentent certaines limites, notamment en termes de temps de calcul et d'espace mémoire. Pour surmonter ces limitations, nous proposerons, dans le prochain chapitre l'emploi de la stratégie compacte.

Chapitre 3

Matrices aura des couleurs floues

3.1 Introduction

Afin d'exploiter pleinement les informations de texture couleur présentes dans une image, nous avons étendu, dans le chapitre précédent (section 2.3), la notion de matrices aura aux images couleur et aux couleurs floues. Pour cela, nous avons employé deux stratégies différentes. Dans la première, une matrice aura des niveaux des composantes couleur (Color Level Aura Matrix) est calculée pour chaque composante de l'image de manière indépendante (approche marginale). La seconde stratégie consiste à calculer six autres matrices (matrices inter composantes) sur des paires des composantes couleur (approche opposée ou conjointe). Avec ces deux stratégies, trois ou six matrices de grande taille sont nécessaires pour décrire une image couleur. L'exploitation des éléments de ces matrices sous forme d'attributs dans le cadre de la classification d'images couleur texturées se heurte alors au problème d'espace mémoire et temps de calcul. Pour remédier à ce problème, nous proposons dans ce chapitre d'exploiter la stratégie compacte en utilisant une seule matrice aura couleur dans les deux contextes (net et flou) au lieu des trois ou six précédentes, qui prend en considération l'interaction spatiale entre les couleurs, tout en réduisant l'espace mémoire. Pour démontrer l'efficacité de ces matrices, nous avons réalisé une classification des textures couleur et les avons comparées à d'autres attributs de l'état de l'art .

3.2 Matrice aura des couleurs

Avant d'introduire la matrice aura des couleurs floues, il est important de définir d'abord la matrice aura des couleurs dans un contexte net (crisp).

3.2.1 Ensemble de couleur

Soit $\mathbf{I}$ une image couleur définie dans un espace RGB[1], sur une grille finie $\mathtt{S}$, où chaque pixel $\mathbf{s} \in \mathtt{S}$ est caractérisé par sa couleur $\mathbf{I}(\mathbf{s})$. Pour une couleur donnée $\mathbf{x}$ dans l'espace RGB, nous définissons l'ensemble de couleur $\mathtt{S}_\mathbf{x}$ comme l'ensemble des pixels de couleur $\mathbf{x}$, tel que $\mathtt{S}_\mathbf{x} = \{\mathbf{s} \in \mathtt{S} \,|\, \mathbf{I}(\mathbf{s}) = \mathbf{x}\}$. Considérons deux couleurs $\mathbf{x}$ et $\mathbf{x}'$ appartenant à l'espace RGB ($\mathbf{x}$,$\mathbf{x}' \in$ RGB), $\{\mathtt{S}_\mathbf{x}\}_{\mathbf{x}\in\text{RGB}}$ forme une partition de $\mathtt{S}$, de sorte que $\bigcup_{\mathbf{x}\in\text{RGB}} \mathtt{S}_\mathbf{x} = \mathtt{S}$ et $\mathtt{S}_\mathbf{x} \cap \mathtt{S}_{\mathbf{x}'} = \emptyset$ pour $\mathbf{x} \neq \mathbf{x}'$.

3.2.2 Ensemble aura des couleurs

L'ensemble aura des couleurs $\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_\mathbf{x})$, de l'ensemble des pixels de couleur $\mathbf{x}$ ($\mathtt{S}_\mathbf{x}$) par rapport à un autre ensemble des pixels de couleur $\mathbf{x}'$ ($\mathtt{S}_{\mathbf{x}'}$) est défini comme suit:

$$\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_\mathbf{x}) = \bigcup_{\mathbf{s}\in\mathtt{S}_\mathbf{x}} (\mathtt{N}_\mathbf{s} \cap \mathtt{S}_{\mathbf{x}'}), \tag{3.1}$$

$\mathtt{N}_\mathbf{s}$ étant l'ensemble des pixels voisins du pixel $\mathbf{s}$.

Cette relation peut être réécrite comme suit :

$$\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_\mathbf{x}) = \bigcup_{\mathbf{s}\in\mathtt{S}} (\mathtt{N}_{\mathbf{s},\mathtt{S}_\mathbf{x}} \cap \mathtt{S}_{\mathbf{x}'}), \tag{3.2}$$

où $\mathtt{N}_{\mathbf{s},\mathtt{S}_\mathbf{x}}$ est l'ensemble de pixels voisins de chaque pixel $\mathbf{s} \in \mathtt{S}_\mathbf{x}$, tel que :

$$\mathtt{N}_{\mathbf{s},\mathtt{S}_\mathbf{x}} = \{\mathbf{r} \in \mathtt{S}, (\mathbf{s} \in \mathtt{S}_\mathbf{x}) \wedge (\mathbf{r} \in \mathtt{N}_\mathbf{s})\}. \tag{3.3}$$

1. D'autres espaces couleur peuvent être utilisés.

Plusieurs types de voisinages, comme ceux présentés dans le chapitre 2, section 2.2.1, peuvent être exploités. A titre d'exemple, le voisinage $\mathtt{N}_{\mathbf{s}}$ défini par les huit plus proches voisins est defini par : $\mathtt{N}_{\mathbf{s}} \doteq \{\mathbf{r} \in \mathtt{S}, \|\mathbf{r} - \mathbf{s}\|_\infty \leq 1\}$.

L'ensemble aura $\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_{\mathbf{x}})$ est composé des pixels de couleur $\mathbf{x}'$ qui sont présents au voisinage des pixels de couleur $\mathbf{x}$. Il décrit la présence d'un sous-ensemble $\mathtt{S}_{\mathbf{x}'}$ dans le voisinage du sous-ensemble $\mathtt{S}_{\mathbf{x}}$.

La figure 3.1 montre une image couleur, définie par trois sous-ensembles couleur $\mathtt{S}_{\mathbf{R}}$, $\mathtt{S}_{\mathbf{G}}$, et $\mathtt{S}_{\mathbf{B}}$ sur une grille $\mathtt{S}$ de taille 7×7. Pour le voisinage $\mathtt{N}_{\mathbf{s}}$, composé des huit pixels les plus proches de $\mathbf{s}$, l'ensemble aura couleur $\mathtt{A}_{\mathtt{S}_{\mathbf{G}}}(\mathtt{S}_{\mathbf{B}})$ de $\mathtt{S}_{\mathbf{B}}$ par rapport à $\mathtt{S}_{\mathbf{G}}$ est composé des pixels verts marqués par des cercles. Autrement dit, l'ensemble aura couleur $\mathtt{A}_{\mathtt{S}_{\mathbf{G}}}(\mathtt{S}_{\mathbf{B}})$ décrit la présence d'un sous-ensemble $\mathtt{S}_{\mathbf{G}}$ dans le voisinage du sous-ensemble $\mathtt{S}_{\mathbf{B}}$. Nous pouvons remarquer sur la figure 3.1 que $\mathtt{A}_{\mathtt{S}_{\mathbf{G}}}(\mathtt{S}_{\mathbf{B}}) \neq \mathtt{A}_{\mathtt{S}_{\mathbf{B}}}(\mathtt{S}_{\mathbf{G}})$, l'ensemble aura $\mathtt{A}_{\mathtt{S}_{\mathbf{B}}}(\mathtt{S}_{\mathbf{G}})$ de $\mathtt{S}_{\mathbf{G}}$ par rapport à $\mathtt{S}_{\mathbf{B}}$, composé des pixels bleus marqués par des losanges.

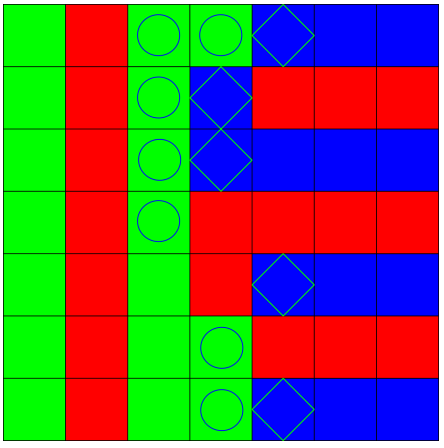


FIG. 3.1: Exemple d'un ensemble aura couleur: Image originale composée de 3 sous-ensembles de pixels couleur $\mathtt{S}_{\mathbf{R}}$,$\mathtt{S}_{\mathbf{G}}$, et $\mathtt{S}_{\mathbf{B}}$, et deux ensembles aura couleur $\mathtt{A}_{\mathtt{S}_{\mathbf{G}}}(\mathtt{S}_{\mathbf{B}})$ (cercles) et $\mathtt{A}_{\mathtt{S}_{\mathbf{B}}}(\mathtt{S}_{\mathbf{G}})$ (losanges).

3.2.3 Mesures et matrices aura des couleurs

Comme pour les GLAMs, les deux mesures aura qui caractérisent la taille d'un ensemble aura des couleurs, ainsi que leurs matrices aura des couleurs associées peuvent être définies.

3.2.3.1 Mesure et matrice aura locale des couleurs

La mesure aura locale d'un sous-ensemble de couleur $\mathtt{S}_{\mathbf{x}}$ par rapport à un autre sous-ensemble de couleur $\mathtt{S}_{\mathbf{x}'}$ pour une fonction de voisinage $\mathtt{N}_{\mathbf{s}}$ estime le nombre de fois qu'un pixel $\mathbf{s} \in \mathtt{S}_{\mathbf{x}}$ comporte dans son voisinage un pixel $\mathbf{r} \in \mathtt{S}_{\mathbf{x}'}$. Cette mesure est décrite par la formule suivante :

$$m_l\left(\mathbf{x},\mathbf{x}'\right) = \sum_{\mathbf{s}\in\mathtt{S}_{\mathbf{x}}} \left|\mathtt{N}^{\mathbf{s}}_{\mathtt{S}_{\mathbf{x}}} \cap \mathtt{S}_{\mathbf{x}'}\right|, \tag{3.4}$$

La mesure aura locale des couleurs évalue la fréquence d'apparition d'une couleur $\mathbf{x}'$ dans le voisinage de la couleur $\mathbf{x}$. Elle donne une interprétation sur la quantité de mélange de deux sous-ensembles $\mathtt{S}_{\mathbf{x}}$ et $\mathtt{S}_{\mathbf{x}'}$. Une valeur élevée de $m_l\left(\mathbf{x},\mathbf{x}'\right)$ indique que les deux sous-ensembles de couleur $\mathtt{S}_{\mathbf{x}}$ et $\mathtt{S}_{\mathbf{x}'}$ sont fortement mélangés alors qu'une faible valeur indique qu'ils sont séparés.

La matrice aura locale des couleurs (Local Color Aura Matrix : CAM$_l$) notée M_l est une représentation des mesures aura locale entre toutes les paires possibles d'ensembles des couleurs $(\mathtt{S}_{\mathbf{x}},\mathtt{S}_{\mathbf{x}'})$, $(\mathbf{x},\mathbf{x}') \in \mathrm{RGB}^2$. Elle est définie comme suit:

$$M_l = \left[m_l\left(\mathbf{x},\mathbf{x}'\right)\right], \tag{3.5}$$

Ainsi, une matrice aura des couleurs capte les variations de couleurs locales et offre une représentation compacte des caractéristiques de textures de l'image.

Il est facile de voir que la CAM$_l$ est la généralisation d'une matrice de co-occurrence des couleurs (Color Cooccurrence Matrix, CCM) [6], comme dans le cas des niveaux de gris (Section 2.2.3.3).

3.2.3.2 Mesure et matrice aura cardinales des couleurs

La mesure aura locale des couleurs, à l'instar de son homologue des niveaux de gris [7], n'évalue pas réellement le nombre de pixels appartenant à l'ensemble $\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_{\mathbf{x}})$. La taille de l'ensemble aura peut-être mesurée d'une façon simple par le biais de son cardinal [8]. Nous proposerons donc la mesure aura cardinale des couleurs et la désignerons par $m_c(\mathbf{x},\mathbf{x}')$, telle

que :

$$m_c(\mathbf{x},\mathbf{x}') = \left|\mathtt{A}_{\mathtt{S}_{\mathbf{x}'}}(\mathtt{S}_{\mathbf{x}})\right| . \tag{3.6}$$

Les mesures aura cardinales pour toutes les paires possibles d'ensembles de couleurs $(\mathtt{S}_{\mathbf{x}},\mathtt{S}_{\mathbf{x}'})$, $(\mathbf{x},\mathbf{x}') \in \mathrm{RGB}^2$, sont regroupées dans une matrice appelée matrice aura cardinale des couleurs ou Cardinal Color Aura Matrix (CAM_c) et notée M_c :

$$M_c = \left[m_c\left(\mathbf{x},\mathbf{x}'\right)\right], \tag{3.7}$$

Les éléments de cette matrice peuvent être a priori utilisés comme attributs pour caractériser des textures. Cependant, lorsqu'on travaille avec des images dont les composantes couleur sont définies sur 256 niveaux, ce qui est généralement le cas, le nombre total de couleurs possibles atteint 256^3, ce qui signifie que les matrices seraient de la taille de $256^3 \times 256^3$. La capacité de mémoire nécessaire pour stocker de telles matrices devient alors énorme et pose un défi pratique. Une manière simple d'éviter cet inconvénient est de réduire le nombre de couleurs (voir Section 3.3.1).

3.3 Matrice aura des couleurs floues

Nous montrons dans cette section comment utiliser une seule matrice aura des couleurs floues pour caractériser les images couleur texturées. Tout d'abord, nous commençons par définir les couleurs floues et l'ensemble aura des couleurs floues. Ensuite, nous introduisons les mesures aura locales et cardinales spécifiques aux couleurs floues, puis la matrice aura des couleurs floues (Fuzzy Color Aura Matrix, FCAM).

3.3.1 Couleur floue

Dans l'espace RGB, chaque couleur est représentée par des valeurs numériques pour les composantes rouge, vert et bleu, qui peuvent varier de 0 à 255, ce qui donne 256^3 combinaisons de couleurs possibles. Cependant, parmi ces 256^3 couleurs, seules quelques-unes sont retenues ; elles correspondent aux équivalents nets des couleurs floues que nous souhaitons

considérer. Le choix de ces couleurs peut être réalisé par l'une des nombreuses méthodes de quantification des couleurs [139]. En ce qui nous concerne, nous avons opté pour la technique de quantification uniforme pour sa rapidité et sa simplicité de mise en œuvre. Soit $\mathscr{C}$ un petit sous-ensemble de C couleurs retenues parmi les 256^3 couleurs possibles. La gamme complète [0,255] de chaque composante couleur $k \in \{R,G,B\}$ est divisée en C^k intervalles disjoints ($[0,L^k-1]$, $[L^k,2L^k-1],\cdots,[(C^k-1)L^k,255]$ centrés sur les nombres $\left\{\lceil\frac{(L^k-1}{2}\rceil,\lceil\frac{3L^k-1}{2}\rceil,\ldots,\lceil 256-\frac{L^k+1}{2}\rceil\right\}$ (arrondis aux entiers supérieurs), où $L^k \doteq 256/C^k$ est la largeur de chaque intervalle. Les C^k centres de ces intervalles définissent la k-ème composante couleur dans $\mathscr{C}$. Les nombres C^R, C^G et C^B sont choisis de sorte que le nombre de couleurs $C = C^R \cdot C^G \cdot C^B$ soit bien inférieur à 256^3.

Dans le cadre flou [6], une couleur floue $\tilde{\mathbf{c}}$ est caractérisée par sa fonction d'appartenance $\mu_{\tilde{\mathbf{c}}}$: RGB $\longrightarrow$ [0,1]. Le degré d'appartenance $\mu_{\tilde{\mathbf{c}}}(\mathbf{x})$ de toute couleur $\mathbf{x} \in$ RGB peut être défini à l'aide de la distance euclidienne entre l'équivalent net $\mathbf{c} \in \mathscr{C}$ et la couleur floue $\tilde{\mathbf{c}}$. Les fonctions d'appartenance utilisées pour les niveaux de gris flous ou les niveaux des composantes couleurs floues peuvent être généralisées aux couleurs floues comme :

– la fonction d'appartenance nette:

$$\mu_{\tilde{\mathbf{c}}}(\mathbf{x}) = \begin{cases} 1 & \text{si } \| \mathbf{x}-\mathbf{c} \|_\infty \leq \lfloor\frac{L^k}{2}\rfloor, \\ 0 & \text{sinon,} \end{cases} \tag{3.8}$$

– la fonction gaussienne symétrique:

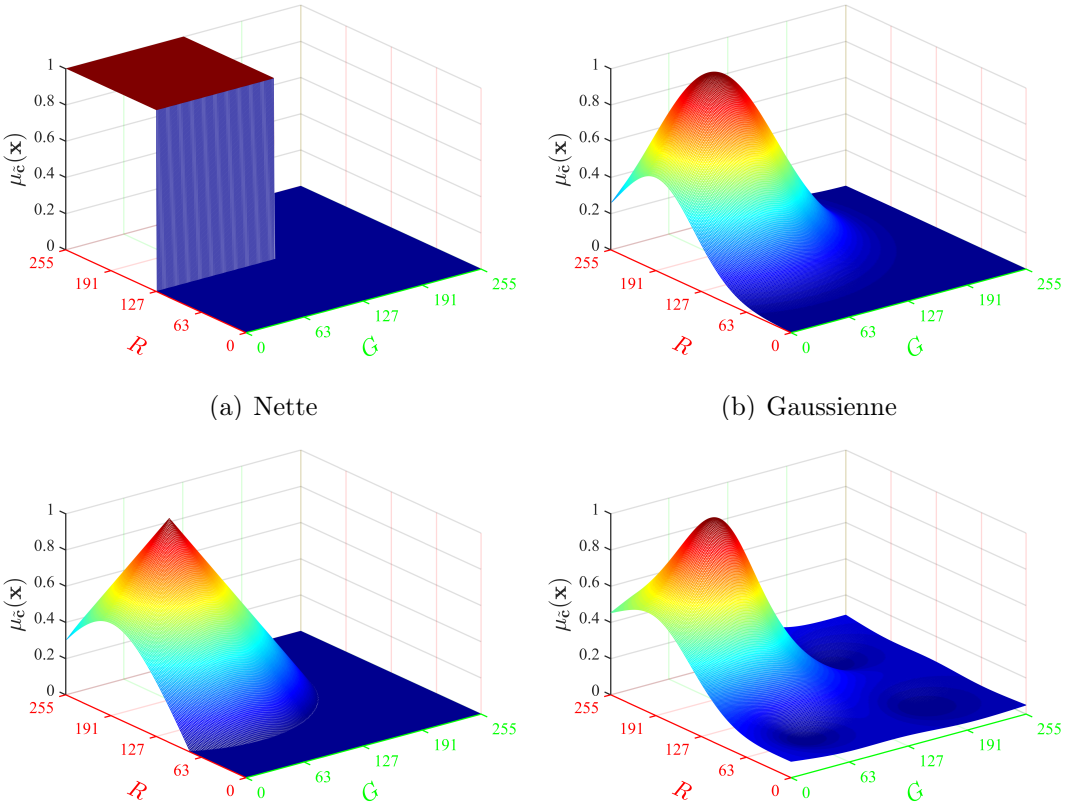
$$\mu_{\tilde{\mathbf{c}}}(\mathbf{x}) = \exp\left(-\frac{\| \mathbf{x}-\mathbf{c} \|_2^2}{2\boldsymbol{\alpha}^2} \right), \tag{3.9}$$

– la fonction triangulaire symétrique:

$$\mu_{\tilde{\mathbf{c}}}(\mathbf{x}) = \max\left(1 - \frac{\| \mathbf{x}-\mathbf{c} \|_2}{\boldsymbol{\beta}}, 0\right), \tag{3.10}$$

– La fonction FCM tirée de l'algorithme de classification fuzzy C-means (FCM) [140] :

$$\mu_{\tilde{\mathbf{c}}}(\mathbf{x}) = \frac{1}{\sum_{\mathbf{c}' \in \mathscr{C}} \left(\frac{\|\mathbf{x}-\mathbf{c}\|_2}{\|\mathbf{x}-\mathbf{c}'\|_2}\right)^{\frac{2}{\zeta-1}}}. \tag{3.11}$$



(a) Nette (b) Gaussienne

(c) Triangulaire (d) FCM

FIG. 3.2: Fonctions d'appartenances $\mu_{\tilde{\mathbf{c}}}$ à $\mathbf{c} = (192,64,64)$ avec $C^R = C^G = C^B = 2$.

$\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$ sont des vecteurs de valeurs réelles positives utilisés pour contrôler l'étendue de la couleur floue, et ζ est un nombre réel supérieur à 1. Dans ce travail, nous avons fixé $\boldsymbol{\beta} = (L^R,L^G,L^B)$, $\boldsymbol{\alpha} = \boldsymbol{\beta}/\sqrt{2\ln(2)}$, et $\zeta = 2$. Les valeurs des paramètres $\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$ sont choisies de telle façon que $\mu_{\tilde{\mathbf{c}}}(\mathbf{x}) = 0.5$ aux limites de chaque domaine centré en $\mathbf{c}$ et de largeur $(L^R,L^G,L^B)^\intercal$.

La figure (3.2) montre les formes des quatre fonctions d'appartenance calculées à $\mathbf{c} = (192,64,64)$, dans le plan (R,G).

3.3.2 Ensemble de couleur floue

Un ensemble de couleur floue $\mathtt{S}_\mathbf{c}$, $\mathbf{c} \in \mathscr{C}$, est un sous-ensemble flou défini par sa fonction d'appartenance $\mu_{\mathtt{S}_\mathbf{c}}$. Le degré d'appartenance $\mu_{\mathtt{S}_\mathbf{c}}(\mathbf{s})$ de chaque pixel $\mathbf{s} \in \mathtt{S}$ à $\mathtt{S}_\mathbf{c}$ est défini par

le degré d'appartenance $\mu_{\tilde{\mathbf{c}}}(\mathbf{I}(\mathbf{s}))$ de sa couleur $\mathbf{I}(\mathbf{s}) \in \text{RGB}$ à la couleur floue $\tilde{\mathbf{c}}$ [6]:

$$\mu_{\mathtt{S}_\mathbf{c}}(\mathbf{s}) = \mu_{\tilde{\mathbf{c}}}(\mathbf{I}(\mathbf{s})). \tag{3.12}$$

3.3.3 Ensemble aura des couleurs floues

Soient $\mathtt{S}_\mathbf{c}$ et $\mathtt{S}_{\mathbf{c}'}$ deux sous-ensembles de couleurs floues $(\mathbf{c},\mathbf{c}') \in \mathscr{C}^2$ définis par leurs fonctions d'appartenance $\mu_{\mathtt{S}_\mathbf{c}}$ et $\mu_{\mathtt{S}_{\mathbf{c}'}}$. L'ensemble aura des couleurs floues $\mathtt{A}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})$ est l'ensemble aura de $\mathtt{S}_\mathbf{c}$ par rapport à $\mathtt{S}_{\mathbf{c}'}$. C'est aussi un sous-ensemble flou défini par le degré d'appartenance de chaque pixel $\mathbf{r} \in \mathtt{S}$:

$$\mu_{\mathtt{A}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})}(\mathbf{r}) = \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{\mathtt{S}_\mathbf{c}}(\mathbf{s}),n_\mathbf{s}(\mathbf{r})\right)\right],\mu_{\mathtt{S}_{\mathbf{c}'}}(\mathbf{r})\right\}. \tag{3.13}$$

La fonction de voisinage floue $n_\mathbf{s}(\mathbf{r})$ représente le degré d'appartenance de $\mathbf{r}$ au voisinage de $\mathbf{s}$.

La relation (3.13) peut être obtenue à partir de l'ensemble aura des couleurs nettes de l'équation (3.2) et en suivant les mêmes transformations utilisées dans le cas des ensembles aura des niveaux de gris (Section 2.2.1).

3.3.4 Mesures et matrices aura des couleurs floues

Les deux mesures aura définies pour les niveaux de gris flous sont étendues aux couleurs floues.

3.3.4.1 Mesure et matrice aura locale de couleurs floues

La mesure aura locale des couleurs floues d'un ensemble de couleur floue $\mathtt{S}_\mathbf{c}$ par rapport à un autre ensemble de couleur floue $\mathtt{S}_{\mathbf{c}'}$ est donné par :

$$\tilde{m}_l\left(\mathbf{c},\mathbf{c}'\right) = \sum_{\mathbf{r}\in\mathtt{S}}\sum_{\mathbf{s}\in\mathtt{S}}\min\left\{\min\left[\mu_{\mathtt{S}_\mathbf{c}}(\mathbf{s}),n_\mathbf{s}(\mathbf{r})\right],\mu_{\mathtt{S}_{\mathbf{c}'}}(\mathbf{r})\right\}. \tag{3.14}$$

On définit alors la matrice aura locale des couleurs floues (Local Fuzzy Color Aura Matrix : FCAM$_l$), notée $\tilde{M}_l$, une matrice de dimension $C \times C$ regroupant des mesures aura locales de toutes les paires de couleurs floues $(\mathbf{c},\mathbf{c}') \in \mathscr{C}^2$:

$$\tilde{M}_l = [\tilde{m}_l\,(\mathbf{c},\mathbf{c}')]. \tag{3.15}$$

Tout comme la CAM$_l$ est une généralisation de la matrice de co-occurrence des couleurs (CCM), la FCAM$_l$ est également une généralisation de la matrice de co-occurrence des couleurs floues (Fuzzy Color Cooccurrence Matrix, FCCM), proposée par Ledoux et al. [6].

3.3.4.2 Mesure et matrice aura cardinale des couleurs floues

La mesure aura cardinale du sous-ensemble de couleur floue $\mathtt{S}_{\mathbf{c}}$ par rapport au sous-ensemble de couleur floue $\mathtt{S}_{\mathbf{c}'}$ est donnée par le cardinal du sous-ensemble aura des couleurs floues :

$$\begin{aligned}
\tilde{m}_c\,(\mathbf{c},\mathbf{c}') &= \left|\mathtt{A}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})\right| \\
&= \sum_{\mathbf{r}\in\mathtt{S}} \mu_{\mathtt{A}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})}(\mathbf{r}) \\
&= \sum_{\mathbf{r}\in\mathtt{S}} \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\Big[\min\left(\mu_{\mathtt{S}_{\mathbf{c}}}(\mathbf{s}),n_{\mathbf{s}}(\mathbf{r})\right)\Big],\mu_{\mathtt{S}_{\mathbf{c}'}}(\mathbf{r})\right\}.
\end{aligned} \tag{3.16}$$

Toutes les mesures aura cardinales des C^2 paires de couleurs $(\mathbf{c},\mathbf{c}') \in \mathscr{C}^2$ peuvent être regroupées dans une matrice de taille $C \times C$, appelée matrice aura cardinale des couleurs floues ou Cardinal Fuzzy Color Aura Matrix ($FCAM_c$), notée $\tilde{M}_c$:

$$\tilde{M}_c = [\tilde{m}_c\,(\mathbf{c},\mathbf{c}')]. \tag{3.17}$$

Notons que, pour définir la $FCAM_c$, l'image RGB est définie sur quelques couleurs parmi les 256^3 possible à savoir, l'ensemble $\mathscr{C}$ de C couleurs tel que $C \ll 256^3$. Cela donne une matrice $FCAM_c$ compacte de dimension $C \times C$ très réduite, dont les éléments représentent les attributs de texture couleur.

3.4 Classification des textures couleur à base des FCAMs

Pour évaluer la pertinence des FCAMs à discriminer des textures couleurs, nous avons effectué une classification supervisée d'images de texture couleur. Cette démarche, représentée par le schéma de la figure 3.3, se déroule en deux étapes distinctes. La première étape consiste à extraire les attributs de texture couleur (FCAMs) pour chaque image de la base d'apprentissage et de la base de test. La seconde étape, dite de classification (décision), utilise le classifieur supervisé PPV (plus proche voisin) pour prédire la classe de chaque image de la base de test en se basant sur des attributs FCAMs de la base d'apprentissage. Le taux de bonne classification (accuracy), défini comme le rapport entre le nombre d'images de la base de test correctement classées sur le nombre total d'images de la base de test, est alors utilisé comme indice d'évaluation de la pertinence des FCAMs.

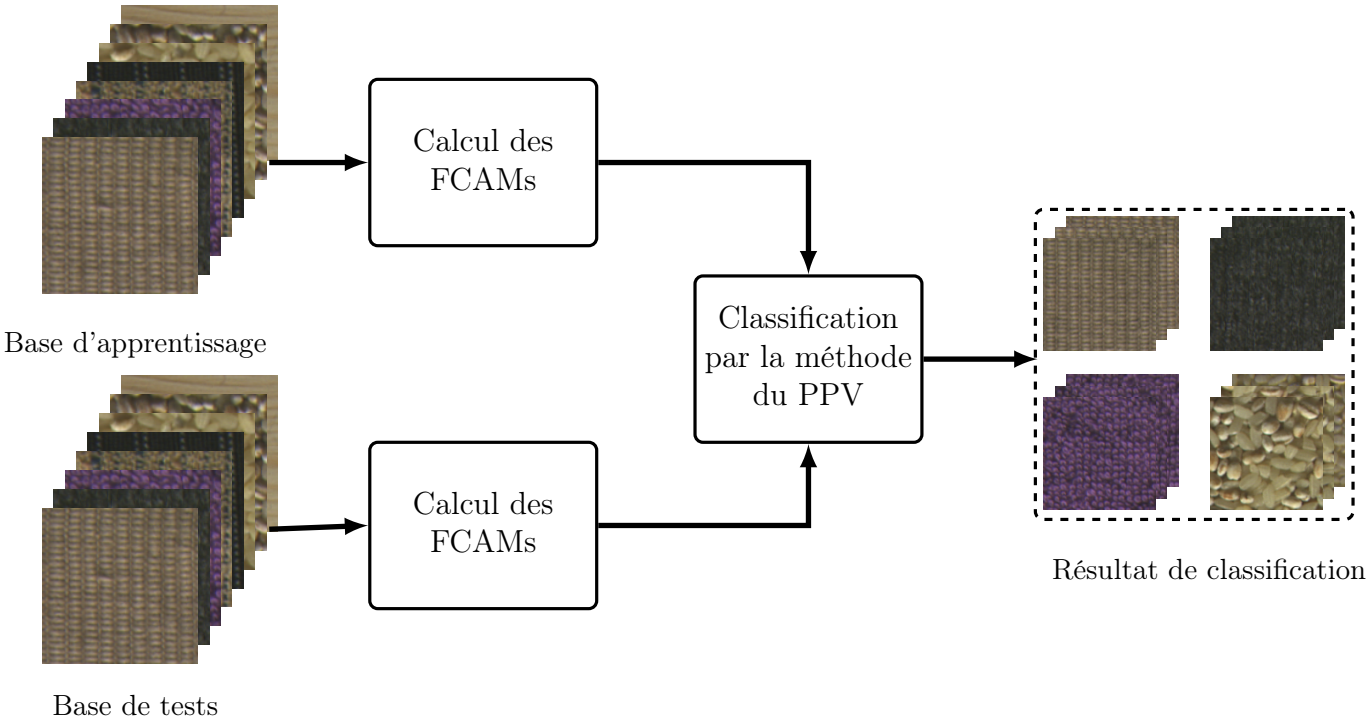


FIG. 3.3: Organigramme de la classification des textures couleur à base des FCAMs

Lors de la phase d'extraction des attributs, le calcul des FCAMs est effectué à partir des équations (3.14) et (3.15) pour la FCAM_l, et (3.16) et (3.17) pour la FCAM_c.

Il y a lieu de noter que ces FCAMs sont auparavant normalisées en divisant chaque élément d'une FCAM par la somme de tous ses éléments, comme suit :

$$\bar{M} = \frac{1}{\sum_{(\mathbf{y},\mathbf{y}')\in\mathscr{C}^2} \tilde{m}\,(\mathbf{y},\mathbf{y}')} \times [\tilde{m}\,(\mathbf{c},\mathbf{c}')]. \tag{3.18}$$

Les éléments de chaque FCAM normalisée sont mémorisés sous forme d'un vecteur d'attributs.

Durant la phase de classification, chaque image de la base de test est caractérisée par un vecteur d'attributs dont les éléments correspondent aux éléments de sa FCAM normalisée. Ces vecteurs d'attributs sont alors soumis au classifieur plus proche voisin (PPV), qui affecte chaque image de test à l'une des classes de textures couleur disponibles.

L'algorithme de classification du plus proche voisin consiste à calculer la distance entre les vecteurs d'attributs de chaque image de la base de test et les vecteurs d'attributs des images d'apprentissage. Une image de la base de test est alors affectée à la classe de l'image de la base d'apprentissage à laquelle la distance est la plus petite. Comme distance, nous avons utilisé la mesure de similarité d'intersection.

Rappelons que la mesure de similarité entre deux images I_{test} et I_{train} est donnée par l'intersection entre leurs FCAMs normalisées, et qui s'exprime comme suit :

$$\mathrm{Sim}(I_{test},I_{train}) = \sum_{\mathbf{c}} \sum_{\mathbf{c}'} \min\left\{\tilde{M}[I_{test}],\tilde{M}[I_{train}]\right\} \tag{3.19}$$

Notons également que le classifieur PPV se prête bien à l'évaluation de la pertinence des attributs car c'est un algorithme simple qui ne dispose d'aucun paramètre en dehors du type de distance et ne nécessite aucune phase d'entraînement.

3.5 Évaluation des FCAMs

Dans cette section, nous évaluons la pertinence des attributs de texture couleur, représentés par les éléments de la matrice aura (cardinale ou locale) des couleurs floues, dans le cadre de la classification des textures couleur. Dans un premier temps, nous examinons les paramètres

utilisés dans le calcul des FCAMs, puis nous comparons leurs performances avec celles des autres variantes, à savoir les MFCLAMs et OFCLAMs. Par la suite, nous étudions la robustesse des FCAMs vis-à-vis des conditions d'acquisition comme la rotation, la résolution et les dégradations (bruit, flou). Au final, nous évaluons les performances des FCAMs en comparaison avec d'autres méthodes d'extraction des attributs. Pour effectuer toutes ces expériences, nous avons utilisé la base *Outex-TC-13* et ses dérivées (*TC-30, TC-31, TC-32, TC-33*), ainsi que la base *KTH-TIPS*.

3.5.1 Base *Outex-TC-13*

Outex est une base d'images de textures couleur acquises avec une caméra 3-CCD dans des conditions contrôlées. Cette base est une collection de matériaux hétérogènes tels que le carton, le tissu, le papier, la laine, etc. Elle contient 68 images originales de textures de taille 746×538 pixels. La base *Outex-TC-13* a été générée en subdivisant chaque image originale de la base *Outex* en 20 images de taille 128×128 pixels, sans chevauchement. Elle contient par conséquent 1360 images. Parmi ces 1360 images de cette base, 680 images sont réservées pour l'apprentissage et les 680 images restantes pour les tests [141]. La figure 3.4 montre quelques images de la base *Outex*, où chaque image représente une classe de texture.

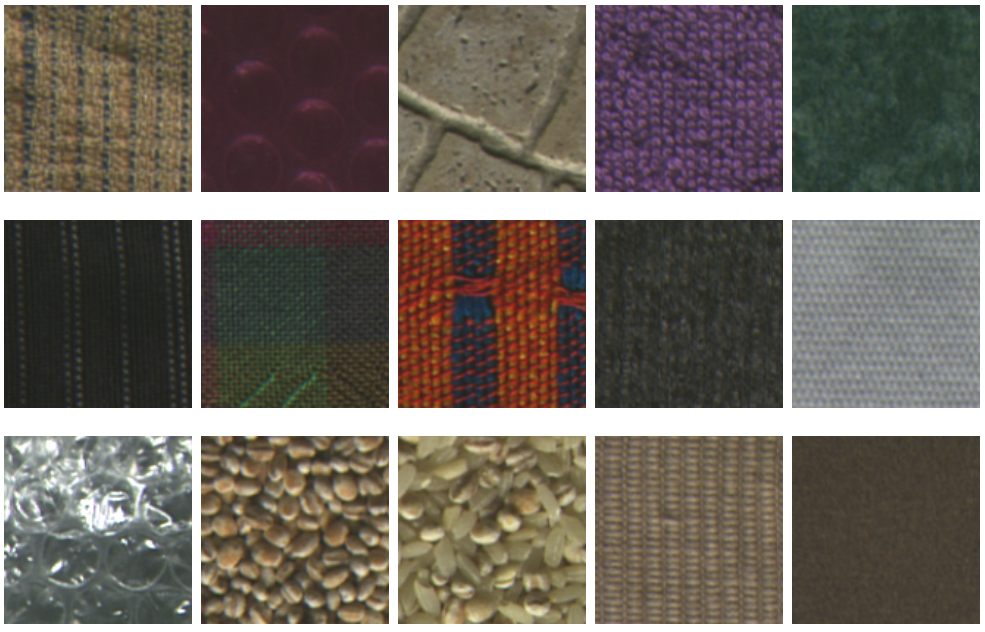


FIG. 3.4: Exemples de textures couleur de la base de données *Outex*. Chaque image illustre une classe de texture.

3.5.2 Base *KTH-TIPS*

La base *KTH-TIPS* comprend 10 classes de textures, comprenant du papier de verre, de l'aluminium froissé, de la mousse de polystyrène, de l'éponge, du velours côtelé, du lin, du coton, du pain brun, de la peau d'orange et des biscuits salés. Chaque classe comprend 81 images de taille 200×200 pixels, ce qui représente au total 810 images. Comme la base *KTH-TIPS* ne dispose pas d'ensembles d'apprentissage et de tests fixes, nous avons sélectionné de manière aléatoire 40 images de chaque classe pour l'apprentissage, tandis que les autres sont utilisées pour les tests. Par conséquent, nous rapportons la moyenne des taux de classification calculés sur 50 répartitions aléatoires d'apprentissage/test. La figure 3.5 montre un exemple d'image pour chaque classe de la base *KTH-TIPS*.

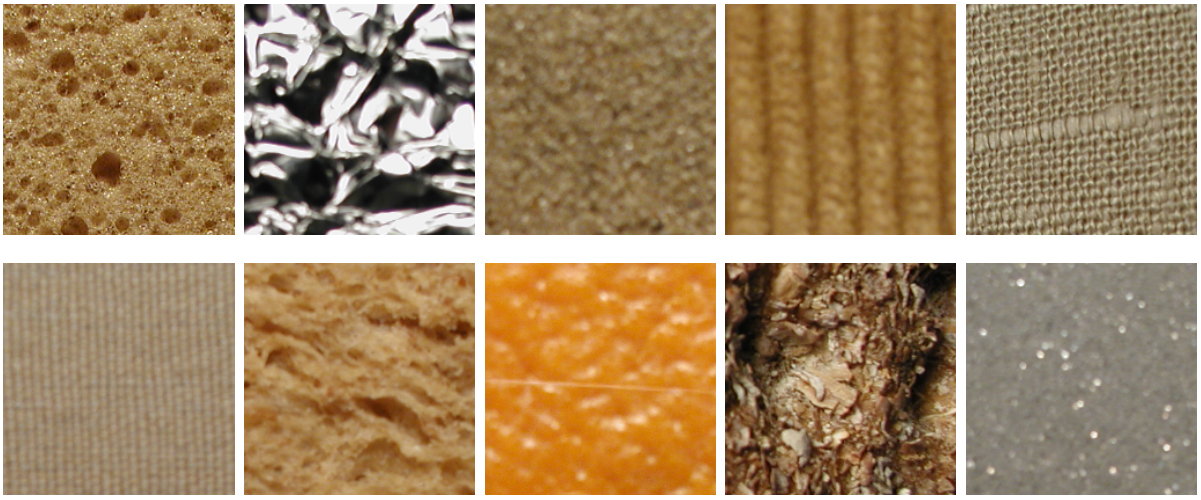


FIG. 3.5: Exemples de textures couleur de la base de données *KTH-TIPS*. Chaque image illustre une classe de texture.

3.5.3 Paramètres de la FCAM

La caractérisation des textures à base des FCAMs dépend de deux paramètres. Le premier est le nombre de couleurs floues C, le second est la fonction d'appartenance $\mu_{\tilde{\mathbf{c}}}$. Nous avons exploré différentes valeurs pour ces deux paramètres et avons évalué les performances de la classification pour chaque combinaison des valeurs utilisées. Ces performances sont mesurées en termes de taux de classification (accuracy) et de temps de calcul.

TAB. 3.1: Comparaison des résultats des taux de classification (%) obtenus avec la $FCAM_l$ et la $FCAM_c$ en utilisant la mesure d'intersection sur la base *Outex-TC-13*.

C	Crisp		FCM		Gaussienne		Triangulaire	
	CAM_l	CAM_c	$FCAM_l$	$FCAM_c$	$FCAM_l$	$FCAM_c$	$FCAM_l$	$FCAM_c$
4	30.74	**33.68**	77.21	**81.18**	78.09	**81.47**	56.76	**62.06**
8	35.15	**35.88**	80.44	**83.38**	87.06	**88.68**	**90.88**	90.29
12	47.06	**49.71**	85.74	**88.68**	85.15	**89.26**	83.24	**85.00**
16	56.62	**59.56**	**89.12**	**89.12**	86.76	**88.09**	85.15	**87.35**
24	69.71	**72.50**	**90.74**	**90.74**	88.24	**89.26**	85.88	**88.24**
32	71.47	**76.76**	90.88	**91.62**	88.97	**89.56**	86.32	**88.82**
64	81.03	**82.94**	91.62	**92.06**	91.76	**92.21**	89.79	**92.35**

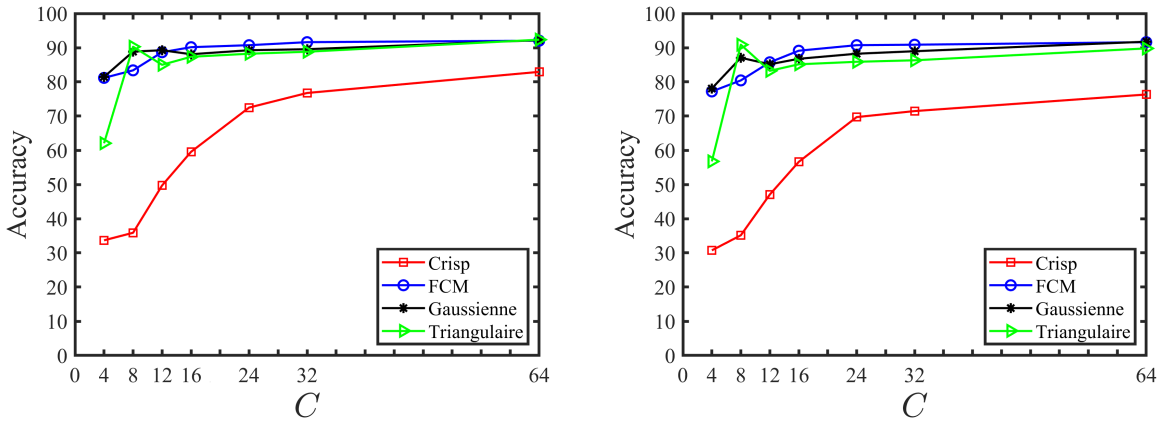
Le nombre de couleurs floues C est fixé par le nombre de niveaux C^R, C^G et C^B considérés pour chaque composante couleur R, G et B (Section 3.3.1). Nous avons testé sept nombres de couleurs : $C = 2 \times 2 \times 1$, $C = 2 \times 2 \times 2$, $C = 2 \times 3 \times 2$, $C = 2 \times 4 \times 2$, $C = 3 \times 4 \times 2$, $C = 4 \times 4 \times 2$, et $C = 4 \times 4 \times 4$. Ces nombres limités de couleurs permettent de réduire la taille de la FCAM (cardinale ou locale) à respectivement 4×4, 8×8, 12×12, 16×16, 24×24, 32×32, et 64×64. Il est important de souligner que nous avons accordé une préférence à la composante de couleur G par rapport à R et B car elle est similaire à la luminance. D'autres combinaisons de couleurs ont fourni des taux de classification similaires.

En ce qui concerne la fonction d'appartenance $\mu_{\tilde{\mathbf{c}}}$, nous avons examiné les quatre fonctions (Crisp, FCM, Gaussienne et Triangulaire), présentées dans la section 3.3.1.

Le tableau 3.1 regroupe les taux de classification en pourcentage obtenus par la méthode de classification basée sur la $FCAM_l$ et la $FCAM_c$ pour différentes fonctions d'appartenance (Crisp, FCM, Gaussienne et Triangulaire) et différents nombres de couleurs (C = 4, 8, 12, 16, 24, 32, 64) sur la base d'images *Outex-TC-13*. Rappelons que dans le cas d'une fonction d'appartenance Crisp, la FCAM est similaire à une CAM (color aura matrix). Les meilleures performances pour chaque nombre de couleurs C sont mis en gras.

Pour interpréter ces résultats, nous proposons d'examiner ce tableau en répondant aux trois questions suivantes:

- La mesure aura des couleurs floues (FCAM) offre-t-elle de meilleurs résultats que la mesure aura des couleurs nettes ou crisp (CAM) ?



(a) FCAM$_c$ (b) FCAM$_l$

FIG. 3.6: Taux de classification (%) obtenus avec la FCAM$_c$ et la FCAM$_l$ sur la base *Outex-TC-13* en fonction du nombre de couleurs (C) pour différentes fonctions d'appartenance.

- Comment le nombre de couleurs floues C influe-t'il sur les taux de classification des FCAMs?
- La FCAM cardinale est-elle meilleure que la FCAM locale?

Pour nous aider à répondre à ces questions, nous avons affiché sur la figure 3.6, les courbes de variation du taux de classification (accuracy) en fonction du nombre de couleurs floues C, et ce pour chaque type de fonction d'appartenance (Crisp, FCM, Gaussienne et Triangulaire).

- *Flou vs. Crisp (FCAM vs. CAM) :* Il apparaît clairement sur la figure 3.6 que les fonctions d'appartenance floues (FCM, Gaussienne et Triangulaire) donnent de meilleurs résultats que la fonction Crisp quel que soit le nombre de couleurs utilisé. Les résultats de la CAM avec la fonction Crisp augmentent généralement avec le nombre de couleurs, mais restent inférieurs à ceux des FCAMs et ce pour les trois fonctions d'appartenance floues. Ceci prouve l'intérêt du flou par rapport au crisp.

- *Influence du nombre de couleurs floues C :* Globalement, l'accuracy des deux FCAMs augmentent avec le nombre de couleurs C pour toutes les fonctions d'appartenance jusqu'à atteindre leurs maximums pour $C = 64$. Cependant, lorsque le nombre de couleurs $C \geq 12$, les fonctions d'appartenance FCM et Gaussienne conduisent à des taux de classification presque similaires. La fonction Triangulaire, en revanche, atteint

un taux de classification maximal de 90.88% pour la FCAM$_l$ et le second meilleur taux 90.29% pour la FCAM$_c$ avec seulement $C = 8$ couleurs.

- *$FCAM_c$ vs. $FCAM_l$:* À partir du tableau 3.1, nous remarquons que pour cette base d'images (*Outex-TC-13*), les taux de classification obtenus avec la FCAM$_c$ (indiqués en gras) sont légèrement supérieurs à ceux obtenus avec la FCAM$_l$, quels que soient le nombre de couleurs et le type de la fonction d'appartenance.

3.5.4 Comparaison entre les trois stratégies

Les FCAMs (locale et cardinale) ont été élaborées selon la stratégie compacte. Nous allons, dans cette section, confronter leurs performances aux autres matrices aura floues couleur élaborées selon la stratégie marginale (MFCLAM$_c$ et MFCLAM$_l$) et la stratégie opposée (OFCLAM$_c$ et OFCLAM$_l$), proposées dans le chapitre 2. Rappelons qu'une seule matrice compacte FCAM de dimension $(C \times C)$ est nécessaire pour caractériser une image couleur, contre trois matrices MFCLAMs (une pour chaque composante couleur) de dimension $(C \times C)$ pour la stratégie marginale et six matrices OFCLAMs (inter-composantes) de dimension $(C \times C)$ dans le cas de la stratégie opposée. Pour les deux stratégies marginale et opposée, les attributs de texture couleur sont obtenus en concaténant les éléments des matrices correspondantes. La comparaison entre les trois stratégies est basée sur le taux de classification et le temps de calcul.

Le tableau 3.2 regroupe les résultats des taux de bonne classification ainsi que des temps de calcul obtenus avec les trois stratégies : compacte (FCAM$_c$ et FCAM$_l$), marginale (MFCLAM$_c$ et MFCLAM$_l$) et opposée (OFCLAM$_c$ et OFCLAM$_l$). Ces résultats sont obtenus sur la base d'images *Outex-TC-13*, avec les différentes fonctions d'appartenance (Crisp, FCM, Gaussienne et Triangulaire) et le nombre de couleurs $C = 8$. Les temps de calcul des deux étapes (caractérisation et classification) sont mesurés séparément, et les temps de caractérisation sont obtenus en moyenne sur 10 images. Les tests ont été réalisés sur un ordinateur équipé d'un processeur Intel Core i5 à 1,30 GHz et 16 Go de RAM.

À partir de ce tableau, nous comparons d'abord les deux mesures, cardinale et locale, pour chaque stratégie afin de déterminer laquelle est la plus intéressante en termes de précision.

TAB. 3.2: Résultats des taux de bonne classification (%) ainsi que des temps de calcul (en secondes) pour les trois stratégies : compacte ($FCAM_c$ et $FCAM_l$), marginale ($MFCLAM_c$ et $MFCLAM_l$) et opposée ($OFCLAM_c$ et $OFCLAM_l$) obtenus sur la base *Outex-TC-13*

Stratégie	Attributs	Fonction d'appartenance	Accuracy	Taille du vecteur d'attributs	Temps de calcul (s)		
					Caractérisation	Classification	Total
Compacte	$FCAM_c$	Crisp	**35.88**	$8^2 = 64$	175.20	68.87	**244.07**
		FCM	**83.38**		193.50	72.92	**266.42**
		Gaussienne	**88.68**		166.90	74.05	**240.95**
		Triangulaire	90.29*		160.70	70.10	**230.80**
	$FCAM_l$	Crisp	35.15		183.80	73.79	257.59
		FCM	80.44		208.90	74.80	283.70
		Gaussienne	87.06		179.90	87.06	266.96
		Triangulaire	**90.88***		175.50	78.40	253.90
Marginale	$MFCLAM_c$	Crisp	**88.82**	$3 \times 8^2 = 192$	245.70	330.21	575.91
		FCM	**90.15***		553.30	332.80	886.10
		Gaussienne	**89.85**		207.80	336.55	543.90
		Triangulaire	**89.26**		639.80	330.10	969.90
	$MFCLAM_l$	Crisp	87.94		245.10	317.92	563.02
		FCM	89.26*****		575.60	329.06	904.66
		Gaussienne	88.97		226.67	322.90	549.57
		Triangulaire	**89.26***		662.70	324.90	987.60
Opposée	$OFCLAM_c$	Crisp	**89.26**	$6 \times 8^2 = 384$	360.50	537.54	898.04
		FCM	**92.21***		691.10	536.45	1227.55
		Gaussienne	**91.91**		333.00	567.77	900.77
		Triangulaire	90.74		920.80	570.10	1490.90
	$OFCLAM_l$	Crisp	88.24		382.10	519.42	901.52
		FCM	91.62		719.30	556.02	1275.32
		Gaussienne	**91.91***		352.50	542.87	895.37
		Triangulaire	**90.88**		992.40	603.20	1595.60

Puis, nous comparons la pertinence de la stratégie compacte (FCAM) par rapport à d'autres stratégies, à savoir la stratégie marginale (MFCLAM) et opposée (OFCLAM). Au final, nous examinerons le temps de calcul de toutes ces méthodes.

- *Mesure cardinale vs. mesure locale :* Quelle que soit la stratégie employée (marginale, opposée ou compacte), les taux de classification obtenus avec la mesure cardinale (indiqués en gras) sont supérieurs à ceux obtenus avec la mesure locale en utilisant les fonctions d'appartenance Crisp, FCM et Gaussienne. En revanche, la mesure locale fournit des taux de classification identiques ou sensiblement meilleurs que ceux de la mesure cardinale lorsque la fonction triangulaire est utilisée. Ces résultats confortent la supériorité globale de la mesure cardinale sur la mesure locale.

- *FCAM vs. MFCLAM vs. OFCLAM :* Lorsqu'on compare les trois stratégies entre elles, nous constatons que, pour une même fonction d'appartenance et pour une même mesure locale ou cardinale, la stratégie opposée fournit les meilleurs taux de classification, suivie de la stratégie compacte, puis de celle marginale. Avec la mesure cardinale, le

meilleur taux de classification (92.21%) est obtenu par OFCLAM$_c$ avec la fonction FCM, le second meilleur taux (90.29%) est obtenu par la FCAM$_c$ avec la fonction Triangulaire, et le troisième taux (90.15%) est celui de MFCLAM$_c$ obtenu avec FCM. Avec la mesure locale, OFCLAM$_l$ (91.91%) est meilleure que FCAM$_l$ (90.88%) et MFCLAM$_l$ (89.26%) avec la fonction d'appartenance Triangulaire pour FCAM$_l$ et MFCLAM$_l$ et la fonction Gaussienne pour OFCLAM$_l$. Les meilleurs taux de classification obtenus par chaque méthode sont indiqués par une astérisque dans le tableau 3.2.

En ce qui concerne le temps de calcul, la stratégie compacte (FCAM) apparaît environ trois fois plus rapide que la stratégie marginale (MFCLAM) et six fois plus rapide que la stratégie opposée (OFCLAM) pour les deux phases de caractérisation et de classification. Ceci s'explique par le fait que la FCAM exige trois fois moins d'espace mémoire que les MFCLAMs et six fois moins que les OFCLAMs.

Finalement, la stratégie opposée (OFCLAM) reste la plus performante, mais nécessite un temps de calcul très élevé. La stratégie marginale (MFCLAM) est plus rapide que la stratégie opposée (OFCLAM), mais affiche un taux de classification plus réduit. À l'inverse, la stratégie compacte offre un bon compromis entre précision et rapidité (trois fois plus rapide que la MFCLAM et six fois plus rapide que la OFCLAM).

3.5.5 Robustesse des matrices aura des couleur floues

Pour évaluer encore plus la pertinence des attributs FCAMs, nous avons étudié leur robustesse vis-à-vis de la rotation, de la résolution, du bruit gaussien et du flou. Pour cela, nous avons réalisé des tests sur d'autres bases de textures, à savoir *Outex-TC-30*, *Outex-TC-31*, *Outex-TC-32* et *Outex-TC-33*. Ces quatre bases, disponibles sur le site `https://color.univ-lille.fr/` ont été générées par l'équipe Imagerie Couleur de Lille à partir de la base *Outex-TC-13* d'origine en lui faisant subir différents types de dégradations. Chacune d'elles contient 68 classes constituées chacune de 40 images couleur, à l'exception de *Outex-TC-30*, qui contient 160 images.

Afin de tester la robustesse des attributs FCAMs vis-à-vis des différentes dégradations, chaque base est divisée en un sous-ensemble d'apprentissage et un sous-ensemble de test.

Le sous-ensemble d'apprentissage, identique pour toutes les bases, contient $68 \times 20 = 1360$ images, acquises sous un éclairage à lampe incandescente et à la résolution de 100 dpi. Les images du sous-ensemble de test représentent les mêmes textures que les images d'apprentissage mais acquises dans différentes conditions :

- TC-30 contient 10880 images test ayant subi une rotation selon 8 valeurs d'angle différentes (5, 10, 15, 30, 45, 60, 75 et 90 degrés) ;
- TC-31 contient 1360 images test acquises avec une résolution de 120 dpi ;
- TC-32 contient 1360 images test dégradées par un bruit Gaussien additif d'écart-type $\sigma = 5$;
- TC-33 contient 1360 images test floutées par un filtre Gaussien d'écart-type $\rho = 0.5$.

Le tableau 3.3 présente les taux de classification obtenus sur les différents types de dégradations de textures : rotation (*TC-30*), résolution (*TC-31*), bruit gaussien additif (*TC-32*), et flou (*TC-33*). Chaque type de dégradation est analysé en fonction du nombre de couleurs $C = 8$ et pour différentes fonctions d'appartenance (Crisp, FCM, Gaussienne et Triangulaire).

- *Crisp vs. flou :* Les taux de classification obtenus avec les fonctions d'appartenance floues (FCM, Gaussienne et Triangulaire) par les trois stratégies, et plus particulièrement par les FCAMs, sur les bases *TC-30*, *TC-31*, *TC-32* et *TC-33*, sont plus élevés que ceux obtenus avec la fonction d'appartenance Crisp. Ces résultats montrent encore une fois l'intérêt du flou sur le Crisp, même en présence des dégradations (rotation, résolution, bruit Gaussien additif et flou).

- *Mesure cardinale vs. mesure locale :* Pour les bases *TC-30* et *TC-31*, la mesure cardinale tend à être légèrement plus performante que la mesure locale pour les différentes stratégies (compacte, marginale et opposée), ce qui indique que la mesure cardinale est globalement plus robuste face à la rotation et à la résolution des textures. En revanche, pour les dégradations dues au bruit gaussien (*TC-32*) et au flou (*TC-33*), la mesure locale (FCAM_l, MFCLAM_l et OFCLAM_l) semble bien plus robuste que la mesure cardinale (FCAM_c, MFCLAM_c et OFCLAM_c).

- *FCM vs. Gaussienne vs. Triangulaire :* La fonction Triangulaire est globalement la plus indiquée pour être utilisée avec la stratégie compacte pour les quatre dégradations. Alors que,

TAB. 3.3: Taux de classification (%) pour les différents cas de dégradations des textures de la base *Outex-TC-13*, à savoir TC-30, TC-31, TC-32 et TC-33. Les valeurs en gras indiquent les meilleurs taux de classification pour chaque combinaison d'attributs et de fonctions.

Stratégie	Attributs	Fonction d'appartenance	TC-13	TC-30	TC-31	TC-32	TC-33
Compacte	$FCAM_c$	Crisp	35.88	36.07	35.66	37.57	41.76
		FCM	83.38	82.72	84.34	65.37	86.69
		Gaussienne	88.68	87.84	92.21	**80.07**	95.88
		Triangulaire	**90.29**	**88.93**	**93.09**	78.24	**96.10**
	$FCAM_l$	Crisp	35.15	34.85	34.71	36.18	42.79
		FCM	80.44	79.31	81.69	71.25	88.16
		Gaussienne	87.06	86.09	90.07	90.96	99.41
		Triangulaire	**90.88**	**87.46**	**91.25**	**91.99**	**99.49**
Marginale	$MFCLAM_c$	Crisp	88.82	90.41	90.44	70.00	91.32
		FCM	**90.15**	**91.55**	92.21	77.35	94.85
		Gaussienne	89.85	91.43	**93.09**	**78.60**	**95.15**
		Triangulaire	89.26	91.39	91.91	77.06	92.35
	$MFCLAM_l$	Crisp	87.94	89.23	90.29	72.72	93.09
		FCM	**89.26**	**90.87**	92.35	82.87	**98.82**
		Gaussienne	88.97	90.81	**92.43**	**90.74**	97.65
		Triangulaire	**89.26**	82.70	91.18	83.46	97.13
Opposée	$OFCLAM_c$	Crisp	89.26	90.82	92.43	75.66	93.60
		FCM	**92.21**	**92.67**	94.63	78.82	**96.76**
		Gaussienne	91.91	92.56	**94.71**	**80.88**	96.32
		Triangulaire	90.74	92.26	94.34	78.60	94.12
	$OFCLAM_l$	Crisp	88.24	89.29	91.69	82.21	95.00
		FCM	91.62	**92.24**	94.49	83.75	**99.49**
		Gaussienne	**91.91**	92.08	**94.56**	**92.06**	99.04
		Triangulaire	90.88	91.34	93.60	84.41	97.35

avec les stratégies marginale et opposée, la fonction Gaussienne apparaît plus performante en cas de changement de la résolution et en présence de bruit gaussien additif. Par ailleurs, la fonction FCM donne de meilleurs résultats face à la rotation et au flou.

- *FCAM vs. MFCLAM vs. OFCLAM :* La stratégie compacte est robuste face au flou (*TC-33*) (96,10 % avec $FCAM_c$ et 99,49 % avec $FCAM_l$) et au bruit gaussien (*TC-32*) (80,07 % avec $FCAM_c$ et 91,99 % avec $FCAM_l$), affichant ainsi une meilleure performance que la stratégie marginale et une robustesse comparable à celle de la stratégie opposée. En revanche, pour la rotation (*TC-30*) et la résolution (*TC-31*), les stratégies marginale et opposée surpassent la stratégie compacte.

3.5.6 Comparaison de la FCAM$_c$ avec d'autres méthodes d'extraction d'attributs de texture couleur

Pour juger de l'efficacité de la classification des images de textures couleur basée sur les FCAMs, nous avons confronté ses résultats avec ceux obtenus à l'aide d'autres méthodes d'extraction d'attributs de texture couleur sur trois bases d'images : *Outex-TC-13*, *Outex-TC-31* et *KTH-TPS*. Ces méthodes comprennent :

- *12 méthodes de la stratégie marginale :* LBP (Local Binary Patterns) [28], CLBP (Completed Local Binary Patterns) [29], ELBP (Extended Local Binary patterns) [44], ILBP (Improved Local Binary patterns) [30], TS (Texture Spectrum) [31], Gabor RGB [33], Hist. RGB (Histogram RGB) [39], FLBP (Fuzzy Local Binary Patterns) [70], FCLCM (Fuzzy Color Level Co-occurrence Matrices) [6], FuzEnC$_{2D}$ (Fuzzy Entropy Color 2D) [81], et nos méthodes MFCLAM$_c$, MFCLAM$_l$.
- *07 méthodes de la stratégie opposée :* MCLBP (Multiple Channels Local Binary Patterns) [53], IOCLBP (Improved Opponent Color Local Binary Patterns) [2], OCLBP (Opponent Color Local Binary Patterns) [51], ICM (Integrative Co-occurrence Matrices) [5], OGabor [33], et nos méthodes OFCLAM$_c$ et OFCLAM$_l$.
- *07 méthodes de la stratégie compacte :* MOCLBP (Mixed Order Color Local Binary Patterns) [57], LBPC (Local Binary Patterns Color) [142], SWOBP (Spatially Weighted Order Binary Patterns) [59], FCCM (Fuzzy Color Co-occurrence Matrices) [6], et nos méthodes : FCAM$_c$ et FCAM$_l$.
- *07 modèles de CNNs :* ResNet-50, VGG VeryDeep 19, VGG VeryDeep 16, VGG M, Caffe AlexNet [2], GoogleNet, et AlexNet [3].

La classification est réalisée en utilisant l'algorithme PPV avec trois différentes distances, à savoir la distance de Manhattan (L1), la distance χ^2, et la mesure de similarité (Sim). Les taux de classification obtenus par toutes ces méthodes sont regroupés dans les tableaux 3.4 et 3.5. Les résultats de ELBP, ILBP, TS, OCLBP et IOCLBP sont tirés de la référence [2]. Les résultats des CNNs (ResNet-50, VGG VeryDeep 16, VGG VeryDeep 19, VGG M, et Caffe AlexNet) sont prélevés de [2], et ceux de AlexNet et GoogleNet de [3]. Il est à souligner

TAB. 3.4: Taux de classification (%) de chaque méthode sur les trois bases de données de textures *Outex-TC-13*, *Outex-TC-31*, et *KTH-TIPS*. Les meilleurs résultats sont en gras et les seconds meilleurs en souligné

Méthodes	**St.**	**Dim.**	**TC-13**			**TC-31**			**KTH-TIPS**		
			L1	χ^2	Sim	L1	χ^2	Sim	L1	χ^2	Sim
Attributs de textures couleur											
LBP	**M**	1665	80.00	83.24	85.74	92.65	84.46	80.59	94.30	94.70	94.02
CLBP		4056	88.09	88.09	88.09	79.63	84.12	79.63	93.18	92.07	95.43
ELBP		321	83.3	-	-	-	-	-	93.8	-	-
ILBP		213	85.5	-	-	-	-	-	95.4	-	-
TS		2502	81.3	-	-	-	-	-	92,9	-	-
Gabor RGB		96	89.56	87.21	19.56	85.37	82.13	11.84	93.66	93.75	30.56
Hist. RGB		768	90.15	89.56	91.76	92.21	94.26	93.16	94.83	95.41	94.58
ICM	**O**	30	91.32	91.62	75.59	83.60	83.90	69.85	91.34	92.31	57.43
MCLBP		531	**91.62**	92.35	91.91	92.50	93.75	95.81	95.81	96.00	95.16
OGabor		264	89.85	87.21	33.38	91.47	91.54	21.84	94.81	94.98	67.51
IOCLBP		1287	91.00	-	-	-	-	-	96.20	-	-
OCLBP		648	90.90	-	-	-	-	-	95.20	-	-
MOCLBP	**C**	-	-	-	87,1	-	-	-	-	-	-
LBPC		256	76.18	77.79	76.18	81.62	84.41	81.62	91.88	92.50	91.88
SWOBP		2244	86.32	85.00	86.32	90.51	**94.71**	90.51	97.87	97.80	**97.87**
Attributs de textures couleur floues											
$FuzEnC_{2D}$	**M**	3	60.00	61.32	-	46.69	46.76	-	56.12	57.74	-
FLBP		768	87.50	88.68	88.53	83.97	84.19	84.85	93.03	93.20	92.66
FCLCM (C=8)		192	88.53	90.88	89.26	91.10	91.54	92.43	93.99	94.75	94.47
$MFCLAM_c$ (C=8)		192	90.29	91.18	90.15	91.40	92.21	93.09	94.16	94.34	94.71
$OFCLAM_l$ (C=8)	**O**	384	91.03	92.50	91.62	94.41	94.41	94.56	95.82	96.19	95.95
$OFCLAM_c$ (C=8)		384	**91.62**	**92.94**	92.21	93.82	93.68	94.71	95.76	95.76	95.77
FCCM	**C**	64	90.88	91.32	90.88	92.06	90.00	91.25	96.19	95.76	96.12
$FCAM_c$ (C=8)		64	90.59	90.59	90.29	92.13	89.93	93.09	96.30	95.95	95.92
$FCAM_c$ (C=64)		4096	91.32	91.76	**92.35**	**95.15**	94.56	**95.07**	**98.19**	**98. 45**	97.35

TAB. 3.5: Comparaison des taux de classification obtenus par $FCAM_c$ avec ceux obtenus par les méthodes basées sur les CNNs sur les bases d'images de textures *Outex-TC-13* et *KTH-TIPS*.

Méthode	ResNet-50	VGGVD 19	VGGVD 16	VGG-M	Caffe AlexNet	GoogleNet	AlexNet	$FCAM_c$ (C=8)	$FCAM_c$ (C=64)
TC-13	87.20	83.80	84.30	84.90	82.90	82.65	85.98	90.59	**91.32**
KTH-TIPS	**99.60**	99.40	99.40	98.70	98.70	-	-	96.30	98.19

que les CNNs ont été entrainés sur la base ImageNet et sont utilisés comme extracteurs d'attributs seulement. La classification est réalisée par la règle du PPV avec la distance L_1.

Les résultats du tableau 3.4 montrent que, pour les trois bases d'images, les FCAMs sont plus efficaces que les méthodes de l'état de l'art, puisqu'elles fournissent les meilleurs ou les seconds meilleurs résultats, et ce quelle que soit la distance utilisée. La $FCAM_c$ avec 64 couleurs se classe en première ou deuxième position 8 fois sur 9, la $FCAM_l$ 4 fois sur 9 et $OFCLAM_c$ (avec 8 couleurs) 3 fois sur 9. La $FCAM_c$ semble également plus performante que les méthodes compactes (MOCLBP, LBPC et SWOBP) et les méthodes floues (FLBP et FCCM) même avec $C = 8$ couleurs seulement. La $FCAM_c$ apparaît également, plus

performante que les CNNs sur la base *TC-13* et comparable avec les CNNs sur la base *KTH-TIPS*.

3.6 Conclusion

Dans ce chapitre, nous avons principalement proposé d'étendre les matrices aura aux images couleur selon la stratégie compacte, afin de caractériser complètement une texture couleur par une seule matrice aura couleur floue (FCAM). Pour mettre en évidence la capacité de cette matrice à caractériser les textures couleur, nous avons effectué une classification des textures couleur sur la base de ces dernières. Des expériences menées sur la base d'images de textures couleur *Outex-TC-13* ont montré que :

- La matrice aura des couleurs floues (FCAM) est, dans tous les cas, plus performante que la matrice aura des couleurs nettes (CAM), attestant ainsi de l'intérêt du flou par rapport au crisp.
- La FCAM cardinale (FCAM_c) est souvent plus efficace que la FCAM locale (FCAM_l).
- La FCAM obtenue par la stratégie compacte offre un bon compromis entre précision et rapidité, puisqu'elle fournit des résultats de classification proches de ceux obtenus avec la stratégie opposée (OFCLAM), et meilleurs que ceux de la stratégie marginale (MFCLAM), mais avec un temps de calcul et un espace mémoire bien plus réduits.

Des tests menés sur d'autres bases de *Outex-TC-13*, dégradées, ont confirmé d'une part la supériorité du flou sur le crisp en présence de dégradations de type rotation, résolution, bruit gaussien additif et flou, et la robustesse des FCAMs vis-à-vis surtout du bruit gaussien et du flou. D'autres tests effectués sur trois bases *TC-13*, *TC-31* et *KTH-TIPS* ont montré l'efficacité des FCAMs par rapport à d'autres méthodes d'extraction d'attributs de l'état de l'art et leur compétitivité avec les CNNs.

Chapitre 4

Segmentation d'images couleur texturées basée sur les FCAMs

4.1 Introduction

Nous avons développé, dans le chapitre précédent, une nouvelle méthode de caractérisation des textures couleur basée sur les matrices aura cardinales des couleurs floues ($FCAM_c$). Nous avons ensuite exploité ces $FCAM_c$ pour classer des textures couleur. Nous poursuivons dans ce chapitre avec une autre application des FCAMs cardinales et nous proposons une méthode de segmentation supervisée d'images couleur texturées, dans laquelle les FCAMs cardinales sont utilisées pour caractériser des superpixels obtenus par l'intermédiaire d'un algorithme SLIC régional que nous avons développé. Ces derniers sont classés par l'intermédiaire d'un classifieur.

Le reste de ce chapitre est scindé en deux parties. Dans la première partie, nous présenterons un aperçu général de la méthode de segmentation proposée, puis nous décrirons en détail chacune de ses étapes. Une attention particulière est accordée à l'algorithme SLIC régional proposé et à la caractérisation de chaque superpixel par une FCAM cardinale. La deuxième partie est consacrée à l'évaluation de l'algorithme régional SLIC proposé, au choix des paramètres de l'algorithme de segmentation développé, ainsi qu'à la validation de ses résultats en les comparant d'une part avec ceux obtenus par des méthodes basées sur le

flou, et d'autre part avec d'autres méthodes de segmentation supervisées d'images couleur texturées de l'état de l'art.

4.2 Méthode de segmentation d'images des couleur texturées proposée

La méthode de segmentation d'images couleur texturées proposée est une méthode de type supervisée qui s'appuie principalement sur les matrices aura cardinales des couleurs floues (FCAM$_c$) afin de caractériser localement les textures couleur, ainsi que sur la méthode de classification supervisée ELM (Extrema Learning Machine). Cette méthode, schématisée sur la figure 4.1, se déroule en deux phases successives : une phase d'apprentissage suivie d'une phase de segmentation.

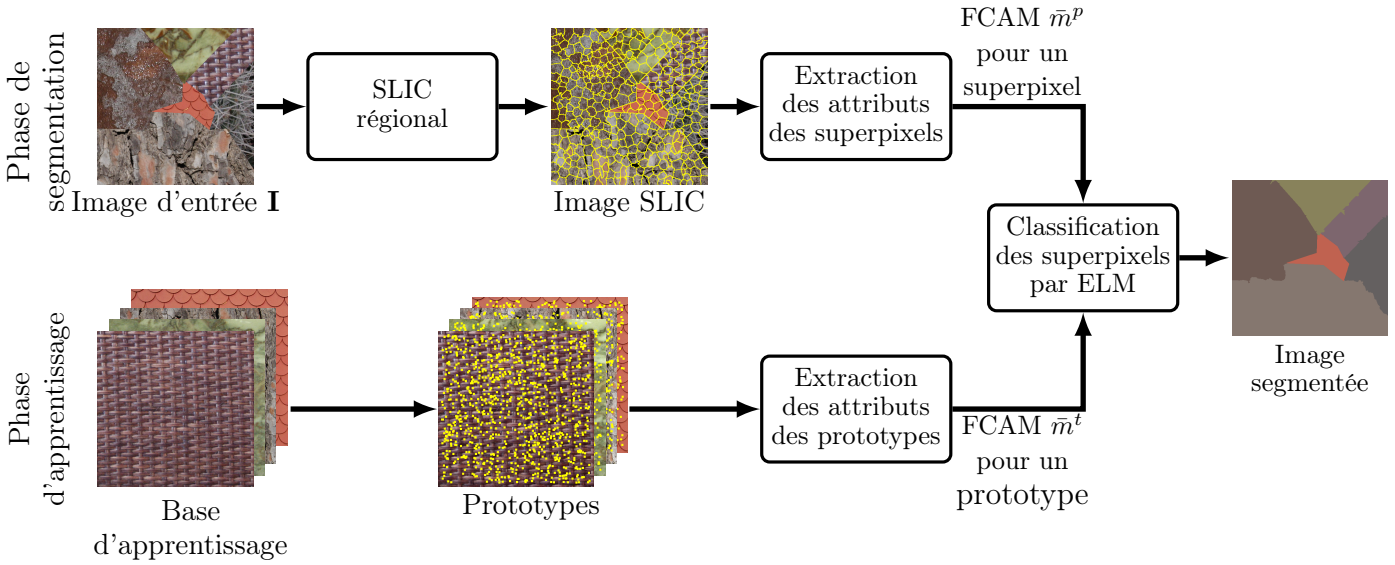
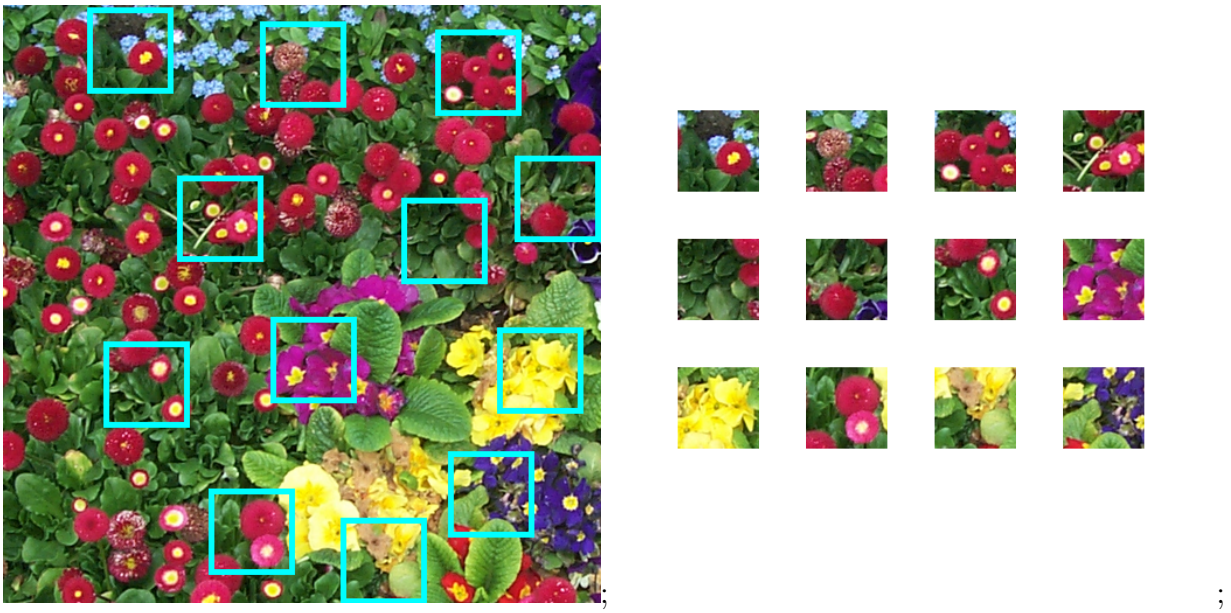


FIG. 4.1: Organigramme de la méthode de segmentation d'images couleur texturées proposée.

4.2.1 Phase d'apprentissage

La phase d'apprentissage se décompose en trois étapes. Il s'agit dans la première étape de générer un ensemble de prototypes à partir d'une base d'apprentissage. La caractérisation



(a) (b)

FIG. 4.2: Génération des pixels prototypes : (a) Image d'apprentissage. (b) Patchs de taille $(2\omega + 1)^2$.

de chaque prototype par des attributs de texture couleur FCAMs cardinales constitue la deuxième étape. La troisième étape concerne la classification.

4.2.1.1 Génération des pixels prototypes

Notre méthode de segmentation s'inscrit dans le cadre de la classification supervisée. Cela suppose que nous disposons d'une base d'apprentissage qui, dans notre cas, est composée d'un ensemble d'images où chaque image contient une seule texture couleur. La classe ou le type de texture de chaque image est parfaitement identifié. Chaque classe ou texture couleur est représentée par une seule ou par quelques images seulement, qui sont généralement de taille très élevée. Ces textures sont représentatives des textures qui composent l'image $\mathbf{I}$ à segmenter. Leur nombre est égal au nombre $K_{\mathbf{I}}$ de classes des textures présentes dans l'image à segmenter $\mathbf{I}$. Pour entraîner le classifieur, nous proposons de générer de manière aléatoire un ensemble de T prototypes par classe à partir des images de la base d'apprentissage. Les prototypes de chaque classe représentent les centres de patchs $\mathbf{W}^t$ $(t = 1, \ldots, T)$ de taille $(2\omega + 1)^2$ (Fig. 4.2).

4.2.1.2 Extraction des attributs de texture

Il s'agit dans cette étape de caractériser chaque prototype (patch) par un ensemble d'attributs, en l'occurrence par les éléments de la matrice aura cardinale des couleurs floues. Une $FCAM_c$ de taille $C \times C$ est construite selon l'équation (3.18) pour chaque prototype t à partir de tous les pixels présents dans le patch $\mathbf{W}^t$ de taille $(2\omega+1)^2$, centré sur le prototype en question.

Le résultat de cette procédure, est un ensemble X de $N = K_{\mathbf{I}} \cdot T$ vecteurs d'attributs ($\mathbf{X} = [\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_N]$) où chaque vecteur $\mathbf{x}_j = [x_{j1},x_{j2},...,x_{jC^2}]$ englobe les C^2 éléments de la FCAM. Cet ensemble X, auquel on associe un vecteur $\mathbf{Y} = [\mathbf{y}_1,\mathbf{y}_2,...,\mathbf{y}_j,...,\mathbf{y}_N]$ indiquant la classe de texture de chaque prototype, servira de base d'entrainement du classifieur. Chaque vecteur $\mathbf{y}_j$ ($j = 1,...,N$) est un vecteur binaire de dimension $K_{\mathbf{I}}$ dont un seul élément est non nul, tel que $\mathbf{y}_j = [y_{j1},...,y_{jl},...,y_{jK_{\mathbf{I}}}]$ avec :

$$y_{jk} = \begin{cases} 1 & \text{si } j \in \text{Classe } l, \\ 0 & \text{sinon.} \end{cases} \tag{4.1}$$

4.2.1.3 Entrainement du classifieur

Plusieurs méthodes de classification supervisée, telles que KNN, SVM, MLP et ELM peuvent être utilisées pour classer l'ensemble d'apprentissage (X,Y). Nous avons opté dans notre travail de segmentation pour l'ELM en raison de sa simplicité, de son apprentissage rapide et de la qualité de ses résultats. Ce choix sera par ailleurs justifié dans la section 4.3.6.

L'ELM est un réseau de neurones organisé en trois couches entièrement connectées (Fig. 4.3) : une couche d'entrée de C^2 neurones qui reçoit les éléments de la FCAM_c, une couche cachée, et une couche de sortie avec $K_{\mathbf{I}}$ neurones. Le nombre de neurones dans la couche cachée est choisi empiriquement à $L = 100 \cdot K_{\mathbf{I}}$.

On définit par $\mathbf{w}_i = [w_{i1},w_{i2},\ldots,w_{iC^2}]^T$, le vecteur de poids d'entrée qui relie le i-ème neurone caché à tous les neurones d'entrée, et par $\beta_i = [\beta_{i1},\beta_{i2},\ldots,\beta_{iL}]^T$ le vecteur de poids reliant

le i-ème neurone caché a tous les neurones de sortie. On désigne également par b_i le biais du i-ème neurone de la couche cachée.

La sortie h_{ij} du i-ème neurone caché lorsqu'il reçoit l'entrée $\mathbf{x}_j$, $\mathbf{x}_j = [x_{j1},x_{j2},...,x_{jC^2}]$ est définie par :

$$\mathbf{h}_{ij} = g(\mathbf{w}_i\mathbf{x}_j + b_i) \tag{4.2}$$

où g est une fonction d'activation dont la plus utilisée est la fonction sigmoïde :

$$g(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{x}}} \tag{4.3}$$

La réponse de l'ELM au vecteur d'entrée $\mathbf{x}_j$ est fournie par la couche de sortie sous forme d'un vecteur de dimension $K_\mathbf{I}$:

$$\mathbf{S}_j = \sum_{i=1}^{L} \beta_i \mathbf{h}_{ij} \tag{4.4}$$

tel que $\mathbf{S}_j = [s_{j1},s_{j2},...,s_{jK_\mathbf{I}}]$.

Lors de l'apprentissage, les poids initiaux w_i et les biais b_i des neurones cachés sont fixés à des valeurs aléatoires, et les poids de sortie β_i sont déterminés de sorte que les réponses de l'ELM à tout l'ensemble d'apprentissage X soient identiques à leurs labels, c'est-à-dire :

$$\sum_{i=1}^{L} \beta_i \mathbf{h}_{ij} = \mathbf{y}_j \qquad\qquad \forall j = 1,...,N \tag{4.5}$$

L'équation 4.4 peut être réécrite sous forme matricielle comme suit :

$$\mathbf{H}\beta = \mathbf{Y} \tag{4.6}$$

avec

$$\mathbf{H}(\mathbf{w}_1,\mathbf{w}_2,\ldots,\mathbf{w}_L,\mathbf{b}_1,\mathbf{b}_2,\ldots,\mathbf{b}_L,\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N) = \begin{bmatrix} g(w_1\cdot x_1 + b_1) & \ldots & g(w_L\cdot x_1 + b_L) \\ \vdots & \ddots & \vdots \\ g(w_1\cdot x_N + b_1) & \ldots & g(w_L\cdot x_N + b_L) \end{bmatrix},$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix} \text{ et } \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix}.$$

L'entraînement de l'ELM revient simplement à trouver la solution $\hat{\beta}$ de l'équation (4.6). Dans le cas où la matrice $\mathbf{H}$ est carée ($N = L$), la solution est donnée par $\hat{\beta} = \mathbf{H}^{-1}\mathbf{Y}$. Or le nombre de neurones L de la couche cachée est souvent inférieur au nombre N d'échantillons d'entrainement, par conséquent $\mathbf{H}$ n'est pas carrée donc non inversible. Dans ce cas la solution de l'équation (4.6) peut être obtenue en minimisant l'erreur quadratique par la méthode des moindres carrées $\min_\beta \|\mathbf{H}\beta - \mathbf{Y}\|$. La solution de cette équation est donnée par la relation suivante:

$$\hat{\beta} = \mathbf{H}^{+}\mathbf{Y} \tag{4.7}$$

où $\mathbf{H}^{+}$ est l'inverse généralisée Moore-Penrose de la matrice $\mathbf{H}$.

Il est à noter que l'apprentissage de l'ELM ne nécessite aucune règle de rétro-propagation, ni ajustement itérative des poids [143].

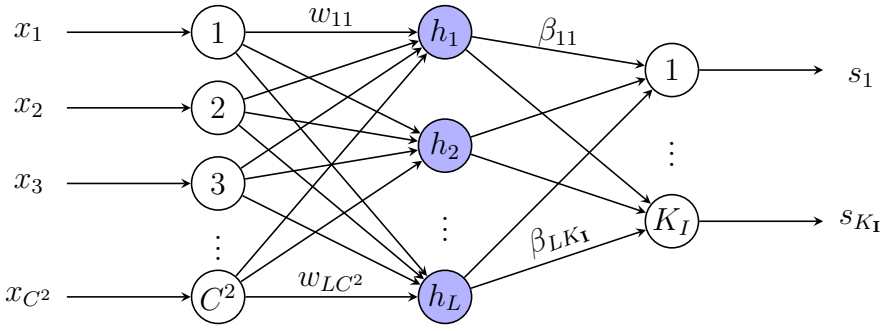


FIG. 4.3: Architecture du réseau de neurones ELM

4.2.2 Phase de segmentation

La phase de segmentation proprement dite consiste à segmenter une image test $\mathbf{I}$. Son objectif est d'affecter chaque pixel de l'image $\mathbf{I}$ à l'une des $K_\mathbf{I}$ classes de texture présentes dans l'image ($K_\mathbf{I}$ étant connu apriori). En principe, chaque pixel de l'image $\mathbf{I}$ doit être caractérisé

par les attributs de la FCAM cardinale, puis affecté à l'une des $K_{\mathbf{I}}$ classes à l'aide de l'ELM. Cependant, cette façon de procéder peut engendrer des temps de calcul très élevés. Pour accélérer les calculs, nous proposons de réduire le nombre de pixels à traiter en pré-segmentant l'image $\mathbf{I}$ en un nombre P de superpixels. Ces superpixels seront caractérisés par les attributs de texture (FCAM_c) puis affectés à l'une des $K_{\mathbf{I}}$ classes à l'aide du réseau ELM déjà entraîné lors de la phase d'apprentissage.

4.2.2.1 Pré-segmentation

Plusieurs méthodes de classification non supervisée peuvent être employées pour pré-segmenter une image. Parmi ces méthodes, on peut citer l'algorithme normalized cut [144], la méthode basée sur les graphes [145], l'algorithme Mean shift [146], la méthode Quick shift [147], et la méthode Simple Linear Iterative Clustering (SLIC) [148]. Notre choix s'est porté sur le SLIC en raison de sa simplicité, de sa rapidité et de sa capacité à respecter les limites des régions présentes dans l'image.

Algorithme SLIC

L'algorithme SLIC proposée par Achanta et al. [148] est une version modifiée de l'algorithme des k-means qui permet de sur-segmenter une image en un nombre donné de régions appelées superpixels en regroupant les pixels selon une distance spatiale et de couleur.

Pour une image couleur $\mathbf{I}$ définie dans l'espace RGB sur une grille $\mathtt{S}$, chaque pixel $\mathbf{s} \in \mathtt{S}$ de coordonnées $(x_{\mathbf{s}},y_{\mathbf{s}})$ est caractérisé par ses trois composantes couleur $(I^R(\mathbf{s}),I^G(\mathbf{s}),I^B(\mathbf{s}))^\intercal$. Initialement, l'algorithme SLIC transforme ces composantes couleur (R,G,B) en composantes couleur La^*b^*. Chaque pixel $\mathbf{s}$ est ainsi représenté par un vecteur d'attributs de dimension cinq, incluant les trois composantes couleur (L,a^*,b^*) et spatiales $(x_{\mathbf{s}},y_{\mathbf{s}})$. L'algorithme SLIC divise ensuite la grille $\mathtt{S}$ en P superpixels $\{\mathtt{P}^p\}_{p=1}^P$, tels que $\bigcup_{p=1}^P \mathtt{P}^p = \mathtt{S}$ et $\mathtt{P}^p \cap \mathtt{P}^{p'} = \emptyset$ pour $p \neq p'$. Un superpixel $\mathtt{P}^p$ est défini comme un ensemble connexe de pixels connectés tels que deux pixels $\mathbf{s}$ et $\mathbf{s}'$ dans un superpixel $\mathtt{P}^p$ sont reliés par un chemin constitué de pixels de $\mathtt{P}^p$. Cette méthode est constituée de trois étapes principales : l'initialisation, le regroupement de pixels et le post-traitement.

Lors de l'étape d'initialisation, les pixels sont regroupés en superpixels formant une division régulière de l'image sous forme de grille de taille $S \times S$, où $S = \sqrt{|\mathtt{S}| / P}$, avec $|\mathtt{S}|$ le nombre total de pixels dans l'image et P le nombre souhaité de superpixels. D'autres configurations peuvent également être envisagées, comme des superpixels hexagonaux.

La deuxième étape de l'algorithme SLIC est une phase itérative de regroupement des pixels en superpixels suivant une stratégie de regroupement similaire à l'algorithme k-means. Chaque pixel de l'image est affecté au superpixel le plus proche en fonction d'une distance entre deux pixels $\mathbf{s}$ et $\mathbf{s}'$, notée D, qui prend en compte à la fois la distance colorimétrique d_c dans l'espace CIELAB et la distance spatiale d_s dans le plan xy. Une mise à jour des caractéristiques des superpixels est effectuée en calculant les moyennes des couleurs et de position des superpixels. Les étapes d'affectation des pixels et de mise à jour des centres des superpixels sont répétées jusqu'à ce que les centres des superpixels ne changent plus de manière significative.

La distance D entre deux pixels $\mathbf{s}$ et $\mathbf{s}'$ est donnée par la formule suivante :

$$D(\mathbf{s},\mathbf{s}') = \sqrt{d_c^2(\mathbf{s},\mathbf{s}') + m^2 \cdot d_s^2(\mathbf{s},\mathbf{s}')/S^2} \tag{4.8}$$

avec d_c la distance colorimétrique telle que:

$$d_c^2(\mathbf{s},\mathbf{s}') = (I^L(\mathbf{s}) - I^L(\mathbf{s}'))^2 + (I^a(\mathbf{s}) - I^a(\mathbf{s}'))^2 + (I^b(\mathbf{s}) - I^b(\mathbf{s}'))^2 \tag{4.9}$$

et d_s la distance spatiale définie par:

$$d_s^2(\mathbf{s},\mathbf{s}') = (x_\mathbf{s} - x_{\mathbf{s}'})^2 + (y_\mathbf{s} - y_{\mathbf{s}'})^2 \tag{4.10}$$

Le paramètre m contrôle la compacité des superpixels et est généralement fixé à 1. Cela signifie que la distance spatiale et la distance colorimétrique sont traitées de manière équivalente.

La figure 4.4(b) montre le résultat de SLIC sur une image couleur composée de six régions de textures différentes (Fig.4.4(a)).

La méthode de pré-segmentation utilisant SLIC de base [148] présente plusieurs avantages, notamment sa simplicité et son efficacité sur des images couleur non texturées. Cependant, SLIC regroupe les pixels uniquement en fonction de leur localisation et de leur couleur, ce qui engendre parfois des erreurs de sur-segmentation sur des images texturées (voir les flèches sur la figure 4.4(b)).

Pour remédier à ces inconvénients, plusieurs variantes de SLIC ont été proposées. Parmi elles, on peut citer la méthode SCALP (Superpixels with Contour Adherence using Linear Path) [149], Turbo Pixels [150], la méthode NNSC (Nearest Neighbor-based Superpixel Clustering), qui produit des superpixels très réguliers avec une grande adhérence aux contours et qui est robuste au bruit [151], TASP (Texture-Aware SuperPixel) [152] qui est une méthode capable de générer des superpixels en tenant compte de la texture.

Algorithme SLIC régional

Pour effectuer une pré-segmentation des images couleur texturées, nous avons proposé une nouvelle méthode, dénommée SLIC régional [153], qui est une version améliorée de SLIC. Cette méthode intègre l'information régionale en plus de l'information de couleur et spatiale. Elle applique initialement l'algorithme SLIC de base sur l'image à segmenter, puis calcule l'attribut régional $f_{\mathtt{R}}$ pour chaque superpixel $\mathtt{R}$ en utilisant la formule suivante [154] :

$$f_{\mathtt{R}} = [p_{\mathtt{R}} - p_{\mathtt{R}} \cdot \log(p_{\mathtt{R}})] + \frac{1}{|\mathscr{N}_{\mathtt{R}}|} \sum_{\mathtt{R}' \in \mathscr{N}_{\mathtt{R}}} [p_{\mathtt{R}'} - p_{\mathtt{R}'} \cdot \log(p_{\mathtt{R}'})], \tag{4.11}$$

où $p_{\mathtt{R}} = |\mathtt{R}| \,/\, |\mathtt{S}|$ est le rapport de la surface de chaque superpixel $\mathtt{R}$ par rapport à la taille de l'image et $\mathscr{N}_{\mathtt{R}}$ l'ensemble des superpixels adjacents à $\mathtt{R}$.

L'attribut régional $f_{\mathtt{R}}$ est composé de deux termes : le premier est proportionnel à la taille du superpixel $\mathtt{R}$ et augmente de manière monotone avec sa taille, car $p_{\mathtt{R}} \in]0{,}1[$. Le second terme prend en compte le contexte du superpixel en considérant l'influence des superpixels adjacents. Un petit superpixel entouré de petits superpixels aura une faible valeur de $f_{\mathtt{R}}$, tandis que si $\mathtt{R}$ et les superpixels adjacents sont grands, $f_{\mathtt{R}}$ aura une valeur élevée (voir Fig. 4.4(c)).

Par la suite, la valeur de la luminance $I^L(\mathbf{s})$ de chaque pixel $\mathbf{s}$ du superpixel $\mathtt{R}$ est remplacé par l'attribut régional $f_{\mathtt{R}}$ du superpixel. Finalement, l'algorithme SLIC est à nouveau appliqué sur l'image $\mathbf{I}$ en utilisant $(f_{\mathtt{R}},I^a(\mathbf{s}),I^b(\mathbf{s}),x_{\mathbf{s}},y_{\mathbf{s}})^\intercal$ comme vecteur d'attributs. Cette méthode est résumée par l'algorithme 1 :

Algorithme 1: SLIC Régional.

Entrée: Image RGB $\mathbf{I}$

1. Convertir $\mathbf{I}$ de l'espace RGB à l'espace de couleur La^*b^*;
2. Appliquer le SLIC à $\mathtt{S}$. Chaque pixel $\mathbf{s}$ est caractérisé par $(I^L(\mathbf{s})$, $I^a(\mathbf{s})$, $I^b(\mathbf{s}),x_{\mathbf{s}},y_{\mathbf{s}})^\intercal$;
3. Calculer l'attribut regional $f_{\mathtt{R}}$ de chaque superpixel $\mathtt{R}$ de la carte SLIC en utilisant l'équation (4.11);
4. Appliquer à nouveau le SLIC à $\mathtt{S}$. Chaque pixel $\mathbf{s}$ est caractérisé par $(f_{\mathtt{R}}$, $I^a(\mathbf{s})$, $I^b(\mathbf{s}),x_{\mathbf{s}},y_{\mathbf{s}})^\intercal$.

Sortie: Partition de $\mathbf{I}$ en superpixels $\{\mathtt{P}^p\}_{p=1}^P$

La figure 4.4(d) montre le résultat de l'application de la méthode SLIC régionale sur l'image de la figure 4.4(a). Il convient de noter que les zones de textures différentes sont mieux délimitées avec le SLIC régional (voir flèches), qu'avec sa version de base, bien que l'attribut régional décrit l'information de texture du superpixel dans l'image pré-segmentée et ne peut être considéré comme un attribut de texture proprement dit.

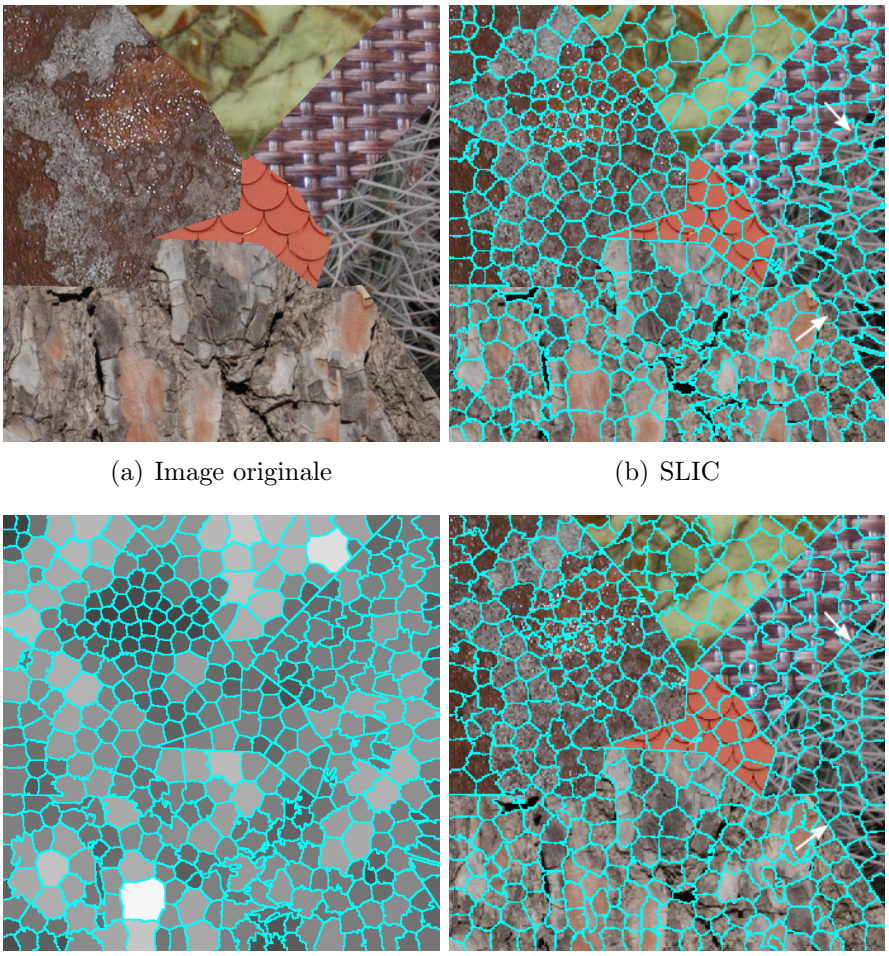
4.2.2.2 Calcul de la FCAM normalisée d'un superpixel

Dans cette étape, chaque superpixel $\mathtt{P}^p$ $(p = 1,2,...,P)$ est caractérisé par un ensemble d'attributs. Ces attributs correspondent aux éléments de la matrice aura cardinale des couleurs floues (FCAM$_c$). Pour un superpixel $\mathtt{P}^p$, la FCAM$_c$ est construite uniquement à partir des pixels contenus dans le superpixel $\mathtt{P}^p$.

Les mêmes formules déployées pour le calcul des FCAMs des images peuvent être employées pour le calcul des FCAMs d'un superpixel.

Pour un superpixel $\mathtt{P}^p$, l'ensemble aura $\mathtt{A}^p_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})$ de $\mathtt{S}_{\mathbf{c}}$ par rapport à $\mathtt{S}_{\mathbf{c}'}$ est alors défini comme suit :

$$\mathtt{A}^p_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}}) = \bigcup_{\mathbf{s}\in\mathtt{S}} (\mathtt{N}^p_{\mathbf{s},\mathtt{S}_{\mathbf{c}}} \cap S_{\mathbf{c}'}), \tag{4.12}$$



(a) Image originale (b) SLIC

(c) $f_{\mathtt{R}}$ map (d) SLIC régional

FIG. 4.4: Résultats de pré-segmentation SLIC ($P = 400$ superpixels). $f_{\mathtt{R}}$ dans (**c**) est remis à l'échelle de $[\![0{,}255]\!]$.

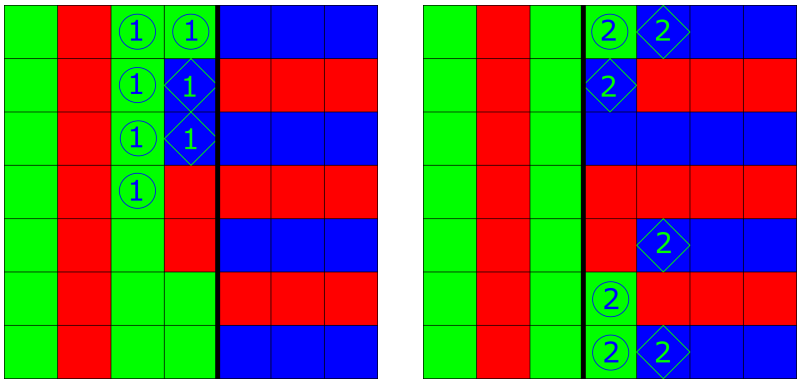
où $\mathtt{N}^{p}_{\mathbf{s},\mathtt{S}_\mathbf{c}}$ est l'ensemble des pixels voisins de chaque pixel $\mathbf{s} \in \mathtt{S}_\mathbf{c}$ situés dans le superpixel $\mathtt{P}^p$:

$$\mathtt{N}^{p}_{\mathbf{s},\mathtt{S}_\mathbf{c}} = \left\{\mathbf{r} \in \mathtt{S}, (\mathbf{s} \in \mathtt{S}_\mathbf{c} \cap \mathtt{P}^p) \wedge (\mathbf{r} \in \mathtt{N}_\mathbf{s} \cap \mathtt{P}^p)\right\}. \tag{4.13}$$

$\mathtt{N}_\mathbf{s}$ étant l'ensemble des pixels voisins du pixel $\mathbf{s}$.

La mesure cardinale de l'ensemble aura $\mathtt{A}^{p}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})$ relatif au superpixel $\mathtt{P}^p$ est :

$$m^p(c,c') = \left|\mathtt{A}^{p}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_\mathbf{c})\right|. \tag{4.14}$$



(a) (b)

FIG. 4.5: Exemples de deux ensembles aura dans une image couleur. (**a**) Deux superpixels $\mathtt{P}^1$ (**à gauche**) et $\mathtt{P}^2$ (**à droite**), et 2 ensembles aura $\mathtt{A}^1_{\mathtt{S_G}}(\mathtt{S_B})$ (①) et $\mathtt{A}^1_{\mathtt{S_B}}(\mathtt{S_G})$ (◇1◇). $\mathtt{A}^2_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^2_{\mathtt{S_B}}(\mathtt{S_G})$ sont vides. (**b**) Super-pixels $\mathtt{P}^1$ (**à gauche**) et $\mathtt{P}^2$ (**à droite**), et 2 ensembles aura $\mathtt{A}^2_{\mathtt{S_G}}(\mathtt{S_B})$ (②) et $\mathtt{A}^2_{\mathtt{S_B}}(\mathtt{S_G})$ (◇2◇). $\mathtt{A}^1_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^1_{\mathtt{S_B}}(\mathtt{S_G})$ sont vides.

Un exemple d'ensemble aura couleur d'un superpixel est présenté sur la figure 4.5 où une image couleur contenant deux textures, l'une représentée par des bandes verticales rouges et vertes, et l'autre formée de bandes horizontales rouges et bleues que l'on peut voir sur les trois colonnes de droite. Ces deux textures sont séparées par la quatrième colonne, qui est composée de pixels de couleurs rouge, vert et bleu.

Cette image est partitionnée en deux superpixels $\mathtt{P}^1$ et $\mathtt{P}^2$ délimités par une ligne verticale noire (Fig. 4.5(a)). $\mathtt{P}^1$ couvre les quatre colonnes de gauche et $\mathtt{P}^2$ les trois de droite. Les ensemble aura $\mathtt{A}^1_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^1_{\mathtt{S_B}}(\mathtt{S_G})$ ne sont pas vides, puisque $\mathtt{P}^1$ contient des pixels verts et bleus voisins, mais $\mathtt{A}^2_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^2_{\mathtt{S_B}}(\mathtt{S_G})$ sont vides car $\mathtt{P}^2$ ne contient pas de pixels vert.

La figure 4.5(b) montre une autre partition où $\mathtt{P}^1$ couvre les trois colonnes de gauche et $\mathtt{P}^2$ les quatre de droite. Dans ce cas, $\mathtt{A}^1_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^1_{\mathtt{S_B}}(\mathtt{S_G})$ sont vides car $\mathtt{P}^1$ ne contient pas de pixels bleus, mais $\mathtt{A}^2_{\mathtt{S_G}}(\mathtt{S_B})$ et $\mathtt{A}^2_{\mathtt{S_B}}(\mathtt{S_G})$ ne sont pas vides car $\mathtt{P}^2$ contient de pixels verts et bleus voisins.

Cet exemple montre que les ensembles aura couleur dépendent des frontières des superpixels. En effet, le long de la quatrième colonne qui sépare les deux textures, un seul pixel vert (sur la première ligne) et un pixel bleu (deuxième ligne) appartiennent aux ensembles aura de superpixels dans les deux Figures 4.5(a) et 4.5(b) .

Dans le cas flou, un ensemble de pixels de couleur floue $\mathtt{S}_{\mathbf{c}}$, $\mathbf{c} \in \mathscr{C}$ est caractérisé par sa fonction d'appartenance $\mu_{\mathtt{S}_{\mathbf{c}}}$, où le degré d'appartenance $\mu_{\mathtt{S}_{\mathbf{c}}}(\mathbf{s})$ de chaque pixel $\mathbf{s} \in \mathtt{S}$ à $\mathtt{S}_{\mathbf{c}}$ est déterminé par l'équation 3.12. L'ensemble aura des couleurs floues $\mathtt{A}^{p}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})$ de $\mathtt{S}_{\mathbf{c}}$ par rapport à $\mathtt{S}_{\mathbf{c}'}$ est un ensemble flou, constitué de pixels flous ayant chacun un degré d'appartenance $\mu_{\mathtt{S}_{\mathbf{c}}}(\mathbf{r})$ correspondant à chaque pixel $\mathbf{r} \in \mathtt{S}$ et est défini comme suit :

$$\mu_{\mathtt{A}^{p}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})}(\mathbf{r}) = \min\left\{\sup_{\mathbf{s}\in\mathtt{S}}\left[\min\left(\mu_{\mathtt{S}_{\mathbf{c}}}(\mathbf{s}),n^{p}(\mathbf{r},\mathbf{s})\right)\right],\mu_{\mathtt{S}_{\mathbf{c}'}}(\mathbf{r})\right\}. \tag{4.15}$$

La fonction de voisinage $n^{p}(\mathbf{r},\mathbf{s})$ exprime le degré d'appartenance du pixel $\mathbf{r}$ dans le voisinage du pixel $\mathbf{s}$ dans un superpixel $\mathtt{P}^{p}$, $p \in [\![1,P]\!]$. Le support de cette fonction peut avoir une taille et une forme quelconque, et peut prendre n'importe quelle valeur réelle comprise entre 0 et 1. Dans le cas simple, cette fonction de voisinage est nette (crisp) :

$$n^{p}(\mathbf{r},\mathbf{s}) = \begin{cases} 1 & \text{if } (\mathbf{s} \in \mathtt{P}^{p}) \wedge (\mathbf{r} \in \mathtt{N}_{\mathbf{s}} \cap \mathtt{P}^{p}), \\ 0 & \text{sinon.} \end{cases} \tag{4.16}$$

La mesure aura cardinale de l'ensemble aura $\mathtt{S}_{\mathbf{c}}$ par rapport à l'ensemble $\mathtt{S}_{\mathbf{c}'}$ pour un superpixel $\mathtt{P}^{p}$ est définie de la même manière que dans le cas d'une image :

$$\tilde{m}^{p}(\mathbf{c},\mathbf{c}') = \sum_{\mathbf{r}\in\mathtt{P}^{p}} \mu_{\mathtt{A}^{p}_{\mathtt{S}_{\mathbf{c}'}}(\mathtt{S}_{\mathbf{c}})}(\mathbf{r}) = \sum_{\mathbf{r}\in\mathtt{P}^{p}} \min\left\{\sup_{\mathbf{s}\in\mathtt{P}^{p}}\left[\min\left(\mu_{S_{\mathbf{c}}}(\mathbf{s}),n^{p}(\mathbf{r},\mathbf{s})\right)\right],\mu_{S_{\mathbf{c}'}}(\mathbf{r})\right\}. \tag{4.17}$$

La matrice aura cardinale des couleurs floues (FCAM$_c$) est une matrice carrée de taille $C \times C$, où C est le nombre de couleurs pris dans l'ensemble de couleurs $\mathscr{C}$. Elle regroupe toutes les mesures aura cardinales floues entre deux ensembles aura des couleurs floues.

Les superpixels ont généralement des tailles différentes. Nous avons alors normalisé leurs FCAMs afin qu'elles soient comparables :

$$\bar{M}^{p} = \frac{[\tilde{m}^{p}(\mathbf{c},\mathbf{c}')]}{\sum_{(\mathbf{y},\mathbf{y}')\in\mathscr{C}^{2}} \tilde{m}^{p}(\mathbf{y},\mathbf{y}')}. \tag{4.18}$$

Au final, chaque superpixel $\mathtt{P}^p$ de l'image à segmenter $\mathbf{I}$ est caractérisé par C^2 attributs correspondant aux éléments de la FCAM normalisée $\bar{M}^p$.

4.2.2.3 Classement des superpixels

Le but de cette étape est d'affecter chaque superpixel $\mathtt{P}^p$ à l'une des $K_\mathbf{I}$ classes de l'image $\mathbf{I}$ à l'aide du réseau ELM, entrainé lors de la phase d'apprentissage. Chaque superpixel $\mathtt{P}^p$ est représenté par un vecteur d'attributs $\mathbf{x}^p$, obtenu à partir de la FCAM$_c$ normalisée $\bar{M}^p$. Ce vecteur $\mathbf{x}^p$ est alors founi à l'entrée du ELM.

Pour chaque vecteur d'attributs $\mathbf{x}^p$, nous calculons les activations des neurones cachés $\mathbf{h}_i^p$, $i =$ 1,...,L, en utilisant les poids d'entrée $\mathbf{w}_i$, les biais b_i, et la fonction d'activation g dont les valeurs ont été fixées lors de l'entrainement de l'ELM :

$$\mathbf{h}_i^p = g(\mathbf{w}_i \cdot \mathbf{x}^p + b_i) \tag{4.19}$$

Le vecteur de sortie $\hat{\mathbf{y}}_p$ de dimension $K_\mathbf{I}$ pour le superpixel $\mathtt{P}^p$ est récupéré sur la couche de sortie de l'ELM :

$$\hat{\mathbf{y}}_p = \sum_{i=1}^{L} \hat{\beta}_i \cdot \mathbf{h}_i^p \tag{4.20}$$

$\hat{\beta}_i$ étant les poids de sortie appris durant l'entraînement du modèle ELM. Le superpixel $\mathtt{P}^p$ est finalement affecté à la classe k^* telle que:

$$l^* = \arg\max_{l=1,\ldots,K_\mathbf{I}}(\mathbf{y}_{pk}) \tag{4.21}$$

4.2.3 Étape de raffinement

L'image segmentée obtenue après classification des superpixels peut contenir des petites régions non significatives. Afin d'améliorer sa qualité, nous proposons d'appliquer une méthode de post-traitement. Cette méthode consiste à fusionner les petites régions dont la surface est inférieure à un seuil $\tau\%$ donné avec la région voisine la plus grande, puis étiqueter la

région résultante avec son indice de classe. La procédure itérative de fusion des petites régions continue tant qu'il existe une région dont la surface est inférieure au seuil $\tau\%$. Dans nos expériences, la valeur du seuil τ est fixée à 0.5.

La méthode de segmentation proposée est résumée par l'algorithme 2 suivant :

Algorithme 2: Segmentation d'images de texture en couleur.

Entrée : Image test $\mathbf{I}$, $K_{\mathbf{I}}$ images d'entraînement
Paramètres : Nombre T de prototypes par classe, Nombre P de superpixels, Nombre C de couleurs floues, Largeur du patch ω, Fonction d'appartenance $\mu_{\tilde{c}}$.
Étape 1 : Phase d'entraînement

1. Dans chaque image d'entraînement, sélectionner aléatoirement T pixels prototypes.
2. À chaque pixel prototype, calculer sa FCAM normalisée $\bar{m}^t$ sur le patch carré $\mathtt{W}^t$ correspondant de taille $(2\omega+1)^2$ en utilisant l'équation (3.17).
3. Entraîner le classificateur ELM avec les $K_{\mathbf{I}} \cdot T$ FCAM normalisées.

Étape 2 : Phase de segmentation

1. Exécuter SLIC régional (Algorithme 1) sur $\mathbf{I}$ pour obtenir P superpixels $\{\mathtt{P}^p\}_{p=1}^{P}$.
2. Calculer le FCAM normalisé $\bar{m}^p$ de chaque superpixel $\mathtt{P}^p$ en utilisant l'équation (4.18).
3. Soumettre $\bar{m}^p$ à l'ELM entraîné et assigner chaque pixel de $\mathtt{P}^p$ à la classe de sortie de l'ELM.

Étape 3 : Raffinement (optionnel)
Sortie : Image segmentée

4.3 Tests et résultats

Nous avons effectué une série de tests afin d'évaluer notre méthode de segmentation d'images de textures couleur basée sur les FCAMs. Dans cette section, nous procédons d'abord à l'évaluation de l'algorithme SLIC régional et au choix des paramètres de la méthode de segmentation proposée. Puis, nous comparons les résultats obtenus par notre méthode avec ceux obtenus par d'autres attributs flous, dans le but de démontrer la pertinence des attributs FCAMs. Au final, nous comparons les résultats obtenus par notre méthode avec ceux obtenus par plusieurs méthodes de segmentation de texture couleur de l'état de l'art.

Pour effectuer les premières évaluations, nous avons utilisé la base d'images couleur texturées de Prague.

4.3.1 Base d'images Prague

La base Prague Texture Segmentation Data Generator and Benchmark, plus communément appelée la base Prague proposée par Haindl et Mikes [155] est constituée de 114 images qui sont regroupées en 10 catégories de textures naturelles et artificielles, telles que le bois, le marbre, le métal, le papier, la pierre, le sable et le tissu, ainsi que des textures artificielles générées par ordinateur.

Dans cette expérience, nous avons utilisé la base de Prague normale, destinée aux méthodes supervisées, qui contient 20 mosaïques de textures couleur à segmenter (images test d'entrée). Chacune de ces images test $\mathbf{I}$ représente $K_{\mathbf{I}}$ régions (classes), ce nombre variant entre 3 et 12 selon les images. Chaque région contient une seule texture couleur. De plus une image d'apprentissage est fournie pour chaque classe de texture, c'est-à-dire que chaque image test $\mathbf{I}$ à segmenter est accompagnée de $K_{\mathbf{I}}$ images d'apprentissage contenant l'une des textures couleur relatives aux $K_{\mathbf{I}}$ régions de $\mathbf{I}$. Ces images test ont été générées de manière synthétique à l'aide d'un algorithme basé sur un générateur aléatoire polygones de Voronoi à partir du même nombre d'images originales de texture couleur. Trois exemples d'images de cette base sont présentées sur la figure 4.6 (première ligne), accompagnées de leurs images de vérité terrain. La figure 4.6 (de la deuxième à la quatrième ligne) montre quelques images de la base d'apprentissage avec lesquelles les images mosaïques ont été constituées. Toutes les images de la base d'images Prague ont une taille de 512×512 pixels.

4.3.2 Évaluation de l'algorithme SLIC régional proposé

Pour démontrer l'efficacité de l'algorithme de SLIC régional proposé, nous avons comparé ses résultats à ceux obtenus par l'algorithme SLIC de base proposé par Achanta et al [148] et ses variantes SCALP, turbo pixels, NNSC et TASP en s'appuyant sur quatre métriques couramment utilisées dans la littérature pour évaluer et comparer les algorithmes de génération de superpixels. Il s'agit de la mesure ASA (achievable segmentation accuracy) qui évalue la superposition des superpixels avec la vérité terrain [148, 156], le BR (boundary recall) qui mesure la qualité des frontières de segmentation en évaluant l'adhérence des frontières de segmentation par rapport aux frontières de vérité terrain [157], l'UE (under-segmentation

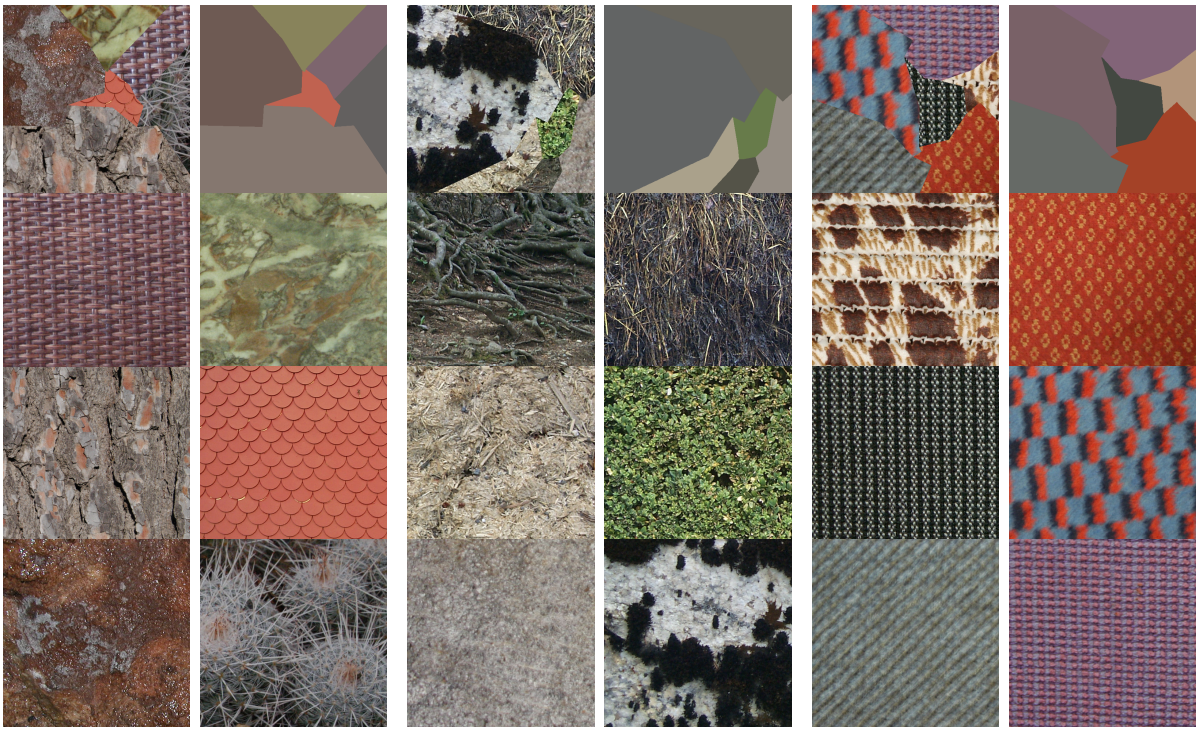


FIG. 4.6: Exemples d'images couleur texturées, représentant $K_{\mathbf{I}} = 6$ classes, accompagnées de leurs images de vérité terrain **(première ligne)**, ainsi que les images de la base d'apprentissage avec lesquelles les images mosaïques de la première ligne ont été constituées (**de la deuxième à la quatrième ligne**).

error) qui évalue l'exactitude de la frontière de segmentation en quantifiant le chevauchement entre les superpixels et les régions de vérité terrain [148, 156], et enfin, le COM (compactness) qui mesure la compacité des superpixels [157]. Des valeurs élevées pour BR, ASA et COM et des valeurs plus faibles pour UE indiquent une meilleure pré-segmentation. Plus de détails sur ces métriques sont présentés dans l'annexe B.

Tous les algorithmes SLIC de base, SCALP, Turbo Pixels, NNSC, TASP et SLIC régional exigent la définition d'un nombre de superpixels, qui doit être suffisamment petit pour obtenir des superpixels suffisamment grands pour contenir des motifs complets de textures, mais suffisamment petits pour s'adapter finement aux frontières entre les régions. Nous avons empiriquement déterminé que $P = 400$ était un bon compromis pour la base d'images Prague. Le tableau 4.1 montre la moyenne et l'écart-type des quatre métriques obtenues par les six algorithmes SLIC sur les 20 images de la base Prague. À partir de ce tableau, nous constatons que les résultats obtenus par l'algorithme SLIC régional sont meilleurs que ceux obtenus par d'autres algorithmes, particulièrement en termes de rappel des frontières

TAB. 4.1: Performances de SLIC de base et régional sur la base Prague. Les flèches indiquent la direction de la métrique (résultats d'autant meilleurs qu'elle est élevée ↑ ou faible ↓), et les meilleurs résultats sont en gras.

Metric	↑BR	↓UE	↑ASA	↑COM
SLIC de base	0.46 ± 0.15	0.36 ± 0.08	0.79 ± 0.05	0.56 ± 0.05
SCALP	0.49 ± 0.13	$\mathbf{0.12 \pm 0.07}$	$\mathbf{0.94 \pm 0.03}$	0.31 ± 0.06
Turbo Pixels	0.06 ± 0.03	0.28 ± 0.07	0.86 ± 0.04	$\mathbf{0.71 \pm 0.02}$
NNSC	0.32 ± 0.13	0.43 ± 0.09	0.73 ± 0.08	-
TASP	0.32 ± 0.12	0.44 ± 0.10	0.73 ± 0.08	0.051 ± 0.05
SLIC Régional	$\mathbf{0.52 \pm 0.13}$	0.32 ± 0.08	0.82 ± 0.05	0.63 ± 0.04

(BR) et de compacité (COM). Notre SLIC régional apparait également plus performant que TASP, alors que ce dernier est principalement conçu pour les images texturées.

4.3.3 Choix des paramètres la méthode de segmentation

Pour effectuer la segmentation de texture couleur proposée, plusieurs paramètres doivent être définis. Certains paramètres comme le nombre de prototypes T et la taille du patch sont propres à la phase d'apprentissage et les autres sont liés à la FCAM. Le nombre de prototypes par classe est fixé à $T = 1000$, ce qui correspond à la génération de 1000 images d'apprentissage pour chaque classe de l'image test $\mathbf{I}$ à segmenter (voir section 4.2.1.1). Cette démarche est conforme à celle utilisée par les méthodes basées sur les réseaux de neurones convolutionnels (CNN) [118] où 1000 images d'apprentissage sont générées pour chaque image à segmenter. La taille $(2\omega + 1)^2$ du patch centré sur chaque pixel prototype est fixée en fonction de la taille $N_{\mathbf{I}}$ de l'image test $\mathbf{I}$ et du nombre P de superpixels, de manière à avoir $(2\omega + 1)^2 \approx N_{\mathbf{I}}/P$. Dans notre cas, $N_{\mathbf{I}} = 512 \times 512$ et $P = 400$, la taille du patch est donc égale à 27×27 pixels.

Pour caractériser un pixel par une FCAM, nous devons définir à la fois le nombre de couleurs floues C et la fonction d'appartenance $\mu_{\tilde{\mathbf{c}}}$ qui permet de déterminer le degré d'appartenance $\mu_{\tilde{\mathbf{c}}}(\mathbf{s})$ de chaque pixel $\mathbf{s}$ à une couleur floue $\tilde{\mathbf{c}}$, en fonction de sa couleur $\mathbf{I}(\mathbf{s})$.

Comme pour la classification des textures couleurs (chapitre 3), nous avons à choisir parmi l'un des sept ensembles de couleurs. Plus précisément, nous avons limité le nombre de couleurs floues à l'une des valeurs suivantes : $C = 2 \times 2 \times 1$, $C = 2 \times 2 \times 2$, $C = 2 \times 3 \times 2$,

$C = 2 \times 4 \times 2$, $C = 3 \times 3 \times 2$, $C = 3 \times 4 \times 2$, ou $C = 4 \times 4 \times 2$ afin de réduire la taille des FCAMs à respectivement 4×4, 8×8, 12×12, 16×16, 18×18, 24×24, ou 32×32. Concernant la fonction d'appartenance, le choix doit se faire sur l'une des quatre fonctions d'appartenance Crisp, Gaussienne, Triangulaire ou FCM.

Pour effectuer ce choix, nous avons déterminé le taux de classification en fonction des sept ensembles de couleurs floues et des quatre fonctions d'appartenance. Pour rendre ce choix indépendant vis-à-vis de l'algorithme de classification, nous avons utilisé le classifieur du plus proche voisin (PPV) au lieu de ELM, et sans étape de raffinement.

La figure 4.7 montre le taux de classification moyen obtenu avec chaque fonction d'appartenance en fonction du nombre C de couleurs floues sur les 20 images de la base Prague. Il apparaît clairement sur cette figure que les fonctions d'appartenance floues (Gaussienne, Triangulaire et FCM) surpassent largement la fonction Crisp, en particulier lorsque le nombre de couleurs floues est faible. Ceci démontre encore une fois l'importance de la prise en compte du flou. Cependant, lorsque $C \geq 12$, les trois fonctions d'appartenance floues conduisent à des taux de classification presque similaires, même si la fonction d'appartenance Gaussienne semble légèrement meilleure. De plus, les taux de classification ne varient pas de manière significative au-delà de $C = 16$. Par conséquent, toutes les expériences suivantes seront effectuées avec $C = 16$ couleurs floues et la fonction d'appartenance Gaussienne.

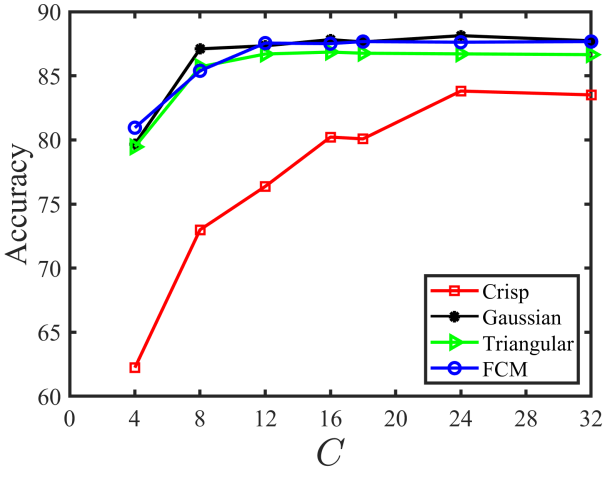


FIG. 4.7: Taux de classification moyen (%) en fonction du nombre de couleurs (C) pour différentes fonctions d'appartenance.

4.3.4 Comparaison de la FCAM cardinale avec d'autre méthodes d'extraction d'attributs

Nous comparons dans cette section la pertinence de la FCAM cardinale par rapport à d'autres attributs de texture floue, à savoir les matrices aura locales des niveaux des composantes couleur flous marginales (MFCLAM$_l$), la matrice de co-occurrence de couleurs floues (FCCM) [6], et les matrices aura cardinales des niveaux des composantes couleur flous marginales (MFCLAM$_c$) [8].

Il est utile de rappeler que les MFCLAM$_l$, obtenues en utilisant la mesure aura locale des niveaux des composantes flous décrite dans l'équation (2.61), sont similaires aux matrices de co-occurrences des niveaux des composantes couleurs [8]. D'autre part, la matrice FCCM est semblable à la matrice aura de couleur floue (FCAM$_l$) qui est obtenue en utilisant la mesure aura locale des couleurs floues définie dans l'équation (3.14) avec une fonction de voisinage binaire.

Les attributs de texture MFCLAM$_l$ et MFCLAM$_c$ sont formées en concaténant les attributs marginaux des trois composantes de couleur, à savoir le rouge (R), le vert (G) et le bleu (B). Pour une comparaison équitable, toutes ces matrices ont été calculées en utilisant les mêmes paramètres que la FCAM cardinale, notamment la fonction d'appartenance Gaussienne, $C^R = C^G = C^B = 16$ pour MFCLAM$_l$ et MFCLAM$_c$, ainsi que $C = 16$ pour FCCM.

La comparaison est basée sur le critère principal adopté dans de nombreuses études similaires, à savoir le taux de classification, ainsi que sur le temps de calcul requis par chaque méthode d'extraction d'attributs. Les tests ont été menés en utilisant le classificateur PPV au lieu de l'ELM, sur un ordinateur équipé d'un processeur Intel Core i7 à 3,60 GHz et de 8 Go de RAM.

Les résultats obtenus sur les 20 images de la base Prague sont résumés dans le tableau 4.2. Ils montrent clairement que les attributs FCAM$_c$ sont plus pertinents, trois fois plus rapides à calculer et nécessitent moins de mémoire que les attributs marginaux (MFCLAM$_l$ et MFCLAM$_c$). FCAM$_c$ apparaît également plus efficace et légèrement plus rapide à calculer que FCCM.

TAB. 4.2: Accuracy moyenne (%) et temps de calcul (s) pour différents attributs flous sur la base Prague. « nr » désigne un résultat sans raffinement de segmentation et « wr » un résultat avec raffinement de segmentation.

Attributs	**Taille**	**Accuracy nr**	**Accuracy wr**	**Temps de calcul (s)**
MFCLAM_l	$3 \times 16^2 = 768$	86.69	90.97	123.53
$\text{FCAM}_l \equiv \text{FCCM}$	$16^2 = 256$	86.52	90.77	37.62
MFCLAM_c	$3 \times 16^2 = 768$	87.24	91.28	112.32
FCAM_c	$16^2 = 256$	**87.82**	**92.00**	**33.12**

4.3.5 Temps de calcul

Dans la section 4.3.4, nous avons estimé uniquement le temps de calcul des FCAMs cardinales pour les superpixels. Afin de donner un aperçu des exigences en termes de calcul de l'ensemble de la méthode de segmentation de texture couleur basée sur la FCAM, nous avons estimé son temps d'exécution global sur la base d'images Prague sur un ordinateur Intel Core i7 à 3,60 GHz avec 8 Go de RAM. Les temps de calcul moyens de notre méthode sont consignés dans le tableau 4.3. Ces temps de traitement sont divisés en deux parties : le temps d'entraînement et le temps de segmentation. A titre indicatif, les temps de segmentation des réseaux de neurones convolutifs (EWT-FCNT, FCNT, U-Net, DA et PSP-Net) sont inclus dans le tableau 4.3. Ces temps sont extraits de [118] et ont été mesurés sur un ordinateur portable équipé d'un processeur Intel Core i7 quad-core à 2,5 GHz et de 16 Go de mémoire, équipé d'un GPU externe GTX 1080Ti avec 11 Go de mémoire. Malheureusement, les temps d'entraînement de ces CNN n'ont pas été fournis.

TAB. 4.3: Temps de calcul moyen par image de FCNT, EWT-FCNT, U-Net, DA, PSP-Net et FCAM sur les 20 image de la base Prague.

Phase	**FCNT**	**EWT-FCNT**	**U-Net**	**DA**	**PSP-Net**	**FCAM**
Segmentation	3.61 ms	1.830 s	4.98 ms	3.80 ms	14.39 ms	41.14 s
Apprentissage	-	-	-	-	-	260.9 s

Il est clair que la phase de segmentation de notre méthode nécessite le temps de traitement le plus long, soit 41.14 s pour une image de 512×512. La majeure partie de ce temps est consacrée au calcul de la FCAM pour les superpixels, qui prend 33.12 s. Les étapes restantes comprennent le calcul de SLIC régional, qui prend 8 s, et la classification des superpixels par

ELM, qui prend 0,02 s. Il est important de souligner que ces temps de calcul ont été obtenus en utilisant du code MATLAB sans accélération GPU.

Par contre, la phase d'apprentissage de notre méthode prend moins de 5 min au total, dont 260 s pour le calcul de la FCAM au niveau des prototypes et 0,9 s pour l'entraînement de l'ELM. Le temps d'entraînement global de 260,9 s est beaucoup plus court que celui des CNN, qui nécessitent généralement plusieurs heures sur des ordinateurs puissants équipés de GPU à grande mémoire.

4.3.6 Comparaison avec d'autres classifieurs

Les expériences précédentes ont été menées à l'aide de l'algorithme de classification PPV afin d'évaluer la pertinence des FCAM indépendamment de l'algorithme de classification. Cependant, la méthode de segmentation de texture couleur proposée effectue réellement la classification des superpixels à l'aide de l'algorithme ELM (voir algorithme 2). Pour justifier ce choix, nous avons comparé la moyenne des taux de classification obtenus par ELM sur les 20 images de Prague avec d'autres méthodes classiques de classification supervisée, à savoir la règle du plus proche voisin (PPV), la méthode SVM [158] et le perceptron multicouche (MLP) [159]. Rappelons que le MLP est un réseau de neurones artificiels avec une architecture similaire à celle de l'ELM, mais avec un algorithme d'apprentissage différent et plus lent (algorithme de rétro-propagation).

Le tableau 4.4 montre la moyenne des taux de classification obtenue par chaque classifieur avec les FCAMs cardinales sur les 20 images de la base Prague. Les résultats indiquent clairement la supériorité de l'ELM par rapport aux autres classifieurs, ce qui explique notre choix.

TAB. 4.4: Accuracy moyenne (%) obtenue par notre méthode de segmentation avec différents classifieurs sur la base Prague.

Classifieur	**PPV**	**SVM**	**MLP**	**ELM**
Sans raffinement	87.82	88.98	90.19	**92.60**
Avec raffinement	92.00	92.13	93.13	**95.20**

4.3.7 Comparaison avec les méthodes de l'état de l'art

Pour valider les résultats obtenus précédemment, nous avons comparé notre méthode de segmentation avec d'autres méthodes de segmentation supervisée présentées dans la littérature. La comparaison est menée sur deux bases d'images couleur texturées : la base Prague et la base d'image ALI.

Les performances de la segmentation de ces deux bases ont été évaluées en utilisant les mesures classiques accessibles sur le site web de segmentation de texture de Prague [160], qui incluent : (1) *les critères basés sur les régions:* segmentation correcte (CS), sur-segmentation (OS), sous-segmentation (US), erreur manquée (ME) et erreur de bruit (NE); (2) *les critères basés sur les pixels:* erreur d'omission (O), erreur de commission (C), précision de classe (CA), rappel (CO), précision (CC), erreur de type I (I.), erreur de type II (II.), estimation de la précision de la classe moyenne (EA), score de cartographie (MS), erreur d'estimation de la proportion quadratique moyenne (RM) et indice de comparaison (CI); (3) *les critères d'erreur de cohérence:* erreur de cohérence globale (GCE) et erreur de cohérence locale (LCE). Notons que, la mesure CO correspond réellement aux taux de bonne classification (accuracy).

4.3.7.1 Base d'images Prague

Les méthodes impliquées dans la comparaison de la segmentation des images de la base Prague sont : (1) l'algorithme MRF basé sur un modèle de classification des pixels par champ aléatoire de Markov [98], (2) l'algorithme COF qui utilise les attributs de co-occurrences et le classifieur PPV [160], (3) l'algorithme Con-Col [160], (4) le réseau de neurones entièrement convolutif adapté aux textures (FCNT : fully convolutional network for texture) [118] sans raffinement et (5) avec raffinement, (6) le réseau de neurones entièrement convolutif pour la texture basé sur la transformée en ondelettes empiriques (EWT-FCNT) [118] qui combine la transformée en ondelettes empiriques avec FCNT, (7) U-Net [161], (8) le modèle visuel profond (DA) [119], (9) le réseau de neurones basé sur l'analyse syntaxique de scènes pyramidales (PSP-Net) [120], et (10) notre méthode sans raffinement (FCAM$_c$nr) et (11) avec raffinement (FCAM$_c$wr).

Les résultats de segmentation obtenus par EWT-FCNT, FCNT, MRF, COF et Con-Col proviennent du site web de référence de Prague [160]. Quant à ceux de U-Net, DA et PSP-Net, ils sont tirés de l'étude menée par Huang et al. [118]. Toutes ces méthodes, sauf MRF, COF et Con-Col, impliquent une étape de raffinement comme post-traitement afin d'améliorer les résultats de segmentation.

En ce qui concerne le nombre de prototypes, aucune information n'est disponible pour MRF, COF et Con-Col. En revanche, les réseaux de neurones convolutionnels (CNN) ont été entraînés avec un ensemble d'apprentissage de 1000 images mosaïques de textures de taille 512×512, spécialement créées à partir des images originales de Prague, pour chaque image à segmenter.

Les résultats du tableau 4.5, indiquent que notre méthode est plus efficace que les méthodes supervisées classiques telles que MRF, COF et Con-Col, que ce soit avec ou sans raffinement. Cependant, les méthodes basées sur les CNN, telles que EWT-FCNT, offrent de meilleurs résultats de segmentation. Néanmoins, une analyse approfondie de ces résultats montre que, à l'exception de EWT-FCNT, notre méthode a obtenu des résultats similaires à ceux de U-Net, FCNT, DA et PSP-Net. Les écarts de l'accuracy entre notre méthode (CO = 95.20% avec raffinement) et ces autres méthodes étaient inférieures à 1,7%. En outre, contrairement aux méthodes d'apprentissage profond, notre méthode basée sur les FCAMs cardinales n'a rencontré aucun problème de sous-segmentation ou de sur-segmentation (les mesures US et OS étaient toutes deux égales à 0). Il est également important de signaler que notre méthode sans raffinement a obtenu de meilleurs résultats que FCNT sans raffinement compte tenu de la plupart des critères.

Pour analyser plus en détails ces résultats, nous affichons dans le tableau 4.6 les taux de classification obtenus par notre méthode de segmentation et par les autres méthodes MRF, COF, Con-Col, CNN, FCNT (avec raffinement) et EWT-FCNT, pour chacune des 20 images tests.

TAB. 4.5: Résultats des méthodes supervisées sur la base Prague (20 test images). Les flèches ↑ , ↓ indiquent la direction requise du critère; « nr » indique un résultat sans raffinement de segmentation et « wr » avec raffinement. Les meilleurs résultats sont en gras et les seconds en italique.

Criteria	**MRF**	**COF**	**Con** Col	**FCNT** nr	**FCNT** wr	**EWT** FCNT	**U-Net**	**DA**	**PSP** Net	**FCAM** nr	**FCAM** wr
↑ CS	46.11	52.48	84.57	87.52	96.01	**98.45**	*96.71*	94.18	96.45	84.24	91.27
↓ OS	0.81	**0.00**	**0.00**	**0.00**	1.56	**0.00**	1.71	**0.00**	*0.17*	**0.00**	**0.00**
↓ US	4.18	1.94	1.70	**0.00**	1.20	**0.00**	**0.00**	1.18	*0.41*	**0.00**	**0.00**
↓ ME	44.82	41.55	9.50	6.70	0.78	**0.37**	*0.68*	3.42	1.23	11.45	5.93
↓ NE	45.29	40.97	10.22	6.90	0.89	**0.46**	*0.48*	3.24	1.12	11.39	5.36
↓ O	14.52	20.74	7.00	7.46	2.72	*0.93*	**0.72**	3.13	2.75	4.91	2.96
↓ C	16.77	22.10	5.34	6.16	2.29	*1.04*	**0.70**	1.32	2.39	5.89	2.72
↑ CA	65.42	67.01	86.21	87.08	93.95	**97.67**	*95.86*	94.53	93.89	87.28	91.54
↑ CO	76.19	77.86	92.02	92.61	96.73	**98.78**	*96.91*	96.23	96.06	92.60	95.20
↑ CC	80.30	78.34	92.68	93.26	97.02	**98.81**	*97.38*	97.01	96.41	93.65	95.96
↓ I.	23.81	22.14	7.98	7.39	3.27	**1.22**	*3.09*	3.77	3.94	7.40	4.80
↓ II.	4.82	4.40	1.70	1.49	0.68	**0.25**	*0.41*	0.58	0.69	1.32	0.87
↑ EA	75.40	76.21	91.72	92.68	96.68	**98.77**	*97.01*	96.24	96.08	92.58	95.13
↑ MS	64.29	66.79	88.03	88.92	95.10	**98.17**	*95.37*	94.35	94.08	88.90	92.80
↓ RM	6.43	4.47	2.08	1.38	0.86	**0.24**	*0.61*	1.07	0.70	1.64	1.24
↑ CI	76.69	77.05	92.02	92.81	96.77	**98.78**	*97.08*	96.41	96.15	92.84	95.35
↓ GCE	25.79	23.94	11.76	12.54	5.55	*2.33*	**2.13**	3.50	4.67	11.60	7.45
↓ LCE	20.68	19.69	8.61	9.94	3.75	*1.68*	**1.46**	2.47	3.52	8.76	5.31

FIG. 4.8: Résultats de segmentation sur la base Prague, de gauche à droite: image 07, image 08, image 10 et image 13.

Notre méthode basée sur les $FCAM_c$ montre une accuracy supérieure à 96% pour la moitié des 20 images de test et a surpassé MRF, COF et Con-Col pour presque la totalité des images. Sur les 20 images testées, notre méthode s'est classée deuxième derrière EWT-FCNT pour 6 d'entre elles et a fourni la meilleure accuracy pour l'image 11. La figure 4.9 affiche quelques unes de ces images segmentées. Cependant, pour les images 7, 8, 10 et 13, l'accuracy obtenue reste médiocre (inférieure à 91%), ce qui explique la baisse de l'accuracy moyenne à 95.20%. Les résultats de ces quatre images sont présentés sur la figure 4.8.

m Dans l'ensemble, notre méthode de segmentation utilisant les FCAMs cardinales comme attributs a fourni des résultats de segmentation visuelle satisfaisants et proches de la vérité terrain. Contrairement à d'autres méthodes, la FCAM cardinale n'a pas présenté de sur-segmentation, et les erreurs de classification ont été principalement observées aux limites des régions. Cela est principalement dû à notre pré-segmentation basée sur le SLIC régional qui, malgré sa supériorité sur le SLIC de base, manque de précision pour déterminer correctement les frontières entre deux ou plusieurs régions de textures différentes.

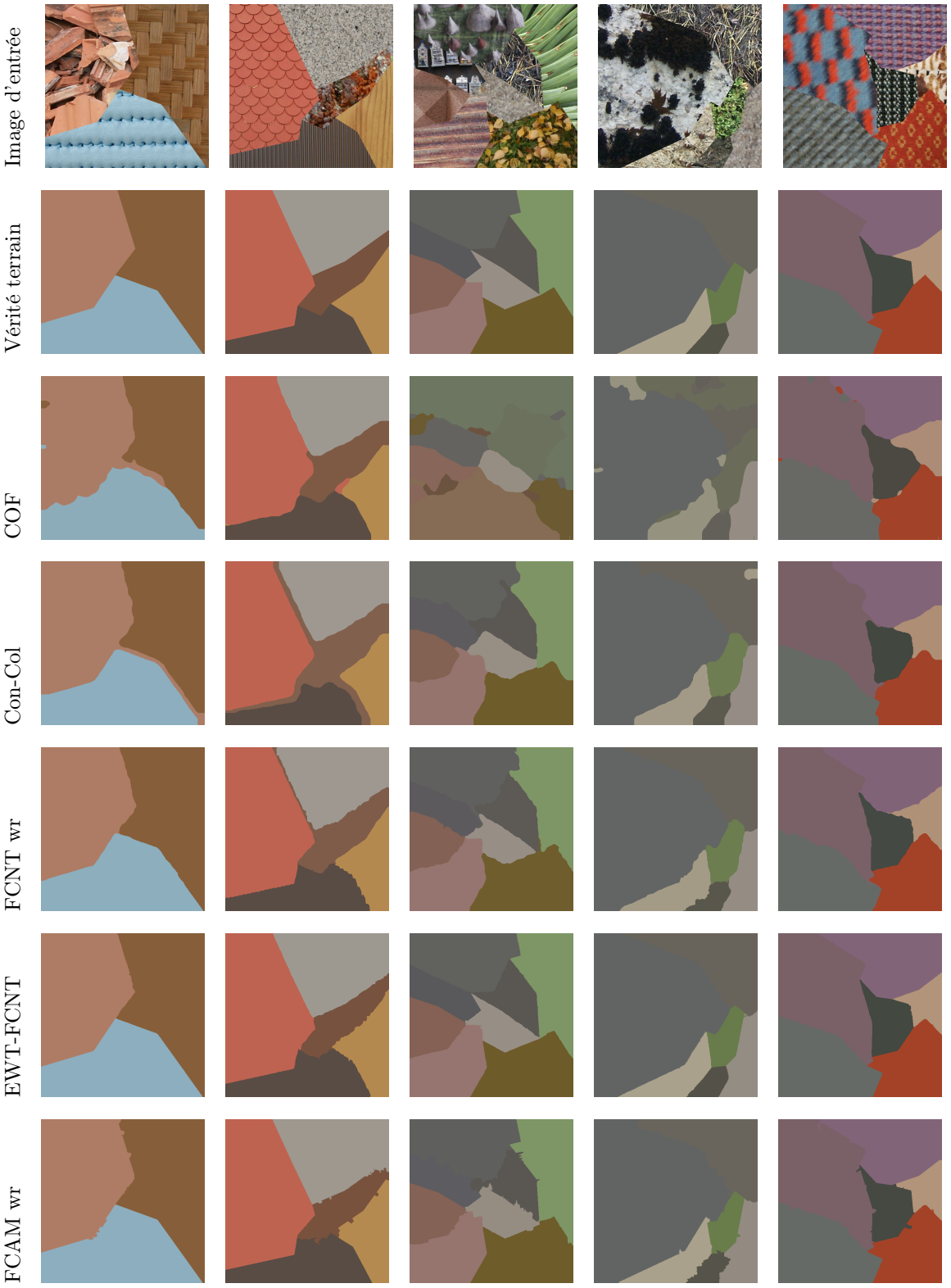


FIG. 4.9: Résultats de segmentation de quelques images de la base Prague, de gauche à droite: image 01, image 03, image 06, image 15, et image 19.

TAB. 4.6: Accuracy (CO) pour chaque image test de la base Prague. « nr » indique un résultat sans raffinement de segmentation et « wr » avec raffinement. Les meilleurs résultats sont en gras et les seconds en italique.

Image	MRF	COF	Con-Col	FCNTwr	EWT-FCNT	FCAMnr	FCAMwr
01	*99.79*	96.19	96.92	99.12	**99.91**	99.54	99.72
02	77.36	93.02	92.70	*97.71*	**99.51**	92.66	96.79
03	95.60	96.56	92.33	97.47	**99.21**	98.66	*99.03*
04	73.20	68.63	93.73	98.41	**98.77**	96.38	*98.58*
05	89.72	89.67	91.64	*96.64*	**98.95**	92.95	93.83
06	84.00	57.59	95.78	97.30	**99.52**	96.97	*97.64*
07	67.74	58.08	89.71	*96.09*	**96.71**	79.65	84.70
08	58.12	71.27	90.10	*95.67*	**98.80**	84.16	90.21
09	72.95	61.29	93.23	*96.70*	**99.35**	94.90	95.47
10	71.52	58.97	85.72	*92.52*	**96.69**	86.36	89.51
11	61.28	80.88	*96.82*	96.72	95.32	94.44	**97.12**
12	81.55	63.01	87.20	*96.03*	**99.51**	92.74	95.84
13	74.35	84.32	77.02	*96.10*	**98.48**	87.99	90.11
14	91.29	90.32	96.51	*97.75*	**99.17**	94.25	96.39
15	57.77	79.61	96.26	97.70	**99.56**	95.31	*98.61*
16	61.33	60.63	91.19	*94.31*	**99.46**	86.23	93.92
17	62.74	72.31	91.92	*96.96*	**99.38**	91.21	95.10
18	77.81	92.96	96.16	98.24	**99.58**	97.18	*98.54*
19	76.07	94.98	97.02	98.55	**99.78**	97.64	*99.39*
20	89.68	86.87	88.48	*94.66*	**97.91**	92.82	93.44
Moyenne	76.19	77.86	92.02	*96.73*	**98.78**	92.60	95.20

Toutefois, il est important de souligner que les méthodes basées sur l'apprentissage profond, y compris EWT-FCNT, exigent la création d'un grand ensemble d'entraînement (1000 images pour chaque image à segmenter) à partir de la base de données d'entraînement originale. De plus, elles nécessitent une étape d'apprentissage coûteuse en temps de calcul pour extraire les attributs et classifier les pixels.

4.3.7.2 Base d'images ALI

La base d'images ALI est également fournie dans le générateur de disponible de segmentation de texture de Prague (Prague texture segmentation Data-generator) [162]. Cette base

contient 31 textures extraites d'images réelles de télédétection acquises par le capteur Advanced Land Imager (ALI) et 10 images synthétiques réservées aux tests, ainsi que leur vérité terrain correspondante. Ces images ont également été générées de manière synthétique à l'aide d'un algorithme basé sur un générateur aléatoire de polygones de Voronoi à partir du même nombre d'images originales. Chaque image, de taille 512×512 pixels, comporte 10 bandes spectrales et une résolution spatiale de 30 mètres par pixel. Le nombre de classes de texture représentées dans chaque image varie de 3 à 12. Contrairement à la base Prague précédente, plusieurs images d'apprentissage sont fournies pour chaque classe. Les textures sont naturelles et les limites sont rectilignes.

La figure 4.10 montre trois images sélectionnées dans la base d'images ALI, accompagnées de leurs images de vérité terrain correspondantes (première ligne). La deuxième à la quatrième ligne présentent également quelques images de la base d'apprentissage utilisées pour générer les images synthétiques.

Pour évaluer les performances de notre méthode de segmentation basée sur la FCAM_c (sans et avec raffinement) sur la base d'image ALI, nous comparons ses résultats avec d'autres méthodes de segmentation supervisée [163, 164] : eCognition (eCog) [163], la méthode Dynamic Hierarchical Classification with manual selection (DHC/M) [97], et les méthodes CNN proposées par Basaeed et al. avec fusion intra-bande avec raffinement (BCCNN-wr) et sans raffinement (BCCNN-nr) [164], et la méthode No-Boosting Convolutional Neural Networks (NBCNN) [162].

Sur le tableau 4.7, nous constatons que la méthode basée sur FCAM_c se distingue selon plusieurs critères par rapport à d'autres méthodes : la méthode FCAM_c affiche un taux de Segmentation Correcte (CS) maximal de 97.69%, ce qui indique que la plupart des régions texturées sont correctement détectées. La FCAM_c affiche un score de Sous-segmentation (US) et de Sur-segmentation (OS) nuls (0.00), indiquant que le nombre de régions dans l'image segmentée est le même que celui de l'image vérité terrain et aucune région n'a été divisée, ni regroupée avec d'autres régions. La FCAM_c présente une Précision par Classe (CA) de 97.19% qui avoisine celui de BCCNN (97.88%). Ceci démontre que la FCAM_c est performante pour toutes les classes de textures. FCAM_c obtient un taux de classification maximal de 98.53% indiqué par le critère CO. Le taux d'erreur (I) obtenu par la FCAM_c est

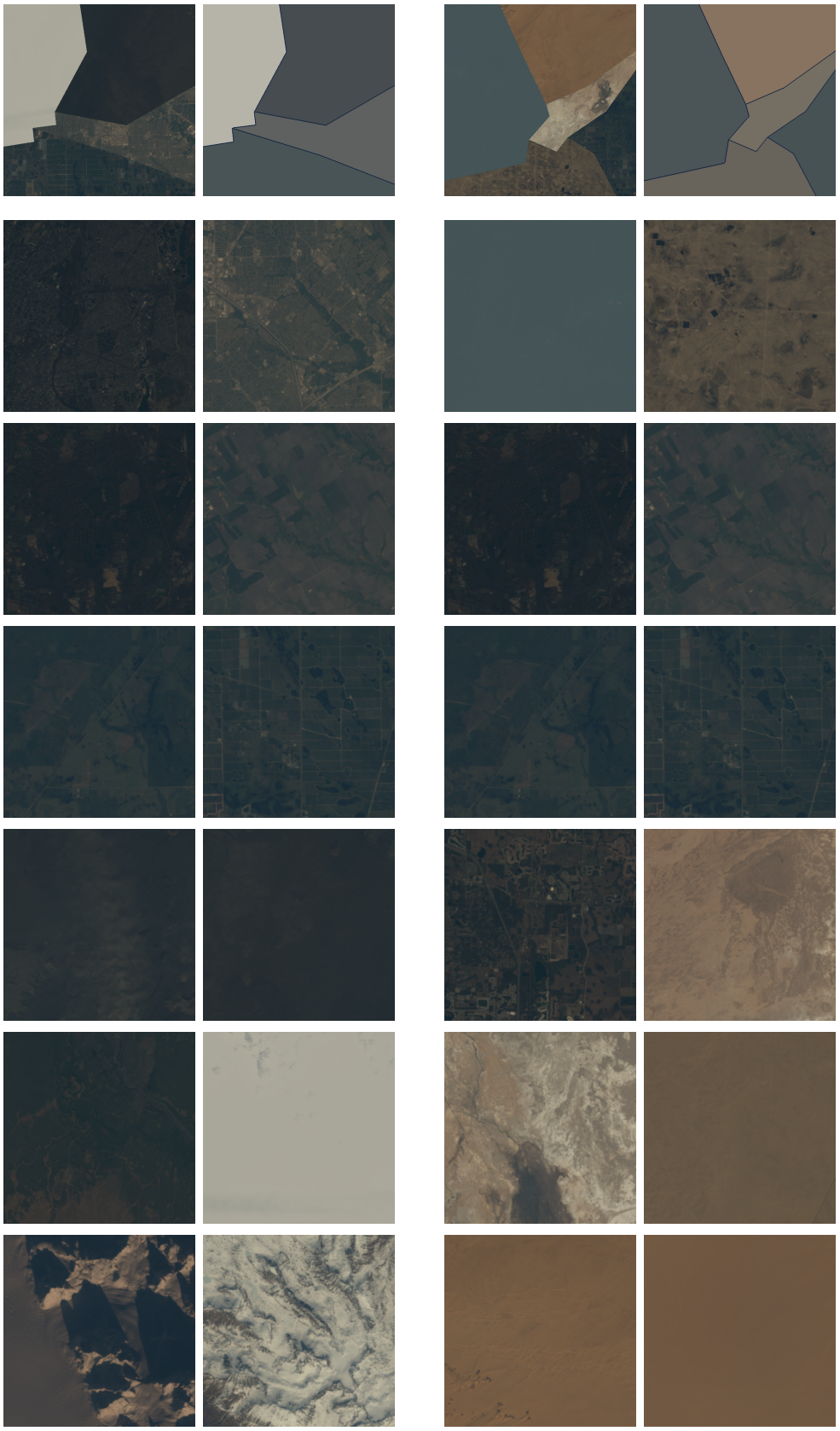


FIG. 4.10: Exemples d'images couleur texturées accompagnées de leurs images de vérité terrain **(première ligne)**, ainsi que des images de leur base d'apprentissage **(de la deuxième à la septième ligne)**.

TAB. 4.7: Résultats des méthodes supervisées sur la base ALI. Les flèches ↑ , ↓ indiquent la direction requise du critère; « nr » indique un résultat sans raffinement de segmentation et « wr » avec raffinement. Les meilleurs résultats sont en gras et les seconds en italique.

Method	BCCNN wr	BCCNN nr	NBCNN	eCog	DHC/M	FCAM$_c$ nr	FCAM$_c$ wr
↑ CS	96.62	96.03	94.27	91.91	84.59	*97.25*	**97.69**
↓ OS	*4.02*	25.43	47.42	10.54	6.78	**0.00**	**0.00**
↓ US	**0.00**	**0.00**	4.72	*1.11*	8.73	**0.00**	**0.00**
↓ ME	**0.00**	**0.00**	**0.00**	1.20	4.70	*0.47*	1.15
↓ NE	**0.00**	**0.00**	**0.00**	0.98	5.33	*0.41*	0.87
↓ O	*0.27*	0.34	0.36	**0.06**	1.69	2.10	1.52
↓ C	0.91	25.22	50.13	**0.53**	*0.70*	1.34	0.88
↑ CA	**97.88**	97.16	96.64	94.13	89.69	95.54	*97.19*
↑ CO	*98.19*	97.46	97.73	95.42	92.80	97.56	**98.53**
↑ CC	*99.69*	**99.70**	98.90	98.16	92.78	97.63	98.60
↓ I.	*1.81*	2.54	2.27	4.58	7.20	2.44	**1.47**
↓ II.	**0.06**	**0.06**	0.43	0.24	0.90	0.35	*0.23*
↑ EA	**98.66**	98.22	97.60	96.13	92.29	97.56	*98.51*
↑ MS	**98.03**	97.30	97.01	94.40	89.59	96.34	*97.79*
↓ RM	0.75	0.80	0.69	1.62	1.94	*0.37*	**0.34**
↑ CI	**98.79**	98.39	97.70	96.44	92.53	97.58	*98.54*
↓ GCE	0.63	*0.60*	**0.51**	2.67	4.38	4.43	2.68
↓ LCE	*0.43*	*0.43*	**0.29**	1.16	2.67	3.59	1.82
↓ dD	**1.07**	1.43	*1.23*	2.91	4.61	2.44	1.47
↓ dM	**0.50**	0.77	1.03	1.24	2.17	1.10	*0.64*

par conséquent le plus bas et l'erreur quadratique moyenne (RM=0.34%) entre le nombre prédit de pixels et le nombre réel de pixels dans chaque classe est le plus faible. La méthode BCCNN se montre plus performante compte tenu des autres critères tels que l'estimation de la précision de la classe moyenne (EA = 98.66%), le score de cartographie (MS = 98.03%), et l'indice de comparaison (CI = 98.79%).

4.4 Conclusion

Dans ce chapitre, nous avons exploité les matrices aura cardinales de couleur floues (FCAM$_c$) pour caractériser localement les textures couleur et segmenter des images couleur texturées.

La méthode de segmentation proposée est basée sur la classification de superpixels, générés à partir d'une version modifiée de l'algorithme SLIC pour incorporer des informations régionales. Une FCAM cardinale est calculée pour chaque superpixel grâce à une fonction de voisinage localement adaptative. Les superpixels sont finalement classés à l'aide d'un simple classifieur supervisé (ELM).

Des expériences menées sur deux bases d'images (Prague et ALI) ont montré que la segmentation d'images de textures couleur proposée surpasse les méthodes de segmentation classiques de l'état de l'art et est compétitive avec les méthodes récentes basées sur l'apprentissage profond. Cependant, contrairement aux approches basées sur les CNNs qui nécessitent une procédure d'apprentissage coûteuse et un grand ensemble d'images de textures segmentées pour l'entraînement, notre méthode est appliquée directement à partir d'une base de données beaucoup plus petite.

Malgré ses performances, notre méthode de segmentation reste sensible aux résultats de présegmentation. Bien que le SLIC régional améliore les résultats de segmentation par rapport au SLIC de base, la détection des frontières de texture couleur n'est pas encore suffisamment précise.

Conclusion générale et perspectives

Nous avons développé dans cette thèse une méthode d'analyse des textures couleur basée sur les matrices aura des couleurs floues. Ces matrices sont utilisées comme attributs pour classifier et segmenter des images couleur texturées. Nos contributions dans cette thèse sont présentées dans les quatre chapitres de ce manuscrit.

1 Synthèse des chapitres

Dans le premier chapitre, nous avons présenté quelques notions fondamentales sur la couleur et la texture. Cela nous a permis, d'une part, de distinguer les textures en niveaux de gris des textures couleur et, d'autre part, de mettre en évidence le caractère flou et imprécis des textures, qu'elles soient en niveaux de gris ou en couleur. Ceci nous a conduit à établir un état de l'art sur les différentes approches d'analyse d'images couleur texturées, en portant une attention particulière aux attributs flous et à leurs implications dans la classification et la segmentation des images couleur texturées. Il ressort de cette étude que très peu de méthodes ont été consacrées à l'extraction des attributs de textures couleur floues.

C'est ainsi que nous nous sommes intéressé, dans le deuxième chapitre, à la méthode d'extraction des attributs de texture basée sur les matrices aura des niveaux de gris (GLAMs) et leur extension aux images couleur floues. Les GLAMs se basent sur un concept théorique original d'ensemble aura et de mesure aura qui évalue le degré de mélange entre les deux ensembles de pixels voisins. Elle offre un moyen efficace pour caractériser les interactions spatiales, définies par une fonction de voisinage, entre les niveaux de gris présents dans une image. Deux types de GLAMs sont définis en fonction de la mesure utilisée, GLAM locale et GLAM cardinale, puis étendues au cadre flou. Les matrices aura des niveaux de gris flous

(Fuzzy Gray Level Aura Matrices, FGLAMs) résultant de cette extension ont été appliquées aux images couleur suivant une approche marginale. Dans cette approche, trois MFCLAMs (Marginal Fuzzy Color Level Aura Matrices) sont utilisées pour caractériser les interactions spatiales entre les différents niveaux au sein d'une même composante couleur. Cependant, cette approche marginale ne tient pas compte des interactions entre les composantes couleurs. Pour pallier cet inconvénient, nous avons proposé une autre approche (opposée ou conjointe) dans laquelle six matrices (inter-composantes) sont définies pour caractériser les interactions spatiales entre les différents niveaux de deux composantes couleur différentes. Ces matrices OFCLAMs (Opponent Fuzzy Color Level Aura Matrices), qui constituent notre première contribution, engendrent un grand nombre d'attributs, ce qui les rend difficilement exploitables, surtout dans le cadre de la segmentation d'images.

Pour remédier à ces problèmes, nous avons proposé, dans le troisième chapitre, notre deuxième contribution, à savoir une autre approche (compacte) capable de caractériser complètement une texture couleur par une seule matrice aura couleur floue (FCAM). Cette FCAM a l'avantage de caractériser les interactions spatiales entre les différentes couleurs, tout en réduisant l'espace mémoire et le temps requis pour son calcul.

Des résultats de classification des textures couleur obtenus sur la base d'images *Outex-TC-13* et ses versions dégradées par la rotation (*Outex-TC-30*), la résolution (*Outex-TC-31*), le bruit gaussien additif (*Outex-TC-32*), et le flou *Outex-TC-33*, ont révélé :

- L'importance du flou puisque les OFCLAMs et les FCAMs se sont révélées plus performantes que leurs versions crisp (OCLAMs) et (CLAMs).
- La supériorité de la FCAM cardinale (FCAM_c) sur la FCAM locale (FCAM_l).
- La robustesse des FCAMs vis-à-vis surtout du bruit gaussien et du flou, et des OFCAMs vis-à-vis de la rotation et de la résolution.
- Le bon compromis entre précision et rapidité de la FCAM, puisque ses résultats de classification sont proches de ceux obtenus avec la stratégie opposée (OFCLAM) et meilleurs que ceux de la stratégie marginale (MFCLAM), mais avec un temps de calcul et un espace mémoire bien plus réduits.

D'autres tests menés sur trois bases de textures couleur *TC-13*, *TC-31* et *KTH-TIPS* ont montré l'efficacité des FCAMs par rapport à plusieurs méthodes d'extraction d'attributs de l'état de l'art et leur compétitivité avec les CNNs.

Notre troisième contribution dans cette thèse est une méthode de segmentation supervisée d'images couleur texturées que nous avons proposée dans le quatrième chapitre. Dans cette méthode, une FCAM cardinale est calculée localement pour chaque superpixel grâce à une fonction de voisinage localement adaptative. Les superpixels sont par la suite classés à l'aide d'un simple classifieur supervisé (ELM). En plus d'utiliser les FCAMs, l'originalité de cette méthode réside dans l'algorithme SLIC régional développé afin de réduire le nombre de pixels à traiter en pré-segmentant l'image en superpixels. Les résultats obtenus sur la base d'images couleur texturées Prague et la base d'images satellitaire ALI attestent, d'une part, de la pertinence des FCAMs cardinales à discriminer des textures couleur et démontrent que notre méthode de segmentation est meilleure que les méthodes classiques et très compétitive par rapport aux méthodes récentes par apprentissage profond.

2 Perspectives

Les résultats auxquels nous sommes parvenus suscitent des réflexions et ouvrent la voie à des perspectives pouvant faire l'objet de travaux futurs.
Certaines de ces perspectives concernent directement les paramètres des FCAMs et OF-CLAMs, comme:

- La fonction d'appartenance : Trois fonctions d'appartenance ont été utilisées dans notre travail (Gaussienne, Triangulaire et FCM). Il serait intéressant de tester d'autres fonctions comme la fonction trapézoïdale et celle de Bell généralisée.
- La fonction de voisinage flou : Pour effectuer nos tests, nous avons employé une fonction de voisinage crisp. Ceci ne nous a pas permis d'exploiter complètement le flou pour cette fonction. Nous préconisons d'utiliser des fonctions de voisinage flou comme celles utilisées en morphologie mathématique floue.
- L'espace couleur RGB : Dans cet espace, utilisé tout au long de cette thèse, les trois composantes sont fortement corrélées. L'utilisation d'autres espaces couleur tels que

La*b*, HSV, I_1,I_2,I_3, pourrait être une solution pour éviter cet inconvénient et améliorer le pouvoir discriminant des FCAMs et OFCAMs.

– Couleur : Les couleurs nettes, qui sont les équivalentes des couleurs floues, sont choisies à l'aide d'une simple procédure de quantification uniforme sans tenir compte des couleurs présentes dans l'image. Une voie d'amélioration des résultats de classification et de segmentation serait d'effectuer une quantification plus élaborée.

Les matrices OFCLAMs sont censées fournir de meilleurs résultats en augmentant le nombre de niveaux de couleur, mais au détriment d'un nombre d'attributs très élevé. Pour réduire ce nombre, nous envisageons d'extraire des attributs de Haralick à partir de ces matrices.

En ce qui concerne la méthode de segmentation proposée, celle-ci demeure sensible aux résultats de pré-segmentation. Malgré l'amélioration apportée par le SLIC régional par rapport au SLIC de base, la détection des frontières qui séparent les textures n'est pas encore suffisamment précise. Dans les travaux futurs, nous avons l'intention d'utiliser une procédure de superpixelisation sensible à la texture couleur. En outre, cette méthode de segmentation a été développée dans un contexte supervisé ; nous prévoyons également de l'adapter à la segmentation non supervisée.

Annexe A

Propriétés des sous-ensembles flous

Dans cette annexe, nous présenterons les opérations les plus couramment utilisées sur les sous-ensembles flous qui sont équivalentes aux opérations classiques.

Soit A, B deux sous-ensembles flous de l'ensemble X, définis respectivement par leurs fonctions d'appartenance μ_A et μ_B. $\mu_A(x)$ et $\mu_B(x)$ représentent les degrés d'appartenance de l'élément x au sous-ensemble A et B, respectivement.

Noyau d'un sous-ensemble flou : Le noyau d'un sous-ensemble flou A, noté $Noy(A)$, est défini comme suit:

$$Noy(A) = \{x \in X / \mu_A(x) = 1\} \tag{A.1}$$

Point de croisement d'un sous-ensemble flou : Le point de croisement d'un sous-ensemble flou A de X, noté $Cr(A)$, est l'ensemble des éléments de X pour lesquels le degré d'appartenance au sous-ensemble flou A est égal à 0.5.

$$Cr(A) = \{x \in X / \mu_A(x) = 0.5\} \tag{A.2}$$

Hauteur d'un sous-ensemble flou : La hauteur du sous-ensemble flou A, notée $h(A)$, est le plus fort degré avec lequel un élément de X appartient à A. Un sous-ensemble flou A est normalisé si $h(A) = 1$.

$$h(A) = \{\sup_{x \in X} \mu_A(x)\} \tag{A.3}$$

Support d'un sous-ensemble flou : Le support d'un sous-ensemble flou A de X, noté Supp(A), correspond à toutes les valeurs x de X pour lesquelles $\mu_A(x) \neq 0$.

$$\text{Supp}(A) = \{x \in X / \mu_A(x) \neq 0\} \tag{A.4}$$

Cardinal d'un ensemble flou : Le cardinale du sous-ensemble flou A, notée $|A|$, est le nombre d'éléments appartenant à A pondéré par leur degré d'appartenance.

$$|A| = \sum_{x \in A} \mu_A(x) \tag{A.5}$$

Égalité : On dit que deux ensembles flous A et B de X sont égaux, si leurs fonctions d'appartenance prennent la même valeur pour tous les éléments x de X. On a $A = B$ si et seulement si :

$$\forall x \in X, \mu_A(x) = \mu_B(x) \tag{A.6}$$

Inclusion : On dit que A est inclus dans B, qu'on note alors $A \subseteq B$, si tout élément x de X qui appartient à A appartient aussi à B avec un degré au moins aussi grand. On a $A \subseteq B$ si et seulement si :

$$\forall x \in X, \mu_A(x) \leq \mu_B(x) \tag{A.7}$$

Union : L'union de deux sous-ensembles flous A et B de X est le sous-ensemble flou constitué des éléments de X affectés du plus grand des degrés avec lesquels ils appartiennent à A et B. On a $A \cup B$ est défini par :

$$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x)), \quad \forall x \in X \tag{A.8}$$

Intersection : L'intersection de deux sous-ensembles flous A et B de X est le sous-ensemble flou $A \cap B$ défini par la fonction d'appartenance :

$$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)), \quad \forall x \in X \tag{A.9}$$

Complémentaire : Le complémentaire d'un sous-ensemble flou A, noté $\bar{A}$, est défini par :

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x), \quad \forall x \in X \tag{A.10}$$

Annexe B

Critères d'évaluation

Cette annexe présente les différents critères utilisés pour l'évaluation de la qualité des résultats de segmentation d'images.

Soient $X = \{x_1,x_2,\ldots,x_N\}$ l'ensemble des pixels de l'image (N étant le nombre total des pixels de l'image), S la segmentation à évaluer, et S' la segmentation de référence. On note par R_i, $i = 1,\ldots,k$, l'ensemble des régions de S et par R'_j, $j = 1,\ldots,k'$, l'ensemble des régions de S'. $N_i = |R_i|$ est le cardinal de la région R_i et $N'_j = |R'_j|$ est le cardinal de la région R'_j (on a $\sum_{i=1}^{k} N_i = \sum_{j=1}^{k'} N'_j = N$). La matrice de confusion est un outil qui permet de mesurer la concordance entre les pixels de S et de S'. Elle est de taille $k \times k'$. Chaque élément de cette matrice est le nombre de pixels en commun entre les régions $R_i \in S$ et $R'_j \in S'$, noté N_{ij}, et défini par: $N_{ij} = |R_i \cap R'_j|$.

B.1 Critères basés sur les régions

Hoover [165] a proposé cinq critères pour comparer les régions des segmentations S et S'. Ces critères sont basés sur un seuil de recouvrement minimum t défini par l'utilisateur dans l'intervalle [0.5 1] (dans nos évaluations t est fixé à 0.75).

↑ **CS : Correct Detection**. Les régions R_i dans S et R'_j dans S' sont classées en détection correcte si et seulement si:

$$\begin{cases} |R_i \cap R'_j| \geq t \times |R_i| \\ |R_i \cap R'_j| \geq t \times |R'_j| \end{cases} \tag{B.1}$$

↓ **OS : Over-Segmentation**. La sur-segmentation est détectée si la région R'_j dans S' est divisée en plusieurs régions $R_{i1}, \ldots, R_{il}$ (avec $l \in [2 \;\; k]$) dans S comme suit:

$$\begin{cases} \forall s \in [1 \;\; l], \quad |R_{is} \cap R'_j| \geq t \times |R_{is}| \\ \sum_{s=1}^{l} |R_{is} \cap R'_j| \geq t \times |R'_j| \end{cases} \tag{B.2}$$

↓ **US : Under-Segmentation**. La sous-segmentation est détectée si la région R_i dans S est divisée en plusieurs régions $R'_{j1}, \ldots, R'_{jl}$ (avec $l \in [2 \;\; k']$) dans S' comme suit:

$$\begin{cases} \sum_{s=1}^{l} |R_i \cap R'_{js}| \geq t \times |R_i| \\ \forall s \in [1 \;\; l], \quad |R_i \cap R'_{js}| \geq t \times |R'_{js}| \end{cases} \tag{B.3}$$

↓ **ME : Missed Error**. La région R'_j dans S' est non détectée dans S si et seulement si :

$$\begin{cases} R'_j \notin & \text{Correcte segmentation} \\ R'_j \notin & \text{Sur-segmentation} \\ R'_j \notin & \text{Sous-segmentation} \end{cases} \tag{B.4}$$

↓ **NE : Noise Error**. La région R_i dans S est une région bruit (R_i ne possède pas de correspondante dans S') si et seulement si:

$$\begin{cases} R_i \notin & \text{Correcte segmentation} \\ R_i \notin & \text{Sur-segmentation} \\ R_i \notin & \text{Sous-segmentation} \end{cases} \tag{B.5}$$

B.2 Critères basés sur les pixels

Étant donné $N_{i,\bullet} = \sum_{j=1}^{k'} N_{i,j}$ et $N_{\bullet,i} = \sum_{j=1}^{k} N_{j,i}$ avec k et k' sont respectivement le nombre de régions (classes) de S et S', et $N_{i,j}$ le nombre de pixels attribués à la i-ème classe mais

qui appartiennent à la j-ème classe. Soit $K = \max\{k,k'\}$. La matrice de confusion étendue à $K \times K$ est obtenue en remplissant les entrées $\{N_{i,j}\}$ manquantes par des zéros. Dans le cas supervisé $\hat{\imath}$ est i alors que dans le cas non supervisé $\hat{\imath}$ est la projection de la i-ème classe de S' dans S en utilisant la méthode de Munkres [166].

↓ **O : Omission error**. L'erreur d'omission représente la proportion globale des pixels perdus par une classe lors de la classification. Elle est définie par:

$$O = \text{median} \left\{ \frac{O_i}{N_{\bullet,i}} \right\}_{i=1}^{k'} = \text{median} \left\{ \frac{(N_{\bullet,i} - N_{\hat{\imath},i})}{N_{\bullet,i}} \right\}_{i=1}^{k'} \tag{B.6}$$

O_i est l'erreur d'omission de la i-ème classe.

↓ **C : Commission error**. L'erreur de commission représente la proportion globale des pixels ajoutés par une classe lors de la classification. Elle est définie par:

$$C = \text{median} \left\{ \frac{C_i}{N_{\hat{\imath},\bullet}} \right\}_{i=1}^{k} = \text{median} \left\{ \frac{(N_{\hat{\imath},\bullet} - N_{\hat{\imath},i})}{N_{\hat{\imath},\bullet}} \right\}_{i=1}^{k} \tag{B.7}$$

C_i est l'erreur de commission de la i-ème classe.

↑ **CA : Class Accuracy**. La mesure de précision de classe évalue le taux de confusion entre les classes après classification, telle que:

$$CA = \frac{1}{N} \sum_{i=1}^{K} \frac{N_{\hat{\imath},i} N_{\bullet,i}}{N_{\bullet,i} + N_{\hat{\imath},\bullet} - N_{\hat{\imath},i}} \tag{B.8}$$

↑ **CO : Recall - Accuracy**. Le rappel (taux de classification) représente le taux des pixels bien classés. Il est défini comme suit:

$$CO = \frac{1}{N} \sum_{i=1}^{K} N_{\bullet,i} CO_i = \frac{1}{N} \sum_{i=1}^{K} N_{\hat{\imath},i} \tag{B.9}$$

CO_i est le rappel de la i-ème classe.

↑ **CC : Precision**:

$$CC = \frac{1}{N} \sum_{i=1}^{K} N_{\bullet,i} CC_i = \frac{1}{N} \sum_{i=1}^{K} \frac{N_{\hat{\imath},i} N_{\bullet,i}}{N_{\hat{\imath},\bullet}} \tag{B.10}$$

CC_i est la précision de la i-ème classe.

↓ **I. : Type I error**:

$$I = \frac{1}{N}\sum_{i=1}^{K}\left(N_{\bullet,i} - N_{\hat{\imath},i}\right) = 1 - CO \tag{B.11}$$

↓ **II. : Type II error**:

$$II = \frac{1}{N}\sum_{i=1}^{K}\frac{N_{\hat{\imath},\bullet}N_{\bullet,i} - N_{\hat{\imath},i}N_{\bullet,i}}{N - N_{\bullet,i}} \tag{B.12}$$

↑ **EA : Mean Class Accuracy Estimate**:

$$EA = \frac{1}{N}\sum_{i=1}^{K}\frac{2N_{\hat{\imath},i}N_{\bullet,i}}{N_{\bullet,i} + N_{\hat{\imath},\bullet}} \tag{B.13}$$

↑ **MS : Mapping Score**:

$$MS = \frac{1}{N}\sum_{i=1}^{K}\left(1.5N_{\hat{\imath},i} - 0.5N_{\hat{\imath},\bullet}\right) \tag{B.14}$$

↓ **RM : Root Mean Square Proportion Estimation Error** :

$$RM = \sqrt{\frac{1}{K}\sum_{i=1}^{K}\left(\frac{N_{\hat{\imath},\bullet} - N_{\bullet,i}}{N}\right)^2} \tag{B.15}$$

↑ **CI : Comparison Index** :

$$CI = \frac{1}{N}\sum_{i=1}^{K}N_{\hat{\imath},i}\sqrt{\frac{N_{\bullet,i}}{N_{\hat{\imath},\bullet}}} = \frac{1}{N}\sum_{i=1}^{K}N_{\bullet,i}\sqrt{CC_iCO_i} \tag{B.16}$$

B.3 Critères basés sur les erreurs de cohérences

Martin et al. [167] ont défini quatre critères basés sur les erreurs de cohérences pour comparer les segmentations S et S'. Ces critères sont calculés sur chaque pixel de l'image à partir des erreurs, dites de raffinement. Pour un pixel x qui appartient à la région R_i dans S et à R'_j dans S', ces erreurs sont: i) l'erreur de raffinement local de S par rapport à S' définie par $LRE(S,S',x) = \frac{|R_i \backslash R'_j|}{|R_i|}$ et ii) l'erreur de raffinement local de S' par rapport à S définie par $LRE(S',S,x) = \frac{|R'_j \backslash R_i|}{|R'_j|}$. Ces quatre critères sont les suivants:

↓ **LCE : Local Consistency Error** :

$$LCE(S,S') = \frac{1}{N}\sum_{x}^{N} \min\left\{LRE(S,S',x), LRE(S',S,x)\right\} \tag{B.17}$$

↓ **GCE : Global Consistency Error** :

$$GCE(S,S') = \frac{1}{N}\min\left\{\sum_{x}^{N} LRE(S,S',x), \sum_{x}^{N} LRE(S',S,x)\right\} \tag{B.18}$$

B.4 Critères de comparaison de classification

Ces critères se basent sur le calcul d'une métrique entres les deux segmentations S et S'.

↓ **dM : Mirkin Metric.** Cette métrique est liée au nombre pixels en désaccord dans les segmentations S et S'. Elle est définie comme suit:

$$dM(S,S') = \frac{1}{N^2}\left(\sum_{i}^{k} N_i^2 + \sum_{j}^{k'} N'^2_j - 2\sum_{i}^{k}\sum_{j}^{k'} N_{ij}^2\right) \tag{B.19}$$

↓ **dD : Van Dongen Metric [168].** Cette métrique est également basée sur la distance de Hamming (DHD) entre les segmentations S et S'. Elle est définie par:

$$dD(S,S') = \frac{DHD(S,S') + DHD(S',S)}{2N} \tag{B.20}$$

$$dD(S,S') = 1 - \frac{1}{2N}\left(\sum_{i}^{k} \max_{j} N_{ij} - \sum_{j}^{k'} \max_{i} N_{ij}\right) \tag{B.21}$$

Bibliographie

[1] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[2] Francesco Bianconi, Raquel Bello-Cerezo, and Paolo Napoletano. Improved opponent color local binary patterns: an effective local image descriptor for color texture classification. *Journal of Electronic Imaging*, 27(1):011002–011002, 2018.

[3] Paolo Napoletano. Hand-crafted vs learned descriptors for color texture classification. In *Computational Color Imaging: 6th International Workshop, CCIW 2017, Milan, Italy, March 29-31, 2017, Proceedings 6*, pages 259–271. Springer, 2017.

[4] Vincent Arvis, Christophe Debain, Michel Berducat, and Albert Benassi. Generalization of the cooccurrence matrix for colour images: application to colour texture classification. *Image Analysis & Stereology*, 23(1):63–72, 2004.

[5] Christoph Palm. Color texture classification by integrative co-occurrence matrices. *Pattern Recognition*, 37(5):965–976, 2004.

[6] Audrey Ledoux, Olivier Losson, and Ludovic Macaire. Texture classification with fuzzy color co-occurrence matrices. In *Procs. IEEE International Conference on Image Processing (ICIP 2015)*, pages 1429–1433, Québec, Canada, September 2015.

[7] Ibrahim M. Elfadel and Rosalind W. Picard. Gibbs random fields, cooccurrences, and texture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):24–37, 1994.

[8] Kamal Hammouche, Olivier Losson, and Ludovic Macaire. Fuzzy aura matrices for texture classification. *Pattern Recognition*, 53:212–228, 2016.

[9] Zohra Haliche and Kamal Hammouche. The gray level aura matrices for textured image segmentation. *Analog Integrated Circuits and Signal Processing*, 69:29–38, 2011.

[10] Zohra Haliche, Kamal Hammouche, and Jack-Gérard Postaire. Texture image segmentation based on the elements of gray level aura matrices. In *2014 Global Summit on Computer & Information Technology (GSCIT)*, pages 1–6. IEEE, 2014.

[11] Nicolas Vandenbroucke, Laurent Busin, and Ludovic Macaire. Unsupervised color-image segmentation by multicolor space iterative pixel classification. *Journal of Electronic Imaging*, 24(2):023032–023032, 2015.

[12] Yu-Ichi Ohta, Takeo Kanade, and Toshiyuki Sakai. Color information for region segmentation. *Computer graphics and image processing*, 13(3):222–241, 1980.

[13] Farid Garcia-Lamont, Jair Cervantes, Asdrúbal López, and Lisbeth Rodriguez. Segmentation of images by color features: A survey. *Neurocomputing*, 292:1–27, 2018.

[14] Yining Deng, BS Manjunath, Charles Kenney, Michael S Moore, and Hyundoo Shin. An efficient color representation for image retrieval. *IEEE Transactions on image processing*, 10(1):140–147, 2001.

[15] Haim Permuter, Joseph Francos, and Ian Jermyn. A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern recognition*, 39(4):695–706, 2006.

[16] Jing Huang, S Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zabih. Image indexing using color correlograms. In *Proceedings of IEEE computer society conference on Computer Vision and Pattern Recognition*, pages 762–768. IEEE, 1997.

[17] Imtnan-Ul-Haque Qazi, Olivier Alata, and Zoltan Kato. Parametric stochastic modeling for color image segmentation and texture characterization. *Advanced Color Image Processing and Analysis*, pages 279–325, 2013.

[18] Jack Sklansky. Image segmentation and feature extraction. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(4):237–247, 1978.

[19] Li Liu, Jie Chen, Paul Fieguth, Guoying Zhao, Rama Chellappa, and Matti Pietikäinen. From BoW to CNN: Two decades of texture representation for texture classification. *International Journal of Computer Vision*, 127(1):74–109, 2019.

[20] Anne Humeau-Heurtier. Texture feature extraction methods: A survey. *IEEE access*, 7:8975–9000, 2019.

[21] Charles Bouman and Bede Liu. Multiple resolution segmentation of textured images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):99–113, 1991.

[22] James M Keller, Susan Chen, and Richard M Crownover. Texture description and segmentation through fractal geometry. *Computer Vision, Graphics, and image processing*, 45(2):150–166, 1989.

[23] Bangalore S Manjunath and Rama Chellappa. Unsupervised texture segmentation using markov random field models. *IEEE transactions on pattern analysis and machine intelligence*, 13(5):478–482, 1991.

[24] Anil K Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern recognition*, 24(12):1167–1186, 1991.

[25] Michael Unser. Texture classification and segmentation using wavelet frames. *IEEE Transactions on image processing*, 4(11):1549–1560, 1995.

[26] Jarbas Joaci de Mesquita Sá Junior and André Ricardo Backes. ELM based signature for texture classification. *Pattern Recognition*, 51:395–401, 2016.

[27] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29:51–59, 1996.

[28] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[29] Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE transactions on image processing*, 19(6):1657–1663, 2010.

[30] Hongliang Jin, Qingshan Liu, Hanqing Lu, and Xiaofeng Tong. Face detection using improved lbp under bayesian framework. In *Third International Conference on Image and Graphics (ICIG'04)*, pages 306–309. IEEE, 2004.

[31] Dong-Chen He and Li Wang. Texture unit, texture spectrum, and texture analysis. *IEEE transactions on Geoscience and Remote Sensing*, 28(4):509–512, 1990.

[32] Alexandru Drimbarean and Paul F Whelan. Experiments in colour texture analysis. *Pattern recognition letters*, 22(10):1161–1167, 2001.

[33] Francesco Bianconi, Richard Harvey, Paul Southam, and Antonio Fernández. Theoretical and experimental comparison of different approaches for color texture classification. *Journal of Electronic Imaging*, 20(4):043006–043006, 2011.

[34] Anne Humeau-Heurtier. Color texture analysis: A survey. *IEEE Access*, 10:107993–108003, 2022.

[35] Topi Maenpaa, Matti Pietikainen, and Jaakko Viertola. Separating color and pattern information for color texture discrimination. In *2002 International Conference on Pattern Recognition*, volume 1, pages 668–671. IEEE, 2002.

[36] O Nakyoung, Jiwon Choi, Daeyeong Kim, and Changick Kim. Supervised classification and segmentation of textured scene images. In *2015 IEEE International Conference on Consumer Electronics (ICCE)*, pages 473–476. IEEE, 2015.

[37] Peizhong Liu, Jing-Ming Guo, Kosin Chamnongthai, and Heri Prasetyo. Fusion of color histogram and lbp-based features for texture image retrieval and classification. *Information Sciences*, 390:95–111, 2017.

[38] Dana E Ilea and Paul F Whelan. Image segmentation based on the integration of colour–texture descriptors–A review. *Pattern Recognition*, 44(10):2479–2501, 2011.

[39] Matti Pietikainen, Sami Nieminen, Elzbieta Marszalec, and Timo Ojala. Accurate color discrimination with classification based on feature distributions. In *Proceedings of 13th international conference on pattern recognition*, volume 3, pages 833–838. IEEE, 1996.

[40] Christoph Palm and Thomas M Lehmann. Classification of color textures by gabor filtering. *Machine Graphics and Vision*, 11(2/3):195–220, 2002.

[41] Ricardo Dutra da Silva, Rodrigo Minetto, William Robson Schwartz, and Helio Pedrini. Satellite image segmentation using wavelet transforms based on color and texture features. In *International symposium on visual computing*, pages 113–122. Springer, 2008.

[42] Maria E Barilla and Michael Spann. Colour-based texture image classification using the complex wavelet transform. In *2008 5th International Conference on Electrical Engineering, Computing Science and Automatic Control*, pages 358–363. IEEE, 2008.

[43] Raquel Bello-Cerezo, Francesco Bianconi, Francesco Di Maria, Paolo Napoletano, and Fabrizio Smeraldi. Comparative evaluation of hand-crafted image descriptors vs. off-the-shelf cnn-based features for colour texture classification under ideal and realistic conditions. *Applied Sciences*, 9(4):738, 2019.

[44] Li Liu, Lingjun Zhao, Yunli Long, Gangyao Kuang, and Paul Fieguth. Extended local binary patterns for texture classification. *Image and Vision Computing*, 30(2):86–99, 2012.

[45] Uma G Maheswari, K Ramar, D Manimegalai, and V Gomathi. An adaptive region based color texture segmentation using fuzzified distance metric. *Applied soft computing*, 11(2):2916–2924, 2011.

[46] Belal Khaldi, Oussama Aiadi, and Mohammed Lamine Kherfi. Combining colour and grey-level co-occurrence matrix features: a comparative study. *IET Image Processing*, 13(9):1401–1410, 2019.

[47] Orlando J Hernandez, John Cook, Michael Griffin, Cynthia De Rama, and Michael McGovern. Classification of color textures with random field models and neural networks. *Journal of Computer Science & Technology*, 5, 2005.

[48] Dileep Kumar Panjwani and Glenn Healey. Markov random field models for unsupervised segmentation of textured color images. *IEEE Transactions on pattern analysis and machine intelligence*, 17(10):939–954, 1995.

[49] Amit Jain and Glenn Healey. A multiscale representation including opponent color features for texture recognition. *IEEE Transactions on Image Processing*, 7(1):124–128, 1998.

[50] Ayyasamy Vadivel, Shamik Sural, and Arun K Majumdar. An integrated color and intensity co-occurrence matrix. *Pattern Recognition Letters*, 28(8):974–983, 2007.

[51] Topi Mäenpää and Matti Pietikäinen. Texture analysis with local binary patterns. In *Handbook of pattern recognition and computer vision*, pages 197–216. World Scientific, 2005.

[52] Francesco Bianconi, Raquel Bello-Cerezo, Paolo Napoletano, and Francesco Di Maria. Improved opponent colour local binary patterns for colour texture classification. In *Computational Color Imaging: 6th International Workshop, CCIW 2017, Milan, Italy, March 29-31, 2017, Proceedings 6*, pages 272–281. Springer, 2017.

[53] Xin Shu, Zhigang Song, Jinlong Shi, Shucheng Huang, and Xiao-Jun Wu. Multiple channels local binary pattern for color texture representation and classification. *Signal processing: image communication*, 98:116392, 2021.

[54] Cuiyu Song, Peijun Li, and Fengjie Yang. Multivariate texture measured by local binary pattern for multispectral image classification. In *2006 IEEE International Symposium on Geoscience and Remote Sensing*, pages 2145–2148. IEEE, 2006.

[55] Alice Porebski, Nicolas Vandenbroucke, and Ludovic Macaire. Haralick feature extraction from lbp images for color texture classification. In *Proceeding of International*

Conference on Image Processing Theory, Tools and Applications, pages 1–8. IEEE, 2008.

[56] Michael Häfner, Michael Liedlgruber, Andreas Uhl, Andreas Vécsei, and Friedrich Wrba. Color treatment in endoscopic image classification using multi-scale local color vector patterns. *Medical image analysis*, 16(1):75–86, 2012.

[57] Audrey Ledoux, Olivier Losson, and Ludovic Macaire. Color local binary patterns: compact descriptors for texture classification. *Journal of Electronic Imaging*, 25(6):061404–061404, 2016.

[58] Rushi Lan, Huimin Lu, Yicong Zhou, Zhenbing Liu, and Xiaonan Luo. An lbp encoding scheme jointly using quaternionic representation and angular information. *Neural Computing and Applications*, 32:4317–4323, 2020.

[59] Tiecheng Song, Jie Feng, Shiyan Wang, and Yurui Xie. Spatially weighted order binary pattern for color texture classification. *Expert Systems with Applications*, 147:113167, 2020.

[60] Míriam Mengíbar-Rodríguez and Jesús Chamorro-Martínez. An image-based approach for building fuzzy color spaces. *Information Sciences*, 616:577–592, 2022.

[61] Paul Kay and Chad K McDaniel. The linguistic significance of the meanings of basic color terms. *Language*, 54(3):610–646, 1978.

[62] José M Soto-Hidalgo, Pedro Manuel Martinez-Jimenez, Jesús Chamorro-Martinez, and Daniel Sánchez. Jfcs: A color modeling java software based on fuzzy color spaces. *IEEE Computational Intelligence Magazine*, 11(2):16–28, 2016.

[63] Jesus Chamorro-Martínez, Jose Manuel Soto-Hidalgo, Pedro Manuel Martínez-Jiménez, and Daniel Sánchez. Fuzzy color spaces: A conceptual approach to color vision. *IEEE Transactions on Fuzzy Systems*, 25(5):1264–1280, 2016.

[64] Lotfi A Zadeh. A theory of approximate reasoning machine intelligence. In J. E. Hayes, D. Michie, and L. I. Mikulich, editors, *Machine Intelligence*, pages 149–194. Elsevier, Amsterdam, 1979.

[65] Konstantinos Konstantinidis, Antonios Gasteratos, and Ioannis Andreadis. Image retrieval based on fuzzy color histogram processing. *Optics Communications*, 248(4-6):375–386, 2005.

[66] Lotfi A Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

[67] James C Bezdek and Sankar Kumar Pal. Fuzzy models for pattern recognition. Technical report, USDOE Pittsburgh Energy Technology Center (PETC), PA (United States); Oregon ..., 1994.

[68] Pilar Sobrevilla and Eduard Montseny. Fuzzy sets in computer vision: an overview. *Mathware & Soft Computing*, 10(3):71–83, 2003.

[69] Aina Barcelo, Eduard Montseny, and Pilar Sobrevilla. Fuzzy texture unit and fuzzy texture spectrum for texture characterization. *Fuzzy Sets and Systems*, 158(3):239–252, 2007.

[70] Eystratios Keramidas, Dimitris Iakovidis, and Dimitris Maroulis. Fuzzy binary patterns for uncertainty-aware texture representation. *Electronic Letters on Computer Vision and Image Analysis*, 10(1):63–78, 2011.

[71] Dubois Didier and Prade Henri. *Fuzzy sets and systems: theory and applications*, volume 144 of *Mathematics in science and engineering*. Academic Press, New York, 1980.

[72] Jawahar CV and Ajoy K Ray. Incorporation of gray-level imprecision in representation and processing of digital images. *Pattern Recognition Letters*, 17(5):541–546, 1996.

[73] Yih-Gong Lee, Jia-Hong Lee, Yuang-Cheh Hsueh, et al. Fuzzy uncertainty texture spectrum for texture analysis. *Electronics Letters*, 31(12):959–960, 1995.

[74] Dimitris K Iakovidis, Eystratios G Keramidas, and Dimitris Maroulis. Fuzzy local binary patterns for ultrasound texture characterization. In *Image Analysis and Recognition: 5th International Conference, ICIAR 2008, Póvoa de Varzim, Portugal, June 25-27, 2008. Proceedings 5*, pages 750–759. Springer, 2008.

[75] Raissa Tavares Vieira, Carlos Eduardo de Oliveira Chierici, Carolina Toledo Ferraz, and Adilson Gonzaga. Local fuzzy pattern: A new way for micro-pattern analysis. In *Intelligent Data Engineering and Automated Learning-IDEAL 2012: 13th International Conference, Natal, Brazil, August 29-31, 2012. Proceedings 13*, pages 602–611. Springer, 2012.

[76] Yutthana Munklang, Sansanee Auephanwiriyakul, and Nipon Theera-Umpon. A novel fuzzy co-occurrence matrix for texture feature extraction. In *Procs. 13th International Conference on Computational Science and its Applications (ICCSA 2013)*, volume 7973 of *Lecture Notes in Computer Science*, pages 246–257. Springer Berlin Heidelberg, June 2013.

[77] Ju Han and Kai-Kuang Ma. Fuzzy color histogram and its use in color image retrieval. *IEEE Transactions on Image Processing*, 11(8):944–952, 2002.

[78] Fu-ping Yang and Mei-li Hao. Effective image retrieval using texture elements and color fuzzy correlogram. *Information*, 8(1):27, 2017.

[79] Joao B Florindo and Estevao Esmi Laureano. Boff: A bag of fuzzy deep features for texture recognition. *Expert Systems with Applications*, 219:119627, 2023.

[80] Belal Khaldi and Mohammed Lamine Kherfi. Modified integrative color intensity co-occurrence matrix for texture image representation. *Journal of Electronic Imaging*, 25(5):053007, 2016.

[81] Mirvana Hilal, Andreia S Gaudêncio, Pedro G Vaz, João Cardoso, and Anne Humeau-Heurtier. Colored texture analysis fuzzy entropy methods with a dermoscopic application. *Entropy*, 24(6):831, 2022.

[82] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[83] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

[84] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[85] Poornima Raikar and SM Joshi. Efficiency comparison of supervised and unsupervised classifier on content based classification using shape, color, texture. In *2020 International Conference for Emerging Technology (INCET)*, pages 1–7. IEEE, 2020.

[86] Azzam Sleit, Abdel Llatif Abu Dalhoum, Mohammad Qatawneh, Maryam Al-Sharief, Rawa' Al-Jabaly, and Ola Karajeh. Image clustering using color, texture and shape features. *KSII Transactions on Internet and Information Systems (TIIS)*, 5(1):211–227, 2011.

[87] Lorenzo Bruzzone, Mingmin Chi, and Mattia Marconcini. A novel transductive svm for semisupervised classification of remote-sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 44(11):3363–3373, 2006.

[88] Yann LeCun, Yoshua Bengio, et al. The handbook of brain theory and neural networks, 1998.

[89] Philippe Bolon, Jean-Marc Chassery, Jean-Pierre Cocquerez, Didier Demigny, Christine Graffigne, Annick Montanvert, Sylvie Philipp, Rachid Zéboudj, Josiane Zerubia, and Henri Maître. Analyse d'images: filtrage et segmentation, 1995.

[90] Junqing Chen, Thrasyvoulos N Pappas, Aleksandra Mojsilovic, and Bernice E Rogowitz. Adaptive perceptual color-texture image segmentation. *IEEE transactions on image processing*, 14(10):1524–1536, 2005.

[91] Irene Fondón, Carmen Serrano, and Begoña Acha. Color-texture image segmentation based on multistep region growing. *Optical engineering*, 45(5):057002–057002, 2006.

[92] Patrick C Chen and Theodosios Pavlidis. Segmentation by texture using a co-occurrence matrix and a split-and-merge algorithm. *Computer graphics and image processing*, 10(2):172–182, 1979.

[93] Mark F Doherty, Carolyn M Bjorklund, and Mark T Noga. Split-merge-merge: An enhanced segmentation capability. In *IEEE Conférence on Computer Vision and Pattern Recognition, Miami Beach, Florida (USA)*, pages 325–330, 1986.

[94] Kan-Min Chen and Shu-Yuan Chen. Color texture segmentation using feature distributions. *Pattern Recognition Letters*, 23(7):755–771, 2002.

[95] Xiangyun Hu, C Vincent Tao, and Björn Prenzel. Automatic segmentation of high-resolution satellite imagery by integrating texture, intensity, and color features. *Photogrammetric Engineering & Remote Sensing*, 71(12):1399–1406, 2005.

[96] Raffaele Gaetano, Giuseppe Scarpa, and Giovanni Poggi. Recursive texture fragmentation and reconstruction segmentation algorithm applied to vhr images. In *2009 IEEE International Geoscience and Remote Sensing Symposium*, volume 4, pages IV–101. IEEE, 2009.

[97] Giuseppe Scarpa, Giuseppe Masi, Raffaele Gaetano, Luisa Verdoliva, and Giovanni Poggi. Dynamic hierarchical segmentation of remote sensing images. In *Procs. International Conference on Image Analysis and Processing (ICIAP 2013)*, volume 8156 of *Lecture Notes in Computer Science*, pages 371–380. Springer Berlin Heidelberg, 2013.

[98] Zoltan Kato, Ting-Chuen Pong, and John Chung-Mong Lee. Color image segmentation and parameter estimation in a Markovian framework. *Pattern Recognition Letters*, 22(3-4):309–321, 2001.

[99] Jiří Borovec, Jan Švihlík, Jan Kybic, and David Habart. Supervised and unsupervised segmentation using superpixels, model estimation, and graph cut. *Journal of Electronic Imaging*, 26(6):061610–061610, 2017.

[100] Abderezak Salmi, Kamal Hammouche, and Ludovic Macaire. Constrained feature selection for semisupervised color-texture image segmentation using spectral clustering. *Journal of Electronic Imaging*, 30(1):013014–013014, 2021.

[101] Chunlong Zhang, Kunlin Zou, and Yue Pan. A method of apple image segmentation based on color-texture fusion feature and machine learning. *Agronomy*, 10(7):972, 2020.

[102] S Jenicka and S Jenicka. Supervised texture-based segmentation using basic texture models. *Land Cover Classification of Remotely Sensed Images: A Textural Approach*, pages 73–88, 2021.

[103] Omar S Al-Kadi. Supervised texture segmentation: a comparative study. In *2011 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pages 1–5. IEEE, 2011.

[104] Hong-Ying Yang, Xiang-Yang Wang, Qin-Yan Wang, and Xian-Jin Zhang. Ls-svm based image segmentation using color and texture information. *Journal of Visual Communication and Image Representation*, 23(7):1095–1112, 2012.

[105] Victor RS Laboreiro, Thelmo P de Araujo, and Jose Everardo Bessa Maia. A texture analysis approach to supervised face segmentation. In *2014 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2014.

[106] Jiangye Yuan, Deliang Wang, and Anil M Cheriyadat. Factorization-based texture segmentation. *IEEE Transactions on Image Processing*, 24(11):3488–3497, 2015.

[107] Michal Haindl and Stanislav Mikeš. A competition in unsupervised color image segmentation. *Pattern Recognition*, 57:136–151, 2016.

[108] Yong Yang, Shoudong Han, Tianjiang Wang, Wenbing Tao, and Xue-Cheng Tai. Multilayer graph cuts based unsupervised color–texture image segmentation using multivariate mixed student's t-distribution and regional credibility merging. *Pattern recognition*, 46(4):1101–1124, 2013.

[109] Yaman Akbulut, Yanhui Guo, Abdulkadir Şengür, and Muzaffer Aslan. An effective color texture image segmentation algorithm based on hermite transform. *Applied Soft Computing*, 67:494–504, 2018.

[110] Shuyuan Yang, Yuan Lv, Yu Ren, Lixia Yang, and Licheng Jiao. Unsupervised images segmentation via incremental dictionary learning based sparse representation. *Information Sciences*, 269:48–59, 2014.

[111] Martin Baatz. Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation. *Angewandte geographische informationsverarbeitung*, pages 12–23, 2000.

[112] Gholamreza Akbarizadeh and Masoumeh Rahmani. Efficient combination of texture and color features in a new spectral clustering method for polsar image segmentation. *National Academy Science Letters*, 40:117–120, 2017.

[113] Yuhui Quan, Huan Teng, Tao Liu, and Yan Huang. Weakly-supervised sparse coding with geometric prior for interactive texture segmentation. *IEEE Signal Processing Letters*, 27:116–120, 2019.

[114] Priyambada Subudhi and Susanta Mukhopadhyay. An efficient graph reduction framework for interactive texture segmentation. *Signal Processing: Image Communication*, 74:42–53, 2019.

[115] Arnav V Bhavsar. An efficient weakly supervised approach for texture segmentation via graph cuts. *Journal of Intelligent Systems*, 22(3):253–267, 2013.

[116] Vincent Andrearczyk and Paul F Whelan. Texture segmentation with fully convolutional networks. *arXiv preprint arXiv:1703.05230*, 2017.

[117] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Pablo Martinez-Gonzalez, and Jose Garcia-Rodriguez. A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41–65, 2018.

[118] Yuan Huang, Fugen Zhou, and Jérôme Gilles. Empirical curvelet based fully convolutional network for supervised texture image segmentation. *Neurocomputing*, 349:31–43, 2019.

[119] Wenguan Wang and Jianbing Shen. Deep visual attention prediction. *IEEE Transactions on Image Processing*, 27(5):2368–2378, 2017.

[120] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Procs. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, pages 6230–6239, Honolulu, HI, USA, July 2017.

[121] Jean Serra. *Image analysis and mathematical morphology*. Academic Press, Inc., 1983.

[122] Xuejie Qin and Herb Yang. Representing texture images using asymmetric gray level aura matrices. 2005.

[123] Xuejie Qin and Yee-Hong Yang. Similarity measure and learning with gray level aura matrices (GLAM) for texture image retrieval. In *Procs. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages I–326–I–333, Washington, DC, USA, June 2004. IEEE Computer Society.

[124] Xuejie Qin and Yee-Hong Yang. Texture image classification using basic gray level aura matrices. *Technical Report TR06-14*, 2006.

[125] Zohra Haliche. Classification et segmentation d'images texturées basée sur la théorie des ensembles. *Thèse Magistère*, 2010.

[126] Olivier D Faugeras and William K Pratt. Decorrelation methods of texture feature extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (4):323–332, 1980.

[127] Rama Chellappa. Two-dimensional discrete gaussian markov random field models for image processing. *IETE Journal of Research*, 35(2):114–120, 1989.

[128] George R Cross and Anil K Jain. Markov random field texture models. *IEEE Transactions on pattern analysis and machine intelligence*, (1):25–39, 1983.

[129] Dalila Benboudjema. Champs de markov triplets et segmentation bayesienne non supervisée d'images. *Optimisation et sûreté des systèmes, Université de Technologie de Troyes-Institut National des Télécommunications*, 2005.

[130] Rosalind W Picard, Ibrahim M Elfadel, and Alexander Pentland. Markov/gibbs texture modeling: aura matrices and temperature effects. In *CVPR*, volume 91, pages 371–377, 1991.

[131] Fa-Yueh Wu. The potts model. *Reviews of modern physics*, 54(1):235, 1982.

[132] Xuejie Qin and Yee-Hong Yang. Aura 3D textures. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):379–389, 2007.

[133] Zohra Haliche and Kamal Hammouche. The gray level aura matrices for textured image segmentation. *Analog Integrated Circuits and Signal Processing*, 69(1):29–38, 2011.

[134] Topi Mäenpää and Matti Pietikäinen. Classification with color and texture: jointly or separately? *Pattern Recognition*, 37:1629–1640, 2004.

[135] Johan Debayle and Jean-Charles Pinoli. Spatially adaptive morphological image filtering using intrinsic structuring elements. *Image Analysis and Stereology*, 24(3):145–158, 2005.

[136] Vladimir Ćurić, Anders Landström, Matthew J Thurley, and Cris L Luengo Hendriks. Adaptive mathematical morphology–a survey of the field. *Pattern Recognition Letters*, 47:18–28, 2014.

[137] Isabelle Bloch and Henri Maître. Fuzzy mathematical morphologies: A comparative study. *Pattern Recognition*, 28(9):1341–1387, 1995.

[138] Debashis Sen and Sankar K. Pal. Image segmentation using global and local fuzzy statistics. In *Procs. IEEE India Council International Conference (INDICON 2006)*, pages 1–6, New Delhi, India, September 2006.

[139] Alain Tremeau and Bernard Laget. Color quantization and image analysis. quantification couleur et analyse d'image color quantization and image analysis quantification couleur et analyse d'image.

[140] James C. Bezdek. Pattern recognition with fuzzy objective function algorithms. In *Advanced Applications in Pattern Recognition*, 1981.

[141] Vinh Truong Hoang. Multi color space lbp-based feature selection for texture classification. Littoral, 2018.

[142] Chandan Singh, Ekta Walia, and Kanwal Preet Kaur. Color texture description with novel local binary patterns for effective image retrieval. *Pattern recognition*, 76:50–68, 2018.

[143] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[144] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[145] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[146] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[147] Andrea Vedaldi and Stefano Soatto. Quick shift and kernel methods for mode seeking. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10*, pages 705–718. Springer, 2008.

[148] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[149] Rémi Giraud, Vinh-Thong Ta, and Nicolas Papadakis. Robust superpixels using color and contour features along linear path. *Computer Vision and Image Understanding*, 170:1–13, 2018.

[150] Alex Levinshtein, Adrian Stere, Kiriakos N Kutulakos, David J Fleet, Sven J Dickinson, and Kaleem Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2290–2297, 2009.

[151] Rémi Giraud and Yannick Berthoumieu. Texture superpixel clustering from patch-based nearest neighbor matching. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2019.

[152] Rémi Giraud, Vinh-Thong Ta, Nicolas Papadakis, and Yannick Berthoumieu. Texture-aware superpixel segmentation. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1465–1469. IEEE, 2019.

[153] Zohra Haliche, Kamal Hammouche, Olivier Losson, and Ludovic Macaire. Fuzzy color aura matrices for texture image segmentation. *Journal of Imaging*, 8(9):244, 2022.

[154] Xiaohui Chen, Chen Zheng, Hongtai Yao, and Bingxue Wang. Image segmentation using a unified Markov random field model. *IET Image Processing*, 11(10):860–869, 2017.

[155] Michal Haindl and Stanislav Mikeš. Unsupervised texture segmentation. *Pattern Recognition Techniques, Technology and Applications*, pages 227–248, 2008.

[156] Ming-Yu Liu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. Entropy rate superpixel segmentation. In *Procs. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)*, pages 2097–2104, 2011.

[157] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, 2018.

[158] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[159] Martin Riedmiller. Advanced supervised learning in multi-layer perceptrons—from backpropagation to adaptive learning algorithms. *Computer Standards & Interfaces*, 16(3):265–278, 1994.

[160] Stanislav Mikes and Michal Haindl. Texture segmentation benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (in press):1–1, 2021.

[161] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Procs. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)*, pages 234–241, Munich, Germany, October 2015.

[162] *Prague texture segmentation data generator and benchmark* available from: http://mosaic.utia.cas.cz, (accessed February 24, 2021).

[163] Stanislav Mikeš, Michal Haindl, Giuseppe Scarpa, and Raffaele Gaetano. Benchmarking of remote sensing segmentation methods. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(5):2240–2248, 2015.

[164] Essa Basaeed, Harish Bhaskar, and Mohammed Al-Mualla. Supervised remote sensing image segmentation using boosted convolutional neural networks. *Knowledge-Based Systems*, 99:19–27, 2016.

[165] Adam Hoover, Gillian Jean-Baptiste, Xiaoyi Jiang, Patrick J Flynn, Horst Bunke, Dmitry B Goldgof, Kevin Bowyer, David W Eggert, Andrew Fitzgibbon, and Robert B Fisher. An experimental comparison of range image segmentation algorithms. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):673–689, 1996.

[166] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[167] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

[168] Stijn Dongen. *Performance criteria for graph clustering and Markov cluster experiments*. CWI (Centre for Mathematics and Computer Science), 2000.

Matrices aura des couleurs floues : Classification et segmentation d'images couleur texturées.

Résumé

La classification et la segmentation d'images couleur texturées sont des opérations très utilisées dans divers domaines d'application de la vision par ordinateur. Elles reposent essentiellement sur l'extraction d'attributs qui a pour but de caractériser globalement une image (classification) ou localement un pixel (segmentation) par un ensemble d'attributs. Parmi la multitude de techniques d'extraction d'attributs, on s'est intéressé dans cette thèse aux matrices aura. Ces matrices, initialement proposées pour des images en niveaux de gris (GLAM), ont été étendues aux images couleur suivant une approche marginale (MCLAM), qui ne prend pas en compte les interactions entre les composantes couleur. Pour tenir compte de la corrélation entre ces composantes, nous proposons dans cette thèse d'appliquer la stratégie opposée qui consiste à calculer les matrices aura inter-composantes (OCLAM). Toutefois, ces deux stratégies présentent des défis en termes de mémoire et de temps de calcul. Pour surmonter ces inconvénients, nous proposons d'appliquer la stratégie compacte pour définir une seule matrice aura couleur (CAM) contenant les informations vectorielles de la couleur.

D'autre part, afin de tenir compte des imprécisions et ambiguïtés présentes dans les images de textures couleur, nous avons étendu les matrices aura couleur au cadre flou. Ces extensions ont été également réalisées selon trois stratégies : marginale (MFCLAM), opposée (OFCLAM) et compacte (FCAM). Les résultats de la classification ont montré l'intérêt du flou et le pouvoir discriminatif des OFCAMs et FCAMs par rapport aux autres attributs de texture couleur. Nous avons par ailleurs proposé une nouvelle méthode de segmentation d'images de textures couleur, dans laquelle les FCAMs cardinales sont exploitées pour caractériser les superpixels de l'image à segmenter, obtenus par l'intermédiaire d'un algorithme SLIC régional que nous avons également développé dans le cadre de cette thèse. Des tests menés sur deux bases d'images Prague et ALI ont montré l'efficacité de la méthode de segmentation proposée par rapport aux autres méthodes de segmentation classiques de l'état de l'art et sa compétitivité avec les méthodes récentes basées sur l'apprentissage profond.

Mots clés — Textures couleur; Textures couleur floues; Matrices aura couleur; Matrices aura couleur floue; Segmentation d'images couleur texturées; Classification des textures couleur.

Fuzzy Color Aura Matrices: Classification and Segmentation of Textured Color Images.

Abstract

Classification and segmentation of textured color images are widely used operations in various computer vision application domains. They primarily rely on feature extraction, which aims to globally characterize an image (classification) or locally characterize a pixel (segmentation) through a set of attributes. Among the numerous feature extraction techniques, this thesis focuses on aura matrices. Initially proposed for grayscale images (GLAM), these matrices were extended to color images following a marginal approach (MCLAM), which does not account for interactions between color components. To address the correlation between these components, we propose in this thesis to apply the opposite strategy, which involves calculating inter-component aura matrices (OCLAM). However, both strategies face challenges in terms of memory and computation time. To overcome these limitations, we propose applying a compact strategy to define a single color aura matrix (CAM) that contains the vectorial information of the color.

Moreover, to address the imprecision and ambiguities present in textured color images, we have extended color aura matrices to the fuzzy framework. These extensions have also been realized following the three strategies: marginal (MFCLAM), opposite (OFCLAM), and compact (FCAM). Classification results demonstrated the relevance of the fuzzy approach and the discriminative power of OFCAMs and FCAMs compared to other color texture attributes. Additionally, we proposed a novel method for segmenting textured color images, in which cardinal FCAMs are used to characterize the superpixels of the image to be segmented. These superpixels are obtained using a regional SLIC algorithm, which we also developed as part of this thesis. Tests conducted on two image datasets, Prague and ALI, demonstrated the effectiveness of the proposed segmentation method compared to other classic state-of-the-art segmentation methods and its competitiveness with recent deep learning-based methods.

Keywords — Color textures; Fuzzy color textures; Color aura matrices; Fuzzy color aura matrices; Color texture segmentation; Textures color classification.