Habilitation à diriger des recherches

présentée à

L'université de Reims

Ecole doctorale de Sciences Exactes et Biologique

par

Luc Brun

Traitement d'images couleur et Pyramides combinatoires

PARTIE I : mémoire scientifique

Soutenue le 13/12/2002 devant la Commission d'Examen composée de :

W. Kropatsch	Président	Professeur, Université Technique de Vienne
P. Lienhardt	Rapporteur	Professeur, Université de Poitiers
C. Fernandez	Rapporteur	Professeur, Université de Poitiers
A. Braquelaire	Rapporteur	Professeur, Université de Bordeaux I
N. Bonnet	Membre du Jury	Professeur, Université de Reims
A. Trémeau	Membre du Jury	Professeur, Université de Saint Étienne

— 2002 —

Remerciements

Je tiens à remercier Walter Kropatsch pour avoir accepté de participer à mon jury et pour l'enthousiasme qu'il a apporté tout au long de notre collaboration sur les pyramides de cartes combinatoires.

Je remercie également Alain Trémeau qui a accepté de participer à mon jury.

Je remercie également Noël Bonnet qui a accepté de faire partie de mon jury et de lire une dernière fois ce manuscrit.

Je tiens également à remercier Pascal Lienhardt, Achille Braquelaire et Christine Fernandez qui ont accepté de rapporter ce long mémoire.

Je remercie Éric Andres qui m'a gentiment envoyé sa thèse d'habilitation afin que je puisse me faire une idée de ce qu'il fallait obtenir. De nombreux points concernant la forme de ce manuscrit sont d'inspiration Andrésienne.

Je tiens également à remercier ma femme, Myriam, qui a prêté son image pour ce manuscrit (page 104) et a bien voulu dégager quelques heures de son emplois du temps pour corriger les multiples fautes d'orthographe. Je profite de ces pages pour m'excuser de lui avoir fait passer tout un mois d'août avec un mari coincé dans la cave avec un ordinateur.

Je remercie enfin mon fils Loïs qui a également prêté son image pour le manuscrit (pages 13, et 104) et qui a bien voulu comprendre qu'il ne faut pas appuyer sur le gros bouton bleu (On/Off) quand papa travaille sur l'ordinateur.

Table des matières

Chapitre 1

Introduction

Dans le cadre de mon intervention dans le module Vision du D.E.A. OSS (université de technologie de Troyes), j'utilise régulièrement la figure 1.1 pour représenter l'ensemble des phénomènes physiques et des traitements intervenant en traitement d'images[1]. Un agent représente ici un système utilisant l'informatique. Il peut donc s'agir d'un ordinateur couplé à un bras robotisé ou d'un individu utilisant des données informatiques. Cet agent doit exercer une influence sur le monde dans lequel il intervient. Dans le cadre d'un contrôle industriel, cette influence se traduit typiquement par un rejet de pièces défectueuses. Le monde dans lequel évolue l'agent est composé d'objets éclairés par des sources lumineuses. La composition de la lumière est modifiée lors de sa réflexion sur un objet. Ce phénomène crée l'aspect coloré des objets, et la modification de la lumière incidente par chaque objet est représentée par une fonction de réflectance. La lumière réfléchie impressionne les capteurs d'une caméra ce qui crée une image numérique.

Le but du traitement d'image consiste alors à calculer des attributs sur les valeurs de l'image de façon à pouvoir regrouper des ensembles connexes de pixels. Un tel processus est appelé un processus de segmentation et les regroupements obtenus des régions. Chaque objet de la scène est usuellement décomposé en une union de plusieurs régions. Une étape d'analyse de l'image possédant des informations *a priori* sur les objets de la scène permet de regrouper les régions en objets. La reconnaissance des objets de la scène permet d'obtenir un modèle de celle-ci. Ce modèle est alors utilisé par l'agent pour agir sur le monde extérieur.

Une des originalité de ce mémoire tient sans doute au fait que l'ensemble des modèles et des méthodes décrits dans celui-ci permet de couvrir l'ensemble des points défini précédemment. Si nous nous référons à la figure 1.1, les différents thèmes que nous allons aborder couvrent en effet :

- les modèles de réflexion (Chapitre 2). Ces modèles permettent de décrire la modification d'un rayon lumineux lors d'une réflexion sur un objet de la scène (Fig. 1.1, étapes (3) et (4)). Ces modèles seront utilisés pour segmenter une image et caractériser certain objets de celle-ci (étapes (6) à (8)). Cette caractérisation nous fournira un modèle de la scène dans un cadre simplifié (étape (9)) ;
- nous étudierons également les méthodes de quantification qui permettent de calculer des attributs communs à plusieurs pixels de la scène (Chapitre 3, étape (5)). Ces méthodes ne constituent pas des méthodes de segmentation mais peuvent être utilisés comme pré-

[1] Cette figure a été initialement conçue par Walter Kropatsch de l'Université Technique de Vienne (Autriche)

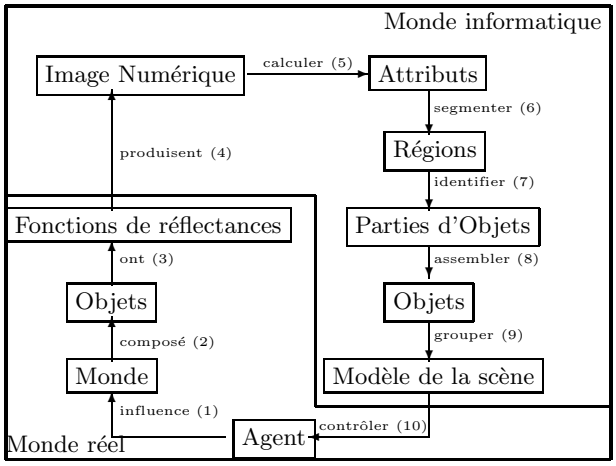


FIG. 1.1 – Cycle des phénomènes et traitements intervenant en traitement d'images.

traitement par celles-ci. Nous étudierons également les méthodes d'inversion de table de couleur qui suivent généralement les méthodes de quantification dans le cadre de l'affichage d'image ;

– nous étudierons enfin les représentations hiérarchiques (Chapitre 4) qui permettent de structurer l'ensemble des régions d'une partition. Ces modèles peuvent être utilisés dans le cadre de la segmentation ou pour regrouper plusieurs régions issues d'une segmentation afin de former des objets (étapes (6) à (8)).

Les différentes domaines abordés dans ce mémoire sont décrits plus précisément dans la section suivante. Mon apport spécifique à chacun de ces domaines est décrit en section 1.2. Le plan de ce mémoire est quand à lui décrit en section 1.3..

1.1 Domaines abordés

Les modèles de réflexion dépendent de paramètres géométriques et de paramètres optiques. Toutefois, pour une source lumineuse et un objet composé d'un seul matériau, les modèles de réflexions peuvent se concevoir comme des formules dépendant uniquement de paramètres géométriques et de constantes à déterminer. La détermination de ces constantes (section 2.5) sert un double objectif : ces constantes étant fonction des propriétés optiques du matériau, l'état de celui-ci peut être caractérisé par la valeur des constantes. De plus, la détermination de ces constantes permet d'extraire les paramètres géométriques afin de retrouver la forme $3D$ de l'objet.

La connaissance de ces paramètres ainsi que celle des intensités permet alors de retrouver l'altitude de chacun des points de la scène (section 2.6). Cette inversion du modèle (retrouver les paramètres géométriques à partir des intensités) est effectuée soit à partir de techniques de minimisations soit en accumulant un nombre suffisant de données pour chaque pixel.

J'ai également mené des recherches en quantification d'images couleur (section 3.2). Intuitivement, une méthode de quantification prend en paramètre une image couleur et un entier K et fournit les K couleurs les plus représentatives de l'image. Ces méthodes ont des applications

évidentes dans l'affichage d'images avec une nombre réduit de couleurs. Toutefois, définir les K couleurs les plus représentatives d'une image est souvent équivalent à décomposer l'espace couleur en une partition en K ensembles homogènes. L'homogénéité d'un ensemble de couleur est mesurée par son erreur quadratique qui représente la somme des carrés des distances de chaque couleur de l'ensemble à la couleur moyenne de celui-ci. L'homogénéité d'une partition en K ensembles est alors mesurée par l'erreur de partition définie comme la somme des erreurs quadratiques des ensembles formant la partition. De ce point de vue, les méthodes de quantification peuvent se concevoir comme des méthodes de classification particulières.

Étant donné K couleurs, l'affectation de chacune des couleurs de l'image à une de ces K couleurs est effectuée par une étape appelée l'inversion de table de couleur (section 3.3). L'ensemble des K couleurs initiales est appelé l'ensemble des couleurs représentatives et peut être généré par une méthode de quantification ou imposé par la table de couleurs par défaut de l'écran. J'ai également étendu mes activités de recherche à cette dernière étape.

Toutefois, comme l'a souligné Alain Trémeau [187], une image ne peut être ramenée à l'ensemble de ses couleurs. Nous avons en effet à traiter des données $3 \times 2D$ et l'arrangement spatial des couleurs ne peut être ignoré. J'ai donc travaillé en collaboration avec Walter Kropatsch de l'Université technologique de Vienne sur le codage de partitions. Ma motivation initiale pour cette activité de recherche était basée sur ce qui constitue pour moi une limitation des méthodes de segmentation usuelles. Supposons en effet que l'on ait obtenu une segmentation de l'image en regions homogènes et que l'on dispose d'une connaissance *a priori* des objets à détecter. L'étape suivante de l'algorithme de segmentation consistera à guider les fusions de régions de façon à obtenir une région pour chaque objet de la scène. Toutefois, le niveau de détail de la partition finale reste souvent délicat à définir. Par exemple, si l'on veut segmenter une série de livres posés à plat sur une table, veux t'on obtenir le contour extérieur des livres ou désire t'on également une segmentation des titres et illustrations présents sur la couverture. La réponse à une telle question dépend de l'application mais est souvent ambiguë. Idéalement, on souhaiterait avoir les deux niveaux de représentations. Un tel codage est possible à l'aide des pyramides de graphes qui permettent de coder une hiérarchie de partitions avec des relations père-fils entre deux partitions. On peut ainsi coder avec un seul formalisme les deux niveaux de détails d'une partition. Notons que les pyramides présentent de nombreuses autres propriétés extrêmement intéressantes en vision et en modélisation. Le codage des niveaux de détails d'une partition permet en effet d'aider les algorithmes de segmentation. On a donc un échange entre les critères qui définissent les partitions et le modèle qui permet de coder celles-ci. Le codage des niveaux détails est également intéressant dans le cadre de la modélisation en dehors de toute préoccupation de segmentation. Il est en effet assez fréquent de rencontrer des applications où l'on dispose d'une quantité extrêmement importante de données et où l'on voudrait simplifier la modélisation afin d'appliquer certains traitements. Les modélisations de terrains et de couches géologiques constituent des exemples de modélisation ou de tels besoins se font sentir.

Historiquement, les premières pyramides ont été définies comme une pile d'images de taille décroissante. De telles pyramides sont appelées des pyramides régulières (section 4.2) et sont construites en affectant un ensemble connexe de pixel d'une image à un pixel de l'image réduite de niveau supérieur. Cette affectation entre les pixels de deux niveaux consécutifs définit une relation père-fils et l'ensemble des fils d'un pixel dans l'image précédente est appelé sa fenêtre de réduction. Si l'on itère cette relation père-fils en partant d'un pixel de la pyramide jusqu'au niveau de base, l'ensemble des descendants du pixel dans le niveau de base est appelé son champ récepteur. Les pyramides régulières présentent de nombreux avantages dans le cadre du traitement d'image mais ce sont révélées peu adaptées au codage de partitions. En effet, de telles pyramides ne peuvent

par exemple coder des régions allongées (section 4.2). De plus, les relations d'adjacences entre les régions sont également difficiles à établir. Meer, Montanvert et Jolion [142, 146, 114] ont levé ces limitations en définissant les pyramides comme des piles de graphes simples successivement réduits (section 4.3). L'association avec l'image est réalisée en associant le graphe de base à une grille discrète. Ce type de pyramides permet de coder correctement tout type de régions. Toutefois, l'utilisation de graphes simples impose de n'avoir qu'une seule arête entre deux sommets. On ne peut donc coder les frontières multiples entre les régions avec ce type de pyramides. De plus, l'on est incapable de distinguer à partir du graphe une relation d'adjacence d'une relation d'inclusion entre deux régions. Intuitivement, résoudre ce problème impose d'autoriser les arêtes multiples entre sommets. Toutefois, il faut à ce moment là être capable de distinguer des arêtes codant une information d'adjacence des arêtes redondantes. Ces limitations des graphes simples ont été levées par Walter Kropatsch qui stocke dans la pyramide un graphe codant l'adjacence des régions et son dual représentant les contours (section 4.5). Aucune contrainte n'impose aux deux graphes d'être simples. Le stockage du graphe dual permet de distinguer les arêtes redondantes des arêtes significatives.

1.2 Mes Contributions

1.2.1 Segmentation et reconstruction d'objets métalliques

Mes activités de recherche sur la segmentation et la reconstruction d'objets métalliques ont été effectuées dans le cadre de l'encadrement de la thèse de doctorat de Laurent Hussenet [109]. Cette thèse, financée par une bourse CIFRE, s'est déroulée en collaboration avec l'entreprise Axon Cable[2] qui usine de nombreuses pièces métalliques.

Les échanges avec cette entreprise ont été nombreux et enrichissants. Les visites que nous avons effectuées sur le site de production nous ont permis de comprendre les contraintes techniques de l'entreprise et les différents problèmes de contrôle de qualité qui se posent à elle. La première application que nous avons développée pour l'entreprise a porté sur la détection de défauts sur des câbles plats. La principale difficulté de cette application résidait dans le temps de traitement puisque le câble devait défiler devant la caméra à plusieurs mètres/secondes. Cette tâche ne requérant pas une expertise en traitement d'images suffisante pour être intégrée à la thèse, la conception de l'application a été effectuée dans le cadre d'un stage d'IUT encadré par Laurent Hussenet sous ma supervision. De son côté, l'entreprise a financé l'achat d'une caméra CMOS permettant des temps d'acquisition suffisamment courts pour ce type de traitement. L'application a été livrée à l'entreprise et devrait prochainement faire l'objet de tests sur le site de production.

La thèse de doctorat de Laurent Hussenet a donc porté sur d'autres besoins de l'entreprise. Les besoins initiaux dont celle-ci nous a fait part portaient sur le contrôle de brasures. L'entreprise utilise des brasures notamment pour fixer des câbles électriques à des connecteurs informatiques (Fig. 1.2). Dans ce cas, l'aspect volumique du dépôt d'étain est un paramètre important dans le contrôle de la qualité de la brasure. Par la suite, les différents contacts que nous avons eus avec l'entreprise nous ont amenés à élargir notre champ d'étude afin de pouvoir contrôler l'aspect volumique de nombreux objets métalliques également produits par l'entreprise.

Nous avons commencé par étudier les modèles de réflexion permettant de décrire l'intensité ou la couleur d'un pixel en fonction de paramètres géométriques et optiques tels que la normale du

[2]Site web : http ://www.axon-cable.fr

(a) (b) (c)

FIG. 1.2 – Brasures sur un connecteur ((a) et (b)) ainsi qu'une vue agrandie de deux fûts d'un connecteur (c)

point de réflexion, les directions de la lumière incidente et d'observation ainsi que la composition spectrale de la source lumineuse et les propriétés optiques du matériau.

Nous avons ensuite utilisé ces modèles dans le cadre de la segmentation en matériaux en nous basant sur des modèles spécifiquement conçus pour des objets métalliques. En effet, la plupart des méthodes de segmentation en matériaux utilisent les modèles de Shafer ou Healey (sections 2.2.5 et 2.4.1) qui ne fournissent qu'une description qualitative du phénomène de réflexion. Les couleurs d'une image réelle s'écartant toujours légèrement des valeurs prédites par le modèle, il est souvent délicat de faire la part entre le bruit d'acquisition et les limitations du modèle. L'utilisation de modèles physiques quantitatifs nous a permis de mieux contrôler les approximations faites par les modèles usuels. Nous avons également utilisé des méthodes de classification supervisée afin de tenir compte de la faible diversité des matériaux intervenant dans les images que nous avions à contrôler.

Notre contribution à l'estimation de paramètres se place dans le cadre des techniques utilisant plusieurs sources lumineuses (ou illuminants) et une caméra fixe. Les méthodes usuelles utilisent entre 8 et 60 sources lumineuses pour estimer les constantes en chaque pixel. Ce type de méthode réclame une procédure d'acquisition trop complexe et des temps de calculs trop élevés pour les applications envisagées par la société Axon Cable. En supposant que l'image est préalablement segmentée en matériaux, nous avons réduit le nombre d'illuminants nécessaires à l'estimation à 4. L'idée de base de notre méthode est de remplacer un nombre important de données pour chaque pixel par une mise en relation des données de plusieurs pixels.

Les constantes du modèle de réflexion étant données, nous avons conçu une méthode de reconstruction basée sur deux sources lumineuses qui permet de retrouver les normales en chaque point de la surface avec uniquement une ambiguïté sur le signe de l'une des composantes de la normale. Cette faible indétermination du résultat nous permet d'éviter le recours à des techniques de minimisation souvent coûteuses.

1.2.2 Traitement d'images couleurs

Mes activités de recherches en traitement d'images couleur ont débuté durant ma thèse de doctorat où j'ai étudié les algorithmes de quantification et plus particulièrement les algorithmes de quantification basés sur une approche descendante. Ma principale contribution à ce domaine de recherche a été une meilleure description de l'évolution de l'erreur de partition lors du déplacement

d'un plan destiné à découper un ensemble de couleurs en 2 sous ensembles. J'ai également effectué en collaboration avec Alain Trémeau un état de l'art assez complet des méthodes de quantification dans le cadre de la rédaction du chapitre d'un livre dédié au traitement d'images couleur [43]. Ceci m'a permis de faire les constations suivantes :

- un nouveau courant qui optimise simultanément la sélection et le placement des couleurs représentatives prend de plus en plus d'importance et constitue une approche très prometteuse dans le cadre de l'affichage d'images. Toutefois, de telles méthodes, que nous appellerons méthodes de quantification spatiales, ne peuvent plus être interprétées comme des méthodes de classification particulières ;
- les méthodes basées uniquement sur la sélection des couleurs représentatives peuvent être classifiées en trois grandes familles correspondant aux méthodes descendantes, ascendantes et mixtes. Cette décomposition correspond aux méthodes de découpe, fusion et découpe-fusion rencontrée en segmentation. Les méthodes ascendantes et descendantes ont été fortement développées ces vingt dernières années alors que curieusement les méthodes mixtes semblent assez peu explorées.

La méthode de quantification mixte que j'ai développée permet pourtant d'obtenir, pour de faibles valeurs du nombre K de couleurs représentatives, de meilleurs résultats que les méthodes descendantes avec des temps de calculs généralement 10 fois inférieurs. De plus, l'étude des performances de cette méthode montre que l'approche par quantification mixte possède encore un fort potentiel.

Enfin, je me suis également intéressé aux méthodes d'inversion de tables de couleurs. La principale caractéristique de la méthode proposée est d'être indépendante du nombre de couleurs représentatives, ce qui la rend plus rapide que les méthodes usuelles dès que le nombre de couleurs représentatives devient important (autour de 200).

1.2.3 Pyramides combinatoires

Le terme pyramide combinatoire est un raccourci pour pyramide de cartes combinatoires. Ce sujet de recherche a débuté en 1998 par un échange de courrier éléctroniques entre Walter Kropatsch et moi même. Fortement impliqué dans la recherche sur les cartes combinatoires, je commençais alors à m'intéresser aux modèles hiérarchiques. Inversement, Walter Kropatsch fortement impliqué dans les modèles hiérarchiques a manifesté une certaine curiosité puis de l'intérêt vis-à-vis des cartes combinatoires. Cette convergence d'intérêt a marqué la naissance des pyramides combinatoires.

Les cartes combinatoires et les cartes combinatoires généralisées permettent de coder les partitions d'un espace de dimension n en objets orientables ou non orientables avec ou sans bords. Le concept de carte a tout d'abord été introduit par Edmond [70] en 1960 puis étendu par plusieurs auteurs [84, 44, 132]. Ce modèle a été ensuite appliqué à l'informatique graphique et à la vision par de nombreux auteurs [131, 26, 66, 15, 76, 21, 30, 24, 60, 64].

Grossièrement, une carte combinatoire $2D$ peut être comprise comme un graphe planaire dans lequel on stocke explicitement l'ordre des arêtes autour de chaque sommet. Afin de distinguer les deux extrémités d'une arête, on introduit le concept de brin. Une arête est donc composée de deux brins codant chacune de ses extrémités.

Dans le cadre des pyramides usuelles (pyramides régulières, de graphes simples ou de graphes duaux), les relations entre les niveaux de la pyramides sont exprimées en terme d'ensembles. Dans le cadre d'une construction ascendante, un pixel ou un sommet de la pyramide possédera un ensemble de fils au niveau précédent et un ensemble de descendants dans le niveau de base. Dans le cadre des cartes combinatoires, un ordre est défini localement autour de chaque sommet. Les opérations

de réduction permettant de construire les différents niveaux de la pyramides ont pour effet de fusionner les relations d'ordres établies indépendamment pour chaque sommet. Les relations entre les niveaux dans le cadre des pyramides combinatoires sont donc codées non pas avec des ensembles mais avec des séquences de brins. Cette propriété en apparence anodine a des répercussions dont nous tachons de mesurer les conséquences depuis maintenant 4 ans. En effet, une séquence est un ensemble ordonné. Il possède donc une structure et des propriétés. Ces propriétés se sont avérées extrêmement riches ce qui nous a amenés à redéfinir non seulement le mode de construction d'une pyramide mais également l'ensemble des concepts utilisés dans les représentations hiérarchiques. Notre schéma de construction d'une pyramide combinatoire est toutefois identique à celui d'une pyramide de graphes duaux. Les objets manipulés et leurs relations seront simplement différents dans les deux cas.

Les propriétés spécifiques des pyramides combinatoires sont les suivantes :

1.2.3.a Codage implicite du dual

Dans le cadre des pyramides de graphes duaux, le graphe dual permet de caractériser les arêtes redondantes. Toutefois, il serait trop coûteux de calculer le dual de chacun des graphes définis dans la pyramide. On stocke donc explicitement le graphe et son dual au niveau de base de la pyramide et les deux structures sont mises à jour en parallèle durant la construction de celle-ci.

Dans le cadre des cartes combinatoires, la carte duale peut être calculée sans surcoût et donc codée implicitement. On a donc une seule structure de données à stocker et maintenir.

1.2.3.b Codage de l'orientation

Les opérations de réduction utilisées pour construire une pyramide combinatoire préservent l'orientation. Cette information n'est pas préservée ou codée explicitement dans les autres types de pyramides. Intuitivement, si la pyramide code une partition, chaque sommet de la pyramide code une région dans la carte de base. Le codage de l'orientation des arêtes autour de chaque sommet nous donne la séquence des régions rencontrées en tournant par exemple dans le sens positif autour de la région associée à ce sommet. Une telle propriété peut, par exemple, être utile dans le cadre du suivi de contours.

Des résultats récents semblent suggérer, mais cela reste à démontrer, que le codage de l'orientation nous permet de définir la frontière entre deux régions comme un chemin orienté. Ceci signifie que :

1. l'ensemble des pixels d'un chemin peut être retrouvé simplement en suivant l'orientation du chemin.
2. en tout point d'un chemin entre deux régions, l'orientation nous permet d'identifier chacune des deux régions de part et d'autre du chemin. Cette propriété est similaire à celle du bonhomme d'Ampère que tout un chacun a du rencontrer au cours de ses études : un homme marchant le long de chemin dans le sens de l'orientation aura toujours la même région à main droite.

1.2.3.c Codage implicite des relations père-fils

Les pyramides usuelles stockent explicitement le lien parent-fils entre les différents niveaux de la pyramide. Nous avons montré que dans le cadre des pyramides combinatoires l'ensemble de la

pyramide peut être retrouvé en codant pour chaque arête :
- l'opération de réduction qui l'a supprimée ;
- le niveau dans la pyramide où cette opération a été effectuée.

On peut donc coder l'ensemble de la pyramide uniquement en rajoutant des attributs dans la carte de base. Ce "pliage" de la pyramide sur un seul niveau peut sembler paradoxal pour un modèle hiérarchique où l'on désire coder plusieurs niveaux de résolution. Il présente toutefois 2 intérêts :

1. Le codage implicite des relations père-fils permet une plus faible occupation de la mémoire (ou du disque). Ceci peut s'avérer un avantage important dans certaines applications où le stockage mémoire d'une seule partition pose déjà des problèmes (par exemple dans le cas des applications décrites dans la section 1.2.3.d).
2. ce type de codage implicite doit permettre un accès plus direct à la géométrie. Ainsi, pour un brin survivant de niveau n, le codage implicite permet d'éviter d'itérer les relations père-fils du niveau n au niveau de base pour retrouver son champ récepteur. Celui-ci peut être retrouvé directement dans la carte de base à partir du codage implicite.

1.2.3.d Extension à des dimensions quelconques

Comme nous l'avons vu, les cartes combinatoires permettent de coder des partitions d'un espace de dimension quelconque. Notre étude s'est jusqu'à présent limitée aux cartes $2D$. Toutefois, des travaux de Damiand et Lienhardt [60] semblent montrer que le schéma que nous avons suivi en $2D$ peut être étendu en dimension quelconque. Les principales applications d'une telle extension du formalisme des pyramides combinatoires concernent en priorité la $3D$ avec la segmentation d'images volumiques. On peut également envisager des partitions d'images $4D$ dans le cadre d'animations d'images volumiques. L'extension à des dimensions supérieures n'est pas aussi aisée avec les pyramides usuelles et constitue une spécificité des pyramides combinatoires.

1.3 Plan du mémoire

L'ensemble des activités de recherche dans lesquelles je me suis impliqué depuis ma thèse de doctorat est représenté sur la figure 1.3. Comme l'on peut le voir, cette figure comporte trois branches correspondant de droite à gauche :
- au traitement et l'affichage d'images couleur ;
- à la segmentation et reconstruction d'objets métalliques ;
- à la structuration de partitions par des modèles pyramidaux.

Ces trois parties reflètent le plan de ce mémoire et sont respectivement traitées dans les chapitres 3, 2 et 4. L'inversion entre les chapitres 2 et 3 a été dictée par la description des modèles de réflexion dans le chapitre 2. Ces modèles décrivent la formation d'une image couleur et il était logique de les présenter avant le traitement de telles images. Le chapitre 4 traite du codage de hiérarchie de partitions et peut être abordé indépendamment des deux premiers.

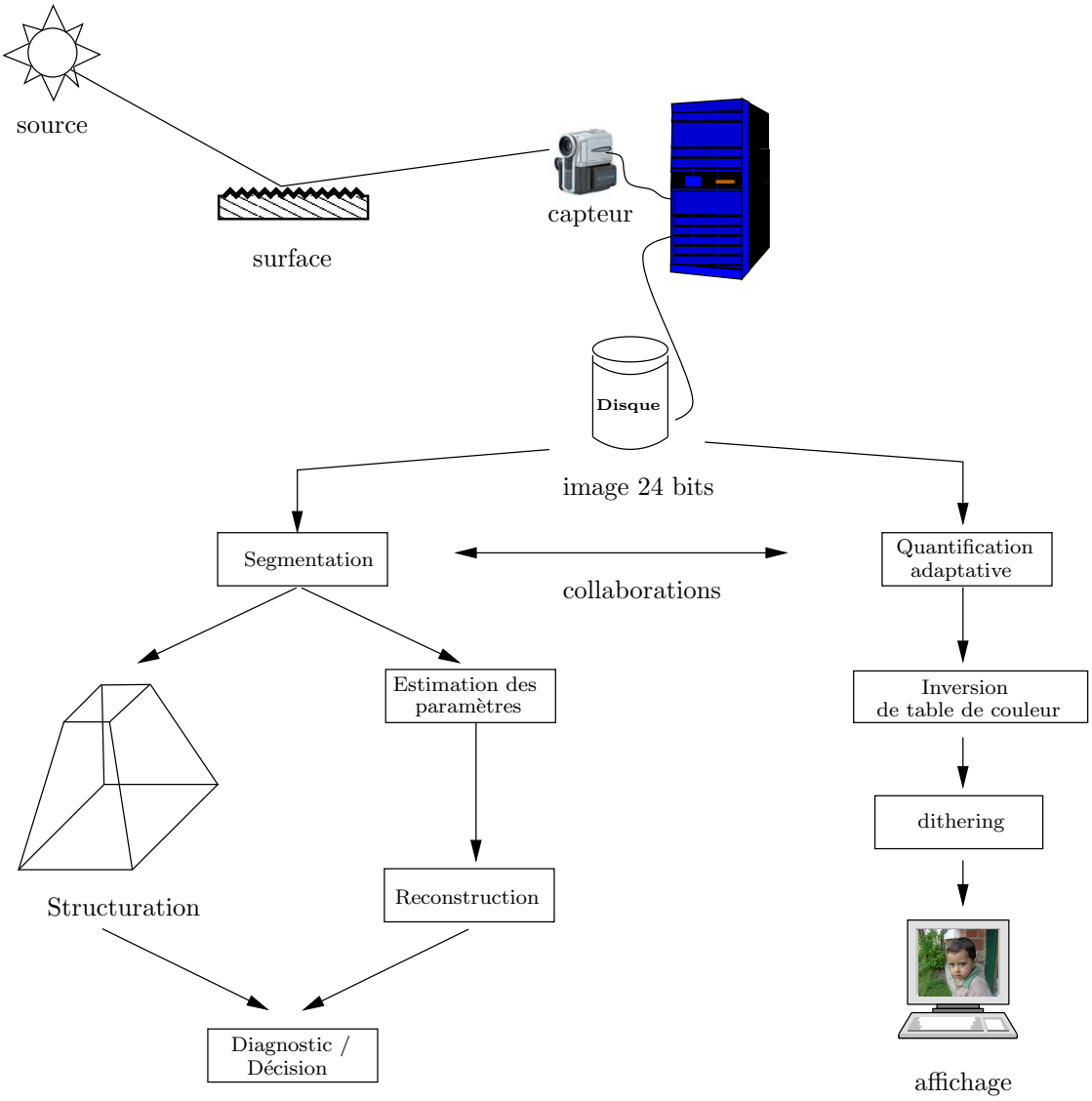


FIG. 1.3 – *L'ensemble des opérations appliquées à une image auxquelles je me suis intéressé.*

Chapitre 2

Modèles d'acquisition d'images

2.1 Introduction

J'ai été amené à m'intéresser aux modèles décrivant l'intensité d'un pixel en fonction de la géométrie d'une scène dans le cadre de la thèse de Laurent Hussenet [109]. La thèse réalisée en partenariat avec l'entreprise Axon Cable portait sur le contrôle industriel de brasures et plus généralement d'objets manufacturés métalliques. L'entreprise nous a tout d'abord demandé de concevoir des méthodes permettant le contrôle de brasures. Cette entreprise effectue en effet de nombreuses brasures afin de souder des câbles électriques sur des connecteurs (Fig. 2.1). Pour ce type d'application, les principaux paramètres à prendre en compte pour le diagnostic de la brasure sont l'aspect colorimétrique de celle-ci qui peut traduire une surchauffe de l'apport (généralement de l'étain) et l'aspect volumique de la brasure qui permet de mesurer si la quantité optimale d'apport a été utilisée. Par la suite, l'entreprise Axon Cable nous a également demandé de prévoir le contrôle de pièces métalliques susceptibles de recevoir des chocs lors du processus de fabrication. La partie de la pièce soumise au choc présente alors un aspect concave qui peut être caractérisé par un contrôle volumique de la pièce (Fig. 2.2).

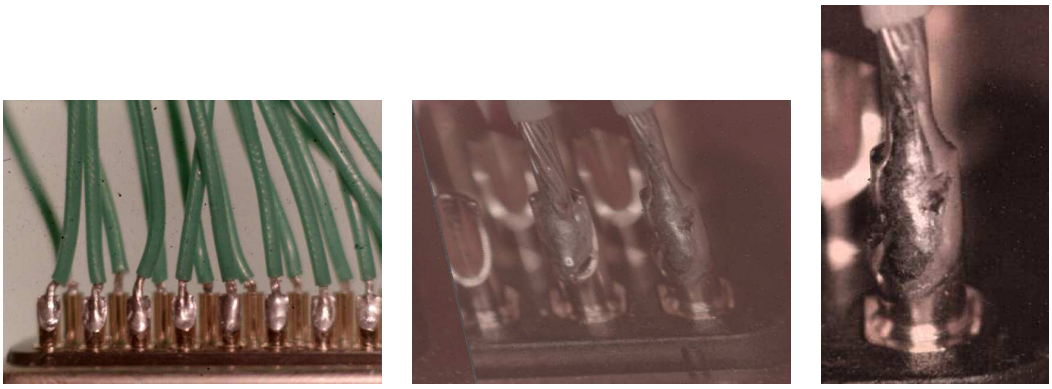
Le contrôle volumique d'une pièce se déroule classiquement en deux phases :

1. La segmentation d'une image contenant l'objet de façon à extraire celui-ci.
2. Une reconstruction $3D$ de l'objet à partir d'une ou plusieurs images afin de contrôler la forme de celui-ci.

Malgré ce cadre assez vaste, les tâches à accomplir comportaient un certain nombre de spécificités qui ont guidé nos directions de recherche :

1. Le nombre d'objets à contrôler peut être assez vaste mais dans chaque cas les images comportent peu de matériaux différents et les objets à contrôler sont composés d'un seul matériau (l'étain pour les brasures).
2. Les objets à contrôler sont tous de type métallique.
3. Le diagnostic doit pouvoir s'effectuer dans un temps relativement court (de l'ordre de la seconde) avec des moyens minimums.

La première spécificité nous a amenés à nous intéresser aux algorithmes de segmentation de scènes en matériaux [111, 94, 95, 91]. En effet, les objets à extraire des images étant composés



(a) (b) (c)

FIG. 2.1 – Un ensemble de brasures sur un connecteur avec trois facteurs de zoom.

d'un seul matériau, une segmentation de la scène en matériaux doit nous fournir une région pour chaque objet. De plus chaque image étant composée d'un nombre restreint de matériaux, nous pouvons envisager d'utiliser un mécanisme d'apprentissage pour reconnaître spécifiquement certains matériaux. Enfin la segmentation d'une scène en matériaux suppose une certaine caractérisation des matériaux composant les objets de la scène. Cette caractérisation peut être utilisée comme prédiagnostic de l'objet avant sa reconstruction. Par exemple, une brasure trop chauffée présentera une couleur légèrement différente d'une brasure «normale». La segmentation en matériaux doit pouvoir nous indiquer cette altération de la couleur du matériau.

La deuxième spécificité nous a amenés à nous intéresser aux modèles physiques de réflexions (section 2.2). En effet, la plupart des méthodes de reconstruction sont basées sur le modèle Lambertien (section 2.2.1). Ce modèle est efficace pour les matériaux peu réfléchissants mais ne rend que très partiellement compte des phénomènes optiques qui se produisent lors de la réflexion d'un rayon lumineux sur un objet métallique.

La reconstruction $3D$ d'un objet suppose au minimum une source lumineuse et une caméra. La troisième contrainte nous a amenés à partir de ce minimum et à complexifier le mécanisme

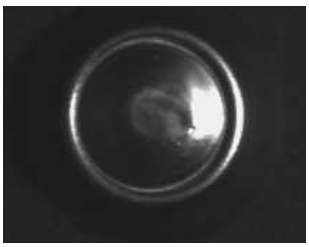


FIG. 2.2 – Exemple de pièce métallique dont le diagnostic nécessite un contrôle volumique. La partie concave au centre de la pièce est consécutive à un choc et constitue un défaut.

d'acquisition uniquement lorsque les critères de qualité que suppose le diagnostic ne pouvaient être atteints avec le mécanisme d'acquisition existant.

Notons enfin avant d'entrer dans le vif du sujet que les applications que nous verrons dans les sections suivantes ont uniquement trait à la vision. Les modèles de réflexion ont toutefois des applications évidentes en synthèse d'images (voir par exemple [175, 158])

2.2 Les différents modèles de réflexion

Comme nous l'avons mentionné dans le chapitre 1, une image est la projection $2D$ d'une scène $3D$. Chaque objet de cette scène réfléchi la lumière incidente en modifiant sa couleur ou son intensité. Cette modification de la lumière incidente définie la couleur de chaque point d'un objet. Les modèles de réflexions décrivent simultanément la couleur d'un objet et l'évolution de cette couleur lorsque l'on change l'un des paramètres de l'acquisition. Il est relativement clair que de nombreux matériaux réfléchissent la lumière de façons très variées. Un pot de céramique et de cuivre auront par exemple des propriétés optiques très différentes. Ces différents types de réflexions seront donc caractérisés par différents modèles décrivant différent types de matériaux. On distingue notamment [172, 93, 108] :

1. Les matériaux conducteurs comme les métaux. Comme nous le verrons par la suite ces matériaux atténuent rapidement l'onde incidente si bien que le phénomène de réflexion est essentiellement un phénomène de surface.
2. Les matériaux peu conducteurs également appelés *diélectriques* tels que le verre. Ces matériaux laissent au contraire pénétrer profondément l'onde incidente dans l'objet. La description de la réflexion exige donc pour ces matériaux une modélisation des phénomènes optiques dans le matériau.
3. Les matériaux optiquement homogènes. Ces matériaux ont un indice de réfraction constant à l'intérieur du matériau. Pour ce type d'objet la réflexion de l'onde incidente peut être décrite uniquement à partir de la réflexion de l'onde sur la surface du matériau. Les métaux, le verre, les cristaux sont des exemples communs de matériaux homogènes.
4. Les matériaux optiquement inhomogènes sont composés d'un matériau qui inclue de nombreuses particules colorantes dont les propriétés optiques sont différentes de celles du support. Dans ce cas, le calcul de l'onde réfléchie doit tenir compte de l'interaction de l'onde incidente avec les particules de colorant. Les plastiques, le papier, les textiles et les peintures font partie des matériaux optiquement inhomogènes.

Notez que les points 1 et 2 indiquent si le phénomène de réflexion est un phénomène de surface. Les points 3 et 4 décomposent les matériaux en fonction des phénomènes optiques se produisant à l'intérieur de ceux ci.

Un autre paramètre important d'une surface est sa rugosité. Ainsi, des matériaux homogènes et parfaitement lisses réfléchissent la lumière dans une direction symétrique au rayon incident par rapport à la normale. Ce phénomène est appelé une *réflexion spéculaire*. Inversement des matériaux homogènes plus rugueux diffusent la lumière autour de la réflexion spéculaire. L'ensemble des rayons réfléchis est appelé le *lobe spéculaire*.

L'outil le plus adapté pour décrire le mécanisme de la réflexion est la théorie des ondes électromagnétiques et les équations de Maxwell. Une onde électromagnétique est composée d'un champ électrostatique $\overrightarrow{E}$ et d'un champ magnétique $\vec{B}$.

Le flux d'énergie par unité de surface ($W.m^{-2}$) d'une onde électromagnétique est défini par son *vecteur de Poynting* :

$$\vec{P} = \frac{\vec{E} \wedge \vec{B}}{\mu} = \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\|E\|^2\vec{k}$$

où μ et ϵ sont respectivement appelés la perméabilité magnétique et la permittivité électrique du matériau dans lequel se propage l'onde. Le vecteur $\vec{k}$ est un vecteur unitaire indiquant la direction de propagation de l'onde. Ce vecteur est normal au plan dans lequel évoluent les vecteurs $\vec{E}$ et $\vec{B}$.

L'irradiance d'une onde électromagnétique est définie comme la quantité d'énergie de l'onde par unité de surface ($W.m^{-2}$) :

$$\mathrm{Ir} = \frac{d\Phi_i}{dA}$$

où $d\Phi_i$ (W) représente le flux d'énergie et dA (m^2) un élément de surface.

La *radiance* ($W.m^{-2}.sr^{-1}$) d'un élément de surface dans une direction (θ_r, ψ_r) est définie comme la quantité d'énergie émise par la surface par unité de surface et unité d'angle solide. La radiance d'un patch de surface dA dans la direction (θ_r, ψ_r) est donc définie par :

$$L = \frac{d^2\Phi_r}{dA\cos(\theta_r)d\omega_r}$$

où $d\omega_r$ est l'angle solide sous lequel le patch de surface voit l'observateur et $d^2\Phi_r$ l'énergie émise dans le cône défini par $d\omega_r$.

Horn [102] à montré que la radiance émise par un patch de surface était proportionnelle à l'irradiance à l'entrée des capteurs de la caméra. Plus précisément ces deux quantités sont liées par :

$$I_r = L_r\frac{\pi}{4}\left(\frac{d}{f}\right)^2\cos(\gamma)$$

où d, f et γ sont des paramètres de la caméra illustrés sur la figure 2.7. Dans la plupart des applications nous pourrons supposer $\gamma = 0$.

Finalement la *réflectance* (sr^{-1}) d'un matériau est égale au rapport entre la radiance émise par un patch de surface dans une direction et l'irradiance reçu par ce même patch de surface à partir d'une autre direction

$$R = \frac{L}{\mathrm{Ir}}.$$

La réflectance est souvent également désignée par BRDF [158] (Bi directional Reflectance Distribution Function). Du point de vue d'un utilisateur de la physique, la BRDF est sans doute la quantité la plus utile puisqu'elle nous indique ce que renvoie une surface en fonction de ce quelle reçoit. Les différentes quantités mentionnées ci-dessus sont résumés dans la Table 2.1.

Une source lumineuse peut se concevoir comme la superposition d'un ensemble d'ondes monochromatiques. Le *spectre* d'une source lumineuse est défini comme la fonction donnant la puissance de l'onde en Watt (W) pour chaque longueur d'onde. Le spectre visible se situe approximativement entre 400 et 700 nano mètres ($1nm = 10^{-9}m$). Les impressions de couleur en fonction des longueurs d'ondes se répartissent approximativement comme suit :

- autour de 450 nm : impression de bleu ;
- entre 500 et 570 nm : impression de vert ;
- entre 570 et 600 nm : impression de jaune ;

Nom	Symbole	Définition	Unité
Flux d'énergie	$d\Phi$		W
Irradiance	Ir	$\frac{d\Phi}{dA}$	$W.m^{-2}$
Radiance	L	$\frac{d^2\Phi}{dA\cos(\theta_r)d\omega_r}$	$W.m^{-2}.sr^{-1}$
Réflectance	R	$\frac{L}{I}$	sr^{-1}

TAB. 2.1 – Quantités utilisées en électro-magnétisme. Les symboles W, m et sr désignent respectivement des Watts, mètres et stéradians

– entre 600 et 700 nm : impression de rouge.

Nous verrons plus précisément dans la section 2.3 le lien existant entre le spectre d'une onde électromagnétique et la sensation colorée. Une classification des ondes électromagnétiques en fonction de leur longueur d'onde est donnée sur la figure 2.3.

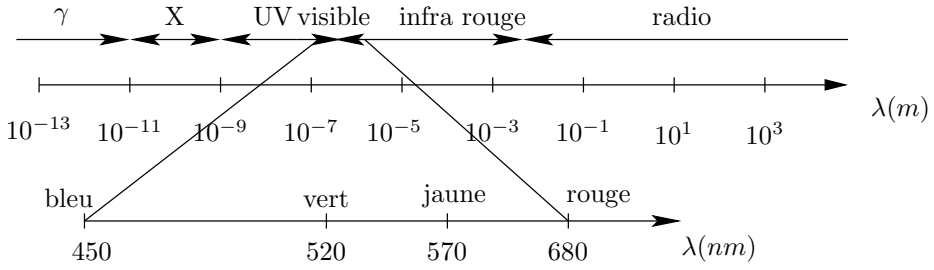


FIG. 2.3 – *Classification des ondes électromagnétiques en fonction de leur longueur d'ondes.*

L'électromagnétisme permet de décrire les propriétés optiques d'un matériau à l'aide des constantes μ, ϵ et σ décrivant respectivement la perméabilité magnétique, la permittivité électrique et la conductivité du matériau (Table 2.2). Les propriétés optiques d'un matériau peuvent également être résumées à l'aide d'une seule variable appelée *l'indice complexe de réfraction* et notée $M = n - iK_0$ où n et K_0 sont deux réels. La constante n est appelée la part réfractive de M. Dans des matériaux n'atténuant pas le signal ($K_0 = 0$), la constante n est égale au rapport entre la vitesse de la lumière dans le vide et dans le matériau. La constante K_0 est directement impliquée dans l'atténuation de l'onde électromagnétique dans le matériau. En effet, l'irradiance d'une onde planaire de fréquence ω de direction $x > 0$ qui heurte le plan (Ozy) est égale à :

$$\mathrm{Ir}(x) = \mathrm{Ir}(0)e^{-\frac{2\omega K_0 x}{c}}$$

où c est la vitesse de la lumière dans le vide.

Cette atténuation de l'énergie de l'onde incidente se fait au bénéfice de l'apparition d'un courant appelé *courant de surface*. La profondeur $x = \frac{c}{2\omega K_0}$ est appelée la profondeur de peau du matériau. Les constantes n et K_0 peuvent être calculées en fonction de μ, ϵ et σ en utilisant les lois de

Nom	Symbole	Unité	Valeur dans le vide
Perméabilité magnétique	μ	$\frac{C^2}{N.m^2}$	$4\pi 10^{-7}$
Permittivité électrique	ϵ	$\frac{N.sec^2}{C^2}$	$(36\pi 10^9)^{-1}$
Conductivité	σ	$(\Omega.m)^{-1}$	0

TAB. 2.2 – Constantes électro-magnétiques

Maxwell [177] :

$$\begin{aligned} n(\lambda) &= \frac{\mu\epsilon c^2}{2}\left[1+\sqrt{1+\left(\frac{\lambda\sigma}{2\pi c\epsilon}\right)^2}\right] \\ K_0(\lambda) &= \frac{\mu\epsilon c^2}{2}\left[-1+\sqrt{1+\left(\frac{\lambda\sigma}{2\pi c\epsilon}\right)^2}\right]. \end{aligned} \tag{2.1}$$

Nous pouvons immédiatement observer que n et K_0 (et donc M) ne dépendent que de la longueur d'onde de l'onde incidente. De plus, pour des matériaux conducteurs σ sera important et donc K_0 également. L'onde électromagnétique pénètre donc faiblement dans les matériaux conducteurs pour lesquels le phénomène de la réflexion est essentiellement un phénomène lié à la surface du matériau. Inversement, des matériaux peu conducteurs auront un K_0 faible ce qui favorisera la pénétration de l'onde dans le matériau. Ce phénomène devra donc être pris en compte dans la modélisation de la réflexion.

Un dernier paramètre fondamental dans la description de la réflexion d'une onde électromagnétique est le *Coefficient de Fresnel* [29] qui décrit la fraction de l'onde incidente réfléchie par la surface d'un matériau. Dans le cas d'un matériau isotropique et homogène, la réflexion d'une onde non polarisée heurtant une surface lisse avec un angle θ_l produit un coefficient de Fresnel égal à :

$$F(\theta_l,\lambda) = \frac{1}{2}\left(R_\parallel(\theta_l,\lambda) + R_\perp(\theta_l,\lambda)\right) \tag{2.2}$$

où $R_\parallel(\theta_l,\lambda)$ et $R_\perp(\theta_l,\lambda)$ décrivent le coefficient de Fresnel dans le cas d'une onde polarisée respectivement parallèlement et perpendiculairement au plan d'incidence (plan contenant la normale à la surface et le rayon incident).

Les termes $R_\parallel(\theta_l,\lambda)$ et $R_\perp(\theta_l,\lambda)$ peuvent se déduire de la théorie des ondes électromagnétiques. Toutefois la forme explicite de ces termes n'étant pas utile pour la suite de ce document nous nous contenterons de noter que $R_\parallel(\theta_l,\lambda)$ et $R_\perp(\theta_l,\lambda)$ peuvent s'exprimer sous forme d'une fraction de termes dépendant de θ_l, $n(\lambda)$ et $K_0(\lambda)$.

2.2.1 Le modèle Lambertien

Le modèle Lambertien est le modèle le plus utilisé pour décrire les phénomène de réflexion à l'intérieur des diélectriques. Ce modèle permet de relier l'irradiance incidente à un capteur à l'angle θ formé par l'onde incidente et la normale à la surface (figure 2.2.1) à l'aide de l'équation suivante :

$$I = I_{diff} = \mathrm{K}_{diff}\cos(\theta) \quad \text{si } \theta \in [0,\frac{\pi}{2}], \quad 0 \text{ sinon} \tag{2.3}$$

où K_{diff} est une constante dépendant du matériau.

Notez que dans un tel modèle la direction de l'observateur n'est pas prise en compte. Intuitivement, un objet Lambertien sera donc un objet dont la couleur ne varie pas lorsque l'on se déplace. Du bitume en milieu de journée fournit un bon exemple de surface Lambertienne.

Une explication qualitative de ce modèle généralement acceptée est la suivante : l'énergie incidente à une surface pénètre dans celle-ci et est réfléchie aléatoirement à l'intérieur de l'objet par de microscopiques in-homogénéités du matériau. Au cours de ces multiples réflexions une partie de l'énergie incidente est ré-émise par la surface et ressort de l'objet suivant une direction aléatoire. Les réflexions multiples dans le matériau ne subissant aucune contrainte particulière, l'énergie est ré-émise de façon uniforme par la surface. L'intensité de l'énergie émise par un point est donc indépendante de la direction d'observation et uniquement fonction de la quantité d'énergie incidente tombant sur la surface. Cette quantité s'exprime comme un cosinus de l'angle entre la normale à la surface et la direction de la source.

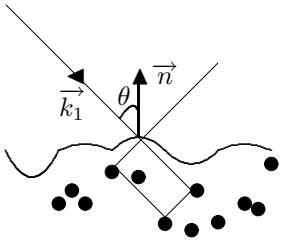


FIG. 2.4 – Réflexion Lambertienne : l'onde incidente se propage dans la direction définie par le vecteur $\vec{k_1}$. Elle est réfléchie plusieurs fois à l'intérieur du matériau avant de ressortir de celui-ci suivant une direction aléatoire.

Une explication plus quantitative du modèle Lambertien peut être donnée dans le cas d'un diélectrique inhomogène. Ce type de matériau a été étudié par Reichmann [165] qui à étendu un modèle initialement développé par Kubelka et Munk [125]. Reichmann modélise la répartition des pigments de colorants dans le matériau à l'aide d'un ensemble de couches superposées. L'interaction entre chaque couche élémentaire et l'onde incidente est décrite par les fonctions $\alpha(\lambda)$ et $\beta(\lambda)$ qui décrivent respectivement la fraction de l'onde incidente absorbée et réfléchie par unité de longueur.

La théorie de Kubelka et Munk repose sur la résolution de plusieurs équations différentielles du premier ordre dont la résolution fournit un coefficient de réflectance donné par :

$$R_\infty = \frac{2 - \omega(\lambda) - 2\sqrt{1 - \omega(\lambda)}}{\omega(\lambda)} \text{ avec } \omega(\lambda) = \frac{\beta(\lambda)}{\alpha(\lambda) + \beta(\lambda)}.$$

Toutefois, le modèle de Kubelka et Munk repose sur plusieurs hypothèses non réalistes dans le cadre d'applications réelles. Il suppose notamment que le matériau et l'air possèdent le même indice de réfraction afin d'éviter d'avoir à considérer les réflections à la surface du matériau. Ceci est visible dans la formule de R_∞ qui ne dépend d'aucun paramètre géométrique.

L'extension proposée par Reichmann permet de lever ces limitations et nous donne une expression de la réflectance plus générale :

$$R_B(\theta, \lambda) = (1 - R_S)\frac{C(\theta, \lambda)(1 - r_i(\lambda))\,(R_\infty(\lambda) - D(\theta))}{2\,(1 - r_i(\lambda)R_\infty(\lambda))\cos(\theta)} \tag{2.4}$$

où R_S est la fraction de l'onde incidente réfléchie par la surface, $(1 - R_S)$ est donc la fraction de l'onde qui pénètre dans le matériau. Le terme $r_i(\lambda)$ représente la réflectance de surfaces internes au matériau [152] tandis que les fonctions $C(\theta, \lambda)$ et $D(\theta)$ viennent de la résolution des équations du modèle et sont données par :

$$\begin{aligned} C(\theta,\lambda) &= \frac{\omega(\lambda)\cos(\theta)(2\cos(\theta)+1)}{1-4(1-\omega(\lambda))\cos^2(\theta)} \\ D(\theta) &= \frac{2\cos(\theta)-1}{2\cos(\theta)+1} \end{aligned}$$

L'équation 2.4 semble beaucoup plus complexe que l'équation Lambertienne (équation 2.3). Toutefois dans le cas de matériaux n'absorbant pas l'onde incidente ($\alpha(\lambda) \approx 0$) nous avons $\omega(\lambda) \approx 1$ et le lecteur peut vérifier que l'équation 2.4 se réécrit $R_B(\theta, \lambda) = (1 - R_S)$. La partie de l'onde qui n'est pas réfléchie par la surface est donc renvoyée uniformément indépendamment de la longueur d'onde et de l'angle θ formé par l'onde incidente et la normal à la surface. On retrouve donc bien dans ce cas l'explication qualitative de la réflexion Lambertienne donnée au début de cette section. Notez toutefois que pour avoir une irradiance à l'entrée du capteur fonction uniquement de $\cos(\theta)$ nous devons supposer le terme $(1 - R_S)$ approximativement constant. Cette hypothèse n'est pas valable dans le cas des matériaux conducteurs (voir par exemple la section 2.2.2). Notons de plus que quelque soit le type du matériau celui-ci acquiert un comportement spéculaire pour un angle d'incidence rasant ($\theta \approx \frac{\pi}{2}$). Dans ce cas le modèle Lambertien n'est plus valable. Ce phénomène explique par exemple, les éblouissements dues à la réverbération du soleil sur le bitume au coucher du soleil.

2.2.2 Le modèle de Beckmann-Spizzichino

Le modèle de Beckmann-Spizzichino [8] est basé sur les lois de l'électromagnétisme et sur une modélisation des micro-aspérités de la surface.

Beckmann modélise les micro-aspérités d'une surface à l'aide d'une fonction aléatoire h des variables x et y décrivant la surface. La forme de la surface est donc déterminée par la densité de probabilité de la fonction h. Beckmann propose d'utiliser une distribution normale de moyenne nulle et d'écart type σ_h. La distribution de h est alors égale à :

$$\rho_h(h) = \frac{1}{\sqrt{2\pi}\sigma_h} e^{-\frac{h^2}{2\sigma_h^2}}.$$

Toutefois, une telle modélisation ne permet pas un contrôle efficace de la forme de la surface. En effet, plusieurs tirages aléatoires de la fonction h avec le même σ_h peuvent présenter des aspects très différents. Intuitivement, la raison de cette limitation est que le paramètre σ_h contrôle l'altitude moyenne des pics de la fonction h mais pas l'écart entre deux pics (figure 2.5).

Afin de pallier cette limitation, nous définissons un paramètre $C(\tau)$ qui représente la corrélation entre les altitudes de deux points séparés par une distance τ. Beckmann propose de représenter cette corrélation par la fonction :

$$C(\tau) = e^{-\frac{\tau^2}{T^2}}$$

où T est la *distance de corrélation* pour laquelle $C(\tau)$ est égal à e^{-1}.

Étant donnée notre modélisation de la surface par les paramètres σ_h et T, la position d'un patch de surface vis à vis de la source lumineuse et de la caméra est représentée sur la figure 2.6(a). Le

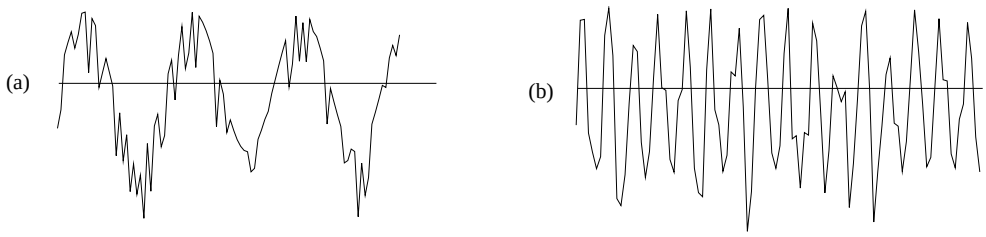
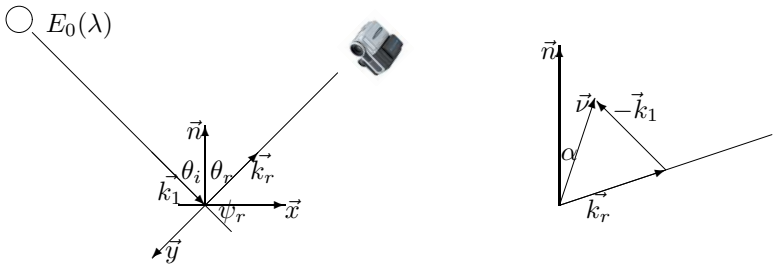


FIG. 2.5 – Deux surfaces aléatoires de valeur σ_h identiques avec une forte (a) et une faible (b) distance de corrélation

repère est choisi tel que l'axe z soit confondu avec la normale $\overrightarrow{n}$ et l'axe x est tel que le vecteur $\overrightarrow{k_1}$ décrivant la direction de la source lumineuse soit dans le plan (Ozx) avec une projection positive sur l'axe x. La puissance de la source lumineuse est décrite par la norme de son vecteur de Poynting $P(\lambda) = \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}E_0^2(\lambda)$. Le vecteur $\vec{k_r}$ décrit la direction de la caméra tandis que le vecteur $\vec{\nu}$ est défini comme la bissectrice de $-\overrightarrow{k_1}$ et $\overrightarrow{k_r}$. Ce vecteur fait un angle α avec la normale à la surface (figure 2.6(b)). Notez que le vecteur $\overrightarrow{k_r}$ n'est pas nécessairement dans le même plan que $\overrightarrow{n}$ et $\overrightarrow{k_1}$. L'angle ψ_r représente son écart vis-à-vis de ce plan (figure 2.6(a)).



(a) Réflexion (b) angle α

FIG. 2.6 – Schéma de réflexion

Les paramètres σ_h, T et les variables illustrés sur la figure 2.6 étant définis, Nayar [148] en se basant sur les travaux de Beckmann, a montré que l'irradiance à l'entrée d'un capteur dA_{im} de la caméra provoquée par un patch de surface d'un conducteur parfait de cotés X et Y (figure 2.7) pouvait être exprimée par :

$$\begin{aligned}\mathrm{Ir} = \mathrm{Ir}_{spec}(\lambda) =& \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}E_o(\lambda)^2\frac{\pi}{4}\left(\frac{d}{f}\right)^2\cos^4(\gamma)\frac{\cos^2(\theta_i)}{\lambda^2}e^{-g} \\ & \left(\left(\frac{z}{f}\right)^2\frac{dA_{im}\cos(\gamma)}{\cos^2(\theta_r)}\rho_0^2 + \frac{\pi T^2 D^2 g}{cos(\theta_r)}\sum_{m=1}^{m=\infty}\frac{g^m}{m!m}e^{-\upsilon_{xy}^2\frac{T^2}{4m}}\right).\end{aligned} \tag{2.5}$$

Les paramètres d, f et γ dans l'équation 2.5 sont relatifs à la caméra et illustrés sur la figure 2.7.

Notez que si la taille de la scène est négligeable par raport à la distance de celle-ci à la caméra nous pouvons supposer $\gamma \approx 0$. Les termes g, ρ_0, ν_{xy} et D sont issus de la résolution de l'intégrale d'Helmhotz et sont égaux à :

$$\begin{cases} g & = & \left(2\pi\frac{\sigma_h}{\lambda}(\cos(\theta_i) + cos(\theta_r)\right)^2 \\ \rho_0 & = & \frac{\sin(\nu_x X)}{\nu_x X}\frac{\sin(\nu_y Y)}{\nu_y Y} \\ \nu_{xy} & = & \sqrt{\nu_x^2 + \nu_y^2} \\ D & = & \frac{1+\cos(\theta_i)\cos(\theta_r)-\sin(\theta_i)\sin(\theta_r)\cos(\psi_r)}{\cos(\theta_i)(\cos(\theta_i)+\cos(\theta_r))} \end{cases} \tag{2.6}$$

où $\overrightarrow{\nu} = (\nu_x, \nu_y, \nu_z)$ (figure 2.6(b)).

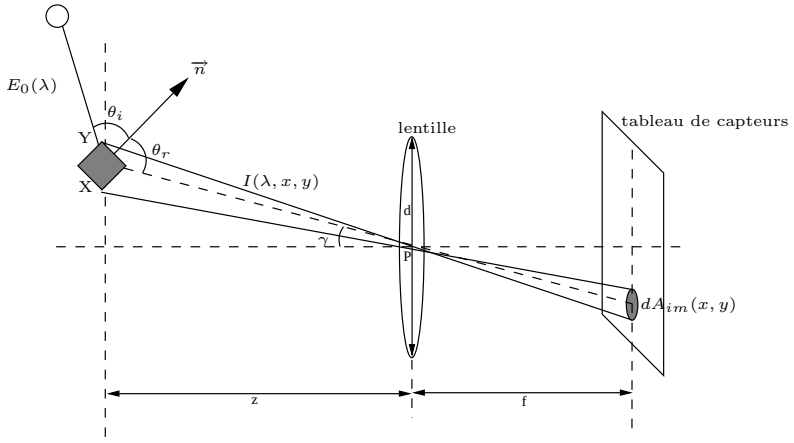


FIG. 2.7 – *Réflexion d' une onde électromagnétique frappant un des capteurs de la caméra*

Le paramètre g est proportionnel au carré du rapport $\frac{\sigma_h}{\lambda}$ qui représente le rapport entre l'altitude moyenne de nos irrégularités et la longueur d'onde de la source incidente. Le facteur g représente donc la rugosité de la surface et les cas $g << 1$, $g \approx 1$ et $g >> 1$ représenteront respectivement une surface lisse, modérément rugueuse et rugueuse.

Le terme ρ_0 est une fonction qui décroît très rapidement dès que $\nu_x X$ ou $\nu_y Y$ n'est pas proche de 0. Les coordonnées ν_x et ν_y sont égales à 0 lorsque $\overrightarrow{\nu}$ est confondu avec $\overrightarrow{n}$. Dans ce cas, $\overrightarrow{n}$ est la bissectrice des vecteurs sources et destination $-\overrightarrow{k_1}$ et $\overrightarrow{k_r}$. Ce type de réflexion est appelé une réflexion spéculaire tandis que le terme décrivant ce type de réflexion est appelé un *pic spéculaire*. Le terme comprenant le facteur ρ_0^2 dans l'équation 2.5 correspond donc à une fonction pic dont le maximum est atteint pour une réflexion spéculaire.

Le terme comprenant une somme infinie dans l'équation 2.5 est appelé *le lobe spéculaire*. Ce terme correspond à une fonction qui prend son maximum lorsque la direction d'observation correspond à la direction spéculaire mais décroît plus lentement que le pic spéculaire. Notez tout de même le facteur $e^{-\upsilon_{xy}^2 \frac{T^2}{4}}$ qui décroît rapidement lorsque $\overrightarrow{k_r}$ ne correspond pas à la direction spéculaire ($\vec{\nu} = (0, 0, 1)$).

L'équation 2.5 se simplifie en fonction de la rugosité de la surface de la façon suivante :

– pour une surface lisse ($g << 1$) :

$$\mathrm{Ir}_{spec}(\lambda) = \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\cos^2(\theta_i)e^{-g}\left(\frac{\kappa_{ps}}{\cos^2(\theta_r)}\rho_0^2 + \frac{\kappa_{ls}D^2 g}{cos\theta_r}e^{-\upsilon_{xy}^2\frac{T^2}{4}}\right); \tag{2.7}$$

– pour une surface rugueuse ($g >> 1$) :

$$\mathrm{Ir}_{spec}(\lambda) = \kappa_{ls}\frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\frac{e^{\left(\frac{-\upsilon_{xy}^2T^2}{4\upsilon_z^2\sigma_h^2}\right)}}{\upsilon_z^2\sigma_h^2}\cos(\theta_i)D^2. \tag{2.8}$$

où κ_{ps} etκ_{ls} sont deux constantes.

Notez que pour $g = 0$, le lobe spéculaire est nul alors que le pic spéculaire est maximum. Au fur et à mesure que la rugosité de la surface augmente, le pic spéculaire diminue jusqu'à disparaître tandis que le lobe spéculaire devient prépondérant.

Une description plus qualitative du pic et du lobe spéculaire sera donnée dans la section 2.2.4 ; Nous pouvons toutefois dès à présent indiquer les conditions de validité de l'équation 2.5 :

1. Les irrégularités de la surface sont supposées suivre une distribution uniforme de moyenne nulle. Cette hypothèse est raisonnable lorsque l'on ne dispose d'aucune information à priori sur le type des irrégularités. Notons également que Beckmann [8] a défini l'irradiance pour de nombreux types d'irrégularités. Cette restriction peut donc être levée si l'on dispose d'informations plus précises sur la surface.
2. Le rayon de courbure des irrégularités doit être supérieur en tout point de la surface à la longueur d'onde du rayon incident. Cette restriction est indispensable pour pouvoir développer les équations physiques menant à l'équation 2.5. Les valeurs mesurées peuvent donc s'écarter de celles prédites par l'équation 2.5 si les irrégularités comportent des pics très étroits.
3. Le matériau doit être un conducteur parfait. Cette restriction est encore une fois dictée par des considérations pratiques afin de pouvoir intégrer l'intégrale d'Helmotz donnant le champ magnétique en un point P à partir du champ magnétique sur la surface. Supposer que la matériau est un conducteur parfait revient à supposer que le coefficient de Fresnel (équation 2.2) F est égal à $+1$ ou -1 selon la polarisation de l'onde. Le phénomène de réflexion n'induit dans ce cas aucune atténuation de l'onde incidente. Beckmann propose d'approximer l'irradiance d'un matériau de conductivité finie en approximant le coefficient de Fresnel F en chaque point de la surface par sa moyenne $< F >$ sur la surface. L'irradiance est alors donnée par :

   $$\mathrm{Ir}_f(\lambda) =< FF^* > \mathrm{Ir}_\infty(\lambda) \tag{2.9}$$

   où les indices f et ∞ représentent l'irradiance pour un matériau de conductivité finie et infinie tandis que F^* représente le conjugué de F. Cette solution revient à approximer une somme de produits par un produit de sommes.

   Contrairement aux deux restrictions précédentes qui peuvent être ignorées dans la plupart des applications pratiques, cette dernière restriction interdit *a priori* d'utiliser l'équation 2.5 pour des isolants ou des matériaux de faible conductivité. Nous pouvons toutefois considérer que l'équation 2.5, sans nous donner une expression exacte de l'irradiance, nous fournit une bonne description qualitative du phénomène de réflexion à la surface d'un matériau. Cette hypothèse est confirmée par l'équation 2.9.

4. Les ombrages et masquages ne sont pas pris en compte par le modèle. On peut tenir compte de cet effet en substituant $S(x,y)h(x,y)$ à $h(x,y)$ où $S(x,y)$ est une fonction de masquage égale à 0 ou 1 selon que point est masqué ou pas. Il reste toutefois à définir dans ce cas un modèle pour la fonction S.
5. Les réflexions multiples ne sont également pas prises en compte. Encore une fois cette restriction est imposée pour obtenir une expression explicite de l'irradiance.
6. Le champ électromagnétique incident est supposé plan et polarisé perpendiculairement. Ces restrictions sont encore une fois imposées pour des raisons de simplicité. Beckmann a proposé plusieurs approches pour traiter le cas d'ondes de polarisation quelconque. L'hypothèse d'onde plane est justifié lorsque la source lumineuse est à une distance importante de l'objet. Ceci sera vérifié lors de nos acquisitions.

2.2.3 Le modèle de Torrance-Sparrow

Contrairement au modèle de Beckmann-Spizzichino [8] (section 2.2.2), le modèle de Torrance-Sparrow est basé sur l'optique géométrique. Ce modèle néglige donc l'aspect électromagnétique de la lumière. Cette approximation n'est valide que si les irrégularités de la surface sont bien supérieures à la longueur d'onde de la source.

Le modèle de surface utilisé par Torrance-Sparrow est basé sur une modélisation des irrégularités par une série de micro-facettes. Chaque facette est décrite par l'angle β entre sa normale et la normale à la surface macroscopique (figure 2.8). Si nous supposons la surface isotropique, la distribution des normales de facettes est rotationnellement symétrique par rapport à $\overrightarrow{n}$. La distribution de β peut alors être modélisée par une fonction unidimensionnelle telle qu'une distribution normale de moyenne nulle et d'écart-type σ_α. Sachant que β ne peut varier qu'entre 0 et $\frac{\pi}{2}$, la fonction de densité de probabilité de β est égale à :

$$\rho_\beta(\beta) = ce^{-\frac{\beta^2}{2\sigma_\alpha^2}} \text{ avec } c = \left(\int_0^{\frac{\pi}{2}} e^{-\frac{\beta^2}{2\sigma_\alpha^2}} d\beta\right)^{-1}$$

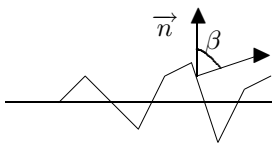


FIG. 2.8 – Modèle de surface de Torrance-Sparow

Ce modèle de surface et les lois de l'optique géométrique permettent d'obtenir une expression explicite de l'irradiance incidente à un capteur de la caméra générée par un patch de surface :

$$\mathrm{Ir}_{ls} = \kappa_{spec} \frac{L_i dw_i}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}$$

où α, L_i et dw_i représentent respectivement l'angle entre la normale et le vecteur $\vec{\nu}$ (figure 2.6(b)), la radiance de la source et l'angle solide sous lequel le patch de surface voit la source. La constante

κ_{spec} est donnée par :

$$\kappa_{spec} = \frac{\pi}{4}\left(\frac{d}{f}\right)^2 \cos^4(\gamma)\frac{ca_f F'(\theta_i', \eta')G(\theta_i, \theta_r, \psi_r)}{4} \tag{2.10}$$

où $F'(\theta_i', \eta')$ représente le coefficient de Fresnel et $G(\theta_i, \theta_r, \psi_r)$ un facteur de visibilité entre le patch de surface et la source. Les angles θ_i, θ_r et ψ_r sont représentés sur la figure 2.6 (section 2.2.2). L'angle θ_i' représente l'angle entre le rayon incident et la normales aux micro-facettes susceptibles d'éclairer le capteur. La variable η' représente l'indice complexe de réfraction tandis que a_f représente la surface d'une micro-facette.

Le terme $e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}$ a approximativement la même signification que le terme $e^{-v_{xy}^2 \frac{T^2}{4}}$ dans le modèle de Beckmann-Spizzichino (section 2.2.2) et correspond à un lobe spéculaire. La modélisation de la réflexion de Torrance-Sparrow en utilisant des micro-facettes et les lois de l'optique géométrique conduisent donc à un modèle ne présentant qu'un lobe spéculaire. Ce résultat est attendu dans la mesure ou les lois de l'optique géométriques ne sont valides que pour des surfaces rugueuses. Or le pic spéculaire de Beckmann-Spizzichino n'apparaît que pour des surfaces lisses ou modérément rugueuses (section 2.2.2).

Torrance et Sparrow ajoutent un terme Lambertien à leur équation de réflexion qui devient :

$$\begin{array}{rcccl} \mathrm{Ir} & = & \mathrm{Ir}_{diff} & + & \mathrm{Ir}_{ls} \\ \mathrm{Ir} & = & \kappa_{diff} L_i dw_i \cos(\theta_i) & + & \kappa_{spec}\frac{L_i dw_i}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \quad \text{pour } \theta_i \in [0, \frac{\pi}{2}], 0 \text{ sinon.} \end{array} \tag{2.11}$$

où κ_{diff} représente le coefficient de réflexion diffuse (ou Lambertienne).

L'utilisation de l'optique géométrique conduit à des formules moins complexes que celles induites par les lois de l'électromagnétique. Torrance et Sparrow sont donc conduits à faire moins d'hypothèses simplificatrices que Beckmann et Spizzichino. Leur modèle inclue notamment le coefficient de Fresnel et un coefficient de masquage. Ce modèle est donc applicable à des objets non conducteurs et permet de tenir compte du masquage entre différents éléments de la scène. Ce modèle a toutefois un certain nombre de limitations :

1. L'angle α est supposé avoir une distribution normale. Cette limitation est équivalente à la supposition d'une distribution normale de la hauteur h dans le modèle de Beckmann. Le modèle peut également être facilement adapté à d'autres types de distributions.
2. La taille des micro-facettes doit être beaucoup plus importante que la longueur d'onde du rayon incident. Cette contrainte correspond à la définition du domaine de rugosité de surface pour laquelle les lois de l'optique géométrique peuvent se substituer à celles de l'électromagnétique.
3. La source lumineuse est supposée être éloignée de la scène. Cette contrainte correspond à la modélisation par ondes planaires plutôt que sphériques dans le modèle de Beckmann.

2.2.4 Le modèle de Nayar

Le modèle de Nayar [148] peut se concevoir comme une synthèse des modèles de Beckmann-Spizzichino et Torrance-Sparrow (sections 2.2.2 et 2.2.3). Plusieurs expériences menées par Nayar montrent que le coefficient de Fresnel F et le facteur d'atténuation G du modèle de Torrance-Sparrow (équation 2.10) restent approximativement constants en fonction de θ_i et θ_r. Le coefficient κ_{spec} peut donc être considéré comme constant. De plus si l'on se place dans un protocole

expérimental où la source est variable tandis que la direction d'observation reste constante, les angles θ_r et ψ_r peuvent être considérés comme constants. Sous ces conditions, l'irradiance du lobe spéculaire peut s'exprimer par :

$$\mathrm{Ir}_{ls} = \mathrm{K}_{ls} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \tag{2.12}$$

où K_{ls} est une constante dépendant du matériau et du protocole expérimental.

En revanche si l'on considère des variations simultanées de la source lumineuse et de l'observateur nous ne pouvons négliger le terme $1/cos(\theta_r)$ dans le modèle de Torrance-Sparrow (équation 2.10). L'expression du lobe spéculaire devient alors :

$$\mathrm{Ir}_{ls} = \frac{\mathrm{C}_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \text{ avec } \mathrm{K}_{ls} = \frac{\mathrm{C}_{ls}}{\cos(\theta_r)}. \tag{2.13}$$

Notons que l'équation 2.13 devra être utilisée si l'on considère simultanément plusieurs pixels et donc plusieurs normales avec des angles θ_i, θ_r et ψ_r différents. L'équation 2.12 sera en revanche utilisée lorsque l'on considérera un même pixel soumis à différents illuminants. Dans ce dernier cas θ_i et α sont variables tandis que θ_r et ψ_r peuvent être considérés comme des constantes.

Le pic spéculaire du modèle de Beckmann-Spizzichino peut être approximé par une fonction δ valant 1 dans la direction spéculaire et 0 partout ailleurs. L'intensité du pic spéculaire est alors égale à :

$$\mathrm{Ir}_{ps} = \mathrm{K}_{ps} \delta(\theta_i - \theta_r)\delta(\psi_r)$$

où K_{ps} est égale à la valeur du pic spéculaire (équation 2.5) dans la direction spéculaire.

Finalement, le lobe diffus correspondant à la réflexion Lambertienne peut être ajouté au modèle de façon à avoir une intensité de pixel liée à la géométrie de la scène par :

– si $\theta_i \in [0, \frac{\pi}{2}]$:

$$\mathrm{Ir} = \mathrm{K}_{diff} \cos(\theta_i) + \mathrm{K}_{ls} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + \mathrm{K}_{ps}\delta(\theta_i - \theta_r)\delta(\psi_r) \quad \text{Observateur fixe} \tag{2.14}$$

$$\mathrm{Ir} = \mathrm{K}_{diff} \cos(\theta_i) + \frac{\mathrm{C}_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + \mathrm{K}_{ps}\delta(\theta_i - \theta_r)\delta(\psi_r) \quad \text{Observateur variable;} \tag{2.15}$$

– si $\theta_i \geq \frac{\pi}{2}$, $\mathrm{Ir} = 0$.

Notez encore une fois que le cas d'un observateur variable (équation 2.15) peut s'appliquer soit :

1. à l'étude d'un pixel avec des positions successives de la caméra. Notons que si la source lumineuse est supposé d'orientation constante le terme $\mathrm{K}_{diff} \cos(\theta_i)$ est dans ce cas également constant. Le cas de l'observateur variable s'applique également
2. à l'étude de plusieurs pixels avec une seule caméra fixe. Dans cas aucun des angles $\theta_i, \theta_r, \psi_r$ et α ne peut être considéré constant.

Nayar a de plus établi des ponts entre les deux modèles en remarquant que puisque α est l'angle entre $\overrightarrow{\nu}$ et la normale nous avons (équation 2.6) :

$$\tan(\alpha) = \frac{\nu_{xy}}{\nu_z}. \tag{2.16}$$

Donc si nous posons $\tan(\alpha_0) = \frac{2\sigma_h}{T}$ les lobes spéculaires des modèles de Beckmann et Torrance sont liés par :

$$e^{-\frac{\nu_{xy}^2 T^2}{4\nu_z^2 \sigma_h}} = e^{-\frac{\tan^2(\alpha)}{\tan^2(\alpha_0)}}. \tag{2.17}$$

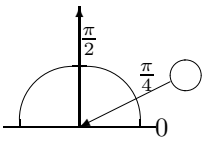
En utilisant l'approximation $\tan(\alpha) \approx \alpha$ nous obtenons :

$$e^{-\frac{v_{xy}^2 T^2}{4\nu_z^2 \sigma_h}} = e^{-\frac{\alpha^2}{2\left(\frac{\alpha_0}{\sqrt{2}}\right)^2}}.$$

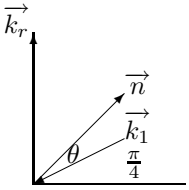
L'écart type σ_α du modèle de Torrance peut donc être relié aux paramètres σ_h e T du modèle de Beckmann par :

$$\sigma_\alpha = \frac{\alpha_0}{\sqrt{2}} = \frac{1}{\sqrt{2}} \tan^{-1}\left(\frac{\sigma_h}{T}\right)$$

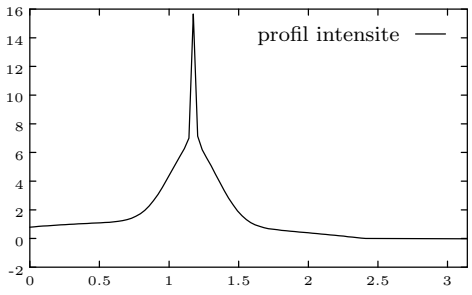
Notez de plus que le modèle de Beckmann définit le lobe spéculaire à partir de $\tan(\alpha)$ plutôt que α. Le modèle de Torrance-Sparrow peut donc se comprendre comme une approximation du modèle de Beckmann avec $\tan(\alpha) \approx \alpha$.



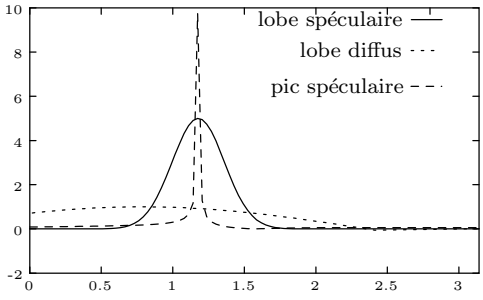
(a) Source-Observateur



(b) Un point du cercle



(c) Profil d'intensité



(d) Contribution des lobes

FIG. 2.9 – Profil d'intensité le long d'une cercle (c) et contribution des différents lobes (d). Le protocole expérimental est illustré sur les Figures (a) et (b)

La figure 2.9 illustre le modèle de Nayar sur un cercle de rayon 1 éclairé par une source lumineuse placée en $\frac{\pi}{4}$ et observé en $\frac{\pi}{2}$ (figure 2.9(a)). Le modèle utilisé est l'équation 2.15. Un point du cercle faisant un angle θ avec l'horizontale vérifiera (figure 2.9(b)) :

$$\begin{cases} \theta_i &= \theta - \frac{\pi}{4} \\ \theta_r &= \frac{\pi}{2} - \theta \\ \alpha &= \frac{\theta_r - \theta_i}{2} = \frac{3\pi}{8} - \theta. \end{cases}$$

La direction spéculaire ($\alpha = 0$) se situe donc en $\frac{3\pi}{8} \approx 1.2$.

L'intensité le long du cercle en fonction de l'angle θ est représentée sur la figure 2.9(c). La contribution de chacune des composante est quand à elle illustrée sur la figure 2.9(d). Les constantes choisies pour cette figure sont $\mathrm{K}_{ps} = 10, \mathrm{C}_{ls} = 5, \mathrm{K}_{diff} = 1$ et $\sigma_\alpha^2 = \frac{1}{30}$. Le choix $\sigma_\alpha^2 = \frac{1}{30}$ correspond à une surface lisse tandis que les valeurs de K_{ps} et C_{ls} et K_{diff} induisent une réflexion spéculaire non négligeable.

2.2.5 Les modèles de Shafer et Healey

Le modèle de Shafer [172] exprime la radiance d'un patch de surface comme la somme d'une composante diffuse et d'une composante spéculaire[1] :

$$\begin{aligned} \mathrm{Ir}(\lambda, \theta_i, \theta_r, g) &= \mathrm{Ir}_{diff}(\lambda, \theta_i, \theta_r, g) + \mathrm{Ir}_{spec}(\lambda, \theta_i, \theta_r, g) && (2.18) \\ &= m_{diff}(\theta_i, \theta_r, g) c_{diff}(\lambda) + m_{spec}(\theta_i, \theta_r, g) c_{spec}(\lambda) && (2.19) \end{aligned}$$

où g représente l'angle entre les vecteurs $-\overrightarrow{k_1}$ et $\overrightarrow{k_r}$ (figure 2.6).

Ce modèle est donc essentiellement qualitatif dans la mesure où il ne précise pas la valeur des fonctions m_{diff}, m_{spec} et c_{diff}, c_{spec}. Il fait toutefois deux hypothèses extrêmement fortes sur la nature de la réflexion :

1. L'irradiance incidente à un capteur est supposée pouvoir être décomposée en la somme de deux termes l'un exprimant une réflexion diffuse et l'autre une réflexion spéculaire. Une telle décomposition de l'irradiance en la somme de deux termes correspondant à deux phénomènes physiques différents a été validé par Beckmann (section 2.2.2). Notons également que Torrance-Sparrow (section 2.2.3) et Nayar (section 2.2.4) font les mêmes hypothèses.
2. La seconde hypothèse, sans doute plus discutable, est que l'irradiance de chacun des termes de la somme peut être décomposée en un produit de deux termes l'un dépendant uniquement de facteurs géométriques (m_{diff} et m_{spec}) l'autre dépendant uniquement de la longueur d'onde λ (c_{diff} et c_{spec}). Une telle hypothèse a été reprise par Nayar dans le cadre de l'unification des modèles de Torrance-Sparrow et Beckmann-Spizzichino. Notons toutefois que cette supposition n'a rien d'évident au vue de l'équation de l'irradiance de Beckmann (équation 2.5).

Un autre modèle du a Healey [93] exprime approximativement la même idée. Selon Healey, la réflectance d'un matériau peut être approximée par la formule suivante :

$$R(\lambda, g) = \begin{cases} M_{spec}(g) C_{spec}(\lambda) & \text{pour les métaux} \\ M_{spec}(g) C_{spec}(\lambda) + M_{diff}(g) C_{diff}(g) & \text{pour les diélectriques inhomogènes} \end{cases} \quad (2.20)$$

où g représente les facteurs géométriques tels que θ_i, θ_r et ψ_r.

On retrouve donc bien une décomposition en une réflectance spéculaire et Lambertienne avec pour chacun des termes une sous décomposition en un produit d'un terme dépendant de la géométrie et d'un terme ne dépendant que de la longueur d'onde. Toutefois, la composante Lambertienne est supprimée pour les métaux. Cette suppression s'explique par le modèle usuellement accepté pour expliquer la réflexion Lambertienne (section 2.2.1). En effet, celle-ci est issu de réflexions

[1] Shafer préfère les termes de réflexion de l'interface et de l'intérieur du matériau aux termes de réflexion spéculaire et Lambertienne ou diffuse. Nous garderons toutefois ces dernières notations afin de rester consistants avec les notations précédentes

à l'intérieur du matériau. Les métaux possédant une forte conductivité ont un fort coefficient d'absorption K_0 (équation 2.1), si bien que l'onde incidente pénètre peu dans le matériau et la réflexion reste un phénomène de surface décrit par le terme spéculaire. Notons toutefois que les deux modèles sont sensiblement équivalents ; en effet rien n'empêche de poser m_{diff} ou c_{diff} à 0 dans le modèle de Shafer dans le cas d'une réflexion sur un métal.

2.2.6 Conclusion

Nous avons abordé dans cette section 3 modèles décrivant les différents types de réflexion (sections 2.2.1, 2.2.2 et 2.2.3) et 2 modèles de synthèse (sections 2.2.4 et 2.2.5). Le modèle de Lambert (section 2.2.1) permet de décrire les réflexions à l'intérieur d'un matériau. L'utilisation de ce modèle n'est pertinente que si une partie significative de l'onde incidente pénètre dans le matériau avant de ressortir de celui-ci. Ce modèle est donc plus adapté aux matériaux diélectriques inhomogènes. Les modèles de Beckmann et Torrance (sections 2.2.2 et 2.2.3) décrivent quand à eux la réflexion à la surface du matériau. Ces modèles sont donc adaptés au cas où la plus grande partie de l'onde est réfléchie par la surface (cas des matériaux conducteurs) ou dans le cas où la partie de l'onde qui pénètre dans le matériau n'est pas réfléchie vers la surface (cas des matériaux homogènes). La principale différence entre les modèles de Beckmann et Torrance tient à ce que le modèle de Beckmann reste valide pour des matériaux lisses et rugueux alors que le modèle de Torrance ne peut s'appliquer pour des matériaux parfaitement lisses. La figure 2.10 illustre les différents domaines d'applications de ces modèles. Notons toutefois que les matériaux ne sont généralement ni parfaitement conducteurs ou isolants ni parfaitement homogènes. La décomposition illustrée par cette figure reste donc schématique.

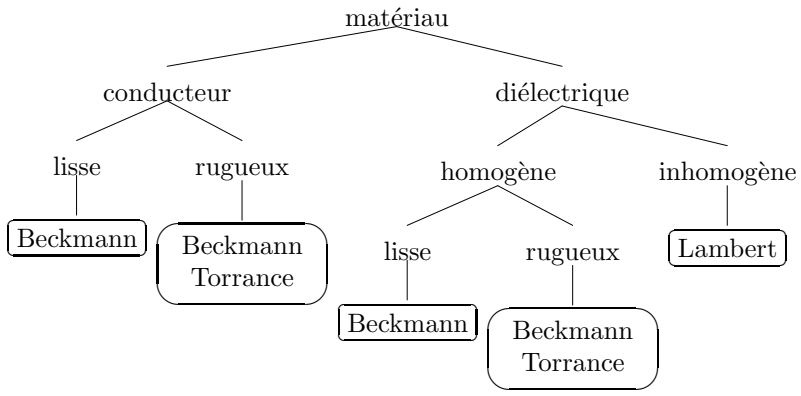


FIG. 2.10 – Utilisation des différents modèles de réflexion en fonction du type de matériau

Les modèle de Nayar (section 2.2.4) et Safer (section 2.2.5) décrivent respectivement la réflexion d'un point de vue quantitatif et qualitatif. Ces modèles font une synthèse des modèles précédemment décris. Le modèle de Nayar en particulier décrit le phénomène de réflexion comme une somme de réflexions issues des modèles de Beckmann, Torrance et Lambert. Les coefficients affectés à chaque modèle de base dans le modèle de Nayar dépendent du type de matériau. Les modèles de Shafer ou Healey sont des modèles qualitatif et ne nécessitent donc pas de tels ajustements. Il faut toutefois faire attention aux domaines de validités des modèles de base sur lesquels s'appuient ces modèles.

Par exemple, l'utilisation de la composante Lambertienne pour des diélectriques n'est plus valide pour des sources avec un angle d'incidence rasant (section 2.2.1).

2.3 Vision des couleurs

Les modèles de réflexion décrivent les modifications du spectre d'une source lumineuse induites par le phénomène de réflexion. Il nous reste toutefois à définir les phénomènes physiologiques qui associent une sensation colorée à un spectre. La partie optique de la couleur se limite à l'oeil qui comprend le cristallin et la rétine (figure 2.11). Le cristallin agit comme une lentille épaisse qui concentre les rayons lumineux sur la rétine. L'image inversée ainsi obtenue n'est nette que dans la partie centrale de la rétine appelée fovéa.

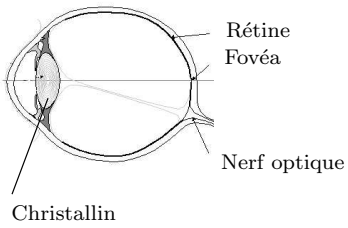


FIG. 2.11 – *Schéma simplifié de l'oeil*

Les récepteurs neuronaux tapissant la rétine sont de deux types :
- les bâtonnets pour la vision scotopique (à faible luminance i.e. en dessous de 10^{-6} candela (cd) par m^2) sont achromatiques et se rencontrent principalement dans les zones périfovéales et périphériques de la rétine ;
- les cônes utilisés en vision photopique (luminance élevée, au dessus de $10cd/m^2$) sont présents principalement en région fovéale et parafovéale.

Les cônes se répartissent en trois grandes familles L (long), M (medium) et S (short) correspondant respectivement à leur sensibilité aux longues, moyennes et grandes longueurs d'ondes. On peut donc considérer que les cônes L sont sensibles au rouge alors que les cônes M sont sensibles au vert et les cônes S au bleu (Fig. 2.3).

L'excitation d'un cône dépend de la puissance du spectre incident pour les longueurs d'ondes auxquelles celui-ci est sensible. La sensation colorée ressentie lors de la perception d'une onde lumineuse de spectre $\xi(\lambda)$ peut donc être modélisée par un vecteur $3D$ $\vec{C}$ représentant la réponse des cônes L, M et S :

$$c_i = \int_{\lambda_{min}}^{\lambda_{max}} s_i(\lambda)\xi(\lambda)d\lambda, \quad i = 1, 2, 3$$

où $\vec{C} = (c_1, c_2, c_3)^t$ tandis que la fonction $s_i(\lambda)$ et l'intervalle $[\lambda_{min}, \lambda_{max}]$ représentent respectivement la sensibilité du i^{e} type de cône et l'intervalle en dehors duquel la sensibilité des cônes est nulle (typiquement $[360nm, 830nm]$). Le symbole $\vec{C}^t$ représente la transposée du vecteur $\vec{C}$.

Si nous discrétisons l'intervalle $[\lambda_{min}, \lambda_{max}]$ en N valeurs, $\{\lambda_1, \ldots, \lambda_N\}$ uniformément réparties

sur $[\lambda_{min}, \lambda_{max}]$, l'intégrale précédente peut se réécrire sous forme matricielle :

$$\vec{C} = S^t \vec{\xi} \tag{2.21}$$

où $\vec{\xi}$ est à présent un vecteur de N valeurs : $(\xi(\lambda_1), \ldots, \xi(\lambda_N))$ et S une matrice $N \times 3$ représentant la sensibilité des cônes :

$$S = [\vec{s_1}, \vec{s_2}, \vec{s_3}] = \begin{pmatrix} s_1(\lambda_1) & s_2(\lambda_1) & s_3(\lambda_1) \\ & \vdots & \\ s_1(\lambda_N) & s_2(\lambda_N) & s_3(\lambda_N) \end{pmatrix}$$

2.3.1 Définition des espaces couleur

L'équation 2.21 nous permet de définir une couleur à partir des fonctions de sensibilité des cônes. Toutefois, une représentation équivalente d'une couleur pourrait être obtenue à partir de n'importe quelle transformation du vecteur $\vec{C}$ par une matrice inversible. De plus, l'utilisation de la matrice S est peu pratique puisque les fonctions de sensibilité des cônes sont difficiles à mesurer. On utilise donc une représentation des couleurs équivalente. On considère pour cela trois sources lumineuses $\vec{p_1}, \vec{p_2}$ et $\vec{p_3}$ *colorimétriquement indépendante*, c'est à dire telles que $S^t\vec{p_1}, S^t\vec{p_2}, S^t\vec{p_3}$ forment un système libre de $\mathbb{R}^3$. De telles sources sont appelées des *primaires*. Les vecteurs $S^t\vec{p_1}, S^t\vec{p_2}, S^t\vec{p_3}$ formant une base de $\mathbb{R}^3$, une couleur $\vec{C}$ associée à un spectre f peut être représentée par un vecteur $\vec{a}(\vec{\xi})$ tel que :

$$\vec{C} = S^t\vec{\xi} = a_1(\vec{\xi})S^t\vec{p_1} + a_2(\vec{\xi})S^t\vec{p_2} + a_3(\vec{\xi})S^t\vec{p_3}.$$

Cette transformation nous permet de passer d'une représentation déterminée par la sensibilité des cônes (codée par la matrice S) à une représentation déterminée par la sensation colorée induite par les trois sources lumineuses $\vec{p_1}, \vec{p_2}$ et $\vec{p_3}$. Les composantes $(a_1(\vec{\xi}), a_2(\vec{\xi}), a_3(\vec{\xi}))$ du vecteur $\vec{a}(\vec{\xi})$ sont mesurées à l'aide d'expériences d'*appariement* (Fig. 2.12). Un observateur «virtuel» appelé l'*Observateur standard* regarde deux spots lumineux séparés par un angle α. L'un de ces spots provient du spectre $\vec{\xi}$ tandis que l'autre est issue de la superposition des spectres $\vec{p_1}, \vec{p_2}$ et $\vec{p_3}$. L'observateur doit modifier l'intensité des sources lumineuses $\vec{p_1}, \vec{p_2}$ et $\vec{p_3}$ jusqu'à ce que les deux spots lumineux apparaissent visuellement identiques. Les coefficients réglant l'intensité de chacune des source correspondent aux composantes $(a_1(\vec{\xi}), a_2(\vec{\xi}), a_3(\vec{\xi}))$ du vecteur $\vec{a}(\vec{\xi})$.

Nous pouvons définir à partir de la base canonique $(\vec{e_i})_{i\in\{1,\ldots,N\}}$ de $\mathbb{R}^N$, les vecteurs $\vec{a}(\vec{e_i})$ associés à chaque spectre $\vec{e_i}$. On peut alors montrer [174] qu'une couleur peut être représentée de façon équivalente par un triplé $\vec{C'}$ défini par :

$$\vec{C'} = A^t\vec{\xi}$$

où A est la matrice $N \times 3$ telle que chaque ligne correspond à un vecteur $\vec{a}(\vec{e_i})$.

La matrice A est appelée la *matrice d'appariement* et chaque colonne de A une *fonction d'appariement*.

Le triplé $\vec{C'}$ définit une couleur tandis que les paramètres définissant ce triplé définissent l'espace couleur. Celui-ci est donc défini par :

1. Les spectres des trois sources lumineuses $\vec{p_1}$, $\vec{p_2}$ et $\vec{p_3}$;
2. l'angle α entre les deux spots.

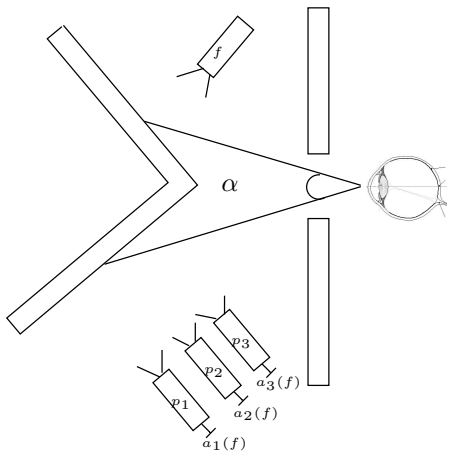


FIG. 2.12 – *Expérience d'appariement de couleurs*

Toute modification des spectres $\vec{p_1}$, $\vec{p_2}$ et $\vec{p_3}$ ou de l'angle α définit un autre espace couleur. Si nous utilisons par exemple trois primaires $(\vec{q_1}, \vec{q_2}, \vec{q_3})$ différentes de $(\vec{p_1}, \vec{p_2}, \vec{p_3})$, nous obtenons une matrice d'appariement B liée à la matrice A par [189] :

$$B^t = TA^t \quad \text{avec } T = (A^tQ)^{-1}$$

où Q est la $N \times 3$ matrice $[\vec{q_1}, \vec{q_2}, \vec{q_3}]$. Notez que la matrice T est une matrice 3×3 inversible.

Ainsi, étant donné un spectre $\vec{\xi}$ et son expression $\vec{C_1}$ et $\vec{C_2}$ dans les espaces couleur respectivement définis par $(\vec{p_1}, \vec{p_2}, \vec{p_3})$ et $(\vec{q_1}, \vec{q_2}, \vec{q_3})$, nous avons :

$$\vec{C_2} = B^t\vec{\xi} = TA^t\vec{\xi} = T\vec{C_1}.$$

La transformation d'un espace à un autre se fait donc à l'aide de la matrice 3×3, T qui peut s'interpréter comme une matrice de changement de base.

Il existe de nombreux espaces couleurs souvent construits pour répondre à des besoins spécifiques. Ceux-ci peuvent être construits directement à partir de trois sources lumineuses et des expériences d'appariement ou être définis à partir d'une transformation d'un espace existant. Notez que ces transformations ne sont pas obligatoirement linéaires.

A l'exception de l'algorithme décrit dans la section 2.4.2, toutes les méthodes proposées dans ce mémoire sont indépendantes des propriétés spécifiques des différents espaces couleurs. Nous nous contenterons donc de citer brièvement les principaux espaces couleurs définis par la *Commission Internationale de l'Éclairage*(CIE). Notons toutefois que l'utilisation d'algorithmes fonctionnant dans n'importe quel espace couleur ne permet pas de négliger l'influence de ces derniers. En effet, le choix de l'espace couleur dans lequel va fonctionner l'algorithme influence le résultat. Le choix d'un espace couleur est donc dans ce cas un des paramètre implicite de l'algorithme.

Les principaux espaces définis par la CIE sont [19, 174, 31] :

– les espaces CIE RGB 1931 et 1964 [116, 52]. Ces espaces sont définis à partir de trois primaires rouge, verte et bleu. L'espace CIE RGB 1931 correspond à un angle α de 2 degrés tandis

que l'espace CIE RGB 1964 est défini pour un angle α de 10 degré. Un des principaux inconvénients de ces espaces est que certaines couleurs peuvent avoir une composante négative et sont donc non représentables en *synthèse additive*. Autrement dit, ces couleurs ne peuvent pas être visualisées à l'aide de la superposition des trois primaires ;

- l'espace CIE XYZ 1931 [134, 202, 72] permet de représenter toute couleur visible à l'aide de coordonnées positives. La composante Y de cet espace code la luminance ;
- les espaces CIE 1976 Luv et Lab [52, 202, 182] sont déduits de l'espace CIE XYZ à l'aide d'une transformation non linéaire. La distance euclidienne entre deux couleurs dans ces espaces approxime la perception humaine [150] des distances entre couleur. On parle alors d'espace *uniforme* ou *pseudo uniforme*.

2.4 Segmentation d'une image en matériaux

La *segmentation* d'une image en matériaux consiste à créer une partition de l'image en *régions* telle que chaque région soit composée d'un seul matériau. La segmentation d'une image est usuellement définie comme la partition de celle-ci en régions, chaque région codant une partie d'un objet de la scène. Une définition formelle d'un algorithme de segmentation a été donnée par Horowitz et Pavlidis [106, 105] en 1975.

Définition 1 *Soit X le domaine de l'image et f la fonction qui associe à chaque pixel une valeur $f(x, y)$. Si nous définissons un prédicat P sur l'ensemble des parties de X, la segmentation de X est définie comme une partition de X en n sous-ensembles $\{R_1, ..., R_n\}$ tels que :*

1. $X = \bigsqcup_{i=1}^{m} R_i$
2. $\forall i \in \{1, \ldots, n\}$ R_i *est connexe.*
3. $\forall i \in \{1, \ldots, n\} P(R_i) = vrai$
4. $\forall i, j \in \{1, \ldots, n\}^2$ / R_i *est adjacent à* R_j *et* $i \neq j \Rightarrow P(R_i \cup R_j) = faux$

où $\bigsqcup$ *représente une union d'ensemble disjoints.*

Le prédicat P est utilisé pour tester l'homogénéité des ensembles R_i. Ces sous-ensembles constituent les régions de l'image. Une segmentation de l'image est donc sa décomposition en un ensemble de régions homogènes, le critère d'homogénéité P restant à déterminer. Zucker [206] a résumé les conditions de la définition 1 comme suit : la première condition implique que tout pixel de l'image appartienne à une région et une seule. Cela signifie que l'algorithme de segmentation ne doit pas se terminer avant d'avoir traité tous les points. La seconde condition implique que chaque région doit être constituée d'un ensemble connexe de pixels. La connexité des régions est induite par le voisinage défini sur l'image. La troisième condition implique que chaque région doit être homogène. Enfin, la quatrième condition est une condition de maximalité indiquant que la fusion de deux régions voisines ne doit pas donner une région homogène. Il est important de remarquer que le nombre n de régions formant la partition de l'image reste indéterminé. Il peut donc exister plusieurs segmentations possibles pour un prédicat P donné.

Intuitivement un "bon" prédicat P doit nous fournir une région pour chaque objet de l'image. Toutefois, la notion d'objet est généralement assez vague si bien que la problématique de la segmentation est par essence mal définie. La segmentation en matériaux au moins prise comme prétraitement peut permettre une meilleure définition du problème. La restriction de la problématique

de la segmentation à une segmentation en matériaux possède en effet deux avantages fondamentaux :

1. Tout d'abord le but a atteindre par l'algorithme peut être décrit précisément.
2. De plus, notre décomposition de l'image en matériaux peut s'appuyer sur les modèles physiques vus dans la section 2.2 pour interpréter les valeurs des pixels.

Ce dernier avantage a de nombreuses conséquences pratiques. En effet le prédicat P utilisé en segmentation est souvent défini à l'aide des distances entre les couleurs de pixels. On peut par exemple imposer qu'une région soit homogène si la couleur de chacun de ses pixels est à une certaine distance de la couleur moyenne de la région. On peut également imposer que le gradient moyen le long de chacune des frontières de la partition dépasse un certain seuil. Toutefois, nous avons vu dans la section 2.2 que deux pixels adjacents correspondant au même matériau et au même objet peuvent avoir des intensités très différentes (voir par exemple le profil d'intensité sur la figure 2.9). Ces limitations du processus de segmentation sont essentiellement dues à l'absence d'un modèle nous permettant d'interpréter la distance entre deux pixels adjacents. La segmentation en matériaux permet de ce baser sur des modèles physiques permettant d'interpréter les valeurs des pixels et leur différences. Ce processus ne fournit pas une segmentation en objets mais peut être utilisée comme première étape d'un processus de segmentation plus complet fournissant un ensemble de régions pas simplement homogènes mais composées d'un unique matériau que l'on peu éventuellement identifier. Les processus de plus haut niveau sont dans ce cas grandement simplifiés. Notons toutefois que la segmentation en matériau ne permet à elle seule de segmenter correctement des images fortement texturées. Il faut pour ce type d'image combiner les informations issues des modèles de réflexion avec des méthodes permettant de caractériser les textures.

La majorité des algorithmes de segmentation en matériaux sont basés sur les modèles de Shafer [172] ou Healey [93] (section 2.2.5). Nous allons dans la section 2.4.1 établir les fondements de ces méthodes avant de présenter notre contribution à ce domaine en section 2.4.2.

2.4.1 Méthodes de Segmentation basées sur les modèles de Shafer et Healey

Le modèle de Shafer [172](section 2.2.5) exprime l'irradiance sur un capteur de la caméra par la somme d'une composante Lambertienne et d'une composante Spéculaire :

$$\mathrm{Ir}(\lambda, \theta_i, \theta_r, g) = m_{diff}(\theta_i, \theta_r, g) c_{diff}(\lambda) + m_{spec}(\theta_i, \theta_r, g) c_{spec}(\lambda).$$

Si nous considérons que les angles θ_i, θ_r et g sont fonctions de la position (x, y) du pixel cette équation se réécrit :

$$\mathrm{Ir}(\lambda, x, y) = m_{diff}(x, y) c_{diff}(\lambda) + m_{spec}(x, y) c_{spec}(\lambda). \tag{2.22}$$

Notez que dans un tel modèle les valeurs respectives de m_{diff} et c_{diff} ainsi que celles de m_{spec} et c_{spec} ne peuvent être déterminées qu'à un facteur multiplicatif prés. Shafer pose donc, sans perte de généralité : $0 \leq m_{diff}(x, y) \leq 1$ et $0 \leq m_{spec}(x, y) \leq 1$.

Étant donné une caméra couleur, la couleur d'un pixel $\vec{C}(x, y)$ se déduit de l'irradiance sur ses capteurs par :

$$\forall i \in \{1, 2, 3\} \quad C_i(x, y) = \int_{\lambda_{min}}^{\lambda_{max}} \mathrm{Ir}(\lambda, x, y) s_i(\lambda) d\lambda$$

où $C_i(x,y)$ représente la $i^{\text{ème}}$ composante de $\vec{C}(x,y)$ et $(s_i(\lambda))_{i\in\{1,2,3\}}$ représentent la sensibilité de la caméra aux 3 composantes chromatiques (usuellement le rouge, le vert et le bleu). Les deux longueurs d'ondes λ_{min} et λ_{max} représentent les deux bornes de sensibilités de la caméra ($\lambda_1 \approx 360nm$ et $\lambda_2 \approx 830nm$). Ces bornes correspondent au spectre visible par l'oeil humain (section 2.3).

Si nous reprenons l'équation 2.22 nous obtenons :

$$\forall i \in \{1,2,3\} \quad C_i(x,y) = m_{diff}(x,y)\int_{\lambda_{min}}^{\lambda_{max}} c_{diff}(\lambda)s_i(\lambda)d\lambda + m_{spec}(x,y)\int_{\lambda_{min}}^{\lambda_{max}} c_{spec}(\lambda)s_i(\lambda)d\lambda.$$

Donc :

$$\vec{C}(x,y) = m_{diff}(x,y)\vec{C}_{diff} + m_{spec}(x,y)\vec{C}_{spec} \tag{2.23}$$

avec

$$\forall i \in \{1,2,3\} \quad \left(\vec{C}_{diff}\right)_i = \int_{\lambda_{min}}^{\lambda_{max}} c_{diff}(\lambda)s_i(\lambda)d\lambda \text{ et } \left(\vec{C}_{spec}\right)_i = \int_{\lambda_{min}}^{\lambda_{max}} c_{spec}(\lambda)s_i(\lambda)d\lambda.$$

L'équation 2.23 nous montre donc que pour un matériau l'information couleur *a priori* tridimensionnelle est en fait bidimensionnelle puisque l'ensemble des couleurs d'un matériau est contenu dans le plan défini par $\vec{C}_{diff}$, $\vec{C}_{spec}$. Plus précisément, puisque nous supposons m_{diff} et m_{spec} bornés entre 0 et 1, l'ensemble des couleurs d'un matériau est contenu dans le parallélogramme défini par les vecteurs $\vec{C}_{diff}$ et $\vec{C}_{spec}$ (figure 2.13). Notons que dans le cas d'un seul illuminant un des coins du parallélogramme doit être l'origine de l'espace (cas où $m_{diff} = m_{spec} = 0$).

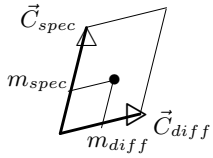


FIG. 2.13 – Représentation d'une couleur dans le parallélograme de Shafer.

Le modèle de Healey est basé sur une décomposition supplémentaire en métaux et diélectriques. On obtient en en effet à partir de l'équation 2.20 :

$$\vec{C}(x,y) = \begin{cases} m_{spec}(x,y)\vec{C}_{spec} & \text{pour les métaux} \\ m_{spec}(x,y)\vec{C}_{spec} + m_{diff}(x,y)\vec{C}_{diff} & \text{pour les diélectriques inhomogènes} \end{cases}$$

Donc l'ensemble des couleurs d'un métal peut être décrit par une droite. Alors qu'*a priori* l'ensemble des couleurs d'un diélectriques inhomogène peut être d'écrit par un parallélogramme.

Klinker [94] a affiné cette description en remarquant que dans le cas d'un diélectrique non homogène les points tels que le vecteur $\vec{C}_{spec}$ n'est pas négligeable correspondent à des zones de l'image fortement illuminées. De tels points correspondent à des zones très localisées dans l'image

pour lesquelles on peut négliger les variations de la composante diffuse. Si nous nous référons au modèle de Nayar (section 2.2.4). Un telle approximation, revient à négliger les variations de la composante Lambertienne lorsque le lobe spéculaire ou le pic spéculaire n'est pas négligeable.

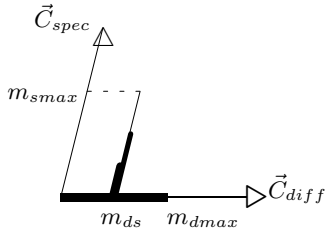


FIG. 2.14 – Répartition des couleurs dans le parallélogramme de Shafer selon Klinker.

La répartition des couleurs dans le parallélogramme de Shafer n'est donc pas uniforme mais suit une distribution similaire à celle représentée sur la figure 2.14. L'ensemble des couleurs d'un diélectrique non homogène dans des régions non hautement éclairées peut donc être décrit uniquement par le vecteur $\vec{C}_{diff}$. Dans le cas d'une zone de l'image hautement illuminée la composante de ce vecteur sur $\vec{C}_{diff}$ reste constante (valeur m_{ds} sur la figure 2.14) alors que les variations se produisent sur le vecteur $\vec{C}_{spec}$. L'histogramme des couleurs d'un matériau a donc la forme d'un T renversé et légèrement distordu. On peut également obtenir un histogramme en L si m_{ds} est égal à la valeur maximum m_{dmax} sur l'axe $\vec{C}_{diff}$. Une étude de l'illumination d'un cylindre à amené Klinker [94] à conclure que le seuil m_{ds} ne pouvait se trouver que dans la seconde moitié de l'intervalle des variations $[0, m_{dmax}]$. Cette hypothèse confirmée par des expériences sur des images réelles est appelée *l'hypothèse des 50% supérieurs*. Nous appellerons respectivement les lignes définies par $\vec{C}_{diff}$ et $m_{ds}\vec{C}_{diff} + \vec{C}_{spec}$ la *ligne diffuse* et la *ligne spéculaire*.

Cette modélisation de l'ensemble des couleurs d'un matériau nous permet d'interpréter l'ensemble des couleurs du voisinage d'un point. En effet, étant donné un pixel de l'image et une fenêtre centrée sur celui-ci, l'ensemble des couleurs des pixels contenus dans cette fenêtre peut avoir une dimension 0, 1, 2 ou 3. On parlera donc de fenêtre de dimension i pour indiquer une fenêtre dans l'image telle que l'ensemble des couleurs des pixels contenus dans cette fenêtre a dans l'espace couleur choisi une dimension i. La dimension de cet ensemble de couleurs peut être testée en utilisant les valeurs propres de la matrice de covariance de l'ensemble de couleurs (section 3.2.2, équation 3.2).

- dans le cas d'une fenêtre de dimension 0, le point considéré ne peut se trouver que sur un matériau avec une faible courbure ou sur une zone mal éclairée de ce même matériau ;
- dans le cas ou la fenêtre a pour dimension 1, la couleur du pixel se trouve sur la ligne diffuse ou sur la ligne spéculaire d'un matériau. La fenêtre peut également inclure deux matériaux de telle façon que les couleurs de l'un deux sont situées sur la ligne diffuse alors que celles de l'autre corresponde à un seul point de la ligne diffuse. Le fenêtre intersecte alors une zone plate ou peu éclairée du second matériau ;
- si la fenêtre à pour dimension 2 la fenêtre est soit incluse dans un seul matériau avec les lignes diffuse et spéculaire soit intersecte deux matériaux de telle façon qu'une seule des deux droites de chaque matériau est comprise dans la fenêtre ;

- dans le cas de fenêtre de dimension 3, le pixel peut se trouver :
  - soit dans un seul matériau. La troisième dimension est dans ce cas créée par le bruit d'acquisition,
  - soit à l'intersection de deux matériaux avec les deux droites diffuses et spéculaires pour l'un et une seule de ces droites pour l'autre,
  - enfin la fenêtre peut également avoir intersecté une frontière entre au moins 3 matériaux.

Le cas ou le pixel est dans une fenêtre de dimension 3 est évidemment celui pour lequel on a le moins d'information. Il est toutefois remarquable de pouvoir faire de telles hypothèses à partir d'une fenêtre centrée sur un pixel.

Klinker [94] utilise cette classification pour décomposer l'image en matériaux. Schématiquement, sa méthode de segmentation se déroule de la façon suivante :

1. décomposer l'image en fenêtres et étiqueter chaque fenêtre en fonction de la dimension de l'ensemble couleur correspondant ;
2. fusionner les fenêtres adjacentes de même dimension si l'ensemble obtenu a la même dimension ;
3. affiner la segmentation de toutes les régions de dimension 1 par une croissance de région lancée au centre de la région. L'ensemble des couleurs de chaque région de dimension 1 correspond alors à une ligne diffuse ou à une ligne spéculaire ;
4. fusionner des régions de dimension 1 entre elles si l'ensemble des couleurs obtenues à une configuration en T ou en L. Ces fusions sont contraintes par l'hypothèse des 50% supérieurs. Les régions obtenues ont pour dimension 2 ;
5. lancer un algorithme de croissance de régions dans chaque région générée à l'étape précédente. Cet algorithme agrège à la région les pixels des régions de dimension 2 compatibles avec la configuration des couleurs de la région initiale.

Notons que les pixels dont la fenêtre associée a pour dimensions 0 et 3 ne sont pas directement utilisés par l'algorithme. En effet l'information liée à ces pixels est difficilement exploitable et ces pixels sont absorbés par les étapes de croissance de région dans des régions de dimension 1 ou 2.

La méthode présentée par Klinker utilise une approche ascendante. Healey [91] a présenté une méthode comparable basée sur une approche descendante.

La méthode d'Healey néglige initialement la composante spéculaire des diélectriques inhomogènes. On a donc pour les métaux comme pour les diélectriques une équation de la forme :

$$\mathrm{Ir}(\lambda, x, y) = m(x, y)c(\lambda)$$

où m représente m_{spec} pour les métaux et m_{diff} pour les diélectriques (équation 2.20). De même, c représente C_{spec} pour les métaux et C_{diff} pour les diélectriques.

Étant données les fonctions de sensibilité $(s_i)_{i\in\{1,2,3\}}$ de la caméra, le vecteur couleur $\vec{C} = (C_1, C_2, C_3)$ d'un pixel est donné par :

$$\forall i \in \{1,2,3\} \quad C_i = \int_{\lambda_{min}}^{\lambda_{max}} m(x,y)c(\lambda)s_i(\lambda)d\lambda = m(x,y)K_i \text{ avec } K_i = \int_{\lambda_{min}}^{\lambda_{max}} c(\lambda)s_i(\lambda)d\lambda.$$

Si nous notons par $\vec{C}_{mat}$ le vecteur (K_1, K_2, K_3), la couleur $\vec{C}(x, y)$ d'un pixel est donc égale à :

$$\vec{C}(x, y) = m(x, y)\vec{C}_{mat}.$$

L'ensemble des couleurs d'un matériau appartient donc à une droite de vecteur directeur $\vec{C}_{mat}$.

Si nous normalisons les coordonnées de chaque pixel par sa norme dans L_2 (norme Euclidienne) nous obtenons :

$$\forall i \in \{1,2,3\} \quad \frac{C_i}{\|\vec{C}(x,y)\|} = \frac{M(x,y)K_i}{\sqrt{m(x,y)^2\,(K_1^2+K_2^2+K_3^2)}} = \frac{K_i}{\|\vec{K}\|}.$$

Les coordonnées normalisées de la couleur d'un pixel sont donc indépendantes de la géométrie. Notons qu'un résultat similaire aurait été obtenu en utilisant la norme L_1 (valeur absolue) plutôt que la norme L_2. Toutefois, la norme L_2 permet de définir une distance entre les droites qui ne dépend que de l'angle formé par celles-ci. Inversement la norme L_1 conduit à la définition d'une distance dépendant simultanément de l'angle entre les deux droites et de l'angle que forme une de ces deux droites avec un des axes de coordonnée. La norme L_2 est donc utilisée préférentiellement à la norme L_1.

La segmentation d'une image se ramène donc à une reconnaissance de droites dans un espace couleur ou à la reconnaissance d'un ensemble de distributions normales dans l'espace normalisé par L_2. C'est cette dernière option que choisi Healey. Un matériau m_i est caractérisé par sa moyenne $\vec{\mu}_i$ et sa matrice de covariance Σ_i (section 3.2.2, équation 3.2). La distribution des couleurs dans le matériau est alors modélisée par la distribution normale $N(\vec{\mu}_i, \Sigma_i)$ définie par

$$p(\vec{C}/m_i) = \frac{1}{2\pi\sqrt{2\pi}\sqrt{|\Sigma_i|}} e^{-\frac{1}{2}(\vec{C}-\vec{\mu}_i)^t \Sigma_i^{-1}(\vec{C}-\vec{\mu}_i)}.$$

La normalisation et effectué en considérant plutôt la distribution $N(\overrightarrow{\hat{\mu}_i}, \widehat{\Sigma_i})$ avec :

$$\overrightarrow{\hat{\mu}_i} = \frac{\vec{\mu}_i}{\|\vec{\mu}_i\|} \text{ et } \widehat{\Sigma_i} = \frac{\Sigma_i}{\|\mu_i\|^2}.$$

Étant donnée la modélisation de chaque matériau, la classification d'une couleur $\vec{C}$ a un matériau parmi N s'effectue en utilisant les fonctions suivantes basées sur la théorie de la décision de Bayes [68] :

$$g_i(\overrightarrow{\hat{C}}) = \log(p(\overrightarrow{\hat{C}}/m_i)) + \log(p_i)$$

où $\overrightarrow{\hat{C}} = \frac{\vec{C}}{\|\vec{C}\|}$ et p_i représente la probabilité qu'une couleur prise au hasard appartienne à m_i.

Si nous supposons que tous les matériaux sont équitablement représentés, p_i devient indépendant de i et l'équation précédente peut se simplifier en :

$$g_i(\overrightarrow{\hat{C}}) = -\frac{1}{2}(\overrightarrow{\hat{C}} - \overrightarrow{\hat{\mu}_i})^t \widehat{\Sigma_i}^{-1}(\overrightarrow{\hat{C}} - \overrightarrow{\hat{\mu}_i}) - \frac{1}{2}\log(|\widehat{\Sigma_i}|). \tag{2.24}$$

La couleur $\vec{C}$ est alors affectée au matériau i pour lequel $g_i(\overrightarrow{\hat{C}})$ est maximum.

Étant donnée cette méthode de classification, Healey commence par appliquer un opérateur gradient sur les intensités de l'image. L'idée sous-jacente est de caractériser les zones correspondant à un seul matériau comme des zones de faible gradient. Notons toutefois que cette supposition est discutable. En effet, l'opérateur gradient va non seulement réagir à des changements de matériaux mais également à de brusques changements de géométrie sur un objet composé d'un seul matériau.

La présence de bruit dans l'image risque également de fournir de fortes valeurs de gradients pour des zones composées d'un seul matériau. Inversement, un lissage de l'image afin d'atténuer le bruit risque de diminuer la valeur du gradient sur des zones constituées de plusieurs matériaux.

A partir de l'image de gradient, Healey décompose l'image à l'aide d'un quadtree [54] et initialise une liste vide de matériaux. Tout noeud du quadtree couvrant une zone de l'image de faible gradient est supposé composée d'un seul matériau. Healey calcule donc sa couleur moyenne normalisée :

$$\overrightarrow{\hat{\mu}} = \frac{\vec{\mu}}{\|\vec{\mu}\|} \text{ avec } \mu = \sum_{(x,y)\in R} \vec{C}(x,y)$$

où R représente la région couverte par le noeud du quadtree.

La moyenne $\overrightarrow{\hat{\mu}}$ est alors comparée aux valeurs moyennes des N matériaux présents dans la liste à l'aide de l'équation 2.24. Si la valeur maximum de $g_i(\hat{\mu})$ est supérieure à un seuil T la région est affectée à un des matériaux de la liste. Sinon Healey considère que la région est composée d'un nouveau matériau et celui-ci est ajouté à la liste en utilisant les données de la région R.

La méthode de Healey segmente donc l'image en un ensemble de régions, tel que l'ensemble des couleurs de chaque région décrit une droite dans l'espace de couleurs. Il convient ensuite de fusionner les régions adjacentes composée d'un seul matériau telle que l'une des régions correspond à la droite diffuse tandis que l'autre correspond à la droite spéculaire. Schématiquement, une fusion est réalisée entre deux régions si :

1. Les couleurs des pixels de la région supposée correspondre à la ligne spéculaire ont des composantes supérieures ou égales à celle de la région correspondant à la ligne diffuse.
2. Les droites des deux régions se coupent en un point vérifiant l'hypothèse des 50% supérieurs (Fig. 2.14).

La méthode de Healey repose donc sur une découpe récursive de l'image afin d'obtenir des patchs de surface composés d'un seul matériau. Malgré l'approche différente de celle de Klinker, ces deux méthodes sont basés sur les mêmes idées de base :

1. Exprimer la réflectance d'un matériau comme un produit d'un terme dépendant de la géométrie et d'un terme dépendant des longueurs d'ondes.
2. Utiliser cette modélisation pour décrire la structure de l'ensemble des couleurs d'un matériau.

Étant donnée une telle modélisation, l'algorithme de segmentation fusionne ou découpe des régions en s'appuyant sur les modèles décrivant les matériaux. De nombreuses adaptations de ces algorithmes ont été développées [93, 27, 96, 82, 173]. Toutefois, l'idée principale de ces algorithmes repose généralement sur les deux points mentionnés précédemment et notre contribution à ce type de méthode ne fait pas exception.

2.4.2 Notre contribution à la segmentation en matériaux

Notre contribution aux méthodes de segmentation en matériaux [110] est basée sur le modèle de Beckmann-Spizzichino (section 2.2.2). En effet, ce modèle est plus particulièrement conçu pour décrire la réflexion des métaux, principal objet de notre étude. Si nous reprenons l'expression de l'irradiance à l'entrée d'un capteur pour un métal parfait, nous avons (équations 2.7 et 2.8) :

$$\mathrm{Ir}_\infty(x,y,\lambda) = \begin{cases} \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\cos^2(\theta_i)e^{-g}\left(\frac{\kappa_{ps}}{\cos^2(\theta_r)}\rho_0^2 + \frac{\kappa_{ls}D^2 g}{cos\theta_r}e^{-v_{xy}^2\frac{T^2}{4}}\right) & \text{pour une surface lisse } (g << 1) \\ \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\kappa_{ls}\frac{e^{\left(\frac{-v_{xy}^2T^2}{4v_z^2\sigma_h^2}\right)}}{v_z^2\sigma_h^2}\cos(\theta_i)D^2 & \text{pour une surface rugueuse } (g >> 1). \end{cases}$$

où nous condensons dans la notation (x, y) l'ensemble des paramètres géométriques du point (x, y).

Nous obtenons en négligeant g dans le cas d'une surface lisse :

$$\mathrm{Ir}_\infty(x,y,\lambda) = \begin{cases} \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\cos^2(\theta_i)\frac{\kappa_{ps}}{\cos^2(\theta_r)}\rho_0^2 & \text{pour une surface lisse } (g \approx 0) \\ \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}\frac{E_o(\lambda)^2}{\lambda^2}\kappa_{ls}\frac{e^{\left(\frac{-v_{xy}^2T^2}{4v_z^2\sigma_h^2}\right)}}{v_z^2\sigma_h^2}\cos(\theta_i)D^2 & \text{pour une surface rugueuse } (g >> 1). \end{cases} \tag{2.25}$$

Nous avons vu dans la section 2.2.2 que les équations de Beckmann pouvaient être étendues à des métaux non parfaits en multipliant l'irradiance par la moyenne du module du coefficient de Fresnel $< FF^* > (\theta_i, \lambda)$ (équation 2.9). Toutefois, Healey à montré [93] que le coefficient de Fresnel pouvait être approximé dans le spectre visible par un produit de fonction $< FF^* > (\theta_i, \lambda) = \mathcal{F}_1(\theta_i)\mathcal{F}_2(\lambda)$. L'angle θ_l étant une fonction de la position (x, y) du pixel nous avons :

$$\mathrm{Ir}(\lambda, x, y) = \mathcal{F}_2(\lambda)\mathcal{F}_1(x, y)\mathrm{Ir}_\infty(x, y, \lambda). \tag{2.26}$$

Le seul terme dépendant de λ dans $\mathrm{Ir}(\lambda, x, y)$ est $\frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}E_0^2(\lambda)\frac{\mathcal{F}_2(\lambda)}{\lambda^2}$ (équation 2.25). On peut donc exprimer l'équation 2.26 de la façon suivante :

$$\mathrm{Ir}(x, y, \lambda) = G(x, y)\rho(\lambda)\varphi(\lambda)$$

où $\varphi(\lambda) = \frac{1}{2}\sqrt{\frac{\mu}{\epsilon}}E_0^2(\lambda)$ représente le flux d'énergie par unité de surface de l'onde incidente, $\rho(\lambda)$ notre facteur d'atténuation de l'énergie réfléchie fonction de la longueur d'onde et $G(x, y)$ notre facteur d'atténuation géométrique. Notez que $G(x, y)$ prend des valeurs différentes suivant le type de la surface ($g \approx 0$ ou $g >> 1$). Nous nous retrouvons donc avec un système d'équations similaires à ceux de Shafer et Healey (sections 2.2.5 et 2.4.1). Notre méthode de résolution sera toutefois quelque peu différente en raison des contraintes spécifiques de notre application.

Si nous utilisons une caméra à réponse logarithme, le vecteur couleur $\vec{C} = (C_1, C_2, C_3)$ d'un pixel est déterminé par :

$$\forall i \in \{1, 2, 3\} \quad \begin{cases} C_i(x, y) &= \Delta + \gamma E_i(x, y) \text{ avec} \\ E_i(x, y) &= \log(1 + \frac{1}{E_{noir}}\int_{\lambda_{min}}^{\lambda_{max}} s_i(\lambda)\mathrm{Ir}(x, y, \lambda)d\lambda) \end{cases} \tag{2.27}$$

où

- (x, y) représente les coordonnées d'un pixel de l'image et $\vec{C} = (C_1, C_2, C_3)$ le vecteur couleur du pixel dans l'espace RGB ;
- Δ et γ représentent respectivement le décalage («offset») et le gain de la caméra ;
- E_{noir} est l'irradiance minimum détectable ;
- $\mathrm{Ir}(x, y, \lambda)$ est l'irradiance incidente au capteur associée au pixel de coordonnées (x, y) ;

- $s_1(\lambda)$, $s_2(\lambda)$ et $s_3(\lambda)$ représentent respectivement la sensibilité spectrale de la caméra au rouge, au vert et au bleu ;
- λ_{min} et λ_{max} représentent les bornes d'intégration des capteurs de la caméra. Ces bornes correspondent au spectre visible (section 2.3).

Notons que l'équation 2.27 nous a été fournie par le manuel du constructeur de la caméra.

Pour de fortes ou moyennes irradiances (et donc de fortes intensités de pixel), nous pouvons approximer $\log(1+x)$ par $\log(x)$ dans l'équation 2.27. De même, pour de faibles irradiances, nous pouvons utiliser l'approximation $\log(1+x) \approx x$. Ces considérations permettent de simplifier l'expression de E dans l'équation 2.27 en fonction de l'intensité :

$$E_{fort}(x,y)_i = \log\left(\frac{G(x,y)}{E_{noir}}\right) + \log\left(\int_{\lambda_{min}}^{\lambda_{max}} s_i(\lambda)\phi(\lambda)\rho(\lambda)d\lambda\right) \tag{2.28}$$

$$E_{faible}(x,y)_i = \frac{G(x,y)}{E_{noir}} \int_{\lambda_{min}}^{\lambda_{max}} s_i(\lambda)\phi(\lambda)\rho(\lambda)d\lambda \tag{2.29}$$

pour tout i dans $\{1,2,3\}$.

L'espace colorimétrique antagoniste $H_1H_2H_3$ [20] est conçu pour simuler le mécanisme d'opposition des couleurs de la vision Humaine [19]. Cet espace se déduit de l'espace RGB par la transformation :

$$\begin{pmatrix} H_1 \\ H_2 \\ H_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 1 & -1 & 0 \\ -\frac{1}{2} & -\frac{1}{2} & 1 \end{pmatrix} \begin{pmatrix} C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} \Delta + \frac{\gamma}{3}\sum_{i=1}^{3} E_i(x,y) \\ \gamma\left(E_1(x,y) - E_2(x,y)\right) \\ \gamma\left(E_3(x,y) - \frac{E_1(x,y)+E_2(x,y)}{2}\right) \end{pmatrix}. \tag{2.30}$$

Notons qu'il existe de nombreux espaces antagonistes [74, 151, 171, 170]. Comme nous verrons ultérieurement, tout espace antagoniste permettrait d'obtenir des résultats équivalents à ceux obtenus dans l'espace $H_1H_2H_3$. Le choix d'un l'espace antagoniste ne peut donc pas dans ce cas s'appuyer sur une propriété spécifique à un de ces espaces puisque tous les espaces antagonistes vérifieront la propriété recherchée. L'espace $H_1H_2H_3$ a été initialement conçu [20] dans le cadre de la quantification d'images (section 3.2). La propriété qui a déterminé notre choix est l'efficacité du calcul de la transformation entre les espaces RGB et $H_1H_2H_3$. Cette transformation travaille en effet en coordonnée entières et peut s'effectuer à l'aide d'additions, de décalages et d'une seule division.

Si nous reprenons l'équation 2.28 et exprimons la couleur d'un pixel de forte ou moyenne luminance dans l'espace $H_1H_2H_3$, nous obtenons (équation 2.28) :

$$\begin{aligned} H_2(x,y) &= \gamma \log\left(\frac{\int_{\lambda_{min}}^{\lambda_{max}} s_1(\lambda)\phi(\lambda)\rho(\lambda)d\lambda}{\int_{\lambda_{min}}^{\lambda_{max}} s_2(\lambda)\phi(\lambda)\rho(\lambda)d\lambda}\right) \\ H_3(x,y) &= \gamma \log\left(\frac{\int_{\lambda_{min}}^{\lambda_{max}} s_3(\lambda)\phi(\lambda)\rho(\lambda)d\lambda}{\sqrt{\int_{\lambda_{min}}^{\lambda_{max}} s_1(\lambda)\phi(\lambda)\rho(\lambda)d\lambda \int_{\lambda_{min}}^{\lambda_{max}} s_2(\lambda)\phi(\lambda)\rho(\lambda)d\lambda}}\right). \end{aligned} \tag{2.31}$$

Nous obtenons de même pour un pixel de faible luminance (équations 2.29 et 2.30) :

$$\frac{H_2(x,y)}{H_3(x,y)} = \frac{\int_{\lambda_{min}}^{\lambda_{max}} (s_1(\lambda) - s2(\lambda))\phi(\lambda)\rho(\lambda)d\lambda}{\int_{\lambda_{min}}^{\lambda_{max}} \left(s_3(\lambda) - \frac{1}{2}(s_1(\lambda) + s_2(\lambda))\right)\phi(\lambda)\rho(\lambda)d\lambda}. \tag{2.32}$$

Notez que les équations 2.31 et 2.32 sont indépendantes de la géométrie de l'objet codé par la fonction $G(x,y)$. Donc dans le cas d'une moyenne ou d'une forte luminance (équation 2.31), l'ensemble des couleurs d'un matériau doit se projeter en un seul point dans le plan Euclidien H_2H_3. De même, dans le cas d'une faible luminance l'ensemble des pixels d'un matériau décrit une droite passant par l'origine dont la pente est définie par le rapport $\frac{H_2}{H_3}$.

Notre algorithme de segmentation est basé sur la remarque précédente. Étant donné un ensemble d'images pour lesquelles nous connaissons l'ensemble des matériaux devant intervenir :

1. Nous calculons la localisation du point de haute luminance dans le plan H_2H_3 à l'aide d'un ensemble d'images tests composées d'un seul matériau. Pour chaque image test, nous calculons la moyenne des pixels de haute luminance. Les coordonnées finales du point sont simplement définies comme la moyenne des coordonnées des points calculées sur les images tests.
2. Pour les faibles luminances, la pente de la droite dans le plan H_2H_3 est approximée en prenant la moyenne du premier vecteur propre de la matrice de covariance de l'ensemble des pixels de faible luminance sur chaque image test. La pente finale de la droite est approximée par la moyenne des pentes obtenues sur les images tests.

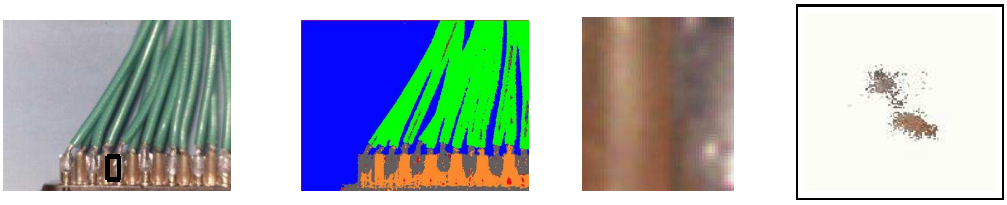
Cette phase d'apprentissage nous permet de calculer l'ensemble des points de haute luminance et les pentes de faible luminance pour chacun des matériaux devant apparaître dans les images à segmenter. La segmentation s'effectue à l'aide d'un seuil permettant de classifier les pixels de hautes ou faible luminance. Les pixels de haute luminance sont affectés au matériau dont le point de haute luminance dans l'espace H_2H_3 est le plus proche tandis que les points de faible luminance sont affectés au matériau dont la droite de faible luminance est la plus proche du point.

2.4.2.a Expériences

La figure 2.15 représente une segmentation d'une image de brasures par notre méthode. Cette image est composée de 4 matériaux : l'étain pour la brasure, le cuivre pour le support, le plastique entourant les câbles et le papier composant le fond de l'image. Notre algorithme a été validé sur 20 images composées des même matériaux.

Le vecteur directeur de la droite de faible luminance de chacun des matériaux est représenté dans l'espace RGB sur la Table 2.3. La figure 2.15(c) représente un grossissement du rectangle surimposé à la figure 2.15(a). Cette image est composée uniquement de deux matériaux (l'étain et le cuivre). La projection des couleurs des pixels de la figure 2.15(c) dans le plan H_2H_3 est représentée sur la figure 2.15(d). On observe bien deux nuages de points distincts correspondant aux deux matériaux.

On peut noter sur la figure 2.15(b) la présence de plusieurs pixels mal classifiés notamment sur les câbles (voir par exemples les câbles 1 et 3 en partant de la droite) et les brasures. Ces erreurs de classification se produisent pour les pixels de très forte luminance et peuvent être expliquées par deux phénomènes :



(a) Image originale

(b) Image segmentée

(c) Grossissement de (a)

(d) Projection dans H_2H_3

FIG. 2.15 – Segmentation d'une image de brasures. L'image (c) représente un grossissement de l'image (a) sur une zone ne comportant que deux matériaux. L'image (d) représente la projection de l'image (c) dans le plan H_2H_3.

	Vx	Vy	Vz
Fond	1.00000	0.93693	0.90385
Étain	1.00000	0.96767	0.93836
Cuivre	0.96186	1.00000	0.91242
Cable plastique	1.00000	0.90317	0.73708

TAB. 2.3 – Droites de faibles luminances dans l'espace RGB. Les vecteurs ont une norme égale à 1.

1. ces pixels de très haute luminance saturent les capteurs de la caméra ce qui produit un décalage des couleurs ;
2. ces pixels de haute luminance sont produits par le pic spéculaire qui renvoie le spectre incident en le modifiant peu. Étant donné un illuminant blanc tous ces pixels apparaissent donc blancs et sont par conséquent proches les uns des autres dans l'espace colorimétrique choisi et assez peu classifiables par leur couleur.

Afin de remédier à ce type de problème, nous avons décidé de ne pas classifier les pixels dont l'intensité dépasse un seuil fixé. Ces pixels non classifiés sont généralement assez dispersés dans l'image. Nous les classifions donc dans une étape de correction en calculant pour chaque pixel classifié un indice de confiance. L'indice que nous avons choisi est défini par :

$$Conf(\vec{C}) = 1 - \frac{d_1(\vec{C})}{d_1(\vec{C}) + d_2(\vec{C})} \in [0, 1]$$

où $d_1(\vec{C})$ et $d_2(\vec{C})$ représentent les distances Euclidiennes de $\vec{C}$ aux deux matériaux les plus proches. Ces distances sont évaluées différemment selon l'intensité de $\vec{C}$ (section 2.4.2).

Nous calculons ensuite pour chaque pixel non classifié, les matériaux présents dans son voisinage 8 connexe et l'indice moyen de confiance pour chaque matériau. Le pixel est classifié au matériau de plus fort indice de confiance. Son indice de confiance est fixé à la valeur de confiance moyenne de ce matériau dans le voisinage 8 connexe du pixel.

2.4.2.b Conclusions et perspectives

Notre méthode de segmentation est basée sur une expression de l'irradiance sur un capteur sous forme d'un produit de deux termes, l'un dépendant uniquement de la géométrie de la scène, l'autre dépendant uniquement des longueurs d'ondes. En ce sens, notre méthode est similaire aux méthodes basées sur les modèles de Shafer ou Healey [172, 93, 96, 91, 82, 92].

Toutefois, notre méthode s'appuie sur le modèle de Beckmann (section 2.2.2) qui propose une explication quantitative plutôt que qualitative du phénomène de réflexion. Ceci nous permet de mesurer plus précisément la liste des approximations nécessaires pour obtenir la séparation entre la géométrie et les longueurs d'ondes. Outre les restrictions définissant le domaine de validité du modèle de Beckmann (section 2.2.2), la principale restriction faite par notre méthode est la dichotomie des types de surface. En effet, nous ne nous intéressons qu'à des surfaces parfaitement lisses ($g \approx 0$) ou rugueuses ($g >> 1$). Dans tous les cas intermédiaires, le second terme de l'équation 2.7 ne peut être négligé et nous n'obtenons pas une expression de l'irradiance sous la forme d'un produit de deux termes l'un dépendant de la géométrie et l'autre des longueurs d'ondes.

Ce type de restriction ne peut être mis en évidence qu'en utilisant un modèle quantitatif. L'utilisation de modèles qualitatifs par Shafer et Healey peut expliquer une utilisation parfois un peu abusive du bruit d'acquisition pour expliquer les déviations des données par rapport au modèle. Les limitations intrinsèques du modèle interviennent également dans la déviation des points par rapport aux valeurs prévues.

Notons également que les influx nerveux transmis au cerveau ont une intensité proportionnelle au log de l'irradiance de l'onde électromagnétique frappant les cônes (récepteurs responsable de la vision des couleurs). De plus, ces signaux qui codent initialement des intensités rouge, verte et bleu sont transformés dans un espace d'opposition de couleurs proche de l'espace $H_1H_2H_3$ (figure 2.16).

Cette caractéristique de la vision humaine pourrait donc s'expliquer comme un phénomène adaptatif lié à la structure de la lumière réfléchie par les matériaux.

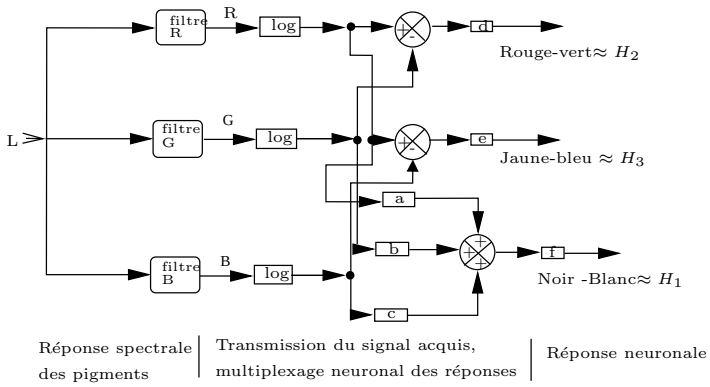


FIG. 2.16 – *Un modèle simplifié des pré-traitements de la vision Humaine*

2.5 Estimation de paramètres

Les méthodes d'estimation de paramètres consistent à déterminer les différentes constantes d'un modèle de réflexion. Ces paramètres peuvent être utilisés pour caractériser une surface ou pour reconstruire celle-ci. En effet, si nous utilisons des modèles de réflexion physique les paramètres d'un modèle peuvent être reliés à des données physique du matériau telles que la rugosité ou le coefficient de Fresnel. L'extraction de ces paramètres peut donc servir comme premier diagnostic d'un objet. De plus, la connaissance de ces paramètres est indispensable à l'extraction des normales de la surface à partir de l'intensité (section 2.6).

2.5.1 Estimation des paramètres dans le modèle Lambertien

Historiquement le modèle Lambertien (section 2.2.1) est un des premiers a avoir été utilisé en Vision [101]. Les premiers algorithmes d'estimation de paramètres [104, 155, 156, 28, 157, 103, 205] ont donc été conçus pour ce modèle. Le problème peut se formuler ainsi :

Étant donné un repère $(Oxyz)$ tel que z pointe vers la caméra, la direction $\overrightarrow{k}$ de la source lumineuse est donnée par : $\overrightarrow{k} = (\cos(\tau)\sin(\gamma), \sin(\tau)\sin(\gamma), \cos(\gamma))$ où τ correspond à l'angle entre la projection de $\overrightarrow{k}$ dans le plan (Oxy) et le vecteur $\overrightarrow{x}$. Cet angle est appelé l'azimut de l'illuminant. L'angle γ correspond à l'angle entre $\overrightarrow{k}$ et l'axe z. L'angle $\frac{\pi}{2} - \gamma$ étant appelé l'élévation de $\overrightarrow{k}$.

Le vecteur $\overrightarrow{n}$ normal à la surface $z(x, y)$ peut de même être exprimé en fonction de son azimut α et de l'angle β entre $\vec{n}$ et l'axe z : $\vec{n} = (\cos(\alpha)\sin(\beta), \sin(\alpha)\sin(\beta), \cos(\beta))$. L'intensité d'un pixel dans le modèle Lambertien (section 2.2.1) est liée à $\overrightarrow{k}$ et $\overrightarrow{n}$ par :

$$\begin{aligned} I(\alpha, \beta) &= \mathrm{K}_{diff} \max(0, \overrightarrow{n} . \overrightarrow{k}) \\ &= \mathrm{K}_{diff} \max(0, \cos(\alpha - \tau)\sin(\beta)\sin(\gamma) + \cos(\beta)\cos(\gamma)) \end{aligned} \tag{2.33}$$

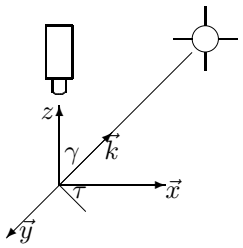


FIG. 2.17 – Le angles γ et τ du vecteur $\vec{k}$.

où K_{diff} est la constante du modèle Lambertien.

L'estimation des paramètres dans le cadre du modèle Lambertien revient donc à déterminer K_{diff}, τ et γ. Cette estimation est usuellement effectuée à l'aide des calculs statistiques sur les intensités ou leurs dérivées. L'idée sous-jacente est que l'on ne peut extraire les paramètres en chaque point si l'on ne connaît pas la valeur des angles α et β. En revanche nous pouvons faire des hypothèses raisonnables sur la distribution de α et β et calculer des moyennes ou des variances pour obtenir des équations incluant γ, τ et K_{diff} mais indépendantes de la position des normales.

Un bon exemple d'un tel procédé nous est fournit par Zheng et Chelappa [205]. Ceux-ci estiment que l'angle α de la normale $\overrightarrow{n}$ n'est soumis à aucune contrainte. En conséquence, il est libre de varier entre 0 et 2π et sa densité de probabilité est égale à :

$$f_\alpha = \frac{1}{2\pi}.$$

L'azimut β de la normale d'un objet est libre de varier entre 0 et π, toutefois seuls seront visibles les points dont la normale pointe sur la caméra. Nous ne travaillerons donc que sur des valeurs de β comprises entre 0 et $\frac{\pi}{2}$. Il nous faut donc estimer la densité de points projetés sur l'image dont l'azimut est égal à $\beta \in [0, \frac{\pi}{2}]$. Une telle distribution dépend bien entendu du type de surface considérée. Toutefois, Zheng et Chelappa estiment la densité de probabilité de β à partir de la longueur de la projection de la normale sur le plan image : $\cos(\beta)$. L'idée sous-jacente étant que des points possédant un grande projection sont plus visibles par la caméra que des points de faible projection. Ces points seront donc plus nombreux à être projetés et auront une fréquence plus importante. La densité de probabilité de β est donc fixée à :

$$f_\beta = \cos(\beta).$$

Les angles α et β étant *a priori* indépendants, la densité de probabilité codant l'orientation de la normale est donnée par :

$$f_{\alpha,\beta} = \frac{1}{2\pi}\cos(\beta).$$

L'image peut donc être considérée comme un tirage de variables aléatoires α, β produisant une intensité $I(\alpha, \beta)$ (équation 2.33). La moyenne des intensités est donc donnée par :

$$\begin{aligned} \overline{I} &= \int_0^{2\pi}\int_0^{\frac{\pi}{2}} I(\alpha,\beta) f_{\alpha,\beta}(\alpha,\beta) d\alpha d\beta \\ &= K_{diff}\int_0^{2\pi}\int_0^{\frac{\pi}{2}} \left(\cos(\alpha-\tau)\sin(\beta)\sin(\gamma) + \cos(\beta)\cos(\gamma)\right) f_{\alpha,\beta}(\alpha,\beta) d\alpha d\beta. \end{aligned}$$

Puisque nous sommons sur α et β, le terme :

$$\int_0^{2\pi}\int_0^{\frac{\pi}{2}} \left(\cos(\alpha-\tau)\sin(\beta)\sin(\gamma)+\cos(\beta)\cos(\gamma)\right) f_{\alpha,\beta}(\alpha,\beta)d\alpha d\beta$$

ne dépend *a priori* que de γ et τ. De plus, on peut facilement vérifier que :

$$\int_0^{2\pi} \cos(\alpha-\tau)d\alpha = 0.$$

L'influence de τ disparaît donc de l'intégrale et $\overline{I}$ peut se définir comme :

$$\overline{I} = \mathrm{K}_{diff} g_1(\gamma)$$

où g_1 ne dépend que de γ.

De même, on peut montrer que :

$$\overline{I^2} = {\mathrm{K}_{diff}}^2 g_2(\gamma) \text{ avec } g_2(\gamma) = \int_0^{2\pi}\int_0^{\frac{\pi}{2}} \left(\frac{I(\alpha,\beta)}{\mathrm{K}_{diff}}\right)^2 f_{\alpha,\beta}(\alpha,\beta)d\alpha d\beta.$$

L'angle γ peut donc être déterminé en résolvant l'équation :

$$g_3(\gamma) = \frac{g_1(\gamma)}{\sqrt{g_2(\gamma)}} = \frac{\overline{I}}{\sqrt{\overline{I^2}}}.$$

Étant donné γ les équations donnant $\overline{I}$ et $\overline{I^2}$ en fonction de K_{diff} et γ nous fournissent deux estimations de K_{diff}. Ces deux estimations sont combinées par une simple moyenne donnant :

$$\mathrm{K}_{diff} = \frac{1}{g_1^2(\gamma)+g_2(\gamma)}\left(\overline{I}g_1(\gamma)+\sqrt{\overline{I^2}g_2(\gamma)}\right).$$

Une méthode similaire a été également proposée antérieurement par Lee et Rosenfeld [104] en se basant sur une modélisation de chaque patch de surface par une sphère. Ceci leur permet de modéliser la distribution des angles α et β décrivant $\overrightarrow{n}$ par :

$$f_{\alpha,\beta} = \frac{1}{2\pi}\sin(2\beta).$$

L'estimation de γ et K_{diff} est ensuite réalisée par une méthode similaire à celle développée ultérieurement par Zheng et Chellapa.

Zheng et Chellapa proposent dans le même article [205] une méthode similaire à celle de Pentland [155] qui permet d'estimer l'azimut τ de l'illuminant à l'aide de moyennes et de variances des dérivées directionelles de l'image.

Zheng et Chellapa proposent également une alternative élégante à cette dernière méthode basée sur les contours de l'objet à estimer. En effet, la normale aux points du contour d'un objet à un angle β proche de $\frac{\pi}{2}$ que l'on peut supposer constant. De plus l'angle α de la normale peut être approximé pour ces pixels par la normale au contour de l'objet dans l'image. Le principal problème dans l'estimation des paramètres est l'absence d'information sur la normale. En étudiant des pixels pour lesquels on peut avoir une bonne estimation de celle-ci Zheng et Chellapa tournent de façon élégante cette difficulté.

2.5.2 Estimation des paramètres avec une connaissance *a priori* des normales

Le calcul des paramètres pour le modèle Lambertien est relativement aisé dans le mesure où ce modèle est défini par une seule constante K_{diff} et un seul paramètre géométrique θ codant l'angle entre $\vec{n}$ et la direction de l'illuminant $\vec{k}$. Le problème est toutefois plus complexe si l'on utilise des modèles plus complets tels que le modèle de Nayar (section 2.2.4).

Ikeuchi [111] propose une méthode d'estimation des paramètres à partir d'une image des intensités et d'une image des normales obtenues à partir de mesures laser. Ikeuchi utilise une version simplifiée du modèle de Nayar, qui n'inclut pas le pic spéculaire. L'intensité d'un pixel $I(x, y)$ est donc donnée par (équation 2.15) :

$$I(x,y) = \mathrm{K}_{diff} \cos(\theta_i(x,y)) + \frac{\mathrm{C}_{ls}}{\cos(\theta_r(x,y))} e^{-\frac{\alpha^2(x,y)}{2\sigma_\alpha^2}} \text{ si } \theta_i \in [0, \frac{\pi}{2}], 0 \text{ sinon.}$$

La composante Lambertienne de ce modèle peut également se définir par :

$$I_{lam}(x,y) = \mathrm{K}_{diff} \max\left(0, \cos(\theta_i(x,y))\right) = \mathrm{K}_{diff} \max\left(0, \vec{n}(x,y).\vec{k}\right) = \max(0, \vec{n}(x,y).\vec{K})$$

où $\vec{k}$ est le vecteur de l'illuminant de norme 1 dirigé vers la source (figure 2.17) et $\vec{K} = \mathrm{K}_{diff}\vec{k}$. Le vecteur $\vec{k}$ correspond donc à $-\vec{k_1}$ sur la figure 2.6(a).

Connaissant les normales Ikeuchi obtient une première estimation de K_{diff} et $\vec{k}$ en minimisant sur l'ensemble de l'image :

$$\sum_{x,y} \left[I(x,y) - (K_x n_x(x,y) + K_y n_y(x,y) + K_z n_z(x,y))\right]^2$$

où $\vec{K} = (K_x, K_y, K_z)$.

Cette première estimation n'est évidemment pas précise en raison des zones spéculaires de l'image. Elle permet toutefois de déterminer les pixels se situant dans des zones non spéculaires à l'aide d'un seuil th. Ikeuchi considère qu'un pixel satisfaisant le test :

$$|I(x,y) - \vec{K}.\vec{n}(x,y)| < th$$

a une intensité raisonnablement bien approximée par la composante Lambertienne. Il applique donc à nouveau le processus de minimisation sur les pixels ainsi sélectionnés afin d'obtenir une meilleure estimation du vecteur $\vec{K}$. Ce processus est itéré jusqu'à convergence. Étant donné le vecteur $\vec{K}$, K_{diff} et $\vec{k}$ se déduisent par :

$$\begin{cases} \mathrm{K}_{diff} &= \|\vec{K}\| \\ \vec{k} &= \frac{1}{\|\vec{K}\|}\vec{K}. \end{cases}$$

Étant donnée la constante K_{diff} et le vecteur $\vec{k}$, on peut supprimer la composante Lambertienne en créant l'image :

$$d(x,y) = I(x,y) - \mathrm{K}_{diff} \max(0, \vec{k}.\vec{n}(x,y)) \quad \forall x, y$$

ce qui nous donne une image de la composante spéculaire.

Notons que les vecteurs $\vec{k}, \vec{k_r}$ et $\vec{n}$ étant connus, nous pouvons calculer l'angle $\alpha(x,y)$ entre $\vec{n}$ et la bissectrice $\vec{\nu}$ de $\vec{k_r}$ et $\vec{k}$ (figure 2.6(b)). On peut donc sélectionner les pixels tels que $\alpha(x,y) \approx 0$ pour lesquels :

$$d(x,y) = \frac{\mathrm{C}_{ls}}{\cos(\theta_r(x,y))} e^{-\frac{\alpha^2(x,y)}{2\sigma_\alpha^2}} \approx \frac{\mathrm{C}_{ls}}{\cos(\theta_r(x,y))}.$$

Connaissant $\vec{k_r}$ et $\vec{n}(x,y)$, nous pouvons calculer $\theta_r(x,y)$ et avoir donc une première estimation de C_{ls}. Le paramètre σ_α est alors estimé à partir de C_{ls} en minimisant sur l'image d la quantité :

$$\sum_{x,y} \left[\log(d(x,y)) + \log(\cos(\theta_r(x,y))) - \log(\mathrm{C}_{ls}) + \frac{\alpha^2(x,y)}{2\sigma_\alpha^2} \right]^2. \tag{2.34}$$

La valeur de C_{ls} peut être affinée à partir de l'estimation de σ_α en minimisant :

$$\sum_{x,y} \left[d(x,y) - \frac{\mathrm{C}_{ls}}{\cos(\theta_r(x,y))} e^{-\frac{\alpha^2(x,y)}{2\sigma_\alpha^2}} \right]^2. \tag{2.35}$$

Cette meilleure estimation de C_{ls} nous permet d'obtenir par l'équation 2.34 une meilleure estimation de σ_α qui nous donne à son tour une meilleure estimation de C_{ls}. Les deux processus de minimisation sont itérés jusqu'à convergence.

Notons que la méthode proposée par Ikeuchi repose sur une connaissance *a priori* des normales de la surface. Or bien souvent l'estimation des paramètres n'est qu'une étape intermédiaire pour retrouver les normales. Le problème est donc ici légèrement biaisé. De plus, des expériences menées en collaboration avec le laboratoire PRIP à Vienne nous ont montré que la technologie laser préconisée par Ikeuchi pour retrouver les normales fonctionne très mal sur des objets métalliques. En effet, l'aspect spéculaire de telles surfaces ne permet pas un retour d'une quantité suffisante d'énergie sur le capteur pour obtenir des informations fiables.

2.5.3 Estimation de paramètres à partir de plusieurs acquisitions

Si nous nous référons au modèle de Nayar, l'intensité d'un pixel est donnée par l'équation :

$$I = \mathrm{K}_{diff} \cos(\theta_i) + \frac{\mathrm{C}_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + \mathrm{K}_{ps} \delta(\theta_i - \theta_r)\delta(\psi_r) \text{ si } \theta_i \in [0, \frac{\pi}{2}], 0 \text{ sinon}$$

L'intensité de chaque pixel est donc déterminée par 8 paramètres $(\mathrm{K}_{diff}, \mathrm{C}_{ls}, \mathrm{K}_{ps}, \sigma_\alpha, \theta_i, \theta_r, \alpha, \psi_r)$ et il est évident que sans hypothèses simplificatrices l'obtention simultanée des normales et des paramètres de la surface ne peut se faire avec une seule acquisition. L'obtention simultanée de plusieurs acquisitions pour chaque pixel peut schématiquement s'obtenir de trois manières :

1. Déplacer la source lumineuse,
2. Déplacer la caméra,
3. Déplacer l'objet.

Les techniques utilisant plusieurs sources lumineuses sont connues sous le nom de Reconstruction par plusieurs illuminants ("Shape from photometric sampling") [180, 179, 181, 118]. Les techniques basées sur plusieurs caméras ou sur le déplacement d'une caméra sont appelées des

techniques de Reconstruction par Stéréovision [144, 143, 185, 140], dans le cas de deux vues ou de Reconstruction par déplacement de caméra ("Shape from motion") [73, 88] pour une séquence continue d'images. Les techniques consistant à déplacer l'objet sont assez variées et comprennent notamment les techniques de Reconstruction à partir de la silhouette [186].

Les méthodes de reconstruction à partir d'un déplacement de l'objet ou d'un déplacement de la caméra ne passent généralement pas par une estimation des paramètres d'un modèle physique. Aussi nous ne nous intéresserons dans la suite de cette section qu'aux techniques de Reconstruction par plusieurs illuminants et par photo-géométrie.

Les techniques utilisant plusieurs illuminants sont basées sur la prise de plusieurs acquisitions d'un même pixel en déplacent l'illuminant. Le montage généralement utilisé dans ce type de techniques est décrit dans la figure 2.18. La position de la source est décrite par les deux angles θ_i et ψ_i tandis que la caméra reste fixe dans l'alignement de l'axe Z.

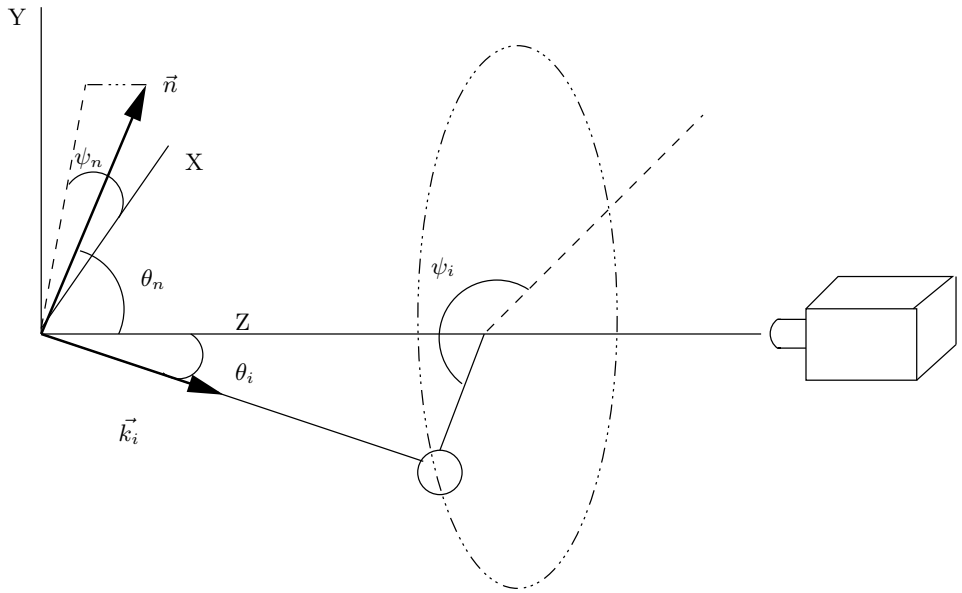


FIG. 2.18 – *Montage expérimental généralement utilisé pour prendre plusieurs acquisitions*

L'objectif de ce type de méthode est donc de prendre plusieurs acquisitions et de reconstruire la surface à partir de celles ci. Deux questions émergent immédiatement dans ce type de méthodes :

1. Existe t'il un nombre minimal d'acquisitions permettant de déterminer de manière unique l'ensemble des paramètres ?
2. Étant donné un certain nombre d'acquisitions, quel est le meilleur placement des sources lumineuses permettant d'effectuer l'estimation ?

La réponse à la première question a été apportée par Tagare [180] qui a montré, en utilisant un modèle de réflexion qui inclut les modèles Lambertien et celui de Nayar que la fonction de réflectance pouvait être inversée si chaque point est éclairé par au moins 3 sources lumineuses de

même angle θ et avec des angles ψ_i vérifiant :

$$\psi_1 = 0 \text{ et } \pi > \psi_2 = -\psi_3 > 0.$$

Tagare montre également que nous pouvons assurer que chaque point de la surface est éclairé par au moins 3 sources lumineuses en prenant 6 sources.

Ce résultat ne concerne toutefois que l'existence d'une solution. Il peut en effet exister des configurations de sources lumineuses telles que le système d'équations induit par l'inversion de la fonction de réflectance soit mal conditionné. Dans ce cas on est incapable de trouver la solution même si celle-ci est théoriquement unique.

Cette mésaventure est arrivée à Tagare [179] un an plus tard lorsqu'il a voulu effectuer effectivement l'inversion en utilisant 8 sources lumineuses de même angle θ et d'angles ψ uniformément répartis sur $[0, 2\pi[$. L'angle θ étant constant, l'intensité d'un pixel n'est plus fonction que de l'angle ψ (figure 2.18). Sa méthode basée sur une décomposition en série de Fourier de $I(\psi)$ conduit à une matrice mal conditionnée avec des vecteurs colonnes extrêmement proches. La méthode est donc extrêmement sensible au bruit et non utilisable sans une contrainte de régularisation.

Ce problème a été repris par Kay [118] qui a analysé exhaustivement les facteurs pouvant conduire à un problème mal conditionné. Le modèle de réflexion utilisé par Kay est basé sur le modèle de Nayar (équation 2.14) :

$$I_i = \mathrm{K}_{diff} \max(0, \cos(\theta_i)) + \mathrm{K}_{ls} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + \mathrm{K}_{ps} e^{-\frac{\alpha^2}{2\sigma'^2}}$$

où le terme $e^{-\frac{\alpha^2}{2\sigma'^2}}$ correspond à la double fonction de Dirac de Nayar. On a donc $\sigma' << \sigma_\alpha$.

Le problème peut se formaliser ainsi, la fonction de réflectance $f(z_i, \beta)$ est paramétrée par un vecteur de variables z_i et un vecteur de paramètres β. Le vecteur z_i peut correspondre à θ_i, ψ_i ou à tout vecteur déduit de θ_i, ψ_i. Le vecteur β est fonction de tous les paramètres de notre modèle de réflexion et ne dépend pas de θ_i, ψ_i. Il dépend en revanche de $\mathrm{K}_{diff}, \mathrm{K}_{ls}, \mathrm{K}_{ps}, \sigma_\alpha, \sigma', \theta_n$ et ψ_n. Étant données n acquisitions, le problème se ramène donc a trouver le vecteur β qui minimise :

$$\sum_{i=1}^{n} \left(I_i - f(z_i, \beta)\right)^2.$$

Kay utilise pour cela la méthode de Gauss-Newton qui utilise des approximations linéaires pour améliorer itérativement β à partir d'une estimation initiale. Étant donnée une estimation initiale des paramètres β^0, la valeur de $f(z_i, \beta)$ est approximée en utilisant la formule de Taylor :

$$f(z_i, \beta) = f(z_i, \beta^0) + \sum_{j=0}^{m} (\beta_j - \beta_j^0) \frac{\partial f}{\partial \beta_j}(z_i, \beta^0) \tag{2.36}$$

où m est le nombre de paramètres.

Le but de la méthode est de trouver les paramètres décrivant au mieux les intensités observées donc de minimiser la norme du résidu

$$r(z_i, \beta) = I - f(z, \beta)$$

où $I = (I_1, \ldots, I_n)$ est le vecteur des n observations et $f(z, \beta) = (f(z_1, \beta), \ldots, f(z_n, \beta))$ est le vecteur des n prédictions en fonction des paramètres β.

Nous obtenons par l'équation 2.36 :

$$r(z_i, \beta) = I - f(z, \beta) = I - f(z, \beta^0) - V^0\delta = r^0 - V^0\delta$$

où V^0 est la matrice $n \times m$ de coefficients $\nu_{i,j} = \frac{\partial f}{\partial \beta_j}(z_i, \beta^0)$ et δ et le $m \times 1$ vecteur $(\beta - \beta^0)$.

La minimisation de $r(z_i, \beta)$ est donc équivalente à la minimisation de la norme de $r^0 - V^0\delta$. Kay propose une résolution de ce système à l'aide d'une décomposition en valeurs singulières qui consiste à exprimer V^0 sous la forme d'un produit de 3 matrices :

$$V^0 = UDA$$

où U et A sont des matrices orthogonales ($U^tU = A^tA = Id$) de tailles respectives $n \times m$ et $m \times m$ et D une matrice diagonale de taille $m \times m$ de valeurs positives sur la diagonale. Le vecteur δ s'obtient alors par

$$\delta = A^tD^{-1}U^tr^0.$$

Le principal intérêt de cette décomposition est qu'elle permet de tester si le système est bien conditionné. En effet, si $(d_j)_{j\in\{1,\ldots,m\}}$ représente les valeurs de la diagonale de D, le rapport :

$$\kappa = \frac{\max_{j=1,\ldots,m} d_j}{\min_{j=1,\ldots,m} d_j}$$

mesure le mauvais conditionnement de la matrice V^0. Des expériences menées par Belsley [9] suggèrent qu'une valeur de κ supérieure à 30 peuvent poser des problèmes de résolution.

La détermination de conditions optimales d'éclairement peut donc se formuler ainsi. Étant donné un ensemble de paramètres possibles $\mathcal{P}$ et un ensemble de conditions d'acquisitions $\mathcal{A}$ déterminer l'élément $A \in \mathcal{A}$ qui maximise le nombre de cas où $\kappa \leq 30$ sur toutes les scènes possibles avec tous les paramètres $\beta \in \mathcal{P}$. Le problème formulé ainsi est encore trop général pour être résolu, toutefois Kay considère un sous ensemble finis $\mathcal{A}_f$ mais représentatif de conditions d'acquisition. Il suppose notamment que pour chaque angle θ_i nous disposons de q_j illuminants régulièrement répartis sur $]-\pi, \pi[$. Une acquisition $A \in \mathcal{A}_f$ est donc définie par un ensemble de couples $\{(\theta_1, q_1), \ldots (\theta_n, q_n)\}$. Kay définit également un sous ensemble fini et également représentatif $\mathcal{P}_f$ de l'ensemble des paramètres et fixe pour chaque condition d'acquisition, la normale à la surface de telle façon qu'elle induise un maximum de difficultés dans la résolution du système. Kay définit alors la fonction :

$$\Gamma : \begin{array}{ccc} \mathcal{A}_f & \rightarrow & \mathbb{R}_+ \\ A & \mapsto & \frac{\int_{\mathcal{P}_f} \psi(\beta, A) d\beta}{\int_{\mathcal{P}_f} d\beta} \end{array} \quad \text{avec } \psi(\beta, A) = \begin{cases} 1 & \text{si } \kappa \leq 30 \\ 0 & \text{si } \kappa > 30. \end{cases}$$

La fonction Γ compte donc la proportion de paramètres pour lesquels le système est bien conditionné. Les conditions d'acquisitions optimum sont donc celles qui maximisent Γ. Intuitivement, les meilleures conditions d'acquisitions sont obtenues avec un nombre maximum d'illuminants. Afin d'éviter une "explosion" du nombre d'illuminants Kay maximise la fonction :

$$\Gamma_{moy}(k) = \frac{1}{|\mathcal{N}(k)|} \sum_{A \in \mathcal{N}(k)} \Gamma(A)$$

où $\mathcal{N}(k) \subset \mathcal{A}_f$ est le nombre de conditions d'acquisition comportant k illuminants.

Cette dernière formule permet d'obtenir une mesure des performances moyennes du système pour des acquisitions comportant k illuminants.

Kay constate que Γ_{moy} reste approximativement constant au delà de 55 illuminants. Il décide donc d'utiliser 55 illuminants et effectue une évaluation exhaustive de Γ sur les domaines finis $\mathcal{N}(55)$ et $\mathcal{P}_f$. Ceci lui permet de déterminer la meilleure condition d'acquisition définie par $\{(15^\circ, 3), (20^\circ, 13), (25^\circ, 9), (30^\circ, 30)\}$. Ce qui nous fait au total 55 positions d'illuminants avec 4 valeurs différentes de θ_i. De tels chiffres sont très éloignés des 3 illuminants initialement suggérés par Tagare [180].

Notons que la maximisation de Γ minimise la proportion de pixels pour lesquels la matrice V^0 est mal conditionnée sans toutefois exclure de tels cas. Il faut par exemple un total de 240 illuminants à Kay pour obtenir une valeur maximum de Γ de l'ordre de 74%. Ce nombre sera certainement plus faible pour 55 illuminants. Kay évalue donc les conditions qui peuvent conduire à un mauvais conditionnement du système et modifie la méthode de résolution lorsque cela est possible. Les principaux cas pouvant se produire lors de l'extraction des paramètres sont :

1. un nombre insuffisant de données. La normale au point est telle que la plupart des intensités sont proches de 0. Dans ce cas le problème est insoluble sans changer les conditions d'acquisition ;
2. les donnés sont extrêmement similaires. Ce cas ce produit si $\theta_n \approx 0$, dans ce cas l'intensité devient approximativement indépendante de l'angle ψ_i de l'illuminant. Ce qui dans le cas de notre acquisition idéale nous ramène à uniquement 4 valeurs d'intensités différentes correspondant aux 4 valeurs de θ_i ;
3. les constantes K_{ls} et K_{ps} du modèle sont approximativement nulles. Ceci va nous donner un modèle sur-paramétrisé puisque les valeurs de σ_α et σ' n'ont dans ce cas aucune influence sur les intensités. Il faut donc utiliser un modèle uniquement Lambertien ;
4. si la surface est rugueuse, le terme codant le pic spéculaire est négligeable (section 2.2.2). On peut donc poser $\mathrm{K}_{ps} = 0$. De plus, le paramètre σ_α est important et $e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}$ peut être approximé par $1 - \frac{\alpha^2}{2\sigma_\alpha^2}$. Cette approximation fait disparaître un paramètre qui sans cela conduirait à une sur paramétrisation du problème ;
5. le pixel est essentiellement spéculaire. On a donc dans ce cas $\mathrm{K}_{diff} \approx 0$. Le modèle le plus adéquat est donc un modèle comportant le lobe spéculaire et le pic spéculaire ;
6. le pixel ne comporte qu'un lobe spéculaire. On restreint dans ce cas le modèle en posant $\mathrm{K}_{diff} = \mathrm{K}_{ps} = 0$;
7. le modèle est complet et tous les paramètres significatifs.

Notons que dans les cas 1 et 2 nous ne pouvons rien faire si ce n'est constater l'insuffisance de données. En revanche dans les cas 3 à 7 le mauvais conditionnement de la matrice V^0 peut être résolu en utilisant le modèle approprié. Kay utilise une méthode définie par Gallat [83] pour comparer les modèles et sélectionner celui qui correspond le mieux aux observations.

Notons que la méthode de Kay est extrêmement coûteuse puisqu'elle exige d'appliquer pour chaque pixel :

1. la résolution de plusieurs systèmes pour déterminer le meilleur modèle ;
2. une résolution complète pour le modèle sélectionné.

A titre d'exemple, la méthode de Kay exige 7 heures de temps de calculs sur une image de 115×150 pixels avec un PC 486 à $33Mz$. L'utilisation d'une machine plus évoluée et d'optimisations conseillées par Kay pourraient sans doute ramener ces temps de calculs en dessous de 1 heure. On reste toutefois bien au-dessus de temps proche de la seconde exigés dans le cadre d'un contrôle industriel. Malgré ces temps de calculs prohibitifs cette méthode est toutefois une des méthodes les plus complètes d'estimation des paramètres par déplacement d'illuminants qu'il m'ai été donné d'étudier. De plus nous avons repris certaines des idées de Kay en les adaptant à notre problématique. Nous avons notamment repris l'idée qu'un modèle complet comme celui de Nayar pouvait ne pas être adéquat pour décrire l'ensemble des pixels d'une image. Nous avons également repris la notion de résolution par pelage utilisée par Kay [118] et Tagare [179]. Intuitivement, l'idée est de déterminer un des paramètres de l'équation indépendamment des autres. Ce paramètre, une fois obtenu permet d'obtenir un second paramètre puis un troisième jusqu'à l'obtention de l'ensemble des paramètres. Dans le cas des méthodes de Kay et Tagare le premier paramètre recherché est l'angle ψ_n entre la projection de la normale sur le plan Oxy et l'axe x.

2.5.4 Notre contribution à l'estimation de paramètres

L'application que nous avons développée se place dans le cadre d'un contrôle industriel de pièces manufacturées métalliques. L'aspect métallique des pièces à contrôler nous interdit d'utiliser le modèle Lambertien qui ne rend pas compte de l'aspect spéculaire de telles surfaces. Nous avons donc opté pour le modèle de Nayar (section 2.2.4) qui permet de décrire le phénomène de réflexion sur de telles surfaces. Nous avons toutefois négligé le pic spéculaire dans un premier temps. Le modèle est donc défini pour un observateur variable par (équation 2.15) :

$$I = \mathrm{K}_{diff} \cos(\theta_i) + \frac{\mathrm{C}_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \text{ si } \theta_i \in [0, \frac{\pi}{2}], 0 \text{ sinon.}$$

Les méthodes d'estimation des paramètres basées sur des modèles physiques (section 2.5.3) utilisent généralement un nombre important d'illuminants et ne font aucune supposition sur les valeurs des constantes du modèle. Du fait de leur généralité ces méthodes réclament souvent un protocole expérimental assez complexe (déplacer l'illuminant avec deux axes de liberté), un temps d'acquisition assez long du fait du nombre important d'images à acquérir (une pour chaque illuminant) et des temps calculs souvent trop longs pour une application industrielle.

Notre estimation des paramètres devra donc faire un certain nombre d'hypothèses afin de simplifier le problème et obtenir une méthode d'acquisition plus simple et des temps de calculs proches de la seconde (voir section 2.1).

Afin de réaliser ces objectifs nous allons supposer que :

1. les conditions d'acquisition (direction de l'illuminant et de l'observateur) sont parfaitements connues par le protocole d'acquisition et ne font donc pas partie des paramètres à estimer ;
2. l'image est décomposée en matériaux. Une telle décomposition peut s'obtenir à partir d'algorithmes de segmentation en matériaux (section 2.4) ;
3. dans chaque région le matériau est supposé optiquement uniforme. Ceci signifie que les valeurs de $\mathrm{K}_{diff}, \mathrm{C}_{ls}$ et σ_α sont supposées constantes dans chaque région ;
4. le maximum d'intensité se produit pour le pic spéculaire. Une telle supposition est raisonnable dans le cas de matériaux métalliques pour lesquels le lobe spéculaire prédomine la composante Lambertienne ;

5. les matériaux sont supposés non rugueux ($\frac{1}{2\sigma_\alpha^2} \gtrapprox 1$).

Le principal obstacle à l'estimation des constantes d'un modèle tel que celui de Nayar est que l'intensité de chaque pixel est une combinaison des composantes Lambertienne et Spéculaires. Les méthodes basées sur de multiples acquisitions résolvent généralement ce problème en prenant un nombre important d'acquisitions. L'approche que nous avons privilégiée est basée sur les hypothèses précédentes et consiste à se placer dans des zones de l'image où l'on peut faire des hypothèses raisonnables sur les composantes Lambertienne et spéculaire.

2.5.4.a Estimation de K_{diff}

L'estimation de la constante K_{diff} du modèle Lambertien est effectuée en se plaçant dans une zone de l'image où nous pouvons légitimement supposer l'influence du lobe spéculaire négligeable. La localisation de ces zones est basée sur les hypothèses 4 et 5 définissant le cadre de notre méthode :

1. Les maxima d'intensités dans l'image sont supposés correspondre aux maxima du lobe spéculaire,
2. Le lobe spéculaire décroît rapidement autour de chaque maxima.

Nous pouvons donc légitimement supposer que les zones où le lobe spéculaire est non négligeable se situent autour des zones de fortes intensités. Inversement, les zones de faibles intensités correspondent à des zones essentiellement Lambertiennes. Étant donnée une image I_1 on aura donc dans ces zones :

$$I_1(j) = \mathrm{K}_{diff} \max\left(0, \cos(\theta_i(j))\right)$$

où j représente l'indice du pixel.

Nous fixons un seuil η et sélectionnons l'ensemble des pixels tels que :

$$0 < I_1(j) = \mathrm{K}_{diff} \max\left(0, \cos(\theta_i(j))\right) < \eta$$

La sélection d'un ensemble de pixels tels que $I_1(j) > 0$ nous permet de garantir qu'en chacun des points sélectionnés $\theta_i^1(j) < \frac{\pi}{2}$. De plus η étant supposé proche de 0, la contrainte $I_1(j) < \eta$ impose à l'angle θ_i^1 d'être proche de $\frac{\pi}{2}$ sans l'atteindre.

Les contraintes précédentes sur l'angle $\theta_i(j)$ peuvent être explicitées en posant $\theta_i^1(j) = \frac{\pi}{2} - \epsilon_j$ avec $\epsilon_j > 0$. On obtient alors :

$$I_1(j) = \mathrm{K}_{diff} \max(0, \cos(\theta_i^1(j))) = \mathrm{K}_{diff} \cos(\theta_i^1(j)) = \mathrm{K}_{diff} \cos\left(\frac{\pi}{2} - \epsilon_j\right) = \mathrm{K}_{diff} \sin(\epsilon_j) \quad (2.37)$$

pour tout pixel sélectionné.

Si $\{1, \ldots, n\}$ désigne l'ensemble des indices de pixels sélectionnés on obtient :

$$\forall j \in \{1, \ldots, n\} \quad 0 < \sin(\epsilon_j) < \frac{\eta}{\mathrm{K}_{diff}} \Leftrightarrow \epsilon_j \in \left]0, \arcsin\left(\frac{\eta}{\mathrm{K}_{diff}}\right)\right[.$$

Aucune autre contrainte n'étant imposée à ϵ_j, nous supposons que ϵ_j a une distribution uniforme sur $]0, \arcsin(\frac{\eta}{\mathrm{K}_{diff}})[$. Sa fonction de densité de probabilité est donc définie par :

$$f_\epsilon : \begin{array}{ccc}]0, k[& \rightarrow & [0, 1] \\ \epsilon & \mapsto & \frac{1}{k} \end{array} \qquad \text{avec } k = \arcsin\left(\frac{\eta}{\mathrm{K}_{diff}}\right).$$

Nous utilisons alors une méthode similaire à celle de Zheng [205] (section 2.5.1) et estimons l'intensité moyenne de notre échantillon par :

$$\int_0^k I_1(\epsilon) f_\epsilon(\epsilon) d\epsilon = \mathrm{K}_{diff} \int_0^k \frac{1}{k} \sin(\epsilon) d\epsilon = \overline{I_1}$$

où $\overline{I_1}$ représente la valeur moyenne de I_1 sur notre échantillon de n valeurs.

Nous avons donc :

$$\mathrm{K}_{diff} \int_0^k \frac{1}{k} \sin(\epsilon) d\epsilon = \frac{\eta \tan(\frac{k}{2})}{k} = \overline{I_1},$$

ce qui nous donne :

$$\frac{\tan(\frac{k}{2})}{\frac{k}{2}} = \frac{2\overline{I_1}}{\eta}.$$

La fonction $\frac{\tan(x)}{x}$ étant strictement croissante pour $x > 0$, il existe une seule valeur k_0 vérifiant l'égalité ci-dessus. On en déduit alors K_{diff} par $\mathrm{K}_{diff} = \frac{\eta}{\sin(k_0)}$.

Nous verrons dans les sections suivantes que l'utilisation de plusieurs illuminants est nécessaire pour l'estimation des constantes C_{ls} et σ_α. On peut donc profiter de ces illuminants pour obtenir plusieurs estimations de K_{diff}. Un façon simple d'utiliser ces différents illuminants est d'appliquer la méthode précédente pour chaque acquisition est de combiner les différentes estimations. Toutefois la combinaison de plusieurs estimations implique de pouvoir affecter un poids ou un coefficient de confiance à chaque estimation.

La principale hypothèse que nous avons faite pour calculer K_{diff} est la supposition que ϵ suit une loi uniforme. Or nous avons pour chacun des pixels sélectionnés :

$$I_1(j) = \mathrm{K}_{diff} \sin(\epsilon_j)$$

avec ϵ_j proche de 0.

On a donc $I_1 \approx \mathrm{K}_{diff}\epsilon_j$ et si ϵ_j suit une loi uniforme I_1 doit suivre la même loi. Notre supposition que ϵ suit une loi uniforme peut donc être testée en vérifiant que les n échantillons de I_1 suivent une loi uniforme sur $]0, \eta[$. Si nous subdivisons l'intervalle $]0, \eta[$ en p sous intervalles et calculons le nombre $f^{obs}(i)$ d'échantillons parmi n qui appartiennent au $i^{\text{ème}}$ intervalle de $]0, \eta[$ nous devrions avoir :

$$\forall i \in \{1, \ldots, p\} \quad f^{obs}(i) \approx \frac{n}{p}.$$

On peut montrer [48] que la variable :

$$\chi = \sum_{i=1}^{p} \frac{(f^{obs}(i) - \frac{n}{p})^2}{\frac{n}{p}} \tag{2.38}$$

suit la loi du χ^2 avec $p-1$ degrés de libertés ce qui permet de faire des tests statistiques indiquant avec quelle marge d'erreur nous pouvons supposer que I_1 suit une loi uniforme. Plus exactement, on suppose que I_1 suit une loi uniforme avec un seuil de signification α si $\chi < \chi^2_{\alpha,p-1}$. Toutefois, dans le cadre d'une combinaisons de plusieurs estimations nous sommes plus intéressés par un indicateur continu que par une réponse booléenne. Cet opérateur continu nous est fourni par l'inverse de la variable χ.

Étant données q acquisitions nous calculons donc q estimations de K_{diff} : ${\mathrm{K}_{diff}}^1, \ldots, {\mathrm{K}_{diff}}^q$ et appliquons pour chaque estimation l'équation 2.38 qui nous fournit q indicateurs de confiance $\chi_1^{-1}, \ldots, \chi_q^{-1}$. L'estimation finale de K_{diff} est donnée par :

$$\mathrm{K}_{diff} = \frac{\sum_{i=1}^{q} \chi_i^{-1} {\mathrm{K}_{diff}}^i}{\sum_{i=1}^{q} \chi_i^{-1}}.$$

J'ai également montré avec Laurent Hussenet [109] que K_{diff} pouvait être directement obtenu en utilisant conjointement les informations de 2 acquisitions. Étant données deux acquisitions I_1 et I_2 telles que l'angle entre les deux illuminants est égal à β nous sélectionnons n échantillons tels que :

$$\forall j \in \{1, \ldots, n\} \quad \left\{ \begin{array}{rcl} 0 & < & I_1(j) < \eta \\ 0 & < & I(j) \end{array} \right. \quad \text{avec } I(j) = \frac{I_2(j) - \cos(\beta) I_1(j)}{\sin(\beta)}.$$

Cette sélection peut donc se voir comme un filtre supplémentaire à la sélection précédente dans lequel on impose la condition :$I > 0$. Le coefficient K_{diff} s'obtient alors par :

$$\mathrm{K}_{diff} = \frac{\eta}{sin\left(\frac{2\eta}{\pi \overline{I}}\right)}$$

où $\overline{I}$ est la moyenne de $I(j)$ sur nos n échantillons.

Le coefficient de confiance d'une telle estimation est toutefois plus délicat à estimer pour ce type de méthode.

2.5.4.b Estimation de C_{ls} et σ_α

Comme nous l'avons vu (section 2.5.4.a), l'estimation de K_{diff} peut se faire à partir d'une seule acquisition ou de plusieurs acquisitions indépendantes. Toutefois, l'estimation de C_{ls} et σ_α va nous amener à utiliser simultanément des images de plusieurs acquisitions. Il nous faut donc préciser notre protocole d'acquisition et le repère dans lequel on se place.

Le protocole d'acquisition est constitué de n sources lumineuses coplanaires (figure 2.19). Chaque source lumineuse est défini par un vecteur $\vec{k_i}$ faisant un angle γ_i avec la verticale. La caméra est dans l'axe vertical défini par l'axe Z.

Le repère est choisi de telle façon que l'axe Z coïncide avec le vecteur d'observation $\vec{k_r}$. L'axe X est dans le plan de la caméra et des sources lumineuses et est choisi de telle façon que les vecteurs $\vec{k_i}$ aient une projection positive sur l'axe X (figure 2.19).

Nous avons donc dans ce repère :

$$\vec{k_r} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \text{ et } \forall i \in \{1, \ldots, n\} \; \vec{k_i} = \begin{pmatrix} \sin(\gamma_i) \\ 0 \\ \cos(\gamma_i) \end{pmatrix}.$$

Le vecteur normal $\vec{n}$ s'exprime quand à lui en fonction de son angle θ avec l'axe Z et de son angle ψ avec le plan (Oxz) :

$$\vec{n} = \begin{pmatrix} \sin(\theta)\cos(\psi) \\ \sin(\theta)\sin(\psi) \\ \cos(\theta) \end{pmatrix}.$$

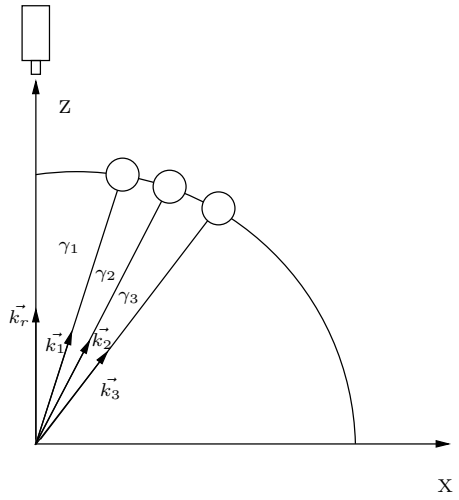


FIG. 2.19 – *Notre protocole d'acquisition*

L'intensité d'un pixel dans le modèle de Nayar est donnée par :

$$I(\vec{n}) = \mathrm{K}_{diff}\cos(\theta_i) + \frac{\mathrm{C}_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}.$$

Pour une position d'illuminant γ_j, l'intensité est donc déterminée par les trois angles $(\theta_i, \theta_r, \alpha)$. Or un angle γ_j étant fixé les angles θ_i, θ_r et α ne sont fonctions que de la position de la normale définie par les angles θ et ψ. Afin d'éliminer les variables liées dans la formule de Nayar nous avons exprimé l'intensité d'un pixel directement en fonction de θ et ψ et montré [109] que sous l'approximation $\cos(\alpha) \approx 1 - \frac{\alpha^2}{2}$, le modèle de Nayar pouvait se réécrire de la façon suivante :

$$I_i(l,\theta) = \frac{\mathrm{C}_{ls}}{\Delta_{spec}(1,\theta,0)} e^{-\frac{1}{\sigma_\alpha^2}\left(1-\Delta_{spec}(l,\theta,\frac{\gamma_i}{2})\right)} + \mathrm{K}_{diff}\Delta_{spec}(l,\theta,\gamma_i)$$

avec $l = \sin^2(\frac{\psi}{2}) \in [0,1]$ et

$$\begin{array}{rccl} \Delta_{spec}: & \mathbb{R}^3 & \to & \mathbb{R} \\ & (l,\theta,\gamma) & \mapsto & l\cos(\theta+\gamma) + (1-l)\cos(\theta-\gamma) \end{array}$$

où θ représente l'angle entre la normale et l'axe Z.

Notons que des valeurs de l égales à 0 ou 1 correspondent au cas où $\vec{n}$ est dans le plan (Ozx) alors que $l = \frac{1}{2}$ correspond au cas ou $\vec{n}$ est orthogonal à ce plan.

Outre la disparition de variables liées, cette nouvelle formulation permet d'avoir une formule simple et unifiée pour l'ensemble des positions d'illuminants.

L'estimation des paramètres C_{ls} et σ_α de la composante spéculaire nécessite de se placer dans une zone de l'image où cette composante n'est pas négligeable. Nous utilisons donc l'hypothèse 4 et sélectionnons n points d'intensité maximale dans l'image de notre première acquisition.

On a donc pour chacun de ces points :

$$\left.\begin{array}{l}\forall j \in \{1,\ldots,q\}\\ \forall i \in \{1,\ldots,n\}\end{array}\right\} \quad I_j(i) = \frac{C_{sl}}{\Delta_{spec}(1,\theta_i,0)} e^{-\frac{1}{\sigma_\alpha^2}\left(1-\Delta_{spec}(l_i,\theta_i,\frac{\gamma_j}{2})\right)} + \mathrm{K}_{diff}\Delta_{spec}(l_i,\theta_i,\gamma_j)$$

où q représente le nombre d'acquisitions.

Ces points ont du fait de notre hypothèse des normales proches de la direction spéculaire pour l'image 1. On a donc $l_i \approx 0$ et $\theta_i \approx \frac{\gamma_1}{2}$. Afin de déterminer K_{ls} et σ_α nous posons deux hypothèses simplificatrices :

1. nous supposons que l'écart entre les intensité mesurées et l'intensité maximale de la direction spéculaire est essentiellement due a la variable θ_i et posons $l_i = 0$. Ceci revient à négliger le déplacement de la normale par rapport au plan contenant $\overrightarrow{k_1}$ et $\overrightarrow{k_r}$;
2. l'écart entre les intensités mesurées et l'intensité maximum est principalement du à la composante spéculaire pour les point sélectionnés. Nous posons donc $\theta_i = \frac{\theta^1}{2}$ pour la composante Lambertienne.

Sous ces hypothèses, l'intensité d'un pixel i dans l'image j est donnée par :

$$\begin{aligned} I_j(i) &= \frac{\mathrm{C}_{ls}}{\Delta_{spec}(1,\theta,0)} e^{-\frac{1}{\sigma^2}\left(1-\Delta_{spec}(l_i,\theta_i,\frac{\gamma_j}{2})\right)} + \mathrm{K}_{diff}\Delta_{spec}(l_i,\theta_i,\gamma_j) \\ &\approx \frac{\mathrm{C}_{ls}}{\cos(\theta_i)} e^{-\frac{1}{2\sigma_\alpha^2}\left(\theta_i-\frac{\gamma_j}{2}\right)^2} + \mathrm{K}_{diff}\cos(\tfrac{\theta^1}{2}+\gamma_j). \end{aligned}$$

Si nous supposons K_{diff} connu, nous avons 3 inconnues pour chaque pixel d'indice i ($\mathrm{C}_{ls}, \theta_i$ et σ_α). Il nous faut donc *a priori* 3 acquisitions ($q = 3$) qui nous donnent le système suivant pour chaque pixel :

$$\left\{\begin{array}{rcll} I_1 &=& \frac{\mathrm{C}_{ls}}{\cos\left(\frac{\gamma_1}{2}+\epsilon\right)} e^{-\frac{\epsilon^2}{2\sigma^2}} & + \quad \mathrm{K}_{diff}\cos(\frac{\theta^1}{2}) \\ I_2 &=& \frac{\mathrm{C}_{ls}}{\cos\left(\frac{\gamma_1}{2}+\epsilon\right)} e^{-\frac{1}{2\sigma^2}\left(\epsilon-\frac{\gamma_2}{2}\right)^2} & + \quad \mathrm{K}_{diff}\cos(\frac{\theta^1}{2}+\gamma_2) \\ I_3 &=& \frac{\mathrm{C}_{ls}}{\cos\left(\frac{\gamma_1}{2}+\epsilon\right)} e^{-\frac{1}{2\sigma^2}\left(\epsilon-\frac{\gamma_3}{2}\right)^2} & + \quad \mathrm{K}_{diff}\cos(\frac{\theta^1}{2}+\gamma_3) \end{array}\right. \tag{2.39}$$

où $\epsilon = \theta_i - \frac{\gamma_1}{2}$ et I_1, I_2, I_3 représentent respectivement les intensités du pixel sélectionné dans les images 1, 2 et 3.

Ce système de 3 équations a 3 inconnues ce résout simplement en appliquant l'opérateur log sur les deux membres de chaque équation afin d'obtenir un système plus facile à manipuler.

La résolution de ce système nous permet d'obtenir une estimation de C_{ls} et σ_α pour chacun des pixels sélectionnés. Sachant que les hypothèses que nous avons faites pour résoudre ce système seront d'autant plus valables que l'intensité dans l'image 1 est importante, un moyen simple de combiner ces résultats est de pondérer les estimations obtenues pour chaque pixel d'indice i par $I_1(i)$. On obtient donc :

$$\mathrm{C}_{ls} = \frac{1}{\sum_{i=1}^n I_1(i)} \sum_{i=1}^n I_1(i) {\mathrm{C}_{ls}}^i \quad \text{et } \sigma_\alpha = \frac{1}{\sum_{i=1}^n I_1(i)} \sum_{i=1}^n I_1(i)\sigma_\alpha^i$$

où ${\mathrm{C}_{ls}}^i$ et σ_α^i représentent les estimations de C_{ls} et σ_α pour le pixel i.

Les estimations de C_{ls} et σ_α définies par l'équation précédente peuvent être affinées en minimisant la distance entre les intensités mesurées et les valeurs prédites par le modèle. L'intensité d'un pixel dont la normale est définie par les paramètres l et θ est égale à :

$$I = I_j(l, \theta, \mathrm{C}_{ls}, \sigma_\alpha) = \frac{C_{sl}}{\Delta_{spec}(1, \theta, 0)} e^{-\frac{1}{\sigma_\alpha^2}\left(1 - \Delta_{spec}(l, \theta, \frac{\gamma_j}{2})\right)} + \mathrm{K}_{diff}\Delta_{spec}(l, \theta, \gamma_j).$$

En utilisant la sélection définie précédemment, nous avons une première approximation σ_α^0 et ${\mathrm{C}_{ls}}^0$ de σ_α et C_{ls}. Nous savons de plus que pour chaque pixel d'indice i, $(l_i, \theta_i) \approx (0, \frac{\gamma_1}{2} + \epsilon_i)$ où $\epsilon_i = \theta_i - \frac{\gamma_1}{2}$ est une des inconnues de l'équation 2.39. Les 4 inconnues σ_α, C_{ls}, l_i et θ_i peuvent donc être estimées en minimisant la fonction H définie sur notre sélection avec 4 acquisitions et des valeur initiales de nos inconnues égales à $(0, \frac{\gamma_1}{2} + \epsilon_i, {\mathrm{C}_{ls}}^0, \sigma_\alpha^0)$:

$$H((l_i)_{i\in\{1,\ldots,n\}}, (\theta_i)_{i\in\{1,\ldots,n\}}, K_{sl}, \sigma_\alpha) = \sum_{j=1}^{4}\sum_{i=1}^{n} \left(I_j(i) - I_j(l_i, \theta_i, K_{sl}, \sigma_\alpha)\right)^2 .$$

La minimisation de la fonction H permet d'atténuer l'importance des hypothèses faites pour obtenir une estimation initiale. Nous avons minimisé H à l'aide de la méthode du gradient conjugué qui permet facilement d'introduire des notions telles que "trouver l'optimum local le plus proche de la solution initiale". Toutefois, la résolution du système défini par l'équation 2.39 a fourni lors de nos expérimentations des valeurs très proches des valeurs théoriques si bien que l'amélioration induite par l'optimisation de la fonction H est souvent faible.

2.5.4.c Discussion

L'estimation du paramètre K_{diff} est sans doute le point faible de notre méthode d'estimation. En effet la méthode proposée est basée sur la sélection d'un ensemble de pixels dont l'intensité est inférieure à un seuil η. Dans le cas d'images bruitées et pour des valeurs trop faibles de η le bruit dans l'image peut considérablement gêner l'estimation. Nous envisageons pour remédier à ce problème d'effectuer l'acquisition de plusieurs images pour un même illuminant et de prendre comme intensité l'intensité moyenne obtenues sur toutes les acquisitions. Cette méthode est similaire à celle utilisé par Kay [118] (section 2.5.3) et peut permettre une forte diminution du bruit si celui-ci est additif. On peut également envisager d'utiliser l'écart-type σ_i des intensités calculé sur les différentes acquisitions d'un pixel i et rejeter de la sélection les pixels tels que $\sigma_i \geq \frac{\eta}{p}$ où p est un paramètre à estimer.

Il serait également tentant d'estimer comme pour K_{diff}, les paramètres C_{ls} et σ_α à l'aide de moyennes calculées sur une seule image. Une telle méthode nous permettrait de réduire le nombre d'illuminants. Toutefois la densité de probabilité la plus adaptée pour la variable θ serait une Gaussienne de moyenne $\frac{\gamma_1}{2}$. La densité de probabilité de l pourrait être obtenue en tenant compte de $l = \sin^2(\frac{\psi}{2})$ et en supposant que l'angle ψ entre le vecteur $\vec{n}$ et le plan (Oxy) a une distribution normale de moyenne nulle. Il resterait également à déterminer les écarts types des distributions de ψ et θ. Nous n'avons toutefois pas encore eu le temps de nous pencher sur ce problème.

2.6 Reconstruction $3D$ à partir des modèles de réflexion

La reconstruction de surfaces occupe sans aucun doute une part très importante de la littérature et certaines grandes conférences de Vision telle que ICPR y consacrent des sessions entières. Toutefois, les premières méthodes de reconstruction étaient basées sur le modèle Lambertien. Ces méthodes donnent des résultats intéressants sur des images de synthèse mais échouent généralement lors de la reconstruction d'objets réels. L'introduction de modèles physiques dans les méthodes de reconstruction est assez récente puisqu'elle date approximativement des années 90. Nous allons donc rappeler les fondements des méthodes de reconstruction à partir du modèle Lambertien avant d'examiner les différentes méthodes basées sur des modèles physiques et présenter notre contribution à cette problématique.

2.6.1 Reconstruction Lambertienne

La reconstruction Lambertienne est basée sur une inversion de l'équation

$$I(\vec{n}) = \mathrm{K}_{diff} \cos(\theta)$$

où θ représente l'angle entre la normale $\vec{n}$ et la source.

Cette équation est souvent normalisée afin de supprimer K_{diff} (section 2.5.1). On obtient alors :

$$R(\vec{n}) = \cos(\theta)$$

où $R(\vec{n}) = \frac{I(\vec{n})}{\mathrm{K}_{diff}}$ est supposé connu.

Étant donné un repère $Oxyz$ où x et y correspondent aux coordonnées de l'image et z à l'altitude d'un point sur la surface, la normale $\vec{n}$ peut également s'exprimer à l'aide des dérivées partielles de z par rapport à x et y. On a en effet :

$$\vec{n} = \begin{pmatrix} -p \\ -q \\ 1 \end{pmatrix} \text{ avec } p = \frac{\partial z}{\partial x} \text{ et } q = \frac{\partial z}{\partial y}.$$

L'angle θ à lui seul ne permet pas de déterminer $\vec{n}$. Géométriquement la connaissance de l'angle θ fournit un ensemble de solutions correspondant à un cône centré sur le vecteur source et d'angle θ. La détermination des normales à partir d'une seule source impose donc d'ajouter des contraintes supplémentaires [204]. Ces contraintes sont souvent exprimées sous forme d'une énergie à minimiser si bien que la détermination des normales revient à minimiser un terme de la forme :

$$\int\int \left(\sum_{i=1}^{p} \lambda_i \mathcal{E}_i(x,y) \right) dxdy$$

où $\mathcal{E}_i(x,y)$ et λ_i représentent respectivement l'énergie liée à la contrainte i au point (x,y) et la pondération de cette contrainte. L'indice p code le nombre de contraintes.

Les contraintes usuellement utilisées dans ce type de méthode sont :

Contrainte d'intensité : Il s'agit simplement de contraindre le modèle à correspondre aux valeurs observées. On minimise pour cela l'expression :

$$\int\int (I^{obs}(x,y) - I^{mod}(x,y))^2 dxdy$$

où I^{obs} et I^{mod} correspondent respectivement aux intensités mesurées et aux intensités prédites par le modèle.

Contrainte de gradient : Contraint les gradients de l'image et les gradients obtenus à partir du modèle à correspondre :

$$\int\int \left((\bigtriangledown_x I^{obs}(x,y) - \bigtriangledown_x I^{mod}(x,y))^2 + (\bigtriangledown_y I^{obs}(x,y) - \bigtriangledown_y I^{mod}(x,y))^2 \right) dxdy$$

où $\bigtriangledown_x I^{obs}(x,y), \bigtriangledown_y I^{obs}(x,y), \bigtriangledown_x I^{mod}(x,y), \bigtriangledown_y I^{mod}(x,y)$ représentent respectivement les dérivées des intensités mesurées et modélisées suivant les directions x et y.

Contrainte de Courbure : Cette contrainte impose à la surface des variations douces de la normale. Cette information *a priori* sur la surface stabilise la convergence vers une solution unique :

$$\int\int (p_x^2 + p_y^2 + q_x^2 + q_y^2) dxdy$$

où p_x, p_y, q_x, q_y représentent respectivement les dérivées de p et q par rapport à x et y.

Cette contrainte peut être relâchée en contraignant uniquement les variations d'altitudes suivant les directions x et y (donc sans considérer $\frac{\partial^2 z}{\partial x \partial y}$ et $\frac{\partial^2 z}{\partial y \partial x}$) :

$$\int\int (p_x^2 + q_y^2) dxdy.$$

Contrainte de différentiabilité : Cette contrainte impose à la surface d'être au moins de classe C^2 en respectant la contrainte $\frac{\partial^2 z}{\partial x \partial y} = \frac{\partial^2 z}{\partial y \partial x}$. Ceci peut être obtenu en minimisant :

$$\int\int (p_y - q_x)^2 dxdy.$$

Cette contrainte est également appelée la contrainte d'intégration. En effet, étant donné deux points A et B et un chemin Γ reliant ces deux points dans l'image l'altitude de B peut être déduite de celle de A en utilisant l'intégration :

$$Z(B) = Z(A) + \int_\Gamma p dx + q dy.$$

Toutefois si la contrainte de différentiabilité n'est pas vérifiée l'altitude du point B dépendra du chemin Γ utilisé pour relier les deux points. La surface est dite dans ce cas non intégrable puisque l'on ne peut la retrouver de manière unique à partir de ses dérivées codées par p et q.

Contrainte sur la norme du gradient : Cette contrainte impose au gradient d'avoir une norme à 1. On minimise pour cela :

$$\int\int (\|\vec{n}(x,y)\|^2 - 1) dxdy.$$

Par exemple, la méthode de Brooks et Horn [28] minimise le terme :

$$\int\int \left((I^{obs}(x,y) - I^{mod}(x,y))^2 + \alpha_1(p_x^2 + q_y^2) + \alpha_2(\|\vec{n}(x,y)\|^2 - 1)\right) dxdy$$

où α_1 et α_2 sont les poids associés aux contraintes de courbure et de norme du gradient.

Worthington et Hancock [197] utilisent une contrainte un peu plus sophistiquée :

$$\int\int \left((I^{obs}(x,y) - I^{mod}(x,y))^2 + \lambda(x,y)(p_x^2 + q_y^2)\right) dxdy$$

où $\lambda(x,y)$ est une fonction du point (x,y). L'idée étant de relâcher la contrainte de faible courbure lorsque les itérations de l'algorithme de minimisation tendent à montrer qu'il y a une forte probabilité pour qu'il existe une forte courbure au point (x,y).

La détermination des normales à partir d'un seul illuminant est un problème mal posé que l'on résout généralement à l'aide de procédures de minimisation. Notons que la pondération à apporter aux différentes contraintes est souvent assez difficile à établir *a priori* tout en restant un des paramètres fondamentaux de la méthode. On se retrouve ici avec un problème similaire à celui des contours actifs [117] où la pondération entre la force du gradient, les contraintes de tensions et les contraintes de courbures sont souvent difficiles à établir *a priori* mais influencent de façon déterminante le résultat final.

Si nous utilisons deux acquisitions, nous avons en chaque point un couple d'équations :

$$\begin{array}{rcl} I_1(x,y) & = & \mathrm{K}_{diff}\cos(\theta_1) \\ I_2(x,y) & = & \mathrm{K}_{diff}\cos(\theta_2) \end{array}$$

où θ_1 et θ_2 représentent respectivement l'angle entre la normale et les sources 1 et 2.

L'ensemble des solutions de ce couple d'équations peut géométriquement s'interpréter comme l'intersection des deux cônes centrés sur chacune des directions d'illuminants et d'angles respectifs θ_1 et θ_2. On a donc *a priori* deux solutions en chaque point [121]. Ceci est assez clair si l'on étudie le problème simplifié où l'une des sources est alignée avec l'axe z et l'autre est située dans le plan (Ozx). Si nous utilisons un repère où l'axe z est orienté de la source vers le point l'on a :

$$\left\{ \begin{array}{rcl} \overrightarrow{s_1} & = & (0,0,-1) \\ \overrightarrow{s_2} & = & (\alpha,0,\beta) \text{ avec } \alpha^2+\beta^2 = 1 \\ \overrightarrow{n} & = & (p,q,-1). \end{array} \right.$$

où $\overrightarrow{s_1}$ et $\overrightarrow{s_2}$ représentent les vecteurs directeurs des deux sources lumineuses.

On obtient dans ce cas :

$$\begin{array}{rcccl} NI_1(x,y) & = & \frac{\vec{s_1}.\vec{n}}{\|\vec{n}\|} & = & \frac{1}{\sqrt{p^2+q^2+1}} \\ NI_2(x,y) & = & \frac{\vec{s_2}.\vec{n}}{\|\vec{n}\|} & = & \frac{\alpha p-\beta}{\sqrt{p^2+q^2+1}} \end{array}$$

où $NI_1(x,y)$ et $NI_2(x,y)$ représentent les intensités normalisées $\frac{I_1(x,y)}{\mathrm{K}_{diff}}$ et $\frac{I_2(x,y)}{\mathrm{K}_{diff}}$.

Les solutions de ce système d'équation sont :

$$\begin{aligned} p(x,y) &= \frac{1}{\alpha}\left(\beta + \frac{NI_2(x,y)}{NI_1(x,y)}\right) \\ q(x,y) &= \epsilon(x,y)\sqrt{NI_1^{-2}(x,y) - p^2 - 1} \\ &= \epsilon(x,y).\frac{\sqrt{\Lambda(x,,y)}}{\alpha.NI_1(x,y)} \end{aligned}$$

avec $\epsilon(x,y) \in \{+1,-1\}$ et

$$\Lambda(x,y) = \left[1 - NI_1^2(x,y) - NI_2^2(x,y)\right] - < \vec{s_1}|\vec{s_2} > [< \vec{s_1}|\vec{s_2} > -2NI_1(x,y)NI_2(x,y)]$$

où $< \vec{s_1}|\vec{s_2} >$ représente le produit scalaire de $\vec{s_1}$ et $\vec{s_2}$.

On peut montrer [121] que si la surface est supposée C^1, la valeur de la fonction $\epsilon(x,y)$ reste constante sur tout domaine tel que $\Lambda(x,y) \neq 0$. Ce résultat peut intuitivement s'interpréter comme une interdiction faite à une des composante de la normale de changer brutalement de signe sur une surface C^1.

Notons que si nous supposons la surface de classe C^2, l'ambiguïté sur q peut souvent être levée sur les domaines $\Lambda(x,y) \neq 0$ par la contrainte :

$$\frac{\partial^2 z}{\partial x \partial y} = \frac{\partial p}{\partial y} = \frac{\partial^2 z}{\partial y \partial x} = \frac{\partial q}{\partial x}.$$

Finalement, Horn [102] a montré que dans le cas de trois sources lumineuses la normale pouvait être déterminée de façon unique. Intuitivement la troisième source lumineuse nous permet de lever l'ambiguïté sur le signe.

2.6.2 Reconstructions basées sur des modèles physiques

Une majorité de méthodes basées sur les modèles physiques utilisent plusieurs acquisitions et une caméra fixe (section 2.5.3). Dans ce cadre, la normale est considérée comme un paramètre au même titre de C_{ls} et K_{diff} et est déterminée à l'aide d'un nombre important d'acquisitions.

2.6.2.a Reconstruction par minimisation

Lee [128] utilise une technique inspirée des méthodes de minimisation appliquées aux modèles Lambertiens (section 2.6.1). Le modèle physique utilisé par Lee [128] correspond à celui de Torrance-Sparrow (section 2.2.3) avec une composante Lambertienne et un lobe spéculaire. Lee simplifie le problème de l'estimation des normales en utilisant un pavage triangulaire de l'image. Ce pavage triangulaire de l'image correspond à une triangularisation de la surface et permet d'exprimer en fonction des coordonnées des sommets de chaque triangle :

1. la normale à la surface ;
2. les paramètres du modèle physique (figure 2.6) :
   - l'angle θ_i entre la normale et la direction de la source $\vec{k_i}$,
   - l'angle θ_r entre la normale et la direction d'observation $\vec{k_r}$,
   - l'angle α entre la normale et la bissectrice ν de $\vec{k_i}$. et $\vec{k_r}$.

Sachant que les coordonnées X et Y des sommets du triangle sont déterminées par les coordonnées de leur projection sur l'image, le problème ce ramène à la détermination de l'altitude Z des sommets du triangle.

Étant données J acquisitions de la scène avec J illuminants différents, Lee minimise l'expression suivante en fonction de l'altitude des sommets des triangles :

$$\mathcal{E} = \sum_{k=1}^{N_t} \sum_{j=1}^{J} \mathcal{E}_k^j = \sum_{k=1}^{N_t} \sum_{j=1}^{J} (I_k^{j\,obs} - R_k^j)^2$$

où N_t représente le nombre de triangles et $I_k^{j\,obs}$ et R_k^j représentent respectivement l'intensité mesurée et prédite par le modèle sur le triangle k dans l'image j.

De plus, une linéarisation de la fonction de réflexion basée sur une décomposition de Taylor à l'ordre 1 permet à Lee d'exprimer l'énergie du système sous la forme :

$$\mathcal{E} = \frac{1}{2} z^t A z - b^t z + c$$

où z est le vecteur d'altitudes des triangles et A une matrice symétrique.

Le problème de minimisation se ramène donc à la résolution du système $Az = b$. Notons toutefois que le modèle de Torrance Sparrow comporte une exponentielle qui est mal approximée par la formule de Taylor à l'ordre 1.

2.6.2.b Reconstruction par rotation d'objet

Une méthode originale a également été présentée par Lu et Little [138]. Cette méthode est basée sur une rotation de l'objet suivant un axe Yqui constitue donc un axe de symmétrie de l'objet. La caméra et la source lumineuse restent fixes et positionnées sur l'axe Z (figure 2.20).

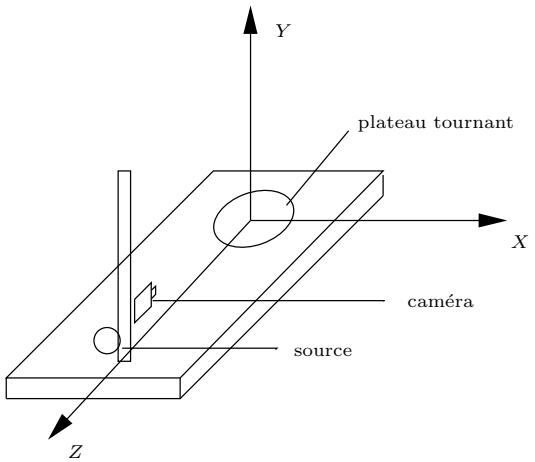


FIG. 2.20 – *Le système d'acquisition de Lu et Little [138]*

La caméra et la source lumineuse étant co-linéaires les angles θ_i, θ_r et α entre la normale $\vec{n}$ et respectivement le vecteur source $\vec{k_i}$, le vecteur d'observation $\vec{k_r}$ et la bissectrice de $\vec{k_i}$ et $\vec{k_r}$ sont confondus. Tous les modèles de réflexions basés sur des surfaces isotropiques sont fonctions de ces 3 paramètres ou d'un sous ensemble de ceux-ci. La fonction de réflectance $R(\theta_i, \theta_r, \alpha)$ peut donc dans ce cas s'écrire simplement comme une fonction $R(\theta)$ de l'angle θ entre la normale et l'axe Z (qui porte les vecteurs $\vec{k_i}$ et $\vec{k_r}$).

La méthode de Lu et Little consiste à déterminer la fonction de réflectance d'un objet en recherchant les points d'intensité maximum dans l'image originale $image_0$. Lu et Little assument que ces points correspondent à des zones spéculaires de l'image pour lesquels on a $\theta = 0$. Après une rotation de l'objet de $\frac{\pi}{2}$, ces points deviennent des points de contour dans l'image $image_{\frac{\pi}{2}}$ avec $\theta = \frac{\pi}{2}$. Le lieu décrit par un point lors d'une rotation de 0 à $\frac{\pi}{2}$ est un segment horizontal du fait de la rotation autour de l'axe Y. Lu peut donc grâce à la rotation de l'objet mesurer sa réflectance pour des angles allant de 0 à $\frac{\pi}{2}$. Lu suppose de plus que la fonction $R(\theta)$ est monotone ce qui lui permet de calculer la fonction inverse $R^{-1}(I)$ donnant l'angle θ entre la normale d'un point et l'axe Z en fonction de son intensité.

Étant donné l'image initiale de l'objet $image_0$ et une image $image_\alpha$ du même objet après une rotation de α, les angles θ_0 et θ_α codant l'angle entre la normale d'un point et l'axe Z dans l'état initial et après une rotation de α sont liés par :

$$\cos(\theta_\alpha) = \cos(\theta_0)\,(\cos(\alpha) - p_0 \sin(\alpha))$$

où $\vec{n}_0 = (-p_0, -q_0, 1)$ représente les coordonnées de la normale avant la rotation. Étant données les intensités I_α et I_0 de ce point dans $image_\alpha$ et $image_0$, on obtient par la fonction R^{-1} les cosinus des angles θ_0 et θ_α : $\cos(\theta_0) = \cos(R^{-1}(I_0))$ et $\cos(\theta_\alpha) = \cos(R^{-1}(I_\alpha))$. La coordonnée p_0 s'obtient ensuite facilement à partir de l'équation précédente :

$$p_0 = \frac{1}{\tan(\alpha)} - \frac{1}{\sin(\alpha)} \frac{\cos(\theta_\alpha)}{\cos(\theta_0)}.$$

Étant donné p_0, la coordonnée q_0 se déduit immédiatement par :

$$\cos(\theta_0) = \frac{1}{\sqrt{p_0^2 + q_0^2 + 1}} \Rightarrow q_0 = \pm \sqrt{\frac{1}{\cos(R^{-1}(I_0))} - p_0^2 - 1}.$$

La méthode de Lu et Little [138] permet de reconstruire une image sans préciser la fonction de réflectance tout en faisant des hypothèses sur celle-ci compatibles avec les modèles physiques existants. Toutefois, cette méthode réclame un nombre important d'acquisitions de façon à pouvoir interpoler $R(\theta)$ et ne peut s'appliquer que sur des objets possédant un axe de symétrie.

2.6.3 Notre contribution à la reconstruction

Comme nous l'avons mentionné dans la section 2.1, notre méthode de reconstruction doit être applicable dans une cadre industriel dans lequel :

1. Les temps de calculs doivent être aussi courts que possible,
2. Le protocole d'acquisition doit rester simple.

La première contrainte nous a conduit à rejeter les méthodes utilisant des procédures de minimisations itératives. La seconde contrainte nous a conduit à essayer de minimiser le nombre d'illuminants. Nous allons montrer dans la suite de cette section que les normales à la surface peuvent être retrouvées à l'aide de seulement deux illuminants si nous supposons la surface à reconstruire optiquement homogène (Hypothèse 3, section 2.5.4). L'algorithme de reconstruction travaillera donc sur deux images prises du même point avec deux illuminants placés différemment.

Notre méthode de reconstruction est basée, tout comme notre méthode d'estimation des paramètres (section 2.5.4) sur le modèle de Nayar (section 2.2.4) :

$$I = \mathrm{K}_{diff}\cos(\theta_i) + \frac{C_{ls}}{\cos(\theta_r)} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}}$$

où θ_i, θ_r et α représentent l'angle entre la normale et respectivement le vecteur source $\vec{k_i}$, le vecteur d'observation $\vec{k_r}$ et la bissectrice $\vec{\nu}$ de $\vec{k_i}$ et $\vec{k_r}$.

Un des inconvénients de la formule de Nayar vient de l'utilisation simultanée de termes en cosinus $(\cos(\theta_r), \cos(\theta_i))$ et de l'angle α. Certain auteurs [179, 118] tournent d'ailleurs cette difficulté en approximant $\cos(\alpha)$ par $1 - \frac{1}{2}\alpha^2$ si bien que α^2 peut être remplacé par $2(1-\cos(\alpha))$. Nous avons utilisé cette approximation dans le cadre de l'estimation des paramètres (section 2.5.4). Toutefois, nous avons décidé de résoudre cette difficulté dans le cadre de la reconstruction en remplaçant α par $\tan(\alpha)$. On obtient donc :

$$I = \mathrm{K}_{diff}\cos(\theta_i) + \frac{C_{ls}}{\cos(\theta_r)} e^{-\frac{\tan^2(\alpha)}{2\sigma_\alpha^2}}. \tag{2.40}$$

Notons que si nous nous référons au modèle de Beckmann [8, 149] (section 2.2.2) le lobe spéculaire est défini par (équations 2.16 et 2.17) :

$$e^{-\frac{\nu_{xy}^2 T^2}{4\nu_z^2\sigma_h^2}} = e^{-\frac{\tan^2(\alpha)}{2\sigma^2}} \text{ avec } \begin{cases} \tan(\alpha) & = & \frac{\nu_{xy}}{\nu_z} \\ \sigma^2 & = & \frac{2\sigma_h^2}{T^2}. \end{cases}$$

On peut donc renverser le raisonnement en considérant que le modèle de Torrance-Sparrow est une approximation du modèle de Beckmann où l'on approxime $\tan(\alpha)$ par α. Notre modification du modèle de Nayar nous rapproche donc du modèle de Beckmann qui reste valide sur une plus grande variété de surfaces. Cette substitution devrait nous fournir davantage de précision plutôt qu'une approximation supplémentaire.

Notons toutefois que l'équation 2.40 n'est pas définie pour $\alpha = \frac{\pi}{2}$. De plus, le modèle du lobe spéculaire n'est certainement plus valide pour de telles valeurs de α. Il nous faudra donc chercher pour chaque image une condition suffisante pour que $\tan(\alpha)$ soit inférieur à une constante ω donnée. Plus précisément nous allons chercher dans nos deux acquisitions les constantes Ω_i $i \in \{1, 2\}$ telles que :

$$I_i > \Omega_i \Rightarrow \tan(\alpha_i) \leq \omega.$$

Nous pouvons effectuer la classification suivante en fonction des constantes Ω_i :

1. si $I_i > \Omega_i$, l'intensité sera définie par le modèle de Nayar modifié :

$$I_i = \frac{C_{ls}}{\cos(\theta_r)} e^{-\frac{\tan^2(\alpha_i)}{2\sigma_\alpha^2}} + \mathrm{K}_{diff}\cos(\theta_i);$$

2. si $I \leq \Omega_i$, l'intensité du lobe spéculaire est jugée négligeable et l'intensité est uniquement décrite par la composante Lambertienne :

$$I_i = \mathrm{K}_{diff} \cos(\theta_i).$$

La reconstruction de la surface à partir des normales impose de connaître une variation en z à partir d'une variation en x ou y. Nous ne pouvons donc plus comme dans les sections précédentes considérer $\vec{n}$ comme un vecteur normé. Nous simplifions toutefois le système en supposant que le vecteur $\vec{n}$ à une coordonnée positive sur l'axe z (Il regarde donc la caméra). On peut donc normaliser ce vecteur en fixant la composante z à 1. On obtient alors :

$$\vec{n} = \begin{pmatrix} u_x \\ u_y \\ 1 \end{pmatrix}.$$

Les variables u_x et u_y sont donc liées aux variations sur la surface par :

$$u_x = -\frac{\partial z}{\partial x} \text{ et } u_y = \frac{\partial z}{\partial y}.$$

Les conditions expérimentales que nous avons fixées dans la section 2.5.4 (figure 2.19) imposaient que les vecteurs $\vec{k_r}$, $\vec{k_1}$ et $\vec{k_2}$ soient coplanaires avec $\vec{k_1}$ dans le secteur angulaire défini par $\vec{k_r}$ et $\vec{k_2}$. Ceci nous donnait un ensemble d'équations paramétrées par les constantes (γ_1, γ_2) où γ_i représente l'angle entre $\vec{k_r}$ et $\vec{k_i}$. Nous allons préciser d'avantage les conditions expérimentales en imposant $\gamma_1 = 0$. La première source et la direction d'observation sont donc confondues.

Étudions à présent l'expression de l'intensité dans les deux images. Afin d'indiquer dans quelle image nous calculons les angles, nous indexons les variables θ_i, θ_r et α par l'indice de l'image correspondante. Ainsi les angles θ_i^1, θ_r^1 et α^1 correspondent aux angles θ_i, θ_r et α mesurés dans l'image 1. Notons que la direction d'observation $\vec{k_r}$ restant fixe, nous avons $\theta_r^1 = \theta_r^2$.

2.6.3.a Intensités dans l'image 1

L'image 1 est illuminée par une source dont la direction est parallèle à la direction d'observation. On a donc dans ce cas :

$$\vec{k_1} = \vec{k_r} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \Rightarrow \vec{\nu_1} = \frac{\vec{k_1} + \vec{k_r}}{\|\vec{k_1} + \vec{k_r}\|} = \vec{k_1}.$$

Le vecteur $\vec{\nu_1}$ étant confondu avec $\vec{k_1}$ les angles α_1, θ_i^1 et θ_r^1 sont égaux. Si nous notons ces angles par θ_1 nous avons :

$$\tan^2(\alpha^1) = \frac{\sin^2(\alpha^1)}{\cos^2(\alpha^1)} = \frac{1 - \cos^2(\alpha^1)}{\cos^2(\alpha^1)} = -1 + \frac{1}{\cos^2(\alpha^1)} == -1 + \frac{1}{\cos^2(\theta_1)}.$$

On a donc en reprenant l'équation 2.40 :

$$I_1 = \mathrm{C}_{ls}\sqrt{1 + tan^2(\theta_1)}e^{-\frac{\tan^2(\theta_1)}{2\sigma_\alpha^2}} + \frac{\mathrm{K}_{diff}}{\sqrt{1 + tan^2(\theta_1)}} = \frac{\mathrm{K}_{diff}}{\sqrt{1 + tan^2(\theta_1)}} + Spec(tan(\theta_1))$$

avec $Spec(x) = \mathrm{C}_{ls}\sqrt{1+x^2}e^{-\frac{x^2}{2\sigma_\alpha^2}}$.

J'ai montré avec Laurent Hussenet [109] que la fonction *Spec* est strictement décroissante dans $\mathbb{R}_+$. Ainsi, si $tan(\theta_1)$ est strictement supérieur à une constante ω, on a :

$$\left\{\begin{array}{lcl} f(\tan(\theta_1)) & < & f(w) = C_{ls}\sqrt{1+\omega^2}e^{-\frac{\omega^2}{2\sigma_\alpha^2}} \\ \frac{\mathrm{K}_{diff}}{\sqrt{1+tan^2(\theta_1)}} & < & \frac{\mathrm{K}_{diff}}{\sqrt{1+\omega^2}} \end{array}\right.$$

et pour $\tan(\theta_1) > \omega$ nous avons :

$$I_1 = \frac{C_{ls}}{\cos(\theta_1)}e^{-\frac{\tan^2(\theta_1)}{2\sigma_\alpha^2}} + \mathrm{K}_{diff}\cos(\theta_1) < \frac{1}{\sqrt{w^2+1}}\left((w^2+1)C_{ls}e^{-\frac{\omega^2}{2\sigma_\alpha^2}} + \mathrm{K}_{diff}\right).$$

On fixe donc :

$$\Omega_1 = \frac{1}{\sqrt{w^2+1}}\left((w^2+1)C_{ls}e^{-\frac{\omega^2}{2\sigma_\alpha^2}} + \mathrm{K}_{diff}\right)$$

Ainsi, si $I_1 > \Omega_1$, $\tan(\alpha^1) \leq \omega$ et :

$$I_1 = \frac{C_{ls}}{\cos(\theta_1)}e^{\frac{1}{2\sigma_\alpha^2}\left(1-\frac{1}{\cos^2(\theta_1)}\right)} + \mathrm{K}_{diff}\cos(\theta_1).$$

L'intensité est décrite dans ce cas comme une fonction continue d'une seule variable. De plus :

$$\frac{dI_1}{d\theta_1} = -\frac{\sin(\theta_1)}{\cos^4(\theta_1)}\left[\left(\frac{1}{\sigma_\alpha^2} - \cos^2(\theta_1)\right)C_{ls}e^{\frac{1}{2\sigma_\alpha^2}\left(1-\frac{1}{\cos^2(\theta_1)}\right)} + \mathrm{K}_{diff}\cos^4(\theta_1)\right].$$

Si nous nous référons à Kay [118], $\frac{1}{\sigma_\alpha^2}$ varie entre 1 et 20. L'expression $\left(\frac{1}{\sigma_\alpha^2} - \cos^2(\theta_1)\right)$ est donc positive et I_1 est une fonction décroissante de θ_1. L'angle θ_1 peut donc être retrouvé par une simple dichotomie.

De même, si $I_1 \leq \Omega_1$, nous avons plus simplement :

$$I_1 = \mathrm{K}_{diff}\cos(\theta_1) \Rightarrow \cos(\theta_1) = \frac{I_1}{K_{diff}}.$$

Nous pouvons donc dans les deux cas retrouver $\cos(\theta_1)$. La valeur θ_1 étant l'angle entre les vecteurs $\vec{k_1}$ et $\vec{n}$, on a :

$$\cos(\theta_1) = \vec{k_1}.\vec{n} = \frac{1}{\sqrt{u_x^2+u_y^2+1}}. \tag{2.41}$$

2.6.3.b Intensités dans l'image 2

L'image 2 est prise avec un illuminant faisant un angle γ_2 avec l'axe Z (figure 2.19). On a donc :

$$\vec{k_r} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},\ \vec{k_2} = \begin{pmatrix} \sin(\gamma_2) \\ 0 \\ \cos(\gamma_2) \end{pmatrix} \text{ et } \vec{\nu_2} = \frac{\vec{k_2}+\vec{k_r}}{\|\vec{k_2}+\vec{k_r}\|} = \begin{pmatrix} \sin(\frac{\gamma_2}{2}) \\ 0 \\ \cos(\frac{\gamma_2}{2}) \end{pmatrix}.$$

Nous obtenons donc :

$$\begin{array}{rcccl} \cos(\alpha^2) & = & \vec{\nu_2}.\vec{n} & = & \frac{\sin(\frac{\gamma_2}{2})u_x+\cos(\frac{\gamma_2}{2})}{\sqrt{u_x^2+u_y^2+1}} \\ \cos(\theta_i^2) & = & \vec{k_2}.\vec{n} & = & \frac{\sin(\gamma_2)u_x+\cos(\gamma_2)}{\sqrt{u_x^2+u_y^2+1}}. \end{array}$$

L'angle θ_r^2 reste inchangé puisque la direction d'observation $\vec{k_r}$ reste fixe entre les deux acquisitions. On a donc $\theta_r^2 = \theta_1$.

De plus, si nous utilisons l'équation 2.41, on a :

$$\begin{array}{rcccl} \cos(\alpha_2) & = & \vec{\nu_2}.\vec{n} & = & \left(\sin(\frac{\gamma_2}{2})u_x + \cos(\frac{\gamma_2}{2})\right)\cos(\theta_1) \\ \cos(\theta_2) & = & \vec{k_2}.\vec{n} & = & (\sin(\gamma_2)u_x + \cos(\gamma_2))\cos(\theta_1). \end{array} \tag{2.42}$$

La constante Ω_2 telle que $I_2 > \Omega_2 \Rightarrow \tan(\alpha^2) \leq \omega$ peut se calculer [109] par un procédé analogue à celui effectué pour l'image 1. On obtient :

$$\Omega_2 = \frac{C_{sl}}{\cos(\theta_1)} e^{-\frac{w^2}{2\sigma_\alpha^2}} + \mathrm{K}_{diff}\left(2\frac{\cos(\frac{\gamma_2}{2})}{\sqrt{\omega^2+1}} - \cos(\theta_1)\right).$$

Notez que le seuil Ω_2 dépend de l'angle θ_1 déterminé dans l'image 1. Décomposons à présent les cas en fonction de la valeur de I_2. Nous avons en utilisant l'équation 2.42 :

1. si $I_2 > \Omega_2$

$$\begin{array}{rcl} I_2 & = & \frac{C_{ls}}{\cos(\theta_1)} e^{\frac{1}{2\sigma_\alpha^2}\left(1-\frac{1}{\cos^2(\alpha_2)}\right)} + \mathrm{K}_{diff}\cos(\theta_2) \\ & = & \frac{C_{ls}}{\cos(\theta_1)} e^{\frac{1}{2\sigma_\alpha^2}\left(1-\frac{1}{\cos^2(\theta_1)\left(\sin(\frac{\gamma_2}{2})u_x+\cos(\frac{\gamma_2}{2})\right)^2}\right)} + \mathrm{K}_{diff}\left(\sin(\gamma_2)u_x + \cos(\gamma_2)\right)\cos(\theta_1). \end{array}$$

   L'intensité I_2 n'est alors plus fonction que d'une seule variable u_x. Notons de plus que :

$$\frac{dI_2}{du_x} = \frac{\sin(\frac{\gamma_2}{2})}{\sigma^2\cos^3(\alpha_2)} C_{ls} e^{\frac{1}{2\sigma^2}\left(1-\frac{1}{\cos^2(\alpha_2)}\right)} + \mathrm{K}_{diff}\cos(\theta_1)\sin(\gamma_2)$$

   Puisque $\theta_1 \in [0, \frac{\pi}{2}]$ et α_2 est supposé petit, nous avons $\frac{dI_2}{du_x} \geq 0$. L'intensité I_2 est donc une fonction croissante de u_x. Nous pouvons donc extraire u_x de l'équation par dichotomie. La coordonnée u_y se déduit de u_x par l'équation 2.41 :

$$u_y^2 = \frac{1}{\cos^2(\theta_1)} - u_x^2 - 1 \; ; \tag{2.43}$$

2. si $I_2 \leq \Omega_2$, nous avons :

$$I_2 = \mathrm{K}_{diff}\cos(\theta_2) \Rightarrow \cos(\theta_2) = \frac{I_2}{K_{diff}} = (\sin(\gamma_2)u_x + \cos(\gamma_2))\cos(\theta_1)$$

   d'où :

$$u_x = \frac{1}{\sin(\gamma_2)}\left(\frac{I_2}{\mathrm{K}_{diff}\cos(\theta_1)} - \cos(\gamma_2)\right).$$

   La valeur de u_y se déduit comme précédemment par l'équation 2.43.

Notons que nous retrouvons dans l'équation 2.43 la même indétermination du signe de u_y que dans le cas Lambertien (section 2.6.1). Nous résolvons cette difficulté en supposant la surface de classe C^2. Cette supposition impose l'égalité :

$$\frac{\partial^2 z}{\partial x \partial y} = -\frac{\partial u_x}{\partial y} = \frac{\partial^2 z}{\partial y \partial x} = -\frac{\partial u_y}{\partial x}.$$

Si nous posons :

$$u_y(x,y) = \epsilon(x,y)|u_y(x,y)| \text{ avec } \left\{ \begin{array}{lcl} |u_y(x,y)| & = & \sqrt{\frac{1}{\cos^2(\theta_1)} - u_x^2 - 1} \\ \epsilon(x,y) & \in & \{-1,1\}. \end{array} \right.$$

L'approximation des dérivées par des différences finies nous fournit en chaque point le système d'équation :

$$\frac{\partial u_x}{\partial y}(x,y) = \frac{\partial u_y}{\partial x} \Rightarrow u_x(x,y) - u_x(x,y-1) \approx \epsilon(x,y)|u_y(x,y)| - \epsilon(x-1,y)|u_y(x-1,y)|.$$

Nous avons donc pour chaque point (x,y), 4 possibilités selon les valeurs de $\epsilon(x,y)$ et $\epsilon(x-1,y)$. Nous appliquons une méthode directe en calculant pour chaque pixel les 4 possibilités et en retenant celle qui minimise la différence entre les deux membres de l'équation. Notez que nous ne traitons pas le cas où la valeur $\epsilon(x-1,y)$ calculée au pixel (x,y) est différente de celle calculée en $(x-1,y)$. Toutefois, malgré sa simplicité, cette méthode nous a donné de bons résultats sur nos images tests. Ces bons résultats nous ont conduit à préférer cette méthode à une méthode d'optimisation globale plus coûteuse.

La méthode précédente nous fournit en chaque point la normale définie par le vecteur $\vec{n}(x,y) = (-u_x(x,y), -u_y(x,y), 1)$. La reconstruction de l'altitude $Z(x,y)$ de chaque point s'effectue en utilisant un point d'altitude fixé arbitrairement et en utilisant une approximation de u_x et u_y par les différences finies :

$$\begin{array}{lcl} \frac{\partial Z}{\partial x}(x,y) = -u_x \approx Z(x,y) - Z(x-1,y) & \Rightarrow & Z(x-1,y) \approx Z(x,y) + u_x \\ \frac{\partial Z}{\partial y}(x,y) = -u_y \approx Z(x,y) - Z(x,y-1) & \Rightarrow & Z(x,y-1) \approx Z(x,y) + u_y. \end{array}$$

Donc connaissant $Z(x,y)$ on en déduit $Z(x-1,y)$ et $Z(x,y-1)$. Les altitudes $Z(x+1,y)$ et $Z(x,y+1)$ se déduisent suivant le même principe.

Notre algorithme de reconstruction est basé sur ce principe et est construit comme un algorithme de croissance de régions qui propage l'information d'altitude en utilisant l'image des normales. Notez que cet algorithme ne vérifie pas si la surface obtenue est C^2, donc si $\frac{\partial^2 z}{\partial x \partial y} = \frac{\partial^2 z}{\partial y \partial x}$. Cette méthode privilégie la rapidité au détriment de la robustesse, ce qui est l'objectif recherché. Notons toutefois que la qualité des images obtenues par croissance de régions nous a paru suffisante pour détecter les défauts de fabrications des pièces que nous devions contrôler. De plus, cette méthode peut servir d'initialisation à une méthode minimisant le terme :

$$\mathcal{E} = \sum (\frac{\partial Z}{\partial x} + u_x)^2 + (\frac{\partial Z}{\partial y} + u_y)^2$$

où $\frac{\partial Z}{\partial x}$ et $\frac{\partial Z}{\partial y}$ sont calculés par différences finies.

(a) 0° (b) 20°

FIG. 2.21 – Une image d'un buste de mozart prise avec illuminant de face (a) et illuminant à 20°. La reconstruction est affichée sur la Figure 2.22.

La figure 2.21 représente deux images de synthèse illuminées en utilisant le modèle de Nayar défini par l'équation 2.15 avec une source lumineuse au dessus de l'objet $\gamma_1 = 0^{\circ}$ (figure 2.21(a)) et faisant un angle de $\gamma_2 = 20^{\circ}$ par rapport à la verticale (figure 2.21(b)). La reconstruction $3D$ de l'image est représentée sur la figure 2.22. Notons que notre algorithme de reconstruction a utilisé un modèle légèrement différent de celui utilisé pour générer les images puisque le modèle utilisé pour la reconstruction utilise $\tan(\alpha)$ plutôt que α. Notre algorithme de reconstruction résiste donc bien à de légères perturbations du modèle.

2.6.3.c Discussions

Les méthodes de reconstruction sont avec les méthodes d'estimation de paramètres des outils parfaitement adaptés au contrôle de la forme, de la rugosité et des propriétés électriques d'objets $3D$. Ces propriétés déterminent les propriétés photométriques des objets. La méthode que nous avons développée en partenariat avec l'entreprise Axon Câble est spécifiquement conçue pour des objets métalliques et permet d'obtenir la forme $3D$ d'un objet ainsi que ces paramètres en des temps suffisamment courts avec un protocole d'acquisition simple. Nous comptons développer notre partenariat avec Axon Câble et étudier avec eux différentes adaptations de cette méthode aux différents objets manufacturés dont ils ont à effectuer un contrôle qualité.

2.7 Conclusion

Nous avons vu que les modèles de réflexion (section 2.2) permettent de décrire la radiance d'un point, et donc l'intensité ou la couleur du pixel correspondant en fonction de plusieurs paramètres dont la position de la source lumineuse, la position de l'observateur et les propriétés optiques du matériau (conducteur, isolant, homogène, inhomogène).

2.7.1 Segmentation en matériaux

Dans le cadre des images couleur, ces modèles permettent de caractériser l'ensemble des couleurs d'un matériau (section 2.4). Cette propriété est très intéressante dans la mesure où elle offre de

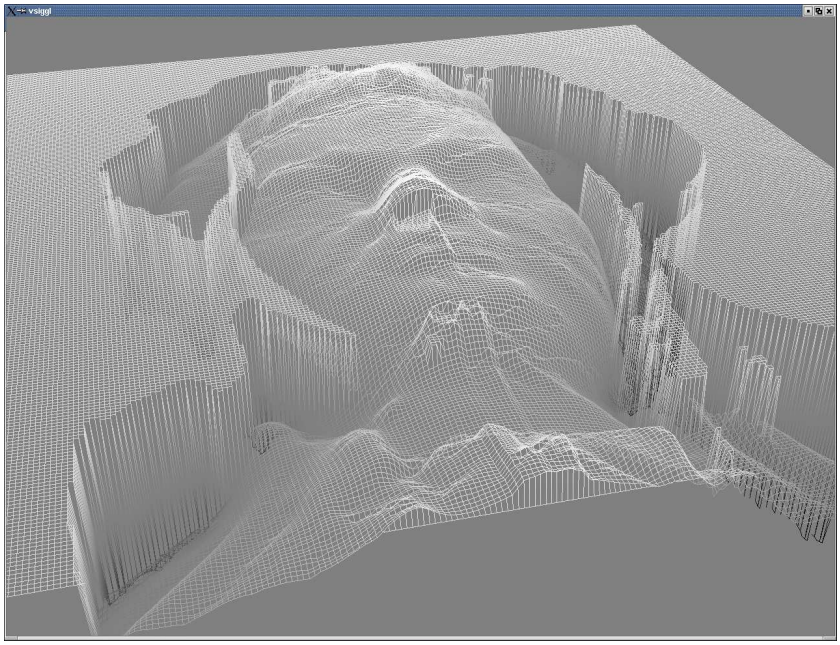


FIG. 2.22 – Une reconstruction du buste de Mozart à partir des deux images représentées sur la Figure 2.21

nouvelles perspectives aux méthodes de classification et de segmentation. En effet, les méthodes de classification appliquées aux images couleur créent une partition de l'ensemble des couleurs de l'image en sous ensembles appelés clusters. Les méthodes utilisées pour créer ces clusters considèrent souvent que la fonction de densité de chaque cluster peut être approximée par une Gaussienne centrée sur un point. D'autres méthodes plus évoluées définissent les clusters à partir des lignes de partage des eaux définies sur l'image de la fonction de densité [98]. Le problème commun à ces deux types de méthodes est que l'on cherche à regrouper des données sans connaître *a priori* la forme ou les propriétés que doivent avoir ces regroupements. L'étude des modèles de réflexion permet de résoudre ce problème en caractérisant *a priori* les objets à rechercher. De même dans le cadre de la segmentation la notion d'objet est souvent assez floue si bien que la segmentation est un problème intrinsèquement mal posé. Les méthodes de segmentation en matériaux ne peuvent généralement pas fournir une segmentation regroupant en une seule région chaque objet de la scène. Toutefois ces méthodes peuvent être utilisées comme pré-traitement de bas niveau en conjonction avec des algorithmes de regroupement de plus haut niveau effectuant des fusions sur des connaissances *a priori* de la composition des objets.

Notre travail sur ce type de méthodes a consisté à définir une méthode de segmentation basée non pas sur un modèle de réflexion qualitatif tels que les modèles de Shafer ou Healey (section 2.2.5) mais sur le modèle de Beckmann qui fournit une description quantitative du phénomène de réflexion. L'utilisation du modèle de Beckmann permet de mieux contrôler les approximations utilisées pour obtenir la segmentation.

De plus, nous pensons utiliser le modèle de Beckmann pour extraire des paramètres physiques de la surface. Ceci permettrait d'établir un pré-diagnostic des pièces à contrôler à partir de l'étape de segmentation. Plus généralement nous comptons étudier plus attentivement les modèles de réflexion afin de fournir une explication à un résultat très surprenant fourni par Otha [151] en 1980. Otha a en effet montré que 90% à 99% de l'information contenue dans une image couleur pouvait être approximée par une projection sur le plan défini par les deux premiers vecteurs propres de la matrice de covariance de l'ensemble des couleurs (Tables 3.5 et 3.6). Autrement dit, l'information couleur est essentiellement bi-dimensionnelle. Les résultats de Shafer, Healey et Klinker (section 2.4.1) montrent que l'ensemble des couleurs d'un matériau évolue sur un plan. Si nous considérons que l'un des vecteurs du plan contenant l'ensemble des couleurs d'un matériau correspond au lobe spéculaire qui modifie très peu le spectre de la lumière incidente nous pouvons expliquer qualitativement pourquoi les plans de tous les matériaux d'une scène partagent au moins un vecteur directeur (le vecteur intensité). Toutefois ces modèles ne permettent pas d'expliquer pourquoi tous les vecteurs correspondant au lobe diffus sont très similaires. Une explication complète de ce phénomène apporterait une meilleure compréhension de la formation et de la nature des images couleur et nous comptons nous atteler à cette tâche. L'intérêt de cette question n'est pas purement académique. En effet, comme nous le verrons au chapitre 3, la connaissance *a priori* d'information sur la répartition des couleurs d'une image permet d'optimiser de nombreux traitements du point de vue de la qualité des résultats ou du point de vue des temps de calculs. Ce type d'optimisation peut s'illustrer par les deux exemples suivants :

1. dans le cadre de la quantification, Wu [200] effectue un premier traitement sur le premier vecteur propre des couleurs de l'image (section 3.2.3.a). Ce premier traitement permet d'obtenir un gain significatif sur la qualité des images obtenues. Wu utilise dans ce cas l'information *a priori* suivante : le premier vecteur propre aura une valeur propre significativement supérieure à celle des deux autres vecteurs propres ;

2. Dans le cadre de l'inversion de table de couleur, j'ai utilisé [42] (section 3.3.6) une projection de l'ensemble des couleurs de l'image sur le plan défini par les deux premiers vecteurs propres. Cette projection permet un gain significatif en temps de calculs. J'utilise dans ce cas l'information *a priori* suivante : la perte d'information induite par la projection des couleurs sur le plan défini par les deux premiers vecteurs propres reste suffisamment faible pour pouvoir être «rattrapée» par une étape de correction (section 3.3.6).

2.7.2 Estimation de paramètres et reconstruction

L'estimation de paramètre est un terme assez général qui recouvre deux types de méthodes :

1. Dans le cadre de la reconstruction par acquisitions multiples l'estimation des paramètres comprend pour chaque point l'estimation des constantes d'un modèle de réflexion ainsi que celle des normales.
2. Dans le cadre de méthodes utilisant un plus faible nombre d'acquisitions, l'estimation de paramètres ne comprend que l'estimation des constantes du modèle de réflexion. Ces méthodes supposent généralement le matériau optiquement uniforme, ce qui signifie que les constantes du modèle de réflexion sont supposées identiques sur tous les pixels considérés.

L'estimation des constantes d'un modèle de réflexion permet d'établir un premier diagnostic de la surface si les constantes du modèle peuvent être reliées à des paramètres physiques du matériau tel que sa rugosité ou son coefficient de Fresnel. Ces constantes permettent également de reconstruire la surface ce qui permet de diagnostiquer la forme de l'objet étudié.

Notre méthode d'estimation des paramètres est basée sur un faible nombre d'illuminants. L'idée de base de cette méthode est de supposer :

1. que le matériau est optiquement homogène et
2. que le lobe spéculaire est suffisamment étroit pour pouvoir extraire des zones de l'image où le lobe spéculaire et la composante Lambertienne peuvent être alternativement négligées.

Ces suppositions nous permettent de compenser le nombre élevé d'acquisitions généralement utilisées par les méthodes basées sur des acquisitions multiples par des calculs basés sur l'ensemble des pixels des zones Lambertiennes ou spéculaires. En d'autres termes l'on compense un nombre important de données pour chaque pixel par des calculs effectués sur plusieurs pixels.

Les techniques de reconstruction représentent un domaine extrêmement riche et ancien si bien qu'une description exhaustive de toutes les méthodes ne réclamerait pas un livre mais une encyclopédie. Nous nous sommes restreints dans ce mémoire aux méthodes utilisant des modèles de réflexion possédant une base physique. Ces méthodes comprennent les méthodes de reconstructions utilisant de multiples acquisitions, celles basées sur une minimisation de l'écart entre les intensités observées et celles prédites par le modèle ainsi que celles basées sur des déplacements de l'objet.

Les méthodes basées sur des déplacements d'objets ne se basent généralement pas sur un modèle physique explicite. De plus, tous les objets ne peuvent pas être aisément translatés ou tournés. Les méthodes de minimisation requièrent généralement des temps de calcul trop élevés pour des applications temps réel. De plus, les méthodes basées sur de multiples acquisitions utilisent généralement un nombre important d'acquisitions ce qui complexifie le protocole d'acquisition et introduit un temps de latence entre la première acquisition et le traitement. La méthode que nous avons proposée tente de faire un compromis entre les avantages et inconvénients de chacune de ces méthodes. Nous utilisons le principe des acquisitions multiples mais en nous restreignant

à 2 acquisitions. Les hypothèses sur l'uniformité optique du matériau et sur l'étroitesse du lobe spéculaire nous permettent alors de déterminer les normales de la surface grâce à l'inversion de deux fonctions monotones en chaque point.

Nous pensons dans un premier temps étendre notre modèle de réflexion en prenant en compte le pic spéculaire. Comme son nom l'indique, cette composante des modèles de réflexion se traduit dans l'image par un pic d'intensité lorsque le point de la surface correspondant est éclairé dans la direction spéculaire. De tels points doivent pouvoir être détectés dans l'image en utilisant des filtres. Ces points correspondent aux cas où la normale est confondue avec la bissectrice de la direction de la source et de la direction d'observation. On a donc directement la direction de la normale sur ces points. Il nous reste à étudier si l'intensité des même points dans l'autre image nous permet de calculer la norme de la normale.

A plus long terme, nous pensons poursuivre nos activités de recherche dans la reconstruction par déplacement d'illuminants mais en étendant la classe des matériaux à des objets isolants ou faiblement conducteurs. Nous comptons également étudier plus largement l'apport éventuel des modèles physiques aux différentes méthodes de reconstruction.

2.8 Lexique des principaux symboles

- $\vec{C}$ ..... vecteur couleur
- $\vec{C}_{diff}$ ..... vecteur couleur induit par la composante diffuse
- C_{ls} ..... constante du lobe spéculaire avec observateur variable
- $\vec{C}_{mat}$ ..... couleur d'un matériau selon les modèles de Shafer et Healey
- $\vec{C}_{spec}$ ..... vecteur couleur induit par les composantes spéculaires
- $f_\alpha, f_\beta, f_{\alpha,\beta}, f_\epsilon$ ..... fonctions de densités de probabilité
- F ..... coefficient de Fresnel
- I ..... intensité d'un pixel
- Ir ..... irradiance
- Ir_{diff} ..... irradiance diffuse
- Ir_{ls} ..... irradiance induite par le lobe spéculaire
- Ir_{ps} ..... irradiance induite par le pic spéculaire
- Ir_{spec} ..... irradiance spéculaire
- K_{diff} ..... constante du lobe diffus
- K_{ls} ..... constante du lobe spéculaire avec observateur fixe
- K_{ps} ..... constante du pic spéculaire
- $\vec{k}_i$ ..... vecteur du point de réflexion vers la source lumineuse
- $\vec{k}_r$ ..... vecteur du point de réflexion vers l'observateur
- L ..... radiance
- M ..... indice complexe de réfraction
- m_{diff} ..... coefficient diffus dans les modèle de Shafer et Healy
- m_{spec} ..... coefficient spéculaire dans les modèle de Shafer et Healy
- NI ..... intensité normalisée dans les modèles Lambertiens
- $\vec{n}$ ..... normale d'une surface
- $\vec{P}$ ..... vecteur de Poynting
- R ..... réflectance
- T .... coefficient d'auto - corrélation dans le modèles de surface de Beckmann - Spizzichino
- α ..... angle entre le vecteur $\vec{\nu}$ et la normale $\vec{n}$
- Δ_{spec} . fonction valant 0 lorsque la réflexion est spéculaire. Utilisée pour l'estimation de C_{ls}
- γ_i ..... angle entre le vecteur source $\vec{k}_i$ et l'axe z du repère choisi
- ϵ ..... permittivité électrique du matériau
- θ_i ..... angle entre la source $\vec{k}_i$ et la normale $\vec{n}$
- θ_r ..... angle entre la direction d'observation $\vec{k}_r$ et la normale $\vec{n}$
- κ_{ls} ..... constante du pic spéculaire dans le modèle de Beckmann
- κ_{ps} ..... constante du pic spéculaire dans le modèle de Beckmann
- κ_{spec} ..... constante du lobe spéculaire dans le modèle de Torrance - Sparrow
- λ ..... longueur d'onde du champ électromagnétique incident ou réfléchi
- λ_{max} ..... borne supérieure de la plage de longueurs d'ondes perceptibles par l'oeil humain.
- λ_{min} ..... borne inférieure de la plage de longueurs d'ondes perceptibles par l'oeil humain
- μ ..... perméabilité magnétique du matériau
- $\vec{\nu}$ ..... bissectrice des vecteurs source $\vec{k}_i$ et d'observation $\vec{k}_r$
- ξ ..... spectre énergétique
- σ ..... conductivité du matériau

- σ_α ..... paramètre de rugosité dans le modèle de Torrance - Sparrow
- σ_h ..... paramètre de rugosité dans le modèle de Beckmann - Spizzichino
- ψ_r ....angle entre le vecteur d'observation $\vec{k}_r$ et le plan défini par la normale à la surface $\vec{n}$ et le vecteur source $\vec{k}_i$
- Ω_i variables utilisées pour délimiter la validité du modèle de Nayar lors de la reconstruction

Chapitre 3

Traitement d'images couleur

3.1 Introduction

L'étude des traitements d'images couleur s'inscrit dans la continuité de notre activité sur l'étude des modèles de réflexion. En effet, les propriétés optiques d'un matériau ainsi que la normale à sa surface vont déterminer la couleur du pixel correspondant. Cette couleur peut être codée dans différents espaces en fonction des contraintes de l'application devant manipuler l'image (section 2.3). Toutefois, l'image initiale est usuellement codée dans l'espace couleur RGB en utilisant 1 octet non signé pour chaque composante. Le nombre de couleurs affichables simultanément est donc égal à $256^3 \approx 16.10^6$. Avant la généralisation des cartes graphiques 24 bits la plupart des écrans étaient incapables d'afficher plus de 256 couleurs simultanément. Encore aujourd'hui la plupart des terminaux X sont limités à 256 couleurs. Un problème similaire se pose également pour l'impression d'images couleur où le nombre de couleurs disponibles reste limité. Il existe donc un réel besoin pour des algorithmes capables de sélectionner les K couleurs les plus représentatives d'une image, avec K usuellement proche de 256. Ces méthodes sont appelées des méthodes de *quantification*. Étant donnée une image d'entrée et le nombre K de couleurs à extraire ces algorithmes créent une table de K couleurs appelée *table de couleur* ou *palette*. L'affichage de l'image à partir de la table de couleur est effectué en affectant chaque couleur de l'image à sa couleur la plus proche dans la table. Cette étape est appelée l'étape d'*inversion de table de couleurs*. Finalement, les images obtenues par quantification présentent souvent de large zones affectées à une même couleur. L'image est donc découpée en plusieurs zones de couleur uniforme avec des changements brusques de couleur entre deux zones. Afin de gommer cet effet peu agréable à l'oeil on utilise généralement des méthodes de *dithering* qui ont pour effet de distribuer l'erreur de quantification sur plusieurs pixels ce qui permet d'atténuer l'uniformité des zones uniformes et l'intensité de leurs contours.

L'ensemble de ces traitements est représenté sur la figure 3.1. Le trajet (1) sur cette figure correspond à une *quantification uniforme* où la table de couleurs est pré-calculée. Dans ce cas l'affichage de l'image se résume aux étapes d'inversion de table de couleurs et éventuellement de dithering. Le trajet (2) inclut une étape de *quantification adaptative* dans lequel la table de couleurs est calculée en fonction de l'image à afficher.

Notre contribution à ce domaine de recherche concerne l'étape de quantification adaptative

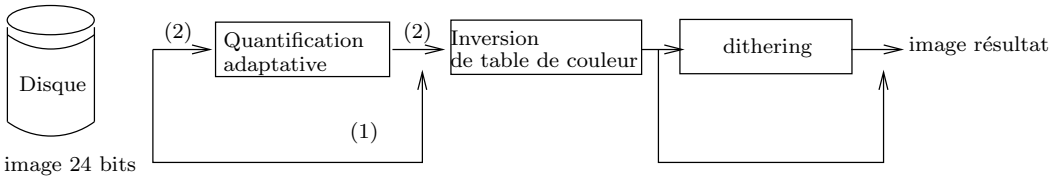


FIG. 3.1 – Affichage d'une image couleur

et celle d'inversion de table de couleur. Les sections 3.2 et 3.3 présentent un aperçu de l'histoire de ces méthodes et des différentes techniques utilisées. Nous avons intégré nos contributions à ces différentes techniques à l'intérieur des sections correspondantes. Ce style de rédaction permet de mettre clairement en lumière chaque contribution par rapport à un courant de pensée ou une problématique.

3.2 La quantification

3.2.1 Introduction

La quantification d'images couleur est un procédé visant à réduire le nombre de couleurs d'une image tout en préservant la qualité visuelle de celle-ci. Jusqu'à récemment, les algorithmes de quantification étaient couramment utilisés pour afficher des images en 24 bits sur des moniteurs ne pouvant afficher simultanément qu'un nombre réduit de couleurs. Bien que les cartes graphiques 24 bits tendent à se généraliser, la croissance d' Internet et les besoins en algorithmes de compression qui en découlent justifient la continuité des efforts de recherche en ce domaine.

L'utilisation de la quantification pour l'affichage d'images couleur a imposé à ces algorithmes deux contraintes antagonistes : d'un coté, la quantification doit être effectuée juste après le chargement de l'image et avant son affichage. L'aspect interactif de ce type de traitement étant primordial, la rapidité de calcul est une des contraintes forte imposée aux algorithmes de quantification. Parallèlement, l'image reproduite doit être la plus proche possible de l'original. L'estimation du rapport entre la qualité de l'image et le temps de calcul autorisé dépend bien sur de l'application et de nombreux algorithmes de quantification ont été conçus en fonction de différentes contraintes sur ce rapport.

Plus formellement, le processus de quantification peut se définir comme suit : étant donnée une image couleur I et son ensemble de couleurs C_I de cardinal M, la quantification de I en K couleurs (avec $K < M$ et généralement $K << M$) consiste à déterminer K couleurs représentatives $(q_1, \ldots, q_K)$ appelée table de couleur ou palette et à remplacer chaque couleur de l'image par sa plus "proche" couleur représentative. Chaque couleur de C_I étant affectée à une seule couleur représentative, l'ensemble $(q_1, \ldots, q_K)$ définit une partition de C_I en K ensembles $C_1, \ldots, C_K$ où C_i représente l'ensemble des couleurs de C_I affectées à q_i.

Les notions de couleurs représentatives et de partition de l'espace couleur sont donc intimement liées et beaucoup de méthodes de quantification définissent d'abord une partition de C_I avant d'en déduire un ensemble de couleurs représentatives. La méthode triviale consistant à énumérer toutes les partitions possibles d'un ensemble C_I de M couleurs en K sous ensembles est hors de question

dans notre cas puisque le nombre de partitions possibles est égal à [3, 201] :

$$\frac{1}{K!}\sum_{k=0}^{K}(-1)^{K-k}C_K^k k^M$$

ce qui est énorme même pour de petites valeurs de K et M. Le symbole C_K^k représente l'ensemble des combinaisons de tirage de k éléments parmi K.

Les critères utilisés en quantification ne permettant pas de déduire une partition optimale autrement que par une énumération des partitions possibles, les algorithmes de quantification doivent utiliser des heuristiques. Ces heuristiques ne permettent pas de garantir un résultat optimal mais fournissent toutefois des images de bonne qualité avec des temps de calculs restant approximativement interactifs.

Les heuristiques utilisées en quantification peuvent être classifié suivant une démarche analogue à celle communément utilisée en segmentation. On distingue en effet :

1. les méthodes descendantes (section 3.2.3), découpent l'ensemble de couleur initial jusqu'à l'obtention des K ensembles requis. Ces méthodes s'apparentent aux méthodes de segmentation basées sur une découpe récursive de l'image [127] ;
2. les méthodes ascendantes (section 3.2.5) fusionnent les couleurs initiales en K ensembles. Ces méthodes s'apparentent aux méthodes de segmentation basées sur une approche croissance de régions [1] ;
3. les méthodes mixtes (section 3.2.6) obtiennent la partition en K ensembles de couleurs à l'aide d'une série d'opérations de découpes et de fusions d'ensembles de couleurs. Ces méthodes utilisent la même approche que les méthodes de segmentation basées sur des découpes et fusions de régions [33] ;
4. les méthodes que nous avons qualifié de méthode de quantification spatiale (section 3.2.8) sont un peu à part dans cette classification dans la mesure où ces méthodes tentent d'optimiser simultanément la sélection des couleurs représentatives et leur placement dans l'image. Ces méthodes s'apparentent aux méthodes de segmentation non supervisées qui optimisent simultanément les paramètres des régions et l'appartenance de chaque pixel à une région [55].

Après avoir défini les principaux concepts utilisés en quantification (section 3.2.2), nous allons détailler les principales familles d'heuristiques utilisées par ce type de méthodes (sections 3.2.3 à 3.2.8). Nous préciserons dans chaque cas quelle a été notre contribution à ce domaine de recherche. Notons toutefois que nous ne décrirons pas ici toutes les heuristiques utilisées en quantification mais simplement celles que nous jugeons indispensables à une bonne compréhension du domaine. De même, je ne décrirais que brièvement les méthodes ascendantes (section 3.2.5) auxquelles je n'ai pas contribué. En revanche nous insisterons plus sur les méthodes de quantification spatiale qui nous semblent prometteuses. Le lecteur intéressé pourra se référer à [43] pour une description plus complète de la chaîne de traitements impliqués dans l'affichage d'images couleurs.

3.2.2 Les multi-ensembles

La notion de multi-ensemble empruntée à la logique floue [67] permet de coder dans un seul formalisme un ensemble d'éléments et la fréquence d'occurrences de chaque élément dans cet ensemble. Formellement, l'ensemble des multi-ensembles de dimension n se définit de la façon suivante :

Définition 2 *Soit $\mathcal{PB}(\mathbb{R}^n)$ l'ensemble des parties bornées de $\mathbb{R}^n$. Nous définissons $\mathcal{ME}_n$ l'ensemble des* **multi-ensembles** *de $\mathbb{R}^n$ par :*

$$\mathcal{ME}_n = \{(C, f) \in \mathcal{PB}(\mathbb{R}^n) \times \mathcal{F}(\mathbb{R}^n, \mathbb{R}_+) / f_{|\mathbb{R}^n - C} == 0\}$$

où $\mathcal{F}(\mathbb{R}^n, \mathbb{R}_+)$ représente l'ensemble des fonctions de $\mathbb{R}^n$ dans $\mathbb{R}_+$ et le symbole $==$ représente l'égalité de fonctions.

Un multi-ensemble est donc la donnée d'un couple (C, f), C décrivant un ensemble borné de $\mathbb{R}^n$ et f un fonction positive nulle hors de cet ensemble. Sauf mention contraire, tous les ensembles considérés dans ce chapitre seront des ensembles discrets. D'un point de vue informatique, les multi-ensembles peuvent être vus comme une extension de la notion d'histogramme à des espaces de dimension n. De fait, étant donnés un espace de couleur Ω et une image I nous pouvons associer à I, le multi-ensemble (C_I, f_I) où C_I est l'ensemble des couleurs de l'image I dans l'espace Ω, et f_I la fréquence d'apparition de chacune de ces couleurs. Du fait de son utilisation, la fonction f est souvent appelée la *fonction de fréquence* ou de *distribution*.

La définition d'un multi-ensemble est donc très proche de la notion de "cluster" sans être tout à fait similaire. En effet, la définition d'un cluster suppose qu'une fonction de fréquence soit définie sur tout l'espace, un cluster étant alors défini comme une partie de $\mathbb{R}^n$. Les multi-ensembles permettent de manipuler simultanément plusieurs fonctions de fréquence ce qui peut s'avérer utile dans le cadre de la segmentation où chaque région définit un multi-ensemble avec sa propre fonction de fréquence. Inversement, étant donnée une fonction de fréquence f définie sur $\mathbb{R}^n$ et un ensemble d'ensembles bornés $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_n$, les multi-ensembles associés à chacun d'eux peuvent être définis par (C_i, f_i) avec $f_i = \chi_{\boldsymbol{C}_i} f$ où $\chi_{\boldsymbol{C}_i}$ est la fonction caractéristique de $\boldsymbol{C}_i$. Dans ce cas, on omet généralement la fonction f_i et (C_i, f_i) est simplement noté $\boldsymbol{C}_i$. On retrouve donc bien dans ce cas la notion usuelle de "cluster".

Dans ce mémoire, les multi-ensembles seront principalement utilisés pour la quantification. Les algorithmes de quantification n'utilisant qu'une fonction de fréquence définie globalement sur l'espace couleur, nous allons simplifier les notations en notant un multi-ensemble $(\boldsymbol{C}, f)$ simplement $\boldsymbol{C}$. Nous conserverons toutefois le terme de multi-ensemble qui met plus en évidence la multiplicité de chaque élément que le terme "classe" usuellement utilisé pour traduire cluster. Dans un même esprit de simplification, nous nous restreignons au traitement des images couleur et sauf mention contraire, tous les multi-ensembles considérés seront de dimension 3.

De nombreux algorithmes de traitement d'images doivent caractériser les multi-ensembles à l'aide de paramètres statistiques tels que la moyenne ou la variance. Ces paramètres peuvent se déduire du calcul des *moments* définis par :

$$\begin{array}{lcl} M_0(\boldsymbol{C}) & = & \sum_{\mathbf{c} \in \boldsymbol{C}} f(\mathbf{c}) \\ M_1(\boldsymbol{C}) & = & \sum_{\mathbf{c} \in \boldsymbol{C}} f(\mathbf{c})\mathbf{c} \\ M_2(\boldsymbol{C}) & = & \sum_{\mathbf{c} \in \boldsymbol{C}} f(\mathbf{c})(\mathbf{c}_1^2, \mathbf{c}_2^2, \mathbf{c}_3^2) \\ R_2(\boldsymbol{C}) & = & \sum_{\mathbf{c} \in \boldsymbol{C}} f(\mathbf{c})\mathbf{c}.\mathbf{c}^t \end{array} \tag{3.1}$$

où $(\mathbf{c}_i^2)_{i \in \{1,2,3\}}$ représentent le carré de la $i^{\text{ème}}$ coordonnée du vecteur couleur $\mathbf{c}$ de l'ensemble $\boldsymbol{C}$ tandis que le vecteur $\mathbf{c}^t$ représente le vecteur transposé de $\mathbf{c}$.

Les quantités $M_0(\boldsymbol{C})$, $M_1(\boldsymbol{C})$ et $M_2(\boldsymbol{C})$ sont respectivement appelées les moments d'ordre $0, 1$ et 2 de $\boldsymbol{C}$. Notons que, $M_0(\boldsymbol{C})$ est un scalaire appelé le *cardinal du multi-ensemble* alors que $M_1(\boldsymbol{C})$ et $M_2(\boldsymbol{C})$ sont des vecteurs $3D$ et $R_2(\boldsymbol{C})$ une matrice 3×3 dont la diagonale est égale à $M_2(\boldsymbol{C})$.

Le cardinal $M_0(\boldsymbol{C})$ d'un multi-ensemble est souvent également noté $|\boldsymbol{C}|$. Le principal avantage des quantités M_0, M_1, M_2 et R_2 dans le cadre de la quantification est qu'elles peuvent être efficacement mises à jour lors des opérations de découpe ou de fusion de multi-ensembles. De fait, donné deux multi-ensembles C_1 et C_2, les moments de $\boldsymbol{C}_1 \cup \boldsymbol{C}_2$ sont définis par :

$$\forall i \in \{0,1,2\} \quad M_i(C_1 \cup \boldsymbol{C}_2) = M_i(\boldsymbol{C}_1) + M_i(\boldsymbol{C}_2).$$

Le même type de relation intervient pour le calcul de $R_2(\boldsymbol{C}_1 \cup \boldsymbol{C}_2)$.

Inversement, si un multi-ensemble $\boldsymbol{C}$ est découpé en deux sous multi-ensembles $\boldsymbol{C}_1$ et $\boldsymbol{C}_2$ et si les paramètres de $\boldsymbol{C}$ et $\boldsymbol{C}_1$ sont tous deux connus, les moment de $\boldsymbol{C}_2$ sont définis par :

$$\forall i \in \{0,1,2\} \quad M_i(\boldsymbol{C}_2) = M_i(\boldsymbol{C}) - M_i(\boldsymbol{C}_1).$$

Comme précédemment, $R_2(\boldsymbol{C}_2)$ est détermine par le même type de relation.

La *moyenne* d'un ensemble $\boldsymbol{C}$ se déduit de $M_1(\boldsymbol{C})$ et $|\boldsymbol{C}|$ par :

$$\mu = \frac{M_1(\boldsymbol{C})}{|\boldsymbol{C}|}.$$

De même, la *variance* du multi-ensemble le long de l'un des axes Ω_i de l'espace de couleur ainsi que sa matrice de *covariance* sont définis par :

$$\begin{array}{rcl} var_i & = & \frac{M_2(\boldsymbol{C})_i}{|\boldsymbol{C}|} - \mu_i^2 \quad \forall i \in \{1,2,3\} \\ Cov & = & \frac{R_2(\boldsymbol{C})}{|\boldsymbol{C}|} - \mu.\mu^t \end{array} \tag{3.2}$$

où μ_i désigne la composante i du vecteur moyenne de $\boldsymbol{C}$.

La matrice de covariance est utilisée dans le cadre de la quantification pour déterminer la direction de plus grande variation d'un multi-ensemble. Cette direction se définit comme le vecteur propre de plus grande valeur propre de la matrice de covariance (figure 3.2). En effet, la matrice de covariance étant symétrique, définie et positive, elle peut être diagonalisée sur une base orthogonale. Chaque valeur propre de la matrice est égale à la variance du multi-ensemble le long de l'axe défini par le vecteur propre associé. La quantité d'informations portée par chaque vecteur propre est mesurée par :

$$\frac{v_i}{\sum_{i=1}^{3} v_i} \tag{3.3}$$

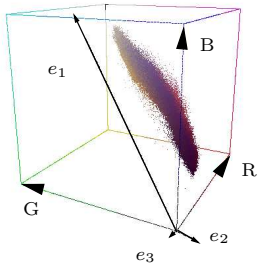
où v_i représente la valeur propre associée au vecteur propre e_i.

Étant donnée une partition en K sous multi-ensembles $\{C_1, \ldots, C_K\}$ du multi-ensemble d'une image couleur, les algorithmes de quantification associent une couleur représentative à chaque multi-ensemble. La somme des erreurs commises en affectant les couleurs appartenant à $\boldsymbol{C}_i$ à sa couleur représentative c_i est donné par la somme pondérée des carrés des distances entre chaque couleur $c \in \boldsymbol{C}_i$ et c_i :

$$\sum_{c \in \boldsymbol{C}_i} f(c)\|c - c_i\|^2.$$

Cette erreur peut être comprise comme l'erreur que l'on commet en assimilant l'ensemble des éléments de $\boldsymbol{C}_i$ à c_i. Un résultat classique en analyse des données montre que cette erreur est

a)



b)

FIG. 3.2 – Image test Lenna a) et son multi-ensemble b) avec les 3 vecteurs propres (e_1, e_2, e_3) de sa matrice de covariance. La longueur de chaque vecteur est proportionnelle à sa valeur propre.

minimale lorsque la couleur représentative c_i est égale à la moyenne du multi-ensemble $\boldsymbol{C}_i$. Le terme correspondant se nomme alors l' *erreur quadratique* :

$$\boldsymbol{SE}(\boldsymbol{C}_i) = \sum_{c \in \boldsymbol{C}_i} f(c) \|c - \mu_i\|^2 \tag{3.4}$$

où μ_i représente la moyenne de $\boldsymbol{C}_i$.

L'erreur quadratique est liée au calcul des variances par la formule suivante :

$$\boldsymbol{SE}(\boldsymbol{C}) = |\boldsymbol{C}| \sum_{i=1}^{3} var_i. \tag{3.5}$$

Chaque terme $|\boldsymbol{C}|var_i$ de l'équation 3.5 représente l'*erreur quadratique marginale* sur l'axe Ω_i de l'espace couleur Ω. On peut donc se représenter le terme $|\boldsymbol{C}|var_i$ comme la contribution de l'axe Ω_i à l'erreur quadratique totale. Si nous effectuons une rotation de l'espace couleur de façon à ce que chaque axe de coordonnée coïncide avec un des vecteurs propres de la matrice de covariance, les variances sur chaque axe nous sont données par les valeurs propres de la matrice de covariance et l'erreur quadratique s'écrit :

$$\boldsymbol{SE}(\boldsymbol{C}) = |\boldsymbol{C}| \sum_{i=1}^{3} v_i. \tag{3.6}$$

Enfin, l'erreur quadratique d'un multi-ensemble peut être efficacement calculée en utilisant les moments (équation 3.1, voir également les équations 3.2 et 3.5) :

$$\boldsymbol{SE}(\boldsymbol{C}) = \sum_{j=1}^{3} M_2(\boldsymbol{C})_j - \frac{M_1(\boldsymbol{C})_j^2}{|\boldsymbol{C}|}.$$

Étant donnée une partition $\mathcal{P} = \{C_1, \ldots, C_K\}$ de C_I en K multi-ensembles, l'*erreur de partition* $E(\mathcal{P})$ se définit à partir de la somme des erreurs quadratiques des multi-ensembles $C_1, \ldots, C_K$:

$$E(\mathcal{P}) = \sum_{i=1}^{K} \boldsymbol{SE}(C_i). \tag{3.7}$$

L'erreur de partition peut donc être considérée comme la somme des erreurs commises en affectant chaque couleur de l'image à sa couleur représentative. Ce critère est souvent utilisé par les méthodes de quantification descendantes, ascendantes et mixtes respectivement décrites dans les sections 3.2.3, 3.2.5 et 3.2.6. En revanche, les méthodes décrites en section 3.2.8 optimisent simultanément la sélection des couleurs et leur arrangement dans l'image. Ces dernières méthodes ne font donc pas référence explicitement à l'erreur de partition.

3.2.3 Les méthodes descendantes

Les méthodes descendantes ont été intensivement explorées [193, 196, 17, 194, 201, 18, 200, 4, 25] depuis l'article fondateur d'Heckbert en 1982 [97]. Ces méthodes créent une partition de l'ensemble des couleurs de l'image en K multi-ensembles par une séquence de $K - 1$ découpes.

Le multi-ensemble initial est donc découpé par un ensemble de plans appelés *plans de coupes* orthogonaux à des *axes de coupes* et passant par un point le long de l'axe de coupe appelée *position de coupe*. L'utilisation de plans pour découper un multi-ensemble peut sembler arbitraire. Une justification de ce choix est fournie par l'étape d'inversion de table de couleurs qui suit le processus de quantification (section 3.2.1). En effet, étant donné un ensemble de couleurs représentatives $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$, l'inversion de table de couleurs affecte chaque couleur de l'image à sa couleur représentative la plus proche. Ce processus induit un partitionnement de l'espace couleur par un diagramme de Voronoï 3D [160, 13] défini par $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$. Les cellules d'un diagramme de Voronoï étant définies par un ensemble de plans, le choix du plan comme surface de séparation entre deux multi-ensembles n'induit donc pas une perte de généralité.

Comme nous l'avons vu dans la section 3.2.1, une énumération exhaustive de toutes les partitions est irréaliste dans le cadre de la quantification. Les méthodes de quantification descendantes utilisent donc un ensemble d'heuristiques pour réduire le nombre de partitions envisagées. Les principales heuristiques utilisées par les algorithmes de cette famille peuvent être décomposées en 4 étapes communes à tous les algorithmes de cette famille :

1. sélection d'une stratégie de découpe ;
2. sélection du prochain multi-ensemble à découper ;
3. sélection de l'axe de découpe ;
4. sélection de la position de découpe.

Nous allons dans les sections suivantes décrire les principales heuristiques utilisées pour chacune de ces étapes. Ces heuristiques seront récapitulées dans la Table 3.1 (page 94).

3.2.3.a Les stratégies de découpe

La plupart des méthodes de quantification descendantes [193, 196, 17, 194, 201, 18, 4] découpent récursivement le multi-ensemble initial en deux sous multi-ensembles jusqu'à obtenir une partition de celui-ci en K multi-ensembles. Cette stratégie de découpe peut être codée à l'aide d'un arbre

binaire complet appelé un *arbre de découpe*. Les noeuds internes de l'arbre représentent les multi-ensembles intermédiaires tandis que les multi-ensembles finaux sont codés par les feuilles de l'arbre (figure 3.3).

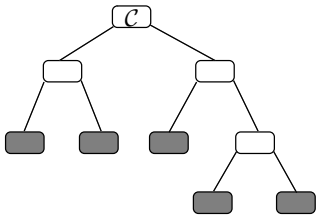


FIG. 3.3 – *Un arbre de découpe définissant une partition en* 5 *multi-ensembles.*

Le nombre de découpes récursives qui peuvent être effectuées en utilisant cette stratégie est égal au nombre d'arbre binaires complets ayant exactement K feuilles : $\frac{1}{K}C_{2(K-1)}^{K-1}$ [81]. Ce nombre étant encore trop élevé pour permettre une énumération exhaustive, Chou et al. [51] et Lin et al. [135] créent un arbre de découpe possédant N feuilles avec $K < N << \frac{1}{K}\binom{2(K-1)}{K-1}$ puis élaguent cet arbre de façon à sélectionner les K feuilles qui induisent une erreur de partition minimale. Toutefois, d'après les expérimentations menées par Wu [200], cette stratégie induit un important sur-coût de calcul sans faire baisser de façon significative l'erreur de partition. Il recommande donc de ne générer que les K multi-ensembles nécessaires.

Le principal inconvénient d'une découpe récursive du multi-ensemble initial est que la découpe de chaque multi-ensemble s'effectue sans considérer l'impact de celle-ci sur les découpes ultérieures. Cette stratégie à un coup ne prend donc pas en compte les relations entre les opérations de découpe et ne permet pas une minimisation globale de l'erreur de partition. Wu [200] a proposé d'effectuer les premières découpes du multi-ensemble initial grâce à un ensemble de κ plans orthogonaux à l'axe principal du multi-ensemble. La position relative de ces plans le long de l'axe principal est optimisée globalement grâce à un algorithme utilisant la programmation dynamique. Les $\kappa + 1$ multi-ensembles ainsi générés sont ensuite découpés récursivement en deux en utilisant la stratégie de découpe précédemment décrite jusqu'à obtenir une partition en K multi-ensembles. La valeur du paramètre κ est estimée durant la construction des $\kappa + 1$ multi-ensembles. Selon les expériences menées par Wu, cette valeur varie généralement entre 4 et 8 en fonction de la distribution du multi-ensemble initial. Notons que pour $\kappa = 2$, la méthode de Wu ce ramène à une méthode classique de découpe récursive. Le succès de cette méthode peut s'expliquer qualitativement à l'aide des modèles de réflexions (section 2.2). En effet, une image est généralement composée d'un nombre limité de matériaux différents. Dans chaque région correspondant à un matériau les variations de couleurs sont dues à un changement de la géométrie qui induit des variations de couleurs suivant l'intensité. L'ensemble des couleurs d'une image varie donc principalement suivant l'axe des intensités si bien que les premières découpes effectuées par les algorithmes de quantification descendante se font généralement suivant des directions proches de cet axe. L'heuristique de Wu permet d'optimiser globalement ces premières découpes.

3.2.3.b Choix du multi-ensemble à découper

Les méthodes basées sur une découpe récursive du multi-ensemble doivent sélectionner à chaque étape une des feuilles de l'arbre de découpe et découper le multi-ensemble associé en deux sous multi-ensembles. Les heuristiques utilisées pour choisir cette feuille varient en fonction du critère minimisé par l'algorithme de quantification.

L'algorithme de quantification par axe médian proposé par Heckbert [97] partitionne l'espace de couleur Ω en K multi-ensembles de même cardinal M_0 (équation 3.1). Pour atteindre ce but, Heckbert sélectionne à chaque étape le multi-ensemble de plus grand cardinal. Plusieurs algorithmes basés sur des $k-d$ arbres (section 3.3.3) utilisent une approche similaire à celle d'Heckbert. Cette stratégie n'est toutefois pas très adaptée à la quantification. Il n'existe en effet pas de justification claire au fait que tous les multi-ensembles doivent avoir approximativement le même cardinal. Ce critère peut en outre négliger la découpe de multi-ensembles avec un faible cardinal mais une importante variation des couleurs tout en forçant la découpe de multi-ensembles composés d'un faible nombre de couleurs proches mais possédant des fréquences importantes.

Une large majorité de méthodes [193, 196, 17, 194, 201, 18, 200, 25] tentent de minimiser l'erreur de partition. Dans ce cadre, l'objectif va donc être de découper en priorité les multi-ensembles les moins homogènes. Bouman [18] a proposé de découper à chaque étape le multi-ensemble dont la variance est maximum le long de l'axe de coupe. Bouman découpant les multi-ensembles perpendiculairement à leur axe principal, il sélectionne à chaque étape le multi-ensemble dont la première valeur propre est la plus importante.

Wan et al [193] ont proposé de découper à chaque étape le multi-ensemble d'erreur quadratique maximale. L'idée étant ici de découper le multi-ensemble dont la contribution à l'erreur de partition est maximale. Les stratégies de Wan et Bouman peuvent être comparées en écrivant l'erreur quadratique dans la base des vecteurs propres :

$$\boldsymbol{SE}(\boldsymbol{C}) = |\boldsymbol{C}| \sum_{i=1}^{3} v_i$$

où $(v_i)_{i\in\{1,2,3\}}$ représente la valeur propre associée à chacun des trois vecteurs propres.

L'heuristique de Bouman peut donc être comprise comme une approximation de celle de Wan négligeant les variations du multi-ensemble le long des deux vecteurs propres restants.

Une stratégie légèrement différente a été proposée par Wu [201]. Celui-ci propose en effet de découper à chaque étape le multi-ensemble dont la découpe induira la plus grande diminution de l'erreur de partition. Étant donné une partition du multi-ensemble initial $\mathcal{P}_k = \{\boldsymbol{C}_1, \ldots, \boldsymbol{C}_k\}$ en k multi-ensembles, la découpe d'un multi-ensemble $\boldsymbol{C}_i$ en deux sous multi-ensembles $\boldsymbol{C}_i^1$et $\boldsymbol{C}_i^2$ modifie l'erreur de partition comme suit :

$$E(\mathcal{P}_{k+1}) = E(\mathcal{P}_k) + \boldsymbol{SE}(\boldsymbol{C}_i^1) + \boldsymbol{SE}(\boldsymbol{C}_i^2) - \boldsymbol{SE}(\boldsymbol{C}_i). \tag{3.8}$$

Notons que si $\boldsymbol{C}_i^1$ et $\boldsymbol{C}_i^2$ sont tous deux non vides, la valeur de $\boldsymbol{SE}(\boldsymbol{C}_i^1) + \boldsymbol{SE}(\boldsymbol{C}_i^2) - \boldsymbol{SE}(\boldsymbol{C}_i)$ est strictement négative. L'erreur de partition est donc une fonction strictement décroissante du nombre de découpes.

Wu sélectionne donc à chaque étape le multi-ensemble $\boldsymbol{C}_i$ tel que $\boldsymbol{SE}(\boldsymbol{C}_i^1) + \boldsymbol{SE}(\boldsymbol{C}_i^2) - \boldsymbol{SE}(\boldsymbol{C}_i)$ est minimum (maximum en valeur absolue) alors que Wan sélectionne simplement le multi-ensemble tel que $\boldsymbol{SE}(\boldsymbol{C}_i)$ est maximum. L'heuristique de Wan peut donc à son tour être considérée comme une approximation de celle de Wu négligeant l'erreur quadratique des deux sous multi-ensembles

générés par l'opération de découpe. Toutefois, l'heuristique de Wu impose une découpe préventive de chaque multi-ensemble afin d'estimer la diminution de l'erreur de partition induite par cette découpe. D'après les expériences menées par Wu [200], la différence entre les deux heuristiques en terme d'erreur de partition finale n'est pas significative. De plus, l'heuristique de Wu impose une découpe inutile des feuilles de l'arbre de découpe final. La sélection à chaque étape du multi-ensemble de plus grande erreur quadratique semble donc être un bon compromis entre la qualité de l'image résultat et les temps de calcul.

3.2.3.c Détermination de l'axe de coupe

Étant donné un multi-ensemble $\boldsymbol{C}_i$ à découper, il convient de déterminer la normale du plan de coupe et sa position de long de l'axe de découpe. Puisque la décroissance de l'erreur de partition après la découpe de $\boldsymbol{C}_i$ en $\boldsymbol{C}_i^1$ et $\boldsymbol{C}_i^2$ est égale à $\boldsymbol{SE}(\boldsymbol{C}_i^1) + \boldsymbol{SE}(\boldsymbol{C}_i^2) - \boldsymbol{SE}(\boldsymbol{C}_i)$, le plan de coupe optimal est celui qui minimise $\boldsymbol{SE}(\boldsymbol{C}_i^1) + \boldsymbol{SE}(\boldsymbol{C}_i^2)$. Toutefois, les possibilités pour placer un tel plan sont encore une fois trop importantes pour envisager une énumération exhaustive de tous les plans de coupe possibles et nous devons préalablement fixer la normale de ce plan avant d'envisager une recherche exhaustive. Puisque la découpe d'un multi-ensemble décroît essentiellement la variance le long de l'axe de découpe, une heuristique couramment utilisée consiste à découper le multi-ensemble suivant un axe de grande variance.

Heckbert [97] approxime la variance en enfermant chaque multi-ensemble dans une boite rectangulaire dont les cotés sont parallèles aux axes des coordonnées. Étant donné un multi-ensemble $\boldsymbol{C}_i$, les deux faces de sa boite englobante orthogonales à un axe Ω_j seront définies par les plus petites et plus grandes projections des couleurs de $\boldsymbol{C}_i$ sur l'axe Ω_j. La variation d'un multi-ensemble suivant Ω_j est alors déterminé par la taille de sa boite englobante le long de cet axe et la boite est découpée perpendiculairement à son coté de plus grande taille.

La sélection du plus long coté de la boite englobante permet donc à Heckbert d'approximer la variance du multi-ensemble sur chacun des axes de coordonnées. Toutefois, l'axe de coordonnée de plus grande variance n'est généralement pas celui suivant lequel les données varient le plus fortement. Cette direction est définie par l'axe principal du multi-ensemble et la relation entre la variance suivant cette direction et l'erreur quadratique est fournie par l'équation 3.3. Notons que si λ_1 représente la variance le long de l'axe principal et $(var_i)_{i\in\{1,2,3\}}$ les variances suivant chacun des axes de coordonnés Ω_i, nous avons : $v_1 \geq var_i \, \forall i \in \{1,2,3\}$. Une plus grande décroissance de l'erreur de partition peut donc être attendue en coupant le multi-ensemble perpendiculairement à son axe principal. Cette dernière heuristique est utilisée par par Wan [193, 194], Wu [201, 200] et Bouman [17, 18].

3.2.3.d Position de découpe

Étant donné un multi-ensemble $\boldsymbol{C}$ et un axe de coupe défini par un vecteur A, considérons les valeurs m et M définissant respectivement les projections minimales et maximales des couleurs de $\boldsymbol{C}$ sur A. Puisque le plan de coupe est orthogonal à A, sa projection sur l'axe de découpe est définie par une valeur $t \in [m, M]$. Comme nous pouvons le constater sur la figure 3.4, toute valeur de t dans l'intervalle $]m, M[$ définit une découpe de $\boldsymbol{C}$ en deux sous multi-ensembles $\boldsymbol{C}_t^1$ et $\boldsymbol{C}_t^2$. Notons que, si par convention le plan de coupe appartient à $\boldsymbol{C}_t^2$, nous avons $\boldsymbol{C}_m^1 = 0$ et $|\boldsymbol{C}_m^2| = |\boldsymbol{C}|$ tandis que $|\boldsymbol{C}_{M+\epsilon}^1| = |\boldsymbol{C}|$ et $|\boldsymbol{C}_{M+\epsilon}^2| = 0$ où ϵ représente un nombre quelconque strictement positif. Plus

précisément, la fonction δ définie par :

$$\delta(t) = \frac{|\boldsymbol{C}_t^1|}{|\boldsymbol{C}|}$$

est une fonction croissante de t définie sur $[m, M]$ et à valeurs dans $[0, 1[$.

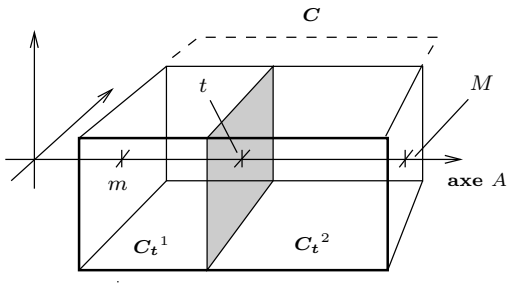


FIG. 3.4 – *Découpe de $\boldsymbol{C}$ suivant l'axe A*

La valeur de t telle que $\delta(t) = \frac{1}{2}$ correspond à une coupe médiane découpant $\boldsymbol{C}$ en deux sous multi-ensembles de même cardinal. L'algorithme de quantification par coupe médiane d'Heckbert [97] déplace donc le plan de coupe de long de l'axe A depuis la position $t = m$ jusqu'à $\delta(t) = \frac{1}{2}$.

Bouman [17, 18] utilise comme valeur de t la projection de la moyenne du multi-ensemble : $t = \mu.A$ où μ représente la moyenne de $\boldsymbol{C}$. Cette heuristique permet d'éviter un parcours du plan de coupe ; toutefois, la validité de ce choix n'est démontrée que pour des multi-ensembles de cardinal important avec une distribution gaussienne. Cette dernière restriction est importante dans la mesure où l'algorithme de découpe privilégie la découpe de multi-ensembles au moins bi-modaux.

Wan et al [193] maximisent la décroissance de l'erreur quadratique marginale le long de l'axe de découpe (section 3.2.2 et équation 3.5). La valeur t_{opt} maximisant la décroissance de l'erreur quadratique marginale le long de l'axe de découpe est définie par [193] :

$$t_{opt} = \arg \max_{t \in [m,M]} [var - (\delta(t) var_1(t) + (1 - \delta(t)) var_2(t))]$$

où $var_1(t)$ et $var_2(t)$ représentent respectivement la variance de $\boldsymbol{C}_t^1$ et $\boldsymbol{C}_t^2$ le long de l'axe principal de $\boldsymbol{C}$.

La formule ci-dessus utilise simultanément des paramètres des multi-ensembles $\boldsymbol{C}_t^1$ et $\boldsymbol{C}_t^2$ ce qui nous oblige à mettre simultanément à jour les paramètres de $\boldsymbol{C}_t^1$ et $\boldsymbol{C}_t^2$ lorsque l'on déplace le plan de coupe depuis $t = m$ jusqu'à $t = M$. Wan et al. [193, 196] ont prouvé que la position du plan de coupe est également donnée par :

$$t_{opt} = \arg \max_{t \in [m,M]} \left[\frac{\delta(t)}{1 - \delta(t)} \left((\mu - \mu_1(t)) \bullet A \right)^2 \right] \tag{3.9}$$

où μ et $\mu_1(t)$ représentent respectivement la moyenne de $\boldsymbol{C}$ et $\boldsymbol{C}_t^1$.

Le principal avantage de cette dernière formule est que t_{opt} est à présent uniquement fonction des paramètres de $\boldsymbol{C}$ et $\boldsymbol{C}_t^1$. Les paramètres d'un seul des deux multi-ensembles doivent donc

être mis à jour lors du déplacement du plan de coupe. De plus, cette dernière formule, combinée avec des résultats établis par Wong [196], permet de restreindre le domaine de recherche $[m, M]$ à :$[\frac{m+\mu\bullet A}{2}, \frac{M+\mu\bullet A}{2}]$. Notons que ce dernier intervalle contient la position de coupe $\mu\bullet A$ sélectionnée par Bouman.

Toutefois, Wan et al maximisent simplement la décroissance de l'erreur quadratique marginale alors que selon l'équation 3.6, l'erreur quadratique est égale à la somme des erreurs quadratiques marginales calculées sur chacun des trois vecteurs propres. Wu [201, 200] maximise la décroissance de l'erreur de partition (équation 3.8) définie par : $\boldsymbol{SE}(\boldsymbol{C_t}^1) + \boldsymbol{SE}(\boldsymbol{C_t}^2) - \boldsymbol{SE}(\boldsymbol{C})$. Wu a prouvé que cette dernière équation est minimale pour une valeur t_{opt} telle que :

$$t_{opt} = \arg \max_{t\in[m,M]} \frac{\|M_1(\boldsymbol{C_t}^1)\|^2}{|\boldsymbol{C_t}^1|} + \frac{\|M_1(\boldsymbol{C}) - M_1(\boldsymbol{C_t}^1)\|^2}{|\boldsymbol{C}| - |\boldsymbol{C_t}^1|}. \tag{3.10}$$

Cette dernière équation utilise uniquement le cardinal et le moments d'ordre 1 de $\boldsymbol{C}_t^1$ qui peuvent être efficacement mis à jour lors du déplacement du plan de coupe le long de la direction de coupe. Toutefois, l'utilisation du carré de la norme du moment d'ordre 1 de $\boldsymbol{C}_t^1$ ($\|M_1(\boldsymbol{C_t}^1)\|^2$) impose de manipuler de grande quantités ce qui conduit à des erreurs d'arrondis qui peuvent être importantes.

3.2.4 Ma contribution aux méthodes descendantes

Ma contribution aux méthodes descendantes [30, 25] a été effectuée en collaboration avec Jean-Pierre Braquelaire durant ma thèse de doctorat. Les différentes heuristiques utilisées par cette méthode peuvent être décrites à l'aide de la décomposition des méthodes descendantes décrite dans la section 3.2.3.

3.2.4.a Stratégie de découpe et choix du multi-ensemble

Notre méthode est basée sur une découpe récursive du multi-ensemble initial. À chaque itération de notre algorithme, nous découpons le multi-ensemble de plus grande erreur quadratique. En effet, comme nous l'avons vu dans la section 3.2.3.b, cette heuristique est celle qui permet un meilleur équilibre entre la qualité de l'image résultat et les temps de calculs.

3.2.4.b Axe de coupe

Notre choix de l'axe découpant un multi-ensemble sélectionné est basé sur un compromis entre :
- la méthode d'Heckbert [97] qui estime la variation des données d'un multi-ensemble sur chaque axe par la taille de sa boite englobante (section 3.2.3.c) ;
- les méthodes de Wan et al [193, 194], Bouman [17, 18] et Wu [201] qui découpent le multi-ensemble perpendiculairement à son axe principal.

L'utilisation de l'axe principal d'un multi-ensemble nécessite de stocker dans chaque multi-ensemble sa matrice de covariance. Il est donc nécessaire de calculer et stocker les covariances entre les différentes coordonnées. Notons que ces covariances ne sont utilisées que pour la détermination de l'axe de coupe puisque la détermination de la position de coupe ne nécessite que des paramètres ne dépendant que d'une coordonnée (équation 3.10). De plus, l'utilisation de l'axe principal implique une diagonalisation de la matrice de covariance préalablement à chaque découpe.

L'utilisation de l'axe principal augmente donc la complexité et les temps de calculs d'un algorithme de quantification. D'un autre coté, la méthode d'Heckbert ne fournit qu'une estimation assez pauvre de la variation de données suivant chaque axe.

Nous avons donc décidé de sélectionner comme axe de coupe, l'axe de coordonnée de plus grande variance. Cette heuristique *a priori* moins efficace que la méthode reposant sur l'axe principal offre toutefois plusieurs avantages. Tout d'abord, cette méthode nous permet de ne pas utiliser la matrice de covariance et donc de s'affranchir des calculs de covariance et de la diagonalisation de cette matrice. De plus, le multi-ensemble étant découpé perpendiculairement à des directions fixes, le stockage du multi-ensemble initial peut être optimisé de façon à accélérer ses découpes.

3.2.4.c Position de découpe

Notre contribution à la détermination de la position de coupe a porté sur une étude de l'évolution de l'erreur de partition en fonction de l'abscisse t du plan de coupe sur l'axe de coupe (section 3.2.3.d). Cette étude nous a permis de simplifier l'équation 3.10 déterminant la position t_{opt} du plan de coupe :

$$t_{opt} = \arg \max_{t \in [m,M]} g(t) \text{ avec } g(t) = \frac{\delta(t)}{1-\delta(t)} \left(\mu - \mu_1(t)\right)^2 . \tag{3.11}$$

Cette dernière équation peut être considérée comme l'extension en 3 dimensions de la formule de Wan (équation 3.9). Elle permet de manipuler des quantités plus petites que l'équation de Wu (équation 3.10) et donc d'éviter les erreurs d'arrondis inhérentes à cette dernière.

Une étude plus poussée [30] des variations de la fonction $g(t)$ nous a permis d'encadrer les variations de la fonction $g(t)$ entre une parabole U et une hyperbole L (figure 3.2.4.c). La parabole U permet d'arrêter la recherche de t_{opt} dès que $\delta(t) \geq \delta_{max}$, comme indiqué sur la figure 3.2.4.c. Les expériences que nous avons menées nous ont permis de conclure que cette dernière propriété permet d'ignorer en moyenne 10% de l'intervalle $[m, M]$. Notons de plus que puisque les fonctions U et L prennent leur maximum quand δ est proche de $\frac{1}{2}$, le maximum de $g(t)$ doit être obtenu pour des valeurs de $\delta(t)$ proche de $\frac{1}{2}$. Cette valeur $\delta = \frac{1}{2}$ correspond à la coupe médiane sélectionnée par Heckbert.

3.2.5 Les méthodes ascendantes

La famille des méthodes ascendantes regroupe l'ensemble des méthodes de quantification qui génèrent les K couleurs représentatives en un seul parcours de l'image. Les méthodes de cette famille génèrent ou mettent à jour chaque couleur représentative en fonction de la couleur du pixel courant et de l'ensemble des couleurs déjà traversé. Les deux méthodes les plus représentatives de cette famille sont sans doute la méthode de quantification par Octree [85] et l'algorithme du max-min [107, 203].

3.2.5.a La quantification par Octree

La méthode proposée par Gervautz [85] est basée sur une décomposition hiérarchique de l'espace RGB par un octree. En effet chaque triplet dans l'espace RGB peut être représenté par un cube de coté 1. Si nous découpons récursivement chaque axe du cube RGB en deux intervalles, l'ensemble

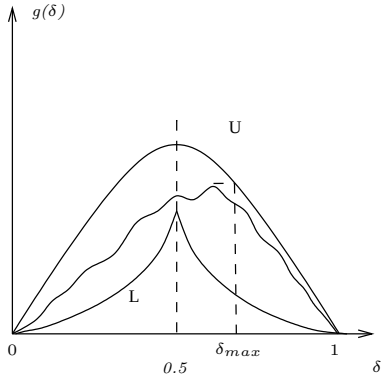


FIG. 3.5 – Évolution de l'erreur de partition induite par la découpe d'un multi-ensemble par un plan représentée comme une fonction du cardinal d'un des deux sous multi-ensembles.

		1	2	3	4	5
Choix du multi-ensemble	Plus grande erreur quadratique		⋆		⋆	⋆
	Plus grande valeur propre			⋆		
	Plus grand cardinal	⋆				
Axe de découpe	Axe de coordonées le plus long	⋆				
	Axe de coordonnées de plus grande variance					⋆
	Axe principal		⋆	⋆	⋆	
Position de découpe	Coupe médiane	⋆				
	Erreur quadratique marginale		⋆			
	Minimisation de l'erreur de partition				⋆	⋆
	Passage par la moyenne			⋆		

(1)	:	Heckbert [97]
(2)	:	Wan et Wong[194]
(3)	:	Bouman et Orchard[18, 17]
(4)	:	Wu [201]
(5)	:	Braquelaire et Brun [25]

TAB. 3.1 – Les différentes stratégies utilisées par 5 méthodes de quantification descendantes

des décompositions ainsi produit peut être représenté à l'aide d'un octree dont les feuilles codent les cubes de coté 1, i.e. les triplets de l'espace RGB.

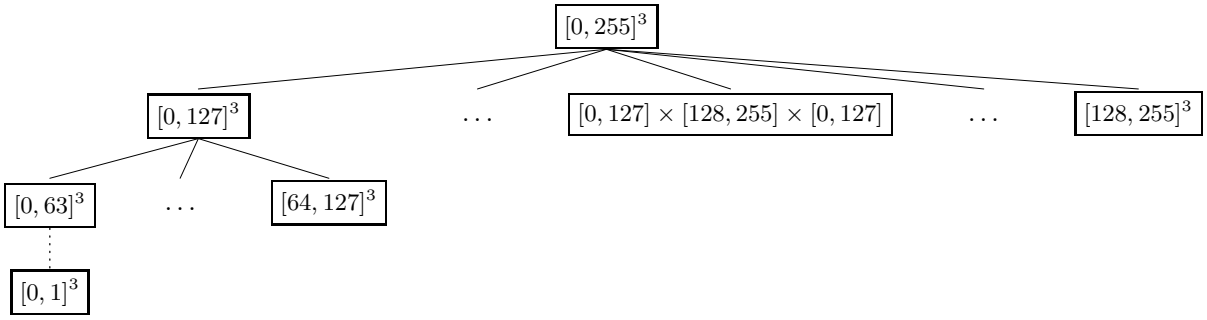


FIG. 3.6 – Codage du cube RGB par un octree

L'idée de base de la méthode de Gervautz est de gérer la structure de l'octree durant le parcours de l'image de façon à ne conserver que K feuilles à chaque étape de l'algorithme. Chacune de ces feuilles définit alors une couleur représentative à la sortie de l'algorithme. Plutôt que de générer préalablement la totalité de l'arbre, Gervautz parcourt donc l'image et crée une branche dans l'arbre pour chaque couleur rencontrée. Étant donné un triplet (R, G, B) à rajouter à l'octree, deux cas peuvent se présenter : si la branche correspondant au triplet existe déjà sa couleur moyenne est mise à jour. La structure de l'arbre reste alors inchangée. Si la branche associée à cette couleur n'existe pas, elle est ajoutée à l'octree. Si cette mise à jour génère plus de K feuilles, deux feuilles de l'arbre sont fusionnées en transformant leur plus proche parent commun en une nouvelle feuille.

3.2.5.b L'algorithme max-min

Les méthodes basées sur une minimisation de l'erreur de partition tentent de minimiser la somme des distances de chaque couleur à sa couleur représentative. L'approche des algorithmes max-min est légèrement différente. Plutôt que de créer des multi-ensembles aussi homogènes que possible, ces méthodes maximisent la distance entre les couleurs représentatives associées à chaque multi-ensemble.

Un algorithme basé sur ce principe a été proposé par Houle et Dubois [107] en 1986 et Xiang [203] en 1997. Il semblerait à la lecture des articles que Xiang ait ignoré l'article de Houle et Dubois. Les deux méthodes itèrent les points suivants :

- sélectionner arbitrairement une couleur représentative c_1 et initialiser son multi ensemble $\boldsymbol{C}_1$ à partir de l'ensemble des couleurs de l'image ;
- pour i=1 à K
  1. pour chaque multi-ensemble $(\boldsymbol{C}_j)_{j\in\{1,\ldots,i-1\}}$, calculer la distance :

$$d_j = \max_{c\in\boldsymbol{C}_j} \|c - c_j\|^2,$$

  2. sélectionner une couleur c_i ayant une distance à sa couleur représentative égale à $max_{j\in\{1,\ldots,i-1\}} d_j$,

3. créer un nouveau multi-ensemble $\boldsymbol{C}_i$ en y affectant toutes les couleurs plus proches de c_i que leur actuelle couleur représentative.

Cet algorithme tente donc de maximiser la distance entre les couleurs représentatives et donc l'enveloppe convexe de l'ensemble des couleurs représentatives dans l'espace de couleur. Ce dernier critère est un avantage pour les méthodes de dithering qui suivent usuellement l'étape de quantification (section 3.2.1). Toutefois, la qualité des images produites par cette méthode sans dithering est généralement moins bonne que celle des méthodes basées sur une minimisation de l'erreur de partition.

3.2.6 Les méthodes mixtes

Les méthodes de quantification descendantes sont rapidement apparues comme très prometteuses et un intense effort de recherche a été effectué dans ce domaine. Ces recherches, principalement menées par Wu [201, 199], Wan [193, 194], Bouman [18] et Balasubramanian [4], ont abouti en 1992 à une méthode conçue par Wu [200] permettant d'obtenir des images de très bonne qualité grâce à une optimisation des premières étapes de découpe. Cet article a conclu une intense activité de recherches sur les heuristiques de découpe menées depuis l'article fondateur d'Heckbert [97]. En effet, depuis lors la recherche sur les méthodes de quantification descendantes s'oriente plus sur la définition de nouveaux critères de découpe [136] que sur l'heuristique de découpe elle même. Il apparaît donc désormais difficile d'apporter une amélioration significative à ce type de méthodes. De plus, malgré la qualité des images généralement produites, la découpe récursive du multi-ensemble initial $\boldsymbol{C}_I$ impose de parcourir chaque couleur en moyenne $\log_2(K)$ fois. Ceci induit une complexité en $\mathcal{O}(8.M)$ pour une quantification en 256 couleurs (M représente le nombre de couleurs de $\boldsymbol{C}_I$). Cette complexité induit des temps de calculs souvent trop élevés pour une utilisation interactive.

Inversement, les méthode ascendantes construisent la table de couleurs en un seul parcours de l'image. Ces méthodes sont donc généralement plus rapides que les méthodes descendantes. En revanche, lors du parcours de chaque couleur de l'image, un algorithme de quantification ascendante doit soit créer une nouvelle couleur représentative soit affecter la couleur d'entrée à une des couleurs représentatives déjà sélectionnée. Cette heuristique appliquée à chaque pixel doit être extrêmement simple pour rester compatible avec des temps de calculs interactifs. La qualité des images produites par ces algorithmes est donc généralement moins bonne que celle produite par les méthodes descendantes.

Les méthodes mixtes peuvent être comprises comme des méthodes intermédiaires entre les méthodes descendantes (découpe pure) et les méthodes ascendantes (fusion pure). Étant donné un multi-ensemble initial $\boldsymbol{C}_I$ à partitionner en K sous multi-ensembles ces méthodes effectuent une première étape de découpe de façon à obtenir une partition en $N > K$ multi-ensembles puis effectuent une série de fusions de façon à se ramener au nombre requis K de multi-ensembles formant une partition de $\boldsymbol{C}_I$. Le but de la première étape de découpe est de réduire le nombre de données tout en préservant les principales propriétés de la distribution $\boldsymbol{C}_I$. Partant de cet ensemble réduit de données, l'algorithme de fusion peut appliquer des heuristiques plus coûteuses en temps de calcul (et donc *a priori* plus complexes) que celles généralement utilisées par les méthodes ascendantes.

Les N multi-ensembles résultants de l'étape de découpe peuvent être codés à l'aide d'un graphe où chaque noeud représente un multi-ensemble et chaque arête une fusion possible. La fusion de deux multi-ensembles est alors réalisée par la contraction de l'arête codant cette possibilité de

fusion et la suppression de toute arête multiple entre le noeud issue de la fusion et les noeuds adjacents.

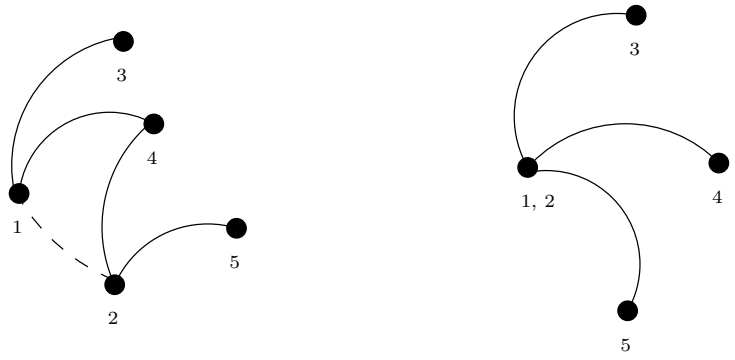


FIG. 3.7 – *Fusion de deux noeuds*

Intuitivement, l'affichage de l'image avec $N > K$ couleurs doit produire une image de meilleure qualité qu'un affichage avec simplement K couleurs. Le processus de fusion va donc progressivement détériorer la qualité de l'image et tout critère évaluant cette qualité doit rendre compte de ce phénomène. Afin de minimiser cette perte de qualité, il convient d'attribuer une valeur à chaque arête en fonction de la dégradation induite par la fusion des deux sous multi-ensembles adjacents par cette arête. Si nous utilisons l'erreur de partition (équations 3.4 et 3.7), chaque arête est étiquetée par l'augmentation de l'erreur de partition induite par la fusion. Étant donnée une partition $\mathcal{P}'$ déduite de $\mathcal{P}$ par la fusion de deux multi-ensembles C_k et $C_{k'}$ appartenant à $\mathcal{P}$, cette augmentation est définie par [71] :

$$E(\mathcal{P}') = E(\mathcal{P}) + \frac{|\boldsymbol{C}_k| \,.\, |\boldsymbol{C}_{k'}|}{|\boldsymbol{C}_k| + |\boldsymbol{C}_{k'}|} \parallel \mu_k - \mu_{k'} \parallel^2 . \tag{3.12}$$

La valeur de l'arête associé a deux multi-ensembles C_k et $C_{k'}$ est donc égale à :

$$\Delta_{k,k'} = \frac{|\boldsymbol{C}_k| \,.\, |\boldsymbol{C}_{k'}|}{|\boldsymbol{C}_k| + |\boldsymbol{C}_{k'}|} \parallel \mu_k - \mu_{k'} \parallel^2 . \tag{3.13}$$

Le coût principal de ce type de méthode vient de l'algorithme de fusion qui doit considérer toutes les arêtes du graphe à chaque itération. Deux type d'heuristiques ont donc été proposées pour réduire les temps de calculs : la diminution du nombre N de multi-ensembles résultants de l'opération de découpe ou l'utilisation d'heuristiques permettant de négliger certaines arêtes lors des opérations de fusion.

L'algorithme d'Equitz [71] est basé sur un rejet *a priori* de certaines possibilités de fusion. Equitz réduit en effet le coût de l'algorithme de fusion en utilisant un arbre binaire qui décompose le multi-ensemble initial $\boldsymbol{C}_I$ en N sous multi-ensembles chacun composé de 8 couleurs. Le graphe complet représentant l'ensemble des fusions est donc décomposé en un ensemble de sous-graphes complets chacun composé de 8 noeuds. Chaque itération de l'algorithme d'Equitz consiste à déterminer dans chaque sous-graphe les deux noeuds les plus proches en utilisant l'équation 3.12. La liste de ces candidats à la fusion est alors ordonnée en fonction de la valeur de leur arête commune et une fraction d'entre eux (telle que 50%) est effectivement fusionnée. Après chaque itération,

le partitionnement du graphe initial doit être mis à jour de façon à ce que chaque sous-graphe contienne un nombre approximativement égal de noeuds. La restriction des opérations de fusion à une fraction des sous-graphes permet de minimiser les risque de fusion entre multi-ensembles de distributions très différentes. Notons toutefois que ce risque n'est pas complètement écarté. De plus, l'heuristique d'Equitz néglige les fusions possibles entre des noeuds n'appartenant pas au même sous-graphe. Cet algorithme a été ensuite amélioré par Balasubramanian et Allebach [5] qui ont diminué le nombre de couleurs du multi-ensemble initial grâce à une étape de pré-quantification basée sur l'arrangement local des couleurs dans l'image originale. Cet arrangement est également utilisé par Balasubramanian et Allebach pour pondérer la distance entre deux multi-ensembles définie par l'équation 3.12.

L'algorithme de Dixit [65] est basé quand à lui sur une réduction du nombre de données initiales. Dixit définit l'ensemble initial de couleurs en effectuant des tirages aléatoires dans l'image. L'ensemble des couleurs de chaque pixel tirés aléatoirement défini alors le multi-ensemble initial. D'après les expériences menées par Dixit, 1024 tirages sont suffisants pour avoir une bonne estimation de la distribution des couleurs d'une image 512×512. Dixit utilise l'ensemble des couleurs tirées aléatoirement pour définir un multi-ensemble par couleur et trie cet ensemble selon le cardinal des multi-ensembles. L'algorithme de fusion parcourt ensuite l'ensemble trié des multi-ensembles suivant un ordre croissant du cardinal et fusionne chaque multi-ensemble parcouru avec le multi-ensemble le plus proche en utilisant l'équation 3.12. Le multi-ensemble issu de la fusion est alors exclu des opérations de fusion pour cette itération. Chaque multi-ensemble participe donc à une seule opération de fusion à chaque itération et le nombre de multi-ensembles est divisé par 2 à chaque itération. Notons que ce facteur de décimation fixe peut conduire à la fusion de multi-ensembles très éloignés.

3.2.7 Ma contribution aux méthodes mixtes

Ma contribution aux méthode mixtes a été effectuée en collaboration avec Myriam Mokhtari [41]. Notre méthode de fusion est basée sur une étape de quantification uniforme du cube RGB fournissant un ensemble N de multi-ensembles. Ces multi-ensembles sont alors utilisés pour définir un graphe complet connectant chacun des N noeuds à tous les autres. L'algorithme de fusion sélectionne à chaque étape l'arête de plus faible coût en utilisant l'équation 3.12. La fusion de deux noeuds implique une mise à jour de la structure du graphe en utilisant l'opération de contraction (figure 3.7) et la mise à jour de la valeur des arêtes incidentes au nouveau noeud. Chaque contraction entraîne donc la disparition d'un noeud et l'algorithme itère $N - K$ contractions de façon à obtenir le nombre requis K de multi-ensembles finaux.

Notre méthode utilisant un graphe complet, sa complexité sera essentiellement déterminée par le nombre N de multi-ensembles initiaux. Nous avons donc étudié l'influence du paramètre N sur l'erreur de partition finale. Une partie de ces expériences est représentée sur la figure 3.8 qui illustre l'évolution de l'erreur de partition en fonction du nombre initial de multi-ensembles pour 4 images tests et une quantification en 16 couleurs. La figure 3.8 montre que l'erreur de partition après une décroissance importante reste approximativement constante pour un nombre de multi-ensembles initiaux variant entre 100 et 900. Cette stabilité des courbes au delà de 100 valide l'hypothèse d'une taille optimale des multi-ensembles initiaux pour un nombre de multi-ensembles finaux donné. En effet, au delà d'une certaine limite, la découpe d'un multi-ensemble initial n'est pas utilisé par l'étape de fusion qui regroupe majoritairement ces sous multi-ensembles dans un même multi-ensemble final.

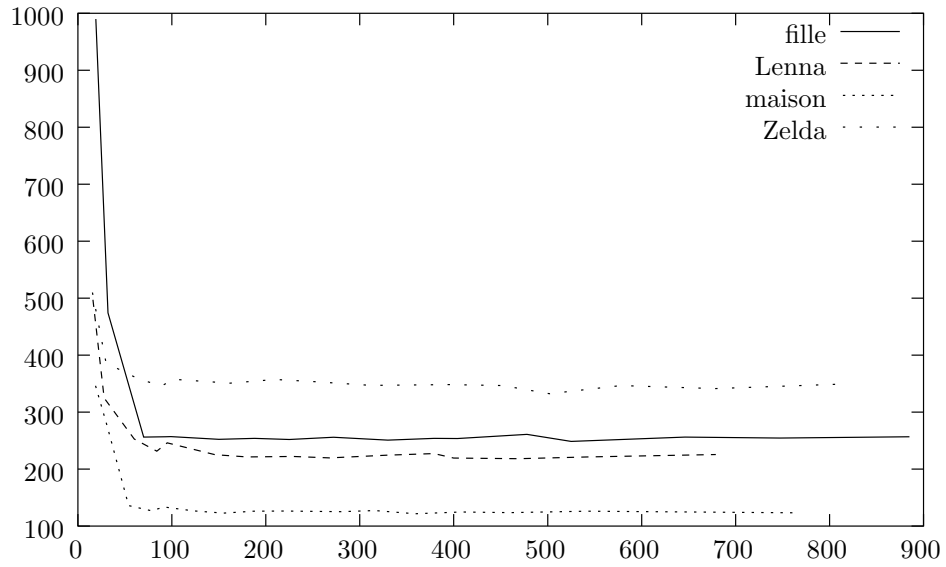


FIG. 3.8 – *Évolution de l'erreur de partition en fonction du nombre de multi-ensembles initiaux*

Notons toutefois que cette dernière propriété n'est vraie que pour un nombre de multi-ensembles finaux donnés. La figure 3.9 montre l'évolution du rapport entre les erreurs de partitions produites par notre méthode descendante [20] (section 3.2.4) et notre méthode mixte pour 4 images tests et un nombre de couleurs variant entre 2 et 256. Notre méthode descendante est basée sur une découpe récursive et reste donc insensible au nombre de multi-ensemble initiaux. Cette méthode nous sert ici d'étalon afin de définir une erreur de partition de référence pour un nombre de couleurs et une image donnés. Le nombre de multi-ensembles initiaux pour la méthode mixte a été fixé à 330, 280, 329 et 396 (ces valeurs sont issues d'une partition du cube RGB en 2048 sous cubes) respectivement pour les images fille, maison, Lenna et Zelda (Fig. 3.11). Nous pouvons constater que l'algorithme mixte produit de plus faibles erreurs de partitions que la méthode descendante pour un nombre de couleurs approximativement inférieur à 100. Au delà de cette limite, le nombre de multi-ensembles initiaux fournis à l'algorithme de fusion est trop faible pour produire des images de qualité. De fait, l'utilisation de 280 multi-ensembles initiaux pour une quantification en 256 couleurs de l'image maison produira un résultat proche de celui obtenu par une partition uniforme.

Notons de plus, que la complexité de l'algorithme de fusion augmente quadratiquement en fonction du nombre de multi-ensembles initiaux. Notre méthode est donc plus spécifiquement conçue pour des quantifications en un nombre peu élevé de couleurs.

Les erreurs de partitions représentées sur la Table 3.2 montrent que notre méthode produit des images de plus faible erreur de partition que celle de Wu [201] pour une quantification en 16 couleurs. De plus, les temps de calculs représentés sur la Table 3.3 montrent que notre méthode est approximativement 10 fois plus rapide que les méthodes de Wu [201] et de Xiang [203]. Ces temps de calculs ont été obtenus sur un Athlon XP1800 $1533Mz$. Les erreurs de partitions de l'algorihme de Xiang ne sont pas représentées sur la Table 3.2 puisque cette méthode ne minimise pas ce critère.

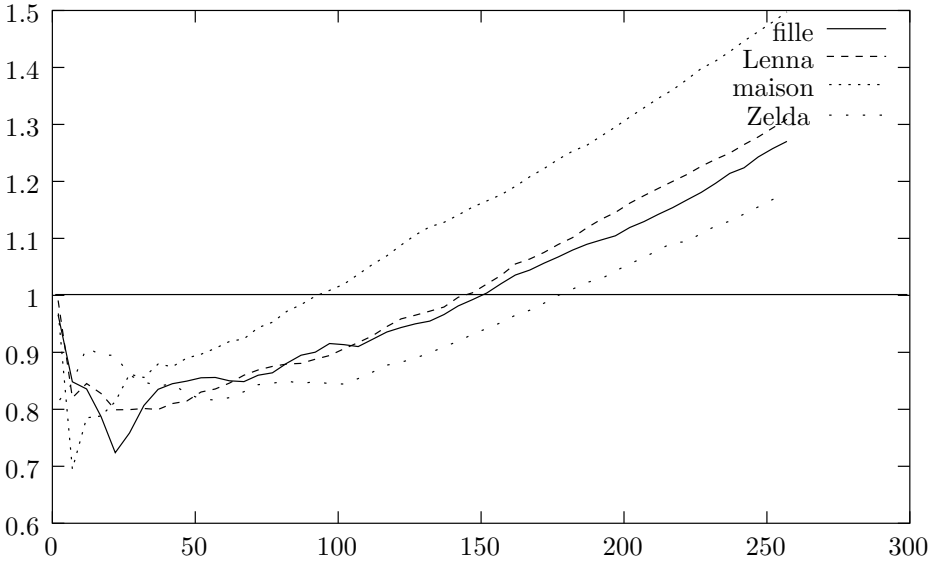


FIG. 3.9 – *Rapport entre les erreurs de partitions produites par notre méthode mixte utilisant un nombre de multi-ensembles initiaux constant et notre méthode de découpe (section 3.2.3 et Table 3.1) en fonction du nombre de couleurs finales*

SE	anemone	fille	fleurs	grenouille	Lenna
notre méthode	538	260	672	435	218
méthode de Wu	580	288	770	532	241

SE	maison	meloid	papillon	poisson	Zelda
notre méthode	124	463	288	368	346
méthode de Wu	133	532	311	418	367

TAB. 3.2 – Erreur de partition calculée dans l'espace RGB produites par notre méthode et celle de Wu lors de la quantification de 4 images tests en 16 couleurs. Notre méthode utilise un partitionnement initial du cube RGB en 4096 sous cubes.

temps en secondes	anemone	fille	fleurs	grenouille	Lenna
notre méthode	0.75	0.08	0.37	0.24	0.17
méthode de Wu	5.25	1.16	3.1	1.2	4.3
méthode de Xiang	4.16	0.56	1.86	0.81	2.64

temps en secondes	maison	meloid	papillon	poisson	Zelda
notre méthode	0.06	0.02	0.31	0.04	0.29
méthode de Wu	0.98	0.03	4.0	0.1	2.9
méthode de Xiang	0.6	0.08	2.68	0.11	1.76

TAB. 3.3 – Temps de calculs requis par notre méthode, celle de Wu et celle de Xiang pour quantifier les 4 images tests en 16 couleurs. Notre méthode utilise un partitionnement initial du cube RGB en 4096 sous cubes.

Afin de comparer les performances de notre algorithme et celles de de la méthode de Xiang, nous avons utilisé les méthodes de comparaisons d'images couleurs définies par Alain Trémeau dans son habilitation à diriger des recherches [187]. Celui-ci a en effet conçu des algorithmes permettant de comparer deux images couleurs I_1 et I_2 en effectuant les opérations suivantes pour chaque pixels (i, j) des deux images :

1. calculer deux valeurs moyennes sur un voisinage de (i, j) dans I_1 et I_2 ;
2. soustraire ces valeurs et les normaliser entre 0 et 1 en fonction d'un critère local. Une valeur égale à 0 signifie une différence maximum.

Les valeurs associées à chaque pixel de l'image résultat dépendent des critères utilisés pour la comparaison. Les 6 critères définis par Trémeau sont :

1. la luminance : la valeur calculée en chaque pixel et sur chaque image est la luminance moyenne sur le voisinage ;
2. la chrominance : les moyennes des coordonnées chromatiques sont calculées sur chaque image ;
3. l'émergence : l'émergence d'un pixel est définie comme la moyenne des distances Euclidienne entre sa couleur et celle de ses voisins ;
4. la corrélation : la covariance locale entre les couleurs des deux images est utilisée. La covariance est normalisée par le produit des écarts types calculés sur les deux images.
5. la détection de contours : un gradient couleur est appliqué sur les deux images.
6. le critère de Fisher : ce critère calcule pour chaque pixel la distance entre les couleurs moyennes calculées sur un voisinage du pixel dans les deux images.

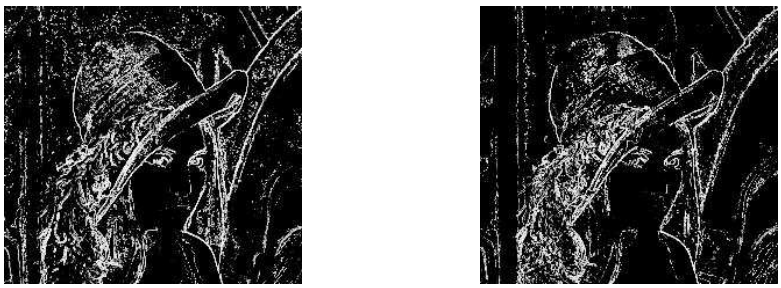
Une description plus complète de ces critères peut être trouvée dans [187].

Les critères décrits précédemment forment une image de différence en niveaux de gris à partir de deux images couleurs. Trémeau [187] propose une méthode permettant de sommer les niveaux gris et de normaliser le nombre obtenu entre 0 et 100 de façon à ce qu'une valeur égale à 100 corresponde à deux images identiques pour le critère considéré tandis qu'une valeur nulle correspond à une différence maximale. Les résultats présentés sur la Table 3.4 montrent que notre méthode obtient de plus grande valeurs (et donc une plus forte similarité avec l'image originale) que la méthode de Xiang pour tous les critères considérés à l'exception du critère de luminance. Notre méthode

produit donc des images plus proches des images originales que la méthode de Xiang selon les critères définis par Alain Trémeau.

	luminance	chrominance	emergence	correlation	gradient	Fisher
notre méthode	77.4	20.7	11.1	40.6	35.5	15.0
méthode de Xiang	64.4	21.1	8.6	31.9	24.0	12.3

TAB. 3.4 – Différences entre l'image Lenna originale et les image quantifiées en 16 couleurs par notre méthode et celle de Xiang. L'espace couleur utilisé est l'espace CIE *Lab*.



(a) (b)

FIG. 3.10 – Différence de Fisher entre l'image Lenna originale et ses versions quantifiées en 16 couleurs par notre méthode (a) et celle de Xiang (b).

3.2.8 Les méthodes de quantification spatiale

Nous avons vu dans les section 3.2.3 et 3.2.6 plusieurs méthodes basées sur une minimisation de l'erreur de partition. L'utilisation de l'erreur de partition dans le cadre de la quantification repose sur l'hypothèse qu'une quantification avec une faible erreur de partition produira une image proche de l'originale. Cette hypothèse est partiellement confirmée par l'équation suivante :

$$E(\mathcal{P}) = \sum_{i=1}^{K} \boldsymbol{SE}(\boldsymbol{C}_i) = \sum_{(i,j)\in\{1,\ldots,m\}\times\{1,\ldots,n\}} \|I(i,j) - I'(i,j)\|^2 \tag{3.14}$$

où $m \times n$ représente la taille de l'image originale et $I(i,j)$, $I'(i,j)$ représente la couleur du pixel (i,j) respectivement dans l'image originale et l'image quantifiée.

L'erreur de partition peut donc être comprise comme la somme des carrés des distances entre les couleurs des pixels de l'image originale et de l'image quantifiée. Malgré cette intéressante propriété, l'erreur de partition ne peut pas être considérée comme une distance visuelle entre les deux images. Ceci est confirmé par l'expérience illustrée sur la figure 3.12. Les 62.750 couleurs de l'image représentée sur la figure 3.12a) sont ramenées à 8 couleurs par notre algorithme de quantification mixte [41] (section 3.2.7b)). L'image quantifiée (figure 3.12b)) est alors améliorée par une étape


anemone ($722 \star 471$)


fille ($256 \star 256$)


fleurs ($462 \star 416$)


grenouille ($256 \star 256$)


Lenna ($512 \star 480$)


maison ($256 \star 256$)


meloid ($129 \star 200$)


papillon ($512 \star 512$)


poisson (256×256)


Zelda ($704 \star 574$)

FIG. 3.11 – Images couleurs utilisées pour les expériences représentées sur les tables 3.2 et 3.3. Le nom et la taille de chaque image sont représentés au dessous de celle-ci.

de dithering basée sur l'algorithme de Floyd-Stenberg [79] (figure 3.12c). Malgré l'amélioration de la qualité de l'image apportée par l'algorithme de dithering, l'erreur de partition de l'image représentée sur la figure 3.12c) est une fois et demi plus importante que l'erreur de partition de l'image représentée sur la figure 3.12b). Ce résultat *a priori* surprenant est dû aux différents critères utilisés par les algorithmes de quantification et de dithering. L'algorithme de dithering optimise l'arrangement local des couleurs représentatives alors que l'algorithme de quantification tente d'obtenir une partition en un ensemble de multi-ensembles homogènes.

(a) (b) (c)

FIG. 3.12 – L'image originale (a), l'image quantifiée en 8 couleurs (b) et (b) améliorée par une étape de dithering(c)

La prise en compte des propriétés locales de l'image dans le processus de quantification peut s'effectuer en remplaçant la fonction de fréquence f par une fonction de fréquence pondérée W, où $W(c)$ est défini comme une somme d'attribus locaux calculés sur tous les pixels de couleur c. Notons que de ce point de vue, la fonction de fréquence f n'est qu'un type particulier de fonction de fréquence pondérée où le poids de chaque pixel est égal à 1. Ce type de méthode a été employé avec succès par Bouman [17, 18] et Balasubramanian [6, 5], les deux auteurs ayant ultérieurement combiné leurs méthodes [4].

Une autre méthode sans doute plus prometteuse, consiste à optimiser conjointement la sélection des couleurs représentatives et leur localisation dans l'image. Ces méthodes font donc une double optimisation dans l'espace des couleurs choisi et dans le plan de l'image. Nous avons choisi d'appeler ces méthodes des méthodes de quantification *spatiales* afin de souligner l'aspect placement des couleurs représentatives qui constitue l'originalité de ce type de méthodes. Cette dénomination à également été choisie par Puzicha et al. [162] qui ont écrit un des articles fondateur de ce type de méthode. Ces méthodes sont basées sur une modélisation de certaines caractéristiques du système visuel humain. Une de ces caractéristiques est la décroissance rapide des capacités du système visuel humain à distinguer différentes couleurs pour de hautes fréquences spatiales. Ce phénomène crée des couleurs additionnelles par un moyennage local autour de chaque pixel. Ce phénomène est abondamment utilisé par les méthodes de dithering pour améliorer la qualité de l'image mais ne peut pas être facilement intégré dans un processus de quantification puisque les couleurs représentatives ne sont connues qu'en sortie du processus de quantification.

Plusieurs auteurs [78, 154, 162] ont proposé de modéliser le moyennage spatial effectué par l'oeil humain à l'aide de trois filtres passe bas $(W_k)_{k\in\{1,2,3\}}$. Étant donnée une image $I = (I_i)_{i\in\{1,\ldots,\mathcal{N}\}}$

où $\mathcal{N}$ représente la taille de l'image, la $k^{\text{ème}}$ composante de l'image perçue $\tilde{I}$ est alors définie par :

$$\forall k \in \{1,2,3\}\ \forall i \in \{1,\ldots,\mathcal{N}\} \quad \tilde{I}_k(i) = \sum_{j \in S_{i,k}} W_k(j) I(j)_k$$

où $S_{i,k}$ est le support du filtre W_k en position i.

Étant donné un ensemble de couleurs représentatives $\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}$, la localisation de chaque couleur représentative peut être codée à l'aide d'une matrice $\mathcal{L}$ de taille $\mathcal{N} \times K$ telle que $\mathcal{L}(i,j)$ est égal à 1 si le pixel I_i est affecté à la couleur représentative c_j et 0 sinon. Si nous stockons l'ensemble des couleurs représentatives dans une matrice T de taille $K \times 3$, telle que $T(i,k)$ représente la $k_{\text{ème}}$ composante de la couleur représentative c_i, l'image résultat I_Q peut s'écrire :

$$\forall i \in \{1,\ldots,\mathcal{N}\} \quad I_Q(i) = \mathcal{L}(i,.)T$$

où $\mathcal{L}(i,.)$ représente la $i^{\text{ème}}$ ligne de la matrice $\mathcal{L}$.

L'image quantifiée perçue $\tilde{I}$ est alors définie par :

$$\forall k \in \{1,2,3\}\ \forall i \in \{1,\ldots,\mathcal{N}\} \quad \tilde{I_Q}(i)_k = \sum_{j \in S_{i,k}} W_k(j)\mathcal{L}(j,.)T(.,k)$$

où $T(.,k)$ représente la $k^{\text{ème}}$ colonne de la matrice T. Notez que $I_Q(i)_k = \mathcal{L}(i,.)T(.,k)$.

La distance entre la perception de l'image originale est celle de l'image quantifiée peut alors s'écrire comme la somme des carrés des distances entre les couleurs perçues :

$$\begin{aligned} H(\mathcal{L},T) &= \sum_{k=1}^{3}\sum_{i=1}^{\mathcal{N}} \left(\tilde{I}(i)_k - \tilde{I_Q}(i)_k\right)^2 \\ &= \sum_{k=1}^{3}\sum_{i=1}^{\mathcal{N}} \left(\sum_{j \in S_{i,k}} W_k(j) I(j)_k - \sum_{j \in S_{i,k}} W_k(j)\mathcal{L}(j,.)T(.,k)\right)^2. \end{aligned}$$

La minimisation de $H(\mathcal{L},T)$ nécessite donc de fixer conjointement les valeur des matrices $\mathcal{L}$ et T. Une minimisation de $H(\mathcal{L},T)$ utilisant une valeur fixe de T a été proposée par plusieurs méthodes de dithering [78, 154]. Toutefois, une minimisation de $H(\mathcal{L},T)$ fixant simultanément les valeurs de $\mathcal{L}$ et T n'a été proposée que récemment par Puzicha [162]. Celui-ci utilise le schéma de minimisation suivant :

1. Une minimisation de $H(\mathcal{L},T)$ basée sur $\mathcal{L}$ en fixant T,
2. une minimisation de $H(\mathcal{L},T)$ par rapport à T en gardant $\mathcal{L}$ fixé.

Cette stratégie en deux étapes est itérée jusqu'à convergence. Puzicha utilise également une décomposition hiérarchique de l'image pour accélérer le processus de convergence.

3.3 L' inversion de tables de couleurs

L'inversion d'une table de couleurs est un processus permettant d'afficher une image avec un nombre restreint de couleurs représentatives. Ces couleurs peuvent être issues d'un algorithme de quantification, imposées par la table de couleurs par défaut de l'écran ou encore définies arbitrairement par l'utilisateur. Afin de minimiser la distance visuelle entre l'image de sortie et l'image originale, les algorithmes d'inversion de tables de couleurs affectent chaque couleur d'entrée c à sa couleur représentative la plus proche $\boldsymbol{Q}(c)$. La fonction $\boldsymbol{Q}$ peut donc être définie par :

$$\boldsymbol{Q}\left(\begin{array}{ccl} \mathcal{C} & \rightarrow & \{c_1,\ldots,c_K\} \\ c & \mapsto & \boldsymbol{Q}(c) = \arg\min_{z\in\{\mathbf{c}_1,\ldots,\mathbf{c}_K\}} \|z-c\| \end{array}\right) \tag{3.15}$$

où $\mathcal{C}$ représente l'ensemble des couleurs de l'image originale et $\{\mathbf{c}_1,\ldots,\mathbf{c}_k\}$ l'ensemble des couleurs représentatives. Le symbole $\|z-c\|$ représente la norme du vecteur $z-c$ définie dans un espace métrique donné (par exemple la norme Euclidienne dans l'espace RGB). Notons qu'il existe d'autres types de distances [58, 59]. Certaines de ces distances sont par exemple basées sur une projection de l'ensemble des couleurs sur un axe réel [126]. De telles projections permettent de définir un ordre sur l'ensemble des couleurs qui est utilisé dans le cadre du filtrage d'images couleurs. Cet ordre définit implicitement une distance. Toutefois ce type de distance n'est pas utilisé dans le cadre de l'inversion de table de couleur.

La valeur de $\boldsymbol{Q}(c)$ pour une couleur d'entrée c donnée peut être calculée grâce à une recherche exhaustive du minimum de $\|z-c\|$ pour z parcourant l'ensemble des couleurs représentatives. Étant donnée une image de taille 256×256 et une table de couleurs de taille 256, cet algorithme trivial nécessitera plus de 16 millions de calculs de distances (voir équation 3.15). Malgré sa simplicité, cette méthode ne peut donc être utilisée pour de grandes images ou des applications interactives. Plusieurs méthodes ont donc été mises au point pour optimiser la recherche de la plus proche couleur représentative. La complexité des principales méthodes est représentée dans la Table 3.9.

3.3.1 Amélioration des calculs de distance

Les améliorations de la méthode triviale sont nombreuses : Poskanzer [159] a proposé d'améliorer la recherche en utilisant une table de hachage de façon à ne pas recalculer la couleur représentative d'une couleur déjà rencontrée. Toutefois cette amélioration reste inefficace pour des images comportant un nombre important de couleurs différentes. Une autre approche consiste à approximer la norme L_2 par une norme moins coûteuse en temps de calculs. Chaudhuri et al. [47] ont proposé la norme L_α en tant qu'approximation de la distance Euclidienne définie par L_2. La norme L_α d'une couleur $\mathbf{c}$ étant définie par :

$$\begin{array}{rcl} \|\mathbf{c}\|_\alpha & = & (1-\alpha)\|\mathbf{c}\|_1 + \alpha\|\mathbf{c}\|_\infty \\ & = & (1-\alpha)\sum_{j=1}^{3}|\mathbf{c}^j| + \alpha\max_{j\in\{1,2,3\}}|\mathbf{c}^j|. \end{array}$$

D'après les expériences menées par Verevka [192], la norme $L_{\frac{1}{2}}$ accélère la recherche de façon significative sans introduire une perte notable de la qualité de l'image résultat.

La recherche peut être encore réduite en utilisant les considérations suivantes [100] :

1. Somme partielle :

   Si nous utilisons le carré de la norme L_2, ou les normes L_1 ou $L_{\frac{1}{2}}$, la distance entre la couleur d'entrée et une couleur représentative est définie comme la somme de trois termes. La somme partielle doit être comparée à la distance minimale avant chaque addition. Le calcul de distance n'a en effet aucune raison de continuer si une somme partielle est plus grande que la distance minimale.

2. Tri sur une coordonnée :

   Supposons que les couleurs représentatives sont triées suivant un des axes de coordonnées (par ex. le premier). La recherche débute alors sur la couleur représentative dont la première coordonnée est la plus proche de celle de la couleur d'entrée et continue ensuite dans l'ordre

croissant des distances sur la première coordonnée. La recherche se termine quand la distance sur la première coordonnée entre la couleur représentative courante et la couleur d'entrée est plus grande que la distance minimale. Notons que le temps moyen de recherche sera d'autant plus court que la variance sur l'axe des coordonnées choisi sera grand.

3. Distance du plus proche voisin :
   Étant donné une couleur représentative $\mathbf{c}_j$ avec $j \in \{1, \ldots, K\}$, sa plus proche couleur représentative $\mathbf{c}_{q(j)}$ est définie par :

$$q(j) = \arg \min_{k \in \{1,\ldots,K\}, k \neq j} d(\mathbf{c}_j, \mathbf{c}_k).$$

   Supposons que la couleur représentative courante $\mathbf{c}_i$ parcourue par l'algorithme soit la plus proche de la couleur d'entrée $\mathbf{c}$. Supposons de plus que la distance minimale courante $\Sigma_{min} = d(\mathbf{c}, \mathbf{c}_i)$ soit plus petite que la moitié de $d(\mathbf{c}_i, \mathbf{c}_{q(i)})$. On a donc :

$$\begin{cases} \Sigma_{min} & = & d(\mathbf{c}, \mathbf{c}_i) \\ \Sigma_{min} & \leq & \frac{1}{2} d(\mathbf{c}_i, \mathbf{c}_{q(i)}). \end{cases}$$

   Pour toute autre couleur représentative $\mathbf{c}_k$ on a :

$$\begin{array}{rlll} & 2\Sigma_{min} & \leq & d(\mathbf{c}_i, \mathbf{c}_{q(i)}) \qquad \leq d(\mathbf{c}_i, \mathbf{c}_k) \leq d(\mathbf{c}_i, \mathbf{c}) + d(\mathbf{c}, \mathbf{c}_k) \\ \Rightarrow & 2\Sigma_{min} & \leq & \Sigma_{min} + d(\mathbf{c}, \mathbf{c}_k) \\ \Rightarrow & \Sigma_{min} & \leq & d(\mathbf{c}, \mathbf{c}_k). \end{array}$$

   L'algorithme d'inversion de table de couleur peut donc retourner directement $\mathbf{c}_i$ puisqu'aucune autre couleur représentative ne peut être plus proche de $\mathbf{c}$ que $\mathbf{c}_i$.

La figure 3.13 représente les différents temps d'exécution en secondes sur l'image test Lenna (figure 3.2(a)) pour des tailles de table de couleurs variant de 16 à 520. Ces temps ont été mesurés sur un PC Pentium II 350. On peut constater sur cette figure que la méthode du plus proche voisin et celle n'effectuant qu'une somme partielle n'induisent pas un gain significatif dans les temps de calculs. Assez curieusement, la méthode basée sur le calcul de la norme $L_{\frac{1}{2}}$ induit systématiquement des temps de calculs plus longs que la méthode triviale. Ceci peut s'expliquer par l'architecture du Pentium II qui n'utilise qu'un seul pipeline pour les instructions. La substitution de la norme L_2 par la norme $L_{\frac{1}{2}}$ revient en effet à remplacer une série de multiplications par des tests pour calculer la norme infinie. Ces tests brisent l'ordonnancement des instructions dans le pipeline et induisent des temps de calculs plus longs que la méthode triviale. Les seules méthodes qui induisent une amélioration significative sont les méthodes basées sur une fonction de hachage et celles utilisant un tri sur les coordonnées. Notons que ces deux dernières optimisations peuvent aisément être combinées.

3.3.2 Les méthodes dévolues à une méthode de quantification

Comme nous l'avons mentionné dans la section 3.1, tout algorithme de quantification nécessite une étape d'inversion de table de couleurs pour créer l'image résultat. L'ensemble des couleurs représentatives est construit pour minimiser une erreur de quantification définie dans un espace de couleur donné (comme $CIELuv$ ou YIQ). D'un autre coté, les algorithmes d'inversion de table de couleur sont basés sur des calculs de distances définies dans un espace de couleurs également donné.

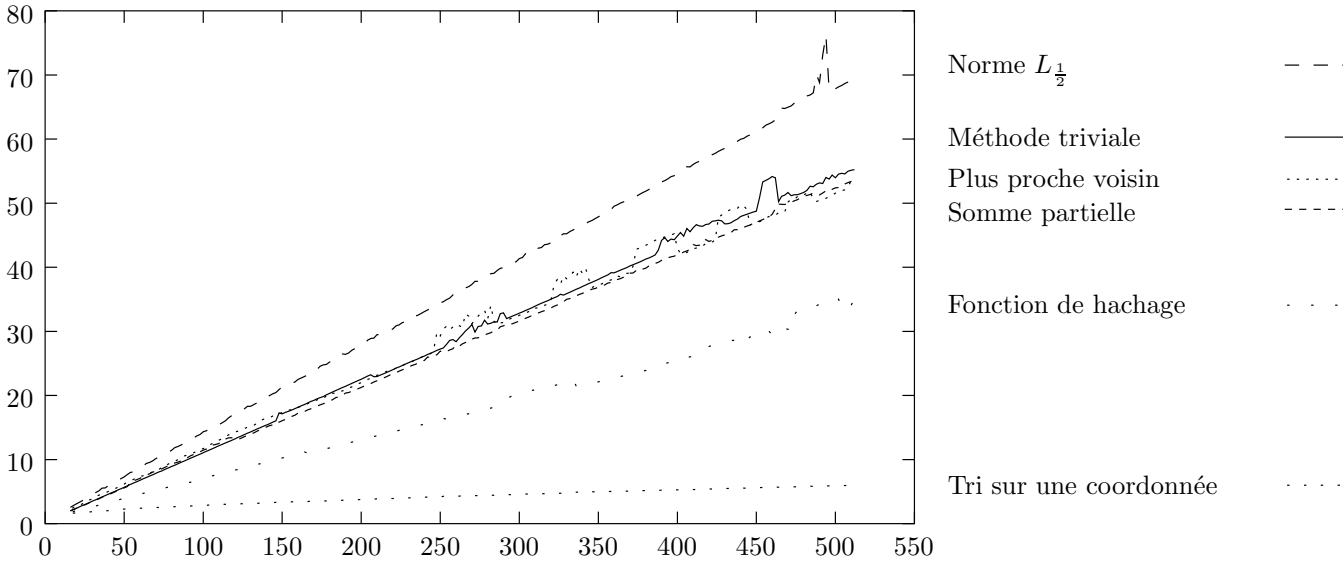


FIG. 3.13 – *Temps d'exécution en secondes des différentes méthodes sur l'image Lenna. L'axe des abscisses représente la taille de la table de couleurs*

Un algorithme d'inversion de tables de couleurs doit donc utiliser le même espace de couleurs que l'algorithme de quantification auquel il est associé afin d'être consistant avec le critère minimisé par celui-ci.

L'affectation de chaque couleur à sa plus proche couleur représentative induit une partition de l'espace couleur par un diagramme de Voronoï 3D [160, 13] défini par les couleurs représentatives. Cette partition de l'espace par un diagramme de Voronoï $3D$ ne signifie pas que celui-ci doit être obligatoire calculé. De fait, les partitions définies par les algorithmes de quantification ne constituent généralement pas des diagrammes de Voronoï. Toutefois, un tel diagramme peut être efficacement approximé par la partition définie par un algorithme de quantification si l'erreur de partition de ce dernier est suffisamment faible. De plus, la prise en compte de la partition définie par un algorithme de quantification permet d'utiliser les structures de données définies par celui-ci pour réduire le coût de la recherche. De fait, l'utilisation de l'arbre de découpe généré par un algorithme descendant (section 3.2.3) permet de retrouver la couleur représentative d'une couleur d'entrée en $\mathcal{O}(\log_2(K))$ tests (Fig. 3.14).

3.3.3 Découpe de l'espace couleur par des $k-d$ arbres

L'inversion de tables de couleurs peut également être étudiée dans le cadre plus général de la recherche des plus proches voisins. Friedman [11] a proposé un algorithme basé sur une optimisation des $k-d$ arbres afin de trouver les m plus proches voisins d'une couleur donnée. Un $k-d$ arbre est un arbre binaire construit par une découpe récursive d'un multi-ensemble initial (section 3.2.3), chaque multi-ensemble étant découpé perpendiculairement à son axe de coordonnée de plus grande variance par une coupe médiane. On minimise donc ainsi la variance des deux sous multi-ensembles produit,

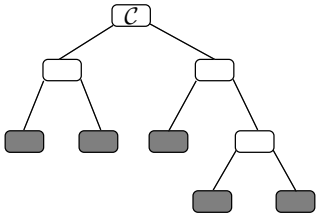


FIG. 3.14 – Arbre de découpe défini par une méthode de quantification descendante.

chaque multi-ensemble ayant approximativement le même cardinal. Cette découpe récursive se termine lorsque chaque feuille de l'arbre contient un nombre de couleurs inférieur à une constante.

La procédure de recherche initialise une liste des m plus proches voisins de la couleur d'entrée précédemment rencontrée. Chaque fois qu'une couleur est examinée et trouvée plus proche de la couleur d'entrée qu'un des m plus proches voisins, la liste est mise à jour. Si le noeud courant n'est pas une feuille, la recherche est relancée sur celui de ces fils qui contient la couleur d'entrée. En retour de fonction, un test est effectué pour déterminer s'il est nécessaire d'examiner l'autre fils. Ce test, dont le nom pourrait se traduire par test de chevauchement «cube-boule» (*bounds overlap-ball*) vérifie si le cube contenant le fils non encore testé intersecte la boule centrée sur la couleur d'entrée et ayant pour rayon la distance entre cette couleur et le $m^{\text{ème}}$ voisin. Si le test échoue, i.e. si l'intersection entre le cube et la boule est vide, le fils restant n'a pas à être examiné puisqu'aucune de ses couleurs ne pourra modifier la liste des m plus proche voisins. Si le cube et la boule ont une intersection non vide, alors la procédure de recherche est relancée récursivement sur le fils restant.

La figure 3.15 illustre le fonctionnement d'un $k-d$ arbre sur un cas $2D$. La détermination de la boule englobant les m plus proches voisins est effectuée dans le noeud contenant la couleur fournie en entré (le noeud 1.1 dans notre exemple). Cette boule ne coupe pas le noeud 1.2 et la recherche des m plus proche voisins n'est donc pas relancée sur ce noeud. En revanche, elle coupe le noeud 2 et une recherche devra être relancée sur celui-ci lorsque le processus de recherche reviendra sur le noeud 0. Le test «cube-boule» permettra à ce moment là de rejeter *a priori* le noeud 2.2 qui ne coupe pas la boule.

Cet algorithme peut être appliqué à l'inversion de tables de couleurs en fixant m à 1 et en utilisant la table de couleurs comme multi-ensemble initial stocké dans l'arbre de découpe. D'après Friedman, la couleur représentative de chaque couleur d'entrée peut ainsi être retrouvée en $\mathcal{O}(\log(K))$ tests.

3.3.4 Inversion de tables de couleurs par un tri local

La méthode d'Heckbert par tri local [97] utilise le même principe que le test d'intersection cube-boule de Friedman, mais plutôt que de définir une découpe récursive de la table de couleur Heckbert définit une partition uniforme du cube RGB en N cubes. Chaque cube est associé à une liste de couleurs représentatives. Chacune de ces couleurs représentatives est la couleur représentative d'au moins un point du cube. La couleur représentative de toute couleur contenue dans un cube appartient donc à la liste associée à celui-ci. La liste de chaque cube est définie en calculant la

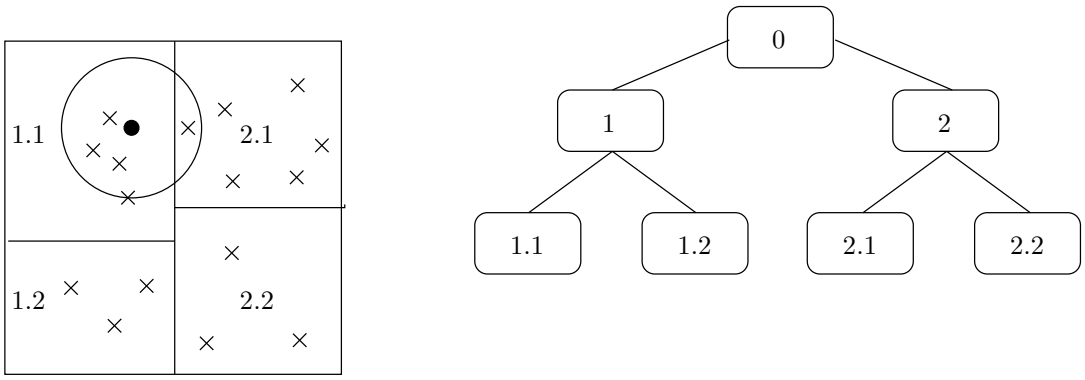


FIG. 3.15 – *Fonctionnement d'un $k-d$ arbre sur un exemple $2D$. Les crois ($\times$) symbolisent les éléments de l'ensemble.*

distance r entre la couleur représentative la plus proche du centre du cube (la couleur A sur la figure 3.16) et le coin du cube le plus éloigné de cette couleur représentative. Cette distance majore la distance entre n'importe qu'elle couleur du cube et sa couleur représentative. Donc, toute couleur représentative à une distance supérieure à r du cube peut être rejetée de la liste.

Étant donnée une couleur c, la méthode d'Heckbert détermine le cube contenant c, et parcourt sa liste de couleurs représentatives associée de façon à déterminer la couleur représentative la plus proche. La complexité de cette méthode est proportinnelle à la taille moyenne des listes stockées dans chaque cube. Cette taille est déterminée par le nombre et la distribution des couleurs représentatives ainsi que par le nombre N de cubes formant une partition du cube RGB. Si nous notons L cette taille moyenne, la complexité du pré-traitement définissant la partition du cube RGB et la liste des couleurs représentatives associées à chaque cube est égale à $\mathcal{O}(NK + NL\log(L))$. Les tests effectués par Heckbert montrent que le nombre de calculs de distances requis pour calculer la couleur représentative $\boldsymbol{Q}(c)$ d'une couleur c est divisé par 23 pour une quantification en $K = 256$ couleurs utilisant une partition du cube RGB en $N = 512$ cubes. Toutefois, pour un nombre N fixe de cubes composant la partition, la méthode d'Heckbert reste approximativement linéaire en fonction du nombre de couleurs représentatives. De plus, la valeur optimum de N reste difficile à évaluer.

La méthode d'Heckbert à été spécifiquement conçue pour l'espace RGB. On peut toutefois l'adapter à d'autres espaces couleur en calculant la boite englobante de la transformée du cube RGB dans le nouvel espace (Fig. 3.17)[1]. Celle-ci peut être définie en calculant les valeurs minimum et maximum de chaque coordonnée dans le nouvel espace pour tout triplé RGB appartenant au cube. Cette solution nécessite toutefois d'allouer inutilement un nombre de cubes initiaux important. Nous devrons en effet, allouer des cubes :

- ne correspondant à la transformation d'aucun triplé RGB appartenant au cube RGB ;
- ne correspondant à la transformation d'aucun triplé RGB présent dans l'image.

Dans les deux cas, le cube ne pourra être utile lors de l'étape d'inversion de table de couleur et a donc été alloué inutilement.

[1] cette figure est issue de [43] et a été conçu par Alain Trémeau

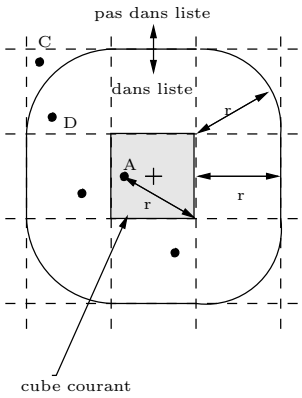


FIG. 3.16 – La recherche par tri local d'Heckbert. La couleur représentative A est la plus proche du centre du cube courant. La distance entre A et le coin du cube le plus éloigné définit r. La couleur représentative D appartient à la liste du cube alors que C situé à une distance supérieure à r du centre du cube est rejeté de la liste

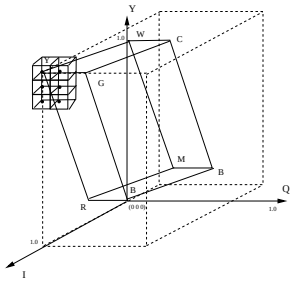


FIG. 3.17 – Boite englobante de la transformée du cube RGB dans l'espace YIQ.

3.3.5 Découpe de l'espace couleur par un diagramme de Voronoï 3D

La méthode de Thomas [184] est basée sur un diagramme de Voronoï $3D$ discret [160, 13] défini par l'ensemble des couleurs représentatives. Un diagramme de Voronoï discret défini par p points est une partition d'une image discrète en un ensemble de p cellules, chaque pixel d'une cellule étant plus proche du point associé à celle-ci que de tous les autres. Un des principaux avantages des diagrammes de Voronoï discrets sur les diagrammes de Voronoï réels est qu'ils peuvent être calculés par des méthodes incrémentales qui initialisent l'image à partitionner. Donc, pour un diagramme de Voronoï discret, l'index de la cellule contenant un pixel donné est stocké dans l'image à l'adresse du pixel.

La méthode de Thomas stocke le diagramme de Voronoï $3D$ par un tableau d'entiers, chaque entrée de ce tableau représente une couleur et contient l'index de sa plus proche couleur représentative. Le principal avantage de la méthode de Thomas est qu'une fois le diagramme de Voronoï $3D$ calculé, la couleur représentative de n'importe quelle couleur de l'image peut être retrouvée sans aucun calcul. Cette méthode est donc tout a fait adaptée pour affecter n'importe quelle couleur à sa plus proche couleur représentative dans la table de couleurs par défaut de l'écran. Elle a toutefois plusieurs désavantages : tout d'abord, cette méthode calcule la couleur représentative de n'importe quelle couleur affichable. Ceci implique donc de nombreux calcul inutiles lorsque la méthode est uniquement destinée à l'affichage de quelques images. De plus, si nous utilisons l'espace RGB, le calcul d'un diagramme de Voronoï $3D$ implique l'initialisation et le stockage de 256^3 indices. Thomas propose de réduire les temps de calculs et la place mémoire requis par l'algorithme en supprimant les 3 bits de poids faibles de chaque composante RGB. Le diagramme de Voronoï $3D$ est alors calculé sur un cube de taille $32 \times 32 \times 32$. Cette heuristique réduit les temps de calculs et la place mémoire requis par l'algorithme mais introduit différents artéfacts liés à cette perte de résolution. De plus, une majorité de méthode de quantification [193, 201, 200, 25] utilisent d'autres espaces que l'espace RGB tels que $CIELuv$, $CIELab$ [182] ou YC_rC_b [6] qui sont plus adaptés à la modélisation de la vision humaine des couleurs que l'espace RGB. Dans ce cas, comme nous l'avons vu en section 3.3.2, l'algorithme d'inversion de table de couleurs et l'algorithme de quantification doivent utiliser le même espace de couleur. La méthode de Thomas doit donc dans ce cas allouer et initialiser un tableau $3D$ qui englobe la transformation du cube RGB dans le nouvel espace. Selon les caractéristiques de cette transformation, les contraintes liées au nombre important de données à stocker et initialiser peuvent être renforcées.

3.3.6 Ma contribution à l'inversion de table de couleurs

Nous avons vu dans la section 3.2.2 que la quantité d'informations d'un multi-ensemble portées par un axe de coordonnée pouvait se mesurer par la variance le long de cet axe. De même, si nous effectuons une transformation en composantes principales d'un multi-ensemble, la quantité d'information portée par chacun des vecteurs propres se mesure à l'aide de la valeur propre de la matrice de covariance associée à celui-ci (voir équations 3.1 et 3.2 section 3.2.2). En effet, les valeurs propres de la matrice de covariance sont égales aux variances calculées le long de chacun des vecteurs propres. Comme nous l'avons vu en section 3.2.2, la quantité d'informations portées par un vecteur propre e_i se mesure plus précisément par :

$$\frac{v_i}{\sum_{i=1}^{3} v_i}.$$

Supposons, pour le reste de cette section, que les valeurs propres sont ordonnées de façon décroissante ($v_1 \geq v_2 \geq v_3$). Si nous définissons le plan principal, P_{princ} par les deux premiers vecteurs propres de la matrice de covariance, la quantité d'informations portées par celui-ci est égale à :

$$\frac{v_1 + v_2}{\sum_{i=1}^{3} v_i}.$$

alors que l'information restante portée par le vecteur propre e_3 est égale à :

$$\frac{v_3}{\sum_{i=1}^{3} v_i}.$$

Nous avons vu dans la section 3.3.5 que la principale limitation du diagramme de Voronoï $3D$ défini par Thomas venait du nombre important de couleurs inutiles (car non affichées) qui devaient être initialisées et stockées. Nous avons donc conçu une méthode basée sur une projection de l'ensemble des couleurs d'une image sur son plan principal P_{princ}. Le principal avantage de cette méthode est de réduire la quantité de données à traiter.

3.3.6.a L'étape de projection

Les expériences menées par Otha [151] et confirmées par nos propres expériences (voir Tables 3.5 et 3.6) montrent que l'on peut généralement s'attendre à ce qu'au moins 90% de l'information d'une image soit contenue dans le plan P_{princ}. Il est bien sur toujours possible de choisir une image telle que ce pourcentage soit moins élevé. Toutefois, les expériences que nous avons menées sur une série de 105 images fournies avec le logiciel PhotoShop^{TM} (Table 3.6) nous ont montré que de telles images sont excessivement rares. Les images choisies comprenaient des visages, des paysages et des photos de jardins ou de fleurs.

De plus, la matrice de passage dans la base des vecteurs propres étant orthogonale, les distances calculées dans l'espace de couleur original sont équivalentes à celles calculées dans la base des vecteurs propres. Ces distances entre vecteurs $3D$ peuvent donc être efficacement approximées par des distances $2D$ calculées dans le plan P_{princ}. Notre méthode utilise cette propriété pour approximer le diagramme de Voronoï $3D$ défini par un ensemble de couleurs représentatives $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ par un diagramme $2D$ défini à partir des projections $\{p(\mathbf{c}_1), \ldots, p(\mathbf{c}_K)\}$ des couleurs représentatives sur le plan P_{princ} (figure 3.18).

Image	v_1	v_2	v_3	$\frac{v_1 + v_2}{\sum_{i=1}^{3} v_i}$
Lenna	2212	138	4	99%
Zelda	795	170	27	97%
Fille	2101	307	26	99%
House	1144	206	18	99%

TAB. 3.5 – Le pourcentage d'information contenu dans le plan P_{princ}.

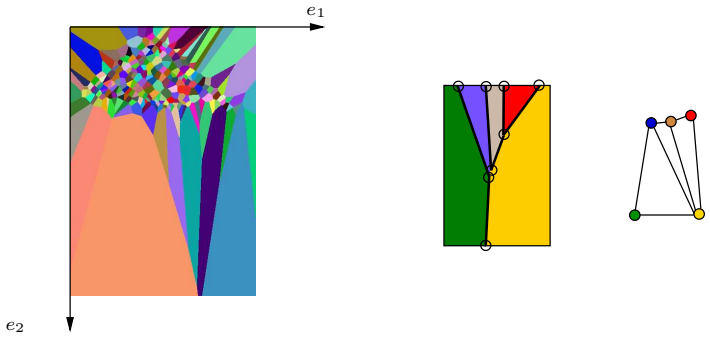


FIG. 3.18 – Deux diagrammes de Voronoï $2D$ discrets basés sur une projection sur le plan P_{princ}. Le diagramme (a) a été produit à partir de l'image Fille et d'une table de couleur de taille 256. Notez que la taille des cellules de Voronoï est petite pour de grande tailles de table de couleurs. Le diagramme (b) a été produit à partir de l'image Lenna avec 5 couleurs. Le diagramme de Delaunay associé est représenté en (c).

	Min	*Max*	*Moyenne*
$\frac{v_3}{\sum_{i=1}^{3} v_i}$	0.14%	7.07%	2.56%

TAB. 3.6 – Les valeurs min, max et moyenne en pourcentage du ratio $\frac{v_3}{\sum_{i=1}^{3} v_i}$ calculées sur 105 images naturelles. Le terme "image naturelle" représente ici des images issues de photos de scènes naturelles. Ce terme s'oppose donc aux images artificielles issues de la synthèse d'images.

La taille du tableau $3D$ codant un diagramme de Voronoï $3D$ discret est définie à partir des dimensions du cube $3D$ englobant l'ensemble des couleurs d'une image. Pour un diagramme de Voronoï $2D$ discret, cette taille est définie à partir des dimensions du carré englobant l'ensemble des projections des couleurs de l'image sur le plan P_{princ}. Une fois le carré englobant déterminé, le diagramme de Voronoï $2D$ peut être calculé en utilisant la méthode de Danielson [63] avec les sites $\{p(\mathbf{c}_1), \ldots, p(\mathbf{c}_K)\}$. La complexité de cette méthode est égale à $\mathcal{O}(|V_I|)$ où $|V_I|$ représente la taille du tableau $2D$ codant le diagramme de Voronoï.

Le calcul de V_I réclame deux parcours de l'image : dans une première étape, nous calculons les moments d'ordre 0 et 1 ainsi que la quantité R_2 (équation 3.1) du multi-ensemble de l'image de façon à déterminer sa matrice de covariance (équation 3.2). Les deux premiers vecteurs propres sont déduits de la matrice de covariance à l'aide d'une méthode incrémentale [161]. Dans un second temps, nous calculons le carré englobant des projections de chaque couleur de l'image sur P_{princ} définit par les deux premiers vecteurs propres.

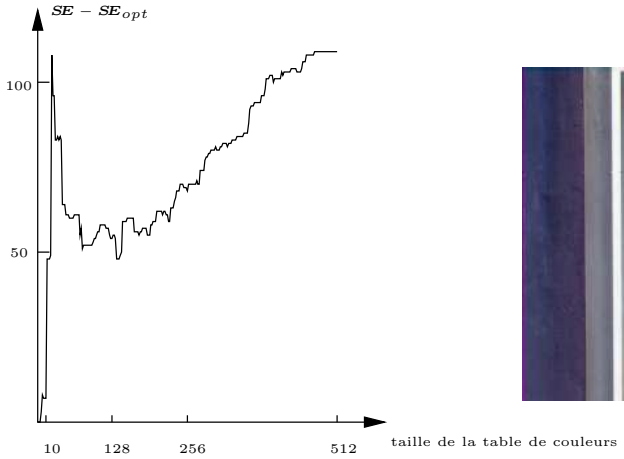
La matrice de covariance se déduit immédiatement du calcul des moments et de celui R_2. De même, le calcul des deux premiers vecteurs propres de la matrice de covariance a une complexité négligeable pour une matrice 3×3. La complexité de notre algorithme est donc déterminée par les

deux parcours de l'image et l'initialisation du tableau V_I. Cette complexité est égale à $\mathcal{O}(2|I| + |V_I|)$ où $|I|$ représente la taille de l'image.

3.3.6.b L'étape de correction

L'utilisation du diagramme V_I, pour l'inversion de tables de couleurs implique deux approximations : la première est liée à la projection sur le plan P_{princ} qui néglige les variations des couleurs suivant le troisième vecteur propre ; la seconde approximation est liée à l'utilisation d'un diagramme de Voronoï discret qui impose d'arrondir la projection de chaque couleur sur P_{princ} au point entier le plus proche. Du fait de ces deux approximations, l'index d'une cellule de Voronoï contenant la projection d'une couleur peut être différent de celui de sa couleur représentative la plus proche.

Ceci est illustré sur la figure 3.19. Cette figure montre la différence entre l'erreur de partition produite par notre méthode et celle produite par la méthode d'Heckbert qui fournit un résultat exact. L'image utilisée pour cette expérience est l'image Fille tandis que les tables de couleurs ont été générées par notre algorithme de quantification mixte (section 3.2.7). L'aspect croissant de cette courbe peut s'expliquer par la diminution de la taille des cellules de Voronoï au fur et à mesure que le nombre de couleurs représentatives augmente. Les erreurs d'arrondis dues à nos deux approximations sont alors renforcées.



(a) (b)

FIG. 3.19 – Différence entre l'erreur de partition SE produite par notre algorithme d'inversion de table de couleurs et celle (SE_{opt}) produite par une méthode exacte (a). Les données de la courbe (a) proviennent de l'image test Fille (b)

Toutefois, ces approximations ne produisent que de faibles erreurs sur les calculs de distances si bien que les erreurs se produisent souvent entre deux cellules de Voronoï adjacentes. Afin de corriger ces erreurs, nous calculons durant la construction du diagramme de Voronoï V_I le graphe dual de V_I, D_I appelé un diagramme de Delaunay [13]. Ces deux structures sont alors combinées de la façon suivante :

Étant donnée une couleur c, nous lisons dans $V_I[p(\mathbf{c})]$ l'index de la cellule de Voronoï contenant

$p(\mathbf{c})$. L'ensemble $D_I[V_I[p(\mathbf{c})]]$ contient $V_I[p(\mathbf{c})]$ ainsi que l'ensemble des indices de cellules de Voronoï adjacentes à celui-ci. Puisqu'une majorité des erreurs se produit entre cellules de Voronoï adjacentes, nous déterminons la couleur représentative la plus proche de $\mathbf{c}$ dont l'index appartient à $D_I[V_I[p(\mathbf{c})]]$. Cet index est donc déterminé par :

$$q(c) = \arg \min_{i \in D_I[V_I[p(c)]]} \|\mathbf{c}_i - \mathbf{c}\| \tag{3.16}$$

où $\mathbf{c}_i$ représente une couleur représentative dans $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ et $\|\mathbf{c}_i - \mathbf{c}\|$ la norme Euclidienne $3D$ du vecteur $\mathbf{c}_i - \mathbf{c}$.

La couleur $\mathbf{c}_{q(\mathbf{c})}$ est prise comme la couleur représentative de $\mathbf{c}$. Notez que l'index $q(c)$ est déterminé à partir de calculs de distances $3D$. Le diagramme de Voronoï $2D$ n'étant utilisé que pour restreindre le domaine de recherche. Notre méthode s'apparente donc à la méthode d'Heckbert (section 3.3.4) qui effectue une partition de l'espace RGB de façon à restreindre le domaine de recherche. D'un point de vue utilisateur, la différence principale entre les deux méthodes réside dans leurs complexités respectives.

La complexité de la fonction q est déterminée par le nombre de cellules de Voronoï adjacentes à une cellule donnée. Le théorème suivant permet de majorer cette quantité :

Théorème 1 *Soit un diagramme de Voronoï défini par K sites tels que l'on ne puisse pas trouver 4 sites co-circulaires, nous avons :*

$$\frac{1}{K} \sum_{i=1}^{K} d_i \leq 6$$

où d_i est le nombre de cellules adjacentes à la cellule i.

Preuve:

Si nous ne pouvons trouver 4 sites co-circulaires, le diagramme de Delaunay associé au diagramme de Voronoï est une triangulation du plan $\mathbb{R}^2$. Dans ce cas, nous avons [87] :

$$e - 3(v - 2) \leq 0$$

où e et v représentent respectivement, le nombre d'arêtes et de sommets du diagramme de Delaunay. Si nous notons par d'_i le cardinal de chaque noeud $(v_i)_{i \in \{1,\ldots,v\}}$, nous pouvons facilement montrer que :

$$\sum_{i=1}^{v} d'_i \leq 2e.$$

Donc :

$$\sum_{i=1}^{v} d'_i \leq 6(v-2) \Rightarrow \frac{\sum_{i=1}^{v} d'_i}{v} \leq 6 - \frac{12}{v} \leq 6$$

On a par construction :

$$v = K \text{ et } \forall i \in \{1, \ldots, K\} \quad d'_i = d_i$$

Donc :

$$\frac{1}{K} \sum_{i=1}^{K} d_i \leq 6.$$

□

Étant donné une couleur c, l'ensemble $D_I[V_I[p(c)]]$ inclut $V_I[p(c)]$ et l'ensemble des index de cellules de Voronoï adjacentes à celui-ci. La taille de $D_I[V_I[p(c)]]$ est donc égale à $d_{V_I[p(c)]} + 1$, où $d_{V_I[p(c)]}$ représente la taille du voisinage du noeud $V_I[p(c)]$ dans le diagramme de Delaunay (Théorème 1). Si nous notons par S_K le nombre moyen de calculs de distances nécessaires pour affecter un couleur à sa couleur représentative nous avons donc :

$$S_K = \frac{1}{K} \sum_{i=1}^{K} d_i + 1 \leq 7.$$

Nous avons vu en section 3.3.6.a que la complexité de notre étape de pré-traitement est égale à $\mathcal{O}(2|I| + |V_I|)$, où $|I|$ et $|V_I|$ représentent respectivement la taille de l'image et celle du diagramme de Voronoï $2D$ discret V_I. La complexité moyenne de notre algorithme est donc égale à $\mathcal{O}((S_K + 2)|I| + |V_I|)$. Cette complexité étant majorée par $\mathcal{O}(9|I| + |V_I|)$.

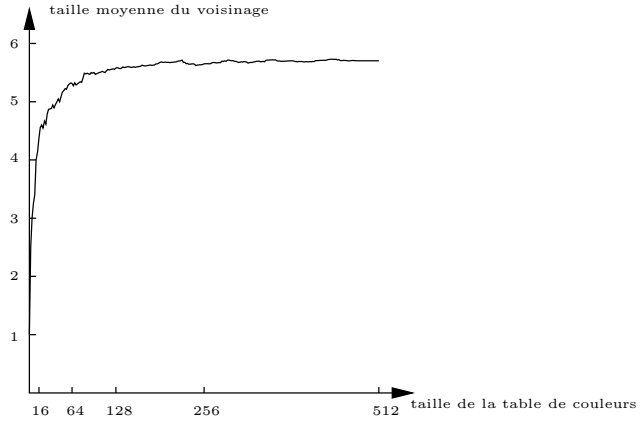


FIG. 3.20 – *Taille moyenne du voisinage des cellules de Voronoï en fonction de la taille de la table de couleurs. Les tables de couleurs ont été générées par notre algorithme de quantification mixte (section 3.2.7) en utilisant l'image Fille (figure 3.19).*

La figure 3.20 illustre la taille moyenne d'une cellule de Voronoï pour des tailles de tables de couleurs variant entre 2 et 512. Il apparaît que la taille des voisinages varie entre 5 et 6 pour des tables de couleurs de taille supérieures à 64. De plus, la taille moyenne des voisinages calculés sur un même jeu d'essais est égale à 5.5 (Table 3.7).

Cette valeur étant proche de la taille des voisinages lorsque la taille des tables de couleurs est supérieure à 64, nous pouvons considérer que S_K est approximativement constant et égal à 5.5. La complexité totale de notre algorithme est donc indépendante de la taille K des tables de couleurs pour K supérieur à 64 et égale à :$\mathcal{O}(8.5[I[+|V_I|)$. La Table 3.8 montre des résultats similaires obtenus sur d'autres jeux d'essais. Les conclusions tirés de la figure 3.20 et de la Table 3.7 peuvent donc être étendues à d'autres images tests et d'autres tables de couleurs.

	Min	*Max*	*Moyenne*
$\frac{1}{K}\sum_{i=1}^{K} d_i$	1.0	5.73	5.51

TAB. 3.7 – Les valeurs min, max et moyenne de $\frac{1}{K}\sum_{i=1}^{K} d_i$ calculées sur 255 tables de couleurs. Les tables de couleurs utilisées sont les mêmes que celles utilisées pour la Figure 3.20.

Image	16 couleurs	256 couleurs
Lenna	4.25	5.7
Zelda	4.4	5.7
Fille	4.3	5.7
House	4.1	5.6

TAB. 3.8 – La taille moyenne du Voisinage des cellules de Voronoï calculée sur 4 images tests en utilisant des tables de couleurs de tailles 16 et 256.

3.3.7 Comparaison de différentes méthodes

Dans cette section nous allons comparer les performances des méthodes suivantes : La méthode triviale (sections 3.3 et 3.3.1), la méthode d'Heckbert (section 3.3.4), la méthode de Thomas (section 3.3.5) et enfin notre méthode (section 3.3.6). Un récapitulatif de la complexité des différentes méthodes est également fournie dans la Table 3.9.

La méthode triviale a été légèrement améliorée par l'utilisation d'une table de hachage permettant d'éviter le calcul de la couleur représentative d'une couleur déjà rencontrée. Le nombre N de cubes formant la partition du cube RGB défini par Heckbert a été fixé à 512 comme recommandé par celui-ci. De même, la méthode de Thomas a été testée en utilisant la pré-quantification en 5 bits recommandée par celui-ci. En revanche, notre méthode, la méthode triviale et celle d'Heckbert ont été utilisées sans cette pré - quantification puisqu'elles offrent de bons résultats sans ce pré - traitement. De plus, la méthode de Thomas étant plus spécifiquement conçue pour l'espace RGB, tous les algorithmes ont été testés dans cet espace.

Les deux critères que nous avons utilisés pour comparer ces algorithmes sont les temps de calculs et l'erreur de partition (section 3.2.2, équation 3.7). Cette erreur est calculée directement à partir de l'image résultat en effectuant la somme des carrés des distances entre la couleur des pixels de l'image résultat et de l'image originale (voir équation 3.14). Les temps de calculs ont été mesurés sur une station de Travail HP 9000.

La figure 3.21(a) illustre les temps de calculs requis par la méthode triviale ainsi que celle d'Heckbert, de Thomas et la notre. Ces temps de calculs ont été obtenus à partir de l'image Fille (Fig. 3.19(b)) et des tables de couleurs de taille comprise entre 2 et 512 obtenues par notre méthode de quantification mixte (section 3.2.7). La complexité linéaire de la méthode triviale ainsi que celle d'Heckbert apparaissent clairement sur cette figure. Notons que la méthode triviale reste compétitive vis-à-vis des méthodes plus sophistiquées pour des tailles de tables de couleurs comprises entre 2 et 64. Toutefois, la forte pente de la droite décrivant les temps de calculs requis par cette méthode conduit à des temps de calculs importants dès que la taille de la table de couleur dépasse 64. La complexité de la méthode d'Heckbert est également linéaire mais la pente de la droite

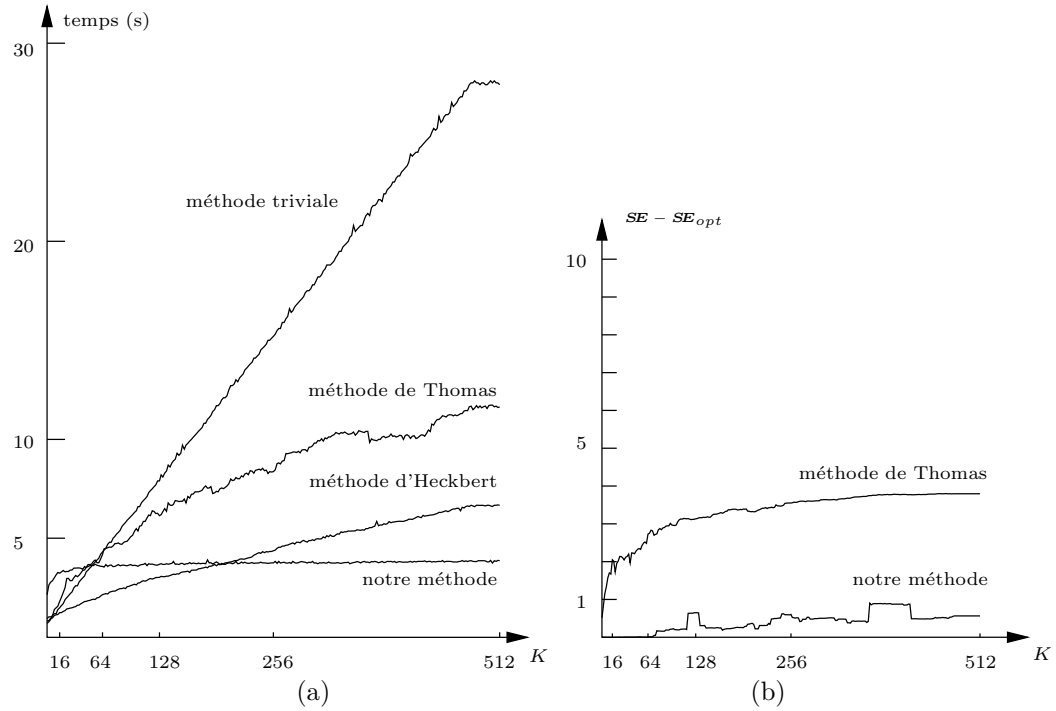


FIG. 3.21 – Les temps en secondes requis par les différentes méthodes d'inversion de table de couleurs (a) ainsi qu'une comparaison entre les erreurs de partitions produites par notre méthode et celle de Thomas (b).

décrivant les temps de calculs requis par cette méthode reste moins élevée que celle de la méthode triviale. Cette faible pente permet à la méthode d'Heckbert d'obtenir les temps d'exécution les plus faibles pour une taille de tables de couleurs variant entre 2 et 200. Toutefois, au delà de 200, les temps de calculs requis par la méthode d'Heckbert sont supérieurs à ceux de notre méthode. La dépendance logarithmique entre les temps de calculs requis par la méthode de Thomas (Table 3.9) et la taille des tables de couleurs est approximativement vérifiée sur la figure 3.21(a).

		Complexité	
Algorithme	Calcul exact	inversion de table de couleurs	Pré-traitement
Méthode triviale	oui	$\mathcal{O}(K\|I\|)$	
$k-d$ arbre	oui	$\mathcal{O}(\|I\|\log(K))$	$\mathcal{O}(K\log(K))$
Recherche par tri local	oui	$\mathcal{O}(L\|I\|)$	$\mathcal{O}(NK+NL\log(L))$
Diagramme Voronoï 3D	non	$\mathcal{O}(\|I\|)$	$\mathcal{O}(2^{15}\log(K))$
Diagramme Voronoï 2D	non	$\mathcal{O}(7\|I\|)$	$\mathcal{O}(2\|I\|+\|V_I\|)$

TAB. 3.9 – Complexité des principales méthodes d'inversion de tables de couleurs. Les symboles $|I|$ et K représentent respectivement la taille de l'image et la taille de la table de couleurs. Les symboles L et N sont définis en section 3.3.4 tandis que le symbole $|V_I|$ est défini dans la section 3.3.6.

La figure 3.21(a) confirme également que les temps de calculs requis par notre méthode sont indépendants de la taille de la table de couleurs. La forte croissance de la courbe décrivant notre algorithme pour des tailles extrêmement basses (entre 2 et 16) peut être expliquée par la forte croissance de la taille moyenne des voisinages pour ces même valeurs (Fig. 3.20). Pour des tables de couleurs de tailles plus importantes, la taille des voisinages reste approximativement constante ainsi que les temps requis par notre méthode.

Les courbes illustrées sur la figure 3.21(b) représentent l'écart entre les erreurs de partition produites par notre méthode ou celle de Thomas et celles fournissant un résultat exact. Les courbes associées aux méthodes d'Heckbert et à la méthode triviale ne sont donc pas représentées puisque ces méthodes fournissent un résultat exact (Table 3.9).

L'utilisation d'une pré-quantification en 5 bits sur chaque composante conduit la méthode de Thomas à plusieurs erreurs caractérisées par des valeurs non nulles de sa courbe. Au fur et à mesure que le nombre de couleurs représentatives augmente, la distance moyenne entre les couleurs représentatives diminue ce qui augmente les risques d'erreurs induits par la pré-quantification. Ce dernier phénomène explique l'aspect croissant de la courbe de Thomas.

La courbe associée à notre méthode reste à 0 pour des tailles de table de couleurs comprises entre 2 et 64. Ce dernier point confirme l'hypothèse faite en section 3.3.6.b qui postule que les erreurs dues à nos approximations se produisent majoritairement entre cellules de Voronoï adjacentes. Toutefois, pour des tailles de tables de couleurs plus importantes, la diminution des cellules de Voronoï conduit à des erreurs d'affectation malgré l'étape de correction. Notons toutefois que ces erreurs ne dépassent par l'unité alors que l'erreur de partition optimale varie pour K entre 64 et 512 entre 3710 et 32. Notre erreur inférieure à l'unité reste donc négligeable dans tous les cas.

La figure 3.22 représente différentes images de la grenouille (Fig. 3.11) générées à partir d'une table de 256 couleurs et les algorithmes d'inversion de table de couleurs décrits dans cette section. L'image originale est composée de 43 408 couleurs, l'algorithme de quantification utilisé est notre algorithme de quantification descendante [20] (section 3.2.4). Comme on peut le voir sur cette figure,

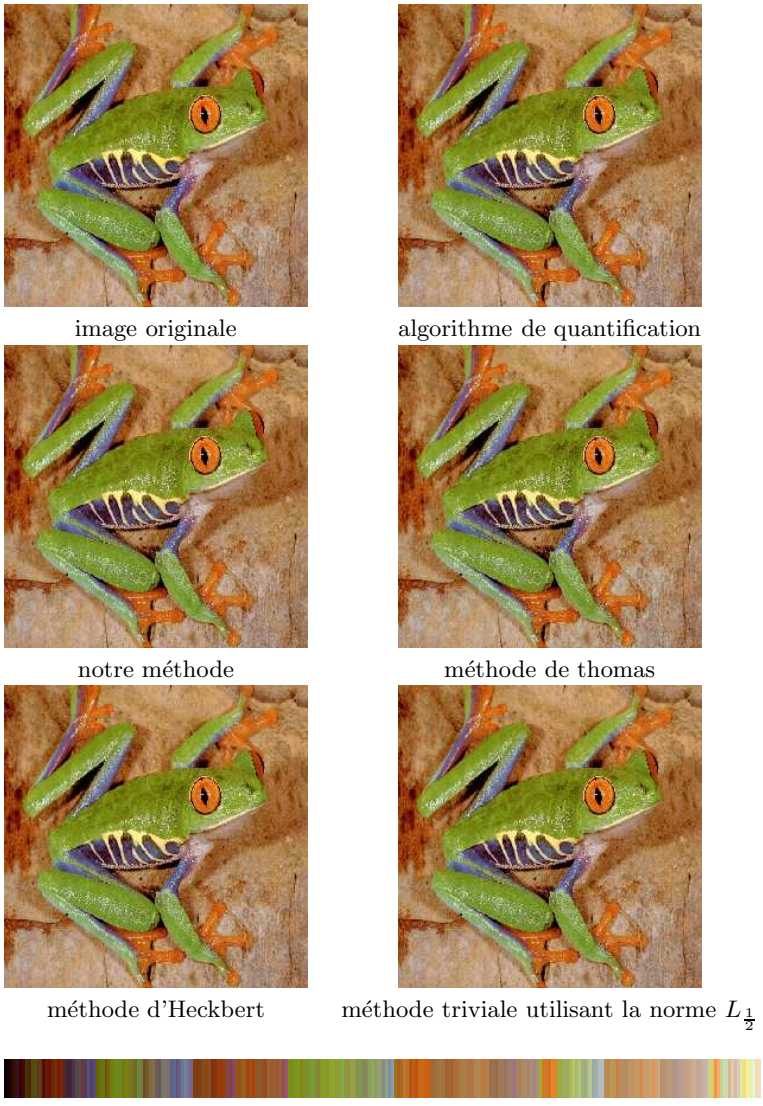


FIG. 3.22 – Image grenouille représentée en 256 couleurs à l'aide des différentes méthodes d'inversion de table de couleur. La table de couleur utilisée est représentée au bas de l'image.

les différentes images sont de qualité sensiblement équivalentes. Dans ce cas le principal critère qui va déterminer le choix d'un algorithme est le temps de calcul. Dans le cas de l'image grenouille, les temps en seconde mesurés pour chacun des algorithmes sont : $0,17$ pour notre méthode, $0,2$ pour la méthode d'Heckbert, $0,29$ pour la méthode de Thomas et $2,38$ pour la méthode triviale.

3.4 Conclusions

La quantification est un problème $\mathcal{N}P$ complet qui ne peut être résolu que par des heuristiques (section 3.2.1). Les principales familles de méthodes utilisées pour trouver une solution approchée de ce problème utilisent des approches descendantes (section 3.2.3), ascendantes (section 3.2.5) ou mixtes (section 3.2.6). Nous retrouvons ici une classification utilisée dans beaucoup de domaines dont la segmentation. La quantification peut être comprise suivant deux objectifs : Si la méthode de quantification est uniquement utilisée pour afficher des images la qualité visuelle de celles-ci est un élément important d'appréciation et une étape de dithering est souvent requise pour améliorer cette qualité. Dans ce cas les méthodes de quantification spatiales (s ection 3.2.8) qui incluent l'étape de dithering dans l'étape de quantification représentent certainement l'avenir de ce domaine de recherche. La quantification peut également être utilisée pour comprimer les données d'une image (voir également [75]). La plupart des images couleurs peuvent en effet être codées sans grande perte visuelle avec 256 couleurs plutôt qu'avec leur nombre initial de couleurs. Dans ce cas, la quantification est extrêmement intéressante pour simplifier les données d'une image et peut donc être utilisée comme pré-traitement d'une méthode de segmentation. En effet, connaître les K couleurs les plus représentatives peut grandement faciliter la tâche d'une méthode de segmentation lorsque K est suffisamment petit (entre 2 et 16). J'ai utilisé cette propriété dans le cadre de ma thèse de doctorat pour concevoir des algorithmes de segmentation semi-interactifs [33, 30, 32, 34] basés sur des méthodes de quantification. Dans ce cadre, les méthodes de quantification descendantes, ascendantes et mixtes gardent tout leur intérêt. Les méthodes de quantification peuvent également être incorporée à des algorithmes de compression d'image. En effet, dans le cadre des méthodes de compression avec perte d'information, les méthodes de quantification permettent de réduire le nombre de données de l'image tout en préservant la qualité visuelle de celle-ci.

Notre contribution à ce domaine de recherche à portée sur les méthodes de quantification descendantes et mixtes. Notre principal apport aux méthodes descendantes a certainement été la description de l'évolution de l'erreur de partition en fonction de la position du plan de coupe. Notre principale contribution aux méthodes mixtes à consisté a étudier l'influence du nombre de multi-ensembles initiaux sur l'erreur de partition finale. L'algorithme déterminé à partir de cette étude fourni des images avec des erreur quadratiques plus faibles que la plupart des méthodes descendantes [194, 201, 25] en des temps 10 fois inférieurs.

Comme nous l'avons mentionné dans la section 3.2.6 les méthodes de quantification descendantes sont certainement arrivées à un palier et il est probable qu'il ne se produira plus d'évolutions majeures dans ce domaine de recherche. En revanche notre étude de l'évolution de l'erreur de partition en fonction du nombre de multi-ensembles initiaux dans les méthodes mixtes montre que l'erreur de partition ne diminue pas linéairement en fonction du nombre de multi-ensembles initiaux (figure 3.8). De fait, on atteint très vite un palier au delà duquel augmenter le nombre de multi-ensembles initiaux n'a aucune influence sur l'erreur de partition finale. Ceci ouvre la voie à des heuristiques de fusion plus coûteuses que celle que nous avons développée mais utilisant un nombre restreint de multi-ensembles initiaux. En effet, notre méthode comme la plupart des

méthodes mixtes effectue à chaque étape la meilleure fusion pour l'étape courante. Il s'agit donc d'une stratégie gloutonne. Il serait très intéressant d'étudier des stratégies qui minimisent l'erreur de partition au bout d'un nombre donné de fusions. A plus court terme, il nous faudra également étudier l'influence du partitionnement initial.

Nous avons également présenté dans ce chapitre les principales méthodes d'inversion de tables de couleurs. Le principal avantage de la méthode que nous avons présentée est sans conteste l'indépendance de sa complexité vis-à-vis de la taille des tables de couleurs. Cette méthode est basée sur les résultats d'Otha [151] qui montrent que l'ensemble des couleurs d'une image peut être efficacement approximé par une projection $2D$. Ce type de renseignement *a priori* sur l'ensemble des couleurs d'une image illustre l'importance que les modèles de réflexion (sections 2.2, 2.4 et 2.7) pourraient avoir dans le traitement des images couleur. En effet, les méthodes de traitement d'images couleur ont de nombreux points communs avec les méthodes de traitement d'images multi - composantes [98]. Ces deux types de méthodes se différencient pourtant sur deux points :

- en amont du traitement : l'ensemble des vecteurs couleurs d'une image est produit par des phénomènes physiques décrits par les modèles de réflexion ;
- en aval du traitement : l'interprétation d'une image couleur par l'être humain repose sur un phénomène physiologique.

Le premier point permet d'obtenir des informations *a priori* sur l'ensemble des couleurs d'une image en fonction du type de scène qu'elle représente. En effet, dans le cadre d'une application devant traiter un type particulier de scène, les propriétés de celle-ci induisent des propriétés sur l'ensemble des couleurs des images de cette scène. En l'absence de modèles de réflexion ces propriétés ne peuvent être estimée qu'au moyen d'outils statistiques tels que l'analyse en composante principale. Toutefois, ces outils ne permettent pas d'expliquer les propriétés et donc de définir dans quel cadre celles-ci resteront valides. Nous avons présenté dans ce chapitre deux utilisations d'informations générales connues *a priori* :

1. La méthode de quantification de Wu [200] (section 3.2.3.a) est basée sur une découpe initiale du multi - ensemble par κ plans perpendiculaires à l'axe principal de celui-ci. Cette méthode ne présente un intérêt que si κ est supérieur à 2. La valeur de κ étant déterminée par la quantité d'information portée par l'axe principal, Wu utilise une information *a priori* sur l'ensemble des couleurs d'une image : l'axe principal du multi - ensemble d'une image couleur porte beaucoup plus d'information que les deux vecteurs propres restant. Notons que cette information n'est valide que pour le multi - ensemble correspondant à l'image. De fait, Wu découpe les multi-ensembles issues de cette première étape par un processus récursif découpant chaque multi - ensemble en deux. Il n'utilise donc plus l'information précédente qui n'est plus valide pour des multi-ensembles issues d'opérations de découpe.
2. Notre méthode d'inversion de tables de couleurs utilise également une information *a priori* sur l'ensemble des couleurs : Nous supposons que la projection du multi - ensemble initial $3D$ sur son plan principal induit une perte d'information suffisament faible pour que celle-ci puisse être ensuite «rattrapée» par l'utilisation conjointe des diagrammes de Voronoï et Delaunay.

Ce type d'information peut s'expliquer qualitativement par les modèles de réflexion. En effet, comme nous l'avons vu dans la section 2.4.2, les propriétés optiques d'un matériau influent principalement sur la chromaticité des couleurs tandis que les changements de géométrie vont principalement influencer l'intensité. Dans le cas d'images non texturées, les objets de la scène sont généralement constitués de peu de matériaux mais ont une géométrie qui varie suffisamment pour

induire d'importants changements de normales. Les variations de couleur d'une image s'effectuent donc principalement sur l'axe des intensités. Ceci explique qualitativement la prédominance de la valeur propre de l'axe principal et le fait que celui-ci soit distribué autour de l'axe des intensités. Nous ne disposons en revanche pas encore de modèle permettant d'expliquer le fait que la seconde valeur propre est généralement significativement plus importante que la troisième. On doit donc dans ce cas se «contenter» de résultats statistiques tels que ceux définis par Otha [151].

L'aspect physiologique de la couleur influe de façon importante sur les algorithmes de traitement d'image. En effet, ces algorithmes doivent souvent produire des résultats qui coïncident avec les opérations qu'aurait fait un opérateur humain ou qui correspondent à l'intuition humaine. Les algorithmes de traitement d'images couleur doivent donc prêter attention aux points suivants :

- le système visuel humain utilise un système de représentation des couleurs antagonistes avec des directions privilégiées (Fig. 2.16) et une métrique non Euclidienne [198] entre les couleurs. L'utilisation d'espaces couleurs pseudo - uniformes tels que *Lab* ou *Luv* [52, 202, 182] (section 2.3) ne permet de tenir compte de ces phénomènes que de façon partielles ;
- les combinaisons de couleurs peuvent produire des résultats contre intuitifs. Par exemple dans le cas d'un filtre moyenne [126] la moyenne d'un rouge et d'un vert produira un jaune qui n'était pas initialement présent dans l'image. L'être humain n'interprétera pas la couleur jaune comme une moyenne de rouge et de vert mais comme une couleur supplémentaire ;
- la perception humaine de la distance entre deux images couleurs doit être prise en compte dans tout algorithme minimisant un critère entre deux images. Il est en effet inutile de poursuivre les itérations d'un algorithme si celles-ci n'apporteront pas d'amélioration perceptibles. Cette notion de distance entre images couleur est difficile à évaluer et reste encore une question ouverte malgré les nombreux apports à ce domaine [188, 187, 150].

3.5 Lexique des principaux symboles

- $\boldsymbol{C}$ ..... multi-ensemble
- Cov ..... matrice de covariance
- $\mathbf{c}$ ..... vecteur couleur
- D_I ..... diagramme de Delaunay caculé à partir de l'image I dans notre méthode d'inversion de table de couleurs
- e_i ..... i$^{\text{ème}}$ vecteur propre
- $E(\mathcal{P})$ ..... erreur de la partition partition $\mathcal{P}$
- f ..... fonction de fréquence
- I ..... image
- K ..... cardinal de la table de couleur
- $\mathcal{L}$ ..... matrice codant la localisation de chaque couleur repésentative dans les méthodes spatiales
- L ..... taille moyenne des listes dans la méthode d'inversion de table de couleur d'Heckbert
- $\mathcal{ME}_n$ ..... ensemble des multi-ensembles de dimension n
- $M_i(\boldsymbol{C})$ ..... Moment d'ordre i du multi-ensemble $\boldsymbol{C}$
- M ..... nombre de couleurs de l'image
- $\mathcal{N}$ ..... cardinal de l'image
- N ..... cardinal d'une partition initiale du cube RGB
- $\mathcal{P}$ ..... partition
- $\boldsymbol{Q}$ ..... fonction d'inversion de table de couleurs
- $R_2(C)$ ..... somme des matrices 3×3 $\mathbf{c}.\mathbf{c}^t$ pour tout $\mathbf{c} \in \boldsymbol{C}$.
- $\boldsymbol{SE}(\boldsymbol{C})$ ..... erreur quadratique du multi-ensemble $\boldsymbol{C}$
- T ..... matrice codant la table de couleur dans les méthodes spatiales
- V_I ..... diagramme de Voronoï caculé à partir de l'image I dans notre méthode d'inversion de table de couleurs
- var_i ..... variance calculée sur l'axe de coordonné i
- v_i ..... $i^{\text{ème}}$ valeur propre de la matrice de covariance
- $\Delta_{k,k'}$ ..... augmentation de l'erreur de partition induite par la fusion des multi-ensembles d'indice k et k'
- $\delta(t)$ ..... fraction parcourue du cardinal d'un multi-ensemble lorsque le plan de coupe est en position t (méthodes descendantes)

- μ ..... moyenne d'un multi-ensemble.

Chapitre 4

Représentations hiérarchiques

4.1 Introduction

Nous avons présenté dans les chapitres 2 et 3 différentes méthodes permettant d'interpréter ou de modifier la valeur des pixels d'une image. Toutefois, dans ces deux chapitres, nos traitements ont été effectués individuellement sur chaque pixel ou sur les différentes valeurs d'un pixel considérées dans une séquence d'image.

De tels traitements, souvent qualifiés de bas niveaux, ne peuvent permettre une bonne interprétation du contenu d'images complexes. En effet, l'interprétation d'une image nécessite de décomposer celle-ci en différents groupes de pixels correspondant aux différents objets de la scène.

L'extraction des différents objets contenus dans une image passe généralement par une étape intermédiaire appelée l'étape de segmentation (définition 1). Cette étape permet de regrouper en une seule entité, appelée région, un ensemble connexe de pixels vérifiant un critère d'homogénéité P.

Étant donné un prédicat P, un algorithme de segmentation ne fournit généralement pas une région pour chaque objet composant la scène. De fait, chaque région représentera généralement une partie d'un objet et différentes régions doivent être fusionnées en fonction de critères de haut niveau pour former les objets de l'image. Par exemple, dans le cas d'une segmentation en matériaux (section 2.4), l'algorithme de segmentation fournit une région pour chacun des matériaux composant un objet. Ces régions doivent être fusionnées afin d'obtenir une région pour chaque objet. Enfin, en dehors d'applications très précises, la définition d'un objet est souvent assez imprécise. Par exemple, pour une image représentant une table, doit on considérer que chaque pied de la table constitue un objet ou que l'ensemble de la table doit être stocké comme un seul objet. La réponse à cette question dépend bien évidement du niveau de détail que l'on désire obtenir.

Les représentations hiérarchiques ou *pyramides* permettent de résoudre élégamment les problèmes précédents en calculant une hiérarchie de partitions. La partition initiale appelée la base de la pyramide permet de stocker individuellement chaque pixel. Les niveaux suivants permettent d'agréger les pixels en régions puis les régions entre elles. Ainsi les premiers niveaux de la pyramide basés uniquement sur des critères d'homogénéité bas niveaux permettent de décomposer une image en régions homogènes alors que des fusions se situant plus haut dans la pyramide peuvent assembler les régions en objets. L'aspect hiérarchique des pyramides permet donc de stocker le fait qu'un

objet codant une table à un certain niveau de la pyramide est issu de la fusion de plusieurs régions codant les différents éléments de celle ci (chapitre 1).

Nous allons décrire dans les sections 4.2 à 4.5 les différents types de Pyramides usuellement utilisées en traitement d'images. Nous présenterons ensuite dans la section 4.6 un nouveau type de représentation hiérarchique appelé *Pyramides Combinatoires* qui constitue ma contribution à ce domaine de recherche.

4.2 Pyramides régulières

Une pyramide régulière est définie comme une séquence d'images dont la taille décroît exponentiellement [183, 45, 115, 168, 16, 120, 163, 49, 153, 53, 119, 137, 166]. Chaque image de cette série est appelé un *niveau* de la pyramide. Le premier niveau correspond à l'image originale tandis que le niveau le plus élevé est généralement composé d'un seul pixel dont la valeur est une moyenne pondérée des pixels de l'image. Si nous utilisons les relations de voisinage définies sur l'image, la *fenêtre de réduction* d'un pixel de la pyramide relie celui-ci à un ensemble de pixels dans l'image de niveau précédent. L'ensemble des pixels d'une fenêtre de réduction sont les *enfants* ou les fils du pixel définissant celle-ci. Inversement, un tel pixel est le *père* (cercles pleins dans la figure 4.1) des pixels appartenant à la fenêtre de réduction. Cette relation père-enfant peut être étendue par transitivité à deux niveaux quelconques de la pyramide. L'ensemble des enfants d'un pixel de la pyramide dans l'image de base est appelé son *champ récepteur*. Lorsque la pyramide régulière est utilisée pour calculer une valeur ou un ensemble de valeurs, celles-ci sont stockées à un niveau particulier de la pyramide. Dans ce cas, la taille et le cardinal des fenêtres de réduction est identique pour tous les pixels de la pyramide.

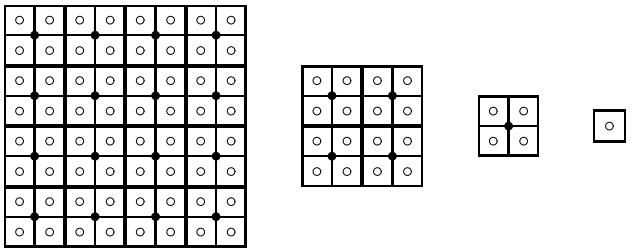


FIG. 4.1 – Une pyramide régulière $2 \times 2/4$

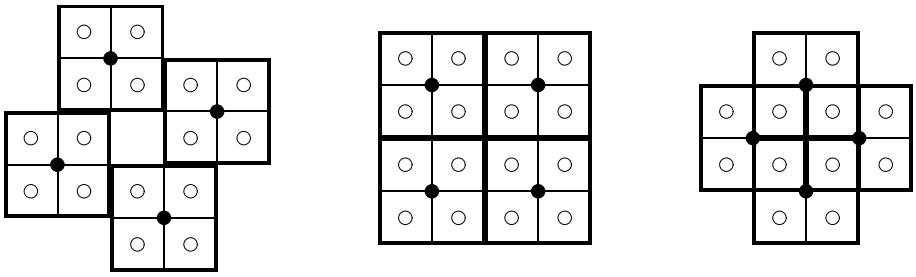
Un autre paramètre des pyramides régulières est le *facteur de réduction* qui code le rapport entre la taille de deux images successives. Ce rapport reste constant entre deux niveaux consécutifs de la pyramide. Une *Fonction de réduction* calcule le contenu d'un père à partir de celui de ses fils contenus dans la fenêtre de réduction (figure 4.2). Une pyramide régulière peut donc se définir formellement par le rapport $N \times N/r$ où $N \times N$ représente la taille de la fenêtre de réduction tandis que r représente le facteur de réduction. Différents types de pyramides peuvent être distingués en fonction du rapport $N \times N/r$:

- si $N \times N/r < 1$, la pyramide est appelée une *Pyramide trouée et non recouvrante*. Dans le cadre de ces pyramides, certains pixels ne possèdent pas de père [122] (voir par exemple le pixel central sur la figure 4.3(a)) ;

(a) 256^2 (b) 128^2 (c) 64^2 (d) 32^2 (e) 16^2 (f) 8^2

FIG. 4.2 – Une $2 \times 2/4$ pyramide. La fonction de réduction est une gaussiènne centrée sur le pixel survivant. La taille de chaque image est indiquée en dessous de celle-ci.

- si $N \times N/r = 1$, la pyramide est appelée une *pyramide non recouvrante non trouée* (voir par exemple la figure 4.3(b)) ;
- si $N \times N/r > 1$, la pyramide est appelée une *pyramide recouvrante*. Chaque pixel d'une telle pyramide a plusieurs *pères potentiels* [46]. Si chaque enfant sélectionne un père parmi ces pères potentiels, l'ensemble des enfants de la fenêtre de réduction de chaque pixel est réorganisé. Le champ récepteur d'un pixel peut dans ce cas prendre n'importe qu'elle forme incluse dans le champ récepteur original.



a) $2 \times 2/5$ b) $2 \times 2/4$ c) $2 \times 2/2$

FIG. 4.3 – Trois types de pyramides régulières

Les pyramides régulières présentent plusieurs propriétés intéressantes énumérées par Bister [16] :

1. Une réduction de l'influence du bruit par suppression des détails dans les versions réduites de l'image.
2. L'utilisation d'algorithmes indépendants de la résolution des régions recherchées.
3. La conversion de propriétés globales en propriétés locales.
4. La réduction des temps de calculs par utilisation du principe diviser pour conquérir
5. La détermination de certaines régions d'une image à faible coût en utilisant des images de faible résolution.

Malgré ces propriétés intéressantes, les pyramides régulières présentent plusieurs limitations. De fait, la taille limitée et la forme fixe des fenêtres de réduction alliée à la valeur constante du facteur de réduction ne permet pas aux pyramides régulières de s'adapter facilement à la variabilité des données. Cette rigidité induit plusieurs problèmes tels que la non stabilité de la structure des pyramides lors de légers décalages de l'image, le nombre limité de régions qui peuvent être codées à un certain niveau de la pyramide et l'incapacité à coder des régions occupant beaucoup plus de pixels suivant une des directions de l'image que l'autre. De telles régions sont appelées des régions allongées [16].

(a) (b)

FIG. 4.4 – Niveaux de gris du niveau de base et du niveau 1 sur une image 4×4 (a) et segmentation de l'image en fonction des valeurs de pixels de niveau 1(b)

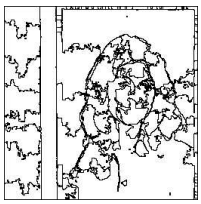
Cette dernière limitation est illustrée sur la figure 4.4 tirée de l'article de Bister [16]. Les pixels représentant la base de cette pyramide sont représentés par des carrés de niveaux de gris différents tandis que le niveau 1 est représenté par les cercles centrés sur leurs fenêtre de réduction. Notons que le niveau de gris de chacun des pères est choisi idéalement de façon à coder les 4 régions présentes à la base de la pyramide. Si la pyramide est une pyramide recouvrante $4 \times 4/4$, le pixel $(3, 0)$ (en bas à gauche sur la figure 4.4) n'appartient pas à la fenêtre de réduction du pixel blanc de niveau 1 (de coordonnée $(0.5, 0.5)$ dans notre repère). Ce pixel ne pourra donc être rattaché qu'au pixel gris $(2.5, 0.5)$ de niveau 1. La région blanche est donc artificiellement amputée d'un pixel du fait de la taille réduite des fenêtres de réduction. Le même phénomène peut s'observer pour les pixels $(0, 1), (0, 3)$ et $(3, 2)$ (Fig. 4.4).

L'effet de cette troncature des régions allongées est illustré sur la figure 4.5 où l'image test fille a été segmentée à l'aide de l'algorithme de Brister [16] (voir Annexe, section 6.1) en utilisant une Pyramide $5 \times 5/4$. Plusieurs régions allongées telles que la barre horizontale à gauche de l'image ainsi que les cheveux ont été artificiellement découpées en plusieurs petites régions.

Notons toutefois que les limitations des pyramides régulières peuvent devenir des avantages dans les applications où les objets à détecter sont compacts. En effet, la difficulté des pyramides régulières à coder des régions allongées, peut devenir un avantage si les objets à extraire doivent nécessairement être compacts. Un exemple d'application de cette propriété des pyramides régulières nous est fourni par Rosenfeld [166]. Celui-ci utilise les pyramides régulières pour détecter des chars d'assaut sur des images infrarouges. De par les conditions d'acquisition nous sommes certains que les chars apparaissent comme des objets compacts sur l'image et peuvent être réduits à un seul pixel à un certain niveau de la pyramide.

Les transformation par ondelettes discrètes [191, 139] ou par représentation échelle/signal (scale-space transform) [195] constituent un autre courant des représentation hiérarchiques. À l'intérieur

(a) Image originale



(b) Image segmentée

FIG. 4.5 – Segmentation de l'image test fille, à l'aide de l'algorithme de Bister [16] utilisant une pyramide recouvrante $5 \times 5/4$.

de ce courant, la description hiérarchique d'une image est obtenue par l'application successive d'une séquence de filtres ψ_j. Chaque image de niveau j est définie par un sous - échantillonnage de l'image de niveau $j-1$. Notons que de ce point de vue les pyramides régulières peuvent être considérées comme un cas particulier de représentation échelle/signal. De fait, dans le cadre des pyramides régulières chaque image est définie par un sous - échantillonage de l'image précédente et la valeur de chaque pixel est habituellement calculée en appliquant un filtre sur la fenêtre de réduction de chaque pixel. Étant donnée une tache particulière en traitement d'images, les applications basées sur une représentation échelle/signal ou sur une ondelette discrète définissent un filtre adapté à la tache à accomplir. Des algorithmes de traitement d'images sont alors utilisés sur cette représentation multi- échelle de l'image. Des applications en compression d'image, réduction de bruit, détection d'arêtes, flot optique et stéréo-vision montrent le fort potentiel de ce courant des représentations hiérarchiques [139]. La principale différence entre les pyramides régulières et les représentations échelle/signal se situe dans l'utilisation des filtres. En effet, dans le cadre des pyramides régulières le processus de décimation peut être adapté durant la construction des pyramides. Par exemple, dans le cadre de la segmentation, on peut appliquer des critères bas niveaux lors des premières opérations de réduction de façon à obtenir des région homogènes. Des critères de plus haut niveaux peuvent ensuite être utilisés en fonction de connaissances *a priori* sur la scène. Dans le cadre des transformation par ondelette discrètes ou par représentation échelle/signal, les filtres initiaux sont choisis en fonction de l'application mais ne sont usuellement pas modifiés durant la construction de la représentation hiérarchique.

4.3 Pyramides de graphes simples

Les pyramides irrégulières permettent de s'affranchir des limitations des pyramides régulières mentionnées dans la section 4.2 tout en préservant leurs principaux avantages. Ces pyramides sont définies comme une pile de graphes $(G_1, \ldots, G_n)$ successivement réduits (figure 4.6). Chaque graphe est construit à partir du graphe précédent en sélectionnant un ensemble de sommets appelés *sommets survivants* (section 4.3.1). Chacun des sommets restant est qualifié de sommet non survivant et est affecté à un survivant [146]. Chaque sommet non survivant est donc l'*enfant* d'un sommet

survivant qui représente l'ensemble des sommets non survivants qui lui sont affectés et dont il est le *père* (section 4.3.2). Si le graphe initial $G_0 = (V_0, E_0)$ est défini à partir d'une grille régulière, on peut associer un pixel à chaque sommet de V_0 et coder par une arête de E_0 chaque adjacence entre pixels (voir par exemple G_0 dans la figure 4.6). Les ensembles V_0 et E_0 sont respectivement appelés l'ensemble des sommets et des arêtes du graphe.

L'opération de réduction d'un graphe a été introduite pour la première fois par Meer [142] comme un processus stochastique. Dans ce cadre, un graphe G_{l+1} défini au niveau $l+1$ se déduit du graphe de niveau l par les étapes suivantes :

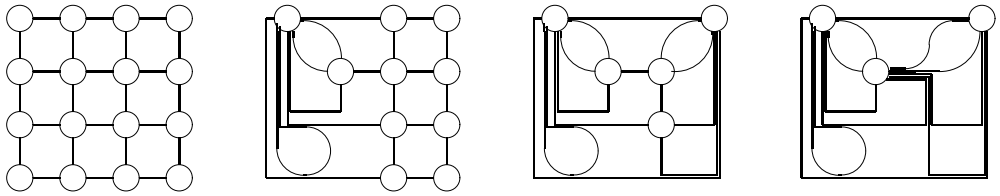


FIG. 4.6 – Une pyramide irrégulière basée sur une grille 4-connexe

1. Sélection des sommets de G_{l+1} parmi ceux de V_l. Ces sommets sont les sommets survivants du processus de décimation.
2. Établissement d'une liaison entre chaque sommet non survivant et un sommet survivant. Cette dernière étape établie une partition de V_l.
3. Définition des relations d'adjacence entre les sommets de G_{l+1} de façon à définir E_{l+1}.

4.3.1 Sélection des sommets survivants

Afin d'obtenir un facteur de décimation fixe entre deux niveaux de la pyramide, Meer [142] impose les contraintes suivantes sur l'ensemble des sommets survivants :

1. Stabilité externe :
$$\forall v \in V_l - V_{l+1} \; \exists v' \in V_{l+1} : \; (v, v') \in E_l. \tag{4.1}$$
2. Stabilité interne :
$$\forall (v, v') \in V_{l+1}^2 : \; (v, v') \notin E_l. \tag{4.2}$$

La contrainte (4.1) impose à chaque sommet non survivant d'être adjacent à au moins un sommet survivant. La contrainte (4.2) interdit à deux sommets adjacents de survivre simultanément. Le sous ensemble des sommets d'un graphe qui possède conjointement ces deux propriétés de stabilité est appelé un *noyau*. [167]

Meer [142] définit un noyau sur l'ensemble des sommets du graphe courant à l'aide d'un processus stochastique. Un des avantages de ce procédé est de permettre un codage aisé en parallèle. Un tirage d'une variable aléatoire uniformément distribuée sur $[0, 1]$ est associé à chaque sommet. Tout sommet correspondant à un maximum local de la variable aléatoire est alors considéré comme

un sommet survivant. Une itération de ce processus permet de vérifier la condition de stabilité interne (condition 4.2) puisque deux sommets adjacents ne peuvent correspondre simultanément à un maximum local de variable aléatoire. Toutefois, un sommet ne correspondant pas à un maximum local peut n'avoir dans son voisinage que des sommets non survivants. Une telle configuration est illustrée sur la figure 4.7 où les valeurs à l'intérieur de chaque sommet représentent la variable aléatoire. Les sommets de valeurs 9 représentent des maxima locaux et sont donc sélectionnés comme sommets survivants. Les sommets 7 et 8 adjacents à des sommets survivants ne peuvent survivre (condition 4.2). Le sommet 6 ne représente pas un maxima local, toutefois il doit être marqué comme survivant afin de respecter la condition 4.1.

FIG. 4.7 – Décimation d'un graphe "1D". Les sommets survivants sont représentés par un cercle double.

Il convient donc d'itérer le processus précédent jusqu'à ce que les deux conditions soient vérifiées. On attache pour cela trois variables x_i, p_i et q_i à chaque sommet $v_i \in V_l$ du graphe G_l. La variable x_i code le tirage de la variable aléatoire pour le sommet v_i. Les variables p_i et q_i sont deux booléens dont la valeur code les états suivants :

– si p_i est vrai, le sommet v_i est considéré comme un sommet survivant ;
– si q_i est vrai, v_i peut devenir un sommet survivant lors des itérations futures.

Soient $(p_i^{(k)})_{k\in\{1,\ldots,n\}}$ et $(q_i^{(k)})_{k\in\{1,\ldots,n\}}$ les différentes valeurs de p_i et q_i au cours des n itérations de l'algorithme. L'initialisation des variables p_i et q_i est réalisée par les affectations suivantes :

$$\begin{array}{lcl} p_i^{(1)} & = & x_i = max_{j\in V(v_i)}\{x_j\} \\ q_i^{(1)} & = & \bigwedge_{j\in V(v_i)} \overline{p_j}^{(1)} \end{array} \tag{4.3}$$

où $V(v_i)$ représente l'ensemble des sommets adjacents à v_i et $\bigwedge$ l'opérateur logique "et". Par convention, le sommet v_i appartient à $V(v_i)$.

En d'autres termes un sommet est considéré comme survivant ($p_i = vrai$) s'il contient un maximum local de la variable aléatoire. Un sommet peut devenir un sommet survivant ($q_i^{(1)} = vrai$) s'il ne l'est pas déjà ($\overline{p_i}^{(1)} = vrai$) et si aucun de ses voisins ne survit.

La mise à jour itérative des prédicats p_i et q_i est réalisée à l'aide des règles suivantes :

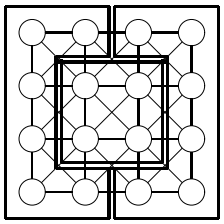
$$\begin{array}{lcl} p_i^{(k+1)} & = & p_i^{(k)} \vee \ (q_i^{(k)} \wedge \ x_i = max_{j\in V(v_i)}\{q_j^{(k)} x_j\}) \\ q_i^{(k+1)} & = & \bigwedge_{j\in V(v_i)} \overline{p_j}^{(k+1)} \end{array} \tag{4.4}$$

où le symbole $\vee$ représente le "ou" logique.

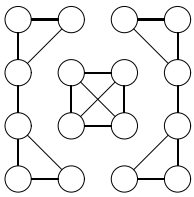
Un sommet survivant à l'étape k restera donc survivant aux étapes suivantes. De plus, un sommet non survivant peut devenir survivant s'il est un maximum local des sommets restants (tels que $q_j^{(k)} = 1$). Enfin, un sommet reste un candidat à la survie, s'il n'est pas un sommet survivant et si aucun sommet ne survit dans son voisinage. Ce processus est itéré jusqu'à ce que l'ensemble

des sommets sélectionnés forme un noyau. Notons que ce processus est entièrement local. A chaque itération, un sommet calcule son état en fonction de son état précédent et de celui de ses voisins. Un tel processus peut donc facilement être implémenté sur une machine parallèle.

Ce processus purement aléatoire a été adapté par Montanvert et al. [146] à l'analyse de composantes connexes. Montanvert et al. restreignent le processus de décimation à un ensemble de sous - graphes du graphe initial. Notons que cette restriction est équivalente à une application du processus de décimation indépendamment sur chaque sous - graphe. Deux sommets survivants peuvent donc être adjacents dans le graphe à décimer s'ils n'appartiennent pas au même sous graphe. La définition d'un sous graphe est équivalente à la définition d'une fonction λ de E_l dans $\{0, 1\}$. Pour chaque $(v, v') \in E_l$, $\lambda((v, v'))$ est positionné à 1 si v et v' appartiennent au même sous graphe et à 0 sinon. Dans le cadre de l'analyse de composantes connexes, la fonction λ est définie en positionnant $\lambda((v, v'))$ à 1 si v et v' appartiennent à la même composante connexe. La figure 4.8 illustre une application de la fonction λ pour l'analyse d'une image 4×4 composée de 3 composantes connexes. Chaque composante de la figure 4.8 est délimitée par une courbe fermée.



a) Grille 4−connexe



b) Sous graphes définis par la fonction λ

FIG. 4.8 – Les composantes connexes définies par la fonction λ.

Dans le cadre d'un codage des composantes connexes d'une image quantifiée (chapitre 3.2), $\lambda(v, v')$ sera positionné à 1 si les pixels associés à v et v' ont la même couleur représentative et à 0 sinon. Toutefois, dans le cadre d'une application à la segmentation, la fonction λ doit être redéfinie à chaque niveau de façon à prendre en compte la plus grande complexité des données. Jolion [114] a amélioré la flexibilité du processus de décimation en utilisant les maxima locaux d'un opérateur d'intérêt plutôt que les maxima locaux des tirages de la variable aléatoire. Par exemple, dans le cadre de la segmentation, Jolion définit l'opérateur d'intérêt comme une fonction décroissante de la variance des niveaux de gris calculée sur un voisinage de chaque sommet du graphe. Ce critère tend donc à sélectionner les sommets survivants situés dans des régions homogènes. Diverses applications ont montrées l'utilité de ce procédé pour la segmentation en régions homogènes [112, 10, 50].

4.3.2 Définition du lien parent - enfant

Étant donné l'ensemble des sommets survivants, la contrainte de stabilité externe (équation 4.1) nous assure que chaque sommet non survivant est adjacent à au moins un sommet survivant. Si nous utilisons le processus stochastique de Meer [142] et Montanvert [146], chaque sommet du graphe est attaché au tirage d'une variable aléatoire. Meer [142] et Montanvert [146] attachent donc chaque sommet non survivant au sommet adjacent survivant dont le tirage de la variable aléatoire est maximum. Jolion [114] utilise une fonction de contraste telle que la différence des

niveaux de gris afin d'attacher chaque sommet non survivant au sommet survivant qui contraste le moins avec celui-ci. Un sommet non survivant djacent à un seul sommet survivant est attaché à celui-ci. Le sommet survivant réalise dans ce cas le minimum de la variable aléatoire ou de la fonction de contraste.

Chaque sommet survivant est donc le père des sommets non survivants attachés à celui-ci. L'ensemble des enfants d'un sommet survivant est appelé sa fenêtre de réduction par référence aux notations utilisées dans les pyramides régulières. La figure 4.9 représente les tirages de la variable aléatoire sur un graphe défini à partir d'une grille 4×4 8-connexe. Les sommets survivants induits par ce tirage sont entourés d'un cercle supplémentaire tandis que les frontières de chaque fenêtre de réduction sont superposées à la figure.

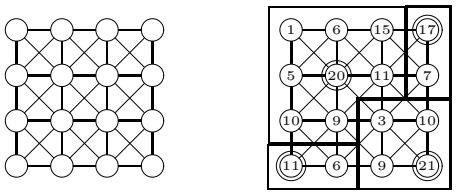


FIG. 4.9 – Construction d'un noyau en utilisant les tirages d'une variable aléatoire. La valeur de chaque tirage est représentée à l'intérieur de chaque sommet.

4.3.3 Connexion des sommets survivants

La dernière étape du processus de décimation consiste à connecter les sommets survivants dans le graphe G_{l+1}. Meer [142] connecte deux pères par une arête si leurs fenêtres de réductions dans le graphe G_l sont adjacentes par au moins un sommet. Sur l'exemple de la figure 4.10, les sommets survivants associés aux valeurs 20 et 17 seront adjacents en raison de :

- l'adjacence entre le sommet 15 (attaché à 20) et le sommet survivant 17 ;
- l'adjacence entre le sommet 11 (attaché à 20) et le sommet survivant 17 ;
- l'adjacence entre le sommet 11 et le sommet non survivant 7 (attaché à 17).

Notons que n'importe laquelle de ces trois conditions suffirait pour établir la connexion entre les deux sommets survivants 20 et 17.

Puisque chaque enfant est adjacent à son père, deux sommets adjacents dans G_{l+1} sont connectés dans G_l par un chemin de longueur inférieure à 3 (figure 4.9). De tels chemins sont appelés des *chemins de connexion*. De par la contrainte de stabilité interne (équation 4.2), deux pères ne peuvent être adjacents dans G_l. La longueur des chemins de connexion ne peut donc être égale à 1 et doit être égale à 2 ou 3. Notons toutefois, que si nous restreignons le processus de décimation à des sous graphes [146, 114] définis par la fonction λ, deux sommets survivants v et v' peuvent être adjacents par une arête vérifiant $\lambda((v, v)) = 0$.

4.3.4 Définition du sommet d'une pyramide

Une pyramide irrégulière est donc construite récursivement à partir de G_0 (la grille initiale) jusqu'au sommet G_t. Le sommet de la pyramide est défini comme un graphe qui ne peut plus être

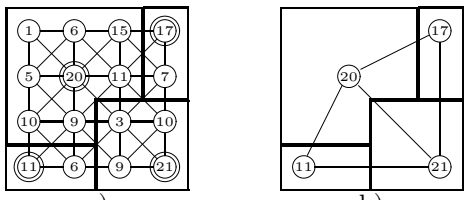


FIG. 4.10 – Le graphe réduit (b) déduit du noyau (a).

réduit par le processus de décimation. Ceci se produit lorsque G_t est réduit à un seul sommet ou lorsqu'aucune règle de décimation ne peut être appliquée. Par exemple, dans le cadre de l'analyse de composantes connexes, les graphes codant la pyramide sont réduits jusqu'à ce que chaque sous graphe codant une composante connexe soit réduit à un seul sommet [146].

Dans le cadre de la segmentation, le processus de décimation peut également être stoppé quand la différence des valeurs entre les sommets adjacents dépasse un certain seuil. Une autre stratégie, similaire à celle employée dans le cadre des pyramides régulières [46] consiste à réduire la pyramide à un seul sommet puis à sélectionner un ensemble de *racines* à différents niveaux de la pyramide. Jolion [114] définit une racine comme un sommet dont le contraste avec son père dépasse un certain seuil. Intuitivement, cette stratégie également utilisée dans les pyramides régulières revient à refuser la fusion d'un ensemble de régions lorsque celles-ci ne définissent pas un tout homogène. La non homogénéité de la région étant ici définie en fonction du seuil. La valeur associée à un sommet peut être générée à partir d'un ensemble quelconque d'attributs associés au sommet. On peut par exemple, stocker dans chaque sommet le nombre de ces enfants dans le graphe initial et la somme des vecteurs couleurs de ceux-ci. La couleur moyenne de la région associée à un sommet de la pyramide est alors calculée en divisant les deux attributs. La fonction de contraste peut être simplement définie comme la norme de la différence des deux vecteurs couleur.

4.4 Récentes améliorations du procesus de construction d'une pyramide

Comme nous l'avons mentionné dans la section 4.3.1, les principaux avantages de la méthode de décimation définie par Meer [142] et Montanvert [146] sont :

1. Le processus de décimation est purement local donc aisément parallélisable.
2. L'utilisation de noyaux permet d'obtenir un facteur de décimation approximativement constant entre chaque niveau.

Jolion [113] et Haxhimusa [90] ont récemment proposé deux alternatives à ce processus de décimation.

4.4.1 Un processus de décimation guidé par les données

Comme nous l'avons vu dans la section 4.3.1, une fois qu'un sommet est désigné sommet survivant, il reste dans cet état jusqu'à la fin des itérations (équation 4.4). De plus, les sommets adjacents à un sommet survivant ne peuvent devenir survivants du fait de la contrainte de stabilité

interne (équation 4.2). Un noeud désigné comme survivant dans les premières itérations et ses voisins non survivants pourraient donc être immédiatement réduits en un seul sommet. Toutefois, l'algorithme de décimation stochastique créant une structure de noyau, la réduction de ce sommet ne pourra se faire que lorsque le dernier sommet survivant aura été calculé. Pour des graphes importants ce temps de latence proportionnel au nombre d'itérations peut être conséquent.

L'idée de base de la méthode proposée par Jolion [113] consiste à n'effectuer qu'une seule itération du processus de décimation à chaque niveau de la pyramide. Les deux variables booléennes p_i et q_i attachées à chaque sommet v_i et définies dans la section 4.3.1 conservent la même signification mais sont à présent globales à l'ensemble de la pyramide. Ces variables sont initialisées à vrai au niveau de base de la pyramide. On a donc :

$$p_i^{(0)} = q_i^{(0)} = vrai, \quad \forall v_i \in V_0$$

où $G_0 = (V_0, E_0)$ représente le graphe de base de la pyramide.

La mise à jour des variables p_i et q_i entre chaque niveau est réalisée par les équations suivantes :

$$\begin{aligned} p_i^{(k+1)} &= \left((p_i^{(k)} \vee q_i^{(k)}) \wedge x_i = max_{j \in V_k(v_i)} \{ q_j^{(k)} x_j \} \right) \\ q_i^{(k+1)} &= \bigwedge_{j \in V_k(v_i)} \overline{p_j}^{(k+1)} \wedge (V_k(v_i) \neq \{v_i\}). \end{aligned}$$

En d'autres termes, un sommet est déclaré survivant s'il était survivant ou candidat au niveau précédent et s'il réalise un maximum local de la variable aléatoire dans le graphe G_k. Un sommet est déclaré candidat s'il n'est pas adjacent à un sommet survivant et s'il n'est pas isolé.

L'ensemble des sommets survivants ainsi sélectionnés vérifient la propriété de stabilité interne (équation 4.2) mais pas la propriété de stabilité externe (équation 4.1). Plus précisément, les variables $p_i^{(k+1)}$ et $q_i^{(k+1)}$ nous permettent de décomposer l'ensemble des sommets du graphe en trois classes : les sommets survivants, les sommets candidats et les sommets non candidats. Les sommets non candidats sont nécessairement adjacents à un sommet survivant et doivent donc être réduits dans le graphe suivant. Inversement, les sommets survivants doivent nécessairement faire partie de l'ensemble des sommets du graphe réduit. L'ensemble des sommets candidats représente l'ensemble des sommets pour lesquels des itérations supplémentaires auraient été nécessaires pour les classer parmi les survivants ou les non survivants. Cette décision est prise à des niveaux plus élevés dans la pyramide. L'ensemble des sommets de niveau $k+1$ est donc l'ensemble des sommets survivants et candidats :

$$N_{k+1} = \{ v_i \in N_k \mid p_i^{(k+1)} \vee q_i^{(k+1)} \}.$$

L'ensemble E_{k+1} des arêtes de niveau $k+1$ se déduit par le même processus que celui décrit dans la section 4.3.3. Notons toutefois qu'il convient ici de tenir compte également des sommets directement adjacents.

La principale motivation de ce processus de décimation asynchrone est de favoriser la réduction des régions de l'image particulièrement homogènes ou présentant un intérêt particulier vis-à-vis du critère de décimation. En effet, du fait du processus de décimation asynchrone, de telles régions se forment à des niveaux moins élevés que les zones de l'images ou les données ne permettent pas de décider facilement quels sont les sommets survivants. Ces régions sont donc détectées plus tôt.

Ce mécanisme de réduction devrait également permettre de simuler des résultats d'expériences psycho-visuelles [178] qui tendraient à montrer que l'activation d'un neurone dépend non seulement de l'intensité de ses entrées mais également de l'ordre dans lequel ses entrées sont activées. L'information est donc véhiculée de façon asynchrone dans le cerveau.

4.4.2 Décimation par construction d'un couplage maximal

La méthode de décimation stochastique définie par Meer [142] et Montanvert [146] (section 4.3.1) repose sur la définition d'un noyau. Du fait de la contrainte de stabilité interne (équation 4.2), le nombre de sommets survivants est fortement dépendant du nombre et de la répartition des arêtes du graphe. De plus, un sommet est élu survivant s'il réalise le maximum des tirages d'une variable aléatoire sur son voisinage. La probabilité *a priori* qu'un sommet survive est donc relative à la taille de son voisinage. Les relations d'adjacence entre les sommets ont donc une influence importante sur :

1. La hauteur de la pyramide
2. Le nombre d'itérations nécessaires pour construire chaque niveau à partir du précédent.

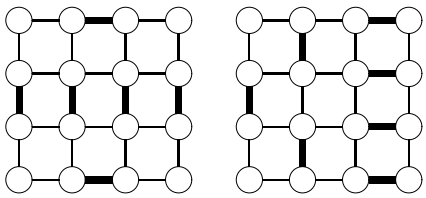
Des expériences réalisées par Haxhimusa et al. [89] ont montré que le degré moyen des sommets tend à augmenter dans la pyramide. Ce phénomène induit une décroissance du facteur de décimation et augmente par conséquent la taille de la pyramide.

Ce phénomène peut être extrêmement gênant puisqu'une des propriétés des pyramides est d'avoir une hauteur sensiblement égale au log de la taille du graphe initial. Afin de corriger cette limitation, Haxhimusa propose de baser le processus de décimation sur la définition d'un *couplage maximal* [167] dans le graphe à décimer (Définition 3 et figure 4.11).

Définition 3 *Étant donné un graphe $G = (V, E)$, un ensemble $C \subset E$ est appelé un couplage si aucune des arêtes de C n'est adjacente au même sommet.*

- *un couplage est dit maximal si on ne peut y ajouter d'arête sans perdre la propriété de couplage ;*
- *un couplage est dit maximum s'il comporte un nombre maximum d'arêtes ;*
- *un couplage est dit parfait si tout sommet est incident à une arête de C. Un couplage parfait est maximum.*

Un sommet incident aux arêtes d'un couplage est dit saturé sinon il est dit insaturé.



(a) 6 arêtes (b) 7 arêtes

FIG. 4.11 – Deux couplages maximaux comportant un nombre différent d'arêtes. Les arêtes faisant partie du couplage sont représentées en gras.

Notons que la définition d'un couplage maximal sur un graphe $G = (V, E)$ est équivalente à la définition d'un noyau sur le graphe $G' = (E, E')$ où deux éléments de E sont connectés par une arête de E' s'ils sont adjacents à un même sommet.

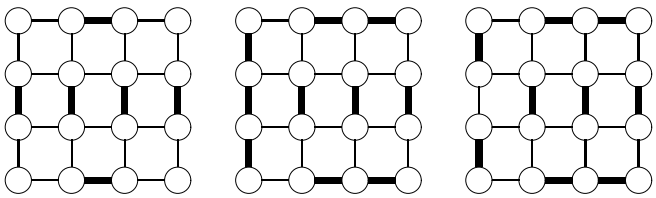
Le but de la méthode d'Haxhimusa et al. [89] est de définir une forêt K du graphe initial G telle que :

- chaque sommet de G appartient à exactement un arbre de la forêt ;
- chaque arbre est composé d'au moins deux sommets.

La construction de cette forêt est réalisée à l'aide des étapes suivantes (figure 4.12) :

1. construire un couplage maximal C sur G ;
2. Supprimer les sommets insaturés en ajoutant de nouvelles arêtes. Le nouvel ensemble noté C^+ ne constitue plus un couplage. En revanche si le couplage est maximum, l'ensemble C^+ forme un recouvrement minimum de G [12] ;
3. supprimer des arêtes dans C^+ de façon à construire une forêt K composée d'arbres de profondeur 1.

La construction d'un couplage maximal C ne conduit généralement pas à la création de couplage parfait (Définition 3). Soit v un sommet insaturé et $e = (v, w)$ avec $v \neq w$ une arête incidente à v. Le sommet w est nécessairement incident à une arête du couplage de par sa propriété de maximalité. Soit C^+ l'ensemble C auquel nous avons ajouté une telle arête pour chaque sommet insaturé. Par construction, tout sommet du graphe est incident à C^+ et les composantes connexes de celui-ci sont des arbres de profondeur 1 ou 2. Les arbres de profondeur 2 peuvent être découpés en deux arbres de profondeur 1 en supprimant de l'ensemble C^+ les arêtes dont les deux sommets sont incidents à deux arêtes de C^+.



(a) C (b) C^+ (c) K

FIG. 4.12 – Les trois étapes du processus de décimation de Haxhimusa [89]

L'ensemble K obtenu forme une forêt recouvrante du graphe G composée d'arbres de profondeur 1. On peut donc désigner pour chaque arbre une racine induisant une profondeur 1. Cette décomposition de chaque arbre en racine et feuilles induit une sélection naturelle de l'ensemble des sommets survivants et non survivants : pour chaque arbre, la racine est désignée comme sommet survivant tandis que les sommets restants sont les fils de celle-ci et sont donc classifiés comme non survivants. Étant donnée la sélection de l'ensemble des sommets survivants et de leurs enfants, la construction des nouvelles arêtes entre les sommets survivants peut s'effectuer de façon similaire à celle définie dans la section 4.3.3.

La principale amélioration de la méthode d'Haxhimusa est de permettre une meilleure stabilité du facteur de décimation entre les niveaux. Les expériences menées par celui-ci montrent que le facteur de décimation reste approximativement égal à 2 tout au long du processus de décimation.

4.5 Pyramides de graphes duaux

4.5.1 Introduction

Les pyramides stochastiques utilisées par Meer [142] et Montanvert [146] ou les pyramides adaptatives définies par Jolion [114] combinent les avantages des pyramides régulières et une plus grande capacité d'adaptation aux données de l'image. Toutefois les processus de décimation stochastiques et adaptatifs sont tous deux basés sur des *graphes simples*, c'est à dire sur des graphes sans *boucles* ni *arêtes multiples* entre les sommets. Si les graphes de la pyramide sont utilisés pour coder une pile de partitions, une arête entre deux sommets codera donc simplement l'existence d'au moins une frontière commune entre deux régions. L'utilisation de graphes simples oblige ainsi à coder un ensemble déconnecté de frontières entre les régions par une seule arête. De plus, l'absence de boucles ne permet pas de différencier une adjacence de régions d'une relation d'inclusion entre celles-ci.

La figure 4.13 illustre ces limitations. Les régions gauche et droite de l'image ont deux frontières communes. Toutefois, un processus de décimation tel que le processus stochastique nous fournira le graphe réduit représenté sur la figure 4.13(b) où les frontières entre les régions droite et gauche sont représentées par une seule arête.

(a) (b)

FIG. 4.13 – Limitation des processus de décimation adaptatifs et stochastiques

Cette dernière limitation est une conséquence de la définition des arêtes dans le graphe réduit. De fait, deux sommets survivants ayant plusieurs enfants adjacents dans G_l ne seront connectés que par une seule arête dans G_{l+1}. Une meilleure compréhension de cette limitation peut être obtenue en interprétant le processus de décimation en termes de *contraction* et *suppression* d'arêtes. La contraction d'une arête (v, v') consiste à identifier les sommets v et v' et à supprimer l'arête (v, v'). Notons que si v et v' sont connectés par une autre arête, celle-ci devient une boucle du nouveau sommet. Une contraction d'arête consistant à identifier deux sommets, nous considérons la contraction d'une boucle comme non définie. La suppression d'une arête revient simplement à enlever celle-ci de l'ensemble des arêtes du graphe.

Si nous utilisons les opérations de contraction et de suppression, les processus de décimation définis par Meer [142], Montanvert [146] et Jolion [114, 113] sont équivalents à l'application des étapes suivantes :

1. La sélection d'un ensemble S de sommets survivants.

2. La définition d'un ensemble N d'arêtes connectant chaque sommet non survivant à un sommet survivant.
3. La contraction de l'ensemble N d'arêtes.
4. La suppression de toutes les boucles ou arêtes multiples.

La construction des ensembles S et N varie en fonction du processus de décimation considéré. Toutefois, l'étape 4 des processus de décimation décrit dans les sections 4.3 et 4.4 supprime toutes les arêtes multiples et les boucles entre les sommets survivants.

4.5.2 Noyaux de contraction

Kropatsch [122] code l'étape 3 du processus de décimation à l'aide d'un *Noyau de Contraction*. Un tel noyau est défini sur un graphe $G = (V, E)$ par un ensemble de sommets survivants S et un ensemble d'arêtes non survivantes N (représentées par des lignes épaisses sur la figure 4.14) telles que :

– (V, N) est une forêt recouvrante de G ;
– la racine de chaque arbre de (V, N) appartient à S.

Notons qu'un noyau de contraction est une forêt du graphe initial. Un noyau de contraction et un noyau (section 4.3.1) sont donc deux concepts différents. La langue anglaise distingue plus nettement ces deux termes en désignant par *Maximal Independent Vertex Set* un noyau et par *Contraction Kernel* un noyau de contraction.

Puisque l'ensemble N des arêtes à contracter forme une forêt du graphe initial G, aucune boucle ne peut être contractée. L'opération de contraction est donc bien définie. Si le graphe G est déduit d'une grille discrète, chaque sommet appartenant à la bordure de la grille doit être adjacent à un sommet spécial codant l'extérieur de la grille. La région associée à ce sommet étant infinie, celui-ci est appelé le *sommet infini*. Si nous voulons préserver la bordure de l'image, toute arête codant une relation d'adjacence entre un sommet du graphe et le sommet infini doit être exclue de N. Un graphe codant une grille discrète peut donc être au plus réduit à un graphe G_t composé de deux sommets connectés par une seule arête. Si chaque sommet du graphe initial code un pixel de la grille discrète, les deux sommets de G_t codent l'ensemble de l'image et l'extérieur de celle-ci. L'arête entre ces deux sommets code la bordure de l'image.

La décimation d'un graphe utilisant les noyaux de contraction diffère des méthodes décrites dans les sections 4.3 et 4.4 sur les deux points suivants :

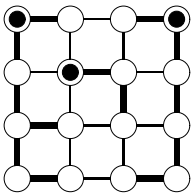
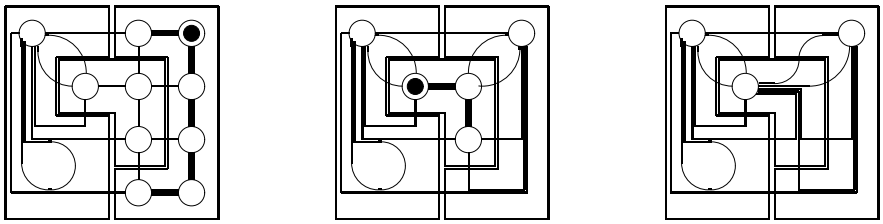


FIG. 4.14 – Un noyau de contraction (S, N) composé de trois arbres. Les sommets appartenant à S contiennent un cercle plein (•).

– si nous utilisons un noyau de contraction, l'ensemble des arêtes à contracter ne doit pas nécessairement former un noyau. Deux sommets survivants peuvent donc être adjacents dans le graphe contracté. Cette dernière propriété permet d'éviter l'utilisation de la fonction λ définie par Montanvert [146] et Jolion [114] (section 4.3.1, page 132) ;
– l'utilisation d'un noyau de contraction n'impose pas aux sommets non survivants d'être directement adjacents à leur père. La connection entre un sommet non survivant et son père est réalisée par une branche de l'arbre qui les contient (figure 4.14). L'ensemble des fils d'un sommet survivant est donc défini par l'arbre contenant ce sommet.



G_1 G_2 G_3

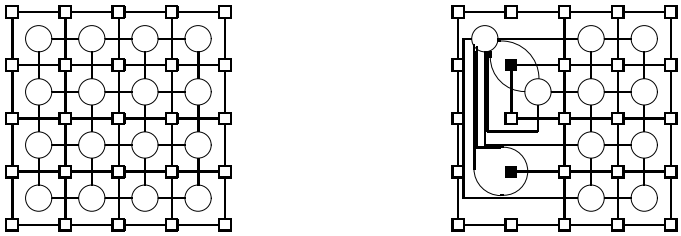
FIG. 4.15 – La contraction successive des arbres définis sur la figure 4.14

La figure 4.15 illustre la décimation induite par le noyau de contraction défini sur la figure 4.14. Notez que ces contractions sont effectuées séquentiellement uniquement dans un but pédagogique. La contraction en parallèle des arêtes du noyau de contraction induit des temps d'exécution bien inférieurs à ceux d'un processus séquentiel et doit donc être appliquée chaque fois que cela est possible.

La contraction de l'ensemble des arêtes d'un noyau de contraction réduit le nombre de sommets d'un graphe tout en préservant l'ensemble des arêtes entre sommets survivants. Cette survie de toutes les arêtes entre sommets survivants peut induire la création d'arêtes redondantes. La caractérisation de ces arêtes nécessite une meilleure description des relations topologiques entre les objets décrits par les sommets du graphe.

Les images peuvent souvent être comprises comme la projection d'une scène réelle sur un plan 2D. La structure d'une image est donc bidimensionnelle et de telles images peuvent être décrites par des graphes planaires. De plus, si le graphe initial se déduit d'une grille planaire (comme la grille 4-connexe), le graphe initial et ses différentes simplifications sont planaires. Étant donné un graphe planaire initial G, les sommets du graphe dual $\overline{G}$ sont positionnés à l'intérieur de chaque face de G. Les arêtes de $\overline{G}$ codent les adjacences de faces (arêtes connectant les carrés dans la figure 4.16(a)). Notons que la construction du graphe dual induit une correspondance 1 à 1 entre les arêtes des deux graphes. Si N représente un ensemble d'arêtes du graphe initial, nous pouvons donc noter par $\overline{N}$ l'ensemble des arêtes duales associées. Un résultat bien connu de la théorie des graphes [190] établit que si G' est obtenu à partir de G par la contraction des arêtes contenues dans N, le dual de G' peut être obtenu à partir du dual de G en supprimant dans $\overline{G}$ l'ensemble d'arêtes $\overline{N}$. De même si G' est obtenu à partir de G en supprimant un ensemble $N \subset E$ d'arêtes, $\overline{G'}$ peut être obtenu à partir de $\overline{G}$ en contractant l'ensemble d'arêtes $\overline{N}$.

La caractérisation des arêtes redondantes en utilisant le graphe dual est illustrée sur la figure 4.16. Si le graphe initial est déduit de la grille 4-connexe, chaque sommet du graphe dual peut



a) $(G_0, \overline{G_0})$ b) $(G_1, \overline{G_1})$

FIG. 4.16 – La grille planaire initiale avec le graphe associé(a) et la caractérisation des arêtes redondantes(b). Les sommets du graphe dual incidents à des arêtes duales redondantes sont représentés par des carrés pleins.

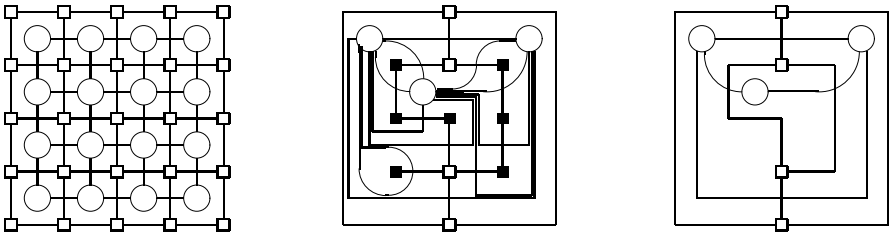
s'interpréter comme un coin de pixel. De même, chacune de ses arêtes peut s'interpréter comme un coté de pixel. Le carré plein en haut à gauche de la figure 4.16(b) représente un sommet du graphe dual de degré 2. Les deux sommets du graphe initial adjacents par les arêtes qui définissent cette face de degré 2 codent deux régions partageant une frontière commune artificiellement découpée par les sommets du graphe dual. Intuitivement, l'opération de contraction a fusionné plusieurs pixels en une seule région. Les adjacences de chacun de ces pixels aux régions avoisinantes sont conservées dans le graphe contracté et une opération de simplification est nécessaire afin de supprimer ces relations redondantes. Dans le cas d'un sommet dual de degré 2, la simplification consiste à contracter l'une de ces deux arêtes duales. Notons que cette contraction dans le graphe dual doit être suivie par un opération de suppression dans le graphe initial afin de maintenir la dualité des deux graphes.

De même, le carré plein situé en bas, à gauche de la figure 4.16(b) représente un sommet du graphe dual de degré 1. La boucle associée à ce sommet dual code une adjacence entre deux pixels contractés en une même région par le noyau de contraction. Cette relation d'adjacence inutile est supprimée en contractant l'arête incidente au sommet dual de degré 1 et en supprimant la boucle associée dans le graphe initial.

La construction d'un nouveau niveau de la pyramide à partir d'une paire de graphes duaux $(G_0, \overline{G_0})$ et d'un noyau de contraction (S, N) se décompose donc en deux étapes respectivement illustrées sur les figures 4.17(b) et 4.17(c) :

1. Un ensemble d'opérations de contraction codées par les noyaux de contraction (S, N). Le dual du graphe contracté G_1 se déduit de $\overline{G_0}$ en supprimant dans celui-ci l'ensemble $\overline{N}$ d'arêtes.
2. La suppression des arêtes redondantes à l'aide d'un noyau de contraction appliqué sur le graphe dual. Un tel noyau est appelé un *Noyau de Suppression*. Les contractions d'arêtes effectuées dans le graphe dual doivent être suivies par des contractions d'arêtes dans le graphe initial afin de maintenir la dualité des deux graphes réduits.

Notons que toutes les boucles ne sont pas inutiles. En effet, une boucle contenant un sommet permet de coder une relation d'inclusion entre deux régions. Cette configuration est illustrée sur la figure 4.18 où la contraction de l'arête épaisse sur la figure 4.18(a) implique la fusion des régions situées à droite et à gauche de la région centrale. La contraction de cette arête crée une boucle qui :



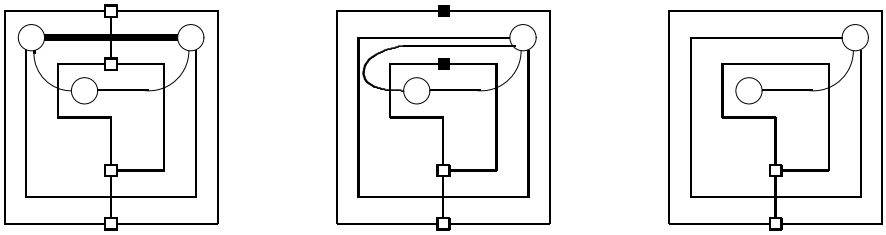
a) $(G_0, \overline{G_0})$ b) $(G_3, \overline{G_3})$ c) $(G_4, \overline{G_4})$

FIG. 4.17 – Les deux étapes de la réduction d'un graphe codé par des noyeaux de contraction.

– entoure le sommet codant la région incluse ;
– coupe l'arête duale qui connecte le contour de la région centrale à la région qui l'inclue.

Cette boucle code la relation d'inclusion entre les deux régions et n'est donc pas supprimée par l'opération de suppression d'arêtes redondantes. Cette dernière opération supprime simplement une arête double entre les deux sommets restant (figure 4.18(b)). Les seules boucles considérées comme inutiles sont donc les boucles vides.

Si chaque sommet du graphe initial code un pixel de la grille 4-connexe, chaque sommet du graphe contracté codera une région de la grille. La préservation d'arêtes multiples entre les sommets et de boucles non vides induit une correspondance 1 à 1 entre les arêtes du graphe réduit et les frontières de régions. Un tel schéma de simplification a été appliqué avec succès à l'analyse de composante connexes [124] et l'analyse de dessins industriels [123].



a) $(G_4, \overline{G_4})$ b) $(G_5, \overline{G_5})$ c) $(G_6, \overline{G_6})$

FIG. 4.18 – Création d'une arête fictive permettant de coder les relations d'inclusions.

4.6 Notre contribution aux représentation hiérarchiques : les pyramides combinatoires

Les cartes combinatoires et les cartes combinatoires généralisées sont des outils de modélisation particulièrement puissants pour la modélisation d'objets ou de partitions. En effet, les cartes combinatoires permettent de coder les partitions d'un espace de dimension n en objets orientables ou non orientables avec ou sans bords. Le concept de carte a tout d'abord été introduit par Edmond [70]

en 1960 puis étendu par plusieurs auteurs [84, 44, 132]. Ce modèle à été introduit dans la communauté informatique graphique par Lienhardt [129, 131] pour la modélisation d'objets 3D [130, 15]. L'utilisation des cartes combinatoires pour l'édition d'image $2D$ à été introduite par Braquelaire, Domenger, Guitton [26, 66]. Les thèses de Fiorio [76] et Brun [30] ont permis d'étendre le domaine d'application des cartes combinatoires à la segmentation d'images 2D. Plus récemment, les thèses de Damiand [60] et Desbarat [64] ont utilisé les cartes combinatoires 3D pour segmenter des images volumiques.

Notre contribution à ce domaine de recherche a consisté à introduire la notion de hiérarchie dans le codage de partitions codées par des cartes combinatoires 2D. De telles pyramides sont appelées des *Pyramides Combinatoires*. Après quelques définitions générales (section 4.6.1), nous allons étudier les deux opérations permettant de réduire une carte combinatoire (section 4.6.2) et définir des outils théoriques et des méthodes permettant d'effectuer cette réduction (section 4.6.3). Nous étudierons également quelques propriétés de ces opérations (section 4.6.4). Nous généraliserons ensuite les outils de réduction introduits dans la section 4.6.3 de façon à pouvoir mettre en relation non pas deux niveaux consécutifs de la pyramide mais deux niveaux quelconques (section 4.6.5). Ces outils peuvent être utilisés pour construire soit un niveau particulier d'une pyramide (section 4.6.6), soit l'ensemble de ses niveaux (section 4.6.7) à partir d'un codage de la base de celle-ci. Finalement, nous donnerons l'équivalent des notions de fenêtre de réduction et de champ récepteur dans le cadre des pyramides combinatoires avant d'indiquer quelques conjectures (section 4.6.9) et résultats d'implémentation (section 4.6.10). La conclusion de cette section se trouve en section 4.6.11. Nous indiquons également dans cette dernière section plusieurs perspectives de recherche ouvertes par ce travail. La combinaisons des représentations hiérarchiques et des cartes combinatoires étant un domaine jusqu'ici inexploré nos contributions débutent immédiatement après la section 4.6.1 donnant les définitions générales. Nous avons également adapté des concepts généraux sur les cartes combinatoires aux notations et contraintes utilisées dans le cadre des pyramides combinatoires (section 4.6.1, de la définition 8 à la fin de la section).

4.6.1 Les cartes combinatoires

Les cartes combinatoires sont basées sur la définition des cartes topologiques qui codent la décomposition d'une surface en un ensemble fini de points $\mathcal{V}$ et un ensemble fini $\mathcal{E}$ de courbes ouvertes de Jordan. Plus exactement [70, 84] :

Définition 4 Carte topologique

Étant donné un ensemble non vide $\mathcal{E}$ d'espaces topologiques (appelés arêtes) chacun homéomorphe à l'intervalle $I = [0, 1]$ ou au cercle unité S^1, nous définissons l'ensemble $\mathcal{V} \subset \mathcal{G} = \cup_{e \in \mathcal{E}} e$ ($\mathcal{V}$ est l'ensemble des sommets) tel que si $\Delta e = e \cap \mathcal{V}$ et $e^{\#} = e \setminus \Delta e$ alors :

1. *Si e est homéomorphe à S^1 alors $|\Delta e| = 1$ (e est appelé une boucle) alors que si e est homéomorphe à $I = [0, 1]$ Δe contient une ou deux des extrémités de e (e est alors respectivement appelé une arête libre et un segment).*
2. *Pour toute arête distincte $e_1, e_2 \in \mathcal{E}$, $e_1^{\#} \cap e_2^{\#} = \emptyset$.*
3. *Pour tout $v \in \mathcal{V}$, le nombre d'arêtes e telles que $v \in \Delta e$ est fini.*

Considérons la figure 4.19 et supposons pour simplifier que chaque contour de la figure 4.19(b) code une courbe réelle (le même type de raisonnement peut être conduit avec des courbes discrètes).

L'union de l'ensemble des points de contours définis par ces courbes défini l'ensemble $\mathcal{G}$. L'intersection de plusieurs contours défini un sommet appartenant à $\mathcal{V}$. Une courbe comprise entre deux sommets code une arête appartenant à $\mathcal{E}$. Si les deux sommets sont confondus, la courbe est dite fermée et est homéomorphe à l'ensemble S^1. C'est donc une boucle. Sinon la courbe est ouverte et est homéomorphe à l'intervalle $I = [0, 1]$, une telle courbe est appelée un segment.

Plus généralement, une carte topologique correspond au tracé sur une surface d'un ensemble de segments (éventuellement fermés). Ces segments ont au plus deux extrémités (condition 1) et ne peuvent se croiser que sur celles-ci (condition 2). De plus, la condition 3 assure qu'un nombre fini de segments se rencontrent en chaque sommet. Cette définition étant un peu trop générale pour les

(a) Lenna

(b) Contours d'une segmentation

FIG. 4.19 – Ensemble des contours d'une segmentation illustrant la notion de carte topologique.

applications qui vont suivre, nous allons supposer que l'ensemble $\mathcal{G}$ est connexe par arc. Comme nous verrons par la suite cette contrainte impose aux graphes décrivant la carte topologique d'être connexe. Nous supprimons également la possibilité des arêtes libres. Ceci revient à imposer dans la condition 1 que $|\Delta e| = 2$ si e est homéomorphe à $I = [0, 1]$. Dans ce cas, chaque arête privée de son ou ses sommets est homéomorphe à l'intervalle ouvert $]0, 1[$. Une telle arête comprend donc deux extrémités appelées brins. Chaque brin appartient par construction à une seule arête. De plus, les sommets étant situés aux extrémités des arêtes (condition 2), un brin n'appartient qu'a un seul sommet alors qu'un sommet peut correspondre à plusieurs extrémités d'arêtes et donc à plusieurs brins. Le nombre de brins d'un sommet est appelé son degré. Notons que ce degré est forcément fini (condition 3). Le degré de tout point p intérieur à un segment ($p \in e^{\#}, e \in \mathcal{E}$) est fixé à 2 par convention.

On définit sur l'ensemble des brins une application α qui associe à chaque brin d'une arête le brin restant dans la même arête. L'application α est clairement une involution. Intuitivement, l'involution α permet de passer d'une extrémité d'une arête à l'autre (figure 4.20(a)) et ses cycles codent les arêtes du graphe.

La définition des sommets suppose une condition supplémentaire permettant d'éviter les plongements pathologiques : Si $\mathcal{G}$ est plongé sur une surface $\mathcal{S}$ connexe, orientée sans bords, nous supposerons que :

Propriété 1 *Pour tout point $p \in \mathcal{G}$, de degré k, il existe un voisinage V_p de p dans $\mathcal{S}$ et un*

homéomorphisme ψ_p *de* V_p *dans* $D = \{z \in \mathbb{C} \mid |z| < 1\}$ *tel que* $\psi_p(p) = 0$ *et* $\psi_p(V_p \cap \mathcal{G}) = \{z \in \mathbb{C} \mid z^k \in [0, 1[\subset \mathbb{R}\}$.

Autrement dit, pour tout point p, l'ensemble $V_p \cap \mathcal{G}$ peut se voir comme un ensemble de rayons irradiant du point p. Nous pouvons donc définir un ordre cyclique sur l'ensemble des brins de p à partir de l'ordre $arg(z) = \{0, \frac{2\pi}{k}, \frac{4\pi}{k}, \ldots, \frac{2(k-1)\pi}{k}\}$ défini sur $\psi_p(V_p \cap \mathcal{G})$. Cet ordre définit une permutation sur l'ensemble des brins d'un sommet. Puisque chaque brin appartient à un seul sommet, la composition des cycles définis sur chacun des sommets définit une permutation σ sur l'ensemble des brins. Intuitivement, la permutation σ affecte chaque brin au brin suivant trouvé en tournant dans le sens positif autour d'un sommet (figure 4.20(b)).

(a) α (b) σ

FIG. 4.20 – Illustration des permutations α et σ

Pour une carte topologique $(\mathcal{V}, \mathcal{E})$ connexe, nous pouvons donc définir un ensemble $\mathcal{B}$ de brins et coder les arêtes $e \in \mathcal{E}$ par les cycles de la permutation α et les sommets $v \in \mathcal{V}$ par les cycles de la permutation σ. Ce codage nous est confirmé par Cori [56] :

Lemme 1 *Étant donnée une carte topologique connexe* $(\mathcal{V}, \mathcal{E})$ *et deux permutations* α *et* σ *sur l'ensemble des brins telles que deux brins* b *et* b' *appartiennent au même cycle*
- *de* α *s'ils appartiennent à une même arête et*
- *de* σ *s'ils appartiennent à un même sommet,*

il existe une bijection naturelle entre les arêtes de $\mathcal{E}$ *et les cycles de la permutation* α *ainsi qu'entre les sommets de* $\mathcal{V}$ *et les cycles de la permutations* σ.

Le triplet défini par $(\mathcal{B}, \sigma, \alpha)$ est appelé une carte combinatoire [56].

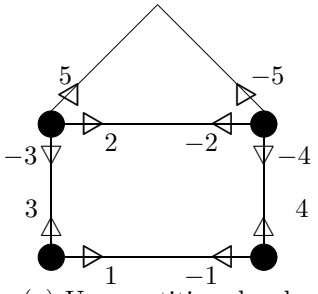
Définition 5 Carte combinatoire
Une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ *est définie par un ensemble de brins* $\mathcal{B}$ *et deux permutations* σ *et* α *sur* $\mathcal{B}$ *telles que* α *et une involution sans point fixe.*

$$\forall b \in \mathcal{B}\ \alpha^2(b) = b \text{ et } \alpha(b) \neq b.$$

La carte topologique étant supposée connexe par arc, la carte combinatoire associée est qualifiée de connexe. Intuitivement, le graphe associé à une telle carte est connexe.

Notez que l'absence de point fixe pour l'involution α correspond à la suppression des arêtes libres des cartes topologiques. Étant donné un brin b, le sommet possédant ce brin est codé par le cycle de σ contenant b. Ce cycle est noté $\sigma^*(b)$. De même, l'arête contenant un brin b est codée par le cycle de α contenant b. Ce cycle est noté $\alpha^*(b)$. Plus généralement, pour une permutation π, le π-cycle d'un brin b sera noté $\pi^*(b)$. La π-orbite d'un brin b correspond à l'ensemble des brins du cycle correspondant.

Un codage d'une partition du plan par une carte combinatoire est présenté sur la figure 4.21 où l'involution α est implicitement codée par le signe. Ce type de codage implicite est souvent utilisé dans les applications [30].



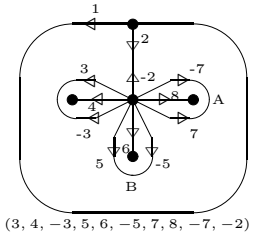
(a) Une partition du plan

$$
\begin{aligned}
G &= (\mathcal{B}, \sigma, \alpha) \\
\mathcal{B} &= \{-5, \ldots, -1, 1, \ldots, 5\} \\
\alpha &= (-1,1)(-2,2)(-3,3)(-4,4)(-5,5) \\
\sigma &= (1,3)(-1,4)(-4,-5,-2)(2,5,-3)
\end{aligned}
$$

(b) La carte combinatoire associée

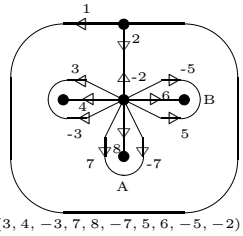
FIG. 4.21 – Codage d'une partition du plan par une carte combinatoire.

Notez que le codage d'une partition du plan par des graphes simples (section 4.3) ou des graphes duaux (section 4.5) ne fait pas appel à la notion de carte combinatoire. Cette correspondance explicite entre les arêtes de la carte combinatoire et celles de la carte topologique nous garantit *a priori* une description correcte de la partition. De fait, la figure 4.22 représente deux partitions différentes dans lesquelles nous avons échangé les sommets A et B. Ces partitions ne sont clairement pas représentables avec des graphes simples du fait des boucles. De plus, les boucles entourant A et B étant plongées dans la même face, ces deux partitions seront codées de manière identique dans le cadre des graphes duaux.



(3, 4, −3, 5, 6, −5, 7, 8, −7, −2)

(a)



(3, 4, −3, 7, 8, −7, 5, 6, −5, −2)

(b)

FIG. 4.22 – Deux cartes combinatoires distinctes non codables par des graphes simples et non distinguables par des graphes duaux. Le σ cycle du sommet central est indiqué sous chaque figure.

Il est assez clair qu'une re-numérotation des brins d'une carte combinatoire, si elle est accompagnée de la mise à jour correspondante des permutations σ et α, code la même carte topologique. Un codage de la carte topologique représentée sur la figure 4.21 avec des brins numérotés de a à e plutôt que de 1 à 5 fournirait par exemple un codage équivalent. Cette notion d'équivalence entre

les cartes combinatoires a été formalisée par Cori [56] :

Définition 6 Morphisme de cartes combinatoires

Étant données deux cartes combinatoires $G_1 = (\mathcal{B}_1, \sigma_1, \alpha_1)$ et $G_2 = (\mathcal{B}_2, \sigma_2, \alpha_2)$, un morphisme de carte combinatoire est une application ψ de $\mathcal{B}_1$ dans $\mathcal{B}_2$ telle que :

$$\forall d \in \mathcal{B}_1 \begin{cases} \psi(\alpha_1(d)) = \alpha_2(\psi(d)) \\ \psi(\sigma_1(d)) = \sigma_2(\psi(d)). \end{cases} \tag{4.5}$$

Si ψ est bijective elle est appelée un isomorphisme et l'équation 4.5 peut se réécrire :

$$\forall d \in \mathcal{B}_1 \begin{cases} \alpha_1(d) = \psi^{-1}(\alpha_2(\psi(d))) \\ \sigma_1(d) = \psi^{-1}(\sigma_2(\psi(d))). \end{cases} \tag{4.6}$$

Si nous prenons deux ensembles de brins $\mathcal{B}_1$ et $\mathcal{B}_2$ tels qu'il existe une application bijective π de $\mathcal{B}_1$ dans $\mathcal{B}_2$, l'application π établit un isomorphisme entre toute carte $G_1 = (\mathcal{B}_1, \sigma, \alpha)$ et la carte $G_2 = (\mathcal{B}_2, \pi \circ \sigma \circ \pi^{-1}, \pi \circ \alpha \circ \pi^{-1})$. De fait, les cartes G_1 et G_2 seront identiques modulo la renumérotation définie par π.

Cette notion de morphisme peut être très utile lorsque l'on désire comparer deux cartes ou une carte d'entrée à un ensemble de modèles. Elle nous sera également utile pour montrer la validité de nos algorithmes.

La construction précédente a défini les cartes combinatoires comme un codage d'une partition du plan par une carte topologique. Les cartes combinatoires peuvent toutefois se voir également comme un codage particulier d'un graphe planaire. On peut en particulier définir le dual d'une carte combinatoire.

Définition 7 Dual d'une carte combinatoire

Le dual d'une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ connexe noté $\overline{G}$ est une carte combinatoire définie par le triplet $\overline{G} = (\mathcal{B}, \varphi = \sigma \circ \alpha, \alpha)$.

Notons que cette définition (comme celles qui précèdent) est définie dans le cadre de cartes connexes.

Les cycles de la permutation $\varphi = \sigma \circ \alpha$ codent donc les faces de la carte combinatoire. Si G est connexe, on peut également montrer [84] qu'il existe une bijection naturelle entre les cycles de φ et l'ensemble des composantes connexes de $\mathcal{S}/\mathcal{G}$. Notre notion de face algébrique correspond donc bien aux faces définies par l'ensemble des arêtes de la carte topologique. Notons qu'un résultat bien connu de la théorie des graphes établissant que $\overline{\overline{G}} = G$ pour tout graphe planaire devient trivial dans le cadre des cartes combinatoires. En effet, α étant une involution, nous avons $\varphi \circ \alpha = \sigma \circ \alpha \circ \alpha = \sigma$, donc :

$$\overline{\overline{G}} = (\mathcal{B}, \varphi \circ \alpha, \alpha) = (\mathcal{B}, \sigma, \alpha) = G.$$

Si nous reprenons l'exemple de la figure 4.21, nous avons pour le brin -1 :

$$\varphi(-1) = \sigma(\alpha(-1)) = \sigma(1) = 3.$$

De même, $\varphi(3) = 2$, $\varphi(2) = -4$ et $\varphi(-4) = -1$. Notez qu'en raison de l'orientation positive choisie pour la permutation σ, chaque brin a sa face associée à main droite. L'ensemble des cycles de la permutation φ pour la figure 4.21 est égal à :

$$\varphi = (-1, 3, 2, -4)(1, 4, -5, -3)(-2, 5).$$

Les notions usuelles en théorie des graphes peuvent également se transcrire aisément en termes de cartes combinatoires. On distingue notamment les configurations suivantes (voir également la figure 4.23) :

Définition 8 Configurations particulières de brins
Soit une carte $G = (\mathcal{B}, \sigma, \alpha)$. Un sous ensemble $\mathcal{B}'$ de $\mathcal{B}$ sera dit symétrique si :

$$\alpha^*(\mathcal{B}') = \mathcal{B}'.$$

De plus, un brin $b \in \mathcal{B}$ sera appelé :
- *une boucle si $\alpha(b) \in \sigma^*(b)$;*
- *une boucle directe si $\sigma(b) = \alpha(b)$;*
- *un pont si $\alpha(b) \in \varphi^*(b)$;*
- *un brin pendant si $\sigma(b) = b$.*

Par extension, nous dirons qu'une arête est une boucle, une boucle directe, un pont ou une arête pendante si un des ses brins vérifie la propriété correspondante.

La donnée d'un ensemble de brins $\mathcal{B}' \subset \mathcal{B}$ induit la donnée d'un ensemble d'arêtes puisque chaque brin appartient à une seule arête. Dans un ensemble symétrique, les deux brins d'une arête appartiennent simultanément à l'ensemble $\mathcal{B}'$.

Notez que si $\alpha(b) \in \sigma^*(d)$ alors $b \in \sigma^*(\alpha(b))$. De même si $\alpha(b) \in \varphi^*(b)$ alors $b \in \varphi^*(\alpha(b))$. Les deux brins d'une arête sont donc simultanément des boucles ou des ponts. De plus, si $\sigma(b) = b$ alors $\varphi(\alpha(b)) = \sigma(b) = b$. Une arête pendante est donc également un pont. Chacune de ces caractérisations d'arête dans la carte initiale correspond à un autre type de configuration particulière dans la carte duale [35]. La Table 4.1 établit les correspondances entre ces configurations (Voir également la figure 4.23).

Configuration dans G	Configuration dans $\overline{G}$
boucle	pont
pont	boucle
boucle directe	arête pendante
arête pendante	boucle directe

TAB. 4.1 – Relations entre les configurations particulières d'arêtes dans la carte initiale et son dual

(a) : $G = (\mathcal{B}, \sigma, \alpha)$ (b) : $\overline{G} = (\mathcal{B}, \varphi, \alpha)$

FIG. 4.23 – *Les arêtes $\alpha^*(1)$,$\alpha^*(2)$,$\alpha^*(3)$,$\alpha^*(4)$ codent respectivement une boucle directe, un pont, une boucle et une arête pendante de la carte combinatoire G(a). Chacune de ces arêtes devient une autre configuration particulière dans la carte duale $\overline{G}$(b)*

La notion usuelle de sous-graphe [12] s'exprime dans le cadre des cartes combinatoires par une modification de la permutation σ par un opérateur appelé opérateur de restriction.

Définition 9 Opérateur de restriction

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et un sous ensemble symétrique $\mathcal{B}'$ de $\mathcal{B}$. L'opérateur de restriction $p_{\mathcal{B},\mathcal{B}'}$ est défini par :

$$\begin{array}{rccl} p_{\mathcal{B},\mathcal{B}'} : & \mathcal{B}' & \rightarrow & \mathcal{B} \\ & d & \mapsto & \sigma^{n-1}(d) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \sigma^k(d) \in \mathcal{B}'\} \end{array}$$

Intuitivement, pour tout brin $d \in \mathcal{B}'$, $p_{\mathcal{B},\mathcal{B}'}(d)$ est le premier brin que l'on rencontre en tournant dans le sens positif autour du sommet en partant du brin d tel que $\sigma(p_{\mathcal{B},\mathcal{B}'}(d)) \in \mathcal{B}'$. L'opérateur p permet donc de "sauter" une séquence de brins appartenant à $\mathcal{B} - \mathcal{B}'$ afin de connecter deux brins de $\mathcal{B}'$ (figure 4.24). Partant de l'opérateur de restriction, une sous carte combinatoire se définit naturellement comme :

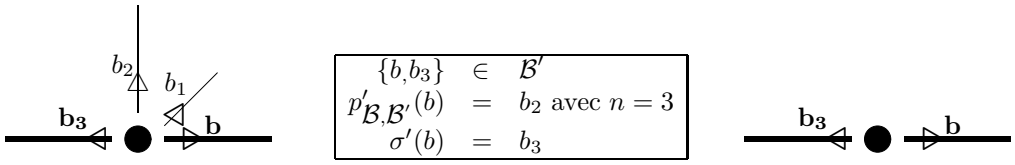
Définition 10 Sous-carte combinatoire

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et un sous-ensemble symétrique $\mathcal{B}'$ de $\mathcal{B}$. La carte $G' = (\mathcal{B}', \sigma', \alpha')$ telle que :

– $\sigma' = \sigma \circ p_{\mathcal{B},\mathcal{B}'}$;

– α' est la restriction de α à $\mathcal{B}'$

est appelée une sous-carte de G.



(a) permutation σ (b) calcul de σ' (c) permutation σ'

FIG. 4.24 – Construction d'une sous-carte. L'ensemble $\mathcal{B}'$ contient les brins b et b_3 notés en gras. Les arêtes correspondantes sont représentées en traits épais

La permutation α' n'étant que la restriction de la permutation α à un sous ensemble de brins, nous confondrons par la suite les deux notations et noterons α' simplement α. Notons que cette définition suppose que σ' et α' définissent respectivement une permutation et une involution sans point fixe sur $\mathcal{B}'$. La preuve de ce résultat se trouve dans [35].

La notion de sous-carte combinatoire nous permet d'introduire des sous-cartes particulières appelées *arbres* et *forêts* [12] :

Définition 11 Arbre

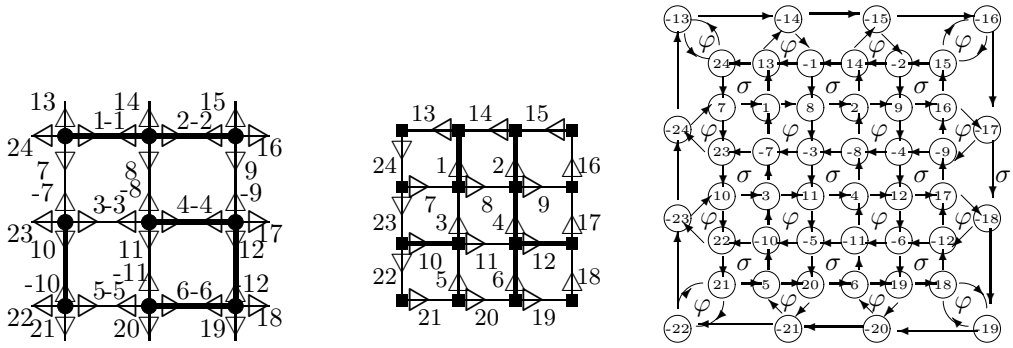
Une sous carte $A = (\mathcal{B}', \sigma', \alpha)$ d'une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ sera appelée un arbre de G si et seulement si la permutation $\varphi' = \sigma' \circ \alpha$ ne possède qu'un seul cycle.

Définition 12 Forêt

Une sous carte $F = (\mathcal{B}', \sigma', \alpha)$ d'une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$, sera appelée une forêt de G si et seulement si chaque composante connexe de F est un arbre.

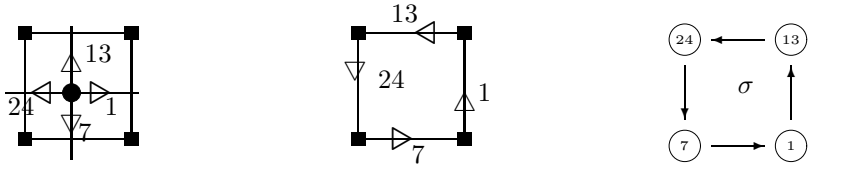
Notez qu'un arbre ne possède qu'un seule cycle de φ et donc qu'une seule face. Nos notions d'arbres et de forêts correspondent donc à celles utilisées en théorie des graphes.

Une carte combinatoire peut également être représentée par un graphe orienté $\mathcal{OG} = (V, E)$ tel que l'ensemble V des sommets est égal à l'ensemble des brins et une arête $e = (b_1, b_2)$ appartient à E si et seulement si $b_2 = \sigma(b_1)$ ou $b_2 = \varphi(b_1)$. Dans cette représentation les sommets et les faces de la carte combinatoire sont représentés par des faces du graphe $\mathcal{OG}$ (figure 4.25(c)). Notons que chaque sommet de $\mathcal{OG}$ a deux arcs rentrants (les σ et φ antécédents du brin) et deux arcs sortants (les σ et φ images du brin). Cette représentation nous sera utile pour représenter des séquences de brins dans la carte combinatoire qui ne codent pas des chemins et sont donc difficilement représentables.



(a) G (b) $\overline{G}$ (c) $\mathcal{OG}$

FIG. 4.25 – Codage d'une grille 3×3 par une carte combinatoire



(a) $\sigma^*(1)$ dans G (b) $\sigma^*(1)$ dans $\overline{G}$ (c) $\sigma^*(1)$ dans $\mathcal{OG}$

FIG. 4.26 – Le σ-cycle $\sigma^*(1)$ codant le pixel en haut à gauche de l'image (Figure 4.25) représenté dans la carte initiale G (a), la carte duale $\overline{G}$ (b) et le graphe orienté $\mathcal{OG}$ (c).

La figure 4.25 illustre le codage d'une grille 3×3 4-connexe par une carte combinatoire. Chaque sommet de la carte combinatoire (figure 4.25(a) et 4.26(a)) code un pixel de la grille. Le σ-cycle

d'un sommet code ses relations d'adjacence avec les sommets voisins. Les successeurs par α des brins 13 à 24 ne sont pas représentés sur cette figure afin de ne pas la surcharger. Le σ-cycle du sommet infini codant l'extérieur de l'image (section 4.5.2) est égal à la séquence de brins -13 à -24 : $(-13, -14, \ldots, -23, -24)$. La carte combinatoire duale est représentée sur la figure 4.25(b). Les α successeurs des brins positifs ne sont également pas représentés sur cette figure afin de ne pas la surcharger. Chaque sommet de cette carte duale peut être associé à un coin de pixel appelé *pointel* [80]. De plus, chacun de ses brins peut être compris comme un coté de pixel munis d'une orientation. Un coté de pixel étant dénommé un *lignel*. Cette interprétation se comprend aisément lorsque l'on considère la carte topologique (Définition 4) associée à une grille discrète comme la donnée d'un ensemble $\mathcal{V}$ de points codant les coins de pixels et un ensemble $\mathcal{E}$ d'arêtes codant les cotés de ces mêmes pixels. Plus simplement, si nous nous représentons un pixel comme un carré, celui-ci a 4 cotés et 4 coins. Par exemple le brin 1 sur la figure 4.25(b) code le coté droit du pixel situé en haut à gauche avec une orientation de bas en haut (figure 4.26(b)). Le brin $\alpha(1) = -1$ code le même lignel avec une orientation de haut en bas. Si nous utilisons cette interprétation des brins, le σ-cycle de chaque pixel défini la séquence de lignels qui délimitent celui-ci. Par exemple, le pixel en haut à gauche de la figure 4.25(b) (voir également figure 4.26(b)) est codé par le σ-cycle $(1, 13, 24, 7)$. De même, le pixel situé sur la première ligne, seconde colonne est codé par le σ-cycle $(2, 14, -1, 8)$. Le fait que ces deux pixels partagent un même lignel avec une orientation différente est codé par les brins 1 et -1 codant les deux orientation du même lignel.

Nous avons donc deux alternatives pour coder une partition d'une grille discrète à l'aide d'une carte combinatoire. Étant donnée une carte G, codant une partition, les régions de cette partition peuvent être codées par les sommets ou les faces de G. Une majorité de personnes travaillant dans le domaine des pyramides irrégulières codent les régions d'une partition par les sommets d'un graphe. Inversement, la majorité des personnes travaillant dans le domaine des cartes combinatoires codent les régions d'une partition par les faces de la carte combinatoire. Afin de supprimer cette ambiguïté, nous définissons les notions de carte des régions et cartes des frontières dans le cas où les cartes sont connexes.

Définition 13 Cartes de Régions et de Frontières

Une carte combinatoire G connexe associée à une partition d'une image est appelée :

- *une carte des régions si chaque sommet de G est associé à une région ;*
- *une carte des frontières si chaque face de G est associée à une région.*

Notez que si G est une carte des frontières, $\overline{G}$ est une carte des régions et inversement.

La figure 4.25(c) illustre le codage par le graphe $\mathcal{OG}$ de la carte combinatoire représentée sur la figure 4.25(a). Le sommet $\sigma^*(1)$ (figure 4.26(c)) est codé dans cette représentation par la face délimitée par les sommets 1, 13, 24 et 7.

4.6.2 Noyaux de contractions

Comme nous l'avons vu dans les section 4.3 et 4.5, une pyramide irrégulière est constituée d'une pile de graphes successivement réduits. La transposition de ces opérations dans le cadre des cartes combinatoires suppose donc une définition des opérations de contractions et suppressions.

L'opération de suppression se définit naturellement à l'aide de l'opérateur de restriction (Définition 9). La carte résultat d'une opération de suppression peut être vue comme une sous-carte (Définition 10) de la carte originale :

Définition 14 Opération de Suppression

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et un sous ensemble symétrique $\mathcal{B}'$ de $\mathcal{B}$. La sous carte de G, $G' = (\mathcal{B} - \mathcal{B}', \sigma', \alpha')$ est la carte combinatoire issue de la suppression des brins $\mathcal{B}'$. L'ensemble $\mathcal{B} - \mathcal{B}'$ est appelé l'ensemble des brins survivants. La carte réduite est notée $G \setminus \mathcal{B}'$.

Une expression plus simple de l'opération de suppression peut être donnée lorsque l'ensemble des brins supprimés se réduit à une seule arête. En effet, pour une carte $G = (\mathcal{B}, \sigma, \alpha)$ et une arête $\alpha^*(b)$ ne définissant dans $\mathcal{B}$ ni un pont ni une boucle directe, la permutation σ' de la carte $G = (\mathcal{B}, \sigma', \alpha) = G \setminus \alpha^*(b)$ est égale à :

$$\left\{ \begin{array}{lll} \forall b' \in \mathcal{B} - \{\sigma^{-1}(b), \sigma^{-1}(-b)\}, & \sigma'(b') = \sigma(b') \\ \sigma'(\sigma^{-1}(b)) & = & \sigma(b) \\ \sigma'(\sigma^{-1}(\alpha(b))) & = & \varphi(b). \end{array} \right. \tag{4.7}$$

Notez que l'expression ci-dessus reste définie si $\alpha^*(b)$ est un pont de G. Nous excluons toutefois de telles opérations afin d'éviter une déconnexion de la carte combinatoire. L'expression de la permutation σ' dans le cas où $\alpha^*(b)$ définit une boucle directe est donnée dans [35].

Nous avons vu dans la section 4.5 que la suppression d'un ensemble d'arêtes N dans un graphe G impliquait la suppression des arêtes correspondantes dans $\overline{G}$. Inversement, la contraction d'un ensemble d'arêtes N dans G implique la suppression des arêtes associées dans $\overline{G}$. Dans le cadre des cartes combinatoires, une carte et son dual sont définis à partir du même ensemble de brins (Définition 14). On peut donc définir l'opération de contraction comme une opération de suppression effectuée dans la carte duale :

Définition 15 Contraction

Soient une carte $G = (\mathcal{B}, \sigma, \alpha)$ et un sous ensemble symétrique de brins $\mathcal{B}' \subset \mathcal{B}$. La carte contractée est définie par :

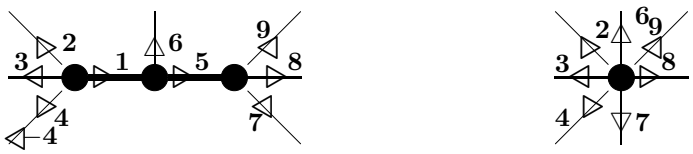
$$G' = G/\mathcal{B}' = \overline{\overline{G} \setminus \mathcal{B}'}.$$

Le résultat d'une opération de contraction est représenté sur la figure 4.27. Notez que l'opération de contraction préserve, comme l'opération de suppression, l'orientation des arêtes autour du sommet contracté. Ainsi, si nous tournons dans le sens positif(ou trigonométrique) autour de l'union des 3 sommets définis sur la figure 4.27(a) en partant du brin 2, nous rencontrons la séquence de brins $(2, 3, 4, 7, 8, 9, 6)$. Cet ordre est conservé autour du sommet contracté (figure 4.27(b)). Si les sommets de la carte combinatoire codent les régions d'une partition, l'ordre cyclique défini sur chaque sommet par la permutation σ permet de coder la séquence de régions que l'on rencontre en tournant dans le sens positif autour de la région.

Comme pour l'opération de suppression, une expression plus simple de l'opération de contraction peut être donnée lorsque l'ensemble des brins supprimés se réduit à une seule arête. Soient une carte $G = (\mathcal{B}, \sigma, \alpha)$ et une arête $\alpha^*(d)$ ne définissant dans $\mathcal{B}$ ni une boucle ni une arête pendante, la permutation σ' de la carte $G = (\mathcal{B}, \sigma', \alpha) = G/\alpha^*(d)$ est égale à :

$$\left\{ \begin{array}{lll} \forall d' \in \mathcal{B} - \sigma^{-1}(\alpha^*(d)) & \sigma'(d) = \sigma(d) \\ \sigma'(\sigma^{-1}(d)) & = & \varphi(d) \\ \sigma'(\sigma^{-1}(\alpha(d))) & = & \sigma(d). \end{array} \right. \tag{4.8}$$

Notez que les configurations interdites pour la contraction sont les configurations duales de celles interdites pour la suppression (équation 4.7 et Table 4.1). Une expression analytique de la contraction d'une arête pendante peut être trouvée dans [35].



(a) G (b) $G/\alpha^*(1,5)$

FIG. 4.27 – Opération de contraction. Les arêtes correspondant à des brins contractés sont représentées en traits épais

Notons que la Définition 15 n'impose pas de contraintes particulières à l'ensemble des brins contractés si ce n'est d'appliquer la même opération sur les deux brins d'une même arête. Il est donc parfaitement possible de contracter une boucle ou un ensemble d'arêtes formant un cycle. Cette opération peut se définir formellement mais conduit à une déconnexion de la carte combinatoire. De même, la suppression d'un ensemble quelconque d'arêtes peut conduire à la suppression d'un pont ce qui encore une fois déconnecte la carte combinatoire. Afin d'éviter de telles déconnexions, nous introduisons le concept de noyaux de contraction et de suppression :

Définition 16 Noyaux de Contraction

Soit une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$. Un sous-ensemble symétrique de brins $K \subset \mathcal{B}$ sera appelé un noyau de contraction si et seulement si :

- *K forme une forêt de G (Définition 12) ;*
- *K ne contient pas tous les brins de G :*

$$\mathcal{BS} = \mathcal{B} - K \neq \emptyset.$$

L'ensemble $\mathcal{BS}$ est appelé l'ensemble des brins survivants.

L'ensemble des brins contractés formant une forêt de la carte initiale, l'opération de contraction ne peut conduire à la contraction d'une boucle. La connexité de la carte combinatoire est donc assurée par l'opération de contraction. Dans le cadre des cartes combinatoires, un sommet est implicitement défini par l'ensemble des ces arêtes incidentes qui forment un cycle de la permutation σ. Nous devons donc préserver au moins une arête de la carte initiale durant l'opération de contraction. Cette propriété est assurée par la seconde condition d'un noyau de contraction.

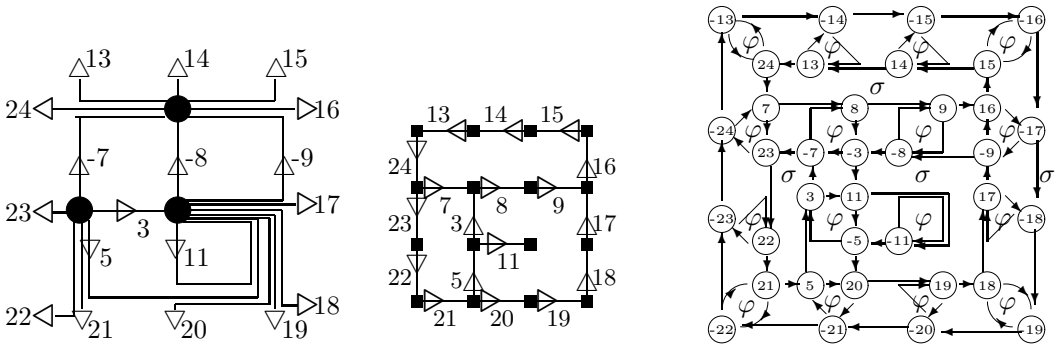
De même, un noyau de suppression peut se définir comme un noyau de contraction appliqué à la carte duale.

Définition 17 Noyau de Suppression

Soit une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$. Un sous-ensemble symétrique de brins $K \subset \mathcal{B}$ sera appelé un noyau de suppression s'il forme un noyau de contraction de $\overline{G}$.

Un noyau de suppression étant un noyau de contraction de la carte duale $\overline{G}$, celle-ci reste connexe après l'opération de contraction. Les deux cartes G et $\overline{G}$ possédant le même nombre de composantes connexes [84, 35], la carte initiale reste connexe après l'opération de suppression.

Cette définition d'un noyau de suppression est différente de celle introduite dans le cadre des graphe duaux (section 4.5.2) où l'opération de contraction était utilisée pour fusionner des régions tandis que les opérations de suppression étaient restreintes à la suppression d'arêtes rendues redondantes par l'opération de contraction. La suppression des arêtes redondantes peut dans le cadre des graphes duaux comme dans le cadre des cartes combinatoires être codée par un noyau vérifiant les propriétés de la Définition 17. Toutefois, spécifier explicitement l'ensemble des brins à supprimer ne facilite pas la démonstration de la plupart des résultats que nous verrons dans les sections suivantes. De plus, cette dualité entre les définitions des noyaux de contraction et de suppression permet de rester libre du type de carte utilisé pour coder la partition (Définition 13). Ainsi, si la partition est codée par une carte des régions, l'opération utilisée pour fusionner les régions sera l'opération de contraction, l'opération de suppression n'étant utilisée que pour "nettoyer" les frontières de la partition. Inversement, si la partition est codée par une carte des frontières, la fusion de régions sera codée par des noyaux de suppression alors que les opérations de contraction seront restreintes à la suppression d'arêtes redondantes.



(a) $G_1 = G/K_1$ (b) $\overline{G_1} = \overline{G/K_1}$ (c) $\mathcal{OG}_1$

FIG. 4.28 – Réduction de la grille de la Figure 4.25 par le noyeau de contraction $K_1 = \alpha^*(1, 2, 4, 12, 6, 10)$

La figure 4.28 illustre la contraction de la carte représentée sur la figure 4.25 par un noyau de contraction K_1 défini par les arbres $\alpha^*(1, 2)$, $\alpha^*(4, 12, 6)$ et $\alpha^*(10)$. Puisque chaque sommet de la carte possède un brin contracté, cette forêt recouvre la carte combinatoire initiale et nous obtenons 3 sommets survivants codant 3 régions. L'arbre $\alpha^*(1, 2)$ contracte la première ligne de l'image en un seul sommet. Cette opération est respectivement représentée sur les figures 4.29 et 4.30 dans la carte initiale G et son dual $\overline{G}$.

La suppression des arêtes redondantes de la carte contractée représentée sur la figure 4.28 par le noyau de suppression

$$K_2 = \{\alpha^*(15, 14, 13, 24), \alpha^*(9), \alpha^*(11, 3), \alpha^*(19, 18, 17), \alpha^*(22, 21)\}$$

(a) Ligne initiale (b) Ligne contracté

FIG. 4.29 – Contraction de la première ligne représentée sur la Figure. 4.28(a). Les arêtes contractées sont représentées par des traits épais.

(a) Ligne initiale (b) Ligne contractée

FIG. 4.30 – Contraction de la première ligne dans la carte duale (Figure. 4.28(b)). Les arêtes à supprimer sont représentées en traits épais.

est illustrée sur la figure 4.31.

4.6.3 Chemins de connexion

Soient une carte initiale $G = (\mathcal{B}, \sigma, \alpha)$ et un noyau de suppression K. La permutation σ' de la carte réduite $G' = (\mathcal{BS} = \mathcal{B} - K, \sigma', \alpha)$ peut se définir à partir de l'opérateur de restriction (Définition 9). Nous avons :

$$\forall d \in \mathcal{BS} \quad \sigma'(d) = \sigma^n(d) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \sigma^k(d) \in \mathcal{BS}\}. \tag{4.9}$$

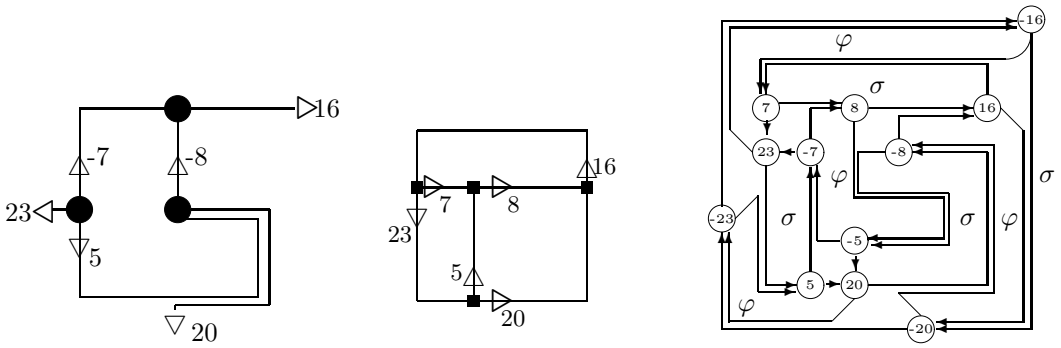
De même, l'opération de contraction est définie comme une opération de suppression appliquée dans la carte duale. Pour une carte $G = (\mathcal{B}, \sigma, \alpha)$ et un noyau de contraction K, la contraction des brins de K est donc équivalente à leur suppression dans $\overline{G} = (\mathcal{B}, \varphi = \sigma \circ \alpha, \alpha)$. Si $\overline{G'} = (\mathcal{BS} = \mathcal{B} - K, \varphi', \alpha)$ désigne la carte déduite de $\overline{G}$ par la suppression des brins de K, nous avons de par la définition de l'opérateur de restriction (Définition 9) :

$$\forall d \in \mathcal{BS} \quad \varphi'(d) = \varphi^n(d) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \varphi^k(d) \in \mathcal{BS}\}. \tag{4.10}$$

Puisque $\varphi'(\alpha(d)) = \sigma'(\alpha^2(d)) = \sigma'(d)$ l'équation précédente peut se réécrire :

$$\forall d \in \mathcal{BS} \quad \sigma'(d) = \varphi^n(\alpha(d)) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \varphi^k(\alpha(d)) \in \mathcal{BS}\}. \tag{4.11}$$

Pour un brin survivant b, la séquence de brins $\sigma(b) \ldots . \sigma^{n-1}(b)$ avec $\sigma^n(b) \in \mathcal{BS}$ représente la séquence de brins non survivants qu'il nous faut traverser pour connecter b à $\sigma'(b)$ si K est un



(a) $(G/K_1) \setminus K_2$ (b) $\overline{(G/K_1) \setminus K_2}$ (c) $\mathcal{OG}_2$

FIG. 4.31 – Suppression de brins sur la carte contractée représentée sur la Figure 4.28 par le noyeau de suppression $K_2 = \{\alpha^*(15,14,13,24), \alpha^*(9), \alpha^*(11,3), \alpha^*(19,18,17), \alpha^*(22,21)\}$

noyau de suppression (équation 4.9). De même, si K est un noyau de contraction, $\varphi(b)\ldots.\varphi^{p-1}(b)$ avec $\varphi^p(b) \in \mathcal{BS}$ représente la séquence de brins non survivants à traverser pour connecter b à $\varphi'(b)$(équation 4.10). De telles séquences de brins non survivants, préfixées par le brin initial b sont appelées des chemins de connexion.

Définition 18 Chemins de connexion

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$, un noyau K et un brin $b \in \mathcal{BS}$. Le chemin de connexion (ou connecting walk) associé à b est défini par :

– *Si K est un noyau de contraction*

$$CW(b) = b\varphi(b)\ldots\varphi^{n-1}(b) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \varphi^k(b) \in \mathcal{BS}\};$$

– *Si K est un noyau de suppression*

$$CW(b) = b\sigma(b)\ldots\sigma^{n-1}(b) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \sigma^k(b) \in \mathcal{BS}\}.$$

Pour un noyau K et un brin survivant b tels que $CW(b) = b.b_1\ldots.b_p$ les équations 4.9 et 4.10 peuvent se réécrire :

$$\begin{array}{lcll} \varphi'(b) & = & \varphi(b_p) & \text{Si } K \text{ est un noyau de contraction} \\ \sigma'(b) & = & \sigma(b_p) & \text{Si } K \text{ est un noyau de suppression} \end{array} \quad (4.12)$$

où $G' = (\mathcal{BS}, \sigma', \alpha)$ désigne la carte réduite.

Les algorithmes représentés sur la figure 4.32 sont basés sur l'équation 4.12 et calculent la carte réduite après une opération de suppression (figure 4.32(a)) ou de contraction (figure 4.32(b)). Notons que l'algorithme de suppression parcourt pour chaque brin survivant b la séquence $CW(b)$ alors que l'algorithme de contraction parcourt la séquence $CW(\alpha(b))$. On a de fait (équation 4.12) :

$$\sigma'(b) = \varphi'(\alpha(b)) = \varphi(b_p) \text{ avec } CW(\alpha(b)) = \alpha(b).b_1\ldots.b_p.$$

```
suppression(carte G,
            noyeau K)
{
    Pour tout b ∈ BS = B − K
    {
        b' = b ;
        tant que b' ∈ K
            b' = σ(b')
        σ'(b) = b'
    }
}
```

(a) Suppression

```
1   contraction(carte G,
2               noyeau K)
3   {
4       Pour tout b ∈ BS = B − K
5       {
6           b' = α(b) ;
7           tant que b' ∈ K
8               b' = φ(b')
9           σ'(b) = b'
10      }
11  }
```

(b) Contraction

FIG. 4.32 – Algorithmes calculant la carte réduite définie par la carte initiale G et un noyau de contraction (a) ou de suppression (b)

Chaque chemin de connexion contient par construction un et un seul brin survivant qui définit le chemin. Plus précisément, nous avons montré [36] que l'ensemble des chemins de connexion forme une partition de l'ensemble $\mathcal{B}$ des brins initiaux :

$$\forall d \in \mathcal{B} \quad \exists! d' \in \mathcal{BS} \mid d \in CW(d'). \tag{4.13}$$

où le symbole $\exists!$ représente l'assertion il existe un et un seul.

Autrement dit, soit un brin survit auquel cas il définit un chemin de connexion, soit il est contracté et appartient dans ce cas à un et un seul chemin de connexion. Cette propriété semble suggérer qu'un algorithme calculant la carte contractée ne devrait pas avoir à parcourir plusieurs fois les mêmes brins. De fait, on peut montrer[36] que les algorithmes représentés sur la figure 4.32 parcourent une fois et une seule chaque brin de la carte initiale $G = (\mathcal{B}, \sigma, \alpha)$. Leur complexité est donc égale à $\mathcal{O}(|\mathcal{B}|)$.

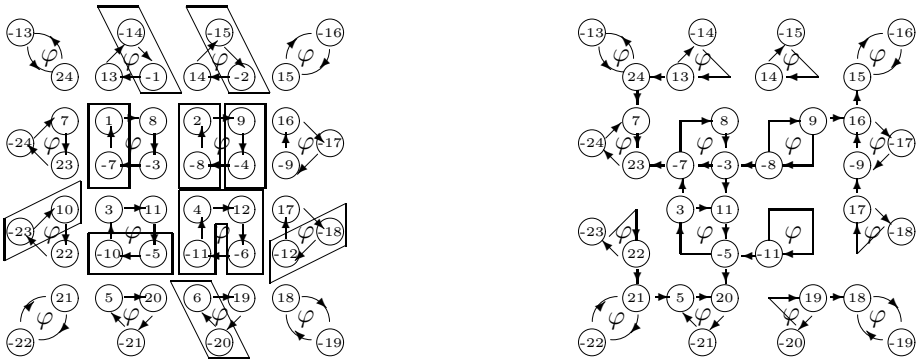
Si K est un noyau de contraction et si nous munissons l'ensemble $\mathcal{B}_K = \{CW(d),\ d \in \mathcal{BS}\}$ des chemins de connexion des applications :

$$\begin{array}{lccl} \alpha_K : & \mathcal{B}_K & \rightarrow & \mathcal{B}_K \\ & CW(d) & \mapsto & CW(\alpha(d)) \\ & & & \\ \varphi_K : & \mathcal{B}_K & \rightarrow & \mathcal{B}_K \\ & CW(d) & \mapsto & CW(\varphi^n(d)) \text{ avec } CW(d) = d.\ldots.\varphi^{n-1}(d) \end{array} \tag{4.14}$$

nous pouvons montrer [36] que les applications φ_K et α_K définissent respectivement une permutation et une involution sans point fixe sur $\mathcal{B}_K$. Nous pouvons donc structurer l'ensemble des chemins de connexion en une carte combinatoire appelée carte des chemins de connexion :

Définition 19 Carte des chemins de connexion

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et un noyau de contraction K. Le triplé $G_K = (\mathcal{B}_K, \sigma_K, \alpha_K)$, avec $\sigma_K = \varphi_K \circ \alpha_K$ et φ_K, α_K définis par l'équation 4.14, est une carte combinatoire appelée carte des chemins de connexion.



(a) $\mathcal{OG}$ (b) $\mathcal{OG}_1$

FIG. 4.33 – Graphe $\mathcal{OG}$ superposé aux chemins de connexions définis par $K_1 = \alpha^*(1, 2, 4, 12, 6, 10)$ (a). La figure (b) représente le graphe $\mathcal{OG}_1$ issue de cette contraction. Seules les liaisons φ sont représentées dans les deux figures. Les brins non encadrés dans la figure (a) ont des chemins de connexion triviaux ($CW(b) = b$).

Une carte des chemins de connexions est donc une carte dans laquelle nous avons regroupé les brins appartenant à un même chemin de connexion. Dans le cas d'un noyau de contraction, le φ successeur d'un de ces regroupement est défini comme le φ successeur du dernier brin du chemin. Si nous reprenons l'équation 4.14, nous avons pour tout brin survivant b :

$$\varphi_K(CW(b)) = CW(\varphi^n(b)).$$

Or, d'après la définition des chemins de connexion (définition 18, équation 4.12), nous avons également $\varphi^n(b) = \varphi'(b)$ où φ' représente la permutation φ de la carte contractée. On a donc :

$$\varphi_K(CW(b)) = CW(\varphi^n(b)) = CW(\varphi'(b)).$$

La permutation φ_K associe donc au chemin de connexion d'un brin survivant b, le chemin de connexion du φ'-successeur de b. La carte des chemins de connexion code donc les mêmes φ-successeurs que la carte contractée, à la différence prêt que l'on conserve l'ensemble des brins dans la carte des chemins de connexion. Plus intuitivement, la carte des chemins de connexion peut se concevoir comme une étape intermédiaire entre la carte initiale et la carte contractée. Ce résultat est illustré sur la figure 4.33 où nous avons représenté côte à côte, la carte initiale avec les chemins de connexions et la carte contractée. Afin de simplifier les figures, nous n'avons représenté que les liaisons φ entre les brins. Il ressort de la réflexion précédente que le φ_K successeur d'un chemin de connexion correspond au brin indiqué par la flèche sortante du dernier brin de chaque chemin. On a par exemple $\varphi_K(CW(-7)) = CW(8)$. On peut constater que les relations «φ-successeur de» entre les chemins de connexion et entre les brins survivants sont identiques.

Plus formellement, puisque chaque chemin de connexion possède un et un seul brin survivant, l'application CW qui associe à chaque brin son chemin de connexion est une application bijective de $\mathcal{BS}$ dans $\mathcal{B}_K$. Plus précisément, cette application définit un isomorphisme (Définition 6) de la carte contractée $G' = G/K$ dans la carte des chemins de connexion [35] :

Théorème 2 *Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et un noyau de contraction K. La carte contractée $G' = G/K = (\mathcal{BS}, \sigma', \alpha')$ est isomorphe à la carte des chemins de connexion.*

Si nous utilisons l'équation 4.6 de la Définition 6 nous obtenons :

$$\forall d \in \mathcal{BS} \quad \left\{ \begin{array}{l} \alpha'(d) = CW^{-1}(\alpha_K(CW(d))) \\ \sigma'(d) = CW^{-1}(\sigma_K(CW(d))). \end{array} \right.$$

Pour un brin $d \in \mathcal{BS}$ tel que $CW(\alpha(d)) = \alpha(d). \ldots, \varphi^{n-1}(\alpha(d))$, nous avons :

$$\left\{ \begin{array}{lcll} \alpha'(d) & = & CW^{-1}(\alpha_K(CW(d))) & = & CW^{-1}(CW(\alpha(d))) \\ \sigma'(d) & = & CW^{-1}(\sigma_K(CW(d))) & = & CW^{-1}(\varphi_K(CW(\alpha(d)))) = CW^{-1}(CW(\varphi^n(\alpha(d)))). \end{array} \right.$$

L'application CW étant bijective, nous obtenons :

$$\left\{ \begin{array}{lcl} \alpha'(d) & = & \alpha(d) \\ \sigma'(d) & = & \varphi^n(\alpha(d)). \end{array} \right.$$

La permutation α de la carte contractée reste donc inchangée tandis que l'image par σ' d'un brin survivant est le premier brin survivant rencontré en itérant φ à partir du brin $\alpha(d)$. Un résultat équivalent peut être obtenu si K est un noyau de suppression en prenant α_K comme défini dans l'équation 4.14 et σ_K égal à :

$$\begin{array}{rccl} \sigma_K : & \mathcal{B}_K & \rightarrow & \mathcal{B}_K \\ & CW(d) & \mapsto & CW(\sigma^n(d)) \text{ avec } CW(d) = d. \ldots . \sigma^{n-1}(d). \end{array}$$

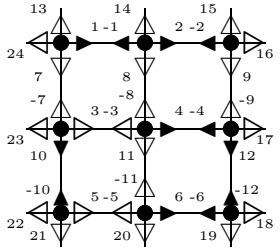
On obtient dans ce cas :

$$\left\{ \begin{array}{lcl} \alpha'(d) & = & \alpha(d) \\ \sigma'(d) & = & \sigma^n(d). \end{array} \right.$$

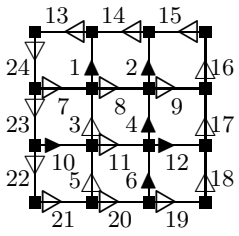
Ces résultats sont bien compatibles avec la définition des opérations de suppression et contraction dont les équations 4.9 et 4.11 sont une conséquence directe. L'intérêt de l'isomorphisme entre la carte contractée et la carte des chemins de connexion ne réside donc pas dans la formulation de la permutation σ' (qui peut se déduire plus simplement des équations 4.9 ou 4.11) mais dans le lien qu'il établit entre la carte initiale et la carte contractée. En effet, chaque brin de la carte initiale appartient à un et un seul chemin de connexion (équation 4.13). De plus, à chaque chemin de connexion correspond par l'isomorphisme un unique brin de la carte contractée. Chaque brin de la carte initiale peut donc être associé de façon unique à un brin de la carte contractée. Cette correspondance permet de décrire des propriétés de la carte contractée à partir de la carte initiale et de son noyau de contraction.

La figure 4.34(a) représente les chemins de connexion sur la carte G définis par le noyau de contraction $K_1 = \alpha^*(1, 2, 4, 12, 6, 10)$ (section 4.6.2). Si nous considérons, par exemple, le brin $-11 \in \mathcal{BS}_1$, nous avons :

$$\{\varphi(-11) = 4, \varphi^2(-11) = 12, \varphi^3(-11) = -6\} \subset K_1 \text{ et } \varphi^4(-11) = -11 \in \mathcal{BS}_1$$



(a) G



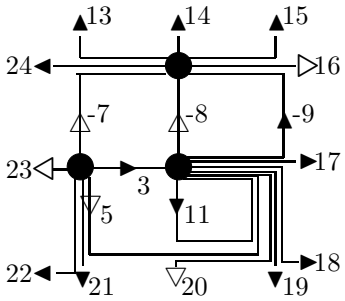
(b) $\overline{G}$

$CW_1(-7)$	=	-7.1	$CW_1(-11)$	=	$-11.4.12.-6$
$CW_1(-14)$	=	$-14.-1$	$CW_1(-18)$	=	$-18.-12$
$CW_1(-8)$	=	-8.2	$CW_1(-20)$	=	-20.6
$CW_1(-15)$	=	$-15.-2$	$CW_1(-23)$	=	-23.10
$CW_1(9)$	=	$9.-4$	$CW_1(-5)$	=	$-5.-10$

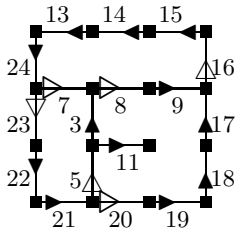
(c) Chemins de connexion

FIG. 4.34 – Chemins de connexion définis par $K_1 = \alpha^*(1, 2, 4, 12, 6, 10)$ sur la grille 3×3. Les brins contractés sont représentés par des triangles pleins. Les chemins de connexion non représentés sur la Figure (c) sont réduits à un seul brin.

Le chemin de connexion associé à -11 est donc : $CW(-11)$ =-11.4.12.-6. Nous avons de plus par l'équation 4.12 : $\varphi'(-11) = \varphi(-6) = -11$, ce qui équivaut à $\varphi'(-11) = \sigma'(11) = -11$ (figure 4.28). En d'autres termes, le noyau de contraction a contracté toutes les arêtes d'une face excepté $\alpha^*(11)$ qui devient une boucle directe dans la carte contractée (figure 4.35(a)).



(a) $G_1 = G/K_1$



(b) $\overline{G_1} = \overline{G/K_1}$

$CW_2(16)$	=	16.15.14.13.24	$CW_2(-8)$	=	$-8. - 3.11. - 11$
$CW_2(-23)$	=	$-23. - 24. - 13. - 14. - 15$	$CW_2(5)$	=	5.3
$CW_2(8)$	=	8.9	$CW_2(-20)$	=	$-20. - 21. - 22$
$CW_2(20)$	=	$20.19.18.17. - 9$	$CW_2(23)$	=	23.22.21
$CW_2(-16)$	=	$-16. - 17. - 18. - 19$			

(c) Chemins de connexion

FIG. 4.35 – Réduction de la grille contractée par le noyeau de suppression K_2(équation 4.15). Les brins supprimés sont représentés par des triangles pleins. Les chemins de connexion non représentés sur la Figure (c) sont réduits à un seul brin.

De même, la figure 4.35 représente les chemins de connexion définis sur G_1 (figure 4.28) par le noyau de suppression

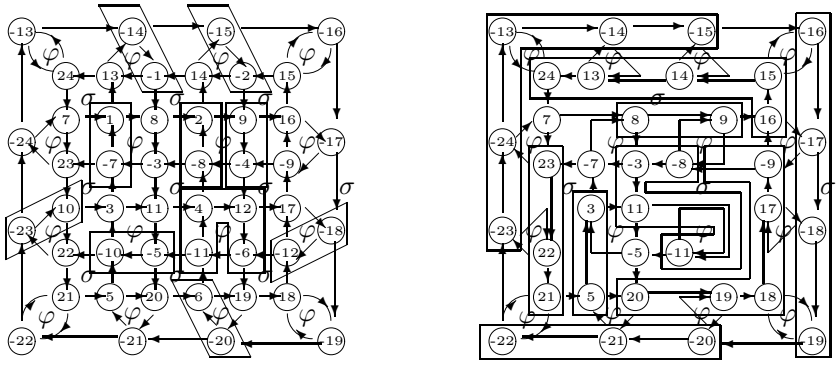
$$K_2 = \{\alpha^*(15, 14, 13, 24), \alpha^*(9), \alpha^*(11, 3), \alpha^*(19, 18, 17), \alpha^*(22, 21)\}. \tag{4.15}$$

Si nous considérons le brin 23, nous avons :

$$\{\sigma_1(23) = 22, \sigma_1^2(23) = 21\} \subset K_2 \text{ alors que } \sigma_1^3(23) = 5 \in \mathcal{BS}_2.$$

Le chemin de connexion associé au brin 23 est donc $CW_2(23) = 23.22.21$ et nous avons $\sigma_2(23) = 5$ (figure 4.31).

Les représentations des chemins de connexion sur les graphes orientés $\mathcal{OG}$ (figure 4.25) et $\mathcal{OG}_1$ (figure 4.28) sont données sur la figure 4.36.



(a) Chemins de connexion dans $\mathcal{OG}$ définis par K_1

(b) Chemins de connexion dans $\mathcal{OG}_1$ définis par K_2

FIG. 4.36 – Chemins de connexion définis par K_1 (a) et K_2(b).Voir également Figures 4.34(c) et 4.35(c)

4.6.4 Noyaux de contraction équivalents

Nous avons vu dans la section 4.6.2 qu'une pyramide était définie à partir de noyaux de contraction et de suppression. Dans la suite de cet ouvrage, nous nous référerons au *type* d'un noyau pour désigner l'opération impliquée par celui-ci. Ainsi, deux noyaux seront dis de même type si ce sont deux noyaux de contraction ou de suppression. On distingue également les relations suivantes entre les noyaux :

Définition 20 Relations entre les noyaux

Soient deux noyaux K_1, K_2 et les cartes combinatoires G, $G_1 = G/K_1$ et $G_2 = G_1/K_2$. Nous dirons dans ce cas que K_1 et K_2 sont successifs. Cette relation sera noté $K_1 \prec K_2$.

Soient deux noyaux de même type K_1 et K_2 tout deux définis sur une même carte G. Nous dirons que K_2 inclut K_1 si $K_1 \subset K_2$.

Pour deux noyaux successifs de même type K_1 et K_2 appliqués sur une carte $G = (\mathcal{B}, \sigma, \alpha)$ et produisant respectivement les cartes réduites $G_1 = (\mathcal{BS}_1, \sigma_1, \alpha)$ et $G_2 = (\mathcal{BS}_2, \sigma_2, \alpha)$, nous avons :

$$K_2 \subset \mathcal{BS}_1 \subset \mathcal{B}.$$

L'ensemble de brins K_1 étant également inclus dans $\mathcal{B}$, l'ensemble $K_1 \cup K_2$ est un sous ensemble de $\mathcal{B}$ représentant l'ensemble des brins à contracter ou supprimer pour passer du niveau 0 au niveau 2. Il est donc légitime de se demander si $K_1 \cup K_2$ ne définit pas un noyau permettant de regrouper les opérations définies par K_1 et K_2.

Inversement, soient une carte $G = (\mathcal{B}, \sigma, \alpha)$ et deux noyaux K_1 et K_2 définis sur G tels que $K_1 \subset K_2$. L'ensemble de brins $K_2 - K_1$ représente l'ensemble des brins de K_2 qu'il est nécessaire de contracter (resp. supprimer) après l'application de K_1 pour avoir contracté (resp. supprimé) l'ensemble des brins de K_2. On peut donc se demander si l'ensemble $K_2 - K_1$ ne définit pas un

noyau tel que l'application de K_1 suivie de celle de $K_2 - K_1$ produit le même résultat qu'une application directe de K_2.

La réponse à ces deux questions est apportée par le théorème suivant (voir illustration figure 4.37) :

Théorème 3 *Soit une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et deux noyaux de même type K_1 et K_2 tels que $K_1 \prec K_2$. L'ensemble des brins $K_1 \cup K_2$ forme un noyau tel que :*

$$(G/K_1)/K_2 = G/(K_1 \cup K_2).$$

Un tel noyau est appelé un noyau équivalent.

Inversement, soient deux noyaux K'_1 et K'_2 défini sur G tels que $K'_1 \subset K'_2$, alors $K'_2 - K'_1$ est un successeur de K'_1 tel que :

$$(G/K'_1)/(K'_2 - K'_1) = G/K'_2.$$

Preuve:

Voir [35] théorèmes 4 et 6 □

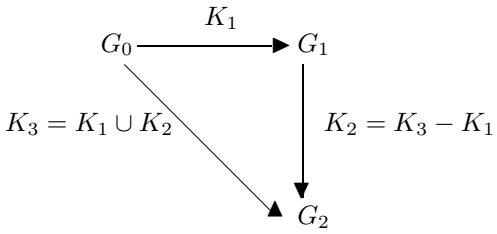


FIG. 4.37 – Diagramme de commutativité des noyeaux de contraction

Pour une carte combinatoire initiale $G_0 = (\mathcal{B}, \sigma, \alpha)$ et n noyaux successifs $K_1, \ldots, K_n$, le noyau équivalent permettant de passer directement de la carte G_0 à la carte G_i noté $K_{0,i}$ se déduit de noyaux successifs par (Théorème 3) :

$$\forall i \in \{1, \ldots, n\} \quad K_{0,i} = \bigcup_{k=1}^{i} K_k.$$

Notons que l'ensemble des brins survivants de niveau i est défini par $\mathcal{BS}_i = \mathcal{B} - K_{0,i}$. Nous avons donc la double série d'inclusions suivantes :

$$\begin{array}{ccccc} K_{0,1} & \subset & K_{0,2} & \subset \cdots \subset & K_{0,n} \\ \mathcal{BS}_n & \subset & \mathcal{BS}_{n-1} & \subset \cdots \subset & \mathcal{B}. \end{array}$$

Ces relations entre les noyaux induisent des relations entre les chemins de connexion définis par ceux-ci. Ainsi, soient $K_1, \ldots, K_n$ une séquence de noyaux successifs et un brin $b \in \mathcal{BS}_i$. Le chemin de connexion $CW_{0,i}(b)$, défini dans G_0 par b et par le noyau de contraction $K_{0,i}$ vérifie (Définition 18) :

– Si $K_1, \ldots, K_n$ sont des noyaux de contraction

$$CW_{0,i}(b) = b.\varphi(b) \ldots \varphi^{n-1}(b) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \varphi^k(b) \in \mathcal{BS}_i\};$$

– Si $K_1, \ldots, K_n$ sont des noyaux de suppression

$$CW_{0,i}(b) = b.\sigma(b) \ldots \sigma^{n-1}(b) \text{ avec } n = Min\{k \in \mathbb{N}^* \mid \sigma^k(b) \in \mathcal{BS}_i\}.$$

Pour un niveau i et un brin $b \in \mathcal{BS}_i$, nous avons pour tout $j < i$, $\mathcal{BS}_i \subset \mathcal{BS}_j$. La construction de $CW_{0,i}(b)$ réclamera donc plus d'itérations de la permutation φ ou σ que la construction de $CW_{0,j}(b)$:

$$\forall (i,j) \in \{1, \ldots, n\}^2, i > j \; \forall d \in \mathcal{BS}_i \quad CW_{0,j}(b) \subset CW_{0,i}(b).$$

Plus précisément, si $CW_i(b) = b_1 \ldots . b_p$ dénote le chemin de connexion défini dans G_i par b et K_i, nous avons [36] :

$$\forall i \in \{1, \ldots, n-1\}, \forall b \in \mathcal{BS}_{i+1} \quad CW_{0,i+1}(b) = CW_{0,i}(b_1) \ldots . CW_{0,i}(b_p).$$

Un chemin de connexion construisant la carte réduite de niveau $i+1$ à partir du niveau 0 est donc défini par une concaténation de chemins de connexion permettant de passer du niveau 0 au niveau i.

Toute carte de niveau i peut donc être retrouvée directement à partir du niveau 0 en utilisant les algorithmes représentés sur la figure 4.32 et le concept de noyau équivalent.

Ces résultats permettent de décomposer un noyau important en sous noyaux ou au contraire de grouper plusieurs noyaux consécutifs de façon à ne former qu'un seul noyau équivalent. Cette propriété nous permet de rattacher le concept de noyau au processus de décimation défini par Meer, Montanvert et Jolion [142, 146, 114, 113] qui construit un ensemble d'arêtes connectant chaque sommet non survivant à un sommet survivant. L'ensemble de ces sommets et arêtes forme une forêt recouvrante du graphe initial composée d'arbres de profondeur 1. Le théorème 3 nous assure que la décomposition d'un noyau en une série de forêts composée d'arbres de profondeur 1 fournira le même résultat que le noyau initial. Inversement, la suppression d'un niveau de la pyramide ne présentant pas d'intérêt pour l'application peut s'interpréter comme la composition de deux noyaux.

4.6.5 Séquences de connexion

Les résultats obtenus dans la section 4.6.4 sont définis dans le cadre d'une pyramide construite à partir de noyaux de même type. On peut donc partir d'une carte des régions et effectuer une série de contractions ou partir d'une carte des frontières et construire la pyramide à partir de noyaux de suppression. La carte obtenue au sommet de la pyramide peut alors être simplifiée à l'aide d'un noyau de suppression ou de contraction en fonction du type de carte utilisée. Un tel schéma de construction est intéressant si l'on s'intéresse uniquement au sommet de la pyramide. En revanche si l'on s'intéresse à l'ensemble des partitions contenues dans celle-ci, il est nécessaire d'alterner les étapes de fusion de régions avec des étapes de simplification afin d'avoir une structure de données minimale à chaque niveau de la pyramide. De plus, même dans le cas où l'on ne s'intéresse qu'au sommet de la pyramide, des étapes de simplification de la partition opérées à des niveaux intermédiaires peuvent diminuer la complexité des cartes manipulées et donc accélérer les temps de calcul.

Pour un niveau d'une pyramide construite à partir d'un seul type de noyau, l'ensemble des brins non survivants peut être structuré en un ensemble de chemins de connexion en utilisant le noyau de contraction ou de suppression équivalent (Théorème 3). Cette structuration est ensuite utilisée pour calculer les cartes réduites (équation 4.12). Malheureusement, le théorème n'est défini que pour deux noyaux de même type et la structuration des brins non survivants en chemins de connexion n'a donc plus de sens lorsque la pyramide est définie simultanément à partir de noyaux de contraction et de suppression. Il nous faut donc étendre la notion de chemin de connexion afin d'obtenir un concept équivalent dans le cadre d'une pyramide définie à partir des nos deux opérations de base.

Considérons une pyramide de cartes combinatoires $G_0, \ldots, G_n$ définie à partir des noyaux $K_1 \ldots, K_n$ de type quelconque. Intuitivement, un chemin de connexion $CW_i(b)$ correspond à la séquence de brins que nous devons traverser dans G_{i-1} pour connecter b à $\varphi_i(b)$ si K_i est un noyau de contraction et à $\sigma_i(b)$ si K_i est un noyau de suppression (équation 4.12). Considérons à présent la séquence de brins que nous devons traverser dans la carte de base G_0 pour connecter b à $\varphi_i(b)$ ou $\sigma_i(b)$ selon que K_i est un noyau de contraction ou de suppression. Une telle séquence est appelée une *Séquence de connexion* et est définie de la façon suivante :

Définition 21 Séquences de connexion
Soient une carte combinatoire $G_0 = (\mathcal{B}, \sigma, \alpha)$ et une séquence de noyaux de contraction ou de suppression $K_1 \ldots, K_n$. Les séquences de connexion (SC) sont définies par la construction récursive suivante :

$$\forall b \in \mathcal{B} \quad SC_0(b) = b.$$

Pour chaque niveau $i \in \{1, \ldots, n\}$ et pour chaque $b \in \mathcal{BS}_i$

- *Si K_i et K_{i-1} ont le même type :*

$$SC_i(b) = SC_{i-1}(b_1) \ldots . SC_{i-1}(b_p);$$

- *Si K_i et K_{i-1} ont des types différents :*

$$SC_i(b) = b_1 . SC^*_{i-1}(\alpha(b_1)) \ldots . b_p . SC^*_{i-1}(\alpha(b_p)).$$

Où $(b_1, \ldots, b_p)$ est égal à $CW_i(b)$ et $SC^*_i(b)$ désigne la séquence de connexion $SC_i(b)$ sans son premier brin. Les noyaux $K_0 = \emptyset$ et K_1 ont le même type par convention.

Notons qu'un chemin de connexion défini sur G_0 par le noyau K_1 connecte dans G_0 un brin survivant b à $\sigma_1(b)$ ou $\varphi_1(b)$ selon le type de K_1. Les séquences de connexion peuvent donc se concevoir comme une extension des chemins de connexion à plusieurs noyaux de types différents. Effectivement, les séquences de connexion vérifient des propriétés similaires à celles des chemins de connexion. On a notamment :

Propriété 2 *Soit une pyramide combinatoire définie par n noyaux successifs K_1, ...,K_n :*

1. *Les séquences de connexion au niveau 1 de la pyramide sont égales aux chemins de connexion*

$$\forall b \in \mathcal{BS}_1 \quad SC_1(b) = CW_1(b).$$

2. *Le premier brin d'une séquence de connexion définie par un brin b est b*

$$\forall i \in \{0, \ldots, n\} \quad \forall b \in \mathcal{BS}_i \quad SC_i(b) = b.b_1 \ldots . b_p.$$

3. Une séquence de connexion définie au niveau i et privée de son premier brin est uniquement composée de brins contractés ou supprimés aux niveaux précédents

$$\forall i \in \{0, \ldots, n\} \quad \forall b \in \mathcal{BS}_i \quad \mathcal{SC}_i^*(b) \subset \bigcup_{j=0}^{i} K_k.$$

Preuve:
Voir [37] propositions 11, 12 et 13. □

Le fait que les séquences de connexion coïncident avec les chemins de connexion de niveau 1 confirme qu'elles correspondent bien à une extension des chemins de connexion. On peut de plus montrer [37] que si tous les noyaux ont le même type, l'on à :

$$\forall i \in \{0, \ldots, n\} \quad \forall b \in \mathcal{BS}_i \quad \mathcal{SC}_i(b) = CW_{0,i}(b)$$

où $CW_{0,i}(b)$ est le chemin de connexion défini par le noyau équivalent $K_{0,i}$ (section 4.6.3).

Les séquences de connexion correspondent donc dans ce cas aux chemins de connexion associés aux noyaux équivalents. La différence majeure entre les deux concepts est que les séquences de connexion restent définies lorsque tous les noyaux n'ont pas le même type.

Les propriétés 2.2 et 2.3 sont des propriétés triviales dans le cadre des chemins de connexion qui sont également vérifiées par les séquences de connexion.

Afin de montrer l'intuition conduisant à la Définition 21, considérons un noyau de contraction K_i. Si K_{i+1} est également un noyau de contraction, le chemin de connexion $CW_{i+1}(b) = b.b_1 \ldots .b_p$ connecte dans G_i le brin b à $\varphi_{i+1}(b)$. Chaque brin de $CW_{i+1}(b)$ étant relié au suivant par la relation (Définition 18) :

$$b_1 = \varphi_i(b) \text{ et } \forall j \in \{1, \ldots, p-1\}\ b_{j+1} = \varphi_i(b_j).$$

Puisque K_i est un noyau de contraction, $\mathcal{SC}_i(b_j)$ connecte par hypothèse b_j à $\varphi_i(b_j)$ dans G_0 et ce pour tout j dans $\{1, \ldots, p-1\}$. La connexion de b à $\varphi_{i+1}(b)$ se fait donc en concaténant les séquences de connexion et nous avons :

$$\mathcal{SC}_{i+1}(b) = \mathcal{SC}_i(b).\mathcal{SC}_i(b_1).\ldots.\mathcal{SC}_i(b_p).$$

De même, si K_{i+1} est un noyau de suppression, $CW_{i+1}(b)$ connecte dans G_i b à $\sigma_{i+1}(b)$ et chaque brin du chemin est relié au suivant par :

$$b_1 = \sigma_i(b) \text{ et } \forall j \in \{1, \ldots, p-1\}\ b_{j+1} = \sigma_i(b_j). \tag{4.16}$$

Le noyau K_{i+1} étant un noyau de suppression, $\mathcal{SC}_{i+1}(b)$ doit connecter b à $\sigma_{i+1}(b)$ dans G_0. Le noyau K_i étant un noyau de contraction, $\mathcal{SC}_i(b_j)$ connecte b_j à $\varphi_i(b_j)$ dans G_0 pour tout j dans $\{1, \ldots, p-1\}$. Il nous faut donc trouver une séquence de brins connectant b_j à $\sigma_i(b_j)$ dans G_0 afin de pouvoir utiliser l'équation 4.16. La séquence $\mathcal{SC}_i(\alpha(b_j))$ connecte $\alpha(b_j)$ à $\varphi_i(\alpha(b_j)) = \sigma_i(b_j)$ dans G_0. La connexion entre b_j et $\sigma_i(b_j)$ est donc réalisée par la séquence de brins $b_j\mathcal{SC}_i^*(\alpha(b_j))$. Une telle séquence est simplement égale à la séquence $\mathcal{SC}_i(\alpha(b_j)$ dans laquelle nous avons substitué b_j à $\alpha(b_j)$ (Propriété 2.2). La séquence $\mathcal{SC}_{i+1}(b)$ est donc égale à :

$$\mathcal{SC}_{i+1}(b) = b.\mathcal{SC}_i^*(\alpha(b)).b_1.\mathcal{SC}_i^*(\alpha(b_1)).\ldots.b_p.\mathcal{SC}_i^*(\alpha(b_p)).$$

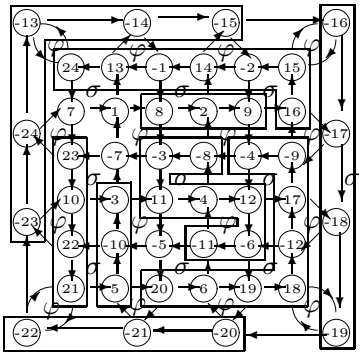


FIG. 4.38 – Séquences de connexion définies par K_1 et K_2.

La figure 4.38 représente les séquences de connexion définies par l'application successive de K_1 et K_2. Si nous considérons, par exemple, le brin $20 \in \mathcal{BS}_2$. Nous avons (figure 4.36) :

$$CW_2(20) = 20.19.18.17.-9 \text{ avec } \begin{cases} CW_1(-20) &= -20.6 \\ CW_1(-19) &= -19 \\ CW_1(-18) &= -18.-12 \\ CW_1(-17) &= -17 \\ CW_1(9) &= 9.-4. \end{cases}$$

Puisque $SC_1(b) = CW_1(b)$ pour tout brin de $\mathcal{BS}_1$ (Propriété 2.1), nous avons :

$$\begin{array}{rcl} SC_2(20) &=& 20.CW_1^*(-20).19.CW_1^*(-19).18.CW_1^*(-18).17.CW_1^*(-17).-9.CW_1^*(9) \\ &=& 20.6.19.18.-12.17.-9.-4 \end{array}$$

où $CW_1^*(b)$ désigne le chemin de connexion $CW_1(b)$ sans son premier brin b.

4.6.6 Reconstruction d'un niveau d'une pyramide

Nous avons vu dans la section 4.6.5 que les séquences de connexion coïncident avec les chemins de connexion définis par les noyaux équivalents lorsque tous les noyaux ont le même type (section 4.6.4). Les chemins de connexion définis par des noyaux équivalents permettent de calculer un niveau de la pyramide directement à partir de sa base en utilisant l'équation 4.12. Les séquences de connexion vérifient une propriété similaire à l'équation 4.12. On a en effet :

Théorème 4 *Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et une séquence de noyaux de contraction ou de suppression successifs $K_1, \ldots, K_n$. La séquence de connection d'un brin $b \in \mathcal{BS}_i$ définie au niveau $i \in \{1, \ldots, n\}$:*

$$SC_i(b) = b_1.\ldots.b_p$$

vérifie :

– *Si K_i est un noyau de contraction :*

$$\begin{array}{l} \textit{Si } p = 1 \\ \qquad \varphi_i(b) = \varphi(b) \\ \textit{sinon} \\ \qquad \varphi_i(b) = \begin{cases} \varphi(b_p) & \textit{Si } b_p \textit{ est contracté} \\ \sigma(b_p) & \textit{Si } b_p \textit{ est supprimé.} \end{cases} \end{array} ;$$

– *Si K_i est un noyau de suppression :*

$$\begin{array}{l} \textit{Si } p = 1 \\ \qquad \sigma_i(b) = \sigma(b) \\ \textit{sinon} \\ \qquad \sigma_i(b) = \begin{cases} \varphi(b_p) & \textit{Si } b_p \textit{ est contracté} \\ \sigma(b_p) & \textit{Si } b_p \textit{ est supprimé.} \end{cases} \end{array} .$$

Preuve:
Voir [37] propriétés 15 et 17. □

Notons que si $p = 1$, la séquence de connexion $SC_i(b)$ est réduite à $SC_i(b) = b$ (Propriété 2.2). Dans ce cas, les affectations du Théorème 4 correspondent exactement à celle de l'équation 4.12. Si $p > 1$, le brin b_p est forcément un brin contracté ou supprimé à un niveau précédent (Propriété 2.3). Un test supplémentaire doit donc être effectué en fonction de l'opération appliquée à ce brin.

Le Théorème 4 permet *a priori* de calculer n'importe quel niveau de la pyramide à partir de sa base. Toutefois, la définition des séquences de connexion fournie par la Définition 21 est basée sur une construction récursive, les séquences de connexion de niveau i étant construites à partir des séquences de niveau $i-1$ et des chemins de connexion de niveau i. Si nous utilisons le mécanisme de construction de la Définition 21, le calcul de G_i par le Théorème 4 nécessite de construire la pyramide au moins jusqu'au niveau $i-1$ pour pouvoir construire les séquences de connexion de niveau i. Une telle méthode bien que théoriquement possible est parfaitement inefficace puisque la carte G_i peut se déduire de G_{i-1} et K_i en utilisant les chemins de connexion. Les chemins de connexion sont plus courts que les séquences définies au même niveau ([37], Proposition 14). Un algorithme basé sur les chemins de connexion de G_{i-1} sera donc bien plus efficace qu'un algorithme basé sur les séquences de connexion définies dans G_0.

Il nous faut donc définir un mécanisme de construction des séquences de connexion qui ne nécessite pas une construction explicite de tous les niveaux de la pyramide. Un tel schéma de construction nous est fourni par le théorème suivant :

Théorème 5 *Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et une séquence de noyaux de contraction ou de suppression successifs $K_1, \ldots, K_n$. Pour toute séquence de connexion $SC_i(b) = b.b_1 \ldots b_p$ définie par $b \in \mathcal{BS}_i$ avec $i \in \{1, \ldots, n\}$ nous avons si $p > 1$:*

$$b_1 = \begin{cases} \varphi(b) & \textit{Si } K_i \textit{ est un noyau de contraction} \\ \sigma(b) & \textit{Si } K_i \textit{ est un noyau de suppression} \end{cases}$$

et

$$\forall j \in \{2, \ldots, p\} \quad b_j = \begin{cases} \varphi(b_{j-1}) & \textit{Si } b_{j-1} \textit{ est contracté} \\ \sigma(b_{j-1}) & \textit{Si } b_{j-1} \textit{ est supprimé.} \end{cases}$$

Le Théorème 5 permet donc de parcourir une séquence de connexion si nous connaissons son brin initial ainsi que l'opération qui a réduit chacun des brins de la séquence. Ceci suggère un codage de la pyramide à l'aide de deux fonctions :

Définition 22 Codage implicite d'une pyramide
Soit une pyramide définie par une carte initiale $G_0 = (\mathcal{B}, \sigma, \alpha)$ et n noyaux successifs $K_1, \ldots, K_n$. Un codage implicite de la pyramide est un triplet composé de la carte initiale G_0 et de deux fonctions **état** *et niveau telles que :*

- *la fonction état est définie de $\{1, \ldots, n\}$ dans les deux états binaires $\{$Contracté,Supprimé$\}$ et code le type de chaque noyau.*

$$\textbf{état} \left(\begin{array}{ccc} \{1, \ldots, n\} & \rightarrow & \{\textit{Contracté}, \textit{Supprimé}\} \\ i & \mapsto & \left\{ \begin{array}{l} \textit{Contracté si } K_i \textit{ est un noyau de contraction,} \\ \textit{Supprimé sinon ;} \end{array} \right. \end{array} \right.$$

- *La fonction niveau code pour chaque brin de la carte initiale $G = (\mathcal{B}, \sigma, \alpha)$ le niveau maximal où il survit.*

$$\begin{array}{rccl} \textbf{niveau} : & \mathcal{B} & \rightarrow & \{1, \ldots, n+1\} \\ & b & \mapsto & \max\{i \in \{1, \ldots, n+1\} \mid b \in \mathcal{BS}_{i-1}\}. \end{array}$$

Notons qu'un brin survivant jusqu'au niveau n appartient à $\mathcal{BS}_n$ et a donc pour niveau $n+1$. De plus, les noyaux étant symétriques la fonction *niveau* doit satisfaire :

$$\forall b \in \mathcal{B} \quad \textbf{niveau}(\alpha(b)) = \textbf{niveau}(b). \tag{4.17}$$

Cette fonction peut donc être stockée uniquement sur la moitié des brins. Si l'involution α est implicitement codée par le signe, on peut par exemple ne stocker que les niveaux des brins positifs.

Les noyaux et l'ensemble des brins survivants aux différents niveaux peuvent être retrouvés par le codage implicite à l'aide de l'équation suivante

$$\forall i \in \{1, \ldots, n\} \left\{ \begin{array}{rcl} K_i & = & \{b \in \mathcal{B} \mid \textbf{niveau}(b) = i\} \\ \mathcal{BS}_i & = & \{b \in \mathcal{B} \mid \textbf{niveau}(b) > i\}. \end{array} \right. \tag{4.18}$$

De plus, étant donné un brin $b \in \mathcal{BS}_i$ les fonctions *niveau* et **état** nous permettent de réécrire le parcourt de la séquence de connexion $\mathcal{SC}_i(b) = b.b_1 \ldots .b_p$ définie par le Théorème 5 de la façon suivante :

$$\begin{array}{rcl} b_1 & = & \left\{ \begin{array}{ll} \varphi(b) & \text{Si } \textbf{état}(\text{i})=\text{Contracté} \\ \sigma(b) & \text{Si } \textbf{état}(\text{i})=\text{Supprimé} \end{array} \right. \\ & \text{et} & \\ \forall j \in \{2, \ldots, p\} \quad b_j & = & \left\{ \begin{array}{ll} \varphi(b_{j-1}) & \text{Si } \textbf{état}(\textbf{niveau}(b_{j-1})) = \text{Contracté} \\ \sigma(b_{j-1}) & \text{Si } \textbf{état}(\textbf{niveau}(b_{j-1})) = \text{Supprimé} \end{array} \right. \end{array} \tag{4.19}$$

Soit une séquence de connexion $\mathcal{SC}_i(b) = b.b_1 \ldots .b_p$. Si nous notons b_{p+1} le brin égal à $\varphi(b_p)$ si b_p et contracté et $\sigma(b_p)$ si b_p est supprimé, b_{p+1} est égal à $\varphi_i(b)$ ou $\sigma_i(b)$ en fonction du type de K_i (Théorème 4). Nous avons donc dans les deux cas $b_{p+1} \in \mathcal{BS}_i$ et $\textbf{niveau}(b_{p+1}) > i$ (équation 4.18). Cette dernière remarque nous permet de caractériser le dernier brin d'une séquence de connexion comme celui dont le successeur défini par l'équation 4.19 a un niveau strictement supérieur à i.

```
1   construire_σ_i(  carte G_0,
2                    fonction niveau,
3                    fonction état,
4                    entier i)
5   {
6       Pour tout brin b ∈ BS_i
7       {
8           Si état(i) == Contracté
9               σ_i(b) = survivant(G_0, i, α(b))
10          sinon
11              σ_i(b) = survivant(G_0, i, b)
12      }
13  }
```

(a) Définition de la permutation σ_i

```
1   brin survivant( entier i,
2                   brin b)
3   {
4       b'=b ;
5       tant que niveau(b') ≤ niveau(b)
6       {
7           Si état(niveau(b')) == Contracté
8               b' = φ(b')
9           sinon
10              b' = σ(b')
11      }
12      retourner b'
13  }
```

(b) Contraction

FIG. 4.39 – Algorithmes calculant la carte réduite de niveau i à partir du codage implicite de la pyramide.

Les considérations précédentes permettent de définir le couple d'algorithmes représentés sur la figure 4.39. L'algorithme `construire_`σ_i (figure 4.39(a)) parcourt chaque brin survivant $b \in \mathcal{BS}_i$ et appelle pour celui-ci la fonction `survivant`. Cette fonction parcourt la séquence de connexion du brin survivant passé en second paramètre en utilisant l'équation 4.19 et renvoie le premier brin survivant de niveau i rencontré. Si K_i est un noyau de contraction, la fonction `survivant` est appelée avec les paramètres i et $\alpha(b)$ et parcourt $\mathcal{SC}_i(\alpha(b))$. On a donc (Théorème 4 et Algorithme 4.39(a) ligne 6)

$$\varphi_i(\alpha(b)) = \sigma_i(b) = \varphi(b_p)$$

où b_p est le dernier brin de $\mathcal{SC}_i(\alpha(b))$.

De même, si K_i est un noyau de suppression, le successeur du dernier brin de $\mathcal{SC}_i(b)$ est égal à $\sigma_i(b)$ (Théorème 4 et Algorithme 4.39(a) ligne 8)

Nous avons montré que pour tout niveau $i \in \{1, \ldots, n\}$ l'ensemble des séquences de connexion définies à ce niveau forme une partition de l'ensemble des brins de la carte initiale $G_0 = (\mathcal{B}, \sigma, \alpha)$:

$$\forall i \in \{1, \ldots, n\} \quad \forall b \in \mathcal{B}\ \exists! b' \in \mathcal{BS}_i \mid b \in \mathcal{SC}_i(b'). \tag{4.20}$$

Notez que cette dernière équation est très similaire à l'équation 4.13 définissant la propriété équivalente pour les chemins de connexion.

La combinaison des algorithmes `construire_`σ_i et `survivant` induit un parcours de l'ensemble des séquences de connexion définies au niveau i. Ces séquences formant une partition de l'ensemble des brins initiaux (équation 4.20), la complexité de ces algorithmes est égale à $\mathcal{O}(|\mathcal{B}|)$. D'un autre coté, la construction du niveau i de la pyramide en utilisant une construction explicite de tous les niveaux réclame de parcourir :

$$|\mathcal{B}| \sum_{j=0}^{i} \frac{1}{r^j} \approx |\mathcal{B}| \frac{r}{r-1}$$

brins, où r représente le facteur de réduction entre deux niveaux consécutifs de la pyramide.

Nous obtenons pour $k = 2$ une complexité approximativement égale à $\mathcal{O}(2||\mathcal{B}|)$.

4.6.7 Reconstruction d'une pyramide

Les algorithmes décrits dans la section 4.6.6 (figure 4.39) n'utilisent pas la totalité des informations fournies par le codage implicite d'une pyramide. En effet, ces algorithmes parcourent l'ensemble des brins d'une séquence de connexion en utilisant pour chaque brin le type du noyau qui l'a fait disparaître. L'information non utilisée est donc le niveau de la pyramide auquel le noyau de contraction ou de suppression a été appliqué.

Nous allons donner dans cette section deux algorithmes permettant de calculer l'ensemble des niveaux d'une pyramide en parcourant une seule fois l'ensemble des brins du niveau de base. Ces algorithmes utilisent l'ensemble des informations fournies par le codage implicite de la pyramide. Ce codage implicite est transformé en un codage explicite par l'initialisation de la structure de données codant la fonction Σ :

Définition 23 Fonction Σ

Soit une pyramide de n niveaux codée implicitement, par les fonctions niveau et **état**. *L'application Σ est définie par :*

$$\begin{array}{rccc} \Sigma : & \{1,\ldots,n\} \times \mathcal{B} & \rightarrow & \mathcal{B} \\ & (i,b) & \mapsto & \sigma_i(b). \end{array}$$

D'un point de vue formel, la fonction Σ permet d'uniformiser les affectations au σ ou φ successeur d'un brin de niveau i à l'aide de l'équation suivante :

$$\forall i \in \{1,\ldots,n\}\ \forall b \in \mathcal{BS}_i \left\{ \begin{array}{rcl} \sigma_i(b) & = & \Sigma(i,b) \\ \varphi_i(b) & = & \Sigma(i,\alpha(b)). \end{array} \right. \tag{4.21}$$

D'un point de vue plus pratique, la fonction Σ a également l'avantage, comme nous le verrons par la suite, de pouvoir être implémentée à l'aide d'une structure de données relativement simple.

Afin d'éviter de donner trop abruptement les définitions et résultats indispensables aux algorithmes construisant la pyramide, nous allons illustrer les principaux concepts à travers différents exemples. Considérons la pyramide représentée sur la figure 4.40 construite à partir d'une carte des régions $G_0 = (\mathcal{B}, \sigma_0, \alpha)$ codant une grille 3×3. Notez que l'adjacence entre les pixels du bord et le sommet infini n'est pas codé dans la carte. Ce type de codage, bien que non optimum (on s'interdit de simplifier le bord de l'image) permet de travailler sur des exemples plus simples.

Un exemple de codage de la fonction Σ par un tableau de tableaux codant la Pyramide représentée sur la figure 4.40 est donné Table 4.2. Notez que dans ce cas la fonction *niveau* code simultanément le nombre de niveaux où apparaît un brin et la taille du tableau associé à ce brin dans la structure codant Σ.

On peut également noter que la pyramide est constituée à partir d'une alternance de noyaux de contraction et de suppression. Une telle séquence est appelée une *Séquence de noyaux alternés*. Nous avons montré [39] que toute séquence de noyaux peut se ramener à une séquence de noyaux alternés. Ceci revient à dire que la construction des séquences de connexion reste valide si l'on insère un noyau vide entre deux noyaux successifs de même type. L'utilisation de noyaux alternés permet de simplifier la définition des séquences de connexion (Définitions 21) qui se ramène à :

$$\forall i \in \{1,\ldots,n\} \forall b \in \mathcal{BS}_i \quad \mathcal{SC}_i(b) = b.\mathcal{SC}_{i-1}(\alpha(b)).b_1.\mathcal{SC}_{i-1}(\alpha(b_1)).\ldots.b_p.\mathcal{SC}_{i-1}(\alpha(b_p)).$$

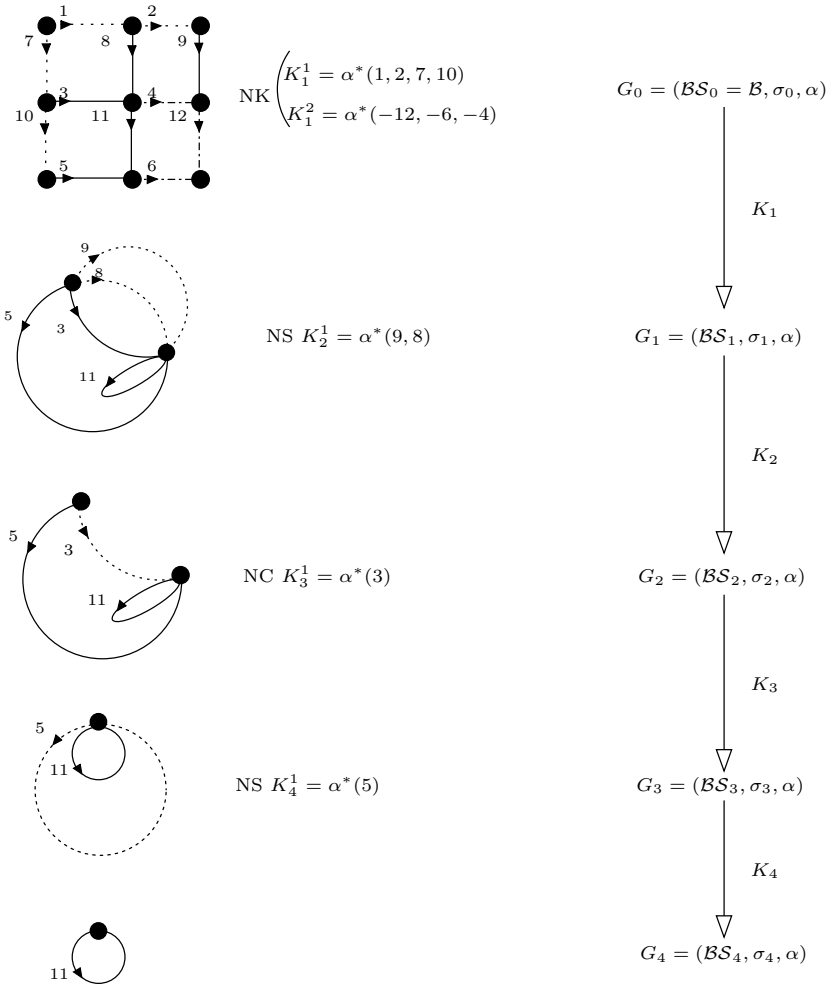


FIG. 4.40 – *Construction d'une pyramide sur une grille* 3×3. *Les noyaux d'indice impairs correspondent à des noyaux de contraction (NC) alors que les indices pairs correspondent à des noyaux de suppression (NS).*

b		$\Sigma(i,b)$				
brin	niveau	0	1	2	3	4
1	1	7				
-1	1	8				
2	1	-1				
-2	1	9				
3	3	-7	8	5		
-3	3	11	11	11		
4	1	-8				
-4	1	12				
5	4	-10	3	3	11	
-5	4	6	-9	-3	5	
6	1	-11				
-6	1	-12				
7	1	1				
-7	1	10				
8	2	2	9			
-8	2	-3	-3			
9	2	-2	5			
-9	2	-4	-8			
10	1	3				
-10	1	5				
11	5	4	-11	-11	-11	-11
-11	5	-5	-5	-5	-5	11
12	1	-9				
-12	1	-6				

TAB. 4.2 – Une implémentation possible de la fonction Σ. La pyramide codée dans cet exemple est représentée Figure 4.40. Pour plus clarté, nous avons regroupé dans cette figure les brins positifs avec leurs α successeurs. Notez qu'une implémentation réelle peut nécessiter un ré-ordonnancement des lignes en commençant par les brins négatifs : $-12 \ldots, -1, 1, \ldots, 12$.

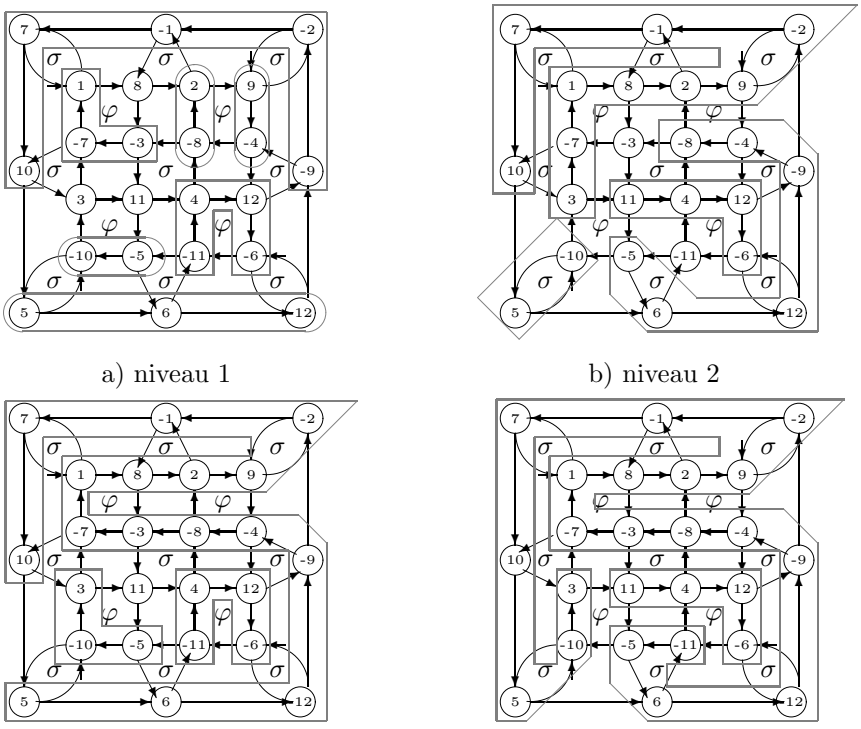


FIG. 4.41 – Définition récursive des séquences de connexion

Les séquences de connexion de la pyramide représentée sur la figure 4.40 sont illustrées sur la figure 4.41.

4.6.7.a Détermination des successeurs d'un brin survivant

Considérons l'arête $\alpha^*(5)$ sur les figures 4.40 et 4.41. Les noyaux K_1 et K_4 étant des noyaux de contraction, les chemins de connexion définis par ces noyaux seront des séquences de φ-successeurs du brin initial. De même, le noyau K_2 étant un noyau de suppression, ses chemins de connexion seront définis comme des séquences de σ successeurs du brin initial. L'arête $\alpha^*(5)$ étant supprimée au niveau 4, les brins de cette arête définissent des séquences de connexion jusqu'au niveau 3.On a donc d'après la définition des séquences de connexion :

$$SC_1(5) = 5\varphi_0(5)\dots \tag{4.22}$$
$$SC_2(-5) = -5SC_1^*(5)\sigma_1(-5).... \text{ et} \tag{4.23}$$
$$SC_3(5) = 5SC_2^*(-5).\varphi_2(5).... \quad . \tag{4.24}$$

Si nous insérons les équations 4.22 et 4.23 dans l'équation 4.24 nous obtenons :

$$SC_3(5) = 5\varphi_0(5)....\varphi_1(5).....\varphi_2(5)\dots \quad . \tag{4.25}$$

où nous avons substitué $\varphi_1(5)$ à $\sigma_1(-5)$ dans l'équation 4.23.

La traversée de $SC_3(5)$ nous permet donc de collecter des renseignements sur les φ successeurs de 5 du niveau 0 au niveau 2. Notons également que le successeur du dernier brin de $SC_3(5)$ doit dans ce cas être égal à $\varphi_3(5)$ (Théorème 4).

La récupération de ces informations suppose toutefois que nous soyons capables de retrouver les informations pertinentes à l'intérieur d'une séquence de connexion. Il s'agit donc d'être capable de déterminer $\varphi_1(5)$ ou $\varphi_2(5)$ lors d'un parcourt de la séquence de brins définie par l'équation 4.25. Nous utilisons pour cela la propriété 2.3 qui nous indique que le premier brin d'une séquence de connexion de niveau i est le seul à survivre au niveau i. Autrement dit, selon l'équation 4.23, $\sigma_1(-5)$ est contracté au niveau 2 par définition des chemins de connexion et a donc un niveau égal à 2 (équation 4.18). Inversement, tous les brins de $SC_1^*(5)$ ont été contractés au niveau 1 et ont donc un niveau égal à 1. Le brin $\sigma_1(-5)$ est donc le premier brin de niveau 2 que l'on rencontre en parcourant $SC_2^*(-5)$. De même on peut montrer que $\varphi_2(5)$ est le premier brin de niveau 3 que l'on rencontre en parcourant $SC_3^*(5)$. Le parcours de $SC_3^*(5)$ va donc débuter par le brin $\varphi_0(5)$ (Théorème 5) puis, après avoir traversé un nombre quelconque de brins de niveaux 1, le premier brin de niveau 2 rencontré sera égal à $\sigma_1(-5) = \varphi_1(5)$. Enfin, après avoir "passé" $\varphi_1(5)$, le premier brin de niveau 3 rencontré sera égal à $\varphi_2(5)$.

La formalisation de ces résultats nécessite donc de définir le niveau maximum que l'on peut rencontrer dans une séquence de connexion et l'index du premier brin dont le niveau est supérieur à une valeur donnée. Cette tache est accomplie par la définition suivante où ϵ désigne une séquence de brins vide :

Définition 24 *Fonctions niveau max et index*

Soient une pyramide de n niveaux codée implicitement par les fonctions niveau et **état** *et un brin $b \in \mathcal{BS}_i$ tel que $SC_i^*(b) \neq \epsilon$, la fonction L_i affecte chaque brin au niveau maximum contenu dans sa séquence de connexion.*

$$\begin{array}{rccl} L_i : & \mathcal{BS}_i & \rightarrow & \{1,\dots,i\} \\ & b & \mapsto & \max_{b' \in SC_i^*(b)}\{\textbf{niveau}(b')\}. \end{array}$$

La fonction d'index $\mathbf{l}_{i,b}$*, affecte chaque entier* j *compris entre* 1 *et* $L_i(b)$ *à l'index du premier brin dans* $\mathcal{SC}_i^*(b)$ *dont le niveau est supérieur à* j*.*

$$\begin{array}{rccl} \mathbf{l}_{i,b}: & \{1,\ldots,L_i(b)\} & \rightarrow & \mathbb{N}^* \\ & j & \mapsto & \min\{k \in \{1,\ldots,p\} \mid \mathbf{niveau}(b_k) \geq j\} \end{array}$$

avec $\mathcal{SC}_i(b) = b.b_1,\ldots.b_p$*,* $p > 1$*.*

Le fait que $L_i(b)$ varie entre 1 et i pour tout b dans $\mathcal{BS}_i$ s'explique par la Propriété 2.3. En effet :

$$\forall i \in \{1,\ldots,n\}\ \forall b \in \mathcal{BS}_i \quad \mathcal{SC}_i^*(b) \subset \bigcup_{j=1}^{i} K_j \Rightarrow L_i(b) \leq i.$$

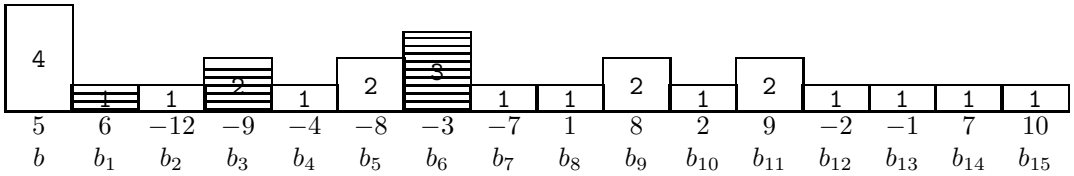


FIG. 4.42 – La séquence de connexion $\mathcal{SC}_3(5)$. Le niveau de chaque brin est indiqué dans la barre qui le représente

La figure 4.42 représente la séquence de connexion $\mathcal{SC}_3(5)$ (figure 4.41) où les premiers brins de niveau supérieur à 1, 2 et 3 sont hachurés de motifs horizontaux. Le niveau maximal rencontré dans la séquence étant égal à 3, nous avons $L_3(5) = 3$. De plus, nous avons, en prenant l'indice des brins hachurés dans la séquence :

$$\mathbf{l}_{3,5}(1) = 1;\ \mathbf{l}_{3,5}(2) = 3;\ \mathbf{l}_{3,5}(3) = 6$$

Étant données les fonctions L_i et $\mathbf{l}_{i,b}$, les résultats obtenus précédemment pour le brin 5 se généralisent à l'aide du Théorème suivant :

Théorème 6 *Soient une pyramide de* n *niveaux codée implicitement par les fonctions niveau et* **état** *et un brin* $b \in \mathcal{BS}_i$ *tel que* $\mathcal{SC}_i(b) \neq \epsilon$*, les* σ *ou* φ *successeurs de* b *du niveau* 0 *au niveau* i *sont égaux à :*

$$\forall j \in \{1,\ldots,L_i(b)\} \quad \left\{ \begin{array}{rcll} \varphi_{j-1}(b) & = & b_{li,b(j)} & \textit{si } K_i \textit{ est un noyau de contraction} \\ \sigma_{j-1}(b) & = & b_{li,b(j)} & \textit{si } K_i \textit{ est un noyau de suppression} \end{array} \right.$$

$$\forall j \in \{L_i(b),\ldots,i\} \quad \left\{ \begin{array}{rcll} \varphi_j(b) & = & \varphi_i(b) & \textit{si } K_i \textit{ est un noyau de contraction} \\ \sigma j(b) & = & \sigma_i(b) & \textit{si } K_i \textit{ est un noyau de suppression.} \end{array} \right.$$

Preuve:
Voir [41] Corollaire 2 et Théorème 3 □

Le Théorème 6 nous permet donc d'assurer que :

$$\begin{array}{rcccccc} \varphi_0(5) & = & b_{\mathbf{l}_{3,5}(1)} & = & b_1 & = & 6 \\ \varphi_1(5) & = & b_{\mathbf{l}_{3,5}(2)} & = & b_3 & = & -9 \\ \varphi_2(5) & = & b_{\mathbf{l}_{3,5}(3)} & = & b_6 & = & -3. \end{array}$$

Le brin 10 qui termine $SC_3(5)$ est contracté au niveau 1, on a donc également par le Théorème 4 $\varphi_3(5) = \varphi_0(10) = 5$.

Le second cas du Théorème 6 est intéressant lorsque $L_i(b) < i$. Dans ce cas, les φ ou σ successeurs d'un brin b ne peuvent être retrouvés jusqu'au niveau $i-1$ dans la séquence de connexion. Ceci se produit par exemple pour le brin 5 au niveau 2 (figure 4.41(b)) où l'on a : $SC_2(5) = 5.-10.$ Le brin -10 étant contracté au niveau 1, on a $L_2(5) = 1 < 2$. La première partie du Théorème 4 nous informe, dans ce cas uniquement, de la relation $\sigma_0(5) = -10$. Les σ successeurs de 5 au niveau 1 et 2 nous sont donnés par la seconde partie de ce théorème. On a en effet :

$$\sigma_1(5) = \sigma_2(5) = \varphi_0(-10) = 3.$$

Le Théorème 6 pourrait se formuler de façon imagée de la façon suivante : la séquence de connexion d'un brin b fournie des renseignements sur les σ ou φ successeurs de b jusqu'au niveau maximal atteint dans la séquence. Les informations manquantes sont fournies par le σ ou φ successeur du dernier brin de la séquence.

Un cas limite d'information manquante correspond au cas rejeté dans le Théorème 6 où $SC_i^*(b) = \epsilon$ pour un brin b donné. Dans ce cas, toute les informations sont fournies par le σ ou φ successeur de b :

Théorème 7 *Soient une pyramide de n niveaux codée implicitement par les fonctions niveau et* **état** *et un brin $b \in \mathcal{BS}_i$ tel que $SC_i(b) = \epsilon$. Les σ ou φ successeurs de b du niveau 0 au niveau i sont définis par :*

$$\forall j \in \{0, \ldots, i\} \begin{cases} \varphi_j(b) = \varphi_0(b) & \text{si } K_i \text{ est un noyau de contraction} \\ \sigma_j(b) = \sigma_0(b) & \text{si } K_i \text{ est un noyau de suppression.} \end{cases}$$

Le brin 11 vérifie au niveau 3 $SC_3(11) = 11$. On a donc par le Théorème 7 :

$$\varphi_3(11) = \varphi_2(11) = \varphi_1(11) = \varphi_0(11) = -5.$$

Intuitivement, une telle configuration s'explique par la survie conjointe jusqu'au niveau 3 des arêtes 11 et 5. En effet, le second brin d'une séquence de connexion est $\varphi_0(b)$ ou $\sigma_0(b)$ en fonction du noyau (Théorème 5). De plus, une séquence de contraction privée de son premier brin ne doit comporter que des brins contractés ou supprimés aux niveaux précédents (Propriété 2.3). Dans le cas de noyaux de contraction, la séquence de connexion de 11 sera donc bloquée immédiatement par $\varphi_0(11) = 5$ qui survit au niveau 3. On aura donc $SC_i^*(11) = \epsilon$ avec $i \in \{1, 3\}$ (figures 4.41(a) et (c)). De même, si le noyau est un noyau de suppression, la séquence de connexion de -11 sera bloquée sur $\sigma_0(-11) = \varphi_0(11) = 5$ et on aura $SC_2^*(-11) = \epsilon$ (figure 4.41(b)).

4.6.7.b Détermination des successeurs d'un brin non survivant

Les Théorèmes 6 et 7 nous permettent de parcourir l'ensemble des séquences de connexion définies à un niveau i et de calculer pour chaque brin dans $\mathcal{BS}_i$ l'ensemble de ses σ ou φ successeurs du niveau 0 au niveau i. Ceci dit, ce résultat ne nous permet pas de calculer l'ensemble des cartes réduites du niveau 0 au niveau i. En effet, une carte de niveau $j < i$ sera définie à partir des brins $\mathcal{BS}_j \supset \mathcal{BS}_i$. Il nous manquera donc les σ ou φ successeurs des brins $\mathcal{BS}_j - \mathcal{BS}_i$. Toutefois, l'ensemble des séquences de connexion de niveau i formant une partition de $\mathcal{B}$ les brins $\mathcal{BS}_j - \mathcal{BS}_i$ sont contenus dans des séquences de connexion de niveau i.

Examinons, par exemple, l'arête $\alpha^*(5)$ au niveau 4. Le noyau $K_4 = \alpha^*(5)$ est un noyau de suppression. On a donc (figure 4.40) :

$$CW_4(-11) = -11.\sigma_3(-11).\sigma_3^2(-11) = -11.-5.5.$$

On aura donc :

$$SC_4(-11) = -11.SC_3^*(11).-5.SC_3^*(5).5.SC_3^*(-5). \tag{4.26}$$

La figure 4.43 représente la séquence de connexion $SC_4(-11)$ dans laquelle nous retrouvons $SC_3^*(5)$ (brins b_2 à b_{16} figure 4.43). L'ensemble des informations que nous avons retrouvé sur $SC_3(5)$ peut donc *a priori* être retrouvé en parcourant $SC_4(-11)$. On retrouve effectivement les brins $6, -9$ et -3 respectivement égaux à $\sigma_0(-5), \sigma_1(-5)$ et $\sigma_2(-5)$. De plus, puisque l'ensemble des brins de $SC_3^*(5)$ à un niveau inférieur à 3, $5 = \sigma_3(-5)$ est le premier brin de niveau 4 que l'on rencontre après avoir passé -5 dans l'équation 4.26 (figure 4.43).

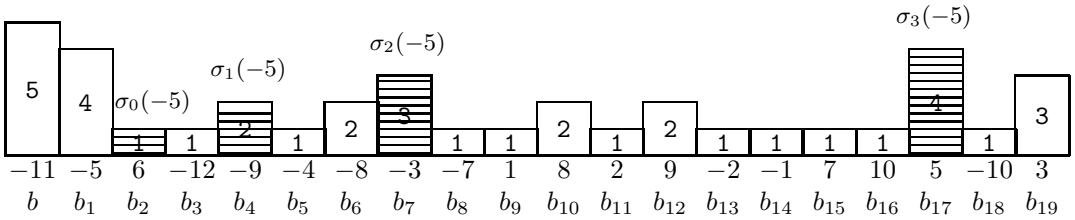


FIG. 4.43 – La séquence de connexion du brin -11 au niveau 4.

L'extension des Théorèmes 6 et 7 à des brins intérieurs à une séquence de connexion nécessite que l'on puisse identifier le brin survivant dont la séquence de connexion de niveau i contient un brin b non survivant. Il nous faut également étendre les définitions des fonctions L_i et $\mathbf{l}_{i,b}$ à des brins non survivants. Cela est réalisé à l'aide de la définition suivante :

Définition 25 *Soient un codage implicite d'une pyramide définie par une carte initiale $G = (\mathcal{B}, \sigma, \alpha)$, n noyaux et un brin $b \in \mathcal{B}$*

- *pour tout niveau $i \in \{1, \ldots, n\}$, le brin survivant de niveau i dont la séquence de connexion contient b est noté b^i ;*
- *l'index de b dans $SC_i(b^i)$ est noté $Ind_i(b)$;*
- *l'ensemble des niveaux pour lesquels b n'est ni le dernier ni le seul brin de $SC_i(b^i)$ correspond à l'ensemble des niveaux pour lesquels b a un successeur dans sa séquence de connexion et est noté $\mathcal{DS}_b$.*

$$\mathcal{DS}_b = \{i \in \{1, \ldots, n\} \mid \exists S_1, S_2,\ SC_i(b^i) = S_1 b S_2 \text{ avec } S_2 \neq \epsilon\};$$

- *pour tout brin b et tout niveau $i \in \mathcal{DS}_b$, $\mathcal{L}_i(b)$ est défini comme le minimum entre le niveau de b et le niveau maximum rencontré dans $SC_i(b^i)$ après b.*

$$\mathcal{L}_i(b) = \min(\mathbf{niveau}(b), \max_{k \in \{Ind_i(b)+1, \ldots, p\}} \{\mathbf{niveau}(b_k)\});$$

où $SC_i(b^i) = b^i.b_1 \ldots .b_p$

– *le $j^{ème}$ successeur d'un brin b à un niveau $i \in \mathcal{DS}_b$ avec $j \in \{1, \ldots, \mathcal{L}_i(b)\}$ est l'index du premier brin de niveau supérieur à j qui suit b dans $SC_i(b^i)$.*

$$\left.\begin{array}{ll} \forall b \in & \mathcal{B} \\ \forall i \in & \mathcal{DS}_b \\ \forall j \in & \{1, \ldots, \mathcal{L}_i(b)\} \end{array}\right\} \mathbf{succ}_{i,b}(j) = \min\{k \in \{Ind_i(b)+1, \ldots, p\} \mid \mathbf{niveau}(b_k) \geq j\}$$

avec $SC_i(b^i) = b_1, \ldots, b_p$.

Les notations b^i et $Ind_i(b)$ permettent simplement de noter la séquence de connexion contenant un brin non survivant et son indice dans cette séquence. On a par exemple sur la figure 4.43 $5^4 = -11$ et $Ind_4(5) = 17$. Notons toutefois que ces notations restent définies si le brin est un brin survivant, on a par exemple $5^i = 5$ pour $i \in \{1, 2, 3\}$ et $Ind_i(5) = 0$ pour les mêmes indice.

L'ensemble $\mathcal{DS}_b$ définit l'ensemble des niveaux pour lesquels un brin b a des successeurs dans la séquence de connexion qui le contient. Le brin 3 étant le dernier de $SC_4(11)$ on a par exemple $4 \notin \mathcal{DS}_3$. De même, $\mathcal{DS}_{-10} = \{3, 4\}$ (figure 4.41).

Enfin, les fonctions $\mathcal{L}_i$ et $\mathbf{succ}_{i,b}$ sont respectivement les extensions des fonctions L_i et $\mathbf{l}_{i,b}$ aux niveaux pour lesquels le brin ne survit plus. Nous avons d'ailleurs montré [41] que ces fonctions coïncident tant que le brin survit. La valeur de $\mathcal{L}_i(b)$ est majorée par $\mathbf{niveau}(b)$ puisque l'ensemble des niveaux pour lesquels on peut rechercher un successeur de b est $\{0, \ldots, \mathbf{niveau}(b)-1\}$. Comme on le verra par la suite, l'existence de brins de niveaux supérieurs à $\mathbf{niveau}(b)$ dans la séquence de connexion qui contient b est une information trop fine pour la détermination des successeurs de b. Il suffit pour cela de savoir qu'au moins un brin a un niveau supérieur ou égal à celui de b dans la séquence. Si nous considérons le brin -5 dans $SC_4(-11)$, nous avons (figure 4.43)

$$\mathcal{L}_4(-5) = 4; \text{ et } \left\{\begin{array}{lcl} \mathbf{succ}_{4,-5}(1) & = & 2 \\ \mathbf{succ}_{4,-5}(2) & = & 4 \\ \mathbf{succ}_{4,-5}(3) & = & 7 \\ \mathbf{succ}_{4,-5}(4) & = & 17. \end{array}\right.$$

Étant données les fonctions $\mathcal{L}_i$ et $\mathbf{succ}_{i,b}$, les résultats illustrés précédemment se formalisent de la façon suivante :

Théorème 8 *Soient un codage implicite d'une pyramide définie par n niveaux et un brin b de niveau l. Pour tout $i \in \{l, \ldots, n\} \cap \mathcal{DS}_b$, le φ ou σ successeur de b au niveau $j < \mathcal{L}_i(b)$ peut être retrouvé dans $SC_i(b^i)$ à l'aide de l'équation suivante :*

$$\left.\begin{array}{l} \forall i \in \{l, \ldots, n\} \cap \mathcal{DS}_b, \\ \forall j \in \{1, \ldots, \mathcal{L}_i(b)\} \end{array}\right\} \left\{\begin{array}{lcll} \varphi_{j-1}(b) & = & d_{\mathbf{succ}_{i,b}(j)} & \textit{si } b \textit{ est contracté} \\ \sigma_{j-1}(b) & = & d_{\mathbf{succ}_{i,b}(j)} & \textit{si } b \textit{ est supprimé.} \end{array}\right.$$

Preuve:
Voir [39] Théorèmes 4 et 5. □

L'utilisation de ce théorème pour le brin -5 au niveau 4 nous donne :

$$\left\{\begin{array}{lcllcll} \sigma_0(-5) & = & b_{\mathbf{succ}_{4,-5}(1)} & = & b_2 & = & 6 \\ \sigma_1(-5) & = & b_{\mathbf{succ}_{4,-5}(2)} & = & b_4 & = & -9 \\ \sigma_2(-5) & = & b_{\mathbf{succ}_{4,-5}(3)} & = & b_7 & = & -3 \\ \sigma_3(-5) & = & b_{\mathbf{succ}_{4,-5}(4)} & = & b_{17} & = & 5. \end{array}\right.$$

Le Théorème 8 nous a permis de retrouver l'ensemble des successeurs de -5 dans la pyramide. Il existe toutefois deux cas où ce théorème ne peut être appliqué ou ne nous fournit pas toutes les informations désirées.

Le premier cas se produit lorsque b est le dernier brin d'une séquence de niveau i. On a dans ce cas $i \notin \mathcal{DS}_b$ et le Théorème 8 ne nous fournit aucune information. Dans ce cas, l'information manquante est fournie par le σ_0 ou φ_0 successeur de b. On a en effet :

Proposition 1 *Soient une pyramide définie par n niveaux et un brin b de niveau l. Si b est le dernier brin d'une séquence de connexion de niveau $i > l$, on a :*

$$\forall j \in \{0, \ldots, l-1\} \left\{ \begin{array}{lcll} \varphi_j(b) & = & \varphi_0(b) & \textit{Si } b \textit{ est contracté} \\ \sigma_j(b) & = & \sigma_0(b) & \textit{Si } b \textit{ est supprimé.} \end{array} \right.$$

Preuve:
Voir [39] Proposition 8 □

Notons que dans ce cas, $\varphi_0(b)$ ou $\sigma_0(b)$ est égal à $\varphi_i(b^i)$ ou $\sigma_i(b^i)$ selon l'opération appliquée sur b et le type de K_i (Théorème 4).

Le brin 3 contracté au niveau 3 est le dernier brin de $SC_4(-11)$ (figures 4.41 ou 4.43). On a donc par la Proposition 1 :

$$\varphi_1(3) = \varphi_2(3) = \varphi_0(3) = 11.$$

Un autre cas particulier intervient lorsque $\mathcal{L}_i(b) < \textbf{niveau}(b)$. Dans ce cas, le Théorème 8 ne peut nous fournir les φ ou σ successeurs du brin b pour les niveaux $\{\mathcal{L}_i(b), \ldots, \textbf{niveau}(b) - 1\}$. Un tel cas se produit lorsque le brin b contracté au niveau l est le dernier brin de $CW_l(b^l)$. En effet, dans ce cas $b.SC_{l-1}(\alpha(b))$ est un suffixe de $SC_l(b^l)$. Le φ ou σ successeur de b au niveau $l-1$ ne sera donc pas présent dans $SC_l(b^l)$. Plus généralement, tous les successeurs de b du niveau $L_{l-1}(\alpha(b))$ à $l-1$ ne seront pas présents dans $SC_l(b^l)$. Cette situation peut perdurer sur plusieurs niveaux tant que les brins b^i seront les seuls ou les derniers brins de leurs séquences de connexion. Dans ce cas, la solution est encore une fois fournie par le σ_0 ou φ_0 successeur du dernier brin de la séquence.

Proposition 2 *Soient une pyramide définie par n niveaux et un brin b de niveau l. Pour tout niveau $i > l$ tel que $\mathcal{L}_i(b) < l$, on a :*

$$\forall j \in \{\mathcal{L}_i(b), \ldots, l-1\} \left\{ \begin{array}{lcll} \varphi_j(b) & = & b' & \textit{si } b \textit{ est contracté} \\ \sigma_j(b) & = & b' & \textit{si } b \textit{ est supprimé} \end{array} \right.$$

avec :

$$b' = \left\{ \begin{array}{ll} \varphi_i(b^i) & \textit{si } K_i \textit{ est un noyau de contraction} \\ \sigma_i(b^i) & \textit{si } K_i \textit{ est un noyau de suppression.} \end{array} \right.$$

Preuve:
Voir [39] Corollaire 5 et Proposition 13 □

Notons la similitude entre la Proposition 2 et le second cas du théorème 6. De fait, si $\mathcal{L}_i(b) < l$ on a $\mathcal{L}_i(b) = L_{l-1}(\alpha(b))$ ([39] Proposition 13) et la Proposition 2 peut se concevoir comme une application de ce théorème à la séquence de connexion $SC_{l-1}(\alpha(b))$. Notons également que cette

proposition s'utilise en conjonction avec le Théorème 8. En effet, le Théorème 8 nous fournit les φ ou σ successeurs d'un brin b des niveaux 0 à $\mathcal{L}_i(b) - 1$ tandis que la Proposition 2 nous fournit les successeurs pour les niveaux restants : $\{\mathcal{L}_i(b), \ldots . l-1\}$.

Si nous considérons le brin 5 au niveau 4, nous avons $\mathcal{L}_4(5) = 3 < 4$. Le σ successeur de 5 au niveau 3 n'est donc pas présent dans $SC_4(11)$. Nous le retrouvons à l'aide de la Proposition 2 :

$$\sigma_3(5) = \sigma_4(-11) = \varphi_0(3) = 11.$$

Notez que la dernière égalité ($\sigma_4(-11) = \varphi_0(11)$) provient du Théorème 4.

4.6.7.c Un algorithme séquentiel calculant l'ensemble de la pyramide

L'Algorithme 1 prend en paramètre un niveau i et un brin appartenant à $\mathcal{BS}_i$ et initialise l'ensemble des σ ou φ successeurs des brins contenus dans $SC_i(b)$. L'ensemble des brins parcourus par cet algorithme est égal à $SC_i(b^i)\varphi_i(b^i)$ si K_i est un noyau de contraction et $SC_i(b^i)\sigma_i(b^i)$ si K_i est un noyau de suppression. Le calcul du brin suivant est effectué entre les lignes 11 et 15 et se base sur le Théorème 5. Cet algorithme parcourt donc les mêmes brins que l'algorithme `survivant` (figure 4.39(b)).

Les résultats obtenus dans le Théorème 8 et les propositions 1 et 2 peuvent se résumer de la façon suivante :

Soient un niveau i et un brin b de niveau l, le σ ou φ successeur de b à un niveau $j \in \{0, \ldots, l-1\}$ est le premier brin de niveau supérieur ou égal à j qui suit b dans la séquence $SC_i(b^i)\varphi_i(b^i)$ si K_i est un noyau de contraction et dans $SC_i(b^i)\sigma_i(b^i)$ si K_i est un noyau de suppression.

D'un point de vue algorithmique, le principal inconvénient de cette formulation est que le φ ou σ successeur de niveau j d'un brin b se trouve en avant de b dans la séquence de connexion. Si l'on parcourt par exemple $SC_4(-11)$ (figure 4.43), on rencontrera donc -5 avant de connaître ses σ successeurs. Afin de pallier cet inconvénient, nous stockons un tableau *prec* de taille $i+1$ initialisé de telle façon que $prec[j]$ contienne à chaque étape le dernier brin rencontré de niveau supérieur ou égal à j. Ce tableau est donc initialement rempli par le premier brin de la séquence de connexion (Algorithme 1 lignes 6 et 7) puis rempli jusqu'au niveau $\mathbf{niveau}(b')$ pour chaque brin b' traversé.

Si b' représente le brin courant durant notre parcours et si $b' \in SC_i(b)$, tous les brins entre $prec[j]$ et b' auront donc un niveau inférieur à j. Le brin b' est donc le premier brin de niveau supérieur à j que l'on rencontre en parcourant $SC_i(b)$ depuis $prec[j]$ et l'on a (Définition 25) :

$$\left.\begin{array}{lll} \forall & b' \in & SC_i(b) \\ \forall & j \in & \{1, \ldots, \mathbf{niveau}(b')\} \end{array}\right\} b_{\mathbf{succ}_{i,prec[j]}(j)} = b'. \tag{4.27}$$

On peut donc appliquer le Théorème 8. De même, les Propositions 1 et 2 peuvent s'appliquer si b' est égal à $\varphi_i(b^i)$ ou $\sigma_i(b^i)$. Les affectations induites par le Théorème 8 et les Propositions 1 et 2 sont effectuées lignes 17 à 25.

Nous avons montré [39] que cet algorithme calcule bien l'ensemble des successeurs de tout brin $b \in \mathcal{B}$ de 0 à $\mathbf{niveau}(b)$. Plutôt que de fournir une preuve exhaustive de la validité de cet algorithme, nous allons illustrer son fonctionnement sur la séquence de connexion $SC_4(-11)$ (figure 4.43). Le tableau *prec* est initialement rempli par le premier brin de la séquence c'est à dire 11 (ligne 6 et 7). Le premier brin considéré dans la boucle est le second brin de la séquence -5. Le brin -5 étant de niveau 4 et K_4 étant un noyau de suppression, nous pouvons appliquer le Théorème 6 :

$$\sigma_3(-11) = \sigma_2(-11) = \sigma_1(-11) = \sigma_0(-11) = -5.$$

```
construire_pyramide_seq(entier i,brin b)
{
    brin b' ;
    brin prec[i+1]

    Pour j=1 à min(niveau(b), i + 1)
        prec[j]=b ;

    Faire
    {
        Si ((état(niveau(b')) == Contracté) ou
        (niveau(b') > i et état(i) == Contracté))
            b' = φ(b') ;
        sinon
            b' = σ(b') ;

        Pour j=1 à min(niveau(b'), i + 1)
        {
            Si (état(niveau(prec[j])) == Contracté ou
                (niveau(prec[j]) > i et état(i) ==Contracté)
            )
                Σ (j − 1, α(prec[j])) = b' ; //φ_{j−1}(prec[j]) = b'
            sinon
                Σ(j − 1, prec[j]) = b' ; //σ_{j−1}(prec[j]) = b'
        }
        Pour j=1 à min(niveau(b'), i + 1)
            prec[j]=b' ;
    }
    tant que (niveau(b') ≤ i)
}
```

Algorithme 1: *construction séquentielle de la pyramide*

		Valeurs successives du tableau *prec*					Affectations
brins	niveau	prec[5]	prec[4]	prec[3]	prec[2]	prec[1]	
-5	4	-11	-11	-11	-11	-11	$\sigma_3(-11) = -5 \quad \sigma_2(-11) = -5$ $\sigma_1(-11) = -5 \quad \sigma_0(-11) = -5$
6	1	-11	-5	-5	-5	-5	$\sigma_0(-5) = 6$
-12	1	-11	-5	-5	-5	6	$\varphi_0(6) = -12$
-9	2	-11	-5	-5	-5	-12	$\sigma_1(-5) = -9 \quad \varphi_0(-12) = -9$
-4	1	-11	-5	-5	-9	-9	$\sigma_0(-9) = -4$
-8	2	-11	-5	-5	-9	-4	$\sigma_1(-9) = -8 \quad \varphi_0(-4) = -8$
-3	3	-11	-5	-5	-8	-8	$\sigma_2(-5) = -3 \quad \sigma_1(-8) = -3$ $\sigma_0(-8) = -3$
-7	1	-11	-5	-3	-3	-3	$\varphi_0(-3) = -7$
1	1	-11	-5	-3	-3	-7	$\varphi_0(-7) = 1$
8	2	-11	-5	-3	-3	1	$\varphi_1(-3) = 8 \quad \varphi_0(1) = 8$
2	1	-11	-5	-3	8	8	$\sigma_0(8) = 2$
9	2	-11	-5	-3	8	2	$\sigma_1(8) = 9 \quad \varphi_0(2) = 9$
-2	1	-11	-5	-3	9	9	$\sigma_0(9) = -2$
-1	1	-11	-5	-3	9	-2	$\varphi_0(-2) = -1$
7	1	-11	-5	-3	9	-1	$\varphi_0(-1) = 7$
10	1	-11	-5	-3	9	7	$\varphi_0(7) = 10$
5	4	-11	-5	-3	9	10	$\sigma_3(5) = 5 \quad \varphi_2(-3) = 5$ $\sigma_1(9) = 5 \quad \varphi_0(10) = 5$
-10	1	-11	5	5	5	5	$\sigma_0(5) = -10$
3	3	-11	5	5	5	-10	$\sigma_2(5) = 3 \quad \sigma_1(5) = 3$ $\varphi_0(-10) = 3$
11	5	-11	5	3	3	3	$\sigma_4(-11) = 11 \quad \sigma_3(5) = 11$ $\varphi_2(3) = 11 \quad \varphi_1(3) = 11$ $\varphi_0(3) = 11$

TAB. 4.3 – Les differentes valeurs prises par le tableau *prec* et les affectations effectuées durant le parcours de $SC_4(-11)$ par l'Algorithme 1.

Ces affectations sont effectuées lignes 17 à 25 (voir également la Table 4.3).

Supposons à présent que le brin courant soit le brin -3. Le brin -3 est le premier brin de niveau supérieur à 2 que l'on rencontre en parcourant $SC_4(-11)$ depuis -8 et le premier brin de niveau supérieur à 3 que l'on rencontre en parcourant $SC_4(-11)$ depuis -5 (figure 4.43). On a donc :

$$b_{\mathbf{succ}_{1,-8}} = b_{\mathbf{succ}_{2,-8}} = -3 \text{ et } b_{\mathbf{succ}_{3,-5}} = -3.$$

Les brins -8 et -5 étant tout deux supprimés dans la pyramide, on a d'après le Théorème 8 :

$$\begin{array}{lllllll} \sigma_2(-5) & = & -3 & \sigma_1(-8) & = & -3 \\ \sigma_0(-8) & = & -3. & & & \end{array}$$

L'algorithme 1 comporte deux boucles imbriquées dans la boucle `tant que` (lignes 9 à 29). L'ensemble des séquences de connexion de niveau i formant une partition de l'ensemble des brins initiaux (équation 4.20), la boucle `tant que` considérera chaque brin initial une fois et une seule. Si nous prenons $i = n$, $\min(\mathbf{niveau}(b'), i+1)$ est égal à $\mathbf{niveau}(b')$ (lignes 17 et 26). La complexité de l'algorithme 1 est alors égale à :

$$2 \sum_{b \in \mathcal{B}} \mathbf{niveau}(b) = 2 \sum_{i=1}^{n+1} |\mathcal{BS}_i|.$$

Intuitivement, cette dernière équation signifie que nous effectuons pour chaque brin initial 2 opérations pour chacun des niveaux où il survit (affectation du tableau *prec* et du σ ou φ successeur). Notez que cette complexité est très proche de la complexité optimale puisque l'initialisation de la pyramide suppose au moins de parcourir l'ensemble des brins initiaux et d'initialiser pour chacun l'ensemble des ses σ successeurs pour les niveaux où il survit.

4.6.7.d Un algorithme parallèle calculant l'ensemble de la pyramide

Les fonctions présentées dans l'Algorithme 2 calculent en parallèle le σ successeurs de chaque brin pour l'ensemble des niveaux où il intervient. Nous avons montré [39] la validité de cet algorithme et montré que sa complexité parallèle est égale à $\mathcal{O}(\log_2(|SC_i(d^{max})|))$ où $SC_i(d^{max})$ est la plus longue séquence de connexion définie au niveau i. Si nous supposons que toutes les séquences ont approximativement la même longueur on a :

$$|SC_i(d^{max})| \approx \frac{|\mathcal{B}|}{|\mathcal{BS}_i|} \Rightarrow \mathcal{O}(\log_2(|SC_i(d^{max})|)) \approx \mathcal{O}(\log_2(|\mathcal{B}|) - \log_2(|\mathcal{BS}_i|)).$$

La complexité de l'algorithme est donc dans ce cas approximativement en $\mathcal{O}(\log_2(|\mathcal{B}|))$.

Plutôt que de donner une explication exhaustive de ces fonctions, nous nous contenterons de donner l'idée principale et le rôle des principales variables. Des explications plus détaillées peuvent être trouvées dans [39] section 6.

Afin de simplifier les notations, notons par $S_i(b)$ la séquence égale à $SC_i(b).\sigma_i(b)$ si K_i est un noyau de suppression et $SC_i(b)\varphi_i(b)$ si K_i est un noyau de contraction. L'idée de l'algorithme 2 est de demander à chaque brin initial de calculer pour tout $j \in \{0, \ldots, i\}$ le premier brin de niveau supérieur à j rencontré dans une séquence de connexion. Ces résultats sont stockés dans les tableaux $First(., b)$ définis de la façon suivante :

```
pyramide_par(carte G,fonction niveau, entier i )
{
    Pour tout b ∈ B faire en parallèle
            Next[b] = σ(b)
    Pour tout b ∈ B faire en parallèle
            suivants_par(b,i) ;
    Pour tout b ∈ B faire en parallèle
        Pour j= 0 à niveau(b) -1
            Σ(j,b) = First(j,b)
}
suivants_par(brin b, entier i)
{
    brin d'=Next[d]
    entier max_level=MIN(level(d'),i+1) ;

    // Initialisation du tableau First
    Pour j=1 à max_level
        First(j-1,b) = b'

    min_level[b]=max_level+1 ;


    tant que (niveau(b')≤ i)
    {
        Si (état(niveau(d')) == Contracté)
            b'' = α(b') ;
        sinon
            b'' =b' ;
        Si (min_level[b''] > min_level[b])
        {
            // b' a trouvé des brins intéressants
            // => on les recopie dans First(.,b)

            Pour j=min_level[d] à min_level[b'']-1
                First(j-1,b) = First(j-1,b'')
            min_level[b]=min_level[b''] ;
        }
        b'=Next[b]=Next[b''] ;
    }
}
```

Algorithme 2: *Initialisation en parallèle de l'ensemble du σ successeur de chaque brin b pour l'ensemble des niveaux où il intervient*

– si b est supprimé, $First(j, b)$ est défini comme le premier brin qui suit b dans $S_i(b^i)$ et dont le niveau est supérieur ou égal à $j + 1$;
– si b est contracté, $First(j, b)$ est défini comme le premier brin qui suit $\alpha(b)$ dans $S_i(\alpha(b)^i)$ et dont le niveau est supérieur ou égal à $j + 1$.

Si nous oublions momentanément le dernier brin des séquences $S_i(b^i)$, le processus associé à b parcourra $SC_i(b^i)$ si b est supprimé et $SC_i(\alpha(b)^i)$ si b est contracté. Cette dissymétrie dans les traitements permet d'assurer que $First(j, b)$ est égal à $\sigma_j(b)$ en fin de traitement (Algorithme 2 lignes 7 à 9). Il aurait également été possible de demander au processus associé à b de parcourir $S_i(b^i)$ quelle que soit l'opération ayant réduit b. Dans ce cas, $First(j, b)$ contiendrait $\varphi_j(b)$ si b est contracté.

Il est relativement clair que la complexité de cet algorithme est proportionnelle à la longueur de la plus longue séquence de connexion de niveau i si l'on n'autorise pas de communications entre les processus. En effet, dans ce cas le temps total sera déterminé par les temps d'exécution des processus dont les brins se trouvent en début de séquences. Afin d'améliorer la rapidité de l'algorithme, nous échangeons les résultats partiels des processus.

Illustrons un exemple de communications entre processus à l'aide de la figure 4.43. Les brins -5 et 4 sont respectivement supprimés et contractés dans la pyramide. Le noyau K_4 étant un noyau de suppression, les processus associés à -5 et 4 parcourent respectivement les séquences $S_4(-5^4) = SC_4(-5^4 = 11)\sigma_4(-11)$ et $S_4(\alpha(4)^4) = SC_4((-4)^4 = 11)\sigma_4(-11)$. Les deux processus parcourent donc $SC4(-11)\sigma_4(-11)$. Supposons qu'à une itération de l'algorithme le brin courant du processus associé à -5 soit -4 alors que celui de 4 soit -7. Le brin courant de chaque processus est stocké dans le tableau partagé $Next[]$. L'état des variables à ce moment là est :

$$\begin{array}{lllllllllll} First(0,-5) & = & 6 & First(1,-5) & = & -9 & & & & Next[-5] & = & -4 \\ First(0,4) & = & -8 & First(1,4) & = & -3 & First(2,4) & = & -3 & Next[4] & = & -7. \end{array}$$

En lisant le tableau $First(.,4)$, le processus associé à -5 peut donc savoir sans parcourir la séquence de brins de -4 à -7 que le prochain brin de niveau 3 rencontré après -4 est -3 (puisque $First(2,4) = -3$)(Algorithme 2 lignes 34, 35). De plus, ayant complété son tableau $First(.,-5)$ à partir de $First(.,4)$ ce processus peut sauter la séquence de brins de -4 à -7 et continuer le parcours à partir du brin -7(Algorithme 2 ligne 38). On a donc en fin de boucle, pour le processus -5 :

$$First(0,-5) = 6 \quad First(1,-5) = -9 \quad First(2,-5) = -3 \quad Next[-5] = -7.$$

4.6.8 Fenêtres de réduction et Champs Récepteurs

Le plongement d'une carte combinatoire permet de décrire l'ensemble des pixels et des frontières de pixels associés à un sommet d'une carte des régions ou une face d'une carte des frontières. Nous allons dans cette section étudier plus particulièrement le plongement d'une carte des régions. Chaque sommet de la carte combinatoire codera donc une région et le σ-cycle d'un brin de la carte de base pourra être interprété comme l'ensemble des lignels qui délimitent un pixel (section 4.6.1).

Les cartes combinatoires étant définies à partir des brins l'étude des partitions décrites par les cartes suppose tout d'abord une définition des régions en termes de brins :

Définition 26 Région d'une carte combinatoire

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et son graphe orienté $\mathcal{OG} = (V, E)$ associé. Un ensemble de brins $R \subset \mathcal{B}$ sera appelé une région si et seulement si :

1. *L'ensemble R est connecté dans $\mathcal{OG}$: Quels que soient b, b' appartenant à R, il existe un chemin de $\mathcal{OG}$ inclus dans R connectant b à b' ou b' à b.*
2. *L'ensemble R contient tous ces pixels :*

$$\sigma^*(R) = R$$

Intuitivement, deux brins b et b' adjacents dans $\mathcal{OG}$ appartiennent soit au même sommet ($b' = \sigma(b)$) soit à deux sommets adjacents par l'arête $\alpha^*(b)$ ($b' = \varphi(b)$). La notion d'ensemble connexe de brins recouvre donc bien la notion d'ensemble connexe de sommets et celle d'ensemble connexe de pixels si chaque sommet est associé à un pixel. Le second point de la définition des régions impose que chaque sommet dont la σ-orbite intersecte la région soit inclus dans celle-ci. En d'autres termes :

$$\forall b \in \mathcal{B}\ \sigma^*(b) \cap R = \emptyset \text{ ou } \sigma^*(b) \subset R.$$

Une région ne peut donc pas contenir "une moitié" de pixel.

La σ-orbite d'un brin est donc soit incluse dans la région soit disjointe de celle-ci. En revanche, aucune contrainte n'est imposée sur l'α-orbite d'un brin. L'ensemble des brins d'une région peuvent être décomposés en deux classes en fonction de l'intersection de leurs α-orbites avec la région :

Définition 27 Brins Internes et Frontières

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$ et une région R de G. Un brin $b \in R$ sera qualifié de :

- *brin interne de R si $\alpha(b)$ appartient également à R. On a dans ce cas :*

$$\sigma^*(b) \subset R \text{ et } \sigma^*(\alpha(b)) \subset R;$$

- *brin frontière, si $\alpha(b) \notin R$. Dans ce cas :*

$$\sigma^*(b) \subset R \text{ alors que } \sigma^*(\alpha(b)) \cap R = \emptyset.$$

Dans le cadre des pyramides, la notion de plongement est codée grâce aux notions de fenêtre de réduction et de champs récepteurs (section 4.2).

4.6.8.a Fenêtres de réduction

Dans le cadre des pyramides usuelles, une fenêtre de réduction relie un sommet survivant de niveau i à un ensemble de sommets non survivants de niveau $i-1$. Dans le cadre des cartes combinatoires, un sommet est défini par son σ-cycle. Nous devons donc tout d'abord définir la notion de fenêtre de réduction d'un brin puis agréger les fenêtres de réduction des brins définissant un σ cycle afin d'obtenir la fenêtre de réduction d'un sommet. Deux brins successifs b et b' d'un sommet survivant sont reliés par $\sigma'(b) = b'$ où σ' représente la permutation σ de la carte réduite. La fenêtre de réduction d'un brin b est donc définie comme l'ensemble des brins non survivants qu'il nous faut traverser pour connecter b à $\sigma'(b)$:

Définition 28 Fenêtre de réduction d'un brin

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$, un noyau K et la carte réduite $G' = (\mathcal{BS}, \sigma', \alpha)$. La fenêtre de réduction d'un brin $b \in \mathcal{BS}$, notée $RW(b)$ est définie par :

$$RW(b) = \begin{cases} CW(b) & \text{Si } K \text{ est un noyau de contraction} \\ bCW^*(\alpha(b)) & \text{Si } K \text{ est un noyau de suppression} \end{cases}$$

où $CW^*(b)$ représente le chemin de connexion $CW(b)$ sans son premier brin b.

La fenêtre de réduction d'un brin b, $RW(b) = b.b_1 \ldots .b_p$ est donc composée du brin b suivi d'une séquence de brins non survivants. On a de plus par l'équation 4.12 :

$$\sigma'(b) = \begin{cases} \varphi(b_p) & \text{si } K \text{ est un noyau de contraction} \\ \sigma(b_p) & \text{si } K \text{ est un noyau de suppression.} \end{cases} \tag{4.28}$$

Une fenêtre de réduction de brins connecte donc bien un brin à son σ successeur dans la carte réduite. De plus, nous avons montré [38] qu'une fenêtre de réduction définit un chemin dans $\mathcal{OG}$ et est donc connectée.

Notez que cette définition d'une fenêtre de réduction est dictée par le choix d'utiliser une carte des régions. Si nous utilisions un carte des frontières, les régions seraient décrites par des φ-cycles et la fenêtre de réduction d'un brin devrait connecter celui-ci à son φ successeur dans la carte réduite.

Un sommet de la carte réduite G' est défini par son σ-cycle $\sigma'^*(b_1) = (b_1, \ldots, b_p)$, chaque brin b_j étant lié à b_{j+1} par $b_{j+1} = \sigma'(b_j)$. Le plongement d'un sommet de la carte réduite dans la carte initiale est donc défini comme la concaténation des fenêtres de réduction connectant chaque brin b_j à b_{j+1}.

Définition 29 Fenêtre de réduction d'un sommet

Soient une carte combinatoire $G = (\mathcal{B}, \sigma, \alpha)$, un noyau K et la carte réduite $G' = (\mathcal{BS}, \sigma', \alpha)$. La fenêtre de réduction d'un sommet $\sigma'^*(b_1) = (b_1, \ldots, b_p)$, notée $FR_{\sigma'^*(b_1)}$ est définie par :

$$FR_{\sigma'^*(b_1)} = \bigodot_{j=1}^{p} RW(b_j)$$

où $\bigodot$ désigne l'opérateur de concaténation.

Chaque fenêtre de réduction d'un brin est connectée. De plus par l'équation 4.28 chaque premier brin d'une fenêtre de réduction est le σ ou φ successeur du dernier brin de la séquence de connexion précédente. La fenêtre de réduction d'un sommet est donc connectée. Plus précisément, le dernier brin de $RW(b_p)$ étant connecté à b_1, on peut montrer que $FR_{\sigma'^*(b_1)}$ forme un cycle dans le graphe $\mathcal{OG}$.

Si K est un noyau de suppression, nous avons montré [38] que la fenêtre de réduction du sommet reconstitue l'orbite originale du sommet. En d'autres termes :

$$\forall b \in \mathcal{BS} \quad FR_{\sigma'^*(b)} = \sigma^*(b). \tag{4.29}$$

La région étant réduite à un seul sommet contient trivialement tous ces sommets.

Si K est un noyau de contraction, la preuve qu'une région contient tous ses sommets est un peu plus délicate et est basée sur la proposition suivante [36, 38] :

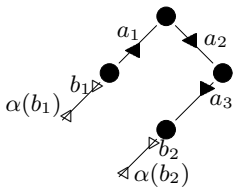
Proposition 3 *Soient une carte initiale $G = (\mathcal{B}, \sigma, \alpha)$, un noyau de contraction K, et la carte contractée $G' = (\mathcal{BS} = \mathcal{B} - K, \sigma', \alpha)$. Les composantes connexes de K vérifient :*

$$\forall \mathcal{T} \in \mathcal{CC}(K), \quad \forall b_1 \in \sigma^*(\mathcal{T}) \cap \mathcal{BS} \quad \sigma'^*(b_1) = \sigma^*(\mathcal{T}) \cap \mathcal{BS} \tag{4.30}$$

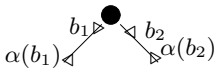
$$\mathcal{T} = \cup_{j=1}^{p} CW^*(\alpha(b_j)) \tag{4.31}$$

où $\mathcal{CC}(K)$ représente l'ensemble des composantes connexes de K et $\sigma'^*(b_1) = (b_1, \ldots, b_p)$.

Le noyau de contraction définissant une forêt de K, chacune de ses composantes connexes est un arbre qui contracte en un seul sommet un ensemble de sommets non survivants. L'équation 4.30 nous indique que l'ensemble des brins survivants appartenant aux sommets d'un arbre définissent la σ-orbite d'un sommet survivant. Le noyau K étant un noyau de contraction, chaque séquence $CW^*(\alpha(b_j))$ peut s'interpréter comme la séquence de brins non survivants connectant b_j à $\sigma'(b_j) = b_{j+1}$ (figure 4.44(a)). L'équation 4.31 nous montre que l'union de telles séquences définit l'ensemble des brins non survivants de l'arbre. Notons d'ailleurs que si nous substituons l'opérateur d'union à l'opérateur de concaténation, nous définissons un sens de parcours dans l'arbre qui nous fait passer 2 fois par chaque arête (voir par exemple la séquence $CW^*(\alpha(b_1)).CW^*(\alpha(b_2))$ sur la figure 4.44).



(a) $\mathcal{T}$



(b) $\sigma'^*(b_1)$

$$
\begin{array}{lcl}
\mathcal{T} & = & \alpha^*(a_1, a_2, a_3) \\
\mathcal{BS} & = & \alpha^*(b_1, b_2) \\
\sigma^*(\mathcal{T}) & = & \alpha^*(a_1, a_2, a_3) \cup \{b_1, b_2\} \\
\sigma'^*(b_1) & = & (b_1, b_2) = \sigma^*(\mathcal{T}) \cap \mathcal{BS} \\
 & & \\
CW^*(\alpha(b_1)) & = & a_1.a_2.a_3 \\
CW^*(\alpha(b_2)) & = & \alpha(a_3).\alpha(a_2).\alpha(a_1) \\
\mathcal{T} & = & CW^*(\alpha(b_1)) \cup CW^*(\alpha(b_2))
\end{array}
$$

(c) Relations entre les entités

FIG. 4.44 – Un arbre d'un noyau de contraction (a) et le sommet réduit associé (b). Les brins contractés sont représentés par des triangles pleins. La Figure (c) représente les différentes relations entre les entités décrites dans les Figures (a) et (b).

Considérons un sommet survivant $\sigma'^*(b_1) = (b_1, \ldots, b_p)$ et un arbre $\mathcal{T} = \cup_{j=1}^{p} CW^*(\alpha(d_j))$. Puisque $\mathcal{T}$ est une composante connexe de K, un brin $b \in \sigma^*(\mathcal{T}) \cap K$ doit appartenir à $\mathcal{T}$ autrement $\mathcal{T}$ serait connecté à un autre arbre de K. On a donc $\sigma^*(\mathcal{T}) \cap K = \mathcal{T}$. De plus puisque $\mathcal{BS} = \mathcal{B} - K$, on a $\mathcal{B} = \mathcal{BS} \cup K$ et :

$$
\begin{array}{rcll}
\sigma^*(\mathcal{T}) & = & \sigma^*(\mathcal{T}) \cap \mathcal{B} = \sigma^*(\mathcal{T}) \cap (\mathcal{BS} \cup K) & \\
 & = & (\sigma^*(\mathcal{T}) \cap \mathcal{BS}) \cup (\sigma^*(\mathcal{T}) \cap K) & \\
 & = & \sigma'^*(b_1) \cup \mathcal{T} & \text{(équation 4.30)} \\
 & = & \sigma'^*(b_1) \cup \bigcup_{j=1}^{p} CW^*(\alpha(b_j)) & \text{(équation 4.31)} \\
 & = & \bigcup_{j=1}^{p} b_j.CW^*(\alpha(b_j)) & \\
 & = & \bigcup_{j=1}^{p} RW(b_j) &
\end{array}
$$

où $\sigma'^*(b_1) = (b_1, \ldots, b_p)$.

La région $FR_{\sigma'^*(b_1)} = \bigodot_{j=1}^{p} RW(b_j)$ définit donc un ordre sur $\sigma^*(\mathcal{T})$ et l'on a :

$$\sigma^*(FR_{\sigma'^*(b_1)}) = \sigma^*(\sigma^*(\mathcal{T})) = \sigma^*(\mathcal{T}) = FR_{\sigma'^*(b_1)}.$$

La fenêtre de réduction d'un sommet contient bien l'ensemble de ses sommets et définit une région.

L'ensemble des chemins de connexion définissant une partition de l'ensemble des brins initiaux, l'ensemble des fenêtres de réduction vérifie la même propriété. L'ensemble des σ orbites de la carte

réduite définit également une partition de l'ensemble des brins survivants. Un brin de la carte initiale appartient donc à une et une seule fenêtre de réduction de brin qui est elle même impliquée dans la création d'une et une seule fenêtre de réduction de sommet. L'ensemble des fenêtres de réduction de sommets forme donc une partition de l'ensemble des brins initiaux.

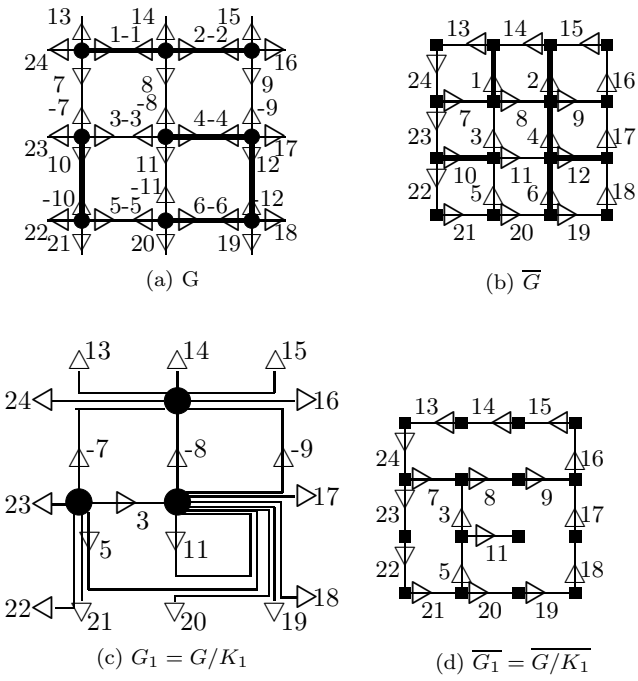


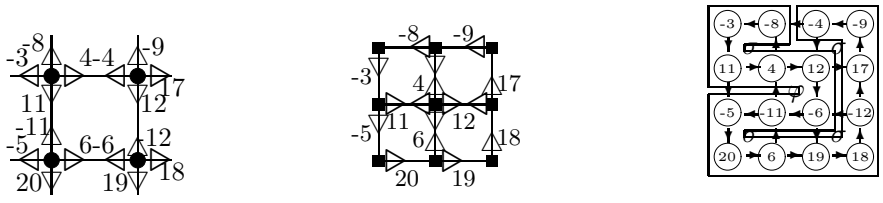
FIG. 4.45 – Une carte des régions codant une grille 3×3 (a) et son dual (b) contractés par un noyeau $K_1 = \alpha^*(1, 2, 4, 12, 6, 10)$. Les arêtes contractées sont représentées en traits épais sur les figures (a) et (b).

La figure 4.45 reprend l'exemple de la figure 4.25 où une carte des régions codant une grille 3×3 est contractée par un noyau de contraction K_1 (figure 4.45). Les graphes orientés $\mathcal{OG}$ et $\mathcal{OG}_1$ sont représentés sur le figures 4.25(c) et 4.28(c). La figure 4.46 représente la fenêtre de réduction du sommet $\sigma'^*(-8)$ (sommet cental sur la figure 4.45(c)) dans la carte des régions, la carte duale et le graphe orienté $\mathcal{OG}$. Le sommet survivant est représenté par le σ' cycle $(-8, -3, 11, -11, -5, 20, 19, 18, 17, -9)$ (figure 4.45(c)). Si nous calculons la fenêtre de réduction de chaque brin survivant (figure 4.34), nous obtenons :

$$FR_{\sigma'^*(-8)} = \boxed{\text{-8}}.\boxed{\text{-3}}.\boxed{\text{11.4.12.-6}}.\boxed{\text{-11}}.\boxed{\text{-5}}.\boxed{\text{20.6}}.\boxed{\text{19}}.\boxed{\text{18.-12}}.\boxed{\text{17}}.\boxed{\text{-9.-4}} \tag{4.32}$$

où chaque boite entoure une fenêtre de réduction.

La région $FR_{\sigma'^*(-8)}$ est composée de 4 pixels avec 4 lignels correspondant à des brins internes et 8 lignels correspondant à des brins frontière (Définition 27). On peut noter que la dénomination de ces brins correspond à la notion intuitive que l'on peut avoir d'un brin interne et d'un brin



(a) $FR_{\sigma_1^*(-8)}$ dans G (b) $FR_{\sigma_1^*(-8)}$ dans $\overline{G}$ (c) $FR_{\sigma_1^*(-8)}$ dans $\mathcal{OG}$

FIG. 4.46 – La fenêtre de réduction $FR_{\sigma_1^*(-8)}$ définie par K_1 (Figure 4.45)

frontière. Les brins internes connectent deux pixels de la région alors que les brins frontières codent les relations d'adjacence de la région et délimitent sa frontière. De plus, l'on peut noter que la séquence de brins frontières rencontrée dans $FR_{\sigma'^*(-8)}$: -8,-3,-5,20,19,18,17,-9 correspond à une orientation de la frontière dans le sens négatif (figure 4.46). Cette dernière propriété bien que vérifiée sur de nombreux exemples n'est pas encore complètement démontrée.

4.6.8.b Champs récepteurs

Dans le cadre des pyramides usuelles, l'ensemble des enfants d'un sommet est appelé sa fenêtre de réduction (pages 128 et 131). Chacun des sommets d'une fenêtre de réduction possède lui même des enfants et l'itération de cette relation parent - enfant permet d'associer à un sommet de la pyramide un ensemble de sommets définis dans le graphe ou l'image à la base de celle-ci. Cet ensemble d'enfants à la base de la pyramide est appelé le champ récepteur du sommet (page 128).

Dans le cadre des pyramides combinatoires, cette notion d'itération de relations parents-enfants est prise en compte dans la notion de séquences de connexion qui sont construites récursivement à partir des séquences de connexion de niveau inférieur. Nous définissons donc le champ récepteur d'un brin b suivant le même schéma que les fenêtres de réduction mais en utilisant des séquences de connexion plutôt que des chemins :

Définition 30 Champs récepteur d'un brin
Soient une pyramide définie par n noyaux, un niveau $i \in \{1, \ldots, n\}$ et un brin $b \in \mathcal{BS}_i$. Le champ récepteur de b de niveau i, noté $RF_i(b)$ est défini par

$$RF_i(b) = \begin{cases} \mathcal{SC}_i(b) & \text{Si } K_i \text{ est un noyau de suppression} \\ b.\mathcal{SC}_i^*(\alpha(b)) & \text{Si } K_i \text{ est un noyau de contraction.} \end{cases}$$

Soit un brin $b \in \mathcal{BS}_i$ avec $RF_i(b) = b.\ldots.b_p$. Si K_i est un noyau de suppression, nous avons $RF_i(b) = \mathcal{SC}_i(b)$ et $\sigma_i(b)$ et égal à $\sigma_0(b_p)$ ou $\varphi_0(b_p)$ selon l'opération qui a réduit b_p (Théorème 4). De même, si K_i est un noyau de contraction, on a $RF_i(b) = b.\mathcal{SC}_i(\alpha(b))$ et $\varphi_i(\alpha(b)) = \sigma_i(b)$ est égal à $\sigma_0(b_p)$ ou $\varphi_0(b_p)$. On a donc dans les deux cas :

$$\sigma_i(b) = \begin{cases} \varphi_0(b_p) & \text{Si } b_p \text{ a été contracté} \\ \sigma_0(b_p) & \text{Si } b_p \text{ a été supprimé} \end{cases} \tag{4.33}$$

où b_p est le dernier brin de $RF_i(b)$.

Notez que le champ récepteur d'un brin est trivialement connecté en vertu du Théorème 5. Plus précisément, le champ récepteur d'un brin définit un chemin dans le graphe $\mathcal{OG}$.

Étant donnée la notion de champ récepteur d'un brin, nous agrégeons comme précédemment les champs récepteurs des brins de façon à former le champ récepteur d'un sommet.

Définition 31 Champ récepteur d'un sommet

Soient une pyramide définie par n noyaux, un niveau $i \in \{1, \ldots, n\}$ et un brin $b \in \mathcal{BS}_i$. Le champ récepteur de $\sigma_i^*(b)$ est défini par :

$$R_{\sigma_i^*(b)} = \bigodot_{j=1}^{p} RF_i(b_j)$$

avec $\sigma_i^*(b) = (b_1, \ldots, b_p)$.

Le φ_0 ou σ_0 successeur du dernier brin d'un champ récepteur $RF_i(b_j)$ étant égal à $b_{j+1} = \sigma_i(b_j)$ (équation 4.33), le champ récepteur d'un sommet est trivialement connecté dans la carte initiale. De plus, l'ordre défini dans une σ orbite étant un ordre circulaire, le champ récepteur d'un sommet décrit un cycle dans le graphe $\mathcal{OG}$. La preuve que le champ récepteur d'un brin contient tous ces sommets est encore une fois un peu plus délicate et est basée sur le théorème suivant :

Théorème 9 *Soient une pyramide définie par une carte initiale $G = (\mathcal{B}, \sigma, \alpha)$ et n noyaux $K_1, \ldots, K_n$. Le champ récepteur d'un sommet de niveau $i \geq 1$, $\sigma_i^*(b_1) = (b_1, \ldots, b_p)$ vérifie :*

- *si K_i est un noyau de suppression :*

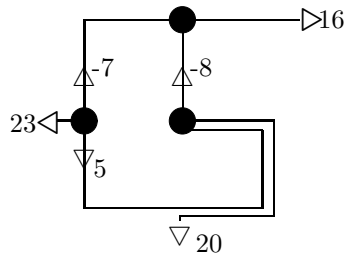
$$R_{\sigma_i^*(b_1)} = R_{\sigma_{i-1}^*(b_1)};$$

- *si K_i est un noyau de contraction, le sommet $\sigma_i^*(b)$ est associé à un arbre $\mathcal{T} = \bigodot_{j=1}^{p} CW_i(\alpha(b_j))$ dans G_{i-1} (Proposition 3). Si nous sélectionnons un brin d_i pour chaque sommet de $\mathcal{T}$, l'ensemble de brins $d_1, \ldots, d_q$ vérifie*

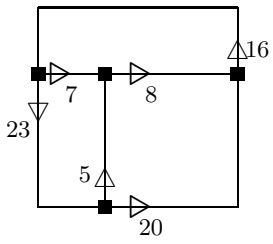
$$R_{\sigma_i^*(b_1)} = \bigcup_{j=1}^{q} R_{\sigma_{i-1}^*(d_j)}.$$

En d'autres termes, si K_i est un noyau de suppression, le champ récepteur d'un sommet survivant ne varie pas entre les deux niveaux. Cette propriété est analogue à celle vue dans le cadre des fenêtres de réduction (équation 4.29). Si K_i est un noyau de contraction, le sommet $\sigma_i^*(b)$ peut s'interpréter comme une union de tous les sommets définis au niveau $i-1$ et fusionnés dans $\sigma_i^*(b)$ au niveau i. Le champ récepteur associé à ce sommet est donc logiquement une union des champs récepteurs associés aux sommets fusionnés. Un simple raisonnement par induction basé sur le Théorème 9 permet de montrer que le champ récepteur d'un sommet de niveau i contient tous ses sommets de la carte de base. C'est donc une région au sens de la Définition 26. On peut de même montrer par induction que l'ensemble des champs récepteurs de sommets de niveau i forme une partition de l'ensemble des brins initiaux.

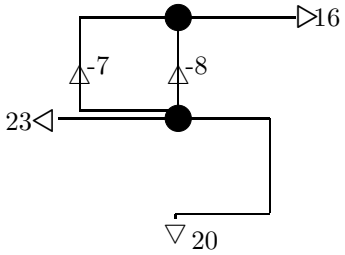
Afin d'illustrer la notion de champ récepteur, développons la pyramide représentée sur la figure 4.45 sur deux niveaux supplémentaires. Le noyau de suppression K_2 permet de simplifier la carte combinatoire en enlevant les arêtes multiples et les boucles vides (voir section 4.6.3 et équation 4.15). La carte ainsi simplifiée décrit une partition en 3 régions. Cette carte est à nouveau



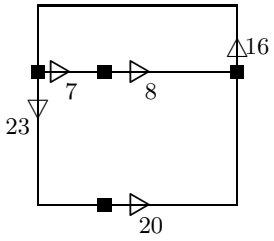
(a) $G_2 = (G/K_1) \setminus K_2$



(b) $\overline{G_2} = \overline{(G/K_1) \setminus K_2}$



(c) $G_3 = G_2/\alpha^*(5)$



(d) $\overline{G_3} = \overline{G_2} \setminus \alpha^*(5)$

FIG. 4.47 – Simplification de la carte représentée sur la Figure 4.45 suivie d'une nouvelle contraction par $K_3 = \alpha^*(5)$.

contractée par le noyau de contraction $K_3 = \alpha^*(5)$ qui fusionne les deux régions $\sigma_2^*(5)$ et $\sigma_2^*(-5)$ en une seule décrivant les deux dernières lignes de l'image. (figure 4.47(d)).

Le noyau K_2 étant un noyau de suppression, nous avons d'après le Théorème 9 :

$$\forall b \in \mathcal{BS}_2 \quad R_{\sigma_2^*(b)} = R_{\sigma_1^*(b)}.$$

Intéressons nous donc aux régions définies au niveau 3. L'arête $\alpha^*(5)$ étant la seule arête contractée les seuls chemins de connexion de niveau 3 non réduits à un singleton sont :

$$\begin{array}{lcl} CW_3(-23) & = & -23.5 \\ CW_3(8) & = & 8.-5. \end{array}$$

On aura donc $SC_3(-23) = -23.SC_2^*(23).5.SC_2^*(-5)$ et $SC_3(8) = 8.SC_2^*(-8).-5.SC_2^*(5)$ alors que toutes les autres séquences de connexion subissent simplement la transformation $SC_3(b) = b.SC_2^*(\alpha(b))$ (figures 4.48(a) et (b)). Le noyau K_3 étant un noyau de contraction, les champs récepteurs se déduisent aisément des séquences de connexion en permutant b et $\alpha(b)$ au début de chaque séquence (figure 4.48(c)). Finalement, les fenêtres de réduction des sommets sont construites en concaténant les fenêtres de réduction des brins de chaque sommet. On obtient ainsi, 3 cycles $R_{\sigma_3^*(-8)}$, $R_{\sigma_3^*(8)}$, $R_{\sigma_3^*(-23)}$ dans le graphe $\mathcal{OG}$ formant une partition de l'ensemble des brins de ce graphe. Le plongement de chaque région peut s'observer en comparant les cycles sur le graphe $\mathcal{OG}$ avec la carte initiale (figures 4.45). La région $R_{\sigma_3^*(8)}$ code la première ligne de l'image tandis que $R_{\sigma_3^*(-8)}$ code les deux lignes du bas et $R_{\sigma_3^*(-23)}$ code l'extérieur de l'image.

4.6.9 Résultats récents et conjectures

Le codage de la pyramide par les fonctions *niveaux* et **état**, permet d'envisager un stockage de celle-ci à l'aide d'une image de taille $(L+1)\times(H+1)$ où L et H représente respectivement la largeur et la hauteur de l'image. Étant donnée une carte G_0 correspondant à une carte des régions d'une grille 4-connexe, chaque sommet de $\overline{G_0}$ peut être associé à un coin de pixel ou pointel (section 4.6.1 et figure 4.45(b)). L'ensemble des pointels peut être stocké sur une image $(L+1)\times(H+1)$. Si nous convenons de stocker dans chaque pointel les niveaux des arêtes situés en haut et à gauche nous obtenons un stockage des niveaux de l'ensemble des arêtes. Par exemple, dans la figure 4.45(b), les pointels des 2 premières lignes stockeront les niveaux des arêtes suivantes :

Aucune arête	$\alpha^*(13)$	$\alpha^*(14)$	$\alpha^*(15)$
$\alpha^*(24)$	$\alpha^*(1), \alpha^*(7)$	$\alpha^*(2), \alpha^*(8)$	$\alpha^*(16), \alpha^*(9)$

Si la pyramide est limitée à n niveaux, chaque pointel stockera au plus $2\log_2(n)$ bits. Le coût total en mémoire sera donc de :

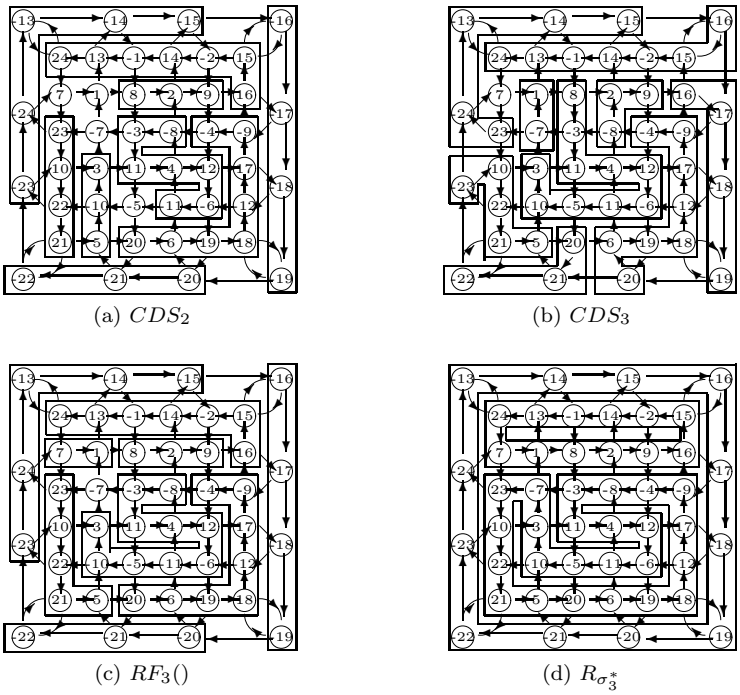
$$n + 2\log_2(n)(L+1)(H+1) \text{ bits}$$

où le premier n code le coût de stockage de la fonction **état**.

A titre de comparaison, le stockage de la géométrie d'une partition en utilisant la topologie inter-pixel est égal à [30] :

$$4(L+1)(H+1) \text{ bits.}$$

Si nous limitons par exemple la taille de la pyramide à 256 niveaux, le coût mémoire de la pyramide sera de $\mathcal{O}(16(L+1)\times(H+1))$. On a donc un rapport de 4 entre un stockage d'une partition et le stockage des cartes combinatoires codant 255 partitions.



(a) CDS_2 (b) CDS_3

(c) $RF_3()$ (d) $R_{\sigma_3^*}$

FIG. 4.48 – Les séquences de connexion d'ordre 2 (a) et 3 (b) avec les fenêtres de réductions de brins (c) et de sommets (d) de niveau 3.

Si nous stockons explicitement l'ensemble des niveaux de la pyramide, en utilisant par exemple la fonction Σ (Définition 23 et Table 4.2) et un codage implicite de l'involution α, chaque entrée du tableau Σ peut être codée avec $\log_2([\mathcal{B}|)$ bits. Le coût mémoire est donc de :

$$\log_2(|\mathcal{B}|)\sum_{i=0}^{n}\mathcal{BS}_i = \log_2(|\mathcal{B}|)\sum_{i=0}^{n}\frac{|\mathcal{B}|}{r^i} \approx |\mathcal{B}|\log_2(|\mathcal{B}|)\frac{r}{r-1} \text{ bits}$$

où $(\mathcal{BS}_i)_{i\in\{0,\ldots,n\}}$ représente l'ensemble des brins survivants de niveau i et r le facteur de décimation entre deux niveaux. Les coûts du codage implicite et explicite peuvent être comparés en utilisant la relation :

$$\frac{|\mathcal{B}|}{r^n} = 1 \Rightarrow n\log_2(r) = \log_2(|\mathcal{B}|).$$

Si nous prenons $r = 2$, le coût du stockage explicite est donc de :

$$2|\mathcal{B}|\log_2(|\mathcal{B}|) = 2|\mathcal{B}|n \approx 2n(L+1)(H+1) \text{ bits.}$$

Le coût mémoire d'un stockage explicite de la pyramide varie donc linéairement en fonction du niveau en utilisant un stockage explicite alors qu'il varie comme un log du niveau avec un stockage implicite.

Un avantage subsidiaire du codage implicite devrait être une étude plus aisée du plongement du champ récepteur d'un sommet. En effet, ce codage sous entend une association entre le numéro d'un brin et le lignel correspondant. Une telle association peut facilement être réalisée à l'aide d'une convention quelconque de numérotation des brins. Notons toutefois que nous n'avons pas défini précisément la notion de plongement d'un champ récepteur. Intuitivement, il est assez "clair" que le champ récepteur d'un sommet correspond à une région de l'image et que l'ensemble de ses brins correspond à l'ensemble des lignels appartenant à la région. Ce résultat n'a toutefois pas encore été démontré. Une piste certainement prometteuse consiste à expliciter la carte topologique codant une grille discrète comme la donnée d'un ensemble de pointel et lignel associé respectivement aux σ et α orbites de la carte combinatoire initiale. Il convient ensuite d'interpréter les contractions et suppressions en termes de suppression de pointels et de suppression ou de concaténation de lignels.

Une étude plus précise du plongement d'un sommet devrait également nous permettre d'accélérer les algorithmes calculant une carte combinatoire donnée de la pyramide (Algorithmes (a) et (b) figure 4.39). En effet, la complexité de ces algorithmes est proche de celle des Algorithmes calculant l'ensemble des cartes réduites de la pyramide (Algorithme 1 et 2). Ces algorithmes bien que linéaires dans le nombre de brins initiaux ne sont donc clairement pas optimum. Une possibilité d'optimisation se trouve très certainement dans le concept de brins internes et de brins frontières. Dans le cadre d'une carte des régions, il semble en effet, mais cela reste à démontrer que :

1. Les brins internes correspondent :
   - soit à des brins contractés,
   - soit à des brins supprimés parce qu'ils correspondent à des boucles éventuellement imbriquées. Ces brins peuvent être caractérisé de la façon suivante (figure 4.49(a)) :

$$\sigma^{2n-1}(b) = \alpha(b) \text{ avec } \forall i \in \{1,\ldots,n-1\} b_i = \sigma^i(b) \text{ vérifie } \sigma^{2(n-i)-1}(b_i) = \alpha(b_i)$$

2. Les brins frontières correspondent
   - soit à des brins survivants,

(a) Boucle imbriquée avec $n = 3$

(b) Arête double

FIG. 4.49 – Un exemple de boucle imbriquée et d'arête double

– soit à des brins supprimés parce qu'ils correspondent à des arêtes doubles. Ces brins peuvent être caractérisés par la relation $\varphi^2(b) = b$ (figure 4.49(b)).

Si nous affinons la notion de noyau de suppression en codant la raison pour laquelle un brin a été supprimé (boucle ou arête double), nous devrions donc pouvoir caractériser les brins internes et frontières. Si $\mathcal{F}_i$ représente l'ensemble des brins frontières de niveau i, nous pouvons définir l'intérieur et la frontière du champ récepteur d'un sommet comme :

– frontière d'un champ récepteur :

$$\partial R_{\sigma_i^*(b)} = R_{\sigma_i^*(b)} \cap \mathcal{F}_i;$$

– intérieur d'un champ récepteur :

$$\overset{\circ}{R}_{\sigma_i^*(b)} = R_{\sigma_i^*(b)} \cap (\mathcal{B} - \mathcal{F}_i) = R_{\sigma_i^*(b)} - \partial R_{\sigma_i^*(b)}.$$

où l'ordre dans $\partial R_{\sigma_i^*(b)}$ et $\overset{\circ}{R}_{\sigma_i^*(b)}$ est déduit de celui de $R_{\sigma_i^*(b)}$.

Si nous nous référons à l'exemple de la section précédente, on a (figures 4.48(d) et 4.50) :

$$\partial R_{\sigma_3^*(16)} = 16.15.14.13.24.7.8.9.$$

Cet ensemble de brins correspond à l'ensemble des lignels délimitant $R_{\sigma_3^*(16)}$, on a de plus une orientation dans le sens positif autour de la région.

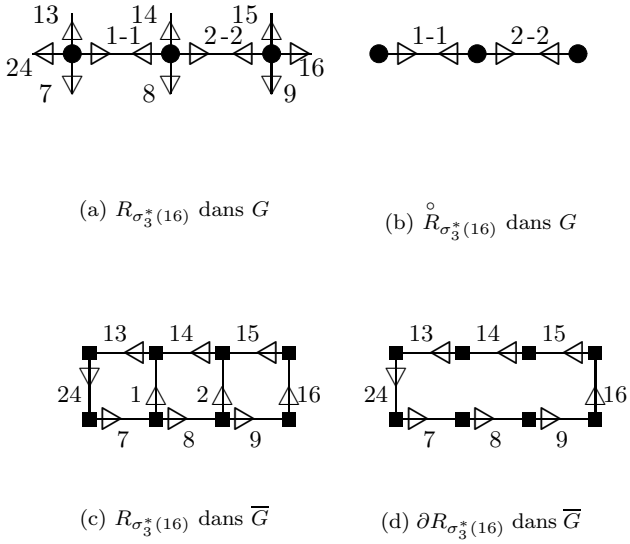
La frontière d'un champ récepteur de sommet semble donc pouvoir s'interpréter comme une frontière de région. L'ensemble des brins de la frontière doit pouvoir être retrouvé par la conjecture suivante :

$$\forall j \in \{1, \ldots, n\}\ b_{j+1} = \varphi^{n_j}(\alpha(b_j)) \text{ avec } n_j = Min\{k \in \mathbb{N}^* \mid \varphi^k(\alpha(b_j)) \in \mathcal{F}_i\} \tag{4.34}$$

où $\partial R_{\sigma_i^*(b)} = b.b_1 \ldots, b_n$.

Notez que puisque nous utilisons une carte des régions, les φ orbites de brins sont forcément bornées (par 4 dans le cas de la 4-connexité). Si la conjecture précédente est vraie, la frontière

d'une région doit donc pouvoir être déterminée dans un temps proportionnel à son nombre de brins. De plus, la frontière de la région comprenant les brins frontières comprend également les brins survivants. Par construction d'une région l'ordre des brins dans $\sigma_i^*(b)$ est respecté dans $R_{\sigma_i^*(b)}$. Il est donc également respecté dans $\partial R_{\sigma_i^*(b)}$. On peut donc retrouver les σ_i successeurs des brins survivants de niveau i en parcourant la frontière des champs récepteurs de sommet plutôt que les champs récepteurs eux mêmes. Cette dernière propriété devrait nous permettre d'optimiser les algorithmes calculant une carte réduite à partir de la base de la pyramide.



(a) $R_{\sigma_3^*(16)}$ dans G

(b) $\overset{\circ}{R}_{\sigma_3^*(16)}$ dans G

(c) $R_{\sigma_3^*(16)}$ dans $\overline{G}$

(d) $\partial R_{\sigma_3^*(16)}$ dans $\overline{G}$

FIG. 4.50 – Le champ récepteur $R_{\sigma_3^*(16)}$.

4.6.10 Une implémentation des pyramides combinatoires

L'implémentation que nous avons réalisée est basée sur un codage implicite de la pyramide. Celle-ci se compose donc d'une carte combinatoire codant le sommet de la pyramide et d'un tableau codant le niveau de chaque brin de la carte initiale. Nous avons décomposé l'opération de suppression en deux étapes de façon à pouvoir distinguer les opérations de suppression d'arêtes doubles des opérations de suppression de boucle (section 4.6.9). La réduction d'une carte combinatoire s'effectue donc en trois étapes :

1. contraction d'arêtes à l'aide de noyaux de contraction ;
2. suppression des boucles redondantes ;
3. suppression des arêtes doubles.

Cette décomposition pénalise légèrement les algorithmes parallèles puisque les opérations de suppression de boucles et d'arêtes redondantes doivent être effectuées séquentiellement. Elle permet toutefois de distinguer l'opération de suppression d'arêtes doubles et de coder implicitement la

fonction **état** comme une opération modulo à partir du niveau. De fait, les trois opérations : contraction, suppression de boucles redondantes et suppression d'arêtes doubles sont effectuées séquentiellement. Nous avons donc pour tout brin b :

- si $niveau(b)$ mod $3 = 1$, **état**(niveau(b))= Contracté ;
- si $niveau(b)$ mod $3 = 2$, **état**(niveau(b))= Supprimé et b correspond à une boucle redondante ;
- si $niveau(b)$ mod $3 = 0$, **état**(niveau(b))= Supprimé et b correspond à une arête double.

Nous avons implémenté deux méthodes permettant de coder l'opération de contraction. L'opération de suppression est déterminée à partir des arêtes redondantes de la carte contractée et est donc effectuée automatiquement.

4.6.10.a Contraction parallèle

Notre première opération de contraction est basée sur les méthodes de Jolion [114] et Montanvert [146]. Celle-ci prend en paramètre les pointeurs de fonctions suivants :

- `double intérêt(pyramide, niveau,sommet)` : renvoi une valeur indiquant l'intérêt d'un sommet. Les sommets dont la valeur représente un minima local seront sélectionnés comme sommets survivants (section 4.3.1) ;
- `double similitude(pyramide, niveau,sommet,sommet)` : cette fonction renvoie une valeur indiquant la similitude entre deux sommets. Chaque sommet non survivant est attaché au sommet survivant adjacent dont il est le plus similaire (section 4.3.2) ;
- `booleen lambda(pyramide,niveau,sommet,sommet)` : une valeur fausse renvoyée par cette fonction indique que les deux sommets ne doivent pas être fusionnés quelque soit leur similitude (section 4.3.2).

Notons que ces trois fonctions utilisent généralement des attributs attachés à chaque sommet. Jolion [114] définit par exemple la fonction `interet` à partir de la variance des niveaux de gris et la fonction `similitude` en fonction de la distance entre les niveaux de gris moyens des régions associées aux sommets.

Notre première méthode de contraction utilise donc les fonctions `intérêt`, `similitude` et `lambda` pour calculer un noyau (section 4.3.1). L'ensemble des arêtes à contracter forme alors une forêt composée d'arbres de profondeur 1. La technique utilisée pour construire ces arbres dépend de la valeur des pointeurs de fonctions passés à notre fonction de contraction. Les différents passages de paramètres sont les suivants :

1. si les pointeurs de fonctions `interet` et `similitude` sont non nuls alors le pointeur de fonction `lambda` sera utilisé s'il n'est pas nul ;
2. si les pointeurs de fonctions `interet` et `similitude` sont nuls alors le pointeur de fonction `lambda` ne doit pas être nul.

Nous utilisons les fonctions `interet` et `similitude` dans les cas où le calcul des noyaux de contractions (et donc des régions) doit être effectué durant l'opération de contraction. Si nous faisons un parallèle avec les algorithmes de croissance de région, la fonction `interet` nous permet de sélectionner les germes du processus de croissance de région tandis que la fonction `similitude` permet de faire croître ces même germes par des opérations de fusion. Notons toutefois que l'ensemble des sommets survivants doit former un noyau (section 4.3.1) afin d'obtenir un facteur de réduction fixe entre deux niveaux de la pyramide. Cette contrainte peut être relâchée en excluant du graphe certaines arêtes à l'aide de la fonction `lambda` (section 4.3.1).

La fonction `lambda` peut également être utilisée seule lorsque l'on connaît *a priori* les regroupements de régions qui doivent être effectués. Par exemple, dans le cas de l'analyse des composantes connexes issues d'une opération de quantification (section 3.2), la fonction `lambda` se définie simplement au niveau de base en indiquant qu'elle doit renvoyer :

- vrai si les sommets adjacents ont la même couleur représentative ;
- faux sinon.

Dans ce cas, l'utilisation des fonctions `interet` et `similitude` est inutile et notre opération de contraction est similaire à celle définie par Montanvert [146](section 4.3). Les pointeurs sur les fonctions `interet` et `similitude` sont donc, dans ce cas, positionnés à nul. Notre implémentation de l'ensemble des sommets nous permet d'affecter automatiquement un entier à chaque sommet. Cette valeur est utilisée de préférence à la variable aléatoire utilisée par Montanvert pour ne pas avoir à reparcourir les sommets de la carte dans une implémentation séquentielle.

4.6.10.b Contraction par un algorithme séquentiel

Afin de mieux appréhender l'intérêt d'un algorithme spécifiquement conçu pour une architecture séquentielle, étudions la complexité séquentielle de l'algorithme précédent sur un cas simple : supposons que le but de l'opération de contraction est de réduire une image de taille $2^{2n} \times 2^{2n}$ à un seul sommet. Le même type de raisonnement peut être appliqué pour la contraction d'une région d'une image en un seul sommet. Nous pouvons appliquer pour ce type d'application la méthode de Montanvert en utilisant uniquement la fonction `lambda`. Selon Montanvert [146], le facteur de réduction entre deux niveaux est approximativement égal à 4 et le nombre d'itérations nécessaires pour obtenir une noyau (section 4.3.1) est égal à 5. La pyramide contractant l'image $2^{2n} \times 2^{2n}$ en un seul sommet sera donc composée de n niveaux et nous devrons parcourir l'ensemble des sommets de chaque niveau en moyenne 5 fois. La complexité séquentielle de cet algorithme est donc :

$$5 \sum_{i=0}^{n} \frac{2^{2n} \times 2^{2n}}{4^i} \approx 5(2^{2n} \times 2^{2n}) \frac{4}{4-1} = 6.6(2^{2n} \times 2^{2n}).$$

L'algorithme parallèle nous oblige donc à parcourir au moins 6 fois l'image pour la coder en un seul sommet. Cette complexité importante est due à deux facteurs :

1. le nombres de niveaux de la pyramide est due à l'utilisation de noyaux qui restreignent les opérations de contraction à des arbres de profondeur 1 ;
2. les itérations nécessaires à chaque niveau sont dues à la méthode utilisée par Montanvert. Cette méthode conçue dans le cadre d'une implémentation parallèle, permet une indépendance des données distribuées sur les différents processeurs. Cette indépendance des données n'est plus nécessaire dans le cadre d'un calcul séquentiel.

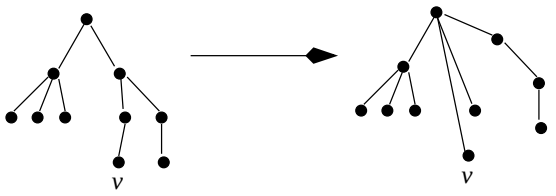
Notre méthode séquentielle est basée sur une définition directe des noyaux de contractions (section 4.5.2) définissant les différentes régions. Notre méthode va donc construire pour chaque région un arbre qui recouvre l'ensemble des sommets de celle-ci. Si la carte à contracter est située à la base de la pyramide, chaque sommet est associé à un pixel. L'utilisation de noyaux de contraction plutôt que de noyaux permet donc de s'affranchir de la contrainte imposant aux arbres d'avoir une profondeur 1. Nous supprimons également les itérations nécessaires pour construire de tels arbres en utilisant des techniques d'unir-trouver importées en traitement d'image par Fiorio [77, 76]. Notre adaptation des techniques d'unir-trouver de Fiorio, initialement conçues pour des pixels est la suivante :

– nous attachons à chaque sommet du graphe l'indice de son père. Initialement chaque sommet est son propre père et nous avons **père**$(v) = v$ pour tout sommet v ;
– deux sommets sont attachés au même arbre si ils doivent être fusionnés et si ils n'ont pas le même père. Lors de l'opération de fusion de deux sommets v_1 et v_2, le père d'un des deux sommets, disons v_1, est mis à jour par : **père**$(v_1) = v_2$.

Cette mise à jour de la relation **père** peut impliquer de multiples indirections lorsque les arbres ont une profondeur supérieure à 1. En effet, si le sommet v_1 est fusionné à v_2 puis v_2 est fusionné à v_3 nous auront les relations :

$$\textbf{père}(v_1) = v_2 \text{ et } \textbf{père}(v_2) = v_3$$

Il faut alors une indirection pour déterminer le père de v_1. Afin de corriger ce problème, nous utilisons l'algorithme `trouve_compresse` [77] (algorithme 3) qui remet à jour les pères de l'ensemble des sommets se trouvant sur le chemin (dans l'arbre) entre le sommet passé en paramètre et la racine de l'arbre.




```
trouve_compresse(tableau père,sommet v)
{
    si (père(v) == v )
        renvoyer v ;
    sinon
    {
        père(v) = trouve_compresse(père,
                                   père(v))
        renvoyer père(v)
    }
}
```

Algorithme 3: *L'algorithme trouve_compresse. La figure en regard de l'algorithme montre l'effet de celui-ci sur un arbre.*

L'algorithme `trouve_compresse` nous permet d'avoir un identifiant unique pour chaque arbre du noyau de contraction. La construction de celui-ci est alors réalisée par l'algorithme 4. La contraction effective de la carte combinatoire à partir du noyau de contraction est alors réalisée en utilisant les chemins de connexion (section 4.6.3, figure 4.32)

Les temps nécessaires à notre méthode séquentielle et à celle basée sur les méthodes de Jolion et Montanvert (section 4.6.10.a) pour contracter une image de taille $2^n \times 2^n$ en un seul sommet sont représentés sur la table 4.4. L'ordinateur séquentiel utilisé pour cette expérience est un Athlon XP1800 $1533Mz$.

4.6.10.c Sauvegarde et restauration de pyramides combinatoires

Nous avons utilisé le format de sauvegarde décrit dans la section 4.6.9 pour sauver la pyramide. Celle-ci est donc sauvegardée dans une image de taille $(L + 1) \times (H + 1)$ où L et H représentent respectivement la largeur et la hauteur de l'image. Chaque pixel de cette image stocke le niveau de deux arêtes en fonction de la convention décrite dans la section 4.6.9. Afin de pouvoir visualiser nos

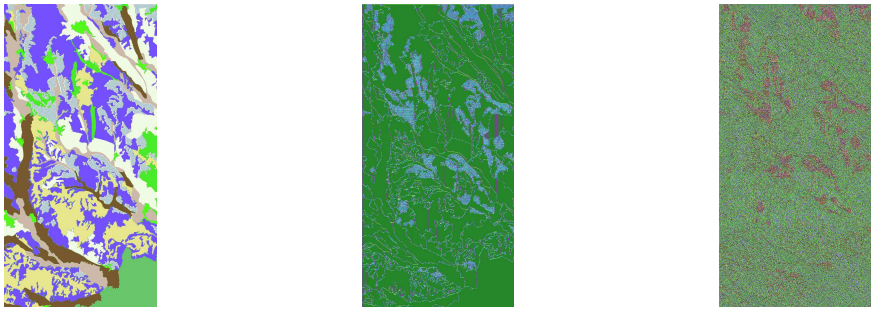
```
contraction_sequentielle(pyramide P, fonction lambda)
{
    noyau de contraction K = ∅
    pour tout sommet v au plus haut niveau de la pyramide
            père(v) = v
    Pout tout brin b de la carte au sommet de la pyramide
    {
        v1= sommet de b
        v2= sommet de α(b)
        Si (trouve_compresse(père, v1) !=trouve_compresse(père, v2) et
            lambda(P,v1, v2))
        {
            ajouter b au noyeau de contraction K
            père(v1) = v2
        }
    }
    renvoyer K
}
```

Algorithme 4: *Détermination du noyau de contraction*

temps	Algorithme séquentiel	Jolion-Montanvert
32×32	0.00 s	0.06 s
64×64	0.01 s	0.41 s
128×128	0.05 s	3.34 s
256×256	0.23 s	26.15 s
512×512	2.35 s	3 min 27 s
1024×1024	1 min 5 s	27 min 37 s

TAB. 4.4 – Temps requis par nos algorithmes (sections 4.6.10.a et 4.6.10.b) pour contracter une image de taille variable en un seul sommet. Les symboles min et s représentent respectivement les minutes et le secondes.

images de sauvegardes, nous avons stocké le niveau de chaque arête sur 12 bits. Ce format permet donc de stoker des pyramides composées de $2^{12} = 4096$ niveaux. Sachant que l'obtention de chaque partition nécessite deux étapes de suppression d'arêtes, le nombre de partitions correspondant à ces niveaux est de $\frac{2^{12}}{3} = 1365$. Nous utilisons comme format de sauvegarde le format *ppm* qui permet de coder une image RGB sur 24 bits. Nous utilisons la composante R et les 4 bits de poids faible de la composante G pour la première arête. La seconde arête est codée sur les 4 bits de poids fort de la composante G et la composante B. La figure 4.51 représente en fausses couleurs, deux sauvegardes d'une pyramide codant les composantes connexes de l'image représentée sur la figure 4.51(a). L'image de la figure 4.51 est obtenue à partir de notre méthode séquentielle (section 4.6.10.b) tandis que la figure 4.51(c) est obtenue à partir de notre méthode parallèle (section 4.6.10.a). La structure de l'image apparaît clairement dans la figure 4.51. En effet, l'utilisation de noyaux de contractions de profondeur quelconque permet d'appliquer à un niveau la même opération à un grand nombre d'arêtes. Inversement, le nombre élevé d'itérations effectué par l'algorithme parallèle se traduit par un nombre plus élevé de couleurs sur la figure 4.51(c).



(a) Image originale (b) Pyramide séquentielle (c) Pyramide parallèle

FIG. 4.51 – Sauvegarde de pyramides combinatoires codant les composantes connexes de la figure (a). La figure (b) a été obtenue à l'aide de notre méthode séquentielle (section 4.6.10.b) tandis que la figure (b) a été obtenue par notre méthode parallèle (section 4.6.10.a).

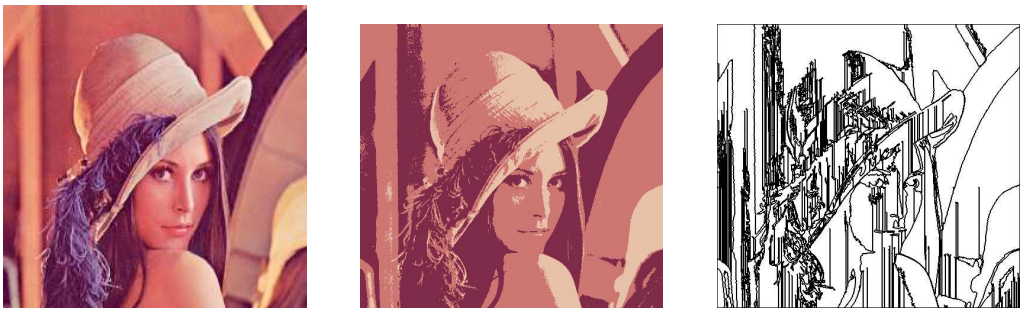
Notre implémentation des pyramides combinatoires étant basé sur un codage implicite des pyramides, le chargement d'une pyramide à partir d'un fichier de sauvegarde nécessite de rechercher le niveau de chaque brin de la carte initiale et de calculer la carte située au sommet de la pyramide. Nous utilisons pour cette dernière opération la conjecture définie dans la section 4.6.9 (équation 4.34). Cette équation nous permet de retrouver la carte combinatoire située au sommet de la pyramide en ne parcourant que les frontières de la partition associées à cette carte. Notre procédure de chargement effectue donc les opérations suivantes :

1. parcourir l'image de sauvegarde afin de déterminer le nombre de brins initiaux pour chaque niveau ;
2. parcourir une seconde fois l'image de sauvegarde afin de stocker les brins initiaux dans un tableau. La place nécessaire pour chaque niveau a été réservée à la fin de la première étape ;
3. pour chaque brin de plus haut niveau parcourir la frontière qui lui est associée en utilisant l'équation 4.34. Le dernier brin parcouru est le σ-successeur du brin initial.

Notre procédure de restauration de la pyramide réclame donc deux parcours de l'image de sauvegarde et un parcours des frontières de la partition définies au sommet de la pyramide.

4.6.10.d Obtention des contours d'une partition

Les contours d'une partition sont obtenus à nouveau par l'équation 4.34 (section 4.6.9) qui nous permet de retrouver l'ensemble des lignels définissant une partition en parcourant uniquement l'ensemble des frontières. La figure 4.52 représente les contours d'une partition produite par un algorithme de quantification [41] (section 3.2). Cette partition est codée par une pyramide utilisant une fonction `lambda` qui renvoie vrai si deux sommets codent deux pixels ou deux régions de même couleur (section 4.6.10.a). Les contours verticaux sur cette figure correspondent aux arêtes fictives codant les relations d'inclusion entre les régions. Les nombreuses petites régions de cette image peuvent être supprimées en rajoutant une nouvelle étape qui fusionne à une région adjacente toute région dont la taille est inférieure à seuil (fixé ici à 20 pixels). L'image résultat est représentée sur la figure 4.53 sans les contours des arêtes fictives.



(a) Image originale (b) Image quantifiée (c) Contours

FIG. 4.52 – Contours (c) des composantes connexes obtenues à partir d'un algorithme de quantification (b) appliqué sur l'image Lenna (a).

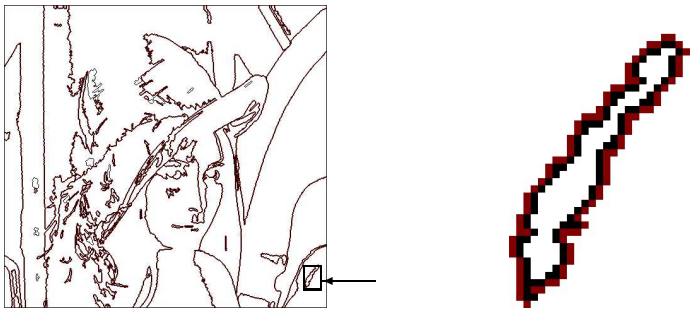
Notons que les lignels que nous parcourons séparent des pixels appartenant à des régions différentes. Nous pouvons accéder à ces pixels en calculant les sommets de la carte de base qui possèdent les deux brins associés à chaque lignel. La figure 4.54 représente un grossissement des contours de l'image Lenna dans lesquels nous avons affecté des couleurs différentes aux pixels de part et d'autres de chaque frontière. Dans le cadre d'un calcul de la frontière d'une région et non plus de l'ensemble des frontières de la partition, l'orientation définie sur chaque frontière nous permet de définir la frontière intérieure d'une région. En effet, pour un sommet v et un brin b appartenant à ce sommet, l'ensemble des brins frontières calculés par l'équation 4.34 appartient au champs récepteurs de b. Le champ récepteur d'un sommet étant défini comme la concaténation des champs récepteurs de ses brins (définition 31), les sommets de la carte de base appartiendront au champ récepteur de v et donc à la région associée à celui-ci.

(a) Image bruitée

(b) Image (a) sans arêtes fictives

(c) Suppression des petites régions

FIG. 4.53 – Suppression des régions de taille inférieures à 20 dans l'image (a).



(a) Contours intérieur

(b) Région en bas à droite

FIG. 4.54 – Zoom (b) de la région située en bas à droite de l'image des contours intérieurs de Lenna (a). Les pixels de part et d'autre de chaque frontière ont été tracés avec les couleurs noires et rouge.

4.6.11 Conclusions et perspectives

Nous avons décrit dans la section 4.6 un nouveau type de modèle pyramidal basé sur les cartes combinatoires. Ce modèle se distingue des modèles basés sur des graphes simples ou des graphes duaux (sections 4.3 et 4.5) par un codage implicite des sommets et un codage explicite de l'orientation. Ajoutons que les cartes combinatoires sont le seul modèle basé explicitement sur un codage d'une carte topologique (section 4.6.1). Cette mise en correspondance entre les arêtes de la carte combinatoire et les frontières de la partition permet de décrire finement celle-ci. Nous avons décrit dans les sections 4.6.2 et 4.6.3 les principaux outils permettant de construire explicitement la pyramide comme un suite de cartes combinatoires successivement réduites. Nous avons ensuite étudié des outils permettant de combiner des noyaux de même type (section 4.6.4) ou de types différents (section 4.6.5). Les outils décrits dans les sections 4.6.5 et 4.6.6 nous ont permis de proposer un nouveau codage de la pyramide basé uniquement sur le niveau où disparaît un brin et l'opération qui a conduit à sa disparition. Ce type de codage a été utilisé pour retrouver soit un niveau particulier d'une pyramide (section 4.6.6) soit l'ensemble des niveaux (section 4.6.7). Enfin nous avons décrit dans la section 4.6.8 les notions de fenêtres de réduction et de champs récepteurs dans le cadre des pyramides combinatoires. Notez que les fenêtres de réduction et les champs récepteurs sont respectivement basés sur les chemins et les séquences de connexion qui ont été utilisés pour construire la pyramide. Ces outils servent donc à la fois dans le cadre d'une analyse ascendante pour construire la pyramide mais également dans le cadre d'une analyse descendante pour retrouver l'ensemble des fils d'un brin ou d'un sommet.

Finalement, nous pouvons mentionner des résultats extrêmement intéressants de Guillaume Damiand et Pascal Lienhardt [61] concernant l'extension à n dimensions de nos travaux définis pour des cartes $2D$. Ces travaux sont basés sur une extension des cartes combinatoires appelée cartes combinatoires généralisées qui permettent de coder une partition d'un espace de dimension n en objets avec ou sans bords orientables ou non orientables :

Définition 32 Cartes généralisée

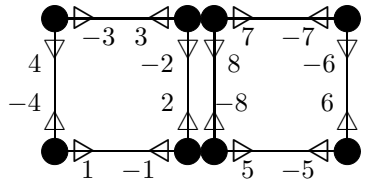
Soit $n \geq 1$. Une carte combinatoire généralisée de dimension n (n-G-carte) est une algèbre $G = (\mathcal{B}, \alpha_0, \ldots, \alpha_n)$ où :

1. *$\mathcal{B}$ est un ensemble finis de brins.*
2. *$\forall i \in \{0, \ldots, n\}$ α_i est une involution*
3. *$\forall i \in \{0, \ldots, n\}$, $\forall j \in \{i+2, \ldots, n\}$ $\alpha_j \circ \alpha_i$ est une involution.*

En dimension 2, une n-G-carte est un quadruplet $G = (\mathcal{B}, \alpha_0, \alpha_1, \alpha_2)$ où l'involution α_0 permet de retrouver les deux brins d'une même arête et l'involution α_1 associe deux brins incidents au même sommet dans une même face. Enfin, l'involution α_2 met en relation deux brins associés à la même arête et au même sommet mais dans deux faces différentes (figure 4.55).

La permutation α_0 joue donc un rôle identique à notre permutation α, alors que notre permutation σ correspond à $\alpha_2 \circ \alpha_1$. Plus précisément, il a été montré [132] que le triplet $(\mathcal{B}, \alpha_0 \circ \alpha_2, \alpha_1 \circ \alpha_1)$ définit une carte appelée carte des hyper-volumes de G.

Une cellule de dimension i d'une $n - G$-carte se définit comme l'orbite d'un brin b par le groupe de permutations engendré par $< \alpha_0, \ldots, \alpha_{i-1}, \alpha_{i+1} \ldots, \alpha_n >$ [61]. Ce dernier groupe de permutations est notée $<>_{N-i}$ et la i-cellule d'un brin b est donc noté $<>_{N-i}(b)$. La i cellule d'un brin b correspond donc à l'ensemble des brins que l'on peut atteindre à partir de b en utilisant toutes les involutions sauf α_i. Intuitivement, une cellule de dimension 0 correspond à un point



(a) $(\{-8,\ldots,-1,1,\ldots,8\},\alpha_0,\alpha_1,\alpha_2)$

$$\begin{array}{lcr}
\alpha_0(2) & = & -2 \\
\alpha_1(2) & = & -1 \\
\alpha_2(2) & = & -8 \\
 & & \\
<>_{N-0}(2) & = & \{2,-8,5,-1\} \\
<>_{N-1}(2) & = & \{2,-2,8,-8\} \\
<>_{N-2}(2) & = & \{2,-2,3,-3,4,\\
 & & -4,1,-1\}
\end{array}$$

(b) α successeurs et i-cellules de 2

FIG. 4.55 – Une 2-G-carte (a) et les α successeurs et i orbites de 2(b)

tandis que les cellules de dimensions 1, 2 et 3 correspondent respectivement à une arête, une face et un volume (figure 4.55).

En dimensions 2, les opérations de contraction et de suppression peuvent s'interpréter en termes d'opérations sur des cellules. En effet :

- la contraction d'une arête peut s'interpréter comme la contraction d'une cellule de dimension 1 ;
- la suppression d'une boucle peut s'interpréter comme la suppression d'une cellule de dimension 1 ;
- ma suppression d'une arête double peut simultanément s'interpréter comme la suppression d'une cellule de dimension 0 dans le dual ou la contraction d'une cellule de dimension 2 dans le primal.

La simplification d'une partition d'un espace de dimension n peut donc s'exprimer en termes de suppressions et contractions de cellules de dimensions i. On peut notamment définir une notion analogue à celle des chemins de connexion :

Définition 33 Chemins de connexion dans des G cartes

Soit un ensemble de brins K. On dira que K définit un ensemble d'i-cellules si et seulement si :

$$K = <>_{N-i}(K).$$

Dans ce cas, le chemin de connexion de tout brin $b \in \mathcal{BS} = \mathcal{B} - K$ est défini par :

- *si K désigne un ensemble d'i-cellules à supprimer tel que $i = n-1$ ou*

$$\forall b \in K \quad \alpha_{i+2} \circ \alpha_{i+1}(b) = \alpha_{i+1} \circ \alpha_{i+2}(b)$$

alors le chemin de connexion de b noté $CW^i(b)$ est défini par :

$$CW^i(b) = b.(\alpha_{i+1} \circ \alpha_i(b)) \ldots (\alpha_{i+1} \circ \alpha_i(b))^{n-1} \text{ avec } n = Min\{k \in \mathbb{N} \mid (\alpha_{i+1} \circ \alpha_i(b'))^k \circ \alpha_i(b) \in \mathcal{BS}\};$$

– *si K désigne un ensemble d'i-cellules à contracter tel que $i = 1$ ou*

$$\forall b \in K \quad \alpha_{i-2} \circ \alpha_{i-1}(b) = \alpha_{i-1} \circ \alpha_{i-2}(b)$$

alors le chemin de connexion de b noté $CW^i(b)$ est défini par :

$$CW^i(b) = b.(\alpha_{i-1} \circ \alpha_i(b)) \dots (\alpha_{i-1} \circ \alpha_i(b))^{n-1} \text{ avec } n = Min\{k \in \mathbb{N} \mid (\alpha_{i-1} \circ \alpha_i(b'))^k \circ \alpha_i(b) \in \mathcal{BS}\}.$$

La condition $K =<>_{N-i} (K)$ correspond à la condition $K = \alpha^*(K)$ utilisée dans le cadre des cartes $2D$. De la même façon que l'on ne désire pas appliquer une opération sur une moitié d'arête pour les cartes $2D$, l'on désire appliquer l'opération de contraction ou suppression sur l'ensemble des brins d'une $i-$cellule dans le cadre des cartes généralisées. Les conditions $\alpha_{i+2} \circ \alpha_{i+1}(b) = \alpha_{i+1} \circ \alpha_{i+2}(b)$ et $\alpha_{i-2} \circ \alpha_{i-1}(b) = \alpha_{i-1} \circ \alpha_{i-2}(b)$ utilisées respectivement pour les suppressions et les contractions correspondent intuitivement aux configurations d'arêtes pour lesquelles une simplification est possible. Ces cas correspondent donc à notre suppression de boucles directes et d'arêtes doubles utilisée pour simplifier la partition après l'application d'un noyau de contraction.

Notez la similitude entre la définition des chemins de connexion de la Définition 33 et celle donnée dans le cadre des cartes $2D$ (Définition 18). Cette similitude entre la forme des chemins de connexion s'étend aux résultats que l'on peut obtenir à partir de ceux-ci. En effet, pour un ensemble d'$i-$cellules K et un brin $b \in \mathcal{BS}$ tel que $CW^i(b) = b.b_1 \dots .b_p$, nous avons [61] :

$$\begin{cases} \forall j \in \{0, \dots, n\}, j \neq i & \alpha'_j(b) = \alpha_j(b) \\ & \alpha'_i(b) = \alpha_i(b_p) \end{cases}$$

où $G' = (\mathcal{BS}, \alpha'_0, \dots, \alpha'_n)$ représente la carte issue de la contraction ou suppression de K.

Cette dernière équation est très similaire à l'équation 4.12 obtenue dans la cadre des cartes $2D$. G. Damiand et P. Lienhardt proposent dans le même article [61] une généralisation de ces résultats à des noyaux comprenant simultanément des contractions et des suppressions de cellules de dimensions quelconques. Il reste toutefois à interpréter ces résultats en termes de fusions de régions et de simplification de partitions et à étudier le plongement des brins d'une carte réduite.

4.7 Lexique des principaux symboles utilisés

- b^i ..... brin survivant de niveau i dont la séquence de connexion contient b
- $\mathcal{BS}_i$ ..... ensemble des brins survivants de niveau i
- CW ..... chemin de connexion de b
- $\mathcal{DS}_b$ ..... ensemble des niveaux pour lesquels b à un successeur dans la séquence de connexion qui le contient
- $\mathcal{E}$ ..... ensemble des arêtes d'une carte topologique
- E_i ..... ensemble des arêtes du graphe G_i
- $\mathcal{F}_i$ ..... ensemble des frontières de niveau i
- $FR_{\sigma_i^*(b)}$ ..... fenêtre de réduction de niveau i du sommet $\sigma_i^*(b)$
- $\mathcal{G}$ ..... ensemble des points appartenant a une carte topologique
- G ..... graphe ou carte combinatoire
- $\overline{G}$ ..... dual de graphe ou de la carte G
- G_i ..... graphe ou carte de niveau i dans une pyramide
- $Ind_i(b)$ ..... indice du brin b dans la séquence de connexion qui le contient
- K ..... noyau de contraction ou de suppression
- $\mathbf{L}i(b)$ ..... niveau maximum atteint dans la séquence de connexion définie par b
- $\mathcal{L}_i(b)$ ..... minimum entre le niveau de b et le niveau maximum atteint après b dans la séquence de connexion le contenant
- $\mathbf{l}_{i,b}(j)$ ..... index du premier dans la séquence de connexion de niveau i contenant b dont le niveau est supérieur ou égal à j
- N ..... ensemble des arêtes non survivantes dans un noyau de contraction
- N_{k+1} ..... ensemble des sommets survivants dans le processus de décimation amélioré défini par Jolion
- $\mathcal{OG}$ ..... graphe orienté codant une carte planaire
- p_i ..... variable booléenne indiquant si un sommet survit lors de la définition d'un noyau par la méthode de Meer et Montanvert
- q_i variable booléenne indiquant si un sommet reste candidat lors de la définition d'un noyau par la méthode de Meer et Montanvert
- $RW_i(b)$ ..... fenêtre de réduction de niveau i de b
- $R_{\sigma_i^*(b)}$ ..... champ récepteur d'un sommet de niveau i
- $\overset{\circ}{R}_{\sigma_i^*(b)}$ ..... intérieur de la région $R_{\sigma_i^*(b)}$
- $\mathcal{T}$ ..... arbre
- $RF_i(b)$ ..... champ récepteur de niveau i du brin b
- r ..... facteur de réduction
- S ..... ensemble des sommets survivants dans un noyau de contraction
- $\mathcal{SC}_i(b)$ ..... séquence de connexion de niveau i de b
- $\mathbf{succ}i, b(j)$ ..... premier brin de niveau supérieur à j rencontré après b dans la séquence de connexion de niveau i qui le contient
- V ..... ensemble des sommets d'un graphe
- $\mathcal{V}$ ..... ensemble des sommets d'une carte topologique
- V_i ..... ensemble des sommets du graphe de niveau i dans une pyramide irrégulière
- x_i ..... variable aléatoire utilisée dans le processus de décimation de Meer et Montanvert.
- α ..... involution codant les arêtes dans les cartes combinatoires

- α_K ..... involution α de la carte des chemins de connexion
- ϵ ..... séquence de brins vides
- Σ ..... fonction codant explicitement toutes les permutations σ des cartes d'une pyramide combinatoire
- σ ..... permutation codant l'ordre des arêtes autour de chaque sommet dans les cartes combinatoires
- σ_K ..... permutation σ de la carte des chemins de connexion
- φ ..... permutation codant l'ordre des arêtes autour de chaque face dans les cartes combinatoires
- φ_K ..... permutation φ de la carte des chemins de connexion
- $\partial R_{\sigma_i^*(b)}$ ..... frontière de la région $R_{\sigma_i^*(b)}$.

Chapitre 5

Programme de recherche

Les thèmes abordés dans ce mémoire bien que se rapportant tous au traitement et à l'analyse d'image sont extrêmement divers et concernent à la fois des traitements bas niveaux et des traitements hauts niveaux. Notre objectif à moyen terme est d'arriver à combiner tous ces éléments ou à établir des collaborations entre ces différentes activités.

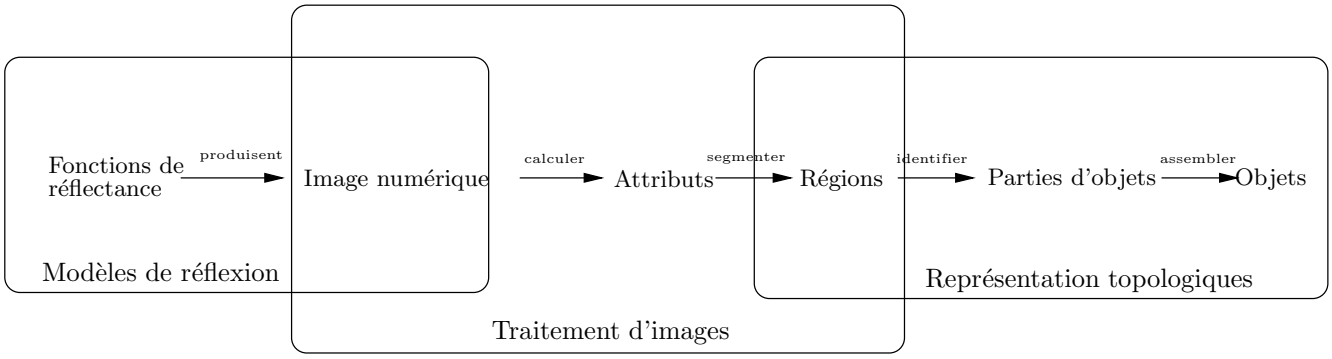


FIG. 5.1 – *Suite des processus et traitements intervenants en traitement d'images surimposée avec les thèmes abordés dans ce mémoire*

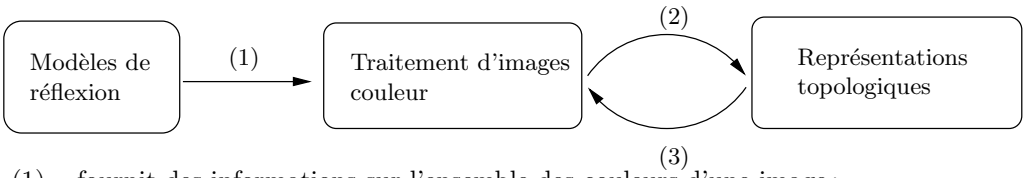
Toutefois, cet objectif de cohésion ne répond pas seulement à des contraintes de confort intellectuel mais également à une complémentarité réelle entre les différents thèmes abordés. La figure 5.1 superpose à la séquence d'opérations impliquées dans un processus d'analyse d'image (Fig. 1.1) les domaines couverts dans ce mémoire.

Les modèles de réflexion fournissent des outils permettant de décrire le processus de formation d'une image. Il est donc possible, dans un environnement contrôlé, de prédire les propriétés de l'ensemble des couleurs d'une image à partir des propriétés de la scène. Certains résultats statistiques [151, 42] suggèrent que des informations sur l'ensemble des couleurs d'une image peuvent être supposées *a priori* lorsque l'environnement d'acquisition est moins contrôlé et les scènes plus

variées. Les modèles de réflexion peuvent dans ce cas aider à définir le cadre dans lequel ces informations peuvent être supposées. Ces informations statistiques ont été utilisées deux fois dans le chapitre 3 (section 3.4).

Les algorithmes de traitement d'images couleurs utilisent les informations fournies par les modèles de réflexions pour calculer des attributs de pixels et définir à partir de ceux-ci des quantités invariantes pour chaque matériau de l'image (section 2.4). Les algorithmes de traitement d'images ne se limitent pas à l'utilisation des modèles de réflexion et une grande variété de critères et de traitements peuvent être utilisés pour traiter une image ou extraire des informations de celle-ci. Dans ce cadre, les méthodes de quantification (section 3.2) permettent de regrouper certaines couleurs de l'image afin d'obtenir une première partition de celle-ci en régions homogènes. La quantification peut donc être utilisée comme pré-traitement de méthodes de segmentation plus complexes ou pour découper des régions dans le cadre d'un algorithme de découpe récursive ou de découpe - fusion [33].

Les régions définies par les algorithmes de traitement d'image peuvent être codées par une représentation topologique telle que les pyramides combinatoires (section 4.6). Les algorithmes de traitement d'images fournissent donc aux modèles de représentation topologiques des critères permettant de définir des régions. Inversement, les représentations topologiques fournissent aux méthodes de traitement d'images des informations topologiques sur la partition telles que les relations d'adjacences, d'inclusion ou la liste des frontières d'une ou plusieurs régions. Ces informations sont utilisées par les algorithmes de traitement d'image pour modifier la partition. Les collaborations entre les différents domaines abordés dans ce mémoire sont représentées sur la figure 5.2. Notons toutefois, que dans le cadre d'une application les trois composantes représentées sur la figure 5.2 sont étroitement imbriquées. On obtient alors plus simplement, un algorithme de traitement d'image basé sur une représentation topologique et qui utilise des modèles de réflexion.



(1) fournit des informations sur l'ensemble des couleurs d'une image ;
(2) fournit des critères d'homogénéité définissant les régions ;
(3) fournit des informations topologiques permettant de définir les critères d'homogénéité.

FIG. 5.2 – Relations entre les thèmes abordés dans les différents chapitres

Nous avons évoqué tout au long de ce mémoire de multiples questions qui demeurent en suspend. Certaines de ces questions sont internes à une problématique tandis que d'autres concernent simultanément plusieurs domaines abordés dans ce mémoire. Ces problèmes peuvent se regrouper en différents thèmes de recherche que nous allons brièvement énumérer avant de les développer et indiquer la démarche que nous comptons utiliser pour les aborder.

Liens entre les modèles de réflexion et le traitement d'images

Les modèles de réflexion permettent de prédire la couleur d'un pixel en fonction des pa-

ramètres de la scène. Ces modèles permettent de prédire *a priori* des informations sur l'ensemble des couleurs d'une image en fonction de la scène. Toutefois, les applications actuelles de ces modèles concernent essentiellement :
- la segmentation en matériaux ;
- les méthodes de mesure des propriétés optiques de surface grâce à l'estimation des paramètres des modèles de réflexion ;
- les méthodes de reconstruction basées sur une inversion des modèles de réflexion. Il s'agit en effet, de retrouver la normale en chaque point de la scène en fonction de la couleur des matériaux.

Nous envisageons d'élargir les domaines d'applications de ces modèles en étudiant plus exhaustivement les relations entre les paramètres de la scène et la distribution de l'ensemble des couleurs d'une image. Cette étude devrait permettre de nombreuses applications en imagerie couleur et confirmer certains résultats statistiques précédemment obtenus [151] tout en précisant les domaines de validité de ceux-ci.

Méthodes de quantification mixtes

Comme nous l'avons vu dans la section 3.2.6, les méthodes de quantification mixtes ont été jusqu'ici assez peu explorées malgré leurs fort potentiel. Le développement de ces méthodes doit pouvoir s'effectuer en travaillant simultanément sur :

1. les méthodes de découpes initiales : il s'agira dans ce cas d'étudier des méthodes de découpe permettant de réduire la quantité de données de l'image avec une perte d'information minimale ;
2. les méthodes de fusion : il faudra étudier plus attentivement les heuristiques de fusion et évaluer le coût de celle-ci en regard des améliorations apportées à la qualité de l'image.

Notons que les deux problèmes sont liés. En effet, une réduction importante des données de l'image se justifie si elle permet l'utilisation d'un algorithme de fusion plus efficace. Dans ce domaine, l'erreur de partition finale reste un critère déterminant pour juger de la validité d'une heuristique.

Liens entre représentation topologiques et hiérarchiques

Les représentations topologiques basées sur les cartes combinatoires ou les cartes combinatoires généralisées ont été utilisées en synthèse et traitement d'images pour modéliser respectivement des scènes $3D$ [15, 133] et des partitions d'images $2D$ [26, 21, 2] ou $3D$ [23, 22, 14, 62]. Notre contribution à ce domaine de recherche a consisté à introduire la notion de hiérarchies dans le cadre des représentations topologiques. Cette recherche a abouti au modèle des pyramides combinatoires $2D$ que nous avons développé et implémenté. Il reste toutefois à élargir l'intégration des modèles hiérarchiques dans les représentations topologiques. Les deux directions sur lesquelles nous comptons travailler sont :

1. l'extension à des dimensions quelconques ;
2. l'utilisation d'une carte initiale quelconque pour la pyramide.

Le premier point doit nous permettre de traiter des données $3D$ utilisées en imagerie médicale. Des données de dimensions supérieures pourraient également être traitées si une application se présentait. Le second point doit nous permettre d'étendre le domaine d'application des représentations hiérarchiques à base topologiques aux domaines n'utilisant pas une grille discrète à la base de la pyramide. Les applications que nous envisageons sont la représentation hiérarchique de maillages modélisant des scènes $3D$, des terrains ou des couches géologiques.

Liens entre les représentations topologiques et les méthodes de traitement d'images

L'utilisation de graphes en traitement d'images n'est pas nouvelle et le comité technique numéro 15 d'IAPR (IAPR-TC15) a montré lors de ses différents groupes de travail que les applications des graphes en traitement d'images ouvraient de nombreuses perspectives à différents sous-domaines du traitement et de l'analyse d'images tels que la segmentation [86], la reconnaissance de formes [176, 169] ou l'analyse de scènes [7].

Nous comptons étudier l'apport des cartes combinatoires au problèmes d'appariement de formes codées par des graphes planaires. Ce problème est un des domaines importants abordé par l'IAPR-TC15 et a des fortes applications en reconnaissance de formes et en classification d'images [99, 145]. Nous pensons que l'orientation codée explicitement par les cartes combinatoires peut permettre une meilleure discrimination des graphes et donc des formes ou images représentées par ceux-ci.

Nous pensons également poursuivre notre effort dans le domaine des pyramides combinatoires. La poursuite de cet effort se traduira par des applications utilisant le traitement d'images couleurs dans un cadre hiérarchique ascendant. Nous comptons notamment définir des méthodes de segmentation prenant en compte simultanément les phénomènes physiques qui créent un vecteur couleur et les phénomènes physiologiques qui permettent une interprétation de celui-ci par l'être humain.

Axes de recherches à court terme

Nous comptons rapidement appliquer les pyramides combinatoires à la segmentation en matériaux. Nous comptons également poursuivre nos investigations sur les méthodes de quantification mixtes en utilisant une découpe de la projection du multi-ensemble initial sur son plan principal.

Axes de recherches à moyen terme

A moyen terme, nous comptons résoudre les derniers problèmes théoriques qui se posent dans le cadre des pyramides combinatoires. Ces problèmes concernent essentiellement le plongement de la carte combinatoire et la conception de méthodes permettant un accès efficace aux informations souvent sollicitées par les algorithmes de traitement d'images.

Nous comptons également étudier les liens existants entre les modèles de réflexion et la distribution de l'ensemble des couleurs d'une image.

Axe de recherches à long terme

Nos axes de recherche à long terme se placent dans le cadre d'un renforcement des liens entre les traitements haut et bas niveaux. Nous comptons notamment étudier les méthodes d'appariement de cartes combinatoires $2D$ dans le cadre de la reconnaissance de formes. Nous comptons également intégrer dans le cadre de traitements hiérarchiques deux aspects fondamentaux de la couleur : la prise en compte des phénomènes physiques qui créent le triplé couleur à la base de la pyramide (donc lors de la formation de petites régions) et prise en compte de la perception des images colorées à des niveaux plus élevés. Les relations topologiques entre les régions devraient être un outil fort utile dans cette dernière étape.

Nous comptons également étudier avec l'équipe modélisation du laboratoire IRCOM-SIC de Poitiers l'extension des pyramides combinatoires à des dimensions supérieures. Cet axe de

recherche a des application évidentes en imagerie médicale où l'on doit traiter de nombreuses images $3D$. Plus généralement cet axe de recherche devrait ouvrir de nombreuses opportunités d'applications dans tous les domaines où l'on doit traiter et structurer des masses importantes de données.

Nous comptons enfin étudier les problèmes impliqués par les opérations de «relinking» [147] dans le cadre des pyramides combinatoires. Cette opération consiste à modifier le père d'un sommet non survivant à un certain niveau de la pyramide. Cette opération à des répercussions sur les niveaux supérieurs à celui-ci. Ce type d'opération permet donc de remettre en cause une fusion entre deux régions effectuée à un certain niveau de la pyramide. Ce type de méthode peut s'avérer très utile dans le cadre de l'adaptation de pyramides à de nouvelles données.

5.1 Traitements d'images couleur et modèles de réflexion

Nos activités de recherche sur le traitement et l'affichage d'images couleur ont porté sur les méthodes de quantification et d'inversion de tables de couleurs. Comme nous l'avons mentionné, il est peu probable qu'une évolution majeure intervienne dans le domaine des méthodes de quantifications descendantes (section 3.2.3). Toutefois, les méthodes de quantification mixtes (section 3.2.6) semblent présenter un fort potentiel jusqu'ici peu exploité. Le principal avantage de ce type de méthode est de pouvoir être appliqué simultanément dans le cadre de l'affichage d'images et de la classification. Les méthodes de quantification spatiale sont quand à elles plus dévolues à l'affichage d'images et représentent certainement l'avenir de ce domaine de recherche.

Nos perspectives de recherche dans ce domaine sont doubles : nous pensons tout d'abord approfondir notre étude des méthodes mixtes. Nous comptons pour cela étudier prochainement l'intérêt d'une méthode de quantification mixte basée sur une découpe du plan principal du multi-ensemble d'une image. Utiliser une partition du plan principal plutôt qu'une partition du multi-ensemble initial revient à s'interdire des découpes suivant le troisième vecteur propre du multi-ensemble. Cette projection devrait permettre de s'affranchir de certaines des limitations de notre méthode basée sur une découpe du cube RGB. En effet, un des inconvénient d'une découpe uniforme $3D$ du cube RGB est qu'un nombre important de cubes issus du partitionnement uniforme sont vides ou ne contiennent que très peu de couleurs. Ce type de partitionnement utilise donc inutilement de la place mémoire. De plus, la relation entre le nombre de cubes utilisés pour le partitionnement initial et le nombre de cubes non vides effectivement utilisés par l'algorithme de fusion dépend de la distribution de couleurs de l'image et est donc difficile à paramétrer. La projection sur le plan principal doit nous permettre de travailler sur un espace beaucoup plus «plein» dans lequel le nombre de carrés utilisés pour le partitionnement initial sera proche du nombre utilisé effectivement par l'algorithme de fusion.

Toutefois, cette méthode ne sera intéressante que si la projection sur le plan principal n'occasionne pas une perte trop importante d'information. Ici encore, nous nous appuyons sur les travaux d'Otha [151] confirmés par nos propres expériences (section 3.3.6). En effet, selon Otha, la valeur de la troisième valeur propre d'un multi-ensemble est souvent négligeable par rapport aux deux premières. Le multi-ensemble varie donc peu *a priori* suivant le troisième vecteur propre. Il est donc probable que des cubes découpant le multi-ensemble initial perpendiculairement à ce vecteur seront réunis ultérieurement par l'algorithme de fusion. Projeter le multi-ensemble sur son plan principal revient donc à anticiper cette action.

Plus généralement, nous aimerions étudier l'apport éventuel des modèles de réflexions sur les méthodes de quantification. En effet, les modèles de réflexion peuvent fournir des informations *a priori* sur l'ensemble des couleurs d'une image qui peuvent être utilisées dans le cadre de la quantification. Notons qu'une approche similaire a été effectuée par Ramamoorthi [164] dans le cadre de l'analyse d'objets acquis à l'aide d'illuminants de différentes directions. Ramamoorthi, utilise un modèle Lambertien pour mener une étude théorique permettant de définir un ensemble de directions d'illumination fournissant des informations maximales et non corrélées.

Nous comptons nous investir d'avantage dans la segmentation en matériaux en étudiant plus attentivement les liens existants entre les méthodes d'estimation de paramètres pour la reconstruction et les méthodes de segmentation en matériaux. En effet, les constantes des modèles de réflexion sont déterminées par les propriétés optiques des matériaux ; les évaluer permet donc d'obtenir des invariants pour chacune des régions codant un matériau de l'image. Par exemple, si nous comparons les méthodes de segmentation en matériaux de Klinker [94](section 2.4.1) et d'estimation de paramètres de Kay [118] (section 2.5.3) nous pouvons remarquer que :

1. Fixer le modèle de réflexion d'un pixel détermine les variations de sa couleur en fonction de la normale à la surface. La détermination par Kay du meilleur modèle permettant de décrire la réflexion d'un pixel a donc forcément un lien avec la dimension du voisinage de chaque pixel calculée par Klinker.
2. L'estimation de la valeur des constantes effectuée par Kay pour chaque pixel est extrêmement proche de l'affectation par Klinker de chaque pixel à un matériau.

5.2 Représentation hiérarchiques et traitement d'images

L'introduction des représentation hiérarchiques en traitement d'images se situe dans le milieu des années 70 [183]. Le lien entre représentations hiérarchiques et traitement d'images est donc ancien et ne s'est jamais rompu [45, 115, 146, 114, 141, 53, 124, 40].

Nous comptons étudier prochainement l'apport des représentation hiérarchiques à la segmentation en matériaux. Nous allons pour cela combiner la méthodes de segmentation en matériaux de Klinker [94] (section 2.4.1) basée sur une approche ascendante avec la méthode de décimation définie par Jolion [114](section 4.3).

La méthode de Klinker repose sur une estimation de la dimension de l'ensemble des couleurs au voisinage de chaque pixel de l'image. L'espace couleur étant de dimension 3, la dimension associée à chaque pixel est donc égale à 1, 2 ou 3. Klinker effectue dans un premier temps des fusions de pixels ou de régions de dimension 1 si le résultat conserve cette dimension. Dans un second temps, Klinker fusionne des régions de dimension 1 adjacentes si l'ensemble des couleurs de la région fusionnée vérifie une condition appelée l'hypothèse des 50% supérieurs (section 2.4.1). De son côté, Jolion a défini une méthode de segmentation ascendante basée sur les pyramides irrégulières. Dans le cadre du processus de décimation défini par Jolion, les sommets survivants ont une variance minimale. Cette variance est calculée sur un voisinage de chaque sommet. Les sommets non survivants sont alors attachés au sommet survivant adjacent dont le niveau de gris est le plus proche.

Une adaptation de la méthode définie par Jolion à la segmentation en matériaux consistera à attacher à chaque sommet initial une valeur représentant l'écart entre la distribution des couleurs du voisinage d'un sommet et une droite. Cet écart peut être défini par le rapport :

$$\frac{v_2 + v_3}{v_1 + v_2 + v_3}$$

où (v_1, v_2, v_3) représentent les 3 valeurs propres de la matrice de covariance calculée à partir de l'ensemble des couleurs du voisinage d'un sommet.

Chaque sommet non survivant est alors attaché au sommet survivant dont il modifie le moins la distribution des couleurs. Ce processus de décimation peut être combiné avec la fonction λ définie par Montanvert [146](section 4.3) de façon à exclure temporairement du processus de fusion les pixels de dimension 2 ou 3. On peut également utiliser un processus de décimation asynchrone (section 4.4) de façon à privilégier la fusion des sommets de dimension 1.

La construction de la pyramide va donc agréger les pixels de telle façon que l'ensemble des couleurs de chaque région forme une droite dans l'espace couleur choisi (cet espace doit tout de même être déduit des réponses des capteurs par une transformation linéaire). Un second critère de fusion peut alors être utilisé dans la pyramide de façon à fusionner toutes les régions adjacentes de dimension 1 dont la fusion produit une région de dimension 2 vérifiant l'hypothèse des 50% supérieurs. Un troisième critère de fusion peut être utilisé pour fusionner les sommets de dimension 2 ou 3 initialement rejetés du processus de fusion. Chaque pixel de dimension 2 ou 3 est alors fusionné à la régions de dimension 2 dont il modifie le moins la distribution.

Nous comptons également étudier l'apport des pyramides combinatoires aux problèmes de reconnaissance de formes basées sur d'appariement de graphes. Nous pensons en effet que l'orientation peut être extrêmement intéressante dans ce cadre. Étudions, à titre d'exemple, le graphe représenté sur la figure 5.3(a). Ce graphe est isomorphe au graphe représenté sur la figure 5.3(b) si nous apparions les arêtes (p_1, p_2) et (q_1, q_2). Dans le cadre d'un contrôle industriel où la figure 5.3(a) coderait le modèle d'une pièce et la figure 5.3(b), une pièce à contrôler les deux pièces sont rigoureusement identiques pour un algorithme d'appariement de graphe classique. Dans le cadre des cartes combinatoires, le seul isomorphisme possible est celui qui apparie (p_1, q_2) et (q_1, p_2). Des critères géométriques peuvent dans ce cas permettre de rejeter cet isomorphisme et de conclure à un défaut de la pièce.

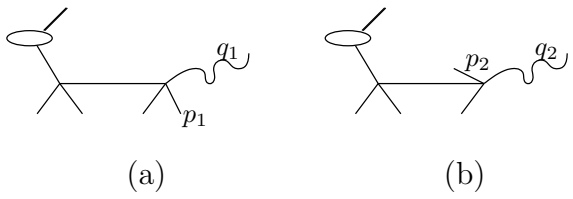


FIG. 5.3 – *Deux graphes similaires codés par deux cartes distinctes*

Les travaux précurseurs de R. Cori [57] sur les automorphismes de cartes combinatoires semblent montrer qu'un isomorphisme de cartes combinatoires obéit à des contraintes beaucoup plus fortes qu'un isomorphisme de graphe. De fait, un isomorphisme de cartes combinatoires est déterminé par l'association de deux brins dans chacune des cartes. Cette propriété extrêmement forte permet de calculer un isomorphisme entre deux cartes dans un temps au pire égal à $\mathcal{O}(n^2)$ où n est le nombre de brins des deux cartes. Le travail que nous avons initié avec Katerina Schindler (PRIP, Université Technique de Vienne) consiste tout d'abord à améliorer cette complexité puis à étendre ces résultats aux morphismes de cartes.

Notons que cette activité de recherche n'exclut pas les représentations pyramidales. En effet, une solution à l'appariement de graphes complexes consiste à construire des pyramides réduisant

les deux graphes puis à apparier les versions réduites par une approche descendante. On procède donc des graphes les plus simples aux plus complexes.

5.3 Représentations topologiques et hiérarchiques

Les travaux que nous avons effectués dans le cadre des pyramides combinatoires sont indépendants du type de carte utilisée pour coder la partition et de la grille discrète associée à la carte de base. On peut ainsi, sans changer le formalisme, utiliser indifféremment une carte des régions ou une carte des frontières. Plus généralement, le formalisme actuel supporte l'utilisation d'une carte quelconque comme carte initiale. On peut donc par exemple, définir une pyramide à partir d'une vectorisation du squelette d'une forme. La pyramide définit dans ce cas une pyramides de formes.

Toutefois, une telle généralité ne permet pas de préciser efficacement le plongement. En effet, le plongement d'une carte décrit l'objet géométrique associé à un sommet ou une face d'une carte de la pyramide. Préciser la géométrie de l'objet associé par exemple à un sommet implique de savoir :

1. ce que représente ce sommet, donc si nous utilisons une carte des frontières ou des régions,
2. quelle est la géométrie associée à chaque brin de la carte de base. Autrement dit, quel type de grille discrète a été réduit dans la pyramide.

Nous comptons entamer cette étude en nous penchant sur le cas d'une carte des régions associée à une grille 4 connexe. Il est en effet fort probable que la plupart des résultats pourront facilement se généraliser à d'autres types de grilles ou à des cartes des frontières. Il est donc inutile d'étudier le problème dans toute sa complexité dès le départ. Nous comptons montrer la validité des algorithmes qui :

- parcourent la frontière entre deux régions associées à un brin survivant ;
- permettent de déterminer une carte réduite de niveau i en ne parcourant que les frontières des régions de niveau i.

Nous comptons également agréger les résultats précédents en déterminant la frontière d'une région et l'intérieur de celle-ci. Il faudra bien évidement définir des algorithmes permettant un parcours rapide de l'intérieur d'un région.

A plus long terme, nous pensons orienter nos recherches suivant deux directions :

Tout d'abord, nous comptons travailler en collaboration avec l'équipe modélisation du laboratoire IRCOM-SIC de Poitiers et plus particulièrement avec Pascal Lienhardt et Guillaume Damiand de façon à étendre les résultats obtenus à des dimensions quelconques. Les premiers résultats obtenus semblent prometteurs (section 4.6.11).

Nous comptons également étudier les problèmes de relinking dans les pyramides combinatoires. Le relinking dans une pyramide irrégulière [147] consiste à changer le père d'un sommet de la pyramide et à mettre à jour les niveaux de la pyramide supérieurs à celui où est défini ce sommet. Le principal intérêt de cette méthode est de permettre de remettre en cause dans le cadre d'une approche descendante des choix qui ont été faits lors de la construction de la pyramide (donc dans le cadre d'une analyse ascendante). Ce type d'opération permet également d'adapter des pyramides à de nouvelles données en changeant les valeurs à la base de celle-ci et en modifiant les relations père - fils en fonction de ces nouvelles valeurs.

5.4 Conclusions

Notre objectif à long terme est donc double : d'une part travailler sur les méthodes bas niveau afin de mieux comprendre et exploiter l'information couleur. D'autre part, définir un modèle permettant de coder des hiérarchies de partitions de dimensions quelconques et d'étudier leurs applications pour la modélisation et la vision.

La poursuite de ces deux objectifs complémentaires permettra de :

1. créer des ponts souvent inexistants entre les communautés scientifiques spécialisées dans la modélisation et le traitement d'image ;
2. mettre en relation des approches en traitement d'images qui se distinguent souvent uniquement par la dimension de l'espace ou la structure de données utilisée pour la modélisation des partitions. Ce dernier point fait parti des problématiques abordées par l'action spécifique AS STIC CNRS : imagerie, vision et analyse de scènes.

Créer une dynamique entre les méthodes de modélisation et de traitement d'images ouvrira des perspectives insoupçonnées aux deux communautés.

Chapitre 6

Annexe

6.1 Algorithme de Bister

Cette section décrit l'algorithme de Bister [16] utilisé en section 4.2 (voir également la figure 4.5). Nous reprenons ici les notations de Bister.

6.1.1 Notations

P	:	pixel P
P'	:	fils du pixel P
P^o	:	père de P
P'^o	:	père du pixel P' c'est à dire père du fils du pixel P
$P^{o\prime}$	:	fils de P^o : fils du père du pixel P
$v(P)$	:	niveau de gris du pixel P
$a(P)$	:	facteur de surface du pixel P
$w(P, P^o)$	:	poids du lien entre P et son père P^o

Dans le cadre de pyramides recouvrantes (section 4.2) un pixel a plusieurs fils et chaque fils à plusieurs pères *potentiels*. Le père d'un pixel qui a le plus fort lien avec celui-ci est appelé sont *père légitime*. L'algorithme général de segmentation est le suivant :

6.1.2 Construction de la pyramide

1. initialisation : Valeur(Père) := valeur moyenne des fils ;
2. itérations :
   (a) lien (père, fils),
   - similarité en «niveaux de gris» entre le père et le fils,
   - 1/Distance(Père,Fils),
   - Similarité en «niveaux de gris» entre les fils.

   Notons que d'autres attributs que les niveaux de gris peuvent être utilisés pour calculer les liens.

   (b) Valeur(père) := moyenne pondérée des valeurs des fils,

3. segmentation : tous les fils sans père deviennent des racines la valeur d'un père est donnée à tous ses enfants tels que Lien(père,fils) > seuil. L'algorithme 5 présente en pseudo-code l'algorithme général de construction d'une pyramide.

6.1.3 Attributs utilisés dans les pyramides

L'initialisation des attributs des pixels de la pyramide est effectuée des lignes 1 à 16 de l'algorithme 5. On distingue différent types d'initialisation :

- pyramides initialisée : basée sur un simple filtre passe bas suivi d'un sous échantillonage [69] (les lignes 9 à 16 du pseudo code ne sont pas utilisées) ;
- pyramides normalisées : basée sur une moyenne pondérée calculée itérativement. Les poids sont inversement proportionnels à la différence des niveaux de gris entre les pères et leurs fils. La somme des poids des liens entre un pixel et ses 4 pères est normalisée à 1 (lien normalisé) ;
- pyramides forcées : basée sur une moyenne itérative de tous les fils «légitimes» du pixel courant (c'est à dire de tous les fils qui ont un lien maximum avec le pixel courant, les liens étant inversement proportionnels à la différence des niveaux de gris entre un père et son fils).

6.1.4 Segmentations pyramidales

La segmentation à partir des attributs calculés dans la pyramide est effectuée des lignes 18 à 38 de l'algorithme 5. Selon la version de la formule L (section 6.1.5) qui est utilisée, nous pouvons avoir une segmentation non normalisée, normalisée ou forcée. La combinaison de différentes méthodes de calcul d'attributs et de segmentations produisent différents types de pyramides. Bister distingue 5 d'entre elles représentées par les symboles suivants :

FNN	:	Pyramide forcée, segmentation non normalisée
NNN	:	Pyramide normalisée, segmentation non normalisée
INN	:	Pyramide initialisée, segmentation non normalisée
IN	:	Pyramide initialisée, segmentation normalisée
IF	:	Pyramide initialisée, segmentation forcée

6.1.5 Formules utilisées

formule I :

- niveau de gris :
  un filtre passe bas (voir par exemple Meer [69], p 157, Table IV) ;
- facteur de surface :
$$\frac{\text{nombre de pixels dans l'image originale}}{\text{nombre de pixels au niveau courant}};$$

formule G :

- niveau de gris :
$$v(P) = \frac{\sum_{P'} v(P')a(P')w(P',P)}{\sum_{P'} a(P')w(P',P)};$$

```
1 Du niveau le plus bas au niveau le plus haut de la pyramide
2   pour tous les pixels du niveau courant
3     initialiser les niveaux de gris et les facteurs de
4     surface (formules I)
5   pour tous les pixels du niveau courant
6     initialiser les liens entre chaque pixel et ses fils
7     (formule L)
8
9   faire
10    pour tous les pixels du niveau courant
11      re - calculer les niveaux de gris et les
12      facteurs de surface (formule G)
13    pour tous les pixels du niveau courant
14      re - calculer les liens entre chaque pixel et ses fils
15    (formule L)
16  tant que ( le père légitime de tous les pixels ne reste pas invariant)
17
18  Du niveau le plus bas au niveau le plus haut de la pyramide
19    pour tous les pixels du niveau courant
20      recalculer les liens (formule L)
21
22  Du niveau le plus haut au niveau le plus bas de la pyramide
23    pour tous les pixels du niveau courant
24      sélectionner les racines
25
26  Du niveau le plus haut au niveau le plus bas de la pyramide
27    pour tous les pixels du niveau courant
28      si non racine
29        lier à la même région que son père légitime
30      si une étape de relaxation est sélectionnée pour ce niveau
31        pour tous les pixels du niveau courant
32          pour tous les frères du pixel courant
33            si le frère appartient à une région différente
34              si le niveau de gris du pixel est plus proche du niveau de gris
35              du père légitime de son frère que de celui de son propre père légitime
36                lier à la même région que celle de son frère
37                Pour tous les frère
38                appliquer la procédure de relaxation récursivement.
```

Algorithme 5: *Algorithme de Bister*

– facteur de surface :

$$a(P) = \frac{\sum_{P'} a(P')w(P', P)}{\sum_{P^{o\prime}} w(P', P'^o)};$$

formule L :

– liens non normalisés :

$$w(P, P^o) = (v(P) - v(P^o))^{-2};$$

– liens normalisés :

$$w(P, P^o) = \frac{(v(P) - v(P^o))^{-2}}{\sum_{P^{o\prime}} \left(v(P^{o\prime}) - v(P^o)\right)^{-2}};$$

– liens forcés :

$$w(P, P_1^o) = \begin{cases} 1 & si \quad (v(P) - v(P_1^o))^2 > (v(P) - v(P_2^o))^2 \, \forall P_2^o \neq P_1^o \\ 0 & sinon \end{cases};$$

Bibliographie

[1] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6) :641–647, 1994.

[2] E. Ahronovitz, J. Aubert, and C. Fiorio. The star-topology : a topology for image analysis. In *5th DGCI Proceedings*, pages 107–116, 1995.

[3] M. Anderberg. *Cluster analysis for applications*. Academic Press, New York, 1973.

[4] R. Balasubramaian, J. Allebach, and C. A. Bouman. Color-Image Quantization with Use of a Fast Binary Splitting Technique. *Journal of the Optical Society of America*, 11(11) :2777–2786, November 1994.

[5] R. Balasubramanian and J. Allebach. A new approach to palette selection for color images. *Journal of imaging technology*, 17(6) :284–290, december 1991.

[6] R. Balasubramanian and J. Allebach. A New Approach to Palette Selection for Color Images. *Human Vision, Visual Processing, and Digital Display III (1991)*, SPIE 1453 :58–69, 1991.

[7] C. Bauckhage, E. Braun, S. Wachsmuth, and G. Sagerer. 3d assembly recognition by matching functional subparts. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 95–104. IAPR-TC15, May 2001.

[8] P. Beckmann and A. Spizzichino. *The scattering of electromagnetic waves from rough surfaces*. Artech House Inc, 2nd edition, 1987.

[9] D. A. Belsley. *Conditioning Diagnostic*. Wiley, New York, 1991.

[10] S. Ben Yacoub and J. M. Jolion. Hierarchical line extraction. *IEEE Vision, Image and Signal Processing*, 142(1) :7–14, 1995.

[11] J. L. Bentley, J. H. Friedman, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Software*, 3 :209–226, September 1977.

[12] C. Berge. *Graphes*. Gauthier-villars, bordas edition, 1983.

[13] E. Bertin. *Diagrammes de Voronoi 2D et 3D : application en analyse d'images*. PhD thesis, Université J. Fourier, Grenoble, 1994.

[14] Y. Bertrand, G. Damiand, and C. Fiorio. Topological map : Minimal encoding of 3d segmented images. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *Workshop on Graph-based Representations in Pattern Recognition*, pages 64–73, Ischia(Italy), May 2001. IAPR-TC15, CUEN.

[15] Y. Bertrand and J. Dufourd. Algebraic specification of a 3D-modeler based on hypermaps. *CVGIP : Graphical Models and Image Processing*, 56(1) :29–60, Jan. 1994.

[16] M. Bister, J. Cornelis, and A. Rosenfeld. A critical view of pyramid segmentation algorithms. *Pattern Recognit Letter.*, 11(9) :605–617, September 1990.

[17] C. Bouman and M. Orchard. Color Image Display with a Limited Palette Size. *Visual Communications and Image Processing IV (1989)*, SPIE 1199 :522–533, 1989.

[18] C. Bouman and M. Orchard. Color Quantization of Images. *IEEE Transactions on Signal Processing*, 39(12) :2677–2690, December 1991.

[19] R. M. Boynton. *Human Color Vision*. Optical Society of America, 1992.

[20] J. P. Braquelaire and L. Brun. Comparison and optimization of methods of color image quantization. *IEEE Transactions on Image Processing*, 6(7) :1048–1052, july 1997.

[21] J. P. Braquelaire and L. Brun. Image segmentation with topological maps and inter-pixel representation. *Journal of Visual Communication and Image representation*, 9(1) :62–79, 1998.

[22] J. P. Braquelaire, P. Desbarats, and J. P. Domenger. 3d split and merge with 3-maps. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *Workshop on Graph-based Representations in Pattern Recognition*, pages 32–43, Ischia(Italy), May 2001. IAPR-TC15, CUEN.

[23] J. P. Braquelaire, P. Desbarats, J.-P. Domenger, and C. Wüthrich. A topological structuring for aggregates of 3d discrete objects. In W. Kropatsch and J.-M. Jolion, editors, 2^{nd} *IAPR-TC-15 Workshop on Graph-based Representations*, volume 126, pages 145–154, Haindorf, Austria, May 1999. Österreichische Computer Gesellschaft.

[24] J. P. Braquelaire and J. Domenger. Geometrical, topological, and hierarchical structuring of overlapping 2-d discrete objects. *Computers & Graphics*, 21(5) :587–597, September 1997.

[25] J. P. Braquelaire and J. P. Domenger. Representation of region segmented images with discrete maps. Technical report, RR-112797 LaBRI, June 1996.

[26] J. P. Braquelaire and P. Guitton. $2\frac{1}{2}$ scene update by insertion of contour. *Computer and Graphics*, 15(1) :41–48, 1991.

[27] M. H. Brill. Image segmentation by object color : a unifying framework and connection to color constancy. *Opt. Soc. Am. A.*, 7(10) :2041–2047, October 1990.

[28] M. J. Brooks and B. P. K. Horn. Shape and source from shading. In *Proc. Int. Joint Conf. Artificial Intell.*, pages 932–936, Los Angeles, August 1985.

[29] G. Bruhat. *Cours de physique générale : optique*. Masson, 1959.

[30] L. Brun. *Segmentation d'images couleur à base Topologique*. PhD thesis, Université Bordeaux I, 351 cours de la Libération 33405 Talence, December 1996.

[31] L. Brun. Traitement d'images couleur. Cours premier cycle,
URL : www.univ-reims.fr/leri/membre/luc/enseignement.html, 2002.

[32] L. Brun and J. M. Bazin. Amélioration des performances d'un système de segmentation par l'utilisation d'un système expert. In *Advances in Intelligent Computing- IPMU'98*. IPMU, 1998.

[33] L. Brun and J. P. Domenger. A new split and merge algorithm with topological maps and inter-pixel boundaries. In *The fifth International Conference in Central Europe on Computer Graphics and Visualization*, february 1997.

[34] L. Brun, J. P. Domenger, and J. P. Braquelaire. Discrete maps : a framework for region segmentation algorithms. In *Workshop on Graph based representations*, Lyon, April 1997. published in Advances in Computing (Springer).

[35] L. Brun and W. Kropatsch. Dual contractions of combinatorial maps. Technical Report 54, Institute of Computer Aided Design, Vienna University of Technology, lstr. 3/1832,A-1040 Vienna AUSTRIA, January 1999.

[36] L. Brun and W. Kropatsch. Pyramids with combinatorial maps. Technical Report PRIP-TR-057, PRIP, TU Wien, 1999.

[37] L. Brun and W. Kropatsch. The construction of pyramids with combinatorial maps. Technical Report 63, Institute of Computer Aided Design, Vienna University of Technology, lstr. 3/1832,A-1040 Vienna AUSTRIA, June 2000.

[38] L. Brun and W. Kropatsch. Defining regions within the combinatorial pyramid framework. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the Computer Vision Winter Workshop*, pages 198–207, Bad Ausse Austria, February 2002.

[39] L. Brun and W. Kropatsch. Labeled pyramids with combinatorial maps. Technical Report PRIP-TR-yy, PRIP, TU Wien, 2002.

[40] L. Brun and W. Kropatsch. Receptive fields within the combinatorial pyramid framework. *Graphical Modeling*, 2002. In Press.

[41] L. Brun and M. Mokhtari. Two high speed color quantization algorithms. In Cépaduès, editor, *Proceedings of CGIP'2000*, pages 116–121, Saint Etienne, October 2000.

[42] L. Brun and C. Secroun. A fast algorithm for inverse color map computation. *Computer Graphics Forum*, 17(4) :263–271, December 1998.

[43] L. Brun and A. Trémeau. *Digital Color Imaging Handbook*, chapter 9 : Color quantization, pages 589–637. Electrical and Applied Signal Processing. CRC Press, 2002. In Press.

[44] R. P. Bryant and D. Singerman. Foundations of the theory of maps on surfaces with boundary. *Quart. J. Math. Oxford*, 2(36) :17–41, 1985.

[45] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *IEEE Transaction on Communications*, 31(4) :532–540, April 1983.

[46] P. Burt, T.-H. Hong, and A. Rosenfeld. Segmentation and estimation of image region properties through cooperative hierarchial computation. *IEEE Transactions on Sustems, Man and Cybernetics*, 11(12) :802–809, December 1981.

[47] C. A. Chaudhuri, W. T. Chen, and J. Wang. A modified metric to compute distance. *Pattern Recognition*, 7(25) :667–677, 1992.

[48] C.-H. Chen. *Statistical Pattern recognition*. Hayden, 1973.

[49] R.-J. Chen and B.-C. Chieu. Three-dimensional morphological pyramid and its application to color image sequence coding. *Signal Processing*, 44(2) :163–180, 1995.

[50] E. Chiarello, J. M. Jolion, and C. Amoros. Regions growing with the stochastic pyramid : application in landscape ecology. *Pattern Recognition*, 29(1) :61–75, 1996.

[51] P. A. Chou, L. T., and G. R. M. Optimal pruning with applications to tree-structured source coding and modeling. *IEEE Trans. Inf. Theory*, 2 :299–315, 1989.

[52] CIE. Colorimetry. Technical Report 15.2, CIE, Centr. Bureau CIE, Vienna, Austria, 1986.

[53] L. Cinque, S. Levialdi, and A. Rosenfeld. Fast pyramidal algorithms for image thresholding. *Pattern Recognition*, 28(6) :901–906, June 1995.

[54] J.-P. Cocquerez and S. Philipp. *Analyse d'images : Filtrage et segmentation*. Masson, 1995.

[55] J.-P. Cocquerez and S. Philipp. *Analyse d'images : Filtrage et segmentation*, chapter Cahapitre XI Approche région et méthodes markoviennes, pages 280–304. Masson, 1995.

[56] R. Cori. *Un code pour les graphes planaires et ses applications*. PhD thesis, Université Paris VII, 1975.

[57] R. Cori. Computation of the automorphism group of a topological graph embedding. Technical Report I-8612, UER de mathématique et informatique, CNRS équipe du laboratoire associé 226, Université Bordeaux I, 1985.

[58] G. Cui, M. Luo, B. Rigg, and W. Li. Colour difference evaluation using crt colours. part i : Data gathering and testing colour difference formulae. *Color Research and Application*, 26(5) :394–402, August 2001.

[59] G. Cui, M. Luo, B. Rigg, and W. Li. Colour difference evaluation using crt colours. part ii : Parametric effects. *Color Research and Application*, 26(5) :403–412, August 2001.

[60] G. Damiand. *Définition et étude d'un modèle topologique minimal de représentation d'images 2d et 3d*. PhD thesis, Université des Sciences et Techniques du Languedoc, Décembre 2001.

[61] G. Damiand and P. Lienhardt. Removal and contraction for n-dimensional generalized maps. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the Computer Vision Winter Workshop*, pages 208–221, Bad Ausse Austria, February 2002.

[62] G. Damiand and P. Resch. Topological map based algorithms for 3d image segmentation. In A. Braquelaire, J.-O. Lachaud, and A. Vialard, editors, *Discrete Geometry for Computer Imagery*, volume 2301 of *LNCS*, pages 220–231, Bordeaux, April 2002. Springer-Verlag. ISBN 3-540-43380-5, ISSN 0302-9743.

[63] P.-E. Danielson. Euclidean distance mapping. *Computer Vision, Graphics, and Image Processing*, 14 :227–248, 1980.

[64] P. Desbarats. *Structuration d'images segmentées 3D discrètes*. PhD thesis, LaBRI, Université Bordeaux I, 2001.

[65] S. S. Dixit. Quantization of Color Images for Display/Printed on Limited Color Output Devices. *Computers and Graphics*, 15(4) :561–568, 1991.

[66] J. P. Domenger. *Conception et implémentation du noyau graphique d'un environnement 2D$\frac{1}{2}$ d'édition d'images discrètes*. PhD thesis, Labri Université Bordeaux I, 351 cours de la libération 33405 Talence, avril 1992.

[67] D. Dubois and H. Prade. *Fuzzy sets ans systems, theory and applications*. Academic Press, 1980.

[68] R. Duda and P. Hart. *Pattern classification and scene analysis*. Wiley New York, 1973.

[69] P. M. E., E. Baugher, and A. Rosenfeld. Frequency domain analysis and synthesis of image pyramid generating kernels. *IEEETPAMI*, 9 :512–522, 1987.

[70] J. Edmonds. A combinatorial representation for polyhedral surfaces. *Notices American Society*, 7, 1960.

[71] W. H. Equitz. A new vector quantization clustering algorithm. *IEEE transactions on acoustics, speech, and signal processing*, 37(10) :1568–1575, october 1989.

[72] H. S. Fairman and M. H. H. Brill. How the cie 1931 color-matching functions were derived from wright-guild data. *Color Res. Appl.*, 22(1) :11–23, Feb. 1997.

[73] O. Faugeras. *Three dimensional computer vision - A geometric viewpoint.* MIT Press, 1993.

[74] O. D. Faugeras. Digital color image processing within the framework of a human visual model. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, ASSP-27(4), Aug. 1979.

[75] C. Fernandez-Maloigne, N. Richard, and E. Laize. Progressive color coding with run length. In Cépaduès, editor, *Proceedings of CGIP'2000*, pages 122–124, Saint Etienne, October 2000.

[76] C. Fiorio. *Approche interpixel en analyse d'images : une topologie et des algorithmes de segmentation.* Thèse de doctorat, Université Montpellier II, 24 novembre 1995.

[77] C. Fiorio and J. Gustedt. Two linear time union-find strategies for image processing. Technical Report 375/1994, Technische Universitat Berlin, 1994.

[78] T. J. Flohr, B. W. Kolpatzik, R. Balasubramanian, D. A. Carrara, C. A. Bouman, and J. P. Allebach. Model Based Color Image Quantization. *Human Vision, Visual Processing, and Digital Display IV (1993)*, SPIE 1913 :270–281, 1993.

[79] R. Floyd and L. Steinberg. An adaptative algorithm for spatial greyscale. *Proc. SID*, 17(2) :75–77, 1976.

[80] J. Françon. Topologie de Khalimsky et Kovalevsky et algorithmes graphiques. In *First Colloquium on Discrete Geometry in Computer Imagery*, Strasbourg, September 1991.

[81] C. Froidevaux, M.-C. Gaudel, and M. Soria. *Types de données et algorithmes.* Mc Graw-Hill, 1990.

[82] B. V. Funt and G. D. Finlayson. Color constant color indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5) :522–529, May 1995.

[83] A. R. Gallat. *Nonlinear Satistical Models.* Wiley, New York, 1987.

[84] A. J. Gareth and D. Singerman. Theory of maps on orientable surfaces. *Proceedings of the London Mathematical Society*, 3(37) :273–307, 1978.

[85] M. Gervautz and W. Purgathofer. A Simple Method for Color Quantization : Octree Quantization. In N. Magnenat-Thalmann and D. Thalmann, editors, *New Trends in Computer Graphics*, pages 219–231. Springer-Verlag, New York, NY, 1988.

[86] L. Guigues, H. l. Men, and J.-P. Cocquerez. Graphs, cocoons and image segmentation. In J.-M. Jolion, W. G. Kropatsch, and M. Vento, editors, 3^{rd} *Workshop on Graph Based Representations in Pattern Recognition (GbR'2001)*, pages 22–31, Ischia, Italy, May 2001. IAPR-TC15, CUEN. ISBN 88 7146 579-2.

[87] F. Harary. *Graph Theory*, chapter 11, page 118. Addison-Wesley, 1972.

[88] R. Hartley and A. Zisserman. *Multiple view Geomerty in Computer Vision.* Cambridge Press, Cambridge, UK, 2000.

[89] Y. Haxhimusa, R. Glantz, G. Langs, and W. Kroparsch. Reduction factors of pyramids on undirected and directed graphs. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the Computer Vision Winter Workshop*, pages 29–38, Bad Ausse Austria, February 2002.

[90] Y. Haxhimusa, R. Glantz, M. Saib, G. Langs, and W. G. Kropatsch. Reduction Factors of Graph Pyramids. Submitted. In *Proceedings of the ICPR2002.* IEEE Computer Society, 2002.

[91] G. Healey. Segmenting images using normalized color. *IEEE Transactions on Systems, Man and CYBERNETICS*, 22(1) :64–73, January 1995.

[92] G. Healey and D. Slater. Computing illumination-invariant descriptors of spatially filtered color image regions. *IEEE Transactions on Image Processing*, 6(7) :1002–1013, July 1997.

[93] G. E. Healey. Using color for geometry insensitive segmentation. *J. Opt. Soc. Am. A.*, 6 :920–937, June 1989.

[94] G. E. Healey, S. A. Shafer, and L. B. Wolff, editors. *Color*, chapter A Physical Approach to Color Image Understanding, pages 134–165. Jones and Bartlett, 1992.

[95] G. E. Healey, S. A. Shafer, and L. B. Wolff, editors. *Color*, chapter Segmenting Images using Normalized Color, pages 166–198. Jones and Bartlett, 1992.

[96] G. E. Healey, S. A. Shafer, and L. B. Wolff, editors. *Color*, chapter Color Image Segmentation. Jones and BArtlett, 1992.

[97] P. Heckbert. Color image quantization for frame buffer display. In *Proceedings of SIGGRAPH'82*, pages 297–307, 1982.

[98] M. Herbin, N. Bonnet, and P. Vautrot. A clustering method based on the estimation of the probability density function and on the skeleton by influence zones. *Pattern Recognition Letters*, 17 :1141–1150, 1996.

[99] Y. Hodé and A. Deruyver. Image interpretation with a semantic graph : labeing oversegmented images and detection of unexpected objects. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 137–148, Ischia Italy, May 2001. IAPR-TC15, CUEN.

[100] M. H. Hodgson. Reducing the computation requirements of the minimum distance classifier. *Remote sencing of Environnement*, 25 :117–128, 1988.

[101] B. K. P. Horn. *Shape from Shading : A method for obtaining the shape of smooth opaque object from one view.* PhD thesis, MIT, Dept. of Electrical Eng., Cambridge, MA, 1970.

[102] B. P. K. Horn. *Robot vision.* Cambridge MA, MIT Press, 1986.

[103] B. P. K. Horn. Height and gradient from shading. *Int. Journal Comp. Vision*, 5(1) :584–595, August 1990.

[104] B. P. K. Horn and M. J. Brooks, editors. *Shape from Shading*, chapter Improved methods of estimating shape from shading using the light source coordinate system, pages 323–569. Cambridge MA, MIT Press, 1989.

[105] S. L. Horowitz and T. Pavlidis. Picture segmentation by a directed split-and-merge procedure. Technical report, Departement of Electrical Engineering, Princeton University , N. J. 08540, 1975.

[106] S. L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *Journal of The Association for Computing Machinery*, 23(2) :368–388, April 1976.

[107] G. Houle and E. Dubois. Quantization of colour images for display on graphics terminals. In *Proc. IEEE Global Telecomun. Conf.*, GLOBE-COM86, pages 1138–1142, december 1986.

[108] R. S. Hunter. *The measurement of Appearance.* Wiley, New York, 1975.

[109] L. Hussenet. *De l'analyse d'images à la reconstruction 3D : Application au contrôle d'objets métalliques.* PhD thesis, Université de Reims, 2002.

[110] L. Hussenet and L. Brun. Physic based classification for color images. In *Physics in Signal & Image Processing*, pages 173–176, Marseille, January 2001. Société de l'Électricité, de l'Électronique et des Technologies de l'Information et de la Communication.

[111] K. Ikeuchi and K. Sato. Determining reflectance properties of an object using range and brightness images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11) :1139–1153, November 1991.

[112] J. M. Jolion. A hierarchical framework for robust extraction and delineation of geometric features. *Pattern Recognition*, 26 :1295–1304, 1993.

[113] J.-M. Jolion. Data driven decimation of graphs. In J.-M. Jolion, W. Kropatsch, and M. Vento, editors, *Proceedings of* 3^{rd} *IAPR-TC15 Workshop on Graph based Representation in Pattern Recognition*, pages 105–114, Ischia-Italy, May 2001.

[114] J. M. Jolion and A. Montanvert. The adaptative pyramid : A framework for 2d image analysis. *Computer Vision, Graphics, and Image Processing*, 55(3) :339–348, May 1992.

[115] J.-M. Jolion and A. Rosenfeld. *A Pyramid Framework for Early Vision.* Kluwer Academic Publishers, 1994.

[116] D. B. Judd. Reduction of data on mixture of color stimuli. *Bureau Stand.J. Res.*, 4 :515–548, 1930.

[117] M. Kass, A. Witkin, and D. Terzopoulos. Snakes : Active contour models. *International Journal of Computer Vision*, 1(4) :321–331, 1988.

[118] G. Kay and T. Caelli. Estimating the parameters of an illumination model using photometric stereo. *GMIP*, 57(5) :365–388, September 1995.

[119] H. Konik, V. Lozano, and B. Laget. Color pyramids for image processing. *Journal of Imaging Science and Technology*, 40(6) :535–542, nov 1996.

[120] H. Konik, A. Trémeau, and B. Laget. Multiresolution color image analysis. In *Proceedings of the 3rd Color Imaging Conference*, pages 82–85, Scottsdale, nov 1995. IS&T/SID.

[121] R. Kozera. On shape recovery from two shading patterns. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(4) :673–692, 1992.

[122] W. G. Kropatsch. From equivalent weighting functions to equivalent contraction kernels. In E. Wenger and L. I. Dimitrov, editors, *Digital Image Processing and Computer Graphics (DIP-97) : Applications in Humanities and Natural Sciences*, volume 3346, pages 310–320. SPIE, 1998.

[123] W. G. Kropatsch and M. Burge. Minimizing the Topological Structure of Line Images. In A. Amin, D. Dori, P. Pudil, and H. Freeman, editors, *Advances in Pattern Recognition, Joint IAPR International Workshops SSPR'98 and SPR'98*, volume Vol. 1451 of *Lecture Notes in Computer Science*, pages 149–158, Sydney, Australia, August 1998. Springer, Berlin Heidelberg, New York.

[124] W. G. Kropatsch and H. Macho. Finding the structure of connected components using dual irregular pyramids. In *Cinquième Colloque DGCI*, pages 147–158. LLAIC1, Université d'Auvergne, ISBN 2-87663-040-0, September 1995.

[125] P. Kubelka and F. Lunk. Ein beitrag zur optik der farbanstriche. *Z. Techn. Physik*, 12(593), 1931.

[126] P. Lambert and L. Macaire. Filtering and segmentation : The specificity of color images. In Cépaduès, editor, *Proceedings of CGIP'2000*, pages 57–71, Saint Etienne, October 2000.

[127] C.-H. Lee. Recursive region splitting at hierarchical scope views. *Computer Vision, Graphics, and Image Processing*, 33 :237–258, 1986.

[128] C.-H. Lee, Y.-M. Koo, and Y. G. Shin. Template-based rendering of run-length encoded volumes. In *Pacific Conference on Computer Graphics and Applications*, volume 5. IEEE, 1997.

[129] P. Lienhardt. *Modélisation et évolution de surfaces libres.* PhD thesis, Université Louis Pasteur, Strasbourg, 1987.

[130] P. Lienhardt. Extension of the notion of map and subdivisions of a three-dimensional space. In *5th Annual Symposium on Theoretical Aspects of Computer Science*, volume 294 of *LNCS*, pages 301–311, Bordeaux, France, 11–13 Feb. 1988. Springer.

[131] P. Lienhardt. Free-form surfaces modeling by evolution simulation. In D. A. Duce and P. Jancene, editors, *Eurographics '88*, pages 327–341. North-Holland, Sept. 1988.

[132] P. Lienhardt. Topological models for boundary representations : a comparison with n-dimensional generalized maps. *Computer-Aided Design*, 23(1) :59–82, 1991.

[133] P. Lienhardt. Aspects in topology-based geometric modeling. *Lecture Notes in Computer Science*, 1347 :33–??, 1997.

[134] J. O. Limb, C. B. Rubinstein, and J. Thompson. Digital coding of color video signals. *IEEE Trans. Commun.*, COMM-25 :1349–1385, Nov. 1977.

[135] J. Lin, J. Storer, and M. Cohn. On the complexity of the optimal tree pruning for source coding. In *Proc. of data compression conference*, pages 63–72. IEEE computer society Press Los Angeles, 1991.

[136] W.-J. Lin and J.-C. Lin. Color quantization by preserving color distribution features. *Signal Processing*, 78 :201–214, 1999.

[137] V. Lozano. *Contribution de l'analyse d'image couleur au traitement des images textile.* PhD thesis, Université Jean Monnet, LIGIV, Saint-Etienne, France, 1998.

[138] J. Lu and J. J. Little. Reflectance and shape from images using a colinear light source. *International Journal of Computer Vision*, 32(3) :213–240, 1999.

[139] S. Mallat. Wavelets for a vision. *Proceedings of the IEEE*, 84(4) :604–614, 1996.

[140] J. Matas, P. Bilek, and O. Chum. Rotational invariants for wide-baseline stereo. In H. Wildenauer and W. Kropatsch, editors, *Proceedings of the Computer Vision Winter Workshop*, pages 296–305, Bad Ausse Austria, February 2002.

[141] C. Mathieu, I. E. Magnin, and C. Baldy-Porcher. Optimal stochastic pyramid : segmentation of mri data. *Medical Imaging IV : Image Processing (SPIE)*, 1652 :14–22, 1992.

[142] P. Meer. Stochastic image pyramids. *Computer Vision Graphics Image Processing*, 45 :269–294, 1989.

[143] C. Menard and W. G. Kropatsch. Adaptive stereo matching in correlation scale space. In A. Del Bimbo, editor, *Image Analysis and Processing, ICIAP'97*, volume Vol. 1310-I of *Lecture Notes in Computer Science*, pages 677–684, Florence, Italy, 1997. Springer, Berlin Heidelberg, New York.

[144] C. Menard and W. G. Kropatsch. An adaptive strategy for finding stereo correspondences. In W. Burger and M. Burge, editors, *Pattern Recognition 1997, Proc. of 21th ÖAGM Workshop*, pages 185–195. OCG-Schriftenreihe, Österr. Arbeitsgemeinschaft für Mustererkennung, R. Oldenburg, 1997. Band 103.

[145] B. Messmer and H. Bunke. A new algorithm for error tolerant subgraph isomorphism. *IEEETPAMI*, 20 :493–505, 1998.

[146] A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical image analysis using irregular tessellations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4) :307–316, APRIL 1991.

[147] P. F. Nacken. Image segmentation by connectivity preserving relinking in hierarchical graph structures. *Pattern Recognition*, 28(6) :907–920, June 1995.

[148] S. K. Nayard, K. Ikeuchi, and T. Kanade. Surface reflection : Physical and geometrical perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7) :611–634, July 1991.

[149] S. K. I. Nayard and T. Kanade. Surface reflection : Physical and geometrical perspectives. In *Image Understanding Workshop*, volume 2, pages 185–212, September 1990.

[150] L. O. Neumann, K. Matkovic, and W. Purgathofer. Perception based color image difference. *Computer Graphics Forum*, 17(3) :233–242, 1998. ISSN 1067-7055.

[151] Y.-I. Ohta, T. Kanade, and T. Sakai. Color information for region segmentation. *Computer Vision, Graphics, and Image Processing*, 13 :222–241, 1980.

[152] S. Orchard. Reflection and transmission of light by diffusing suspensions. *J. Opt. Soc. Am.*, 59 :1584–1597, 1969.

[153] L. A. Overturf, M. L. Comer, and E. J. Delp. Color image coding using morphological pyramid decomposition. *IEEE Transactions on Image Processing*, 4(2) :177–185, 1995.

[154] T. Pappas. Model based halftoning of color images. *IEEE Trans. Image Processing*, 6 :1014–1024, July 1997.

[155] A. P. Pentland. Finding the illuminant direction. *J. Opt. Soc. Amer. A*, 72(4) :448–455, April 1982.

[156] A. P. Pentland. Local shape analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2) :170–187, March 1984.

[157] A. P. Pentland. Shape information from shading : A theory about human perception. In *Proc. Int. Conf. Comput. Vision*, pages 404–413, 1988.

[158] M. Pharr and P. Hanrahan. Monte carlo evaluation of Non-Linear scattering equations for subsurface reflection. In S. Hoffmeyer, editor, *Proceedings of the Computer Graphics Conference 2000 (SIGGRAPH-00)*, pages 75–84, New York, July 23–28 2000. ACMPress.

[159] J. Poskanzer. Pbm+ image processing software package, 1991.

[160] F. P. Preparata and M. Shamos. *Computational Geometry*. Springer-Verlag, New York, 1985, 1985.

[161] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C*, chapter 11. Cambridge University Press, 1991.

[162] J. Puzicha, M. Held, J. Ketterer, J. M. Buhmann, and D. W. Fellner. On spatial quantization of color images. *IEEE Transaction on Image Processing*, 9(4) :666–682, April 2000.

[163] Z. Pyzlot, A. Rosenfeld, and J. Epelboim. An exponential pyramid model of the time-course of size processing. *Vision Res.*, 35(8) :1089–1107, 1995.

[164] R. Ramamoorthi. Analytic pca construction for theoretical analysis of lighting variability in images of a lambertian object. *IEEETPAMI*, 24(10) :1322–1333, October 2002.

[165] J. Reichmann. Determination of absorption and scattering coefficients for non homogeneous media. *Applied Optics*, 12 :1811–1815, 1973.

[166] A. Rosenfeld and C.-Y. Sher. Detecting image primitives using feature pyramids. *Journal of Information Sciences*, 107 :127–147, 1998.

[167] B. Roy. *Algèbre moderne et théorie des graphes*, volume 1, chapter III, page 192. Henri Hierche, dunod edition, 1969.

[168] A. Rozenfeld and A. Sher. Detection and delineation of compact objects using intensity pyramids. *Pattern Recognition*, 21 :147–151, 1988.

[169] M. Salotti and N. Laachfoubi. Topographic graph matching for shift estimation. In J. M. Jolion, W. Kropatsch, and M. Vento, editors, 3^{rd} *IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition*, pages 54–63. IAPR-TC15, May 2001.

[170] B. Schneier and H. J. Smith. Color models. *Dr. Dobb's Journal of Software Tools*, 18(7) :38, 40, 42–43, 96, July 1993.

[171] M. W. Schwarz, W. B. Cowan, and J. C. Beatty. An experimental comparison of RGB, YIQ, LAB, HSV, and opponent color models. *ACM Transactions on Graphics*, 6(2) :123–158, Apr. 1987.

[172] S. A. Shafer. Using color to separate reflection components. *Color Research and Application*, 10(4) :210–218, 1985.

[173] G. Sharma, editor. *Digital Color Imaging Handbook*. Electrical and Applied Signal Processing. CRC Press, 2002.

[174] G. Sharma and H. J. Trussell. Digital color imaging. *IEEE Transactions on Image Processing*, 6(7) :901–932, 1997.

[175] P. S. Shirley. *Physically Based Lighting Calculations for Computer Graphics*. PhD thesis, University of Illinois at Urbana-Champaign, 1991.

[176] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1) :13–32, 1999.

[177] R. Siegel and J. Howell. *Thermal Radiation heat transfer*. Mc Graw-Hill, New York, 1981.

[178] S.Thorpe, D.Fize, and C. Marlot. Speed of precessing in the human visual system. *Nature*, 381 :520–522, 1996.

[179] H. D. Tagare. Silmutaneous estimate of shape and reflectance map from photometric stereo. *Computer Vision, Graphics, and Image Processing. Image Understanding*, 55(3) :275–286, May 1992.

[180] H. D. Tagare and R. J. deFigueiredo. A theory of photometric stereo for a class of diffuse non-lambertian surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2) :133–152, February 1991.

[181] H. D. Tagare and R. J. deFigueiredo. A framework for the construction of reflectance maps for machine vision. *Computer Vision, Graphics, and Image Processing. Image Understanding*, 57(3) :265–282, May 1993.

[182] J. Tajima. Uniform color scale applications to computer graphics. *Computer Vision, Graphics, and Image Processing*, 21(3) :305–325, march 1983.

[183] S. L. Tanimoto and T. Pavlidis. A hierarchical data structure for picture processing. *Computer Graphics and Image Processing*, 4 :104–119, 1975.

[184] S. W. Thomas. Efficient Inverse Color Map Computation. In J. Arvo, editor, *Graphics Gems II*, pages 116–125, 528–535. Academic Press Professional, Cambridge, MA, 1991.

[185] J. R. A. Torreao and J. L. Fernandes. Matching photometric-stereo images. *JOSAA*, 15(12) :2966–2975, December 1998.

[186] S. Tosovic and R. Sablatnig. 3d modeling of archaeological vessels using shape from silhouette. In *Proc. of 3rd IEEE Intl. Conference on 3-D Digital Imaging and Modeling*, pages 51–58, 2001.

[187] A. Trémeau. Analyse d'images couleurs : Du pixel à la scène. Équipe Ingénérie de la Vision, Laboratoire Traitement du Signal et Instrumentations, UMR 5516., mai 1998. Habilitation à Diriger des Recherches.

[188] A. Trémeau, É. Dinet, and É. Favier. Measurement and display of color image differences based on visual attention. *Journal of Imaging Science and Technology*, 40(6) :522–534, 1996. IS&T/SID.

[189] H. J. Trussel. Application of set theoric models to color systems. *Color Res. Appl.*, 16(1) :31–41, 1991.

[190] W. Tutte. *Graph Theory*, volume 21. Addison-Wesley, encyclopedia of mathematics and its applications edition, 1984.

[191] P. Vautrot, N. Bonnet, and M. Herbin. Comparative study of different spatial-frequency method (gabor filters, wavelets, wavelet packets) for texture segmentation/classification. In *IEEE International Conference Image Processing ICIP'96*, Lausanne, 1996.

[192] O. Verevka and J. Buchanan. Local K-Means Algorithm for Color Image Quantization. *Graphics/Vision Interface '95*, May 1995.

[193] S. Wan, S. Wong, and P. Prusinkiewicz. An algorithm for multidimensional data clustering. *ACM Trans. Math. Softw.*, 14(4) :153–162, 1988.

[194] S. J. Wan, P. Prusinkiewicz, and S. K. M. Wong. Variance-Based Color Image Quantization for Frame Buffer Display. *Color Research and Application*, 15(1) :52–58, 1990.

[195] J. Weickert, editor. *Anisotropic Diffusion in Image Processing*. Teubner, B.G., 1998.

[196] S. Wong, S. Wan, and P. Prusinkiewicz. Monochrome image quantization. In *Canadian conference on electrical and computer engineering*, pages 17–20, September 1989.

[197] P. L. Worthington and E. R. Hancock. Surface topography using shape-from-shading. *Pattern Recognition*, 34 :823–840, 2001.

[198] W. Wright. The sensitivity of the eye to small colour differences. *Proc. Phys. Soc.*, 53(296) :93–112, 1941. Part 2.

[199] X. Wu. Efficient statistical computations for optimal color quantization. In J. Arvo, editor, *Graphics Gems II*, volume II, chapter 3, pages 126–133. Academic Press, 1991.

[200] X. Wu. Color quantization by dynamic programing and principal analysis. *ACM Transaction on Graphics*, 11(4) :348–372, 1992.

[201] X. Wu and K. Zhang. A better tree-structured vector quantizer. In *Proceedings of the IEEE Data Compression Conference*, pages 392–401. IEEE Computer Society Press, 1991.

[202] G. Wyszecki and W. S. Stiles. *Color Science : Concepts and Methods, Quantitative Data and Formulae*. Wiley, 1982.

[203] Z. Xiang. Color image quantization by minimizing the maximum intercluster distance. *ACM Transactions on Graphics*, 16(3) :260–276, July 1997.

[204] R. Zhang, P.-S. Tsai, J. E. Cryer, and S. Mubarak. Shape from shading : A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8) :690–705, August 1999.

[205] Q. Zheng and R. Chellappa. Estimation of illuminant direction, albedo, and shape from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7) :680–702, July 1991.

[206] S. W. Zucker. Region growing : Childhood and adolescence. *Computer Graphics, and Image Processing*, 5(3) :382–399, Sept. 1976. ZUCKER76b.

Index