Système avancé d'interpolation spatiale de signaux de télévision pour affichage sur écrans Haute-Définition

Eric van Reeth

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Signal Image Parole Télécommunication**

Arrêté ministériel : 7 août 2006

Présentée par

Éric VAN REETH

Thèse dirigée par **Jean-Marc CHASSERY** et
codirigée par **Pascal BERTOLINO**

préparée au sein du **Laboratoire GIPSA-lab** et de l'entreprise STMicroelectronics
dans **l'École Doctorale EEATS**

Système avancé d'interpolation spatiale de signaux de télévision pour affichage sur écrans haute-définition

Thèse soutenue publiquement le **10 mai 2011**,
devant le jury composé de :

Mme Valérie PERRIER
Présidente
Mme Laure BLANC-FERAUD
Rapporteur
M. Frédéric TRUCHETET
Rapporteur
M. Yannick BERTHOUMIEU
Membre
M. Jean-Marc CHASSERY
Membre
M. Pascal BERTOLINO
Membre
Mme Marina NICOLAS
Membre

Université de Grenoble

N° attribué par la bibliothèque

THÈSE

pour obtenir le grade de

Docteur de l'Université de Grenoble

Spécialité : **Signal Image Parole Télécommunication**

préparée au sein du laboratoire **GIPSA-lab**
Grenoble Images Parole Signal et Automatique

et de l'entreprise **STMicroelectronics**

dans le cadre de l'École Doctorale **EEATS**
(Électronique-Électrotechnique-Automatique-Traitement du Signal)

présentée et soutenue publiquement par

Eric VAN REETH

le 10 mai 2011

Titre :

Système avancé d'interpolation spatiale de signaux de télévision pour affichage sur écrans haute-définition

Directeur de thèse GIPSA-lab : **Jean-Marc CHASSERY**
Co-Directeur de thèse GIPSA-lab : **Pascal BERTOLINO**
Encadrante industrielle STMicroelectronics : **Marina NICOLAS**

Jury

Madame Valérie PERRIER,	Présidente du jury
Madame Laure BLANC-FERAUD,	Rapporteur
Monsieur Frédéric TRUCHETET,	Rapporteur
Monsieur Yannick BERTHOUMIEU,	Examinateur
Monsieur Jean-Marc CHASSERY,	Examinateur
Monsieur Pascal BERTOLINO,	Examinateur
Madame Marina NICOLAS,	Invitée

Remerciements

Afin de permettre au lecteur de commencer sur une note joyeuse, je souhaiterais débuter ce mémoire en remerciant les personnes qui ont participé de près ou de loin à sa rédaction.

Je commencerai par remercier les membres du jury qui ont accepté d'évaluer mes travaux. Madame Valérie Perrier, pour avoir accepté de prendre en charge la gestion et la présidence de ce jury ; Madame Laure Blanc-Féraud, pour la qualité de ses remarques écrites ainsi sa vision globale et son recul dans le large domaine du traitement d'image ; Monsieur Frédéric Truchetet, pour son expertise dans le domaine des ondelettes et ses nombreuses interventions sur la globalité de l'approche ; Monsieur Yannick Berthoumieu, pour la précision de ces questions durant la soutenance et l'intérêt qu'il a porté à nos travaux.

Bien évidemment, je souhaite remercier mes encadrants de thèse, à commencer par Marina pour la qualité de son encadrement au jour le jour et pour la liberté de recherche qu'elle m'a laissée. Je tiens ensuite à remercier Jean-Marc pour la confiance qu'il accorde aux collaborations industrielles et pour la qualité de ses retours sur le mémoire et sur la soutenance. Enfin, je remercie Pascal pour la qualité à la fois scientifique et personnelle de son encadrement et son soutien pendant ces trois années.

Je tiens également à remercier Jocelyn Chanussot et Jérôme Mars du Gipsa-lab qui ont chacun leur tour été présents lors de ma formation, et qui m'ont donné le goût du traitement d'image.

Un grand merci aux collègues de ST, ainsi qu'aux ex-ST ; Jérôme, pour les innombrables discussions et débats scientifiques, à qui je joins Stéphane et Nico pour les éreintantes séances d'escalades du midi et les non-moins éreintantes soirées poker. Je n'oublie pas non plus Pascal, David et la glorieuse équipe de handball de ST (Herve, Seb, Pawel, Fred, Romain, Jacques, Alex, Perrine, Julien, Henri, Aurélien, Armand, Xavier,...) que j'abandonne à regret. Une pensée spéciale pour Claire qui a eu le courage d'aller jusqu'au bout de sa thèse, et avec qui j'ai passé entre autres, de très bons moments en conférence.

Enfin, un immense merci à Mélie qui m'a soutenu pendant trois ans, à sa compréhension, ses sourires et ses gâteaux. A ma famille, Françoise, Patrick, Aude, Hugo et Colin, ainsi qu'à tous mes amis qui ont été là, et dont la liste est aussi longue qu'incomplète : Mag, Clem, Sara, Max, Aurore(s), Chris, Manue, Fab, Sam, Léo, Henri, Hugo, Coco, au SHMO, Arnaud, Julien,...
Grâce à eux tous, la thèse ne sera plus qu'un bon souvenir.

Table des matières

Table des figures

Introduction Générale

L'amélioration de résolution (ou interpolation) d'images ou de vidéos est une technique très utilisée et largement étudiée du fait de son grand nombre d'applications. Ce procédé consiste à adapter le nombre de pixels contenu dans l'image ou la vidéo source à la résolution du support sur lequel cette source est affichée. L'arrivée massive des écrans haute-définition ainsi que la variété de la résolution des sources à afficher (téléphones portables, appareils photos et vidéos numériques, télévisions numériques) impliquent un besoin grandissant de procédés d'interpolation, illustré par la Figure 1 ci-dessous. Les utilisations de tels procédés sont variés allant de l'agrandissement

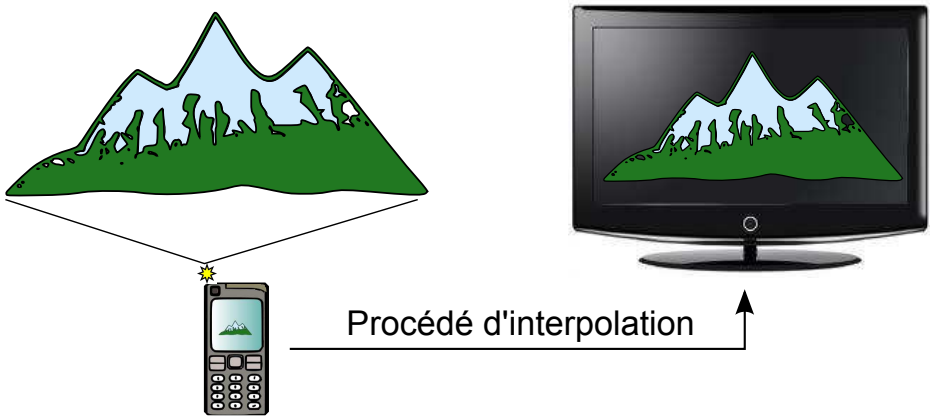


FIGURE 1 – Illustration d'un procédé nécessitant une interpolation.

d'images médicales utiles pour l'aide au diagnostic, au traitement d'images satellites, ou encore à la conversion d'un flux vidéo de résolution standard (SD : 720×480 pixels pour les DVD standards) à des contenus plus résolus dits de haute-définition (HD : 1920×1080 pixels dans le cas du Blu-ray et du HD-DVD). Dans ce cas précis, il est nécessaire de créer environ six nouveaux pixels pour chaque pixel de l'image originale.

Toute la difficulté de ces méthodes réside dans le fait qu'il n'est pas possible d'obtenir plus d'information dans l'image HD que celle contenue dans l'image de départ. Les nouveaux pixels créés doivent donc optimiser le rendu visuel de l'information de départ sur une grille de pixels plus dense, en essayant de conserver au mieux les caractéristiques des objets contenus dans la source. De plus, l'utilisation d'écrans haute-définition dans de plus en plus d'applications diverses (TV, biomédical, imagerie sismique, imagerie satellitaire) implique l'affichage de flux vidéos de contenus extrêmement variés. Les procédés d'interpolation doivent donc idéalement être polyvalents, même s'il est difficile d'effectuer des hypothèses communes cohérentes pour des signaux dont les caractéristiques sont très différentes. Selon les besoins et les contraintes du procédé qui requiert une interpolation, des méthodes plus ou moins complexes peuvent être mises en place. En effet, certains procédés comme l'interpolation de flux de télévision doivent être traités en temps réel, alors que l'interpolation de photographies par un logiciel de retouche spécialisé peut prendre jusqu'à quelques dizaines de secondes par image. Naturellement, la qualité des résultats obtenus

dépend de la complexité des algorithmes utilisés ainsi que des ressources mises en oeuvre pour l'application de l'interpolation.

De manière générale, les méthodes d'interpolation se divisent en deux grandes catégories, que sont les méthodes linéaires et les méthodes non-linéaires. Les méthodes linéaires les plus courantes sont par exemple :

- la réplication de pixels qui consiste à reproduire les pixels de l'image originale sur les pixels manquants de la grille haute-résolution.
- l'interpolation bilinéaire qui effectue une régression linéaire entre les pixels existants pour créer les nouveaux.

Par la suite, des procédés d'ordres supérieurs ont été introduits tels que l'interpolation cubique [Keys 1981], ou encore spline [Unser 1999]. Bien qu'efficaces au niveau temps de calcul, ces méthodes présentent des artefacts bien connus qui sont surtout visibles au niveau des contours des objets de l'image (textures comprises). Les plus visibles sont : le flou, l'effet d'escalier, le ringing (échos de contours), ou encore l'aliasing. Ces défauts sont particulièrement préjudiciables puisqu'en altérant la qualité des contours, ils affectent les structures qui attirent le plus l'attention visuelle. En effet, les études sur les cartes de saillance montrent que les hautes fréquences créées par les discontinuités de niveaux de gris au niveau des contours, concentrent l'attention de l'oeil humain.
Pour corriger ces défauts, des méthodes dites adaptatives ou non-linéaires sont apparues au début des années 90. Elles cherchent à s'adapter aux contours de l'image pour préserver au mieux leurs propriétés dans l'image haute résolution. Une phase préalable d'analyse des caractéristiques des contours ou de modélisation de la chaîne d'acquisition de l'image est nécessaire pour ce genre de méthodes, ce qui explique leur plus grande complexité algorithmique. Parmi les méthodes d'interpolation non-linéaires, nous pouvons citer : [Carrato 1996], [Biancardi 2002] et [Aly 2005].
Toujours dans un esprit de préserver l'aspect des contours, d'autres méthodes se sont intéressées à adapter le repère cartésien orthogonal dans lequel sont généralement représentées les images en fonction de l'orientation des contours, comme illustré dans la Figure 2. Cette rotation permet de représenter la discontinuité créée par le contour comme étant régulière sur l'axe parallèle au contour (axe x), et discontinue sur l'axe orthogonal (axe y). L'interpolation peut alors être réalisée le long de l'axe régulier afin d'éviter les problèmes introduits par l'interpolation de structures discontinues, et l'aspect de la transition peut alors être préservé.

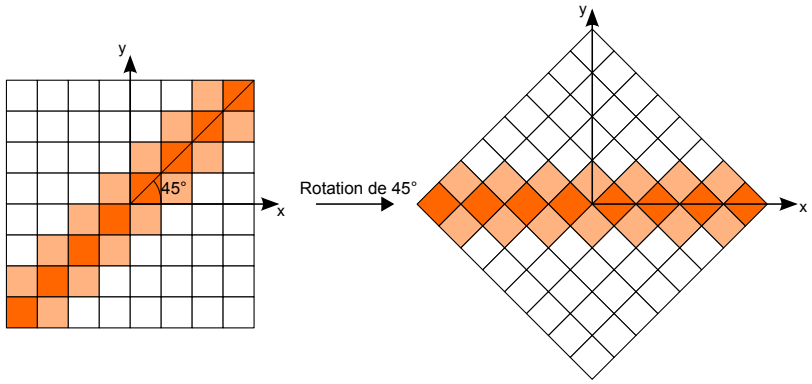


FIGURE 2 – Changement d'orientation pour le traitement plus efficace d'un contour.

De nombreuses méthodes se sont focalisées sur la détection de la direction de contours, afin de privilégier l'interpolation de pixels sur des zones régulières. Nous pouvons citer les premières approches de [Algazi 1991], [Bayrakeri 1995], [Jensen 1995], [Allebach 1996] sur le sujet. Plus récemment, d'autres auteurs se sont intéressés aux méthode dites d'interpolations directionnelles parmi lesquelles se trouvent les approches de [Li 2001], [Jiang 2002], [Muresan 2005], [Zhang 2006], [Wang 2007] et [Xiong 2009]. Par rapport aux interpolations linéaires, ces méthodes ont une com-

plexité algorithmique supérieure, et impliquent donc un nombre de calculs plus important. Elles sont en effet en général basées sur une étape préalable d'estimation de la direction des contours et les résultats obtenus sont forcément dépendants de la qualité de cette estimation. Cependant, l'amélioration visuelle des résultats par rapport aux autres techniques est nettement visible, et c'est pourquoi l'interpolation directionnelle est un axe de recherche prédominant au niveau académique mais également au niveau industriel.

Démarche industrielle

C'est en effet dans cette optique que cette thèse a été mise en place à Grenoble, en tant que collaboration entre GIPSA-lab, laboratoire de recherche dépendant du CNRS et de Grenoble INP, spécialisé dans les domaines de l'automatique, parole, cognition, traitement du signal et de l'image, et STMicroelectronics, entreprise internationale dans le domaine des micro-conducteurs et menant sur le site de Grenoble des activités de recherche en traitement d'images. Plus précisément, ces activités ont lieu au sein de l'équipe de recherche et développement du département HVD (Home Video Display) comptant une soixantaine de personnes travaillant principalement sur la conception de circuits destinés à être incorporés dans des boîtiers multimédia, ou *set-top box*. Ces boîtiers sont principalement chargés du décodage d'un flux vidéo pour l'affichage du contenu sur un téléviseur. Ils contiennent également des algorithmes de traitement d'images chargés d'optimiser le rendu visuel de l'image affichée. Un de ces algorithmes est chargé du redimensionnement de la source. Le redimensionnement consiste à la fois à agrandir l'image source, mais également à la réduction de la taille de l'image, procédé qui pose également problème en particulier dans le cas du re-échantillonnage des hautes-fréquences de l'image. Actuellement, les méthodes de redimensionnement utilisées dans les boîtiers numériques sont principalement limitées à des méthodes linéaires, de type interpolation de Lanczos, et autorisent un traitement rapide utilisant peu de mémoire nécessaire au stockage des variables temporaires requises par l'algorithme. Avec l'arrivée massive des écrans haute-définition, la conversion des flux vidéos en HD fait face à une demande croissante des utilisateurs. Les entreprises impliquées dans la fabrication des boîtiers multimédia cherchent à optimiser le rendu visuel d'images HD et s'orientent donc vers des algorithmes d'interpolation plus complexes. Les travaux de Mallat sur la super-résolution basée sur la transformée en bandelettes, idéale pour représenter une image dans des espaces adaptés aux objets qui la composent, ont été remarqués dans le milieu industriel lors de la fondation de l'entreprise *Let It Wave*, en 2001 entre autres par Stéphane Mallat.

Les travaux réalisés pendant cette thèse s'inscrivent dans la continuité de cette démarche, et le sujet de thèse précis consiste à développer un algorithme avancé et novateur d'interpolation d'images pour l'affichage sur des écrans de haute-résolution. Nous nous sommes donc naturellement orientés vers les méthodes d'interpolations directionnelles, qui d'après nous présentaient des résultats prometteurs, et vers les analyses par transformations dans des espaces d'ondelettes géométriques comme par exemple les bandelettes [Mallat 2005], [Mallat 2007], [LePennec 2002], les curvelets [Starck 2002] ou encore les contourlets [Do 2005], qui autorisent l'analyse d'images dans des espaces optimaux.

Guide de lecture

Nos travaux présentés ici s'inscrivent dans le cadre des méthodes d'interpolation directionnelles non-linéaires, basés sur une analyse multirésolution inspirée des transformées en ondelettes géométriques. Nous nous sommes principalement intéressés au traitement d'images fixes afin de bien observer les difficultés rencontrées et les enjeux à cerner, sans introduire le paramètre temporel qui aurait impliqué un surplus de données à traiter important. Cependant, l'algorithme mis

au point pour des images fixes, peut tout-à-fait être appliqué à des séquences d'images.

Dans **une première partie** dédiée à l'état de l'art, les concepts théoriques utilisés dans notre étude sont introduits afin de lui donner un cadre théorique cohérent. Les concepts de géométrie et de directions de contours, primordiaux dans notre approche et utilisés dans l'espace discret dans lequel les images numériques sont représentées, seront introduits dans un premier temps. Puis, nous évoquerons la construction théorique des espaces multirésolutions engendrés par les transformées en ondelettes. Nous verrons que la projection d'une image dans des espaces de résolutions différentes introduit une répartition des composantes de l'image en fonction de leur fréquence. Enfin, un état de l'art des méthodes d'interpolation linéaires est rédigé pour conclure cette partie. Il vise à évoquer les principales techniques d'interpolation, encore souvent utilisées dans les applications de tous les jours, du fait de leur simplicité algorithmique et du faible temps de calcul qu'elles requièrent. Nous illustrerons les artefacts engendrés par ces méthodes, et nous justifierons le besoin d'introduire des méthodes plus complexes pour optimiser le rendu visuel des images interpolées. La suite du manuscrit décrit le contenu ainsi que l'apport de nos travaux et peut être clairement divisée en deux parties pouvant être lues et comprises indépendamment l'une de l'autre.

La deuxième partie s'intéresse à la recherche des directions des contours d'une image. Les contours, selon leur fréquence, sont répartis grâce aux transformées d'ondelettes dans des espaces de résolution adaptés afin d'optimiser l'estimation de leur direction. L'estimation de direction est ensuite effectuée de manière non-linéaire et gérée par une division en *quadtree* qui permet le découpage de l'image en régions carrées ne contenant qu'une seule direction de contour. La direction du contour isolé est ensuite estimée en trouvant la direction de la droite parallèle au contour. La méthode d'estimation de direction sera évaluée à la fin de la première partie, et comparée à deux autres techniques pour étudier les points faibles et forts de chacune, et mettre en avant l'apport de notre approche.

En troisième et dernière partie, notre méthode d'interpolation basée sur les orientations de contours estimées est présentée, ainsi qu'un état de l'art récent des méthodes d'interpolation directionnelles. Nous verrons comment dans notre cas, l'information de direction est utilisée et comment les artefacts qui apparaissent lors des interpolations classiques sont éliminés, sans que de nouveaux défauts soient introduits. Notre approche sera comparée à celles introduites dans l'état de l'art de cette partie afin de mesurer l'apport visuel de notre méthode, et d'en montrer également les limites. Enfin, les perspectives et améliorations seront évoquées afin de mesurer le travail restant pour une éventuelle implémentation dans un circuit.

Première partie

Etat de l'art

Chapitre 1

Notions de géométrie discrète

Les notions de géométrie discrète abordées dans cette partie ont pour but d'introduire l'espace discret dans lequel nous évoluerons tout au long de ce rapport, ainsi que les objets qui lui sont associés. Des analyses détaillées de ces principes peuvent être trouvées dans : [Montanvert 1991] et [Sivignon 2004].
Nous appelons espace discret de dimension n, tout pavage régulier de $\mathbb{R}^n$. En dimension 2, le pavage régulier du plan le plus commun est un pavage composé de carrés. Un point discret est défini comme étant le centre de gravité d'une cellule du pavage du plan, et sont représentés par leur coordonnées discrètes. La images que nous allons étudier tout au long de cette étude sont représentées dans des espaces de dimension 2, dans lesquels les pavés sont des carrés appelés **pixels**. Les points discrets de l'espace discret de dimension 2 sont alors les points de $\mathbb{Z}^2$, voir figure 1.1.

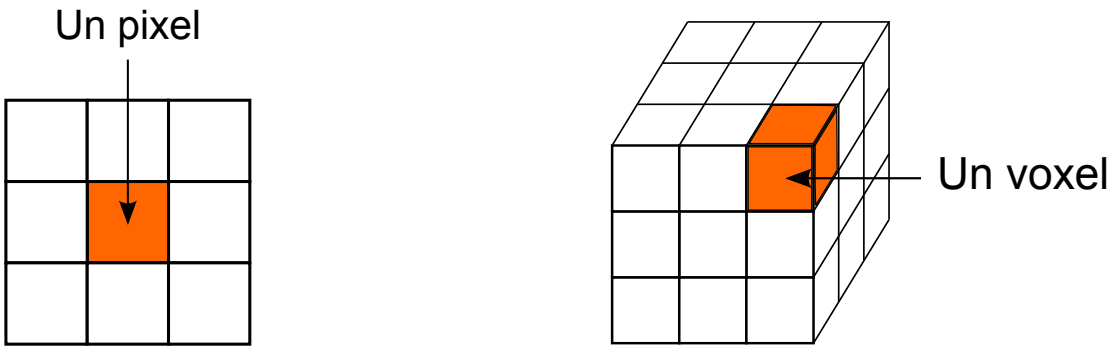


FIGURE 1.1 – Représentation des espaces (a) $\mathbb{Z}^2$ et (b) $\mathbb{Z}^3$

Afin de pouvoir définir des objets discrets, il est nécessaire de définir la notion de voisinage. En dimension 2, il y a deux situations pour lesquelles deux pixels sont dits *voisins*.

- Le cas dit 4-connexité, où deux pixels sont voisins s'ils ont une arête commune. Dans ce cas là, un pixel a au plus quatre voisins (voir Figure 1.2.(a))
- La cas dit 8-connexité, où deux pixels sont voisins s'ils ont une arête ou un sommet communs. Dans ce cas là, un pixel a au plus huit voisins (voir Figure 1.2.(b)).

Ces notions de voisinages sont utilisées pour décrire des droites discrètes que nous allons étudier maintenant.

(a) Voisinage 4-connexité (b) Voisinage 8-connexité

FIGURE 1.2 – Représentation des voisinages connexes possibles dans $\mathbb{Z}^2$

1 Droites discrètes

Les droites discrètes font partie des éléments de base de la géométrie discrète. Elles sont la représentation sur une grille discrète $2D$ d'une droite continue, selon une méthode de discrétisation donnée. La caractérisation d'une droite discrète peut se faire de plusieurs manières : par caractérisation géométrique ou par caractérisation paramétrique. La caractérisation géométrique s'intéresse aux enchaînements des pixels qui composent la droite discrète alors que la caractérisation paramétrique propose une identification grâce à quatre paramètres. Tout au long de ce manuscript, nous nous intéressons à l'étude de structures orientées discrètes dont la direction doit être estimée. La représentation paramétrique présente l'avantage de définir simplement l'orientation des droites grâce à deux de ses paramètres (a, b). C'est pour cette raison que cette représentation est utilisée tout au long de l'étude. Cette définition a été proposée par Reveillès [Reveilles 1991] en 1991.

Définition 1.1 ***(Droites discrètes 2D analytiques)***
Une droite discrète de paramètres (a, b, μ) et d'épaisseur ω (avec a, b, μ et ω dans $\mathbb{Z}$ et a et b premiers entre eux), notée $D(a, b, \mu, \omega)$, est définie comme l'ensemble des points discrets (x, y) vérifiant la double inégalité :

$$0 \leq ax - by + \mu < \omega \tag{1.1}$$

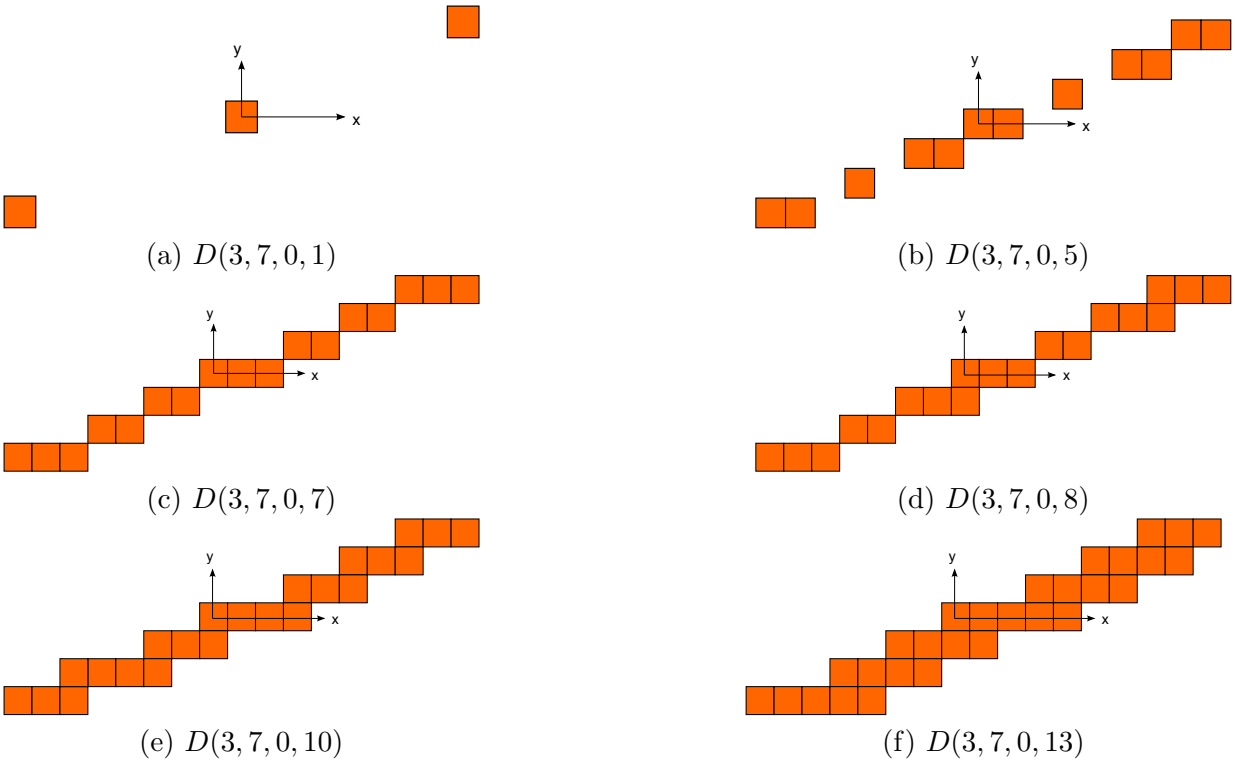
Cette droite a pour vecteur normal $(a, -b)$, pour décalage μ et pour épaisseur ω.
Une direction quelconque θ est donc représentée par la droite discrète dont les paramètres sont tels que : $\arctan(\frac{a}{b}) = \theta$.
Le paramètre d'épaisseur ω permet de gérer la connexité de la droite. La connexité de la droite selon ses paramètres est étudiée dans le théorème suivant :

Théorème 1.1 ***(Connexité des droites analytiques)***
Soit D une droite discrète de paramètres (a, b, μ) et d'épaisseur ω. Alors :
- *si $\omega < max(|a|, |b|)$, D n'est pas connexe ;*
- *si $\omega = max(|a|, |b|)$, D est 8-connexe, aussi appelée droite naïve ;*
- *si $max(|a|, |b|) < \omega < |a| + |b|$, D est $*$-connexe ;*
- *si $\omega = |a| + |b|$, D est 4-connexe ;*
- *si $\omega > |a| + |b|$, D est dite épaisse ;*

Une droite $*$-connexe est une droite dont deux pixels successifs sont 4-connexes, ou 8-connexes.

Dans notre étude, nous nous intéressons également à des droites d'épaisseur minimale. Ces droites appelées **droites fines** sont définies telles que : $\omega = 1$. Elles présentent la particularité d'être non connexes, et d'être composées de pixels tous situés à la même distance les uns des autres. En effet, chaque pixel est distant de (a, b) de ses voisins, ce qui fait de ces droites la représentation discrète la plus précise possible d'une direction continue. Nous utiliserons ces droites dans la troisième

partie de cette étude, concernant l'interpolation directionnelle. Une illustration de chacune des droites définie est montrée en Figure 1.3.



(a) $D(3,7,0,1)$ (b) $D(3,7,0,5)$

(c) $D(3,7,0,7)$ (d) $D(3,7,0,8)$

(e) $D(3,7,0,10)$ (f) $D(3,7,0,13)$

FIGURE 1.3 – Représentation de la connectivité de la droite discrète $(3,7,0)$ en fonction du paramètre d'épaisseur ω.

Enfin, la notion de reste, autrement appelé décalage, sert à situer des points sur une droite discrète.

Définition 1.2 ***(Reste)***
Soit D une droite discrète de paramètres (a,b,μ), d'épaisseur ω et p un point discret de coordonnées (x,y). Alors le reste associé au point p est la valeur :

$$r(p) = ax - by + \mu \tag{1.2}$$

On peut déduire d'après l'équation 1.1 qu'un point p appartient à la droite discrète D si et seulement si $0 \leq r(p) \leq \omega - 1$. Cette équation permet de conclure que le reste peut prendre ω valeurs différentes. Ainsi pour une droite naïve, le reste prend $\omega = max(|a|,|b|)$ valeurs différentes comme le montre le tracé d'une droite naïve en Figure 1.4. En faisant varier le décalage μ, il est donc possible d'obtenir ω représentations discrètes disctinctes de la même direction. Ainsi, une droite naïve aura $max(|a|,|b|)$ représentations distinctes de la même direction (sept dans le cas de l'exemple), alors qu'une droite fine n'aura qu'une seule représentation discrète possible d'une direction ($\omega = 1$). Ceci confirme le fait que la représentation discrète d'une direction sur une droite fine est théoriquement parfaite.

Les propriétés des droites discrètes définies dans cette partie seront utilisées par la suite pour rechercher la direction d'un contour dans un voisinage donné. Les voisinages utilisés sont des blocs carrés de taille $(N \times N)$ pixels. Contrairement au cas continu, le nombre de droites discrètes distinctes que l'on peut construire sur une grille discrète dans un voisinage donné est limité, car a et b sont bornés par N. Le nombre de directions que l'on peut tracer est donc de la même manière limité par la taille du voisinage utilisé. Pour connaître le nombre de directions présentes dans un

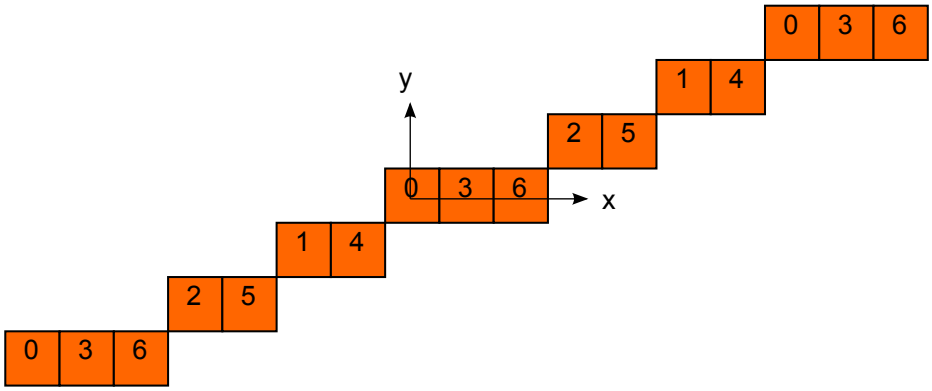


FIGURE 1.4 – Valeurs du reste pour une droite naïve de décalage nul : $D(3, 7, 0, 7)$

voisinage borné de taille $(N \times N)$ pixels, il faut calculer le nombre de couples d'entiers $(a, b) \in \mathbb{Z}^2$ premiers entre eux tels que $|a| < N$ et $|b| < N$. Cette série de couple particulière s'appelle en réalité une suite de Farey, et nous allons l'étudier dans le paragraphe suivant afin de calculer le nombre de directions possibles que l'on peut tracer dans un bloc carré.

2 Suites de Farey

La construction des suites de Farey remonte au début des années 1800. Son invention présente aujourd'hui des utilisations nombreuses de par son application directe au domaine discret. Les suites (ou séries) de Farey sont une technique qui énumère et représente de manière ordonnée toutes les fractions rationnelles positives irréductibles $\frac{a}{b}$ où a et b sont premiers entre eux. Un descriptif détaillé des définitions présentées ici peut être trouvé dans [Crowe 2000], [Guy 1995] et [Patashnik 1994].

2.1 Définition et propriétés

Définition 2.1 *La suite de Farey d'ordre N, notée $\mathcal{F}_N$, est la suite de fractions rationnelles irréductibles comprises entre 0 et 1 dont le dénominateur est inférieur ou égal à N.*

Cette suite permet donc de connaître les couples d'entiers (a, b) premiers entre eux, bornés par N pour $a \leq b$. Ces couples correspondent aux paramètres de direction d'une droite discrète (a, b), dont la direction est comprise dans le premier octant d'angle $(a \leq b)$, à savoir : $[0, \pi/4]$, dans un voisinage de taille $(N + 1) \times (N + 1)$ pixels. Par exemple, la suite de Farey pour $N = 5$ est telle que :

$$\mathcal{F}_5 = \frac{0}{1}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{1}{1} \tag{1.3}$$

Ainsi, dans un bloc de taille (6×6) pixels, il y a onze droites discrètes dont les paramètres de direction (a, b) sont bornés par 5.

La construction d'une suite de Farey d'ordre q se fait de manière récursive à partir de la suite d'ordre 1 : $\mathcal{F}_1 = \frac{0}{1}, \frac{1}{1}$. La suite de Farey d'ordre q contient tous les éléments des suites de Farey d'ordre inférieur, auxquels on ajoute les médians des éléments consécutifs de $\mathcal{F}_{q-1}$ qui ont un dénominateur inférieur ou égal à q. La notion de médian, dans ce contexte, de deux fractions rationnelles est définie dans la définition suivante :

Définition 2.2 *Le médian de deux fractions rationnelles $\frac{u}{v}$ et $\frac{u'}{v'}$ est la fraction rationnelle $\frac{u+u'}{v+v'}$.*

Le tableau de la Figure 1.5 contient l'enchaînement des suites de Farey en suivant la définition 2.2. Notons également que le médian d'une suite de Farey est toujours égal à $\frac{1}{2}$. Cette propriété peut

F_1	0/1															1/1
F_2	0/1								1/2							1/1
F_3	0/1				1/3				1/2				2/3			1/1
F_4	0/1		1/4		1/3				1/2				2/3		3/4	1/1
F_5	0/1	1/5	1/4		1/3		2/5		1/2		3/5		2/3		3/4	4/5	1/1

FIGURE 1.5 – Enchaînement des suites de Farey pour $q = 1, \ldots, 5$.

être intéressante au niveau de l'implémentation lors de la construction de droites discrètes puisque seules les droites appartenant à la première moitié du premier octant doivent être calculées. Les autres sont obtenues par symétrie.

Pour calculer de manière systématique le nombre d'éléments composant une suite de Farey, il faudrait connaître le nombre d'éléments ajoutés lors du passage à une suite d'ordre supérieur. Par récurrence, il serait alors possible de calculer le nombre d'éléments composant une suite de Farey de n'importe quel ordre. Cette quantité a été étudiée par Euler et est appelée *indicateur d'Euler*. Ce dernier va être introduit dans le paragraphe suivant, à partir duquel nous pourrons calculer le nombre d'éléments composant une suite de Farey d'ordre quelconque.

2.2 Indicateur d'Euler

En théorie des nombres, l'indicateur $\varphi(n)$ d'un entier positif n est défini comme le nombre d'entiers positifs inférieurs ou égaux à n et premiers avec n. La fonction $\varphi(n)$ ainsi définie est la fonction indicatrice d'Euler. Par exemple, on a $\varphi(8) = 4$ car les chiffres 1, 3, 5, 7 sont premiers avec 8.
Il est intéressant de chercher à généraliser le calcul de cet indicateur. Il découle de la définition de $\varphi(n)$ certaines propriétés. Ainsi, on a $\varphi(1) = 1$, et :

$$\varphi(n) = (p-1)p^{k-1} \tag{1.4}$$

quand n est la k-ième puissance d'un nombre premier p^k. De plus, si m et n sont premiers entre eux alors :

$$\varphi(mn) = \varphi(m)\varphi(n) \tag{1.5}$$

L'indicateur d'Euler peut donc être calculé grâce à ces propriétés et grâce à la théorie fondamentale de l'arithmétique fondée par Gauss et basée sur le lemme d'Euclide. Ce théorème dit que chaque entier strictement positif peut être écrit comme un produit de nombres premiers d'une unique façon, à l'ordre près des facteurs. Soit n un nombre entier strictement positif. On a donc :

$$n = p_1^{k_1} \ldots p_r^{k_r} \tag{1.6}$$

où les p_r sont des nombres premiers disctincts.
D'après l'équation 1.6, on peut donc écrire $\varphi(n)$ de la manière suivante :

$$\varphi(n) = \varphi(p_1^{k_1} \ldots p_r^{k_r}) \tag{1.7}$$

Ordre de la suite de Farey	1	2	3	4	5	6	7	8	9	10	11	12
Nombre d'éléments	2	3	5	7	11	13	19	23	29	33	43	47

FIGURE 1.6 – Nombre d'éléments contenus dans les 12 premières suites de Farey.

D'après l'équation 1.5, on a donc :

$$\varphi(n) = \varphi(p_1^{k_1}) \dots \varphi(p_r^{k_r}) \tag{1.8}$$

Enfin, d'après l'équation 1.4, l'indicateur d'Euler se calcule comme suit :

$$\varphi(n) = (p_1 - 1)p_1^{k_1 - 1} \dots (p_r - 1)p_r^{k_r - 1} \tag{1.9}$$

La définition de l'indicatrice d'Euler va nous permettre de calculer le nombre de termes composant une suite de Farey $\mathcal{F}_N$. En effet, l'indicatrice d'Euler d'un nombre N correspond au nombre d'éléments supplémentaires composant la suite de Farey $\mathcal{F}_N$ par rapport à la suite d'ordre inférieur $\mathcal{F}_{N-1}$. Soit $Card(\mathcal{F}_N)$ le nombre de termes de $\mathcal{F}_N$, on a donc :

$$Card(\mathcal{F}_N) = Card(\mathcal{F}_{N-1}) + \varphi(N) \tag{1.10}$$

Ce qui revient à écrire :

$$Card(\mathcal{F}_N) = 1 + \sum_{m=1}^{N} \varphi(m) \tag{1.11}$$

En intégrant le résultat de l'équation 1.9 dans cette équation ou en cherchant directement les valeurs de $\varphi(n)$ dans une table, le nombre d'éléments composant une suite de Farey $\mathcal{F}_N$, peut être déterminé de manière exacte. Le résultat obtenu correspond également au nombre de directions définies par des paramètres de droites discrètes (a, b), appartenant au premier octant d'angle et à l'intérieur d'un bloc de taille $(N + 1) \times (N + 1)$. A partir de ce résultat, le nombre d'angles de l'intervalle complet $[0, \pi]$ peut être calculé en évitant toutefois de compter deux fois les directions correspondant aux bornes des intervalles. L'évolution du nombre d'éléments composant les suites de Farey jusqu'à l'ordre douze est illustrée en Figure 1.6. Il est possible de remarquer dans ce tableau qu'à part la suite d'ordre 1, toutes les suites contiennent un nombre impair d'éléments.

Ce chapître nous a permi de définir le domaine discret dans lequel nous évoluerons tout au long de cette étude. Il a également servi d'introduction aux droites discrètes, ainsi qu'à leur représentation paramétrique qui permet de représenter efficacement leur direction. La notion d'épaisseur a également été abordée. Celle-ci fait varier la connexité des droites, en leur conférant des propriétés particulières selon qu'elles soient fines ou épaisses. Ces concepts seront utilisés plus tard dans l'étude pour l'analyse directionnelle des contours, effectuée dans des blocs carrés.
D'autre part, la taille des blocs utilisés influence le nombre de droites discrètes contenue dans ce voisinage que l'on peut construire, et donc le nombre de direction que l'on peut représenter. L'étude des suites de Farey puis de l'indicatrice d'Euler nous ont permis de quantifier exactement l'évolution de ce nombre. Cette étude va donc nous permettre de connaître la résolution angulaire maximale que l'on peut obtenir selon la taille du bloc utilisé, ce qui nous sera utile pour choisir par la suite les tailles de blocs optimales.

Chapitre 2

Ondelettes

Avant que les ondelettes ne deviennent un outil commun à de nombreux domaines, elles furent introduites par Gabor dans des domaines tels que le traitement de signal ou encore la physique théorique et par Morlet lors de ses travaux en sismologie. Pour étudier les couches terrestres trop fines pour être détectées par les ondes classiques, il eut l'idée de réduire la longueur de l'onde et ainsi d'en augmenter la fréquence. Cette transformation est réalisée grâce au changement d'échelle d'une fonction de base appelée *ondelette*. Grâce notamment à des collaborations avec Grossmann et Meyer, la théorie de la transformée en ondelettes continue fut établie au début des années 1980 [Grossmann 1984]. Le rapprochement avec les analyses harmoniques et le traitement d'image multi-échelle fut ensuite réalisé.

Ce chapitre introduit tout d'abord la transformée en ondelettes continue à une dimension, telle qu'elle a été définie pour le traitement du signal par Grossmann ou encore Meyer [Meyer 1992] pour compléter les études temps-fréquence effectuées par la transformée de Fourier. Nous verrons qu'un des intérêt principal des ondelettes réside dans la possibilité d'adapter leur fréquence (ou échelle) en fonction du signal à étudier ce qui a donné naissance aux analyses temps-échelle. Les bases d'ondelettes seront ensuite présentées, permettant ainsi de définir les espaces multirésolutions engendrés lors de la transformée en ondelettes d'un signal. Mallat montre dans [Mallat 1989] qu'une représentation complète d'un signal peut être obtenue à partir de la projection de celui-ci dans un nombre fini d'espaces d'approximations et de détails. Nous introduisons ensuite la transformée en ondelettes orthogonale mise au point par Mallat, ainsi que son extension à deux dimensions pour l'application au traitement d'image.
Nous verrons à cette occasion que le passage d'une à deux dimensions ne modifie pas les propriétés d'adaptation fréquentielle des ondelettes au contenu de l'image à étudier. Ce sont ces propriétés qui nous intéresseront par la suite. En effet grâce à celles-ci, nous adapterons l'échelle de la transformée en fonction des contours de l'image afin d'étudier leurs caractéristiques (et en particulier leur direction) dans un espace de résolution optimale.

1 Transformée en ondelettes continue

Une ondelette ψ est une fonction de moyenne nulle :

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0 \tag{2.1}$$

qui est décalée par un paramètre u et dilatée par un paramètre d'échelle s :

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right) \tag{2.2}$$

Une fonction d'ondelette doit également être d'énergie finie :

$$\int_{-\infty}^{+\infty} |\psi(t)|^2 \, dt < \infty \tag{2.3}$$

La transformée en ondelettes de f à l'échelle s et à la position u est calculée en corrélant f avec une ondelette :

$$Wf(u,s) = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{s}} \psi^{\star}\left(\frac{t-u}{s}\right) dt \tag{2.4}$$

où l'opérateur $\star$ représente le complexe conjugué.
De la même manière qu'une analyse de Fourier à fenêtre glissante (ou transformée de Fourier à court terme), une transformée en ondelettes permet de réaliser une étude temps-fréquence des composantes spectrales d'un signal temporel. La résolution temps-fréquence est fixe lors d'une analyse de Fourier puisque la fenêtre d'analyse est de taille constante. Ce n'est pas le cas lors d'une analyse par ondelettes puisque le paramètre s fait varier les composantes spectrales de ψ. En effet, quand l'échelle s diminue, le support temporel de ψ diminue mais son support fréquentiel augmente et est décalé vers les hautes fréquences. Soit $\hat{\psi}$ la transformée de Fourier de ψ. L'énergie de $\hat{\psi}$ est concentrée dans un intervalle fréquentiel centré en η. L'énergie de la transformée d'un atome $\hat{\psi}_{u,s}(\omega)$ est donc concentrée dans un intervalle fréquentiel centré sur $\frac{\eta}{s}$ et dont la taille varie en $\frac{1}{s}$. L'illustration de cette dualité est présentée en Figure 2.1, dans laquelle il est possible d'observer que lorsque le support temporel est réduit, la taille du support fréquentiel augmente.

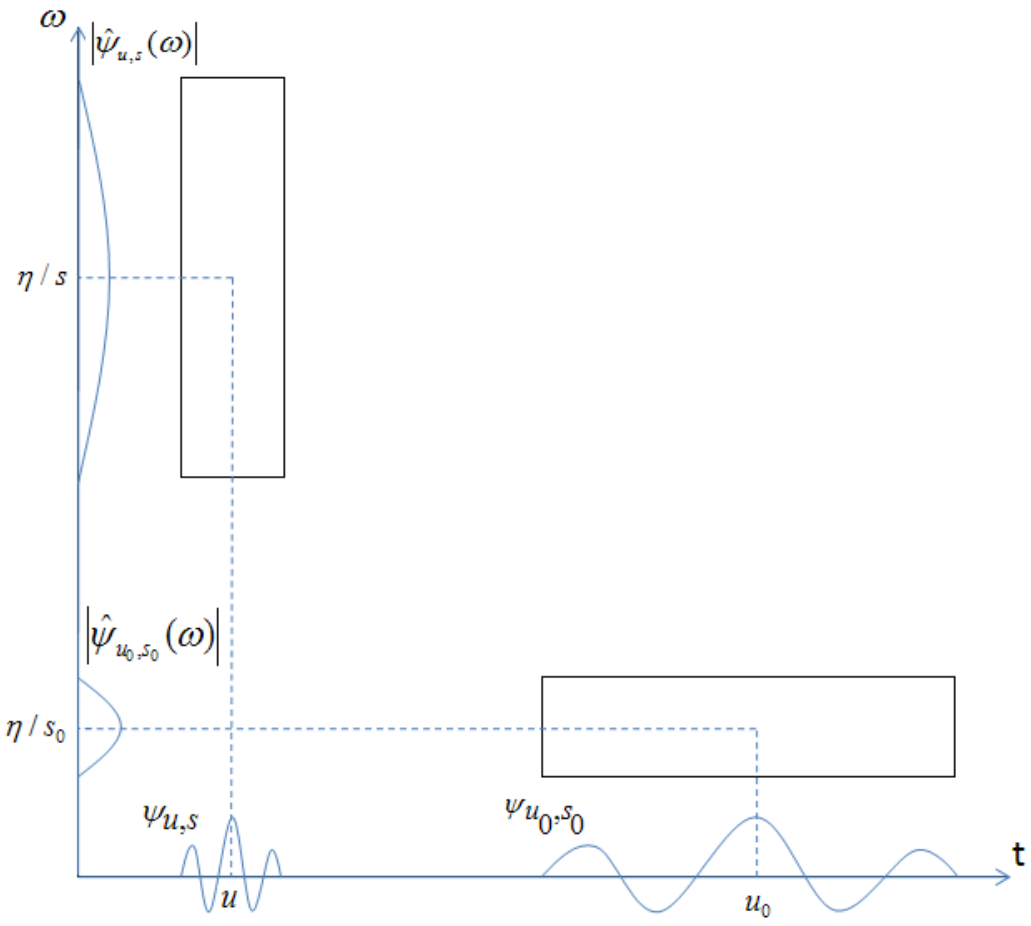
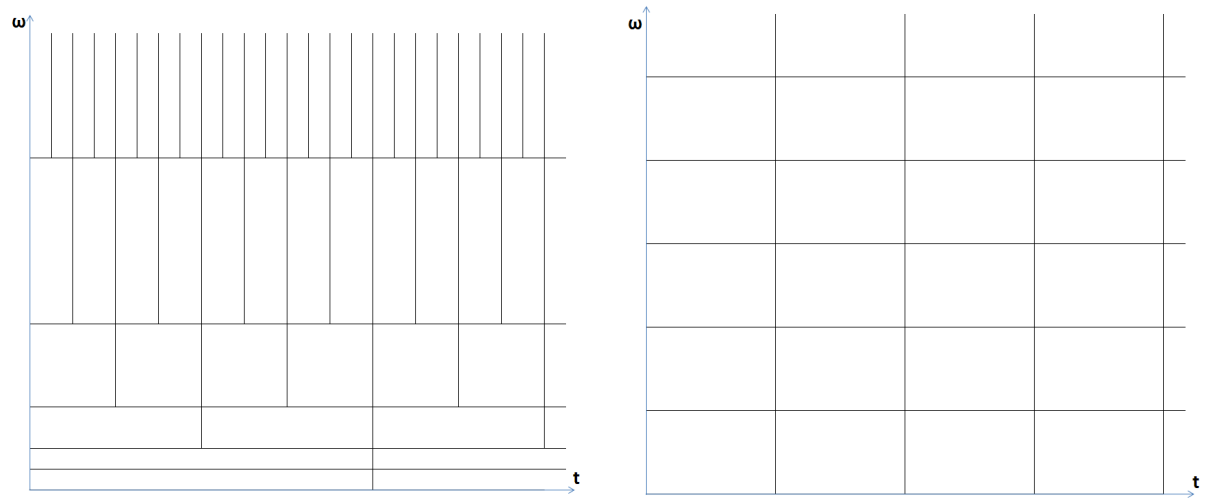


FIGURE 2.1 – Représentation des supports temps-fréquence de deux ondelettes $\psi_{u,s}$ et ψ_{u_0,s_0}.

Autrement dit, dans le plan temps-fréquence le support des atomes de la transformée de Fourier est fixe, alors que la représentation d'un atome de base d'une ondelette est un rectangle centré en $(u, \eta/s)$ et dont l'étendue temporelle et fréquentielle varient respectivement en s et $\frac{1}{s}$. Quand l'échelle varie, la hauteur et la largeur du rectangle change mais l'aire du carré reste identique. La différence entre les deux comportements est montrée dans la Figure 2.2.



(a) (b)

FIGURE 2.2 – Différence du pavage temps-fréquence entre la transformée en ondelettes (a) et la transformée de Fourier à court terme (b).

Le changement de forme du support selon l'échelle permet d'optimiser l'observation de signaux très basses fréquences ou à l'inverse très hautes fréquences.

2 Bases d'ondelettes

A partir de la définition donnée dans la section précédente, une base orthonormée de $\mathbf{L}^2(\mathbb{R})$ peut être construite à partir de la dilatation et du décalage d'une ondelette de référence (voir [Mallat 1999] pour les justifications théoriques).

$$\left\{ \psi_{j,n}(t) = \frac{1}{\sqrt{2^j}} \psi \left(\frac{t - 2^j n}{2^j} \right) \right\}_{(j,n)\in\mathbb{Z}^2} \tag{2.5}$$

Remarquons que dans cette formule les paramètres de décalage et d'échelle sont discrets. Cela signifie que le nombre de coefficients d'ondelettes obtenus après transformation sera dénombrable, ainsi que le nombre d'échelles sur lesquelles sera projeté le signal. Le fait de discrétiser le paramètre d'échelle par pas de 2^j engendre une transformée dite dyadique. Le choix de ce pas permet de simplifier les calculs, et d'avoir une résolution qui décroit d'un facteur deux à chaque échelle. Mallat et Meyer dans [Mallat 1989] et [Meyer 1992] montrent que cela n'empêche pas d'obtenir une représentation complète du signal.
Les ondelettes ainsi dilatées de 2^j donnent des indications sur les variations de résolution 2^{-j} du signal. La construction de telles bases peut donc mener à une étude multi-résolution d'un signal (voir partie 2.2.1). Nous verrons également que ces approximations peuvent être caractérisées grâce à des bancs de filtres discrets et que ces filtres permettent de réaliser de manière simple et rapide des transformées en ondelettes orthogonales discrètes.

2.1 Approximations multi-résolutions

Les approximations multi-résolutions consistent à calculer des approximations du signal à différentes résolutions grâce à des projections sur des espaces $\{\mathbf{V}_j\}_{j\in\mathbb{Z}}$. En traitement d'image en particulier, il peut être intéressant d'adapter les traitement en fonction des caractéristiques des éléments de l'image (contours par exemple). Burt et Adelson ont notamment étudié des procédés d'adaptation de résolution en fonction des caractéristiques locales ([Burt 1983]), grâce à une pyramide multi-résolution.
L'approximation d'une fonction f à la résolution 2^{-j} est donnée par les moyennes locales de f calculées sur des voisinages dont la taille est proportionnelle à l'échelle 2^j. Le moyennage est dû à la discrétisation du paramètre de décalage. Une approximation multi-résolution est donc composée d'une série d'approximation dont les échelles varient pour donner une représentation globale du signal. Plus concrètement, les approximations à la résolution 2^{-j} sont définies comme des projections orthogonales dans l'espace $\{\mathbf{V}_j\}_{j\in\mathbb{Z}}$. Cet espace comprend toutes les approximations de f à la résolution 2^{-j}. Les propriétés des espaces d'approximation sont listées dans [Mallat 1999] et [Meyer 1992].

2.2 Fonction d'échelle

Dans [Mallat 1999], il est montré que la projection orthogonale de f dans chaque espace $\mathbf{V}_j$ peut se faire grâce à une fonction unique Φ appelée fonction d'échelle. Pour que la famille $\{\Phi_{j,n}\}_{n\in\mathbf{Z}}$ forme une base orthonormée de $\mathbf{V}_j$, il faut que Φ soit de la forme :

$$\Phi_{j,n}(t) = \frac{1}{\sqrt{2^j}}\Phi\left(\frac{t-n}{2^j}\right) \tag{2.6}$$

Un exemple de fonction d'échelle est donnée en Figure 2.3. La projection de f dans $\mathbf{V}_j$ se fait

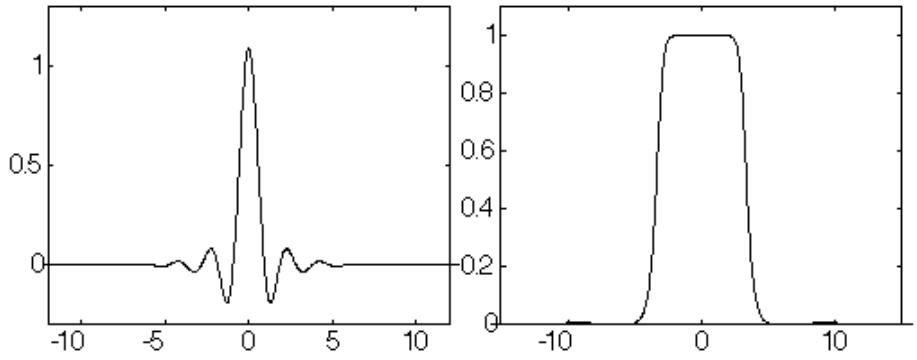


FIGURE 2.3 – Fonction d'échelle spline (gauche) et transformée de Fourier (droite).

donc ainsi :

$$P_{V_j} f = \sum_{-\infty}^{+\infty} \langle f, \Phi_{j,n}\rangle \, \Phi_{j,n} \tag{2.7}$$

Les coefficients d'approximations appelés $a_j[n]$, définis comme tels :

$$a_j[n] = \langle f, \Phi_{j,n}\rangle \tag{2.8}$$

donnent une approximation discrète de f à l'échelle 2^j. Notons que la fonction d'échelle se comporte comme un filtre passe-bas appliqué à f et échantillonné à des intervalles proportionnels à 2^j. Les différentes échelles de l'approximation multi-résolutions sont créées par dilatation de la fonction d'échelle.

2.3 Filtres miroirs conjugués

Dans l'optique d'améliorer l'implémentation de la transformée en ondelettes, l'approche par bancs de filtres a été étudiée. Il a donc été prouvé que toute fonction d'échelle peut être définie par un filtre discret appelé filtre miroir conjugué. On peut montrer que comme les résolutions successives sont imbriquées ($\mathbf{V}_j \subset \mathbf{V}_{j-1}$), il existe nécessairement une suite de réels $h[n]$ telle que :

$$\frac{1}{\sqrt{2}}\Phi(\frac{t}{2}) = \sum_{-\infty}^{+\infty} h[n]\Phi(t-n) \tag{2.9}$$

avec

$$h[n] = \left\langle \frac{1}{\sqrt{2}}\Phi(\frac{t}{2}), \Phi(t-n) \right\rangle \tag{2.10}$$

Ce filtre définit donc entièrement la fonction d'échelle. Mallat et Meyer ont prouvé que si $\Phi \in \mathbf{L}^2$ est une fonction d'échelle intégrable, alors la transformée de Fourier de $h[n]$ tel qu'il est défini dans l'équation 2.10 satisfait :

$$\forall \omega \in \mathbb{R}, \left|\hat{h}(\omega)\right|^2 + \left|\hat{h}(\omega+\pi)\right|^2 = 2 \tag{2.11}$$

L'équation 2.11 justifie le nom de filtre miroir conjugué. Ainsi, les coefficients d'approximation $a_1[n]$ de l'espace $\mathbf{V}_1$ s'obtiennent à partir des coefficients d'approximation $a_0[n]$ de l'espace $\mathbf{V}_0$ par convolution avec le filtre sous-échantillonné $\bar{h}[n]$, avec $\bar{h}[n] = h[-n]$.

$$a_1[n] = a_0 \star \bar{h}[2n] \tag{2.12}$$

2.4 Ondelettes orthogonales

Les sections précédentes ont permis de définir les approximations d'une fonction f à différents niveaux de résolution grâce aux projections sur les espaces $\mathbf{V}_j$. Cette projection pouvant être réalisée soit grâce à la fonction d'échelle soit par filtrage grâce au filtre discret $h[n]$. D'autre part, nous savons que $\mathbf{V}_j$ est inclu dans l'espace $\mathbf{V}_{j-1}$. Soit $\mathbf{W}_j$ le complément orthogonal de $\mathbf{V}_j$ dans $\mathbf{V}_{j-1}$.

$$\mathbf{V}_{j-1} = \mathbf{V}_j \oplus \mathbf{W}_j \tag{2.13}$$

La projection orthogonale de f dans $\mathbf{V}_{j-1}$ peut être décomposée comme la somme des projections dans $\mathbf{V}_j$ et $\mathbf{W}_j$:

$$P_{V_{j-1}} f = P_{V_j} f + P_{W_j} f \tag{2.14}$$

Par opposition aux approximations, la projection de f sur l'espace $\mathbf{W}_j$ est appelée *détails* de f. De la même manière que pour la projection sur les approximations, Mallat et Meyer montrent qu'il est possible de projeter une fonction f dans un espace $\mathbf{W}_j$ grâce à une fonction appelée fonction d'*ondelette* ψ. Pour que la famille $\{\psi_{j,n}\}_{n\in\mathbf{Z}}$ forme une base orthonormée de $\mathbf{W}_j$, il faut que la fonction d'ondelette soit de la forme :

$$\psi_{j,n}(t) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{t-2^j n}{2^j}\right) \tag{2.15}$$

On peut de la même manière que pour la fonction d'échelle en déduire un filtre discret $g[n]$ qui définit ψ entièrement. Ce filtre peut s'exprimer en fonction du filtre $h[n]$ de la manière suivante :

$$g[n] = (-1)^{1-n} h[1-n] \tag{2.16}$$

La transformée de Fourier de ψ montrée en Figure 2.4 montre que la fonction d'ondelette se comporte comme un filtre passe-bande complémentaire au filtre passe-bas engendré par la fonction d'échelle. Le filtre discret $g[n]$ permet donc de créer les coefficients de détails de la fonction f,

appelés $d_j[n]$. La création des détails $d_1[n]$ s'exprime donc à partir des coefficients d'approximation de la résolution inférieure et du filtre $\bar{g}[n]$ sous-échantillonné ($\bar{g}[n] = g[-n]$) :

$$d_1[n] = a_0 \star \bar{g}[2n] \tag{2.17}$$

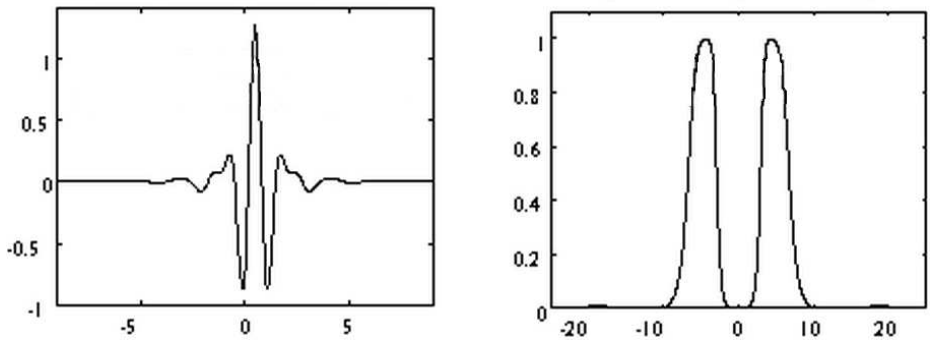


FIGURE 2.4 – Fonction d'ondelette spline (gauche) et sa transformée de Fourier (droite).

2.5 Transformée en ondelette rapide orthogonale

La transformée en ondelette rapide est réalisée grâce aux filtres présentés plus haut par décompositions successives des coefficients d'approximation en des coefficients d'approximation et des coefficients de détails de résolutions inférieures. La transformée consiste en une cascade de convolutions discrètes et de sous-échantillonage. La décomposition est donc effectuée comme suit :

$$a_{j+1}[n] = \sum_{p=-\infty}^{+\infty} h[p-2n]a_j[p] = a_j \star \bar{h}[2n] \tag{2.18}$$

$$d_{j+1}[n] = \sum_{p=-\infty}^{+\infty} g[p-2n]a_j[p] = a_j \star \bar{g}[2n] \tag{2.19}$$

Une reconstruction parfaite peut être réalisée à partir des approximations et détails de la résolution inférieure par convolution avec les coefficients sur-échantillonnés. Le sur-échantillonnage est réalisé par insertion de zéros entre chaque coefficients. Soit $\check{h}[p] = h[n]$ si $p = 2n$ et 0 si $p = 2n+1$.

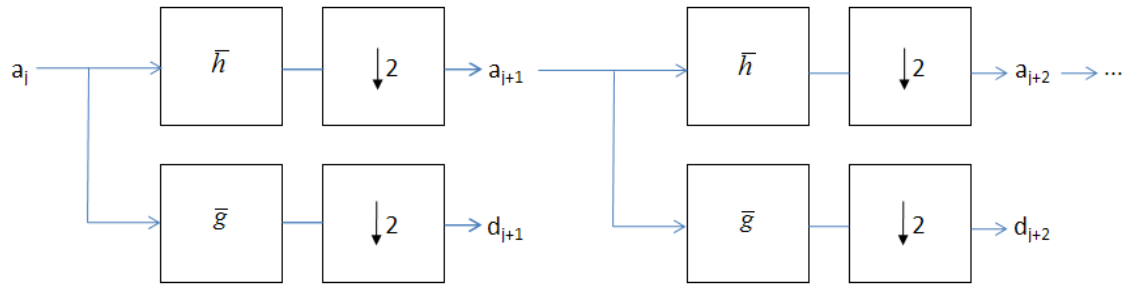
$$a_j[n] = \sum_{p=-\infty}^{+\infty} h[n-2p]a_{j+1}[p] + \sum_{p=-\infty}^{+\infty} g[n-2p]d_{j+1}[p] = \check{a}_{j+1} \star h[n] + \check{d}_{j+1} \star g[n] \tag{2.20}$$

Cette transformée peut être synthétisée par le schéma de la Figure 2.5.

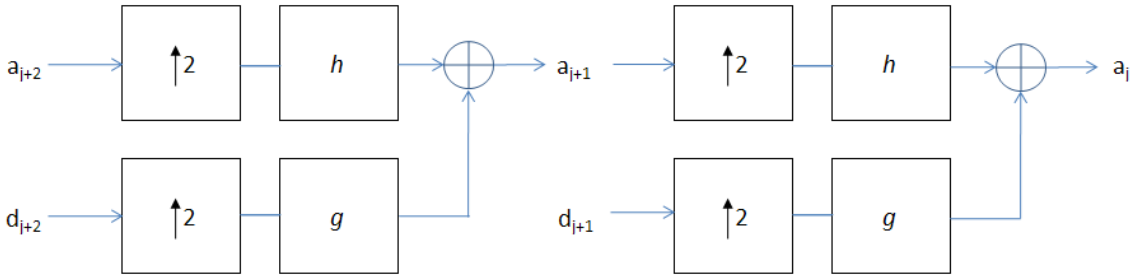
3 Transformée en ondelettes discrète 2D

3.1 Cadre d'approximation

La transformée en ondelettes discrètes 2D peut se déduire directement de la transformée 1D présentée dans la section précédente. L'approche la plus intuitive consiste à interpréter la fonction d'échelle 2D $\Phi(x, y)$ comme un produit des deux fonctions d'échelle 1D : $\Phi(x)\Phi(y)$ qui représente



(a)



(b)

FIGURE 2.5 – (a) Schéma de la transformée en ondelettes rapide par cascade de filtres et sous-échantillonage. (b) Schéma de reconstruction (transformée inverse rapide) par cascade de filtres et sur-échantillonage des coefficients.

l'espace d'approximation $\mathbf{V}_j^2$. Mallat montre que pour représenter une base orthonormée de l'espace $\mathbf{W}_j^2$, la famille d'ondelette doit être composée de trois ondelettes définies telles que :

$$\begin{aligned} \psi^1(x,y) &= \Phi(x)\psi(y) \\ \psi^2(x,y) &= \psi(x)\Phi(y) \\ \psi^3(x,y) &= \psi(x)\psi(y) \end{aligned} \tag{2.21}$$

Lorsque la fonction transformée est une image, la fonction d'ondelette ψ^1 permet d'obtenir les détails horizontaux, ψ^2 les détails verticaux et ψ^3 les détails diagonaux.
En appliquant ce principe et en utilisant les filtres associés aux fonctions d'échelle ($h[n]$) et d'ondelette ($g[n]$), on peut traduire la transformée de la manière suivante :

$$\begin{aligned} a_{j+1}[n] &= a_j \star \bar{h}\bar{h}[2n] \\ d^1_{j+1}[n] &= a_j \star \bar{h}\bar{g}[2n] \\ d^2_{j+1}[n] &= a_j \star \bar{g}\bar{h}[2n] \\ d^3_{j+1}[n] &= a_j \star \bar{g}\bar{g}[2n] \end{aligned} \tag{2.22}$$

Notons que les filtres utilisés sont séparables, ce qui permet de réaliser le même filtrage successivement sur les deux dimensions de la fonction transformée. De la même manière que la transformée 1D, la reconstruction parfaite est possible grâce à la formule :

$$a_j[n] = \check{a}_{j+1} \star hh[n] + \check{d}^1_{j+1} \star hg[n] + \check{d}^2_{j+1} \star gh[n] + \check{d}^3_{j+1} \star gg[n] \tag{2.23}$$

3.2 Application aux images

La transformée discrète orthogonale rapide 2D est la plus utilisée dans les différents traitements d'images utilisant les ondelettes. Les coefficients d'approximation et de détails sont souvent

disposés comme montré dans la Figure 2.6. Cette disposition provient de la décomposition pyrami-

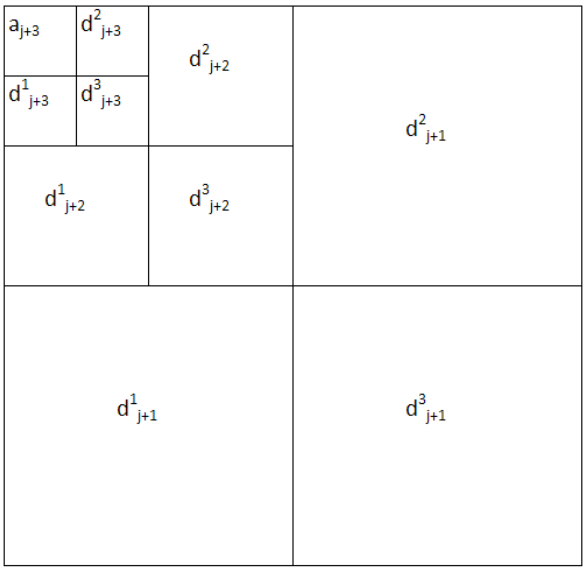


FIGURE 2.6 – Disposition usuelle des coefficients de la transformée en ondelettes discrète rapide 2D.

dale effectuée pour transformer l'image. Le schéma de la Figure 2.7 illustre ce principe. Le premier filtrage effectué sur les lignes génère deux images Figure 2.7(b). La première image, L correspond à la sortie du filtre passe-bas $h[n]$ dont les colonnes ont été sous-échantillonnées. La deuxième image H, correspond à la sortie du filtre $g[n]$ dont les colonnes ont été sous-échantillonnées. Les deux filtres sont ensuite appliqués sur les colonnes de ces images pour obtenir la Figure 2.7(c) après sous-échantillonnage des lignes. Dans cette image, nous pouvons faire la correspondance avec les notations du schéma de la Figure 2.6 :

- L'image filtrée deux fois passe-bas LL correspond à l'image d'approximation a_{j+1}
- L'image filtrée passe-bas puis passe-haut LH correspond à l'image de détails horizontaux d^1_{j+1}
- L'image filtrée passe-haut puis passe-bas HL correspond à l'image de détails verticaux d^2_{j+1}
- L'image filtrée deux fois passe-haut HH correspond à l'image de détails diagonaux d^3_{j+1}

La même série de filtrage est ensuite itérée sur les lignes puis les colonnes de l'image d'approximation dans la Figure 2.7(d-e) pour obtenir les images de résolution inférieures. Un exemple de transformée en ondelettes à deux niveaux de résolution d'une image réelle (Lena) est présentée en Figure 2.8. Dans cet exemple, il est possible de voir que l'image d^1_{j+1} (en bas à gauche) contient principalement les détails horizontaux (sourcils et bouche), alors que l'image d^2_{j+1} (en haut à droite) contient majoritairement les détails verticaux (cheveux et épaule). Enfin, l'image d^3_{j+1} qui contient les détails diagonaux fait ressortir les bords obliques du chapeau. La transformée réalisée ensuite sur l'image d'approximation génère des images de détails sous-échantillonnées permettant ainsi de mettre en valeur les détails de l'image à une résolution inférieure. L'étude multirésolution peut se poursuivre jusqu'au niveau de résolution requis par l'utilisateur en vue d'un futur traitement ou analyse de l'image. Notons également que cette transformée offre des propriétés de reconstruction exacte lorsque les coefficients de détails ou d'approximation de sont pas modifiés. Les deux applications principales de la transformée en ondelettes en traitement d'image sont la compression et du débruitage.

Si la transformée en elle même ne constitue pas une compression, la répartition fréquentielle qu'elle engendre rend la quantification des coefficients de la transformée plus efficace. En effet, la majeure

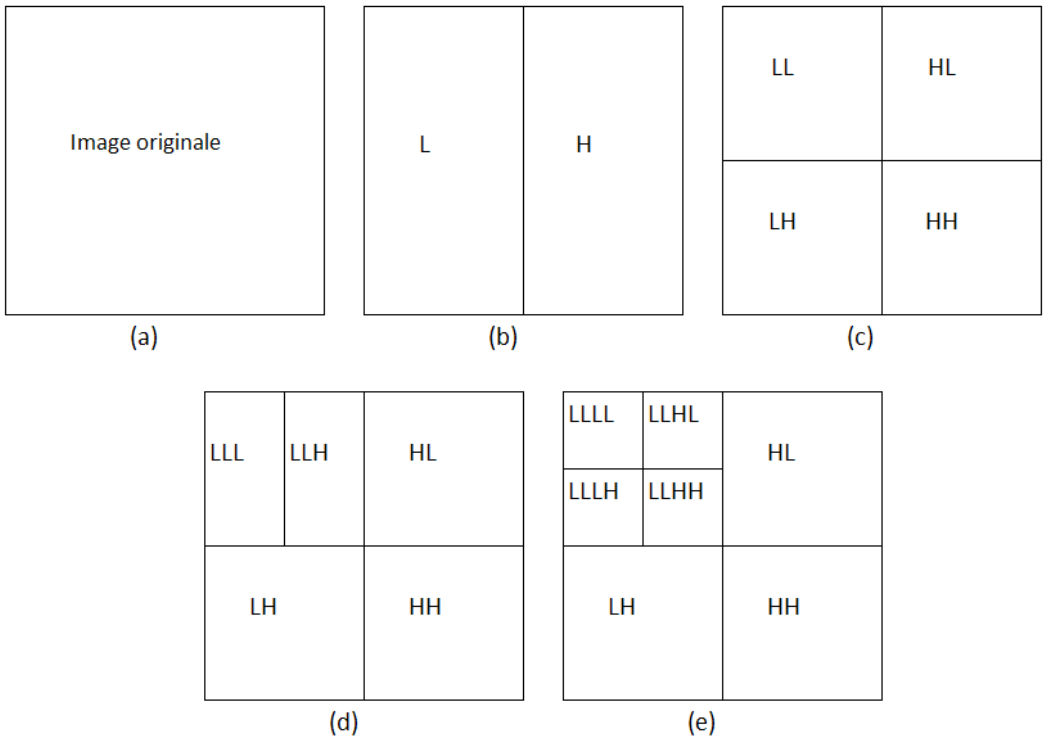


FIGURE 2.7 – Schéma pyramidal de la transformée en ondelettes 2D rapide.

partie de l'information est concentrée dans les images d'approximation, alors que les images de détails ont une entropie beaucoup plus faible. La quantification effectuée sur les coefficients tient donc compte de cette répartition de l'entropie en quantifiant de manière beaucoup plus grossières les coefficients de détails. De telles méthodes de compression basées sur les ondelettes peuvent être trouvées dans : [Grgic 2001], [Yeung 1997], [Albanesi 1997], [Villasenor 1995], [Taubman 1994] et [Mallat 1989].
La transformée en ondelettes est également intéressante pour le débruitage. En effet, le bruit présent dans les images apparaissent principalement dans les images de détails aux résolutions les plus hautes. La quantification ou le seuillage de ces coefficients permet alors de ne traiter que les coefficients susceptibles de contenir le bruit, tout en laissant intacts les autres images de la transformée. Visuellement, ce traitement spécifique améliore souvent le rendu visuel du débruitage par rapport aux techniques classiques. Quelques exemples de débruitage par ondelettes peuvent être trouvés dans : [Luisier 2007], [Rahman 2008] [Donoho 1995], [Donoho 1994] et [Chang 2000].

4 Conclusion du chapitre

Cette introduction aux ondelettes a permi de présenter la transformée classique telle qu'elle a été créée pour le traitement des ondes sismiques. Nous avons présenté ses avantages par rapport aux analyses temps-fréquences réalisées avec la transformée de Fourier à fenêtre glissante, grâce à l'adaptation de la fréquence de l'ondelette. Ainsi un compromis optimal peut être trouvé entre précision spatiale et précision fréquentielle lors de l'analyse d'un signal.

Par la suite, nous avons introduit les bases d'ondelettes permettant de définir le cadre nécessaire aux analyses multirésolution. Ces analyses autorisent une représentation complète d'un signal réparti dans un nombre fini d'espaces complémentaires appelés espaces d'approximation et de détails. La projection dans ces espaces se fait grâce des fonctions d'échelles et d'ondelettes qui

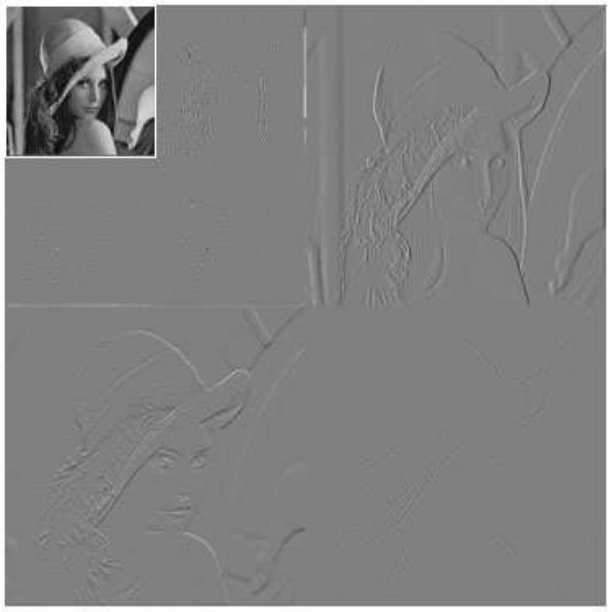


FIGURE 2.8 – Transformée en ondelettes à deux niveaux de résolution de l'image Lena.

projettent respectivement le signal dans les espaces d'approximation et de détails. En pratique, les projections sont réalisées par des filtres discrets communément appelés h et g. La séparabilité de ces filtres entraîne une correspondance directe de la transformée à une dimension aux signaux à deux dimensions, permettant ainsi l'application au traitement d'image.

La projection successives d'une image dans les espaces d'approximations et de détails à des résolutions décroissantes permet d'analyser ses composantes à l'échelle souhaitée. Encore une fois le gain par rapport à la transformée de Fourier est important puisque cette dernière permettait de réaliser l'analyse fréquentielle d'une image en perdant l'information spatiale, alors que les coefficients d'ondelettes autorisent une localisation à la fois spatiale et fréquentielle des composantes de l'image. La transformée en ondelettes est donc un outil d'analyse puissant en particulier pour étudier la répartition fréquentielle des composantes d'une image.

Du fait de ces propriétés nous utiliserons la transformée en ondelettes dans notre étude pour analyser en particulier la répartition fréquentielle des contours d'une image dans le but d'optimiser l'estimation de leur direction. Nous chercherons donc à trouver la résolution qui permet de placer les contours dans un espace de résolution optimale, facilitant ainsi leur analyse. Le critère de choix de résolution optimale sera explicité en partie 5.5, ainsi que l'implémentation concrète de cet algorithme.

Chapitre 3

Méthodes d'interpolation classiques

Le dernier chapitre de cet état de l'art s'intéresse aux méthode de référence d'interpolation d'image. L'interpolation d'image consiste à augmenter le nombre de pixels qui composent une image pour la répartir sur une grille discrète uniforme plus dense. Les pixels manquants doivent être interpolés de façon à reproduire au mieux les caractéristiques de l'image basse résolution, sans générer d'artefacts. Les méthodes présentées ici sont encore largement utilisées dans un grand nombre d'applications du fait de leur simplicité algorithmique et donc leur faible coût d'implémentation. Si ces méthodes parviennent relativement bien à interpoler les zones de l'image contenant de faibles variations, nous verrons que de nombreux artefacts sont introduits en particulier au niveau des contours des objets de l'image. Bien que la proportion des pixels appartenant à des contours soit relativement faible, l'impact visuel de la dégradation de ces éléments est très important. En effet, l'étude des cartes de saillance montre que la qualité d'une image est fortement corrélée avec la qualité de ses contours. C'est pourquoi des approches qui s'adaptent aux contours sont nécessaires, afin d'optimiser le rendu visuel des images interpolées.

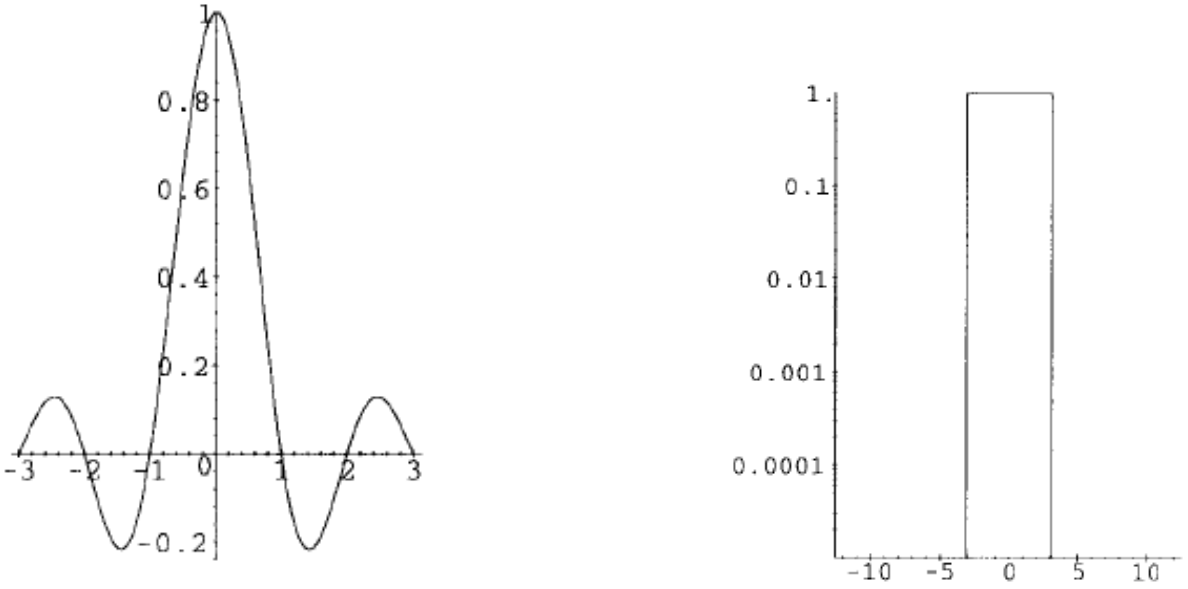
Dans cette partie, nous allons comparer cinq méthodes d'interpolation largement utilisées dans la littérature et expliciter leurs avantages et inconvénients. Par soucis de simplicité, seuls des facteurs d'interpolation entiers seront illustrés mais la plupart des techniques sont applicables à des facteurs non-entiers. Les résultats obtenus avec certaines de ces interpolations seront utilisés comme référence lors de la comparaison avec notre méthode présentée en dernière partie de ce manuscript.

1 Interpolation idéale

D'après le théorème de l'échantillonnage de Shannon-Nyquist, une fonction $f(x)$ peut être reconstruite de façon exacte à partir de ses échantillons $f(x_k) = f(kx)$ par une interpolation utilisant le sinus cardinal, si la fréquence d'échantillonnage $f_s = 1/T$ est au dessus de la borne de Nyquist, qui correspond à deux fois la fréquence maximale du signal d'entrée. La formule d'interpolation suivante permet la reconstruction d'une fonction continue $g(x)$ à partir des échantillons de $f(x)$.

$$g(x) = \sum_{k=-\infty}^{+\infty} f(x_k).sinc\left(\frac{x - x_k}{T}\right) \tag{3.1}$$

La représentation spatiale et fréquentielle du noyau de cette interpolation est montrée en Figure 3.1. Remarquons que l'interpolateur parfait correspond à un filtre passe-bas idéal. Cette formule effectue une somme sur une infinité de coefficients. Pour rendre cette interpolation réalisable, une approximation du sinus cardinal est nécessaire sur un nombre fini de coefficients. Les interpolations suivantes sont donc toutes des approximations plus ou moins précises du sinus cardinal idéal.



(a) (b)

FIGURE 3.1 – Interpolation cubique spline cardinale. (a) Noyau de l'interpolation sur l'intervalle $[-3, 3]$. (b) Tracé logarithmique du module de la transformée de Fourier du noyau.

2 Méthode du plus proche voisin

La méthode du plus proche voisin est la plus simple qui soit. Elle consiste à associer au point x la valeur de l'échantillon le plus proche. Le noyau de l'interpolation (approximation du sinus cardinal) est donc le suivant [Lehmann 1999] :

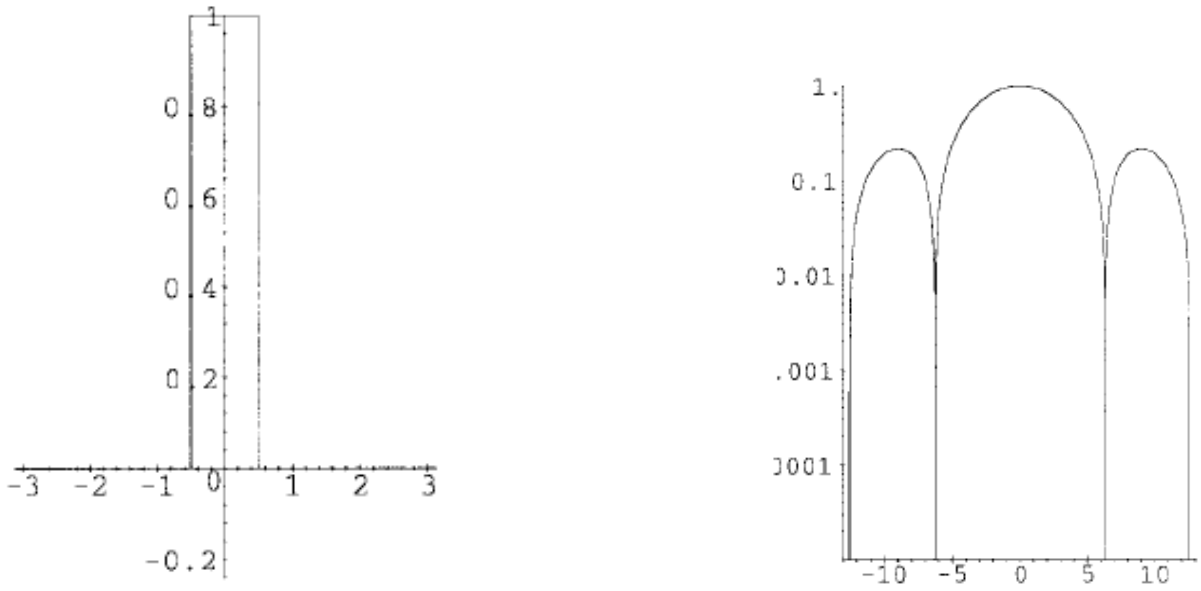
$$h(x) = \begin{cases} 1 & \text{pour} \quad 0 \leq |x| < \frac{1}{2} \\ 0 & \text{sinon} \end{cases} \tag{3.2}$$

La formule d'interpolation suit donc le même schéma que l'équation 3.1, mais le noyau ainsi que le nombre de coefficients sur lequel $g(x)$ n'est pas nul changent.

$$\begin{aligned} g(x) &= \sum_{k=-\infty}^{+\infty} f(x_k).h(x - x_k) \\ g(x) &= f(\lfloor x \rfloor) \end{aligned} \tag{3.3}$$

avec $\lfloor x \rfloor$ la partie entière de x. Le noyau ainsi que sa transformée de Fourier sont représentés dans la Figure 3.2.

Le passage à deux dimensions est direct puisque $h(x)$ est séparable. Les lignes puis les colonnes de l'image sont tour à tour interpolées grâce à $g(x)$. Les images interpolées ont donc la forme de l'exemple de la Figure 3.3.
Le principal avantage de cette méthode est qu'elle est très simple à implémenter puisqu'un seul échantillon est nécessaire pour remplir l'intervalle entre deux échantillons distincts. Cependant, comme aucun nouveau pixel n'est créé, l'impression d'avoir une image de résolution supérieure est nulle. Cette interpolation donne l'impression d'avoir grossi les pixels d'origine. Ce phénomène est d'autant plus visible et gênant pour des grands facteurs d'interpolation. Les faibles performances de cette interpolation peuvent être expliquées par le comportement médiocre du filtre passe-bas de la Figure 3.2(b).



(a) (b)

FIGURE 3.2 – Interpolation par la méthode du plus proche voisin. (a) Noyau de l'interpolation. (b) Tracé logarithmique du module de la transformée de Fourier du noyau.

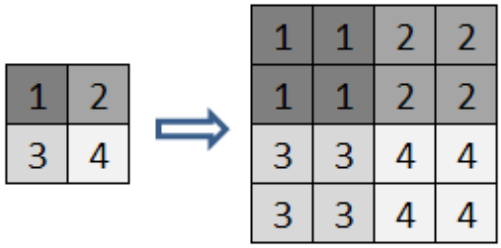


FIGURE 3.3 – Schéma de principe de l'interpolation par réplication de pixel.

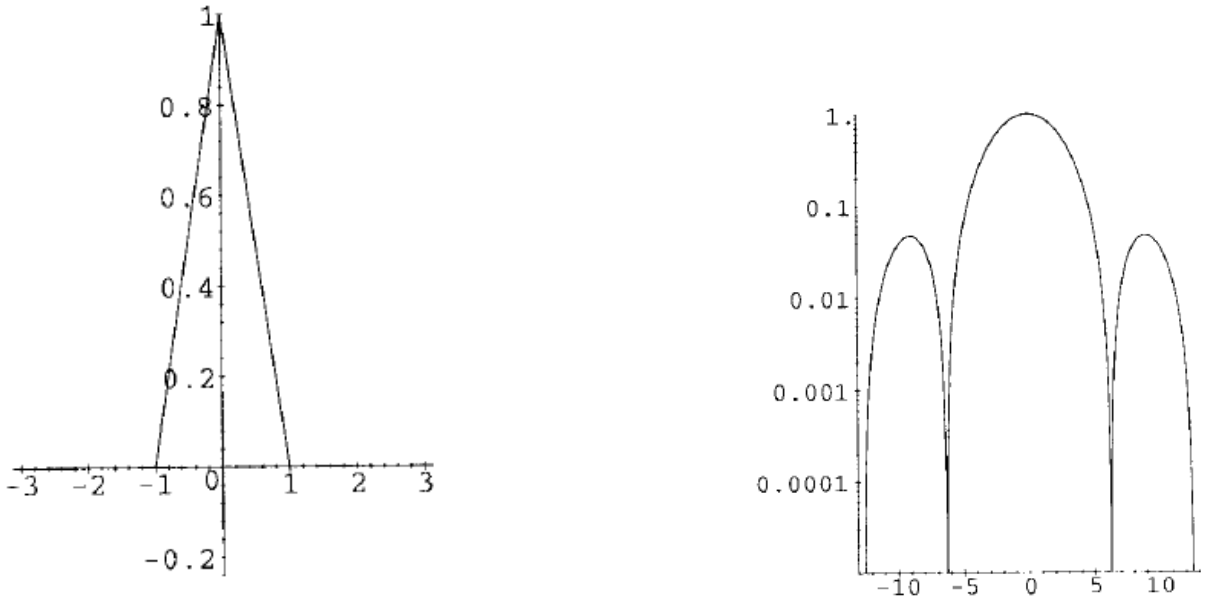
3 Interpolation bilinéaire

Le noyau de l'interpolation bilinéaire est une opération de moyenne entre deux échantillons successifs. Le noyau de cette interpolation, noté $h(x)$ est tel que :

$$h(x) = \begin{cases} 1 - |x| & \text{pour} \quad 0 \leq |x| < 1 \\ 0 & \text{sinon} \end{cases} \tag{3.4}$$

Le noyau ainsi que sa transformée de Fourier sont représentés dans la Figure 3.4.

Pour une interpolation à une dimension, deux pixels sont nécessaires pour interpoler les valeurs de $g(x)$ entre deux échantillons. Le passage à deux dimensions se fait par interpolation successive des lignes puis des colonnes de l'image. Les performances moyennes du filtre passe-bas montré dans la Figure 3.4(b) entraînent l'apparition de flou et d'aliasing dans les images interpolées. La section 3.7 dresse un bilan global de toutes les interpolations en illustrant des exemples réels.



(a) (b)

FIGURE 3.4 – Interpolation linéaire. (a) Noyau de l'interpolation. (b) Tracé logarithmique du module de la transformée de Fourier du noyau.

4 Interpolation bicubique

Cette technique a été notamment étudiée par Keys dans [Keys 1981]. Il définit par morceaux le noyau de l'interpolation cubique sur les intervalles $]-2,-1[$, $]-1,0[$, $]0,1[$ et $]1,2[$ de la manière suivante :

$$P(x) = \begin{cases} A_1|x|^3 + B_1|x|^2 + C_1|x| + D_1 & \text{pour} \quad 0 < |x| < 1 \\ A_2|x|^3 + B_2|x|^2 + C_2|x| + D_2 & \text{pour} \quad 1 < |x| < 2 \\ 0 & \text{pour} \quad |x| = 1 \\ 1 & \text{pour} \quad |x| = 0 \\ 0 & \text{pour} \quad |x| \geq 2 \end{cases} \tag{3.5}$$

Le noyau P de l'interpolation cubique doit être de classe $\mathcal{C}^1(\mathbb{R})$. C'est à dire que P est dérivable (et donc continue) et que sa dérivée P' est continue. Ceci entraîne les propriétés suivantes :

$$\begin{aligned} P(0) &= D_1 = 1 \\ P(1^-) &= A_1 + B_1 + C_1 + D_1 = 0 \\ P(1^+) &= A_2 + B_2 + C_2 + D_2 = 0 \\ P(2^-) &= 8A_2 + 4B_2 + 2C_2 + D_2 = 0 \\ P'(0^-) &= P'(0^+) = -C_1 = C_1 \\ P'(1^-) &= P'(1^+) = 3A_1 + 2B_1 + C_1 = 3A_2 + 2B_2 + C_2 \\ P'(2^-) &= P'(2^+) = 0 = 12A_2 + 4B_2 + C_2 \end{aligned} \tag{3.6}$$

Ce système à sept équations pour huit inconnues permet d'exprimer le noyau d'interpolation en fonction d'un seul paramètre appelé a. On a alors :

$$P(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1 & \text{pour} \quad 0 < |x| < 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a & \text{pour} \quad 1 < |x| < 2 \\ 0 & \text{pour} \quad |x| \geq 2 \end{cases} \tag{3.7}$$

Supposons que l'on veuille interpoler la fonction $f(x_k)$ dans l'intervalle défini par deux échantillons consécutifs x_j et x_{j+1}. Soit x la position du point à interpoler. La fonction d'interpolation cubique $g(x)$ est définie telle que $g(x_k) = f(x_k)$; c'est-à-dire que la fonction d'interpolation passe par les échantillons du signal d'entrée. Comme le noyau $P(x)$ est non nul seulement sur l'intervalle $]-2, 2[$ et qu'il s'annule aux points d'échantillonnage x_j, la fonction d'interpolation peut s'exprimer comme une combinaison linéaire de quatre paramètres : c_{k-1}, c_k, c_{k+1} et c_{k+2}. Soit s la position relative de x dans l'intervalle $[x_k, x_{k+1}]$ telle que $s = \frac{x-x_k}{h}$ où h est le pas d'échantillonnage.

$$g(x) = c_{k-1}P(s+1) + c_kP(s) + c_{k+1}P(s-1) + c_{k+2}P(s-2) \tag{3.8}$$

Le fait que le noyau s'annule aux points d'échantillonnage implique que les coefficients de g sont égaux aux échantillons d'entrée $c_j = f(x_j)$. En développant $g(x)$ d'après l'équation 3.7, on obtient donc :

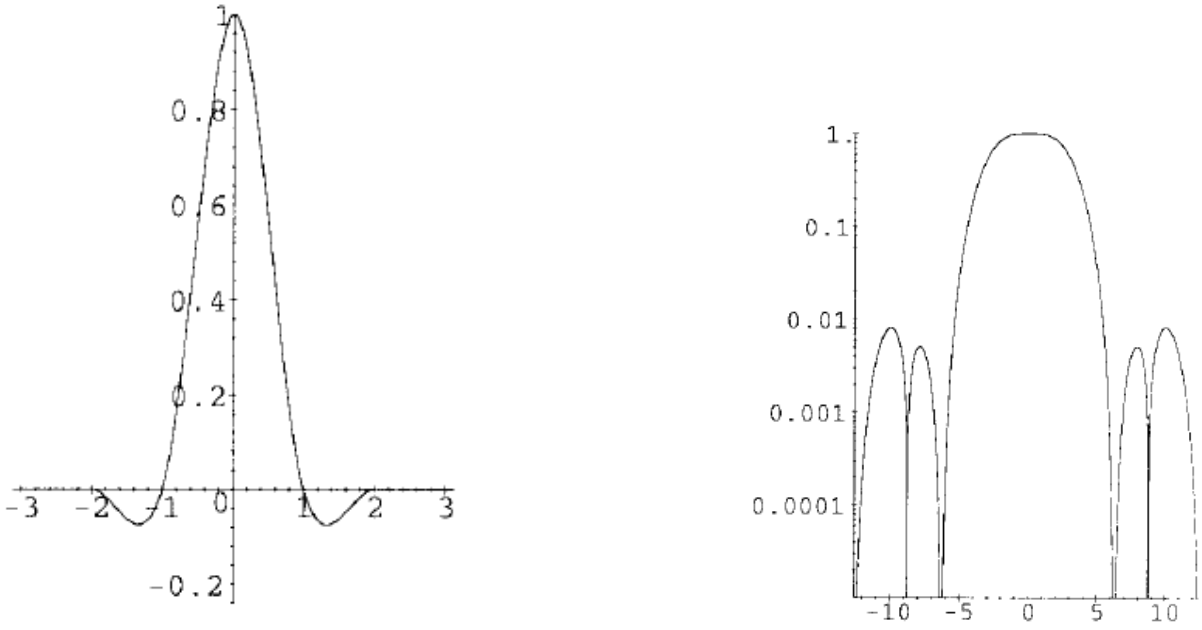
$$\begin{aligned} g(x) = &- [a(f(x_{k+2}) - f(x_{k-1})) + (a+2)(f(x_{k+1}) - f(x_k))]s^3 + \\ &[2a(f(x_{k+1}) - f(x_{k-1})) + 3(f(x_{k+1}) - f(x_k)) + a(f(x_{k+2}) - f(x_k))]s^2 - \\ &a(f(x_{k+1}) - f(x_{k-1}))s + f(x_k) \end{aligned} \tag{3.9}$$

Keys montre que le paramètre a fixé à $-1/2$ offre un taux de décroissance de l'erreur d'interpolation optimal. En effet, il montre que dans ce cas précis, l'erreur d'interpolation décroit de façon proportionnelle à une fonction cubique : $f(x) - g(x) = 0(h^3)$. La fonction d'interpolation est donc une approximation du troisième degré de la fonction f, qui permet de reconstruire de manière exacte toute fonction f du second degré. Par comparaison, notons que l'interpolation linéaire ne pourra reconstruire exactement que des fonctions de degré un, et que l'interpolation par réplication de pixel ne pourra reconstruire exactement uniquement des fonctions constantes. En prenant $a = -1/2$, la fonction d'interpolation sur l'intervalle $[0, 1]$ devient alors :

$$\begin{aligned} f(x) = &\left(-\frac{1}{2}f(-1) + \frac{3}{2}f(0) - \frac{3}{2}f(1) + \frac{1}{2}f(2)\right)x^3 + \left(f(-1) - \frac{5}{2}f(0) + 2f(1) - \frac{1}{2}f(2)\right)x^2 + \\ &\left(-\frac{1}{2}f(-1) + \frac{1}{2}f(1)\right)x + f(0) \end{aligned} \tag{3.10}$$

Cette fonction est représentée en Figure 3.5.

Le passage à la dimension deux est assez direct du fait de la séparabilité de la fonction d'interpolation ([Burge 2009]). Alors qu'à une dimension, l'interpolation de l'intervalle $[0, 1]$ nécessite quatre points connus, l'interpolation de l'intervalle $[0, 1] \times [0, 1]$ nécessite seize points. Concrètement, l'interpolation se fait d'abord sur les lignes de l'image puis sur les colonnes grâce à la formule générale de l'interpolation à une dimension de l'équation 3.9. Pour plus de rapidité, les valeurs des seize paramètres peuvent être calculées à l'avance comme nous l'avons fait pour le cas à une dimension. Ceci permet une exécution très rapide de l'algorithme.
Le gain en qualité est également significatif par rapport à une interpolation bilinéaire (voir partie 3.7). Cette technique présente donc un compromis très intéressant entre performance et temps de calcul. Ceci explique en grande partie sa très large utilisation (Gimp, Photoshop, appareils photo numériques).



(a) (b)

FIGURE 3.5 – Interpolation cubique pour $a = -1/2$. (a) Noyau de l'interpolation. (b) Tracé logarithmique du module de la transformée de Fourier du noyau.

5 Interpolation de Lanczos

Le filtre de Lanczos est une version fenêtrée du sinus cardinal permettant ainsi de réduire à zéro les valeurs du sinus cardinal au delà d'un certain seuil A et donc de limiter le nombre de coefficients du filtre. La fenêtre utilisée (appelée fenêtre de Lanczos : W_L) correspond au lobe central d'un sinus cardinal dont la largeur est déterminée en fonction de A. On a alors :

$$W_L = \begin{cases} sinc\left(\frac{x}{A}\right) & \text{pour} \quad |x| < A \\ 0 & \text{sinon} \end{cases} \tag{3.11}$$

La Figure 3.6(a) illustre trois tailles de fenêtres. Le filtre de Lanczos F_L est donc construit comme suit, [Lanczos 1989], [Duchon 1979] :

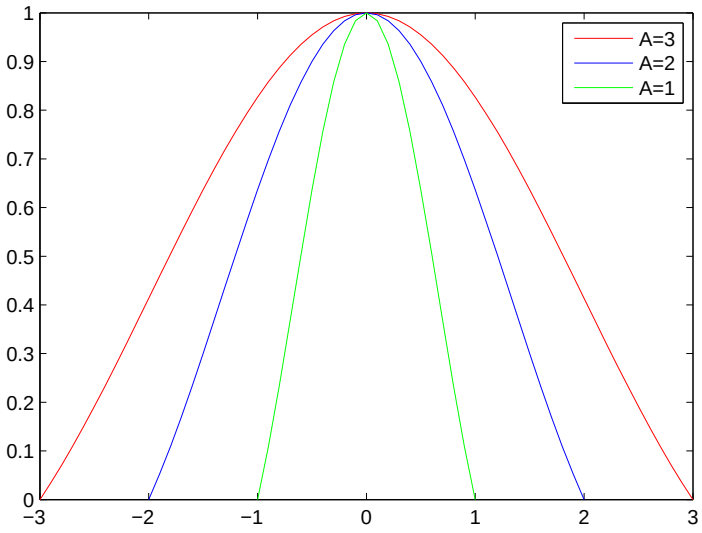
$$h(x) = \begin{cases} W_L.sinc\,(x) & \text{pour} \quad |x| < A, x \neq 0 \\ 1 & \text{pour} \quad |x| = 0 \\ 0 & \text{sinon} \end{cases} \tag{3.12}$$

Plus la fenêtre est large, plus le filtre de Lanczos sera proche du sinus cardinal et mieux le signal sera reconstruit. Par contre le nombre de calculs sera plus important. La Figure 3.6(b) montre les trois filtres construits pour trois fenêtres différentes.
La formule utilisée pour l'interpolation en tout point x est identique à l'équation d'interpolation de Shannon (3.1), au sinus cardinal près. L'interpolation est donc une simple convolution entre le signal de départ échantillonné aux points x_k et un filtre de Lanczos.

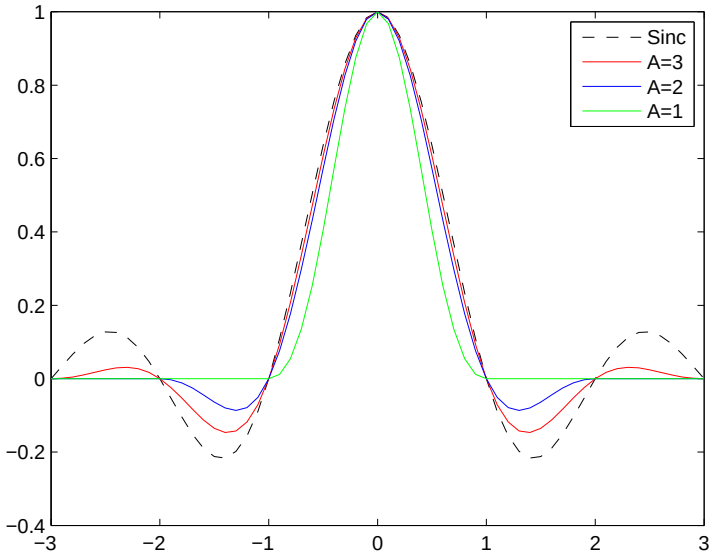
$$g(x) = \sum_{k=\lfloor x \rfloor - A + 1}^{\lfloor x \rfloor + A} f(x_k) h(x - x_k) \tag{3.13}$$

L'application au cas à deux dimensions est directe. En effet, d'après [Burge 2009], le filtre de Lanczos à deux dimensions est tel que $h(x, y) = h(x).h(y)$. En deux dimensions avec $A = 3$,

l'interpolation requiert donc 36 pixels soit 20 de plus que l'interpolation bicubique. Nous verrons dans la section 3.7 que des artefcats de *ringing* (écho de contour) apparaissent du fait des rebonds de la fonction d'interpolation. Le ringing disparaît donc quand A diminue mais l'interpolation en elle-même devient alors moins bonne car la bande passante de la fonction d'interpolation est moins élevée.



(a)



(b)

FIGURE 3.6 – (a) Illustration de trois fenêtres pour plusieurs valeurs de A. (b) Construction des filtres de Lanczos correspondants.

6 Interpolation par spline cubique

L'interpolation par spline cubique a notamment été étudiée dans [Hou 1978], [Boor 2001], [Unser 1991] et [Unser 1993]. Le noyau d'interpolation est assez proche du noyau de l'interpo-

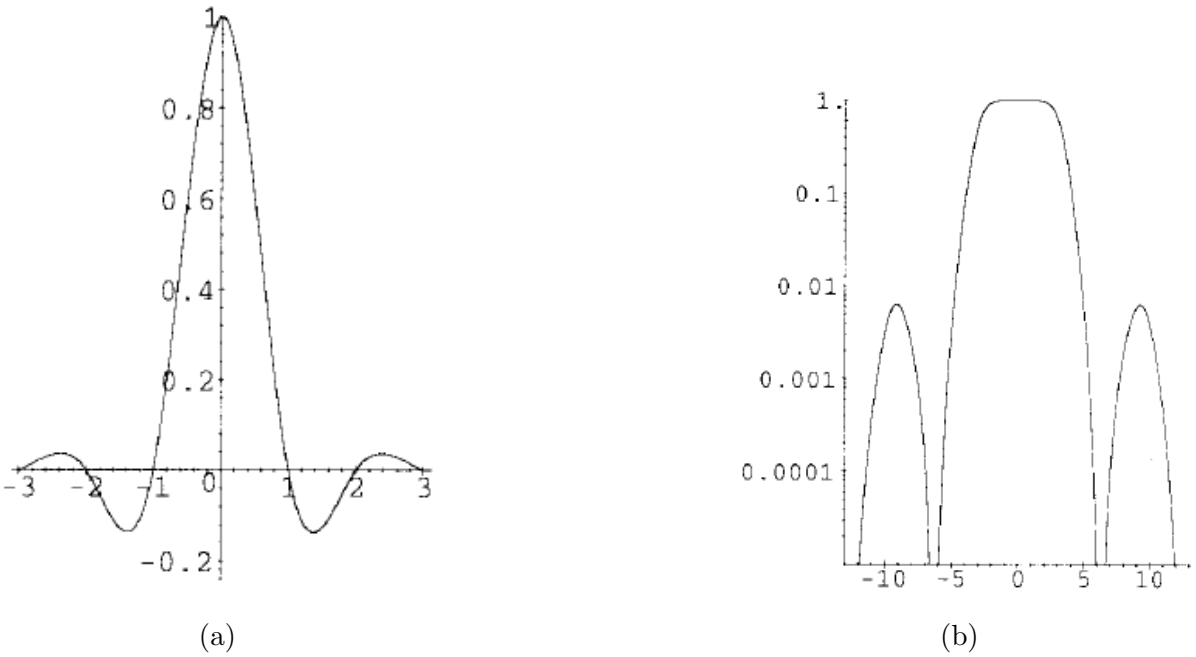
lation cubique polynomiale, mais elle impose des conditions de régularité plus strictes. En effet le noyau d'interpolation d'une spline doit être de classe $\mathcal{C}^2(\mathbb{R})$. Pour garantir cette régularité, des conditions sur la dérivée seconde de la fonction d'interpolation sont imposées. De la même manière que pour l'interpolation cubique, la fonction d'interpolation $g(x)$ peut être exprimée en fonction des échantillons du signal d'entrée $f(x_k)$. Le schéma d'interpolation est donc le même que précédemment, à savoir :

$$g(x) = \sum_{k=-\infty}^{+\infty} f(x_k)\eta^3(x - x_k) \tag{3.14}$$

Dans cette équation, introduite dans [Hou 1978], η représente le noyau de l'interpolation et est appelée spline cubique cardinale.

$$\eta^3(x) = \frac{-6\alpha}{(1-\alpha^2)} \sum_{k=-\infty}^{+\infty} \alpha^{|k|}\beta^3(x - x_k) \tag{3.15}$$

Dans cette expression, β^3 est une fonction B-spline d'ordre trois, et α est le paramètre qui dépend des zéros de la réponse impulsionnelle du filtre de β^3. La construction des B-splines et des filtres associés est explicitée dans [Unser 1991] et [Unser 1993].
Par construction, $g(x)$ s'annule en tout point entier (points d'échantillonnage) sauf en $x = 0$ où elle vaut $g(0) = 1$. Ce comportement la rapproche du sinus cardinal et donc de l'interpolateur théoriquement parfait. La Figure 3.7 illustre le comportement de la fonction cubique spline cardinale à la fois dans le domaine spatial et fréquentiel. Remarquons que la transformée de Fourier de la spline se rapproche du filtre passe-bas idéal.



(a) (b)

FIGURE 3.7 – Interpolation cubique spline cardinale. (a) Noyau de l'interpolation. (b) Tracé logarithmique du module de la transformée de Fourier du noyau.

Il est possible de voir dans cet exemple que le noyau d'interpolation s'étale sur plus de coefficients que le noyau de l'interpolation cubique et correspond à celui de l'interpolation de Lanczos pour $A = 3$. A une dimension, six coefficients seront donc nécessaires à l'interpolation de l'intervalle entre deux échantillons. De plus, à l'instar de l'interpolation cubique, la détermination

des paramètres de la fonction d'interpolation n'est pas directe, et demande la résolution d'un système matriciel tridiagonal. Ce supplément de calcul est conséquent mais améliore les capacités théoriques d'interpolation puisque Keys dans [Keys 1981] prouve que l'erreur d'interpolation décroit en $0(h^4)$. Des fonctions jusqu'au troisième degré peuvent donc être interpolées de façon exacte.

Le passage à la dimension deux se fait comme pour les interpolations précédentes grâce à la séparabilité de la fonction d'interpolation. De manière générale, l'interpolation spline cubique donnera des images plus homogènes que l'interpolation bicubique du fait de la continuité de la dérivée seconde. Généralement, les transitions seront mieux approchées ce qui réduira l'effet de flou dans les images interpolées, mais un *overshoot* sera présent proche des transitions franches ce qui introduit du *ringing* au bord des contours. La section suivante propose une comparaison des résultats des méthodes d'interpolation présentées dans cette partie.

7 Comparaison des résultats

Les méthodes d'interpolation présentées dans la section précédente se basent sur des approximations plus ou moins grossières de l'interpolation idéale par sinus cardinal. Cette partie présente différents résultats d'interpolation sur des images présentant des propriétés intéressantes Théoriquement, plus le nombre de coefficients utilisés pour l'interpolation d'un intervalle entre deux échantillons est important, meilleure est la qualité de l'interpolation. Les défauts classiques des méthodes d'interpolation, ou artefacts d'interpolation, sont connus et apparaissent principalement au niveau des contours de l'image. Ces derniers sont des transitions plus ou moins franches de niveaux de gris qui peuvent être interprétées comme des discontinuités dans le signal. Par définition, les discontinuités ne sont pas représentables exactement par un polynôme de degré fini. Nous allons donc observer le comportement de ces interpolations au niveau des contours de l'image et étudier la présence des artefacts suivants :

- Le flou. Il est présent lorsqu'une transition de l'image d'origine n'est plus aussi franche dans l'image interpolée.
- L'aliasing. Il apparaît quand les hautes fréquences de l'image d'origine sont mal reproduites dans l'image interpolée. L'aliasing apparaît alors comme une basse fréquence parasite.
- Le ringing. Ce phénomène aussi appelé *écho*, est présent le long des contours. Il est souvent dû à l'*overshoot* (ou l'*undershoot*) introduit par les rebonds de la fonction d'interpolation.
- L'effet d'escalier apparaît sur les contours dont la direction n'est ni horizontale ni verticale. Le contour rectiligne de l'image originale est crénelé dans l'image interpolée, et l'information de direction est perdue au fur et à mesure des interpolations. Il est dû au fait que l'interpolation lorsqu'elle réalisée sur les lignes et les colonnes de l'image dépend de la phase. En effet, une différence de phase d'une ligne à l'autre (ou d'une colonne à l'autre) créé un décalage sur le contour interpolé. En deux dimensions, l'accumulation de ce déphasage créé l'effet d'escalier sur le contour.

Le premier exemple est une partie de l'image Barbara montrée en Figure 3.8, comportant des rayures hautes fréquences. La Figure 3.9 contient les interpolations par un facteur quatre par quatre (quatre en horizontal et quatre en vertical) de l'image 3.8(b).

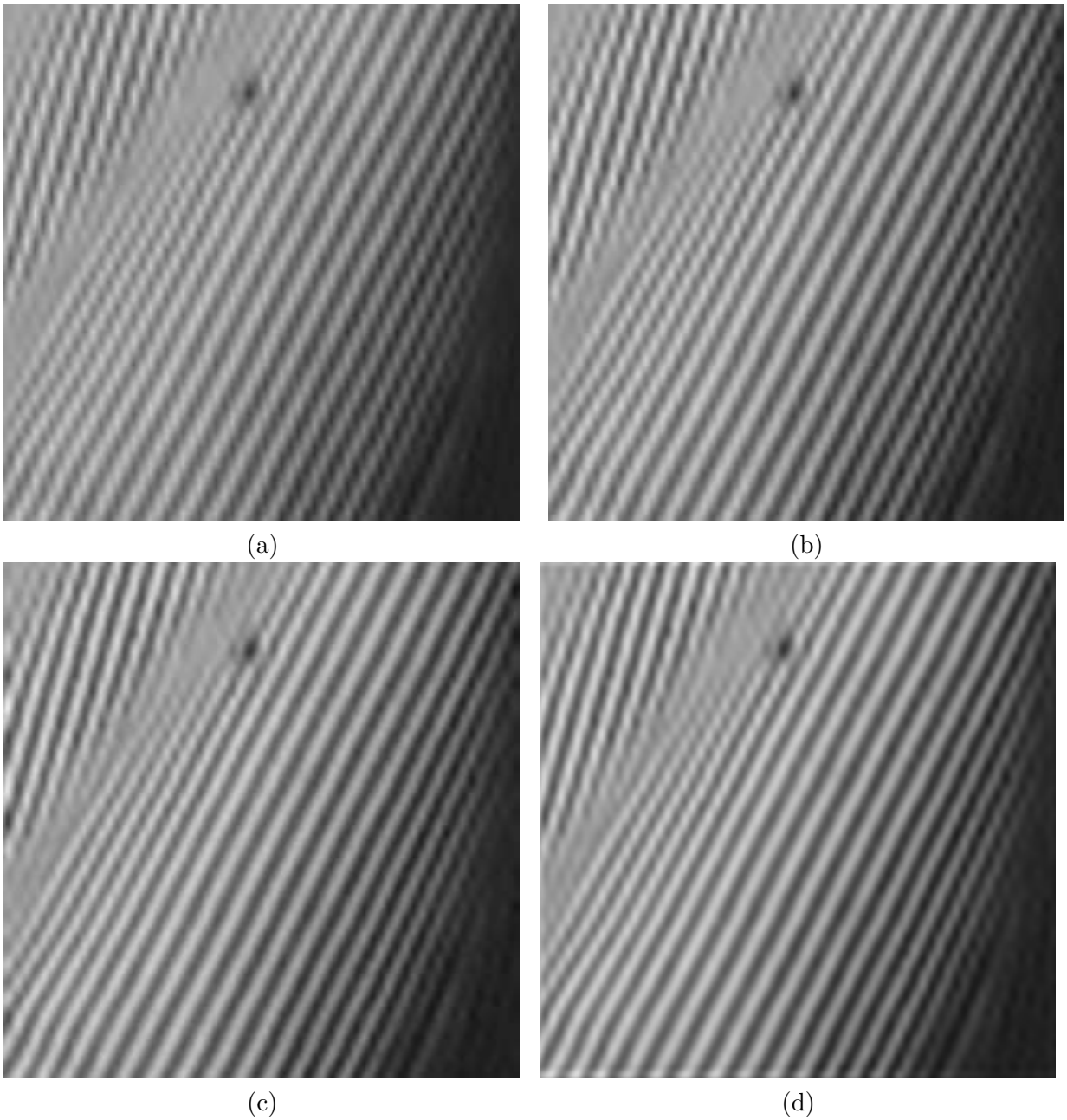
Cet exemple montre que l'interpolation linéaire en Figure 3.9(a) atteint très rapidement ses limites. En effet, l'utilisation d'un voisinage trop restreint ne permet pas de reconstruire efficacement les rayures. Le moyennage effectué éclaircit donc les bandes noires et assombrit les blanches, ce qui réduit le contraste général de l'image et donne une impression de flou. De plus, l'aliasing de la partie supérieure gauche de l'image est très prononcé, ainsi que l'effet d'escalier sur la plupart des rayures.
L'interpolation cubique permet de mieux gérer les transitions noir/blanc des rayures du fait du

plus haut degré du noyau d'interpolation. L'effet de flou est donc réduit et le contraste est mieux conservé. Par contre l'aliasing et l'effet d'escalier restent assez prononcés.
Enfin, les interpolations de Lanczos et spline cubique donnent des résultats assez similaires. Le noyau d'interpolation des deux méthodes est en effet fortement similaire. Le contraste est encore amélioré et l'aliasing sur la partie supérieure gauche réduit. L'effet d'escalier sur les rayures les plus larges s'avère également diminué. En revanche, dès que la fréquence des rayures est augmentée, l'aliasing est à nouveau présent.

Pour faire la distinction entre la méthode spline cubique et la méthode de Lanczos, il faut observer le comportement près de discontinuités franches. L'exemple de la Figure 3.10 est une croix blanche sur fond noir agrandie d'un facteur 4×4. Dans cet exemple, le rendu des contours est encore une fois meilleur pour les interpolations de Lanczos et spline, en particulier à cause de la faiblesse de l'effet d'escalier sur ces derniers. Cependant, contrairement aux interpolations linéaire et cubique, il est possible d'observer l'effet de ringing le long des contours dans les images de la Figure 3.10(c)-(d). Ce dernier est engendré par l'enchaînement des lobes positifs et négatifs des noyaux d'interpolation. Notons, et ce de manière générale, que l'interpolation spline introduit un ringing moins important que l'interpolation de Lanczos.

(a) (b)

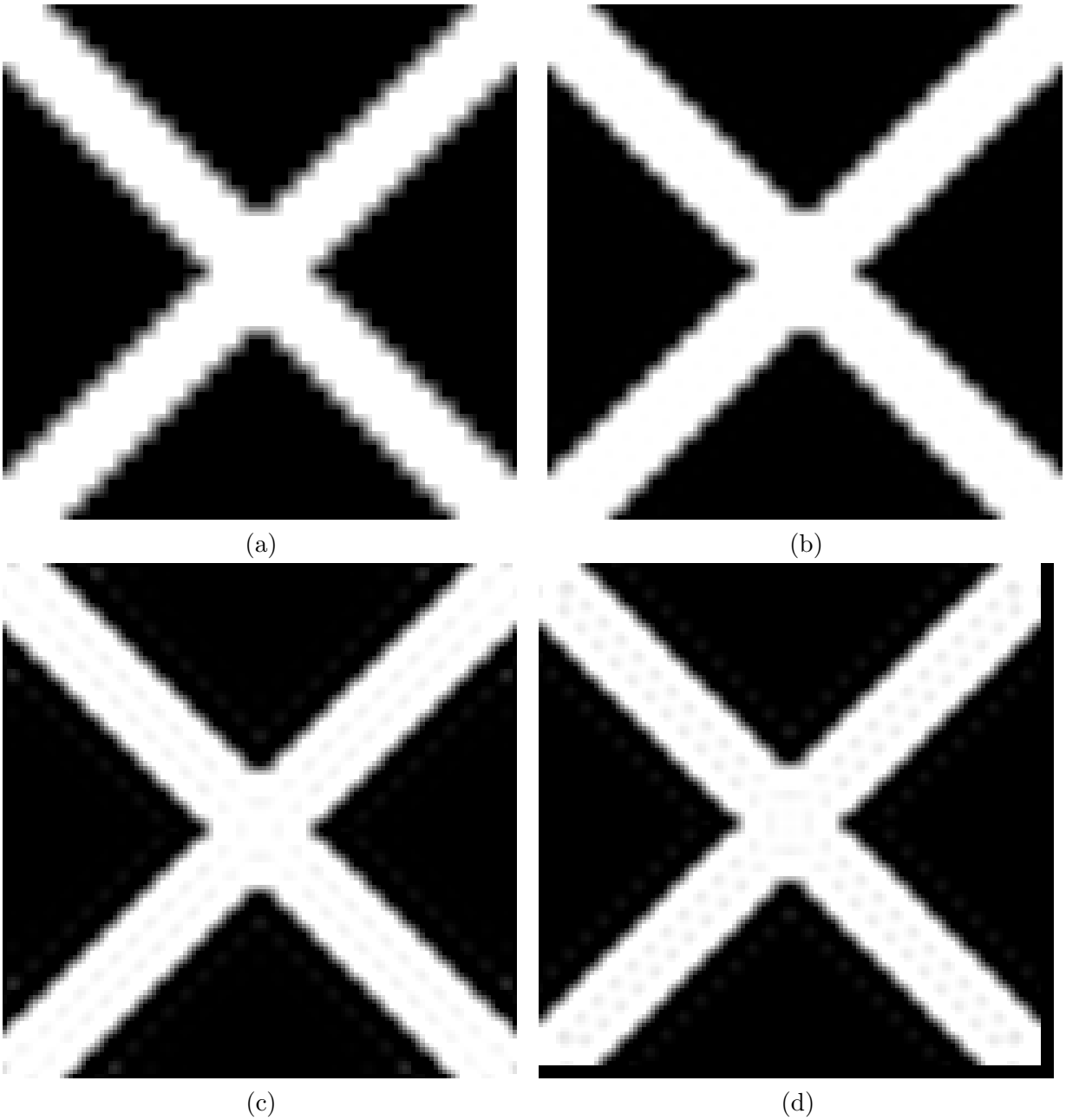
FIGURE 3.8 – (a) Image originale. (b) Partie de l'image à analyser, agrandie ici par réplication de pixels.



(a) (b)

(c) (d)

FIGURE 3.9 – (a) Interpolation linéaire. (b) Interpolation cubique. (c) Interpolation spline cubique. (d) Interpolation de Lanczos (taille de fenêtre $A = 3$).



(a) (b)

(c) (d)

FIGURE 3.10 – (a) Interpolation linéaire. (b) Interpolation cubique. (c) Interpolation spline cubique. (d) Interpolation de Lanczos (taille de fenêtre $A = 3$).

8 Conclusion sur les méthodes d'interpolation

Pour conclure, nous avons illustré avec ces deux exemples l'intérêt de prendre en compte des voisinages importants lors de l'interpolation. Le surplus de calcul engendré, s'il est acceptable, permet d'obtenir une qualité d'interpolation bien meilleure que les interpolations linéaire ou cubique avec des comportements se rapprochant de l'interpolation idéale par sinus cardinal. Ce gain de qualité est à relativiser puisque comme les voisinages utilisés ne sont pas infinis comme le voudrait l'interpolation théoriquement idéale, l'artefact de ringing est introduit. Notons cependant que de nombreux algorithmes parviennent a posteriori retirer efficacement cet artefact, et que de nombreux post-traitement de ce genre sont présents dans les systèmes d'interpolation industriels. De manière générale, toutes les méthodes présentées ici engendrent des artefacts plus ou moins prononcés. Selon le type d'images à interpoler et le domaine d'application, certains artefacts seront plus gênants que d'autres. Le choix de l'algorithme est donc un compromis entre le temps qu'il est possible d'allouer à l'interpolation et les artefacts qui vont être introduits.

L'impact visuel des dégradations engendrées par les différentes méthodes d'interpolation est d'autant plus important qu'elles sont présentes sur les contours des images interpolées. En effet, les contours peuvent être perçus comme des discontinuités de niveaux de gris qu'il est impossible d'interpoler de manière efficace avec les outils d'interpolation classiques. Ce sont les erreurs réalisées lors de l'interpolation de ces discontinuités qui engendrent les artefacts présentés précédemment. Afin de contrer ce problème, de nouvelles méthodes d'interpolation ont été étudiées. Elles visent non pas à augmenter le degré du polynôme d'interpolation mais à orienter le noyau d'interpolation dans la direction du contour, permettant ainsi de ne plus voir ce dernier comme une discontinuité mais comme une zone régulière. Interpoler le contour le long de sa propre direction ne nécessite alors qu'un noyau d'interpolation très simple, tout en préservant l'aspect du contour de l'image basse-résolution dans l'image haute-résolution.

C'est dans cette démarche que s'inscrit la méthode que nous avons mis au point dans cette étude. Pour construire une telle méthode d'interpolation directionnelle, nous avons fait le choix de nous appuyer sur une méthode de détection de direction des contours de l'image réalisée en amont de l'interpolation, et permettant d'adapter l'interpolation en fonction des direction obtenues. La suite de ce mémoire s'intéresse donc à présenter notre méthode d'analyse directionnelle, ainsi que les méthodes faisant référence dans le domaine pour ensuite introduire notre interpolation directionnelle qui en découle. Les résultats que nous obtenons seront comparés avec les résultats des interpolations classiques, mais également avec ceux obtenus par des méthodes d'interpolations directionnelles récentes, faisant référence dans le domaine.

Deuxième partie

Analyse directionnelle d'une image

Cette partie a pour but de décrire notre méthode d'estimation de direction des contours d'une image, dans l'optique de réaliser une interpolation directionnelle. Nous verrons que la détection de directions dans le domaine discret est une tâche difficile pour laquelle il est souvent nécessaire de faire des compromis entre une bonne localisation spatiale et une bonne précision angulaire. Nous étudierons dans un premier chapître l'état de l'art de certaines méthodes de détection de directions de contours faisant référence dans la littérature.

Par la suite, nous détaillerons notre propre méthode d'estimation de directions de contours. Nous commencerons par l'utilisation que nous faisons des ondelettes qui offrent à notre méthode la possibilités de décrire les contours dans des espaces multirésolutions. Ceux-ci permettent d'adapter la résolution locale de l'étude afin d'optimiser l'analyse de la direction du contour. La notion d'échelle doit en effet être prise en compte afin d'adapter l'étude à tous types de contours.

Puis nous aborderons la problématique d'estimation d'orientation de contours en elle-même, basée sur une segmentation en *quadtree* de l'image initiale. Cette approche vise à isoler dans chaque bloc du *quadtree* au plus une direction de contour afin d'estimer sa direction. La méthode de détection de la direction du contour dans chaque bloc se fait en recherchant la direction de la droite naïve (8-connexe) qui présente le moins de variations, et qui est considérée comme parallèle au contour. Cette opération est effectuée dans chaque bloc de la division auxquels la direction détectée est associée.

Enfin, l'approche que nous proposons est comparée à deux méthodes d'analyses directionnelles présentées dans l'état de l'art. Nous essaierons d'observer dans quel cadre notre méthode améliore les deux autres approches, ainsi que les comportements de chaque méthode dans des cas concrets. Pour que l'évaluation soit pertinente, les trois méthodes sont en effet testées sur des images de synthèse idéales, puis sur ces mêmes images corrompues (ajout de bruit blanc puis de bruit de compression), et enfin en les comparant sur des extraits d'images naturelles.

Chapitre 4

Méthodes de détection de direction

1 Direction du gradient

Cette méthode d'estimation de direction de contours est une des plus simple qui soit. Elle se base sur les images de gradients verticaux et horizontaux pour donner la direction du gradient en chaque point (x, y) de l'image originale $I(x, y)$. Soit G_x et G_y respectivement les images de gradients horizontaux et verticaux, calculées ici avec les filtres de Sobel [Feldman 1969], dans un voisinage de (3×3) pixels.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * I \tag{4.1}$$

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I \tag{4.2}$$

Dans cette équation l'opérateur $*$ représente la convolution. La matrice contenant les directions du gradient est calculée ainsi :

$$\Theta(x, y) = \arctan(\frac{G_y}{G_x}) \tag{4.3}$$

Notons que la direction du contour indique la direction où les variations sont maximales. Les directions des contours sont donc perpendiculaires aux directions du gradient puisque les niveaux de gris le long d'un contour sont homogènes.
Un exemple de détection de directions calculé à partir de l'équation 4.3 est montré en Figure 4.1. D'autres approximations du gradient tels que les filtres de Prewitt peuvent être utilisées. Des résultats très similaires sont obtenus.
Bien que rapide et très simple, cette méthode présente des lacunes dès qu'une bonne précision est désirée sur la valeur des angles. Elle est en effet très sensible aux gradients parasites qui ne constituent pas de contours d'objets. Par exemple, l'ajout de bruit, ou la présence d'un dégradé de niveaux de gris au niveau d'un contour perturbera fortement la bonne détection de sa direction. De plus, l'estimation du gradient étant réalisée sur un voisinage restreint, (3×3) dans notre cas, ne permet pas d'obtenir une bonne précision sur la valeur de l'angle calculé. L'approche basée sur le gradient est donc très bien localisée spatialement, à défaut d'une bonne robustesse et d'une bonne précision au niveau de la valeur de l'angle à estimer.

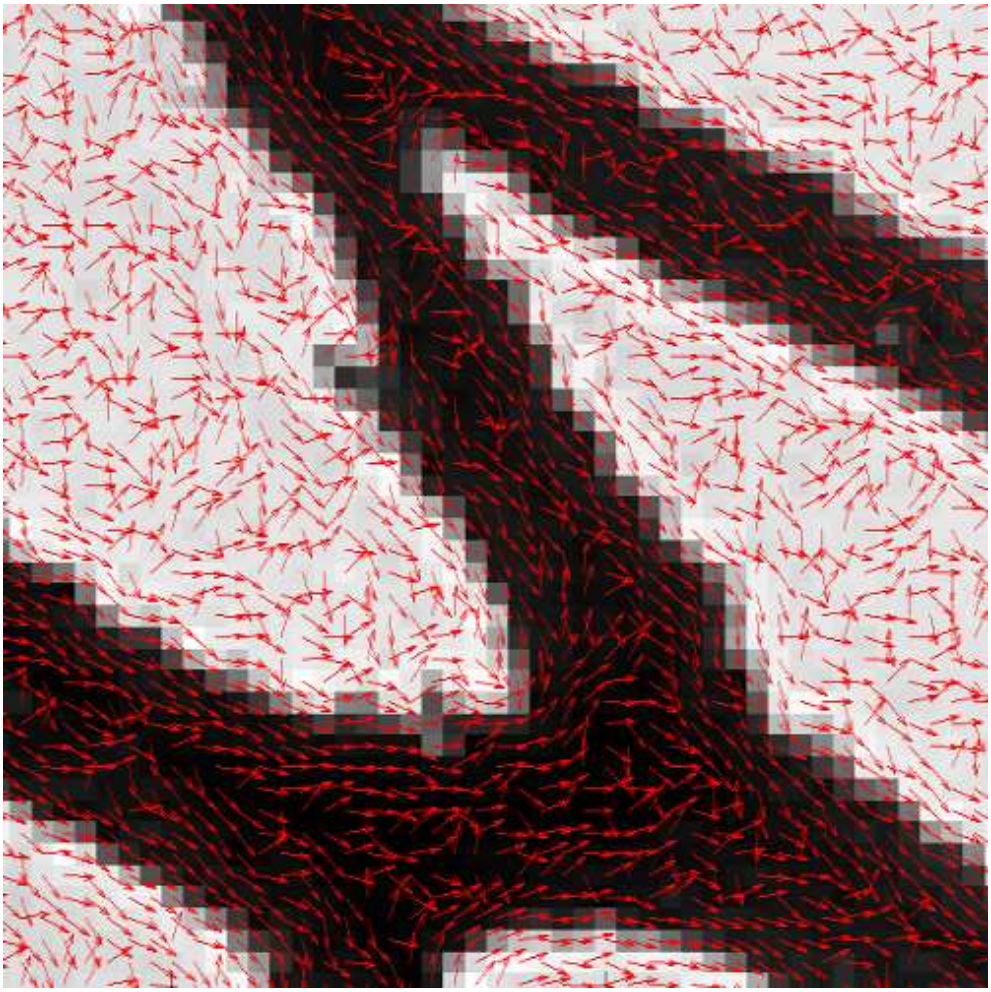


FIGURE 4.1 – Directions de contours calculées à partir des gradients horizontaux et verticaux de l'image.

2 Diffusion d'orientations

Afin d'éviter les problèmes de bruit et d'estimation trop locale et donc peu fiable des directions, Perona [Perona 1998] a proposé le concept de diffusion d'orientations. Celui-ci a pour but de lisser les orientations de l'estimation initiale (obtenue à partir de la méthode du gradient), en diffusant de façon itérative les orientations prépondérantes. Cette méthode est basée sur l'algorithme de descente du gradient d'une fonction d'énergie inspirée des modèles physiques de diffusion.

2.1 Principe

Plus concrètement, l'équation de diffusion de la chaleur est utilisée dans ce cas, dont voici la formulation :

$$u_t = \Delta u \tag{4.4}$$

avec $u_t = \frac{\partial u}{\partial t}$ et Δ l'opérateur Laplacien d'une fonction réelle u définie sur un plan à deux dimensions. La solution de cette équation peut être obtenue grâce à la minimisation d'une fonction de coût $E(u)$, décrite dans l'équation 4.5. Cette fonction peut s'exprimer comme la somme des coûts associés à chaque voisinage dx et dy de u.

$$E(u) = \int_{x,y} |\nabla u|^2 \, dxdy \tag{4.5}$$

L'algorithme de descente du gradient permet de trouver un minimum de $E(u)$. Dans le cas à une dimension ou k est la coordonnée spatiale, la formulation de l'algorithme de descente du gradient s'exprime de la manière suivante :

$$\frac{\partial u_k}{\partial t} = -\lambda \frac{\partial E(u)}{\partial u_k} = \lambda(u_{k-1} - 2u_k + u_{k+1}) \tag{4.6}$$

où λ est un paramètre de vélocité qui fait varier la vitesse de convergence de l'algorithme.

Dans le cas de diffusion d'orientations, il faut définir la distance ou le coût entre deux orientations voisines. Le problème est résolu en prenant par exemple comme valeur de coût entre deux orientations successives θ_A et θ_B, la valeur :

$$E(\theta_A, \theta_B) = 1 - \cos(\theta_A - \theta_B) \tag{4.7}$$

Remarquons que dans ce cas, si $\theta_A = 0$ et $\theta_B = 2\pi$, le coût est nul. En généralisant l'équation 4.5 au cas discret à une dimension, la fonction de coût devient :

$$E(\theta) = \sum_k (1 - \cos(\theta_{k+1} - \theta_k)) \tag{4.8}$$

La minimisation de $E(\theta)$ par l'algorithme de descente de gradient donné dans l'équation 4.6 devient alors :

$$\theta(k, 0) = \theta_0(k) \tag{4.9}$$

$$\frac{\partial \theta_k}{\partial t} = \lambda[\sin(\theta_{k+1} - \theta_k) + \sin(\theta_{k-1} - \theta_k)] \tag{4.10}$$

L'itération de cet algorithme permet de diffuser les orientations des contours sur des voisinages de plus en pus grands au fur et à mesure des itérations. Ceci a pour effet de diminuer le bruit de détection de la solution initiale. Il est à noter que d'autres fonctions de coût peuvent être utilisées et que les résultats seront peu impactés. Pour l'extension de l'équation 4.10 au cas à deux dimensions, le voisinage choisi pour le calcul de la fonction de coût est un voisinage de 3×3 pixels autour du pixel central.

2.2 Résultats

La Figure 4.2 montre la détection réalisée à partir de cette méthode sur la même image que précédemment. Il est évident de constater la bien meilleure homogénéité des résultats obtenus

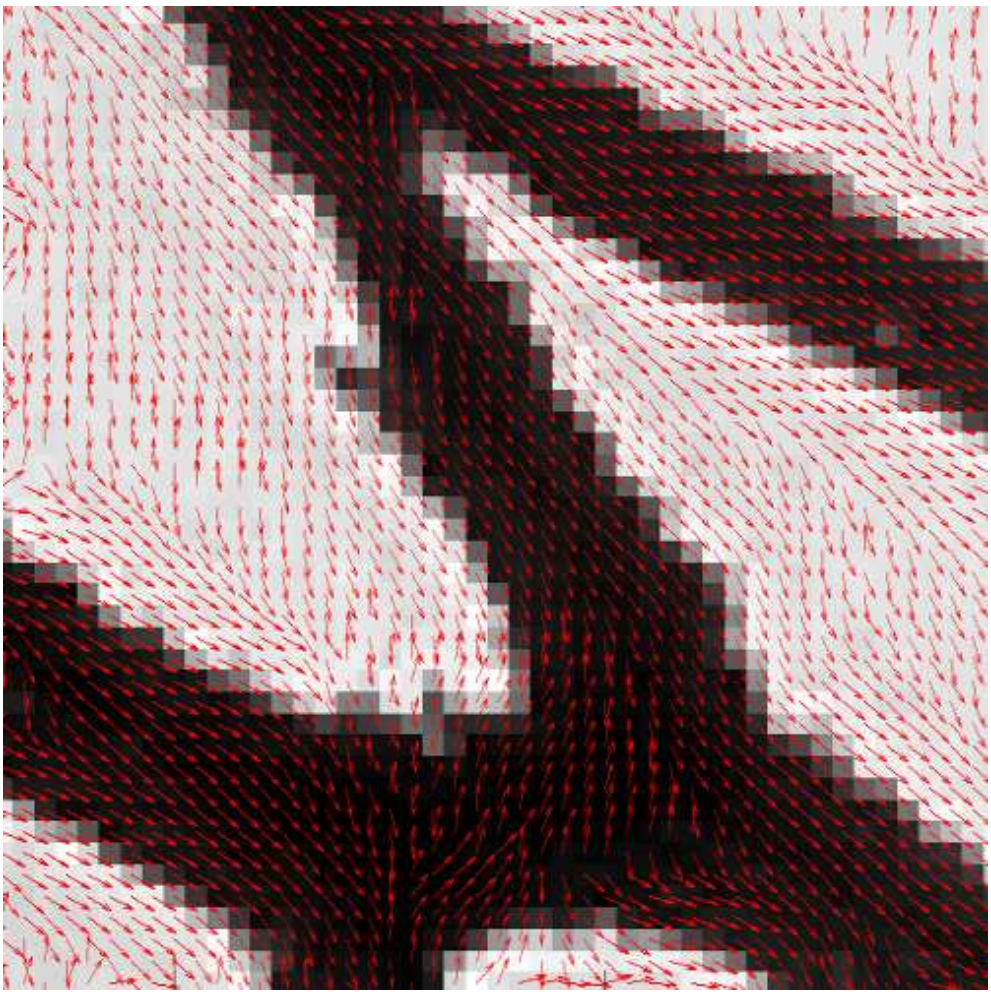


FIGURE 4.2 – Directions de contours calculées à partir de la méthode de diffusion d'orientations de Perona.

par rapport à l'estimation initiale (Figure 4.1). La diffusion permet en effet de diminuer l'effet des gradients parasites lorsqu'ils sont entourés de contours ayant une direction bien marquée. Les résultats obtenus sont plus robustes que ceux donnés par la méthode du gradient, et donc plus exploitables en vue d'applications futures. Nous étudierons en particulier en partie 3 une méthode d'interpolation directionnelle basée sur cette méthode.

Si la méthode de diffusion est efficace pour détecter les orientations de contours continus, elle ne peut pas, par principe, détecter efficacement les contours qui présentent des discontinuités dans leur direction. En effet, le modèle physique de diffusion de chaleur choisi est applicable uniquement dans des cas continus. Cette méthode ne permet donc pas de capter les changements de directions brusques (intersections de contours par exemple), ce qui peut être préjudiciable pour certaines applications. Ce phénomène peut être observé en Figure 4.3 dans laquelle la transition entre les deux directions au centre de la croix se fait de façon continue, alors que le changement de direction est net. Localement, cela implique des détections de direction erronées.

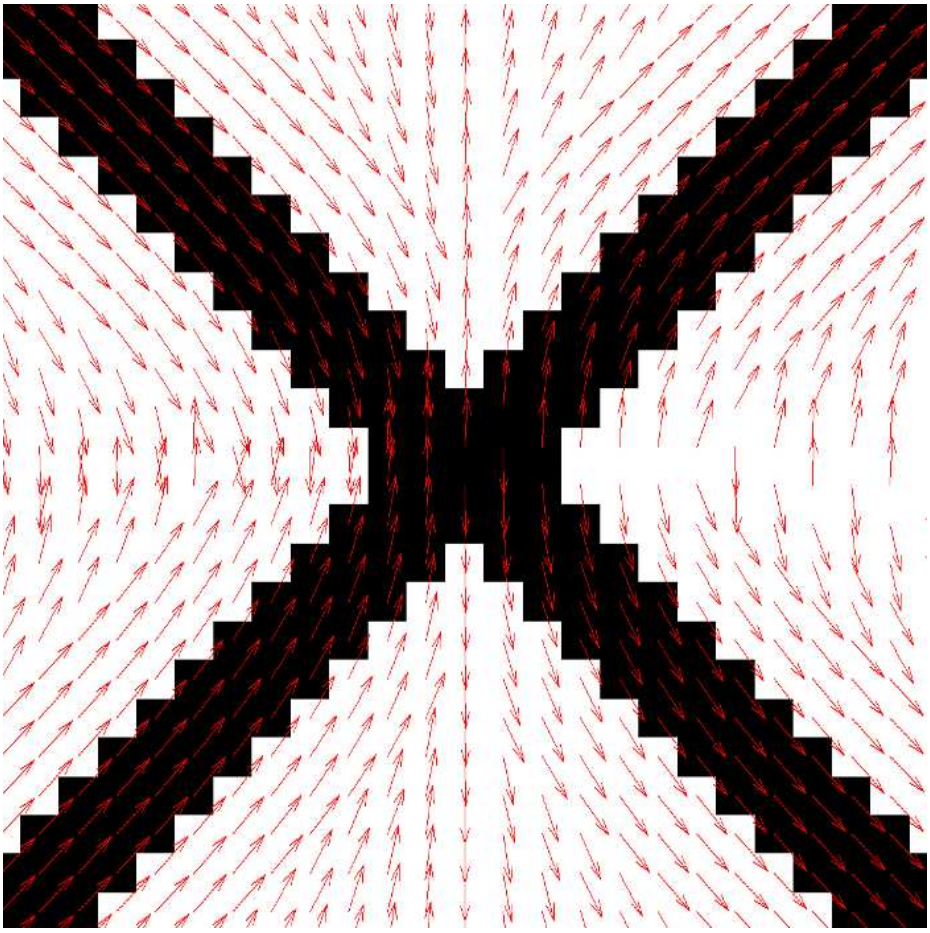


FIGURE 4.3 – Comportement de l'algorithme de Perona au niveau d'une singularité sur une image binaire.

3 Transformée de Radon

La transformée de Radon est un outil qui peut être utilisé pour détecter des directions de contours. Il s'agit en effet une projection d'une fonction à deux dimensions, $f(x, y)$ sur les droites $D(\theta, r)$ de direction θ par rapport à l'axe des ordonnées ([Deans 1983]).

$$R_{r,\theta}[f(x,y)] = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\delta(r - x\cos(\theta) - y\sin(\theta))dxdy \tag{4.11}$$

Dans cette équation, r est la distance sur la droite de direction perpendiculaire à θ, entre l'origine (centre de l'image) et la ligne de direction θ (voir le schéma de la Figure 4.4). En théorie, r varie de $-\infty$ à $+\infty$. Selon l'équation 4.11, un point de l'espace de Radon (r, θ) correspond à l'intégration des niveaux de gris le long d'une ligne de direction θ du plan de l'image. Lorsque la fonction f est définie dans le domaine discret, la définition des lignes de projection doit alors être adaptée. De nombreux papiers se sont intéressés aux propriétés de la transformée de Radon discrète, en particulier concernant les propriétés de reconstruction. Nous n'aborderons pas ces aspects mais les références suivantes donnent un aspect global de l'état de l'art : [Beylkin 1987], [Ramesh 1989]. En général, les projections de l'image discrète $f[m, n]$ sont calculées en intégrant les lignes définies par interpolation des pixels de la grille. Ce procédé permet de calculer les projections sur le nombre d'angles voulu, en tenant tout de même compte du fait qu'il y aura une forte redondance entre deux projections voisines si les directions de ces dernières sont très proches.

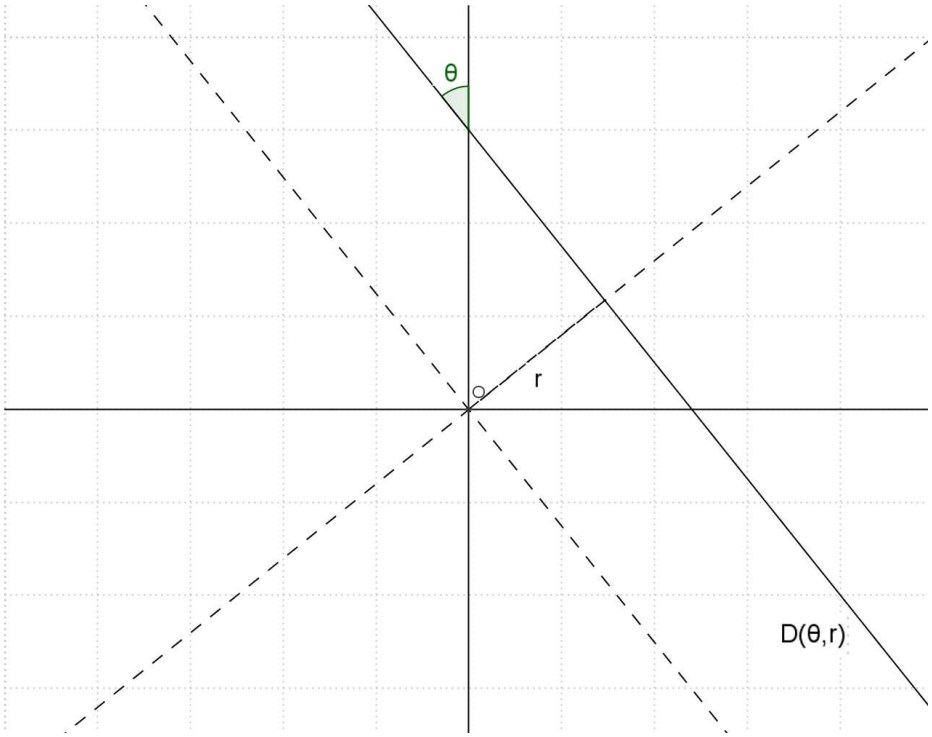


FIGURE 4.4 – Illustration de la droite D d'équation $D : y\cos(\theta) - x\sin(\theta)$, le long de laquelle sont intégrés les niveaux de gris de l'image pour obtenir la transformée de Radon.

Le résultat de l'intégration le long des projections permet donc d'évaluer la régularité des niveaux de gris de l'image le long de lignes droites. L'article [Jafari Khouzani 2005] propose une manière intéressante d'extraire l'information d'orientation à partir de la transformée de Radon. Elle part du principe que la projection le long de la ligne se trouvant sur le contour aura une valeur très différente de la ligne (de même direction) voisine ne se situant pas sur le contour. Ainsi la variance σ_θ^2 de toutes les projections de cette direction est maximale. Toujours d'après [Jafari Khouzani 2005], la détection peut être améliorée en prenant la dérivée seconde de la variance, ce qui a pour effet d'améliorer la visibilité de la direction dominante. La direction estimée, $\hat{\theta}$ est donc :

$$\hat{\theta} = arg\min_{\theta}(\frac{d^2\sigma_\theta^2}{d\theta^2}) \tag{4.12}$$

où σ_θ^2 est la variance de la projection de l'image selon θ.
La Figure 4.5 montre un résultat de détection pour une texture présentant majoritairement deux directions à $\pi/4$ et $3\pi/4$. Nous pouvons remarquer dans cet exemple que le minimum de la dérivée seconde est obtenu pour la direction à $\pi/4$ puisqu'il y a plus de lignes droites contenues dans les éléments ayant cette orientation que dans les éléments orientés à $3\pi/4$.

Contrairement aux méthodes présentées plus tôt, le résultat donné par la transformée de Radon sera global à un voisinage de l'image. Cette méthode permet de détecter la direction de contour prédominante dans un voisinage donné et s'applique donc particulièrement bien à l'étude de textures directionnelles. Une étude plus approfondie de cette méthode, ainsi qu'une comparaison avec d'autres techniques d'estimation d'orientation sont présentées en partie 7.1.

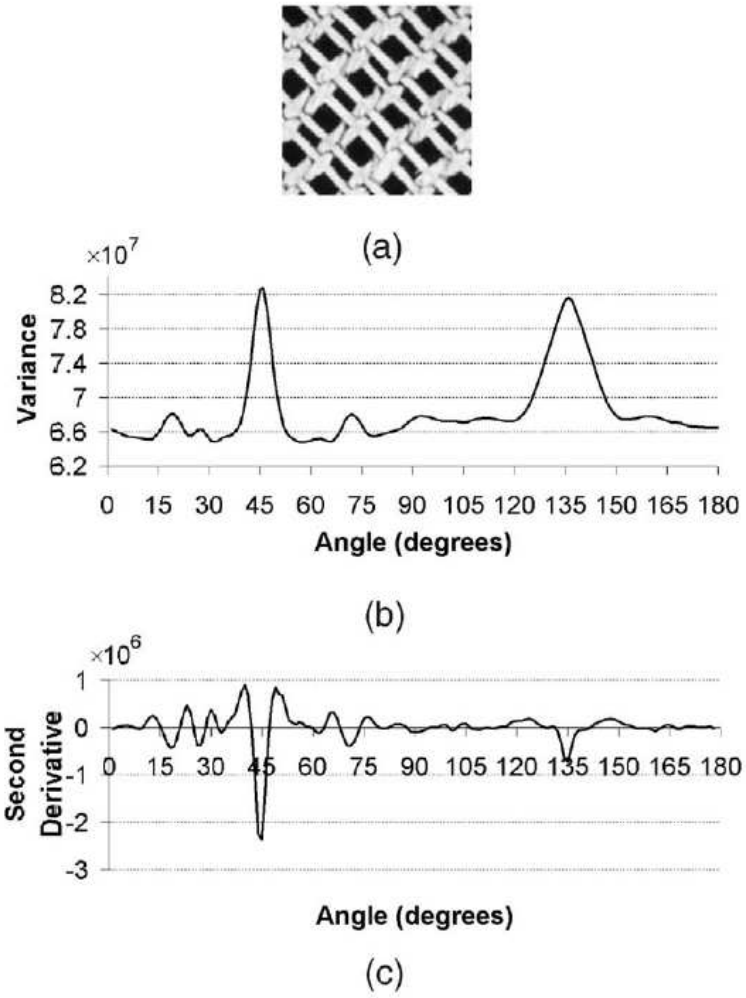


FIGURE 4.5 – Exemple tiré de [Jafari Khouzani 2005]. (a) Texture présentant plusieurs directions. (b) Variance de la projection pour différents angles. (c) Dérivée seconde de (b).

4 Méthode utilisée pour la création des bases de bandelettes

La méthode de détection d'orientation utilisée pour la création des bases de bandelettes est également intéressante. Cette méthode vise à estimer les directions locales des contours dans des voisinages dont la taille est adaptée en fonction du contenu de l'image.

4.1 Introduction aux bandelettes

Les bandelettes ont été créées par S. Mallat et E. Le Pennec ([LePennec 2002] [Mallat 2005]). Elles optimisent la représentation d'une image en la plaçant dans un domaine qui utilise la géométrie des objets pour pouvoir coder de manière efficace les informations contenues dans les contours. L'intérêt d'une telle transformation est important puisque les transitions entre objets (discontinuités) sont bien mieux représentées que lors d'une transformée en ondelettes classiques qui ne traitent seulement les directions de la grille cartésienne discrète. Dans le domaine des ondelettes, le nombre de coefficients nécessaires à la représentation correcte d'une discontinuité est très important (voir la Figure 4.6). A l'inverse, si la transformée suit la direction du contour comme illustré dans la Figure 4.7, le contour n'est plus vu comme une discontinuité mais comme une série de courbes régulières. Le nombre de coefficients nécessaire pour représenter la discontinuité initiale est alors fortement réduit.

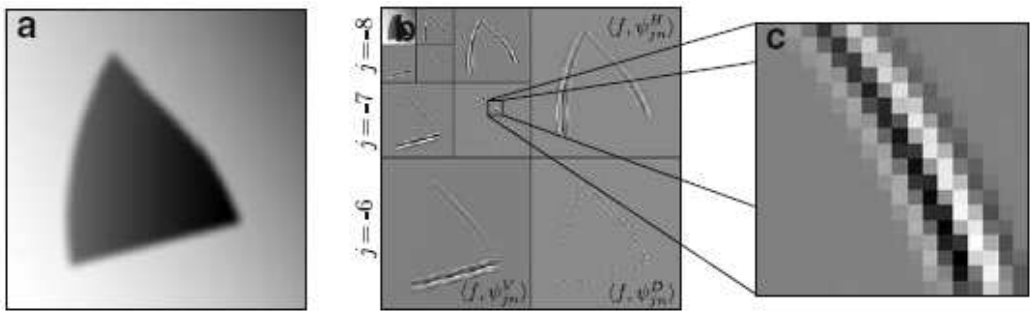


FIGURE 4.6 – Transformée orthogonale en ondelettes 2D (d'après [Mallat 2007]).

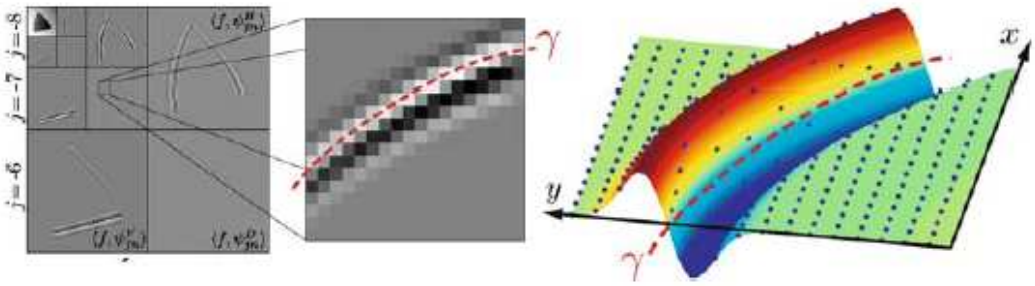


FIGURE 4.7 – Utilisation du rayon de courbure γ pour optimiser la représentation des coefficients d'ondelettes dans l'espace des bandelettes (d'après [Mallat 2007]).

Une seconde génération de bandelettes a vu le jour grâce à G. Peyré [Peyre 2005b], [Mallat 2007], [Peyre 2005a]. Il propose une discrétisation ainsi qu'une implémentation grâce à un algorithme qui optimise l'utilisation des bandelettes pour des applications à des images réelles.
De nombreuses applications bénéficient de ce genre de transformations, et en particulier la compression d'image [LePennec 2005], [LePennec 2006] et le débruitage [LePennec 2007], ou encore l'inpainting [Maalouf 2006]. D'autres méthodes de transformées géométriques avec des applications similaires ont été étudié en suivant un concept de base proche. Nous pouvons citer parmi

celles-ci les curvelets [Starck 2002], les contourlets [Do 2007], [Do 2006a], [Do 2006b], [Do 2005], [Do 2003], ou encore les directionlets [Velisavljevic 2003], [Velisavljevic 2006]. D'autres travaux sur des méthodes utilisant des ondelettes orientées peuvent être trouvés dans [Chappelier 2006], [Taubman 1994], [Chang 2006], [Meyer 1996], [Chang 2007].

4.2 Algorithme de calcul

Le fonctionnement de l'algorithme de transformée en bandelettes rapide et discrète tel qu'il est décrit dans [Peyre 2005b] est explicité dans ce paragraphe.

Transformée en ondelettes

Une transformée en ondelettes 2D orthogonale est tout d'abord réalisée sur l'image originale I. Elle est réalisée sur un nombre d'échelle choisie, et pour chaque orientation $k \in H, V, D$. Les étapes suivantes qui sont décrites dans les paragraphes suivants sont effectuées sur les trois images d'orientation différente et pour chaque échelle.

Estimation de l'orientation

Chaque image de coefficients d'ondelettes est tout d'abord divisée en blocs carrés dyadiques notés S. Pour chaque carré S, l'orientation dominante d est estimée. Le nombre de directions candidates du carré S dépend de la taille de celui-ci. Plus S est grand, plus le nombre de directions candidates est important. Le nombre exact de directions d est calculé dans la partie 6.1.1.
Pour estimer la direction des contours présents dans le bloc, celui-ci est projeté dans chaque direction contenue dans ce bloc. Pour réaliser la projection, une ordonnance discrète des coordonnées des pixels de S est réalisée pour chaque direction d. Les indices des pixels de S sont ainsi « triés »dans la direction d pour donner un vecteur 1D noté f_d. Ce vecteur est ensuite transformé en ondelettes 1D pour donner les coefficients b_k. Il est intéressant de noter que la création du vecteur f_d revient à projeter les pixels du bloc le long de droites discrètes fines de direction d. Le schéma de la Figure 4.8 présente un exemple de projection pour la création du vecteur f_d le long de la direction discrète $(3, 1)$ représentée par la flèche rouge. Dans cet exemple, seuls les enchaînements $(5, 4); (9, 8); (13, 12)$ sont représentatifs de la direction donnée en exemple. Le bloc est en effet trop petit pour pouvoir contenir, pour tous les pixels du bloc, des pixels alignés dans la direction testée. Notons enfin que tous les pixels du blocs sont présents dans le vecteur f_d.

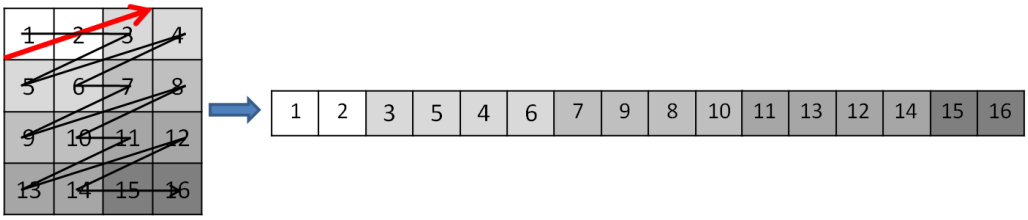


FIGURE 4.8 – Enchaînement de pixels d'un bloc de 4×4 pixels dans la direction représentée par la flèche rouge et par les paramètres discrets $(3, 1)$, selon l'algorithme de la Bandlet Toolbox.

Afin de détecter la direction qui représente le mieux la géométrie des contours contenus dans le bloc S, une quantification du vecteur b_k est réalisée. Pour un seuil donné T, nous cherchons à trouver la direction d qui minimise l'erreur d'approximation par rapport à la géométrie des contours du bloc, c'est-à-dire la direction qui est le moins affectée par la quantification effectuée. Soit R_B le nombre de bits nécessaire pour coder les coefficients quantifiés $Q_T b_k$, R_G le nombre de bits nécessaire pour coder la direction d, et f_{dR} le signal f_d reconstruit par transformée inverse

de b_k après quantification. La meilleure estimation de la géométrie est donc celle qui minimise le lagrangien suivant :

$$l(f_d, R) = \|f_d - f_{dR}\|^2 + \Lambda T^2 (R_G + R_B) \tag{4.13}$$

Notons que le vecteur f_d orienté dans la direction du contour du bloc S contient de faibles variations. Seuls peu de coefficients de sa transformée en ondelettes 1D, b_k sont nécessaires pour le représenter de manière efficace. Le terme R_B du lagrangien sera donc minimal pour cette direction. L'autre terme $\|f_d - f_{dR}\|^2$ sera également minimal pour cette direction, puisque du fait de sa régularité, le vecteur f_{dR} est mieux reconstruit (et donc plus proche de f_d) que pour d'autres directions.
De plus, pour une application telle que la compression, l'approche qui consiste à choisir l'orientation qui nécessite le moins de bits pour coder les informations du bloc, est efficace. Mallat montre dans [Mallat 2005] qu'une compression par bandelettes améliore à la fois le rendu visuel et le PSNR par rapport à une compression par ondelettes.

Construction de la partition optimale

A chaque carré S est donc associée une direction qui est considérée comme celle qui représente au mieux la géométrie des contours du bloc. Afin de tirer parti d'éventuels contours qui traversent plusieurs blocs, quatre carrés voisins peuvent être regroupés lorsque ceux-ci présentent des directions similaires, en suivant une hiérarchisation en *quadtree*. L'algorithme de regroupement utilise les propriétés additives du lagrangien qui permettent de comparer les variations internes du grand bloc avec la somme des variations internes aux quatre petits blocs afin de choisir le meilleur compromis et de conserver la taille de bloc optimale.

4.3 Résultats

Le résultat d'une segmentation d'un image de synthèse est montré en Figure 4.9. Le découpage est montré su l'image de détails verticaux de la première échelle de la transformée en ondelettes 2D. Rappelons que la segmentation en *quadtree* se fait pour chaque orientation et chaque échelle de la transformée. Il apparaît clairement dans cet exemple que l'algorithme permet d'isoler une direction de contours par bloc grâce à la segmentation en *quadtree*. De ce fait, un découpage plus fin est effectué aux coins du triangle alors que des blocs plus grands sont utilisés pour représenter les contours des cotés. La méthode d'estimation d'orientation présentée dans cette partie sera évaluée plus en détail dans la partie 7.1.

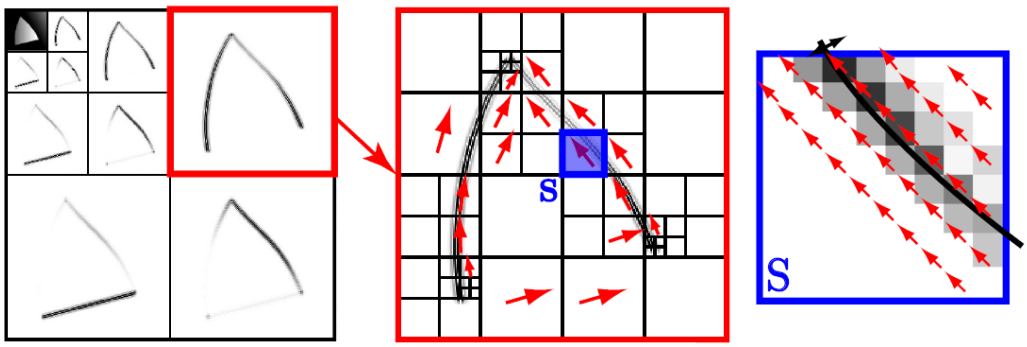


FIGURE 4.9 – Résultat de la segmentation en *quadtree* et direction associée à chaque bloc (d'après [Mallat 2007]).

5 Méthode IRON

A l'image de la méthode de gradient et de diffusion, la méthode IRON (Isotropic and Recursive Oriented Network : [Michelet 2007], [Da Costa 2002], [Michelet 2004]) consiste à estimer l'orientation de composantes directionnelles de l'image pixel à pixel. Elle est à rapprocher des méthodes telles que les filtres de Gabor ou les filtres orientables ([Bigun 1992], [Freeman 1991]) qui visent également à donner une information de direction à un pixel donné. La méthode IRON a été créée dans le but d'étudier la ou les directions de textures directionnelles.

5.1 Principe

L'opérateur IRON est un opérateur constitué d'un réseau de L lignes parallèles. Les lignes sont orientées dans la direction à étudier et sont composées d'un nombre de points P. L'ensemble du réseau de lignes forme le support utilisé pour l'étude d'une direction particulière. L'intersection des supports rectangulaires créés pour étudier les directions de l'intervalle $[0, \pi]$ forment un cercle dont le rayon C est :

$$C = \sqrt{\frac{(P-1)^2}{2} + \frac{(L-1)^2}{2}} \tag{4.14}$$

La Figure 4.10 montre la construction du réseau de lignes symétrique et centré sur le pixel où l'analyse est effectuée. L'exemple illustré ici contient trois lignes et cinq points par ligne. Le choix du nombre de lignes et du nombre de points est un compromis entre la précision spatiale désirée et la robustesse de la détection. En effet, pour l'étude d'une texture dont l'orientation change fréquemment, il vaut mieux utiliser un support compact. A l'inverse si la direction de la texture est stable, il est préférable d'utiliser un nombre de points plus important pour obtenir une bonne précision sur la direction de l'angle. Le fait d'utiliser un grand nombre de points par ligne permet d'allonger le support dans la direction à étudier, ce qui améliore la précision de la détection.

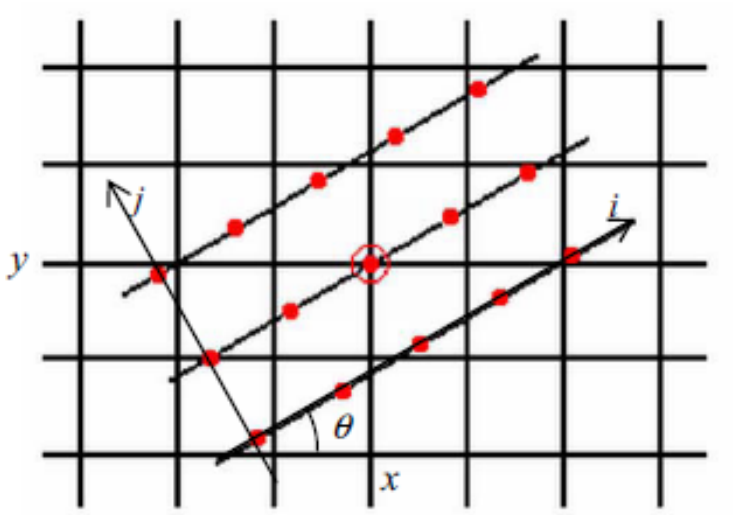


FIGURE 4.10 – Création des lignes pour l'étude directionnelle du pixel central avec la méthode IRON.

Les points situés sur les lignes du réseau ne tombant pas sur des pixels de la grille, ils sont interpolés de façon linéaire. Le calcul de variation est ensuite réalisé en sommant la variance de chaque ligne. Le minimum de variance correspond à la direction de la structure étudiée.
Grâce à cette technique, le nombre de directions testées peut être fixé par l'utilisateur en fonction de la résolution angulaire qu'il souhaite obtenir. Il faut bien sûr compter sur une augmentation du temps de calcul quand la résolution angulaire est améliorée.

5.2 Résultats

Les auteurs de [Michelet 2007] montrent que la méthode IRON se comporte de façon plus robuste au bruit que les filtres de Gabor ou que les filtres orientables. Pour obtenir une précision similaire sur la valeur de l'angle et une même robustesse au bruit, les filtres de Gabor et les filtres orientables doivent être calculés dans des supports beaucoup plus grands. Ceci introduit alors des temps de calculs plus importants.
Les auteurs montrent également que lorsque plusieurs directions passent par le même pixel, la méthode IRON donne des résultats plus précis sur la valeur des deux angles. De plus, cette méthode fait preuve d'une meilleure sélectivité dans le cas où les deux directions sont proches. Elle permet en effet à la fois de détecter la présence de deux directions disctinctes mais également de donner une valeur précise des deux directions. Ceci est valable aussi bien sur des images de synthèse que sur des images réelles.

La méthode IRON s'applique donc particulièrement bien à l'étude de textures directionnelles grâce à plusieurs points :

- Elle laisse le choix à l'utilisateur de la taille du support. Ceci permet d'adapter l'étude aux caractéristiques de la texture à étudier.
- Le support est allongé dans la direction étudiée, ce qui permet de gagner en précision.
- Elle propose de très faibles biais de détection ainsi qu'une très bonne sélectivité, même dans le cas de supports compacts.

Malgré ses bonnes caractéristiques, nous expliquerons plus tard le choix de ne pas utiliser cette approche pour l'étude d'images naturelles.

6 Synthèse et application à l'interpolation

Parmi les méthodes présentées dans cette section, deux catégories principales peuvent être distinguées. La première comporte les méthodes qui estiment l'orientation des contours pixel à pixel (méthode du gradient, diffusion d'orientation, filtres de Gabor, IRON), alors que les méthodes de la deuxième catégories associent une orientation à un voisinage donné (transformée de Radon, bandelettes). La méthode de Perona se situe à l'intersection de ces deux catégories puisqu'elle utilise, grâce à la diffusion, les directions des pixels voisins pour estimer l'orientation d'un pixel.
Pour une application telle que l'interpolation, la direction de contour est une information particulièrement intéressante. Par défaut, une interpolation classique isotrope utilise les pixels voisins du pixel manquant. Un contour oblique sera donc dégradé par une telle interpolation puisque l'information de direction contenue par des pixels éloignés est perdue. C'est pour cette raison que les artefacts classiques tels que l'effet d'escalier ou l'aliasing apparaissent.
Ainsi, connaître la direction du contour permet d'identifier les pixels à mettre en relation afin de conserver l'information de direction dans l'image agrandie. En faisant intervenir ces pixels, qui peuvent être relativement éloignés du pixel d'étude, les artefacts cités plus haut sont évités. L'information de direction du contour est donc primordiale afin d'éliminer ces artefacts. Cette partie vise à expliquer comment cette information délivrée par les méthodes que l'on a présentées peut être exploitée dans un cadre d'interpolation, et de définir la solution qui nous paraît la plus adaptée.

6.1 Méthodes associant une direction par pixel

Un avantage indéniable de ces méthodes réside dans le fait qu'elles permettent une très bonne localisation spatiale des directions. La méthode IRON permet notamment de diminuer voire d'annuler le biais de détection, et d'avoir une bonne sélectivité lorsque deux directions sont présentes sur le même pixel. Elles permettent également de capter des changements de direction brusques puisque la direction associée à un pixel est décorrélée de la direction associée un pixel voisin (une

partie de la détection est malgré tout réalisée sur les mêmes pixels). Ceci n'est pas valable pour la méthode de diffusion de direction qui elle, diffuse les directions au fur et à mesure des itérations, et donc empêche de détecter des changements brutaux de directions.
Toutefois, ces deux approches se justifient parfaitement et le choix de l'une ou l'autre se fait en fonction de l'application voulue. En effet, la méthode de diffusion, bien qu'incapable de capter les discontinuités de directions donne un résultat homogène et très peu bruité. Cette propriété est particulièrement intéressante pour des applications telles que l'interpolation ou le débruitage. A l'inverse, les autres méthodes peuvent capter les discontinuités de directions, mais reste plus sensible au bruit selon la taille du voisinage utilisée.
Ces méthodes présentent néanmoins quelques inconvénients communs. Pour obtenir des résultats fiables, un grand voisinage doit être pris en compte pour l'estimation d'une direction (un grand nombre d'itérations pour la méthode de diffusion). Cela engendre un nombre de calcul important d'autant plus que l'opération est répétée pour chaque pixel de l'image. De plus, le voisinage d'étude duquel dépend la robustesse de l'analyse mais aussi la précision spatiale du résultat doit être fixé par l'utilisateur. Il en va de même pour le nombre d'itérations dans la méthode de diffusion. Ce dernier point rend difficile l'application à des images réelles à l'intérieur desquelles les propriétés des contours (fréquence, densité, régularité) varient fortement. Elles demandent une plus grande connaissance *a priori* des propriétés des images à étudier.
Il existe cependant quelques méthodes d'interpolation basées sur des méthodes de détection pixelliques ([Algazi 1991], [Bayrakeri 1995], [Li 2001], [Muresan 2005]). L'interpolation de Moloney et Jiang ([Jiang 2002]) est elle basée sur la méthode de diffusion de Perona. Nous étudierons plus tard dans ce mémoire en particulier les approches de *Moloney* et *Jiang* ainsi que celle de Li et Muresan dans un but d'évaluation et de comparaison des résultats avec ceux obtenus par notre méthode.

6.2 Méthodes associant une direction pour un voisinage

Parmi les méthodes que nous avons présenté, celles qui associent une direction à un voisinage sont la transformée de Radon et la méthode utilisée pour les bandelettes. Plus précisément, elles permettent de détecter la direction prédominante du voisinage dont la taille est déterminée à l'avance. Le problème principal de ces méthodes consiste à adapter de façon automatique la taille du voisinage pour gagner en précision spatiale.
Sur ce point, la transformée de Radon qui n'est en fait qu'un outil de projection ne permet pas de réaliser cette adaptation en tant que telle. Le traitement qui est effectué sur les données de projection et qui est présenté dans [Jafari Khouzani 2005] permet principalement de sortir la direction prépondérante du voisinage mais donne peu d'information sur l'éventuelle présence d'autres directions à l'intérieur du voisinage.
A l'inverse, la méthode des bandelettes propose une technique en *quadtree* intéressante puisqu'elle permet d'adapter la taille des voisinages d'études (blocs) en fonction de l'unicité d'une direction à l'intérieur de chaque bloc. Cette méthode présente donc plusieurs avantages :

- Bonne précision spatiale due à l'adaptation de la taille des blocs.
- Capacité à capter les changements brusques d'orientation d'un bloc à l'autre.
- Le calcul de direction est réalisé pour chaque bloc et non pour chaque pixel comme lors des méthodes pixelliques.
- Une direction unique est attribuée à chaque bloc. Cela permet d'effectuer un traitement homogène selon l'application postérieure. Ce dernier point est en particulier utile pour la compression, débruitage ou encore l'interpolation.

De plus, la résolution angulaire qu'elle offre est directement liée à la taille du bloc. La direction dominante détectée est donc forcément décrite par des pixels contenus à l'intérieur du bloc. En vue d'une application d'interpolation, ceci présente l'avantage d'éviter d'aller chercher des pixels dans des blocs voisins qui ont potentiellement une direction différente. Le rayon de recherche est

donc limité à la taille du bloc d'étude.
Il existe également quelques inconvénients lors de l'utilisation de ce genre d'approches en vue d'une interpolation :

- Il faut gérer la continuité entre deux blocs qui n'ont pas détecté la même direction.
- Il faut déterminer une taille minimale de bloc qui ne pourra plus être divisé, et il peut arriver que plusieurs directions soient encore présentes dans le plus petit bloc autorisé. Dans ce cas, comme une seule direction est attribuée par bloc, une des deux directions ne sera pas détectée.
- La partition en *quadtree* n'est pas invariante par translation.

Pour conclure, l'algorithme de segmentation des bandelettes offre des possibilités intéressantes pour l'étude des directions de contours dans le but d'une future interpolation. La méthode de recherche de directions que nous allons présenter s'inspire donc de cette technique. Cependant, plusieurs points seront modifiés pour en optimiser le fonctionnement et pour gérer les cas délicats d'une détection de direction par bloc. Ces points concernent particulièrement la manière dont les blocs sont projetés, dont les variations le long de chaque direction sont calculées, et la manière dont la construction du *quadtree* est effectuée.

Chapitre 5

Notre approche multirésolution

Contrairement à la plupart des méthodes citées précédemment, nous avons fait le choix de détecter les directions de contours de manière non-linéaire. Nous espérons ainsi pouvoir capter de manière efficace les changements d'orientation brusques des contours. La méthode que nous utilisons est à rapprocher de celle qui permet la création des bases de bandelettes. Elle passe également par un découpage de l'image en blocs qui permet d'isoler au plus une direction de contour par bloc. Cependant, il existe plusieurs divergences entre ces deux méthodes. La méthode de la Bandlet Toolbox (BT) passe par la minimisation de manière itérative d'une fonction de coût (Lagrangien), pour à la fois déterminer la taille des blocs et la direction comprise dans chacun des blocs. Le nombre d'itérations nécessaire à la convergence de l'algorithme dépend donc de la complexité de l'image à traiter, ce qui est peu souhaitable lors d'une éventuelle implémentation. Notre méthode effectue de manière séparée la recherche de directions à l'intérieur de chaque bloc, et le découpage de l'image.
De plus, la détection de directions est effectuée dans le domaine des ondelettes à une résolution localement adaptée au contour étudié. Les contours d'une image peuvent avoir des propriétés très différentes, selon la nature de l'objet qu'ils définissent. Les objets flous seront ainsi définis par des contours basses fréquences, alors que des objets texturés seront définis et composés de contours hautes fréquences. Cette disparité dans la nature des contours nécessite une adaptation locale de l'étude, pour assurer une bonne description des caractéristiques de tous types de transitions. De telles approches ont notamment été étudiées par [Lindeberg 1996].

1 La transformée IUWT

1.1 Principe et formalisation pour le cas à une dimension

La transformée IUWT (Isotropic Undecimated Wavelet Transform), a été introduite par Starck [Starck 2007]. Cette transformée est caractérisée par les deux filtres de décomposition : h et g respectivement filtres passe-bas et filtre passe-haut. Ces filtres doivent être symétriques mais pas forcément orthogonaux. Par simplicité pour le cas $2D$, la séparabilité peut être imposée. La relation entre les deux filtres est définie comme :

$$g[x] = \delta[x] - h[x] \tag{5.1}$$

avec $\delta[x] = 1$ pour $x = 0$ et $\delta[x] = 0$ pour tout $x \neq 0$. On peut vérifier que si h remplit les conditions exprimées plus haut, l'ondelette définie par le filtre g remplit la condition d'admissibilité, à savoir :

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0 \tag{5.2}$$

D'après la structure de g, il est facile de voir que les détails sont obtenus par différence entre deux approximations successives. Soit a_j et d_j respectivement les approximations et détails d'un signal

à la résolution 2^{-j}.

$$d_{j+1}[x] = a_j[x] - a_{j+1}[x] \tag{5.3}$$

Pour le passage à une approximation de résolution inférieure, le signal n'est pas sous-échantillonné comme dans la transformée en ondelettes classique, mais le filtre h est sur-échantillonné. Des zéros sont insérés entre chaque coefficient du filtre pour le passage à une résolution deux fois inférieure (transformée dyadique). On a alors :

$$a_{j+1}[x] = (h^{(j)} \star a_j)[x] \tag{5.4}$$

avec $h^{(j)}[x] = h[x]$ si $x/2^j$ est entier et $h^{(j)}[x] = 0$ sinon.
Les coefficients d'approximation et de détails sont donc de même taille que le signal original, et ce pour chaque échelle.
La reconstruction se fait également de manière simple, puisque le signal de départ $y[x]$ est reconstitué en ajoutant à l'approximation à la résolution la plus basse 2^{-J}, les détails de chaque résolution :

$$y[x] = a_J[x] + \sum_{j=1}^{j=J} d_j[x] \tag{5.5}$$

Le schéma de principe de la transformée IUWT est montré en Figure 5.1 pour un cas à deux dimensions. Notons que la théorie de la transformée IUWT introduite dans cette partie est parfaitement applicable à deux dimensions étant donné que les filtres utilisés sont séparables.

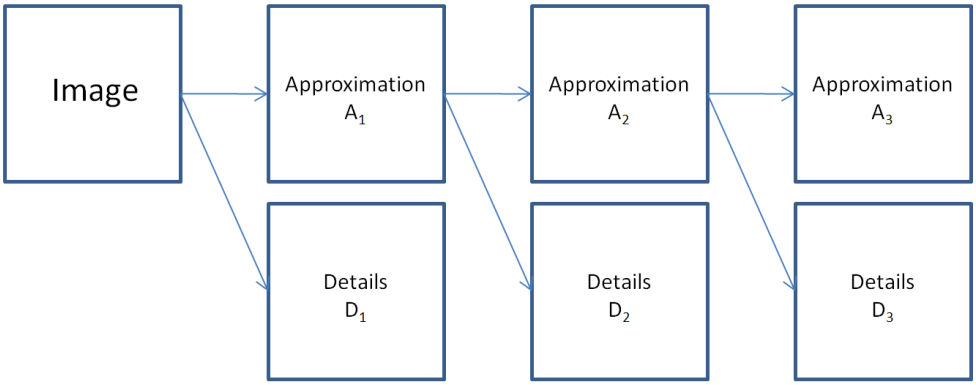


FIGURE 5.1 – Schéma de principe de l'algorithme de transformée IUWT.

1.2 Avantage de la transformée IUWT

Lors de la transformée d'une image avec l'algorithme IUWT, les images d'approximations et de détails obtenues contiennent le même nombre de pixels que l'image originale. Il est évident qu'une telle transformée introduit de la redondance d'information dans les images transformées par rapport à la transformée classique qui est, elle, sous-échantillonnée. Contrairement à des applications de compression où cette redondance est à éviter, elle peut permettre dans des applications d'analyse d'images par exemple, de rendre le choix de résolution optimale plus robuste (même si la quantité d'information reste objectivement identique), ou encore d'améliorer la détection de direction de contours.
D'autre part, comme son nom l'indique, cette transformée est isotrope, ce qui signifie qu'elle a les mêmes propriétés pour toutes les directions. La transformée en ondelettes discrète classique ne possède pas cette isotropie. En effet, les images de détails de l'image transformée sont réparties dans trois sous-bandes différentes correspondant chacune à une orientation particulière (verticale, horizontale et diagonale). Il est donc difficile de détecter des directions autres que celles-ci puisque

leurs caractéristiques sont réparties dans les trois images de détails. Grâce à la transformée IUWT, toutes les directions ont les mêmes propriétés, ce qui évite l'introduction d'un biais qui favorise certaines directions.

1.3 Exemples

Pour illustrer les paragraphes précédents, un exemple de transformée en ondelettes de la Figure 5.2 est présenté.

FIGURE 5.2 – Image originale utilisée pour la détection de résolution optimale.

La série d'images de la figure 5.3 représente les approximations successives de l'image originale (A_1, A_2 et A_3) et la série d'images de la Figure 5.4 représente les détails (A_1, A_2 et A_3) pour trois résolutions de cette même image.

(a) A_1 (b) A_2 (c) A_3

FIGURE 5.3 – Images d'approximation de l'image originale. La résolution décroit de gauche à droite. Les résolutions respectives sont : (a) 2^{-1}, (b) 2^{-2} et (c) 2^{-3}.

Du fait de leurs caractéristiques hautes fréquences, les contours de l'images sont bien mieux restitués dans les images de détails. L'analyse des directions des contours se fera donc dans l'espace des détails $\mathbf{W_j}$. De plus, les contours sont répartis dans les différentes images de détails en fonction de leur contenu fréquentiel. En effet, les détails du foulard (hautes fréquences) sont visibles surtout dans l'image 5.4(a) qui a la résolution la plus grande. A l'inverse, les détails du visage (basses fréquences) sont mieux représentés dans l'image à plus faible résolution 5.4(c).

Les apports majeurs de la transformée IUWT sont donc les suivants :

(a) D_1 (b) D_2 (c) D_3

FIGURE 5.4 – Images de détails de l'image originale. La résolution décroit de gauche à droite. Les résolutions respectives sont : (a) 2^{-1}, (b) 2^{-2} et (c) 2^{-3}.

- représenter les hautes fréquences indépendamment des basses fréquences. Ceci permet de s'affranchir des basses fréquences parasites (dégradé de niveaux de gris par exemple) qui pourraient gêner l'analyse, et de se concentrer uniquement sur les propriétés des contours.
- localiser les contours dans l'espace des fréquences.
- localiser spatialement les contours d'une image. Ce point est un avantage par rapport à la transformée de Fourier discrète qui permet elle aussi une bonne localisation fréquentielle mais en perdant l'information spatiale.

Ces propriétés de localisations spatiales et fréquentielles des contours permettent de choisir localement la résolution la mieux adaptée au(x) contour(s) présent(s) dans le voisinage choisi. Cela a pour but d'optimiser l'analyse de direction des contours. La technique de choix de résolution est décrite dans la partie suivante.

2 Critère de résolution optimale

2.1 Principe

Cette partie explique la méthode qui consiste à choisir parmi J résolutions, laquelle représente au mieux le(s) contour(s) présent(s) dans le voisinage choisi. Tout d'abord, notons que seules trois résolutions sont mises en concurrence. En effet, nous considérons qu'à partir de $J = 4$, la localisation spatiale des contours n'est pas assez précise pour pouvoir être utilisée par la suite. Le choix de résolution est donc effectué parmi les trois images de détails : D_1, D_2 et D_3.
Pour réaliser ce choix, une partition en blocs des images de détails est créée. Soit Δ_j^N le N-ième bloc de la partition de D_j. Tous les blocs créés sont carrés, et de taille identique, à savoir (16×16) pixels. Ce choix est justifié par la suite. La résolution optimale J_{opt} pour ce bloc est choisie comme étant celle qui maximise la moyenne de l'amplitude des coefficients d'ondelettes pour chaque ligne et chaque colonne du bloc. Soit M_{vert}^j et M_{hor}^j respectivement les moyennes des amplitudes de chaque colonne et de chaque ligne du bloc à l'échelle j contenant X colonnes et Y lignes.

$$M_{vert}^j = \frac{\sum_{x=1}^{X} \left(\sup_{y=1,\dots,Y}(\Delta_j(x,y)) - \inf_{y=1,\dots,Y}(\Delta_j(x,y))\right)}{X} \tag{5.6}$$

$$M_{hor}^j = \frac{\sum_{y=1}^{Y} \left(\sup_{x=1,\dots,X}(\Delta_j(x,y)) - \inf_{x=1,\dots,X}(\Delta_j(x,y))\right)}{Y} \tag{5.7}$$

Soit M_j la moyenne des amplitudes globales du bloc à l'échelle j.

$$M_j = \frac{M_{vert}^j + M_{hor}^j}{2} \tag{5.8}$$

L'échelle optimale est donc celle qui maximise l'amplitude moyenne du bloc.

$$J_{opt} = arg \sup_{j=1,...,3} [\sup(M_j)] \tag{5.9}$$

2.2 Justification du critère de résolution

Le choix de cette méthode de calcul de résolution optimale provient des caractéristiques de la fonction d'ondelette. En reprenant ses caractéristiques en une dimension, nous montrons qu'une telle fonction correspond à un filtre passe-bande. Or, la transformée en ondelettes d'un signal est une corrélation entre ce signal et une fonction d'ondelette à une certaine échelle, a, et pour un certain décalage ,b. Dans notre cas, le fait de choisir la résolution d'amplitude maximale revient donc à sélectionner l'échelle (ou la résolution) pour laquelle la corrélation entre la fonction d'ondelette et le contenu du bloc est maximale.

2.2.1 Rappel sur la transformée en ondelettes

Rappelons la formule de la transformée en ondelettes $1D$ continue :

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} x(t)\psi^*(\frac{t-b}{a})dt \tag{5.10}$$

où $*$ représente l'opération de complexe conjugué, a le facteur d'échelle, b le facteur de translation et ψ l'ondelette continue. Cette transformée peut être réécrite comme un produit de convolution :

$$X(a,b) = \int_{-\infty}^{+\infty} x(t)\frac{1}{\sqrt{a}}\psi^*(\frac{t-b}{a})dt = x \star \bar{\psi}_a(b) \tag{5.11}$$

avec

$$\bar{\psi}_a(t) = \frac{1}{\sqrt{a}}\psi^*(\frac{-t}{a}) \tag{5.12}$$

La transformée de Fourier de $\bar{\psi}_a(t)$ peut être calculée :

$$\widehat{\bar{\psi}_a}(\omega) = \sqrt{a}\hat{\psi}^*(a\omega) \tag{5.13}$$

D'après la définition d'une fonction d'ondelette, on a : $\hat{\psi}(0) = \int_{-\infty}^{+\infty} \psi(t)dt = 0$. $\hat{\psi}(\omega)$ est donc nulle pour $\omega = 0$ et correspond à une fonction de transfert d'un filtre passe-bande dont la fréquence centrale dépend de a. La transformée en ondelettes présentée dans l'équation 5.11, est bien une convolution entre le signal $x(t)$ et des filtres passe-bande dilatés et décalés.

2.2.2 Interprétation pour le cas discret à deux dimensions

L'interprétation de la transformée en ondelettes réalisée dans la partie précédente peut être appliquée au cas à deux dimensions. En effet, grâce à la séparabilité des filtres d'ondelettes utilisés, les propriétés passe-bande des fonctions d'ondelettes restent identiques. Lorsque l'image est transformée en ondelettes à J résolutions différentes, les images de détails $(D_1, \ldots, D_J)$ correspondent à la projection de l'image originale dans J bandes de fréquences différentes. Rappelons que dans notre cas, $J = 3$. Ainsi, dans chaque bande de fréquence, les forts coefficients d'ondelettes indiquent une bonne corrélation entre la fréquence de la bande et celle des contours du bloc. Notre critère de choix qui consiste à comparer les amplitudes de coefficients d'ondelettes de chaque bande permet donc de choisir la résolution correspondant au mieux à la fréquence des contours de l'image.

2.2.3 Exemple d'application

Un exemple de choix de résolution sur une image réelle est montré en Figure 5.5, selon la méthode décrite précédemment. Dans cet exemple, qui contient des contours de fréquences différentes, il apparaît clairement dans l'image résultat 5.5 (e) que les zones du visage sont représentées à la résolution la plus grossière, alors que les zones texturées sont représentées avec la résolution la plus fine.
Un deuxième exemple est illustré en Figure 5.6. Cette image de synthèse contient des contours dont la fréquence augmente en se rapprochant du centre de l'image. Les choix de résolution effectués sont cohérents avec les caractéristiques fréquentielles des contours puisque la résolution la plus fine est choisie au centre de l'image, puis la résolution moyenne est sélectionnée pour représenter les contours plus excentrés, et enfin la résolution la plus grossière est choisie pour représenter les contours des bords de l'image.
Ce critère est assez simple et donne des résultats satisfaisants dans la plupart des cas. Cependant, notons que cette méthode n'est pas infaillible. Elle est mise en difficulté dans certaines situations particulières, telles que les images bruitées ou lors de la présence de contours de fréquences différentes à l'intérieur d'un même bloc.

2.3 Discussion sur la méthode de choix de résolution

Cette partie vise à présenter les situations pour lesquelles notre méthode de choix de résolution est mise en difficulté. Nous discuterons les choix effectués et nous justifierons les compromis réalisés en vue des applications qui suivent cette analyse, à savoir la détection de direction de contours et l'interpolation du bloc.

2.3.1 Présence de contours de fréquences différentes

Il arrive que des contours de fréquences différentes se retrouvent à l'intérieur du même bloc. Dans un tel cas, le critère donne la priorité, par construction, au contour dont l'amplitude des variations est la plus élevée. Le contour moins marqué peut alors ne pas être représenté à sa résolution optimale, même si l'occurrence de ce dernier est plus importante. Ce choix, bien que non idéal, a été réalisé en se basant sur l'hypothèse que le contour qui a le plus fort contraste est à traiter en priorité puisqu'il a en général un impact visuel plus fort. La détection de direction sera donc optimisée pour détecter l'orientation de ce contour et l'interpolation sera également appliquée de sorte à ce que ce contour soit représenté de manière optimale dans la grille haute résolution.

2.3.2 Choix d'une partition par bloc

La taille des blocs dans lesquels l'étude de résolution optimale est effectuée est fixée à (16×16) pixels. Cette dimension est adaptée à notre étude puisqu'elle est assez grande pour que la fréquence des contours puisse être correctement évaluée, et suffisamment petite pour s'adapter à des éventuels changements de fréquences des contours. De plus, nous verrons que l'étude directionnelle effectuée après ce choix de résolution n'est pas pertinente dans des blocs plus grands que (16×16) pixels. En effet, le surplus de calcul engendré par une taille de bloc supérieure deviendrait trop important par rapport au gain en précision sur l'angle estimé.
Enfin, le fait de choisir une partition par bloc entraîne, comme dans l'exemple de la figure 5.5, la création d'une image résultat dans laquelle les frontières entre les blocs de la partition sont très marquées. Ceci n'est en aucun cas préjudiciable pour la suite de l'étude puisque la recherche des directions de contours se fait indépendamment dans chaque bloc, et donc à une résolution donnée, unique pour chaque bloc. L'image résultat telle qu'elle est montrée en Figure 5.5(e) par exemple, ne sert donc pas directement à la détection de direction mais chaque bloc qui la compose est traité séparément.

2.3.3 Impact du bruit

Tout d'abord, il est important de noter que cette méthode n'est pas insensible au bruit. En effet, prenons l'exemple d'un bruit blanc qui est ajouté à l'image originale. Nous pouvons observer en Figure 5.7 que l'impact du bruit est très important à la résolution fine, et qu'il décroit au fur et à mesure que la résolution diminue. Ceci est dû aux filtrages passe-bas successifs effectués sur l'image originale lors du passages aux résolutions inférieures.
Rappelons que le critère de choix de résolution consiste à sélectionner la résolution dont l'amplitude des coefficients est maximale. Lorsque le niveau de bruit ajouté devient plus élevé que les amplitudes de niveaux de gris du contour, les variations dues au bruit sont considérées comme prépondérantes par rapport aux variations du contour. Dans ce cas, notre critère penchera vers le choix de la résolution la plus fine, quelle que soit la nature du contour. Ceci est illustré en Figure 5.8, où le choix de résolution est calculé pour différentes valeurs de bruit. L'augmentation du niveau de bruit engendre donc une sélection de plus en plus fréquente de l'échelle la plus fine lorsque l'amplitude des variations du bruit (hautes fréquences) dépasse l'amplitude de variation des contours. De plus, une asymétrie des résultats peut être observée en raison de la nature aléatoire du bruit ajouté. La détection de direction future sera donc basée sur des données où le bruit a un impact très important.
Bien qu'il soit logique que la résolution fine soit choisie dans un cas bruité puisqu'elle correspond au contenu fréquentiel du bloc, une solution consisterait à ne pas autoriser le choix de la résolution la plus fine lorsque l'image est bruitée. En effet, l'impact du bruit diminue quand l'échelle augmente en raison des filtrages passe-bas réalisés pour passer aux échelles supérieures. Cela permettrait de baser la détection de direction sur des données où l'énergie du bruit est moins présente, au risque de mal détecter les directions des contours hautes fréquences. Cela implique également une connaissance à priori de la présence ou non de bruit dans l'image, ainsi que son amplitude. De tels estimateurs sont aujourd'hui présents dans la plupart des systèmes industriels.

3 Bilan du choix de résolution

Le choix de résolution passe tout d'abord par la transformation de l'image originale dans le domaine des ondelettes à trois échelles différentes. La méthode IUWT a été choisie pour réaliser la transformée. Cette technique présente deux avantages principaux pour notre étude. Elle permet tout d'abord de conserver la même taille d'image quelle que soit la résolution choisie, ce qui facilite notre traitement par blocs de tailles identiques. De plus, la redondance d'information présente aux faibles résolutions permet d'obtenir de meilleurs résultats lors du calcul de la direction du contour du bloc. Enfin, la transformée IUWT est isotrope contrairement à la transformée en ondelettes classique. De ce fait, l'analyse de direction de contour pourra se faire dans un seul espace, dans lequel toutes les directions ont les mêmes caractéristiques. Aucun biais ne sera alors introduit lors de la détection de direction.

Ensuite, une analyse locale par blocs de dimension fixe (16×16 pixels) est effectuée pour choisir la résolution la plus adaptée aux contours du bloc. Ce choix est effectué pour chaque bloc, en prenant l'échelle pour laquelle l'amplitude des coefficients d'ondelettes sont maximaux, c'est à dire la résolution pour laquelle la corrélation entre la fréquence de l'ondelette et la fréquence du contour est la plus forte. Cette technique permet d'optimiser la représentation du contour en le plaçant dans un espace propice, dans le but d'améliorer la détection de direction qui suit.

Enfin, certains inconvénients de cette technique ont été mis en avant, en particulier dans le cas de blocs contenant des contours de fréquences différentes et dans le cas d'images bruitées. Pour les blocs contenant plusieurs types de contours, une approche plus fine pourrait être réalisée en diminuant la taille des blocs ou en en modifiant la forme. Cependant, cela aurait une grande influence

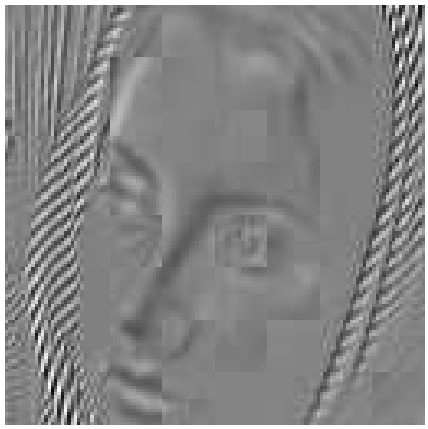
sur la détection de direction qui devrait alors être effectuée dans une région plus restreinte pour conserver la cohérence sur la résolution étudiée, diminuant ainsi la précision et la robustesse de la direction estimée. Le compromis réalisé consiste à donner la priorité au contour dont l'amplitude est la plus forte en partant du principe qu'il aura un impact visuel plus important et qu'il est donc a traiter en priorité, et de prendre des blocs de taille fixe, en l'occurrence (16×16) pixels. Cette dimension est suffisamment grande pour que le calcul de résolution optimale soit pertinent, mais suffisamment petite pour que les changements de fréquences des contours puissent être pris en compte.
Le calcul de résolution se fait donc manière simple, avec une complexité algorithmique et un nombre de calculs limités, et permet d'optimiser l'estimation de direction du contour que nous allons décrire dans la partie suivante.

(a) - Image originale

(b) - Image D_1

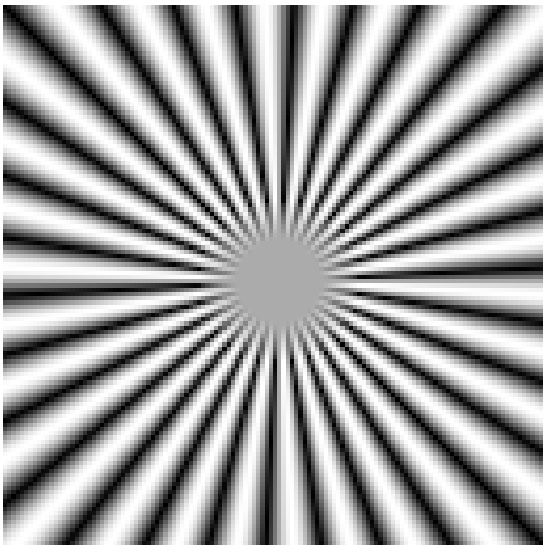
(c) - Image D_2

(d) - Image D_3

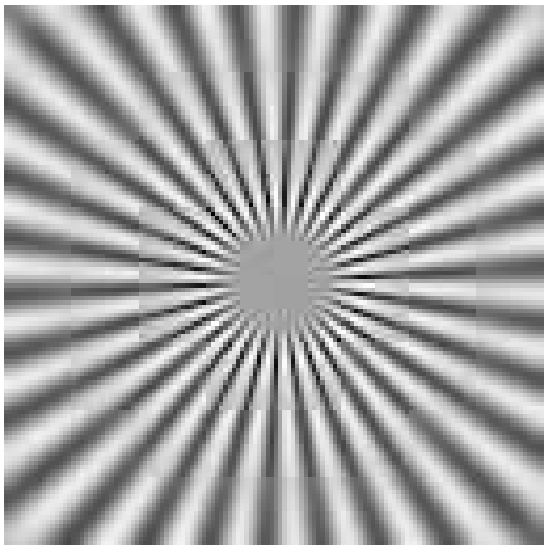


(e) - Image dont la résolution a été adaptée

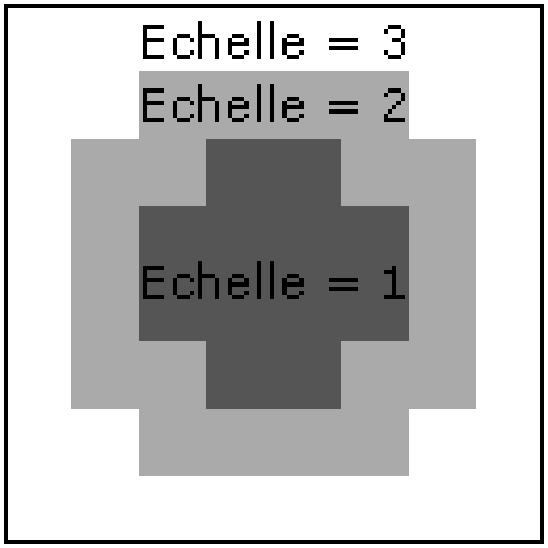
FIGURE 5.5 – Choix local de résolution la plus adaptée aux contours sur une image réelle.



(a) - Image originale

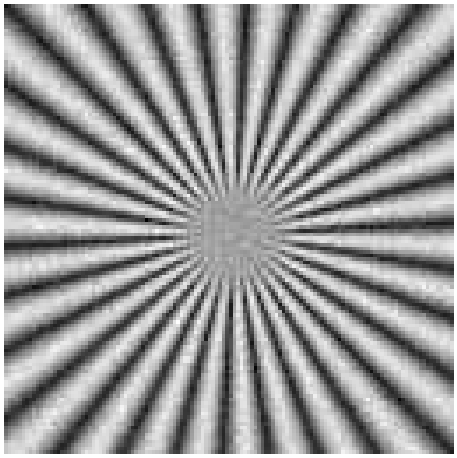


(b) - Image composée des résolutions choisies

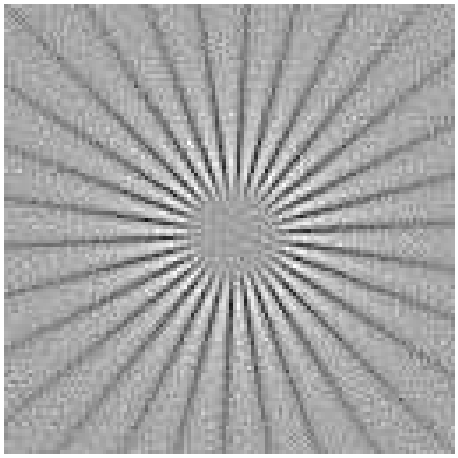


(c) - Choix explicit des échelles

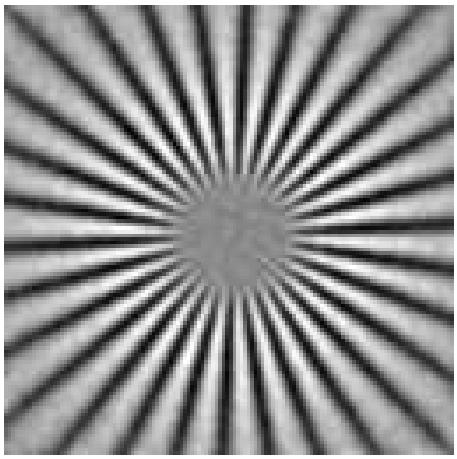
FIGURE 5.6 – Choix local de résolution la plus adaptée aux contours sur une image de synthèse.



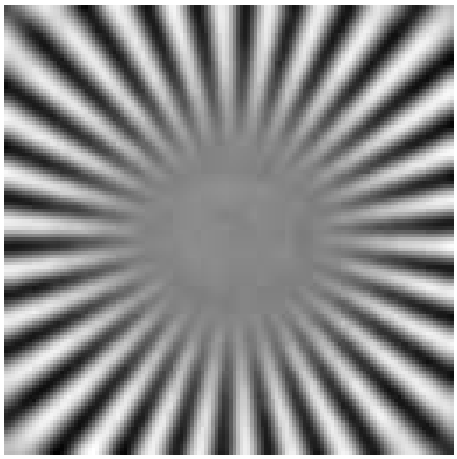
(a) - Image de synthèse bruitée



(b) - Image D_1



(c) - Image D_2



(d) - Image D_3

FIGURE 5.7 – Impact du bruit sur les différentes échelles

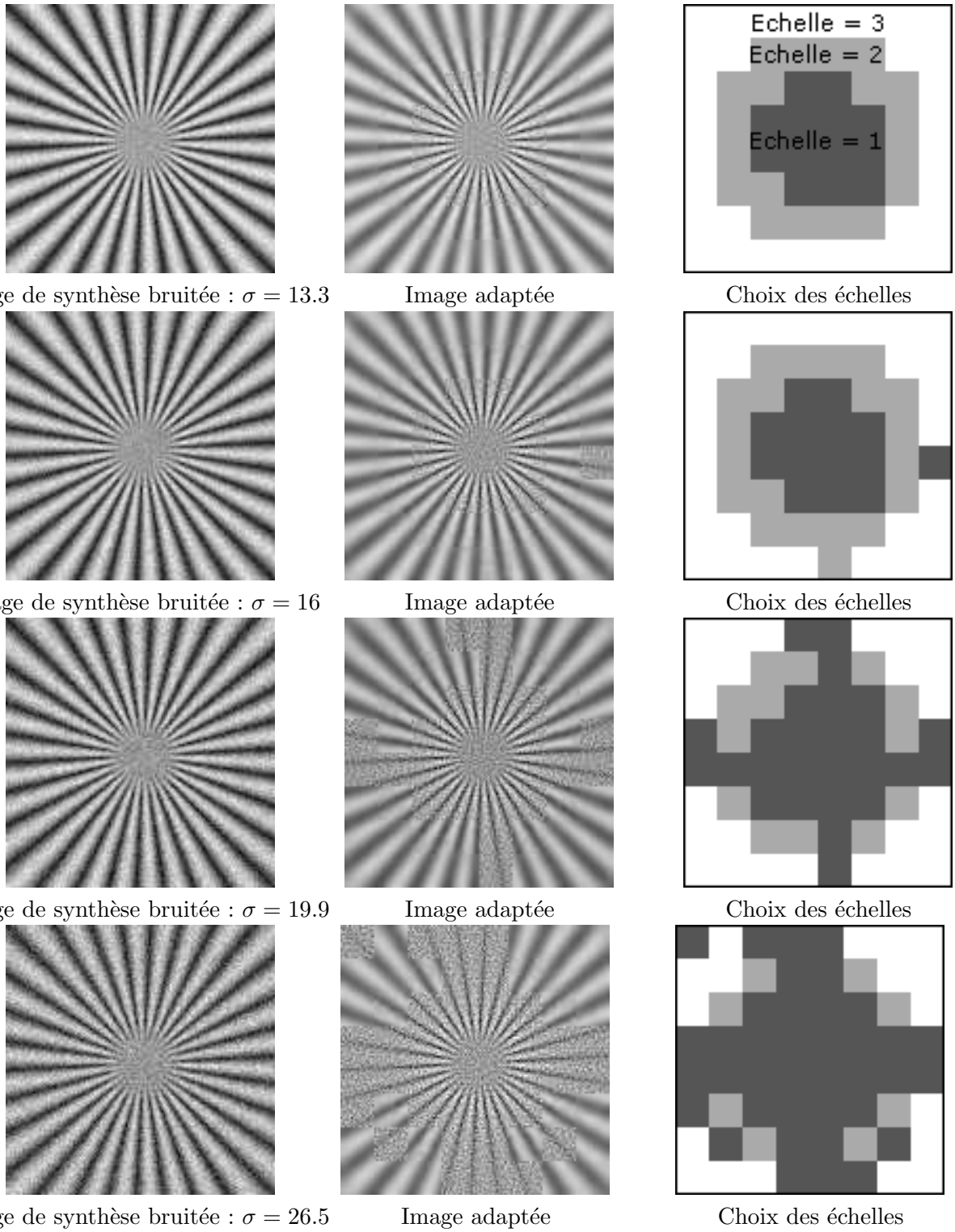


Image de synthèse bruitée : $\sigma = 13.3$ Image adaptée Choix des échelles

Image de synthèse bruitée : $\sigma = 16$ Image adaptée Choix des échelles

Image de synthèse bruitée : $\sigma = 19.9$ Image adaptée Choix des échelles

Image de synthèse bruitée : $\sigma = 26.5$ Image adaptée Choix des échelles

FIGURE 5.8 – Impact du bruit sur le critère de choix de résolution

Chapitre 6

Contribution à la recherche de la direction prédominante

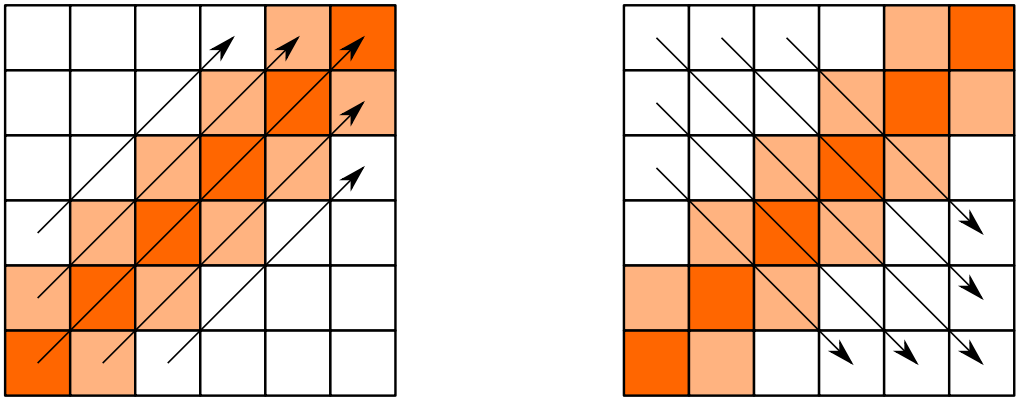
Cette partie vise à introduire notre méthode de détection de direction de contours. La première étape de cette détection, expliquée dans la partie précédente, consiste à représenter les contours de l'image à leur résolution optimale en les transformant dans des espaces d'ondelettes. Ce choix de résolution étant effectué bloc par bloc, l'estimation de direction qui en découle sera appliquée sur ces mêmes blocs, représentés dans l'espace des ondelettes à leur résolution optimale.
La recherche de direction de contours dans une image n'est une tâche facile puisque dans le domaine discret, l'estimation précise d'une direction nécessite de travailler dans de grands voisinages. Mais plus les voisinage utilisés sont grands, plus la précision spatiale sur l'emplacement du contour est faible. Il est donc nécessaire de trouver un compromis entre précision sur l'angle estimé et localisation spatiale du contour.

La méthode de détection de direction que nous utilisons, consiste à trouver la droite naïve (introduite dans le chapître 1.2.2) qui est parallèle au contour du bloc. Cela revient à trouver la droite naïve le long de laquelle les variations des coefficients d'ondelettes sont les plus faibles. Nous étudierons la manière dont le bloc est projeté selon une droite naïve, et comment les quantités de variations sont calculées.
Dans le cas où plusieurs directions de contours se trouvent à l'intérieur du même bloc, nous utiliserons un algorithme de division en *quadtree* qui permet de découper le bloc principal en plusieurs sous-blocs de taille identique de façon récursive, afin d'isoler à terme une seule direction de contour par sous-bloc. Ce découpage permet de réduire la taille des voisinages utilisés, et donc de gagner en précision spatiale, mais réduit la précision et la robustesse sur la valeur de l'angle détecté. Cet algorithme de découpage permet donc de conserver des grands voisinages pour l'étude de directions isolées, et à l'inverse de réduire les tailles de voisinages pour les zones comprenant plusieurs directions de contours. Nous expliciterons la mise au point de notre critère de division, qui décide de la présence d'une ou plusieurs directions contours à l'intérieur d'un même bloc.

Enfin, nous comparerons notre méthode d'estimation de direction de contour avec deux autres techniques qui elles aussi effectuent une détection de direction par bloc, à savoir la méthode de la Bandlet Toolbox (BT), et la transformée de Radon. Nous nous intéresserons aux erreurs de détection pour plusieurs tailles de bloc, afin d'observer le comportement de chacune des méthodes en particulier dans des petits voisinages, mais également pour plusieurs fréquences de contours pour étudier la robustesse des détections à des contours de nature différentes. Enfin, des cas bruités (bruit blanc et bruit de compression) et des exemples réels seront testés pour conclure cette partie sur des résultats se rapprochant d'une utilisation concrète.

1 Projection du bloc selon une direction

La détection de direction à l'intérieur d'un bloc revient à trouver la direction des droites naïves qui contiennent les variations les plus faibles. En effet, plus une projection contient de variations, plus le nombre d'intersections avec le contour est grand. A l'inverse, lorsque les variations sont minimales, la direction dans laquelle a été projetée le bloc est considérée comme parallèle au contour. Ce principe est illustré en Figure 6.1. La direction de la droite la plus régulière est alors définie comme orientation prédominante du bloc. Cette partie explique comment un bloc est projeté sur des droites, et comment les variations sont calculées pour chaque projection effectuée.



(a) (b)

FIGURE 6.1 – Tracé des droites dans un bloc contenant un contour. Les droites de l'exemple (a) contiennent les variations minimales car elles sont parallèles au contour. A l'inverse, l'exemple (b) illustre les droites contenant le plus grand nombre de variations dues aux multiples intersections avec le contour.

1.1 Cadre de la projection

La projection du bloc s'effectuant dans le domaine discret, il est nécessaire de revenir sur les notions de géométrie discrète introduites dans le chapître 1.2.2, et notamment sur la définition des droites discrètes, leur construction et leur nombre dans un bloc de taille fixe.

1.1.1 Nombre de directions présentes dans le bloc

La première étape consiste à évaluer le nombre de directions d sur lesquelles va être projeté le bloc de taille $(N \times N)$. Ce nombre peut être calculé grâce aux suites de Farey présentées dans le chapître 1.2.
La suite de Farey suivante : $\mathcal{F}_{N-1}$: $(\frac{a_1}{b_1}, \ldots, \frac{a_r}{b_r})$, nous permet d'obtenir l'ensemble des droites discrètes dont les paramètres de direction (a, b) sont tels que :

- $(a < N)$ et $(b < N)$
- (a, b) premiers entre eux
- $(a, b) \in \mathbb{N}^2$
- $\left(\frac{a}{b}\right) \leq 1$

Sous ces conditions, les paramètres (a, b) décrivent des droites dont la direction est comprise dans le premier octant, soit dans l'intervalle : $[0, \pi/4]$. En effet, dans une suite de Farey on a toujours $a \leq b$, ce qui correspond à des paramètres de droites discrètes de directions inférieures à $\frac{\pi}{4}$.
Pour construire les droites de l'intervalle d'angle complet : $[0, \pi]$, il suffit d'appliquer une symétrie par $\frac{\pi}{4}$ dans un premier temps, puis par $\frac{\pi}{2}$ dans un deuxième temps. Ce dernier point est d'ailleurs intéressant au point de vue implémentation car la construction des droites sur l'intervalle d'angle

complet nécessite uniquement le calcul des droites sur un quart de l'intervalle.
Le nombre de directions représentées par des droites discrètes dans l'intervalle complet $[0, \pi]$ est donc égal à quatre fois le nombre de droites comprises dans l'intervalle $[0, \pi/4[$ (premier octant en excluant la borne supérieure pour ne pas compter deux fois la même direction). Nous avions décrit dans le chapître 1.2, la méthode de calcul du nombre d'éléments composant une suite de Farey. Soit $D_{[0,\pi/4[}$ le nombre de directions présentes dans l'intervalle $[0, \pi/4[$. On a :

$$D_{[0,\pi/4[} = Card(\mathcal{F}_{N-1}) - 1 \tag{6.1}$$

Le terme (-1) correspond au retrait de la borne supérieure de l'intervalle. En développant le terme de droite, on obtient :

$$D_{[0,\pi/4[} = \sum_{m=1}^{N-1} \varphi(m) \tag{6.2}$$

En généralisant à tout l'intervalle $[0, \pi]$, on a :

$$D_{[0,\pi]} = 4 \times \sum_{m=1}^{N-1} \varphi(m) \tag{6.3}$$

Le nombre de droites dont les paramètres sont contenus dans le bloc augmente donc (de façon non linéaire) avec la taille du bloc, ce qui a pour effet d'augmenter la résolution angulaire de l'étude directionnelle. Le schéma 6.2 montre le nombre de directions disctinctes dans le premier octant d'un bloc (4×4), avec les couples de paramètres (a, b) correspondant. Il y a ainsi 16 directions détectables dans un bloc de taille (4×4), 72 dans un bloc de taille (8×8) et 288 dans un bloc de taille (16×16).

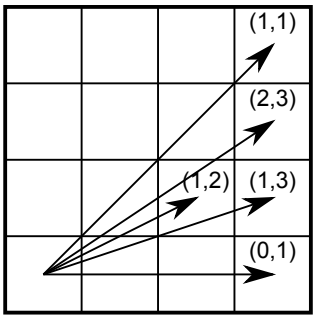


FIGURE 6.2 – Nombre de directions distinctes dans le premier octant d'un bloc (4×4).

Le fait de se limiter aux droites dont les paramètres (a, b) sont contenus dans le bloc $((a < N)$ et $(b < N))$ se justifie par l'application que l'on souhaite faire de cette étude directionnelle. Augmenter la résolution angulaire reviendrait à détecter des angles de droites discrètes dont les paramètres seraient telles qu'au moins un des deux paramètres serait supérieur à la taille de bloc : $(max(a, b)) \geq N$ avec N la taille du bloc. Cela signifierait que les pixels à mettre en relation pour l'interpolation directionnelle (éloignés de (a, b)) seraient trop éloignés du bloc auquel on a attribué cette direction. Nous avons donc fait le choix de limiter la recherche de directions à celles décrites par des paramètres tels que : $(a < N)$ et $(b < N)$. Cela permet de limiter les risques de mise en relation de pixels n'ayant pas la même direction ou appartenant à des objets différents, et ainsi éviter l'apparition de faux pixels dans l'image interpolée.

1.1.2 Principe de la projection

Ce paragraphe explique à partir de la construction de droites discrètes naïves introduite dans le chapître 1.1, comment un bloc entier est projeté le long d'une direction. Nous nous intéresserons

seulement au cas de la construction de droites dans le premier octant, sachant que les autres droites sont obtenues par symétrie.
En pratique, la projection d'un bloc selon une direction θ consiste à projeter tous les pixels de ce bloc sur des droites discrètes naïves de paramètres (a, b, μ) (avec a et b premiers entre eux, $(a \leq b < N)$, $(a, b) \in \mathbb{N}^2$ et $\mu < \sup(a, b)$ et $\mu \in \mathbb{N}$). Dans notre étude, les droites sont construites de sorte à ce que le décalage soit identique pour chaque droite afin d'homogénéiser leurs caractéristiques. Un décalage nul, $\mu = 0$, est choisi. $D_\theta(a, b, 0)$ est donc la droite discrète naïve représentative de la direction $\theta = \tan^{-1} \frac{a}{b}$, qui décalée, permettra la projection du bloc selon la direction θ.
Un ensemble de segments, s_i, est ainsi obtenu pour $i = \{1, \ldots, I_\theta\}$ avec I_θ le nombre total de segments créés pour la direction θ.
Un exemple de projection est montré en Figure 6.3, avec l'illustration des segments obtenus. La flèche à l'intérieur du bloc montre la direction de la projection θ telle que : $\theta = \tan^{-1}\left(\frac{a}{b}\right) = \tan^{-1}\left(\frac{1}{3}\right)$, et les flèches oranges montrent le trajet effectué par l'algorithme, qui suit la droite naïve de paramètres $(a, b) = (1, 3)$. On obtient ici cinq segments de droite représentatifs de la direction dans ce bloc.

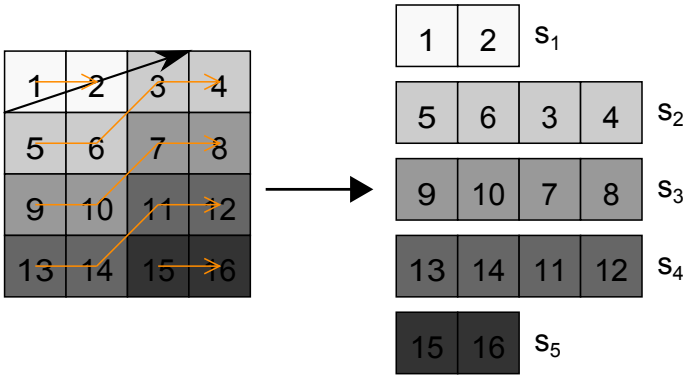


FIGURE 6.3 – Création des segments s_i grâce à la droite discrète naïve de paramètres $(1, 3, 0)$, dans un bloc de taille 4×4.

Ce principe de projection permet d'avoir une transformation bijective puisque tous les pixels du bloc sont présents exactement une fois dans un des segments créés. Cela garantit également la réversibilité de la transformation.

1.2 Avantages des droites naïves et discussion

Cet algorithme de projection d'un bloc le long de droites naïves permet d'une direction à l'autre, d'homogénéiser la distance (au sens de la norme de Chebyshev) entre les pixels successifs des segments. Cette homogénéisation favorise l'isotropie de l'étude. Cela n'aurait pas été le cas si des droites discrète fines non connexes avaient été utilisées pour réaliser la projection. En effet, la distance entre deux éléments successifs de droites non connexes peut varier fortement d'une direction à une autre. C'est en partie ce qui explique le fort biais présent lors de la détection de direction de la méthode utilisée pour la création des bases de bandelettes, que nous étudierons à la fin de cette partie.
Par ailleurs, cet algorithme permet de créer des segments composés de pixels existants de la grille discrète. Aucune interpolation n'est donc nécessaire pour réaliser la projection du bloc, contrairement par exemple à la méthode IRON ou encore à la transformée de Radon.

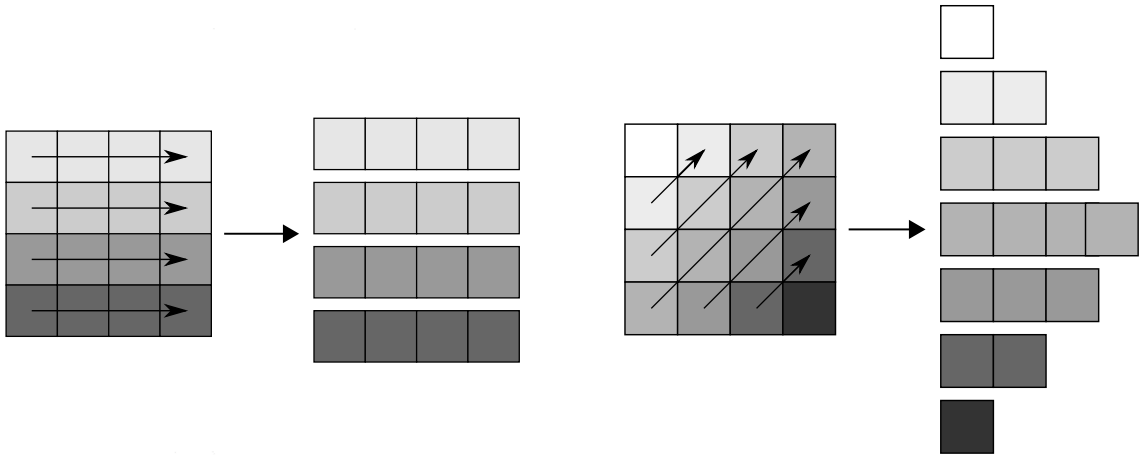
Malgré l'isotropie améliorée par l'utilisation des droites discrètes naïves par rapport aux droites non connexes, nous allons voir qu'il existe toujours d'une direction à l'autre, des différences importantes entre les caractéristiques des segments. En effet, le nombre de pixels par segment ainsi

que le nombre de segments varie selon la direction. Ce problème ne peut être résolu si le support carré (bloc) est conservé. Cette notion primordiale de support va donc être abordée dans la partie suivante.

2 Support de l'analyse directionnelle

2.1 Disparité des propriétés des segments

Pour illustrer l'importance de la notion de support, prenons deux exemples extrêmes. Lorsque le bloc est projeté dans la direction horizontale, tous les segments créés ont la même longueur (voir schéma 6.4 (a)). Dans le d'une projection à 45° (voir schéma 6.4 (b)), la longueur des segments créés ainsi que le nombre de segments varient. Dans le but d'étudier les variations sur les segments s_i, il paraît évident que les segments courts apportent peu voire aucune information quant à une certaine régularité le long de la direction qu'ils décrivent. Il est donc nécessaire de tendre vers une homogénéisation des caractéristiques des segments (nombre total et longueur), pour toutes les directions afin d'améliorer l'isotropie et la précision de l'étude.



(a) - Segments créés pour la direction 0° (b) - Segments créés pour la direction 45°

FIGURE 6.4 – Création de segments pour différentes directions par projection sur des droites naïves.

2.2 Solution de la méthode IRON : le support rectangulaire

Pour limiter l'anisotropie et donc éviter le biais qui favoriserait la détection d'une direction par rapport à une autre (voir partie sur l'étude du biais 7.1), il est préférable de modifier le support carré. Ceci est réalisé dans la méthode IRON ([Da Costa 2002], [Michelet 2004]) où le support est modifié en fonction de la direction à étudier. En effet, pour l'étude d'une direction particulière, le support est composé de L lignes parallèles à la direction d'étude, chacune composée de P points dépendant de la taille du support voulu (voir Figure 6.5). Notons que l'union de tous les supports est contenue dans un cercle centré sur le pixel d'étude et de diamètre $S = \sqrt{(P-1)^2 + (L-1)^2}$. Dans cet exemple, le nombre de lignes est égal à $L = 3$ et il y a $P = 5$ points par segment. Notons que dans cette méthode, la projection des pixels sur les lignes se fait par interpolation puisque les lignes créées ne correspondent pas à des pixels de la grille discrète. Le choix de nombres impairs permet d'assurer la symétrie du support par rapport au pixel étudié. Le support créé est volontairement allongé dans le sens de la direction à étudier de façon à pouvoir à la fois augmenter la résolution angulaire autour de l'angle étudié, et être plus précis sur l'angle estimé.

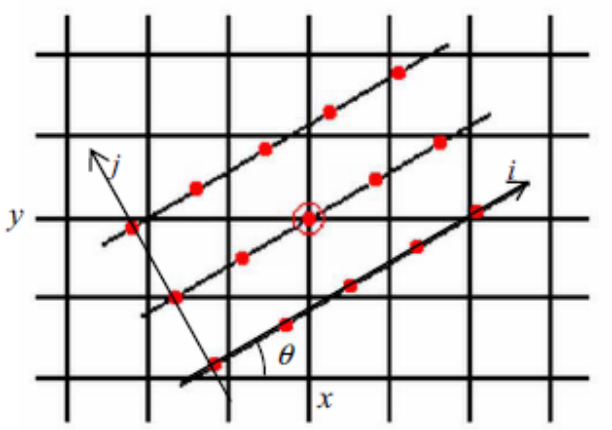


FIGURE 6.5 – Support de la méthode IRON pour la recherche de direction pour $L = 3$ et $P = 5$. Schéma tiré de [Michelet 2007].

2.3 Critique du support rectangulaire

Bien qu'il permette une très bonne discrimination angulaire, le support rectangulaire n'est pas adapté à notre étude. En effet, le fait de choisir un support rectangulaire allongé dans le sens de la direction à étudier implique forcément une forte disparité entre les supports créés pour l'étude de différentes directions. Ceci est illustré en Figure 6.6, où il est évident de constater qu'une proportion importante de pixels sur lesquels sont basés l'estimation appartiennent exclusivement à l'un des deux supports. Cet exemple peut se généraliser pour d'autres directions.
Lorsque ce procédé est appliqué à des images, il arrive que certains supports contiennent des contours d'objets que d'autres supports ne contiennent pas. Il est donc peu fiable d'estimer la direction du bloc en se basant sur des supports aussi disjoints, qui potentiellement contiennent des objets différents.

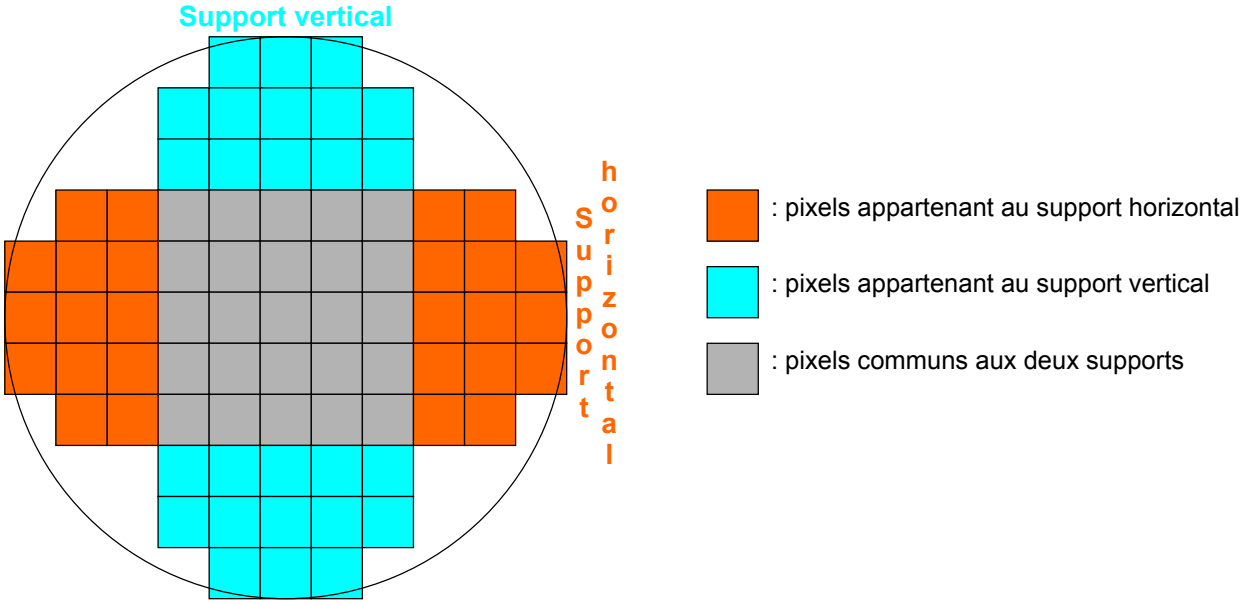


FIGURE 6.6 – Illustration de deux supports rectangulaires contenus dans un même cercle. Le support vertical (bleu) et le support horizontal (orange) ont une faible proportion de pixels en commun.

2.4 Support circulaire

Compte tenu des précédentes remarques, nous avons cherché un support ayant les caractéristiques suivantes :

- Homogénéité dans le nombre et la longueur des segments créés
- Support commun pour toutes les directions
- Support proche du bloc pour lequel la détection de direction est à faire

Le support qui rempli ces conditions est donc un support circulaire. Ce dernier permet en effet de rendre l'étude la plus isotrope possible, de façon à ne favoriser aucune direction. Soit S ce support, défini autour du bloc de taille $c \times c$, et de pixel central : (x_c, y_c). Le pixel de coordonnées $(x, y) \in \mathbb{Z}^2$, appartient donc au support S s'il rempli la condition suivante :

$$\sqrt{(x - x_c)^2 - (y - y_c)^2} < c \times \sqrt{2} \tag{6.4}$$

Autrement dit, ce support est le plus petit cercle contenant tous les pixels du bloc. Comme souhaité, il est restreint au niveau spatial, isotrope et permet la création de segments relativement homogènes pour toutes les directions. Une illustration du support circulaire est montrée en Figure 6.7.

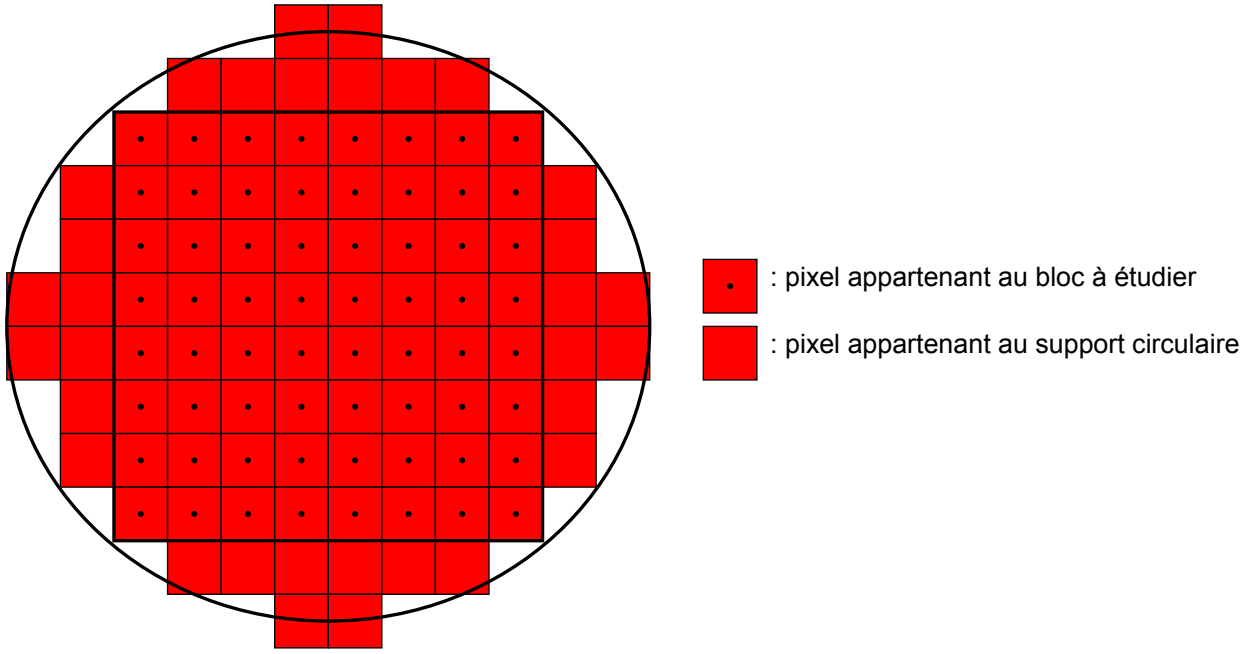
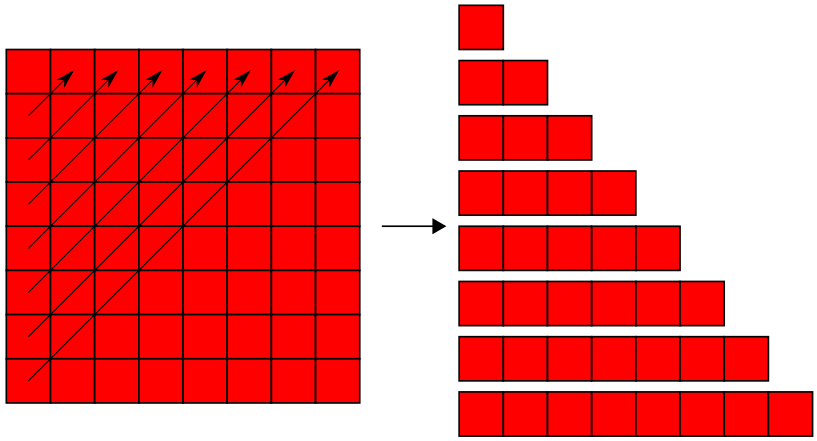


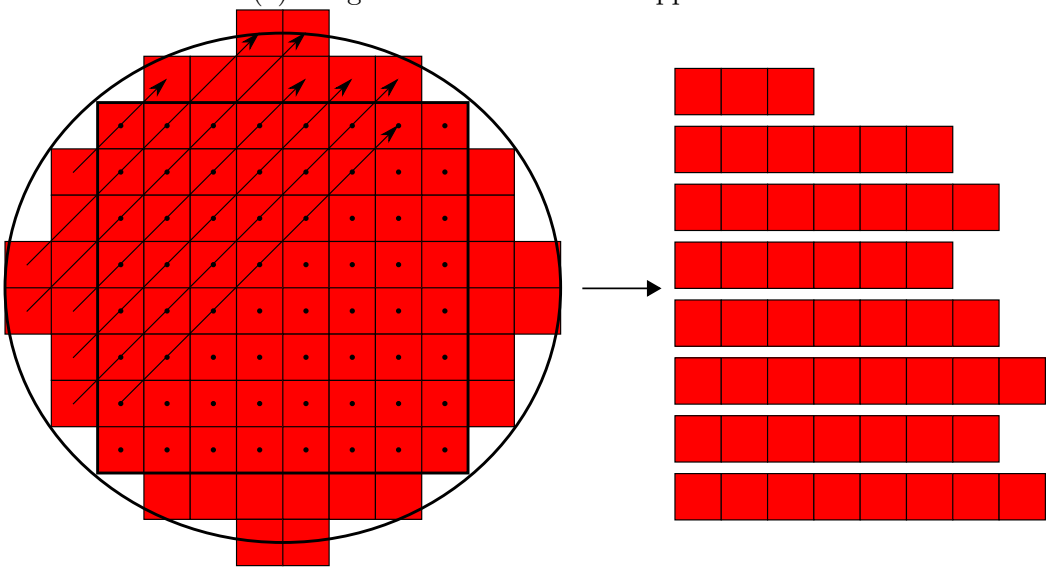
FIGURE 6.7 – Illustration du support circulaire pour un bloc de taille 8×8.

La comparaison entre les segments donnés par le support carré et le support circulaire est montrée en Figure 6.8. Il est facile de voir que la longueur des segments créés est beaucoup plus homogène que dans le support carré. Outre le fait d'obtenir un support compact et équivalent pour toutes les directions, ce support permet d'avoir un nombre et une longueur de segments homogènes mais aussi de mieux capter les directions des contours situés dans les coins du bloc. En effet dans le cas d'un support carré, seuls quelques pixels contiendront l'information de régularité le long de cette direction. A l'inverse dans un support circulaire, les segments créés dans les coins du bloc ont une longueur comparable aux autres segments et permettent de prendre une décision plus robuste et plus fiable sur la direction à estimer.

L'étude directionnelle utilisée par la suite sera donc réalisée sur des supports circulaires, et nous mesurerons l'apport d'un tel support par rapport à un support carré dans le chapître 7.1. Nous verrons que le surplus de calcul dû au nombre de pixels plus importants dans le support circulaire se justifie pleinement à la vue de l'amélioration des résultats de détection.



(a) - Segments créés dans un support carré.



(b) - Segments créés dans un support circulaire.

FIGURE 6.8 – Comparaison entre les segments créés pour la direction 45°, par les supports carrés et circulaires dans la première moitié d'un bloc de taille 8×8.

A partir de la création des segments provenant des projections du bloc, les variations de chaque projection doivent être calculées afin de décider de la direction dominante présente à l'intérieur du bloc. La partie suivante explique comment ce calcul est effectué.

3 Calcul de la quantité de variations

Cette partie présente la manière dont est calculée la quantité de variations des segments créés pour chaque direction. La finalité de cette étude consiste à trouver la direction qui minimise cette quantité de variations. Cette dernière sera considérée comme parallèle au contour du bloc.
Le calcul se fait de manière assez simple et d'une manière assez similaire à la construction du critère de choix de résolution. Soit V_i la variation du i-ème segment résultant de la projection du bloc dans la direction θ. V_i est égal à l'amplitude des coefficients de ce segment :

$$V_i(\theta) = \sup(s_i(\theta)) - \inf(s_i(\theta)) \tag{6.5}$$

La variation globale V_{tot} de la direction θ est calculée en moyennant les variations de chaque segment :

$$V_{tot}(\theta) = \frac{\sum_{i=1}^{I_\theta} V_i}{I_\theta} \tag{6.6}$$

V_{tot} est donc calculée pour toutes les directions du bloc. La direction dominante du bloc sera donc définie par les paramètres discrets de la droite naïve pour laquelle $V_{tot}(\theta)$ est minimale.

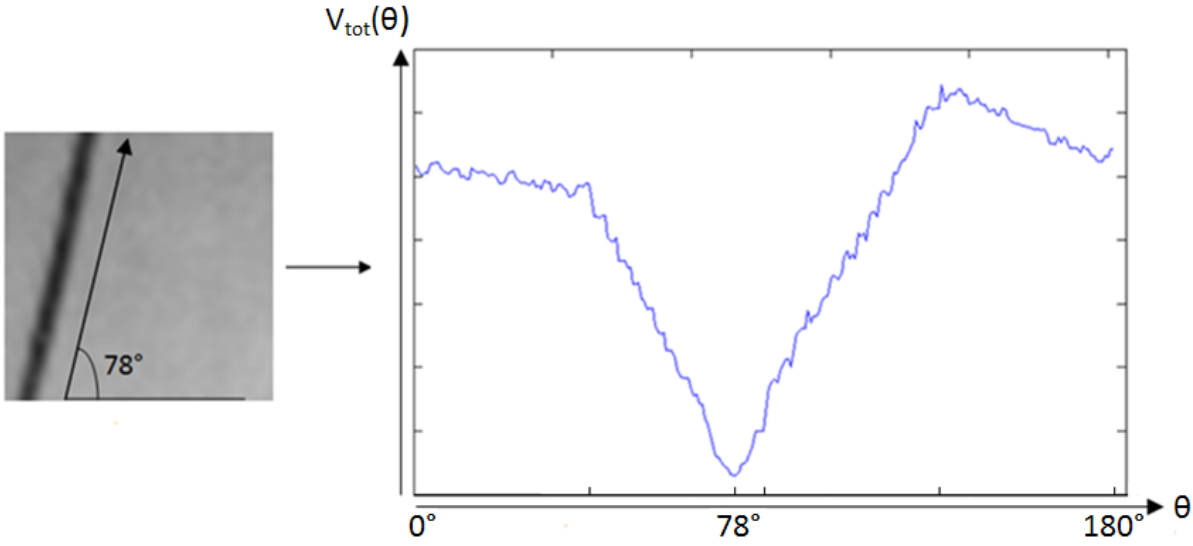


FIGURE 6.9 – Tracé des valeurs de $V_{tot}(\theta)$ pour un bloc contenant un contour oblique mesuré à la main à 78°.

Un tracé de $V_{tot}(\theta)$ pour toutes les directions présentes dans un bloc contenant une direction franche est montré en Figure 6.9. Dans cet exemple, nous observons un minimum correspondant à la droite naïve le long de laquelle les variations sont minimales. Nous verrons dans la partie suivante qu'une droite naïve peut ne pas correspondre à une valeur d'angle unique. En effet, plusieurs angles peuvent avoir la même droite discrète naïve, selon le décalage utilisé lors de la discrétisation de la droite.

4 Incertitude sur l'angle estimé

Les droites naïves sur lesquelles sont projetés les blocs dépendent de trois paramètres : a, b définissent l'orientation de la droite et μ le décalage. Par soucis d'homogénéité, l'algorithme de projection impose un décalage nul pour toutes les directions. Or si la valeur du paramètre μ est changée, les droites construites sont décalées. Il arrive donc que deux droites naïves, $D_1(a_1, b_1, \mu_1)$ représentative de la direction θ_1, et $D_2(a_2, b_2, \mu_2)$ représentative de la direction θ_2, soient identiques pour $(a_1 \neq a_2)$ et/ou $(b_1 \neq b_2)$ pour des valeurs de décalages différentes $(\mu_1 \neq \mu_2)$ (avec $(a_1, a_2, b_1, b_2) \leq N$ et (a_1, b_1) et (a_2, b_2) premiers entre eux). Ce cas se produit évidemment seulement lorsque θ_1 et θ_2 sont suffisamment proches. Une droite naïve peut représenter plusieurs directions.
C'est ce que nous appelons le cône d'incertitude d'une droite. Il est défini pour chaque droite et est déterminé en amont de l'analyse directionnelle, ce qui a pour avantage de ne pas alourdir les calculs lors de l'estimation directionnelle. Ce cône regroupe tous les couples (a, b) qui décrivent la même droite naïve pour au moins une valeur de décalage. Il permet ainsi d'avoir une vision exhaustive de tous les angles contenus dans la droite naïve qui a les variations les plus faibles.

Cette information sera utilisée au moment de l'interpolation du bloc, où seule une de ces directions sera choisie.
La figure 6.10 montre le nombre d'angles contenus dans chacune des droites discrètes créées dans un bloc de taille 8×8 pixels. Les valeurs représentées en abscisse de cette courbe correspondent chacune à une droite discrète (88 droites discrètes au total créées dans un bloc de taille 8×8 pixels). Cette courbe montre que certaines droites ne représentent qu'un seul angle (c'est le cas des directions 0°, 45°, 90° et 135°), alors que d'autres droites sont la représentation discrète de six angles différentes. C'est le cas en particulier pour les droites dont la valeur maximale de (a, b) est grande (grands nombre de décalages possibles).

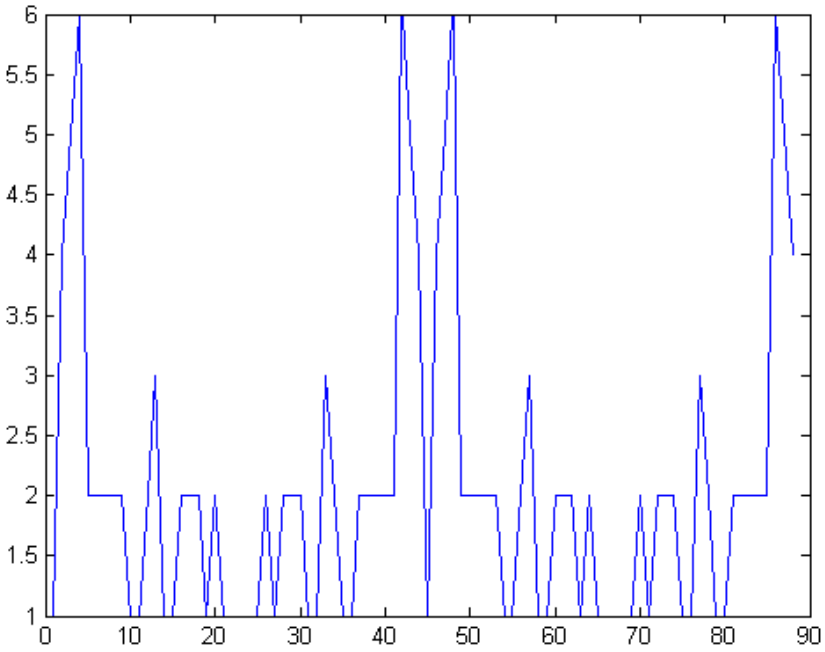


FIGURE 6.10 – Nombre d'angles compris dans chaque droite naïve d'un bloc 8×8. En abscisse, le numéro de la droite discrète représentant des angles de 0° à 180°. En ordonnée, le nombre d'angles compris dans la droite correspondante.

Le procédé de recherche de directions qui vient d'être décrit permet de détecter de façon précise la direction du contour présent dans le bloc. Il peut évidemment arriver que plusieurs directions de contours différentes soient présentes à l'intérieur d'un même bloc. Il est donc nécessaire d'introduire dans l'étude directionnelle un critère qui décide quant à la présence d'une ou plusieurs directions à l'intérieur d'un même bloc. La partie suivante décrit la construction de ce critère, ainsi que l'algorithme de découpage en *quadtree* utilisé pour diviser, si nécessaire, le bloc parent en sous-blocs.

5 Découpage en *Quadtree*

Pour adresser le problème de la présence de plusieurs directions de contours à l'intérieur d'un bloc, la méthode de division en *quadtree* est utilisée. Le *quadtree* est une méthode de division courante qui consiste à analyser l'homogénéité d'un bloc parent selon un certain critère, et à le diviser en quatre sous-blocs de taille égale si celui-ci n'est pas considéré comme homogène. Le procédé est ensuite réitéré dans chaque sous-blocs créé jusqu'à obtenir des blocs homogènes ou

atteindre une taille de bloc minimale.
Ce principe s'applique particulièrement bien à notre étude puisqu'il autorise d'abord l'estimation de la direction du contour dans un grand voisinage, c'est-à-dire avec une résolution angulaire optimale. La diminution progressive de la taille des blocs dans le cas où le bloc contient plusieurs contours permet de gagner en précision spatiale.

5.1 Critère d'homogénéité

Dans notre cas, le critère d'homogénéité qui décide de la division ou non du bloc, est la présence de plusieurs directions de contours à l'intérieur du bloc parent. Pour savoir si une direction unique est présente dans le bloc, on étudie la valeur définie précédemment : $V_{tot}(\theta_{bloc})$, qui correspond aux variations des segments le long de la direction prédominante associée au bloc, θ_{bloc}. Si $V_{tot}(\theta_{bloc})$ est faible, il y a de fortes chances pour que la direction de contour soit unique puisque les segments créés ne traversent a priori pas de contours ayant une autre direction que θ_{bloc}. Cependant, il est difficile d'appliquer un seuil fixe qui conviendrait à tous types de contenu, pour déduire de $V_{tot}(\theta_{bloc})$ l'unicité de la direction de contour détectée. C'est pourquoi notre critère d'homogénéité est construit de manière relative aux variations internes du bloc. Ainsi, si la valeur $V_{tot}(\theta_{bloc})$ est suffisamment faible par rapport à une valeur de variation de référence du bloc, le bloc n'est pas divisé et la direction détectée θ_{bloc}, est considérée comme unique. Les parties suivantes décrivent dans un premier temps la construction de la valeur de variation de référence du bloc, puis la détermination du critère d'homogénéité associé.

5.1.1 Définition de la variation de référence

La variation de référence notée V_{ref}, est la moyenne des amplitudes des coefficients du bloc dans la direction horizontale et des amplitudes des coefficients du bloc dans la direction verticale.

$$V_{ref} = \frac{V_{0^\circ} + V_{90^\circ}}{2} \tag{6.7}$$

Les valeurs V_{0° et V_{90° sont calculées sur le même principe que pour le choix de résolution présenté dans le chapitre 5. Elles correspondent respectivement aux moyennes des amplitudes des coefficients d'ondelettes sur les lignes et sur les colonnes du bloc. Ici les directions horizontale et verticale sont utilisées comme référence par simplicité de construction (calcul sur les lignes et les colonnes). Le principe serait identique en moyennant les variations d'une direction quelconque et de sa perpendiculaire.

5.1.2 Construction du critère d'homogénéité

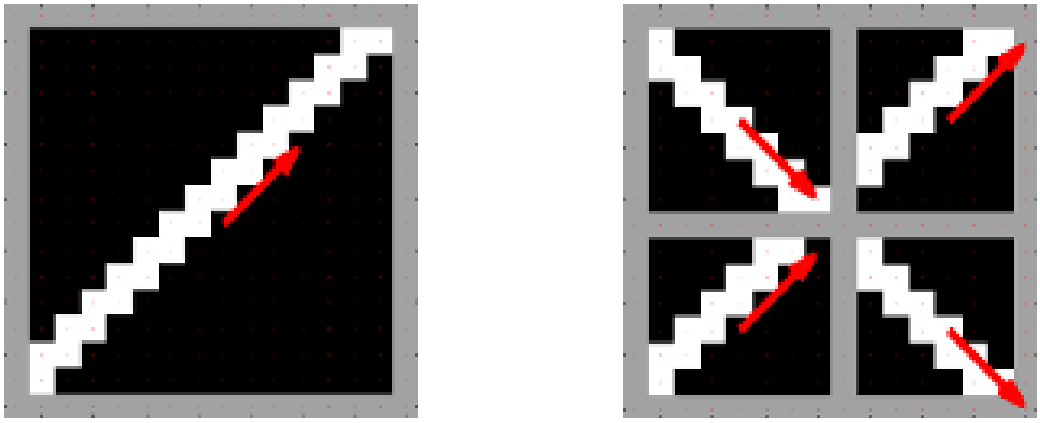
Le critère d'homogénéité est construit en comparant $V_{tot}(\theta_{bloc})$ avec la valeur de variation de référence V_{ref}. Un facteur de comparaison α est introduit pour la définition du critère d'homogénéité, noté C. Le facteur de comparaison α permet donc de définir le critère tel que :

$$C = V_{ref} \times \alpha \tag{6.8}$$

Le facteur α a été fixé empiriquement à 1/2 et permet d'obtenir des résultats satisfaisants pour tous les types d'images testées.
Si $V_{tot}(\theta_{bloc}) \leq C$, c'est à dire si les variations le long de θ_{bloc} sont inférieures à la moitié des variations de référence, le bloc n'est pas divisé. A l'inverse, si $V_{tot}(\theta_{bloc}) > C$ alors les coefficients le long de θ_{bloc} ne sont pas assez homogènes pour que θ_{bloc} soit considérée comme direction unique à l'intérieur du bloc. Il y a donc division du bloc dans ce cas.
Remarquons ici que la construction du critère qui décide de la division d'un bloc est rendu adaptatif au contenu du bloc grâce à l'introduction de la valeur de variation de référence. Le critère sera donc efficace pour tous types d'images et ne nécessite pas le réglage de paramètres supplémentaires.

5.1.3 Division du bloc

Dans le cas ou une division du bloc de taille $(N \times N)$ est nécessaire, ce dernier est divisé en quatre sous-blocs de taille identique, à savoir $(N/2 \times N/2)$. La recherche de direction prédominante ainsi que le test d'homogénéité sont ensuite itérés dans chaque sous-bloc pour permettre des éventuelles nouvelles sub-divisions. Le procédé est itéré jusqu'à ce que chaque bloc contienne au plus une direction de contour, ou jusqu'à ce que la taille minimale de bloc admise soit atteinte. Notons que les tailles de blocs doivent être des multiples de 2^n pour que les divisions puissent donner des blocs de tailles entières. Ceci justifie l'utilisation des tailles de blocs qui sont utilisées tout au long de notre étude directionnelle, à savoir (16×16), (8×8) et (16×16) pixels. Un exemple de division sur des images de synthèse est montré en Figure 6.11.



(a) - Direction unique : pas de division (b) - Plusieurs directions : division du bloc

FIGURE 6.11 – Exemple de division dans le cas d'une direction unique et dans un cas à plusieurs directions.

5.2 Détails d'implémentation

5.2.1 Tailles de blocs

Dans notre cas, le plus petit bloc autorisé est un bloc de taille (4×4) pixels. En dessous de cette taille, la résolution angulaire devient trop faible pour être exploitable et porteuse d'information dans une optique d'interpolation. A l'inverse, le plus grand bloc autorisé est un bloc de taille (16×16). Au dessus de cette taille, la résolution angulaire devient inutilement grande (de l'ordre du dixième de degré). Le nombre de droites que l'on peut créer est en effet très important, et les calculs sur la totalité des droites créées seraient trop lourds. De plus, détecter des directions dans des grands blocs signifie que l'on peut potentiellement filtrer des pixels de plus en plus éloignés lors de l'interpolation. Le risque de filtrer des pixels appartenant à des objets de l'image différents est alors trop important pour être pris. Notre étude est donc limitée à trois tailles de bloc : (4×4), (8×8) et (16×16).
D'autre part, la taille de bloc maximale doit correspondre à la taille de bloc choisie pour l'étude de résolution. De cette manière, l'analyse directionnelle se fera en se basant sur des coefficients d'ondelettes de même échelle. Rappelons que la taille choisie pour l'étude de résolution était jugée optimale pour des blocs de (16×16) pixels. L'utilisation des blocs (16×16) pixels comme blocs de taille maximale est donc cohérente avec le reste de l'étude directionnelle.

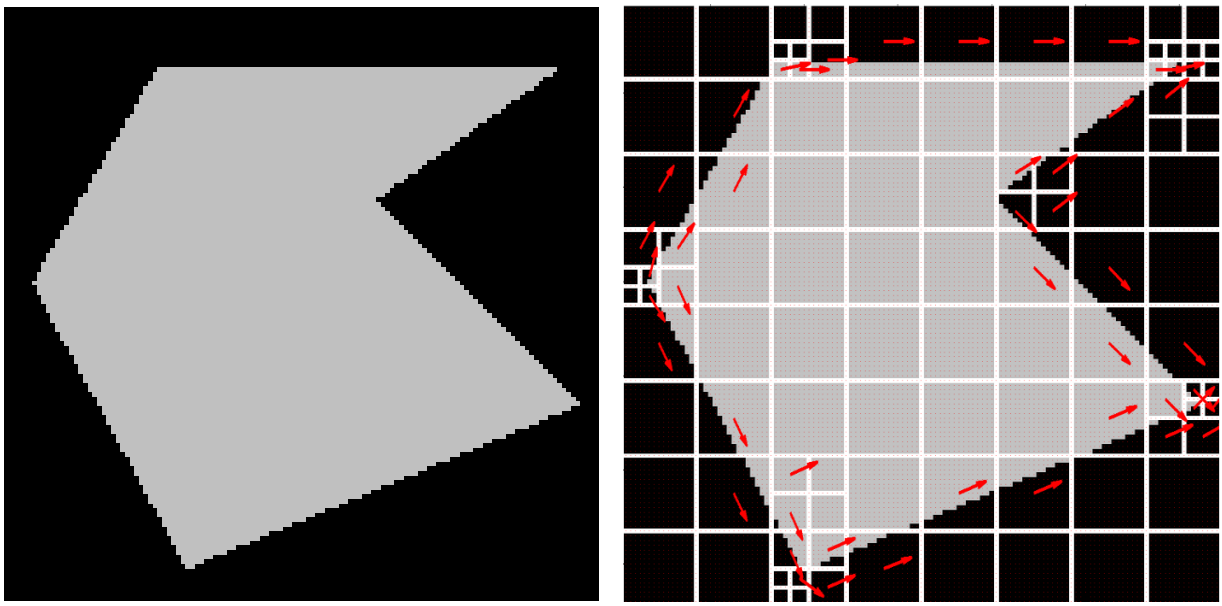
5.2.2 Cas particulier du bloc (4×4)

Le plus petit bloc est traité d'une manière particulière puisque sa résolution angulaire est assez faible. Comme expliqué dans le chapitre 6.1.1, les droites discrètes créées pour la projection du

bloc vers des segments $1D$ ont un décalage nul, $\mu = 0$. Or pour les blocs 4×4, tous les décalages sont pris en compte de façon à tracer toutes les droites discrètes possibles présentes dans le bloc. Cela permet d'introduire de la redondance dans la détection puisqu'une même direction sera représentée par plusieurs droites discrètes, et permet d'augmenter la chance de détecter la direction du contour. Cette particularité est intéressante pour les blocs de cette taille puisque seuls peu de pixels permettent de différencier deux directions voisines. Il est donc primordial de capter l'enchaînement de pixels qui permet de détecter la direction du contour.
Ceci augmente le nombre de droites à calculer mais ce dernier reste raisonnable pour des blocs de cette taille. Dans les cas des autres tailles de bloc, la résolution angulaire est suffisamment bonne pour que seules les droites avec un décalage nul soient testées. De plus, les calculs auraient été bien trop lourds si tous les décalages avaient été pris en compte. Nous verrons dans le chapitre 7.1 que cette particularité des blocs (4×4) améliore considérablement les capacités de détection de notre méthode.

5.3 Exemple de division

Un exemple de découpage d'une image de synthèse est montré en Figure 6.12. Par soucis de clarté, le découpage est montré sur l'image originale mais il est important de rappeler que le découpage se fait en réalité sur les coefficients de la transformées en ondelettes IUWT de l'image. Il est possible d'observer dans cet exemple que chaque bloc contient au plus une seule direction de contour, et que le découpage devient plus fin aux intersections de contours (coins du polygone). Le découpage obtenu est à mettre en relation avec les directions détectées puisque la division dépend de la direction détectée. Le *quadtree* peut être donc vu comme une sécurité vis à vis de la détection de direction puisqu'en cas de mauvaise détection, le bloc est découpé et une nouvelle étude directionnelle a lieu. La Figure 6.12(b) montre donc les directions associées au découpage à l'intérieur de chaque bloc.



(a) - Image à analyser (b) - Résultat de l'analyse directionnelle

FIGURE 6.12 – Résultat du découpage en *quadtree* de l'image de gauche, et directions des contours détectés.

6 Conclusion sur l'analyse directionnelle

Ce chapitre nous a permis de présenter notre technique de détection de direction, basée sur l'adaptation préalable de résolution qui avait pour but de représenter le contenu des blocs créés dans l'espace d'ondelette de résolution optimale. L'estimation de directions des contours s'effectue donc dans un premier temps à l'intérieur de chaque bloc créé par l'analyse de résolution, à savoir des blocs de (16×16) pixels.

Afin de trouver la direction prédominante de chaque bloc, ce dernier est projeté le long des droites naïves (8-connexes), dont les paramètres de direction (a, b) sont inférieurs à la taille du bloc. Ainsi, plus la taille du bloc augmente, plus le nombre de droites sur lesquelles est projeté le bloc est important, et meilleure est la résolution angulaire de l'étude. L'étude des variations des projections du bloc selon toutes les directions permet de détecter la droite possédant les variations minimales et d'en déduire la direction de contour associée.

Dans le but d'optimiser l'isotropie de l'étude les projections sont appliquées sur un support circulaire, défini comme le plus petit disque entourant le bloc. L'utilisation d'un tel support permet d'homogénéiser la taille et le nombre des segments construits lors de la projection, et donc de limiter l'apparition de biais qui favoriserait la détection de certaines directions par rapport à d'autres. L'apport d'un tel support est quantifié dans le chapitre suivant, qui s'intéresse à évaluer notre méthode d'estimation de direction.

Le critère d'homogénéité permet, à partir de la comparaison des variations de la projection dans la direction détectée avec les variations de référence du bloc, de détecter si le bloc contient plus d'une direction de contour. Dans un tel cas, une division en *quadtree* du bloc original est effectuée afin d'isoler de façon récursive au plus une direction de contour par sous-bloc. L'analyse directionnelle est donc répétée à l'intérieur de chaque sous-bloc afin d'en calculer également la direction prédominante, ou de les diviser à leur tour.
Le principe de division en *quadtree* est particulièrement bien adapté à notre étude directionnelle puisqu'il permet de rechercher d'abord une direction de contour dans le plus grand bloc ((16×16) pixels) et donc avec une résolution angulaire maximale, et de le diviser uniquement si nécessaire. De plus, si la direction à l'intérieur du grand bloc est considérée comme unique, cette taille de bloc ainsi que la direction associée sont conservés et aucun calcul supplémentaire n'est nécessaire. A l'inverse, la méthode utilisée pour la construction des bases de bandelettes utilise le fonctionnement opposé de l'algorithme de *quadtree*. Les directions sont tout d'abord estimées dans les plus petits blocs ((4×4) pixels) qui sont ensuite regroupés si la fusion de ces blocs entraîne la création d'un bloc parent plus homogène. Non seulement cette méthode nécessite l'estimation de direction dans au moins deux tailles de blocs différentes, mais elle estime en premier lieu les directions de contour dans des petits blocs, et donc avec une résolution angulaire minimale.

Enfin, le chapitre suivant consiste à évaluer notre méthode d'estimation de direction décrite jusqu'à maintenant. Plus précisément l'évaluation porte sur la détection d'orientation en elle-même, c'est-à-dire la méthode de projection du bloc sur des droites naïves et la manière dont les variations sont calculées. Les directions estimées seront comparées à la valeur réelle (connue) des contours, ainsi qu'à la méthode de la BT (Bandlet Toolbox) et à la transformée de Radon, pour plusieurs tailles de supports, plusieurs fréquences de contours et plusieurs types de dégradation de l'image originale (bruit blanc et bruit de compression). Nous justifierons alors les choix réalisés dans notre approche, et nous discuterons les avantages et inconvénients de chaque méthode. Notons que cette méthode a été publiée dans une conférence internationale [Van Reeth 2010b], et a donné lieu au dépôt d'un brevet (Europe et USA) en 2010 [Van Reeth 2010a].

Chapitre 7

Evaluation de la détection de direction de contours

1 Procédé d'évaluation

Une manière objective de juger la qualité d'une méthode de détection de direction est de quantifier le biais de détection. Nous appelons biais de détection la différence en valeur absolue entre l'angle détecté et l'angle réel des contours de l'image de test. Nous avons procédé à des tests sur le biais engendré par les méthodes qui définissent une direction **par bloc** afin de pouvoir comparer les résultats avec notre méthode. Les méthodes évaluées sont donc :

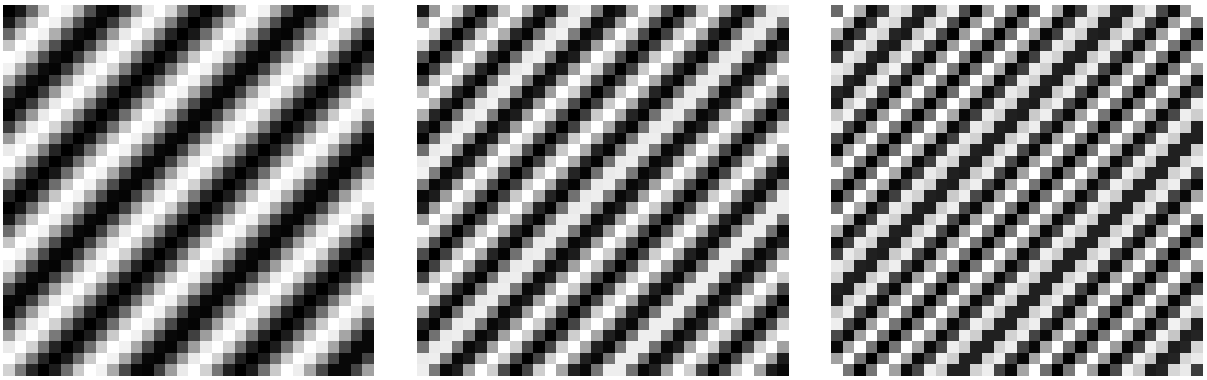
- La méthode de détection utilisant la transformée de Radon présentée dans le chapitre 4.3.
- La méthode utilisée dans la construction des bases de bandelettes présentée dans le chapitre 4.4, telle qu'elle est décrite dans la Bandlet Toolbox (BT) [Peyre 2007].
- Notre méthode de détection de direction. Notons que l'optimisation de résolution introduite en partie 5 n'est pas appliquée en amont de cette étude, afin de comparer uniquement les méthodes d'estimation de direction entre elles indépendamment de tout autre traitement.

1.1 Images de tests

Plusieurs séries de tests sont effectués dans cette partie pour évaluer chaque méthode dans différentes conditions. Les premiers tests sont effectués sur des images de synthèse contenant des contours générés par le tracé de sinusoïdes. Plusieurs fréquences de contours sont utilisées afin de tester les méthodes de détection dans plusieurs situations : f_1 est la fréquence la plus basse des trois, f_2 est la fréquence correspondant à la fréquence de Nyquist (pas d'aliasing), et f_3 est supérieure à la fréquence de Nyquist. Dans ce dernier cas, la grille d'échantillonnage n'est pas suffisamment dense pour représenter les variations des sinusoïdes. La Figure 7.1 présente un exemple de sinusoïdes orientées à 45°, pour les trois fréquences testées. Les différentes évaluations sont effectuées pour les trois tailles de supports utilisées dans la division en *quadtree*, à savoir des supports circulaires de taille 24×24, 12×12 et 6×6 pixels afin d'évaluer les performances de notre méthode dans le cadre précis de notre application.

1.2 Série de tests

Le premier test consiste à calculer le biais sur les images de synthèse contenant les sinusoïdes orientées telles quelles. Ce cas théorique va nous permettre d'évaluer chaque méthode dans le cas de contours idéaux puisque parfaitement rectilignes et non corrompus. Ce test va également nous permettre d'étudier l'impact de l'utilisation d'un support circulaire par rapport au support carré. Pour cela, les mêmes images de test sont utilisées afin de comparer les biais du support circulaire avec ceux obtenus en prenant des blocs carrés de taille 16×16, 8×8 et 4×4 pixels.



(a) (b) (c)

FIGURE 7.1 – (a) Sinusoïde orientée à 45° à la fréquence f_1. (b) Sinusoïde à 45° à la fréquence f_2. (c) Sinusoïde à 45° à la fréquence f_3.

La deuxième série de test consiste à ajouter aux mêmes images de synthèse un bruit blanc d'amplitude croissante. Nous voulons ainsi tester la robustesse de chaque méthode à ce type de dégradation. Ceci représente une première étape vers l'application à des images naturelles. Dans cette même optique, les images de tests sont compressées en jpeg de façon à créer des effets de blocs visibles et étudier leur impact sur les trois méthodes de détection de direction.
Enfin, les trois méthodes sont comparées sur des extraits d'images naturelles illustrant des situations bien particulières pour lesquelles nous avons remarqué des comportements intéressants. Nous essaierons de déduire de ces exemples des comportements généraux pour chacune des méthodes afin de faire ressortir celle qui paraît le plus adapté à l'analyse de situations réelles.

1.2.1 Nombre de directions testées

La résolution des méthodes de détection de directions dans un voisinage borné étant finie, il n'est possible de détecter qu'un certain nombre d'angles. Dans les tests effectués, le choix de l'orientation des sinusoïdes est effectué de sorte à ce que chaque direction soit détectable par la méthode analysée. Pour la méthode des bandelettes et notre méthode, les directions testées sont celles qui sont représentées par des droites dont les paramètres discrets de direction (a, b) sont inférieurs à la taille du support testé. Cela signifie que le nombre de directions à tester varie en fonction de la taille du bloc analysé. Pour la méthode de Radon, l'approche est différente puisque la résolution angulaire ne dépend pas de la taille du bloc analysé. Dans ce test, la résolution angulaire de la méthode de Radon est fixée à 1 degré (soit 0°, 1°, 2°, ...). Les angles testés sont donc, par soucis d'homogénéité, les mêmes que pour les deux méthodes précédentes. Dans le cas ou ces angles n'auraient pas des valeurs entières, ils sont arrondis à l'entier le plus proche afin qu'ils soient détectables par la méthode de Radon.

2 Test sur les images de synthèse

La Figure 7.2 illustre les biais obtenus pour les méthodes de la Bandlet Toolbox (BT), Radon et notre méthode pour les trois tailles de supports circulaires, et pour les trois fréquences de contours testées. Les tailles de support indiquées dans le graphique correspondent au diamètre du cercle circonscrit au bloc à étudier. De même, la Figure 7.3 illustre les biais obtenus pour la même étude réalisée dans des supports carré. Les tailles de support indiquées correspondent cette fois-ci aux tailles des blocs de la division en *quadtree*.
La valeur du biais indiqué dans les deux graphiques correspond à la moyenne des biais obtenus pour tous les angles testés. Enfin, pour une taille de support donnée, les trois valeurs de biais indiquées

pour chaque méthode correspondent aux trois fréquences de contours testées, soit respectivement de gauche à droite f_1, f_2 et f_3.

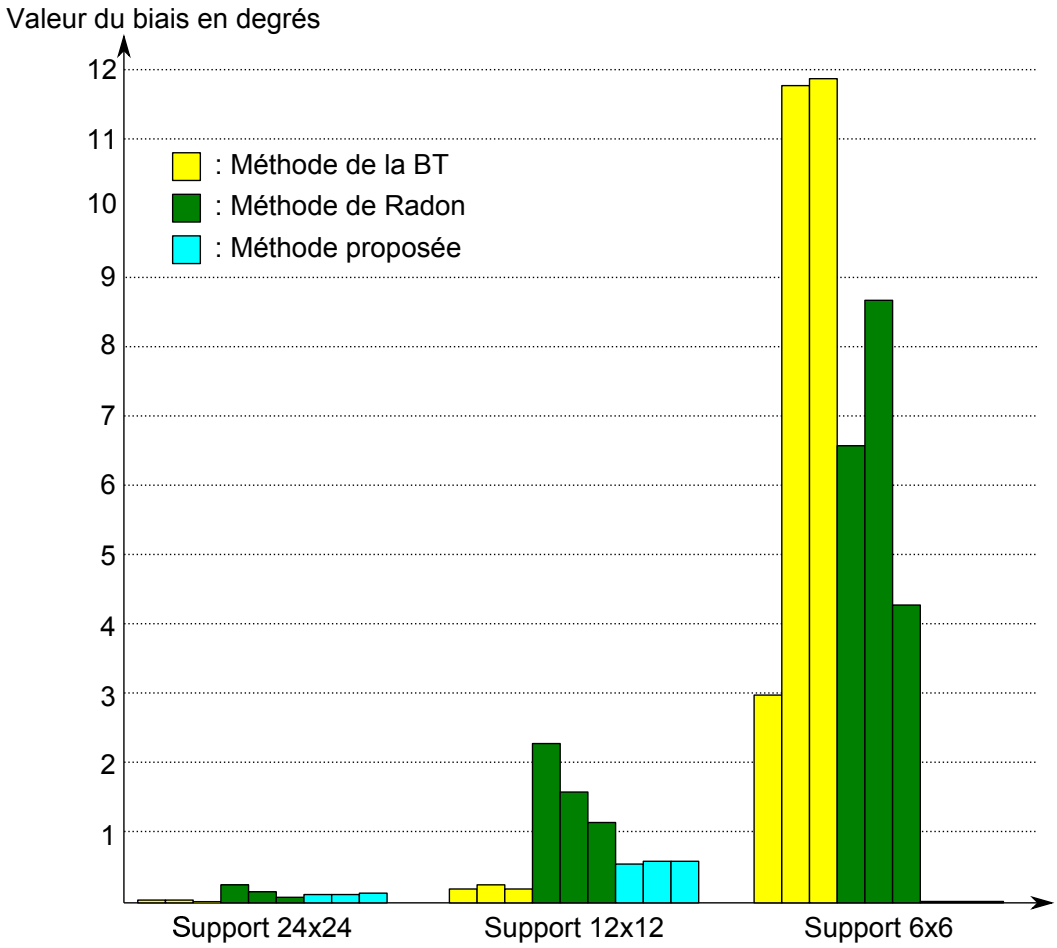


FIGURE 7.2 – Biais obtenus sur les images de sinusoïdes en fonction de la taille du support **circulaire** utilisée.

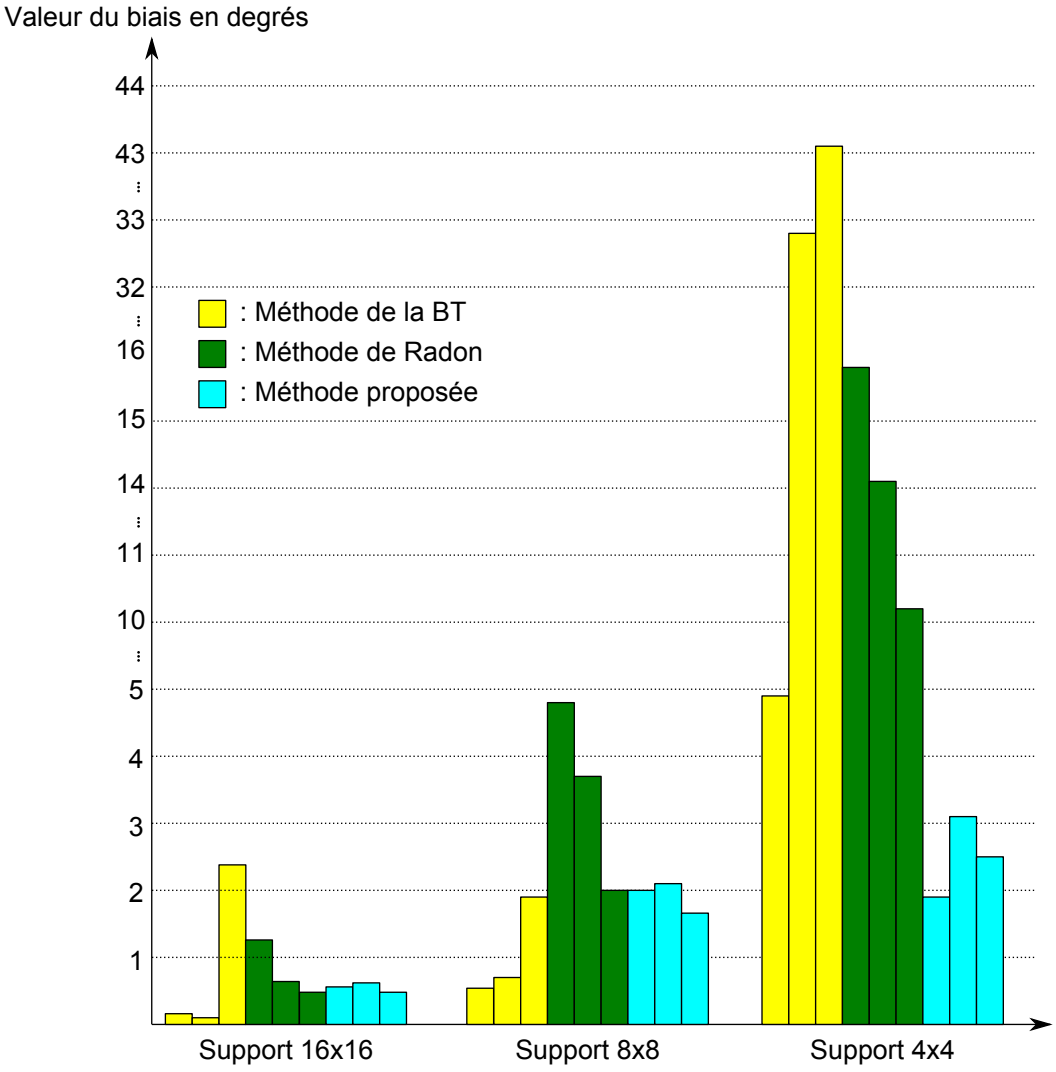


FIGURE 7.3 – Biais obtenus sur les images de sinusoïdes en fonction de la taille du support **carré** utilisée.

2.1 Comparaison du support carré avec le support circulaire

Remarquons tout d'abord que le passage au support circulaire a une influence positive sur le biais des trois méthodes. Cette amélioration est d'autant plus remarquable pour les biais des plus petits supports, où la valeur maximale est de 43° pour la méthode de la BT dans le support carré de 4×4 pixels, alors qu'il atteint au maximum 12° dans le support circulaire circonscrit à ce bloc. Ce gain en précision implique cependant un nombre de calcul supplémentaire du fait du plus grand nombre de pixels inclus dans les supports circulaires. Le schéma 7.4 permet d'illustrer le calcul du rapport entre l'aire du carré : $A_{carre} = c^2$ et l'aire du cercle circonscrit : $A_{cercle} = \pi r^2$. Avec $r = \frac{c}{\sqrt{2}}$, nous pouvons observer que le rapport entre l'aire du support circulaire avec le support carré est égal à $\frac{\pi}{2}$. Ce résultat n'est cependant pas tout à fait exact dans le domaine discret puisque l'aire totale des pixels appartenant au support circulaire est légèrement inférieure à l'aire du cercle. De manière générale pour toutes les tailles de supports, le rapport entre l'aire des pixels inclus dans le support circulaire et l'aire des pixels du support carré est de $\frac{3}{2}$. L'augmentation du nombre de pixels du support circulaire se justifie cependant à la vue de la réduction des biais obtenue.

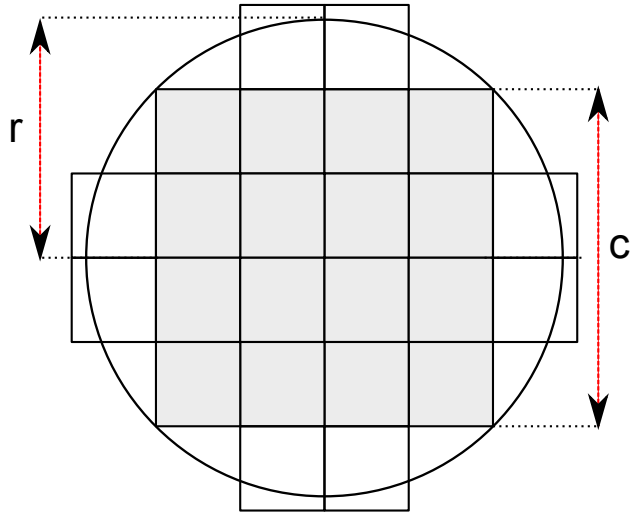


FIGURE 7.4 – Schéma illustrant le calcul du rapport de surface entre le support carré (gris) et le support circulaire (gris + blanc).

2.2 Interprétation des résultats pour le support circulaire

La Figure 7.2 contient les biais obtenus pour la détection dans les supports circulaires. Ce graphique montre que pour les deux plus grandes tailles de bloc, c'est la méthode de la BT qui présente les biais les plus faibles. Notre méthode semble arriver en seconde position, devant la méthode de Radon. Le passage au plus petit bloc montre en revanche que notre méthode parvient à obtenir un biais nul alors que les deux autres techniques ont des valeurs de biais très importantes. Nous expliquons ces résultats dans les parties suivantes, qui traitent indépendamment chaque approche.
Enfin, il est intéressant de noter que le passage au dessus de la borne de Nyquist (fréquence f_3) n'a pas d'influence négative sur les performances de détection des trois méthodes. Les performances de la méthode de Radon sont même meilleurs à cette fréquence.

2.3 Discussion sur la méthode de la BT

Cette méthode présente les meilleurs résultats pour les deux plus grandes tailles de bloc. Nous expliquons ceci par deux raisons principales. La première provient de la manière dont le calcul de variations est effectué. Le seuillage des coefficients d'ondelettes le long de chaque droite projetée est une méthode efficace pour estimer les variations d'une droite, et permet donc de mieux capter les directions homogènes. Ceci est vrai quelle que soit la fréquence des contours puisque la transformée en ondelettes est réalisée sur plusieurs échelles.
La deuxième raison provient du choix de la méthode de projection des pixels du bloc selon une direction (voir le chapitre 4.4). Ici, le bloc est projeté sur des droites discrètes fines. Nous avons vu dans le chapitre 1.2.2 que les droites discrètes fines étaient la représentation discrète la plus précise d'une direction, car deux pixels successifs de ces droites sont exactement alignés dans la direction à représenter. Dans un cas théorique comme les images de test utilisées dans cette partie, il est logique d'obtenir un très bon résultat lorsque cette méthode de projection est utilisée.

Cependant, les performances de cet algorithme se dégradent pour le plus petit support. Le fort biais obtenu pour ces supports s'explique, selon nous, par le fait qu'il n'est pas possible de projeter correctement les pixels du support sur des droites fines lorsque ce dernier est trop petit. Ceci est illustré par le schéma de la Figure 7.5. Dans cet exemple, les flèches représentent les enchaînements de pixels appartenant à la même droite. Ces flèches permettent de voir qu'il y a

assez peu de droites qui contiennent plus d'un pixel. Par conséquent, il est difficile d'évaluer les variations le long de la droite créée puisque celle-ci contient beaucoup d'enchaînements de pixels non représentatifs de la direction à étudier.

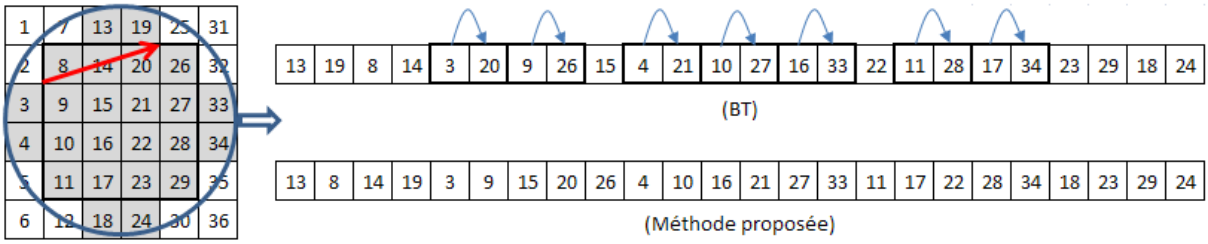


FIGURE 7.5 – Enchaînement de pixels d'un support circulaire d'un diamètre de 6 pixels dans la direction représentée par la flèche rouge et par les paramètres discrets $(3, 1)$, selon les algorithmes de la Bandlet Toolbox et la méthode proposée.

2.4 Discussion sur la méthode de Radon

Les performances de la détection de direction par la méthode de Radon sont légèrement moins bonnes que la méthode de la BT dans le plus grand support, mais équivalentes à notre algorithme. Pour le support 12×12, les performances sont moins bonnes que les deux autres approches. Enfin, pour le support 6×6, les biais engendrés sont moins importants que ceux de la méthode de la BT. Pour expliquer les biais obtenus, nous rappelons brièvement le fonctionnement de la transformée de Radon expliquée dans la partie 4.6.2.
La transformée de Radon consiste à projeter les pixels du support le long de lignes droites continues. Lorsque les droites ne tombent pas sur des pixels de la grille, ces derniers sont interpolés pour obtenir les valeurs intermédiaires. Cette technique de projection permet de tester toutes les orientations souhaitées, contrairement aux deux autres méthodes. Les niveaux de gris de l'image sont ensuite intégrés le long des droites créées. L'algorithme de transformée de Radon que nous utilisons est celui implémenté dans Matlab.
La difficulté de cette méthode consiste à extraire l'information de direction dominante à partir des valeurs intégrales obtenues pour chaque direction. La méthode utilisée ici est celle qui est explicitée dans [Jafari Khouzani 2005]. Elle consiste à étudier la courbe des valeurs de variance de la transformée de Radon pour chaque angle, et à en prendre la dérivée seconde. Le minimum de la dérivée seconde est ensuite détecté afin de trouver l'angle contenant le moins de variations. La conclusion que nous pouvons tirer de cette étude est donc assez similaire au cas de la méthode de la BT, à savoir que cette méthode semble perturbée par les petits supports, même si les biais engendrés sont inférieurs à la méthode de la BT pour les supports 6×6. L'interprétation de ce résultat est malgré tout moins évidente que pour la BT, puisque les résultats des projections sont intégrés et donc moins faciles à étudier. La méthode de projection en elle-même n'est *a priori* pas à mettre en cause, puisqu'elle permet de représenter exactement une direction, en interpolant les pixels de la grille afin d'obtenir la valeur des pixels alignés dans la direction étudiée. Cependant, nous pensons que les biais introduits peuvent venir de la manière dont les projections et l'interpolation sont gérées sur les bords des supports. Ceci expliquerait la différence de biais entre grands et petits supports, puisque l'impact des effets de bords dans les petits supports est effectivement beaucoup plus important.

2.5 Discussion sur la méthode proposée

Notre méthode permet d'obtenir des résultats légèrement moins bons que la méthode de la BT dans les deux plus grands supports, mais permet cependant d'obtenir un biais nul dans les

plus petits supports. Nous allons expliquer dans les paragraphes suivants successivement ces deux comportements.

La raison principale pour laquelle nous obtenons des biais plus forts que la méthode de la BT pour les grands supports vient de la méthode de projection utilisée. Nous utilisons une projection sur des droites naïves alors que les droites fines sont utilisées par la méthode de la BT. Les droites naïves, introduites dans le chapitre 1.2.2, sont moins précises au niveau de la représentation discrète d'une direction. En effet, plusieurs droites naïves distinctes de décalage différent peuvent représenter la même direction. Or, nous choisissons de projeter le bloc sur une droite naïve unique, de décalage nul. Il est donc possible que la représentation discrète choisie pour évaluer la direction ne soit pas celle qui représente le mieux le contour de l'image. L'incertitude créée par le choix d'une unique représentation discrète d'une direction parmi toutes celles possibles engendre selon nous un biais plus important que dans le cas d'une projection théoriquement parfaite.

Cependant, il est possible de voir que pour le plus petit support notre méthode obtient un biais nul. Cela provient principalement de deux raisons. Le premier vient, encore une fois, du choix de la méthode de projection. Nous avons vu dans la partie d'analyse des résultats de la BT que la projection sur des droites fines n'était pas approprié dans des petits supports puisque les segments contiennent au plus deux pixels à l'intérieur du support. Il n'est donc pas facile dans ce cas d'évaluer les variations des pixels du support. En revanche, la projection que nous utilisons créé des segments de droites connexes. La comparaison avec la projection de la méthode de la BT est illustrée en Figure 7.5. Grâce à cette méthode de projection, il est plus facile d'évaluer les variations puisque chaque segment créé contient suffisamment de pixels.
Par ailleurs, nous avons vu dans le chapitre 6.5.2.2 que le cas du plus petit support était traité différemment des autres. En effet, tous les décalages (μ) possibles sont pris en compte pour la construction des droites discrètes servant à la détection de direction. Ceci permet de tester toutes les représentations discrètes 8-connexes d'une direction. La redondance introduite, si elle augmente le nombre de droites testées et donc le nombre de calculs effectués, permet d'augmenter les chances d'évaluer correctement la direction du contour. Le surplus de calculs ajouté (projection du bloc sur 36 droites au lieu de 16) est relativement faible puisque les segments créés font au plus six pixels de long. Ce choix se justifie donc largement à la vue de l'amélioration des performances. Par curiosité, nous avons testé l'utilisation de tous les décalages de droites pour les deux plus grands supports. Les biais obtenus ne dépassent jamais 0.05 degré et sont du même ordre de grandeur que les résultats de la BT, quelle que soit la fréquence et la taille du support testés. Ceci permet de valider notre hypothèse qui attribue le biais plus important de notre méthode à la méthode de projection utilisée. Par ailleurs, notons que l'utilisation de tous les décalages pour les deux plus grands supports n'est pas réalisable puisque le nombre de calculs ajouté est trop important pour une application concrète.

2.6 Temps de calcul

Les temps de calcul indiqués dans le tableau de la Figure 7.6, correspondent aux temps moyens en secondes qu'il faut pour l'estimation d'une direction dans chaque taille de support. Les calculs ont été réalisés sous Matlab et sur un processeur deux coeurs à 2.4Ghz. Ce tableau permet de voir que les temps de calcul sont pratiquement similaires dans les deux plus petites tailles de supports, mais qu'elles commencent à diverger pour le grand support. La transformée de Radon montre alors les meilleures performances en termes de rapidité alors que la méthode de la BT est plus de deux fois plus lente. A la vue du nombre de calculs nécessaires pour effectuer l'estimation à partir de la BT, ce résultat est logique. En effet une fois la projection réalisée, une transformée en ondelettes doit être effectuée sur le segment projeté, suivie d'un seuillage des coefficients puis d'une transformée en ondelettes inverse. Ce surplus de calcul se fait donc ressentir au niveau du

Taille du support	BT (sec)	RADON (sec)	NOTRE METHODE (sec)
24×24	0.35	0.14	0.19 (0.73)
12×12	0.04	0.03	0.04 (0.07)
6×6	0.01	0.02	0.01 (0.01)

FIGURE 7.6 – Temps de calcul nécessaire à l'estimation d'une direction dans chaque taille de support.

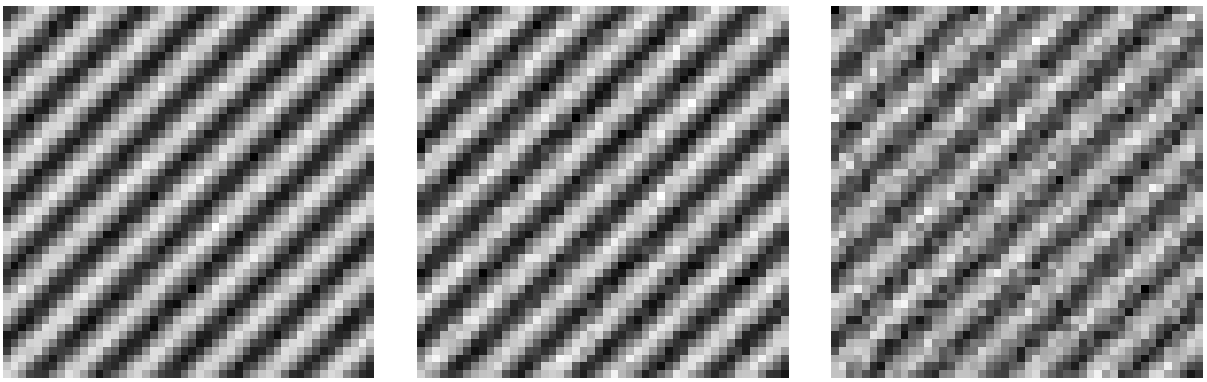
grand support puisque la longueur des segments à transformer en ondelettes devient significative. A l'inverse, la méthode choisie pour calculer les variations dans notre approche montre qu'en plus d'avoir des performances de détection fiables, elle reste efficace au niveau temps de calcul.

3 Influence du bruit

La section précédente nous a permis d'évaluer les performances d'estimation d'orientation des différentes méthodes sur des images de synthèse. L'avantage de ces images de test est de maîtriser l'orientation des sinusoïdes et donc d'évaluer le biais théorique de chaque méthode. Cette partie vise à estimer la robustesse de l'estimation à différents types de bruit, sur ces mêmes images de synthèse. Cette propriété est importante puisque il est probable que les images réelles sur lesquelles sera appliquée l'estimation contiendront du bruit. Nous avons donc simulé deux types de bruits qui peuvent éventuellement corrompre des images réelles. Le premier est un bruit blanc de différentes variances, additionné directement sur l'image. Le deuxième est engendré en compressant l'image originale en Jpeg, avec un taux de compression assez fort de façon à faire apparaître les frontières de blocs. Ces contours fictifs peuvent perturber la détection d'orientation, et sont assez fréquents dans les images naturelles étant donnée la popularité des formats Jpeg ou des encodages vidéos Mpeg2 et H264.

3.1 Bruit blanc

La Figure 7.7 montre les trois différents niveaux de bruit, σ_1, σ_2 et σ_3 que nous avons testés. De la même manière que précédemment, trois fréquences de sinusoïdes sont évaluées, un support circulaire est utilisé, et le nombre de directions testées est adapté en fonction de la taille du support. Le bruit blanc étant ajouté de façon aléatoire, l'estimation a été réalisée sur vingt réalisations par méthode et par taille de support, et les résultats présentés dans la Figure 7.8 correspondent à la moyenne des biais de toutes les réalisations.



(a) (b) (c)

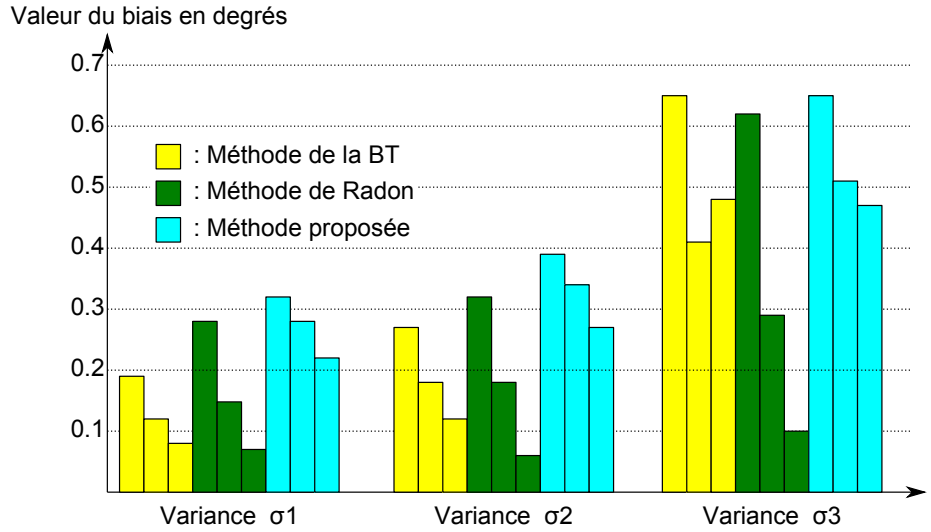
FIGURE 7.7 – Images de sinusoïdes à la fréquence f_1 auxquelles est ajouté un bruit blanc de variance (a) $\sigma_1 = 1.18$, (b) $\sigma_2 = 1.77$, (c) $\sigma_3 = 3.54$.

3.1.1 Interprétation des résultats

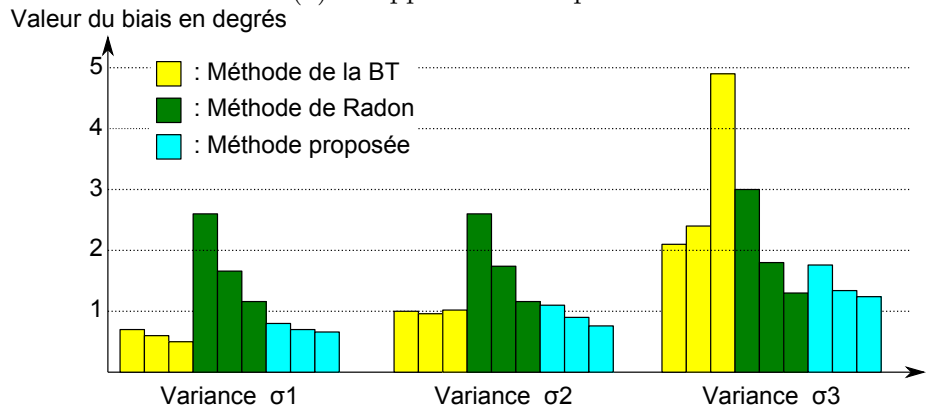
Les biais calculés dans les grands et moyens supports sont présentés en Figure 7.8(a) et (b). Dans le cas non bruité, c'est la méthode de la BT qui donnait les meilleurs résultats pour ces deux tailles de support. Dans le cas bruité, cette méthode parvient à conserver sa supériorité uniquement dans les cas de variance de bruit faible (σ_1). Pour la variance de bruit moyenne (σ_2), la méthode de Radon et la nôtre proposent des résultats sensiblement équivalents respectivement pour les grands supports et moyens supports. Mais pour la plus forte variance de bruit (σ_3) et dans le cas du grand support, la méthode de Radon est celle qui permet d'obtenir les biais les plus faibles, et notre méthode et celle de la BT proposent des biais équivalents. Pour les supports de taille moyenne, notre méthode présente les meilleurs résultats, et la méthode de Radon parvient à doubler celle de la BT.

Pour les petits supports, notre méthode conserve l'avantage qu'elle avait dans le cas non bruité en présentant des biais qui restent bien inférieurs à ceux des deux autres méthodes. Notons par ailleurs que la méthode de Radon semble moins influencée par l'augmentation de la variance du bruit que celle de la BT. En effet, les biais de la méthode de Radon sont comparables d'une variance à l'autre alors que les biais de la méthode de la BT augmentent.

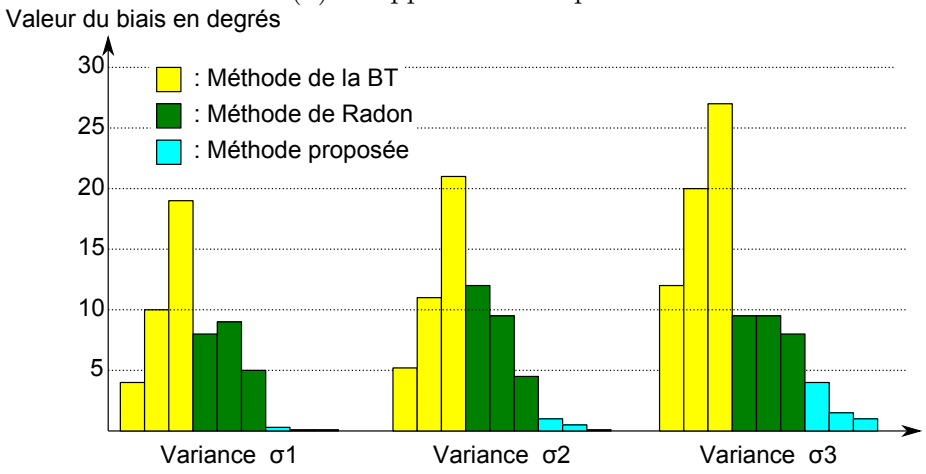
En résumé, la méthode de la BT semble être la méthode la plus influencée par l'ajout d'un bruit blanc puisqu'à partir d'une certaine valeur de variance de bruit, elle perd l'avantage qu'elle avait sur les deux autres méthodes dans le cas non bruité. La méthode de Radon montre un comportement plus robuste au bruit mais ce bon comportement ne suffit pas à combler les lacunes de cette méthode lorsque les tailles de supports diminuent. Enfin, le comportement de notre méthode semble se situer entre les deux autres puisqu'elle parvient d'un côté à conserver l'avantage qu'elle avait dans les petits supports et de l'autre à combler le retard voire améliorer les résultats de la BT pour les tailles de support supérieures. Ce comportement justifie l'utilisation des droites naïves dans notre méthode par rapport aux droites fines pour la projection des pixels, puisque ces dernières permettent de gagner en robustesse dans le cas d'images altérées.



(a) - Support 24×24 pixels



(b) - Support 12×12 pixels



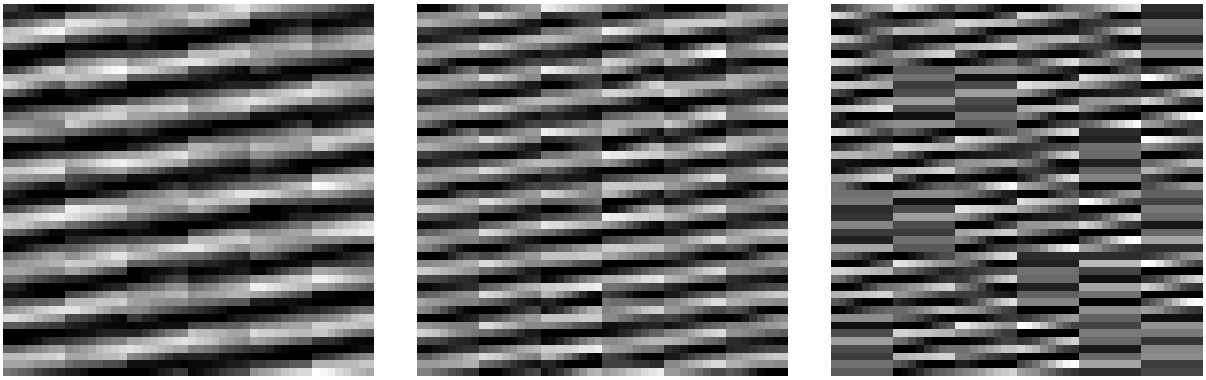
(c) - Support 6×6 pixels

FIGURE 7.8 – Valeurs des biais pour les images bruitées aux fréquences f_1, f_2 et f_3 pour les trois valeurs de variance de bruit testées et les trois tailles de supports.

3.2 Bruit de compression

La Figure 7.9 illustre les images de sinusoïdes compressées en Jpeg pour les trois mêmes fréquences que précédemment. Les effets de blocs sont nets, et il est même possible de voir l'effet de la quantification à l'intérieur de chaque bloc, en particulier pour la fréquence la plus élevée f_3. Les résultats sont illustrés en Figure 7.10. Il est tout d'abord intéressant de noter que les trois méthodes sont particulièrement affectées par les dégradations de l'image à la fréquence f_3. En plus des effets de bloc, des directions parasites apparaissent du fait de la fréquence trop élevée des sinusoïdes par rapport au taux d'échantillonnage de la grille de pixels. La détection de direction est alors perturbée et les biais liés à toutes les méthodes sont fortement augmentés. Ce cas représente un cas extrême au delà duquel il est difficile d'envisager une méthode de détection efficace, puisque déjà visuellement les directions n'apparaissent plus clairement.

De manière générale, la conclusion est assez similaire que dans le cas du bruit blanc. La méthode de la BT est plus fortement perturbée par la compression que les deux autres méthodes. La méthode de Radon est plutôt robuste pour le grand support, mais éprouve des difficultés quand la taille du support diminue. Notre méthode parvient à conserver un comportement correct dans les grands supports, et garde sa supériorité dans les moyennes et petites tailles de supports.
Notre technique de projection semble donc plus robuste à ce type de perturbation que la projection sur droites fines. En effet, les frontières de blocs engendrées par la compression ne perturbent qu'une faible proportion des éléments projetés sur la droite naïve. A l'inverse dans le cas des droites fines non-connexes, les frontières de bloc ont un impact plus fort puisque deux éléments successifs de la droite projetée ont de fortes chances de se retrouver dans des blocs différents et donc de perturber l'évaluation de la régularité le long de la projection.



(a) (b) (c)

FIGURE 7.9 – Images de sinusoïdes compressées et à différentes fréquences : (a) f_1, (b) f_2, (c) f_3.

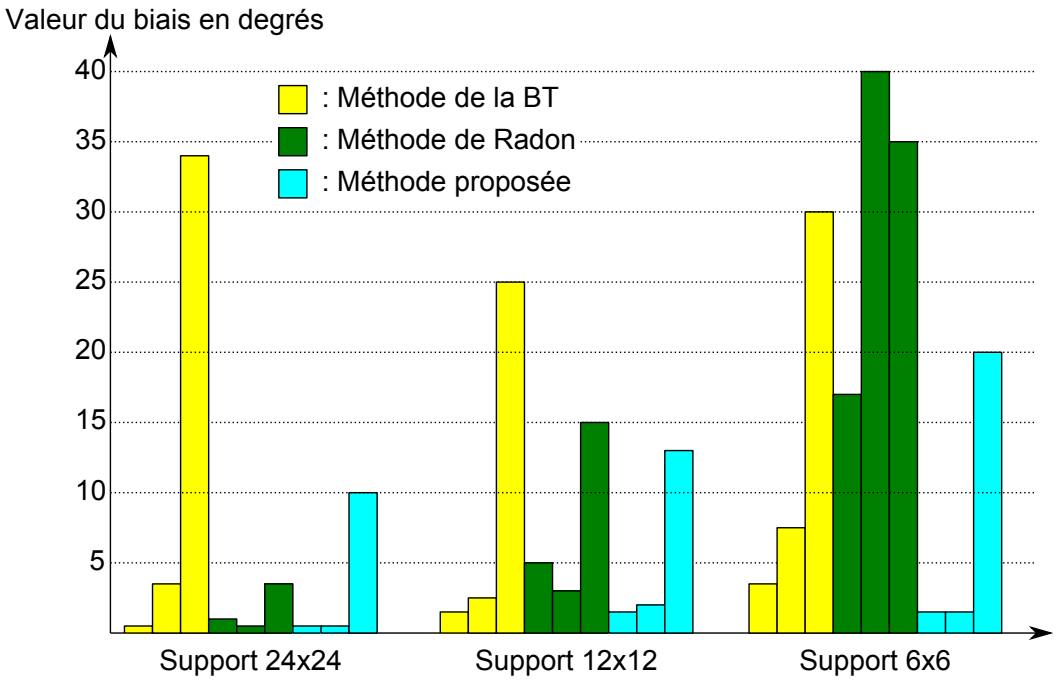


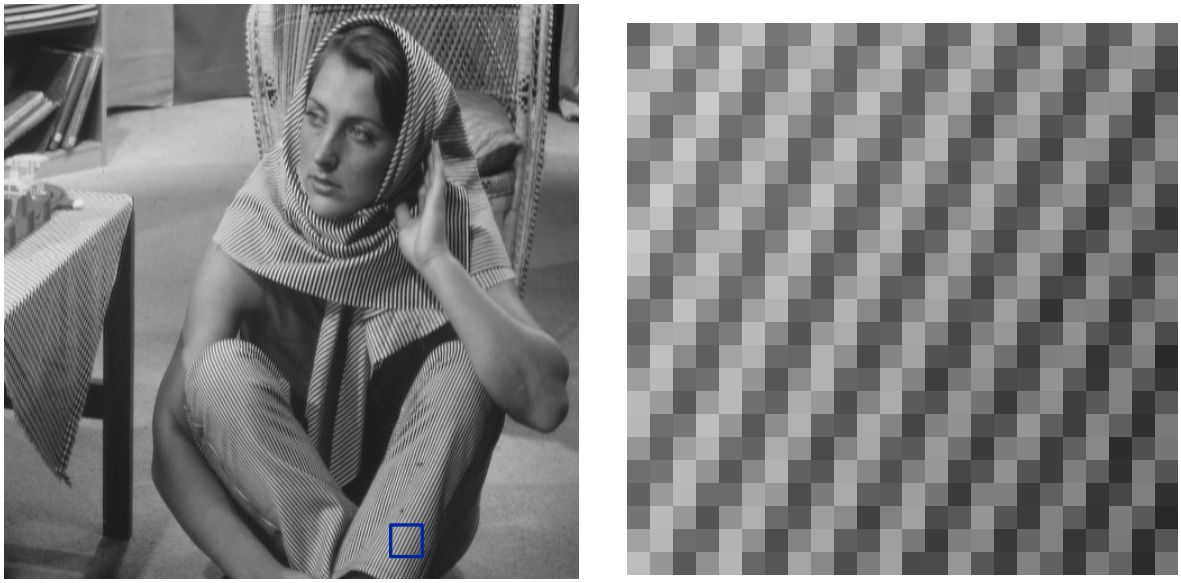
FIGURE 7.10 – Valeurs des biais obtenus sur les images compressées en Jpeg, pour les trois fréquences f_1, f_2 et f_3.

4 Cas d'images naturelles

La section précédente nous a permis d'évaluer les biais de chaque méthode sur des exemples théoriques dont l'orientation était maîtrisée, ce qui permettait d'évaluer le biais théorique de chaque méthode. Malgré l'ajout de différents types de bruits à ces images, certaines caractéristiques des images réelles restent malgré tout bien différentes de ces images de test. Nous verrons donc qu'en jouant sur certains paramètres du contour (contraste, épaisseur ou longueur), des failles apparaissent dans les méthodes de détection.
Les trois méthodes seront comparées de manières qualitative puisque les exemples que nous allons prendre sont issus d'images réelles. La démarche derrière cette partie est d'illustrer les inconvénients récurrents que nous avons pu relever pour chaque méthode et de justifier notre démarche. Pour chacune d'entre elles, les mesures de régularité associées seront montrées pour illustrer le raisonnement. Ainsi sont tracés le Lagrangien pour la méthode de la BT, la dérivée seconde de la variance de chaque projection pour la transformée de Radon et la moyenne des variations des amplitudes pour la nôtre.

4.1 Cas d'une texture directionnelle

Afin de bien comprendre la démarche de comparaison, le premier exemple est une partie de l'image *Barbara* de taille 24×24 qui contient des rayures orientées, où les trois méthodes estiment correctement l'orientation des contours. L'image originale et la partie choisie pour réaliser l'estimation sont montrées en Figure 7.11. La Figure 7.12 montre les résultats du calcul de régularité pour respectivement, la méthode de la BT, la transformée de Radon et la méthode proposée. Le minimum de chaque courbe indique l'angle (ou la droite selon la méthode) la plus régulière selon le critère choisi. L'angle réel des rayures calculé directement sur l'image est d'environ 63.4°. L'exemple étant tiré d'un cas réel, l'angle n'est pas aussi constant que dans le cas des images théoriques. Les détections peuvent donc légèrement diverger de la valeur calculée sans que celles-ci ne soient considérées comme inexactes pour autant.
Ainsi, les résultats déduits des courbes de la Figure 7.12 montrent que les trois détections sont



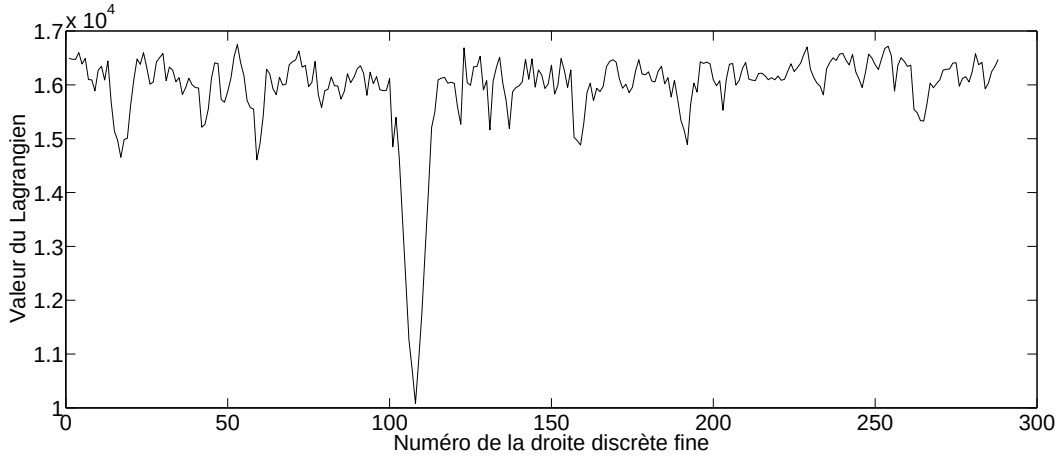
(a) (b)

FIGURE 7.11 – (a)Image *Barbara*. Le carré indique le bloc de taille 24×24 illustré en (b) et utilisé pour l'estimation.

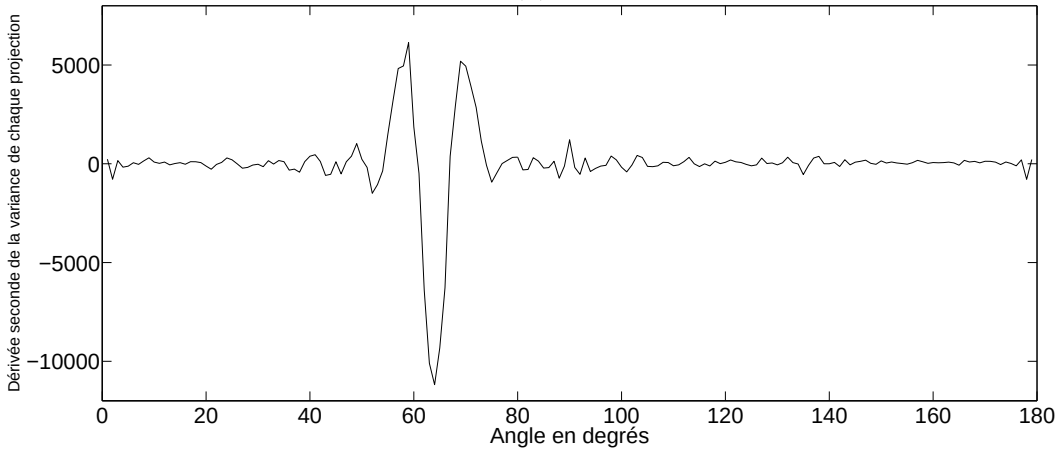
correctes :

- 63.43° pour la méthode de la BT
- 65° pour la méthode de Radon
- 63.43° pour notre méthode

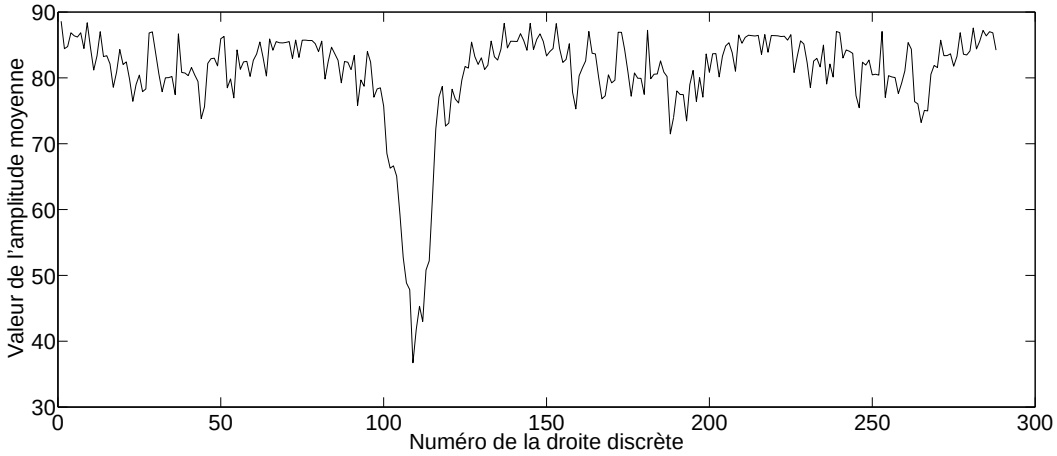
De plus, remarquons que les minimums des courbes sont clairs et que les détections ne sont pas ambigues. Nous verrons que dans les exemples suivants, certaines interprétations de courbes sont moins évidentes.



(a)



(b)



(c)

FIGURE 7.12 – Résultat des mesures de régularité pour le bloc de l'image *Barbara* : (a) la méthode de la BT, (b) la méthode issue de la transformée de Radon, (c) notre méthode.

4.2 Contours tronqués

L'exemple de cette partie contient un contour rectiligne qui ne traverse pas tout le bloc analysé. L'extrait de l'image, de taille 24×24 est montré en Figure 7.13 et contient un contour rectiligne partiel dont l'angle est mesuré à 75.07°.

(a) (b)

FIGURE 7.13 – (a)Image *Plane*. Le carré indique le bloc de taille 24×24 illustré en (b) et utilisé pour l'estimation.

Les minimums obtenus pour chacune des courbes de la Figure 7.14 correspondent aux angles suivants :

- 0° pour la méthode de la BT
- 76° pour la méthode de Radon
- 73.03° pour notre méthode

Ces résultats montrent que la méthode de Radon et la nôtre donnent un résultat correct alors que la méthode de la BT a son minimum pour un angle de 180°. Notons tout de même que le minimum de la courbe des variations de la méthode de Radon est détecté avec très peu de marge par rapport à d'autres pics parasites. En revanche, le minimum des variations de notre méthode apparaît clairement. Les paragraphes suivants expliquent pourquoi notre méthode parvient plus facilement à détecter la bonne direction que les deux autres.

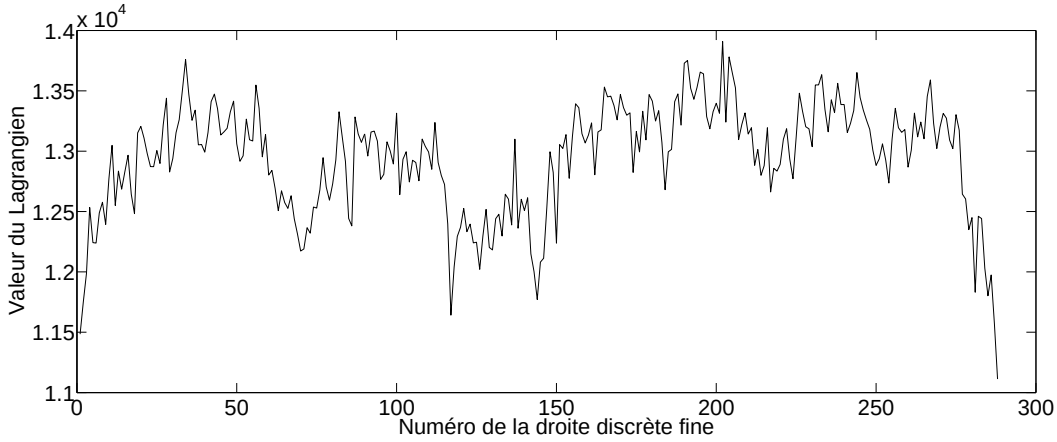
4.2.1 Présence de pics parasites

De manière générale, les courbes de la méthode de Radon et de la BT sont perturbées par des pics parasites situées aux angles 0°, 45°, 90° et 135°. Ces perturbations, bien que déjà réduites par l'introduction du support circulaire sont présentes de manière assez systématique lors de détections moins évidentes que celles du premier cas présenté. Selon nous, ces pics sont dus aux propriétés particulières des projections réalisées pour ces angles.
En effet pour la méthode de la BT, la projection le long de droites fines selon ces quatre directions revient à chaîner les pixels de façon connexe, qui sont les mêmes que nous construisons avec notre méthode de projection. Les segments créés lors de la projection contiennent donc un nombre de pixels maximal par segment, ce qui permet de réduire le nombre total de segments. Par conséquent, les projections effectuées pour ces quatre directions ont des propriétés très différentes de celles d'autres directions. En particulier, le faible nombre total de segments créés réduit le nombre de sauts de ligne contenus dans la droite projetée. Les variations calculées sur des droites contenant peu de sauts de ligne ont donc des valeurs moins fortes, ce qui explique la présence de pics minimums parasites pour ces quatre directions particulières.
Pour la méthode de Radon, la projection selon ces directions ne nécessitent pas d'interpolation

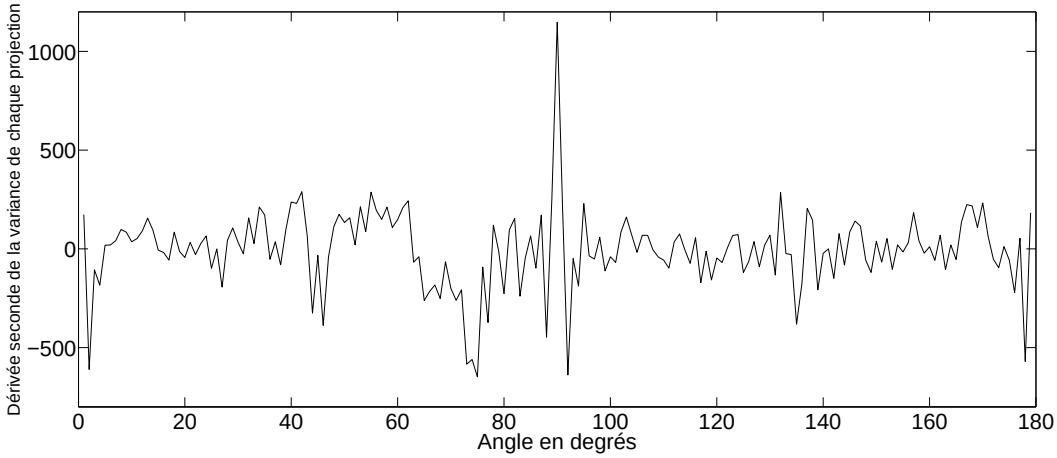
puisque les lignes passent exactement par les pixels de la grille de l'image. Les caractéristiques par rapport aux droites voisines interpolées, bien que légèrement différentes, ressortent de façon évidente lors du calcul de la dérivée seconde en faisant apparaître des extremums locaux pour ces angles. Ces pics parasites sont en général visibles dans les courbes de variation mais ne gênent pas systématiquement la détection de direction comme c'est le cas ici.
Notre méthode ne semble pas perturbée par cet exemple. Cela vient en grande partie de l'homogénéité des droites projetées pour toutes les directions qui leur permet à toutes d'avoir les mêmes caractéristiques.

4.2.2 Influence du contour tronqué

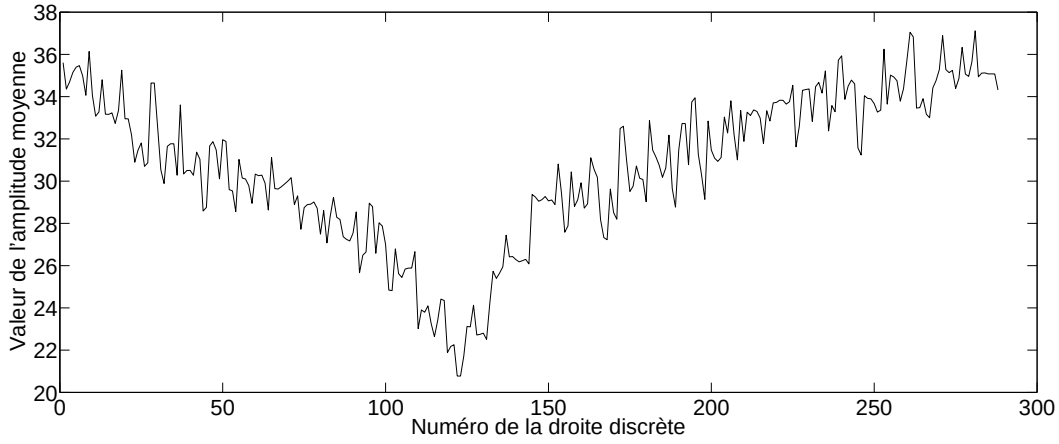
Le fait que le contour de l'exemple ne traverse pas tout le bloc joue également un rôle important dans la difficulté qu'on les méthodes de Radon et de la BT à obtenir un minimum franc.
Pour la méthode de Radon, lorsque l'intégrale des niveaux de gris est calculée pour la ligne située sur le contour, les niveaux de gris à la fois du contour et du fond sont intégrés. La variance entre l'intégrale cette ligne et celle d'une ligne parallèle n'est donc pas aussi élevée que si le contour traversait le bloc, ce qui ne fait pas ressortir un minimum franc lors de l'étude de la courbe.
Pour la méthode de la BT, l'erreur de détection vient plutôt de la méthode de projection sur les droites fines que de la manière de calculer les variations. La droite située sur le contour sera composée de pixels non connexes où certains appartiennent au contour, et d'autres appartiennent au fond. Au lieu de capter la régularité le long du contour, la droite est considérée comme non homogène.
Une fois de plus, l'utilisation de droites connexes pour projeter les pixels permet de mieux capter la régularité d'un contour même lorsque celui-ci ne traverse pas tout le bloc. De plus, la méthode de calcul des variations est plus adaptée que celui de la méthode de Radon ou l'intégration des niveaux de gris n'est pas une mesure représentative de la régularité d'une ligne, dans le cas d'un contour partiel.



(a)



(b)



(c)

FIGURE 7.14 – Résultat des mesures de régularité pour le bloc de l'image *Plane* : (a) la méthode de la BT, (b) la méthode issue de la transformée de Radon, (c) notre méthode.

4.3 Contours épais

Le troisième exemple réel que nous allons étudier ici est un contour épais et peu contrasté. L'exemple est illustré en Figure 7.15. La difficulté à détecter la direction de ce genre de contour

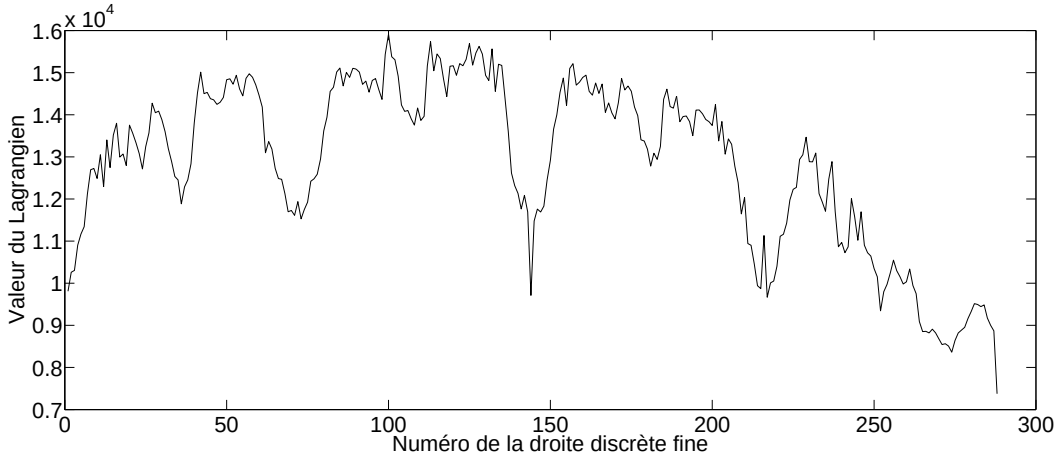
(a) (b)

FIGURE 7.15 – (a)Image *Lena*. Le carré indique le bloc de taille 24×24 illustré en (b) et utilisé pour l'estimation.

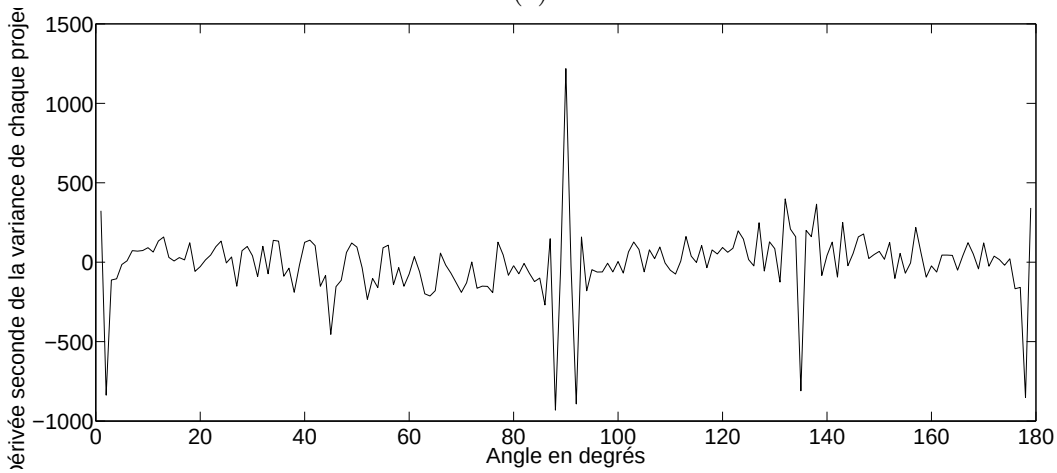
réside dans le fait que peu de pixels contiennent l'information de direction, et que ces pixels n'ont pas de gradients forts dans la direction perpendiculaire. En observant les courbes de la Figure 7.16, nous pouvons voir que ces deux critères impliquent une forte présence des pics parasites aux orientations 0°, 45°, 90° et 135° introduits plus haut. Cette fois-ci, le contour n'est pas suffisamment marqué pour que les méthodes de Radon et de la BT aient leur minimum au niveau de l'angle du contour (calculé autour de 164°). Les résultats obtenus à partir des courbes sont les suivants, et montrent l'influence des pics parasites :

- 0° pour la méthode de la BT
- 89° pour la méthode de Radon
- 168° pour notre méthode

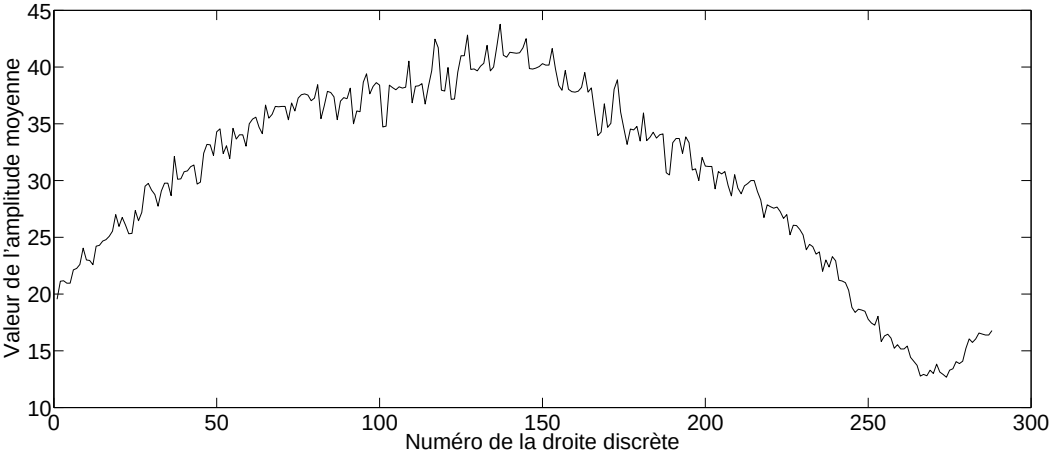
La courbe correspondant à notre méthode indique que la mesure n'est pas perturbée malgré le contexte délicat de la détection. De plus, rappelons que notre étude directionnelle est précédée d'une adaptation de résolution qui place le contour à l'échelle ou il est le mieux représenté. Lorsque cette technique est appliquée avant notre détection, la courbe de la Figure 7.17 montre que le minimum apparaît plus clairement et que l'orientation estimée est plus proche de la réalité terrain (estimation de l'angle : 163°). Dans ce cas précis, l'échelle choisie est la plus grossière ($J = 3$).



(a)



(b)



(c)

FIGURE 7.16 – Résultat des mesures de régularité pour le bloc de l'image *Lena* : (a) la méthode de la BT, (b) la méthode issue de la transformée de Radon, (c) notre méthode.

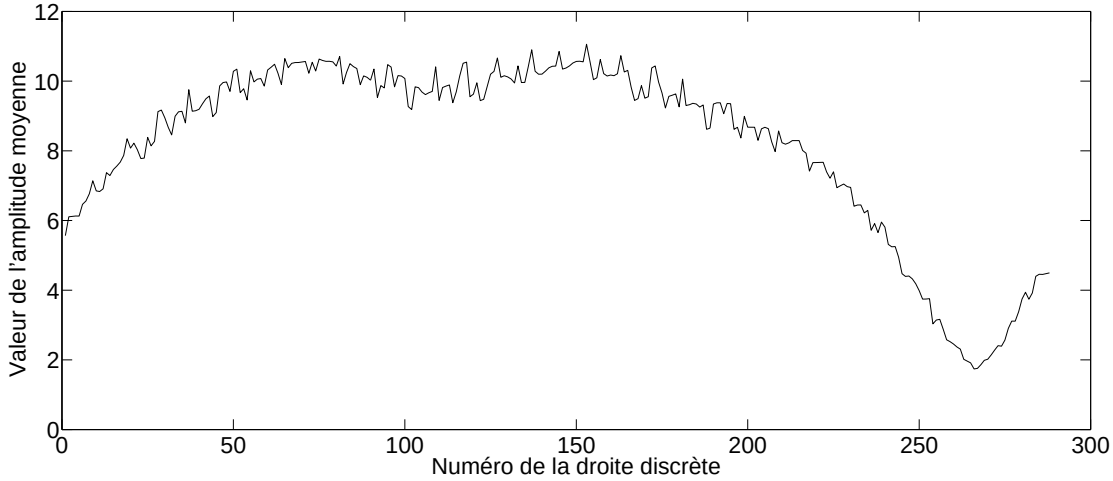


FIGURE 7.17 – Résultat de la mesure des variations du bloc de l'image *Lena* avec notre méthode précédée d'une adaptation de résolution.

5 Conclusion sur l'évaluation des estimations de direction

Dans un premier temps, nous avons observé le comportement des trois méthodes sur des images de synthèse contenant des contours théoriquement parfaits. Nous avons pu remarquer le bon comportement de la méthode de la BT dans les deux plus grands supports. Les faibles biais qu'elle permet d'obtenir sont dus à la méthode de projection qu'elle utilise qui est la représentation discrète théoriquement optimale d'une direction, et donc particulièrement bien adaptée à ce test. En revanche, l'étude du plus petit support montre que la méthode de projection utilisée par notre méthode ainsi que le traitement particulier effectué pour les supports de cette taille permettent d'obtenir un biais nul, alors que les deux autres méthodes présentent des biais importants.
De plus, nous avons illustré l'apport du support circulaire par rapport au support carré, qui permet de réduire fortement le biais pour les trois méthodes au prix d'un surplus de calcul limité. Par ailleurs, l'étude des temps de calcul montre que la méthode de Radon est la plus efficace mais que notre méthode se situe en deuxième position, en particulier grâce à l'utilisation d'une méthode simple du calcul de variations.

Dans un deuxième temps, l'ajout de bruit bruit blanc ainsi que d'un bruit de compression nous ont permis de mettre en avant le bon comportement de notre méthode, ainsi que celle de Radon (pour les grands supports) face à ces perturbations. En effet à partir d'une certaine valeur de variance de bruit, la méthode de Radon donne les meilleurs résultats pour les grands supports, et notre méthode obtient les biais les plus faibles pour les deux autres tailles de supports. D'autre part, ces exemples ont montré les limites de la méthode de projections sur des droites fines dès que l'on sort d'un cas théoriquement idéal.

Enfin, l'étude d'exemples naturels nous a permis de mettre en avant certaines difficultés rencontrées par les méthodes de Radon et de la BT. Ces exemples ayant été choisies de façon purement subjective, il ne s'agit pas d'en tirer des conclusions systématiques mais juste d'illustrer des comportements auxquels nous avons été souvent confrontés lors de l'étude de ces méthodes.
De la même manière que lors de l'étude des exemples bruités, il ressort de cette partie une tendance à un comportement plus robuste de la part de notre méthode par rapport aux deux autres, pour des contours pouvant avoir des caractéristiques très variées. Les estimations de direction de contour avec notre méthode sont en général plus fiables car le minimum de la courbe de variation est net et n'est pas perturbé par d'autres minima locaux. Les pics parasites présents et récurrents dans les deux autres courbes ne viennent en effet pas perturber ici le processus de détection. Ceci est dû à la fois à la connexité des droites projetées le long des directions testées et à la méthode utilisée pour calculer les variations qui rend bien compte des variations des niveaux de gris de l'image même dans des cas de figures particuliers.

De manière générale, les différences de comportement entre les trois méthodes sont accentuées lorsque la taille du bloc d'étude est diminuée, et ce en faveur de notre méthode. Cela a été remarqué lors de l'étude théorique, dans les cas bruités ainsi que lors d'expérimentations sur des exemples naturels. La méthode proposée apparaît donc comme la plus adaptée à l'étude de cas réels de par sa robustesse à des contours de natures très différentes, et à son comportement correct sur des exemples théoriques. En particulier, ses bonnes performances dans des voisinages de petites tailles permettent d'estimer efficacement les directions dans le cas de changements fréquents de direction ou de contours courbes, où l'algorithme de *quadtree* divise le bloc initial jusqu'au plus petit bloc.

Troisième partie

Interpolation directionnelle

Cette partie a pour but d'appliquer l'estimation de direction de contours présentée dans la partie précédente à l'interpolation d'image. Nous attendons à ce que l'utilisation des directions de contours permette d'éliminer les artefacts classiques qui apparaissent sur des images interpolées avec les méthodes d'interpolations classiques isotropes, à savoir le jaggy, les effets d'escalier et l'aliasing.

Dans un premier temps, nous étudierons un état de l'art des méthodes d'interpolation directionnelles récentes afin d'observer à la fois comment les directions de contours sont estimées et comment elles sont utilisées pour guider l'interpolation.

Dans un deuxième temps, nous introduirons notre méthode d'interpolation. Nous verrons comment l'interpolation est réalisée dans chaque bloc du *quadtree*, en tenant compte de la direction associée et dans le soucis de ne pas introduire de frontières de blocs visibles. Nous verrons que les artefacts usuels sont éliminés en imposant une régularité le long de la direction détectée grâce à un correcteur Gaussien orienté. Nous verrons qu'il est également nécessaire d'introduire une protection sur les zones du bloc ne contenant pas de contours, afin que l'application du correcteur Gaussien orienté ne les dégrade pas.

Dans un dernier chapître, nous comparerons les méthodes introduites dans l'état de l'art avec la méthode que nous proposons. Nous effectuerons une comparaison visuelle des résultats pour observer la capacité de chacune des méthodes à reproduire l'aspect des contours de l'image originale dans une grille de pixels plus dense, à éliminer les artefacts usuels et à ne pas en introduire de nouveaux. Par la suite, une comparaison objective sera effectuée grâce au PSNR et au SSIM. Nous verrons si ces métriques parviennent à retranscrire l'impression visuelle que nous avions conclu auparavant. Enfin, des perspectives seront données sur le développement futur de l'algorithme, ainsi que les étapes qui le sépare d'une intégration dans un circuit industriel.

Chapitre 8

Etat de l'art sur les méthodes d'interpolation directionnelles

Pour commencer cette partie, nous introduisons un certain nombre de méthodes d'interpolation directionnelles présentées dans la littérature récente. Ces méthodes sont plus ou moins complexes selon leurs applications, mais visent toutes à améliorer le rendu visuel des résultats sur la grille haute-résolution, en tenant compte de l'orientation des directions de contours lors de l'interpolation. Certaines de ces approches sont accessibles par l'intermédiaire du site Internet de l'auteur où intégrées dans des logiciels libres. Nous avons implémenté d'autres méthodes qui étaient suffisamment bien décrites dans l'article faisant référence, et certains autres auteurs nous ont fourni les résultats dans un but de comparaison.
Toutes les méthodes présentées dans cet état de l'art seront évaluées dans le dernier chapitre de cette partie, et comparées avec notre approche.

1 Interpolation AQua

L'interpolation AQua (Adaptive Quadratic) est mise au point par Muresan en 2005 [Muresan 2005]. Elle a depuis été intégrée dans les logiciels *Pictura* et *Visere Pro*. Ce dernier étant libre de droits, nous avons pu tester les performances de l'interpolation sur toutes les images souhaitées. L'analyse de cette méthode se fera en deux parties, la première expliquera la détection des directions de contours, et la deuxième s'intéressera à l'utilisation de ces directions pour l'interpolation.

1.1 Estimation des directions

La détection d'orientation réalisée dans cette méthode est assez simple puisqu'elle consiste à choisir pour chaque pixel une des quatre directions de base, à savoir 0°, 45°, 90° et 135°. Le choix de la direction se fait donc dans un voisinage 3×3 pixels de l'image originale, représenté en Figure 8.1. Un filtre passe-haut est appliqué dans chacune des quatre directions, et les sorties du filtre sont comparées pour décider de la direction dominante du voisinage étudié. Les différences de pixels correspondants aux sorties du filtre sont calculées de la manière suivante :

$$d_1 = |\frac{1}{2}(P_4 + P_6) - P_5|$$
$$d_2 = |\frac{1}{2}(P_3 + P_7) - P_5|$$
$$d_3 = |\frac{1}{2}(P_2 + P_8) - P_5|$$
$$d_4 = |\frac{1}{2}(P_1 + P_9) - P_5|$$

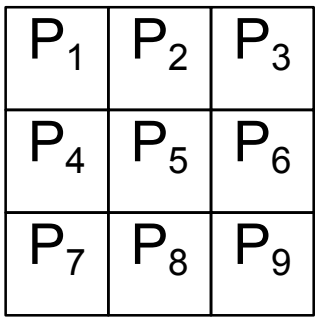


FIGURE 8.1 – Voisinage 3×3 pixels utilisé pour le choix de la direction attribuée au pixel central.

Les différences d_2 et d_4 représentent les variations des deux directions diagonales, et d_1 et d_3 représentent les variations des directions non-diagonales. L'estimation de direction est réalisée de manière indépendante pour les directions diagonales et non-diagonales. Le seuil S est la valeur à partir de laquelle une direction domine l'autre. Ainsi pour les directions non-diagonales, on a :

- Si $|d_1 - d_3| < S$ alors aucune des deux directions non-diagonales ne domine l'autre. Le pixel est dit neutre pour les directions non-diagonales.
- Sinon si $d_1 < d_3$ alors la direction associée au pixel est l'horizontale.
- Sinon on attribue au pixel la direction verticale.

De la même manière, on estime pour chaque pixel les directions diagonales :

- Si $|d_2 - d_4| < S$ le pixel est neutre pour les directions diagonales.
- Sinon si $d_2 < d_4$ alors la direction associée au pixel est la première diagonale (45°).
- Sinon on attribue au pixel la deuxième diagonale (135°).

Ces deux estimations permettent d'obtenir une image contenant l'information sur les directions non-diagonales et une autre contenant l'information sur les directions diagonales. Chacune de ces deux images est filtrée avec un filtre médian afin de retirer les éventuelles directions isolées.

1.2 Application à l'interpolation

L'utilisation de ces directions pour l'interpolation est assez directe. La première interpolation est réalisée grâce à l'image des directions diagonales. Elle se déroule dans un voisinage défini de manière à ce qu'il contienne au moins quatre pixels originaux. Ce voisinage est représenté par le carré épais de la Figure 8.2. On attribue ensuite à ce voisinage sa direction dominante définie comme étant la direction majoritaire des pixels originaux contenus dans le voisinage. Le voisinage est donc labellisé soit diagonal-1, diagonal-2 ou diagonal-neutre. Ensuite, les pixels situés dans la direction attribuée au voisinage (en incluant le pixel central) sont interpolés de manière linéaire. Puis, les pixels le long de l'autre diagonale sont interpolés (également de manière linéaire), en utilisant le pixel central obtenu précédemment ce qui permet de diviser le taux d'échantillonnage par deux. A la fin de ce processus, tous les pixels diagonaux sont interpolés, formant ainsi des losanges reliant les pixels originaux.

Pour interpoler les pixels restants, l'information des directions non-diagonales est utilisée. De la même manière que précédemment, des voisinages sont créés. Ils sont représentés par les carrés épais de la Figure 8.3, et labellisés en fonction de la direction majoritaire des pixels originaux du voisinage. Les pixels situés à l'intérieur des losanges formés par les pixels déjà connus sont donc interpolés par interpolation linéaire, dans la direction attribuée au voisinage. A la fin de ce procédé, une fois que les voisinages ont couvert toute l'aire de l'image, tous les pixels de la grille haute résolution ont une valeur connue.

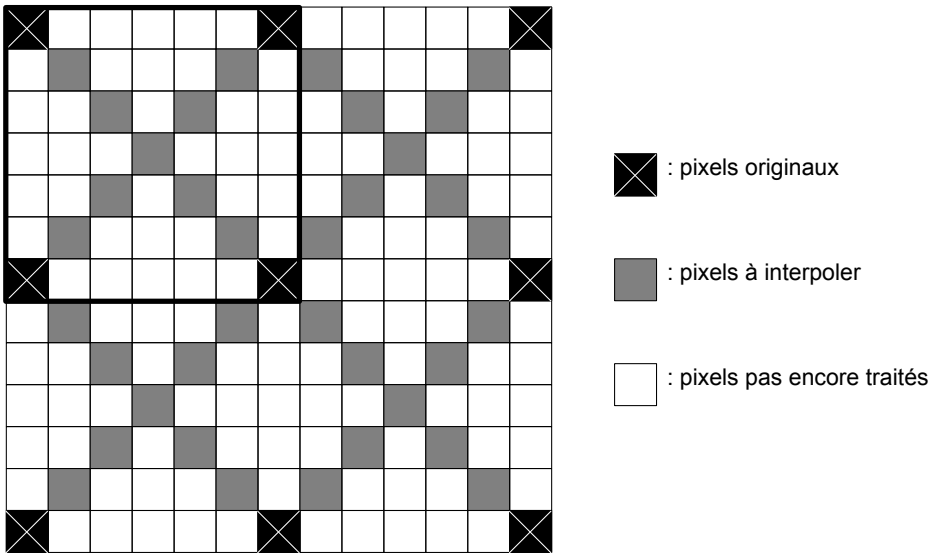


FIGURE 8.2 – Schéma d'interpolation pour les directions diagonales. Ce voisinage illustre les pixels à interpoler, les pixels pas encore traités et les pixels originaux.

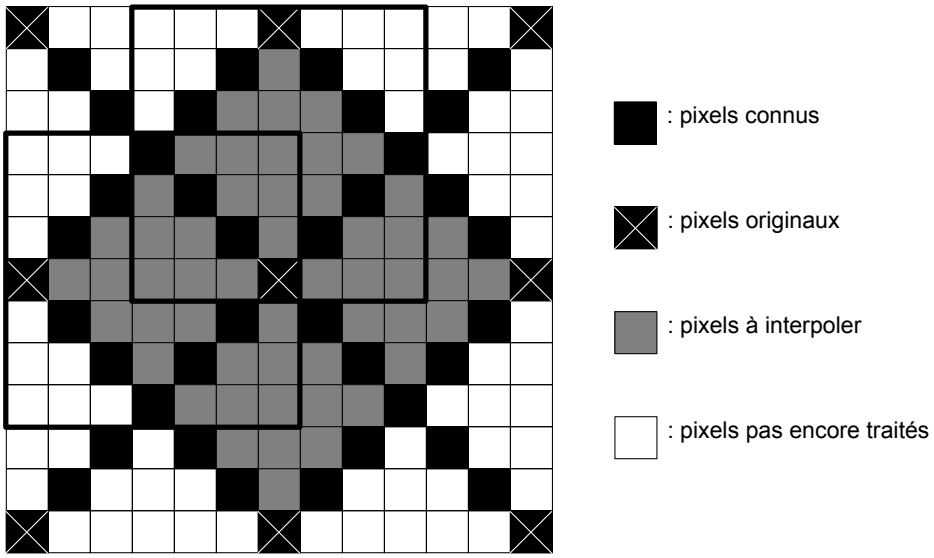


FIGURE 8.3 – Schéma d'interpolation pour les directions non-diagonales. Ce voisinage illustre les pixels à interpoler, les pixels pas encore traités, les pixels originaux et les pixels déjà connus.

2 Interpolation IAD

Cette technique d'interpolation que nous appelons IAD (Interpolation Adaptée aux Directions) a été mise au point par Moloney et Jiang en 2002 et peut être trouvée dans l'article suivant : [Jiang 2002]. Elle est basée sur l'estimation de Perona que nous avons présenté dans la partie 4.2. Cette dernière consiste à calculer les directions des contours en diffusant de manière itérative les directions du gradient, grâce aux équations de la diffusion de la chaleur. Par rapport à la méthode AQua, les directions obtenues sont beaucoup plus précises puisqu'elles ne sont pas limitées au nombre de quatre. La méthode est cependant beaucoup plus complexe et demande donc un nombre de calcul plus important. Etant donné que la méthode d'estimation d'orientation est déjà expliquée dans l'état de l'art du chapitre 4.6.2, nous allons directement passer à l'explication de la méthode d'interpolation.

Le problème d'interpolation est formulé ici comme un problème variationnel, qui consiste à minimiser l'intégrale suivante :

$$\int_x \int_y |\nabla I(x,y) dx dy| \tag{8.1}$$

avec les contraintes suivantes sur le gradient :

$$\begin{aligned} \frac{I_x}{\sqrt{I_x^2 + I_y^2}} &= \cos(\hat{\theta}(x,y)) \\ \frac{I_y}{\sqrt{I_x^2 + I_y^2}} &= \sin(\hat{\theta}(x,y)) \end{aligned} \tag{8.2}$$

Dans cette équation, $\hat{\theta}(x,y)$ est l'image contenant les orientations estimées dans un premier temps. L'équation 8.1 permet donc d'obtenir une image interpolée qui a de plus faibles variations le long des directions détectées. L'équation d'Euler de l'équation 8.1 est :

$$\nabla . \left(\frac{\nabla I(x,y)}{\|\nabla I(x,y)\|} \right) = 0 \tag{8.3}$$

En incluant les contraintes de l'équation 8.2 dans l'équation 8.3, et après simplification, nous obtenons l'équation suivante :

$$I_{xx} \sin^2(\hat{\theta}) + I_{yy} \cos^2(\hat{\theta}) - (I_{xy} + I_{yx}) \cos(\hat{\theta}) \sin(\hat{\theta}) = 0 \tag{8.4}$$

dans laquelle I_{xx} représente la dérivée de l'image selon l'axe x, et I_{xy} le terme de dérivée croisée. Après discrétisation et approximation des dérivées dans un voisinage 3×3 centré autour du pixel $I(m,n)$ (voir Figure 8.4), nous obtenons les équations suivantes :

$$\begin{aligned} I(m,n) = &\frac{1}{4} \left(I(m+1,n+1) + I(m-1,n+1) + I(m+1,n-1) + I(m-1,n-1) \right) + \\ &\frac{1}{2} \left(I(m+1,n+1) + I(m-1,n-1) - I(m-1,n+1) - I(m+1,n-1) \right) \cos(\hat{\theta}) \sin(\hat{\theta}) \end{aligned}$$

Cette formulation permet de pondérer les pixels originaux qui composent le voisinage proche du pixel à interpoler, en fonction de la direction détectée pour ce pixel.
L'interpolation se déroule donc en deux étapes. La première consiste à interpoler les pixels dits centraux, qui sont entourés de pixels originaux. C'est le cas du pixel $I(m,n)$ de l'exemple donné en Figure 8.4. Ainsi, si la direction détectée est de 45°, le pixel central à interpoler sera le résultat d'une interpolation linéaire des deux pixels originaux alignés dans cette direction : $I(m-1,n+1)$ et $I(m+1,n-1)$. Notons que si la direction du pixel central est soit verticale soit horizontale, on

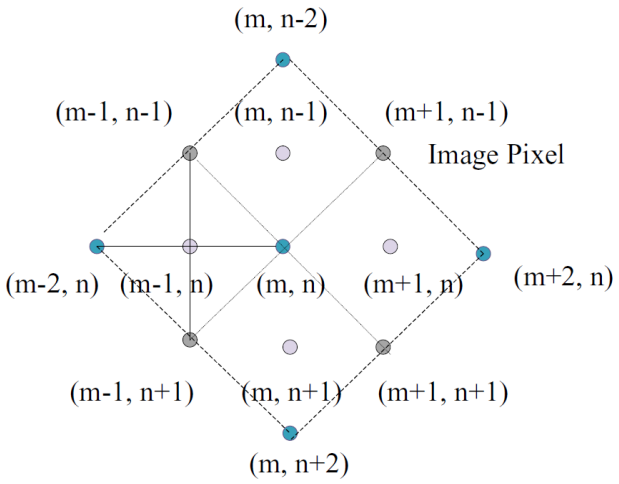


FIGURE 8.4 – Schéma d'interpolation pour la méthode IAD. Les pixels originaux sont représentés en bleu.

attribue à $I(m,n)$ la valeur moyenne des quatre pixels originaux qui l'entourent.
Le même schéma est appliqué pour l'interpolation des pixels restants, puisqu'ils sont maintenant également entourés de quatre pixels connus. Une simple rotation de 45° des positions de pixels suffit pour se retrouver exactement dans le même cas de figure.
Notons que la méthode présentée ici ne permet de réaliser que des interpolations d'un facteur de puissance de deux, par itération de ce même procédé.

3 Interpolation NEDI

L'interpolation NEDI (New Edge Directed Interpolation) publiée en 2001 par Li et Orchard : [Li 2001]. L'originalité de cette méthode vient du fait qu'elle ne nécessite pas d'estimation préalable des directions de contours. En effet, son principe de base consiste à calculer les covariances (variance à deux dimensions) locales de l'image originale et de faire en sorte que l'image interpolée possède localement les même covariances. La covariance est en effet une caractéristique pertinente et fortement liée aux contours, car la variance dans la direction perpendiculaire au contour est élevée, et qu'à l'inverse elle est faible dans la direction du contour. Ainsi en assurant la préservation de cette valeur dans l'image haute résolution, les caractéristiques des contours (de direction arbitraire) sont conservées.

Concrètement, les auteurs partent du principe qu'une image peut être considérée comme un processus Gaussien localement stationnaire. Les pixels interpolés sont donc obtenus à partir des pixels de la grille originale qui sont pondérés par des coefficients optimaux, obtenus par filtrage de Wiener et minimisation de l'erreur quadratique moyenne. Soit $\vec{\alpha}$ le vecteur contenant les coefficients optimaux et $(R, \vec{r})$ les covariances appelées *haute résolution*. On a alors le schéma suivant :

$$\vec{\alpha} = R^{-1}\vec{r} \tag{8.5}$$

Le problème revient donc à calculer les covariances dans l'image haute résolution, qui ne sont pas disponible au départ. Dans son article, Li montre que les covariances dites *haute résolution* peuvent être obtenues par correspondance directe des covariances dites *basse résolution*. A partir de cette hypothèse, les valeurs de covariances peuvent donc être calculées directement comme suit

(voir [Jayant 1984]) :

$$\begin{aligned} \vec{R} &= \frac{1}{M^2} C^T C \\ \vec{r} &= \frac{1}{M^2} C^T \vec{y} \end{aligned} \tag{8.6}$$

Dans ces équations, M représente le voisinage dans lequel l'interpolation est réalisée, $\vec{y}$ est le vecteur contenant tous les pixels du voisinage, et C est une matrice contenant les quatre pixels voisins diagonaux des pixels de $\vec{y}$. Par soucis de clarté, nous allons prendre l'exemple d'un voisinage de 2×2 pixels (voir schéma 8.5) autour du pixel à interpoler. Le calcul des covariances nécessitent requiert cependant un voisinage de 4×4 pixels. En pratique, les voisinages utilisés ont une taille d'au moins 4×4 pixels, ce qui revient à calculer les covariances dans un voisinage de taille 6×6 pixels.

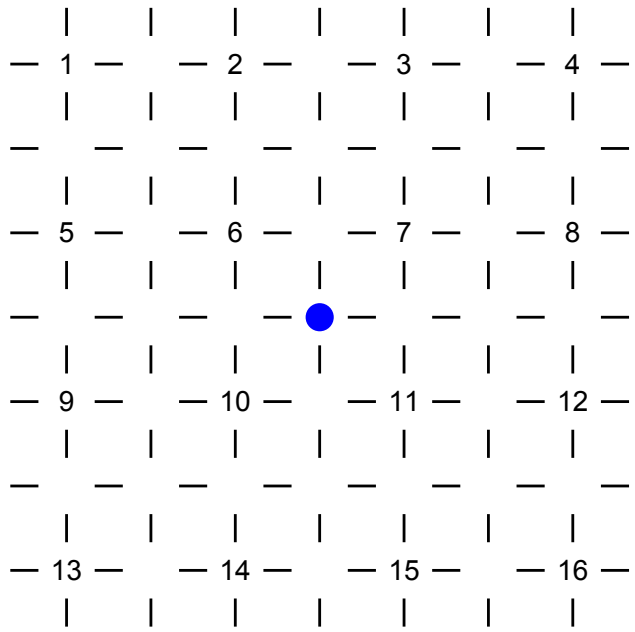


FIGURE 8.5 – Schéma illustrant la création des variables pour le calcul de la covariance.

Dans cet exemple, les nombres indiqués correspondent aux indices des pixels de la grille originale, les pixels manquants sont laissés vides et le pixel à interpoler est représenté en bleu. Dans ce cas, les valeurs des variables sont :

$$y^T = [6, 7, 10, 11]$$

$$C = \begin{bmatrix} 1 & 3 & 9 & 11 \\ 2 & 4 & 10 & 12 \\ 5 & 7 & 13 & 15 \\ 6 & 8 & 14 & 16 \end{bmatrix}$$

Notons que ce schéma d'interpolation n'est valable que pour la configuration ou le pixel à interpoler est entouré de quatre pixels connus. L'interpolation se fait donc en deux passes, la première consiste à n'interpoler uniquement les pixels se trouvant dans cette situation. Une fois celle-ci réalisée, les pixels restants se trouveront également entourés de quatre pixels connus et le même schéma pourra être appliqué après rotation des indices de pixels de 45°.

Une fois les covariances calculées, les équations 8.5 et 8.6 peuvent être combinées pour l'obtention des coefficients $\vec{\alpha}$.

$$\vec{\alpha} = (C^T C)^{-1}(C^T \vec{y}) \tag{8.7}$$

Ces coefficients obtenus permettent la pondération optimale des pixels originaux pour l'interpolation des pixels manquants, en assurant la conservation de la covariance *basse résolution* dans la grille haute résolution. La conservation de la covariance assure donc par la même occasion la conservation de l'aspect des contours de l'image originale dans l'image haute résolution, sans l'introduction d'artefacts supplémentaires.

4 Interpolation NOAI

La technique NOAI (New Orientation Adaptive Interpolation) a été introduite par Wang et Ward en 2007 dans [Wang 2007]. Elle propose une approche originale du problème d'interpolation puisqu'elle cherche à n'interpoler de façon directionnelle uniquement les pixels situés sur des contours rectilignes dont l'orientation a préalablement été estimée. Les autres pixels sont interpolés au moyen d'une technique classique (bicubique en l'occurrence). Ce mélange de deux interpolations permet de ne pas introduire d'artefacts supplémentaires dans les zones ou l'information sur la direction d'éventuels contours n'est pas pertinente (régions sans contours, textures non directionnelles), et d'éviter l'apparition d'artefacts classiques sur les contours (jaggy, effets d'escalier) grâce à la prise en compte de l'information de direction lors de l'interpolation. La principale difficulté de cette méthode consiste donc à détecter et situer les contours rectilignes, aussi appelé *isophotes* (contours d'équi-intensité) ou encore *level-sets contours*.
Certaines études préalables se sont également concentrées sur la détection d'isophotes afin d'en imposer la régularité, et de produire une image interpolée dont les contours ne présentent pas d'artefacts. Parmi celles-ci nous pouvons citer les travaux de [Morse 2001], [Aly 2003] et [Ballester 2001]. Contrairement à l'interpolation NOAI, ces méthodes utilisent des approches variationnelles qui consistent à minimiser de façon itérative les irrégularités le long des isophotes détectés pour éviter les artefacts mentionnés plus haut. En plus de la complexité algorithmique et du nombre important de calculs nécessaires, ces méthodes introduisent souvent un effet d'à plat (ou cartoon) sur les régions traitées.
Dans un premier temps, nous allons résumer la méthode de détection de direction et de discrimination des contours rectilignes de la méthode NOAI, puis nous présentons dans un deuxième temps la technique d'interpolation associée.

4.1 Détection des isophotes

Afin de détecter la direction des contours, l'opérateur de Sobel est utilisé afin d'estimer les gradients horizontaux (G_x) et verticaux (G_y) de l'image originale et d'en déduire l'image contenant les directions des gradients : θ^T.

$$\theta(x,y)^T = \tan^{-1}\left(-\frac{G_x(x,y)}{G_y(x,y)}\right) \tag{8.8}$$

Cette méthode a été étudiée dans ce document dans la partie 4.1. Afin d'attribuer une direction à tous les pixels de la grille haute résolution, les directions obtenues sont interpolées avec un noyau bicubique. Notons que seuls les pixels ayant une norme de gradient supérieure à un certain seuil sont interpolés de façon directionnelle, de façon à ne pas influencer ceux qui n'appartiennent pas à des contours.

A partir de cette image de directions, la courbure des pixels est calculée. En effet, seuls les pixels ayant une courbure faible pourront être interpolés correctement avec la méthode utilisée ici, qui

part de l'hypothèse que les pixels sont situés sur des contours rectilignes. L'exemple de la Figure 8.6 montre que dans le cas d'un contour courbe, les pixels choisis pour l'interpolation (sommets du parallélogramme) du pixel $F(x, y)$ ne sont pas représentatifs du contour à interpoler. Il faudrait en effet que le contour soit contenu à l'intérieur du parallélogramme ABCD.

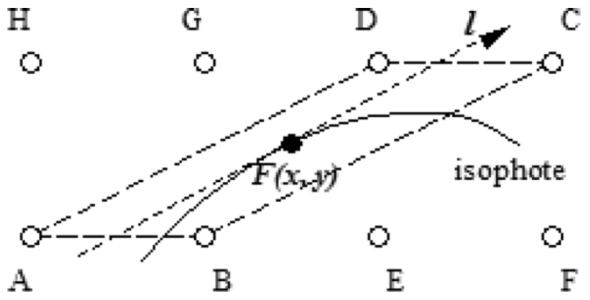


FIGURE 8.6 – Exemple d'un contour possédant une forte courbure, et où le schéma d'interpolation de NOAI ne peut être appliqué (d'après [Wang 2007]).

La courbure des contours est donc calculée de la manière suivante en utilisant les valeurs des gradients calculées plus haut :

$$\gamma(x,y) = \left| \frac{-G_x^2 G_{xx} + 2G_x G_y G_{xy} - G_y^2 G_{yy}}{\left(G_x^2 + G_y^2\right)^{\frac{3}{2}}} \right| \tag{8.9}$$

Les valeurs G_{xx}, G_{yy} et G_{xy} correspondent aux dérivées des gradients obtenus grâce à l'opérateur de Sobel. Les auteurs précisent que la valeur de G_{xy}, pas directement obtenable dans le domaine discret, est définie comme la moyenne de $\frac{\partial G_x}{\partial y}$ et de $\frac{\partial G_y}{\partial x}$.
La valeur de courbure obtenue est donc seuillée pour chaque pixel, afin de ne sélectionner uniquement les pixels ayant des valeurs de courbures faibles. Ainsi, seuls les pixels appartenant à des contours rectilignes seront traités avec la méthode d'interpolation directionnelle. Nous allons maintenant expliciter la méthode d'interpolation.

4.2 Interpolation directionnelle

La direction calculée pour chaque pixel de la grille haute résolution nous permet de connaître la direction le long de laquelle les niveaux de gris doivent être réguliers (isophotes). Dans le cas d'une interpolation classique isotrope, seuls les pixels connexes sont utilisés pour l'interpolation. L'information de direction nous permet de répartir les poids donnés aux pixels de la grille originale en fonction de la direction du contour, et d'élargir le voisinage d'interpolation pour inclure les pixels originaux porteurs de l'information de direction.

La première étape du processus consiste à sélectionner les quatre pixels de la grille originale qui vont servir à l'interpolation de $F(x, y)$ et définir ainsi le parallélogramme d'interpolation entourant $F(x, y)$. La Figure 8.7 illustre la création d'un parallélogramme pour une direction donnée. Dans cet exemple, la droite l de direction θ^T est tracée est les intersections avec la grille originale sont nommée t_1 et t_2. Les points de la grille originale les plus proches de ces intersections sont alors sélectionnés comme étant les sommets du parallélogramme. Une fois ce dernier défini, le schéma d'interpolation suivant est appliqué au pixel $F(x, y)$:

$$F(x,y) = A + (B-A)\cos(\theta)x' + (D-A)y' + (A+C-B-D)\cos(\theta)y'. \tag{8.10}$$

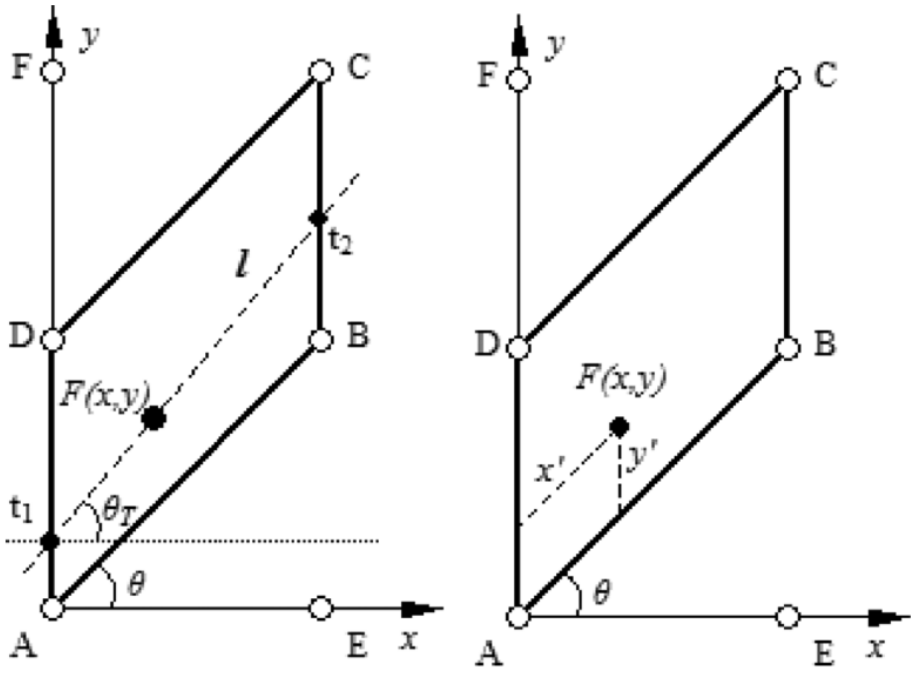


FIGURE 8.7 – Exemple de création du parallélogramme d'interpolation pour une direction donnée par la droite l.

Nous pouvons remarquer que l'angle θ^T de la droite l, est assimilé à l'angle θ formé par le grand coté du parallélogramme. Plus le voisinage d'interpolation est grand, plus le nombre de parallélogramme que l'on peut construire est élevé. L'erreur d'approximation réalisée sur la direction réelle du contour est donc d'autant plus faible que le nombre de parallélogrammes possibles est grand. Cependant, il n'est pas toujours possible d'utiliser des voisinages d'interpolation trop grands, c'est-à-dire de sélectionner les sommets du parallélogramme trop éloignés du pixel à interpoler. En effet, cela réduirait probablement la cohérence du pixel interpolé dans son environnement proche. Ainsi, les auteurs choisissent de limiter la recherche des sommets dans un voisinage de 5×5 pixels centré sur le pixel A. Les directions associées aux parallélogramme sont donc limitées aux suivantes : $\{0°, 11.3°, 14.0°, 18.4°, 26.6°, 45.0°, 63.4°, 71.6°, 76.0°, 78.7°, 90\}$. Remarquons que l'interpolation des angles proches ou égaux à 0° et 90° se font dans un carré (ADBE dans l'exemple), identique à celui qui serait utilisé pour les interpolations classiques (bicubique ou bilinéaire).
Grâce à l'hypothèse faite sur l'aspect rectiligne des contours, ce procédé permet d'interpoler les pixels uniquement à partir de pixels originaux appartenant à ce même contour. Les artefacts de jaggy, d'effets d'escalier et même de flou sont donc réduits.

Chapitre 9

Interpolation basée sur l'analyse directionnelle

La plupart des approches récentes visant à interpoler des images essaient de préserver au mieux les aspects des contours, puisqu'il est admit que la qualité de ces derniers est fortement liée à la qualité globale perçue d'une image. L'information de direction des contours que nous avons étudiée jusqu'à maintenant est un des moyens de prendre en compte les caractéristiques des contours pour les représenter au mieux dans un espace plus résolu. Elle permet en effet de connaître la direction le long de laquelle les niveaux de gris doivent avoir une certaine régularité, et à l'inverse celle le long de laquelle les transitions doivent rester franches.
Cependant cette hypothèse de régularité, si elle permet d'améliorer le rendu des contours dans la plupart des cas peut engendrer des dégradations importantes si elle n'est pas appliquée au bon endroit ou avec la bonne direction. La mise en pratique concrète de ces méthodes peut donc dans certains cas dégrader l'aspect global de l'image en créant des zones de régularité inattendues et peu naturelles dans l'image interpolée. C'est la raison pour laquelle la plupart des méthodes d'interpolation directionnelles cherchent à localiser les régions dans lesquelles l'information de direction est pertinente pour l'interpolation, afin de préserver l'aspect naturel des autres régions. Dans les méthodes présentées précédemment, cette localisation est réalisée à l'aide de techniques plus ou moins évoluées, du simple calcul de la norme du gradient (voir partie 8.2), à la recherche d'isophotes rectilignes par seuillage de la courbure des directions du gradient (voir partie 8.4).
Nous verrons en partie 10.4.3 que la localisation des pixels à interpoler de façon directionnelle est souvent très délicate à réaliser puisqu'en cas de mauvaise détection, des artefacts importants apparaissent dans les images interpolées (faux pixels et faux contours). Nous verrons également qu'indépendamment de la localisation des pixels à interpoler de façon directionnelle, le noyau d'interpolation en lui-même peut entraîner des dégradations. En effet, les méthodes utilisant des noyaux de faibles tailles (interpolations AQua et IAD) ne parviennent pas à éviter l'apparition des artefacts classiques sur les contours, et à l'inverse l'utilisation d'un grand noyau d'interpolation (parallélogramme de l'interpolation NOAI) peut entraîner un manque de cohérence entre le pixel interpolé et son voisinage proche. Il est donc obligatoire de réaliser un compromis entre la cohérence des pixels interpolés dans leur voisinage proche et l'éloignement des pixels originaux utilisés pour l'interpolation directionnelle.

La technique que nous présentons dans cette partie vise à optimiser le rendu visuel des images interpolées, en **évitant l'apparition des artefacts classiques** dus à l'interpolation tout en **conservant l'aspect naturel** des images.
Nous introduisons donc dans un premier temps notre technique d'interpolation, basée sur l'interpolation spline. Le fait de prendre comme référence ce noyau d'interpolation classique permet de conserver l'aspect naturel de l'image originale. Afin d'éviter l'apparition des artefacts habituellement engendrés par cette méthode, une contrainte de régularité des niveaux de gris est ajoutée

a posteriori, lorsque les pixels se trouvent sur un contour. Cette contrainte peut être vue comme une correction de l'aspect des contours donnés par l'interpolation spline, en vue d'assurer leur régularité.
Dans un deuxième temps, nous introduisons notre technique de localisation des contours qui permet de savoir quelles régions contiennent des contours à traiter de manière directionnelle. Cette localisation est basée sur un filtrage de Gabor qui permet de créer un masque (non binaire).

1 Interpolation spline et correction par filtrage Gaussien

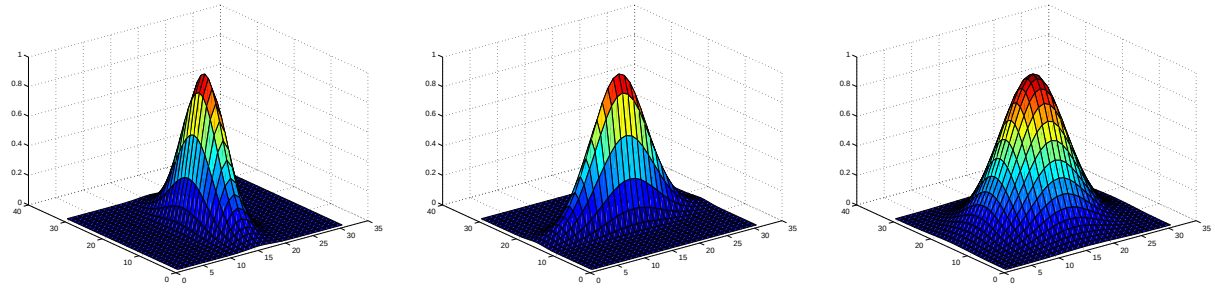
L'interpolation spline, introduite en partie 3 est souvent considérée comme un bon compromis entre rendu de l'image interpolée et quantité de calcul. Ce sont pour ces raisons que nous prenons cet interpolateur comme point de départ de notre interpolation. Pour ajouter la contrainte de régularité, nous utilisons un filtre Gaussien à deux dimensions qui est détaillé dans la partie suivante.

1.1 Filtres Gaussiens à deux dimensions

La formule d'une Gaussienne à deux dimensions est rappelée ici :

$$G(x,y) = Ae^{-\left(\frac{(x-x_0)^2)}{2\sigma_x^2}+\frac{(y-y_0)^2)}{2\sigma_y^2}\right)} \tag{9.1}$$

Ces filtres sont en particulier utilisés pour réaliser des flous sur des images, du fait de leur caractéristiques passe-bas. Lorsqu'un filtre Gaussien normalisé est appliqué à une image, la conservation de l'information initiale est garantie par le fait que le coefficient central correspondant au pixel filtré est le plus fort. Les coefficients correspondants aux pixels voisins ont des valeurs qui décroissent avec l'éloignement du pixel filtré, permettant ainsi de diffuser l'information de son voisinage. Les caractéristiques passe-bas de ce filtre viennent du fait que la transformée de Fourier d'une Gaussienne est également une Gaussienne. Un exemple de filtre Gaussien à deux dimensions est montré en Figure 9.1 pour différentes combinaisons des paramètres σ_x et σ_y. Ces deux paramètres de variances permettent d'étirer la Gaussienne dans l'une des deux directions du repère (horizontale ou verticale).



(a) (b) (c)

FIGURE 9.1 – Exemple de Gaussiennes à deux dimensions pour (a) $\sigma_x = 2$, $\sigma_y = 5$ (b) $\sigma_x = 5$, $\sigma_y = 2$ et (c) $\sigma_x = 5$, $\sigma_y = 5$

.

Nous utilisons donc les caractéristiques de diffusion de ces filtres pour lisser les contours de l'interpolation spline le long de leur direction. Afin de ne pas introduire de flou qui nuirait au rendu final de l'image, le filtre Gaussien est orienté dans la direction du contour. Le fait de diffuser l'information le long des contours permet donc d'assurer la cohérence des pixels dans cette direction et d'empêcher l'apparition d'artefacts sur ce contour. Pour orienter le filtre Gaussien dans la

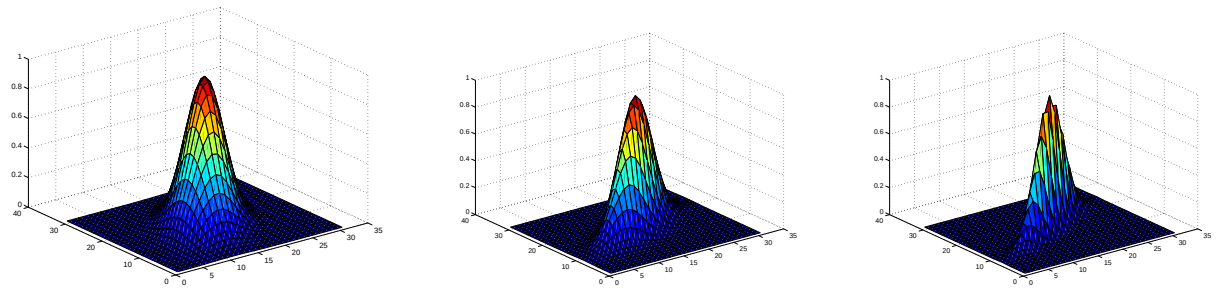
direction voulue, il est nécessaire de construire une Gaussienne dite elliptique, qui fait intervenir les termes croisés $(x - x_0)(y - y_0)$ permettant ainsi d'étirer la Gaussienne dans une direction arbitraire. La formule d'une Gaussienne elliptique s'écrit donc :

$$G(x, y) = Ae^{-a(x-x_0)^2+2b(x-x_0)(y-y_0)+c(y-y_0)^2} \tag{9.2}$$

Avec les paramètres suivants :

$$\begin{aligned} a &= \frac{\cos^2(\theta)}{2\sigma_x^2} + \frac{\sin^2(\theta)}{2\sigma_y^2} \\ b &= \frac{-\sin(2\theta)}{4\sigma_x^2} + \frac{\sin(2\theta)}{4\sigma_y^2} \\ c &= \frac{\sin^2(\theta)}{2\sigma_x^2} + \frac{\cos^2(\theta)}{2\sigma_y^2} \end{aligned}$$

Ce changement de formulation permet d'introduire l'orientation θ souhaitée, et de conserver les deux mêmes paramètres de variance : σ_x et σ_y qui correspondent maintenant à la variance le long de la direction et dans la direction perpendiculaire. Par soucis de clarté, nous appelons tout au long du chapitre $\sigma_\theta = \sigma_y$ et $\sigma_{\theta^\perp} = \sigma_x$ respectivement la variance dans la direction θ et la variance dans la direction perpendiculaire à θ. La Figure 9.2 illustre plusieurs Gaussiennes pour diverses orientations et plusieurs valeurs de variances.



(a) (b) (c)

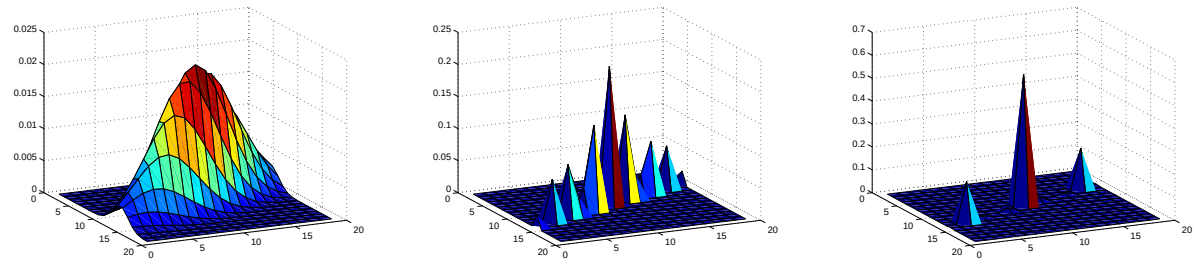
FIGURE 9.2 – Exemple de Gaussiennes elliptiques pour (a) $\theta = 120°$, $\sigma_\theta = 5$, $\sigma_{\theta^\perp} = 3$ (b) $\theta = 150°$, $\sigma_\theta = 5$, $\sigma_{\theta^\perp} = 2$ et (c) $\theta = 150°$, $\sigma_\theta = 5$, $\sigma_{\theta^\perp} = 1$

Le correcteur que nous introduisons dans cette partie utilise donc la direction détectée préalablement par l'analyse directionnelle. Nous rappelons que cette analyse consiste à diviser l'image originale grâce à un *quadtree* qui permet d'isoler dans des blocs au plus une direction de contour. Par soucis de clarté, les exemples présentés seront des cas de blocs isolés, qui contiennent une direction de contour unique et correctement détectée. La partie suivante détaille comment l'information de direction donnée par le *quadtree* est exploitée en vue de construire le filtre Gaussien adapté.

1.2 Construction du correcteur

1.2.1 Choix des paramètres

Parmi les trois paramètres qui permettent de construire le filtre correcteur, seul celui de l'angle est pour l'instant connu. Il reste donc à déterminer les valeurs de variances qui donneront les résultats optimaux. Le but étant de diffuser l'information des pixels appartenant au contour dans la direction de celui-ci, et non pas dans la direction perpendiculaire, il apparaît logique de choisir des valeurs de σ_θ fortes, et des valeurs de $\sigma_{\theta^\perp}$ faibles. En effet, cela permet d'éviter de filtrer des pixels n'appartenant pas au même objet, en limitant également l'introduction de flou.
La Figure 9.3 montre des exemples de correcteurs lorsque $\sigma_{\theta^\perp}$ diminue. A partir d'une certaine



(a) (b) (c)

FIGURE 9.3 – Correcteurs pour plusieurs valeurs de $\sigma_{\theta^\perp}$. (a) $\sigma_{\theta^\perp} = 1.5$. (b) $\sigma_{\theta^\perp} = 0.15$. (c) $\sigma_{\theta^\perp} = 0.05$. La valeur de σ_θ est fixée à 7.

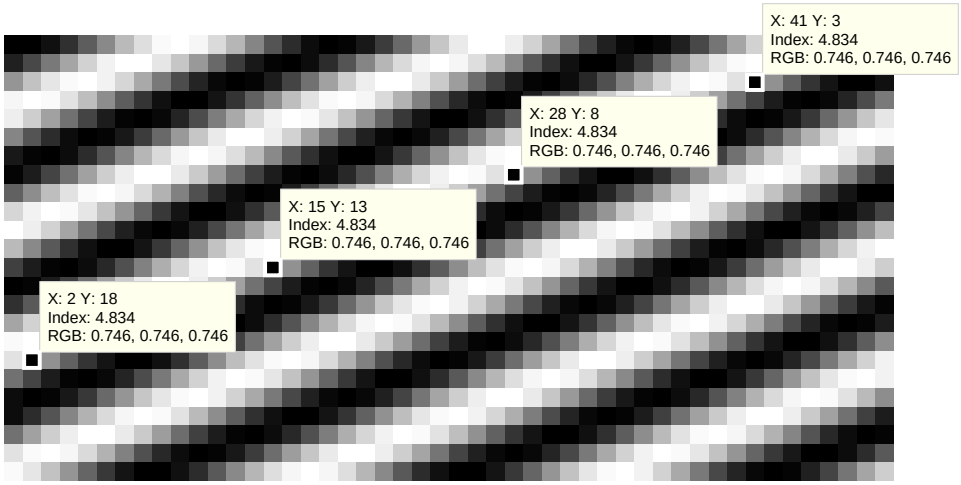
valeur, seuls quelques coefficients ne sont pas nuls. Le meilleur rendu des images est obtenu avec $\sigma_{\theta^\perp} = 0.1$. Nous verrons que le choix de σ_θ dépend de la direction à filtrer, mais qu'il est entre vingt à soixante fois supérieur à $\sigma_{\theta^\perp}$.

1.2.2 Lien avec les droites discrètes fines

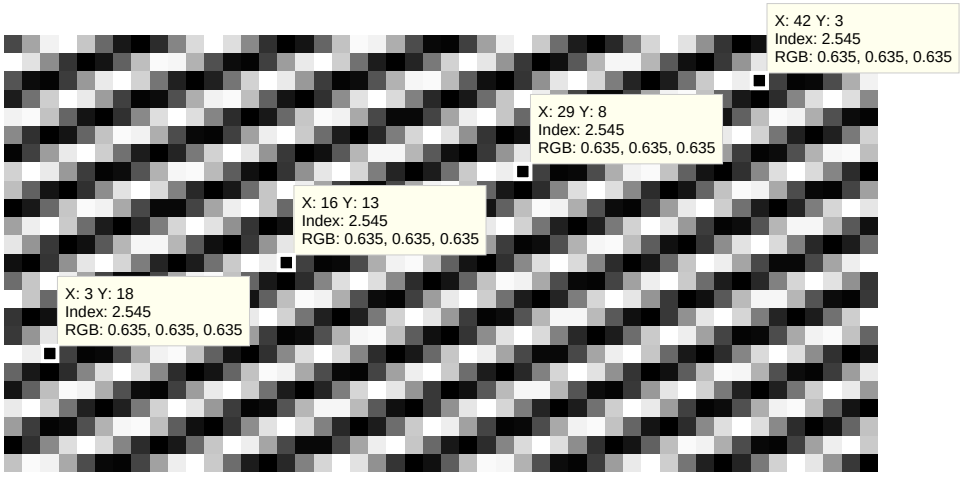
Il est intéressant de noter que plus $\sigma_{\theta^\perp}$ diminue, plus l'emplacement des coefficients non nuls du correcteur se situent sur une droite discrète dont l'épaisseur ω diminue. En admettant que σ_θ soit infinie (pas de décroissance de l'amplitude de la Gaussienne dans le direction du contour), et que $\sigma_{\theta^\perp}$ soit suffisamment faible, les coefficients non nuls du correcteur se situent sur une droite discrète finie ($\omega = 1$) de même orientation.
Le fait que les coefficients du filtre soient situés sur une droite discrète fine justifie l'utilisation de tels correcteurs. En effet, nous avons déjà remarqué que la discrétisation d'une direction par une droite discrète fine était la représentation la plus précise. Au niveau du correcteur, cela permet de filtrer l'image en utilisant les pixels de la grille discrète qui sont situés exactement dans la direction du contour, et ainsi de diffuser de manière précise l'information contenue par les pixels orientés selon cette direction. En effet, soit un contour de direction θ représenté sur une grille discrète, l'information de direction est portée par les pixels éloignés de (a, b) tels que $\theta = \tan^{-1}\left(\frac{a}{b}\right)$. Pour illustrer cette affirmation, reprenons l'exemple des sinusoïdes à deux dimensions utilisées pour l'évaluation des méthodes de détection d'orientation. Soit θ l'orientation des sinusoïdes et (a, b) les paramètres discrets de cette direction. Il est alors possible de remarquer dans l'exemple de la Figure 9.4, que deux pixels successifs éloignés de (a, b) ont la même valeur. Dans cet exemple, nous avons choisi $(a, b) = (13, 5)$. Remarquons également que ce principe se vérifie également lorsque le taux d'échantillonnage des sinusoïdes est supérieur à celui de la grille de pixel sur laquelle elles sont projetées.
C'est d'ailleurs de ce principe que part la méthode de la BT lorsqu'elle cherche à estimer l'orientation d'un contour, en détectant la droite discrète fine la plus régulière. Nous avons cependant observé lors de l'étude des erreurs de détection que cette méthode de projection n'était pas optimale pour l'estimation de direction (en particulier dans les petits voisinages). Cependant, elle nous parait la plus adaptée pour corriger et diffuser l'information de direction d'un contour.

1.2.3 Gestion de la cohérence du pixel corrigé

Comme nous l'avons remarqué précédemment, certaines directions nécessitent un grand voisinage pour être représentées par une droite discrète fine. C'est le cas des directions dont au moins un des deux paramètres discrets (a, b) est important. Lorsqu'une telle direction doit être corrigée, les coefficients du correcteur sont logiquement très éloignés du pixel central à corriger. Cela peut nuire à la cohérence du pixel corrigé dans son voisinage proche dans le cas ou les pixels éloignés



(a)

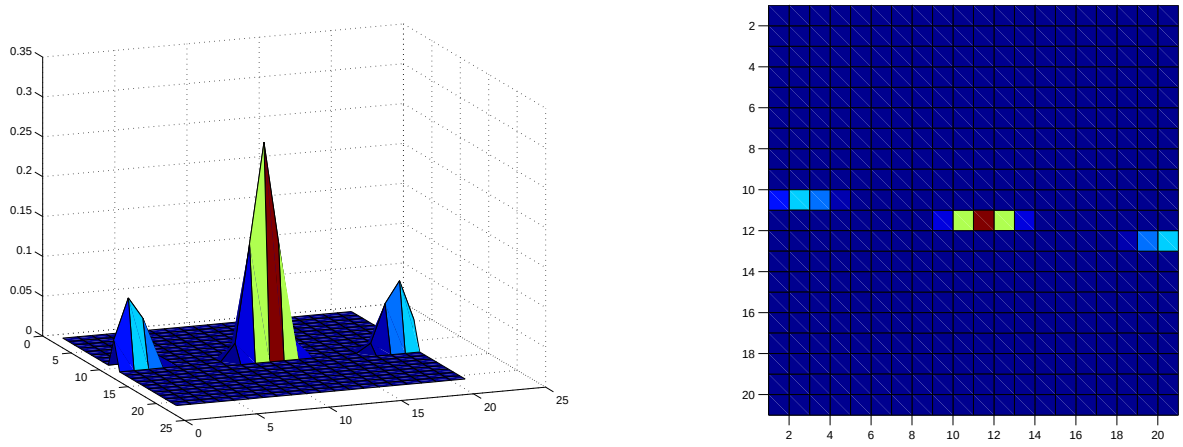


(b)

FIGURE 9.4 – Illustration de la régularité des sinusoïdes sur une droite discrète fine dont la direction a les paramètres discrets $(a, b) = (13, 5)$, pour les fréquences (a) f_1 et (b) f_3.

utilisés pour la correction n'appartiennent plus au même objet que le pixel à traiter. C'est à ce niveau qu'intervient le paramètre de variance σ_θ. Il joue le rôle de modérateur puisqu'il permet de réduire le poids donné aux pixels éloignés, par rapport au poids attribué au pixel central. Ce paramètre est donc proportionnel à la taille du filtre nécessaire à la correction, pour faire en sorte que le coefficient correspondant aux pixel central ait un poids d'environ un tiers de la norme totale du filtre.

De plus, comme nous l'avons précisé plus haut, la valeur de $\sigma_{\theta^\perp}$ est égale à 0.1 et fait en sorte que la droite sur laquelle sont situés les coefficients du filtre ne soit pas exactement fine. Ceci permet l'apparition de *pixels intermédiaires* situés entre le pixel central et les *pixels correcteurs* lorsque ceux-ci sont trop éloignés. Nous appelons *pixels correcteurs* les pixels éloignés de (a, b) du pixel central, théoriquement idéaux pour corriger une direction puisqu'ils sont situés sur la **droite discrète fine** dans la direction du contour. Les *pixels intermédiaires* sont également alignés dans la direction du contour mais constituent une représentation moins précise de cette direction en étant situés sur une **droite discrète non fine**. L'apparition des pixels intermédiaires pondèrent automatiquement le poids donnés aux pixels correcteurs lorsque ceux-ci sont éloignés, ce qui permet d'effectuer une correction moins précise au niveau de la direction mais moins risquée quant à la cohérence du pixel corrigé dans son voisinage. Un tel filtre est illustré en Figure 9.5, dans lequel il est possible de voir que les coefficients sont situés sur une droite non fine, et que les pixels



(a) (b)

FIGURE 9.5 – Filtre correcteur montrant l'apparence de coefficients intermédiaires pondérant l'amplitude des coefficients des pixels correcteurs.

correcteurs ont un poids relativement faible par rapport au pixel central. En pratique, cela donne des valeurs de σ_θ d'environ 2 à 6.

1.3 Application du correcteur

1.3.1 Quantification de l'erreur introduite sur la direction

Nous allons dans cette partie étudier l'erreur introduite par le correcteur lorsqu'il est appliqué sur les images de sinusoïdes à deux dimensions. Logiquement, un correcteur parfait composé uniquement des pixels correcteurs et appliqué dans la direction des sinusoïdes ne devrait pas modifier l'image puisque chaque pixel filtré est le résultat d'une convolution avec des pixels de mêmes valeurs. Cependant, l'apparition des pixels intermédiaires va introduire une erreur lors du filtrage que nous allons quantifier. Cette erreur est illustrée en Figure 9.6. Par soucis de clarté, nous imposons une variance longitudinale infinie pour que les erreurs soient comparables pour toutes les directions. Ceci implique que les coefficients du filtre correcteur ne décroissent pas lorsqu'ils s'éloignent du pixel central. A partir des images d'erreur obtenues, nous pouvons quantifier la norme de l'erreur pour différentes valeurs de variances transversales et différents angles.

Ainsi, pour une variance transversale très faible ($\sigma_{\theta^\perp} = 0.01$), l'erreur obtenue, calculée comme étant l'intégrale de la valeur absolue de la différence entre l'image originale et l'image filtrée, est de l'ordre de 1×10^{-13} soit quasi-nulle comme il est possible de le constater sur la Figure 9.6(c). A l'inverse, pour la valeur de variance utilisée pour la correction $\sigma_{\theta^\perp} = 0.1$, l'erreur introduite n'est plus négligeable comme illustré en Figure 9.6(e). De plus, l'erreur introduite dépend de la direction. En effet, plus les coefficients correspondant aux pixels correcteurs sont éloignés du pixel central, plus les coefficients intermédiaires prennent de l'importance. Ce sont eux qui introduisent l'erreur lors du filtrage. La courbe 9.7 montre l'évolution de la norme de l'erreur introduite en fonction de la distance des pixels correcteurs par rapport au pixel central. La présence des pixels intermédiaires se justifie cependant pleinement dans le cas d'images réelles où filtrer le pixel à corriger avec des pixels trop éloignés peut altérer la cohérence de ce pixel dans son voisinage connexe.

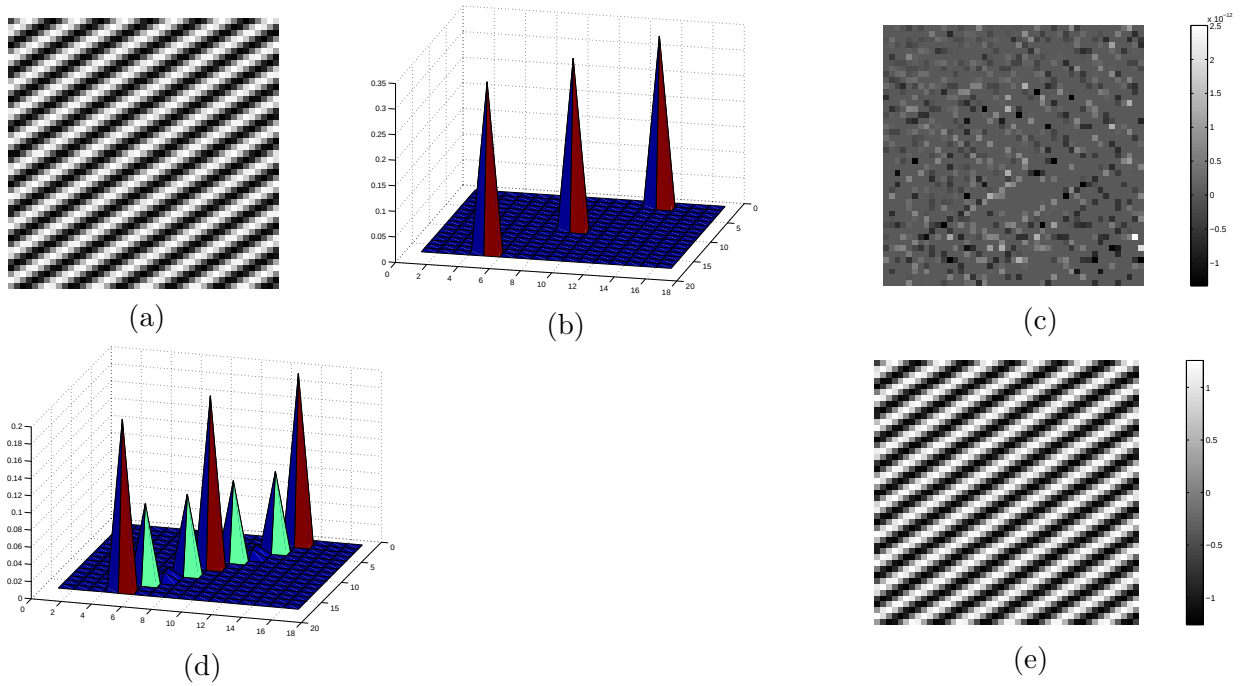


FIGURE 9.6 – Illustration des erreurs engendrées lors du filtrage de l'image (a) par les filtres suivants : (b) $\sigma_{\theta^\perp} = 0.01$ et (d) $\sigma_{\theta^\perp} = 0.1$. Les Figures (c) et (e) illustrent l'erreur obtenue respectivement pour les filtres (b) et (d).

1.3.2 Application sur des images interpolées

Pour montrer l'efficacité du correcteur, prenons tout d'abord l'exemple d'un filtrage sur une image de synthèse composée de droites rectilignes (voir Figure 9.8(a)). Cette image est interpolée par un noyau spline cubique par un facteur deux (en horizontal et vertical) (Figure 9.8(b)). Il est possible de voir sur l'image interpolée, les effets d'escaliers qui apparaissent sur les contours. Après application du filtre correctement orienté, les artefacts introduits par l'interpolation spline disparaissent ce qui permet d'obtenir des contours sans artefact (Figure 9.8(c)). La direction est correctement reconstruite grâce à la diffusion des pixels situés sur une droite *quasiment* fine, orientée dans la direction du contour.
Le même constat peut être effectué sur un extrait d'une image réelle contenant des rayures rectilignes. La Figure 9.9 indique la portion de l'image sélectionnée. Le résultat de l'interpolation spline par une facteur deux est montré en Figure 9.10(a), et l'image de sortie du correcteur est donnée en Figure 9.10(b). Cet exemple met également en évidence les propriétés de reconstruction du correcteur en rendant aux contours de l'image un aspect régulier. Les variations parasites le long du contour sont en effet supprimées. De plus, la transition entre le contour et le fond est conservée ce qui n'engendre pas de flou ni de perte de contraste.

Si les résultats présentés sont visuellement satisfaisants dans les exemples de cette partie, il est cependant évident qu'un tel filtrage ne peut être appliqué sur toute l'image sans dégradation du contenu qui ne serait pas orienté dans la même direction que le correcteur (texture, fond de l'image, etc ...). La partie suivante explique donc la méthode mise au point pour sélectionner les zones à corriger, c'est-à-dire un contour correctement orienté, pour ne pas affecter les régions que le correcteur dégraderait.

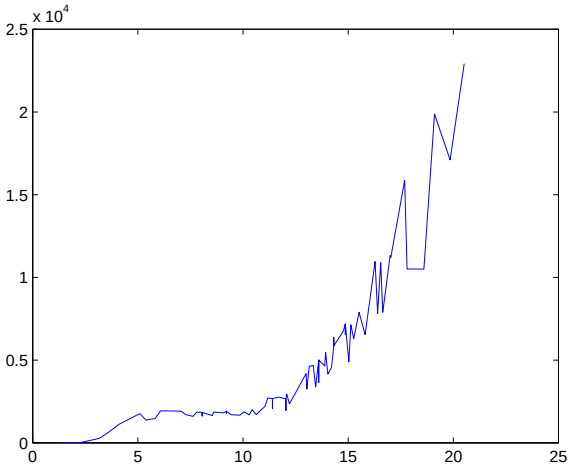


FIGURE 9.7 – Evolution de l'erreur de filtrage en fonction de la distance des pixels correcteurs par rapport au pixel central, pour une valeur de variance transversale $\sigma_{\theta^\perp} = 0.1$

(a) (b) (c)

FIGURE 9.8 – (a) Image originale (b) Interpolation spline cubique par un facteur 2 (c) Résultat du filtrage de l'interpolation spline corrigée.

(a)

(b)

FIGURE 9.9 – Image *Lighthouse*. Le rectangle bleu indique la portion de l'image sélectionnée pour le filtrage. L'extrait utilisé est montré en (b).

(a)

(b)

FIGURE 9.10 – (a) Résultat de l'interpolation spline. (b) Sortie du correcteur.

2 Localisation des zones à corriger

La localisation est effectuée grâce à des filtres de Gabor orientés dans la direction du correcteur. Ils permettent de sélectionner les régions correctement orientées sur lesquelles le correcteur sera appliqué. Nous commençons donc par décrire les filtres de Gabor puis nous expliquons dans un deuxième temps comment ils sont appliqués en vue de différencier les zones correctement orientées des autres.

2.1 Filtres de Gabor

Les filtres de Gabor créés par Denis Gabor [Gabor 1946], sont principalement utilisés dans la littérature pour l'étude des textures et la discrimination de zones dont l'orientation est une information pertinente. Ils ont été notamment étudiés par [Strohmer 1998], [Daugman 1988] dans un contexte d'application à l'analyse d'image.
Dans le domaine spatial, un filtre de Gabor est composé d'un noyau Gaussien modulé par une fonction harmonique. Il s'écrit de la manière suivante :

$$G(x, y, \lambda, \theta, \psi, \sigma, \gamma) = e^{-\left(\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)} e^{i\left(2\pi\frac{x'}{\lambda}+\psi\right)} \tag{9.3}$$

avec $x' = x\cos(\theta) + y\sin(\theta)$ et $y' = -x\sin(\theta) + y\cos(\theta)$. Les paramètres de ce filtre sont les suivants :

- λ permet de placer le filtre à la fréquence désirée en intervenant sur la fréquence de la fonction harmonique.
- θ correspond à l'orientation du filtre.
- ψ permet de décaler la phase de la fonction harmonique.
- σ correspond à la variance de la Gaussienne utilisée pour la modulation.
- γ correspond au ratio entre les deux variances de la Gaussienne (celles que nous avons appelé σ_θ et $\sigma_{\theta^\perp}$ plus haut) et permet d'étirer plus ou moins le filtre dans la direction θ.

Un exemple de filtre est montré en Figure 9.11.

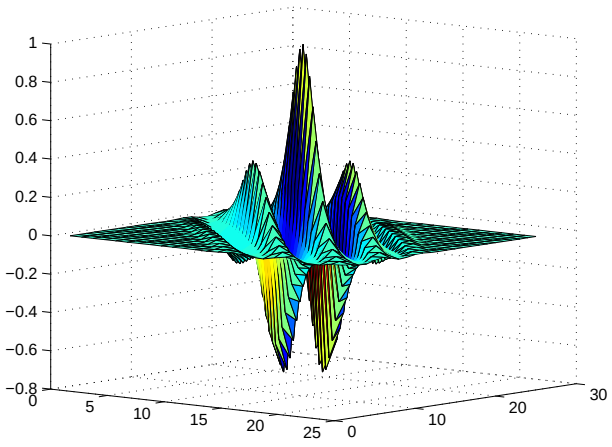
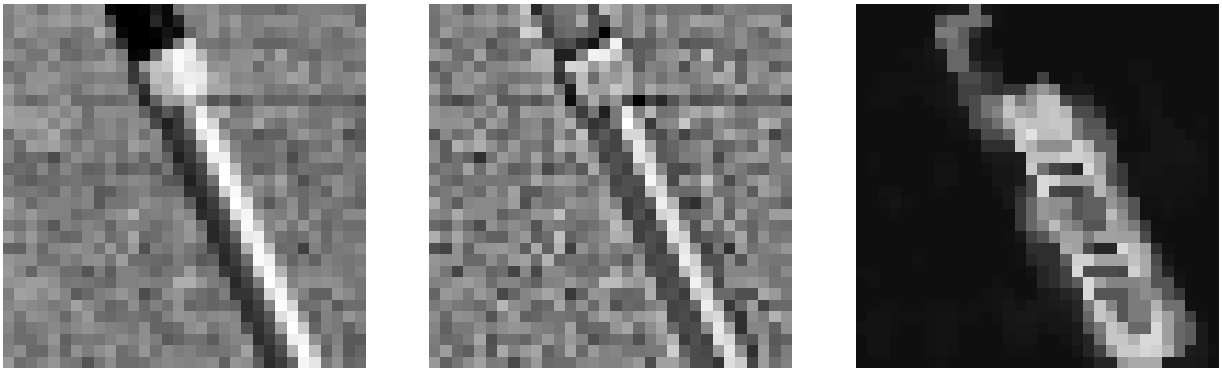


FIGURE 9.11 – Exemple de filtre de Gabor.

2.2 Création du masque de Gabor

Le résultat d'un filtrage de Gabor permet de faire ressortir les éléments de l'image orientés dans la direction du filtre. Ces filtres sont donc appliqués dans chacun des blocs créés par la



(a) (b) (c)

FIGURE 9.12 – (a) Bloc original. (b) Première échelle des détails de la transformée IUWT. (c) Sortie du filtrage de Gabor utilisée comme masque.

segmentation en *quadtree* qui, par construction, ne contient qu'une seule direction de contour. Le but est donc de détecter les pixels du bloc qui appartiennent au -ou aux- contours orientés dans la direction détectée.
Lorsqu'un filtre de Gabor est appliqué, les pixels de l'image appartenant à des objets orientés dans la direction du filtre donneront de forts coefficients. A l'inverse, les pixels situés sur des objets *mal* orientés ou appartenant à des zones sans contours, donneront de faibles coefficients. La sortie de ce filtre peut donc être utilisée comme un masque qui indique l'emplacement des pixels orientés dans la direction du filtre (et donc du correcteur). Notons que nous prenons tout au long de l'étude le module de la sortie du filtre de Gabor (complexe), pour construire le masque.

Afin de s'adapter à tous types de contours et d'éviter les perturbations dues à des éventuels dégradés de niveaux de gris parasites, le filtrage s'effectue sur l'image contenant les détails de l'échelle la plus fine de la transformée IUWT. Ceci permet d'homogénéiser les caractéristiques fréquentielles des contours à analyser, et donc de fixer à l'avance et de garder les mêmes paramètres (sauf l'orientation) du filtre de Gabor utilisé. Il aurait été logique de se servir de l'information d'échelle optimale donnée par l'analyse fréquentielle réalisée en partie 5, mais travailler sur des faibles échelles aurait entraîné une perte de précision spatiale sur les contours détectés. Or, une mauvaise localisation des pixels à corriger engendrerait des artefacts puisque le filtrage directionnel dégrade les zones non orientées dans la même direction que le filtre correcteur. Il est donc préférable de travailler sur l'échelle la plus fine, de façon à avoir une bonne précision spatiale, même si la représentation de tous les contours dans cet espace n'est pas optimale.

Un exemple de masque de Gabor est présenté dans la Figure 9.12. Ce masque est construit à partir d'un bloc contenant un contour rectiligne au milieu d'une zone de texture. Il est possible de constater à partir de l'étude du masque illustré en Figure 9.12(c), que l'emplacement des forts coefficients du masque correspondent à l'emplacement du contour du bloc. Le masque ainsi créé permet de différencier le contour de la texture, et permet d'éviter la détérioration des autres régions.

En pratique, le masque est créé à partir d'un bloc à la même résolution que l'image originale. En effet, si le masque était créé à partir d'un bloc interpolé, les artefacts introduits lors de l'interpolation perturberaient l'analyse. Par contre, ceci implique que le masque une fois construit, doit être interpolé pour avoir la même résolution que le bloc agrandi par interpolation spline sur lequel il est appliqué. Après plusieurs tests, il est apparu que l'interpolation du masque plutôt que du bloc permettait de construire des masques plus précis. Enfin, l'analyse présentée dans cette partie pour un bloc particulier, est appliquée à chaque bloc de la partition donnée par le *quadtree*, et

donc pour chacune des directions détectées.
D'autre part, il est important de préciser que les filtres de Gabor sont appliqués sur un voisinage plus grand que le bloc à traiter. Concrètement, ils sont appliqués sur un support de taille $(3N \times 3N)$ englobant le bloc à traiter de taille $(N \times N)$. Cela permet d'éviter les effets de bords à l'intérieur du bloc à traiter lors du filtrage, et de mieux capter les orientations des contours situés dans les coins du bloc qui seraient difficiles à traiter si le voisinage n'était pas étendu.

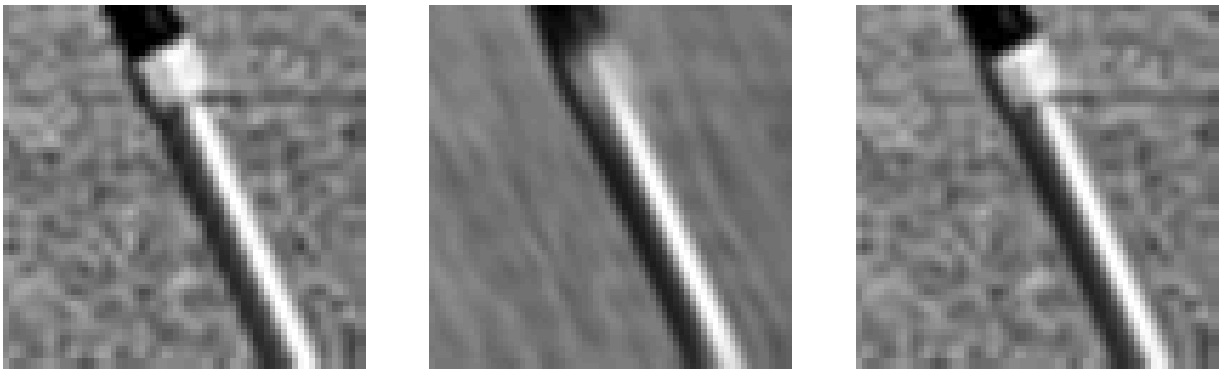
2.3 Schéma général de l'interpolation

Le calcul du masque de Gabor $G(x, y)$, ainsi que du bloc corrigé $C(x, y)$ obtenu par le filtrage Gaussien, permettent de définir le schéma d'interpolation global de la méthode. Ces deux images étant obtenues séparément pour chaque bloc du *quadtree*, l'interpolation est également effectuée bloc par bloc. Soit $B_{interp}(x, y)$ le bloc final interpolé et $B_{spline}(x, y)$ l'interpolation spline du bloc, le schéma d'interpolation d'un bloc est le suivant :

$$B_{interp}(x, y) = C(x, y) \times G(x, y) + B_{spline}(x, y) \times (1 - G(x, y)) \tag{9.4}$$

Notons que dans cette équation, le masque $G(x, y)$ est normalisé de sorte à ce que ses valeurs soit comprises entre 0 et 1, pondérant ainsi l'interpolation spline avec la version corrigée. Le schéma global d'interpolation illustrant cette équation se trouve en Figure 9.14.
Les forts coefficients du masque correspondent à des pixels dont l'orientation est la même que celle du correcteur. Il est donc logique que ces pixels aient un poids important lors de la pondération pour la construction de l'image interpolée. A l'inverse, un poids important est donné aux pixels obtenus par l'interpolation spline lorsque les coefficients du masque de Gabor sont faibles. Le fait de se baser sur ce schéma d'interpolation permet de pondérer efficacement les zones à corriger (contours rectilignes) des zones où l'interpolation spline est considérée comme meilleure (régions sans contours ou textures). Ce principe est illustré en Figure 9.13, à partir du même bloc tiré de l'image *Cameraman* sur lequel nous avions procédé à la création du masque de Gabor. L'image 9.13(a) est l'interpolation spline du bloc par un facteur 2×2, l'image 9.13(b) est l'image filtrée dans la direction du contour, et enfin l'image 9.13(c) est le résultat de la pondération de (a) et (b) par le masque obtenu en Figure 9.12(c).
Cet exemple met en valeur l'importance d'une bonne localisation des contours à corriger, puisque la dégradation de la texture induite par la correction est importante. En revanche, il est primordial que le contour garde l'aspect de la version corrigée puisqu'il ne présente plus d'effet d'escalier une fois filtré.

Pour obtenir l'image entière, l'interpolation est successivement réalisée de cette manière sur tous les blocs de la partition du *quadtree*. Afin d'éviter l'apparition de frontières de blocs, un recouvrement est introduit entre chaque bloc. Ainsi, un bloc de $(N \times N)$ pixels est étendu à une surface de $\left(\frac{3}{2}N \times \frac{3}{2}N\right)$ pixels. Notons enfin que cette méthode a été publiée dans deux conférences récentes : [Van Reeth 2011b], [Van Reeth 2011a].



(a) (b) (c)

FIGURE 9.13 – (a) Bloc interpolé par le noyau spline par un facteur 2×2. (b) Bloc corrigé dans la direction du contour. (c) Bloc final obtenu après pondération de (a) et (b) par le masque de Gabor.

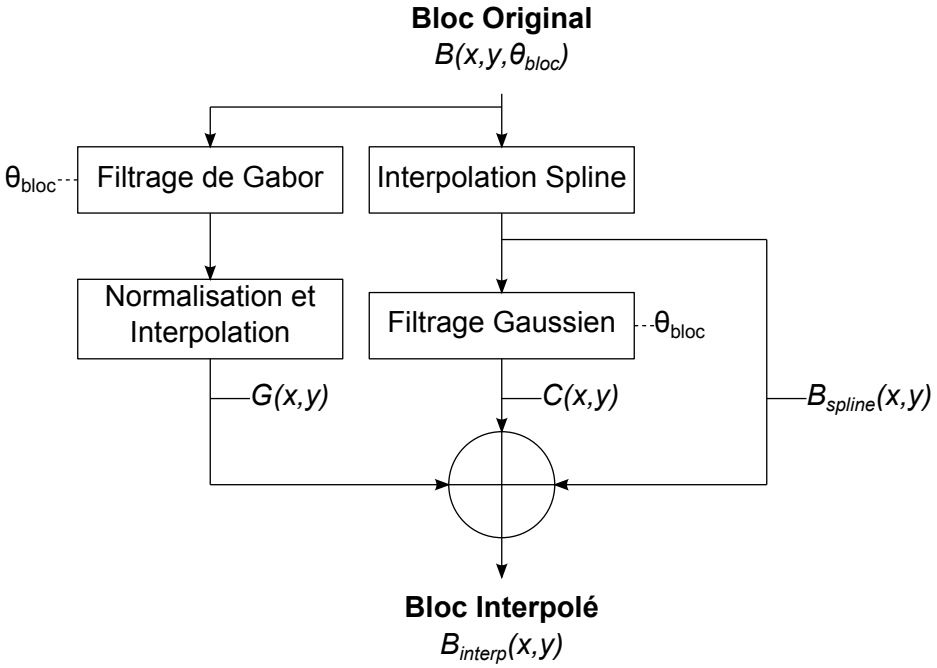


FIGURE 9.14 – Schéma global d'interpolation pour un bloc de la partition en *quadtree* de l'image.

2.4 Conclusion sur la méthode d'interpolation

Le schéma d'interpolation présenté dans cette partie est donc basé sur la détection de direction introduite précédemment. Le correcteur est orienté dans la direction détectée puis appliqué au bloc déjà interpolé avec un noyau spline. Le bloc final est ensuite construit grâce à une pondération entre l'interpolation spline de référence et la version corrigée. La pondération est réalisée grâce à un masque créé à partir d'un filtrage de Gabor orienté dans la direction détectée dans le bloc, permettant de localiser les pixels appartenant au contour à corriger. Cette technique permet de ne corriger que les pixels appartenant au contour et donc d'éliminer les artefacts introduits par l'interpolation spline. A l'inverse, elle permet également de ne pas dégrader l'aspect des zones sans contours ou les zones texturées, pour lesquelles l'interpolation spline donne de meilleurs résultats. L'étude du comportement du correcteur sur des images de sinusoïdes orientées nous a permis de voir que l'erreur introduite par le filtrage directionnel était faible, mais pas nulle. Cette erreur provient de la présence de coefficients intermédiaires situés entre le pixel central et les pixels correcteurs qui apparaissent à cause de la valeur du paramètre de variance transversale choisie pour le filtre Gaussien. Le choix de ce paramètre se justifie cependant pleinement puisque dans le cas de directions ou les pixels correcteurs seraient situés trop loin du pixel central, les coefficients intermédiaires permettent une correction non optimale mais toutefois efficace, grâce à des pixels situés relativement proche du pixel à corriger. Le risque de filtrer des pixels appartenant à des objets différents est alors fortement réduit.

La méthode de détection de direction joue donc un rôle prépondérant pour la qualité de l'image interpolée. En effet, la précision sur l'angle estimé influe sur la qualité du contour corrigé. Plus l'angle estimé sera proche de l'angle réel du contour, meilleure sera l'élimination des artefacts. En revanche, une mauvaise estimation de direction impliquera une perte de contraste et l'apparition de flou lors de la correction. En effet, lorsqu'il est mal orienté, le correcteur filtre des pixels du contour avec des pixels d'un objet hors contour, ce qui affecte la transition du contour et implique l'apparition de flou et une perte de contraste du contour par rapport à son environnement. On voit donc ici l'intérêt d'une méthode de détection de direction précise qui offre la possibilité de détecter un grand nombre d'orientations.
Le principe de *quadtree* utilisé pour la détection de direction est donc cohérent avec l'application d'interpolation qui en découle. En effet, le traitement par bloc utilisé permet l'estimation précise de la direction de contour en essayant de garder au maximum de grandes tailles de blocs afin d'optimiser la résolution angulaire de l'étude. Nous avons montré en partie 7.1 que même dans le cas ou de petites tailles de blocs doivent être choisies, l'erreur d'estimation avec notre technique reste faible. Enfin, l'algorithme de *quadtree* permet d'isoler une seule direction par bloc, ce qui autorise la construction d'un unique filtre correcteur par bloc. Notre estimation directionnelle basée sur une méthode de calcul de direction prédominante efficace, et sur une partition en *quadtree* apparaît donc adaptée à la technique d'interpolation introduite dans cette partie. Les chapitres suivants vont nous permettre d'évaluer notre méthode d'interpolation par rapport à celles introduites dans l'état de l'art. Des comparaisons objectives et subjectives seront données pour étudier l'apport de notre méthode par rapport à des techniques récentes d'interpolation directionnelles.

Chapitre 10

Evaluation et analyse de la méthode d'interpolation

Le but de cette section consiste à analyser de manière subjective les résultats de notre méthode d'interpolation introduite dans le chapitre précédent, ainsi que de comparer ces résultats avec ceux obtenus par les méthodes présentées en état de l'art en partie 4.6.2 qui sont les suivantes :

- La méthode AQua, disponible dans le logiciel libre *Visere*.
- La méthode IAD, que nous avons implémenté nous-même d'après l'article de [Jiang 2002].
- La méthode NEDI, disponible en accès libre sur le site Internet de l'auteur [Li 2001].
- La méthode NOAI, pour laquelle l'auteur nous a fourni directement les résultats interpolés.
- L'interpolation spline utilisée est celle implémentée dans Matlab.

Comme les résultats de l'interpolation NOAI qui nous ont été envoyés ont été agrandis d'un facteur deux (deux fois en horizontal et deux fois en vertical), tous les exemples seront illustrés pour ce facteur d'agrandissement afin de permettre la comparaison.

Nous observerons tout au long de cette partie le rendu visuel de toutes ces méthodes pour quelques exemples choisis pour mettre en valeur les défauts ou avantages de chacune des méthodes. En particulier, il sera intéressant de remarquer dans quelles conditions les artefacts d'interpolation classiques (*jaggy*, effets d'escalier) disparaissent. Nous remarquerons que la plupart des méthodes qui parviennent à éliminer ces artefacts en font apparaître de nouveaux (faux contours, faux pixels et destruction de textures) qui donnent un aspect peu naturel aux images interpolées. Nous discuterons alors le meilleur compromis entre l'élimination des artefacts classiques et un rendu visuellement agréable. Pour clarifier la comparaison, nous présenterons à la fin de chaque exemple un tableau récapitulatif des artefacts éliminés et ajoutés pour chaque méthode d'interpolation en évaluant leur comportement (trois étoiles : très bon comportement ; une étoile : mauvais rendu visuel). Ce tableau ne contient pas une liste exhaustive de tous les artefacts habituellement cités pour les techniques d'interpolation, mais regroupe ceux qui sont les plus critiques pour les méthodes d'interpolation directionnelle.

1 Comparaison sur des exemples naturels

Nous allons dans un premier temps nous intéresser à des exemples d'interpolation sur des images naturelles afin de comparer le rendu des méthodes dans un but d'application pratique. Des images en noir et blanc ainsi qu'en couleur seront traitées de façon à vérifier que le passage dans l'espace couleur ne pose pas de problème pour ces méthodes.

1.1 Contours diagonaux, horizontaux et verticaux

La première série d'exemples que nous allons étudier contient des contours rectilignes de direction horizontale, verticale et diagonale. Ces directions sont relativement faciles à traiter puisque les pixels contenant l'information de direction sont situés dans le voisinage connexe du pixel à interpoler. Si la détection de direction est correcte, toutes les méthodes devraient donner des résultats satisfaisants.

L'observation des résultats (Figure 10.4 à la Figure 10.7) permet de voir que toutes les méthodes donnent des résultats assez similaires, ne présentant pas d'artefacts majeurs sur les contours. Cependant, les interpolations AQua, IAD et NOAI se démarquent un peu dans le sens où elles conservent une transition du contour par rapport au fond plus franche que les autres. Le noyau d'interpolation directionnel permet en effet d'interpoler en premier lieu dans la direction du contour, en utilisant les pixels dans la direction du contour. L'interpolation spline par exemple, ne parvient pas à construire une transition aussi nette car le noyau d'interpolation, isotrope, ne prend pas en compte la géométrie du contour comme illustré en Figure 10.1.
Notre interpolation, basée sur le noyau spline, ne permet donc pas non plus d'obtenir une transition aussi nette. Cependant, le correcteur Gaussien appliqué dans le sens du contour permet de réduire les variations parasites le long du contour, et de le rendre plus homogène.
Enfin, les interpolations NEDI et IAD montrent dans certains cas des difficultés à rendre un contour d'aspect correct. Même si les transitions entre le contour et le fond sont nettes, l'image interpolée manque de régularité le long du contour. L'apparition de faux pixels donne au résultat un effet moins naturel que les autres.

Le tableau 10.2 résume le comportement des interpolations sur les exemples présentés dans cette partie. Ce tableau est basé sur une interprétation subjective de notre part, d'après l'étude des exemples de la Figure 10.4 à la Figure 10.7. L'étude des résultats permet de voir que l'utilisation d'une interpolation directionnelle n'est pas mise en évidence pour des contours orientés de la sorte.

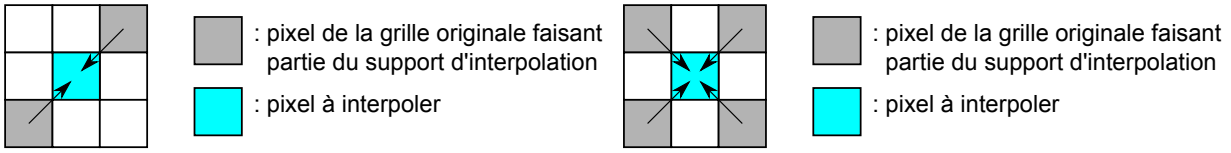
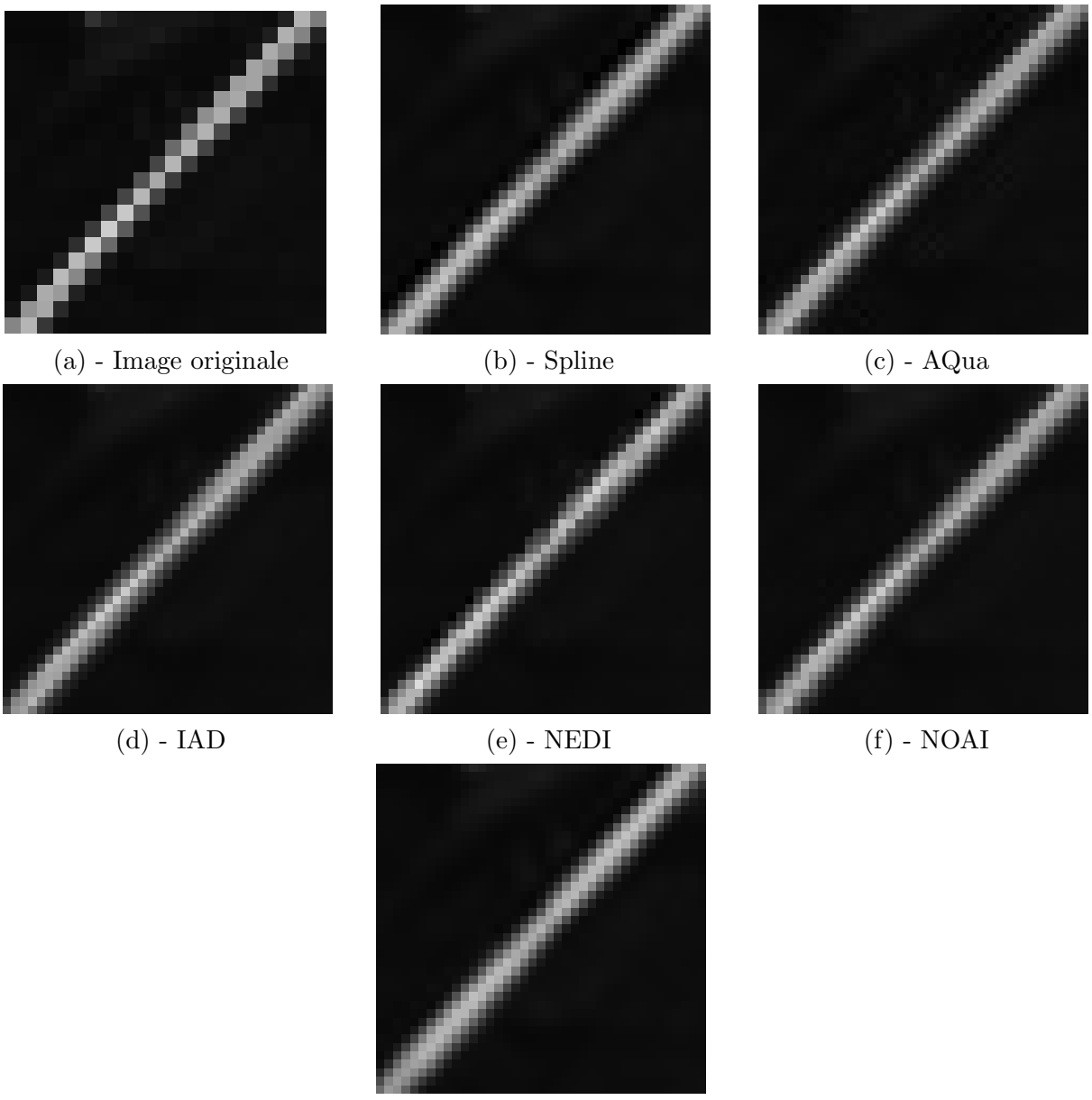


FIGURE 10.1 – Comparaison des supports d'interpolation pour l'interpolation directionnelle (a) et l'interpolation spline (b).

Artefacts / Interpolation	Jaggy	Effets d'escalier	Netteté	Faux pixels	Faux contours
Spline	☆ ☆ ☆	☆ ☆ ☆	☆ ☆	☆ ☆ ☆	☆ ☆ ☆
AQua	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆
IAD	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆	☆ ☆ ☆
NEDI	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆	☆ ☆ ☆
NOAI	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆	☆ ☆ ☆
Méthode proposée	☆ ☆ ☆	☆ ☆ ☆	☆ ☆	☆ ☆ ☆	☆ ☆ ☆

FIGURE 10.2 – Tableau récapitulatif des résultats obtenus pour les contours de direction 0°, 45°, 90° et 135°.

FIGURE 10.3 – Illustration de la région sélectionnée, contenant le contour à 45°.



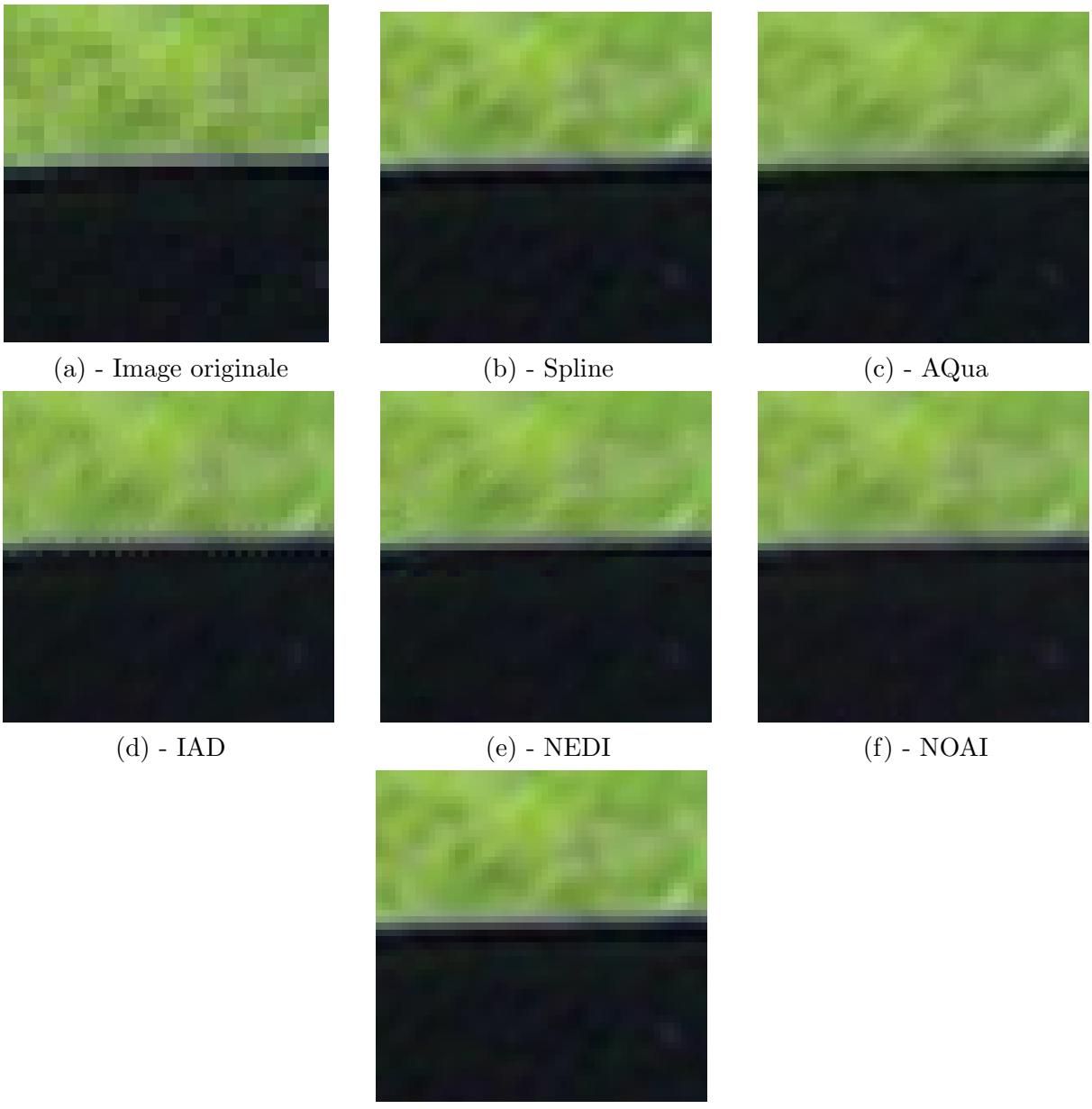
(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.4 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour orienté à 45°.

FIGURE 10.5 – Illustration des régions sélectionnées, contenant des contour horizontaux et verticaux.

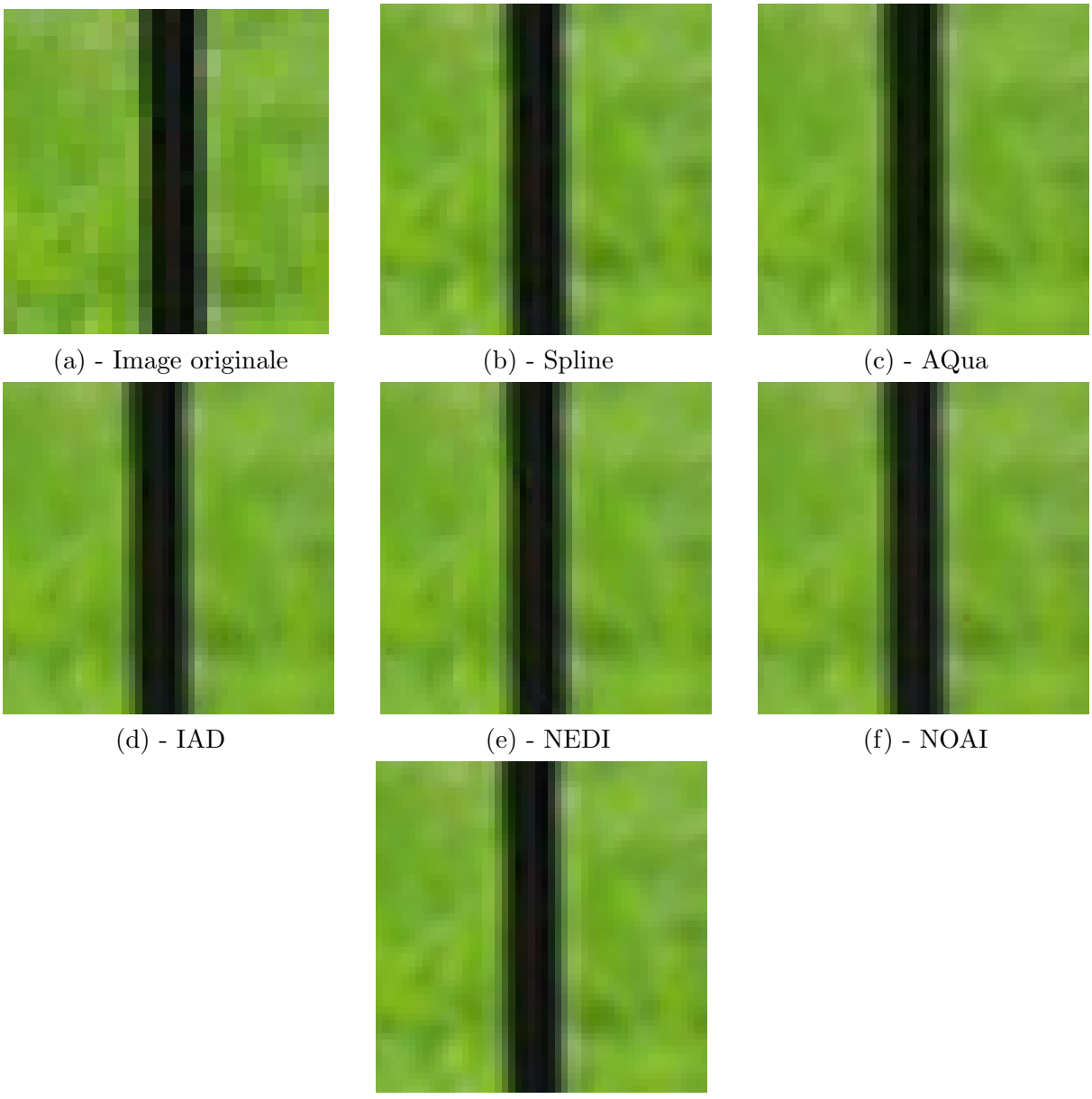


(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.6 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour horizontal.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.7 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour vertical.

1.2 Contour rectiligne de direction arbitraire

L'étude d'un contour orienté de façon arbitraire permet de voir que les comportements des méthodes d'interpolation diffèrent de la situation précédente. En effet, nous allons mettre en évidence dans cette partie le fait que les méthodes qui n'utilisent pas un noyau d'interpolation suffisamment grand ne parviennent pas à interpoler de façon satisfaisante les contours de direction arbitraire. En effet, les artefacts de *jaggy* et d'effets d'escalier apparaissent lorsque le voisinage d'interpolation ne contient pas les pixels alignés dans la direction du contour. Nous étudierons le cas de plusieurs orientations de contours. Nous verrons que plus le contour se rapproche de la verticale ou de l'horizontale, moins les méthodes d'interpolation donnent des résultats satisfaisants.

1.2.1 Contours de direction arbitraire

Le premier exemple tiré 10.9 de et illustré en Figure 10.10 est un contour orienté à environ 75°. L'étude de cet exemple montre que les méthodes IAD et AQua reconstruisent mal le contour, et ont un comportement similaire à l'interpolation spline. Ce comportement s'explique majoritairement par la petite taille du voisinage utilisé pour l'interpolation. De plus, la méthode de détection de direction utilisée dans la méthode AQua ne permet pas de détecter des directions autres que 0°, 45°, 90° ou 135°, ce qui justifie également le mauvais comportement de cette méthode sur des contours de directions différentes des quatre directions citées.
En revanche, les méthodes NEDI, NOAI ainsi que notre approche parviennent à représenter correctement le contour sur la grille haute résolution en utilisant des voisinages plus grands, permettant ainsi d'interpoler les nouveaux pixels grâce aux pixels originaux situés dans la direction du contour. L'interpolation NEDI en particulier se comporte très bien puisque le contour est à la fois très net et sans artefacts. Notre méthode donne un contour un peu moins net que l'interpolation NEDI mais également sans artefacts. Le résultat obtenu par la méthode NOAI est globalement satisfaisant même si quelques pixels situés à droite du contour apparaissent peu naturels.

1.2.2 Contours proches de l'horizontale ou de la verticale

L'étude de contours proches de l'horizontale (ou de la verticale) permet d'illustrer de manière encore plus claire les différences de comportements entre les méthodes. En effet, les résultats de plusieurs exemples sont illustrés en Figure 10.12, 10.14 et 10.16. Ils montrent que seule notre interpolation parvient à conserver l'aspect rectiligne du contour sans introduire d'autres artefacts. Le voisinage de l'interpolation NOAI, limité à (5×5) pixels n'est plus suffisant. L'interpolation NEDI, si elle parvient à éliminer les artefacts de *jaggy* et les effets d'escalier, présente un nombre important d'irrégularités le long des contours interpolés. Nous appelons *faux pixels* ces irrégularités, à cause du fait qu'ils dénotent avec leur voisinage connexe. En général, ils sont visibles quand de mauvais pixels sont utilisés pour l'interpolation. Ceci se produit quand :

- les objets à interpoler sont relativement petits.
- les contours changent souvent de directions.
- une mauvaise estimation de direction du contour est réalisée.

En conclusion, les résultats observés sur les résultats présentés dans cette partie sont résumées dans le tableau 10.8. Lorsque la direction de contour diffère de 0°, 45°, 90° ou 135°, l'interpolation NEDI et la nôtre semblent pouvoir éliminer les artefacts de *jaggy* et les effets d'escalier, mais seule la nôtre permet de le faire sans introduire d'artefact supplémentaire.

Artefacts / Interpolation	Jaggy	Effets d'escalier	Netteté	Faux pixels	Faux contours
Spline	☆	☆	☆☆	☆☆☆	☆☆☆
AQua	☆	☆	☆☆	☆☆☆	☆☆☆
IAD	☆	☆	☆	☆☆	☆☆☆
NEDI	☆☆☆	☆☆☆	☆☆☆	☆☆	☆☆☆
NOAI	☆☆	☆☆	☆☆☆	☆☆	☆☆☆
Méthode proposée	☆☆☆	☆☆☆	☆☆	☆☆☆	☆☆☆

FIGURE 10.8 – Tableau récapitulatif des résultats obtenus pour les contours de direction arbitraire.

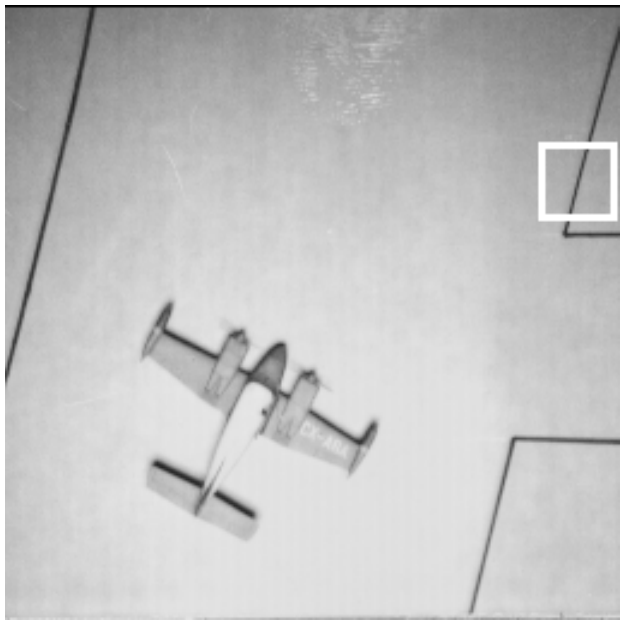
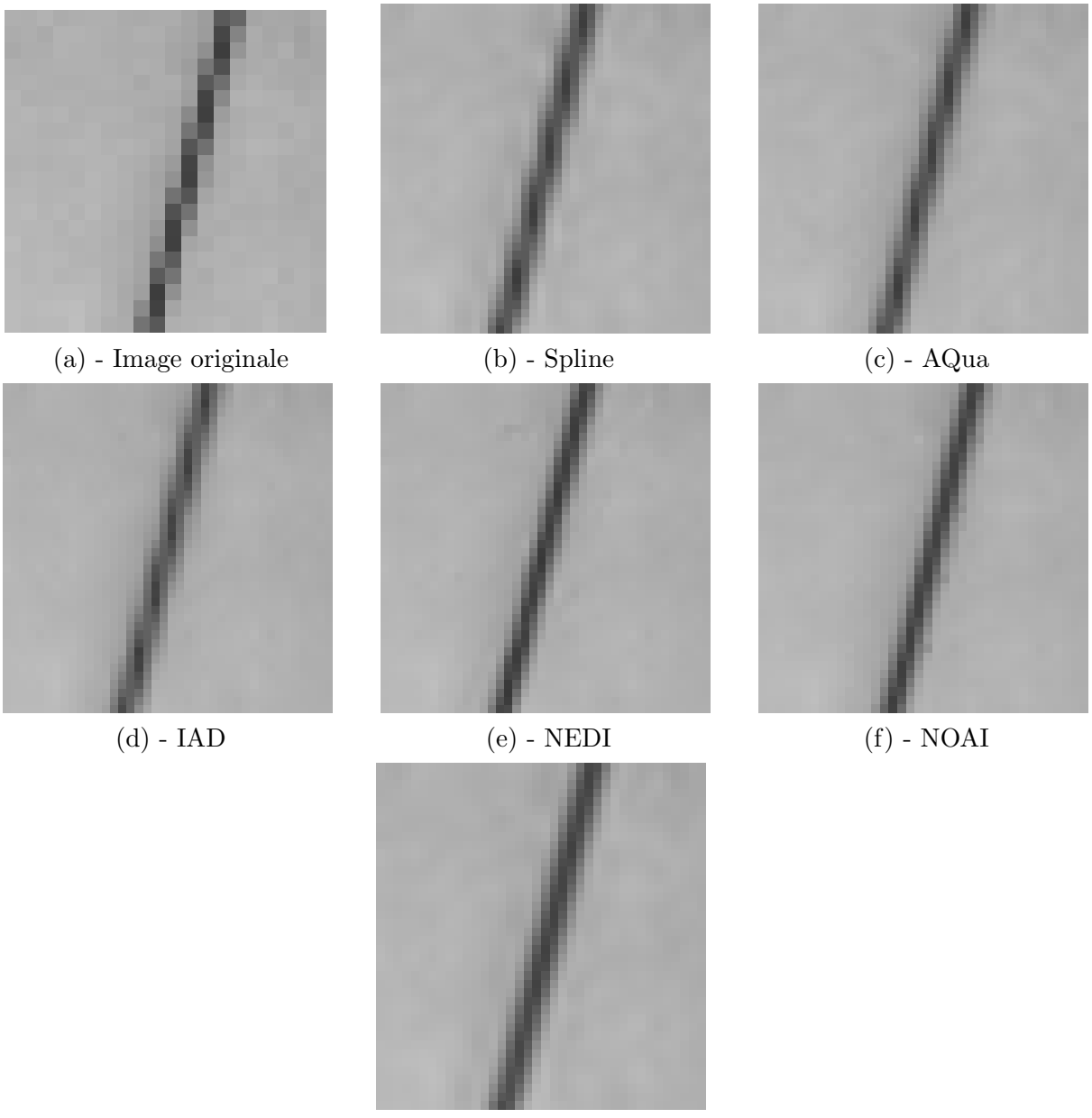


FIGURE 10.9 – Illustration de la région sélectionnée, contenant le contour de direction arbitraire.



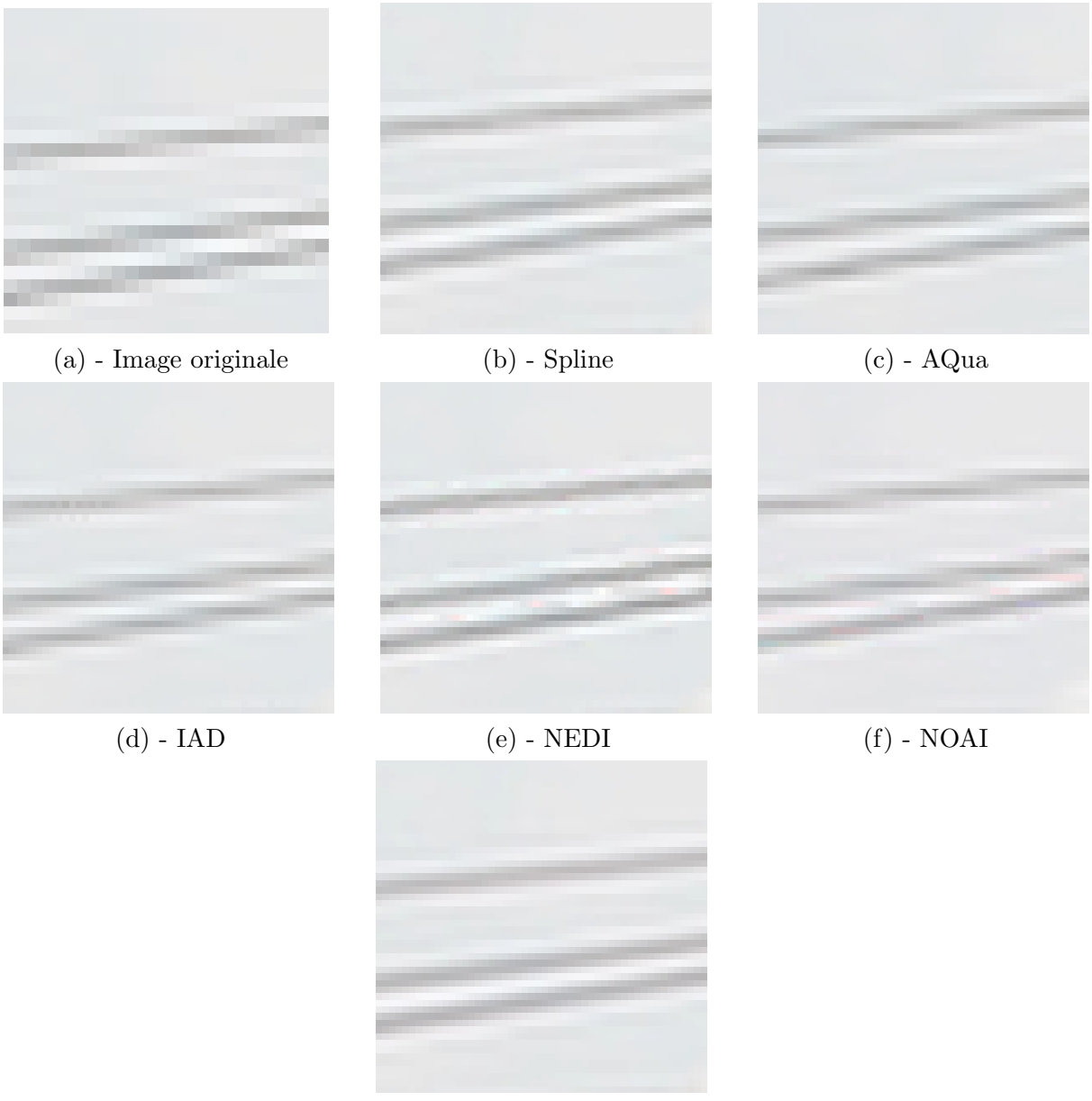
(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.10 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour rectiligne de direction arbitraire.

FIGURE 10.11 – Illustration de la région sélectionnée, contenant le contour de direction proche de l'horizontale.



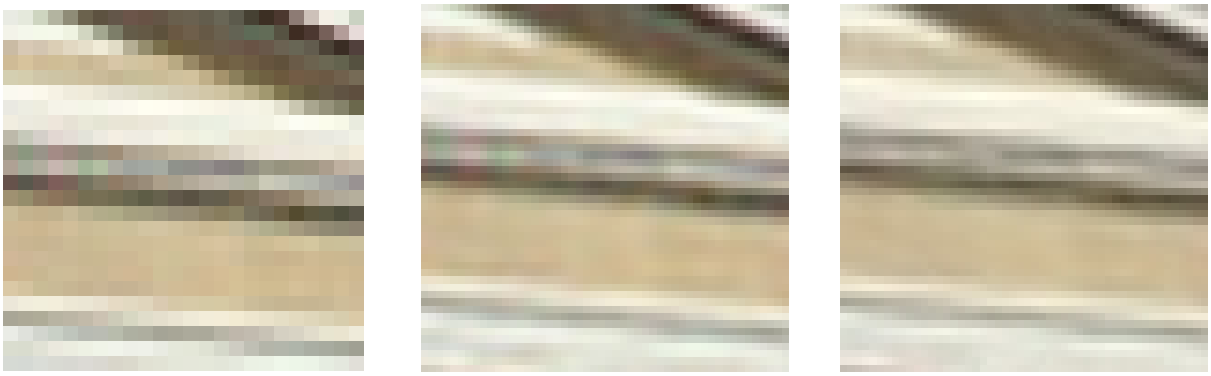
(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

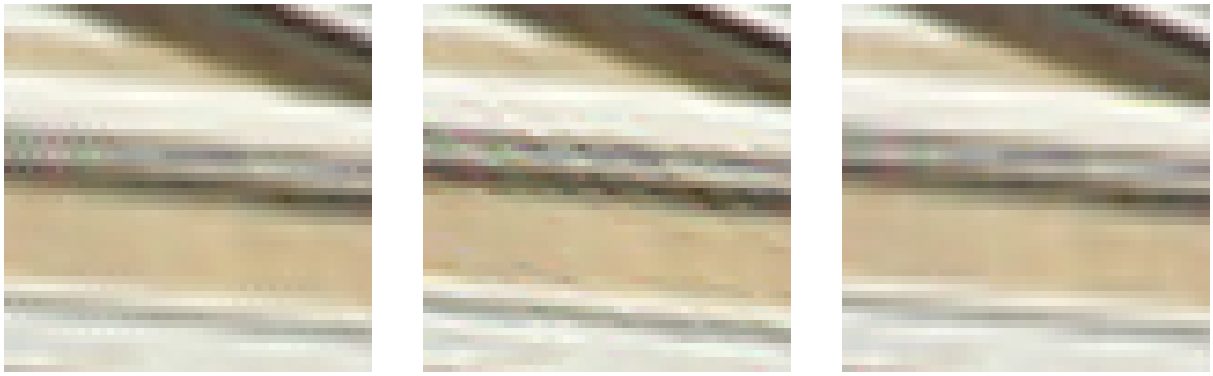
(g) - Méthode proposée

FIGURE 10.12 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour rectiligne proche de l'horizontale.

FIGURE 10.13 – Illustration de la région sélectionnée, contenant le contour de direction proche de l'horizontale.



(a) - Image originale (b) - Spline (c) - AQua

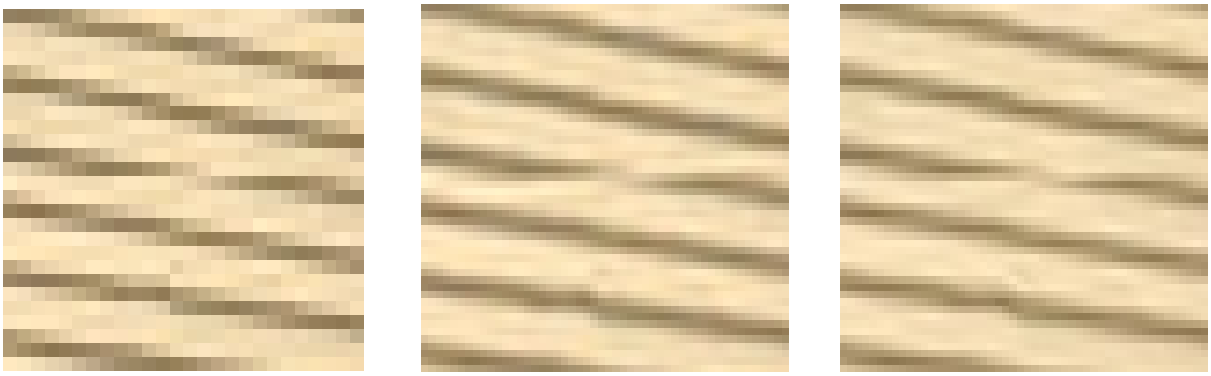


(d) - IAD (e) - NEDI (f) - NOAI

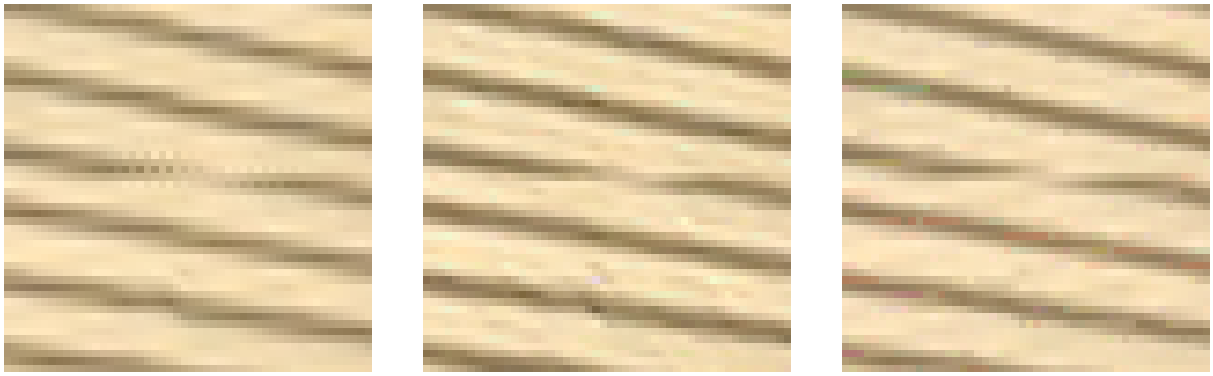
(g) - Méthode proposée

FIGURE 10.14 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour rectiligne proche de l'horizontale.

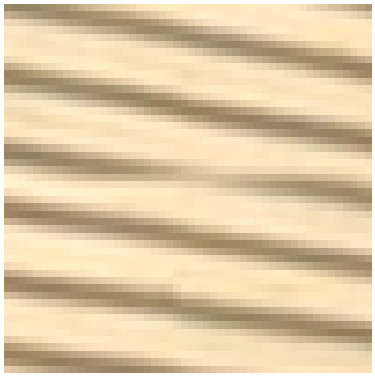
FIGURE 10.15 – Illustration de la région sélectionnée, contenant le contour de direction proche de l'horizontale.



(a) - Image originale (b) - Spline (c) - AQua



(d) - IAD (e) - NEDI (f) - NOAI



(g) - Méthode proposée

FIGURE 10.16 – Résultats obtenus par les différentes méthodes d'interpolation pour un contour rectiligne proche de l'horizontale.

1.3 Texte

La série d'exemple suivant contient des zones de texte. Ce cas est intéressant à étudier puisqu'il permet de mettre en évidence le cas ou les changements d'orientation de contours sont fréquents. En effet, la petite taille des lettres ainsi que les changements fréquents de directions des contours mettent généralement en difficulté les interpolations directionnelles. De plus, le fort contraste entre les lettres et le fond ainsi que la courbure de certaines lettres devraient favoriser l'apparition de faux pixels très visibles.

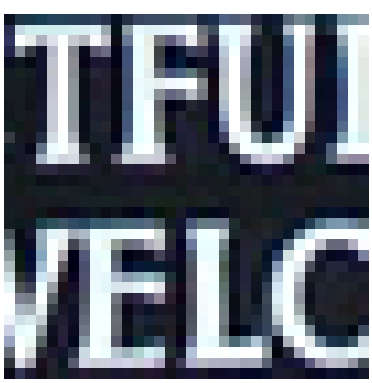
L'étude des résultats (de la Figure 10.19 à la Figure 10.23) montre en effet que des faux pixels apparaissent dans certaines images interpolées. Ils sont particulièrement visibles dans les résultats des interpolations AQua, IAD et NEDI. Dans ce cas, ils apparaissent principalement dans les régions où les changements d'orientation sont fréquents, et où l'estimation de direction n'a pas détecté le changement de direction.
Les interpolations AQua et NEDI engendrent également des contours peu naturels (particulièrement dans l'exemple 10.23), que nous appelons *faux contours*. Ces derniers sont créés dans des régions où l'information de direction n'est pas pertinente, et donc pour lesquelles une interpolation directionnelle n'est pas nécessaire (zones sans contours, ou contenant des détails). Les artefacts engendrés par ces méthodes mettent bien en évidence le fait qu'utiliser une interpolation directionnelle sur toute l'image entraîne l'apparition d'artefacts supplémentaires.

Les autres résultats (NOAI et notre interpolation) permettent d'obtenir un rendu homogène et sans artefact flagrant par rapport aux autres méthodes d'interpolation directionnelles, en se rapprochant de l'aspect de l'interpolation spline. Le meilleur comportement de ces méthodes est dû à la bonne localisation des contours à interpoler de façon directionnelle. En effet, la détection d'isophotes rectilignes pour la méthode NOAI et la création de masques de Gabor pour notre méthode prouvent leur efficacité dans ce genre de situations. Elles permettent d'éviter l'apparition de faux pixels et de faux contours, en appliquant l'interpolation (ou la correction) directionnelle uniquement dans les régions où l'information de direction est pertinente. Ces deux méthodes n'améliorent pas l'interpolation spline, mais limitent l'apparition d'artefacts supplémentaires, comme illustré dans le tableau 10.17.

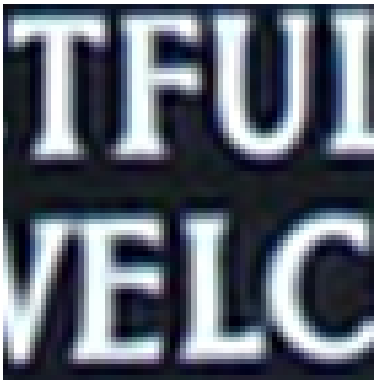
Artefacts / Interpolation	Jaggy	Effets d'escalier	Netteté	Faux pixels	Faux contours
Spline	☆☆	☆☆	☆☆	☆☆☆	☆☆☆
AQua	☆☆☆	☆☆☆	☆☆☆	☆	☆
IAD	☆☆	☆☆	☆☆	☆	☆☆☆
NEDI	☆☆☆	☆☆☆	☆☆☆	☆	☆
NOAI	☆☆	☆☆	☆☆	☆☆☆	☆☆☆
Méthode proposée	☆☆☆	☆☆☆	☆☆	☆☆☆	☆☆☆

FIGURE 10.17 – Tableau récapitulatif des résultats obtenus pour les contours de direction arbitraire.

FIGURE 10.18 – Illustration de la région sélectionnée, contenant le texte blanc sur noir à étudier.



(a) - Image originale

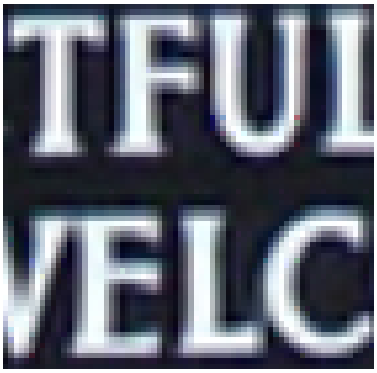


(b) - Spline

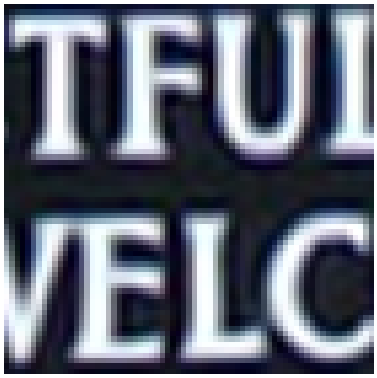
(c) - AQua

(d) - IAD

(e) - NEDI



(f) - NOAI



(g) - Méthode proposée

FIGURE 10.19 – Résultats obtenus par les différentes méthodes d'interpolation sur un texte blanc sur noir.

FIGURE 10.20 – Illustration de la région sélectionnée, contenant le texte à étudier.

(a) - Image originale

(b) - Spline

(c) - AQua

(d) - IAD

(e) - NEDI

(f) - NOAI

(g) - Méthode proposée

FIGURE 10.21 – Résultats obtenus par les différentes méthodes d'interpolation pour un texte de petite taille.

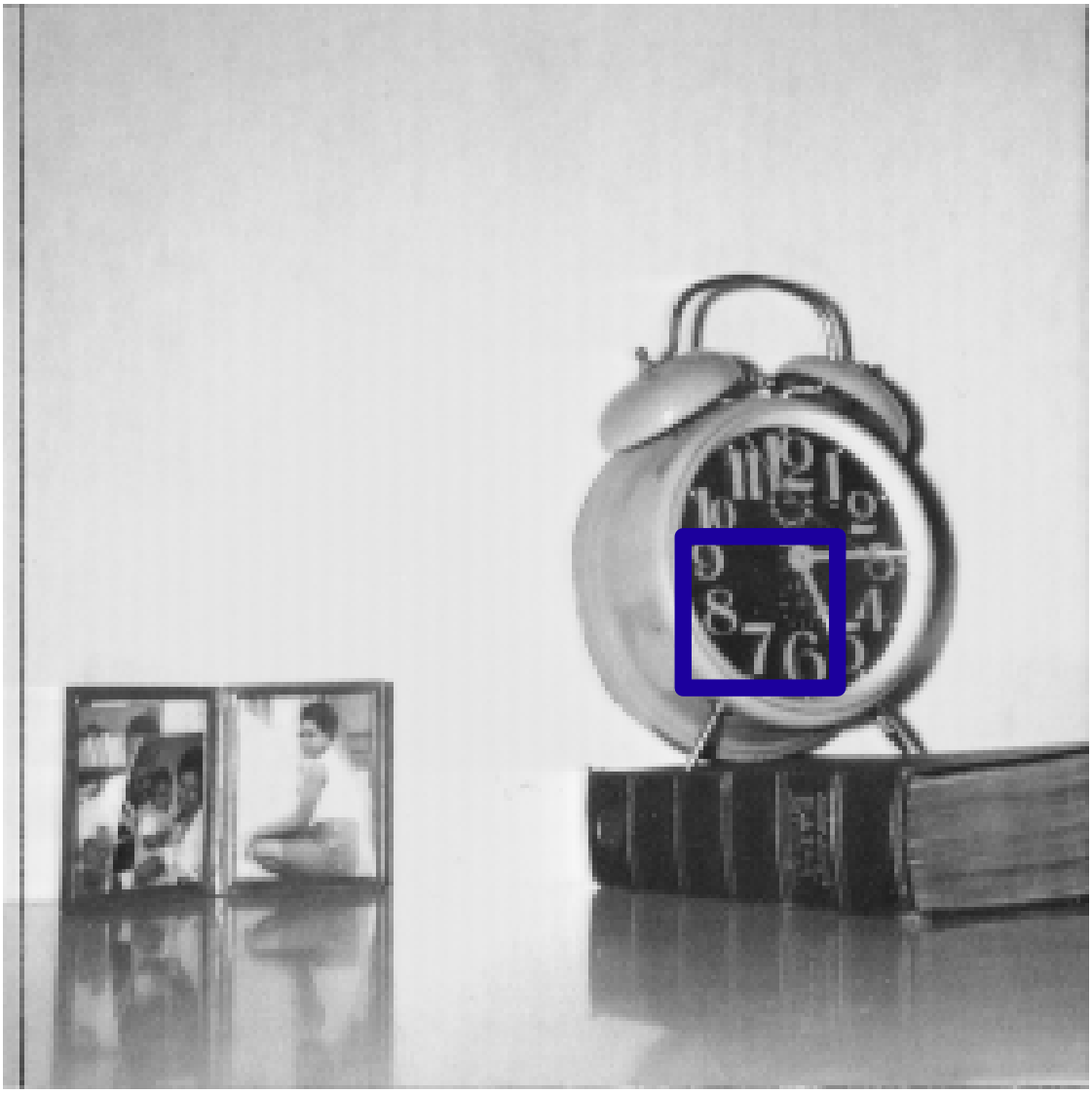
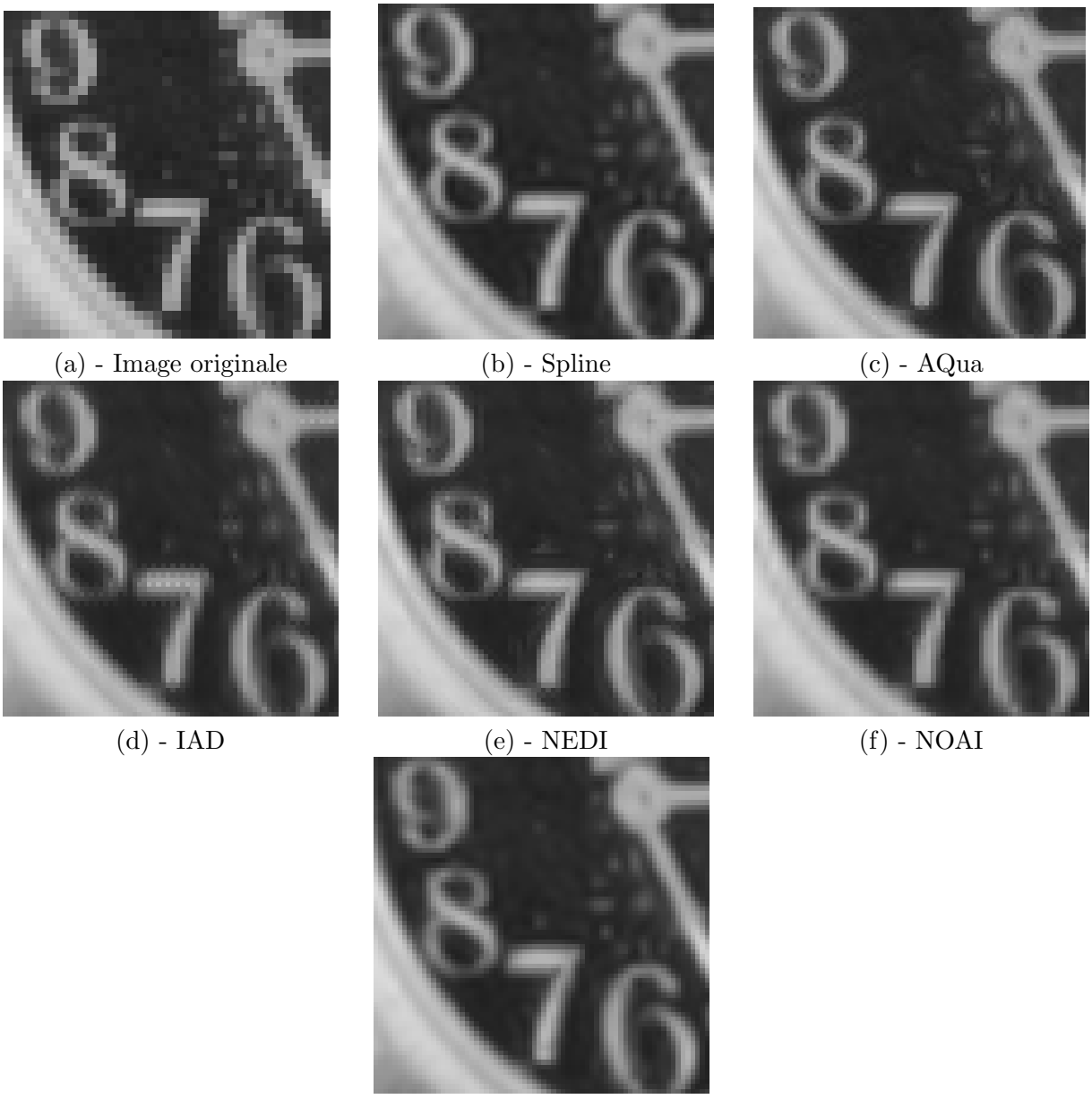


FIGURE 10.22 – Illustration de la région sélectionnée, contenant un texte composé de chiffres.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.23 – Résultats obtenus par les différentes méthodes d'interpolation sur des chiffres.

1.4 Textures

La série d'exemples suivante s'intéresse à un autre cas délicat pour les méthodes d'interpolation directionnelles, à savoir les textures. L'intérêt de cette étude réside dans le fait que la direction des contours qui forment la texture peut être déterminante (textures orientées) ou alors très peu intéressante dans le cas de structures plus aléatoires ou contenant beaucoup de détails. Il est alors intéressant d'étudier comment réagissent les différentes méthodes, qui devraient dans l'absolu fortement améliorer le rendu visuel des textures directionnelles et ne pas dégrader les détails contenus dans les textures aléatoires. Les deux types de textures sont analysées tour à tour dans cette partie.

1.4.1 Textures directionnelles

Un exemple de texture directionnelle contenant des rayures est illustré en Figure 10.26. Il montre que seule notre méthode parvient à interpoler correctement les rayures de la région, sans créer de *jaggy* ni de faux pixels. Les contours sont en effet correctement reconstruits et la texture conserve son aspect naturel. L'aliasing présent dans l'image originale est même éliminé. En revanche, les interpolations IAD, AQua et NEDI produisent des contours contenant des effets d'escalier, de *jaggy* et conserve l'aliasing présent dans l'image originale. L'interpolation NEDI crée même des structures parasites le long des rayures, altérant ainsi l'aspect naturel de la texture.
L'interpolation NOAI parvient à correctement capter la direction des rayures ainsi qu'à les reconstruire de façon satisfaisante, tant que celles-ci sont bien marquées. Cependant, cette technique semble à la fois perturbée par l'aliasing présent dans l'image originale et par la diminution de l'amplitude des rayures puisque celles-ci ne sont pas toutes considérées comme étant des contours rectilignes. L'interpolation réalisée utilise alors un noyau isotrope qui entraîne l'apparition des artefacts classiques d'interpolation. Visuellement, lorsque les deux types d'interpolation se côtoient, les transitions entre les pixels interpolés de façon directionnelle et les autres sont perturbantes (pixels appartenant à la tâche sombre par exemple). Ce sentiment est d'autant plus fort que l'oeil perçoit la continuité des rayures, et est gêné par le caractère non-homogène des rayures créées par cette interpolation.

1.4.2 Textures aléatoires

La deuxième série d'exemples contient des textures pour lesquelles l'information de direction est difficile voire impossible à utiliser. Les exemples choisis contiennent différents types de textures :

- textures aléatoires en Figure 10.31 et 10.28
- textures contenant des détails fins en Figure 10.29 et 10.32.
- texture de type maillage en Figure 10.34.

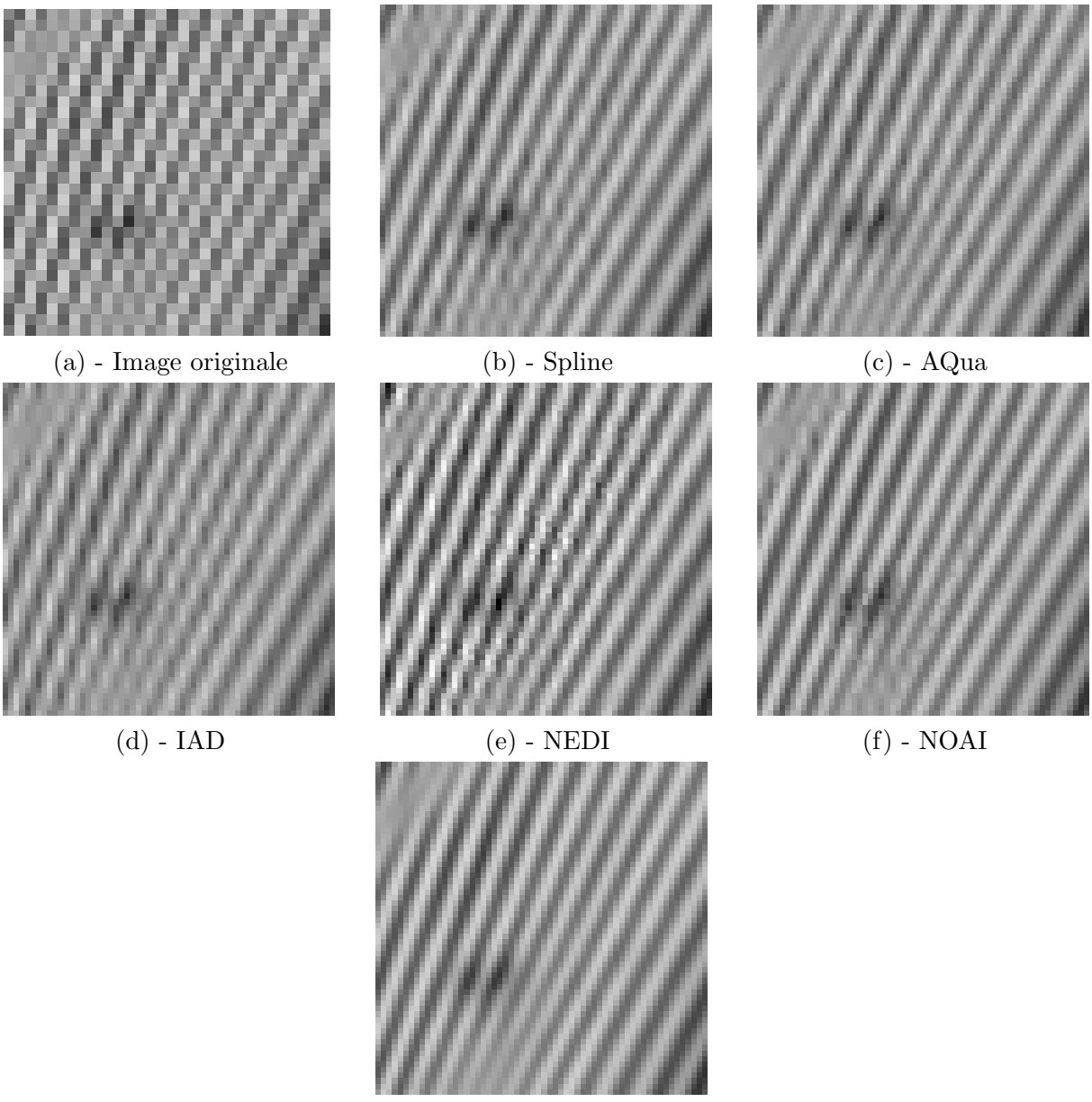
Les principaux défauts visibles sur les résultats sont l'apparition de faux pixels et faux contours. Les faux pixels apparaissent pour les mêmes raisons que précédemment, c'est-à-dire lorsque des mauvais pixels sont utilisés pour l'interpolation. Les faux contours sont créés lorsque l'interpolation introduit de la cohérence qui n'existait pas dans l'image originale, entre des pixels de l'image haute-résolution. Cela se produit lorsque deux pixels proches de forts gradients sont considérés comme appartenant au même contour. Ce cas est fréquent pour les textures que nous étudions, et met en défaut la plupart des interpolations directionnelles. Le but de cette étude est donc de montrer quelles méthodes introduisent le moins de dégradation.
L'étude des différents exemples montrent que des faux pixels sont systématiquement présents dans les résultats des méthodes AQua, IAD et NEDI, et que des faux contours apparaissent pour les méthodes AQua et NEDI. De la même manière que pour les exemples sur les zones de texte, la méthode NOAI et la nôtre semblent les plus robustes aux textures dites aléatoires ou contenant une forte concentration de détails. Elles évitent d'utiliser à mauvais escient l'interpolation directionnelle et permettent à l'image de conserver un aspect homogène. Cependant, les résultats de

la méthode NOAI pour les Figures 10.32 et 10.34 montrent que certains contours présentent des effets d'escalier. La méthode de localisation des contours à interpoler de façon directionnelle n'est pas assez précise pour corriger certains contours dans un contexte tel que celui-ci. Cet exemple met bien en avant la difficulté de définir une méthode de localisation des contours à interpoler de façon directionnelle, puisque cette dernière doit à la fois être sensible aux contours rectilignes même de faible amplitude ou contenant de l'aliasing, mais sans prendre en compte les zones de textures contenant de forts gradients. Notre méthode semble être un meilleur compromis, même si quelques faux contours apparaissent sur quelques textures aléatoires. Le tableau 10.24 reprend les éléments de la comparaison effectuée.

Artefacts / Interpolation	Jaggy	Effets d'escalier	Netteté	Faux pixels	Faux contours
Spline	☆☆	☆☆	☆☆	☆☆☆	☆☆☆
AQua	☆☆☆	☆☆☆	☆☆☆	☆	☆
IAD	☆	☆	☆☆	☆	☆☆☆
NEDI	☆☆☆	☆☆☆	☆☆☆	☆	☆
NOAI	☆☆	☆☆	☆☆	☆☆	☆☆☆
Méthode proposée	☆☆☆	☆☆☆	☆☆	☆☆☆	☆☆

FIGURE 10.24 – Tableau récapitulatif des résultats obtenus sur des textures.

FIGURE 10.25 – Illustration de la région sélectionnée, contenant une texture directionnelle.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.26 – Résultats obtenus par les différentes méthodes d'interpolation sur une texture directionnelle.

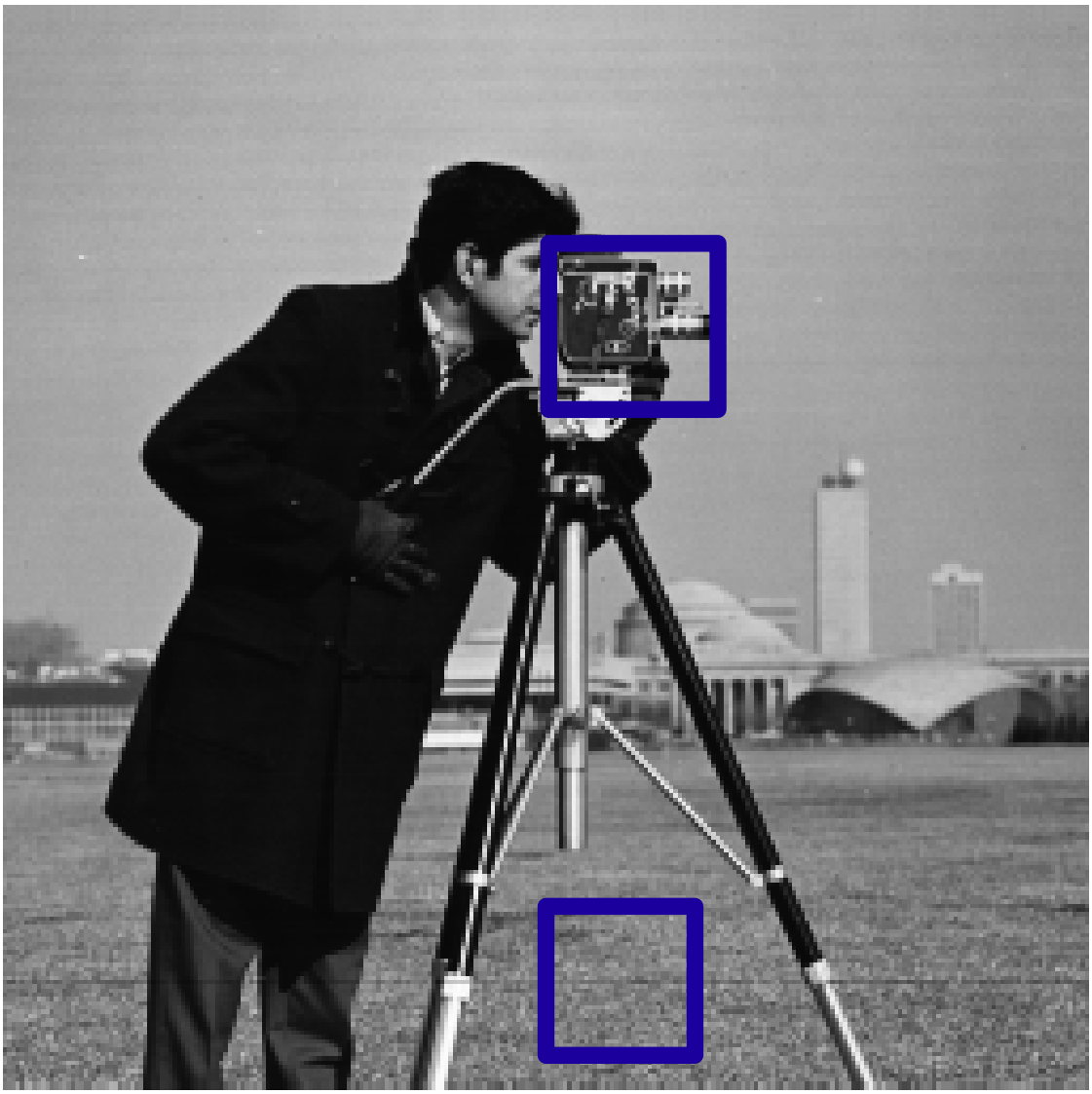
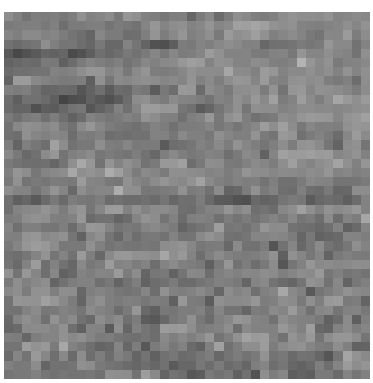
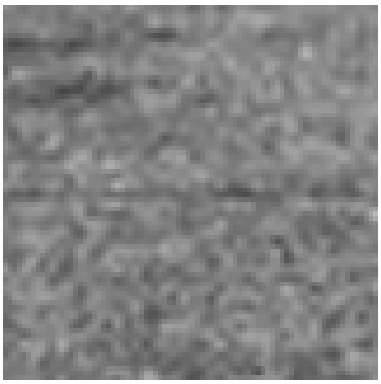


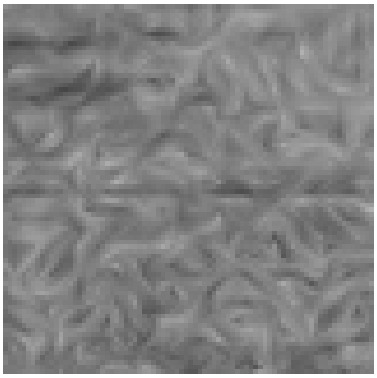
FIGURE 10.27 – Illustration des deux régions sélectionnées, contenant une texture aléatoire et une zone de détails.



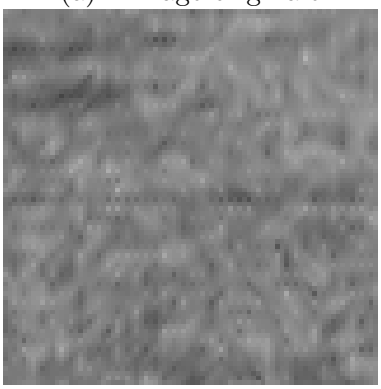
(a) - Image originale



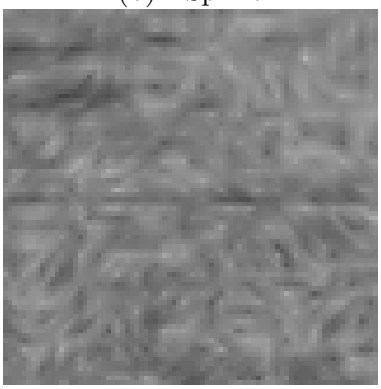
(b) - Spline



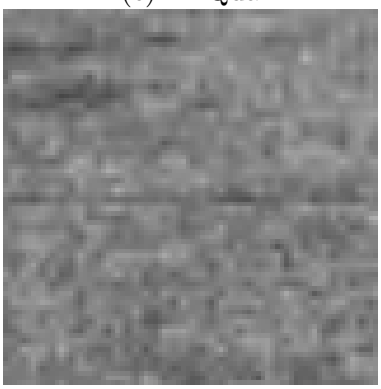
(c) - AQua



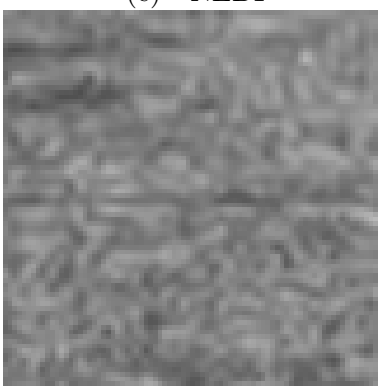
(d) - IAD



(e) - NEDI

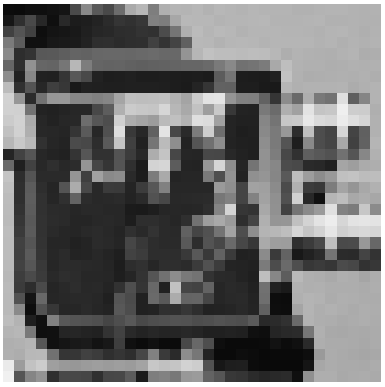


(f) - NOAI

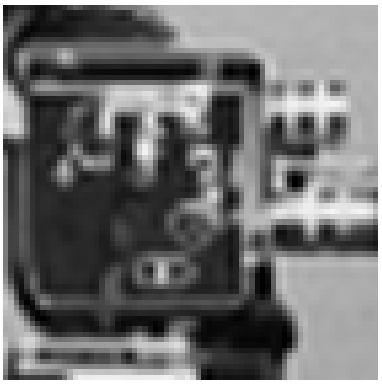


(g) - Méthode proposée

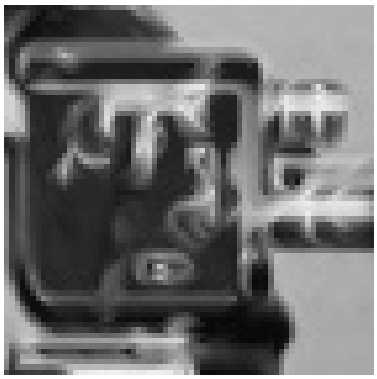
FIGURE 10.28 – Résultats obtenus par les différentes méthodes d'interpolation pour une texture aléatoire.



(a) - Image originale

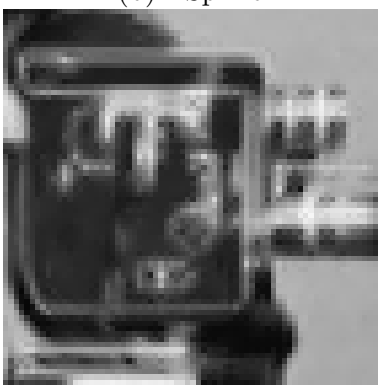


(b) - Spline



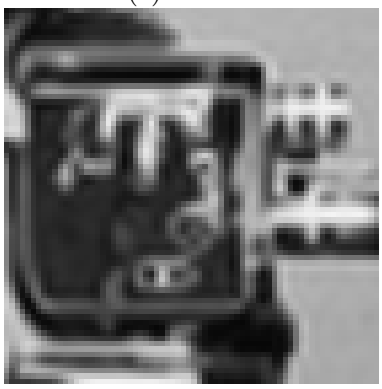
(c) - AQua

(d) - IAD



(e) - NEDI

(f) - NOAI



(g) - Méthode proposée

FIGURE 10.29 – Résultats obtenus par les différentes méthodes d'interpolation sur la zone de détails.

FIGURE 10.30 – Illustration des deux régions sélectionnées, contenant une texture aléatoire et une zone de détails.

(a) - Image originale

(b) - Spline

(c) - AQua

(d) - IAD

(e) - NEDI

(f) - NOAI

(g) - Méthode proposée

FIGURE 10.31 – Résultats obtenus par les différentes méthodes d'interpolation sur une texture aléatoire.

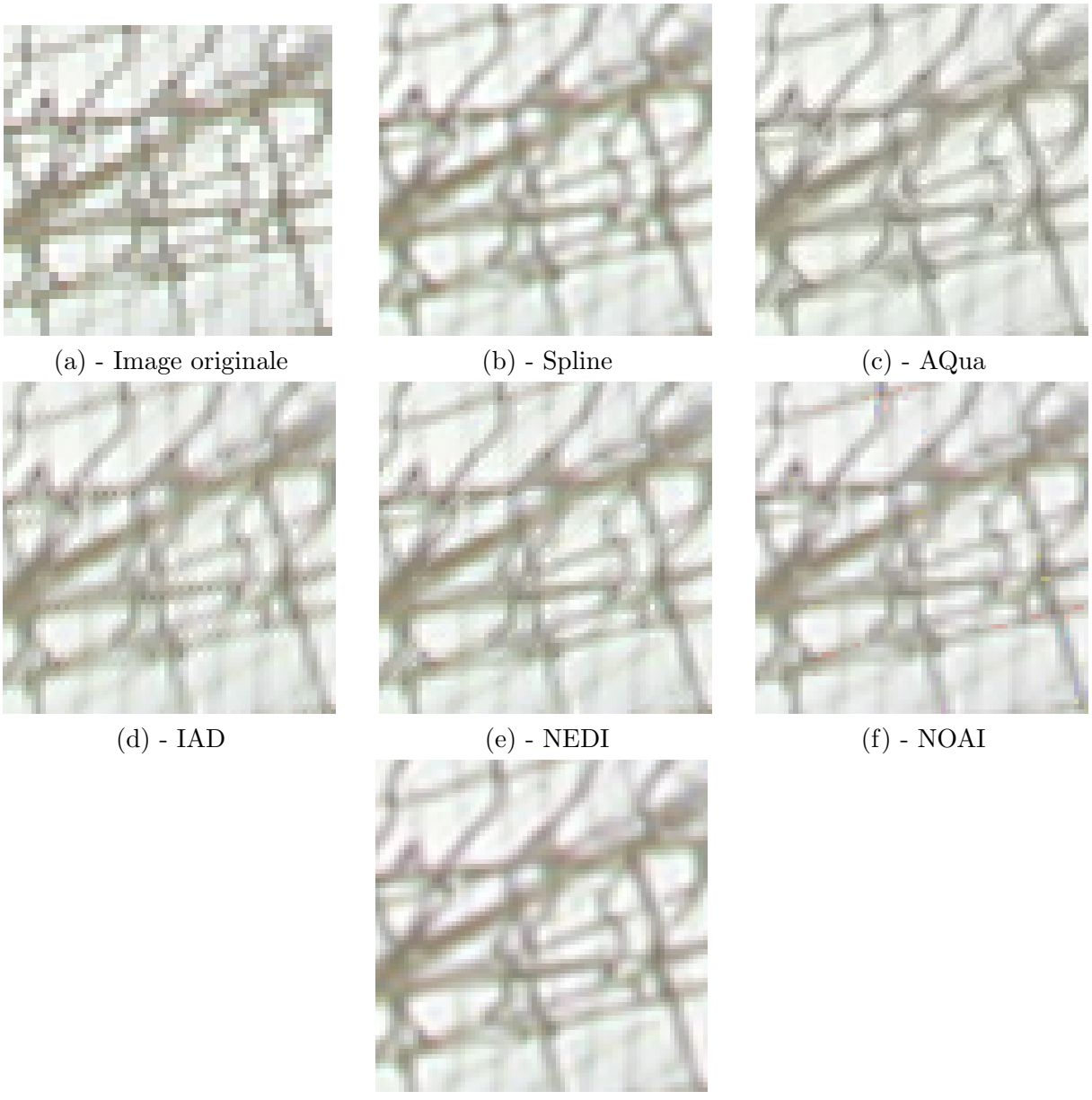
(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.32 – Résultats obtenus par les différentes méthodes d'interpolation pour une zone de détails.

FIGURE 10.33 – Illustration de la région sélectionnée, contenant une texture de type maillage.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.34 – Résultats obtenus par les différentes méthodes d'interpolation sur une texture de type maillage.

1.5 Conclusion sur les exemples naturels

L'étude des exemples naturels de la partie précédente nous a permis de comparer les différentes méthodes d'interpolation directionnelles, et d'observer leur capacité à éliminer les artefacts classiques d'interpolation (jaggy, effet d'escalier, flou) sans en ajouter de nouveaux (faux pixels, faux contours). Un tableau récapitulatif, basé sur la conclusion de chaque exemple est présenté en Figure 10.35.

Artefacts / Interpolation	Réduction des anciens artefacts	Ajout de nouveaux artefacts	Aspect naturel
AQua	☆ ☆	☆	☆
IAD	☆	☆	☆
NEDI	☆ ☆	☆	☆
NOAI	☆ ☆	☆ ☆	☆ ☆ ☆
Méthode proposée	☆ ☆ ☆	☆ ☆	☆ ☆ ☆

FIGURE 10.35 – Tableau récapitulatif des résultats obtenus sur l'ensemble des exemples.

Ce tableau, bien que non exhaustif, met en valeur la supériorité de notre méthode au niveau de la suppression des anciens artefacts. En effet, nos résultats contiennent très peu de *jaggy* et d'effet d'escalier même pour des contours orientés dans des directions proches de la verticale ou de l'horizontale que les autres interpolations ne peuvent pas reconstruire. En effet, notre interpolation est celle qui utilise le plus grand voisinage d'interpolation et permet donc de reconstruire plus de directions. Elle se base également sur une détection de direction précise, qui permet de détecter les directions de contours avec une résolution angulaire optimale.

Si notre méthode corrige efficacement les anciens artefacts, elle limite également l'apparition de nouvelles dégradations bien que les transitions créées par notre méthode soient en général moins franches. Les autres méthodes directionnelles ont un noyau d'interpolation qui permet de conserver la transition originale lorsque la direction est correctement détectée. A l'inverse, notre méthode se base sur un noyau d'interpolation spline isotrope qui a tendance a flouter les transitions, et effectue la correction directionnelle après l'interpolation. C'est toutefois cet enchaînement qui nous permet d'utiliser de grands voisinages pour la correction Gaussienne sans que la cohérence des pixels dans leur voisinage connexe ne soit altérée. La netteté supérieure des autres méthodes est obtenue au prix de l'introduction d'artefacts supplémentaires que sont les faux pixels et les faux contours.

Ce tableau illustre enfin le fait que l'interpolation NOAI et la nôtre permettent d'obtenir les résultats visuellement les plus cohérents, du fait de la bonne localisation des zones à ne pas interpoler de façon directionnelle. L'utilisation d'un noyau d'interpolation isotrope pour gérer ces zones permet de limiter le nombre d'artefacts supplémentaires introduits qui nuisent à l'aspect naturel de l'image. Dans la perspective d'une éventuelle implémentation dans des circuits industriels, les faux contours et faux pixels régulièrement introduits par certaines méthodes sont rédhibitoires. Il est en effet important de noter que les industriels favorisent les algorithmes performants pour une grande variété de contenu plutôt que les algorithmes efficaces seulement dans quelques situations. A titre d'exemple, dix images interpolées avec notre méthode sont illustrées en Annexe B, afin d'observer le comportement stable de notre algorithme sur des images entières de contenu varié.

2 Comparaison sur des images de synthèse

L'évaluation des méthodes grâce à des images de synthèse permet de trouver et de quantifier les limites des différentes méthodes, en contrôlant le contenu des images de test. Nous nous intéresserons principalement à deux situations :

- Nous essaierons de détecter l'écart d'angle minimal entre les directions horizontale ou verticale à partir duquel les méthodes parviennent à reconstruire un contour sans effet d'escalier. Nous avons vu précédemment que plus un contour était proche de la verticale ou de l'horizontale, plus les pixels qui doivent être utilisés pour l'interpolation directionnelle sont éloignés du pixel à interpoler.
- Nous étudierons enfin le comportement des interpolations lorsque de l'aliasing est introduit dans l'image à interpoler. Nous verrons à quel moment la détection de direction est induite en erreur, et les conséquences sur l'image interpolée.

2.1 Etude de l'écart d'angle minimal

2.1.1 Image de test

Dans un premier temps, nous allons étudier l'écart d'angle minimal avec la direction verticale à partir duquel les contours sont corrigés afin de confirmer la conclusion faite sur les images naturelles étudiées dans la partie précédente. Pour cela, l'image de test choisie contient des contours rectilignes dont la pente s'éloigne progressivement de la verticale, créant une alternance de bandes noires et blanches. Cette image est présentée en Figure 10.37. Notons que les contours de gauche et de droite des bandes créées n'ont pas exactement la même orientation. L'interpolation de cette exemple va permettre de discerner l'angle à partir duquel le contour ne présente plus d'effets d'escalier, c'est-à-dire l'angle à partir duquel les méthodes d'interpolation traitent les contours de manière efficace. Les angles des bandes de l'image ont été mesurées avec l'outil de mesure d'angle du logiciel GIMP, afin de quantifier précisément les résultats obtenus. Les directions suivantes correspondent aux orientations successives des contours situés à droite de la bande blanche centrale de direction verticale.

- Le premier contour est orienté à $89,38°$ (côté gauche de la première bande noire)
- Le deuxième contour est orienté à $88,09°$ (côté droit de la première bande noire)
- Le troisième contour est orienté à $86,99°$ (côté gauche de la deuxième bande noire)
- Le quatrième contour est orienté à $85,60°$ (côté droit de la deuxième bande noire)
- Le cinquième contour est orienté à $84,29°$ (côté gauche de la troisième bande noire)
- Le sixième contour est orienté à $82,87°$ (côté droit de la troisième bande noire)
- Le septième contour est orienté à $81,87°$ (côté gauche de la quatrième bande noire)
- Le huitième contour est orienté à $80,54°$ (côté droit de la troisième bande noire)

D'après l'étude précédente, nous savons déjà que les méthodes d'interpolation utilisant uniquement les pixels du voisinage proche (IAD et AQua) devraient faillir lors de l'interpolation de contours dont la direction est proche de la verticale. Par contre, il est intéressant de mesurer et de comparer les performances des méthodes NEDI, NOAI avec la nôtre puisque ces méthodes s'autorisent des voisinages d'interpolation plus grands, permettant ainsi d'augmenter le nombre de directions correctement traitées. Les résultats des différentes interpolations sont montrés en Figure 10.38.

2.1.2 Interprétation des résultats

- Comme prévu, les méthodes **IAD**, **AQua** et **spline** créent un effet d'escalier prononcé sur tous les contours présents dans l'extrait utilisé dans la comparaison de la Figure 10.38.
- La méthode **NEDI** ne parvient pas non plus à éviter les effets d'escalier. Elle paraît même accentuer cet artefact en créant un nombre assez importants de faux pixels sur les contours

des bandes. Toutefois, l'impact visuel de l'effet d'escalier semble diminuer sur le huitième contour, soit environ autour de 80°.

– La méthode **NOAI** traite clairement les contours de façon directionnelle à partir du septième contour, soit à partir de $81,87°$. A partir de cet angle, les contours ont un aspect très correct même si quelques faux pixels apparaissent le long du huitième et neuvième contour.
– Enfin, l'observation des résultats obtenus par **notre interpolation** montre que l'effet d'escalier est diminué à partir du troisième contour, soit $86,99°$. Il n'est plus visible à partir du sixième contour, c'est-à-dire $82,87°$. Remarquons que notre méthode permet, sur les premiers contours uniquement, de diminuer l'effet d'escalier sans toutefois l'éliminer complètement. Ceci est dû au réglage du paramètre de variance σ_θ du correcteur Gaussien, qui fait décroître les coefficients trop éloignés du pixel central. Ainsi, lorsque le contour est filtré avec un tel correcteur, il n'est pas tout à fait reconstruit même lorsque sa direction est correctement détectée. Ce choix se justifie cependant dans le cas d'images naturelles, où il permet de conserver une bonne cohérence des pixels interpolés dans leur voisinage proche.

Le tableau 10.36 permet de synthétiser les résultats observés pendant cette étude. Il illustre l'angle à partir duquel l'effet de la correction directionnelle est visible, et l'angle à partir duquel le contour est corrigé de façon parfaite (aucun effet d'escalier visible). Du fait de la symétrie de la détection de direction et de la construction du correcteur Gaussien, cette interprétation peut également être appliquée aux directions horizontales. Ainsi par analogie, les contours orientés à 3° ou plus, seront traités par notre correcteur Gaussien.

Méthode d'interpolation	Effet visible de la correction	Correction parfaite du contour
AQua	15.9°	18.4°
IAD	18.4°	26.6°
NEDI	9.4°	14.0°
NOAI	8.1°	8.1°
Méthode proposée	3.0°	7.2°

FIGURE 10.36 – Tableau récapitulatif des résultats obtenus pour le test d'angle minimal de correction.

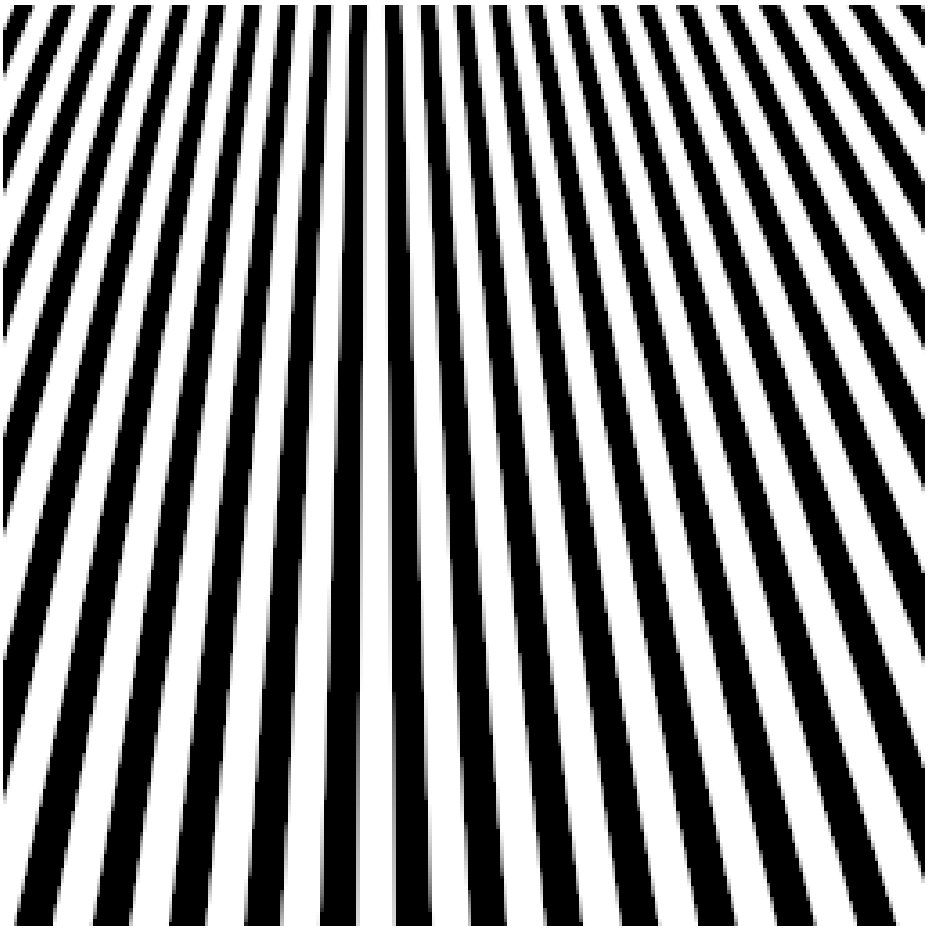
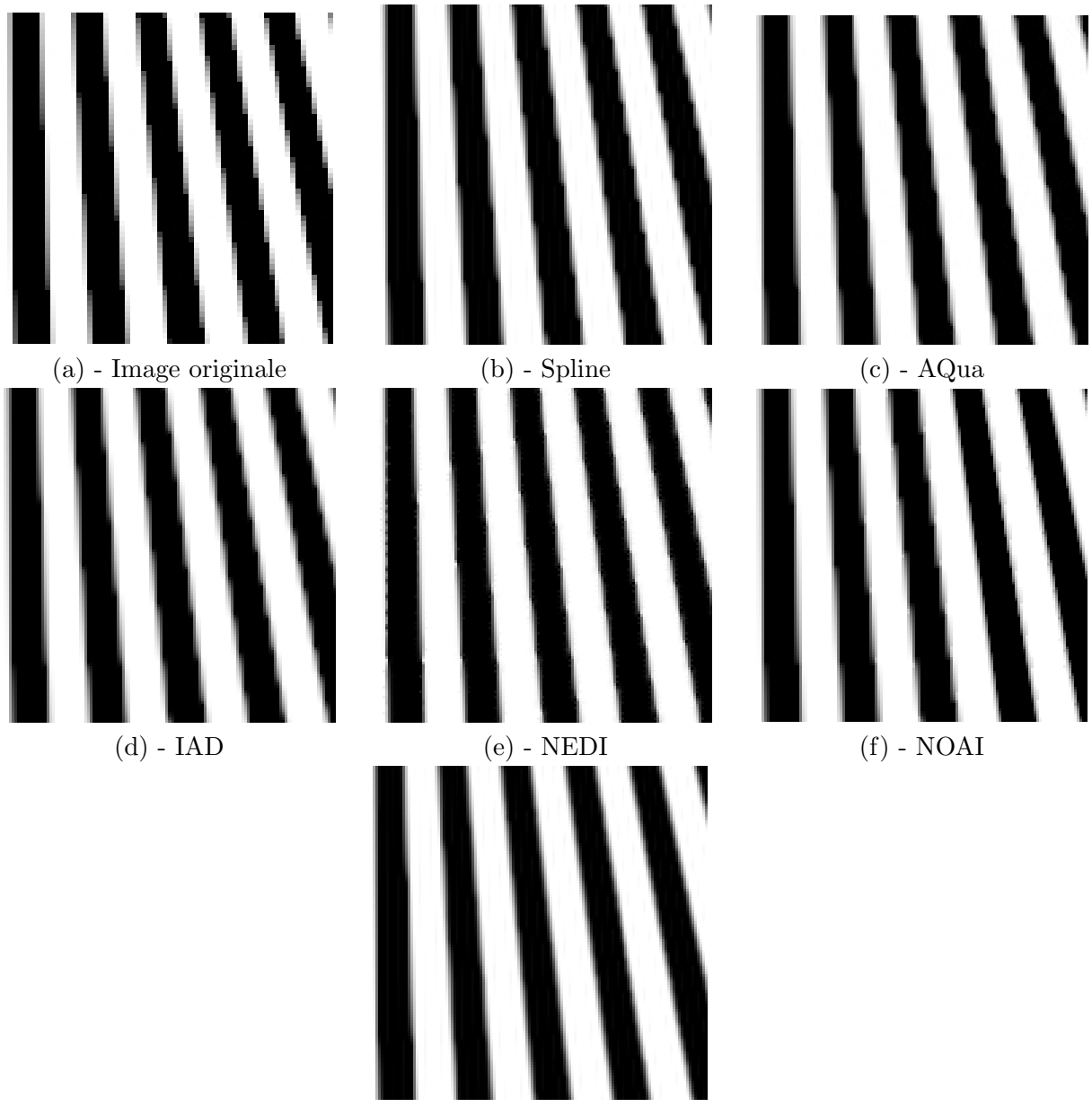


FIGURE 10.37 – Image utilisée pour tester la direction à partir de laquelle les contours sont traités correctement par les différentes méthodes d'interpolation.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.38 – Résultats obtenus par les différentes méthodes d'interpolation sur l'image de synthèse contenant les contours orientés.

2.2 Etude de l'effet de l'aliasing

2.2.1 Image de test

Ce second test effectué sur des images de synthèse permet d'illustrer le comportement de chacune des méthodes d'interpolation dans un cas où de l'aliasing est présent. Cet artefact est créé en augmentant progressivement la fréquence de cercles concentriques construits à partir du centre de l'image, jusqu'à ce que la fréquence des cercles deviennent supérieure à la fréquence de la grille d'échantillonnage. L'aliasing, dû au repliement de spectre, se manifeste en faisant apparaître des contours circulaires de direction localement perpendiculaire aux contours des cercles. Cette image est présentée en Figure 10.39. Les résultats de l'interpolation de l'image *zoneplate* sont présentés dans l'image 10.40. L'extrait utilisé pour comparer les méthodes permet d'étudier leur comportement dans une zone comportant de un fort aliasing.

2.2.2 Interprétation des résultats

- L'interpolation **spline** ne parvient à reconstruire les contours des cercles, lorsque la distance entre deux cercles devient trop faible. L'aliasing est donc toujours visible. Cependant, il est réparti de façon assez homogène ce qui a pour conséquence de ne pas trop attirer l'attention.
- Les interpolations **IAD** et **AQua** ont un comportement assez similaire sur cette image. Elles paraissent augmenter l'effet d'aliasing en renforçant l'importance visuelle des cercles parasites. Elles parviennent uniquement à reconstruire correctement les contours des cercles orientés à 45°, mais dégradent l'aspect des autres contours. Pour ces deux méthodes, à la fois la détection de direction de contour et l'interpolation sont mises en défaut, même lorsque l'aliasing est peu présent.
- La méthode **NEDI** est fortement perturbée par l'aliasing introduit. Les covariances basse-résolution qu'elle estime pour l'interpolation sont très influencées par les cercles parasites de l'image basse-résolution. En conséquence, l'interpolation accentue l'aliasing et produit une image fortement dégradée par rapport à l'image basse-résolution, et par rapport aux autres interpolations.
- L'interpolation **NOAI** est également perturbée par l'aliasing et ne parvient pas à reconstruire les contours des cercles. De plus, les endroits où la détection de direction est mise en défaut sont très visibles sur l'image. Dans ces zones, ce sont les contours dus aux cercles parasites qui sont reconstruits. Lorsque l'aliasing est trop présent, il est même difficile à l'oeil nu de savoir quelle est la direction des contours. Cependant, l'alternance d'une interpolation dans une direction puis dans l'autre produit une image peu réaliste et peu agréable à l'oeil puisque la continuité des contours est perturbée.
- Enfin, **notre interpolation** montre également ses limites sur cette image. La détection de direction est en effet mise en échec aux endroits où les cercles parasites ont une importance pratiquement équivalente aux cercles originaux. Ces endroits correspondent d'ailleurs assez bien à ceux pour lesquels la méthode utilisée dans l'interpolation NAOI était également défaillante. Cela a pour effet de produire des frontières de blocs visibles, provenant de la division en quad-tree, dans lesquels il est clairement possible de voir que l'interpolation a été réalisée dans la direction de l'aliasing. L'interprétation de nos résultats est donc sensiblement identique à celle de la méthode NOAI. En revanche, lorsque la détection de direction capte la bonne direction des contours, ces derniers sont mieux reconstruits que pour les autres méthodes, même lorsque leur fréquence est élevée. Ceci prouve l'efficacité du correcteur Gaussien qui permet de retrouver l'information de direction même lorsque celle-ci n'est plus visible dans l'interpolation spline de référence.

De façon assez paradoxale, l'étude de cet exemple entraîne une conclusion inverse de celle tirée dans l'exemple précédent. Parmi les méthodes directionnelles, les interpolations qui génèrent les

artefacts les moins gênants sont les interpolations AQua et IAD. L'impact visuel des artefacts engendrés est en effet relativement faible puisque leur répartition est assez homogène. Même si la méthode NOAI et la nôtre reconstruisent mieux la plupart des directions, les discontinuités engendrées sont trop perturbantes pour que l'image soit réaliste. L'oeil est en effet plus habitué à une répartition homogène des artefacts dans une image telle que le propose l'interpolation spline. Cela provient sûrement du fait que la répartition homogène des défauts sur toute l'image les rend moins visibles, mais également au fait que l'oeil est habitué aux artefacts produits par cette interpolation puisque celle-ci est souvent utilisée dans des applications usuelles et les logiciels de traitement d'images grand public.

2.3 Conclusion sur les exemples de synthèse

L'étude de ces deux exemples de synthèse nous a permit de mieux évaluer le comportement des différentes méthodes d'interpolation. La première étude a mis en avant l'efficacité de notre méthode pour traiter les contours proches de l'horizontale et de la verticale. L'effet d'escalier est en effet fortement réduit par rapport aux autres méthodes sur ces contours. La raison principale réside dans le fait que c'est notre méthode qui s'autorise le voisinage d'interpolation le plus important, ce qui permet d'utiliser des pixels éloignés du pixel à interpoler. Les autres méthodes limitent ce voisinage afin de garantir la cohérence du pixel interpolé dans son voisinage proche. Notre méthode, elle, peut se permettre d'utiliser des pixels éloignés sans risquer de dégrader la cohérence des pixels interpolés car elle se base sur une interpolation de référence (spline) qui assure déjà cette cohérence. L'utilisation d'une interpolation de référence se justifie donc par la possibilité de prendre un voisinage d'interpolation plus grand.
De plus, cet exemple met en valeur l'intérêt d'utiliser en priorité des grands blocs lors de la détection de direction. Ce sont eux qui offrent la meilleure résolution angulaire et donc qui permettent de détecter les angles proches de la verticale ou de l'horizontale. Sans l'estimation efficace préalable, l'interpolation ne pourrait évidemment pas corriger les artefacts engendrés.

Dans un deuxième temps, nous avons étudié une image comportant de l'aliasing qui perturbe la détection de direction réalisée en amont de l'interpolation. L'aliasing introduit est suffisamment important pour qu'il trompe l'estimation de direction. Cet exemple nous a donc permit d'observer le comportement des méthodes d'interpolation lorsque la détection de direction est défaillante. Il est alors possible de remarquer que les méthodes qui donnaient de meilleurs résultats jusqu'à maintenant (NOAI et la notre), créent les artefacts les plus gênants puisqu'ils parviennent à reconstruire de façon cohérente la direction parasite créée par l'aliasing. A l'inverse, les autres méthodes ne reconstruisent ni la bonne direction, ni la mauvaise ce qui produit un résultat au final plus homogène et plus habituel du point d'un utilisateur classique. Cet argument doit être pris en compte lors du choix de la méthode d'interpolation puisque selon le contenu des images à traiter, ce genre d'artefacts peut être rédhibitoire pour des applications comme l'agrandissement de photographies ou le médical.

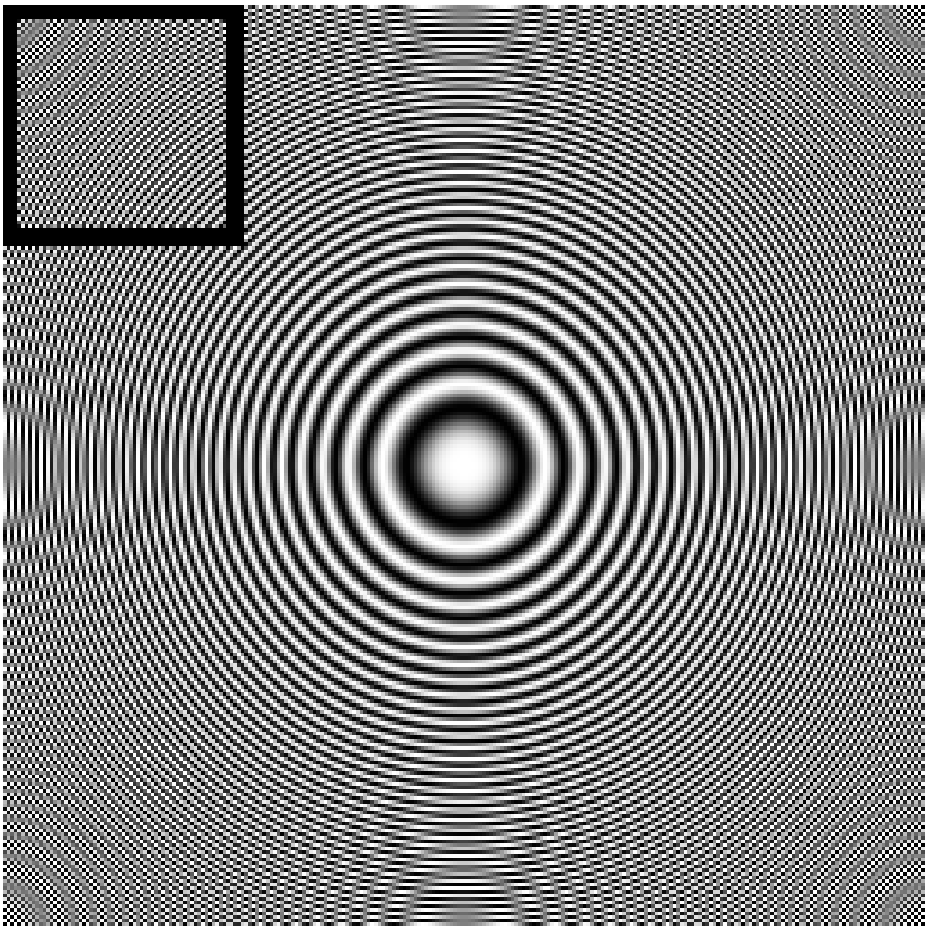
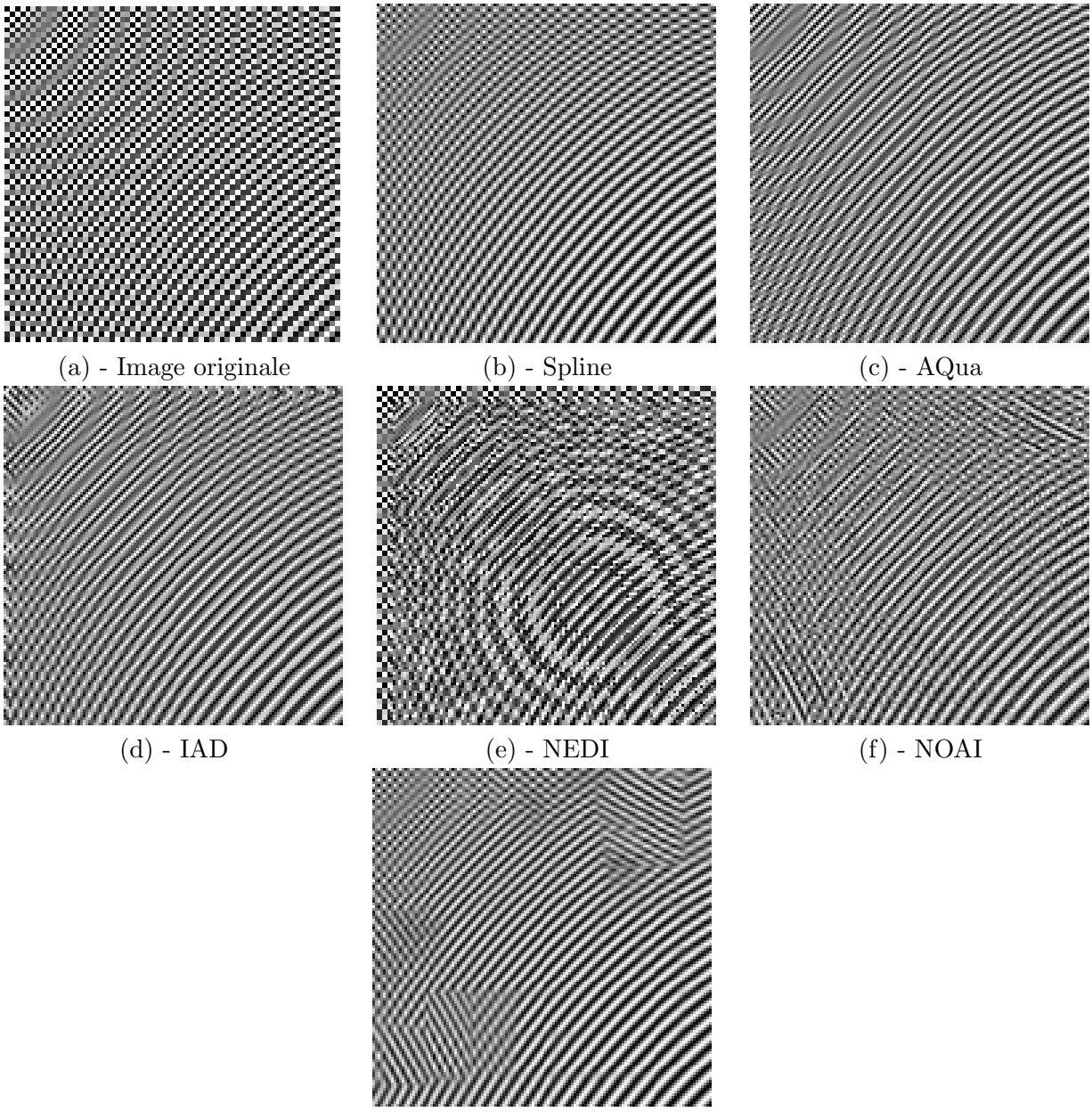


FIGURE 10.39 – Image *zoneplate* utilisée pour étudier le comportement des méthodes d'interpolation dans le cas ou de l'aliasing est présent.



(a) - Image originale (b) - Spline (c) - AQua

(d) - IAD (e) - NEDI (f) - NOAI

(g) - Méthode proposée

FIGURE 10.40 – Résultats obtenus par les différentes méthodes d'interpolation sur l'image de synthèse *zoneplate*.

3 Comparaison objective des résultats

La partie précédente a mis en valeur la supériorité du rendu visuel de notre méthode dans la plupart des situations présentées, du fait du faible nombre d'artefacts gênants et du rendu homogène des résultats. Il faudrait cependant une évaluation subjective des résultats pour confirmer cette impression. Afin d'étoffer cette étude, nous avons comparé les résultats obtenus grâce à des métriques objectives, à savoir le SSIM et le PSNR. Ces deux métriques sont choisies du fait de leur large utilisation et de leur complémentarité.

3.1 Procédure de test

Les tests ont été réalisés sur des images naturelles classiques, en essayant d'avoir les contenus les plus variés possible. Les images utilisées pour ce test sont présentées en Annexe B. Les images sont tout d'abord sous-échantillonnées d'un facteur deux, sans filtrage passe-bas préalable, puis interpolées à leur résolution d'origine avec chaque méthode. Le filtrage passe-bas avant le sous-échantillonnage n'est pas effectué afin de faire apparaître des artefacts d'aliasing sur les contours. Les résultats sont ensuite comparés à l'image originale grâce au PSNR et au SSIM. Notons que la méthode NOAI, pour laquelle seules les images originales agrandies nous ont été fournies, ne fait pas partie de cette étude. Afin de mieux étudier l'impact de chaque interpolation sur les contours, les deux métriques sont calculées uniquement sur les contours des images interpolées. La carte de contours est construite à partir d'un opérateur de Canny, suivi d'une dilatation de trois pixels de chaque côté du contour.

3.2 Résultats

Les résultats obtenus sont montrés dans les deux graphiques en Figure 10.41 et 10.42. Le tableau contenant les résultats exacts est présenté en Annexe C. De manière générale, les valeurs de PSNR obtenues sont légèrement en faveur de notre méthode (six images sur dix). La moyenne des PSNR sur les dix images est également légèrement meilleure que celle de IAD (0.1 dB) et que celle de la spline (0.175 dB), mais significativement supérieure par rapport à NEDI (0.54 dB) et à AQua (1.38 dB). Par contre, la métrique SSIM donne des résultats beaucoup moins clairs. En effet, les moyennes sont très proches pour les cinq méthodes comparées et c'est la spline qui obtient la meilleure moyenne. Mais c'est la méthode AQua qui obtient cinq fois les meilleurs résultats, alors que les méthodes spline et IAD les obtiennent respectivement quatre et une fois. L'interprétation de ces résultats va être effectuée en plusieurs parties. La première partie s'intéressera à tirer des conclusions sur les résultats en fonction de l'image étudiée, et la deuxième tentera d'expliquer les différences de résultats obtenues entre PSNR et SSIM.

3.3 Interprétation des résultats

Notons avant tout que la disparité des résultats obtenus entre PSNR et SSIM rend l'interprétation des résultats peu évidente. De ce fait, nous essaierons de baser nos conclusions dans des cas ou les deux métriques ont une tendance commune.

3.3.1 Comportement général de l'interpolation spline

De manière générale, l'interpolation spline est performante pour les images contenant des textures de type aléatoire. Ainsi, les résultats de l'interpolation sont bons pour les images *Cameraman* (texture au niveau de l'herbe), *Lena* (texture au niveau des cheveux), *Mandrill* (texture sur le pelage) et *Cemetry* (texte et textures au niveau des briques, arbres et herbe). L'image *Lighthouse*, qui contient également des textures (herbe), ne donne pas de bons résultats en raison des nombreux contours rectilignes qui sont mal interpolés par le noyau spline. Les métriques

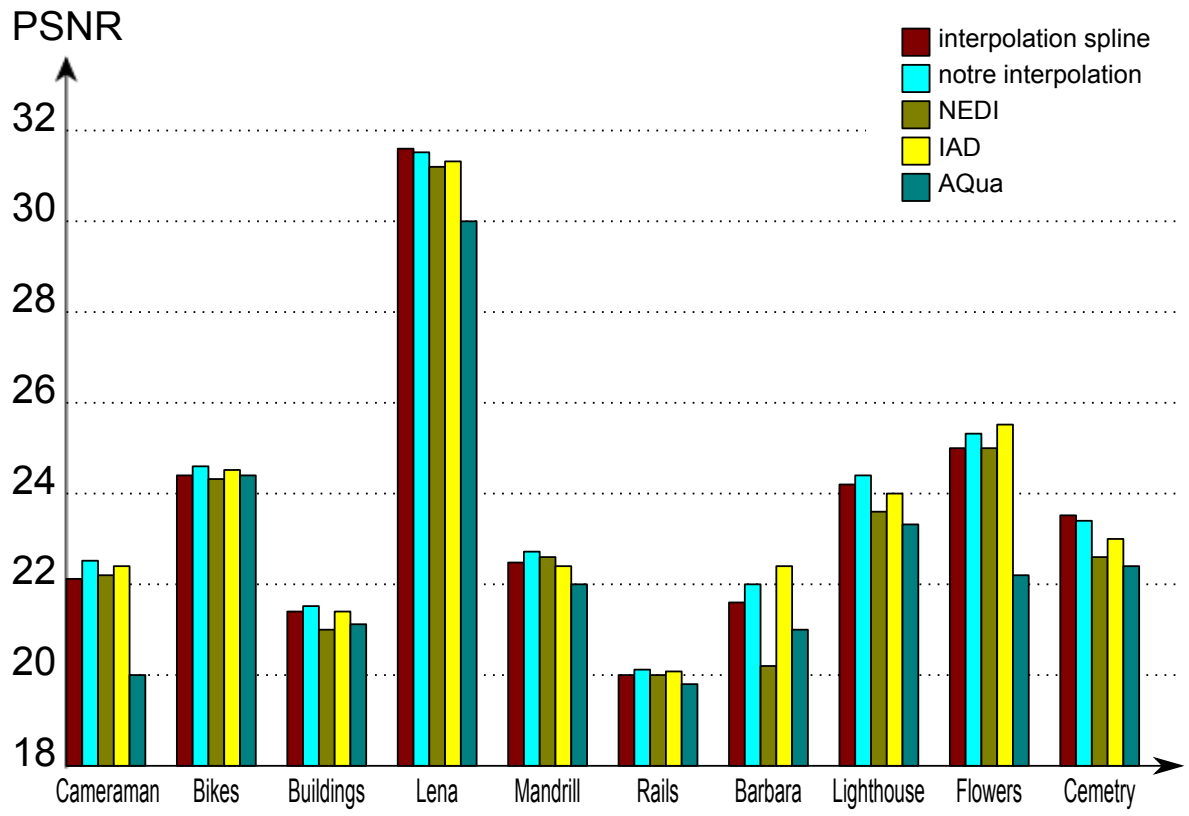


FIGURE 10.41 – Graphique contenant les résultats de PSNR.

objectives permettent ici de confirmer la supériorité visuelle de l'interpolation spline par rapport aux méthodes directionnelles pour ce type de contenu.

3.3.2 Comportement général des interpolations directionnelles

D'autre part, les méthodes directionnelles donnent de meilleurs résultats pour les images qui présentent une forte densité de contours de forte amplitude, voire même de textures directionnelles (*Rails*, *Buildings*, *Barbara*). Dans ces situations, l'impact des interpolations directionnelles est plus fort et ces structures sont interpolées correctement, ce qui justifie leurs bons résultats au niveau des métriques. Certaines images (*Mandrill* et *Cameraman*) donnent un meilleur résultat de PSNR pour notre méthode et un meilleur SSIM pour la spline. Ce résultat mixte provient sûrement du fait que ces deux images contiennent à la fois un nombre important de contours difficiles à interpoler pour la spline (barres du tripode) et correctement interpolés par notre méthode, et des zones de textures aléatoires favorables à la spline (herbe et mur du phare).

3.3.3 Comportement général de notre interpolation

Si notre méthode donne de meilleurs résultats globaux de PSNR, il est cependant difficile de conclure quant à sa supériorité objective du fait de la proximité des résultats obtenus par rapport au nombre d'images testées, et des résultats mitigés donnés par le SSIM. En effet, remarquons que notre méthode n'améliore dans aucun cas le SSIM de l'interpolation spline, et que son PSNR est assez proche des autres méthodes alors que visuellement, l'amélioration était significative. Nous allons tenter dans la partie suivante d'expliquer ce phénomène.

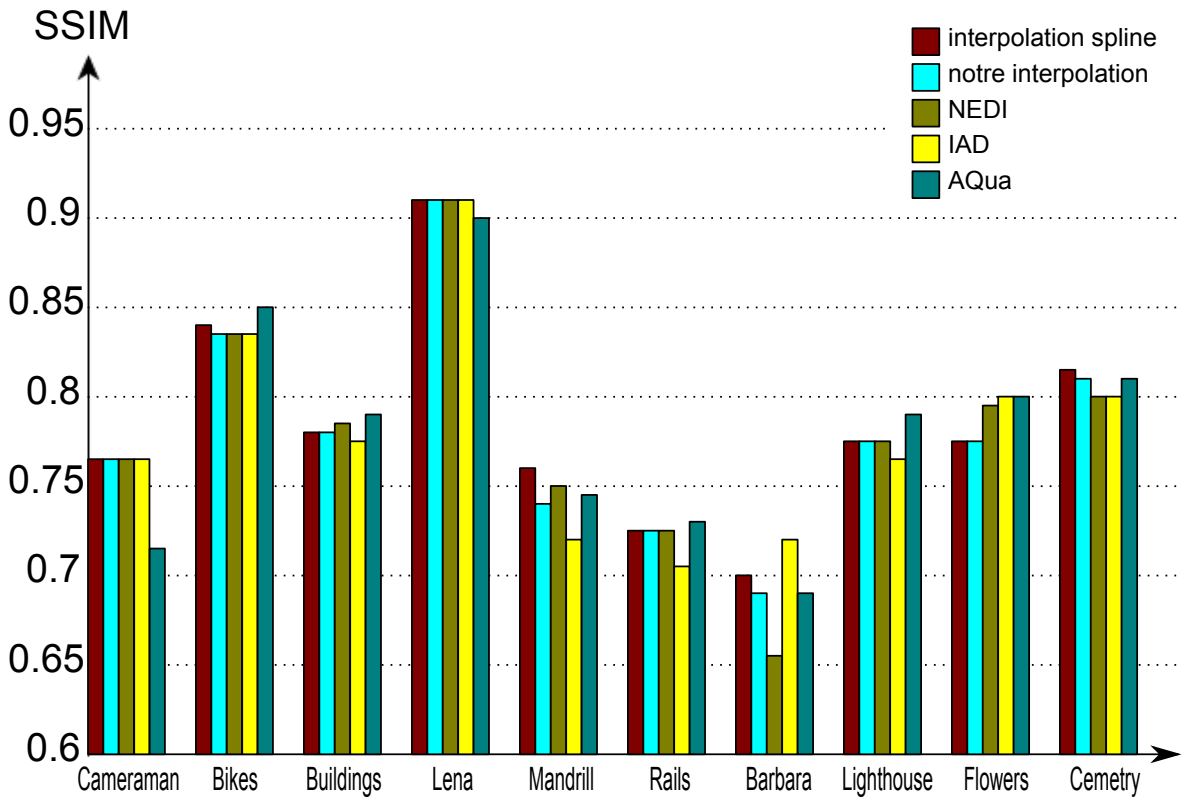


FIGURE 10.42 – Graphique contenant les résultats de SSIM.

3.4 Discussion sur les mesures de PSNR et SSIM

Dans cette partie, nous allons interpréter les différences de comportement des deux métriques sous deux angles différents. Nous aborderons dans un premier temps les causes du faible écart des mesures de PSNR de notre méthode par rapport aux autres méthodes, visuellement moins bonnes. Puis nous expliquerons pourquoi d'après nous, les méthodes IAD et AQua donnent de si bons résultats de SSIM par rapport à leur comportement observé dans la partie précédente.

3.4.1 Interprétation des résultats de PSNR

Alors que notre méthode semblait rendre des résultats visuellement plus cohérents, il apparaît que les mesures de PSNR ne sont pas significativement supérieures en particulier par rapport aux méthodes spline et IAD. A notre avis, le fait que notre méthode ne conserve pas les pixels originaux, du fait de l'application du correcteur gaussien, peut expliquer cet écart. En effet, lors d'une interpolation d'un facteur 2×2, les pixels originaux représentent un quart du nombre de pixels total et ceux-ci sont conservés par toutes les autres méthodes d'interpolation comparées dans cette partie.
De plus, les objets contenus dans les images sous-échantillonnées étant deux fois plus petits que ceux des images originales, les changements d'orientation de contours sont plus fréquents et les méthodes de détection de direction sont plus facilement mise en défaut. Par rapport à l'interpolation des images originales, les bénéfices de l'interpolation directionnelle nous ont en effet paru moindres. Ceci justifie également les faibles écarts de PSNR entre l'interpolation spline et les méthodes directionnelles.

3.4.2 Interprétation des résultats de SSIM

Enfin, au niveau des résultats du SSIM, l'article [Wang 2004] indique que cette mesure est basée sur trois facteurs. Le premier est une comparaison locale de luminance, le deuxième est une comparaison de contraste et le troisième une comparaison de structure. Le correcteur gaussien appliqué dans notre méthode induit, en plus de la perte des pixels originaux, une perte de contraste et un léger flou qui sont pris en compte dans le calcul de la valeur de SSIM. En revanche, les faux pixels et faux contours introduits par les méthodes IAD et AQua ne semblent pas affecter cette métrique puisque leurs résultats sont assez élevés par rapport à l'impression visuelle. Il se peut même que ces artefacts jouent en leur faveur, puisqu'ils peuvent être interprétés comme des transitions franches, et donc donner l'impression d'être en présence de contours nets.

3.5 Conclusion sur l'évaluation objective

L'étude des résultats de ces deux métriques objectives ne fait pas apparaître clairement la supériorité d'une méthode sur les autres, alors que visuellement, les différences sont tout de même importantes. Ceci est d'autant plus étonnant que les deux métriques sont calculées sur les contours des images uniquement. L'intérêt des méthodes directionnelles n'est donc pas facilement démontrable par ces métriques. D'ailleurs, il est en général difficile de trouver des articles sur les interpolations directionnelles qui les utilisent. Lorsque des comparaisons sont faites, la plupart se limitent à des comparaisons subjectives.
Nous pensons que le fait de travailler sur des images sous-échantillonnées est un inconvénient pour les méthodes directionnelles. En effet, ces dernières sont plus performantes lorsque les contours des objets sont plus éloignés les uns des autres. Cela facilite à la fois l'estimation de direction et l'interpolation dans le cas des méthodes utilisant des grands voisinages d'interpolation. Les résultats obtenus sur des images sous-échantillonnées sont donc peu valorisants pour les méthodes d'interpolation directionnelles.
Malgré cela, quelques tendances ont pu être relevées. Les méthodes d'interpolation directionnelles se comportent en général mieux que l'interpolation spline isotrope sur des images contenant un grand nombre de contours, et peu de textures aléatoires. La solution pour réaliser une évaluation plus fiable serait donc de faire passer des tests subjectifs, dans le contexte défini par les normes de l'ITU ([BT.500-12 2009]), afin de faire comparer les différentes méthodes par des utilisateurs non experts et dans des conditions de tests optimales.

4 Perspectives

4.1 Amélioration de l'interpolation

4.1.1 Meilleure localisation des contours à corriger

Un point faible de notre méthode est la dégradation des textures, et la perte de certains détails. Ces dégradations sont induites par le masque de Gabor qui manque quelques fois de précision. En effet, le masque est construit à partir de filtres de Gabor puis est normalisé pour la pondération de l'image corrigée avec l'image spline dans le schéma général de l'interpolation décrit en partie 9.2.3. La normalisation effectuée sur le masque implique donc l'utilisation exclusive de la version corrigée pour les coefficients maximaux de la sortie du filtre de Gabor. Dans les cas où la direction détectée n'est pas représentative du bloc (mauvaise estimation ou contenu aléatoire), l'utilisation exclusive de la version corrigée entraîne des pertes de détails ou altère l'aspect décorrélé de textures aléatoires en ajoutant de la cohérence selon la direction détectée. Ce phénomène explique l'apparition de faux contours dans certaines situations.

Pour remédier à ce problème, il faudrait utiliser la sortie du filtre de Gabor avant la normalisation,

afin de conserver l'amplitude réelle de la sortie et de mieux étudier la corrélation entre le contour du bloc et la direction étudiée. Cependant, selon la fréquence et l'amplitude des contours, l'amplitude de sortie peut varier fortement selon les paramètres fréquentiels utilisés pour la construction des filtres de Gabor. C'est pourquoi la sortie du filtre ne peut être utilisée telle quelle. Elle doit obligatoirement être traitée de façon relative au contenu du bloc. Elle pourrait par exemple être reliée directement à la norme des gradients des contours pour adapter la réponse du filtre à l'amplitude du contour. Enfin, pour gérer le placement fréquentiel des filtres de Gabor, nous pourrions penser à utiliser l'information de résolution optimale détectée en amont de l'analyse directionnelle. Jusqu'à maintenant, cette information ne nous servait qu'à optimiser la recherche de direction du bloc. Elle pourrait également être utilisée pour adapter la fréquence des filtres de Gabor à la fréquence du contour afin de rendre plus efficace la localisation des pixels du contour. Nous avons testé cette dernière approche sur plusieurs images, mais le gain sur la localisation est encore faible car le fait d'utiliser des filtres de Gabor *basses-fréquences* fait implicitement diminuer la résolution spatiale de l'étude.

Une autre approche que les filtres de Gabor pourrait alors être envisagée, pour essayer de localiser sans normalisation les contours orientés dans la direction estimée. L'utilisation d'un autre procédé permettrait également de réduire le nombre et le temps de calcul qui est largement impacté par les filtres de Gabor, car ils demandent de grands supports pour être appliqués et impliquent donc un grand nombre de calculs.

4.1.2 Amélioration de la netteté

La netteté de l'image interpolée est un point sur lequel notre méthode peut également être améliorée. Les informations obtenues précédemment concernant l'orientation du contour et la localisation de ce contour dans le bloc, peuvent être utilisées pour réhausser la transition. Il existe en effet des techniques basées sur des filtrages passe-haut qui effectuent cette opération. L'inconvénient de l'utilisation de tels filtres est l'augmentation du niveau de bruit puisque toutes les transitions sont réhaussées.
Le fait de connaître les pixels appartenant aux contours ainsi que l'orientation des transitions permettrait d'appliquer uniquement ces transformations dans la direction et aux endroits pertinents.

4.2 Facteurs non entiers

Dans la plupart des applications industrielles, l'agrandissement d'un facteur non entier est nécessaire, dû par exemple à la différence de résolution d'acquisition et d'affichage. Il arrive également que les proportions de la résolution de l'image affichée changent par rapport à l'image originale, comme c'est le cas lors d'un affichage en 16 :9 par exemple. Pour l'instant, notre méthode n'est applicable que dans le cas de facteurs d'interpolation entiers, mais nous allons voir que seules quelques modifications doivent être appliquées pour gérer ce cas.

Certaines méthodes présentées plus haut, comme les méthodes IAD et NEDI, ne peuvent par construction qu'interpoler des images par des facteurs entiers. Notre interpolation étant basée sur un noyau spline, elle peut être effectuée pour n'importe quel facteur. C'est le correcteur Gaussien qui doit être adapté en fonction des proportions de la nouvelle grille d'échantillonnage pour utiliser les bons pixels correcteurs et reconstruire correctement le contour. En effet, lors d'un changement de proportion de la grille d'échantillonnage, la direction détectée doit être adaptée à la nouvelle grille. Cette situation est mise en évidence dans le schéma de la Figure 10.43 pour le passage d'un bloc de taille 5×5 pixels en un bloc de 16×9 pixels. Dans cet exemple, il est possible de voir que la direction de la droite dans le cas du bloc 5×5 n'est pas la même que dans le bloc interpolé et la recherche des pixels correcteurs devra être adaptée en fonction. Cependant, cela ne

pose aucun problème étant donné que la transformation géométrique a réaliser est fixe, et qu'elle est facilement calculable en tenant compte du changement de proportion entre les deux grilles.

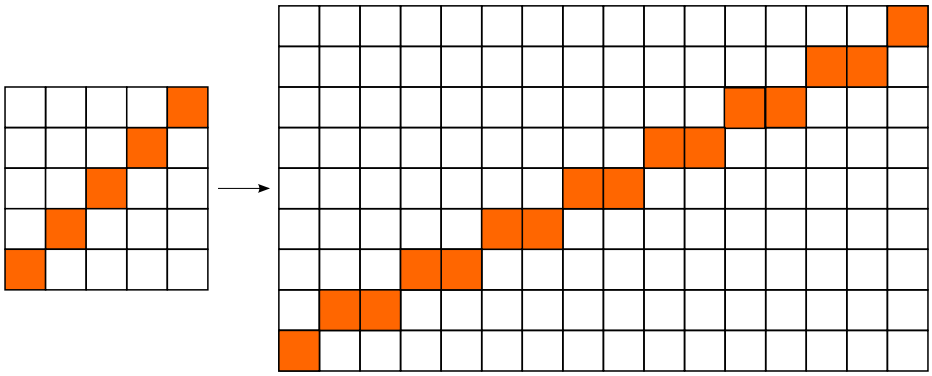


FIGURE 10.43 – Exemple de re-échantillonnage d'un bloc carré de 5×5 pixels en un bloc de 16×9 pixels.

4.3 Application à des séquences d'images

L'amélioration de résolution de séquences d'images a été brièvement testée en utilisant notre algorithme, et comparée à des séquences interpolées grâce à la fonction spline. Le gain en qualité apporté par notre interpolation sur des images a déjà été illustré lors des parties précédentes (élimination des artefacts classiques apparaissant sur les contours), et il est intéressant d'observer l'impact de cette amélioration en ajoutant la composante temporelle.

Les artefacts présents sur les contours de l'interpolation spline (principalement du jaggy et des effets d'escalier) influent sur la perception temporelle de la séquence d'images. En effet, cette non-homogénéité spatiale implique également une non-homogénéité temporelle, visuellement perturbante puisque des hautes fréquences temporelles qui attirent l'oeil sont créées. Le rendu homogène des contours obtenus par notre méthode permet d'éviter ce genre de dégradation. De plus, la réduction de l'aliasing obtenue par notre interpolation augmente la supériorité visuelle de notre méthode pour un artefact temporellement particulièrement gênant.

En revanche, les faux contours introduits par notre méthode sur des textures aléatoires, ont un impact temporel important du fait du changement rapide de structures orientées image après image. Les hautes fréquences temporelles introduites par ces variations rapides attirent l'attention et perturbent la qualité de la séquence. Lors d'une comparaison subjective de ces méthodes, nous pouvons imaginer que celles qui introduisent des faux contours ainsi que des faux pixels seront fortement pénalisées du fait des variations temporelles indésirables.

Afin de confirmer les conclusions que nous avançons dans cette partie, il serait évidemment plus correct d'effectuer des évaluations subjectives sur des séquences. En effet, nous avons pu voir que les métriques spatiales avaient déjà du mal à mettre en valeur les apports des méthodes directionnelles. Ceci est donc d'autant plus vrai lorsque la composante temporelle est ajoutée, puisque les artefacts temporels sont encore plus difficiles à quantifier par des métriques que les artefacts spatiaux. Grâce à ces tests subjectifs, nous pourrions donc évaluer l'impact visuel sur les séquences du meilleur rendu de contours que notre méthode propose. De plus, nous pourrions évaluer l'influence visuelle de la netteté supérieure des autres méthodes par rapport à la nôtre, au détriment de l'ajout de faux pixels et de contours moins homogènes. Cette évaluation devra être réalisée afin de proposer une comparaison plus complète des méthodes d'interpolation. Les résultats de cette

comparaison nous permettraient de guider les prochaines évolutions de notre algorithme vers les résultats optimaux du point du vue de l'observateur.

Conclusion générale

L'objectif initial de cette thèse consistait à créer un algorithme d'interpolation d'images basé sur une approche par ondelettes géométriques pour l'affichage d'images de télévision sur des écrans haute-définition. Au bout de trois ans, nous sommes parvenus à mettre au point une méthode d'analyse multirésolution des directions des contours, ainsi qu'une méthode d'interpolation directionnelle qui tire partie des résultats de l'estimation de direction.

Synthèse des travaux

L'estimation de direction des contours est basée sur la création d'une partition de l'image en *quadtree*, qui permet, contrairement aux méthodes d'estimation linéaires, de capter les discontinuités des orientations par exemple dans des cas d'intersections de contours. Les blocs du *quadtree* sont adaptés en fonction des caractéristiques des contours afin d'isoler dans chaque bloc au plus une direction de contour. La direction associée à chaque bloc est détectée en calculant la droite naïve le long de laquelle le bloc présente le moins de variations. L'approche multirésolution réalisée en amont de l'étude directionnelle permet d'analyser efficacement les contours hautes et basses fréquences.
L'efficacité de cette méthode a ensuite été testée et comparée à deux autres méthodes sur des images de synthèse, des images bruitées ainsi que des images naturelles. Il est ressorti de cette comparaison que les trois méthodes ont un comportement similaire dans des grandes et moyennes tailles de blocs, mais que notre méthode offre des résultats bien supérieurs pour les petites tailles de blocs en ayant d'assez loin l'erreur d'estimation la plus faible. Dans les cas bruités, ces comportements se sont vérifiés et l'étude d'images naturelles a mis en évidence la meilleure robustesse de notre approche dans des situations critiques (contours tronqués et contours à faible contraste).

Par la suite, nous avons présenté notre méthode d'interpolation basée sur l'estimation directionnelle préalable. Nous utilisons une interpolation isotrope de référence (cubique spline), que nous corrigeons par filtrage Gaussien dans la direction des contours. Le fait de se baser tout d'abord sur une méthode isotrope a deux avantages principaux. Cela autorise une prise de risque plus grande lors du filtrage directionnel, c'est-à-dire un voisinage de correction plus grand, ce qui permet de corriger plus d'artefacts. De plus, cela permet d'éviter d'introduire de nouveaux artefacts, du fait de la cohérence initiale des pixels dans leur voisinage proche. Enfin, différentes méthodes d'interpolation directionnelles faisant référence dans l'état de l'art actuel ont été comparées avec la nôtre dans une dernière partie. Globalement, notre méthode permet d'obtenir un rendu visuel supérieur aux autres méthodes. De plus la robustesse de notre algorithme à des images de contenus très variés se prête bien à un affichage d'images de télévision proposant des images de natures très différentes. Cette évaluation nous a permis de mettre en évidence les points sur lesquels notre méthode peut être améliorée, en particulier au niveau de la netteté des transitions Enfin, l'homogénéité des résultats obtenus et le faible nombre d'artefacts introduits offrent éventuellement à notre méthode la possibilité d'être implémentée dans un futur circuit industriel.

Perspectives industrielles

Les deux parties de la thèse ont été réalisées dans une volonté d'une future application industrielle de la part de STMicroelectronics. Les méthodes mises au point ont donc été développées de sorte à être à la fois performantes, mais également adaptable à d'autres applications. Par exemple, l'algorithme de division en *quadtree*, chargé de l'analyse directionnelle multirésolution des contours d'une image peut être utilisé pour un certain nombre d'autres traitements qui pourraient tirer parti d'une telle analyse. Nous pensons notamment au débruitage ou à la compression d'images, qui peuvent utiliser l'information de directions de contours pour préserver l'aspect des transitions qui sont souvent dégradées lors de ces deux traitements. L'analyse directionnelle pourrait donc être placée en amont des différents algorithmes réalisés dans les circuits de boîtiers multimédia, afin d'améliorer toute la chaîne de traitements. De plus, la correction directionnelle effectuée grâce au filtrage Gaussien directionnel, peut être appliquée à n'importe quel noyau d'interpolation et en particulier à celui déjà en place dans les circuits de STMicroelectronics. Les concepts novateurs mis en place durant cette thèse peuvent donc être utilisés indépendamment et placés dans la chaîne de traitements actuellement mise en place.

Nous sommes cependant bien conscients que le chemin qui sépare nos travaux d'une implémentation dans un circuit est encore long, et ce pour plusieurs raisons. La première étape du transfert de notre algorithme aux équipes de recherche de STMicroelectronics a été réalisée au courant du mois de novembre 2010. A cette période, j'ai travaillé avec l'équipe de développement algorithmique basée à Singapour pendant une semaine, afin de présenter et d'intégrer mes travaux aux leurs. Sur place, des remarques très intéressantes sur les perspectives de mon approche en vue d'une implémentation concrète m'ont été faites :

- Premièrement, l'algorithme dans son état actuel est écrit sous l'environnement Matlab qui reste un language de simulation. Afin d'être intégré dans les chaînes de tests, une ré-écriture du code dans le langage utilisé à Singapour serait donc nécessaire. C'est un langage propriétaire de STMicroelectronics, proche du CUDA, qui autorise la parallélisation des calculs. L'algorithme entier doit donc être écrit en fonction pour satisfaire les contraintes imposées par ce langage.
- Deuxièmement, l'algorithme doit évidemment subir des modifications voire des simplifications en vue d'une application à des procédés en temps réel. En effet, les calculs nécessaires au fonctionnement de l'algorithme tel qu'il est écrit maintenant sont beaucoup trop importants pour que la méthode soit applicable en temps réel. Les simplifications possibles peuvent être effectuées à plusieurs endroits de l'algorithme, et en particulier sur le nombre de directions que la détection de direction peut estimer ainsi que sur l'utilisation des filtres de Gabor qui nécessitent à eux-seuls une proportion importante du nombre de calculs total.
- Enfin, les aspects de mémoires nécessaires au stockage des variables temporaires doivent également être abordés puisqu'ils ont une grande influence au niveau de l'architecture du circuit. Par exemple, le fait d'utiliser de grands voisinages lors de la correction directionnelle requiert le stockage en mémoire d'un grand nombre de lignes de l'image.

L'évolution des technologies d'architecture des circuits ainsi que l'importance de plus en plus grande attribuée à la qualité de l'image affichée ont pour conséquence l'augmentation des ressources disponibles dédiées aux algorithmes de traitement d'images. Notre méthode telle que nous l'avons laissée en fin de thèse, n'est pas en mesure d'être implémentée dans les circuits actuels. Cependant, nous espérons que les concepts que nous avons développés et intégrés au sein des équipes de développement algorithmique seront utilisés dans les circuits de STMicroelectronics à moyen terme.

Annexe A

Tableaux des biais des méthodes d'analyses directionnelles

	Taille du support	**BT**	**RADON**	**NOTRE METHODE**
Fréquence f_1				
	24×24	**0.0306**	0.2613	0.1184 (0.0483)
	12×12	**0.1912**	2.3380	0.5229 (0.0224)
	6×6	2.9665	6.6250	**0** (0)
Fréquence f_2				
	24×24	**0.0317**	0.1463	0.1182 (0.0487)
	12×12	**0.2536**	1.6056	0.6361 (0.0448)
	6×6	11.7680	8.7333	**0** (0)
Fréquence f_3				
	24×24	**0.0191**	0.0801	0.1261 (0.0512)
	12×12	**0.1975**	1.1549	0.5932 (0.0224)
	6×6	11.9665	4.2667	**0** (0)

FIGURE A.1 – Valeur des biais pour les trois fréquences testées dans des supports circulaires.

	Taille du support	**BT**	**RADON**	**NOTRE METHODE**
Fréquence f_1				
	16×16	**0.1525**	1.2578	0.5611
	8×8	**0.5290**	4.8310	2.0789
	4×4	6.8720	15.7333	**1.9011**
Fréquence f_2				
	16×16	**0.1052**	0.6376	0.6169
	8×8	**0.7145**	3.7042	2.1392
	4×4	32.8820	14.1333	**3.1181**
Fréquence f_3				
	16×16	2.3761	**0.4739**	0.5546
	8×8	1.9081	1.9859	**1.6633**
	4×4	43.1460	10.2667	**2.5096**

FIGURE A.2 – Valeur des biais pour les trois fréquences testées dans des supports carrés.

		Taille du support	**BT**	**RADON**	**NOTRE METHODE**
Fréquence f_1					
	σ_1	24×24	**0.1934**	0.2822	0.3193
		12×12	**0.7117**	2.4648	0.7972
		6×6	4.0658	8.6667	**0.2514**
	σ_2	24×24	**0.2709**	0.3206	0.3963
		12×12	**1.0107**	2.5775	1.1067
		6×6	5.2492	12.4000	**0.8279**
	σ_3	24×24	0.6452	**0.6201**	0.6521
		12×12	2.1358	3.0563	**1.7609**
		6×6	12.253	9.5333	**4.3901**
Fréquence f_2					
	σ_1	24×24	**0.1252**	0.1481	0.2781
		12×12	**0.6088**	1.6577	0.7103
		6×6	10.5982	9.5733	**0.1091**
	σ_2	24×24	0.1777	**0.1707**	0.3356
		12×12	0.9623	1.7402	**0.9051**
		6×6	11.3245	9.5407	**0.3142**
	σ_3	24×24	0.4112	**0.2892**	0.5102
		12×12	2.3957	1.8512	**1.3480**
		6×6	20.2656	9.5037	**1.5551**
Fréquence f_3					
	σ_1	24×24	0.0877	**0.0679**	0.2183
		12×12	**0.5055**	1.1690	0.6608
		6×6	18.8659	5.1333	**0.0909**
	σ_2	24×24	0.1152	**0.0662**	0.2695
		12×12	1.0268	1.1634	**0.7733**
		6×6	21.0086	4.6733	**0.1451**
	σ_3	24×24	0.4761	**0.1045**	0.4736
		12×12	4.9186	1.3211	**1.2443**
		6×6	26.8055	8.1467	**0.8545**

FIGURE A.3 – Valeur des biais pour trois fréquences et trois variances de bruit différentes.

	Taille du support	**BT**	**RADON**	**NOTRE METHODE**
Fréquence f_1				
	24×24	0.6287	1.0767	**0.5069**
	12×12	1.3995	5.0845	**1.3767**
	6×6	3.6090	16.8000	**1.7408**
Fréquence f_2				
	24×24	3.5110	**0.4390**	0.6647
	12×12	2.7450	2.8310	**1.7658**
	6×6	7.7040	40.6667	**1.5590**
Fréquence f_3				
	24×24	34.3633	**3.6551**	10.0818
	12×12	25.6036	15.0986	**13.0006**
	6×6	30.3410	35.0667	**27.8687**

FIGURE A.4 – Valeur des biais pour les trois fréquences sur des images compressées présentant des effets de blocs.

Annexe B

Résultats de notre interpolation

FIGURE B.1 – Image *Barbara* interpolée par un facteur 2×2 par la méthode proposée.

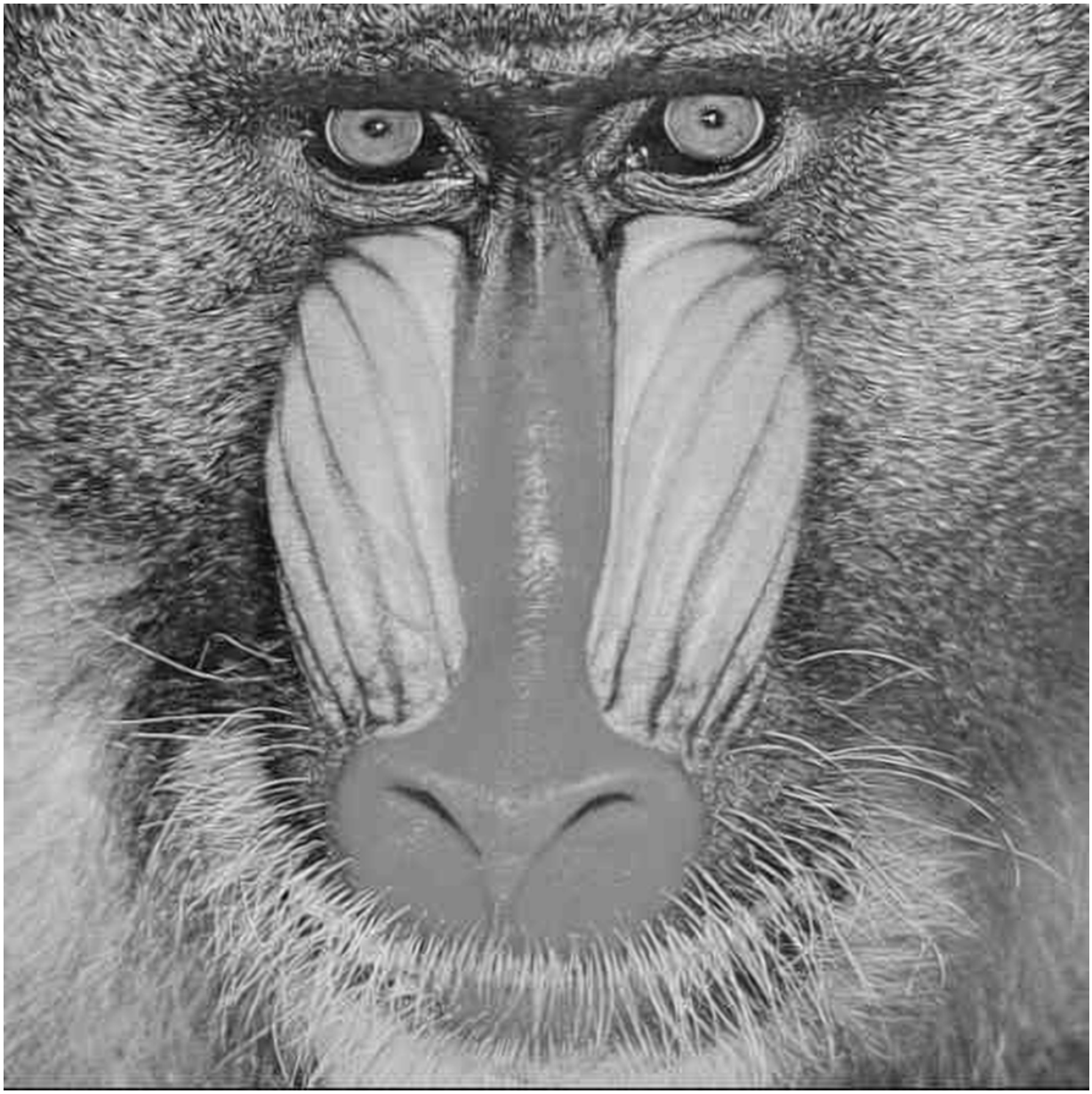


FIGURE B.2 – Image *Mandrill* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.3 – Image *Lighthouse* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.4 – Image *Flowers* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.5 – Image *Cameraman* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.6 – Image *Bikes* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.7 – Image *Cemetry* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.8 – Image *Buildings* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.9 – Image *Lena* interpolée par un facteur 2×2 par la méthode proposée.

FIGURE B.10 – Image *Rails* interpolée par un facteur 2×2 par la méthode proposée.

Annexe C

Résultats de PSNR et SSIM

Image	**Métrique**	**Spline**	**Méthode proposée**	**NEDI**	**IAD**	**AQua**
Cameraman	PSNR	22.131	**22.549**	22.186	22.461	20.081
	SSIM	**0.767**	0.766	0.765	0.766	0.727
Bikes	PSNR	24.382	**24.656**	24.293	24.502	24.445
	SSIM	0.839	0.836	0.836	0.834	**0.852**
Buildings	PSNR	21.407	**21.577**	21.032	21.419	21.172
	SSIM	0.779	0.782	0.787	0.775	**0.789**
Lena	PSNR	**31.654**	31.535	31.186	31.310	30.111
	SSIM	**0.913**	0.908	0.910	0.910	0.901
Mandrill	PSNR	22.487	**22.709**	22.588	22.402	22.059
	SSIM	**0.763**	0.742	0.750	0.720	0.748
Rails	PSNR	19.968	**20.136**	20.078	20.105	19.787
	SSIM	0.724	0.724	0.724	0.707	**0.733**
Barbara	PSNR	21.649	21.958	20.114	**22.358**	21.126
	SSIM	0.701	0.690	0.657	**0.721**	0.688
Lighthouse	PSNR	24.192	**24.367**	23.614	24.016	23.336
	SSIM	0.778	0.777	0.775	0.766	**0.779**
Flowers	PSNR	25.068	25.281	25.033	**25.548**	22.190
	SSIM	0.776	0.775	0.797	0.799	**0.806**
Cemetry	PSNR	**23.471**	23.393	22.665	22.994	22.386
	SSIM	**0.817**	0.809	0.799	0.798	0.808
Moyennes	PSNR	23.641	**23.816**	23.279	23.711	22.679
	SSIM	**0.786**	0.781	0.780	0.779	0.783

FIGURE C.1 – Résultats des PSNR et SSIM calculés sur les contours d'image naturelles.

Bibliographie

[Albanesi 1997] M.G. Albanesi. *Thresholding wavelets for image compression.* pages 374 –389, jun. 1997.

[Algazi 1991] V.R. Algazi, G.E. Ford et R. Potharlanka. *Directional interpolation of images based on visual properties and rank order filtering.* In International Conference on Acoustics, Speech, and Signal Processing, pages 3005 –3008 vol.4, 14-17 1991.

[Allebach 1996] J. Allebach et Ping Wah Wong. *Edge-directed interpolation.* International Conference on Image Processing, vol. 3, pages 707–710 vol.3, Sep 1996.

[Aly 2003] E. Aly H. ; Dubois. *Regularized image up-sampling using a new observation model and the level set method.* International Conference on Image Processing, vol. 3, pages III–665–8 vol.2, 2003.

[Aly 2005] E. Aly H.A. ; Dubois. *Image up-sampling using total-variation regularization with a new observation model.* IEEE Transactions on Image Processing, vol. 14, no. 10, pages 1647–1659, 2005.

[Ballester 2001] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro et J. Verdera. *Filling-in by joint interpolation of vector fields and gray levels.* Image Processing, IEEE Transactions on, vol. 10, no. 8, pages 1200–1211, Aug 2001.

[Bayrakeri 1995] S.D. Bayrakeri et R.M. Mersereau. *A new method for directional image interpolation.* International Conference on Acoustics, Speech, and Signal Processing, vol. 4, pages 2383–2386 vol.4, 1995.

[Beylkin 1987] G Beylkin. *Discrete Radon Transform.* IEEE Transactions on acoustics, speech and signal processing, vol. 35-2, pages 162–172, 1987.

[Biancardi 2002] A. Biancardi, L. Cinque et L. Lombardi. *Improvements to image magnification.* Pattern Recognition, vol. 35, no. 3, pages 677 – 687, 2002.

[Bigun 1992] J. Bigun et J.M.jia Hans du Buf. *Geometric image primitives by complex moments in Gabor space and the application to texture segmentation.* In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 648 –650, 15-18 1992.

[Boor 2001] Carl De Boor. A practical guide to splines. Applied Mathematics Science, 2001.

[BT.500-12 2009] ITU-R BT.500-12. *Methodology for the subjective assessment of the quality of television pictures*, Sep 2009.

[Burge 2009] W. Burger ; Mark J. Burge. Principles of digital image processing : core algorithms. Springer, 2009.

[Burt 1983] P. Burt et E. Adelson. *The Laplacian Pyramid as a Compact Image Code.* IEEE Transactions on Communications, vol. 31, no. 4, pages 532 – 540, apr. 1983.

[Carrato 1996] S. Carrato, G. Ramponi et S. Marsi. *A simple edge-sensitive image interpolation filter.* International Conference on Image Processing, vol. 3, pages 711–714 vol.3, Sep 1996.

[Chang 2000] S.G. Chang, Bin Yu et M. Vetterli. *Adaptive wavelet thresholding for image denoising and compression.* IEEE Transactions on Image Processing, vol. 9, no. 9, pages 1532 –1546, sep. 2000.

[Chang 2006] Chuo-Ling Chang et B. Girod. *Direction-Adaptive Discrete Wavelet Transform via Directional Lifting and Bandeletization.* IEEE International Conference on Image Processing, vol. -, pages 1149–1152, 2006.

[Chang 2007] C.-L. Chang et B. Girod. *Direction-Adaptive Discrete Wavelet Transform for Image Compression.* IEEE Transactions on Image Processing, vol. 16, no. 5, pages 1289–1302, 2007.

[Chappelier 2006] V. Chappelier et C. Guillemot. *Oriented Wavelet Transform for Image Compression and Denoising.* IEEE Transactions on Image Processing, vol. 15, no. 10, pages 2892–2903, Oct. 2006.

[Crowe 2000] Anatole Beck ; Michael N. Bleicher ; Donald W. Crowe. Excursions into mathematics : The millennium edition. AK Peters, Ltd., 2000.

[Da Costa 2002] J.P. Da Costa, C. Germain et P. Baylou. *Orientation difference statistics for texture description.* In 16th International Conference on Pattern Recognition, volume 1, pages 652 – 655 vol.1, 2002.

[Daugman 1988] John G. Daugman. *Complete Discrete 2-D Gabor Transforms by Neural Networks for Image Analysis and Compression.* IEEE Transactions on acoustics, speech ans signal processing, vol. 36, pages 1169–1179, 1988.

[Deans 1983] S. R. Deans. The radon transform and some of its applications. Wiley, 1983.

[Do 2003] Do. *The finite ridgelet transform for image representation.* IEEE Transactions on image processing, vol. 12, pages 16–28, 2003.

[Do 2005] Do. *The contourlet transform : an efficient directional multiresolution image representation.* IEEE Transactions on image processing, vol. 14, pages 2091–2106, 2005.

[Do 2006a] Do. *Directional multiscale modeling of images using the contourlet transform.* IEEE Transactions on image processing, vol. 15, pages 1610–1620, 2006.

[Do 2006b] Do. *The nonsubsampled contourlet transform : Theory, design, and applications.* IEEE Transactions on image processing, vol. 15, pages 3089–3101, 2006.

[Do 2007] Do. *Multidimensional directional filter banks and surfacelets.* IEEE Transactions on image processing, vol. 16, pages 918–931, 2007.

[Donoho 1994] D.L. Donoho et I.M. Johnstone. *Threshold selection for wavelet shrinkage of noisy data.* In IEEE Transactions on Information Theory, pages A24 –A25 vol.1, nov. 1994.

[Donoho 1995] D.L. Donoho. *De-noising by soft-thresholding.* IEEE Transactions on Information Theory, vol. 41, no. 3, pages 613 –627, may. 1995.

[Duchon 1979] C.E. Duchon. *Lanczos filtering in one and two dimensions.* Journal of Applied Meteorology, vol. 18, pages 1016–1023, 1979.

[Feldman 1969] Jerome A. Feldman, C. M. Feldman, Gilbert Falk, C. Grape, J. Pearlman, Irwin Sobel et Jay M. Tenenbaum. *The Stanford Hand-Eye Project.* In IJCAI, pages 521–526, 1969.

[Freeman 1991] W.T. Freeman et E.H. Adelson. *The design and use of steerable filters.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 9, pages 891 –906, sep 1991.

[Gabor 1946] D. Gabor. *Theory of communication.* J . Inst. Elec. Eng ., vol. 93, pages 429–457, 1946.

[Grgic 2001] S. Grgic, M. Grgic et B. Zovko-Cihlar. *Performance analysis of image compression using wavelets.* IEEE Transactions on Industrial Electronics, vol. 48, no. 3, pages 682 –695, jun. 2001.

[Grossmann 1984] A. Grossmann et J. Morel. *Decomposition of Haardy functions into square integrable wavelets of constant shape.* SIAM, vol. 15(4), pages 723–736, 1984.

[Guy 1995] John H. Conway ; Richard Guy. The book of numbers. Springer, 1995.

[Hou 1978] Hsieh Hou et H. Andrews. *Cubic splines for image interpolation and digital filtering.* IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 26, no. 6, pages 508 – 517, dec. 1978.

[Jafari Khouzani 2005] H. Jafari Khouzani K.and Soltanian Zadeh. *Radon transform orientation estimation for rotation invariant texture analysis.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, pages 1004–1008, 2005.

[Jayant 1984] N.S. Jayant et P. Noll. Digital coding of waveforms : Principles and applications to speech and video. Prentice Hall, 1984.

[Jensen 1995] K. Jensen et D. Anastassiou. *Subpixel edge localization and the interpolation of still images.* IEEE Transactions on Image Processing, vol. 4, no. 3, pages 285 –295, mar 1995.

[Jiang 2002] Hao Jiang et C. Moloney. *A new direction adaptive scheme for image interpolation.* ICIP, vol. 3, pages III–369 – III–372 vol.3, 2002.

[Keys 1981] R. Keys. *Cubic convolution interpolation for digital image processing.* IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 29, no. 6, pages 1153–1160, Dec 1981.

[Lanczos 1989] C. Lanczos. Applied analysis. Dover Publications Inc., 1989.

[Lehmann 1999] T.M. Lehmann, C. Gonner et K. Spitzer. *Survey : interpolation methods in medical image processing.* IEEE Transactions on Medical Imaging, vol. 18, no. 11, pages 1049 –1075, 1999.

[LePennec 2002] LePennec. *Bandelettes et representation geometrique des images.* PhD thesis, POLYTECHNIQUE, 2002.

[LePennec 2005] LePennec. *Bandelet Image Approximation and Compression.* 2005.

[LePennec 2006] LePennec. *Compression d'image.* 2006.

[LePennec 2007] LePennec. *Debruitage geometrique d'images dans des bases orthonormees de bandelettes.* 2007.

[Li 2001] Xin Li et M.T. Orchard. *New edge-directed interpolation.* ICIP, vol. 10, no. 10, pages 1521–1527, 2001.

[Lindeberg 1996] T. Lindeberg. *Edge detection and ridge detection with automatic scale selection.* In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 465 –470, 18-20 1996.

[Luisier 2007] F. Luisier, T. Blu et M. Unser. *A New SURE Approach to Image Denoising : Interscale Orthonormal Wavelet Thresholding.* IEEE Transactions on Image Processing, vol. 16, no. 3, pages 593 –606, mar. 2007.

[Maalouf 2006] Aldo Maalouf. *Coopération des ondelettes et des équations aux dérivées partielles pour le traitement d'images multispectrales.* PhD thesis, Univ. Poitiers, 2006.

[Mallat 1989] S.G. Mallat. *A theory for multiresolution signal decomposition : the wavelet representation.* Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 7, pages 674–693, 1989.

[Mallat 1999] S. Mallat. A wavelet tour of signal processing. Academic Press, 1999.

[Mallat 2005] E. LePennec ; S. Mallat. *Sparse Geometric Image Representations With Bandelets.* IEEE Transactions on image processing, vol. 14, page All pages, 2005.

[Mallat 2007] S. Mallat et G. Peyre. *A Review of Bandlet Methods for Geometrical Image Representation.* Numerical Algorithms, vol. 44(3), pages p. 205–234, 2007.

[Meyer 1992] Y. Meyer. Wavelets and operators. Advanced Mathematics, 1992.

[Meyer 1996] F.G. Meyer et R.R. Coifman. *Adaptive directional image compression with oriented wavelets*. vol. 1, pages 601–604 vol.1, Sep 1996.

[Michelet 2004] F. Michelet, C. Germain, P. Baylou et J.P. da Costa. *Local multiple orientation estimation : isotropic and recursive oriented network*. In 17th International Conference on Pattern Recognition, volume 1, pages 712 – 715 Vol.1, 23-26 2004.

[Michelet 2007] Franck Michelet, Jean-Pierre Da Costa, Olivier Lavialle, Yannick Berthoumieu, Pierre Baylou et Christian Germain. *Estimating local multiple orientations*. Signal Processing, vol. 87, pages 1655–1669, 2007.

[Montanvert 1991] Jean-Marc Chassery ; Annick Montanvert. Géometrie discrete en analyse d'images. Traité des nouvelles technologies série Images, 1991.

[Morse 2001] B.S. Morse et D. Schwartzwald. *Image magnification using level-set reconstruction*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pages I–333–I–340 vol.1, 2001.

[Muresan 2005] D. Muresan. *Fast edge directed polynomial interpolation*. ICIP, vol. 2, pages II–990–993, 2005.

[Patashnik 1994] Donald E. Knuth ; Ronald Graham ; Oren Patashnik. Concrete mathematics : A foundation for computer science (2nd edition). Addison-Wesley Professional, 1994.

[Perona 1998] P. Perona. *Orientation diffusions*. IEEE Transactions on Image Processing, vol. 7, no. 3, pages 457–467, 1998.

[Peyre 2005a] Peyre. *Géométrie multi-échelles pour les images et les textures*. PhD thesis, Ecole Polytechnique, 2005.

[Peyre 2005b] G. Peyre et S. Mallat. *Discrete bandelets with geometric orthogonal filters*. ICIP, vol. 1, pages I – 65–8, sep. 2005.

[Peyre 2007] G. Peyre. *Bandlet Toolbox, www.mathworks.com/matlabcentral/fileexchange*, 2007.

[Rahman 2008] S.M.M. Rahman, M.O. Ahmad et M.N.S. Swamy. *Bayesian Wavelet-Based Image Denoising Using the Gauss Hermite Expansion*. IEEE Transactions on Image Processing, vol. 17, no. 10, pages 1755 –1771, oct. 2008.

[Ramesh 1989] BR Ramesh, N Srinivasa et K Rajgopal. *An Algorithm for Computing the Discrete Radon Transform With Some Applications*. In EEE Region 10 International Conference,TENCON, 1989.

[Reveilles 1991] Jean-Pierre Reveilles. *Geometrie discrete, Calcul en nombres entiers et algorithmique*. PhD thesis, Universite Louis Pasteur, Strasbourg, France, 1991.

[Sivignon 2004] Isabelle Sivignon. *De la caractérisation des primitives a la reconstruction polyédrique de surfaces en géométrie discrete*. PhD thesis, INPG, 2004.

[Starck 2002] J.L. Starck, E.J. Candes et D.L. Donoho. *The curvelet transform for image denoising*. IEEE Transactions on image processing, vol. 11, no. 6, pages 670–684, 2002.

[Starck 2007] J.L. Starck, J. Fadili et F. Murtagh. *The Undecimated Wavelet Decomposition and its Reconstruction*. ICIP, vol. 16, no. 2, pages 297–309, Feb. 2007.

[Strohmer 1998] Hans G. Feichtinger ; Thomas Strohmer. Gabor analysis and algorithms : theory and applications. BirkHäuser, 1998.

[Taubman 1994] D. Taubman et A. Zakhor. *Orientation adaptive subband coding of images*. IEEE Transactions on Image Processing, vol. 3, no. 4, pages 421–437, 1994.

[Unser 1991] M. Unser, A. Aldroubi et M. Eden. *Fast B-spline transforms for continuous image representation and interpolation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 3, pages 277 –285, mar. 1991.

[Unser 1993] M. Unser, A. Aldroubi et M. Eden. *B-spline signal processing. I. Theory*. IEEE Transactions on Signal Processing, vol. 41, no. 2, pages 821 –833, Février 1993.

[Unser 1999] M. Unser. *Splines : a perfect fit for signal and image processing.* Signal Processing Magazine, IEEE, vol. 16, no. 6, pages 22–38, Nov 1999.

[Van Reeth 2010a] E. Van Reeth, P. Bertolino et M. Nicolas. *Procédé de détection d'orientation de contours, Brevet B09-4817FR*, 2010.

[Van Reeth 2010b] E. Van Reeth, P. Bertolino, M. Nicolas et J.M. Chassery. *Adaptive edge orientation analysis.* In IS&T/SPIE Electronic Imaging, 2010.

[Van Reeth 2011a] E. Van Reeth, P. Bertolino et M. Nicolas. *Agrandissement d'image basé sur une carte directionnelle multirésolution.* GRETSI, 2011.

[Van Reeth 2011b] E. Van Reeth, P. Bertolino et M. Nicolas. *Image interpolation based on a multi-resolution directional map.* In IS&T/SPIE Electronic Imaging, 2011.

[Velisavljevic 2003] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli et P.L. Dragotti. *Discrete multidirectional wavelet bases.* In International Conference on Image Processing, volume 1, pages I–1025–8 vol.1, Sept. 2003.

[Velisavljevic 2006] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli et P.L. Dragotti. *Directionlets : anisotropic multidirectional representation with separable filtering.* IEEE Transactions on Image Processing, vol. 15, no. 7, pages 1916–1933, 2006.

[Villasenor 1995] J.D. Villasenor, B. Belzer et J. Liao. *Wavelet filter evaluation for image compression.* IEEE Transactions on Image Processing, vol. 4, no. 8, pages 1053 –1060, aug. 1995.

[Wang 2004] Z. Wang, A. C. Bovik, H. R. Sheikh et E. P. Simoncelli. *Image quality assessment : From error visibility to structural similarity.* IEEE Transactions on Image Processing, vol. 13, pages 600–612, 2004.

[Wang 2007] Qing Wang et R.K. Ward. *A New Orientation-Adaptive Interpolation Method.* ICIP, vol. 16, no. 4, pages 889 –900, april 2007.

[Xiong 2009] Zhiwei Xiong, Yonghua Zhang, Xiaoyan Sun et Feng Wu. *Fast directional image interpolation with difference projection.* In Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on, pages 81 –84, june 2009.

[Yeung 1997] E. Yeung. *Image compression using wavelets.* volume 1, pages 241 –244 vol.1, may. 1997.

[Zhang 2006] Lei Zhang et Le Nan Xiaolin Wu. *"An edge-guided image interpolation algorithm via directional filtering and data fusion,".* IEEE Transactions on Image Processing, vol. 15, pages pp. 2226–2238, 2006.

Résumé

L'arrivée massive d'écrans LCD dits de haute-définition sur le marché, entraîne un besoin acru d'algorithmes d'augmentation de résolution pour l'affichage d'images ou de vidéos dont la résolution est inférieure à celle de l'écran.
Nous proposons un schéma novateur d'interpolation d'images, basée sur une analyse multirésolution de la direction des contours. Le but de cette approche est de corriger les artefacts classiques d'interpolation qui apparaissent lorsque des méthodes habituelles sont utilisées (bilinéaire, bicubique), tout en évitant l'apparition des artefacts engendrés par la plupart des méthodes d'interpolation directionnelle. Notre estimation d'orientation de contours, basée sur une division de l'image originale en quadtree et une étude fréquentielle des contours est comparée à deux méthodes faisant référence dans l'état de l'art (transformée de Radon et algorithme de projection utilisé pour la création des bandelettes). Cette comparaison permet d'étudier les comportements de chaque méthode en vue d'une application à des images naturelles.
Par la suite, l'interpolation en elle-même est introduite. Elle est basée sur l'utilisation d'un noyau d'interpolation isotrope (cubic-spline), qui est corrigée grâce à un filtrage Gaussien localement orienté dans la direction des contours. Les régions ne contenant pas de contour sont préservées grâce à la création d'un masque construit à partir de filtres de Gabor. Enfin, les résultats de notre interpolation sont comparés à des méthodes d'interpolation directionnelle récentes, afin d'illustrer les bonnes performances de notre algorithme sur des images naturelles de natures variées.

Abstract

The recent success of high definition screens has increased the need of interpolation algorithms, to display images or videos which resolution is smaller than the screen resolution.
We propose a new image interpolation process, based on a multiresolution edge orientation analysis. The goal of this technique is to correct usual artifacts that appear on edges when classical interpolation methods are used (bilinear, bicubic), without introducing new artifacts that are often produced by directional interpolations. Our orientation estimation, based on a quadtree division and a multiresolution approach is evaluated and compared to two other state-of-the-art methods (Radon transform, and the projection method used in the Bandlet transform algorithm), to study its advantages in the context of an application to natural images.
Then, we introduce our interpolation technique, based on an isotropic reference interpolation (cubic-spline) that is corrected by a two-dimension Gaussian filter, locally oriented in the direction of the edge. Edge-free regions are preserved with a Gabor mask that is built to protect pixels which do not need any correction. Finally, our results are compared to recent state-of-the-art directional interpolations to illustrate the good performance of our algorithm on various contents of natural images.

Mot-clefs

Interpolation, Analyse directionnelle, Analyse multirésolution, Filtres de Gauss, Filtres de Gabor

Titre

Système avancé d'interpolation spatiale de signaux de télévision pour affichage sur écrans haute-définition

Keywords

Upscaling, Directional Analysis, Multiresolution Analysis, Gauss Filtering, Gabor Filtering

Adrr : Gipsa-Lab, 961 rue de la Houille Blanche BP 46 F - 38402 GRENOBLE Cedex
ISBN :