

**THÈSE DE DOCTORAT DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE  
FRANCHE-COMTÉ**

**PRÉPARÉE À L'UNIVERSITÉ DE BOURGOGNE**

École doctorale n°37

Sciences Pour l'Ingénieur et Microtechniques

Doctorat d'Instrumentation et Informatique de l'Image

par

**YOAN MARIN**

**Développement de nouvelles techniques de Super résolution de séquences vidéo:  
Vers une implémentation temps réel sur Smart Caméra  
Projet EXIST**

Thèse présentée et soutenue à Dijon, le 04 Décembre 2020

Composition du Jury :

MÉNARD DANIEL	Professeur à l'Université de Rennes	Président du jury
BERRY FRANÇOIS	Professeur à l'Université Clermont-Auvergne	Examinateur
DUBOIS JULIEN	Maître de conférences HDR à l'Université de Bourgogne	Examinateur
ATRI MOHAMED	Professeur à King Khalid University	Rapporteur
MORAIN-NICOLIER FRÉDÉRIC	Professeur à l'Université de Reims Champagne-Ardenne	Rapporteur
GINHAC DOMINIQUE	Professeur à l'Université de Bourgogne	Directeur de thèse
MITÉРАН JOHEL	Professeur à l'Université de Bourgogne	Co-directeur de thèse
HEYRMAN BARTHÉLÉMY	Maître de conférences à l'Université de Bourgogne	Co-encadrant de thèse





**Titre :** Développement de nouvelles techniques de Super résolution de séquences vidéo: Vers une implémentation temps réel sur Smart Caméra

**Mots-clés :** Amélioration de la qualité des images, FPGA, Super Résolution, Analyse de texture en temps-réel, Interpolation spatiale, Implantation en temps réel, métrique de la qualité de l'image, déconvolution temps-réel.

**Résumé :**

Ces travaux de thèse s'inscrivent dans le contexte d'un projet européen visant à créer une caméra très haute résolution 8k. Au sein de ce projet notre équipe a eu la tâche de travailler sur deux aspects technologiques : (1) la création d'un démonstrateur réalisant une déconvolution en temps réel d'un flux vidéo provenant d'une caméra très haute résolution créée par le consortium, (2) la création d'un prototype permettant d'augmenter la résolution et le niveau de détail de flux vidéo d'une résolution d'entrée de 4k vers le 8k en utilisant des méthodes de Super Résolution (SR). Ce manuscrit présente principalement les travaux liés à la création du prototype réalisant une méthode de SR. Afin de pouvoir évaluer l'apport qualitatif de la SR, une étude sur la pertinence des métriques est réalisée afin de sélectionner les métriques les plus adaptées à évaluer des traitements d'amélioration de la qualité des images. Ensuite, une nouvelle méthode de SR Spatiale nommée LASSR pour *Local Adaptive Spatial Super Resolution* a été mise en place pour corriger certains artefacts observés lors de l'implantation d'une méthode de SR de l'état de l'art. Notre méthode permet d'adapter localement le traitement en fonction de la texture,

cette adaptation est réalisée automatiquement en utilisant un modèle de décision généré lors d'un apprentissage supervisé. Suite à la validation de notre méthode par un panel d'experts lors d'une évaluation psychovisuelle, le module IP a été développé pour fonctionner avec des volumes de données très importants sur des cibles FPGA et pour fonctionner en temps réel. Une *boîte à outils* de composants a été créée afin de pouvoir décrire le module de manière générique. Cela permet d'adapter le module LASSR à différentes configurations (taille de l'image, dynamique des pixels ...). L'élément majeure de cette *boîte à outil* est le filtre de convolution 2-D générique : sa taille et ses coefficients sont paramétrables (la puce FPGA peut limiter ce filtre). Des résultats encourageants ont été obtenus concernant l'augmentation de résolution pour une image de sortie 4k en niveaux de gris, il est possible pour un seul module LASSR d'obtenir une fréquence de sortie de 17.03 im/s. Sur le FPGA de la carte de développement VC707 de Xilinx il est théoriquement possible d'atteindre une fréquence de 136.2 im/s en parallélisant 8 modules LASSR.

**Title:** Développement de nouvelles techniques de Super résolution de séquences vidéo: Vers une implémentation temps réel sur Smart Caméra

**Keywords:** Image quality enhancement, FPGA, Super resolution, real-time texture analysis, spatial interpolation, real-time implementation, image quality assesment, real time deconvolution

**Abstract:**

These thesis works are part of an european project aiming to design a very high resolution (8k) video camera. Within this project our team had the task of working on two technological aspects: (1) the design of a demonstrator carrying out a real-time deconvolution of a video stream coming from a very high resolution camera created by the consortium , (2) the design of a prototype allowing to increase the resolution and the level of detail of video streams from an input resolution of 4k to 8k using Super Resolution (SR) methods. This manuscript mainly presents the work related to the creation of the prototype realizing a Super Resolution method. In order to be able to assess the qualitative contribution of SR, a study on the relevance of the metrics is carried out in order to select the most suitable metrics for evaluating image quality improvement processes. Then, a new method of Spatial SR named LASSR for *Local Adaptive Spatial Super Resolution* was implemented to correct certain artefacts observed with a State of the Art SR method. Our method allows to locally adapt the processing according

the texture, this adaptation is carried out automatically using a decision model generated during supervised learning. Following the validation of our method by a panel of experts during a psychovisual assesment, the IP module was developed to operate with very large volumes of data on FPGA targets and to operate in real time. A *toolbox* of components has been created in order to describe the module in a generic way. This allows the LASSR module to be adapted to different configurations (image size, pixel dynamics ...). The major element of this *toolbox* is the generic 2-D convolution filter : its size and its coefficients can be tuned (the FPGA chip limits this filter). Encouraging results have been obtained regarding the increase in resolution for a 4k grayscale output image, it is possible for a single LASSR module to achieve an output frequency of 17.03 fps. On the FPGA of the Xilinx VC707 development board, it is theoretically possible to reach a frequency of 136.2 fps by parallelizing 8 LASSR modules.



# REMERCIEMENTS

Je remercie tout d'abord mon directeur de thèse, Dominique GINHAC, qui par son travail a pu obtenir ce financement de thèse au travers du projet européen Horizon 2020 - *EXIST Extended Image Sensing Technologies* et pour la confiance qu'il a eue envers moi.

Je remercie ensuite mon co-directeur et mon co-encadrant, Johel Mitéran et Barthélémy Heyrman, pour leur patience, leur aide et leurs conseils durant ces années. De même, je remercie Julien Dubois, co-encadrant officieux, qui a toujours été présent tout au long de ces années, qui a pu fournir de bons conseils aussi bien scientifiques que personnels et qui a pu faire preuve d'une grande humanité. Je vous remercie pour votre implication dans la relecture des articles et de ce manuscrit.

Je tiens à remercier Messieurs Mohamed Atri et Frédéric Morain-Nicolier d'avoir accepté de rapporter mon travail de doctorat. Mes remerciements vont aussi à Messieurs François Berry et Daniel Ménard d'avoir pris du temps pour examiner mes travaux.

Je remercie les membres du laboratoire ImViA (Le2i), enseignants-chercheurs et les services techniques, qui m'ont permis de part leur gentillesse et leur bonne humeur de réaliser cette thèse dans de bonnes conditions. Même si il est vrai que les rires qui traversent les couloirs peuvent parfois nous déconcentrer. Je remercie ensuite les équipes administratives successives du laboratoire qui ont permis que l'organisation des missions puissent avoir lieu dans de bonnes conditions.

J'apporte toute ma gratitude aux différentes équipes pédagogiques de l'ESIREM et du département IEM de l'UFR Sciences et Techniques qui m'ont apporté leur confiance et permis de découvrir l'enseignement lors de ces dernières années.

Nostalgique, je remercie les doctorants des débuts, les doctorants avec qui nous nous sommes approprié ces nouveaux locaux et partagé des moments importants : Nicolas, Mustapha, Axel, Serge, Pierre, Anthony, Richard, Roberto mais aussi Ali, Yangjing et Romain.

Je remercie les membres de ma famille qui, même si pour eux les termes "*thèse*" et "*doctorat*" sont abstraits, me soutiennent et sont fiers de moi. De plus, je remercie mon père qui m'aide sans défaillance à réaliser un tout autre projet en parallèle de ces travaux de thèse.

Enfin, je remercie sincèrement ma compagne Préscillia avec qui nous traversons ensemble les épreuves, qui me supporte depuis tant d'années, me soutient dans mes choix et m'encourage à vite terminer ce travail. Je la remercie d'autant plus qu'elle nous a permis de créer notre propre famille en me faisant père. Merci à Elena et Diego de supporter ces jours où je tape sur mon clavier au lieu d'être avec vous. Vous me montrez ce qui est réellement important, je vous aime.



# SOMMAIRE

<b>1</b>	<b>Introduction Générale</b>	<b>1</b>
1.1	Motivations et contexte de l'étude . . . . .	2
1.1.1	La course aux pixels . . . . .	2
1.1.2	Insertion dans le projet Européen H2020-EXIST . . . . .	4
1.2	Objectifs de la thèse . . . . .	5
1.3	Structure du manuscrit . . . . .	6
<b>2</b>	<b>État de l'art</b>	<b>7</b>
2.1	Pertes et dégradations : les éléments bloquants . . . . .	8
2.1.1	La prise de vue . . . . .	9
2.1.2	Le système d'acquisition . . . . .	12
2.1.2.1	Les dégradations optiques . . . . .	12
2.1.2.2	Les dégradations électroniques . . . . .	14
2.1.3	La transmission, le stockage et l'affichage . . . . .	17
2.1.4	Discussion . . . . .	19
2.2	Les métriques pour l'évaluation des images . . . . .	21
2.2.1	Les métriques pleine-référence . . . . .	21
2.2.2	Les métriques sans-référence . . . . .	26
2.2.3	Discussion . . . . .	30
2.3	L'imagerie computationnelle pour l'amélioration de la qualité de l'image . . . . .	30
2.3.1	La déconvolution, une technique de défloutage . . . . .	33
2.3.1.1	Déconvolution non-aveugle . . . . .	35
2.3.1.2	Déconvolution aveugle . . . . .	38
2.3.1.3	Discussion et orientation . . . . .	40
2.3.2	Les techniques de super résolution . . . . .	40
2.3.2.1	Multiple Image Super Resolution <i>MISR</i> . . . . .	42
2.3.2.2	Simple Image Super Résolution <i>SISR</i> . . . . .	47
2.3.2.3	Discussion et orientations . . . . .	55
2.4	Discussion et Conclusion . . . . .	56

<b>3</b>	<b>Étude et développement</b>	<b>59</b>
3.1	Évaluation de la pertinence des métriques de la qualité de l'image . . . . .	60
3.1.1	La base d'images . . . . .	60
3.1.2	Les traitements appliqués aux images . . . . .	60
3.1.3	Classement des traitements par les métriques sélectionnées . . . . .	61
3.1.3.1	Métriques sans-référence . . . . .	62
3.1.3.2	Métriques pleine-référence . . . . .	63
3.1.4	Discussion et proposition . . . . .	64
3.2	Évaluation de la méthode de Gohshi . . . . .	66
3.2.1	Présentation complète . . . . .	66
3.2.1.1	Espace de couleur de travail . . . . .	66
3.2.1.2	Interpolation . . . . .	67
3.2.1.3	Filtre passe Haut <i>HPPF</i> . . . . .	68
3.2.1.4	Fonction Non Linéaire <i>NLF</i> . . . . .	69
3.2.1.5	Limitation & Fusion . . . . .	72
3.2.2	Analyses critiques . . . . .	73
3.2.2.1	Analyse visuelle . . . . .	74
3.2.2.2	Tests approfondis . . . . .	74
3.3	Solution proposée : LASSR . . . . .	76
3.3.1	Classification de Texture . . . . .	76
3.3.1.1	Méthodes de classification . . . . .	77
3.3.1.2	Données d'entraînement . . . . .	80
3.3.1.3	Extraction d'attributs pour l'analyse de texture . . . . .	81
3.3.1.4	Performance de classification et sélection d'attributs . . . . .	81
3.3.2	Fusion . . . . .	84
3.4	Évaluation de la méthode LASSR . . . . .	85
3.4.1	Critique visuelle de la méthode LASSR . . . . .	85
3.4.2	Résultats des métriques conventionnelles . . . . .	86
3.4.3	Mise en place de l'évaluation psychovisuelle <i>PVA<sub>2</sub></i> . . . . .	86
3.4.4	Conclusion . . . . .	90
<b>4</b>	<b>Accélérateur matériel</b>	<b>93</b>
4.1	Mise en œuvre d'un prototype intégrant la méthode LASSR . . . . .	94
4.1.1	Méthodologie de développement et validation . . . . .	94
4.1.1.1	Méthodologie de développement du modèle de décision . . . . .	95

4.1.1.2	Méthodologie de développement du prototype logiciel . . . . .	96
4.1.1.3	Méthodologie de développement matériel . . . . .	96
4.1.2	Développement matériel . . . . .	99
4.1.2.1	Filtre générique . . . . .	100
4.1.2.2	Méthode de Ghoshi . . . . .	103
4.1.2.3	Analyse de texture . . . . .	105
4.1.2.4	Fusion Locale . . . . .	109
4.1.3	Résultats d'implantation et discussion . . . . .	110
4.1.3.1	Résultats . . . . .	110
4.1.3.2	Implantations matérielles sur puces Artix-7 et Spartan-7 . . . . .	111
4.1.3.3	Mise en parallèle de modules LASSR . . . . .	112
4.1.3.4	Comparaison avec l'état de l'art . . . . .	116
4.2	Déconvolution temps réel - GPU . . . . .	118
4.2.1	Méthode . . . . .	119
4.2.1.1	Gestions des PSFs . . . . .	119
4.2.1.2	Application du filtre de Wiener . . . . .	120
4.2.2	Résultats . . . . .	121
4.2.2.1	Résultats images préchargées et image fixe provenant de la caméra adimec . . . . .	122
4.2.2.2	Résultats caméra Adimec . . . . .	123
4.2.3	Discussion . . . . .	125
<b>5</b>	<b>Conclusion et Perspectives</b>	<b>129</b>
5.1	Super Résolution . . . . .	129
5.2	Étude sur la pertinence des métriques . . . . .	131
5.3	Déconvolution temps réel . . . . .	133
5.4	Valorisations scientifiques . . . . .	133
<b>I</b>	<b>Annexes</b>	<b>155</b>
<b>A</b>	<b>Les séparateurs à vaste marge</b>	<b>157</b>
A.1	Cas linéairement séparables - Séparateurs linéaires . . . . .	157
A.2	Cas non linéairement séparables - Séparateurs linéaires . . . . .	159
A.3	Séparateurs non linéaires . . . . .	159
<b>B</b>	<b>Génération du modèle de décision</b>	<b>161</b>

B.1	Fichiers Rom . . . . .	161
B.1.1	Fichiers C++ . . . . .	162
B.1.2	Transcodage et directives . . . . .	165



# INTRODUCTION GÉNÉRALE

**C**e chapitre introductif présente le contexte général de ce travail intitulé "Développement de nouvelles techniques de Super résolution de séquences vidéo : Vers une implémentation temps réel sur Smart Caméra". Le Laboratoire ImViA a en effet participé de 2015 à 2018 à un projet Européen H2020 sur le thème de la très haute résolution 8k ( $8192 \times 4096$  pixels), projet qui nous a permis d'apporter des contributions dans les techniques de Super Résolution et dans le domaine matériel via l'introduction d'un accélérateur matériel compétitif avec l'état de l'art.

## Sommaire

---

<b>1.1</b>	<b>Motivations et contexte de l'étude</b>	<b>2</b>
1.1.1	La course aux pixels	2
1.1.2	Insertion dans le projet Européen H2020-EXIST	4
<b>1.2</b>	<b>Objectifs de la thèse</b>	<b>5</b>
<b>1.3</b>	<b>Structure du manuscrit</b>	<b>6</b>

---

## 1.1/ MOTIVATIONS ET CONTEXTE DE L'ÉTUDE

### 1.1.1/ LA COURSE AUX PIXELS . . .

La télévision a largement et constamment évolué depuis sa création. Tout d'abord en noir et blanc, les images sont passées en couleurs avec une très basse résolution. Ainsi les téléviseurs à tubes cathodiques disposaient pour la plupart de la résolution maximum de diffusion que nous appellerions aujourd'hui "**Small Definition**" (SD) avec une résolution de  $800 \times 500$  pixels. Les connections "péritel" qui reliaient les décodeurs satellites, les magnétoscopes et même par la suite les lecteurs DVD ne pouvaient d'ailleurs pas diffuser de contenu de plus haute définition.

Dans les années 1990-2000, les téléviseurs à tubes cathodiques ont laissé la place aux téléviseurs à base de cristaux liquides. Ces nouveaux écrans, disposant de technologie numérique, imposent les premiers standards Haute Définition (HD-720p) avec une résolution de  $1280 \times 720$  pixels. En 2006 ce standard évolue pour apporter encore plus de détails aux images avec le Full HD ( $1920 \times 1080$  pixels). Cette norme s'est généralisée depuis 2013 pour les écrans d'ordinateur et les smartphones.

En 2012 la norme *Ultra HD*, avec une résolution de  $3840 \times 2160$  pixels, s'impose pour remplacer la Full HD. Cette norme de diffusion, appelée 4k (le véritable 4k étant théoriquement réservé à la norme DCI du cinéma, avec une définition de  $4096 \times 2160$  pixels), est rendue possible avec l'évolution constante des technologies de compression vidéo (H.265 HEVC) et l'augmentation du débit des connexions apportant le contenu vidéo (fibre optique, VDSL et satellite).

Bien que l'*Ultra HD* ne soit pas répandue dans tous les foyers au moment où nous écrivons, et que le contenu de diffusion haute définition ne soit pas généralisé, la chaîne nationale de diffusion de télévision Japonaise NHK <sup>1</sup> a annoncé dès 2017 qu'elle diffusera les Jeux Olympiques du Japon de 2020 en 8k ( $7680 \times 4320$  pixels). Les scènes seront capturées à une fréquence de 240 images/secondes avant d'être traitées et de permettre la diffusion des séquences en *slow motion*. C'est une performance que prévoit cette chaîne de télévision, sachant que pour le moment les téléviseurs et projecteurs cinéma en 8k sont rares et chers. NHK prévoit de développer la chaîne complète, de la capture des images à haute cadence à l'affichage, en passant par le transfert<sup>2</sup> (figure.1.1).

C'est en Septembre 2018, lors du salon international de la radiodiffusion de Berlin (IFA 2018) que de nombreux téléviseurs et projecteurs 8k ont été présentés. Ces écrans de grande résolution et de grande taille sont une révolution dans le domaine du grand public, même si pour le moment les prix restent prohibitifs (au-delà de 6000€ pour le plus petit moniteur Samsung 65").

Des caméras à très haute résolution sont déjà sur le marché, à des prix très élevés. RED, par exemple, vend une large gamme de caméras à haute résolution comme 2K, 4K, 5K, 6K, et 8K. Ces caméras sont plutôt destinées aux professionnels fournissant des flux vidéos peu compressés, ce qui assure une qualité d'image répondant aux standards de la diffusion audiovisuelle.

Cependant, de plus en plus de téléphones incluent des caméras permettant de filmer en 4K avec une qualité telle que des cinéastes comme Claude Lelouch ont pris la décision de tourner certains films à l'aide d'un smartphone <sup>3</sup>.

---

1. [http://www.nhk.or.jp/8k/index\\_e.html](http://www.nhk.or.jp/8k/index_e.html)

2. [http://www.nhk.or.jp/8k/tech/index\\_e.html](http://www.nhk.or.jp/8k/tech/index_e.html)

3. <http://www.premiere.fr/Cinema/News-Cinema/Claude-Lelouch-tournera-son-prochain-film-au-smartphone>

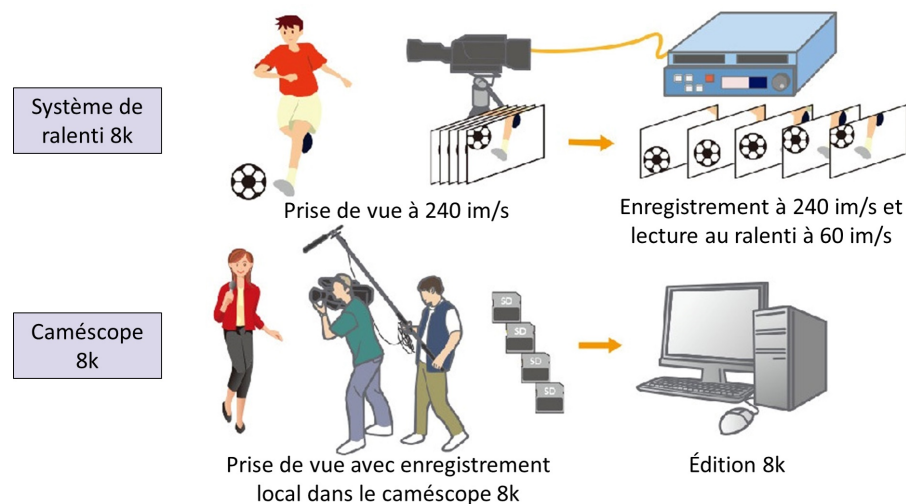


FIGURE 1.1 – NHK prévoit la diffusion des JO 2020 en 8k, @240im/s. Illustration présentant l'utilisation du système de ralenti 8k et du caméscope 8k pour la production de programmes télévisuels.

L'évolution de ces technologies reste possible grâce à l'évolution conjointe des normes de transmission et des techniques de compression. En effet la transmission HDMI 2.1 permet de transférer des images 8k à une fréquence de 60 im/s, compressées avec la compression H265-HEVC.

Nous venons de voir que les constructeurs du domaine télévisuel participent activement à cette course aux pixels, fabriquant des écrans de plus en plus résolus pour tous les types d'afficheurs (téléviseurs, moniteurs d'ordinateurs, téléphone portable, casque de réalité virtuelle, tablettes...). Ces écrans arriveront un jour à surpasser le besoin du consommateur, et surtout surpasser la perception de l'œil humain. Ces constructeurs ont déjà annoncé la norme de diffusion 10k, possible elle aussi grâce aux connexions HDMI 2.1.

Le futur nous dira si cette évolution aura été nécessaire, ou une mode comme il a été possible de le voir avec la limitation de la fabrication des écrans 3D.

### Utilité de la très haute résolution

Il est possible de penser qu'augmenter indéfiniment la résolution des écrans de télévision est superflue, puisque dans la plupart des installations domestiques, le recul de l'observateur est tel que la résolution 4k atteint ou dépasse la résolution de l'œil humain.

Cependant, dans un certain nombre de domaines, il est nécessaire que les images contiennent un nombre de détails très élevés. Nous pouvons citer par exemple :

- Le domaine de la sécurité : un haut niveau de détails peut assurer une bonne qualité de reconnaissance d'individus (y compris si celui-ci est loin de la caméra) et permet d'extraire plus d'informations sur les scènes surveillées.
- Les domaines de la télédétection et de l'astronomie : leur principe étant de capturer des images d'objets situés très loin de la caméra, tout en couvrant un champ large, une grande résolution permet d'analyser plus d'objets plus précisément (étoiles, planètes), ou par exemple d'obtenir des informations de textures plus fiables.
- Le domaine de la médecine : L'augmentation des détails peut conduire à une amélioration des diagnostics, par exemple dans le cadre de l'utilisation d'images endoscopiques. Il est toutefois à noter que dans ce domaine particulier, la création de pixels artificiels (par inter-

polution ou autre) est à réaliser avec précaution étant donné l'impact sur le diagnostic que peut avoir ce type de traitement.

Comme il a été vu jusque-là, l'augmentation de la résolution des images permet d'améliorer le confort visuel, les traitements informatiques, les diagnostics et analyses. Toutefois, l'obtention de ces images haute résolution peut poser des problèmes de coût ou de matériel.

### 1.1.2/ INSERTION DANS LE PROJET EUROPÉEN H2020-EXIST

Tout comme le Japon, l'Union Européenne a initié des projets dédiés à la très haute résolution 8K. Parmi ceux-ci, Le projet H2020 nommé EXIST, pour EXtended Images Sensing Technologies, a pour but de concevoir une caméra complète très haute résolution. Le consortium d'industriels et d'universitaires a pour tâche de concevoir des caméras haute résolution, de l'objectif au traitement embarqué pour piloter ces caméras. Les domaines principaux visés par ce projet sont les suivants :

- La diffusion (broadcasting) ;
- La vidéosurveillance et la sécurité ;
- La santé.

Le consortium réunit un grand nombre d'instituts publics et privés. Ces instituts sont regroupés dans deux catégories :

- Développement de capteurs d'image, et du traitement d'images : *IMEC, Fraunhofer IMS, ImViA (Le2i), TNO, iMinds et TU Delft* ;
- Fabricants/concepteurs de semi-conducteurs et intégrateur finaux : *CMOSIS, Grass Valley, Softkinetic, Thales Angenieux, Adimec, Silios, Sofradir, Focal, et Quest.*

D'une manière générale, il est possible de créer des images haute résolution (*HR*) à l'aide de matériel nativement de haute résolution, grâce à des optiques et des capteurs adaptés. Toutefois, quel que soit le domaine d'application, créer ou renouveler le matériel peut être extrêmement coûteux. C'est notamment le cas des appareils médicaux, ou encore des équipements de satellites. Dans le cadre du projet EXIST, c'est le cas des objectifs et des capteurs. Ainsi, pour limiter les coûts de développement matériel, le projet EXIST a recours à des techniques de Super Résolution (*SR*) qui permettent de créer des images *HR* à l'aide d'algorithmes qui transforment une image basse résolution (*LR*) en image *HR*.

La Super Résolution peut être vue comme la technique idéale dans un grand nombre d'applications, qui dépassent le cadre de ce projet :

- Réduction des coûts de renouvellement : cette technique permet d'augmenter le niveau de détails des caméras sans pour autant effectuer de mise à jour matérielle. La Super Résolution peut être appliquée juste avant l'affichage, ou même avant l'enregistrement des vidéos ou des images.
- Réduction du coût de fabrication : il est possible de diminuer les coûts liés à la fabrication du capteur et de l'objectif de la caméra tout en faisant un compromis entre la résolution matérielle du capteur d'image et le traitement embarqué dans une caméra. Le traitement permet d'obtenir une résolution de sortie plus grande que la résolution de capture, tout en réduisant les coûts des capteurs et des objectifs. Cependant cela augmente le coût de la partie traitement de la caméra.
- Permettre la requalification d'anciens satellites : au lieu de mettre en place une mission spatiale de maintenance pour changer la caméra d'un satellite, il est possible d'améliorer le niveau de détails de cette caméra depuis la terre via des traitements de super résolution.
- Réduire les espaces de stockage nécessaires : Les images très haute résolution 8k néces-

sitent de grands espaces de stockage et une bande passante très importante. Il est possible de réduire les deux en stockant et en transférant les images en basse résolution, et de les analyser et les afficher en haute résolution.

Cette technique suppose une évaluation de la qualité des images obtenues, qui doivent se rapprocher le plus possible de l'image haute résolution qui aurait été obtenue avec du matériel adéquat. De nombreuses métriques ont été étudiées et sont utilisées dans la littérature pour évaluer cette qualité : nous reviendrons sur ce point en détail dans l'état de l'art.

## 1.2/ OBJECTIFS DE LA THÈSE

Le laboratoire ImViA a eu pour tâche, dans le cadre du projet EXIST-H2020, de participer aux études sur la réduction des coûts liés à la fabrication d'objectif et de caméra très haute résolution. Pour cela nous avons eu la tâche de travailler sur deux aspects différents :

- Déconvolution temps réel,
- Super Résolution temps réel.

Le travail sur la déconvolution a pour objectif de trouver un compromis entre la qualité de l'objectif (au sens de la netteté et des aberrations chromatiques) et la complexité de traitements de correction "post-capture" embarqués dans la caméra afin d'avoir des images de bonnes qualité. Ce travail doit conclure sur la réalisation d'un démonstrateur final permettant de corriger les déformations optiques en temps réel sur les images provenant de la caméra haute résolution (8k) créée par les membres du consortium. L'étude de la partie déconvolution a été réalisée essentiellement par Ali DOUIYEK, ingénieur de développement, en étroite collaboration avec mes propres investigations.

D'autre part, le travail sur la Super résolution doit permettre la construction d'images haute résolution à partir d'images basse résolution. L'objectif de la méthode de SR est de fonctionner en temps réel et augmenter la résolution spatiale (ainsi que le niveau de détails) des caméras  $4k-4096 \times 2048$  *pixels* pour créer du  $8k-8192 \times 4096$  *pixels*. Ce traitement doit être implanté dans une cible de type FPGA.

Ce manuscrit traite principalement de la recherche et du développement de techniques de Super Résolution, parmi les méthodes de l'état de l'art, pouvant être embarquées et fonctionner en temps réel. L'objectif principal de cette thèse est de concevoir un prototype matériel intégrant une méthode de SR pour la reconstruction d'images 4k vers 8k à une vitesse minimum de 25 im/s. La difficulté majeure réside dans l'utilisation d'images très haute résolution tout en conservant la notion de temps réel.

Nous avons attaché une grande importance à l'étude de la qualité des images créées. Une étude approfondie des métriques existantes a été réalisée et nous avons établi un protocole d'évaluation rigoureux. Cette étape est nécessaire pour la validation de la méthode de SR.

La déconvolution, élément majeur dans le projet EXIST-H2020, n'est qu'une partie secondaire de ce travail de thèse, le travail de développement a été essentiellement réalisé par Ali Douiyek (ingénieur de recherche). Cependant, ce manuscrit met en avant une partie des recherches et du développement permettant la création du démonstrateur.

Enfin, les principales contributions des travaux présentés dans ce manuscrit sont les suivantes :

- Étude sur la pertinence des métriques de l'image,
- Création d'une méthode originale de Super Résolution Spatiale *LASSR (Local Adaptive Spatial Super Resolution)*,
- Implantation matérielle sur FPGA de la méthode LASSR, et gestion du parallélisme,

- Création d'une *boite à outils* de composants génériques réutilisables dans différents projets de développement (FPGA).

### 1.3/ STRUCTURE DU MANUSCRIT

Chaque chapitre débute par une courte introduction du sujet traité.

Le chapitre 2, intitulé "État de l'art", présente en première partie un état des lieux des différentes dégradations pouvant se produire durant l'acquisition d'une image : les dégradations présentes de la capture de la scène à son stockage avant d'être visionnée. Ensuite, une grande partie des métriques pour l'évaluation des images sont présentées. Les métriques présentées sont les plus utilisées dans les travaux scientifiques pour quantifier la qualité des images. Ces métriques bien connues sont classées en deux catégories : les métriques pleine-référence et les métriques sans-référence. Puis, dans une dernière partie, sont présentées séparément deux méthodes d'imagerie computationnelle pour l'amélioration de la qualité des images : la déconvolution (des techniques de défloutage) et la Super Résolution (des techniques pour l'augmentation de la résolution spatiale).

Le chapitre 3, intitulé "Étude et développement", présente dans une première section notre étude réalisée sur la pertinence des métriques. Cette étude permet de comparer les résultats des métriques les plus courantes dans le but de vérifier si ces résultats sont cohérents avec les traitements appliqués sur les images de tests. Ensuite, dans la section suivante, nous évaluons une méthode de Super Résolution présente dans l'état de l'art : la méthode de Gohshi [127]. À partir de cette évaluation nous proposerons une nouvelle méthode originale qui permet d'augmenter le niveau de détails des images tout en pouvant être intégrée dans une cible matérielle. Cette méthode LASSR, pour Local Adaptive Spatial Super Resolution, permet d'adapter localement le traitement de Super Résolution en fonction du voisinage de pixel. Enfin, nous concluons ce chapitre par l'évaluation de notre méthode.

Le chapitre 4, intitulé "Accélérateur matériel" met en avant la réalisation du démonstrateur pour le projet H2020-EXIST. La première section de ce chapitre présente les étapes nécessaires à l'implantation matérielle de notre méthode LASSR. Tout d'abord la méthodologie de développement matérielle que nous avons suivie est présentée, puis toutes les parties importantes de notre architecture sont explicitées dans les sous-sections suivantes. La seconde section traite de la création d'un démonstrateur pour le projet H2020-EXIST réalisant la déconvolution en temps réel. Cette partie est détaillée dans le but de présenter la méthode permettant d'appliquer la déconvolution avec les contraintes matérielles que nous avons dans ce projet.

Enfin, nous concluons par le chapitre 5. Une première section de ce chapitre est consacrée à une discussion sur nos contributions. Une discussion à propos de notre étude sur des métriques, sur la mise en place de notre méthode de Super Résolution et à propos de notre travail sur la déconvolution temps réel. Ensuite une section "Perspectives" présente les évolutions et améliorations possibles de nos travaux. Enfin, une dernière section énumère les valorisations scientifiques que nous avons pu faire lors de ce travail de thèse.

**C**e Chapitre présente l'état de l'art des techniques permettant de répondre aux exigences du projet européen, et de trouver des solutions techniques relatives à l'augmentation de la résolution et à la correction des déformations optiques à l'aide de techniques de déconvolution.

La première section 2.1 présente un inventaire de différentes dégradations qu'il est possible de rencontrer lors de la capture des images, de sa mémorisation et de son affichage.

La section 2.2 de cet état de l'art présente des métriques couramment utilisées dans le domaine de la correction et de l'amélioration des images, en présentant des métriques avec ou sans image de référence.

La section 2.3 présente un état de l'art du domaine de l'imagerie computationnelle, un domaine permettant la correction et l'amélioration des images capturées, traitées dans le flux, ou en post-traitement, et nous mettrons l'accent sur des techniques de déconvolution qui permettent de réduire le flou et certaines déformations optiques et sur des techniques de Super Résolution (SR) des techniques les plus complexes aux techniques les plus simples à embarquer de Super Interpolation.

Enfin, une conclusion de cet état de l'art est apportée dans la section 2.4.

## Sommaire

<b>2.1</b>	<b>Pertes et dégradations : les éléments bloquants</b>	<b>8</b>
2.1.1	La prise de vue	9
2.1.2	Le système d'acquisition	12
2.1.3	La transmission, le stockage et l'affichage	17
2.1.4	Discussion	19
<b>2.2</b>	<b>Les métriques pour l'évaluation des images</b>	<b>21</b>
2.2.1	Les métriques pleine-référence	21
2.2.2	Les métriques sans-référence	26
2.2.3	Discussion	30
<b>2.3</b>	<b>L'imagerie computationnelle pour l'amélioration de la qualité de l'image</b>	<b>30</b>
2.3.1	La déconvolution, une technique de défloutage	33
2.3.2	Les techniques de super résolution	40
<b>2.4</b>	<b>Discussion et Conclusion</b>	<b>56</b>



## 2.1/ PERTES ET DÉGRADATIONS : LES ÉLÉMENTS BLOQUANTS

De la peinture rupestre à la photographie numérique, l'humain a toujours eu la volonté d'immortaliser et de retranscrire les scènes et objets qu'il pouvait voir. Malheureusement, même les artistes les plus talentueux n'arrivent pas à reproduire exactement ce qu'il est possible de voir, et ce problème n'est pas forcément lié au talent, pour la peinture, mais aussi aux limites technologiques des appareils d'acquisition utilisés.

La difficulté de capturer une image telle que nous l'observons est grande, en effet qui n'a jamais été déçu en regardant ses photographies d'étoiles, ou de couchers de soleil prises avec un appareil d'acquisition non adapté à la scène observée ?

Dans le domaine de l'acquisition numérique des images, cette différence de représentation entre la scène observée et l'image sauvegardée est due à de nombreuses dégradations et déformations optiques et électroniques qui se produisent lors de la construction de l'image au travers des différents éléments atmosphériques, optiques et électroniques présents.

La figure 2.1 présente un modèle de formation d'une image qu'il est possible de trouver dans la littérature [67, 82] où  $X$  représente la scène réelle que nous désirons capturer,  $B^{atm}$  et  $B^{opt}$  représentent les flous causés respectivement par les perturbations atmosphériques et par le système d'acquisition (plus particulièrement l'objectif).  $W$  est l'opérateur de déformation "warping", les pixels ont subi une rotation et une translation, ce qui crée une distorsion visuelle car ces transformations ne sont pas uniformes, elles sont en générales faibles au centre de l'image, et plus fortes vers les bords.  $D$  est l'opérateur de décimation "downsampling",  $N$  correspond au bruit et  $Y$  est l'image résultante. Le bruit  $N$  est un bruit additif provenant de divers facteurs tel que du bruit atmosphérique, et de bruit provenant du système d'acquisition (optique et électronique). L'image finale  $Y$  correspond à la version bruitée, basse résolution, floutée, déformée de l'image d'origine  $X$ . Le modèle optique présenté dans la figure 2.1 est exprimé de manière mathématique dans sa forme matricielle dans l'équation 2.1.

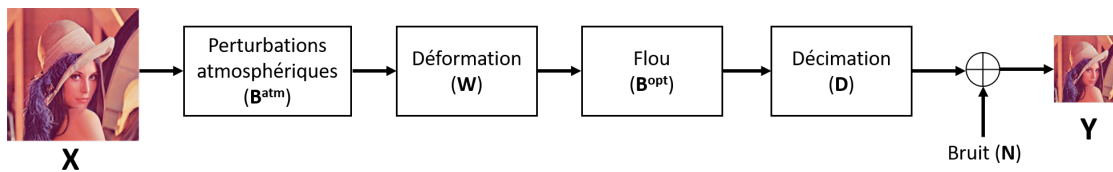


FIGURE 2.1 – Acquisition d'image présentée dans la thèse de Hiroyuki Takeda [82]

$$Y = D.B^{opt}.W.B^{atm}.X + N \quad (2.1)$$

Dans le but de simplifier l'inventaire des différents niveaux de dégradation et de perte d'informations lors de la construction des images, nous avons découpé le modèle précédent en trois parties, définies par l'origine des dégradations qui peuvent affecter l'image. Ces trois parties sont présentées dans la figure 2.2. La partie "prise de vue", détaillée section 2.1.1, regroupe des dégradations liées aux conditions atmosphériques, et au bougé éventuel. La section 2.1.2 présente les différentes dégradations liées au système d'acquisition. La dernière partie concerne les pertes qu'il peut avoir lors du stockage et de l'affichage de ces mêmes images, section 2.1.3.



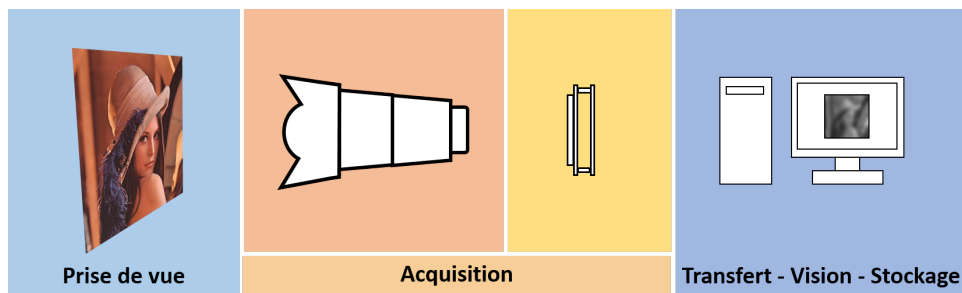


FIGURE 2.2 – Pertes d’informations et dégradations se produisant de la prise de vue à la visualisation de l’image

### 2.1.1/ LA PRISE DE VUE

Cette première partie de dégradations est directement liée à tout ce qui est extérieur aux systèmes d’acquisition, de sauvegarde et de diffusion. Il s’agit donc des problèmes liés à tout ce qui se trouve entre l’objet que nous désirons capturer et le système d’acquisition lui-même, qui regroupent les conditions atmosphériques, les conditions dans lesquelles la prise de vue a été faite et le choix du système d’acquisition.

#### Les conditions atmosphériques

Il est nécessaire que la lumière provenant de la scène et de l’objet puisse parvenir à l’œil, ou au système d’acquisition, pour qu’une image puisse se former. Dans l’idéal, il faudrait que cette lumière ne soit pas altérée pour que la représentation de la scène puisse se faire sans déformation et sans perte d’information. Toutefois les systèmes d’acquisition doivent faire face aux conditions atmosphériques, car l’atmosphère elle-même diffuse de l’énergie lumineuse, et des particules sont présentes entre l’objet et le système d’acquisition ce qui modifie l’image capturée. En effet ces particules représentent des obstacles dégradant la transmission de la lumière. Les éléments peuvent être présents dans l’air (brouillard, fumée, aérosols) ou dans l’eau. Quand ces particules sont abondantes, la visibilité s’amointrit, ce qui est problématique pour l’humain lorsqu’il conduit par exemple, et pose notamment des problèmes aux concepteurs de véhicules autonomes basés sur la vision.

Les interactions de la lumière avec l’atmosphère peuvent être classées en trois grandes catégories : diffusion, absorption et émission<sup>1</sup>. Il est possible de trouver dans la littérature [5, 11] plusieurs modèles décrivant les impacts visuels des perturbations atmosphériques sur les images acquises.

De nombreux chercheurs [181, 173] ont modélisé la dégradation des images comme étant une combinaison linéaire entre la radiance, et la lumière dans l’atmosphère. Cette dégradation dépend aussi de la distance entre l’objet et le système d’acquisition. Les dégradations dues à l’atmosphère sont modélisées dans la figure 2.3 et son modèle mathématique correspondant est présenté dans l’équation 2.2 pour un pixel  $x$  donné.

$$I(x) = J(x)t(x) + A(1 - t(x)) \quad (2.2)$$

Avec  $I$  l’image dégradée avec un flou atmosphérique,  $J$  l’image nette de l’objet,  $A$  la lumière atmosphérique et  $t(x)$  l’atténuation directe. L’atténuation directe dépend de la distance à l’objet  $d$ , du coefficient de diffusion  $\beta$  et des coordonnées du pixel  $x$ , l’atténuation est exprimée dans

1. [http://harold-clenet.com/?page\\_id=250](http://harold-clenet.com/?page_id=250)

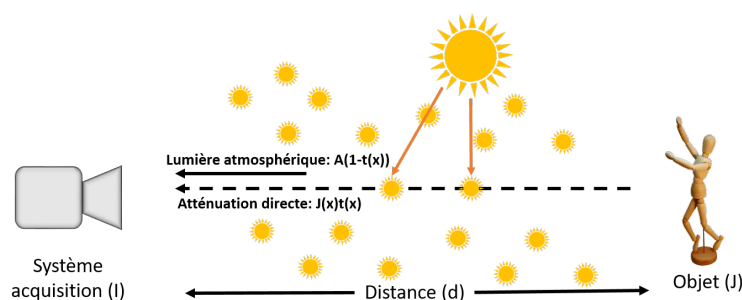


FIGURE 2.3 – Perturbations atmosphériques modifiant la formation d'une image présentée dans les travaux de Lee et al. [173]

l'équation 2.3. Dans des conditions météorologiques claires, nous avons  $\beta \approx 0$  et par conséquent  $I \approx J$ .

$$t(x) = \exp^{-\beta d(x)} \quad (2.3)$$

Ces dernières années les chercheurs ont montré un intérêt croissant pour le développement de techniques permettant l'amélioration des images affectées par des phénomènes climatiques. Le principal objectif [41, 43, 44, 54] a été d'améliorer les images prises avec une visibilité médiocre. L'image obtenue doit être la plus proche possible d'une image acquise avec une visibilité parfaite [50, 57, 47].

Ces techniques s'intitulent généralement "dehazing" dans l'état de l'art et de bons résultats ont été publiés comme dans les travaux de Schechner et al. [58] où les auteurs présentent plusieurs méthodes de dehazing ainsi que leurs résultats.

### Les différents flous (hors système d'acquisition)

En dehors des déformations et flous qui peuvent se produire en raison de mauvaises conditions atmosphériques, il existe d'autres types de flous dont le système d'acquisition n'est pas forcément responsable. Parfois le flou peut être produit de façon volontaire par le photographe afin de renforcer l'expression de la photo, mais il apparaît pour la plupart du temps de manière involontaire, réduisant la qualité de l'image. Ce flou peut être causé par une mauvaise mise au point, le mouvement de l'objet, le tremblement du photographe et bien d'autres causes [122]. La figure 2.4 montre un flou de bougé créé par le mouvement d'un bébé. Il est possible de voir que le flou est au premier plan, et que l'arrière-plan, qui est fixe, est net.

Des effets de flou apparaissent également lorsque la mise au point n'est pas bien réalisée. Des exemples de flou liés aux réglages de l'utilisateur sont représentés dans la figure 2.5 où deux mises au point différentes sont présentées. Dans le premier cas, le fond est net. Dans le second cas, l'objet au premier plan est net. Ici, la notion de flou dépend de l'intention du photographe, et n'est donc pas forcément un défaut au sens du traitement d'image.

Des solutions techniques permettent d'aider l'utilisateur à améliorer la qualité de ses prises de vue. La méthode du "steady cam" permet lorsque le système d'acquisition est mobile de stabiliser mécaniquement l'appareil réduisant les mouvements non désirés. Lorsque le système d'acquisition est fixe, il est nécessaire de le stabiliser avec un trépied par exemple. De plus certains appareils photos ou caméras disposent maintenant de stabilisateurs d'image mécaniques placés à l'arrière du capteur.

Enfin il est nécessaire d'adapter le temps d'exposition du système d'acquisition en fonction de la



FIGURE 2.4 – Une photographie contenant un flou de bougé

FIGURE 2.5 – Problème de mise au point, le point est fait sur le premier plan (*gauche*), le point est fait sur le fond (*droite*)

vitesse de l'objet. Plus la vitesse de l'objet est grande plus il faudra réduire le temps d'exposition, réduisant dans le même temps la luminance de l'image résultante.

Une fois l'image acquise il sera possible de réduire l'effet de flou et de renforcer certains contours en appliquant des techniques de déconvolution. Plus de détails seront apportés à ces techniques dans la section 2.3.1.

### Les limites du système d'acquisition

Le choix du système d'acquisition limite lui aussi la qualité de l'image résultante. Le choix du capteur va, par exemple, limiter un grand nombre de paramètres :

- La résolution,
- La sensibilité,
- La taille du pixel,
- Les temps d'exposition possibles,
- La nature du capteur (IR, noir et blanc, couleur, logarithmique, ...),
- La profondeur des couleurs, la dynamique de codage des valeurs des pixels.

Ces différents paramètres doivent être pris en considération en fonction du type d'image que nous désirons capturer, et du type d'informations que nous devons traiter.

Cependant il est possible, comme dans les travaux de Lapray et al.[171], avec certains capteurs, de générer des images ayant plus de dynamique en capturant par exemple plusieurs images avec des temps d'exposition différents et en appliquant des algorithmes d'augmentation de la dynamique de l'image (HDR) [92].

De même, des solutions de super résolution peuvent être appliquées dans le but d'augmenter la résolution des images capturées. La super résolution permet après la capture, ou dans le flux, d'augmenter le niveau de détail. Ces techniques de Super Résolution (SR) sont présentées dans la section 2.3.2.

### 2.1.2/ LE SYSTÈME D'ACQUISITION

Cette partie correspond aux dégradations et déformations provoquées par le système d'acquisition. Pour l'imagerie numérique, le système d'acquisition est la plupart du temps divisé en deux parties : une partie optique (objectif, lentille) et une partie électronique (capteur, carte électronique de gestion du capteur).

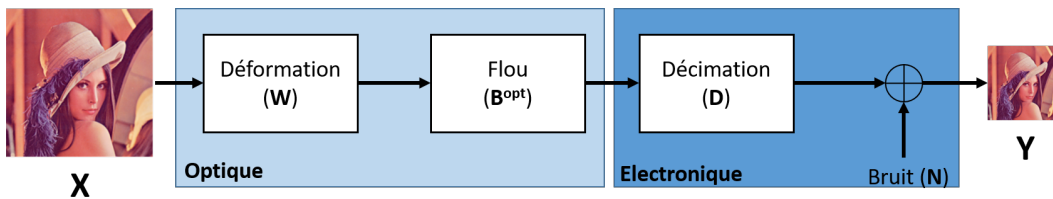


FIGURE 2.6 – Dégradations de l'image présentées par [67] où nous avons différencié les dégradations optiques et électroniques

En guise de rappel, la figure 2.6 met en avant un modèle de dégradations optiques proposé par Farsiu et al. [67]. Ce modèle est formalisé dans l'équation 2.4.

$$Y = D.B^{opt}.W.X + N \quad (2.4)$$

Nous considérons une image source  $X$  de très haute qualité et de grande résolution représentant exactement la scène que nous désirons capturer.

La première dégradation subie par  $X$  est le  $W$  "warping". Ceci correspond à une déformation de l'image, radiale dans la majorité des cas. Les pixels ont subi une rotation et une translation, ce qui crée une distorsion visuelle car ces transformations ne sont pas uniformes.

Les dégradations atmosphériques ne sont pas présentées dans cette partie, de ce fait seul le flou  $B^{opt}$ , directement lié à l'objectif, est appliqué à l'image déformée par  $W$ . Ensuite une décimation  $D$  est appliquée à l'image floue, cette décimation est directement liée à la résolution du capteur. Enfin le bruit numérique  $N$  est considéré comme additif.

#### 2.1.2.1/ LES DÉGRADATIONS OPTIQUES

Le choix de l'optique va directement influencer la qualité de l'image. Dans certain cas, un effet de vignettage apparaît. Cet effet, illustré figure 2.7, correspond à un assombrissement des zones périphériques de l'image par rapport au centre. Malheureusement cette diminution de la luminosité s'observe dans la plupart des objectifs. Cependant cet assombrissement est quantifié et les constructeurs des objectifs peuvent fournir les profils de l'objectif, ce qui permet de corriger ce type de dégradations, à condition qu'il ne soit pas trop prononcé.



FIGURE 2.7 – Exemple d’assombrissement périphérique dû à l’objectif : le vignettage.

En fonction de la qualité de l’objectif et des verres qui le constituent, des aberrations, faibles ou importantes, peuvent se produire lors de la capture des images. On distingue deux types d’aberrations.

- Les aberrations chromatiques. Ces aberrations sont dues aux différents indices de réfraction des matériaux qui composent un objectif. Cette réfraction produit des mises au point différentes en fonction des longueurs d’onde présentes dans la lumière. Par exemple, lorsque la mise au point est faite sur le rouge, le bleu sera flou, voir figure 2.8.

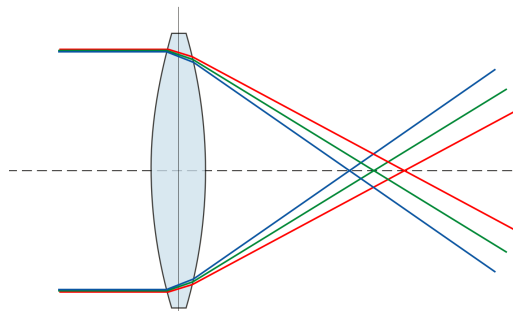


FIGURE 2.8 – Principe des aberrations chromatiques, décalage des différents canaux dû à la réfraction de la lentille.

- Les aberrations géométriques, qui conduisent à des déformations de l’image sont directement liées à l’objectif. Il existe trois principales distorsions représentées dans la figure 2.9. Ces déformations sont principalement dues à la forme de la première lentille de l’objectif. En effet, quand cette lentille est bombée, l’image a plus de risque d’être distordue en "barillet".

Pour réduire ces aberrations, la solution idéale serait d’augmenter la qualité des éléments constituant les objectifs. Cependant, cela rendrait encore plus onéreux ce genre de matériel. La majorité des fabricants d’objectifs proposent des modèles optiques de leurs objectifs, permettant de faire des corrections a posteriori à l’aide de logiciels spécifiques. La technique la plus simple pour corriger les déformations géométriques est de faire du "remapping", c’est à dire de replacer les pixels à leur emplacement d’origine dans  $I_s$ . L’inconvénient de cette technique est que l’image corrigée comporte des zones sans pixels et il est nécessaire de recadrer l’image. Cependant cette technique est très simple à mettre en place, car il suffit que le fabricant fournisse une table de concordance

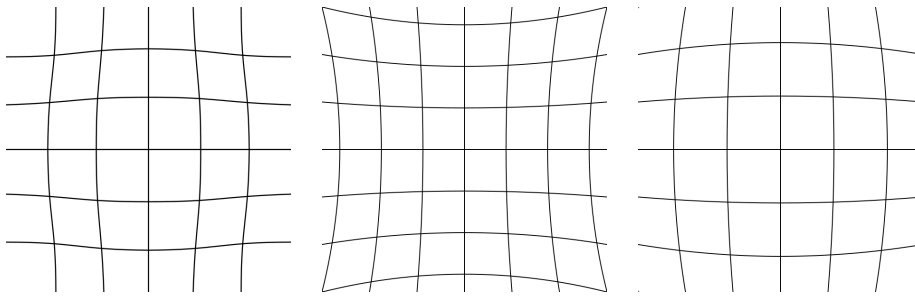


FIGURE 2.9 – Trois principales déformations géométriques dues à l'objectif : moustache, coussinet, barillet (de gauche à droite)

correspondant à la déformation. Concernant les aberrations chromatiques il est possible dans certain cas de les corriger avec des méthodes de déconvolution, section 2.3.1, appliquées à chaque canal.

#### 2.1.2.2/ LES DÉGRADATIONS ÉLECTRONIQUES

Des dégradations et pertes apparaissent dans les composants électroniques. En premier lieu, des pertes peuvent être directement liées au choix du capteur en lui-même. Le capteur va tout d'abord fixer la résolution spatiale des images. Ensuite, il se peut que le capteur comporte des défauts, le plus répandu étant le "pixel mort". Cela veut dire qu'un des photorécepteurs qui composent le capteur d'image ne fonctionne pas. La valeur du pixel mort va apparaître clairement lorsqu'une image uniforme est capturée. En fonction du nombre de pixels non fonctionnels, le capteur pourra être utilisé ou non. Le moyen le plus simple pour corriger un pixel mort est de remplacer la valeur de celui-ci par la moyenne ou la valeur médiane des pixels voisins. Dans certain cas il s'agit tout simplement de la recopie d'un pixel proche. De ce fait l'information de ce pixel n'est pas la bonne mais elle reste cohérente avec les pixels alentours.

Outres ces limitations et défauts, la plupart des capteurs d'images ne sont pas parfaits. L'image résultante va dépendre des effets que les composants électroniques vont avoir sur le signal électrique. Les dégradations présentes dans les capteurs d'imagerie numérique correspondent au bruit électronique. Ce bruit occasionnera les informations parasites et dégradations de l'image depuis son acquisition dans le capteur jusqu'à son enregistrement sur un support de type mémoire vive ou mémoire de masse. Les sources de ces bruits sont multiples, elles peuvent être liées aux conditions d'éclairage et à la température du capteur lors de l'acquisition, et d'autres perturbations peuvent se produire lors du transfert des informations.

La figure 2.10, issue des travaux de Gow et al. [83], représente la chaîne de transmission du signal de la photodiode jusqu'à la transmission en dehors du capteur de l'information codée représentant la valeur du pixel. Il est possible de voir qu'un grand nombre de bruits sont ajoutés au fur et à mesure que le signal traverse différents éléments électroniques du capteur d'image.

Le dernier bruit présenté dans cette chaîne est le bruit de quantification. Il correspond aux pertes d'informations qui se produisent lors de la conversion du signal analogique vers le signal numérique, lors de l'échantillonnage. Pour réduire ces pertes d'informations il est nécessaire d'utiliser des CAN " *convertisseurs analogique/numérique*" les plus précis possible. Plus le nombre de bits du CAN est grand, plus la précision sera grande pour un signal analogique donné. Le choix du CAN fixe par exemple la dynamique du capteur, les pixels étant communément codés sur 8, 10, 12 ou 14 bits.



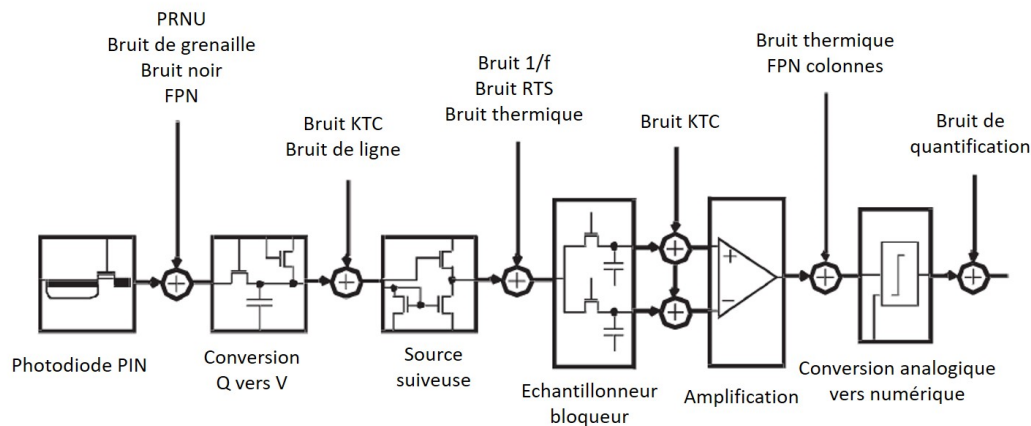


FIGURE 2.10 – Représentation des origines des principaux bruits électroniques pour un capteur de technologie CMOS présentés dans les travaux de Gow et al. [83]

Ensuite, il est possible de séparer en deux catégories distinctes les bruits qui sont présentés dans la figure 2.10 : le bruit spatial et le bruit temporel. Certains de ces bruits sont explicités dans les paragraphes suivants.

### Le Bruit spatial

Le bruit spatial correspond à la variation spatiale de la réponse des pixels dans une image, c'est-à-dire, pour une même illumination, des valeurs variables d'un pixel par rapport aux autres. Ces différences de réponses sont stables dans le temps lorsque les conditions de prise de vue ne changent pas. Le bruit spatial peut être évalué en calculant le coefficient de variation pour une image.

Le bruit de motif, couramment appelé FPN pour "Fixed Pattern Noise", est facilement reconnaissable sur les images provenant de capteur CMOS car ce bruit est visible sur les colonnes. Ce bruit est très visible lorsque le capteur n'est pas exposé à la lumière. Le FPN correspond à la différence pixel à pixel d'une image capturée dans l'obscurité. Ces décalages peuvent provenir des différents éléments présents dans la chaîne de conversion du signal provenant du photorécepteur. De plus les pistes du circuit ne sont pas forcément réalisées de façon identique (longueur de piste) ce qui peut faire varier la valeur finale du pixel et augmenter la différence d'un pixel avec son voisin. Dans la majorité des caméras du commerce, ce bruit est supprimé directement dans la caméra en soustrayant une image acquise dans le noir à l'image courante. Ce principe simple est illustré dans la figure 2.11 où il est possible de voir sur l'image de gauche un bruit sur les colonnes, très visible dans les zones saturées. Ce bruit est isolé dans l'image du centre en réduisant le temps d'exposition au minimum et en obturant l'objectif, cette image est stockée et est ensuite utilisée comme un masque. Enfin l'image de droite est créée en soustrayant la masque à l'image courante. Cependant il est nécessaire de suivre certaines consignes pour améliorer la prise de l'image de noir : le temps d'intégration doit être réduit pour limiter les effets du courant d'obscurité. De plus il est conseillé de faire la moyenne sur plusieurs images de noir afin de réduire les effets induits par d'autres bruits qui pourraient être temporels.

Comme le FPN, le PRNU pour *Photo Response Non uniformity* est un bruit spatial qui ne varie pas en fonction du temps. Cependant ce bruit varie en fonction de la luminosité. Ce bruit provient dans la majorité des cas de différences de propriétés optiques entre chaque pixel et de différences de valeurs des condensateurs de conversion des canaux de lecture. Ces différences sont dues aux imperfections de réalisation des composants électroniques (capacité, diode de jonction). Chaque

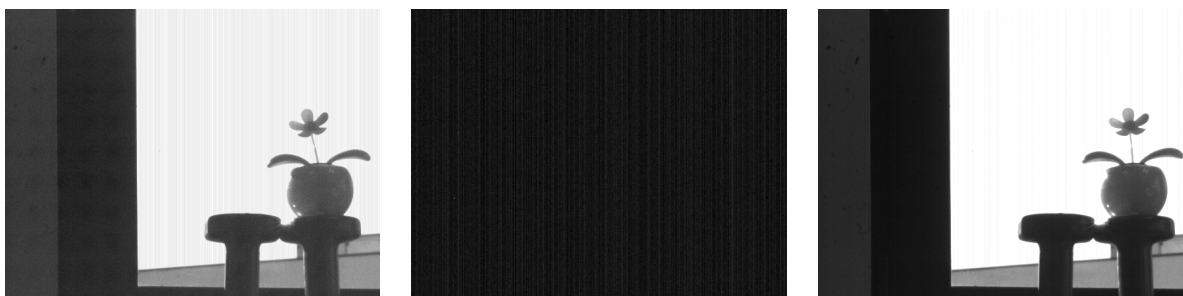


FIGURE 2.11 – Principe de suppression du FPN, L'image avec FPN à gauche, l'image "noire" de FPN au centre, et enfin l'image corrigée à droite

pixel aura une réponse légèrement différente pour la même quantité de lumière. Cette réponse peut être corrigée en modifiant le gain de chaque pixel [170], cependant cette correction se réalise généralement après capture, que ce soit de façon logicielle, ou matérielle, en suivant une routine de calibration fournie par le fabricant du capteur généralement fournie dans la documentation technique, ou des notes d'application<sup>2</sup>.

Généralement que ce soit au niveau esthétique, ou simplement pour améliorer l'efficacité des systèmes d'analyses, il est la plupart du temps nécessaire de réduire l'apparition de ces bruits. Cependant ces bruits résiduels peuvent être utilisés pour l'authentification des images, voir si l'image a été falsifiée en ajoutant ou supprimant des éléments [101, 151, 104]. Il est tout autant possible d'identifier l'origine des capteurs [91] en utilisant le FPN et PRNU. Ces techniques sont possibles grâce à l'unicité des bruits comme le PRNU, en effet étant donné que le bruit est spécifique à un même capteur, il sera possible de détecter dans une image les zones ne comportant pas la même signature de bruit.

Fridrich a pu montrer dans [101] la robustesse de son traitement de vérification de falsification de l'image lorsque l'image a été débruitée et compressée avec JPEG.

### Le Bruit temporel

Le bruit temporel est, contrairement au bruit spatial, un bruit qui varie au cours du temps même quand la lumière arrivant au capteur est constante. Ce bruit provient de deux causes différentes. Tout d'abord des perturbations peuvent intervenir au cours du temps en fonction du lieu où le système d'acquisition se trouve. Il est possible en fonction de ce lieu que le système soit victime de perturbations électromagnétiques provenant d'ondes se propageant dans l'air mais aussi se propageant par ce qui est connecté électriquement au système d'acquisition. Afin de réduire ce genre de perturbations il est nécessaire de prévoir l'utilisation du système d'acquisition en respectant les règles de compatibilité électromagnétique prévue à l'endroit où il sera utilisé. Il est possible qu'un système qui fonctionne bien chez un particulier donne des images bruitées lorsqu'il est utilisé en milieu industriel.

Mais ces perturbations peuvent se produire à l'intérieur même des circuits du système d'acquisition. Le signal circulant dans des pistes électroniques peut perturber les signaux voisins. Ces perturbations vont se manifester par un décalage de phase, ou retard sur le signal. Ce bruit correspond au bruit de "gigue" (jitter en anglais). De ce fait, lors de la lecture de la valeur du pixel, il sera possible que cette valeur soit décalée par rapport à l'horloge de lecture, et la valeur lue ne sera donc pas la bonne.

2. <https://www.cypress.com/file/142911/download>



Ensuite d'autres bruits temporels apparaissent lors du passage des signaux électriques dans les composants qui constituent le capteur, la carte capteur, et les cartes électroniques qui traitent et envoient les données vers l'extérieur du système d'acquisition. Ce type de bruit est pour la plupart du temps thermique, et se nomme différemment selon son origine de celui-ci :

- Le bruit de Johnson est un bruit qui se produit lorsque des électrons traversent les composants résistifs.
- Le bruit KTC créé par les condensateurs du circuit, peut être réduit en utilisant des circuits de double échantillonnage dans le circuit de lecture du pixel ; le but est de soustraire de la valeur du pixel intégré la valeur de ce même pixel qu'il avait au début de l'intégration du signal. De ce fait le bruit thermique courant produit par les condensateurs est réduit pixel à pixel.

De plus, le bruit en  $1/f$  est présent dans tous les composants actifs et présent dans certains composants passifs. Il est aussi appelé bruit de scintillation, bruit de scintillement, bruit de papillotement, bruit de basse fréquence, bruit en excès ou bruit de flicker. Il peut être dû aux impuretés présentes dans les matériaux.

En guise d'exemple de bruit thermique, la figure 2.12 illustre quatre images issue d'une séquence vidéo contenant du bruit dit "poivre et sel". La proportion de bruit diminue en fonction du temps et du refroidissement du capteur d'image. Ces images proviennent d'une caméra infrarouge InGaAs couplée avec un refroidisseur [184]. Cette figure met en avant le bruit thermique qui peut se produire lors de la capture d'image avec cette caméra. Ces images ont été prises avec un temps d'exposition assez important (2s) permettant d'être plus sensibles aux effets du bruit thermique accumulé et des autres bruits associés. L'évolution de la température du capteur diminue de 300K à 77K.

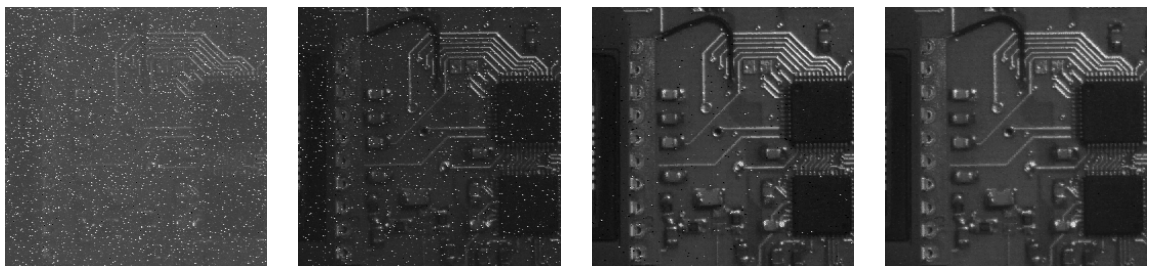


FIGURE 2.12 – Images capturées par une caméra IR InGaAs mettant en avant l'évolution du bruit thermique [184] en fonction d'un refroidissement du capteur de 300K à 77K avec un temps d'exposition de 2 secondes.

Des techniques de traitement d'image permettent parfois de corriger ou d'améliorer les images bruitées. Par exemple, il est possible de réduire le bruit "poivre et sel" en appliquant un filtre médian sur l'image [133]. Cette technique permet de remplacer les valeurs extrêmes (0 ou 255 pour un capteur 8 bits) par la valeur médiane des pixels voisins. La figure 2.13 présente une image bruitée artificiellement avec "poivre et sel" et corrigée avec un filtre Médian  $3 \times 3$ .

### 2.1.3/ LA TRANSMISSION, LE STOCKAGE ET L'AFFICHAGE

#### Compression pour la transmission et le stockage

Cette dernière partie correspond aux pertes et limitations dues au transfert, au stockage et à la représentation des données capturées et sauvegardées.

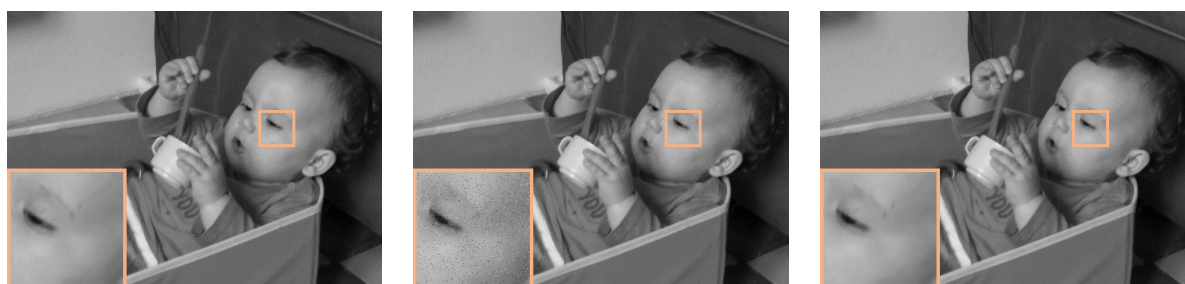


FIGURE 2.13 – Exemple de bruit "poivre et sel" (*centre*) sur l'image source (*gauche*) et corrigé avec un filtre médian 2d (*droite*) effectué sur un voisinage de  $3 \times 3$  pixels.

Il s'agit donc de dégradations post-capture qui ne sont ni liées aux conditions de prise de vue, ni au système d'acquisition.

Le système d'acquisition va tout de même limiter la qualité des images, puisqu'il en détermine la dynamique via sa taille de codage. Cette qualité va aussi dépendre de la nécessité de compresser l'image avant de la transmettre afin de respecter les débits disponibles de transfert entre le système d'acquisition et le système de stockage.

L'idéal est que l'image soit transférée et sauvegardée dans un format brut, en général appelé "RAW". L'image est transmise avec un minimum de modifications. Ce format est idéal dans le domaine de la photographie, où il est ainsi possible de traiter manuellement et hors ligne chaque image séparément, tirant ainsi le meilleur parti de chacun des pixels. C'est aussi le cas pour la plupart des traitements d'images nécessitant une grande précision, comme la reconnaissance de formes ou la détection de défauts sur des produits industriels.

Cependant, il est très souvent nécessaire de réduire la taille des données pour faciliter le transfert et le stockage de celles-ci. Cette réduction de taille peut être réalisée à l'aide des techniques de Compression/Décompression.

Dans la plupart des cas, la compression est possible en réduisant le volume des informations redondantes : redondance spectrale, spatiale et temporelle [59, 68, 69, 187].

Deux types de compressions existent dans la littérature : la compression avec pertes, et la compression sans perte. Quelques informations sur ces méthodes sont présentées dans la tableau 2.1.

Type	Sans perte	Avec pertes
Utilisation	Données médicales, conservation des détails dessins techniques	Audiovisuel, la diffusion, réduction de la bande passante
Méthodes usuelles	RLE, Codage entropique [142] Codage arithmétiques [124], Codage de Huffman [4].	Scalar quantization [190], Vector quantization [187], DCT [51], Fractal compression [33].
Taux de compression	< 2 [78]	> 10 [177, 201]
Avantages	réversible	taux de compression
Inconvénients	taux de compression	irréversible

TABLE 2.1 – Comparaison des deux types de méthodes de compression d'image : avec pertes et sans perte.

La figure 2.14 présente une image de référence (gauche) ainsi qu'une version de celle-ci compressée avec le CoDec JPEG-2000 (avec perte). En terme de stockage, l'image compressée représente seulement 10% de la taille de l'image de référence, ce qui représente un gain important en taille, au prix de dégradations importantes.

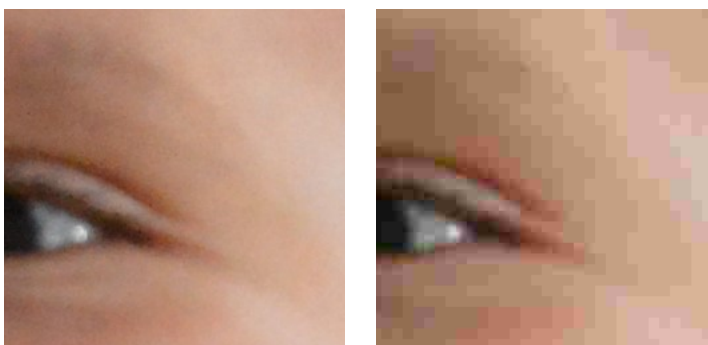


FIGURE 2.14 – Exemples de pertes d'information à la suite de compression avec pertes JPEG-2000. Image non compressée (*gauche*) et la version de l'image compressée avec un taux de compression de 90% (*droite*)

La plupart des chaînes ou systèmes de diffusion en ligne adaptent automatiquement la résolution et la qualité du contenu diffusé en fonction de la bande passante disponible. La compression, quand la bande passante est faible, induit parfois une perte de qualité visuelle importante de l'image. Toutefois, la télévision Japonaise diffuse depuis le 1<sup>er</sup> Décembre 2018 la chaîne NHK en 8k, en grande partie grâce à l'évolution des techniques de compression.

### La visualisation

Enfin, le dernier élément de cette chaîne correspond à l'étape de visualisation de l'image capturée. La visualisation peut avoir pour but le simple divertissement (film, photographie) ou l'analyse précise de contenu (vidéo assistance, chirurgie à distance, analyse de défauts, etc.). Dans tous ces domaines, la perception de la qualité de l'image peut être altérée en fonction de la qualité du support de visionnage (grand écran de type LCD, écran de téléphone, écran de projection, etc.). Il est donc nécessaire d'adapter l'écran à la problématique considérée, en terme de gamme de couleur, de résolution, de dynamique. De plus il est souvent nécessaire que le système de visualisation soit calibré pour assurer une reproduction fidèle des couleurs et des contrastes d'un poste à l'autre. Cette calibration est réalisée à l'aide d'une sonde colorimétrique et d'un logiciel spécifique, il existe dans la littérature des méthodes de calibration automatiques réalisées avec des webcams non calibrées [131, 96, 110].

#### 2.1.4/ DISCUSSION

Il a été possible de voir qu'un grand nombre de facteurs bien différents interviennent tout au long de l'acquisition d'une image : ces perturbations atmosphériques, optiques ou électroniques dégradent et limitent la qualité de l'image acquise.

La majeure partie de ces perturbations peuvent être corrigées directement dans le système d'acquisition, comme les bruits électroniques liés au capteur PRNU, FPN, et d'autres peuvent être limités en soignant l'utilisation du système d'acquisition (gestion de l'environnement, de l'éclairage, de perturbations électromagnétiques). Cependant certaines corrections complexes peuvent être réalisées après le stockage initial, ou même dans le flux. Ces corrections numériques

font partie de la famille de l'imagerie computationnelle, dont nous allons préciser les contours et les méthodes liées à notre projet dans la section 2.3, après avoir étudié les métriques permettant l'évaluation de la qualité des images dans la section suivante. Une bonne connaissance de ces métriques est en effet nécessaire pour évaluer les traitements présentés dans les différentes parties de ce travail de thèse.

## 2.2/ LES MÉTRIQUES POUR L'ÉVALUATION DES IMAGES

L'évaluation de la qualité des images est une part très importante dans le développement d'applications liées à l'image. Un des domaines qui utilise le plus les métriques d'évaluation de la qualité de l'image est le domaine des techniques de compression. En effet, lors du développement d'algorithmes de compression avec pertes, les images décompressées ne sont pas identiques aux images avant la compression : plus il y a de perte de compression/décompression moins la qualité de l'image résultante est bonne.

Dans le cas plus général ces métriques sont ainsi utilisées pour valider les performances de nombreux algorithmes liés à l'image, comme par exemple l'évaluation de méthode de "dehazing", de HDR, de défloutage et même de Super Résolution.

Les métriques liées à la qualité de l'image peuvent être classées dans plusieurs catégories. Il existe deux grande familles distinctes : les métriques subjectives, et les métriques objectives.

Les métriques dites objectives consistent à évaluer des entités purement mathématiques déterminées à partir des valeurs des données contenues dans les images, sans intervention humaine, et en faisant abstraction du système de visualisation. Ces métriques sont basées sur des grandeurs comme l'erreur quadratique moyenne EQM, le PSNR *Peak Signal to Noise Ratio*, qui permettent d'évaluer la similitude entre deux images ou imagerie.

A contrario, les métriques subjectives prennent en considération le jugement humain. Ce jugement est différent d'un être humain à un autre. Seuls les humains, qui sont les utilisateurs finaux de ces applications, sont capables de juger précisément de la qualité perçue. C'est pour cela qu'un grand nombre de métriques subjectives ont été créées pour tenter de simuler l'évaluation humaine, afin de pouvoir rendre ces évaluations automatiques et plus rapides. Les métriques subjectives sont largement utilisées pour l'évaluation de la qualité perçue des images, que ce soit dans le domaine du HDR, de la déconvolution ou de la super résolution.

Certaines de ces métriques subjectives prennent en considération le fonctionnement des champs récepteurs de la rétine. Ces champs récepteurs sont plus sensibles au contraste près des contours qu'au niveau de luminance dans toute l'image comme ont pu le montrer Hubel et Wisel [42].

Ces deux familles sont parfois sous-divisées en fonction de la nécessité d'avoir des images de référence pour l'évaluation de la métrique. Le EQM par exemple est une métrique pleine référence car il est nécessaire d'avoir une image de référence pour calculer l'erreur quadratique moyenne avec une autre image. D'autres métriques ne nécessiteront pas de référence, comme par exemple certaines évaluations subjectives où seule l'image en sortie de traitement est évaluée.

La finalité de cette section est de résumer les différentes métriques de l'image couramment utilisées, et de vérifier enfin si ces métriques sont cohérentes avec le jugement humain, et applicables pour l'évaluation de techniques d'amélioration de la qualité et de la résolution des images. Nous nous sommes particulièrement intéressés aux métriques dont le code source est disponible.

### 2.2.1/ LES MÉTRIQUES PLEINE-RÉFÉRENCE

#### PSNR

Le PSNR, pour *Peak Signal to Noise Ratio*, est une métrique pleine référence et totalement objective. En effet, cette métrique n'est basée que sur des valeurs de comparaison entre deux images

qui ne prennent pas en compte le système visuel humain. Le PSNR (équation 2.5), est le rapport entre la valeur de la dynamique  $d$  du signal au carré, divisé par l'erreur quadratique moyenne EQM (équation 2.6) entre les deux images  $f$  ( image de référence) et  $g$  (image à tester), et est exprimé en décibel. La dynamique  $d$  est la valeur maximale que peut prendre le signal, elle est par exemple de 255 niveaux de gris pour une image codée sur 8bits.

$$PSNR(f, g) = 10 \log_{10} \left( \frac{d^2}{EQM(f, g)} \right) \quad (2.5)$$

$$EQM(f, g) = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (f(i, j) - g(i, j))^2 \quad (2.6)$$

Cette métrique est couramment utilisée dans l'évaluation des méthodes de compression et décompression des images, avec pertes. Le PSNR (ou EQM) mesure l'erreur pixel à pixel entre les deux images ; plus EQM est faible, plus l'image traitée est proche de l'image source.

Cette métrique étant strictement objective, des auteurs ont pu montrer les limites de cette évaluation. En effet, dans [105], les auteurs Wang et Bovik, montrent avec plusieurs tests que pour des qualités d'images visuellement totalement différentes il est possible d'avoir les mêmes valeurs de EQM (donc de PSNR). Cette limite a été montrée aussi dans les travaux de Nasonov et al. [119].

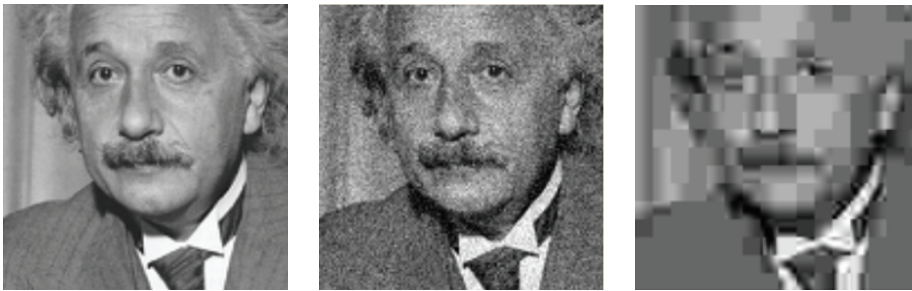


FIGURE 2.15 – Limite de EQM présentée dans l'article de Wang et Bovik [105], les trois images dégradées ont la même mesure de EQM=309, alors qu'elles présentent des caractéristiques visuelles très différentes.

### SSIM [71]

Les auteurs de SSIM, Wang et Bovik [71], précisent que cette métrique n'utilise pas une méthode habituelle de sommation d'erreurs entre les deux images à évaluer  $f$  et  $g$ , avec  $f$  l'image de référence et  $g$  l'image à tester. Les auteurs ont mis en place une modélisation des distorsions entre les deux images avec une combinaison de trois facteurs différents (équation 2.7). Cette métrique est une évolution de leur précédente métrique IQI (Image Quality Index [63]).

$$SSIM(f, g) = l(f, g) \cdot c(f, g) \cdot s(f, g) \quad (2.7)$$

Le premier terme  $l(f, g)$ , équation 2.8, correspond à une comparaison de la luminance moyenne entre les deux images  $f$  et  $g$ , la proximité des deux images, en terme de luminance, se calcul en utilisant la moyenne de celles-ci  $\mu$ .

La constante  $C1$  permet d'éviter que le diviseur s'annule lorsque  $\mu_f^2 + \mu_g^2 \approx 0$ , avec  $C1 = (K_1 L)^2$

où  $L$  est la dynamique des pixels (255 pour une image en niveaux de gris codée sur 8 bits, et  $K_1$  une constante très inférieure à 1.

$$l(f, g) = \frac{2\mu_f\mu_g + C1}{\mu_f^2 + \mu_g^2 + C1} \quad (2.8)$$

Le second terme  $c(f, g)$ , équation 2.9, permet de mesurer la proximité des deux images en fonction du contraste. Le contraste est mesuré en utilisant l'écart type standard  $\sigma^2$  des images.

La constante  $C2$  permet d'éviter que le diviseur s'annule lorsque  $\sigma_f^2 + \sigma_g^2 \approx 0$ , avec  $C2 = (K_2L)^2$  où  $L$  est la dynamique des pixels (255 pour une image en niveaux de gris codée sur 8 bits, et  $K_2$  une constante très inférieure à 1.

$$c(f, g) = \frac{2\sigma_f\sigma_g + C2}{\sigma_f^2 + \sigma_g^2 + C2} \quad (2.9)$$

Le troisième terme  $s(f, g)$ , équation 2.10, permet de comparer la structure des deux images. La comparaison de la structure est calculée à l'aide d'un coefficient de corrélation entre les deux images. Ici l'auteur définit la constante  $C3$  tel-que  $C3 = C2/2$ .

$$s(f, g) = \frac{\sigma_{fg} + C3}{\sigma_f\sigma_g + C3} \quad (2.10)$$

Les auteurs précisent que les constantes  $C1$ ,  $C2$  et  $C3$  permettent de stabiliser le SSIM lorsque les moyennes  $\mu$  et les variances  $\sigma$  deviennent petites.

La métrique SSIM est une évolution de l'index de qualité IQI [63] développé par les mêmes auteurs, la différence entre ces deux métriques est l'ajout d'une stabilité des calculs avec grâce aux constantes  $C1$ ,  $C2$  et  $C3$ .

### VIF [74]

VIF pour *Visual Information Fidelity* est une métrique qui consiste à mesurer les informations mutuelles entre l'image dégradée  $F$  et l'image de référence  $E$  dans le référentiel humain, figure 2.16.

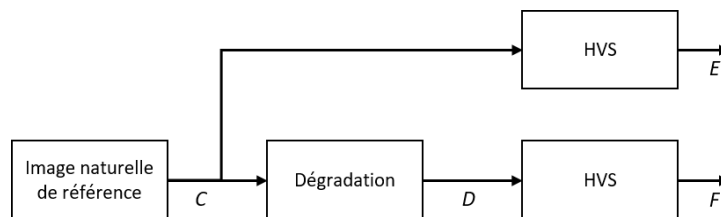


FIGURE 2.16 – Principe de mesure de VIF [74]

L'auteur de cette métrique précise que chaque élément de la chaîne de traitement de VIF est modélisé séparément pour ensuite pouvoir mesurer le VIF. Ces éléments, présents dans la figure 2.16, sont :

- Le modèle de la formation de la source,
- Le modèle de dégradation,
- Le modèle du système visuel humain HVS.



Tout d'abord l'image naturelle  $C$  est modélisée par un modèle de mélange gaussien *GMM* (*Gaussian Mixture Model* en anglais) telle que  $C = S \cdot U = \{S_i \cdot \vec{U}_i : i \in I\}$ , avec  $S$  un champ aléatoire scalaire positif,  $U$  un vecteur Gaussien aléatoire avec une moyenne nulle et une covariance  $C_u$ .

L'image dégradée  $D$  est modélisée par une combinaison du signal atténué (gain  $G$ ) et d'un bruit additif  $V$ , ce qui donne :  $D = GC + V = \{g_i \vec{C}_i + \vec{V}_i : i \in I\}$ .

Ensuite, le modèle du système de vision humain est représenté par l'ajout de bruit sur les signaux d'entrées tel que  $E = C + N$  et  $F = D + N'$  avec  $N = \{\vec{N}_i : i \in I\}$  et  $N' = \{\vec{N}'_i : i \in I\}$  des bruits non corrélés gaussiens avec les mêmes dimensions que  $C$ .

L'équation 2.11 présente le calcul du critère VIF sur un nombre  $j$  de canaux de l'image avec  $I(\vec{C}^{N,j}, \vec{F}^{N,j} | S^{N,j})$  et  $I(\vec{C}^{N,j}, \vec{E}^{N,j} | S^{N,j})$  les informations que le cerveau humain peut extraire pour respectivement l'image dégradée  $F$  et l'image de référence  $E$ .

$$VIF = \frac{\sum_{j \in \text{canaux}} I(\vec{C}^{N,j}, \vec{F}^{N,j} | S^{N,j})}{\sum_{j \in \text{canaux}} I(\vec{C}^{N,j}, \vec{E}^{N,j} | S^{N,j})} \quad (2.11)$$

### VSI [156]

La métrique VSI, pour Visual Saliency Index, se base tout d'abord sur la carte de saillance de l'image  $S_{VS}$ . La saillance représente tout ce qui capte le regard, ce qui ressort de l'image lorsque l'œil humain la perçoit. Il s'agit pour cette métrique plus particulièrement de relief et de contours de l'image.

Cependant la carte de saillance ne suffit pas pour justifier la qualité d'une image, et la différence de saillance entre l'image de référence et l'image dégradée ne permet pas de justifier une réelle évaluation entre les deux images. La mesure de  $S_{VS}$  n'est pas pertinente lorsque le contraste de l'image varie entre l'image de référence  $I_1$  et l'image dégradée  $I_2$ .

Afin de compléter la métrique, l'auteur ajoute deux caractéristiques : le module du gradient de l'image, et les informations de chrominance extraites de  $I_1$  et  $I_2$ . Ces deux nouvelles caractéristiques sont extraites de l'image dans un autre espace de couleur. L'auteur utilise l'espace de couleur LMN [79] où L est le canal correspondant à la luminance, M et N sont les canaux décrivant la chrominance de l'image.

Le module du gradient de Sharr est appliqué sur le canal L des images, ce gradient suit le principe des gradients de Sobel et de Prewitt avec des coefficients plus élevés. Ce module permet la génération de la seconde caractéristique de la métrique  $S_G$ .

Enfin, le dernier paramètre de cette métrique,  $S_C$ , est basé sur les deux canaux de chrominance de l'espace de couleur LMN.

L'équation 2.12 représente les contributions des trois composants de la métrique, avec  $\alpha$  et  $\beta$  deux paramètres permettant d'ajuster l'importance entre les différentes parties de la métrique (la saillance, le gradient et les caractéristiques de la chrominance).

$$S(x) = S_{VS}(x) \cdot [S_G(x)]^\alpha \cdot [S_C(x)]^\beta \quad (2.12)$$

Finalement l'équation 2.13 présente la métrique VSI entre les images  $I_1$  et  $I_2$ . Avec l'ajout de



$VS_m(x)$  où  $VS_m(x) = \max(VS_1(x), VS_2(x))$ ;  $VS_m(x)$  permet de pondérer l'importance de  $S(x)$  et  $\Omega$  correspond à tout le domaine spatial.

$$VSI = \frac{\sum_{x \in \Omega} S(x) \cdot VS_m(x)}{\sum_{x \in \Omega} VS_m(x)} \quad (2.13)$$

### FSIM, FSIMc [125]

Cette métrique a été créée, tout comme VSI, par Lin Zhang [125]. Le but de cette métrique est de mesurer les caractéristiques similaires *Feature SIMilarity*. Cette mesure est basée sur deux éléments : la congruence de phase  $PC$  et la magnitude du gradient  $GM$ . Ces deux éléments sont complémentaires dans cette métrique.

D'après des études psychologiques le système visuel humain répond fortement aux points dans une image où les composantes de phase sont les plus élevées. De ce fait l'auteur, au lieu de chercher des changements d'intensité brusques dans l'image, passe par le domaine de Fourier afin de détecter des points dans la composante de phase.

La congruence de phase n'est pas sensible aux variations de contraste, c'est pour cela qu'elle est associée au calcul du module d'un gradient (GM). Le gradient est calculé en utilisant l'un des filtres de détection de contour comme Sobel, Prewitt ou encore Scharr.

L'équation 2.14 représente les contributions des deux composants de la métrique, avec  $\alpha$  et  $\beta$  deux paramètres permettant d'ajuster l'influence entre  $PC$  et  $GM$ .

$$S_L(x) = [S_{PC}(x)]^\alpha \cdot [S_G(x)]^\beta \quad (2.14)$$

Finalement, l'équation 2.15 présente la métrique FSIM entre les images  $I_1$  et  $I_2$ . Avec l'ajout de  $PC_m(x)$  où  $PC_m(x) = \max(PC_1(x), PC_2(x))$ ;  $PC_m(x)$  permet de pondérer l'importance de  $S_L(x)$  et  $\Omega$  correspond à tout le domaine spatial.

$$FSIM = \frac{\sum_{x \in \Omega} S_L(x) \cdot PC_m(x)}{\sum_{x \in \Omega} PC_m(x)} \quad (2.15)$$

FSIM est généralement appliqué aux images codées en niveau de gris, cependant l'auteur a étendu cette métrique dans le domaine de la couleur avec la métrique FSIMc où "c" correspond à la variante destinée aux images couleurs. L'auteur convertit alors l'image RGB dans l'espace YIQ. Le calcul de  $PC$  et  $GM$  se fait sur le canal correspondant à la luminance  $Y$  et calcule ensuite la similarité sur les canaux chromatiques  $I$  et  $Q$ . L'équation 2.16 présente le calcul de FSIMc, où  $S_C(x)$  correspond à la similarité de chrominance avec  $S_C(x) = S_I(x) \cdot S_Q(x)$ .

$$FSIMc = \frac{\sum_{x \in \Omega} S_L(x) \cdot [S_C(x)]^\lambda \cdot PC_m(x)}{\sum_{x \in \Omega} PC_m(x)} \quad (2.16)$$

Avec  $\lambda$  un paramètre utilisé pour moduler l'influence des canaux de chrominance.

De ce fait nous utilisons la métrique FSIMc pour évaluer les images couleurs, et la métrique FSIM pour les images en niveaux de gris.

### 2.2.2/ LES MÉTRIQUES SANS-RÉFÉRENCE

#### BRISQUE [130]

Cette métrique, pour *Blind/Referenceless Image Spatial Quality Evaluator*, est une métrique sans référence pour l'évaluation de la qualité des images. Son principe est de déduire un score de qualité en utilisant des méthodes d'apprentissage, et plus précisément de modèle SVR. Le SVR correspond à *Support Vector Regression* [52], de la famille des SVM. Pour la construction du modèle, des attributs sont extraits de la base d'images LIVE [77] de références naturelles associées à des images dégradées. L'adjectif *Naturelles*, ici, fait référence à des images capturées directement sans traitement post-capture. Cette base de données comporte aussi des évaluations humaines des images, BRISQUE se servant des notations humaines lors de son apprentissage.

Des attributs sont extraits sur ces images afin de créer un modèle de décision. Ces mêmes attributs sont utilisés lors de la phase de décision, qui correspond alors à la phase de notation BRISQUE. Le principe de la méthode d'apprentissage est d'extraire au total 18 attributs sur une image de référence et sur cette même image dont la taille est réduite d'un facteur de 2 dans les deux directions. Cette extraction d'attributs permet alors de créer un vecteur d'apprentissage de 36 éléments.

Les attributs correspondent, selon les auteurs, à des statistiques représentant au mieux la perception humaine. Le coefficient de base au calcul des attributs est le *MSCN-Mean Substracted Contrast Normalization*, qui correspond alors à la normalisation d'un pixel en lui soustrayant la moyenne locale, et en le divisant par l'écart type local. Le calcul du MSCN est détaillé dans l'équation 2.17.

$$\hat{I}(i, j) = \frac{I(i, j) - \mu(i, j)}{\sigma(i, j) + C} \quad (2.17)$$

Avec  $(i, j)$  les coordonnées spatiales du pixel dans l'image,  $C$  une constante qui permet la stabilité lorsque le dénominateur tend vers 0, et  $\mu(i, j)$  et  $\sigma(i, j)$  correspondent respectivement à la moyenne locale, calculée selon l'équation 2.18 et à l'écart type local, calculé selon l'équation 2.19, où  $w_{k,l}$  est une fonction symétrique circulaire Gaussienne, et  $K = L = 3$ .

$$\mu(i, j) = \sum_{k=-K}^K \sum_{l=-L}^L w_{k,l} I_{k,l}(i, j) \quad (2.18)$$

$$\sigma(i, j) = \sqrt{\sum_{k=-K}^K \sum_{l=-L}^L w_{k,l} (I_{k,l}(i, j) - \mu(i, j))^2} \quad (2.19)$$

Ensuite les attributs sont construits en associant le calcul de MSCN avec le voisinage direct de pixels (représenté sur la figure 2.17).

Les attributs correspondent aux paires de pixels, associés localement,  $H$ ,  $V$ ,  $D1$  et  $D2$ . Ces associations sont respectivement pour Horizontal, Vertical, Diagonale 1 et Diagonale 2. Les équations 2.20 présentent les calculs des différents éléments pour la construction des attributs.

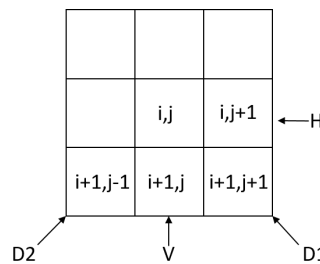


FIGURE 2.17 – Représentation du voisinage et des associations de pixels pour le calcul des attributs nécessaires à l'apprentissage de BRISQUE

$$\begin{aligned}
 H(i, j) &= \hat{I}(i, j) \hat{I}(i, j + 1) \\
 V(i, j) &= \hat{I}(i, j) \hat{I}(i + 1, j) \\
 D1(i, j) &= \hat{I}(i, j) \hat{I}(i + 1, j + 1) \\
 D2(i, j) &= \hat{I}(i, j) \hat{I}(i + 1, j - 1)
 \end{aligned}
 \tag{2.20}$$

Ensuite, les 18 attributs sont calculés en faisant correspondre les coefficients calculés dans les équations 2.20 à une distribution gaussienne asymétrique généralisée AGGD, et les attributs sont alors calculés en suivant plusieurs orientations et formes différentes. La description complète de ces attributs est détaillée dans [130].

Selon les auteurs, la métrique est fortement corrélée avec le jugement visuel humain ; néanmoins elle est tout de même limitée à la distorsion dans laquelle l'apprentissage a été réalisé, c'est pourquoi ils ont ensuite créé une variante appelé NIQE [141].

### NIQE [141]

Cette métrique, pour *Natural Image Quality Estimator*, a été créée à la suite de BRISQUE par les mêmes auteurs. Elle est basée sur des attributs extraits d'une grande base d'images de bonne qualité mais cette fois sans apprentissage. Contrairement à BRISQUE cette métrique n'est pas limitée à une sorte de déformation précise (déformations disponibles dans la base LIVE [77]).

Le principe de cette méthode est de mesurer la distance entre des modèles gaussiens multivariés MVGs provenant de l'image de test avec des MVGs extraits de bases d'images naturelles de bonne qualité. Tout comme BRISQUE, l'adjectif *Naturelles* fait ici référence aux images en sortie du capteur qui n'ont subi aucun traitement.

Plus cette distance est faible, plus la qualité de l'image est jugée bonne.

La création des MVGs est réalisée à partir des mêmes vecteurs d'attributs de la méthode BRISQUE, ces vecteurs sont calculés dans le but d'extraire les statistiques naturelles de la scène NSS. Ces statistiques NSS correspondent aux attributs MSCN de la métrique BRISQUE, équation 2.17. En effet d'après les auteurs, ce sont ces coefficients qui permettent le mieux de représenter le jugement visuel humain.

La première étape consiste à calculer les MVGs dans l'image. Pour cela l'image est divisée en blocs. Les blocs sont alors sélectionnés en fonction de leur intérêt : l'homme a la capacité de faire abstraction des zones hors-champs pour émettre un jugement global de l'image, de ce fait il est nécessaire que la métrique ne prenne pas en compte ces mêmes zones. Les zones hors-

champs correspondent aux zones qui comportent beaucoup de flou. Pour cette sélection, les auteurs calculent l'écart type moyen de chaque bloc, ce qui donne une information sommaire de l'acutance  $\delta$  du bloc (équation 2.21).

$$\delta(b) = \sum_{(i,j) \in \text{patche}} \sigma(i, j) \quad (2.21)$$

Une fois les blocs sélectionnés, les coefficients NSS sont calculés de la même manière que pour la métrique BRISQUE dans l'équation 2.17.

Un modèle gaussien multivarié est alors créé à partir des coefficients NSS. Ce modèle est ensuite utilisé dans la dernière étape, équation 2.22, pour calculer la "qualité" de l'image de test. Cette qualité de l'image de test est exprimée comme la distance entre le MVG extrait des images de bonne qualité avec le MVG de cette image de test.

$$D(v_1, v_2, \Sigma_1, \Sigma_2) = \sqrt{\left( (v_1 - v_2)^T \left( \frac{\Sigma_1 + \Sigma_2}{2} \right)^{-1} (v_1 - v_2) \right)} \quad (2.22)$$

Avec  $v_1, v_2$  et  $\Sigma_1, \Sigma_2$  les moyennes des vecteurs et covariances des matrices correspondant respectivement au modèle MVG naturel et au modèle MVG de l'image dégradée.

L'avantage de cette métrique, selon les auteurs, est qu'il n'est pas nécessaire de faire une correspondance entre des bases d'images dont la qualité a été annotée par des experts humains avant de faire un apprentissage sur la globalité des bases. Selon les auteurs, cette métrique est meilleure que le PSNR et que le SSIM.

### CPBD [111, 118]

La métrique CPBD pour *Cumulative Proportion of Blur Detection* permet de quantifier l'acutance de l'image. Plus la valeur de CPBD est grande plus l'image comporte de détails, ces détails correspondant alors aux contours présents.

Tout d'abord, un gradient de Sobel est appliqué sur toute l'image. Cette image de gradient est alors divisée en blocs de  $64 \times 64$  pixels. La première étape consiste à caractériser ces blocs en fonction de la quantité de contours qu'ils contiennent. Le nombre de pixels qui composent les contours sont comptés et comparés à un seuil  $T$ , en fonction du total  $N$  de pixels le bloc est alors caractérisé en *avec*,  $N > T$ , ou *sans*,  $N < T$ , contours. Les blocs notés comme "sans contour" ne sont pas traités par la suite. Ensuite le contraste  $C$  du bloc (avec contour) est calculé, le contraste correspond alors à la valeur maximale de la luminance  $I_{max}$  moins la valeur minimale  $I_{min}$ . Cette valeur de contraste  $C$  permet alors d'avoir l'information  $w_{JNB}$  (équation 2.23).  $w_{JNB}$  correspond à la largeur du contour calculée selon JNB *Just Noticeable Blur* [100].

$$w_{JNB} = \begin{cases} 5, & \text{si } C \leq 50 \\ 3, & \text{si } C \geq 51 \end{cases} \quad (2.23)$$

Pour chaque contour noté  $e_i$  la largeur de celui-ci  $w(e_i)$  est calculée en utilisant la méthode présente dans [62]. Il s'agit de la différence entre la position de début et de fin du contour.

La probabilité de détection de flou est alors calculée en suivant l'équation 2.24. Avec  $w_{JNB}(e_i)$  la largeur du contours calculée sur JNB qui dépend du contraste local  $C$  (équation 2.23). La valeur de  $\beta$  est choisie pour augmenter la correspondance de l'équation 2.24 avec la fonction psychométrique pour une distorsion donnée [100].

$$P_{BLUR}(e_i) = 1 - \exp\left(-\left|\frac{w(e_i)}{w_{JNB}(e_i)}\right|^\beta\right) \quad (2.24)$$

Enfin la métrique CPBD, équation 2.25, correspond au pourcentage de contours pour lesquels la probabilité de détection de flou est inférieure à la probabilité de flou "JNB", et donc cela correspond au pourcentage pour lequel le flou n'est pas susceptible de se produire.

$$\begin{aligned} CPBD &= P(P_{BLUR} \leq P_{JNB}) \\ &= \sum_{P_{BLUR}=0}^{P_{BLUR}=P_{JNB}} P(P_{BLUR}) \end{aligned} \quad (2.25)$$

Où  $P(P_{BLUR})$  est la probabilité de la fonction de distribution pour  $P_{BLUR}$ .

Selon l'auteur la métrique CPBD a une meilleure corrélation avec le jugement subjectif humain que d'autres méthodes de mesure d'acutance.

### PSI [136]

Cette métrique, PSI est pour *Perceptual Sharpness Index*, développée par Feichtenhofer et al., est une métrique qui mesure le niveau d'acutance perçue dans les images. Cette métrique a été créée afin de se rapprocher au mieux du jugement humain. Elle travaille directement dans le domaine spatial ce qui permet à cette métrique de consommer très peu de ressource. Cette métrique est applicable dans un grand nombre de domaines comme l'amélioration de la qualité de l'image, la déconvolution, la Super Résolution, et peut aussi être facilement embarquée pour faire de la mise au point automatique.

Tout d'abord un gradient de Sobel est réalisé sur l'ensemble de l'image. Ce gradient a pour but de faire ressortir les zones qui contiennent des contours. Cette sélection de zone est réalisée avec l'application d'un seuil  $T$  sur le gradient mesuré. L'équation 2.26 présente le seuil adaptatif  $T$  en fonction de la moyenne du module  $\bar{G}$  du gradient  $G$  dans l'image,  $\alpha$  est un coefficient permettant d'améliorer les performances de la sélection.

$$T = \alpha \bar{G} \quad (2.26)$$

Ensuite un gradient est calculé sur les directions  $(x, y)$  avec la méthode de la différence finie. Ce gradient n'est calculé que dans les zones sélectionnées avec le gradient de Sobel. L'angle  $\phi(x)$  de ce gradient composé de  $(I_x, I_y)$  est calculé en suivant l'équation 2.27.

$$\phi(x) = \tan^{-1}\left(\frac{I_y(x)}{I_x(x)}\right) \quad (2.27)$$

L'angle est utilisé pour calculer la largeur du contour, équation 2.28, où  $w_{up}(x)$  et  $w_{down}(x)$  sont respectivement les distances entre le contour et les maxima et minima locaux de ce contour.

$$w(x) = \frac{w_{up}(x) + w_{down}(x)}{\cos(\phi(x))} \quad (2.28)$$

Cette largeur  $w(x)$  est alors utilisée pour le calcul de la pente du contour (équation 2.29).

$$m(x) = \frac{I_{max}(x) - I_{min}(x)}{w(x)} \quad (2.29)$$

Une fois la pente  $m(x)$  et la largeur  $w(x)$  calculées, la largeur finale  $w_{PSI}(x)$  est calculée (équation 2.30). Cette largeur est une normalisation de la largeur  $w(x)$  précédemment calculée en fonction de la largeur "Just Noticeable Blur"  $w_{JNB}$ .

$$w_{PSI}(x) = \begin{cases} w(x) - m(x), & \text{si } w(x) \geq w_{JNB} \\ w(x), & \text{ailleurs} \end{cases} \quad (2.30)$$

Selon Ferzli et al., cette largeur,  $w_{PSI}$ , représente une valeur d'acutance proche de celle perçue par le système visuel humain. Enfin l'image est divisée en blocs de  $32 \times 32$  pixels dans lesquels les largeurs des contours  $w_{PSI}$  sont calculées. La valeur finale  $PSI$  est calculée en ne prenant en compte seulement un pourcentage des blocs de l'image pour le calcul final de la moyenne des  $w_{PSI}$ . Ce pourcentage apporterait plus de similitude avec le système visuel humain qui écarte les zones d'arrière-plan qui comportent plus de flou [103]. L'inverse de cette moyenne représente la valeur de  $PSI$ , ce qui lui permet d'avoir une valeur comprise entre 0 et 1. L'auteur, dans l'évaluation de sa nouvelle métrique, fait le rapprochement de  $PSI$  avec des ensembles d'images qui ont été annotées par des humains. Il arrive à montrer que sa métrique, bien qu'automatique, se rapproche énormément de l'évaluation humaine.

### 2.2.3/ DISCUSSION

Cet état de l'art des métriques de l'image nous montre qu'il existe un grand nombre de métriques différentes. Dans la plupart des cas, les métriques jugent la qualité d'une image en calculant la proportion de contraste, ou par la proportion de contours francs. Toutes ces métriques, les meilleures selon leurs auteurs, doivent être étudiées plus profondément. Une étude est nécessaire pour voir si les résultats de ces métriques sont cohérents avec le jugement humain, dans le cadre de notre problématique, pour les traitements ou familles de traitements que nous avons abordés. Cette étude sera détaillée dans la section 3.1.

## 2.3/ L'IMAGERIE COMPUTATIONNELLE POUR L'AMÉLIORATION DE LA QUALITÉ DE L'IMAGE

Les techniques et les algorithmes correspondants permettant de corriger et d'améliorer des images sont regroupées dans le domaine de l'imagerie computationnelle. Dans ce domaine de recherche, les traitements peuvent être appliqués directement dans le flux d'acquisition de l'image, ou à partir d'images sauvegardées.

L'efficacité de ces techniques s'améliore constamment avec l'évolution conjointe de la puissance des plateformes de calcul, comme les GPU, FPGA ou processeurs multi-coeurs, et par le

développement et l'optimisation d'algorithmes de traitement d'images.

Des chercheurs ont proposé une classification en sous catégories de l'imagerie computationnelle, comme Nayar et al. [120] dans un rapport où l'auteur décrit quatre techniques de l'imagerie computationnelle. Ce découpage, présenté dans la figure 2.18, montre les quatre sous-domaines de l'imagerie computationnelle, et un recouvrement partiel d'un sous-domaine à un autre. Ce recouvrement montre les limites de séparation des sous-domaines, car il peut arriver des situations où toutes les techniques sont réunies. Cette figure illustre donc bien la possibilité de définir et de mettre en relation les termes photographie digitale, photographie computationnelle, caméra d'imagerie numérique et capteurs d'imagerie computationnelle. Ainsi, le domaine des caméras de calcul possède une intersection non nulle avec les domaines de la photographie computationnelle et des capteurs d'imagerie computationnelle.

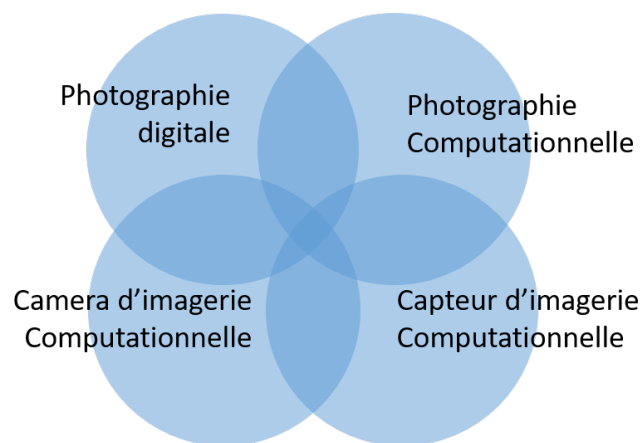


FIGURE 2.18 – Les 4 subdivisions de l'imagerie computationnelle selon [120], ces subdivisions se recoupent car elles englobent parfois les mêmes méthodes.

Ces techniques sont classées en prenant en compte le nombre d'images utilisées et l'endroit où est réalisé le traitement, à savoir dans la chaîne de traitement. Ces différentes techniques sont explicitées ci-dessous :

- **La Photographie digitale** correspond aux traitements appliqués sur des images capturées en vue de les transformer en "meilleures" images, c'est à dire que l'image soit plus facile à être traitée et à être utilisée par la suite. Cette amélioration peut être esthétique (l'amélioration de la qualité), peut-être d'ordre pratique (compression pour faciliter le stockage), mais aussi pour la réalisation d'interpolation, de filtrage 2-dimensions. L'important est qu'une seule et unique image soit utilisée avec un traitement numérique.
- **La Photographie computationnelle** correspond aux traitements qui nécessitent un ensemble d'images pour créer une ou plusieurs nouvelles images. Par exemple la technique du "mosaïcing" permet d'assembler un ensemble d'images afin d'en créer une plus grande. Cette technique de mosaïquage permet de créer des panoramas ou encore de réaliser des images satellites de taille importante. Nous pouvons aussi citer des techniques d'augmentation de la dynamique des images (HDR) nécessitent plusieurs images basses dynamique pour concevoir une image haute dynamique. Ainsi que des techniques de Super résolution qui permettent de créer une image haute résolution à partir d'images basse résolution.

- **La camera d'imagerie computationnelle**, communément appelé "*smart camera*" [108], correspond à un ensemble comprenant le système d'acquisition et un processeur effectuant un traitement spécifique. Cet ensemble est composé d'une optique non-conventionnelle, dont le modèle optique est connu, et d'un traitement effectué à l'intérieur même du système d'acquisition, traitement matériel ou logiciel. Ce traitement est prévu pour décoder l'image qui résulte de l'optique et est prévu pour produire une nouvelle image contenant de nouvelles informations.  
Dans les années 1990, Cathey et al. [34] ont proposé un système hybride formé d'un système optique "cubic phase plate" et d'un processeur dans le but d'étendre la profondeur de champ. Plus récemment, Lapray et al. [150] ont conçu une plateforme permettant de construire du contenu HDR à partir d'un capteur d'images et d'une carte de développement embarquant un FPGA.
- **Le capteur d'imagerie computationnelle** est quant à lui l'association d'un capteur et d'un traitement qui est embarqué à l'intérieur même de ce capteur. Cette association permet donc de transformer un pixel standard en un pixel dit "intelligent". Des exemples de traitements comme dans [48] où les auteurs créent un capteur avec traitement programmable.  
Plusieurs équipes de recherche ont développé des capteurs capables d'effectuer une détection d'objet ainsi qu'un prétraitement d'image [23, 31, 32].

Les différents exemples d'imagerie computationnelle présentés précédemment permettent d'aller au-delà des limitations matérielles. Cependant, il existe des correspondances matérielles de systèmes de capture spécialement conçus pour la capture d'image qui ne nécessitent aucun "calcul" ou phase de décodage.

Par exemple, une diminution du temps d'exposition permet de réduire le flou de bougé [81], et la fermeture du diaphragme permet d'étendre la profondeur de champ [106]. Un dernier exemple matériel [95] correspond à l'utilisation de masque sténopé "pinhole" pour l'acquisition de l'information de profondeur de champs lumineux pour la création d'une caméra plénoptique. Ce type de caméra permet d'acquérir toutes les informations des différents plans focaux ce qui donne la possibilité de choisir après la capture le plan focal désiré.

De plus, il est possible d'ajouter des filtres optiques entre l'objectif et le capteur afin de ne laisser passer que certaines longueurs d'onde, transformant un système d'acquisition standard en un système d'acquisition spectral ou multi-spectral.

La figure 2.19, issue des travaux de Cossairt [135], met en avant des comparaisons entre des solutions de caméras computationnelles et l'équivalent en caméras standards. Les deux techniques présentées sont réalisées pour obtenir le même résultat en terme d'image finale. Cependant l'auteur met en avant le fait que l'imagerie computationnelle nécessite un décodage qui, selon lui, amplifie le bruit, contrairement à l'imagerie standard qui consiste à capturer directement l'image sans avoir à faire de traitements.

Étant donné les contraintes liées au projet EXIST, nous allons dans la suite de cette section nous intéresser plus particulièrement à deux parties de l'imagerie computationnelle : La photographie digitale et la photographie computationnelle. La technique de déconvolution sera tout d'abord approfondie, dans la section 2.3.1 une technique permettant de corriger un grand nombre de problèmes de flou, de distorsion, et d'aberrations chromatiques sur une image.

Ensuite, dans la section 2.3.2, nous approfondirons les différentes techniques de photographie digitale et computationnelle pour la génération d'images haute résolution en utilisant une seule ou plusieurs images basse résolution.



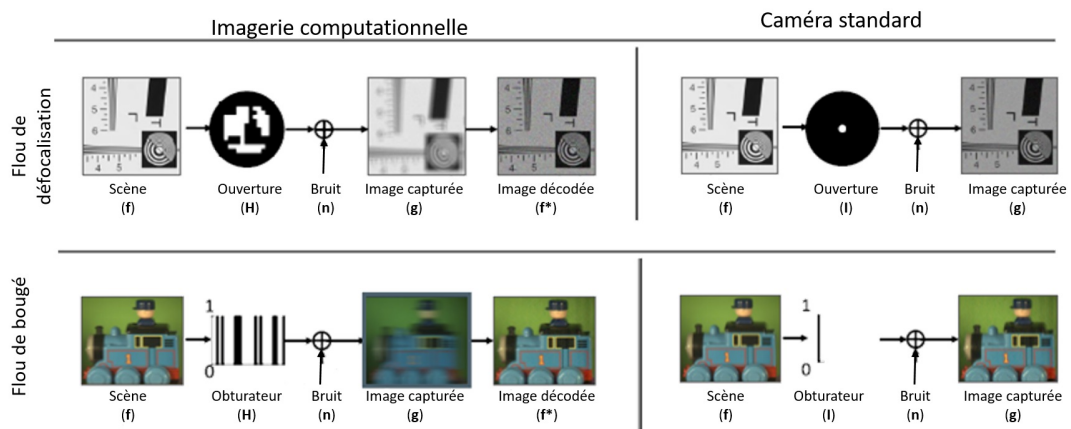


FIGURE 2.19 – Différence entre Imagerie Computationnelle (*gauche*) et Imagerie standard (*droite*) pour avoir le même résultat concernant la réduction de flou de défocalisation et de bougé. Les images proviennent respectivement de [106, 81].

### 2.3.1/ LA DÉCONVOLUTION, UNE TECHNIQUE DE DÉFLOUTAGE

Le but de la déconvolution est de restaurer une image qui a été floutée. Le processus de flou peut être dû à divers facteurs, comme nous l'avons rappelé dans la première partie de ce chapitre, notamment les caractéristiques inhérentes à l'équipement optique et les éventuels défauts de celui-ci, mais aussi, par des mouvements lors de la prise de vue [86].

Il est possible de trouver dans la littérature un grand nombre de modélisations mathématiques de la formation de flou, comme dans l'équation 2.31, présentée dans les travaux de Yuan [97]. Le flou est ici modélisé mathématiquement comme une opération de convolution à deux dimensions entre l'image d'origine "nette"  $I$  et un noyau appelé  $PSF$  (pour Point Spread Function ou fonction d'étalement du point).

La  $PSF$  représente la réponse d'un système d'imagerie à une source ponctuelle. Cette fonction peut être considérée comme une distribution de probabilité décrivant les destinations potentielles des photons provenant d'un point donné (ce qui se traduit concrètement par un étalement du point dans l'image résultante). Cette fonction de déformation optique est représentée dans la figure 2.20.

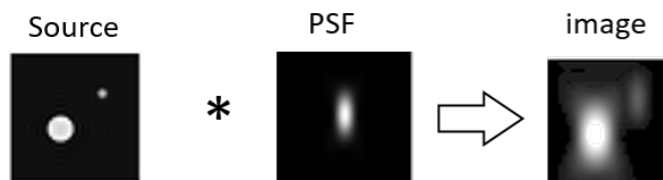


FIGURE 2.20 – Exemple de convolution entre une source lumineuse (*gauche*) et la  $PSF$  (*centre*), l'image résultante est floue (*droite*)

Une  $PSF$  est spécifique à chaque appareil d'imagerie. Cette fonction d'étalement peut être décrite mathématiquement à l'aide de modèles du chemin optique, ou plus généralement déduite de manière empirique par la mesure de l'étalement d'une source ponctuelle sur un fond noir.

L'image dégradée  $B$  est donc la convolution  $\otimes$  entre l'image d'origine  $I$  et la  $PSF$ . Cette modélisation comprend aussi la contribution d'un bruit  $N$ , ce bruit est parfois représenté comme

une addition.

$$B = I \otimes PSF + N \quad (2.31)$$

La déconvolution, ou le défloutage, a pour objectif d'éliminer les effets néfastes de la  $PSF$  afin de reconstruire l'image d'origine, non déformée et nette,  $I$  à partir de l'image convoluée et bruitée  $B$  et de la fonction de flou  $PSF$ .

De nombreuses recherches ont été menées pour reconstruire l'image  $I$ . Toutefois une transformée inverse du processus de génération de flou n'est généralement pas possible car les données utilisées sont mathématiquement incomplètes. Les données de l'observation permettent de trouver une infinité de combinaisons possibles ( $I, PSF$ ). Le problème est insoluble, cela correspond à un problème mal posé [1] car il n'existe pas de solution unique. Pour cette raison, des techniques d'estimation statistique sont généralement nécessaires. Ces techniques tentent d'estimer les données avec un maximum de vraisemblance compte tenu de l'observation  $B$  et du  $PSF$ .

La déconvolution est un thème de recherche très important car ses applications sont très variées : astronomie, télédétection, imagerie médicale, photographie, etc.

Par exemple, en Avril 1990, le télescope Hubble est mis en orbite, et quelques mois plus tard les premières images sont enfin produites. À la surprise des scientifiques et des ingénieurs les images produites sont floues. L'origine du problème est un défaut d'un des miroirs du satellite. Après une commission d'enquête, il s'avère que le miroir primaire est trop plat à sa périphérie, faisant d'Hubble un satellite ne réussissant pas à générer des images meilleures qu'un grand télescope terrestre. Cette erreur était due à un mauvais étalonnage de l'instrument de vérification de courbure. C'est en 1993 que la première mission de sauvetage STS-61 a été réalisée. Le défaut du miroir étant homogène il est prévu d'ajouter un élément optique qui a la même anomalie, mais inversée.

La figure 2.21 présente deux images générées par Hubble, une image prise lors des premiers tests montrant un flou (à gauche), et une image prise en 1993 après la mission STS-61.



FIGURE 2.21 – Exemple d'une image floue prise par Hubble après son lancement (*gauche*), et une image prise après la mission de réparation de son miroir STS-61 (*droite*)

Le temps de mettre en place cette mission de sauvetage, de nombreux chercheurs ont tenté de corriger ce flou en utilisant des méthodes de déconvolution.

En août 1990, seulement deux mois après les premiers résultats, un workshop [24] est organisé

afin de mettre en avant des méthodes d'imagerie computationnelle permettant de restaurer les images provenant de Hubble. En 1990, Robert Gonsalves [191], un ingénieur, a ainsi aidé à corriger les images du satellite avec de nouvelles méthodes de déconvolution.

Les techniques de déconvolution sont décomposées en deux familles distinctes :

- **La déconvolution non-aveugle**, présentée section 2.3.1.1, est une technique de déconvolution pour laquelle on suppose que l'image dégradée  $B$  et le noyau  $PSF$  sont tous deux connus.
- **La déconvolution aveugle**, présentée section 2.3.1.2, est, a contrario, une technique de correction de l'image où ni l'image source  $I$ , ni la fonction de transfert  $PSF$  ne sont connues. Dans la majeure partie des cas, ces méthodes tentent tout d'abord de retrouver ou d'estimer la  $PSF$  pour ensuite appliquer les techniques de convolution non-aveugle.

La déconvolution, aveugle ou non, peut être réalisée dans le domaine spatial tout comme le domaine fréquentiel.

### 2.3.1.1/ DÉCONVOLUTION NON-AVEUGLE

#### Le filtrage inverse [195, 93]

Le flou est ici considéré comme un filtre passe-bas, de ce fait, dans cette approche, un filtre passe-haut est utilisé pour reconstruire l'image d'origine.

La figure 2.22 montre les opérations de *floutage* et *défloutage* que l'image subit lors d'un filtrage direct inverse, où  $g$  est l'image floutée et bruitée provenant de la convolution entre l'image nette  $f$  et le filtre de floutage  $h$ . Du bruit  $n$  est ensuite ajouté à ce produit de convolution. De plus le filtrage inverse est directement réalisé en utilisant l'inverse du filtre de floutage  $h_{inv}$ .

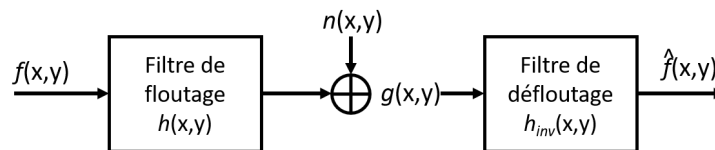


FIGURE 2.22 – Technique du filtrage inverse directe présentée dans [149]

Il est alors possible de formaliser les deux parties de ce traitement : la fonction de flou (équation 2.32 et la fonction de défloutage inverse (équation 2.33).

$$g(x, y) = f(x, y) \otimes h(x, y) + n \quad (2.32)$$

$$\hat{f}(x, y) = g(x, y) \otimes h_{inv}(x, y) \quad (2.33)$$

L'objectif est de déterminer la fonction  $g(x, y)$  afin de calculer l'estimée  $\hat{f}(x, y)$ .

Ces équations peuvent être représentées dans le domaine fréquentiel :

$$G(u, v) = F(u, v).H(u, v) + N \quad (2.34)$$

Si  $N$  est considéré nul :

$$\hat{F}(u, v) = G(u, v).H_{inv}(u, v) = \frac{G(u, v)}{H(u, v)} \quad (2.35)$$

Ce qui donne, si  $N$  est non nul :

$$\hat{F}(u, v) = \frac{F(u, v).H(u, v) + N}{H(u, v)} = F(u, v) + \frac{N}{H(u, v)} \quad (2.36)$$

Où  $N$  représente la densité spectrale de bruit. Cette méthode donne de bons résultats lorsqu'il n'y a pas, ou très peu, de bruit. Lorsqu'il n'y a pas de bruit,  $N$  est considéré comme nul et l'image estimée est similaire à l'image source 2.37.

$$\hat{F}(u, v) = F(u, v) \quad (2.37)$$

Si la fonction de dégradation  $H(u, v)$  est faible, alors le bruit est dominant sur l'image.

Malheureusement, étant donné que le filtre inverse représente un filtre passe haut, la réponse du filtrage inverse est très mauvaise en présence du bruit. Car de par sa nature, le filtre passe-haut amplifie les hautes fréquences formant le bruit.

Des méthodes existent afin d'essayer de contrer l'apparition et l'amplification de bruits [149] : techniques de seuillage itératifs. Ces méthodes doivent trouver un compromis entre le défloutage et le débruitage.

### Wiener [6, 195]

Dans les années 1940, Norbert Wiener a introduit une technique de déconvolution qui a la possibilité de reconstruire une image dont la fonction de déformation, modélisée par une PSF, est connue. Cette méthode est basée sur le principe du filtrage inverse, cependant une fonction de minimisation du bruit est ajoutée. L'avantage de cette méthode est qu'elle permet à la fois la suppression du bruit et de l'effet de flou. Le schéma explicitant le fonctionnement de ce filtre est présentée dans la figure 2.23.

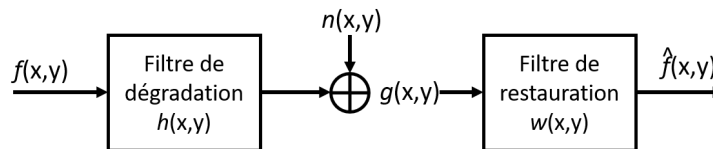


FIGURE 2.23 – Diagramme en bloc représentant la technique du filtrage de Wiener

L'image dégradée  $g(x, y)$  et l'image estimée  $\hat{f}$  sont décrites dans les équations suivantes :

$$g(x, y) = f(x, y) \otimes h(x, y) + n(x, y) \quad (2.38)$$

$$\hat{f}(x, y) = w(x, y) \otimes g(x, y) \quad (2.39)$$

Le but est de trouver une estimation  $\hat{f}$  non dégradée et débruitée de la version d'origine  $f$  telle que l'erreur quadratique moyenne EQM entre ces deux images (équation 2.40) soit la plus petite

possible. Le filtre de Wiener est largement utilisé dans les applications de déconvolution d'images car le spectre de fréquences peut facilement être calculé pour la majorité des images.

$$e^2 = E \left\{ (f - \hat{f})^2 \right\} \quad (2.40)$$

Il est nécessaire de faire l'hypothèse que les signaux  $f(x, y)$  et  $n(x, y)$  sont des processus aléatoires périodiques et de densité spectrale de puissance  $S_N(u, v)$  et  $S_F(u, v)$  connues.

L'expression du filtre de Wiener représenté dans le domaine fréquentiel est présentée dans l'équation 2.41.

$$\hat{F}(u, v) = W(u, v) \cdot G(u, v) \quad (2.41)$$

Avec  $\hat{F}(u, v)$  l'image estimée à partir du filtre de Wiener  $W(u, v)$  et de l'image dégradée  $G(u, v)$ .

$$W(u, v) = \frac{\bar{H}(u, v) \cdot S_f(u, v)}{|H(u, v)|^2 \cdot S_f(u, v) + S_N(u, v)} \quad (2.42)$$

Où  $\bar{H}(u, v)$  est le conjugué de  $H(u, v)$ , sachant que  $|H|^2 = \bar{H}H$  il est possible de mettre cette équation sous la forme suivante :

$$W(u, v) = \frac{1}{H(u, v)} \left[ \frac{|H(u, v)|^2}{|H(u, v)|^2 + \frac{S_n(u, v)}{S_f(u, v)}} \right] \quad (2.43)$$

À partir de l'équation 2.43 il est possible de voir que si la densité spectrale du bruit  $S_N(u, v)$  est nulle alors le filtre de Wiener correspond au filtre inverse de la fonction de dégradation  $H(u, v)$ .

La solution pour que l'estimation de l'erreur quadratique moyenne EQM soit la plus faible possible, dans le domaine fréquentiel, est présentée dans l'équation 2.44 :

$$\hat{F}(u, v) = \left[ \frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + \frac{S_n(u, v)}{S_f(u, v)}} \right] G(u, v) \quad (2.44)$$

Avec  $H(u, v)$  la fonction de dégradation,  $S_n(u, v)$  la densité spectrale du bruit,  $S_f(u, v)$  la densité spectrale de l'image d'origine et  $G(u, v)$  est l'image dégradée.

### Richardson-Lucy [9, 10]

Il s'agit d'un algorithme de déconvolution itératif qui a été proposé indépendamment par Lucy [10] et Richardson [9].

L'algorithme de Richardson-Lucy (LRA) est une méthode importante pour la suppression du flou sur les images. Faisant partie de la déconvolution non-aveugle il est nécessaire de connaître la fonction de déformation PSF. LRA part du principe que la statistique de l'image suit la loi de Poisson.

Le but de cet algorithme est d'inverser le flou ajouté à l'image et d'estimer l'image sans dégradation  $f$ . Pour chaque itération de l'algorithme, une version de l'image corrigée  $f_{(n+1)}$  est calculée en prenant en considération l'image estimée  $f_{(n)}$  au temps  $n$ , la fonction de déformation PSF  $H$  et l'image dégradée  $g$ .

$$f_{(n+1)} = f_{(n)} \left( \frac{g}{H \otimes f_{(n)}} \otimes H^{-1} \right) \quad (2.45)$$

Il faut considérer  $f_{(0)} = g$  pour la première itération. Cette méthode minimise la différence entre l'image floue et l'image prédite en fonction des statistiques de Poisson [132]. L'inconvénient de cet algorithme est l'apparition de signaux parasites sur les contours de l'image (ringing artefact) et le temps d'exécution dû aux itérations.

Étant une méthode itérative il est nécessaire de définir un critère d'arrêt. Ce critère est à paramétrer en fonction de l'utilisation de l'algorithme de déconvolution et c'est à l'utilisateur d'expérimenter l'algorithme afin de définir le moment où la qualité de l'image corrigée est convenable. Ce critère peut tout aussi bien être une limite de temps pour laquelle la déconvolution est trop longue. Il est par exemple possible d'avoir des contraintes de temps réel où il est nécessaire de toujours avoir le même temps d'exécution.

Cet algorithme a pu être implanté dans des cibles matérielles comme les GPU ou FPGA [86]. Par exemple Sims [86] a implanté une version de LRA en le modifiant un peu afin qu'il converge plus vite à l'image finale, et ceci en réduisant le nombre d'itérations.

$$f_{(n+1)} = f_{(n)} \left( \frac{g}{H \otimes f_{(n)}} \otimes H^{-1} \right)^\beta \quad (2.46)$$

$\beta$  a été ajouté par l'auteur afin d'accélérer la convergence de l'algorithme, et réduire le nombre d'itérations.

### 2.3.1.2/ DÉCONVOLUTION AVEUGLE

La plupart des travaux sur la déconvolution aveugle ont commencé au début des années 1970. La déconvolution aveugle est utilisée dans un grand nombre de domaines comme l'imagerie astronomique et l'imagerie médicale.

La déconvolution aveugle, comme l'explique Kundur [38], est une technique de déconvolution qui permet de retrouver une image nette en ne connaissant pas la fonction de déformation  $PSF$ . Le but est donc d'estimer les deux signaux convolués, à savoir l'image nette  $I$  et le noyau de convolution  $PSF$ , à partir d'une seule ou de plusieurs images dégradées  $B$ .

Dans la plus part des cas, pour des raisons de simplification, le bruit ajouté est négligé comme le montre l'équation 2.47.

$$B = I \otimes PSF \quad (2.47)$$

Cependant, même en négligeant le bruit, il existe une infinité de possibilités de combinaisons  $(I, PSF)$  dont le résultat de convolution donne l'image floue  $B$ , ce qui, selon Hadamard [1, 3], fait de la déconvolution un problème mal posé du fait que la solution de la déconvolution n'est pas forcément unique, peut être instable, et il est aussi possible qu'aucune solution n'existe.

En général, le but de la déconvolution aveugle est de tout d'abord faire une estimation de la  $PSF$  à partir d'une ou de plusieurs images, pour ensuite pouvoir utiliser les techniques de déconvolution non-aveugle.

L'estimation de la  $PSF$  peut se faire avec un grand nombre de techniques [185, 94, 29], de manière

itérative ou statique, avec une ou plusieurs images de la scène. Certaines techniques itératives vont nécessiter un grand nombre de tentatives afin de se rapprocher du véritable noyau  $PSF$  et d'améliorer leurs estimations.

D'autres techniques de l'état de l'art permettent d'extraire à partir des images des informations sur la création de la  $PSF$ , et de cette manière estimer le flou et de corriger sans itération les images. Plus l'estimation de la  $PSF$  est juste, plus la convolution inverse apportera de bons résultats, lorsque le bruit est faible.

L'estimation de la  $PSF$  n'est pas triviale, en effet il arrive que cette fonction de transfert ne soit pas constante et varie dans l'image. Il est donc nécessaire de rechercher et estimer plusieurs  $PSF$  différentes dans une seule image.

#### Les méthodes d'estimation de la $PSF$

Il est parfois possible d'extraire la  $PSF$  de manière directe dans l'image. Cette méthode consiste à extraire les informations de la  $PSF$  à partir d'une seule observation. Cependant cette technique peut être considérée comme étant à la limite de la déconvolution "aveugle". Le principe est d'observer un objet connu dans l'image [94].

En Astronomie, par exemple, il est possible assez rapidement d'extraire une  $PSF$  lorsqu'il est possible de capturer l'image dégradée d'une seule étoile. En effet les étoiles sont supposées être des sources de lumière ponctuelles "nettes", et sans étalement. Et ces observations représentent parfois des "taches" blanches sur fond noir, qui peuvent être assimilées à l'étalement du point lumineux diffusant la lumière dans des trajectoires différentes.

Ces estimations rapides prennent en compte des connaissances *a priori*. Par exemple dans le domaine de l'imagerie médicale par rayon X, il est possible de connaître certains repères lors de la prise de vue [29]. Ce procédé se répète avec l'imagerie satellitaire lorsqu'on capture une image connue, qui est considérée alors comme une cible.

Dans ses travaux, Joshi [94] a présenté une méthode d'estimation automatique de la  $PSF$  lorsque la scène peut être contrôlée et qu'il est possible d'introduire une mire. Joshi a conçu une mire spéciale qui peut être automatiquement détectée dans l'image. Connaissant la version nette de cette mire, il est ainsi possible d'estimer la  $PSF$ .

La méthode d'estimation de la  $PSF$  à partir de mires spéciales est utilisée par les fabricants de systèmes d'acquisition afin de pouvoir récupérer les informations sur les aberrations optiques, et de pouvoir par la suite corriger exactement l'image. Ces méthodes peuvent être vues comme un système de calibration.

D'autres méthodes, sans utiliser de mire, vont partir du principe qu'elles reconnaissent certains points dans l'image, par exemple des contours de certains objets. Ces méthodes sont souvent utilisées dans l'industrie en faisant l'hypothèse que l'estimation sur une scène donnée peut ensuite servir à déconvoluer tout un ensemble d'observations.

Les auteurs, Joshi et al. [94] et Chalmond et al. [29], se servent d'une estimation des contours afin de pouvoir estimer la  $PSF$ . Joshi a présenté un algorithme pour l'estimation de contours nets à partir d'images floues ; l'auteur estime que si il peut repérer où se trouvent les contours nets dans l'image, alors il peut retrouver la fonction  $PSF$ . Le principe de sa méthode est de prédire une version nette  $I_{pred.}$  de l'image floue  $B$ , à partir des contours détectés, et se sert ensuite du couple  $(I_{pred.}, B)$  pour calculer la fonction  $PSF$ . Cette méthode lui permet d'estimer une  $PSF$  à une échelle inférieure à



la résolution d'origine.

Par exemple, Boscaro et al. [182] ont mis en place une technique d'estimation de la *PSF* pour les problèmes de mise au point "out of focus". Cette méthode permet d'estimer la *PSF* de façon automatique, rapide et robuste afin d'être intégrée en temps réel. Leur méthode est basée sur l'utilisation de la transformée en cosinus discret DCT et la transformée de radon pour estimer le noyau PSF.

### 2.3.1.3/ DISCUSSION ET ORIENTATION

Un des objectifs du projet européen H2020-EXIST était de participer à l'élaboration d'une nouvelle caméra computationnelle très haute résolution 8k. Pour cette tâche, le rôle de notre laboratoire de recherche était d'étudier la possibilité d'un traitement embarqué permettant de faire de la correction d'images à l'aide de technique de déconvolution en temps réel.

Le but de cette déconvolution est d'améliorer les images très haute-résolution obtenues avec un capteur et un objectif créés spécialement pour le projet. Cette amélioration computationnelle permet de réduire le coût de fabrication de l'objectif.

Le fabricant de l'optique pour le projet nous a communiqué les *PSF*, ce qui permet d'utiliser la déconvolution non-aveugle.

La preuve de concept de cette déconvolution a été réalisée sur une cible GPU en utilisant la méthode du filtrage de Wiener. La cible GPU a été choisie car elle permet de réaliser un développement rapide avec une forte puissance de calcul. Une fois que cette méthode sera validée il sera possible d'étudier la possibilité d'intégrer une puce GPU dans une caméra computationnelle, ou voir s'il est possible de transposer le code pour fonctionner sur une cible de type FPGA.

Le travail de mise en place du traitement et de programmation GPU a été réalisé par un ingénieur de recherche financé par le projet H2020. Les résultats d'implantation de la déconvolution, sur différentes cibles (GPU et CPU), sont présentés dans la section 4.2.

### 2.3.2/ LES TECHNIQUES DE SUPER RÉOLUTION

L'évolution constante dans les technologies de l'image a provoqué l'émergence de caméras avec des pixels de plus en plus petits et des résolutions de plus en plus grandes. Ces nouvelles caméras permettent de capturer des détails de plus en plus fins. La nécessité d'avoir des contenus avec des résolutions de plus en plus grandes se fait ressentir dans un grand nombre de domaines où il est nécessaire d'avoir toujours une meilleure perception des détails. Ces hautes résolutions permettent d'améliorer les traitements et analyses effectués sur les images.

Le développement des capteurs Gigapixels capables de fournir de telles images hautement résolues a naturellement été mis en avant par les besoins de marchés clés tels que dans le domaine du médical, de la sécurité et de la télédiffusion. A ce jour, les caméras Ultra Haute Définition (UHD), capables de capturer du contenu 8k ( $8192 \times 4320$  pixels) sont d'ores et déjà commercialisées.

Néanmoins la fabrication de ce type de caméras, qui nécessitent des optiques très précises, coûte très cher. C'est le principal problème pour les applications réelles, ce qui limite l'utilisation des caméras UHD dans ces marchés de niche. De plus, l'augmentation de la résolution des systèmes d'acquisition limite aussi leurs performances en termes de vitesse. Cette augmentation de la résolution augmente directement le nombre de pixels ce qui a donc tendance à ralentir la vitesse de capture (le temps de récupération des pixels entre chaque image étant très grand), limiter le



nombre d'images stockées (pour un espace de stockage donné) et à ralentir aussi les traitements appliqués sur ces grandes images. Des ressources matérielles importantes (interfaces haute vitesse, processeurs puissants) sont nécessaires pour faire face à cette bande passante sans précédent, ce qui devient un élément bloquant pour la conception de systèmes temps réels.

Donc, dans le but de satisfaire la grande demande de contenu UHD, la Super Résolution (SR) est une solution alternative et abordable pour la construction de contenu UHD. La Super Résolution est une technique algorithmique permettant de créer du contenu très haute résolution (HR) (images ou vidéos) à partir d'une ou de plusieurs images basse résolution (LR), et ceci en augmentant le niveau de détail, présent dans les hautes fréquences de l'image, et en supprimant les dégradations introduites lors du processus de génération de l'image [114].

Ces techniques s'appuient sur l'utilisation d'optiques standards couplées à des caméras basse résolution (généralement inférieure à 2k), associées à un traitement de l'image permettant de convertir les images LRs en images de résolution supérieure (haute résolution jusqu'au 8k).

Les techniques de SR sont largement utilisées, et ce, dans des domaines différents :

- Image satellitaire,
- Vidéo surveillance,
- Amélioration de la qualité vidéo et restauration,
- Microscopie,
- Mosaïquage digitale,
- Imagerie médicale.

Depuis les premiers travaux novateurs de Tsai et Huang [20], la super résolution est un thème de recherche très attractif [39, 66, 80], à l'origine dans un grand nombre d'articles.

Ceci peut être expliqué par la difficulté de reconstruire une image HR à partir d'une image LR. Pour rappel l'équation 2.48 modélise la génération des images LRs en suivant le chemin optique, ici présenté dans les travaux de Irani et al. [25].

$$g(x, y) = d.h.w.f(x, y) + n(x, y) \quad (2.48)$$

Avec  $g(x, y)$  la version dégradée, décimée et bruitée de l'image HR  $f(x, y)$ , les opérateurs  $d, h, w$  correspondent respectivement aux opérateurs de décimation, flou et déformation. Le bruit additif est représenté par  $n$ . Du fait du nombre élevé d'inconnues dans cette équation, tenter de retrouver l'image réelle  $f$  haute résolution est un problème mathématique mal-posé car il existe une multitude de solutions.

La SR peut produire des séquences vidéos avec plus de niveaux de détails, améliorant les performances des applications de vision par ordinateur, typiquement en imagerie médicale, imagerie satellitaire ou pour la vidéo surveillance [113, 116, 162, 180].

De plus les algorithmes de Super Résolution peuvent être plus compétitifs financièrement que les approches matérielles car il est possible dans ces conditions de réutiliser les systèmes d'acquisition basse résolution qui ont des systèmes optiques à bas prix [89, 153, 176].

Les méthodes pour la génération d'images HRs à partir d'images LRs sont très variées. Nasrollah reprend dans une étude approfondie [152] une taxonomie présentant et catégorisant différentes techniques de SR. La figure 2.24 montre une classification possible des méthodes de SR. Certains auteurs séparent d'abord la SR en deux domaines de travail distincts : le domaine spatial (travail directement dans l'espace des images d'origine) et le domaine fréquentiel (les images sont transformées à l'aide d'une transformée de Fourier ou en ondelettes). Ces dernières transformées ont en général pour but de simplifier les calculs nécessaires pour résoudre les équations. Dans le domaine fréquentiel, les méthodes de SR tentent la plupart du temps de construire des images HR à l'aide

de filtrages inverses (pour le flou, le bruit et surtout la décimation).

Ensuite, les auteurs séparent encore ces domaines en sous-domaines en fonction du nombre d'images utilisées pour construire les images HRs ; ces deux nouveaux sous-domaines correspondent aux méthodes utilisant une seule image SISR pour *Simple Image Super Resolution* et des méthodes de MISR pour *Multiple Images Super Resolution* (qui utilisent plusieurs images de la même scène). La classification en nombre d'images utilisées est seulement faite dans le domaine spatial, en effet la plupart des méthodes du domaine fréquentiel utilisent plusieurs images LR.

Super résolution		
Domaine fréquentiel	Domaine spatial	
	MISR: <i>Multiple Image Super Resolution</i>	SISR: <i>Simple Image Super Resolution</i>
<ul style="list-style-type: none"> <li>• Transformée de Fourier</li> <li>• Transformée en Ondelettes</li> </ul>	<ul style="list-style-type: none"> <li>• IBP- <i>Iterative Back Projection</i></li> <li>• Filtrage itératif adaptatif</li> <li>• Directe</li> <li>• POCS- <i>Projection onto Convex Sets</i></li> <li>• Probabilités</li> </ul>	<ul style="list-style-type: none"> <li>• Apprentissage</li> <li>• Reconstruction</li> </ul>

FIGURE 2.24 – Extrait de la taxonomie des différentes familles de super résolution présentée dans l'étude approfondie de Nasrollahi et al. [152].

Le but de cette section de l'état de l'art est de mettre en avant les solutions permettant de répondre à la problématique de la thèse en satisfaisant les contraintes du projet européen.

En guise de rappel, le but du projet européen H2020 est de créer une caméra computationnelle qui intègre un capteur basse résolution UHD-4k ( $4096 \times 2048$  *pixels*), couplé à une optique compatible avec le capteur, et de pouvoir générer des images haute résolution UHD-8k ( $8192 \times 4096$  *pixels*). Le traitement embarqué doit respecter les contraintes temps réels en traitant jusqu'à 60 images par seconde.

Une grande partie des techniques de la taxonomie de Nasrollahi, figure 2.24, seront présentées dans les sections suivantes, cependant, contrairement à la classification proposée par l'auteur, l'état de l'art ne sera pas présenté en fonction de l'espace dans lequel les techniques de SR sont appliquées (fréquentiel ou spatial). Nous avons choisi une présentation fonction du nombre d'images LR nécessaires à la création d'images SRs, ce qui représente une contrainte d'implantation matérielle forte.

Les deux sous-sections suivantes correspondent d'une part aux méthodes utilisant plusieurs images LRs MISR "Multiple Image Super Resolution" (section 2.3.2.1) , et d'autre part aux méthodes utilisant une seule image SISR " Simple Image Super Resolution" (section 2.3.2.2).

### 2.3.2.1/ MULTIPLE IMAGE SUPER RESOLUTION MISR

Ces méthodes nécessitent d'avoir plusieurs images en entrée. Ces images doivent absolument représenter la même scène, et ce, avec les mêmes conditions d'éclairage et de temps d'intégration [112]. Les méthodes MISR ont pour but principal d'exploiter les informations présentes au niveau sub-pixelique comme présenté dans les travaux de Gross [21]. Ces techniques partent de l'hypothèse qu'il existe des décalages inférieurs au pixel entre des images consécutives. Cette théorie permet alors de recouvrer des informations en fusionnant des images qui ont été prises

consécutivement. Il est donc nécessaire que des micro-décalages existent entre deux images capturées à la suite. Cependant si ce décalage n'est pas présent, ou qu'il correspond à un multiple de la taille d'un pixel, la plupart de ces méthodes ne peuvent pas être appliquées. Le cas idéal est celui où les micro-décalages  $Pix_{shift}$  sont inférieurs à la distance entre deux pixels  $Dist_{pix}$ , ce qui permet d'obtenir des informations inter-pixels.

Généralement les méthodes MISR se décomposent en plusieurs étapes comme présenté dans la figure 2.25. Dans un premier temps, les images LR sont acquises. Ces images doivent représenter la même scène et comporter des micro-décalages et peuvent être acquises avec un ou plusieurs appareils d'acquisition différents. Une étape dite de "Registration", ou recalage des images, est appliquée par la suite ; cette étape permet de déterminer l'emplacement du pixel LR dans l'image HR. Enfin, une dernière étape de fusion est réalisée afin de mettre les pixels LR à leur emplacement : c'est la *projection*. Une dernière étape de correction du flou est parfois réalisée sur l'image HR obtenue, cette correction utilise des méthodes de déconvolution.

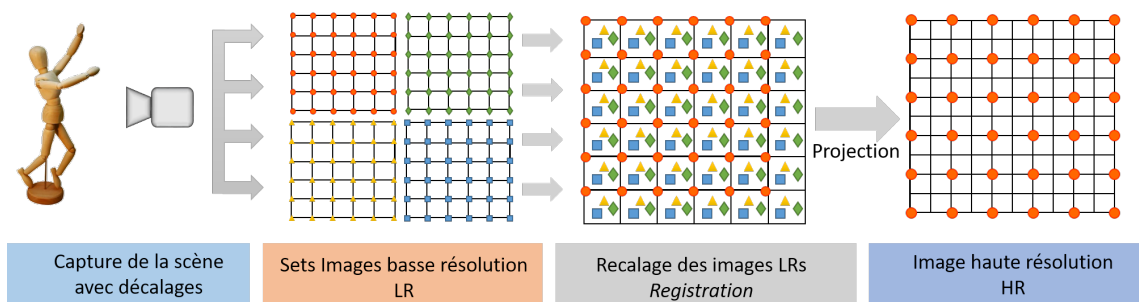


FIGURE 2.25 – Principe permettant la reconstruction d'images HR (*droite*) à partir d'images basse-résolution LR (*gauche*). Les décalages inter-pixel fournissent les informations complémentaires contenues dans les images LR. L'étape de recalage est au centre.

### Recalage des images - Registration

L'étape de recalage est commune à toutes les méthodes MISR. C'est une étape importante sans laquelle ce type d'algorithme de Super Résolution ne pourrait pas fonctionner. Le recalage permet à partir d'une image de référence d'estimer la translation qui existe entre la référence et l'image suivante. Le recalage permet de positionner plusieurs images dans le même système de coordonnées différentes images. Dans ce cas, il s'agit de mettre dans une image de taille HR les pixels provenant des images LR. Cette étape permet de projeter les pixels provenant des images LR dans une image HR de taille supérieure, la projection est réalisée en calculant les coordonnées que les pixels ont dans l'image HR.

Un grand nombre d'auteurs ont abordé ce problème, comme Lucas-Kanade [16], Keren et al.[22], et d'autres ont plus particulièrement traité le problème des décalages sub-pixel [72], [90]. La majorité des méthodes implémentées en temps réels sont des méthodes basées sur le "Block Matching" pour *correspondance de blocs*.

#### - Correspondance de blocs - Block matching :

Les images sont divisées en blocs de taille  $m \times n$ . Chaque bloc est comparé avec les blocs des autres images de la séquence. Par exemple, pour la Super Résolution, le bloc  $B$  issu de l'image  $I_n$  est comparé avec les blocs de l'image  $I_{n+1}$ . Le but est de trouver la meilleure correspondance. Cette comparaison suppose que la luminosité d'un objet dans l'image (ou d'une zone) reste constante entre deux images. Les performances des recherches de correspondances entre les blocs des images dépendent de plusieurs variables : la taille du bloc, le pas de recherche, le point de départ de la

recherche et enfin la direction de cette recherche. La taille du bloc est importante, car si elle est très grande alors il ne sera pas possible d'isoler les vecteurs mouvement de petits objet.

Une fois la correspondance entre deux blocs établie, un vecteur de mouvement peut alors être calculé. Ce vecteur représente le déplacement du bloc de l'image  $I_n$  vers l'image  $I_{n+1}$ . On considère alors que tous les pixels d'un même bloc se déplacent de la même manière.

Plusieurs métriques sont utilisées pour juger la correspondance des blocs. La métrique courante pour ce genre de techniques est la métrique SAD pour Sum of Absolute Difference (Somme Absolue des différences). Cette métrique est formalisée dans l'équation 2.49 et la correspondance est la meilleure lorsque la valeur de la métrique SAD est la plus petite.

$$SAD = \sum_{i=1}^N \sum_{j=1}^M |A(i, j) - B(i, j)| \quad (2.49)$$

Avec  $N \times M$  la taille définie pour la recherche de correspondance. Dans la majorité des cas  $N = M$ .  $A$  et  $B$  sont respectivement le bloc de l'image  $I_n$  et le bloc de l'image  $I_{n+1}$ . La comparaison du bloc  $A$  est faite avec un grand nombre de blocs de l'image suivante. Le couple de blocs  $(A, B)$  dont la valeur SAD sera la plus faible est choisi.

L'étape de recalage a des limites, par exemple si le mouvement est trop important il sera difficile de retrouver le bloc correspondant dans les images LR suivantes et l'estimation de mouvements sera mauvaise. Il se peut que des effets de Ghost (*Fantômes*) apparaissent alors, ce problème pouvant être réduit en diminuant le temps d'intégration des images en utilisant des caméras rapides [112]. L'effet de Ghost correspond à un dédoublement de l'image lors de la fusion de deux images qui ne sont pas alignées. En effet, lorsque le temps d'intégration est faible, le temps entre chaque image prise consécutivement est réduit, ce qui amoindrit le déplacement d'objet entre deux images consécutives.

Différents algorithmes sont présentés ci-dessous, en mettant l'accent sur des méthodes qui ont été embarquées dans le passé sur des cibles de type FPGA. Un grand nombre de méthodes n'ont jamais été implantées en matériel, car elles ne sont souvent pas compatibles avec des systèmes embarqués ou ne peuvent pas répondre aux contraintes temps réel. Les méthodes suivantes de cette section de l'état de l'art ont toutes en commun l'utilisation des techniques de recalage d'images.

### Méthode Directe

Tout d'abord, une fois que les images LR sont acquises, une image est choisie comme référence pour que les autres soient recalées. Ensuite, l'image prise pour référence est interpolée par un facteur déterminé (celui de l'agrandissement). Ensuite les autres images LR sont placées (pixel par pixel) sur l'image interpolée en prenant en considération les paramètres de recalage acquis précédemment.

Enfin, l'image HR est générée en fusionnant les images et il est possible ensuite d'appliquer un filtre de défloutage sur l'image finale.

L'ordre des étapes mentionnées ci-dessus a parfois été modifié par certains auteurs. Par exemple, dans Mancas et al. [75, 85], après avoir trouvé les informations de mouvement à partir des images LR, celles-ci sont mappées sur une grille HR pour établir une première estimation de l'image super résolue. Ensuite, un filtre Teager quadratique [26, 53], qui est un filtre non flou, est appliqué aux

images LR et celles-ci sont mappées sur la grille HR en utilisant les informations de mouvement précédemment trouvées pour générer une seconde image sur-résolue. Enfin, ces deux images super-résolues sont fusionnées en utilisant un filtre médian pour générer le résultat final. Il a été montré dans [75, 85] que cette méthode peut augmenter la lisibilité des images de texte.

En résumé, une fois les images LR acquises, il faut trois étapes pour pouvoir reconstruire une image HR :

1. *Recalage* : Cette étape permet de mettre dans le même référentiel spatial (coordonnées spatiales) des différentes images LR utilisées pour reconstruire l'image HR. Les différentes coordonnées sont calculées à l'aide de vecteurs de mouvement déduits des images LR.
2. *Interpolation* : Cette étape permet de construire une image qui a la taille de l'image HR. Cette image sert ensuite de grille où les pixels "recalés" pourront être placés et fusionnés pour intégrer leurs informations dans l'image HR.
3. *Défloutage* : Une dernière étape de défloutage permet d'améliorer la qualité de l'image à la suite des fusions des pixels provenant des images HRs.

### Iterative Back Projection

L'algorithme IBP proposé par Irani et Peleg [30], pour Iterative Back Projection, est comme son nom l'indique un algorithme itératif.

Son principe est de tenter de trouver une image HR à partir d'un ensemble d'images LR et de l'estimation du modèle de dégradation. Pour rappel ce modèle, qui représente les dégradations optiques que subit l'image HR jusqu'à ce qu'elle soit capturée en LR, est formalisé dans l'équation 2.48 où  $g$  correspond à une image basse résolution LR, une version déformée  $w$ , dégradée  $h$ , décimée  $d$  et bruitée  $n$  de  $f$ .

La figure 2.26 représente le fonctionnement algorithmique de la méthode IBP.

Tout d'abord une première image HR est estimée en fusionnant les informations contenues dans les images LR  $g_k$  projetées sur une grille de la taille de l'image HR. Cette grille est générée avec une interpolation linéaire à partir d'une première image LR  $g_k$ . Une fois cette image reconstruite la méthode prévoit de la dégrader en suivant le chemin de dégradation présenté dans l'équation 2.48. Une fois l'image LR estimée, elle est comparée à une image LR provenant de l'observation. L'erreur  $e^n$  est alors calculée entre la version simulée  $g_k^{(n)}(x, y)$  et la version provenant de l'observation réelle  $g_k(x, y)$ , cette erreur est calculée via l'équation 2.50 où  $g_k$  est la  $k$ -ième image LR observée, et  $g_k^{(n)}$  la  $k$ -ième image LR estimée à l'itération  $n$ .

$$e^n = \sqrt{\sum_k \sum_{x,y} (g_k(x, y) - g_k^{(n)}(x, y))^2} \quad (2.50)$$

Le but est alors de minimiser à chaque itération l'erreur  $e^n$  en estimant de nouvelles images HR, en suivant l'équation 2.51, où  $f_e$  est la projection de correction de l'erreur  $e$ . Cette projection  $f_e$  est présentée dans l'équation 2.52 avec  $H^{BP}$  le noyau de "back-projection", qui permet, selon Borman et al. [40], d'appliquer une estimation des dégradations inverses que subit l'image de référence  $f(x, y)$ , avec  $H^{BP^{-1}} = d.h.w$ .

$$f^{n+1} = f^n + f_e \quad (2.51)$$

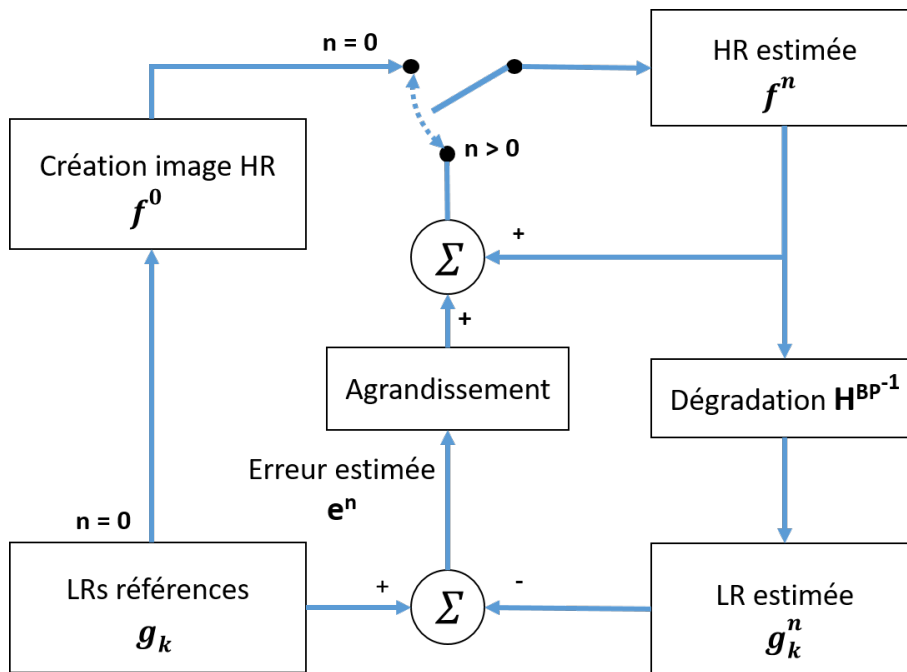


FIGURE 2.26 – Méthode IBP schématisée dans les travaux de Patel et al. [121]

Avec

$$f_e = H^{BP} (g_k - g_k^{(n)}) \quad (2.52)$$

Lorsque l'erreur  $e^n$  est très proche de 0 alors l'image estimée est considérée comme étant de bonne qualité.

Cependant les auteurs précisent dans [30] que les fréquences présentes dans l'image HR originale ne peuvent pas toutes être restaurées. Ceci est dû au filtre de flou qui est appliqué lors de la génération de l'image. Ce filtre passe-bas supprimant une partie des hautes-fréquences rendant le problème de restauration d'image HR insoluble, car il existe un grand nombre d'images HR qui peuvent être estimées à partir des images LR en estimant le noyau  $H^{BP}$ .

Un grand nombre d'articles présentent des implantations matérielles de ce type de méthode. Bowen et al. présentent dans [89] une implantation temps réel sur FPGA d'une méthode de la famille de IBP exploitant les informations inter-pixel créées par des translations [73]. Cependant, dans l'optique de réduire les accès mémoire, l'auteur restreint l'accès à seulement deux images LR d'une taille de  $320 \times 180 \text{ pixels}$ . Son traitement est réalisé au niveau du pixel avec une moyenne pondérée, le poids de cette pondération est basé sur l'estimation du mouvement inter-image. L'auteur affirme que son traitement, embarqué dans un Virtex II XC2V6000, permet de réaliser un agrandissement d'un facteur 4, avec une image de sortie de  $1280 \times 720 \text{ pixels}$  à une fréquence de 58MHz soit 61 images par seconde. Enfin pour cette implantation le nombre d'itérations est fixé à 10. Selon Szydzik et al. [123] 20 itérations seraient nécessaires pour atteindre une qualité satisfaisante. Ce nombre d'itérations était simplement limité par la mémoire disponible dans leur FPGA.

Angelopoulou [88] a implémenté une méthode IBP temps réel dans un FPGA Virtex-2. Sa contrainte principale était d'atteindre le temps réel avec une vitesse de traitement de 25 images

par seconde avec une fréquence de 60 MHz. Pour atteindre ces performances il est nécessaire de réduire le nombre d'itérations lorsque la taille de l'image augmente. De fait il est possible d'avoir 585 itérations avec une image HR de sortie de  $64 \times 64$  pixels et seulement 2 itérations pour une image de sortie de  $1024 \times 1024$ .

Dans leurs travaux, Szydzik et al. [123] partent du fonctionnement de IBP mais en version non itérative et utilisent seulement la mémoire embarquée dans leur puce FPGA de la famille des Virtex 5. Leurs travaux permettent de générer 25 images par seconde tout en utilisant jusqu'à 16 images de référence LR. Le traitement embarqué dans le Virtex 5 atteint une fréquence de 109 MHz et génère 25 images par seconde, et ceci avec un facteur de  $2\times$  sur des images QCIF  $176 \times 144$  pixels, ce qui crée donc des images CIF ( $288 \times 352$  pixels).

Dans sa thèse, [178, 160], Seyid implante un traitement similaire dans un Virtex-7. Le système atteint une fréquence de 265 MHz. Cette fréquence permet de générer 25 images par seconde avec une résolution de sortie de  $512 \times 512$  pixels et en prenant 17 itérations.

L'inconvénient de ces méthodes est la nécessité de stocker les images LR consécutives afin d'estimer une première image HR.

Ensuite il est nécessaire de réaliser plusieurs itérations de l'algorithme afin de minimiser l'erreur entre l'image LR simulée  $g_k^{(n)}$  et l'image LR observée  $g_k$ .

Afin d'atteindre des contraintes de temps réel, les auteurs précédents [88, 89, 123] réduisent le nombre d'itérations au détriment de la qualité des images HRs.

Cependant ces travaux atteignent certes plus de 25 images par seconde, mais avec des résolutions assez faibles comparées à la résolution cible de la caméra du projet H2020-EXIST.

### 2.3.2.2/ SIMPLE IMAGE SUPER RÉOLUTION SISR

Les méthodes SISR nécessitent seulement une image LR pour générer une image SR. Ce type de technique peut être bénéfique dans un grand nombre de cas :

- Il n'est pas possible d'avoir une séquence d'images LR. La SISR peut être utilisée pour augmenter la résolution et les détails d'une ancienne photographie, d'un document texte qui avait été numérisé avec une mauvaise qualité.
- Lorsqu'il est nécessaire de réduire la taille de stockage des images LR,
- Lorsqu'il n'est possible de procéder au recalage des images (par exemple, le temps inter-image est trop important).

Les algorithmes de Super Résolution basés sur l'utilisation d'une seule image sont la plupart du temps basés sur des méthodes d'apprentissage. Ils tentent d'estimer les informations manquantes en utilisant et en apprenant les relations entre des images LR et HR contenues dans l'ensemble d'entraînement. La différence majeure entre la SISR et les méthodes d'amélioration de la résolution habituelles (dont la succession des étapes est résumée dans l'algorithme 1) est que dans la Super Résolution les détails ajoutés ne peuvent pas provenir de simple rehaussement de contours d'une image précédemment interpolée. Les méthodes de SISR sont censées recouvrer les informations qui étaient initialement présentes dans l'image. Certaines techniques de la famille SISR seront présentées dans les paragraphes suivants. Cependant, une différenciation sera faite avec des techniques dites de SISR qui sont à la limite entre la SISR et l'amélioration de résolution. Ces techniques sont dites de Super Interpolation (SI).



**Algorithme 1** Méthodes d'amélioration de la résolution habituelles

- 1: Lissage avec filtre passe-bas (gaussien, de Wiener, moyenne, médian)
- 2: Interpolation (Voisin plus proche, bilinéaire, bicubique, Lanczos, etc.)
- 3: Accentuation des contours ("sharpening").

**Iterative Back Projection SISR**

Certains auteurs ont adapté la méthode IBP présentée précédemment en remplaçant tout d'abord l'étape de recalage des images par une interpolation.

Le processus reste le même, présenté dans la figure 2.27, mais avec l'ajout d'une étape d'amélioration de l'image interpolée via un détecteur de contour (type Canny pour Bareja [126], Gabor pour Marwana [140]). De ce fait, l'image HR estimée correspond à l'image LR interpolée dont les contours sont renforcés.

L'ajout de ces détecteurs de contours permettrait de réduire l'erreur entre l'image LR observée et l'image LR dégradée à partir de l'image HR estimée. Cette réduction serait possible grâce à l'ajout de hautes fréquences provenant des filtres passe-haut (Gabor, et Canny).

L'équation 2.53 présente la formation d'une image HR  $f^{n+1}$  à partir de l'image HR obtenue à l'itération précédente  $f^n$  et de l'ajout des hautes fréquences  $H_{cont}(f^n)$  et de la projection de correction de l'erreur  $f_e$ .

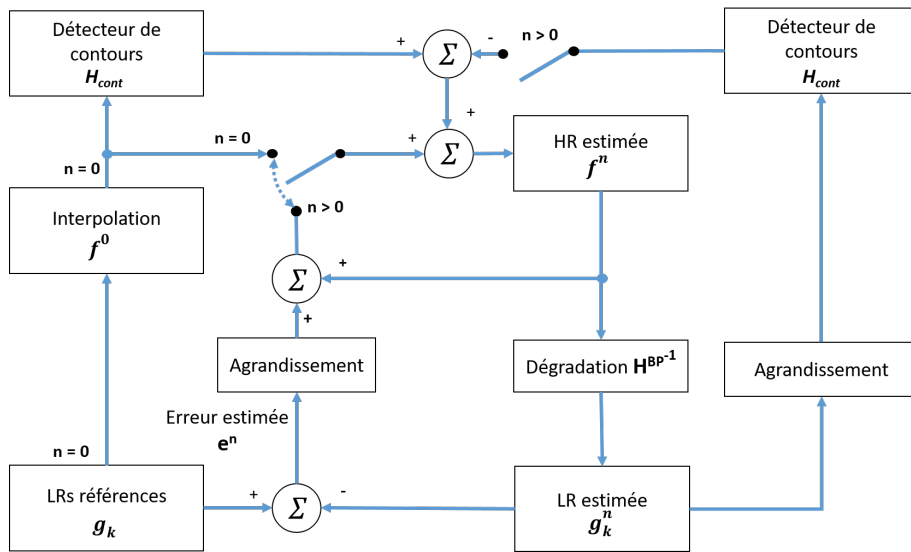


FIGURE 2.27 – Méthode IBP simple image avec ajout d'un détecteur de contour [140]

$$f^{n+1} = f^n + f_e + H_{cont}(f^n) \quad (2.53)$$

**Transformation en ondelettes**

Demirel et al. dans [107] puis dans [154, 109] proposent des solutions de SISR en passant par des décompositions en ondelettes. Dans le but d'obtenir des images SR de bonne qualité, il est nécessaire de préserver les contours. Dans leurs travaux, les transformées en ondelettes discrète DWT sont utilisées afin de conserver les hautes fréquences contenue dans l'image LR. Tout d'abord l'image d'entrée est décomposée en ondelettes DWT. Les imageries sous-bandes sont générées LL, LH, HL et HH. Les imageries qui contiennent les informations hautes fréquences sont les LH, HL, HH. Le déroulement du processus est explicité dans l'algorithme 2 et est présenté dans la



figure 2.28.

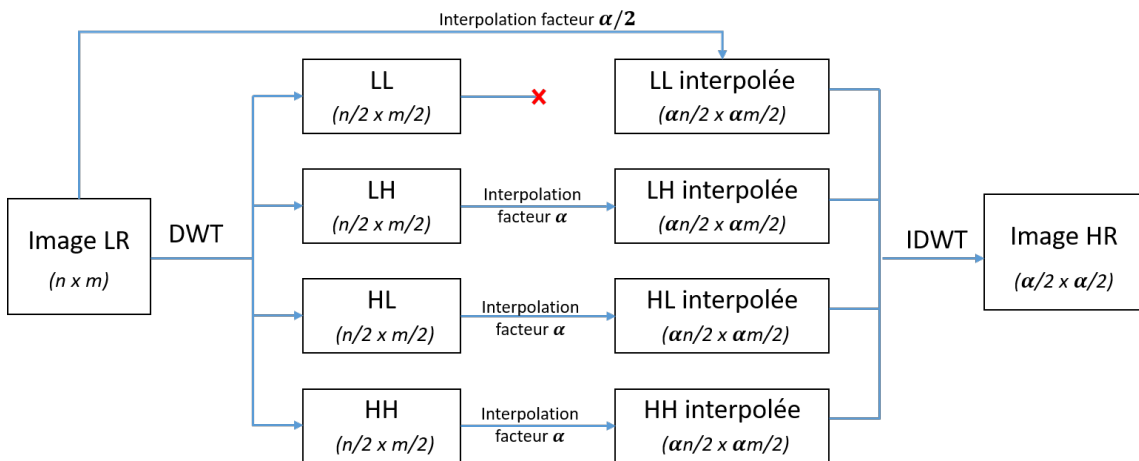


FIGURE 2.28 – Méthode DASR, présentée par Demirel et al. [107]

Dans [107] Demirel et al. comparent les transformées basées d'une part sur une décomposition de Haar [2], et d'autre part sur une décomposition de Daubechies 9/7 [87]. Dans les deux cas l'interpolation bicubique est utilisée car c'est avec cette interpolation que l'auteur obtient les meilleurs résultats (comparativement avec la méthode d'interpolation "voisin le plus proche" et l'interpolation bilinéaire). De plus, la meilleure méthode de décomposition est obtenue avec la décomposition Daubechies 9/7.

### Méthodes basées sur l'apprentissage

Dans leurs travaux, Freeman et al.[61] résument une de leurs premières idées de créer un algorithme de super résolution à partir d'exemples de patches issus d'un *dictionnaire*. Le principe de ce dictionnaire est de stocker des couples de patches ( $LR, HR$ ), plus précisément des caractéristiques des patches, permettant à partir d'un patch  $LR$  donné de trouver directement son équivalent  $HR$ . Néanmoins cette idée n'est pas applicable, il n'est pas possible d'utiliser directement un couple ( $HR, LR$ ) pour reconstruire une image  $HR$ , même si le dictionnaire est très grand. L'auteur montre qu'il existe un grand nombre d'extraits patches  $HR$  possibles qui répond aux caractéristiques du patch  $LR$ . La figure 2.29 met en avant un patch extrait d'une image  $LR$  (*gauche*) et cinq patches les plus proches (*droite*) qui ont les mêmes caractéristiques (quand ils sont normalisés).



FIGURE 2.29 – Des patches extraits des travaux de Freeman et al. [61], le patch de gauche a les mêmes caractéristiques que les autres patches du dictionnaire, ce qui rend impossible le simple remplacement du patch par son équivalent  $HR$ .

Freeman et al. introduisent alors l'idée d'une technique de super résolution basée sur un apprentissage de couples de patches ( $LR, HR$ ) mais cette fois-ci en exploitant les informations présentes dans leur voisinage [45, 49, 60]. Dans [60] les auteurs mettent en place un apprentissage supervisé basé sur la création de couple de patches ( $HR, LR$ ) nécessaires à l'apprentissage.

Cette base d'apprentissage est créée de la manière suivante :

**Algorithme 2** Méthode de Hasan Demirel DASR

- 1: Décomposition de l'image LR en utilisant la transformée DWT : génération d'image sous-bandes LL, LH, HL, HH
- 2: Interpolation des imagerie LH, HL, HH par un facteur  $\alpha$
- 3: Interpolation par un facteur  $1/\alpha$  de l'image LR d'entrée.
- 4: Transformée en ondelettes inverse (IDWT) pour obtenir l'image HR, l'image LR interpolée est utilisée à la place de la transformée LL.

1. Un ensemble d'images de références haute résolution est construit. Cet ensemble d'images est ensuite dégradé pour former des images LR, en appliquant particulièrement un flou puis une décimation. La décimation est faite avec un facteur de 2 dans chaque direction.
2. Décomposition des images en patches, avec les correspondances HR LR  $X_i, Y_i$ , et suppression de la variabilité par normalisation des contrastes locaux,
3. Construction d'une interprétation probabiliste entre les  $X_i, Y_i$  à l'aide d'un réseau de Markov.

$$P(x|y) = \frac{1}{Z} \prod_{ij} \psi_{ij}(x_i, x_j) \prod_i \phi(x_i, y_i) \quad (2.54)$$

Où  $\psi_{ij}(x_i, y_i)$  désigne la fonction de compatibilité entre les deux patches voisins  $x_i, x_j$  et  $\phi(x_i, y_i)$  représente la fonction de compatibilité régissant la probabilité d'émission d'un patch LR  $y_i$  étant donné un patch latent HR  $x_i$ .

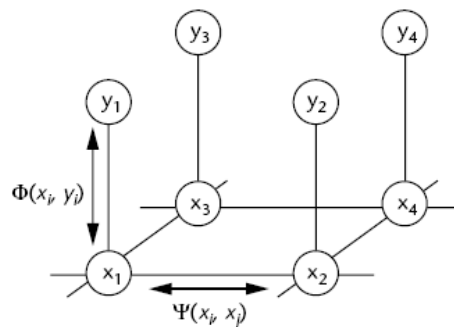


FIGURE 2.30 – Modèle du réseau de Markov présenté dans les travaux de Freeman et al. [61]

Freeman et al. modélisent la relation spatiale entre les patches en utilisant les champs aléatoires de Markov, souvent utilisés en traitement de l'image [19]. Ces champs sont représentés dans la figure 2.30 où les cercles représentent les nœuds du réseau, et les lignes représentent les dépendances statistiques entre les nœuds.

Les performances des techniques de SR ont été largement améliorées grâce à l'utilisation des méthodes d'apprentissage profond *Deep Learning* et atteignent les performances des techniques habituelles MISR de l'état de l'art [199, 168, 186, 169, 145, 198].

En 2016 Manabe et al. [174] ont réalisé une implantation FPGA temps réel d'une solution employant des réseaux de neurones convolutifs. Ils ont réalisé l'apprentissage avec des images qui

ont été générées de la même manière que Freeman et al. mais en incluant aussi des déplacements géométriques dans les images LR générées. Leur implantation matérielle, dans un FPGA de type Virtex Ultrascale, permet de générer à une fréquence de 48 images par seconde une image  $1920 \times 1080$  pixels créée à partir d'une image  $960 \times 540$  pixels.

En 2018, les auteurs Yongwoo et al. présentent dans [194] une implantation sur FPGA d'une technique permettant de transformer du contenu 2k (FHD) vers du contenu UHD-4k avec une vitesse de 60 images par seconde. La plate-forme de test est une carte de développement XILINX KCU105 embarquant un FPGA Kintex UltraScale et les auteurs ont atteint une fréquence de fonctionnement de 150 MHz. Leur méthode utilise des réseaux de neurones convolutifs *CNN*.

#### Super Interpolation SI

Les méthodes de Super Interpolation (SI) correspondent à un domaine de la Super Résolution travaillant sur une seule image. Ces méthodes ne sont pas reconnues en tant que vraies méthodes de Super Résolution car elles ne permettent pas de recouvrer les vraies informations qui étaient présentes dans les images HR de référence.

Ces traitements sont couramment basés sur une première étape d'interpolation, et une seconde étape d'amélioration de la qualité.

La différence majeure entre une technique de Super Résolution et une technique de Super Interpolation est le fait que la Super Résolution reconstruit les réels détails présents dans l'image HR, tandis que les techniques de Super Interpolation n'améliorent que l'effet de détail d'une image LR interpolée en image HR. L'amélioration de l'impression de détail est réalisée dans la majorité des cas par des algorithmes d'accentuation des contours.

Cependant, il sera possible de voir qu'un grand nombre de méthodes de SI peuvent être considérées comme des méthodes de SR, et vice-versa. Ce sont les auteurs eux-mêmes qui se définissent d'une méthode ou de l'autre.

#### - La méthode de Gohshi [128]

La bibliographie met en avant une méthode de Super résolution, travaillant sur une seule image. Cette méthode, de l'auteur Seiichi Gohshi, peut être vue comme un filtre 2D avancé. Selon l'auteur, cette méthode apporte des hautes fréquences permettant d'ajouter du détail dans l'image et de se rapprocher de l'image plus résolue. Gohshi affirme qu'il est possible d'ajouter des composantes du spectre de fréquences dépassant la fréquence de Nyquist. Selon l'auteur cette méthode fait partie de la Super Résolution, cependant elle peut tout aussi bien être classée dans la Super Interpolation.

Le principe de sa méthode est d'associer une interpolation 2D avec un traitement avancé des contours. Cette méthode, proche d'un masque d'accentuation des contours et d'amélioration de la qualité de l'image [7, 15] se distingue par l'utilisation d'une fonction non-linéaire *NLF*. Les étapes de cette méthode sont explicitées dans la figure 2.31. Tout d'abord, l'image basse résolution *LR* est interpolée (bicubique, lanczos). Un filtre passe haut *HPF* est appliqué sur cette image interpolée, et ensuite une fonction non linéaire est appliquée à la sortie de ce filtre (cubique [138], carrée) en faisant attention de conserver le signe de la sortie du filtre passe haut.

Le résultat de la fonction non linéaire peut tendre rapidement vers l'infini à cause de la fonction cubique. Il est alors limité par le traitement *limiteur* avant d'être ajouté à l'image interpolée.

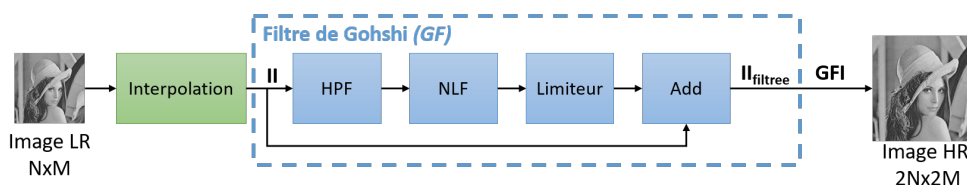


FIGURE 2.31 – Méthode dites de Gohshi

L'équation 2.55 formalise le fonctionnement de la méthode de Gohshi, où  $GFI$  représente l'image  $HR$  résultat de la méthode de Gohshi,  $II$  est l'image interpolée, et  $II_{filtree}$  est l'image interpolée filtrée suivant les différentes étapes de la méthode.

$$GFI = II + II_{filtree} \quad (2.55)$$

$II_{filtree}$  est détaillée dans l'équation 2.56 et correspond au résultat du filtre passe haut  $HPF$  mis au cube avec la fonction non linéaire et limitée  $lim$ .

$$II_{filtree} = lim((II * HPF)^3) \quad (2.56)$$

L'auteur a montré la possibilité d'implanter sa méthode sur une architecture matérielle embarquant des FPGA [137, 138].

En 2013, le traitement de Gohshi a été implémenté dans un FPGA [139] et l'auteur a été capable de convertir du contenu HDTV  $1920 \times 1080$  en 4k-TV  $3840 \times 2160$  à une fréquence de 24 images par seconde. L'auteur a créé un "convertisseur" (figure 2.32) qui peut être branché sur des téléviseurs afin d'adapter du contenu basse-résolution à la résolution de l'écran.



FIGURE 2.32 – Convertisseur HD-UHD créé par Gohshi

En 2014, le même auteur décrit une méthode [147] permettant de générer du contenu 8k à partir d'images 4k-TV. Le 8k correspondant à un nombre de pixel  $4 \times$  supérieur à la 4k, l'auteur propose de découper l'image en 4 zones afin de traiter en parallèle les 4 quarts d'image avant de construire l'image 8k finale. Ce découpage est l'occasion d'une part de montrer le fort parallélisme potentiel de son traitement, mais aussi d'utiliser quatre convertisseurs qu'il a créés.

L'auteur a protégé sa technologie par le dépôt de brevets [127, 167], et a validé dans plusieurs articles les performances de sa méthode, en réalisant des études psycho-visuelles dans lesquelles des humains ont noté directement la qualité du traitement [128, 148, 157]. Ces études subjectives ont été réalisées à chaque fois par des observateurs qui donnaient des notes aux traitements. Les observateurs disposaient de deux téléviseurs haute-résolution et devaient noter le traitement du

premier téléviseur et du second. Pour chaque évaluation le traitement issu de la méthode de Gohshi était noté comme étant le meilleur.

-Perez et al.

La méthode développée par les auteurs Perez et al. [158] met en avant un principe dans lequel différents traitements sont appliqués en fonction du pixel et de son voisinage.

Le principe est de traiter différemment les zones de l'image en fonction de leur dégradation, et cela, dans le but de réduire le temps de calcul nécessaire à la reconstruction de l'image HR.

Un classifieur basé sur une méthode de Boosting [35] est utilisé pour séparer l'image en différentes zones en fonction de leurs classes. Cette classification permet de séparer en deux classes distinctes les différentes zones de l'image (*patch*), les *patches* ayant subi des dégradations ou non. Le terme *dégradation* présente le fait que l'image basse résolution LR provient d'une image haute résolution HR dont des détails ont été perdus. Les zones *sans dégradation* représentent des zones de l'image LR lisses (sans texture) qui proviennent probablement de zones originalement lisses de l'image HR.

La classification permet alors de représenter l'image sous forme d'une *carte de probabilité des dégradations*. En fonction des probabilités de dégradations, la zone de l'image est traitée de deux manières différentes :

- La zone est lisse : un *upscaler* rapide est utilisé,
- La zone a subi des dégradations : un *upscaler* complexe est utilisé.

L'auteur précise que l'*upscaler* rapide correspond à une interpolation bicubique car, selon l'auteur, il s'agit d'une méthode rapide et largement utilisée dans l'état de l'art.

L'*upscaler* complexe utilisé est une méthode de super résolution basée sur une régression linéaire construite à partir d'un apprentissage basé sur les exemples [115], et l'utilisation d'une régression linéaire [143].

Cette méthode est présentée dans la figure 2.33.

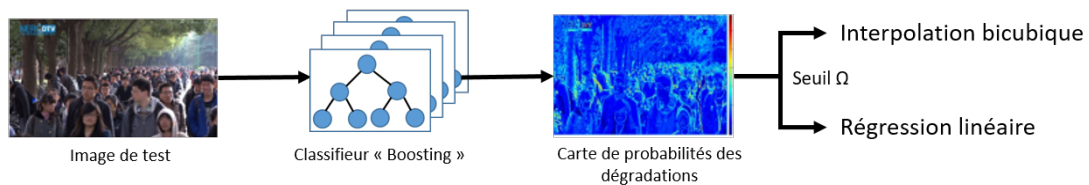


FIGURE 2.33 – La méthode de Perez présentée dans [158]. Pour chaque *patch* de l'image de test, une carte des dégradations possibles est estimée avec un arbre de décision boosté. En fonction de cette probabilité une interpolation bicubique, ou une méthode complexe basée sur une régression linéaire est réalisée.

Les auteurs utilisent, pour leurs travaux, une version modifiée de Adaboost [98]. Un apprentissage aveugle des dégradations présentes dans les images est réalisé en entraînant le système avec des images réelles, naturelles, pour lesquelles une bonne reconstruction est possible. Cet apprentissage est basé sur les limites d'échantillonnages du pipeline de reconstruction présenté dans l'équation 2.58. Les images LR ont été créées avec une décimation réalisée avec une interpolation bicubique, et les pixels ont été seuillés pour soustraire 34 dB.

$$Y = \downarrow X * h_s \quad (2.57)$$

Avec  $X$  l'image réelle haute résolution,  $h_s$  un noyau de dégradation contenu du flou et du bruit et  $\downarrow$  l'opération réalisant la décimation *downsampling*. La version agrandie et améliorée de  $X$ ,  $\hat{X}$  est présentée dans l'équation 2.58, avec  $v$  l'opérateur d'agrandissement.

$$\hat{X} = v(\downarrow X * h_s) \quad (2.58)$$

Pour l'apprentissage, les paires de patches utilisées sont  $\{(y_j, x_j)\}$ , où  $y_j$  correspond à un patch issu de l'image  $Y$ , et  $x_j$  provient de l'image  $X$ . Dans le but de transformer des paires d'apprentissage  $\{(y_j, x_j)\}$  en paires d'apprentissage annotées  $\{(y_j, A_j)\}$  les auteurs prennent  $A_j = \{-1, +1\}$ .  $A_j$  est calculé en suivant l'équation 2.59 en comparant la différence entre l'image réelle  $X$  et l'image reconstruite  $\hat{X}$  comparée à un seuil  $\epsilon$ . Les auteurs ne donnent pas de précisions supplémentaires sur le seuil  $\epsilon$ .

$$A_j = \begin{cases} -1, & \|x_j - \hat{x}_j\|_2^2 \leq \epsilon \\ 1, & \|x_j - \hat{x}_j\|_2^2 > \epsilon \end{cases} \quad (2.59)$$

En traitant avec une régression linéaire seulement les parties qui en ont besoin dans l'image, les auteurs ont réussi à accélérer le traitement d'un facteur de 1200 par rapport aux résultats de Yang et al.[115] pour reconstruire une image de 2k vers 4k. De plus, Perez et al. obtiennent de meilleurs résultats, en terme de qualité de reconstruction, que Yang et al. sur la même base d'images traitées.

Cette technique est plus proche des techniques basées sur l'apprentissage réel, et non sur de la super interpolation. Le but principal est d'accélérer une méthode d'apprentissage basée sur des exemples de paires (*learning based example* en diminuant les zones où sont appliquées la SR, mise en place avec des méthodes de régression linéaire.

#### Choi et al.

Cette méthode [166] nécessite une phase d'entraînement dans laquelle les patches provenant des images LR sont classés en fonction de l'orientation de leurs contours. Durant la phase d'augmentation de la résolution, phase qui se déroule cette fois-ci en ligne, les patches HR sont générés en appliquant des fonctions de correspondances basées sur l'orientation des contours. Ensuite ces travaux sont améliorés dans [193] permettant de répondre aux exigences temps réel pour l'implantation dans un FPGA.

Cette méthode a été implantée sur une carte d'évaluation de Xilinx KCU105 embarquant un Kintex UltraScale à une fréquence de 150 MHz. Pour permettre la génération de contenu 4k à une fréquence 60 images par seconde, 4 pixels sont traités par cycle d'horloge.

#### He et al.

He et al. présentent une approche similaire dans [192]. Cette méthode combine une interpolation bilinéaire à un réseau de neurones. Le principe de cette méthode est de découper l'image basse résolution LR  $X$  en blocs  $x$ . Pour chaque bloc  $x$  la variation totale [70, 27]  $TV(x)$  est mesurée. En fonction de la valeur de  $TV(x)$  le bloc est soit traité par le réseau de neurones, soit tout simplement interpolé. La variation totale du bloc  $x$  permet de mettre en avant le taux de contours et la texture présente dans ce même bloc. L'algorithme 3 explicite le fonctionnement de cette méthode. Il est à noter que l'interpolation bilinéaire est notée *CheapUpscale* et le traitement utilisant le réseau de neurones est noté *Upscale*.

Le système développé par He et al. permet de convertir une image FHD ( $1920 \times 1080$ ) en une image de sortie UHD ( $3840 \times 2160$ ) avec une fréquence de 31.7 images par secondes sur une carte de développement Xilinx ZC706.

---

**Algorithme 3** Méthode de Super Résolution proposée par He et al.[192]

---

**Entrée:** Image LR  $X$ , facteur d'interpolation  $n$ , seuil  $T$

**Sortie:** Image HR  $Y$

```

1: Découpage de  $X$  en imagerie  $x$ 
2: pour tout imagerie  $x$  faire
3:   si  $TV(x) \geq T$  alors
4:      $y \leftarrow Upscale$ 
5:   sinon
6:      $y \leftarrow CheapUpscale$ 
7:   fin si
8: fin pour
9: Assemblage des blocs  $y$  pour générer  $Y$ 

```

---

### 2.3.2.3/ DISCUSSION ET ORIENTATIONS

Les méthodes considérées comme étant les meilleures pour la reconstruction d'images haute-résolution HR, au niveau de la qualité des images, sont les méthodes utilisant des groupes d'images consécutives, appelée Multi-Images Super Résolution. En effet, d'après la définition de la Super Résolution cette famille de méthode semble être la seule à tenter de récupérer les informations présentes au niveau subpixel, grâce aux micro-décalages qu'il peut exister entre deux images consécutives.

Cependant ce genre de techniques semble être complexe à implémenter, et cela, surtout pour l'étape de recalage des images. En effet il est nécessaire d'estimer les mouvements des pixels des images LR pour pouvoir créer une image HR. De plus il est nécessaire de stocker au moins quatre images consécutives pour pouvoir construire une image HR.

Dans le cadre de notre projet H2020-EXIST, qui est donc de créer une caméra computationnelle permettant à partir d'un capteur 4k de générer des images 8k, nous avons cherché à limiter les coûts de mémorisation et les coûts liés à la complexité de l'estimation précise du mouvement. Nous avons donc décidé de travailler directement dans le flux et de réduire autant que possible l'utilisation de la mémoire. De ce fait, nous nous sommes orientés par la suite vers les méthodes de type Simple Image Super Résolution SISR dont l'avantage est de permettre travailler avec une seule image sans nécessité de stocker les images 4k précédentes.

Comme nous l'avons vu, de nombreuses techniques SISR sont proposées dans la littérature, avec des complexités très variées. Actuellement il est possible de voir les méthodes d'apprentissage profond appliquées dans une multitude de domaines, et cela au prix d'une puissance de calculs très importante. Certaines méthodes basées sur l'apprentissage se disent de la SISR et donnent de très bon résultats. Cependant ce genre de traitement réduit en fait l'erreur entre l'image HR réelle et l'image HR estimée. Il n'est pas encore possible de recouvrer exactement les détails qui ont été perdus, mais ces méthodes se rapprochent très vite de l'objectif. Enfin, ces méthodes n'étaient pas forcément toutes disponibles au début du projet et il semblait très complexe d'intégrer celles-ci dans une caméra computationnelle.

La famille correspondant à la méthode de Gohshi semble être la plus adaptée pour travailler dans le flux de l'image (sans stocker d'image) et permettre de traiter des images d'une résolution de 4k en entrée pour générer des images 8k en sortie. L'auteur a pu montrer qu'il était possible de paralléliser le traitement pour atteindre de meilleures performances.



Cette méthode peut être vue comme un filtre 2D avancé, pour lesquels les composants de type FPGA sont particulièrement adaptés.

L'auteur inscrit sa méthode dans la famille des méthodes de SISR, cependant une nouvelle dénomination des méthodes est apparue pour différencier ces méthodes de l'état de l'art : la Super Interpolation. La différenciation entre SISR et Super Interpolation est très faible. Certains auteurs, comme Gohshi, considèrent leurs méthodes comme faisant partie de la SISR alors que les traitements effectués ne peuvent pas forcément retrouver les informations qui ont été perdues lors de l'acquisition de l'image.

## 2.4/ DISCUSSION ET CONCLUSION

L'imagerie computationnelle peut être une alternative aux systèmes d'acquisition non conventionnelle. Ces techniques permettent de trouver un compromis entre le coût du capteur d'image, de l'optique et de la complexité des traitements qu'il est possible d'embarquer. Il est nécessaire de prendre en compte la complexité du traitement dans le coût global car si le traitement est très complexe le temps de développement sera très long, et la carte de traitement pourra elle aussi embarquer un composant coûteux. La création d'une caméra computationnelle est au cœur du projet européen dans lequel la thèse s'inscrit.

Tout d'abord, l'état de l'art sur les métriques de l'images, section 2.2 ne nous permet pas de choisir des métriques efficaces pour juger l'amélioration des traitements que nous voulons mettre en place. En effet, des problèmes existent dans les deux types de métriques étudiées. Concernant les métriques pleines références il est toujours question de différence entre l'image à tester et l'image de référence. Si par exemple un traitement améliore grandement la qualité d'une image, cette image sera plus éloignée que l'image de référence. Le score de la métrique sera mauvais alors que l'image traitée sera de meilleure qualité. Ensuite, la majorité des métriques sans références se basent sur les proportions mathématiques de contraste ou de contours francs dans les images. Pour ces raisons il est nécessaire de mettre en place une étude approfondie sur la pertinence des métriques pour juger de la qualité des images dans notre contexte.

Ensuite, concernant la déconvolution il est nécessaire d'implanter un algorithme qui permet de corriger les images en temps réel. Cette partie du projet n'étant pas le sujet principal de cette thèse, l'implantation de traitement de correction d'aberrations optiques n'est donc présentée qu'en second lieu. Cependant l'état de l'art des techniques de déconvolution nous a permis de choisir l'algorithme à implanter et à tester. Le fabricant de l'optique pour le projet nous a communiqué les PSF, de ce fait il est possible d'utiliser la déconvolution non-aveugle. Le filtre de Wiener a été choisi pour cette déconvolution, et la cible de test est un GPU. Cette cible a été choisi tout d'abord pour réaliser une preuve de concept avant de connaître la possibilité d'implanter le traitement sur une puce GPU intégrée dans la caméra computationnelle, ou s'il sera nécessaire de transposer le programme pour une cible FPGA.

Enfin, concernant la création d'une caméra computationnelle réalisant de la Super Résolution, l'état de l'art nous permet de choisir le type de méthode que nous désirons implanter. Le principe premier de la Super Résolution est de recouvrir les informations des pixels manquants à partir d'images consécutives. Cette possibilité réside dans la présence de micros mouvements inter-images faisant apparaître des informations présentes entre deux pixels : les informations sub-pixéliques. Ces techniques MISR *Multiple Images Super Résolution* nécessitent d'avoir plusieurs images consécutives et sont excessivement gourmandes en calculs et en mémoire, notamment pour le stockage des images consécutives nécessaires pour calculer le mouvement présent entre chaque image, dans le but de recalculer les pixels d'une image à l'autre. Afin de réduire les ressources né-



cessaires à la création d'images HR 8k, nous avons décidé de mettre en place une technique SISR, *Simple Image Super Résolution*. L'étude de l'état de l'art nous a conduit à nous baser sur la méthode originale créée par Seiichi GOHSHI. La méthode qu'il a mise en place permet de construire des images HR à partir d'une seule image LR en réalisant un traitement évolué sur l'image HR interpolée. Le traitement réalisé sur l'image interpolée permet de faire ressortir les détails présents dans les hautes fréquences. Cette méthode est, selon l'auteur, parfaitement parallélisable sur FPGA ce qui permet d'avoir un fonctionnement en temps réel et qui est compatible avec une fréquence image élevée.



## ÉTUDE ET DÉVELOPPEMENT

**C**e chapitre présente les différentes contributions apportées lors de l'étude et le développement de notre méthode originale LASSR (*Local Adaptive Spatial Super Resolution*).

Tout d'abord, la section 3.1 présente notre étude sur la pertinence des métriques usuelles utilisées pour juger la qualité des images. Cette étude permet de vérifier la cohérence des résultats des différentes métriques dans le contexte de la super résolution.

La section 3.2 présente notre mise en œuvre et l'évaluation de la méthode de Gohshi. Cette étude mettra en avant certaines limites de la méthode de Gohshi que nous avons pu mettre en place à partir de ses publications scientifiques.

La section 3.3 met en avant notre contribution par la création d'une méthode originale LASSR. Cette méthode est destinée à améliorer la méthode de Gohshi en adaptant localement le traitement appliqué au pixel en fonction de sa texture. Cette section décrit la recherche de solutions techniques permettant de résoudre ce problème et ceci en restant compatible avec une future implantation matérielle.

Enfin, la section 3.4 présente l'évaluation de notre méthode LASSR.

### Sommaire

---

<b>3.1</b>	<b>Évaluation de la pertinence des métriques de la qualité de l'image . . . . .</b>	<b>60</b>
3.1.1	La base d'images . . . . .	60
3.1.2	Les traitements appliqués aux images . . . . .	60
3.1.3	Classement des traitements par les métriques sélectionnées . . . . .	61
3.1.4	Discussion et proposition . . . . .	64
<b>3.2</b>	<b>Évaluation de la méthode de Gohshi . . . . .</b>	<b>66</b>
3.2.1	Présentation complète . . . . .	66
3.2.2	Analyses critiques . . . . .	73
<b>3.3</b>	<b>Solution proposée : LASSR . . . . .</b>	<b>76</b>
3.3.1	Classification de Texture . . . . .	76
3.3.2	Fusion . . . . .	84
<b>3.4</b>	<b>Évaluation de la méthode LASSR . . . . .</b>	<b>85</b>
3.4.1	Critique visuelle de la méthode LASSR . . . . .	85
3.4.2	Résultats des métriques conventionnelles . . . . .	86
3.4.3	Mise en place de l'évaluation psychovisuelle $PVA_2$ . . . . .	86
3.4.4	Conclusion . . . . .	90

---

### 3.1/ ÉVALUATION DE LA PERTINENCE DES MÉTRIQUES DE LA QUALITÉ DE L'IMAGE

Au vu de l'état de l'art et de la multitude des méthodes employées par les auteurs, il est difficile de choisir objectivement une métrique pertinente dans le cas spécifique de l'estimation de l'amélioration de la résolution spatiale. En effet, les algorithmes évalués et les ensembles d'images utilisés sont parfois très différents d'un article à l'autre. Nous avons donc mené une sélection expérimentale, en nous basant sur un ensemble d'images acquises en 8k, et en étudiant un nombre limité d'algorithmes proches de nos préoccupations, à savoir un traitement de type super interpolation (SI), compatible avec l'implantation temps réel requise par notre projet.

Les métriques qui sont évaluées dans cette étude correspondent aux métriques dont les auteurs ont rendu le code accessible. Les métriques sans références utilisées dans cette partie sont le PSI, CPBD, EME, NIQE, BRISQUE. Les métriques pleines-références sont PSNR, SSIM, VIF, VSI, FSIMc, IQI.

#### 3.1.1/ LA BASE D'IMAGES

Notre base d'images  $S_{métrique}$  est composée de 623 patches ou imageries de résolution spatiale de  $1024 \times 1024$  pixels extraits d'images 8k capturées à l'aide d'un appareil photo NIKON D800. Ces images sont caractérisées comme *naturelles* car elles sont enregistrées dans un format sans compression, et aucun post-traitement n'a été réalisé. Ces images représentent une multitude de scènes allant du portrait au paysage, comportant plusieurs niveaux de détails. Des exemples de  $S_{métrique}$  sont présentés sur la figure 3.1.



FIGURE 3.1 – Exemple d'imageries provenant de la base  $S_{métrique}$  utilisées dans l'étude sur les métriques.

Ces images sources qui forment la vérité de terrain sont alors considérées comme images de référence pour les métriques qui utilisent des références.

#### 3.1.2/ LES TRAITEMENTS APPLIQUÉS AUX IMAGES

Nous avons évalué trois traitements couramment utilisés dans les méthodes de type SI. Ces traitements ne modifient que le piqué de l'image (aucune déformation ou distorsion n'est appliquée). Le traitement  $T_1$  réalise un floutage sur l'image de référence, le traitement  $T_2$  réalise un rehaussement des contours de type *sharpening* à partir des images issues de  $T_1$  et le traitement  $T_3$  rehausse

les contours de l'image de référence (Source). L'organisation de ces traitements est explicitée dans la figure 3.2.

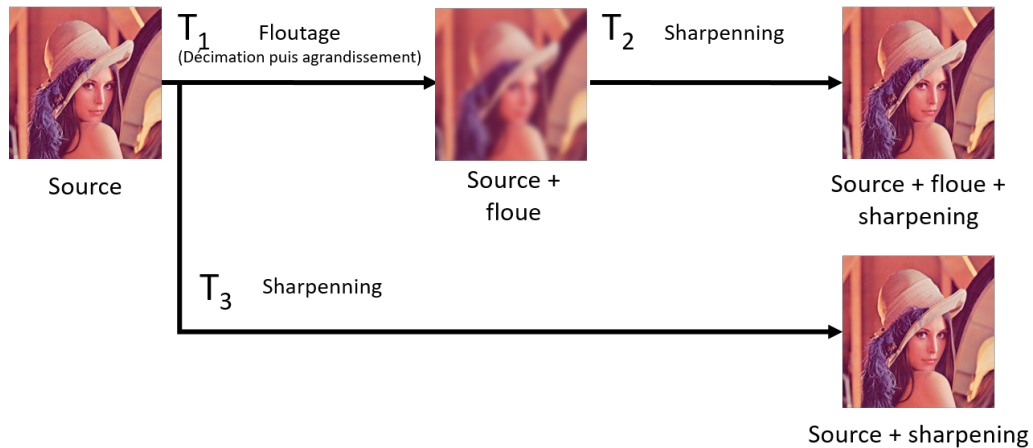


FIGURE 3.2 – Traitements appliqués sur les imagerie 1024 × 1024 pixels extraits d'images de résolution d'origine 8k

Les traitements appliqués sont :

- Le floutage  $T_1$  est réalisé en employant une technique d'élimination des hautes fréquences de l'image en réalisant une décimation d'un facteur 2, puis un agrandissement spatial de l'image d'un même facteur, afin de retrouver la taille de l'image d'origine, en appliquant une interpolation standard bicubique.
- Le rehaussement de contour est réalisé avec de type sharpening standard. Un filtre gaussien est appliqué à l'image, ce qui produit une image floue, puis une somme pondérée est réalisée entre l'image d'origine et l'image floue (avec un coefficient négatif pour cette dernière). Ce filtre est appliqué sur le canal correspondant à la luminance dans l'espace de couleur Lab. Le même algorithme est utilisé pour  $T_2$  et  $T_3$ , sur des sources différentes.

Intuitivement, au niveau de la qualité visuelle, nous attendons que les traitements soient classés comme suit  $T_3 > T_2 > T_1$ . La qualité visuelle ici correspond au niveau de détails et de netteté de l'image. En effet  $T_1$  est la version floutée de l'image de référence,  $T_2$  est l'image floutée avec des contours renforcés et enfin  $T_3$  correspond à l'image de référence avec un renforcement de contours. Enfin il est aussi possible de classer intuitivement  $T_3 > Source > T_1$ , mais il est impossible d'inclure le traitement  $T_2$  dans cette comparaison. En effet il n'est pas possible, intuitivement, de quantifier le niveau de pertes de piqué de l'image source et de savoir si le renforcement des contours permettra de retrouver le piqué d'origine de l'image source ou de l'améliorer.

Un exemple d'application de ces trois traitements est présenté dans la figure 3.3 avec comme image de référence la figure 3.3-a, et les traitements  $T_1$ ,  $T_2$  et  $T_3$  correspondent respectivement aux figures 3.3-b, 3.3-c et 3.3-d.

### 3.1.3/ CLASSEMENT DES TRAITEMENTS PAR LES MÉTRIQUES SÉLECTIONNÉES

Les résultats correspondant aux métriques sans références sont tout d'abord présentés dans le tableau 3.1 dans la section 3.1.3.1, et les résultats des métriques pleine-référence sont présentés dans le tableau 3.2 dans la section 3.1.3.2. Ces tableaux présentent les moyennes des résultats obtenus pour la totalité des images de l'ensemble  $S_{métrique}$ .



FIGURE 3.3 – Exemple d’images issues des traitements utilisés pour la comparaison des métriques, (a) correspond à l’image source, (b) image floue provenant de  $T_1$ , (c) image issue de  $T_2$  : image provenant de  $T_1$  suivit d’un rehaussement de contours et l’image (d) correspond au traitement  $T_3$ , rehaussement de contours appliqué à l’image source.

### 3.1.3.1/ MÉTRIQUES SANS-RÉFÉRENCE

Nous comparons dans la première ligne du tableau 3.1 l’image source aux autres traitements afin d’estimer si des traitements améliorent la qualité originale de l’image. Le classement pour cette image n’est donné qu’à titre indicatif et n’influencera pas le choix des métriques.

Tout d’abord, il est possible de constater dans le tableau 3.1 que les métriques n’évoluent pas de la même manière pour les différents traitements. Cependant le traitement  $T_3$  arrive en première position (comme étant le meilleur) pour 80% des métriques sélectionnées (PSI, CPBD, EME et



### 3.1. ÉVALUATION DE LA PERTINENCE DES MÉTRIQUES DE LA QUALITÉ DE L'IMAGE63

	PSI	CPBD	EME	NIQE	BRISQUE
source	0.36 (3)	0.54 (2)	4.53 (2)	6.01 (4)	36.04 (2)
$T_1$	0.33 (4)	0.27 (4)	3.53 (4)	5.89 (3)	44.54 (3)
$T_2$	0.38 (2)	0.44 (3)	4.23 (3)	<b>5.27 (1)</b>	44.58 (4)
$T_3$	<b>0.46 (1)</b>	<b>0.71 (1)</b>	<b>5.97 (1)</b>	5.70 (2)	<b>32.97 (1)</b>

TABLE 3.1 – Classement des traitements par les métriques sans référence sélectionnées dans la section 2.2 (l'ordre est présenté entre parenthèses). Les valeurs correspondent aux moyennes des métriques obtenues sur  $S_{métrique}$ .

BRISQUE). Ce classement rejoint le classement *a priori* réalisé dans la description de cette étude section 3.1.2. L'image source arrive dans 60% des cas en seconde position. Les images issues du traitement  $T_1$  sont classées dans 60% des métriques en quatrième position. Les métriques sélectionnées prennent en compte essentiellement le niveau de piqué de l'image, ce qui en résulte par le classement dans les deux dernières positions du traitement  $T_1$ .

Il n'existe pas de réelle tendance issue des résultats obtenus par les métriques réalisées sur le traitement  $T_2$ , son classement oscille entre la première et la dernière position. Les images issues de ce traitement ne peuvent pas être clairement évaluées avec les métriques sélectionnées.

#### 3.1.3.2/ MÉTRIQUES PLEINE-RÉFÉRENCE

Nous pouvons tout d'abord remarquer, que contrairement au cas précédent, aucune tendance forte ne ressort des résultats présentés dans le tableau 3.2. En effet le traitement  $T_3$  arrive en tête pour les métriques SSIM, VIF, et IQI ce qui représente seulement la moitié des métriques testées. Le classement des traitements varie fortement en fonction des métriques.

	PSNR	SSIM	VIF	VSI	FSIMc	IQI
source	Inf	1.00	1.000	1.00	1.00	1.00
$T_1$	44.66 (1)	0.94 (2)	0.986 (3)	0.99986 (1)	0.9991 (1)	0.70 (2)
$T_2$	43.52 (3)	0.92 (3)	1.005 (2)	0.99969 (3)	0.9977 (3)	0.67 (3)
$T_3$	44.36 (2)	0.97 (1)	1.010 (1)	0.99970 (2)	0.9981 (2)	0.86 (1)

TABLE 3.2 – Classement des traitements par les métriques avec référence (l'ordre est présenté entre parenthèse et est obtenu avec une précision plus grande que celle affichée dans le tableau). Les valeurs correspondent aux moyennes des métriques obtenues sur  $S_{métrique}$ .

Dans cette seconde analyse nous ne traitons pas directement les résultats obtenus avec l'image source, car étant des métriques pleines références il n'est rien de plus similaire que deux images identiques.

Nous pouvons remarquer que le traitement de flou  $T_1$  est classé premier pour 50% des métriques, tout comme le traitement  $T_3$ . Ce classement va à l'encontre du classement intuitif *a priori* où le classement de flou  $T_1$  arrive en dernière position. Cela met en avant une partie des limites de l'utilisation des métriques pleine-référence. Car même si ces métriques évoluées se basent sur des similitudes de structure, de contraste, ou autres paramètres qui se rapprochent de la perception humaine, le principe est toujours de mesurer une distance entre l'image de référence et l'image traitée.

Il est possible de voir que la moyenne de SSIM est la plus grande pour le traitement  $T_3$ . Cependant il est aussi possible de voir que le traitement qui arrive en seconde position est le traitement de

flou  $T_1$ . Cela veut dire que pour cette métrique l'image floue (et donc mauvaise au sens où nous l'entendons) est plus proche de l'image de référence que l'image issue du traitement  $T_2$ .

La seule métrique ici qui suit le classement intuitif est VIF en classant les images issues des traitements de la manière suivante :  $T_3 > T_2 > T_1$ . Nous pouvons voir que les métriques VSI et FSIMc donnent de meilleurs résultats pour le traitement de flou  $T_1$  que pour les traitements  $T_2$  et  $T_3$ , ce qui n'est donc pas cohérent avec le niveau de détail présent dans les images, mais cela veut aussi dire que la distance entre l'image source et l'image floue  $T_1$  est plus faible comparée aux autres traitements.

Tout comme ces deux métriques, le PSNR ne donne pas les indications souhaitées. Cependant, cela vient de la méthode de mesure de cette métrique, le fait de faire la différence pixel à pixel n'est pas utilisable dans les traitements d'amélioration de la qualité. En effet, les contours améliorés sont parfois meilleurs que les contours présents dans l'image source. De plus, il suffit qu'il y ait un décalage d'au minimum un pixel entre les contours présents dans l'image source et ceux présents dans les images issues de  $T_2$  et  $T_3$  pour donner de mauvais résultats et augmenter les différences pixel à pixel entre les deux images. Ensuite il est possible que certaines images de  $S_{métrique}$  ne comportent pas assez de piqué, de ce fait l'image sera généralement floue. Dans ce cas, le traitement  $T_1$  sera donc celui qui se rapproche le plus de l'image source.

### 3.1.4/ DISCUSSION ET PROPOSITION

Il a été montré dans la section 3.1.3.2 que la plupart des métriques pleine-référence ne donnent pas de résultats cohérents avec le classement *a priori* des trois traitements appliqués aux différentes images. Ces métriques donnent des meilleurs résultats pour le traitement de flou  $T_1$  que pour les traitements de renforcement des contours  $T_2$  et  $T_3$ . Seule la métrique VIF permet d'avoir des résultats pertinents pour les traitements d'amélioration de la qualité.

Ensuite, concernant les métriques sans-référence, section 3.1.3.1, là aussi très peu de ces métriques donnent des résultats pertinents pour notre étude. Par exemple la métrique BRISQUE ne suit pas du tout le jugement *a priori*, de ce fait elle ne sera pas prise en compte dans l'évaluation des traitements que nous mettons en place pour l'amélioration de la qualité. BRISQUE étant une métrique basée sur de l'apprentissage de déformations spécifiques et de notation humaine, le fait que l'amélioration de la résolution ne fasse pas partie des déformations présentes dans la base d'apprentissage explique la raison de cette incohérence des résultats. Nous évaluerons cependant les métriques sans référence PSI, EME, CPBD et NIQE tout en prenant en compte le fait que ces métriques se basent généralement sur la mesure des contours ou du contraste pour juger de la qualité d'une image : plus l'image a des contours et des contrastes élevés, plus l'image est jugée comme étant de bonne qualité. Comme il a été possible de voir dans la section 2.2, les métriques sans-référence ont pour objectif de donner des résultats aussi proches qu'une évaluation humaine. Ce rapprochement des résultats des métriques subjectives avec le jugement humain est nécessaire car dans la majorité des cas l'humain est l'utilisateur final des images améliorées.

Suite à cela nous réalisons une sélection de 5 métriques VIF, PSI, EME, CPBD et NIQE. En effet, les métriques sélectionnées permettent d'avoir des résultats assez pertinents et cohérents avec le ressenti humain. De plus, le principe de fonctionnement du traitement  $T_2$  est similaire à la méthode de Super Résolution de l'auteur Gohshi, sélectionnée dans l'état de l'art des techniques de Super Résolution. Cette méthode améliore les contours d'une image interpolée.

Pour compléter les évaluations automatiques, nous avons décidé de mettre en place des évaluations psychovisuelles humaines. Ces évaluations psychovisuelles PVA, *Psychovisual Assessment*,



### 3.1. ÉVALUATION DE LA PERTINENCE DES MÉTRIQUES DE LA QUALITÉ DE L'IMAGE65

permettent de donner directement le ressenti humain. Cependant ces évaluations sont totalement tributaires des goûts des personnes réalisant l'évaluation.

Plusieurs outils ont déjà été créés par le passé pour réaliser diverses études psychovisuelles comme par exemple l'outil Quickeval [161], développé par le laboratoire Norvégien NTNU ColorLab, qui permet de réaliser diverses études psychovisuelles, avec une ou plusieurs images à comparer ou à classer. Cet outil permet de réaliser des études en ligne : de ce fait, les conditions d'évaluation (éclairage, écrans : calibration des couleurs et résolution) ne sont pas fixes pour l'ensemble des membres du panel de test. Toutefois, cet outil pourrait être utilisé sur une seule et même station de travail afin de contrôler parfaitement l'environnement de l'évaluation, malheureusement cet outil n'est plus disponible à l'heure actuelle.

Nous avons donc créé un outil logiciel permettant de réaliser des études PVA de manière simple pour le testeur, mais aussi pour la gestion des résultats. Le panéliste a la possibilité de classer et de choisir l'image qui lui paraît être la meilleure pour suivre le protocole de test.

La figure 3.4 présente une version de notre outil dans le cas où le panéliste doit classer trois images en fonction de son appréciation par rapport à une image de vérité de terrain affichée à gauche. L'interface de cet outil n'est pas primordiale, car elle varie en fonction du protocole de test. L'outil permet, à la demande, d'afficher les différentes images de manière aléatoire ce qui permet de faire varier la position des différents stimuli de l'évaluation pour éviter que le panéliste fasse l'évaluation en suivant une sorte de routine.

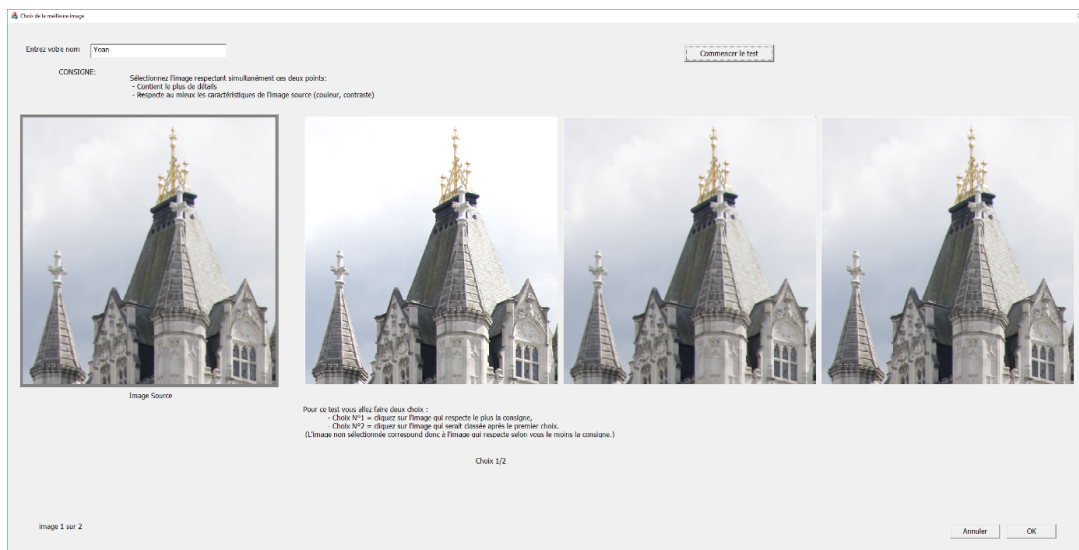


FIGURE 3.4 – Capture d'écran de l'application développée pour réaliser un test psychovisuel PVA de manière automatique

Cet outil, développé spécifiquement pour nos travaux, n'est pas fixe et évolue à la demande en fonction des protocoles de tests, de la procédure et des stimuli. L'avantage de cet outil est le fait de pouvoir réaliser des tests à la suite de manière simple et de mesurer la durée d'évaluation de chaque panéliste. Cette durée peut entrer en considération pour la validation des résultats du panéliste. Le principal inconvénient des évaluations psychovisuelles est la difficulté à recruter un panel important. D'autre part, ce genre de métrique est très chronophage comparée aux métriques courantes de l'état de l'art. Mais elle présente une réelle appréciation humaine. Cet outil permettant de réaliser une étude psychovisuelle est utilisé par la suite pour évaluer la méthode de Gohshi et notre méthode LASSR.

## 3.2/ ÉVALUATION DE LA MÉTHODE DE GOHSHI

### 3.2.1/ PRÉSENTATION COMPLÈTE

La méthode de Gohshi a été sélectionnée dans l'état de l'art comme base de notre implantation du traitement de Super Résolution. En effet, dans [128] l'auteur met en avant le fait que sa méthode ne nécessite pas d'avoir d'images enregistrées en mémoire et travaille seulement sur l'image courante. L'auteur met aussi en avant le fait que cette méthode ajoute des hautes fréquences dans les images HR à partir des images basse résolution LR. Le but de cette partie est de présenter entièrement les éléments qui composent la méthode de Gohshi.

L'auteur n'ayant pas donné dans ses articles l'intégralité des modalités de calcul de sa méthode, nous en avons repris le principe général, mais nous avons dû le paramétrer expérimentalement. Il est donc possible que nos résultats soient différents de ceux des articles originaux.

La figure 3.5 rappelle les différentes parties qui composent la méthode de Gohshi : la phase d'interpolation, le filtre passe haut, le filtre non linéaire suivi du limiteur, ce qui donne l'image accentuée, et enfin l'addition entre l'image interpolée et l'image accentuée. Chaque partie fait l'objet d'une description dans les sections qui suivent. Nous présenterons toutefois en premier lieu le choix de l'espace couleur de travail, car l'intégralité de la chaîne de traitement peut être appliquée sur la luminance ou sur des composantes de chrominance.

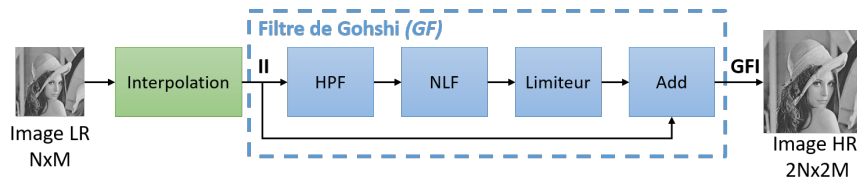


FIGURE 3.5 – Principe de la méthode de Gohshi

#### 3.2.1.1/ ESPACE DE COULEUR DE TRAVAIL

Le but de cette étape préliminaire est de sélectionner un espace colorimétrique de travail dans lequel les résultats d'amélioration de la qualité seront les meilleurs.

Dans [99], Asmare présente une étude comparative pour la sélection des espaces de couleurs adaptée aux traitements d'amélioration de la qualité des images. Il compare les espaces de couleurs de travail les plus fréquemment utilisés dans la littérature. Pour l'auteur il est nécessaire de conserver en priorité les contours des images pour préserver et améliorer leur qualité visuelle.

Pour ses travaux, l'auteur dispose d'un ensemble d'images formé par une image idéale et une image dégradée représentant la même scène. L'image dégradée est une image capturée avec de mauvaises conditions d'éclairage. L'auteur convertit tout d'abord les images dégradées RGB dans huit espaces de couleur différents : HSV, HSI, LAB, LUV, LCH, YCbCr, YDbDr et YUV. Il applique ensuite un traitement des contours permettant d'améliorer la qualité de l'image finale. À la suite de ces traitements les images sont converties dans l'espace colorimétrique de départ (RGB). Pour classer les espaces colorimétriques, l'auteur évalue finalement la qualité des images traitées en comparant la similarité de l'image traitée avec l'image d'origine *idéale*. La moyenne des valeurs de la métrique SSIM est utilisée pour réaliser le classement des espaces colorimétriques utilisés pour l'amélioration de la qualité des images. L'espace de couleurs HSI est classé comme

étant le plus pertinent pour les traitements d'amélioration de la qualité de l'image, et cela est dû, pour l'auteur, au fait que les espaces représentant l'intensité et les informations de la couleurs sont clairement séparés.

Une majorité d'auteurs de la littérature dans le domaine de la super résolution précisent ne travailler que sur le canal représentant la luminance [194, 146, 179, 159]. Comme Dong et al. [146] appliquent les traitements de SR seulement sur le canal correspondant à la luminance car la vision humaine est, selon eux, plus sensible aux changements présents sur ce canal.

C'est pourquoi, pour le développement de la méthode de Gohshi, nous avons choisi d'appliquer la méthode sur le canal correspondant à la luminance dans l'espace de couleur HSV (canal  $V$ ). De plus la formule de conversion entre les espaces RGB vers HSV est mathématiquement simple ce qui facilite l'intégration de cette conversion dans une cible matérielle.

### 3.2.1.2/ INTERPOLATION

Dans [175], Parsania et al. comparent plusieurs méthodes d'interpolation d'images, en fonction de leur complexité calculatoire. Les méthodes des plus proches voisins (*Nearest Neighbor*), Bili-néaire, bicubique, cubique b-spline, catmull-Rom, Mitchell-Netravali et Lanczos sont comparées dans cette étude, qui montre que l'interpolation de Lanczos (ordre 3) est la plus complexe à mettre en place, et ses résultats sont assez proches de la méthode bicubique.

Dans [155], l'auteur compare la qualité de différentes méthodes d'interpolation (bilinéaire, b-spline, Lanczos). L'auteur procède à une comparaison d'une image de référence  $I_{ref}$  qui a été décimée pour être à nouveau interpolée  $I_{im}$ . Le PSNR est alors calculé entre les deux images. L'auteur montre pour un exemple de dix images, de résolutions différentes, que le PSNR est le meilleur lorsque les images sont interpolées avec la méthode de Lanczos.

Dans ses publications, Gohshi [139, 147] précise qu'il utilise dans ses exemples l'interpolation de Lanczos. Cependant son choix de méthode d'interpolation n'est pas constant. Il laisse généralement le choix entre une interpolation bicubique et une interpolation de Lanczos [12] (d'ordre 2 ou 3).

Étant donné sa position dominante dans la littérature, et les bons résultats qu'elle permet d'obtenir, nous avons donc sélectionné l'interpolation de Lanczos 2 dans les calculs qui suivront. Sa complexité calculatoire est toutefois plus élevée que celle de l'interpolation bicubique, bilinéaire ou des plus proches voisins *nearest Neighbor*.

L'interpolation de Lanczos [28] pour un signal à une dimension (1-D) est explicitée dans l'équation 3.1.

$$w(x) = \Psi_{Ln}(x).sinc(x) \quad (3.1)$$

Le noyau d'interpolation de Lanczos  $\Psi_{Ln}(x)$  est défini dans l'équation 3.2, avec  $n$  l'ordre du filtre.

$$\Psi_{Ln}(x) = \begin{cases} 1 & \text{pour } |x| = 0 \\ \frac{\sin(\pi x/n)}{\pi x/n} & \text{pour } 0 < |x| < n \\ 0 & \text{pour } |x| \geq n \end{cases} \quad (3.2)$$

Le filtre de Lanczos comporte  $2n - 1$  lobes, un lobe positif au centre puis une alternance de lobes négatifs et positifs de chaque cotés.

L'équation 3.3 représente l'interpolation 2-D. Le noyau d'interpolation 2-D est séparable en  $x$  et  $y$ .

$$W_{Ln}(x, y) = w_{Ln}(x) \cdot w_{Ln}(y) \quad (3.3)$$

$$\begin{aligned} \hat{I}(x_0, y_0) &= \sum_{v=\lfloor y_0 \rfloor - n + 1}^{\lfloor y_0 \rfloor + n} \left[ \sum_{u=\lfloor x_0 \rfloor - n + 1}^{\lfloor x_0 \rfloor + n} [I(u, v) \cdot W_{Ln}(x_0 - u, y_0 - v)] \right] \\ &= \sum_{j=0}^{2n-1} \left[ w_{Ln}(y_0 - v_j) \cdot \sum_{i=0}^{2n-1} [I(u_i, v_i) \cdot w_{Ln}(x_0 - u_i)] \right] \end{aligned} \quad (3.4)$$

Avec  $\lfloor x \rfloor$  est la fonction retournant la partie entière de  $x$ .  $u_i = \lfloor x_0 \rfloor + i - n + 1$  et  $v_i = \lfloor y_0 \rfloor + j - n + 1$

### 3.2.1.3/ FILTRE PASSE HAUT HPF

Le filtre passe haut ou HPF permet de mettre en valeur les détails et les contours, avant que ces derniers ne soient amplifiés par la fonction non linéaire.

D'après [137], le filtre HPF utilisé par Gohshi est un filtre du second ordre basé sur le Laplacien, dont les coefficients ne sont pas précisés. Le filtre Laplacien est un filtre de convolution généralement utilisé pour mettre en avant les détails qui ont une variation rapide de la luminosité. Il s'agit d'un filtre de la catégorie des "zero crossing", en effet lorsque le résultat de ce filtre passe par zéro cela met en avant un gradient. D'un point de vue mathématique le Laplacien est une dérivée de second ordre à deux dimensions (équation 3.5).

$$L(x, y) = \frac{\partial^2 I(x, y)}{\partial x^2} + \frac{\partial^2 I(x, y)}{\partial y^2} \quad (3.5)$$

Les formes habituelles des noyaux  $3 \times 3$  Laplacien sont présentées dans l'équation 3.6. Il s'agit alors d'approximations réalisant des dérivées secondes. Il est à noter que la somme des coefficients doit être nulle. La détection de contour est réalisée simplement par une convolution d'un noyau Laplacien avec l'image.

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (3.6)$$

Nous mettons en place un filtre de type Laplacien 2-D basé sur une gaussienne. Ce filtre extracteur de contours correspond au filtre utilisé dans la fonction d'amélioration des contours *imsharpen* présente dans Matlab. La formulation du noyau gaussien utilisé est présentée dans l'équation 3.7.

$$F_{gauss}(i, j) = e^{-\frac{i^2 + j^2}{2\sigma^2}} \quad (3.7)$$

Avec  $i$  et  $j$  les coordonnées dans le noyau, et  $\sigma$  l'écart type désiré. Le noyau généré  $F_{gauss}$ , de dimension  $5 \times 5$ , de  $\sigma = 1$ , est présenté dans l'équation 3.8. Les coefficients de ce noyau sont arrondi pour l'affichage.

$$F_{gauss} = \begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.983 & 0.0596 & 0.0133 \\ 0.0219 & 0.983 & 0.1621 & 0.983 & 0.0219 \\ 0.0133 & 0.0596 & 0.983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \quad (3.8)$$

À partir du filtre Gaussien  $F_{gauss}$ , un filtre extracteur de contours  $F_{extrac}$  est généré en suivant l'équation 3.9. La somme des coefficients de  $F_{gauss} = 1$ , cependant la somme des coefficients d'un filtre laplacien doit être égale à 0.

$$F_{extrac} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.983 & 0.0596 & 0.0133 \\ 0.0219 & 0.983 & 0.1621 & 0.983 & 0.0219 \\ 0.0133 & 0.0596 & 0.983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \quad (3.9)$$

Le résultat de cette opération nous donne le noyau présenté dans l'équation 3.10. La somme des coefficients est alors égale à 0.

$$F_{extrac} = \begin{bmatrix} -0.0030 & -0.0133 & -0.0219 & -0.0133 & -0.0030 \\ -0.0133 & -0.0596 & -0.0983 & -0.0596 & -0.0133 \\ -0.0219 & -0.0983 & 0.8379 & -0.0983 & -0.0219 \\ -0.0133 & -0.0596 & -0.0983 & -0.0596 & -0.0133 \\ -0.0030 & -0.0133 & -0.0219 & -0.0133 & -0.0030 \end{bmatrix} \quad (3.10)$$

Pour des raisons de simplification, les coefficients de  $F_{extrac}$  sont modifiés pour qu'ils ne soient composés que de nombres entiers. Cela permet de faciliter le stockage des coefficients et de simplifier les opérations arithmétiques de convolution pour que le traitement soit plus embarquable dans une cible matérielle. Pour réaliser cette simplification nous multiplions les coefficients par 256, de ce fait, une normalisation par 256 est prévue par la suite. Nous utilisons dans la suite de ces travaux le noyau pré-calculé présenté dans l'équation 3.11. Cette approximation crée une imprécision dans le noyau et le rend différent du noyau présent originalement dans la fonction *imsharpen* de Matlab. Du fait des différents arrondis la somme des coefficients n'est plus égale à 0, de ce fait nous modifions alors le coefficient central du noyau.

$$F_{extrac} = \frac{1}{256} \times \begin{bmatrix} 0 & -3 & -5 & -3 & 0 \\ -3 & -15 & -25 & -15 & -3 \\ -5 & -25 & +204 & -25 & -5 \\ -3 & -15 & -25 & -15 & -3 \\ 0 & -3 & -5 & -3 & 0 \end{bmatrix} \quad (3.11)$$

Nous présentons dans la figure 3.6 la différence de résultats obtenu avec la convolution d'un signal "marche" de 0 à 255 (fonction échelon) avec les deux noyaux  $F_{extrac}$  arrondi et non arrondi. La différence des convolutions est inférieure à 5.

#### 3.2.1.4/ FONCTION NON LINÉAIRE *NLF*

Gohshi précise que la fonction non-linéaire permet de générer des harmoniques à des fréquences supérieures aux fréquences présentes dans l'image source LR. Ces harmoniques sont générées au

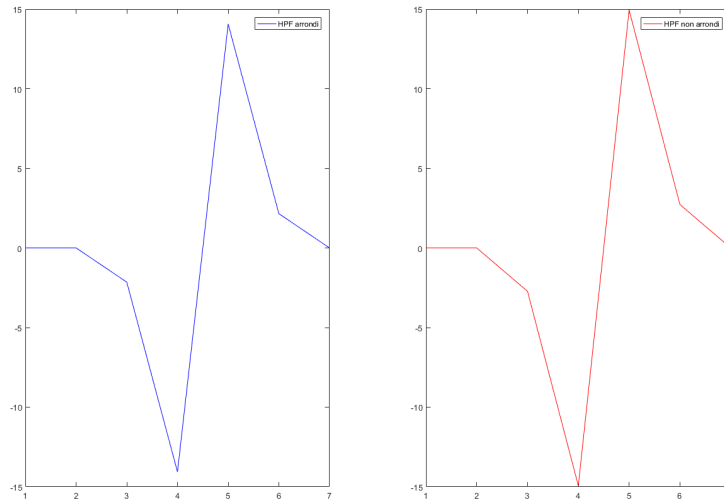


FIGURE 3.6 – Comparaison des résultats obtenus avec la convolution d'un signal "marche" de 0 à 255 (fonction échelon) avec les deux noyaux  $F_{extract}$  arrondi (à gauche) et non arrondi (à droite).

niveau des contours qui ont été amplifiés par le filtre passe-haut. D'après ce raisonnement, le filtre ne devrait pas générer d'harmonique dans les zones "lisses", les zones peu texturées représentant des fréquences très faibles.

Gohshi utilise dans [137, 147] la fonction non linéaire cubique  $y = x^3$ . Cette fonction a notamment la propriété de conserver le signe issu de l'opérateur de type Laplacien. Selon l'auteur, d'autres fonctions non-linéaires peuvent être utilisées, l'important étant de conserver le signe du signal issu du filtre passe-haut HPF.

Un exemple d'application est présenté dans les figures 3.7 et 3.8. Cet exemple est réalisé sur une image de synthèse représentant un signal échelon basé sur une sigmoïde (figure 3.7). Le résultat du filtre extracteur de contours  $F_{extract}$  est illustré sur la courbe présente dans la figure 3.8. Cette courbe correspond à une ligne extraite dans l'image 3.7-a. Les fonctions non-linéaires  $y = x^3$  et  $y = x^5$  sont appliquées au résultat du filtre passe-haut HPF.

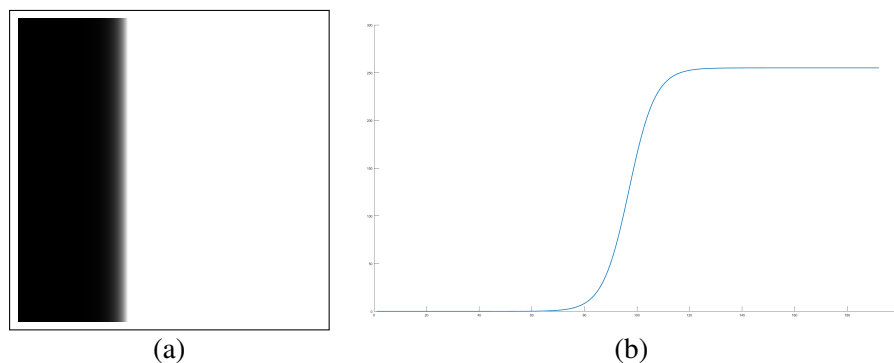


FIGURE 3.7 – Image de synthèse "échelon" de type sigmoïde (a), et visualisation d'une ligne de cette image sur le graphique représenté en (b). L'amplitude de ce signal est de 0 à 255.

Comme attendu les fonctions non linéaires amplifient le signal de sortie du filtre extracteur de contours tout en conservant son signe comme nous pouvons le voir dans la figure 3.8-b.

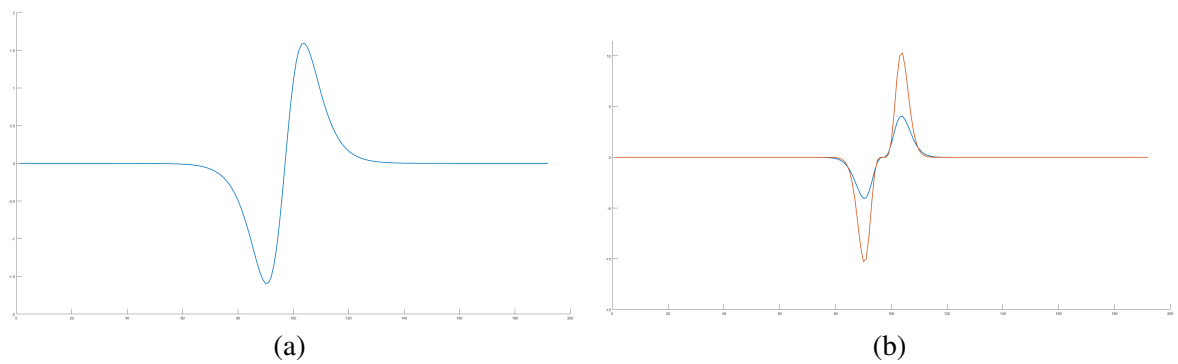


FIGURE 3.8 – La courbe présente en (a) correspond au résultat du filtre passe-haut HPF sur l'image 3.7-a, les courbes de (b) représentent le résultat de la fonction non linéaire  $y = x^3$  (bleue) et  $y = x^5$  (rouge).

Pour approfondir l'exemple d'application des fonctions non linéaires, le même test est réalisé sur une image réelle présentée dans la figure 3.9-a. Une ligne de cette image est sélectionnée au hasard afin de représenter dans la figure 3.9-b toutes ses variations en niveaux de gris.

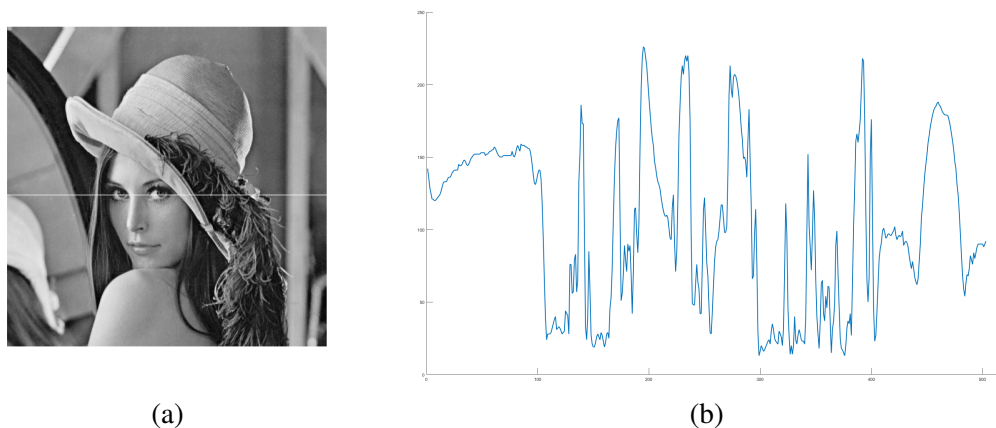


FIGURE 3.9 – Image réelle de test "Lenna" présentée en (a) dont une ligne est sélectionnée "blanche" pour la visualisation des résultats. La courbe présente en (b) illustre les variations en niveaux de gris le long de la ligne blanche.

Comme précédemment le filtre extracteur de contours  $F_{extract}$  est appliqué sur cette image, et le résultat de ce filtre HPF sur la ligne est présenté dans la figure 3.10.

Le rôle de cette fonction non linéaire, dont la dérivée est quadratique et positive et donc à croissance rapide, est d'amplifier fortement les petites valeurs de Laplacien, donc d'accentuer fortement les contours. Toutefois, cette accentuation peut conduire à des dépassements de capacité (valeurs supérieures à 255). En effet, dans la figure 3.11, une grande partie des valeurs des courbes ne font pas partie de l'intervalle  $[-255, 255]$  : 52.5% des résultats de la fonction non linéaire  $y = x^3$  sortent de cet intervalle, et 70.3% pour  $y = x^5$ . Ces dépassements impliquent d'utiliser le limiteur présenté dans la section 3.2.1.5. En effet, les signaux d'entrée de cette fonction sont compris entre  $-255$  et  $255$ , donc les signaux de sortie de la fonction seront compris entre  $(-255)^3$  et  $255^3$  pour l'utilisation de  $y = x^3$ .

Les résultats des fonctions non linéaire  $y = x^3$  et  $y = x^5$  sont ajoutés à l'image source et les images résultantes sont présentées dans la figure 3.12. Cette figure montre que lorsqu'une fonction de

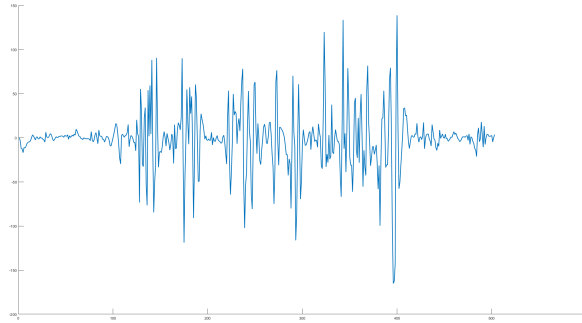


FIGURE 3.10 – Résultat du filtre passe-haut du second ordre appliqué sur l'image de Lenna, seuls les résultats correspondants à la ligne blanche de la figure 3.9-a sont présentés.

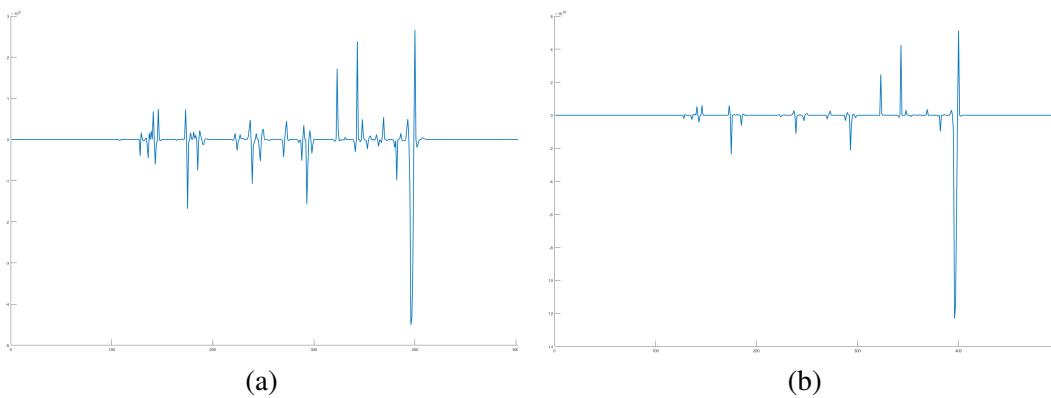


FIGURE 3.11 – Le signal en (a) représente le résultat de la fonction non linéaire  $y = x^3$  appliqué sur l'image de contours et le signal en (b) les résultats de  $y = x^5$ .

degré supérieur est utilisée cela crée des artefacts plus importants dans l'image finale. Les images sont générées en ajoutant le signal sortant du filtre passe haut HPF traité par les fonctions non linéaires à l'image d'origine. Nous utilisons par la suite la fonction non linéaire cubique  $y = x^3$ .

### 3.2.1.5/ LIMITATION & FUSION

Le limiteur introduit par Gohshi permet de contraindre la dynamique entre  $-255$  et  $255$ . Il est possible pour cela de réaliser un simple écrêtage, ou d'utiliser une fonction de limitation plus complexe, éventuellement stockée sous forme de LUT. Cette fonction doit conserver le signe du Laplacien amplifié, avant l'addition finale avec l'image interpolée. Différentes méthodes de limitation sont présentées dans la figure 3.13.

Nous avons implanté le limiteur de type écrêtage franc qui correspond à l'équation 3.12.

$$Pix_{LMT} = \begin{cases} 255, & \text{si } Pix_{NLF} \geq 255 \\ -255, & \text{si } Pix_{NLF} \leq -255 \\ Pix_{NLF}, & \text{ailleurs} \end{cases} \quad (3.12)$$

Les valeurs résultantes  $Pix_{LMT}$  du bloc limiteur sont alors comprises entre  $-255$  et  $255$ . Ces valeurs sont ensuite ajoutées au pixel interpolé  $Pix_{inter}$  correspondant, en suivant l'équation 3.13.





FIGURE 3.12 – Comparaison de différentes fonctions non linéaires appliquées sur le signal de contours :  $y = x^3$  (a),  $y = x^5$  (b), le résultat des fonctions non linéaires est ajouté à l'image d'origine 3.11-a

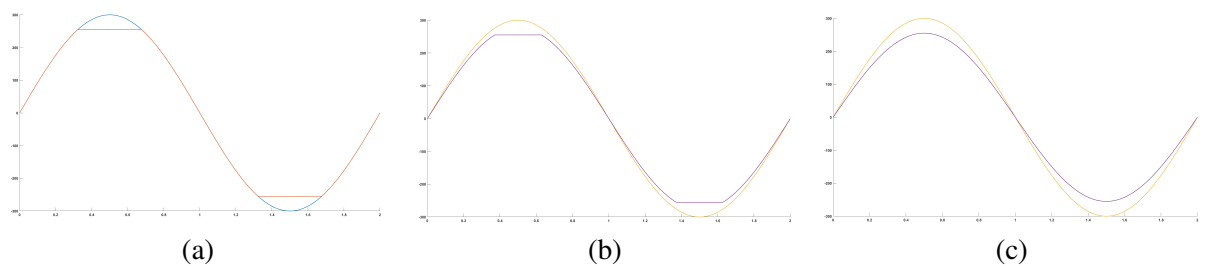


FIGURE 3.13 – Comparaison de différentes techniques de limitation du signal. (a) correspond à un écrêtage franc, (b) correspond à un écrêtage plus doux et (c) correspond à une fonction de limitation qui peut être assimilée à une compression.

$$Pix_{Gohshi} = Pix_{inter} + \beta \times Pix_{LMT} \quad (3.13)$$

Nous avons introduit le coefficient  $\beta = 0.03$  afin d'éviter un nouveau dépassement de capacité après addition. Ce coefficient a été déterminé de manière empirique.

### 3.2.2/ ANALYSES CRITIQUES

Nous avons implanté notre version de la méthode de Gohshi de manière logicielle afin d'en évaluer les performances sur des images variées. Nous avons ainsi créé des images HR 8k à partir d'images LR 4k, qui elles-mêmes provenaient d'une base de 22 images 8k sous-échantillonnées. Nous disposons donc de la vérité de terrain.

### 3.2.2.1/ ANALYSE VISUELLE

Dans un premier temps, nous avons mené une analyse purement visuelle du résultat, pour des images dites naturelles représentant une grande variété de scènes allant de paysages aux portraits et présentant de nombreux niveaux de détails. Nous avons mis en évidence quelques points forts et quelques faiblesses :

- Points forts : le traitement de Gohshi permet de fournir des images hautes résolutions à partir d'images moins résolues, et ce, en ayant de grands niveaux de détails. La méthode semble améliorer la sensation de détail dans les zones de l'image contenant des contours (reliefs).
- Points faibles : il a été possible de voir que, comme dans la plupart des traitements de ce type, des artefacts apparaissent dans les zones lisses (les zones représentant des faibles fréquences spatiales) et ces artefacts ressortent visuellement comme du bruit et non un ajout de détail comme il est prévu dans la méthode.

La figure 3.14 met en évidence l'amélioration des contours (rectangle 1), mais aussi l'apparition de bruit dans une zone "lisse" (rectangle 2).

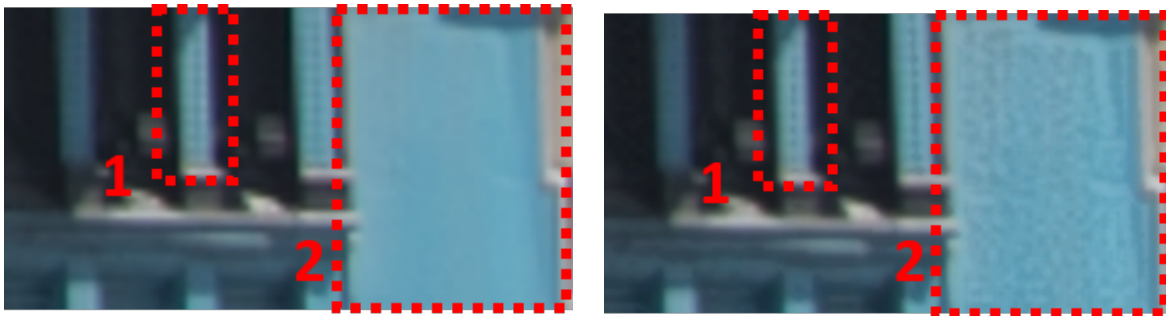


FIGURE 3.14 – Différences entre une interpolation de Lanczos (gauche) et la méthode de Gohshi(droite) pour les zones 1,2

### 3.2.2.2/ TESTS APPROFONDIS

Dans le but de quantifier l'impact des pertes d'informations (saturations) et des artefacts dans les zones lisses de l'image, nous avons mis en place une comparaison des résultats du traitement de Gohshi avec une interpolation de Lanczos, en fonction du niveau de détail de l'image.

D'après l'étude réalisée sur les métriques de l'image, présentée section 3.1.4, nous avons décidé d'utiliser les métriques *automatiques* SSIM et PSI pour mesurer respectivement la similarité de l'image traitée avec l'image source, et de mesurer l'acutance de l'image traitée. En considérant alors que l'acutance est un élément important dans le jugement de la netteté et du niveau de détail d'une image.

Un ensemble de données  $S_1$  a été créé afin de pouvoir évaluer la méthode de Gohshi.  $S_1$  est composé de 350 imagettes  $128 \times 128$  pixels extraits manuellement de 35 photographies représentant des scènes variées (paysages et portraits). Ces images ont été capturées avec un appareil photo Nikon D800 avec une résolution spatiale de 8k.

Quelques exemples de  $S_1$  sont présentés dans la figure 3.15. Ces imagettes ont été annotées par un expert en utilisant le label  $y_i = \{BF; MF; HF\}$  où  $BF$ ,  $MF$  et  $HF$  sont respectivement les labels pour basses fréquences, fréquences moyennes et hautes fréquences. L'ensemble  $S_1$  comporte 100 imagettes peu texturées ( $BF$ ), 100 imagettes moyennement texturées ( $MF$ ) et 150 imagettes fortement texturées ( $HF$ ). L'ensemble est défini comme suit :  $S_1 = \{P_i, y_i\}_{i=1}^{350}$



FIGURE 3.15 – Exemples provenant de l'ensemble  $S_1$ , triés en fonction de la texture. De haut en bas : hautes fréquences  $HF$ , fréquences moyennes  $MF$  et basses fréquences  $BF$

Ces métriques *automatiques* sont complétées par une évaluation psychovisuelle  $PVA_1$ . L'évaluation  $PVA_1$  a été mise en place en utilisant l'outil que nous avons créé lors de l'étude sur les métriques. La définition du protocole de  $PVA_1$  est la suivante :

- Panel : Le panel est composé de 20 experts provenant du laboratoire ImVia/Le2i.
- Stimuli : Chaque membre du panel doit évaluer et comparer deux images, l'une provenant du traitement de Gohshi, et l'autre d'une simple interpolation. Les images sont affichées aléatoirement à gauche ou à droite de l'écran. Chaque panéliste évalue les 350 imagettes de  $S_1$ .
- Procédure : Le panéliste doit sélectionner l'imagette qui contient selon lui le plus haut niveau de détail et de qualité. Le test se déroule sur un seul et même ordinateur dont l'écran est calibré. Le test est réalisé dans une pièce modérément éclairée.

### Résultats

Les résultats de cette étude sont résumés dans le tableau 3.3.

PSI et SSIM donnent un résultat proche de 1 lorsque ce résultat est considéré "bon", et 0 dans le cas contraire. Pour PSI, le résultat est considéré bon lorsqu'il y a un grand niveau de détails. Pour SSIM, le résultat est considéré bon lorsque les deux images comparées sont très proches.

$PVA_1$  donne un résultat en pourcentage, qui correspond au pourcentage de fois que le traitement a été sélectionné par le panel comme étant le meilleur parmi les deux traitements.

Il est possible de voir dans le tableau 3.3 que la différence entre LII et GFI n'est pas significative avec l'utilisation des métriques SSIM et PSI. La différence est plus importante avec les résultats de l'évaluation psychovisuelle  $PVA_1$ .

Les résultats  $PVA_1$  obtenus montrent que le panel préfère le traitement de Gohshi GFI lorsqu'il est appliqué sur des images contenant des hautes  $HF$  et moyennes  $MF$  fréquences. Ces résultats montrent que pour les images basses fréquences  $BF$  de meilleurs résultats sont obtenus avec l'interpolation de Lanczos LII. Par conséquent, cette évaluation démontre que la texture de l'image influe sur la qualité des résultats de Gohshi.

		<b>HF</b>	<b>MF</b>	<b>BF</b>
<b>GFI</b>	<b>PSI</b>	0.25	0.23	0.48
	<b>SSIM</b>	0.89	0.88	0.93
	<b>PVA(%)</b>	<b>76.48</b>	<b>63.05</b>	40.54
<b>LII</b>	<b>PSI</b>	0.27	0.20	0.47
	<b>SSIM</b>	0.91	0.91	0.94
	<b>PVA(%)</b>	23.52	36.95	<b>59.46</b>

TABLE 3.3 – Comparaison entre l'image provenant du filtre de Gohshi (GFI) et l'image interpolée avec Lanczos (LII) en fonction du niveau de fréquence de l'image (*HF*, *MF* et *BF*)

Comme il était possible de le penser, cette approche est efficace pour la partie très texturée de l'image. Néanmoins, elle ajoute du bruit dans les zones peu texturées pour lesquelles une interpolation Lanczos est préférable. De plus, la méthode de Gohshi augmente la luminance globale de l'image.

Ensuite, comme les résultats de *MF* et de *HF* ne sont pas significativement différents pour les différentes métriques (PSI, SSIM et PVA), il est donc possible de fusionner ces deux classes *MHF* pour les prochaines expériences.

Basé sur les observations précédentes, nous proposons de mettre en place une approche hybride locale, mélangeant la méthode de Gohshi et l'interpolation en fonction du niveau de texture local. Cette nouvelle approche, appelée LASSR pour Local Adaptive Spatial Super Resolution (Super résolution spatiale adaptative localement) est détaillée dans la section 3.3.

### 3.3/ SOLUTION PROPOSÉE : LASSR

Afin de limiter les effets indésirables du filtre de Gohshi, notamment le fait que du bruit apparaisse dans les zones à basses fréquences, nous avons mis en place un traitement adaptatif local en fonction du niveau de texture. Le principe de ce traitement nommée "LASSR" pour *Super Résolution Spatiale Adaptative Localement* est présenté dans la Fig.3.16. Cette méthode a pour but de fusionner les images issues d'une interpolation de Lanczos (LII) avec les images provenant de la méthode de GOHSHI (GFI), et ceci en fonction de la texture.

Cette méthode peut être vue comme une évolution du filtre de Gohshi, ajoutant deux étapes de traitement supplémentaires : une classification de texture (TC) et une étape de fusion.

La classification de texture sera explicitée dans la section 3.3.1. Nous décrirons tout d'abord l'état de l'art des techniques couramment utilisées dans le domaine, et notre choix d'implantation.

Ensuite la partie associée à la fusion finale sera explicitée dans la section 3.3.2.

#### 3.3.1/ CLASSIFICATION DE TEXTURE

Un grand nombre de paramètres sont disponibles dans la littérature permettant de quantifier les fréquences locales, autour du pixel  $p(x, y)$ , dans l'image. Les approches classiques sont généralement basées sur les transformées de Fourier [18], les transformées en ondelettes [64], les

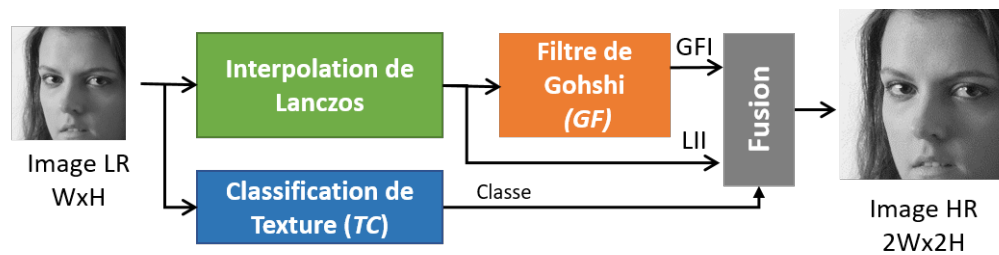


FIGURE 3.16 – Méthode de Super Résolution Spatiale Adaptative Localement en fonction du niveau de texture (LASSR).

paramètres d'Haralick [13], les paramètres de Laws [14] ou encore sur des approches statistiques standards. Des approches basées sur le Deep Learning [164, 188] et les réseaux convolutionnels (CNN)[163] pourraient également répondre à nos besoins.

Cependant notre approche nécessite seulement de séparer l'image en deux classes, correspondant aux zones lisses et aux zones texturées. Il nous est inutile pour notre approche d'être aussi précis que Laws [14]. En effet, les paramètres de Laws permette de connaître la nature exacte de la texture (tissu, eau, herbes ...).

D'autre part, les contraintes liées à l'implantation matérielle temps réel restreignent le champ d'investigation des méthodes d'analyse de texture. Nous avons donc choisi de travailler dans le domaine spatial, et de baser le calcul des attributs de texture sur des statistiques standards calculées à partir du gradient de Sobel  $SGI$  de l'image basse résolution d'entrée LR. En effet, une méthode de type Machine Learning traditionnelle est tout à fait suffisante pour ce type de segmentation. Comparée au Deep Learning, ce type de méthode nécessite peu d'images d'apprentissage, et dans notre cas, les attributs sont bien identifiés (hautes et basses fréquences spatiales) et ne nécessitent pas les nombreuses convolutions d'un CNN.

Il aurait pu être possible d'appliquer un seuillage fixe à partir d'un détecteur de contour, mais une approche par classification automatique permet de rendre l'approche plus robuste, moins arbitraire.

En résumé, nous avons choisi une méthode classique consistant à extraire pour chaque pixel des attributs caractéristiques de la fréquence locale. Ces attributs forment le vecteur d'entrée d'une méthode de classification. La sortie de la classification sera binaire et déterminera si le pixel doit être considéré comme appartenant à une région "lisse" ou "texturée". Nous avons pour cela étudié principalement deux méthodes de classification, dont nous allons rappeler les principes ci-dessous.

### 3.3.1.1/ MÉTHODES DE CLASSIFICATION

Il existe un grand nombre de méthodes compatibles avec l'intégration de la classification dans des systèmes embarqués, comme les réseaux de neurones [55], les Séparateurs à Vastes Marges SVM (*Support Vector Machine*) [144], le Boosting [46], les forêts aléatoires RF (*Random Forest*) [56], les k-NN [17], etc .

Pour cette application il est nécessaire d'avoir une étape de décision très rapide, puisqu'il est nécessaire de prendre une décision à la fréquence du pixel. Pour cette raison nous ne présenterons ici que le principe général de deux méthodes couramment utilisées dans les applications embarquées : les SVM et les méthodes de type Boosting.

Il existe principalement deux types d'apprentissage :

- L'apprentissage *supervisé* : L'apprentissage est alors réalisé à partir d'exemples qui ont été annotés par un expert. L'expert intervient pour créer des couples constitués d'un vecteur décrivant la donnée d'entrée et une classe associée. L'algorithme d'apprentissage construit alors un modèle permettant de séparer les classes.
- L'apprentissage *non-supervisé* : Cette méthode, aussi appelée *clustering*, n'utilise aucune annotation sur les données d'entrée. Les algorithmes d'apprentissage ont pour but de trouver le modèle de la distribution des données. Le principe est de découper l'espace de représentation des échantillons en parties homogènes par le biais d'agrégations. La difficulté principale est de définir le critère d'homogénéité.

Pour cette étude nous utiliserons une méthode supervisée. Un expert a annoté des échantillons avec la classe correspondante. De manière générale la classification se déroule en deux phases distinctes : une phase d'apprentissage (hors-ligne) et une phase de décision (en-ligne) comme illustré dans la figure 3.17.

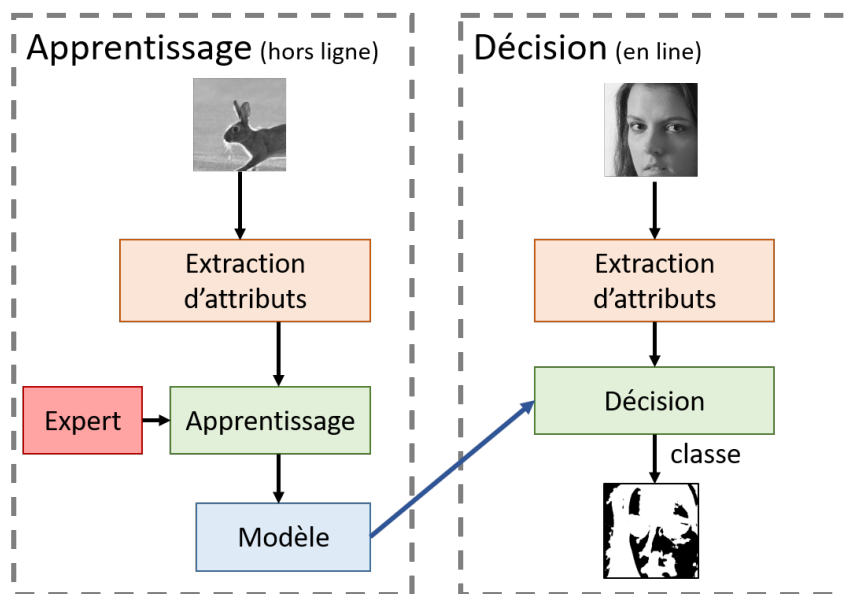


FIGURE 3.17 – Analyse de texture basée sur le Machine learning : Apprentissage hors ligne et Décision de classification en ligne

La phase d'apprentissage a pour but principal de créer un modèle de classification, qui est ensuite utilisé lors de la phase de décision. La phase de décision permet alors de réaliser l'analyse de texture avec les deux classes différentes. La sortie du classifieur est donc, dans notre cas, une information binaire (*BF* ou *MHF*) et cette information sera utilisée dans la phase de Fusion entre les images provenant de la méthode de Gohshi et les images provenant de l'interpolation de Lanczos LII.

Le but de l'étude est de trouver la meilleure solution de classification en vue de l'embarquer dans une caméra intelligente. Du fait de l'implantation matérielle il est nécessaire de trouver un compromis entre le coût matériel, la fréquence d'utilisation et le taux d'erreur globale du classificateur.



### Les Séparateurs à Vastes Marges SVM

Les Séparateurs à Vastes Marges ont été proposés comme nouvelle technique de classification en 1995 par Vapnik et al. [36]. Il s'agit d'une technique très largement utilisée en classification, en régression et aussi en estimation de densité.

Les Séparateurs à Vastes Marges cherchent à séparer au mieux des ensembles de données appartenant à des classes différentes. Chaque donnée est représentée par un vecteur de caractéristiques ou attributs  $x$ . Les fonctions de décision sont présentées dans les équations 3.14 et 3.15. Plus de précisions sur le fonctionnement des SVM et sur les notations des grandeurs utilisées sont présentées en annexe A.

Cas linéairement séparables - Séparateurs linéaires :

$$g(\vec{x}) = \text{signe} \left( \sum_{i=1}^{N_v} \alpha_i y_i \vec{S}_i \cdot \vec{x} + b \right) \quad (3.14)$$

Cas non linéairement séparables - séparateur non linéaire :

$$g(\vec{x}) = \text{signe} \left( \sum_{i=1}^{N_v} \alpha_i y_i K(\vec{S}_i, \vec{x}) + b \right) \quad (3.15)$$

### Le Boosting

Le principe du Boosting, pensé par Freund [35] en 1995, est de produire une fonction de décision non linéaire après plusieurs itérations à partir de modèles prédictifs linéaires faibles. L'idée est donc de combiner des classifieurs dit *faibles* pour créer un classifieur dit *fort*.

La forme initiale du Boosting se déroule en plusieurs étapes :

1. Première hypothèse  $h_1$  sur un sous-ensemble  $S_1$  d'apprentissage de taille  $m_1 < m$  avec  $m$  la taille de  $S$  (l'ensemble d'échantillon d'apprentissage disponible) Il s'agit alors du premier classifieur faible  $C_1$ .
2. Apprentissage d'une seconde hypothèse  $h_2$  du second classifieur  $C_2$  sur un sous-ensemble  $S_2$  de taille  $m_2$  choisi dans  $S - S_1$  dont la moitié des exemples sont mal classés par  $h_1$ .
3. Apprentissage d'une troisième hypothèse  $h_3$  du classifieur  $C_3$  sur  $m_3$  exemples tirés du sous-ensemble  $S_3 = S - S_1 - S_2$  pour lesquels  $h_1$  et  $h_2$  sont en désaccord.
4. Hypothèse finale obtenue par un vote à la majorité des trois hypothèses apprises et qui correspond alors à la règle du classifieur final  $C$  :

$$H = \text{vote majoritaire}(h_1, h_2, h_3)$$

Les sous-ensemble  $S_1, S_2$  et  $S_3$  doivent représenter tout l'ensemble d'apprentissage  $S$ . Ce processus peut être répété plusieurs fois pour chaque classifieur faible. Il est possible de l'améliorer en pondérant des échantillons d'apprentissage. Cette amélioration correspond à l'évolution de Boosting en une variante plus utilisée : Adaboost.

### Le Boosting adaptatif : AdaBoost

Le principe d'Adaboost, proposé par Freund et Schapire [37], est de pondérer les classifieurs en fonction de leur prédiction. De ce fait, cette technique permet de donner plus d'importance aux valeurs difficiles à prédire, ce qui permet de mettre en avant les classifieurs qui réalisent

de meilleurs prédictions. Les classifieurs qui prédisent bien ont un poids plus élevé que les classifieurs qui prédisent moins bien.

Au début du processus, les échantillons ont tous le même poids. Un premier apprentissage est alors réalisé et fournit une première hypothèse de classification  $h_1$ . L'erreur de classification  $e_1$  est alors mesurée en prenant en compte la somme des poids des échantillons mal classés par  $h_1$ . Si le classifieur faible est efficace alors un coefficient  $\alpha$  lui est affecté, augmentant son importance dans le classifieur de décision finale. Les poids des échantillons sont mis à jour. Si l'échantillon est bien classé alors son poids est diminué, et si l'échantillon est mal classé alors son poids est augmenté.

Le processus est répété tant que l'erreur  $e_1 < 0.5$  et que le nombre d'itération  $t < T_{max}$ . Le fonctionnement d'Adaboost est explicité dans l'algorithme 4.

---

**Algorithme 4** Algorithme d'apprentissage Adaboost [134]

---

- 1: Soit une base d'apprentissage  $D_L = \{x_i, y_i\}_{i=1}^p$ , nombre maximal d'itérations  $T_{max}$
- 2: Initialisation des poids de chaque échantillon  $g_i^{(t)} = 1/p \quad \forall i = \{1, \dots, p\}$
- 3: **pour**  $t = 1, \dots, T_{max}$  **faire**
- 4:   Entraînement du classifieur faible à partir des échantillons pondérés  $\{D_L, g^{(t)}\}$  et obtenir l'hypothèse de classification  $h_t : x \rightarrow \{-1, +1\}$
- 5:   Calcul de l'erreur pondérée  $e_t$  de  $h_t : e_t = \sum_{i=1}^p g_i^{(t)} I(y_i \neq h_t(x_i))$
- 6:   Calcul des coefficients  $\alpha = \frac{1}{2} \log \left( \frac{1 - e_t}{e_t} \right)$
- 7:   Mise à jour des poids  $g_i^{t+1} = \frac{g_i^t}{Z_t} \exp \{-\alpha_t y_i h_t(x_i)\}$
- 8:   où  $Z_t$  est une constante de normalisation  $Z_t = 2 \sqrt{e_t(1 - e_t)}$
- 9:   **si**  $e_t = 0$  **ou**  $e_t \geq \frac{1}{2}$  **alors**
- 10:      $T = t - 1$
- 11:   **fin si**
- 12: **fin pour**
- 13: Le classifieur final est alors défini par :

$$H(x) = \text{signe} \left( \sum_{t=1}^T \alpha_t h_t(x) \right) \quad (3.16)$$


---

De plus, l'algorithme Adaboost permet à la fois d'apprendre les frontières de décision et de sélectionner les attributs les plus discriminants. La sélection d'attributs permet alors de réduire le nombre d'attributs nécessaires à la classification. Cette diminution d'attributs réduit alors le coût et le temps de traitement nécessaire au calcul des attributs.

### 3.3.1.2/ DONNÉES D'ENTRAÎNEMENT

Une phase d'apprentissage est requise et réalisée en mode "hors ligne", basée sur un ensemble de patches annotés visuellement par un expert en fonction de leur texture.

Des exemples des patches utilisés pour l'apprentissage, correspondant aux deux classes sont présentés figure 3.18 où les patches contenant des hautes fréquences sont sur la ligne supérieure tandis que les patches basses fréquences sont présentés sur la ligne du bas.

35 photographies 8k ont été utilisées comme source pour l'extraction d'un total de 2000 patches



avec une distribution de 1000 patches par classe. Ce nouvel ensemble de données est défini de la manière suivante :  $S_2 = \{P_i, y_i\}_{i=1}^{2000}$  avec  $y_i = \{BF, MHF\}$ .



FIGURE 3.18 – Exemples de patches texturés et non texturés : Les patches hautes fréquences (MHF) (1<sup>ère</sup> ligne), patches basses fréquences (BF) (2<sup>ème</sup> ligne)

### 3.3.1.3/ EXTRACTION D'ATTRIBUTS POUR L'ANALYSE DE TEXTURE

Prenant en compte les contraintes d'implantation temps réel, nous avons défini un ensemble d'attributs basés sur des statistiques standards extraites du gradient de Sobel  $S_{GI}$ , qui est un filtre passe-haut.

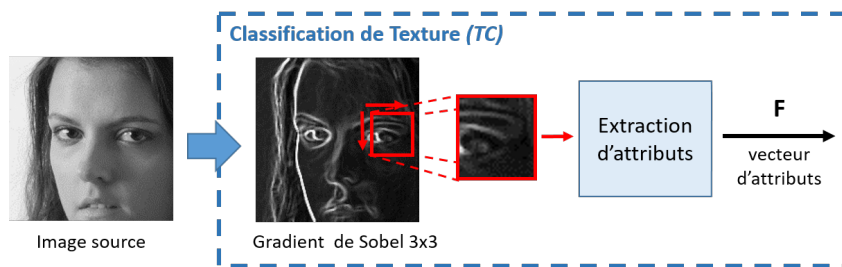


FIGURE 3.19 – Extraction d'attributs à partir du gradient de Sobel

Nous définissons un voisinage de pixel  $N_1$  d'une taille  $n \times n$  centré autour du pixel  $p(i, j)$  qui doit être classé, où  $(i, j)$  sont les coordonnées du pixel. Le choix de la valeur  $n$  peut influencer les résultats, à la fois en terme de complexité de calcul, et de performances de classification. L'influence de sa valeur sera donc étudiée dans la section 3.3.1.4. Dans ce premier voisinage  $N_1$ , deux attributs sont calculés : la moyenne  $\bar{X}$  et la variance  $\sigma^2$  du gradient de Sobel  $S_{GI}$ . Un deuxième voisinage  $N_2$  est défini, d'une taille de fixe de  $3 \times 3$  pixels. Dans ce voisinage  $N_2$ , nous utilisons les neuf valeurs du gradient de Sobel, formant le sous vecteur d'attributs  $\mathbf{G}_2$ , et calculons également la moyenne  $\bar{X}_2$  de  $S_{GI}$ .

Le vecteur d'attributs résultant est le suivant :  $\mathbf{F}(i, j) = \{\bar{X}, \sigma^2, \mathbf{G}_2, \bar{X}_2\}$ .

### 3.3.1.4/ PERFORMANCE DE CLASSIFICATION ET SÉLECTION D'ATTRIBUTS

Dans le but de choisir la meilleure méthode de classification, en terme de compromis entre l'erreur de classification globale et les ressources matérielles nécessaires (complexité, fréquences, utilisation du FPGA). Nous étudions l'influence du paramètre  $n$  (taille du fenêtrage pour le calcul des attributs) sur les performances des SVM et du Boosting, avec  $n = 4, 8, 16, 32, 64$ .

Un outil logiciel créé au laboratoire permet d'évaluer les performances des SVM et du Boosting [76]. Cet outil donne accès aux erreurs globales de classification, d'estimation des ressources FPGA et permet de générer les modèles de classification (SVM et Boosting) dans divers formats ce qui nous permet d'évaluer les différentes méthodes de classification.

Les dimensions initiales du vecteur d'attributs  $\mathbf{F}(i, j)$  est de  $d = 12$ . Le nombre d'attributs de ce vecteur peut être réduit en utilisant la capacité que le Boosting possède pour sélectionner les caractéristiques discriminantes.

Les résultats sont présentés dans la figure 3.20. Lorsque la taille du voisinage  $N_1$  augmente l'erreur globale de classification diminue. Cependant, les différences, en terme d'erreur globale, entre les méthodes de SVM et de Boosting ne sont pas significatives. Le meilleur compromis entre les performances de classification et la complexité est l'implémentation de la méthode de Boosting. De plus, le Boosting a la propriété intrinsèque, puissante, de pouvoir sélectionner automatiquement les attributs nécessaires lors de la phase d'apprentissage.

Enfin, l'algorithme de Boosting a réalisé une sélection des attributs les plus pertinents : seul  $\sigma^2$  a été conservé parmi les 12 attributs. Le vecteur d'attributs final est donc  $\mathbf{F}(i, j) = \{\sigma^2\}$ . Cette solution permet d'implémenter la fonction de décision en utilisant seulement une itération de l'équation 3.16 ou en utilisant une simple mémoire ROM de  $256 \times 1 \text{ bits}$ .

Les ressources matérielles dépendent principalement de l'architecture du filtre utilisé pour l'extraction de l'attribut (voir Chapitre 4 pour une étude détaillée).

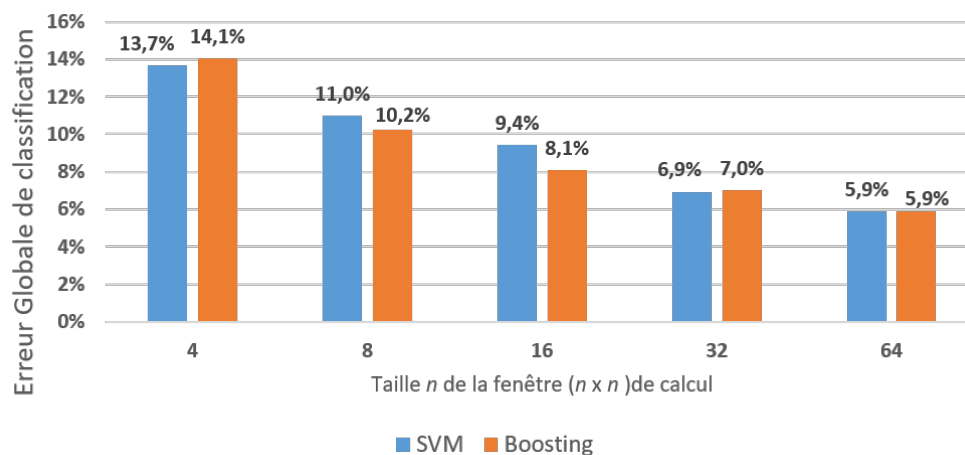


FIGURE 3.20 – Évolution de l'erreur de classification globale pour les méthodes de SVM et de Boosting en fonction de l'évolution de la taille  $n$  de fenêtrage ( $n \times n$ ) utilisée pour le calcul de l'attribut.

Des exemples de classification de texture sont présentés dans la figure 3.21. Les pixels en blanc correspondent aux zones classées comme étant des "hautes-fréquences" où la méthode de Gohshi doit être appliquée et les pixels en noir correspondent aux zones lisses dites "basses-fréquences" où une simple interpolation sera appliquée.

La figure 3.21-a montre un exemple de classification de la texture pour un portrait. Il est possible de voir que les contours des cheveux, des yeux, de la bouche ou du nez sont bien classés comme zones de hautes fréquences. Ensuite les zones "lisses" comme la peau et le fond sont bien en noir. Le même constat peut être fait pour les zones basses fréquences de la figure 3.21-b (Londres) où les zones lisses (ciel et nuages) sont considérées comme des zones non texturées, et tous les détails des régions avec des textures telles que les pierres de la tour seront traités avec la méthode de Gohshi. Dans la figure 3.21-c (Vienne), les détails du mur ressortent bien comme étant texturés.



FIGURE 3.21 – Les Images originales (*gauche*) et les images représentant la classification de texture (*droite*). Et plus précisément de haut en bas : "Portrait" (a), "Londres" (b) et "Vienne" (c)

## 3.3.2/ FUSION

Le principe de cette étape est de mélanger l'image interpolée  $LII$  avec l'image sortante du filtre de Gohshi  $GFI$  en fonction de la classe correspondante à la texture  $y$  (voir équation 3.17).

$$LASSR(i, j) = LII(i, j) \times (1 - y(i, j)) + GFI(i, j) \times y(i, j) \quad (3.17)$$

Lors du développement de notre méthode nous avons pu voir que notre version du traitement de Gohshi mise en place augmente la luminosité de l'image, de ce fait lorsque l'équation 3.17 est réalisée un halo est mis en avant autour des zones classées comme texturées. Pour réduire l'effet de halo il est nécessaire de réaliser une égalisation locale de la luminosité, prenant en compte la luminosité locale moyenne des deux images. La dernière étape de Fusion est réalisée afin de suivre l'équation 3.18. Le principe de fonctionnement de cette fonction de fusion est présenté dans la figure 3.22.

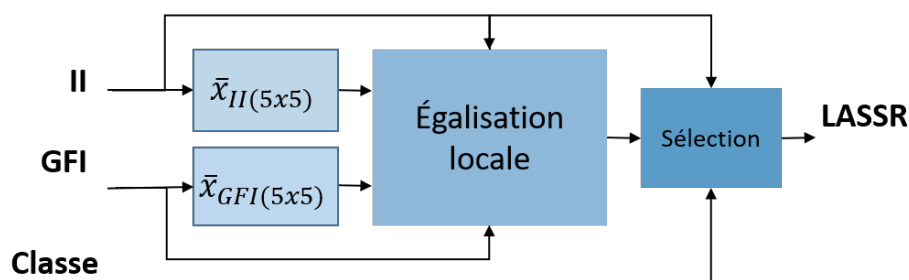


FIGURE 3.22 – Schéma bloc présentant le fonctionnement de la fusion Locale

$$LASSR(i, j) = LII(i, j) \times (1 - y(i, j)) + GFI_{eq}(i, j) \times y(i, j) \quad (3.18)$$

Avec  $(i, j)$  les coordonnées du pixel à traiter,  $LII(i, j)$  est la valeur de la luminosité de l'image interpolée et  $GFI_{eq}(i, j)$  est la version égalisée localement de  $GFI(i, j)$ , calculée avec la moyenne des luminosités sur un voisinage de  $5 \times 5$  pixels.

$$GFI_{eq}(i, j) = GFI(i, j) + (\overline{X_{LII}(i, j)} - \overline{X_{GFI}(i, j)}) \quad (3.19)$$

où

$$\overline{X_{LII}(i, j)} = \frac{1}{25} \sum_{k=i-2}^{2+i} \sum_{l=j-2}^{2+j} LII(k, l) \quad (3.20)$$

et

$$\overline{X_{GFI}(i, j)} = \frac{1}{25} \sum_{k=i-2}^{2+i} \sum_{l=j-2}^{2+j} GFI(k, l) \quad (3.21)$$

Des exemples de fusion avec et sans égalisation locale de luminosité sont présentés dans la figure 3.23. Il est possible de voir dans la figure 3.23-a que GFI élève énormément la luminosité de l'image, et fait perdre des informations dans les zones lisses (ciel). Contrairement à l'image interpolée LII (figure 3.23-b) où il est possible d'avoir la couleur du ciel. Un halo apparaît (figure 3.23-c) lorsque nous fusionnons directement (a) et (b) en suivant l'équation 3.17, et ceci dans les zones classées comme étant texturées. Ces artefacts sont supprimés lors de l'égalisation locale de la luminosité, en suivant l'équation 3.18 (voir figure 3.23-d).



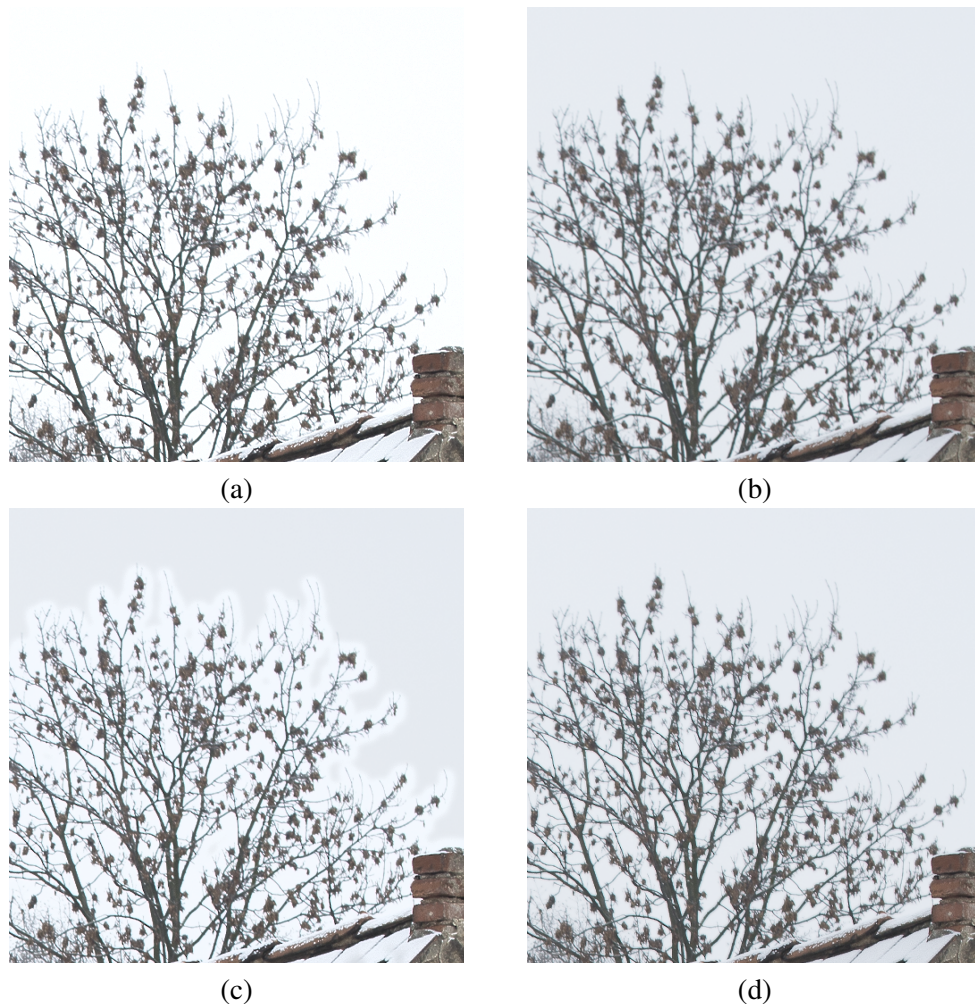


FIGURE 3.23 – Exemple de fusion d’une image issue de la méthode de Gohshi (a) et de l’image interpolée (b) avec (d) et sans (c) égalisation locale de la luminance

### 3.4/ ÉVALUATION DE LA MÉTHODE LASSR

#### 3.4.1/ CRITIQUE VISUELLE DE LA MÉTHODE LASSR

La méthode LASSR a été implémentée et évaluée sous Matlab. Trois exemples sont présentés figure 3.24 : un portrait et deux paysages (urbain et forestier). Les caractéristiques, comme la luminance et la couleur, des images issues de la méthode LASSR sont similaires aux caractéristiques de l’image originale (vérité de terrain). De plus, il n’y a pas d’artefacts locaux dus à la luminance entre les zones texturées et non-texturées (par exemple dans les cheveux et les détails de la peau de la figure 3.24-a (portrait)). Ces premiers résultats valident la fusion avec égalisation locale de la luminance des images GFI provenant de la méthode de Gohshi avant la fusion finale.

Afin d’illustrer la relative robustesse au bruit des méthodes employées, nous avons représenté sur la figure 3.25 les résultats des traitements obtenus sur un extrait d’une image 8k présentant un bruit élevé, car acquise avec une sensibilité de capteur réglée sur 3200 ISO. Il est possible de voir que même en présence de bruit, il n’apparaît pas d’artefacts particuliers. Ces traitements fonctionnent normalement avec ce type de bruit.

Une évaluation plus approfondie de la méthode est présentée dans les sections 3.4.2 et 3.4.3 avec l'utilisation de métriques sélectionnées dans l'étude sur les métriques, section 3.1.4, métriques couramment utilisées dans le domaine de l'amélioration de la qualité. Ces résultats sont complétés avec la mise en place d'une seconde évaluation psychovisuelle  $PVA_2$ .

### 3.4.2/ RÉSULTATS DES MÉTRIQUES CONVENTIONNELLES

À la suite de l'étude sur les métriques nous avons sélectionné les métriques PSI, CPBD, EME, NIQE et VIF comme étant les métriques les plus pertinentes pour l'évaluation des techniques d'amélioration de la résolution et de la qualité des images. Cependant, il a été précisé dans la section 3.1.4 que les métriques sans référence étaient pour la majorité des cas basées sur la somme de caractéristiques influençant la perception humaine. Ces caractéristiques sont le contraste et la netteté des contours. Le tableau 3.4 présente l'évaluation des trois traitements en utilisant notre sélection de métriques. Il est possible de voir que le traitement le mieux classé correspond à la méthode de Gohshi.

	PSI	CPBD	EME	NIQE	VIF
LII	0.251 (3)	0.312 (3)	5.660 (3)	4.909 (3)	0.819 (1)
GFI	0.309 (1)	0.547 (1)	7.903 (1)	4.519 (2)	0.811 (2)
LASSR	0.304 (2)	0.518 (2)	7.213 (2)	3.927 (1)	0.809 (3)

TABLE 3.4 – Résultats de la méthode *LASSR* comparés au traitement de Gohshi *GFI* et d'une interpolation *LII* pour des métriques avec et sans-référence sélectionnées dans la section 3.1.4

Les métriques sans référence classent la méthode de Gohshi en première position pour la majorité des cas et la méthode *LASSR* en seconde position. L'interpolation *LII* arrive pour les métriques sans référence en dernière position.

Au contraire la métrique conservée pleine-référence classe l'interpolation en première position.

Il est cependant possible de voir dans le tableau que les résultats de ces métriques pour *GFI* et *LASSR* sont toujours assez proches, et un plus grand écart de valeur est présent avec le traitement *LII*.

Avec ces métriques il serait logique de considérer alors la méthode de Gohshi *GFI* comme étant meilleure que les méthodes *LASSR* et *LII*. Cependant il est préférable de réaliser une évaluation psychovisuelle *PVA* au regard de l'observation des artefacts présentés dans la figure 3.3.

### 3.4.3/ MISE EN PLACE DE L'ÉVALUATION PSYCHOVISUELLE $PVA_2$

L'évaluation psychovisuelle *PVA*, *Psycho-Visual Assessment*, que nous avons mis en place a pour objectif de comparer trois traitements du domaine de l'amélioration de la qualité et de la résolution d'image. Les traitements à comparer sont notre traitement *LASSR* et deux traitements présents dans l'état de l'art : la méthode de Gohshi *GFI* et une interpolation de Lanczos *LII*. Cette évaluation permet aussi de confirmer et comparer les résultats obtenus avec des métriques habituelles de l'état de l'art. Pour la réalisation de  $PVA_2$  nous avons modifié notre outil logiciel afin d'utiliser un nouveau protocole de test avec les caractéristiques suivantes :





FIGURE 3.24 – Comparaison visuelle pour trois patches d’une dimension de  $1024 \times 1024$  pixels. Avec (a) extraits d’images d’une résolution de 8k ; (b) résultats d’une interpolation de Lanczos *LII*, (c) résultats du traitement de Gohshi *GFI* et (d) résultats avec la méthode proposée *LASSR*

**Le Panel :**

Le panel est composé de 30 élèves de différentes classes d’une école d’ingénieur spécialisée en



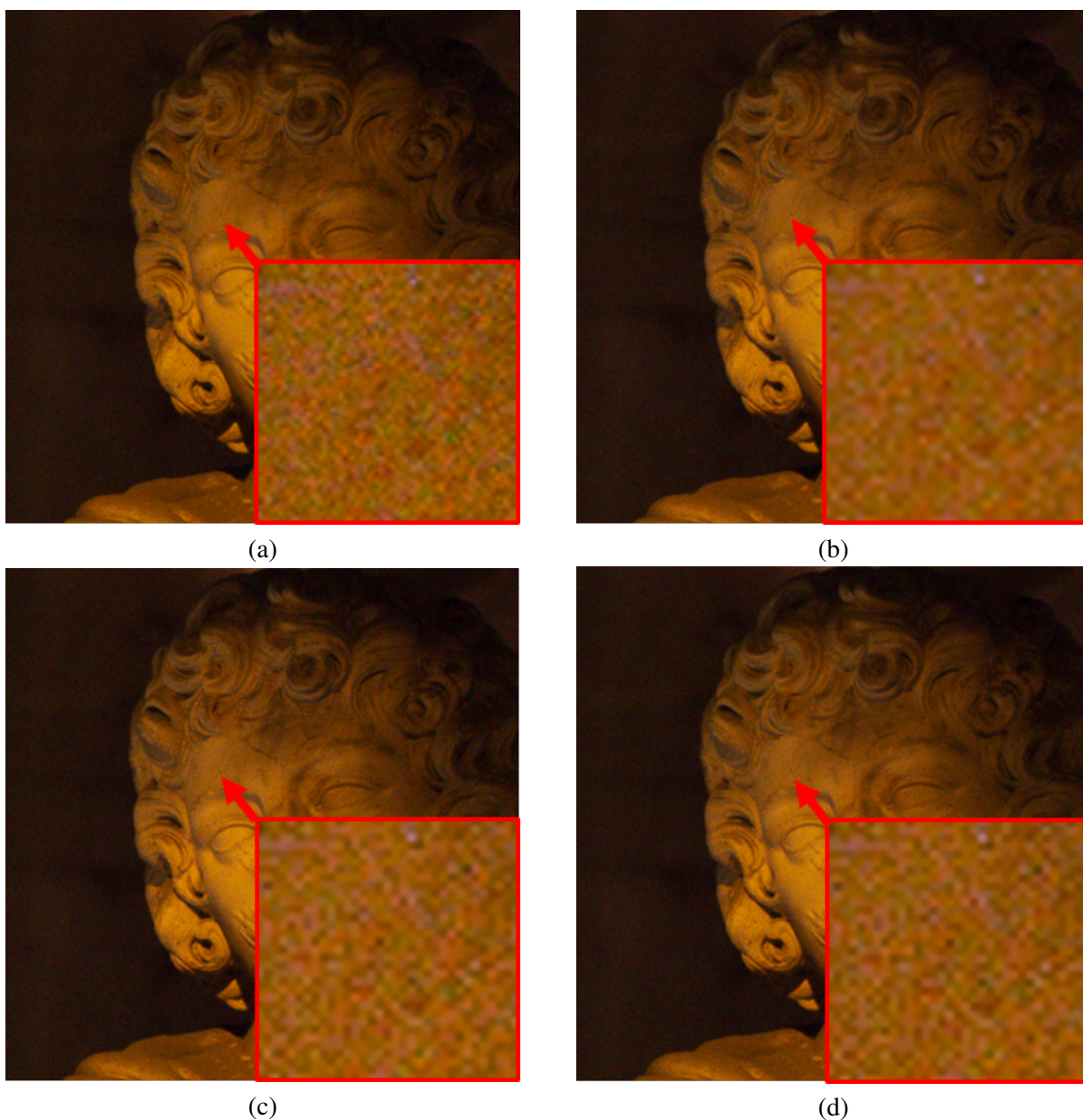


FIGURE 3.25 – Comparaison visuelle d'un extrait  $1024 \times 1024$  pixels d'une image 8k bruitée (3200 iso). Avec (a) un extrait de l'image source, (b) résultat d'une interpolation de Lanczos *LII*, (c) résultat du traitement de Gohshi *GFI* et (d) résultat avec la méthode proposée *LASSR*. Les patches encadrés en rouge correspondent à des agrandissements de taille  $64 \times 64$  pixels.

électronique et en informatique, cependant l'évaluation de seulement 26 étudiants a été conservée. Les résultats de 4 panélistes ont été enlevés pour deux raisons : (1) le test avait été réalisé dans un temps trop court, (2) les résultats montraient que le panéliste sélectionnait toujours le même emplacement dans la fenêtre dans l'outil. L'âge du panel  $P_{PVA_2}$  est compris entre 19 et 25 ans.

#### Les stimuli :

Chaque membre du panel doit évaluer un set de 150 images d'une résolution de  $1024 \times 1024$ . Pour chaque évaluation quatre images sont affichées dans la fenêtre de l'outil *Choix.exe*. L'image

de gauche représente l'image source (vérité de terrain), les trois autres images correspondent aux trois traitements. Pour chaque test les images des traitements sont affichées de manière aléatoire afin de ne pas toujours afficher les mêmes traitements aux mêmes endroits.

Les images sont issues de la base d'image  $S_3$ , quelques exemples sont présentés dans la figure 3.26. Ces images ont été extraites d'images 8k capturées à l'aide d'un appareil de photographie professionnel Nikon D800. Le set d'images  $S_3$  se compose d'une grande variété de type de scènes, du paysage urbain, naturel au portrait humain, avec plusieurs niveaux de détail.

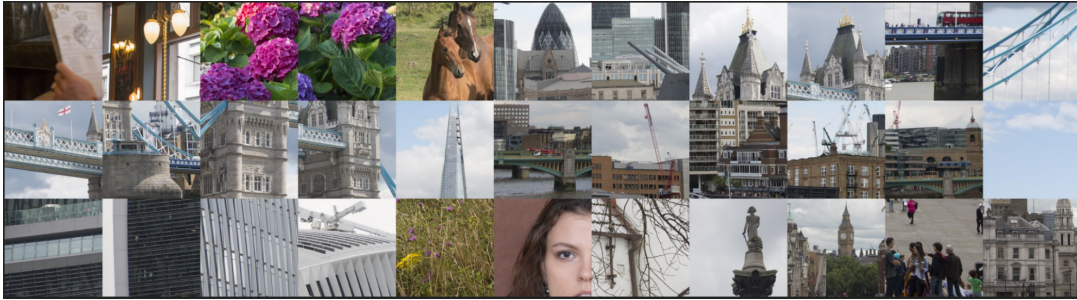


FIGURE 3.26 – Exemple d'images provenant de  $S_3$  utilisées durant le test psychovisuel  $PVA_2$ . Les images de taille  $1024 \times 1024$  sont affichées 1 pixel pour un

#### La procédure :

Chaque membre du panel  $P_{PVA_2}$  évalue un set de trois images provenant des traitements GFI, LII et LASSR. L'affichage de ces trois images dans la fenêtre de l'outil est réalisé de manière aléatoire. De plus aucune information sur le type de traitement n'est donnée au panel  $P_{PVA_2}$ . L'évaluation est alors considérée comme aveugle. Cependant les règles à suivre pour l'évaluation peuvent permettre aux testeurs de déduire que les traitements à comparer sont issus du domaine lié à la qualité de l'image.

Parmi les trois images affichées le panéliste a la tâche de les classer, selon ses propres préférences, mais en prenant en compte les règles suivantes :

- Règle N°1 : L'image doit avoir le niveau de détails le plus élevé et doit respecter la règle N°2,
- Règle N°2 : La couleur et la luminosité de l'image choisie doit être proche de l'image de référence.

En suivant les règles précédemment citées le panéliste classe alors les trois traitements du meilleur au moins bon. Ce classement se fait par un "click" sur l'image choisie. Pour classer les trois images il suffit alors de cliquer sur les deux meilleures, la dernière étant déduite automatiquement.

L'évaluation se déroule sur un même ordinateur qui dispose d'une carte graphique permettant d'afficher des images en résolution 4k. Cela permet de pouvoir afficher les quatre images en respectant le rapport un pixel de l'image pour un pixel de l'écran.

Chaque panéliste est placé à une distance d'environ 60 cm de l'écran dans une pièce éclairée.

La durée de l'évaluation n'est pas imposée, le panéliste prend le temps qui lui semble nécessaire pour évaluer les trois images.

Le choix est considéré comme "forcé" car il n'est pas possible au panéliste de choisir deux fois la même image, de plus le choix est obligatoire pour passer à l'évaluation de l'image suivante.

De plus, il est important de préciser qu'il s'agit d'une évaluation psychovisuelle réalisée à titre volontaire et gratuite. En effet, le panéliste n'est en aucun cas forcé de participer à cette étude, et il ne reçoit aucune contribution à la suite de sa participation.

### Les résultats :

Au total, 30 étudiants ont réalisé l'évaluation  $PVA_2$ . En moyenne 22 minutes étaient nécessaires pour réaliser l'évaluation des 150 images  $1024 \times 1024$  de  $S_3$ .

Cependant les résultats de 4 étudiants ont été retirés de cette évaluation car il a été possible de voir pour certains de ces panélistes que l'évaluation n'avait pas été faite sérieusement : la durée était trop courte et la sélection du classement se faisait toujours au même emplacement dans la fenêtre de l'application de test.

Le tableau 3.5 présente les résultats obtenus avec l'évaluation  $PVA_2$ . Il est possible de voir dans la première ligne de résultats que la méthode LASSR a été classée comme étant la meilleure des trois par 40% du panel. Selon le panel la méthode qui arrive en seconde position est la méthode de Gohshi GFI avec un total de 32% du panel.

La seconde ligne de résultats montre que la méthode LASSR arrive en second choix pour 44% du panel lorsqu'elle n'est pas choisie en premier. Les deux autres méthodes ne sont pas significativement différentes avec des résultats proches : 29% pour la méthode de Gohshi, et 27% pour l'interpolation.

Enfin la dernière ligne du tableau montre que la méthode LASSR est considérée comme étant la pire des méthodes pour seulement 16% du panel alors que les deux autres méthodes se situent autour de 40% (39% et 45%).

Rang	GFI	LII	LASSR
1 <sup>er</sup>	32%	28%	<b>40%</b>
2 <sup>ème</sup>	29%	27%	<b>44%</b>
3 <sup>ème</sup>	39%	<b>45%</b>	16%

TABLE 3.5 – Résultat de l'évaluation  $PVA_2$  pour les méthodes GFI, II and LASSR

Les résultats de  $PVA_2$  permettent de positionner notre méthode LASSR devant la méthode de Gohshi et une simple interpolation.

### 3.4.4/ CONCLUSION

D'après les sections 3.4.2 et 3.4.3 il est possible de voir que les résultats entre les métriques habituelles ne vont pas exactement dans le sens des métriques purement subjectives. En effet, même si les métriques sélectionnées dans la section 3.4.2 sont basées sur la perception humaine nous pouvons voir qu'il y a des limites.

Par exemple, il est possible de voir dans les images issues du traitement de Gohshi GFI des dépassements, où nous avons l'impression que le pixel sature. Les métriques basées sur le contraste donnent une note très bonne car la valeur du contraste est localement très élevée.

De plus, nous pouvons voir que les résultats de l'évaluation psychovisuelle  $PVA_2$  sont vraiment liés aux goûts de chaque panéliste. En effet tous les classements établis par le panel sont rarement similaires : ceci est dû à la variabilité inter-individuelle. Et la sensibilité du panéliste aux variations de contraste et au piqué de l'image n'est pas la même pour tous. Il serait intéressant de voir, pour

chacun des panélistes, si son classement est toujours le même pour les 150 images présentées. La répétabilité du choix est un critère important pour valider une étude. Augmenter le nombre de panéliste peut être une solution mais cela n'enlèvera pas la variabilité, cependant il serait tout aussi intéressant de créer un panel composé seulement d'experts de l'image.

D'après le classement des métriques conventionnelles nous pouvons établir que selon celles-ci le classement serait le suivant  $GFI > LASSR > LII$ . Cependant ce classement n'est pas confirmé avec l'évaluation  $PVA_2$  qui donne  $LASSR > GFI > LII$ . En effet, le panel donne quand même de bons résultats pour la méthode de Gohshi GFI mais de meilleurs résultats pour la méthode LASSR. Le point commun entre les deux types d'évaluation est le fait que le classement d'interpolation LII arrive toujours en dernière position pour les métriques sans-référence. Seule la métrique NIQE suit exactement le classement établi par le panel  $P_{PVA_2}$ .

Nous pouvons conclure que notre méthode apporte de meilleurs résultats que les méthodes de Gohshi GFI et d'interpolation LII pour l'amélioration de la résolution des images.

De plus nous pouvons aussi voir qu'il est assez difficile de bien choisir une métrique pour l'évaluation d'un traitement de l'image. La métrique la plus réaliste est celle qui est directement réalisée par l'homme. Cependant ces évaluations subjectives sont très difficiles à mettre en place, notamment lorsqu'elles ne font pas l'objet d'un dédommagement.



## ACCÉLÉRATEUR MATÉRIEL

La contribution de notre équipe de recherche pour le projet européen EXIST-H2020 est la création d'un système hybride permettant à la fois de corriger les images issues d'un capteur haute résolution 4k par déconvolution et ensuite de réaliser un traitement d'augmentation de la résolution tout en conservant un bon niveau de détail.

L'augmentation de la résolution spatiale, présentée section 4.1, est réalisée en implémentant, de façon matérielle, la méthode LASSR. Cette implémentation matérielle sur cible FPGA est réalisée pour fonctionner en temps réel sur des volumes de données très importants. Il a été montré dans le chapitre 3 que la méthode LASSR est efficace au niveau de la perception visuelle. Tout d'abord, la méthodologie complète relative à ce développement est présentée dans la section 4.1.1. Ensuite, la section 4.1.2 met en avant le développement matériel avec la création d'une *boite à outils* regroupant des composants génériques. Enfin, les analyses des ressources matérielles et des performances de l'architecture proposée sont présentées dans la section 4.1.3.

Ensuite, la section 4.2 présente l'implantation d'une méthode de déconvolution pour la correction de l'effet de flou causé par le système optique. Cette déconvolution se base sur les spécifications techniques fournies par le constructeur. J'ai pu participer à l'élaboration de cette partie, cependant le développement et l'intégration de la méthode de déconvolution ont été effectués par Ali Douiyek, ingénieur de recherche. La déconvolution a été mise en place de façon logicielle avec une intégration GPU, ce qui permet de traiter rapidement de grandes images. Tout d'abord, la méthode utilisée pour traiter des images de grandes tailles associées à un grand nombre de PSFs est présentée dans la section 4.2.1. Ensuite, la section 4.2.2 met en avant les différents résultats obtenus. Enfin, la section 4.2.3 fait un état des lieux du démonstrateur fourni pour le projet européen et ouvre les perspectives sur les évolutions possibles concernant l'étude sur la déconvolution.

### Sommaire

<b>4.1</b>	<b>Mise en œuvre d'un prototype intégrant la méthode LASSR</b>	<b>94</b>
4.1.1	Méthodologie de développement et validation	94
4.1.2	Développement matériel	99
4.1.3	Résultats d'implantation et discussion	110
<b>4.2</b>	<b>Déconvolution temps réel - GPU</b>	<b>118</b>
4.2.1	Méthode	119
4.2.2	Résultats	121
4.2.3	Discussion	125



#### 4.1/ MISE EN ŒUVRE D'UN PROTOTYPE INTÉGRANT LA MÉTHODE LASSR

L'architecture du prototype intégrant la méthode LASSR est destinée à pouvoir s'adapter à différents systèmes. Sa structure est suffisamment générique pour pouvoir être reliée à un grand nombre d'interfaces de communication. Nous ne détaillerons pas dans le manuscrit les interfaces de communication mises en œuvre au sein du prototype développé. La structure vise à exploiter pleinement le parallélisme intrinsèque de l'algorithme ainsi que la régularité de certaines tâches mettant en particulier en œuvre un grand nombre de voisinages de pixels.

Nous présentons dans la section 4.1.1 la méthodologie de développement que nous avons utilisée lors de nos travaux de mise en place du traitement LASSR. Cette section présente les différentes étapes nécessaires à la création d'un core IP intégrable dans une cible matérielle.

L'architecture mise en place ainsi que la description des composants les plus importants pour l'implantation de notre méthode sont présentés dans la section 4.1.2. De plus, l'approche de développement proposée consiste à concevoir des composants génériques pouvant être réutilisés de multiples fois dans cette architecture ou dans d'autres systèmes. Une *boîte à outils* regroupant ces composants génériques a été développée. Elle intègre notamment des filtres 2D dont la taille et les coefficients sont paramétrables par l'utilisateur. Cette section décrit l'étude permettant de définir la taille de la sous-fenêtre de calcul des attributs utilisés dans le bloc d'analyse de texture. Cette étude permet de trouver un compromis entre l'erreur globale de classification et les ressources nécessaires à son calcul.

Enfin, les analyses des ressources matérielles et des performances de l'architecture proposée sont présentées dans la section 4.1.3. Ces résultats permettent de mettre en avant les avantages de notre méthode, et les perspectives permettant d'envisager l'amélioration de l'implantation matérielle proposée.

##### 4.1.1/ MÉTHODOLOGIE DE DÉVELOPPEMENT ET VALIDATION

Cette section a pour but d'expliciter notre méthodologie de développement, du prototypage logiciel à l'implantation matérielle. La finalité de notre projet est la création d'un Core IP intégrable dans une chaîne complète d'acquisition et de traitement de l'image dans une cible FPGA. Pour limiter le temps de développement, généralement important pour ce type de cibles matérielles, une approche de prototypage rapide sera intégrée à notre démarche.

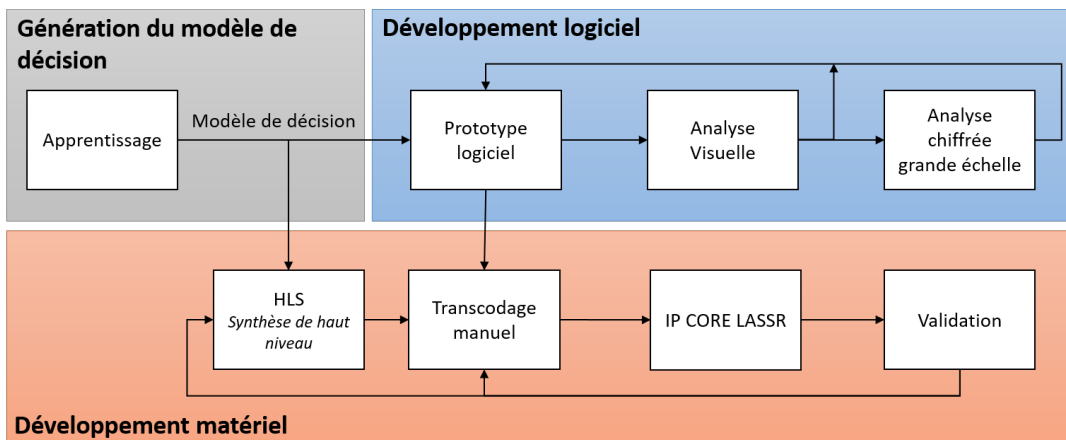


FIGURE 4.1 – Méthodologie de développement de la méthode LASSR



La figure 4.1 illustre les différentes parties et étapes nécessaires à la création du composant matériel.

1. La génération du modèle de décision (section 4.1.1.1) issu de l'apprentissage, nécessaire à la partie en charge de l'analyse de texture de la méthode LASSR,
2. Le développement logiciel (section 4.1.1.2) permettant de réaliser un prototype logiciel de la méthode pour fixer l'algorithme,
3. Le développement matériel (section 4.1.1.3) permettant la mise en place de l'algorithme LASSR dans une cible matérielle de type FPGA.

La génération du modèle de décision et le prototypage logiciel sont réalisés conjointement afin de valider l'algorithme de la méthode LASSR. Le développement matériel est naturellement réalisé à l'issue de la validation logicielle.

#### 4.1.1.1/ MÉTHODOLOGIE DE DÉVELOPPEMENT DU MODÈLE DE DÉCISION

La génération du modèle de décision s'appuie sur une longue expérience acquise au sein du laboratoire au niveau de classifieurs de type SVM ou une approche basée sur le boosting. Ce type de classifieurs a notamment été retenu en raison de travaux antérieurs qui ont permis de réaliser des implantations temps réel performantes sur des cibles de type FPGA [76, 102]. Un outil logiciel développé au sein du laboratoire a été utilisé pour tester différents classifieurs, fournir une estimation de l'erreur globale de classification, réaliser une sélection d'attributs et générer automatiquement le modèle de décision.

Le modèle de décision est généré sous différentes formes possibles en fonction du choix du classifieur, et du nombre d'attributs. On peut noter que la structuration du modèle est réalisée dans l'objectif d'une implantation matérielle. Le modèle peut ainsi être généré sous un format permettant le stockage dans une mémoire ROM (directement compatible avec les blocs mémoires embarqués au sein d'un FPGA) mais également sous la forme d'une structure dans un fichier C++ pour la validation logicielle.

La méthodologie de génération du modèle de décision est présentée dans la figure 4.5, les phases sont présentées de la création de la base d'apprentissage à la génération du modèle. Il est nécessaire de créer une base d'images d'apprentissage composée de  $N$  classes de patches extraits manuellement d'images haute résolution. Comme il s'agit d'un apprentissage supervisé, les *étiquettes*, correspondant ici aux classes *texturée* et *non-texturée*, sont définies à la main par un expert humain.

Ensuite, les attributs sont extraits de ces patches et les vecteurs d'attributs sont ensuite enregistrés dans un format interprétable par le logiciel de décision.

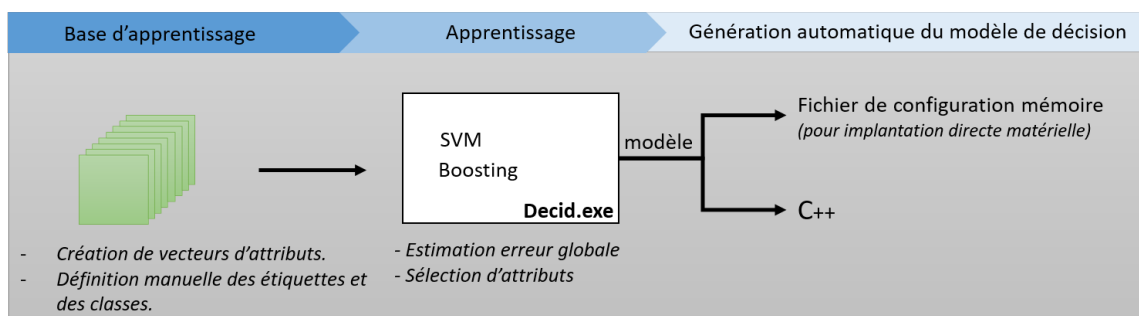


FIGURE 4.2 – Méthodologie de génération du modèle de décision

Le classifieur est choisi en fonction de l'erreur globale de classification, des ressources matérielles nécessaires au calcul des attributs et des ressources matérielles nécessaires à l'intégration du modèle de décision dans la cible matérielle.

Enfin, le modèle de décision peut être utilisé par le prototype logiciel permettant de valider l'algorithme LASSR. Ce modèle est utilisé par la suite pour l'implantation matérielle.

#### 4.1.1.2/ MÉTHODOLOGIE DE DÉVELOPPEMENT DU PROTOTYPE LOGICIEL

La méthodologie de développement logiciel est présentée dans la figure 4.3. Ce développement permet de réaliser et d'évaluer un prototype logiciel de manière simple de la méthode LASSR.

Ce développement se réalise en suivant plusieurs étapes :

- Création du prototype logiciel réalisant notre fonction,
- Analyse visuelle des résultats nous permettant d'adapter et d'optimiser l'algorithme,
- Validation expérimentale de la méthode en utilisant des analyses chiffrées réalisées sur un nombre important d'échantillons de test.

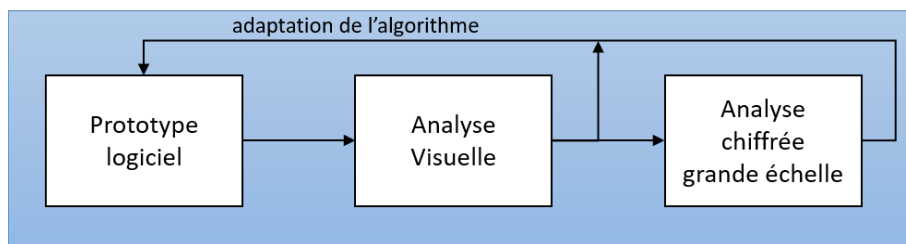


FIGURE 4.3 – Méthodologie de développement logiciel de la méthode LASSR

Pour rappel, la description de cette méthode et l'analyse chiffrée sont présentées respectivement dans les sections 3.3 et section 3.4.

Ce prototype est réalisé sous Matlab, cependant il aurait été possible de le réaliser avec d'autres langages comme le C++ ou Python. Nous avons décidé d'utiliser cette plateforme car elle offre la possibilité de faire des développements hybrides entre plusieurs langages de programmation différents. Par exemple, il nous était possible, via l'utilisation de l'outil Simulink, d'associer des éléments décrits avec Matlab, des éléments en C++ et d'autres éléments décrits en langage à haut niveau d'abstraction matériel (VHDL).

#### 4.1.1.3/ MÉTHODOLOGIE DE DÉVELOPPEMENT MATÉRIEL

Le but du développement matériel est de créer un core IP intégrable dans une caméra computationnelle. Dans ce mémoire, l'interface de communication complète n'est pas décrite. Le core IP doit fonctionner avec des signaux de synchronisation provenant d'un capteur. Ce capteur n'étant pas défini dans notre projet, nous utilisons des synchronisations usuelles de capteur CMOS comme un exemple de synchronisation illustré dans la figure 4.4 où la valeur d'un pixel est validée à l'aide d'un signal d'horloge et d'un signal de validation comme "*line\_valid*".

La validation sur cible FPGA est précédée de deux étapes de transcodage et d'une validation du composant décrit en VHDL.

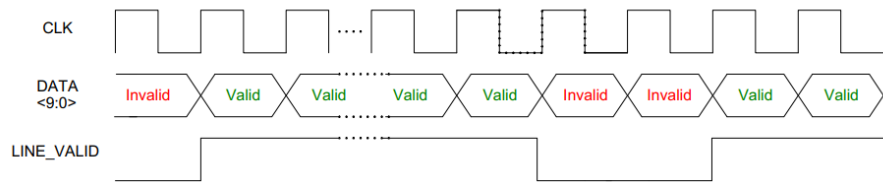


FIGURE 4.4 – Exemple de synchronisation d'un capteur CMOS

#### - Le transcodage du prototype logiciel :

L'algorithme de la méthode LASSR, validé lors du développement logiciel, est transcodé en langage de haut niveau VHDL. La description matérielle est réalisée manuellement ce qui permet de décrire très précisément le fonctionnement de tous les composants nécessaires à l'implantation de la méthode. Le développement est réalisé sous VIVADO. Cet outil de Xilinx permet de fournir l'estimation des ressources matérielles utilisées sur une cible FPGA donnée et la fréquence de fonctionnement maximum possible.

Les composants sont décrits de façon générique, ce qui permet de les utiliser avec plusieurs configurations dans notre projet, mais aussi de pouvoir les utiliser dans d'autres projets. Cette description générique permet ensuite de gagner du temps dans le développement matériel.

#### - Le transcodage du modèle de décision :

Le modèle de décision est généré par l'outil de décision en deux formats (fichier de configuration mémoire ROM ou fichier C++) en fonction du nombre d'attributs et du classifieur choisi. Lorsque le modèle est généré comme étant le fichier de configuration d'une mémoire ROM, il suffit alors de l'intégrer directement dans le core IP.

Lorsque le nombre d'attributs est important et que le classifieur choisi n'est pas compatible avec une intégration directe dans une mémoire ROM, le modèle est généré sous forme de code C++ et il est alors nécessaire de transcoder ces fichiers en langage VHDL. Il serait possible de les transcoder manuellement, cependant, le fonctionnement de ces codes a été validé lors de tests logiciels, le transcodage automatique n'apporte aucune dégradation en terme de fonctionnement. La figure 4.5 présente les différentes étapes nécessaires au transcodage du code C++ en VHDL.

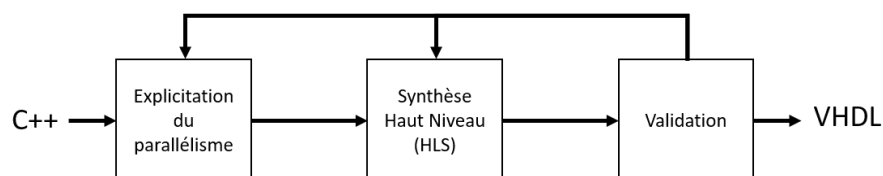


FIGURE 4.5 – Transcodage C++ vers VHDL depuis Vivado HLS

À ce niveau, le transcodage est réalisé de manière semi-automatique à l'aide du logiciel de synthèse haut niveau VIVADO HLS. Cet outil permet de transcoder directement des programmes écrits originellement en C++ vers des langages de plus haut niveau comme VHDL, Verilog, et SystemC. Il est cependant nécessaire d'explicitier manuellement le parallélisme du programme, à partir de directives appliquées sur des portions de code. Par exemple, il est ainsi possible de réaliser le

déroulage partielle ou total de boucles. Les boucles sont déroulées et les différentes itérations sont parallélisées.

Le modèle varie en fonction de la base d'apprentissage. Si les dimensions et le nombre d'attributs restent constants seul un fichier d'en-tête (*header .h*) est modifié parmi tous les fichiers générés en C++. Ce fichier d'en-tête contient tous les coefficients et seuils nécessaires à la fonction de décision. Comme le reste du code C++ reste inchangé, l'explicitation du parallélisme n'est réalisée qu'une seule fois.

### - La validation

Des étapes de validation sont réalisées durant le processus de développement matériel pour vérifier le bon fonctionnement de chaque composant. Ces vérifications de fonctionnement sont réalisées à l'aide d'outils de simulation tel-que Modelsim, ou le logiciel de simulation directement intégré à VIVADO. Ces outils permettent de générer des chronogrammes qui aident à la validation des diverses synchronisations, la transmission et la gestion des signaux à travers les différents composants.

Dans le but d'améliorer et de simplifier les validations nous avons décidé d'utiliser l'outil de prototypage rapide proposé par Matlab : Simulink. En effet nous désirons pouvoir valider et tester notre composant LASSR avec différentes images en entrée, et pouvoir vérifier visuellement si le traitement ne présente pas d'artefact, comme un éventuel décalage entre les images fusionnées. Simulink donne la possibilité de créer des interfaces entre l'espace de travail Matlab, et l'environnement de simulation Modelsim, il est alors possible de réaliser une co-simulation entre une modélisation à haut niveau d'abstraction et une description bas niveau optimisée pour une implantation matérielle.

La figure 4.6 présente l'intégration des différents outils permettant de réaliser la co-simulation, faisant communiquer des données présentes dans Matlab avec l'outil de simulation Modelsim.

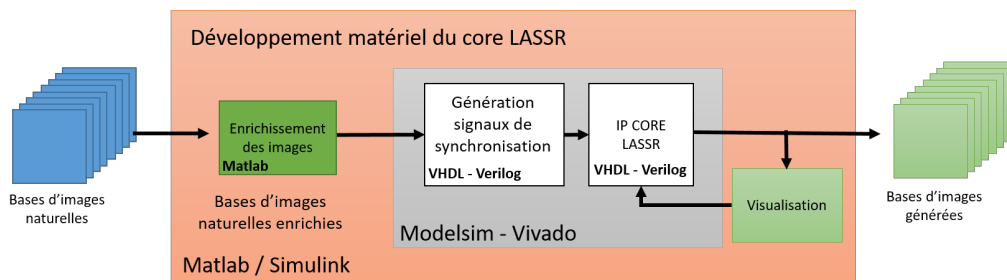


FIGURE 4.6 – Méthode de test et de validation de l'implantation matérielle

Cependant, Simulink ne nous permet pas de transférer les images au core VHDL comme si elles provenaient d'un capteur. En effet, le logiciel ne permet pas de créer et gérer des synchronisations d'envoi. Afin de générer une synchronisation semblable aux synchronisations usuelles des capteurs CMOS (validation ligne, validation image, temps *inter-image*, temps *inter-ligne*) les images ont été enrichies sous Matlab en insérant des valeurs codées spécifiques à chaque début d'images, et à chaque début de lignes. Ces images envoyées pixel par pixel avec une horloge prédéfinie dans Simulink, sont décodées dans un premier composant présent dans le projet VHDL permettant de générer les signaux de synchronisation pour les lignes et les images (*Syncho\_line* et *Syncho\_frame*).

Avec le même principe que la co-simulation, Simulink permet de réaliser du co-design *FPGA in the loop*. Le co-design permet d'envoyer et de recevoir des signaux de l'espace de travail de Matlab

à la carte FPGA. Cette autre méthode donne alors la possibilité de tester directement le composant sur le FPGA.

#### 4.1.2/ DÉVELOPPEMENT MATÉRIEL

Suite à la validation de notre méthode de façon logicielle, nous avons mis en place l'architecture globale de la méthode LASSR proposée en section 3.3. La figure 4.7 présente de façon globale l'architecture que nous avons pu décrire pour la mise en place du traitement. Cette architecture est décomposée en plusieurs blocs réalisant des tâches suivantes : l'interpolation de Lanczos, le filtre de Gohshi (GF), la classification de texture (TC) et la fusion locale (LM). Ces différents blocs sont décrits dans les sous-sections suivantes.

Tous ces blocs, et les composants qui composent ces blocs, sont décrits en VHDL ou en Verilog, de façon manuelle afin de pouvoir gérer au mieux le parallélisme des données et la gestion du grand volume de données.

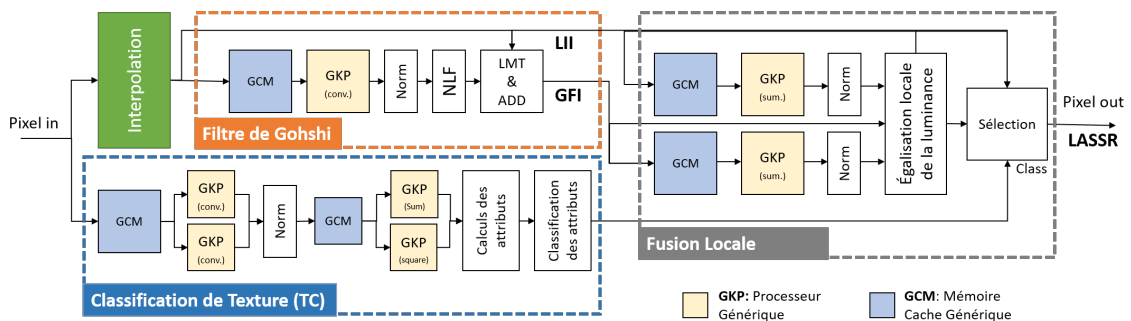


FIGURE 4.7 – Schéma bloc présentant l'architecture globale de la méthode LASSR avec une division en sous-parties des différents blocs qui composent cette méthode : interpolation, Filtre de Gohshi (GF), Classification de texture (TC) et fusion locale.

Dans la majorité des blocs qui composent l'architecture de la méthode LASSR (figure 4.7), il est nécessaire de mettre en place des filtres 2D, dont les tailles et les coefficients peuvent varier. Pour cette raison, nous avons développé un filtre générique afin de nous permettre de mettre en oeuvre plus facilement les différents filtres dont nous avons besoin pour l'architecture de la méthode LASSR. Ce filtre générique, présenté dans la section 4.1.2.1 est l'association de deux blocs distincts, un voisinage et un processeur générique, qui ont été développés pour fonctionner conjointement.

Afin de garantir une haute fréquence de traitement et de limiter les accès mémoire, l'architecture proposée tente d'augmenter le niveau de parallélisme des différentes tâches nécessaires à la réalisation du traitement LASSR.

Cette représentation met en avant les différents niveaux de parallélisme que nous avons choisis de mettre en place dans cette description matérielle.

Une taxonomie des différentes solutions de parallélisme a été présentée en 1966 par Michael Flynn [8]. L'auteur sépare en quatre catégories les différents types de parallélisme :

- SISD, unique flux d'instructions, unique flux de données, ce qui entraîne aucun parallélisme.
- SIMD, unique flux d'instructions, multiple flux de données, ce qui permet un parallélisme de données.
- MISD, multiple flux d'instructions, unique flux de données, ce qui permet un parallélisme

de tâches.

- MIMD, multiple flux d'instructions, multiple flux de données qui combine les deux précédent en permettant un parallélisme de tâches et de données.

Dans notre cas il est possible de voir qu'une seule donnée est traitée à deux niveaux différents. La chaîne de traitement est décomposée en deux branches : la première branche correspond à l'interpolation du pixel d'entrée, suivi par le traitement de Gohshi et la fusion locale. La seconde branche correspond alors à la classification de la texture locale. L'information de la classe correspondante est alors communiquée au bloc de fusion locale. Le parallélisme de cette architecture est donc MISD. Dans chaque chemin, et dans chaque bloc, le traitement est réalisé en pipeline. Le pipeline correspond au fonctionnement d'une chaîne de montage industrielle : les données successives sont traitées à la suite, et chaque tâche est appliquée sur les données successives. Le principe du pipeline est important, car il n'est pas nécessaire que la chaîne de traitement soit terminée pour commencer à traiter un nouveau pixel.

#### 4.1.2.1/ FILTRE GÉNÉRIQUE

La majorité des composants nécessaires à la réalisation de la méthode LASSR utilise des filtres 2D. Cependant ces filtres n'ont pas les mêmes dimensions et les mêmes fonctions dans les différentes parties de l'architecture de la méthode. De plus, il est nécessaire d'étudier l'impact de la taille du voisinage de calcul des attributs nécessaires à la classification de texture.

C'est pourquoi un filtre 2D générique a été conçu dans le but de pouvoir générer automatiquement différents filtres 2D aux dimensions désirées. La généralité s'étend aussi sur la dynamique des pixels, ou sur la dynamique des coefficients lorsque le filtre correspond à un filtre convolutif. En effet la taille de ces bus est automatiquement adaptée dans le reste du projet.

Le filtre générique est décomposé en deux parties qui sont explicitées dans les sous-sections suivantes. Ces deux parties correspondent respectivement à un générateur de voisinage de pixels GCM, pour *Generic Cache Memory* et à un processeur générique GKP pour *Generic Kernel Processor*.

#### Mémoire cache générique - GCM

Le générateur de voisinage permet de fournir au processeur tous les pixels présents dans le voisinage. Un voisinage entier est transmis à chaque cycle d'horloge. Ce voisinage peut être considéré comme une mémoire cache qui stocke temporairement les voisinages de pixels.

La taille du voisinage est personnalisable par l'utilisateur lors de l'ajout du composant dans le projet HDL. Il est possible de définir une taille de voisinage de  $N \times M$ , où  $N$  et  $M$  sont respectivement la largeur et la hauteur du voisinage. Tous les pixels de ce voisinage ont la particularité d'être accessibles en un seul cycle d'horloge.

Tout comme la taille du voisinage, la dynamique du pixel  $D_{pix}$  est aussi personnalisable avant la synthèse du core.

Lors de la synthèse, le nombre de lignes à retard (FIFO) et de registres à décalage (S) est adapté en fonction des valeurs fixées par l'utilisateur au niveau des paramètres du modèle. Les lignes à retard sont générées en utilisant les FIFO fournies par Xilinx et utilisent des blocs RAMs embarqués, tandis que les registres à décalage utilisent des bascules (flip-flop) internes.

La figure 4.8 présente un exemple de génération de GCM où il est possible de voir les branchements des différentes lignes à retard avec les registres. Les pixels du voisinage sont concaténés dans un seul vecteur  $V_{pix(N \times M)}$  en fonction de la taille du noyau désirée ( $N \times M$ ). La largeur de

l'image, indiquée comme paramètre au composant GCM, permet de configurer la largeur des FIFOs qui composent la ligne à retard. La taille  $T_{Vpix}$  du vecteur  $Vpix_{(N \times M)}$  est donc calculée pour chaque noyau de la manière suivante  $T_{Vpix} = N \times M \times D_{pix}$  bits où  $D_{pix}$  est la dynamique en bits des pixels.

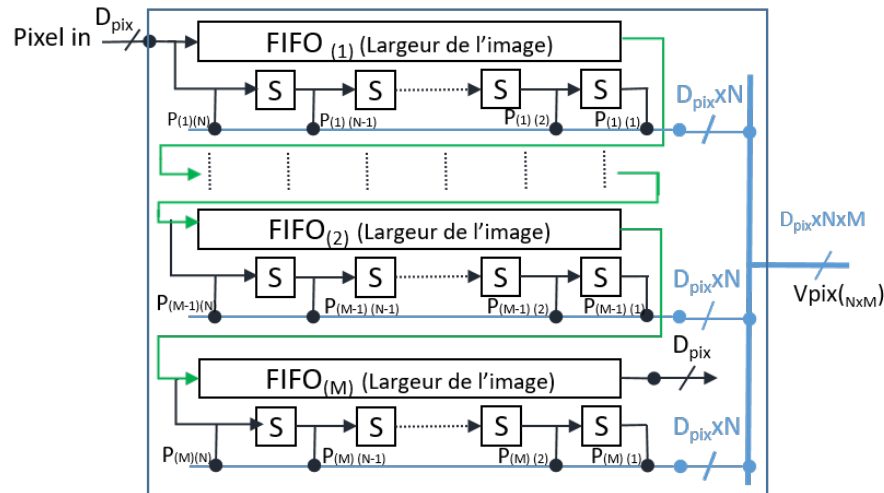


FIGURE 4.8 – Schéma bloc du fonctionnement du Voisinage de Pixel Générique GCM

La concaténation de tous les pixels dans un seul vecteur permet de simplifier les connexions entre le GCM et le GKP. Comme les deux composants sont génériques, ils sont capables d'adapter la taille du vecteur d'entrée (ou sortie) automatiquement.

Comme indiqué précédemment, les lignes à retard dépendent de la génération 7 des FPGAs Xilinx, et des versions 2015.2 et supérieures du logiciel VIVADO. Ce choix est pragmatique. Il serait intéressant dans de futurs travaux de décrire manuellement les FIFOs afin de ne plus être dépendant ni des cibles matérielles, ni de la version du logiciel de synthèse. Il serait alors possible de rendre le composant GCM totalement générique et compatible avec tout type de FPGA fabriqué par d'autres fournisseurs.

### Processeur générique - GKP

Le processeur générique GKP, *Generic Kernel Processor*, est conçu pour fonctionner avec la mémoire cache GCM. Tout comme GCM, sa taille et la dynamique des entrées sont paramétrables, ce qui permet une construction interne automatique. Il permet de réaliser des opérations de convolution 2D à partir du voisinage de pixels concaténés dans le vecteur  $Vec_{(N \times M)}$ . En ayant recours à des jeux de coefficients spécifiques, ce processeur peut effectuer des opérations particulières comme la somme des pixels, la somme des carrés ou une convolution 2D classique avec l'utilisation de coefficients en entrée.

- La somme des pixels  $\sum X$ ,
- La somme des carrés des pixels  $\sum X^2$ ,
- Une convolution 2D.

En effet la somme des pixels correspond alors à une convolution dont tous les coefficients sont égaux à 1, la somme des pixels au carré correspond à une convolution dont les coefficients sont égaux à la valeur courante du pixel. De ce fait, la construction du parallélisme se fait de la même manière quelle que soit l'opération.

Pour réaliser cette convolution un arbre binaire est construit de manière automatique en fonction



de la taille du voisinage de pixel. Une première partie de cette arbre correspond à l'étage des multiplieurs. Un multiplieur est généré pour un pixel du voisinage. Lors de la convolution 2D les multiplieurs sont connectés directement aux coefficients correspondants. En fonction du traitement, cet étage est alors réalisé en employant des multiplieurs  $A \times 1$ ,  $A \times A$  ou encore  $A \times B$ , avec  $A$  le pixel d'entrée et  $B$  le coefficient. Ces différents multiplieurs permettent alors de réaliser respectivement la somme des pixels, la somme des carrés des pixels ou une convolution 2D générique.

Un exemple de génération d'un processeur générique d'une taille de  $5 \times 5$  est présenté dans la figure 4.9. Nous pouvons observer au regard de ce schéma bloc que les coefficients et les pixels proviennent de deux vecteurs différents :  $Vpix_{(N \times M)}$  et  $Vcoeff_{(N \times M)}$ , toutes les données (pixels et coefficients) sont concaténées dans un seul vecteur. Cette concaténation simplifie l'utilisation du composant lorsqu'il est nécessaire de modifier la taille du voisinage.

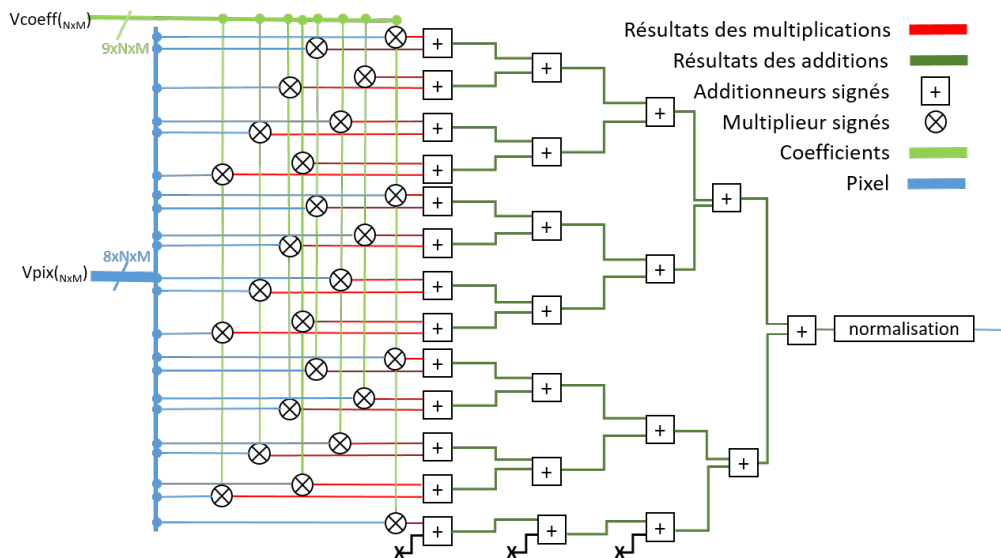


FIGURE 4.9 – Exemple de génération d'un arbre binaire de convolution d'une taille  $N \times M$  à l'aide du processeur générique (avec  $N = M = 5$ ). Les pixels d'entrée, codés sur 8 bits, sont concaténés dans le vecteur  $Vpix_{(N \times M)}$ , et les coefficients, codés sur 9 bits, dans  $Vcoeff_{(N \times M)}$ .

Le composant génère automatiquement le nombre  $2^{\max(M,N)}$  de multiplieurs signés. De ce fait si le voisinage n'est pas carré il génère  $\{2^{\max(M,N)} - (N \times M)\}$  multiplieurs en trop. Chaque coefficient est connecté à un multiplieur, et les multiplieurs excédents sont alors reliés à des coefficients nuls, et sont supprimés lors de la synthèse du composant. La dynamique des multiplieurs est automatiquement mise à jour en fonction de la dynamique des entrées (coefficients et pixel).

Le composant génère les différents étages de l'arbre binaire. De ce fait, il génère un nombre  $S$  d'étages d'additionneurs signés, où  $S = \text{round}(\log_2(N \times M))$ . Les composants nécessaires au fonctionnement de cet arbre sont les multiplieurs et les additionneurs. Ces composants sont aussi décrits de manière générique afin d'optimiser au mieux l'implémentation matérielle. Les additionneurs sont codés pour pouvoir réaliser des additions signées. Le multiplieur générique et le composant de normalisation sont explicités dans les paragraphes suivants.

### Multiplieurs génériques

Les multiplieurs nécessaires aux différents fonctionnements sont décrits manuellement et de manière générique. Cette description générique permet de s'adapter dans ce cas à la dynamique des entrées des multiplieurs. Pour un fonctionnement de type convolution 2D, le multiplieur doit être

adapté à la dynamique des pixels d'entrée, mais aussi aux coefficients d'entrées. Les multiplieurs sont décrits en convention signé et ont la capacité d'utiliser, lorsque cela est possible, les DSP48 disponibles sur certains FPGA. Cela permet d'utiliser au mieux les ressources disponibles dans chaque cible FPGA et d'accélérer les temps de traitements. Ces multiplieurs génériques sont aussi réutilisables dans d'autres composants de l'architecture de la méthode LASSR, mais peuvent tout aussi bien être utilisés dans d'autres projets.

Trois variantes de multiplieurs ont été décrites dans ces travaux :

- Multiplieur signé, dynamique générique ;
- Multiplieur réalisant la fonction carré, dynamique générique ;
- Multiplieur par 1, dynamique générique ;

Le modèle intègre une description unique avec des multiplieurs par 1 mais lors de la synthèse ce type de description est simplifié par l'outil, cependant il a été créé pour des raisons de simplification de développement du processeur générique. Étant donné que le processeur permet de calculer la somme des pixels, la somme des carrés des pixels et de réaliser des convolutions 2D générique, il est plus simple de garder le premier étage de l'arbre binaire construit de manière identique pour les différentes opérations.

#### Normalisation

La normalisation du résultat des calculs du processeur générique est définie dans le composant. Cependant elle est conçue pour seulement normaliser avec des divisions en puissance de  $2^n$ . Lorsque cette normalisation est active, il suffit de donner la puissance  $n$  de la division. Cette division correspond alors à un décalage de bit vers la droite en supprimant les bits de poids faibles du vecteur de résultats. Lors de la normalisation les bits de poids faibles ne sont pas conservés et le résultat de cette normalisation est alors un nombre entier.

Cette normalisation est par exemple utilisée pour la réalisation du filtre passe haut (HPF) du filtre de Gohshi en réalisant une division par 256. Dans d'autres parties du développement, des normalisations sont réalisées en connectant un diviseur issu du catalogue IP de Xilinx lorsque le diviseur n'est pas une puissance de 2.

#### Conclusion

Ce filtre générique peut être utilisé comme un core IP en supportant un grand nombre de configurations différentes. Ce composant utilise à l'état actuel des composants issus des bibliothèque Xilinx (FIFO, DSP), mais cette dépendance peut être réduite en créant des composants "libres".

Les deux composants présents dans le filtre (GCM et GKP) sont décrits de manière générique pour s'adapter au mieux aux configurations désirées par l'utilisateur. Cependant des optimisations se produisent lors de la synthèse du projet VHDL avec l'outil VIVADO ; les composants non utilisés, et les multiplications par 1 sont simplifiées afin de réduire la consommation de ressources dans le FPGA.

Lors de la conception du filtre générique, nous avons pu accroître les composants présents dans notre *boite à outils*. Ces composants de par leur nature *générique* peuvent alors être utilisés dans d'autres blocs de notre architecture et dans d'autres projets de description matérielle.

#### 4.1.2.2/ MÉTHODE DE GHOSHI

Nous avons implémenté la méthode de Gohshi à partir de notre étude réalisée section 3.2. Des travaux de l'auteur [139] montrent que cette méthode a déjà pu être implantée de façon matérielle dans des cibles FPGA. Cependant, très peu d'informations ont pu être extraites de ces publications

scientifiques. C'est pourquoi nos travaux correspondent à une adaptation de la méthode de Gohshi. Cette méthode est décomposée en plusieurs blocs, voir figure 4.10, lui permettant de fonctionner en pipeline, ce qui permet d'accélérer le traitement et de travailler directement dans le flux.

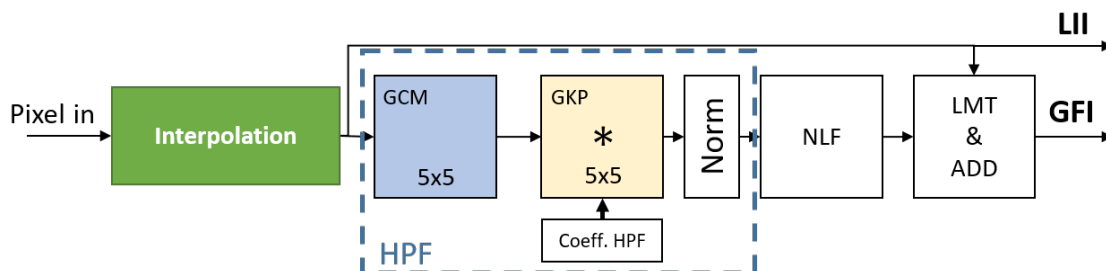


FIGURE 4.10 – Décomposition en blocs de la méthode de Gohshi implantée, le filtre passe-haut *HPF* utilise le filtre générique présenté dans la section 4.1.2.1.

Les différents blocs décrivant la chaîne de Gohshi sont explicités dans les sous-sections suivantes.

### Interpolation

Le bloc d'interpolation provient des travaux de Kronstein, D. [117]. Ce bloc est libre et permet de réaliser des redimensionnements d'image en niveaux de gris et en couleur, avec des tailles d'entrée et de sortie paramétrables. Ce composant, décrit en Verilog, est fourni avec une documentation assez sommaire. Il a été développé pour des cibles Altera. L'auteur a testé le composant sur une cible Altera Cyclone III 3C120, avec une configuration de 10 bits par pixel sur un seul canal de couleur. Les images interpolées au format SXGA 60 Hz sont générées soit avec une interpolation bilinéaire, soit une version plus proche voisin.

Pour fonctionner, le composant génère de manière automatique les lignes à retard nécessaires à la réalisation de l'interpolation. Cette méthode permet de fonctionner tout aussi bien en Altera qu'en Xilinx car la description de ce composant n'est nullement dépendante du type ou de la famille de FPGA utilisé, ni à la version de développement de celui-ci. L'utilisation d'IP propriétaire pourrait amoindrir la généricité du composant.

Nous avons développé un *wrapper* afin de l'intégrer au mieux dans notre projet. Cela nous permet de mieux prendre en main ce composant, de le configurer et de gérer les signaux de contrôle plus simplement.

Depuis Juin 2019, Xilinx propose un core IP nommé Multi-Scaler<sup>1</sup>. Ce core permet maintenant d'interpoler des vidéos dans différentes tailles d'une résolution de  $64 \times 64$  à  $7680 \times 4320$  pixels. Ce core nécessite d'être connecté et interfacé à la mémoire du bus AXI4. Les cibles FPGA compatibles sont très limitées car ce core utilise énormément de ressources matérielles. Ce core pourrait sans aucun doute être utilisé dans de futurs projets de développement matériel en limitant alors la famille de FPGA compatibles. Ce core IP n'étant pas disponible lors de nos travaux de développement nous n'avons pas pu le tester. De plus, ce composant n'est pas libre et ne peut pas être utilisé sur des FPGA de constructeurs concurrents.

### Filtre passe-haut

Le filtre passe-haut est implanté en utilisant le filtre générique précédemment décrit. Cette

1. <https://www.xilinx.com/products/intellectual-property/v-multi-scaler.html>

implémentation est basée sur l'étude réalisée section 3.2.1.3. Pour ce faire, nous configurons le voisinage de pixels pour fournir un voisinage d'une taille de  $5 \times 5$ . De plus ce voisinage est alors connecté au processeur générique, configuré en convolution 2D. Les 25 coefficients sont alors concaténés dans un vecteur correspondant à une entrée du composant.

Afin de réduire la complexité des calculs embarqués au FPGA, les coefficients du noyau Laplacien sont modifiés afin d'avoir des valeurs entières (équation 4.1).

À la suite des calculs, les valeurs sont alors divisées par 256 ce qui correspond à un décalage de 8 bits vers la droite. Cette opération ne coûte rien en terme de ressources matérielles et en latence.

$$\mathbf{X} = \frac{1}{256} \times \begin{bmatrix} 0 & -3 & -5 & -3 & 0 \\ -3 & -15 & -25 & -15 & -3 \\ -5 & -25 & +214 & -25 & -5 \\ -3 & -15 & -25 & -15 & -3 \\ 0 & -3 & -5 & -3 & 0 \end{bmatrix} \quad (4.1)$$

### Fonction non linéaire

La fonction non linéaire à mettre en place est une fonction cubique. Pour ce faire nous mettons en cascade deux multiplieurs signés. La valeur résultante du filtre passe haut est alors multipliée par elle-même deux fois de suite. L'importance de cette fonction, selon l'auteur, Gohshi, est la conservation du signe. Il arrive que l'auteur utilise aussi la fonction carrée  $y = x^2$ . Lors de l'utilisation de la fonction carré, il est alors important de diffuser le bit de signe du vecteur correspondant à la valeur de sortie du filtre passe haut. La sortie de la fonction cubique, NLF, est alors signée. Sa valeur est alors comprise entre  $(-255)^3$  et  $255^3$ .

### Limiteur & addition finale

Le limiteur bloque et écrête les valeurs entre  $-255$  et  $255$ . Cette limitation est simplement réalisée avec un comparateur de valeur, et les valeurs sont alors modifiées lorsqu'elles n'appartiennent pas à l'intervalle  $[-255, 255]$ .

Le dernier bloc de la méthode de Gohshi correspond à l'addition finale du pixel résultant de l'interpolation, et du pixel en sortie du bloc NLF. La valeur du pixel interpolé varie entre  $[0, 255]$ , et la valeur du pixel limité varie entre  $[-255, 255]$ . Il est alors possible que la valeur de sortie du pixel soit négative, aucune information n'est communiquée par l'auteur à ce sujet.

Pour notre implémentation, en référence avec l'étude réalisée en section 3.2.1.5, l'addition finale est réalisée en suivant l'équation 3.13. Seule une partie de la valeur des pixels limités est ajoutée au pixel interpolé. Un nouvel écrêtage est réalisé afin que la valeur du pixel de sortie soit comprise dans l'intervalle  $[0, 255]$ .

#### 4.1.2.3/ ANALYSE DE TEXTURE

Comme il a été décrit dans la section 3.3, le bloc réalisant l'analyse de texture utilise une méthode d'apprentissage basée sur l'extraction d'attributs de l'image source. La méthode du Boosting a été choisie lors de la comparaison de ses performances avec la méthode de SVM (section 3.3.1.4). La méthode de Boosting a sélectionné la variance du gradient de Sobel comme unique attribut pertinent pour la classification. Le vecteur d'attributs est donc  $F(i, j) = \{\sigma^2\}$ . Cet attribut est calculé dans une sous-fenêtre glissante de taille  $n$ .

Le traitement correspond à une mise en pipeline de différents blocs de traitement. Tout d'abord un gradient de Sobel est réalisé, le calcul de l'attribut est ensuite effectué et la dernière étape correspond à la phase de décision.

L'architecture du bloc correspondant à l'analyse de texture est présentée dans la figure 4.11, le calcul du gradient de Sobel est réalisé en utilisant le filtre générique précédemment décrit.

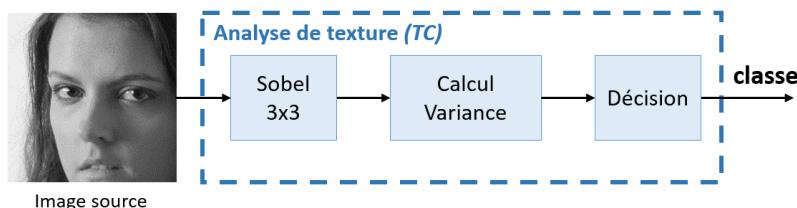


FIGURE 4.11 – Architecture du bloc correspondant à l'analyse de texture (TC), mise en pipeline du gradient de Sobel  $3 \times 3$ , du calcul de la variance  $\sigma^2$  et de la fonction de décision.

Le but de cette sous-section est de tout d'abord expliciter la description matérielle de ce composant et de choisir la dimension  $n$  de la sous-fenêtre de calcul de l'attribut afin de trouver un compromis entre l'erreur globale de classification et la quantité de ressources matérielles utilisées pour son calcul.

### Fitltre de Sobel

Tout d'abord le gradient de Sobel est réalisé en utilisant le filtre générique configuré avec une taille de voisinage de  $3 \times 3$ . Comme le filtre de Sobel se calcule à partir du gradient vertical  $G_x$  et du gradient horizontal  $G_y$ , il est nécessaire de connecter deux processeurs génériques au voisinage  $3 \times 3$ ; un processeur pour  $G_x$  et un autre pour  $G_y$ .

La normalisation se fait à la suite des processeurs avant d'ajouter  $G_x$  à  $G_y$ .

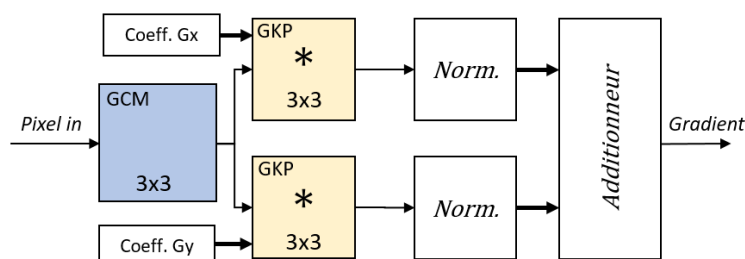


FIGURE 4.12 – Schéma bloc du calcul du gradient de Sobel en utilisant les composants du filtre générique en mode convolution

La normalisation se réalise en faisant la valeur absolue, ce qui correspond à un simple test sur le bit de signe, et un complément à 2 lorsqu'il est négatif.

### Calcul de la variance du gradient de Sobel

Le calcul de la variance  $\sigma^2$  et de la moyenne ont été proposées dans des études précédentes [65]. Cependant l'architecture de gestion des données n'étant pas la même, nous avons préféré utiliser les composants génériques présentés précédemment afin de mettre en place la description structurale matérielle réalisant le calcul de la variance présenté dans l'équation 4.2. Le calcul est réalisé dans une fenêtre glissante d'une taille  $N = n \times n \text{ pixels}$ .

$$\sigma^2 = \frac{\sum_{i=1}^N (x_i^2) - (\sum_{i=1}^N x_i)^2 / N}{N} \quad (4.2)$$

Il est alors possible de modifier l'équation 4.2 afin de pouvoir la décrire plus facilement en langage de haut niveau, nous obtenons alors l'équation 4.3. Cette équation permet alors de décomposer le calcul de la variance en différents blocs qui peuvent alors être calculés en pipeline en suivant le schéma bloc présenté dans la figure 4.13.

$$\sigma^2 = \frac{N \sum_{i=1}^N (x_i^2) - (\sum_{i=1}^N x_i)^2}{N^2} \quad (4.3)$$

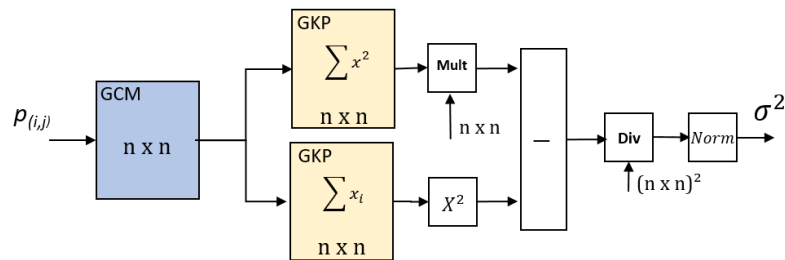


FIGURE 4.13 – Calcul de la variance en utilisant les composants GKP et GCM sur une fenêtre de taille  $n \times n$ , avec  $N$  le nombre de pixels de la fenêtre.

Tous les composants nécessaires à la description de cette architecture ont été décrits lors de la création du filtre générique. De ce fait il a été nécessaire de connecter le voisinage générique à deux processeurs génériques. Le premier processeur configuré comme *somme des carrés* des éléments qui composent le voisinage, et le second processeur permettant de réaliser la somme des éléments du voisinage. Pour réaliser la division finale nous utilisons un diviseur disponible dans le catalogue IP de Xilinx. En effet, la taille de la fenêtre de calcul  $N = n \times n$  n'est pas obligatoirement une puissance de 2. Une normalisation est effectuée afin que la variance soit comprise entre 0 et 255. Cette mise à l'échelle finale est réalisée de la même façon lors de l'apprentissage hors-ligne.

### Mise en place de la fonction de décision

L'outil *Decid* développé dans le laboratoire permet de générer les fonctions de décisions pour les méthodes d'apprentissages SVM et Boosting dans différents types. Il est possible d'exporter directement la fonction de Boosting en langage VHDL pour une intégration matérielle, mais aussi en C++ pour une utilisation logicielle. De plus les programmes décrits en C++ peuvent être transcodés en VHDL, Verilog et SystemC pour une implantation matérielle.

De plus, en fonction encore de la complexité de la fonction de décision il est aussi possible à *Decid* de générer la fonction de décision dans une mémoire ROM, en créant automatiquement le fichier de configuration COE, lorsque le nombre d'attributs est faible (un ou deux). Cette méthode permet d'intégrer directement la fonction de décision dans une mémoire ROM, ce qui donne un accès très rapide à la classe de sortie (un cycle d'horloge).

Pour notre application, en fonction des choix de classifieurs que nous avons pu faire dans les sections précédentes, il nous a été possible d'implanter notre fonction de décision dans une mémoire

ROM. En effet, un seul attribut  $F(i, j) = \{\sigma^2\}$  a été sélectionné par le Boosting. La mémoire Rom contient alors 256 valeurs codées sur 1bit qui correspondent à la classe de sortie de la fonction de décision. L'attribut  $F(i, j) = \{\sigma^2\}$  correspond alors à l'adresse de cette mémoire ROM.

### Choix de la taille de la fenêtre utilisée pour le calcul du vecteur d'attributs

Il a été nécessaire de définir la taille de la sous-fenêtre utilisée pour le calcul du vecteur d'attributs. En effet, l'étude précédente, section 3.3.1.4, a présenté le choix de la méthode de classification en fonction de l'erreur globale de classification. Le choix s'est porté sur la méthode de Boosting. Cependant la taille de la sous-fenêtre n'a pas été définie. Ce dernier choix repose sur le compromis entre l'erreur globale de classification du Boosting, et du taux d'utilisation du FPGA.

Le taux d'utilisation des ressources du FPGA est obtenu lors de l'étape de placement-routage réalisé avec le logiciel Vivado. La cible d'implémentation est le Virtex 7<sup>2</sup> intégré à la carte de développement VC707.

Le graphique présent dans la figure 4.14 met en avant l'évolution du taux d'erreur globale de classification et du taux d'utilisation en cellules logiques (noté "Slices") du FPGA de la carte de développement VC707 en fonction de la taille  $n$  de la sous-fenêtre nécessaire au calcul du vecteur d'attributs. La mise en place du filtre générique nous permet alors de tester simplement l'évolution de cette taille de sous-fenêtre. En effet, l'architecture nécessaire au calcul du vecteur d'attributs reste la même et il est seulement nécessaire de réaliser différentes synthèses à l'aide de l'outil Vivado pour obtenir le taux d'utilisation des ressources du FPGA.

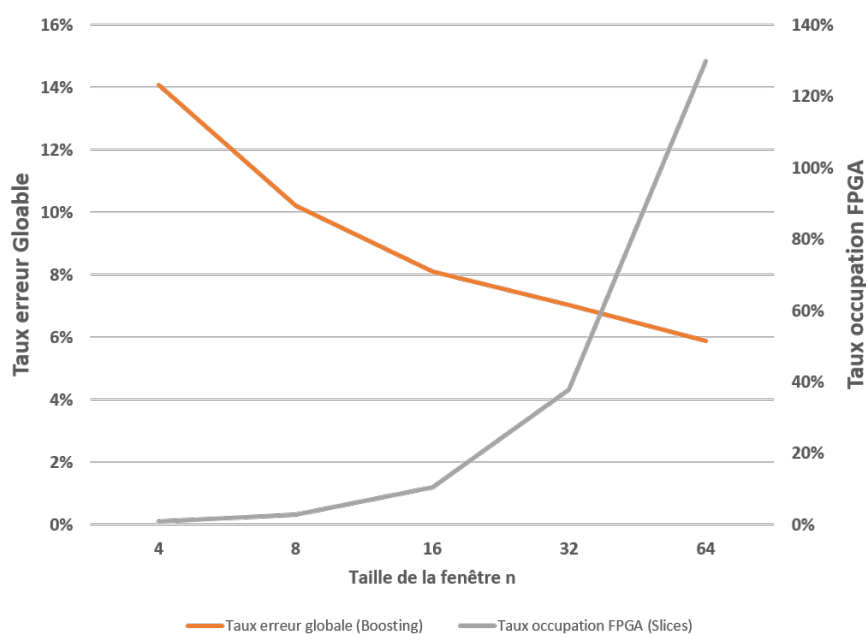


FIGURE 4.14 – Évolution du taux d'erreur globale pour la méthode de Boosting et du taux l'utilisation des ressources du FPGA Virtex 7 (Slices) en fonction de la taille de la sous-fenêtre  $n$  pour le calcul des attributs.

Nous pouvons voir que lorsque la taille  $n$  croît les ressources nécessaires à la création du filtre générique augmentent rapidement. Il est possible de voir que les ressources du FPGA sont insuffi-



santes pour la création d'un filtre générique d'une taille  $n = 64$ . La création de ce filtre nécessiterait à lui seul 130% des ressources du FPGA.

Pour cette application, le seuil d'acceptabilité de l'erreur de classification est fixé empiriquement à 10%. Ce taux d'erreur est atteint lorsque la taille de la fenêtre est de  $n = 8$ , ce taux diminue lorsque la taille de la sous-fenêtre augmente. En terme d'utilisation des ressources matérielles, les tailles  $n = 8$  et  $n = 16$  sont acceptables, elles occupent respectivement 2.8% et 10.5% des ressources du FPGA sélectionné pour les tests.

Nous choisissons alors la taille de sous-fenêtre  $n = 16$  comme meilleur compromis entre le taux d'utilisation des ressources du FPGA et le taux d'erreur globale de classification.

#### 4.1.2.4/ FUSION LOCALE

L'objectif de ce composant est de fusionner les pixels sortants de GFI et de l'interpolation de Lanczos en fonction du masque binaire obtenu lors de la classification de texture. Par conséquent, la luminance des pixels du voisinage est utilisée afin d'ajuster la luminance du pixel LASSR résultant. Ce bloc est décrit dans la figure 4.15.

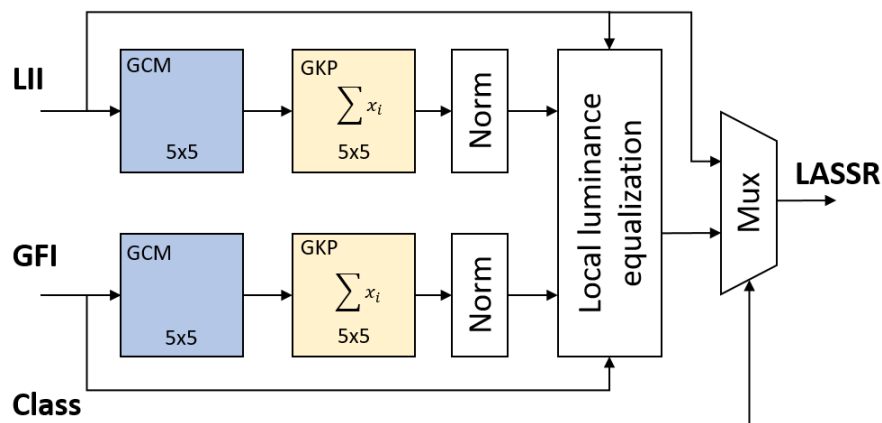


FIGURE 4.15 – Fonctionnement de la fusion locale, décomposé en différents blocs utilisant le filtre générique

Puisque notre implémentation de la méthode de Gohshi élève la luminance, nous mettons en place une égalisation locale de la luminance d'une taille de voisinage  $5 \times 5$  (équation 3.18) décrit précédemment dans la section 3.3.2), rappelé dans les équations 4.4 et 4.5.

$$LASSR(i, j) = LII(i, j) \times (1 - y(i, j)) + GFI_{eq}(i, j) \times y(i, j) \quad (4.4)$$

$$GFI_{eq}(i, j) = GFI(i, j) + (\overline{X_{LII}(i, j)} - \overline{X_{GFI}(i, j)}) \quad (4.5)$$

L'adaptation de la luminance, présentée dans l'équation 3.18, permet d'ajuster la luminance du pixel issu de la méthode de Gohshi avec le pixel interpolé. Il est nécessaire de calculer la moyenne de la luminance locale avec un voisinage  $5 \times 5$ . Pour calculer les deux moyennes, un voisinage d'une taille de  $5 \times 5$  est configuré et connecté au processeur générique. Le processeur générique est alors configuré en sommateur des pixels du voisinage, ce qui permet après une normalisation, de calculer la moyenne de la luminance dans cette sous-fenêtre. La taille du voisinage n'étant pas

une puissance de 2, il est alors nécessaire de connecter un diviseur. Ce diviseur provient de la bibliothèque d'IP de Xilinx. Cependant le "reste" de la division n'est pas utilisé, ce qui permet de conserver des valeurs de pixels entières.

La figure 4.15 met en avant l'architecture en blocs du composant réalisant la fusion locale. Le calcul des moyennes locales est réalisé en parallèle dans des processus différents avant d'être utilisés dans un bloc commun pour calculer la valeur du pixel issu de la méthode de Gohshi normalisé localement au niveau de la luminance. Les valeurs des pixels sont alors utilisées pour la génération du pixel LASSR. L'équation réalisant la fusion finale en fonction du pixel de Gohshi égalisée, du pixel interpolé et de la classification binaire correspond alors à un multiplexeur à deux entrées de données et une entrée de sélection. La sélection correspond alors au bit correspondant à la classe locale de la texture.

Un bloc permettant de gérer la synchronisation entre le bit de sélection et les signaux GFI et LII est créé afin d'éviter les erreurs. L'analyse de texture est réalisée sur les pixels d'entrée en parallèle de la branche réalisant l'interpolation pour le traitement de Gohshi (figure 4.7). Le bit de sélection étant généré avant le pixel provenant de GFI, il est nécessaire de stocker ce signal et de le synchroniser par la suite avec les signaux provenant de GFI et LII. De surcroit, le bit de sélection est généré à partir de l'image LR et est, de ce fait, utilisé par 4 pixels de l'image HR.

### 4.1.3/ RÉSULTATS D'IMPLANTATION ET DISCUSSION

#### 4.1.3.1/ RÉSULTATS

La méthode LASSR a été implémentée en utilisant le logiciel de synthèse Vivado 2019.2. Cette implémentation atteint une fréquence de fonctionnement de 142.85 MHz sur un FPGA Virtex 7<sup>3</sup> de Xilinx en respectant une contrainte horloge de 7 ns. Cette fréquence de travail permet d'obtenir des images HR 4k (4096 × 2048 pixels) à partir d'images 2k (2048 × 1024 pixels) à une fréquence de 17,03 images par seconde. Comme indiqué dans le tableau 4.1, le core LASSR utilise 15245 Slices LUT, 28 BRAMs, et 315 blocs DSP, représentant seulement 5.02% des Slices LUT, 2.72% des BRAMs, et 11.25% des blocs DSP du FPGA utilisé.

Parmi les trois modules principaux, c'est le bloc réalisant la fonction de classification de texture qui consomme la plus grande proportion de ressources matérielles, avec environ 75% des Slices LUT (environ 4% des Slices du FPGA) et 87% des blocs DSP (environ 10% des DSP du FPGA). Il serait intéressant d'étudier l'impact réel du taux d'erreur de classification sur la qualité de l'image LASSR. En fonction de cet impact il serait sans doute envisageable d'augmenter le taux d'erreur global de classification dans le but de réduire la taille  $n$  de la fenêtre de calcul de l'attribut.

Ces résultats, en terme d'implantation matérielle, permettent d'imaginer deux scénarios possibles :

- Utilisation de puces FPGA plus petites et moins onéreuses en remplacement du Virtex-7 xc7vx485tffg1761-2, en conservant les mêmes performances ;
- Atteindre de meilleures fréquences images en divisant l'image d'entrée en plusieurs sous-images afin de les traiter en parallèle sur le Virtex-7 en exécutant chaque instance du core en parallèle sur des portions différentes de l'image d'entrée.

Les deux scénarios ont été étudiés, de façon théorique dans les sous sections suivantes. Les résultats d'implémentations sur deux familles de puces FPGA plus petites sont présentés dans la section 4.1.3.2, et une parallélisation du core réalisant notre méthode LASSR dans le Virtex-7 est explicitée section 4.1.3.3.

---

3. xc7vx485tffg1761-2

Ressources Matérielles	Slices LUT	BRAMs	DSP
Super Résolution	1546	6	38
Classification de Texture	11450	10.5	275
Fusion Locale	2249	11.5	2
Utilisation Totale	15245 (5.02%)	28 (2.72%)	315 (11.25%)

TABLE 4.1 – Utilisation des ressources matérielles du FPGA Virtex 7 (VC 707) pour la méthode LASSR séparée en blocs SR , TC & Fusion

#### 4.1.3.2/ IMPLANTATIONS MATÉRIELLES SUR PUCES ARTIX-7 ET SPARTAN-7

Nous choisissons comme cibles *économiques* les puces Artix-7 a200tfbg676-2 (carte de développement AC 701) et Spartan-7 7s100fgga676-2 (carte de développement SP 701). Le prix de ces FPGA, au moment de ces travaux, est environ 15 fois inférieure au Virtex-7 de la carte VC 707 pour l'Artix-7 et 33 fois inférieure pour le Spartan-7. De plus les capacités matérielles sont nettement inférieures (tableau 4.2).

	Virtex 7	Artix 7	Spartan 7
Slices LUTs	485 760	215 360	64 000
DSP	2 800	740	160
BRAMs	37 080 kb	13 140 kb	4 320 kb
Prix	4 178.07 €	270.27 €	126.36 €

TABLE 4.2 – Comparaison entre le Virtex-7 (xc7vx485tffg1761-2), l'Artix-7 (7a200tfbg676-2) et le Spartan-7 (7s100fgga676-2) en terme de ressources matérielles disponibles et de coût. Le prix correspond aux prix constatés sur mouser.fr pour la puce FPGA seule.

Les tableaux 4.3 et 4.4 présentent respectivement les résultats d'implantation d'un seul core LASSR sur les cibles Artix-7 et Spartan-7. Ces résultats d'implantation ont été obtenus avec le logiciel Vivado 2019.2.

Concernant l'implantation sur la puce Artix-7, l'utilisation en ressources matérielles est significativement la même que sur la puce Virtex-7 (tableau 4.1). Ceci est cohérent étant donné qu'aucune modification de code n'est réalisée pour l'implémentation sur les deux cibles. Ceci est possible d'une part grâce à la compatibilité de toutes les puces FPGA de la série 7 de Xilinx, et d'autre part au fait que la puce Artix-7 comporte assez de ressources matérielles pour l'implantation d'un seul core. Cependant le taux d'utilisation est beaucoup plus important, car le FPGA utilisé est beaucoup plus petit que le Virtex-7. Enfin une fréquence de 133 MHz est atteinte avec une horloge de 7.5 ns. Cette fréquence de travail permet d'obtenir des images HR 4k (4096 × 2048 pixels) à partir d'images 2k (2048 × 1024 pixels) à une fréquence de 15.89 images par seconde.

Enfin, concernant l'implantation sur la puce Spartan-7, l'utilisation des ressources est différente que pour les implantations sur Virtex-7 et Artix-7. Cette modification d'utilisation des ressources est principalement due au manque de blocs DSP sur le Spartan-7. Pour fonctionner les blocs DSP

Ressources Matérielles	Slices LUT	BRAMs	DSP
Super Résolution	1548	6	38
Classification de Texture	11431	10.5	275
Fusion Locale	2248	11.5	2
Utilisation Totale	15227 (11.39%)	28 (7.67%)	315 (42.57%)

TABLE 4.3 – Utilisation des ressources matérielles du FPGA Artix 7 (AC 701) pour la méthode LASSR séparée en blocs SR , TC & Fusion

manquant sont remplacés automatiquement par des cellules logiques (Slices LUT) modifiant alors l'utilisation des ressources, mais aussi les fréquences de fonctionnement. Par exemple, une opération mathématique est réalisée beaucoup plus rapidement sur un bloc DSP que sur les cellules logiques. C'est pourquoi une fréquence de 83 MHz est atteinte avec une horloge de 12 ns. Cette fréquence de travail permet d'obtenir des images HR 4k (4096 × 2048 pixels) à partir d'images 2k (2048 × 1024 pixels) à une fréquence de 9.93 images par seconde.

Ressources Matérielles	Slices LUT	BRAMs	DSP
Super Résolution	3078	6	25
Classification de Texture	17 441	10.5	127
Fusion Locale	2242	11.5	2
Utilisation Totale	22744 (35.53%)	28 (23.33%)	154 (96.25%)

TABLE 4.4 – Utilisation des ressources matérielles du FPGA Spartan 7 (SP 701) pour la méthode LASSR séparée en blocs SR , TC & Fusion

Ces deux implantations montrent la possibilité d'implanter un module LASSR sur des puces FPGA moins onéreuses. Cependant, cela met en avant le fait, qu'en utilisant des puces FPGA plus petites, la vitesse de traitement diminue. La fréquence de travail possible sur l'Artix-7 n'est pas significativement différente de la fréquence de travail obtenue avec le Virtex-7 et l'Artix-7 peut être une solution lorsqu'il est nécessaire de mettre en place un module LASSR sur une cible FPGA moins onéreuse que le Virtex-7.

#### 4.1.3.3/ MISE EN PARALLÈLE DE MODULES LASSR

Le but de cette section est d'évaluer la possibilité d'implanter plusieurs modules LASSR dans la carte de développement VC707 de Xilinx afin d'étudier la possibilité d'accélérer le traitement. Le schéma de parallélisation correspond à du parallélisme de données et de tâches de type MIMD : il

s'agit d'une suite d'opérations faites sur des données différentes.

Cette étude est décomposée en deux parties distinctes :

- Une étude sur la duplication des modules : étude visant à voir combien il est possible de faire fonctionner de modules LASSR dans le FPGA, et à quelle fréquence il est possible de travailler ;
- Une étude de faisabilité sur l'accélération du traitement LASSR par duplication de processus : étude visant à vérifier la possibilité de traiter des images en utilisant plusieurs modules en parallèle tout en gérant les accès mémoire disponibles.

### Duplications des cores LASSR

Dans cette partie de l'étude, seule la duplication des modules est mise en avant. Il est nécessaire de considérer ce FPGA comme une plateforme dont la seule fonction est d'upscaler les images en entrée, sans avoir à se soucier de la gestion des données elles-mêmes.

Il est alors possible d'imaginer qu'une image soit divisée en plusieurs imagettes, et que les données de chaque imagette soient envoyées en parallèle aux différents modules intégrés à la puce FPGA Virtex-7 disponible sur la carte de développement VC 707 de Xilinx.

Le tableau 4.5 présente le taux d'utilisation des ressources du FPGA en fonction du nombre de modules mis en parallèle. Cette mise en parallèle est réalisée pour convertir des images de résolution 2k vers une résolution 4k. La fréquence image indiquée est calculée pour des images de sortie 4k ( $4096 \times 2048$  pixels) avec une contrainte d'horloge de 7 ns (142.85MHz).

Nombre de modules	im/s	Slices LUT	BRAMs	DSP
× 1	17.03	15398 (5%)	28 (3%)	315 (11%)
× 4	68.12	60285 (20%)	88 (9%)	1260 (45%)
× 5	85.15	75238 (25%)	110 (11%)	1575 (56%)
× 6	102.2	90361 (30%)	132 (13%)	1890 (68%)
× 7	119.2	105263 (35%)	154 (15%)	2205 (79%)
× 8	136.2	120407 (40%)	176 (17%)	2520 (90%)

TABLE 4.5 – Utilisation des ressources FPGA pour l'implantation en parallèle de plusieurs modules LASSR et estimation de la fréquence image (im/s) à partir de l'horloge contrainte à 7 ns.

Ce tableau montre qu'il est possible d'intégrer 8 modules LASSR dans le Virtex 7 permettant une accélération de ×8. Lorsque 8 modules sont implantés en parallèle le taux d'occupation du FPGA est assez important avec 40% des cellules logiques *Slices LUT* et 90% des blocs DSP. La méthode LASSR travaillant directement dans le flux, l'utilisation de mémoire BRAMs est assez faible. Il est possible de voir que beaucoup de ressources sont encore disponibles. Cependant, les ressources en DSP limite la possibilité d'augmenter encore le niveau de parallélisme. Lorsque le nombre de modules est supérieur à 8, des cellules logiques sont utilisées pour remplacer les DSP manquants et les contraintes d'horloges ne sont plus respectées.

### Intégration théorique et possibilité d'échanges mémoire

L'objectif de cette partie est de mettre en avant la possibilité de d'accélération du traitement LASSR dans le FPGA pour atteindre une fréquence image d'au moins 60 im/s. Cette étude théorique, contrairement à l'étude sur la duplication seule des modules, est de vérifier la possibilité de traiter la division de l'image en imagette et de la mise en oeuvre de méthode LASSR.

Afin d'atteindre une vitesse de traitement de 60 im/s il est nécessaire de mettre quatre modules LASSR en parallèle (tableau 4.5 pour une conversion d'images 2k vers 4k). Chaque module doit traiter un quart de l'image. Cette implantation utilise 60285 slices, 88 BRAMs et 1260 blocs DSP, représentant environ 19.86% des slices et 45% des blocs DSP du Virtex-7 utilisé.

Jusque-là le traitement LASSR pouvait fonctionner directement dans le flux sans avoir besoin de stocker des images. Cependant, afin de permettre une mise en parallèle des quatre modules il est nécessaire d'avoir une mémoire image donnant accès aux différentes parties de l'image en même temps.

Cette étude théorique est basée sur l'utilisation de la carte de développement VC 707.

Pour fonctionner il est nécessaire de mettre en place une unité de routage des données. Cette unité, présentée figure 4.16, permet de gérer la division de l'image d'entrée en imagerie, et d'alimenter en données les modules LASSR mis en parallèle. Cette unité gère l'accès à la mémoire DDR3 présente sur la carte de développement VC 707.

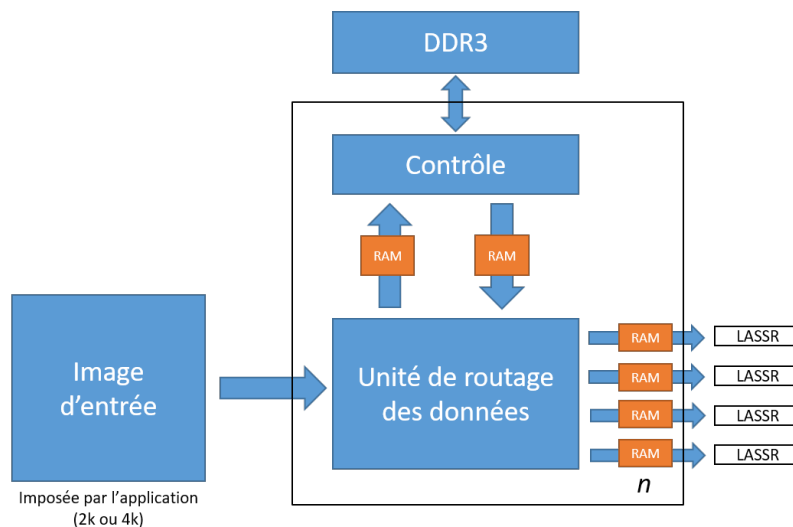


FIGURE 4.16 – Unité de routage des données permettant la gestion des données pour la mise en parallèle de modules LASSR.

Il existe un potentiel goulot d'étranglement au niveau des banques de stockage externes (mémoire DDR3) et potentiellement de l'entrée de l'unité de traitement (LASSR). Le goulot d'étranglement de la mémoire DDR3 est formalisé dans l'équation 4.6, il correspond à la somme de la bande passante entrante  $BP_{IN}$  (équation 4.7) et de la bande passante sortante  $BP_{OUT}$  (équation 4.8).

$$BP_{DDR3} > BP_{IN} + BP_{OUT} \quad (4.6)$$

$$BP_{IN} = (X \times Y) \times D \times F \quad (4.7)$$

Avec  $X$  la largeur de l'image,  $Y$  la hauteur,  $D$  la dynamique en bits et  $F$  la fréquence image.

$$BP_{OUT} = n \times BP_{LASSR} \quad (4.8)$$

Avec  $n$  le nombre de modules LASSR mis en parallèle et  $BP_{LASSR}$  la bande passante sortante du module LASSR.  $BP_{LASSR}$  correspond au volume de données traitées à la fréquence de fonctionnement du module sur une seconde.

Il faut vérifier que l'équation 4.6 soit vérifiée pour  $n = 4$  pour que le compromis complexité / performance soit validé.

#### Vérification de l'équation 4.6 :

D'après les données techniques de la carte de développement, la bande passante  $BP_{DDR3}$  de la mémoire DDR3 est de 12.8 Go/s. La mémoire a une taille de 1 Go ( $128Mo \times 64 bits$ ).

Nous calculons d'abord séparément les bandes passantes  $BP_{LASSR}$ ,  $BP_{OUT}$  et  $BP_{IN}$ .

$$\begin{aligned} BP_{LASSR} &= 142.85 \times 10^6 \times 8 \\ BP_{LASSR} &= 1.06 \text{ Go/s} \end{aligned} \quad (4.9)$$

Une fois  $BP_{LASSR}$  calculée nous pouvons ensuite estimer la bande passante  $BP_{OUT}$  :

$$\begin{aligned} BP_{OUT} &= 4 \times 1.06 \times 10^9 \\ BP_{OUT} &= 4.24 \text{ Go/s} \end{aligned} \quad (4.10)$$

Et ensuite la bande passante  $BP_{IN}$  :

$$\begin{aligned} BP_{IN} &= 2048 \times 1024 \times 8 \times 60 \\ BP_{IN} &= 960 \text{ Mo/s} (0.937 \text{ Go/s}) \end{aligned} \quad (4.11)$$

$$\begin{aligned} BP_{DDR3} &> BP_{IN} + BP_{OUT} \\ 12.8\text{Go/s} &> 0.937 \text{ Go/s} + 4.24 \text{ Go/s} \\ 12.8\text{Go/s} &> 5.177 \text{ Go/s} \end{aligned} \quad (4.12)$$

Il est toutefois nécessaire de gérer un recouvrement des pixels car le traitement que nous réalisons génère des effets de bords. Pour avoir de bons résultats il serait nécessaire de prévoir des recouvrements d'au moins 8 pixels (Le noyau de calcul de l'attribut de classification a une taille de  $8 \times 8$  pixels, et le noyau du filtre passe haut HPF a lui une taille plus petite de  $5 \times 5$  pixels. La bande passante  $BP_{DDR3}$  est largement plus grande que la bande passante nécessaire donc il n'y aura aucun souci pour gérer les recouvrements.

#### Discussion à propos de l'intégration théorique :

Cette partie de l'étude a permis de vérifier de façon théorique la gestion d'une image d'entrée d'une résolution de 2k et la possibilité d'accélérer le traitement LASSR par un facteur de 4. Cette parallélisation des modules LASSR permet aussi d'utiliser au mieux l'espace disponible dans le Virtex-7. Cependant l'inconvénient de cette application est l'ajout d'une unité de routage des données avec l'utilisation d'une mémoire image ce qui fait que ce traitement parallélisé ne peut plus travailler dans le flux mais nécessite d'avoir une grande partie de l'image d'accessible au même moment pour permettre de diviser le flux en quatre.

Les ressources en terme de mémoire BRAM sont assez grandes pour permettre une gestion interne au FPGA car seulement 9% des BRAMs sont utilisés lorsque 4 modules sont mis en parallèle (tableau 4.5). En effet le Virtex-7 dispose encore d'environ 32.18 Mo et une image d'une résolution de 2k codée sur 8 bits mesure 16 Mo. Cependant cette solution interne intensifierait l'utilisation du FPGA et compliquerait la transmission des signaux réduisant alors la fréquence de travail.



#### 4.1.3.4/ COMPARAISON AVEC L'ÉTAT DE L'ART

Le tableau 4.6 présentent les détails de différentes implémentations matérielles sur FPGA de méthodes de Super Résolution. Deux de ces méthodes [174, 194], présentées dans l'état de l'art, utilisent des réseaux de neurones et l'une d'entre elles utilise un algorithme itératif[160].

Seyid et al. [160] ont implémenté la méthode IBP, pour *Iterative Back Projection*, qui est une méthode itérative qui permet d'avoir des résultats qualitatifs de Super Résolution. Leur implémentation matérielle permet de générer des images  $512 \times 512$  à partir d'un ensemble de 20 images basses résolution consécutives ( $256 \times 256$  pixels) à une fréquence de 25 im/s. Cette méthode a été implémentée dans un Virtex 7 mais les ressources utilisées ne sont pas communiquées. Cette méthode est prévue pour utiliser toujours les mêmes ressources du FPGA peut importe la taille de l'image d'entrée. De plus il est nécessaire de réaliser un grand nombre d'itérations pour améliorer la qualité. Les "itérations" étant parallélisées il est possible par exemple d'avoir 17 itérations pour une image de  $256 \times 256$  pixels, et seulement 4 itérations pour une image de  $1024 \times 1024$  pixels. Plus l'image d'entrée est grande, moins il est possible de réaliser d'itérations et moins la qualité de l'image sera élevée. Selon les auteurs il est nécessaire d'avoir au moins 5 itérations pour avoir une qualité suffisante.

Manabe et al. [174] proposent une implémentation matérielle temps réel d'une méthode de SR efficace pour les images contenant des mouvements. Leur méthode utilise des réseaux de neurones convolutifs et permet d'obtenir des images  $1920 \times 1080$  pixels à partir d'images  $960 \times 540$  pixels à une fréquence images de 48 im/s. Leur méthode a été implémentée sur un FPGA Virtex UltraScale dont ils utilisent une très grande partie des ressources disponibles avec 49% des cellules logiques et la totalité des blocs DPS.

Yongwoo et al. [194] proposent eux aussi l'implémentation d'une méthode de SR basée sur l'utilisation de réseaux de neurones convolutifs, mais les auteurs obtiennent des images de meilleures qualités visuelles que d'autres méthodes similaires de l'état de l'art. De plus, leur méthode permet de générer des images haute-résolution 4k à une fréquence image de 60 im/s.

Notre méthode LASSR est capable de traiter un plus gros volume de données que la méthode IBP [160] ( $1920 \times 1080$  contre  $256 \times 256$  pixels) pour des ordres de fréquences images comparables (16 im/s contre 25 im/s). Il nous est possible avec une mise en parallèle de différents blocs LASSR d'augmenter la fréquence image. La méthode LASSR atteint le même niveau de performances que les meilleurs techniques de l'état de l'art basées sur les CNN. De plus, notre méthode utilise moins de ressources du FPGA car les CNN sont connus pour avoir besoin de nombreuses opérations arithmétiques. Par exemple, 100 % DSP sont utilisés dans [174, 194] alors que LASSR nécessite moins de 1 % de DSP.

Ces premiers résultats ont montré les capacités de notre traitement à traiter de grands volumes de données en temps réel, et ce, sans utiliser toutes les ressources matérielles disponibles dans le FPGA de test. De ce fait, il est possible d'augmenter la fréquence image de sortie en parallélisant le traitement dans une multitude de blocs réalisant la fonction LASSR. Il est possible par exemple de diviser l'image d'entrée en 4 sous-images et de traiter en parallèle les 4 sous-images ce qui nous permet d'atteindre des fréquence image de 67 im/s.

Ensuite, lorsqu'un seul module LASSR est implémenté, aucune mémoire tampon n'est utilisée pour stocker les images d'entrées. Il est tout à fait possible de le configurer pour traiter des images plus grandes afin de fournir des images avec des résolutions de sorties égales au format 8k. Cependant la fréquence image serait réduite.

De plus, il pourrait être intéressante de pouvoir réduire les ressources matérielles nécessaires pour

Publication	[160]	[174]	[194]	Notre méthode	
Méthode	IBP	CNN	CNN	LASSR	LASSRx4
Puce FPGA	Xilinx Virtex-7 XC7VX485T	Virtex UltraScale XCVU095	Xilinx Kintex UltraScale XCKU040	Xilinx Virtex-7 XC7VX485T	
Ressources FPGA	N/A	266k LUT (49%) 131 BRAM (7%) 768 DSP (100%)	110k LUT (45%) N/A 1920 DSP (100%)	31k LUT (10%) 28 BRAM (3%) 315 DSP (11.25%)	60k LUT (20%) 88 BRAM (9%) 1260 DSP (45%)
Fréquence Max.	265 MHz	133 MHz	150 MHz	142 MHz	
Résolution entrée	256x256	960x540	1920x1080	1920x1080	
Résolution sortie	512x512	1920x1080	3940x2160	3940x2160	
fréquence image	25 im/s	48 im/s	60 im/s	16 im/s	67 im/s

TABLE 4.6 – Caractéristiques d’implémentations matérielles présentes dans l’état de l’art de méthodes de Super Résolution comparées à notre méthode LASSR avec la mise en place de 1 ou 4 cores en parallèles.

le bloc réalisant la fonction d’analyse de texture, en effet ce bloc utilise à lui seul environ 10.46% des ressources du FPGA, quand le traitement global utilise un peu plus de 12.59%. Cette réduction de ressources permettrait d’implanter notre traitement dans des cibles plus petites tout en conservant les vitesses de traitement actuelles.

Une prochaine étape serait de mettre en place une chaîne complètement embarquée dans une caméra rapide afin de pouvoir augmenter la résolution spatiale de caméras embarquant un FPGA. En effet, pour le moment seuls les principes de co-simulation et de co-design ont pu être testés mais sans pouvoir être entièrement embarqués dans une caméra.

Au sein du laboratoire ImViA, il serait possible d’embarquer, par exemple, notre core IP dans une plateforme [171] embarquant un Virtex d’une génération antérieure et un capteur d’image pour pouvoir tester ce nouveau traitement en temps réel.

## 4.2/ DÉCONVOLUTION TEMPS RÉEL - GPU

L'objectif de cette section est de mettre en avant les étapes nécessaires à la mise en place du démonstrateur à créer pour répondre aux prérogatives du projet EXIST-H2020. Le démonstrateur est composé de la caméra du projet relié à un PC. La tâche de ce système d'imagerie est de corriger automatiquement les défauts des images provenant d'une caméra. Cette correction doit être réalisée en temps réel et les images doivent être traitées dans le flux. Le challenge principal est d'implémenter une méthode capable de traiter les données en temps réel, et à une grande vitesse.

La figure 4.17 présente les éléments du démonstrateur, et le partage des tâches entre les différents acteurs du consortium :

- Thalès Angénieux (TAGX) conçoit l'objectif,
- Adimec conçoit la caméra,
- Imec et notre laboratoire contribuent séparément aux corrections des images provenant de la caméra (caméra + objectif).

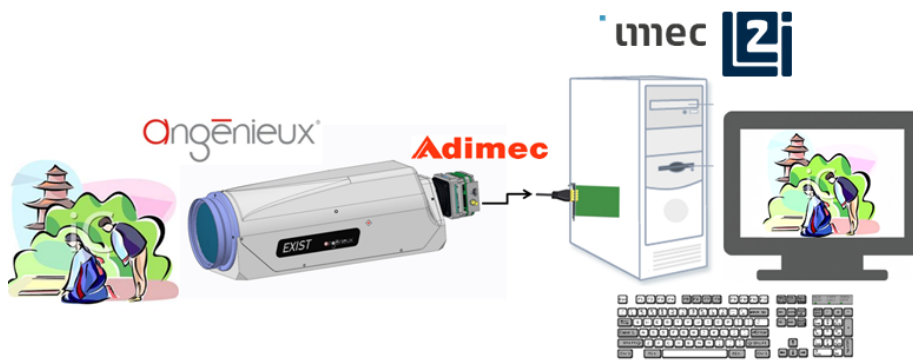


FIGURE 4.17 – Représentation du démonstrateur à réaliser dans le cadre du projet H2020-EXIST.

Les déformations de la lentille sont caractérisées par des PSFs. Le modèle de la lentille et de ses déformations ont été communiqués par le constructeur TAGX. Le fait de connaître les déformations nous donne la possibilité de mettre en place une méthode de déconvolution dite "non-aveugle".

Les méthodes de Richardson-Lucy [9, 10] (RLA) et Wiener [6] sont les méthodes de déconvolution non-aveugles les plus couramment utilisées. Ces deux méthodes donnent des résultats similaires, cependant la méthode utilisant le filtre de Wiener est plus rapide que RLA. Le filtre Wiener n'étant pas itératif, la méthode de Wiener est la meilleure candidate pour travailler dans le flux et à grande vitesse. Elle est utilisée dans le domaine fréquentiel. Le filtre de Wiener permet de donner une estimation du signal d'origine en connaissant les PSFs. Le signal restauré est donné dans l'équation 4.13.

$$\hat{X}(f) = \frac{H^*(f)Y(f)}{|H(f)|^2 + N(f)} \quad (4.13)$$

Où  $\hat{X}(f)$  est le signal corrigé estimé du signal d'origine  $X(f)$ ;  $Y(f)$  correspond à l'image floue et déformée;  $H(f)$  la PSF et  $H^*(f)$  la conjuguée de la PSF  $H(f)$  et  $N(f)$  est la densité spectrale du bruit additif.

La section 4.2.1 présente la méthode de développement implémentée pour tester et réaliser le démonstrateur. La méthode correspond à la gestion des PSFs et à l'application du filtre de Wiener dans le domaine fréquentiel. Ensuite, la section 4.2.2 présente les résultats obtenus avec l'utili-

sation d'image pré-chargées, et des résultats obtenus sur une caméra de transition. Les images pré-chargées correspondent aux images fournies par le fabricant de l'objectif. La caméra de transition est une caméra prêtée par notre partenaire Adimec afin de tester la déconvolution sur un flux d'images provenant réellement d'une caméra. Enfin, la section 4.2.3 présente l'état des travaux au moment de la fin du projet EXIST-H2020 et présente des perspectives sur de possibles évolutions du démonstrateur.

#### 4.2.1/ MÉTHODE

##### 4.2.1.1/ GESTIONS DES PSFs

Le fabricant de l'objectif, Thalès Angénieux TAGX, a fourni pour la mise en place de la déconvolution des images de test d'une résolution de  $3940 \times 2160$  pixels ainsi que les PSFs correspondantes. Les images de tests, mires de développement, comprennent la mire parfaite sans déformations et sans dégradations et des mires déformées et dégradées par les PSF. Un total de 579 PSF par canal ont été fournies pour la correction de l'image entière. Ce nombre de PSF correspond seulement aux PSFs utiles pour la correction d'une demi-ligne de l'image. Les déformations de l'optique sont cylindriques, de ce fait les pixels qui ont la même distance par rapport au centre de l'image utilisent les mêmes PSFs, cependant ces PSFs doivent être pivotées pour que tous les éléments de la PSF conservent toujours leur distance par rapport au centre de l'image. La figure 4.18 présente schématiquement le déplacement de la  $PSF_{ori}$  vers la  $PSF_{rot}$  en suivant un angle de révolution  $\alpha$ , et la figure 4.19 représente un exemple de rotation de PSF.

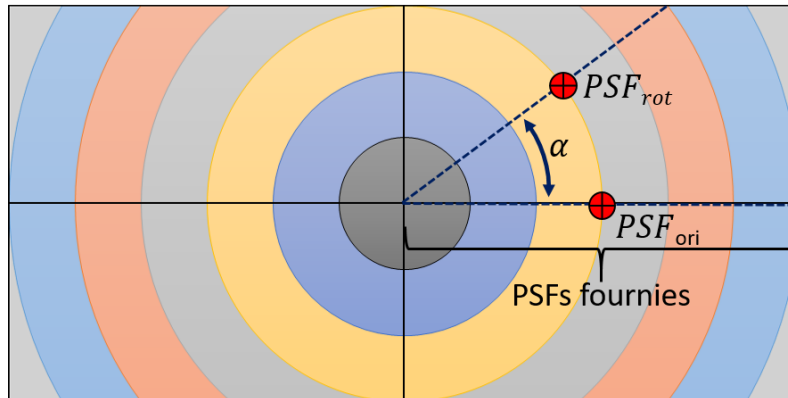


FIGURE 4.18 – Révolution des PSFs en fonction de la distance au centre de l'image. La moitié seulement des PSFs d'une ligne sont fournies, il est nécessaire de les dupliquer en les faisant pivoter. Les différents cercles concentriques représentent de façon schématique diverses PSF. La rotation des PSF est illustrée dans la figure 4.19

La méthode consiste à diviser l'image en blocs. Chaque bloc est traité avec une PSF différente. Tous les blocs peuvent être traités en parallèle. Il est nécessaire de créer un recouvrement pour éviter un effet de "bloc" lors de la reconstruction de l'image. Les PSFs sont pré-calculées (rotation et FFT) pour chaque bloc. La figure 4.20 représente les différentes PSFs utilisées pour corriger l'image. Il est possible de voir dans cette figure, et plus particulièrement dans l'agrandissement d'une partie de l'image, que les blocs se superposent en partie. Cette superposition permet de réaliser un recouvrement nécessaire à la diminution des effets de "blocs" (les effets visibles lors de compression d'images par exemple). L'image correspond à une mire fournie par le constructeur de l'objectif. Les couleurs du quadrillage représentent l'utilisation de la même PSF pour une même

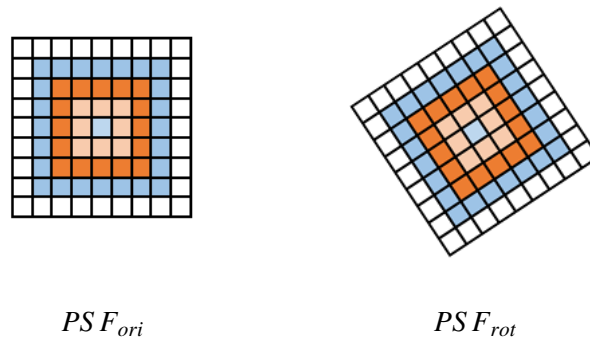


FIGURE 4.19 – Représentation schématique de deux PSFs. La PSF d’origine  $PSF_{ori}$  (fournie par le constructeur) et la PSF pivotées  $PSF_{rot}$  pour conserver la distance des coefficients par rapport au centre de l’image.



FIGURE 4.20 – Image source divisée en blocs, chaque bloc utilise une PSF différente.

distance du pixel au centre de l’image.

Théoriquement, il est nécessaire que chaque pixel soit filtré en utilisant sa PSF correspondante. Cependant, dans la pratique cela demande un grand nombre d’accès mémoire pour la lecture des PSFs. En effet, les accès mémoire sont des éléments qui ralentissent le traitement. Afin de réduire ces accès mémoire, des tests ont été effectués pour utiliser moins de PSF. La figure 4.21 représente la mire de travail divisée avec un nombre moins important de PSFs. La solution est alors d’utiliser une même PSF pour plusieurs pixels voisins.

#### 4.2.1.2/ APPLICATION DU FILTRE DE WIENER

La figure 4.22 présente la méthode générale pour appliquer le filtre de Wiener. L’image à corriger, et la PSF sont tout d’abord transformées du domaine spatial au domaine fréquentiel via une transformée de Fourier rapide (FFT). Ensuite le filtre de Wiener est appliqué dans le domaine fréquentiel. Enfin une transformée de Fourier inverse (IFFT) est appliquée au signal de l’image pour reconstruire le signal en image dans le domaine spatial.

Comme il a été dit dans la section précédente, la correction des images nécessite un grand nombre



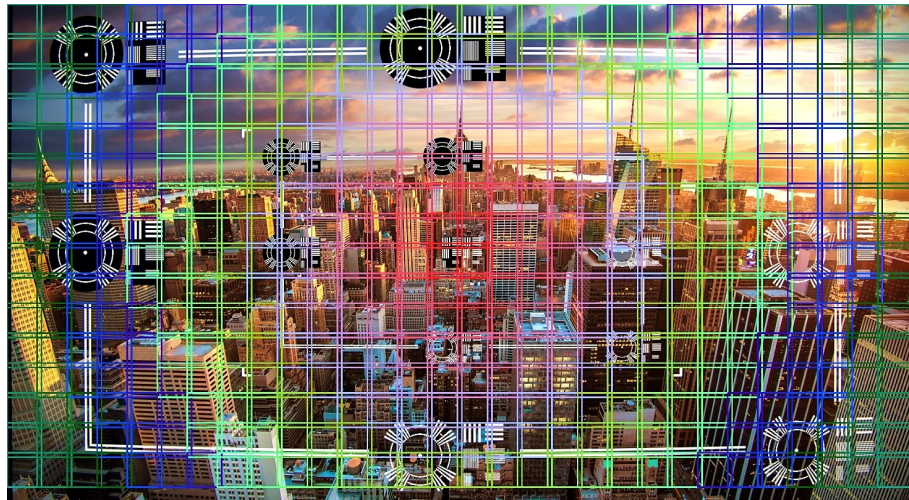


FIGURE 4.21 – Image source divisée en blocs où chaque bloc utilise une PSF différente.

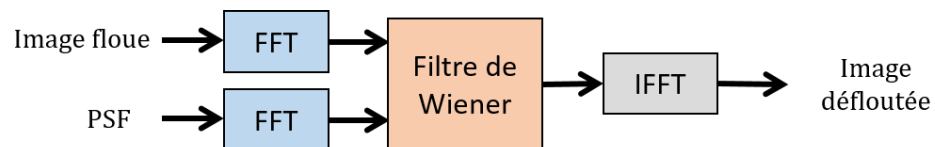


FIGURE 4.22 – Déconvolution non-aveugle d’une image floue avec l’utilisation du filtre de Wiener dans le domaine fréquentiel.

de PSFs. L’image est alors subdivisée en blocs où chaque bloc utilise une PSF différente. Les PSFs restant les mêmes pour corriger deux images consécutives, il est alors possible de les pré-calculer et de stocker en mémoire les transformées de Fourier de ces PSFs. La figure 4.23 met en avant les différentes étapes réalisées pour la correction d’une image avec une multitude de PSFs différentes. L’image étant subdivisée en blocs, chaque bloc est traité en parallèle avec la FFT et l’application du filtre de Wiener. Les blocs sont ensuite transformés dans le domaine spatial avant d’être assemblés pour reconstruire l’image finale.

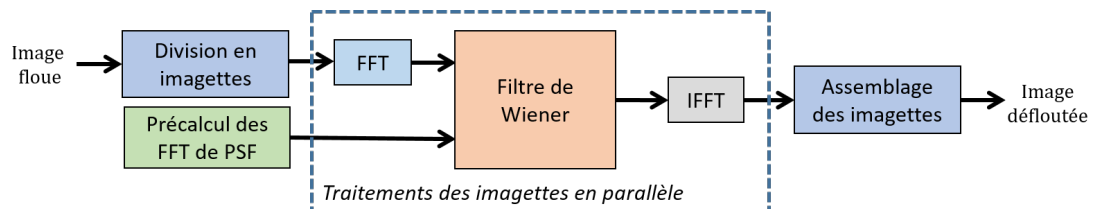


FIGURE 4.23 – Application du filtre de Wiener sur une image subdivisée en blocs. Chaque bloc de l’image correspond à une PSF différente et est traité en parallèle.

#### 4.2.2/ RÉSULTATS

Le développement et la fabrication des objectifs étant très long, notre partenaire Thalès Angenieux TAGX nous a fourni pour les tests un ensemble d’images et de PSFs. L’ensemble d’images est composé d’une image de référence sans dégradation et sans déformation, et une image dégradée. L’image dégradée a été générée avec le modèle mathématique de l’objectif que TAGX conçoit.

Les résultats présentés dans les sections suivantes ont été obtenus de deux manières différentes :

- Tests purement logiciels, avec des images préchargées (section 4.2.2.1).
- Tests sur une caméra de "transition", une caméra intermédiaire au projet permettant de tester l'algorithme de déconvolution dans le flux avec des images provenant d'une vraie caméra (section 4.2.2.1).

Cependant dans les deux cas, le développement a été implémenté en C++ en utilisant la version 3.2 de OpenCv. L'algorithme de déconvolution a été testé sur deux cibles différentes : un processeur (CPU) et un processeur graphique (GPU). Ces deux cibles permettent de comparer l'efficacité de la parallélisation. En effet, le processeur Intel i7-7700K, dispose de 8 coeurs de travail alors que le processeur graphique, GPU présent sur la carte graphique Nvidia GeForce 1080-ti, contient 3584 coeurs. Lors du développement de la méthode de déconvolution la carte graphique était l'une des plus performantes du marché.

#### 4.2.2.1/ RÉSULTATS IMAGES PRÉCHARGÉES ET IMAGE FIXE PROVENANT DE LA CAMÉRA ADIMEC

Les tests ont été effectués sur une image préchargée. En effet, nous ne disposons pour ce test que d'une seule image dégradée et d'une version parfaite sans dégradation, fournies toutes les deux avec une résolution de  $3940 \times 2160$  pixels (4k). Pour cet exemple 550 PSFs par canal sont fournies. Seulement 20 PSF différentes sont utilisées pour ce test.

Il a été possible d'obtenir un temps de traitement de 250 ms avec le processeur i7 4790k, soit une estimation de 4 im/s. Et nous obtenons un temps de traitement de 14.2 ms (70 im/s) sur le processeur graphique de la carte Nvidia 1080-ti. Il existe une grande différence de fréquence entre ces deux résultats. Ces performances sont obtenues grâce à la parallélisation du traitement.

Des études métriques ne sont pas réalisables pour cette partie de l'étude car nous avons à notre disposition un seul couple d'images de test. Cependant il est possible, visuellement, de vérifier si les images sont corrigées. Tout d'abord, la figure 4.24 met en avant les différences visuelles entre une imagerie (200 × 200 pixels) extraite du centre de l'image de test. Il est possible par exemple de voir dans cette figure que l'image corrigée (au centre) est plus nette que l'image dégradée (gauche). Les bords des rectangles blancs semblent être plus francs, et les rectangles sont plus blancs. Nous pouvons considérer qu'aucune distorsion géométrique n'est présente au centre de l'image. Enfin, il est assez difficile de voir les différences entre l'image corrigée et l'image de référence.



FIGURE 4.24 – Imagerie d'une taille de  $200 \times 200$  pixels extraite au centre de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence.

Ensuite les figures 4.25 et 4.26 illustrent des agrandissements ( $100 \times 100$  pixels) d'images extraites de différents bords de l'image de test. Sur ces agrandissements, nous remarquons que le flou



et les aberrations chromatiques visibles dans les images dégradées (gauche) ne sont plus visibles dans les images corrigées (centre). Tout comme il a été vu dans la figure 4.24 les bords sont significativement plus francs même si les distorsions géométriques sont plus présentes sur les bords de l'image.

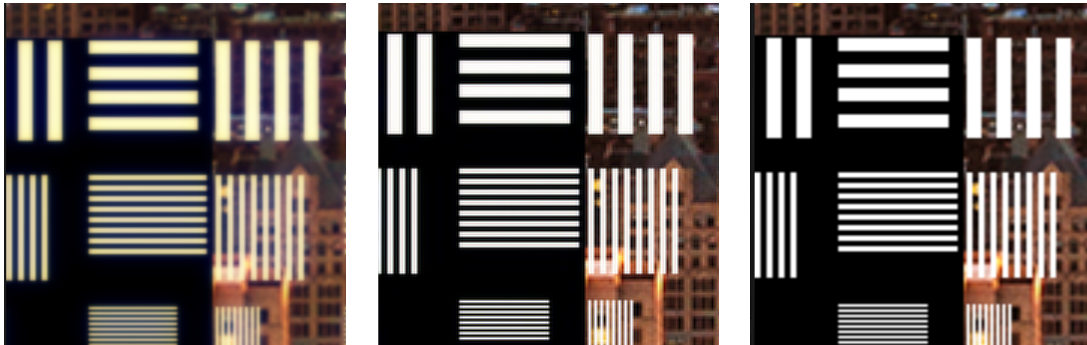


FIGURE 4.25 – Imagerie d'une taille de  $100 \times 100$  pixels extraite sur un bord de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence.



FIGURE 4.26 – Imagerie d'une taille de  $100 \times 100$  pixels extraite sur un bord de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence.

#### 4.2.2.2/ RÉSULTATS CAMÉRA ADIMEC

Nous avons utilisé la caméra fournie par notre partenaire Adimec pour valider l'implémentation, et faire un test réel sur une caméra. La caméra permet d'envoyer des images 2k ( $2064 \times 1544$  pixels) à une vitesse de 60 images par seconde. Le format des images envoyées est sans compression et sans pré-traitement "raw". Contrairement à la caméra développée pour le projet H2020, moins de PSFs sont fournies pour la correction des images : seulement 3 PSFs au lieu de 550 dans l'étude précédente. Afin d'avoir une meilleure correction, des PSFs intermédiaires sont créées par interpolation bicubique. Cela permet d'avoir une meilleure correction.

L'utilisation des PSFs se fait de la même manière que dans l'étude précédente, bien que la caméra soit différente, la méthode de déconvolution est identique.

Un test est réalisé sur une image enregistrée (figure 4.27). Cette image correspond à une capture d'une mire imprimée sur une feuille de papier A4 réalisée à une distance de 3 mètres. Cette image, d'une résolution de  $2064 \times 1544$  pixels est traitée en 45 ms sur le processeur Intel I7-4790k ( soit environ 22 im/s). Cette même image est traitée en 3.3 ms sur le processeur graphique de la carte Nvidia 1080-ti, soit une fréquence image de 300 im/s.

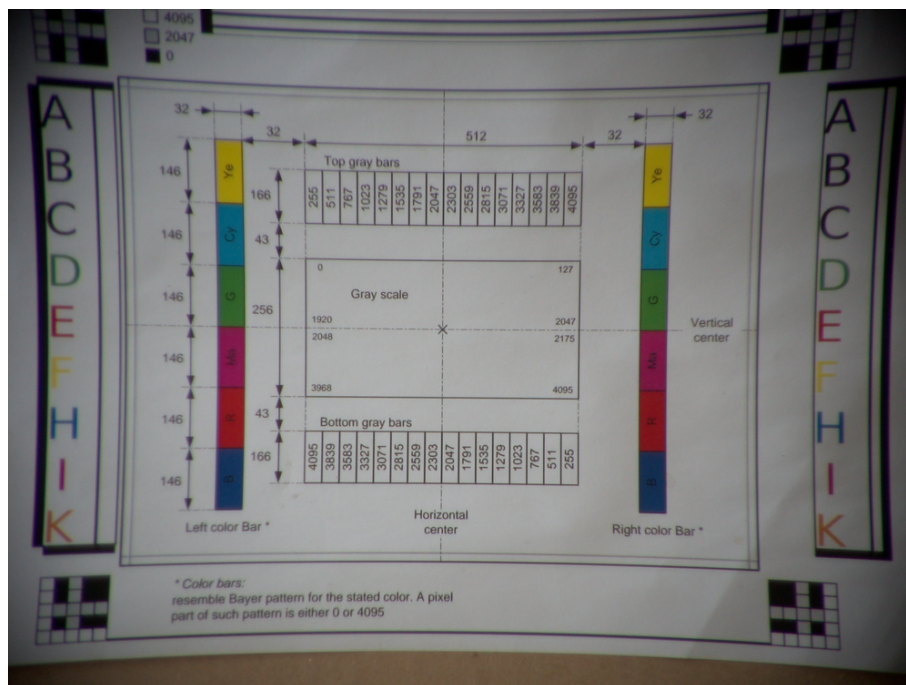


FIGURE 4.27 – L'image d'entrée d'une mire imprimée sur une feuille A4 (210 mm × 297 mm) capturée avec la caméra créée par Adimec. La distance à laquelle l'image a été capturée est de 3 mètres.

Deux imagettes sont extraites de la mire avant et après correction. Ces imagettes sont présentées dans les figures 4.28 et 4.29. Les images provenant directement de la caméra, il n'est pas possible d'avoir l'image parfaite sans dégradation. La figure 4.28 est un extrait au centre de la mire. Les traits et pointillés sont plus nets sur l'image corrigée. Le flou est réduit par la déconvolution.

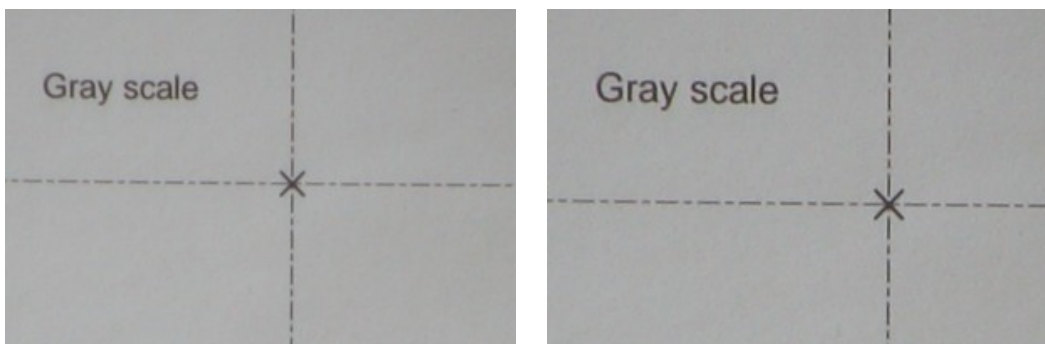


FIGURE 4.28 – Imagette d'une taille de 300 × 200 pixels extraite au centre de la mire papier. L'imagette sans correction à gauche, et l'imagette corrigée à droite.

La figure 4.29 met en avant, à droite, la correction réalisée sur l'image. Les lettres de couleurs sont un peu plus nettes. Cependant, il reste des aberrations chromatiques même si la majeure partie de ces aberrations sont corrigées. L'image de droite (corrigée) paraît moins floue.

#### Réalisation d'un démonstrateur de déconvolution temps réel :

Un démonstrateur a été créé avec la caméra de transition fournie par notre partenaire Adimec. L'architecture du démonstrateur est présentée dans la figure 4.30. Ce démonstrateur est composé de

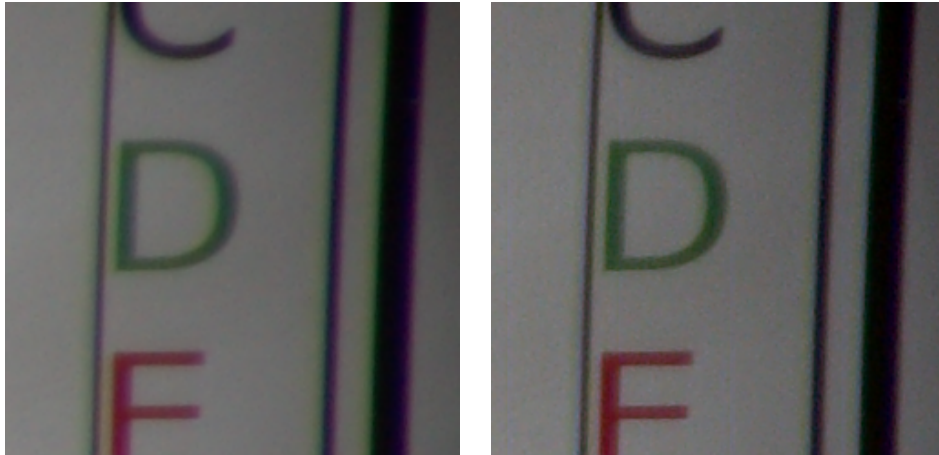


FIGURE 4.29 – Imagerie d'une taille de  $250 \times 250$  pixels extraite sur le coté droit de la mire papier. L'imagerie sans correction à gauche, et l'imagerie corrigée à droite.

la caméra, d'un ordinateur dans lequel se trouve une carte d'acquisition des images, un processeur Intel i7-4790k, et d'une carte vidéo avec un processeur graphique (GPU). Cet ordinateur est relié à un écran afin de visualiser en temps réel la correction des images. Il est possible d'activer et de désactiver la correction en temps réel.

L'implantation sur CPU, bien qu'optimisée, n'est pas assez rapide pour traiter les vidéos en temps réel. Cependant, l'implantation utilisant le GPU permet de traiter les images 2k dans le flux et en temps réel.

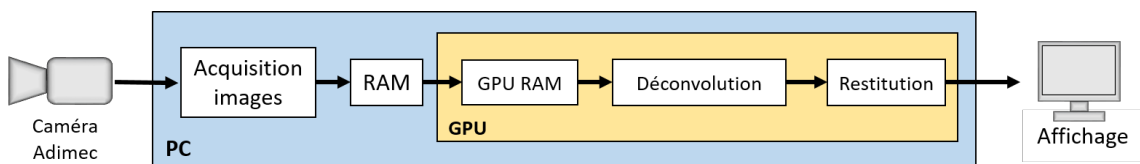


FIGURE 4.30 – Architecture du démonstrateur

La figure 4.31 présente deux images capturées par la caméra. L'image de gauche provient directement de la caméra, sans correction. L'image de droite représente l'image issue du démonstrateur, capturée et corrigée dans le flux. La taille des images étant petite il n'est pas possible de voir les détails des corrections dans cette figure. La figure 4.32 correspond à des extraits des images de la figure 4.31 ce qui permet de visualiser les détails.

Dans la figure 4.32-a des aberrations chromatiques sont bien visibles près des formes noires. Ces aberrations vertes et violettes sont réduites avec la correction par déconvolution (figure 4.32-b). De plus, les contours des perles sont beaucoup plus nets et brillants. De la même manière, il est possible de voir que l'image figure 4.32-d est moins floue que la version originale (figure 4.32-c). De plus, les cils de la marionnette sont nets. Enfin, la suppression du flou laisse apparaître de la texture dans les zones rouges de la marionnette.

### 4.2.3/ DISCUSSION

#### Le démonstrateur

Lors de la fin des travaux de l'ingénieur de recherche, Ali DOUYIEK, qui travaillait sur l'implé-





FIGURE 4.31 – Image 2k capturée par la caméra (*gauche*) et image corrigée par en temps réel (*droite*)

mentation de la méthode de déconvolution, il ne nous a pas été possible d'avoir et de tester la caméra finale, figure 4.33, du projet EXIST-H2020. Cependant, des partenaires du projet nous ont fait parvenir des séquences vidéos acquises par cette caméra. Les résultats du traitement des vidéos n'étaient pas aussi rapides que le traitement réalisé sur une seule image. Il a été possible de traiter environ 15 im/s de résolution 4k avec l'utilisation de 3 PSF. Pour rappel, la vitesse de traitement est de 70 im/s dans l'étude précédente (section 4.2.2.1) réalisée sur une seule image sur des images 4k en utilisant 3 PSFs.

En conclusion, concernant le projet EXIST-H2020, deux démonstrateurs ont pu être créés et sont fonctionnels. L'algorithme de déconvolution a pu être testé pour corriger en temps réel des images de grande résolution 2k et 4k. Cet algorithme fonctionne en utilisant un processeur standard ou un processeur graphique. Cependant, seule l'implémentation sur le processeur graphique (GPU) permet de respecter les contraintes temps-réel pour ces images.

### L'étude qualitative

Il n'a pas été possible de mettre en place une réelle étude qualitative sur les résultats obtenus avec la méthode de déconvolution. En effet, seulement deux images ont été fournies pour ce projet : une image de référence "parfaite", et une image dégradée. Afin de mesurer l'efficacité de notre traitement il aurait été nécessaire d'avoir un grand nombre d'images. Cela aurait permis de réaliser une étude sur la qualité des images. Comme nous l'avons vu auparavant, les métriques mathématiques ne sont pas forcément les plus adaptées pour juger de la qualité. La présence de distorsion perturbe en effet ces métriques.

Enfin, il aurait fallu faire une étude approfondie sur l'impact du nombre de PSF utilisées pour la correction des images. En effet, à l'état actuel le choix du nombre de PSF utilisées est seulement basé sur une appréciation visuelle.

### Perspectives

Il serait intéressant de tester cette méthode de déconvolution sur FPGA. Des implémentations de déconvolution ont déjà été réalisées par le passé. Par exemple, Oliver Sims [86] propose une implémentation de l'algorithme itératif de Richardson-Lucy. Son implémentation permet de traiter des images VGA ( $640 \times 480$  pixels) à une fréquence de 60 im/s. Sa méthode utilise un noyau de convolution de  $11 \times 11$  alors que les nôtres sont de  $25 \times 25$ . L'implémentation a été faite sur un

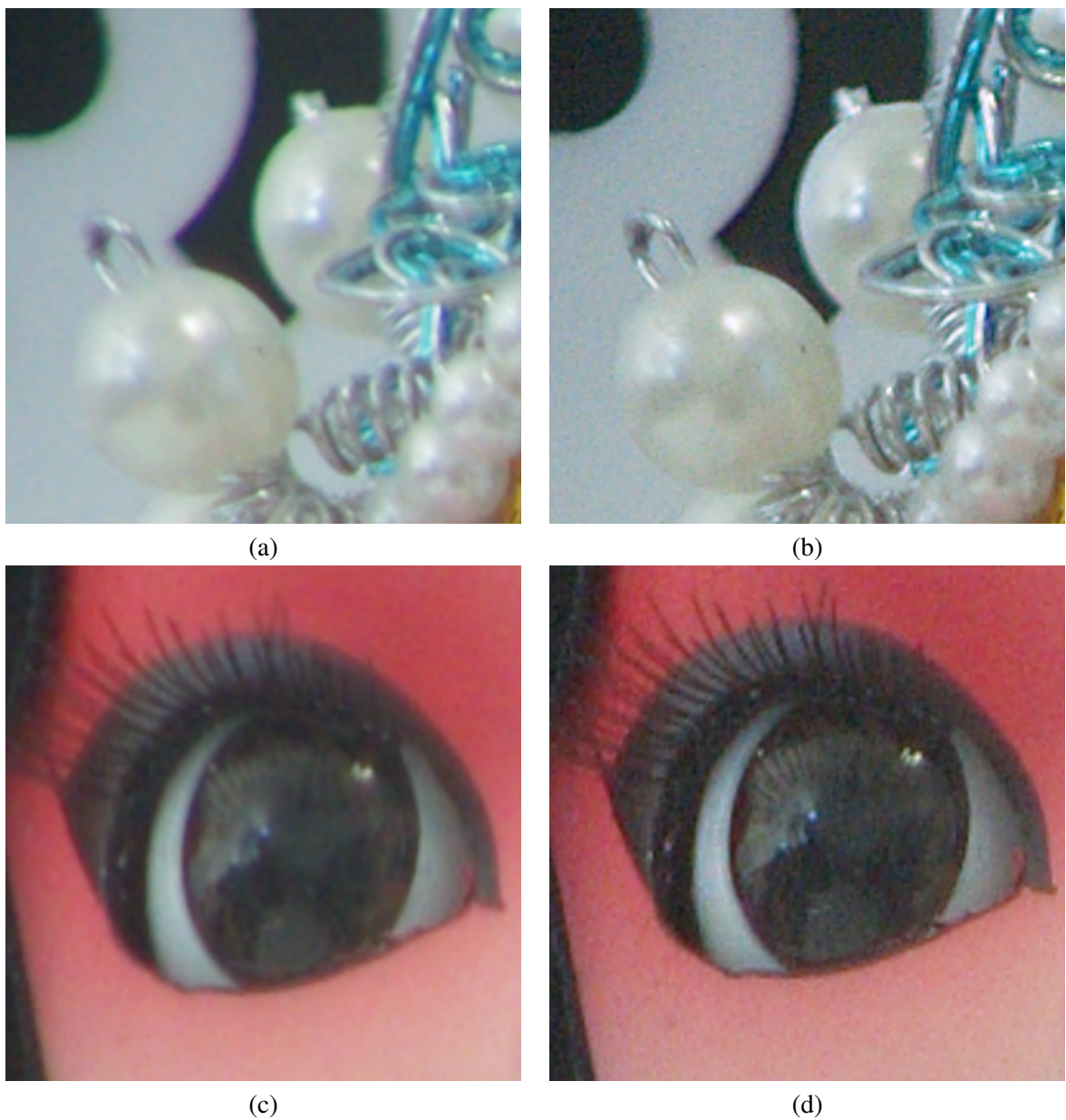


FIGURE 4.32 – Imagettes extraites de l'image brute sans correction (a et c), imagettes extraites de l'image corrigée (b et d)



FIGURE 4.33 – Caméra conçue pour le projet EXIST-H2020, le corps de la caméra développée par ADIMEC, et l'objectif par Thalès Angénieux (TAGX).

virtex-2. Des accélérations restent possibles en utilisant les dernières versions des FPGA (avec des fréquences et capacités bien plus grande). Le traitement d'image 4k/8k sur un FPGA serait un véritable challenge.

## CONCLUSION ET PERSPECTIVES

Les contributions de ces travaux sont liées à la fois aux impératifs applicatifs du projet européen EXIST mais également à la volonté de proposer une méthode originale visant l'amélioration de la qualité d'images et d'en réaliser une étude aussi bien quantitative que qualitative. En effet, une des difficultés majeures liée à l'amélioration de la qualité d'images est inhérente à l'appréciation la "plus objective" possible de ce traitement. De plus, la définition et la mise en œuvre d'une chaîne de traitement temps réel peut être, selon nous, considérée comme une autre des contributions principales de cette étude. Finalement, on pourra également souligner que cette implantation a été réalisée dans un souci de réutilisation (généricité de certains accélérateurs matériels) mais également en considérant des approches de prototypage rapide et de simulation conjointe.

### 5.1/ SUPER RÉOLUTION

Il existe deux grandes familles de méthodes de Super Résolution : MISR *Multiple Images Super Resolution* et SISR *Simple Image Super Resolution*. Les méthodes MISR utilisent des images consécutives dans le but d'extraire des informations inter-pixel présentes entre chaque image grâce aux micro-mouvements. Ces informations sub-pixéliques sont utilisées pour reconstruire le pixel original haute résolution. Les méthodes MISR ont besoin de beaucoup de ressources de calculs et de mémoires. Contrairement aux méthodes de MISR, les méthodes SISR réalisent un traitement de Super Résolution sur une seule image. Certaines techniques de SISR sont basées sur de l'apprentissage : un apprentissage est réalisé avec des bases d'images composées de paires basse résolution (LR) et haute résolution (HR). Des méthodes simples remplacent le patch LR par le patch HR, et d'autres méthodes utilisent un apprentissage profond *Deep Learning* pour reconstruire une image HR à partir d'une seule image LR.

Des auteurs ont ajouté une nouvelle catégorie de méthode permettant d'augmenter la résolution : la Super Interpolation (SI). La SI est une méthode permettant d'augmenter le niveau de détail, d'augmenter la netteté de l'image, sans que le pixel "créé" soit exactement identique au pixel original. En effet, selon ces auteurs certaines méthodes dites de Super Résolution ne le sont pas réellement. Pour eux, la réelle SR doit recouvrir la valeur originale du pixel manquant et non donner une approximation de celui-ci. De ce fait, seules les méthodes MISR, à condition que les images utilisées contiennent réellement des informations sub-pixéliques, appartiennent aux méthodes de Super Résolution. Les méthodes du type *face hallucination* [84] mettent en avant les limites de l'appellation "Super Résolution". Il s'agit de méthodes basées sur de l'apprentissage et permettent de reconstruire des images qui paraissent réelles alors que le résultat obtenu n'est pas aussi proche de l'image d'origine, ces méthodes reconstruisent des visages dont la ressemblance n'est pas identique au visage d'origine. Ce qui compromettrait une reconnaissance faciale.



Dans notre cas, en liens avec le projet H2020-EXIST et les contraintes liées aux démonstrateurs, nous avons focalisé nos recherches sur des méthodes de SISR et de SI. Nous nous sommes alors intéressés à la méthode de Gohshi. Une méthode mise en avant par son auteur pour son fonctionnement dans des cibles FPGA et traitant de grands volumes de données. Nous avons tout d'abord tenté d'implanter cette méthode et nous avons pu en voir ses limites. En effet nous avons pu remarquer que la méthode de Gohshi que nous avons implantée donnait de mauvais résultats dans les zones de l'image où il n'y avait pas de texture (zones lisses, basses fréquences).

De cette implantation nous avons mis en place notre propre méthode originale permettant de réaliser de la SR, une méthode qui est compatible avec un fonctionnement en temps réel sur FPGA. Cette méthode, LASSR pour *Local Adaptive Spatial Super Resolution*, est une amélioration de la méthode de Gohshi. La méthode LASSR adapte le traitement localement en fonction de la texture. Nous avons validé cette méthode grâce à une évaluation psychovisuelle.

Notre méthode permet alors d'améliorer la qualité des images, ajouter des détails à celles-ci. Tout en notant, que les détails sont ajoutés via une fonction non linéaire à la suite d'une interpolation spatiale. Ces détails ne sont alors en aucun cas des informations recouvrées, mais des informations présentes qui sont améliorées et qui, par conséquent, peuvent être au final loin de la vérité de terrain.

L'avantage de notre contribution est le fait que nous avons réussi à surmonter les limites de la méthode de Gohshi en ajoutant une étape de décision dans le traitement. Cette étape de décision est réalisée à la suite d'un apprentissage simple permettant de séparer de façon automatique différentes textures de l'image. Nous avons alors créé une méthode adaptative en fonction de paramètres locaux.

### Le développement matériel

Le développement de LASSR a été réalisé avec le souci de le rendre le plus générique possible et réutilisable dans des configurations variées. Premièrement, LASSR est configurable au niveau pixel (profondeur en nombre de bits) et au niveau image (dimensions), les paramètres étant à indiquer avant la synthèse du module. Deuxièmement, LASSR est presque complètement indépendant du matériel sur lequel il doit s'exécuter. C'est effectivement le cas pour l'acquisition des images car LASSR est interfaçable avec n'importe quel capteur d'images. En revanche, les cibles FPGA supportées sont pour l'instant restreintes à celles fournies par le constructeur Xilinx. Cette contrainte est liée à l'utilisation de FIFOs spécifiques fournies par Xilinx. Une généricité totale passerait par une réécriture complète de ces FIFOs.

Nous avons utilisé le principe de la co-simulation et du co-design pour le développement du module LASSR. Le principe de la co-simulation de Simulink nous permet d'avoir un accès à l'espace de travail de Matlab mais aussi d'accélérer le développement matériel : il nous était possible de seulement développer le module LASSR sans nous préoccuper de la configuration d'un quelconque capteur et de gérer toute l'interface nécessaire à l'obtention des premières images. La co-simulation est une simulation très approfondie mêlant l'espace de travail de Matlab et les logiciels de simulation matériel (ModelSim). Il nous a été possible d'utiliser la co-simulation et le co-design *FPGA in the loop* car nous utilisons une carte de développement de Xilinx qui était configurée de base avec le logiciel Simulink.

Au final, nous avons réussi à mettre en place un module réalisant la méthode LASSR en temps réel et créer notre *boite à outils* de composants génériques afin qu'ils puissent être utilisés tout au long de notre projet de développement matériel actuel, et être utilisés dans de futurs projets.

L'élément le plus important de notre *boite à outils* est notre filtre de convolution 2-D générique. En effet, il n'existe pas dans le "commerce" de module réalisant de façon simple un filtre à deux

dimensions. Xilinx fournit dans ses IP des filtres 1-D. Notre filtre 2-D, dont la taille est configurable, permet de réaliser différentes convolutions dans lesquelles les dimensions des coefficients et des pixels sont configurables. Ce filtre se décompose en deux sous-modules : un générateur de voisinage de pixel, et un processeur générique.

Pour conclure sur cette partie, nous avons réussi à implanter la méthode LASSR dans une cible FPGA et gérer au mieux le parallélisme pour atteindre les contraintes temps réel. Nous avons pu mettre en avant la possibilité de dupliquer les modules LASSR afin d'accélérer le traitement. Il nous a été possible d'intégrer 8 modules LASSR, ce qui permettrait théoriquement d'atteindre une vitesse de traitement de 136 im/s pour des images de 4k ( $4096 \times 2048$  pixels) en sortie.

Il manque cependant une réelle validation expérimentale sur une caméra complète intégrant différentes cibles FPGA, nous permettant de confronter nos résultats actuels avec une vérité de terrain.

Pour cela, il serait nécessaire de continuer les développements matériels. Premièrement, la gestion mémoire de la réception d'images LR et l'envoi des images HR pourrait être améliorée en utilisant des mécanismes de gestion de lecture/écriture mémoire en temps partagé. De tels travaux ont fait l'objet d'investigations au laboratoire dans le cadre de la conception d'une caméra HDR à multiple acquisitions.

Ensuite, il serait intéressant de décrire manuellement les composants utilisés dans ce projet, et plus particulièrement les FIFO. Décrire ces éléments manuellement nous permettrait de ne plus être dépendant de la cible matérielle pour pouvoir tester notre module sur d'autre famille de FPGA (Intel, Xilinx, Actel, ...).

Enfin, nous pourrions implanter le module LASSR dans la caméra intelligente HDR-ARtiSt [172] développée au laboratoire. Cette plateforme, développée initialement pour produire de la vidéo HDR en temps réel, possède toutes les caractéristiques matérielles pour intégrer le module LASSR et ainsi devenir une caméra intelligente dédiée à la Super Résolution en temps réel.

## 5.2/ ÉTUDE SUR LA PERTINENCE DES MÉTRIQUES

Travaillant sur l'amélioration de la qualité des images, il nous a fallu trouver de bons référentiels pour évaluer les méthodes que nous mettions en place. La mise en place de l'étude portant sur la pertinence des métriques a permis d'avoir une réflexion sur notre analyse, souvent personnelle, de la qualité d'image. En effet, que penser lorsque la métrique utilisée nous démontre que le résultat est médiocre, alors que visuellement celui-ci était, pour nous, très bon ?

Il existe un grand nombre de métriques pour l'évaluation de la qualité des images. Ces diverses métriques, avec ou sans référence, sont pour leurs auteurs les métriques les plus adaptées à quantifier la qualité des images. Cependant, notre étude a pu mettre en lumière le fait qu'avec un ensemble d'images donné, les résultats des métriques testées n'allaient pas forcément dans la même direction, or nous nous attendons à ce que toutes les métriques censées évaluer la qualité des images donnent des résultats évoluant de la même manière.

Le point commun pour la plupart des métriques est le fait qu'elles mesurent un paramètre de l'image précis. Les auteurs se basent sur des études mettant en avant le fait que l'œil humain est très sensible aux changements de contraste, et aux contours francs. Ces paramètres sont mesurés dans l'image et permettent alors de quantifier la qualité d'une image. Un excès de ces paramètres pourrait donner un avantage au score de l'image, mais rendre l'image visuellement de mauvaise qualité.

Ensuite, le PSNR est toujours une référence dans le traitement d'images pour juger de l'amélio-

ration de la qualité. Cependant cette mesure globale ne reflète pas la qualité d'une image, mais elle présente la distance entre cette image et l'image de référence. Cette métrique, décrite dans un grand nombre de publications scientifiques, n'est alors pas adaptée lorsque l'image testée est de meilleure qualité que l'image de référence. En effet, si l'image à comparer est meilleure que l'image de référence, cela met en avant qu'elles ne sont pas identiques.

Des nouvelles métriques se basent maintenant sur le deep learning [189, 183], cela dans le but d'apprendre ce qui est de "bonne qualité". La métrique BRISQUE [130] utilise un modèle pour réaliser son évaluation. Ce modèle est généré lors d'un apprentissage en utilisant la base d'images LIVE [129]. Cette base d'images contient d'une part les images ayant subis des distorsions standards (compression, flou gaussien, erreur de pixel, ...) et d'autre part le résultat d'évaluation subjective. Toutes ces informations permettent aux réseaux de neurones d'apprendre le type de distorsion et son évaluation. De cet apprentissage, le réseau de neurones pourra alors évaluer une image.

La plupart des métriques tentent de se rapprocher du jugement humain, des études psychovisuelles sont alors mises en place pour vérifier que les résultats des métriques créées se rapprochent au mieux de l'évaluation humaine. Cependant, suite à notre étude, nous pouvons voir qu'il est toujours nécessaire de réaliser une étude psychovisuelle pour valider réellement une méthode d'amélioration de la qualité des images sans pour autant oublier l'avantage des métriques dites "mathématiques" en terme de temps. En effet, les évaluations psychovisuelles sont très chronophage, et il est parfois compliqué de trouver un grand nombre de panélistes. Les métriques mathématiques peuvent être réalisées à temps perdu de façon automatique.

Les tests psychovisuels seraient les plus fiables, cependant il est important que les conditions d'évaluation des membres du panel soient les mêmes.

Il serait sans doute important de créer entièrement un panel en fonction de la destination du traitement. Si le traitement de SR est destiné à la diffusion le panel devra être constitué de "cibles". Par exemple une radiographie ne doit pas être améliorée de la même manière qu'une image de scanner, ou encore d'une image endoscopique.



FIGURE 5.1 – Photographie d'un test comparatif sur un écran de cinéma 4k aux Ateliers du Cinéma à Beaune. Deux images sont présentées : une image issue de notre traitement LASSR, et une image d'une résolution de 4k.

Des tests d'envergure pourraient être mis en place dans des salles de cinéma, cela permettrait de

composer des panels et de les réunir au même endroit pour faire des évaluations. Les salles de cinéma ont l'avantage d'avoir toujours les mêmes conditions d'éclairage. Des tests préliminaires ont pu être réalisés aux Ateliers du Cinéma à Beaune<sup>1</sup>. Ces tests ont permis de comparer différents traitements sur des images projetées avec un projecteur 4K. La figure 5.1 illustre une des comparaisons visuelles réalisée aux Ateliers du Cinéma.

### 5.3/ DÉCONVOLUTION TEMPS RÉEL

La contribution majeure et primordiale pour les membres du projet H2020-EXIST consistait en la conception et la mise en œuvre d'un démonstrateur temps réel réalisant une déconvolution. Ce démonstrateur doit gérer le flux d'images provenant de la caméra très haute résolution créée par nos partenaires. Au niveau du projet, l'ensemble des livrables ont été réalisés et ont contribué au succès de ce dernier. Ainsi, l'algorithme de déconvolution a pu être testé pour corriger en temps réel des images à grande résolution 2k et 4k. Le démonstrateur proposé permet ainsi de traiter en temps réel le flux d'images provenant de la caméra très haute résolution créée par nos partenaires à une fréquence de 60 im/s pour la caméra 2k. Deux implantations matérielles ont été proposées et réalisées : l'une basée sur un processeur standard et l'autre sur un processeur graphique (GPU). Cependant, seule l'implantation sur le processeur graphique (GPU) permet de respecter les contraintes temps-réel fixée par l'application.

D'un point de vue visuel, les résultats obtenus sont satisfaisants, en effet l'observation des contours et détails de l'image traitée apparaissent plus nets que ceux de l'image d'origine ; l'effet de flou diminue de manière très significative. De plus, en traitant les différents canaux séparément il est possible de voir que les aberrations chromatiques sont elles aussi diminuées.

Nous pouvons dire que notre équipe a réussi à mettre en place une chaîne complète de déconvolution. En effet, il a été possible de gérer le flux de la caméra depuis la carte d'acquisition à son traitement sur processeur standard ou processeur graphique. Ce développement a permis de fournir deux démonstrateurs fonctionnels au consortium du projet européen.

Puis, en terme de perspective sur la déconvolution, il serait intéressant de mettre en place une étude qualitative complète, avec plusieurs images vérité de terrain afin d'étudier plus en profondeur l'importance de certains paramètres lors de la correction des images. En effet, nous n'avions qu'un seul ensemble d'image (réelle (parfaite) et d'image dégradée). De ce fait, aucune étude qualitative n'était possible. Il aurait été intéressant de mesurer l'influence du nombre de PSF utilisées pour la correction d'image. Cette étude aurait permis de trouver un compromis entre le nombre de PSF utilisées, les ressources nécessaires, le temps d'exécution et la qualité de l'image corrigée. Cette étude manque réellement à ces travaux portant sur la déconvolution.

Enfin, il serait aussi intéressant de tester l'algorithme de déconvolution sur de nouvelles cibles GPU, en effet il y a eu de grandes évolutions sur le marché concernant la puissance des GPU. Il serait sans doute possible d'augmenter encore la fréquence de travail.

### 5.4/ VALORISATIONS SCIENTIFIQUES

1. [196] Marin, Y., Douiyek, A., Mitéran, J., Dubois, J., Heyrman, B. and Ginjac, D., **Image quality enhancement based on real-time deconvolution and super resolution**, In *Unconventional Optical Imaging (2018)*, vol. 10677, International Society for Optics and

---

1. <https://ateliersducinema.org/>

*Photonics*, p. 106772P.

2. [197] Marin, Y., Mitéran, J., Dubois, J., Heyrman, B. and Ginhac, D., **An fpga-based design for real-time super resolution**, In *Proceedings of the 12th International Conference on Distributed Smart Cameras (2018), ICDS'18, Association for Computing Machinery*.
3. [200] Marin, Y., Mitéran, J., Dubois, J., Heyrman, B. and Ginhac, D., **An fpga-based design for real-time super resolution**, In *Journal of Real-Time Image Processing (2020), Association for Computing Machinery*.

## BIBLIOGRAPHIE

- [1] HADAMARD, J. **Sur les problèmes aux dérivées partielles et leur signification physique.** *Princeton university bulletin* (1902), 49–52.
- [2] HAAR, A. **Zur theorie der orthogonalen funktionensysteme.** Georg-August-Universität, Göttingen., 1909.
- [3] DANIELL, P. **Lectures on cauchy’s problem in linear partial differential equations. by j. hadamard. pp. viii+ 316. 15s. net. 1923.(per oxford university press.).** *The Mathematical Gazette* 12, 171 (1924), 173–174.
- [4] HUFFMAN, D. A. **A method for the construction of minimum-redundancy codes.** *Proceedings of the IRE* 40, 9 (1952), 1098–1101.
- [5] MIDDLETON, W. E. K. **Vision through the Atmosphere.** Springer Berlin Heidelberg, Berlin, Heidelberg, 1957, pp. 254–287.
- [6] WIENER, N. **Extrapolation, Interpolation, and Smoothing of Stationary Time Series.** The MIT Press, 1964.
- [7] SCHREIBER, W. F. **Wirephoto quality improvement by unsharp masking.** *Pattern recognition* 2, 2 (1970), 117–121.
- [8] FLYNN, M. J. **Some computer organizations and their effectiveness.** *IEEE transactions on computers* 100, 9 (1972), 948–960.
- [9] RICHARDSON, W. H. **Bayesian-based iterative method of image restoration.** *Journal of the Optical Society of America*, pp. 62 :55–59.
- [10] LUCY, L. B. **An iterative technique for the rectification of observed distributions.** *Astron. J.* 79 (1974), 745–754.
- [11] MCCARTNEY, E. J. **Optics of the atmosphere : scattering by molecules and particles.** *New York, John Wiley and Sons, Inc., 1976. 421 p.* (1976).
- [12] DUCHON, C. E. **Lanczos filtering in one and two dimensions.** *Journal of applied meteorology* 18, 8 (1979), 1016–1022.
- [13] HARALICK, R. M. **Statistical and structural approaches to texture.** *Proceedings of the IEEE* 67, 5 (May 1979), 786–804.
- [14] LAWS, K. I. **Textured image segmentation.** PhD thesis, 1980.
- [15] LEE, J.-S. **Digital image enhancement and noise filtering by use of local statistics.** *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 2 (1980), 165–168.
- [16] LUCAS, B. D., KANADE, T., AND OTHERS. **An iterative image registration technique with an application to stereo vision.**
- [17] COOMANS, D., AND MASSART, D. L. **Alternative k-nearest neighbour rules in supervised pattern recognition : Part 1. k-nearest neighbour classification by using alternative voting rules.** *Analytica Chimica Acta* 136 (1982), 15–27.
- [18] MATSUYAMA, T., MIURA, S.-I., AND NAGAO, M. **Structural analysis of natural textures by fourier transformation.** *Computer vision, graphics, and image processing* 24, 3 (1983), 347–362.



- [19] GEMAN, S., AND GEMAN, D. **Stochastic relaxation, gibbs distributions, and the bayesian restoration of images.** *IEEE Transactions on pattern analysis and machine intelligence*, 6 (1984), 721–741.
- [20] TSAI, R. Y., AND HUANG, T. S. **Multiframe image restoration and registration.** In *Advances in Computer Vision and Image Processing*, R. Y. Tsai and T. S. Huang, Eds. JAI Press Inc., 1984, pp. 317–339.
- [21] GROSS, D. **Super-resolution from subpixel shifted pictures.** Tel Aviv University, 1986.
- [22] KEREN, D., PELEG, S., AND BRADA, R. **Image sequence enhancement using sub-pixel displacements.** In *Proceedings CVPR'88 : The Computer Society Conference on Computer Vision and Pattern Recognition* (1988), IEEE, pp. 742–746.
- [23] MEAD, C. **Analog vlsi and neural systems.** NASA STI/Recon Technical Report A 90 (1989).
- [24] ALLEN, R. J., AND WHITE, R. L. **The restoration of HST Images and Spectra.** Baltimore : Space Telescope Science Institute, 1990.
- [25] IRANI, M., AND PELEG, S. **Super resolution from image sequences.** In *[1990] Proceedings. 10th International Conference on Pattern Recognition* (1990), vol. 2, IEEE, pp. 115–120.
- [26] KAISER, J. F. **On a simple algorithm to calculate the 'energy' of a signal.** In *International conference on acoustics, speech, and signal processing* (1990), IEEE, pp. 381–384.
- [27] OSHER, S., AND RUDIN, L. I. **Feature-oriented image enhancement using shock filters.** *SIAM Journal on numerical analysis* 27, 4 (1990), 919–940.
- [28] TURKOWSKI, K. **Filters for common resampling tasks.** In *Graphics gems* (1990), Academic Press Professional, Inc., pp. 147–165.
- [29] CHALMOND, B. **Psf estimation for image deblurring.** *CVGIP : Graphical Models and Image Processing* 53, 4 (1991), 364–372.
- [30] IRANI, M., AND PELEG, S. **Improving resolution by image registration.** *CVGIP : Graphical models and image processing* 53, 3 (1991), 231–239.
- [31] WYATT, J. L., KEAST, C., SEIDEL, M., STANDLEY, D., HORN, B., KNIGHT, T., SODINI, C., LEE, H.-S., AND POGGIO, T. **Analog vlsi systems for image acquisition and fast early vision processing.** *International Journal of Computer Vision* 8, 3 (Sep 1992), 217–230.
- [32] KANADE, T., AND BAJCSY, R. **Computational sensors.** In *Report from the DARPA Workshop* (May 1993), T. Kanade and R. Bajcsy, Eds.
- [33] FISHER, Y. **Fractal image compression.** *Fractals* 2, 03 (1994), 347–361.
- [34] DOWSKI, E. R., AND CATHEY, W. T. **Extended depth of field through wave-front coding.** *Appl. Opt.* 34, 11 (Apr 1995), 1859–1866.
- [35] FREUND, Y. **Boosting a weak learning algorithm by majority.** *Information and computation* 121, 2 (1995), 256–285.
- [36] VAPNIK, V. **The nature of statistical learning theory.** Springer-Verlag, 1995.
- [37] FREUND, Y., SCHAPIRE, R. E., AND OTHERS. **Experiments with a new boosting algorithm.** In *icml* (1996), vol. 96, Citeseer, pp. 148–156.
- [38] KUNDUR, D., AND HATZINAKOS, D. **Blind image deconvolution.** *IEEE signal processing magazine* 13, 3 (1996), 43–64.
- [39] BORMAN, S., AND STEVENSON, R. **Super-resolution from image sequences-a review.** In *1998 Midwest Symposium on Circuits and Systems (Cat. No. 98CB36268)* (1998), pp. 374–378.
- [40] BORMAN, S., AND STEVENSON, R. L. **Super-resolution from image sequences-a review.** In *1998 Midwest Symposium on Circuits and Systems (Cat. No. 98CB36268)* (1998), IEEE, pp. 374–378.

- [41] GREWE, L. L., AND BROOKS, R. R. **Atmospheric attenuation reduction through multi-sensor fusion**. In *Sensor Fusion : Architectures, Algorithms, and Applications II* (1998), vol. 3376, International Society for Optics and Photonics, pp. 102–110.
- [42] HUBEL, D. H., AND WIESEL, T. N. **Early exploration of the visual cortex**. *Neuron* 20, 3 (1998), 401–412.
- [43] KOPEIKA, N. S. **A system engineering approach to imaging**. SPIE Optical Engineering Press Bellingham, 1998.
- [44] OAKLEY, J. P., AND SATHERLEY, B. L. **Improving image quality in poor visibility conditions using a physical model for contrast degradation**. *IEEE transactions on image processing* 7, 2 (1998), 167–179.
- [45] FREEMAN, W. T., AND PASZTOR, E. C. **Learning to estimate scenes from images**. In *Advances in Neural Information Processing Systems* (1999), pp. 775–781.
- [46] FREUND, Y., SCHAPIRE, R., AND ABE, N. **A short introduction to boosting**. *Journal-Japanese Society For Artificial Intelligence* 14, 771-780 (1999), 1612.
- [47] NAYAR, S. K., AND NARASIMHAN, S. G. **Vision in bad weather**. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on* (1999), vol. 2, IEEE, pp. 820–827.
- [48] SHERIF, S. S., GRIEBEL, S. K., AU, A. S. F., HUI, D. K. N., SZYMANSKI, T. H., AND HINTON, H. S. **Field-programmable smart-pixel arrays : design, vlsi implementation, and applications**. *Applied optics* 38 5 (1999), 838–46.
- [49] FREEMAN, W. T., PASZTOR, E. C., AND CARMICHAEL, O. T. **Learning low-level vision**. *International journal of computer vision* 40, 1 (2000), 25–47.
- [50] NARASIMHAN, S. G., AND NAYAR, S. K. **Chromatic framework for vision in bad weather**. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on* (2000), vol. 1, IEEE, pp. 598–605.
- [51] SAHA, S. **Image compression—from dct to wavelets : a review**. *XRDS : Crossroads, The ACM Magazine for Students* 6, 3 (2000), 12–21.
- [52] SCHÖLKOPF, B., SMOLA, A. J., WILLIAMSON, R. C., AND BARTLETT, P. L. **New support vector algorithms**. *Neural computation* 12, 5 (2000), 1207–1245.
- [53] SICURANZA, G. **Nonlinear image processing**. Elsevier, 2000.
- [54] TAN, K., AND OAKLEY, J. P. **Enhancement of color images in poor visibility conditions**. In *ICIP* (2000), vol. 2.
- [55] ZHANG, G. P. **Neural networks for classification : a survey**. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 30, 4 (2000), 451–462.
- [56] BREIMAN, L. **Random forests**. *Mach. Learn.* 45, 1 (Oct. 2001), 5–32.
- [57] NARASIMHAN, S. G., AND NAYAR, S. K. **Removing weather effects from monochrome images**. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (2001), vol. 2, IEEE, pp. II–II.
- [58] SCHECHNER, Y. Y., NARASIMHAN, S. G., AND NAYAR, S. K. **Instant dehazing of images using polarization**. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* (Dec 2001), vol. 1, pp. I–I.
- [59] BARLAUD, M. **Compression et codage des images et des vidéos**. Hermès, 2002.
- [60] FREEMAN, W., JONES, T., AND PASZTOR, E. **Example-based super-resolution**. *IEEE Computer Graphics and Applications* 22 (2002), 56–65.

- [61] FREEMAN, W. T., JONES, T. R., AND PASZTOR, E. C. **Example-based super-resolution**. *IEEE Computer graphics and Applications*, 2 (2002), 56–65.
- [62] MARZILIANO, P., DUFAUX, F., WINKLER, S., AND EBRAHIMI, T. **A no-reference perceptual blur metric**. In *Proceedings. International Conference on Image Processing* (2002), vol. 3, IEEE, pp. III–III.
- [63] WANG, Z., AND BOVIK, A. C. **A universal image quality index**. *IEEE Signal Processing Letters* 9, 3 (March 2002), 81–84.
- [64] ARIVAZHAGAN, S., AND GANESAN, L. **Texture classification using wavelet transform**. *Pattern recognition letters* 24, 9-10 (2003), 1513–1521.
- [65] DUBOIS, J., AND MATTAVELLI, M. **Embedded co-processor architecture for cmos based image acquisition**. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on* (2003), vol. 2, IEEE, pp. II–591.
- [66] PARK, S. C., PARK, M. K., AND KANG, M. G. **Super-resolution image reconstruction : a technical overview**. *IEEE Signal Processing Magazine* 20 (2003), 21–36.
- [67] FARSIU, S., ROBINSON, M. D., ELAD, M., AND MILANFAR, P. **Fast and robust multiframe super resolution**. *IEEE Transactions on Image Processing* 13, 10 (Oct 2004), 1327–1344.
- [68] RICHARDSON, I. E. **H. 264 and MPEG-4 video compression : video coding for next-generation multimedia**. John Wiley & Sons, 2004.
- [69] SALOMON, D. **Data compression : the complete reference**. Springer Science & Business Media, 2004.
- [70] SOCHEN, N. **Affine invariant flows in the beltrami framework**. *Journal of Mathematical Imaging and Vision* 20, 1-2 (2004), 133–146.
- [71] WANG, Z., BOVIK, A. C. AND SHEIKH, H. R., AND SIMONCELLI, E. P. **Image quality assessment : from error visibility to structural similarity**. *IEEE Transactions on Image Processing* 13(4) (2004), 600–612.
- [72] BARRETO, D., ALVAREZ, L., AND ABAD, J. **Motion estimation techniques in super-resolution image reconstruction : a performance evaluation**. In *Proceedings of the International Workshop Virtual Observatory : Plate Content Digitalization, Archive Mining and Image Sequence Processing* (2005), pp. 254–268.
- [73] BARRETO, D., CALICO, G., LOPEZ, S., GARCÍA, L., AND NUNEZ, A. **Real-time super-resolution over raw video sequences**. In *VLSI Circuits and Systems II* (2005), vol. 5837, International Society for Optics and Photonics, pp. 628–638.
- [74] BOVIK, A. C. **A visual information fidelity approach to video quality assessment**.
- [75] MANCAS-THILLOU, C., AND MIRMEHDI, M. **Super-resolution text using the teager filter**. In *First International Workshop on Camera-Based Document Analysis and Recognition* (2005), Citeseer, pp. 10–16.
- [76] MITÉRAN, J., MATAS, J., BOURENNANE, E., PAINDAVOINE, M., AND DUBOIS, J. **Automatic hardware implementation tool for a discrete adaboost-based decision algorithm**. *EURASIP Journal on Applied Signal Processing* 2005 (2005), 1035–1046.
- [77] SHEIKH, H. **Live image quality assessment database release 2**. <http://live.ece.utexas.edu/research/quality> (2005).
- [78] YANG, M., AND BOURBAKIS, N. **An overview of lossless digital image compression techniques**. In *48th Midwest Symposium on Circuits and Systems, 2005.* (2005), IEEE, pp. 1099–1102.

- [79] DOHNAL, M. **The hs chromaticity diagram and the lmn colour space**. In *Photonics, Devices, and Systems III* (2006), vol. 6180, International Society for Optics and Photonics, p. 61801W.
- [80] OUWERKERK, J. v. **Image super-resolution survey**. *Image and Vision Computing* 24 (2006), 1039–1052.
- [81] RASKAR, R., AGRAWAL, A., AND TUMBLIN, J. **Coded exposure photography : motion de-blurring using fluttered shutter**. In *ACM transactions on graphics (TOG)* (2006), vol. 25, ACM, pp. 795–804.
- [82] TAKEDA, H., FARSIU, S., MILANFAR, P., AND OTHERS. **Kernel regression for image processing and reconstruction**. PhD thesis, Citeseer, 2006.
- [83] GOW, R. D., RENSHAW, D., FINDLATER, K., GRANT, L., MCLEOD, S. J., HART, J., AND NICOL, R. L. **A comprehensive tool for modeling cmos image-sensor-noise performance**. *IEEE Transactions on Electron Devices* 54, 6 (June 2007), 1321–1329.
- [84] LIU, C., SHUM, H.-Y., AND FREEMAN, W. T. **Face hallucination : Theory and practice**. *International Journal of Computer Vision* 75, 1 (2007), 115–134.
- [85] MANCAS-THILLOU, C., AND MIRMEHDI, M. **An introduction to super-resolution text**. In *Digital document processing*. Springer, 2007, pp. 305–327.
- [86] SIMS, O. **Efficient implementation of video processing algorithms on FPGA**. PhD thesis, University of Glasgow, 2007.
- [87] VONESCH, C., BLU, T., AND UNSER, M. **Generalized daubechies wavelet families**. *IEEE Transactions on Signal Processing* 55, 9 (2007), 4415–4429.
- [88] ANGELOPOULOU, M. E., BOUGANIS, C.-S., CHEUNG, P. Y. K., AND CONSTANTINIDES, G. A. **FPGA-Based Real-Time Super-Resolution on an Adaptive Image Sensor**. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 125–136.
- [89] BOWEN, O., AND BOUGANIS, C.-S. **Real-time image super resolution using an FPGA**. In *2008 International Conference on Field Programmable Logic and Applications* (2008), pp. 89–94.
- [90] CALICO, G. M., LÓPEZ, S., SOSA, O., LOPEZ, J. F., AND SARMIENTO, R. **Analysis of fast block matching motion estimation algorithms for video super-resolution systems**. *IEEE Transactions on consumer electronics* 54, 3 (2008), 1430–1438.
- [91] CHEN, M., FRIDRICH, J., GOLJAN, M., AND LUKÁS, J. **Determining image origin and integrity using sensor noise**. *IEEE Transactions on information forensics and security* 3, 1 (2008), 74–90.
- [92] DEBEVEC, P. E., AND MALIK, J. **Recovering high dynamic range radiance maps from photographs**. In *ACM SIGGRAPH 2008 Classes* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 31 :1–31 :10.
- [93] GONZALES, R. C., AND WOODS, R. E. **Digital image processing third edition**, 2008.
- [94] JOSHI, N., SZELISKI, R., AND KRIEGMAN, D. J. **Psf estimation using sharp edge prediction**. In *2008 IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), pp. 1–8.
- [95] LIANG, C.-K., LIN, T.-H., WONG, B.-Y., LIU, C., AND CHEN, H. H. **Programmable aperture photography : multiplexed light field acquisition**. In *ACM SIGGRAPH 2008 papers*. 2008, pp. 1–10.
- [96] MIKALSEN, E. B., HARDEBERG, J. Y., AND THOMAS, J. **Verification and extension of a camera-based end-user calibration method for projection displays**. In *4th European Conference*

- on Colour in Graphics, Imaging, and Vision and 10th International Symposium on Multispectral Colour Science, CGIV 2008/MCS'08, Terrassa, Spain, June 9-13, 2008* (2008), IS&T - The Society for Imaging Science and Technology, pp. 575–579.
- [97] YUAN, L., SUN, J., QUAN, L., AND SHUM, H.-Y. **Progressive inter-scale and intra-scale non-blind image deconvolution**. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 74 :1–74 :10.
- [98] ZHANG, C., AND VIOLA, P. A. **Multiple-instance pruning for learning efficient cascade detectors**. In *Advances in neural information processing systems* (2008), pp. 1681–1688.
- [99] ASMARE, M. H., ASIRVADAM, V. S., AND IZNITA, L. **Color space selection for color image enhancement applications**. In *2009 International Conference on Signal Acquisition and Processing* (2009), IEEE, pp. 208–212.
- [100] FERZLI, R., AND KARAM, L. J. **A no-reference objective image sharpness metric based on the notion of just noticeable blur (jnb)**. *IEEE transactions on image processing* 18, 4 (2009), 717–728.
- [101] FRIDRICH, J. **Digital image forensics**. *IEEE Signal Processing Magazine* 26, 2 (2009), 26–37.
- [102] KHATTAB, K., DUBOIS, J., AND MITERAN, J. **Cascade boosting-based object detection from high-level description to hardware implementation**. *EURASIP Journal on Embedded Systems 2009*, 1 (2009), 235032.
- [103] MOORTHY, A. K., AND BOVIK, A. C. **Visual importance pooling for image quality assessment**. *IEEE journal of selected topics in signal processing* 3, 2 (2009), 193–201.
- [104] ROSENFELD, K., AND SENCAR, H. T. **A study of the robustness of prnu-based camera identification**. In *Media Forensics and Security* (2009), vol. 7254, International Society for Optics and Photonics, p. 72540M.
- [105] WANG, Z., AND BOVIK, A. C. **Mean squared error : Love it or leave it? a new look at signal fidelity measures**. *IEEE signal processing magazine* 26, 1 (2009), 98–117.
- [106] ZHOU, C., AND NAYAR, S. **What are good apertures for defocus deblurring?** In *2009 IEEE international conference on computational photography (ICCP)* (2009), IEEE, pp. 1–8.
- [107] ANBARJAFARI, G., AND DEMIREL, H. **Image super resolution based on interpolation of wavelet domain high frequency subbands and the spatial domain input image**. *ETRI journal* 32, 3 (2010), 390–394.
- [108] BELBACHIR, A. N. **Smart cameras**, vol. 2. Springer, 2010.
- [109] DEMIREL, H., AND ANBARJAFARI, G. **Image resolution enhancement by using discrete and stationary wavelet decomposition**. *IEEE transactions on image processing* 20, 5 (2010), 1458–1460.
- [110] GERHARDT, J., AND THOMAS, J. **Toward an automatic color calibration for 3d displays**. In *18th Color and Imaging Conference, CIC 2010, San Antonio, Texas, USA, November 8-12, 2010* (2010), IS&T - The Society for Imaging Science and Technology, pp. 5–10.
- [111] NARVEKAR, N., AND KARAM, L. J. **An improved no-reference sharpness metric based on the probability of blur detection**. In *Workshop on Video Processing and Quality Metrics* (2010).
- [112] NGUYEN, H.-T., NGUYEN, V.-D., AND LE, H. V. **Super-resolution image construction from high-speed camera sequences**. In *Advanced Technologies for Communications (ATC), 2010 International Conference on* (2010), IEEE, pp. 152–157.
- [113] ROBINSON, M. D., CHIU, S. J., TOTH, C. A., IZATT, J. A., LO, J. Y., AND FARSIU, S. **New Applications of Super-resolution in Medical Imaging**. In *Super-resolution imaging*, P. Milanfar, Ed. CRC Press, 2010, pp. 383–412.



- [114] YANG, J., AND HUANG, T. **Image Super-Resolution : Historical Overview and Future Challenges**. In *Super-resolution imaging*, P. Milanfar, Ed. CRC Press, 2010, pp. 1–33.
- [115] YANG, J., WRIGHT, J., HUANG, T. S., AND MA, Y. **Image super-resolution via sparse representation**. *IEEE transactions on image processing* 19, 11 (2010), 2861–2873.
- [116] ZHANG, L., ZHANG, H., SHEN, H., AND LI, P. **A super-resolution reconstruction algorithm for surveillance images**. *Signal Processing* 90 (2010), 848–859.
- [117] KRONSTEIN, D. **Video stream scaler : [www.opencores.org](http://www.opencores.org)**, 2011.
- [118] NARVEKAR, N. D., AND KARAM, L. J. **A no-reference image blur metric based on the cumulative probability of blur detection (cpbd)**. *IEEE Transactions on Image Processing* 20, 9 (2011), 2678–2683.
- [119] NASONOV, A. V., AND KRYLOV, A. **Image enhancement quality metrics**.
- [120] NAYAR, S. **Computational cameras : approaches, benefits and limits**.
- [121] PATEL, V., MODI, C. K., PAUNWALA, C. N., AND PATNAIK, S. **Hybrid approach for single image super resolution using isef and ibp**. In *2011 International Conference on Communication Systems and Network Technologies* (2011), IEEE, pp. 495–499.
- [122] SU, B., LU, S., AND TAN, C. L. **Blurred image region detection and classification**. In *Proceedings of the 19th ACM International Conference on Multimedia* (New York, NY, USA, 2011), MM '11, ACM, pp. 1397–1400.
- [123] SZYDZIK, T., CALLICO, G. M., AND NUNEZ, A. **Efficient fpga implementation of a high-quality super-resolution algorithm with real-time performance**. *IEEE Transactions on Consumer Electronics* 57, 2 (May 2011), 664–672.
- [124] YERVA, S., NAIR, S., AND KUTTY, K. **Lossless image compression based on data folding**,|| [ieeep.org](http://ieeep.org).
- [125] ZHANG, L., ZHANG, L., MOU, X., AND ZHANG, D. **Fsim : A feature similarity index for image quality assessment**. *IEEE transactions on Image Processing* 20, 8 (2011), 2378–2386.
- [126] BAREJA, M. N., AND MODI, C. K. **An effective iterative back projection based single image super resolution approach**. In *2012 international conference on communication systems and Network technologies* (2012), IEEE, pp. 95–99.
- [127] GOHSHI, S., AND ECHIZEN, I. **Video image display screen, video image display system, and method for detecting camera used in illegal camcording**, Dec. 13 2012. US Patent App. 13/580,898.
- [128] GOHSHI, S., SEKIGUCHI, H., SHIMIZU, Y., AND IKENAGA, T. **Subjective assessment about super resolution image resolution**. *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering* 6, 9 (2012), 1172–1177.
- [129] JAYARAMAN, D., MITTAL, A., MOORTHY, A., AND BOVIK, A. **Live multiply distorted image quality database**. online at : [http://live.ece.utexas.edu/research/quality/live\\_multidistortedimage.html](http://live.ece.utexas.edu/research/quality/live_multidistortedimage.html) (2012).
- [130] MITTAL, A., MOORTHY, A. K., AND BOVIK, A. C. **No-reference image quality assessment in the spatial domain**. *IEEE Transactions on Image Processing* 21, 12 (2012), 4695–4708.
- [131] THOMAS, J., AND GERHARDT, J. **Webcam based display calibration**. In *20th Color and Imaging Conference, CIC 2012, Los Angeles, California, USA, November 12-16, 2012* (2012), IS&T - The Society for Imaging Science and Technology, pp. 82–87.
- [132] WU, J.-L., CHANG, C.-F., AND CHEN, C.-S. **An improved richardson-lucy algorithm for single image deblurring using local extrema filtering**. In *2012 International Symposium on Intelligent Signal Processing and Communications Systems* (2012), IEEE, pp. 27–32.



- [133] BALAMURUGAN, E., SENGOTTUVELAN, P., AND SANGEETHA, K. **An empirical evaluation of salt and pepper noise removal for document images using median filter.** *International Journal of Computer Applications* 82, 4 (2013).
- [134] CHARFI, I. **Détection automatique de chutes de personnes basée sur des descripteurs spatio-temporels : définition de la méthode, évaluation des performances et implantation temps-réel.** PhD thesis, Université de Bourgogne, 2013.
- [135] COSSAIRT, O., GUPTA, M., AND NAYAR, S. K. **When does computational imaging improve performance?** *IEEE Transactions on Image Processing* 22, 2 (Feb 2013), 447–458.
- [136] FEICHTENHOFER, C., FASSOLD, H., AND SCHALLAUER, P. **A perceptual image sharpness metric based on local edge gradient analysis.** *IEEE Signal Processing Letters* 20(4) (2013), 379–382.
- [137] GOHSHI, S. **A New Signal Processing Method for Video Image-Reproduce the Frequency Spectrum Exceeding the Nyquist Frequency Using a Single Frame of the Video Image.** Springer New York, New York, NY, 2013, pp. 593–604.
- [138] GOHSHI, S., AND ECHIZEN, I. **Limitations of super resolution image reconstruction and how to overcome them for a single image.** In *2013 International Conference on Signal Processing and Multimedia Applications (SIGMAP)* (July 2013), pp. 71–78.
- [139] GOHSHI, S., NAKAMURA, S., AND OGASAWARA, J. **Real-time up-converter from hdtv to 4k with super resolution -breaking the nyquist frequency barrier-.** 585–585.
- [140] MAKWANA, R. R., AND MEHTA, N. D. **Single image super-resolution via iterative back projection based canny edge detection and a gabor filter prior.** *International Journal of Soft Computing & Engineering* 3, 1 (2013), 379–384.
- [141] MITTAL, A., SOUNDARARAJAN, R., AND BOVIK, A. C. **Making a “completely blind” image quality analyzer.** *IEEE Signal Processing Letters* 20, 3 (March 2013), 209–212.
- [142] SRIKANTH, S., AND MEHER, S. **Compression efficiency for combining different embedded image compression techniques with huffman encoding.** In *2013 International Conference on Communication and Signal Processing* (2013), IEEE, pp. 816–820.
- [143] TIMOFTE, R., DE SMET, V., AND VAN GOOL, L. **Anchored neighborhood regression for fast example-based super-resolution.** In *Proceedings of the IEEE international conference on computer vision* (2013), pp. 1920–1927.
- [144] VAPNIK, V. **The nature of statistical learning theory.** Springer science & business media, 2013.
- [145] DONG, C., LOY, C. C., HE, K., AND TANG, X. **Learning a deep convolutional network for image super-resolution.** In *Computer Vision – ECCV 2014, 13th European Conference, Zurich, Switzerland* (2014), pp. 184–199.
- [146] DONG, C., LOY, C. C., HE, K., AND TANG, X. **Learning a deep convolutional network for image super-resolution.** In *European conference on computer vision* (2014), Springer, pp. 184–199.
- [147] GOHSHI, S. **4k-to-8k tv up-converter with super resolution.** In *SMPTE 2014 Annual Technical Conference Exhibition* (Oct 2014), pp. 1–6.
- [148] GOHSHI, S., HIROI, T., AND ECHIZEN, I. **Subjective assessment of hdtv with superresolution function.** *EURASIP Journal on Image and Video Processing* 2014, 1 (2014), 11.
- [149] GUPTA, A. **Design and analysis of image restoration algorithm using radon transform and bilateral filter.** *International Journal of Engineering and Management Research (IJEMR)* 4, 2 (2014), 212–216.

- [150] LAPRAY, P.-J., HEYRMAN, B., AND GINHAC, D. **Hdr-artist : an adaptive real-time smart camera for high dynamic range imaging**. *Journal of Real-Time Image Processing* (2014), 1–16.
- [151] LAWGALY, A., KHELIFI, F., AND BOURIDANE, A. **Weighted averaging-based sensor pattern noise estimation for source camera identification**. In *2014 IEEE International Conference on Image Processing (ICIP)* (2014), IEEE, pp. 5357–5361.
- [152] NASROLLAHI, K., AND MOESLUND, T. B. **Super-resolution : a comprehensive survey**. *Machine vision and applications* 25, 6 (2014), 1423–1468.
- [153] PÉREZ, J., MAGDALENO, E., PÉREZ, F., RODRÍGUEZ, M., HERNÁNDEZ, D., AND CORRALES, J. **Super-Resolution in Plenoptic Cameras Using FPGAs**. *Sensors* 14 (2014), 8669–8685.
- [154] RASTI, P., LÜSI, I., DEMIREL, H., KIEFER, R., AND ANBARJAFARI, G. **Wavelet transform based new interpolation technique for satellite image resolution enhancement**. In *2014 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology* (Nov 2014), pp. 185–188.
- [155] SAFFINAZ, S. **An efficient algorithm for image scaling with high boost filtering**. *International Journal of Scientific and Research Publications* 4, 5 (2014), 1–5.
- [156] ZHANG, L., SHEN, Y., AND LI, H. **Vsi : A visual saliency-induced index for perceptual image quality assessment**. *IEEE Transactions on Image Processing* 23, 10 (Oct 2014), 4270–4281.
- [157] MORI, C., SUGIE, M., TAKESHITA, H., AND GOHSHI, S. **Subjective assessment of super-resolution**. In *2015 10th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)* (2015), IEEE, pp. 1–3.
- [158] PEREZ-PELLITERO, E., SALVADOR, J., RUIZ-HIDALGO, J., AND ROSENHAHN, B. **Accelerating super-resolution for 4k upscaling**. In *2015 IEEE International Conference on Consumer Electronics (ICCE)* (2015), IEEE, pp. 317–320.
- [159] SCHULTER, S., LEISTNER, C., AND BISCHOF, H. **Fast and accurate image upscaling with super-resolution forests**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3791–3799.
- [160] SEYID, K., BLANC, S., AND LEBLEBICI, Y. **Hardware implementation of real-time multiple frame super-resolution**. In *2015 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)* (Daejeon, South Korea, 2015), pp. 219–224.
- [161] VAN NGO, K., DOKKEBERG, C. A., FARUP, I., PEDERSEN, M., AND OTHERS. **Quickeval : a web application for psychometric scaling experiments**. In *Image Quality and System Performance XII* (2015), vol. 9396, International Society for Optics and Photonics, p. 93960O.
- [162] YANG, D., LI, Z., XIA, Y., AND CHEN, Z. **Remote Sensing Image Super-Resolution : Challenges and Approaches**. In *2015 IEEE International Conference on Digital Signal Processing (DSP)* (2015), pp. 196–200.
- [163] ANDREARCZYK, V., AND WHELAN, P. F. **Using filter banks in convolutional neural networks for texture classification**. *Pattern Recognition Letters* 84 (2016), 63–69.
- [164] BASU, S., KARKI, M., MUKHOPADHYAY, S., GANGULY, S., NEMANI, R., DIBIANO, R., AND GAYAKA, S. **A theoretical analysis of deep neural networks for texture classification**. In *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), IEEE, pp. 992–999.
- [165] BOURRASSET, C. **Le modèle flot de données appliqué à la synthèse haut-niveau pour le traitement d’images sur caméra intelligente à base de FPGA. Application aux systèmes d’apprentissage supervisés**. PhD thesis, Université Blaise Pascal-Clermont-Ferrand II, 2016.

- [166] CHOI, J., AND KIM, M. **Super-interpolation with edge-orientation-based mapping kernels for low complex2×upscaling**. *IEEE Transactions on Image Processing* 25, 1 (Jan 2016), 469–483.
- [167] GOHSHI, S., OGASAWARA, J., AND NAKAMURA, S. **Image processing apparatus and image processing method**, June 21 2016. US Patent 9,374,508.
- [168] KIM, J., KWON LEE, J., AND MU LEE, K. **Deeply-recursive convolutional network for image super-resolution**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 1637–1645.
- [169] KIM, J., LEE, J. K., AND LEE, K. M. **Accurate Image Super-Resolution Using Very Deep Convolutional Networks**. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 1646–1654.
- [170] KŁOSOWSKI, M., JENDERNALIK, W., JAKUSZ, J., BLAKIEWICZ, G., AND SZCZEPAŃSKI, S. **A cmos pixel with embedded adc, digital cds and gain correction capability for massively parallel imaging array**. *IEEE Transactions on Circuits and Systems I : Regular Papers* 64, 1 (2016), 38–49.
- [171] LAPRAY, P.-J., HEYRMAN, B., AND GINHAC, D. **Hdr-artist : an adaptive real-time smart camera for high dynamic range imaging**. *Journal of Real-Time Image Processing* 12, 4 (Dec 2016), 747–762.
- [172] LAPRAY, P.-J., HEYRMAN, B., AND GINHAC, D. **Hdr-artist : an adaptive real-time smart camera for high dynamic range imaging**. *Journal of Real-Time Image Processing* 12, 4 (2016), 747–762.
- [173] LEE, S., YUN, S., NAM, J.-H., WON, C. S., AND JUNG, S.-W. **A review on dark channel prior based image dehazing algorithms**. *EURASIP Journal on Image and Video Processing* 2016, 1 (Jan 2016), 4.
- [174] MANABE, T., SHIBATA, Y., AND OGURI, K. **FPGA implementation of a real-time super-resolution system using a convolutional neural network**. In *2016 International Conference on Field-Programmable Technology (FPT)* (Xi'an, 2016), pp. 249–252.
- [175] PARSANIA, P., AND PARESH V. VIRPARIA, D. **A comparative analysis of image interpolation algorithms**. *IJARCCCE* 5 (01 2016), 29–34.
- [176] REDLICH, R., ARANEDA, L., SAAVEDRA, A., AND FIGUEROA, M. **An Embedded Hardware Architecture for Real-Time Super-Resolution in Infrared Cameras**. In *2016 Euromicro Conference on Digital System Design (DSD)* (2016), pp. 184–191.
- [177] ROY, A., AND SAIKIA, L. P. **A comparative study on lossy image compression techniques**. *Int. J. Curr. Trends Eng. Res.(IJCTER)* 2, 6 (2016), 16–25.
- [178] SEYID, K. **Hardware Implementation and Applications of High Performance Multiple Camera Architectures**. PhD thesis, 2016.
- [179] SHI, W., CABALLERO, J., HUSZÁR, F., TOTZ, J., AITKEN, A. P., BISHOP, R., RUECKERT, D., AND WANG, Z. **Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 1874–1883.
- [180] YUE, L., SHEN, H., LI, J., YUAN, Q., ZHANG, H., AND ZHANG, L. **Image super-resolution : The techniques, applications, and future**. *Signal Processing* 128 (2016), 389–408.
- [181] BERMAN, D., TREIBITZ, T., AND AVIDAN, S. **Air-light estimation using haze-lines**. In *2017 IEEE International Conference on Computational Photography (ICCP)* (2017), IEEE, pp. 1–9.

- [182] BOSCARO, A., JACQUIR, S., SANCHEZ, K., PERDU, P., AND BINCZAK, S. **Pattern image enhancement by automatic focus correction.** *Microelectronics Reliability* (07 2017).
- [183] BOSSE, S., MANIRY, D., MÜLLER, K.-R., WIEGAND, T., AND SAMEK, W. **Deep neural networks for no-reference and full-reference image quality assessment.** *IEEE Transactions on Image Processing* 27, 1 (2017), 206–219.
- [184] DARSON, D., DUBOIS, J., BOURDERNANE, M., HEYRMAN, B., MORFIN, P., DOUIYEK, A., AND GINHAC, D. **Real-time high dynamic range based on multiple non destructive readout during a single exposure : Application to ir imaging.** In *Proceedings of the 11th International Conference on Distributed Smart Cameras* (New York, NY, USA, 2017), ICDS 2017, ACM, pp. 21–28.
- [185] DIAZ ZAMBONI, J. E., AND CASCO, V. H. **Estimation methods of the point spread function axial position : A comparative computational study.** *Journal of Imaging* 3, 1 (2017), 7.
- [186] LIM, B., SON, S., KIM, H., NAH, S., AND LEE, K. M. **Enhanced Deep Residual Networks for Single Image Super-Resolution.** In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2017), pp. 1132–1140.
- [187] SAYOOD, K. **Introduction to data compression.** Morgan Kaufmann, 2017.
- [188] BASU, S., MUKHOPADHYAY, S., KARKI, M., DiBIANO, R., GANGULY, S., NEMANI, R., AND GAYAKA, S. **Deep neural networks for texture classification—a theoretical analysis.** *Neural Networks* 97 (2018), 173–182.
- [189] BIANCO, S., CELONA, L., NAPOLETANO, P., AND SCETTINI, R. **On the use of deep learning for blind image quality assessment.** *Signal, Image and Video Processing* 12, 2 (2018), 355–362.
- [190] CHIRANJEEVI, K., AND JENA, U. R. **Image compression based on vector quantization using cuckoo search optimization technique.** *Ain Shams Engineering Journal* 9, 4 (2018), 1417–1431.
- [191] GONSALVES, R. A. **Phase diversity : math, methods and prospects, including sequential diversity imaging.** In *Unconventional Optical Imaging* (2018), vol. 10677, International Society for Optics and Photonics, p. 106771S.
- [192] HE, Z., HUANG, H., JIANG, M., BAI, Y., AND LUO, G. **Fpga-based real-time super-resolution system for ultra high definition videos.** In *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)* (2018), IEEE, pp. 181–188.
- [193] KIM, Y., CHOI, J., AND KIM, M. **2x super-resolution hardware using edge-orientation-based linear mapping for real-time 4k uhd 60 fps video applications.** *IEEE Transactions on Circuits and Systems II : Express Briefs* 65, 9 (Sep. 2018), 1274–1278.
- [194] KIM, Y., CHOI, J.-S., AND KIM, M. **A Real-Time Convolutional Neural Network for Super-Resolution on FPGA with Applications to 4K UHD 60 fps Video Services.** *IEEE Transactions on Circuits and Systems for Video Technology PP* (2018), 1–1.
- [195] MARHABA, B. **Satellite image restoration by nonlinear statistical filtering techniques.** PhD thesis, 2018.
- [196] MARIN, Y., DOUIYEK, A., MITÉRAN, J., DUBOIS, J., HEYRMAN, B., AND GINHAC, D. **Image quality enhancement based on real-time deconvolution and super resolution.** In *Unconventional Optical Imaging* (2018), vol. 10677, International Society for Optics and Photonics, p. 106772P.

- [197] MARIN, Y., MITÉLAN, J., DUBOIS, J., HEYRMAN, B., AND GINHAC, D. **An fpga-based design for real-time super resolution reconstruction.** In *Proceedings of the 12th International Conference on Distributed Smart Cameras* (New York, NY, USA, 2018), ICDSC '18, Association for Computing Machinery.
- [198] SHOCHER, A., COHEN, N., AND IRANI, M. **Zero-Shot Super-Resolution Using Deep Internal Learning.** In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 3118–3126.
- [199] WANG, Z., CHEN, J., AND HOI, S. C. H. **Deep Learning for Image Super-resolution : A Survey.** *arXiv* (2019).
- [200] MARIN, Y., MITÉLAN, J., DUBOIS, J., HEYRMAN, B., AND GINHAC, D. **An fpga-based design for real-time super-resolution reconstruction.** *Journal of Real-Time Image Processing* (2020), 1–17.
- [201] MATHUR, G., MATHUR, R., AND MATHUR, M. K. **A comparative study of various lossy image compression techniques.** *ETRASCT 14*, 165–169.

## TABLE DES FIGURES

1.1	NHK prévoit la diffusion des JO 2020 en 8k, @240im/s. Illustration présentant l'utilisation du système de ralenti 8k et du caméscope 8k pour la production de programmes télévisuels. . . . .	3
2.1	Acquisition d'image présentée dans la thèse de Hiroyuki Takeda [82] . . . . .	8
2.2	Pertes d'informations et dégradations se produisant de la prise de vue à la visualisation de l'image . . . . .	9
2.3	Perturbations atmosphériques modifiant la formation d'une image présentée dans les travaux de Lee et al. [173] . . . . .	10
2.4	Une photographie contenant un flou de bougé . . . . .	11
2.5	Problème de mise au point, le point est fait sur le premier plan ( <i>gauche</i> ), le point est fait sur le fond ( <i>droite</i> ) . . . . .	11
2.6	Dégradations de l'image présentées par [67] où nous avons différencié les dégradations optiques et électroniques . . . . .	12
2.7	Exemple d'assombrissement périphérique dû à l'objectif : le vignettage. . . . .	13
2.8	Principe des aberrations chromatiques, décalage des différents canaux dû à la réfraction de la lentille. . . . .	13
2.9	Trois principales déformations géométriques dues à l'objectif : moustache, coussinet, barillet ( <i>de gauche à droite</i> ) . . . . .	14
2.10	Représentation des origines des principaux bruits électroniques pour un capteur de technologie CMOS présentés dans les travaux de Gow et al. [83] . . . . .	15
2.11	Principe de suppression du FPN, L'image avec FPN à gauche, l'image "noire" de FPN au centre, et enfin l'image corrigée à droite . . . . .	16
2.12	Images capturées par une caméra IR InGaAs mettant en avant l'évolution du bruit thermique [184] en fonction d'un refroidissement du capteur de 300K à 77K avec un temps d'exposition de 2 secondes. . . . .	17
2.13	Exemple de bruit "poivre et sel" ( <i>centre</i> ) sur l'image source ( <i>gauche</i> ) et corrigé avec un filtre médian 2d ( <i>droite</i> ) effectué sur un voisinage de $3 \times 3$ pixels. . . . .	18
2.14	Exemples de pertes d'information à la suite de compression avec pertes JPEG-2000. Image non compressée ( <i>gauche</i> ) et la version de l'image compressée avec un taux de compression de 90% ( <i>droite</i> ) . . . . .	19
2.15	Limite de EQM présentée dans l'article de Wang et Bovik [105], les trois images dégradées ont la même mesure de EQM=309, alors qu'elles présentent des caractéristiques visuelles très différentes. . . . .	22



2.16	Principe de mesure de VIF [74] . . . . .	23
2.17	Représentation du voisinage et des associations de pixels pour le calcul des attributs nécessaires à l'apprentissage de BRISQUE . . . . .	27
2.18	Les 4 subdivisions de l'imagerie computationnelle selon [120], ces subdivisions se recoupent car elles englobent parfois les mêmes méthodes. . . . .	31
2.19	Différence entre Imagerie Computationnelle ( <i>gauche</i> ) et Imagerie standard ( <i>droite</i> ) pour avoir le même résultat concernant la réduction de flou de défocalisation et de bougé. Les images proviennent respectivement de [106, 81]. . . . .	33
2.20	Exemple de convolution entre une source lumineuse ( <i>gauche</i> ) et la <i>PSF</i> ( <i>centre</i> ), l'image résultante est floue ( <i>droite</i> ) . . . . .	33
2.21	Exemple d'une image floue prise par Hubble après son lancement ( <i>gauche</i> ), et une image prise après la mission de réparation de son miroir STS-61 ( <i>droite</i> ) . . . . .	34
2.22	Technique du filtrage inverse directe présentée dans [149] . . . . .	35
2.23	Diagramme en bloc représentant la technique du filtrage de Wiener . . . . .	36
2.24	Extrait de la taxonomie des différentes familles de super résolution présentée dans l'étude approfondie de Nasrollahi et al. [152]. . . . .	42
2.25	Principe permettant la reconstruction d'images HR ( <i>droite</i> ) à partir d'images basse-résolution LR ( <i>gauche</i> ). Les décalages inter-pixel fournissent les informations complémentaires contenues dans les images LR. L'étape de recalage est au centre. . . . .	43
2.26	Méthode IBP schématisée dans les travaux de Patel et al. [121] . . . . .	46
2.27	Méthode IBP simple image avec ajout d'un détecteur de contour [140] . . . . .	48
2.28	Méthode DASR, présentée par Demirel et al. [107] . . . . .	49
2.29	Des patches extraits des travaux de Freeman et al. [61], le patch de gauche a les mêmes caractéristiques que les autres patches du dictionnaire, ce qui rend impossible le simple remplacement du patch par son équivalent HR. . . . .	50
2.30	Modèle du réseau de Markov présenté dans les travaux de Freeman et al. [61] . . . . .	50
2.31	Méthode dites de Gohshi . . . . .	52
2.32	Convertisseur HD-UHD créé par Gohshi . . . . .	52
2.33	La méthode de Perez présentée dans [158]. Pour chaque patch de l'image de test, une carte des dégradations possibles est estimée avec un arbre de décision boosté. En fonction de cette probabilité une interpolation bicubique, ou une méthode complexe basée sur une régression linéaire est réalisée. . . . .	53
3.1	Exemple d'imagettes provenant de la base $S_{métrique}$ utilisées dans l'étude sur les métriques. . . . .	60
3.2	Traitements appliqués sur les imagettes $1024 \times 1024$ pixels extraits d'images de résolution d'origine 8k . . . . .	61

3.3	Exemple d'images issues des traitements utilisés pour la comparaison des métriques, (a) correspond à l'image source, (b) image floue provenant de $T_1$ , (c) image issue de $T_2$ : image provenant de $T_1$ suivit d'un rehaussement de contours et l'image (d) correspond au traitement $T_3$ , rehaussement de contours appliqué à l'image source. . . . .	62
3.4	Capture d'écran de l'application développée pour réaliser un test psychovisuel PVA de manière automatique . . . . .	65
3.5	Principe de la méthode de Gohshi . . . . .	66
3.6	Comparaison des résultats obtenus avec la convolution d'un signal "marche" de 0 à 255 (fonction échelon) avec les deux noyaux $F_{extrac}$ arrondi (à gauche) et non arrondi (à droite). . . . .	70
3.7	Image de synthèse "échelon" de type sigmoïde (a), et visualisation d'une ligne de cette image sur le graphique représenté en (b). L'amplitude de ce signal est de 0 à 255. . . . .	70
3.8	La courbe présente en (a) correspond au résultat du filtre passe-haut HPF sur l'image 3.7-a, les courbes de (b) représentent le résultat de la fonction non linéaire $y = x^3$ (bleue) et $y = x^5$ (rouge). . . . .	71
3.9	Image réelle de test "Lenna" présentée en (a) dont une ligne est sélectionnée "blanche" pour la visualisation des résultats. La courbe présente en (b) illustre les variations en niveaux de gris le long de la ligne blanche. . . . .	71
3.10	Résultat du filtre passe-haut du second ordre appliqué sur l'image de Lenna, seuls les résultats correspondants à la ligne blanche de la figure 3.9-a sont présentés. . .	72
3.11	Le signal en (a) représente le résultat de la fonction non linéaire $y = x^3$ appliqué sur l'image de contours et le signal en (b) les résultats de $y = x^5$ . . . . .	72
3.12	Comparaison de différentes fonctions non linéaires appliquées sur le signal de contours : $y = x^3$ (a), $y = x^5$ (b), le résultat des fonctions non linéaires est ajouté à l'image d'origine 3.11-a . . . . .	73
3.13	Comparaison de différentes techniques de limitation du signal. (a) correspond à un écrêtage franc, (b) correspond à un écrêtage plus doux et (c) correspond à une fonction de limitation qui peut être assimilée à une compression. . . . .	73
3.14	Différences entre une interpolation de Lanczos (gauche) et la méthode de Gohshi(droite) pour les zones 1,2 . . . . .	74
3.15	Exemples provenant de l'ensemble $S_1$ , triés en fonction de la texture. De haut en bas : hautes fréquences $HF$ , fréquences moyennes $MF$ et basses fréquences $BF$ .	75
3.16	Méthode de Super Résolution Spatiale Adaptative Localement en fonction du niveau de texture (LASSR). . . . .	77
3.17	Analyse de texture basée sur le Machine learning : Apprentissage hors ligne et Décision de classification en ligne . . . . .	78
3.18	Exemples de patches texturés et non texturés : Les patches hautes fréquences (MHF) (1 <sup>ère</sup> ligne), patches basses fréquences (BF) (2 <sup>ème</sup> ligne) . . . . .	81
3.19	Extraction d'attributs à partir du gradient de Sobel . . . . .	81

3.20	Évolution de l'erreur de classification globale pour les méthodes de SVM et de Boosting en fonction de l'évolution de la taille $n$ de fenêtrage ( $n \times n$ ) utilisée pour le calcul de l'attribut. . . . .	82
3.21	Les Images originales ( <i>gauche</i> ) et les images représentant la classification de texture ( <i>droite</i> ). Et plus précisément de haut en bas : "Portrait" (a), "Londres" (b) et "Vienne" (c) . . . . .	83
3.22	Schéma bloc présentant le fonctionnement de la fusion Locale . . . . .	84
3.23	Exemple de fusion d'une image issue de la méthode de Gohshi (a) et de l'image interpolée (b) avec (d) et sans (c) égalisation locale de la luminance . . . . .	85
3.24	Comparaison visuelle pour trois patches d'une dimension de $1024 \times 1024$ pixels. Avec (a) extraits d'images d'une résolution de 8k ; (b) résultats d'une interpolation de Lanczos <i>LII</i> , (c) résultats du traitement de Gohshi <i>GFI</i> et (d) résultats avec la méthode proposée <i>LASSR</i> . . . . .	87
3.25	Comparaison visuelle d'un extrait $1024 \times 1024$ pixels d'une image 8k bruitée (3200 iso). Avec (a) un extrait de l'image source, (b) résultat d'une interpolation de Lanczos <i>LII</i> , (c) résultat du traitement de Gohshi <i>GFI</i> et (d) résultat avec la méthode proposée <i>LASSR</i> . Les patches encadrés en rouge correspondent à des agrandissements de taille $64 \times 64$ pixels. . . . .	88
3.26	Exemple d'images provenant de $S_3$ utilisées durant le test psychovisuel <i>PVA</i> <sub>2</sub> . Les images de taille $1024 \times 1024$ sont affichées 1 pixel pour un . . . . .	89
4.1	Méthodologie de développement de la méthode <i>LASSR</i> . . . . .	94
4.2	Méthodologie de génération du modèle de décision . . . . .	95
4.3	Méthodologie de développement logiciel de la méthode <i>LASSR</i> . . . . .	96
4.4	Exemple de synchronisation d'un capteur CMOS . . . . .	97
4.5	Transcodage C++ vers VHDL depuis Vivado HLS . . . . .	97
4.6	Méthode de test et de validation de l'implantation matérielle . . . . .	98
4.7	Schéma bloc présentant l'architecture globale de la méthode <i>LASSR</i> avec une division en sous-parties des différents blocs qui composent cette méthode : interpolation, Filtre de Gohshi (GF), Classification de texture (TC) et fusion locale. . .	99
4.8	Schéma bloc du fonctionnement du Voisinage de Pixel Générique GCM . . . . .	101
4.9	Exemple de génération d'un arbre binaire de convolution d'une taille $N \times M$ à l'aide du processeur générique (avec $N = M = 5$ ). Les pixels d'entrée, codés sur 8 bits, sont concaténés dans le vecteur $Vpix_{(N \times M)}$ , et les coefficients, codés sur 9 bits, dans $Vcoef_{(N \times M)}$ . . . . .	102
4.10	Décomposition en blocs de la méthode de Gohshi implantée, le filtre passe-haut <i>HPF</i> utilise le filtre générique présenté dans la section 4.1.2.1. . . . .	104
4.11	Architecture du bloc correspond à l'analyse de texture (TC), mise en pipeline du gradient de Sobel $3 \times 3$ , du calcul de la variance $\sigma^2$ et de la fonction de décision. . . . .	106
4.12	Schéma bloc du calcul du gradient de Sobel en utilisant les composants du filtre générique en mode convolution . . . . .	106

4.13	Calcul de la variance en utilisant les composant GKP et GCM sur une fenêtre de taille $n \times n$ , avec $N$ le nombre de pixels de la fenêtre. . . . .	107
4.14	Évolution du taux d'erreur globale pour la méthode de Boosting et du taux l'utilisation des ressources du FPGA Virtex 7 (Slices) en fonction de la taille de la sous-fenêtre $n$ pour le calcul des attributs. . . . .	108
4.15	Fonctionnement de la fusion locale, décomposé en différents blocs utilisant le filtre générique . . . . .	109
4.16	Unité de routage des données permettant la gestion des données pour la mise en parallèle de modules LASSR. . . . .	114
4.17	Représentation du démonstrateur à réaliser dans le cadre du projet H2020-EXIST. . . . .	118
4.18	Révolution des PSFs en fonction de la distance au centre de l'image. La moitié seulement des PSFs d'une ligne sont fournies, il est nécessaire de les dupliquer en les faisant pivoter. Les différents cercles concentriques représentent de façon schématique diverses PSF. La rotation des PSF est illustrée dans la figure 4.19 . . . . .	119
4.19	Représentation schématique de deux PSFs. La PSF d'origine $PSF_{ori}$ (fournie par le constructeur) et la PSF pivotées $PSF_{rot}$ pour conserver la distance des coefficients par rapport au centre de l'image. . . . .	120
4.20	Image source divisée en blocs, chaque bloc utilise une PSF différente. . . . .	120
4.21	Image source divisée en blocs où chaque bloc utilise une PSF différente. . . . .	121
4.22	Déconvolution non-aveugle d'une image floue avec l'utilisation du filtre de Wiener dans le domaine fréquentiel. . . . .	121
4.23	Application du filtre de Wiener sur une image subdivisée en blocs. Chaque bloc de l'image correspond à une PSF différente et est traité en parallèle. . . . .	121
4.24	Imagette d'une taille de $200 \times 200$ pixels extraite au centre de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence. . . . .	122
4.25	Imagette d'une taille de $100 \times 100$ pixels extraite sur un bord de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence. . . . .	123
4.26	Imagette d'une taille de $100 \times 100$ pixels extraite sur un bord de l'image. De gauche à droite : Image dégradée, image corrigée, et image de référence. . . . .	123
4.27	L'image d'entrée d'une mire imprimée sur une feuille A4 ( $210 \text{ mm} \times 297 \text{ mm}$ ) capturée avec la caméra créée par Adimec. La distance à laquelle l'image a été capturée est de 3 mètres. . . . .	124
4.28	Imagette d'une taille de $300 \times 200$ pixels extraite au centre de la mire papier. L'imagette sans correction à gauche, et l'imagette corrigée à droite. . . . .	124
4.29	Imagette d'une taille de $250 \times 250$ pixels extraite sur le coté droit de la mire papier. L'imagette sans correction à gauche, et l'imagette corrigée à droite. . . . .	125
4.30	Architecture du démonstrateur . . . . .	125
4.31	Image 2k capturée par la caméra ( <i>gauche</i> ) et image corrigée par en temps réel ( <i>droite</i> ) . . . . .	126
4.32	Imagettes extraites de l'image brute sans correction (a et c), imagettes extraites de l'image corrigée (b et d) . . . . .	127

4.33	Caméra conçue pour le projet EXIST-H2020, le corps de la caméra développée par ADIMEC, et l'objectif par Thalès Angénieux (TAGX). . . . .	127
5.1	Photographie d'un test comparatif sur un écran de cinéma 4k aux Ateliers du Cinéma à Beaune. Deux images sont présentées : une image issue de notre traitement LASSR, et une image d'une résolution de 4k. . . . .	132
A.1	Exemple de répartitions de données : (a) linéairement séparables, (b, c) non linéairement séparables . . . . .	157
A.2	Principe de fonctionnement des SVM sur une classification avec deux classes et des données séparables [165] . . . . .	158

## LISTE DES TABLES

2.1	Comparaison des deux types de méthodes de compression d'image : avec pertes et sans perte. . . . .	18
3.1	Classement des traitements par les métriques sans référence sélectionnées dans la section 2.2 (l'ordre est présenté entre parenthèses). Les valeurs correspondent aux moyennes des métriques obtenues sur $S_{métrique}$ . . . . .	63
3.2	Classement des traitements par les métriques avec référence (l'ordre est présenté entre parenthèse et est obtenu avec une précision plus grande que celle affichée dans le tableau). Les valeurs correspondent aux moyennes des métriques obtenues sur $S_{métrique}$ . . . . .	63
3.3	Comparaison entre l'image provenant du filtre de Gohshi (GFI) et l'image interpolée avec Lanczos (LII) en fonction du niveau de fréquence de l'image ( $HF$ , $MF$ et $BF$ ) . . . . .	76
3.4	Résultats de la méthode LASSR comparés au traitement de Gohshi $GFI$ et d'une interpolation $LII$ pour des métriques avec et sans-référence sélectionnées dans la section 3.1.4 . . . . .	86
3.5	Résultat de l'évaluation $PVA_2$ pour les méthodes GFI, II and LASSR . . . . .	90
4.1	Utilisation des ressources matérielles du FPGA Virtex 7 (VC 707) pour la méthode LASSR séparée en blocs SR , TC & Fusion . . . . .	111
4.2	Comparaison entre le Virtex-7 (xc7vx485tffg1761-2), l'Artix-7 (7a200tfgg676-2) et le Spartan-7 (7s100fpga676-2) en terme de ressources matérielles disponibles et de coût. Le prix correspond aux prix constatés sur mouser.fr pour la puce FPGA seule. . . . .	111
4.3	Utilisation des ressources matérielles du FPGA Artix 7 (AC 701) pour la méthode LASSR séparée en blocs SR , TC & Fusion . . . . .	112
4.4	Utilisation des ressources matérielles du FPGA Spartan 7 (SP 701) pour la méthode LASSR séparée en blocs SR , TC & Fusion . . . . .	112
4.5	Utilisation des ressources FPGA pour l'implantation en parallèle de plusieurs modules LASSR et estimation de la fréquence image (im/s) à partir de l'horloge contrainte à 7 ns. . . . .	113
4.6	Caractéristiques d'implémentations matérielles présentes dans l'état de l'art de méthodes de Super Résolution comparées à notre méthode LASSR avec la mise en place de 1 ou 4 cores en parallèles. . . . .	117





# I

## ANNEXES



## LES SÉPARATEURS À VASTE MARGE

Les Séparateurs à Vastes Marges ont été proposés comme nouvelle technique de classification en 1995 par Vapnik et al. [36]. Il s'agit d'une technique très largement utilisée en classification, en régression et aussi en estimation de densité.

Les Séparateurs à Vastes Marges cherchent à séparer aux mieux des ensembles de données appartenant à des classes différentes. La figure A.1 présente trois différents cas de répartitions et séparations de données. En effet plusieurs cas sont possibles comme dans la figure A.1-a où les données sont séparables linéairement, et les figures A.1-b et A.1-c ne sont pas linéairement séparables.

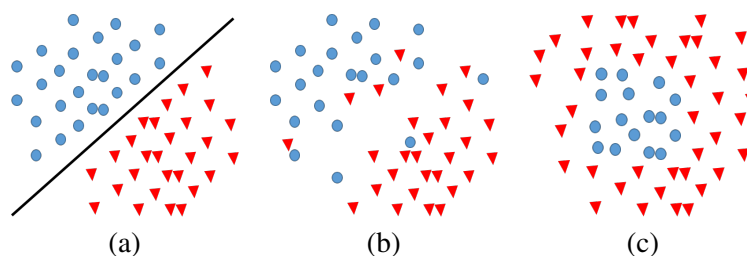


FIGURE A.1 – Exemple de répartitions de données : (a) linéairement séparables, (b, c) non linéairement séparables

Cette séparation est effectuée par un hyperplan. Cet hyperplan, qui correspond à une droite dans un plan à deux dimensions, est généré, ou supporté, par certains vecteurs appartenant aux classes à séparer. Ces vecteurs sont alors appelés vecteurs supports. Les hyperplans supports correspondent aux frontières des classes. La marge  $M$  correspond à la distance entre les hyperplans supports. Le but des SVM est de rechercher l'hyperplan séparateur qui permet de séparer les classes tout en maximisant la marge  $M$  entre les classes.

### A.1/ CAS LINÉAIREMENT SÉPARABLES - SÉPARATEURS LINÉAIRES

Pour deux classes d'échantillons linéairement séparables, chaque échantillon de l'ensemble d'apprentissage est représenté par le couple  $(x_i, y_i)$  où  $x_i$  est l'échantillon associé à l'étiquette de classe  $y_i$ , avec  $y_i \in \{-1; 1\}$  pour deux classes différentes. Notons  $w_i$  la normale à l'hyperplan séparateur  $H$  qui a pour équation :

$$H : \quad \vec{w} \cdot \vec{x}_i + b = 0 \quad (\text{A.1})$$

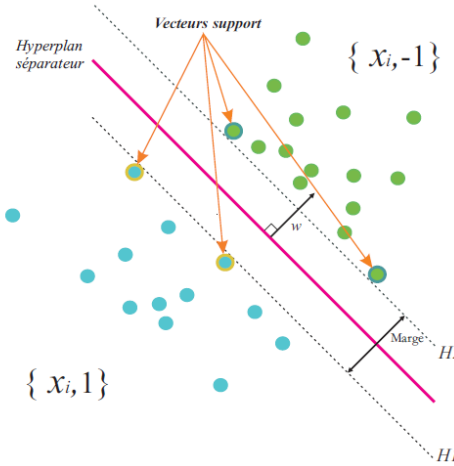


FIGURE A.2 – Principe de fonctionnement des SVM sur une classification avec deux classes et des données séparables [165]

La méthode d'apprentissage SVM consiste à maximiser la marge  $M$  entre l'hyperplan séparateur et les hyperplans supports  $H_1$  et  $H_2$ . Les hyperplans supports ont pour équation :

$$H_1 : \quad \vec{w} \cdot \vec{x}_i + b = 1 \quad H_2 : \quad \vec{w} \cdot \vec{x}_i + b = -1 \quad (\text{A.2})$$

Ces deux plans doivent vérifier l'inégalité donnée par l'équation A.3 suivante pour qu'aucun point de la base d'apprentissage n'existe entre eux :

$$\forall i \in [1 : n] \quad f(\vec{x}_i) = y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 \quad (\text{A.3})$$

La distance entre un vecteur  $\vec{x}$  et l'hyperplan  $H$  et la marge  $M$  sont définis par :

$$d(\vec{x}, H) = \frac{f(\vec{x})}{\|\vec{w}\|} \quad M = \frac{2}{\|\vec{w}\|} \quad (\text{A.4})$$

Pour maximiser  $M$  il faut donc minimiser  $\|\vec{w}\|$ . Il s'agit donc d'un problème d'optimisation quadratique qui peut être résolu grâce à la méthode de Lagrange. L'optimisation de Lagrange n'est pas explicitée. Après optimisation, la fonction de décision  $g(x)$  de décision pour un nouvel échantillon  $x$  est donnée dans l'équation suivante :

$$g(\vec{x}) = \text{signe} \left( \sum_{i=1}^{N_v} \alpha_i y_i \vec{S}_i \cdot \vec{x} + b \right) \quad (\text{A.5})$$

où  $\alpha_i$  sont des multiplicateurs optimaux de Lagrange. L'hyperplan séparateur est construit à partir des vecteurs supports  $N_v$ , pour lesquelles  $\alpha_i \neq 0$ . Les vecteurs  $S_i, i = 1, \dots, N_v$  avec l'étiquette  $y_i \in \{-1, +1\}$  sont les vecteurs supports.

## A.2/ CAS NON LINÉAIREMENT SÉPARABLES - SÉPARATEURS LINÉAIRES

Dans la plupart des cas, les problèmes de classification sont loin d'être résolus à travers une séparation linéaire. Il est possible de voir (figure A.1-b) un cas où il n'est pas possible de séparer avec une droite les données. Cependant il est possible de résoudre ce problème en introduisant une marge *souple*. Cette marge permet de tolérer des erreurs de classification. Ces erreurs sont le résultat de la présence d'échantillons du mauvais côté de la frontière de séparation lors de la phase d'apprentissage. La marge d'erreur  $\varsigma_i$  est ajoutée à l'équation A.3, ce qui donne l'équation suivante :

$$\forall i \in [1 : n] \quad f(\vec{x}_i) = y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \varsigma_i \quad (\text{A.6})$$

Afin de maximiser la marge il faut cette fois-ci minimiser la quantité suivante :

$$\frac{1}{2} \|\vec{w}\|^2 + C \sum_i^n \varsigma_i \quad (\text{A.7})$$

Où  $C$  est le coût attribué à un échantillon mal classé.

Une séparation linéaire n'est parfois plus capable de séparer correctement tous les échantillons et il faut parfois utiliser des séparateurs non linéaires.

## A.3/ SÉPARATEURS NON LINÉAIRES

Les cas non linéairement séparables utilisent une fonction de noyau non linéaire  $K(x, y)$  afin de représenter les données dans un nouvel espace où il est possible de séparer les données par un hyperplan. Cette fonction de noyau  $K$  est utilisée dans le problème de l'optimisation quadratique afin de déterminer les nouveaux vecteurs supports.

$$g(\vec{x}) = \text{signe} \left( \sum_{i=1}^{N_v} \alpha_i y_i K(\vec{S}_i, \vec{x}) + b \right) \quad (\text{A.8})$$

La forme de  $K(x, y)$  est pour la majorité des cas choisie de manière empirique selon l'application et la répartition des données. Pour information, les trois principales formes de  $K$  sont :

- Fonction de base radiale gaussienne :  $K(\vec{x}, \vec{y}) = e^{(\gamma \|\vec{x} - \vec{y}\|^2)}$
- Noyau Polynomial :  $K(\vec{x}, \vec{y}) = [\gamma(\vec{x} \cdot \vec{y}) + \theta]^q$
- Noyau Sigmoidé ou tangent hyperbolique :  $K(\vec{x}, \vec{y}) = \tanh[\gamma(\vec{x} \cdot \vec{y}) - \theta]$

Les différents paramètres des fonctions noyau ( $\gamma, \theta, q$ ) sont eux aussi à optimiser en fonction de l'application









---

Code B.3 – Extrait du fichier DefTabBoost.h généré automatiquement pour un apprentissage donné

```

#include "BoostHardware.h"
#include "DefTabBoost.h"

CBoostHardware::CBoostHardware(void)
{
}

CBoostHardware::~CBoostHardware(void)
{
}

int CBoostHardware::MyPredict(int t, float *x)
{
    int res;
    float val=x[t];
    float Bor= Borne[t];

    if (Type[t]==1)
    {
        if ( (Bor-val)>=0.0) res=1; else res=0;
    }
    else
    {
        if ( (Bor-val)>0.0 ) res=0; else res=1;
    }
    return res;
}

char CBoostHardware::FinalClass(float *x)
{
    float ClasseFinale=0.0;
    SBoostOutput Fin;
    unsigned char cf;
    int t,v;
    FinalClass_label12:for (t=0;t<NBITERATION;t++)
    {
        v = MyPredict(t,x);
        if (v==1) ClasseFinale+=MonAlpha[t];
        else ClasseFinale+=-MonAlpha[t];
    }
    if (ClasseFinale <0.0) cf=-1.0;else cf=1.0;
    return cf;
}

int CBoostHardware::MyPredictRectangle(int t, unsigned char *x)
{
    int ext=0;
    int TypeLocal;
    int idx=t*CONSTNPAR;
    TypeLocal=Type[idx];

    MyPredictRectangle_label10:for (int par=0;par<CONSTNPAR;par++)
    {
        if (x[par]<BorneInf[idx]) {ext=1;break;}
        if (x[par]>BorneSup[idx]) {ext=1;break;}
        idx++;
    }
}

```

```

        if (ext==0) return TypeLocal; else return -TypeLocal ;
    }
char CBoostHardware::FinalClassRectangle(unsigned char *x)
{
    int ClasseFinale=0;
    SBoostOutput Fin;
    char cf;
    int t,v;

FinalClassRectangle_labell :
    for (t=0;t<NBITERATION;t++)
    {
        v = MyPredictRectangle(t,x);
        if (v==1) ClasseFinale+=(int)MonAlpha[t];
        else ClasseFinale+==-(int)MonAlpha[t];
    }
    if (ClasseFinale <0.0) cf=-1.0;else cf=1.0;
    return cf;
}

```

Code B.4 – Extrait du fichier BoostHardware.cpp valable quel que soit le résultat de l'apprentissage

```

#pragma once

#include "StructBoost.h"

class CBoostHardware
{
public:
    CBoostHardware(void);
    ~CBoostHardware(void);

    int MyPredict(int t, float *x);
    int MyPredictRectangle(int t, unsigned char *x);
    char FinalClass(float *x);
    char FinalClassRectangle(unsigned char *x);
};

```

Code B.5 – Extrait du fichier BoostHardware.h

```

#pragma once

struct SHyperplan
{
    float Borne;
    int Par;
    float Error;
    int Type;
};

struct SHyperInter
{
    float BorneInf;
    float BorneSup;
    float Error;
};

```

```

        int Par;
        int Type;
};

struct SBoostOutput
{
    int cf;
    float Foutput;
};

struct SHyperInterHard
{
    float BorneInf;
    float BorneSup;
    int Type;
};

struct SHyperInterHardBYTE
{
    unsigned char BorneInf;
    unsigned char BorneSup;
    char Type;
};

```

Code B.6 – Extrait du fichier StructBoost.h

### B.1.2/ TRANSCODAGE ET DIRECTIVES

Le fichier code B.7 présente les directives ajoutés aux différents fichiers C++. Ces directives ont été ajoutées manuellement pour accélérer le traitement. En effet, il existe différentes boucles ce qui augmente les latences d'exécution. Il existe deux manières d'ajouter une directive à une partie du code :

- Ajouter à l'aide de `#pragma HLS directive;`
- ajouter directement, à l'aide d'un clique droit une directive.

```

#####
## This file is generated automatically by Vivado HLS.
## Please DO NOT edit it.
## Copyright (C) 2015 Xilinx Inc. All rights reserved.
#####
set_directive_unroll "CBoostHardware::FinalClass/FinalClass_label10"
set_directive_unroll "CBoostHardware::FinalClass/FinalClass_label11"
set_directive_unroll "CBoostHardware::MyPredictRectangle/
    MyPredictRectangle_label12"
set_directive_unroll "CBoostHardware::FinalClassRectangle/
    FinalClassRectangle_label13"
set_directive_pipeline "CBoostHardware::FinalClassRectangle"
set_directive_unroll "CBoostHardware::FinalClass"
set_directive_pipeline "CBoostHardware::FinalClass/FinalClass_label10"
set_directive_interface -mode ap_none "top" a
set_directive_interface -mode ap_none "top" b
set_directive_pipeline "CBoostHardware::MyPredict"
set_directive_unroll "CBoostHardware::FinalClass/FinalClass_label14"

```

Code B.7 – Extrait du fichier directives.tcl

Les directives utilisées sont les suivantes :

- `set_directive_interface -mode ap_none` : elle permet de définir les variables `a` et `b` du fichier `top` comme des "ports".
- `set_directive_unroll` : elle permet de transformer une boucle en une multitude de copies du corps de la boucle. Cela crée autant de copie qu'il y a d'itération. Cette directive est appliquée lorsque les itérations de la boucle ne sont pas indépendantes les unes des autres.
- `set_directive_pipeline` : elle permet de diminuer les latences entre deux itérations consécutives. Cela est possible lorsque pour chaque itérations d'une boucle plusieurs exécutions consécutives sont réalisées.

Les indications entre guillemets permettent d'indiquer où sont appliquées les directives.





