Contrôle des performances et conciliation d'erreurs dans les décodeurs d'image

Ghislain Takam Tchendjou

Communauté UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE ALPES

Spécialité : **NANO ELECTRONIQUE ET NANO TECHNOLOGIES**

Arrêté ministériel : 25 mai 2016

Présentée par

Ghislain TAKAM TCHENDJOU

Thèse dirigée par **Emmanuel SIMEU,** MCF, UGA

préparée au sein du **Laboratoire Techniques de l'Informatique et de la Microélectronique pour l'Architecture des systèmes intégrés**
dans **l'École Doctorale Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)**

Contrôle des performances et conciliation d'erreurs dans les décodeurs d'image

Performance monitoring and errors reconciliation in image decoders

Thèse soutenue publiquement le **12 décembre 2018**,
devant le jury composé de :

M. Ian O'CONNOR
Professeur, Ecole Centrale de Lyon, Président du jury
M. Patrick GIRARD
Directeur de Recherche, Laboratoire LIRMM, Rapporteur
M. Fabrice MONTEIRO
Professeur, Université de Lorraine, Laboratoire LGIPM, Rapporteur
M. Salvador MIR
Directeur de Recherche, Laboratoire TIMA, Examinateur
M. Fritz LEBOWSKY
Ingénieur sénior, ST Microelectronics Grenoble, Examinateur
M. Emmanuel SIMEU
MCF, Université de Grenoble Alpes, Laboratoire TIMA, Directeur de thèse

A ma famille ...

Remerciements

Je suis très heureux d'être arrivé au terme de cette belle et enrichissante expérience qu'est la thèse. Je voudrais remercier toutes les personnes qui m'ont accompagné, de près ou de loin dans la réalisation de ce travail.

Je tiens avant tout à remercier mon directeur de thèse M. Emmanuel SIMEU, de m'avoir permis d'effectuer cette thèse et d'avoir suivi ce travail. Merci pour l'encadrement, les conseils, la disponibilité, les remarques, la rigueur et la confiance que tu as su m'accorder tout au long de cette thèse.

Je remercie M. Ian O'CONNOR, Professeur à l'INL École Centrale de Lyon, d'avoir présidé mon jury de thèse. Merci à M. Patrick GIRARD du laboratoire LIRMM de Montpellier et M. Fabrice MONTEIRO de l'Université de Lorraine d'avoir accepté d'être les rapporteurs de ce travail, et pour le temps qu'ils ont accordé à la lecture de ce manuscrit. Je remercie également M. Salvador MIR du laboratoire TIMA de Grenoble et M. Fritz LEBOWSKY de STMicroelectronics Grenoble d'avoir accepté de faire partie de mon jury de thèse.

J'adresse mes sincères remerciements à mes collègues de bureau et amis Rshdee ALHAKIM et Hani MALLOUG qui m'ont accompagné autant dans mes travaux de recherche, que dans la vie courante. Merci à tous les membres de l'équipes RMS (Reliable Mixed-signal Systems) pour le soutien et les bons moments, je pense ici à Manuel, Daniel, Chadi, Diane, Halim, Mohamed, Renato, Florent, Marc et Athanasios.

Je tiens également à remercier l'ensemble des membres du laboratoire TIMA, permanents, thésards et personnels pour l'aide et le soutien pendant cette thèse ; je pense ici à Anne-Laure, Laurence, Youness, Mamadou, Aurore, Alice, Frédéric, Ahmed et Nicolas.

Je voudrais également saisir cette occasion pour remercier mon grand-père, Papa David TCHENDJOU, merci pour la relecture qui a dû t'empêcher de dormir pendant plusieurs jours, merci pour tes remarques et pour les nombreuses leçons de la langue de Molière.

Un grand merci à tous mes amis qui m'ont encouragé et accompagné durant cette thèse. Merci à mes camarades et amis du LYBIBAF, ceux de l'UdM et ceux de Grenoble, vous avez tous autant que vous êtes contribué au bon déroulement de cette thèse.

Je souhaite remercier particulièrement mes parents, ce travail est une occasion pour moi de vous témoigner tout mon amour et ma reconnaissance pour tous vos sacrifices. Puisse le Seigneur vous combler de ses grâces et bénédictions. Un merci tout particulier à mon grand-frère Cédric, à mon petit-frère Juvenal, et à mes sœurs Carine et Babette pour le soutien et les encouragements. Merci à mes beaux frères Patrick et Steve, et à ma

petite nièce Kate. Merci à mes grand-mères, et à toute ma grande famille, mes oncles, mes tantes, mes cousins, mes cousines, mes neuves et mes nièces.

Ils sont nombreux ceux que je n'ai pas cité ici, mais que chacun trouve dans ce mémoire une profonde pensée pour lui et l'expression de ma gratitude.

Je voudrais finir par dire toute ma reconnaissance à Dieu, pour la santé, la forme et surtout la force qu'il m'a accordé durant ces années. Sans ces éléments, je n'aurai sûrement pas pu arriver au bout de ce long et tumultueux périple. À lui soit la gloire !

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Chapitre 1

Introduction générale

Chaque jour, des millions d'images sont prises par des caméras des utilisateurs divers ou générées par des ordinateurs, ça et là, à travers le monde. Toutes ces images sont, le plus souvent, stockées ou transmises via des canaux de communication. Les canaux de transmission tout comme les espaces de stockage étant très limités, les images capturées doivent, la plupart du temps, passer par des phases de compression et de décompression tout au long de leur processus d'acquisition. Ce processus d'acquisition consiste en cinq phases principales : la capture, le codage, la transmission, le stockage et l'affichage. A chacune de ces phases, de multiples dégradations s'ingèrent dans les données initiales au point d'entraîner en fin de parcours des images altérées, perturbantes pour la perception visuelle. La Figure 1.1 décrit cette chaîne d'acquisition, ainsi que quelques dégradations pouvant intervenir à chacune des diverses phases.

Les images numériques résultant de la chaîne d'acquisition sont utilisées dans divers domaines de la vie dont les plus courant sont : la photographie, la télévision numérique, la vidéo surveillance, l'ingénierie médicale, la technologie mobile [1, 2]. Pour son bon fonctionnement chacun de ces domaines a des contraintes de qualité différentes selon l'usage et aussi en fonction de l'image. Ainsi, une caméra embarquée dans une voiture ne présente pas les mêmes contraintes de qualité d'image produite qu'une caméra utilisée

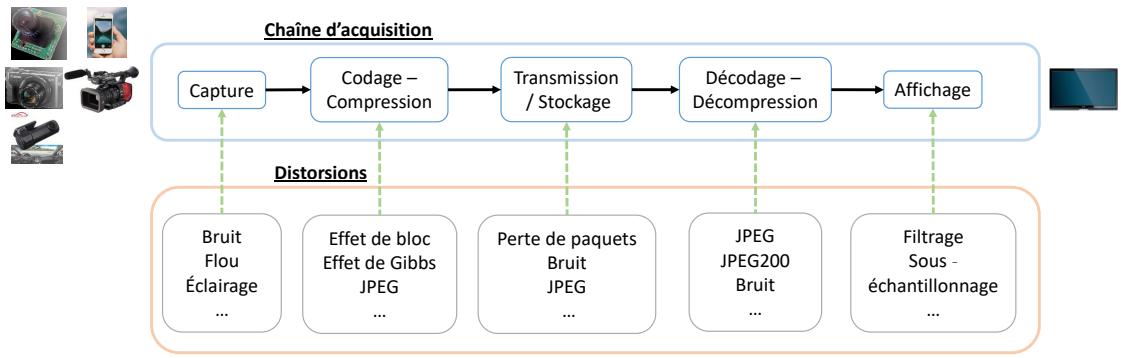


FIGURE 1.1 – Chaîne d'acquisition d'une image numérique

dans le domaine médical, par exemple. C'est cette différence dans les seuils de qualité dans chacun des domaines d'utilisation des images numériques qui donne une importance accrue aux domaines de l'évaluation de la qualité des images (Image Quality Assessment : IQA) et de correction des images dégradées. En télévision numérique, les utilisateurs n'ont plus les mêmes contraintes et exigences en terme de qualité qu'il y a une vingtaine d'années [2]. Aujourd'hui, nous assistons davantage à l'avènement de la télévision à ultra haute définition.

Cette recherche s'inscrit dans le cadre du Projet $NANO2017$, qui est un projet collaboratif entre des partenaires industriels et académiques. La partie du projet retenue dans la présente étude concerne l'utilisation des méthodes avancées d'analyse statistique pour la détection et la correction d'erreurs intégrées dans un décodeur numérique d'images à ultra haute définition. Ce projet vise principalement des décodeurs vidéos basés sur le CODEC (Codeur / Décodeur) $HEVC$ (« High Efficiency Video Coding »), permettant la transmission de signaux compressés pour l'ultra haute définition (2K, 4K, 8K). $MPEG-4\ HEVC$ ou encore $H.265$ est une norme de codage vidéo développée conjointement par ISO/CEI Video Coding Experts Group ($VCEG$) et $UIT-T$ Moving Picture Experts Group ($MPEG$) [3], dont le but principal est d'améliorer la compression vidéo en conservant la même qualité en termes de perception visuelle humaine des vidéos compressées et transmises.

1.1 Évaluation de la qualité d'image

La correction des distorsions causées tout au long de la chaîne d'acquisition dans les images, passe obligatoirement par une phase d'évaluation de la qualité des images. Cette évaluation permet ainsi de déterminer quelles zones de ces images ont besoin d'être corrigées. De plus, cette opération est à la base de la définition du seuil d'acceptabilité de la qualité des images prises et corrigées selon les différents domaines. Puisque les êtres humains sont les utilisateurs finaux et les interprètes des images numériques, la meilleure manière d'évaluer la qualité d'une image est de demander à un groupe d'observateurs humains de regarder l'image et d'en estimer la qualité. Cette approche est une méthode d'**évaluation subjective**. Dans la pratique, les méthodes d'évaluation subjectives de la qualité des images sont très coûteuses tant en termes de temps et de matériels qu'en ceux des ressources humaines et de moyens financiers pour leur mis en œuvre. Elles ne sont donc pas utilisables pour les applications temps-réelles ou embarquées [4, 5].

Pour pallier aux divers problèmes posés par les méthodes subjectives d'évaluation de la qualité des images, des méthodes automatiques d'évaluation ont été mises sur pied afin de prédire la qualité des images. Ces dernières se basent sur un ensemble de caractéristiques spécifiques ou sur la comparaison entre plusieurs images ; il s'agit de la méthode d'**évaluation objective** de la qualité des images.

Depuis peu, des recherches sont menées pour construire des modèles objectifs d'évaluation de la qualité des images en rapport avec la perception visuelle humaine. Certains de ces modèles utilisent les résultats de l'évaluation subjective comme des références pour la construction des modèles objectifs d'évaluation de qualité des images. Quelques-uns sont

basés sur des méthodes d'apprentissage automatiques, à savoir : les réseaux de neurones artificiels, les k-plus proches voisins, les machines à vecteurs supports, la régression non linéaire, etc. [6, 7, 8, 9].

Dépendant de la disponibilité de l'image de référence, en l'occurrence de l'image originale à partir de laquelle est obtenue celle déformée à évaluer, l'évaluation objective de la qualité des images peut être divisée en trois (03) grandes familles :

1. Mesures avec référence : elles sont utilisées pour estimer la qualité entre deux images dont une image originale et sa version dégradée. Elles sont utiles pour le contrôle des performances des algorithmes tels que les algorithmes de compression, de diagnostic, de correction des images. La majorité des métriques proposées dans la littérature sont des métriques avec référence, car elles produisent de très bons résultats en ce qui concerne la prédiction de qualité d'images. Les plus connues sont entre autres : $PSNR$ (rapport signal sur bruit de crête), $SSIM$ (Structural SIMilarity) [5], VIF (Visual Information Fidelity) [10].
2. Mesures à référence réduite : dans ces types de métriques, la qualité d'une image dégradée est évaluée à partir des informations tirées de cette même image et d'une partie d'informations tirées de l'image de référence. Ces méthodes à référence réduite sont très peu développées dans la littérature. Parmi les métriques à référence réduite les plus populaires, il y a lieu de citer : $RRED$ (Reduced Reference Entropic Differencing) [11] ou encore $SPCRM$ (Similarity index of PC Regularity Measures) [12].
3. Mesures sans référence : elles sont plus attrayantes d'un point de vue applicatif, car elle ne nécessite aucune information sur l'image de référence. Elles sont utilisées pour estimer la qualité d'une image dégradée même en n'ayant aucune information sur sa version originale ou de référence. Les domaines d'applications les plus évident de telles métriques s'observent dans l'évaluation de la qualité d'une image sortie d'un appareil photo, ou d'un décodeur numérique d'images. Il existe dans la littérature un bon nombre de métriques sans référence, dont les plus connues sont : $DIIVINE$ [13], $NIQE$ [14], $BRISQUE$ [15, 16].

En traitement d'image numérique, l'évaluation objective de la qualité des images peut être utilisées à plusieurs fins, parmi lesquelles : le contrôle dynamique et l'ajustement de la qualité des images lors de la phase de correction d'une image déformée [5], l'optimisation des paramètres des algorithmes de traitement des images [17], le contrôle des taux de compression des images numériques.

On retrouve de nombreuses recherches portant sur l'évaluation objective de la qualité des images dans la littérature. Ces méthodes existantes sont, pour la plupart, basées sur un unique axes d'extraction des caractéristiques des images ; ce qui empêche de voir le côté multidimensionnel de celles-ci. Aussi les modèles produits sont-ils, pour bon nombre, trop complexes, et surtout difficiles à implémenter sur un système matériel tel qu'un $FPGA$ ou un $ASIC$. En ce qui concerne les évaluations à partir des métriques sans référence, un grand nombre d'index présents dans la littérature sont liés à un type de distorsions bien spécifiques. Ce qui requiert une nécessité de toujours savoir quels types de dégradations sont présents dans une image avant de pouvoir en évaluer la qualité.

Dans une image naturelle, prise ici comme une image de bonne qualité visuelle, la présence d'un pixel décalé par rapport à son environnement perturbe énormément la perception visuelle des observateurs de cette image. Cette présence inopinée a pour conséquence de détériorer le score de qualité visuelle de cette image. Une méthode d'évaluation possible de la qualité d'une image serait de vérifier le taux de pixels défectueux présents dans ladite image.

1.2 Détection et correction des pixels défectueux

Un pixel défectueux peut se définir comme un pixel dont la valeur n'est pas en harmonie avec son environnement. Ce qui correspond à un pixel dont la valeur réelle est significativement différente de la valeur attendue en étudiant son voisinage. Une image ayant un taux très élevé de pixels défectueux sera forcément de mauvaise qualité, notamment sur le plan visuel. Généralement, la correction des pixels défectueux améliore la qualité visuelle de l'image[18].

La détection des pixels défectueux (DPD) est un processus de test consistant à rechercher les pixels s'écartant de leur environnement dans les images. Le taux de pixels défectueux présents dans une image peut permettre de présumer du score de qualité sur le plan de sa perception visuelle. Le processus de correction des pixels défectueux intervient, quant à lui, après la phase de détection, pour remplacer la valeur des pixels détectés comme défectueux par une valeur en accord avec son voisinage.

A la sortie des capteurs d'images de la fabrication en industrie, plusieurs tests industriels sont effectués, notamment celui de détection des pixels défectueux. Si le taux de pixels défectueux lors de ce test est supérieur à un certain seuil [19], défini suivant le domaine d'utilisation, le capteur n'est pas mis sur le marché, parce que jugé inutilisable. Cependant, même en cours d'utilisation, certains facteurs l'électrique, optique, voire de vieillissement peuvent conduire à une apparition de pixels défectueux sur les capteurs. D'où la nécessité de développer des méthodes temps-réelles de détection des pixels défectueux dans les capteurs d'images. Ces méthodes utilisent, pour la plupart, les images produites par un capteur pour détecter les défauts présents sur celui-ci.

La présence des méthodes temps-réel de détection des pixels défectueux dans les capteurs d'images est peu significative. Ces méthodes, d'après la Littérature consultée au cours de la présente étude, permettent de détecter uniquement des erreurs catastrophiques sur des valeurs extrêmes des pixels, soit 0 ou 255 pour une image en niveau de gris sur 8 bits. En plus, ces méthodes existantes produisent un taux très élevé de « faux positifs » ainsi dénommé lorsque le pixel est détecté « défectueux » par la méthode alors qu'il est « bon ». En plus des informations erronées sur la qualité réelle des images évaluées, les « faux positifs » augmentent la complexité de la phase de correction des pixels détectés comme défectueux. Une autre insuffisance des méthodes existantes s'observe dans la phase de correction des pixels détectés comme défectueux, lorsque celles-ci ne permettent pas de trouver des substituts véritablement appropriés susceptibles de remplacer les défaillants en améliorant la qualité visuelle de l'image.

1.3 Contribution

Dans cette thèse, deux contributions majeures ont été définies : la première portant sur la construction des modèles objectifs d'évaluation de la qualité des images ; la seconde portant sur la mise en place des algorithmes de détection et de correction des pixels défectueux dans une image ou un capteur d'image.

S'agissant du premier objectif qui est la construction des modèles d'évaluation de la qualité des images, elle est basée soit sur un ensemble de caractéristiques extraites de l'image à évaluer soit sur sa comparaison avec son image de référence. Le processus général de construction des modèles d'évaluation de la qualité d'une image proposé dans la présente étude consiste tout d'abord à sélectionner une base de données d'images d'apprentissage, avec des scores subjectifs pour chacune des images présentes dans celle-ci. Un ensemble de caractéristiques est extrait des images présentes dans cette base de données, et à partir de ces caractéristiques, du score subjectif des images et des méthodes d'apprentissage automatiques, se construit un modèle d'évaluation objectif. A titre d'illustration : une image est prise dans une base de données d'images de test, qui peut ne pas être nécessairement la même que la base de données d'apprentissage ; de cette image sont extraites les mêmes caractéristiques que celles extirpées lors de la phase d'apprentissage ; puis ces caractéristiques sont passées en entrée du modèle construit pour obtenir le score objectif de qualité pour l'image examinée.

Sur la base du processus d'évaluation décrit ci-haut, les recherches ont contribué à construire pour l'évaluation de la qualité des images, des modèles d'évaluation objectifs basés sur des métriques avec référence et des modèles sans référence et à distorsion non spécifique c'est-à-dire sans aucune information sur le type de dégradation ayant infecté l'image. La comparaison des résultats obtenus à l'issue des divers modèles d'évaluation étudiés avec ceux décrits dans la littérature, montre que les modèles expérimentés dans le cadre de la présente étude donnent des scores objectifs de qualité qui sont plus corrélés avec les scores subjectifs de différentes bases de données d'images. Une corrélation avec les scores subjectif implique une corrélation avec la qualité visuelle des images. En outre, les modèles construits ont été implémentés sur un système $FPGA$ et l'examen de leur complexité a démontré qu'ils peuvent être intégrés dans des décodeurs vidéos pour une évaluation de la qualité des images projetées.

Le second objectif est la mise en place des algorithmes temps-réel et en ligne de détection et correction des pixels défectueux dans les images. Les algorithmes proposés sont basés sur les distances entre le pixel à tester et ses pixels voisins, ainsi que sur la dispersion dans les différents blocs de pixels dans l'image. Le développement de ces algorithmes a permis d'augmenter le taux de détection des pixels défectueux dans les images. Ils ont aussi permis de réduire le taux de « faux positifs » détectés. La méthode de correction proposée est basée sur les filtres médians qui consiste à remplacer la valeur d'un pixel détecté défectueux par une méthode de détection, par une valeur en accord avec son voisinage. Ces méthodes de correction ont permis d'obtenir des images avec une qualité visuelle très élevée par rapport à celle des images dégradées, voire à celle des méthodes de détection et de correction présentes dans la littérature.

En plus de l'évaluation et de la correction de la qualité des images, la détection des pixels défectueux permet aussi d'évaluer la qualité d'un capteur d'image, de contrôler le vieillissement du capteur. L'opération permet, en outre, d'assurer la fiabilité du système global dans lequel il est utilisé, en fonction du taux de pixels défectueux présents dans les images prises avec ce capteur d'image.

1.4 Organisation

Ce étude est structurée de 4 principaux chapitres.

Chapitre 2 : État de l'art sur l'évaluation de la qualité des images. L'objectif de ce chapitre est de définir les notions de bases sur la qualité des images. Il commence par une étude de quelques dégradations pouvant infecter une image et en altérer la qualité. Un état de l'art sur les méthodes d'évaluation subjectives est fait, dans lequel sont décrits les différents protocoles d'évaluation de la qualité des images à partir des méthodes subjectives. Dans le même temps, est présenté un ensemble de bases de données d'images qui seront utilisées comme références dans la construction des modèles objectifs d'évaluation de la qualité visuelle des images. Puis suit une étude des indices de performances utilisés pour l'évaluation et la comparaison des différentes méthodes étudiées. Enfin sont répertoriées et décrites diverses méthodes automatiques d'apprentissage utilisées pour la construction des modèles expérimentaux d'évaluation de la qualité visuelle des images.

Chapitre 3 : Évaluation de la qualité des images basée sur des métriques avec référence. Ce chapitre est consacré à la construction des mesures globales d'évaluation objective de la qualité visuelle des images, basée sur des métriques avec à référence complète. Le processus d'évaluation passe par la sélection d'un ensemble de métriques qui seront utilisées pour la construction de des divers modèles objectifs. Ensuite est présentée la construction des différents modèles d'évaluation à partir des métriques sélectionnées et des méthodes d'apprentissage automatiques. Les résultats des modèles construits et leur comparaison avec les résultats des modèles existants dans la littérature constituent l'essence de la section sur les résultats expérimentaux de ce chapitre. Il s'en suit une étude des architectures et des résultats de l'implémentation sur un système matériel des modèles obtenus, à savoir le système $FPGA$. En fin de ce chapitre apparaissent le bilan et les différents apports significatifs en ce qui concerne l'évaluation objective de la qualité des images avec référence.

Chapitre 4 : Évaluation de la qualité des images à partir des mesures sans référence. Ce chapitre illustre un aspect plus applicatif et concret dans la démarche suivie. Il présente la construction d'une mesure globale d'évaluation de la qualité des images sans référence. Ici, les caractéristiques sont extraites directement de l'image dégradée sans aucune information relative à l'image de référence. Quatre principaux axes d'extractions sont explorés. Ensuite intervient la sélection d'un ensemble de caractéristiques devant permettre la construction des modèles

d'évaluation à partir des méthodes d'apprentissage automatiques. Les résultats de ces modèles d'évaluation sur différentes bases de données d'images, ainsi que leur comparaison aux résultats produits par des modèles existants, sont aussi étudiés dans ce chapitre. Une autre section de ce chapitre porte sur les architectures et les résultats de la phase d'implémentation sur $FPGA$ des modèles objectifs d'évaluation sans référence construits. Ce chapitre s'achève par le bilan et les apports concernant l'évaluation objective de la qualité des images sans référence.

Chapitre 5 : Détection et correction de pixels défectueux. Ce chapitre porte sur les méthodes de détection et de correction des pixels défectueux dans les images ou les capteurs d'images numériques. Il commence par un état de l'art sur les méthodes existantes dans le domaine de la détection et correction des pixels défectueux dans les images. Puis sont présentées les méthodes proposées en vue de résoudre le problème des pixels défectueux dans les images et capteurs d'images. Les résultats expérimentaux obtenus lors de la phase de simulation sous $MATLAB$ sont présentés, suivis des architectures tout autant que les résultats obtenus lors de la phase d'implémentation sur $FPGA$. Le chapitre s'achève par le bilan ainsi que les apports dans le domaine de la détection et de la correction des pixels défectueux.

Une conclusion générale clôt l'étude, suivie d'une ouverture des perspectives suggérées par le cheminement des travaux.

Chapitre 2

État de l'art sur l'évaluation de la qualité des images

2.1 Introduction

Les images numériques subissent tout au long des chaînes de communication, un ensemble de traitement permettant l'acquisition, la compression, la transmission, le stockage et l'affichage des scènes capturées. Cependant, chacune des étapes de ce processus de communication peut induire des distorsions souvent désignées « artéfacts » dans l'image numérique produite. On distingue une grande variété de types de distorsions qui dégradent la qualité visuelle des images. Parmi ces distorsions il y a lieu de citer : le flou, le bruit, le changement de contraste, la compression $JPEG$ et $JPEG2000$, l'effet de bloc, l'effet de Gibbs (« ringing »), etc.

— Le **flou** est un artéfact qui se manifeste par une perte de la finesse et de la visibilité des détails résultant d'une diminution du contraste. Dans la phase de compression, le flou est dû à l'étape de sélection des coefficients et à une quantification brutale. Dans la phase d'acquisition, il peut être induit par un mouvement de l'objet capturé, ou un mouvement du capteur lors de la prise de vue. Une image déformée par le flou est présentée dans l'image (a) de la Figure 2.1.

— Le **bruit** est une distorsion se matérialisant par la présence d'informations parasites qui s'ajoutent aléatoirement dans une image numérique. Elle a pour conséquence la perte de netteté dans les détails de l'image. On distingue plusieurs types de bruits, qui peuvent impacter soit les couleurs des pixels de l'image, soit le niveau de luminance des pixels. Le bruit peut être dû à l'augmentation de la température du capteur, à la présence d'éléments parasites sur le capteur, à une perte d'information lors de la phase de transmission de l'image, ou encore à des défauts liés aux convertisseurs analogique-numérique des capteurs. L'image (b) de la Figure 2.1 a été déformée par un bruit aléatoire.

— L'**effet de bloc** se caractérise par une apparition de blocs dans l'image. Cette dégradation est due à une quantification indépendante des blocs, produisant ainsi une discontinuité dans le voisinage de ces blocs. Mais elle aussi peut provenir

(a) Exemple de flou (b) Exemple de bruit

(c) Exemple d'effet de bloc (d) Exemple d'effet de Gibbs (Ringing)

FIGURE 2.1 – Images déformées par différents types d'artéfacts

d'un défaut de codage lié à une forte compression de l'image qui, au moment de l'affichage en agrandit la taille des pixels pour compenser la perte d'informations. L'image (*c*) de la Figure 2.1 a été déformée par un effet de bloc.

— L'**effet de Gibbs**, encore appelé effet de « ringing » : il s'agit d'un artéfact associé à l'apparition d'oscillation à proximité des zones à fort contraste. Il est dû à une erreur dans la quantification des coefficients à hautes fréquences. L'image (*d*) de la Figure 2.1 présente un exemple de l'apparition de l'effet de Gibbs dans une image.

L'un des objectifs de ce travail est de pouvoir détecter ces distorsions qui affectent les images, et partant d'évaluer la qualité visuelle celles-ci. Pour évaluer la qualité d'une image, il y a lieu quelques fois d'utiliser des méthodes subjectives, qui font appel à un ensemble d'observateurs humains pour donner une note de qualité aux images dans certaines conditions préalablement définies. Pour une analyse plus raffinée, l'approche par des méthodes objectives s'impose. Ces dernières se basent sur une investigation portant sur un ensemble de caractéristiques extraites des images. Parfois le point d'appui se trouve être le recours à des algorithmes d'apprentissage automatiques susceptibles de, donner un score de qualité aux images. Le processus général de construction des modèles et d'évaluation de la qualité d'une image proposé dans cette étude apparaît dans la Figure 2.2.

Ce chapitre constitue un état de l'art sur l'évaluation de la qualité visuelle des images et sur les méthodes d'apprentissage automatique aidant à la construction de certains

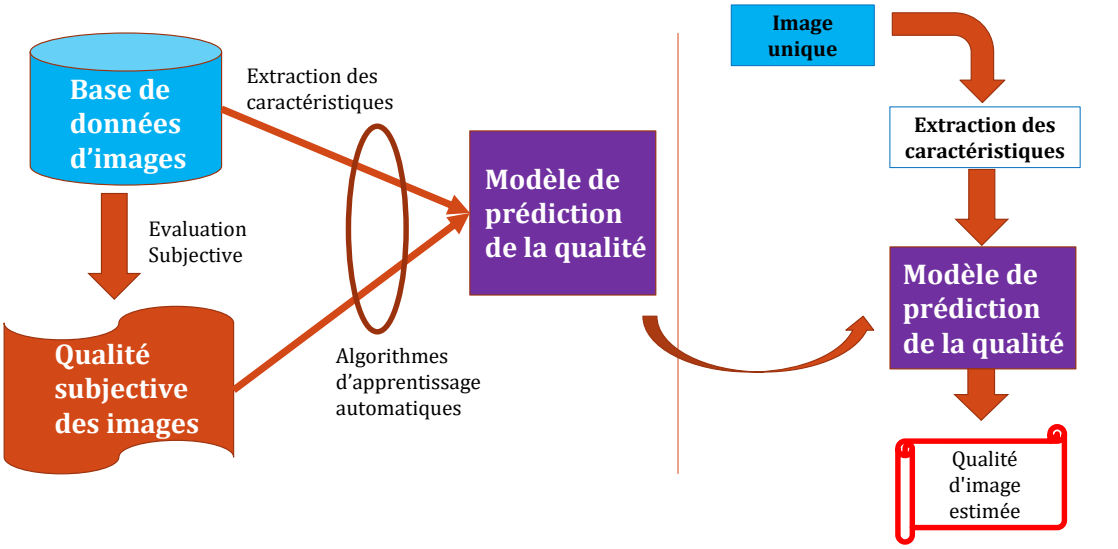


FIGURE 2.2 – Processus d'évaluation objective

index objectifs d'évaluation de la qualité des images. Ainsi la Section 2.2 présente l'évaluation subjective et les différentes bases de données d'images utilisées. La Section 2.3 répertorie les différents indices de performance utilisés pour l'évaluation des méthodes. La Section 2.4 définit et décrit les différentes méthodes d'apprentissage automatique utilisées tout au long de ce travail, et enfin, la Section 2.5 est une conclusion au chapitre.

2.2 Évaluation subjective

Le moyen le plus naturel d'évaluer la qualité visuelle d'une image est de demander à un groupe d'observateurs humains de noter l'image suivant un protocole bien défini ; puis d'effectuer la moyenne des notes données par chacun des observateurs : la note moyenne trouvée est appelée « score moyen d'opinion » (MOS : Mean Opinion Score) ou « score moyen d'opinion différentiel » ($DMOS$: Differential Mean Opinion Score). Cette méthode est appelée méthode d'évaluation subjective du fait que les résultats qu'elle produit sont fortement impactés par la subjectivité du jugement humain. Plusieurs facteurs implicites influençant l'évaluation de la qualité des images par des observateurs humains, peuvent impacter la note finale donnée aux images. Les facteurs examinés ci-après permettent de mieux les distinguer :

— **L'écran d'affichage** : le choix de l'écran d'affichage, de sa taille, de sa calibration, de sa capacité de reproduction des couleurs, de son contraste, et de sa résolution constitue un élément déterminant de l'évaluation de la qualité visuelle des images qui y sont projetées. Ainsi est-il important lors d'une évaluation subjective de la qualité d'avoir toutes les informations sur les écrans d'affichage utilisés, afin de

tenir compte des artéfacts liés aux écrans et à leur calibration [20].

— **La distance d'observation** : la visibilité des images lors de l'évaluation dépend fortement de la distance entre l'observateur et l'écran ; il est recommandé de fixer cette distance entre 4 à 6 fois la hauteur de l'image à projeter, pour permettre une visibilité optimale c'est-à-dire ni très proche, ni très éloigné de l'écran. De plus, une fois fixée, cette distance doit être maintenue tout au long de l'expérience [21].

— **Les conditions de visualisation** : les salles dans lesquelles sont menées les différentes expériences, l'éclairage, ainsi que les couleurs de fond jouent un rôle important dans l'évaluation de la qualité des images. Elles influencent fortement les notes données aux images [21].

— **La durée** : il est préférable de mener des expériences de courte durée pour éviter la fatigue chez l'observateur et partant de fausser les résultats. L'Union Internationale des Télécommunications (ITU) recommande de ne pas faire des séances de plus de 30 minutes, afin d'éviter la fatigue chez l'observateur [22].

— **Les observateurs** : idéalement, le choix des observateurs doit dépendre du domaine d'application visé. Aussi, avant de débuter l'expérience, une phase d'apprentissage doit être faite avec les observateurs pour leur expliquer clairement l'objectif du test ainsi que le protocole utilisé, sans pour autant influencer leur jugement. Il importe aussi d'avoir un nombre suffisant d'observateurs qui participent à l'expérience. Il reste entendu qu'un test n'est statistiquement valide que si l'on a au moins 16 observateurs pour l'expérience [22, 9].

L'évaluation subjective dépend aussi d'autres facteurs tel que l'humeur, l'âge, la culture, le niveau intellectuel, ... Les méthodes d'évaluation subjectives ainsi que le protocole d'évaluation de ces méthodes subjectives diffèrent suivant la base de données d'images considérée. On distingue les protocoles à simple et double stimulus, ainsi que des protocoles à stimulus comparatif.

2.2.1 Protocole d'évaluation

Dans le document des normes définies par l'ITU [22], plusieurs protocoles d'évaluation subjectifs ont été définis en fonction de la manière dont les images sont présentées à l'observateur pour l'évaluation tout autant qu'en fonction des échelles de notes d'évaluation proposées. Ces protocoles peuvent être répertoriés en trois grands groupes : les protocoles à simple stimulus, les protocoles à double stimulus, et les protocoles à stimulus comparatifs.

1. **Le protocole d'évaluation à simple stimulus** : encore appelé « Single-Stimulus Continuous Quality-Scale » ($SSCQS$), il permet de juger de la qualité d'une image dégradée par un stimulus unique, comme présenté dans la Figure 2.3. Le processus d'évaluation est le suivant : une image s'affiche, puis l'observateur a un temps de latence pour donner une note à cette image en fonction de l'échelle des notes proposée dans l'expérience, et ensuite on passe à l'image suivante. Des échelles à 5, 6, 10 ou même 100 niveaux peuvent être utilisées. Le Tableau 2.1 présente un exemple d'échelle à 5 niveaux. La limitation de cette méthode à simple stimulus est qu'elle

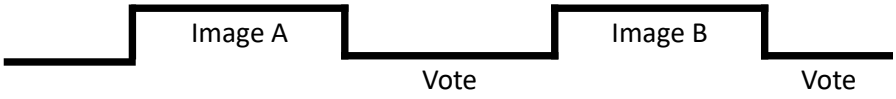


FIGURE 2.3 – Méthode à simple stimulus

TABLE 2.1 – Échelle d'évaluation à 5 niveaux pour simple stimulus

Excellente
Bonne
Moyenne
Mauvaise
Très mauvaise

nécessite une grande expertise et un apprentissage très poussé de la part des observateurs, pour permettre une évaluation optimale. De plus, les images dégradées ne sont pas contextualisées, et dès lors, les résultats peuvent varier énormément d'un observateur à un autre.

2. **Le protocole d'évaluation à double stimulus** : également appelé « Double-Stimulus Continuous Quality-Scale » ($DSCQS$), cette méthode d'évaluation a pour objectif de mesurer la qualité d'une image dégradée par rapport à sa version originale ou encore dite de référence, comme présenté dans la Figure 2.4. Dans cette méthode, l'image de référence est tout d'abord affichée, puis un écran gris, ensuite la version dégradée est affichée, et enfin on laisse un temps de latence à l'observateur pour qu'il puisse noter l'image dégradée. Peu après le processus recommence. Le temps d'affichage de l'écran gris et le temps de vote doivent être identiques pour tous les stimuli. L'ordre d'affichage entre l'image de référence et l'image dégradée peut être modifié, et on peut aussi afficher deux images originales, ou même deux images dégradées. Différentes échelles peuvent être utilisées pour cette évaluation. Le Tableau 2.2 présente un exemple d'échelle à 5 niveaux pour une évaluation à double stimulus. Cette méthode permet de résoudre les problèmes liés au contexte de l'image dégradée, en affichant une image de référence pour permettre à l'observateur de jauger du niveau de dégradation.

3. **Le protocole d'évaluation à stimulus comparatif** : les méthodes comparatives permettent d'évaluer la qualité d'une image en fonction d'une ou plusieurs autres images, venant toutes de la même image de référence. Les images sont comparées entre elles comme présenté dans la Figure 2.5. Le but dans cette méthode est de

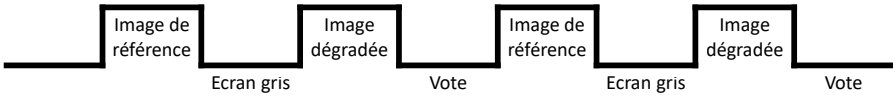


FIGURE 2.4 – Méthode à double stimulus

TABLE 2.2 – Échelle d'évaluation à 5 niveaux pour double stimulus

Note	**Qualité**	**Dégradation**
5	Excellente	Imperceptible
4	Bonne	Perceptible
3	Moyenne	Peu dégradée
2	Mauvaise	Dégradée
1	Très mauvaise	Très dégradée

déterminer l'image qui a la meilleure ou à contrario la moins bonne qualité visuelle, parmi les celles qui ont été présentées. Dans cette méthode, on qualifie la perception visuelle entre les différentes images du même stimulus. Aucune information liée à la qualité absolue d'une image n'est donnée. Tout au plus pourrait être évoquée la qualité relative de l'image par rapport à une ou plusieurs autres. Ainsi, une image de très mauvaise qualité, peut être jugée comme étant la meilleure dans un stimulus, si elle est comparée avec d'autres images de plus mauvaise qualité qu'elle. Le Tableau 2.3 présente un exemple d'échelle de notation pour cette méthode comparative.

A la fin des expériences, après avoir normalisé les résultats obtenus et supprimé les valeurs aberrantes, le score subjectif moyen (MOS) de l'image est évalué à partir des notes données par tous les observateurs qui ont évalué cette image. Ce score subjectif se calcule comme dans (2.1).

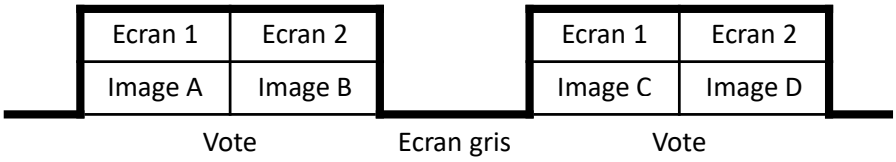


FIGURE 2.5 – Méthode à stimulus comparatif

TABLE 2.3 – Échelle d'évaluation à 5 niveaux pour un stimulus comparatif

Beaucoup moins bon
Moins bon
Identique
Mieux
Largement mieux

FIGURE 2.6 – Quelques images de référence de la base de données LIVE

$$MOS(i) = \frac{1}{N_{obs}} \sum_{j=1}^{N_{ons}} Note_i(j) \tag{2.1}$$

où N_{obs} est le nombre total d'observateurs ayant évalué l'image, et $Note_i(j)$ est la note donnée à l'image i par l'observateur j.

2.2.2 Bases de données d'images

Ces dernières années, plusieurs bases d'images avec des protocoles d'évaluations subjectifs différents ont vu le jour. Trois de ces bases de données d'images ont été retenues dans la présente étude. Pour chacune d'elles sont décrites les images de référence utilisées, les dégradations considérées, le protocole d'évaluation subjectif utilisé ainsi que les données mises à disposition des utilisateurs de la base de données (MOS, notes par observateur, ...).

TABLE 2.4 – Types de distorsions dans la base de données Live

N°	Type de distorsion
1	Compression JPEG2000
2	Compression JPEG
3	Bruit blanc (White Noise : WN)
4	Flou gaussien (Gaussian Blur : GB)
5	Décoloration ou évanouissement rapide de Rayleigh (simulated Fast Fading Rayleigh channel : FF)

2.2.2.1 Base de données d'images LIVE

Il s'agit ici de la version 2 de la base de données d'images LIVE[10, 23, 24] parue en 2006, et construite par le laboratoire d'ingénierie de l'image et de la vidéo de l'université du Texas à Austin [1]. Cette base de données est construite à partir de 29 images de référence prises sur internet ou dans des CD-ROM. Ces images de référence incluent des images de faces, de personnes, d'animaux, de gros plans, de scènes naturelles, d'objets fabriqués, etc.... Certaines de ces images de référence sont présentées dans la Figure 2.6. Ces images ont été redimensionnées en utilisant l'interpolation bicubique de l'outil $MATLAB$. Les images utilisées dans cette base de données sont de tailles différentes, mais la plupart ont une résolution de 768×512 pixels.

Cinq (05) types de distorsions (dégradations) présentés dans le Tableau 2.4 ont été injectés dans les 29 images de référence de la base de données LIVE, avec plusieurs niveaux par dégradation, pour former les 779 images déformées que compte cette base de données d'images. Toutes ces images contenues dans la base de données LIVE sont stockées au format Bitmap ($.BMP$).

Les 779 images déformées résultant de l'opération ont été évaluées par deux douzaines d'observateurs humains, issus du département de traitement d'images et de vidéos de l'Université du Texas à Austin. Chacune des images déformées a été évaluée en moyenne 23 fois, ce qui donne au final près de 25 000 jugements humains de la qualité des images.

Les évaluations de la qualité des images par les observateurs humains sont effectuées sur des postes identiques, avec des écrans de 21 pouces chacun, ayant une résolution de 1024×768, et presque tous ayant le même âge. Chaque session dure au maximum 30 minutes pour éviter la fatigue.

La méthodologie d'évaluation consiste à projeter les images sur un écran, un être humain doit ranger l'image projetée dans une des catégories suivantes : « excellent », « bon », « moyen », « mauvais » et « très mauvais » : protocole d'évaluation à simple stimulus avec une échelle à cinq niveaux. A la fin de la session, les différentes catégories sont converties en valeurs numériques entre 1 et 100.

1. LIVE : Laboratory for Image and Video Engineering

FIGURE 2.7 – Images de référence de la base de données CSIQ

Une fois l'expérience terminée, les données brutes extraites des différentes sessions d'évaluation subjectives sont traitées et combinées entre elles. Après l'exclusion des valeurs aberrantes [2], la qualité d'image est définie par son score moyen d'opinions différentiel ($DMOS$). Le $DMOS$ d'une image est évalué comme la moyenne des différents scores obtenus par cette image. Dans la base de données d'images LIVE, le $DMOS$ est compris entre 0 et 100, avec 0 dénotant une qualité d'image excellente, et 100 une très mauvaise qualité d'image.

2.2.2.2 Base de données d'images CSIQ

CSIQ (Categorical Subjective Image Quality) [25, 26] est une base de données d'images conçue par le laboratoire d'analyse de la qualité et de perception d'images, de l'Université de l'État d'Oklahoma [3] ; parue en 2010. Elle est constituée de 30 images de référence, prises pour la plupart au service du parc national des États-Unis. Ces images ont été choisies pour couvrir cinq catégories : animaux, paysages, personnes, plantes et scènes naturelles. Les images de référence de cette base de données sont présentées dans la Figure 2.7.

Chaque image de référence est déformée grâce à six (06) types de distorsions présentés dans le Tableau 2.5, avec quatre (04) ou cinq (05) niveaux pour chaque type de distorsion. Soit un total de 866 images déformées résultant de l'opération. Elles sont toutes stockées au format « Portable Network Graphics » ($.PNG$), et redimensionnées en une résolution de 512×512 pixels.

2. Une valeur hors de l'intervalle de largeur $\triangle$ de l'écart type autour de la valeur moyenne de l'image
3. Computational Perception and Image Quality Lab

TABLE 2.5 – Types de distorsions dans la base de données CSIQ

N°	Type de distorsion
1	Compression JPEG2000
2	Compression JPEG
3	Changement de contraste global (global contrast decrements)
4	Bruit gaussien rose (additive pink Gaussian noise)
5	Bruit gaussien blanc (additive white Gaussian noise)
6	Flou gaussien (Gaussian blurring)

Les images déformées sont évaluées subjectivement en fonction d'un déplacement linéaire des images sur quatre (04) moniteurs LCD identiquement calibrés, de résolution 1920×1200, placés côte à côte et à distance d'observation égale (70 cm) de l'observateur. Les toutes les images s'affichant simultanément sur quatre moniteurs doivent être des images déformées d'une même image de référence. Toutes les images déformées d'une image de référence doivent apparaitre au moins une fois dans le même stimulus d'évaluation. Ainsi, toutes les images déformées d'une même image de référence seront placées les unes par rapport aux autres en fonction de leur qualité visuelle. La qualité de chaque image est déterminée en fonction des différents rangs obtenus au cours l'expérience. 5 000 évaluations subjectives ont été effectuées, par 35 observateurs humains.

La qualité d'une image dans cette base de données $CSIQ$ est décrite par son score moyen d'opinions différentiel ($DMOS$) obtenu en évaluant la moyenne des scores subjectifs obtenus par cette image durant l'expérience. Les DMOS sont tous compris entre 0 et 1, où 0 dénote une excellent qualité d'image et 1 la qualité d'image moins bonne.

2.2.2.3 Base de données d'images TID2013

La base de données d'images TID2013[27, 4] est une version évoluée parue en 2013 de la base de données d'images TID2008[28, 29], destinées à l'évaluation des mesures de la qualité visuelle. Elles ont été toutes les deux mises au point au sein de l'Université Technologique de Tampere [4] en Finlande. TID2013 contient 25 images de référence, dont 24 ont été prises de la base de données d'images Kodak [30], et la 25éme image a été construite artificiellement. Les images de référence de cette base de données sont présentées dans la Figure 2.8. Certains contenus de ces images ont des textures différentes, tandis que d'autres sont homogènes. Toutes les images de cette base de données ont été redimensionnées, elles ont toutes la même résolution qui est de 512×384 pixels, et sont stockées au format Bitmap ($.BMP$) sans aucune compression.

Toutes les images de référence de la base de données $TID2013$ ont été déformées à partir de 24 types de distorsions décrites dans le Tableau 2.6, avec cinq (05) niveaux

4. Tampere University of Technology

TABLE 2.6 – Types de distorsions utilisées dans la base de données TID2013

N°	Type de distorsion
1	Additive white Gaussian noise
2	Additive noise in color components is more intensive than additive noise in the luminance component
3	Spatially correlated noise
4	Masked noise
5	High frequency noise
6	Impulse noise
7	Quantization noise
8	Gaussian blur
9	Image denoising
10	JPEG compression
11	JPEG2000 compression
12	JPEG transmission errors
13	JPEG2000 transmission errors
14	Non eccentricity pattern noise
15	Local block-wise distortions of different intensity
16	Mean shift (intensity shift)
17	Contrast change
18	Change of color saturation
19	Multiplicative Gaussian noise
20	Comfort noise
21	Lossy compression of noisy images
22	Image color quantization with dither
23	Chromatic aberrations
24	Sparse sampling and reconstruction

FIGURE 2.8 – Images de référence de la base de données TID2013

pour chaque distorsion ; soit un total de 3000 images déformées (25 images de référence $\times$24 types de distorsions $\times$5 niveaux par distorsion).

Au total 524 340 évaluations de la qualité visuelle d'images ont été effectuées, découlant de 971 expériences réalisées avec des observateurs humains bénévoles issus de cinq (05) pays différents (Finlande, France, Italie, Ukraine et États-Unis). Chacune des expériences dure en moyen 17 minutes (ce qui est $<$ 30 minutes, comme recommandées par l'ITU), soit en moyenne 540 évaluations par observateurs. Dans le processus d'évaluation, une paire d'images toutes deux déformées provenant de la même image de référence, apparaît sur la partie haute de l'écran, tandis que sur la partie basse de ce même écran, apparaît l'image de référence à partir de laquelle ont été construites les images déformées à évaluer ; comme présenté dans la Figure 2.9. Il est demandé à l'observateur de choisir la meilleure [5] des 2 images déformées qui lui sont présentées. Après la sélection, de nouvelles images déformées apparaissent, et le processus d'évaluation recommence. La décision de l'observateur doit être claire et rapide à savoir : 03 secondes pour décider, sinon le test passe aux images suivantes. Chaque observateur n'évalue que les images déformées d'une même image de référence. Chaque image déformée participe au même nombre de comparaisons par paire (soit 09 comparaisons) ; et lorsqu'une image est choisie lors de la comparaison, elle gagne un point. A la fin de l'expérience, les points donnés à chacune des images déformées sont sommés pour déterminer le score final de l'expérience pour de cette image. Au début de l'expérience, les images apparaissant dans les évaluations par paire sont sélectionnées aléatoirement ; mais au bout d'un certain nombre d'expériences aléatoires, les images déformées ayant approximativement le même score sont comparées entre elles.

5. Il s'agit ici de l'image la moins déformée par rapport à l'image de référence.

FIGURE 2.9 – Capture d'écran de la méthodologie d'évaluation de la base TID2013

TABLE 2.7 – Tableau récapitulation des bases de données d'images

BD	*Date*	*Protocole*	N_{obs}	N_{exp}	$N_{réf}$	$N_{dég}$	$N_{déf}$	*Échelle*
LIVE	2006	SSCQS	24	25 000	29	5	779	0 - 100
CSIQ	2010	Comparatif	35	5 000	30	6	866	0 - 1
TID2013	2013	Comparatif	971	524 340	25	24	3 000	0 - 9

Les résultats bruts obtenus lors des expériences de comparaison sont sauvegardés, et les données aberrantes sont exclues de l'étude. Les données restantes sont combinées entre elles pour évaluer le score d'opinions moyen (MOS) des observateurs. Pour la base de données d'image TID2013, le MOS est compris entre 0 et 9, cependant il ne dépasse pas 7.5 pour les images déformées étudiées dans cette base de données. Plus l'image est bonne, plus son score subjectif MOS est élevé.

2.2.3 Récapitulatif sur les bases de données d'images

Le Tableau 2.7 présente un récapitulatif de la présentation des bases de données d'images LIVE, CSIQ et TID2013. Ce tableau donne pour chacune des bases de données étudiées, la date de parution ($Date$), le protocole d'évaluation subjectif utilisé ($Protocole$), le nombre d'observateurs (N_{obs}), le nombre d'expériences subjectives effectuées (N_{exp}), le nombre d'images de référence ($N_{réf}$), le nombre de dégradations utilisées ($N_{dég}$), le nombre d'images déformées ($N_{déf}$), et enfin l'échelle d'évaluation ($Échelle$) des scores moyens donnés par les observateurs (MOS / $DMOS$).

Tous ces tests nécessitent un matériel et des outils informatiques spécifiques (salle

dédiée, environnement, éclairage, écran, sonde de calibration, observateurs, ...), rendant ainsi l'évaluation subjective lourde, complexe et difficilement reproductible. Ce sont toutes ces raisons qui ont conduit à l'élaboration de mesures objectives permettant d'évaluer automatiquement la qualité des images.

2.3 Indices de performance

Pour évaluer les performances des méthodes présentées, plusieurs indices de performance ont été utilisés, il s'agit principalement de : l'erreur quadratique moyenne (MSE), le coefficient de détermination (R^2), le coefficient de corrélation linéaire de Pearson ($PLCC$), le coefficient de corrélation de rang de Spearman ($SRCC$), le coefficient de rang Kendall ou tau de Kendall ($KRCC$), et la distance de corrélation de Brownian ($dCor$).

2.3.1 Erreur quadratique moyenne et coefficient de détermination

Dans une estimation de variable aléatoire, il existe presque toujours un écart entre la valeur estimée par l'outil de mesure et la valeur réelle. Cet écart entre la valeur prédite et la valeur réelle d'une mesure est appelée « erreur ». L'erreur quadratique moyenne est donc la moyenne des carrés des écarts (ou résidus), entre les valeurs réelles et les valeurs prédites à partir d'un estimateur. Les écarts sont évalués au carré pour majorer l'importance des grosses erreurs. L'objectif de toute méthode d'estimation est d'annuler cette erreur quadratique, mais en général, elle n'est pas nulle, alors il faut la minimiser. La formule de calcul de l'erreur quadratique moyenne (MSE) entre deux variables X (x_i) et Y(y_i) est présentée dans (2.2) [31]. Où la variable X représente le vecteur des données mesurées (réelles) et la variable Y représente le vecteur des données prédites par une méthode.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{2.2}$$

Le coefficient de détermination (R^2) quant à lui, mesure l'adéquation entre un modèle issu d'une régression, et les données observées qui ont permis de l'établir. Plus simplement, il est une mesure de la qualité de la prédiction d'une régression : la part de variance expliquée dans la variance totale [32]. Le coefficient de détermination se définit comme le rapport entre la variance expliquée et la variance totale, comme présenté dans (2.3).

$$R^2 = 1 - \frac{SCR}{SCT} = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.3}$$

où SCR est la somme des carrés des résidus, et SCT la somme des carrés totale (somme des carrés des différences entre la valeur réelle d'une expérience et la valeur moyenne des valeurs réelles). Avec y_i représentant les valeurs des mesurées, $\widehat{y_i}$ les valeurs prédites et $\bar{y}$ la moyenne des valeurs mesurées.

Le coefficient de détermination est toujours compris entre 0 et 1, avec par exemple un coefficient de $0,8$ indiquant que 80% de la dispersion est expliquée par le modèle de régression.

2.3.2 Coefficients de corrélation

Les coefficients de corrélation sont utilisés pour mesurer le degré de liaison entre deux ou plusieurs variables aléatoires. Généralement, le coefficient de corrélation suppose une valeur située entre -1 et +1. Pour ce qui est de deux variables aléatoires, si une variable tend à augmenter tandis que l'autre diminue, le coefficient de corrélation est négatif ; inversement, si les deux variables évoluent dans le même sens, le coefficient de corrélation est positif. Un coefficient de corrélation tendant vers 1 en valeur absolue, indique une liaison forte entre les variables aléatoires, tandis qu'un coefficient de corrélation tendant vers 0 indique que les variables aléatoires n'ont pas de liaison suivant l'axe étudié [33]. Un problème majeur des coefficients de corrélation reste leur interprétation, c'est-à-dire de définir à partir de quel niveau il y a lieu de dire que les variables sont dépendantes ou indépendantes entre elles. Pour cela, on utilise un test non paramétrique, dépendant du nombre d'observations dans l'échantillon permettant de définir le degré de liberté, ainsi que du niveau de confiance qu'on se donne. Il s'agit, pour les coefficients de corrélation étudiés dans cette partie, du « test statistique de Student ».

Le **coefficient de corrélation linéaire de Pearson** ($PLCC$) aussi noté « r » permet de détecter la présence ou l'absence d'une relation linéaire entre deux variables aléatoires [34]. Le coefficient de corrélation linéaire de Pearson entre deux variables aléatoires réelles X et Y ayant chacune une variance non nulle, sur un échantillon de taille n est donné par (2.4).

$$PLCC = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)\sigma_x\sigma_y} \tag{2.4}$$

où σ_x et σ_y désignent les écarts types des variables X et Y, respectivement. Ces écarts-types sont définis par $\sigma_x = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$. Et $\overline{x}$ est la moyenne de variables X sur un échantillon, définie par $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, idem pour $\overline{y}$ pour la variable Y.

Le **coefficient de corrélation de rang de Spearman** ($SRCC$) ou rho (ρ) de Spearman est une mesure de dépendance statistique non paramétrique entre deux variables. La corrélation de Spearman est étudiée lorsque deux variables statistiques semblent corrélées sans que la relation entre les deux variables soit de type affine [35]. Elle consiste à trouver un coefficient de corrélation, non pas entre les valeurs prises par les deux variables mais entre les rangs de ces valeurs. Elle estime à quel point la relation entre deux variables peut être décrite par une fonction monotone. S'il n'y a pas de données répétées, une corrélation de Spearman parfaite de +1 ou -1 est obtenue quand l'une des variables est une fonction monotone parfaite de l'autre. Le coefficient de Spearman est aussi un cas particulier du coefficient de Pearson, calculé à partir des transformations des variables originelles [36]. Mais il présente l'avantage d'être non paramétrique. L'idée est de substituer aux valeurs observées leurs rangs. Dans une observation sur un échantillon de taille

n, la plus petite valeur de la variable sur l'échantillon prend le rang 1, tandis que la plus grande prend le rang n. Il faut donc définir deux nouvelles variables, $R_i = Rang(x_i)$, correspondant au rang de l'observation x_i pour la variable X ; et $S_i = Rang(y_i)$ pour la variable Y. Le coefficient de corrélation de Spearman est défini par (2.5).

$$SRCC = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \tag{2.5}$$

où $d_i$est l'écart entre les rangs, $d_i = R_i - S_i$; et n la taille de l'échantillon.

Le **coefficient de corrélation de rang de Kendall** ($KRCC$) ou encore tau (τ) de Kendall est défini pour mesurer l'association entre des variables ordinales, typiquement à partir des classements (ou rangs) affectés par des juges [37, 38]. Il repose sur la notion de paires concordantes et discordantes. Les paires observations i et j sont dites concordantes si et seulement si : « si $x_i > x_j$ alors $y_i > y_j$ », ou « si $x_i < x_j$ alors $y_i < y_j$ » ; ce qui équivaut à $(x_i - x_j) \times (y_i - y_j) > 0$. De même, les paires i et j sont dites discordantes si et seulement si : « si $x_i > x_j$ alors $y_i < y_j$ », ou « si $x_i < x_j$ alors $y_i > y_j$ » ; en d'autre termes $(x_i - x_j) \times (y_i - y_j) < 0$. Le coefficient de Kendall (τ de Kendall) se définit comme le rapport de la différence entre le nombre de paires concordantes et le nombre de paires discordantes, sur le nombre total de paires. Cela se traduit par l'équation (2.6).

$$KRCC = \frac{C - D}{n(n-1)/2} \tag{2.6}$$

où C est le nombre de paires concordantes, D est le nombre de paires discordantes, et n la taille de l'échantillon (le nombre total de paires est la combinaison de 2 dans n).

Le τ de Kendall s'interprète comme le degré de correspondance entre 2 classements. Si toutes les paires sont concordantes, le classement selon X concorde systématiquement avec le classement selon Y alors $\tau = 1$; si toutes les paires sont discordantes alors $\tau = -1$; enfin, si les deux classements sont totalement indépendants, équilibre entre le nombre de paires concordantes et de paires discordantes, $\tau = 0$ [36].

L'interprétation de ces différents coefficients de corrélation (noté ici r_c) se fait à partir du **test de Student**. Ce test porte sur les différents coefficients de corrélation étudiés dans cette section, pour n observations. Les hypothèses testées sont les suivantes :

H_0 **:** l'absence de liaison entre les variables aléatoires X et Y ; $r_c = 0$.

H_1**(bilatérale) :** il existe une liaison entre les variables aléatoires X et Y ; $r_c \neq 0$

H_1**(unilatérale) :** il existe une liaison positive entre les variables aléatoires X et Y, $r_c > 0$; ou il existe une liaison négative entre les variables aléatoires X et Y, $r_c < 0$

La statistique de ce test sur les coefficients de corrélation, qui est la loi sous l'hypothèse nulle H_0 est donnée par l'équation (2.7).

$$t_{(ddl=n-2)obs} = \frac{|r_c|}{\sqrt{\frac{(1-r_c^2)}{n-2}}} \tag{2.7}$$

où ddl est le degré de liberté du test, qui est proportionnel à la taille de l'échantillon n. Et le résultat du test s'interprète comme suit, étant donné un degré de liberté $ddl = n-2$, et un niveau de confiance $1 - \alpha$:

— $t_{obs} < t_{ddl,\alpha}$: H_0 est vraie, d'où l'absence de liaison entre les variables aléatoires X et Y ;
— $t_{obs} \geq t_{ddl,\alpha}$: H_0 est fausse, d'où une liaison positive ou négative entre les variables aléatoires X et Y, avec un niveau de confiance de $1 - \alpha$.

2.3.3 Distance de Brownian

La distance de Brownian ($dCor$) est un coefficient de corrélation multiple, qui permet de mesurer la dépendance statistique entre deux variables (vecteurs) de dimensions arbitraires et donc pas nécessairement égales. Cette distance est égale à zéro si et seulement si les variables sont statistiquement indépendantes, contrairement à la corrélation linéaire qui peut être égale à zéro même pour des variables dépendantes. Il s'agit donc d'un coefficient de corrélation non linéaire (qui prend en compte les non-linéarités entre les variables) [39, 40].

L'évaluation de la distance de Brownian ($dCor$) entre deux (02) variables aléatoires $X \in \mathbb{R}^p$ et $Y \in \mathbb{R}^q$ passe par la construction des matrices de distances euclidiennes $(a_{kl}) = (|X_k - X_l|_p)$ et $(b_{kl}) = (|Y_k - Y_l|_q)$, définies sur un échantillon de taille n, tel que $(X, Y) = \{(X_k, Y_k) : k = 1, ..., n\}$, par l'équation (2.8) [40].

$$A_{kl} = a_{kl} - \bar{a}_{k.} - \bar{a}_{.l} - \bar{a}_{..} \tag{2.8}$$

où les moyennes $\bar{a}_{k.}$, $\bar{a}_{.l}$, et $\bar{a}_{..}$ sont définies dans (2.9) .

$$\begin{array}{c} \bar{a}_{k.} = \frac{1}{n} \sum_{l=1}^{n} a_{kl} \\ \bar{a}_{.l} = \frac{1}{n} \sum_{k=1}^{n} a_{kl} \\ \bar{a}_{..} = \frac{1}{n^2} \sum_{k,l=1}^{n} a_{kl} \end{array} \tag{2.9}$$

La covariance de distance de Brownian ($dCov$) entre les deux variables aléatoires X et Y sur un échantillon de taille n, encore noté $\nu_n(X, Y)$ est alors définie par (2.10) ; et le coefficient de corrélation de distance de Brownian équivalent entre X et Y sur un échantillon de taille n, encore noté $\Re_n(X, Y)$ est donné par (2.11).

$$\nu_n^2(X, Y) = dCov_n^2(X, Y) = \frac{1}{n^2} \sum_{k,l=1}^{n} A_{kl} B_{kl} \tag{2.10}$$

$$\Re_n^2(X, Y) = dCor_n^2(X, Y) = \begin{cases} \frac{\nu_n^2(X,Y)}{\sqrt{\nu_n^2(X)\nu_n^2(Y)}} & \nu_n^2(X)\nu_n^2(Y) > 0 \\ 0 & \nu_n^2(X)\nu_n^2(Y) = 0 \end{cases} \tag{2.11}$$

où $\nu_n(X)$ est la variance des distances de la variable aléatoire X, défini par (2.12).

$$\nu_n^2(X) = \frac{1}{n^2} \sum_{k,l=1}^{n} A_{kl}^2 \tag{2.12}$$

La distance de Brownian est toujours comprise entre 0 et 1 ; avec $dCor = 0$ si et seulement si $X \in \mathbb{R}^p$ et $Y \in \mathbb{R}^q$ sont indépendants, et $dCor$ proche de 1 implique une

corrélation forte entre les variables aléatoires X et Y. Il s'agit donc d'un coefficient de corrélation non-linéaire et multiple [40].

2.3.4 Validation croisée

La validation croisée (cross-validation) est une méthode d'estimation de la fiabilité des modèles générés par une méthode d'apprentissage automatique. La validation croisée a aussi pour objectif de trouver les paramètres optimaux et les données d'apprentissage optimales permettant la construction du meilleur modèle d'évaluation possible. Le modèle construit doit tout autant donner de bons résultats lors de la phase d'apprentissage, que lors de la phase de test, tout en évitant des problèmes de « sur-apprentissage ». La validation croisée permet aussi de prédire l'efficacité d'un modèle sur un ensemble de validation hypothétique, lorsqu'un ensemble de validation indépendant et explicite n'est pas disponible [41].

Dans un apprentissage automatique, l'idée est de diviser le jeu de données en deux sous-ensembles : un jeu d'entraînement, et un jeu de test. Seul le jeu de données d'entraînement est utilisé dans la phase idoine et de construction des modèles d'évaluation. Ainsi peuvent se calculer les performances des modèles construits sur le jeu de test. Ce résultat est une bonne approximation de la performance car il s'effectue sur des données inconnues c'est-à-dire non utilisées lors de la phase d'apprentissage ou de construction des modèles. En fonction de la répartition des données d'entraînement et de test, il existe plusieurs variantes de validations croisées :

1. **La validation croisée par stratification ou « k-fold cross-validation »** [42] : dans cette méthode de validation croisée, on divise l'échantillon original en k échantillons (ou parties), puis on sélectionne l'un des k échantillons comme ensemble de validation et les $k-1$ autres échantillons constitueront l'ensemble d'apprentissage. L'apprentissage s'effectue sur l'ensemble des $k-1$ échantillons, puis la validation se fait sur l'échantillon sélectionné. On calcule ensuite le score de performance, puis on répète l'opération en sélectionnant un autre échantillon de validation parmi les $k-1$ échantillons qui n'ont pas encore été utilisés pour la validation du modèle. L'opération se répète ainsi k fois pour qu'en fin de compte chaque sous-échantillon ait été utilisé exactement une fois comme ensemble de validation. La moyenne des k erreurs quadratiques moyennes et celle des scores de performances sont calculées pour estimer l'erreur et la performance de prédiction lors de la phase de validation croisée.
2. **La méthode « Leave-one-out cross-validation »** ($LOOCV$) [43] : cette méthode de validation croisée est juste un cas particulier de la méthode précédente, dite par stratification, où $k = n$. Dans cette méthode LOOCV, on apprend sur $n-1$ observations, puis la validation du modèle est faite sur la $n-ième$ observation qui n'a pas été utilisée pour l'apprentissage du modèle. Cette opération est répétée n fois de sorte que toutes les observations soient utilisées pour la validation.
3. **La validation croisée de Monte Carlo ou « Monte Carlo cross-validation »** ($MCCV$) [44] : dans la méthode de validation croisée de Monte Carlo, on exécute

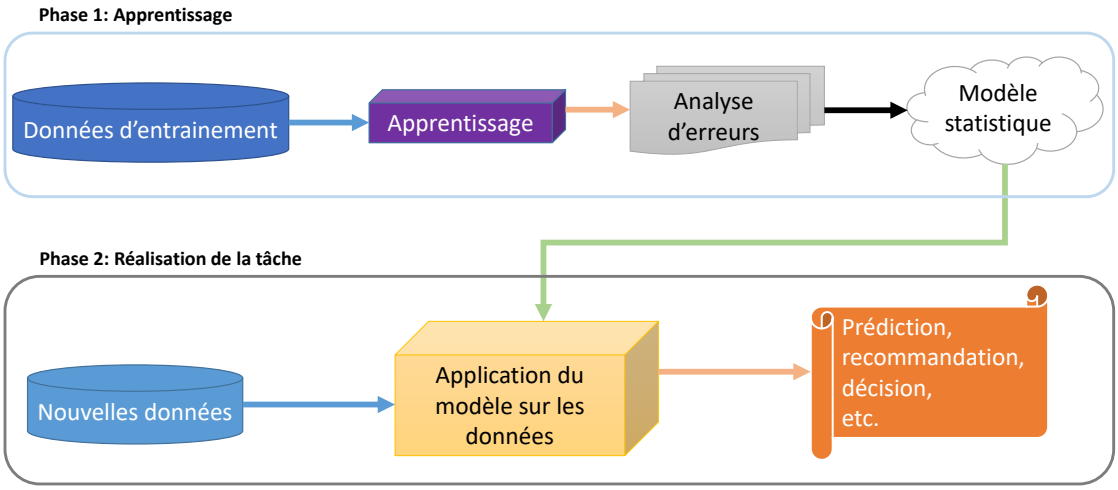


FIGURE 2.10 – Phase du processus d'apprentissage automatique

k fois les phases d'apprentissage et de test, en répartissant les données aléatoirement pour chacune de ces phases. Cette phase de validation permet de tester la robustesse et la précision des différentes méthodes. Le processus de validation est le suivant : pour commencer, on divise aléatoirement la base de données en deux parties : les données d'entraînement, représentant $\alpha\%$ de l'échantillon total, et les données de test constituées du reste de données de l'échantillon , soit $(100 - \alpha)\%$ de l'échantillon total. Ensuite, on construit les modèles à partir des données d'apprentissage, et on évalue les modèles construits à partir des données de test. On obtient ainsi les scores de performances pour chacune des méthodes. Enfin, les 2 phases précédentes sont répétées $k - fois$; les scores et l'erreur de prédiction de la méthode sont calculés comme la moyenne des scores et des erreurs des k scores obtenus à chaque exécution.

Toutes ces méthodes de validation croisée sont basées sur un même principe qui est la répétition des phases d'entraînement et de test, en changeant à chaque fois la répartition entre les données d'entraînement et de test. Ces méthodes de validation permettent de vérifier que les performances produites par une méthode ne sont pas liées à la répartition entre les données utilisées lors de la phase d'apprentissage et celles utilisées lors de la phase de test.

Après l'étude des indices de performance, la prochaine section de ce chapitre permet de faire une brève introduction sur les méthodes d'apprentissage utilisées tout au long de ce travail.

2.4 Méthodes d'apprentissage automatique

L'apprentissage automatique (en anglais Machine Learning : ML) peut se définir comme étant la construction d'un modèle d'évaluation d'une mesure par un algorithme.

Cette construction est faite grâce à des données d'entraînement, pour accomplir une tâche, en ajustant les paramètres de l'algorithme d'apprentissage en fonction de la mesure des performances [45]. Un problème d'apprentissage automatique peut donc se diviser en quatre éléments spécifiques : les données, une tâche à accomplir, la mesure des performances, et un algorithme d'apprentissage. Les différentes étapes qui interviennent dans un apprentissage automatique sont décrites dans le schéma de la Figure 2.10.

1. **Les données** peuvent être de plusieurs types : des bases de données, des données brutes, des images, des vidéos, des informations venant des capteurs, et même du texte tout simplement. La quantité et la qualité des données utilisées pour une méthode d'apprentissage automatique influencent sur la qualité de la prédiction. Il faut donc faire attention au nombre de données car plus il y a des données mieux cela affermit l'apprentissage. Cependant, l'on ne saurait aussi négliger la qualité des données et leurs attributs, comme le taux de bonnes données, le bruit dans les données, les valeurs aberrantes, pour ne citer que ceux-là.
2. **La tâche spécifique à accomplir** correspond au problème qu'on cherche à résoudre grâce à la modélisation du phénomène. Là aussi, les tâches à accomplir varient suivant les domaines. On peut avoir des tâches telles que les recommandations de produits, l'identification de transactions frauduleuses, la prédiction de l'impact d'une campagne marketing sur le taux de conversion, ou même la prédiction de la qualité d'un produit. Chaque tâche se traduira différemment et nécessitera, à coup sûr, un choix d'algorithmes d'apprentissage différents.
3. **La mesure des performances** d'un modèle constitue l'une des étapes clés du processus d'apprentissage. Elle indique l'efficacité de la notation (prédiction) d'un jeu de données par un modèle construit par une méthode d'apprentissage. Cette mesure des performances permet ainsi de dire si le modèle est correct et s'il permet de réaliser correctement les tâches pour lesquelles il a été créé. Lors de l'évaluation des performances, on injecte de nouvelle données dans les modèles d'apprentissage, et s'il produit de meilleurs résultats en sortie, alors on continue l'apprentissage. Mais il faut aussi faire attention au problème de « sur-apprentissage » (overfitting). Le sur-apprentissage se caractérise la plupart du temps par des performances très élevées sur des données d'apprentissage, mais des performances très basses sur des données de test.
4. **L'algorithme d'apprentissage** constitue la méthode avec laquelle le modèle statistique va se paramétrer à partir des données d'apprentissage. Il est choisi en fonction de la tâche que l'on souhaite accomplir, ou du type de données dont on dispose. Les algorithmes d'apprentissage peuvent être regroupés en plusieurs catégories, dont les deux principales sont : l'apprentissage non-supervisé et l'apprentissage supervisé. Dans l'apprentissage non-supervisé l'algorithme doit découvrir par lui-même la structure plus ou moins cachée des données, et regrouper ainsi les données entre elles, sans aucune information sur les étiquettes, les classes ou les données de sorties. Par contre, dans l'apprentissage supervisé, les données de sortie sont disponibles lors de l'apprentissage en plus des données d'entrées. Le but

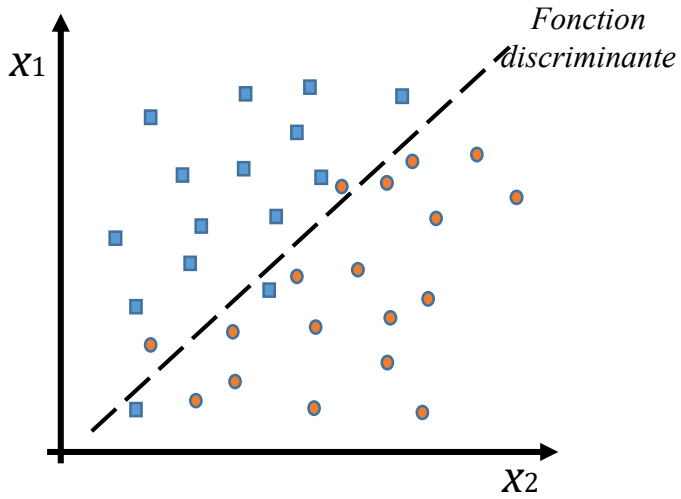


FIGURE 2.11 – Exemple de classification avec l'analyse discriminante

consiste alors à trouver la fonction qui approxime le mieux les données de sorties qui sont connues lors de l'apprentissage en fonction des données d'entrées.

Une méthode d'apprentissage automatique se divise en 3 grandes phases : l'entraînement du modèle (apprentissage), le test du modèle construit, ainsi que la validation de modèle et de la méthode. La phase d'entraînement du modèle utilise les algorithmes d'apprentissage pour trouver le modèle qui permettra de prédire les résultats de sortie en fonction d'un ensemble de valeurs des variables d'entrées. La phase de test permet d'évaluer l'exactitude des modèles construits, en évaluant l'erreur et la corrélation entre les données réelles et les données prédites par les modèles générés. La phase de validation de la méthode quant à elle évalue la robustesse et la précision des différents modèles générés.

Il existe plusieurs algorithmes d'apprentissage, parmi lesquels les méthodes de classification telles que : l'analyse discriminante (DA), les k-plus proches voisins (KNN) ; et les méthodes de régression telles que : les réseaux de neurones artificiels (ANN), les machines à vecteurs supports (SVM), la régression non linéaire (NLR), les arbres de décision (DT), et la logique floue (FL). Ces différentes méthodes d'apprentissage seront utilisées dans les chapitres suivants pour construire des modèles d'évaluation de la qualité des images, à partir d'un ensemble de caractéristiques extraites des images et des scores subjectifs pris dans les bases de données d'images étudiées dans la Section 2.2.

2.4.1 Analyse discriminante

L'analyse discriminante (AD) est une technique de classification permettant la modélisation d'une variable qualitative (discrète) Y à K modalités (classes), à partir de P variables explicatives $X(X_1, ...,X_p)$ [46]. L'analyse discriminante comporte deux principales approches : l'analyse discriminante descriptive et l'analyse discriminante décisionnelle. L'aspect descriptif de l'analyse discriminante consiste en un apprentissage supervisé, qui

est utilisé pour déterminer les combinaisons linéaires de variables qui permettent de séparer le mieux possible les données d'apprentissage en K classes. L'aspect décisionnel quant à lui, est utilisé lorsque l'on a un nouvel individu pour lequel on connaît les valeurs explicatives et qu'il faut décider dans quelle classe l'affecter [47]. La Figure 2.11 présente un exemple de classification grâce à la méthode d'analyse discriminante, avec $P = 2$ variables explicatives, et $K = 2$ classes, et une seule fonction discriminante.

Les données utilisées pour l'analyse discriminante consistent en un échantillon de N observations de la variable qualitative Y à K modalités, et des P variables explicatives $X(X_1, ...,X_p)$. On note y_i la valeur de sortie de l'individu i ; par x_{ij} la valeur de l'individu i pour la variable explicative j ; et par $x_i(x_{i1}, ..., x_{ip})$ le vecteur des variables explicatives pour l'individu i.

L'objectif de l'analyse discriminante est de trouver de nouvelles variables représenté par des équations discriminantes, qui sont des combinaisons linéaires des variables explicatives, qui discriminent au mieux les groupes définis par les modalités de la variable à expliquer. Ces variables notées D_t se définissent comme dans (2.13).

$$D_t = u_t X = \sum_{i=1}^{P} u_{ti} X_i \tag{2.13}$$

où $u_t(u_{t1}, ..., u_{tp})$ est le vecteur des coefficients de la combinaison linéaire pour la variable t.

On définit les éléments suivants : V est la matrice de variance-covariance globale définie par (2.14) ; W est la matrice de variance-covariance intra-classe définie par (2.15) ; et B est la matrice de variance-covariance inter-classes définie par (2.16).

$$V = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})' \tag{2.14}$$

$$W = \sum_{k=1}^{K} n_k V_k \tag{2.15}$$

$$B = \sum_{k=1}^{K} n_k (\bar{x_k} - \bar{x})(\bar{x_k} - \bar{x})' \tag{2.16}$$

avec $\bar{x} = \frac{1}{N} \sum_i x_i$ qui représente le vecteur centre de gravité ; $n_k = \frac{N_k}{N}$ qui est le rapport du nombre d'individus dans un groupe sur le nombre total d'individus, et aussi $V = W + B$.

Les différents groupes sont bien séparés si la dispersion à l'intérieur des groupes, encore appelée dispersion intra-groupe, est très faible, et que la dispersion entre les groupes, encore appelé dispersion intergroupes, est très grande. Le problème d'optimisation des coefficients des équations discriminantes se résume donc en la minimisation des variances intra-groupe, dans la mesure où tous points dans le même groupe sont proches du centre de gravité ; et en la maximisation de la variance intergroupes, quand les centres de gravité des différents groupes se trouves éloignés. Cela revient au final à maximiser le

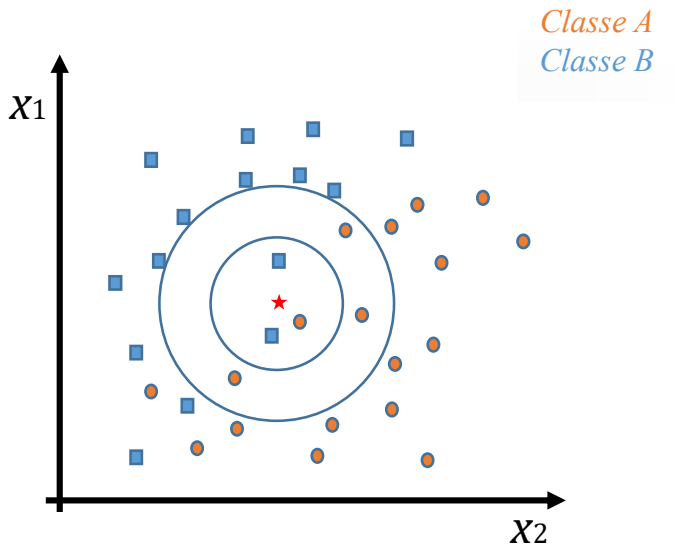


FIGURE 2.12 – Exemple de classification avec $k - NN$

déterminant des matrices $W^{-1}B$ ou $V^{-1}B$; et donc de trouver le vecteur propre u de $W^{-1}B$ associé à sa plus grande valeur propre.

Cette méthode a été utilisée dans [48, 49] comme méthode de classification de la qualité des images. L'apprentissage se fait sur un ensemble de métriques avec références et à distorsions spécifiques dans [48] ; et sur des métriques sans référence et basés sur les distorsions liées au flou, à l'effet de bloc, au bruit et à l'effet de Gibbs, dans [49].

2.4.2 K-Plus proches voisins

La méthode des k-plus proches voisins ($k - NN$) est une méthode d'apprentissage supervisée qui raisonne avec le principe de classification d'un objet en fonction de ses voisins les plus proches [50, 51]. L'algorithme des $k - NN$ est un algorithme intuitif, aisément paramétrable, et permettant de traiter les problèmes de classification ayant un nombre quelconque de classes. Nous considérons que la variable aléatoire dépendante Y possède L modalités (ou classes), et doit être décrite par P variables explicatives indépendantes X_1, ..., X_p, groupées dans une variable aléatoire notée X.

Le principe de l'algorithme des k-plus proches voisins est particulièrement simple : pour chaque nouveau point x à classifier, on commence par déterminer l'ensemble de ses $k-$plus proches voisins, parmi les points d'apprentissage que l'on peut noter ici comme $V_k(x)$ (avec k compris entre 1 et N, où N représente le nombre d'individus dans l'ensemble d'apprentissage). La classe du point x est alors la classe majoritairement représentée dans l'ensemble de ses voisins $V_k(x)$. La Figure 2.12 illustre une classification grâce à la méthode de $k - NN$, avec $P = 2$ variables explicatives, et $L = 2$ classes. On voit, dans cette figure, l'influence du choix du nombre k de voisins sur le résultat de la classification : si $k = 3$, le point à classifier est de la *classe B*, mais si $k = 6$, le point à

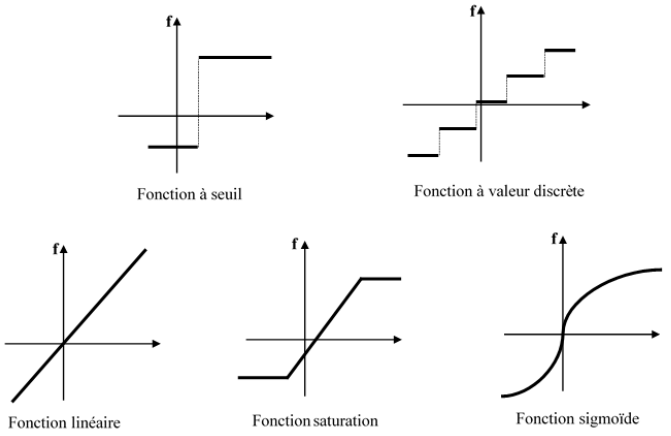


FIGURE 2.13 – Fonctions d'activation

classifier est de la *classe A*.

Pour une définition plus formelle et précise de la méthode, il faut commencer par choisir un type de distance, qui permettra d'évaluer la distance entre 2 points de espace circonscrit. Pour un nouveau point x, on définit l'ensemble de ses $k-$ plus proches voisins $V_k(x)$ au sens de cette distance. Pour définir cet ensemble, on calcule la distance entre le point à classifier et tous les points de l'espace d'apprentissage, puis on prend les k points ayant les distances les plus petites entre eux et le point à classifier x. Dans l'ensemble $V_k(x)$ défini, on compte le nombre de points pour chacune des L modalités de la variable dépendante Y, et la classe du point à classifier x est celle qui a le nombre maximum de points dans l'ensemble des voisins.

Dans [52] les auteurs utilisent la méthode des k-plus proches voisins pour évaluer la qualité des images prises par un appareil photo, en considérant en entrée de l'algorithme $k - NN$ un ensemble de caractéristiques extraites de l'image, et liées aux couleurs, aux formes, aux contours, et aux bruits. De même, dans [53] les auteurs utilisent cette méthode de classification pour construire des modèles d'évaluation des images dans le domaine de la médecine et tout particulièrement de la détection des cancers gastriques.

2.4.3 Réseaux de neurones artificiels

Un réseau de neurones artificiels (RNA ou ANN) est un réseau fortement connecté de processeurs élémentaires ou neurones artificiels fonctionnant en parallèle [54]. Dans le RNA, chaque neurone artificiel ou neurone formel calcule sa sortie unique sur la base d'une ou plusieurs informations reçues en entrées ; et la sortie ainsi évaluée est distribuée sur le réseau. Le fonctionnement du neurone artificiel est calqué sur celui du neurone biologique, qui est le suivant : tout d'abord il doit recevoir un flux nerveux d'autres neurones, et cela grâce à la « dendrites » ; puis il va transmettre les flux reçus à travers le neurone grâce à « l'axone » ; l'étape suivante consiste à évaluer les informations reçues grâce au « corps cellulaire », et enfin il va connecter sa sortie à d'autres neurones grâce

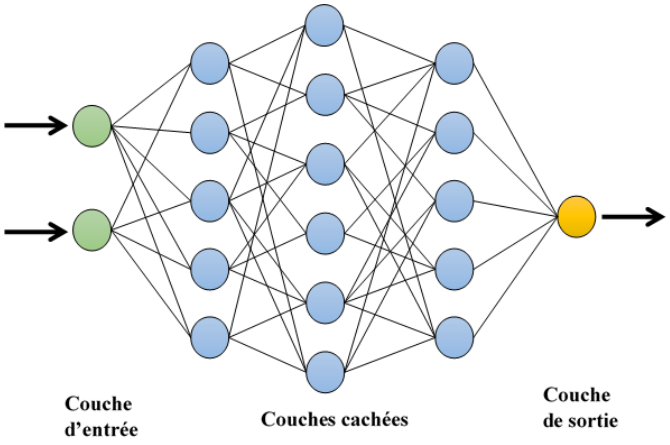


FIGURE 2.14 – Exemple d'un réseau de neurones à 5 couches

aux « synapses ».

Un neurone formel est caractérisé par son état, le niveau d'activation qu'il reçoit en entrée, et le poids de ses connections. Il peut généralement se décomposer en deux parties : une partie d'évaluation de la stimulation reçue, et une seconde partie d'évaluation de son activation. L'évaluation des stimulations reçues est faite par des fonctions d'entrées qui sont généralement des sommes pondérées des signaux d'entrées, affectés d'une unité fictive dont le poids permet de régler le seuil de déclenchement du neurone, appelé « biais d'entrée ». La partie d'évaluation de l'activation quant à elle, se fait à partir des fonctions d'activation. Il existe plusieurs types de fonctions d'activation, dont certaines sont présentées dans la Figure 2.13.

Un réseau de neurones est en général composé d'une succession de couches dont chacune prend ses entrées sur les sorties de la couche précédente. Chaque couche i est composée de N_i neurones formels, prenant leurs entrées sur les N_{i-1} neurones de la couche précédente. À chaque synapse est associé un poids synaptique, de sorte que les entrées N_{i-1} soient multipliées par ces poids, puis additionnées entre elles par les neurones de niveau i, pour obtenir la sortie du réseau ; ce qui équivaut à multiplier le vecteur d'entrée par une matrice de transformation. La Figure 2.14 est une illustration d'un exemple de structure d'un réseau de neurones à cinq (05) couches, parmi lesquelles une couche d'entrée à deux (02) neurones, trois (03) couches cachées à 5, 6 et 5 neurones respectivement, et une couche de sortie à un neurone.

L'apprentissage du réseau de neurones se fait par modification du comportement du réseau jusqu'à l'obtention du comportement désiré. Il est possible de faire un apprentissage supervisé ou un apprentissage non-supervisé. Au cours de l'apprentissage, le système modifie les poids des différentes connections entre les neurones [55], pour donner, au final, un modèle d'évaluation constitué de couches interconnectées entre elles.

La méthode d'apprentissage basée sur les réseaux de neurones fait partie des méthodes les plus utilisées dans l'évaluation de la qualité des images. On peut à cet effet citer les

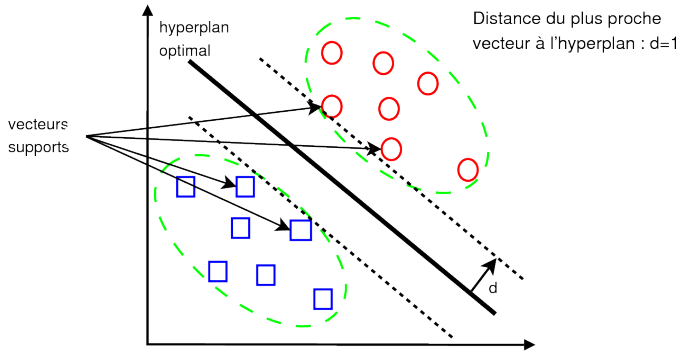


FIGURE 2.15 – Exemple de classification binaire avec SVM

travaux présentés dans [7] sur l'évaluation de la qualité des vidéos à partie de plusieurs métriques avec références, ou encore ceux présentés dans [56] sur l'évaluation de la qualité de 352 images déformées avec les compressions $JPEG$ et $JPEG2000$.

2.4.4 Machine à vecteurs supports

Une machine à vecteurs supports ou séparateur à vastes marges (SVM) est une méthode d'apprentissage utilisée pour résoudre les problèmes de discrimination linéaire ou non linéaire, en classification (trouver la classe d'un échantillon) ou en régression (trouver la valeur numérique précise d'une variable) [57, 58]. Il s'agit d'une méthode d'apprentissage supervisée, qui a fait ses preuves et est aujourd'hui très utilisée dans de nombreux domaines, tels que la bio-informatique, la recherche d'information, le traitement d'image, etc. Cette méthode est très utile et produit de bonnes performances surtout dans des domaines utilisant des données à grandes dimensions, tels que l'évaluation de la qualité des images à partir des métriques sans référence [15, 59]. La méthode d'apprentissage automatique SVM est basée sur deux idées générales : la notion de marges maximales et la notion de fonction de noyau.

Dans la méthode SVM, la marge est la distance entre la frontière de séparation et les échantillons les plus proches. Les échantillons les plus proches de cette frontière de séparation sont appelés des vecteurs supports. La frontière de séparation optimale est la fonction qui sépare les données tout en maximisant les marges : cette frontière est aussi appelée « hyperplan optimal ». Le problème à résoudre devient donc celui de la recherche de ces frontières de séparation optimales à partir d'un ensemble d'apprentissage, et donc de la résolution d'un problème d'optimisation quadratique. La Figure 2.15 présente un exemple de séparation par un hyperplan optimal des données dans une classification binaire linéairement séparable. Elle présente aussi les marges et les vecteurs supports utilisés pour cette classification.

Lorsque les données ne sont pas linéairement séparables, la seconde idée générale de la méthode SVM qui est la notion de « fonction noyau » intervient. Il s'agit de transformer l'espace de représentation des données d'entrées en un espace ayant de plus grandes dimensions dans lequel le problème est linéairement séparable. Cette transformation est

faite grâce à des fonctions noyaux qui permettent d'effectuer le changement d'espace. Une fonction noyau permet de transformer un produit scalaire dans un espace de grande dimension en une simple évaluation ponctuelle d'une fonction dans un espace à plus grandes dimensions.

L'apprentissage supervisé dans la méthode SVM a pour but de trouver la fonction $h(x)$, qui fait correspondre une sortie y au vecteur d'entrée x tel que $y = h(x)$. Le but de cet apprentissage est donc de trouver la fonction optimale $h(x)$ qui permet de séparer au mieux les données d'apprentissage. En considérant un problème de discrimination binaire linéairement séparable, il importe de trouver $h(x)$ tel que x soit dans la première classe si $h(x) > 0$ soit dans la seconde classe sinon. $h(x)$ est alors une combinaison linéaire du vecteur d'entrée x et d'un vecteur de poids $w = (w_1, ..., w_n)$ comme présenté dans (2.17).

$$h(x) = w'x + w_0 \tag{2.17}$$

Dans le cas d'un problème multidimensionnel et non linéairement séparable, il s'avère nécessaire d'utiliser les fonctions noyaux pour redimensionner le problème et se mettre dans un environnement où ledit problème soit linéairement séparable [60].

Les machines à vecteurs de supports sont très utiles lorsqu'on a un grand nombre de variables d'entrées, ce qui est généralement le cas dans le domaine de l'évaluation de la qualité des images, tout particulièrement lorsqu'il s'agit d'une évaluation basée sur des caractéristiques extraites d'une image sans référence. On retrouve cette méthode d'apprentissage basée sur les SVM dans des travaux tel que [13, 16, 61], portant sur l'évaluation de la qualité des images sans référence.

2.4.5 Régression non linéaire

La régression non linéaire (NLR) consiste à ajuster un modèle non linéaire pour un ensemble de valeurs afin de déterminer la courbe qui se rapproche le plus de celle des données d'une variable aléatoire dépendante Y en fonction de variables de prédiction indépendantes $X(X_1, ..., X_p)$ [62, 63]. Le but de cette méthode est donc de trouver une équation qui permet de décrire les données de sortie en fonction des données d'entrées grâce à un ensemble de données d'apprentissage, et partant de prédire la sortie pour une nouvelle observation dont les paramètres sont pris en entrée [64].

L'ajustement du modèle non linéaire consiste à déterminer les paramètres de l'équation qui permettront de minimiser l'erreur $S = ||r_i||$; avec $r_i = y_i - f(x_i)$ étant l'écart entre la sortie attendue (valeur mesurée) et la sortie produite (valeur prédite) par le modèle. Il existe plusieurs types de normes utilisées pour effectuer cette évaluation de l'erreur, mais la plus courante est la norme euclidienne. Cette minimisation de l'erreur requiert pour sa mise en œuvre la méthode des moindres carrés.

La méthode des moindres carrés est une méthode permettant de comparer et de minimiser l'erreur entre les données expérimentales et les données produites par un modèle [65]. Cette méthode permet de déterminer, parmi un ensemble de fonctions, celle qui reproduit le mieux les données expérimentales. Le but, reste donc d'estimer au mieux la variable dépendante Y en fonction des variables indépendantes regroupées en X, en

minimisant l'erreur d'estimation. Cette technique s'appelle aussi l'ajustement par la méthode des moindres carrés. La méthode des moindres carrés permet de construire un estimateur $f(x, \theta)$ qui décrit au mieux les données. Son objectif est de trouver les paramètres optimaux θ qui minimisent la somme $S(\theta)$ des erreurs quadratiques entre les données prédites et les données de mesures. Cette somme des erreurs quadratiques est définie par (2.18).

$$S(\theta) = \sum_{i=1}^{N} r_i^2(\theta) = \sum_{i=1}^{N} (y_i - f(x_i, \theta))^2 \tag{2.18}$$

L'apprentissage dans la méthode de régression non linéaire consiste à trouver les paramètres optimaux θ, donc de trouver θ tel que la somme $S(\theta)$ soit minimale. Pour avoir la valeur minimale de $S(\theta)$, il faut trouver la valeur de θ qui annule la dérivée de $S(\theta)$ en fonction de θ [66]. L'ajustement des paramètres peut se faire grâce à un algorithme itératif consistant à exécuter les étapes suivantes :

1. définir une valeur initiale $\hat{\theta}_0$ du paramètre ;
2. calculer l'estimateur $i + 1$en fonction de l'estimation i ;
3. on s'arrête lorsque l'écart entre l'erreur obtenue à l'étape i et l'étape $i + 1$ est négligeable

Il peut exister des minimums locaux dans la fonction, qui ne donnent pas la valeur optimale du paramètre, mais qui annulent tout de même la dérivée de la fonction, par conséquent, il importe de prendre plusieurs valeurs initiales du paramètre pour augmenter les chances de tomber sur le minimum global de la fonction.

Cette méthode basée sur la régression non linéaire a été utilisée pour l'évaluation de la qualité des images et des vidéos dans [8, 67]. Les modèles d'évaluation sont construits à partir d'un ensemble de caractéristiques extraites soit d'une image (dans les cas d'une évaluation sans référence), soit de la comparaison de l'image originale à l'image déformée.

2.4.6 Arbre de décision

La méthode d'apprentissage automatique basée sur les arbres de décision est une méthode d'apprentissage visant à créer un modèle sous forme d'arbre qui prédit la valeur d'une cible (ou variable dépendante), sur la base d'un ensemble de valeurs d'entrées (ou variables indépendantes) [68]. Un exemple d'un modèle d'arbre de décision est présenté dans la Figure 2.16. Il est constitué de quatre variables indépendantes prises en entrée et d'une variable dépendante continue en sortie. Les méthodes d'apprentissage automatique basées sur les arbres de décision se divisent en deux grandes catégories [69] :

— les arbres de régression, lorsqu'il s'agit d'un problème de prédiction avec la variable à prédire continue ;
— les arbres de classification, dont l'objectif est de construire des groupes homogènes dans l'espace de descripteurs.

Cette méthode est plutôt simple à implémenter. Elle consiste à trouver un partitionnement des individus que l'on représente sous la forme d'un arbre de décision. L'objectif

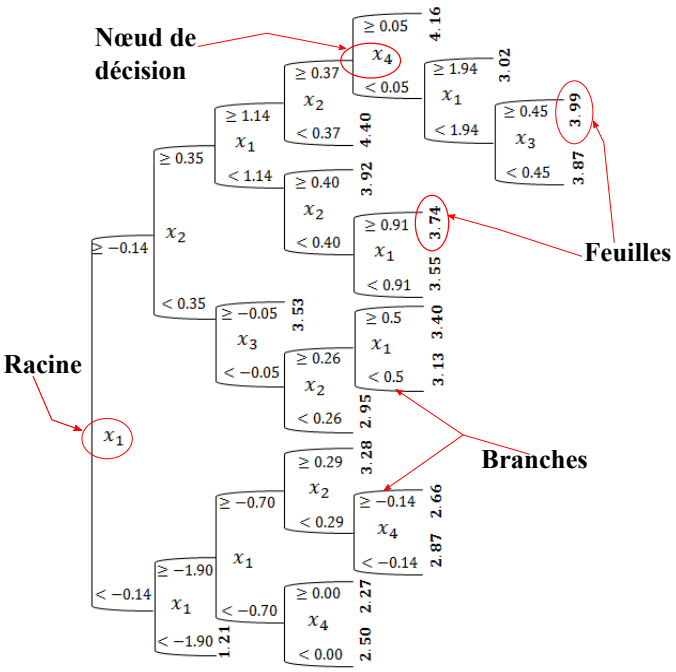


FIGURE 2.16 – Exemple d'un arbre de décision binaire

est de produire des groupes d'individus les plus homogènes possibles du point de vue de la variable à prédire. Un arbre de décision se construit selon un processus récursif et itératif qui est le partitionnement [70]. Ce processus de partitionnement divise les données en partitions ou branches de l'arbre jusqu'à l'obtention de feuilles pures. Dans ces structures d'arbres, les feuilles représentent les étiquettes des classes et les branches représentent les conjonctions de caractéristiques qui conduisent à ces étiquettes de classe. Le premier sommet (situé au premier niveau dans l'arbre) est la « racine » de l'arbre.

Le modèle de prédiction peut être très facilement lu. On peut traduire un arbre en une base de règles sans altération d'informations. Le chemin menant d'un sommet vers la racine de l'arbre peut être traduit en une partie prémisse d'une règle de prédiction de type attribut-valeur (« SI attribut 1 = valeur 1 ET attribut 2 = valeur 2 » ... alors « sortie = valeur x ») [71]. Pour classer un nouvel individu, il suffit de l'injecter dans l'arbre, et de lui associer la conclusion attachée à la feuille dans laquelle il aboutit.

L'apprentissage automatique utilisé par la méthode des arbres de décision est un apprentissage supervisé. Le gros du problème est donc de construire l'arbre à partir des données d'apprentissage. Pour ce faire, il importe de répondre aux questions suivantes : comment faire le choix de la variable de segmentation sur un nœud ? Comment fixer les seuils ? Comment définir la taille de l'arbre ? Comment prendre une décision en cas d'ambigüité ? Pour ne se limiter aux seules question dont les réponses préfigurent dans cette étude. Il existe plusieurs méthodes d'apprentissage pour les arbres de décision, parmi lesquelles les méthodes : « $ID3$ » (Inductive Decision Tree) et son successeur « $C4.5$ »,

« $CART$ » (Classification and Regression Tree), « $CHAID$ » (Chi-Square Automatic Interaction Detection), « $QUEST$ » (Quick, Unbiased, Efficient Statistical Trees) [72, 73].

En ce qui concerne la segmentation des variables d'entrées, pour chaque variable candidate, réaliser le partitionnement des observations et calculer un indicateur de qualité ; la variable retenue sera alors celle qui optimise cet indicateur. Les méthodes diffèrent selon la mesure utilisée : par exemple pour évaluer la pertinence de la variable dans la segmentation, $CHAID$ propose d'utiliser le « Khi-2 » d'écart à l'indépendance, dont la formule est donnée en (2.19). Ce critère varie en 0 et $+\infty$, ce qui le rend parfois difficile à manipuler et à interpréter. Pour résoudre ce problème, il est parfois préférable d'utiliser le critère normalisé en 0 et 1en fonction du degré de liberté qu'est le t de $Tschuprow$, dont la formule est donnée en (2.20). Pour les méthodes ID3 et C4.5 le critère utilisé est plutôt l'entropie, dont la formule est donnée en (2.21).

$$\chi^2 = \sum_{k=1}^{K} \sum_{l=1}^{L} \frac{\left(n_{kl} - \frac{n_{k.} * n_{.l}}{n}\right)^2}{\frac{n_{k.} * n_{.l}}{n}} \tag{2.19}$$

$$t = \frac{\chi^2}{n\sqrt{(K-1) * (L-1)}} \tag{2.20}$$

$$H(E) = -\sum_{k} P(w_k) log_2 \left(P(w_k)\right) \tag{2.21}$$

où n est le nombre de données dans l'échantillon d'apprentissage, K est le nombre de classes, L est le nombre de descripteurs (variables indépendantes), $P(w_k)$ est la probabilité de la classe w_k.

Pour déterminer le seuil des variables continues, deux tâches sont effectuées : sélectionner la meilleure valeur de coupure pour chaque variable continue ; et sélectionner globalement la meilleure segmentation en comparant la pertinence de tous les descripteurs. Il s'agit, dans un premier temps, de trier les données selon les valeurs de la variable indépendante continue, puis de tester chaque borne de coupure possible entre deux valeurs de la variable en calculant le critère de segmentation choisit ; pour retenir enfin celui qui optimise l'indicateur de qualité du partitionnement qu'est ce critère. Le processus est itératif et récursif, et s'arrête lorsque le taux d'erreur calculé sur les données d'apprentissage commence à stagner.

2.4.7 Logique floue

La logique floue (FL) est une généralisation de la logique classique qui permet la modélisation des imperfections des données et se rapproche, dans une certaine mesure, de la flexibilité du raisonnement humain [74, 75]. Elle introduit une notion de degré ou probabilité de vérification d'une condition, appelé aussi degré d'appartenance à un ensemble, dans la logique classique. Cela permet ainsi d'avoir des conditions dans des états autres que 0 ou 1 [76]. En logique classique, la notion d'appartenance est très simple : étant donné un élément, soit il appartient à un ensemble, la fonction d'appartenance prend la valeur 1 ; soit il n'appartient pas à cet ensemble, alors la fonction d'appartenance

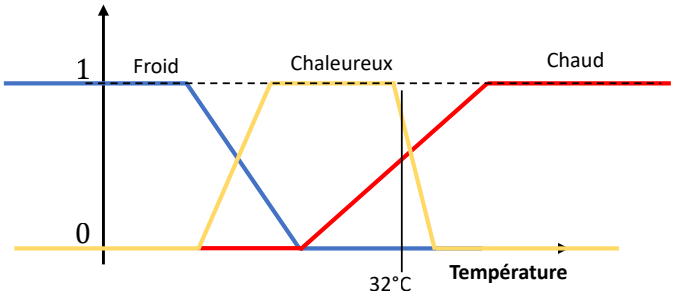


FIGURE 2.17 – Exemple de logique floue pour la gestion de la température

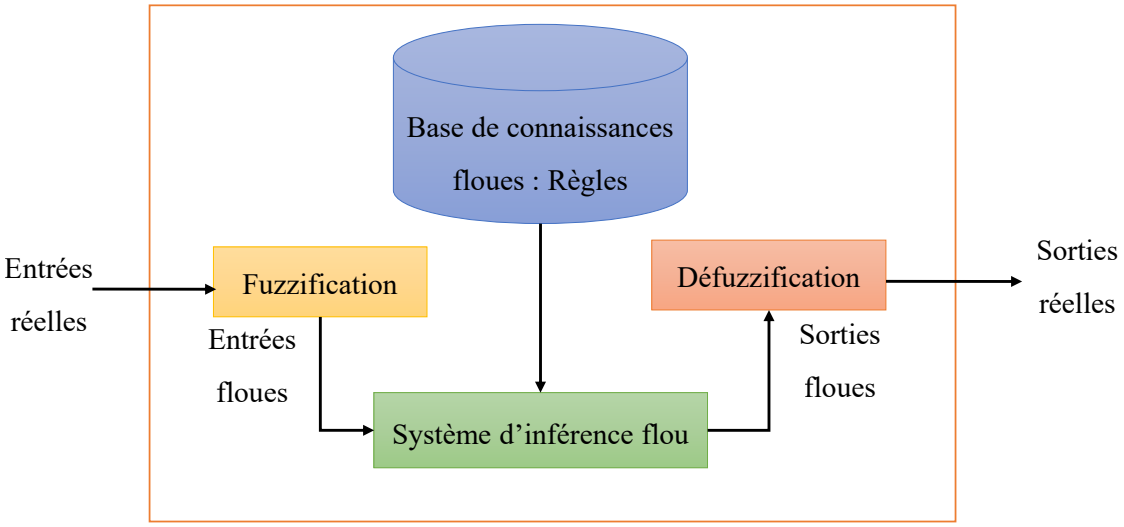


FIGURE 2.18 – Architecture générale du système de logique floue

prend la valeur 0. Le concept d'appartenance est différent en logique floue, et constitue la base de cette théorie : « un élément peut appartenir à un ensemble avec une un degré d'appartenance ». Par exemple, contrairement à la logique classique où l'on dira : « il fait chaud ! » (état de sortie à 1), ou « il fait froid ! » (état de sortie à 0) [77] ; en logique floue, on peut avoir des états intermédiaires dans lesquels « il fait chaud » avec un degré de liberté de 0.7, et « froid » avec un degré de liberté de 0.6. La Figure 2.17 montre un exemple de mise en œuvre de la logique floue au travers de la gestion de la température.

L'architecture générale d'un système de logique floue est illustrée dans la Figure 2.18 ; elle peut se détailler comme suit :

— **Les entrées** : les entrées d'un système de logique floue sont les valeurs réelles nettes des variables aléatoires indépendantes notées X. Il s'agit, par exemple, de la valeur 10, pour spécifier une température de $10°C$, en entrée du système.

— **Le module de « Fuzzification »** : il s'agit de l'étape dans laquelle on transforme les entrées nettes ou valeurs réelles en ensembles flous ou valeurs floues. Cela passe par la définition des fonctions d'appartenance (Membership Function : MF

TABLE 2.8 – Équivalence entre les opérations du système classique et du système flou

Dénomination	**Intersection « ET » :** $\mu_{A\cap B}(x)$	**Réunion « OU » :** $\mu_{A\cup B}(x)$	**Complément « NON » :** $\mu_{\bar{A}}(x)$
Opérateur de Zadeh MIN / MAX	$min(\mu_A(x), \mu_B(x))$	$max(\mu_A(x), \mu_B(x))$	$1 - \mu_A(x)$
Opérateur probabiliste PROD / PROBOR	$\mu_A(x) \times \mu_B(x)$	$\mu_A(x) + \mu_B(x) - \mu_A(x) \times \mu_B(x)$	$1 - \mu_A(x)$

en anglais) pour chacune des variables d'entrées du système flou. Une fonction d'appartenance d'un sous-ensemble A de X, est une fonction $\mu_A : X \to [0, 1]$. Il s'agit là d'une fonction qui, à chaque valeur de l'ensemble X, associe un degré d'appartenance qui est une valeur réelle comprise entre 0 et 1.

— **La base de connaissances ou les règles** : elle englobe un ensemble de règles sous la forme « SI - ALORS », produit par des experts du domaine étudié. Ces règles sont sous la forme de condition, tel que présenté en (2.22).

$$if\ x_1 \in A_1\ et, ...,\ et\ x_n \in A_n\ then\ f \in S \tag{2.22}$$

où les x_i sont les valeurs de l'ensemble d'entrée X, les A_i sont les sous-ensembles flous caractérisés par des fonctions d'appartenance trouvées lors de la phase de « fuzzification » ; f est la sortie de la règle, et S le sous-ensemble flou de sortie caractérisé par une fonction d'appartenance.

— **Le système d'inférence** : c'est ici qu'est simulé le processus de raisonnement humain. Ceci est fait en effectuant une déduction des données de sorties floues en utilisant les données d'entrées floues ainsi que les règles. Un système d'inférence flou (FIS : Fuzzy Inference System ; en anglais) est un système complet de règles basées sur la logique floue, qui prend comme entrées, la sortie des fonctions d'appartenance des variables d'entrées d'un système de logique floue, et produit une sortie floue, interprétable grâce à la fonction d'appartenance de sortie du système. Les équivalences entre les opérations dans un système de logique classique et système d'inférence flou sont données dans le Tableau 2.8.

— **Le module de « Defuzzification »** : c'est le module de transformation de l'ensemble de valeurs floues en sortie du système d'inférence, en valeurs nettes (valeurs réelles). Cette transformation utilise les fonctions d'appartenance de sortie.

— **La sortie** : la sortie du système de logique flou est la valeur réelle de la variable

aléatoire dépendante Y correspondant aux valeurs réelles prises comme entrées du système.

Les fonctions d'appartenance généralement utilisées sont : la gaussienne ($GaussMF$), la cloche Bell ($GbellMF$), la logarithmique (SMF), la triangulaire ($TriMF$) et la trapézoïdale ($TrapMF$) ; les formes des courbes de ces fonctions sont données dans la Figure 2.19.

1. La fonction d'appartenance Gaussienne ($GaussMF$) est définie par deux paramètres $\{m, \sigma\}$, sa moyenne et son écart type, respectivement. L'équation d'implémentation de cette fonction est donnée dans (2.23).

$$\mu_A(x; \sigma, m) = exp\left(\frac{-(x-m)^2}{2\sigma^2}\right) \tag{2.23}$$

2. La fonction d'appartenance en cloche ($GbellMF$) est définie par trois paramètres $\{a, b, c\}$, où les paramètres a et c permettent de définir la largeur et le centre de la fonction, respectivement, et le paramètre b permet de contrôler sa pente de croissance et de décroissance. L'équation d'implémentation de la fonction en cloche est donnée en (2.24).

$$\mu_A(x; a, b, c) = \frac{1}{1 + |\frac{x-c}{a}|^{2b}} \tag{2.24}$$

3. La fonction d'appartenance triangulaire ($TriMF$) est définie par trois paramètres $\{a, b, c\}$, comme présenté dans l'équation (2.25) ; où les paramètres a et c représentent les pieds du triangles, qui sont les sommets dont les ordonnées sont égales à 0, et b représente le pic, qui est le sommet dont l'ordonnée est égale à 1.

$$\mu_A(x; a, b, c) = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ \frac{c-x}{c-b} & b \leq x \leq c \\ 0 & c \leq x \end{cases} \tag{2.25}$$

4. La fonction d'appartenance trapézoïdale ($TrapMF$) est définie par quatre paramètres $\{a, b, c, d\}$, comme présenté dans l'équation (2.26) ; où les paramètres a et d représentent les pieds du trapézoïde (sommets dont les ordonnées sont égales à 0), tandis que b et c représentent les pics (sommets dont les ordonnées sont égales à 1).

$$\mu_A(x; a, b, c, d) = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \leq c \\ \frac{d-x}{d-c} & c \leq x \leq d \\ 0 & d \leq x \end{cases} \tag{2.26}$$

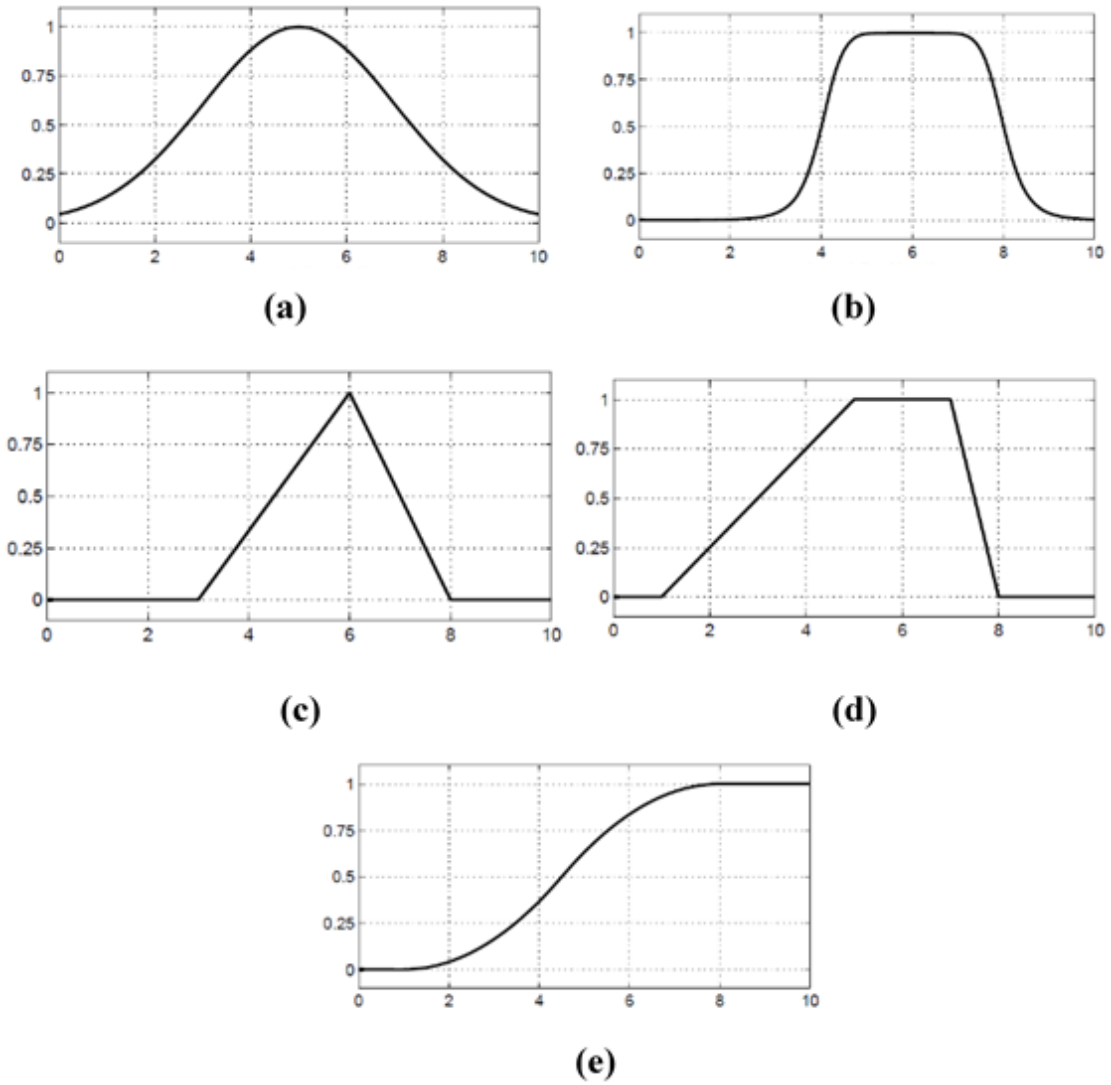


FIGURE 2.19 – Exemple de fonction d'appartenance (MF). (a) GaussMF avec $\{\sigma = 2, m = 5\}$; (b) GbellMF avec $\{a = 2, b = 4, c = 6\}$; (c) TriMF avec $\{a = 3, b = 6, c = 8\}$; (d) TrapMF avec $\{a = 1, b = 5, c = 7, d = 8\}$; (e) SMF avec $\{a = 1, b = 8\}$.

5. La fonction d'appartenance logarithmique (SMF) est définie par deux paramètres $\{a, b\}$, comme présenté dans la (2.27), où le paramètre a représente l'extrémité inférieure de la courbe, et le paramètre b définit la pente de la fonction logarithmique.

$$\mu_A(x; a, b) = \begin{cases} 0 & x \leq a \\ 2\left(\frac{x-a}{b-a}\right)^2 & a \leq x \leq \frac{a+b}{2} \\ 1 - 2\left(\frac{x-b}{b-a}\right)^2 & \frac{a+b}{2} \leq x \leq b \\ 1 & b \leq x \end{cases} \tag{2.27}$$

Dans l'utilisation de la logique floue comme méthode d'apprentissage automatique, le but est de construire à partir d'un ensemble de données d'apprentissage, des fonctions d'appartenance (MF) d'entrées et de sorties ; de déterminer automatiquement les règles et de trouver les coefficients et les paramètres à utiliser dans le système d'inférence flou.

2.5 Conclusion

Le but de ce chapitre était de définir quelques notions de base qui seront utilisées dans la suite du travail. C'est dans cette optique qu'a été faite une description des méthodes d'évaluation subjectives et de quelques bases de données d'images qui seront utilisées dans la détermination de la qualité visuelle des images. Puis a aussi été faite une description des indices de mesure des performances, qui permettront non seulement d'ajuster les paramètres des méthodes utilisés, mais aussi de mesurer leurs performances et de les comparer aux méthodes présentes dans la littérature. A enfin été réalisée, une description des méthodes d'apprentissage automatiques qui permettrons de définir les paramètres d'utilisation optimale de ces méthodes.

Dans cette étude, ces méthodes d'apprentissage automatiques sont indispensable pour la construction des modèles objectifs d'évaluation de la qualité des images à partir de certaines métriques tirées des images et de la comparaison entre une image dégradée et sa version originale. Les indices de performances étudiés dans ce chapitre permettront d'évaluer les performances des différentes méthodes objectives d'évaluation de la qualité des images par rapport aux scores données par des observateurs humains à ces images lors des évaluations subjectives.

Chapitre 3

Évaluation de la qualité des images basée sur des métriques avec référence

3.1 Introduction

L'évaluation subjective est le moyen le plus naturel d'avoir les informations les plus pertinantes sur la qualité visuelle d'une image. Cependant, les méthodes permettant d'effectuer cette évaluation subjective sont très difficiles à mettre en place. Elles demandent beaucoup de temps et de ressources matérielles et humaines pour leur application efficiente. Ces méthodes subjectives ne peuvent donc pas être utilisées pour des applications à temps-réels. De ce fait, l'évaluation objective a été proposée pour pallier à ces problèmes. L'objectif de l'évaluation objective est de remplacer l'évaluation subjective par des méthodes d'évaluation automatique de la qualité d'image. Ces méthodes objectives doivent elles aussi être en corrélation avec la perception visuelle humaine. L'évaluation subjective étant le moyen le plus naturel de déterminer la qualité d'une image corrélée avec la perception visuelle humaine, demeure la référence pour les méthodes d'évaluation objectives.

De plus en plus, les recherches explorent la possibilité d'utiliser des méthodes d'apprentissage automatique sur un ensemble de caractéristiques extraites des images elles-même, en vue de construire des modèles pour en évaluer la qualité. Dans [78, 79] les auteurs proposent des mesures de la qualité des images, basées sur une extraction des caractéristiques locales des blocs de l'image originale et déformée telles que : la moyenne, écart-type, entropie, énergie, ... Un réseau de neurones artificiels est ensuite utilisé pour estimer la qualité de l'image à partir de ces descripteurs. Dans [56] d'autres caractéristiques de comparaison de l'image originale à l'image déformée sont intégrées, telles que : la covariance, l'erreur quadratique moyenne, etc. Dans [80] les caractéristiques sont extraites sur l'image d'erreur, définie ici comme la différence entre l'image originale et l'image déformée. Dans [7, 8, 81] les auteurs proposent d'utiliser des métriques ($SSIM$, $FSIM$, $PSNR$, ...) comme caractéristiques d'entrées des méthodes d'apprentissage telles que : la méthode des séparateurs à vaste marges, les réseaux de neurones artificiels, ou encore la régression non linéaire.

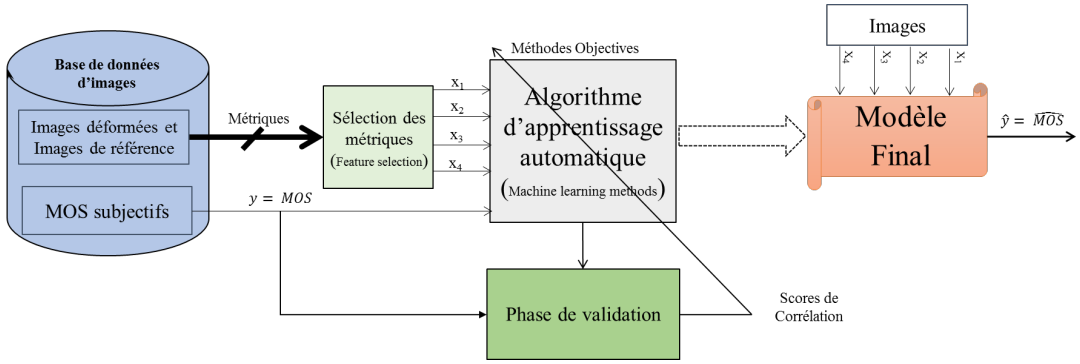


FIGURE 3.1 – Processus de l'évaluation objective à référence complète

Dans ce chapitre, sera présenté une méthode d'évaluation objective basée sur des algorithmes d'apprentissage automatique utilisant des caractéristiques extraites conjointement de l'image déformée et de l'image de référence. Ici, sera aussi présentée une implémentation de certains modèles résultants sur un système $FPGA$. Plusieurs méthodes d'apprentissage ont été proposées, parmi lesquelles les méthodes de classification telles que : l'analyse discriminante (DA), les k-plus proches voisins (KNN). Mais aussi des méthodes de régression telles que : les réseaux de neurones artificiels (ANN), les machines à vecteurs supports (SVM), la régression non-linéaire (NLR) polynomiale, les arbres de décision (DT), et la logique floue (FL). Un accent particulier est mis sur la méthode utilisant la technique de « logique floue ». Cette méthode proposée tout au long de ce travail produit meilleurs résultats que les autres méthodes d'apprentissage étudiées, en terme de corrélation avec l'évaluation subjective.

Le processus d'évaluation objectif présenté dans ce chapitre, est résumé par la Figure 3.1.

1. Il commence par une sélection d'une base de données d'images ; contenant un ensemble d'images originales et déformées ; mais aussi des scores subjectifs correspondant à chacune des images déformées. De ces images déformées, sont extraites un ensemble de caractéristiques (ou métriques).
2. La deuxième phase du processus consiste à sélectionner les métriques les plus indépendantes entre elles et fortement corrélées avec les scores subjectifs des images encore appelés MOS ou $DMOS$.
3. L'étape suivante est la division de la base de données en deux échantillons dont : un ensemble d'apprentissage utilisé pour la construction des différents modèles et un ensemble de test utilisé pour la validation des modèles.
4. Les métriques ainsi que les scores subjectifs ($MOS/DMOS$) de l'ensemble d'apprentissage sont utilisés comme entrées des algorithmes d'apprentissage automatique afin de permettre la construction des modèles. Une fois le modèle finalisé avec les paramètres optimaux déterminés, l'échantillon de test est utilisé pour valider le modèle. Le processus de test, quant à lui, ne prend en entrées que les métriques,

et produit une estimation du score objectif de qualité de l'image ($MOS - estimé$).

5. La phase de validation compare les scores subjectifs aux scores objectifs estimés par les différents modèles grâce aux indices de performance (coefficient de corrélation, erreur quadratique moyenne, ...) pour permettre de déterminer les modèles qui estiment au mieux le score subjectif, et donc la perception visuelle.

Dans la suite de ce chapitre, nous présenterons dans la Section 3.2, le processus de sélection des métriques indépendantes entre elles et fortement corrélées avec le score subjectif. La Section 3.3 quant à elle, présente la construction des modèles destinés à l'évaluation objective grâce aux algorithmes d'apprentissage, en prenant comme entrées les métriques sélectionnées. Dans la Section 3.4, sont présentés les divers résultats obtenus par différentes méthodes, sur des bases de données variées, ainsi que leur comparaison. Suivra la Section 3.5, qui présente les architectures d'implémentations sur un système $FPGA$, des deux modèles ayant les meilleurs résultats lors de la phase de simulation ; ainsi que les résultats obtenus lors de cette phase d'implémentation. Finalement la Section 3.6 fera un récapitulatif du chapitre, en présentant les apports dans le domaine de l'évaluation objective avec des métriques à références complètes, ainsi qu'un bilan du travail effectué dans ce chapitre.

3.2 Sélection des métriques

Dans cette section, les images déformées ainsi que leurs scores subjectifs (MOS) viennent de la base de données d'image TID2013. Trente-quatre (34) métriques avec référence, encore appelé indices de qualité, ont été extraites des images déformées. Les mesures ainsi extraites sont listées dans le Tableau 3.1. Ces mesures sont extraites des images suivant différentes approches :

— basées sur l'erreur quadratique moyenne ($IWMSE$, $PSNR$, ...) ;
— basées sur l'analyse des structures locales de l'image ($SSIM$, $MSSIM$, ...) ;
— inspirées du système visuel humain ($PSNR - HVS$, NQM, ...) ;
— utilisant des approches neuronales (SFF, MAD, ...) ;
— prenant aussi en compte la couleur ($FSIMc$, $PSNRc$, ...).

Les métriques ainsi évaluées sont pour certaines fortement corrélées entre elles. Les métriques fortement corrélées entre elles n'apportent pas d'information pertinente sur la qualité de l'image. Au contraire, elle rendent les modèles produits plus complexes. D'où la nécessité d'effectuer une sélection de métriques pertinentes pour en supprimer celles qui sont superflues. La sélection des métriques pertinentes permet de trouver parmi l'ensemble des métriques évaluées, celles qui sont indépendantes entre elles, tout en estimant au mieux la qualité des images. Deux approches de sélection des métriques ont été étudiées : la première basée sur le regroupement des métriques en blocs fortement corrélées entre elles avec la précaution de la sélection d'une seule métrique par groupe ; et la seconde est basée sur des apprentissages et la sélection progressive des métriques apportant le plus d'information sur la qualité des images au regard du score subjectif.

3.2.1 Sélection basée sur la corrélation mutuelle entre métriques

Dans cette approche de sélection des mesures pertinentes basée sur la corrélation mutuelle entre les métriques ; la première étape consiste en l'évaluation des corrélations mutuelles deux à deux entre toutes les métriques extraites des images. Ces corrélations mutuelles sont utilisées pour construire des classes d'équivalences de métriques ; neufs (9) classes ont été construites et présentées dans le Tableau 3.1. Dans chacune des classes d'équivalence, apparaissent les métriques qui ont une corrélation mutuelle entre elles supérieur ou égale à 90%, soit $cor \geq 0.9$; avec *cor* représentant une des corrélations de Pearson ($PLCC$), Spearman ($SRCC$), Kendall ($KRCC$), ou encore la distance de Brownian ($dCor$).

Dans le but de supprimer les redondances et donc d'avoir en entrée des algorithmes d'apprentissage automatique des variables « indépendantes », la deuxième étape de ce processus de sélection des métriques consiste à dégager dans chacune des classes d'équivalence, une seule métrique. Ici, le choix porte sur la métrique ayant la plus forte corrélation, en se basant sur la distance de Brownian, avec le score subjectif dans chaque groupe.

L'ultime étape de ce processus consiste en la suppression des métriques non corrélées avec le score subjectif, parmi celles sélectionnées dans chacune des classes d'équivalence. A présent, seules les métriques ayant une corrélation supérieure à un certain seuil ($dCor \geq 0,5$) sont conservées ; cela permet d'éliminer les métriques qui malgré le fait qu'elles soient indépendants par rapport aux autres métriques sélectionnées, n'apportent pas suffisamment d'informations sur la qualité des images.

Au final, les métriques sélectionnées sont donc : #19 ($PSNR$), #6 ($FSIMc$), #28 ($SSIM$), #9 ($IW-PSNR$), #20 ($PSNRc$) ; et un apprentissage avec un réseau de neurones artificiels (RNA) à une couche cachée, produit un modèle dont la performance, en terme de corrélation linéaire, est estimé à 89.37% entre les scores subjectifs (MOS) et les scores objectifs produits par le modèle ($MOS-estimé$).

Table 3.1 – Liste des métriques utilisées par groupe dépendant

# Gr.	# Mét.	Métriques	Description	DCor (MOS)
1	1	AD	Moyenne des différences	0.147
	17	**NCC**	**Corrélation mutuelle normalisée**	**0.331**
	25	SC	Contenu structurel	0.188
2	2	DCTune	Estimation de la qualité dans le domaine TCD[82]	0.498
	14	MSE	Erreur quadratique moyenne	0.602
	19	**PSNR**	**Rapport signal à bruit crête**	**0.624**
	23	PSNR-HVS	PSNR avec intégration de la notion de sensibilité fréquentielle du SVH [83]	0.075

TABLE 3.1 – Liste des métriques utilisées par groupe dépendant

# Gr.	# Mét.	Métriques	Description	DCor (MOS)
3	24	PSNR-HVS-M	PSNR-HVS intégrant un modèle de masquage [84]	0.076
	34	WSNR	Évaluation dans le domaine de Fourier, avec une pondération par une fonction CSF [85]	0.075
	3	DSI	Dissemblance entre les images [86]	0.636
	5	FSIM	Mesure basée sur les similitudes des caractéristiques [87]	0.827
	6	**FSIMc**	**FSIM intégrant les caractéristiques de couleur [87]**	**0.839**
	10	IW-SSIM	Similarités structurelles intégrant la pondération des informations [88]	0.788
	11	MAD	Mesure basée sur les modèles de masquage et de filtrage des caractéristiques locales [25]	0.802
	13	MSDDM	Mesure de l'auto-similarité [89]	0.736
	15	MSSIM	Mesure des similarités structurelles à échelles multiples [90]	0.799
	18	NQM	Mesure de la qualité du bruit, prenant en compte un phénomène de masquage dans le SVH [91]	0.673
	21	PSNR-HA	PSNR prenant en compte la sensibilité au contraste CSF [92]	0.808
	22	PSNR-HMA	PSNR-HA prenant en compte un phénomène de masquage dans le domaine TCD [92]	0.803
	26	SFF	Mesure basée sur les caractéristiques manquantes entre les images [93]	0.826
	27	SR-SIM	Mesure basée sur la similitude des résidus spectraux [94]	0.834
	31	VIF	Basée sur l'information mutuelle dans le domaine des ondelettes, avec normalisation des images [10]	0.698
	32	VIFP	VIF dans le domaine spatial [10]	0.794
	33	VSNR	SNR Visuel, basée sur l'analyse des coefficients d'ondelettes [95]	0.229
4	4	CBM	Mesure basée sur les contenus	0.633

TABLE 3.1 – Liste des métriques utilisées par groupe dépendant

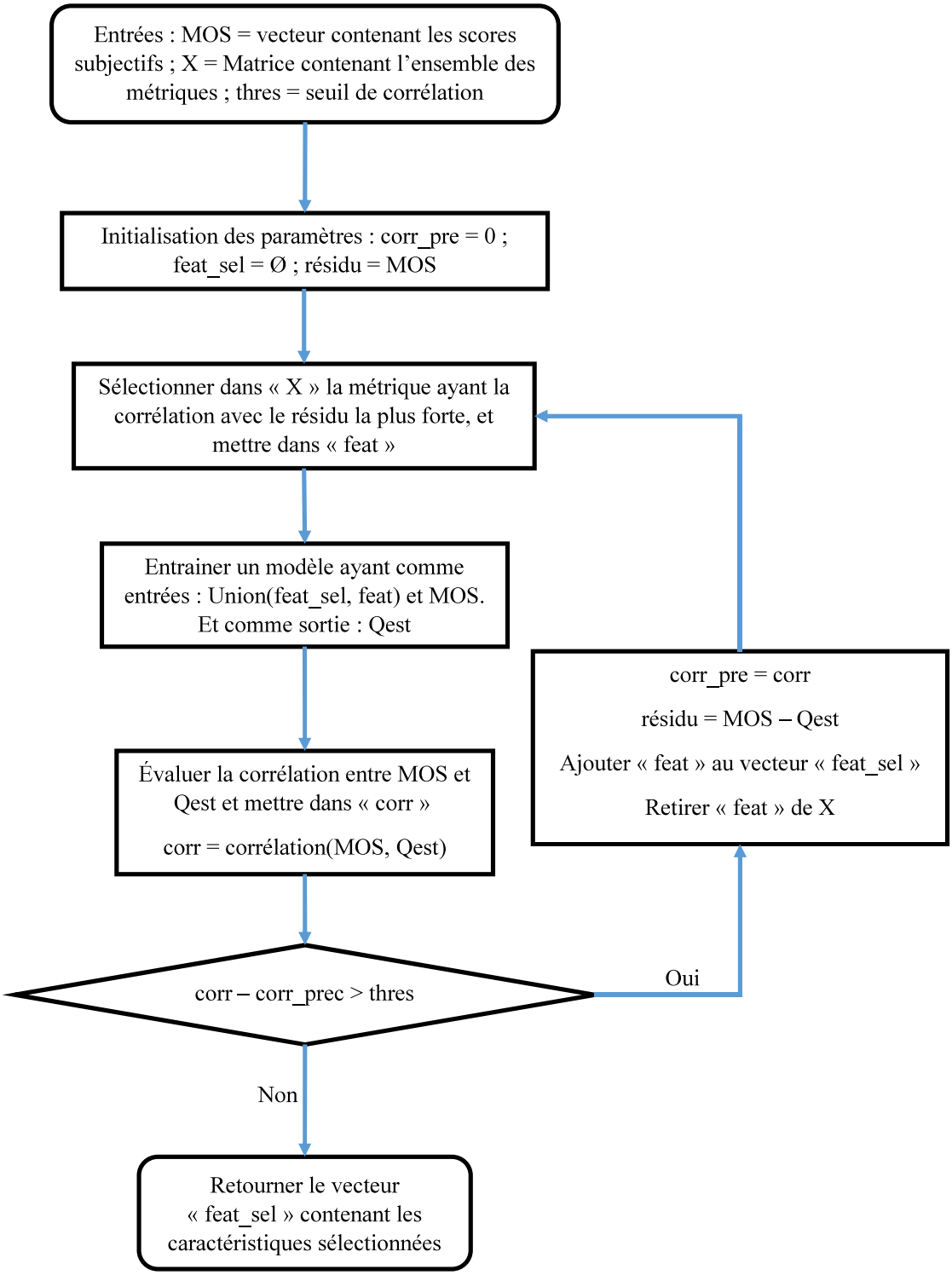
# Gr.	# Mét.	Métriques	Description	DCor (MOS)
	7	IFC	Basée sur l'information mutuelle dans le domaine des ondelettes [61]	0.308
	28	**SSIM**	**Similarité de luminance, de contraste et de structure [5]**	**0.658**
	30	UQI	Index de qualité universel basé sur une analyse locale des moments [96]	0.580
5	8	IW-MSE	MSE intégrant la pondération des informations [88]	0.590
	9	**IW-PSNR**	**PSNR intégrant la pondération des informations [88]**	**0.695**
6	**12**	**MD**	**Mesure basée sur la différence entre les images**	**0.485**
7	**16**	**NAE**	**Mesure basée sur l'erreur absolue**	**0.484**
8	**20**	**PSNRc**	**PSNR intégrant les caractéristiques des couleurs [97]**	**0.675**
9	**29**	**SVD**	**Mesure basée sur la décomposition en valeur singulière [98]**	**0.125**

3.2.2 Sélection basée sur l'apprentissage

La seconde approche proposée, quant à elle, est basée sur un processus d'apprentissage récursif et itératif permettant de sélectionner progressivement les métriques qui ajoutent suffisamment d'informations sur la qualité d'image aux métriques déjà sélectionnées, à chaque nouvelle itération. Cette approche de sélection de métriques est décrite dans l'Algorithme 3.1. Le seuil de performance choisi est de 0.01, et l'indice de performance est la distance corrélation de Brownian ($dCor$). Plusieurs méthodes d'apprentissage automatique parmi lesquelles, les arbre de décision, les réseaux de neurones, la régression non-linéaire polynomiale, pour ne citer que ceux-là, ont été testées pour effectuer l'entraînement des modèles dans l'algorithme. Les résultats au sortir de l'expérience, sont approximativement les mêmes.

Au final, les métriques #6 ($FSIMc$), #20 ($PSNRc$), #16 (NAE), et #17 (NCC) ont été sélectionnées dans cet ordre. L'apprentissage avec un réseau de neurones artificiel à une couche cachée produit un modèle, dont la performance en terme de corrélation linéaire est estimé à 90.12% entre le score subjectif (MOS) et le score objectif produit par le modèle ($MOS - estimé$).

Algorithme 3.1 Procédure de sélection de métriques basée sur l'apprentissage



3.2.3 Description des métriques sélectionnées

La méthode de sélection basée sur la corrélation entre les métriques a permis d'en sélectionner cinq (05). Celle basée sur l'apprentissage récursif et itératif a, quant à elle, permis d'en sélectionner quatre (04). Les différentes métriques ainsi sélectionnées sont décrites dans la suite de cette section.

— **PSNR** : le rapport signal sur bruit de crête est l'une des mesures de distorsions et notamment de la compression, les plus utilisées pour mesurer la qualité d'une image. Ce rapport est simple à évaluer voire à implémenter, et s'applique bien pour des applications temps-réels. Le $PSNR$ est effectué sur une image en niveau de gris ; tandis que, le **PSNRc** qui est la version en couleur du $PSNR$, est calculé en prenant en compte les trois dimensions liées à la couleur. La métrique **IW-PSNR** quant à elle rajoute un facteur de pondération au $PSNR$ classique, des informations entre l'image originale et l'image dégradée. La métrique $PSNR$ donne de bons résultats dans le cas de distorsions aléatoires et étalées dans le signal, mais s'avère inefficace dans le cas de dégradations structurelles localisées c'est-à-dire non corrélée à l'appréciation subjective [9]. Elle est basée sur l'erreur quadratique moyenne, et peut se calculer comme présentée dans (3.1).

$$PSNR = 10log(\frac{d^2}{MSE}) \tag{3.1}$$

où d représente le maximum de l'intensité dans l'image (255 dans le cas d'une image codée sur 8 bits) ; etMSE représente l'erreur quadratique moyenne, donnée par (3.2).

$$MSE = \frac{1}{M.N}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_o(i,j) - I_d(i,j))^2 \tag{3.2}$$

avec $I_o(i,j)$ et $I_d(i,j)$ représentant les intensités du pixel (i,j) de l'image originale et de sa version dégradée, respectivement ; M et N représentent le nombre de lignes et de colonnes de l'image, respectivement.

— **SSIM** : il s'agit d'une mesure de la qualité d'une image basée sur l'analyse des similarités entre les informations structurelles locales des images. Pour évaluer cette mesure, il faut tout d'abord subdiviser chacune des images originale et dégradée en bloc de même taille ; puis évaluer les coefficients de corrélation de luminance tels que présentés dans (3.3), de contraste tel que présenté dans (3.4) et de structure telle que présentée dans (3.5) pour le bloc original et son correspondant dans l'image dégradée. Enfin l'index global est déterminé comme présenté dans (3.6).

$$l(i) = \frac{2 \cdot \overline{x(i)} \cdot \overline{y(i)} + C_1}{\overline{x(i)}^2 + \overline{y(i)}^2 + C1} \tag{3.3}$$

$$c(i) = \frac{2 \cdot \sigma_x(i) \cdot \sigma_y(i) + C_2}{\sigma_x^2(i) + \sigma_y^2(i) + C_2} \tag{3.4}$$

$$s(i) = \frac{2 \cdot \sigma_{xy}(i) + C_3}{\sigma_x(i) \cdot \sigma_y(i) + C_3} \tag{3.5}$$

$$SSIM = \frac{1}{W}\sum_{i=1}^{W} l(i)\cdot c(i)\cdot s(i) \tag{3.6}$$

avec $C_1 = (L.K_1)^2$; $C_2 = (L.K_2)^2$; $C_3 = \frac{C_2}{2}$; où K_1 et K_2 sont des constantes, fixées à 0.01 et 0.03 respectivement. W représente le nombre de blocs dans l'image ; $\underline{x(i)}$ et $\underline{y(i)}$ représentent les blocs i pour l'image originale et dégradée, respectivement ; $\overline{x(i)}$ et $\overline{y(i)}$ représentent la moyenne des ces blocs ; $\sigma_x(i)$ et $\sigma_y(i)$ représentent leurs écarts types ; et $\sigma_{xy}(i)$ représente la covariance du bloc i.

— **FSIMc** : la mesure de la similitude des caractéristiques d'une image $FSIM$ est basée sur le fait que la perception visuelle humaine (SVH). Elle est fortement influencée par la variation des caractéristiques à faible niveau dans l'image ; principalement les caractéristiques de congruence de phase (PC) et d'amplitude du gradient (GM). Ces deux caractéristiques complémentaires (PC et GM) jouent un rôle très important dans la caractérisation de la qualité locale d'une image ; d'où leur utilisation par la métrique $FSIM$ pour l'évaluation de la qualité de l'image. $FSIMc$ est tout simplement une extension de $FSIM$ prenant en compte la dimension de couleur de l'image. L'évaluation de cette mesure passe par la construction de la matrice de congruence de phase de l'image « PC » telle que présentée dans [99] et la construction de la matrice d'amplitude du gradient « GM » basée sur les filtres de convolution, telle que présentée dans [100]. Les matrices RGB sont converties en matrices YIQ en utilisant la matrice de conversion définie dans [101]. L'index global de la mesure $FSIMc$ est donné par (3.7).

$$FSIMc = \frac{\sum_{i\in W} S_{PC}(i)\cdot S_G(i)\cdot [S_I(i)\cdot S_Q(i)]^{\lambda}\cdot PC_m(i)}{\sum_{i\in W} PC_m(i)} \tag{3.7}$$

avec W représentant l'ensemble du domaine spatial de l'image ; $\lambda > 0$ est un paramètre utilisé pour ajuster l'importance des composantes de couleur ; $PC_m(i)$ est le maximum entre $PC_x(i)$ et $PC_y(i)$. $PC_x(i)$ et $PC_y(i)$ étant les composantes i des matrices de congruence de phase (PC) des images originale et dégradée, respectivement. Les $S_A(i)$ sont des coefficients de corrélation calculés comme dans (3.8).

$$S_A(i) = \frac{2\cdot A_x(i)\cdot A_y(i) + T_A}{A_x^2(i) + A_y^2(i) + T_A} \tag{3.8}$$

— **NAE** : l'erreur absolue normalisée permet d'évaluer la qualité de l'image grâce à une formule simple basée sur la somme des écarts des pixels entre l'image originale et l'image dégradée, normalisée par la somme des pixels de l'image originale. Cette mesure est donnée par (3.9).

$$NAE = \frac{\sum_{i,j=1}^{M,N} |I_o(i,j) - I_d(i,j)|}{\sum_{i,j=1}^{M,N} I_o(i,j)} \tag{3.9}$$

— **NCC** : Cette métrique mesure la qualité d'une image dégradée en se basant sur la somme des corrélations croisées entre l'image dégradée et l'image originale, normalisée par la somme des carrés des pixels de l'image originale. Elle se calcule telle que présentée dans (3.10).

$$NCC = \frac{\sum_{i,j=1}^{M,N} I_o(i,j) \cdot I_d(i,j)}{\sum_{i,j=1}^{M,N} (I_o(i,j))^2} \tag{3.10}$$

3.3 Méthode objective avec référence basée sur l'apprentissage

La base de données ($TID2013$) et les métriques de qualité des images ($FSIMc$, $PSNRc$, NAE, et NCC notées par la suite $x1$, $x2$, $x3$, et $x4$; respectivement) ayant été sélectionnées, la prochaine étape consiste en la construction des mesures globales d'évaluation de la qualité des images, à partir des modèles issus des méthodes d'apprentissage automatique. L'index résultant a été nommé $GFRIQ - ML$ [1] (Global Full-Reference Image Quality index based on Machine Learning). Plusieurs méthodes d'apprentissage ont été étudiées, et les différents paramètres de construction des divers modèles retenus sont étudiés dans cette section. Dépendant du type de la variable aléatoire de sortie du modèle, il peut s'agir d'un modèle de classification (données de sorties discrètes) : l'analyse discriminante (DA), les k-plus proches voisins (KNN) ; ou d'un modèle de régression (données de sorties continues) : réseaux de neurones artificiels (ANN), les machines à vecteurs supports (SVM), la régression non linéaire (NLR) polynomiale, les arbres de décision (DT), et la logique floue (FL). La phase de test utilise 30% de l'ensemble des données de la base de données d'image $TID2013$, les 70% autre ayant été utilisés pour l'apprentissage des modèles.

3.3.1 GFRIQ basée sur les méthodes de classification

L'outil basé sur la classification utilise une échelle à 10 niveaux (de 0 à 9), où le niveau 0 désigne une qualité d'image très mauvaise, et le niveau 9 une qualité d'image excellente. Il permet ici de décrire, expliquer et prédire l'appartenance des images déformées prises dans une base de données d'images (tel que $TID2013$), à l'un des 10 groupes (entre 0 et 9) décrivant le niveau de la qualité de l'image. Cette description est faite sur la base des quatre variables prédictives $x1$, $x2$, $x3$, et $x4$, dont la notation simplifiée dans les sections suivantes est X.

3.3.1.1 GFRIQ-DA

$GFRIQ - DA$ (Global Full-Reference Image Quality index based on Discriminant analysis) est un index objectif qui permet d'évaluer la qualité d'une image en se basant sur l'analyse discriminante. Le modèle final de cet index a été construit à partir de l'outil $MATLAB$. Cet outil a permis d'effectuer un apprentissage des fonctions discriminantes permettant de séparer au mieux les différents groupes en fonction des valeurs prises par les caractéristiques en entrées de l'algorithme (contenues dans la variable X). Pour un apprentissage sur la base de données $TID2013$, 8 groupes de classification (entre

1. Il s'agit du nom donné à notre métrique globale d'évaluation objective avec référence de la qualité d'image.

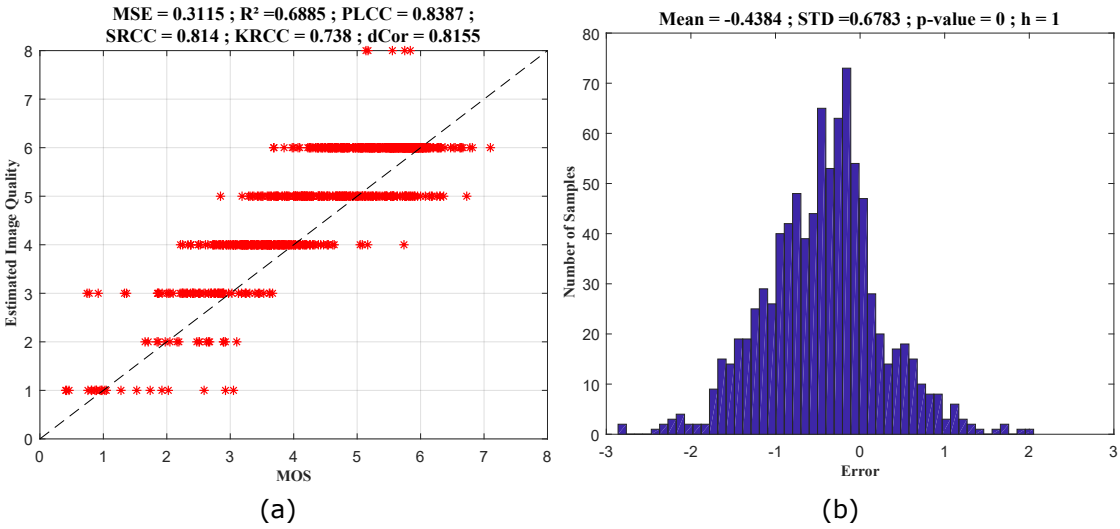


FIGURE 3.2 – Performance du GFRIQ-DA : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

1 et 8) ont pu être détectés, chaque groupe étant séparé des autres par une fonction discriminante linéaire sous la forme (3.11).

$$D_{ij} = \lambda_{ij0} + \lambda_{ij1}x_1 + \lambda_{ij2}x_2 + \lambda_{ij3}x_3 + \lambda_{ij4}x_4 \tag{3.11}$$

où λ_{ijk} pour $k = \{1, 2, 3, 4\}$ sont les coefficients de la fonction discriminante entre la classe i et j. Le modèle final obtenu est donc constitué d'une matrice à trois dimensions de taille $8 \times 8 \times 5$, contenant les coefficients des différentes fonctions discriminantes entre tous les groupes.

Le modèle produit par le $GFRIQ - DA$ effectue une bonne classification dans l'ensemble. Les résultats de la phase de test montrent une corrélation de linaire de 83.87% et une erreur quadratique moyenne de 31.15%, entre le score subjectif (MOS) et le score donné par le classificateur de l'étude basé sur l'analyse discriminant, sur les données de test. La Figure 3.2 présente dans son graphique (a) la comparaison entre le score subjectif (MOS) en abscisse, et les classes estimées ou encore qualité d'image estimée par le modèle en construit basé sur l'analyse discriminante, en ordonné. Ce graphique présente aussi les indices de performances tels que : l'erreur quadratique moyenne (MSE), le coefficient de détermination (R^2), les coefficients de corrélation de Pearson ($PLCC$), Spearman ($SRCC$), et Kendall ($KRCC$), et la corrélation de distance de Brownian ($dCor$) ; entre le score subjectif (MOS) et la qualité objective d'image produite par l'index ($MOS - estimé$). Le graphique (b), quant à lui, présente un histogramme de répartition des erreurs entre le score subjectif (MOS) et la qualité d'image estimée par l'index basé sur l'analyse discriminante ($MOS - estimé$). Il présente aussi la moyenne ($Mean$), et l'écart type (STD) de l'ensemble des erreurs. Le test du Khi-deux a été effectué sur l'ensemble des erreurs entre les scores subjectifs (MOS) et la qualité d'image

estimée par l'index retenu, pour déterminer si elles sont distribuées suivant une loi normale. L'hypothèse nulle est alors la suivante : « les erreurs sont réparties suivant une loi normale ». La probabilité de rejeter l'hypothèse nulle si elle est vraie, aussi appelée « valeur p » ou « $p - value$ » de ce test, ainsi que le résultat h du test d'hypothèse sont donnés dans le graphe (b) de la Figure 3.2. h donne le résultat du test d'hypothèse à 5% : si $h = 0$ les erreurs sont distribuées suivant une loi normale, si $h = 1$ elles ne suivent pas une loi normale. Dans le cas de cet index $GFRIQ - DA$ basé sur l'analyse discriminante, le test du Khi-deux montre que les erreurs entre les scores subjectifs et les qualités d'images produites par l'index ne sont pas distribuées suivant une « loi normale », avec une « $p - value$ » sensiblement nulle.

3.3.1.2 GFRIQ-KNN

$GFRIQ - KNN$ (Global Full-Reference Image Quality index based on k-Nearest Neighbors) est le second index d'évaluation de la qualité des images, basé sur la méthode des « k-plus proches voisins », qui est une méthode de classification. Le modèle final est constitué des données d'apprentissage qui sont les différentes valeurs des quatre métriques d'entrées, regroupées en 8 classes en fonction de la valeur de leur score subjectif (MOS). L'évaluation d'une nouvelle image consiste en la recherche de ces « k » plus proches voisins dans l'ensemble des données d'apprentissage. La classe de l'image à classifier est celle la plus représentée dans l'ensemble des voisins sélectionnés. Les voisins les plus proches sont ceux qui ont les distances les plus faibles entre eux et le point à classifier. La distance utilisée dans l'évaluation de la qualité des images est la distance Euclidienne donnée par (3.12).

$$d(X_i, X_j) = \sqrt{\sum_{k=1}^{4} (x_{ik} - x_{jk})^2} \tag{3.12}$$

où les x_{ik} pour $k = \{1, 2, 3, 4\}$ représentent les valeurs prédictives de l'image pour i.

La Figure 3.3 présente les résultats obtenus sur la base de données d'images $TID2013$ par le modèle $GFRIQ - KNN$ basé sur la méthode des k-plus proches voisins. Elle est constituée du graphique (a) qui donne une comparaison graphique, mais aussi un ensemble d'indices de performance, entre le score subjectif (MOS) et le score estimé par le classificateur. Tandis que le graphique (b) de cette Figure 3.3 montre l'histogramme des erreurs, et les résultats du test de Khi-deux sur la distribution de ces erreurs entre le score subjectif (MOS) et le score estimé de la qualité de l'image donné par le classificateur basé sur la méthode des k-plus proches voisins. Ces résultats montrent une corrélation linéaire de 79.67% et une erreur quadratique moyenne de 41.18% entre le MOS et le score estimé donné par le classificateur. Les erreurs ne sont pas distribuées suivant une loi normale, et la « $p - value$ » est sensiblement nulle.

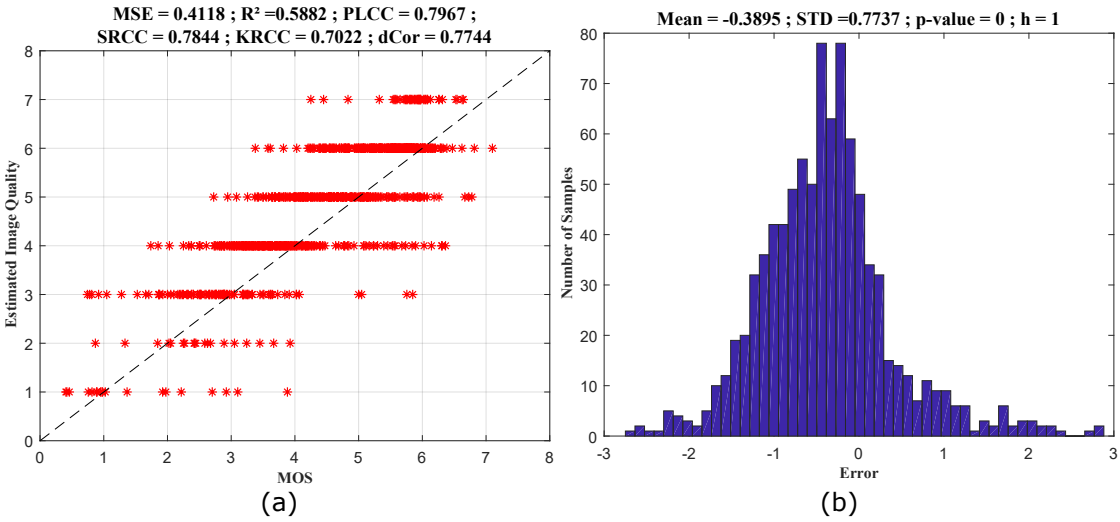


FIGURE 3.3 – Performances du GFRIQ-KNN : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

3.3.2 GFRIQ basée sur les réseaux de neurones artificiels

$GFRIQ - ANN$ (Global Full-Reference Image Quality index based on Artificial Neural Network) est le modèle d'évaluation de la qualité d'image basé sur la méthode de régression des réseaux de neurones artificiels. Un outil d'évaluation de la qualité des vidéos basé sur les réseaux de neurones artificiels a été implémenté dans [7] ; il utilise la base de données de vidéos $EPFL$ [102, 103] contenant 78 séquences de vidéos déformées. Le $GFRIQ - ANN$ prend en entrées les valeurs des quatre mesures de qualité d'image sélectionnées dans la section précédente, représentées ici par $x1$, $x2$, $x3$, et $x4$; et retourne en sortie le score de qualité d'image estimé par le modèle. Ce modèle a une architecture à trois couches pour simplifier son implémentation sur des systèmes à basses capacités : une couche d'entrée à 5 neurones, représentant les valeurs de des variables d'entrées et un biais b_0 ; une couche cachée comportant un neurone qui effectue une somme pondérée des entées avant de les passer dans une fonction d'activation sigmoïde ; et une couche de sortie prenant en entrée, la sortie de la couche cachée et un biais b_1, puis effectue une somme et passe par une fonction d'activation linéaire, avant de produire en sortie le score objectif de qualité d'image estimé par le modèle.

L'architecture finale du modèle obtenu est présentée dans la Figure 3.5. Les différents coefficients de ce modèle ($b_0 = 1.03$, $b_1 = 0.46$, $W_{11} = -1.22$, $W_{12} = -0.32$, $W_{13} = -0.30$, $W_{14} = -0.22$, $W_{21} = -1,05$) ont été déterminés lors de la phase d'apprentissage, déployée grâce à l'outil $MATLAB$.

Les résultats de simulation sur les données de tests de la base de données d'image $TID2013$, sont présentés dans la Figure 3.5. Où le graphe (a) montre une comparaison non seulement graphique, mais aussi grâce aux indices de performance, entre le score subjectif (MOS) et la qualité d'image estimée par l'index $GFRIQ - ANN$. Le graphe

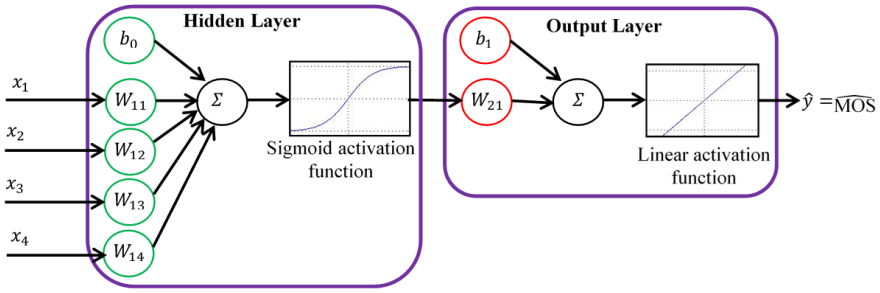


FIGURE 3.4 – Architecture du modèle GFRIQ-ANN

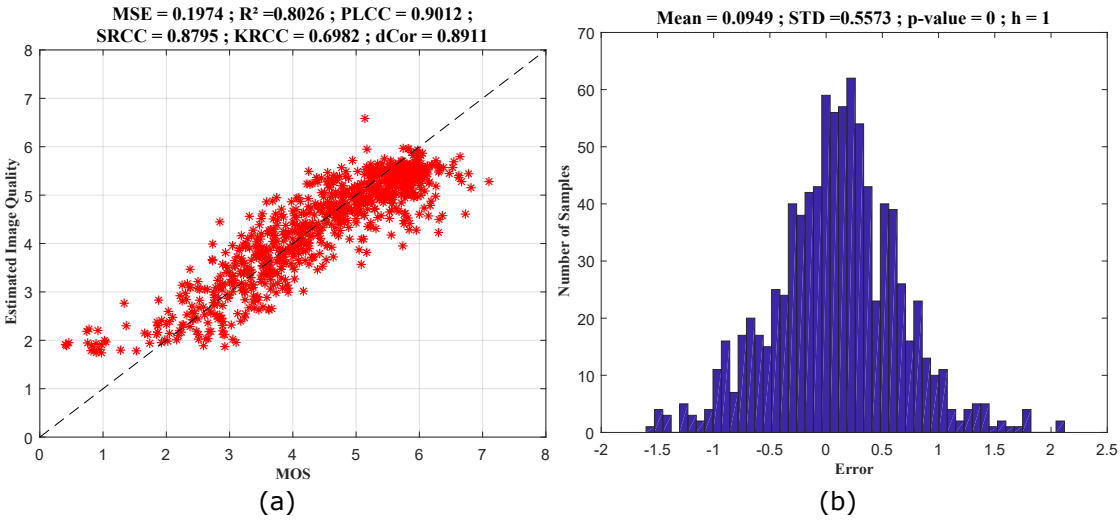


FIGURE 3.5 – Performances du GFRIQ-ANN : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

(b), quant à lui, montre l'histogramme des erreurs, et les résultats du test de Khi-deux sur la distribution de ces erreurs entre le MOS et le score estimé par l'index. Les résultats obtenus montrent une corrélation linéaire de 90.12% et une erreur quadratique moyenne de 19.74% entre le score subjectif et le score estimé par l'index basé sur les réseaux de neurones artificiel. Les erreurs ne suivent pas une distribution « normale », et la « $p - value$ » est sensiblement nulle.

3.3.3 GFRIQ basée sur la régression non-linéaire

Dans un algorithme d'apprentissage basé sur la régression non linéaire, le but est de trouver la courbe qui permet d'estimer au mieux une variable dépendante en fonction de variables indépendantes. D'après le théorème de Stone-Weierstrass [104] sur la densité des fonctions polynomiales dans l'univers des fonctions, toute fonction continue définie sur un segment peut être approchée uniformément par des fonctions polynomiales. En considérant ce théorème, l'évaluation objective de la qualité des images basée sur la régression non linéaire est basée uniquement sur les fonctions polynomiales. L'évaluation des coefficients optimaux du polynôme se fait en utilisant la méthode des moindres carrées, qui peut se simplifier par (3.13).

$$\theta = \left(X^T * X\right)^{-1} X^T * Y \tag{3.13}$$

où X est la matrice contenant les métriques d'entrées de dimension (N, M), avec N étant la taille de l'échantillon d'apprentissage utilisé et $M = 4$ étant le nombre de métriques sélectionnées. Y est un vecteur de taille N contenant les scores subjectifs pour les données d'apprentissage.

$GFRIQ - NLR$ (Global Full-Reference Image Quality index based on Non-Linear Regression) est le modèle d'évaluation de la qualité d'image, basé sur la méthode de régression non-linéaire polynomiale. Dans cet outil, ont été pris en compte les monômes du premier (sous la forme : $a_p x_i$), deuxième (sous la forme : $b_p x_i x_j$) et troisième ordre (sous la forme : $c_p x_i x_j x_k$) uniquement. Où a_p, b_p et c_p et la constante de normalisation de l'équation a_0 sont les coefficients des monômes déterminés grâce à la méthode des moindres carrés, et les x_i sont les métriques sélectionnées plus haut, avec i, j et k $\in\{1, 2, 3, 4\}$. Le modèle $GFRIQ - NLR$ est donc représenté par une équation sous la forme de (3.14).

$$\widehat{y} = a_0 + \sum a_p x_i + \sum b_p x_i x_j + \sum c_p x_i x_j x_k \tag{3.14}$$

Les monômes avec des coefficients très faibles sont supprimés de l'équation, car ils alourdissent les calculs, et n'apportent pas d'informations importantes sur le résultats ; ils sont négligeables par rapport aux autres monômes. Au final, sur les 35 monômes possibles, seuls 16 monômes (4 monômes du premier ordre, 5 monômes du second ordre et 7 monômes du troisième ordre) ont été retenus dans la construction du modèle final.

Les graphes (a) et (b) de la Figure 3.6 présentent les résultats de simulation sur un ensemble de données de test de la base de données d'images $TID2013$. Le graphe (a) présente la comparaison entre le score subjectif (MOS) et la qualité d'image estimée par

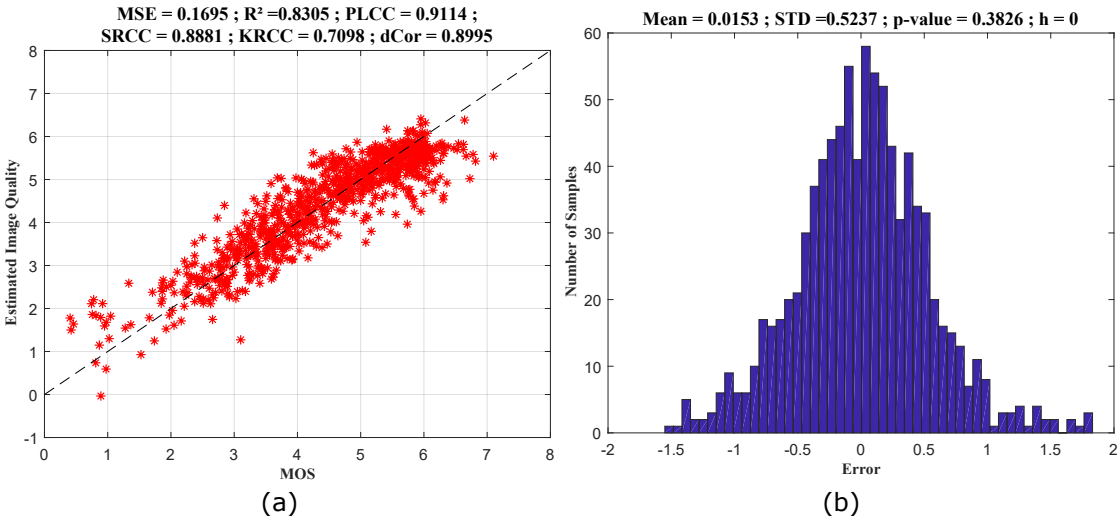


FIGURE 3.6 – Performances du GFRIQ-NLR : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

le modèle $GFRIQ - NLR$, tandis que le graphe (b) présente l'histogramme des erreurs entre le score subjectif et la qualité de l'image estimée par l'index. Les résultats obtenus par l'index basé sur la régression non-linéaire polynomiale dénotent une corrélation linéaire de 91.14% et une erreur quadratique moyenne de 16.95% entre le score subjectif et le score de qualité estimé par l'index. Ces résultats montrent aussi que les erreurs entre le score subjectif et la qualité d'image estimée par le modèle $GFRIQ - NLR$ suivent une loi normale de moyenne $m = 0.0153$ et d'écart-type $\sigma = 0.5237$, avec une $p - value = 0.3826$.

3.3.4 GFRIQ basée sur les arbres de décision

Un arbre de décision est un outil d'aide à la décision qui permet de décrire une variable dépendante en fonction de plusieurs variables indépendantes, sous forme graphique grâce à un arbre binaire. Le $GFRIQ - DT$ (Global Full-Reference Image Quality index based on Decision Tree) est le modèle d'évaluation de la qualité des images, basé sur un arbre de décision. Les feuilles de l'arbre représentent les différents scores de qualité possibles des images ; les nœuds de décision représentent les métriques de qualité sélectionnées ; et les branches sont les différents seuils de comparaison des métriques.

Les branches, les nœuds, ainsi que les feuilles de l'arbre de décision utilisé dans la mesure $GFRIQ - DT$ sont déterminés lors de la phase d'apprentissage sur l'outil $MATLAB$, avec des données d'apprentissage issues de la base de données d'images $TID2013$. L'arbre ainsi constitué est un arbre binaire à 21 niveaux, tel que présenté dans la Figure 3.7.

Les résultats de simulation sur les données de tests de la base de données d'images $TID2013$, sont présentés dans la Figure 3.5. Où le graphe (a) montre une comparaison

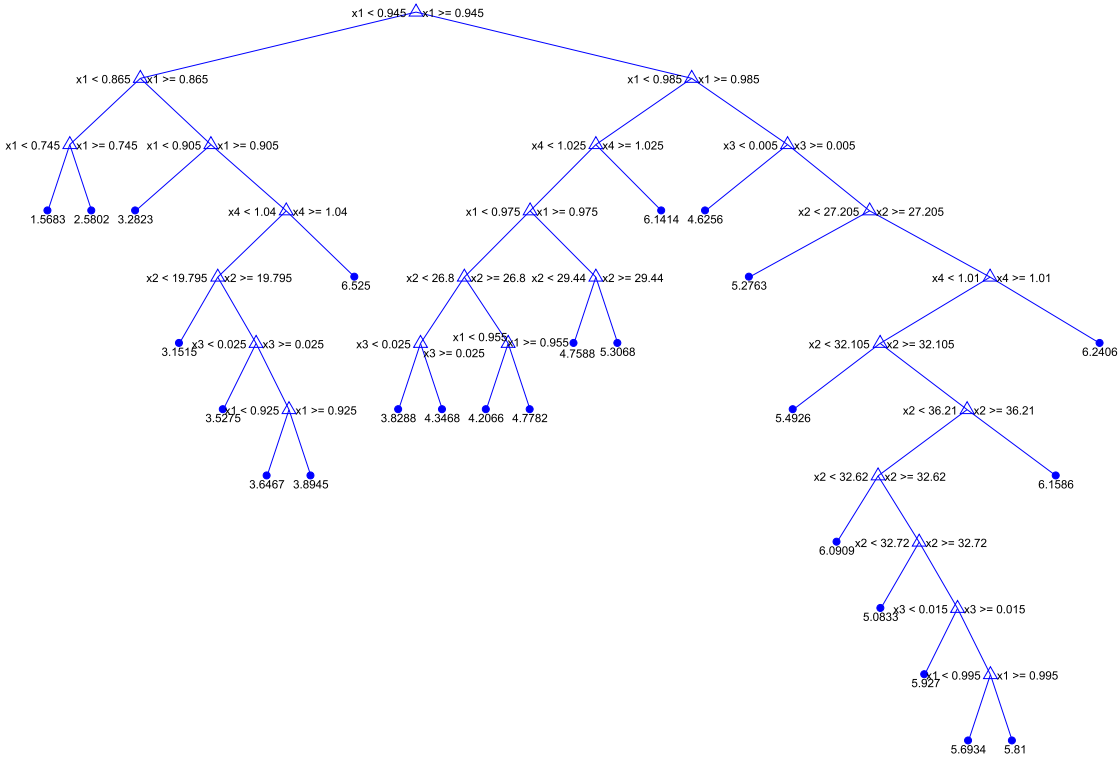


FIGURE 3.7 – Structure d'arbre du GFRIQ-DT

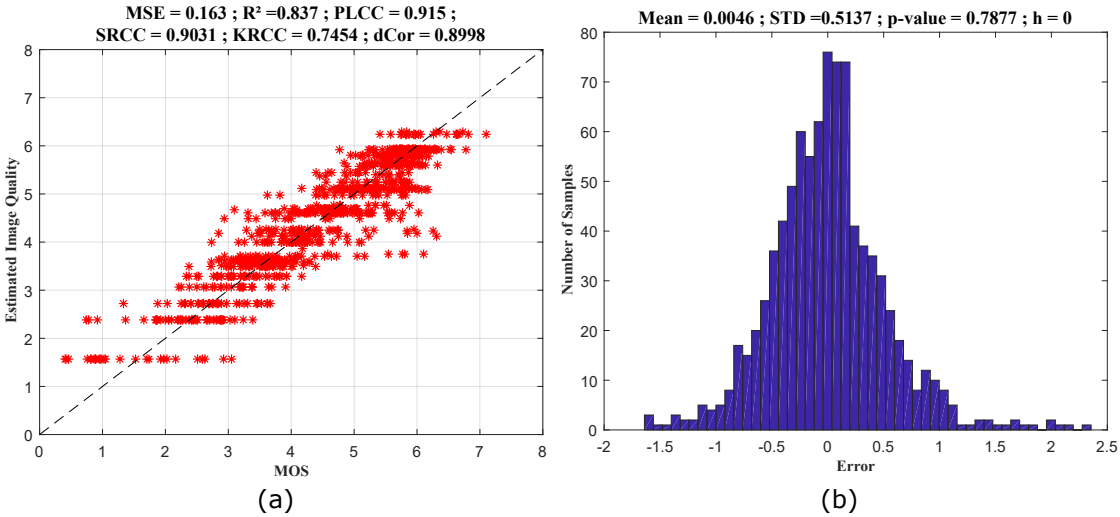


FIGURE 3.8 – Performances du GFRIQ-DT : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

non seulement graphique, mais aussi grâce aux indices de performance, entre les scores subjectifs (MOS) et la qualité d'image estimée par l'index $GFRIQ-DT$. Le graphe (b) quant à lui montre l'histogramme des erreurs, et les résultats du test de Khi-deux sur la distribution de ces erreurs entre les scores subjectifs (MOS) et les scores de qualité d'image estimés par l'index. Les résultats obtenus par cet index basé sur les arbres de décision présentent une corrélation linéaire de 91.50% et une erreur quadratique moyenne de 16.30% entre le score subjectif et le score de qualité estimé par l'index $GFRIQ-DT$. Ces résultats montrent aussi que les erreurs entre le score subjectif et la qualité d'image estimée par le modèle $GFRIQ-DT$ suivent une loi normale de moyenne $m=0.0046$ et d'écart-type $\sigma=0.5137$, avec une $p-value=0.7877$.

3.3.5 GFRIQ basée sur la logique floue

La logique floue permet d'avoir un raisonnement proche de celui de l'être humain, grâce à l'intégration des fonctions d'appartenances qui permettent de définir à quel point une assertion appartient à un ensemble. En comparaison, dans la logique classique, une assertion appartient complètement ou pas à un ensemble tout simplement. L'évaluation des images est un problème non-linaire qui peut être modélisé grâce à la méthode de logique floue. Ce processus de modélisation passe par la construction d'un système d'inférence flou (FIS), qui prend en entrée les quatre variables sélectionnées dans la Section 3.2, et produit en sortie un score objectif de la qualité de l'image.

Le $GFRIQ-FL$ (Global Full-Reference Image Quality index based on Fuzzy Logic) est le modèle d'évaluation de la qualité d'image basé sur la logique floue. Il est basé sur le modèle d'inférence flou de Sugeno [105], et est composé de règles « $SI-ALORS$ », sous forme de fonctions linéaires ayant la forme de l'équation présentée dans (3.15).

$$\begin{array}{l} \textit{Règle } i: \\ \end{array} \begin{array}{c} si\ x_1 \in A_{i,1}\ ,\ x_2 \in A_{i,2}\ ,\ x_3 \in A_{i,3}\ ,\ and\ x_4 \in A_{i,4} \\ alors\ f_i = a_{i0} + a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + a_{i4}x_4 \end{array} \tag{3.15}$$

où $A_{i,j}$ représente la variable linguistique (variable d'entrée, plage de valeurs et sous-ensembles flous) ou fonction d'appartenance de la variable d'entrée j pour la règle i. f_i est la sortie de la règle i, qui est une combinaison linéaire des variables d'entrées par des coefficients a_{ij} déterminés lors de phase d'apprentissage.

Le modèle $GFRIQ-FL$ est construit grâce à un outil qui est une combinaison entre les réseaux de neurones artificiels (ANN) et le système d'inférence flou (FIS), appelé $ANFIS$ (Adaptive Neuro Fuzzy Inference System) [106, 107]. Cet outil $ANFIS$ est utilisé pour construire le modèle $GFRIQ-FL$ à partir des données d'entraînements issues de la base de données TID2013. Il s'agit d'une technique d'apprentissage récursive et itérative, presque identique à un apprentissage d'un réseau de neurones, utilisé dans le but de construire les fonctions d'appartenance, et les règles sous la forme des « $SI-ALORS$ », qui permettent de minimiser l'erreur entre le score subjectif pris dans la base de données d'images, et les scores de qualité des images évalués par notre modèle.

Une architecture typique en couche de l'outil $ANFIS$ avec quatre variables d'entrées, deux règles et une variable de sortie, est présentée dans la Figure 3.9. Il s'agit d'une

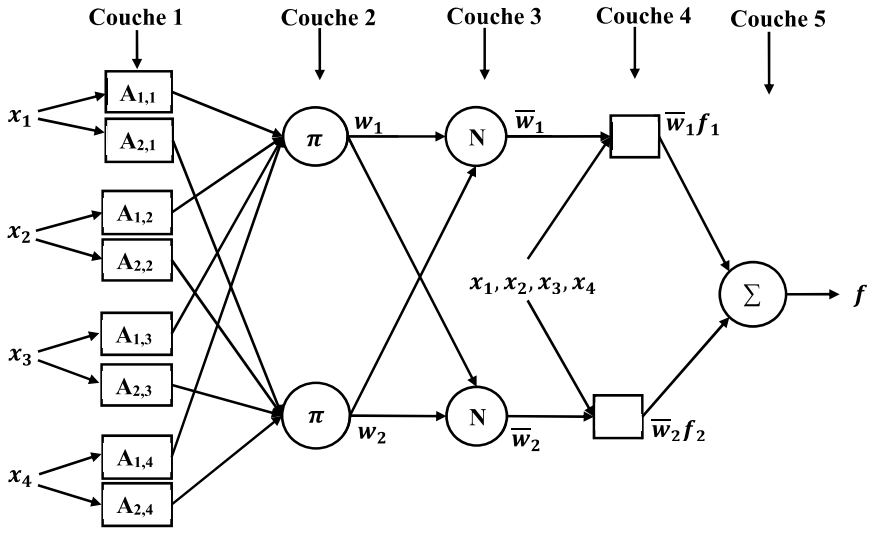


FIGURE 3.9 – Architecture ANFIS à deux règles

architecture composée de 5 couches :

— Couche 1 : la première couche est l'évaluation du degré d'appartenance de chaque variable d'entrée. Lors de la phase d'apprentissage, les fonctions d'appartenance sont construites pour chacune des variables d'entrées du système ; et lors de la phase de test du modèle, on utilise ces différentes fonctions d'appartenance pour déterminer les valeurs floues des valeurs des variables prises en entrées. Chaque nœud de cette couche se représente sous la forme de (3.16).

$$O_{i,j}^1 = \mu_{A_{i,j}}(x_j) \tag{3.16}$$

où $O_{i,j}^1$ est la fonction d'appartenance de $A_{i,j}$ qui donne le degré d'appartenance (entre 0 et 1) de la variable d'entrée x_j ; avec $j \in \{1, 2, 3, 4\}$ qui est l'index de la variable d'entrée, et $i \in \{1, 2\}$ qui est l'index de la règle.

— Couche 2 : cette couche permet le calcul des poids w_i de la règle i, qui sont la multiplication des signaux d'entrées du système (des signaux de la couche précédente). Ils se calculent comme dans (3.17).

$$O_i^2 = w_i = \prod_j \mu_{A_{i,j}}(x_j) \tag{3.17}$$

— Couche 3 : cette couche normalise les poids de chacune des règles par la somme des poids de toutes les règles. Les nœuds de cette couche sont calculés comme dans (3.18).

$$O_i^3 = \overline{w}_i = \frac{w_i}{\sum_k w_k} \tag{3.18}$$

— Couche 4 : la sortie de cette couche est évaluée comme le produit entre le poids normalisé et la fonction de règle de sortie qui est une somme pondérée des valeurs des variables d'entrées. Les nœuds de cette couche sont calculés comme dans (3.19).

$$O_i^4 = \overline{w}_i f_i \tag{3.19}$$

— Couche 5 : cette couche permet l'évaluation du score objectif de qualité finale, qui est le score objectif d'évaluation de la qualité de l'image dont les paramètre sont pris en entrée, comme une somme de tous les signaux de la couche précédente ; tel que présenté dans (3.20).

$$O^5 = f = \sum_i \overline{w}_i f_i \tag{3.20}$$

Dans l'architecture générale de $ANFIS$ présentée dans la Figure 3.9, un cercle indique un nœud fixe, tandis qu'un carré indique un nœud adaptatif ajusté en fonction des valeurs des variables d'entrées ou de la variable de sortie. Ici les couches 1 et 4 sont adaptatives et dépendent des paramètres non-linéaires et des caractéristiques des fonctions d'appartenance pour la couche 1 ; et des paramètres linéaires et des combinaisons polynomiales des règles, pour la couche 4.

Dans l'optique de déterminer le modèle flou qui permettra d'évaluer au mieux la qualité des images avec un fort taux de prédiction, une bonne stabilité et une complexité d'implémentation sur les circuits à faibles capacités, plusieurs paramètres de configuration des modèles flous ont été étudiés dans cette section. Les résultats de sorties dépendent de différents paramètres, parmi lesquels : la stratégie de découpage utilisée pour diviser les espaces des variables d'entrées du système ; le nombre maximum d'itération d'entraînement ; le nombre de règles dans le système ; le type des fonctions d'appartenance permettant de décrire chacune des variables réelles du système.

3.3.5.1 Stratégie de partitionnement

Comme présenté dans l'équation (3.15), les règles floues sont composées de deux parties : une partie de condition (partie « SI »), encore appelée antécédent de la règle floue, qui couvre une zone particulière de l'espace d'entrée ; et une partie conséquence (partie « $ALORS$ »), qui définit le comportement (représentée mathématiquement par une fonction de combinaison linéaire) à avoir quand on se retrouve dans une zone particulière définie dans la première partie de la règle floue. Pour avoir le modèle optimal, il importe donc, de prime abord, d'effectuer un découpage ou encore partitionnement en zone optimale de l'espace constitué des mesures d'entrées ; et ensuite bien définir les conséquences lorsqu'on se retrouve dans chacune des zones prédéfinies.

Trois techniques de partitionnement de l'espace d'entrée ont été étudiées : le partitionnement en grilles (Grid partitioning), le partitionnement sous forme d'arbres (Tree

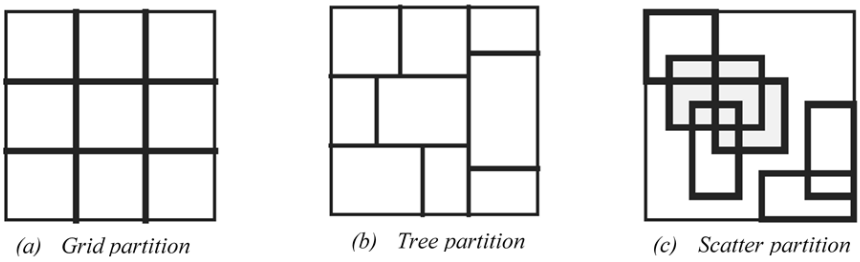


FIGURE 3.10 – Techniques de partitionnements de l'espace : (a) en grilles ; (b) en arbres ; (c) en échantillons

partitioning), et le partitionnement en échantillons (Scatter partitioning) [105, 108]. La Figure 3.10 présente les trois types de partitionnements pour un espace à deux dimensions.

— **Partitionnement en grilles (Grid partitioning)** : ce partitionnement divise l'espace en sous-espaces réguliers, tel que présenté dans la Figure 3.10 (a). L'avantage principal de cette stratégie de partitionnement est qu'elle ne nécessite généralement pas un grand nombre de fonctions d'appartenance pour la description de chacune des variables d'entrées. Par contre, son inconvénient majeur est qu'elle nécessite un très grand nombre de règles pour la mise en œuvre de son système d'inférence flou [109]. Par exemple, pour la construction d'un modèle à quatre entrées et trois fonctions d'appartenances, cette stratégie de partitionnement nécessite $3^4 = 81$ règles floues ; ce qui augmente le nombre de paramètres linéaires, ainsi que la complexité du système global.

— **Partitionnement en arbres (Tree partitioning)** : dans cette stratégie de partitionnement, l'espace est divisé en sous-ensembles non-réguliers sous forme d'arbres, tel que présenté dans la Figure 3.10 (b) pour un espace à deux dimensions. Cette stratégie vient résoudre le problème du nombre de règles très élevé de la stratégie de partitionnement en grilles ; mais elle a comme désavantage de nécessiter un grand nombre de fonctions d'appartenance pour décrire chacune des variables d'entrées ; ce qui augmente le nombre de paramètres non-linéaires, et par la même la complexité du système global.

— **Partitionnement en échantillons (Scatter partitioning)** : cette troisième stratégie de partitionnement permet le découpage de l'espace d'entrée en échantillons pas nécessairement indépendants ou de tailles différentes, tel que présenté dans la Figure 3.10 (c) pour un espace à deux dimensions. Cette stratégie permet de réduire non seulement le nombre de fonctions d'appartenance à utiliser pour chacune des variables d'entrées, mais aussi le nombre de règles du système d'inférence flou [109, 110] ; et donc par là même réduit la complexité du circuit du système global.

La stratégie de partitionnement en échantillons (Scatter partitioning) a été sélectionnée pour la construction de la métrique $GFRIQ - FL$ basée sur la logique floue, pour

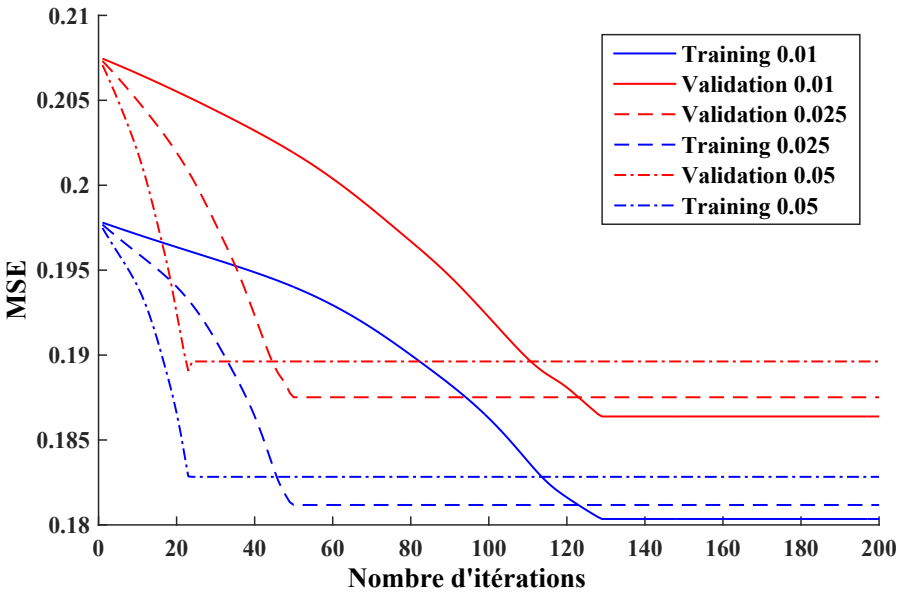


FIGURE 3.11 – Courbe de l'erreur en fonction du nombre d'itérations et du pas

permettre une réduction de la complexité du modèle final. La phase d'apprentissage et le découpage de l'espace des variables d'entrées est fait grâce à la fonction « $genfis$ » de l'outil $MATLAB$. Cette fonction permet une initialisation et un découpage de l'espace d'entrée en fonction de la stratégie de partitionnement choisie. Pour des contraintes liées à l'outil utilisé, et de minimisation de la complexité du modèle final, le nombre de fonctions d'appartenance lors du partitionnement est le même pour toutes les variables d'entrées, et il est aussi égale au nombre de règles floues du système d'inférence.

3.3.5.2 Nombre d'itérations d'entraînement

Une fois la stratégie de partitionnement des données définie, les différentes fonctions d'appartenance ainsi que les règles du système d'inférence flou sont construites grâce à l'outil « ANFIS ». Celui-ci applique un algorithme d'apprentissage hybride sur les données d'entraînement, combinant la méthode des moindres carrées et celle du gradient descendant, dans le but de déterminer les paramètres optimaux pour le modèle final. Cette procédure d'apprentissage hybride doit être exécutée plusieurs fois, pour trouver les paramètres optimaux ; mais un nombre trop grand d'itérations augmente inutilement le temps d'apprentissage. Ce qui est susceptible de poser des problèmes de « sur-apprentissage ». Chaque itération d'apprentissage est composée d'une phase montante et d'une phase descendante. Dans la phase montante, les paramètres linéaires de construction des règles sont modifiés grâce à la méthode des moindres carrées ; tandis que dans la phase descendante, les paramètres non-linéaires de construction des fonctions d'appartenance sont modifiés en utilisant la méthode du gradient descendant [107].

Une étude a été faite entre le nombre d'itérations d'entraînement, la valeur du « pas »

TABLE 3.2 – Complexité des fonctions d'appartenance

MF	# param.	# opérations basiques
GaussMF	2	1 addition ; 3 multiplications ; 1 exponentiel
GbellMF	3	2 additions ; 1 multiplication ; 1 division ; 1 puissance
SMF	2	4 comparaisons ; 2 additions ; 2 multiplications
TriMF	3	4 comparaisons ; 1 addition ; 1 multiplication
TrapMF	4	5 comparaisons ; 1 additions ; 1 multiplications

d'incrémentation entre chaque itération, et l'erreur quadratique moyenne produite par le modèle. Les résultats de cette étude sont présentés dans la Figure 3.11. Ces résultats montrent que plus le « pas » est grand, plus on converge rapidement vers une valeur minimale de l'erreur quadratique moyenne. Ces résultats montrent également que plus le « pas » d'incrémentation entre les itérations est grand, plus on a tendance à converger vers un minimum local supérieur au minimum global qui est la valeur optimale. Par exemple pour un $pas = 0.01$ il faut 130 itérations d'entraînement pour converger vers le minimum, et les paramètres obtenus produisent une erreur quadratique moyenne lors de la phase de validation $MSE = 0.1803$ entre les scores subjectifs et les scores objectifs déterminés par l'index avec ces paramètres. A contrario, on observe que pour un $pas = 0.05$, il faut juste 23 itérations d'entraînement pour converger vers la valeur minimale, et l'erreur quadratique moyenne lors de la phase de validation est $MSE = 0.1829$ entre les scores subjectifs et les scores objectifs.

Pour les simulations suivantes, le nombre d'itérations de la phase d'apprentissage sera fixé à 130 itérations, avec un $pas = 0.01$ d'incrémentation entre chaque itération.

3.3.5.3 Types de fonctions d'appartenance

Le nombre d'itérations lors de la phase d'apprentissage, ainsi que le « pas » d'évolution entre chacune des itérations, et la stratégie de partitionnement ayant été définis, le prochain paramètre à étudier est le type ou encore la forme des fonctions d'appartenance des variables d'entrées. Les différents types de fonctions d'appartenance sont étudiés pour permettre de définir le modèle objectif qui estime au mieux la qualité des images, avec la complexité d'implémentation la plus faible possible.

Différentes fonctions d'appartenance ont été étudiées : la gaussienne ($GaussMF$), la cloche Bell ($GbellMF$), la logarithmique (SMF), la triangulaire ($TriMF$), et la trapézoïdale ($TrapMF$). Les paramètres de description de ces différentes fonctions sont donnés dans le chapitre sur l'état de l'art. Le Tableau 3.2 présente le nombre de paramètres, ainsi que la complexité en termes de nombre d'opérations basiques, nécessaires à l'exécution de chacune des fonctions d'appartenance étudiées.

La Figure 3.12 quant à elle, présente les boîtes à moustaches (box-plots) des distances de Brownian entre le score subjectif pris dans la base de données $TID2013$, et le score ob-

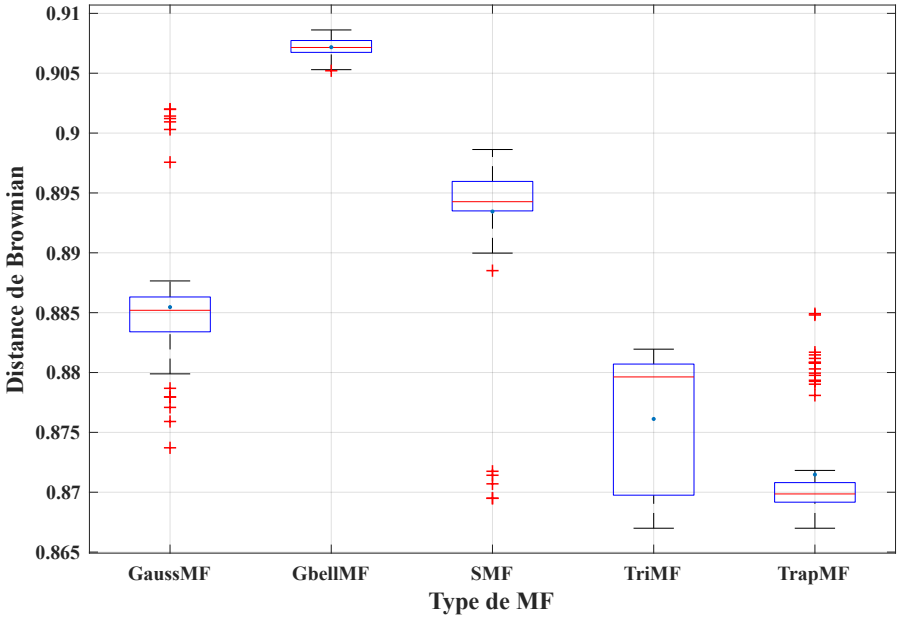


FIGURE 3.12 – Performances pour différents types de MF

jectif de la qualité des images. Ces scores objectifs sont évaluées par différents modèles générés en utilisant la méthode de logique floue pour étudier les différentes fonctions d'appartenance. Cette figure montre non seulement la dispersion des corrélations évaluées, mais aussi la médiane, il s'agit des traits à l'intérieur des boîtes, et la moyenne matérialisée par les points bleus pour chacune des fonctions d'appartenance étudiées.

Une remarque principale mérite d'être soulignée. Il s'agit du constat que les meilleures performances en termes de moyenne et d'écart-type sont obtenues par la fonction d'appartenance en forme de cloche ($GbellMF$). Ces performances sont déterminées par la corrélation entre le score subjectif et le score objectif donné par le modèle construit sur la logique floue.

3.3.5.4 Nombre de règles

Le dernier paramètre à définir est le nombre de règles. Il correspond aussi au nombre de fonctions d'appartenance permettant de décrire chacune des variables d'entrées du système d'inférence flou. La Figure 3.13 illustre l'évolution de l'erreur quadratique moyenne d'entraînement et de validation, en fonction du nombre de règles du système d'inférence flou. Sur cette courbe, on remarque trois pics d'évolution de l'erreur, à savoir : Pour un nombre de règles inférieur à 5, les erreurs d'apprentissage sont supérieures aux erreurs de validation. Pour un nombre de règles supérieur à 5, l'erreur d'apprentissage est généralement inférieure à l'erreur de validation. Et à partir de 12 règles, on remarque un phénomène de sur-apprentissage : l'erreur d'apprentissage continue de diminuer, tandis que l'erreur de validation devient de plus en plus grande lorsqu'on augmente le nombre

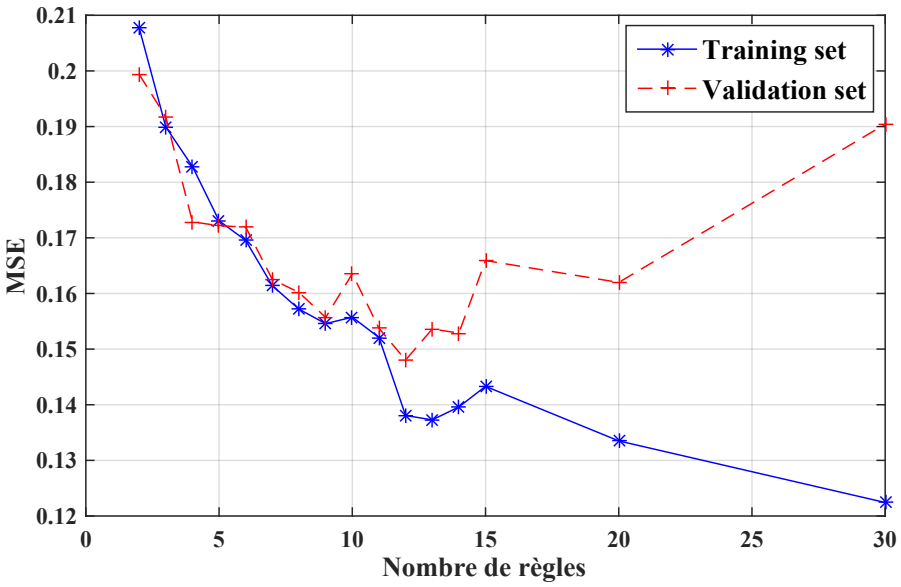


FIGURE 3.13 – Courbes d'erreurs en fonction du nombre de règles

de règles.

Il est important de noter que plus le nombre de règles est élevé, plus le modèle final est complexe. C'est ce que révèlent les équations (3.21) et (3.22) qui donnent respectivement le nombre de paramètres linéaires (NPL) et le nombre de paramètres non-linéaires (NPN) nécessaires à la construction des modèles. Les différents nombre de paramètres sont donnés en fonction du nombre de variables d'entrées (Ne), du nombre de règles du système d'inférence flou (Nr), et du nombre de paramètres utilisés pour la construction de chacune des fonctions d'appartenance (Np). Le nombre de paramètres nécessaire à la construction de chacune des fonctions d'appartenance étudiées est donné dans le Tableau 3.2.

$$NPL = (Ne - 1) \times Nr \tag{3.21}$$

$$NPN = Ne \times Nr \times Np \tag{3.22}$$

En prenant en compte la complexité du modèle final, seules deux règle ont été utilisées pour la construction des modèles. De ce même fait, n'ont été utilisées que deux (02) fonctions d'appartenance pour chacune des variables d'entrées en vue de la construction de l'index $GFRIQ - FL$ d'évaluation de la qualité d'images basée sur la logique floue.

3.3.5.5 Modèle final GFRIQ-FL

Les différents paramètres d'apprentissage du modèle final basé sur la logique floue retenus au terme de l'étude sont les suivantes :

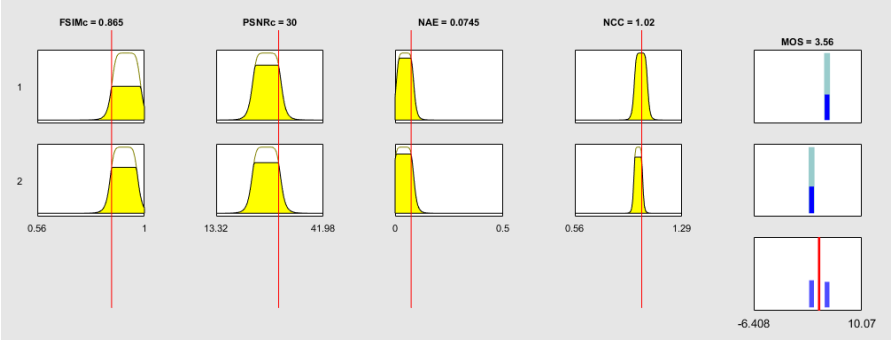


FIGURE 3.14 – Diagramme d'inférence floue du GFRIQ-FL

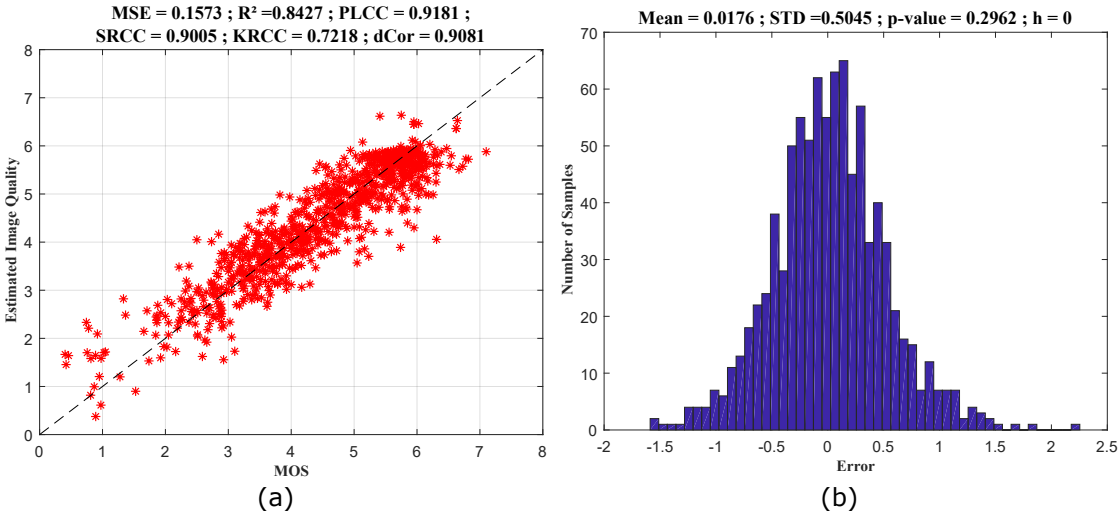


FIGURE 3.15 – Performances du GFRIQ-FL : (a) MOS Vs score estimé ; (b) histogramme des erreurs entre MOS et score estimé

— un apprentissage basé sur la méthode $ANFIS$;
— une stratégie de partitionnement en échantillons de l'espace d'entrée ;
— un apprentissage grâce à un algorithme hybride, combinant la méthode des moindres carrées et celle du gradient descendant. Le nombre d'itérations de la phase d'apprentissage a été fixé à 130 itérations, pour un $pas = 0.01$ d'évolution entre chaque itération ;
— des fonctions d'appartenance en forme de cloche ($GbellMF$) ;
— deux (02) règles pour le système d'inférence flou, et deux (02) fonctions d'appartenance pour chacune des variables d'entrées.

Le modèle produit possède 34 paramètres : 10 paramètres linéaires et 24 paramètres non-linéaires.

La métrique $GFRIQ-FL$ est donc évaluée grâce à un modèle à quatre entrées correspondant aux quatre métriques sélectionnées dans la Section 3.2, et une sortie indiquant

le score de qualité de l'image évaluée. La Figure 3.14 présente le diagramme d'inférence floue de cette métrique objective d'évaluation de la qualité des images, basée sur la logique floue. Les résultats de simulation sur les données de tests de la base de données d'images TID2013, sont présentés dans la Figure 3.15. Le graphe (a) présente une comparaison non seulement graphique, mais aussi numérique au travers des indices de performance, entre le score subjectif (MOS) et la qualité objective des images estimée par l'index $GFRIQ-FL$. Le graphe (b) quant à lui présente l'histogramme des erreurs, et les résultats du test de Khi-deux sur la distribution de ces erreurs entre le score subjectif (MOS) et le score objectif estimé par l'index. Cet index basé sur la logique floue donne une corrélation linéaire de 91.81% et une erreur quadratique moyenne de 15.73% avec le score subjectif (MOS). Ces résultats obtenus montrent que les erreurs entre les scores subjectifs et les scores objectifs de qualité d'images estimées par le modèle $GFRIQ-FL$ suivent une loi normale de moyenne $m = 0.0176$, d'écart-type $\sigma = 0.5045$, avec une $p-value = 0.2962$.

3.4 Résultats expérimentaux

Les sections précédentes avaient pour objectifs de présenter la sélection de la base de données et des métriques à utiliser ; ainsi que la construction des modèles objectifs d'évaluation de la qualité des images à partir des métriques à références complètes et basés sur les méthodes d'apprentissage. Cette section, quant à elle, présente les résultats expérimentaux obtenus sous $MATLAB$ par les différents modèles construits, lors des phases de validation et de test. Il s'agit des ultimes phases du processus de construction des mesures d'évaluation objectives, présenté dans la Figure 3.1.

La méthode de validation croisée de Monte-Carlo $(MCCV)$ est utilisée pour tester la robustesse et la stabilité des différents modèles construits à partir des méthodes d'apprentissage automatique. Cette méthode de validation répète K fois le processus de construction des modèles détaillée dans la section précédente, en sélectionnant aléatoirement les données d'entraînement et de test. Elle assure l'indépendance du modèle final par rapport aux données d'entraînement, ce qui confère une stabilité aux modèles construits. Cette méthode est appliquée avec un nombre de répétitions $K = 1000$ répétitions aléatoires ; où chaque répétition consiste en une phase d'entraînement avec 70% des données, et une phase de test sur le reste des 30% de données non utilisées lors de la phase d'apprentissage.

Les résultats de cette phase de validation sont donnés dans les Tableaux 3.3et 3.4, et dans la Figure 3.16. Le Tableau 3.3 présente les moyennes, et le Tableau 3.4 les écarts-types, des K répétitions des indices de performance obtenus lors des phases de tests des modèles obtenus à partir des différentes méthodes d'apprentissage. Ces résultats prennent en compte aussi bien les scores subjectifs que les scores objectifs évalués par ces modèles. La Figure 3.16 présente les boîtes à moustaches (Box-plots) pour les K répétitions, de la distance de Brownian lors de la phase de test entre les scores objectifs produits par les différents modèles et les scores subjectifs pris dans la base de données d'images.

TABLE 3.3 – Performances (moyenne) des méthodes d'apprentissage

	MSE	**R²**	**PLCC**	**SRCC**	**KRCC**	**dCor**
DA	0.3155	0.6845	0.8376	0.8131	0.7362	0.8138
KNN	0.2772	0.7228	0.8624	0.8496	0.7867	0.8397
ANN	0.1918	0.8082	0.8992	0.8783	0.6978	0.8887
NLR	0.1695	0.8305	0.9114	0.8893	0.7127	0.8986
DT	**0.1396**	**0.8604**	**0.9277**	**0.9151**	**0.7596**	**0.9148**
FL	0.1575	0.8425	0.918	0.8986	0.7213	0.907

TABLE 3.4 – Performances (écart-type) des méthodes d'apprentissage

STD ($\times 10^{-2}$)	**MSE**	**R²**	**PLCC**	**SRCC**	**KRCC**	**dCor**
DA	0.613	0.613	0.279	0.205	0.225	0.228
KNN	1.612	1.612	0.809	1.014	1.056	0.966
ANN	2.615	2.615	2.829	1.888	1.499	2.686
NLR	0.152	0.152	0.083	**0.096**	**0.138**	0.096
DT	0.345	0.345	0.187	0.215	0.312	0.209
FL	**0.126**	**0.129**	**0.07**	0.188	0.262	**0.079**

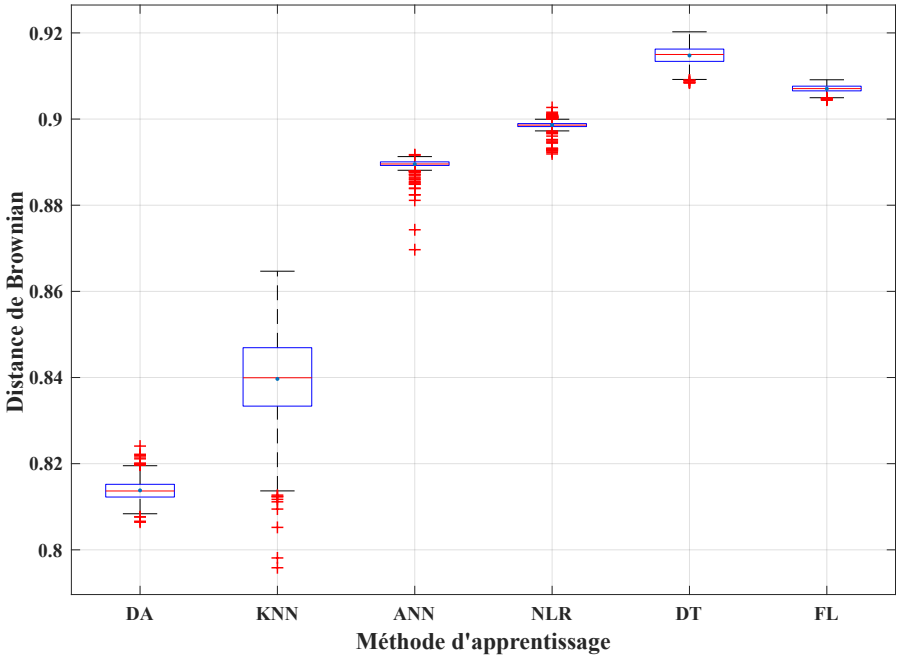


FIGURE 3.16 – Comparaison des performances des méthodes d'apprentissage

Les résultats expérimentaux de la validation croisée montrent que les méthodes d'apprentissage basées sur les arbres de décision et la logique floue donnent les meilleurs résultats aussi bien en termes de précision, explicitée par moyennes des indices de performance, que de stabilité, au vu des écarts-types des indices de performance, entre les scores objectifs obtenus par les différents modèles, et les scores subjectifs pris dans la base de données.

Les modèles produits ont été comparés à d'autres métriques existant dans la littérature, notamment les mesures utilisant aussi une approche neuronales (SFF, MAD). Les résultats de cette comparaison en termes d'indices de performance (coefficients de corrélations de Pearson, Spearman, Kendall et la distance de Brownian) sont donnés dans le Tableau 3.5.

Au regard de ce tableau comparatif, il y a lieu de remarquer que les modèles réalisés donnent des résultats supérieurs à ceux des métriques déjà existantes.

3.5 Architectures et résultats d'implémentation

Cette section présente le processus d'implémentation sur un système $FPGA$ des modèles d'évaluation de la qualité d'image générés par les méthodes d'apprentissage basées sur les arbres de décision et sur la logique floue. Le choix s'est porté sur l'implémentation ces deux méthodes, car elles sont celles qui produisent les meilleurs résultats de simulation sous l'outil $MATLAB$. Les modèles sont implémentés sur une carte $FPGA$

TABLE 3.5 – Comparaison des performances des mesures de qualité d'image

	PLCC	**SRCC**	**KRCC**	**dCor**
FSIMc	0.8322	0.851	0.6669	0.8390
SFF	0.8090	0.8513	0.6581	0.8259
MAD	-0.8074	-0.7807	-0.6035	0.8017
VIF	0.7336	0.677	0.5148	0.6984
GFRIQ-DT	0.915	**0.9031**	**0.7454**	0.8998
GFRIQ-FL	**0.9181**	0.9005	0.7218	**0.9081**

Xilinx de la famille des Virtex 7 (VC707).

L'architecture globale de cette implémentation est présentée dans la Figure 3.17. Les principaux blocs qui composent cette architecture sont : le bloc IP d'implémentation du modèle $GFRIQ-ML$, qui constitue l'index d'évaluation avec référence des images ; un micro-contrôleur (MicroBlaze), la mémoire vive $RAM\ DDR$, des unités d'interconnexion utilisant le protocole AXI ; l'interface de génération de mémoire (MIG) ; et les autres périphériques principalement le composant de liaison entre l'ordinateur et le port série de la carte ($UART$), ainsi que l'horloge.

— **GFRIQ_Core** : il s'agit des blocs IP ($IP-Core$) construits en langage C/C++ grâce à l'outil Vivado HLS, et utilisé pour l'évaluation de la qualité des images. Il communique avec le microcontrôleur à partir des protocoles d'interconnexion AXI. Les détails sur ce bloc IP sont donnés dans la section suivante.
— **Microcontrôleur (MicroBlaze)** : il est l'élément central de l'architecture d'implémentation réalisée. Il exécute les tâches principales suivantes : 1) initialiser et configurer tous les paramètres du système ; 2) coordonner les séquences de déplacements des données à l'intérieur et entre les différentes sous unités du système ; 3) effectuer certaines opérations arithmétiques et logiques.
— **Unité d'interconnexion AXI** : il permet la connexion entre le micro-contrôleur (MicroBlaze) et les autres périphériques ou sous-unités du système.
— **Interface de génération de mémoire (MIG)** : il est utilisé comme un convertisseur entre les protocoles d'interconnexion AXI et ceux de la mémoire vive RAM.
— **UART** : est l'unité qui permet la transmission des résultats contenus dans le micro-contrôleur et la mémoire vers une machine extérieure.
— **Timer** : il est utilisé pour mesurer le temps d'exécution de certains processus du système.

3.5.1 Implémentation HLS des modèles GFRIQ-ML

Deux modèles d'évaluation d'images basées sur les métriques à références complètes ont été implémentés : celui basé sur les arbres de décision ($GFRIQ-DT$) et celui

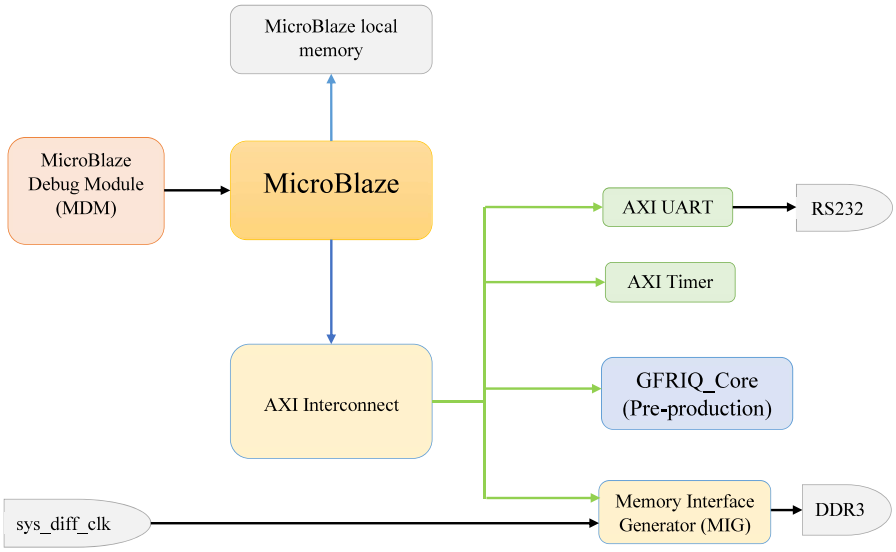


FIGURE 3.17 – Architecture d'implémentation générale du GFRIQ-ML

basé sur la logique floue ($GRFIQ-FL$). Cette implémentation est faite en HLS (en langage C/C++) à partir de l'outil Vivado HLS de Xilinx. Cet outil de Vivado permet la conversion des codes C/C++, vers des méthodes de description des architectures RTL en $VHDL$ / Verilog ; il permet aussi de vérifier si le code peut être synthétisé, ainsi que la production du bloc IP résultant.

L'architecture des blocs IP générés pour les modèles $GFRIQ-ML$ est présentée dans la Figure 3.18 ; elle peut être considérée comme une boîte noire composée des éléments suivants :

— quatre signaux d'entrées, sous forme de nombres flottants, correspondants aux quatre valeurs des métriques d'entrées des modèles ;
— un signal de sortie, sous forme de nombre flottant, correspondant à la valeur de la qualité de l'image produite par le modèle, en fonction des valeurs prises par les entrées ;
— un signal de sortie, sous forme d'un booléen, indiquant si le processus d'évaluation a été effectué avec succès ;
— deux signaux d'entrées, sous forme de booléens, représentant l'horloge et la réinitialisation (le « reset »).

Les entrées et la sortie du bloc IP sont au format flottant à simple précision, rangé sur 32 bits. Le modèle $GFRIQ-DT$ basé sur les arbres de décision s'implémente simplement grâce à des comparaisons consécutives, comme le montre la structure du modèle final présenté dans la Figure 3.7. Le modèle $GFRIQ-FL$ basé sur la logique floue, quant à lui, s'implémente suivant l'architecture en cinq couches du modèle $ANFIS$ présenté dans la Figure 3.9. Dans le but de réduire la complexité de ce modèle final, l'équation de

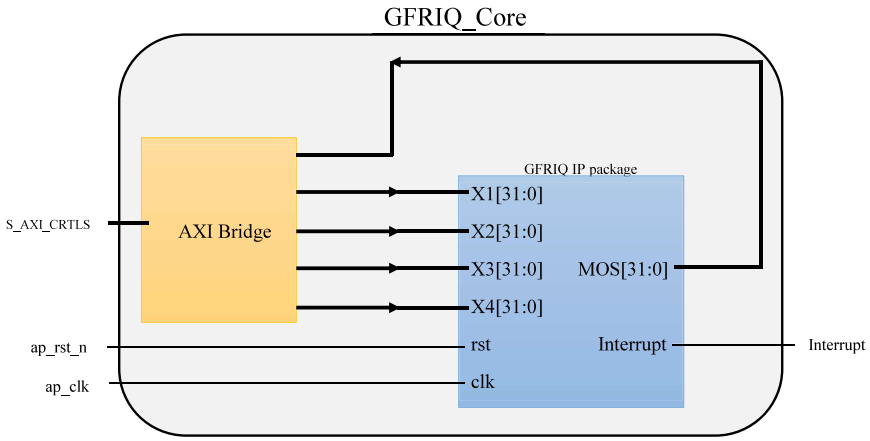


FIGURE 3.18 – Architecture du bloc IP $GFRIQ$

mise en place de la première couche du modèle $ANFIS$, qui est fonction d'appartenance sous forme de cloche Bell ($GbellMF$), a été simplifiée comme dans (3.23). Cela permet de réduire le nombre d'opérations primaires pour la fonction $GbellMF$ à 2 additions, une division et une puissance.

$$\mu_A(x; d, e, c) = \frac{d}{d + (x - c)^e} \; avec \; d = a^{2b}; \; e = 2b \tag{3.23}$$

Le bloc de puissance étant le plus complexe en implémentation dans cette équation, on considère les paramètres b et e comme des entiers pour permettre une simplification de ce bloc. Le Tableau 3.6 présente les performances du modèle $GFRIQ - FL$ en fonction du paramètre b de la fonction d'appartenance. Ce tableau montre que les performances des modèles basés sur la logique floue avec des fonctions d'appartenance en forme de cloche, ne varient pas énormément lorsqu'on considère le paramètre b comme un entier. Lors de l'implémentation la valeur prise a été $2b = 6$ pour simplifier la complexité des calculs.

Le Tableau 3.7 présente les rapports de synthèse générés par les modèles $GFRIQ-DT$ et $GFRIQ - FL$ lors de la phase d'implémentation en HLS sous l'outil Vivado HLS version 2016.4 et mappé sur une carte Xilinx de la famille des Virtex 7 ($VC707$). Ces rapports sont présentés suivants deux axes :

— **L'estimation des performances** : qui présente une estimation du nombre de cycles d'horloge nécessaire à l'exécution complète du bloc IP généré pour différentes périodes d'horloge.

— **L'estimation des utilisations** : qui présente l'utilisation des ressources de la carte $FPGA$ pour différentes périodes d'horloge, en termes de blocs de mémoires (en $Kbits$), registres, portes logiques, DSP, bascules Flip-Flop et LUT.

Ces rapports montrent que les modèles basés sur la méthode des arbres de décision consomment moins de ressources sur la carte, et prennent moins de temps d'exécution

TABLE 3.6 – Performance du GFRIQ-FL en fonction du paramètre b de GbellMF

	b réel	**2b=8**	**2b=6**	**2b=4**
MSE	**0.1573**	0.1703	0.1593	0.2051
R²	**0.8427**	0.8297	0.8407	0.7949
PLCC	0.9181	0.9151	**0.9184**	0.9124
SRCC	**0.9005**	0.8994	0.9001	0.8936
KRCC	**0.7218**	0.7197	0.7213	0.7146
dCor	0.9081	0.904	**0.909**	0.906

TABLE 3.7 – Estimation des performances et utilisations des modèles GFRIQ-ML dans la phase de synthèse HLS.

	Dispo.			**DT**				**FL**	
clock (ns)		2.5	5	10	20	2.5	5	10	20
clock estim. (ns)		2.39	3.78	8.51	12.50	2.81	4.90	9.11	17.41
# cycles clock		73	24	9	5	344	204	115	48
BRAM	2 060	0	0	0	0	0	0	0	0
DSP48E	2 800	0	0	0	0	200	200	200	200
FF	607 200	862	803	1 516	1280	29 336	19 630	12 515	8 878
LUT	303 600	1 808	1 690	2 268	2 094	26 246	23 455	22 065	22 110

TABLE 3.8 – Détail d'utilisation en phase de placement et routage (VC707)

	Dispo.	**DT**		**FL**	
		Utilisation	% Utilisation	Utilisation	% Utilisation
BRAM	1 030	14	1.36 %	14	1.36 %
DSP	2 800	0	0 %	200	7.14 %
FF	607 200	17 500	2.88 %	34 022	5.60 %
LUT	303 600	19 175	6.32 %	37 444	12.33 %
LUTRAM	130 800	2 605	1.99 %	3 291	2.52 %
I/O	700	119	17 %	119	17 %

pour afficher le résultat final, que les modèles basés sur la logique floue. Une autre remarque à souligner est que le nombre de DSP et de mémoires utilisées ne dépend pas de la période de l'horloge. Au vu de l'estimation des performances et des utilisations, ainsi que des capacités de la carte utilisée, pour la suite de ce travail, la fréquence du signal d'horloge a été fixée à 200 Mhz, soit une période de 5 nanosecondes (5 ns).

3.5.2 Implémentation RTL du design global

Les blocs IP ayant été construits grâce à l'outil Vivado HLS, ils sont ensuite intégrés à l'architecture globale d'implémentation, comme présentée dans la Figure 3.19. Le design global ainsi construit sur l'outil Vivado de Xilinx, a été synthétisé et l'étape de placement et routage sur la carte $FPGA$ de la famille des Virtex 7 ($VC707$) a produit la Figure 3.20. Cette figure présente la surface occupée par les différents modèles $GFRIQ - DT$ et $GFRIQ - FL$ sur la carte $FPGA$, sur les graphiques (a) et (b), respectivement. La partie en jaune représente la surface occupée par le bloc IP du modèle d'évaluation de la qualité d'images, et la zone en bleu, est la surface occupée par le reste des composants du système global, tels que : les registres, le microcontrôleur, les interconnexions, ...

Le Tableau 3.8 quant à lui, présente les détails de l'utilisation des ressources, ainsi que le pourcentage d'utilisation de ces ressources sur une carte $FPGA$ de la famille des Virtex 7 (VC707) ; pour le design global des modèles $GFRIQ - DT$ et $GFRIQ - FL$.

La remarque principale issue de la Figure 3.20 et du Tableau 3.8 sur l'utilisation des ressources sur la carte $FPGA$, est que la méthode basée sur les arbres de décision ($GFRIQ - DT$) consomme moins de ressources sur la carte $FPGA$ (partie jaune du graphe (a)), que la méthode basée sur la logique floue (partie jaune du graphe (b)) ; ce qui confirme les résultats obtenus lors de la phase de synthèse HLS.

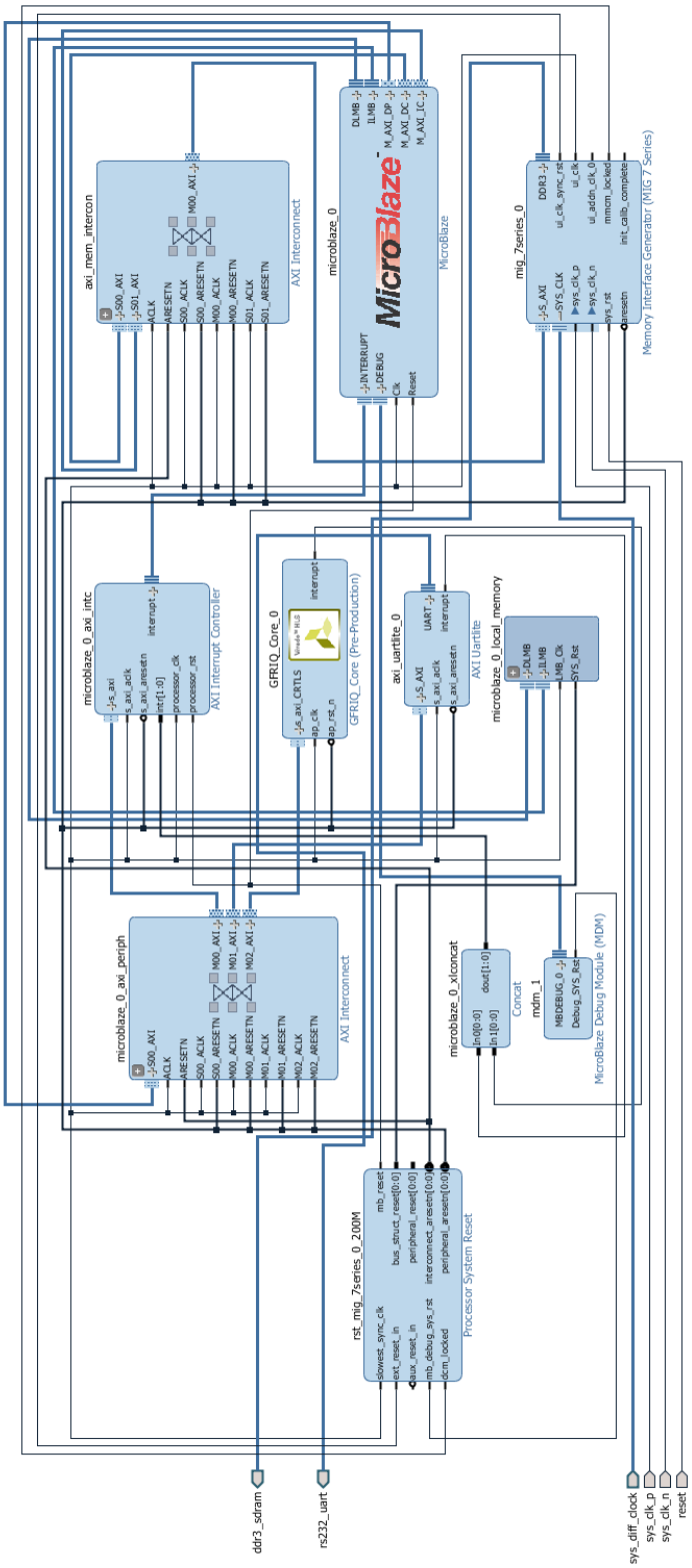
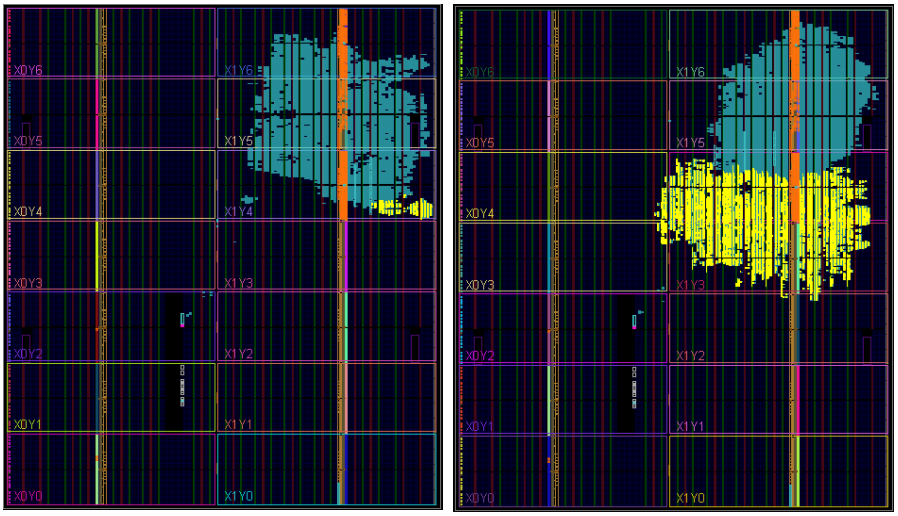


FIGURE 3.19 – Design global du GFRIQ-ML



(a) GFRIQ-DT (b) GFRIQ-FL

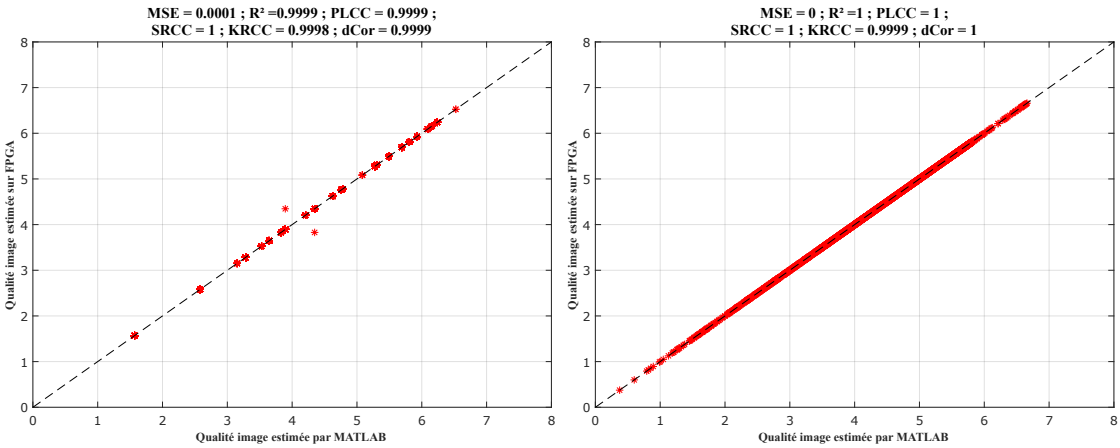
FIGURE 3.20 – Utilisation des ressources en phase de placement et routage (VC707)

3.5.3 Implémentation SDK et résultats

Le design global contenant les modèles d'évaluation de la qualité des images ayant été implémenté sur une carte $FPGA$, l'étape suivante consiste en l'écriture d'un programme dans le microcontrôleur (MicroBlaze), qui permettra de lire les valeurs des métriques qui seront rangées dans la mémoire $RAM\ DDR$, et d'envoyer ces valeurs au bloc IP ($IP-Core$) du modèle d'évaluation de la qualité des images, pour le calcul de la qualité d'image correspondant aux valeurs d'entrées. Ce processus a été réalisé grâce à l'outil SDK de Xilinx. Le programme est écrit en langage C/C++, et il est transmis au micro-contrôleur via l'interface $JTAG$ de la carte $FPGA$.

Le processus d'évaluation d'une image sur une carte $FPGA$ à travers les outils SDK de Xilinx est le suivant : les valeurs des métriques sont transmises dans la mémoire RAM (DDR3), via l'interface $JTAG$ et grâce à l'unité $UART$ de communication avec l'ordinateur présente dans le design global. Les valeurs stockées dans la mémoire sont prises par le micro-contrôleur, et transmises au bloc IP du modèle d'évaluation de la qualité des images ($GFRIQ-ML$), via les périphériques de communication, et grâce au protocole de communication AXI. Le bloc IP du modèle d'évaluation récupère en entrées les valeurs des métriques, et évalue la qualité de l'image à partir de ces valeurs, comme il a été programmé dans le code HLS ; et il transmet le score de qualité obtenu par l'image en sortie dans la mémoire RAM DDR3, via les périphériques de communication grâce au protocole de communication AXI. Ce score stocké dans la mémoire est transmis à l'ordinateur via l'interface $JTAG$ et grâce à l'unité $UART$. Les autres opérations de traitement et d'affichage des résultats sont effectuées au niveau de l'outil SDK.

Les 3000 images déformées de la base de données d'images $TID2013$ ont été évaluées par les modèles réalisés, sur un système $FPGA$; et les résultats de ces évaluations sous



(a) GFRIQ-DT (b) GFRIQ-FL

FIGURE 3.21 – Résultats simulation MATLAB Vs Implémentation FPGA

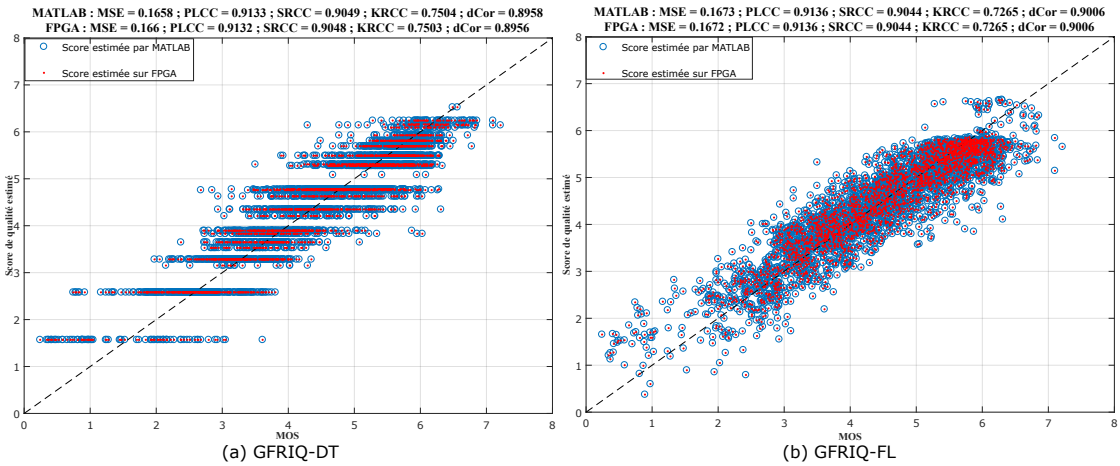
$FPGA$ ont été comparées aux résultats de simulation sous $MATLAB$ et aussi aux scores objectifs pris dans la base de données d'images $TID2013$.

La Figure 3.21 présente la comparaison entre les résultats d'implémentation sous $FPGA$ et les résultats de simulation sous $MATLAB$ pour les deux modèles $GFRIQ-DT$ et $GFRIQ-FL$ dans les graphiques (a) et (b), respectivement. La Figure 3.22 quant à elle, présente les comparaisons entre les résultats de simulation $MATLAB$ (en bleu) et d'implémentation $FPGA$ (en rouge) sur l'axe des ordonnées, et les scores subjectifs issus de la base de données d'images $TID2013$, sur l'axe des abscisses ; pour les deux modèles implémentés $GFRIQ-DT$ et $GFRIQ-FL$ dans les graphiques (a) et (b), respectivement.

3.6 Bilan et apports

Dans ce chapitre a été étudié un processus d'évaluation de la qualité des images avec références complètes, basé sur des méthodes d'apprentissage automatique. Les images ainsi que les scores subjectifs de la base de données d'images TID2013 ont été utilisés pour apprendre et évaluer les différents modèles. Le processus consiste en la construction des modèles à partir d'un échantillon d'apprentissage, suivie d'une évaluation des modèles construits à partir de l'échantillon de test, et enfin une validation des algorithmes utilisés et des modèles construits est faite à partir d'une répétition des phases d'apprentissage et de test 1000 fois en sélectionnant aléatoirement l'échantillon d'entraînement et l'échantillon d'évaluation (de test). L'évaluation des performances des modèles est faite grâce à six (06) indices de performance : l'erreur quadratique moyenne (MSE), le coefficient de détermination (R^2), les coefficients de corrélation de Pearson ($PLCC$), Spearman ($SRCC$) et Kendall ($KRCC$), et enfin la distance de Brownian ($dCor$).

34 métriques sont extraites de la comparaison entre l'image déformée à évaluer et son image de référence. Pour réduire le nombre de métriques à utiliser pour l'évaluation de



(a) GFRIQ-DT (b) GFRIQ-FL

FIGURE 3.22 – MOS Vs résultats simulation MATLAB et implémentation FPGA

la qualité des images, une phase de sélection de métriques indépendantes entre elles a été entreprise. Cette sélection de métrique se fait suivant deux axes : la première basée sur la corrélation entre les différentes métriques consiste à regrouper en classes dépendantes les métriques fortement corrélées entre elles, et de ne sélectionner qu'une métrique dans chaque classe. La seconde méthode de sélection de métriques basée sur l'apprentissage consiste en un processus successif et itératif de sélection progressive des métriques les plus corrélées avec le résidu entre le score subjectif et le score objectif donné par le modèle construit à partir des métriques déjà sélectionnées. A l'issue de cette phase, quatre (04) métriques ont été sélectionnées.

Différentes méthodes d'apprentissage automatique ont été utilisées pour construire les modèles d'évaluation de la qualité des images. Les méthodes de classification : l'analyse discriminante et de la méthode des k-plus proches voisins. Et les méthodes de régression : les réseaux de neurones artificiels, la régression non-linéaire polynomiale, les machines à vecteurs supports, les arbres de décision, et la logique floue. Un accent particulier a été mis sur les méthodes d'apprentissage utilisant les arbres de décision et la logique floue, car ces méthodes ont été proposées dans ce contexte d'évaluation de la qualité des images pour la première fois dans ce travail, mais aussi parce que ces méthodes produisent des résultats meilleurs que les autres méthodes d'apprentissage étudiées.

Les modèles construits à partir des méthodes d'apprentissage basées sur les arbres de décision et sur la logique floue ont été implémentés sur une carte $FPGA$ Xilinx de la famille des Virtex 7 ($VC707$). Cette implémentation est faite en utilisant la gamme d'outils Vivado et Vivado HLS de Xillinx. Les résultats lors de la phase de synthèse du design global montrent que la méthode basée sur les arbres de décision produit un modèle moins complexe en temps et en surface d'utilisation sur la carte, mais la méthode basée sur la logique floue produit un modèle plus précis et stable.

Les apports dans l'évaluation de la qualité des images basée sur des métriques à références complètes se résument en quatre idées principales :

1. La mise en place d'un processus global d'évaluation de la qualité des images combinant un ensemble de métriques existantes pour former une métrique plus performante. Les métriques métriques alors produites donnent des performances supérieures à celles de toutes les métriques à références complètes présentes dans la littérature.
2. Pour la sélection des métriques les plus pertinentes parmi celles qui sont extraites des images, deux processus de sélection de métriques ont été proposés.
3. L'utilisation des méthodes d'apprentissage basée sur les arbres de décision et sur la logique floue pour l'évaluation de la qualité des images.
4. L'implémentation des modèles obtenus sur un système $FPGA$.

Les images de référence n'étant pas toujours disponibles dans certaines applications, un cas plus applicatif est la construction des modèles d'évaluation sans référence. Dans le chapitre suivant, sera présentée l'évaluation de la qualité des images à partir des caractéristiques extraites des images dégradées, sans aucune information sur les images de référence.

Chapitre 4

Évaluation de la qualité des images à partir des métriques sans référence

4.1 Introduction

L'évaluation objective apparaît aujourd'hui, comme une bonne alternative à l'évaluation subjective de la qualité des images. Cette évaluation objective peut se diviser en trois grands groupes en fonction de la disponibilité de l'image originale : l'évaluation à référence complète, à référence réduite et l'évaluation sans référence. Dans la pratique l'image originale est très rarement disponible lors de l'évaluation de la qualité d'une image, ce qui pose un problème pour la mise en œuvre des mesures d'évaluation objective de la qualité des images utilisant une référence. Ce problème lié à la disponibilité de l'image de référence a conduit à la mise en place des techniques d'évaluation de la qualité des images sans référence.

Certaines métriques sans référence se basent sur l'idée suivant laquelle les distorsions déformant une image sont préalablement connues. Il s'agit des métriques d'évaluation de la qualité d'images sans référence à distorsions spécifiques [111, 112]. Ces métriques évaluent la qualité d'une image pour des distorsions spécifiques : elles mesurent le taux de déformation lié à des distorsions bien spécifiques. Par exemple, l'algorithme présenté dans [113] est lié à la détection de la compression « $JPEG$ » ; celui présenté dans [112] permet l'évaluation de l'impact de la compression « $JPEG2000$ » sur les images ; et le modèle présenté par les auteurs de [114] est porté sur la détection du « flou » dans les images.

Cependant, de manière plus pratique, les informations sur les distorsions ayant affectées les images ne sont pas connues. Dans ce cas de figure, il existe des métriques d'évaluation de la qualité des images sans référence à distorsions non-spécifiques [115]. Ces méthodes à distorsions non-spécifiques permettent l'évaluation de qualité d'une image sans aucune information sur l'image de référence et sur le type de distorsions ayant affectées cette image. Il existe dans la littérature plusieurs mesures de la qualité d'images sans référence à distorsions non-spécifiques. Les modèles d'évaluation de certaines de ces mesures sont sensibles aux scores subjectifs (apprentissage supervisé, effectué grâce à des

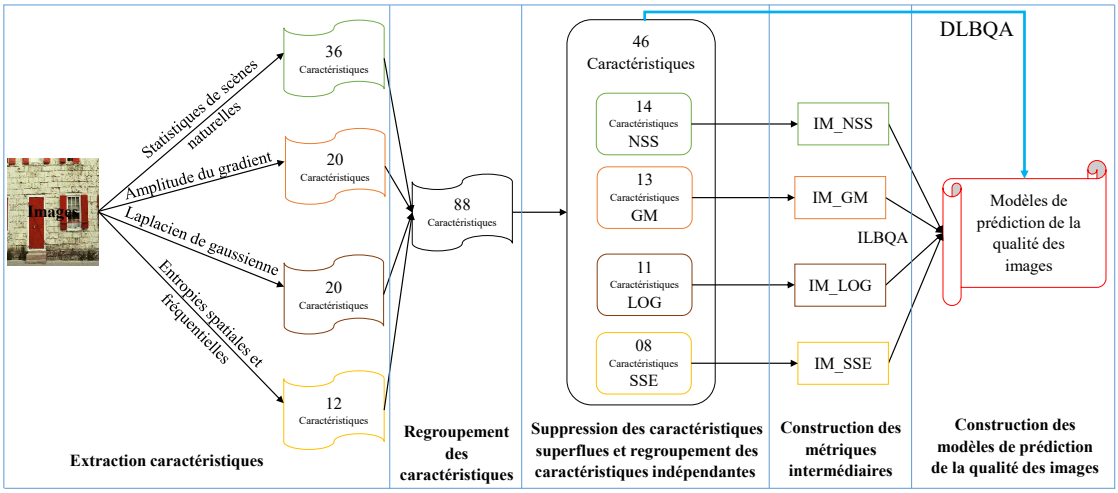


FIGURE 4.1 – Processus de construction du modèle d'évaluation objective sans référence

scores subjectifs), et d'autres non. Parmi les méthodes non sensibles aux scores subjectifs, on retrouve les mesures $NIQE$ [14] (Natural Image Quality Evaluator) et sa version intégrant des caractéristiques locales, $IL-NIQE$ [116] basées sur les scènes naturelles de l'image déformée. Parmi les méthodes sensibles aux scores subjectifs, on retrouve des mesures basées sur les statistiques de scènes naturelles dans l'image, telles que $BRISQUE$ (Blind/Referenceless Image Spatial QUality Evaluator) [15, 16] et $DIIVINE$ (Distortion Identification-based Image Verity and INtegrity Evaluation) [13] ; d'autres basées sur l'amplitude du gradient et les caractéristiques du laplacien, telles que $GMLOGQA$ (Gradient Magnitude and Laplacian of Gaussian Quality Assessment) [59] ; ou encore basées sur les entropies spatiales et les entropies spectrales locales de l'image déformée, $SSEQ$ (Spatial–Spectral Entropy-based Quality)[117].

Dans ce chapitre, sont proposées des méthodes d'évaluation objectives de la qualité visuelle des images, sans référence, à distorsions non-spécifiques, et sensibles aux scores subjectifs. Ces méthodes utilisent des algorithmes d'apprentissage supervisés pour la construction des modèles d'évaluation, et combinent des caractéristiques extraites des images suivant quatre axes : les statistiques de scènes naturelles dans le domaine spatial, l'amplitude du gradient, le laplacien de la gaussienne, et les entropies spatiales et fréquentielles locales de l'image. Le processus de construction des modèles d'évaluation proposé ici est présenté dans la Figure 4.1. Ce processus est constitué des étapes suivantes :

1. Extraction des caractéristiques : dans cette étape ont été extraites les caractéristiques des images issues des bases de données d'images, suivant quatre axes.
2. Suppression des caractéristiques superflues : cela consiste à supprimer certaines des caractéristiques fortement corrélées entre elles, et des caractéristiques présentant des valeurs aberrantes. Cette étape permet de ne conserver que les caractéristiques indépendantes entre elles et apportant de l'information par rapport à la perception visuelle.

3. Construction des métriques intermédiaires : les caractéristiques résultantes sont regroupées dans des classes indépendantes pour construire des métriques intermédiaires. Le passage par cette étape de métriques intermédiaires permet de simplifier les calculs et la complexité des modèles produits.
4. Construction du modèle final : les métriques intermédiaires ainsi construites sont utilisées comme valeurs d'entrées des algorithmes d'apprentissage, pour la construction des modèles d'évaluation de la qualité des images liée avec la perception visuelle.

Ce chapitre s'organise en six sections. La Section 4.2 présente les méthodes d'extraction des caractéristiques permettant l'évaluation de la qualité des images. Dans la Section 4.3 sont expliquées la sélection des caractéristiques indépendantes, et la construction des métriques intermédiaires grâce aux algorithmes d'apprentissage. La Section 4.4 présente les résultats expérimentaux obtenus par différentes méthodes, sur différentes bases de données d'images ; et leurs comparaisons. La Section 4.5 présente les architectures d'implémentation de certains modèles sur un système $FPGA$, ainsi que les résultats obtenus lors de cette phase d'implémentation. Pour terminer, la Section 4.6 permet d'effectuer un récapitulatif du chapitre, en présentant les apports suggérés dans le domaine traité et un bilan du chapitre.

4.2 Extraction des caractéristiques

Les caractéristiques extraites des images suivent quatre axes principaux : les caractéristiques basées sur les statistiques de scènes naturelles dans le domaine spatial (NSS), les caractéristiques basées sur l'amplitude du gradient (GM), les caractéristiques du laplacien (basées sur le laplacien de la gaussienne : LOG), et les caractéristiques basées sur les entropies spatiales et fréquentielles locales de l'image (SSE).

4.2.1 Statistiques de scènes naturelles dans le domaine spatial

L'approche d'extraction des caractéristiques basée sur les statistiques de scènes naturelles est construite sur l'hypothèse suivant laquelle les images naturelles non dégradées occupent un sous-espace spécifique de l'espace de toutes les images possibles. Dans ce cas, on cherche à déterminer la distance entre l'image dégradée et le sous-espace des images naturelles. Cette approche repose sur les changements observés dans les statistiques de l'image lors de l'apparition des dégradations [118].

Les images naturelles étant des stimuli fondamentaux auxquels le système visuel humain est adapté, il est pertinent d'en étudier les propriétés statistiques, en vue de la détermination d'une meilleure qualité en rapport avec la perception visuelle [119, 120]. Lors de l'étude des images naturelles, une remarque particulière est l'invariance à l'échelle des statistiques (lorsque l'on change l'échelle à laquelle on observe l'ensemble des images naturelles, leur distribution statistique demeure inchangée) [121]. Pour prendre en compte cette invariance à l'échelle, les auteurs de [119] ont intégré la fonction « log-contrast »

qui se définit comme le logarithme du niveau de gris des images ramené au niveau de gris moyen, tel que présenté dans (4.1).

$$C(i,j) = ln\left(\frac{I(i,j)}{I_0}\right) \tag{4.1}$$

où $I(i,j)$ est la luminance du pixel (i,j) et I_0 le niveau de gris moyen.

L'extraction des caractéristiques basées sur les statistiques de scènes naturelles (NSS) dans le domaine spatial commence par une normalisation de l'image, représenté ici par la variable $I\,(i,j)$. Cette normalisation permet d'avoir une image avec une moyenne nulle et un écart type unitaire, ce qui permet de s'affranchir des variations de luminosité et de contraste [119]. Cette normalisation est représentée par (4.2).

$$\hat{I}(i,j) = \frac{I(i,j) - \mu(i,j)}{\lambda(i,j) + 1} \tag{4.2}$$

où M et N sont les dimensions de l'image, $i = 1, 2, ..., M$ et $j = 1, 2, ..., N$ sont les indices spatiaux. $\mu(i,j)$ représente la moyenne locale, calculée comme dans l'équation (4.3) ; et $\lambda(i,j)$ représente l'écart type local, calculé comme dans l'équation 4.4).

$$\mu(i,j) = \sum_k \sum_l \omega_{k,l} I(i+k\,,\, j+l) \tag{4.3}$$

$$\lambda(i,j) = \sqrt{\sum_k \sum_l \left[\omega_{k,l} I(i+k\,,\, j+l) - \mu(i,j)\right]^2} \tag{4.4}$$

où $\omega_{k,l}$ est une fonction de pondération gaussienne, circulaire et symétrique à deux dimensions, échantillonnée suivant les axes K et L, et remis à l'échelle unitaire. $k = -K, ..., K$; et $l = -L, ..., L$; avec les échelles d'échantillonnage définies dans [16, 14] comme étant égales à 3 ($K = L = 3$) pour une simplification des calculs.

Un ensemble de caractéristiques liées à la distance entre l'image déformée et le groupe d'images naturelles peuvent être extraites de l'image normalisée. Ces caractéristiques changent en fonction de la présence des distorsions dans l'image ; ainsi la quantification de ces caractéristiques peut permettre de trouver le type de distorsion ayant affectée l'image, mais aussi de définir un certain score de qualité visuelle de l'image. Pour des raisons de simplicité dans le modèle final, les seules caractéristiques étudiées ici sont celles des lois gaussiennes généralisées : symétrique (GGD) et asymétrique ($AGGD$) [122].

La fonction de densité de probabilité de la loi gaussienne généralisée symétrique, pour une image normalisée (moyenne nulle) est donnée par (4.5).

$$f(x, \alpha, \sigma^2) = \frac{\alpha}{2\beta\Gamma\,(1/\alpha)} exp\,(-\,(|x|/\beta)^\alpha) \tag{4.5}$$

où β est donné par l'équation (4.6), et la fonction gamma $\Gamma(.)$ est représentée comme dans (4.7).

$$\beta = \sigma \sqrt{\frac{\Gamma(1/\alpha)}{\Gamma(3/\alpha)}} \tag{4.6}$$

$$\Gamma(a) = \int_0^\infty t^{a-1} e^{-t} dt \qquad pour \quad a > 0 \tag{4.7}$$

Les caractéristiques extraites dans cette densité de probabilité de la loi gaussienne généralisée symétrique sont : le paramètre de position μ qui est toujours à zéro, vu qu'on travaille sur l'image normalisée ; le paramètre d'échelle σ^2 qui contrôle la variance de la fonction de densité de probabilité, et le paramètre de forme α qui permet de définir la forme de la courbe.

La fonction de densité de probabilité de la loi gaussienne généralisée asymétrique, pour une image normalisée est, quant à elle, donnée par le système d'équations (4.8).

$$f(x, \gamma, \sigma_l^2, \sigma_r^2) = \begin{cases} \frac{\gamma}{(\beta_l + \beta_r)\Gamma(1/\gamma)} exp\left(-\left(\frac{-x}{\beta_l}\right)^\gamma\right) & \forall\, x \leq 0 \\ \frac{\gamma}{(\beta_l + \beta_r)\Gamma(1/\gamma)} exp\left(-\left(\frac{x}{\beta_r}\right)^\gamma\right) & \forall\, x \geq 0 \end{cases} \tag{4.8}$$

où le paramètre β_a (avec $a = l$ ou r) est donnée par l'équation (4.9).

$$\beta_a = \alpha_a \sqrt{\frac{\Gamma(1/\gamma)}{\Gamma\,(3/\gamma)}} \tag{4.9}$$

Les caractéristiques extraites dans cette densité de probabilité de la loi gaussienne généralisée asymétrique sont : le paramètre de forme γ ; les paramètres de contrôle à gauche et à droite de la courbe, σ_l^2 et σ_r^2, respectivement ; et un paramètre moyen entre les 2 axes (gauche et droite) de la fonction de densité η, défini par (4.10).

$$\eta = (\beta_r - \beta_l)\, \frac{\Gamma\,(2/\gamma)}{\Gamma(1/\gamma)} \tag{4.10}$$

Dans la statistique liée à la fonction de densité de probabilité de la loi gaussienne généralisée asymétrique, il existe une relation entre un pixel et son voisinage, suivant quatre orientations : horizontale (H), verticale (V), et les deux axes diagonaux ($D1$ et $D2$). Les équations de liaison de ces orientations sont données dans (4.11).

$$\begin{aligned} H &= \hat{I}\,(i,j)\, \hat{I}\,(i,j+1) \\ V &= \hat{I}\,(i,j)\, \hat{I}\,(i+1,j) \\ D1 &= \hat{I}\,(i,j)\, \hat{I}\,(i+1,j+1) \\ D2 &= \hat{I}\,(i,j)\, \hat{I}\,(i+1,j-1) \end{aligned} \tag{4.11}$$

Au final, les caractéristiques liées aux statistiques de scènes naturelles dans le domaine spatial extraites de l'image sont : les deux paramètres non nuls de la fonction de densité de probabilité de la loi gaussienne symétrique (α et σ^2), et les quatre paramètres de la fonction de densité de probabilité de la loi gaussienne asymétrique (η, γ, σ_l^2, σ_r^2), où les

TABLE 4.1 – Caractéristiques extraites suivant les NSS dans le domaine spatial

ID Caract.	**Description des caractéristiques**	**Procédure d'exécution**
f1 - f2	forme(α) et variance (σ^2)	Paramètres du GGD
f3 - f6	moyenne(η), forme (γ), variance gauche(σ_l^2) et variance droite (σ_r^2)	Paramètres de l'AGGD suivant l'orientation horizontale (H)
f7 - f10	moyenne(η), forme (γ), variance gauche(σ_l^2) et variance droite (σ_r^2)	Paramètres de l'AGGD suivant l'orientation verticale (V)
f11 - f14	moyenne(η), forme (γ), variance gauche(σ_l^2) et variance droite (σ_r^2)	Paramètres de l'AGGD suivant la première diagonale (D1)
f15 - f18	moyenne(η), forme (γ), variance gauche(σ_l^2) et variance droite (σ_r^2)	Paramètres de l'AGGD suivant la deuxième diagonale (D1)

paramètres asymétriques sont effectués suivant quatre orientations, comme présentées dans le Tableau 4.1 . Tous ces paramètres symétriques et asymétriques suivants les quatre orientations sont exécutés pour l'image sur deux échelles : l'image en dimension normale tel que reçue, et l'image redimensionnée de moitié ; ce qui donne au final 36 caractéristiques : 2 échelles × [2 paramètres symétriques + 4 paramètres asymétriques × 4 orientations].

4.2.2 Amplitude du gradient et laplacien de gaussienne

La seconde méthode d'extraction des caractéristiques de l'image est basée sur la jointure entre les caractéristiques de l'amplitude du gradient (GM) et sur les caractéristiques du laplacien (LOG : laplacien de gaussienne). Ces deux éléments sont combinés lors de l'extraction, car ils ont des caractéristiques communes qui, combinées, permettent de définir une meilleure qualité d'images. Habituellement, l'amplitude du gradient (GM) et le laplacien de gaussienne (LOG) sont utilisés pour la détection des objets et la détection des contours dans une image (détection des sémantiques structurelles), mais ces éléments peuvent aussi être des caractéristiques d'évaluation de qualité des images.

En supposant une image représentée par $I(i,j)$, son amplitude du gradient est donné par (4.12).

$$G_I = \sqrt{[I \otimes h_x]^2 + [I \otimes h_y]^2} \tag{4.12}$$

où le signe « $\otimes$ » représente l'opérateur de convolution linéaire. h_d est un filtre gaussien appliqué aux dérivées partielles suivant la direction $d \in \{x, y\}$, donnée par (4.13) ; avec la fonction gaussienne isotope de paramètre d'échelle σ, qui est donnée par (4.14).

$$\begin{aligned} h_d(x,y|\sigma) =& \frac{\partial}{\partial d} g\left(x,y|\sigma\right) \\ =& -\frac{1}{2\pi\sigma^2}\frac{d}{\sigma^2} exp\left(-\frac{x^2+y^2}{2\sigma}\right) \end{aligned} \tag{4.13}$$

$$g(x,y|\sigma) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \tag{4.14}$$

Le laplacien de gaussienne de l'image I est donné par (4.15).

$$L_I = I \otimes h_{LOG} \tag{4.15}$$

où le filtre gaussien h_{LOG}est donné par (4.16).

$$\begin{aligned} h_{LOG}(x,y|\sigma) =& \frac{\partial}{\partial x^2} g\left(x,y|\sigma\right) + \frac{\partial}{\partial y^2} g\left(x,y|\sigma\right) \\ =& \frac{1}{2\pi\sigma^2}\frac{x^2+y^2-2\sigma^2}{\sigma^4} exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \end{aligned} \tag{4.16}$$

La distribution marginale des caractéristiques basées sur l'amplitude du gradient dans les images naturelles peut être modélisée suivant la distribution de Weibull, présentée en détails dans [123] ; tandis que la distribution marginale des caractéristiques basées sur le laplacien de gaussienne est décrite suivant une distribution gaussienne généralisée, présentée en détails dans [124]. Ces distributions marginales n'étant pas des statistiques suffisamment stables pour la production des caractéristiques d'évaluation de la qualité des images, les coefficients GM et LOG ont été normalisés pour obtenir des statistiques plus stables, permettant la représentation de l'image. La normalisation commune du GM et du LOG sur chaque pixel de l'image est donnée par le coefficient $F_I(i,j)$, représenté dans l'équation (4.17).

$$F_I(i,j) = \sqrt{G_I^2(i,j) + L_I^2(i,j)} \tag{4.17}$$

Les caractéristiques GM et LOG sont alors normalisées comme dans (4.18).

$$\begin{aligned} \overline{G}_I &= G_I / \left(N_I + \varepsilon\right) \\ \overline{L}_I &= L_I / \left(N_I + \varepsilon\right) \end{aligned} \tag{4.18}$$

où ε est une constante positive à faible valeur, utilisée pour supprimer les instabilités lorsque N_I est très petit. N_I représente le facteur de normalisation adaptatif, qui est exécuté pour chacun des pixels (i,j) de l'image, tel que présenté dans (4.19).

$$N_I(i,j) = \sqrt{\sum \sum_{(l,k)\in\Omega_{i,j}} \omega(l,k) F_I^2(l,k)} \tag{4.19}$$

où $\Omega_{i,j}$ est la fenêtre locale centrée en (i,j) ; et les $\omega(l,k)$ sont des poids définis comme la troncature des coefficients d'un noyau gaussien et normalisé de manière à ce que leur somme soit égale à l'unité : $\sum_{l,k} \omega(l,k) = 1$.

Les caractéristiques du *GM* et du *LOG* décrivent les structures locales de l'image suivant plusieurs axes. Les auteurs de [59] montrent que les caractéristiques issues de l'interaction entre le *GM* et le *LOG* peuvent être utilisées pour évaluer la qualité visuelle d'une image. Ainsi, les caractéristiques normalisées $\overline{G}_I$ et $\overline{L}_I$ ont été quantifiées en M niveaux $\{g_1, g_2, ..., g_M\}$ et N niveaux $\{l_1, l_2, ..., l_N\}$, respectivement. La fonction de probabilité empirique jointe de $\overline{G}_I$ et $\overline{L}_I$ est donnée par le l'histogramme $K_{m,n}$, tel que présenté dans (4.20).

$$K_{m,n} = P\left(\overline{G}_I = g_m, \overline{L}_I = l_n\right) \tag{4.20}$$

où $n = 1, ..., N$, $m = 1, ..., M$.

$K_{m,n}$ étant l'histogramme des caractéristiques normalisées jointes $\overline{G}_I$ et $\overline{L}_I$, il contient beaucoup d'informations permettant de caractériser $\overline{G}_I$ et $\overline{L}_I$. Cependant cet histogramme est de très grande taille (dimension $= M \times N$), et les caractéristiques sont parfois redondantes. Au lieu d'utiliser $K_{m,n}$, il est préférable d'utiliser les densités de probabilités marginales de $\overline{G}_I$ et $\overline{L}_I$, représenté ici par P_G et P_L, respectivement ; et calculées comme dans (4.21).

$$\begin{aligned} P_G\left(\overline{G}_I = g_m\right) &= \sum_{n=1}^{N} K_{m,n} \\ P_L\left(\overline{L}_I = l_n\right) &= \sum_{m=1}^{M} K_{m,n} \end{aligned} \tag{4.21}$$

Les densités de probabilités marginales P_G et P_L ne permettent pas de définir les dépendances entre le GM et le LOG ; d'où la définition d'un index $D_{m,n}$ permettant de mesurer la dépendance entre le GM et le LOG, calculée comme dans (4.22).

$$D_{m,n} = \frac{K_{m,n}}{P\left(\overline{G}_I = g_m\right) \times P\left(\overline{L}_I = l_n\right)} \tag{4.22}$$

Si les caractéristiques GM et LOG de l'image sont indépendantes, alors $D_{m,n} = 1$; mais s'il existe des dépendances entre elles, alors $D_{m,n}$ prendra une valeur différente de 1. Ainsi, on définit la dépendance des valeurs de $\overline{G}_I$ sachant celles de $\overline{L}_I$, et inversement. Les densités de probabilités marginales des dépendances entre $\overline{G}_I$ et $\overline{L}_I$ sont définies par Q_G et Q_L, comme dans (4.23).

TABLE 4.2 – Caractéristiques basées sur le GM et le LOG

ID Caract.	**Description des caractéristiques**
f1 - f10	Densité de probabilité marginale du GM
f11 - f20	Densité de probabilité marginale du LOG
f21 - f30	Densité de probabilité marginale des dépendances suivant le GM
f31 - f40	Densité de probabilité marginale des dépendances suivant le LOG

$$
\begin{aligned}
Q_G\left(\overline{G}_I = g_m\right) =& P\left(\overline{G}_I = g_m\right) \cdot \frac{1}{N} \sum_{n=1}^{N} D_{m,n} \\
=& \frac{1}{N} \sum_{n=1}^{N} \frac{P\left(\overline{G}_I = g_m, \overline{L}_I = l_n\right)}{P\left(\overline{L}_I = l_n\right)} \\
Q_L\left(\overline{L}_I = l_n\right) =& P\left(\overline{L}_I = l_n\right) \cdot \frac{1}{M} \sum_{m=1}^{M} D_{m,n} \\
=& \frac{1}{M} \sum_{m=1}^{M} \frac{P\left(\overline{G}_I = g_m, \overline{L}_I = l_n\right)}{P\left(\overline{G}_I = g_m\right)}
\end{aligned}
\tag{4.23}
$$

Les caractéristiques extraites des images suivant les axes de l'amplitude du gradient et le laplacien de gaussienne sont celles de P_G, P_L, Q_G et Q_L. Dans [59], les auteurs démontent qu'on a des résultats optimaux pour des valeurs $M = N = 10$. Ainsi, sur ces deux axes GM et LOG, on extrait au total 40 caractéristiques, telles que présentées dans le Tableau 4.2.

4.2.3 Entropies spatiales et fréquentielles

L'entropie d'une image révèle la quantité d'information contenue dans celle-ci [125]. L'entropie globale d'une image capture les informations globales de l'image, et ne prend pas en compte la distribution de ces informations dans l'image, contrairement aux entropies locales. Ainsi, des images ayant des entropies égales peuvent s'avérer être très différentes. Cependant, dans [10], les auteurs démontrent qu'il existe un lien entre les entropies locales d'une image et sa qualité visuelle. Dans [117], les auteurs montrent en plus que l'entropie locale dans une image est sensible au type et au degré de distorsion dans l'image qui affecte cette zone, sur le plan spatial comme spectral. Ce qui permet de définir l'hypothèse selon laquelle l'entropie locale d'une image non déformée possède certaines caractéristiques statistiques liées aux pixels adjacents, qui changent avec l'introduction des distorsions dans l'image, et font donc changer l'entropie locale. Par exemple, le bruit augmente l'entropie locale dans l'image, tandis que le flou a plutôt tendance à la réduire.

TABLE 4.3 – Caractéristiques basées sur les entropies locales

ID Caract.	**Description des caractéristiques**
f1 - f2	Moyenne et coefficient d'asymétrie de l'entropie spatiale pour l'image haute
f3 - f4	Moyenne et coefficient d'asymétrie de l'entropie fréquentielle pour l'image haute
f5 - f6	Moyenne et coefficient d'asymétrie de l'entropie spatiale pour l'image moyenne
f7 - f8	Moyenne et coefficient d'asymétrie de l'entropie fréquentielle pour l'image moyenne
f9 - f10	Moyenne et coefficient d'asymétrie de l'entropie spatiale pour l'image faible
f11 - f12	Moyenne et coefficient d'asymétrie de l'entropie fréquentielle pour l'image faible

L'entropie spatiale est une fonction de la distribution de probabilité liée aux valeurs locales des pixels dans un bloc de l'image, tandis que l'entropie fréquentielle est une fonction de la distribution de probabilité liée aux valeurs locales des coefficients de la transformée en cosinus discrète (DCT) dans ce bloc. Le processus d'extraction des caractéristiques basées sur les entropies spatiales et fréquentielles des blocs de l'image consiste en trois étapes :

1. La première étape du processus consiste en un sous-échantillonnage de l'image, grâce à une interpolation bicubique de coefficient 2 (redimensionner l'image). Ici, trois échelles d'images sont produites : haute, moyenne et faible, représentant l'image originale, interpolée avec des dimensions divisées par 2, et interpolée avec des dimensions divisées par 4, respectivement.
2. La deuxième étape du processus est le partitionnement des images issues de la première étape, en blocs de taille $M \times M$, et à l'évaluation des entropies spatiales et fréquentielles locales de chacun des blocs ainsi produits. L'entropie spatiale locale est calculée comme dans (4.24), et l'entropie fréquentielle locale est calculée comme dans (4.25).

$$Es = -\sum_{x} p(x) log_2 p(x) \tag{4.24}$$

où $p(x)$ est la densité de probabilité empirique des pixels x dans un bloc.

$$Ef = -\sum_{i}\sum_{j} P(i,j)\, log_2 P(i,j) \tag{4.25}$$

où $P(x)$ est la probabilité fréquentielle des coefficients normalisés de la transformée en cosinus discrète, dans le bloc. Elle se définit comme dans (4.26).

$$P(i,j) = \frac{C\,(i,j)^2}{\sum_i \sum_j C\,(i,j)^2} \tag{4.26}$$

où $C(i,j)$ est le coefficient normalisé de la transformée en cosinus discrète pour le pixel (i,j) ; $i = 1, ..., M$, $j = 1, ..., M$; $(i,j) \neq (1,1)$ (on exclut le cas où i et j sont tous les deux à 1) ; et $M = 8$.

3. La troisième étape est l'évaluation des caractéristiques à partir des valeurs des entropies spatiales et fréquentielles locales, déterminées à la deuxième étape. Ces valeurs des entropies locales sont triées et rangées dans des vecteurs : $S = (se_1,\ se_2,\ ...,\ se_n)$ pour les entropies spatiales, et $F = (fe_1,\ fe_2,\ ...,\ fe_n)$ pour les entropies fréquentielles ; avec n étant le nombre de blocs dans l'échelle d'image étudiée. Cette évaluation des caractéristiques est faite par la méthode de mise en commun décrite dans [126], qui consiste à évaluer la moyenne des vecteurs S et F et le coefficient d'asymétrie (Skew en anglais) de ces vecteurs S et F, ce qui permet de trouver les caractéristiques mises en exergue. Pour chacune des échelles de l'image, on a les caractéristiques de moyenne et de coefficient asymétrique des vecteurs S et F.

Au final, 12 caractéristiques basées sur les entropies spatiales et fréquentielles locales sont extraites de l'image, telles que présentés dans le tableau 4.3. Il s'agit de la moyenne et du coefficient d'asymétrie des entropies spatiales et fréquentielles locales, sur 3 échelles de l'image obtenues par interpolations bicubiques.

4.3 Construction des modèles de prédiction

Lors de la phase d'extraction des caractéristiques des images, 88 d'entre elles ont été retenues pour l'étude. La prochaine étape du processus d'évaluation de la qualité des images sans référence est la construction des modèles d'évaluation de la qualité d'images. Deux processus de construction des modèles d'évaluations sont proposés :

1. Un processus direct de construction des modèles, nommé $DLBQA - ML$ (Direct Learning Blind image Quality Assessment index based on Machine Learning), dans lequel toutes les caractéristiques extraites résultantes de l'étape de suppression des caractéristiques superflues sont utilisées comme entrées des méthodes d'apprentissage, sans phase intermédiaire, pour la construction d'un index d'évaluation de la qualité des images sans référence.
2. Un processus indirect de construction des modèles, nommé $ILBQA - ML$ (Indirect Learning Blind image Quality Assessment index based on Machine Learning), dans lequel les caractéristiques extraites et sélectionnées après l'étape de suppression des caractéristiques superflues sont regroupées en classes, et utilisées pour construire des modèles d'évaluation des métriques intermédiaires. Ce sont ces

métriques intermédiaires qui sont utilisées pour construire le modèle final d'évaluation des images. Dans ce processus, une étape intermédiaire est donc rajoutée pour simplifier la complexité des modèles d'évaluation de la qualité des images construites. Le processus de construction de ces métriques intermédiaires consiste en trois étapes principales : 1) la suppression des caractéristiques superflues, pour éviter des problèmes de sur-apprentissage, ou encore d'augmentation de la complexité des modèles et des temps d'exécution ; 2) regroupement des caractéristiques indépendantes entre elles, pour former des classes indépendantes ; 3) construction des modèles d'évaluation des métriques intermédiaires, sur la base des algorithmes d'apprentissage.

4.3.1 Suppression des caractéristiques superflues

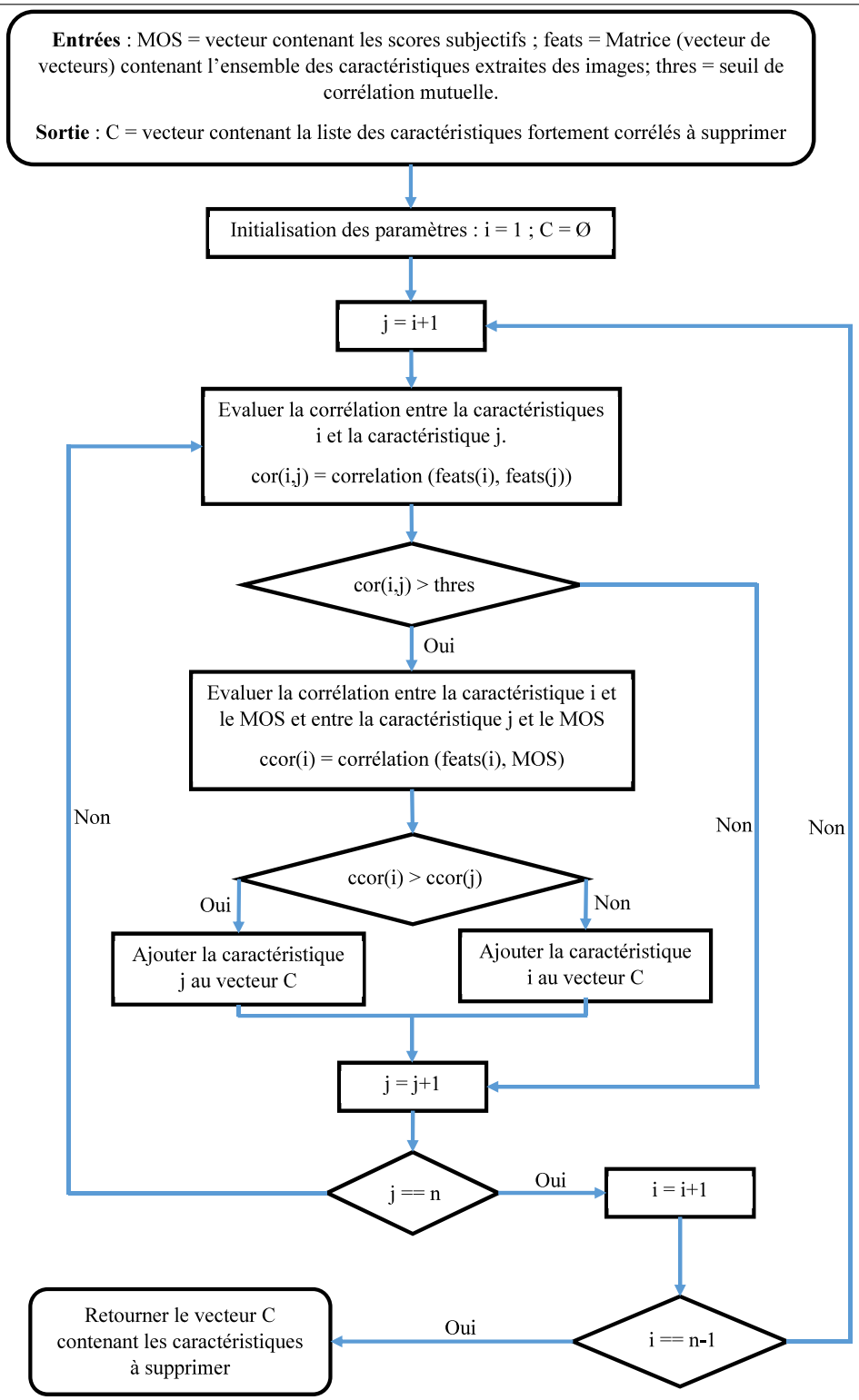
Dans la phase d'extraction des caractéristiques des images, 88 caractéristiques ont été produites suivant 4 axes d'extraction : 36 caractéristiques basées sur les statistiques de scènes naturelles, 20 caractéristiques extraites suivant l'amplitude du gradient, 20 caractéristiques découlant de la caractérisation du laplacien de gaussienne, et 12 caractéristiques basées sur les entropies spatiales et fréquentielles locales. Ces différentes caractéristiques ne sont pas toutes indépendantes entre elles. Ce qui peut causer des redondances d'informations, et complexifier inutilement les modèles produits. Dans cette étape du processus de construction des modèles, le but est donc de supprimer les caractéristiques superflues notamment les valeurs aberrantes, et caractéristiques fortement dépendantes entre elles.

Les caractéristiques aberrantes sont celles qui ont la même valeur, quelle que soit l'image, ou encore celles qui ont un score de corrélation presque nulle avec le score subjectif de la base de données. Parmi les caractéristiques extraites des images, 4 sont des caractéristiques aberrantes.

Les caractéristiques fortement corrélées entre elles apportent approximativement les mêmes informations pour la construction des modèles ; ce qui cause des problèmes de redondance d'informations et augmente la complexité des modèles construits. La proposition pour y remédier est d'effectuer une sélection des caractéristiques à utiliser, en supprimant certaines des caractéristiques fortement corrélées entre elles. Ce processus de suppression des valeurs superflues est basé sur l'idée suivante : « s'il existe dans l'ensemble des caractéristiques, deux fortement corrélées entre elles, avec une corrélation mutuelle supérieure à un certain seuil prédéfini, on supprime celle des deux qui à la moins forte corrélation avec le score subjectif ». Le processus de sélection des caractéristiques superflues, est présenté plus en détail dans l'Algorithme 4.1.

Les caractéristiques contenues dans le vecteur de sortie de cet Algorithme 4.1 sont supprimées de la liste de celles à utiliser dans la suite du travail. Ici a été pris un seuil de corrélation mutuelle entre les caractéristiques de 0.9 ($thes = 90\%$) ; ce qui implique que pour toute corrélation mutuelle entre 2 caractéristiques supérieures à 0.9, une des deux caractéristiques concernées doit être supprimée. Avec ce seuil, le vecteur de sortie contient 38 caractéristiques à supprimer, en plus des 4 caractéristiques aberrantes, soit un total de 42 caractéristiques superflues. Dans la suite de ce travail, seules les 46 caractéristiques

Algorithme 4.1 Procédure de sélection des caractéristiques superflues



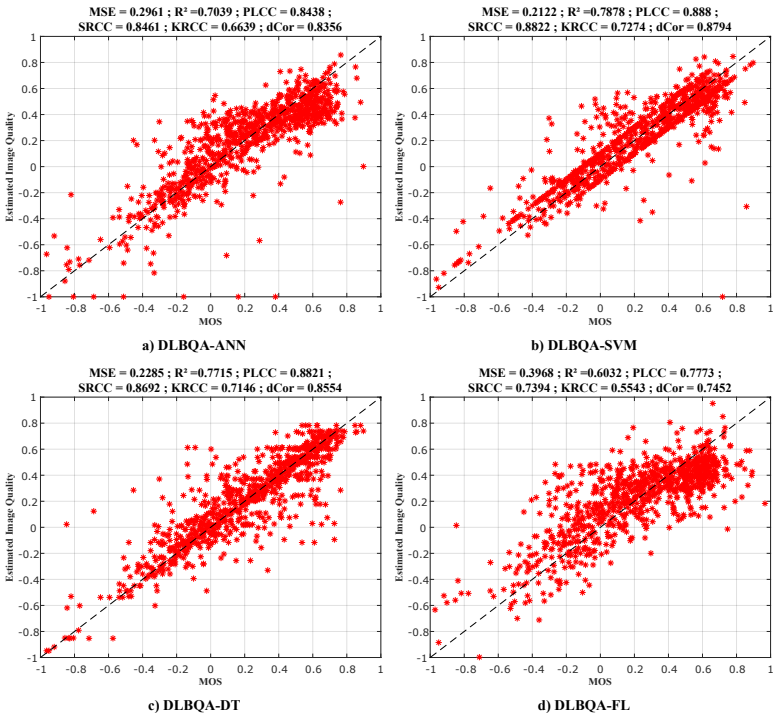


FIGURE 4.2 – Performances des $DLBQA - ML$ sur la base de données $TID2013$

restantes seront utilisées pour construire les modèles.

4.3.2 Processus direct (DLBQA-ML)

Dans la suite du processus, la totalité des 46 caractéristiques étayées et retenues influencent la construction des modèles d'évaluation de la qualité des images. Différentes méthodes d'apprentissage déjà présentées dans les chapitres précédents sont utilisées pour la construction des modèles. Il s'agit notamment des méthodes basées sur les réseaux de neurones artificiels, dont l'index produit a été nommé $DLBQA - ANN$ (Direct Learning Blind image Quality Assessment index based on Artificial Neural Network) ; le modèle basé sur la régression non-linéaire, nommé $DLBQA - NLR$ (Direct Learning Blind image Quality Assessment index based on Non-Linear Regression) ; le modèle basé sur les machines à vecteurs supports nommé $DLBQA - SVM$ (Direct Learning Blind image Quality Assessment index based on Support Vector Machine) ; le modèle basé sur les arbres de décision, nommé $DLBQA - DT$ (Direct Learning Blind image Quality Assessment index based on Decision Tree) et enfin le modèle basé sur la logique floue, nommé $DLBQA - FL$ (Direct Learning Blind image Quality Assessment index based on Fuzzy Logic).

Les Figures 4.2, 4.3 et 4.4 présentent les résultats obtenus sur les données de tests par les modèles produits grâce à ces différentes méthodes d'apprentissage, sur les bases de données d'images $TID2013$, $LIVE$ et $CSIQ$, respectivement. L'échantillon d'apprentis-

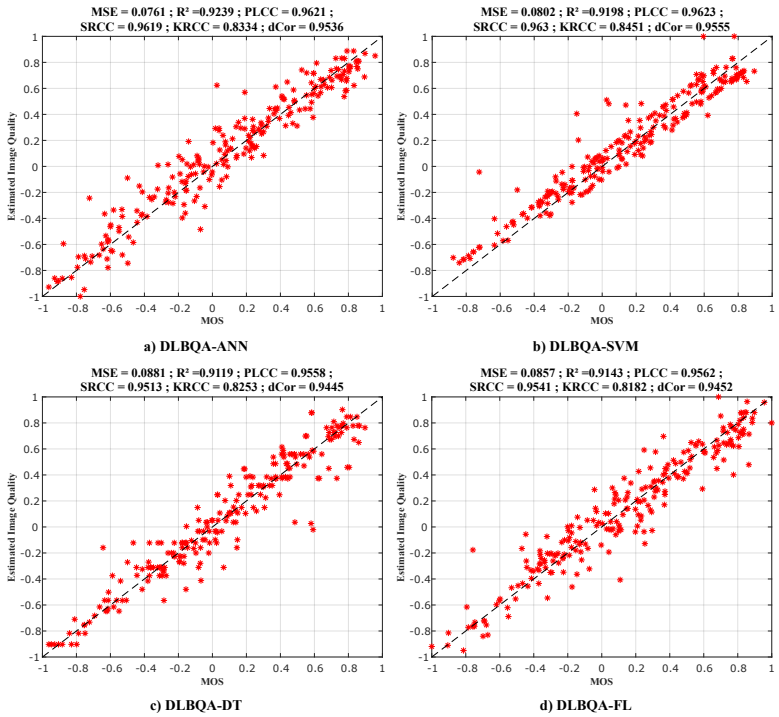


FIGURE 4.3 – Performances des $DLBQA - ML$ sur la base de données $LIVE$

sage utilisé pour la construction des modèles, et celui utilisé pour le test de ces modèles construits, sont tous deux des parties indépendantes, sélectionnées aléatoirement, d'une même base de données d'images. Avec une répartition de 70% des données de la base de données pour l'échantillon d'entraînement (apprentissage) et le reste de données (soit 30% de la base de données) pour l'échantillon de test. Chacun des graphes de ces figures présente la comparaison entre les scores objectifs obtenus lors de l'évaluation des images par les modèles construits (sur l'axe des ordonnées) et les scores subjectifs des images (sur l'axe des abscisses). Ces graphes présentent aussi les indices de performances des modèles, en termes d'erreur quadratique moyenne (MSE), de coefficient de détermination (R^2), de coefficients de corrélation de Pearson ($PLCC$), Spearman ($SRCC$), et Kendall ($KRCC$), et de distance de Brownian ($dCor$), entre les scores subjectifs (MOS / $DMOS$) et les scores objectifs obtenus par les modèles d'évaluation sans référence des images, construits à partir de la méthode directe en un apprentissage.

Les résultats obtenus lors de la phase de test et présentés dans les figures précédentes montrent que les modèles basés sur la méthode des séparateurs à vastes marges ($DLBQA - SVM$) donnent les meilleurs résultats quand l'apprentissage est effectué sur les bases de données d'images $TID2013$ ou $CSIQ$; et que les modèles basés sur la méthode des arbres de décision ($DLBQA - DT$) donnent les meilleurs résultats pour un apprentissage sur la base de données d'images $LIVE$. Toutefois, les résultats obtenus par les modèles $DLBQA - SVM$ et $DLBQA - DT$ ne sont pas très différents quelle que

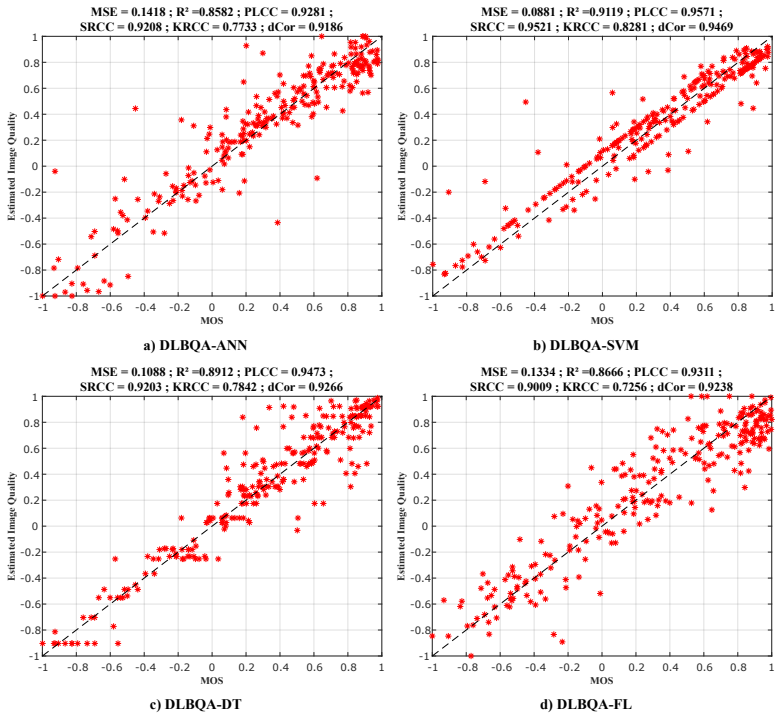


FIGURE 4.4 – Performances des $DLBQA - ML$ sur la base de données $CSIQ$

soit la base de données d'apprentissage. Ces résultats seront confirmés lors de la phase de validation croisée dans la section suivante sur les résultats expérimentaux et leur comparaison. Par la suite en parlant du modèle $DLBQA$ sans précision sur la méthode d'apprentissage utilisée, il s'agira des modèles $DLBQA - ML$ produisant les meilleurs résultats sur les bases de données d'images testées.

4.3.3 Processus indirect (ILBQA-ML)

Dans ce second processus dit indirect, les modèles d'évaluations sont trouvés après deux phases d'apprentissage : une première phase regroupant les caractéristiques extraites suivant le même axe dans des classes pour construire ce que l'on a appelé ici des métriques intermédiaires ; et une seconde étape de construction des modèles d'évaluation de la qualité visuelle des images utilisant ces métriques intermédiaires trouvées dans la première phase du processus.

4.3.3.1 Construction des modèles intermédiaires

Les caractéristiques extraites des images retenues après la phase de suppression des caractéristiques aberrantes et superflues sont regroupées en classes. Les classes produites sont ensuite utilisées avec des méthodes d'apprentissage pour la construction des métriques intermédiaires. Pour des raisons de simplicité et de cohérence dans la construction

TABLE 4.4 – Corrélation de Pearson entre les métriques intermédiaires et le MOS pour **TID2013**

Méthode	$IM-NSS$	$IM-GM$	$IM-LOG$	$IM-SSE$
ANN	0.7161	0.7046	0.6007	0.6464
NLR	0.6195	0.6442	0.5888	0.6034
SVM	0.7297	0.653	0.5709	0.746
DT	**0.8558**	**0.8563**	**0.7717**	**0.8447**
FL	0.6262	0.6553	0.5314	0.5918

des modèles d'évaluation des métriques intermédiaires, les classes de regroupement des caractéristiques utilisent les différents axes d'extraction de caractéristiques pour leur regroupement. Ainsi quatre classes ont été définies :

1. la première basée sur les statistiques de scènes naturelles ($IM-NSS$: Intermediary Metric based on Natural Scene Statistics), et contenant 14 caractéristiques extraites ;
2. la deuxième basée sur l'amplitude du gradient ($IM-GM$: Intermediary Metric based on Gradient Magnitude), et contenant 13 caractéristiques extraites ;
3. puis la troisième basée sur le laplacien de gaussienne ($IM-LOG$: Intermediary Metric based on Laplacian of Gaussian), et contenant 11 caractéristiques extraites ;
4. et enfin la quatrième basée sur les entropies spatiales et fréquentielles locales ($IM-SSE$: Intermediary Metric based on Spatial and Spectral Entropies), et contenant 8 caractéristiques extraites.

Les classes de caractéristiques sont utilisées pour produire les modèles d'évaluation des métriques intermédiaires, en se basant sur les méthodes d'apprentissage automatique suivantes : régression non linéaire polynomiale (NLR), machines à vecteurs supports (SVM), réseaux de neurones artificiels (ANN), arbres de décision (DT), et logique floue (FL). Le Tableau 4.4 présente les résultats en terme de corrélation linéaire de Pearson obtenues par chacune des métriques intermédiaires sur les données de test (constituées de 30% des données de la base de données d'images $TID2013$). Sachant que ces métriques intermédiaires sont construites à partir de différentes méthodes d'apprentissage, et en utilisant les données d'apprentissage constituées des 70% des données restantes de la base de données d'images $TID2013$.

Dans un objectif de simplification des modèles à utiliser, la même méthode d'apprentissage a été retenue pour toutes les métriques intermédiaires. Pour déterminer la méthode d'apprentissage à utiliser, les distances de Brownian multiples entre les 4 métriques intermédiaires et les scores subjectifs (MOS / $DMOS$) ont été évaluées pour chacune des méthodes d'apprentissage utilisées. Le Tableau 4.5 présente ces distances de Brownian obtenues pour les bases de données d'images $CSIQ$, $LIVE$ et $TID2013$.

TABLE 4.5 – Distance de Brownian entre les groupes de métriques intermédiaires et le MOS / DMOS

BD	**ANN**	**NLR**	**SVM**	**DT**	**FL**
TID2013	0.7534	0.7432	0.8381	**0.9043**	0.6663
LIVE	0.9589	0.9599	0.9604	**0.9677**	0.9381
CSIQ	0.8692	0.9046	0.8887	**0.9298**	0.8283

Dans ce tableau, on observe que la méthode d'apprentissage basée sur les arbres de décision produit les meilleurs résultats pour les trois bases de données utilisées, en termes de corrélation multiple de Brownian, entre les quatre métriques intermédiaires, et les scores subjectifs (MOS / $DMOS$). Les métriques intermédiaires sont donc toutes construites en utilisant la méthode d'apprentissage basées sur les arbres de décision (DT).

4.3.3.2 Construction du modèle indirect de prédiction

Une fois les métriques intermédiaires construites, la prochaine étape du processus indirect est la construction des modèles d'évaluation de la qualité des images utilisant ces métriques intermédiaires. L'index d'évaluation final construit est constitué de plusieurs modèles : 4 modèles de construction des métriques intermédiaires, et un modèle d'évaluation du score final à partir des métriques intermédiaires. Les métriques intermédiaires ayant été construites dans la section précédente, il ne reste plus que la construction du modèle d'évaluation de la qualité d'image, prenant comme entrées les métriques intermédiaires précédemment évaluées. Ces modèles sont construits grâce aux algorithmes d'apprentissage automatiques.

Les Figures 4.5, 4.6 et 4.7 présentent les performances de prédiction des modèles $ILBQA - ML$ obtenus sur les échantillons de tests des bases de données d'images $CSIQ$, $LIVE$ et $TID2013$, respectivement. L'échantillon d'apprentissage utilisé pour la construction des modèles d'évaluation des métriques intermédiaires et d'évaluation de la qualité des images à partir des métriques intermédiaires, ainsi que l'échantillon de test utilisé pour l'évaluation des modèles construits sont tous deux, des parties indépendantes sélectionnées aléatoirement d'une même base de données d'images. Avec une répartition de 70% des données de la base de données d'images pour l'échantillon d'entraînement des métriques intermédiaires, puis de la construction du modèle d'évaluation de la qualité des images ; et le reste des données (soit 30% de la base de données) pour l'échantillon de test. Chacun des graphes de ces figures présente la comparaison entre les scores objectifs obtenus lors de l'évaluation des images par les modèles construits (sur l'axe des ordonnées) et les scores subjectifs des images (sur l'axe des abscisses), ainsi que les indices de performances des modèles en termes de corrélations et d'erreurs entre les scores subjectifs (MOS / $DMOS$) et les scores objectifs obtenus par les modèles d'évaluation de la qualité des images, construits à partir de la méthode indirecte de construction des

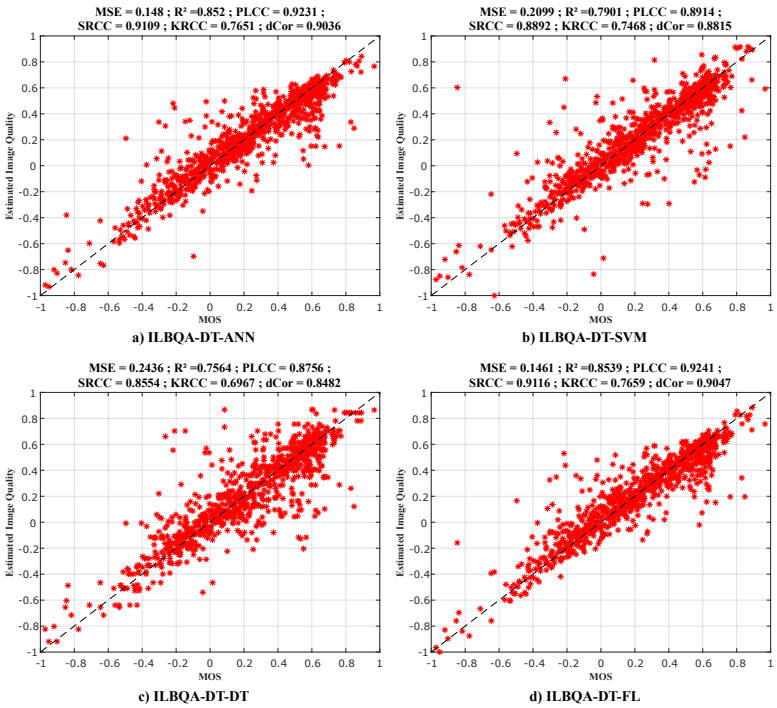


FIGURE 4.5 – Performances des ILBQA-ML sur la base de données TID2013

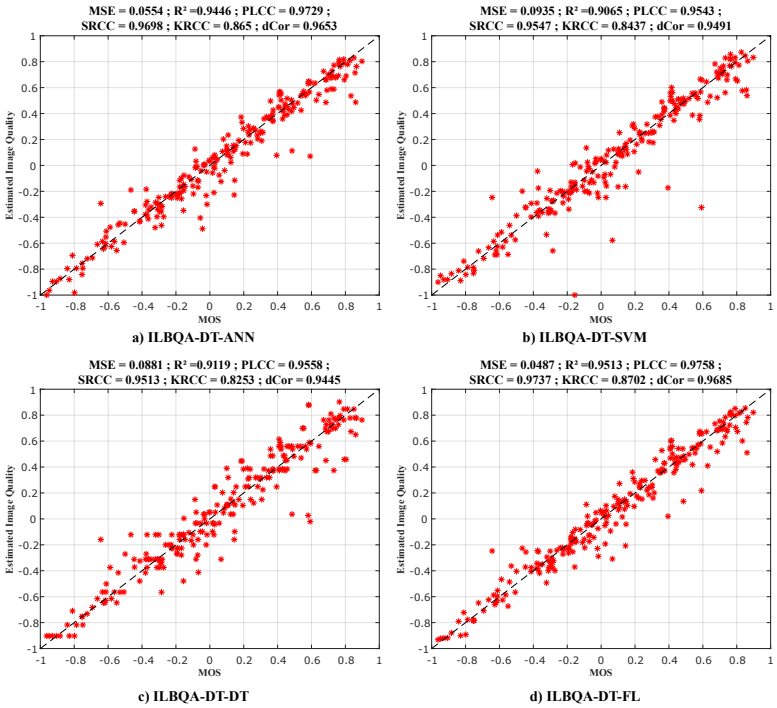


FIGURE 4.6 – Performances des ILBQA-ML sur la base de données LIVE

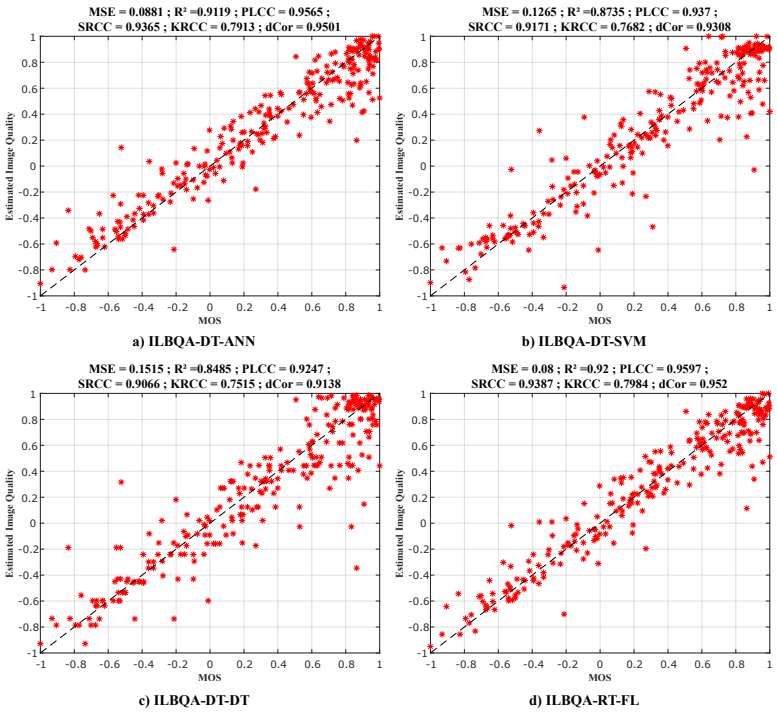


FIGURE 4.7 – Performances des ILBQA-ML sur la base de données CSIQ

modèles.

Les modèles sont construits sur la base de 2 apprentissages successifs. Dans les index, sont précisées les méthodes : celle d'apprentissage des métriques intermédiaires et celle de construction du modèle final d'évaluation de la qualité des images. Les différents index obtenus sont donc : $ILBQA-DT-ANN$, basé sur les arbres de décision pour les métriques intermédiaires, et sur les réseaux de neurones artificiels pour le modèle final ; $ILBQA-DT-SVM$, basé sur les arbres de décision et les machines à vecteurs supports ; $ILBQA-DT-DT$, tous les modèles sont basés sur les arbres de décision ; $ILBQA-DT-FL$, basé sur les arbres de décision et sur la logique floue.

On remarque sur ces graphiques que les modèles basés sur la logique floue produisent les meilleurs résultats en termes d'indices de performances entre les scores objectifs évalués et les scores subjectifs, sur toutes les trois bases de données d'images sur lesquelles ces modèles ont été construits et testés. Par la suite en parlant du modèle $ILBQA$, sans donner de précision sur les méthodes d'apprentissage utilisées, il s'agira du modèle produisant les meilleurs résultats, suivant la base de données d'images utilisée. Ici, quelle que soit la base de données utilisée les meilleurs résultats sont données par le modèle $ILBQA-DT-FL$, basé sur les arbres de décision pour la construction des métriques intermédiaires, et sur la logique floue pour le modèle final utilisant les métriques intermédiaires en vue de produire le score objectif estimé de l'image.

La prochaine étape consiste à tester la stabilité et la robustesse des modèles pro-

duits, puis à comparer les résultats obtenus par ces modèles, avec ceux d'autres modèles existant dans la littérature.

4.4 Résultats expérimentaux

Le processus de construction des modèles d'évaluation se fait en trois phases principales : la phase d'entraînement du modèle à partir de $\alpha\%$ des données de la base de données ; la phase de test, utilisant le reste des $(100-\alpha)\%$ des données non utilisées pour l'apprentissage ; et la phase de validation croisée, qui répète les phases d'apprentissage et de test k fois, en faisant varier à chaque fois les données d'apprentissage et de test de manière aléatoire.

Dans la phase d'entraînement, les caractéristiques extraites des images, ainsi que le score subjectif (MOS / $DMOS$) des images d'entrainement, sont utilisés par les méthodes d'apprentissage pour produire des modèles qui seront repris pour l'évaluation de la qualité des images. La phase de test, quant à elle, consiste à évaluer la qualité des images de tests en utilisant les caractéristiques extraites de ces images de tests, et en comparant les résultats obtenus aux scores subjectifs (MOS / $DMOS$) de ces images de tests. La phase de test permet de jauger les performances des modèles construits lors de la phase d'entraînement, à partir des données qui n'ont pas été utilisées lors cette dernière phase. La phase de validation croisée permet de vérifier que les modèles produits ne sont pas dépendant de la répartition des données d'apprentissage et de test. Cette phase de validation permet non seulement de tester les performances des modèles, mais aussi de tester la stabilité et la robustesse des méthodes d'apprentissage et des modèles produits.

Cinq méthodes d'apprentissage ont été utilisées pour la construction des modèles. Ces méthodes sont basées sur : les réseaux de neurones artificiels (ANN), la régression non-linéaire (NLR), les machines à vecteurs supports (SVM), les arbres de décision (DT) et la logique floue (FL). La phase de validation, quant à elle, est effectuée grâce à la méthode de validation croisée de Monte Carlo [44], décrite dans le chapitre sur l'état de l'art. Ont donc été pris en considération : $\alpha\% = 70\%$ de la base de données d'images pour l'entraînement et les 30% de données restantes pour les tests ; et pour la validation croisée $k = 1000$ fois.

4.4.1 Comparaison des résultats

Cette sous-section établit la comparaison des résultats obtenus entre les différents modèles d'évaluation de la qualité des images construits. Les résultats obtenus sont aussi comparées à ceux d'autres index sans référence d'évaluation de la qualité des images ($BRISQUE$, $GMLOGQA$, $SSEQ$) présents dans la littérature. Ces comparaisons se feront pour la plupart en phase de validation, grâce à la méthode de validation croisée de Monte Carlo, répétée $k = 1000$ fois, avec une séparation des bases de données à $70 - 30\%$ pour les données d'entraînement et de test, respectivement.

Les Figures 4.8 à 4.10 et les Tableaux 4.6 à 4.11 illustrent une comparaison des performances de prédiction de cinq différentes méthodes sans référence d'évaluation de la qualité des images, apprises sur les bases de données d'images $TID2013$, $LIVE$ et $CSIQ$. Les Tableaux présentent les résultats en termes de précision, donnée par les moyennes des indices de performance, et de stabilité des modèles, données par les écarts-types de ces indices de performance. Ces résultats sont obtenus lors de la phase de validation croisée de Monte Carlo, par différents index d'évaluation sans référence de la qualité des images. Les résultats de comparaison sont donnés en termes d'indice de performance des comparaisons entre les scores subjectifs et les scores objectifs obtenus par les images. Les indices de performance présentés ici sont : l'erreur quadratique moyenne(MSE), le coefficient de détermination (R^2) ; les coefficients de corrélation de Pearson ($PLCC$), Spearman ($SRCC$) et Kendall ($KRCC$) ; et la distance de corrélation de Brownian ($dCor$). Les Figures quant à elles présentent les boites à moustaches (box-plots) des distances de Brownian ($dCor$) ; celles des autres indices de performance étant presque identiques à celles présentées par la distance de Brownian.

Dans la Figure 4.11, apparaît une comparaison globale de toutes les cinq méthodes sans référence d'évaluation de la qualité des images présentées dans cette section. Cette comparaison globale est faite pour différentes bases de données d'images : $TID2013$, $LIVE$ et $CSIQ$; sur la corrélation linéaire de Pearson entre les scores subjectifs (MOS / $DMOS$) des bases de données d'images, et les scores objectifs produits par les modèles d'évaluation lors de la phase de validation croisée, répétée 1000 $fois$.

Il apparaît sur ces figures et tableaux que :

1. Les résultats obtenus par les deux modèles $DLBQA$ et $ILBQA$, produisent généralement des meilleurs résultats que d'autres méthodes sans référence existantes, en termes de précision (moyennes plus élevées) et de stabilité (écarts-types plus faibles).
2. Les résultats obtenus par le processus indirect ($ILBQA$) avec deux apprentissages et l'intégration des métriques intermédiaires, sont meilleurs sur les bases de données d'images $LIVE$ et $TID213$ que ceux obtenus par le processus direct ($DLBQA$), avec un seul apprentissage ; mais sur la base de données $CSIQ$, les résultats obtenus par le modèle $DLBQA$ sont meilleurs que ceux obtenus pas le modèle $ILBQA$.

4.4.2 Indépendance de la base de données

Jusqu'à présent, les performances de prédiction des modèles réalisés sont testées sur les échantillons venant de la base de données d'images sur laquelle ils ont été appris. Dans cette section, est testée l'indépendance des modèles par rapport aux bases de données dont on s'est inspiré. Pour cela, les modèles sont appris sur une base de données et testés sur plusieurs bases de données différentes. Les bases de données d'images $CSIQ$ [26, 25], $LIVE$ [10, 23, 24] et $TID2013$ [27, 4] ont été utilisées pour cette étude sur l'indépendance des modèles produits. Les résultats obtenus sont non seulement comparés entre eux, mais aussi à quelques autres obtenus par diverses méthodes sans référence d'évaluation de la qualité d'images, existant dans la littérature.

TABLE 4.6 – Comparaison des performances (**Moyenne**) pour différentes méthodes sans référence ; entraînées sur **TID2013**

	MSE	R^2	**PLCC**	**SRCC**	**KRCC**	**dCor**
BRISQUE	0.3766	0.6234	0.7974	0.7771	0.5967	0.7806
GMLOGQA	0.3795	0.6205	0.8037	0.7933	0.6052	0.7908
SSEQ	0.3744	0.6256	0.7931	0.7678	0.5829	0.7695
DLBQA	0.181	0.819	0.9054	0.8956	0.7383	0.8925
ILBQA	**0.1509**	**0.8491**	**0.9217**	**0.9114**	**0.7669**	**0.9032**

TABLE 4.7 – Comparaison des performances (**Écart-type**) pour différentes méthodes sans référence ; entraînées sur **TID2013**

STD ($\times 10^{-2}$)	**MSE**	R^2	**PLCC**	**SRCC**	**KRCC**	**dCor**
BRISQUE	2.702	2.702	1.452	1.485	1.355	1.381
GMLOGQA	2.94	2.94	1.463	1.404	1.314	1.358
SSEQ	2.154	2.154	1.337	1.536	1.388	1.419
DLBQA	1.724	1.724	0.943	0.906	**0.921**	**0.817**
ILBQA	**1.425**	**1.425**	**0.769**	**0.87**	1.07	0.892

TABLE 4.8 – Comparaison des performances (**Moyenne**) pour différentes méthodes sans référence ; entraînées sur **LIVE**

	MSE	R^2	**PLCC**	**SRCC**	**KRCC**	**dCor**
BRISQUE	0.0966	0.9034	0.9515	0.9501	0.8183	0.9441
GMLOGQA	0.0821	0.9179	0.9589	0.9578	0.827	0.9527
SSEQ	0.1991	0.8009	0.8975	0.9016	0.7295	0.8919
DLBQA	0.0666	0.9334	0.9666	0.9633	0.8578	0.9565
ILBQA	**0.0566**	**0.9434**	**0.9716**	**0.9689**	**0.8643**	**0.9641**

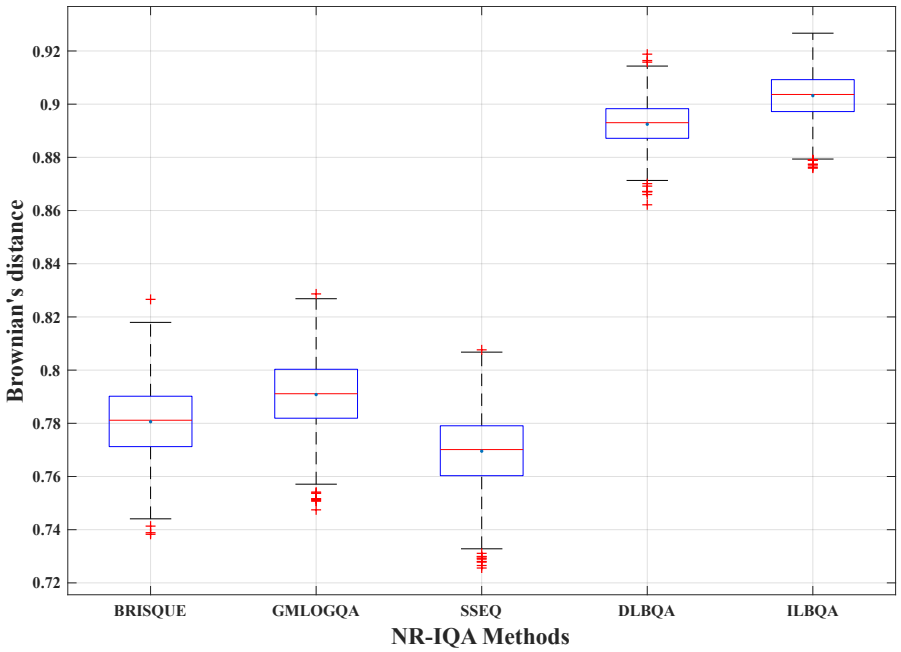


FIGURE 4.8 – Comparaison des performances pour différentes méthodes sans référence ; entraînées sur TID2013

TABLE 4.9 – Comparaison des performances (**Écart-type**) pour différentes méthodes sans référence ; entraînées sur **LIVE**

STD ($\times 10^{-2}$)	**MSE**	R^2	**PLCC**	**SRCC**	**KRCC**	**dCor**
BRISQUE	2.213	2.213	1.119	1.065	1.464	1.138
GMLOGQA	1.343	1.343	0.672	**0.614**	**1.136**	**0.714**
SSEQ	2.373	2.373	1.343	1.299	1.679	1.432
DLBQA	**1.237**	**1.237**	**0.625**	0.714	1.223	0.763
ILBQA	1.299	1.299	0.65	0.655	1.24	0.728

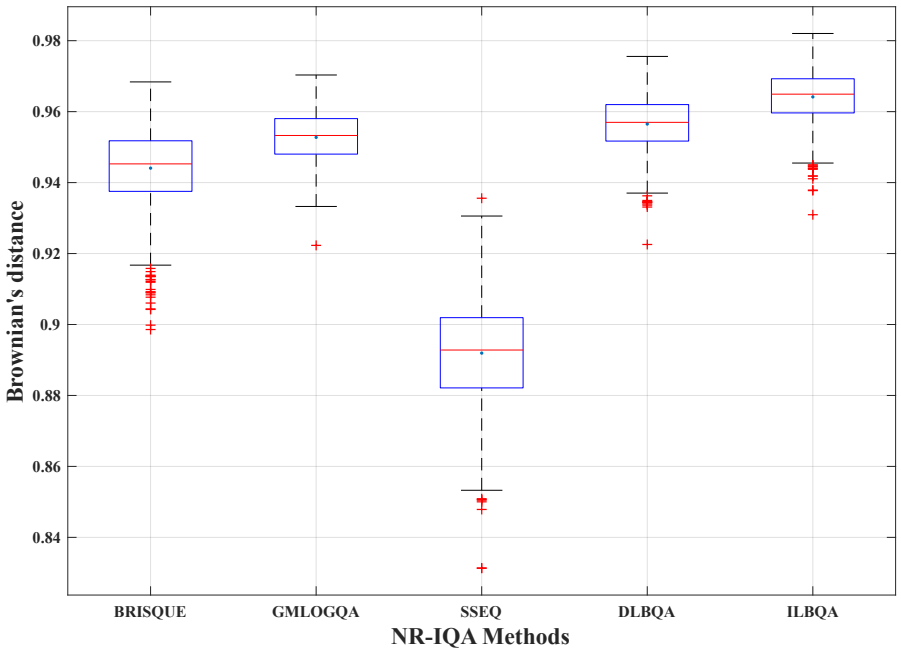


FIGURE 4.9 – Comparaison des performances pour différentes méthodes sans référence ; entraînées sur LIVE

TABLE 4.10 – Comparaison des performances (**Moyenne**) pour différentes méthodes sans référence ; entraînées sur **CSIQ**

	MSE	R^2	**PLCC**	**SRCC**	**KRCC**	**dCor**
BRISQUE	0.2468	0.7532	0.8721	0.8559	0.68020	0.8591
GMLOGQA	0.291	0.7090	0.8476	0.8335	0.6428	0.8325
SSEQ	0.2637	0.7363	0.8601	0.8495	0.6652	0.8424
DLBQA	**0.0898**	**0.9101**	**0.9571**	**0.9471**	**0.8185**	**0.9472**
ILBQA	0.1273	0.8727	0.9349	0.9243	0.7830	0.9203

TABLE 4.11 – Comparaison des performances (**Écart-type**) pour différentes méthodes sans référence ; entraînées sur **CSIQ**

STD ($\times 10^{-2}$)	MSE	R^2	PLCC	SRCC	KRCC	dCor
BRISQUE	3.733	3.733	1.908	2.035	2.168	1.986
GMLOGQA	3.674	3.674	1.941	2.005	2.055	1.962
SSEQ	2.847	2.847	1.683	1.901	2.066	1.959
DLBQA	**1.438**	**1.438**	**0.728**	**0.81**	**1.237**	**0.799**
ILBQA	2.373	2.373	1.235	1.323	1.756	1.369

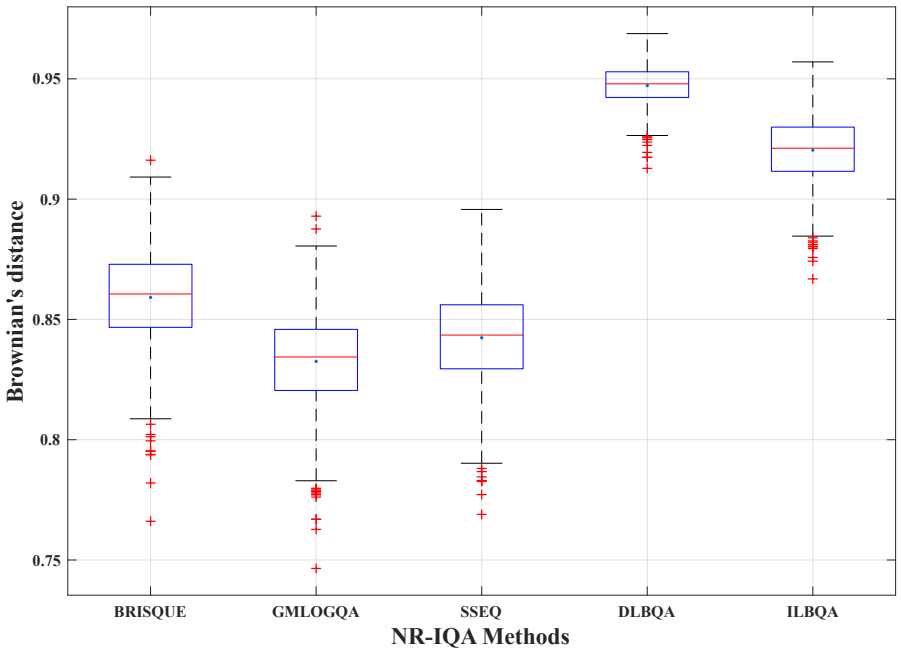


FIGURE 4.10 – Comparaison des performances pour différentes méthodes sans référence ; entraînées sur CSIQ

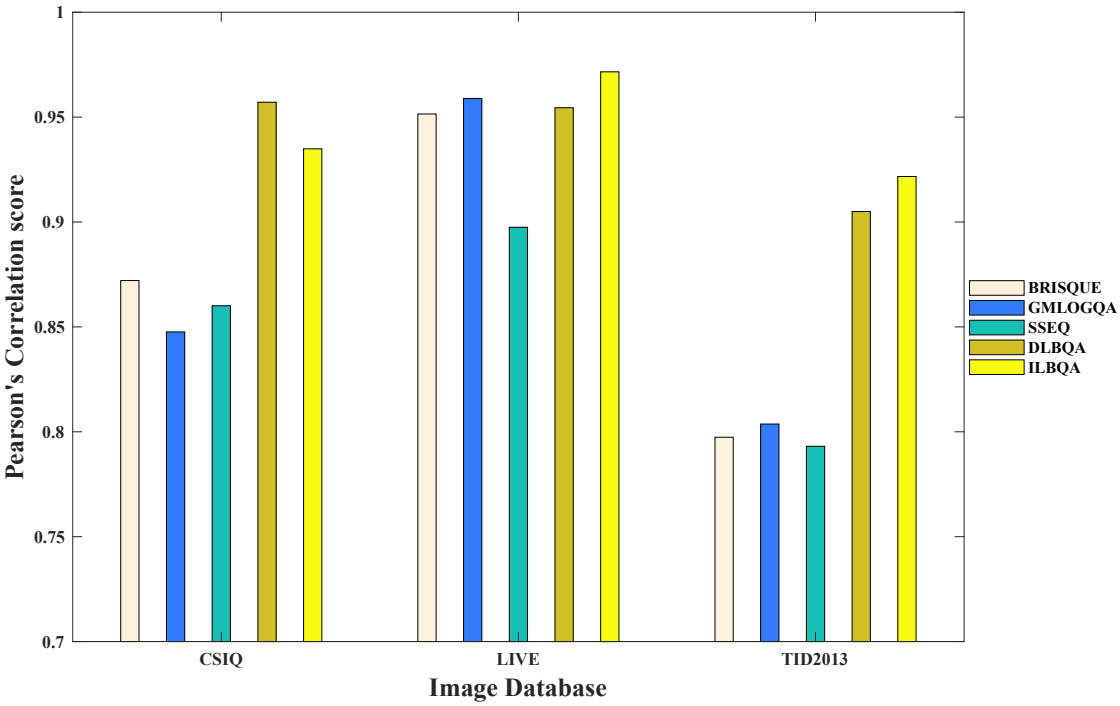


FIGURE 4.11 – Comparaison des performances pour différentes méthodes sans référence sur différentes bases de données

TABLE 4.12 – PLCC des données entraînées sur une base de données et testées sur d'autres bases de données d'images

BD Train	BD test	BRISQUE	GMLOGQA	SSEQ	**DLBQA**	**ILBQA**
TID2013	Test	0.7974	0.8037	0.7931	0.9050	**0.9217**
	TID2013	0.8693	0.9014	0.9005	0.9015	**0.9255**
	LIVE	**0.8207**	0.7809	0.6078	0.6178	0.7548
	CSIQ	**0.6463**	0.5588	0.5182	0.4190	0.6429
LIVE	Test	0.9515	0.9589	0.8975	0.9545	**0.9716**
	TID2013	0.4198	0.3551	**0.5168**	0.3900	0.4587
	LIVE	**0.9777**	0.9694	0.9523	0.9661	0.9769
	CSIQ	0.6845	**0.7112**	0.6204	0.6488	0.6937
CSIQ	Test	0.8721	0.8476	0.8601	**0.9571**	0.9349
	TID2013	0.4389	0.2237	**0.4634**	0.4425	0.4524
	LIVE	0.7337	0.7579	0.6030	0.6789	**0.7936**
	CSIQ	0.9312	0.9001	0.9515	**0.9529**	0.9447
Globale	Test	0.7126	0.7774	0.6126	0.7830	**0.8239**
	TID2013	0.7006	0.7458	0.6654	0.7019	**0.7807**
	LIVE	0.8733	**0.9232**	0.8193	0.8663	0.9082
	CSIQ	0.8393	0.8002	0.7781	0.7931	**0.8432**

Les Tableaux 4.12 et 4.13 présentent les résultats de la comparaison entre les scores subjectifs (MOS / $DMOS$) et les scores objectifs obtenus par les modèles construits par apprentissage avec une partie (70%) de la base de données d'apprentissage, et testé sur différentes bases de données :

— le reste des données non utilisées pour l'apprentissage de la base de données d'apprentissage ($Test$) ;
— toutes les images de la base de données $TID213$;
— toutes les images de la base de données $LIVE$;
— toutes les données de la base de données $CSIQ$.

La base de données « Globale » regroupe toutes les trois bases de données étudiées ici : $TID2013$, $LIVE$ et $CSIQ$). Pour la construction des modèles à partir de cette base de données, l'espace d'apprentissage n'est constitué que de 30% de l'ensemble de la base de données. Les résultats obtenus sont comparés en termes de corrélation linéaire de Pearson et de corrélation de rang de Spearman, dans les Tableaux 4.12 et 4.13, respectivement.

La remarque principale tirée de ces deux tableaux est que les modèles entraïnés sur une

TABLE 4.13 – SRCC des données entraînées sur une base de données et testées sur d'autres bases de données d'images

BD Train	BD test	BRISQUE	GMLOGQA	SSEQ	DLBQA	ILBQA
TID2013	Test	0.7771	0.7933	0.7678	0.8956	**0.9114**
	TID2013	0.8443	0.8891	0.8798	0.8981	**0.9157**
	LIVE	**0.8242**	0.7974	0.6527	0.6482	0.7691
	CSIQ	**0.5855**	0.5432	0.4679	0.4093	0.5576
LIVE	Test	0.9501	0.9579	0.9016	0.9516	**0.9689**
	TID2013	0.3702	0.3517	**0.4790**	0.3540	0.3903
	LIVE	**0.9773**	0.9684	0.9513	0.9630	0.9751
	CSIQ	0.5690	0.5847	0.5778	0.5400	**0.5865**
CSIQ	Test	0.8559	0.8335	0.8495	**0.9471**	0.9243
	TID2013	**0.4314**	0.2906	0.4413	0.4046	0.3996
	LIVE	0.7503	0.7759	0.6317	0.6957	**0.7937**
	CSIQ	0.9239	0.8814	0.9453	**0.9493**	0.9350
Globale	Test	0.7150	0.7469	0.6532	0.7528	**0.7900**
	TID2013	0.7165	0.7125	0.7038	0.6694	**0.7441**
	LIVE	0.8836	**0.9226**	0.8420	0.8580	0.9002
	CSIQ	**0.8283**	0.7597	0.7839	0.7648	0.8127

TABLE 4.14 – Pourcentage de temps d'utilisation pour chaque axe d'extraction

Axe d'extraction des caractéristiques	**% de temps**
Statistiques de scènes naturelles	7.98
Amplitude du gradient	3.10
Laplacien de la Gaussienne	3.10
Entropies spatiales et fréquentielles	85,77

base de données produisent presque toujours de meilleurs résultats lorsque les images de tests sont issues de cette même base de données d'entraînement. Ces écarts sont dus à la différence dans la manière d'évaluer les scores subjectifs des bases de données d'images, mais aussi aux différences de nombre et niveaux de distorsions dans les différentes bases de données d'images.

4.4.3 Analyse de la complexité en temps

Le temps d'exécution est un facteur important pour la comparaison des méthodes objectives d'évaluation de la qualité des images, surtout lorsqu'il s'agit des méthodes sans référence. Car elles sont, pour la plupart, utilisées pour des tests en ligne de la qualité des images et des applications temps-réel. Le Tableau 4.14 présente le pourcentage de la complexité en temps pour chacun des axes d'extraction des caractéristiques d'images, pour la méthode proposée. Le temps d'exécution des modèles produits occupe 0.054% du temps total d'exécution de l'index ; ce ce qui le rend négligeable par rapport au temps d'extraction des caractéristiques.

Les performances ont aussi été comparées en termes de temps d'exécution de l'index ($ILBQA$), avec ceux d'autres index existant tels que $BLIINDS-II$ [127], $DIIVINE$ [13], $BRISQUE$ [15, 16], $GMLOGQA$ [59], et $SSEQ$ [117]. Le Tableau 4.15 présente les temps d'exécution en secondes (s) sous l'outil $MATLAB$, des différentes méthodes d'évaluation de la qualité des images, pour une image de résolution 512×768. Cette évaluation est faite sur un ordinateur personnel de $16\,GB$ de RAM, avec un processeur à deux cœurs Intel Xeon ($1.70Ghz$ chacun). On remarque dans ce tableau que l'index proposé a un temps d'exécution plutôt faible.

4.5 Architectures et résultats d'implémentation

Après la construction des modèles d'évaluation des images sans référence, grâce à l'outil $MATLAB$, la prochaine étape du processus consiste en l'implémentation des modèles obtenus sur un système $FPGA$. Cette implémentation est faite en vue de la production d'un circuit $ASIC$, pouvant être intégrés à la sortie d'un capteur d'images pour évaluer la qualité des images produites. L'index implémenté ici, est celui du processus indirect

TABLE 4.15 – Comparaison de la complexité en temps pour différentes méthodes

Méthode	**Temps d'exécution (s)**
BLIINDS-II	76,12
DIIVINE	25,40
BRISQUE	0.168
GMLOGQA	0.131
SSEQ	1.807
ILBQA	2.126

$ILBQA$ (Indirect Learning Blind image Quality Assessment), qui consiste en une extraction des caractéristiques suivant quatre axes. Les caractéristiques ainsi extraites sont utilisées pour la construction des métriques intermédiaires, ce qui constitue le premier niveau d'évaluation. Par la suite, est mis sur pied un second niveau d'évaluation prenant en entrées les métriques intermédiaires, pour produire en sortie le score de qualité de l'image.

Cette implémentation est faite sur une carte $FPGA$ de la famille des Virtex 7 ($VC707$), grâce à la gamme d'outil Vivado de Xilinx. Le processus général d'implémentation consiste en :

1. une implémentation HLS des blocs IP (IP-Core) permettant la mise en place des modèles conçus dans les sections précédentes, à partir de l'outil Vvado HLS (code en langage C/C++) ;
2. une intégration des blocs IP conçus et synthétisés dans la première partie du processus, dans le design global au niveau RTL, sous l'outil Vivado (langage de description $VHDL$)
3. une intégration sur la carte $FPGA$ de la famille Virtex 7 ($VC707$) de Xillinx, et une évaluation des résultats obtenus sur plusieurs images de tests.

4.5.1 Implémentation HLS des modèles ILBQA

L'implémentation des modèles de la mesure $ILBQA$ en HLS a permis la construction de cinq blocs IP divisés en trois grands groupes devant conduire à l'évaluation du score objectif d'une image prise en entrée des modèles, sans aucune information sur l'image de référence. Les blocs IP de chaque groupe peuvent s'exécuter en parallèle, mais les entrées de chaque groupe dépendent des sorties du groupe précédent. Les trois groupes sont : l'extraction des caractéristiques ; la construction des métriques intermédiaires à partir des modèles basés sur les arbres de décision (DT) ; et enfin l'évaluation du score objectif à partir du modèle basé sur la logique floue, prenant en entrée les métriques intermédiaires. L'architecture en couches des blocs IP produits est donnée dans la Figure 4.12.

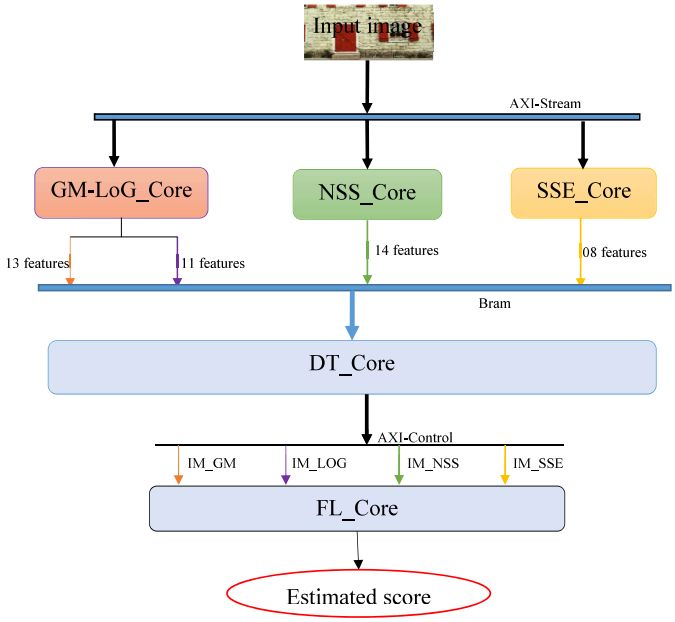


FIGURE 4.12 – Architecture d'implémentation en blocs IP des modèles $ILBQA$

1. **Extraction des caractéristiques** : l'entrée de ce groupe est l'image dont on cherche à évaluer la qualité. Elle est transmise de l'ordinateur au format « $AXI-Stream$ » suivant le protocole de transmission des données AXI. La transmission se fait grâce au bloc d'accès direct à la mémoire appelé $AXI-DMA$ (Direct Acces Memory). Ce groupe comprend trois blocs IP :
   - NSS_Core : Il s'agit du bloc IP produit pour l'extraction des caractéristiques suivant l'axe des statistiques de scènes naturelles dans le domaine spatial. Il produit 14 caractéristiques extraites de l'image prise comme entrée. Les caractéristiques produites sont rangées dans un vecteur stocké dans une mémoire BRAM.
   - GM-LOG_Core : L'amplitude du gradient et le Laplacien de la Gaussienne ayant des caractéristiques communes, ce bloc IP permet une production des caractéristiques suivant ces deux axes. En sortie il produit deux vecteurs contenant, pour le premier les 13 caractéristiques extraites suivant l'axe de l'amplitude du gradient (GM), et pour le second les 11 caractéristiques extraites suivant l'axe du laplacien de la gaussienne (LOG).
   - SSE_Core : Le dernier bloc IP de ce premier groupe, quant à lui, produit les caractéristiques suivant l'axe des entropies spatiales et fréquentielles locales de l'image prise en entrée. Il produit en sortie un vecteur stocké dans une BRAM, contenant 8 caractéristiques extraites de l'image suivant l'axe des entropies locales.
2. **Construction des métriques intermédiaires** : après l'extraction des caractéristiques dans le premier groupe des blocs IP, ces caractéristiques sont transmises à ce second groupe qui prend comme entrées les vecteurs de caractéristiques extraites suivant différents axes, et produit en sortie la métrique intermédiaire suivant cha-

TABLE 4.16 – Estimation des performances et utilisation des blocs IP du modèle ILBQA dans la phase de synthèse HLS

	NSS	**GM-LOG**	**SSE**	**DT**	**FL**	**ILBQA**
clock estim. (ns)	21.79	21.79	21.84	12.50	17.41	21.85
# cycle min	79 302	1 648	570	2	42	80 370
# cycle max	9 400 263	2 909 751	25 231 749	17	48	84 720 454
BRAM	1 235	1 052	554	0	0	2328
DSP	394	436	141	0	200	1 136
FF	59 097	69 705	20 872	10 936	8 878	167 801
LUT	181 656	247 775	50 903	128 122	22 110	497 622

cun des axes, grâce à son modèle produit par la méthode d'apprentissage des arbres de décision. Ce groupe utilise un seul bloc IP (DT_Core) pour la production des quatre métriques intermédiaires ; ces métriques sont produites au format flottant et transmises au groupe suivant grâce au protocole de contrôle AXI (AXI-Control).

3. **Production du score objectif de l'image** : ce dernier groupe utilise les métriques intermédiaires produites dans le groupe précédent pour produire le score objectif de qualité de l'image, au format de nombre flottant. Il n'a qu'un seul bloc IP, qui décrit le modèle issu de l'apprentissage basé sur la logique floue (FL_Core).

Le Tableau 4.16 présente les rapports de la phase de synthèse, générés par les blocs IP du modèle, et par le bloc IP $ILBQA$ global prenant en considération des exécutions en parallèle et en série des blocs IP, lors de la phase d'implémentation HLS sous l'outil Vivado HLS version 2015.4 et mappé sur une carte Xilinx de la famille des Virtex 7 (VC707), pour une image de résolution 512×768. Ces rapports sont présentés suivants deux axes :

— **l'estimation des performances** : qui présente une estimation du nombre de cycles d'horloge nécessaires à l'exécution complète de chacun des blocs IP produits lors de la construction du modèle, généré pour une période d'horloge de $25\,ns$, soit une fréquence du signal d'horloge à $40\,Mhz$.

— **l'estimation des utilisations** : qui présente l'utilisation des ressources de la carte $FPGA$ pour chacun des blocs IP, en termes de blocs de mémoires (en Kbits), registres, portes logiques, DSP, bascules Flip-Flop et LUT.

4.5.2 Implémentation RTL du design global

Après la phase de construction des blocs IP en HLS, la prochaine étape consiste en l'exportation et l'intégration des différents blocs construits dans le design global sur l'outil Vivado de Xilinx. La Figure 4.13 présente le design global permettant l'implémen-

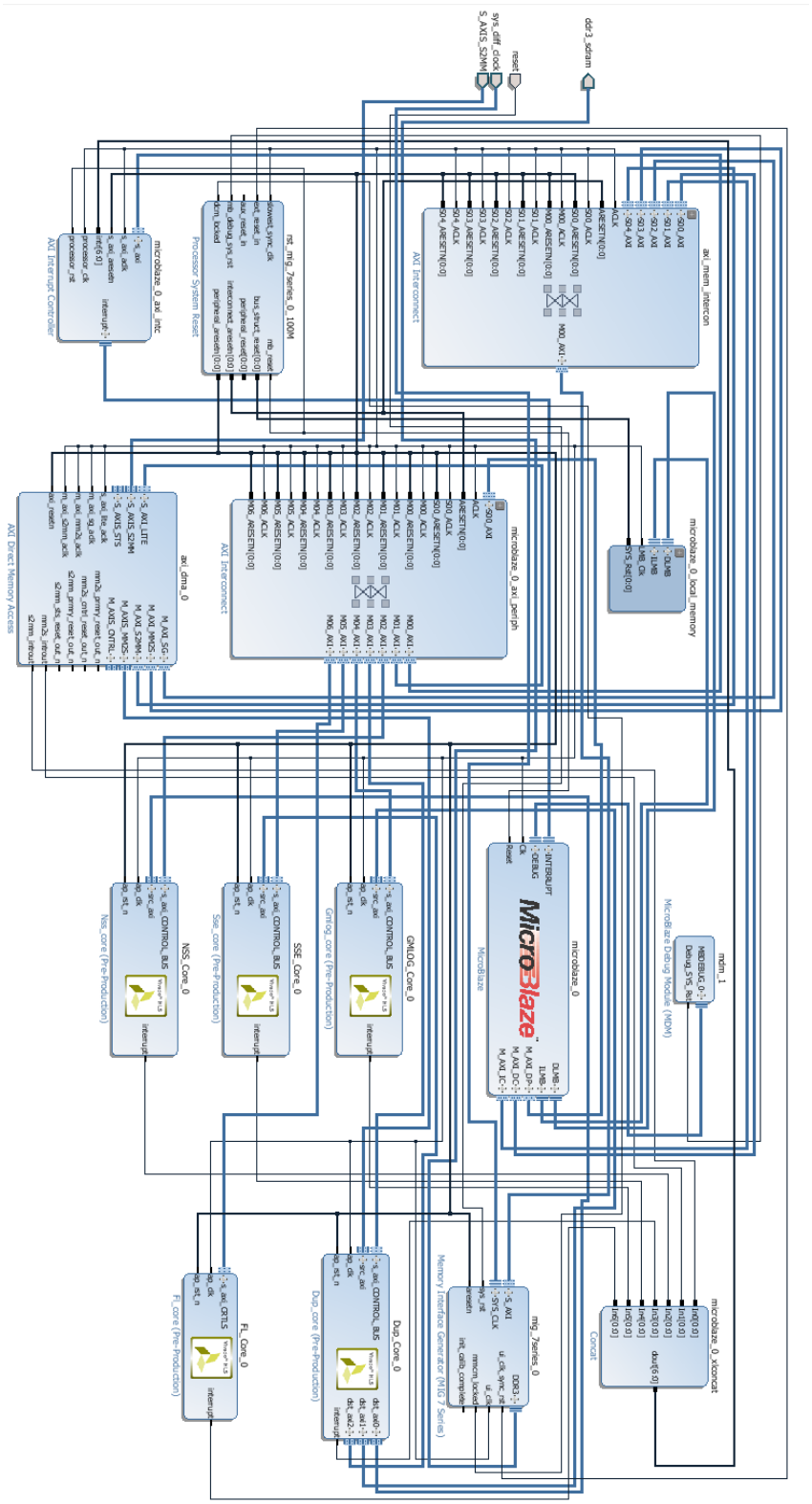


FIGURE 4.13 – Design global du ILBQA

TABLE 4.17 – Détail d'utilisation en phase de placement et routage (VC707) pour ILBQA

	Dispo.	**Utilisation**	**% Utilisation**
BRAM	2 060	2035	98.79 %
DSP	2 800	1 136	40.57 %
FF	607 200	167 801	27.63 %
LUT	303 600	298 678	98.38 %
LUTRAM	130 800	3 015	2.31 %
I/O	700	156	22.29 %
BUFG	32	5	15.63 %
MMCM	14	1	7.14 %
PLL	14	1	7.14 %

tation des modèles d'évaluation des images sans information sur l'image de référence. La Figure 4.14 présentant, quant à elle, la surface occupée par ce design global, a été synthétisée sur Vivado, lors de l'étape de placement et de routage du modèle obtenu, sur une carte $FPGA$ de la famille des Virtex 7 ($VC707$). Dans ce design global, la partie en jaune représente la surface occupée par les différents blocs IP produits lors de la phase de synthèse HLS (NSS_Core, GM-LOG_Core, SSE_Core, DT_Core et FL_Core), et permettant l'évaluation de la qualité des images, et la zone en bleu, celle occupée par le reste des composants du système global (les registres, le micro-contrôleur, les interconnexions, etc.).

Le détail d'utilisation des ressources ainsi que les pourcentages des ressources utilisées sur une carte $FPGA$ de la famille Virtex 7 ($VC707$) est donnée dans le Tableau 4.17 pour le système global. Il s'agit là des résultats pour une image de résolution 512×768.

4.5.3 Implémentation SDK et résultats

Le design global contenant les modèles mis au point ayant été implémenté sur une carte $FPGA$, l'étape suivante consiste en l'écriture d'un programme dans le micro-contrôleur (MicroBlaze). Ce programme permettra de récupérer les images au format « $AXI - Stream$ » grâce au bloc $AXI - DMA$, transmis sur la carte $FPGA$ à travers une extension utilisant le protocole $HDMI$. On pourrait alors évaluer la qualité de l'image ainsi reçue grâce au modèle implémenté sur la carte $FPGA$, pour enfin de retourner le score objectif donné par le modèle réalisé. Ce processus est réalisé grâce à l'outil SDK de Xilinx, et le programme est écrit en langage C/C++, et il est transmis au microcontrôleur via l'interface $JTAG$ de la carte $FPGA$.

Le processus d'évaluation d'une image sur la carte $FPGA$ est le suivant : l'image à

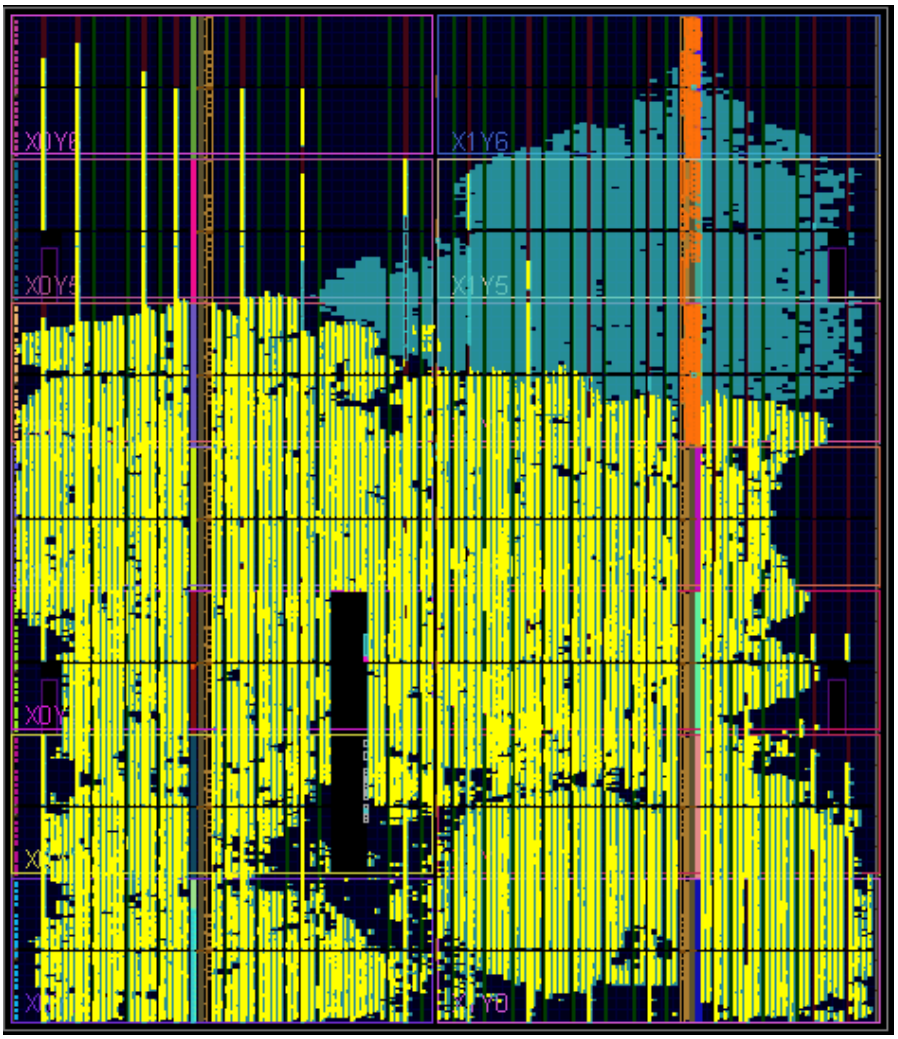


FIGURE 4.14 – Utilisation des ressources en phase de placement et routage ($VC707$) pour $ILBQA$

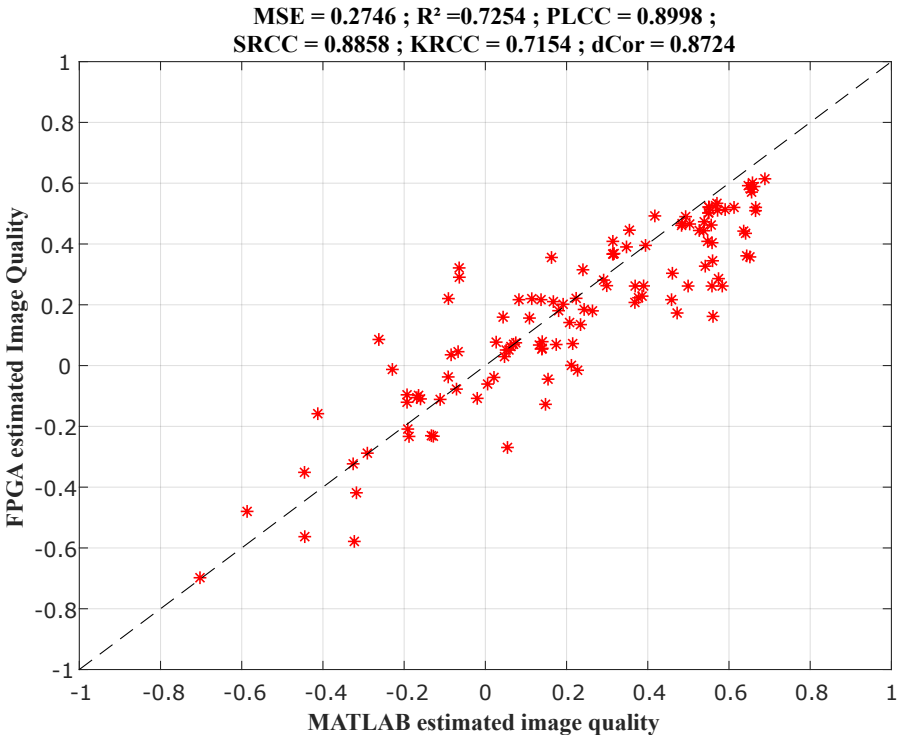


FIGURE 4.15 – Résultats simulation MATLAB Vs Implémentation FPGA

évaluer est transmise grâce au protocole « $AXI-Stream$ » via une interface $HDMI$ au $FPGA$. Les informations sur la taille de l'image sont transmises via l'interface $JTAG$ et grâce à l'unité $UART$ de communication avec l'ordinateur, présente dans le design global. Les valeurs stockées dans la mémoire sont prises par le microcontrôleur, et transmises au bloc IP global d'évaluation de la qualité des images ILBQA_Core, via les périphériques de communication AXI. Le bloc IP du modèle récupère en entrée l'image à évaluer ainsi que ses caractéristiques, évalue sa qualité grâce aux cinq blocs IP, tel que vu dans la sous-section sur l'implémentation HLS. Ensuite, il transmet le score de qualité d'image obtenu en sortie dans la mémoire RAM DDR3, via les périphériques de communication AXI. Ce score stocké dans la mémoire est transmis à l'ordinateur via l'interface $JTAG$ et grâce à l'unité $UART$. Les autres opérations de traitement et d'affichage des résultats sont effectuées au niveau de l'outil SDK.

120 images déformées par 24 types de distorsions, issues de la base de données d'images $TID2013$ ont été évaluées par nos modèles, sur un système $FPGA$; et les résultats d'implémentation sous $FPGA$ ont été comparés aux résultats de simulation sous $MATLAB$ et aussi aux scores objectifs pris dans la base de données d'images $TID2013$.

La Figure 4.15 présente la comparaison entre les résultats d'implémentation sous $FPGA$ et les résultats de simulation sous $MATLAB$ pour le modèle $ILBQA$. Et la Figure 4.16, quant à elle, présente les comparaisons entre les résultats de simulation $MATLAB$ (en bleu) et d'implémentation $FPGA$ (en rouge) sur l'axe des ordonnées,

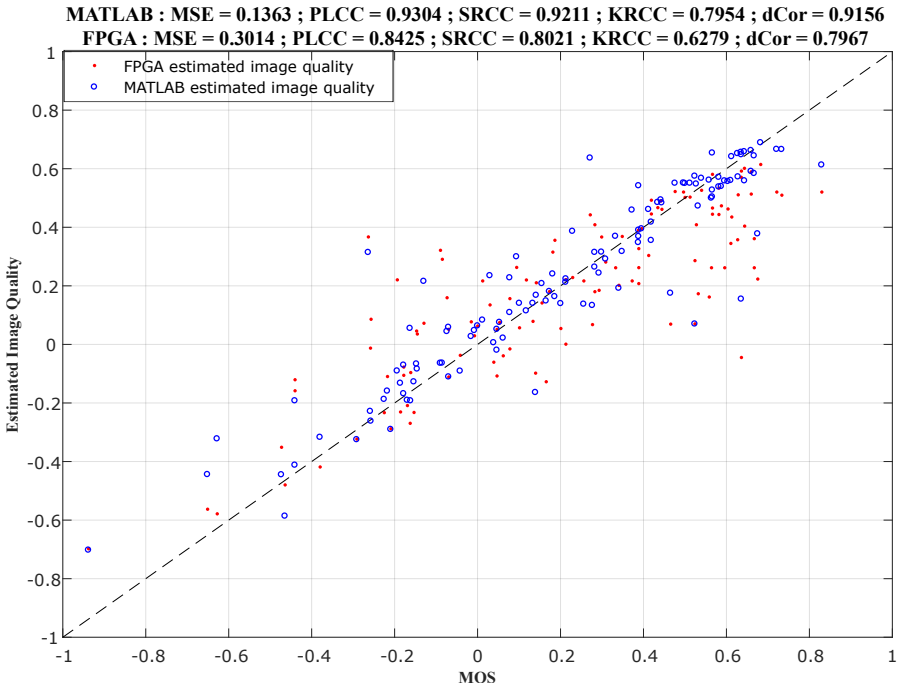


FIGURE 4.16 – MOS Vs résultats simulation MATLAB et implémentation FPGA

et les scores subjectifs issus de la base de données d'images $TID2013$, sur l'axe des abscisses ; pour le modèle $ILBQA$.

La différence entre les scores de qualité d'images estimés sous $MATLAB$ et ceux estimés sur $FPGA$ est causée par plusieurs facteurs, entre autres :

1. la diffusion des erreurs liées à la troncature en nombre flottant sur 32 bits dans le système $FPGA$;
2. la différence entre les résultats des fonctions de bases des outils $MATLAB$ et Vivado HLS. Par exemple les fonctions de conversion en niveau de gris, ou encore les fonctions de redimensionnement grâce à l'interpolation bicubique donnent des résultats légèrement différents sur $MATLAB$ et en HLS, et ces petites erreurs sont propagées tout au long des évaluations, et peuvent en fin de compte causer les différences entre les scores estimés présentes dans les Figures 4.15 et 4.16.

4.6 Bilan et apports

Le but de ce chapitre était de présenter la construction de modèles objectifs d'évaluation de la qualité des images, sans référence et sans information sur les distorsions présentes dans ces images. Le processus de construction de ces modèles d'évaluation a été mis en œuvre en quatre principales étapes, qui sont : l'extraction des caractéristiques ; la suppression des caractéristiques aberrantes et superflues ; la construction des métriques

intermédiaires grâce aux méthodes d'apprentissage automatique ; et la construction du modèle final d'évaluation de la qualité des images à partir des métriques intermédiaires en utilisant les méthodes d'apprentissage automatique. Deux principaux processus d'apprentissage des modèles ont été étudiés : le premier basé sur un apprentissage direct à partir des caractéristiques extraites des images et sélectionnées lors de la phase de suppression des caractéristiques superflues et de sélection des caractéristiques ; et puis le second basé sur un apprentissage indirect utilisant la notion de métriques intermédiaires pour réduire la complexité des calculs et des modèles produits.

L'extraction des caractéristiques se fait suivant quatre (04) axes, qui sont : les statistiques de scènes naturelles dans le domaine spatial ; l'amplitude du gradient ; le laplacien de gaussienne ; et les entropies spatiales et fréquentielles locales. La suppression des caractéristiques superflues et la sélection des caractéristiques à utiliser se fait sur la base de la corrélation mutuelle entre les différentes caractéristiques. Elle permet d'éviter le « sur-apprentissage » et de complexifier inutilement les modèles d'évaluation de la qualité des images produits. La construction des métriques intermédiaires se fait à partir des caractéristiques sélectionnées après suppression des caractéristiques superflues. Ces caractéristiques sélectionnées sont regroupées en fonction de leur axes d'extraction en classes. Chacune des classes ainsi construites est utilisée avec des méthodes d'apprentissage automatique, pour bâtir un modèle d'évaluation d'une métrique intermédiaire. Enfin, la construction du modèle d'évaluation de la qualité des images se fait soit à partir de toutes les caractéristiques sélectionnées, soit à partir des métriques intermédiaires, suivant le processus d'apprentissage adopté.

Les méthodes d'apprentissage automatique utilisées pour la construction des modèles sont des méthodes de régression : les réseaux de neurones artificiels, la régression non-linéaire polynomiale, les machines à vecteurs supports, les arbres de décision, et la logique floue. Les modèles d'évaluation sélectionnés sont construits à partir de la méthode des arbres de décision pour le processus direct, et pour le processus indirect, à partir des arbres de décision pour les métriques intermédiaires et la logique floue en ce qui concerne le modèle final d'évaluation de la qualité des images. Trois bases de données d'images ont été utilisées pour la construction et les tests des modèles, il s'agit des bases de données d'images $TID2013$, $LIVE$ et $CSIQ$. Les résultats obtenus par les modèles réalisés ont été comparés à ceux obtenus par d'autres index sans référence d'évaluation de la qualité des images présents dans la littérature. Les résultats de simulation sous $MATLAB$ ont démontré que les modèles élaborés donnent de meilleures performances que les index existants, comparées aux scores subjectifs pris dans les différentes bases de données.

Les modèles d'évaluation des images sans référence construits dans ce chapitre ont été implémentés sur une carte $FPGA$ de la famille des Virtex 7 ($VC707$). Les résultats obtenus dénotent une bonne corrélation avec les résultats de simulation sous l'outil $MATLAB$ et avec les scores subjectifs.

Les apports de cette étude dans le domaine de l'évaluation objective de la qualité des images sans référence et sans information sur les distorsions présentes dans les images, peuvent se résumer en quatre idées principales :

1. La mise en place d'un processus d'évaluation de la qualité des images sans réfé-

rences, suivant plusieurs axes indépendants d'extraction des caractéristiques. Cette mise en commun de différents axes d'extraction des caractéristiques permet de couvrir un plus grand espace d'exploration de l'image et de ses caractéristiques. Cette combinaison de plusieurs axes d'extraction des caractéristiques des images a permis de produire des modèles d'évaluation très corrélés avec la perception visuelle, et produisant des performances supérieures à celles des modèles présents dans la littérature, avec une complexité en temps faible.

2. La suppression des caractéristiques aberrantes et superflues, basée sur la corrélation mutuelle entre l'ensemble des caractéristiques extraites. Ce qui a permis de construire des modèles pas trop complexes et d'éviter des problèmes de « surapprentissage ».
3. L'utilisation de deux processus (direct et indirect) d'apprentissage des modèles. Où le processus direct consiste en un unique apprentissage à partir de toutes les caractéristiques sélectionnées. Ensuite, le processus indirect qui requiert deux apprentissages successifs : le premier pour la construction des métriques intermédiaires à partir des classes de caractéristiques extraites ; et le second utilisant les métriques intermédiaires pour la construction du modèle final d'évaluation de la qualité des images.
4. L'implémentation des modèles obtenus sur une carte $FPGA$, ce qui a permis d'étudier la complexité en temps et en espace des modèles produits. Les résultats de cette implémentation ont montré que les modèles nouvellement construits pour l'évaluation de la qualité des images sans référence peuvent être implémentés dans des circuits à faibles capacités. Ils peuvent de ce fait permettre une évaluation temps-réel de la qualité des images.

Au-delà des modèles construits à partir des méthodes automatiques d'apprentissage, l'évaluation objective de la qualité des images peut aussi se faire à partir des méthodes et algorithmes plus directs. Le chapitre suivant consiste à la mise sur pied de certains de ces algorithmes. Il s'agit ici de la mise en œuvre d'un ensemble d'algorithmes de détection et de correction de pixels défectueux dans les images.

Chapitre 5

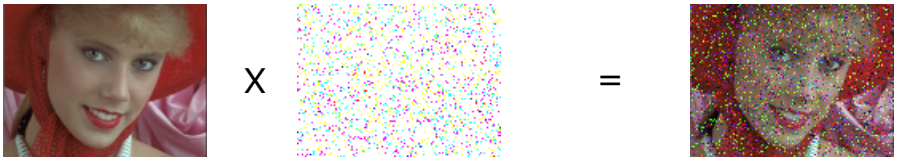
Détection et Correction de Pixels Défectueux

5.1 Introduction

Les caméras numériques sont de plus en plus utilisées dans la vie de tous les jours. La qualité des images prises par ces caméras numériques, ainsi que leurs prix dépendent des capteurs d'images utilisés. Les deux grandes familles de capteurs d'image sont, les capteurs CCD et les capteurs $CMOS$. Ces capteurs diffèrent en fonction de la taille des pixels, du nombre de pixels dans le capteur, de leur capacité, des espaces de stockages, mais aussi du nombre de pixels défectueux dans le capteur et sur les images produites [128].

Avec l'augmentation permanente de la taille des images prises, et de la densité des pixels dans une image, le nombre de pixels défectueux dans les capteurs d'images augmentent aussi [129], impactant en conséquence sur la qualité des images et par-là même sur le prix des caméras. Pour des capteurs ayant des caractéristiques techniques identiques, la qualité des images produites dépend en grande partie du taux de pixels défectueux dans les capteurs d'images. Pour une caméra en fonctionnement, le taux de pixels défectueux change avec le temps et le vieillissement du capteur, affectant ainsi la qualité des images produites au cours du temps par cette caméra [130]. Dès lors, le taux de pixels défectueux est un facteur important dans l'évaluation de la qualité d'un capteur d'image ou même d'une image tout simplement. Le taux de pixels défectueux peut aussi servir pour le contrôle du vieillissement des capteurs d'images et des caméras numériques.

Un pixel défectueux peut se définir comme un pixel dont la valeur diffère significativement, par rapport à un certain seuil prédéfini, de la valeur espérée en prenant en compte l'environnement et le voisinage dudit pixel. Un pixel défectueux ne donne aucune information importante, ou encore donne une information erronée par rapport à son environnement [130, 131, 132]. Il n'existe pas de définition standard d'un pixel défectueux, cependant de manière usuelle, on peut déterminer trois types de pixels défectueux, d'après la norme ISO $13406-2$ sur la tolérance des pixels défectueux sur les écrans LCD



(a) Image originale (b) Matrice de distorsions (c) Image déformée

FIGURE 5.1 – Exemple d'une image déformée par une matrice de distorsions

[19] :

— « **Type 1** » ou encore pixel mort bloqué au « **niveau bas** » : il s'agit d'un pixel qui ne reçoit, ni ne retourne aucun signal. Ce qui a pour effet une présence permanente d'un pixel totalement « noir » (valeur toujours à zéro) au même endroit quelle que soit l'image prise.

— « **Type 2** » ou encore pixel mort bloqué au « **niveau haut** » : il s'agit d'un pixel qui renvoie toujours la valeur maximale. Ce qui a pour effet la présence permanente d'un pixel totalement « blanc » (valeur du pixel à 255 pour une image en niveau de gris sur 8 bits).

— « **Type 3** » : il s'agit des défauts infectant les sous-pixels, composantes RGB : rouge, verte ou bleue du pixel, ou encore certains bits dans un pixel. Ce type de défaut se caractérise par une dégradation du pixel en fonction de la composante ou du bit infecté.

Les différents types de pixels défectueux seront regroupés ici en deux grands groupes de défauts : des défauts dits « **catastrophiques** », regroupant les défauts de « type 1 » et de « type 2 », consistant en des erreurs aux valeurs extrêmes (0 ou 255 pour des images en niveau de gris sur 8 bits) ; et des défauts dits « **paramétriques** », désignant les défauts de « types 3 », consistant en une augmentation ou en une atténuation de la valeur du pixel sans toutefois la bloquer aux valeurs extrêmes. La Figure 5.1 présente une image déformée à partir d'une matrice de distorsion contenant les trois types de défauts.

La détection de pixels défectueux (DPD) est un processus de test consistant en la détermination des pixels défectueux dans une image ou dans un capteur d'images. Les défauts présents dans les capteurs sont causés par des imperfections lors de la fabrication ; mais ils peuvent aussi apparaître avec le vieillissement du capteur lors de son utilisation au fil du temps. Ces défauts peuvent encore être causés par des altérations environnementales, mais aussi de altérations électriques ou optiques. Il existe plusieurs méthodes de détection de pixels défectueux dans un capteur d'images. Ces méthodes peuvent être regroupées en deux grands groupes, comme présentés dans la Figure 5.2 :

1. Les méthodes industrielles : les méthodes de DPD industrielles sont, pour la plupart, basées sur des tests électriques, de luminances, et de points noirs. Ces tests sont effectués en industrie après la fabrication du capteur d'images. Cependant, ils sont couteux et nécessite un environnement particulier pour être mis en œuvre. Ils ne sont pas temps-réel et sont plutôt difficiles à appliquer dès lors que le capteur est sorti de l'usine de fabrication.

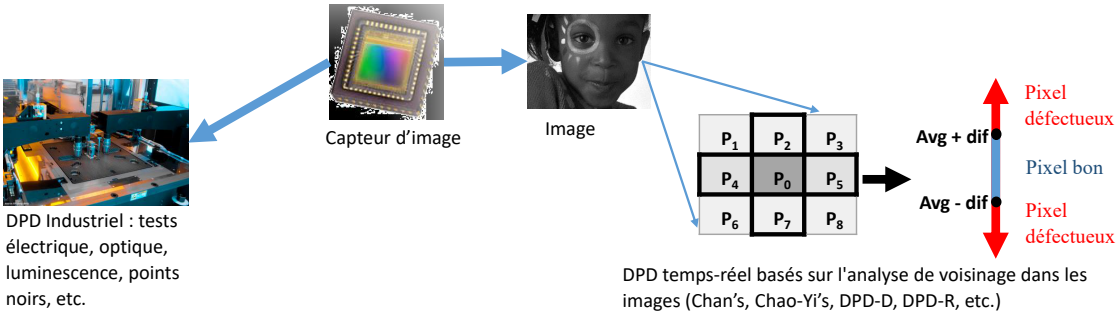


FIGURE 5.2 – Méthodes de détection des pixels défectueux

2. Les méthodes de tests en fonctionnement : ces méthodes peuvent être appliquées sur des capteurs même lorsqu'ils sont déjà mis en vente. Cette méthode de test peut se diviser en deux sous-groupes : les tests en ligne, et les tests hors ligne. Pour les tests hors ligne, des images spécifiques (toutes blanches, toutes noires, en damier, etc.) sont transmises au capteur pour vérifier les sorties sur un écran, ce qui permet de détecter les pixels défectueux. Cette méthode nécessite un arrêt du fonctionnement de la caméra pour effectuer les tests. Les tests en ligne, quant à eux, sont effectués sur des caméras fonctionnant dans leur environnement normal. Ils consistent à effectuer des tests sur un certain nombre d'images prises par le capteur en utilisation. Ces tests en ligne peuvent donc être faits en temps réel et sans perturber le fonctionnement du capteur.

Dans la littérature, on peut retrouver plusieurs algorithmes de détection de pixels défectueux. Par exemple, dans [133], l'algorithme présenté par les auteurs permet la détection des pixels défectueux en comparant la valeur du pixel à tester à la valeur de prédiction du pixel évaluer à partir des pixels de même couleur se trouvant sur la même ligne que ce pixel à tester. Dans [134], les auteurs proposent d'utiliser une série d'images, permettant de suivre l'historique de dégradation des pixels défectueux en analysant leur comporte au cours du temps. Dudas et al. quant à eux, introduisent dans [135, 129], des modèles basés sur des probabilités d'apparition des erreurs, qui pour une densité de 0.5% de pixels défectueux, utilisent 50 images ordinaires pour détecter la totalité des défauts dans le capteur, avec un taux très faible de faux positifs. Cet algorithme a pour inconvénients de n'être pas temps réel, et de demander beaucoup d'images avant de confirmer si un pixel est effectivement défectueux. Chan dans [18]et Cho et al. dans [136, 137] introduisent des méthodes simples et temps-réels de détection de pixels morts, se basant uniquement sur des opérations arithmétiques de bases sur les valeurs des pixels au voisinage du pixel à tester. Ces méthodes ont pour inconvénients d'avoir un fort taux de faux positifs, ce qui augmente inutilement le temps de correction, et dégrade la qualité de l'image corrigée. Aussi toutes ces méthodes ne s'appliquent-elles qu'aux défauts catastrophiques, constitués des valeurs extrêmes des pixels, et pas aux défauts paramétriques.

Dans ce chapitre, sont proposés un ensemble d'algorithmes en ligne et temps-réel de

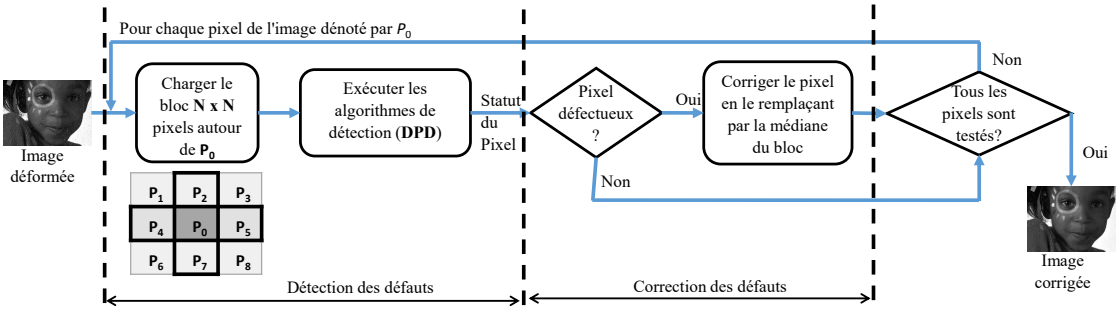


FIGURE 5.3 – Processus de détection et correction des pixels défectueux

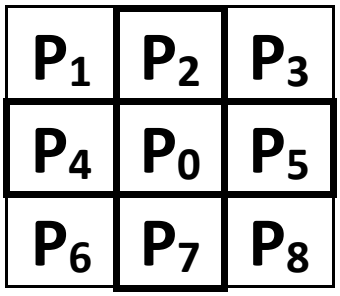


FIGURE 5.4 – Bloc de 3×3 pixels

détection et de correction de pixels défectueux dans les images et capteurs d'images, utilisant pour la plupart des opérations arithmétiques de base, sur les pixels voisins au pixel à tester. Ces algorithmes suivent le processus de base décrit dans la Figure 5.3. Dans la majeure partie de ces algorithmes, seront utilisés les pixels voisins formant avec le pixel à tester (P_0) une matrice de dimension 3×3 comme présenté dans la Figure 5.4. Les algorithmes proposés se basent sur les similitudes entre les pixels d'un même environnement, et partent sur le principe que dans une image naturelle, la dégradation ou encore les changements de couleurs se font de manière progressive. Ces algorithmes utilisent les éléments tels que : les distances entre le pixel central et ses voisins, la moyenne, l'écart-type, l'étendue, l'écart interquartile dans le voisinage, et la notion de détection d'un pixel hypothétiquement défectueux dans un environnement. Les modèles produits ont été comparés aux modèles existant en termes de performances et de temps d'exécution, sur 144 images déformées (24 images de référence $\times$ 6 types de distorsions) en couleur et en niveau de gris. Certains de ces algorithmes ont aussi été implémentés sur un système $FPGA$, pour confirmer les résultats obtenus lors de la phase de simulation, et évaluer les temps d'exécution et l'espace utilisé sur des systèmes physiques.

La suite du chapitre présente dans la Section 5.2 un ensemble de méthodes existantes de détection de pixels défectueux dans une image. Dans la Section 5.3 seront décrites les différentes méthodes proposées pour la détection et la correction des pixels défectueux dans les images, avec leurs avantages et leurs inconvénients. La Section 5.4 présente les résultats expérimentaux obtenus par avec les différents algorithmes, ainsi que leur

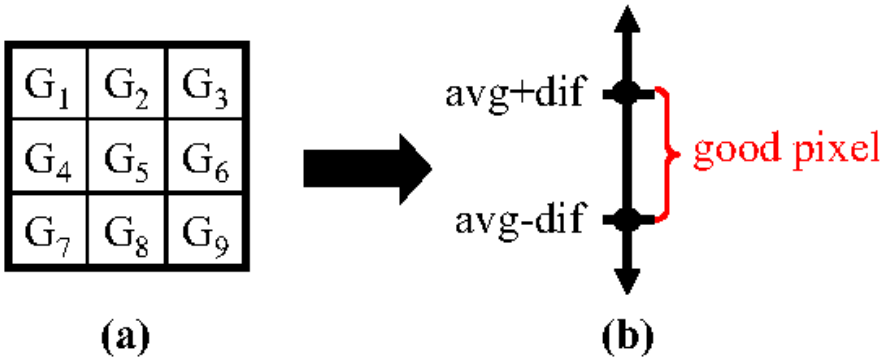


FIGURE 5.5 – Méthode de DPD proposée par Chan

comparaison entre eux voire avec les résultats obtenus par les algorithmes existants. Dans la Section 5.5 sont présentés les architectures et les résultats obtenus lors de la phase d'implémentation des différents algorithmes sur une carte $FPGA$. Pour terminer, la Section 5.6 fait un récapitulatif du chapitre, en mettant en exergue les apports et avancées dans le domaine de la détection des pixels défectueux.

5.2 Méthodes existantes de détection de pixels morts

Il existe dans la littérature un ensemble de méthodes de détection de pixels défectueux. Dans ces méthodes, on prend comme entrée une image, que l'on parcourt pixel par pixel en testant chacun des pixels en fonction de son environnement. On décèle alors l'ensemble des pixels défectueux contenus dans cette image. Dans la suite seront examinés, tour à tour, les principales méthodes en usage et notamment celle présentée par Chan [18] et celle présentée par Cho et al. [137, 136] portant sur la détection des pixels morts dans une image.

5.2.1 Méthode de détection de pixels morts de Chan

Dans l'algorithme de détection de pixels morts proposé par Chan [18], le pixel à tester (G_5) et ses pixels adjacents utilisés pour l'étude forment un bloc de 3×3 pixels en niveau de gris, comme présentés dans la Figure 5.5(a). Le processus de détection des pixels morts présenté par Chan peut se simplifier tel qu'illustré dans la Figure 5.5. Ce processus consiste alors en l'exécution des étapes suivantes pour chacun des pixels de l'image :

1. sélectionner le bloc de 3×3 pixels en niveau de gris formés par le pixel à tester et ses pixels voisins ;
2. trouver les deux pixels de références de la méthode : G_H, le pixel ayant la seconde plus grande valeur du bloc, et G_L, le pixel ayant la seconde plus petite valeur du bloc de pixels ;

3. évaluer la valeur de la différence entre les deux pixels de référence, comme présenté dans (5.1) ;

$$dif = G_H - G_L \tag{5.1}$$

4. évaluer la valeur moyenne des pixels du bloc, en excluant le pixel à tester et les deux pixels de références G_H et G_L, comme présenté dans (5.2) ;

$$avg = \frac{\sum_{i=1}^{9} G_i - (G_5 + G_H + G_L)}{6} \tag{5.2}$$

5. la somme entre la valeur de différence évaluée à l'étape 3 (dif) et la valeur moyenne évaluée à l'étape 4 (avg) donne le seuil haut (pour les défauts de « type 2 »), et la différence entre ces deux valeurs donne le seuil bas (pour les défauts de « type 1 »). Tous les pixels se trouvant entre le seuil haut et le seuil bas sont déclarés « bons pixels », tandis que les pixels dont la valeur est supérieure au « seuil haut » ou inférieure au « seuil bas » sont déclarés comme des « pixels morts » ; tel que le montre le processus présenté dans la Figure 5.5(b).

Cette méthode de détection de pixels morts proposée par Chan a pour avantages d'être intuitive, et de ne nécessiter que des opérations arithmétiques très simples. Cependant le problème de cette méthode est que le taux de détection des pixels morts est réduit si les blocs de 3×3 pixels contiennent plus d'un certain nombre de pixels défectueux. Cela pourrait influencer les deux pixels de références (G_H et G_L), ce qui aurait pour conséquence d'augmenter la valeur de la différence entre ces pixels de références. L'autre fâcheuse conséquence est d'augmenter intervalle de détection de pixels défectueux et par la même, d'empêcher une détection minutieuse de ceux-ci. Un autre inconvénient majeur de cette méthode proposée par Chan est le grand nombre de faux positifs (pixels « bons » déclarés comme « morts »).

5.2.2 Méthode de détection de pixels défectueux de Cho

Contrairement à la l'algorithme proposée par Chan, l'algorithme de détection de pixels morts proposé par Cho et al. [137, 136] utilise uniquement quatre des pixels voisins au pixel à tester. Ceux-ci formant une croix avec le pixel central, comme présenté dans la Figure 5.6 (a). Cet algorithme proposé par Cho et al. utilise le principe suivant lequel une valeur anormale du pixel à tester affecterait la réponse globale du voisinage comme présenté dans la Figure 5.6 (b). Le processus de détection des pixels défectueux proposé par Cho et al. exécute les étapes suivantes, pour chacun des pixels à tester :

1. sélectionner le bloc de pixel formant une croix, avec au centre, le pixel à tester ;
2. tout pixel se trouvant dans l'intervalle allant de 25% à 75% de la valeur maximale du bloc des pixels voisins sélectionnés dans l'étape précédente, est considéré comme « bon pixel », tandis que les autres sont des pixels à tester, activant de ce fait, les étapes suivantes ;

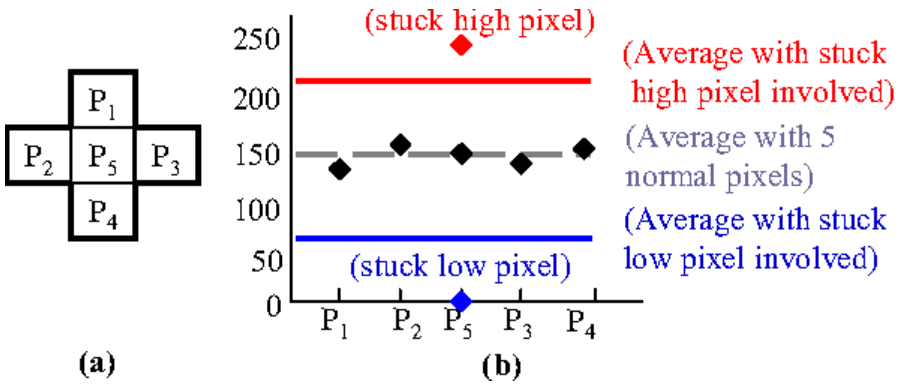


FIGURE 5.6 – Méthode de DPD proposée par Cho et al.

3. évaluer la valeur estimée (est) du pixel à tester, en effectuant la moyenne de ses deux pixels adjacents sur l'axe horizontal, comme présenté dans (5.3) ;

$$est = \frac{P_2 + P_3}{2} \tag{5.3}$$

4. évaluer la différence (dif) entre la valeur du pixel et sa valeur estimée, calculée à l'étape précédente. Plus cette différence est grande, plus la probabilité que le pixel testé soit défectueux est grande. Cette différence est calculée comme présenté dans (5.4) ;

$$dif = |est - P_5| \tag{5.4}$$

5. évaluer la valeur moyenne (avg) entre le pixel central à tester, et ses pixels adjacents, comme présenté dans (5.5). Le calcul de cette moyenne prend non seulement en compte le pixel à tester, mais aussi augmente son poids dans l'équation, en le multipliant par la somme des poids de ses pixels adjacents. Cela a pour effet de lier fortement cette valeur moyenne à la valeur du pixel à tester, par rapport aux valeurs de ses pixels voisins ;

$$avg = \frac{\sum_{i=1}^{4} P_i + 4 \times P_5}{8} \tag{5.5}$$

6. la valeur moyenne (avg) évaluée à l'étape précédente est utilisée comme « valeur seuil » pour la différence (dif) entre la valeur du pixel à tester et sa valeur estimée. Si $dif > avg$ ou $dif > 255 - avg$, alors le pixel à tester P_5 est déclaré « pixel mort ».

Les avantages de cette méthode sont qu'elle est très simple et qu'elle ne nécessite que des opérations arithmétiques de bases pour être mise en œuvre. Elle permet entre autre de réduire le nombre de faux positifs par rapport aux résultats obtenus avec la méthode proposée par Chan. Elle accroît le taux de détection de pixels défectueux, même lorsqu'il existe plusieurs pixels défectueux parmi les pixels adjacents au pixel à tester. Cependant cette méthode ne prend pas en compte tous les axes du voisinage d'un pixel, ce qui a

pour conséquence d'augmenter le nombre de faux positifs pour des zones présentant des bordures dans l'image. De plus, cette méthode ne permet que de détecter que des défauts catastrophiques, c'est-à-dire ceux dont les valeurs sont soit proches de 0 soit de 255 pour une image en niveau de gris sur 8 bits.

Pour résoudre les problèmes causés par ces méthodes proposées par Chan et Cho et al., les propositions apparaissant dans la section suivante peuvent être suggérées. Elles incluent un ensemble d'algorithmes de détection de pixels défectueux dans l'image se basant sur plusieurs axes tels que : la distance entre le pixel à tester et ces pixels voisins ; la médiane des pixels voisins ; l'analyse des variances ; l'analyse des directions des pixels voisins au pixel à tester.

5.3 Méthodes proposées

Différentes méthodes ont été mise en œuvre pour détecter les pixels défectueux dans une image ou un capteur d'images. Celle qui ont été retenues pour ce travail s'appréhendent suivant deux grandes approches : il s'agit de l'approche utilisant des algorithmes basés sur les distances entre le pixel à tester et ses pixels voisins ; et de l'approche s'appesantissant sur les étendues et les dispersions dans les zones de l'image.

5.3.1 Détection de pixels défectueux basée sur les distances

Contrairement aux méthodes existantes, la méthode de détection de pixel défectueux proposée dans cette partie prend en compte les distances entre le pixel central à tester et ces pixels voisins. La méthode ainsi proposée consiste à déterminer la valeur estimer (V_{est}) du pixel à tester en fonction de son environnement, constitué de ses pixels voisins sur N bits. Il s'agit par la suite de comparer cette valeur estimée à la valeur réelle du pixel à tester en fonction d'un certain seuil défini. La Figure 5.7 présente les distances entre le pixel central et ses pixels voisins pour une matrice de taille $N \times N$ autour du pixel central qui est le pixel à tester, avec $N = 7$. Ces distances sont calculées à partir de (5.6), où x et y sont les coordonnées du pixel voisin en fonction du pixel central de la matrice.

$$d_{xy} = \sqrt{x^2 + y^2} \tag{5.6}$$

La valeur estimée du pixel à tester est alors calculée comme la moyenne des pixels voisins P_i pondérés par leur distance d_i avec le pixel central, telle que présentée dans (5.7).

$$V_{est} = \frac{\sum_i P_i/d_i}{\sum_i 1/d_i} \tag{5.7}$$

Cette valeur estimée du pixel à tester (V_{est}) est utilisée pour évaluer l'écart avec la valeur réelle du pixel à tester. Il s'agit de la valeur de différence (V_{dif}) entre la valeur réelle (P_0) du pixel à tester, et sa valeur estimée, telle que présentée dans (5.8). Plus

...	...	...	...	...	...	...	...	...
...	$3\sqrt{2}$	$\sqrt{13}$	$\sqrt{10}$	3	$\sqrt{10}$	$\sqrt{13}$	$3\sqrt{2}$	...
...	$\sqrt{13}$	$2\sqrt{2}$	$\sqrt{5}$	2	$\sqrt{5}$	$2\sqrt{2}$	$\sqrt{13}$	...
...	$\sqrt{10}$	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	$\sqrt{10}$	...
...	3	2	1	0	1	2	3	...
...	$\sqrt{10}$	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	$\sqrt{10}$	...
...	$\sqrt{13}$	$2\sqrt{2}$	$\sqrt{5}$	2	$\sqrt{5}$	$2\sqrt{2}$	$\sqrt{13}$	...
...	$3\sqrt{2}$	$\sqrt{13}$	$\sqrt{10}$	3	$\sqrt{10}$	$\sqrt{13}$	$3\sqrt{2}$	...
...	...	...	...	...	...	...	...	...

FIGURE 5.7 – Matrice 7×7 des distances avec les pixels voisins

cette valeur de différence entre la valeur réelle et la valeur estimée du pixel à tester est grande, plus la la probabilité que le pixel à tester soit défectueux est, elle aussi, grande.

$$V_{dif} = |V_{est} - P_0| \tag{5.8}$$

Le seuil de comparaison de l'écart entre la valeur réelle et la valeur estimée du pixel à tester est évalué en fonction de la moyenne entre la valeur du pixel à tester et la moyenne des pixels de son voisinage. Cette moyenne (V_{avg}) est présentée dans (5.9) ; où m représente le nombre de pixels autour du pixel à tester (pour une matrice $N \times N$, $m = 2 * N - 1$). L'écart entre la valeur estimée et la valeur réelle du pixel à tester est comparé à l'écart entre la valeur moyenne comprenant le pixel à tester (V_{avg}), et les valeurs aberrantes possibles des pixels défectueux (dénoté ici V_a). Le pixel à tester P_0 est donc déclaré défectueux s'il remplit la condition présentée en (5.10) ; où dans le cas des erreurs catastrophiques, $V_a = 0$ ou $V_a = 255$.

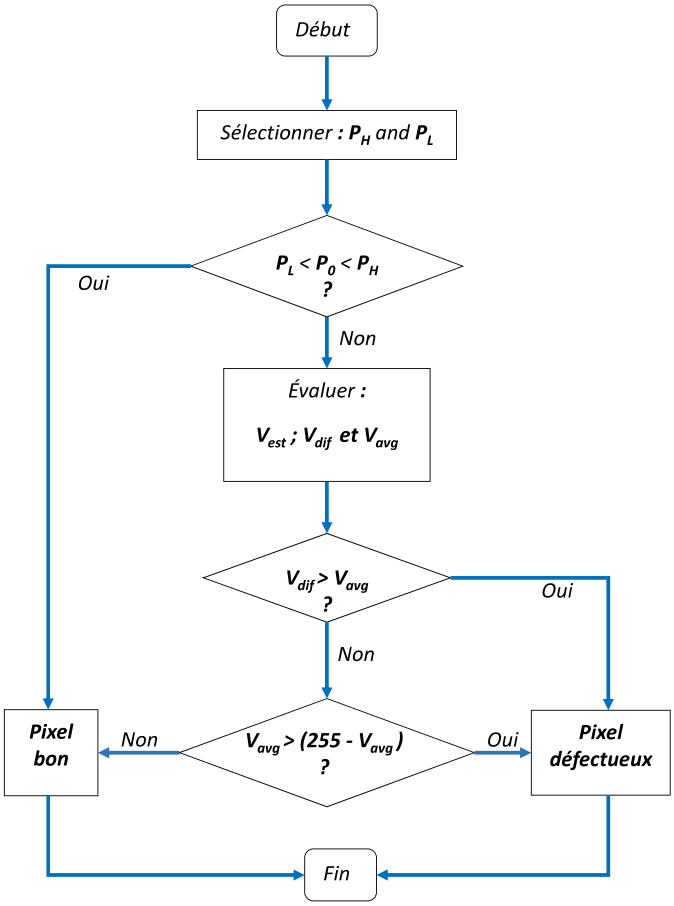
$$V_{avg} = \frac{1}{2}\left(\frac{1}{m}\sum_{i=1}^{m} P_i + P_0\right) \tag{5.9}$$

$$V_{dif} > |V_{avg} - V_a| \tag{5.10}$$

Le processus de détection de pixels défectueux dans une image consiste, pour chaque pixel de l'image, à sélectionner ses pixels voisins dans une matrice de taille $N \times N$, puis à exécuter les étapes présentées dans l'Algorithme 5.1. Le processus complet de détection de pixels défectueux dans une image par la méthode basée sur les distances est constitué des étapes suivantes, pour chacun des pixels de l'image :

1. sélectionner le bloc de $N \times N$ pixels autour du pixel à tester, tel que le pixel à tester soit le centre de la matrice (au point de distance 0, d'après la Figure 5.7) ;
2. trouver les valeurs de références P_H et P_L ; où P_H représente le pixel ayant la plus grande valeur dans le voisinage, et P_L le pixel ayant la plus faible valeur dans le voisinage du pixel à tester ;

Algorithme 5.1 Processus de détection d'un pixel défectueux basé sur les distances



3. si la valeur du pixel à tester se trouve entre les valeurs P_H et P_L($P_L < P_0 < P_H$), le pixel à tester est déclaré « bon pixel » ; sinon poursuivre à l'étape suivante ;
4. évaluer la valeur estimée du pixel à tester (V_{est}) à partir de l'équation (5.7), comme la moyenne des pixels du voisinage du pixel à tester, pondérés par la distance les séparant de ce pixel à tester ;
5. évaluer la valeur de différence (V_{dif}) à partir de l'équation (5.8), comme la valeur absolue de la différence entre la valeur réelle du pixel à tester, et sa valeur estimée ;
6. évaluer le paramètre de seuil (V_{avg}) à partir de l'équation (5.9), comme la moyenne entre le pixel à tester et la moyenne des pixels du voisinage de ce pixel à tester ;
7. cette moyenne V_{avg} est utilisée, avec les valeurs aberrantes possibles d'un pixel (V_a), pour trouver le seuil de la valeur de différence V_{dif}. Si la valeur de différence est supérieure au seuil évalué comme la valeur absolue de la différence entre la valeur moyenne V_{avg} et les valeurs aberrantes possibles V_a, comme présenté dans (5.10) ; alors le pixel à tester est déclaré « défectueux ». Pour les défauts catastrophiques $V_a = 0$ ou $V_a = 255$; et pour les défauts paramétriques, cette valeur dépend de l'environnement du pixel à tester, et est trouvée comme la valeur hypothétiquement défectueuse de cet environnement.

L'avantage majeur de cette méthode basée sur les distances entre le pixel à tester et son voisinage est la réduction du taux de faux positifs et l'augmentation du taux de détection dans les cas où il existe plusieurs pixels défectueux dans la même zone ou encore dans le voisinage des pixels à tester. Ceci est dû au nombre de pixels voisins et d'axes de détections pris en compte, mais aussi au facteur de distance réduisant l'impact des pixels très « éloignés » du pixel à tester. Cependant, cette méthode présente un inconvénient majeur qui est l'augmentation de la complexité des calculs, liés à l'évaluation des distances entre le pixel à tester et ses pixels voisins ; mais aussi cette méthode ne permet pas encore de détecter tous des défauts paramétriques.

Le taux de faux positifs diminue avec l'augmentation du nombre de pixels voisins considérés, mais la complexité du modèle augmente elle aussi avec l'augmentation du nombre de pixels voisins. Au vu de l'importance d'avoir une complexité réduite dans ce travail, le nombre de pixels voisins au pixel à tester a été fixé à 3. Ce qui implique d'avoir une matrice de taille 3×3 autour du pixel à tester, soit 8 pixels voisins, comme présentés dans la Figure 5.4.

En considérant que tous les pixels dans le voisinage du pixel à tester soient de « bons pixels », il importe de mettre sur pied une méthode simplifié de calcul du seuil, dans laquelle on suppose que la valeur estimée du pixel à tester V_{est} est sensiblement égale à la moyenne des pixels du voisinage du pixel à tester. Pour faire cette supposition, on néglige l'impact des distances sur le calcul de la valeur estimée du pixel a tester. On peut ainsi remplacer la moyenne des pixels dans l'évaluation du paramètre de seuil V_{avg} par la valeur estimée du pixel V_{est}. Les équations (5.9) et (5.10) peuvent donc être remplacées par les équations (5.11) et (5.12), respectivement, pour le calcul du seuil et de la condition de détection de pixel défectueux. Si on ne prend en considération que les défauts catastrophiques, le développement de la comparaison donnée en (5.12) permet

d'avoir une version simplifiée, comme présentée dans (5.13), sans calcul du paramètre de seuil V_{avg}, ce qui permet de réduire la complexité de l'algorithme de détection proposé, mais aussi le taux de détection.

$$V_{avgS} = \frac{V_{est} + P_0}{2} \tag{5.11}$$

$$|V_{est} - P_0| > \left| \frac{V_{est} + P_0}{2} - V_a \right| \tag{5.12}$$

$$\begin{cases} P_0 > \lambda \times V_{est} & or \quad P_0 < \frac{1}{\lambda} \times V_{est} \\ & or \\ P_0 < \lambda \times (V_{est} - a) & or \quad P_0 > \frac{1}{\lambda} \times V_{est} + a \end{cases} \tag{5.13}$$

où $\lambda = 3$ et $a = 255 \times 2/3 = 170$

Cette méthode simplifiée a pour avantage de d'amoindrir les calculs, par la suppression du calcul de la valeur de différence V_{dif} et du paramètre de seuil V_{avg} ; mais ces calculs restent toujours relativement lourds. L'inconvénient de cette méthode est qu'elle ne peut pas être utilisée pour la détection des défauts paramétriques, ou plutôt l'adaptation pour une détection des défauts paramétriques va augmenter la complexité de l'algorithme de détection.

Pour réduire la complexité des calculs effectués, et pour augmenter le taux de détection des défauts paramétriques, une seconde méthode basée sur l'analyse des écarts et des dispersion entre les pixels dans les zones de l'image a été implémentée.

5.3.2 Détection de pixels défectueux basée sur les dispersions

Cette seconde méthode se base sur les dispersions entre les pixels dans un même environnement de l'image. Ces dispersions peuvent être liées à la variation des écarts dans l'environnement, à la médiane, à l'étendue, ou encore aux écarts interquartiles des blocs de pixels. Cette méthode peut être scindée en trois principales étapes :

1. Diviser l'image à tester en blocs de $N1 \times N1$ pixels, et rechercher les blocs contenant potentiellement des pixels défectueux, en fonction de l'écart entre la valeur minimale et la valeur maximale dans le bloc, aussi appelé l'étendue du bloc, et de l'écart interquartile ou l'écart-type dans le bloc.
2. Dans chaque bloc déclaré comme contenant potentiellement des pixels défectueux, analyser les valeurs aux extrémités du bloc (valeur maximale et minimale), en fonction des valeurs d'autres pixels dans ce bloc, mais aussi analyser les pixels ayant ces valeurs extrêmes dans un nouvel environnement où ils seront au centre. On recréer de nouveaux blocs de taille $N2 \times N2$ autour des pixels potentiellement défectueux, pour mieux les analyser en fonction de leur voisinage, suivant plusieurs axes directionnels.
3. Si une valeur extrême est détectée comme « défectueuse », passer à la valeur suivante dans le bloc, et repartir à l'étape 2, jusqu'à ce que l'on trouve que les valeurs

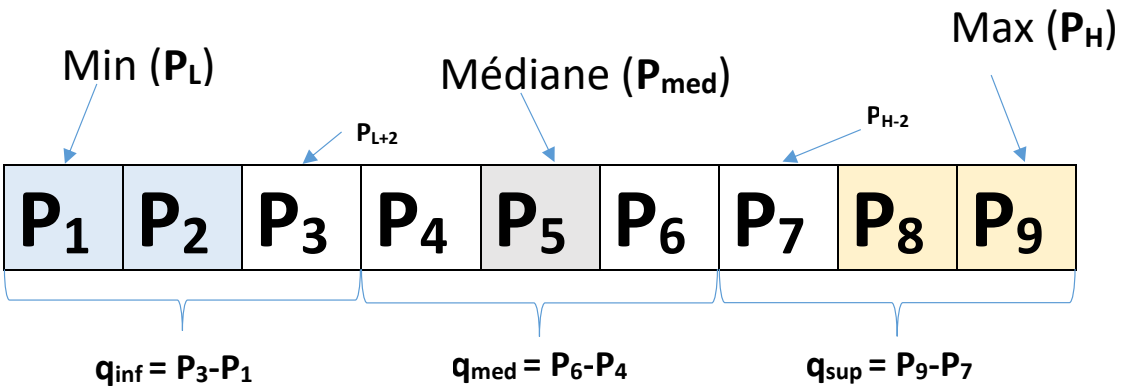


FIGURE 5.8 – Processus basé sur les dispersions, appliqué sur une matrice 3×3

extrêmes soient de toutes deux de bonnes valeurs de pixels dans leur environnement ; puis passer au bloc suivant.

Hypothèse1 : Dans cet algorithme, on considère que pour un bloc de taille $N1 \times N1$, la probabilité d'avoir plus de $2 \times k$ pixels défectueux (soit plus de k défauts de « Type 1 » ou de « Type 2 ») est négligeable. Où k dépend du nombre de pixels dans le bloc, soit $N1 \times N1$. Avec $N1 = 3$ on obtient une matrice de dimension 3×3, les valeurs sont triées et rangées dans un vecteur de taille 9 pixels tel que présenté dans la Figure 5.8, et $k = 2$. Dans cette hypothèse, il y a lieu de considérer que les pixels entre P_3 et P_7 dans le vecteur trié sont de bons pixels. Les pixels potentiellement défectueux sur lesquels les tests seront effectués sont les pixels aux extrémités du vecteur trié, soit P_1, P_2, P_8 et P_9.

Hypothèse2 : Ici l'on considère qu'étant donné le vecteur de la Figure 5.8, le pixel P_2 ne peut être défectueux que si le pixel P_1 a déjà été déclaré défectueux. Il en est de même pour le pixel P_8 qui ne pourra être défectueux que si le pixel P_9 a déjà été déclaré défectueux lui aussi. Cela est dû au fait que dans une zone de pixels, ceux qui sont potentiellement défectueux sont ceux qui ont des valeurs s'écartant de la normale des valeurs de l'environnement. Les pixels défectueux sont donc ceux qui ont les valeurs extrêmes dans l'environnement. Dans l'algorithme global de détection de pixels défectueux, un pixel ne sera testé que si le pixel avant lui (à son extrémité) a déjà été testé et déclaré « défectueux ». De même, si un pixel est déclaré « bon » dans un environnement ou bloc de pixels, les suivants dans le bloc trié jusqu'à la médiane sont tous déclarés « bons pixels ».

Hypothèse3 : étant donné la Figure 5.8 présentant un vecteur trié issu d'une matrice de 3×3 pixels, la méthode proposée basée sur les dispersions considère que si l'écart entre les trois pixels minimums (q_{inf}) est supérieur d'un certain seuil à définir de l'écart des pixels autour de valeur médiane du bloc (q_{med}), alors le pixel à l'extrémité inférieure (P_1) est déclaré comme hypothétiquement défectueux , $q_{inf} < p_3 \times q_{med} + p_4$, donc ainsi défini, $p_3 = 2$ et $p_4 = 0$. Et pareillement pour l'écart entre les trois pixels maximums (q_{sup}), le pixel à l'extrémité supérieure (P_9)

est déclaré comme hypothétiquement défectueux, si cet écart est supérieur d'un certain seuil à l'écart autour de la valeur médiane.

Le processus complet de détection des pixels défectueux dans une image par la méthode basée sur les dispersions dans un environnement est décrit dans l'Algorithme 5.2. Ce processus prend en entrée une image, et retourne un vecteur contenant les indices des pixels ayant été déclaré défectueux ; il est constitué des étapes suivantes :

1. diviser l'image en blocs de $N1 \times N1$.
2. Pour chaque bloc (noté bl), évaluer son étendue dans « R », comme présentée dans (5.14), où P_H représente le pixel ayant la plus grande valeur dans le bloc, et P_L le pixel ayant la plus faible valeur dans le bloc de pixels à tester.

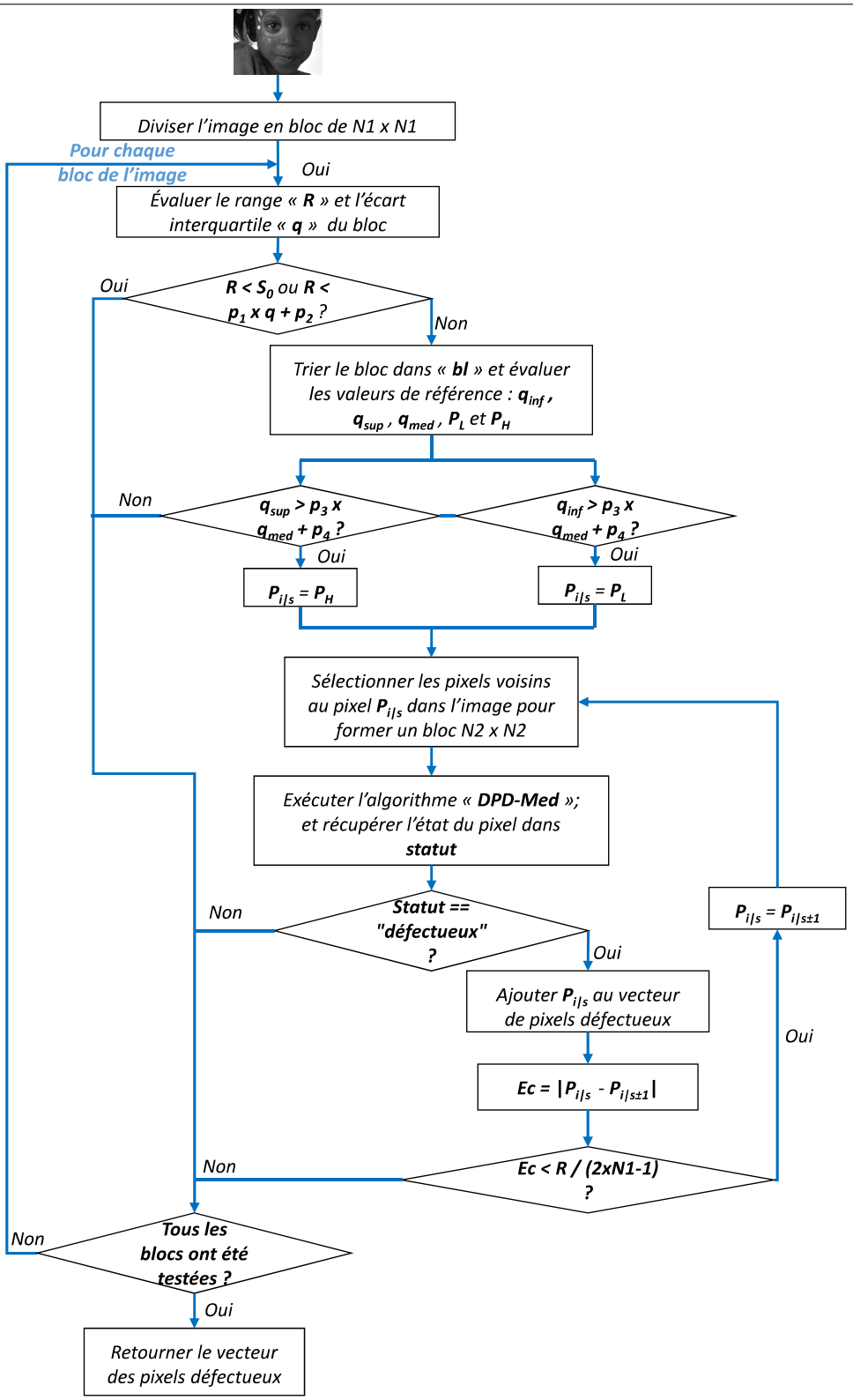
$$R = P_H - P_L \tag{5.14}$$

3. Si cette étendue du bloc est inférieure à un certain seuil $S_0 (R < S_0)$ ou inférieur d'un certain seuil à l'écart interquartile du bloc $(R < p_1 \times iqr(bl) + p_2)$, déclarer tous les pixels de ce bloc comme de « bons pixels » et passer au bloc suivant ; sinon, passer à l'étape suivante. S_0 est défini subjectivement comme le seuil au-delà duquel les différences entre pixels commencent à être visibles, et impacte sur la dégradation de la qualité visuelle de l'image ; $p_1$et p_2 sont deux valeurs définies empiriquement, et permettant de gérer le rapport entre l'étendue et l'écart interquartile pour la détection des valeurs aberrantes dans un ensemble de valeur ; dans la littérature, $p_1 = 1.5$ et $p_2 = 0$).
4. Trier le bloc bl, et évaluer les écarts de références : q_{inf} qui est l'écart entre la valeur minimale du bloc (P_L), et la $k - ième$ valeur après cette valeur minimale dans le bloc, ceci permet d'avoir approximativement la valeur de l'écart entre la valeur minimale et le premier quartile. q_{sup} qui est l'écart entre la valeur maximale du bloc (P_H), et la $k - ième$ valeur avant cette valeur maximale dans le bloc, ceci permet d'avoir approximativement la valeur de l'écart entre la valeur maximale et le troisième quartile. Et q_{med} est l'écart entre les deux valeurs autour de la médiane du bloc (noté P_{med}), telles que ces valeurs soient séparées par k valeurs dans le bloc trié, ceci peut être considéré comme une simplification de l'écart interquartile. Ces écarts sont calculés comme présentés dans (5.15), et peuvent se justifier par « l'Hypothèse 1 » sur le nombre de défauts dans un bloc de taille $N1 \times N1$.

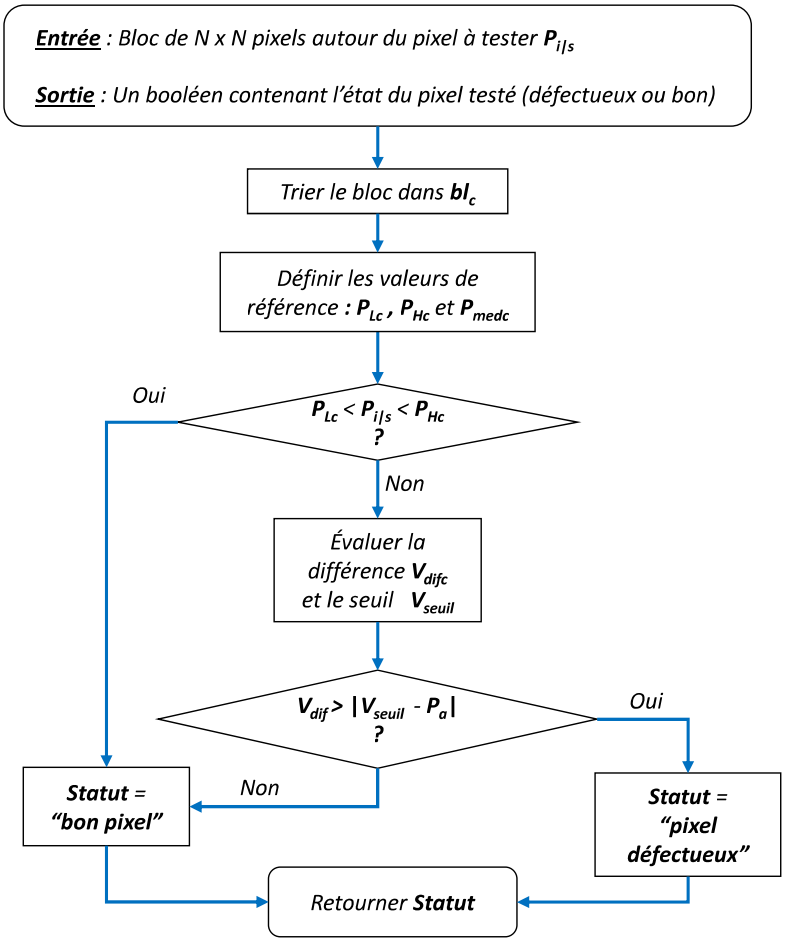
$$\begin{cases} q_{inf} = P_{L+k} - P_L \\ q_{sup} = P_H - P_{H-k} \\ q_{med} = P_{med+k/2} - P_{med-k/2} \end{cases} \tag{5.15}$$

5. Si l'écart de référence de la valeur minimale (ou maximale), q_{inf} (ou q_{sup}) est supérieur d'un certain seuil à la valeur de l'écart médian (q_{med}) : $q_{inf} > p_3 \times q_{med} + p_4$ (ou $q_{sup} > p_3 \times q_{med} + p_4$) ; alors le pixel avec la valeur minimale (ou maximale) dans le bloc de pixels est considéré comme hypothétiquement défectueux. Dans ce cas, il importe d'exécuter l'étape suivante pour vérifier s'il est définitivement

Algorithme 5.2 Processus de détection d'un pixel défectueux basé sur les dispersions



Algorithme 5.3 Processus de détection d'un pixel défectueux basé sur la médiane



défectueux en le considérant dans un nouvel environnement dans lequel il est le centre. Sinon, ces pixels ainsi que tous ceux qui n'ont pas été testés dans le bloc sont déclarés « bons », et on passe au bloc suivant.

6. Lorsqu'un pixel a été déclaré hypothétiquement défectueux, il déclenche cette étape qui exécute un algorithme de détection de pixels défectueux en reconstruisant un nouveau bloc de taille $N2 \times N2$ autour du pixel hypothétiquement défectueux, dénoté ici par $P_{i|s}$, puis exécute les étapes de la méthode ***« DPD_Med »*** décrit dans l'algorithme 5.3.
7. Si ***« DPD_Med »*** retourne que le pixel est défectueux, on teste si le pixel qui lui est adjacent dans le vecteur trié (P_{L+1} si on est dans la bordure inférieure du vecteur, ou P_{H-1} si on est dans la bordure supérieure) est lui aussi un pixel hypothétiquement défectueux. Cela consiste à calculer tout d'abord l'écart entre le pixel défectueux et son pixel adjacent, comme présenté dans (5.16), puis si cet écart est inférieur à un certain seuil (l'étendue du bloc divisé par le nombre d'élément dans le bloc), défini comme dans (5.17), alors le nouveau pixel à tester est déclaré hypothétiquement défectueux, et on retourne à l'étape 6 ; sinon tous les pixels restant du bloc allant du pixel adjacent testé, au pixel à la valeur médiane sont déclarés « bons ».

$$Ec = \left| P_{i|s} - P_{i|s\pm1} \right| \tag{5.16}$$

$$Ec < R/(2 \times N1 - 1) \tag{5.17}$$

Étant donné un bloc de taille $N2 \times N2$ autour du pixel à tester, dénoté ici par $P_{i|s}$, les étapes suivantes décrivent le processus de détection du pixel défectueux, basé sur la médiane du bloc, appelé ***« DPD_Med »*** présenté dans l'Algorithme 5.3 :

1. Trier le bloc dans bl_c par ordre croissant, en excluant le pixel à tester ($P_{i|s}$).
2. Définir de nouvelles valeurs de références P_{Lc}, P_{Hc} et P_{medc}, comme étant respectivement les valeurs : minimale, maximale et médiane du nouveau bloc. On définit aussi P_a comme étant les valeurs aberrantes possibles du pixel à tester.
3. Si le pixel est bien compris dans son environnement ($P_{Lc} < P_{i|s} < P_{Hc}$), le pixel est déclaré « bon », sinon passer à l'étape suivante.
4. Évaluer la différence entre le pixel à tester et la valeur médiane du bloc autour de lui, comme présentée dans (5.18). Calculer aussi la valeur de seuil, définie ici comme étant la valeur absolue de la différence entre moyenne entre le pixel à tester et la valeur médiane, et les valeurs aberrantes possibles du pixel à tester, comme présentée dans (5.19). Puis, effectuer une comparaison entre cette valeur de différence et le seuil évalué.

$$V_{difc} = \left| P_{i|s} - P_{medc} \right| \tag{5.18}$$

$$V_{seuil} = \left| \frac{P_{i|s} + P_{medc}}{2} - P_a \right| \tag{5.19}$$

5. si la valeur de différence V_{difc} est supérieure au seuil V_{seuil}, le pixel est déclaré « défectueux » ; sinon il est déclaré « bon ».

5.3.3 Correction basée sur les médianes

Une fois les pixels défectueux détectés, il existe plusieurs méthodes de correction simple de ces pixels dans la littérature, qui permettent de cacher le défaut sur l'image en sortie. Dans la méthode de détection de pixels morts proposée par Chan [18], l'auteur propose de remplacer le pixel déclaré mort par la moyenne de ses huit pixels voisins. Dans la méthode proposée par Cho et al. [137, 136], les auteurs proposent de remplacer le pixel mort par la moyenne des valeurs des pixels formant une croix autour du pixel déclaré comme défectueux par la méthode, comme dans la Figure 5.6(a).

Ici, la proposition retenue est une technique de correction des pixels déclarés défectueux basée sur un filtre médian. Elle consiste à remplacer le pixel déclaré défectueux par la médiane de ces valeurs voisines. D'où le choix de ce travail d'utiliser la médiane du bloc de 3×3 pixels centré autour du pixel déclaré défectueux. Par cette méthode, les calculs sont simplifiés. Étant donné que le bloc de pixels a déjà été trié pour la détection, il suffit de remplacer la valeur du pixel défectueux par celle du pixel à la cinquième position dans le vecteur trié contenant l'ensemble des valeurs de pixels du bloc.

5.4 Résultats expérimentaux

24 images de références de résolution 384×512, prises parmi les images de références de la base de données d'images $TID2013$ [4, 27] sont utilisées pour construire les images déformées qui permettront de tester non seulement les différentes méthodes de détection et de correction de pixels défectueux construites dans ce chapitre, mais aussi celles présentes dans la littérature. Ces images ont été choisies à cause de leurs diversités et de la variation des caractéristiques qu'elles présentent (faces, personnages, animaux, scènes naturelles, gros plans, texte, etc.). Ces images de références ont été infectées par six types de distorsions liées aux pixels défectueux, que l'on retrouve communément dans les images ou les capteurs d'images. Cela a permis de produire les images déformées utilisées pour l'évaluation des différentes méthodes de détection de pixels défectueux.

5.4.1 Types de défauts et résultats des tests

Habituellement, les pixels défectueux sont distribués de manière tout à fait aléatoire dans une image ; mais, il peut arriver que ceux-ci soient regroupés en un même endroit de l'image pour former des blocs, encore appelés « cluster », de pixels défectueux. Parfois, des défauts sur les convertisseurs analogiques-numériques peuvent causer des distorsions

sur toute une colonne de l'image. Ainsi, pour couvrir toutes ces différentes possibilités, six (06) types de défauts ont été insérés dans les images :

- 0.5% aléatoire : les pixels défectueux sont créés aléatoirement (type, niveau, emplacement), et ils occupent 0.5% de la taille totale de l'image (ou encore du nombre de pixels total dans l'image) ;
- 1% aléatoire : les pixels défectueux sont créés et placés aléatoirement dans l'image, et le nombre de pixels défectueux est égale à 1% du nombre total de pixels dans l'image ;
- 100 sets de blocs 2×2 : les pixels défectueux sont générés en blocs de 2×2, et ces blocs sont placés aléatoirement dans l'image. Étant donné que l'influence d'un seul bloc ne permettrait pas de tester la qualité des différentes méthodes, 100 blocs de 2×2 ont été insérés dans chacune des images de référence, et ces 100 blocs sont placés aléatoirement.
- 100 sets de blocs 3×3 : ces distorsions sont similaires à ceux des blocs de 2×2, mais ici, les blocs sont de de taille 3×3 ; cette fois aussi 100 blocs de 3×3 sont créés et placés aléatoirement dans l'image.
- $1 - \textit{colonne}$: les pixels défectueux sont placés sur une colonne entière de l'image ; ce type de distorsion dans une image peut être causé par un défaut dans l'un des convertisseurs analogique-numérique présent au niveau du capteur d'images. La colonne sur laquelle le défaut apparaît est sélectionnée aléatoirement, ainsi que les distorsions apparaissant sur cette colonne.
- $2 - \textit{colonnes}$: cette distorsion est un cas particulier de la distorsion précédente ($1 - \textit{colonne}$), elles sont toutes les deux causées de manières similaires, mais dans ce cas particulier, 2 colonnes côte-à-côte sont infectées.

Chacun de ces types de défauts a été appliqué à 24 images de références de la base de données d'images TID2013. Ce qui a permis de produire les 144 images déformées, soient 24 images de références $\times 6$ types de défauts, utilisées pour évaluer les différentes méthodes présentées dans les sections précédentes. Les résultats des tests effectués sur ces méthodes sont présentés comme suit :

- Vrai Positif (VP) : correctement identifier un pixel défectueux. Il s'agit du nombre de pixels défectueux dans une image, effectivement détectés comme pixels défectueux par une méthode de détection.
- Vrai Négatif (VN) : correctement identifier un bon pixel. Il s'agit du nombre de bons pixels dans une image, effectivement détectés comme bons pixels par une méthode de détection.
- Faux Positif (FP) : incorrectement identifier un bon pixel comme pixel défectueux. Il s'agit du nombre de bons pixels dans une image, détectés comme pixels défectueux par une méthode de détection.
- Faux Négatif (FN) : incorrectement identifier un pixel défectueux comme bon pixel. Il s'agit du nombre de pixels défectueux dans une image, détectés comme bons pixels par une méthode de détection.

Ces différents résultats des tests ont été produits pour chacune des méthodes de détection de pixels défectueux présentée dans ce chapitre, et pour chacune des 144 images

TABLE 5.1 – Sensibilité pour différentes méthodes

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets 2×2**	**sets 3×3**	**Total**
Chan	97.63 %	97.56 %	62.63 %	14.49 %	36.28 %	23.01 %	55.27 %
Cho	99.80 %	99.75 %	91.16 %	84.05 %	87.49 %	82.77 %	90.84 %
DPD_D.	**99.97 %**	**99.92 %**	**99.69 %**	**91.00 %**	96.48 %	**92.77 %**	**96.64 %**
DPD_R.	99.54 %	99.54 %	98.53 %	57.71 %	**99.12 %**	68.82 %	87.21 %

déformées.

5.4.2 Indices de performances des méthodes

Plusieurs indices de performance sont utilisés pour évaluer ces différentes méthodes de détection de pixels défectueux. Entre autres, on pourrait citer : la « sensibilité », la « spécificité », les valeurs prédictives positive et négative, et le « coefficient Phi ». La sensibilité et la spécificité sont des indices importants de la mesure de la précision des différentes méthodes de détection de pixels défectueux dans une image. Mais ces deux indices ne permettent pas d'estimer la probabilité individuelle d'un pixel d'être défectueux ou bon. Les valeurs prédictives positive (VPP) et négative (VPN) sont utilisées pour combler ce manquement, en définissant pour un pixel donné, sa probabilité d'être un pixel défectueux (VPP), ou un bon pixel (VPN). Le coefficient-phi quant à lui combine ces deux types d'indices, en donnant à la fois la précision de la méthode sur l'ensemble des pixels et sur les pixels pris individuellement.

5.4.2.1 Sensibilité

La sensibilité mesure la proportion des valeurs positives correctement identifiées comme telles [138] ; en d'autres termes, la sensibilité d'une méthode de détection de pixels défectueux est la proportion des pixels défectueux qui sont effectivement détectés comme tels par cette méthode. La sensibilité (Se) est évaluée comme, le nombre de pixels défectueux détectés par la méthode et qui sont réellement des pixels défectueux (les vrais positifs), divisé par le nombre total de pixels défectueux dans l'image (vrais positifs + faux négatifs). L'équation (5.20) présente le calcul de la sensibilité d'une méthode pour une image, étant donné les différents résultats des tests présentés dans la sous-section précédente.

$$Se = \frac{VP}{VP + FN} \tag{5.20}$$

Il est à noter que la sensibilité est évaluée uniquement avec la proportion des pixels défectueux qui sont effectivement détectés comme tels par la méthode ; tous les calculs sont donc basés sur les pixels défectueux. Cela implique que la sensibilité détermine à

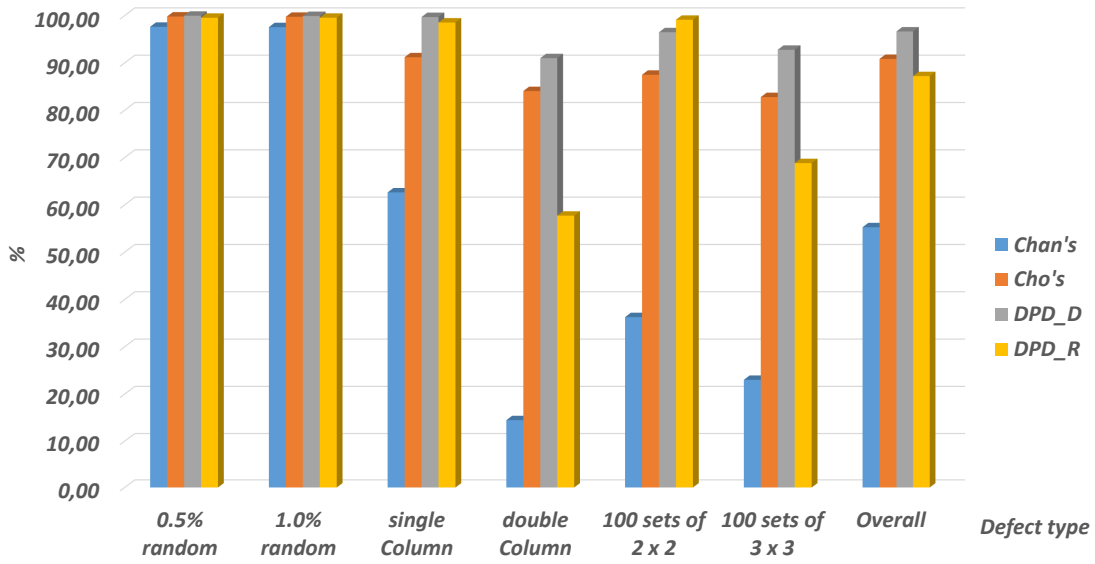


FIGURE 5.9 – Sensibilité pour différentes méthodes avec 6 types de défauts

quel point la méthode permet de détecter les pixels défectueux en ne considérant que l'ensemble des pixels défectueux présents dans l'image.

La Figure 5.9 et le Tableau 5.1 présentent la moyenne des sensibilités, testées sur toutes les 144 images déformées présentées précédemment, regroupées en fonction des six (06) types de défauts, sur les quatre (04) méthodes présentées dans les sections précédentes : méthodes de Chan, de Cho, et les deux proposées, basées sur les distances entre le pixel à tester et ces pixels voisins, et sur les dispersions entre les valeurs des pixels dans les différentes zones de l'image.

Dans cette Figure 5.9 et ce Tableau 5.1, la première remarque basée sur la valeur totale de sensibilité, est que la méthode proposée (DPD_D) basée sur les distances entre le pixel central à tester et ses pixels voisins produit de meilleurs résultats que les autres méthodes étudiées ; cela implique que cette méthode « DPD_D » permet de détecter correctement un plus grand nombre de pixels défectueux dans les images que les autres méthodes. Une seconde remarque est que les défauts consistant à des pixels défectueux « isolés » dans un environnement tel que ceux insérés par les défauts « 0.5% ou 1% aléatoires » sont plus facilement détectable que les défauts consistant en des groupes de pixels défectueux regroupés en un même endroit de l'image, tels que les « blocs de 3×3 », ou encore « $2 - colonnes$ ».

5.4.2.2 Spécificité

La spécificité mesure la proportion des valeurs négatives correctement identifiées comme telles [138]. Dans ce cas de figure relatif à la détection des pixels défectueux dans une

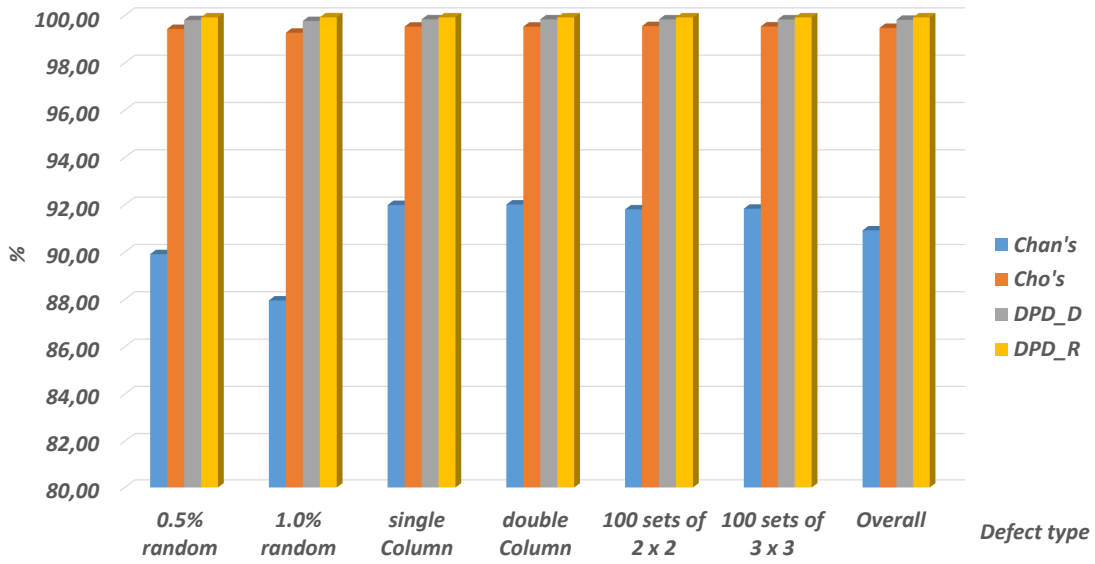


FIGURE 5.10 – Spécificité pour différentes méthodes avec 6 types de défauts

TABLE 5.2 – Spécificité pour différentes méthodes

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets 2×2**	**sets 3×3**	**Total**
Chan	89.92 %	87.96 %	91.99 %	92.01 %	91.81 %	91.84 %	90.92 %
Cho	99.44 %	99.27 %	99.53 %	99.53 %	99.56 %	99.54 %	99.48 %
DPD_D.	99.81 %	99.77 %	99.84 %	99.84 %	99.84 %	99.84 %	99.82 %
DPD_R.	**99.92 %**	**99.92 %**	**99.92 %**	**99.92 %**	**99.92 %**	**99.92 %**	**99.92 %**

image, la spécificité d'une méthode se définit comme la proportion des bons pixels qui sont effectivement détectés comme tels par une méthode. Cette spécificité est alors calculée comme le rapport entre le nombre de bons pixels correctement identifiés comme tels par la méthode (vrai négatif), sur le nombre total de bons pixels au sein de l'image, (vrais négatifs + faux positifs). L'équation (5.21) présente le calcul de la spécificité (Sp) d'une méthode pour une image donnée.

$$Sp = \frac{VN}{VN + FP} \tag{5.21}$$

Ici aussi, il est à noter que la spécificité est évaluée uniquement avec la proportion des bons pixels, qui sont effectivement détectés comme « bons » pixels par la méthode étudiée. Tous les calculs sont donc basés uniquement sur les bons pixels dans l'image et sur la quantité de « bons pixels » détectés par une méthode. La spécificité permet alors

de déterminer à quel point une méthode est bonne à détecter les bons pixels dans une image, en ne considérant que l'ensemble des bons pixels de cette image.

La Figure 5.10 et le Tableau 5.2 présentent les valeurs de spécificités moyennes pour les quatre méthodes étudiées dans ce chapitre, et testée sur les 144 images déformées, regroupées en fonction des six (06) types de défauts. Il découle de ce tableau et de cette figure que les méthodes proposées (DPD_D et DPD_R) basées sur les distances et sur les dispersions, produisent les meilleurs résultats en termes de spécificité, donc de détection de bons pixels dans les images déformées. Cependant les résultats de cette spécificité sont légèrement biaisés par la grande taille des images, et le fort taux de bons pixels dans les images, d'où les valeurs très proches de 100% pour toutes les méthodes étudiées.

Les valeurs de sensibilité et de spécificité étant des indices de mesure de précision des méthodes, et ne donnant aucune information sur la probabilité individuelle d'un pixel à être bon ou défectueux, on été étudiées les valeurs prédictives qui sont des indices permettant de répondre à cette question de probabilité individuelle, ou encore de taux de bonnes détections parmi l'ensemble des détections.

5.4.2.3 Valeurs prédictives

On distingue deux types de valeurs prédictives : la valeur prédictive positive, portant sur le taux de détection dans les pixels défectueux dans ce cas d'étude, et la valeur prédictive négative, portant sur le taux de détection des bons pixels.

La **valeur prédictive positive** est la proportion des valeurs effectivement positives parmi l'ensemble des valeurs détectées comme positives par une méthode [138, 139]. Pour ce qui est de la détection des pixels défectueux dans une image, la valeur prédictive positive représente la proportion de pixels effectivement défectueux, parmi l'ensemble des pixels défectueux détectés par une méthode. En d'autres termes, la proportion de vrai positif parmi l'ensemble des valeurs détectées comme positives. La valeur prédictive positive (VPP) calculée comme dans (5.22), est le rapport du nombre de pixels défectueux effectivement détectés comme tels par une méthode (vrais positifs), sur le nombre total de pixels détectés comme « défectueux » par cette méthode (vrais positifs + faux positifs).

$$VPP = \frac{VP}{VP + FP} \tag{5.22}$$

La valeur prédictive positive permet de décrire les performances d'une méthode de diagnostic, en donnant le taux de vrai positif parmi l'ensemble des valeurs détectées comme positives. Cette valeur (VPP) s'interprète simplement comme étant la probabilité qu'une valeur positive soit effectivement positive. En d'autres termes, il s'agit de la probabilité qu'un pixel qui a été détecté comme pixel défectueux soit effectivement un pixel défectueux. Il s'agit donc de la proportion de pixels défectueux dans l'ensemble des pixels détectés comme tels par une méthode.

La Figure 5.11 et le Tableau 5.3 présentent les valeurs prédictives positives moyennes, pour chacune des 144 images déformées, regroupées par types de défauts, pour les quatre

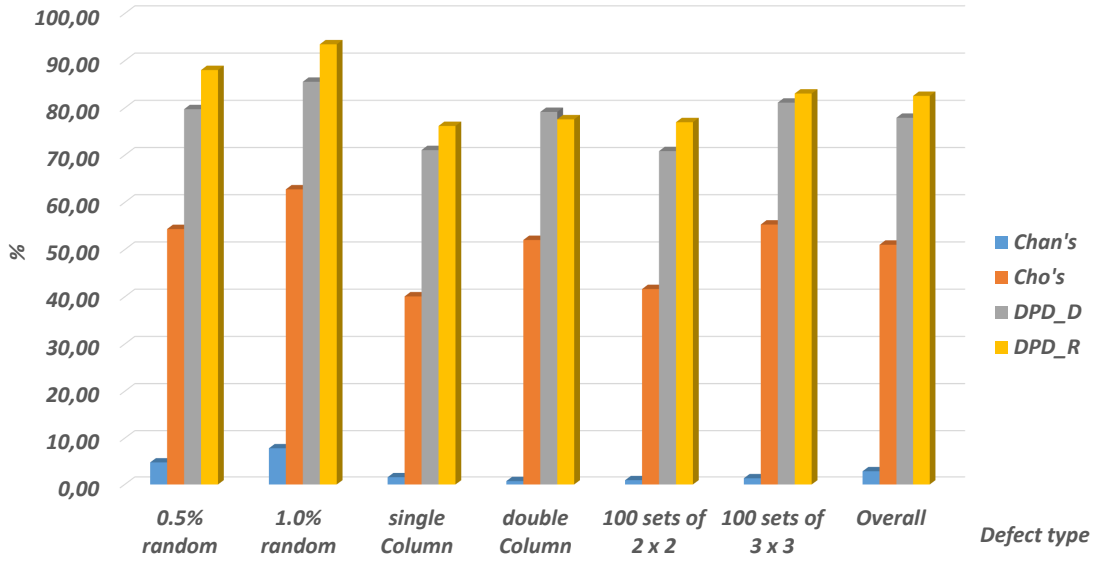


FIGURE 5.11 – Valeurs prédictives positives pour différentes méthodes avec 6 types de défauts

TABLE 5.3 – Valeurs prédictives positives pour différentes méthodes

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets 2×2**	**sets 3×3**	**Total**
Chan	4.76 %	7.82 %	1.57 %	0.73 %	0.92 %	1.33 %	2.85 %
Cho	54.36 %	62.76 %	40.09 %	52.04 %	41.65 %	55.31 %	51.04 %
DPD_D.	79.74 %	85.55 %	71.08 %	79.16 %	70.86 %	81.13 %	77.92 %
DPD_R.	**88.03 %**	**93.48 %**	**76.19 %**	**77.59 %**	**76.99 %**	**83.05 %**	**82.56 %**

méthodes de détection de pixels défectueux présentées dans ce chapitre. Cette figure et ce tableau montrent que les méthodes proposées dans ce chapitre sont meilleures que les méthodes existantes dans la littérature. D'où la conclusion que les méthodes proposées (DPD_D et DPD_R) produisent nettement moins de « faux positifs » que les méthodes existantes (proposées par Chan, et par Cho et al.). On peut aussi remarquer que les valeurs prédictives positives pour les défauts regroupant plusieurs pixels défectueux dans une même zone tels que les défauts « $2 - colonnes$ » et « blocs de 3×3 », sont moins fortes que celles des défauts répartissant les pixels défectueux dans toute l'image. Cela est dû, en grande partie, au taux de détection réduit pour les défauts regroupant les pixels défectueux dans une même zone.

La **valeur prédictive négative** quant à elle est la proportion des valeurs effectivement négatives parmi l'ensemble des valeurs détectées comme telles par une méthode [139, 138]. Dans le présent cas d'étude sur la détection des pixels défectueux dans une image, la valeur prédictive négative donne la proportion de pixels effectivement « bons », parmi l'ensemble des pixels détectés comme tels par une méthode. La valeur prédictive négative (VPN) calculée comme dans (5.23), est le rapport entre le nombre de bons pixels effectivement détectés comme « bons » par une méthode (vrais négatifs), sur le nombre total de pixels détectés comme « bons » par cette méthode (vrais négatifs + faux négatifs).

$$VPN = \frac{VN}{VN + FN} \tag{5.23}$$

La valeur prédictive négative s'interprète comme étant la probabilité qu'un pixel détecté « bon » soit effectivement un bon pixel ; il s'agit donc du taux de bons pixels parmi l'ensemble des pixels détectés comme tels par une méthode. Cependant étant donné la résolution des images considérées ici, et le taux de pixels potentiellement défectueux très faible, cette valeur (VPN) est biaisée par le taux de pixels bons très élevé dans les images. Ainsi, cette valeur prédictive négative est toujours très proche de 100% quelle que soit la méthode, parmi les quatre méthodes présentées.

La sensibilité, la spécificité et les valeurs prédictives positive et négative sont des indices permettant d'évaluer la précision ou les performances d'une méthode à trouver les pixels défectueux ou les bons pixels, mais pas les deux ensembles. Cependant, le coefficient-Phi permet de remédier à ce problème, car il est un indice qui permet d'évaluer la corrélation des méthodes en prenant en compte la précision et les performances de détection des pixels bons et défectueux.

5.4.2.4 Coefficient-Phi

Le coefficient-Phi (ϕ) est une métrique d'association entre deux variables binaires [140]. Cette métrique est similaire au coefficient linéaire de Pearson en interpolation entre deux variables aléatoires, à la différence qu'elle a été conçue pour effectuer la comparaison entre deux variables nominales à distributions dichotomiques (qui ne peuvent prendre que les valeurs de 0 ou 1). Ce coefficient-phi s'interprète donc de la même façon que le coefficient de corrélation linéaire de Pearson. Il est compris entre -1 et 1, où des valeurs

TABLE 5.4 – Table de contingence des résultats de tests

		Image		
		Défect.	Bon	Total
Méthode	Défect.	VP	FP	A
	Bon	FN	VN	B
	Total	C	D	N

proches de 0 représentent une corrélation très faible entre les variables ; et des valeurs proches de 1 en valeur absolue représentent une corrélation (liaison) très forte entre les variables.

La définition du coefficient-phi passe généralement par la définition du coefficient de contingence ou encore Khi-deux (χ^2). Étant donné les résultats des tests effectués plus haut, on peut construire une table de contingence comme celle présentée dans le Tableau 5.4, et définir le coefficient du Khi-deux comme présenté dans (5.24). Ce paramètre du Khi-deux est biaisé par la taille de l'échantillon, d'où l'utilisation du coefficient-Phi défini dans (5.25), qui donne une normalisation du Khi-deux par la taille de l'échantillon et donne un résultat compris entre 0 et 1, facilement interprétable.

$$\chi^2 = \frac{(VP \times VN - FP \times FN)^2 \times N}{A \times B \times C \times D} \tag{5.24}$$

$$\phi = \sqrt{\frac{\chi^2}{N}} \tag{5.25}$$

où N représente la taille de l'image ; A, B, C et D sont des combinaisons des lignes et des colonnes comme présentées dans le Tableau 5.4 de contingence, le calcul de ces variables est donné dans (5.26).

$$\begin{aligned} N &= VP + VN + FP + FN \\ A &= VP + FP \\ B &= FN + VN \\ C &= VP + FN \\ D &= FP + VN \end{aligned} \tag{5.26}$$

Le coefficient-Phi peut aussi se calculer avec une formule simplifiée sans passer par le calcul du Khi-deux, comme présenté dans (5.27).

$$\phi = \frac{VP \times VN - FP \times FN}{\sqrt{A \times B \times C \times D}} \tag{5.27}$$

La Figure 5.12 et le Tableau 5.5 présentent les valeurs moyennes du coefficient-Phi des quatre méthodes de détection des pixels défectueux présentées dans ce chapitre, pour chacune des 144 images déformées, regroupées par types de défauts. Cette figure et

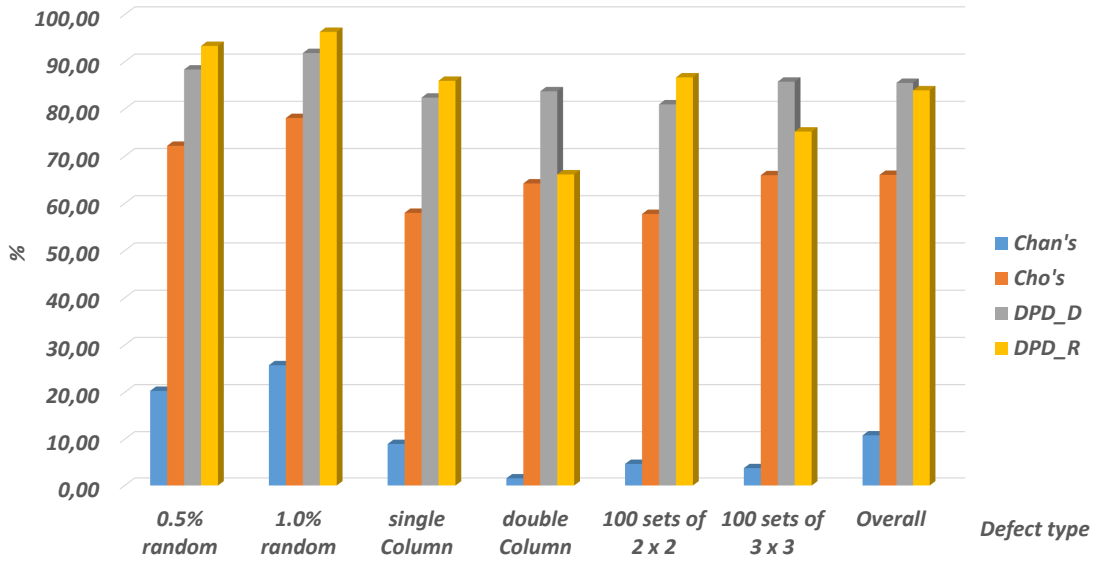


FIGURE 5.12 – Coefficient-Phi pour différentes méthodes avec 6 types de défauts

TABLE 5.5 – Coefficient-Phi pour différentes méthodes

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets** 2×2	**sets** 3×3	**Total**
Chan	20.30 %	25.73 %	8.97 %	1.54 %	4.66 %	3.75 %	10.82 %
Cho	72.19 %	78.08 %	58.00 %	64.23 %	57.77 %	65.97 %	66.04 %
DPD_D.	88.35 %	91.84 %	82.41 %	**83.75 %**	80.99 %	**85.78 %**	**85.52 %**
DPD_R.	**93.35 %**	**96.32 %**	**85.96 %**	66.14 %	**86.70 %**	75.22 %	83.95 %

ce tableau montrent que les méthodes proposées (DPD_D et DPD_R) sont meilleures que les méthodes existantes dans la littérature (proposées par Chan, et par Cho et al.) en termes de corrélation entre les pixels détectés comme défectueux par les méthodes, et le nombre de pixels effectivement défectueux présents dans l'image. On peut aussi remarquer que les corrélations sont moins fortes pour les défauts regroupant plusieurs pixels défectueux dans la même zone telle que « $2 - colonnes$ », « bloc de 3×3 pixels », que les corrélations des défauts répartissant les pixels défectueux dans toute l'image, comme pour les indices précédemment étudiés. Ainsi les méthodes proposées ont-elles par exemple, des corrélations de 91.84% et 91.72% pour le défaut consistant à insérer aléatoirement 1% de pixels défectueux dans les images. Au final, on remarque qu'il existe des corrélations moyennes sensiblement égales à 10.82% et 66.04% pour les méthodes proposées par Chan et par Cho et al., respectivement ; tandis que les méthodes proposées donnent des corrélations sont nettement plus élevées, qui sont sensiblement égales à 83.95% et 85.52% pour les méthodes basées sur les dispersions dans les zones de pixels et sur les distances entre le pixel à tester et ses pixels voisins, respectivement.

On peut alors conclure que la méthode proposée par Chan [18] a une corrélation faible entre les pixels détectés morts par cette méthode et les pixels effectivement morts au sein des images ; la méthode proposée par Cho et al. [137, 136], quant à elle, présente une corrélation moyenne en termes de détection de pixels morts dans les images. Tandis que, les méthodes proposées, nommées DPD_D et DPD_R, dénotent des corrélations très fortes, supérieures à 80%, entre les pixels détectés comme défectueux par ces méthodes, et les pixels effectivement défectueux dans les images.

5.4.3 Résultats de correction des images

Chacune des méthodes étudiées a été utilisée pour détecter et corriger les pixels défectueux présents dans les 144 images déformées. Les résultats de la phase de correction sont donnés en termes de rapport signal sur bruit de crête ($PSNR$) entre l'image de référence et l'image corrigée par la méthode. En rappel, le $PSNR$ se calcul comme présenté dans (5.28) ;

$$PSNR = 10log(\frac{d^2}{MSE}) \tag{5.28}$$

où d représente le maximum de l'intensité dans l'image (255 dans le cas d'une image en niveau de gris codée sur 8 bits) ; etMSE représente l'erreur quadratique moyenne, donnée par (5.29), où N et M qui sont les dimensions des images I_o et I_d.

$$MSE = \frac{1}{M.N} \sum_{i=1}^{M} \sum_{j=1}^{N} (I_o(i,j) - I_d(i,j))^2 \tag{5.29}$$

La Figure 5.13 et le Tableau 5.6 présentent la comparaison du $PSNR$ moyen obtenu par les quatre méthodes de détections et de corrections de pixels défectueux présentées dans ce chapitre, sur toutes les 144 images de déformées, regroupées par types de défauts. Dans cette figure et ce tableau, on retrouve aussi le $PSNR$ de ces images déformées de

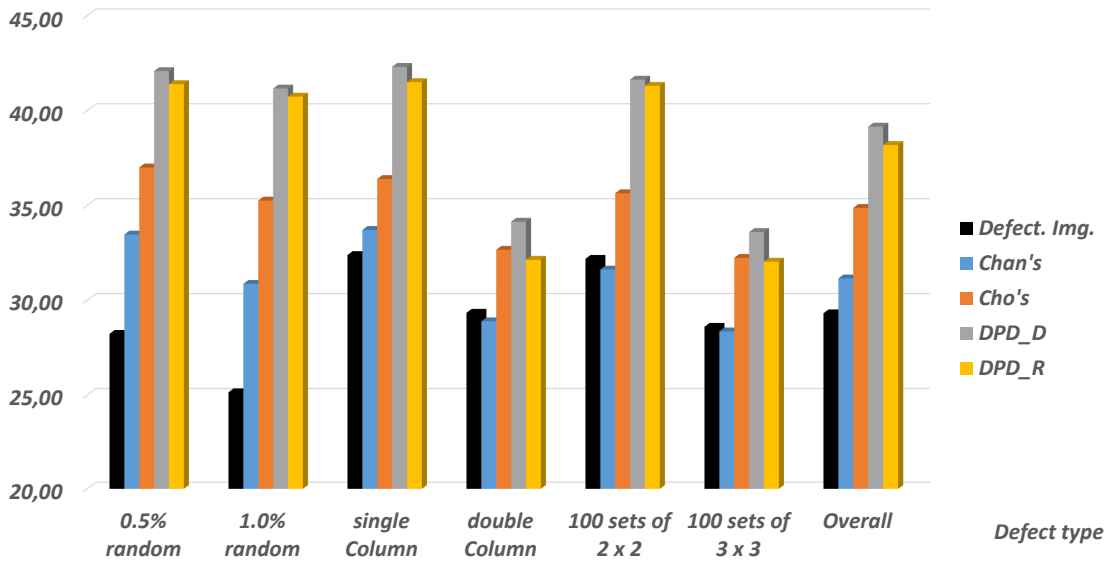


FIGURE 5.13 – PSNR Pour différentes méthodes avec six types de défauts

TABLE 5.6 – PSNR Pour différentes méthodes avec six types de défauts

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets 2×2**	**sets 3×3**	**Total**
Im. Def.	28.23	25.15	32.39	29.34	32.19	28.61	29.32
Chan	33.47	30.87	33.71	28.89	31.62	28.36	31.15
Cho	37.00	35.26	36.39	32.66	35.64	32.24	34.87
DPD_D.	**42.08**	**41.16**	**42.30**	**34.14**	**41.63**	**33.60**	**39.15**
DPD_R.	41.39	40.73	41.50	32.14	41.30	32.04	38.18

tests avant correction. En rappel, les méthodes de correction appliquées pour chacune des méthodes de détection sont :

— pour la méthode proposée par Chan [18], corriger le pixel détecté comme « mort » en le remplaçant par une interpolation (la moyenne) de ses 8 pixels voisins ;
— pour la méthode proposée par Cho et al. [137, 136], corriger un pixel détecté comme « mort » en le remplaçant par l'interpolation (moyenne) de ces 4 pixels voisins formant une croix (en forme de signe +) avec le pixel mort (les 2 pixels horizontaux et les 2 pixels verticaux autour du pixel mort) ;
— pour les méthodes proposées dans ce chapitre, DPD_D basée sur la distance entre le pixel à tester et ces pixels voisins, et DPD_R basée sur la dispersion et les écarts dans les zones de l'image ; la méthode de correction est basée sur les filtres médians. Cette méthode de correction consiste à remplacer le pixel détecté comme défectueux par la médiane du vecteur constitué des pixels formant avec le pixel défectueux une matrice de taille 3×3.

Une des images de référence parmi celles utilisées pour construire les images déformées de tests est présentée dans la Figure 5.14 (a). Cette image est déformée par 2 types de défauts, le défaut consistant à insérer aléatoirement 1% de pixels défectueux dans l'image, et celui consistant à insérer $1 - \textit{colonne}$ entière de pixels défectueux sur une colonne aléatoire de l'image ; l'image déformée résultante est présentée dans la Figure 5.14 (b). Dans cette même Figure 5.14, les graphes de (c) à (f) sont les versions corrigées de cette image déformée, présentée en (b), par les différentes méthodes de détection et de correction des pixels défectueux présentées dans ce chapitre : méthodes de Chan, Cho et al., DPD_D basée sur les distances, et DPD_R basée sur les dispersions, pour les graphes (c), (d), (e) et (f), respectivement. La Figure 5.15, quant à elle, présente un « zoom » sur les images corrigées par les deux méthodes proposées, ce qui permet de voir les effets des fausses détections (effet des faux positifs) sur les images corrigées.

La remarque évidente est que les méthodes proposées détectent mieux, et aussi, corrigent mieux les images que les méthodes proposées par Cho et al., et par Chan. Un élément à noter sur la Figure 5.15, est l'effet des « faux positifs » dans une zone de l'image, sur les images corrigées pour les deux méthodes proposées. Cela permet de voir que la méthode basée sur les dispersions qui produit moins de « faux positifs », corrige mieux certaines zones de l'image que la méthode basée sur les distances, qui détecte plus de pixels défectueux, mais aussi, produit plus de « faux positifs ».

5.4.4 Complexité en temps des méthodes

Au vu du temps de capture d'une image, ou du nombre d'images prises par un capteur dans une vidéo, le temps d'exécution est l'un des plus importants facteurs dans l'évaluation des performances d'une méthode de détection et de correction des pixels défectueux. La Figure 5.16 et le Tableau 5.7 présentent le temps moyen d'exécution en secondes (s), pour les quatre méthodes présentées dans ce chapitre. Aussi bien pour la détection que pour la correction des pixels défectueux dans les images déformées de résolution 512×384 pixels, regroupées en fonction des types de défauts. Ces méthodes sont évaluées sous l'outil $MATLAB$, installé sur un ordinateur personnel de $16\,GB$ de

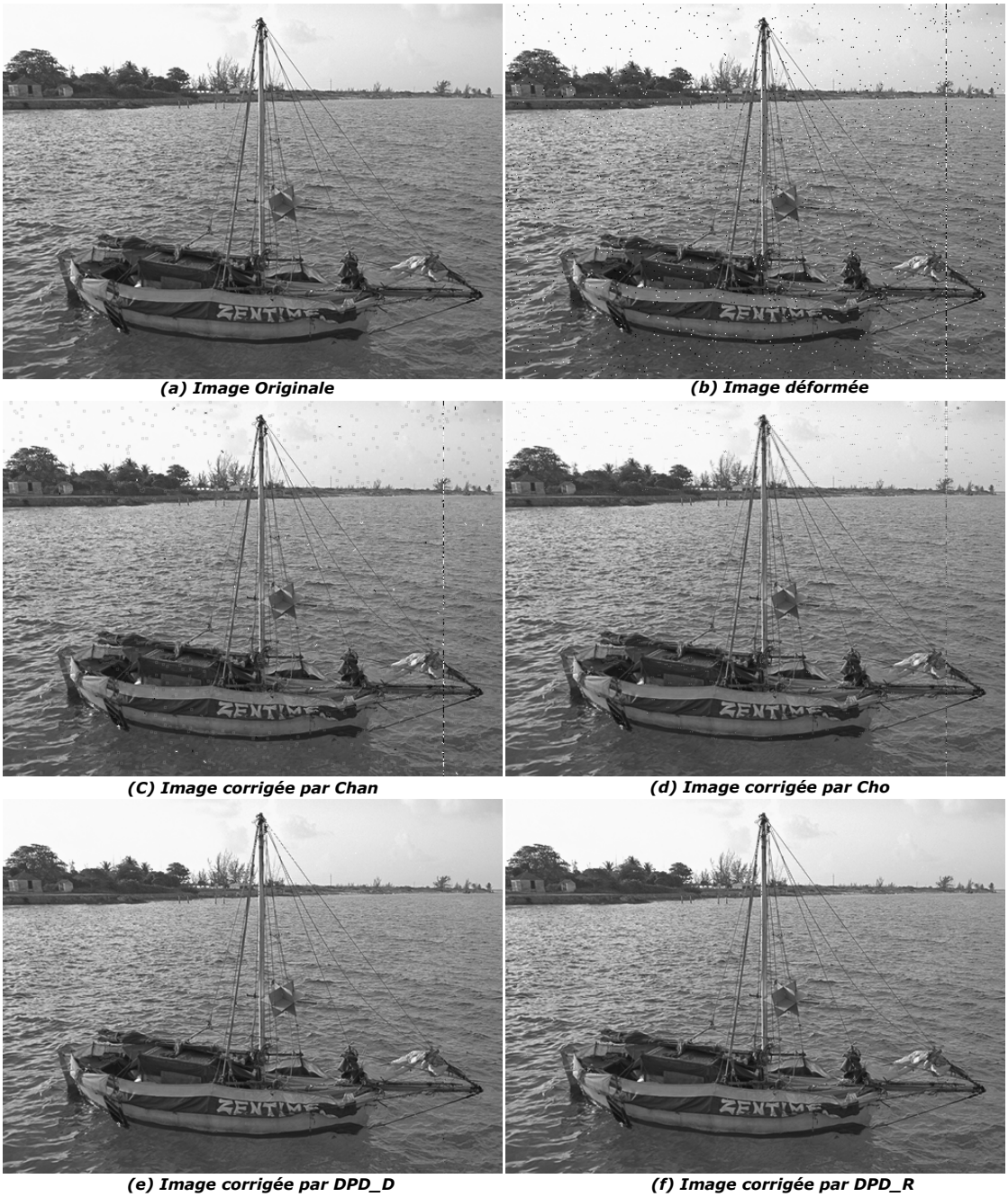


FIGURE 5.14 – Exemple d'une image corrigée par différentes méthodes

(a) Image corrigée par DPD_D **(b) Image corrigée par DPD_R**

FIGURE 5.15 – Zoom sur les images corrigées par les méthodes proposées

RAM, avec un processeur à deux cœurs Intel Xeon ($1.70Ghz$ chacun).

Lorsqu'on regarde les temps moyens d'exécution des différentes méthodes, on remarque tout de suite que la méthode proposée basée sur les dispersions et les écarts dans les zones de l'image s'exécute plus rapidement que toutes les autres méthodes de détection de pixels défectueux présentées dans ce chapitre, avec un temps moyen d'exécution de $0.237\,s$ pour la détection et la correction des pixels défectueux dans une image.

TABLE 5.7 – Temps d'exécution pour différentes méthodes sous MATLAB en seconde (s)

	0.5% rd.	**1% rd.**	**1-col.**	**2-col.**	**sets** 2×2	**sets** 3×3	**Total**
Chan	0.777	0.785	0.782	0.780	0.781	0.782	**0.781**
Cho	0.947	0.948	0.942	0.942	0.943	0.945	**0.944**
DPD_D.	0.831	0.830	0.832	0.829	0.830	0.830	**0.830**
DPD_R.	0.247	0.258	0.230	0.227	0.230	0.229	**0.237**

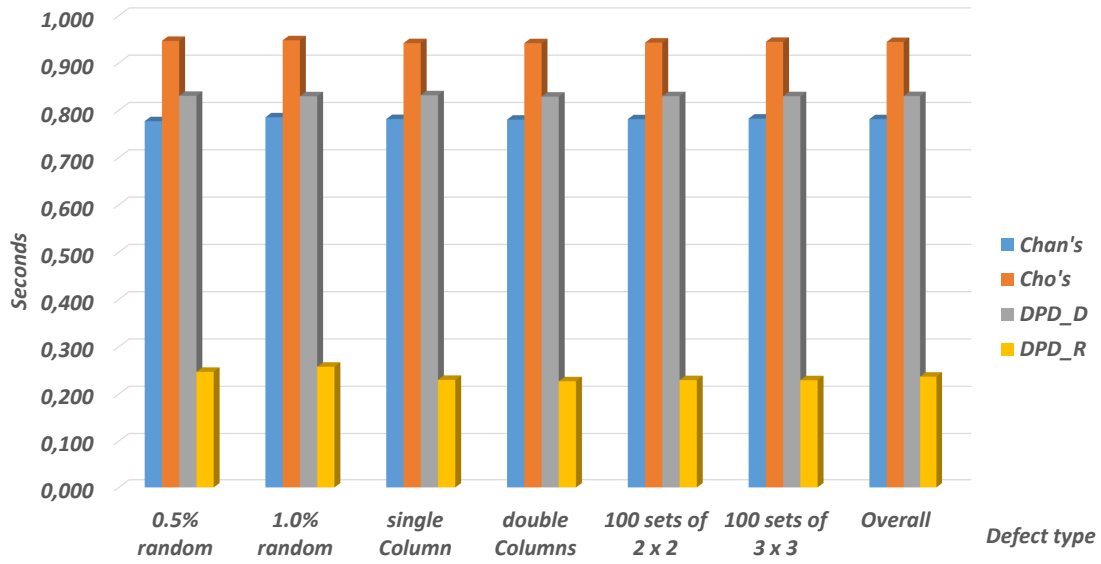


FIGURE 5.16 – Temps d'exécution pour différentes méthodes, groupés par types de défauts

5.5 Architectures et résultats d'implémentation

Une fois les algorithmes mis sur pied dans l'outil $MATLAB$, il importait de savoir ce qu'ils donnent comme résultats sur une carte $FPGA$ en termes de temps d'exécution et de surfaces utilisées, pour toutes les méthodes de détection de pixels défectueux étudiées dans ce chapitre. Chaque algorithme de détection et de correction des pixels défectueux dans les images a été implémenté sur une carte FPGA Xilinx de la famille des Virtex 7 ($VC707$), grâce à la gamme d'outil Vivado et Vivado HLS de Xilinx.

5.5.1 Architecture d'implémentation

Chaque méthode a tout d'abord été implémentée en HLS, grâce à l'outil « Vivado HLS », en langage « C/C++ », et les blocs IP ($IP-Core$) ont été générés pour chacune des méthodes étudiées. Partant des blocs IP générés en HLS, l'architecture générale d'implémentation en RTL a été mise en place comme présenté dans la Figure 5.17. Dans cette architecture générale, on distingue plusieurs composants tels que :

— Microcontrôleur (MicroBlaze) : il est l'élément central de l'architecture d'implémentation. Il exécute les tâches principales suivantes : 1) initialiser et configurer tous les paramètres du système ; 2) coordonner les séquences de déplacement des données à l'intérieur et entre les différentes sous unités du système ; 3) effectuer certaines opérations arithmétiques et logiques.

— bloc d'accès direct à la mémoire DMA : ce bloc DMA, basé sur le protocole de

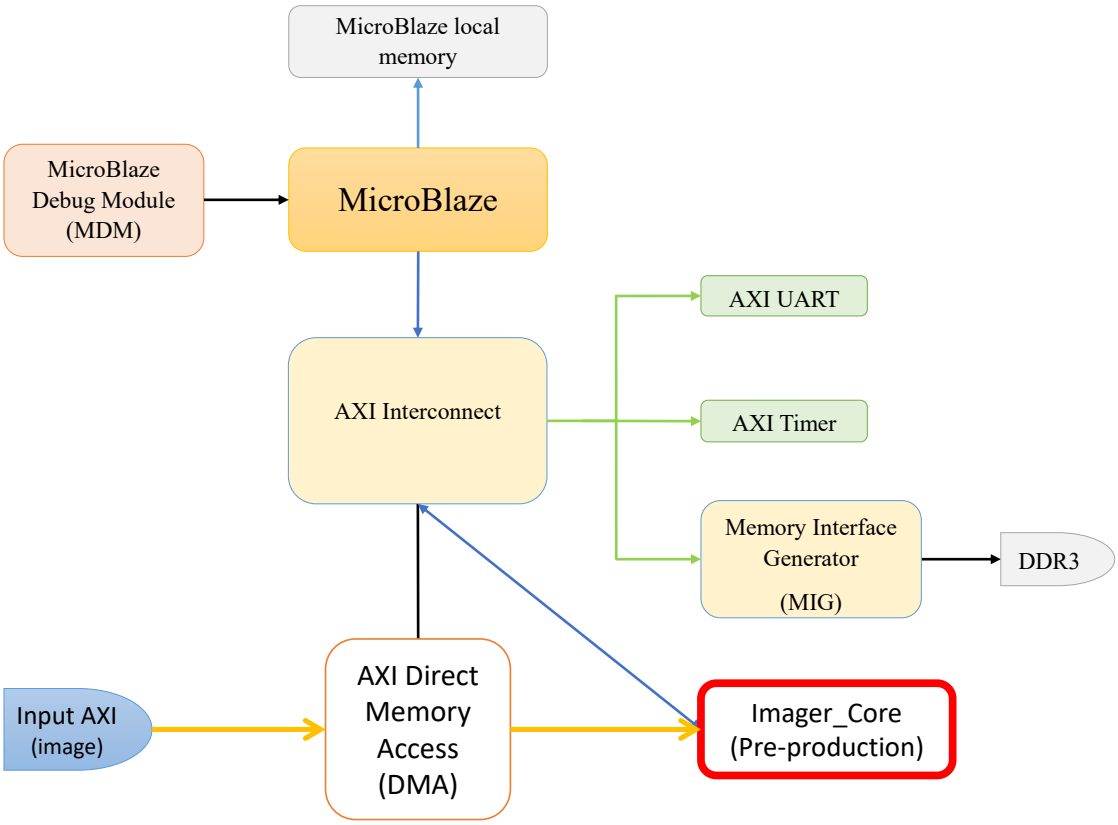


FIGURE 5.17 – Architecture d'implémentation générale des méthodes de DPD

TABLE 5.8 – Estimation des performances et utilisations de différentes méthodes de DPD

	Chan's	Cho's	**DPD_D.**	**DPD_R.**
clock estim. (ns)	6.84	4.26	4.26	4.29
# cycle max.	3 871 856	2 299 509	3 345 015	2 536 620
BRAM	256	256	256	256
DSP	3	13	10	0
FF	3 786	3 370	4 702	6 528
LUT	6 490	4 224	6 873	11 886

communication AXI, permet de transmettre une image de l'ordinateur vers la carte $FPGA$, et inversement. Les images sont transmises pixel par pixel pour éviter une surcharge de la mémoire de la carte $FPGA$. Ce bloc permet d'avoir accès à la mémoire tampon où est stockée l'image, donc de transmettre l'image aux autres blocs du modèle qui en ont besoin.

— Unité d'interconnexion AXI : il permet la connexion entre le microcontrôleur (MicroBlaze) et les autres périphériques (sous unités) du système.
— Interface de génération de mémoire (MIG) : il est utilisé comme un convertisseur entre les protocoles d'interconnexion AXI et ceux de la RAM.
— $UART$: est l'unité qui permet la transmission des résultats contenus dans le microcontrôleur vers une machine extérieure.
— Timer : il est utilisé pour mesurer le temps d'exécution de certains processus du système.

Le processus présenté dans cette architecture consiste pour une image prise en entrée, de la transmettre au bloc IP de traitement (Imager_Core) qui effectue la détection et la correction des pixels défectueux dans cette image, retourne l'image corrigée au bloc $AXI-DMA$ qui se chargera de la retourner à l'ordinateur ou de l'afficher sur un écran, en utilisant le protocole de communication AXI.

Le Bloc « Imager_Core » quant à lui, est le bloc IP généré en HLS qui décrit la méthode de détection et de correction à exécuter. Ce bloc prend comme entrée une image déformée au format $AXI-Stream$ et retourne une image corrigée par l'une des méthodes étudiées précédemment, sous ce même format ($AXI-Stream$). Les résultats obtenus en termes d'estimation des performances et des utilisations lors de la phase de synthèse HLS de ce bloc IP (Imager_Core) pour les différentes méthodes étudiées précédemment, sont données dans le Tableau 5.8. Dans ce tableau apparaissent :

1. l'estimation des performances : représentée pour une période de 5 nanosecondes (soit une fréquence de 200 Mhz), qui donne le nombre de cycle d'horloge nécessaire à l'exécution complète du bloc IP, pour chacune des méthodes de détection de pixels défectueux implémentées, décrites dans la littérature par Chan, par Cho et al. ; ou encore les méthodes que proposées dans le cadre de ce travail, DPD_D

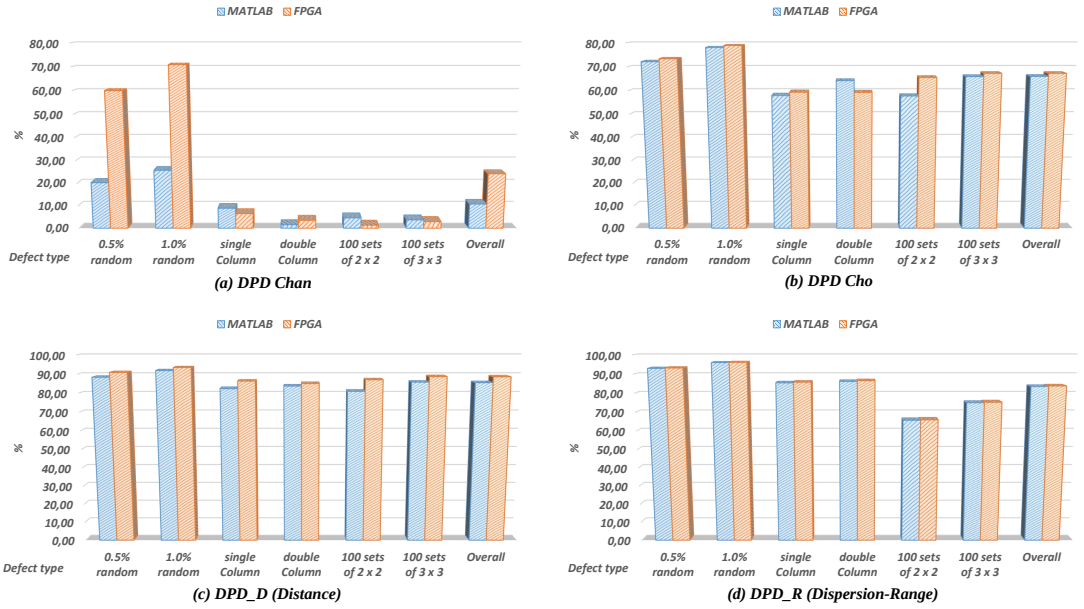


FIGURE 5.18 – Comparaison du Coefficient-Phi en Simulation Vs Implémentation par types de défauts

basée sur les distances et DPD_R basée sur les dispersions.

2. l'estimation des utilisations : qui présente la surface et le nombre de portes utilisées sur le système $FPGA$, en termes de blocs mémoires (en $Kbits$), registres, portes logiques, DSP, bascules Flip-Flop et LUT.

Un constat général est que, les ressources utilisées sont presque pareilles pour toutes les méthodes proposées, mais aussi que la quantité de ressources utilisées est très faible.

5.5.2 Résultats d'implémentation

Toutes ces méthodes de détection de pixels défectueux ayant été implémentées sur une carte $FPGA$. Elles ont été utilisées pour détecter et corriger les pixels défectueux présents dans les 144 images déformées présentées plus haut. La Figure 5.18 présente la comparaison des résultats de détection des pixels défectueux dans les images déformées, en termes de coefficient-Phi, qui représente la corrélation de détection, entre les résultats obtenus lors de la phase de simulation sous $MATLAB$ et la phase d'implémentation sur un système $FPGA$. Ces résultats sont groupés par types de défauts, pour chacune des quatre méthodes de détection des pixels défectueux, proposées par Chan, et par Cho et al. ; DPD_D basée sur les distances et DPD_R basée sur les dispersions, dans les graphiques (a), (b), (c) et (d) de la Figure 5.18, respectivement.

La Figure 5.19, quant à elle, présente la comparaison des moyennes du coefficient-Phi lors des phases de simulation sous $MATLAB$ et d'implémentation sur un système

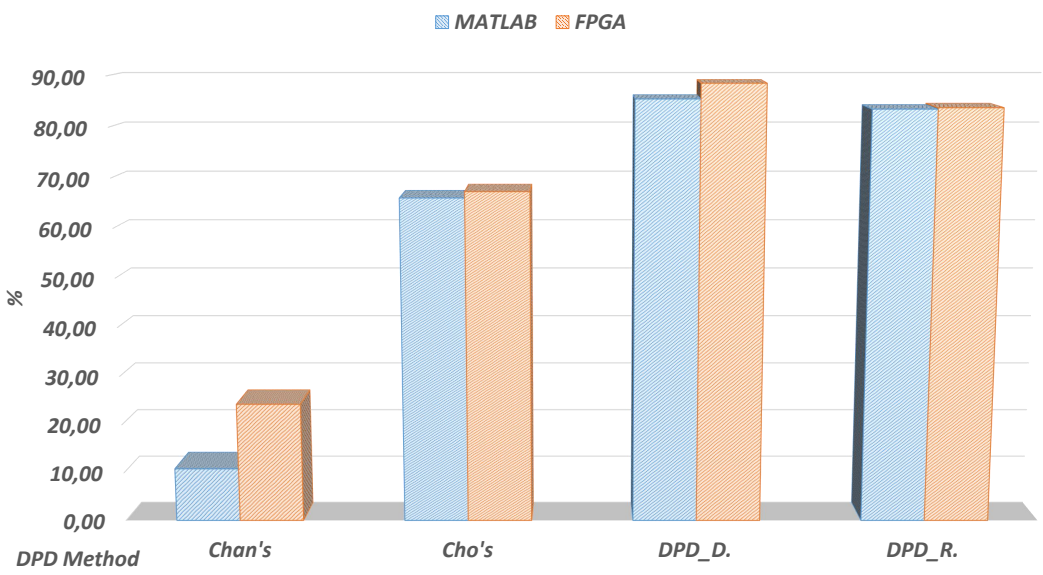


FIGURE 5.19 – Comparaison du Coefficient-Phi en Simulation Vs Implémentation globale

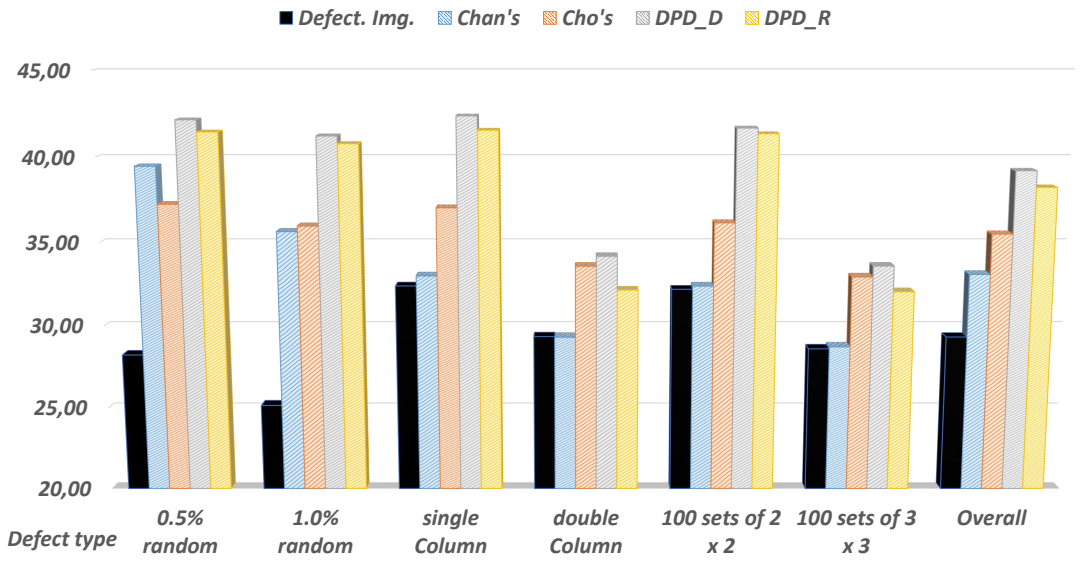


FIGURE 5.20 – PSNR en Implémentation par types de défauts pour différentes méthodes

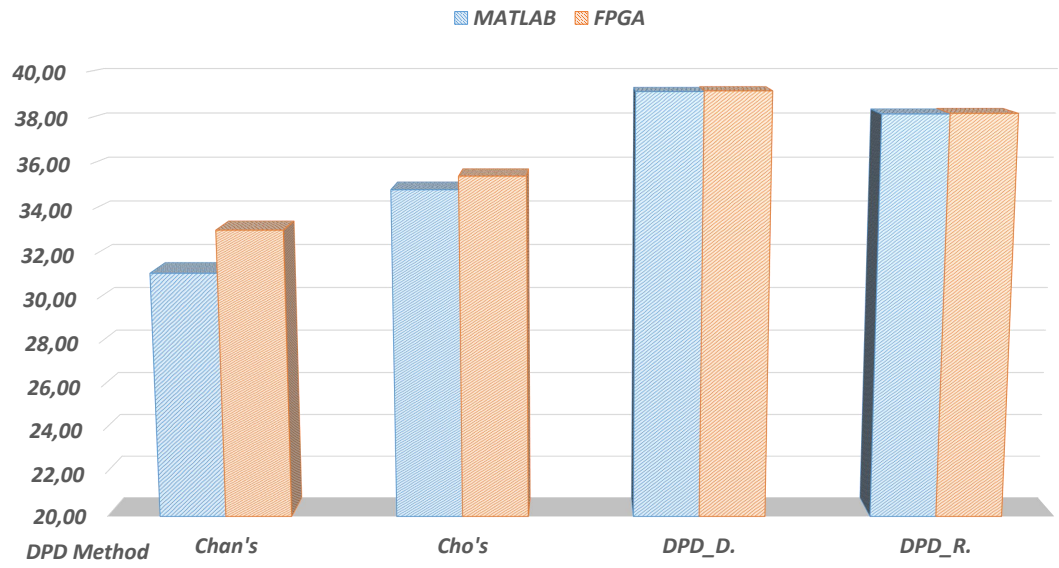


FIGURE 5.21 – Comparaison du $PSNR$ en Simulation Vs Implémentation globale

$FPGA$, pour toutes les 144 images déformées de test, et pour chacune des méthodes étudiées (Chan, Cho, DPD_D et DPD_R).

La Figure 5.20 présente le $PSNR$ obtenu par les différentes méthodes de détection et de correction, lors de la phase d'implémentation sur $FPGA$, sur les 144 images déformées regroupées par types de défauts ; tandis que la Figure 5.21 présente la comparaison des moyennes du $PSNR$ obtenu sur l'ensemble des 144 images déformées, lors des phases de de simulation sous $MATLAB$ et la phase d'implémentation sur un système $FPGA$, pour les quatre méthodes de détection et de correction étudiées dans ce chapitre.

La remarque principale à tirer de ces quatre figures sur les résultats d'implémentation et leur comparaison aux résultats de simulation, est que le coefficient-Phi et le rapport signal sur bruit de crête ($PSNR$) sont presque identiques lors de la phase d'implémentation sur $FPGA$ et lors de la phase de simulation sous $MATLAB$, pour les méthodes proposées.

5.6 Bilan et apports

Le but de ce chapitre était de présenter des algorithmes en ligne et en temps-réel de détection et de correction de pixels défectueux dans les images et capteurs d'images en utilisant, pour la plupart, des opérations arithmétiques de base, sur les pixels voisins au pixel à tester. Ces algorithmes consistent pour une image prise en entrée, à parcourir cette image pixel par pixel, et pour chacun des pixels de l'image, sélectionner ses pixels voisins, puis mener une étude comparative entre ces pixels sélectionnés et le pixel à

tester, pour déterminer s'il est concordant ou pas avec son voisinage. Si le pixel à tester est déclaré défectueux, on enclenche le processus de correction, qui utilise aussi les pixels voisins au pixel déclaré défectueux, pour trouver la valeur la plus à même de remplacer la valeur défectueuse du pixel.

Les algorithmes de détection de pixels défectueux proposés dans ce chapitre se basent sur la distance entre le pixel à tester et ses pixels voisins ainsi que sur les dispersions dans les blocs de pixels de l'image. Et l'algorithme de correction se base sur le principe des filtres médians, qui consiste à remplacer un pixel par la médiane du bloc de pixels constitué des pixels voisins au pixel déclaré défectueux. Les performances des algorithmes proposés ont été comparées aux performances de deux autres algorithmes de détection de pixels défectueux présents dans la littérature, l'un proposé par Chan [18] et l'autre proposé par Cho et al. [137, 136]. Les performances des algorithmes de détection de pixels défectueux sont comparées en termes de sensibilité et spécificité qui permettent de mesurer la précision des méthodes, les valeurs prédictives positive et négative qui permettent de mesurer la probabilité individuelle d'un pixel à être défectueux, et enfin le coefficient-Phi qui est l'équivalent du coefficient linéaire de Pearson pour la comparaison entre deux variables nominales à distributions dichotomiques. Les performances des algorithmes de correction quant à eux, sont comparées en termes de rapport signal sur bruit de crête ($PSNR$), entre l'image déformée et l'image corrigée par une méthode de détection et correction des pixels défectueux dans une image.

Les algorithmes de détection et de correction de pixels défectueux dans les images ont été implémentés sous l'outil $MATLAB$, mais aussi sur une carte $FPGA$ de la famille des Virtex 7 ($VC707$). Les résultats obtenus lors de la phase de simulation sous $MATLAB$ et d'implémentation matérielle sur $FPGA$ ont été comparés, non seulement pour chacune des méthodes, mais aussi entre les différentes méthodes proposées et celles présentes dans la littérature. 144 images déformées ont été utilisées pour cette étude et comparaison. Ces images déformées sont issues de 24 images de références infectées par 06 types de défauts. Ces types de défauts sont les différents types de pixels défectueux pouvant infecter des images ou des capteurs d'images.

Les apports nouveaux dans la détection et correction des pixels défectueux dans une image ou un capteur d'images, peuvent se résumer en quatre principales idées :

1. La mise en place des algorithmes de détection de pixels défectueux dans une image ou un capteur d'images. Le premier algorithme de détection de pixels défectueux proposé est basé sur les distances entre le pixel à tester et ses pixels voisins. Cette notion de distance permet de lier la qualité hypothétique (valeur estimée) d'un pixel en réduisant le poids ou l'apport des pixels très éloignés du pixel à tester. Tandis que le second algorithme proposé est basé sur la dispersion et les écarts entre les pixels des blocs de l'image. Dans cette méthode, l'image est divisée en blocs de même taille ; et pour chacun des blocs, on recherche l'existence des valeurs aberrantes, en considérant l'étendue du bloc, sa médiane, ses valeurs de quartiles, et son écart interquartile.
2. La correction des pixels détectés comme défectueux est basée sur les filtres médians. Cela consiste tout simplement à remplacer le pixel détecté comme défectueux par

la médiane du bloc de pixel qu'il forme avec ses pixels voisins. On suggère d'utiliser la médiane plutôt que la moyenne, pour simplifier les influences des potentielles valeurs aberrantes présentes dans le voisinage du pixel à corriger.

3. L'utilisation de différents indices de performance pour l'évaluation des divers algorithmes de détection et de correction des pixels défectueux dans les images mis en place dans ce chapitre. Parmi ces indices de performance, on retrouve des indices de précision, tels que : la sensibilité, pour les valeurs détectées positives (précision de détection des pixels défectueux) ; et la spécificité, pour les valeurs détectées négatives (précision de détection des bons pixels). Une deuxième catégorie d'indices permet de mesurer de la probabilité individuelle d'un pixel du point de vue des valeurs prédictives positive et négative. Ici, on peut retrouver des indices liés à la performance globale de l'algorithme, c'est-à-dire, précision et probabilité, positive et négative : il s'agit du coefficient-Phi. En ce qui concerne les méthodes de correction, leurs performances sont mesurées grâce au rapport signal sur bruit de crête ($PSNR$) entre l'image déformée et l'image corrigée par la méthode de correction. Les différentes méthodes de détection et de correction de pixels défectueux que mises au point ici pourraient être recommandées au regard des résultats largement supérieurs à ceux des méthodes présentes dans la littérature.
4. L'implémentation des algorithmes proposés sur une carte $FPGA$, ce qui a permis d'étudier la complexité en temps et en matériel desdits algorithmes. Cette implémentation montre que les résultats sur la carte $FPGA$ sont presque identiques aux résultats de simulation sous l'outil $MATLAB$. Elle a aussi révélé que les méthodes proposées ont des complexités en temps et en espace très faible.

Chapitre 6

Conclusion et Perspectives

Cette thèse s'inscrit dans un projet collaboratif entre des partenaires industriels et académiques, ayant pour but la mise en place des méthodes avancées d'analyse statistique pour la détection et la correction d'erreurs intégrées dans un décodeur numérique d'images basé sur le CODEC $HEVC$.

Pour atteindre cet objectif, il a tout d'abord fallu faire un état de l'art sur la qualité des images, et tout ce qui concerne leur évaluation, enfin de mieux circonscrire le problème. Ensuite ce sont imposées des contributions sur l'évaluation de la qualité des images à référence complète, et dans un cadre plus applicatif, sans référence. Enfin, des contributions ont été faites dans le domaine de la détection et la correction des pixels défectueux dans une image ou un capteur d'image numérique. Toutes ces contributions ont ainsi permis d'atteindre l'objectif de cette étude qui était le développement des algorithmes et méthodes de détection et de correction des erreurs intégrées dans un décodeur numérique d'images.

6.1 Contributions dans l'évaluation de la qualité des images

Pour ce qui est de l'évaluation de la qualité des images, ce travail a consisté principalement à construire des modèles d'évaluation objectifs, temps-réels et faciles à implémenter sur une architecture physique du genre $FPGA$ ou $ASIC$. Ces modèles sont envisagés pour permettre d'évaluer la qualité visuelle d'une image, à partir d'un ensemble de caractéristiques extraites de cette-ci, ou d'un ensemble d'images en rapport avec ladite image, à l'aide des méthodes d'apprentissage automatiques. Le processus de construction de ces modèles consiste en une phase d'apprentissage à partir d'une base de données constituée d'images dégradées et originales, ainsi que des scores subjectifs de ces images données par un ensemble d'observateurs humains. De ces images dégradées et pour certaines méthodes des images de références aussi, on extrait un ensemble de caractéristiques ou métriques, qui deviennent les entrées des algorithmes d'apprentissage, avec les scores subjectifs des images dégradées. En sortie de ces algorithmes d'apprentissage, on obtient des modèles, qui permettront l'évaluation de la qualité de nouvelles images prises dans une base de données d'images. La base de données test n'est pas nécessairement la même

que la base de données d'apprentissage.

L'investigation a aussi portée sur plusieurs méthodes d'apprentissage automatiques, notamment les méthodes de classification, et celles de régression. Plus spécifiquement, deux méthodes de classification ont été étudiées, il s'agit de : la méthode d'analyse discriminante, et la méthode des k-plus proches voisins. Ces deux méthodes ont produit des modèles permettant une bonne classification de l'ensemble des images en fonction de leur qualité visuelle. Cinq méthodes de régression ont été utilisées dans ces travaux, il s'agit des méthodes basées sur : les réseaux de neurones artificiels, les machines à vecteurs supports, la régression non-linéaire polynomiale, les arbres de décision, et enfin la logique floue. Les quatre premières méthodes de régression avaient déjà été utilisées pour la construction des modèles objectifs d'évaluation de la qualité des images [141, 8, 7], mais c'est tout au long de cette thèse qu'a été introduit l'utilisation de la logique floue pour la construction des modèles objectifs d'évaluation de la qualité des images.

Les comparaisons entre les scores objectifs obtenus par des modèles et les scores subjectifs pris dans les bases de données d'images, se font à partir des indices de performance tels que : l'erreur quadratique moyenne, le coefficient de détermination, les coefficients de corrélation de Pearson, Spearman et Kendall, et la distance de Brownian. Mais aussi, pour confirmer les résultats obtenus et tester la fiabilité et robustesse des méthodes d'apprentissage et des modèles qu'elles produisent, il a été utilisé la méthode de validation croisée de Monte-Carlo, avec 1000 répétitions des phases d'apprentissage et de test. Dans cette méthode de validation, les données réparties aléatoirement entre les phases d'apprentissage et de test. Les meilleurs résultats en termes de corrélation avec les scores subjectifs des images sont donnés par les modèles construits à partir des méthodes basées sur les arbres de décision et sur la logique floue. Mais Plus spécifiquement, les meilleurs résultats en termes de stabilité des modèles construits lors de la phase de validation croisée, sont produits par la méthode de logique floue.

Deux familles d'évaluations objectives de la qualité des images ont été étudiées dans cette thèse : une évaluation de la qualité des images avec référence et une évaluation des images sans référence à distorsion non spécifique. Dans l'évaluation avec référence, les métriques d'entrées des méthodes d'apprentissage proviennent de la comparaison entre l'image dégradée et l'image de référence. Après la construction des métriques, on sélectionne celles qui sont indépendantes entre elles, et fortement corrélées avec les scores subjectifs. Ce sont ces métriques sélectionnées qui ont été utilisées pour la construction des modèles d'évaluation proposés, avec les scores subjectifs, et à l'aide des méthodes d'apprentissage automatiques.

Dans l'évaluation sans référence, un ensemble de caractéristiques sont extraites de l'image dégradée uniquement, sans aucune information sur l'image de référence suivant plusieurs axes d'extractions. Les axes d'extraction étudiés au cours de cette recherche sont : les statistiques de scènes naturelles dans le domaine spatial ; l'amplitude du gradient ; le laplacien de la gaussienne ; et les entropies spatiales et spectrales des blocs de l'image. Après l'extraction des caractéristiques, les valeurs aberrantes et superflues ont été supprimées, pour ne conserver que les caractéristiques indépendantes entre elles, notamment celles apportant de l'information sur la qualité visuelle des images. Les ca-

ractéristiques restantes sont utilisées comme entrées des algorithmes d'apprentissage, avec les scores subjectifs, pour construire les modèles d'évaluation de la qualité visuelle des images sans référence et à distorsion non spécifique.

Les résultats obtenus par les modèles d'évaluation élaborés ont été comparés à ceux obtenus par des index d'évaluation de la qualité visuelle des images, présents dans la littérature. Ces comparaisons ont montré que les modèles proposés donnent de meilleurs résultats en termes de corrélation avec les scores subjectifs, que ce soit pour des évaluations à référence complète, comme sans référence.

Aussi les modèles produisant les meilleurs résultats en termes d'évaluation de la perception visuelle ont-ils été implémentés sur une architecture matérielle de type carte $FPGA$. Les résultats de cette phase d'implémentation ont montré que ces modèles ont des complexités en temps et en surface très faibles, et qu'ils peuvent donc facilement être intégrés dans les décodeurs numériques d'images pour l'évaluation de la qualité des images reçues.

6.2 Contributions dans la détection et la correction des pixels défectueux

Pour ce qui est de la détection et la correction des pixels défectueux dans les images ou capteurs d'images, notre travail consistait à mettre sur pied des algorithmes temps-réels et en ligne permettant de parcourir les images pour détecter les pixels défectueux et les corriger. Les pixels défectueux étant définis comme ceux dont la valeur n'est pas en accord avec leur environnement. Pour détecter un pixel défectueux, une étude a été faite sur chacun des pixels de l'image et dans son environnement. Si sa valeur dévie d'un certain seuil de la valeur attendue en considérant le voisinage du pixel, il est considéré comme défectueux.

Il a été mis en œuvre, deux algorithmes de détection de pixels défectueux. Le premier est basé sur les distances entre le pixel à tester et ses pixels voisins. Cet algorithme consiste à parcourir l'image pixel par pixel, et pour chaque pixel de l'image, sélectionner ses pixels voisins. Les pixels ainsi sélectionnés sont utilisés pour jauger une valeur estimée du pixel central de l'environnement, qui est celui à tester, grâce à une moyenne des pixels sélectionnées, pondérés par la distance entre chacun d'eux et le pixel central à tester. La valeur réelle du pixel à tester est ensuite comparée à cette valeur estimée. Si l'écart entre les deux valeurs est supérieur à un certain seuil dépendant lui aussi de l'environnement du pixel et des écarts entre les pixels de cet environnement, alors le pixel à tester est déclaré « défectueux ».

Le second algorithme est basé sur la dispersion et sur les écarts dans les zones de l'image. Il consiste pour une image à tester à la diviser en bloc de même taille. Pour chacun des blocs de l'image, on évalue son étendue, son écart interquartile, sa médiane et d'autres paramètres de dispersion du bloc. Ces différents paramètres de dispersion sont utilisés pour déterminer s'il y a de pixels dans le bloc qui ont des comportements particuliers. Les pixels testés ici, sont les extrémités du vecteur trié. Si ces valeurs ont des comportements particuliers, ils sont déclarés comme « potentiellement défectueux ».

Alors se déclenche une phase de vérification, qui utilise les pixels autour du pixel déclaré comme « potentiellement défectueux » pour tester s'il l'est effectivement. Cette méthode permet d'aller plus rapidement, car on ne teste plus pixel par pixel, et on élimine ainsi les zones où tous les pixels sont cohérents et partant tous « bons pixels ».

La phase de correction quant à elle, intervient après la détection des pixels comme « défectueux ». Elle consiste à remplacer le pixel détecté comme défectueux par une valeur en accord avec l'environnement de celui-ci. Pour cela, on utilise les filtres médians, et donc on remplace le pixel défectueux par la médiane du vecteur constitué de ses pixels voisins.

Les résultats obtenus par les méthodes de détection et de correction proposées ont été comparés à ceux obtenus par des méthodes de détection et de correction présentes dans la littérature, à l'instar de la méthode proposée par Chan dans [18], et celle suggérée par Cho et al. dans [137, 136]. Les résultats de détection sont comparés en termes de sensibilité, de spécificité, de valeurs prédictives et de coefficient-Phi. Tandis que les résultats de correction sont comparés en termes de rapport signal sur bruit de crête ($PSNR$). La comparaison de ces résultats a démontré que les algorithmes proposés produisent de meilleurs résultats que les méthodes évoquées dans la littérature. Il s'agit particulièrement bon taux de détection ; du faible taux de « faux positifs », du meilleur coefficient-Phi, et du meilleur rapport signal sur bruit ($PSNR$).

Les algorithmes de détection et de correction proposés ont aussi été implémentés sur une carte $FPGA$. Les résultats de cette implémentation, en plus d'être presque identiques à ceux de la phase de simulation, ont démontré que les algorithmes proposés peuvent facilement être implémentés sur des capteurs d'images ou encore dans des décodeurs numériques d'images, afin de permettre de détecter et corriger les pixels défectueux dans les images en sortie, et ceci sans gêner le fonctionnement du capteur ou du décodeur.

6.3 Perspectives

Cette thèse ouvre des horizons à de nombreuses autres recherches telles que :

1. Développer un indicateur de fiabilité de capteur d'image utilisable durant son cycle de vie : il s'agirait, en l'occurrence, de partir de la qualité des images et du taux de pixels défectueux dans les images produites par un capteur pour suivre et jauger son vieillissement dans le temps.
2. Développer et implémenter des boucles de contrôle basées sur l'évaluation de la qualité des images et sur la correction de celle-ci, à partir des algorithmes de détection et de conciliation des pixels défectueux.
3. Diagnostiquer les capteurs d'image : il s'agirait pour un capteur d'image détecter comme défectueux, de développer des outils permettant de déceler les causes des ces défaillances. Elles pourraient être liées à un défaut dans le pixel, ou encore à un défaut sur l'un des éléments constituant le capteur ; ils comprennent : les convertisseurs analogique numérique (ADC), les amplificateurs, l'alimentation, le générateur d'horloge, etc.

Liste des publications

— Journal

[1] G. T. Tchendjou, E. Simeu, and R. Alhakim, “Fuzzy logic based objective image quality assessment with fpga implementation,” Journal of Systems Architecture, 2017.

— Conférences internationales

[2] G. T. Tchendjou and E. Simeu, “Defective pixel analysis for Image sensor online diagnostic and self-healing”, in 37th IEEE VLSI Test Symposium (VTS’19), April 23-25 2019, Monterey (CA), USA. (Accepted).

[3] G. T. Tchendjou and E. Simeu, “Self-healing imager based on detection and conciliation of defective pixels,” in 2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS). IEEE, 2018, pp. 251–254.

[4] G. T. Tchendjou, E. Simeu, and F. Lebowsky, “Fpga implementation of machine learning based image quality assessment,”, in 2017 29th International Conference on Microelectronics (ICM), IEEE, 2017, pp. 1–4.

[5] G. T. Tchendjou, R. Alhakim, and E. Simeu, “Fuzzy logic modeling for objective image quality assessment,” in 2016 Conference on Design and Architectures for Signal and Image Processing (DASIP), IEEE, 2016, pp. 98–105.

[6] G. T. Tchendjou, R. Alhakim, E. Simeu, and F. Lebowsky, “Evaluation of machine learning algorithms for image quality assessment,” in , 2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS). IEEE, 2016, pp. 193–194.

[7] R. Alhakim, G. T. Tchendjou, E. Simeu, and F. Lebowsky, “Image quality assessment using nonlinear learning methods,”, in 2015 27th International Conference on Microelectronics (ICM), IEEE, 2015, pp. 5–8.

Liste des symboles

AD Average Difference

AGGD Asymmetric Generalized Gaussian Distribution

ANFIS Adaptive Neuro Fuzzy Inference System

ANN Artificial Neural Network

ASIC Application-Specific Integrated Circuit

AXI Advanced eXtensible Interface

BMP BitMaP

BRISQUE Blind/Referenceless Image Spatial QUality Evaluator

CART Classification And Regression Tree

CBM Content Based Metric

CCD Charge-Coupled Device

CD-ROM Compact Disc - Read Only Memory

CHAID CHi-square Automatic Interaction Detection

CMOS Complementary Metal-Oxide-Semiconductor

CSF Contrast Sensitivity Function

CSIQ Categorical Subjective Image Quality

DA Discriminant Analysis

dCor distance Correlation

dCov distance Covariance

DCT Discrete Cosine Transform

DDR Double Data Rate

DIIVINE Distortion Identification-based Image Verity and INtegrity Evaluation

DLBQA Direct Learning Blind image Quality Assessment index

DMA Direct Memory Access

DMOS Differential Mean Opinion Score

DPD Defect Pixel Detection

DSCQS Double-Stimulus Continuous Quality-Scale

DSI DisSimilarity Index

DSP Digital Signal Processing

DT Decision Tree

EPFL École Polytechnique Fédérale de Lausanne

FF Fast Fading

FIS Fuzzy Inference System

FL Fuzzy Logic

FPGA Field Programmable Gate Array

FSIM Feature SIMilarity

FSIMc FSIM based on Color

GB Gaussian Blur

GFRIQ Global Full-Reference Image Quality index

GGD Generalized Gaussian Distribution

GM Gradient Magnitude

GMLOGQA Gradient Magnitude and Laplacian of Gaussian Quality Assessment

HEVC High Efficiency Video Coding

HLS High Level Synthesis

ID3 Inductive Decision Tree

IFC Information Fidelity Criterion

IL-NIQE Integrated Local Natural Image Quality Evaluator

ILBQA Indirect Learning Blind image Quality Assessment index

IP Intellectual Property

IQA Image Quality Assessment

ISO International Organization for Standardization

ITU International Telecommunication Union

IW-MSE Information content Weighted Mean Square Error

IW-PSNR Information content Weighted Peak Signal to Noise Ratio

IW-SSIM Information content Weighted Structural SIMilarity measure

JPEG Joint Photographic Experts Group

JPEG2000 JPEG version introduced in 2000

JTAG Joint Test Action Group

KNN K-Nearest Neighbors

KRCC Kendall Rank Correlation Coefficient

LCD Liquid Crystal Display

LIVE Laboratory for Image and Video Engineering

LOOCV Leave-One-Out Cross-Validation

LUT LookUp Table

MAD Most Apparent Distortion

MCCV Monte Carlos Cross-Validation

MD Maximum Difference

MF Membership Function

Mhz Mega hertz

MIG Memory Interface Generator

ML Machine Learning

MOS Mean Opinion Score

MPEG Moving Picture Experts Group

MSE Mean Squared Error

MSSIM Multi-scale Structural SIMilarity index

NAE Normalized Absolute Error

NCC Normalized Cross Correlation

NIQE Natural Image Quality Evaluator

NLR Non-Linear Regression

NQM Noise Quality Measure

NSS Natural Scene Statistic

PC Phase Congruency

PLCC Pearson Linear Correlation Coefficient

PNG Portable Network Graphics

PSNR Peak Signal to Noise Ratio

PSNR-HVS PSNR taking into account Human Visual System

PSNR-HVS-M PSNR-HVS taking into account visual Masking

PSNRc PSNR based on Color

QUEST Quick, Unbiased, Efficient Statistical Trees

RAM Random Access Memory

RTL Register Transfer Level

SC Structural Content

SDK Software Development Kit

SFF Sparse Feature Fidelity

SRCC Spearman Rank order Correlation Coefficient

SSCQS Single-Stimulus Continuous Quality-Scale

SSEQ Spatial–Spectral Entropy-based Quality

SSIM Structural SIMilarity

SVD Singular Value Decomposition

SVH Système Visuel Humain

SVM Support Vector Machine

TCD Transformée en Cosinus Discrète

TID Tampere Image Database

TID2008 TID release in 2008

TID2013 TID release in 2013

UART Universal Asynchronous Receiver/Transmitter

UQI Universal image Quality Index

VHDL VHSIC Hardware Description Language

VIF Visual Information Fidelity

VIFP VIF based on Pixel-wise

VSNR Visual Signal-to-Noise Ratio

WN White Noise

WSNR Weighted Signal to Noise Ratio

Bibliographie

[1] H. R. Wu, W. Lin, and L. J. Karam, "An overview of perceptual processing for digital pictures," in *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on.* IEEE, 2012, pp. 113–120.

[2] F. Crété-Roffet, "Estimer, mesurer et corriger les artefacts de compression pour la télévision numérique," Ph.D. dissertation, Université Joseph-Fourier-Grenoble I, 2007.

[3] G. J. Sullivan, J.-R. Ohm, W.-J. Han, T. Wiegand *et al.*, "Overview of the high efficiency video coding(hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[4] N. Ponomarenko, O. Ieremeiev, V. Lukin, L. Jin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti *et al.*, "A new color image database tid2013 : Innovations and results," in *International Conference on Advanced Concepts for Intelligent Vision Systems.* Springer, 2013, pp. 402–413.

[5] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment : from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[6] S. Suresh, R. V. Babu, and H. Kim, "No-reference image quality assessment using modified extreme learning machine classifier," *Applied Soft Computing*, vol. 9, no. 2, pp. 541–552, 2009.

[7] B. E. Akoa, E. Simeu, and F. Lebowsky, "Using artificial neural network for automatic assessment of video sequences," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on.* IEEE, 2013, pp. 285–290.

[8] ——, "Video decoder monitoring using non-linear regression," in *On-Line Testing Symposium (IOLTS), 2013 IEEE 19th International.* IEEE, 2013, pp. 175–178.

[9] A. Chetouani, "Vers un système d'évaluation de la qualité d'image multi-critères," Ph.D. dissertation, paris ; Université Paris-Nord-Paris XIII, 2010.

[10] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Transactions on image processing*, vol. 15, no. 2, pp. 430–444, 2006.

[11] R. Soundararajan and A. C. Bovik, "Rred indices : Reduced reference entropic differencing for image quality assessment," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 517–526, 2012.

[12] D. Liu, Y. Xu, Y. Quan, and P. Le Callet, "Reduced reference image quality assessment using regularity of phase congruency," *Signal Processing : Image Communication*, vol. 29, no. 8, pp. 844–855, 2014.

[13] A. K. Moorthy and A. C. Bovik, "Blind image quality assessment : From natural scene statistics to perceptual quality," *IEEE transactions on Image Processing*, vol. 20, no. 12, pp. 3350–3364, 2011.

[14] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, 2013.

[15] A. Mittal, A. K. Moorthy, and A. C. Bovik, "Making image quality assessment robust," in *Signals, Systems and Computers (ASILOMAR), 2012 Conference Record of the Forty Sixth Asilomar Conference on.* IEEE, 2012, pp. 1718–1722.

[16] ——, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, 2012.

[17] Z. Wang, A. C. Bovik, and L. Lu, "Why is image quality assessment so difficult ?" in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 4. IEEE, 2002, pp. IV–3313.

[18] C.-H. Chan, "Dead pixel real-time detection method for image," Sep. 15 2009, uS Patent 7,589,770.

[19] I. ISO, "13406—2 standard."

[20] S. Tourancheau, "Caractérisation objective et modélisation psychovisuelle du flou de mouvement sur les écrans à cristaux liquides-impact sur la qualité perçue," Ph.D. dissertation, Université de Nantes, 2009.

[21] J. Lassalle, "Etude de l'influence de la qualité audiovisuelle sur la qualité d'expérience du spectateur : combinaison d'indicateurs subjectifs, physiologiques et oculaires," Ph.D. dissertation, Télécom Bretagne, Université de Bretagne-Sud, 2013.

[22] U. Séries des Recommandations, "Techniques de mesure objective de la qualité vidéo pour les applications de radiodiffusion utilisant la télévision haute définition en présence d'un signal de référence réduit."

[23] H. R. Sheikh, Z. Wang, H. Huang, and A. C. Bovik, "Live image quality assessment database release 2."

[24] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Transactions on image processing*, vol. 15, no. 11, pp. 3440–3451, 2006.

[25] E. C. Larson and D. M. Chandler, "Most apparent distortion : full-reference image quality assessment and the role of strategy," *Journal of Electronic Imaging*, vol. 19, no. 1, pp. 011 006–011 006, 2010.

[26] E. Larson and D. Chandler, "Categorical subjective image quality csiq database," 2009.

[27] N. Ponomarenko, O. Ieremeiev, V. Lukin, K. Egiazarian, L. Jin, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti *et al.*, "Color image database tid2013 : Peculiarities

and preliminary results," in *Visual Information Processing (EUVIP), 2013 4th European Workshop on*. IEEE, 2013, pp. 106–111.

[28] N. Ponomarenko, F. Battisti, K. Egiazarian, J. Astola, and V. Lukin, "Metrics performance comparison for color image database," in *Fourth international workshop on video processing and quality metrics for consumer electronics*, vol. 27, 2009.

[29] N. Ponomarenko, V. Lukin, A. Zelensky, K. Egiazarian, M. Carli, and F. Battisti, "Tid2008-a database for evaluation of full-reference visual quality assessment metrics," *Advances of Modern Radioelectronics*, vol. 10, no. 4, pp. 30–45, 2009.

[30] R. Franzen, "Kodak lossless true color image suite," *source : http ://r0k.us/graphics/kodak*, vol. 4, 1999.

[31] D. M. Allen, "Mean square error of prediction as a criterion for selecting variables," *Technometrics*, vol. 13, no. 3, pp. 469–475, 1971.

[32] N. J. Nagelkerke *et al.*, "A note on a general definition of the coefficient of determination," *Biometrika*, vol. 78, no. 3, pp. 691–692, 1991.

[33] A. C. Cameron and F. A. Windmeijer, "An r-squared measure of goodness of fit for some common nonlinear regression models," *Journal of Econometrics*, vol. 77, no. 2, pp. 329–342, 1997.

[34] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*. Springer, 2009, pp. 1–4.

[35] L. Myers and M. J. Sirois, "Spearman correlation coefficients, differences between," *Wiley StatsRef : Statistics Reference Online*, 2006.

[36] R. Rakotomalala, "Analyse de corrélation : Étude des dépendances-variables quantitatives," *Document de Cours. Version*, vol. 1, 2012.

[37] H. Abdi, "The kendall rank correlation coefficient," *Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA*, pp. 508–510, 2007.

[38] L. Laurencelle, "Le tau et le tau-b de kendall pour la corrélation de variables ordinales simples ou catégorielles," *Tutorials in Quantitative Methods for Psychology*, vol. 5, no. 2, pp. 51–58, 2009.

[39] G. Leger and M. J. Barragan, "Brownian distance correlation-directed search : A fast feature selection technique for alternate test," *Integration, the VLSI Journal*, vol. 55, pp. 401–414, 2016.

[40] G. J. Székely, M. L. Rizzo *et al.*, "Brownian distance covariance," *The annals of applied statistics*, vol. 3, no. 4, pp. 1236–1265, 2009.

[41] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation," in *Encyclopedia of database systems*. Springer, 2009, pp. 532–538.

[42] Y. Bengio and Y. Grandvalet, "No unbiased estimator of the variance of k-fold cross-validation," *Journal of machine learning research*, vol. 5, no. Sep, pp. 1089–1105, 2004.

[43] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, no. 2. Montreal, Canada, 1995, pp. 1137–1145.

[44] Q.-S. Xu and Y.-Z. Liang, "Monte carlo cross validation," *Chemometrics and Intelligent Laboratory Systems*, vol. 56, no. 1, pp. 1–11, 2001.

[45] A. J. Izenman, "Modern multivariate statistical techniques," *Regression, classification and manifold learning*, 2008.

[46] G. McLachlan, *Discriminant analysis and statistical pattern recognition*. John Wiley & Sons, 2004, vol. 544.

[47] M. Chavent, *Analyse Factorielle Discriminante (AFD)*. Université de Bordeaux, 2016, vol. 14.

[48] A. Chetouani, M. Deriche, and A. Beghdadi, "Classification of image distortions using image quality metrics and linear discriminant analysis," in *Signal Processing Conference, 2010 18th European*. IEEE, 2010, pp. 319–322.

[49] A. Chetouani, A. Beghdadi, A. Bouzerdoum, and M. Deriche, "A new scheme for no reference image quality assessment," in *Image Processing Theory, Tools and Applications (IPTA), 2012 3rd International Conference on*. IEEE, 2012, pp. 270–274.

[50] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[51] L. Devroye, L. Györfi, and G. Lugosi, *A probabilistic theory of pattern recognition*. Springer Science & Business Media, 2013, vol. 31.

[52] Y. Ke, X. Tang, and F. Jing, "The design of high-level features for photo quality assessment," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 419–426.

[53] C. Li, S. Zhang, H. Zhang, L. Pang, K. Lam, C. Hui, and S. Zhang, "Using the k-nearest neighbor algorithm for the classification of lymph node metastasis in gastric cancer," *Computational and mathematical methods in medicine*, vol. 2012, 2012.

[54] M. Clergue, "Réseaux de neurones artificiels."

[55] J.-M. Renders, *Algorithmes génétiques et réseaux de neurones*. Hermès, 1994.

[56] A. Bouzerdoum, A. Havstad, and A. Beghdadi, "Image quality assessment using a neural network approach," in *Signal Processing and Information Technology, 2004. Proceedings of the Fourth IEEE International Symposium on*. IEEE, 2004, pp. 330–333.

[57] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural computation*, vol. 12, no. 5, pp. 1207–1245, 2000.

[58] B. Boster, I. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on computational learning theory*. ACM, 1992, pp. 144–152.

[59] W. Xue, X. Mou, L. Zhang, A. C. Bovik, and X. Feng, "Blind image quality assessment using joint statistics of gradient magnitude and laplacian features," *IEEE Transactions on Image Processing*, vol. 23, no. 11, pp. 4850–4862, 2014.

[60] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in neural information processing systems*, 1997, pp. 155–161.

[61] H. R. Sheikh, A. C. Bovik, and G. De Veciana, "An information fidelity criterion for image quality assessment using natural scene statistics," *IEEE Transactions on image processing*, vol. 14, no. 12, pp. 2117–2128, 2005.

[62] Y. Massiani, J.-P. Crousier, J. Crousier, J. Galea, and R. Romanetti, "Determination des parametres des courbes de polarisation a l'aide d'un programme de calcul par regression non lineaire. application a la corrosion d'un metal en milieu acide en presence d'oxygene dissous," *Electrochimica acta*, vol. 29, no. 12, pp. 1679–1683, 1984.

[63] K. Schittkowski, "Easy-fit : a software system for data fitting in dynamical systems," *Structural and Multidisciplinary Optimization*, vol. 23, no. 2, pp. 153–169, 2002.

[64] R. Khereddine, E. Simeu, and S. Mir, "Utilisation des techniques de regression pour le test et le diagnostic des composantes rf," in *Journées GDR SoC-SiP*, 2007.

[65] A. Charnes, E. L. Frome, and P.-L. Yu, "The equivalence of generalized least squares and maximum likelihood estimates in the exponential family," *Journal of the American Statistical Association*, vol. 71, no. 353, pp. 169–171, 1976.

[66] A. Bjorck, *Numerical methods for least squares problems*. Siam, 1996.

[67] M. D. Dimitrievski, Z. A. Ivanovski, and T. P. Kartalov, "No-reference image visual quality assessment using nonlinear regression," in *Quality of Multimedia Experience (QoMEX), 2011 Third International Workshop on*. IEEE, 2011, pp. 78–83.

[68] L. Rokach and O. Maimon, *Data mining with decision trees : theory and applications*. World scientific, 2014.

[69] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[70] Y. Freund and L. Mason, "The alternating decision tree learning algorithm," in *icml*, vol. 99, 1999, pp. 124–133.

[71] R. Rakotomalala, "Arbres de décision," *Revue Modulad*, vol. 33, pp. 163–187, 2005.

[72] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[73] B. Hssina, A. Merbouha, H. Ezzikouri, and M. Erritali, "A comparative study of decision tree id3 and c4. 5," *International Journal of Advanced Computer Science and Applications*, vol. 4, no. 2, 2014.

[74] N. Ahlawat, A. Gautam, N. Sharma *et al.*, "Use of logic gates to make edge avoider robot," *International Journal of Information & Computation Technology*, vol. 4, no. 6, p. 630, 2014.

[75] V. Novák, I. Perfilieva, and J. Mockor, *Mathematical principles of fuzzy logic*. Springer Science & Business Media, 2012, vol. 517.

[76] L. A. Zadeh, "Fuzzy logic= computing with words," *IEEE transactions on fuzzy systems*, vol. 4, no. 2, pp. 103–111, 1996.

[77] F. Chevrie and F. Guély, "La logique floue," *Cahier technique*, vol. 191, 1998.

[78] P. Carrai, I. Heynderickz, P. Gastaldo, and R. Zunino, "Image quality assessment by using neural networks," in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol. 5. IEEE, 2002, pp. V–V.

[79] D. Yue, X. Huang, and H. Tan, "Image quality assessment based on wavelet coefficients using neural network," in *International Symposium on Neural Networks*. Springer, 2007, pp. 853–859.

[80] S. Chen, Z. He, and P. M. Grant, "Artificial neural network visual model for image quality enhancement," *Neurocomputing*, vol. 30, no. 1-4, pp. 339–346, 2000.

[81] W. Ding, Y. Tong, Q. Zhang, and D. Yang, "Image and video quality assessment using neural network and svm," *Tsinghua Science & Technology*, vol. 13, no. 1, pp. 112–116, 2008.

[82] A. B. Watson, "Dctune : A technique for visual optimization of dct quantization matrices for individual images," in *Sid International Symposium Digest of Technical Papers*, vol. 24. Citeseer, 1993, pp. 946–946.

[83] K. Egiazarian, J. Astola, N. Ponomarenko, V. Lukin, F. Battisti, and M. Carli, "New full-reference quality metrics based on hvs," in *Proceedings of the Second International Workshop on Video Processing and Quality Metrics*, vol. 4, 2006.

[84] N. Ponomarenko, F. Silvestri, K. Egiazarian, M. Carli, J. Astola, and V. Lukin, "On between-coefficient contrast masking of dct basis functions," in *Proceedings of the third international workshop on video processing and quality metrics*, vol. 4, 2007.

[85] T. Mitsa and K. L. Varkur, "Evaluation of contrast sensitivity functions for the formulation of quality measures incorporated in halftoning algorithms," in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, vol. 5. IEEE, 1993, pp. 301–304.

[86] K. Egiazarian, M. Ponomarenko, V. Lukin, and O. Ieremeiem, "Statistical evaluation of visual quality metrics for image denoising," *arXiv preprint arXiv :1711.00693*, 2017.

[87] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "Fsim : A feature similarity index for image quality assessment," *IEEE transactions on Image Processing*, vol. 20, no. 8, pp. 2378–2386, 2011.

[88] Z. Wang and Q. Li, "Information content weighting for perceptual image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1185–1198, 2011.

[89] N. Ponomarenko, L. Jin, V. Lukin, and K. Egiazarian, "Self-similarity measure for assessment of image visual quality," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2011, pp. 459–470.

[90] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, vol. 2. Ieee, 2003, pp. 1398–1402.

[91] N. Damera-Venkata, T. D. Kite, W. S. Geisler, B. L. Evans, and A. C. Bovik, "Image quality assessment based on a degradation model," *IEEE transactions on image processing*, vol. 9, no. 4, pp. 636–650, 2000.

[92] N. Ponomarenko, O. Ieremeiev, V. Lukin, K. Egiazarian, and M. Carli, "Modified image visual quality metrics for contrast change and mean shift accounting," in *CAD Systems in Microelectronics (CADSM), 2011 11th International Conference The Experience of Designing and Application of*. IEEE, 2011, pp. 305–311.

[93] H.-W. Chang, H. Yang, Y. Gan, and M.-H. Wang, "Sparse feature fidelity for perceptual image quality assessment," *IEEE Transactions on Image Processing*, vol. 22, no. 10, pp. 4007–4018, 2013.

[94] L. Zhang and H. Li, "Sr-sim : A fast and high performance iqa index based on spectral residual," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 1473–1476.

[95] D. M. Chandler and S. S. Hemami, "Vsnr : A wavelet-based visual signal-to-noise ratio for natural images," *IEEE transactions on image processing*, vol. 16, no. 9, pp. 2284–2298, 2007.

[96] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 2, no. 4, pp. 81–84, 2002.

[97] C. Fernandez-Maloigne, *Advanced color image processing and analysis*. Springer Science & Business Media, 2012.

[98] A. Shnayderman, A. Gusev, and A. M. Eskicioglu, "An svd-based grayscale image quality measure for local and global assessment," *IEEE transactions on Image Processing*, vol. 15, no. 2, pp. 422–429, 2006.

[99] P. Kovesi, "Image features from phase congruency," *Videre : Journal of computer vision research*, vol. 1, no. 3, pp. 1–26, 1999.

[100] R. Jain, R. Kasturi, and B. G. Schunck, *Machine vision*. McGraw-Hill New York, 1995, vol. 5.

[101] C. C. Yang and S. H. Kwok, "Efficient gamut clipping for color image processing using lhs and yiq," *Optical Engineering*, vol. 42, no. 3, pp. 701–712, 2003.

[102] F. De Simone, M. Naccari, M. Tagliasacchi, F. Dufaux, S. Tubaro, and T. Ebrahimi, "Subjective assessment of h. 264/avc video sequences transmitted over a noisy channel," in *Quality of Multimedia Experience, 2009. QoMEx 2009. International Workshop on*. IEEE, 2009, pp. 204–209.

[103] F. De Simone, M. Tagliasacchi, M. Naccari, S. Tubaro, and T. Ebrahimi, "A h. 264/avc video database for the evaluation of quality metrics," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2430–2433.

[104] L. De Branges, "The stone-weierstrass theorem," *Proceedings of the American Mathematical Society*, vol. 10, no. 5, pp. 822–824, 1959.

[105] J.-S. Jang and C.-T. Sun, "Neuro-fuzzy modeling and control," *Proceedings of the IEEE*, vol. 83, no. 3, pp. 378–406, 1995.

[106] F.-J. Chang and Y.-T. Chang, "Adaptive neuro-fuzzy inference system for prediction of water level in reservoir," *Advances in water resources*, vol. 29, no. 1, pp. 1–10, 2006.

[107] J.-S. Jang, "Anfis : adaptive-network-based fuzzy inference system," *IEEE transactions on systems, man, and cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[108] Y. H. Joo and G. Chen, "Fuzzy systems modeling : an introduction," in *Encyclopedia of artificial intelligence*. IGI Global, 2009, pp. 734–743.

[109] L. K. PinPin, D. F. T. Gamarra, C. Laschi, and P. Dario, "Topology based fuzzy clustering for robust anfis creation," in *Cybernetic Intelligent Systems, 2008. CIS 2008. 7th IEEE International Conference on*. IEEE, 2008, pp. 1–6.

[110] B. Fritzke, "Incremental neuro-fuzzy systems," in *Applications of Soft Computing*, vol. 3165. International Society for Optics and Photonics, 1997, pp. 86–98.

[111] T. Brandão and M. P. Queluz, "No-reference image quality assessment based on dct domain statistics," *Signal Processing*, vol. 88, no. 4, pp. 822–833, 2008.

[112] P. Marziliano, F. Dufaux, S. Winkler, and T. Ebrahimi, "Perceptual blur and ringing metrics : application to jpeg2000," *Signal processing : Image communication*, vol. 19, no. 2, pp. 163–172, 2004.

[113] L. Meesters and J.-B. Martens, "A single-ended blockiness measure for jpeg-coded images," *Signal Processing*, vol. 82, no. 3, pp. 369–387, 2002.

[114] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas, "The blur effect : perception and estimation with a new no-reference perceptual blur metric," in *Human vision and electronic imaging XII*, vol. 6492. International Society for Optics and Photonics, 2007, p. 64920I.

[115] P. Ye and D. Doermann, "No-reference image quality assessment using visual codebooks," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3129–3138, 2012.

[116] L. Zhang, L. Zhang, and A. C. Bovik, "A feature-enriched completely blind image quality evaluator," *IEEE Transactions on Image Processing*, vol. 24, no. 8, pp. 2579–2591, 2015.

[117] L. Liu, B. Liu, H. Huang, and A. C. Bovik, "No-reference image quality assessment based on spatial and spectral entropies," *Signal Processing : Image Communication*, vol. 29, no. 8, pp. 856–863, 2014.

[118] C. Charrier, "Modélisation statistique et classification par apprentissage pour la qualité des images," Ph.D. dissertation, Université de Caen, 2011.

[119] D. L. Ruderman, "The statistics of natural images," *Network : computation in neural systems*, vol. 5, no. 4, pp. 517–548, 1994.

[120] E. P. Simoncelli and B. A. Olshausen, "Natural image statistics and neural representation," *Annual review of neuroscience*, vol. 24, no. 1, pp. 1193–1216, 2001.

[121] H. Le Borgne, "Analyse de scènes naturelles par composantes indépendantes," Ph.D. dissertation, Institut National Polytechnique de Grenoble-INPG, 2004.

[122] N.-E. Lasmar, Y. Stitou, and Y. Berthoumieu, "Multiscale skewed heavy tailed model for texture analysis," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE, 2009, pp. 2281–2284.

[123] J.-M. Geusebroek and A. W. Smeulders, "A six-stimulus theory for stochastic texture," *International Journal of Computer Vision*, vol. 62, no. 1-2, pp. 7–16, 2005.

[124] J. Huang and D. Mumford, "Statistics of natural images and models," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference On.*, vol. 1. IEEE, 1999, pp. 541–547.

[125] J. Sponring, "The entropy of scale-space," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 1. IEEE, 1996, pp. 900–904.

[126] A. K. Moorthy and A. C. Bovik, "Visual importance pooling for image quality assessment," *IEEE journal of selected topics in signal processing*, vol. 3, no. 2, pp. 193–201, 2009.

[127] M. A. Saad, A. C. Bovik, and C. Charrier, "Blind image quality assessment : A natural scene statistics approach in the dct domain," *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3339–3352, 2012.

[128] J. An, W. Lee, and J. Kim, "Adaptive detection and concealment algorithm of defective pixel," in *Signal Processing Systems, 2007 IEEE Workshop on*. IEEE, 2007, pp. 651–656.

[129] J. Dudas, C. Jung, G. H. Chapman, Z. Koren, and I. Koren, "Robust detection of defects in imaging arrays," in *Image Quality and System Performance III*, vol. 6059. International Society for Optics and Photonics, 2006, p. 60590X.

[130] Y.-P. Tan and T. Acharya, "A robust sequential approach for the detection of defective pixels in an image sensor," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 4. IEEE, 1999, pp. 2239–2242.

[131] J. S. J. Li and S. Randhawa, "Adaptive order-statistics multi-shell filtering for bad pixel correction within cfa demosaicking," in *TENCON 2009-2009 IEEE Region 10 Conference*. IEEE, 2009, pp. 1–6.

[132] S. Ghosh, D. Froebrich, and A. Freitas, "Robust autonomous detection of the defective pixels in detectors using a probabilistic technique," *Applied optics*, vol. 47, no. 36, pp. 6904–6924, 2008.

[133] G. Meynants and B. Dierickx, "A circuit for the correction of pixel defects in image sensors," in *Solid-State Circuits Conference, 1998. ESSCIRC'98. Proceedings of the 24th European.* IEEE, 1998, pp. 312–315.

[134] A. Bosco, A. Bruna, F. Naccari, and I. Guarneri, "Hardware/software solution for high precision defect correction in digital image sensors," in *Consumer Electronics, 2008. ISCE 2008. IEEE International Symposium on.* IEEE, 2008, pp. 1–4.

[135] J. Dudas, C. Jung, L. Wu, G. H. Chapman, I. Koren, and Z. Koren, "On-line mapping of in-field defects in image sensor arrays," in *Defect and Fault Tolerance in VLSI Systems, 2006. DFT'06. 21st IEEE International Symposium on.* IEEE, 2006, pp. 439–447.

[136] C.-Y. Cho, T.-M. Chen, W.-S. Wang, and C.-N. Liu, "Real-time photo sensor dead pixel detection for embedded devices," in *Digital Image Computing Techniques and Applications (DICTA), 2011 International Conference on.* IEEE, 2011, pp. 164–169.

[137] C.-W. Chen, C.-Y. Cho, Y.-F. Sun, T.-M. Chen, and C.-L. Su, "Low complexity photo sensor dead pixel detection algorithm," in *Circuits and Systems (APCCAS), 2012 IEEE Asia Pacific Conference on.* IEEE, 2012, pp. 360–363.

[138] A. K. Akobeng, "Understanding diagnostic tests 1 : sensitivity, specificity and predictive values," *Acta paediatrica*, vol. 96, no. 3, pp. 338–341, 2007.

[139] W. Leisenring, T. Alono, and M. S. Pepe, "Comparisons of predictive values of binary medical diagnostic tests for paired designs," *Biometrics*, vol. 56, no. 2, pp. 345–351, 2000.

[140] C. E. Jurgensen, "Table for determining phi coefficients," *Psychometrika*, vol. 12, no. 1, pp. 17–29, 1947.

[141] B. E. Akoa, "Détection et conciliation d'erreurs intégrées dans un décodeur vidéo : utilisation des techniques d'analyse statistique," Ph.D. dissertation, Université de Grenoble, 2014.

CONTRÔLE DES PERFORMANCES ET CONCILIATION D'ERREURS DANS LES DECODEURS D'IMAGE

Résumé : Cette thèse porte sur le développement et l'implémentation des algorithmes de détection et de correction des erreurs dans les images, en vue de contrôler la qualité des images produites en sortie des décodeurs numériques. Pour atteindre les objectifs visés dans cette étude, nous avons commencé par faire l'état de lieu de l'existant. L'examen critique des approches en usage a justifié la construction d'un ensemble de méthodes objectives d'évaluation de la qualité visuelle des images, basées sur des méthodes d'apprentissage automatique. Ces algorithmes prennent en entrées un ensemble de caractéristiques ou de métriques extraites des images. En fonction de ces caractéristiques, et de la disponibilité ou non d'une image de référence, deux sortes de mesures objectives ont été élaborées : la première basée sur des métriques avec référence, et la seconde basée sur des métriques sans référence ; toutes les deux à distorsions non spécifiques. En plus de ces méthodes d'évaluation objective, une méthode d'évaluation et d'amélioration de la qualité des images basée sur la détection et la correction des pixels défectueux dans les images a été mise en œuvre. Les applications ont contribué à affiner aussi bien les méthodes d'évaluation de la qualité visuelle des images que la construction des algorithmes objectifs de détection et de correction des pixels défectueux par rapport aux diverses méthodes actuellement en usage. Une implémentation sur cartes FPGA des techniques développées a été réalisée pour intégrer les modèles présentant les meilleures performances dans de la phase de simulation.

Mots clés : Qualité visuelle, évaluation objective, implémentation FPGA, apprentissage automatique, logique floue, pixel défectueux, conciliation d'erreur.

PERFORMANCE MONITORING AND ERRORS RECONCILIATION IN IMAGE DECODERS

Abstract: This thesis deals with the development and implementation of error detection and correction algorithms in images, in order to control the quality of produced images at the output of digital decoders. To achieve the objectives of this work, we first study the state-of the-art of the existing approaches. The examination of classically used approaches justified the study of a set of objective methods for evaluating the visual quality of images, based on machine learning methods. These algorithms take as inputs a set of characteristics or metrics extracted from the images. Depending on these characteristics and the availability or not of a reference image, two kinds of objective evaluation methods have been developed: the first one is based on full reference metrics, and the second approach is based on no-reference metrics, both of them with non-specific distortions. In addition to these objective evaluation methods, a method of evaluating and improving the quality of the images based on the detection and correction of the defective pixels in the images has been implemented. The proposed results have contributed to refining visual image quality assessment methods as well as the construction of objective algorithms for detecting and correcting defective pixels compared to the various currently used methods. An implementation on an FPGA has been carried out to integrate the models with the best performances during the simulation phase.

Keywords: Visual quality, objective evaluation, FPGA implementation, machine learning, fuzzy logic, defective pixel, error concealment.

Thèse préparée au laboratoire TIMA (Techniques de l'Informatique et de la Microélectronique pour l'Architecture des ordinateurs), 46 Avenue Félix Viallet, 38031, Grenoble Cedex, France

ISBN: 978-2-11-129251-2