Méthodes d'apprentissage automatique pour la prise en compte du bruit dans les images de synthèse

Jérôme Buisine

Thèse de Doctorat

Mention : Sciences et Technologies de l'Information et de la Communication
Spécialité: Informatique et applications

présentée à *l'Ecole Doctorale en Sciences Technologie et Santé (ED 585)*

de l'Université du Littoral Côte d'Opale

par

Jérôme Buisine

pour obtenir le grade de Docteur de l'Université du Littoral Côte d'Opale

Méthodes d'apprentissage automatique pour la prise en compte du bruit dans les images de synthèse

Soutenue le 8 décembre 2021, après avis des rapporteurs, devant le jury d'examen :

M. Sébastien VEREL, Professeur, Univ. du Littoral Côte d'Opale	**Président**
M^me^ Laëtitia JOURDAN, Professeure, Univ. de Lille	**Rapporteure**
M. Daniel MENEVEAUX, Professeur, Univ. de Poitiers	**Rapporteur**
M. Philippe PREUX, Professeur, Univ. de Lille	**Examinateur**
M. Matéu SBERT, Professeur, Univ. de Gérone	**Examinateur**
M. Christophe RENAUD, Professeur, Univ. du Littoral Côte d'Opale	**Directeur de thèse**
M. Samuel DELEPOULLE, Maître de Conférences, Univ. du Littoral Côte d'Opale	**Co-encadrant**

Chaque homme doit décider s'il marchera dans la lumière de l'altruisme créatif ou dans les ténèbres de l'égoïsme destructeur.

Martin Luther King

MÉTHODES D'APPRENTISSAGE AUTOMATIQUE POUR LA PRISE EN COMPTE DU BRUIT DANS LES IMAGES DE SYNTHÈSE

Résumé

Les méthodes de simulation de l'éclairage, utilisées en synthèse d'images, permettent d'obtenir des vues dites photo-réalistes d'environnements virtuels 3D. Pour ce faire, elles utilisent des méthodes stochastiques, s'appuyant sur la théorie des grands nombres, qui explorent l'espace des chemins lumineux et se caractérisent par une convergence progressive de l'image vers la solution. Cette progressivité se traduit visuellement par la présence de bruit, qui se résorbe progressivement au fur et à mesure de l'avancée des calculs. Ce bruit doit être identifié et quantifié, afin de disposer de critères perceptifs permettant d'arrêter les algorithmes dans les différentes zones de l'image. Ceci est d'autant plus important que les temps de calcul d'une image se comptent en heures, voire en dizaines d'heures de calcul. Disposer de critères fiables pour arrêter les calculs en différents points d'une image permettrait donc de réaliser des gains de temps importants. Dans cette thèse, nous proposons d'utiliser des méthodes statistiques et d'apprentissage automatique pour la réduction et détection de ce bruit généré. Les contributions réalisées dans le cadre de cette thèse sont : (i) la constitution d'une base d'images de synthèse avec recueil de seuils subjectifs humains du bruit résiduel, (ii) l'étude et la gestion d'un bruit local hautement perceptible, (iii) la création de modèles d'apprentissage profond sur cette base d'images étiquetées et (iv) une phase de validation des images reconstruites obtenues (apprises ou non) à partir des modèles de perception à partir d'évaluations subjectives. Des travaux connexes à la thématique de la thèse, notamment relatifs à la gestion d'un bruit spécifique dans les images nommé « firefly », ont été proposés, tout comme l'application d'une méthode permettant de cibler les caractéristiques de bruit étudiées.

Mots clés : simulation d'éclairage, perception visuelle, apprentissage automatique, classification

Abstract

Lighting simulation methods, used in image synthesis, make it possible to obtain so-called photorealistic views of 3D virtual environments. To do this, they use stochastic methods, based on the theory of large numbers, which explore the space of light paths and are characterised by a progressive convergence of the image towards the solution. This progressiveness is visually expressed by the presence of noise, which is gradually reduced as the calculations progress. This noise must be identified and quantified in order to have perceptual criteria for stopping the algorithms in the different areas of the image. This is all the more important as the calculation time for an image can be counted in hours or even tens of hours. Having reliable criteria to stop the calculations at different points of an image would therefore allow significant time savings. In this thesis, we propose to use statistical and machine learning methods for the reduction and detection of this generated noise. The contributions made in the framework of this thesis are: (i) the constitution of a base of synthetic images with the collection of human subjective thresholds of the residual noise, (ii) the study and management of a highly perceptible local noise, (iii) the creation of deep learning models on this base of labelled images and (iv) a phase of validation of the reconstructed images obtained (learned or not) from the models of perception from subjective evaluations. Related work to the thesis research area, notably concerning the management of a specific noise in images called « firefly », has been proposed, as well as the application of a method allowing the targeting of the noise characteristics studied.

Keywords: lighting simulation, visual perception, machine learning, classification

Remerciements

Tout d'abord, je souhaiterais remercier mes deux encadrants de thèse : Christophe Renaud et Samuel Delepoulle pour leurs conseils et temps consacrés tout le long de cette thèse. Ils ont su m'orienter dans mes travaux de recherche et me permettre de rester positif face aux difficultés.

J'aimerais également remercier mes co-auteurs et partenaires de recherche : André Bigand, Rémi Synave, Fabien Teytaud, Vasiliki Myrodia et Laurent Madelain avec qui j'ai pu avoir le plaisir de collaborer sur différents travaux. Je remercie également l'ensemble des membres de mon équipe, mais également du laboratoire de Calais. Chaque échange a permis l'avancée de mon travail, car c'est au travers de discussions informelles qu'une idée surgie. Je souhaite remercier également Daniel Meneveaux et Philippe Preux pour avoir accepté de rapporter ce document, et plus globalement tous les membres du jury pour avoir accepté d'examiner mes travaux de thèse.

Je souhaite remercier le service de Calculco proposé par l'Université, qui m'a permis de mener mes travaux de recherche dans les meilleures conditions lors de besoins en temps de calculs conséquents.

Enfin, je remercie tous les membres de ma famille et amis qui m'ont épaulés durant ces 3 années de thèse, et plus particulièrement Tania qui a su faire preuve de patience et trouver les bons mots pour m'apporter davantage de soutien durant les moments plus délicats.

Table des matières

Résumé **iii**
Remerciements iv

Table des matières **v**

Acronymes **xii**

Liste des tableaux **xvii**

Table des figures **xix**

Introduction **1**
Contexte 2
Problématique et objectifs 2
Plan 3
Contributions et publications 4

I État de l'art **5**

1 Le rendu d'images photo-réaliste **6**
1.1 Le transport de la lumière 7
1.1.1 Quantités radiométriques 8
1.1.1.1 Flux 9
1.1.1.2 Éclairement 9
1.1.1.3 Luminance 9
1.1.2 Interaction de la lumière avec les matériaux 10
1.1.3 Processus d'intégration et équation de rendu 12
1.1.3.1 Définition de l'équation de rendu 12
1.1.3.2 Reformulation surfacique 12
1.1.3.3 L'espace des chemins 13

1.2 Intégration par méthode de Monte-Carlo 14
1.2.1 Formulation générale 14
1.2.1.1 Propriétés de l'estimateur 15
1.2.1.2 Dimension infinie d'intégration 16
1.2.1.3 Stratégies d'échantillonnage 16
1.2.2 Échantillonnage préférentiel 17
1.2.2.1 Formulation générale 17
1.2.2.2 Stratégies d'échantillonnage préférentiel 18
1.2.2.3 Échantillonnage préférentiel multiple 19
1.3 Méthodes d'éclairage global 21
1.3.1 Tracé de chemins 21
1.3.1.1 Tracé de chemins classique 21
1.3.1.2 Tracé de chemins par estimation du pas suivant . . . 22
1.3.2 Tracé de chemins bidirectionnel 23

2 Apports statistiques pour l'amélioration du rendu 27
2.1 Méthodes de reconstruction dans l'espace image 28
2.2 Reconstruction et échantillonnage adaptatif multidimensionnels . . . 31
2.3 Rendu adaptatif de Monte-Carlo dans l'espace image 32

3 Vers l'utilisation de l'apprentissage automatique pour le rendu 38
3.1 Apprentissage automatique 39
3.1.1 Définition 39
3.1.2 Types d'apprentissage 40
3.1.2.1 Apprentissage supervisé 40
3.1.2.2 Apprentissage non supervisé 40
3.1.2.3 Apprentissage semi-supervisé 40
3.1.2.4 Apprentissage par renforcement 41
3.1.2.5 Apprentissage par transfert 41
3.1.3 Score de performance d'un modèle 41
3.1.3.1 Classification 41
3.1.3.2 Régression 42
3.1.4 Surapprentissage et généralisation 44
3.1.4.1 Surapprentissage 44
3.1.4.2 Validation croisée 45
3.1.5 Machines à vecteurs de support 45
3.1.5.1 Définition et idée 46
3.1.5.2 Notion de marge maximale 46
3.1.5.3 Cas non séparable 49
3.1.5.4 Marge souple 50
3.1.6 Réseau neuronal 51

3.1.6.1 Perceptron 51
3.1.6.2 Perceptrons multicouches 52
3.1.6.3 Fonction de perte 52
3.1.6.4 Réseau convolutif 53
3.1.6.5 AutoEncoder 54
3.1.6.6 Réseau de neurones récurrents 55
3.1.6.7 Fonction d'activation 56
3.2 Méthodes de reconstruction avec apprentissage automatique pour le rendu 58
3.2.1 Réseaux multicouches pour la prédiction des pixels 58
3.2.2 Réseaux convolutifs pour le débruitage 59
3.2.3 Utilisation des réseaux GAN 62

4 Problématique de la perception du bruit et mesures de qualité 66
4.1 Propriétés de la perception visuelle humaine 67
4.1.1 Le Système Visuel Humain 68
4.1.2 Adaptation visuelle 68
4.1.2.1 Gamme de luminance 69
4.1.2.2 Les cônes 69
4.1.2.3 Les bâtonnets 70
4.1.2.4 Temps d'adaptation 70
4.1.3 Sensibilité CSF 70
4.1.4 Autres propriétés visuelles 72
4.1.4.1 Acuité 72
4.1.4.2 Masquage 73
4.2 Modèles d'attention et différences 73
4.2.1 Carte de saillances 73
4.2.1.1 Définition et origine 74
4.2.1.2 Autres modèles 75
4.2.2 Modèles de différences 75
4.2.2.1 Prédicteur de Différence Visuelle 75
4.2.2.2 Modèle de Discrimination Visuelle 76
4.2.3 Métriques de qualité d'images 76
4.2.3.1 Avec référence 77
4.2.3.2 Sans référence 78
4.2.3.3 Métrique intégrée dans un contexte de rendu 79
4.3 Détection de bruit dans les images de synthèse 80
4.3.1 Procédure d'intégration 80
4.3.2 Propositions de critère d'arrêt 81

II Critère d'arrêt et méthode de réduction du bruit 87

5 Base d'images photo-réalistes 88
5.1 Choix de paramètres de génération . . . 89
5.1.1 Moteur de rendu . . . 89
5.1.2 Choix des paramètres . . . 89
5.1.3 Variété des scènes . . . 90
5.2 Format des images . . . 91
5.2.1 Nouvelle extension de fichier . . . 91
5.2.2 Génération des images d'expérimentation . . . 91
5.3 Processus de recueil de seuils subjectifs . . . 92
5.3.1 Présentation de l'application . . . 93
5.3.2 Seuils subjectifs obtenus . . . 94
5.3.3 Labellisation des blocs d'images . . . 95
5.3.4 Ouverture de la base d'images . . . 96

6 Gestion des valeurs aberrantes lors du rendu 98
6.1 Problème d'apparition de *firefly* . . . 99
6.1.1 Rappel d'échantillonnage de MC . . . 99
6.1.2 Visualisation du problème . . . 99
6.1.3 Origine des apparitions de *fireflies* . . . 100
6.2 Travaux relatifs . . . 101
6.2.1 Approches statistiques pour la suppression des valeurs aberrantes 101
6.2.2 Suppression des *fireflies* . . . 102
6.2.3 *Fireflies* dans les intégrateurs basés sur le tracé de chemins . . 103
6.3 MoN dans le rendu d'images de synthèse . . . 103
6.3.1 Définition . . . 104
6.3.2 Étude du comportement de MoN . . . 104
6.3.3 Choix dynamique du paramètre M . . . 105
6.4 MoN adaptatif avec le coefficient de Gini . . . 107
6.4.1 Coefficient de Gini . . . 107
6.4.2 Choix binaire de l'estimateur . . . 109
6.4.3 MoN adaptatif . . . 109
6.5 Comparaisons et résultats . . . 110
6.5.1 Configuration d'expérimentation . . . 110
6.5.2 Étude de convergence . . . 111
6.5.3 Impact visuel des estimateurs . . . 114
6.5.4 Comparaisons non locales . . . 115
6.5.5 Coût de calcul . . . 117
6.5.6 Coût mémoire . . . 117
6.6 Discussion . . . 118

7 Modèle de perception du bruit résiduel de Monte-Carlo 121
7.1 Problématique de dimensionnalité . . . 122
7.1.1 Complexité d'apprentissage des SVM . . . 122
7.1.2 Reproductibilité des travaux précédents . . . 122
7.2 Réseau de neurones récurrents et mesure de l'entropie appliquée à la SVD . . . 123
7.2.1 Mesure de la *SVD-Entropy* . . . 124
7.2.1.1 Aperçu de la décomposition en valeurs singulières (SVD) 124
7.2.1.2 Caractéristiques du bruit basées sur la SVD . . . 125
7.2.1.3 Mesure de l'entropie appliquée à la SVD . . . 126
7.2.2 La méthode proposée . . . 128
7.2.2.1 Réseau de neurones récurrents . . . 128
7.2.2.2 Paramètres du RNN . . . 130
7.2.2.3 Extraction des caractéristiques . . . 130
7.2.3 Résultats expérimentaux . . . 132
7.2.3.1 Les paramètres de la méthode . . . 132
7.2.3.2 Validation des paramètres . . . 133
7.2.3.3 Comparaisons avec la littérature . . . 136
7.2.3.4 Simulation d'éclairage assistée par un modèle de perception . . . 137
7.2.3.5 Reconstruction d'images . . . 140
7.2.4 Discussion . . . 144

8 Génération automatique de caractéristiques de bruit 146
8.1 Définition du problème . . . 147
8.2 Caractérisation du bruit MC . . . 147
8.3 *Guided-Generative Network* . . . 148
8.3.1 *Denoising Autoencoder* . . . 149
8.3.2 *Feature Map Generator* . . . 151
8.3.3 Discriminateur pour la classification binaire . . . 152
8.4 Apprentissage et résultats . . . 153
8.4.1 Configuration d'expérience . . . 153
8.4.2 Résultats obtenus . . . 154
8.4.3 Comparaisons des modèles . . . 155
8.5 Discussion . . . 157

9 Problème d'optimisation pour la sélection d'attributs 160
9.1 Vers un problème d'optimisation . . . 161
9.1.1 Méthodes de sélection d'attributs . . . 161
9.1.2 Considération d'un espace de recherche . . . 162
9.1.3 Représentation d'une solution . . . 163

9.1.4 Définition analytique du problème 164
9.1.5 Métaheuristiques 165
9.1.6 Opérateurs de variation 165
9.1.7 Recherche de solutions : compromis exploration et exploitation 166
9.2 Optimisation assistée par méta-modèle 167
9.2.1 Définition d'un modèle de substitution (ou surrogate) 168
9.2.2 Choix du modèle de substitution 169
9.3 Procédure expérimentale 170
9.3.1 Définition des attributs étudiés 171
9.3.2 La sélection d'attributs par l'optimisation assistée par méta-modèle 172
9.3.3 Formulation de la métaheuristique assistée par méta-modèle . 173
9.4 Résultats et comparaisons 176
9.4.1 Jeux de paramètres testés 176
9.4.2 Résultats des expérimentations 176
9.4.2.1 Comparaisons des performances 176
9.4.2.2 Apprentissage du méta-modèle 178
9.4.2.3 Comparaisons des temps d'exécutions 179
9.4.2.4 Attributs de bruit identifiés par la méthode 180
9.4.3 Comparaisons à des méthodes existantes 182
9.5 Discussion 184

10 Validations des modèles **187**
10.1 Analyse des détections erronées 187
10.2 Expérience de validation 190
10.2.1 Paramètres et configuration de l'expérience 190
10.2.2 Analyse des résultats 192
10.3 Prédiction de nouveaux seuils 195
10.4 Discussion 197

Conclusion et perspectives **199**
Extension de la base d'images et benchmark des modèles de perception du bruit 199
Étude des estimateurs pour la gestion des *fireflies* 201
Extensions des modèles de perception de bruit 202
Sélection d'attributs par l'optimisation assistée par méta-modèle 203

Bibliographie **205**

Annexes **224**

A Base d'images de synthèse **225**
A.1 Seuils subjectifs des participants de l'expérience 225
A.2 Écart-type des seuils des participants de l'expérience 226
A.3 Coefficient de variation de l'expérience 227
A.4 Variété des scènes de la base d'images de synthèse 228

B Gestion des *fireflies* **229**
B.1 Algorithme d'application de MoN lors du rendu d'images 229
B.2 Reformulation du coefficient de Gini 230

C Modèle de perception avec *SVD-Entropy* **231**
C.1 Résultats modèles RNN avec *SVD-Entropy* 231
C.2 Erreur critique globale des modèles RNN 232
C.3 Images les plus bruitées pour 6 points de vue utilisés pour l'apprentissage de modèle RNN 233
C.4 Images de référence pour 6 points de vue utilisés pour l'apprentissage de modèle RNN 234
C.5 Images les plus bruitées pour 4 points de vue non utilisés pour l'apprentissage 235
C.6 Images de référence pour 4 points de vue non utilisés pour l'apprentissage 236

Acronymes

ACI analyse en composantes indépendantes. 202

ACP Analyse en composantes principales. 202

ANR Agence Nationale de la Recherche. 2, 96, 202

AUC ROC l'aire sous la courbe ROC (*receiver operating characteristic)*. xvii, xxii, 42, 133, 134, 135, 136, 137, 138, 155, 156, 163, 164, 172, 174, 177, 178, 179, 180, 182, 183, 184, 188, 189

BDPT *Bidirectional Path Tracing* en anglais, est une méthode de rendu d'images de synthèse où les rayons lumineux sont à la fois émis depuis la caméra et la source de lumière. 23, 25, 89, 100, 103, 106, 110, 111, 201

BRDF *Bidirectional Reflectance Distribution Function* en anglais, est une fonction modélisant l'interaction lumineuse sur des matériaux réflectifs. xix, 10

BSDF *Bidirectional Scattering Distribution Function* en anglais, est une fonction plus généralisée modélisant l'interaction lumineuse sur des matériaux à la fois réflectifs et transmissifs. 10, 11, 13, 18, 20, 21, 22

BSVD *Block based SVD* en anglais, est une méthode exploitant la SVD sur des sous-blocs d'informations. 125

BTDF *Bidirectional Transmittance Distribution Function* en anglais, est une fonction modélisant l'interaction lumineuse sur des matériaux transmissifs. xix, 10

CDF *Cumulative Distribution Function* en anglais, est la fonction de distribution de X, évaluée en x, est la probabilité que X prenne une valeur inférieure ou égale à x. 17, 18

CIE La Commission internationale de l'éclairage est une organisation internationale dédiée à la lumière, l'éclairage, la couleur et les espaces de couleur. 67

CNN *Convolutional Neural Network* en anglais, représenté un type de réseau de neurones artificiels acycliques. 53, 57, 59, 60, 62

CPU *Central Processing Unit* en anglais, est un composant présent dans de nombreux dispositifs électroniques qui exécute les instructions machine des programmes informatiques. 89, 179, 199

CSF *Contrast Sensibility Function* en anglais, est une fonction de mesure de la sensibilité aux contrastes en fonction des fréquences spatiales. xx, 68, 70, 71, 72, 75, 76, 77, 78

EMA L'erreur moyenne absolue (en anglais MAE, pour *Mean Absolute Error*). 42, 61, 63, 150

EQM L'erreur quadratique moyenne (MSE en anglais, pour *Mean Square Error*). 33, 43, 53, 58, 60, 77, 150

EXR *OpenEXR* est un format de fichier, destiné à stocker des images de haute qualité (HDR). 91

FFT *Fast Fourier Transform* en anglais, fait référence à la méthode de transformée de Fourier rapide. 75

FN Faux négatifs. 41

FP Faux positifs. 41

GAN *Generative Adversarial Network* en anglais, est un modèle qui va être composé de deux réseaux de neurones s'affrontant l'un l'autre.. xiii, 55, 62, 64, 147

GGN *Guided-Generative Network* en anglais, est une architecture proposée de réseau de neurones opposée à celle du GAN. xvii, xxii, 146, 148, 153, 155, 156, 157, 158, 159, 184, 185, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 202

GPU *Graphics Processing Unit* en anglais, est une unité de calcul, pouvant être présente sous forme de circuit intégré (ou puce) sur une carte graphique. 75, 89, 111, 117, 154, 199

HDR *High Dynamic Range* en anglais, représente un format d'image à grande gamme de luminance. xiii, 61, 77, 91

HVS *Human Visual System* en anglais, représente le système visuel humain qui est l'ensemble des organes participant à la perception visuelle humaine, de la rétine au système sensori-moteur. 66, 68, 75, 76, 77, 111, 190

IMAP Équipe Image et Apprentissage. 93

JND *Just Noticeable Difference* en anglais, désigne la différence perceptible entre deux éléments proches. 76, 94

KDE *Kernel Density Estimation* en anglais, l'estimation par noyau est une méthode non-paramétrique d'estimation de la densité de probabilité d'une variable aléatoire. 40

LISIC Laboratoire d'Informatique Signal et Image de la Côte d'Opale. 93

LSTM *Long Short-Term Memory* en anglais, est une cellule spécifique pour un réseau de nerones récurrent. xx, xxii, 56, 57, 129, 130, 133, 155, 156, 157

Macop *Minimalist And Customisable Optimisation Package* est un package Python permettant la résolution de problème d'optimisation tout en se basant sur une structure générique de l'algorithme de résolution. 174, 176

MC Monte-Carlo, désigne une famille de méthodes algorithmiques visant à calculer une valeur numérique approchée en utilisant des procédés aléatoires. viii, 14, 16, 27, 66, 98, 99, 106, 124, 126, 127, 145, 146, 147, 148, 157, 160, 181, 186, 187, 200, 201, 202, 203

MCMC *Monte-Carlo Markov Chain* en anglais, représente une méthode orientée Monte-Carlo par chaîne de Markov. 103

MIS *Multiple Importance Sampling* en anglais, fait référence à un échantillonnage préférentiel multiple d'une fonction. 20, 23

MLT *Metropolis Light Transport* en anglais, est une méthode de rendu d'images de synthèse basée sur l'algorithme de Métropolis pour la construction des chemins lumineux. 89, 103, 201

MoN *Median of meaNs* en anglais, représente un estimateur calculé par la médiane des moyennes. 29, 98, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 114, 115, 116, 117, 120, 201, 229

MOS *Mean Opinion Score* en anglais, est un score d'opinion moyen obtenu à partir d'une population. 76, 78, 79, 94

NFM *Noise Feature Map* en anglais, est une carte de caractéristiques. 147, 148, 151, 152, 153, 154, 155

NMF Non-negative matrix factorization. 202

PBRT *Physically Based Rendering* en anglais, est un moteur de rendu d'images photo-réaliste. xx, 89, 90, 91, 92, 110, 116, 199

PCA *Principal Component Analysis* en anglais, est une analyse des composantes principales d'un ensemble de données visant à réduire les dimensions de ces données. 40

PDF *Probability Density Function* en anglais, est une fonction dont la valeur à tout échantillon (ou point) donné dans l'espace d'échantillonnage peut être interprétée comme fournissant une probabilité relative que la valeur de la variable aléatoire soit égale à cet échantillon. 14, 15, 16, 17, 18, 19, 24, 103

PNG *Portable Network Graphics* est un format ouvert d'images numériques. 91, 92, 96

PSNR *Peak signal-to-noise Ratio* en anglais, est une mesure de différence entre deux images. 77

PT *Path Tracing* en anglais, est une méthode classique de rendu d'images de synthèse où les rayons lumineux sont émis aléatoirement depuis la caméra. 35, 100, 117

RAW *RAW* est la désignation générique d'un type de fichier d'images numériques issues d'appareils photo numériques ou de scanners. 91

RAWLS *RAW Light Simulation* est un format d'images stockant les valeurs de luminances brutes. 91, 92, 96

RBF Le *Radial Basis Function* est un noyau utilisé pour les modèles SVM pour réduire la complexité de la séparabilité des données. 50, 183

rEQM La racine carré de l'erreur quadratique moyenne (RMSE en anglais, pour *Root Mean Square Error*). xxi, 35, 36, 43, 106, 111, 115

RFE *Recursive Feature Elimination* en anglais, est un processus visant à réduire le nombre de caractéristiques relativement aux performances d'un modèle. xxii, 162, 182, 183, 184

RGB Canaux de couleurs Rouge, Vert et Bleu (*Red, Green, Blue* en anglais). 54, 60, 61, 62, 149, 150

RNN *Recurent Neural Network* en anglais, est un réseau de neurones dit récurrents spécifique aux traitement temporel d'informations. xvii, xviii, xx, xxi, xxii, xxiii, 55, 56, 124, 128, 129, 130, 131, 132, 133, 136, 137, 138, 139, 140, 141, 142, 143, 144, 155, 156, 157, 184, 185, 196, 232, 235

ROC *Receiver Operating Characteristic* en anglais, est un graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. xix, 42, 43, 133

spp *Samples per pixel* en anglais, désigne le nombre d'échantillons par pixel d'une image calculée. 36, 95

SSIM *Structural SIMilarity* est une mesure de similarité entre deux images numériques. xvii, xx, xxi, 30, 77, 78, 94, 106, 111, 112, 113, 114, 115, 116, 118, 140, 141, 143, 150, 154, 190, 196

SURE *Stein's Unbiased Risk Estimator* est un estimateur non biaisé de l'erreur quadratique moyenne (EQM). 33, 34, 35, 58

SV *Singular Value* en anglais, est le vecteur de valeurs singulières obtenu avec à partir de la SVD. 125, 127, 132

SVD *Singular Value Decompositon* en anglais, est une méthode de décomposition en valeurs singulières d'une matrice d'informations. xii, xv, xvi, xxi, 34, 40, 78, 121, 124, 125, 127, 131, 144

SVM Les *Support Vector Machine* en anglais, sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de classification et de régression. xv, xvi, xx, 45, 46, 47, 48, 49, 82, 83, 84, 85, 121, 122, 123, 145, 160, 172, 182, 183, 184, 185, 186, 203, 204

SVR *Support Vector Regression* en anglais, est un type de modèe SVM orienté sur de la régression continue. 79

SVs *Singular Values* en anglais, est une ou plusieurs composantes du vecteur de valeurs singulières obtenu avec à partir de la SVD. 125, 126

TFP Taux de faux négatifs. 42

TVP Taux de vrais positifs. 42

UCB *Upper Confidence Bound* en anglais, est un algorithme permettant de réaliser un compromis entre exploration et exploitation notamment dans le cadre du problème des bandits manchots. 167, 173, 174

VAE *Variational AutoEncoder* en anglais, est un type de réseau de neurones d'encodage de données. 152

VDM *Visual Discrimination Model* en anglais, est un modèle produisant une carte de différences visibles entre deux images. 76

VDP *Visual Difference Predictor* en anglais, est un modèle de perception de différence entre deux images. 72, 75, 76, 82

VN Vrais négatifs. 41

VP Vrais positifs. 41

WSaO *Walsh functions Surrogate-assisted Optimization algorithm* en anglais, est un framework proposé pour de l'optimisation assistée par modèle de substitution à base de fonction de Walsh. 172

Liste des tableaux

4.1 Représentation des 13 images obtenues après application de filtres sur l'image de canal L. 84

6.1 Comparaison SSIM avec l'image de référence et différentes valeurs de M 113
6.2 Échantillons requis par pixel afin d'atteindre un score SSIM par rapport aux images de référence 114

7.1 Résultats des 20 meilleurs modèles pour chaque combinaison de paramètres 134
7.2 Résultats des 20 meilleurs modèles de réseau de neurones récurrent (RNN) avec les attributs 26 de (J. CONSTANTIN, BIGAND et al. 2015) . 138
7.3 Pourcentage de bonnes prédictions par les modèles (critère d'arrêt) avec une marge d'erreur autorisée autour du seuil humain pour l'ensemble des scènes. 139

8.1 Performances de GGN comparés au RNN sur les ensembles d'apprentissage et de test 156

9.1 Nombre de coefficients pour les décompositions multilinéaires ou de Walsh, en fonction de la dimension du problème n et de l'ordre L. . . . 170
9.2 Comparison des temps de calcul moyens de la recherche locale classique et assistée par méta-modèle 180
9.3 Scores AUC ROC obtenus sur la base de test par les deux approches de sélection d'attributs en fonction d'un type de modèle. 184

10.1 Pourcentage de critères d'arrêts inférieurs ou supérieurs aux seuils humains en fonction du seuil de classification t. 188
10.2 Nombre d'images et pourcentage d'images considérées comme non bruitées par les utilisateurs lors de l'expérience. 192
10.3 Résultats de l'expérience pour chacune des images en fonction du nombre de participants. 193
10.4 Pourcentage d'échantillons économisés par chacun des modèles comparés aux seuils humains des blocs d'une image. 195

A.1 Moyenne des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image. 225
A.2 Écart-type des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image. 226
A.3 Le coefficient relatif des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image. 227

C.1 Les 40 meilleurs modèles obtenus avec les différents jeux de paramètres pour des attributs issus sur la *SVD-Entropy* en entrée. 231
C.2 Pourcentage de bonnes prédictions avec une marge d'erreur pour les meilleurs modèles RNN avec H_{SVD} et les 26 attributs en entrée. 232

Table des figures

1.1 Représentation de l'angle solide 8
1.2 Transmission de la lumière : matériaux réflectifs (BRDF) et matériaux transmissifs (BTDF) 10
1.3 Intégration de la *Probability Density Function* pour obtention de la *Cumulative Distribution Function* 18
1.4 Représentation du tracé de chemins 22
1.5 Illustration du tracé de chemins avec estimation du pas suivant 22
1.6 Aperçu du bruit résiduel de Monte-Carlo 23
1.7 Illustration de l'algorithme du tracé de chemins bidirectionnel 24
1.8 Comparaisons de deux méthodes de résolutions de Monte-Carlo 25

2.1 Pipeline de rendu d'une image avec processus de post-traitement 28
2.2 Résultats de comparaisons de l'approche Bayesienne de (BOUGHIDA et al. 2017) avec celle de (DELBRACIO et al. 2014) (RHF) 30
2.3 Méthodes de reconstruction au sein du moteur de rendu 31
2.4 Fonctionnement d'un moteur de rendu avec application d'un échantillonnage adaptatif. 32
2.5 Comparaisons de différentes approches d'échantillonnage adaptatives . 35
2.6 Comparaisons des approches adaptatives (WLR) (MOON, CARR et al. 2014) et celle de (Y. LIU et al. 2018) 36

3.1 Représentation de la courbe ROC et de l'aire sous cette courbe 43
3.2 Aperçu du problème du surapprentissage d'un modèle 45
3.3 Représentation de la robustesse d'un modèle de type machine à vecteur support 47
3.4 Représentation d'un modèle perceptron et de ses arêtes 51
3.5 Représentation d'un réseau de neurones composés d'un ensemble de nœuds (neurones) et de connexion entre ces nœuds 52
3.6 Représentation d'un réseau de neurones composés de couches convolutives 54
3.7 Aperçu d'un réseau de neurones Auto*AutoEncoder*encoder 54

3.8 Illustration du *Generative Adversarial Network* où deux réseaux s'affrontent . . . 55
3.9 Illustration des réseaux RNN possibles . . . 56
3.10 Illustration de fonctionnement d'une cellule LSTM dans un réseau de neurones récurrent . . . 57
3.11 Représentation de fonctions d'activation connues . . . 58
3.12 Comparaisons des approches de (LBF) (KALANTARI et al. 2015) et de (BAKO et al. 2017) (KP, pour *Kernel Prediction*) . . . 60
3.13 Illustration d'un réseau de neurones *AutoEncoder* en U . . . 61
3.14 Représentation de l'approche proposée par (XU et al. 2019) . . . 63

4.1 Luminance d'un papier blanc selon différents éclairements ($cd \cdot m^{-2}$) (DELORME et al. 2014) . . . 69
4.2 Aperçu de l'amplitude du contraste en fonction de la fréquence spatiale 71
4.3 Fonction de la sensibilité au contraste en fonction de la fréquence spatiale (CSF) . . . 72
4.4 Comparaisons de la SSIM et du HDR-VDP sur un point de vue de la scène *Bedroom* . . . 78
4.5 Processus de rendu d'une image de synthèse avec interaction et interrogation du modèle de perception. . . . 81
4.6 Processus de rendu d'une image de synthèse avec interaction et interrogation du modèle de perception sur des parties de l'image . . . 81
4.7 Processus d'extraction des masques de bruit à partir du bloc pour conception des données d'entrée au modèle SVM par l'approche de (TAKOUACHET et al. 2017). . . . 83
4.8 Illustration du processus d'extraction des 26 caractéristiques proposé par l'approche de (J. CONSTANTIN, BIGAND et al. 2015) . . . 84
4.9 Illustration du processus de conception de l'image à partir du bloc de l'approche de (J. CONSTANTIN, I. CONSTANTIN et al. 2016) comme entrée au modèle SVM. . . . 85

5.1 Aperçu d'images photoréalistes générées par le moteur de rendu PBRT avec 10 000 échantillons par pixel. . . . 90
5.2 Aperçu de différentes valeurs de correction γ appliquées à l'image originale ($\gamma = 1$) de la scène *Kitchen*. . . . 92
5.3 Processus de division des blocs de taille 200×200 sur les images générées. 93
5.4 Aperçu du fonctionnement de l'application de recueil de seuils subjectifs 94
5.5 Reconstruction des blocs de l'image et comparaison de l'image *humaine* à l'image de référence. . . . 95
5.6 Labellisation des images disponibles d'un bloc en fonction du seuil humain subjectif . . . 96

6.1 Aperçu des *fireflies* d'une partie recadrée de la scène de Veach 100
6.2 Aperçu de l'impact de la contribution élevée des échantillons obtenus sur $\bar{\theta}$, l'estimateur de la moyenne arithmétique. 100
6.3 Comparaisons de MoN et de la moyenne sur des échantillons de rendu (spectre rouge) avec et sans valeur aberrante. 105
6.4 Vue d'ensemble de la carte dynamique du paramètre M de (JUNG et al. 2015). 106
6.5 Comparaison de la méthode classique Mean, (JUNG et al. 2015) avec le paramètre automatisée M. 106
6.6 Évolution du coefficient de Gini sur les moyennes M obtenues lors du rendu avec $M \in \{11, 15, 21\}$ 107
6.7 Illustration de l'équation 6.8 pour $c = 1$, où les deux moyennes $\hat{\theta}_1$ et $\hat{\theta}_M$ sont exclues. 110
6.8 Images de référence utilisées pour les comparaisons des estimateurs . . 111
6.9 Étude de convergence des estimateurs basés sur G avec $M = 21$ 112
6.10 Comparaisons de la rEQM et du SSIM obtenus à partir de différents estimateurs 115
6.11 Étude de convergence de la méthode de (ZIRR et al. 2018) et des estimateurs étudiés précédemment avec $M \in \{5, 25\}$ sur une scène avec *fireflies* 116
6.12 Comparaisons des méthodes G-MoN et (ZIRR et al. 2018) avec $M = 25$ 117
6.13 Étude de convergence de la méthode de (ZIRR et al. 2018) et des estimateurs étudiés précédemment avec $M \in \{5, 25\}$ sur une scène sans *fireflies*. 118
6.14 Comparaison des estimateurs G-MoN (MoN adaptatif avec l'utilisation du coefficient de Gini) 119

7.1 Vue d'ensemble des composantes de Σ sur le point de vue de *Living room*, mises à l'échelle pour afficher les composantes les plus faibles. . 126
7.2 Reconstruction SVD d'une image d'un bloc de taille 200×200 de l'image *Living room* 127
7.3 Aperçu de l'évolution de H_{SVD} pour l'image *Living room*. 128
7.4 Architecture prévue du RNN avec S comme taille de la fenêtre glissante. 129
7.5 Pipeline complet avec architecture RNN et paramètres testés associés . 131
7.6 Aperçu de l'évolution des composantes H^1_{SVD} et H^2_{SVD} pour les 4 blocs sélectionnés 135
7.7 Prédictions du modèle pendant le rendu du RNN sélectionné avec les données d'entrée H_{SVD}. 139
7.8 Cas de prédiction de seuil erroné obtenu par le modèle. 140
7.9 Images reconstruites à partir des seuils prédits en utilisant le RNN avec le modèle SVD-Entropy et les seuils humains. 141

7.10 Images reconstruites à partir des seuils prédits en utilisant le RNN avec le modèle SVD-Entropy 142
7.11 Seuils prédits à partir du modèle RNN H_{SVD} 143

8.1 Architecture du modèle GGN proposé 148
8.2 Architecture détaillée du *Denoising Autoencoder* 149
8.3 Architecture détaillée du *Feature Map Generator* 151
8.4 Architecture du *Discriminator* 153
8.5 Résultat du débruitage par le *Denoising AutoEncoder* sur un bloc de l'image *Kitchen* 154
8.6 Résultats de sortie du *Feature Map Generator* 155
8.7 Comparaison des prédictions des modèles LSTM et GGN sur certains blocs de 3 images de l'ensemble de test 157
8.8 Résultat de prédictions du GGN sur le bloc problématique de *Classroom* 158

9.1 Présentation du processus de sélection d'attributs. 161
9.2 Problème de sélection d'attributs vu comme un problème d'optimisation où $f(x)$ représente le score AUC ROC obtenu par le modèle d'apprentissage sur la base de données de test réduite à partir de la solution de filtrage des attributs x. 163
9.3 Illustration théorique de différents espaces de recherche de solutions . . 166
9.4 Représentation de l'évolution des performances d'un méta-modèle au cours de la recherche de solutions 168
9.5 Illustration d'un modèle de type forêt aléatoire composé de différents arbres 173
9.6 Comparaisons de résultats de recherche de solutions avec méta-modèle en fonction du paramètre LS 177
9.7 Comparaisons de résultats de recherche de solutions avec méta-modèle en fonction du paramètre d'ordre L 178
9.8 Courbes d'apprentissage des méta-modèles lors de la recherche de solutions 179
9.9 Aperçu des attributs sélectionnés lors de la recherche de solutions assistées par méta-modèle. 181
9.10 Aperçu des attributs sélectionnés lors de la recherche de solutions assistées par méta-modèle en fonction de L 182
9.11 Aperçu des attributs sélectionnés par la méthode proposée comparée à ceux sélectionnés l'approche RFE. 183
9.12 Comparaisons des prédictions obtenues à partir du modèle de forêt aléatoire retenue sur des blocs non appris. 185

10.1 Répartition des écarts entre les seuils produits et les seuils subjectifs. . . 189

10.2 Représentation de l'application de validation des images reconstruites avec les prédictions des modèles de perception. 191
10.3 Aperçu d'un bloc de l'image *Sanmiguel vue 2* où plusieurs *fireflies* sont apparus. 192
10.4 Seuils objectifs obtenus sur les images *Classroom* et *Living-room* calculées jusqu'à 64 000 échantillons. 196

A.1 Aperçu des images de référence avec 10 000 échantillons par pixel de la base d'images. 228

C.1 Images les plus bruitées (20 échantillons par pixel) de 6 points de vue parmi les 35 disponibles de la base d'apprentissage. 233
C.2 Images de référence (à 10 000 échantillons par pixel) de 6 points de vue parmi les 35 disponibles de la base d'apprentissage. 234
C.3 Images bruitées (20 échantillons par pixel) de 4 points de vue non utilisés en apprentissage par le modèle RNN. 235
C.4 Images de référence (10 000 échantillons par pixel) de 4 points de vue non utilisés pour l'apprentissage. 236

Introduction

L'utilisation d'images réalistes générées par ordinateur s'est développée et diffusée dans de nombreux domaines d'activité. Par exemple, ces images sont utilisées dans l'industrie du divertissement pour produire des contenus (films, jeux vidéo) ou pour prévisualiser rapidement des idées/prototypes futurs. La synthèse d'image vise donc à produire une image calculée par ordinateur à partir d'un environnement virtuel modélisé en trois dimensions (3D), appelé scène.

Le processus de développement des images de synthèse passe principalement par deux grandes étapes :

La modélisation virtuelle d'une scène où l'on représente les coordonnées 3D et propriétés géométriques d'objets modélisés :

- les sources lumineuses ;
- les surfaces et/ou volumes ainsi que leurs matériaux qui interagissent avec la lumière provenant de sources lumineuses (réflexion, réfraction, diffusion, absorption, etc.) ;
- une caméra virtuelle qui représente le spectateur.

La simulation d'éclairage (processus de rendu) à partir de la caméra virtuelle au sein d'une scène, cherche à restituer ce que l'observateur serait amené à visualiser sous forme d'une image en deux dimensions. La génération de telles images se fait en simulant les interactions de la lumière avec les objets présents dans la scène via un moteur de rendu.

Les images générées ont plusieurs niveaux de réalisme. On peut distinguer deux grandes familles de rendus d'images de synthèse : le rendu temps réel, principalement utilisé dans les jeux vidéos afin d'avoir un rendu synthétisé de l'environnement en temps interactif (30 ms par image en moyenne) et le rendu photo-réaliste qui vise à rendre une image générée par un ordinateur indistinguable d'une image naturelle capturée (appareil photo notamment).

Contexte

Les travaux de thèse présentés dans ce mémoire se focalisent principalement sur la seconde grande famille de rendu d'image, celle du photo-réalisme. Ce type de rendu a pour but de restituer une image la plus réaliste possible par des méthodes d'éclairage global. Ces méthodes sont basées sur les interactions physiques réelles de la lumière dans notre monde et permettent de capturer des effets lumineux existants complexes, comme les effets d'ombrage, de réflexion de la lumière, de caustique, etc.

Ces méthodes cherchent à résoudre une équation intégrale, nommée « équation de rendu » (KAJIYA 1986), qui décrit le flux lumineux réfléchi par un élément de surface dans une direction en fonction de l'éclairement et des propriétés de réflexion du matériau. Cette équation ne peut généralement pas être résolue de manière analytique et de nombreuses méthodes de résolution numériques ont été proposées dans la littérature. Parmi celles-ci, une classe de méthodes s'appuie sur de l'échantillonnage de Monte-Carlo et se fonde sur la théorie des grands nombres afin d'approcher au mieux le résultat attendu de cette équation pour chaque pixel. Elles utilisent des méthodes stochastiques, qui explorent l'espace des chemins lumineux et se caractérisent par une convergence progressive de l'image vers une solution visuelle correcte.

La convergence de l'image souhaitée peut en fonction de la complexité de la scène nécessiter des heures, voire des jours de calcul. Au fur et à mesure du rendu de cette image, un bruit perceptuel se réduit, aussi appelé bruit résiduel de Monte-Carlo. Ce bruit est dû à un manque de précision des valeurs obtenues à différents stades du processus d'échantillonnage des pixels d'une zone de l'image où certains effets lumineux sont difficiles à restituer car nécessitant un plus grand temps de calcul.

Problématique et objectifs

Durant cette thèse, nous cherchons à comprendre l'impact du bruit présent lors de la génération d'une image sur la perception humaine, à l'analyser, et à identifier à partir de quel niveau de précision l'humain ne le perçoit plus afin de le quantifier et le traiter.

Les travaux rapportés par ce mémoire visent à contribuer au projet de l'Agence Nationale de la Recherche (ANR [1]) nommé *Perception, Interactions et Simulation d'Éclairage 3D* (PrISE-3D) dans lequel la thèse s'insère. Le projet PrISE-3D s'intéresse à la perception humaine du bruit dans les images de synthèse pour des images classiques, mais aussi sur des périphériques de rendus stéréoscopiques, tels que les casques de réalité virtuelle. Il vise également à étudier les méthodes d'amélioration et d'accélération d'image

1. Support de financement ANR : ANR-17-CE38-0009

photo-réaliste afin de restituer uniquement dans la vision centrale de l'œil une image photo-réaliste en temps réel et de proposer un *rendu mixte* à l'utilisateur.

Ainsi, les travaux de thèse visent principalement deux objectifs :

— la détection de bruit dans les images de synthèse durant leur génération via un moteur de simulation d'éclairage (moteur de rendu), afin de déterminer un critère d'arrêt de rendu. L'objectif est d'atteindre un seuil de qualité d'image pour lequel l'humain ne percevra plus de différence ;

— l'étude des méthodes d'intelligence artificielle pour la perception du bruit résiduel de Monte-Carlo ;

Plan

Les travaux rapportés dans la suite de ce mémoire sont organisés en deux grandes parties. La première relative à l'état de l'art et la seconde présentant les différents travaux réalisés dans le cadre de cette thèse.

La première partie propose un aperçu de la littérature des méthodes d'éclairage global, de l'utilisation de méthode d'intelligence artificielle pour ces techniques de simulation d'éclairage et de la perception du bruit dans les images. Le sujet traité dans ce mémoire étant axé sur plusieurs domaines de recherche, il est nécessaire d'aborder les fondements théoriques pour chacun d'entre eux. Le chapitre 1 définit les modèles mathématiques et physiques proposés pour simuler numériquement l'éclairage au sein d'une scène 3D. Il détaille également les méthodes existantes de production d'images de synthèse photo-réalistes, en les déclinant en deux catégories, les méthodes biaisées et non biaisées. Le chapitre 2 offre un panorama des travaux de la littérature traitant de l'application de méthodes statistiques pour l'amélioration du rendu d'images de synthèse. Il aborde la notion de rendu progressif et débruitage des images en sortie du moteur de rendu. Le chapitre 3 propose dans un premier temps de définir ce qu'est l'intelligence artificielle. Puis, dans un second temps, il expose les travaux récents où les méthodes d'intelligence artificielle pour la synthèse d'image ont été utilisées visant à améliorer rapidement l'image obtenue. Le chapitre 4 introduit les travaux présents dans la littérature relatifs à la thématique sous-jacente de la thèse, ceux de la perception du bruit présent dans une image et de la mesure subjective de qualité d'une image.

La deuxième partie présente les contributions scientifiques relatives à la perception du bruit dans les images de synthèse réalisées dans le cadre de nos travaux, ainsi que les perspectives qu'ils offrent à la suite de cette thèse. Le chapitre 5 présente tout d'abord une base d'images de référence pour l'étude de la perception du bruit dans l'image de synthèse. Le chapitre 6 s'intéresse à la présence de données aberrantes qui peuvent apparaître durant les calculs et qui génèrent un artefact fortement perceptible par l'homme. Ce

chapitre propose ainsi une méthode d'estimation de la valeur finale du pixel permettant d'éviter et d'atténuer ce genre de problèmes. Le chapitre 7 tire parti d'une analyse approfondie de notre base d'images et propose une première méthode d'apprentissage automatique supervisé pour répondre au problème de perception de bruit dans les images de synthèse. Le chapitre 8 propose une nouvelle approche d'apprentissage automatique qui, contrairement à la précédente, apprend seule la caractéristique de bruit ciblé afin de répondre au mieux à la tâche demandée. Le chapitre 9 présente une nouvelle approche de sélection d'attributs en entrée au modèle d'apprentissage, qui vise à formuler ce problème de sélection d'attributs comme un problème d'optimisation binaire. Le chapitre 10 propose une étude de validation des modèles d'apprentissage automatique obtenues et enfin, le manuscrit se termine par un chapitre de conclusion, qui énumérera en outre les nombreuses perspectives envisagées à la suite de nos travaux.

Contributions et publications

Les travaux présentés dans ce mémoire de thèse ont donné lieu aux contributions suivantes :

— **J. Buisine**, A. Bigand, R. Synave, S. Delepoulle & C. Renaud (Janvier 2021). « Stopping Criterion during Rendering of Computer-Generated Images Based on SVD-Entropy ». *Entropy - Vol. 23* ;

— **J. Buisine**, S. Delepoulle, R. Synave & C. Renaud (Février 2021). « Subjective human thresholds over computer generated images ». *Zenodo* (base d'images de synthèse) ;

— **J. Buisine**, S. Delepoulle & C. Renaud (Mars 2021). « Minimalist And Customisable Optimisation Package ». *The Journal of Open Source Software (JOSS) - Vol. 6* ;

— **J. Buisine**, S. Delepoulle & C. Renaud (Juin 2021). « Firefly removal in Monte Carlo rendering with adaptive Median of meaNs ». *Eurographics Symposium on Rendering (EGSR)* ;

— V. Myrodia, **J. Buisine** & L. Madelain (Septembre 2021). « Comparison of threshold measurements in laboratory and online studies using a Quest+ algorithm ». *Journal of Vision - Vol. 21* ;

— **J. Buisine**, F. Teytaud, S. Delepoulle & C. Renaud (Accepté, Décembre 2021). « Guided-Generative Network for noise detection in Monte-Carlo rendering ». *International Conference on Machine Learning and Applications (ICMLA).*

Première partie

État de l’art

Chapitre 1

Le rendu d'images photo-réaliste

Introduction

Ce premier chapitre donne une définition formelle des notions physiques et mathématiques utilisées pour la simulation d'éclairage lors du calcul d'une image de synthèse. Il présente la différence entre l'éclairage direct et indirect qui peut être perçu en chaque point de l'image afin d'introduire, ensuite, la notion d'éclairage global. Il détaille l'approche globale de résolution de l'équation du rendu proposée dans la littérature, fondée sur la méthode d'échantillonnage de Monte-Carlo. Enfin, il propose une liste non exhaustive d'algorithmes de rendu apportant chacun de bonnes solutions pour la résolution et l'approximation de l'équation.

Nous allons résumer dans ce premier chapitre les aspects techniques et mathématiques utilisés autour de la simulation d'éclairage, permettant la génération d'images de synthèse à haut niveau de réalisme. Nous nous intéresserons dans le cadre de la thèse plus particulièrement au rendu d'images dit physiquement réaliste qui vise à restituer une image telle que l'on peut la percevoir dans la réalité. L'intérêt porté sur ce type de rendu, est qu'il nécessite un calcul important de simulation de l'éclairage pour atteindre une restitution de qualité, ce qui est le cas pour la génération de films d'animation. Une image de synthèse est une image générée par un ordinateur où l'on a à calculer une image depuis un point de vue d'une scène 3D modélisée. Une scène est composée de sources lumineuses, d'objets, de matériaux et d'une caméra spécifiant l'emplacement du regard de l'observateur.

C'est depuis le point de vue du spectateur (coordonnées et orientation de la caméra) qu'est restitué le contenu de la scène sous la forme d'une image. Dans le cadre de la conception d'une image photo-réaliste, où l'on simule les interactions lumineuses globales, restituer une telle image implique de déterminer les objets visibles et leur couleur en chaque pixel de celle-ci. La lumière est exprimée par un flux émis par des sources lumineuses qui atteint progressivement un équilibre énergétique par rapport aux

différentes interactions des matériaux des objets présents. Cependant, dans la vie réelle, la vitesse de la lumière est si rapide que l'équilibre énergétique est atteint instantanément. L'objectif des techniques de rendu basées sur la physique est d'évaluer numériquement cet équilibre afin de calculer une image finale.

Pour tout élément de surface de notre scène, visible au travers d'un pixel, on cherche à connaître la contribution lumineuse de l'environnement sur ce point. Celle-ci peut être fournie directement (s'il s'agit d'une source lumineuse) et indirectement au travers de l'ensemble des interactions lumineuses entre les objets et de leurs propriétés physiques liées à leurs matériaux.

Toutefois, déterminer le cheminement de la lumière est difficile en raison de la complexité des différentes interactions lumineuses. Dans la section 1.1, nous présenterons les modèles physiques utilisés pour décrire les différents chemins de la lumière et leurs interactions. L'ensemble des chemins lumineux possibles est infini, une approximation est alors nécessaire. Une solution consiste à utiliser les méthodes de Monte-Carlo qui créent de manière aléatoire des trajets lumineux dans la scène (section 1.2). C'est à partir de ces chemins échantillonnés que la solution est calculée comme la moyenne des contributions récupérées par pixel, pour produire une image. Cette solution est élégante et peut fournir des images de qualité. Toutefois, nous verrons que certaines restitutions sont plus difficiles que d'autres et peuvent amener un bruit visuel perceptible. La dernière partie de ce chapitre, la section 1.3 présentera les différentes méthodes d'intégration de Monte-Carlo proposées au sein de la littérature pour répondre à certaines problématiques et effets de la lumière.

1.1 Le transport de la lumière

Cette première section va présenter les différentes mesures physiques de quantification de la lumière, en passant par l'énergie émise d'un photon jusqu'à la quantité de lumière reçue par la une surface d'un objet.

Propagation de la lumière dans un espace 3D La lumière peut atteindre (depuis une direction entrante) ou quitter (selon une direction sortante) une surface. Elle peut également être émise, réfléchie, diffusée ou absorbée en proportions variables en tenant compte des directions incidente et sortante. La propagation de la lumière peut être exprimée dans le domaine de la direction (direction de la diffusion) ou dans le domaine surfacique. Dans cette section, nous passerons en revue les formulations dans ces deux domaines ainsi que les mesures associées afin de les utiliser dans la suite du manuscrit.

Domaine surfacique Dans un environnement 3D, la géométrie de la scène est représentée par un ensemble fini de surface dans $\mathbb{R}^3$. L'union de toutes les surfaces est dénotée $\mathcal{M}$. La mesure d'unité de surface est dénotée $\mathrm{d}A(x)$ à l'emplacement du point x. Chaque

point se voit associer une normale, notée ici $\vec{n}$, qui décrit l'orientation de la surface en ce point dans l'espace 3D.

Domaine directionnel Dans ce domaine, chaque direction est représentée par un vecteur noté $\vec{\omega} \in \mathbb{R}^3$. Le domaine des directions en un point x d'une surface peut être divisé en deux parties.

La première partie, l'espace de l'hémisphère supérieur relativement au vecteur $\vec{n}$:

$$\Omega_+ = \{\vec{\omega} : |\vec{\omega}| = 1, \vec{\omega} \cdot \vec{n} \geq 0\} \tag{1.1}$$

La seconde partie, l'espace de l'hémisphère inférieur relativement au vecteur $\vec{n}$:

$$\Omega_- = \{\vec{\omega} : |\vec{\omega}| = 1, \vec{\omega} \cdot \vec{n} \leq 0\} \tag{1.2}$$

Pour une raison de simplification, l'ensemble des directions possibles de $\vec{\omega}$ au point x d'une surface, sera défini par Ω. De plus, l'égalité suivante sera simplifiée, en supposant les vecteurs utilisés normés :

$$\vec{\omega} \cdot \vec{n} = |\vec{\omega}| \cdot |\vec{n}| \,.\, cos\theta \Leftrightarrow \vec{\omega} \cdot \vec{n} = cos\theta \tag{1.3}$$

où θ représente l'angle polaire entre la direction $\vec{\omega}$ et la normale à la surface $\vec{n}$.

Enfin, la mesure dans le domaine directionnel est l'angle solide, dénoté $\mathrm{d}\sigma(\vec{\omega})$ pour une direction donnée $\vec{\omega}$. Un angle solide est l'équivalent d'un angle 2D, mais ici défini dans un espace 3D (voir figure 1.1). Les unités des angles solides ne sont pas exprimées en degrés, mais en *steradians* (unité d'aire et d'angle).

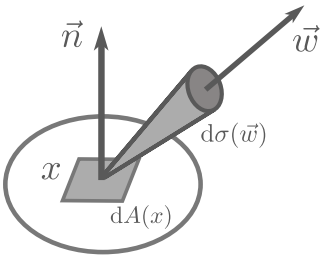


FIGURE 1.1 – Représentation de l'angle solide avec $\mathrm{d}A(x)$, qui représente l'unité d'aire de surface en x et $\mathrm{d}\sigma(\vec{\omega})$ pour l'unité de l'angle solide pour la direction $\vec{\omega}$.

1.1.1 Quantités radiométriques

La lumière peut être considérée comme un rayonnement électromagnétique qui se mesure à l'aide de quantités physiques appelées quantités radiométriques.

1.1.1.1 Flux

Le flux (ou puissance radiante), notée Φ, est la grandeur fondamentale qui exprime l'énergie lumineuse totale Q reçue ou émise par unité de temps vers une surface

$$\Phi = \frac{\mathrm{d}Q}{\mathrm{d}t} \tag{1.4}$$

Elle s'exprime en Watt (W) ou encore Joule par seconde ($\mathrm{J}\cdot s^{-1}$).

1.1.1.2 Éclairement

Cette mesure notée ici E, est le flux reçu par unité d'aire de surface. L'éclairement est exprimé par :

$$E(x) = \frac{\mathrm{d}\Phi}{\mathrm{d}A(x)} \tag{1.5}$$

Son unité est le Watt par mètre carré ($\mathrm{W}\cdot m^{-2}$). La relation entre la quantité et le flux est donnée par :

$$\Phi = \int_S E(x)\mathrm{d}A(x) \tag{1.6}$$

où S est une surface. Une même quantité existe pour l'émission et est nommée émittance.

1.1.1.3 Luminance

Notée L, la luminance est le flux lumineux émis par une surface, par unité d'aire et par unité d'angle solide. La luminance émise en un point x dans la direction $\vec{\omega}$ sera dénotée par $L(x \rightarrow \vec{\omega})$.

L'équation de la luminance émise est donnée par :

$$L(x \rightarrow \vec{\omega}) = \frac{\mathrm{d}^2\Phi}{\mathrm{d}\sigma(\vec{\omega})\mathrm{d}A(x)cos\theta} \tag{1.7}$$

où $\mathrm{d}^2\Phi$ est le flux différentiel de la lumière émise au point x et θ est l'angle entre la normale $\vec{n}$ et la direction $\vec{\omega}$. Cette quantité est la plus importante dans notre cadre, car elle exprime la quantité de lumière perçu depuis une surface et en une direction.

À noter que ces quantités radiométriques doivent être adaptées à la vision humaine qui correspond au récepteur visé pour nos images. Ceci est fait au travers de quantités photométriques, qui seront détaillées dans le chapitre 4.

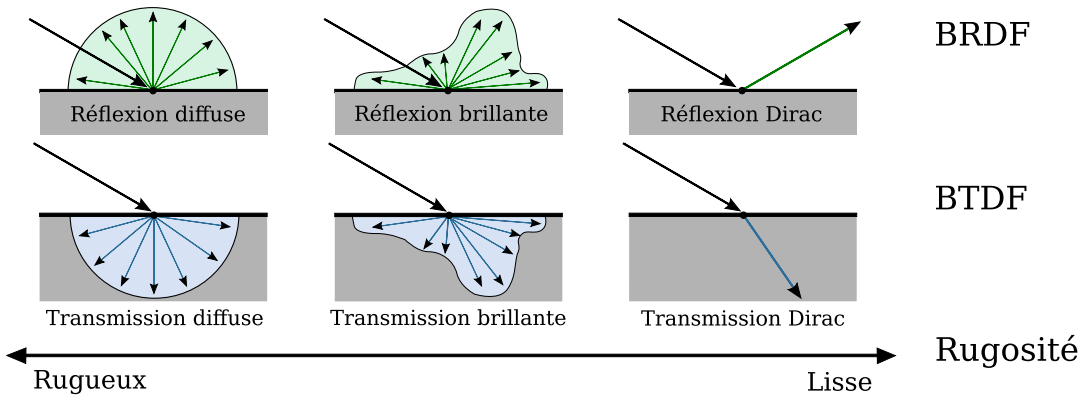


FIGURE 1.2 – Deux principaux types de matériaux existants : matériaux réflectifs (BRDF) et matériaux transmissifs (BTDF). La transmission suit la loi de Snell-Descartes. Pour ces deux types de matériaux, différentes interactions sont possibles : diffuse, brillante et Dirac. La rugosité de ces matériaux implique une interaction de la lumière en surface différente. Une interaction diffuse fait référence à des surfaces rugueuses, et au contraire, des surfaces lisses amènent une simple réflexion de lumière (schéma inspiré de la thèse de (GRUSON 2015)).

1.1.2 Interaction de la lumière avec les matériaux

Lors de la propagation de la lumière un certain nombre d'interactions avec les propriétés physiques de la surface d'un objet ou le volume d'espace traversé interviennent.

La figure 1.2 propose un aperçu des 2 types d'interactions possibles avec une surface. Toutes ces interactions sont traitées par un modèle plus général nommé BSDF (pour *bidirectional scattering distribution function*), que nous nommons ici f_r. Ce modèle inclut deux comportements de la lumière : la réflexion et la transmission. Le modèle BSDF est approximé par différents modèles mathématiques afin de définir les interactions d'un type de matériau (Robert L COOK et al. 1982 ; OREN et al. 1994 ; TORRANCE et al. 1992 ; WALTER et al. 2007).

La BSDF est une propriété locale importante, car elle influe sur l'éclairage global de la scène provenant des inter-réflexions entre les objets. Elle est définie par l'équation 1.8 comme le rapport entre la luminance réfléchie et l'éclairement provenant de $\vec{\omega}_i$ en x qui est réfléchie dans la direction $\vec{\omega}_o$.

$$f_\mathrm{r}(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) = \frac{dL_o(x\vec{\omega}_o)}{dE(x \rightarrow \vec{\omega}_i)} = \frac{dL_o(x \rightarrow \vec{\omega}_o)}{L_i(x \rightarrow \vec{\omega}_i)\cos\theta_{\vec{\omega}_i} d\vec{\omega}_i} \tag{1.8}$$

où $\vec{\omega}_i$ est la direction incidente, $\vec{\omega}_o$ est la direction sortante et $L(x \rightarrow \vec{\omega}_o)$ la luminance dans la direction $\vec{\omega}_o$ au point x.

Pour qu'un modèle BSDF soit correct, il doit respecter trois lois de la physique, à savoir : la non-négativité, la réciprocité et la conservation de l'énergie.

1. *Non-négativité* : la BSDF est une valeur scalaire strictement positive telle que :

$$\forall x, \vec{\omega}_i, \vec{\omega}_o : f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) > 0 \tag{1.9}$$

2. *Conservation de l'énergie* : pour toutes les directions $\vec{\omega}_o$, l'énergie totale de la lumière réfléchie doit répondre à la contrainte suivante :

$$\int_\Omega f_\mathrm{r}(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) \cos\theta_{\vec{\omega}_i} \mathrm{d}\omega_i \leq 1 \tag{1.10}$$

3. *Principe de réciprocité d'Helmoltz* : une BSDF caractérisant la réflectance doit être symétrique. Les directions d'éclairement et d'observation sont donc interchangeables, d'où sa qualification de fonction bidirectionnelle. Pour chaque paire de direction $\vec{\omega}_i$ et $\vec{\omega}_o$, nous avons :

$$f_\mathrm{r}(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) = f_\mathrm{r}(x, \vec{\omega}_o \rightarrow \vec{\omega}_i) \tag{1.11}$$

Ces 3 propriétés forment la base d'un modèle BSDF et permettent dans le cadre du rendu d'images de synthèse photoréalistes de simuler correctement l'éclairage.

À partir d'un modèle BSDF et des propriétés de flux lumineux entrant et sortant sur une surface en une direction, nous pouvons intégrer cette équation et exprimer la luminance sortante dans la direction $\vec{\omega}_o$:

$$\begin{aligned} \mathrm{d}L(x \rightarrow \vec{\omega}_o) &= f_\mathrm{r}(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L(x \leftarrow \vec{\omega}_i) \cos\theta_{\vec{\omega}_i} \mathrm{d}\omega_i \\ L(x \rightarrow \vec{\omega}_o) &= \int_\Omega f_\mathrm{r}(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L(x \leftarrow \vec{\omega}_i) \cos\theta_{\vec{\omega}_i} \mathrm{d}\omega_i \end{aligned} \tag{1.12}$$

L'interaction de la lumière n'intervient pas uniquement sur une surface, mais peut se faire en milieu participant (tels que la fumée, le feu, les nuages, la poussière, etc.), qui, eux aussi, interagissent avec la lumière (contrairement au milieu *vide*).

Dans ces milieux, différentes interactions lumineuses peuvent se produire :

— *l'absorption et la diffusion* : ces deux phénomènes sont responsables de la perte d'énergie à l'intérieur du milieu ;

— *la diffusion entrante* : c'est la lumière diffusée dans la direction de la vue. En chaque point du milieu, la lumière peut provenir de toutes les directions et être partiellement diffusée dans la direction de visée ;

— *émission* : l'émission de lumière, par exemple dans le cadre d'un milieu tel qu'un gaz enflammé (flamme, explosion, ...).

1.1.3 Processus d'intégration et équation de rendu

Nous avons précédemment présenté les propriétés physiques de la lumière et ses interactions, telles que la quantité radiométrique exploitée, les propriétés des matériaux en surface et en volume. Nous allons maintenant définir comment il est possible d'intégrer les quantités de lumière attendues en un pixel de l'image dans le cadre du rendu d'images de synthèses. Nous généralisons ici l'interaction de la lumière pour les modèles à la fois surfacique et volumique.

1.1.3.1 Définition de l'équation de rendu

Dans le rendu basé sur la physique, nous devons calculer la luminance sortante d'une surface vue par la caméra. Il est possible de l'exprimer en utilisant l'équation 1.12. Pour ce faire, nous devons inclure l'émission (éventuelle) de la surface L_e, qui donne l'*équation de rendu* :

$$L(x \to \vec{\omega}_o) = L_e(x \to \vec{\omega}_o) + \int_\Omega f_\mathrm{r}(x, \vec{\omega}_i \to \vec{\omega}_o) L_i(x \leftarrow \vec{\omega}_i) \cos\theta_{\vec{\omega}_i} \mathrm{d}\omega_i \tag{1.13}$$

où $L(x \to \vec{\omega}_o)$ est la luminance de x vue par la caméra et $L_i(x \leftarrow \vec{\omega}_i)$ la luminance incidente. À noter que dans l'intégrale, la luminance incidente $L_i(x \leftarrow \vec{\omega}_i)$ est inconnue, mais peut s'exprimer par la même équation en connaissant le point visible dans la direction $\vec{\omega}_i$. On note ainsi que cette équation est définie de manière récursive.

1.1.3.2 Reformulation surfacique

L'équation de rendu peut être exprimée comme proposée initialement par (KAJIYA 1986), dans le domaine des surfaces $\mathcal{M}$ comme :

$$L(x \to \vec{\omega}_o) = L_e(x \to \vec{\omega}_o) + \int_\mathcal{M} f_\mathrm{r}(x, (x-y) \to \vec{\omega}_o) L_i(x \leftarrow y) V(x,y) G(x \leftrightarrow y) \mathrm{d}A(y) \tag{1.14}$$

où la direction d'incidence est remplacée par y un point de la surface $\mathcal{M}$, $\vec{\omega}_i = (x-y)$ et $L_i(x \leftarrow y)$ représente la luminance incidente émise par le point y vers le point x. Un terme de visibilité est également introduit et noté $V(x,y)$. Il est défini tel que :

$$V(x,y) = \begin{cases} 1 & \text{Si } x \text{ et } y \text{ sont mutuellement visibles} \\ 0 & \text{Sinon} \end{cases} \tag{1.15}$$

Une image générée par ordinateur est un tableau 2D de pixels, chaque pixel ayant une

surface. Pour résoudre des problèmes d'aliasing ou de bruit entre les pixels, chaque pixel est sur-échantillonné en différents points x'. Plus précisément, un ensemble de rayons est émis depuis la caméra, chacun d'eux passant par un point x' situé sur la surface du pixel. Ces rayons peuvent intersecter la scène en un point x. En chaque point x, la luminance réfléchie $L(x \to x')$ doit être évaluée. Pour calculer la luminance vue depuis un pixel j, une intégration sur ce pixel est nécessaire :

$$I_j = \int_{\mathcal{M}\times\mathcal{M}} W_e^{(j)}(x' \to x)\, L(x' \to x)\, V(x,y) G(x' \leftrightarrow x)\, \mathrm{d}A(x')\, \mathrm{d}A(x) \tag{1.16}$$

où x' se situe sur les pixels dans l'espace image et $W_e^{(j)}(x \to x')$ représente le poids de chaque échantillon pour le pixel j. Ce terme inclut par exemple l'opération de filtrage effectuée du côté du pixel.

1.1.3.3 L'espace des chemins

Une formulation plus générale de l'équation du rendu a été introduite par (VEACH 1998). Elle utilise le domaine de la surface pour définir le problème de l'équation du rendu pour un pixel j donné afin d'y obtenir l'intensité lumineuse I_j perçue en ce pixel :

$$I_j = \int_{\mathbb{P}} f_j(\overline{\mathbf{x}}) \mathrm{d}\mu(\overline{\mathbf{x}}) \tag{1.17}$$

où $\mathbb{P}$ est l'espace des chemins, $\overline{\mathbf{x}} = x_0, x_1, ..., x_k$ un chemin défini par k sommets et $\mathrm{d}\mu(\overline{\mathbf{x}})$ la mesure radiométrique finale obtenue par le produit des mesures intermédiaires obtenues dans le domaine de la surface pour chaque x_i :

$$\mathrm{d}\mu(\overline{\mathbf{x}}) = \prod_{i=0}^{k} \mathrm{d}A(x_i) \tag{1.18}$$

La fonction de contribution $f(\overline{\mathbf{x}})$ peut être exprimée comme le produit de la BSDF, de l'émission des sources lumineuses et de la réponse du récepteur :

$$f_j(\overline{\mathbf{x}}) = L_e(x_0 \to x_1)\, T(\overline{\mathbf{x}}) W_e^j(x_{k-1} \to x_k) \tag{1.19a}$$

$$T(\overline{\mathbf{x}}) = G(x_0 \leftrightarrow x_1) \prod_{i=1}^{k-1} f_r(x_{i-1} \to x_i \to x_{i+1})\, V(x,y) G(x_i \leftrightarrow x_{i+1}) \tag{1.19b}$$

où $W_e^j(x_{k-1} \to x_k)$ est la réponse du récepteur et $T(\overline{\mathbf{x}})$ la quantité radiométrique récupérée du chemin en incluant le facteur géométrique et les valeurs BSDF.

Cependant, l'intégrale du chemin inclut tous les chemins de longueurs possibles. Une possibilité est de diviser les différentes longueurs de chemin pour obtenir des intégrales différentes :

$$I_j = \sum_{k=1}^{\infty} \int_{\mathcal{M}}^{k+1} f_j(x_0 \dots x_k) \, \mathrm{d}A(x_0) \dots \mathrm{d}A(x_k) \tag{1.20}$$

Il est ainsi possible de cette manière de prendre en considération l'ensemble des valeurs radiométrique obtenues de chemin de longueurs différentes.

1.2 Intégration par méthode de Monte-Carlo

Quelques solutions d'approximation de l'équation du rendu ont été présentées. De plus, nous avons précédemment défini que le rendu basé sur la physique est un problème d'évaluation d'intégrale (équation 1.20) au travers d'un espace de chemins. Dans cette section, nous allons détailler comment évaluer efficacement cette intégrale en un temps fini. Les approches basées sur la méthode de Monte-Carlo sont de bonnes candidates pour résoudre ce problème en raison de leur simplicité et de leur flexibilité.

1.2.1 Formulation générale

Le terme méthode de Monte-Carlo (MC) couvre toutes les techniques qui se basent sur un échantillonnage stochastique afin d'obtenir une solution numérique approximative pour des problèmes dont la solution analytique est très complexe. Cette méthode se base sur la loi des grands nombres. Elle a été nommée, en 1946, par (METROPOLIS et al. 1949) en référence aux jeux de hasard pratiqués au casino de Monte-Carlo. Par la suite, la méthode de Monte-Carlo a connu un large champ d'application reposant principalement sur les contributions importantes de (VON NEUMANN et al. 1951). Cette méthode est devenue un outil probabiliste incontestable notamment pour évaluer les intégrales multidimensionnelles difficiles à résoudre analytiquement. Elle a été introduite en synthèse d'images par Kajiya (ECKHARDT 1987), pour évaluer l'équation intégrale d'éclairage global. Avant de spécifier comment la méthode de MC est utilisée pour le rendu d'images de synthèses, nous allons la définir sous sa forme générale.

La valeur attendue d'un tel estimateur de MC est $E_p\left[\frac{f(x)}{p(x)}\right]$, où $p(x)$ est la fonction de densité de probabilité (PDF pour *probability density function*). Cette valeur est égale à la valeur intégrale d'une fonction $f(x)$ sur un domaine Ω :

$$\int_{\Omega} f(x)\mathrm{d}x = \int_{\Omega} \frac{f(x)}{p(x)} p(x)\mathrm{d}x = E_p\left[\frac{f(x)}{p(x)}\right] \tag{1.21}$$

L'estimateur de Monte-Carlo est une estimation de la valeur attendue. Cependant, pour calculer l'intégrale de $f(x)$, la méthode de Monte-Carlo estime la valeur attendue de $f(x)/p(x)$. Étant donné un ensemble de N variables aléatoires uniformes $x_i \in \Omega$, l'estimateur de Monte-Carlo F_N est défini par :

$$E_p\left[\frac{f(x)}{p(x)}\right] \approx F_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{p(x_i)} \tag{1.22}$$

Lorsque l'on utilise l'approche de Monte-Carlo, la PDF $p(x)$ doit être différente de zéro pour tous les x, d'où le besoin pour cet estimateur de respecter les conditions suivantes :

$$\forall x \in \Omega, p(x) > 0 \text{ et } \int_\Omega p(x)dx = 1$$

Pour la méthode de Monte-Carlo, la variance correspond à l'erreur d'estimation. L'objectif est de réduire cette variance autant que possible, la variance de l'estimateur F_N étant définie comme suit :

$$V[F_N] = E\left[(F_N)^2\right] - E[F_N]^2 \tag{1.23}$$

1.2.1.1 Propriétés de l'estimateur

D'un point de vue statistique, un estimateur peut être consistant et/ou non biaisé. On dit qu'une solution est non biaisée lorsque l'espérance mathématique de son estimateur est égale à la valeur exacte recherchée $\int f(x)\mathrm{d}x - E[F_N] = 0$. Dans le cas d'une solution biaisée, des approximations des valeurs estimées peuvent être utilisées pour permettre une convergence plus rapide. Toutefois, la convergence vers la solution exacte n'est pas assurée.

Un estimateur de Monte-Carlo F_N est dit consistant s'il converge vers la solution correcte avec un nombre infini d'échantillons. Ceci peut être exprimé par la formule suivante :

$$\lim_{N\to\infty} \mathrm{Prob}\left[\left(F_N - \int f(x)\mathrm{d}x\right) = 0\right] = 1 \tag{1.24}$$

Un estimateur sans biais n'est pas automatiquement consistant. En effet, certains algorithmes de rendu utilisent une étape de pré-calcul pour évaluer certaines valeurs. Ces valeurs sont ensuite utilisées par un algorithme de rendu non biaisé. Cependant, comme ces valeurs ne sont pas affinées, l'algorithme résultant peut être non biaisé, mais non consistant (VEACH 1998).

1.2.1.2 Dimension infinie d'intégration

Comme présenté dans l'équation 1.20, un chemin est composé de k sommets. Fixer un nombre maximum de sommets (dimension de l'intégrale) rendrait l'estimateur biaisé (VEACH 1998). Une solution proposée consiste à utiliser la roulette russe qui est une technique de Monte-Carlo non biaisée qui permet d'arrêter d'une manière probabiliste une marche stochastique récursive (LEWIS et al. 1984 ; SPANIER et al. 2008). L'application de cette technique en synthèse d'image a été introduite en 1990 par (ARVO et al. 1990). Elle fixe une probabilité q d'arrêter le chemin. Cette valeur q peut être choisie de presque n'importe quelle manière ; par exemple, elle pourrait être basée sur une estimation de la valeur de l'intégrande pour l'échantillon particulier choisi, augmentant au fur et à mesure que la valeur de l'intégrande devient plus petite. Avec la probabilité q, l'intégrande n'est pas évaluée pour l'échantillon particulier et une certaine valeur constante c est utilisée à sa place ($c = 0$ est souvent utilisée). Avec la probabilité $1 - q$, l'intégrande est toujours évaluée, mais elle est pondérée par un terme, $1/(1 - q)$, qui tient effectivement compte de tous les échantillons qui ont été ignorés.

Ainsi, la valeur attendue de l'estimateur résultant est la même que la valeur attendue de l'estimateur original :

$$E[I'] = (1 - q)\left(\frac{E[I] - qc}{1 - q}\right) + qc = E[I] \tag{1.25}$$

Cette technique rend l'estimateur sans biais même si elle peut augmenter sa variance. En effet, la roulette russe, à moins que d'une manière ou d'une autre $c = I$, augmente toujours la variance.

1.2.1.3 Stratégies d'échantillonnage

Pour réduire plus efficacement la variance, des stratégies d'échantillonnage ont été utilisées telles que :

— *échantillonnage stratifié* : le principe de cette méthode est de diviser le domaine d'intégration Ω en un nombre d de sous-domaines indépendants $\Omega_1, \Omega_2, ..., \Omega_d$. Sont ensuite générés des échantillons aléatoirement au sein de chacun de ces sous-domaines. L'estimateur de l'intégrale sur le domaine Ω est finalement calculé par la somme des estimateurs des intégrales sur chacun des sous-domaines ;

— *échantillonnage préférentiel* : dans le cas d'une fonction non-uniforme, la convergence de l'estimation de MC par échantillonnage uniforme peut être particulièrement lente. Pour pallier ce problème, l'utilisation d'une PDF p proportionnelle à l'intégrande est privilégiée. Cette technique réduit la variance et est souvent utilisée en éclairage global.

Cette liste n'est pas exhaustive. Il existe bien d'autres techniques de réduction de variance de méthodes d'intégration de Monte-Carlo (BRIOL et al. 2015 ; LEPRÊTRE, TEYTAUD et al. 2017 ; PRESS et al. 1990 ; RASMUSSEN et al. 2003). Dans le cadre de l'application de ces méthodes en simulation d'éclairage, l'échantillonnage préférentiel est une technique courante pour réduire la variance. Cette approche est présentée en détail dans la sous-section suivante.

1.2.2 Échantillonnage préférentiel

Comme précisé précédemment, l'échantillonnage préférentiel vise à choisir une PDF p proportionnelle à l'intégrande f.

1.2.2.1 Formulation générale

Plus la forme de la PDF choisie est similaire à la fonction, plus la variance de l'estimateur est réduite. Un cas extrême correspond à une PDF proportionnelle à la fonction, c'est-à-dire $c \cdot p(x) = f(x)$, où c est une constante. Dans ce cas spécifique, la variance de l'estimateur est 0 :

$$\frac{1}{N} V\left[\frac{f(x)}{p(x)}\right] = \frac{1}{N} V\left[\frac{1}{c}\right] = 0$$

Cependant, ce n'est pas réalisable dans la pratique. La raison principale est que le facteur constant qui est inconnu est intimement lié à l'intégrale que nous essayons d'évaluer :

$$c = \frac{1}{\int_\Omega f(x)\mathrm{d}x}$$

Dans le cadre de la résolution de l'équation du rendu, l'intégrande de l'équation 1.20 est un produit de plusieurs termes. Certains termes sont inconnus (par exemple, la luminance incidente), ce qui rend plus difficile la détermination de la fonction de distribution cumulative (CDF, pour *Cumulative Distribution Function*). La CDF est utilisée pour échantillonner proportionnellement à une PDF (voir figure 1.3).

Le danger d'un mauvais échantillonnage préférentiel où les formes des distributions $p(x)$ et $f(x)$ sont très différentes peut conduire à une variance supérieure à celle obtenue avec un échantillonnage uniforme et par conséquent un bruit perceptible accru.

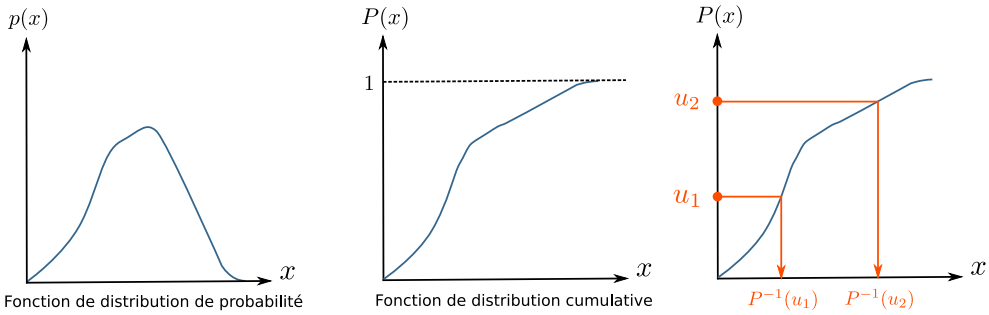


FIGURE 1.3 – La CDF est calculée en intégrant une PDF. Des échantillons uniformes (u_1 et u_2) sont ensuite projetés en utilisant la CDF inverse. Les échantillons projetés sont distribués proportionnellement à la PDF (schéma inspiré de la thèse de (GRUSON 2015)).

1.2.2.2 Stratégies d'échantillonnage préférentiel

En guise d'exemple, nous nous orientons sur la méthode de rendu direct. Deux stratégies d'échantillonnage sont définies ici, en mettant en avant la force et la faiblesse de chacune d'entre elle.

Échantillonnage en fonction de la BSDF Une stratégie envisageable d'échantillonnage préférentiel serait de prendre en considération les propriétés des matériaux au travers de leur modèle BSDF. Pour échantillonner le modèle BSDF, la direction incidente $\vec{\omega}_i$ est échantillonnée aléatoirement selon une PDF $p(\vec{\omega}_i)$. Cette PDF est proportionnelle à la valeur de la BSDF :

$$p\left(\vec{\omega}_i\right) \propto f_{\mathrm{r}}\left(x, \vec{\omega}_i, \vec{\omega}_o\right)$$

où p est exprimé en utilisant les angles solides projetés. En pratique, pour rendre une image, nous traçons d'abord un rayon depuis la caméra à travers un pixel donné. Ce rayon peut intersecter la scène 3D en un point x. Ensuite, nous choisissons une direction de lumière potentielle incidente $\vec{\omega}_j$ selon la BSDF et nous traçons un rayon depuis le point x dans cette direction. Pour un point d'intersection x', nous évaluons la luminance émise $L_e(x \leftarrow x')$. Pour cette stratégie d'échantillonnage, l'estimateur de Monte-Carlo est représenté par :

$$L\left(\vec{x} \rightarrow \vec{\omega}_o\right) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{f_{\mathrm{r}}\left(x, \vec{\omega}_j \rightarrow \vec{\omega}_o\right) L_e\left(x \leftarrow \vec{\omega}_j\right)}{p\left(\vec{\omega}_j\right)} \tag{1.26}$$

où N est le nombre d'échantillons.

Échantillonnage en fonction de la luminance émise Pour pouvoir échantillonner les sources de lumière, nous devons exprimer l'équation de rendu direct dans le domaine de

la surface :

$$L(x \to x^c) = \int_{\mathcal{M}_l} f_{\mathrm{r}}(y \to x \to x^c)\, L_e(x \leftarrow \vec{y}) V(x,y) G(x \leftrightarrow y) \mathrm{d}A(y) \tag{1.27}$$

où $\mathcal{M}_l$ est le domaine de surface des sources lumineuses et y un point sur une source lumineuse.

En pratique, cette stratégie commence de la même manière que la précédente. Premièrement un rayon est émis depuis la caméra à travers un pixel donné (à la position x^c). Ce rayon peut couper la scène 3D en un point x. Nous échantillonnons ensuite aléatoirement une position sur la source de lumière y et évaluons la visibilité de x. Si x est visible depuis y, alors nous évaluons la valeur BSDF $f_{\mathrm{r}}(y \to x \to x^c)$. Pour cette stratégie d'échantillonnage, l'estimateur de Monte-Carlo est défini par :

$$L(x \to \vec{\omega_o}) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{L_e\left(x \leftarrow y_j\right) V(x,y) G\left(x \leftrightarrow y_j\right)}{p\left(y_j\right)} f_{\mathrm{r}}\left(y_j \to x \to x^c\right) \tag{1.28}$$

où $p(y_j)$ est la PDF utilisée pour échantillonner un point y_j sur les surfaces de la source lumineuse et N le nombre d'échantillons.

1.2.2.3 Échantillonnage préférentiel multiple

Déterminer une PDF $p(x)$ pour un domaine d'intégration de grande dimension est une tâche difficile. Une solution consiste à créer cette PDF $p(x)$ en utilisant différentes PDF $p_i(x)$, chacune étant définie sur un sous-domaine qui lui est propre.

Construire $p(x)$ en utilisant plusieurs PDF pour échantillonner la fonction $f(x)$ semble être une bonne stratégie. Toutefois, il faut savoir comment exploiter les différentes distributions d'échantillons pour obtenir une variance réduite. Une solution est de combiner les différents estimateurs avec un estimateur d'échantillonnage préférentiel multiple. Cet estimateur est défini de la manière suivante :

$$F = \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} w_i\left(x_{i,j}\right) \frac{f\left(x_{i,j}\right)}{p_i\left(x_{i,j}\right)} \tag{1.29}$$

où N est ici le nombre de stratégies d'échantillonnage et n_i est le nombre d'échantillons alloués à chaque stratégie. $x_{i,j}$ est le $j^{\text{ème}}$ échantillon de la distribution p_i. Un poids w_i est attribué à chaque échantillon. Cette formule représente la somme pondérée des différents estimateurs $f(x_i, j)/p_i(x_i, j)$.

Pour obtenir un estimateur sans biais, la fonction de pondération w_i doit remplir deux conditions :

1. $\sum_{i=1}^{n} w_i(x) = 1$ pour $f(x) \neq 0$,
2. $w_i(x) = 0$ à chaque fois que $p_i(x) = 0$

Ces conditions impliquent que l'ensemble des techniques d'échantillonnage doit échantillonner les cas où $f(x) \neq 0$. Une technique d'échantillonnage p_i n'a cependant pas besoin d'échantillonner l'ensemble du domaine, mais seulement un sous-domaine.

Différentes stratégies de pondération sont présentées par (VEACH 1998). Une stratégie de pondération possible est l'utilisation d'un paramètre de puissance :

$$w_i(x) = \frac{(n_i p_i(x))^\beta}{\sum_k (n_k p_k(x))^\beta} \tag{1.30}$$

Dans le cas de $\beta = 1$, nous avons une stratégie de pondération appelée heuristique d'équilibre.

L'estimateur par échantillonnage préférentiel multiple (MIS, pour *Multiple Importance Sampling*) s'avère être robuste, mais coûteux en calcul. En effet, pour chaque échantillon, nous devons évaluer le $p_i(x)$ pour toutes les stratégies d'échantillonnage. Ce calcul supplémentaire reste cependant négligeable par rapport à l'opération de lancer de rayons.

Exemple de combinaison Pour pouvoir combiner les deux stratégies d'échantillonnage présentées précédemment avec MIS, c'est-à-dire celle liée à la BSDF et celle liée à la source de lumière, nous devons les exprimer dans le même domaine. L'échantillonnage BSDF $p(\vec{\omega}_i)$ est exprimé selon l'angle solide projeté et l'échantillonnage de la source lumineuse est exprimé dans le domaine de la surface. Une possibilité est de les exprimer tous les deux dans le domaine de la surface en utilisant la transformation suivante :

$$p(y) = p(\vec{\omega}_i)\, G(x \leftrightarrow y) \tag{1.31}$$

où ω_i est l'angle solide associé à la direction $x \to y$. De cette manière, il est possible d'échantillonner à partir de ces deux stratégies, permettant de tirer parti des forces de chacune d'entre elles. Il est par exemple possible d'utiliser la stratégie de pondération de l'équation 1.30 avec un paramètre de puissance adapté afin d'y intégrer nos deux stratégies d'échantillonnage.

1.3 Méthodes d'éclairage global

Contrairement au rendu direct, le rendu par éclairage global tient compte des rebonds multiples de la lumière. Les chemins lumineux composés de plus d'un rebond de lumière sont appelés *chemins indirects*. Le calcul de la part d'éclairage indirect peut prendre beaucoup de temps, mais procure de très bons résultats. De plus, un algorithme de rendu doit trouver un chemin composé de plusieurs interactions qui relie la caméra à une source de lumière. Ce type de chemin est souvent appelé chemin contributif. Il existe plusieurs raisons pour lesquelles un chemin n'est pas contributif : chemin aléatoire avec un accès difficile à la source, source trop petite, etc.

Nous présentons ici quelques algorithmes d'éclairage global offrant différentes techniques permettant de construire efficacement de tels chemins : *tracé de chemins*, *tracé de chemins par estimation du pas suivant* et *tracé de chemins bidirectionnel*. Les algorithmes telles que le *placage de photons* (JENSEN 2001) et *Metropolis* (VEACH et L. J. GUIBAS 1997) ne sont pas explorés dans le cadre de manuscrit.

1.3.1 Tracé de chemins

Le tracé de chemins (ou en anglais *path tracing*), proposé par (KAJIYA 1986), est une extension de la technique de lancer de rayons (Robert L. COOK et al. 1984 ; KAY et al. 1979). Il permet une évaluation complète et rigoureuse de l'équation du rendu. Le tracé de chemins se base sur la méthode de Monte-Carlo pour évaluer la contribution des sources en fonction des propriétés des matériaux.

1.3.1.1 Tracé de chemins classique

Cet algorithme construit un chemin lumineux de la caméra aux sources lumineuses de manière progressive (voir figure 1.4). Après avoir lancé le rayon primaire depuis la caméra et trouvé le premier point d'intersection, un rayon continue de rebondir jusqu'à ce qu'il rencontre une source lumineuse. Cette technique est similaire à l'échantillonnage BSDF pour le rendu direct, car elle ne repose que sur cette stratégie. Ainsi, cet estimateur présente le même inconvénient que le rendu direct avec seulement un échantillonnage BSDF. Dans le cas d'une petite source de lumière, cet estimateur aura une variance élevée.

Pour chaque pixel, selon la méthode de Monte-Carlo, la luminance finale est la moyenne des contributions de tous les chemins utilisés. Le calcul récursif s'arrête lorsque le rayon atteint une source lumineuse ou lorsque la construction du chemin est arrêtée par la technique de la roulette russe.

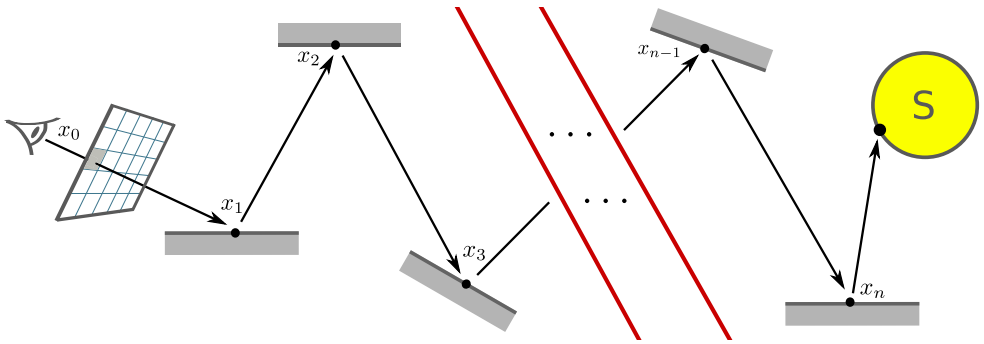


FIGURE 1.4 – Exemple de chemin généré avec la méthode du path tracing classique

1.3.1.2 Tracé de chemins par estimation du pas suivant

Les sources lumineuses ne constituent qu'un petit pourcentage de la scène 3D. Pour l'implémentation classique du tracé de chemins, la probabilité qu'un rayon intersecte une source lumineuse est donc très faible voir nulle si des sources ponctuelles sont utilisées. D'autre part, la technique de la roulette russe, utilisée pour contrôler la récursivité de l'algorithme, stoppe la majorité des évaluations avant d'atteindre les sources lumineuses.

Pour pallier ce problème, une deuxième façon d'implémenter le tracé de chemins est d'utiliser une connexion lumineuse dites *explicite* (SHIRLEY, C. WANG et al. 1996). Comme pour le tracé de chemins classique, de nouveaux sommets (nœuds du chemin en construction) sont générés selon la BSDF. Mais, à chaque point d'intersection, la composante directe est évaluée à l'aide d'une technique d'échantillonnage considérant la luminance émise (voir figure 1.5).

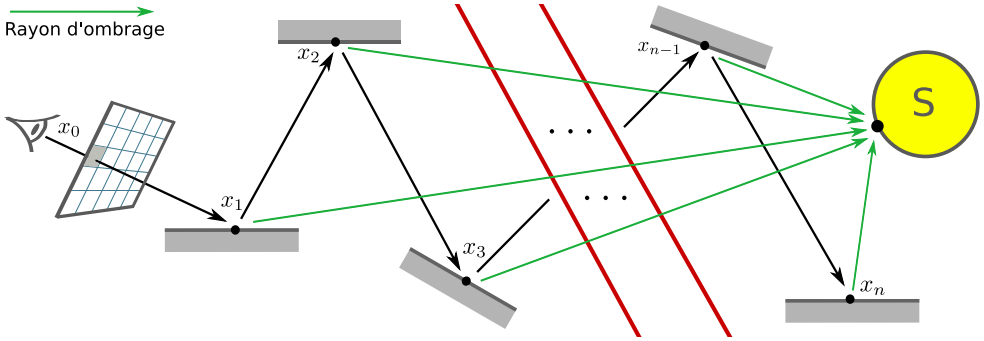


FIGURE 1.5 – Exemple de chemin généré avec la méthode du path tracing avec estimation du pas suivant. À chaque nouvelle intersection x_i, on lance un nouveau rayon d'ombrage en direction de la source lumineuse.

Les deux stratégies d'échantillonnage relativement à la BSDF et à la luminance

émise, peuvent être combinées grâce à l'échantillonnage préférentiel multiple. Lorsque le MIS n'est pas utilisé, le fait de rencontrer aléatoirement une source de lumière (par la procédure de rebond classique) n'apporte aucune contribution. En effet, il n'est pas possible d'avoir deux façons d'échantillonner la même composante (directe) sans les combiner.

(a) 20 échantillons (b) 500 échantillons (c) 10 000 échantillons

FIGURE 1.6 – Aperçu du bruit de Monte-Carlo généré avec plusieurs niveaux d'échantillons lors du rendu d'une image photo-réaliste avec l'algorithme du tracé de chemins.

La façon d'arrêter un chemin généré est la même que le tracé de chemins classique tout comme la méthode de calcul de l'estimateur final du pixel via la méthode de Monte-Carlo, c'est-à-dire, la moyenne des contributions des échantillons obtenus. Même si l'approche semble récolter plus d'informations que le tracé de chemins classique, la figure 1.6 témoigne de la persistance du bruit perceptible obtenu lors du rendu d'image avec l'estimation du pas suivant.

1.3.2 Tracé de chemins bidirectionnel

Le tracé de chemins bidirectionnel (BDPT, pour *Bidirectional Path Tracing*) est une extension de l'algorithme de tracé de chemins. Il a été présenté par Lafortune et Willems sous sa forme originale (LAFORTUNE et al. 1993) et indépendamment par Veach et Guibas (VEACH et L. GUIBAS 1995) qui ont proposé plus tard une stratégie de pondération optimale basée sur l'espace MIS. Cet algorithme présente l'avantage de combiner le tracer de rayons depuis le point de vue ainsi que depuis les sources lumineuses. L'objectif de cette combinaison est d'exploiter le fait que certains phénomènes lumineux sont plus efficacement pris en compte en traçant les rayons depuis le point de vue alors que d'autres phénomènes sont mieux modélisés en traçant les rayons depuis les sources lumineuses.

Le BDPT construit deux chemins différents : un chemin $\overline{\mathbf{x}} = x_0, x_1, \ldots, x_n$ est lancé depuis le point de vue et simultanément un autre chemin $\overline{\mathbf{y}} = y_0, y_1, \ldots, y_m$, est lancé

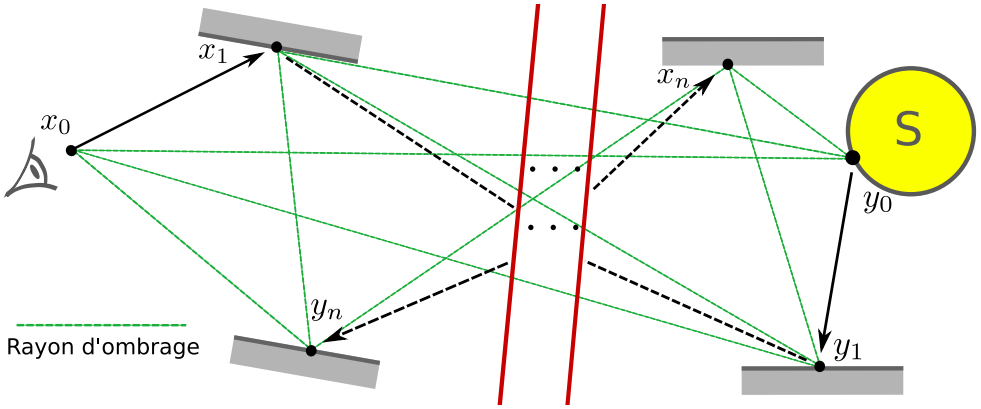


FIGURE 1.7 – Illustration de l'algorithme bidirectionnel où deux chemins sont tracés, l'un partant de la caméra, le second de la source. Les rayons d'ombrages sont générés afin d'essayer d'obtenir des interconnexions.

depuis la source lumineuse. Chaque fois qu'un nouveau sommet x_i ($i \in [0,\dots,n]$) est échantillonné, une connexion explicite entre chacun des points y_j ($j \in [0,\dots,m]$) est réalisée en traçant des rayons d'ombrage de façon déterministe. Ainsi, comme le montre la figure 1.7, chacune des interconnexions résulte en un chemin complet du parcours de la lumière depuis la source jusqu'à l'œil. À chaque fois qu'une connexion est possible, il faut évaluer le poids utilisé pour réaliser l'échantillonnage préférentiel, en calculant la PDF de toutes les manières possibles de construire ce chemin. Les contributions $L_{i,j}$ sont pondérées et sommées. L'estimateur primaire C_p, issu de l'ensemble des contributions $C_{i,j}$ à l'éclairage est calculé selon la formule suivante :

$$C_p = \sum_{i=0}^{n} \sum_{j=0}^{m} w_{i,j} C_{i,j} \tag{1.32}$$

Avec :

— $i = 0$, $j = 0$: $C_{0,0}$ est l'estimation de la contribution de la luminance directement visible par l'œil depuis la source lumineuse ;
— $i = 0$, $j > 0$: $C_{0,j}$ est l'estimation de la contribution qui atteint l'œil depuis la source lumineuse. Elle correspond au cas du tracé de chemins classique où l'on a tracé un chemin depuis la caméra vers la source ;
— $i > 0$, $j > 0$: $C_{i,j}$ est l'estimation de la contribution résultant de la connexion du point x_i (après i intersections depuis l'œil) avec le point y_j (après j intersections depuis la source de lumière) ;
— $w_{i,j}$ est le poids associé à la contribution $C_{i,j}$.

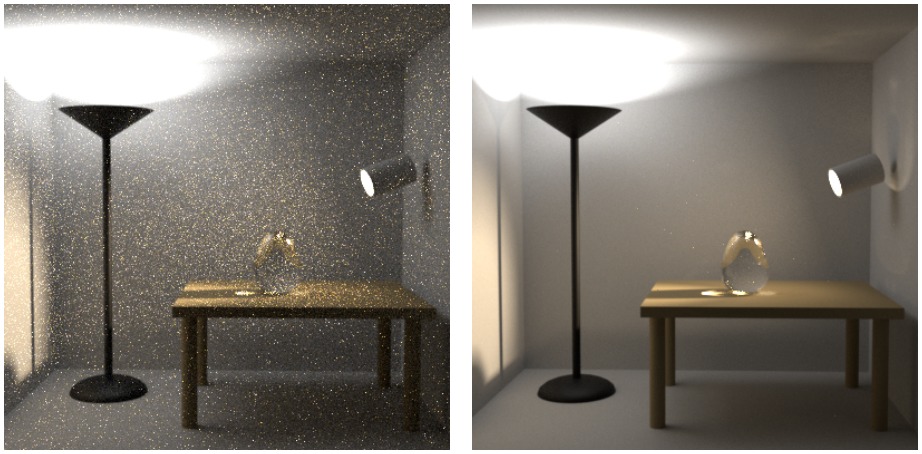
Il existe plusieurs techniques pour calculer le poids $w_{i,j}$ par échantillonnage d'importance multiple. Il est possible, par exemple, de calculer les contributions en tracé de

chemins classique par la fonction suivante :

$$w_{i,j} = \begin{cases} 1 & i = 0 \\ 0 & \text{sinon} \end{cases}$$

(VEACH et L. GUIBAS 1995) ont proposé une technique de pondération heuristique présentée par l'équation 1.30 avec le paramètre $\beta = 2$.

L'utilisation intensive de l'échantillonnage préférentiel multiple rend l'algorithme BDPT robuste. Dans le cas de scènes d'intérieur simples, l'algorithme BDPT peut fournir de meilleurs résultats que les techniques précédemment décrites. C'est sa capacité à échantillonner efficacement les chemins d'éclairage indirect forts qui lui permet d'obtenir de si bons résultats. La figure 1.8 compare les résultats de différentes méthodes pour un nombre d'échantillons identiques.



(a) Tracé de chemins (b) BDPT

FIGURE 1.8 – Comparaisons de deux méthodes de résolutions de l'algorithme de Monte-Carlo pour 1 000 échantillons. Le tracé de chemins comporte malheureusement une variance trop importante pour estimer correctement les pixels de l'image. Il est possible de rapidement souligner les résultats assez significatifs de l'algorithme BDPT.

Résumé

Ce premier chapitre donne une définition formelle de la physique utilisée et des quantités radiométriques traitées dans le cadre du rendu d'image photo-réaliste. Il propose une explication sur la génération des images de synthèses dans une scène 3D, tout comme les composantes d'une telle scène 3D. Il met en avant le processus de simulation d'éclairage réalisé pour ce type de rendu et détaille la solution basée sur la méthode Monte-Carlo proposée au travers de l'équation de rendu de (KAJIYA 1986). Cette équation de rendu permet de simuler à la fois l'éclairage direct et indirect ce qui la qualifie de méthode d'éclairage global. Elle fournit de très bons résultats et un haut niveau de réalisme des images de synthèse. Toutefois, son estimation via l'intégration de Monte-Carlo reste coûteux et plusieurs méthodes ont été proposées dans la littérature pour explorer l'espace des chemins lumineux de la scène de manière plus efficace. Chacune des méthodes proposées dans la littérature possède ses avantages et ses inconvénients, voire ses limites relativement au type de scène 3D dans laquelle l'image est générée. Certaines sont dites sans biais, c'est-à-dire qu'elles n'entraînent pas d'erreur d'estimation de la valeur d'estimation, comme les méthodes de tracé de chemins. Au contraire, certaines sont dites biaisées, comme celles exploitant des cartes de photons, mais permettent de générer certains effets lumineux dans des scènes complexes. Enfin, certaines méthodes visent à combiner plusieurs estimateurs, tels que le tracé de chemins bidirectionnel et le tracé de photons afin de tirer profit des avantages de chacun des estimateurs. Dans le chapitre suivant, nous allons présenter les travaux relatifs à l'utilisation de méthodes d'intelligence artificielle dans le cadre du rendu d'images de synthèses. Notamment sur les méthodes d'apprentissage automatique permettant d'obtenir un rendu d'images adaptatif et accéléré.

Chapitre 2

Apports statistiques pour l'amélioration du rendu

Introduction

Ce deuxième chapitre présente les diverses approches proposées au sein de la littérature visant à améliorer le rendu d'une image de synthèse par méthode de Monte-Carlo (MC) à partir de données et d'analyses statistiques. Contrairement aux travaux présentés dans la section 1.3 qui visaient à améliorer la manière d'explorer l'espace des chemins Ω, les méthodes présentées dans ce nouveau chapitre visent à appliquer des processus de post-traitement afin d'améliorer l'image générée. Ces traitements sont généralement appliqués à la suite d'un processus d'intégration de type tracé de chemins, sur les couleurs des pixels, et peuvent tenir compte d'informations supplémentaires de la scène.

Beaucoup de travaux ont été menés ces dernières années pour améliorer la rapidité de rendu de l'image tout en cherchant à préserver au mieux sa qualité. D'autant plus que tous les algorithmes de MC produisent du bruit, même si certains convergent plus rapidement que d'autres. Dans ce chapitre, nous nous intéressons aux méthodes qui exploitent des informations extraites de l'image et/ou de la scène, soit pour réduire le temps de calcul, soit pour améliorer la qualité des résultats obtenus.

Dans cette première section, nous nous intéressons aux approches qui exploitent uniquement des statistiques extraites au cours du processus de rendu d'une image. Il est possible de distinguer les différentes méthodes statistiques proposées au sein de la littérature et de les agencer en trois sous-catégories. Premièrement, nous présentons les méthodes qui cherchent à approximer une image rapidement à partir d'informations extraites par un processus post-traitement de filtrage lors du rendu. Généralement ces méthodes essaient d'obtenir cette image avec très peu d'échantillons. Puis, nous exposons les méthodes existantes visant à la fois à échantillonner dans l'espace multidimensionnel

de l'intégration, à partir d'informations plus fines et à reconstruire l'image rapidement. Enfin, nous présentons les approches proposant un rendu adaptatif au sein de l'espace de l'image, où les échantillons dans le plan de l'image sont distribués stratégiquement suivant certains critères et où l'image est ensuite reconstruite itérativement jusqu'à un critère d'arrêt.

2.1 Méthodes de reconstruction dans l'espace image

FIGURE 2.1 – Illustration du pipeline de rendu d'une image avec un post-traitement visant à la reconstruction de l'image par le biais d'algorithmes de filtrage.

Le filtrage d'images a été une approche populaire pour supprimer le bruit dans les images générées par le tracé de rayons MC, en raison de sa simplicité et de son efficacité. Le processus commun de l'application de telles méthodes est illustré par la figure 2.1. (H. E. RUSHMEIER et al. 1994) ont proposé un filtre non linéaire qui distribue l'énergie des valeurs aberrantes (grandes valeurs de contributions) aux pixels voisins de manière à préserver l'énergie et permettre de diminuer les écarts de valeurs entre pixels voisins. (MEYER et al. 2006) ont appliqué l'analyse en composantes principales pour supprimer le bruit dans une séquence d'animation. De cette manière, les composantes les plus faibles ont été écartées permettant une suppression du bruit et un débruitage de l'image. Toutefois, ces méthodes peuvent amener à une perte de détails des séparations des objets, générant un effet plus lisse.

Classiquement, l'application d'une méthode de réduction de bruit de l'image générée est appliquée sur les valeurs d'intensité des pixels liées au plan de l'image (grille 2D de pixels). Cependant, il a été reconnu que les informations géométriques qu'un moteur de rendu peut fournir en chaque pixel (carte de normale, carte de profondeur, carte des textures, etc.), jouent un rôle important dans la bonne prédiction des bords dans les images rendues, notamment lorsque peu d'échantillons sont disponibles. (MCCOOL 1999) a ainsi proposé un processus de diffusion anisotrope (contrairement à un filtre Gaussien classique) guidé par des caractéristiques géométriques. Les caractéristiques géométriques ont également été largement utilisées pour produire rapidement une qualité d'images raisonnables dans les techniques de rendu interactif (DAMMERTZ et al. 2010 ; RITSCHEL et al. 2009 ; SEGOVIA et al. 2006 ; SHIRLEY, AILA et al. 2011).

Une méthode proposée par (SEN et DARABI 2012) tire également profit des informations géométriques. La méthode vise à supprimer le bruit MC en utilisant un processus de filtrage bilatéral lors du rendu MC d'une image. Un filtre bilatéral va permettre, contrairement à un filtre Gaussien classique de préserver les contours d'objets de l'image, car il ne prend pas uniquement en compte la distance spatiale des valeurs de pixels, mais également les paramètres géométriques de la scène.

(DELBRACIO et al. 2014) proposent d'exploiter les rayons (coordonnées et valeurs d'échantillons obtenues pour chaque rayon) de chaque pixel. En effet, chaque pixel de l'image est caractérisé par les couleurs calculées à partir de rayons ayant atteint la surface ciblée du pixel. Le filtre proposé dans cette publication utilise une distance statistique appelée histogramme des rayons pour comparer entre elles les distributions de couleurs des rayons associées à différents pixels. Sur la base de cette distance, il décide si deux pixels peuvent partager les échantillons obtenus à partir des rayons ou non et ainsi réduire considérablement le bruit de l'image.

Plus récemment, (BITTERLI et al. 2016) proposent une régression de premier ordre à pondération non linéaire, qui exploite la corrélation entre différentes informations géométriques et les échantillons obtenus. Les informations géométriques utilisées sont la carte de normales, la carte de profondeur, la carte de visibilité et la carte d'albedo (rapport de l'énergie lumineuse réfléchie à l'énergie lumineuse incidente en une surface). Ils utilisent le filtre *Non-Local Means* (BUADES et al. 2005) pour estimer les poids de la régression.

On peut aussi noter les méthodes orientées sur la réduction d'un bruit plus précis, que l'on appelle *firefly*. Un *firefly* fait référence à un pixel ayant une valeur très éloignée de ses pixels voisins. On parle ici d'une très forte contribution récupérée pour ce pixel pouvant être considérée comme un *outlier*, une donnée aberrante relativement aux autres contributions obtenues. (DECORO et al. 2010) proposent de construire un Kd-tree (arbre Kd) d'échantillons MC sur la base de leur position dans l'espace image et de leur luminosité, et de ne les accepter comme non aberrants que lorsque la densité d'échantillons dans le voisinage de chaque échantillon dépasse un certain seuil. (JUNG et al. 2015) utilisent l'estimateur *Médiane des Moyennes*, où les contributions obtenues sont décomposées en M estimateurs (moyennes) et où la moyenne des médianes (MoN) est utilisée pour supprimer le bruit de type *firefly*. Également, sur la même base d'idée que (DECORO et al. 2010), (ZIRR et al. 2018) proposent de décomposer pour chaque pixel les valeurs d'échantillons de manière stratifiée et d'associer un poids à l'échantillon relativement à sa position dans cette stratification. Ils ne suppriment donc pas la contribution d'une valeur aberrante, mais proposent plutôt de la pondérer.

(BOUGHIDA et al. 2017) ont présenté un nouvel algorithme de débruitage basé sur l'échantillonnage qui combine les avantages des histogrammes des rayons (DELBRACIO

et al. 2014) et d'un filtre de débruitage Bayesien non local (*Non-Local Bayes* en anglais) (LEBRUN et al. 2013) extension du filtre *Non-Local Means*. Le filtre *Non-Local Bayes* utilise l'estimateur MAP (Maximum A Posteriori) pour débruiter les zones de l'image. Pour calculer cet estimateur, il s'appuie sur une approche bayésienne, qui nécessite une estimation de la distribution locale. En particulier, ils conçoivent un cadre bayésien spécifique qui ne tient compte que de statistiques locales extraites de la distribution des échantillons par pixel afin de débruiter l'image en question. Un aperçu des résultats obtenus par cette méthode en comparaison avec les travaux de (DELBRACIO et al. 2014) sont disponibles dans la figure 2.2. La métrique de comparaison de qualité d'image utilisée est la SSIM (Z. WANG et al. 2004), pour *Structural SIMilarity*. Elle permet de mesurer la similarité entre deux images, où le score de 0, indique une différence totale et le score de 1, que les deux images sont identiques. Elle utilise notamment des statistiques locales des valeurs de luminance pour produire ce score. Par la suite, (VICINI et al. 2019) ont également exploité le filtre *Non-Local means* dans un contexte de rendu par profondeur où plusieurs images sont générées séparément pour une meilleure retransmission de la profondeur de l'image.

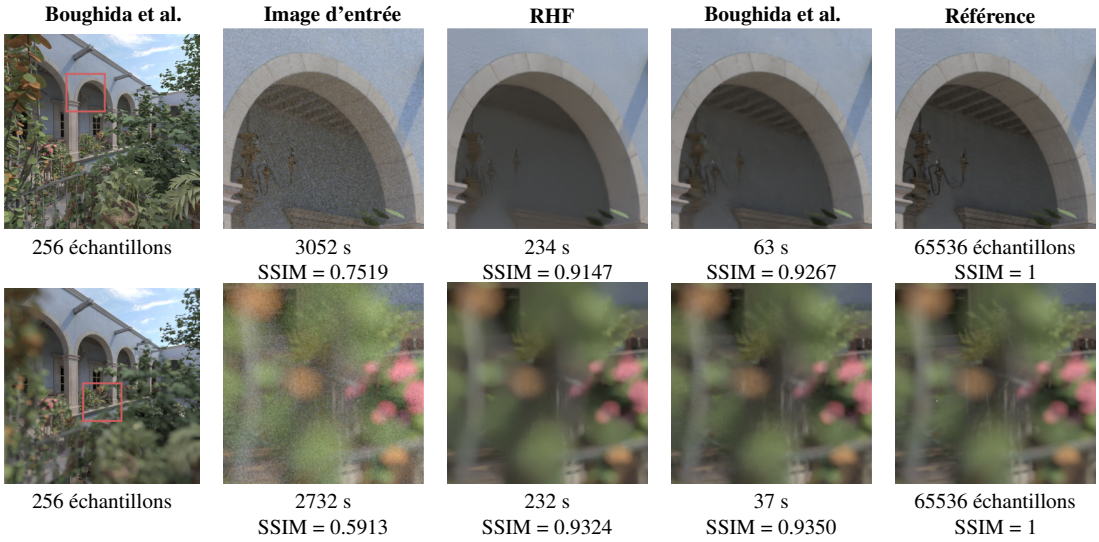


FIGURE 2.2 – Présentation des résultats de comparaisons entre l'approche Bayesienne de (BOUGHIDA et al. 2017) et celle de (DELBRACIO et al. 2014) (RHF). La SSIM est utilisée comme critère de comparaison des images obtenues à la référence. Les temps de calculs de rendu pour chaque approche sont également spécifiés. Pour chaque méthode, le filtre est appliqué à l'image d'entrée à partir des statistiques obtenues.

2.2 Reconstruction et échantillonnage adaptatif multidimensionnels

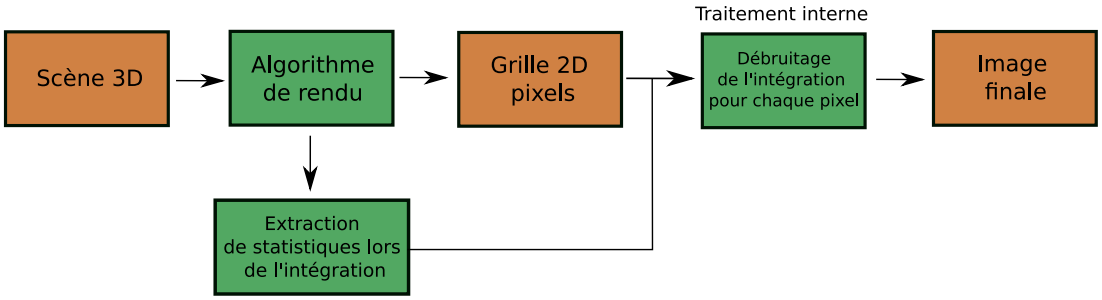


FIGURE 2.3 – Illustration du fonctionnement de méthodes de reconstruction d'image au sein d'un moteur de rendu au niveau du processus d'intégration.

D'autres méthodes ont également exploité la reconstruction d'images, mais dans l'espace des intégrandes, où la reconstruction et l'échantillonnage sont effectués dans des dimensions plus élevées que les deux dimensions du plan de l'image. La figure 2.3 illustre l'interaction de ce genre d'applications au sein d'un moteur de rendu d'images de synthèse. (HACHISUKA et al. 2008) ont proposé une méthode générale de rendu adaptatif multidimensionnel qui reconstruit des images lissées en utilisant des sommes de Riemann. L'utilisation d'un *Kd-tree* (arbre Kd) a été privilégiée pour représenter l'espace multidimensionnel des valeurs d'échantillons relativement aux coordonnées spatiales du rayon émit depuis la caméra. Ainsi, un nœud (feuille) de ce Kd-tree représente une zone de l'image comprenant les échantillons stockés et leurs coordonnées dans cette zone. À partir d'un découpage *Kd-tree* initial et de quelques échantillons (nombre d'échantillons fixé par l'utilisateur), l'algorithme se décompose ensuite en deux étapes. La première où l'on va échantillonner de manière adaptative, la seconde où l'on va reconstruire l'image de sortie.

Dans la continuité, d'autres travaux orientés sur des filtres de reconstruction partagée basés sur l'analyse de fréquence ont été conçus pour rendre efficacement les effets de profondeur de champ (SOLER et al. 2009), le flou de mouvement (EGAN, TSENG et al. 2009), les ombres douces (EGAN, HECHT et al. 2011) et les occlusions ambiantes (EGAN, DURAND et al. 2011). Plus récemment, (BELCOUR et al. 2013) ont proposé une analyse fréquentielle 5D pour traiter efficacement les effets de profondeur de champ et de flou de mouvement. (LEHTINEN, AILA, CHEN et al. 2011) ont présenté une méthode de reconstruction en utilisant les informations de profondeur et de mouvement de chaque échantillon pour synthétiser la profondeur de champ, le flou de mouvement et les ombres douces. Par la suite, (LEHTINEN, AILA, LAINE et al. 2012) ont développé un filtre

avancé qui réutilise les chemins des rayons à travers les pixels pour une reconstruction de haute qualité des illuminations indirectes.

Si les méthodes de reconstruction et d'échantillonnage multidimensionnelles ont montré des performances exceptionnelles même avec un petit nombre d'échantillons par pixel, elles se sont souvent concentrées sur des effets de rendu spécifiques. Ces méthodes possèdent donc certaines faiblesses. Nous allons maintenant présenter des approches de rendu adaptatif orientées uniquement sur l'espace de l'image, c'est-à-dire, la grille de pixels.

2.3 Rendu adaptatif de Monte-Carlo dans l'espace image

Comme mentionné précédemment, les méthodes de reconstruction et d'échantillonnage adaptatif dans le domaine de l'intégrande sont généralement spécifiques à certains effets. Les méthodes que nous présentons dans le cadre du rendu adaptatif au sein de l'espace image permettent de généraliser les effets d'éclairage et de les approximer correctement. Ces méthodes s'orientent sur un processus d'amélioration progressif. La figure 2.4 illustre un tel fonctionnement au sein d'un moteur de rendu. Nous présentons davantage ces méthodes, car nous verrons par la suite que leur fonctionnement est fortement lié aux méthodes que nous développons dans le cadre de la thèse.

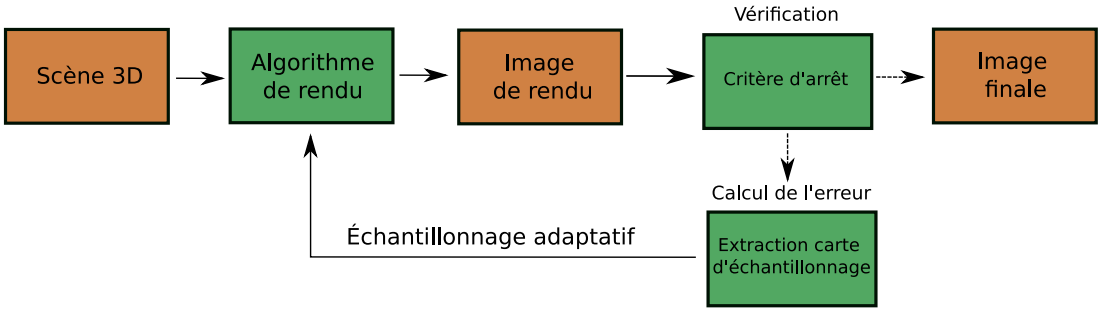


FIGURE 2.4 – Fonctionnement d'un moteur de rendu avec application d'un échantillonnage adaptatif.

(OVERBECK et al. 2009) ont développé un cadre qui traite l'échantillonnage et la reconstruction comme un processus itératif couplé, d'où le terme adaptatif. Ils décomposent l'image en ondelettes et appliquent un traitement de compression des coefficients pour réduire le bruit. Après une phase d'initialisation qui consiste à échantillonner uniformément quelques échantillons pour chaque pixel, ils distribuent ensuite de manière adaptative les échantillons de Monte-Carlo afin de réduire la variance des coefficients d'ondelettes. Ils calculent un critère de priorité d'échantillonnage des pixels en soustrayant les amplitudes des ondelettes obtenues à la variance mise à l'échelle.

(ROUSSELLE, KNAUS et al. 2011) proposent une nouvelle approche d'échantillonnage adaptatif lors du rendu de l'image de synthèse. Initialement, quelques échantillons sont estimés par pixel. Ensuite, cette approche se décompose en 2 étapes, l'une visant à estimer l'erreur quadratique moyenne (EQM (définie dans le chapitre 3 par l'équation 3.2)) après échantillonnage, la seconde cherchant à échantillonner de manière adaptative à partir de cette erreur.

En extension à leurs travaux précédents, (ROUSSELLE, KNAUS et al. 2012) proposent une nouvelle approche basée sur un schéma itératif composé de trois étapes après une phase d'échantillonnage de quelques échantillons par pixel permettant l'initialisation du processus. Premièrement, ils distribuent de manière adaptative les échantillons dans le plan de l'image. Deuxièmement, ils débruitent l'image en utilisant un filtre non linéaire. Troisièmement, à partir de l'image débruitée, une erreur résiduelle par pixel du rendu filtré est obtenue et l'estimation de l'erreur guide la distribution des échantillons dans l'itération suivante. L'efficacité de cette approche repose sur l'utilisation d'une technique de débruitage *Non-Local Means* (BUADES et al. 2005), qui est étendue à un cadre de rendu adaptatif. Ils proposent de répartir les échantillons obtenus de manière égale dans deux images calculées parallèlement. L'objectif est d'améliorer les performances de débruitage mais également de faciliter l'estimation de la variance et des erreurs.

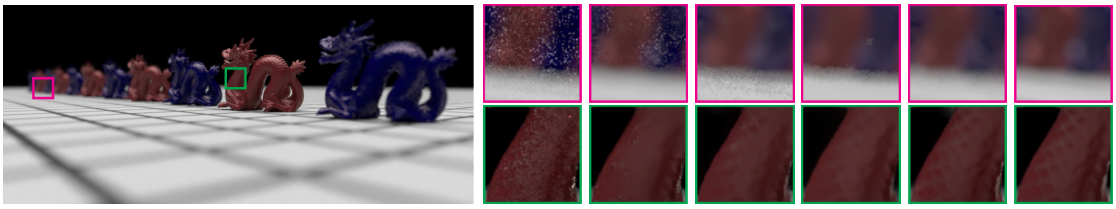
Une autre approche également intéressante est celle de (T.-M. LI et al. 2012) qui exploitent à la fois les approches de filtrage et d'informations géométriques de la scène. Ils proposent notamment d'utiliser la métrique d'erreur SURE (*Stein's Unbiased Risk Estimator*) (STEIN 1981) qui est un estimateur non biaisé de l'erreur quadratique moyenne. Cet estimateur de risque sans biais de Stein (SURE) offre un moyen d'évaluer la précision d'un estimateur donné. SURE stipule que, si y est une mesure sur x avec une distribution normale $N(x, \sigma_y^2)$ et F est une fonction faiblement différentiable, alors l'estimation de SURE est un estimateur sans biais de l'erreur quadratique moyenne (EQM) de $F(y)$. En d'autres termes, SURE fournit une indication de la précision d'un estimateur donné. Ceci est important puisque l'erreur quadratique moyenne réelle d'un estimateur est une fonction du paramètre inconnu à estimer, et ne peut donc pas être déterminée exactement.

Les auteurs exploitent un ensemble d'images filtrées (ROUSSELLE, KNAUS et al. 2011) obtenues depuis plusieurs types de filtres : *isotropic Gaussian*, bilatéral, et *Non-local means* (BUADES et al. 2005). Ils visent à intégrer les informations de la scène (SEN et DARABI 2012) comme les normales, la profondeur de champs et les textures. Leur méthode possède également deux étapes après une phase initiale d'échantillonnage. Pendant l'étape d'échantillonnage adaptatif, un ensemble d'échantillons recueille les couleurs, les normales, les textures et les profondeurs sur le plan de l'image. Ils sont utilisés comme informations secondaires pour les filtres dans l'étape de reconstruction, au cours de laquelle, pour chaque pixel, un ensemble de filtres est obtenu. Dans un second temps, la valeur filtrée avec la valeur SURE minimale est intégrée dans l'image reconstruite. À noter que le filtre bilatéral est construit relativement aux informations

de scène comme présenté dans (SEN et DARABI 2012). En outre, la valeur SURE minimale de chaque pixel est enregistrée pour guider l'échantillonnage adaptatif. Les pixels présentant des erreurs SURE plus importantes sont privilégiés pour de nouveaux échantillons. Comme les approches précédentes, la méthode est itérativement calculée. Les auteurs comparent leurs résultats avec les deux travaux suivants (ROUSSELLE, KNAUS et al. 2011 ; SEN et DARABI 2012) et montrent également une amélioration avec leur méthode.

En réponse à la nouvelle approche de (T.-M. LI et al. 2012), (ROUSSELLE, MANZI et al. 2013) exploitent également la métrique d'erreur SURE et comme (SEN et DARABI 2012) utilisent les informations géométriques pour estimer les filtres. Ils considèrent que les filtres basés uniquement sur les caractéristiques de l'image ont tendance à brouiller les détails de l'image qui ne sont pas bien représentés par les caractéristiques géométriques de la scène. D'autre part, les informations de couleurs de l'image représentent tous les détails, mais elles peuvent être moins efficaces pour déterminer les filtres, car elles sont « contaminées » par le bruit qui est censé être supprimé lors de la génération de l'image. Les auteurs proposent d'obtenir des filtres en utilisant une combinaison de ces deux types d'informations dans un cadre de filtrage via les filtres *Non-local Means* et bilatéral. Comme présenté précédemment, un filtre adapté bilatéral (SEN et DARABI 2012) va permettre de tenir compte uniquement d'informations locales alors qu'au contraire, l'algorithme du filtre *Non-Local means*, va permettre de prendre en considération des informations non locales. Les auteurs déterminent une pondération robuste des couleurs et des caractéristiques en utilisant une estimation d'erreur basée sur SURE. En tirant profit à la fois des informations locales et globales, les améliorations apportées semblent indiquer de meilleurs résultats. À noter qu'ils démontrent également l'utilisation de l'échantillonnage adaptatif et le filtrage spatio-temporel pour les animations via leur méthode.

Toujours dans la même idée, (MOON, CARR et al. 2014) proposent également une amélioration du rendu en deux étapes après initialisation : reconstruction et échantillonnage adaptatif. Ils définissent également un vecteur caractéristique $\mathbf{x} \in \mathbb{R}^D$. Ce vecteur comprend les positions de l'image ainsi qu'un ensemble d'informations géométriques supplémentaires, notamment les textures, la profondeur et les normales. Pour calculer le vecteur caractéristique à un pixel, les auteurs font la moyenne des géométries calculées à partir de plusieurs rayons primaires. Ce vecteur étant toujours fort sensible au bruit, ils identifient un espace réduit de caractéristiques locales qui guide efficacement la reconstruction basée sur une décomposition en valeur singulière tronquée (HALKO et al. 2010) (SVD en anglais pour *Singular Value Decompositon*). Cette décomposition permet de prendre en considération les k composantes principales du vecteur et de supprimer considérablement le bruit présent dans celui-ci. À partir de ces composantes principales, ils peuvent reconstruire une image dont le bruit est réduit.



(a) WLR
16 spp (66s)
rEQM 0.00316

(b) PT
39 spp (66 s)
rEQM 0.01350

(c) NLM
17 spp (66 s)
rEQM 0.00447

(d) SURE
15 spp (66 s)
rEQM 0.00678

(e) RPF
16 spp
rEQM 0.00547

(f) WLR
16 spp (66 s)
rEQM 0.00316

(g) Ref
64K spp

FIGURE 2.5 – Comparaisons de différentes approches adaptatives proposées dans la littérature sur la scène Dragon d'après (MOON, CARR et al. 2014). La référence a été calculée sur 64 000 échantillons. Le temps de rendu de l'approche RPF n'est pas connu. Le diminutif *spp* est utilisée pour *Samples per pixel*.

La figure 2.5 illustre une comparaison de la méthode de (MOON, CARR et al. 2014) comme proposé dans leur article. On la nommera ici WLR pour *Weighted Local Regression*. Ils se comparent aux méthodes précédemment citées (ROUSSELLE, KNAUS et al. 2012) (NLM), (T.-M. LI et al. 2012) (SURE), (SEN et DARABI 2012) (RPF) ainsi qu'une méthode classique de rendu de tracé de chemins (noté ici PT). Une erreur rEQM est exploitée : il s'agit de la racine carrée de l'erreur quadratique moyenne (rEQM), qui permet la comparaison pixel à pixel de l'image. NLM n'utilisant pas les informations géométriques, il semble être difficile pour cette approche dans certaines régions de discerner les caractéristiques de l'image. SURE montre des résultats trop flous sur la texture et perd du détail bien que l'approche utilise des informations géométriques. RPF quant à lui permet également de conserver du détail sur la texture, mais reste assez éloigné de la référence. WLR enfin, semble indiquer une assez bonne fidélité de rendu pour un temps identique à PT, NLM et SURE.

À noter que (SEN, Matthias ZWICKER et al. 2015) et (M. ZWICKER et al. 2015) proposent tous deux une revue des différentes avancées jusqu'à 2015 des approches d'échantillonnage adaptatif et de reconstruction de l'image. Le lecteur intéressé peut donc s'y référer.

Plus récemment, (MOON, MCDONAGH et al. 2016), appliquent une régression polynomiale relativement à un ensemble de filtres. Ils ont remarqué que les fonctions d'ordre élevé peuvent produire de meilleures images que les fonctions d'ordre faible. Ainsi, ils ont sélectionné localement l'ordre de la fonction polynomiale pour améliorer la qualité de l'image. Malgré les résultats impressionnants de ces méthodes basées sur les caractéristiques de la scène, leurs estimations finales restent bruitées à de faibles taux d'échantillonnage. Enfin, (Y. LIU et al. 2018) ont proposé d'utiliser le filtre de Sobel qui est un filtre qui permet une détection des bords des objets dans l'image. Le bruit des caractéristiques supplémentaires (composantes géométriques de la scène) est éliminé à l'aide d'un filtre d'image guidé par un opérateur de Sobel. Cet opérateur est utilisé pour

reconnaître les structures de chaque image en calculant de manière robuste une image de gradient pour chaque composante géométrique de la scène. Compte tenu des informations de gradient, des paramètres optimaux du filtre de débruitage sont obtenus. Enfin, une analyse d'erreur adaptée est réalisée pour guider le processus d'échantillonnage adaptatif. Les résultats obtenus sont meilleurs que l'approche proposée par (MOON, CARR et al. 2014), notamment sur des scènes comprenant beaucoup de détails (voir figure 2.6).

FIGURE 2.6 – Comparaisons des approches (WLR) (MOON, CARR et al. 2014) et celle de (Y. LIU et al. 2018) (SF, pour *Sobel Filter*) sur la scène *San Miguel* qui contient des géométries complexes. Les erreurs rEQM sont réalisées uniquement sur les images encadrées vertes. Le diminutif spp est utilisé pour *Samples per pixel* qui n'est ici pas précisé pour la référence (Y. LIU et al. 2018).

Dans ce chapitre, nous avons présenté un ensemble d'approches visant à réduire le bruit dans les images à partir de statistiques extraites lors du rendu. De cette manière, soit un processus post-traitement était utilisé pour débruiter l'image, soit un processus itératif était exploité pour à la fois guider le rendu de l'image et la débruiter. Ces méthodes ont permis une réduction considérable du bruit et par conséquent une convergence plus rapide de l'image pour un temps égal ou moindre. Nous allons, dans le chapitre 3, nous intéresser à des méthodes plus récentes visant une même finalité, mais exploitant l'apprentissage automatique, qui peuvent avoir l'avantage de permettre une réduction considérable du temps.

Résumé

Ce chapitre résume les derniers travaux proposés au sein de la littérature concernant l'amélioration du rendu d'images de synthèse à partir de l'analyse des données échantillonnées. Il met en avant les approches statistiques pures, exploitant ou non des informations de la scène. Il a pu être présenté qu'un processus post-traitement pouvait être appliqué pour reconstitution d'une image, mais également un processus de rendu progressif avec un échantillonnage adaptatif. Beaucoup de travaux ont été réalisés dans ce sens et ont permis de grandement améliorer la qualité des images produites pour un temps réduit. Le prochain chapitre, fortement en lien avec celui-ci, vise à présenter des méthodes d'amélioration de rendu d'images basées sur des approches d'apprentissage automatique.

Chapitre 3

Vers l'utilisation de l'apprentissage automatique pour le rendu

Introduction

Ce troisième chapitre présente diverses approches proposées au sein de la littérature visant à améliorer le rendu final d'une image de synthèse à partir de l'apprentissage automatique. Dans un premier temps, ce chapitre introduit la notion d'apprentissage, les différentes types possibles d'apprentissage et propose une liste non exhaustive de telles méthodes. Dans un second temps, les méthodes appliquées au rendu d'image sont présentées. Tout comme les travaux présentés dans le chapitre 2, ce sont généralement des processus post-traitement qui sont appliqués à la suite d'un calcul d'intégration de type tracé de chemins classique. À la différence du précédent chapitre, les méthodes présentées dans ce chapitre exploitent un modèle d'apprentissage ayant appris sur un ensemble de données visant à améliorer rapidement la qualité de l'image.

Ce chapitre présente les approches qui se sont développées récemment, orientées sur de l'apprentissage automatique (*machine learning* en anglais), où un modèle a appris à appliquer une tâche spécifique. Ces modèles sont dits *black-box*, car ils réalisent un grand nombre d'interactions qui sont souvent peu explicables. La plupart du temps ce modèle apprend sur des données en hors-ligne, puis est exploité dans un contexte en ligne lors de la génération de l'image. Il est important de préciser que ces approches sont toutes biaisées, étant donné que la valeur finale est approchée. Les travaux qui utilisent ce genre de méthodes l'appliquent dans le cadre du rendu d'image pour du débruitage mais aussi pour du rendu dit adaptatif.

3.1 Apprentissage automatique

L'apprentissage automatique, ou apprentissage machine (en anglais, *machine learning*) est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches mathématiques et statistiques pour donner aux ordinateurs la capacité d'*apprendre* à partir de données, c'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune. Plus largement, il concerne la conception, l'analyse, l'optimisation, le développement et l'implémentation de telles méthodes.

3.1.1 Définition

Le terme *apprentissage automatique* a été inventé en 1959 par Arthur Samuel, un Américain travaillant pour IBM et pionnier dans le domaine des jeux vidéo et de l'intelligence artificielle. Tom M. Mitchell a fourni une définition plus formelle (MITCHELL et al. p. d.), largement citée, des algorithmes étudiés dans le domaine de l'apprentissage automatique : « On dit d'un programme informatique qu'il apprend de l'expérience E en ce qui concerne une certaine classe de tâches T et une mesure de performance P si sa performance aux tâches de T, telle que mesurée par P, s'améliore avec l'expérience E. ». L'apprentissage automatique comporte généralement deux phases. La première consiste à estimer un modèle à partir de données, appelées observations, qui sont disponibles et en nombre fini, lors de la phase de conception du système. Nous verrons que dans certains cas ces informations ne sont pas disponibles et qu'il est nécessaire de procéder et d'apprendre autrement. La seconde, consiste à exploiter et faire appel à ce modèle pour émettre des prédictions répondant à la tâche souhaitée.

Types de modèles On peut distinguer plusieurs types de modèle en fonction de l'utilisation souhaitée. Il peut être utilisé pour diverses tâches telles que la classification d'éléments (détection ou reconnaissance) quand la sortie est une variable discrète, ou encore la régression, dans le cas où la sortie attendue est une valeur continue. Nous verrons d'ailleurs que certains modèles d'apprentissage automatique sont utilisés pour générer des images à partir d'informations d'entrée.

Applications Un modèle d'apprentissage automatique peut permettre d'apprendre à percevoir son environnement pour des tâches telles que la reconnaissance de formes (par exemple des visages), schémas, segmentation d'image, langages naturels, caractères dactylographiés ou manuscrits. Il peut être utilisé au sein des moteurs de recherche au travers d'analyse et indexation d'images et de vidéos, en particulier pour la recherche d'images par le contenu, mais également une aide aux diagnostics, médicaux notamment. Il est ces derniers temps fortement utilisé pour le traitement du langage, notamment pour des tâches de classification de texte, de traduction et de génération de texte. On peut aussi l'utiliser au sein de processus de cybersécurité, ainsi que pour de l'analyse financière, dont l'analyse du marché boursier (notamment pour de la prédiction du cours

des actions).

3.1.2 Types d'apprentissage

On peut énumérer plusieurs types d'apprentissage automatique, intimement liés à un type de tâche attendu par un tel modèle.

3.1.2.1 Apprentissage supervisé

À partir d'un ensemble de données d'apprentissage de taille N formé de paires de données d'entrée et de sortie : $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ où y_i a été généré par une fonction inconnue $y = f(x)$, le modèle doit découvrir une fonction h qui approxime au mieux la fonction réelle f. Un exemple de modèle est celui de la régression logistique proposée par Berkson (BERKSON 1944), qui permet de classifier de façon binaire des données.

3.1.2.2 Apprentissage non supervisé

À partir d'un ensemble de données d'apprentissage de taille N formé de données d'entrée uniquement : $(x_1), (x_2), \ldots, (x_N)$, le modèle doit découvrir par lui-même une structure plus ou moins cachée des données. On dénote souvent les tâches de ce genre d'algorithme comme étant le partitionnement de données (*clustering* en anglais). Les K-moyens (*K-means* en anglais) est un exemple d'algorithme non supervisé, où la sortie de l'algorithme est un groupe d'étiquettes. Il attribue pour chacune des étiquettes un point x_i de référence comme étant le centroïde du groupe k_i. Chaque groupe de données est ainsi défini en créant un centroïde pour chacun de ces groupes. Les centroïdes sont comme le cœur du cluster, qui capture les points les plus proches d'eux et les ajoute au cluster. Les centroïdes sont mis à jour relativement à l'ensemble des données et la mesure de distance choisie en ces données. Il existe bien d'autres algorithmes d'apprentissage non supervisé tels que ceux orientés sur la réduction de dimension des données comme l'analyse en composante principale PCA (JOLLIFFE 2005), la décomposition en valeurs singulières (SVD) (G.H.GOLUB et al. 1983 ; O.ALTER et al. 2000) mais aussi ceux cherchant une estimation de densité de distribution des données, comme l'estimation par noyau (KDE) (TERRELL et al. 1992).

3.1.2.3 Apprentissage semi-supervisé

À partir d'un ensemble de données d'apprentissage de taille N formé de données partiellement labellisées : $(x_1), (x_2, y_2), \ldots, (x_{N-1}), (x_N, y_N)$, le modèle vise à faire apparaître la distribution sous-jacente des exemples dans leur espace de description de manière probabiliste ou non.

Par exemple, les approches génératives basées sur l'apprentissage statistique cherchent d'abord à estimer $p(x|y)$ la distribution des points de données appartenant à chaque classe. La probabilité $p(y|x)$ qu'un point x possède le label y est alors proportionnel à $p(x|y)p(y)$ selon le théorème de Bayes. La classification naïve bayésienne est un type de classification bayésienne probabiliste simple basée sur le théorème de Bayes (RISH 2001).

3.1.2.4 Apprentissage par renforcement

Le modèle apprend ici un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit une valeur de retour (récompense) qui guide l'algorithme d'apprentissage. Cette récompense peut être négative ou positive en fonction de l'attendu au travers de l'action. L'algorithme va donc chercher à maximiser sa récompense et apprendre de ses actions précédentes. Le *Q-learning* (WATKINS et al. 1992) par exemple, est un algorithme d'apprentissage par renforcement sans modèle pour apprendre la valeur d'une action dans un état particulier.

3.1.2.5 Apprentissage par transfert

L'apprentissage par transfert met en avant la capacité d'un système à reconnaître et appliquer des connaissances et des compétences. À partir d'un modèle dont la tâche antérieurement apprise est différente, mais similaire à une nouvelle tâche souhaitée, il est possible de l'affiner et ainsi l'adapter à cette nouvelle tâche.

3.1.3 Score de performance d'un modèle

Il est important de bien identifier la performance d'un modèle après avoir réalisé son apprentissage dans un cas d'apprentissage supervisé. Cela est généralement le cas dans un contexte d'apprentissage supervisé, voire semi-supervisé, car il est nécessaire de posséder des étiquettes pour évaluer la performance d'un modèle. Des métriques connues permettent de mesurer la performance du modèle ; elles sont adaptées pour chacun des types de modèle, classification et régression.

3.1.3.1 Classification

Nous nous limiterons ici à présenter les mesures dans un cadre de classification binaire. Pour l'ensemble des prédictions binaires d'un modèle, on note VP = Vrais positifs, VN = Vrais négatifs, FP = Faux positifs, et FN = Faux négatifs. Voici une liste non exhaustive de métriques de performance de tels modèles :

— *justesse*, en anglais *Accuracy* : la justesse peut être calculée en termes de positifs et de négatifs comme suit :

$$\text{Justesse} = \frac{VP + VN}{VP + VN + FP + FN}$$

— *précision*, permet de connaître la proportion d'identifications positives qui était effectivement correcte :

$$\text{Précision} = \frac{VP}{VP + FP}$$

— *rappel* ou sensibilité, permet de définir la proportion de résultats positifs réels qui a été identifiée correctement :

$$\text{Rappel} = \frac{VP}{VP + FN}$$

— *AUC ROC*, est la mesure numérique entre 0 et 1 de l'aire sous la courbe ROC (*receiver operating characteristic*). (BRADLEY 1997). Cette courbe est initialement connue pour être un outil statistique qui découle de la théorie de la détection du signal. Elle a ensuite été utilisé dans de nombreux domaines (biologie, médical, psychologie, etc.) dès lors qu'il faut quantifier la performance d'un test. Elle est donc naturellement utilisée pour mesurer l'efficacité d'un classifieur. Ainsi, elle représente les performances d'un modèle de classification pour tous les seuils de classification : valeur de seuil de détermination de la classe d'appartenance (classiquement $0,5$ si la sortie du classifieur est comprise entre 0 et 1). Cette courbe trace le taux de vrais positifs (TVP) en fonction du taux de faux positifs (TFP) :

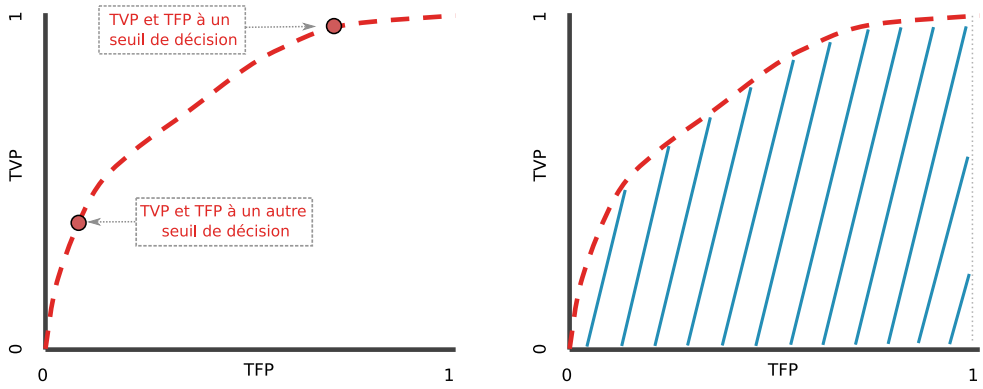
$$TVP = \frac{VP}{VP + FN} \qquad TFP = \frac{FP}{FP + VN}$$

Un aperçu d'une telle courbe est disponible dans la figure 3.1. L'AUC ROC permet de donner un score de performance du modèle, car plus l'aire est grande, plus le modèle sait correctement catégoriser les étiquettes quel que soit le seuil de classification.

3.1.3.2 Régression

Dans un contexte de régression, les sorties sont des valeurs numériques continues. Voici une liste non exhaustive de mesures de bonnes prédictions de tels modèles :

— *EMA*, aussi appelée erreur *L1*, l'erreur moyenne absolue (en anglais MAE, pour *Mean Absolute Error*), donne un score moyen de la valeur absolue de la différence de chacune des prédictions relativement aux valeurs attendues. Sa formule est la suivante :



(a) Courbe ROC (b) Aire sous la courbe ROC

FIGURE 3.1 – La sous-figure 3.1a représente la courbe ROC pour un problème de classification binaire. La sous-figure 3.1b illustre l'aire sous cette courbe.

$$\text{EMA} = \frac{\sum_{i=1}^{n} \left| y_i - y_i' \right|}{n} \tag{3.1}$$

où y_i est la prédiction attendue et y_i' la prédiction fournie par le modèle pour le jeu de données x_i, et n est le nombre de mesures ;

— *EQM*, aussi appelée erreur *L2*, l'erreur quadratique moyenne (en anglais MSE, pour *Mean Square Error*), donne un score moyen de la différence élevée au carré de chacune des prédictions relativement à l'attendue. Sa formule est la suivante :

$$\text{EQM} = \frac{\sum_{i=1}^{n} \left(y_i - y_i' \right)^2}{n} \tag{3.2}$$

où y_i est la prédiction attendue et y_i' la prédiction par le modèle pour le jeu de données x_i, et n est le nombre de mesures ;

— *rEQM*, la racine de l'erreur quadratique moyenne est une extension de l'EQM, où simplement l'erreur obtenue est la racine de l'EQM, soit $\sqrt{EQM}$;

— R^2, le coefficient de détermination linéaire de Pearson (WALKER 1958), noté R^2 ou r^2, est une mesure de la qualité de la prédiction d'une régression linéaire. Il est calculé de la manière suivante :

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \left(y_i - y_i' \right)^2}{\sum_{i=1}^{n} \left(y_i - \bar{y} \right)^2} \tag{3.3}$$

où n est le nombre de mesures, y_i la prédiction attendue, y_i' la valeur prédite

correspondante par le modèle et $\bar{y}$ la moyenne des n mesures.

3.1.4 Surapprentissage et généralisation

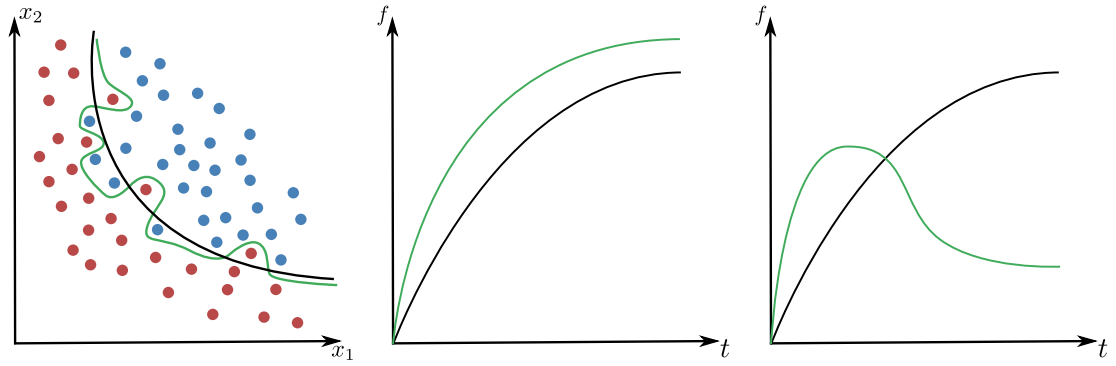
Lors de la phase d'apprentissage d'un modèle, notamment en apprentissage supervisé, on va séparer nos données au minimum en deux sous-ensembles. Celui d'apprentissage, qui constitue généralement 75% de l'ensemble des données et le reste dans un sous-ensemble que l'on appelle ensemble de validation.

Le modèle va donc apprendre à partir des données d'apprentissage, pour lesquelles sa performance sera évaluée. Il sera ensuite testé sur le sous-ensemble de données de validation. Pour qu'un modèle soit correct, il faut qu'il réponde en moyenne aussi bien sur le sous-ensemble d'apprentissage et sur le sous-ensemble de validation. C'est-à-dire, que sa performance (son score d'évaluation des prédictions) soit aussi élevé sur le sous-ensemble d'apprentissage que sur le sous-ensemble de validation.

3.1.4.1 Surapprentissage

Le surapprentissage est un problème assez courant de l'apprentissage automatique. Lors de la phase d'apprentissage, le modèle va se perfectionner sur les données qu'on lui propose jusqu'à « sur apprendre », c'est-à-dire, se « spécialiser » pour cet ensemble de données. La figure 3.2 illustre ce problème, avec deux modèles. Le modèle représenté par les courbes vertes va prendre en compte de manière correcte les données qui lui sont fournies lors de l'apprentissage (3.2a) et fournira de bons résultats lors des tests qui seront appliqués sur cette base d'apprentissage (3.2b). À l'inverse, le modèle représenté par les courbes noires commet quelques erreurs (3.2a), qui se traduisent par des performances moindres lors de son évaluation sur la même base (3.2b). Cependant, lors de l'évaluation des deux modèles sur des données de test, non connues de ceux-ci, on peut avoir un effet inverse, avec des performances moindres du premier modèle par rapport au second (3.2c). La spécialisation associée au premier modèle ne lui permet pas de généraliser correctement ses prédictions sur des données non connues, contrairement au second modèle qui fournit de bien meilleurs résultats. La généralisation est un principe très important de l'apprentissage automatique puisque l'on cherche à faire en sorte que notre modèle soit performant sur l'ensemble des données, celles de l'apprentissage et de validation (base de test). Ainsi, un modèle qui généralise est un modèle qui permet de mieux réduire ce que l'on appelle le risque « réel », risque lié à l'ensemble des données, comprenant celles dont le modèle ne connaît pas encore l'existence.

Dans le cadre de l'apprentissage profond, la notion de *dropout* permet de spécifier un pourcentage de neurones (neurones sélectionnés aléatoirement) d'une couche du réseau où l'erreur n'est pas propagée. L'ajout de *dropout* permet au modèle de réseaux de neurones d'accroître sa capacité de généralisation et d'éviter le surapprentissage. Tous



(a) Performance des modèles depuis sur les données d'apprentissage

(b) Courbe de performance des modèles durant l'apprentissage

(c) Courbe de performance des modèles sur les données de validation

FIGURE 3.2 – Aperçu d'un problème de surapprentissage. On souhaite séparer les données d'étiquettes rouges, des données d'étiquettes bleues. Deux modèles sont proposés, le modèle vert, qui ne commet aucune erreur sur la base d'apprentissage. Le modèle noir qui lui commet quelques erreurs, mais prédit plutôt correctement. Pour les sous-figures b et c, f représente la fonction d'évaluation de performance d'un modèle et t le temps d'apprentissage. Ainsi, les courbes représentent l'évolution du score d'apprentissage et de validation lors de l'avancée de la phase d'apprentissage.

les neurones n'ont pas eu une connaissance globale de chaque donnée dont l'erreur est propagée.

3.1.4.2 Validation croisée

La validation croisée est un processus utilisé pour détecter un surapprentissage ; on sépare les données en k sous-ensembles où $k-1$ ensembles sont utilisés pour l'apprentissage et le dernier ensemble pour valider les performances du modèle. L'ensemble de validation est utilisé pour vérifier la pertinence du modèle et de ses paramètres. Ce processus est répété k fois en changeant l'ensemble de validation à chaque fois. Il est ainsi possible de sélectionner les paramètres d'un modèle étant meilleur globalement en moyenne sur les k itérations.

3.1.5 Machines à vecteurs de support

Les machines à vecteurs de support (en anglais *Support Vector Machine*, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de classification et de régression. Les SVM sont une généralisation des classifieurs linéaires, puisqu'ils visent à séparer les données sur de grandes dimensions à l'aide

d'un hyperplan. Nous détaillons ce modèle, car il est couramment utilisé au sein de la littérature.

3.1.5.1 Définition et idée

Les SVM ont été développés dans les années 1990 par Vladimir Vapnik sur le développement de la théorie statistique de Vapnik-Chervonenkis (HEARST et al. 1998). Les SVM ont rapidement été adoptés pour leur capacité à travailler avec des données de grandes dimensions, le faible nombre d'hyperparamètres, leurs garanties théoriques et leurs bons résultats en pratique. Ils sont notamment très robustes lorsque l'ensemble des données n'est pas trop conséquent.

La résolution des problèmes de classification passe par la construction d'une fonction h qui, à un vecteur d'entrée x, fait correspondre une sortie y : $y = h(x)$. Le cas le plus simple est le cas d'une fonction de classification linéaire, obtenue par combinaison linéaire du vecteur d'entrée $x = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$, avec un vecteur de poids $w = (\mathbf{w}_1, \ldots, \mathbf{w}_N)^T$. Pour cette fonction, on obtient donc :

$$h(x) = w^T x + w_0$$

Avec w_0 un nombre réel qui va nous permettre de travailler avec un hyperplan affine. Il est alors décidé que x est d'étiquette 1 si $h(x) \geq 0$ et d'étiquette -1 sinon. C'est un classifieur linéaire. La frontière de décision $h(x) = 0$ est un hyperplan, appelé hyperplan séparateur. Dans un espace vectoriel de dimension finie N, un hyperplan est un sous-espace vectoriel de dimension $N - 1$. Ainsi, dans un espace de dimension 2 un hyperplan sera une droite, dans un espace dimension 3 un hyperplan sera un plan, etc.

Si le problème est linéairement séparable, on doit alors avoir :

$$l_k h(x_k) \geq 0 \quad \text{soit} \quad l_k \left(w^T x_k + w_0 \right) \geq 0 \tag{3.4}$$

avec p la taille de la base d'apprentissage, l_k, le label associé aux données d'entrée x_k et $1 \leq k \leq p$.

3.1.5.2 Notion de marge maximale

La marge maximale permet un choix plus optimal de l'hyperplan séparateur des données. En effet, le choix de l'hyperplan séparateur n'est pas évident. Il existe potentiellement une infinité d'hyperplans séparateurs, dont les performances en apprentissage sont identiques (le risque empirique est le même), mais dont les performances en généralisation peuvent être très différentes. Pour résoudre ce problème, il a été montré (VAPNIK 2006) qu'il existe un unique hyperplan optimal, défini comme l'hyperplan qui maximise la marge entre les échantillons et l'hyperplan séparateur.

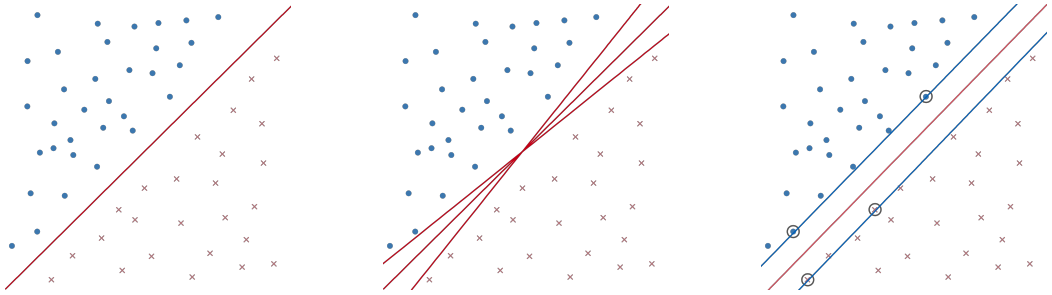
La marge est la distance entre l'hyperplan et les échantillons les plus proches. Ces derniers sont appelés vecteurs supports. L'hyperplan qui maximise la marge est donné par :

$$\arg\max_{w,w_0} \min_{k} \left\{ \|x - x_k\| : x \in \mathbb{R}^N, w^T x + w_0 = 0 \right\}$$

L'objectif est donc de trouver w et w_0 remplissant ces conditions, afin de déterminer l'équation de l'hyperplan séparateur :

$$h(x) = w^T x + w_0 = 0$$

La notion d'hyperplan séparateur dans le cas d'une classification est très importante. La figure 3.3 propose une représentation d'un modèle SVM et de sa marge maximale.



(a) Séparateur linéaire classique (b) Possibilités pour séparateur linéaire classique (c) Machine à vecteur support et sa marge maximale

FIGURE 3.3 – Représentation de la robustesse d'un modèle de type machine à vecteur support dans un contexte de classification binaire sur des données à deux dimensions. Les données sont représentées par leurs labels soit par un rond bleu, soit par une croix rouge. À noter ici que les données utilisées sont très facilement séparables.

Formulation primale :

La marge est la plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur qui satisfasse la condition de séparabilité (voir équation 3.4) La distance d'un échantillon x_k à l'hyperplan est donnée par sa projection orthogonale sur le vecteur de poids :

$$\frac{l_k \left(w^T x_k + w_0 \right)}{\|w\|} \tag{3.5}$$

Ainsi, l'hyperplan séparateur de marge maximale s'exprime également par :

$$\arg\max_{w,w_0}\left\{\frac{1}{\|w\|}\min_k\left[l_k\left(w^T x_k + w_0\right)\right]\right\} \tag{3.6}$$

La recherche de l'hyperplan séparateur de marge maximale est un problème d'optimisation qui consiste à maximiser $\|w\|^{-1}$ étant donné que la marge vaut $\frac{1}{\|w\|}$. La formulation dite primale des SVM s'exprime alors sous la forme suivante :

$$\begin{aligned} &\text{Minimiser } \frac{1}{2}\|w\|^2 \\ &\text{sous les contraintes} \quad l_k\left(w^T x_k + w_0\right) \geq 1 \end{aligned}$$

Ceci peut se résoudre par la méthode classique des multiplicateurs de Lagrange (CRAVEN 1970), où le lagrangien est donné par :

$$L(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{k=1}^{p}\alpha_k\left\{l_k\left(w^T x_k + w_0\right) - 1\right\} \tag{3.7}$$

Le lagrangien doit être minimisé par rapport à w et w_0, et maximisé par rapport à α.

Formulation duale :

En annulant les dérivées partielles du lagrangien, selon les conditions de Kuhn-Tucker (BENNETT et al. 2000 ; GUESTRIN 2006), on obtient :

$$\begin{cases} \sum_{k=1}^{p}\alpha_k l_k x_k &= w^* \\ \sum_{k=1}^{p}\alpha_k l_k &= 0 \end{cases}$$

En réinjectant ces valeurs dans l'équation 3.7, on obtient la formulation duale :

$$\begin{aligned} &\text{Maximiser } \tilde{L}(\alpha) = \sum_{k=1}^{p}\alpha_k - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j l_i l_j x_i^T x_j \\ &\text{sous les contraintes } \alpha_k \geq 0, \quad \text{et} \quad \sum_{k=1}^{p}\alpha_k l_k = 0 \end{aligned} \tag{3.8}$$

Ce qui permet d'obtenir les multiplicateurs de Lagrange optimaux α_k^*. On remplace ainsi w par sa valeur optimale w^* dans l'équation de l'hyperplan $h(x)$ pour d'obtenir l'hyperplan optimal :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k \left(x \cdot x_k\right) + w_0 \tag{3.9}$$

3.1.5.3 Cas non séparable

La notion de marge maximale et la procédure de recherche de l'hyperplan ne permettent de résoudre que des problèmes de classification linéairement séparables. Le SVM ne peut alors résoudre que des problèmes simples, ou très particuliers. Afin de remédier au problème de l'absence de séparateur linéaire, l'idée de l'astuce du noyau (en anglais *kernel trick*) est de reconsidérer le problème dans un espace de dimension supérieure, éventuellement de dimension infinie. Dans ce nouvel espace, il est alors probable qu'il existe une séparation linéaire. Plus formellement, on applique aux vecteurs d'entrée x une transformation non-linéaire ϕ. L'espace de représentation obtenue $\phi(X)$ est appelé espace de redescription. Dans cet espace, on cherche alors l'hyperplan :

$$h(x) = w^T \phi(x) + w_0 \tag{3.10}$$

En utilisant la même procédure que dans le cas sans transformation (voir équation 3.8), on aboutit au problème d'optimisation suivant :

$$\begin{aligned} &\text{Maximiser } \tilde{L}(\alpha) = \sum_{k=1}^{p} \alpha_k - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j l_i l_j \phi\left(x_i\right)^T \cdot \phi\left(x_j\right) \\ &\text{sous les contraintes } \alpha_k \geq 0, \quad \text{et} \quad \sum_{k=1}^{p} \alpha_k l_k = 0 \end{aligned} \tag{3.11}$$

Le problème de cette formulation est qu'elle implique un produit scalaire entre vecteurs dans l'espace de redescription, de dimension élevée, ce qui est coûteux du point de vue des calculs. Pour résoudre ce problème, on utilise une astuce connue sous le nom de l'astuce du noyau, qui consiste à utiliser une fonction noyau, qui vérifie :

$$K\left(x_i, x_j\right) = \phi\left(x_i\right)^T \cdot \phi\left(x_j\right) \tag{3.12}$$

L'intérêt de la fonction noyau est double. Tout d'abord, le calcul se fait dans l'espace d'origine, ceci est beaucoup moins coûteux qu'un produit scalaire en grande dimension. Deuxièmement, la transformation ϕ n'a pas besoin d'être connue explicitement, seule la fonction noyau intervient dans les calculs. On peut donc envisager des transformations complexes, et même des espaces de redescription de dimension infinie.

L'hyperplan séparateur avec fonction noyau peut ainsi être formulé de la manière suivante :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k K(x_k, x) + w_0 \tag{3.13}$$

Il existe plusieurs choix de noyaux pour projeter les données dans l'espace $\phi(X)$ tels que les noyaux linéaire, polynomial ou encore Gaussien :

— linéaire : $K\left(x_i, x_j\right) = x_i^T \cdot x_j$

— polynomial : $K\left(x_i, x_j\right) = \left(x_i^T \cdot x_j + 1\right)^d$

— gaussien : $K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$

— *radial basis function* (RBF) : $K\left(x_i, x_j\right) = e^{-\gamma\|x_i - x_j\|^2}$

À noter que le noyau RBF possède un paramètre $\gamma \in \mathbb{R}$ qui définit l'influence d'un seul exemple (point) de la base d'apprentissage dans l'espace de redescription.

3.1.5.4 Marge souple

En général, il n'est pas possible de trouver une séparatrice linéaire dans l'espace de redescription. Il est aussi possible que des échantillons soient mal étiquetés et que l'hyperplan séparateur ne soit en réalité pas la meilleure solution au problème de classification.

Pour pallier ce problème, (CORTES et al. 1995) ont proposé une technique dite de « marge souple ». L'idée n'est pas de proposer un séparateur qui va classer et séparer parfaitement tous les points, mais qui peut tolérer quelques erreurs. Ils proposent ainsi une technique qui cherche un hyperplan qui minimise le nombre d'erreurs grâce à l'utilisation de « variables ressort » ξ_k, qui permettent de relâcher les contraintes sur les vecteurs d'apprentissage :

$$l_k\left(w^T x_k + w_0\right) \geq 1 - \xi_k \quad \xi_k \geq 0, \quad 1 \leq k \leq p$$

Avec les contraintes précédentes, le problème d'optimisation est modifié par un terme de pénalité, qui pénalise les variables ressort élevées :

$$\text{Minimiser } \frac{1}{2}\|w\|^2 + C\sum_{k=1}^{p} \xi_k \quad , \quad C > 0$$

où C est une constante qui permet de contrôler le compromis entre le nombre d'erreurs de classement et la largeur de la marge. C'est l'utilisateur qui doit fixer la valeur de C, en général par une recherche exhaustive dans l'espace des paramètres, avec l'utilisation de la validation croisée pour éviter le surapprentissage.

3.1.6 Réseau neuronal

Les réseaux neuronaux artificiels sont des systèmes informatiques à l'origine inspirés des réseaux neuronaux biologiques. Ces systèmes apprennent à effectuer des tâches en considérant des exemples, généralement sans être programmés avec des règles spécifiques à la tâche.

Définition et représentation Un réseau de neurones est un modèle basé sur une collection d'unités ou de nœuds connectés appelés *neurones artificiels*, qui modélisent les neurones d'un cerveau biologique. Chaque connexion, comme les synapses du système nerveux, peut transmettre des informations, que l'on appellera ici *signal*, d'un neurone artificiel à un autre. Un neurone artificiel qui reçoit un signal peut le traiter, puis envoyer un signal aux autres neurones artificiels qui lui sont connectés.

3.1.6.1 Perceptron

Le perceptron est un algorithme d'apprentissage supervisé de classifieurs binaires, c'est-à-dire séparant deux classes. Il a été inventé en 1957 par Frank Rosenblatt (ROSENBLATT 1958). Le perceptron peut être vu comme le type de réseau de neurones le plus simple. C'est un classifieur linéaire. Un perceptron possède un vecteur d'entrée x de n attributs $x = (x_1, \ldots, x_n)$ et la seule sortie o est définie par :

$$o = f(x) = \begin{cases} 1 & \text{si} \quad \sum_{i=1}^{n} w_i x_i > \theta \\ 0 & \text{sinon} \end{cases} \tag{3.14}$$

où $w = (w_1, \ldots, w_n)$ sont n poids (ou coefficients synaptiques) et θ un biais (ou seuil). La figure 3.4 propose une représentation d'un tel modèle.

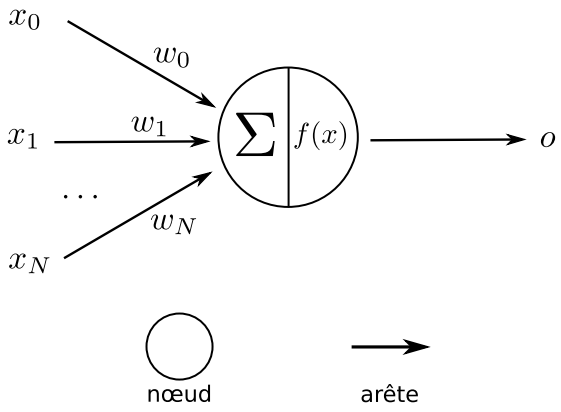


FIGURE 3.4 – Représentation d'un modèle perceptron et de ses arêtes

3.1.6.2 Perceptrons multicouches

Dans les implémentations courantes des réseaux de neurones, la connexion entre neurones artificiels est représentée par un nombre réel et la sortie de chaque neurone artificiel est calculée par une fonction non linéaire de la somme de ses entrées. Les connexions entre les neurones artificiels sont appelées *arêtes*. Les neurones artificiels et les arêtes ont généralement un poids qui s'ajuste au fur et à mesure de l'apprentissage. Le poids augmente ou diminue l'intensité du signal au niveau d'une connexion. Les neurones artificiels peuvent avoir un seuil tel que le signal n'est envoyé que si le signal agrégé franchit ce seuil. En général, les neurones artificiels sont regroupés en couches (voir figure 3.5). Les différentes couches peuvent effectuer différents types de transformations sur leurs entrées. Les signaux voyagent de la première couche (la couche d'entrée) à la dernière couche (la couche de sortie).

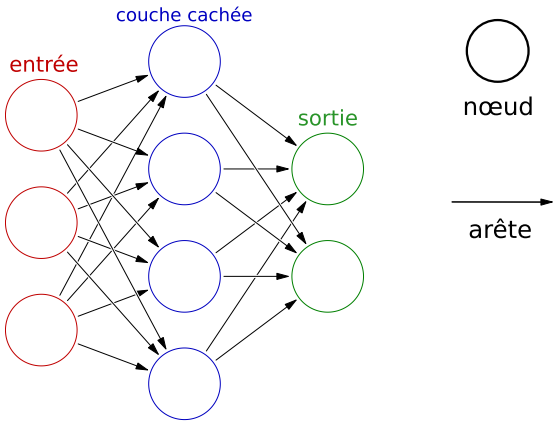


FIGURE 3.5 – Représentation d'un réseau de neurones composés d'un ensemble de nœuds (neurones) et de connexion entre ces nœuds. Il est ici composé d'une couche d'entrée, une couche dite cachée et une couche de sortie. La couche de sortie contient ici deux neurones et retourne donc deux valeurs numériques.

3.1.6.3 Fonction de perte

En optimisation mathématique et en théorie de la décision, une fonction de perte ou une fonction de coût (parfois aussi appelée fonction d'erreur) est une fonction qui met en correspondance un événement ou les valeurs d'une ou plusieurs variables avec un nombre réel représentant intuitivement un certain *coût* associé à cet événement. Un problème d'optimisation cherche à minimiser une fonction de perte. La fonction « objectif » est soit une fonction de perte, soit la valeur négative de cette fonction (dans certains domaines, elle est appelée fonction de récompense, fonction de profit, fonction d'utilité, fonction de fitness, etc.). Dans le cadre de l'apprentissage automatique, on utilise cette fonction de

perte pour mettre à jour les poids du réseau de neurones. Par exemple, la mise à jour d'un poids w_j lors de l'apprentissage d'un perceptron est réalisée de la manière suivante :

$$w_i = w_i + \alpha(y - f(x))x_i$$

où y est la sortie attendue, $f(x)$ la sortie obtenue avec les données d'entrée x, la valeur x_i est la donnée associée au poids w_i et α, le pas d'apprentissage. La fonction de perte est ici simplement représentée par l'erreur relative entre la sortie du réseau et l'attendue. On peut utiliser des fonctions de perte relatives à l'erreur de prédiction telles que l'EQM pour un problème de régression.

La notion de lot est utilisé dans le cadre de l'apprentissage d'un modèle. En effet, les modèles à base de réseaux de neurones sont très performants, encore plus lorsque la quantité de données d'apprentissage est conséquente. Toutefois, il n'est pas toujours possible de calculer en une fois l'erreur relative à la fonction de perte pour l'ensemble des données, car il peut être impossible de charger en mémoire l'ensemble des données d'apprentissage. Cela est généralement le cas quand la donnée traitée est une image. Pour cela, on préfère utiliser un lot, de k données (généralement un petit nombre de données) en entrée au modèle. Puis, relativement aux sorties prédites par le modèle sur ces k données, l'erreur est calculée avec la fonction de perte et les poids du réseau mis à jour itérativement.

3.1.6.4 Réseau convolutif

Un réseau de neurones convolutif (CNN, pour *Convolutional Neural Network*) est un type de réseau de neurones artificiels acycliques (les données sont propagées de l'entrée vers la sortie), dans lequel le motif de connexion entre les neurones est inspiré par le cortex visuel des animaux (O'SHEA et al. 2015). Il traite plus facilement des images en entrées que des réseaux de neurones multicouches classiques, qui sont sujets à une augmentation de la dimension et donc du nombre de poids à mettre à jour. Il possède des couches dites de convolutions (voir figure 3.6). Chaque couche de convolution propose de découper l'image en petites zones (appelées tuiles) et d'y extraire des informations locales. Chaque tuile sera traitée individuellement par un neurone artificiel qui effectue une opération de filtrage classique en associant un poids à chaque pixel de la tuile. Ce traitement s'applique par convolution, ce qui veut dire qu'un processus d'extraction d'informations par tuiles (par exemple de taille 3×3 pixels) peut-être appliqué en décalant uniquement d'un pixel. La couche suivante comporte donc une dimension de données plus réduite et peut également se voir appliquer un processus convolutif. Généralement, après avoir appliqué quelques couches de convolution, un réseau de neurones multicouches (dont les données d'entrée sont un vecteur) est ensuite présent pour obtention de la sortie désirée.

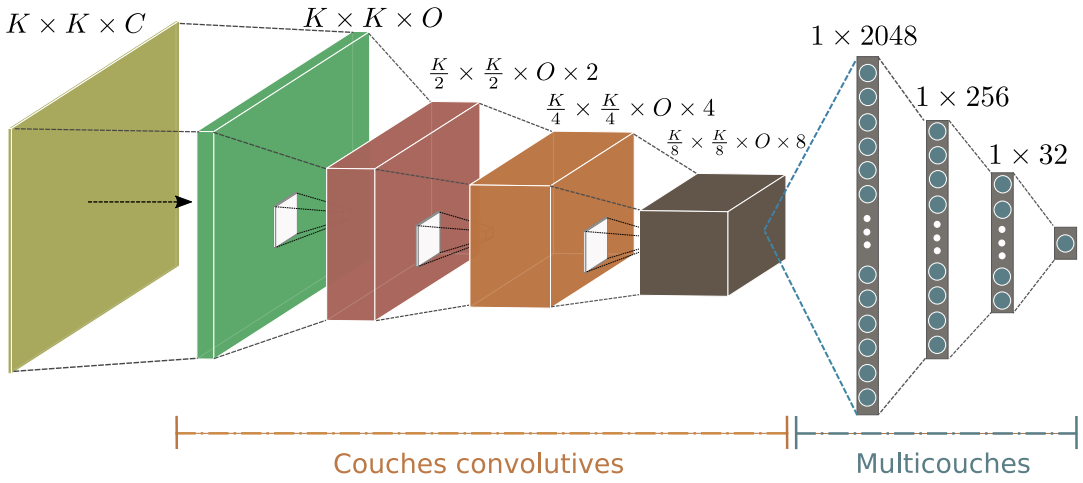


FIGURE 3.6 – Représentation d'un réseau de neurones comprenant des couches convolutives pour extraction de paramètres avant passage des informations traitées à un réseau multicouches classique. Dans ce schéma, pour chaque sortie, une couche de convolution extrait un nombre double de tuiles qui sont réduites en taille par deux. La dernière couche correspond à celle de sortie où ici une probabilité d'appartenance à une classe binaire est obtenue. La dimension de profondeur de l'entrée est représentée par C, et la dimension de profondeur des couches de sortie est déterminée en fonction de O.

3.1.6.5 AutoEncoder

Un *AutoEncoder* est un type de réseau de neurone convolutif ou non, qui va prendre en entrée une donnée, la compresser puis apprendre à la décompresser. Il permet dans sa première partie de réduire les dimensions de la donnée et d'en extraire les informations les plus importantes pour enfin reconstruire la donnée presque à l'identique. Ils ont été introduits par (HINTON et al. 2006) et possèdent un grand nombre d'applications et d'extensions (KINGMA et al. 2014 ; LIOU et al. 2014). Un schéma de ces réseaux est disponible dans la figure 3.7.

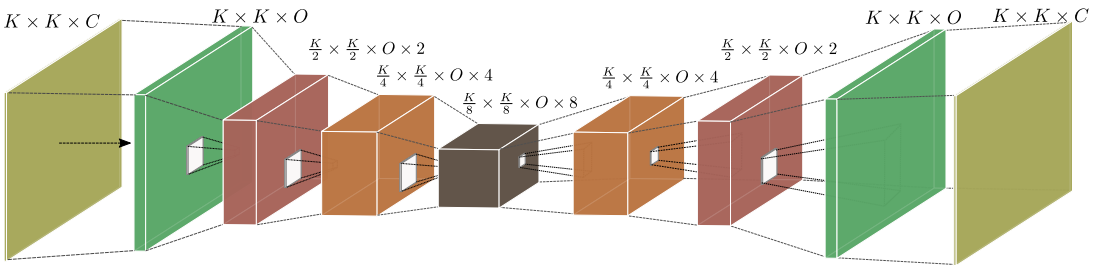


FIGURE 3.7 – Réseau *AutoEncoder* qui prend une image de taille $K \times K \times C$ sur les trois canaux de couleurs RGB, qui va l'encoder puis la décoder, afin de la restituer.

Les réseaux génératifs contradictoires (GAN, pour *Generative Adversarial Network* en anglais) (GOODFELLOW et al. 2014) sont des modèles qui vont être composés de deux réseaux. Le premier réseau G, pour générateur, proposant de générer une donnée à partir de données d'entrée aléatoires, mais où l'on connait l'attendue, par exemple une image. Le second D, pour discriminateur, va lui prendre en entrée soit une image obtenue par le réseau G, soit l'image de référence connue. Ainsi, il va devoir apprendre à distinguer laquelle des deux images a été générée par G est la référence. La force d'une telle architecture est que les deux réseaux s'affrontent tout en apprenant chacun l'un de l'autre. La figure 3.8 donne un aperçu du fonctionnement d'un tel modèle.

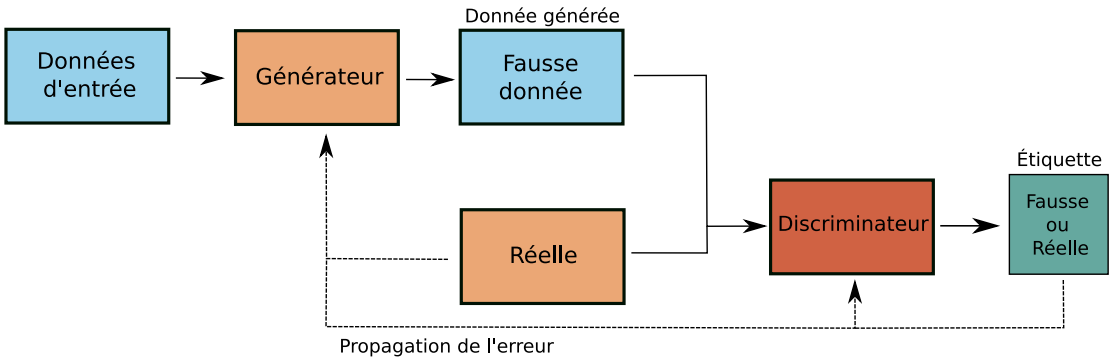


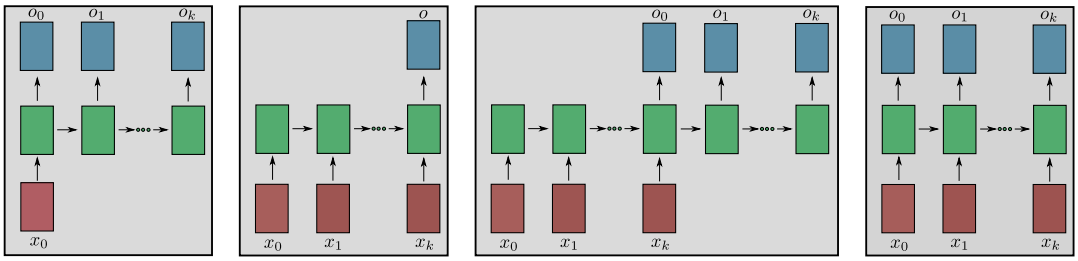
FIGURE 3.8 – Illustration du *Generative Adversarial Network* où deux réseaux s'affrontent. Le générateur va, à partir d'informations d'entrée (aléatoire ou bruitée), se rapprocher au mieux de la référence. Le second, le discriminateur doit différencier l'image générée de l'image réelle. Le discriminateur se met à jour relativement à son erreur de différenciation des deux images. Le générateur se met à jour en fonction de l'image de référence attendue, mais aussi de sa capacité à tromper le discriminateur.

Les deux dernières architectures présentées ont été utilisées pour débruiter des images dans le cadre de rendu d'images de synthèse. Ces méthodes appliqués au rendu d'images seront détaillées dans la suite du chapitre.

3.1.6.6 Réseau de neurones récurrents

RNN pour *Recurent Neural Network* est un type de réseau de neurones qui vise à traiter des données d'entrée issues de séquences d'informations. Les données d'entrée, notées ici X, sont composées de k vecteurs x, où un lien temporel entre les vecteurs x_{k-1} et x_k est présent. L'objectif de ce type de réseau est d'extraire les liens entre ces différentes données temporelles. Pour cela, un RNN propose de traiter les données séparément. Par exemple, il va dans un premier temps traiter x_i, puis demander en entrée x_{i+1} tout en ayant déjà extrait des informations sur x_i. Il cherche ainsi à comprendre les liens entre ces deux vecteurs, d'où le terme récurrent. Les réseaux de neurones

récurrents sont par exemple utilisés pour du traitement du langage, tels que des tâches de traductions, de classification de textes, de prédictions de séquences de mots, etc. Nous présentons ici ce type de réseau de neurones, car nous verrons par la suite qu'il est de plus en plus exploité. Des extensions de tels réseaux ont été proposées, offrant des nœuds du réseau plus sophistiqués nommés *Long Short-Term Memory* (HOCHREITER et Jürgen SCHMIDHUBER 1997a)(LSTM). La spécificité d'une telle cellule est de pouvoir oublier quelques informations précédentes qu'elle ne juge pas utiles à la tâche demandée, amenant une meilleure robustesse des résultats. La figure 3.9 propose un aperçu des différentes structures possibles d'un tel réseau et la figure 3.10 illustre le fonctionnement interne d'une cellule LSTM.



(a) Prédictions de mots (b) Classification (c) Prédictions de mots (d) Traduction de texte

FIGURE 3.9 – Pour chaque représentation possible de réseau RNN, la couche d'entrée est représentée en rouge, celle de la couche récurrente en vert et la sortie en bleu. (a) est une utilisation peu classique de RNN mais existe. On cherche à prédire une phrase à partir d'un mot dans le cadre de traitement de texte. (b) permet d'exploiter un RNN pour de la classification où une seule sortie est attendue après la séquence complète connue par le modèle. (c) peut être utilisé par exemple pour de la prédiction du mot suivant à partir d'une phrase en entrée. Enfin, (d) peut représenter un cas de traduction de texte où la logique de séquence des mots est apprise par le modèle.

3.1.6.7 Fonction d'activation

La fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Le terme de *fonction d'activation* vient de l'équivalent biologique *potentiel d'activation* d'un neurone, seuil de stimulation qui, une fois atteint entraîne une réponse du neurone. La fonction d'activation est souvent une fonction non linéaire. Un exemple de fonction d'activation est la fonction de Heaviside, celle utilisée par le perceptron et décrite par l'équation 3.14, qui renvoie 1 si le signal en entrée est positif, ou 0 s'il est négatif. La figure 3.11 propose un aperçu de trois fonctions d'activations connues : Sigmoïde, Tangente hyperbolique (tanH) et *Rectified Linear Unit* (reLU).

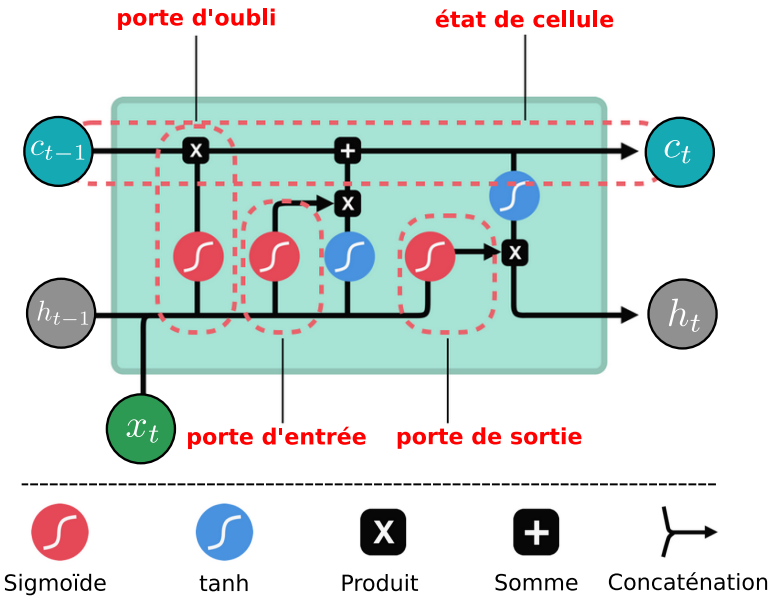


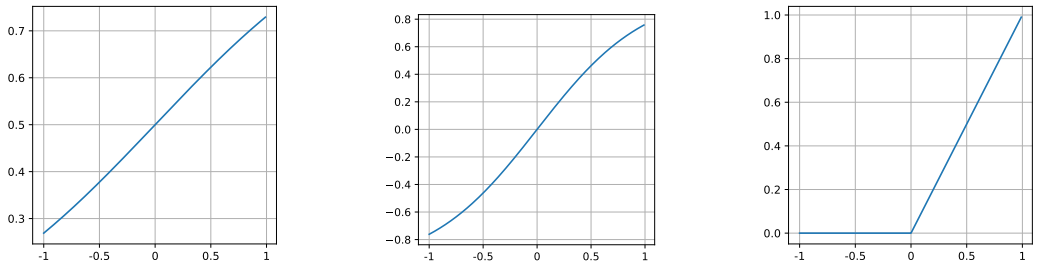
FIGURE 3.10 – Illustration du fonctionnement d'une cellule LSTM où à la fois l'état courant c_{t-1} et l'état caché h_{t-1} de la cellule précédente sont fournis en complément de x_t, les données d'entrée à t. Des calculs internes sont appliqués sur ces données à partir d'opérateurs et de fonctions telles que la *Sigmoïde* et la *Tangente Hyperbolique*. Ainsi, les sorties c_t et h_t actuelles seront exploitées par la cellule LSTM suivante à $t+1$ permettant à la fois de posséder les informations traitées de l'élément x_t en plus des informations de x_{t+1} de la séquence x.

Une autre fonction d'activation connue est la *Softmax* (SHARMA et al. 2017) représentée par l'équation 3.15. Elle permet de donner une probabilité de détection d'une classe dans le cadre d'une classification multi-classe. Ainsi, chaque classe possède une probabilité plus ou moins forte d'être détectée en sortie du réseau de neurones.

$$\text{Softmax}(x,i) = \frac{e^{x_i}}{\sum_k e^{x_k}} \tag{3.15}$$

Chaque fonction possède une spécificité et une préférence d'utilisation (SHARMA et al. 2017), en voici une liste non exhaustive :

— **sigmoïde (logistic)** : utilisée en couche de sortie pour de la classification binaire ;
— **softmax** : utilisée pour de la multi-classification en couche de sortie ;
— **tanH** : utilisée pour des données continues, notamment pour les LSTM ;
— **reLU** : utilisée pour les CNN et les réseaux de multi-couches.



(a) Sigmoïde$(x) = \frac{1}{1+e^{-x}}$ (b) tanH$(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$ (c) reLU$(x) = \max(0, x)$

FIGURE 3.11 – Représentation de fonctions d'activation connues : (a) Sigmoïde, (b) Tangente hyperbolique (tanH) et (c) *Rectified Linear Unit* (reLU).

3.2 Méthodes de reconstruction avec apprentissage automatique pour le rendu

À présent que les fondements de l'apprentissage automatique ont été présentés, nous allons mettre en avant les travaux présentant une utilisation de tels modèles pour le rendu d'images de synthèse. Nous présentons dans cette sous-section les approches de réduction de bruits assistées par des modèles d'apprentissage automatique.

3.2.1 Réseaux multicouches pour la prédiction des pixels

Une des premières approches bénéficiant du support de l'apprentissage automatique est celle de (KALANTARI et al. 2015). Ils mettent en avant la difficulté des méthodes précédentes exploitant les filtres *Non-Local means* et bilatéral (T.-M. LI et al. 2012 ; ROUSSELLE, KNAUS et al. 2012 ; ROUSSELLE, MANZI et al. 2013) à trouver les bons paramètres de filtrage à partir des données d'entrées. Pour rappel, ce genre de méthodes ne s'appuie pas uniquement sur les couleurs des pixels, mais également sur des informations géométriques de la scène. Les auteurs proposent d'exploiter un réseau de neurones multicouches pour estimer la valeur filtrée d'un pixel de sortie de débruitage plutôt que de chercher à trouver les bons paramètres de filtrage suivant une erreur basée sur SURE ou EQM. À partir de caractéristiques primaires moyennes locales comme la couleur, la position du pixel, et des caractéristiques supplémentaires telles que les positions spatiales dans l'image et les normales d'ombrage, ils proposent un processus d'extraction d'informations de 36 caractéristiques qui formeront un vecteur d'entrée x au réseau de neurones multicouches. On parle ici d'informations non locales, car les 36 caractéristiques ne sont pas extraites du pixel uniquement, mais également de ses voisins. Ils distinguent ici des primaires et secondaires. Les caractéristiques primaires sont celles issues du moteur de rendu directement comme la luminance, la position,

la normale d'ombrage, la texture à la première et à la deuxième intersection. Parmi les caractéristiques secondaires on retrouve des statistiques extraites du pixel et de ses voisins au cours du rendu de l'image, comme la moyenne, l'écart type, l'écart moyen (pixel courant par rapport aux pixels voisins), etc. Ainsi, avec ces données récoltées pour un grand nombre de scènes, ils entraînent leur modèle en amont (de manière hors-ligne). Pour cela, pour chaque scène, une image de référence est calculée ; le modèle doit donc apprendre à partir de 36 valeurs d'attributs extraites d'un pixel afin de prédire le pixel correspondant à l'image de référence. Ensuite, le modèle une fois appris peut-être utilisé dans un contexte de débruitage lors du rendu de l'image (dit en ligne). Leurs résultats sont impressionnants, puisqu'ils arrivent à conserver une majorité des effets d'éclairage. De plus, l'appel au modèle de débruitage est réalisé après un faible nombre d'échantillons (8 voire 16) et reste peu coûteux pour débruiter l'image. Ils sont d'ailleurs plus rapides que toutes les autres approches de débruitage de l'image, bien qu'ils travaillent sur le pixel seul (impliquant un grand nombre d'appels au modèle).

3.2.2 Réseaux convolutifs pour le débruitage

Par la suite, (BAKO et al. 2017) exploitent un modèle CNN où ils séparent les images diffuses et spéculaires obtenues par le moteur de rendu. En effet, ils considèrent que les diverses composantes de l'image ont des caractéristiques de bruit et une structure spatiale différentes, ce qui entraîne souvent des contraintes de débruitage contradictoires. Ils visent à atténuer ce problème en décomposant l'image en composantes diffuses et spéculaires. Ils commencent par prétraiter les données diffuses et spéculaires provenant du système de rendu de manière indépendante. Le prétraitement implique la normalisation des données et l'extraction de gradients. Ensuite, ils transmettent ces informations à deux réseaux CNN distincts qui débruitent respectivement l'éclairage diffus et spéculaire. Chaque modèle CNN permet d'obtenir en sortie d'image de même taille que celle de l'entrée. En effet, l'architecture CNN proposée permet de conserver la taille de l'image et ne cherche en aucun cas à la réduire, mais plutôt à lui appliquer des transformations. Chaque modèle CNN est composé de 8 couches cachées de 100 noyaux de 5×5 pour chaque couche. L'objectif des réseaux CNN est d'apprendre à débruiter au mieux l'image d'entrée. La sortie de chaque réseau subit ensuite une reconstruction et un post-traitement avant d'être combinée pour obtenir l'image finale débruitée. Les auteurs utilisent des tuiles d'images de tailles 65×65 comme données d'entrée aux deux modèles. Pour chaque réseau, la fonction de perte, et donc l'erreur d'estimation, est obtenue par la différence de la prédiction (tuile de l'image reconstruite) par rapport à l'image de référence (tuile de l'image attendue). Une comparaison avec la méthode précédente de (KALANTARI et al. 2015) est disponible dans la figure 3.12.

(WONG et al. 2019) fournissent en entrée au modèle des données additionnelles de géométrie de scène en plus des couleurs de l'image. (KUZNETSOV et al. 2018) quant

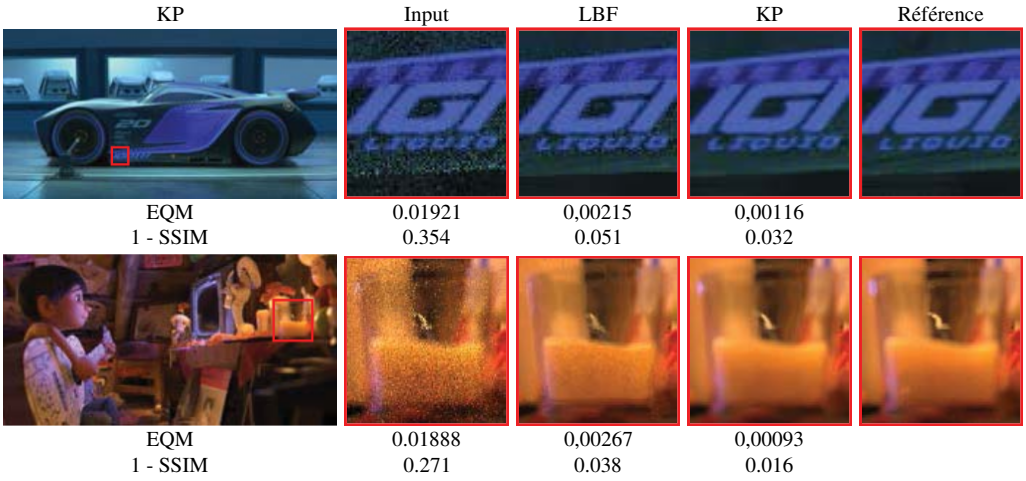


FIGURE 3.12 – Comparaisons des approches (LBF) (KALANTARI et al. 2015) et (BAKO et al. 2017) (KP, pour *Kernel Prediction*). Les erreurs EQM et 1 – SSIM sont calculées pour les deux scènes proposées dans le papier de (BAKO et al. 2017). Pour les deux approches LBF et KP, l'image obtenue est calculée à partir de l'image d'entrée. Les images de références sont calculées avec $4,000$ échantillons (BAKO et al. 2017).

à eux proposent de débruiter une image à partir de deux réseaux CNN. Tout d'abord, l'image est échantillonnée avec un échantillon par pixel. Le premier réseau CNN va prédire une carte d'échantillons optimale. À partir de celle-ci, il est demandé au moteur de rendu d'échantillonner relativement à la carte d'échantillonnage et à un budget alloué. On possède alors en moyenne quatre échantillons par pixel (plus ou moins en fonction de l'échantillonnage adaptatif). Ensuite, un second réseau CNN va débruiter la nouvelle image à laquelle quelques échantillons ont été rajoutés. Le premier réseau CNN pour l'estimation de la carte d'échantillonnage prend l'image bruitée à un échantillon par pixels ainsi que plusieurs caractéristiques auxiliaires comme entrée (11 canaux) pour produire une sortie à un seul canal. Plus précisément, l'entrée de ce CNN est constituée de l'image bruitée au format RGB (trois canaux), des textures au format RGB (trois canaux), des normales d'ombrage (trois canaux), de la profondeur (un canal) et de la visibilité de l'éclairage direct (un canal). Le second CNN, dont la tâche est de debruiter l'image, prend l'image rendue bruitée avec une moyenne de quatre échantillons par pixel (trois canaux) ainsi que des caractéristiques auxiliaires (huit canaux), et produit l'image de sortie (trois canaux). Ils utilisent des tuiles d'image de taille assez conséquente de 512×512 pixels. Les temps de calculs ne sont pas indiqués, mais doivent être relativement faibles en ayant au maximum échantillonné quatre échantillons par pixel via le moteur de rendu. Ils précisent par contre que la carte d'échantillonnage générée, ne permet pas d'interpréter des structures très fines, comme certaines branches d'arbres de certaines scènes.

Plus récemment, (YANG et al. 2019) ont implémenté un réseau de neurones à double

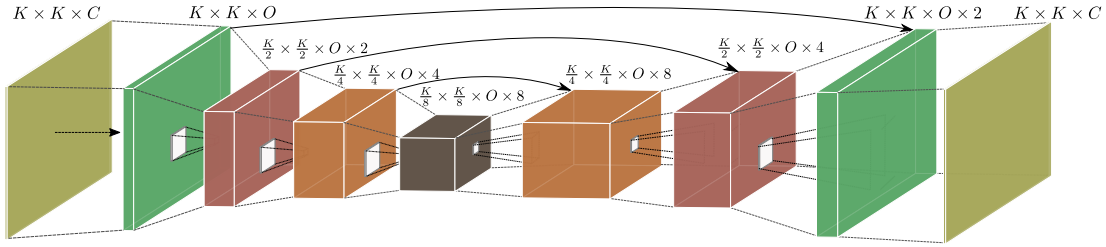


FIGURE 3.13 – *AutoEncoder* en *U*, où chaque information obtenue en sortie d'une couche intermédiaire est transmise à la couche de même taille du décodeur.

encodeur, utilisé pour le débruitage des rendus de Monte-Carlo. Les cartes de caractéristiques et l'image bruitée sont générés à partir du moteur de rendu. Les cartes de caractéristiques sont d'abord fusionnées par un sous-réseau pour obtenir une unique carte de détails de ces informations. Ensuite, la carte de détails obtenue et l'image bruitée sont codées respectivement par un encodeur de caractéristiques et un encodeur d'image HDR (pour *High Dynamic Range*), pour les images à grande gamme de luminance. Enfin, la représentation latente est décodée pour reconstruire une image finale propre. À noter que des connexions intermédiaires permettent au fur et à mesure du processus d'encodage aux deux réseaux de transmettre des informations au réseau décodeur, comme proposé dans les réseaux de neurones *AutoEncoder* dit en *U*, où des informations encore non totalement encodées dans la phase d'encodage, sont transmises au décodeur (RONNEBERGER et al. 2015) (voir figure 3.13). GHARBI et al. 2019 exploitent également les réseaux de neurones *AutoEncoder* en *U*, mais au niveau des échantillons même et des chemins générés. Le réseau traite des cartes d'échantillons, soit la luminance plutôt que des moyennes d'échantillons (valeur RGB des échantillons des pixels).

(CHAITANYA et al. 2017) utilisent un *AutoEncoder* récurrent pour le débruitage de l'image courante dans une séquence d'images. Ils notent que les autoencodeurs orientés débruitage, pour lesquels on donne une image d'entrée bruitée et on attend une image de meilleure qualité, apprennent à supprimer correctement le bruit résiduel présent en entrées (VINCENT et al. 2008). Les données d'entrée ne sont pas limitées aux couleurs des pixels à débruiter mais comportent aussi des informations géométriques de la scène comme la carte de normale, la carte de profondeur, etc. C'est afin de conserver les caractéristiques temporelles à plusieurs échelles, qu'ils introduisent des blocs récurrents entièrement convolutifs après chaque étape de l'encodage des données (première partie d'un *AutoEncoder*). Ce qui leur permet de traiter la temporalité de la séquence d'images et de tenir compte des images précédentes pour débruiter l'image courante. La fonction de perte utilisée est l'EMA (voir équation 3.2) comme proposé par (ZHAO et al. 2017), car il semblerait qu'elle soit mieux adaptée pour la reconstruction d'images. Pour l'apprentissage, les images à débruiter sont découpées en tuiles de 128×128 et sont sélectionnées aléatoirement dans l'image. Une rotation de 90°, 180° ou 270° peut être

appliquée à chaque tuile, ceci dans l'objectif d'agrandir la base d'apprentissage et donc la capacité d'entraîner le modèle. Les résultats obtenus sont également très bons même si quelques détails sont parfois perdus ou lissés en sortie du modèle.

Dans l'idée de traiter également le débruitage d'une image avec une séquence, (VOGELS et al. 2018) utilisent une fenêtre temporelle voisine de taille $2M + 1$:

$$\mathcal{T} = \left[\left\{\mathbf{c}^{i-M}, \mathbf{f}^{i-M}\right\}, \cdots \left\{\mathbf{c}^{i}, \mathbf{f}^{i}\right\}, \cdots \left\{\mathbf{c}^{i+M}, \mathbf{f}^{i+M}\right\}\right]$$

Le tuple d'entrée est obtenu à partir d'un moteur de rendu avec $\mathbf{c}^i$ les couleurs RGB de l'image et $\mathbf{f}^i$, les données auxiliaires facultatives, par exemple la variance de la couleur, la normale à la surface ou encore l'albédo, sur plusieurs échantillons du pixel p. Ils n'exploitent pas un réseau de neurones récurrents, mais une autre manière de procéder pour du débruitage dans un contexte de contenu animé. Premièrement ils extraient des caractéristiques spatiales pour chacune des images d'entrée. Afin d'aligner, les caractéristiques du contenu animé, ils projettent chaque pixel p de l'image centrale à toutes les images d'entrée dans $\mathcal{T}$ en utilisant des vecteurs de mouvement et rassemblent les caractéristiques spatiales des images d'entrée le long de la trajectoire du mouvement à la position p. Ils obtiennent les vecteurs de mouvement à partir du moteur de rendu ou utilisent le flux optique. Les caractéristiques rassemblées pour chaque pixel sont ensuite concaténées et introduites dans un extracteur de caractéristiques temporelles, qui consiste en trois blocs CNN résiduels comme proposés par (HE et al. 2015). Un bloc résiduel permet d'extraire des informations d'entrée, mais les données d'entrée et de sortie sont de nouveau combinées pour former une sortie finale de ce bloc. Pour chaque image, un extracteur produit des caractéristiques qui sont ensuite introduites dans un prédicteur à noyau à une couche. Pour chaque pixel, les multiples noyaux sont normalisés conjointement à l'aide de la fonction d'activation *softmax* (SHARMA et al. 2017). La normalisation conjointe garantit d'obtenir l'image débruitée finale en ajoutant simplement les entrées bruitées pondérées par le noyau. Les résultats montrent qu'avec seulement 16 échantillons par pixel, l'image obtenue après débruitage est de très bonne qualité en comparaison avec une image de référence.

3.2.3 Utilisation des réseaux GAN

L'utilisation de réseaux GAN a été explorée par (XU et al. 2019) pour le débruitage d'images de synthèse. Les auteurs proposent d'utiliser deux réseaux GAN. Contrairement à une approche classique, le réseau G_i ne va pas chercher à générer une donnée, mais à la débruiter respectivement à ses données d'entrée. Un premier réseau pour traiter le débruitage de contenu spéculaire, le second pour le contenu diffus. Le premier réseau prendra donc en entrée l'image bruitée spéculaire et le second l'image bruitée diffuse. En plus de l'image bruitée respective, chaque réseau va prendre en entrée les informations

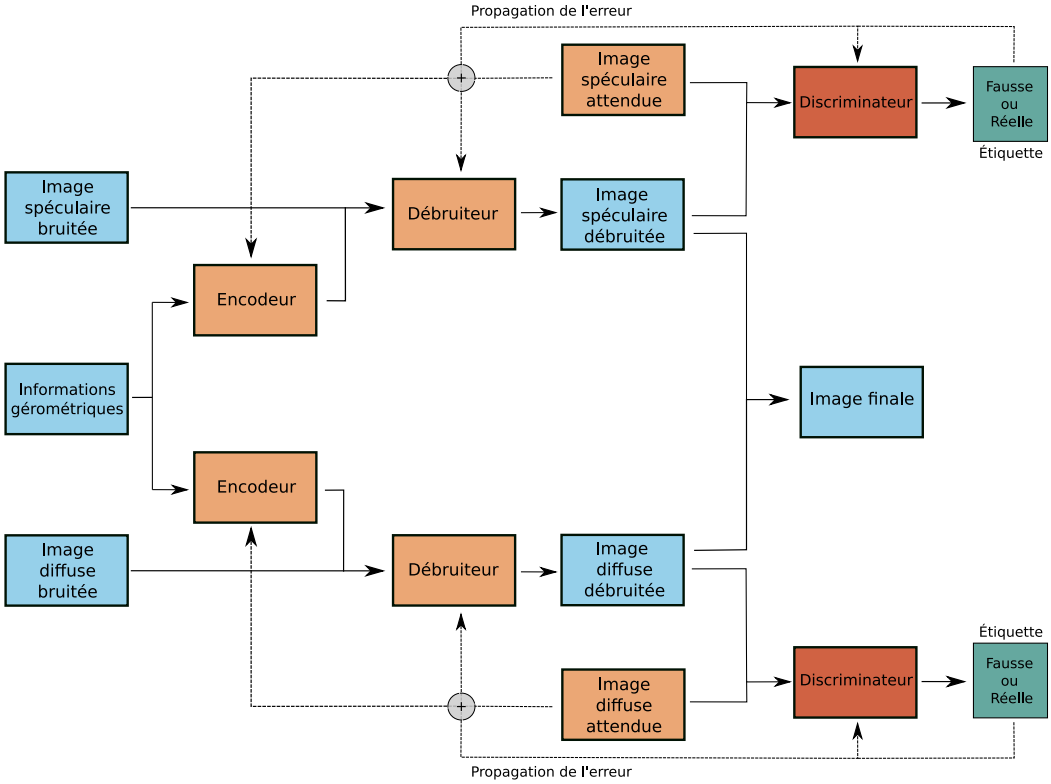


FIGURE 3.14 – Représentation de l'approche proposée par (XU et al. 2019). Le débruitage de l'image est réalisée en deux parties, le débruitage de l'image spéculaire et le débruitage de l'image diffuse. Un discriminateur est ensuite proposé pour chacun des effets lumineux attendus. Chaque discriminateur guide le débruiteur et l'encodeur de données respectivement à son effet lumineux.

géométriques de la scène, telles que la carte de normale, la carte de profondeur et des textures qui seront elles-mêmes déjà filtrées par deux réseaux d'encodage, soit un pour chaque type de contenu (spéculaire/diffus). Ainsi, chaque réseau de débruitage, à partir de leurs données respectives, doit chercher à débruiter l'image attendue suivant son effet (spéculaire/diffus). La combinaison des sorties des images débruitées spéculaire et diffuse est comparée à la référence attendue. Classiquement, les poids de réseaux de débruitage et d'encodage des données additionnelles (données géométriques) seraient tous deux mis à jour relativement à l'erreur obtenue par rapport à la référence attendue. C'est ici qu'interviennent deux discriminateurs pour chacun des effets de lumière. Chaque discriminateur D_i, va essayer de différencier une image générée d'une image attendue relativement à son effet lumineux. La sortie d'un modèle D_i est donc binaire. Il est ensuite possible de calculer une erreur des prédictions de chaque modèle D_i. Ainsi, ils combinent la perte d'erreur EMA de la sortie du débruiteur et la perte du discriminateur

obtenue par le réseau D_i (spéculaire ou diffus). Chaque réseau G_i propage l'erreur et met à jour ses poids relativement à cette nouvelle erreur calculée qui lui est spécifique. La figure 3.14 propose de synthétiser l'idée proposée de l'utilisation du GAN. Cette approche est vraiment très complète et met en avant l'utilisation de nouvelles techniques d'apprentissage automatique dans le rendu d'images de synthèse. Une approche très similaire est celle de (LU et al. 2020) qui, eux aussi, exploitent un GAN pour le débruitage de l'image. Dans le même genre d'idée, mais sans l'utilisation de générateur, (Weiheng LIN et al. 2020) ont entraîné deux débruiteurs séparément pour les effets lumineux spéculaires et diffus. Les résultats obtenus sont meilleurs par comparaison à d'autres méthodes (BAKO et al. 2017 ; YANG et al. 2019). Toutefois, ils exploitent une image disposant d'un plus grand nombre d'échantillons en entrée du modèle (128 échantillons par pixel), au contraire des autres approches qui n'utilisent que très peu d'échantillons.

Le lecteur peut se référer à une étude très récente de (HUO et al. 2021) qui met en avant les derniers travaux d'apprentissage automatique pour l'amélioration de la qualité de rendu d'images de synthèse.

Résumé

Ce chapitre résume les derniers travaux proposés au sein de la littérature concernant l'amélioration du rendu final d'images de synthèse avec utilisation de méthode d'apprentissage automatique. Ces méthodes peuvent être appliquées comme processus post-traitement pour reconstitution d'une image, mais aussi, pour guider le rendu avec un échantillonnage adaptatif. Bien que ces méthodes amènent un biais de calculs certains de l'équation de rendu, leurs résultats sont très impressionnants et permettent de calculer une image de synthèse de très bonne qualité en un temps relativement faible (de moins d'une minute à quelques minutes au lieu de quelques heures). Dans le cadre de la thèse, nous nous sommes également appuyés sur des approches récentes d'apprentissage automatique pour répondre à la problématique de la détection de bruit perceptible. Dans le prochain chapitre, nous nous intéresserons à la notion de perception de manière globale, puis plus précisément à la perception des images de rendu et notamment au bruit perceptuel généré lors du processus de rendu.

Chapitre 4

Problématique de la perception du bruit et mesures de qualité

Introduction

Après avoir présenté les travaux relatifs à l'amélioration du rendu d'une image de synthèse, nous nous intéressons ici à la perception visuelle humaine sur ce genre spécifique d'image. En effet, une image générée par méthode de MC permet de retranscrire au mieux la réalité en proposant une simulation physique de l'éclairage. Toutefois, il est important de prendre en considération l'aspect du bruit généré lors du rendu de telles images. Ce bruit peut impacter considérablement la perception que l'humain a de cette image. Dans ce dernier chapitre d'état de l'art, nous allons présenter les travaux relevant de l'étude des propriétés de la perception humaine, tout comme ceux orientés sur l'application de méthodes permettant de quantifier le bruit présent dans une image photo-réaliste. Nous verrons qu'un grand nombre d'études ont été menées pour des images dites naturelles avant de présenter celles relatives aux images générées par ordinateur. Ce chapitre permettra d'introduire toute la problématique de la thèse en mettant en avant son attendu.

Nous présentons d'abord dans ce chapitre les principales propriétés du système visuel humain HVS, notamment liées à la perception des images. Par la suite, des modèles de discrimination visuelle et de différences entre deux images naturelles, basés sur ces propriétés du HVS seront abordés. Des propositions d'utilisation de ces derniers quant à la détection du bruit dans les images de synthèse seront également présentés. Enfin, des modèles plus récents basés sur de l'apprentissage automatique seront introduits. Ce type de modèle prend en entrée des caractéristiques extraites d'une image de synthèse permettant de la catégoriser comme encore bruitée ou non.

La perception est liée aux mécanismes de cognition permettant de retranscrire l'environnement extérieur et de le comprendre. Concernant une image, la perception de celle-ci

peut renvoyer des sentiments et des émotions, on note aussi que des zones de l'image dites saillantes sont davantage remarquées. Cette perception accentuée sur des parties de l'image est soit relative aux propriétés de l'image ou à la perception qu'a cette personne de l'image. De manière globale, de grands mouvements théoriques ont été introduits concernant la perception dans son sens large comme présenté dans l'ouvrage d'André Delorme (DELORME et al. 2014). On peut y retrouver trois grandes questions où chacune des réponses à ces questions n'est autre qu'un compromis :

Nativisme et empirisme où la question est de savoir si la perception est innée ou acquise. C'est l'une des plus anciennes et des plus importantes provoquant d'ailleurs de grands débats. Cette question met en conflit à la fois l'expérience et inhérence de la perception des êtres humains.

Élémentarisme et globalisme où l'on cherche à comprendre si la perception est le produit de composantes plus élémentaires ou si elle se fonde plutôt sur une perception globale et directe de l'environnement ou de l'objet perçu.

Fonctionnalisme et formalisme met en avant la question de savoir si la perception d'une personne peut être dépendante ou non de ses caractéristiques personnelles. Qu'est-ce qui détermine nos perceptions ? Uniquement le stimulus proposé par l'objet perçu ou les caractéristiques anatomo-physiologiques des systèmes sensoriels ?

Comme déjà annoncé précédemment, chacune de ces questions amène à un compromis, ce qui rend la perception de chaque individu unique et difficile d'interprétation et de quantification. Avant de nous interroger davantage sur la manière dont une image de synthèse peut-être perçue, nous allons présenter les propriétés d'un des sens le plus important de la perception, celui de la vue.

4.1 Propriétés de la perception visuelle humaine

On désigne sous le nom de perception visuelle la modalité sensorielle qui permet d'être sensible à la lumière visible, plus précisément sur la partie des rayonnements électromagnétiques comprise entre 380 et 780 nm (pour l'observateur de référence défini par la CIE). La perception visuelle est un processus complexe qui implique de nombreuses interactions non seulement de l'œil humain, mais aussi du système nerveux. La compréhension de ces mécanismes est encore actuellement l'objet de nombreuses recherches, tout comme son fonctionnement d'ensemble qui soulève encore un grand nombre de questions. Les données quantitatives sont néanmoins suffisantes pour établir certains modèles locaux du fonctionnement du système perceptif humain.

4.1.1 Le Système Visuel Humain

Même si la connaissance actuelle du système visuel humain (HVS pour *Human Visual System*) reste incomplète, son étude remontant à l'antiquité permet aujourd'hui d'en comprendre partiellement le fonctionnement, notamment sur les aspects de ses performances, mais aussi ses limites. La détection et le traitement des informations visuelles sont soumis à de nombreuses étapes depuis l'arrivée des rayons lumineux dans l'œil jusqu'aux traitements et au stockage des informations au niveau du cortex. Il est possible de décomposer ces étapes en trois niveaux impliquant différentes disciplines :

— **le trajet des rayons lumineux dans l'œil** qui concerne essentiellement ses propriétés optiques et physiologiques. Les rayons lumineux arrivent sur la cornée passent ensuite par la pupille dont le diamètre d'ouverture s'adapte automatiquement à l'intensité lumineuse perçue ;

— **la transformation de l'énergie lumineuse en signaux nerveux** fait l'objet d'études bio-chimiques et cellulaires. Les récepteurs de l'œil servent à décomposer les informations lumineuses en signaux électriques qui seront ensuite envoyés au nerf optique. Chez l'être humain, il existe trois types de cônes (rouge, vert, bleu) servant à décomposer la lumière en couleurs et les bâtonnets, mais sans distinction de couleur, plus sensibles que les cônes dans les lieux à faible présence de lumière ;

— **l'interprétation de l'information visuelle** dans le système nerveux est le domaine de la physiologie et de la neurologie. L'organisation des cellules réceptrices, des voies visuelles et des différentes parties du cortex sont à l'origine des différentes étapes du traitement de l'information visuelle jusqu'à la perception du monde extérieur.

Un grand nombre de recherches sur le comportement du HVS ont été réalisées et ont permis la modélisation de certains de ses aspects, tels que l'adaptation visuelle à la lumière, la sensibilité aux contrastes en fonction des fréquences spatiales (CSF), l'acuité visuelle, le masquage visuel, la vision des couleurs, la sensibilité aux différentes orientations, ou encore l'attention visuelle. Les études menées ont permis l'observation non seulement des performances du HVS, mais aussi ses limitations.

4.1.2 Adaptation visuelle

La luminance est une grandeur correspondant à la sensation visuelle de luminosité d'une surface. Comme précisé dans le Chapitre 1, la luminance est la grandeur photométrique utilisée pour la synthèse d'image comme étant la puissance de la lumière visible passant, ou étant émise, par un élément de surface dans une direction donnée, par unité de surface et par unité d'angle solide.

4.1.2.1 Gamme de luminance

À la surface de la terre, l'intensité lumineuse varie énormément notamment au cours de la journée, ou encore entre une journée ensoleillée et une nuit sans lune. L'œil a la capacité de s'adapter constamment pour capter les photons dans l'obscurité, mais aussi de se protéger lorsque l'intensité lumineuse est relativement élevée jusqu'à une certaine limite.

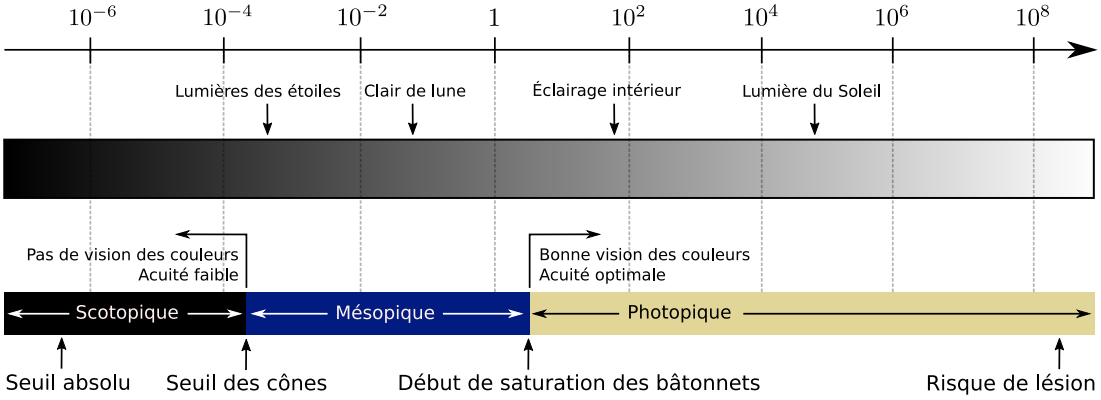


FIGURE 4.1 – Luminance d'un papier blanc selon différents éclairements ($cd \cdot m^{-2}$) (DELORME et al. 2014)

Comme le montre la figure 4.1, notre vision couvre un grand intervalle d'éclairage, dit gamme de luminance, de la plus faible quantité de lumière nécessaire pour la visibilité mesurée à 10^{-6} $cd \cdot m^{-2}$ jusqu'à un seuil de 10^{6} $cd \cdot m^{-2}$. Plus précisément, le seuil absolu d'éclairage rétinien en lumière blanche est connu depuis longtemps. La mesure des seuils supérieurs pose de nombreux problèmes méthodologiques et éthiques. On estime néanmoins que le seuil de douleur est de 10^{4} $cd \cdot m^{-2}$ (DENTON et al. 1954). Au-delà de cette valeur, l'œil risque de subir des altérations telles que des brûlures au niveau des cellules rétiniennes. Pour assurer la vision, le système visuel adapte ses fonctionnalités aux changements de condition de luminosité, plus ou moins rapidement en fonction de l'importance des variations. Ces mécanismes d'adaptation permettent au système visuel de maintenir une sensibilité constante pour des niveaux d'illumination différents. La quantité de lumière qui passe dans l'œil est ajustée par la pupille avant d'atteindre les cellules photo-réceptrices de la rétine : les *cônes* et les *bâtonnets* qui permettent l'envoi de signaux électriques au nerf optique. À ce niveau, de nombreux mécanismes biochimiques assurent également l'adaptation.

4.1.2.2 Les cônes

La couche rétinienne contient aussi environ 7 millions de cônes principalement localisés dans la zone appelée *fovéa*. Il faut savoir que ces cônes réagissent quatre fois

plus vite que les bâtonnets, mais restent moins sensibles que les bâtonnets et nécessitent plus de lumière pour être stimulés. Ces photo-récepteurs servent à la vision dite « diurne » et à la différenciation des couleurs. (YOUNG 1802) a émis l'hypothèse de l'aspect trichromatique de la vision. Cette idée se base sur le fait que notre système de perception de la couleur est basé sur trois types distincts de cellules. Ces cellules sont composées d'un pigment qui présente un maximum de sensibilité pour des ondes lumineuses de longueurs différentes :

— les cônes L (long) sont davantage sensibles au rouge ($580nm$) ;
— les cônes M (medium) sont davantage sensibles au vert ($545nm$) ;
— les cônes S (short) sont davantage sensibles au bleu ($440nm$).

4.1.2.3 Les bâtonnets

Chaque œil est composé d'environ 120 millions de bâtonnets répartis sur la majeure partie de la rétine. Les bâtonnets sont environ 100 fois plus sensibles à la lumière que les cônes, ce qui leur permet d'assurer la vision nocturne, lorsqu'il n'y a pas assez de lumière pour activer les cônes. Les bâtonnets ne réagissent pas aux fortes intensités lumineuses. Leur sensibilité est liée à la *rhodopsine*, pigment visuel qui a la propriété de blanchir avec l'intensité lumineuse. Ils ne permettent donc pas de distinguer comme le font les cônes, les couleurs, mais plutôt les nuances de faibles luminosités.

4.1.2.4 Temps d'adaptation

De manière globale, les cônes et les bâtonnets, comme le montre la figure 4.1, fonctionnent selon les trois modes suivants :

— **photopique** (le jour) où seuls les cônes sont actifs ;
— **scotopique** (la nuit) où les bâtonnets fonctionnent pour capter les faibles intensités ;
— **mésopique** où les deux types de photo-récepteurs réagissent simultanément en fonction de l'environnement.

Toutefois, l'adaptation d'une gamme de luminance à une autre n'est pas un processus instantané (FERWERDA 2001 ; MICHAELS 1956) et ce changement d'un mode de fonctionnement à un autre nécessite du temps. Notamment lors d'un passage brusque d'un environnement fort lumineux à l'obscurité.

4.1.3 Sensibilité CSF

La fonction de sensibilité au contraste en fonction des fréquences spatiales (CSF) est un modèle harmonique simple qui a été utilisé pour déterminer la capacité d'un individu à percevoir les détails. Plus précisément, c'est une mesure de la capacité à discerner des

luminances de niveaux différents dans une image (dans un cas statique), d'où la notion de contraste. Cette sensibilité au contraste varie d'un individu à l'autre et plusieurs facteurs peuvent avoir un impact sur celle-ci. À noter notamment, qu'elle atteint son maximum à l'âge de 20 ans environ pour des valeurs d'environ 2 à 5 cycles par degré (ROSS et al. 1985).

FIGURE 4.2 – Dans cette image, l'amplitude du contraste dépend uniquement de la coordonnée verticale et la fréquence spatiale dépend uniquement de la coordonnée horizontale.

La figure 4.2 illustre un tel contraste avec une amplitude verticale. Cette image peut être caractérisée par ses deux déterminants principaux : le contraste et la fréquence spatiale. Le contraste est simplement une mesure de la différence de luminance entre zones claires et zones sombres, alors que la fréquence spatiale est une mesure de l'espacement entre les différentes zones, définie en cycle du contraste par degré de champ visuel (cycles/degré).

Les expériences menées par Campell et Robson (BLAKEMORE et al. 1969 ; CAMPBELL et al. 1968) proposent à un observateur un visuel d'images composées de plaques rayées ayant plusieurs fréquences spatiales. Une telle image amène à une représentation de bandes verticales (bande noire puis bande blanche répétée) de tailles différentes (voir figure 4.3). Pour chaque image de fréquence différente, il est demandé à l'observateur s'il arrive à distinguer les bandes et donc leur contraste. Le plus petit contraste à partir duquel il peut distinguer les bandes de la plaque montrée, est le seuil de contraste. La fonction CSF est alors l'inverse du plus petit contraste en fonction des fréquences spatiales.

Relativement aux résultats expérimentaux présentés par Campell et Robson (CAMPBELL et al. 1968), la modélisation de la CSF a été pendant des années, un sujet de recherches de plusieurs scientifiques de la vision. Le modèle CSF proposé par (MANNOS et al. 1974) est calculé pour un niveau d'adaptation de 40 $cd \cdot m^{-2}$. Cette CSF a été validée par la suite par Rushmeier (H. RUSHMEIER et al. 1995) pour être applicable dans les modèles

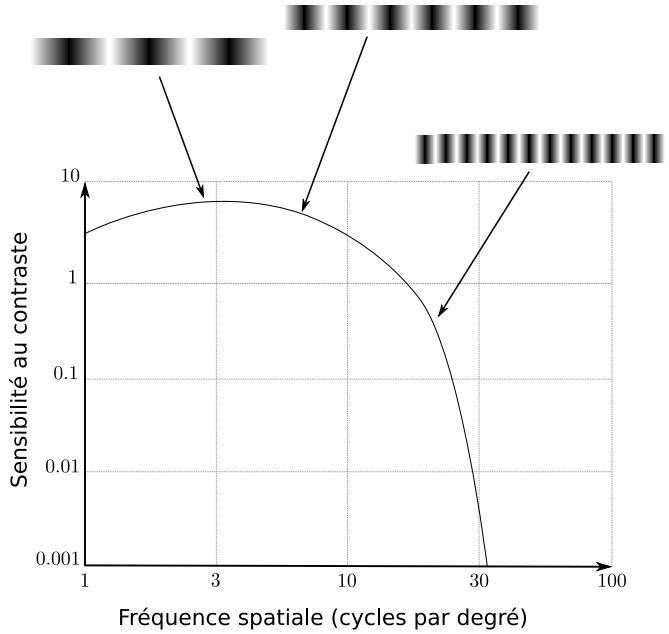


FIGURE 4.3 – Le changement de la sensibilité au contraste en fonction de la fréquence spatiale (CSF) issue des résultats expérimentaux de Campell et Robson (CAMPBELL et al. 1968)

perceptifs de comparaison des images comme le *Visual Difference Predictor* (VDP), présenté par la suite. En 1975, Kelly (KELLY 1975) a développé un modèle mathématique pour décrire les caractéristiques des fréquences spatiales du champ récepteur pour des hauts niveaux d'éclairage. Un modèle de la CSF a été également proposé par Daly (DALY 1992) pour un niveau d'adaptation de 50 $cd \cdot m^{-2}$.

4.1.4 Autres propriétés visuelles

Le contraste n'est pas l'unique propriété du système visuel humain qui a été étudiée ; on peut également y retrouver l'acuité et le masquage. Il en existe d'autres, mais que nous ne détaillons pas dans ce manuscrit.

4.1.4.1 Acuité

L'acuité est la propriété visuelle mesurant la capacité à discerner un petit objet (ou « optotype ») situé le plus loin possible. Ce qui est équivalent à voir à une distance fixe, distance généralement de cinq mètres, un optotype sous le plus petit angle possible exprimant ainsi la faculté de discrimination de la zone. C'est une fonction du contraste et de la fréquence spatiale qui peut être définie comme le plus petit angle visuel permettant

de distinguer deux points noirs séparés sur un fond blanc. Le seuil de l'acuité est approximativement limité à « 30 secondes d'arc de l'angle visuel » dans les conditions standards (THOMAS 1975), ce qui correspond à l'écart entre les cônes dans la rétine (estimé à 3 microns).

4.1.4.2 Masquage

Le masquage visuel est le fait de dissimuler ou de modifier la perception d'une cible visuelle en fonction des propriétés visuelles du contexte environnant telles que les couleurs ou le contenu fréquentiel. Ainsi, par exemple, il est difficile dans une image avec un fond texturé orienté horizontalement de distinguer un objet ayant la même fréquence et la même orientation. Inversement, cet objet est mieux repérable s'il est orienté verticalement. Le masquage est un phénomène très important de la perception humaine.

Nous verrons que l'effet du masquage en image de synthèse peut considérablement réduire la perception du bruit présent dans une image.

4.2 Modèles d'attention et différences

Il est possible de différencier deux types d'études de la perception qu'a l'être humain d'une image. La première concerne la perception relative à une image où l'on va s'intéresser à cibler les zones d'intérêts ; nous verrons que le terme le plus couramment utilisé est celui « zone saillante ». La seconde concerne plutôt le traitement de deux images, où l'on va définir une notion de différence perceptible entre ces deux images.

4.2.1 Carte de saillances

La zone d'acuité maximale correspond à une toute petite fraction du champ visuel (de l'ordre de 3 à 5°). Ce qui veut dire que lorsqu'une image est perçue, seule une faible partie est vue en détail à un instant donné. C'est le système nerveux qui décide quelle partie de l'information disponible doit être sélectionnée pour un traitement ultérieur plus détaillé et quelle partie doit être rejetée : on peut parler ici de zones d'attractions. Les stimuli sélectionnés doivent être classés par ordre de priorité, les plus pertinents étant traités en premier et les moins importants en second, ce qui conduit à un traitement séquentiel des différentes parties de la scène visuelle. Ce processus de sélection et d'ordonnancement est appelé attention sélective. Parmi de nombreuses autres fonctions, l'attention à un stimulus a été considérée comme nécessaire pour qu'il soit perçu consciemment (TSUCHIYA et al. 2008).

4.2.1.1 Définition et origine

Il est difficile de déterminer quels stimuli sont sélectionnés par le processus attentionnel, leurs priorités et ceux écartés. En effet, de nombreux facteurs contribuent à cette décision interne. Ces facteurs peuvent être distingués comme étant ascendants et descendants. Les facteurs ascendants font référence uniquement à l'entrée sensorielle instantanée, sans tenir compte de l'état interne de l'organisme. Au contraire, le contrôle descendant tient compte de cet état interne, comme les objectifs de l'organisme au moment du processus de perception. Cela peut concerner l'histoire personnelle de l'individu, ses expériences, son état émotionnel, etc. Un exemple d'attention ascendante peut concerner l'explosion d'un pétard provoquant une attention soudaine de l'individu, alors qu'un exemple d'attention descendante peut être lié à la recherche d'un objet dans une image (ou encore le fait de compter le nombre de personnages présents dans une image).

Étant donné qu'il n'est pas possible actuellement de mesurer avec précision les états internes d'un organisme, la notion de « cartes de saillances » a été introduite pour pallier ce problème (NIEBUR 2007). Les aspects du contrôle attentionnel, c'est-à-dire l'attention ascendante, sont plus faciles à comprendre que ceux qui sont influencés par les états internes. La tentative la plus influente pour comprendre l'attention ascendante et les mécanismes neuronaux sous-jacents a probablement été faite par Christof Koch et Shimon Ullman en 1985 (C. KOCH et al. 1985). À l'origine de cette idée se trouve la description de l'architecture du système visuel en régions cérébrales sous-tendant des « cartes cognitives » relativement spécialisées dans le traitement de caractéristiques perceptives spécifiques (couleur, orientation, mouvement, etc.). Ainsi, ils proposent que ces différentes caractéristiques visuelles soient combinées en une seule carte topographique orientée, la carte de *saillance*, qui intègre les informations normalisées des cartes de caractéristiques individuelles en une mesure globale de la perceptibilité. La saillance ascendante est déterminée par la différence entre un stimulus et son environnement, dans de nombreuses sous-modalités et à de nombreuses échelles. La saillance à un endroit donné est déterminée principalement par la différence entre cet endroit et son environnement en considérant des critères de couleur, d'orientation, de mouvement, de profondeur, etc.

Koch et Ullman (C. KOCH et al. 1985) ont postulé que l'emplacement le plus saillant (au sens défini ci-dessus) dans une scène visuelle serait un bon candidat pour la sélection attentionnelle. L'étude de Koch et Ullman était purement conceptuelle et la première mise en œuvre réelle d'une telle carte de saillance a été décrite par Niebur et Koch en 1996 (NIEBUR et Christof KOCH 1996). Leur modèle de carte de saillance utilise des indices de couleur, d'intensité, d'orientation et de mouvement, à la fois sur des données visuelles simplifiées et sur des scènes naturelles complexes. Ils ont ainsi proposé un balayage séquentiel de la scène visuelle par ordre de saillance décroissante. Des travaux ultérieurs ont ensuite affiné le modèle (L. ITTI et al. 1998 ; Laurent ITTI et al. 2001).

Le modèle proposé par (L. ITTI et al. 1998 ; Laurent ITTI et al. 2001) prend en compte

le changement de la sensibilité du HVS en fonction de l'intensité visuelle et du contraste.

4.2.1.2 Autres modèles

Un nombre assez important de modèles de saillance existe actuellement, et (KÜMMERER, WALLIS et al. 2018) ont proposé un benchmark pour les comparer qui est disponible en ligne (KÜMMERER, BYLINSKII et al. p. d.). Ils ont également très récemment proposé un état de l'art des méthodes existantes (KÜMMERER et al. 2021). Le modèle *Graph-Based Visual Saliency* (GBVS) est une extension du modèle d'Itti qui prédit les fixations humaines avec une plus grande précision que les algorithmes standards.

L'intégration de tels modèles suscite de plus en plus de travaux dans le contexte de l'informatique graphique, notamment avec l'évolution des périphériques de rendu. Hector Yee a été le premier à utiliser les cartes de saillance pour évaluer la qualité du rendu de l'éclairage global pour l'animation (H. YEE et al. 2001 ; H. Y. YEE et al. 2002). En 2006, Longhust a utilisé l'accélération GPU pour calculer des cartes de saillance en temps réel afin de les intégrer à moteur de rendu (LONGHURST et al. 2006). En 2014, Koulieris a proposé d'utiliser lors du rendu, une attention sélective au niveau de détail (*Lovel Of Details*) (KOULIERIS et al. 2014).

4.2.2 Modèles de différences

La saillance n'a pas été le seul point d'étude sur les stimuli et l'attention visuelle ; de nombreuses autres études ont été menées pour trouver et quantifier les différences perceptibles entre deux images où, généralement une de ces images est de meilleure qualité.

4.2.2.1 Prédicteur de Différence Visuelle

Le *Visual Difference Predictor* (VDP) a à l'origine été développé en 1993 par Daly (DALY 1992). Ce modèle prend en compte les variations de sensibilité du système visuel qui limitent la perception : celles liées à l'intensité lumineuse, au contenu fréquentiel et au contenu structurel de l'image. Tout d'abord, le canal de luminance de chacune des images d'entrée est converti en une représentation non-linéaire. Cette non-linéarité correspond à la réponse rétinienne en fonction de l'intensité lumineuse. L'objectif est de prendre en compte le changement de la sensibilité du HVS entre les zones lumineuses et les zones sombres qui couvrent l'image. La deuxième étape du VDP consiste à estimer la sensibilité fréquentielle. Après le passage dans le domaine fréquentiel en appliquant une transformée de Fourier rapide (FFT), la représentation non-linéaire de l'image est normalisée par la CSF afin d'obtenir une carte de différence des deux images.

4.2.2.2 Modèle de Discrimination Visuelle

Tout comme le VDP, le *Visual Discrimination Model* (VDM) de Sarnoff (LUBIN et al. 1997) produit une carte de différences visibles entre deux images, appelée carte JND (*Just Noticeable Difference*). Mais contrairement au VDP, le VDM opère directement dans l'espace image où le processus commence par une convolution de l'image à partir d'une fonction qui modélise les effets optiques de l'œil sur l'image. Le résultat de ce traitement est par la suite échantillonné selon l'excentricité fovéale afin de prendre en compte la diminution de la résolution spatiale en dehors des régions fovéales. Ensuite, la sensibilité au contraste est obtenue par une décomposition pyramidale de l'image où pour chaque niveau de la pyramide, le mécanisme de détection d'orientation est simulé en appliquant par convolution des filtres Gaussien orientables. Les niveaux des différentes pyramides, dont chacune correspond à un canal de réponse à une orientation, sont sommés et pondérés par la CSF et une fonction de masquage, tout en respectant la non-linéarité de la sensibilité du HVS à la fréquence spatiale ainsi qu'au contraste. Finalement, la carte des différences perceptives (carte JND) est calculée comme la somme des distances sur chaque canal de réponse des deux images d'entrée. Cette carte peut être convertie en mesure de probabilité comme pour le VDP. Le Sarnoff VDM a ensuite modifié cette carte JND en intégrant la sensibilité spatio-temporelle pour le traitement des séquences d'images en couleurs (GROUP et al. 1997).

Une étude de l'utilisation de tels modèles de prédictions de différences a été réalisée pour la première fois par McNamara (MCNAMARA 2001) afin de vérifier leur utilisation dans un contexte d'images de synthèse. Les algorithmes de rendu axés sur la perception sont décrits. Ces algorithmes se concentrent sur l'intégration de modèles du système visuel humain directement dans les calculs d'éclairage global afin d'en améliorer l'efficacité.

4.2.3 Métriques de qualité d'images

Les travaux concernant la prédiction des différences visuelles et perceptibles dans les images, tout comme ceux cherchant à cibler les zones d'attractions, ont conduit vers des travaux visant à obtenir un score de qualité d'une image. On peut distinguer plusieurs métriques de qualité d'images, celles *avec référence*, où l'image attendue est connue et est comparée à une image de moins bonne qualité, celles avec *référence réduite*, où seules quelques informations de l'image de référence sont connues et enfin, celles *sans référence*, où uniquement l'image avec des distorsions est connue. Nous présentons dans le cadre de ce manuscrit une liste non exhaustive de métriques avec et sans référence. Les métriques de qualité d'images fournissent un score dit *objectif* et sont comparées à des scores *subjectifs* moyens obtenus, que l'on appelle couramment *Mean Opinion Score* (MOS).

4.2.3.1 Avec référence

Le *Peak signal-to-noise Ratio* (PSNR) (WELSTEAD 1999) est une métrique permettant d'obtenir un score de qualité d'image. Elle est formulée par le rapport entre la puissance maximale possible d'un signal et la puissance des bruits parasites qui affectent la fidélité de sa représentation et peut donc être appliquée à l'image. Elle est formulée par :

$$PSNR = 10 \cdot \log_{10}\left(\frac{d^2}{EQM}\right)$$

où d est la dynamique du signal (la valeur maximum possible pour un pixel, dans le cas standard d'une image codée sur 8-bits, $d = 255$) et EQM, l'Erreur Quadratique Moyenne (définit par l'équation 3.2) entre l'image de référence et l'image ayant subi des distorsions. Des métriques plus récentes telles que la *Structural Similarity Index Measure* (SSIM) (Z. WANG et al. 2004) tout comme ses extensions multi-échelles (Zhou WANG et al. 2003) ont été proposées pour une meilleure prise en compte du système visuel humain (HVS). La SSIM est un modèle basé sur la perception qui considère la dégradation de l'image comme un changement perçu de l'information structurelle, tout en incorporant également d'importants phénomènes perceptifs, y compris les termes de masquage de luminance et de masquage de contraste. La différence avec d'autres techniques telles que l'EQM ou PSNR est que ces approches estiment les erreurs absolues. L'information structurelle est l'idée que les pixels ont de fortes interdépendances, surtout lorsqu'ils sont spatialement proches. Cette métrique est grandement utilisée pour des comparaisons perceptives de résultats obtenus dans le domaine de l'Informatique Graphique en complément de l'EQM.

La SSIM bien que mieux adaptée au HVS reste encore peu fidèle à la complexité de la perception visuelle que l'on peut avoir d'une image. Le modèle HDR-VDP version 2 (MANTIUK et al. 2011 ; NARWARIA et al. 2015) est une extension du *Visual Discrimination Model* précédemment décrit qui fournit à la fois une carte de probabilité de différence perceptible entre les deux images ainsi qu'un score de la qualité de l'image ayant subi des distorsions par rapport à l'image de référence. Ce score est issu d'un modèle de prédiction ayant appris sur des scores subjectifs, que l'on appelle *Mean Opinion Score*. À noter que ce modèle est adapté aux grandes gammes de luminance, *High Dynamic Range* (HDR). Il est basé sur de nouvelles mesures CSF, effectuées dans des conditions d'observation cohérentes pour une grande variété de luminances de fond et de fréquences spatiales. Il intègre également de nouveaux modèles métriques des sensibilités des cônes L, M, S et des bâtonnets et est sensible aux différentes caractéristiques spectrales de la lumière entrante. La sensibilité à la lumière des photorécepteurs est modélisée séparément pour les cônes et les bâtonnets, bien que les cônes L- et M- partagent la même caractéristique. La métrique utilise une pyramide orientable plutôt qu'une transformée corticale pour

décomposer l'image en bandes sélectives en termes d'espace et d'orientation. Le nouveau modèle de masquage de contraste introduit le masquage inter-bandes et l'effet de l'aplatissement de la CSF. Une formule simple d'intégration spatiale utilisant la sommation de probabilités est utilisée pour tenir compte de l'effet de la taille des stimuli. La figure 4.4 illustre les résultats obtenus à partir du HDR-VDP sur un point de vue de la scène *Bedroom*.

(a) 100 échantillons
SSIM : 0.6315

(b) 10 000 échantillons
SSIM : 1.0

(c) Carte de probabilité
MOS : 53.814

FIGURE 4.4 – Comparaisons de la SSIM et du HDR-VDP sur un point de vue de la scène *Bedroom* où l'image bruitée est calculée avec 100 échantillons et celle de référence de 10 000 échantillons par pixel. La carte de probabilité des différences et le score MOS du modèle HDR-VDP sont extraits, sachant que ce score est compris entre 0 et 100. Le score de 0 signifie une différence totale des deux images et un score de 100 correspond à deux images identiques.

Une métrique à base de décomposition en valeurs singulières (*Singular Value Decomposition*, SVD) a également été proposée (S. WANG et al. 2013). Il s'agit d'une méthode de décomposition matricielle qui peut-être appliquée à une image. La SVD permet d'extraire les valeurs singulières qui correspondent aux racines carrées des valeurs propres de $M^T \times M$. Ces valeurs singulières permettent de reconstruire partiellement les images factorisées, ce qui permet une compression de l'information et de quantifier le bruit afin d'estimer un score de qualité. L'originalité de la méthode vient de l'extraction et la réduction du bruit présent dans l'image à partir de la SVD. La SVD est ici succinctement présentée, car elle a fait l'objet d'études et de contributions dans le cadre de la thèse et sera davantage détaillée dans le chapitre 7.

Ces métriques, bien qu'elles apportent une information intéressante de qualité d'image, ne peuvent néanmoins pas être exploitées directement dans le processus de rendu d'une image de synthèse étant donné que l'image de référence est inconnue.

4.2.3.2 Sans référence

Dans un contexte différent, des métriques essaient de quantifier la distorsion de l'image et de fournir un score de qualité où l'image attendue (de référence) n'est pas

accessible ou n'est pas connue. Il en existe un très grand nombre, comme pour celles avec référence et une liste non exhaustive est également présentée.

La complexité d'évaluation réside ici dans le fait que la référence est inconnue et que le nombre de classes de distorsion peut-être grand. Pour cela, la métrique DIIVINE pour *Distortion Identification-based Image Verity and INtegrity Evaluation* proposée par Moorthy et Bovik en 2011 (A. K. MOORTHY et al. 2011, 2010) est basée sur un cadre en deux étapes impliquant l'identification de la distorsion suivie d'une évaluation de la qualité spécifique à la distorsion. DIIVINE est capable d'évaluer la qualité d'une image déformée pour plusieurs catégories de distorsions, contrairement à la plupart des algorithmes sans référence qui sont spécifiques à une nature de distorsion. La métrique DIIVINE est basée sur les statistiques des scènes naturelles qui régissent le comportement des images naturelles et qui sont couramment utilisées pour d'autres approches sans référence (H. R. SHEIKH et al. 2005). En effet, les statistiques des scènes naturelles expliquent la représentation des catégories de scènes dans le cortex visuel humain (STANSBURY et al. 2013) ce qui a amené un fort attrait pour leur utilisation.

Des approches orientées sur de l'apprentissage ont également émergé. La méthode proposée par (YE et al. 2013) est composée d'un modèle d'extraction de filtres sur l'image qui sont ensuite fournis en entrée à un modèle de régression (modèle SVR) à partir des scores subjectifs humains (MOS). Alors que les approches précédentes traitent généralement l'extraction de caractéristiques locales et l'entraînement du modèle de régression de manière indépendante, ils proposent une méthode supervisée basée sur la propagation de l'erreur du modèle SVR. À partir de cette propagation d'erreur, le premier modèle crée des filtres de caractéristiques locales discriminantes qui peuvent être appliqués aux patchs d'images locales. Ainsi, les filtres de caractéristiques sont générés automatiquement afin de répondre à la tâche demandée. Le Deep Learning a été également utilisé pour répondre à cette problématique. Kang en 2014 (KANG et al. 2014) a notamment exploité les réseaux de neurones par convolution comme modèle de régression pour répondre à la tâche d'émission du score de qualité de l'image. Le réseau se compose d'une couche convolutive dite de *Pooling* max et min [1], puis de deux couches entièrement connectées et d'un nœud de sortie (le score de prédiction). À noter qu'il existe d'autres approches exploitant le Deep Learning (BOSSE et al. 2017 ; Y. LI et al. 2016) ayant des architectures de réseaux différentes, mais qui ne seront pas détaillées dans le cadre de ce manuscrit.

4.2.3.3 Métrique intégrée dans un contexte de rendu

Les différentes métriques de qualité d'images présentées dans les sections précédentes répondent généralement à un bruit additif ou à une perte d'information dans une image,

1. Une couche de convolution appelée *Pooling* vise à ne conserver par convolution que la valeur la plus grande d'un pixel ou la plus petite des pixels présents dans la taille de filtre définie

voire une séquence d'images. Elles sont testées sur des bases de données d'images naturelles pour lesquelles des scores humains subjectifs de qualité ont été obtenus, telles que LIVE (H. SHEIKH et al. 2005) et TID (PONOMARENKO et al. 2015). Dans un contexte de rendu d'image photo-réaliste, le bruit peut-être identifié comme étant un bruit résiduel dû à la convergence de Monte-Carlo pour l'estimation finale de la valeur du pixel. De plus, l'image de référence n'est jamais disponible lors du rendu de telles images.

Pour plus d'informations, le lecteur peut se référer à l'étude de Lavoué et Mantiuk (LAVOUÉ et al. 2015) portant sur les métriques de qualité appliquées au domaine de l'informatique graphique. Ils y mettent en avant les résultats et les limites de telles métriques relativement au réalisme et au bruit résiduel généré lors du rendu d'une image. Ils exposent également l'émergence de méthodes d'apprentissage automatique comme outil potentiel visant à répondre au problème rencontré compte tenu de la difficulté de la tâche.

4.3 Détection de bruit dans les images de synthèse

Lors du rendu d'une image de synthèse, une question pouvant susciter l'intérêt est celle de trouver un critère d'arrêt de calcul local. Précédemment, dans le chapitre 3, nous avons présenté les méthodes visant à améliorer la rapidité et la convergence du calcul. Toutefois, nous ne savons pas si en sortie du moteur de rendu, l'image finale possède encore du bruit perceptible ou non pour un humain. Trouver un tel critère d'arrêt, fidèle à la perception humaine, permettrait à la fois de savoir quelle partie de l'image ne nécessite plus de calcul et celles en ayant encore besoin, mais également, de s'assurer que la qualité de l'image finale est telle qu'attendue pour un humain.

4.3.1 Procédure d'intégration

Pour mesurer et identifier s'il est possible ou non d'arrêter d'échantillonner lors du rendu, un modèle de perception de bruit est nécessaire. Ce modèle prend en entrée l'image qu'il doit traiter et définit s'il est encore nécessaire ou non de l'améliorer, c'est-à-dire, si un bruit résiduel perceptible est encore présent ou non. La figure 4.5 illustre l'interaction d'un tel modèle lors du rendu d'une image de synthèse. Le modèle pourrait émettre un arrêt ou non du calcul lors du rendu de l'image.

Émettre un critère d'arrêt sur l'image complète n'est pas forcément le plus judicieux, le bruit étant présent de manière hétérogène. Au contraire, l'idéal serait de travailler au niveau du pixel même, mais capturer des seuils perceptifs pour chaque pixel semble difficile. La figure 4.6 propose un compromis qui consiste à traiter des blocs de taille régulière de l'image, plutôt que l'image complète ou le pixel. Ainsi, pour chaque bloc, il

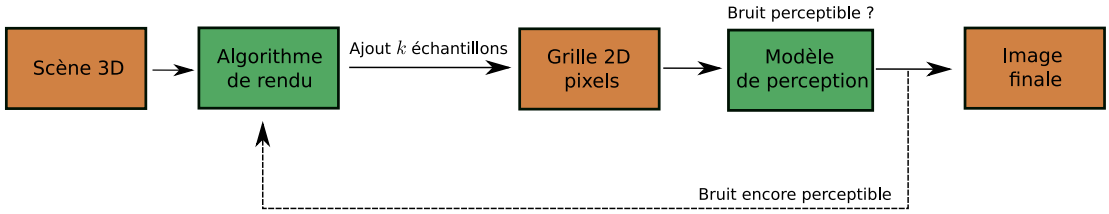


FIGURE 4.5 – Processus de rendu d'une image de synthèse avec interaction et interrogation du modèle de perception.

est possible de savoir si l'échantillonnage est encore nécessaire ou non étant donné qu'un bruit perceptible est encore présent. Le budget alloué à l'échantillonnage peut ainsi être économisé et localisé d'une meilleure manière sur les blocs comportant encore du bruit perceptible. Le temps de calcul est également par conséquent réduit.

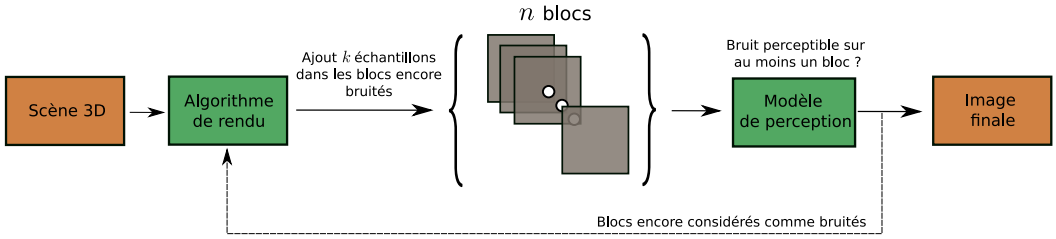


FIGURE 4.6 – Processus de rendu d'une image de synthèse avec interaction et interrogation du modèle de perception sur des parties de l'image pour permettre une meilleure précision et localisation du bruit.

L'obtention d'un tel modèle n'est pas une tâche facile et reste un problème aujourd'hui encore ouvert. L'une des approches possibles est que ce modèle soit au préalable entraîné sur des données subjectives humaines permettant de quantifier le bruit. Dans le cadre d'un bruit localisé comme présenté dans la figure 4.6, il est nécessaire de savoir dans les données d'entraînement si le bloc avec un certain nombre d'échantillons, était encore considéré ou non comme bruité. Pour cela, des seuils de perception humains sont généralement récoltés, permettant de spécifier à partir de combien d'échantillons dans un bloc de l'image, l'humain ne perçoit plus de bruit. Ainsi, à partir de cette information, les blocs échantillonnés avec une valeur inférieure à ce seuil sont considérés comme bruités et dans le cas contraire, comme non bruités.

4.3.2 Propositions de critère d'arrêt

À partir de cette idée de seuil humain subjectif et expérimental, plusieurs approches d'extractions de caractéristiques fines relatives à la perception du bruit résiduel ont été

proposées pour permettre l'entraînement d'un modèle d'apprentissage.

(MYSZKOWSKI 1998) propose d'étudier l'utilisation du VDP au sein du moteur de rendu. Il propose l'étude du VDP sur plusieurs expérimentations dont l'une ciblée sur la comparaison de la progression de la solution d'éclairage indirect. Ainsi à partir de ces mesures de différences fournies par le VDP, il est possible de fixer un seuil pour émettre un critère d'arrêt de calcul de rendu dans une partie de l'image. Il ne propose pas ici directement de modèle d'apprentissage, mais initie l'extraction d'information sur la perception humaine lors du rendu d'images de synthèse.

Les travaux réalisés par (TAKOUACHET et al. 2017) et ceux (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) utilisent une base de 12 images calculées (proposée par (TAKOUACHET et al. 2017)) avec différents niveaux d'échantillonnage. Ces images, de taille 512×512, ont été découpées en 16 blocs disjoints (de taille 128×128) pour lesquels des seuils de prédiction (nombre d'échantillons) de bruit ont été acquis auprès d'utilisateurs humains. Pour ces 3 approches qui vont être détaillées, il a été proposé de générer dans un premier temps une image de référence approximative à l'aide d'une technique de lancer de rayons (WHITTED 1979) qui est calculée une fois pour toute au début du processus de rendu de l'image.

(TAKOUACHET et al. 2017) ont proposé en 2017 de calculer un masque de bruit comme réalisé en imagerie satellite (DRAGESCO 1995) à partir du canal de luminance L, à la fois pour l'image courante obtenue pendant le processus de rendu et pour l'image de référence approchée. Tout d'abord, pour obtenir un masque de bruit, une image floue est calculée à partir de l'image originale en utilisant une convolution gaussienne 3×3 avec un coefficient de convolution $\sigma \in \{0.3, 1.5\}$. Ensuite, ce masque de bruit est obtenu en calculant la différence entre l'image originale et l'image floue. Enfin, les deux masques sont passés à un modèle SVM qui permet de classer le bloc d'image courant comme étant encore bruité ou non, chaque bloc ayant une taille de 128×128. Le processus d'extraction de ces informations est proposé dans la figure 4.7. Le fait de donner 2 images directement en entrée d'un modèle SVM peut conduire à ce que l'on appelle le problème de la malédiction de la dimension, surtout si la taille de l'image augmente, tout comme la quantité de données d'apprentissage. Le modèle SVM aurait donc besoin de plus de temps pour apprendre et trouver son plan d'hyper séparateur.

(J. CONSTANTIN, BIGAND et al. 2015) ont proposé d'extraire 26 caractéristiques des blocs d'image. Ils extraient d'abord le canal L de l'espace couleur CIE $L^*a^*b^*$ puisque le bruit semble affecter essentiellement la composante L (CARNEC et al. 2008). À partir du canal de luminance L, ils appliquent quatre algorithmes de débruitage différents :

— filtrage linéaire avec des filtres de moyennage de tailles 3×3 et 5×5 (2 images filtrées) ;
— filtrage linéaire avec filtres gaussiens de tailles 3×3 et 5×5 et avec des écarts types $\sigma \in \{0.5, 1, 1.5\}$ (6 images filtrées) ;

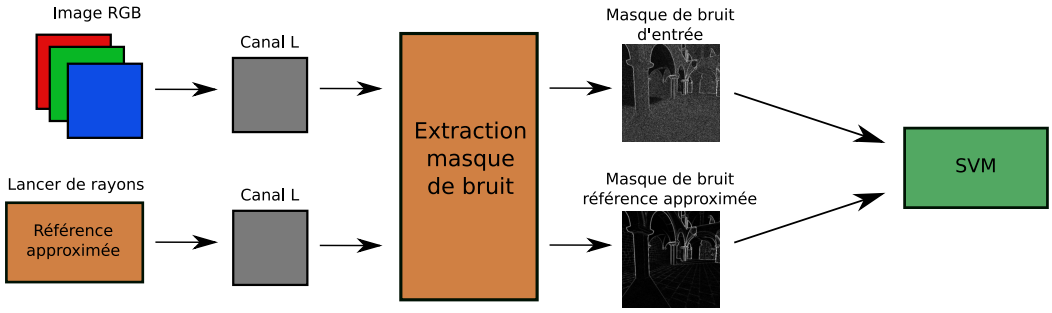


FIGURE 4.7 – Processus d'extraction des masques de bruit à partir du bloc pour conception des données d'entrée au modèle SVM par l'approche de (TAKOUACHET et al. 2017).

— des filtres médians de tailles 3×3 et 5×5 (2 images filtrées) ;
— des filtres de Wiener adaptatifs de tailles 3×3 et 5×5 (2 images filtrées) ;
— filtrage après décomposition en ondelettes 2-D (1 image filtrée).

Il précise qu'un filtrage passe-bas utilisant la moyenne ou des filtres gaussiens permet de modéliser le bruit haute fréquence. Le filtrage médian non linéaire s'attaque au bruit « poivre et sel » et le filtrage de Wiener permet d'adapter la suppression du bruit à la variance locale des pixels.

La dernière image filtrée est obtenue en utilisant la décomposition en ondelettes 2-D où l'image d'entrée est décomposée en quatre sous-bandes, à savoir les sous-bandes basse (LL), basse-haute (LH), haute-basse (HL) et haute-haute (HH). Parmi ces quatre sous-bandes, la sous-bande LL contient les composantes de basse fréquence, tandis que les trois autres sont les composantes de haute fréquence. Dans la littérature, il a été observé que pour une grande classe d'images, les coefficients d'ondelettes dans les sous-bandes LH, HL et HH ne suivent pas une distribution gaussienne et sont utilisés pour reconstruire une image filtrée sans les données des LL.

Le tableau 4.1 propose un aperçu des 13 images obtenues après application des filtres proposés sur l'image en niveaux de gris L.

Ainsi, à partir de chacun des filtres appliqués, 13 images en niveaux de gris sont obtenues et pour chacune la différence absolue à l'image L initiale est calculée. La moyenne et l'écart-type sont extraits de ces différences afin d'obtenir 26 caractéristiques. La même procédure est réalisée pour l'image de référence approchée afin d'obtenir 26 caractéristiques de bruit également. La différence entre les 26 attributs extraits avec l'image courante et l'image de référence approchée est ensuite calculée. Le processus d'obtention de cette différence sur les 26 caractéristiques est illustré dans la figure 4.8. Ces caractéristiques sont fournies comme entrées d'un classifieur *Support Vector Machine* (SVM) qui permet de prédire le label attendu (bruité, non bruité) de chaque

		taille de la convolution	
type de filtre		3×3	5×5
filtre linéaire		image 1	image 2
filtre gaussien	$\sigma = 0,5$	image 3	image 6
	$\sigma = 1,0$	image 4	image 7
	$\sigma = 1,5$	image 5	image 8
filtre médian		image 9	image 10
fitre de Wiener adaptatif		image 11	image 12
ondelettes		image 13	

TABLEAU 4.1 – Représentation des 13 images obtenues après application de filtres sur l'image de canal L.

bloc d'image.

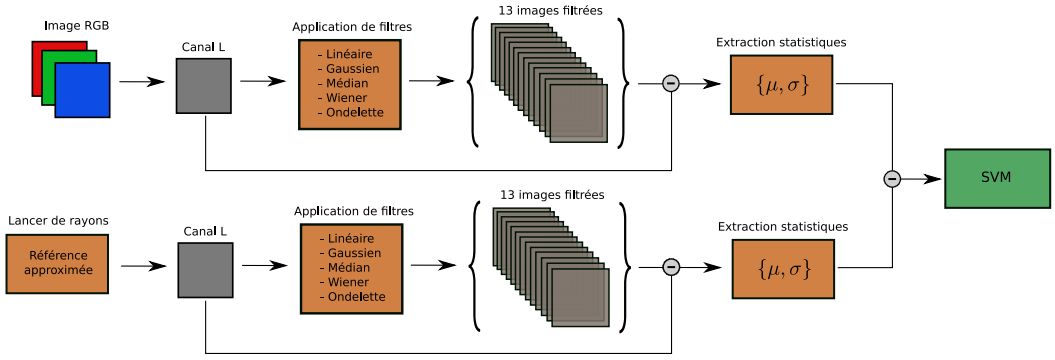


FIGURE 4.8 – Illustration du processus d'extraction des 26 caractéristiques à partir du bloc proposé par l'approche de (J. CONSTANTIN, BIGAND et al. 2015) comme entrée du modèle SVM.

Dans la même idée que leur précédente approche, (J. CONSTANTIN, I. CONSTANTIN et al. 2016) ont utilisé l'apprentissage semi-supersivé pour permettre un apprentissage plus robuste de leur modèle. Contrairement à l'approche précédente, (J. CONSTANTIN, BIGAND et al. 2015), les filtres sont appliqués les uns après les autres sur la luminance L à la fois de l'image courante et de l'image approchée. Les deux images obtenues sont considérées comme des images débruitées. Pour chacune des images débruitées, une image de différence est obtenue par soustraction pixel par pixel entre l'image originale et l'image débruitée. Ainsi, les données d'entrée du modèle sont le vecteur de la différence des deux dernières images générées. La création de l'image d'entrée qui compose le vecteur d'entrée au modèle SVM est schématisée dans la figure 4.9.

Pour ces trois dernières approches, le calcul de la référence approchée ne couvre

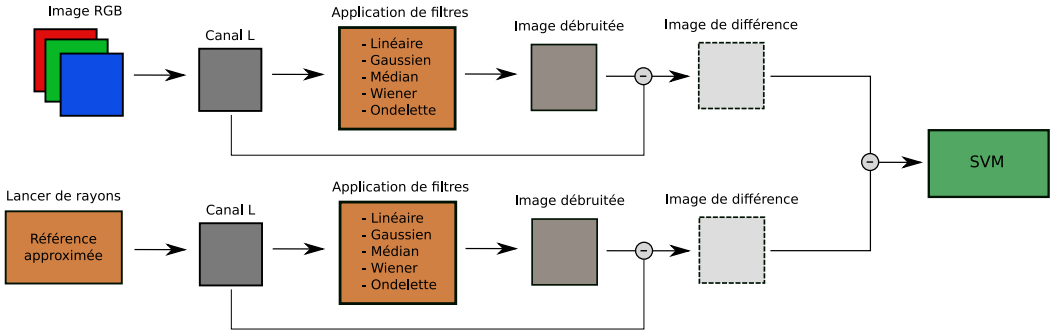


FIGURE 4.9 – Illustration du processus de conception de l'image à partir du bloc de l'approche de (J. CONSTANTIN, I. CONSTANTIN et al. 2016) comme entrée au modèle SVM.

pas tous les effets lumineux, tels que les caustiques, l'éclairage global, la dispersion lumineuse et peut dans certains cas amener quelques difficultés d'obtention de bonnes images approchées. Cela est généralement le cas sur des scènes où l'éclairage indirect est très important. De plus, il est important de noter que les effets visuels apparaissant dans la base d'image exploitée par (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) et (TAKOUACHET et al. 2017), ne couvrent pas tous les effets qui existent dans les images réelles et que les scènes utilisées possèdent une géométrie assez simplifiée.

Résumé

L'exploitation de la perception humaine en image de synthèse a suscité un intérêt croissant au fur et à mesure des années, en particulier du fait de la présence de bruit dans les images générées. Celui-ci est à la fois hautement perceptible et difficilement quantifiable et son « élimination » implique des coûts de calculs importants. L'utilisation de métriques et d'outils pour caractériser le bruit provenant d'images naturelles ne semble pas forcément être le plus adaptée à la nature du bruit de Monte-Carlo. Identifier et détecter la présence de bruit au travers d'un modèle de perception entraîné est une tâche complexe, mais permettrait de répondre à cette problématique. En effet, ce modèle requiert une grande diversité des scènes et des effets lumineux afin de pouvoir répondre au mieux sur des scènes non connues. Trouver les caractéristiques permettant d'identifier un tel bruit est difficile. La partie II de ce manuscrit, vise à présenter les contributions réalisées dans le cadre de la thèse relatives à ces deux problématiques, la diversification des seuils humains pour couvrir un grand nombre d'effets lumineux et la définition de caractéristiques fines pour permettre un apprentissage robuste et généralisé du modèle.

Deuxième partie

Critère d’arrêt et méthode de réduction du bruit

Chapitre 5

Base d'images photo-réalistes

Introduction

Ce chapitre présente l'une des premières contributions réalisées dans le cadre de cette thèse. Comme mentionné dans la section 4.2.3, les bases d'images actuelles possédant des seuils subjectifs de qualité, sont soit issues d'images naturelles, soit dans le domaine de la synthèse d'images, la diversité des scènes proposées reste faible (scènes extérieures ou non, effets lumineux, éclairage indirect ou non, textures et matériaux...) et peut amener à un manque de généralisation pour les modèles de détection de bruit à étudier. De plus, la quantité de données et la diversité des scènes est un atout pour des méthodes d'apprentissage profond qui se montrent de plus en plus performantes et robustes. Dans ce chapitre, la conception d'une nouvelle base d'images de synthèse, son contenu, tout comme sa procédure de récolte de seuils subjectifs, sont détaillés.

Les modèles de perception du bruit qui ont été proposés à ce jour ont été entraînés sur des scènes ayant des effets lumineux restreints, ne couvrant pas ceux liés aux caustiques, à l'éclairage global et à la dispersion lumineuse. Les scènes utilisées pour l'apprentissage de ces modèles possédaient une géométrie parfois simplifiée, car plus ancienne. De plus, la problématique de la quantité et la diversité de données nécessaires pour couvrir l'ensemble des effets lumineux et des types de scènes n'est pas à négliger. La conception de modèles issues de l'apprentissage profond, comme mentionné dans la littérature, apporte de bons résultats en terme de débruitage, car ils sont davantage adaptés à une grande quantité de données. Il serait intéressant d'explorer l'apprentissage profond pour notre problème de détection de bruit.

Dans ce sens, il a été envisagé en début de thèse de créer une base d'images conséquente, permettant de couvrir un grand nombre d'effets lumineux et de types de scènes récentes. Les choix de paramètres pour la génération de telles images et de variété des scènes utilisées sont détaillés dans la section 5.1. Le format des données sauvegardées en sortie du moteur de rendu est présenté dans la section 5.2. Enfin, le processus et

l'application permettant l'acquisition de seuils subjectifs humains est ensuite exposée dans la section 5.3.

5.1 Choix de paramètres de génération

Générer une telle base d'images est assez ambitieux. En effet, le temps de calcul est conséquent pour l'obtention d'images photo-réalistes, surtout en grande quantité. De plus, un grand nombre de paramètres est à prendre en compte et à sélectionner judicieusement.

5.1.1 Moteur de rendu

Afin de générer un ensemble assez grand d'images de synthèses, le moteur de rendu qui a été utilisé dans le cadre de cette thèse est PBRT version 3 (https://www.pbrt.org/), pour *Physically Based Rendering* (PHARR et al. 2016). Il s'agit d'un moteur de rendu basé physique et donc orienté sur la génération d'image photo-réaliste, qui de plus est fortement utilisé dans la littérature et la communauté scientifique. C'est un moteur de rendu ouvert développé en C++, qui offre à la fois une bonne performance de calcul grâce à une parallélisation adaptée de la génération de l'image, et la possibilité de modifier et adapter le code pour de nouvelles implémentations et travaux de recherches.

5.1.2 Choix des paramètres

Comme présenté dans le Chapitre 1, le choix de la méthode de rendu est un paramètre important. En effet, de nombreuses méthodes d'intégration sont disponibles telles que le tracé de chemins, le BDPT ou encore le MLT. L'étude menée par Zhu (ZHU 2020) sur l'utilisation du machine learning dans un cadre de production, mentionne l'intérêt qui est porté sur l'utilisation de la méthode classique de tracé de chemins, où les chemins sont émis depuis la caméra. Parmi les avantages cités figurent une grande facilité de développement, une bonne stabilité des résultats obtenus en animation et une grande facilité à paralléliser aussi bien sur processeur (CPU) que sur carte graphique (GPU). Nous avons donc opté pour une méthode d'intégration orientée tracé de chemins classique pour la génération de nos images.

Un autre paramètre important dans le rendu d'une image de synthèse, concerne la manière d'échantillonner dans le pixel lorsque l'on souhaite envoyer un rayon depuis la caméra vers ce pixel pour obtention de sa contribution. Beaucoup de méthodes d'échantillonnages existent telles qu'un échantillonnage suivant une séquence de *Sobel* (SOBOL' 1967), stratifié (HUNT et al. 2001) pour une réduction de variance ou encore aléatoire. Pour rester conforme au bruit résiduel généré et à la loi des grands nombres pour l'échantillonnage de Monte-Carlo (P.-L. HSU et al. 1947), la manière d'échantillonner exploitée pour la conception de cette base d'images est simplement aléatoire.

5.1.3 Variété des scènes

Un grand nombre de scènes est proposé pour le moteur de rendu PBRT (BITTERLI 2016), aussi bien des scènes d'intérieurs telles que des cuisines, salles de bain, salons, qu'extérieurs où l'éclairage est généralement plus direct et le calcul généralement moins long, ou encore celles d'objets isolés. Certaines sont assez claires et d'autres assez sombres, ce qui dans notre cas est un critère important à considérer afin de permettre de localiser le bruit dans diverses conditions d'éclairage présentes dans une scène. La majorité d'entre-elles sont composées d'éclairage indirect (notamment les scènes d'intérieurs) ce qui entraîne généralement un bruit résiduel plus fort, plus difficile à réduire et nécessitant un temps de calcul plus conséquent. De plus, parmi ces scènes certaines sont également composées d'effets lumineux complexes, comme les caustiques, qui nécessitent également un temps de convergence plus long.

La figure 5.1 offre un aperçu de plusieurs rendus de point de vue de scènes et effets lumineux qui peuvent être obtenus avec le moteur PBRT. La scène *Caustic* permet de simuler l'effet lumineux d'une caustique qui se reflète sur le sol. La scène *Bathroom*, qui est une scène d'intérieur, présente un bruit résiduel fort sur des zones sensibles à l'éclairage indirect, notamment la partie centrale haute. À noter que la texture sur le sol, a tendance à réduire considérablement le bruit résiduel perçu, car elle apporte un effet de masquage de ce bruit. La scène *Landscape* qui est une scène d'extérieur a pour particularité de converger rapidement, car l'éclairage est principalement direct. Elle présente également une profondeur de champs qui impacte la perception. La scène *San-Miguel* qui présente ici aussi bien de l'éclairage direct qu'indirect et une complexité géométrique importante.

(a) *Caustic* (b) *Bathroom* (c) *Landscape* (d) *San-Miguel*

FIGURE 5.1 – Aperçu d'images photoréalistes générées par le moteur de rendu PBRT avec 10 000 échantillons par pixel.

Parmi l'ensemble des scènes PBRT disponibles, 28 ont été sélectionnées pour concevoir la base d'images. Pour certaines scènes complètes (scène d'intérieur ou d'extérieur ne proposant pas le visuel d'un objet seul par exemple), plusieurs points de vue ont été extraits. Cela permet notamment d'augmenter la taille de la base d'images à 40 points de

vue. L'annexe A.4 offre un visuel des différents points de vue qui composent l'intégralité de cette base d'images de synthèse. Les nouveaux points de vue ont été obtenus à l'aide d'un logiciel permettant de lire un fichier de scène PBRT et de naviguer dans celle-ci afin d'obtenir les nouvelles coordonnées de caméra souhaitées. Ce logiciel, nommé *vpbrt*, est disponible en ligne : https://github.com/prise-3d/vpbrt.

5.2 Format des images

Avant de générer directement les images en sortie depuis le moteur de rendu, quelques points importants ont été abordés en amont afin de permettre l'utilisation de cette base d'images à d'autres cas d'études.

5.2.1 Nouvelle extension de fichier

Un format de sortie d'image classique du moteur peut-être le format PNG où la couleur de chaque pixel est disponible dans un intervalle $[0, 2^8]$, ou bien le format EXR qui est un format Open Source, matriciel multicanal à gamme dynamique élevée. Il est généralement exploité pour de l'imagerie HDR dont l'intervalle des couleurs disponibles est $[0, 2^{10}]$, mais peut supporter des dynamiques encore plus importantes (comme des flottants en 32 ou 64 bits). Afin d'étendre les études menées sur cette base d'images et de ne pas la générer uniquement pour répondre à la tâche de recueil de seuils subjectifs et d'apprentissage d'un modèle de perception, un nouveau format de sortie a été défini. En effet, en sortie du moteur de rendu, un opérateur de ton de couleurs de l'image, en anglais *Tone Mapping*, est appliqué sur les intensités lumineuses obtenues avant sauvegarde du fichier. Le plus connu est l'application d'une correction gamma, qui pour une intensité de couleur I, permet d'obtenir une nouvelle valeur de sortie telle que $I' = I^\gamma$. La figure 5.2 propose un aperçu des résultats obtenus après application d'un tel opérateur.

Le format qui a été envisagé, permet de stocker directement les valeurs de luminances calculées avant traitement par le moteur de rendu, soit une valeur réelle de 32 bits. Ce nouveau format de fichier est nommé RAWLS pour *RAW Light Simulation* étant donné qu'un format RAW classique vise à stocker les données brutes. Ainsi, la base d'images générée pourra être utilisée pour des travaux étendus tels que des études d'imagerie HDR et de différence perceptuelle après application d'opérateurs de *Tone Mapping*.

5.2.2 Génération des images d'expérimentation

Pour chacun des 40 points de vue sélectionnés, 10 000 images de résolution 1920×1080 pixels de 1 échantillon par pixel ont été sauvegardées. L'objectif de cette approche est de permettre par la suite des études statistiques fines sur les distributions des pixels

(a) $\gamma = 0.2$ (b) $\gamma = 1$ (c) $\gamma = 1.8$

FIGURE 5.2 – Aperçu de différentes valeurs de correction γ appliquées à l'image originale ($\gamma = 1$) de la scène *Kitchen*.

(ELEK et al. 2019), comme celle proposée pour la contribution détaillée dans le chapitre 6. Cette approche nécessite toutefois un coût de stockage conséquent des images de 12 To environ.

Les images obtenues au format RAWLS ne sont pas en l'état exploitables. En effet, la luminance en question ne serait pas visible pour un utilisateur sans traitement tout comme la tâche de trouver un seuil sur 10 000 images de niveaux de bruits différents serait beaucoup trop longue et difficile, bien que très précise. Pour permettre une acquisition à la fois relativement précise et moins lourde de seuils subjectifs lors d'expérimentations, les 10 000 échantillons sauvegardés (1 image pour 1 échantillon par pixel) ont été moyennés succesivement et, de cette manière, tous les 20 échantillons une nouvelle image a été sauvegardée au format PNG. Ainsi, 500 images de niveaux de bruit différents sont obtenues. Par défaut, pour le moment, le même opérateur de correction gamma que celui de PBRT est utilisé en post-traitement sur le fichier *.rawls* afin d'obtenir une image PNG en sortie qui soit plus facilement lisible. Ainsi, pour les 40 points de vue, 500 images sont obtenues pour chacun d'entre eux, avec des niveaux de bruit allant de 20 à 10 000 échantillons, par pas de 20 échantillons.

5.3 Processus de recueil de seuils subjectifs

À partir des images générées précédemment, un processus d'expérimentation a été proposé pour l'acquisition de seuils de perception du bruit. À cet effet, une application en ligne a été développée par Antoine Sauvage, étudiant en deuxième année d'IUT Informatique que j'ai pu encadrer techniquement.

5.3.1 Présentation de l'application

Cette application a pour objectif le passage d'expérimentations auprès de participants afin de récolter des seuils subjectifs, c'est-à-dire, un nombre d'échantillons à partir duquel le participant ne voit plus de différence avec une image de référence. Pour un point de vue, deux images sont présentées au participant, celle de gauche comportant le niveau de bruit le plus fort, soit celle de 20 échantillons par pixel et à droite celle de référence, soit celle de 10 000 échantillons par pixel.

Dans le cadre de cette application, les images générées sont recadrées en leur centre et font la taille de 800×800 pixels afin de permettre un visuel à la fois de l'image bruitée à gauche et de l'image de référence à droite sur un écran de résolution 1920×1080 pixels. Les seuils subjectifs sont acquis comme pour les travaux de (TAKOUACHET et al. 2017) sur des blocs plus réduits. Ainsi, l'image est découpée en 16 blocs de taille 200×200 afin d'avoir une assez bonne précision de la localisation du bruit. Réduire davantage la taille de ces blocs implique un processus d'acquisition beaucoup plus lourd pour les participants. La figure 5.3 résume la manière dont les blocs de taille 200×200 sont extraits sur les images initialement générées.

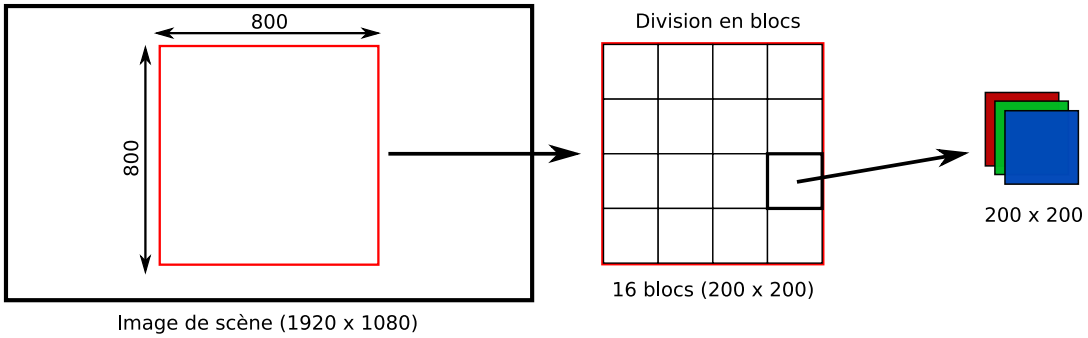
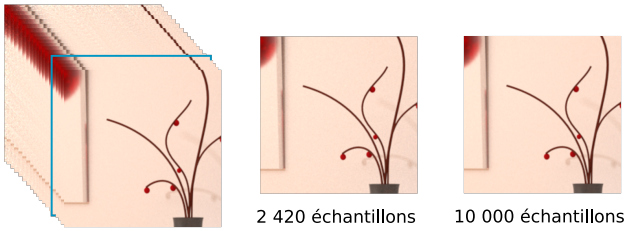


FIGURE 5.3 – Processus de division des blocs de taille 200×200 sur les images générées.

Pour chaque point de vue, 16 blocs sont extraits avec chacun 500 niveaux de bruit. Le participant ayant accès à l'application en ligne peut, sur l'image de gauche, cliquer sur chaque bloc 200×200 pour augmenter ou réduire le nombre d'échantillons, afin de respectivement réduire ou augmenter le bruit perçu. La tâche demandée au participant de cette expérience est de faire correspondre l'image de gauche sur laquelle il peut interagir, avec l'image de droite, celle de référence. La figure 5.4 propose un aperçu de l'application et de la manière dont le participant peut interagir avec les blocs de l'image de gauche. Ces seuils subjectifs ont été obtenus sur une population experte du domaine de l'informatique graphique fort sensible au bruit perçu. Les participants sont les membres de l'équipe Image et Apprentissage (IMAP) du laboratoire d'Informatique Signal et Image de la Côte d'Opale (LISIC). L'objectif visé est que le modèle puisse apprendre sur

des seuils experts et éviter tout seuil moyen pouvant laisser persister un bruit visuel pour certaines personnes.

(a) Vue d'ensemble de l'utilisateur sur le point de vue de la scène *Living-room*. Image partiellement bruitée à gauche, image de référence à droite.



(b) Aperçu de l'interaction possible du participant sur un bloc. À gauche, le tampon pré-calculé d'images du bloc. Au milieu, l'image du bloc correspondant au seuil actuel fixé. À droite, l'image de référence du bloc.

FIGURE 5.4 – Aperçu du fonctionnement de l'application de recueil de seuils subjectifs **(a)** offre un aperçu de l'interface utilisateur pour l'ensemble du point de vue. **(b)** décrit comment le participant peut changer le nombre d'échantillons du bloc : il peut naviguer dans le tampon d'images pré-calculées pour faire correspondre visuellement les deux images pour ce bloc. L'image bordée de bleu dans le tampon, est l'image actuellement sélectionnée et affichée au centre.

5.3.2 Seuils subjectifs obtenus

On peut noter ici que la procédure s'applique dans un cadre dit de *Just Noticeable Difference* (JND), où le participant est supposé percevoir ou ne pas percevoir une différence entre deux images. Lorsque le participant considère que les images gauche et droite correspondent, les différents niveaux d'échantillonnage sont enregistrés et sont supposés être ses seuils visuels de perception du bruit. La moyenne des scores obtenus pour tous les participants, également appelée *Mean Opinion Score* (MOS), est alors considérée comme le seuil humain et sera utilisée dans nos études. La figure 5.5 permet de visualiser l'image obtenue à partir des seuils subjectifs humains sur les 16 blocs de la scène *Living room*. Cette image reconstruite apparaît relativement proche en score SSIM de l'image de référence avec parfois un nombre d'échantillons bien inférieur dans certains blocs de l'image. Les blocs 4, 8 et 9 semblent ainsi ne pas nécessiter un grand nombre d'échantillons avec un ratio de deux à quatre fois moins d'échantillons par rapport à l'image de référence.

Le tableau en annexe A.1 présente les seuils subjectifs moyens (MOS) obtenus par les 5 participants de l'équipe sur les 40 images proposées. On peut noter que certains blocs d'images ont été considérés comme encore bruités même avec 10 000 échantillons

(a) Numérotation des blocs d'images

8680	5286	6553	2293
7153	7086	6420	2626
4420	6580	4560	7820
7086	8286	6286	6693

(b) Seuils subjectifs (MOS)

(c) Image reconstruite SSIM : 0.9903

(d) Référence SSIM : 1.0

FIGURE 5.5 – Reconstruction des blocs de l'image et comparaison de l'image *humaine* à l'image de référence.

par pixel, ce qui montre bien la sensibilité des participants au bruit. Le tableau en annexe A.2 quant à lui, présente les écart-types obtenus sur les valeurs des seuils subjectifs sur chaque bloc pour les 5 participants de l'équipe. Pour une meilleure représentation des écarts, nous utilisons le coefficient de variation défini par le ratio entre l'écart-type et la moyenne des seuils des participants. Il permet une représentation des résultats sans unité pour mieux juger de la dispersion des données. Le tableau en annexe A.3 propose les valeurs de coefficient de variation pour chacun des blocs des 40 images. Plus le coefficient de variation est élevé, plus la dispersion est apparente. Les images *Eponge Fractal vue 1* et *Landscape* sont les images où les plus grandes valeurs de coefficient de variation sont présentes. Le coefficient reste assez élevé pour certaines autres scènes, mais dans l'ensemble, il semble que les participants sont relativement en accord.

5.3.3 Labellisation des blocs d'images

L'objectif final premier de la création de cette base d'images est d'obtenir un ensemble d'apprentissage basé sur des seuils subjectifs humains, afin de pouvoir procéder à la création d'un modèle d'apprentissage automatique. Ce modèle serait utilisé en tant qu'expert, étant donné qu'il aurait appris sur des données humaines, pour décider à quel moment il serait judicieux de s'arrêter lors du rendu d'un bloc de l'image. Afin d'apprendre un tel modèle de décision binaire, il nous faut préciser une étiquette (label) associé à un bloc et à un niveau de bruit (nombre d'échantillons par pixel ou encore spp). La figure 5.6 permet d'illustrer comment les images de différents niveaux de bruits d'un bloc sont labellisées en fonction du seuil humain subjectif moyen obtenu pour ce bloc.

Ainsi, les données une fois labellisées peuvent être exploitées pour permettre l'apprentissage d'un modèle. Les contributions réalisées et présentées dans les chapitres 7 et 8 exploitent cette base d'images afin de proposer un tel modèle.

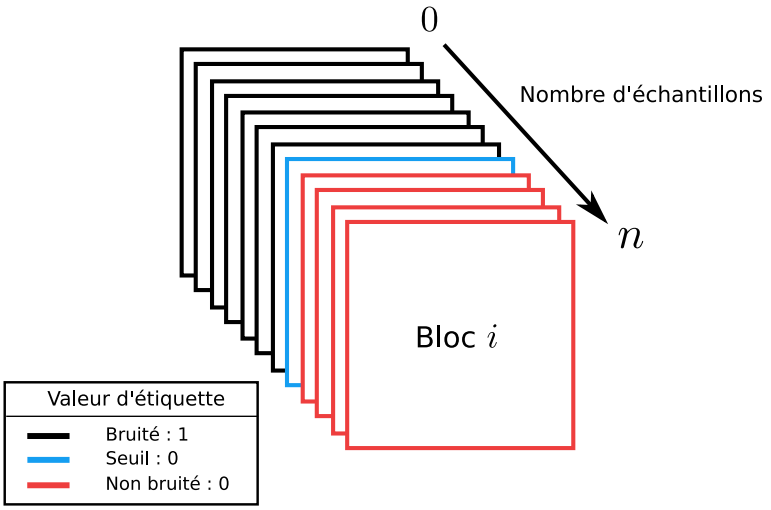


FIGURE 5.6 – Labellisation des images disponibles d'un bloc en fonction du seuil humain subjectif

5.3.4 Ouverture de la base d'images

Afin d'offrir l'accès à cette base d'images et aux seuils obtenus, celle-ci est disponible sur la plateforme Open Source *Zenodo* qui offre la possibilité de partager des quantités de données conséquentes : https://zenodo.org/record/4964303. Cette base d'images est composée de l'ensemble des 500 images des 40 points de vue exploités au format PNG avec leurs niveaux d'échantillonnage allant de 20 à 10 000 échantillons par pixel. Elle contient également l'ensemble des seuils humains obtenus pour ces 40 points de vue. À ce jour, partager les informations brutes des données sauvegardées au format RAWLS n'est pas envisageable étant donné la taille de stockage. Toutefois, un travail est en cours pour un accès partiel à ces données via une application web.

Une partie de ces données ont déjà été exploitées par le laboratoire SCALab de Lille dans le cadre du projet ANR PrISE-3D afin de procéder à des expériences de récoltes de seuils fins sur quelques images. Ils souhaitent proposer des protocoles expérimentaux pour permettre l'obtention de données précises de seuils perceptuels.

Résumé

Dans ce cinquième chapitre, nous avons détaillé une des contributions réalisée dans le cadre de la thèse visant à proposer une base d'images de seuils subjectifs humains pour des images de synthèse. Cette base est d'une taille conséquente et d'une variété de scènes assez importante. Elle comprend un grand nombre d'effets lumineux qui sont dépendants de la nature de la scène tels que des scènes d'intérieur comme une cuisine, salle de bain, salon, qu'extérieur où l'éclairage est plus direct, ou encore de scènes représentant un objet en particulier. Cette base d'images a pour objet d'être exploitée pour la tâche souhaitée qui est la conception d'un modèle expert de détection de bruit du bruit résiduel de Monte-Carlo. La méthodologie abordée pour la conception de cette base d'images met en avant le souhait d'étendre les études sur le bruit de Monte-Carlo à mener. Le prochain chapitre présente les contributions de conception de modèles de perception réalisées à partir de cette base d'images et profitant des dernières avancées sur l'apprentissage profond.

Chapitre 6

Gestion des valeurs aberrantes lors du rendu

Introduction

Ce chapitre introduit une contribution réalisée au cours de la thèse qui traite du problème du choix d'un estimateur lors de l'évaluation de l'équation du rendu par méthode de MC en chaque pixel. Cette contribution n'est pas liée directement à la détection de bruit, mais plutôt à la réduction d'un bruit particulier lors du rendu. En effet, la moyenne est classiquement utilisée pour estimer la valeur d'un pixel. Toutefois, lorsque qu'une valeur aberrante est rencontrée, il est difficile de juger s'il faut la considérer ou non pour le calcul final de la moyenne étant donné que les échantillons sont obtenus les uns après les autres et sommés. La conserver peut impliquer une valeur de pixel erronée qui est caractérisée dans la littérature par le nom de *firefly* (luciole). Ce problème a été rencontré lors de la génération de la base d'images où pour certains points de vue et certains pixels, des valeurs aberrantes ont été rencontrées et génèrent des pixels dits scintillants.

L'estimation de l'équation du rendu à l'aide des méthodes de Monte-Carlo produit des images photoréalistes en évaluant un grand nombre d'échantillons de l'équation du rendu par pixel. La valeur finale pour chaque pixel est alors généralement calculée comme la moyenne de la contribution de chaque échantillon. La moyenne est un bon estimateur, mais il n'est pas nécessairement robuste, ce qui explique l'apparition de certains artefacts visuels tels que les lucioles (en anglais *fireflies*), dus à une surestimation de la valeur de la moyenne. Le MoN (Median of meaNs) est un estimateur plus robuste que la moyenne qui permet de réduire l'impact des valeurs aberrantes qui sont la cause de ces *fireflies*. Cependant, cette méthode converge plus lentement que la moyenne, ce qui réduit son intérêt pour les pixels dont la distribution ne contient pas de valeurs aberrantes. Pour pallier ce problème, nous proposons une extension du MoN basée sur le coefficient de Gini afin d'exploiter le meilleur des deux estimateurs lors du calcul. Cette approche est

simple à mettre en œuvre quel que soit l'intégrateur et ne nécessite pas de paramétrage complexe. Enfin, elle présente un surcoût de calcul réduit et conduit à la disparition des *fireflies*.

6.1 Problème d'apparition de *firefly*

L'équation de rendu (KAJIYA 1986) ne peut pas être résolue analytiquement dans la plupart des cas et comme précisé dans le chapitre 1, les approches de Monte-Carlo (MC) sont généralement utilisées pour estimer la valeur des pixels de l'image finale.

6.1.1 Rappel d'échantillonnage de MC

Un échantillonnage est effectué par la construction de chemins lumineux aléatoires entre la caméra et les sources lumineuses situées dans la scène 3D : dans sa forme la plus simple, un rayon est envoyé depuis l'emplacement de la caméra à travers un pixel et est réfléchi de manière aléatoire par la surface du premier objet qu'il rencontre. Le processus est appliqué récursivement jusqu'à ce qu'une source soit rencontrée ou que la construction du chemin soit arrêtée aléatoirement par l'utilisation d'une roulette russe. De nombreux chemins lumineux sont construits par pixel selon la loi des grands nombres et l'approche de MC. La moyenne de la contribution des échantillons est ensuite calculée pour chaque pixel. Elle converge vers la solution attendue suivant un taux de $1/\sqrt{n}$ où n est le nombre d'échantillons (SHIRLEY, C. WANG et al. 1996). Le calcul d'un très grand nombre d'échantillons par pixel permet d'obtenir des images synthétiques réalistes.

L'approximation finale de l'estimateur de MC de la valeur attendue pour n échantillons est obtenue à partir de la moyenne empirique comme spécifié dans l'équation 6.1.

$$\bar{\theta} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{6.1}$$

où x_i est la valeur d'un échantillon obtenue lors du rendu.

6.1.2 Visualisation du problème

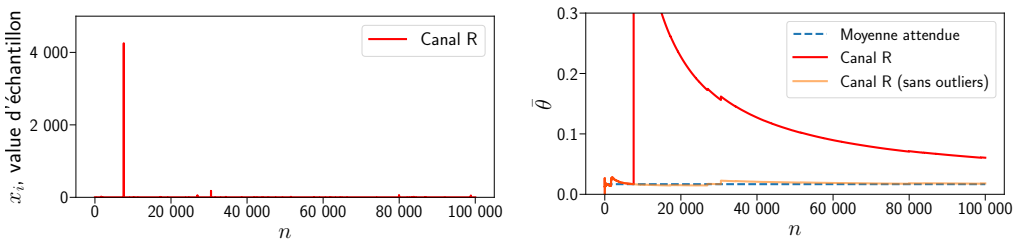
Ce calcul provoque initialement un bruit considérable lors de la génération de l'image, mais au fur et à mesure que le calcul progresse, ce bruit est réduit et devient imperceptible. Cependant, lors de la génération de telles images à partir de certaines scènes, certains artefacts visuels connus sous le nom de *fireflies* peuvent être encore présents et hautement perceptibles par le système visuel humain.

(a) PT (b) PT (c) BDPT (d) BDPT

FIGURE 6.1 – Aperçu des *fireflies* d'une partie recadrée de la scène de Veach utilisant le tracé de chemin classique (PT) et le tracé de chemins bidirectionel (BDPT) avec des échantillons de 100 000 dans les deux cas.

Des exemples de tels *fireflies* sont donnés dans la figure 6.1 où deux images ont été générées à partir de la scène de Veach (VEACH et L. GUIBAS 1995) en utilisant le tracé de chemins classique (PT) et le tracé de chemins bidirectionel (BDPT) (LAFORTUNE et al. 1993), ce dernier permettant de converger plus rapidement vers l'image finale qu'une méthode classique de tracé de chemins pour certaines scènes. On remarque rapidement que de nombreux *fireflies* sont visibles lorsqu'on utilise la méthode de tracé de chemin aléatoire. Bien que la méthode BDPT semble donner un meilleur résultat, certains *fireflies* sont toujours présents même avec 100 000 échantillons par pixel.

6.1.3 Origine des apparitions de *fireflies*



(a) Valeur de luminance des différents échantillons calculés, avec apparitions de plusieurs valeurs à forte contribution.

(b) Comparaison de la moyenne sans et avec valeur aberrante sur les valeurs de luminance du spectre rouge.

FIGURE 6.2 – Aperçu de l'impact de la contribution élevée des échantillons obtenus sur $\bar{\theta}$, l'estimateur de la moyenne arithmétique. L'axe des y de la sous-figure 6.2b a été redimensionné pour permettre une meilleure visualisation de l'impact des valeurs de haute contribution et la moyenne attendue est calculée sur 10 millions d'échantillons. Seul le canal de couleur rouge est affiché, mais ce type de comportement est similaire sur les autres canaux.

Ces artefacts visuels nommés *fireflies* proviennent principalement de chemins de très faible probabilité qui contribuent intensément à l'équation 6.1. Même si la moyenne est considérée comme un bon estimateur, elle est aussi fortement sensible par ce type de valeurs très importantes et leurs contributions ne peuvent être lissés qu'en évaluant de nombreux autres échantillons (voir figure 6.2). Pendant le rendu d'un pixel, il est difficile d'autre part de décider si la contribution d'un chemin est une valeur si rare qu'elle pourrait générer un *firefly* ou la première occurrence d'une estimation importante pour le pixel.

6.2 Travaux relatifs

Les valeurs d'échantillons qui génèrent des *fireflies* sont connues sous le nom de « valeurs aberrantes » dans des problèmes scientifiques plus généraux. Nous passons ici en revue les approches les plus classiques qui ont été étudiées en statistique pour tenir compte de ces valeurs aberrantes ainsi que les approches proposées en informatique graphique.

6.2.1 Approches statistiques pour la suppression des valeurs aberrantes

De nombreux indicateurs sont disponibles afin de bien estimer la valeur attendue en statistique, où la partie *moyenne filtrée* fait référence à des méthodes où les valeurs aberrantes sont supprimées (HUBER et al. 2009) par un processus d'écrêtage. Parmi ces méthodes, on peut en citer quelques-unes, telles que l'écrêtage *Percentile*, *Sigma* et *Médian*, qui visent à supprimer certaines valeurs d'échantillon en dehors d'un intervalle de confiance. Ceci nécessite toutefois de connaître l'ensemble des informations de la distribution des échantillons. Du point de vue du rendu d'images de synthèse, il est souvent difficile ou peu souhaitable de disposer de tous les échantillons et de leur distribution, en raison du coût mémoire élevé que cela requiert et de l'évaluation progressive des échantillons au cours du processus de rendu.

La médiane des moyennes (MoN) a été introduite par (BLAIR 1985 ; JERRUM et al. 1986) et conçue pour résister à l'apparition de données dites polluantes qui sont traitées comme des valeurs aberrantes. Elle consiste à séparer tous les échantillons en M ensembles de même taille (si possible). La moyenne est calculée pour chacun des M ensembles, puis la médiane sur les M ensembles est utilisée comme estimateur final. Cet estimateur semble faire preuve d'une certaine robustesse face à des valeurs qui peuvent se situer en dehors d'un intervalle de confiance (ALON et al. 1999). Plus récemment, des travaux intensifs ont été menés sur l'estimateur MoN qui ont démontré sa robustesse par

rapport à la moyenne pour les distributions non normales telles que les « heavy tail » [1] (BROWNLEES et al. 2015 ; BUBECK et al. 2013 ; D. HSU et al. 2016). Il est maintenant aussi largement étudié dans les approches orientées vers l'apprentissage automatique (BÜHLMANN 2003 ; LUGOSI et al. 2019), comme le *bagging* et le *bootstrapping*.

L'idée de combiner la robustesse de la médiane avec la cohérence de la moyenne a également été largement étudiée. Dans (CHAN et al. 1994) et (DAMILANO et al. 2004) les auteurs étudient l'utilisation de combinaisons linéaires de la moyenne et de la médiane avec des poids choisis selon des critères asymptotiques.

6.2.2 Suppression des *fireflies*

Plusieurs travaux visant à réduire l'impact des *fireflies* ont déjà été mentionnés dans le chapitre 3. Dans (DECORO et al. 2010), une approche basée sur la densité est proposée afin de réduire les *fireflies* en les identifiant et en les supprimant. Ils proposent de travailler sur un espace conjoint des coordonnées de l'image et des valeurs de l'échantillon (espace couleur). Ensuite, ils utilisent un arbre k-d pour stocker itérativement les échantillons qui sont susceptibles d'être aberrants en les comparant aux k échantillons voisins les plus proches. Bien que tous les échantillons ne soient pas stockés dans l'arbre k-d (uniquement les échantillons potentiellement aberrants), cela implique tout de même un coût de stockage supplémentaire, mais qui reste linéaire avec k, et approximativement logarithmique avec le nombre total d'échantillons. La méthode proposée dans (SEN et DARABI 2012) vise à exploiter les informations des échantillons des pixels voisins pour accélérer le rendu de l'image tout en supprimant la plupart de ces *fireflies*.

Dans (JUNG et al. 2015), les auteurs ont introduit l'extraction de la médiane sur les estimateurs de M moyennes sous-échantillonnées qui semble être identique à l'estimateur MoN. Ils proposent un critère pour obtenir la valeur de M à utiliser, en évaluant le rapport entre l'écart-type du pixel et l'écart-type de la scène. Si le pixel n'est pas susceptible d'être un *firefly* alors $M = 1$, conduisant à l'utilisation de l'estimateur moyen. Sinon, $M > 1$ (avec M une valeur impaire) avec une valeur maximale de $M = 17$. Les *fireflies* ont tendance à être tous supprimés avec cette approche lorsque leur nombre n'est pas trop important. Cependant, la valeur obtenue semble être sous-estimée. Dans (ZIRR et al. 2018), les auteurs précisent que la suppression des *fireflies* peut affecter la manière de converger vers la solution finale. Selon les auteurs, même si un échantillon peut être identifié comme un *firefly*, il ne doit pas être supprimé, mais pondéré par rapport à l'avancement du calcul et à la distribution. Ils proposent une approche théorique qui consiste à stocker tous les échantillons, à les trier, puis à pondérer chaque échantillon selon un critère de détection ou non d'une aberration. Le stockage de tous les échantillons

1. En théorie des probabilités, les distributions à queue lourde sont des distributions de probabilités dont les queues ne sont pas limitées de manière exponentielle, c'est-à-dire qu'elles ont des queues plus lourdes que la distribution exponentielle.

n'étant pas réalisable en pratique, ils proposent de stratifier la plage de luminance dans un ensemble de M tampons et de sommer la contribution de chaque échantillon dans un tampon associé à son intervalle de luminance. Ensuite, une fois trié, chaque tampon B_i se voit attribuer un poids de fiabilité, basé sur les informations locales et de voisinage des pixels, ce qui permet de réduire la contribution des aberrations.

Enfin, certaines méthodes proposent des approches de débruitage sans biais, c'est-à-dire garantissant que la solution obtenue converge correctement. Ces méthodes sont principalement orientées vers l'étude statistique des échantillons obtenus lors du rendu de l'image : (BOUGHIDA et al. 2017 ; ELEK et al. 2019 ; SEN et DARABI 2012). En apprenant comment les échantillons sont distribués, ces méthodes visent à réduire la variance et aussi à réduire l'apparition de *fireflies* lors du débruitage.

L'utilisation de techniques d'apprentissage profond a également été proposée permettant à la fois de débruiter considérablement les images et de réduire l'impact des *fireflies* (MUNKBERG et al. 2020 ; VICINI et al. 2019 ; VOGELS et al. 2018), à un coût de calcul faible, mais introduisant un biais.

6.2.3 *Fireflies* dans les intégrateurs basés sur le tracé de chemins

Le tracé de chemins a été amélioré par de nombreux autres intégrateurs, tels que les méthodes BDPT (LAFORTUNE et al. 1993) ou transport de lumière Metropolis (MLT) (VEACH et L. J. GUIBAS 1997), visant principalement à accélérer la convergence de la méthode. Cependant, toutes ces nouvelles méthodes présentent des *fireflies*, plus ou moins importants selon les scènes utilisées. Le MLT, par exemple, génère des mutations par un processus de Monte-Carlo par chaîne de Markov (MCMC) à partir d'un chemin initial dont la contribution est significative. Il peut donc être très sensible à la conservation de *firefly*, qui provient précisément d'une contribution plus élevée.

Des méthodes orientées vers le guidage des chemins ont également été proposées dans le but de contrôler les chemins qui sont échantillonnés par apprentissage par renforcement et d'obtenir une meilleure connaissance de la PDF pour les contributions intéressantes (VORBA et al. 2019). Les *fireflies* sont un aspect connu des approches pratiques de guidage de chemin même si des développements récents atténuent ce problème (DIOLATZIS et al. 2020 ; MÜLLER et al. 2017).

6.3 MoN dans le rendu d'images de synthèse

Le travail proposé par cette contribution porte sur la combinaison du coefficient de Gini (DORFMAN 1979) et de l'estimateur MoN afin de réduire l'apparition des *fireflies*. Avant de détailler les deux modes de combinaison que nous avons étudiés, nous décrivons

ici formellement l'estimateur MoN. Nous détaillons ensuite la manière dont il peut être utilisé dans le rendu et les résultats que l'on peut obtenir.

6.3.1 Définition

La MoN consiste à séparer tous les échantillons obtenus en M ensembles de même taille (si possible). La moyenne est calculée pour chacun des M ensembles et la médiane sur les M ensembles (l'ensemble médian) est utilisée comme estimateur final. Étant donné l'estimation d'une variable aléatoire indépendante et identiquement distribuée x_i, la médiane des moyennes avec M ensembles de taille k avec un total de $n = k \times M$ échantillons, peut être définie par l'équation suivante :

$$\hat{\mu}_{MoN} = \text{mediane}\left(\frac{1}{k}\sum_{i=1}^{k} x_i, ..., \frac{1}{k}\sum_{i=n-k+1}^{n} x_i\right) \tag{6.2}$$

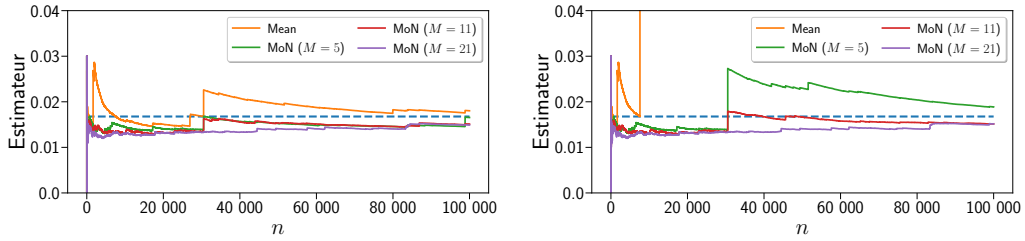
avec des ajustements mineurs si n/k n'est pas un nombre entier.

La mise en œuvre d'une telle approche dans un moteur de rendu est assez simple, mais implique un stockage d'informations M fois plus important que les méthodes de calcul de moyenne conventionnelles. Toutefois, cela reste raisonnable par rapport à la sauvegarde de tous les échantillons calculés. Ensuite, lors du rendu de l'image, les pixels sont échantillonnés un par un, la valeur de chaque échantillon étant ajoutée à l'un des ensembles, par exemple de manière cyclique. Enfin, la valeur finale du pixel est estimée par l'équation 6.2. L'utilisation du MoN lors du rendu est davantage détaillée par l'algorithme disponible en annexe B.1.

6.3.2 Étude du comportement de MoN

La figure 6.3 permet de comparer la robustesse de la méthode MoN à la moyenne arithmétique. Le nombre d'ensembles utilisés est $M = \{5, 11, 21\}$. La méthode MoN ne semble pas du tout affectée par la forte contribution des échantillons autour du 10 000ième échantillon pour chaque M étudié (voir figure 6.3b).

Aux alentours de 30 000 échantillons, certaines contributions, plus élevées que les précédentes, sont successivement collectées et génèrent une discontinuité dans la courbe de convergence. Notez que ces valeurs ne sont pas des valeurs aberrantes puisque leur magnitude est beaucoup plus petite que celle du 10 000ième échantillon. Chacune de ces contributions est stockée dans l'un des M ensembles, ce qui nécessite un nouveau tri des M moyennes pour le calcul de la médiane. Plus M est grand et moins l'ordre des moyennes sera impacté lors du tri pour connaître l'ensemble médian. Cet effet est visible pour $M = 5$ et $M = 11$ fournissant une légère surestimation pour $M = 5$ de l'ensemble médian. Le MoN utilisant des ensembles de taille $M = 21$ n'est pas impacté dans ce



(a) Échantillons de luminance du spectre rouge sans valeurs aberrantes.

(b) Échantillons de luminance du spectre rouge avec valeur aberrante (dans la même échelle).

FIGURE 6.3 – Comparaisons de MoN et de la moyenne sur des échantillons de rendu (canal rouge) avec et sans valeur aberrante. MoN est étudié avec $M \in \{5, 11, 21\}$ pour visualiser l'impact de ce paramètre. La valeur moyenne attendue a été calculée avec 10 millions d'échantillons.

cas puisque le nombre de contributions concernées est faible. Ces contributions sont donc triées dans les ensembles de rang supérieur et ne modifient pas l'ensemble médian. Dans ce cas, l'estimateur MoN sous-estime la valeur du pixel par rapport à la moyenne arithmétique.

La moyenne arithmétique est un bon estimateur, mais pas nécessairement robuste. La médiane, bien que plus robuste que la moyenne (HAMPEL 1971), implique une convergence plus longue vers la valeur attendue que celle-ci ; cela semble également être le cas lors de l'application de l'estimateur MoN, ce qui explique cette sous-estimation.

6.3.3 Choix dynamique du paramètre M

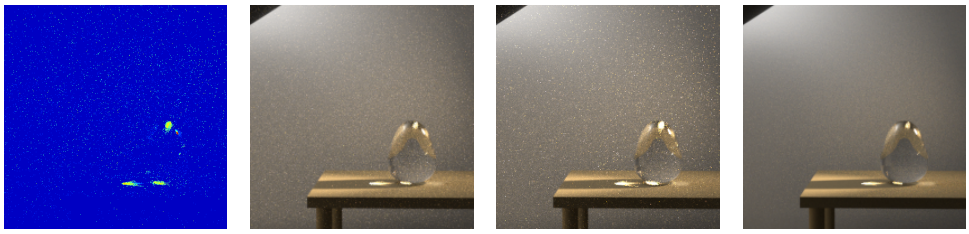
Dans (JUNG et al. 2015), qui a introduit le MoN, un choix dynamique de M a été proposé afin de réduire le coût de stockage supplémentaire et indirectement d'éviter cette sous-estimation. La valeur de M est calculée selon l'équation 6.3 :

$$M = \begin{cases} 1 & \text{pour } \frac{\sigma_{pixel}}{\sigma_{image}} < 1, \\ 2 \times \left(\lfloor \log_2 \frac{\sigma_{pixel}}{\sigma_{image}} \rfloor + 1 \right) + 1 & \text{sinon} \end{cases} \tag{6.3}$$

où σ_{pixel} est l'écart-type de la radiance sortante par un pixel et σ_{image} est l'écart-type dans l'image globale.

La figure 6.4 donne un aperçu des résultats obtenus à l'aide de l'équation 6.3. La valeur de M est représentée sous forme de carte thermique (voir la sous-figure 6.4a) sur une partie de la scène de Veach, avec des valeurs comprises entre 1 (bleu) et 17 (rouge). Elle illustre également que ce paramètre dynamique M tend à éliminer une grande partie

des *fireflies* à 10 000 d'échantillons (voir la sous-figure 6.4b). Cependant, il reste encore quelques *fireflies* et un bruit assez considérable reste visible. Au contraire, bien que l'estimation MoN avec $M = 17$ semble sous-estimer le résultat, elle élimine les *fireflies* et réduit le bruit de MC (voir la sous-figure 6.4d). D'autres résultats de la méthode de (JUNG et al. 2015) avec le choix dynamique de M sont disponibles dans la figure 6.5 avec les scores SSIM et rEQM mais également un meilleur aperçu des *fireflies* restants.



(a) Carte de chaleur du paramètre M (b) Jung et al. (c) Moyenne (d) MoN ($M = 17$)

FIGURE 6.4 – Vue d'ensemble de la carte dynamique du paramètre M de (JUNG et al. 2015) sur une partie recadrée de la scène de Veach avec les résultats visuels des estimateurs finaux de (JUNG et al. 2015) moyenne et MoN. La carte thermique est composée des valeurs possibles de M allant de 1 (bleu) à 17 (rouge) et les images sont calculées avec 10 000 échantillons par pixel.

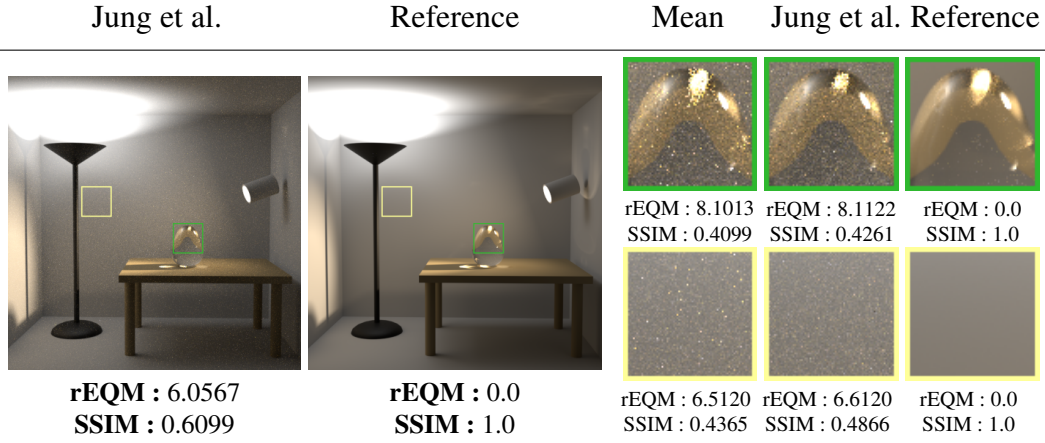


FIGURE 6.5 – Comparaison de la méthode classique Mean, (JUNG et al. 2015) avec le paramètre automatisée M et la référence en utilisant des images BDPT obtenues avec 10 000 échantillons par pixel.

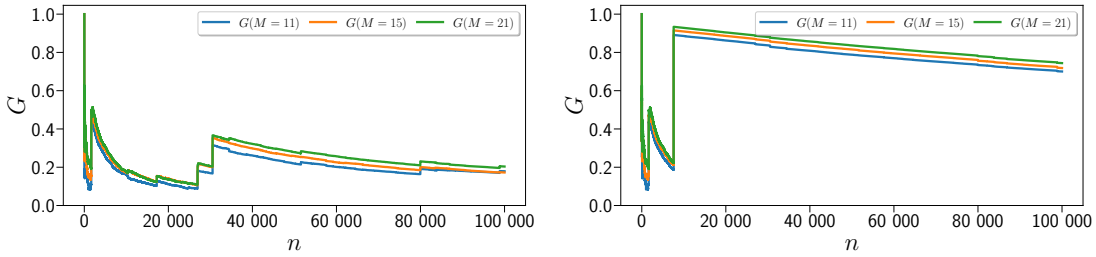
Le calcul de M selon l'équation 6.3 semble avoir une faiblesse qui est liée à la quantité de *fireflies* dans l'image : lorsqu'elle est élevée, elle augmente la variance

globale de l'image. Les *fireflies* provenant de pixels ayant une intensité plus faible ne sont donc pas supprimés.

6.4 MoN adaptatif avec le coefficient de Gini

Nous proposons d'améliorer l'estimateur MoN qui permet de réduire l'impact des valeurs aberrantes et donc l'apparition des *fireflies*. Nous décrirons deux approches qui se concentrent sur l'identification des *fireflies* localement (pour chaque pixel indépendamment). Elles sont toutes deux basées sur l'utilisation du coefficient de Gini, mais de manière différente.

6.4.1 Coefficient de Gini



(a) Évolution du coefficient G sur des échantillons sans valeurs aberrantes.

(b) Évolution du coefficient G sur des échantillons avec quelques valeurs aberrantes.

FIGURE 6.6 – Évolution du coefficient de Gini sur les moyennes M obtenues lors du rendu avec $M \in \{11, 15, 21\}$ pour visualiser l'impact des valeurs aberrantes sur G.

Le coefficient de Gini (DORFMAN 1979) est utilisé en économétrie pour mettre en évidence les inégalités sociales. Si la valeur obtenue à partir de ce coefficient est de 0, alors il y a une égalité parfaite et 1 (qui ne peut être atteint), signifie une inégalité totale. Nous nous concentrons sur ce coefficient afin de détecter la présence ou non d'un *firefly* avec l'idée que l'ajout de la valeur d'un *firefly* à l'une des moyennes MoN devrait augmenter l'inégalité entre ces moyennes.

Si l'on s'intéresse aux revenus d'une population par exemple, l'indice de Gini peut être défini comme la moitié de la différence moyenne relative de la série des revenus, c'est-à-dire :

$$G = \frac{\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left|x_i - x_j\right|}{2 \times \left(\frac{1}{n}\sum_{k=1}^{n} x_k\right)} \tag{6.4}$$

En pratique, on ne dispose pas de cette fonction, mais du revenu par « tranches » (ensembles) de la population. Ainsi, la formulation de ce coefficient fournie dans (DAMGAARD et WEINER 2000) et (DAMGAARD 2003) est définie par :

$$G = \frac{\sum_{i=1}^{n}(2i - n - 1)x_i}{n^2\mu} \tag{6.5}$$

avec $\mu = \frac{\sum_{i=1}^{n} x_i}{n}$ et les données ordonnées par taille croissante des valeurs ($x_i \leq x_{i+1}$).

Si l'on applique le coefficient de Gini à l'estimateur MoN, il peut être calculé par l'équation 6.6, basée sur une reformulation de l'équation proposée dans (DAMGAARD et WEINER 2000) lorsque les données sont regroupées en ensembles. La reformulation est détaillée dans l'annexe B.2.

$$G = \frac{2\sum_{j=1}^{M} j\hat{\theta}_j}{M\sum_{j=1}^{M}\hat{\theta}_j} - \frac{M+1}{M} \tag{6.6}$$

avec $\hat{\theta}_j$, $j \in [1, M]$, les M moyennes calculées avec le MoN, indexées par ordre croissant ($\hat{\theta}_j \leq \hat{\theta}_{j+1}$).

Nous montrons dans la figure 6.6 l'évolution du coefficient G pour les mêmes distributions d'échantillons qui apparaissent dans la figure 6.3, avec un *firefly* qui apparaît autour du 10 000ième échantillon, supprimé dans la sous-figure 6.6a. Cette sous-figure indique clairement que G semble faire confiance aux moyennes obtenues, car les grandes valeurs aberrantes ne sont pas présentes ici. Si une valeur d'échantillon légèrement plus grande apparaît (par exemple autour de $n = 30\ 000$), le coefficient fluctue modérément mais précise toujours que la distribution des moyennes n'est pas inégale, puisque $G \leq 0.4$. Au contraire, dans la sous-figure 6.6b, le coefficient G réagit fortement lorsqu'il rencontre la valeur aberrante autour de $n = 10\ 000$, atteignant une valeur proche de 1, signifiant l'apparition d'une inégalité entre les moyennes. Même avec 100 000 échantillons, la diminution de la valeur n'est pas suffisante pour lever le manque de confiance dans l'égalité des M moyennes obtenues. Notons que plus la valeur de M est grande, plus il semble difficile d'atteindre l'égalité puisque l'inégalité par rapport aux autres ensembles est courante et influence le coefficient G sur différentes distributions.

L'un des avantages de l'utilisation du coefficient de Gini est qu'il s'ajuste aux variations des M moyennes. Dans le contexte de notre approche, cela signifie qu'une valeur identifiée comme aberrante a été détectée, provoquant une forte inégalité dans les ensembles M. Si, toutefois, d'autres ensembles reçoivent également des contributions de même ampleur, la valeur du coefficient de Gini diminuera, en faisant à nouveau confiance à l'égalité des différentes moyennes et en considérant que ces valeurs sont toutes aussi significatives.

6.4.2 Choix binaire de l'estimateur

En raison du comportement de G en fonction de la présence ou de l'absence de *fireflies*, nous proposons une première approche qui consiste à choisir l'estimateur à un instant t du processus de rendu, en fonction de la valeur du coefficient de Gini. Nous définissons ainsi un seuil sur G en dessous duquel nous considérons qu'une confiance totale peut être accordée à l'estimation fournie par la moyenne classique. Par exemple, une valeur de $G \leq 0,25$ correspondra à un cas où l'indicateur de confiance pour la moyenne est sans risque (sans valeur aberrante et donc sans *firefly*). Dans le cas contraire, l'estimateur utilisé sera l'estimateur MoN. Ce seuil est ici défini expérimentalement après visualisation des valeurs de G.

Nous définissons donc un nouvel estimateur basé sur ce critère de confiance, appelé G-MoN$_b$, défini par :

$$G\text{-MoN}_b = \begin{cases} \bar{\theta} & \text{si } G \leq 0.25, \\ \hat{\mu}_{MoN} & \text{sinon} \end{cases} \tag{6.7}$$

Cet estimateur va exploiter le coefficient G calculé en chaque pixel afin de définir s'il doit être orienté vers la moyenne ou vers l'estimateur MoN. Il est noté G-MoN$_b$, pour binaire, car il effectue un choix purement binaire (moyenne ou MoN).

6.4.3 MoN adaptatif

Comme mentionné dans (ORENSTEIN 2019), en dépit de ses propriétés théoriques, l'estimateur MoN semble sous-exploiter les données disponibles en utilisant uniquement l'ensemble médian. Sur la base de cette idée, nous proposons une seconde approche qui vise à tirer profit des informations disponibles dans les ensembles voisins de l'ensemble médian. Elle sera définie par l'équation suivante :

$$\hat{\mu}_{G\text{-}MoN} = \frac{\sum_{j=1+c}^{M-c} \hat{\theta}_j}{M - 2c} \tag{6.8}$$

avec $\hat{\theta}_j$, $j \in [1, M]$, les M moyennes calculées avec le MoN, indexées par ordre croissant ($\hat{\theta}_j \leq \hat{\theta}_{j+1}$) et $c \in \mathbb{N}^{[0, \lfloor \frac{M}{2} \rfloor]}$, le nombre de moyennes de chaque côté extrême des M moyennes ordonnées qui ne seront pas prises en compte pour l'estimateur final. La figure 6.7 propose une illustration du fonctionnement de l'équation quand $c = 1$. Les moyennes représentées en rouge sont relativement à la valeur du paramètre c et les moyennes en vert sont conservées pour obtention de l'estimateur.

En raison de la capacité du coefficient de Gini à fournir certaines informations sur l'égalité/inégalité entre un ensemble d'informations, nous l'utilisons pour calculer dynamiquement la valeur du paramètre c introduit dans l'équation 6.8 : $c = \lfloor G \times k \rfloor$

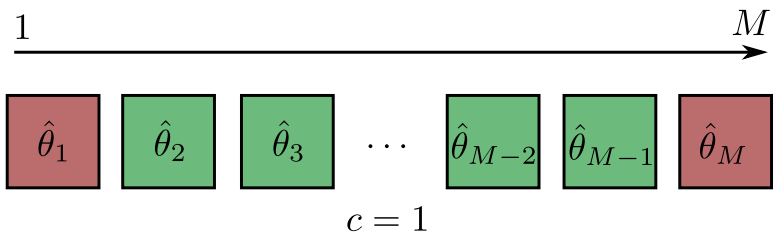


FIGURE 6.7 – Illustration de l'équation 6.8 pour $c = 1$, où les deux moyennes $\hat{\theta}_1$ et $\hat{\theta}_M$ sont exclues.

avec $k = \lfloor \frac{M}{2} \rfloor$. Ainsi, si G nous donne une confiance forte en la non-présence de valeur aberrante, alors l'estimateur final sera proche de la moyenne en utilisant plus de moyennes voisines. Sinon, il sera proche de l'estimateur MoN en utilisant uniquement l'ensemble médian, mais avec potentiellement quelques informations supplémentaires provenant des ensembles voisins.

6.5 Comparaisons et résultats

Nous présentons les résultats de ces deux estimateurs pendant le rendu pour différentes images et nous les comparons tout d'abord à trois autres estimateurs locaux (utilisant uniquement les informations du pixel courant) : la moyenne, la MoN et la version de l'estimateur MoN de (JUNG et al. 2015).

6.5.1 Configuration d'expérimentation

Nos expériences ont été menées en utilisant le moteur de rendu basé sur la physique PBRT (version 4) (PHARR et al. 2016) avec le support du GPU et les scènes de la littérature disponibles (BITTERLI 2016). Dans le contexte des images de synthèse, 100 000 échantillons par pixel reste un nombre conséquent d'échantillons, mais ne semble pas pouvoir effacer la présence de *fireflies* pour chaque scène. Afin de vérifier quels estimateurs semblent être plus intéressants que les autres lors du rendu, des images de référence de 4 scènes ont été calculées et sont disponibles dans la figure 6.8. Les scènes *Veach* et *Villa* dont les références sont disponibles dans les sous-figures 6.8a et 6.8d sont très sensibles aux *fireflies*, contrairement aux scènes *Bathroom* et *Crown* dans les sous-figures 6.8b et 6.8c dont le calcul ne génère aucun *firefly*. L'intérêt est de bien comparer la robustesse de chaque estimateur en fonction de la nature de la scène. Pour pouvoir obtenir une référence de qualité malgré la présence de *fireflies*, nous avons souhaité utiliser le BDPT plutôt que le tracé de chemins classique, car, outre une convergence généralement plus rapide, il a été noté expérimentalement que l'apparition de *fireflies* y est plus rare. Le temps de calcul avec l'intégrateur BDPT (voir section 1.3)

est plus long, car il nécessite de connecter des chemins lumineux émis de la caméra et d'une source de lumière de la scène. Il est cependant plus difficile à porter sur GPU, car plus difficilement parallélisable que le tracé de chemins. C'est pourquoi, un nombre de 100 000 échantillons a été utilisé pour générer les images de référence utilisant le BDPT, contrairement à 1 million pour le tracé de chemins, pour les images qui ne contiennent pas de *fireflies*.

(a) Veach (b) Bathroom

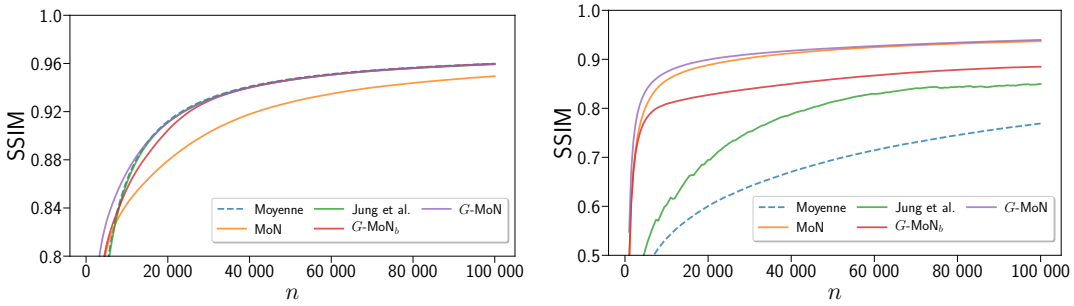
(c) Crown (d) Villa

FIGURE 6.8 – Images de référence utilisées pour les comparaisons des estimateurs : *Veach* et *Villa* sont calculées avec 100 000 échantillons par pixel en utilisant BDPT, *Bathroom* et *Crown* avec 1 million d'échantillons en utilisant le tracé de chemins classique.

Lors de la comparaison des estimateurs basés sur le MoN sur différentes valeurs de M, comme le MoN classique, le G-MoN$_b$ et le G-MoN, la méthode de (JUNG et al. 2015) est exécutée en utilisant son paramètre M dynamique avec $M = 17$ recommandé comme nombre maximum possible d'ensembles. La moyenne reste fixe (non affectée par le choix du paramètre M).

6.5.2 Étude de convergence

Une étude de convergence des différents estimateurs proposés (i.e. G-MoN et G-MoN$_b$) est traitée en comparaison avec les estimateurs de (JUNG et al. 2015) (avec M automatisé), le MoN et la moyenne. Les résultats sont disponibles dans la figure 6.9 en utilisant la mesure de l'indice de similarité structurelle (SSIM), davantage corrélée avec le système visuel humain (HVS), basée sur l'idée que les pixels ont de fortes interdépendances surtout lorsqu'ils sont proches spatialement. La racine carrée de l'erreur quadratique moyenne classique (rEQM) n'a été indiquée que parce qu'elle est peu



(a) Comparaisons de cinq estimateurs utilisant SSIM pendant le rendu sur l'image de la scène *Bathroom*.

(b) Comparaisons de 5 estimateurs utilisant SSIM pendant le rendu de l'image de la scène de *Veach*.

FIGURE 6.9 – Étude de convergence des estimateurs basés sur G avec $M = 21$ en utilisant l'indicateur SSIM jusqu'à 100 000 échantillons par pixel. Les estimateurs sont comparés à ceux de (JUNG et al. 2015), au MoN classique (avec $M = 21$) et à la moyenne sur deux scènes en utilisant les images de référence disponibles dans la figure 6.8 : *Bathroom*, sans *firefly* (à gauche) et *Veach* avec une grande quantité de *fireflies* (à droite).

sensible aux *fireflies* : elle calcule une erreur absolue, alors qu'un *firefly* génère une erreur très locale par rapport à l'image de référence.

La sous-figure 6.9a illustre les résultats pour la scène *Bathroom* qui ne présente pas de *firefly*. Tous les estimateurs ont un taux de convergence très proche, avec un taux légèrement inférieur pour MoN qui est connu pour sous-estimer ses résultats. Aucun des estimateurs n'est donc pénalisé par l'absence de *fireflies*. Lorsque les images mettent en évidence des *fireflies*, comme dans la scène de *Veach*, la différence entre les différents estimateurs devient plus importante (voir la sous-figure 6.9b). La moyenne ainsi que la méthode de (JUNG et al. 2015) échouent en raison de la présence d'un grand nombre de *fireflies* qui sont perçus par SSIM. La façon dont le paramètre M est calculé dans l'approche de (JUNG et al. 2015) est sensible à de telles situations et réduit son efficacité. Les estimateurs G-MoN et MoN fournissent les meilleurs résultats en matière de suppression des *fireflies*, avec un léger avantage à G-MoN. Même si les deux estimateurs ont finalement une convergence similaire lorsque les *fireflies* sont visibles, G-MoN est plus robuste puisque fournissant de meilleurs résultats dans les images sans *fireflies*. G-MoN$_b$ montre une faiblesse en matière de robustesse, son score SSIM convergeant entre ceux de la moyenne et de l'estimateur MoN, mais restant moins bon que ceux obtenus par MoN ou G-MoN.

Pour compléter les résultats de l'étude de convergence, le tableau 6.1 présente les scores SSIM obtenus par les estimateurs G-MoN, G-MoN$_b$ et MoN pour $M = \{5, 11, 15, 21, 25\}$ pour les 4 scènes calculées avec 100 000 échantillons et Path Tracing.

Ces estimateurs sont comparés à (JUNG et al. 2015) et à la moyenne dont les valeurs de score SSIM sont indiquées à nouveau pour chaque valeur de M (la moyenne ne dépend pas de M et nous avons utilisé une valeur maximale constante de $M = 17$ pour (JUNG et al. 2015)). En étudiant le classement proposé basé sur les scores SSIM, G-MoN semble être meilleur à chaque fois sur au moins 3 scènes lorsque $M > 11$. La seule scène où il semble être un peu plus éloigné est *Bathroom*, mais où son score SSIM reste très bon ce qui confirme qu'il s'agit d'un estimateur cohérent et fiable.

Scène		Veach	Bathroom	Crown	Villa
$M = 5$	Mean	0.77061 (5)	0.95987 (2)	0.99377 (2)	0.86733 (5)
	MoN	**0.88315** (1)	0.95275 (5)	0.99296 (4)	0.89738 (2)
	Jung et al.	0.84992 (3)	0.95932 (3)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.84036 (4)	**0.95988** (1)	**0.99378** (1)	0.89493 (3)
	G-MoN	0.88017 (2)	0.95785 (4)	0.99363 (3)	**0.89858** (1)
$M = 11$	Mean	0.77061 (5)	0.95987 (2)	0.99377 (3)	0.86733 (5)
	MoN	0.91602 (2)	0.95074 (5)	0.99280 (4)	**0.90833** (1)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.87084 (3)	**0.95993** (1)	0.99378 (2)	0.90461 (2)
	G-MoN	**0.92090** (1)	0.95935 (3)	**0.99380** (1)	0.89744 (3)
$M = 15$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.92697 (2)	0.95003 (5)	0.99279 (4)	0.91249 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.87818 (3)	0.95979 (2)	0.99377 (3)	0.90814 (3)
	G-MoN	**0.93097** (1)	0.95950 (3)	**0.99383** (1)	**0.91329** (1)
$M = 21$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.93711 (2)	0.94953 (5)	0.99273 (4)	0.91681 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.88508 (3)	0.95978 (2)	0.99374 (3)	0.91190 (3)
	G-MoN	**0.93976** (1)	0.95965 (3)	**0.99383** (1)	**0.91739** (1)
$M = 25$	Mean	0.77061 (5)	**0.95987** (1)	0.99377 (2)	0.86733 (5)
	MoN	0.94146 (2)	0.94915 (5)	0.99278 (4)	0.91894 (2)
	Jung et al.	0.84992 (4)	0.95932 (4)	0.99263 (5)	0.87744 (4)
	G-MoN$_b$	0.88769 (3)	0.95967 (2)	0.99377 (3)	0.91389 (3)
	G-MoN	**0.94359** (1)	0.95953 (3)	**0.99387** (1)	**0.91952** (1)

TABLEAU 6.1 – Comparaison SSIM avec l'image de référence et différentes valeurs de M en utilisant 100 000 échantillons par pixel avec le tracé de chemins. Le rang des estimateurs étudiés est indiqué par des valeurs de (1) à (5). Les estimateurs de la moyenne et de (JUNG et al. 2015) sont ajoutés pour chaque ligne différente de M à des fins de comparaison.

Des résultats supplémentaires sur la convergence de chaque estimateur sont également fournis dans le tableau 6.2. Le nombre d'échantillons requis est donné en fonction d'un score SSIM obtenu pour chacune des 4 scènes. Les scores SSIM proposés $\in \{0,6; 0,7; 0.8\}$. On peut rapidement remarquer que l'estimateur G-MoN atteint presque toujours en premier le score fixe SSIM en moins d'échantillons que tous les autres estimateurs, surtout lorsque $M > 5$. Une interprétation de ces résultats est qu'avec une valeur suffisamment grande de M, c'est-à-dire $M \in [11, 25]$, le coefficient de Gini semble capturer plus d'informations des M moyennes et permet une meilleure estimation finale.

	Scène	Veach			Bathroom			Crown			Villa		
	SSIM	0.6	0.7	0.8	0.6	0.7	0.8	0.6	0.7	0.8	0.6	0.7	0.8
$M=5$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (3)	**5021** (1)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	1518 (3)	**4256** (1)	**20353** (1)	729 (3)	2846 (5)	7501 (5)	239 (2)	302 (2)	383 (2)	343 (3)	469 (3)	1519 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (4)	5233 (3)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	1491 (2)	7515 (3)	37626 (3)	646 (2)	2180 (2)	5152 (2)	241 (3)	304 (3)	387 (3)	340 (2)	463 (2)	1422 (2)
	G-MoN	**1378** (1)	4300 (2)	20394 (2)	**627** (1)	**2163** (1)	5547 (4)	**233** (1)	**293** (1)	**370** (1)	**332** (1)	**448** (1)	**1368** (1)
$M=11$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (2)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	1060 (3)	2222 (2)	6835 (2)	619 (3)	1641 (3)	7066 (5)	240 (3)	303 (3)	384 (3)	364 (3)	503 (3)	1395 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (4)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	1031 (2)	2411 (3)	19398 (3)	607 (2)	1532 (2)	5198 (3)	239 (2)	302 (2)	383 (2)	363 (2)	500 (2)	1378 (2)
	G-MoN	**671** (1)	**1566** (1)	**5243** (1)	**508** (1)	**1254** (1)	**4537** (1)	**229** (1)	**288** (1)	**362** (1)	**344** (1)	**465** (1)	**1126** (1)
$M=15$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (3)	244 (4)	308 (4)	392 (4)	498 (4)	2507 (5)	14515 (5)
	MoN	986 (3)	1977 (2)	5224 (2)	649 (3)	1474 (3)	5636 (5)	242 (3)	305 (3)	387 (3)	392 (3)	551 (3)	1448 (3)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (4)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	983 (2)	2100 (3)	12702 (3)	642 (2)	1447 (2)	4734 (2)	240 (2)	303 (2)	386 (2)	390 (2)	549 (2)	1444 (2)
	G-MoN	**634** (1)	**1361** (1)	**3676** (1)	**520** (1)	**1061** (1)	**3706** (1)	**230** (1)	**288** (1)	**362** (1)	**366** (1)	**501** (1)	**1199** (1)
$M=21$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (4)	244 (3)	308 (3)	392 (3)	498 (4)	2507 (5)	14515 (5)
	MoN	941 (2)	1819 (2)	4377 (2)	719 (3)	1416 (2)	4179 (3)	245 (4)	309 (4)	393 (4)	442 (3)	652 (3)	1593 (2)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (5)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	964 (3)	1927 (3)	7142 (3)	716 (2)	1418 (3)	3961 (2)	243 (2)	307 (2)	391 (2)	441 (2)	650 (2)	1596 (3)
	G-MoN	**625** (1)	**1236** (1)	**2829** (1)	**567** (1)	**1017** (1)	**2766** (1)	**231** (1)	**290** (1)	**365** (1)	**411** (1)	**580** (1)	**1342** (1)
$M=25$	Mean	20007 (5)	52592 (5)	NR (5)	1074 (4)	2236 (4)	5021 (4)	244 (2)	308 (2)	392 (2)	498 (4)	2507 (5)	14515 (5)
	MoN	932 (2)	1780 (2)	4112 (2)	774 (3)	1422 (2)	3804 (3)	246 (4)	311 (4)	395 (4)	479 (3)	736 (3)	1727 (2)
	Jung et al.	9163 (4)	20644 (4)	44387 (4)	1261 (5)	2430 (5)	5233 (5)	265 (5)	337 (5)	435 (5)	525 (5)	2412 (4)	12310 (4)
	G-MoN$_b$	962 (3)	1886 (3)	6025 (3)	772 (2)	1428 (3)	3742 (2)	244 (2)	309 (3)	394 (3)	477 (2)	734 (2)	1732 (3)
	G-MoN	**631** (1)	**1199** (1)	**2626** (1)	**609** (1)	**1040** (1)	**2517** (1)	**232** (1)	**291** (1)	**366** (1)	**443** (1)	**644** (1)	**1446** (1)

TABLEAU 6.2 – Échantillons requis par pixel afin d'atteindre un score SSIM par rapport aux images de référence avec différentes valeurs de M. Le nombre maximum d'échantillons est fixé à 100 000 par pixel. *NR*, pour *Not Reached* est indiqué lorsque le score SSIM ne peut être atteint. La moyenne et (JUNG et al. 2015) sont ajoutés pour chaque ligne de M différente à des fins de comparaison.

6.5.3 Impact visuel des estimateurs

Nous comparons également visuellement les résultats obtenus pour les deux estimateurs par rapport à (JUNG et al. 2015), à la moyenne et à l'estimateur MoN classique. À cette fin, nous comparons les images calculées avec 10 000 échantillons par pixel aux images de référence. La figure 6.10 présente de telles comparaisons pour les deux images *Veach* et *Bathroom*. Deux zones ciblées sont indiquées par un cadran de couleur différente dans l'image complète.

Pour l'image de la scène de *Veach*, qui contient de nombreux *fireflies*, les approches basées sur l'estimateur MoN fournissent les meilleurs résultats visuels à l'exception de l'approche de (JUNG et al. 2015) dont le critère global conserve plusieurs *fireflies*. En terme de score SSIM, G-MoN semble le plus proche de la référence. L'ajout de la connaissance des ensembles voisins de l'ensemble médian apporte une meilleure précision d'estimation.

Pour l'image de la scène *Bathroom*, tous les estimateurs semblent être relativement proches de la référence, que ce soit visuellement ou en terme de score SSIM. Cependant, le MoN semble être plus précis dans les zones uniformes qui ne sont pas sensibles aux *fireflies* (voir la zone rouge sur la figure). Le G-MoN, quant à lui, propose un score fidèle à la référence sur des zones plus complexes ou très proches du meilleur estimateur (comme le MoN pour la zone encadrée en rouge). De plus, G-MoN semble converger

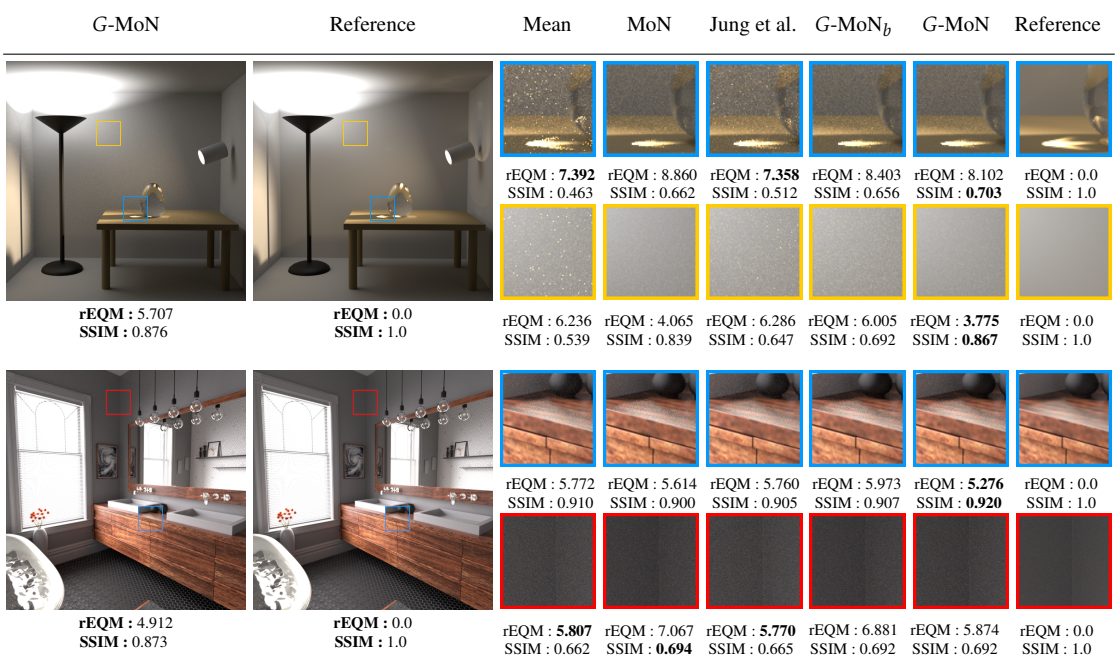


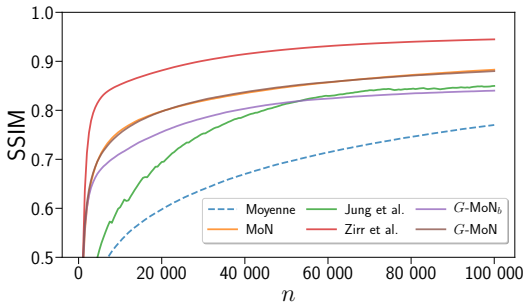
FIGURE 6.10 – Comparaisons de la rEQM et du SSIM obtenus à partir de différents estimateurs avec 10 000 échantillons par pixel sur 2 images. Les images en taille réelle et les zones ciblées sont comparées aux références. MoN, G-MoN$_b$ et G-MoN sont définis avec $M = 21$.

plus rapidement que la moyenne arithmétique. L'avantage de filtrer les valeurs éloignées de la moyenne en appliquant la médiane apporte de meilleurs résultats.

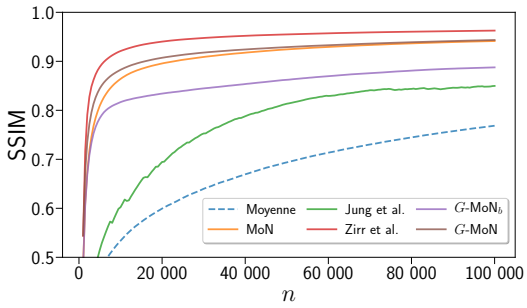
Toutes les expériences et les résultats obtenus semblent indiquer que l'estimateur G-MoN donne une bonne estimation. Il élimine les *fireflies* tout en étant proche ou supérieur à la moyenne classique pour les scènes sans *fireflies*. Le nombre de moyennes à exclure de l'ensemble des M moyennes, définit par c en utilisant le coefficient G semble être assez robuste. Le G-MoN$_b$, quant à lui, permet également la suppression des *fireflies*, mais n'est toujours pas aussi bon que le MoN, notamment sur la scène de *Veach*. Le G-MoN apporte la robustesse et la fidélité de l'estimateur souhaité. Plutôt que d'essayer d'automatiser le paramètre M, l'exploitation de la séparation des échantillons en ensembles permet d'identifier les *fireflies* et d'améliorer la fiabilité de l'estimateur final.

6.5.4 Comparaisons non locales

Nous comparons également notre approche à la méthode de (ZIRR et al. 2018) qui semble être globalement meilleure que la méthode G-MoN. Cette approche a également



(a) Comparaisons de Zirr et des estimateurs étudiés précédemment en utilisant SSIM et $M = 5$ sur la scène *Veach*.



(b) Comparaisons de Zirr et des estimateurs étudiés précédemment en utilisant SSIM et $M = 25$ sur la scène de *Veach*.

FIGURE 6.11 – Étude de convergence de la méthode de Zirr et des estimateurs étudiés précédemment avec $M \in \{5, 25\}$ en utilisant l'indicateur SSIM jusqu'à 100 000 échantillons par pixel. Les estimateurs sont comparés sur la scène de *Veach* avec une grande quantité de *fireflies*.

été implémentée dans PBRT à des fins de comparaison. Cette méthode utilise M tampons avec des intervalles de luminance ordonnés qui sont pondérés pour l'estimation finale. Le poids calculé est relatif à une fiabilité du pixel courant et de ses voisins (taille du noyau 3×3), ce qui le différencie des estimateurs précédemment comparés, où seules les informations locales à un pixel sont utilisées, même si l'approche reste très similaire. Les auteurs ont proposé un paramètre b permettant de définir la taille de chaque intervalle de luminance, où les valeurs associées à un tampon B_j sont dans l'intervalle $[b^{j-1}, b^{j+1}]$. Le paramètre b peut être défini automatiquement avec $b = \sqrt[M]{S_{max}}$ où S_{max} est la luminance maximale attendue fixée à 8^6 comme dans l'article de (ZIRR et al. 2018)

Les courbes de convergence présentées dans la figure 6.11, semblent indiquer que même avec une petite valeur de M (par exemple $M = 5$), l'estimateur de (ZIRR et al. 2018) converge plus rapidement. Pour $M = 25$, G-MoN et MoN semblent converger, mais sont toujours moins performants que la méthode de (ZIRR et al. 2018). Cela est principalement dû à l'utilisation d'informations voisines pour le calcul des poids d'échantillons (c'est-à-dire les tampons d'intervalle de luminance) qui n'ont pas été exploitées dans notre approche. La figure 6.12 illustre les images obtenues pour les méthodes G-MoN et (ZIRR et al. 2018) où l'on peut constater une proximité visuelle des résultats.

Les résultats sur la scène *Bathroom* où aucun *firefly* n'est présent sont également proposés dans la figure 6.13. On peut noter que dans les deux cas, $M = 5$ ou $M = 21$, l'approche de (ZIRR et al. 2018) est toujours légèrement meilleure que les extensions basées sur l'estimateur MoN.

FIGURE 6.12 – Comparaisons des méthodes G-MoN et (ZIRR et al. 2018) avec $M = 25$ après 100 000 échantillons par pixel sur la scène de *Veach*.

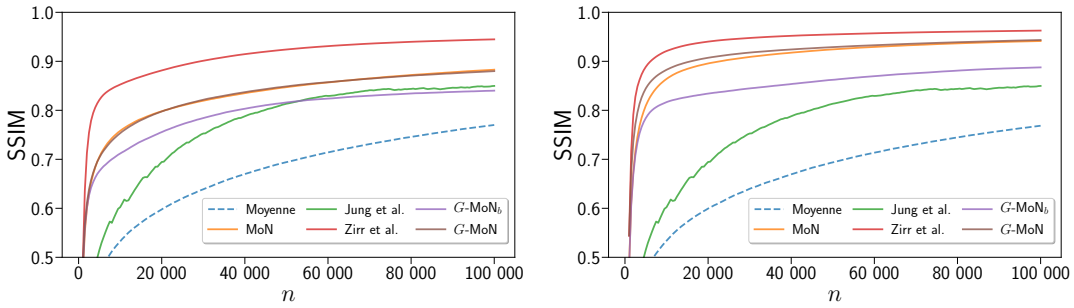
La complexité de la méthode de (ZIRR et al. 2018) basée sur une approche théorique de pondération lui permet d'être robuste aux *fireflies*, mais il faut noter que celle de G-MoN, même si elle est moins performante sur les petites tailles de M, reste compétitive et plus facile à mettre en œuvre du fait de sa simplicité.

6.5.5 Coût de calcul

La figure 6.14 fournit un aperçu du rendu de la scène de Villa pour le G-MoN et les estimateurs de moyenne. Les deux images ont été calculées sur le GPU en utilisant le tracé de chemins (PT) avec 100 000 échantillons par pixel. Visuellement, tous les *fireflies* sont supprimés en utilisant G-MoN. Les temps de calcul sont de $3618,3$ secondes et $3576,3$ secondes pour G-MoN (avec $M = 5$) et la moyenne classique respectivement. Ainsi, l'estimateur G-MoN n'entraîne qu'un coût supplémentaire d'environ $1,17\%$ pour la gestion des ensembles MoN et l'estimation de la valeur finale du pixel. Le temps de calcul pour (ZIRR et al. 2018) est quant à lui de $3590,6$ secondes ce qui est plus rapide que G-MoN.

6.5.6 Coût mémoire

Les intégrateurs basés sur MoN requièrent un rendu utilisant M ensembles différents, chacun représentant une moyenne. Cela implique un coût mémoire supplémentaire, qui reste cependant relativement faible. Chaque ensemble doit stocker une somme partielle et le nombre d'échantillons inclus dans la somme, soit 8 octets si les calculs sont effectués en simple précision. Par conséquent, si les pixels sont calculés de manière indépendante, seulement 8 fois M octets supplémentaires sont nécessaires pour chaque



(a) Comparaisons de Zirr et des estimateurs étudiés précédemment en utilisant SSIM et $M = 5$ sur la scène *Bathroom*.

(b) Comparaisons de Zirr et des estimateurs étudiés précédemment en utilisant SSIM et $M = 25$ sur la scène de *Bathroom*.

FIGURE 6.13 – Étude de convergence de la méthode de Zirr et des estimateurs étudiés précédemment avec $M \in \{5, 25\}$ en utilisant l'indicateur SSIM jusqu'à 100 000 échantillons par pixel. Les estimateurs sont comparés sur la scène de *Bathroom* avec aucun *firefly*.

canal de luminance. Lorsque le filtrage est utilisé, plusieurs pixels doivent être stockés simultanément. Cependant, le calcul parallèle fonctionne souvent sur la base d'un pixel à la fois et seule la taille M est requise pour affecter correctement les échantillons. En supposant des patchs de taille 32×32, trois canaux (R, G, B) et $M = 21$, cela nécessitera moins de 500 ko de mémoire supplémentaire par patch.

Notons cependant qu'en cas de rendu progressif, les tampons de tous les pixels doivent être sauvegardés afin de permettre à l'intégrateur de poursuivre le calcul. Cela revient donc à stocker M tampons d'image. Les tampons utilisés dans l'approche de Zirr doivent également être stockés, avec l'avantage que leur nombre est plus petit.

6.6 Discussion

Dans ce chapitre, nous avons introduit le coefficient de Gini afin d'étudier la présence de valeurs aberrantes localement pour chaque pixel dont la présence est susceptible de générer des artefacts visuels connus sous le nom de *fireflies*. Ensuite, deux nouveaux estimateurs ont été proposés en combinant l'estimateur MoN avec ce coefficient afin d'augmenter sa robustesse.

Ces estimateurs n'utilisent que les informations internes des pixels et leurs comparaisons avec des estimateurs similaires proposés précédemment ont montré que G-MoN était suffisamment robuste pour éliminer les *fireflies* assez rapidement. La mise en œuvre de G-MoN au sein de n'importe quel intégrateur est simple et son coût de traitement supplémentaire reste faible par rapport aux résultats proposés. La méthode est en outre

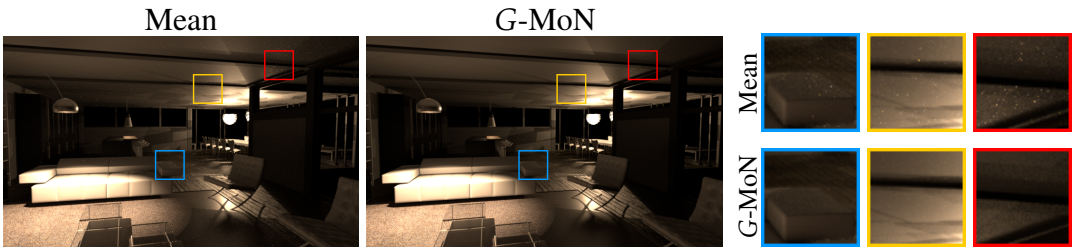


FIGURE 6.14 – Comparaison des estimateurs *G*-MoN et de la moyenne arithmétique sur une scène de Villa avec 100 000 d'échantillons par pixel en utilisant le tracé de chemins dans les deux cas. La moyenne, même à 100 000 échantillons est toujours sensible aux contributions élevées alors que cet estimateur élimine tous les *fireflies* présents dans l'image.

très simple à paralléliser, car elle ne nécessite aucune information globale, mais uniquement des informations locales provenant du pixel lui-même. La comparaison que nous avons faite avec la méthode de Zirr met en évidence que cette dernière fournit une meilleure convergence que la méthode proposée. Elle nécessite des informations de voisinage afin de calculer le poids des tampons avec plus de fiabilité. Lorsque M est assez grand, *G*-MoN semble être assez proche de la méthode de (ZIRR et al. 2018), mais nécessite un stockage mémoire plus important.

Résumé

Ce sixième chapitre présente une contribution réalisée qui traite le problème des *fireflies*, artefacts qui réduisent la qualité des images produites par les simulations d'éclairage. Il a été proposé un rappel du problème de l'estimateur de la moyenne lorsque des valeurs aberrantes apparaissent et l'étude de la médiane des moyennes (MoN) comme une alternative valable à cet estimateur classique. Toutefois, MoN a tendance à converger plus lentement que la moyenne bien qu'il soit plus robuste. Le coefficient de Gini est introduit, afin d'étudier la présence de valeurs aberrantes localement pour chaque pixel. À partir de cet indicateur statistique, deux nouveaux estimateurs sont proposés en le combinant avec le MoN afin d'augmenter la robustesse de l'estimateur MoN. L'un de ces deux estimateurs, le MoN adaptatif exploitant le coefficient de Gini, *G*-MoN, montre une convergence rapide de la solution et supprime les *fireflies*. Toutefois, il reste moins bon que l'estimateur proposé par (ZIRR et al. 2018) qui traite des informations des pixels voisins contrairement à *G*-MoN qui est purement local. Le coefficient de Gini pourrait être étendu à d'autres travaux comme l'extension de celui pour un estimateur non-local, son utilisation pour la détection de bruit, ou encore l'échantillonnage adaptatif.

Chapitre 7

Modèle de perception du bruit résiduel de Monte-Carlo

Introduction

Ce chapitre introduit une seconde contribution réalisée dans le cadre de la thèse, relative à la conception d'un modèle de perception de bruit résiduel de Monte-Carlo. Cette approche est basée sur l'apprentissage profond (*Deep Learning*) et offre une certaine robustesse lors de la prédiction de critères d'arrêt. Elle utilise une mesure de quantification locale du bruit comme attribut d'entrée et la quantité de données disponible de la nouvelle base d'images proposée permet de réaliser la conception d'un tel modèle. Les modèles d'apprentissage profond sont conçus pour traiter une quantité de données importante, ce qui permet de réaliser leur apprentissage en un temps plus abordable qu'un modèle basé sur les SVM. L'approche proposée met en avant la puissance émanant des méthodes d'apprentissage profond pour ce genre de tâche.

La nouvelle base d'images proposée possède une grande variété des types scènes, avec des géométries et des effets lumineux plus complexes. Cette diversité associée à un niveau de bruit précis enregistré entraîne une grande quantité de données. La section 7.1 met en avant le problème des SVM à faire face à des bases de données de taille importante et de grande dimension. La section 7.2 détaille une première approche basée sur l'apprentissage profond pour la tâche de classification d'une image de synthèse (encore bruitée ou non), plus précisément avec l'utilisation de réseaux de neurones récurrents pour pallier ce problème de dimensionnalité tout en exploitant au maximum les données acquises. Des caractéristiques locales pour chaque image issues de la *Singular Value Decomposition* (SVD) sont extraites pour permettre l'apprentissage du modèle. De plus, la notion de l'utilisation d'une fenêtre glissante appliquée aux images d'entrée est également introduite dans cette première contribution. En effet, les étapes intermédiaires d'images générées peuvent être stockées durant le processus de rendu des images de

synthèse, ce qui permet d'exploiter des niveaux de bruit différents en entrée d'un modèle d'apprentissage.

7.1 Problématique de dimensionnalité

Nous cherchons à identifier quel type de modèle d'apprentissage est le plus adapté pour réaliser la tâche de classification à partir de la nouvelle base d'images proposée dans le chapitre précédent. Les travaux précédents portant sur la détection de bruit dans une image de synthèse (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) et (TAKOUACHET et al. 2017) se basent sur un modèle de type SVM.

Les modèles SVM sont des modèles très robustes par le fait que l'hyperplan séparateur trouvé permet une très bonne séparabilité des données et qui de plus, grâce à la notion de marge appliquée à cet hyperplan, permettent davantage de généralisation. Toutefois, ils répondent généralement mieux sur une quantité raisonnable de données (nombre d'exemples inférieur à 100 000) ce qui peut parfois limiter leur utilisation. Ils sont donc de bons modèles quand le problème que l'on souhaite modéliser comporte une quantité faible ou peu conséquente de données. Si l'on souhaite utiliser une quantité de données de taille importante, que ce soit en dimension de la donnée elle-même ou en nombre, les SVM ont tendance à nécessiter un grand temps d'apprentissage.

7.1.1 Complexité d'apprentissage des SVM

Les SVM à noyau nécessitent le calcul d'une fonction de distance entre chaque échantillon du jeu de données. En conséquence, le coût est de l'ordre de $O(n_f \times n_o^2)$ où n_f représente le nombre de caractéristiques d'entrée d'une donnée et n_o, le nombre de données en entrée du modèle. Le stockage des distances entre chaque échantillon est une charge pour la mémoire et elles sont donc recalculées à la volée. Heureusement, seul le stockage des échantillons les plus proches de la frontière de décision (l'hyperplan) est nécessaire la plupart du temps. Les distances fréquemment calculées sont alors stockées dans un cache pour un accès rapide. Comme précisé dans (ABDIANSAH et al. 2015), pour la librairie *libSVM* (CHANG et al. 2011) souvent utilisée pour la conception de modèles SVM, la complexité de la fonction d'apprentissage est de l'ordre de $O(n^3)$ où n représente le nombre de données sans détail sur sa dimension.

7.1.2 Reproductibilité des travaux précédents

L'utilisation des modèles SVM n'a pas pu être envisagée pour la tâche de détection de bruit relativement à la quantité de données de la nouvelle base d'images. Ces modèles

sont moins adaptés sur un grand volume de données et ne nous ont pas permis de reproduire les précédents travaux pour comparaisons.

En effet, la base d'images précédemment exploitée par les travaux de (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) et (TAKOUACHET et al. 2017) est composée de 12 points de vue issus de 10 scènes. Les images de chaque point de vue sont calculées de 100 à 10 000 échantillons, avec un pas de 100 échantillons, soit 100 images de niveaux de bruit. Les images de taille 512×512 sont également découpées en 16 pour l'obtention de seuils. De ce fait, cette base d'images comporte $100 \times 16 \times 12 = 19\ 200$ données d'apprentissage. Au contraire, la nouvelle base de données d'images proposée dans le cadre de cette thèse comporte 500 niveaux de bruit sur 40 points de vue, ce qui permet d'obtenir $500 \times 16 \times 40 = 320\ 000$ données pour l'apprentissage.

La dimension de l'image était précédemment de 128×128 pour maintenant 200×200. Ce qui veut dire que le vecteur d'entrée pour l'approche de (J. CONSTANTIN, I. CONSTANTIN et al. 2016) serait de 40 000 au lieu de 16 384 précédemment et pour l'approche de (TAKOUACHET et al. 2017), de 80 000 (deux images en entrée) au lieu de 32 768.

Il est également important de préciser que lors de l'apprentissage d'un SVM, il est nécessaire de tester plusieurs jeux de paramètres. En effet, trouver le meilleur hyperplan séparateur requiert des paramètres γ, la manière de projeter les données dans une plus grande dimension et C la souplesse de la marge, qui sont optimaux (du moins proches de l'optimalité). À cela s'ajoute également un processus de validation croisée (section 3.1) des données pour diminuer le surapprentissage. Trouver les paramètres optimaux γ et C pour déterminer le meilleur hyperplan séparateur entraîne donc un grand nombre de jeux de paramètres à tester. Les meilleurs paramètres trouvés dans les approches précédentes ont été spécifiés, car testés, mais correspondent aux données de l'ancienne base d'apprentissage.

Du fait de cet accroissement important de la quantité de données à utiliser et des paramètres optimaux à obtenir, reproduire en l'état les approches réalisées par (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) et (TAKOUACHET et al. 2017) sur la nouvelle base d'images n'a pas pu être réalisé dans le cadre de cette thèse, du moins à partir de l'utilisation d'un modèle SVM.

7.2 Réseau de neurones récurrents et mesure de l'entropie appliquée à la SVD

La quantité de données d'apprentissage de la nouvelle base d'images permet d'envisager exploiter des méthodes d'apprentissage profond pour la conception du modèle de perception. Cette section vise à présenter la première contribution réalisée relative à la

création d'un modèle de perception de bruit de MC. Cibler des caractéristiques de bruit reste important pour apprendre correctement un tel modèle. Se pose cependant la question (classique) du choix des attributs représentatifs du bruit qui seront utilisés comme données d'apprentissage lors de l'élaboration de ce modèle. Dans l'approche qui est proposée, nous avons exploité la SVD et plus précisément la mesure de l'entropie (WEHRL 1978) associée à la décomposition obtenue, la notion de *SVD-Entropy* (BANERJEE et Nikhil R PAL 2014a) apparaissant comme un outil intéressant pour la quantification du bruit dans les images. Un réseau de neurones récurrents (RNN) sera introduit pour permettre un apprentissage à partir des caractéristiques *SVD-Entropy*, extraites sur une fenêtre glissante d'images pour réaliser la tâche de classification. Les images successives générées lors du rendu progressif seront sauvegardées au fur et à mesure dans une fenêtre dite glissante, qui va garder en mémoire les S dernières images. L'utilisation de d'une fenêtre glissante d'images a pour objectif de fournir plus d'informations au modèle ainsi qu'une stabilité des prédictions dans le temps.

7.2.1 Mesure de la *SVD-Entropy*

Nous cherchons d'abord à identifier une mesure de quantité de bruit présent dans une image. Cette mesure sera fournie par la suite à notre modèle d'apprentissage pour réaliser la tâche de classification.

L'entropie dans un cadre général permet d'extraire des informations dans un environnement bruité. Elle permet notamment de décrire l'ordre ou le désordre présent dans ces données. En particulier, l'entropie d'une image est une caractéristique statistique qui reflète l'information moyenne dans une image et peut être utilisée comme mesure de la qualité de l'image. Dans le cas de la SVD (G.H.GOLUB et al. 1983) du canal L d'une image (CARNEC et al. 2008), une mesure d'entropie peut être définie, à savoir la *SVD-entropy* (BANERJEE et Nikhil R PAL 2014a) qui permet de quantifier le bruit perceptuel inhérent aux images de synthèse.

7.2.1.1 Aperçu de la décomposition en valeurs singulières (SVD)

La méthode proposée est basée sur la SVD qui est une méthode de factorisation matricielle très puissante. Dans la théorie de la SVD (G.H.GOLUB et al. 1983 ; O.ALTER et al. 2000), toute matrice L de taille $M \times N$, avec $rang(L) = O$, où $O = min(M,N)$ (dans notre cas, L sera une matrice rectangulaire à valeur réelle construite à partir du canal L de l'image dans l'espace couleur CIE $L^*a^*b^*$), peut être décomposée de manière unique sous la forme :

$$L = U\Sigma V^T = \sum_{p=1}^{O} \sigma_p \overrightarrow{u_p}\,\overrightarrow{v_p} \tag{7.1}$$

où U et V sont des matrices orthogonales de tailles respectives $M \times M$ et $N \times N$ avec des vecteurs colonnes $\overrightarrow{u_p}$ (appelés les vecteurs singuliers de gauche) et $\overrightarrow{v_p}$ (appelés vecteurs singuliers de droite) et Σ est une matrice diagonale contenant les racines carrées des valeurs propres de LL^T ($\Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_O)$). Les éléments diagonaux de Σ peuvent être classés dans un ordre décroissant et sont appelés les valeurs singulières de L. La méthode SVD est une méthode fiable de décomposition des matrices. La SVD est construite à partir de deux sous-espaces dominants et sous-dominants orthogonaux. D'après (K.KONSTANTINIDES et al. 1997), il semblerait que les valeurs singulières (SV) obtenues en appliquant la décomposition sur une image en niveaux de gris, dépendent de la luminance de cette image tandis que la paire correspondante de vecteurs singuliers caractérise la géométrie présente dans l'image. Cette propriété intéressante est utilisée dans le filtrage du bruit, l'application de filigranes, la compression d'images, etc.

7.2.1.2 Caractéristiques du bruit basées sur la SVD

Concernant la quantification du bruit, certaines méthodes utilisent la décomposition en valeurs singulières (SVD). Elle est utilisée pour cibler les composantes SV les plus représentatives de l'image et permet d'obtenir une estimation du bruit de l'image. L'approche introduite dans (KONSTANTINIDES et al. 1997) est basée sur le fait que les plus petites valeurs du vecteur de valeurs singulières seraient représentatives du bruit, ce qui facilite sa détection et son filtrage. Elle est ensuite étendue à un découpage par blocs de l'image, appelé *Block based SVD* (BSVD), afin de faciliter les calculs de la décomposition SVD (SAE-BAE et al. 2007). L'idée est (i) d'appliquer un filtre gaussien à l'image bruitée, (ii) de soustraire l'image filtrée de l'image bruitée pour obtenir une image contenant les contours des objets et le bruit, (iii) d'appliquer la BSVD à cette dernière image pour supprimer le bruit, et enfin (iv) d'additionner l'image filtrée et l'image des contours sans bruit pour obtenir l'image originale sans bruit. Selon (W. LIU 2014 ; W. LIU et Weisi LIN 2012 ; Shuigen WANG et al. 2013), les valeurs singulières SVs de plus petit rang (de plus grand module) semblent porter une information plus importante sur la structure du contenu de l'image. De manière complémentaire, les composantes de rangs plus élevés du vecteur singulier seraient davantage liées aux détails présents dans l'image et donc potentiellement au bruit.

Des relations entre le bruit dans l'image et la décomposition SVD ont ainsi été mises en avant ; (H.C.ANDREWS et al. 1976) a montré que la reconstruction d'une image en utilisant uniquement les valeurs singulières de plus petit rang permet de débruiter l'image originale. Ainsi, par le biais des propriétés psycho-visuelles, on considère que la qualité visuelle de l'image n'est pas significativement réduite en ignorant les valeurs singulières de rangs inférieures (A.M.RUFAI et al. 2014) lors de la reconstruction de l'image.

D'après (K.KONSTANTINIDES et al. 1997), lorsque du bruit est ajouté à une image, les valeurs singulières sont modifiées de manière non uniforme : les valeurs moyennes

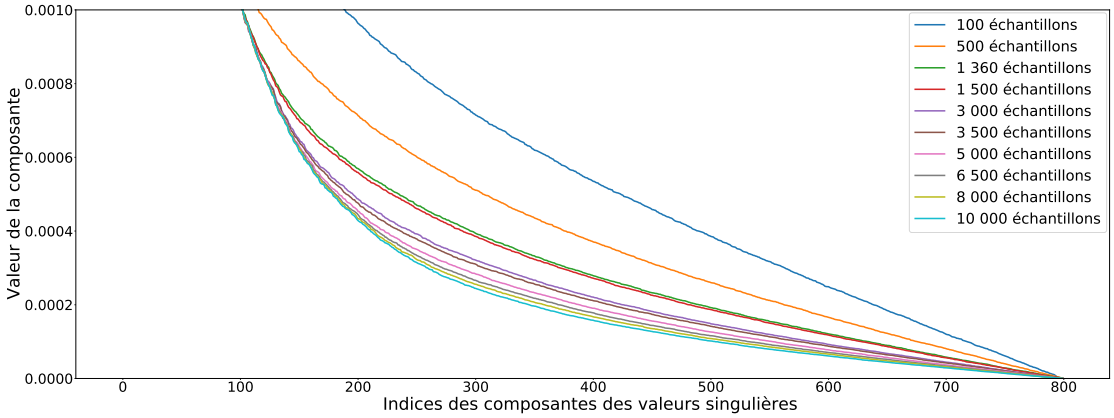


FIGURE 7.1 – Vue d'ensemble des composantes de Σ sur le point de vue de *Living room*, mises à l'échelle pour afficher les composantes les plus faibles.

SVs sont celles qui sont les plus impactées, bien que SVs de plus petite valeur subissent un grand changement relatif. Cette propriété est illustrée dans la figure 7.1, où les valeurs des 800 SVs normalisées obtenues de l'image *Living-room* (précédemment présentée dans le chapitre 5 dans la figure 5.5) sont comparées pour différents niveaux de bruits de MC.

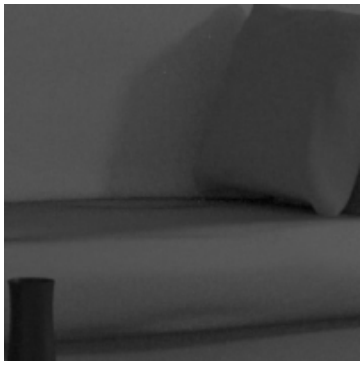
(W. LIU 2014) met en évidence que le premier quart des composantes du vecteur SV semble être lié à la structure de l'image et le reste des composantes au bruit présent dans l'image. La figure 7.2, illustre cette propriété après reconstruction des images en niveaux de gris du canal L de l'espace couleur CIE $L^*a^*b^*$. L'image reconstruite est assez proche de l'image originale lorsque seuls les premiers 25% des composantes sont utilisés.

7.2.1.3 Mesure de l'entropie appliquée à la SVD

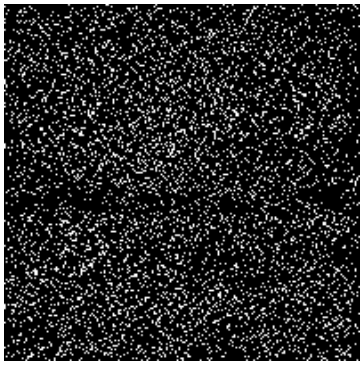
L'entropie est un concept scientifique qui est le plus souvent associée à un état de désordre, d'aléatoire ou d'incertitude. Le concept est appliqué à plusieurs domaines, de la thermodynamique classique, où il a été introduit pour la première fois, à la description microscopique de la nature en physique statistique, ainsi qu'en théorie de l'information, sous différentes formalisations. Elle a trouvé de nombreuses applications en chimie et en physique, dans les systèmes biologiques et leur relation à la vie, en cosmologie, en économie, en sociologie, en météorologie, en changement climatique et dans les systèmes d'information, y compris la transmission d'informations dans les télécommunications (WEHRL 1978).

Dans notre cas, nous nous basons sur l'entropie de Shannon (J. LIN 1991) principalement utilisée dans la théorie de l'information. Si l'on considère les valeurs singulières de l'image L, calculées sous forme de matrice (O est le rang de la matrice

(a) Canal L (500 échantillons)



(b) Reconstruction SVD avec les composantes 1 à 50 de SV



(c) Reconstruction SVD avec les composantes 51 à 200 de SV

FIGURE 7.2 – Reconstruction SVD d'une image d'un bloc de taille 200×200 de l'image *Living room* (a) en divisant le vecteur SV en deux parties, le premier quart (b) et les composantes restantes du vecteur SV (c).

L, $\Sigma = diag(\sigma_1, \sigma_2, ..., \sigma_i ..., \sigma_O)$), l'importance relative σ_i de la i^{e} SV par rapport à la fraction de l'expression globale qu'elle capture, est exprimée comme suit :

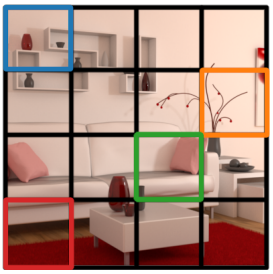
$$\overline{\sigma}_i = \sigma_i^2 / \sum_{p=1}^{O} \sigma_p^2 \tag{7.2}$$

L'entropie SVD (H_{SVD}) est définie par l'équation 7.3 en utilisant l'importance relative des composantes (équation (7.2)).

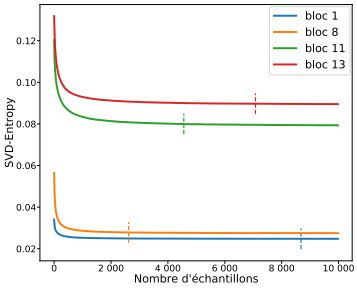
$$H_{SVD} = -\frac{1}{\log(O)} \sum_{i=1}^{O} \overline{\sigma}_i \log(\overline{\sigma}_i) \tag{7.3}$$

Elle mesure la complexité des données à partir de la distribution de l'expression globale entre les différentes valeurs du SV, où $H_{SVD} = 0$ correspond à un ensemble de données ordonné et redondant (toute l'expression est capturée par une seule composante du SV), et $H_{SVD} = 1$ correspond à un ensemble de données désordonné et aléatoire.

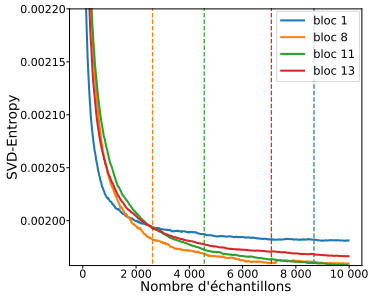
Dans notre problème de détection de bruit résiduel MC, il est difficile de quantifier le bruit lié à une image. H_{SVD} peut être un indicateur sur la composition du bruit encore présent dans une image de synthèse. Étant donné que lors du rendu d'une image le nombre d'échantillons augmente, la *SVD-Entropy* devrait permettre d'indiquer dans quelle mesure le bruit affecte encore l'image. Elle pourrait également être un indicateur de la stabilité de l'image au cours de sa génération (convergence). Ainsi, la *SVD-Entropy*



(a) Blocs sélectionnés pour comparaisons (identifiés par leurs couleurs).



(b) Évolution de H_{SVD} sur les 4 blocs sélectionnés.



(c) Évolution de H_{SVD} sur les 4 blocs sélectionnés (normalisés et remis à l'échelle).

FIGURE 7.3 – Aperçu de l'évolution de H_{SVD} pour l'image *Living room*. La sous-figure (a) met en avant 4 blocs ciblés où les blocs $1, 11, 13$ contiennent différents matériaux, textures et un éclairage principalement indirect. Le bloc 8, au contraire, reçoit un éclairage plus direct. Pour les sous-figures (b) et (c), les lignes pointillées verticales indiquent le seuil humain de perception de bruit pour le bloc qui lui est associé.

(O. ALTER et al. 2000) permet de collecter les caractéristiques de bruit efficaces lors de la génération d'images. La figure 7.3 montre l'évolution de H_{SVD} pour l'image *Living room* à différents niveaux d'échantillonnage.

7.2.2 La méthode proposée

Nous proposons une méthode en deux étapes basée sur la *SVD-entropy* et les réseaux neuronaux récurrents (RNN), dont le pipeline général est présenté dans la figure 7.5 et est détaillé par la suite. Cette méthode vise à détecter si certaines parties de l'image de synthèse en cours de génération est encore bruitée ou non.

7.2.2.1 Réseau de neurones récurrents

Les RNN sont une classe de réseaux neuronaux artificiels où les connexions entre les nœuds forment un graphe dirigé le long d'une séquence temporelle. Cela leur permet d'imiter un comportement temporel dynamique. Dérivés des réseaux neuronaux pleinement connectés, les RNN peuvent utiliser leur état interne (mémoire) pour traiter des séquences d'entrées de longueur variable. Ils sont aujourd'hui largement utilisés, car plusieurs types de tâches peuvent leur être appliqués une fois l'architecture du réseau bien définie (GREGOR et al. 2015 ; MIKOLOV et al. 2011 ; ZAREMBA et al. 2014). Par exemple, ils sont utilisés non seulement en traitement du langage naturel, pour la traduction, l'autocomplétion de phrases (prédiction du mot suivant), mais aussi pour la classification de textes (P. LIU et al. 2016) ou la génération de textes (SUTSKEVER et al.

2011). Selon la nature de l'information, ces tâches peuvent être de nature différente et ne pas être uniquement axées sur le traitement de texte. Les vidéos, en particulier, peuvent être considérées comme des suites d'images dont il est possible d'extraire des informations et de classer certains éléments (BACCOUCHE et al. 2010 ; EBRAHIMI KAHOU et al. 2015). Dans notre cas, nous cherchons à identifier dans cette suite d'images, si la différence de bruit entre ces images est notables ou non.

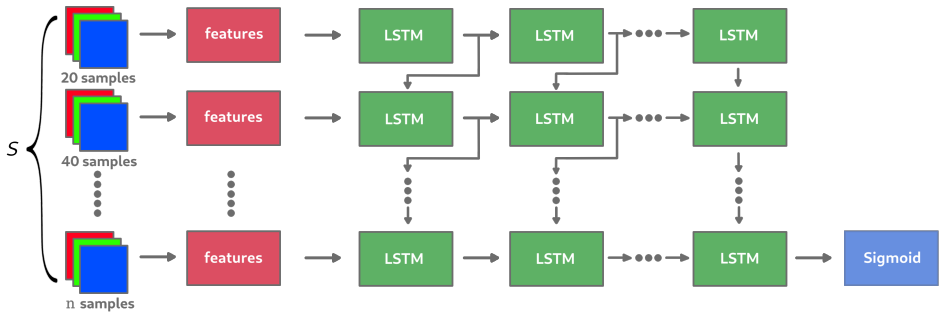


FIGURE 7.4 – Architecture prévue du RNN avec S comme taille de la fenêtre glissante. Le pas d'échantillonnage supplémentaire entre chaque image composant la fenêtre glissante est ici fixé à 20. Les images fournies au RNN dans cet exemple sont donc échantillonnées avec 20, 40, ... $(k-1) \times 20$ chemins.

Lors du rendu d'une image par un processus de Monte-Carlo, il est possible de considérer chaque étape de l'amélioration de l'image (ajout de n échantillons) comme une fenêtre glissante d'images (ou réservoir d'images) : la première image de la fenêtre glissante est la plus bruitée, obtenue lors des premières étapes du calcul ; la dernière image correspond à l'image la moins bruitée à un stade d'avancement du calcul. Ainsi, nous proposons de conserver les S dernières images calculées, chacune avec un nombre croissant d'échantillons, d'extraire des informations liées au bruit dans ces images, puis de fournir ces informations à un RNN afin de savoir si la dernière image de la fenêtre glissante est toujours bruitée. La cellule LSTM (*Long Short Term Memory*) proposée par (HOCHREITER et Jürgen SCHMIDHUBER 1997b) (présentée dans la section 3.1) est utilisée pour cette tâche dans le but d'utiliser au mieux les informations précédentes des images de la fenêtre glissante avant d'estimer l'étiquette. Le modèle serait donc appris dans un environnement dit d'apprentissage supervisé, sur la base des données recueillies lors des expériences réalisées. La figure 7.4 décrit l'architecture générale du réseau RNN où S images, avec un pas d'échantillonnage de $n = 20$ entre chacune d'entre elles, sont utilisées comme données d'entrée.

7.2.2.2 Paramètres du RNN

Notre modèle est composé de 3 couches LSTM avec les tailles de couches respectives suivantes : 512, 128, 32 et un taux de *dropout* (voir section 3.1.4.1) de 40% sur chacune d'entre elles afin d'éviter le surapprentissage. Fixer un pourcentage de *dropout* entre 40% et 50% est généralement proche de l'optimal comme indiqué dans cette thèse (SRIVASTAVA 2013). Après quelques essais, nous avons pu constater que fixer un *dropout* à 40% permettait de mieux répondre au compromis de surapparentissage et généralisation pour notre problème.

Chaque couche LSTM a une fonction d'activation *Sigmoid* pour l'état de la cellule et l'état caché. Une fonction d'activation *Hard Sigmoid* (fonction définie comme linéaire par partie) (SHARMA et al. 2017) est définie pour activer la porte entrée/oubli/sortie. Une couche de sortie de taille un est ensuite utilisée avec la fonction d'activation *Sigmoid* comme sortie afin de prédire l'étiquette attendue. Ensuite, la fonction de perte *Binary Cross Entropy* (BUJA et al. 2005) est employée pour propager l'erreur du réseau pendant la phase d'apprentissage, lors de la comparaison à l'étiquette prédite du modèle. Un équilibrage des données a également été ajouté pour éviter qu'une classe (étiquette bruitée ou non bruitée) ne soit dominée par une autre, en fonction de la distribution des données dans la base d'apprentissage. La fonction de perte est pondérée par l'inverse de la fréquence de la classe lors de la propagation de l'erreur, ce qui conduit à un poids plus élevé pour un échantillon avec une classe d'étiquette moins fréquente.

Comme il est impossible, du point de vue du coût mémoire, de donner toutes les données d'apprentissage, il est préférable d'utiliser des lots de données (voir 3.1.6.3). La taille du lot définit le nombre de données qui seront propagées dans le réseau, afin de mettre à jour les poids de l'ensemble du réseau en utilisant la fonction de perte pour un processus de régression basé sur le gradient. Lors de l'apprentissage de RNN, ce paramètre a également été étudié et la taille du lot a été fixée avec $b_s \in [64, 128]$, pour voir l'impact cette taille dans notre cas sur la fonction de perte, lors de la propagation de l'erreur.

7.2.2.3 Extraction des caractéristiques

Alimenter un tel réseau neuronal avec uniquement la valeur H_{SVD} pour chaque bloc de la fenêtre glissante de taille S ne semblait pas pertinent du point de vue des premiers résultats expérimentaux. Dans un bloc de taille 200×200 pixels, le bruit peut être encore très local. La valeur H_{SVD} peut perdre en sens sur un bloc complet, puisqu'elle traiterait à la fois des zones de ce bloc avec et sans bruit.

Pour permettre au modèle d'apprendre correctement le bruit dans un bloc, une subdivision a été effectuée à nouveau, fournissant m sous-blocs pour lesquels la valeur H_{SVD} a été extraite. À titre d'exemple, si l'image du bloc de taille 200×200 est découpée

en vingt-cinq sous-blocs de taille 40×40, alors le modèle aura à sa disposition un vecteur de vingt-cinq caractéristiques pour chaque bloc, où chacune est une valeur H_{SVD} extraite du sous-bloc m_i avec $i \in [0, 25[$. Il y a donc m caractéristiques extraites pour chaque image de bloc dans la fenêtre glissante de taille S.

L'intuition sous-jacente est que le modèle peut interpréter l'évolution du bruit dans chaque sous-bloc à travers une séquence de caractéristiques issue de la fenêtre glissante d'images. Cela revient à proposer un modèle ascendant, où les informations locales sont utilisées pour prédire les informations globales. Dans notre cas, la question est de savoir si une partie de l'image peut être considérée comme encore bruitée ou non.

Une question peut être ici soulevée : le fait de découper le bloc de l'image en sous-blocs pourrait peut-être introduire un biais vis-à-vis du seuil subjectif. Cependant, ce sont bien les m valeurs H_{SVD} des m sous-blocs qui sont fournies en entrée au modèle pour une image. De plus, pour chaque sous-bloc, le modèle aura connaissance de son évolution pour les S dernières valeurs. Ce qui implique que le modèle prend connaissance de l'ensemble de l'image et va essayer d'identifier un motif permettant de cibler un ou plusieurs sous-blocs considérés comme encore bruités, c'est-à-dire pour lesquels les S valeurs H_{SVD} d'un bloc m_i sont considérées comme non convergées.

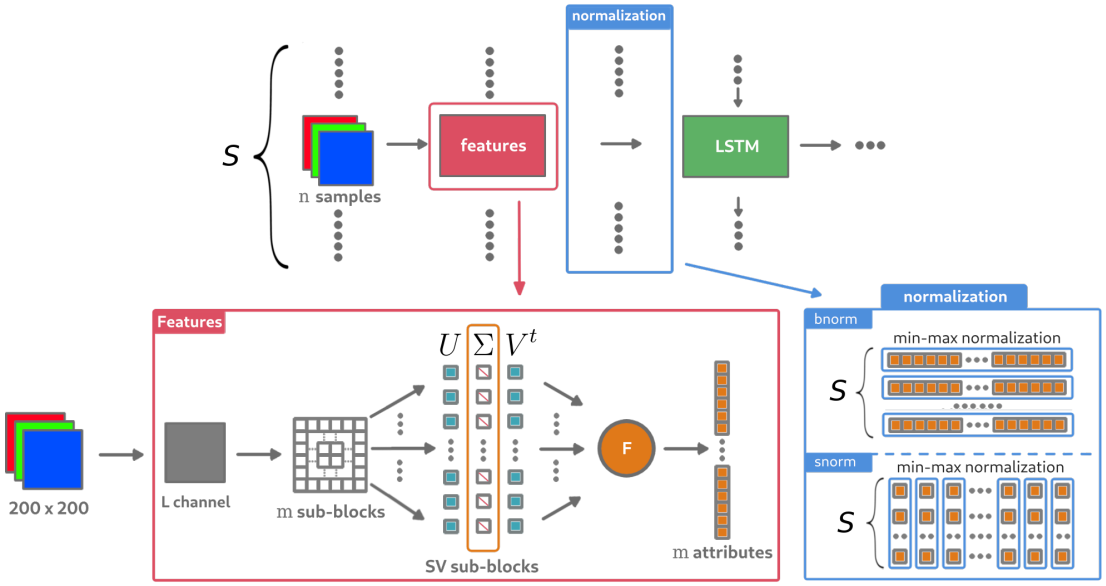


FIGURE 7.5 – Pipeline complet avec architecture RNN et paramètres testés associés. Le terme *features* est utilisé pour déterminer les caractéristiques (ou encore attributs) extraites de l'image.

En résumé, les caractéristiques qui seront utilisées comme entrées du RNN seront formées à partir des valeurs d'entropie associées à chacun des vecteurs SVD des m

sous-blocs d'un bloc d'image de 200×200 (voir le cadre rouge de la figure 7.5).

7.2.3 Résultats expérimentaux

Nous décrivons ici les études que nous avons menées pour évaluer l'approche proposée. Nous commençons par identifier les différents paramètres dont les valeurs pourraient avoir un impact sur les résultats. Nous décrivons ensuite la méthodologie de test et les mesures d'efficacité qui seront utilisées pour évaluer l'approche. Enfin, nous détaillons de manière exhaustive les résultats que nous avons obtenus, tant sur la base d'images d'apprentissage que sur des images inconnues du meilleur modèle obtenu.

7.2.3.1 Les paramètres de la méthode

En relation avec les caractéristiques d'entrée du modèle, plusieurs paramètres sont étudiés :

— la taille de la séquence en entrée du RNN avec $S \in [3, 4, ..., 10]$;
— la taille des lots de données pour l'apprentissage $b_s \in [64, 128]$;
— le nombre m de sous-blocs par bloc d'image, avec $m \in [4, 25, 100, 400]$ et donc la taille du vecteur extrait de l'image du sous-bloc. Les sous-blocs sont respectivement de taille 100×100, 40×40, 20×20 et 10×10 ;
— le pas d'échantillonnage entre les images à passer à l'entrée de la séquence RNN avec $n \in [20, 40, 80]$.

Deux paramètres supplémentaires ont été étudiés, liés au choix des composantes des vecteurs de valeurs singulières à utiliser et à leur normalisation.

Comme mentionné par (W. Liu 2014), les 3/4 des dernières composantes de faible valeur du SV semblent être liées au bruit. La question qui se pose alors est de savoir s'il faut utiliser l'ensemble du vecteur SV, le premier quart ou se limiter aux 3/4 des dernières de ses valeurs. Dans ces deux derniers cas, les mesures d'entropie réduite correspondantes sont respectivement appelées H^1_{SVD} et H^2_{SVD} et sont définies par les équations (7.4) et (7.5). La valeur extraite d'un sous-bloc est alors définie par un nouveau paramètre $F \in [H_{SVD}, H^1_{SVD}, H^2_{SVD}]$.

$$H^1_{SVD} = -\frac{1}{\log(\frac{O}{4})} \sum_{i=0}^{O/4} \overline{\sigma}_i \log_2 \overline{\sigma}_i \tag{7.4}$$

$$H^2_{SVD} = -\frac{1}{\log(O - \frac{O}{4})} \sum_{i=O/4}^{O} \overline{\sigma}_i \log_2 \overline{\sigma}_i \tag{7.5}$$

La normalisation des caractéristiques d'entrée a également été étudiée. Deux types de normalisation ont été testés : l'un où le vecteur des caractéristiques extraites d'un

sous-bloc de l'image est normalisé avant que le vecteur ne soit inséré dans la séquence d'entrée du RNN ; l'autre où chaque caractéristique est normalisée dans la séquence elle-même, afin d'étudier l'évolution de la caractéristique pour le sous-bloc (le processus de normalisation est présenté dans le cadre bleu de la figure 7.5). Ces méthodes de normalisation sont nommées ici respectivement $bnorm$ (*block-normalization*) et $snorm$ (*sequence-normalization*). Il est important de souligner que les données de la séquence entière sont nécessaires pour calculer $snorm$. La normalisation par séquence $snorm$ pourrait permettre de trouver une « pente » pour chaque sous-bloc, comme observé sur la figure 7.3. Le modèle disposerait donc d'une information sur la convergence ou non de F sur la séquence d'images S pour chaque sous-bloc. Le pipeline de la méthode proposée est détaillé dans la figure 7.5.

7.2.3.2 Validation des paramètres

Procédure de comparaisons

Les 40 images disponibles dans la base de données d'images sont divisées en 16 blocs non superposés, dont 12 blocs sont sélectionnés aléatoirement pour l'ensemble d'apprentissage et de validation et les 4 restants pour l'ensemble de test. L'idée est de permettre au modèle d'apprendre autant que possible des différentes structures des scènes. Les mêmes blocs sélectionnés sont sauvegardés et réutilisés pour chaque nouvel apprentissage d'un modèle RNN. L'objectif est de comparer les performances de chaque modèle de la manière la plus juste possible.

Pour comparer les performances d'un modèle, la mesure bien connue AUC ROC (BRADLEY 1997) présentée dans la section 3.1 est utilisée. La courbe ROC représente le taux de vrais positifs par rapport au taux de faux positifs pour différents seuils de classification des étiquettes (voir 3.1). En abaissant le seuil de classification (voir section 3.1.3.1), on classe plus d'éléments comme positifs, ce qui augmente à la fois le nombre de faux positifs et de vrais positifs. L'AUC ROC fournit une mesure agrégée des performances pour tous les seuils de classification possibles. Par conséquent, le score AUC ROC est une mesure de performance pour le problème de classification. Il indique la mesure dans laquelle le modèle est capable de distinguer les classes. La mesure de l'*Accuracy* comme présentée dans la section 3.1 du chapitre 3 sera également fournie pour indication supplémentaire de performance du modèle.

Résultats d'apprentissage et de test

Toutes les combinaisons de paramètres ont été analysées afin de trouver les meilleurs paramètres attendus sur le RNN avec des cellules LSTM. Pour chacune des combinaisons testées, les mêmes ensembles de données sont traités et le modèle de sortie est entraîné sur 30 *époques*. Une époque correspond à un passage de l'ensemble des données de la base d'apprentissage au modèle. Le tableau 7.1 affiche les 20 meilleures performances

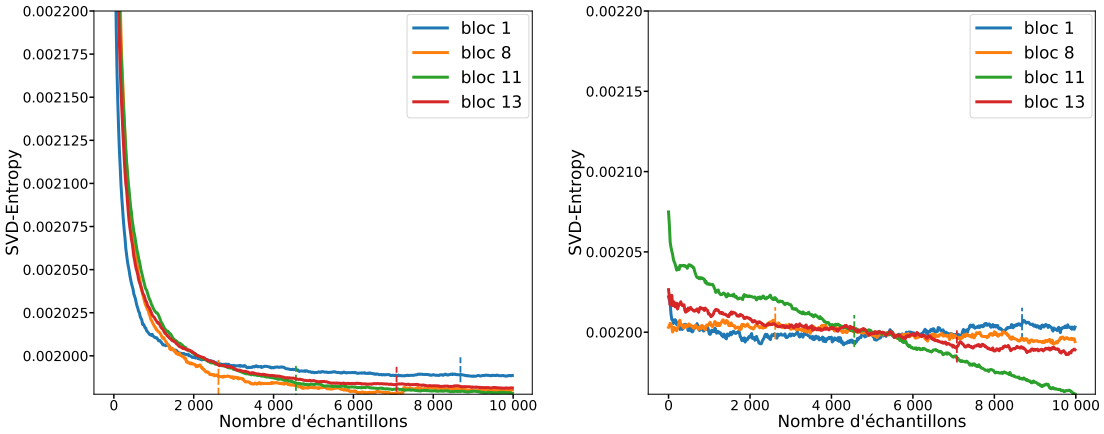
de modèle obtenues. Pour les paramètres $snorm$ et $bnorm$, la valeur 1 signifie que la normalisation correspondante a été appliquée, 0 sinon.

S	m	F	b_s	$bnorm$	$snorm$	n Step	Acc Train	Acc Test	AUC Train	AUC Test
8	100	H_{SVD}	128	0	1	40	84.58%	82.74%	84.44%	82.55%
5	100	H_{SVD}	128	0	1	80	83.78%	82.87%	83.32%	82.47%
6	100	H_{SVD}	64	0	1	40	84.18%	82.61%	84.01%	82.45%
7	100	H_{SVD}	64	0	1	40	84.62%	82.79%	84.24%	82.44%
7	100	H_{SVD}	128	0	1	40	84.65%	82.75%	84.36%	82.42%
7	100	H_{SVD}	64	0	1	80	83.67%	82.36%	83.70%	82.38%
5	100	H_{SVD}	64	0	1	80	83.61%	82.17%	83.85%	82.27%
9	100	H_{SVD}	64	0	1	40	83.46%	81.99%	83.84%	82.21%
10	100	H_{SVD}	128	0	1	40	84.52%	82.58%	84.20%	82.16%
10	100	H_{SVD}	64	0	1	20	84.12%	82.16%	84.28%	82.15%
6	100	H_{SVD}	128	0	1	40	84.05%	82.25%	84.07%	82.13%
9	100	H_{SVD}	128	0	1	40	84.39%	82.82%	83.61%	82.10%
9	100	H_{SVD}	64	0	1	20	84.93%	82.48%	84.50%	82.06%
9	100	H_{SVD}	128	0	1	20	85.37%	82.56%	84.90%	82.04%
10	100	H_{SVD}	128	0	1	20	84.73%	82.53%	84.18%	82.02%
6	100	H_{SVD}	64	0	1	80	83.04%	81.68%	83.59%	82.01%
4	100	H_{SVD}	128	0	1	80	83.17%	81.81%	83.55%	82.00%
5	100	H_{SVD}	128	0	1	40	83.88%	82.23%	83.59%	81.95%
8	100	H_{SVD}	128	0	1	80	83.61%	81.89%	83.73%	81.88%
5	100	H_{SVD}	64	0	1	40	83.85%	82.02%	83.66%	81.84%

TABLEAU 7.1 – Résultats des 20 meilleurs modèles pour chaque combinaison de paramètres, basés sur le score de l'AUC ROC sur l'ensemble de données de test. Le terme *Acc* présente la métrique *Accuracy* sur les prédictions des bases d'apprentissage (*Acc Train*) et de test (*Acc Test*).

En observant les résultats, plusieurs éléments importants apparaissent. Tout d'abord, l'utilisation de l'ensemble du vecteur SV avant de calculer l'entropie semble apporter de meilleurs résultats. En effet, aucun des meilleurs modèles n'inclut l'utilisation des mesures d'entropie réduite telles que H^1_{SVD} et H^2_{SVD}. L'interprétation que l'on peut en tirer, est que le modèle a besoin d'autant d'informations que possible, même si l'indication de bruit sur les valeurs singulières les plus fortes peut sembler faible. La figure 7.6 montre un aperçu des H^1_{SVD} et H^2_{SVD} réduits sur les mêmes blocs sélectionnés qui ont été utilisées pour la figure 7.3.

On note que H^1_{SVD} est très proche des résultats de H_{SVD}, qui exploitent l'ensemble des composantes SV (voir figure 7.3c). En utilisant H^2_{SVD} sur des blocs d'image recevant principalement un éclairage indirect (blocs 1, 11 et 13), l'entropie reste élevée en raison du fait que les composantes de poids inférieur sont très sensibles au bruit. L'utilisation de toutes les composantes avec H_{SVD} semble donner au modèle à la fois des informations sur le bruit et sur la structure, ce qui lui permet de mieux apprendre. Les informations sur la structure de la scène, qui sont principalement disponibles par le biais des valeurs singulières d'ordre inférieur (donc de plus grand module), peuvent permettre au modèle de différencier les scènes et, de manière générique, de différencier les scènes bruitées



(a) Évolution de H^1_{SVD} avec les niveaux d'échantillons normalisés et remis à l'échelle.

(b) Évolution de H^2_{SVD} avec les niveaux d'échantillons normalisés et remis à l'échelle.

FIGURE 7.6 – Aperçu de l'évolution des composantes H^1_{SVD} et H^2_{SVD} pour les 4 blocs sélectionnés dans la sous figure 7.3 de l'image *Living room*. Les lignes pointillées verticales spécifient le seuil humain pour le bloc associé.

et non bruitées. Il convient de noter que le bruit est également présent dans cette partie inférieure du vecteur SV, même si sa magnitude est beaucoup plus faible que les autres valeurs singulières. Le nombre de sous-blocs $m = 100$ semble un bon compromis : l'exploration de l'information locale semble donner plus de détails sur l'image pour le réseau. Au contraire, un excès d'information, comme avec $m = 400$, semble réduire la capacité du modèle à interpréter correctement les données.

La normalisation des données semble également fonctionner beaucoup mieux lorsque l'on normalise la séquence ($snorm$) caractéristique par caractéristique plutôt que d'utiliser $bnorm$. L'intuition derrière cela vient du fait que chaque sous-bloc de l'image d'entrée est traité indépendamment, ce qui donne sans doute au modèle une meilleure compréhension de l'évolution du bruit dans la fenêtre glissante d'images.

Plus généralement, les 40 meilleurs résultats (voir annexe C.1) obtenus sur les différentes combinaisons de paramètres testées ont fourni des scores compris entre $80,2\%$ et $82,5\%$ sur le test AUC ROC, en utilisant principalement $F = H_{SVD}$, $snorm = 1$ et $m = 100$. La seule exception est le 40e meilleur résultat, pour lequel le score AUC ROC était de $80,2\%$ avec $m = 40$. $bnorm = 1$ semble donner les plus faibles résultats et montre que le modèle interprète moins efficacement les données pour ce type de normalisation. Les meilleurs résultats en utilisant $F = H^1_{SVD}$ et $F = H^2_{SVD}$ sont respectivement de $79,12\%$ et $65,83\%$ ce qui souligne l'importance de calculer l'entropie sur toutes les composantes du vecteur SV sur chaque sous-bloc.

Un autre point très important ici est que le modèle ne semble pas surapparendre

puisque les scores d'AUC ROC sont presque du même ordre sur la base d'apprentissage et de test. Cette capacité de généralisation est très importante pour les prédictions futures sur les blocs d'images non apprises. Cette généralisation permet également de montrer la cohérence des données sur la contribution à la détection du bruit.

Parmi les meilleurs modèles obtenus (voir tableau 7.1), nous étudions l'évolution de S et son impact sur la performance du modèle. Il semblerait que les résultats des modèles augmentent plus significativement avec un score AUC ROC évoluant de $80,5\%$ à 82% lorsque $S \in [3,5]$. Ensuite, il évolue plus lentement jusqu'à un pic de convergence vers $S = 8$ (avec 82.5%) puis diminue très légèrement jusqu'à 82.22% pour $S = 10$ qui est la dernière taille de séquence testée. Les résultats semblent donc être peu sensibles à la valeur du paramètre S dans l'intervalle $[5-10]$. Ceci nous incite à considérer qu'une plus grande valeur de S ($S > 10$) ne donnerait pas forcément de meilleurs résultats et que le modèle apprend suffisamment avec un S dans l'intervalle $[5-10]$.

La première ligne du tableau 7.1 indique le modèle RNN sélectionné pour simuler la prédiction des seuils subjectifs humains sur certains points de vue dans la suite de ce chapitre. Pour ce modèle, les paramètres sont $S = 8$, $m = 100$, $F = H_{SVD}$, $b_s = 128$, $n = 40$ comme pas d'échantillons et $snorm$ comme processus de normalisation.

7.2.3.3 Comparaisons avec la littérature

Les méthodes de (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016) et de (TAKOUACHET et al. 2017) précédemment proposées pour émettre un critère d'arrêt lors du rendu sont toutes basées sur une image prétraitée, qui est utilisée comme une approximation de l'image convergée. Le principal inconvénient de cette approche est que l'image calculée n'inclut pas nécessairement certains effets complexes, qui sont généralement importants dans la simulation de l'éclairage et la distribution du bruit. L'approche que nous proposons est *sans référence*, dans le sens où seules les images calculées sont utilisées. Tous les effets lumineux sont pris en compte par le processus de simulation et une comparaison avec les approches avec référence n'aurait probablement pas de sens.

Comme souligné dans la section 7.1 de ce chapitre, le problème de dimensionnalité des données (masque de bruit en entrée) et leur nombre (diversité des scènes) ne permet pas en l'état l'apprentissage d'un SVM. Cependant, nous avons indirectement comparé notre approche avec celle de (J. CONSTANTIN, BIGAND et al. 2015) afin de déterminer si les caractéristiques extraites de la *SVD-Entropy* étaient cohérentes pour caractériser le bruit. L'idée est d'utiliser les 26 attributs (voir section 4.3.2) qu'ils ont proposés à la place des caractéristiques *SVD-Entropy*. Pour chacune des S images de la fenêtre glissante, les 26 attributs sont extraits et transmis au modèle RNN. En raison de l'aspect temporel pris en compte par ce type de réseau, les différences entre chaque élément de la séquence S seront implicitement gérées par le RNN. Cette approche semble assez proche

du mode de fonctionnement de (J. CONSTANTIN, BIGAND et al. 2015), néanmoins sans nécessiter l'utilisation d'une image de référence approximative. Pour rappel, les 26 attributs proposés par (J. CONSTANTIN, BIGAND et al. 2015) sont calculés à partir de la différence entre les 26 attributs de bruit extraits dans l'image de référence approximative et l'image bruitée en cours de calcul. Dans notre comparaison, nous proposons d'extraire pour chaque image les 26 attributs sans appliquer cette différence, mais de laisser le modèle l'interpréter par l'intermédiaire de la fenêtre glissante de taille S. Ce qui induit des données en entrée du modèle de taille $S \times 26$.

Le tableau 7.2 montre les résultats des combinaisons des mêmes paramètres du modèle en utilisant les 26 attributs proposés dans (J. CONSTANTIN, BIGAND et al. 2015) comme entrée. Seuls les paramètres m et F ne sont pas traités ici, car ils sont spécifiques à l'approche *SVD-Entropy*. Les 20 meilleurs modèles varient avec un score AUC ROC de $79,35\%$ à $81,16\%$, ce qui est légèrement inférieur au meilleur modèle avec les attributs basés sur la SVD-Entropy comme entrée du modèle ($82,55\%$). Notez ici que la normalisation de base ($bnorm$ =1) des caractéristiques de l'image elles-mêmes (le vecteur des 26 attributs) combinée à la normalisation par séquence ($snorm$ =1) semble donner des résultats corrects (score AUC ROC de $80,59\%$ pour le meilleur modèle avec ces paramètres en particulier). La normalisation $bnorm$ est en fait utilisée pour ramener au même ordre de grandeur des caractéristiques d'échelles différentes. C'est le cas pour les 26 attributs, contrairement aux attributs de la *SVD-Entropy*, qui sont déjà tous sur la même échelle (score entre 0 et 1). Le paramètre b_s ne semble pas avoir d'impact significatif sur le modèle. En revanche, le paramètre n, représentant le nombre d'échantillons entre chaque élément (image) de la séquence, semble apporter les meilleurs résultats au modèle lorsqu'il est relativement petit ($n = 20$ et $n = 40$). Les résultats indiquent également qu'un S assez grand (tel que $S > 7$) semble également donner de meilleurs résultats, car l'information est plus riche.

La deuxième ligne du tableau 7.2 représente le modèle sélectionné avec la taille de la séquence d'entrée choisie $S = 8$ afin comparer les deux méthodes avec la même quantité de donnée (taille de fenêtre glissante) en entrée du RNN H_{SVD}. Les autres paramètres sélectionnés sont donc $b_s = 64$, $bnorm = 0$, $snorm = 1$ et $n = 20$ comme pas d'échantillons. Comme pour les modèles basés sur H_{SVD}, le paramètre S a été étudié. Le modèle avec les 26 attributs est assez sensible à l'amélioration de ses performances avec un score AUC ROC évoluant de 76% à 80.5% pour $S \in [3, 8]$, avant de diminuer très légèrement vers 80%, donc $S = 8$ semble aussi être un compromis de performance.

7.2.3.4 Simulation d'éclairage assistée par un modèle de perception

Des simulations de rendu ont été traitées, afin de comparer les deux modèles sélectionnés, l'un avec H_{SVD} et l'autre avec 26 attributs, comme caractéristiques d'entrée du bruit pour chaque image de la fenêtre. La procédure d'activation d'une simulation

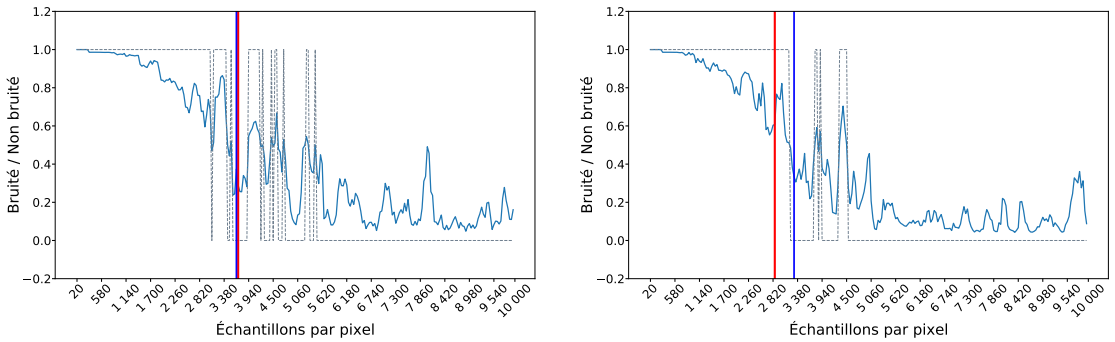
S	batch	*bnorm*	*snorm*	*n* Step	Acc Train	Acc Test	AUC Train	AUC Test
9	128	0	1	20	81.16%	81.04%	81.29%	81.16%
8	64	0	1	20	81.14%	80.93%	80.85%	80.78%
10	128	0	1	20	80.42%	80.30%	80.92%	80.72%
9	64	0	1	20	80.17%	80.19%	80.66%	80.60%
7	128	1	1	20	82.14%	80.74%	81.92%	80.59%
10	128	1	1	20	81.67%	80.35%	81.96%	80.57%
10	64	0	1	20	80.57%	80.26%	80.68%	80.41%
10	128	1	1	40	81.78%	80.94%	80.56%	80.17%
9	128	1	1	20	82.24%	80.66%	81.59%	80.16%
9	64	1	1	20	80.95%	79.57%	81.43%	79.94%
7	128	0	1	20	79.44%	79.21%	80.33%	79.90%
8	128	1	1	40	80.57%	79.63%	80.67%	79.84%
8	64	1	1	40	82.07%	80.49%	81.14%	79.83%
10	128	0	1	40	79.43%	79.09%	80.09%	79.66%
10	64	1	1	40	80.45%	79.23%	80.76%	79.63%
6	128	1	1	20	81.96%	80.15%	80.96%	79.43%
8	64	1	1	20	81.19%	79.56%	81.06%	79.43%
9	128	1	1	40	80.72%	79.68%	80.11%	79.43%
6	64	1	1	40	80.08%	79.10%	80.35%	79.38%
8	128	1	1	20	80.64%	79.22%	80.81%	79.35%

TABLEAU 7.2 – Résultats des 20 meilleurs modèles de réseau de neurones récurrent (RNN) avec les 26 attributs de (J. CONSTANTIN, BIGAND et al. 2015) en fonction du score AUC ROC sur l'ensemble de données de test. Le terme *Acc* présente la métrique *Accuracy* sur les prédictions des bases d'apprentissage (*Acc Train*) et de test (*Acc Test*).

se déroule comme suit : pour chaque niveau de bruit (nombre d'échantillons par pixel), le modèle est appelé pour prédire chaque étiquette de bloc. Le nombre d'échantillons varie de 20 à 10 000, par pixel avec un pas de $n = 40$ pour le modèle H_{SVD} et $n = 20$ pour celui des 26 attributs. Dès qu'un bloc n'est plus détecté comme bruité, le nombre d'échantillons actuel est enregistré comme seuil.

Le modèle RNN présenté dans la figure 7.4, fournit en sortie une probabilité de prédiction comprise entre 0 et 1. Si la probabilité est inférieure à $0,5$, alors l'image est considérée comme n'étant plus bruitée, sinon, elle l'est toujours. En analysant l'évolution de ces prédictions au cours de quelques simulations, un comportement commun a pu être identifié, puisque le modèle semble avoir quelques moments d'hésitation entre les deux étiquettes (bruitée / non bruitée) à proximité du seuil humain sur un bloc (voir Figure 7.7). Pour surmonter ce problème et rendre la prédiction des seuils plus robuste, il a été proposé de considérer qu'un bloc ne soit plus considéré comme bruité après 3 prédictions *non bruitée* ($p < 0,5$) successives.

Le tableau 7.3 indique les performances de chaque modèle sélectionné (H_{SVD} et 26 attributs) sur les 40 points de vue (blocs d'apprentissage et de test) pour lesquels des seuils humains étaient disponibles. La notion de marge d'erreur est introduite comme définissant le pourcentage d'erreurs de prédiction admises par rapport au nombre maximal d'échantillons utilisés pour obtenir l'image de référence autour du seuil humain. Une



(a) Bloc 5 sur le point de vue *Arc sphere*

(b) Bloc 8 sur le point de vue *Arc sphere*

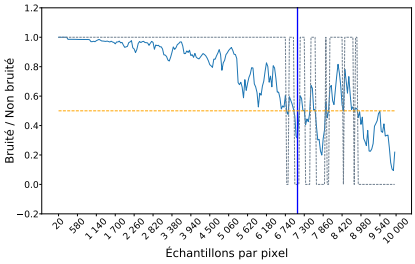
FIGURE 7.7 – Prédictions du modèle pendant le rendu du RNN sélectionné avec les données d'entrée H_{SVD}. La ligne verticale rouge correspond au seuil humain pour le bloc. La courbe bleue correspond aux prédictions du modèle pendant le rendu. La courbe en pointillés bleus fait référence à l'étiquette prédite suivant le critère strict de sélection d'étiquette (Si la probabilité est inférieure à $0,5$, alors l'image est considérée comme n'étant plus bruitée, sinon, elle est toujours bruitée). La ligne bleue verticale, fait référence aux seuils prédits obtenus.

Modèle	H_{SVD}	H_{SVD}	H_{SVD}	26 Attributs	26 Attributs	26 Attributs
Marge (in %)	2	6	10	2	6	10
Bonnes prédictions (en %)	85.80	86.55	87.91	83.01	83.80	85.27

TABLEAU 7.3 – Pourcentage de bonnes prédictions par les modèles (critère d'arrêt) avec une marge d'erreur autorisée autour du seuil humain pour l'ensemble des scènes.

marge de 2% signifie donc qu'un résultat fourni par le modèle se situe dans l'intervalle $[T - 0.01 \times Max, T + 0.01 \times Max]$ sera considéré comme une bonne réponse, avec T le seuil humain et Max le nombre d'échantillons pour les images de référence (ici 10 000 échantillons). Le taux d'erreur présenté dans le tableau correspond à l'erreur globale de prédiction du modèle vis-à-vis de cette nouvelle marge autorisée. Ceci permet de prendre en compte les fluctuations de la vision humaine autour du seuil moyen acquis et d'estimer la performance du modèle en fonction de la largeur de cette marge. Les erreurs détaillées pour chacune des 40 scènes sont disponibles dans l'annexe C.2.

L'utilisation de cette marge d'erreur permet aux modèles d'obtenir des scores de prédiction plus précis, jusqu'à plus de 88% pour une marge de 10%, pour les deux approches (H_{SVD} et 26 attributs). Bien entendu, ces scores augmentent lorsque la marge d'erreur augmente. Néanmoins, on peut noter que le modèle basé sur H_{SVD} conserve un taux de prédiction plus intéressant que celui utilisant les 26 attributs, avec une différence moyenne de $2,5$% entre les deux modèles.



(a) Prédictions sur le bloc 9 du point de vue *Classroom*

(b) Bloc 9 avec 7 000 échantillons

(c) Bloc 9 avec 10 000 échantillons

FIGURE 7.8 – Cas de prédiction de seuil erroné obtenu par le modèle. Ce cas ne dépend pas des données d'entrée (H_{SVD} ou 26 attributs) mais semble généralisé au modèle RNN. La courbe bleue correspond à la prédiction du modèle lors du rendu. La courbe en pointillés bleus fait référence à l'étiquette prédite suivant le critère strict de sélection d'étiquette (si la probabilité est inférieure à $0,5$, alors l'image est considérée comme n'étant plus bruitée, sinon, elle est toujours bruitée). La ligne bleue verticale, fait référence au seuil prédit obtenu. Le bloc est toujours considéré comme bruité avec 10 000 échantillons.

Dans certains cas, le modèle nous fournit des prédictions de seuils erronés définies par un arrêt trop précoce. La figure 7.8 montre un cas d'erreur qui met en évidence une hésitation du modèle pendant un intervalle de temps important et le conduit à fournir une estimation erronée du seuil. Le bloc ciblé ici correspond à la scène *Classroom* et est encore bruité jusqu'à 10 000 échantillons d'où le fait que le seuil humain n'est pas indiqué (voir la sous-figure 7.8c). En effet, à 10 000 échantillons, le bloc contient toujours un bruit résiduel.

7.2.3.5 Reconstruction d'images

Sur la base des simulations précédentes et des seuils prédits par le RNN avec les caractéristiques H_{SVD} en entrée, les images sont reconstruites et comparées aux images de référence. Parmi les 40 scènes, nous avons sélectionné 6 scènes pour lesquelles des seuils humains sont disponibles et présentées en annexe C.3 (images les plus bruitées) et en annexe C.4 (images de référence). La figure 7.9 montre les comparaisons entre l'image reconstruite par le modèle, l'image obtenue à partir des seuils humains et enfin l'image de référence. Un zoom est effectué sur certaines parties de chaque image. La métrique SSIM (Z. WANG et al. 2004) est utilisée pour calculer la différence entre chaque image et la référence. En effet, cette métrique est plus fidèle d'un point de vue visuel humain (SVH) que des métriques telles que la rEQM.

Toutes les prédictions du modèle sont relativement proches des images reconstruites à l'aide des seuils de perception humains, comme en témoignent les valeurs SSIM utilisées

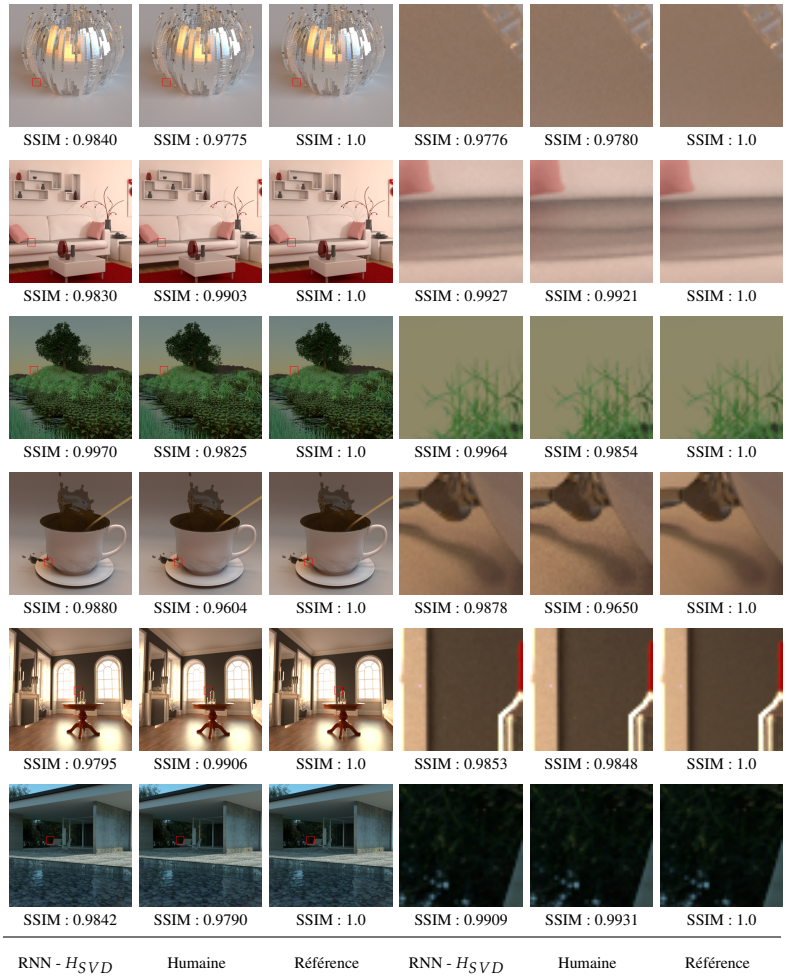


FIGURE 7.9 – Images reconstruites à partir des seuils prédits en utilisant le RNN avec le modèle SVD-Entropy et les seuils humains.

pour la comparaison. La plupart du temps, le modèle fournit un score SSIM plus élevé, le modèle allant souvent un peu plus loin que les seuils. Lorsque la valeur SSIM de l'image reconstruite à partir du modèle est inférieure à la valeur SSIM obtenue à partir des seuils perceptifs, la différence numérique reste faible et visuellement indétectable.

Le but de notre approche est évidemment d'être utilisée sur de nouvelles images que le modèle n'a jamais vues auparavant et pour lesquelles nous n'avons pas de valeur pour le seuil visuel humain. Nous avons donc réalisé de nouveaux tests sur de telles images. Nous présentons 4 points de vue non appris en annexe C.5 (images les plus bruitées obtenues pour ces images) et l'annexe C.6 (images de référence calculées avec 10 000 échantillons par pixel). Comme précédemment, nous utilisons la mesure SSIM pour évaluer la qualité des images reconstruites après que le modèle a décidé d'arrêter le

calcul de chacun des 16 blocs qui les composent. Les résultats obtenus sont présentés dans la figure 7.10 et montrent que le modèle obtenu semble suffisamment robuste pour fournir des seuils d'arrêt qui ne permettent plus la perception du bruit.

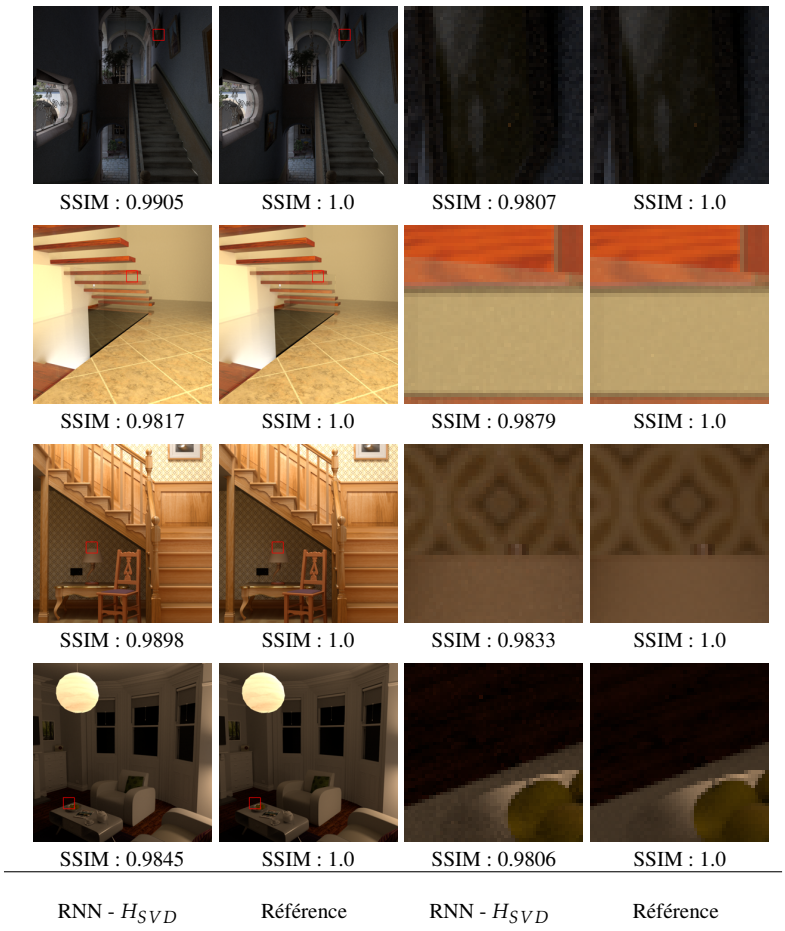


FIGURE 7.10 – Images reconstruites à partir des seuils prédits en utilisant le RNN avec le modèle SVD-Entropy sur les points de vue non appris comparés à la référence.

La figure 7.11 donne un aperçu des seuils obtenus sur chacune de ces quatre images. Même si aucun seuil humain n'est disponible, le nombre d'échantillons prédit par notre modèle est cohérent avec les objets et l'éclairage présents dans chaque bloc. Les blocs des scènes avec un éclairage principalement direct nécessitent beaucoup moins d'échantillons, tandis que les blocs avec un éclairage principalement indirect nécessitent plus d'échantillons. Un cas difficile d'éclairage indirect est illustré par la scène *San Miguel*, où la plupart des blocs nécessitent un grand nombre d'échantillons. Notez que plusieurs blocs sont évalués comme nécessitant 10 000 échantillons : pour ces blocs, la convergence n'est pas atteinte dans l'image de référence qui a été calculée avec 10 000 échantillons et, dans le contexte de cette étude, nous avons arrêté les calculs

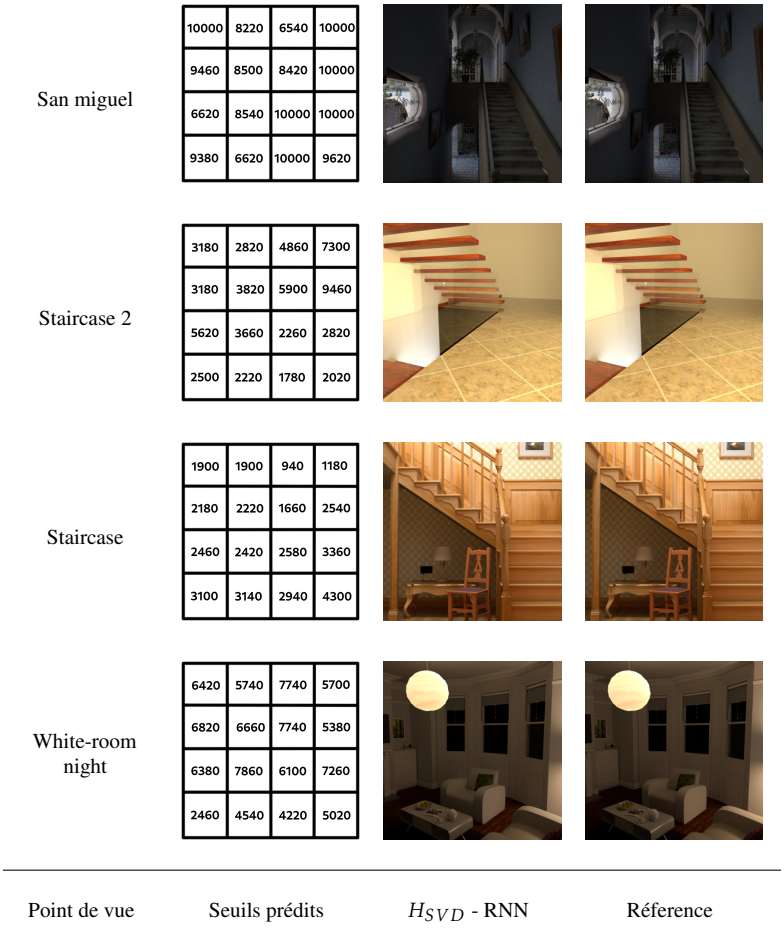


FIGURE 7.11 – Seuils prédits à partir du modèle RNN H_{SVD} et vue d'ensemble des blocs reconstruits de 4 images non apprises avec comparaison de chacun à l'image de référence correspondante.

avec cette valeur maximale. La réponse du modèle, ayant atteint cette limite, est que le bloc est toujours bruité. L'image de la scène de la *Whiteroom-night* est un autre exemple d'éclairage indirect (la source lumineuse est incluse dans le luminaire semi-transparent au plafond), pour lequel le modèle propose des seuils avec des valeurs importantes, mais inférieures au maximum autorisé. Le dessous de la scène *Staircase* est éclairé indirectement, mais le mur qui s'y trouve est recouvert d'une texture de papier peint. Le phénomène de masquage de texture réduit ainsi la perception du bruit et limite le nombre d'échantillons nécessaires. Notons également les blocs 9 et 10 pour la scène *Staircase 2*, qui nécessitent un nombre relativement important d'échantillons. La convergence du tracé du chemin est ici rendue plus difficile en raison de la présence d'une balustrade en verre, qui génère du bruit pendant une période plus longue. Dans l'ensemble, les résultats des prédictions obtenues sont assez représentatifs des scores SSIM présentés

précédemment dans la figure 7.10 et montrent que les prédictions du modèle semblent correctes sur les nouvelles données d'image.

7.2.4 Discussion

Les méthodes de simulation d'éclairage utilisées en synthèse d'images fournissent des images qualifiées de photoréalistes et la puissance de ces techniques en fait un outil de plus en plus prisé par les sociétés de production multimédia. Néanmoins, ces simulations sont intrinsèquement sujettes au bruit visuel, qui dégrade la qualité des images obtenues, en raison de leur utilisation de méthodes de calcul basées sur l'intégration de Monte-Carlo. La réduction du bruit nécessite une augmentation du nombre d'échantillons de chemins lumineux à utiliser, sans qu'un critère fiable ne soit disponible à l'heure actuelle pour définir un seuil d'arrêt des calculs.

Ce chapitre vise à proposer une méthode entièrement générique pour établir un critère d'arrêt perceptif pour la génération de ce type d'image. Elle est basée sur l'utilisation de seuils perceptifs humains obtenus à partir d'une base de données d'images. Cette approche est basée sur un puissant outil d'algèbre linéaire (SVD), la *SVD-Entropy* associée et leur exploitation par RNN. Cette méthode générique donne de bons résultats pour tous les types d'images avec des matériaux diffus et spéculaires, ainsi qu'avec un éclairage direct et indirect. Elle semble donc offrir une certaine robustesse, grâce au couplage de l'information locale de la *SVD-Entropy* et à l'utilisation du traitement temporel de cette information au sein d'un réseau neuronal récurrent. L'utilisation de la *SVD-Entropy* comme attribut d'entrée semble plus intéressante que d'autres attributs publiés précédemment, dont l'utilisation donne des résultats légèrement inférieurs si on les utilise au sein de la même architecture.

Soulignons que le modèle de perception de bruit fournit des prédictions intéressantes aussi bien sur les blocs appris que non appris. Ainsi, les critères d'arrêt permettent d'obtenir des images où le bruit est fortement réduit. Toutefois, le modèle fournit un seuil d'arrêt trop précoce dans quelques rares cas (voir figure 7.8), ce qui peut amener à une présence de bruit encore perceptible.

Dans ce chapitre, nous avons étudié une approche utilisant la factorisation matricielle SVD pour générer les attributs représentatifs du bruit. Dans la conclusion de ce mémoire consacré en partie aux perspectives, nous envisageons de nouvelles pistes portant sur d'autres méthodes de factorisation et de compression de données.

Résumé

Ce septième chapitre met en avant dans un premier temps les problèmes de dimensionnalités et ceux liés aux quantités importantes de données à traiter pour les méthodes d'apprentissage qui s'appuient sur les SVM. Il propose une première contribution de modèle de perception de bruit de MC basée sur une méthode d'apprentissage profond. Le modèle exploite un réseau de neurones récurrents et une fenêtre glissante d'images en entrée. Pour chaque image sont extraits des attributs locaux de bruit du bloc de l'image à partir de la *SVD-Entropy* qui semblent correctement caractériser le bruit. Les prédictions du modèle sont dans l'ensemble correctes et les images obtenues à partir de ces seuils prédits sont relativement proches de celles obtenues avec les seuils subjectifs humains. Toutefois, dans certains cas rares, le modèle propose des prédictions précoces de seuils qui ne sont pas désirées.

Chapitre 8

Génération automatique de caractéristiques de bruit

Introduction

Ce chapitre introduit une étude réalisée dans le cadre de la thèse également relative à la conception d'un modèle de perception de bruit résiduel de Monte-Carlo. Cette approche comme pour le chapitre précédent est également basée sur de l'apprentissage profond (*Deep Learning*). Elle propose l'utilisation d'une architecture avancée, composée de 3 modèles d'apprentissage profond : le premier permettant de débruiter une image, le second permettant de générer une carte caractéristique du bruit présent dans l'image et le dernier ayant pour objectif d'identifier la présence ou non du bruit dans l'image. L'approche a l'avantage d'apprendre du problème d'identification du bruit pour extraire des informations fines du bruit présent dans les images.

L'estimation des caractéristiques à extraire d'une image pour des tâches de classification est parfois difficile, surtout si les images sont liées à un type de bruit particulier. L'objectif de ce chapitre est de proposer une architecture de réseau de neurones nommée *Guided-Generative Network* (GGN), pour extraire des informations raffinées permettant de quantifier correctement le bruit présent dans une fenêtre glissante d'images. Le GGN tend à trouver des cartes de bruit souhaitées pour émettre un critère de détection de ce bruit. Le GGN proposé est appliqué sur la problématique de la thèse, soit les images photoréalistes qui sont rendues par des méthodes de Monte-Carlo en évaluant un grand nombre d'échantillons par pixel pour quantifier le bruit résiduel. La section 8.1 propose un rappel du problème lié à la tâche de classification demandée des images. La section 8.2 remémore les travaux liés à la quantification du bruit de MC avant de présenter l'architecture GGN proposée 8.3. Les résultats obtenus avec ce nouveau modèle sont comparés à la précédente approche dans la section 8.4 avant de discuter des perspectives

d'une telle approche 8.5

8.1 Définition du problème

La convergence de la génération d'une image de synthèse nécessite souvent plusieurs heures (voire plusieurs jours) avant qu'une image visuellement utilisable soit disponible, en raison à la fois de la complexité du calcul des chemins et du nombre élevé d'échantillons requis.

De plus, l'information sur le nombre de chemins réellement nécessaires pour que l'image soit visuellement convergente est inconnue. Arrêter le calcul de manière anticipée fournit des images avec du bruit visuel et calculer trop d'échantillons peut entraîner une perte de temps et des coûts de production plus élevés. De plus, l'identification des caractéristiques pertinentes pour l'identification du bruit visuel est difficile, étant donné la nature du bruit généré, qui est très dépendant des algorithmes de calcul utilisés et de la complexité des scènes virtuelles (propriétés des matériaux, effets caustiques, éclairage indirect...).

Nous proposons dans ce chapitre d'exploiter des méthodes de *Deep Learning* telles que l'autoencodeur de débruitage U-Net (RONNEBERGER et al. 2015) et les réseaux neuronaux génératifs (GAN) (GOODFELLOW et al. 2014), pour générer automatiquement des *Noise Feature Maps* (NFM), carte de caractéristiques du bruit, qui vont guider un réseau neuronal discriminateur afin de mieux caractériser la tâche d'identification du bruit humainement perceptible dans les images.

8.2 Caractérisation du bruit MC

Trouver des informations quantitatives relatives au bruit résiduel Monte-Carlo n'est pas une tâche facile. Identifier les caractéristiques représentatives du bruit de MC est également une tâche complexe et nous avons déjà examiné les propositions faites antérieurement à nos travaux, telles que celles de (TAKOUACHET et al. 2017) et (J. CONSTANTIN, BIGAND et al. 2015 ; J. CONSTANTIN, I. CONSTANTIN et al. 2016).

Nous avons également proposé dans le chapitre précédent d'utiliser la mesure *SVD-Entropy* (BANERJEE et Nikhil R. PAL 2014b) comme caractéristique de bruit. La *SVD-Entropy* est calculée localement sur chaque bloc, en prenant en compte une fenêtre glissante composée de S images pour laquelle chaque version du bloc a un niveau de bruit décroissant. Le modèle obtenu semble généraliser correctement sur les blocs non appris de nombreuses images mais fournit parfois des résultats imprécis.

8.3 *Guided-Generative Network*

Il est difficile d'identifier les caractéristiques représentatives du bruit dans les images générées par les méthodes de MC. Nous avons pour objectif ici de proposer une approche où ces caractéristiques sont générées automatiquement avant la classification binaire. Pour cela, nous nous appuyons sur la notion de masque de bruit (TAKOUACHET et al. 2017) qui sera fournie par un modèle de réseau de neurones génératif, mais aussi de l'utilisation d'une fenêtre glissante d'images qui semble apporter une meilleure robustesse des modèles (voir chapitre 7). La proposition d'une telle approche est justifiée par l'émergence de méthodes d'apprentissage par convolution profonde robustes, tant pour le traitement du bruit que pour la reconnaissance de « pattern » dans l'image (HASELMANN et al. 2018 ; PRAVIN et al. 2020).

L'architecture GGN que nous proposons, a pour but d'obtenir des caractéristiques automatiques liées au bruit. Elle est composée de 3 modèles de réseaux neuronaux, pour lesquels une fenêtre glissante d'images de différents niveaux de bruit de taille S est utilisée. Avant d'entrer dans les détails de chaque sous-modèle, la figure 8.1 illustre l'interaction souhaitée entre chacun d'eux.

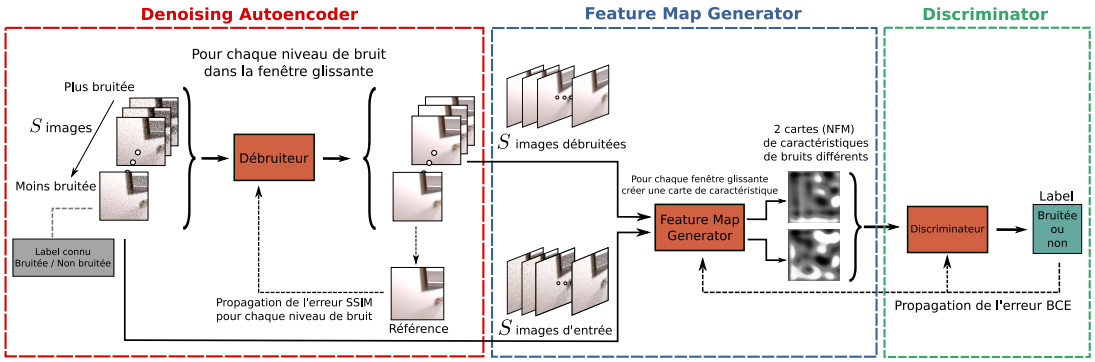


FIGURE 8.1 – Architecture de modèle GGN proposé, où une fenêtre glissante d'images de différents niveaux de bruit et de taille S est donnée en entrée au modèle *AutoEncoder* pour le débruitage afin d'obtenir une fenêtre glissante d'images de référence approchées de taille S. L'une après l'autre, les deux fenêtres coulissantes d'images, celle d'entrée et celle de référence approchée, sont envoyées au modèle Feature Map Generator pour obtenir pour chacune une carte de caractéristiques de bruit (NFM). Enfin, les deux NFM obtenues sont transmises au Discriminateur pour évaluer si la dernière image de la fenêtre glissante d'entrée est considérée comme étant toujours bruitée ou non. La propagation de l'erreur est alors possible grâce à l'étiquette connue (bruitée / non bruitée) de la dernière image de la fenêtre glissante d'entrée.

8.3.1 *Denoising Autoencoder*

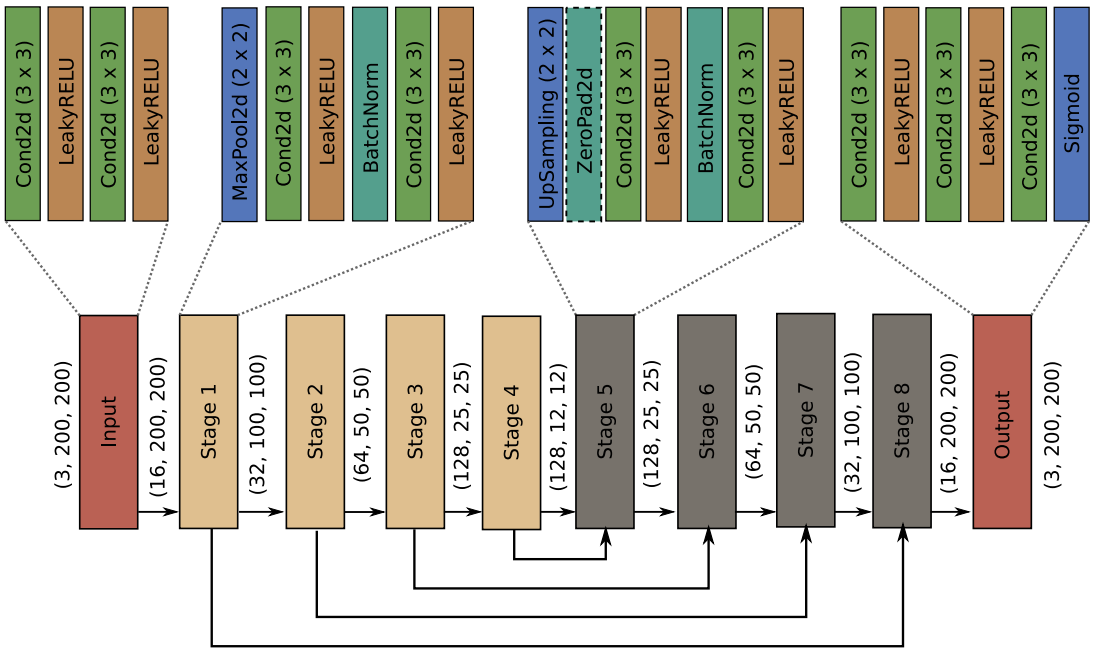


FIGURE 8.2 – Architecture du *Denoising Autoencoder* exploitée prenant en entrée une image RGB de taille $3 \times 200 \times 200$ bruitée et proposant en sortie une image de même taille débruitée. Pour la première couche du réseau, le passage de 3 canaux à 16 est réalisé par la convolution appliquée à l'image, où il est possible de définir le nombre de canaux attendus en sortie de cette couche. Il est important de préciser que seul le *Stage 5* comporte une couche *ZeroPad2d* pour permettre d'obtenir une taille de sortie désirée après suréchantillonnage, ici impaire ($128 \times 25 \times 25$). Les connections intermédiaires des différentes étapes de l'encodeur et du décodeur, qui sont propres au modèle U-Net, sont également illustrées.

Dans (TAKOUACHET et al. 2017), un masque de bruit est calculé à partir de l'image en cours de calcul (potentiellement bruitée) et d'une image de référence approximative, obtenue par la méthode du lancer de rayons. Le principal inconvénient est que certains effets lumineux importants ne peuvent pas être simulés par le lancer de rayons et induisent donc des erreurs par rapport à l'image finale qui devrait être calculée. L'idée proposée pour remédier à ce problème est un réseau neuronal *AutoEncodeur* permettant de débruiter au mieux une image, d'où l'appellation *Denoising Autoencoder*. En informatique graphique, la préservation des structures et des effets lumineux est importante, c'est pourquoi l'*AutoEncodeur* utilisé est de type U-Net tel qu'exploité pour la tâche de débruitage dans (JIANG et al. 2019 ; RONNEBERGER et al. 2015), il permet de préserver plus facilement la structure de l'image tout en proposant un débruitage performant de

l'image d'entrée. U-Net est un *AutoEncodeur* convolutif avec des connexions par saut et une progression régulière de la dimensionnalité (voir section 3.1). La figure 8.2 illustre l'architecture d'un tel réseau U-Net utilisée dans le cadre de nos expériences, avec une image en entrée de taille $3 \times 200 \times 200$. L'encodeur et le décodeur proposés ont une structure symétrique : chacune des quatre étapes de l'encodeur utilise deux couches de convolution avec un noyau de 3×3 et double la profondeur dimensionnelle K qui correspond au départ aux canaux RGB de l'image. Au contraire, chacune des 4 étapes du décodeur possède deux couches de déconvolution de noyau 3×3 et réduit la profondeur de moitié afin de restituer la taille de l'image initiale.

Toutes les étapes intermédiaires utilisent une normalisation par lot et des fonctions d'activation *LeakyReLU* [1] (SHARMA et al. 2017). L'étape de sortie comporte deux couches de déconvolution de noyau 3×3 avec une couche *LeakyReLU*, et une convolution finale de 1×1 avec une activation *LeakyReLU* pour produire l'image débruitée finale. Chaque étape du décodeur utilise une couche *MaxPooling* de noyau 2×2, et les étapes de l'encodeur utilisent un suréchantillonnage bilinéaire [2] de noyau 2×2. Une couche supplémentaire de *ZeroPadding* [3] est utilisée pour une étape de décodeur lorsqu'il est nécessaire d'obtenir une taille de tuile impaire après le suréchantillonnage. Nous avons fixé $K = 16$, car l'image d'entrée a une taille de 200×200, ce qui implique un stockage assez important et un nombre de neurones conséquent. De plus, la métrique de similarité structurelle (SSIM) (Z. WANG et al. 2004) a été utilisée comme fonction de perte, c'est-à-dire, $L(\hat{y}, y) = 1 - SSIM(\hat{y}, y)$, où $\hat{y}$ est l'image d'un bloc de référence connu calculée avec 10 000 échantillons, et y est l'image de sortie du réseau neuronal U-Net. La SSIM offre également une bonne préservation de la structure (BUJA et al. 2005 ; SHI et al. 2020) par rapport à une fonction de type L1 (EMA) ou L2 (EQM), toutes deux liées aux valeurs des pixels. L'utilisation d'une fenêtre glissante d'images d'entrée composée de S images est en fait traitée pour une meilleure robustesse de détection par la suite. Chaque image de niveau de bruit différent est envoyée l'une après l'autre au U-Net afin d'être débruitée et d'obtenir une fenêtre glissante d'images de référence approchées composée de S images. Ainsi, le modèle apprend à débruiter les images de plusieurs niveaux de bruit.

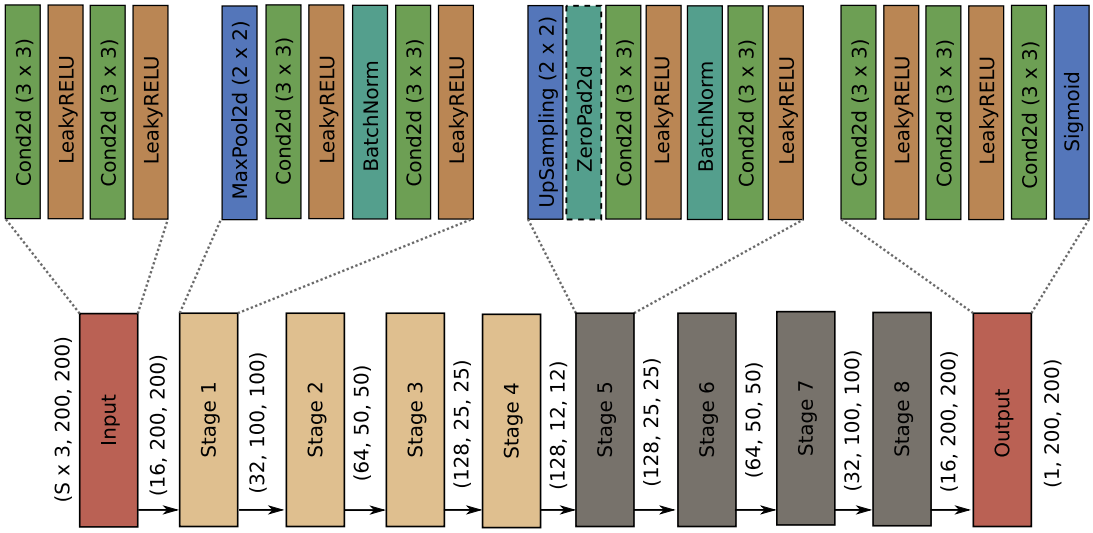


FIGURE 8.3 – Architecture du *Feature Map Generator* exploité prenant en entrée une fenêtre glissante d'images de taille $S \times 3 \times 200 \times 200$ et proposant une NFM. Pour la première couche du réseau, le passage de $S \times 3$ canaux à 16 est réalisé par la convolution appliquée à l'image, où il est possible de définir le nombre de canaux attendus en sortie de cette couche. Il est important de préciser que seul le *Stage 5* comporte une couche *ZeroPad2d* pour permettre d'obtenir une taille de sortie désirée après suréchantillonnage, ici impaire ($128 \times 25 \times 25$).

8.3.2 *Feature Map Generator*

Le générateur de cartes de caractéristiques (*Feature Map Generator*), est également un *AutoEncoder* et possède la même structure que le modèle de débruitage précédent, mais sans connexions de saut et sans progression régulière de la dimensionnalité comme le propose un modèle *U-Net*. Une convolution finale de 1×1 est appliquée avec une activation *LeakyReLU* pour produire l'image grise attendue avec $K = 1$, soit une image en niveaux de gris. Par conséquent, ce modèle prend en entrée une fenêtre glissante d'images composée de S images et vise à produire une NFM sans connaître réellement ce qu'il doit générer. Dans notre approche (voir 8.1), il prendra soit la fenêtre glissante des images d'entrée, soit la fenêtre glissante des images de référence approchées obtenues précédemment avec le *Denoising Autoencoder*. La figure 8.3 propose un aperçu de

1. La fonction d'activation *LeakyReLU* est une extension de la fonction *ReLU* qui peut permettre l'activation du neurone même si $x < 0$, avec $0.01x$ dans cette condition au lieu d'une réponse stricte de 0 proposée par la fonction *ReLU*.

2. Une couche de suréchantillonnage bilinéaire permet d'augmenter la dimension des données, l'objectif étant ici de restituer la taille originale de l'image.

3. Une couche *ZeroPadding* est une technique qui permet de préserver la taille originale de l'entrée. Elle propose l'ajout d'une bordure de pixels ayant tous la valeur zéro autour des bords des images d'entrée.

l'architecture et des paramètres de ce réseau. Il est important de noter que des modèles tels que les *Variational AutoEncoders* (VAE) (KINGMA et al. 2014), qui offrent généralement de meilleures données générées, ont également été testés, mais n'ont pas donné de bons résultats pour ce genre de génération.

8.3.3 Discriminateur pour la classification binaire

Le discriminateur prend en entrée 2 NFM obtenues à partir du *Feature Map Generator*, la NFM de la fenêtre glissante d'entrée et la NFM de la fenêtre glissante des images de référence approchées. Le discriminateur doit fournir en sortie une étiquette binaire qui correspond à la présence de bruit ou non dans la dernière image de la fenêtre glissante. Un discriminateur traitant des images est généralement composé de deux phases (GAYATHRI et al. 2020 ; GOODFELLOW et al. 2014) : la première qui va réduire les dimensions et extraire des informations de l'image, la seconde composée d'un réseau de neurones multi-couches classique qui cherche, à partir des informations précédemment extraites, essayer d'estimer le label attendu. Ainsi, nous proposons un discriminateur composé de trois couches de convolution qui doublent la profondeur dimensionnelle K avec une taille de noyau de 3×3 et un *padding* [4] de 2 pour réduire la dimensionnalité de la NFM. Pour chacune de ces couches de convolution, des couches intermédiaires de normalisation par lots, *LeakyReLU* et *MaxPooling* avec un noyau de 3×3, un *stride* [5] de 2 et un padding de 1 sont traitées. Ensuite, quatre couches linéaires classiques sont exploitées pour propager et réduire l'information, jusqu'à atteindre une probabilité d'appartenance à une étiquette bruitée ou non bruitée. La première couche linéaire renvoie des données de taille $K \times 4$, puis les deux couches suivantes réduisent de moitié les données de sortie. La dernière couche linéaire propose une seule sortie, avant d'appliquer une fonction *Sigmoïde*. Chaque couche linéaire comporte des couches intermédiaires, telles qu'une normalisation par lot, une couche *LeakyReLU* et une couche de *dropout* de 50% pour éviter le surapparentissage (SRIVASTAVA 2013). Seule la dernière couche, où la probabilité est obtenue, ne comporte pas de couche de *dropout* (voir section 3.1.4.1). La figure 8.4 illustre l'architecture et les différentes couches du réseau discriminant envisagé.

La fonction de perte *Binary Cross-Entropy* (BUJA et al. 2005) est utilisée à la fois pour propager l'erreur du discriminateur, mais aussi le générateur de cartes de caractéristiques. Cela permet d'obtenir des NFM représentatives de la tâche de classification attendue et

4. Le *padding* définit l'effet de bordure dans la carte des caractéristiques et la façon dont il peut être surmonté par le remplissage lors de l'application d'un filtre par convolution.

5. La quantité de mouvement entre les applications du filtre à l'image d'entrée est appelée le *stride*, et elle est presque toujours symétrique en hauteur et en largeur. Par exemple, le stride peut être modifié en (2,2). Cela a pour effet de déplacer le filtre de deux pixels vers la droite pour chaque mouvement horizontal du filtre et de deux pixels vers le bas pour chaque mouvement vertical du filtre lors de la création de la carte de caractéristiques.

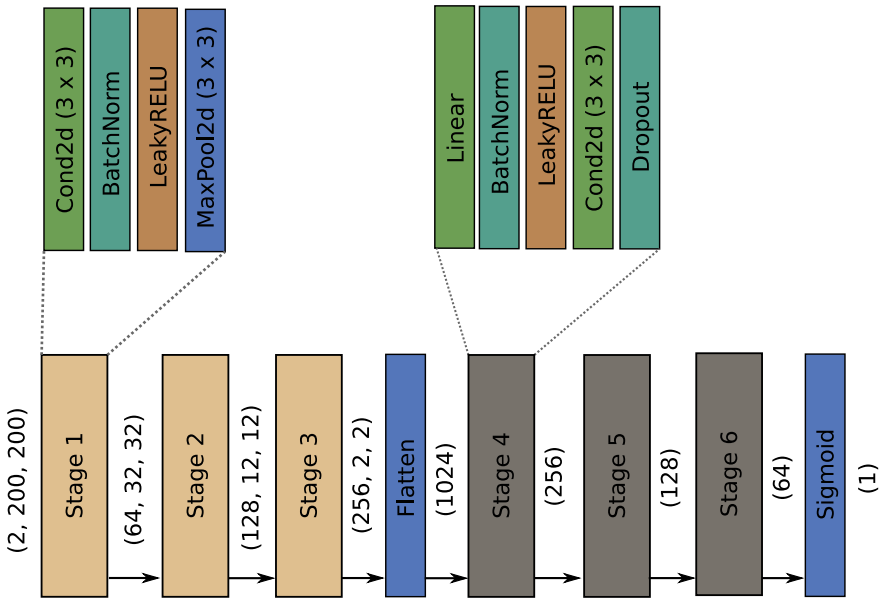


FIGURE 8.4 – Architecture du *Discriminator* prenant en entrée les deux NFM générées, soit une entrée de taille $2 \times 200 \times 200$ et proposant une prédiction de présence de bruit ou non en sortie. Pour la première couche du réseau, le passage de 2 canaux à 64 est réalisé par la convolution appliquée à l'image, où il est possible de définir le nombre de canaux attendus en sortie de cette couche.

de guider le discriminateur, d'où le nom de *Guided-Generative Network* (GGN) pour une telle architecture.

8.4 Apprentissage et résultats

L'architecture proposée, composée des trois modèles décrits ci-dessus, est entraînée sur les données de notre base d'images. Les trois modèles apprennent ensemble par rapport aux données d'entrée, comme le montre la figure 8.1. Parmi les 40 points de vue disponibles de la base d'images, 35 ont été sélectionnés pour l'entraînement, et 5 pour la vérification des performances du modèle.

8.4.1 Configuration d'expérience

La quantité de données est assez importante avec 35 images, où chaque image est composée de 16 blocs de taille 200×200. Pour chaque bloc, 500 différents niveaux de bruit allant de 20 échantillons à 10 000 échantillons sont utilisés. Cela permet d'apprendre à partir d'un total de 280 000 données étiquetées. En ce qui concerne les

paramètres d'apprentissage, le modèle a été entraîné sur les données pendant 30 époques, avec une taille de lot de 64 et une fenêtre glissante d'images composée de $S = 6$ images. Cette taille de fenêtre glissante correspond à la taille maximale possible relativement à la capacité mémoire de notre GPU. De plus, les temps d'apprentissage étant longs (pratiquement 2 semaines), nous nous sommes limités à cette taille de fenêtre glissante.

8.4.2 Résultats obtenus

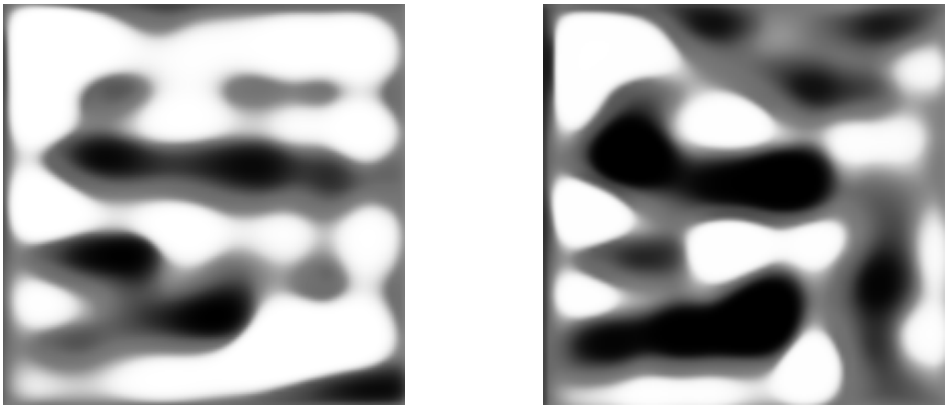
Le modèle U-Net *Denoising AutoEncoder* semble fournir des résultats de suppression du bruit MC proches de l'image de référence, comme l'illustre la figure 8.5, où les scores SSIM montrent une amélioration de la qualité et de la proximité de l'image résultante par rapport à l'image de référence.

(a) Image bruitée SSIM : 0.6433 | (b) Image débruitée SSIM : 0.9859 | (c) Référence SSIM : 1.0

FIGURE 8.5 – Résultat du débruitage par le *Denoising AutoEncoder* sur un bloc de l'image *Kitchen* avec les scores SSIM correspondants.

Une vue d'ensemble des sorties du générateur de cartes de caractéristiques est disponible dans la figure 8.6, pour le même bloc d'image que celui utilisé dans la figure 8.5.

Notons que des différences apparaissent entre les deux NFM obtenues. Celles-ci sont dues au fait que des niveaux de bruit différents coexistent dans la même fenêtre glissante pour les images calculées, alors que ces différences sont atténuées dans les images débruitées dans la seconde fenêtre glissante. Ceci confirme l'intérêt de l'approche, puisque la NFM fournit une information discriminante quant à la présence ou l'absence de bruit dans les images considérées. Ces deux NFM sont ensuite envoyées au discriminateur pour évaluer la probabilité que la dernière image de la fenêtre glissante d'entrée appartienne à l'étiquette 1, bruitée, ou à l'étiquette 0, non bruitée.



(a) *Feature map* des images d'entrée (b) *Feature map* des références approchées

FIGURE 8.6 – Résultats de sortie du *Feature Map Generator* pour le même bloc de la figure 8.5 : la NFM des fenêtres glissantes des images en cours (a) et la NFM des images débruitées (b). Les motifs blancs représentant davantage la présence du bruit dans la fenêtre glissante.

8.4.3 Comparaisons des modèles

Le modèle obtenu après un apprentissage de 30 époques sur le jeu de données d'entraînement est comparé au RNN proposé précédemment, exploitant la *SVD-Entropy* avec les mêmes conditions d'entraînement, c'est-à-dire le même jeu de données d'entraînement et de test, la même taille de lot et la même taille de fenêtre glissante ($S = 6$). Les métriques utilisées pour comparer les performances des deux modèles sont l'*accuracy* et l'aire sous la courbe AUC ROC (BRADLEY 1997), qui définit la façon dont le modèle sépare les deux classes de la classification binaire, quel que soit le seuil de probabilité d'appartenance à une classe.

Les performances des modèles sur les ensembles de données d'entraînement et de test sont présentées dans le tableau 8.1 avec plusieurs seuils de décision (voir section 3.1.3.1) afin de vérifier les performances de détection (avec l'*accuracy*). Le modèle LSTM semble avoir une meilleure généralisation. Cependant, même si le GGN présente un léger surapprentissage, il affiche une performance correcte sur la base de test lorsque son seuil de classification $t \geq 0.95$. La figure 8.7 montre les prédictions des modèles pendant le rendu d'une image et illustre le comportement général des résultats de prédiction obtenus. Le seuil de classification du GGN est fixé à $t = 0,95$, car cette valeur permet d'éviter une prédiction précoce sur l'ensemble des images de la base de données d'entraînement, bien que son *accuracy* soit plus faible. En effet, une capacité conservatrice, bien que moins efficace, est à préférer à une fin précoce, qui impliquerait une mauvaise qualité de l'image finale obtenue. On peut remarquer que le GGN fluctue et hésite beaucoup moins que le LSTM, ce qui indique une stabilité du modèle lors de la prédiction au cours du

	t threshold	Accuracy Train	AUC ROC Train	Accuracy Test	AUC ROC Test	Accuracy Global	AUC ROC Global
LSTM	0.3	0.7619	0.9115	0.8122	0.9297	0.7682	0.9137
	0.4	0.8085	0.9115	0.8456	0.9297	0.8131	0.9137
	0.5	0.8281	0.9115	**0.8519**	0.9297	0.8311	0.9137
	0.6	0.8329	0.9115	0.8385	0.9297	0.8336	0.9137
	0.7	0.8225	0.9115	0.8055	0.9297	0.8204	0.9137
	0.8	0.7965	0.9115	0.7542	0.9297	0.7912	0.9137
	0.9	0.7533	0.9115	0.6896	0.9297	0.7454	0.9137
	0.95	0.7331	0.9115	0.6654	0.9297	0.7246	0.9137
	0.98	0.6888	0.9115	0.6284	0.9297	0.6812	0.9137
GGN	0.3	0.6598	0.9735	0.6641	0.8422	0.6603	0.9571
	0.4	0.7152	0.9735	0.6719	0.8422	0.7098	0.9571
	0.5	0.7902	0.9735	0.7310	0.8422	0.7828	0.9571
	0.6	0.8219	0.9735	0.7415	0.8422	0.8118	0.9571
	0.7	0.8553	0.9735	0.7554	0.8422	0.8429	0.9571
	0.8	0.8809	0.9735	0.7709	0.8422	0.8672	0.9571
	0.9	0.9067	0.9735	0.7858	0.8422	0.8916	0.9571
	0.95	0.9204	0.9735	0.8013	0.8422	0.9055	0.9571
	0.98	0.9201	0.9735	**0.8216**	0.8422	0.9078	0.9571

TABLEAU 8.1 – Performances des modèles sur les ensembles d'apprentissage et de test avec plusieurs seuils de décision de probabilité **t** proposés pour comparer la précision de chaque modèle. Le score AUC ROC ne change donc pas pour la valeur **t**, mais reste indicatif. Les meilleurs scores de précision sont indiqués par un fond gris correspondant à la valeur **t**.

temps. Cela permet notamment dans certains cas d'éviter une prédiction trop précoce du calcul, illustré dans la figure 8.7 pour le bloc 13 de l'image *Classroom*. D'autre part, GGN a parfois tendance à prédire plus tard que LSTM comme pour le bloc 16 de l'image *San-Miguel*.

La figure 8.8 quant à elle, présente également des résultats de prédictions du GGN comparé au RNN avec la même taille de fenêtre glissante $S = 6$, mais sur le bloc 9 de *Classroom*, bloc avec une prédiction d'arrêt anticipée pour le RNN. Le GGN pour lequel nous proposons ici de visualiser la prédiction au-delà de 10 000 échantillons, semble mieux interpréter que le RNN ce bloc et fournit un seuil plus tardif que le modèle RNN. En effet, le RNN entraîné avec des données et quelques paramètres différents ($S = 6$ notamment) que celui proposé précédemment en section 7.2, semble toujours avoir le même problème d'arrêt anticipé.

De manière globale, le GGN montre de bons résultats même si, au départ, aucune indication sur les caractéristiques du bruit n'était donnée. Le modèle s'est adapté au cours de son apprentissage pour répondre à la tâche demandée, trouvant seul les caractéristiques requises et fournit des prédictions assez précises sur la base de test.

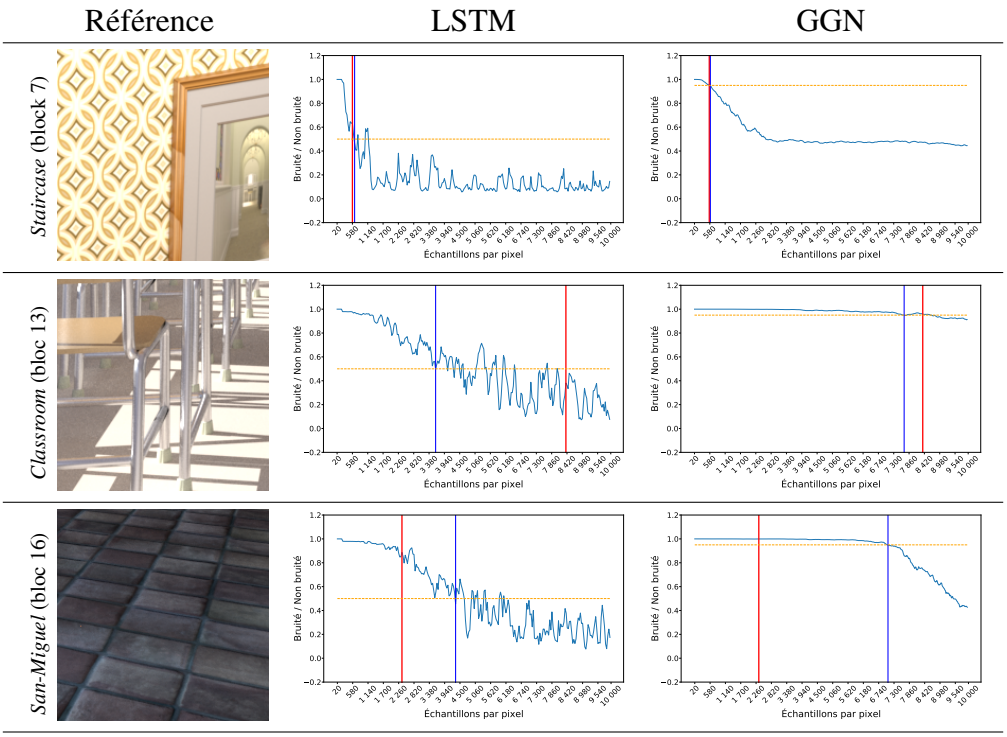
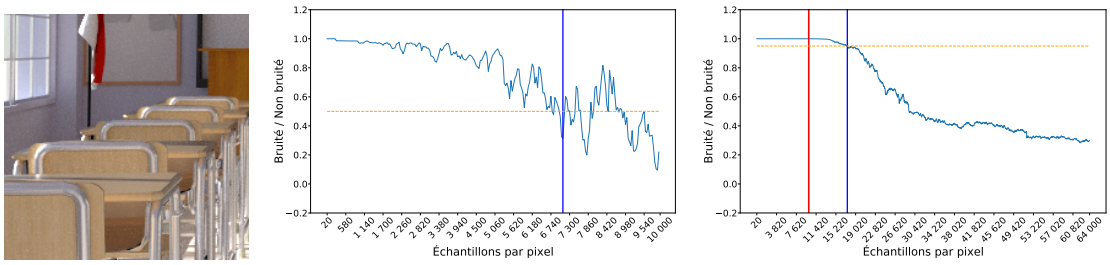


FIGURE 8.7 – Comparaison des prédictions des modèles LSTM et GGN sur certains blocs de 3 images de l'ensemble de test, pour chaque niveau de bruit allant de 20 à 10 000 échantillons. La droite rouge verticale en pointillés représente le seuil humain attendu, la droite orange en pointillés horizontale représente le seuil de prédiction du modèle, et la courbe bleue verticale représente la prédiction du seuil de bruit du modèle une fois le seuil de classification de probabilité atteint. Le seuil de classification **t** du modèle LSTM est fixé à 0.5 et celui du GGN à 0.95, ce qui permet d'éviter une prédiction précoce de la fin du calcul des blocs.

8.5 Discussion

Nous avons proposé une architecture GGN qui permet de générer des données précises de détection du bruit résiduel de MC pour guider la tâche de classification binaire.

Grâce au modèle GGN, il n'est pas nécessaire de se préoccuper de l'utilisation d'une méthode spécifique pour extraire des caractéristiques quantitatives de bruit. Les résultats du modèle GGN proposé suggèrent que la génération automatisée des caractéristiques en entrée d'un modèle de classification relatif à la tâche attendue est pertinente. Par rapport à la précédente approche RNN avec la *SVD-Entropy*, le GGN apporte des résultats tout aussi significatifs, voire plus précis, sur la manière de prédire. En effet, son seuil de



(a) Bloc 9 avec 7 000 échantillons (b) Prédiction RNN pour 10 000 échantillons (c) Prédiction GGN pour 64 000 échantillons

FIGURE 8.8 – Résultat de prédictions du GGN sur le bloc problématique de *Classroom*. Pour les sous-figures (b) et (c), la courbe bleu clair précise les prédictions du modèle, la ligne orange pointillée horizontale précise le seuil de classification du modèle, la ligne bleu foncé verticale le seuil prédit du modèle. La ligne rouge, quant à elle, représente le seuil humain subjectif fixé à 10 000.

classification est plus robuste et « sévère » quant à la présence de bruit, ce qui est un comportement préférable en informatique graphique. En effet, s'arrêter prématurément conduit à un bruit encore perceptible, alors que s'arrêter plus tard permet d'obtenir une qualité identique (ou améliorée), même si le temps de calcul est plus important.

La méthode proposée a été testée ici sur le bruit induit par un algorithme de tracé de chemins de Monte-Carlo. Étant donné la généralité du GGN, la même méthode pourrait être certainement adaptée à d'autres types d'artefacts visuels causés par d'autres méthodes de calcul.

Résumé

Ce huitième chapitre met en avant une architecture qui propose de se séparer de la complexité de recherche de caractéristiques quantitatives de bruit résiduel de Monte-Carlo. À partir d'une architecture où trois réseaux de neurones communiquent entre eux, cette architecture permet la génération de cartes de caractéristiques de bruit afin de distinguer au mieux une image bruitée d'une image non bruitée. Les résultats du modèle GGN sont encourageants, tant sur le plan des précisions que sur les résultats visuels obtenus. Il ne bénéficie d'aucune connaissance explicite sur la nature de bruit, mais arrive au fur et à mesure de l'apprentissage à répondre à la tâche demandée. Ce type d'architecture proposée peut être utilisée pour tout type de bruit à traiter.

Chapitre 9

Problème d'optimisation pour la sélection d'attributs

Introduction

La sélection d'attributs est un problème connu, visant à réduire le nombre de caractéristiques d'une base d'apprentissage et à conserver, parmi celles-ci, celles susceptibles de répondre au mieux à la tâche demandée. Cette sélection permet également de faciliter l'apprentissage du modèle et d'améliorer ses performances. Dans le cadre de cette thèse, les caractéristiques de bruit sont parfois difficilement quantifiables et interprétables. De plus, il est difficile de savoir quelle caractéristique fournit une information importante de discrimination ou non du bruit résiduel présent dans l'image. Ce chapitre vise à introduire une nouvelle méthode expérimentale de sélection d'attributs, en modélisant ce problème comme un problème d'optimisation binaire. L'objectif de cette approche est également de trouver un nombre réduit d'attributs amenant une réduction de la complexité d'apprentissage de certains types de modèles d'apprentissage telle que le SVM.

Les 26 attributs sélectionnés par l'approche (J. CONSTANTIN, BIGAND et al. 2015) représentent un condensé de ce qui est utilisé classiquement pour des problèmes d'analyse d'images (voir section 4.3.2). Nous souhaitons dans ce chapitre étudier la pertinence de l'utilisation de chacun de ces attributs relativement au bruit de MC. Dans ce chapitre, nous introduisons également de nouveaux attributs que nous souhaitons également étudier en plus des 26 attributs initiaux. Le problème soulevé ici est de savoir quels sont les attributs qui sont réellement pertinents pour notre tâche de détection du bruit de MC.

Pour cela, la section 9.1 propose d'énumérer les méthodes de sélection d'attributs classiques et de formaliser ce problème de sélection comme un problème d'optimisation combinatoire binaire. Par la suite, la section 9.2 introduit la notion d'optimisation coû-

teuse et une méthode récente pour permettre une recherche rapide de bonnes solutions (LEPRÊTRE et al. 2019 ; VEREL et al. 2018) dans ce contexte. Les expérimentations réalisées s'appuient sur les travaux de (LEPRÊTRE et al. 2019), appliqué à notre problème de sélection d'attributs, visant à conserver ou non un attribut parmi les attributs disponibles. Les différents attributs de bruit susceptibles d'être exploités ainsi que les résultats obtenus par comparaison aux méthodes classiques de sélection d'attributs, seront présentés dans la section 9.4. Enfin, nous proposons de discuter des premiers résultats obtenus et des travaux futurs envisagés dans la section 9.5.

9.1 Vers un problème d'optimisation

La sélection d'attributs permet de réduire les données d'entrée et d'augmenter les performances d'un modèle. Nous proposons ici une définition de ce problème en un problème d'optimisation visant à chercher une bonne solution de choix de tels attributs.

9.1.1 Méthodes de sélection d'attributs

La sélection d'attributs consiste à obtenir un modèle performant avec un nombre réduit d'attributs en entrée. La diminution de complexité des données en entrée permettant généralement au modèle de mieux interpréter celles-ci et, dans le cadre d'une classification, de les séparer. La figure 9.1 illustre l'idée principale de la sélection d'attributs.

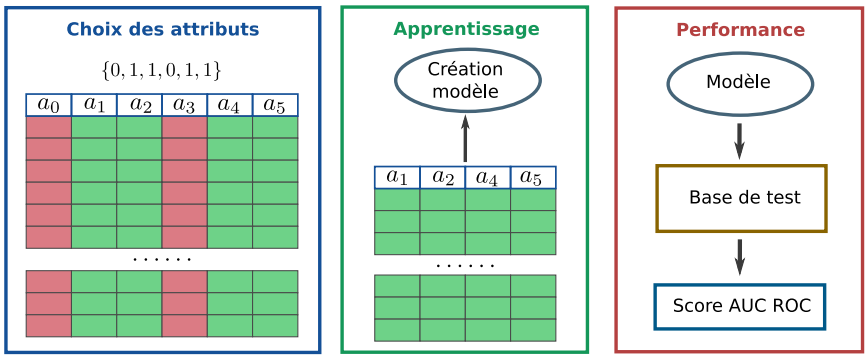


FIGURE 9.1 – Présentation du processus de sélection d'attributs. Après une phase de sélection des attributs les plus pertinents, un modèle est créé par apprentissage. Ce dernier est ensuite évalué sur une base de données de test.

Les approches classiques de sélection d'attributs calculent des coefficients d'importance pour chacun des attributs à partir du modèle appris. À partir de ces coefficients, il est possible de déterminer un seuil associé aux coefficients d'importance les caractéristiques d'entrée qui ont été les plus déterminantes pour l'apprentissage du modèle. On

peut ainsi envisager de sélectionner les k plus importantes caractéristiques, l'objectif étant d'entraîner un modèle avec un nombre réduit d'informations, le rendant plus simple et parfois plus performant. On peut retrouver parmi ces mesures d'importance :

— importance des variables basées sur les impuretés (SCORNET 2020) ;
— importance de la permutation des caractéristiques (ALTMANN et al. 2010) ;

Une méthode connue de réduction de caractéristiques est la *Recursive Feature Elimination* (RFE) (DARST et al. 2018 ; YAN et al. 2015), qui vise à entraîner le modèle puis à se baser sur les coefficients d'importances calculés pour supprimer la caractéristique la moins importante. Ce processus est récursif puisque le modèle apprend de nouveau sans la caractéristique précédente, jusqu'à établissement d'un critère d'arrêt, qui peut-être par exemple le nombre de caractéristiques minimales attendues. Ré-entraîner le modèle à chaque itération permet de mieux interpréter les corrélations entre les différentes caractéristiques et de supprimer à chaque itération la moins pertinente.

9.1.2 Considération d'un espace de recherche

Les approches de sélections précédemment présentées permettent d'obtenir des modèles plus performants et moins complexes. Toutefois, ces approches ont toutes un point commun. Elles ne se basent pas réellement sur l'objectif attendu du modèle, c'est-à-dire, ses performances face à l'inconnu, soit les données non apprises (base de données de test). En effet, ces méthodes de sélection d'attributs estiment des coefficients d'importance uniquement sur les données d'apprentissage. Même en s'appuyant sur de la validation croisée pour éviter le surapprentissage, il reste toujours un risque de ne pas répondre correctement sur les données non apprises. Or, on souhaite généralement posséder un modèle qui répond correctement à la fois sur la base d'apprentissage et celle de test, d'où l'importance de cibler également ses performances sur les données non apprises. L'objectif réel de la sélection d'attributs est par conséquent un modèle performant sur la base de test.

L'idée proposée dans ce chapitre est de modéliser ce problème de sélection d'attributs en un problème d'optimisation. On cherche ainsi à optimiser le choix des attributs sélectionnés pour répondre à l'objectif d'obtenir le modèle le plus performant sur la base de test. La figure 9.2 permet une illustration du problème de sélection d'attributs vu comme un problème d'optimisation et permet une meilleure compréhension des concepts qui vont suivre. De manière plus formelle, résoudre un problème d'optimisation consiste à trouver la meilleure solution pour un critère donné, à partir d'un espace de recherche. L'espace de recherche est défini et contient l'ensemble des solutions possibles, tandis que le critère, également appelé fonction « objectif », évalue la qualité d'une solution. La recherche de la solution optimale s'effectue par la maximisation (resp. minimisation) du critère de qualité. Plus formellement, dans le cadre d'une maximisation, on peut définir :

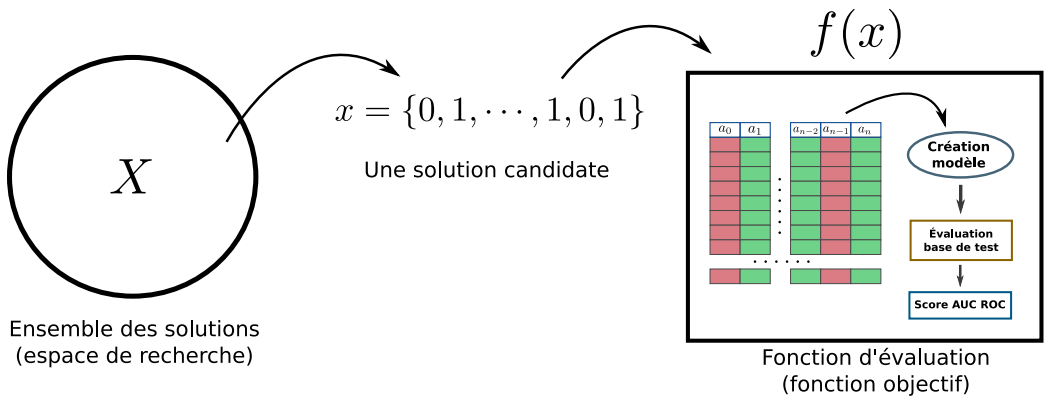


FIGURE 9.2 – Problème de sélection d'attributs vu comme un problème d'optimisation où $f(x)$ représente le score AUC ROC obtenu par le modèle d'apprentissage sur la base de données de test réduite à partir de la solution de filtrage des attributs x.

$$x^* = \arg\max_{x \in \mathcal{X}} f(x) \tag{9.1}$$

où x est une solution candidate au problème, X l'espace de recherche contenant l'ensemble des solutions, f la fonction d'évaluation d'une solution (ou encore fonction objectif) et x^* une solution optimale du problème.

9.1.3 Représentation d'une solution

Une solution candidate est composée de variables de décision :

$$x = (x_1, \ldots, x_n) \tag{9.2}$$

avec n la dimension du problème d'optimisation.

Les algorithmes d'optimisation permettent de rechercher une solution optimale. La procédure consiste à trouver la combinaison des variables de décision qui satisfait au mieux la fonction « objectif ».

On distingue deux grandes familles de problème d'optimisation : ceux dits continus, où chaque variable de décision $x_i \in \mathbb{R}$, et ceux dits combinatoires (ou discret), qui consistent à minimiser (ou maximiser) un critère de qualité à partir d'un espace de recherche fini et dénombrable. Les problèmes combinatoires traitent aussi bien de problèmes académiques, tels que la coloration de graphe (CAO et al. 2019) ou le voyageur de commerce (PURKAYASTHA et al. 2020), que de problèmes du monde réel, comme par exemple, l'affectation des portes d'aéroport (DAŞ et al. 2020).

Dans le cas de notre problème de sélection d'attributs, nous nous trouvons dans un cas particulier de problème d'optimisation combinatoire qui se définit comme un problème d'optimisation pseudo-booléenne. En effet, pour la sélection d'attributs, on souhaite préciser si l'on conserve ou non un attribut parmi les attributs disponibles dans les données. Ainsi, il consiste à optimiser un critère de qualité exprimé par une valeur réelle, notre fonction objectif, à partir d'un espace de recherche composé de variables de décision booléennes :

$$f : \{0,1\}^n \rightarrow \mathbb{R} \tag{9.3}$$

La structure des solutions candidates est une chaîne binaire. Cette représentation permet à chaque variable de décision de s'exprimer (1) ou non (0) lors de l'évaluation de la fonction objectif. Dans notre problème en particulier, il s'agit de la sélection (1) ou non (0) de l'attribut ciblé x_i.

9.1.4 Définition analytique du problème

On distingue généralement les problèmes d'optimisation dits boîtes blanches ou boîtes noires. L'optimisation boîte blanche désigne les problèmes d'optimisation où la fonction objectif est définie formellement (par exemple, les problèmes *MaxSat* [1] (HANSEN et al. 1990), de satisfiabilité de contraintes). C'est-à-dire que l'on peut prévoir le fonctionnement interne, car on connaît les caractéristiques de fonctionnement de l'ensemble des éléments qui le composent. Les méthodes d'optimisation peuvent donc exploiter les caractéristiques algébriques du problème pour le résoudre efficacement.

L'optimisation boîte noire désigne, au contraire, les problèmes d'optimisation où aucune information sur la définition de la fonction objectif n'est disponible. Cela peut également désigner des problèmes pour lesquels la définition de la fonction objectif est connue (par exemple, le code source d'un simulateur de réacteur nucléaire (MUNIGLIA et al. 2017)), mais difficilement prédictible et exploitable par des études analytiques. Dans ce cas, la qualité d'une solution candidate ne peut être évaluée que par une simulation ou un calcul, généralement coûteux en temps et en ressource.

Le problème de sélection d'attributs peut-être considéré comme appartenant à la catégorie de l'optimisation boîte noire. En effet, on souhaite obtenir un score de qualité du modèle une fois appris sur la base de données de test, ce qui pourrait correspondre au score AUC ROC obtenu sur cette base de test.

1. Un problème *MaxSat* consiste à déterminer le nombre maximum de clauses d'une formule booléenne donnée en forme normale conjonctive, qui peuvent être rendues vraies par une affectation de valeurs de vérité aux variables de la formule.

9.1.5 Métaheuristiques

Dans le cadre d'études des problèmes d'optimisation pseudo-booléen, il est important de préciser que la taille des solutions possibles de l'espace de recherche est 2^n. Ce qui peut impliquer, avec un n très grand, un très grand nombre de solutions à envisager.

Étant donné la complexité de certains problèmes, notamment relatif à la taille de leur espace de solutions à parcourir, certains algorithmes appelés *métaheuristiques* sont exploités pour la recherche de solutions. Ces métaheuristiques peuvent être employées pour la résolution d'une large variété de problèmes d'optimisation, et s'avèrent particulièrement exploitées pour l'optimisation de problèmes pour lesquels aucune méthode classique plus efficace n'est connue. Toutefois, elles ne garantissent pas de trouver la solution optimale (bien qu'elles puissent parfois la découvrir), mais peuvent en un temps fini, trouver une bonne solution.

Généralement, les métaheuristiques se distinguent selon deux catégories : les algorithmes à une solution et les algorithmes à population de solutions (TALBI 2009). Dans ce cadre d'études, nous traitons des algorithmes évolutionnaires avec des populations de solutions, inspirées des théories sélectionnistes, telle que la théorie de l'évolution de Darwin (SMITH et al. 1993).

9.1.6 Opérateurs de variation

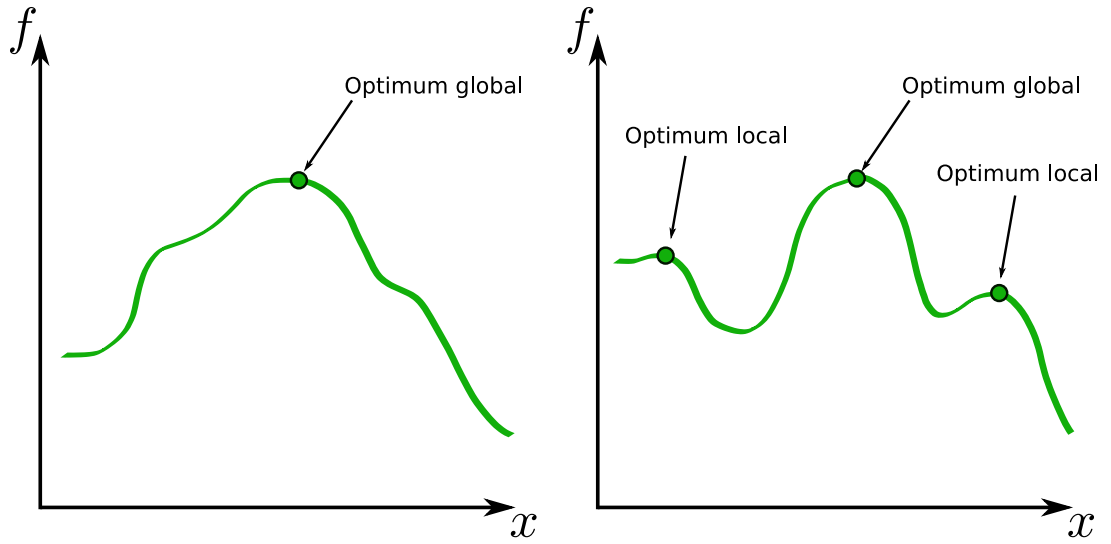
Les opérateurs altèrent les propriétés internes d'une solution x pour en former une autre x', également appelée solution voisine. La variation permet ainsi le parcours de l'espace de recherche, puisqu'elle permet la recherche au sein du voisinage de la solution initiale. Il existe une multitude d'opérateurs de variation, qui dépendent de la représentation numérique de la solution

Dans un espace de recherche booléen, on peut distinguer plusieurs moyens de générer une nouvelle solution voisine, par exemple :

— la **mutation** : on modifie un seul attribut de la solution x, ce qui revient à utiliser un opérateur de variation d'inversion d'une des variables ;
— la **permutation** : un opérateur de variation effectue l'échange de deux variables de décision choisies aléatoirement dans une solution ;
— le **croisement** : (*crossover* en anglais) est un opérateur appliqué entre deux solutions issues de la population afin d'en concevoir une troisième qui hériterait de la combinaison des caractéristiques des deux autres.

9.1.7 Recherche de solutions : compromis exploration et exploitation

La robustesse d'une métaheuristique pour la résolution d'un problème d'optimisation dépend de sa capacité à explorer et à exploiter efficacement l'espace de recherche. Explorer consiste à parcourir l'espace de recherche en largeur, en évaluant des solutions dont les représentations sont éloignées des unes des autres (voisinage éloigné), contrairement à l'exploitation, qui vise à exploiter une solution et parcourir son voisinage. En effet, une bonne solution, avec un score de performance intéressant serait susceptible de fournir des voisins d'un score supérieur, mais cela peut amener un risque d'optimum local dû à la nature du problème (voir figure 9.3). Les opérateurs comme ceux basés sur l'inversion d'une des variables de la solution et de permutations sont plutôt voués à exploiter la solution actuelle. Au contraire, les opérateurs de croisements de solutions permettent une variation considérable des attributs de la solution, plutôt orientés donc sur l'exploration de l'espace de recherche.



(a) Espace de recherche simple composé d'un seul candidat optimum global et d'aucun optimum local

(b) Espace de recherche convexe composé de plusieurs optima locaux et un seul candidat optimum global

FIGURE 9.3 – Illustration théorique de différents espaces de recherche de solutions

Il est généralement difficile de connaître les opérateurs à appliquer relativement au problème et de connaître leur stratégie d'application. Vaut-il mieux explorer davantage au départ, puis exploiter ? Les stratégies habituelles visent effectivement à plutôt explorer puis exploiter. Mais cela peut être conditionné par un taux d'exploration qui varie au cours de la recherche de solutions (FRUIT 2019).

Une méthode consiste à bénéficier de l'apprentissage par renforcement qui, au fur et à mesure de l'algorithme, va apprendre et essayer d'estimer quel opérateur est le plus à même de donner de meilleurs résultats pour l'évaluation courante. Des stratégies de recherche d'équilibre de choix d'opérateurs optimaux ont été définis pour les problèmes de *bandits manchots* (LAI et al. 1985), qui impliquent k bras de bandits avec des probabilités de récompense inconnues. À chaque étape, un bras doit être sélectionné pour être tiré. L'objectif est la maximisation des gains cumulés possibles au cours de toutes les étapes S. Une technique efficace pour résoudre ce problème est l'algorithme d'*Upper Confidence Bound* (UCB). Il est utilisé pour sélectionner le bras à chaque étape de la manière suivante :

$$\max_{i \in k} \left(\hat{r}_i + C \cdot \sqrt{\frac{2 \ln S}{s_i}} \right) \tag{9.4}$$

avec $\hat{r}_i$ la récompense empirique moyenne observée du bras i et s_i le nombre de fois (occurrences) que le bras i a été sélectionné durant les S étapes. Cette équation cherche à assurer un compromis entre l'exploitation et l'exploration, équilibré par une constante C. L'exploitation tend à sélectionner le bras qui a la récompense moyenne optimale, tandis que l'exploration tend à sélectionner le bras qui a été le plus rarement choisi.

L'utilisation de l'algorithme UCB est fortement répandue dans le domaine de l'optimisation, en particulier lorsque le processus d'optimisation peut être réalisé avec divers opérateurs de variation dont le potentiel est inconnu (J. LIU et al. 2017). À chaque itération du processus d'optimisation, c'est-à-dire à chaque fois qu'une variable de décision est altérée pour une nouvelle évaluation, la différence de la fonction objectif est calculée comme suit :

$$r = f(x) - f(x') \tag{9.5}$$

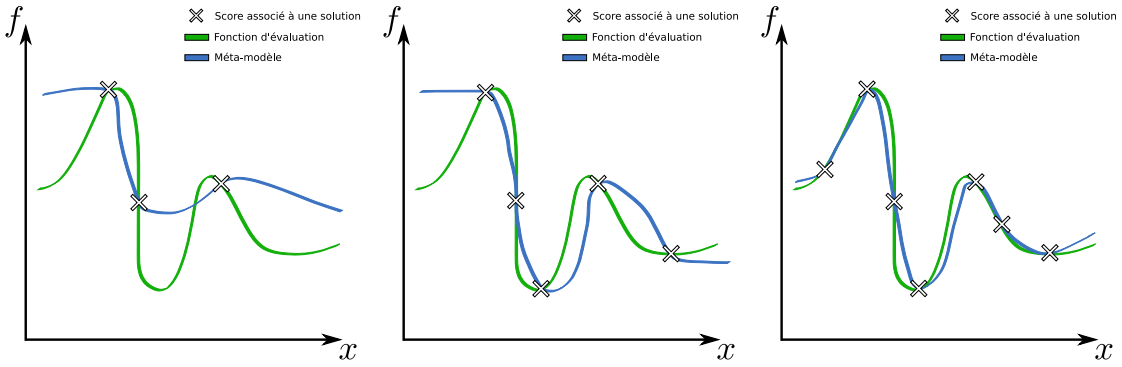
avec x la solution actuelle et x' la nouvelle solution obtenue après application de l'opérateur sur x. Par conséquence $\hat{r}_i$, le terme d'exploitation de la formule de l'équation 9.4 de l'UCB est représenté par la somme empirique des valeurs r_i obtenues. Nous verrons par la suite dans la section 9.4 que cette stratégie de choix d'opérateur a été exploitée dans le cadre de nos expériences.

9.2 Optimisation assistée par méta-modèle

Parfois, la fonction d'évaluation peut être une fonction connue mais difficilement interprétable et généralement coûteuse, comme cela est le cas lorsque celle-ci traite les résultats d'un simulateur de réacteur nucléaire (MUNIGLIA et al. 2017) ou encore de flux de trafic urbain (LEPRÊTRE, FONLUPT et al. 2018).

9.2.1 Définition d'un modèle de substitution (ou surrogate)

Lorsque l'évaluation d'une solution prend quelques minutes, voire quelques heures, il est difficilement réalisable de parcourir une partie significative de l'espace de recherche et de fournir une bonne solution. Pour faire face à ce problème, une technique classique consiste à apprendre un modèle de substitution (*surrogate* en anglais), également appelé méta-modèle. La substitution revient à formuler des modèles mathématiques rapides à évaluer qui approchent la fonction objectif, afin de limiter le recours à son évaluation. La figure 9.4 illustre le principe de l'apprentissage d'un tel méta-modèle lors de la recherche de solutions.



(a) Méta-modèle ayant appris sur un petit nombre de solutions

(b) Méta-modèle ayant appris sur un nombre assez conséquent de solutions

(c) Méta-modèle ayant appris sur un grand nombre de solutions

FIGURE 9.4 – Représentation de l'évolution des performances d'un méta-modèle au cours de la recherche de solutions

Ce modèle doit être suffisamment expressif pour refléter la sophistication de la fonction à substituer. Cependant, il est préférable qu'il conserve une faible complexité calculatoire, afin de faciliter son apprentissage lorsque peu d'échantillons de la fonction objectif originale sont disponibles. C'est à partir de ce méta-modèle qu'il est possible de définir une fonction d'acquisition qui permettra, après un parcours rapide d'un ensemble de solutions, de fournir une bonne solution. L'objectif de la fonction d'acquisition consiste à guider la recherche vers une solution prometteuse, tout en maintenant l'équilibre entre l'exploration qui augmente la qualité globale du modèle de substitution, et l'exploitation qui pousse le modèle vers des solutions de haute qualité. À noter que la nature du méta-modèle est généralement un modèle de régression de la fonction objectif réelle. L'algorithme 1 met en avant l'utilisation d'un tel modèle dans une métaheuristique classique de recherche de solution. À chaque nouvelle évaluation réelle, le modèle M est mis à jour pour permettre d'appliquer une fonction d'acquisition (généralement une recherche locale de solution) et obtenir une nouvelle solution x.

Algorithme 1 : Métaheuristique assistée par un méta-modèle

Result : Meilleure solution trouvée x'

initialiser : $S \leftarrow$ solutions initiales $\{(x, f(x)), ...\}$;

while *critère d'arrêt* **do**
  $M \leftarrow$ construire un méta-modèle M depuis l'ensemble S de solutions ;
  $x \leftarrow$ recherche locale en utilisant le méta-modèle M ;
  Évaluer x en utilisant f ;
  Mettre à jour x' en fonction de x ;
  $S \leftarrow S \cup \{x, f(x)\}$;
end
return x';

9.2.2 Choix du modèle de substitution

Il existe plusieurs approches de méta-modélisation combinatoire : l'une des plus connues utilise une approche basée sur des processus gaussiens, également appelée *Krigeage* (ZAEFFERER, STORK et BARTZ-BEIELSTEIN 2014 ; ZAEFFERER, STORK, FRIESE et al. 2014). Cette approche conçoit la fonction objectif à substituer comme la réalisation d'un processus gaussien. Une autre approche ayant suscité l'intérêt a été proposée par (BAPTISTA et al. 2018). Elle utilise un modèle à base de fonctions multilinéaires nommé *Bayesian Optimisation of Combinatorial Structures* et considère principalement des polynômes multilinéaires quadratiques.

La base de fonctions de (WALSH 1923) comporte une base normale et orthogonale de fonctions discontinues. Elles peuvent être employées comme une base de représentation normale et orthogonale des espaces de recherche pseudo-booléens dans le cadre de l'optimisation de structures combinatoires. Formellement, pour tout entier ℓ écrit sous une forme binaire, la fonction de Walsh φ_ℓ est définie telle que :

$$\varphi_\ell(x) = \left(-1\right)^{\sum_{i=1}^{n} \ell_i x_i} \tag{9.6}$$

avec $\ell_i \in [-1, 1]$ et x_i qui représentent la $i^{\text{ème}}$ variable dans les chaînes binaires ℓ et x.

Une approche récente propose l'utilisation de la base des fonctions de Walsh pour l'élaboration de méta-modèles spécialement dédiés à l'optimisation pseudo-booléenne (VEREL et al. 2018). Les auteurs montrent que la fonction pseudo-booléenne à approcher peut être substituée par le polynôme :

	n					
L	10	15	20	25	50	100
0	1	1	1	1	1	1
1	11	16	21	26	51	101
2	56	121	211	326	1 276	5 051
3	176	576	1351	2,626	20,876	166,751

TABLEAU 9.1 – Nombre de coefficients pour les décompositions multilinéaires ou de Walsh, en fonction de la dimension du problème n et de l'ordre L.

$$W_L(x) = \sum_{\ell \text{ s.t. } o(\varphi_\ell) \leqslant L} a_\ell \varphi_\ell(x) \tag{9.7}$$

avec o l'ordre de la fonction de Walsh, c'est-à-dire le nombre de variables égales à 1 dans la représentation binaire de l'entier ℓ et a le vecteur de poids associé au polynôme.

Le développement du polynôme est ainsi restreint à un ordre L. Par exemple, pour l'ordre 2, on obtient :

$$W_2(x) = a_0 + \sum_{i=1}^{n} a_i (-1)^{x_i} + \sum_{i<j \in N} a_{ij} (-1)^{x_i + x_j} \tag{9.8}$$

Comme mentionné par Leprêtre (LEPRÊTRE 2020), l'approximation de la valeur des coefficients est réalisée par des techniques classiques de régression linéaire parcimonieuse. En effet, la décomposition de l'équation 9.7 peut être interprétée comme un modèle linéaire où les prédicteurs prennent les valeurs des fonctions de Walsh. Il est évident que le potentiel d'approximation de cette formulation croît lorsque l'ordre L augmente, aux dépens du nombre de termes dans le développement du polynôme, comme présenté dans le tableau 9.1. Aussi, les auteurs emploient les techniques *Lars* ou *Lasso* (HASTIE et al. 2019), qui consistent à minimiser le nombre de coefficients non-nuls dans la formulation du modèle. Les résultats mis en avant par l'utilisation de cette base de fonction confirment son intérêt pour les problèmes combinatoires binaires assistés par méta-modèle.

9.3 Procédure expérimentale

Les attributs de quantification du bruit étudiés dans les images de synthèse sont détaillés dans un premier temps dans cette section. Dans un second temps, la fonction d'évaluation d'une solution est décrite avant de développer la procédure envisagée de l'exploration de l'espace de recherche assistée par méta-modèle pour ce problème. Étant

donné les résultats apportés par l'utilisation des bases de fonction de Walsh (LEPRÊTRE et al. 2019), nous les exploitons également dans le cadre de notre problème combinatoire binaire.

9.3.1 Définition des attributs étudiés

Les attributs étudiés concernant la quantification du bruit dans les images de synthèse sont ici au nombre de 32. Les 26 premiers sont ceux extraits par l'approche de (J. CONSTANTIN, BIGAND et al. 2015). Pour rappel, 13 filtres différents ont été utilisés et appliqués à l'image (voir section 4.3.2). Ensuite, la moyenne et l'écart type sont extraits de chacune des images obtenues. Au contraire de leur approche visant à utiliser une image de référence réduite, ici nous nous basons sur les 26 attributs extraits directement de l'image en cours de calcul, présentant un certain niveau de bruit.

À ces premiers 26 attributs, nous ajoutons deux attributs issus des informations spatiales décrivant la complexité d'une image à partir du filtre de Sobel (ROUSSELOT et al. 2019) :

$$\mathrm{SI} = \mathrm{std}[\mathrm{Sobel}(I)] \tag{9.9}$$

où I représente l'image courante. Nous calculons cet indicateur pour des tailles de fenêtre de filtre de Sobel différentes 3×3 et 5×5. Le filtre de Sobel permet notamment de mettre en avant les contours des différents objets présents dans une image (ROUSSELOT et al. 2019). L'utilisation de cet indicateur a pour objectif de prendre en considération la structure de l'image, sachant que le bruit présent peut aussi être considéré comme un bord d'objet détecté.

Ensuite, nous ajoutons également deux caractéristiques liées à la notion de disparité des valeurs de l'image :

— la caractéristique de la complexité de Kolmogorov $K(I)$. Elle correspond approximativement, comme proposé par (HAO et al. 2020), à la taille du fichier de l'image compressée. En effet, la taille de l'image compressée peut donner des informations sur la qualité de cette image et donc du bruit résiduel encore présent dans cette image. Un bruit résiduel est synonyme de variation forte des valeurs de pixels proches et par conséquent d'un besoin de stockage d'informations plus important.
— la *SVD-Entropy* donnée par l'équation 7.3 de la section 7.2.

Enfin, les deux dernières caractéristiques de bruit des 32 considérées sont l'écart-type et la moyenne des valeurs de luminance de l'image, c'est-à-dire, du canal L de la décomposition Lab comme détaillé dans la section 7.2.

9.3.2 La sélection d'attributs par l'optimisation assistée par méta-modèle

Le problème d'optimisation de sélection d'attributs envisagé doit être décrit plus précisément comme un problème d'optimisation coûteuse. En effet, il faut préciser que la fonction d'évaluation de sélection ou non d'attributs est relative à la performance du modèle d'apprentissage sur la base de test une fois le modèle appris avec les attributs sélectionnés par la solution x. Le score est plus précisément l'AUC ROC pour estimer la capacité du modèle à répondre à la tâche de classification du bruit dans les images. Entraîner un modèle sur un ensemble de données, quelle que soit la nature du problème, puis obtenir son score de performance sur la base de test, peut générer un temps de calcul de l'ordre de quelques minutes à plusieurs heures.

Le problème de dimensionnalité pour l'apprentissage d'un modèle de type SVM (voir section 7.1) ne nous permet pas, dans le temps imparti de la thèse et de cette étude, d'exploiter ce modèle pour la recherche de meilleurs attributs. Il serait déjà très long (voire impossible comme déjà mentionné) et coûteux d'entraîner un seul modèle et d'obtenir son score AUC ROC sur la base de test. Utiliser un tel modèle pour une recherche de meilleurs attributs avec le formalisme d'exploration de solutions au travers d'une métaheuristique est encore moins envisageable. C'est pourquoi nous avons opté pour un modèle basé sur les forêts d'arbres décisionnels (ou forêts aléatoires de l'anglais *random forest classifier*) (BREIMAN 2001) plus simples et donnant un temps beaucoup moins long pour l'apprentissage (de l'ordre de quelques minutes généralement) sur les données souhaitées. Ce type de modèle fait partie des techniques d'apprentissage automatique qui permet la classification de données. C'est un algorithme qui combine les concepts de sous-espaces aléatoires et de *bagging*. L'algorithme des forêts d'arbres décisionnels effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents. La prédiction de la classification finale de la forêt aléatoire est alors un simple vote majoritaire des différents estimateurs (arbres décisionnels). La figure 9.5 illustre le fonctionnement d'un tel modèle.

L'ensemble de données des 32 attributs est généré sur les mêmes images et les mêmes blocs que ceux utilisés pour l'approche de la section 7.2 pour les bases d'apprentissage et de test. C'est-à-dire, pour chaque scène disponible de la base d'apprentissage (BUISINE, DELEPOULLE, SYNAVE et al. 2021), les mêmes 12 blocs sont utilisés pour l'apprentissage et les blocs restants pour la base de test du modèle. Nous fixons dans le cadre de cette étude un nombre d'estimateurs de l'ordre de 500 arbres décisionnels pour le modèle de forêt aléatoire constituant la fonction d'évaluation d'une solution x.

Nous basons nos travaux sur le *framework* de Surrogate à base de Walsh nommé WSaO pour *Walsh functions Surrogate-assisted Optimization algorithm* proposé par (LEPRÊTRE et al. 2019). Les travaux mis en avant par les auteurs montrent les perfor-

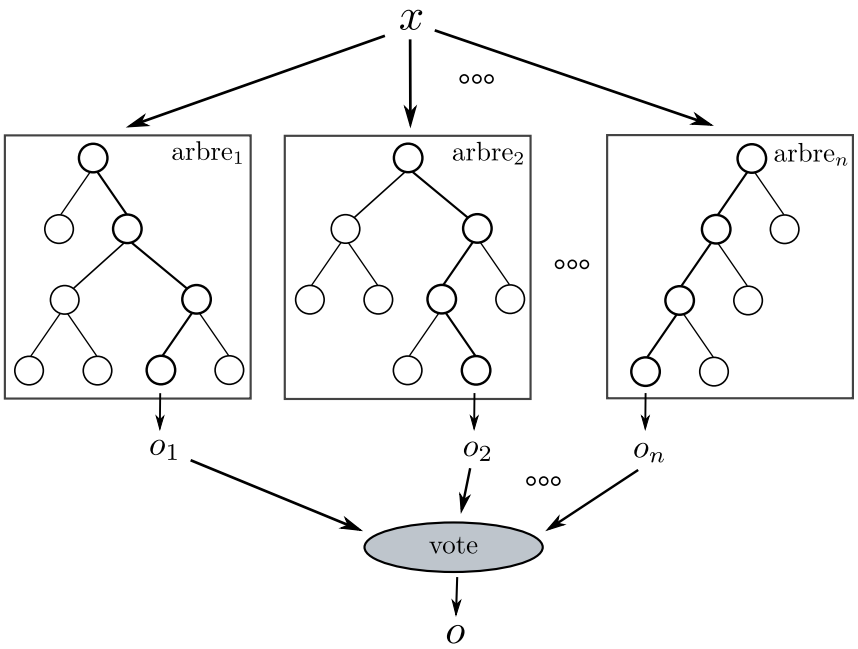


FIGURE 9.5 – Illustration d'un modèle de type forêt aléatoire composé de différents arbres

mances et l'intérêt des bases de fonctions de Walsh sur des problèmes à nature complexe. Nous appliquons ce *framework* pour la recherche des meilleurs attributs parmi n attributs possibles pour répondre au mieux à la complexité de la fonction d'évaluation. Même si le temps d'apprentissage d'une forêt aléatoire est relativement rapide (de l'ordre de quelques dizaines de secondes), cela reste coûteux d'un point de vue optimisation. En effet, l'espace de recherche est composée de 2^{32} solutions ce qui ne rend pas possible la recherche rapide de bonnes solutions en un temps raisonnable.

9.3.3 Formulation de la métaheuristique assistée par méta-modèle

L'algorithme proposé pour cette étude est basé sur une population de solutions. Il exploite également un méta-modèle à base de fonctions de Walsh pour apprendre la fonction d'évaluation réelle qui consiste en l'évaluation de performance sur des données inconnues du modèle de forêt aléatoire. Les trois types d'opérateurs de variation de solutions cités précédemment sont utilisés et un algorithme UCB est exploité pour permettre d'utiliser l'opérateur le plus adéquat permettant une recherche locale de solution. Au final, quatre opérateurs sont définis :

— un de mutation : inverse la valeur d'une variable de la solution ;
— un de permutation : permute aléatoirement deux variables de décisions dans la solution ;

— un opérateur de croisement : génère une nouvelle solution à partir de la moitié des valeurs d'une première solution et de l'autre partie d'une seconde solution de la population ;
— un opérateur de croisement : coupe aléatoirement la première solution et la seconde solution de la population afin d'en former une nouvelle (point de croisement aléatoire).

L'algorithme 2 présente l'approche réalisée. Premièrement, l'ensemble P de la population de solutions est initialisé avec la vraie fonction d'évaluation, c'est-à-dire l'apprentissage d'un modèle de forêt aléatoire et l'obtention de son score AUC ROC sur la base de test. L'ajout des solutions initiales de P à l'ensemble de solutions de S est réalisé. S étant l'ensemble des solutions utilisées pour apprendre le méta-modèle. Il est défini par s, un nombre minimum de solutions nécessaires avant de réaliser une phase de construction d'un méta-modèle M (méta-modèle qui approche la vraie fonction et va assister la recherche de solutions). Ensuite, pour chaque solution de P, une nouvelle solution est générée après sélection du meilleur opérateur par l'algorithme UCB. Enfin, le modèle M est exploité lors d'une recherche locale pour obtenir rapidement une nouvelle solution prometteuse x.

L'apprentissage d'un méta-modèle à chaque évaluation d'une solution par la vraie fonction (impliquant l'ajout de la nouvelle solution dans S), n'est pas forcément judicieux. Cet apprentissage impose un temps de calcul qui n'est pas négligeable. Pour remédier à cela, nous introduisons le paramètre h, qui permet de ne pas obtenir à toutes les évaluations réelles un nouveau méta-modèle de régression M. Ainsi, le score de performance du méta-modèle de régression M relativement à S est évalué avec le R^2 (voir équation 3.3), ce qui permet d'entraîner le méta-modèle tous les $t = h \times R^2$. Si le modèle n'est pas assez performant, il est de nouveau construit dès qu'une nouvelle solution est évaluée par la vraie fonction et ajoutée dans S. Dans le cas contraire, on estime que le modèle est assez confiant pour être exploité pour plusieurs recherches locales avant d'être entraîné de nouveau.

Il est important de préciser que l'algorithme UCB est également appliqué au sein de la recherche locale (*local search* en anglais), où le méta-modèle M est utilisé comme fonction d'évaluation. Toutefois, seuls les opérateurs de mutations et de permutations peuvent être sélectionnés pour effectuer une recherche locale. En effet, l'objectif d'une recherche locale est de réaliser une recherche au voisinage de la solution. L'intérêt d'appliquer un opérateur de croisement n'est pas judicieux, car il impliquera une exploration forte. Au contraire, la recherche locale a pour but avant tout d'exploiter la solution de départ fournie, dans notre cas $p[i]'$ afin d'obtenir une solution performante estimée à partir de son voisinage.

Un package Python nommé Macop pour *Minimalist And Customisable Optimisation Package* a été développé et publié dans le cadre de ces travaux dans *The Journal of*

Algorithme 2 : Méta-heuristique avec population intégrant un méta-modèle à base de fonctions de Walsh

Result : Meilleure solution trouvée x'

initialiser : h, nombre d'évaluations réelles avant nouvel apprentissage du méta-modèle ;
initialiser : $P \leftarrow$ solutions initiales $\{(x, f(x)), ..., (x_p, f(x_p))\}$;
initialiser : $S \leftarrow P$;
initialiser : $s \leftarrow |P|$, nombre minimum de solutions avant apprentissage du méta-modèle ;

$nEvaluations \leftarrow |P|$;

while $nEvaluations < s$ **do**
    initialiser aléatoirement x;
    $S \leftarrow S \cup \{x, f(x)\}$;
    Mettre à jour x' en fonction de x ;
    $nEvaluations \leftarrow nEvaluations + 1$;
end

$M \leftarrow$ construire un méta-modèle M depuis l'ensemble S de solutions ;
$nRechercheLocale = 0$;

while $nEvaluations <$ *critère d'arrêt* **do**
    forall $p[i] \in P$ **do**
        $operator = UCB()$;
        $p[i]' \leftarrow operator(p[i])$;
        $x \leftarrow$ rechercheLocale$(M, p[i]')$;
        $nRechercheLocale \leftarrow nRechercheLocale + 1$;
        Évaluer x en utilisant f ;
        Mettre à jour x' en fonction de x ;
        Mettre à jour $p[i]$ en fonction de x ;
        $S \leftarrow S \cup \{x, f(x)\}$;
        if $nRechercheLocale \geq t$ **then**
            $M \leftarrow$ construire un méta-modèle M depuis l'ensemble S ;
            $t \leftarrow h \times R^2$ de M ;
            $nRechercheLocale \leftarrow 0$;
        end
        $nEvaluations \leftarrow nEvaluations + 1$;
    end
end
return x';

Open Source Software (BUISINE, DELEPOULLE et RENAUD 2021). Macop propose la résolution de problèmes d'optimisation discrets. L'objectif est de permettre à un utilisateur d'exploiter la structure de base proposée par ce package pour résoudre le problème souhaité. L'intérêt est qu'il peut rapidement s'abstraire des complications liées à la manière d'évaluer, de comparer, de sauvegarder la progression de la recherche de bonnes solutions, mais plutôt, et se concentrer si nécessaire sur son propre algorithme. Dans les développements des travaux de ce chapitre, ce package a notamment permis de faciliter l'adaptation de l'utilisation d'un méta-modèle au sein du processus de recherche.

9.4 Résultats et comparaisons

Pour les expérimentations, plusieurs paramètres ont été fixés afin de limiter le nombre de possibilités et de diminuer le temps de calculs qui, pour une seule exécution de l'algorithme, nécessite plusieurs heures.

9.4.1 Jeux de paramètres testés

Tout d'abord, la taille de la population solutions de notre méta-heuristique a été fixée à $|P| = 20$ et le nombre d'évaluations réel maximal à 1000. Pour les expérimentations assistées par méta-modèle, les ordres $L \in [1, 2]$ des fonctions de Walsh ont été testés et le nombre d'évaluations lors d'une recherche locale avec le méta-modèle M a été fixé à $LS \in [100, 500, 1000]$ afin de voir l'impact de la profondeur de cette recherche (exploitation). L'ordre 3 entraînant un trop grand nombre de paramètres n'a pas été envisagé, car la construction d'un tel modèle pouvait impacter considérablement en temps la recherche de solutions. Enfin, le nombre de solutions minimales pour entraîner le méta-modèle a été fixé à $s = 50$ et le paramètre t impactant les conditions d'entraînement de ce modèle à $t = 10$.

Il est important de préciser que les comparaisons sont effectuées sur le nombre réel d'évaluations, étant donné que l'appel au modèle M pour évaluation d'une solution est quasi instantané. Ainsi la recherche locale avec le modèle M est réalisée en un temps très court comparé à une seule évaluation réelle d'une solution. Pour chacun des jeux de paramètres envisagés, cinq exécutions ont été réalisées pour amener une certaine stabilité des résultats.

9.4.2 Résultats des expérimentations

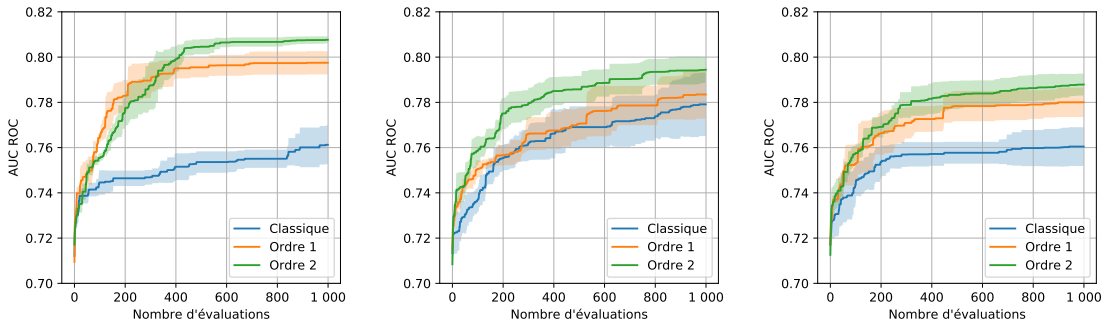
9.4.2.1 Comparaisons des performances

La figure 9.6 propose de comparer les résultats obtenus en fonction du paramètre LS pour les différentes valeurs d'ordre L de la base de fonctions de Walsh envisagées.

Comme mentionné précédemment, cinq exécutions ont été réalisées et la moyenne des meilleures solutions obtenues en fonction du nombre d'évaluations est représentée pour chacune des courbes. Cela a pour but de permettre une meilleure représentation de la stabilité des résultats et donc leur fiabilité. Un intervalle de confiance illustre également la stabilité de la méthode pour les paramètres sélectionnés.

Pour les trois sous-figures proposées, la méthode classique (sans assistance par méta-modèle) a été exécutée pour $LS \in [100, 500, 1\ 000]$ afin de proposer une comparaison des approches bénéficiant d'un méta-modèle pour les paramètres d'ordre $L \in [1, 2]$. Ainsi, quand $LS = 100$, la méthode classique réalise 10 recherches locales, avec $LS = 500$, deux recherches locales et $LS = 1\ 000$, une unique recherche locale.

Chacune des sous-figures met en avant les performances des recherches assistées par méta-modèle pour les différentes valeurs de $LS \in [100, 500, 1\ 000]$. On peut noter rapidement que pour chaque valeur de LS, la méthode classique semble moins performante, ce qui confirme l'intérêt de l'utilisation de tels méta-modèles pour ce type de problèmes de sélection d'attributs. Même si la méthode classique avec $LS = 500$ est en moyenne proche des résultats des méthodes assistées, elle reste encore inférieure. De plus, pour toutes les valeurs de LS, l'ordre $L = 2$ semble être le plus intéressant, indiquant et mettant en avant une meilleure capacité du méta-modèle à interpréter la complexité du modèle de forêt aléatoire. L'ordre 2 bénéficiant de plus de paramètres pour réaliser sa régression et pour comprendre le problème, semble ainsi mieux prédire les scores AUC ROC attendus de la forêt aléatoire.

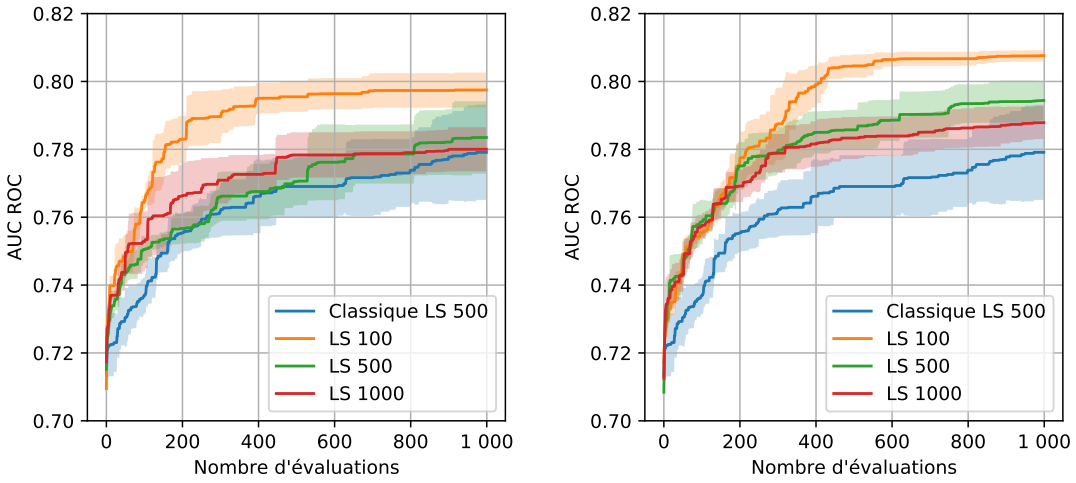


(a) $LS = 100$ (b) $LS = 500$ (c) $LS = 1\ 000$

FIGURE 9.6 – Comparaisons de résultats de recherche de solutions avec méta-modèle en fonction du paramètre LS. Chaque courbe représente la meilleure solution basée sur son score AUC ROC obtenue au cours de la recherche (moyenne des 5 exécutions) et donc du nombre d'évaluations.

La figure 9.7 propose une représentation différente des résultats où les deux sous-figures affichent chacune les performances des méthodes en fonction du paramètre L. Chaque valeur de LS est ainsi plus facilement comparable pour les méthodes assistées. La

méthode classique ayant donné les meilleurs résultats ($LS = 500$) est également affichée. Grâce à cette représentation, on peut noter un point très important indiquant le fait qu'une recherche locale composée de moins d'itérations, c'est-à-dire dans notre cas $LS = 100$, apporte de meilleurs résultats finaux, mais aussi de meilleures solutions plus rapidement au cours de la recherche. Il est possible que le méta-modèle qui intègre en partie dans ses connaissances la complexité du modèle de forêt aléatoire, n'arrive toutefois pas à la prendre en considération totalement. Cela signifie que le méta-modèle surapprend relativement aux connaissances actuelles (solutions obtenues) de la recherche et peut inférer pour une recherche locale avec un paramètre LS grand, une solution obtenue de qualité moindre qu'avec un paramètre LS plus petit. Le biais apporté par le fait que l'ensemble des solutions possibles ne sont pas connues, nous indique qu'il ne faut pas préférer l'exploitation du méta-modèle, mais plus un compromis d'utilisation de celui-ci. Du moins, pour ce genre de problème en particulier qu'est la sélection d'attributs.



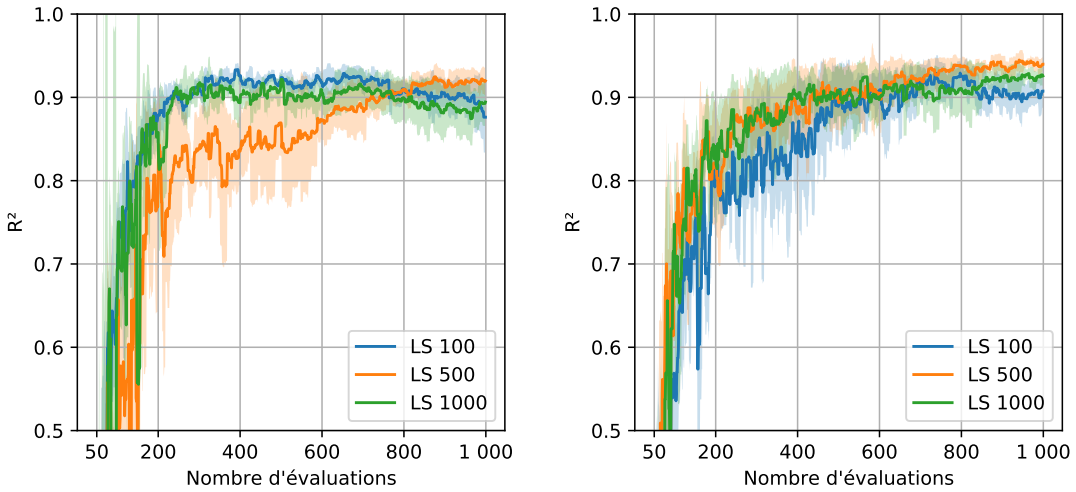
(a) $L = 1$ (b) $L = 2$

FIGURE 9.7 – Comparaisons de résultats de recherche de solutions avec méta-modèle en fonction du paramètre d'ordre de fonctions de Walsh L. Chaque courbe représente la meilleure solution basée sur son score AUC ROC obtenue au cours de la recherche et donc du nombre d'évaluations.

9.4.2.2 Apprentissage du méta-modèle

La figure 9.8 permet de comparer les scores d'apprentissage R^2 pour les différents jeux testés. Les deux sous-figures proposées détaillent chacune indépendamment l'ordre $L = 1$ et l'ordre $L = 2$ des bases de fonctions de Walsh exploitées par le méta-modèle. Il est important de préciser que ces scores ne sont enregistrés qu'à partir de l'évaluation

50, correspondant au nombre de solutions minimales évaluées avant apprentissage du méta-modèle. La sous-figure 9.8a, qui traite des résultats pour $L = 1$, indique les performances d'apprentissage des méta-modèles, et cela, indépendamment la valeur de LS. On peut y remarquer une forte fluctuation des scores R^2 des méta-modèles malgré un grand nombre d'évaluations. En effet, quand $LS = 500$, le modèle semble seulement apprendre correctement à partir de l'évaluation 500. Pour $LS = 100$ et $LS = 1\ 000$, les performances décroissent après un nombre d'évaluations de 700. Au contraire, pour la sous-figure 9.8b où l'ordre $L2$ est représenté, pour toutes les valeurs de LS, l'apprentissage se stabilise lentement quand le nombre d'évaluations est supérieur à 800. Un dernier point qui apparaît, concerne le fait que pour $LS = 100$, le score de performance semble moindre que pour les autres valeurs de LS. Cela peut laisser penser que le modèle surapprend moins, permettant une meilleure généralisation de prédiction du score AUC ROC comme analysé précédemment dans la figure 9.7.



(a) $L = 1$ (b) $L = 2$

FIGURE 9.8 – Courbes d'apprentissage des méta-modèles lors de la recherche de solutions en fonction du paramètre d'ordre de fonctions de Walsh L. Chaque courbe représente le score d'apprentissage R^2 obtenu par le méta-modèle au cours de la recherche et donc du nombre d'évaluations.

9.4.2.3 Comparaisons des temps d'exécutions

Les calculs ont été réalisés sur un CPU *AMD Ryzen 9 3900X 12-Core Processor* composé de 2 *threads* par coeur. L'implémentation de l'évaluation d'une solution à l'aide d'un modèle de forêt aléatoire a été réalisée avec la librairie Python *Scikit-learn*

(PEDREGOSA et al. 2011). Cette librairie a permis notamment de paralléliser l'apprentissage de ce modèle et d'obtenir un coût en temps de la fonction d'évaluation d'environ 20 secondes. Ce temps réduit a permis de réaliser les calculs en un temps raisonnable, d'environ 6 heures par exécution.

Le tableau 9.2 propose de comparer les temps de calculs moyens obtenus pour la recherche locale classique comparées à la recherche assistée par méta-modèle. Le paramètre L est également comparé pour évaluer le coût supplémentaire de l'utilisation d'un méta-modèle (apprentissage du méta-modèle et utilisation pour recherche locale) par rapport à une recherche classique.

	Classique	$L = 1$	$L = 2$
Heures	$6,05$	$6,228$	$13,667$

TABLEAU 9.2 – Comparison des temps de calcul moyens de la recherche locale classique et assistée par méta-modèle

La recherche assistée par méta-modèle avec l'ordre des bases de fonctions de Walsh $L = 1$ implique un coût supplémentaire de $3,79\%$ en temps de calcul, qui reste relativement faible comparé à l'apport des performances. Malheureusement, lorsque l'on utilise l'ordre $L = 2$, l'apprentissage du modèle de substitution implique un coût supplémentaire de calcul global de $127,77\%$. Cela est principalement dû au cas d'étude pour lequel la fonction d'évaluation prend environ 20 secondes, ce qui est très peu pour une fonction dite coûteuse qui normalement prend au moins plusieurs minutes (mais rendant possible la recherche de solutions pour notre cas d'étude). Cela ne permet pas au modèle de substitution d'ordre 2 de se montrer performant étant donné que son apprentissage prend plus de temps comparé au temps gagné pour les performances d'évaluation. Dans un tout autre contexte, le compromis temps d'apprentissage par rapport au temps gagné en recherche locale assistée par le méta-modèle serait meilleur. Toutefois, les résultats obtenus avec le méta-modèle $L = 2$ restent meilleurs.

9.4.2.4 Attributs de bruit identifiés par la méthode

Les deux figures 9.9 et 9.10 illustrent le nombre de fois qu'un des attributs a été sélectionné dans la solution x' pour toutes les exécutions réalisées. La première figure illustre les occurrences indépendamment de la valeur de l'ordre L du méta-modèle, tandis que la seconde prend en considération ce paramètre. En observant la sous-figure 9.10, on peut constater que l'ordre 2 est plus strict, car il évite de retenir certains attributs contrairement à l'ordre 1. L'ordre 2 semble permettre d'obtenir de meilleures solutions par rapport à l'ordre 1 si l'on compare le meilleur score AUC ROC obtenu.

La visualisation des attributs sélectionnés par les solutions x' obtenues avec l'ordre

$L = 2$ (voir figure 9.10) nous permet de cibler certains attributs comme étant plus intéressant en entrée au modèle. Si l'on considère ne conserver que les attributs avec une occurrence supérieure à cinq, alors les attributs les plus sélectionnés sont :

— l'écart-type de l'image obtenue après application d'un filtre de moyennage 3×3 ;
— la moyenne de l'image obtenue après application d'un filtre de gaussien avec $\sigma = 1,5$ et 3×3 ;
— la moyenne de l'image obtenue après application d'un filtre de gaussien avec $\sigma = 0,5$ et 5×5 ;
— l'écart-type de l'image obtenue après application d'un filtre de gaussien avec $\sigma = 0,5$ et 5×5 ;
— la moyenne de l'image obtenue après application d'un filtre de gaussien avec $\sigma = 1$ et 5×5 ;
— l'écart-type de l'image obtenue après application d'un filtrage par ondelette ;
— les informations spatiales de l'image obtenue après application d'un filtre de Sobel 5×5 ;
— l'attribut de Kolmogorov ;
— l'attribut de la SVD-Entropy ;
— la moyenne de l'image en niveaux de gris L ;
— l'écart-type de l'image en niveaux de gris L ;

Les autres attributs ont été très peu ou aucunement sélectionnés. Les informations obtenues semblent nous fournir un élément de réponse pour la sélection d'attributs caractéristiques du bruit résiduel de MC.

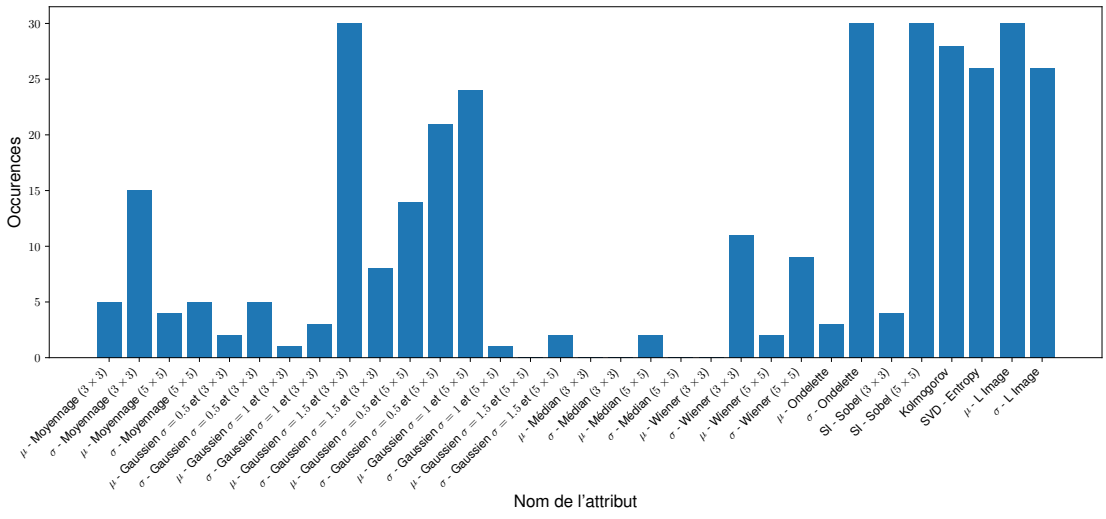


FIGURE 9.9 – Aperçu des attributs sélectionnés lors de la recherche de solutions assistées par méta-modèle.

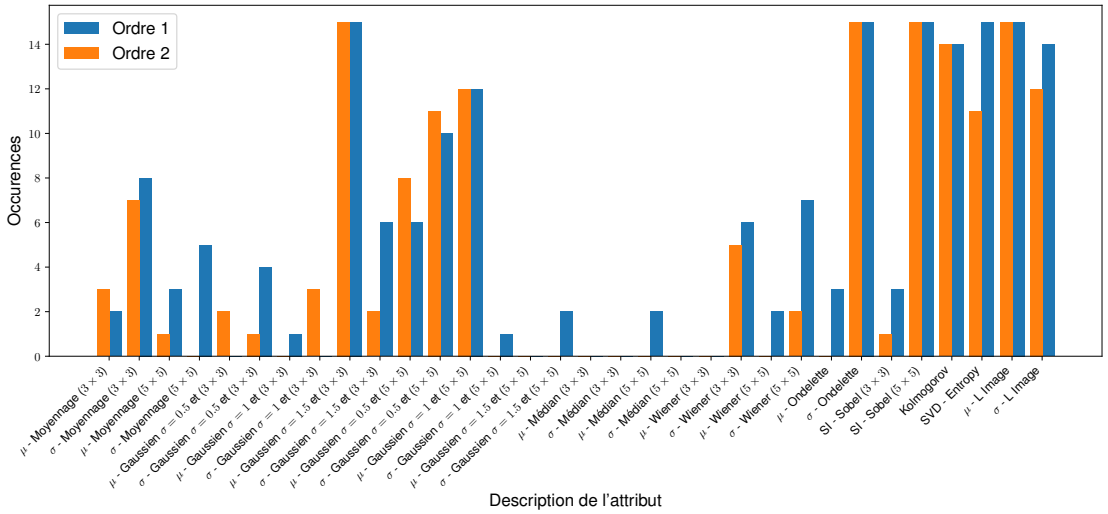


FIGURE 9.10 – Aperçu des attributs sélectionnés lors de la recherche de solutions assistées par méta-modèle en fonction de L

9.4.3 Comparaisons à des méthodes existantes

Après avoir mis en avant les résultats obtenus avec l'aide d'un méta-modèle en comparaison à une méthode de recherche de solutions classique, les méthodes de sélections d'attributs évoquées précédemment ont été comparées. L'idée est ici de vérifier la pertinence des attributs sélectionnés par la meilleure solution x' obtenue. Cette solution a sélectionné 10 attributs et possède un score AUC ROC de $81,01\%$. Les 10 attributs sélectionnés en question sont les mêmes que ceux ayant le plus grand nombre d'occurrences (comme présentés dans la figure 9.10) à l'exception de la moyenne de l'image obtenue après application d'un filtre de gaussien avec $\sigma = 0,5$ et 5×5.

L'approche proposée a été comparée à la méthode RFE pour le choix des attributs. Pour cela, un modèle de type forêt aléatoire a été utilisé pour définir quels seraient les k meilleurs attributs selon la méthode RFE. Les résultats obtenus indiquent que cette méthode sélectionne 9 attributs, soit un de moins que l'approche proposée par optimisation. Le critère de sélection, c'est-à-dire, le nombre d'attributs sélectionnés est inhérent à la méthode. La figure 9.11 présente la différence des attributs sélectionnés par les deux méthodes. La grande différence est que notre approche conserve l'attribut de Kolmogorov, mais ne conserve pas l'information spatiale obtenue avec le filtre de Sobel (3×3) contrairement à la méthode RFE.

À partir des caractéristiques identifiées, aussi bien par la méthode que nous proposons que par la RFE, nous entrainons pour chacune un modèle SVM, où nous limitons les données d'entrée de la base d'images. En effet, comme déjà mentionné, le problème de dimension des données, aussi bien le nombre d'attributs que la quantité des données,

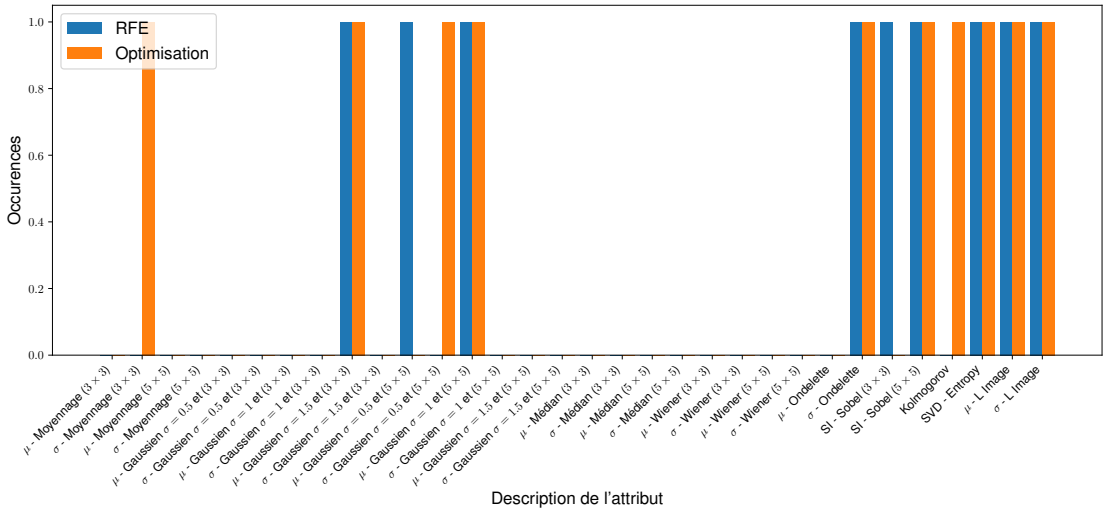


FIGURE 9.11 – Aperçu des attributs sélectionnés par la méthode proposée comparée à ceux sélectionnés l'approche RFE.

impactent considérablement l'apprentissage du SVM. Pour cela, nous limitons la taille de cette base de données en ne conservant que les 32 attributs extraits des images avec un pas de 100 échantillons plutôt que de 20. Cela permet de réduire l'ensemble de données par cinq, soient 48 000 données au lieu de 240 000 pour la même procédure de séparation de l'apprentissage du chapitre 7. Pour le SVM, nous proposons de tester plusieurs jeux de paramètres pour trouver le meilleur hyperplan séparateur : $C \in \{0,001; 0,01; 0,1; 1; 5; 10; 100; 1\ 000\}$ et $\gamma \in \{0,001; 0,01; 0,1; 1; 5; 10; 100; 1\ 000\}$, un noyau RBF (KUO et al. 2013) pour changer la projection des données comme proposé par (J. CONSTANTIN, BIGAND et al. 2015 ; TAKOUACHET et al. 2017) et une validation croisée pour réduire le surapprentissage (de taille 10).

Le tableau 9.3 présente les différents scores AUC ROC obtenus avec les attributs sélectionnés par les deux approches pour les modèles de type forêt aléatoire et SVM. Il est important de préciser que malgré un processus de validation croisée, les scores d'apprentissage des deux modèles obtenus était de 99%. Ce qui semble indiquer que, de manière générale, le modèle de forêt aléatoire trop simplifié pour la tâche, surapprend. Les temps de calculs du SVM sont de 7 heures et 7 minutes pour l'approche proposée et de 6 heures et 49 minutes pour l'approche RFE. Les attributs sélectionnés par l'approche d'optimisation assistée par méta-modèle semblent donner de meilleurs résultats pour la forêt aléatoire. Toutefois, aucune des deux méthodes de sélection d'attributs ne fournit des attributs en entrée pour le modèle SVM qui lui permet d'obtenir un bon score AUC ROC sur la base de test. Il semblerait que les attributs sélectionnés par les deux approches sont liés au modèle initialement ciblé (celui de la forêt aléatoire) et ne peuvent pas être exploités pour un autre type de modèle.

À noter que la méthode RFE nécessite un temps de calcul ≈ 2 heures et 20 minutes dans les mêmes conditions que les simulations proposées pour le processus d'optimisation assisté par méta-modèle. De plus, il est important d'ajouter que le choix de la forêt aléatoire en guise de cas d'études est justifiable étant donné le temps de calcul du SVM pour un jeu de paramètres conséquents et un nombre d'attributs réduit.

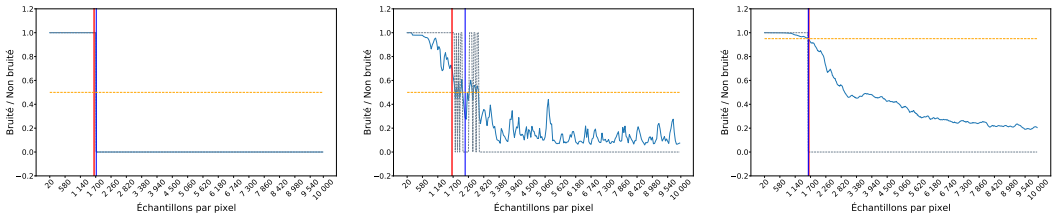
Modèle	*Attributs optimisation*	*Attributs RFE*
Forêt	81,01%	77,26%
SVM	57,17%	58,41%

TABLEAU 9.3 – Scores AUC ROC obtenus sur la base de test par les deux approches de sélection d'attributs en fonction d'un type de modèle.

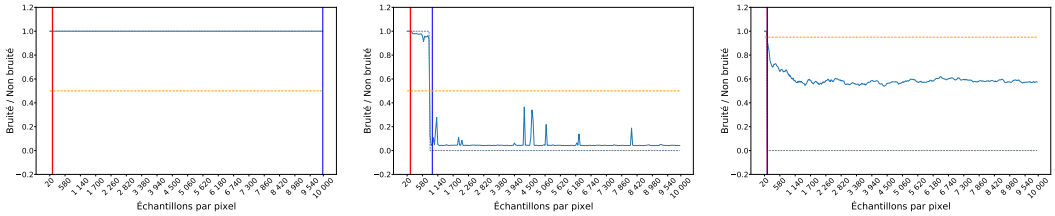
Des simulations de prédictions de critères d'arrêt sur les 40 images de la base d'images ont été réalisées à partir du modèle de forêt aléatoire obtenu avec les attributs sélectionnés par optimisation assistée par méta-modèle. La figure 9.12 illustre ces résultats de prédictions du modèle de forêt aléatoire en comparaisons au modèle *SVD-Entropy* et GGN obtenus respectivement dans les chapitres 7 et 8. Les blocs présentés en guise de comparaisons sont des blocs non appris par les modèles *SVD-Entropy* et forêt aléatoire. Le GGN prédit parfaitement les seuils subjectifs, mais il faut préciser que ce sont deux images qui étaient présentes dans les 35 images de sa base d'apprentissage. Le modèle *SVD-Entropy* (RNN) a tendance à s'arrêter un peu plus tard que le seuil subjectif humain, mais prédit correctement. Quant à lui, le modèle de forêt aléatoire répond parfaitement sur le bloc de l'image *Arc sphere* mais ne trouve aucun critère d'arrêt pour le bloc d'image *Ecosys*. Ce dernier cas est malheureusement assez courant sur certains blocs non appris, ce qui reflète la limite d'un modèle forêt aléatoire pour la tâche demandée, plus simple dans sa structure. À noter que les prédictions d'un modèle de forêt aléatoire sont purement binaire, d'où cette représentation dans la figure.

9.5 Discussion

Les résultats obtenus permettent de valider l'approche entreprise concernant l'utilisation du méta-modèle pour guider la recherche de solutions de sélection d'attributs. Toutefois, plusieurs interrogations nécessitent encore des réponses. Les résultats présentés dans ce chapitre traitent d'une version de modèle *simplifiée* de type forêt aléatoire, il faut envisager cette étude avec un modèle plus complexe tel que le SVM. Dans ce cas, il faut vérifier que le méta-modèle arrive à répondre aussi bien pour un modèle d'apprentissage plus complexe, utilisé dans la fonction d'évaluation coûteuse f. Un deuxième point très important, concerne l'analyse des résultats présentés sur la sélection des attributs et l'importance des attributs sélectionnés. En effet, les attributs sélectionnés



(a) *Arcsphere* bloc 2 - Forêt aléatoire (b) *Arcsphere* bloc 2 - RNN (c) *Arcsphere* bloc 2 - GGN



(d) *Ecosys* bloc 2 - Forêt aléatoire (e) *Ecosys* bloc 2 - RNN (f) *Ecosys* bloc 2 - GGN

FIGURE 9.12 – Comparaisons des prédictions obtenues à partir du modèle de forêt aléatoire retenue sur des blocs non appris de l'image *Arcsphere* et *Ecosys* par le modèle *SVD-Entropy* (RNN) et forêt aléatoire. Seul le modèle GGN avait appris sur ces images.

majoritairement comme étant importants par le méta-modèle, sont peut-être liés au modèle de forêt aléatoire. Un début de réponse à cette dernière question a été proposé par les résultats du tableau 9.3 où les attributs sélectionnés pour le modèle de forêt aléatoire ne répondent pas correctement pour le modèle SVM.

Pour pouvoir justifier qu'une telle approche fonctionne pour la sélection d'attributs, il faudra évidemment bien vérifier la généralisation de la méthode pour d'autres modèles. Ainsi, si l'importance des attributs sélectionnés par les méta-modèles tendent à être similaire, alors il peut y avoir généralisation. Dans le cas contraire, il faudra interpréter davantage les résultats obtenus.

Résumé

Ce neuvième chapitre propose la reformulation du problème de sélection d'attributs connu en apprentissage automatique, en un problème d'optimisation combinatoire binaire. Ce problème, une fois modélisé en un problème d'optimisation, possède généralement un espace de recherche de solutions très grand. De plus, la fonction d'évaluation f modélisée pour ce genre de problème, correspond aux performances sur la base de test d'un modèle d'apprentissage automatique nouvellement appris. L'utilisation d'une telle fonction est plus communément connu comme étant un problème d'optimisation coûteuse, car le parcours de l'espace de recherche de solutions ne peut pas être réalisé de manière classique étant donné un temps d'évaluation long d'une solution. Ainsi, afin de répondre de manière plus rapide et de trouver plus efficacement de bonnes solutions, il a été proposé d'exploiter l'utilisation de méta-modèles afin de guider la recherche. Une approche récemment proposée de conception de méta-modèles à base de fonctions de Walsh semble avoir apporté de bons résultats pour des problèmes d'optimisation combinatoire binaire coûteux, où la fonction d'évaluation d'une solution correspondait à la sortie d'un processus de simulation. À partir de cette approche, nous avons appliqué et testé ce même type de méta-modèle, afin qu'il essaye d'apprendre la complexité de la fonction d'évaluation f liée au problème de sélection d'attribut qui, dans notre cas, correspond à l'obtention des performances d'un nouveau modèle d'apprentissage relativement à la solution courante x. Dans le cas d'étude traité, les attributs en question correspondent à différents attributs caractéristiques de bruit identifiés au sein de la littérature et qui ont été appliqués à la problématique de la thèse, qu'est la détection de bruit dans les images de synthèse photo-réaliste. Ainsi, le modèle d'apprentissage testé est un modèle de forêt aléatoire, moins coûteux en temps qu'un modèle SVM et utilisé donc pour caractériser la présence ou non du bruit dans une image. Les résultats proposés dans ce chapitre semblent indiquer que cette approche s'adapte bien pour ce genre le problème ciblé de sélection d'attributs et que le méta-modèle arrive à interpréter correctement la complexité du modèle d'apprentissage automatique. Toutefois, ces travaux étant exploratoires, il reste à valider que les résultats peuvent être identiques pour tout type de modèle d'apprentissage utilisé dans la fonction f et enfin d'envisager une interprétation du lien entre les attributs sélectionnés et la notion de bruit de MC.

Chapitre 10

Validations des modèles

Introduction

Ce dernier chapitre propose une phase de vérification et validation des modèles de perception du bruit de MC proposés dans les chapitres 7 et 8. En effet, comme il a pu être présenté respectivement dans chacun de ces deux chapitres, les modèles obtenus semblent apporter globalement de bons résultats par comparaison à l'image de référence. Toutefois, il est important de vérifier si les seuils obtenus par ces modèles sont en adéquation avec le système visuel humain et ne laissent pas apparaître de bruit perceptible pour l'homme, en particulier quand la valeur prédite du seuil de bruit par le modèle est inférieure au seuil humain moyen attendu.

Les approches d'apprentissage automatique proposées dans le cadre de cette thèse ont permis d'obtenir des modèles de perception de bruit qui semblent prédire correctement les seuils humains subjectifs souhaités. La section 10.1 analyse plus finement les prédictions des modèles de perception obtenus. Suite à cette analyse, la section 10.2 présente une expérience visant à valider les performances des modèles obtenus, en vérifiant que les images fournies par chacun d'entre eux ne laissent paraître aucun bruit perceptible pour l'humain. Dans la section 10.3, on s'intéresse également à l'application de ces modèles sur des images où le seuil humain n'avait pas été déterminé sur certains blocs avant les 10 000 échantillons. Une analyse des résultats obtenus de ces modèles est également présentée afin de mettre en avant l'intérêt que peuvent susciter de tels modèles.

10.1 Analyse des détections erronées

Les modèles de perceptions proposés dans les chapitres 7 et 8 apportent de bonnes prédictions dans l'ensemble. Toutefois, certaines d'entre-elles ont été reconnues comme incorrectes. Nous souhaitons dans cette section, analyser plus finement ces résultats afin

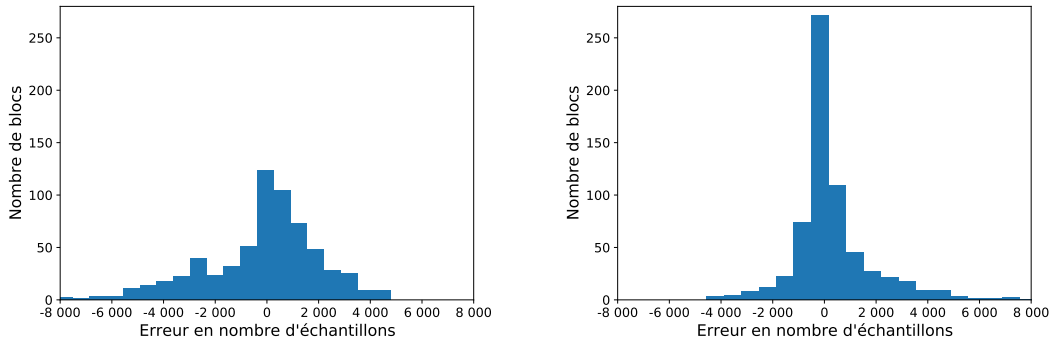
de vérifier la proportion pour chaque modèle de détections d'arrêt erronées : précoces ou tardives.

	Détections	**Avant seuil**						**Après seuil**					
	Modèle	GGN			*SVD-Entropy*			GGN			*SVD-Entropy*		
t	Marge	Apprentissage	Test	Global	Apprentissage	Test	Global	Apprentissage	Test	Global	Apprentissage	Test	Global
0.20	0 %	0.86	11.67	1.88	11.90	10.00	11.72	88.62	71.67	87.03	87.93	90.00	88.12
	2 %	0.52	8.33	1.25	10.34	8.33	10.16	84.66	71.67	83.44	75.69	73.33	75.47
	5 %	0.34	5.00	0.78	8.79	8.33	8.75	79.83	70.00	78.91	70.69	66.67	70.31
0.30	0 %	2.24	15.00	3.44	22.24	18.33	21.88	87.24	68.33	85.47	77.41	80.00	77.66
	2 %	1.03	10.00	1.88	20.34	15.00	19.84	82.41	66.67	80.94	67.41	63.33	67.03
	5 %	0.34	6.67	0.94	17.76	11.67	17.19	76.90	65.00	75.78	61.38	55.00	60.78
0.40	0 %	2.93	16.67	4.22	31.90	28.33	31.56	86.55	66.67	84.69	68.10	71.67	68.44
	2 %	1.38	10.00	2.19	28.10	25.00	27.81	80.69	66.67	79.38	59.48	58.33	59.38
	5 %	0.86	6.67	1.41	25.00	23.33	24.84	73.79	65.00	72.97	51.38	38.33	50.16
0.50	0 %	6.72	16.67	7.66	41.38	45.00	41.72	82.93	66.67	81.41	58.28	55.00	57.97
	2 %	2.76	15.00	3.91	37.07	40.00	37.34	75.86	66.67	75.00	49.83	38.33	48.75
	5 %	1.55	10.00	2.34	32.76	35.00	32.97	66.55	61.67	66.09	40.69	31.67	39.84
0.60	0 %	8.28	16.67	9.06	51.72	63.33	52.81	81.38	66.67	80.00	48.10	33.33	46.72
	2 %	3.97	15.00	5.00	46.72	56.67	47.66	72.76	66.67	72.19	40.00	28.33	38.91
	5 %	2.07	10.00	2.81	41.03	46.67	41.56	61.72	61.67	61.72	31.03	20.00	30.00
0.70	0 %	11.90	20.00	12.66	60.86	73.33	62.03	78.10	63.33	76.72	38.97	26.67	37.81
	2 %	6.03	16.67	7.03	56.38	68.33	57.50	67.76	61.67	67.19	29.14	20.00	28.28
	5 %	2.76	11.67	3.59	51.38	66.67	52.81	55.34	55.00	55.31	22.41	13.33	21.56
0.80	0 %	17.41	25.00	18.12	69.48	78.33	70.31	72.76	60.00	71.56	30.34	21.67	29.53
	2 %	10.00	21.67	11.09	65.34	70.00	65.78	58.97	56.67	58.75	21.03	13.33	20.31
	5 %	5.17	15.00	6.09	59.14	68.33	60.00	46.03	51.67	46.56	11.90	6.67	11.41
0.90	0 %	28.10	35.00	28.75	78.45	88.33	79.38	63.10	50.00	61.88	21.38	11.67	20.47
	2 %	17.24	31.67	18.59	72.59	78.33	73.12	49.31	48.33	49.22	13.79	3.33	12.81
	5 %	9.31	23.33	10.62	66.90	68.33	67.03	35.86	41.67	36.41	6.72	0.00	6.09
0.95	0 %	41.55	40.00	41.41	83.28	91.67	84.06	52.07	46.67	51.56	16.21	8.33	15.47
	2 %	28.97	36.67	29.69	79.48	81.67	79.69	36.55	38.33	36.72	8.28	1.67	7.66
	5 %	18.45	28.33	19.38	71.03	76.67	71.56	25.52	38.33	26.72	3.45	0.00	3.12
0.98	0 %	59.66	48.33	58.59	96.21	98.33	96.41	35.52	38.33	35.78	3.79	1.67	3.59
	2 %	46.90	41.67	46.41	93.79	95.00	93.91	23.97	36.67	25.16	0.17	0.00	0.16
	5 %	34.14	33.33	34.06	90.34	83.33	89.69	17.41	36.67	19.22	0.00	0.00	0.00

TABLEAU 10.1 – Comparaisons des pourcentages de critère d'arrêt inférieurs ou supérieurs aux seuils humains des deux modèles en fonction du seuil de classification t. Les performances sont également séparées relativement à la base d'apprentissage et de test. La notion de marge d'erreur est aussi présentée, avec les valeurs de 2% et 5%, soient respectivement 200 et 500 échantillons par pixel avant et après la valeur du seuil humain.

Le tableau 10.1 présente les pourcentages de valeurs de seuils trop précoces et trop tardifs par rapport aux seuils humains obtenus sur les différents blocs des images. Les résultats obtenus avec une marge d'erreur, sont aussi présentés, avec des valeurs de 2% et 5%, soit respectivement 200 et 500 échantillons par pixel de part et d'autre du seuil. Cette marge permet de vérifier si le modèle propose un arrêt relativement proche des seuils subjectifs humains et autorise donc cette « faible » erreur. On rappelle que les meilleures classifications qui ont été obtenues pour nos deux modèles, sur la base du score AUC ROC (voir section 8.4), avaient un seuil de classification de 0.5 pour le modèle *SVD-Entropy* et 0.95 pour le GGN. Le modèle *SVD-Entropy* émet son critère

d'arrêt après au moins 3 prédictions consécutives non bruitées. Au contraire 1 seule prédiction est considérée pour le GGN, que nous considérons comme plus stable. Pour rappel, ce seuil de classification permet de définir à partir de quelle valeur de prédiction d'un modèle, on considère que l'image est encore bruitée ou non (voir la définition de la métrique AUC ROC dans la section 3.1.3.1). Les résultats figurant dans le tableau 10.1 montrent que pour ces deux seuils de sélection, le modèle *SVD-Entropy* possède un pourcentage global assez élevé d'arrêts anticipés, de l'ordre de $41,72\%$ qui descend à $32,97\%$, si l'on autorise une marge de 5%. Le GGN quant à lui fournit un pourcentage d'environ 41.41% réduit à 19.38% si l'on autorise une marge de 5%. Le GGN semble donc un peu plus robuste par rapport à ce critère d'arrêt anticipé, car il propose un arrêt précoce plus proche du seuil humain. Toutefois, on remarque la présence légère du surapprentissage pour ce dernier, notamment lorsque l'on autorise une marge à 5% : $18,45\%$ d'arrêts précoces pour la base d'apprentissage, contre $28,33\%$ pour la base de test. L'analyse inverse, concerne les détections trop tardives du modèle, soit celles fournissant une valeur de seuil supérieure au seuil humain subjectif. On peut noter que le modèle *SVD-Entropy* a tendance à fournir davantage de seuils plus tardifs que le modèle GGN. Nous avons ciblé dans ce tableau des pourcentages proposant un compromis entre détections trop précoces et trop tardives, illustrés par un fond vert pour chacun des modèles. Ils sont liés aux seuils de classification sélectionnés initialement pour chacun des deux modèles.



(a) *SVD-Entropy* (b) GGN

FIGURE 10.1 – Répartition des écarts entre les seuils produits et les seuils subjectifs, pour chacun des 16 blocs des 40 images.

La figure 10.1 propose une autre représentation de ces erreurs et illustre la répartition des écarts des seuils produits aux seuils subjectifs. Pour chacun des 16 blocs des 40 images, l'erreur commise est calculée en nombre d'échantillons relativement au seuil humain subjectif. Ces répartitions possèdent la même échelle pour interpréter au mieux la différence des écarts entre ces deux modèles. Le modèle *SVD-Entropy* possède une

répartition des erreurs plus étalée, mais a tendance à prédire plus tardivement que précocement. Toutefois, il s'arrête parfois bien trop avant le seuil humain, conduisant à quelques erreurs d'au moins 6 000 échantillons. Le GGN propose des seuils plus proches des seuils humains subjectifs. De plus, il commet moins d'erreurs trop précoces de détection que son homologue. Cependant, il lui arrive pour quelques blocs de prédire plus tardivement.

10.2 Expérience de validation

Les résultats présentés dans la section précédente montrent les écarts des modèles relativement aux seuils subjectifs humains. Ces écarts sont plus ou moins conséquents pour certains blocs, mais dans l'ensemble les deux modèles produisent des prédictions proches des seuils humains. De plus, les résultats présentés dans les chapitres 7 et 8 étaient comparés aux images de référence à 10 000 échantillons à partir de la métrique de comparaison SSIM. Bien que cette métrique soit assez représentative du HVS, elle ne permet pas de valider judicieusement la perception du bruit que peuvent avoir les utilisateurs en visualisant une image de synthèse. Pour réaliser une validation visuelle des prédictions des seuils et juger de l'impact des écarts commis par chacun des modèles, nous avons opté pour une nouvelle expérience. Cette expérience demande aux participants de déterminer si du bruit est encore présent ou non dans les images reconstruites à partir des prédictions des modèles.

Nous avons ainsi développé une nouvelle application, qui permet de comparer deux images, l'une étant l'image reconstruite à partir des seuils prédits d'un modèle, l'autre l'image de référence humaine, c'est-à-dire celle reconstruite à partir des seuils subjectifs humains moyens provenant de la base d'images. La question posée aux participants est la suivante : « Les images vous paraissent-elles identiques ? ». Proposer cette question à un participant a pour objectif de savoir si du bruit est encore présent ou non dans l'image proposée par le modèle, par comparaison à l'image issue des seuils humain.

10.2.1 Paramètres et configuration de l'expérience

Pour certains blocs des 40 images disponibles, du bruit pouvait être encore présent sur l'image considérée comme référence et calculée avec 10 000 échantillons par pixel. Ce n'était pas dérangeant pour les modèles d'avoir connaissance de ces données, car cela revient à avoir des données où le bruit était toujours présent en entrée du modèle. Toutefois, pour la validation des résultats, il est nécessaire que la référence humaine (reconstruite à partir des seuils humains) soit composée d'un niveau de bruit imperceptible humainement. Ainsi, seules les images ayant des seuils subjectifs inférieurs à 10 000 échantillons pour les 16 blocs ont été conservés pour cette expérience, soit un total de 28 images sur les 40 disponibles.

Pour chacune de ces 28 images, trois images sont comparées à l'image humaine reconstruite :

— l'image reconstruite à partir du modèle GGN du chapitre 8 ;
— l'image reconstruite à partir du modèle *SVD-Entropy* présenté au chapitre 7 sur la même base d'apprentissage que le modèle GGN ;
— une image reconstruite à partir des valeurs de seuils humains huit fois moindre pour chaque bloc, pour mettre en avant une image composée de bruit.

L'image reconstruite avec des valeurs huit fois moindre que le seuil subjectif humain pour chaque partie de l'image a pour objectif de laisser apparaître du bruit et d'éviter les faux positifs. En effet, relativement à la question posée, si le participant ne remarque jamais de bruit sur les images reconstruites à partir des deux modèles, alors il risque de répondre positivement à la présence de bruit sur l'une des deux, ce qui peut conduire à la génération de résultats erronés (faux positifs).

FIGURE 10.2 – Représentation de l'application de validation des images reconstruites avec les prédictions des modèles de perception.

La figure 10.2 illustre l'application proposée au participant, où deux images sont présentées. L'une d'entre elle est l'image de référence humaine, la seconde une image reconstruite à partir d'un modèle ou avec un niveau de bruit défini. Il est également important de préciser que l'image humaine est positionnée aléatoirement parmi les deux images proposées, afin de contrebalancer un effet systématique de réponse de la position à droite ou à gauche de l'image de référence. Le participant peut alors répondre s'il perçoit une différence entre les deux images ou non. La référence humaine étant la « vérité terrain », si aucune différence n'est perceptible cela veut dire que les performances du modèle sont correctes et répondent à l'attendu. Un exemple est présenté avant le début de l'expérience, où l'une des deux images est composée d'un bruit fort. Cet exemple

permet au participant de mieux comprendre la tâche et d'avoir une notion de différence perceptible de bruit entre deux images. Lors de cette procédure, le temps de réponse est laissé libre. De plus, la présentation simultanée de l'image de référence et du stimulus à évaluer correspond à la procédure la plus exigeante, puisque l'observateur peut à tout moment comparer les deux images favorisant la détection du bruit.

10.2.2 Analyse des résultats

La population des participants pour cette expérience est assez diverse : étudiants, collègues et familles, soit un total de 17 participants (5 femmes et 12 hommes) d'âge moyen de 31 ans. Deux sujets deutéranopes (daltonisme) ont été inclus dans l'étude, mais leurs données ne semblent pas différer des autres a posteriori. Ils ont d'ailleurs rapporté ne pas avoir été dérangés par leur daltonisme pour effectuer la tâche demandée. Les résultats obtenus sont présentés dans le tableau 10.2 qui met en avant le pourcentage moyen d'images perçues comme non bruitées par rapport à l'image de référence humaine pour les 3 classes d'images reconstruites visées.

	Bruitée	*SVD-Entropy*	GGN
Images	80/476	403/476	419/476
Pourcentage	17%	85%	88%

TABLEAU 10.2 – Nombre d'images et pourcentage d'images considérées comme non bruitées par les utilisateurs lors de l'expérience pour chacune des 3 classes d'images présentées.

(a) Image modèle GGN (b) Bloc modèle GGN (c) Bloc référence humaine

FIGURE 10.3 – Aperçu d'un bloc de l'image *Sanmiguel vue 2* où plusieurs *fireflies* sont apparus lors du rendu après le seuil humain subjectif (après l'ajout d'autres échantillons dont un ayant une grande valeur de contribution).

Modèle	GGN		*SVD-Entropy*		Bruitée	
Scène	Réponses	Pourcentage	Réponses	Pourcentage	Réponses	Pourcentage
Arcsphere	16/17	94.12%	16/17	94.12%	7/17	41.18%
Bunny Fur	16/17	94.12%	16/17	94.12%	3/17	17.65%
Car2	15/17	88.24%	11/17	64.71%	5/17	29.41%
Caustic	14/17	82.35%	13/17	76.47%	2/17	11.76%
Coffee Splash	15/17	88.24%	13/17	76.47%	0/17	0.00%
Crown	14/17	82.35%	15/17	88.24%	4/17	23.53%
Dragon	16/17	94.12%	15/17	88.24%	1/17	5.88%
Ecosys	14/17	82.35%	16/17	94.12%	11/17	64.71%
Eponge Fractal vue 1	15/17	88.24%	16/17	94.12%	8/17	47.06%
Eponge Fractal vue 2	14/17	82.35%	13/17	76.47%	2/17	11.76%
Ganesha	17/17	100.00%	15/17	88.24%	7/17	41.18%
Glass Of Water	17/17	100.00%	15/17	88.24%	2/17	11.76%
Indirect	13/17	76.47%	16/17	94.12%	2/17	11.76%
Landscape*	12/17	70.59%	16/17	94.12%	7/17	41.18%
Living Room vue 2	14/17	82.35%	14/17	82.35%	0/17	0.00%
Living Room vue 4	16/17	94.12%	16/17	94.12%	0/17	0.00%
Low table	13/17	76.47%	15/17	88.24%	1/17	5.88%
Pavilion Day vue 1	15/17	88.24%	12/17	70.59%	2/17	11.76%
Pavilion Day vue 2	14/17	82.35%	12/17	70.59%	1/17	5.88%
Pavilion Day vue 3	15/17	88.24%	15/17	88.24%	3/17	17.65%
Pavilion Night vue 2	16/17	94.12%	12/17	70.59%	0/17	0.00%
Sanmiguel vue 1	14/17	82.35%	15/17	88.24%	3/17	17.65%
Sanmiguel vue 2*	12/17	70.59%	15/17	88.24%	1/17	5.88%
Staircase 1*	12/17	70.59%	16/17	94.12%	1/17	5.88%
Staircase 2	15/17	88.24%	15/17	88.24%	2/17	11.76%
Tt	16/17	94.12%	15/17	88.24%	0/17	0.00%
Vw Van	16/17	94.12%	10/17	58.82%	1/17	5.88%
Pourcentage test	36/51	70.05%	47/51	92.15%	-	-

TABLEAU 10.3 – Résultats de l'expérience pour chacune des images en fonction du nombre de participants. Le nombre de réponses d'images considérées comme identiques à l'image de référence humain et le pourcentage calculé sont indiqués pour chaque image. Le caractère $*$ associé au nom d'une image précise qu'elle fait partie de la base test.

Pour compléter les résultats obtenus et cibler les images problématiques, le tableau 10.3 offre un visuel des résultats obtenus par image (scène) pour chacune des trois images comparées à l'image de référence humaine. Les résultats sur les images issues de la base de test sont également détaillés pour chacun des modèles pour lesquels on peut noter que le modèle GGN est moins performant que le modèle *SVD-Entropy*, $70,05\%$ et $92,15\%$ respectivement. Cela est principalement dû au fait que le modèle GGN (voir le chapitre 8) a tendance à surapprendre. En analysant les 3 images de la base de test qui ont été incluses dans les 28 images sélectionnées, des blocs des images reconstruites *Staircase 1* et *Landscape* par le modèle GGN comportent effectivement encore un léger bruit. Toutefois, le cas de l'image *Sanmiguel vue 2* est particulier : les seuils prédits par le modèle sont tous supérieurs ou au moins égaux aux seuils humains. En réalité, la différence entre les deux images provient de la présence de quelques *fireflies* dans un bloc d'une image qui sont apparus après le seuil subjectif humain. C'est-à-dire, après l'ajout d'autres échantillons dont un ayant une grande valeur de contribution. Le modèle GGN ayant proposé un arrêt plus lointain pour ce bloc a malheureusement hérité de ces

fireflies lors du processus de rendu (voir figure 10.3). Soulignons que le résultat confirme la haute sensibilité que l'être humain peut possèder pour ce bruit. La proposition de l'estimateur *G*-MoN, offrant une gestion de ces *fireflies*, n'était pas encore développée lors de la construction des images utilisées dans la base de test, ce qui explique leur présence dans certaines images.

Les performances indiquées par les tableaux précédents montrent que les deux modèles répondent correctement à la tâche demandée et permettent d'obtenir des images où le bruit résiduel n'est pratiquement plus perceptible. Toutefois, quelques images n'ont été perçues comme étant identiques à la référence humaine qu'à hauteur de 58%. Cela est le cas pour les images *Landscape* et *Sanmiguel vue 2* pour le GGN et pour les images *Pavilion day vue 1* et *Vw Wan* pour le modèle *SVD-Entropy*.

Nous avons dans la section précédente (voir figure 10.1) mis en évidence que certaines prédictions des modèles étaient fortement éloignées des seuils subjectifs attendus. Nous proposons ici, en complément, faire le lien de ces erreurs avec les résultats obtenus par cette nouvelle expérience. On souhaite analyser si les seuils prédits fortement éloignés (écart > 4000 échantillons par pixel par rapport au seuil humain) ont eu un impact ou non sur la perception du bruit dans l'image. Pour le modèle *SVD-Entropy*, les blocs fortement erronés proviennent principalement des images avec des seuils humains fixés à 10 000 échantillons et absentes de cette expérience. Ce qui nous amène à penser que le modèle semble posséder des difficultés à interpréter que le bruit est toujours présent dans un bloc d'une image, même à 10 000 échantillons. Toutefois, pour deux des images étudiées dans cette expérience, *Glass Of Water* et *Vw Van*, il s'avère que chacune possède un bloc avec une prédiction fortement erronée du seuil par le modèle *SVD-Entropy*. Pour l'image *Glass Of Water*, cela ne semble pas avoir eu un impact visuel au vu des résultats de l'expérience qui sont de 88.24%, mais pour l'image *Vw Van*, le pourcentage indiqué de 58.82% est très faible, ce qui nous confirme l'impact qu'a pu avoir cette mauvaise prédiction sur la perception des différences. Pour le modèle GGN, seules 4 images ont été impactées par des détections éloignées dont deux images liées à cette expérience, les deux autres possédant au moins un seuil humain fixé à 10 000 échantillons, exclues de l'expérience. Les images présentes dans l'expérience sont *Landscape* et *Sanmiguel vue 2*, où les prédictions fournies par le modèle sont tardives par rapport au seuil subjectif. Ces détections n'ont donc pas d'impact direct sur la perception, à l'exception de l'apparition de *fireflies* pour la scène *Sanmiguel vue 2*. La perception de différences de la part des participants sur l'image *Landscape* est principalement causée par des détections du modèle GGN qui sont très légèrement précoces (de 20 échantillons) sur certains blocs. Étant donné qu'il s'agit d'une image extérieure, la convergence est très rapide et un seuil légèrement trop précoce peut avoir de telles conséquences. Il reste toujours envisageable, pour éviter ce genre de problème, de réduire le seuil de classification (à $0,80$ par exemple) pour être plus restrictif aux arrêts.

Une donnée complémentaire relative aux performances de chacun de ces modèles est le gain de temps de calcul lié à l'utilisation d'un modèle. Le tableau 10.4 permet de visualiser le pourcentage d'échantillons qui aurait pu être économisé si l'appel au modèle avait été réalisé lors du rendu de l'image jusque 10 000 échantillons. Bien entendu, ces résultats dépendent du nombre maximal d'échantillons par pixel fixé lors du calcul. Le tableau montre ces pourcentages à la fois pour chaque scène et une moyenne sur l'ensemble des scènes. Les pourcentages de gain d'échantillons moyens pour les modèles GGN et *SVD-Entropy* sont très proches. De plus, ils sont aussi similaires au pourcentage de gain donné par l'utilisation des seuils humains, ce qui est logique, puisque ces modèles ont appris sur ces données de seuils. Le gain de temps de calcul, dans cette situation, est de l'ordre d'un facteur 3.

Modèle	Humain	GGN	SVD-Entropy
Arcsphere	72.10%	67.10%	64.60%
Bunny Fur	82.44%	82.59%	60.68%
Car2	76.06%	76.55%	79.10%
Caustic	63.57%	51.59%	47.58%
Coffee Splash	83.07%	81.95%	55.23%
Crown	64.64%	63.53%	66.47%
Dragon	72.76%	71.82%	70.28%
Dragon	87.62%	86.27%	77.58%
Ecosys	98.53%	98.70%	91.85%
Eponge Fractal vue 1	93.25%	93.29%	81.12%
Eponge Fractal vue 2	82.09%	83.60%	80.12%
Ganesha	87.39%	88.59%	83.90%
Glass Of Water	32.75%	20.52%	53.90%
Indirect	77.67%	81.21%	74.08%
Landscape	68.01%	68.39%	63.40%
Living Room vue 2	38.86%	14.81%	56.53%
Living Room vue 4	50.69%	23.77%	56.93%
Low Table	65.14%	65.75%	63.45%
Pavilion Day vue 1	70.86%	69.50%	65.95%
Pavilion Day vue 2	63.17%	66.49%	60.88%
Pavilion Day vue 3	82.82%	84.46%	79.08%
Pavilion Night 2	53.36%	60.29%	57.86%
Sanmiguel vue 1	53.30%	60.28%	57.75%
Sanmiguel vue 2	45.27%	21.55%	49.98%
Staircase 1	93.99%	94.90%	88.25%
Staircase 2	65.53%	50.56%	62.88%
Tt	79.80%	74.80%	63.28%
Vw Van	77.79%	74.83%	65.67%
Moyenne	70.80%	67.06%	67.08%

TABLEAU 10.4 – Pourcentage d'échantillons économisés par chacun des modèles comparés aux seuils humains des blocs d'une image.

10.3 Prédiction de nouveaux seuils

Comme mentionné précédemment, quelques images comportent un certain nombre de blocs pour lesquels 10 000 échantillons par pixel ne sont pas suffisants pour éliminer la

présence de bruit. Afin de compléter l'analyse des performances de chacun des modèles, nous proposons d'estimer les seuils au-delà de 10 000 échantillons sur deux des images de la base d'images, celles de *Classroom vue 1* et de *Living-room vue 2*. Pour cela, nous avons calculé jusque 64 000 échantillons par pixel ces deux images et avons utilisé les deux modèles pour obtenir les prédictions de seuils pour chaque bloc d'une image.

11820	11060	9580	7860
11620	7060	9540	6940
8540	6420	6340	8340
5420	6140	6940	8300

(a) RNN - *Living-room* SSIM : 0.8933

20380	30760	30120	26460
12120	18000	39880	20360
16400	32920	64000	32000
36440	42620	22960	29940

(b) GGN - *Living-room* SSIM : 0.9407

9073	10000	10000	7433
8806	6086	9206	6820
6620	8406	10000	10000
9006	9740	10000	10000

(c) Seuils humains initiaux

(d) 64 000 échantillons

11220	14380	14860	14060
7900	14020	11700	12980
7220	8820	9780	9340
6980	6140	7340	8420

(e) RNN - *Classroom* SSIM : 0.8973

21640	11980	26920	34640
36800	28460	43940	29900
17420	29400	32920	20360
28920	35320	26240	51920

(f) GGN - *Classroom* SSIM : 0.9461

10000	10000	10000	10000
8220	10000	10000	10000
10000	9280	10000	9220
8346	9606	8073	9153

(g) Seuils humains initiaux

(h) 64 000 échantillons

FIGURE 10.4 – Seuils objectifs obtenus sur les images *Classroom* et *Living-room* calculées jusqu'à 64 000 échantillons. Un bloc possédant un seuil humain de 10 000 échantillons correspond au fait que l'image initiale contient encore du bruit pour ce bloc.

Pour chacun des modèles, la fenêtre glissante d'images est fixée par $S = 6$. Les seuils de critère d'arrêt du modèle *SVD-Entropy* (RNN) sont obtenus à partir d'un seuil de classification de $0,5$ et au moins 3 prédictions consécutives non bruitées. Pour le modèle GGN le seuil de classification est fixé à $0,95$ et le critère d'arrêt est défini dès que le modèle atteint ce seuil, soit 1 seule prédiction non bruitée.

La figure 10.4 propose de visualiser les seuils obtenus pour chacun des modèles sur ces deux images. Le modèle GGN est beaucoup plus « sévère » que le modèle *SVD-Entropy* (RNN) quant à la présence de bruit face à l'inconnu. Il propose un critère d'arrêt toujours au-delà des 10 000 échantillons même si le seuil subjectif pouvait être inférieur. Au contraire, le modèle *SVD-Entropy* a tendance à prédire un critère d'arrêt avant le seuil humain, excepté pour quelques blocs. Le score SSIM est indiqué pour chacune des images reconstruites à partir des seuils obtenus des modèles. Le modèle GGN est

évidemment plus proche de la référence (64 000 échantillons) étant donné ses seuils prédits qui sont plus élevés. Il fournit toutefois des critères d'arrêt sur certains blocs d'images amenant à économiser de 2 à 6 fois le nombre d'échantillons maximal fixé de manière subjective. Le modèle GGN propose ainsi une meilleure conservation de la qualité face à l'inconnu.

10.4 Discussion

L'expérience réalisée dans ce chapitre, nous indique que les modèles de perception proposés fournissent des critères d'arrêt cohérents avec la perception humaine du bruit. Le modèle *SVD-Entropy* a tendance à s'arrêter plus tôt que le modèle GGN. Toutefois, il offre une meilleure généralisation que le modèle GGN. En effet, ce second semble mal prédire certains blocs d'images de test, amenant à une présence de bruit dans l'image et nous confirmant son léger surapparentissage. Les images générées jusque 64 000 échantillons par pixel ont permis de visualiser les prédictions des deux modèles face à l'inconnu. Ces deux modèles n'ayant appris que sur des images générées avec au maximum 10 000 échantillons par pixel, il était important de connaître leur prédiction au-delà de cette limite. De plus, certains blocs d'images étaient jugés comme encore composés de bruit à 10 000 échantillons. Les prédictions apportées sur ces images par les modèles nous rapportent que même face à l'inconnu, les modèles arrivent à proposer un critère d'arrêt. À noter que le modèle GGN est plus conservateur et prédit plus tard que son homologue.

Résumé

Ce dernier chapitre propose une analyse et une phase de validation des modèles de perception de bruit. Tout d'abord, il propose une analyse des performances de discrimination du bruit des deux modèles obtenus. Cette analyse met en évidence les détections erronées par ces deux modèles et les quantifie. Dans un second temps, pour proposer une validation de ces modèles, une nouvelle expérience est réalisée dans ce chapitre. Elle apporte des résultats qui montrent la pertinence de l'approche, malgré des seuils précoces ou tardifs. Enfin, sur deux images où les seuils humains n'avaient pas été déterminés avant 10 000 échantillons sur tous les blocs, une simulation des prédictions des deux modèles jusque 64 000 échantillons est proposée. Dans l'ensemble, les résultats obtenus par les deux modèles sont satisfaisants et permettent de valider l'intérêt de telles méthodes lors du rendu d'une image.

Conclusion et perspectives

Ce mémoire traite de la perception du bruit dans les images de synthèse photo-réaliste. La simulation d'éclairage est réalisée par une simulation de Monte-Carlo pour estimer la valeur d'un pixel. Un grand nombre d'échantillons peut être nécessaire pour obtenir une image photo-réaliste de qualité. L'utilisation d'un nombre réduit d'échantillons peut restituer une image dans laquelle un bruit résiduel est perceptible humainement. L'objectif des travaux de cette thèse a été d'explorer les méthodes émanant de l'apprentissage automatique pour caractériser et détecter la présence de bruit dans les images. La démarche proposée dans le cadre de cette thèse a été la suivante : (i) conceptualisation d'une base d'images et recueil de seuils humains subjectifs servant de base d'apprentissage, (ii) création de modèles d'apprentissage automatique orientés apprentissage profond à partir de cette base d'images, (iii) recherche de caractéristiques fines liées au bruit résiduel de Monte-Carlo puis (iv) validation perceptuelle des meilleurs modèles développés. Des travaux connexes, notamment relatifs à la gestion d'un bruit spécifique dans les images nommé *firefly*, ont été proposés, tout comme l'application d'une méthode permettant de cibler les caractéristiques de bruit étudié.

Plusieurs perspectives de travaux peuvent être envisagées à la suite des contributions et explorations réalisées dans le cadre de cette thèse.

Extension de la base d'images et benchmark des modèles de perception du bruit

Les données générées ont été obtenues avec la version 3 de PBRT orientée sur CPU. Nous nous étions limités à un nombre maximum de 10 000 échantillons par pixel pour chacune des images pour générer cette base d'images en un temps convenable. Cette limite a entraîné le fait que, pour certains blocs de l'image, le seuil humain subjectif avait été fixé à 10 000, soit la valeur maximale, car une présence de bruit résiduel était encore perceptible.

Dans un premier temps, nous envisageons d'exploiter la nouvelle version du moteur de rendu PBRT (version 4) qui permet l'utilisation de carte de calculs GPU accélérant

considérablement le rendu d'images. Il est ainsi possible de générer des images avec un moteur de rendu tracé de chemins jusque 100 000 échantillons par pixel pour un temps équivalent à la précédente version du moteur de rendu pour 10 000 échantillons. Cette première extension de la base d'image vise l'augmentation du nombre d'échantillons permettant d'obtenir des images de référence où le bruit devrait être totalement imperceptible, notamment sur les images où les seuils étaient fixés à 10 000 échantillons précédemment. Nous souhaitons également en perspective, fournir des données supplémentaires pour toutes les images, de manière à permettre des études statistiques sur des données de grande taille, telles que les informations géométriques de chaque point de vue (carte de normales, carte de profondeur, etc.) qui pourraient être exploitées par les modèles de perception. Ces données sont aussi fortement utilisées pour la conception de modèle de débruitage et permettraient donc à la base d'images de posséder un usage double : conception de modèles de perception et de modèles de débruitage.

Dans le chapitre 10, nous avons pu constater l'impact que peuvent avoir les *fireflies* sur la perception du bruit, et dans notre cas particulier sur les différences perçues entre deux images. Pour éviter ce genre de biais, nous envisageons de générer les images qui présentent des *fireflies* avec l'estimateur *G*-MoN proposé dans le chapitre 6, en se posant la question de savoir s'il faut toutes les générer avec cet estimateur ou non. D'autant plus qu'il faudra réaliser une analyse précise de l'impact du bruit résiduel de cet estimateur qui risque d'être différent de l'estimateur classique, notamment en début de calcul de l'image.

Nous avons dans cette étude opté pour la mesure de seuils subjectifs humains auprès d'une population experte. Cibler une telle population qui est notamment davantage sensible au bruit résiduel de MC présent dans les images, a pour objectif d'obtenir un modèle assez restrictif à la présence de bruit. Nous envisageons également de capturer des seuils perceptifs avec des non experts, afin de disposer de deux modèles permettant de générer des images à coût « faible » pour un utilisateur moyen ou à coût plus élevé pour un spécialiste. Étant donné que nous souhaitons obtenir des modèles de perception de bruit à partir de ces données étiquetées, nous pouvons imaginer que le choix du modèle de perception peut se faire en fonction du besoin et des contraintes lors de la production des images de synthèse. En effet, exploiter un modèle ayant appris sur une population non experte proposera des critères d'arrêt de calcul dans certaines zones de l'image où potentiellement du bruit pour un expert peut encore être perceptible. Toutefois, il peut permettre d'obtenir des images en un temps plus raisonnable et où le bruit ne serait pas forcément perçu pour la population moyenne. Il pourrait être également intéressant d'imaginer une combinaison des deux modèles par pondération de leur prédiction en fonction de l'usage souhaité.

La base d'images offre une liste non exhaustive des effets lumineux possibles en synthèse d'images. Alimenter davantage cette base d'images est l'une de nos perspectives

afin de couvrir au maximum les effets possibles et d'obtenir un modèle qui généralise correctement. Ainsi, nous envisageons d'augmenter la diversité de la base et d'accroître le nombre d'images disponibles, en interrogeant éventuellement les membres de la communauté sur les images qui seraient intéressantes.

Enfin, la base d'images actuelle propose uniquement des images générées avec une méthode de tracé de chemins classique. Nous ne savons pas à quel point, les modèles de perception de bruit obtenus avec cette base d'images, sont spécifiques ou non à la méthode de génération des images utilisée. Il est envisagé de proposer une version de la base d'images avec d'autres intégrateurs de Monte-Carlo (BDPT, MLT, etc.). Cela, introduirait forcément davantage de temps de calcul, mais permettrait de générer un modèle de perception spécifique à chaque intégrateur ou de généraliser un tel modèle de perception à une gamme plus large d'intégrateurs.

Enfin, suite aux extensions possibles, la nouvelle version de cette base d'images a pour vocation à être de nouveau exploitée pour l'apprentissage de nouveaux modèles de perception de bruit comme ceux proposés dans les chapitres 7 et 8 et améliorer encore plus la généralisation des modèles. L'objectif final de cette nouvelle version de la base d'images, est la proposition d'un benchmark de modèles de perception de bruit visant à ouvrir davantage la problématique liée au bruit résiduel de MC.

Étude des estimateurs pour la gestion des *fireflies*

Les travaux futurs concernant la gestion des *fireflies* devraient se concentrer sur d'autres estimateurs récents basés également sur l'estimateur MoN. Parmi ces estimateurs, on peut citer : l'estimateur de la médiane des moyennes invariant par permutation (POERSCHMANN 2021), l'estimateur MoN par tournoi (LUGOSI et al. 2019) ou l'estimateur MoN bayésien (ORENSTEIN 2019), afin de déterminer s'ils pourraient avoir un intérêt dans la suppression des *fireflies* ou la réduction du bruit. Ensuite, nous envisageons d'étudier plus en profondeur le choix de la meilleure valeur pour le coefficient de Gini. Nous avons constaté expérimentalement que $G = 0,25$ semble fournir de bons résultats globalement sur les images. Cependant, il pourrait être pertinent d'étudier si sa valeur pourrait être ajustée plus localement en fonction des distributions d'échantillons qui peuvent être très différentes entre les pixels. Comme l'illustre la méthode de Zirr, l'information de voisinage est capable d'améliorer les résultats des estimateurs. Nous allons donc étudier les moyens d'utiliser ces informations sur les pixels voisins (données tampons, coefficients de Gini, etc.) afin d'améliorer le taux de convergence de G-MoN.

Il pourrait être également intéressant d'étudier l'utilisation du coefficient de Gini pour la détection du bruit et/ou le débruitage, en raison de sa capacité à juger de l'égalité dans un ensemble de données. De plus, l'utilisation du coefficient de Gini calculé sur les M tampons pourrait également être intéressante dans un rendu progressif. Elle permettrait

de constituer une carte d'échantillonnage adaptative par rapport à la disparité encore présente dans les valeurs de contribution des pixels.

Une autre étude possible, serait l'utilisation de l'estimateur *G*-MoN dans le cas de séquences d'images où généralement l'apparition de *firefly* et de bruit présent dans la suite d'images est hautement perceptible. Il serait donc intéressant de voir à quel point ce nouvel estimateur serait capable ou non de réduire cet effet désagréable.

Extensions des modèles de perception de bruit

Les modèles de perception de bruit dont les analyses de performances figurent dans le chapitre 10 ont pour chacun des pistes d'amélioration. Nous envisageons ainsi pour le modèle proposé basé sur la *SVD-Entropy* de nous intéresser à d'autres méthodes de factorisation et compression des données. Notamment les méthodes en d'analyse en composantes principales (ACP), analyse en composantes indépendantes (ACI) ou encore la factorisation de matrices non négatives (NMF), qui visent à réduire la dimension des données en entrées. Pour le modèle GGN, où un léger surapprentissage a pu être identifié, nous souhaitons étudier différentes approches récentes pour éviter et réduire ce défaut (MUKHERJEE et al. 2020 ; YAZICI et al. 2020). Notamment l'approche proposée par (MUKHERJEE et al. 2020), qui utilise plusieurs paires générateur-discriminateur pour empêcher le modèle de mémoriser trop fortement l'ensemble d'apprentissage.

Par la suite, ces modèles ont pour vocation à être étendus à la détection de bruit dans les images de synthèse stéréoscopiques. Un appareil proposant une vision stéréoscopique (3D) d'un environnement virtuel nécessite a minima deux images, une pour chaque œil, pour permettre à notre cerveau de représenter un effet de relief. La perception du bruit dans un appareil stéréoscopique risque d'être très différente d'un appareil standard 2D. La méthode proposée de récolte de seuils subjectifs humains présentée dans le chapitre 5 a permis de créer des modèles de perception du bruit assez fidèles à la vérité terrain et donc à la perception du bruit résiduel de MC. Afin d'entreprendre et d'étendre ces travaux à un contexte 3D, il est nécessaire, si l'on envisage une même procédure, de récolter des seuils humains subjectifs sur des images affichées dans un appareil stéréoscopique.

Ainsi, dans le cadre du projet ANR PrISE-3D, Quentin Huan a pu réaliser un stage ayant pour objectif de proposer plusieurs expériences permettant d'obtenir des seuils subjectifs dans un tel environnement 3D. L'une de ces expériences est réalisée dans un casque de réalité virtuel de type *HTC Vive*, composée de deux images (droite et gauche), la seconde sur un écran *alioscopy*, proposant une représentation du relief à partir de l'écran directement, mais nécessitant huit points de vue différents.

Étendre les travaux à partir des données récoltées par Quentin Huan, consisterait à considérer un volume de données a minima double, étant donné le bruit présent dans

deux images. Outre la quantité de données plus conséquente, il est important de les traiter judicieusement. En effet, dans un contexte de représentation 3D à partir de deux images (gauche et droite), le bruit résiduel de MC présent dans l'image de gauche peut être considérablement différent de celui de l'image de droite.

L'étude de la cohérence temporelle des modèles est aussi un problème important. Lors du calcul d'une séquence d'images, les variations entre images au sein d'une vidéo sont connues pour provoquer des artefacts dont certains peuvent-être hautement perceptibles ou entraîner des sensations de malaise. Le système visuel humain est fortement sensible à ces variations. L'apparition d'un firefly sera par exemple associée à un clignotement dans l'image. Dans un contexte un peu différent, il a été montré que l'emploi de méthodes évolutives pour le calcul d'éclairage direct peut permettre de réduire le bruit visuel. Cependant, cette réduction peut s'accompagner d'effets de scintillements parfois très brusques qui nuisent à l'efficacité de la méthode (YUKSEL 2019). Aussi, il sera important d'étudier ce problème pour chaque méthode et de les comparer entre-elles sur ce problème spécifique de la cohérence visuelle dans une séquence.

Sélection d'attributs par l'optimisation assistée par méta-modèle

L'utilisation de la formalisation du problème de sélection d'attributs en apprentissage automatique en un problème d'optimisation a permis de montrer que la sélection des attributs était liée à la nature du modèle d'apprentissage utilisée pour la prédiction. L'exemple purement exploratoire de l'utilisation d'une forêt aléatoire comme modèle d'apprentissage automatique était choisi pour rendre l'étude possible en un temps convenable. L'estimateur des performances d'une solution (modèle appris sur un certain nombre d'attributs) prenait un temps moyen de l'ordre de ≈ 20 secondes, ce qui reste raisonnable et ce qui malheureusement ne permettait pas au méta-modèle (surrogate) d'être compétitif en temps pour un ordre $L = 2$.

Dans un premier temps, nous souhaitons pousser cette étude à l'apprentissage de modèle SVM, c'est-à-dire utiliser le modèle SVM comme fonction d'évaluation réelle coûteuse et utiliser le méta-modèle pour apprendre de cette fonction et l'approximer. En effet, la forêt aléatoire reste un modèle d'apprentissage simple et avec un temps d'apprentissage relativement rapide. Un modèle de type SVM impose une contrainte coûteuse forte en temps d'apprentissage qui peut, en fonction du nombre d'attributs sélectionnés être assez grand (lié à la complexité d'apprentissage du modèle en $O(n^3)$.

Comme mentionné, il n'est pas possible en l'état d'entraîner un modèle SVM. Toutefois, il peut être envisageable dans le problème d'optimisation de fixer un nombre maximum k d'attributs à donner en entrée au modèle sur les n disponibles. Cette contrainte

peut être fixée comme un critère de validation ou non d'une solution nouvellement générée. Ainsi, avec un nombre réduit d'attributs, il pourrait être envisagé de réaliser un entraînement de SVM en un temps plus convenable et de l'intégrer dans le processus de recherche d'optimisation coûteuse.

Bibliographie

ABDIANSAH, A. & WARDOYO, R. (2015). Time complexity analysis of support vector machines (SVM) in LibSVM. *International journal computer and application*, *128*(3), 28-34.

ALON, N., MATIAS, Y. & SZEGEDY, M. (1999). The Space Complexity of Approximating the Frequency Moments. *Journal of Computer and System Sciences*, *58*(1), 137-147.

ALTMANN, A., TOLOŞI, L., SANDER, O. & LENGAUER, T. (2010). Permutation importance : a corrected feature importance measure. *Bioinformatics*, *26*(10), 1340-1347. https://doi.org/10.1093/bioinformatics/btq134

A.M.RUFAI, C.ANBARJAFARI & H.DEMIREL. (2014). Lossy image compression using singular value decomposition and wavelet difference reduction. *Digit.SignalProcess. 24, 117-123*.

ARVO, J. & KIRK, D. (1990). Particle transport and image synthesis, In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, Dallas, TX, USA, Association for Computing Machinery. https://doi.org/10.1145/97879.97886

BACCOUCHE, M., MAMALET, F., WOLF, C., GARCIA, C. & BASKURT, A. (2010). Action classification in soccer videos with long short-term memory recurrent neural networks, In *International Conference on Artificial Neural Networks*. Springer.

BAKO, S., VOGELS, T., MCWILLIAMS, B., MEYER, M., NOVÁK, J., HARVILL, A., SEN, P., DEROSE, T. & ROUSSELLE, F. (2017). Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Transactions on Graphics*, *36*(4), 1-14. https://doi.org/10.1145/3072959.3073708

BANERJEE, M. & PAL, N. R. [Nikhil R]. (2014a). Feature selection with SVD entropy : Some modification and extension. *Information Sciences*, *264*, 118-134.

BANERJEE, M. & PAL, N. R. [Nikhil R.]. (2014b). Feature selection with SVD entropy : some modification and extension. *Information Sciences*, *264*, 118-134. https://doi.org/10.1016/j.ins.2013.12.029

BAPTISTA, R. & POLOCZEK, M. (2018). Bayesian optimization of combinatorial structures, In *International Conference on Machine Learning*. PMLR.

BELCOUR, L., SOLER, C., SUBR, K., HOLZSCHUCH, N. & DURAND, F. (2013). 5D Covariance Tracing for Efficient Defocus and Motion Blur. *ACM Transactions on Graphics*, *32*(3), Article No. 31. https://doi.org/10.1145/2487228.2487239

BENNETT, K. P. & BREDENSTEINER, E. J. (2000). Duality and geometry in SVM classifiers, In *ICML*. Citeseer.

BERKSON, J. (1944). Application of the Logistic Function to Bio-Assay. *Journal of the American Statistical Association*, *39*(227), 357-365. https://doi.org/10.2307/2280041

BITTERLI, B. (2016). Rendering resources [https ://benedikt-bitterli.me/resources/].

BITTERLI, B., ROUSSELLE, F., MOON, B., IGLESIAS-GUITIÁN, J. A., ADLER, D., MITCHELL, K., JAROSZ, W. & NOVÁK, J. (2016). Nonlinearly weighted first-order regression for denoising monte carlo renderings. *Computer Graphics Forum*, *35*(4), 107-117. https://doi.org/10.1111/cgf.12954

BLAIR, C. (1985). Problem Complexity and Method Efficiency in Optimization (A. S. Nemirovsky and D. B. Yudin). *SIAM Review*, *27*(2), 264-265.

BLAKEMORE, C. & CAMPBELL, F. W. (1969). On the existence of neurones in the human visual system selectively sensitive to the orientation and size of retinal images. *The Journal of Physiology*, *203*(1), 237-260. https://doi.org/https://doi.org/10.1113/jphysiol.1969.sp008862

BOSSE, S., MANIRY, D., MÜLLER, K.-R., WIEGAND, T. & SAMEK, W. (2017). Deep neural networks for no-reference and full-reference image quality assessment. *IEEE Transactions on image processing*, *27*(1), 206-219.

BOUGHIDA, M. & BOUBEKEUR, T. (2017). Bayesian collaborative denoising for monte carlo rendering. *Computer Graphics Forum*, *36*(4), 137-153. https://doi.org/10.1111/cgf.13231

BRADLEY, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145-1159. https://doi.org/10.1016/S0031-3203(96)00142-2

BREIMAN, L. (2001). Random forests. *Machine learning*, *45*(1), 5-32.

BRIOL, F.-X., OATES, C. J., GIROLAMI, M. & OSBORNE, M. A. (2015). Frank-Wolfe Bayesian Quadrature : Probabilistic Integration with Theoretical Guarantees. *arXiv :1506.02681 [stat]*arxiv 1506.02681. http://arxiv.org/abs/1506.02681

BROWNLEES, C., JOLY, E. & LUGOSI, G. (2015). Empirical risk minimization for heavy-tailed losses. *The Annals of Statistics*, *43*(6).

BUADES, A., COLL, B. & MOREL, J.-M. (2005). A Non-Local Algorithm for Image Denoising, In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, IEEE. https://doi.org/10.1109/CVPR.2005.38

BUBECK, S., CESA-BIANCHI, N. & LUGOSI, G. (2013). Bandits With Heavy Tail. *IEEE Transactions on Information Theory*, *59*(11), 7711-7717.

BÜHLMANN, P. (2003). Bagging, Subagging and Bragging for Improving some Prediction Algorithms. In *Recent Advances and Trends in Nonparametric Statistics* (p. 19-34). Elsevier.

BUISINE, J., DELEPOULLE, S. & RENAUD, C. (2021). Minimalist And Customisable Optimisation Package. *Journal of Open Source Software*, *6*(59), 2812. https://doi.org/10.21105/joss.02812

BUISINE, J., DELEPOULLE, S., SYNAVE, R. & RENAUD, C. (2021). Subjective human thresholds over computer generated images. Zenodo. https://doi.org/10.5281/zenodo.4964303

BUJA, A., STUETZLE, W. & SHEN, Y. (2005). Loss functions for binary class probability estimation and classification : Structure and applications. *Working draft, November*, *3*.

CAMPBELL, F. W. & ROBSON, J. G. (1968). Application of fourier analysis to the visibility of gratings. *The Journal of Physiology*, *197*(3), 551-566. https://doi.org/https://doi.org/10.1113/jphysiol.1968.sp008574

CAO, Y., CHEN, G., JING, G., STIEBITZ, M. & TOFT, B. (2019). Graph edge coloring : A survey. *Graphs and Combinatorics*, *35*(1), 33-66.

CARNEC, M., LE CALLET, P. & BARBA, D. (2008). Objective quality assessment of color images based on a generic perceptual reduced reference. *Signal Processing : Image Communication*, *23*(4), 239-256.

CHAITANYA, C. R. A., KAPLANYAN, A. S., SCHIED, C., SALVI, M., LEFOHN, A., NOWROUZEZAHRAI, D. & AILA, T. (2017). Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics*, *36*(4), 1-12. https://doi.org/10.1145/3072959.3073601

CHAN, Y. M. & HE, X. (1994). A simple and competitive estimator of location. *Statistics & Probability Letters*, *19*(2), 137-142.

CHANG, C.-C. & LIN, C.-J. (2011). LIBSVM : a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, *2*(3), 1-27.

CONSTANTIN, J., BIGAND, A., CONSTANTIN, I. & HAMAD, D. (2015). Image noise detection in global illumination methods based on FRVM. *Neurocomputing*, *164*, 82-95. https://doi.org/10.1016/j.neucom.2014.10.090

CONSTANTIN, J., CONSTANTIN, I., BIGAND, A. & HAMAD, D. (2016). Perception of noise in global illumination based on inductive learning, In *2016 International Joint Conference on Neural Networks (IJCNN)*. 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, IEEE. https://doi.org/10.1109/IJCNN.2016.7727861

COOK, R. L. [Robert L.], PORTER, T. & CARPENTER, L. (1984). Distributed ray tracing, In *Proceedings of the 11th annual conference on Computer graphics and inter-*

active techniques, New York, NY, USA, Association for Computing Machinery. https://doi.org/10.1145/800031.808590

COOK, R. L. [Robert L] & TORRANCE, K. E. (1982). A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, *1*(1), 7-24.

CORTES, C. & VAPNIK, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

CRAVEN, B. (1970). A generalization of Lagrange multipliers. *Bulletin of the Australian Mathematical Society*, *3*(3), 353-362.

DALY, S. J. (1992). Visible differences predictor : an algorithm for the assessment of image fidelity, In *Human Vision, Visual Processing, and Digital Display III*. Human Vision, Visual Processing, and Digital Display III, International Society for Optics ; Photonics. https://doi.org/10.1117/12.135952

DAMGAARD, C. (2003). *"Gini Coefficient." From MathWorld–A Wolfram Web Resource*. https://mathworld.wolfram.com/GiniCoefficient.html

DAMGAARD, C. & WEINER, J. (2000). Describing inequality in plant size or fecundity. *Ecology*, *81*(4), 1139-1142. https://doi.org/https://doi.org/10.1890/0012-9658(2000)081[1139:DIIPSO]2.0.CO;2

DAMILANO, G. & PUIG, P. (2004). Efficiency of a Linear Combination of the Median and the Sample Mean : The Double Truncated Normal Distribution. *Scandinavian Journal of Statistics*, *31*(4), 629-637.

DAMMERTZ, H., SEWTZ, D., HANIKA, J. & LENSCH, H. P. A. (2010). Edge-avoiding &#xc0 ;-Trous wavelet transform for fast global illumination filtering, In *Proceedings of the Conference on High Performance Graphics*, Saarbrucken, Germany, Eurographics Association.

DARST, B. F., MALECKI, K. C. & ENGELMAN, C. D. (2018). Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC Genetics*, *19*(1), 65. https://doi.org/10.1186/s12863-018-0633-8

DAŞ, G. S., GZARA, F. & STÜTZLE, T. (2020). A review on airport gate assignment problems : Single versus multi objective approaches. *Omega*, *92*, 102146.

DECORO, C., WEYRICH, T. & RUSINKIEWICZ, S. (2010). Density-based outlier rejection in monte carlo rendering. *Computer Graphics Forum*, *29*(7), 2119-2125. https://doi.org/https://doi.org/10.1111/j.1467-8659.2010.01799.x

DELBRACIO, M., MUSÉ, P., BUADES, A., CHAUVIER, J., PHELPS, N. & MOREL, J.-M. (2014). Boosting monte carlo rendering by ray histogram fusion. *ACM Transactions on Graphics*, *33*(1), 1-15. https://doi.org/10.1145/2532708

DELORME, A. & FLÜCKIGER, M. (2014). Perception et réalité : une introduction à la psychologie des perceptions [OCLC : 907646979]. De Boeck Supérieur.

DENTON, E. J. & PIRENNE, M. H. (1954). The absolute sensitivity and functional stability of the human eye. *The Journal of physiology*, *123*(3), 417-442.

DIOLATZIS, S., GRUSON, A., JAKOB, W., NOWROUZEZAHRAI, D. & DRETTAKIS, G. (2020). Practical Product Path Guiding Using Linearly Transformed Cosines. *Computer Graphics Forum*, *39*(4), 23-33.

DORFMAN, R. (1979). A formula for the Gini coefficient. *The review of economics and statistics*, 146-149.

DRAGESCO, J. (1995). *High resolution astrophotography* (T. 7). Cambridge University Press.

EBRAHIMI KAHOU, S., MICHALSKI, V., KONDA, K., MEMISEVIC, R. & PAL, C. (2015). Recurrent neural networks for emotion recognition in video, In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*.

ECKHARDT, R. (1987). Stan ulam, john von neumann, and the monte carlo method. *Los Alamos Science*, *15*(30), 131-136.

EGAN, K., DURAND, F. & RAMAMOORTHI, R. (2011). Practical filtering for efficient ray-traced directional occlusion, In *Proceedings of the 2011 SIGGRAPH Asia Conference*, Hong Kong, China, Association for Computing Machinery. https://doi.org/10.1145/2024156.2024214

EGAN, K., HECHT, F., DURAND, F. & RAMAMOORTHI, R. (2011). Frequency analysis and sheared filtering for shadow light fields of complex occluders. *ACM Transactions on Graphics*, *30*(2), 1-13. https://doi.org/10.1145/1944846.1944849

EGAN, K., TSENG, Y.-T., HOLZSCHUCH, N., DURAND, F. & RAMAMOORTHI, R. (2009). Frequency analysis and sheared reconstruction for rendering motion blur. *ACM Transactions on Graphics*, *28*(3), 1-13. https://doi.org/10.1145/1531326.1531399

ELEK, O., THOMAS, M. M. & FORBES, A. (2019). Learning Patterns in Sample Distributions for Monte Carlo Variance Reduction. *arXiv :1906.00124 [cs]*arxiv 1906.00124. http://arxiv.org/abs/1906.00124

FERWERDA, J. (2001). Elements of early vision for computer graphics. *IEEE Computer Graphics and Applications*, *21*(5), 22-33. https://doi.org/10.1109/38.946628

FRUIT, R. (2019). *Exploration-exploitation dilemma in Reinforcement Learning under various form of prior knowledge* (thèse de doct.). Université de Lille 1, Sciences et Technologies ; CRIStAL UMR 9189.

GAYATHRI, S., GOPI, V. P. & PALANISAMY, P. (2020). A lightweight CNN for Diabetic Retinopathy classification from fundus images. *Biomedical Signal Processing and Control*, *62*, 102115.

GHARBI, M., LI, T.-M., AITTALA, M., LEHTINEN, J. & DURAND, F. (2019). Sample-based Monte Carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics*, *38*(4), 125 :1-125 :12. https://doi.org/10.1145/3306346.3322954

G.H.GOLUB & LOAN, C. (1983). *Matrix Computations* (J. H. U. PRESS, Éd.). MD, Baltimore.

GOODFELLOW, I. J., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A. & BENGIO, Y. (2014). Generative Adversarial

Networks. *arXiv :1406.2661 [cs, stat]*arxiv 1406.2661. http://arxiv.org/abs/1406.2661

GREGOR, K., DANIHELKA, I., GRAVES, A., REZENDE, D. J. & WIERSTRA, D. (2015). Draw : A recurrent neural network for image generation. *arXiv preprint arXiv :1502.04623*.

GROUP, V. I. S. R. Et al. (1997). Sarnoff JND vision model algorithm description and testing. *Sarnoff Corporation, Princeton, NJ*.

GRUSON, A. (2015). *Toward more realism and robustness in global illumination* (thèse de doct.). Université Rennes 1.

GUESTRIN, C. (2006). SVMs, Duality and the Kernel Trick (cont.) *Machine Learning*, *10701*, 15781.

HACHISUKA, T., JAROSZ, W., WEISTROFFER, R. P., DALE, K., HUMPHREYS, G., ZWICKER, M. & JENSEN, H. W. (2008). Multidimensional adaptive sampling and reconstruction for ray tracing, In *ACM SIGGRAPH 2008 papers*, Los Angeles, California, Association for Computing Machinery. https://doi.org/10.1145/1399504.1360632

HALKO, N., MARTINSSON, P.-G. & TROPP, J. A. (2010). Finding structure with randomness : Probabilistic algorithms for constructing approximate matrix decompositions [version : 2]. *arXiv :0909.4061 [math]*arxiv 0909.4061. http://arxiv.org/abs/0909.4061

HAMPEL, F. R. (1971). A general qualitative definition of robustness. *The Annals of Mathematical Statistics*, *42*(6), 1887-1896.

HANSEN, P. & JAUMARD, B. (1990). Algorithms for the maximum satisfiability problem. *Computing*, *44*(4), 279-303.

HAO, Q., MA, L., SBERT, M., FEIXAS, M. & ZHANG, J. (2020). Gaze Information Channel in Van Gogh's Paintings. *Entropy*, *22*(5), 540.

HASELMANN, M., GRUBER, D. P. & TABATABAI, P. (2018). Anomaly Detection Using Deep Learning Based Image Completion, In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. https://doi.org/10.1109/ICMLA.2018.00201

HASTIE, T., TIBSHIRANI, R. & WAINWRIGHT, M. (2019). *Statistical learning with sparsity : the lasso and generalizations*. Chapman ; Hall/CRC.

H.C.ANDREWS & C.L.PATTERSON. (1976). Singular vvalue decompositions and digital image processing. *IEEE Trans. on Accoustics, Speech and Signal Processing*.

HE, K., ZHANG, X., REN, S. & SUN, J. (2015). Deep Residual Learning for Image Recognition. *arXiv :1512.03385 [cs]*arxiv 1512.03385. http://arxiv.org/abs/1512.03385

HEARST, M., DUMAIS, S., OSUNA, E., PLATT, J. & SCHOLKOPF, B. (1998). Support vector machines. *IEEE Intelligent Systems and their Applications*, *13*(4), 18-28. https://doi.org/10.1109/5254.708428

HINTON, G. E. & SALAKHUTDINOV, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504-507. https://doi.org/10.1126/science.1127647

HOCHREITER, S. & SCHMIDHUBER, J. [Jürgen]. (1997a). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

HOCHREITER, S. & SCHMIDHUBER, J. [Jürgen]. (1997b). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

HSU, D. & SABATO, S. (2016). Loss minimization and parameter estimation with heavy tails [arXiv : 1307.1827 version : 7]. *arXiv :1307.1827 [cs, stat]*.

HSU, P.-L. & ROBBINS, H. (1947). Complete convergence and the law of large numbers. *Proceedings of the National Academy of Sciences of the United States of America*, *33*(2), 25.

HUBER, P. J. & RONCHETTI, E. (2009). Robust Statistics. 2nd John Wiley & Sons. *Hoboken, NJ*.

HUNT, N. & TYRRELL, S. (2001). Stratified sampling. *Retrieved November*, *10*, 2012.

HUO, Y. & YOON, S.-e. (2021). A survey on deep learning-based monte carlo denoising. *Computational Visual Media*, *7*(2), 169-185. https://doi.org/10.1007/s41095-021-0209-9

ITTI, L. [L.], KOCH, C. & NIEBUR, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(11), 1254-1259. https://doi.org/10.1109/34.730558

ITTI, L. [Laurent] & KOCH, C. (2001). Computational modelling of visual attention. *Nature Reviews Neuroscience*, *2*(3), 194-203. https://doi.org/10.1038/35058500

JENSEN, H. W. (2001). *Realistic image synthesis using photon mapping*. USA, A. K. Peters, Ltd.

JERRUM, M. R., VALIANT, L. G. & VAZIRANI, V. V. (1986). Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, *43*, 169-188.

JIANG, G. & KAINZ, B. (2019). Deep Radiance Caching : Convolutional Autoencoders Deeper in Ray Tracing [version : 1]. *arXiv :1910.02480 [cs]*arxiv 1910.02480. http://arxiv.org/abs/1910.02480

JOLLIFFE, I. (2005). Principal component analysis. *Encyclopedia of statistics in behavioral science*.

JUNG, J. W., MEYER, G. & DELONG, R. (2015). Robust statistical pixel estimation. *Computer Graphics Forum*, *34*(2), 585-596. https://doi.org/https://doi.org/10.1111/cgf.12586

KAJIYA, J. T. (1986). The rendering equation, In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*.

KALANTARI, N. K., BAKO, S. & SEN, P. (2015). A machine learning approach for filtering Monte Carlo noise. *ACM Transactions on Graphics*, *34*(4), 122 :1-122 :12. https://doi.org/10.1145/2766977

KANG, L., YE, P., LI, Y. & DOERMANN, D. (2014). Convolutional Neural Networks for No-Reference Image Quality Assessment, In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, IEEE. https://doi.org/10.1109/CVPR.2014.224

KAY, D. S. & GREENBERG, D. (1979). Transparency for computer synthesized images, In *Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, Chicago, Illinois, USA, Association for Computing Machinery. https://doi.org/10.1145/800249.807438

KELLY, D. (1975). Spatial frequency selectivity in the retina. *Vision Research*, *15*(6), 665-672. https://doi.org/10.1016/0042-6989(75)90282-5

KINGMA, D. P. & WELLING, M. (2014). Auto-Encoding Variational Bayes. *arXiv :1312.6114 [cs, stat]*arxiv 1312.6114. http://arxiv.org/abs/1312.6114

K.KONSTANTINIDES, B.NATARAJAN & G.S.YOVANOV. (1997). Noise estimation and filtering using block-based singular value decomposition. *IEEE Trans. on Image Processing 6, 479-483*.

KOCH, C. [C.] & ULLMAN, S. (1985). Shifts in selective visual attention : towards the underlying neural circuitry. *Human Neurobiology*, *4*(4), 219-227.

KONSTANTINIDES, K., NATARAJAN, B. & YOVANOF, G. S. (1997). Noise estimation and filtering using block-based singular value decomposition. *IEEE Transactions on Image Processing*, *6*(3), 479-483.

KOULIERIS, G. A., DRETTAKIS, G., CUNNINGHAM, D., SIDORAKIS, N. & MANIA, K. (2014). Context-aware material selective rendering for mobile graphics, In *ACM SIGGRAPH 2014 Posters*, Vancouver, Canada, Association for Computing Machinery. https://doi.org/10.1145/2614217.2614246

KÜMMERER, M. & BETHGE, M. (2021). State-of-the-Art in Human Scanpath Prediction. *arXiv :2102.12239 [cs]*arxiv 2102.12239. http://arxiv.org/abs/2102.12239

KÜMMERER, M., BYLINSKII, Z., JUDD, T., BORJI, A., ITTI, L., DURAND, F., OLIVA, A. & TORRALBA, A. (p. d.). MIT/Tübingen Saliency Benchmark.

KÜMMERER, M., WALLIS, T. S. A. & BETHGE, M. (2018). Saliency Benchmarking Made Easy : Separating Models, Maps and Metrics (V. FERRARI, M. HEBERT, C. SMINCHISESCU & Y. WEISS, Éd.). In V. FERRARI, M. HEBERT, C. SMINCHISESCU & Y. WEISS (Éd.), *Computer Vision – ECCV 2018*, Springer International Publishing.

KUO, B.-C., HO, H.-H., LI, C.-H., HUNG, C.-C. & TAUR, J.-S. (2013). A kernel-based feature selection method for SVM with RBF kernel for hyperspectral image

classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *7*(1), 317-326.

KUZNETSOV, A., KALANTARI, N. K. & RAMAMOORTHI, R. (2018). Deep adaptive sampling for low sample count rendering. *Computer Graphics Forum*, *37*(4), 35-44. https://doi.org/https://doi.org/10.1111/cgf.13473

LAFORTUNE, E. P. & WILLEMS, Y. D. (1993). Bi-directional path tracing.

LAI, T. L. & ROBBINS, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, *6*(1), 4-22.

LAVOUÉ, G. & MANTIUK, R. (2015). Quality assessment in computer graphics. In *Visual signal quality assessment* (p. 243-286). Springer.

LEBRUN, M., BUADES, A. & MOREL, J. M. (2013). A nonlocal bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, *6*(3), 1665-1688. https://doi.org/10.1137/120874989

LEHTINEN, J., AILA, T., CHEN, J., LAINE, S. & DURAND, F. (2011). Temporal light field reconstruction for rendering distribution effects. *ACM Transactions on Graphics*, *30*(4), 55 :1-55 :12. https://doi.org/10.1145/2010324.1964950

LEHTINEN, J., AILA, T., LAINE, S. & DURAND, F. (2012). Reconstructing the indirect light field for global illumination. *ACM Transactions on Graphics*, *31*(4), 51 :1-51 :10. https://doi.org/10.1145/2185520.2185547

LEPRÊTRE, F. (2020). *Méta-modélisation, simulation et optimisation de flux urbains* (Theses 2020DUNK0571). Université du Littoral Côte d'Opale. https://tel.archives-ouvertes.fr/tel-03178850

LEPRÊTRE, F., FONLUPT, C., VEREL, S. & MARION, V. (2018). SIALAC benchmark : on the design of adaptive algorithms for traffic lights problems, In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, Kyoto, Japan, Association for Computing Machinery. https://doi.org/10.1145/3205651.3205776

LEPRÊTRE, F., TEYTAUD, F. & DEHOS, J. (2017). Multi-Armed Bandit for Stratified Sampling : Application to Numerical Integration [ISSN : 2376-6824], In *2017 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. 2017 Conference on Technologies and Applications of Artificial Intelligence (TAAI). ISSN : 2376-6824. https://doi.org/10.1109/TAAI.2017.34

LEPRÊTRE, F., VEREL, S., FONLUPT, C. & MARION, V. (2019). Walsh functions as surrogate model for pseudo-boolean optimization problems, In *Proceedings of the Genetic and Evolutionary Computation Conference*.

LEWIS, E. E. & MILLER, W. F. (1984). Computational methods of neutron transport. http://inis.iaea.org/Search/search.aspx?orig_q=RN:17089238

LI, T.-M., WU, Y.-T. & CHUANG, Y.-Y. (2012). SURE-based optimization for adaptive sampling and reconstruction. *ACM Transactions on Graphics*, *31*(6), 1. https://doi.org/10.1145/2366145.2366213

LI, Y., PO, L.-M., FENG, L. & YUAN, F. (2016). No-reference image quality assessment with deep convolutional neural networks, In *2016 IEEE International Conference on Digital Signal Processing (DSP)*. IEEE.

LIN, J. (1991). Divergence measures based on the Shannon entropy. *IEEE Transactions on Information theory*, *37*(1), 145-151.

LIN, W. [Weiheng], WANG, B., WANG, L. & HOLZSCHUCH, N. (2020). A detail preserving neural network model for monte carlo denoising. *Computational Visual Media*, *6*(2), 157-168. https://doi.org/10.1007/s41095-020-0167-7

LIOU, C.-Y., CHENG, W.-C., LIOU, J.-W. & LIOU, D.-R. (2014). Autoencoder for words. *Neurocomputing*, *139*, 84-96. https://doi.org/10.1016/j.neucom.2013.09.055

LIU, J., PÉREZ-LIÉBANA, D. & LUCAS, S. M. (2017). Bandit-based random mutation hill-climbing, In *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE.

LIU, P., QIU, X. & HUANG, X. (2016). Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv :1605.05101*.

LIU, W. (2014). Additive white Gaussian noise level estimation based on block SVD, In *2014 IEEE Workshop on Electronics, Computer and Applications*. IEEE.

LIU, W. & LIN, W. [Weisi]. (2012). Additive white Gaussian noise level estimation in SVD domain for images. *IEEE Transactions on Image processing*, *22*(3), 872-883.

LIU, Y., ZHENG, C., ZHENG, Q. & YUAN, H. (2018). Removing monte carlo noise using a sobel operator and a guided image filter. *The Visual Computer*, *34*(4), 589-601. https://doi.org/10.1007/s00371-017-1363-z

LONGHURST, P., DEBATTISTA, K. & CHALMERS, A. (2006). A GPU based saliency map for high-fidelity selective rendering, In *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, Cape Town, South Africa, Association for Computing Machinery. https://doi.org/10.1145/1108590.1108595

LU, Y., XIE, N. & SHEN, H. T. (2020). DMCR-GAN : Adversarial Denoising for Monte Carlo Renderings with Residual Attention Networks and Hierarchical Features Modulation of Auxiliary Buffers, In *SIGGRAPH Asia 2020 Technical Communications*, Virtual Event, Republic of Korea, Association for Computing Machinery. https://doi.org/10.1145/3410700.3425426

LUBIN, J. & FIBUSH, D. (1997). Sarnoff JND vision model. T1A1.

LUGOSI, G. & MENDELSON, S. (2019). Risk minimization by median-of-means tournaments.

MANNOS, J. & SAKRISON, D. (1974). The effects of a visual fidelity criterion of the encoding of images. *IEEE Transactions on Information Theory*, *20*(4), 525-536. https://doi.org/10.1109/TIT.1974.1055250

MANTIUK, R., KIM, K. J., REMPEL, A. G. & HEIDRICH, W. (2011). HDR-VDP-2 : a calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Transactions on Graphics*, *30*(4), 40 :1-40 :14. https://doi.org/10.1145/2010324.1964935

MCCOOL, M. D. (1999). Anisotropic diffusion for Monte Carlo noise reduction. *ACM Transactions on Graphics*, *18*(2), 171-194. https://doi.org/10.1145/318009.318015

MCNAMARA, A. (2001). Visual perception in realistic image synthesis. *Computer Graphics Forum*, *20*(4), 211-224. https://doi.org/https://doi.org/10.1111/1467-8659.00550

METROPOLIS, N. & ULAM, S. (1949). The Monte Carlo Method. *Journal of the American Statistical Association*, *44*(247), 335-341. https://doi.org/10.1080/01621459.1949.10483310

MEYER, M. & ANDERSON, J. (2006). Statistical acceleration for animated global illumination, In *ACM SIGGRAPH 2006 Papers*, Boston, Massachusetts, Association for Computing Machinery. https://doi.org/10.1145/1179352.1141996

MICHAELS, D. D. (1956). The nature of the photoreceptor process. *Optometry and Vision Science*, *33*(2), 59-76. https://doi.org/10.1097/00006324-195602000-00001

MIKOLOV, T., KOMBRINK, S., BURGET, L., ČERNOCKỲ, J. & KHUDANPUR, S. (2011). Extensions of recurrent neural network language model, In *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE.

MITCHELL, T. M. Et al. (p. d.). Machine learning.

MOON, B., CARR, N. & YOON, S.-E. (2014). Adaptive rendering based on weighted local regression. *ACM Transactions on Graphics*, 13.

MOON, B., MCDONAGH, S., MITCHELL, K. & GROSS, M. (2016). Adaptive polynomial rendering. *ACM Transactions on Graphics*, *35*(4), 1-10. https://doi.org/10.1145/2897824.2925936

MOORTHY, A. K. & BOVIK, A. C. (2011). Blind Image Quality Assessment : From Natural Scene Statistics to Perceptual Quality. *IEEE Transactions on Image Processing*, *20*(12), 3350-3364. https://doi.org/10.1109/TIP.2011.2147325

MOORTHY, A. & BOVIK, A. (2010). A Two-Step Framework for Constructing Blind Image Quality Indices. *IEEE Signal Processing Letters*, *17*(5), 513-516. https://doi.org/10.1109/LSP.2010.2043888

MUKHERJEE, S., XU, Y., TRIVEDI, A. & FERRES, J. L. (2020). Protecting GANs against privacy attacks by preventing overfitting.

MÜLLER, T., GROSS, M. & NOVÁK, J. (2017). Practical Path Guiding for Efficient Light-Transport Simulation. *Computer Graphics Forum*, *36*(4), 91-100.

MUNIGLIA, M., VEREL, S., PALLEC, J.-C. L. & DO, J.-M. (2017). Massive asynchronous master-worker EA for nuclear reactor optimization : a fitness landscape perspec-

tive, In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*.

MUNKBERG, J. & HASSELGREN, J. (2020). Neural denoising with layer embeddings. *Computer Graphics Forum*, *39*(4), 1-12. https://doi.org/https://doi.org/10.1111/cgf.14049

MYSZKOWSKI, K. (1998). The visible differences predictor : applications to global illumination problems. In G. DRETTAKIS & N. MAX (Éd.), *Rendering techniques '98* (p. 223-236). Vienna, Springer Vienna. https://doi.org/10.1007/978-3-7091-6453-2_21

NARWARIA, M., MANTIUK, R. K., DA SILVA, M. P. & LE CALLET, P. (2015). HDR-VDP-2.2 : a calibrated method for objective quality prediction of high-dynamic range and standard images. *Journal of Electronic Imaging*, *24*(1), 010501. https://doi.org/10.1117/1.JEI.24.1.010501

NIEBUR, E. (2007). Saliency map. *Scholarpedia*, *2*(8), 2675. https://doi.org/10.4249/scholarpedia.2675

NIEBUR, E. & KOCH, C. [Christof]. (1996). Control of Selective Visual Attention : Modeling the "Where" Pathway. In D. S. TOURETZKY, M. C. MOZER & M. E. HASSELMO (Éd.), *Advances in Neural Information Processing Systems 8 (NIPS 1995)* (p. 802-808). Cambridge, MA, MIT Press. https://resolver.caltech.edu/CaltechAUTHORS:20160223-153004900

O.ALTER, P.O.BROWN & D.BOLSTEIN. (2000). Singular value decomposition for genome-wide expression data processing and modeling. *Proc.Natl.Acad.Sci. (PNAS) USA97(18)*.

OREN, M. & NAYAR, S. K. (1994). Generalization of Lambert's reflectance model, In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, New York, NY, USA, Association for Computing Machinery. https://doi.org/10.1145/192161.192213

ORENSTEIN, P. (2019). Robust Mean Estimation with the Bayesian Median of Means [arXiv : 1906.01204]. *arXiv :1906.01204 [math, stat]*.

O'SHEA, K. & NASH, R. (2015). An Introduction to Convolutional Neural Networks. *arXiv :1511.08458 [cs]*arxiv 1511.08458. http://arxiv.org/abs/1511.08458

OVERBECK, R. S., DONNER, C. & RAMAMOORTHI, R. (2009). Adaptive wavelet rendering, In *ACM SIGGRAPH Asia 2009 papers*, Yokohama, Japan, Association for Computing Machinery. https://doi.org/10.1145/1661412.1618486

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V. Et al. (2011). Scikit-learn : Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830.

PHARR, M., JAKOB, W. & HUMPHREYS, G. (2016). *Physically based rendering : from theory to implementation* [Google-Books-ID : iNMVBQAAQBAJ]. Morgan Kaufmann.

POERSCHMANN, J. (2021). *Mean estimation : median of means* [Medium]. https://github.com/jakobap/Median-of-Means-Estimator

PONOMARENKO, N., JIN, L., IEREMEIEV, O., LUKIN, V., EGIAZARIAN, K., ASTOLA, J., VOZEL, B., CHEHDI, K., CARLI, M., BATTISTI, F. & JAY KUO, C.-C. (2015). Image database TID2013 : peculiarities, results and perspectives. *Signal Processing : Image Communication*, *30*, 57-77. https://doi.org/10.1016/j.image.2014.10.009

PRAVIN, C. & OJHA, V. (2020). A Novel ECG Signal Denoising Filter Selection Algorithm Based on Conventional Neural Networks, In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. https://doi.org/10.1109/ICMLA51294.2020.00176

PRESS, W. H. & FARRAR, G. R. (1990). Recursive stratified sampling for multidimensional monte carlo integration. *Computers in Physics*, *4*(2), 190. https://doi.org/10.1063/1.4822899

PURKAYASTHA, R., CHAKRABORTY, T., SAHA, A. & MUKHOPADHYAY, D. (2020). Study and Analysis of Various Heuristic Algorithms for Solving Travelling Salesman Problem—A Survey, In *Proceedings of the Global AI Congress 2019*. Springer, Singapore.

RASMUSSEN, C. E. & GHAHRAMANI, Z. (2003). Bayesian monte carlo. *Advances in neural information processing systems*, 505-512.

RISH, I. (2001). *An empirical study of the naive bayes classifier*.

RITSCHEL, T., ENGELHARDT, T., GROSCH, T., SEIDEL, H.-P., KAUTZ, J. & DACHSBACHER, C. (2009). Micro-rendering for scalable, parallel final gathering, In *ACM SIGGRAPH Asia 2009 papers*, Yokohama, Japan, Association for Computing Machinery. https://doi.org/10.1145/1661412.1618478

RONNEBERGER, O., FISCHER, P. & BROX, T. (2015). U-net : convolutional networks for biomedical image segmentation (N. NAVAB, J. HORNEGGER, W. M. WELLS & A. F. FRANGI, Éd.). In N. NAVAB, J. HORNEGGER, W. M. WELLS & A. F. FRANGI (Éd.), *Medical image computing and computer-assisted intervention – MICCAI 2015*, Cham, Springer International Publishing. https://doi.org/10.1007/978-3-319-24574-4_28

ROSENBLATT, F. (1958). The Perceptron : A Probabilistic Model for Information Storage and Organization in The Brain. *Psychological Review*, 65-386.

ROSS, J. E., CLARKE, D. D. & BRON, A. J. (1985). Effect of age on contrast sensitivity function : uniocular and binocular findings. *British Journal of Ophthalmology*, *69*(1), 51-56. https://doi.org/10.1136/bjo.69.1.51

ROUSSELLE, F., KNAUS, C. & ZWICKER, M. (2011). Adaptive sampling and reconstruction using greedy error minimization. *ACM Transactions on Graphics*, *30*(6), 1-12. https://doi.org/10.1145/2070781.2024193

ROUSSELLE, F., KNAUS, C. & ZWICKER, M. (2012). Adaptive rendering with non-local means filtering. *ACM Transactions on Graphics*, *31*(6), 1-11. https://doi.org/10.1145/2366145.2366214

ROUSSELLE, F., MANZI, M. & ZWICKER, M. (2013). Robust denoising using feature and color information. *Computer Graphics Forum*, *32*(7), 121-130. https://doi.org/https://doi.org/10.1111/cgf.12219

ROUSSELOT, M., LE MEUR, O., COZOT, R. & DUCLOUX, X. (2019). Quality assessment of hdr/wcg images using hdr uniform color spaces. *Journal of Imaging*, *5*(1), 18.

RUSHMEIER, H., WARD, G., PIATKO, C., SANDERS, P. & RUST, B. (1995). Comparing real and synthetic images : some ideas about metrics (P. M. HANRAHAN & W. PURGATHOFER, Éd.). In P. M. HANRAHAN & W. PURGATHOFER (Éd.), *Rendering techniques '95*, Vienna, Springer. https://doi.org/10.1007/978-3-7091-9430-0_9

RUSHMEIER, H. E. & WARD, G. J. (1994). Energy preserving non-linear filters, In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, New York, NY, USA, Association for Computing Machinery. https://doi.org/10.1145/192161.192189

SAE-BAE, N. & UDOMHUNSAKUL, S. (2007). Noise suppression using block-based singular value decomposition filtering, In *2007 Asia-Pacific Conference on Communications*. IEEE.

SCORNET, E. (2020). Trees, forests, and impurity-based variable importance. *arXiv :2001.04295 [math, stat]*arxiv 2001.04295. http://arxiv.org/abs/2001.04295

SEGOVIA, B., IEHL, J. C., MITANCHEY, R. & PÉROCHE, B. (2006). *Non-interleaved deferred shading of interleaved sample patterns*. The Eurographics Association. https://doi.org/10.2312/EGGH/EGGH06/053-060

SEN, P. & DARABI, S. (2012). On filtering the noise from the random parameters in Monte Carlo rendering. *ACM Transactions on Graphics*, *31*(3), 18 :1-18 :15. https://doi.org/10.1145/2167076.2167083

SEN, P., ZWICKER, M. [Matthias], ROUSSELLE, F., YOON, S.-E. & KALANTARI, N. K. (2015). Denoising your Monte Carlo renders : recent advances in image-space adaptive sampling and reconstruction, In *ACM SIGGRAPH 2015 Courses*, Los Angeles, California, Association for Computing Machinery. https://doi.org/10.1145/2776880.2792740

SHARMA, S. & SHARMA, S. (2017). Activation functions in neural networks. *Towards Data Science*, *6*(12), 310-316.

SHEIKH, H. R., BOVIK, A. C. & CORMACK, L. (2005). No-reference quality assessment using natural scene statistics : JPEG2000. *IEEE Transactions on image processing*, *14*(11), 1918-1927.

SHEIKH, H., WANG, Z., CORMACK, L. & BOVI, A. (2005). Live image quality assessment database release 2. *http ://live.ece.utexas.edu/research/quality*. https://ci.nii.ac.jp/naid/10030938153/

SHI, H., 0003, L. W., TANG, W., ZHENG, N. & HUA, G. (2020). Loss Functions for Person Image Generation., In *BMVC*.

SHIRLEY, P., AILA, T., COHEN, J., ENDERTON, E., LAINE, S., LUEBKE, D. & MCGUIRE, M. (2011). A local image reconstruction algorithm for stochastic rendering, In *Symposium on Interactive 3D Graphics and Games*, San Francisco, California, Association for Computing Machinery. https://doi.org/10.1145/1944745.1944747

SHIRLEY, P., WANG, C. & ZIMMERMAN, K. (1996). Monte Carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, *15*(1), 1-36. https://doi.org/10.1145/226150.226151

SMITH, J. M. & MAYNARD, S. J. (1993). *The theory of evolution*. Cambridge University Press.

SOBOL', I. M. (1967). On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, *7*(4), 784-802.

SOLER, C., SUBR, K., DURAND, F., HOLZSCHUCH, N. & SILLION, F. (2009). Fourier depth of field. *ACM Transactions on Graphics*, *28*(2), 18 :1-18 :12. https://doi.org/10.1145/1516522.1516529

SPANIER, J. & GELBARD, E. M. (2008). *Monte carlo principles and neutron transport problems* [Google-Books-ID : SrFFEgzMJpoC]. Courier Corporation.

SRIVASTAVA, N. (2013). Improving neural networks with dropout. *University of Toronto*, *182*(566), 7.

STANSBURY, D. E., NASELARIS, T. & GALLANT, J. L. (2013). Natural scene statistics account for the representation of scene categories in human visual cortex. *Neuron*, *79*(5), 1025-1034.

STEIN, C. M. (1981). Estimation of the Mean of a Multivariate Normal Distribution. *The Annals of Statistics*, *9*(6), 1135-1151. https://www.jstor.org/stable/2240405

SUTSKEVER, I., MARTENS, J. & HINTON, G. E. (2011). Generating text with recurrent neural networks, In *ICML*.

TAKOUACHET, N., DELEPOULLE, S., RENAUD, C., ZOGHLAMI, N. & TAVARES, J. M. R. (2017). Perception of noise and global illumination : toward an automatic stopping criterion based on SVM. *Computers & Graphics*, *69*, 49-58. https://doi.org/10.1016/j.cag.2017.09.008

TALBI, E.-G. (2009). *Metaheuristics : from design to implementation* (T. 74). John Wiley & Sons.

TERRELL, G. R. & SCOTT, D. W. (1992). Variable kernel density estimation. *The Annals of Statistics*, 1236-1265.

THOMAS, J. P. (1975). Spatial resolution and spatial interaction. *Handbook of perception*, *5*, 233-264.

TORRANCE, K. E. & SPARROW, E. M. (1992). Theory for off-specular reflection from roughened surfaces. In *Radiometry* (p. 32-41). USA, Jones ; Bartlett Publishers, Inc.

TSUCHIYA, N. & KOCH, C. (2008). Attention and consciousness. *Scholarpedia*, *3*(5), 4173. https://doi.org/10.4249/scholarpedia.4173

VAPNIK, V. (2006). *Estimation of dependences based on empirical data*. New York, Springer-Verlag. https://doi.org/10.1007/0-387-34239-7

VEACH, E. (1998). *Robust monte carlo methods for light transport simulation* (thèse de doct.) [AAI9837162 ISBN-10 : 0591907801]. Stanford University. Stanford, CA, USA. AAI9837162 ISBN-10 : 0591907801.

VEACH, E. & GUIBAS, L. (1995). Bidirectional estimators for light transport (G. SAKAS, S. MÜLLER & P. SHIRLEY, Éd.). In G. SAKAS, S. MÜLLER & P. SHIRLEY (Éd.), *Photorealistic rendering techniques*, Berlin, Heidelberg, Springer. https://doi.org/10.1007/978-3-642-87825-1_11

VEACH, E. & GUIBAS, L. J. (1997). Metropolis light transport, In *Proceedings of the 24th annual conference on computer graphics and interactive techniques - SIGGRAPH '97*. the 24th annual conference, Not Known, ACM Press. https://doi.org/10.1145/258734.258775

VEREL, S., DERBEL, B., LIEFOOGHE, A., AGUIRRE, H. & TANAKA, K. (2018). A surrogate model based on walsh decomposition for pseudo-boolean functions, In *International conference on parallel problem solving from nature*. Springer.

VICINI, D., ADLER, D., NOVÁK, J., ROUSSELLE, F. & BURLEY, B. (2019). Denoising deep monte carlo renderings. *Computer Graphics Forum*, *38*(1), 316-327. https://doi.org/10.1111/cgf.13533

VINCENT, P., LAROCHELLE, H., BENGIO, Y. & MANZAGOL, P.-A. (2008). Extracting and composing robust features with denoising autoencoders, In *Proceedings of the 25th international conference on Machine learning*, Helsinki, Finland, Association for Computing Machinery. https://doi.org/10.1145/1390156.1390294

VOGELS, T., ROUSSELLE, F., MCWILLIAMS, B., RÖTHLIN, G., HARVILL, A., ADLER, D., MEYER, M. & NOVÁK, J. (2018). Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics*, *37*(4), 1-15. https://doi.org/10.1145/3197517.3201388

VON NEUMANN, J. & ULAM, S. (1951). Monte carlo method. *National Bureau of Standards Applied Mathematics Series*, *12*(1951), 36.

VORBA, J., HANIKA, J., HERHOLZ, S., MÜLLER, T., KŘIVÁNEK, J. & KELLER, A. (2019). Path guiding in production. In *ACM SIGGRAPH 2019 Courses* (p. 1-77).

WALKER, H. M. (1958). The Contributions of Karl Pearson. *Journal of the American Statistical Association*, *53*(281), 11-22. https://doi.org/10.1080/01621459.1958.10501419

WALSH, J. L. (1923). A closed set of normal orthogonal functions. *American Journal of Mathematics*, *45*(1), 5-24.

WALTER, B., MARSCHNER, S. R., LI, H. & TORRANCE, K. E. (2007). Microfacet models for refraction through rough surfaces, In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, Grenoble, France, Eurographics Association.

WANG, S. [S.], DENG, C., LIN, W., ZHAO, B. & CHEN, J. (2013). A novel SVD-based image quality assessment metric [ISSN : 2381-8549], In *2013 IEEE International Conference on Image Processing*. 2013 IEEE International Conference on Image Processing. ISSN : 2381-8549. https://doi.org/10.1109/ICIP.2013.6738087

WANG, S. [Shuigen], DENG, C., LIN, W., ZHAO, B. & CHEN, J. (2013). A novel SVD-based image quality assessment metric, In *2013 IEEE International Conference on Image Processing*. IEEE.

WANG, Z. [Z.], BOVIK, A., SHEIKH, H. & SIMONCELLI, E. (2004). Image quality assessment : from error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600-612. https://doi.org/10.1109/TIP.2003.819861

WANG, Z. [Zhou], SIMONCELLI, E. P. & BOVIK, A. C. (2003). Multiscale structural similarity for image quality assessment, In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Ieee.

WATKINS, C. J. C. H. & DAYAN, P. (1992). Q-learning. *Machine Learning*, *8*(3), 279-292. https://doi.org/10.1007/BF00992698

WEHRL, A. (1978). General properties of entropy. *Reviews of Modern Physics*, *50*(2), 221.

WELSTEAD, S. T. (1999). *Fractal and wavelet image compression techniques* (T. 40). Spie Press.

WHITTED, T. (1979). An improved illumination model for shaded display, In *Proceedings of the 6th annual conference on Computer graphics and interactive techniques*.

WONG, K.-M. & WONG, T.-T. (2019). Deep residual learning for denoising monte carlo renderings. *Computational Visual Media*, *5*(3), 239-255. https://doi.org/10.1007/s41095-019-0142-3

XU, B., ZHANG, J., WANG, R., XU, K., YANG, Y.-L., LI, C. & TANG, R. (2019). Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Transactions on Graphics*, *38*(6), 224 :1-224 :12. https://doi.org/10.1145/3355089.3356547

YAN, K. & ZHANG, D. (2015). Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B : Chemical*, *212*, 353-363. https://doi.org/10.1016/j.snb.2015.02.025

YANG, X., WANG, D., HU, W., ZHAO, L.-J., YIN, B.-C., ZHANG, Q., WEI, X.-P. & FU, H. (2019). DEMC : a deep dual-encoder network for denoising monte carlo rendering. *Journal of Computer Science and Technology*, *34*(5), 1123-1135. https://doi.org/10.1007/s11390-019-1964-2

YAZICI, Y., FOO, C.-S., WINKLER, S., YAP, K.-H. & CHANDRASEKHAR, V. (2020). Empirical Analysis of Overfitting and Mode Drop in GAN Training, In *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE.

YE, P., KUMAR, J., KANG, L. & DOERMANN, D. (2013). Real-Time No-Reference Image Quality Assessment Based on Filter Learning [ISSN : 1063-6919], In *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2013 IEEE Conference on Computer Vision and Pattern Recognition. ISSN : 1063-6919. https://doi.org/10.1109/CVPR.2013.132

YEE, H., PATTANAIK, S. & GREENBERG, D. P. (2001). Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Transactions on Graphics*, *20*(1), 39-65. https://doi.org/10.1145/383745.383748

YEE, H. Y., DUTRÉ, P. & PATTANAIK, S. N. (2002). *Fundamentals of Lighting and Perception : The Rendering of Physically Accurate Images. make better games YOU make better games*.

YOUNG, T. (1802). II. The Bakerian Lecture. On the theory of light and colours. *Philosophical Transactions of the Royal Society of London*, *92*, 12-48. https://doi.org/10.1098/rstl.1802.0004

YUKSEL, C. (2019). Stochastic Lightcuts, In *High-Performance Graphics (HPG 2019)*, Strasbourg, France, The Eurographics Association. https://doi.org/10.2312/hpg.20191192

ZAEFFERER, M., STORK, J. & BARTZ-BEIELSTEIN, T. (2014). Distance measures for permutations in combinatorial efficient global optimization, In *International Conference on Parallel Problem Solving from Nature*. Springer.

ZAEFFERER, M., STORK, J., FRIESE, M., FISCHBACH, A., NAUJOKS, B. & BARTZ-BEIELSTEIN, T. (2014). Efficient global optimization for combinatorial problems, In *Proceedings of the 2014 annual conference on genetic and evolutionary computation*.

ZAREMBA, W., SUTSKEVER, I. & VINYALS, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv :1409.2329*.

ZHAO, H., GALLO, O., FROSIO, I. & KAUTZ, J. (2017). Loss Functions for Image Restoration With Neural Networks. *IEEE Transactions on Computational Imaging*, *3*(1), 47-57. https://doi.org/10.1109/TCI.2016.2644865

ZHU, S. (2020). Survey : Machine Learning in Production Rendering. *arXiv :2005.12518 [cs]*arxiv 2005.12518. http://arxiv.org/abs/2005.12518

ZIRR, T., HANIKA, J. & DACHSBACHER, C. (2018). Re-weighting firefly samples for improved finite-sample monte carlo estimates. *Computer Graphics Forum*, *37*(6), 410-421. https://doi.org/https://doi.org/10.1111/cgf.13335

ZWICKER, M. [M.], JAROSZ, W., LEHTINEN, J., MOON, B., RAMAMOORTHI, R., ROUSSELLE, F., SEN, P., SOLER, C. & YOON, S.-E. (2015). Recent advances in adaptive sampling and reconstruction for monte carlo rendering. *Computer Graphics Forum*, *34*(2), 667-681. https://doi.org/https://doi.org/10.1111/cgf.12592

Annexes

Annexe A

Base d'images de synthèse

A.1 Seuils subjectifs des participants de l'expérience

Scène	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Arcsphere	1511	993	1096	1511	2209	1047	1358	1702	2547	1993	1764	2031	1244	1253	1687	1753
Bathroom	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	9673	7880
Bunny Fur	2483	2493	1293	1910	2383	1770	860	1063	3197	1070	800	1550	1377	1893	2323	1637
Car2	624	1002	1431	1478	1031	3304	3704	3522	833	693	820	964	804	502	620	582
Caustic	2292	3690	2882	1735	2015	2062	1535	2108	2508	3065	3215	3160	2485	2562	3385	3262
Classroom	10000	10000	10000	10000	8220	10000	10000	10000	10000	9280	10000	9220	8347	9607	8073	9153
Classroom 1	9673	10000	10000	10000	8880	8347	10000	8873	8747	7947	9407	10000	8740	9340	8940	9880
Coffee Splash	1400	1428	1625	1602	1700	1975	1650	1582	1508	1802	1878	1920	1578	1830	1858	1752
Contemporary Bathroom vue 1	10000	10000	10000	10000	3087	10000	10000	10000	10000	10000	9013	9073	9667	9040	3060	3293
Contemporary Bathroom vue 2	2033	10000	9487	7053	3500	10000	6620	6700	2460	10000	6907	8140	10000	10000	3067	2967
Crown	875	3955	2180	805	1655	3305	2932	2090	2808	1402	1642	2868	2755	2362	2282	2460
Dragon	2032	1832	1882	1912	2058	1735	1430	1958	2032	1143	2922	3572	3335	3765	2585	2862
Dragon	664	700	1093	903	1053	964	1143	1130	1227	1133	1607	1403	1067	967	963	997
Ecosys	127	133	200	150	140	240	190	170	233	180	400	147	247	300	133	280
Eponge Fractal vue 1	387	330	312	390	447	645	480	527	1037	997	557	1263	830	1237	1303	703
Eponge Fractal vue 2	798	920	980	944	1002	946	833	1144	784	828	938	1006	842	924	870	1060
Ganesha	1640	1100	1132	1535	740	520	418	538	568	490	442	630	1212	565	735	1340
Glass Of Water	5150	2076	2206	3072	4764	3906	3048	3990	6068	5462	6022	6134	4064	5762	5642	5842
Glass	10000	9740	420	420	10000	10000	353	353	9007	10000	353	487	8807	10000	4747	753
Indirect	1743	1110	1850	4443	1103	1000	1980	3943	670	1003	2043	2343	700	1310	1477	1907
Kitchen vue 1	4083	5863	4720	4086	3309	5343	5483	2711	2903	5294	4791	3877	4037	4626	5120	5091
Kitchen vue 2	7820	9210	10000	9310	7620	7210	7520	9510	9310	9110	6810	8620	8110	7120	6920	8420
Landscape	1534	2194	2440	3436	896	1140	1722	2068	1266	1498	1242	1408	466	898	1062	1082
Living Room vue 1	8680	5287	6553	2293	7153	7087	6420	2627	4420	6580	4560	7820	7087	8287	6287	6693
Living Room vue 2	4020	6213	6553	7147	5153	4820	4220	2953	5220	3560	3153	5020	4153	3553	5680	7480
Living Room vue 3	6638	6278	6378	5978	4790	4544	4684	3964	6438	5342	5244	6340	6138	5624	4784	4064
Living Room vue 4	7020	7218	6898	6148	6380	5764	6860	5584	5104	6320	7118	6918	5660	5980	6638	6638
Low table	1220	4953	3420	3220	7087	7553	7553	6753	1887	2353	2487	2753	1020	1820	953	753
Pavilion Day vue 1	2437	2469	2954	2157	1751	2100	3097	1923	1289	1097	1151	1269	1409	1629	1443	1409
Pavilion Day vue 2	2344	3076	3431	3853	1073	2987	3516	4364	1180	867	1542	2742	1424	1202	1031	962
Pavilion Day vue 3	1666	1517	1323	914	1143	1609	2123	2037	857	1323	2066	2069	531	429	703	666
Pavilion Night vue 1	5400	5444	6000	5556	6200	5667	5622	5473	5029	3807	3184	3362	3607	3496	4051	2382
Pavilion Night vue 2	7028	6252	5685	6150	6752	6002	6950	6628	5902	6015	6278	6705	6202	5555	4455	5180
Pavilion Night vue 3	3290	1605	1692	2507	3080	5180	5069	2796	4253	2702	2991	2367	1160	2090	1515	1302
Sanmiguel vue 1	9867	6947	7353	6080	3693	4887	2833	4953	3620	4620	2900	3553	2420	5887	1687	3420
Sanmiguel vue 2	5087	6353	8087	8620	8073	8480	5820	6487	2620	4887	5087	6287	3420	3220	2687	2353
Staircase 1	368	674	390	434	497	697	760	1097	477	437	886	714	437	467	657	831
Staircase 2	2473	2096	2607	2762	2407	2384	2940	3076	2673	3653	2542	3318	4407	2496	2478	3229
Tt	767	1277	1600	1003	2160	3440	2943	1877	2340	1610	1800	1330	863	1040	1100	803
Vw Van	1351	1266	1126	1437	1923	1809	5409	5440	1666	2523	1609	1640	949	1409	1351	1580

TABLEAU A.1 – Moyenne des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image.

A.2 Écart-type des seuils des participants de l'expérience

Scène	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Arcsphere	939	578	573	878	1011	765	943	917	1458	1192	1178	912	616	706	946	1025
Bathroom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	462	1122
Bunny Fur	1203	1314	585	974	1481	981	417	443	3383	497	356	931	606	1316	3449	674
Car2	574	772	843	789	529	1900	2240	2223	736	544	978	1117	808	432	757	670
Caustic	1167	1790	1129	866	1015	855	687	1000	1295	1642	1658	1272	1391	1192	1609	1544
Classroom	0	0	0	0	1844	0	0	0	0	894	0	849	1301	556	2725	1197
Classroom 1	462	0	0	0	1080	2075	0	1593	1151	1899	839	0	1508	933	1499	170
Coffee Splash	1054	1044	1567	1457	1476	1821	1693	1071	688	976	1460	1289	1081	1005	1203	729
Contemporary Bathroom vue 1	0	0	0	0	1731	0	0	0	0	0	808	1311	471	1358	1673	1319
Contemporary Bathroom vue 2	653	0	616	2093	1311	0	2391	2480	656	0	2523	2630	0	0	1984	1630
Crown	683	2645	1422	687	1101	2949	2078	910	1426	1250	1297	1366	2401	1551	1489	1173
Dragon	633	574	651	546	851	1487	1885	692	931	1097	1372	1325	1183	1607	1250	1446
Dragon	275	316	925	499	1090	549	975	926	994	724	830	989	829	630	886	726
Ecosys	52	50	0	50	49	118	10	30	125	20	200	41	111	216	50	227
Eponge Fractal vue 1	293	275	245	470	340	783	680	839	1332	1214	663	1467	832	1465	1439	699
Eponge Fractal vue 2	609	658	895	751	670	540	740	720	641	512	877	790	747	767	749	907
Ganesha	1087	911	1114	873	355	349	271	297	261	311	246	419	599	387	682	978
Glass Of Water	3183	2881	3228	2844	3221	3179	2808	3146	3085	3239	3133	3393	3233	3599	3217	3470
Glass	0	368	163	163	0	0	94	94	1405	0	94	249	1688	0	3966	471
Indirect	2028	653	749	2897	1083	695	1242	2269	368	692	1014	991	776	757	837	1148
Kitchen vue 1	2801	3947	3452	2670	2139	4111	3879	1980	2194	3745	3628	2855	3333	3060	3890	3450
Kitchen vue 2	0	790	0	690	1600	2790	1700	490	690	890	3190	200	1890	2300	1100	400
Landscape	1607	2455	2215	2859	893	1206	1587	1594	1188	1452	925	948	466	1057	727	702
Living Room vue 1	933	471	1112	624	1508	94	1178	1306	1296	331	2038	163	1247	1257	249	1472
Living Room vue 2	748	2689	1676	2097	2604	1558	1497	1181	1558	1213	1087	1497	1636	1408	3082	2417
Living Room vue 3	3133	3180	3231	3395	3076	2913	2932	2695	3105	2989	2760	3212	3365	2662	2698	2622
Living Room vue 4	3237	3480	3288	3048	3296	2920	3470	3092	3113	3299	3693	3794	3383	3429	3703	3611
Low table	432	618	849	980	838	1389	1652	1857	471	411	249	189	283	163	94	249
Pavilion Day vue 1	1622	1650	2013	1154	975	1614	2591	967	1124	645	770	523	1138	1489	1263	1240
Pavilion Day vue 2	1833	2407	2520	2978	653	2319	2413	3091	839	609	941	1365	1196	992	640	1062
Pavilion Day vue 3	1393	1125	1108	857	718	1091	1497	1025	546	905	1564	1117	544	239	688	335
Pavilion Night vue 1	4117	3506	3617	4107	3449	3890	3993	3045	3061	2305	1304	2081	1906	2611	2410	2351
Pavilion Night vue 2	2374	2827	3233	3459	2741	2628	2722	2905	2774	2479	2486	2035	3099	2406	3026	2807
Pavilion Night vue 3	3194	1100	948	2326	2938	3780	3425	1957	3529	1816	2316	1553	1139	1435	1192	1284
Sanmiguel vue 1	189	3389	2045	2630	1808	525	763	984	712	993	453	411	1143	660	189	748
Sanmiguel vue 2	1320	1934	1893	1131	2725	1078	1657	573	163	822	957	189	748	864	411	525
Staircase 1	236	511	353	416	273	457	485	436	279	223	535	369	289	201	367	474
Staircase 2	1385	1085	1119	1495	1588	1625	2551	2556	1001	1877	1395	2106	1307	1708	760	2542
Tt	433	644	884	492	1300	1669	1221	928	1396	775	802	630	447	642	781	439
Vw Van	531	734	486	630	576	685	2215	2091	597	919	483	558	455	652	471	545

TABLEAU A.2 – Écart-type des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image.

A.3 Coefficient de variation de l'expérience

Scène	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Arcsphere	0.62	0.58	0.52	0.58	0.46	0.73	0.69	0.54	0.57	0.60	0.67	0.45	0.50	0.56	0.56	0.58
Bathroom	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.14
Bunny Fur	0.48	0.53	0.45	0.51	0.62	0.55	0.48	0.42	1.06	0.46	0.45	0.60	0.44	0.70	1.48	0.41
Car2	0.92	0.77	0.59	0.53	0.51	0.58	0.60	0.63	0.88	0.78	1.19	1.16	1.00	0.86	1.22	1.15
Caustic	0.51	0.49	0.39	0.50	0.50	0.41	0.45	0.47	0.52	0.54	0.52	0.40	0.56	0.47	0.48	0.47
Classroom	0.00	0.00	0.00	0.00	0.22	0.00	0.00	0.00	0.00	0.10	0.00	0.09	0.16	0.06	0.34	0.13
Classroom 1	0.05	0.00	0.00	0.00	0.12	0.25	0.00	0.18	0.13	0.24	0.09	0.00	0.17	0.10	0.17	0.02
Coffee Splash	0.75	0.73	0.96	0.91	0.87	0.92	1.03	0.68	0.46	0.54	0.78	0.67	0.69	0.55	0.65	0.42
Contemporary Bathroom vue 1	0.00	0.00	0.00	0.00	0.56	0.00	0.00	0.00	0.00	0.00	0.09	0.14	0.05	0.15	0.55	0.40
Contemporary Bathroom vue 2	0.32	0.00	0.06	0.30	0.37	0.00	0.36	0.37	0.27	0.00	0.37	0.32	0.00	0.00	0.65	0.55
Crown	0.78	0.67	0.65	0.85	0.67	0.89	0.71	0.44	0.51	0.89	0.79	0.48	0.87	0.66	0.65	0.48
Dragon	0.31	0.31	0.35	0.29	0.41	0.86	1.32	0.35	0.46	0.96	0.47	0.37	0.35	0.43	0.48	0.51
Dragon	0.41	0.45	0.85	0.55	1.04	0.57	0.85	0.82	0.81	0.64	0.52	0.70	0.78	0.65	0.92	0.73
Ecosys	0.41	0.37	0.00	0.33	0.35	0.49	0.05	0.18	0.53	0.11	0.50	0.28	0.45	0.72	0.37	0.81
Eponge Fractal vue 1	0.76	0.83	0.79	1.21	0.76	1.21	1.42	1.59	1.29	1.22	1.19	1.16	1.00	1.18	1.10	0.99
Eponge Fractal vue 2	0.76	0.72	0.91	0.80	0.67	0.57	0.89	0.63	0.82	0.62	0.93	0.78	0.89	0.83	0.86	0.86
Ganesha	0.66	0.83	0.98	0.57	0.48	0.67	0.65	0.55	0.46	0.63	0.56	0.67	0.49	0.68	0.93	0.73
Glass Of Water	0.62	1.39	1.46	0.93	0.68	0.81	0.92	0.79	0.51	0.59	0.52	0.55	0.80	0.62	0.57	0.59
Glass	0.00	0.04	0.39	0.39	0.00	0.00	0.27	0.27	0.16	0.00	0.27	0.51	0.19	0.00	0.84	0.63
Indirect	1.16	0.59	0.40	0.65	0.98	0.70	0.63	0.58	0.55	0.69	0.50	0.42	1.11	0.58	0.57	0.60
Kitchen vue 1	0.69	0.67	0.73	0.65	0.65	0.77	0.71	0.73	0.76	0.71	0.76	0.74	0.83	0.66	0.76	0.68
Kitchen vue 2	0.00	0.09	0.00	0.07	0.21	0.39	0.23	0.05	0.07	0.10	0.47	0.02	0.23	0.32	0.16	0.05
Landscape	1.05	1.12	0.91	0.83	1.00	1.06	0.92	0.77	0.94	0.97	0.74	0.67	1.00	1.18	0.68	0.65
Living Room vue 1	0.11	0.09	0.17	0.27	0.21	0.01	0.18	0.50	0.29	0.05	0.45	0.02	0.18	0.15	0.04	0.22
Living Room vue 2	0.19	0.43	0.26	0.29	0.51	0.32	0.35	0.40	0.30	0.34	0.34	0.30	0.39	0.40	0.54	0.32
Living Room vue 3	0.47	0.51	0.51	0.57	0.64	0.64	0.63	0.68	0.48	0.56	0.53	0.51	0.55	0.47	0.56	0.65
Living Room vue 4	0.46	0.48	0.48	0.50	0.52	0.51	0.51	0.55	0.61	0.52	0.52	0.55	0.60	0.57	0.56	0.54
Low table	0.35	0.12	0.25	0.30	0.12	0.18	0.22	0.27	0.25	0.17	0.10	0.07	0.28	0.09	0.10	0.33
Pavilion Day vue 1	0.67	0.67	0.68	0.53	0.56	0.77	0.84	0.50	0.87	0.59	0.67	0.41	0.81	0.91	0.88	0.88
Pavilion Day vue 2	0.78	0.78	0.73	0.77	0.61	0.78	0.69	0.71	0.71	0.70	0.61	0.50	0.84	0.83	0.62	1.10
Pavilion Day vue 3	0.84	0.74	0.84	0.94	0.63	0.68	0.71	0.50	0.64	0.68	0.76	0.54	1.02	0.56	0.98	0.50
Pavilion Night vue 1	0.76	0.64	0.60	0.74	0.56	0.69	0.71	0.56	0.61	0.61	0.41	0.62	0.53	0.75	0.59	0.99
Pavilion Night vue 2	0.34	0.45	0.57	0.56	0.41	0.44	0.39	0.44	0.47	0.41	0.40	0.30	0.50	0.43	0.68	0.54
Pavilion Night vue 3	0.97	0.69	0.56	0.93	0.95	0.73	0.68	0.70	0.83	0.67	0.77	0.66	0.98	0.69	0.79	0.99
Sanmiguel vue 1	0.02	0.49	0.28	0.43	0.49	0.11	0.27	0.20	0.20	0.22	0.16	0.12	0.47	0.11	0.11	0.22
Sanmiguel vue 2	0.26	0.30	0.23	0.13	0.34	0.13	0.28	0.09	0.06	0.17	0.19	0.03	0.22	0.27	0.15	0.22
Staircase 1	0.64	0.76	0.90	0.96	0.55	0.66	0.64	0.40	0.59	0.51	0.60	0.52	0.66	0.43	0.56	0.57
Staircase 2	0.56	0.52	0.43	0.54	0.66	0.68	0.87	0.83	0.37	0.51	0.55	0.63	0.30	0.68	0.31	0.79
Tt	0.57	0.50	0.55	0.49	0.60	0.49	0.41	0.49	0.60	0.48	0.45	0.47	0.52	0.62	0.71	0.55
Vw Van	0.39	0.58	0.43	0.44	0.30	0.38	0.41	0.38	0.36	0.36	0.30	0.34	0.48	0.46	0.35	0.34

TABLEAU A.3 – Le coefficient relatif des valeurs de seuils subjectifs obtenus par les participants sur chaque bloc d'une image.

A.4 Variété des scènes de la base d'images de synthèse

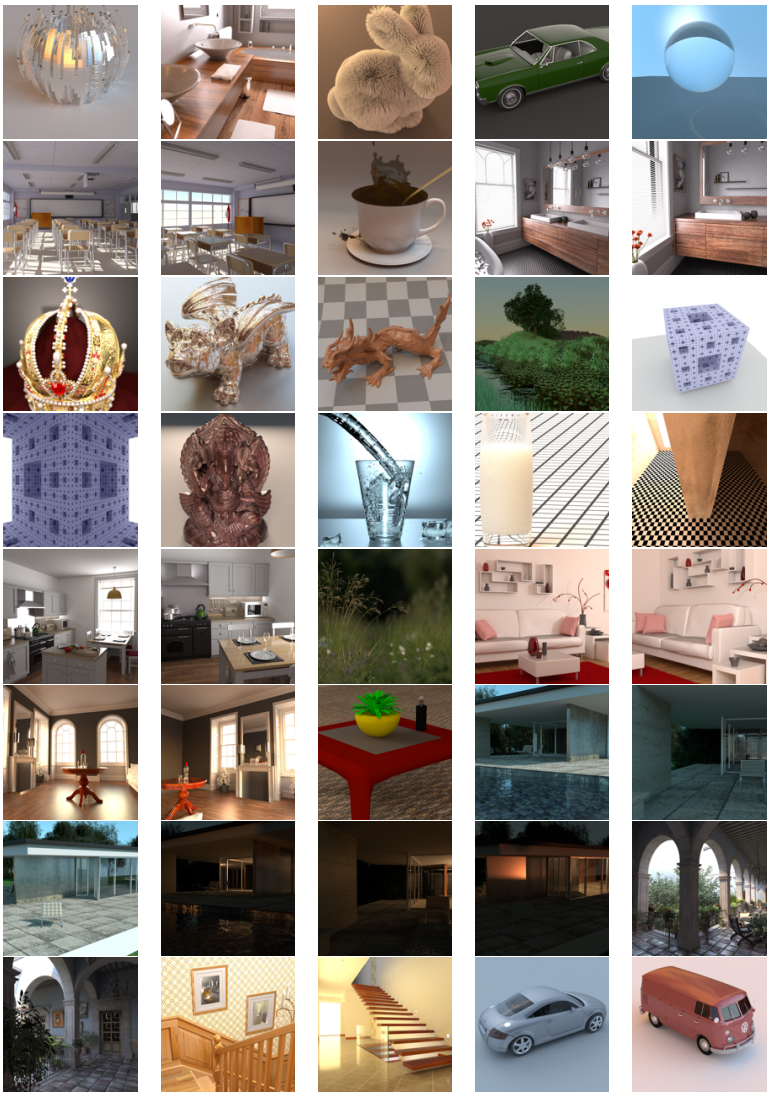


FIGURE A.1 – Aperçu des images de référence avec 10 000 échantillons par pixel de la base d'images.

Annexe B

Gestion des *fireflies*

B.1 Algorithme d'application de MoN lors du rendu d'images

Algorithme 3 : Algorithme de l'estimateur MoN pour le rendu

Result : Estimateur final $\hat{\mu}_{MoN}$

paramètre : n, nombre attendu d'échantillons;
paramètre : M, nombre de sous-estimateurs;
initialiser : $\hat{\theta}_{1,\dots,M}$, les M moyennes;
initialiser : $x_{1,\dots,M}$, initialiser M somme à 0;
initialiser : $k_{1,\dots,M}$, les M compteurs des ensembles à 0;

for $i = 1$ ***à*** n **do**
    $j = i$ % M;
    s = calculerÉchantillon();
    x_j += s;
    k_j += 1;
end

for $j = 1$ ***à*** M **do**
    $\hat{\theta}_j = x_j$ / k_j;
end
return $\hat{\mu}_{MoN}(\hat{\theta}_*)$;

B.2 Reformulation du coefficient de Gini

La formulation du coefficient de Gini G utilisé dans l'équation 6.6 a été facilement dérivée de la formulation fournie dans (DAMGAARD et WEINER 2000) et (DAMGAARD 2003), lorsque les données sont ordonnées par taille croissante des valeurs. G est donné comme $G = \frac{\sum_{i=1}^{n}(2i-n-1)x_i}{n^2\mu}$, avec $\mu = \frac{\sum_{i=1}^{n} x_i}{n}$. Alors :

$$\begin{aligned} G &= 2\frac{\sum_{i=1}^{n} i.x_i}{n^2\mu} - \frac{(n+1)\sum_{i=1}^{n} x_i}{n^2\mu} \\ &= 2\frac{\sum_{i=1}^{n} i.x_i}{n^2\frac{1}{n}\sum_{i=1}^{n} x_i} - (n+1)\frac{1}{n}\sum_{i=1}^{n} x_i \times \frac{1}{n\mu} \\ &= \frac{2\sum_{i=1}^{n} i.x_i}{n\sum_{i=1}^{n} x_i} - (n+1)\frac{\mu}{n\mu} \\ &= \frac{2\sum_{i=1}^{n} i.x_i}{n\sum_{i=1}^{n} x_i} - \frac{n+1}{n} \end{aligned} \tag{B.1}$$

Annexe C

Modèle de perception avec *SVD-Entropy*

C.1 Résultats modèles RNN avec *SVD-Entropy*

S	m	F	b_s	*bnorm*	*snorm*	n Step	Acc Train	Acc Test	AUC Train	AUC Test
8	100	H_{SVD}	128	0	1	40	84.58 %	82.74 %	84.44 %	82.55 %
5	100	H_{SVD}	128	0	1	80	83.78 %	82.87 %	83.32 %	82.47 %
6	100	H_{SVD}	64	0	1	40	84.18 %	82.61 %	84.01 %	82.45 %
7	100	H_{SVD}	64	0	1	40	84.62 %	82.79 %	84.24 %	82.44 %
7	100	H_{SVD}	128	0	1	40	84.65 %	82.75 %	84.36 %	82.42 %
7	100	H_{SVD}	64	0	1	80	83.67 %	82.36 %	83.70 %	82.38 %
5	100	H_{SVD}	64	0	1	80	83.61 %	82.17 %	83.85 %	82.27 %
9	100	H_{SVD}	64	0	1	40	83.46 %	81.99 %	83.84 %	82.21 %
10	100	H_{SVD}	128	0	1	40	84.52 %	82.58 %	84.20 %	82.16 %
10	100	H_{SVD}	64	0	1	20	84.12 %	82.16 %	84.28 %	82.15 %
6	100	H_{SVD}	128	0	1	40	84.05 %	82.25 %	84.07 %	82.13 %
9	100	H_{SVD}	128	0	1	40	84.39 %	82.82 %	83.61 %	82.10 %
9	100	H_{SVD}	64	0	1	20	84.93 %	82.48 %	84.50 %	82.06 %
9	100	H_{SVD}	128	0	1	20	85.37 %	82.56 %	84.90 %	82.04 %
10	100	H_{SVD}	128	0	1	20	84.73 %	82.53 %	84.18 %	82.02 %
6	100	H_{SVD}	64	0	1	80	83.04 %	81.68 %	83.59 %	82.01 %
4	100	H_{SVD}	128	0	1	80	83.17 %	81.81 %	83.55 %	82.00 %
5	100	H_{SVD}	128	0	1	40	83.88 %	82.23 %	83.59 %	81.95 %
8	100	H_{SVD}	128	0	1	80	83.61 %	81.89 %	83.73 %	81.88 %
5	100	H_{SVD}	64	0	1	40	83.85 %	82.02 %	83.66 %	81.84 %
7	100	H_{SVD}	128	0	1	80	82.31 %	81.15 %	83.01 %	81.63 %
10	100	H_{SVD}	64	0	1	80	83.39 %	81.97 %	83.00 %	81.63 %
6	100	H_{SVD}	128	0	1	80	83.60 %	82.41 %	82.64 %	81.62 %
10	100	H_{SVD}	64	0	1	40	84.13 %	82.45 %	83.16 %	81.60 %
8	100	H_{SVD}	64	0	1	20	83.09 %	81.40 %	83.50 %	81.58 %
8	100	H_{SVD}	128	0	1	20	84.29 %	82.04 %	83.71 %	81.54 %
7	100	H_{SVD}	64	0	1	20	83.65 %	81.78 %	83.40 %	81.53 %
3	100	H_{SVD}	128	0	1	80	83.10 %	82.21 %	82.32 %	81.47 %
9	100	H_{SVD}	128	0	1	80	84.10 %	81.85 %	83.62 %	81.37 %
4	100	H_{SVD}	64	0	1	40	83.32 %	81.92 %	82.44 %	81.20 %
9	100	H_{SVD}	64	0	1	80	83.13 %	81.62 %	82.70 %	81.19 %
7	100	H_{SVD}	128	0	1	20	82.84 %	80.84 %	83.35 %	81.16 %
6	100	H_{SVD}	128	0	1	20	83.14 %	81.13 %	83.13 %	81.06 %
6	100	H_{SVD}	64	0	1	20	83.31 %	81.37 %	83.01 %	81.04 %
3	100	H_{SVD}	64	0	1	80	81.89 %	80.68 %	82.33 %	80.99 %
4	100	H_{SVD}	64	0	1	80	81.64 %	80.38 %	82.55 %	80.97 %
10	100	H_{SVD}	128	0	1	80	82.36 %	80.79 %	82.48 %	80.84 %
8	100	H_{SVD}	64	0	1	80	83.24 %	81.76 %	82.19 %	80.84 %
4	100	H_{SVD}	128	0	1	40	83.05 %	81.59 %	81.95 %	80.71 %
8	100	H_{SVD}	64	0	1	40	82.06 %	79.85 %	83.13 %	80.61 %
10	25	H_{SVD}	64	0	1	20	82.09 %	80.84 %	81.64 %	80.46 %

TABLEAU C.1 – Les 40 meilleurs modèles obtenus avec les différents jeux de paramètres pour des attributs issus sur la *SVD-Entropy* en entrée.

C.2 Erreur critique globale des modèles RNN

Modèle	H_{SVD}	H_{SVD}	H_{SVD}	26 Attributs	26 Attributs	26 Attributs
Marge (en %)	2	6	10	2	6	10
Arcsphere	88.03	88.72	89.95	85.76	86.51	87.89
Bathroom	68.80	68.92	69.17	87.66	87.79	88.04
Bunny fur	76.67	77.67	79.67	71.34	72.34	74.34
Car2	89.42	90.22	91.50	87.97	88.81	90.44
Caustic	81.25	82.20	84.20	74.45	75.41	77.29
Classroom vue 1	82.47	82.95	83.83	79.62	80.06	80.94
Classroom vue 2	81.35	82.03	83.20	85.28	85.96	87.11
Coffee splash	69.75	70.72	72.72	59.26	60.26	62.26
Contemporary bathroom vue 1	79.97	80.38	81.12	86.86	87.24	87.99
Contemporary bathroom vue 2	90.65	91.30	92.30	89.09	89.64	90.53
Crown	89.40	90.47	92.15	75.16	76.16	78.15
Dragon	86.80	87.67	89.30	80.39	81.39	83.39
Dragon 250	87.35	88.22	89.78	88.90	89.90	91.80
Ecosys	95.17	95.95	97.33	95.35	96.25	97.94
Eponge fractal vue 1	87.50	88.03	89.20	92.42	92.99	93.86
Eponge fractal vue 2	97.80	98.30	99.05	91.06	92.00	93.81
Ganesha	93.80	94.60	96.12	95.65	96.46	97.75
Glass of water	83.67	84.42	85.88	74.26	75.08	76.70
Glass	83.62	84.28	85.17	84.25	84.59	85.09
Indirect	91.75	92.67	94.10	85.51	86.35	87.86
Kitchen vue 1	83.60	84.25	85.50	76.10	76.97	78.72
Kitchen vue 2	80.38	81.30	82.92	80.01	80.95	82.81
Landscape	92.28	93.03	94.22	80.94	81.91	83.55
Living room vue 1	84.53	85.20	86.45	81.38	82.38	84.38
Living room vue 2	90.47	91.42	93.22	86.88	87.74	88.97
Living room vue 3	85.38	85.83	86.53	81.55	82.17	83.34
Living room vue 4	76.15	76.80	78.03	71.16	71.72	72.80
Low table	86.00	86.97	88.70	83.78	84.78	86.78
Pavilion day vue 1	87.58	88.35	89.85	79.21	80.09	81.84
Pavilion day vue 2	86.78	87.78	89.53	84.95	85.95	87.84
Pavilion day vue 3	91.12	91.85	93.35	90.36	91.26	92.97
Pavilion night vue 1	85.47	86.03	87.03	80.34	80.84	81.72
Pavilion night vue 2	87.22	87.95	89.17	74.38	75.14	76.50
Pavilion night vue 3	89.45	90.25	91.67	87.90	88.59	89.90
Sanmiguel vue 1	93.42	94.30	95.70	89.66	90.59	92.29
Sanmiguel vue 2	88.95	89.90	91.42	83.31	84.12	85.70
Staircase 1	94.88	95.53	96.72	93.60	94.19	95.11
Staircase 2	91.83	92.55	93.80	83.51	84.41	86.16
Tt	77.80	78.80	80.55	83.54	84.29	85.79
Vw van	73.30	74.28	76.28	77.59	78.53	80.40
Bonnes prédictions (en %)	85.80	86.55	87.91	83.01	83.80	85.27

TABLEAU C.2 – Pourcentage de bonnes prédictions avec une marge d'erreur pour les meilleurs modèles RNN avec H_{SVD} et les 26 attributs en entrée.

C.3 Images les plus bruitées pour 6 points de vue utilisés pour l'apprentissage de modèle RNN

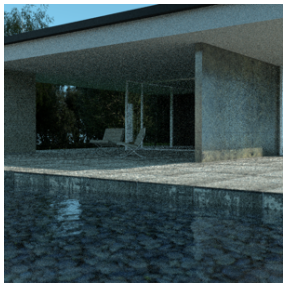
(a) Arcsphere

(b) Living room vue 3

(c) Ecosys

(d) Coffee splash

(e) Living room vue 1



(f) Pavilion day vue 1

FIGURE C.1 – Images les plus bruitées (20 échantillons par pixel) de 6 points de vue parmi les 35 disponibles de la base d'apprentissage.

C.4 Images de référence pour 6 points de vue utilisés pour l'apprentissage de modèle RNN

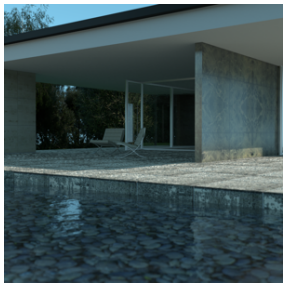
(a) Arcsphere

(b) Living room vue 3

(c) Ecosys

(d) Coffee splash

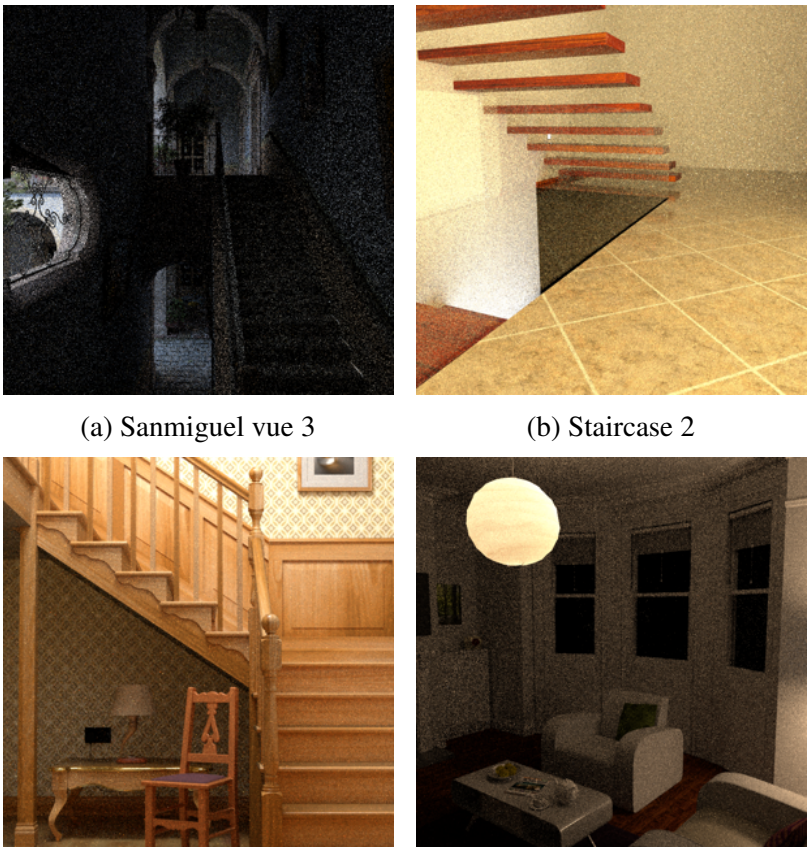
(e) Living room vue 1



(f) Pavilion day vue 1

FIGURE C.2 – Images de référence (à 10 000 échantillons par pixel) de 6 points de vue parmi les 35 disponibles de la base d'apprentissage.

C.5 Images les plus bruitées pour 4 points de vue non utilisés pour l'apprentissage



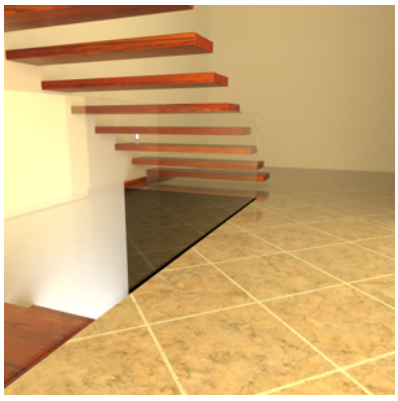
(a) Sanmiguel vue 3 (b) Staircase 2

(c) Staircase 1 (d) White-room night

FIGURE C.3 – Images bruitées (20 échantillons par pixel) de 4 points de vue non utilisés en apprentissage par le modèle RNN.

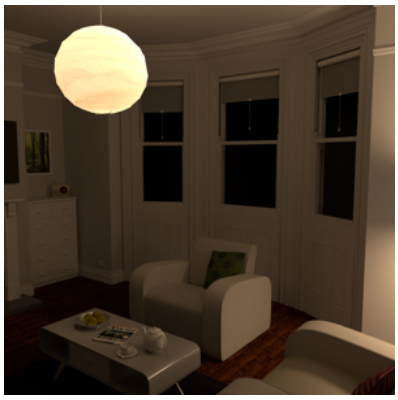
C.6 Images de référence pour 4 points de vue non utilisés pour l'apprentissage

(a) Sanmiguel vue 3



(b) Staircase 2

(c) Staircase 1



(d) White-room night

FIGURE C.4 – Images de référence (10 000 échantillons par pixel) de 4 points de vue non utilisés pour l'apprentissage.

MÉTHODES D'APPRENTISSAGE AUTOMATIQUE POUR LA PRISE EN COMPTE DU BRUIT DANS LES IMAGES DE SYNTHÈSE

Résumé

Les méthodes de simulation de l'éclairage, utilisées en synthèse d'images, permettent d'obtenir des vues dites photo-réalistes d'environnements virtuels 3D. Pour ce faire, elles utilisent des méthodes stochastiques, s'appuyant sur la théorie des grands nombres, qui explorent l'espace des chemins lumineux et se caractérisent par une convergence progressive de l'image vers la solution. Cette progressivité se traduit visuellement par la présence de bruit, qui se résorbe progressivement au fur et à mesure de l'avancée des calculs. Ce bruit doit être identifié et quantifié, afin de disposer de critères perceptifs permettant d'arrêter les algorithmes dans les différentes zones de l'image. Ceci est d'autant plus important que les temps de calcul d'une image se comptent en heures, voire en dizaines d'heures de calcul. Disposer de critères fiables pour arrêter les calculs en différents points d'une image permettrait donc de réaliser des gains de temps importants. Dans cette thèse, nous proposons d'utiliser des méthodes statistiques et d'apprentissage automatique pour la réduction et détection de ce bruit généré. Les contributions réalisées dans le cadre de cette thèse sont : (i) la constitution d'une base d'images de synthèse avec recueil de seuils subjectifs humains du bruit résiduel, (ii) l'étude et la gestion d'un bruit local hautement perceptible, (iii) la création de modèles d'apprentissage profond sur cette base d'images étiquetées et (iv) une phase de validation des images reconstruites obtenues (apprises ou non) à partir des modèles de perception à partir d'évaluations subjectives. Des travaux connexes à la thématique de la thèse, notamment relatifs à la gestion d'un bruit spécifique dans les images nommé « firefly », ont été proposés, tout comme l'application d'une méthode permettant de cibler les caractéristiques de bruit étudiées.

Mots clés : simulation d'éclairage, perception visuelle, apprentissage automatique, classification

Abstract

Lighting simulation methods, used in image synthesis, make it possible to obtain so-called photorealistic views of 3D virtual environments. To do this, they use stochastic methods, based on the theory of large numbers, which explore the space of light paths and are characterised by a progressive convergence of the image towards the solution. This progressiveness is visually expressed by the presence of noise, which is gradually reduced as the calculations progress. This noise must be identified and quantified in order to have perceptual criteria for stopping the algorithms in the different areas of the image. This is all the more important as the calculation time for an image can be counted in hours or even tens of hours. Having reliable criteria to stop the calculations at different points of an image would therefore allow significant time savings. In this thesis, we propose to use statistical and machine learning methods for the reduction and detection of this generated noise. The contributions made in the framework of this thesis are: (i) the constitution of a base of synthetic images with the collection of human subjective thresholds of the residual noise, (ii) the study and management of a highly perceptible local noise, (iii) the creation of deep learning models on this base of labelled images and (iv) a phase of validation of the reconstructed images obtained (learned or not) from the models of perception from subjective evaluations. Related work to the thesis research area, notably concerning the management of a specific noise in images called « firefly », has been proposed, as well as the application of a method allowing the targeting of the noise characteristics studied.

Keywords: lighting simulation, visual perception, machine learning, classification