Rapport CST : *Détection de bruit dans les images de synthèses stéréoscopiques par méthodes d'apprentissage automatiques*

Jérôme BUISINE

July 2019

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

1.1 Présentation de la thématique

Les moteurs de rendus d'illimunations globales sont utilisées de nos jours dans le but de générer des images photo-réalistes. La représentation d'une scène modélisée en 3D fourni en entrée au moteur de rendu permet l'obtention d'une image en sortie de celui-ci. Il définit la valeur de chaque pixel de l'image suivant les propriétés physique des objets qui constituent la scène. Plusieurs algorithmes de rendus ont été développés durant ces dernières années dans le but d'approximer l'image finale souhaitée. Chaque méthode possède ses propres avantages et inconvénients, parfois coûteuse en temps ou pas assez précise si le temps est en manquant. Ces méthodes de rendus se basent toutes sur une méthode d'approximation Monte-Carlo (suivant la loi des grands nombres) dans le but d'approximer l'équation de rendu [1] et chaque valeur de pixel de l'image par le biais d'une moyenne empirique de l'approximation des valeurs des pixels durant le rendu.

Dans l'objectif d'évaluer la qualité d'une image, des approches et métriques de calculs ont été développées et proposées dans la littérature. Ces métriques sont parfois spécifiques pour des distortions d'images telles que le flou gaussien, JPEG2000, compression JPEG bruitée, ringing. Généralement ciblé sur du bruit additif. Globalement, les approches publiées sont appliquées pour la mesure de qualité d'une image dites naturelle et très peu pour mesurer la qualité d'une image photo-réaliste. En effet, lors du rendu de l'image, un bruit peut être perceptible par l'humain. Il ne s'agit ici en aucun cas d'un bruit additif mais plutôt un bruit dit « impulsionnel », dû au fait de la complexité d'approximation des valeurs finales de pixels dans une zone de la scène provoquant une erreur numérique perceptible humainement.

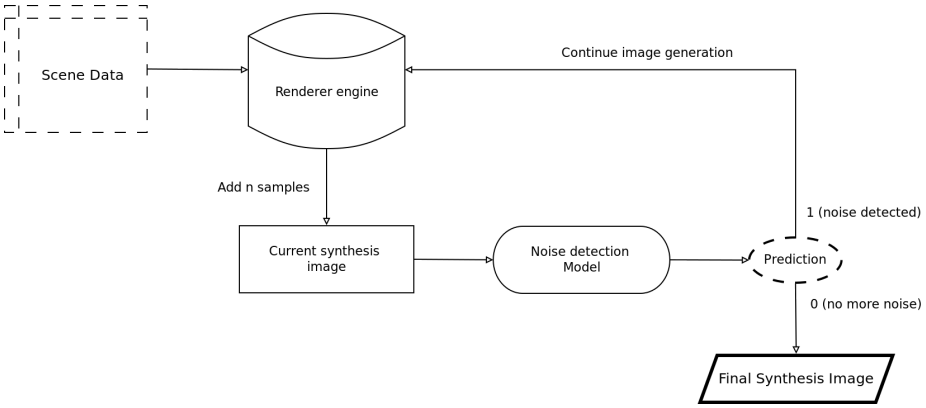


FIGURE 1.1 – Représentation de l'intéraction du modèle de détection souhaité de bruit lors du rendu d'une image de synthèse à partir des données de la scène 3D

1.2 Contexte de la thèse

Le sujet de thèse présenté ici s'intitule « Détection de bruit dans les images stéréoscopiques par méthodes d'apprentissage automatiques ». La thèse s'incruste dans un projet ANR donc l'objectif final serait de pouvoir concevoir un moteur rendu mixte mélant rendu temps réel (une image calculée en moins de 30 ms) et photo-réalisme. C'est à dire cibler la partie à améliorer par rapport au regard de l'utilisateur dans un support de rendu stéréoscopique. L'objectif de la thèse est double, à la fois détecter le bruit, le quantifier à partir de l'image uniquement et non les informations de la scène (si possible) puis proposer une méthode de filtrage/débruitage (post-traitement) permettant d'améliorer son aperçu. Tout en gardant pour objectif que cela puisse être réalisé en temps réel.

1.3 Problématique abordée

Comme évoqué précedemment, l'objectif de la thèse est double pouvoir à la fois détecter le bruit dans une image de synthèse issue d'un moteur de rendu et mais également proposer une méthode de réduction de ce bruit perçu (denoising method).

La Fig. 1.1 détaille davantage la manière dont le problème est abordé et surtout elle permet de cibler la partie sur laquelle la problématique de la thèse s'incruste. Nous avons donc le modèle de détection de bruit (modèle de classification binaire) qui permettrait de prédire si le moteur de rendu a encore la nécessité d'améliorer l'image ou non. On pourrait bien entendu ici, mettre en place une étape de post-traitement pour amélioration de l'image avant évaluation du bruit contenu dans celle-ci.

Chapitre 2

State of the art

2.1 Algorithmes de rendus

2.2 Image quality metrics

2.3 Application détection de bruits images de synthèse

2.4 SVD

Chapitre 3

Travaux réalisés

3.1 La base de données d'images

Cette partie sera consacrée à la présentation des différents travaux réalisés dans le cadre de la première année de thèse. La majorité des travaux ne donnant pas de résultats unanime, chaque approche sera donc détaillée pour mettre en avant ses avantages, ses inconvénients, les raisons pour lesquelles l'approche n'est pas forcément concluante pour enfin s'ouvrir à de nouveaux axes et méthodes d'applications.

Pour l'ensemble des travaux réalisés durant cette première année, l'image de taille 800×800 a été decoupée en 16 zones de 200×200 (voir Fig. 3.1). C'est ensuite une zone qui est traitée pour extraction des informations caractérisant le bruit présent dans le celui-ci. La première partie de la thèse porte donc sur cette étape du processus de classification.

Une base de données d'images a été développée, elle contient 9 scènes ainsi que les seuils perceptifs prelevés lors d'expérience pour chaque zone d'une scène. Chaque sujet de l'expérience devait partir de l'image la plus bruitée augmenter la qualité (nombre d'échantillons) de chaque zone pour se rapprocher au maximum de l'image de référence (image obtenue avec le plus grand nombre d'échantillons). Dans le cadre de ses premières expériences, uniquement les scènes issues du moteur de rendu Maxwell ont été selectionnées (voir tableau 3.1 et Fig. 3.2). La Fig. 3.3 propose un aperçu du bruit perceptible lors du du processus rendu d'une image de synthèse, ici sur la scène Appart1opt02.

La Fig. 3.4 présente la manière dont chaque zone des différentes images (nombre d'échantillons) sont labelisés.

Le processus de génération des données utilisées pour création des bases d'apprentissage, de validation et de test pour le modèle mathématique choisi est détaillé dans la Fig. 3.5. Ce processus vient de l'hypothèse que chaque zone est traitée indépendamment de la scène. Plutôt que d'utiliser 3 scènes en apprentissage et une en test, la sélection de z zones pour former la base d'apprentissage

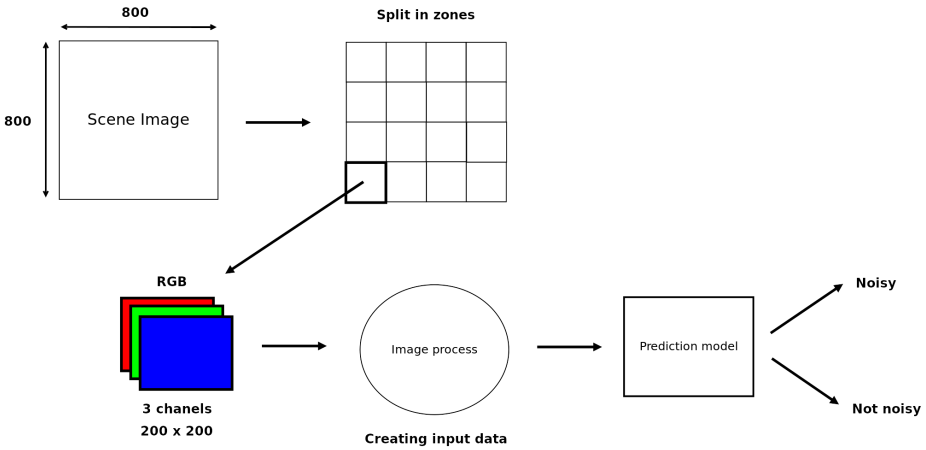


FIGURE 3.1 – Découpage de l'image d'une scène pour extraction des caractéristiques des zones de celle-ci

ID	Name	Indices	Step	Images
A	Appart02	$20 \rightarrow 900$	10	91
D	Cuisine01	$20 \rightarrow 1200$	10	121
G	SdbCentre	$20 \rightarrow 950$	10	96
H	SdbDroite	$20 \rightarrow 950$	10	96

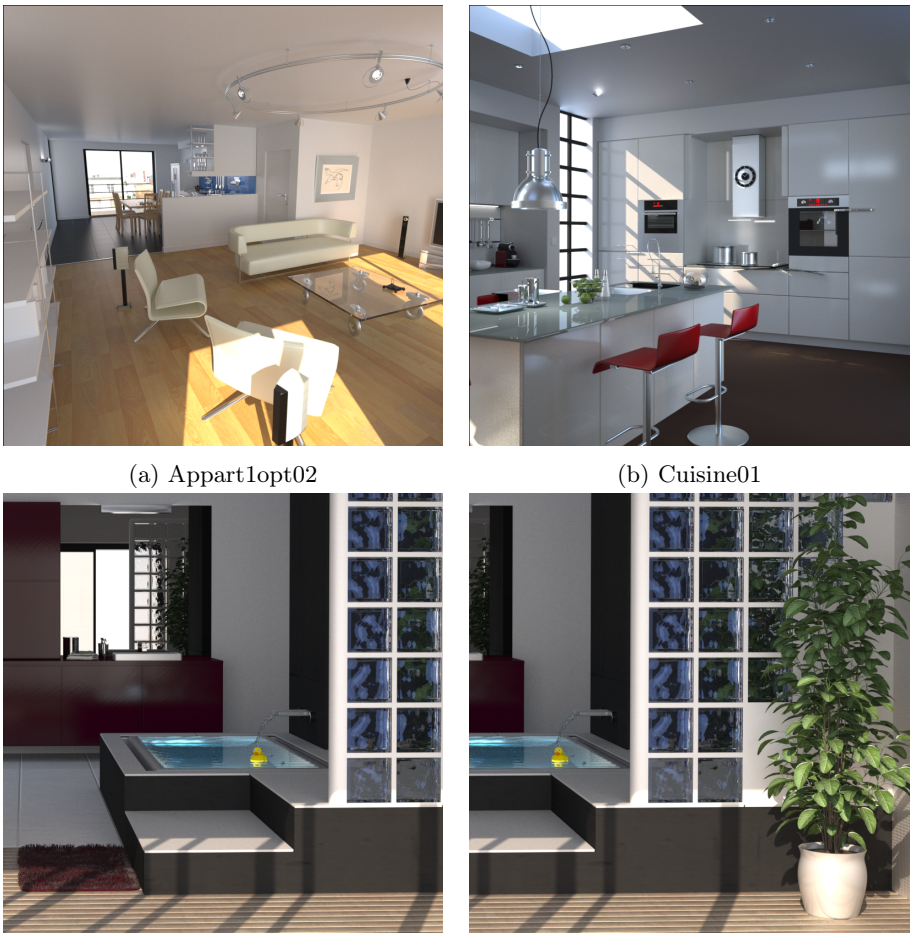
TABLE 3.1 – Information sur la base d'images de synthèse

sont réalisées aléatoirement avec $z \in [4, 6, 8, 10, 12]$. Les zones restantes seront alors utilisées pour les bases de validation et de test du modèle.

Afin de mesurer l'importance de la séparabilité des données à partir du seuil, 3 manières de construire les données finales ont été abordées :

- **all** : consitant à prendre l'ensemble des données.
- **center** : consistant à ne prendre que les images à échantillons proche de 150 du seuil estimé de la zone.
- **split** : les données non sélectionnées par la seconde (données fortement extérieur au seuil).

Enfin, la comparaison des performances de modèles se fera via le score AUC - ROC (Area Under The Curve Receiver Operating Characteristics) sur la base de test. Il permet de mesurer la performance d'un modèle pour un problème de classification, plus particulièrement à quel point il arrive a bien séparer les classes du problème.



(a) Appart1opt02 (b) Cuisine01

(c) SdbCentre (d) SdbDroite

FIGURE 3.2 – Aperçu des images de références de la base d'images

(a) Arpès 20 minutes (b) Après 300 minutes (c) Après 900 minutes

FIGURE 3.3 – Images de la scène Appart1opt02 à différents moments de l'étape de rendu. lors des premières minutes on peut remarquer un bruit perceptible. Il s'agit d'une approximation de pixels non précise obtenue lors du rendu.

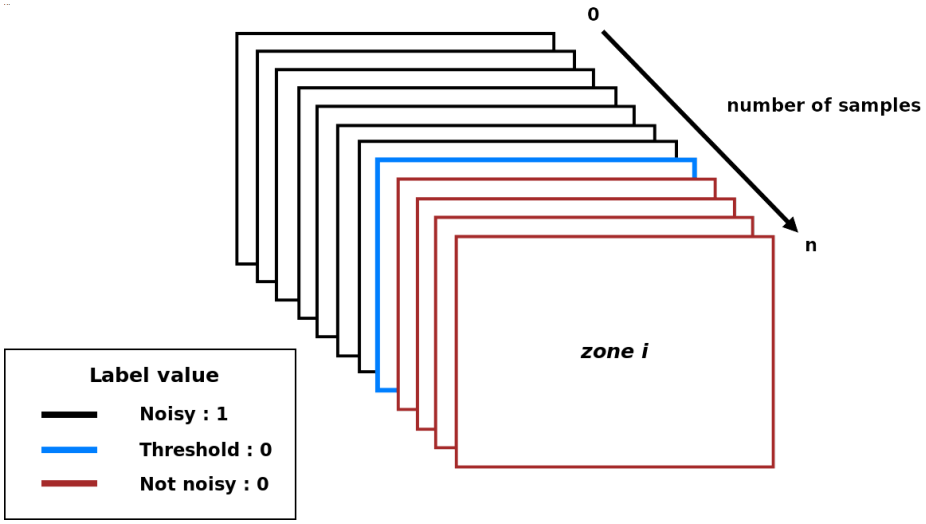


FIGURE 3.4 – Le seuil perceptif est identifié par la moyenne des scores obtenus sur les sujets de l'expérience. Les images situées avant ce seuil perceptif sont considérées comme bruitées, celles situés après, comme non bruitées.

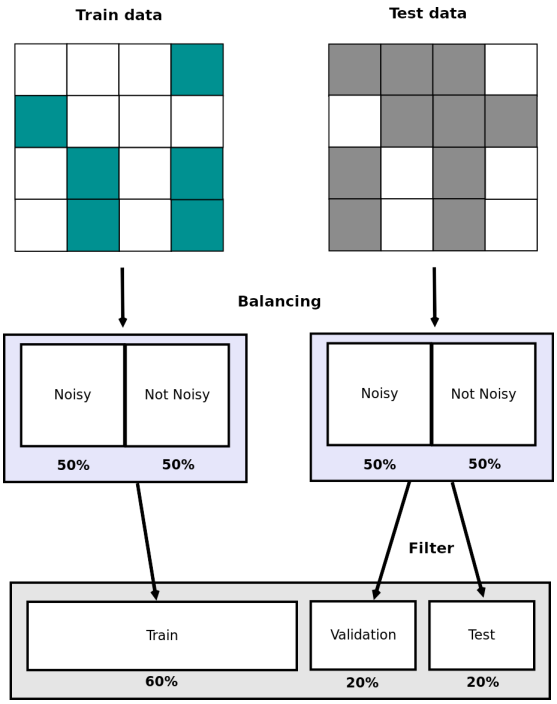


FIGURE 3.5 – Dans cet exemple, 6 zones ont été sélectionnées pour la base d'apprentissage. Un équilibrage des classes est réalise pour ne pas biaiser le modèle lors de l'apprentissage puis la répartition des différentes bases est effectuée.

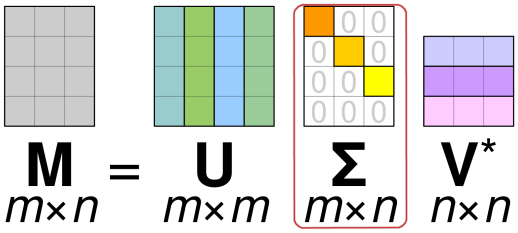


FIGURE 3.6 – Méthode de factorisation utilisée pour réduction de dimension de l'image : décomposition en valeurs singulières

3.2 L'approche décomposition SVD

3.2.1 Decription de la méthode

La décomposition en valeurs singulières (SVD) est une méthode de factorisation de matrice. L'utiliser dans le cadre de la réduction d'informations pour le traitement de l'image pourrait être intéressant. La Fig 3.6 propose un aperçu de la factorisation proposée par la méthode.

La méthode propose donc une factorisation de la matrice de la manière suivante :

$$M = U \times \Sigma \times V^*$$

C'est le vecteur de valeurs singulières (Σ) auquel nous nous sommes principalement intéressé. En effet, la méthode de décomposition appliquée à une image réduite au canal de luminance L de la transformation L*a*b (voir annexe A) ou la transformation MSCN **[DBLP:journals/tip/MittalMB12]** (voir annexe B) qui permet d'obtenir le vecteur de valeurs singulières Σ.

Pour une image de 200×200 d'une zone nous obtenons donc un vecteur de valeurs singulières de taille 200. C'est ce vecteur que nous allons traiter par la suite.

Pourquoi s'être intéressé à la décomposition SVD ? Nous avons pu observer que suivant le nombre d'échantillons utilisés pour générer une image d'un scène, le vecteur de valeurs singulières semblaient significativement bien distinguer le niveau de bruit jusqu'à l'image de référence (voir Fig. 3.7). La Fig. 3.8 propose également un aperçu des valeurs singulières mais ici sur une zone précise de la scène, la zone 3.

3.2.2 Sélection des attributs

Nous avons donc pour chaque image traitée, un vecteur de 200 valeurs à traiter. Dans cette section nous allons présenter les différentes manières (transformations de l'image) d'obtenir ce vecteur ainsi que les différentes approches de sélection des composantes de celui-ci.

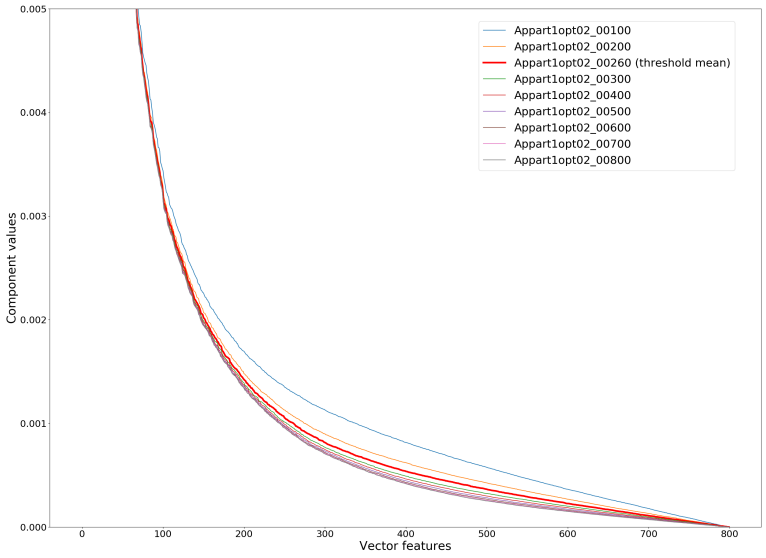


FIGURE 3.7 – Aperçu du vecteur de valeurs singulières des images de la scène Appart1opt02 à différents niveaux de bruits. Ici le vecteur possède 800 valeurs car la décomposition a été appliquée sur l'ensemble de l'image et non une zone spécifique. À noter que le vecteur est obtenu à partir du canal **L** de l'image et qu'il a été ici normalisé.

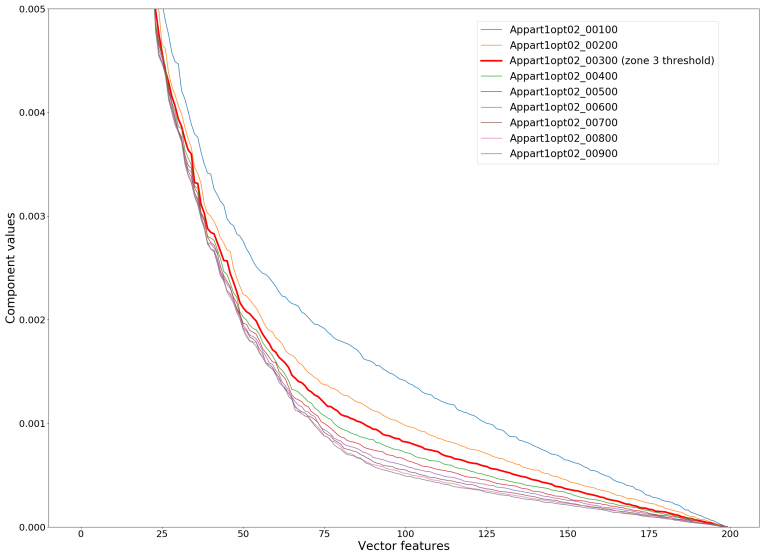


FIGURE 3.8 – Aperçu du vecteur de valeurs singulières des sous-images de la zone 3 de la scène Appart1opt02 à différents niveaux de bruits. Ici le vecteur possède 200 valeurs car la décomposition a été appliquée sur une zone spécifique. À noter que le vecteur est obtenu à partir du canal **L** de la sous-image et qu'il a été ici normalisé.

Liste des attributs extraits

La liste des transformations ci-dessous appliquées à l'image sont utilisées pour obtention du vecteur de valeurs singulières de 200 valeurs via la décomposition SVD.

- **lab** : utilisation de la luminance de l'image.
- **mscn** : utilisation de la matrice MSCN.
- **low_bits_2** : utilisation uniquement des 2 bits de poids faibles de la matrice de luminance (un octet de représentation).
- **low_bits_3** : utilisation uniquement des 3 bits de poids faibles de la matrice de luminance.
- **low_bits_4** : utilisation uniquement des 4 bits de poids faibles de la matrice de luminance.
- **low_bits_5** : utilisation uniquement des 5 bits de poids faibles de la matrice de luminance.
- **low_bits_6** : utilisation uniquement des 6 bits de poids faibles de la matrice de luminance.
- **low_bits_4_shifted_2** : utilisation uniquement des 4 bits de centraux de la matrice de luminance.
- **ica_diff** : utilisation d'une matrice calculée à partir de la différence entre la matrice de luminance et une image reconstruite à partir des 50 composantes principales à partir de la méthode « Independent Component Analysis » (ICA).
- **svd_trunc_diff** : utilisation d'une matrice calculée à partir de la différence entre la matrice de luminance et une image reconstruite à partir des 30 composantes principales à partir de la méthode de décomposition SVD.
- **ipca_diff** : utilisation d'une matrice calculée à partir de la différence entre la matrice de luminance et une image reconstruite à partir des 20 composantes principales à partir de la méthode de « Incremental Principal Components Analysis » (IPCA).
- **svd_reconstruct** : utilisation d'une image reconstruite à partir des 110 dernières composantes de la décomposition SVD (voir Fig. 3.9).

Approche dîtes naïve de sélection d'attributs

La première approche de sélection d'attributs consiste à choisir n composantes du vecteur de valeurs singulières avec $n \in [4, 8, 16, 26, 32, 40]$ à une position P avec $|P| = 5$ (voir Fig. 3.10). Cette approche peut paraître naïve mais elle permet déjà d'essayer un certains nombres de paramètres.

La Fig. 3.11 permet un aperçu récapitulatif de la manière dont l'image est traitée pour la sélection d'attributs d'entrée au modèle d'apprentissage.

Autres approches de sélection des attributs

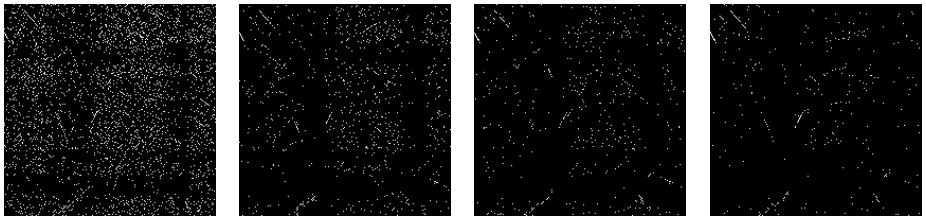
Étant donné l'approche naïve de sélection d'attributs et le fait que les valeurs des composantes d'une scène à une autre diffèrent, la capacité d'un modèle

(a) Scène Appart1opt02 zone 9 avec 100 samples

(b) Scène Appart1opt02 zone 9 avec 250 samples

(c) Scène Appart1opt02 zone 9 avec 400 samples

(d) Scène Appart1opt02 zone 9 avec 550 samples



(e) Scène Appart1opt02 zone 9 avec 100 samples reconstruite

(f) Scène Appart1opt02 zone 9 avec 250 samples reconstruite

(g) Scène Appart1opt02 zone 9 avec 400 samples reconstruite

(h) Scène Appart1opt02 zone 9 avec 550 samples reconstruite

FIGURE 3.9 – Aperçu de la reconstruction de la scène Appart1opt02 zone 9 à différents niveaux d'échantillons (samples) avec les 110 dernières composantes SVD

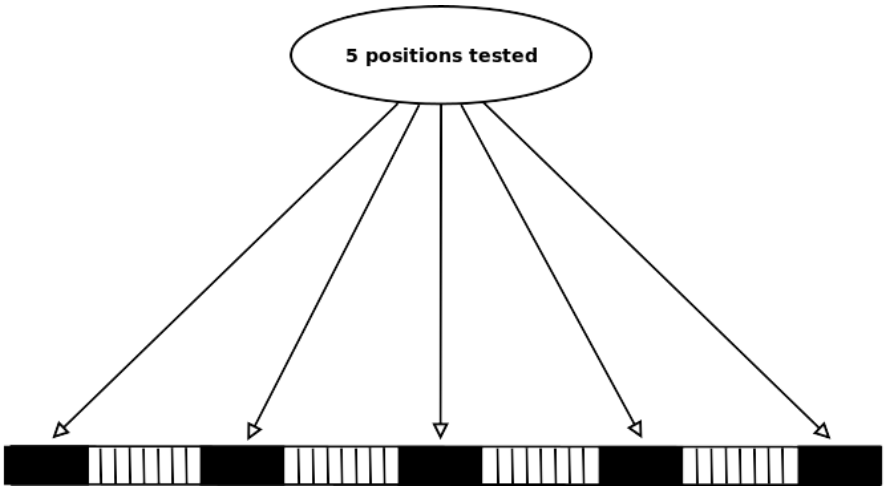


FIGURE 3.10 – Sélection d'attributs sur différentes zones du vecteur de valeurs singulières. La position est choisie comme telle, au début, au quart, au centre ; au trois quarts et en fin de vecteur. Si l'on décide de prendre $n = 20$ composantes à partir de la position P centrale, l'interval sélectionné sera $[90, 120[$

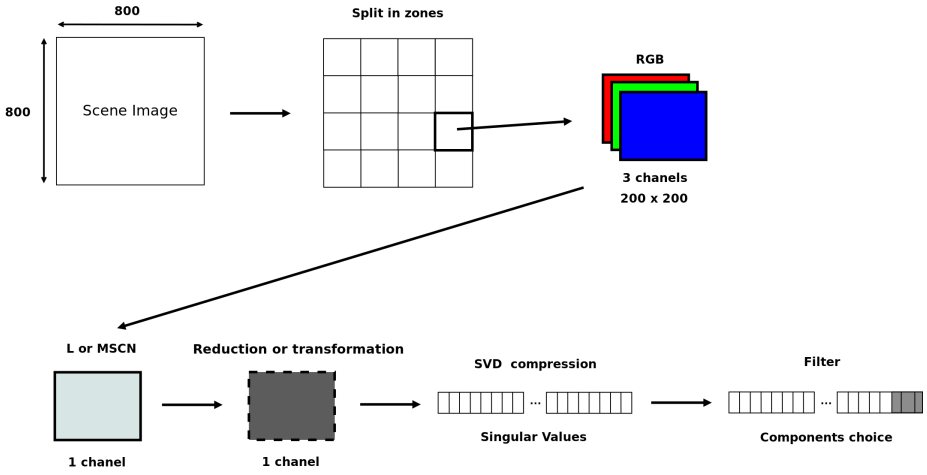


FIGURE 3.11 – Schéma récapitulatif de traitement de l'image dans l'objectif de sélectionner les meilleurs attributs pour entrée au modèle

de classifier correctement une image peut être impactée. Les listes ci-dessous énumèrent les approches exploitées dans le cadre de la thèse.

Sur l'ensemble des données :

— **highest_corr_sv :** récupération des composantes les plus corrélées entre-elles (score de corrélation) sur l'ensemble des données disponibles
— **lowest_corr_sv :** récupération des composantes les moins corrélées entre-elles (score de corrélation) sur l'ensemble des données disponibles

Sur l'image seule :

— **sv_std_filters :** génération de 4 images supplémentaires à l'image de niveau de luminance en entrée, à partir des filtes « wiener » (fenêtres kernel [3, 3] et [5, 5]) et « median » (fenêtres kernel [3, 3] et [5, 5]).
— **wave_sv_std_filters :** génération de 5 images supplémentaires à l'image de niveau de luminance en entrée, à partir des filtes « wiener » (fenêtres kernel [3, 3] et [5, 5]), « median » (fenêtres kernel [3, 3] et [5, 5]) et « wavelet ».
— **sv_std_filters_full :** génération de 13 images supplémentaires à l'image de niveau de luminance en entrée, à partir des filtes « wiener » (fenêtres kernel [3, 3] et [5, 5]), « median » (fenêtres kernel [3, 3] et [5, 5]), « mean » (2 kernels), « gaussian blur » (6 kernels) et « wavelet ».

La décomposition est ensuite appliquée à chacune des images disponibles dans la liste des images. Ce qui donne une matrice de taille $l \times 200 \times 200$ avec l le nombre d'images dans la liste. Pour chacune des composantes, les valeurs qui lui sont trouvées (l valeurs) sont normalisées suivant cette composante et la valeur de l'écart-type est calculée. Les n plus grandes (highest) ou petites (lowest) valeurs d'écart-type permettent de cibler les n composantes à sélectionner (sélection dynamique pour l'image).

Sur l'image seule :

— **sv_entropy_std_filters :** les même filtres que « sv_std_filters_full » soit, 13 images générées supplémentaires sont utilisées. La contribution à l'entropie de chaque composante du vecteur de valeur singulières est calculée. C'est ensuite la valeur de l'écart-type de ces contributions d'entropie qui sont utilisées comme valeurs de sélection des composantes.

Pourquoi s'intéresser à un traitement d'image seul pour sélection des attributs ? Étant donné les résultats sur une approche de sélection d'interval, il semblerait que la sélection des attributs soient fortement liée à la scène elle-même (structure de la scène). Une approche proposant dynamiquement de choisir les composantes pour une scène (ici l'image directement) serait plus judicieuse et permettrait d'avoir une méthode générique qu'importe les images de nouvelles scènes à traiter. Une étude plus approfondie permettrait de mettre en avant pour chaque scène les composantes les plus utilisées (répartition/distribution de sélection des composantes).

3.2.3 Paramètres et résultats

Avant de mettre en avant les résultats issus de la décomposition SVD, il est important de rappeler l'architecture des modèles utilisés, les manières de normaliser les données ainsi que le récapitulatif de l'ensemble des paramètres.

Modèles

L'annexe C présente ces différents modèles. En voici la liste :
— **M1 :** Support Vector Machine
— **M2 :** Ensemble model (3 sub-models)
— **M3 :** Ensemble model v2 (5 sub-models)

Ces ensemble de modèles sont des « voting classifier » avec le principe de le vote équitable et réglementé sur les élections « douces ».

Normalisation des données

Au niveau du traitement des données, nous avons opter sur trois manières de fournir les données en entrée aux modèles :
— **svd :** sans normalisation
— **svdn :** le sous-vector est normalisé avec ces propres valeurs
— **svdne :** le sous-vector est normalisé en utilisant les valeurs minimale et maximale des sous-vectors ou composantes de l'ensemble du dataset.

Résultats

Les 5 meilleurs modèles et leurs résultats issus des attributs extraits avec une sélection dites naïves sont présentés dans le tableau 3.2. Les simulations des scènes « Appart1opt02 » et « SdbDroite » obtenues du meilleur modèle sont disponibles (voir Fig. 3.12 et 3.13).

Model	feature	size	interval	zones	ROC Train	ROC Val	ROC Test
M3	lab (svd)	40	[80, 120[	12	0.9418	0.9023	0.9219
M2	lab (svd)	32	[84, 116[	4	0.9158	0.8724	0.9153
M2	lab (svd)	40	[80, 120[	12	0.9629	0.9049	0.9145
M2	lab (svdne)	26	[87, 113[	6	0.9337	0.8763	0.9089
M3	low_bits_2 (svd)	40	[0, 40[	12	0.9567	0.8417	0.9081

TABLE 3.2 – 5 meilleurs modèles avec sélection « naïve » sur le score ROC AUC

Les 5 meilleurs modèles et leurs résultats issus des attributs extraits avec une sélection dites « automatisés » sont présentés dans le tableau 3.3. Les simulations des scènes « Appart1opt02 » et « SdbDroite » obtenues du meilleur modèle sont disponibles (voir Fig. 3.14 et 3.15).

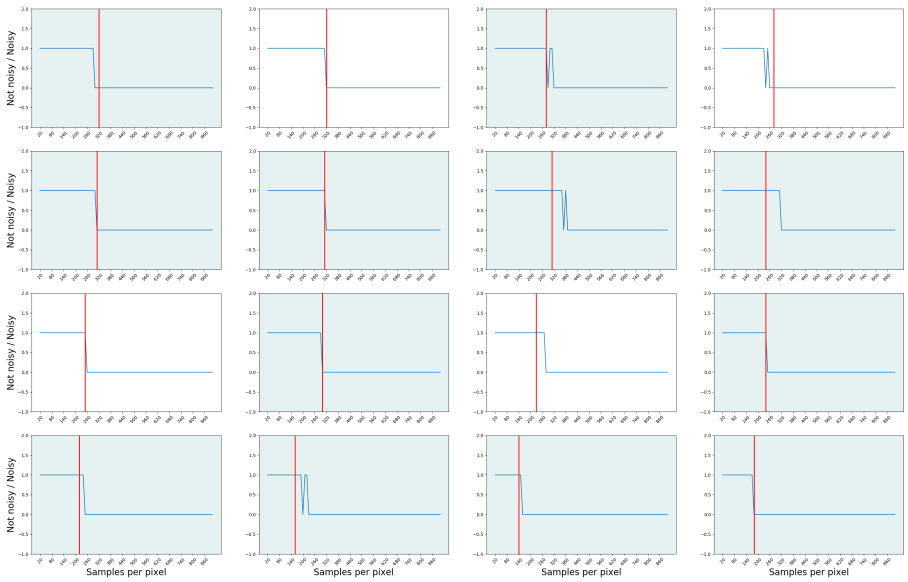


FIGURE 3.12 – Simulation des prédictions sur chaque zone de la scène Appart1opt02 (A) durant le rendu. Ici les zones apprises (zones avec un fond bleu) sont correctement apprises tout comme les zones non apprises où le seuil est pratiquement identiques.

FIGURE 3.13 – Simulation des prédictions sur chaque zone de la scène SbdDroite (H) durant le rendu. Ici les zones apprises (zones avec un fond bleu) sont presque correctement apprises (voir zones 1, 5, 9 qui possèdent pas mal d'oscillations avant détection de non bruit). Par contre, 2 zones sur 4 non apprises possèdent aucun arrêt de la part du modèle (zones 4 et 11), tout comme la zone 7 qui n'est pas correctement prédite.

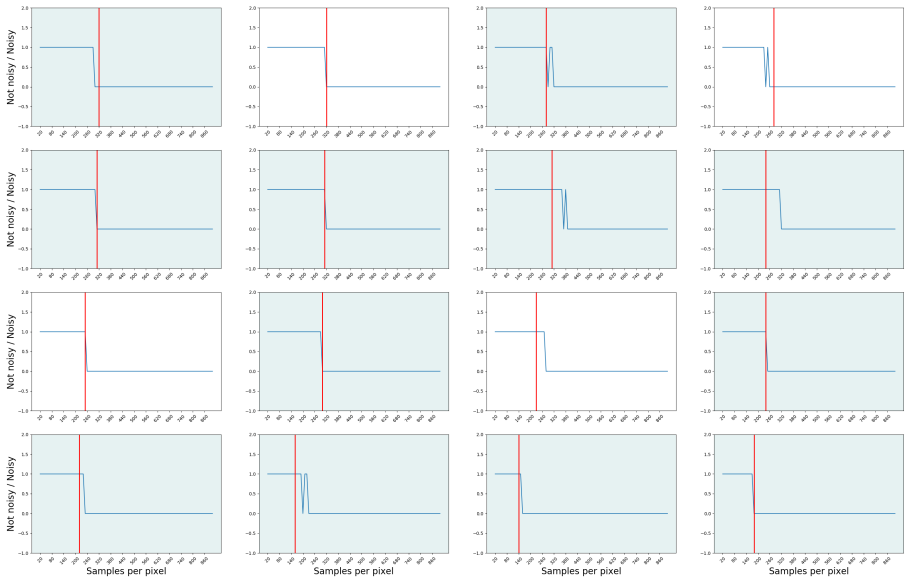


FIGURE 3.14 – Simulation des prédictions sur chaque zone de la scène Appart1opt02 (A) durant le rendu.

FIGURE 3.15 – Simulation des prédictions sur chaque zone de la scène SbdDroite (H) durant le rendu.

Model	feature	size	correlation	kind	zones	ROC Train	ROC Val	ROC Test
M3	filters (svdne)	80	lowest	sv entropy std	12	0.9970	0.8348	0.8864
M1	lab (svd)	10	highest	components	8	0.8510	0.8563	0.8740
M1	lab (svdne)	20	highest	labels	12	0.8100	0.8729	0.8729
M2	lab (svdne)	40	highest	components	12	0.9681	0.8670	0.8709
M1	lab (svdne)	25	highest	components	12	0.9311	0.8596	0.8595
M2	filters (svdne)	80	highest	sv std	12	1	0.7644	0.8519

TABLE 3.3 – 5 meilleurs modèles avec sélection « automatisé » sur le score ROC AUC

Conclusion

Par rapport à l'ensemble de ces résultats obtenus, il est difficile de dire si le modèle peut être assez généraliste sur l'ensemble des scènes ou sur de nouvelles scènes. En effet, il semblerait que l'approche d'apprentissage présente certaines limites et ne permet pas assez bien de généraliser (zones non apprises). Peut-être que la structure de la scène est trop renseignée dans les attributs fournis en entrée au modèle ce qui l'empêche de mesurer le bruit présent dans la scène. Cette raison pourrait expliquer le sur-apprentissage de certaines scènes (comme Appart1opt02). L'approche naïve de sélection des composants amène aussi à cette généralisation non parfaite puisque les composantes du vecteur de valeurs singulières ne porteront certainement pas la même information d'une scène à une autre.

A noter que des approches deep learning (sans convolutions) ont été esssayées pour vérifier si la méthode SVM n'était pas assez robuste à l'apprentissage. Malheureusement les résultats n'ont pas été meilleur que ceux sur les architectures de modèles proposées.

Ressources : Projet github comprenant l'ensemble des développements réalisés dans cette section.

3.3 Autres attributs

Après les différents résultats d'autres approches ont été explorées dans le but d'améliorer la prédiction du modèle. La remarque quand aux problème rencontrés précédemment concernés le fait que le modèle ne pouvait pas comprendre correctement un interval de données, d'où la transition sur la sélection automatisées de composantes du vecteur SV. Ici, nous nous intéresserons à des approches orientées statistiques.

3.3.1 Description des extractions

L'idée principale de ces nouveaux calculs d'attributs viennent du fait que l'on va s'interesser à des valeurs statistiques issues des valeurs de pixels des blocs de la zone (redécoupage de l'image). L'hypothèse était ici que le fait de travailler directement sur des parties plus petites de l'image permettrait d'extraire des propriétés plus intéressantes. La compression SVD est aussi étudiée dans le cadre de ces approches.

Voici la liste des extractions statistiques d'une zone proposées dans le cadre de la thèse :

- **sub_blocks_stats :** la zone est découpée en 4 blocs de taille identiques ($100x100$). Pour chaque bloc on extrait des statistiques telles que la moyenne, la médiane, le premier quartil, le troisième quartil et la variance du vecteur SV extrait de ce bloc. L'aire sous la courbe (l'intégrale) SV est également calculée (suivant la méthode trapézoidale). L'ensemble des statistiques de chaque bloc sont concaténées et utilisées comme entrée au modèle.
- **sub_blocks_area :** la zone est découpée en 16 blocs de taille identiques ($50x50$). Pour chaque bloc l'aire sous la courbe (l'intégrale) SV est calculée (suivant la méthode trapézoidale). L'ensemble des aires de chaque bloc sont concaténées et utilisées comme entrée au modèle.
- **sub_blocks_stats_reduced :** la zone est découpée en 4 blocs de taille identiques ($100x100$). Pour chaque bloc on extrait des statistiques telles que la moyenne, la médiane, le premier quartil, le troisième quartil et la variance du vecteur SV extrait de ce bloc. L'ensemble des statistiques de chaque bloc sont concaténées et utilisées comme entrée au modèle.
- **sub_blocks_area_normed :** Les valeurs utilisées en entrée sont identiques à la l'extraction « sub_blocks_area : » mais sont ici normalisées.
- **mscn_var_4 :** la zone est découpée en 4 blocs. Pour chaque bloc on réalisé la transformation MSCN (voir annexe **??**). Pour chaque nouvelle matrice on calcule la variance. Le vecteur d'information statistiques est ensuite utilisé comme entrée au modèle.
- **mscn_var_16 :** Même processus que pour l'extraction proposée dans « mscn_var_4 » mais avec ici 16 blocs.
- **mscn_var_64 :** Même processus que pour l'extraction proposée dans « mscn_var_4 » mais avec ici 64 blocs.

— **mscn_var_16_max :** Même processus que pour l'extraction proposée « mscn_var_16 » mais les valeurs du vecteur d'entrée sont ordonnées de la valeur de variance la plus grande à la plus petite.
— **mscn_var_64_max :** Même processus que pour l'extraction proposée « mscn_var_64 » mais les valeurs du vecteur d'entrée sont ordonnées de la valeur de variance la plus grande à la plus petite.

3.3.2 Paramètres et résultats

Paramètres

Tout comme les calculs effectués sur les composantes du vecteur de valeurs singulières, les paramètres concernant les modèles et la normalisation des données sont identiques. Pour rappel, l'annexe C présente ces différents modèles. Enfin, comme précédemment, la normalisation des données d'entrée est effectuée de la manière suivante :
— **svd :** sans normalisation
— **svdn :** le sous-vector est normalisé avec ces propres valeurs
— **svdne :** le sous-vector est normalisé en utilisant les valeurs minimale et maximale des sous-vectors ou composantes de l'ensemble du dataset.

Résultats

Le tableau 3.4 indique les 5 meilleurs modèles obtenus dans le cadre de cette étude. Les simulations du meilleur modèle obtenu sont également disponibles.

Model	feature	size	zones	ROC Train	ROC Val	ROC Test
M3	sub_block_stats_reduced (svd)	24	12	1	0.8288	0.8565
M2	sub_block_stats_reduced (svd)	24	10	1	0.8714	0.8539
M2	sub_block_stats_reduced (svd)	24	12	1	0.8030	0.8342
M2	sub_block_stats_reduced (svd)	24	8	1	0.8119	0.8294
M1	sub_block_stats (svd)	24	8	1	0.8595	0.8376

TABLE 3.4 – 5 meilleurs modèles avec approche statistiques sur le score ROC AUC

3.3.3 Conclusion

- Overfitting

Ressources : Projet github comprenant l'ensemble des développements réalisés dans cette section.

3.4 Etude des échantillons de rendu

Dans le cadre d'une étude plus minutieuse, nous nous sommes intéressés à l'étude de l'évolution des valeurs de pixels et des approximations durant le rendu. A chaque échantillon, une approximation du pixel est réalisée, c'est enfin la moyenne empirique de ces esttimations qui permet de représenter la valeur finale d'un pixel. Une base de données d'images a été créée dans le but de pouvoir étudier sur 1000 échantillons chacun des valeurs.

Cette base comprend pour chaque pixel les 1000 estimations (échantillons) réalisés lors du rendu sur son niveau de gris (canal de luminance).

3.4.1 Description de l'objectif

L'objectif au travers de cette base d'images disponibles et de pouvoir chercher un modèle pouvant prédire la moyenne empirique (de l'image de référence) en ayant pris connaissance des n premiers échantillons. Cela revient à dire que le problème est d'estimer par rapport à la distribution actuelle (des n premiers échantillons), la distribution finale des estimations des pixels afin d'en obtenir la moyenne empirique comme présente sur l'image de référence.

Deux approches on été faites, l'une directement sur les valeurs des pixels à chaque échantillon jusqu'à n échantillons, l'autre sur l'évolution de la variance des pixels connus à n échantillons.

3.4.2 Paramètres et résultats

Un ensemble de modèles statistiques ou non ont été essayés pour répondre à cet objectif :

— M1 : Stochastic Gradient Descent
— M2 : Support Vector Regression
— M3 : Ridge regression
— M4 : Deep Learning approach (NN)

C'est le coefficient de détermination qui est utilisé comme validation du modèle (score objectif). La comparaison des modèles se fait toutefois sur l'erreur quadratique moyenne entre l'image actuelle obtenue (image reconstruite via le modèle) et l'image de référence (voir tableau 3.5).

Model	samples	column	row	data size	coefficient	MSE 10 samples	MSE 1000
M4	30	2	4	327680	0.9269	2.0186	6.8187
M4	30	2	1	1310720	0.9637	2.8522	8.8948
M2	25	2	1	1310720	0.8606	3.3147	9.7976
M2	30	2	5	263680	0.9425	3.5712	8.8095

TABLE 3.5 – 4 meilleurs modèles avec approche statistiques sur le score ROC AUC

3.4.3 Conclusion

Ressources : Projet github comprenant l'ensemble des développements réalisés dans cette section.

3.5 Les approches futures

3.5.1 Filtres et statistiques

Comme proposé dans [**DBLP:journals/ijon/ConstantinBCH15**], 13 filtres sont appliqués à l'image de synthèse (zone de l'image plus précisemment). De ces 13 nouvelles images déformées, 2 valeurs statistiques en sont extraites, l'écart-type et la moyenne.

Relativement à ces travaux, l'idée est de proposer via un algorithme génétique, d'évaluer les performances des filtres les plus cohérents en termes de perception du bruit. L'objectif est donc de développer un algorithme de sélection des filtres par rapport à la performance finale du modèle. La performance sera ici évaluée sur une base de test. Toutes les combinaisons possibles d'utilisation de filtres (ou non utilisation) ne seront pas testées mais l'optimum sera recherché via un algorithme de recherche opérationnel.

L'ensemble des développements est disponible pour sur github. Ce projet utilise un framework d'optimisation développé dans le cadre la thèse.

3.5.2 Approche deep learning

Comme utilisé dans la section 3.2.2, les différentes méthodes de compression d'image telles que SVD, PCA, IPCA peuvent être utilisées pour reconstruire l'image avec ses composantes principales. L'hypothèse est ici que les composantes principales ne vont pas forcément garder comme information le bruit généré lors du rendu de celle-ci. Utiliser ses méthodes de compression puis de reconstruction avec suppression des informations principales de l'image pourrait permettre de garder uniquement les informations dites de bruit.

Dans ce but, l'idée serait donc d'appliquer 1 à n méthodes de compression à l'image de synthèse puis d'utiliser ces nouvelles images en entrée à un modèle deep learning avec couches de convolution. Le tranfer learning sera également essayé avec notamment le réseau VGG 2019 car la convergence des poids pour le réseau nécessite énormément de données ce qui n'est pas encore notre cas pour notre problématique.

L'ensemble des développements pour cette approche deep learning a convolution est disponible sur github. Pour le moment, le lancement des calculs nécessitent une haute disponibilité (voir utilisation de calculco ou grid5000 en best effort).

Des approches deep learning telles que l'« autoencoder » ou le « Generative Advsersarial Network » (GAN) sont également à explorer.

3.5.3 Filtres par convolution

Dans la continuité des recherches, l'objectif est ici de mettre en place des filtres qui pourront être appliqué sous forme de convolution sur l'image de synthèse afin d'en extraire des informations précises sur le bruit. Plusieurs idées ont

déjà été développées dans le but d'obtenir à la sortie un score global de l'image sur son niveau de bruits.

En voici une première liste dont les liens amènent sur leurs développement et études :

— diff filter

— plane filter

L'objectif serait ici de trouver un ou plusieurs filtres donnant un score sur la qualité de l'image (présence de bruit). Si la combinaison de plusieurs est nécessaire, alors on pourra opter pour des entrées statistiques comme proposé dans les travaux [**DBLP:journals/ijon/ConstantinBCH15**].

Chapitre 4

Enseignements

— **L1 Mathématiques :** Fondements de l'algorithmique (19.38 Eq TD)
— **L2 Informatique :** Algorithmique avancée en C++ (12 Eq TD)
— **DUT 1re année Informatique :** Bases de la POO (28 Eq TD)

Chapitre 5

Formation

Chapitre 6

Conclusion

Chapitre 7

References

Bibliographie

[1] James T. Kajiya. « The rendering equation ». In : *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1986, Dallas, Texas, USA, August 18-22, 1986*. Sous la dir. de David C. Evans et Russell J. Athay. ACM, 1986, p. 143-150. isbn : 0-89791-196-2. doi : 10.1145/15922.15902. url : http://doi.acm.org/10.1145/15922.15902.

Appendices

Annexe A

Transformation L*a*b

L*a*b est otbenu en utilisant la transformation XYZ de la matrice d'image (espace CIZ XYZ). Les équations (A.1), (A.2), (A.3) détaillent la manière dont chaque canal est calculé.

$$L = 116 f_y - 16 \tag{A.1}$$

$$a = 500(f_x - f_y) \tag{A.2}$$

$$b = 200(f_y - f_z) \tag{A.3}$$

où

$$f_x = \begin{cases} \sqrt[3]{x_r} & \text{if } x_r > \epsilon \\ \frac{\kappa x_r + 16}{116} & \text{otherwise} \end{cases} \tag{A.4}$$

$$f_y = \begin{cases} \sqrt[3]{y_r} & \text{if } y_r > \epsilon \\ \frac{\kappa y_r + 16}{116} & \text{otherwise} \end{cases} \tag{A.5}$$

$$f_z = \begin{cases} \sqrt[3]{z_r} & \text{if } z_r > \epsilon \\ \frac{\kappa z_r + 16}{116} & \text{otherwise} \end{cases} \tag{A.6}$$

$$x_r = \frac{X}{X_r}, \quad y_r = \frac{Y}{Y_r}, \quad z_r = \frac{Z}{Z_r}$$

$$\epsilon = \begin{cases} 0.008856 & \text{Actual CIE standard} \\ 216/24389 & \text{Intent of the CIE standard} \end{cases} \tag{A.7}$$

$$\kappa = \begin{cases} 903.3 & \text{Actual CIE standard} \\ 24389/27 & \text{Intent of the CIE standard} \end{cases} \tag{A.8}$$

Annexe B

Mean Subtracted Contrast Normalized (MSCN)

Pour construire la matrice MSCN, il faut premièrement convertir l'image RGB en une image de niveau de gris. La matrice MSCN va permettre d'extraire les informations naturelles de structures de la scène (Natural Scene Structure : NSS) de cette image en niveau de gris. Une opération est appliquée An operation is applied à la luminance de l'image $I(i,j)$ pour obtenir :

$$\hat{I}(i,j) = \frac{I(i,j) - \mu(i,j)}{\sigma(i,j) + C} \tag{B.1}$$

où $i \in 1, 2...M, j \in 1, 2...N$ sont les indices spatiaux, M, N sont la hauteur et la largeur de l'image respectivement, C est une constante, de valeur 1 pour prévenir des instabilité et où

$$\mu(i,j) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} w_{k,l} I_{k,l}(i,j) \tag{B.2}$$

$$\sigma(i,j) = \sqrt{\sum_{k=-K}^{K} \sum_{l=-L}^{L} w_{k,l} (I_{k,l}(i,j) - \mu(i,j))^2} \tag{B.3}$$

Annexe C

Modèles utilisés

C.1 M1 : SVM classique

Nous utilisons le fameux modèle SVM (Support Vector Machine) pour prédire le bruit et non les images bruyantes. SVM est principalement utilisé et fonctionne très bien pour les problèmes NR-IQA (No Reference Image Quality Assessment) avec ces paramètres :

- Kernel : *rbf*
- $C \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
- $\gamma \in \{0.001, 0.01, 0.1, 1, 5, 10, 100\}$
- $cv = 10$ (paramètre de validation croisée pour les données d'entraînement, ce qui augmente la précision et évite le surapprentissage)

Ensuite, tous ces paramètres sont testés à l'aide du principe de recherche par grille et le meilleur modèle est obtenu.

C.2 M2 : Voting classifier avec 3 estimateurs

Nous proposons également un modèle Voting classifier (ensemble de modèles) qui est composé de 3 sous modèles :

- ***SVM*** avec un kernel RBF comme présenté précédemment.
- ***Random Forest*** avec 100 estimateurs
- ***Logistic Regression*** avec le noyau *liblinear* et le paramètre *ovr* pour la classification

C.3 M3 : Voting classifier avec 5 estimateurs

Ce nouveau modèle est aussi un Voting classifier et est composé de 5 sous modèles :

- ***SVM*** avec le RBF Kernel comme présenté précédemment

- ***Random Forest*** avec 100 estimaeurs
- ***Logistic Regression*** avec le noyau *liblinear* et le paramètre *ovr* pour la classification
- ***KNeighbors Classifier*** avec seulement deux voisins (bruité, non bruité)
- ***Gradient Boosting Classifier*** avec 100 estimateurs, une fonction de perte réglée sur *déviance* et *pas d'apprentissage* de 1.0