Spectro-imagerie et apprentissage profond : application à la détection de maladies de plantes

Clément Douarre

▶ **To cite this version:**

Clément Douarre. Spectro-imagerie et apprentissage profond : application à la détection de maladies de plantes. Vision par ordinateur et reconnaissance de formes [cs.CV]. Université Lumière Lyon 2, 2021. Français. NNT : . tel-03690297

HAL Id: tel-03690297
https://hal.science/tel-03690297

Submitted on 8 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

École Doctorale Informatique et Mathématiques de Lyon

THÈSE

pour obtenir le grade de

DOCTEUR EN INFORMATIQUE

présentée et soutenue publiquement par

Clément DOUARRE

le 25 mai 2021

Spectro-imagerie et apprentissage profond : application à la détection de maladies de plantes

Directeur·rice·s de thèse : **David ROUSSEAU**, **Laure TOUGNE**
Co-encadrant de thèse : **Carlos CRISPIM-JUNIOR**
Tuteur entreprise : **Anthony GELIBERT**

Devant le jury composé de :

Rapporteure	**M^me^ Marie CHABERT**	Professeure, ENSEEIHT, Toulouse
Rapporteure	**M^me^ Christelle GÉE**	Professeure, Agrosup, Dijon
Rapporteur	**M. Antoine VACAVANT**	MCF (HDR), Institut Pascal, Le Puy-en-Velay
Examinateur	**M. Christophe GODIN**	Directeur de recherches, INRIA, Lyon
Directeur de thèse	**M. David ROUSSEAU**	Professeur, Université d'Angers, Angers
Directrice de thèse	**M^me^ Laure TOUGNE**	Professeure, Université Lyon 2, Lyon
Co-encadrant de thèse	**M. Carlos CRISPIM-JUNIOR**	MCF, Université Lyon 2, Lyon
Tuteur entreprise	**M. Anthony GELIBERT**	Docteur, Carbon Bee, Châteauneuf-sur-Isère

Université Lyon 2, LIRIS, UMR CNRS 5205, F-69676, 5 avenue Pierre Mendès-France, 69500 Bron, France
Université d'Angers, LARIS, UMR INRAE IRHS, 62 Avenue Notre Dame du Lac, 49000 Angers, France
Carbon Bee, 11 rue Olivier de Serres, Parc du 45ème Parallele, Rovaltain, 26300 Châteauneuf-sur-Isère, France

Remerciements

Avant toute chose, je remercie, par ordre d'apparition dans mon parcours professionnel, mes quatre encadrants pour leur soutien ininterrompu et leurs conseils avisés :

— David ROUSSEAU, pour avoir fait naître chez moi l'étincelle de la recherche en premier lieu et pour m'avoir fidèlement accompagné sur le chemin qui s'est ouvert en conséquence ;

— Laure TOUGNE, pour sa grande compétence dans le domaine de la vision par ordinateur et sa vista dans le monde de la recherche en général ;

— Carlos CRISPIM-JUNIOR, pour avoir toujours été à jour vis à vis du domaine de l'apprentissage profond et pour ses retours fouillés sur mes ébauches de papiers ;

— Anthony GELIBERT, pour sa véritable âme de chercheur — cette dénomination est parfois un sujet de tendre raillerie au sein de Carbon Bee, mais dans un manuscrit de thèse, n'est-ce pas le plus grand éloge que l'on puisse faire ?

… ainsi que Gérald GERMAIN, président de Carbon Bee pour son expertise technique et sa vision inspirante.

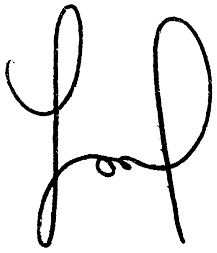
Je remercie par ailleurs Carbon Bee et l'Association Nationale de la Recherche et de la Technologie d'avoir financé ce travail.

Je salue tous mes collègues de l'entreprise et du laboratoire, en particulier Amélie, Aurélie, Cécilia, Debaleena, Dev', Florent, Guillaume, Jean-Patrick, Jonas, Mehdi, Pierre, Rémis[0 : 2] et Romain. Je remercie mes colocataires Alix, Jean-Bob et Tristan d'avoir maintenu une ambiance studieuse à l'appartement même lorsque les semaines de télétravail s'éternisaient au cours des différents confinements. Je remercie aussi mes parents — que j'aurais paraît-il oublié de citer lors de ma cérémonie de remise de diplôme d'ingénieur —, thésés tous les deux, pour m'avoir mollement encouragé à poursuivre le schéma familial.

Special thanks à Rémi[1] pour la maintenance des clusters de calcul du laboratoire et à Floriane pour le soutien moral.

Enfin, en ces temps de complotisme galopant, *fake news* décomplexées, et méfiance généralisée envers la science par une certaine frange de la population, je dédie ce travail aux chercheur·euses intègres de tous les domaines. Merci à ces femmes et ces hommes qui suivent sans transiger la méthode scientifique, en gardant leur cervelle chaude et leur tête froide.

Le 18/02/2021,
Clément DOUARRE



Résumé

Cette thèse est le fruit d'une collaboration entre les laboratoires LIRIS et LARIS et l'entreprise Carbon Bee, un acteur français des technologies numériques pour l'agriculture. Carbon Bee développe une caméra couplée à un algorithme d'apprentissage profond à des fins de pulvérisation ciblée de produits phytosanitaires. Sont regroupés dans cette caméra plusieurs capteurs permettant de réaliser des acquisitions dans différentes gammes de longueurs d'onde. Nous y trouvons en particulier un capteur infrarouge ainsi qu'un capteur hyperspectral instantané peu étudié jusqu'alors : le spectromètre imageur par tomographie (*Computed Tomography Imaging Spectrometer* en anglais, ou CTIS). Ce capteur permet une acquisition rapide d'une information spectrale riche mais qu'il est nécessaire de post-traiter par un algorithme de reconstruction pour la rendre interprétable par l'œil humain. Dans ce travail, nous nous sommes intéressés à l'exploitation optimale des différents capteurs de cette caméra, pour un cas d'étude à fort intérêt agronomique : la détection de la tavelure du pommier.

Nous nous sommes tout d'abord concentrés sur l'exploitation du signal produit par le CTIS, dans un cadre de classification d'images de feuilles saines et atteintes de lésions de tavelure. Nous avons développé une approche qui permet de s'affranchir de l'étape de reconstruction en conduisant un apprentissage directement dans l'espace brut des images CTIS, une démarche dite d'apprentissage comprimé. La conception d'une nouvelle architecture neuronale a permis d'obtenir des performances d'apprentissage supérieures à celles permises par la procédure classique, et ce en réduisant substantiellement les temps de calcul associés. Ces recherches ont par ailleurs mené au développement de plusieurs nouveaux simulateurs d'images permettant de pallier le manque d'images réelles annotées, une difficulté prégnante dans le domaine de l'apprentissage profond, et en particulier lors de l'étude de nouveaux systèmes d'imagerie.

Les travaux portant sur le CTIS ayant été menés à l'échelle de la feuille de pommier individuelle, nous nous sommes par la suite focalisés sur un contexte plus exigeant, proche des situations industrielles rencontrées par Carbon Bee. Nous avons cherché à optimiser des détections de lésions de tavelure menées au niveau du pixel dans des images infrarouges représentant des canopées de feuilles, et ce avec un nombre restreint de données annotées. À cette fin, nous avons développé plusieurs simulateurs d'images inspirés des derniers développements dans la matière en sciences végétales. Nous avons en particulier conçu un simulateur de canopées dont les images ont permis de substantiellement réduire la quantité de données réelles annotées nécessaire pour mener à bien une segmentation dans ce contexte.

Enfin, la présence au sein de la caméra de plusieurs capteurs aux résolutions spatiales et spectrales différentes ouvrait la voie à l'utilisation conjointe des informations qu'ils fournissaient, un procédé connu sous le nom de fusion de données. Nous avons exploré plusieurs pistes de travail dans ce cadre.

Mots-clés : apprentissage profond, vision par ordinateur, apprentissage comprimé, imagerie hyperspectrale, spectromètre imageur par tomographie.

Abstract

This thesis is the result of a collaboration between the LIRIS and LARIS laboratories and Carbon Bee, a French company focused on developing digital technology for agriculture. Carbon Bee develops a camera coupled with a deep learning algorithm in order to conduct spot spraying of crop protection products. This camera contains several sensors which allow for acquisitions in different wavelength ranges. It includes in particular an infrared sensor along with a snapshot hyperspectral spectrometer seldom studied until now : the Computed Tomography Imaging Spectrometer (CTIS). This sensor allows for a fast acquisition of rich spectral information. However, it is necessary to post-process this information via a reconstruction algorithm to make it understandable to the human eye. In this work, we have taken interest in the optimal use of these sensors for a case study with a high agronomic impact : the detection of apple scab.

We focused at first on the analysis of the CTIS signal in the context of a binary classification between images of healthy and diseased leaves. We developed a procedure which allowed to bypass the reconstruction algorithm by training a neural network directly on raw CTIS images, an approach known as compressed learning. Using a novel neural architecture allowed us to achieve a classification performance higher than the one obtained following the classical reconstruction pipeline, while substantially reducing the related training and inference times. This study led to the development of several novel image simulators which allowed to compensate for the low number of annotated images, an oft-encountered hurdle in deep learning studies, especially when working with a new imaging system.

While the work we have conducted on the CTIS images was carried out at the leaf scale, we afterward focused on a more demanding context, closer to the industrial challenges faced by Carbon Bee. We strove to improve scab detection at a pixel level in infrared images of leaf canopies ; what is more, with a limited quantity of annotated data. For this purpose, we developed several image simulators inspired by the latest trends in the plant sciences domain. In particular, we designed a canopy image simulator whose images enabled us to considerably reduce the number of annotated images necessary to conduct a segmentation in this context.

Finally, the presence of several sensors in the camera paved the way to the combination of the information that they gathered, a process known as data fusion. We have explored several pathways within this framework.

Keywords : deep learning, computer vision, compressed learning, hyperspectral imaging, computed tomography imaging spectrometer.

Notations

Acronymes et sigles

Ce tableau présente les acronymes et sigles employés dans ce manuscrit. Leur signification est par ailleurs systématiquement précisée lors de leur première utilisation dans le texte.

Si ce document est lu en version numérique via un logiciel de visualisation de PDF qui supporte les hyperliens, alors cliquer sur les instances des acronymes dans le texte renverra à leur définition dans ce tableau.

Acronyme	Signification	Traduction (si nécessaire)
CASSI	*Coded Aperture Snapshot Spectral Imager*	imageur spectral instantané à ouverture codée[1]
CCD	*Charge-Coupled Device*	dispositif à transfert de charge
CIE	Commission Internationale de l'Éclairage	
CNN	*Convolutional Neural Network*	réseau de neurones convolutif
CT	*Computed Tomopgrahy*	tomographie assistée par ordinateur
CTIS	*Computed Tomopgrahy Imaging Spectrometer*	spectromètre imageur par tomographie[1]
DCGAN	*Deep Convolutional Generative Adversarial Network*	réseau antagoniste génératif convolutif profond
DEL	Diode Électro-Luminescente	
ECC	*Enhanced Correlation Coefficient*	coefficient de corrélation amélioré
EM	*Expectation-Maximization*	espérance-maximisation
FBP	*Filtered Back-Projection*	rétro-projection filtrée
FC	*Fully Connected*	entièrement connecté(es)
FN, FP, VN, FP	Faux Négatif, Faux Positif, Vrai Négatif, Vrai Positif	
FSL	*Few-Shot Learning*	apprentissage en peu d'exemples[1]
GAN	*Generative Adversarial Network*	réseau antagoniste génératif
GAP	*Global Average Pooling*	regroupement par moyennage global[1]
IA	Intelligence Artificielle	
ILSVRC	*ImageNet Large Scale Visual Recognition Challenge*	défi de reconnaissance visuelle à grande échelle d'ImageNet[1]
IR	InfraRouge	
IRHS	Institut de Recherche en Horticulture et Semences	
MART	*Multiplicative Algreabric Reconstruction Technique*	technique de reconstruction algébrique multiplicative[1]
MCC	*Matthews Correlation Coefficient*	coefficient de corrélation de Matthews
MLP	*MultiLayer Perecptron*	perceptron multi-couches
NDVI	*Normalized Difference Vegetation Index*	indice de végétation de différence normalisée
ReLU	*Rectified Linear Unit*	unité linéaire rectifiée
RVB	Rouge Vert Bleu	
SIFT	*Scale-Invariant Feature Transform*	transformation de caractéristiques invariante à l'échelle
VGG	*Visual Geometry Group*	groupe de géométrie visuelle[1]

1. La traduction proposée est la nôtre car il n'existe pas de terme français « officiel ».

Objets mathématiques

Ce tableau présente les notations que nous adoptons dans le manuscrit pour désigner les différents objets mathématiques employés.

Notation	Signification
hauteur × largeur	Dimensions d'un objet bidimensionnel.
hauteur × largeur × profondeur	Dimensions d'un objet tridimensionnel.
[valeur basse, valeur haute]	Intervalle de valeurs.
{valeur 1, valeur 2}	Ensemble de valeurs.
(coordonnée 1, coordonnée 2)	Coordonnées d'un point dans un objet bidimensionnel.
objet[indice]	Élément à la position « indice » d'un objet unidimensionnel.
objet[:, :, indice]	Tranche à la position « indice » d'un objet tridimensionnel.

Table des matières

Remerciements **iii**

Résumé **v**

Abstract **vii**

Notations **ix**

Table des matières **xi**

Introduction **1**
Contexte général 2
Contexte industriel : Carbon Bee 3
Contraintes et verrous scientifiques 4
Plan du manuscrit 6

1 Les défis d'aujourd'hui pour la détection de maladies de plantes **7**
1.1 Intérêt de l'imagerie invisible pour les plantes 8
1.2 Apprentissage profond pour les images de plantes 11
1.3 Tavelure du pommier 23
1.4 Conclusion 26

2 Le CTIS : un capteur hyperspectral atypique à évaluer **27**
2.1 Principes de l'imagerie spectrale 28
2.2 Le CTIS : un capteur du domaine de l'imagerie computationnelle 33
2.3 Conclusion 42

3 Des simulateurs pour évaluer le capteur CTIS **43**
3.1 Simulateur de cubes hyperspectraux de feuilles tavelées 44
3.2 Simulateur de CTIS 53
3.3 Création des jeux simulés 60
3.4 Conclusion 61

4 Apprentissage comprimé sur images CTIS **65**
4.1 Un cadre commun pour les expérimentations 66
4.2 Une performance de référence : l'apprentissage sur cubes reconstruits 69
4.3 L'apprentissage comprimé : une alternative viable pour le CTIS 74
4.4 Une architecture dédiée aux images CTIS 78
4.5 Les performances subsistent malgré l'ajout de bruit 85
4.6 Une plus grande polyvalence par rapport aux imageurs classiques 87
4.7 Une influence forte des paramètres optiques 88
4.8 Conclusion 89

5 Simulation d'images pour alléger la charge d'annotation d'images réelles **91**
5.1 Une annotation chronophage à réduire 92
5.2 Les différentes catégories de simulateurs 98
5.3 Simulateurs implémentés pour notre cas d'étude 101
5.4 Résultats : des simulateurs efficaces 111
5.5 Conclusion 119

6 Vers une fusion spatio-spectrale **121**
6.1 Un pan important de la vision par ordinateur 122
6.2 Un gain pour la fusion d'images alignées 123
6.3 Les difficultés de fusionner des imageries non standard 125
6.4 Conclusion 127

Conclusion et perspectives **129**
Retour et perspectives pour les travaux effectués 130
Vers un monde plus sobre 134
Valorisations 136

Annexes **137**
A Combler « l'écart de la réalité » 138
B Recalage d'images multimodales 146

Références **153**

Introduction

Sommaire

Contexte général **2**
Contexte industriel : Carbon Bee **3**
Contraintes et verrous scientifiques **4**
Plan du manuscrit **6**

Contexte général

L'augmentation rapide de la population mondiale fait de l'agriculture un domaine clé pour l'humanité [Golhani et al., 2018]. La demande alimentaire ne cesse de croître, et les agriculteurs doivent faire face aux défis menaçant les plantations. Les stress dits biotiques, c'est-à-dire ceux liés aux attaques d'éléments vivants tels que les insectes ravageurs, mais aussi des maladies découlant d'infections par des parasites, virus et champignons, constituent la menace principale de l'agriculture moderne [Strange and Scott, 2005]. Tous les types de culture sont affectés par des maladies, y compris les plants consommés par les humains ou le bétail (céréales, légumineuses, fruits, etc.) [Kaur et al., 2019]. Dans ce contexte, les études visant à réduire l'impact de ces maladies sont nombreuses. Un grand nombre de moyens de lutte chimique ont été conçus au cours des dernières décennies, en particulier lors de la « révolution verte » [Cooper and Dobson, 2007]. En parallèle, une lutte génétique s'est développée par le biais de la culture de cultivars, c'est-à-dire des plants sélectionnés pour leurs traits de résistance aux maladies. Que ce soit dans le cadre de l'application de produits phytosanitaires en champ ou de culture de cultivars en serre, la détection des symptômes de ces maladies est primordiale pour évaluer la performance des solutions déployées.

Historiquement, la détection de maladies était conduite par un contrôle visuel humain du phénotype de la plante ou grâce à des tests destructifs et chronophages car réalisés en laboratoire, tels que les tests enzymatiques [Golhani et al., 2018]. Les techniques de détection par imagerie ont naturellement trouvé leur place dans ce domaine en tant que procédés rapides et non-invasifs [Mahlein, 2016]. Ces méthodes consistent en l'acquisition d'images des plantes via des systèmes optiques et des capteurs photographiques. Elles sont particulièrement pertinentes dans le cadre de la détection de maladies végétales car les réactions d'une plante à une attaque se manifestent la plupart du temps par des symptômes visuels. Ces méthodes d'acquisition faciles à mettre en œuvre, bon marché et non-destructives ont mené à une quantité importante de données à traiter. L'analyse de ces données est alors devenue le nouveau goulot d'étranglement de la série d'opérations (*pipeline* en anglais) de détection de symptômes [Singh et al., 2016]. En conséquence, des méthodes d'apprentissage automatique (*machine learning* en anglais) ont été implémentées afin d'accélérer l'analyse des images acquises en proposant une classification automatique de celles-ci sans qu'une expertise humaine soit nécessaire. Dans les années 2000, un *pipeline* typique d'une détection automatique de maladies végétales via un système d'imagerie se déroulait selon les étapes suivantes [Kaur et al., 2019] :

1. acquisition de l'image de la scène via un capteur, en général en couleurs Rouge Vert Bleu (RVB) ;
2. définition et extraction de *caractéristiques* (couleur, forme, etc.) ;
3. classification de l'image en se basant sur ces caractéristiques via un algorithme d'apprentissage automatique.

Cependant, au cours de ces dernières années, deux développements matériels importants ont ouvert de nouvelles perspectives pour les scientifiques et les industriels. Premièrement, le coût des systèmes d'imagerie scientifique a fortement baissé [Mathews, 2008]. Ceci a mené à une adoption forte de capteurs autres que RVB, en particulier des capteurs dits *hyperspectraux*, permettant d'acquérir de l'information dans un grand nombre de longueurs d'onde, y compris hors du domaine visible. Ces capteurs sont particulièrement utiles dans le champ des sciences végétales, puisqu'une grande partie des activités internes d'une plante entraîne des symptômes visibles dans des gammes de l'ultraviolet et de l'infrarouge (IR) [Li et al., 2014]. La figure I.1 illustre par exemple l'intérêt manifeste de l'imagerie IR pour la détection de la tavelure du pommier sur des feuilles. Les taches sombres visibles en IR indiquent les lésions dues à la tavelure. Ces lésions sont beaucoup plus difficiles à distinguer, voire invisibles en imagerie RVB.

Deuxièmement, la capacité de calcul des ordinateurs a fortement progressé et l'utilisation de cartes graphiques a permis le développement d'algorithmes d'apprentissage dits *profonds*

FIGURE I.1 – Illustration de l'intérêt de l'imagerie IR pour la détection de maladies. L'image est un montage composé d'une acquisition RVB (haut) et d'une acquisition IR (bas) de feuilles de pommier atteintes de la tavelure du pommier. Les feuilles avaient été inoculées avec le champignon responsable de la tavelure quatorze jours auparavant. Source : acquisitions à l'IRHS avec la caméra Carbon Bee.

(*deep learning* en anglais) [Goodfellow et al., 2016]. Couplés à des jeux de données de très grande taille, ces algorithmes ont révolutionné le champ de l'apprentissage automatique ainsi que celui de la vision par ordinateur et constituent l'état de l'art pour la grande majorité des tâches de classification d'images [Voulodimos et al., 2018]. Ces deux axes de progression technologique permettent l'acquisition d'une information plus riche et un traitement plus complet et efficace de cette information. Ils sont intégrés de manière croissante dans les *pipelines* actuels de détection de maladies de plantes en tant que domaines d'innovation clés.

Contexte industriel : Carbon Bee

Cette thèse a été réalisée dans le cadre d'un dispositif de Convention Industrielle de Formation par la REcherche (CIFRE) en partenariat avec l'entreprise Carbon Bee (CIFRE n°2017/0639). L'entreprise a été créée en 2015 à Saint-Marcel-Lès-Valence (26). Une filiale dédiée à l'agronomie nommée Carbon Bee AgTech a été créée en 2017. Cette dernière propose des solutions pour l'entretien et le soin des champs dans le cadre de l'agriculture de précision. Dans ce paradigme, le champ est considéré comme l'échelle pertinente de travail [Zhang et al., 2002]. Il s'agit donc de tenir compte des variabilités internes à ce champ plutôt que de le considérer comme un bloc monolithique auquel on appliquerait des traitements phytosanitaires de manière uniforme. Carbon Bee AgTech s'intéresse entre autres à la détection de maladies et propose leur localisation dans la parcelle étudiée.

Pour réaliser cette détection, Carbon Bee se base sur l'imagerie et commercialise une caméra dédiée à cette tâche. Cette caméra est associée à un algorithme d'apprentissage profond qui analyse les images et repère la présence de la cible recherchée. Le but de cet algorithme est de procéder à une *segmentation*, c'est-à-dire la détection de la cible à l'échelle du pixel (figure I.2).

Carbon Bee AgTech propose des solutions où les caméras sont montées sur des drones ou tenues à la main, mais le cas d'usage le plus fréquent pour l'entreprise est le montage d'un ensemble de caméras sur des « rampes » axiales portées par des tracteurs (figure I.3). Ces rampes sont des structures métalliques déployées orthogonalement au sens de déplacement du tracteur, comme les ailes d'un oiseau. Sur ces rampes sont positionnées, à espacement régulier, des contenants de produits phytosanitaires, équipés de buses permettant de réguler leur débit d'épandage. Chaque

(a) Image originale.

(b) Segmentation réalisée. Les pixels indiqués en rouge correspondent aux lésions repérées par l'algorithme d'apprentissage.

FIGURE I.2 – Un exemple de segmentation réalisée par Carbon Bee AgTech sur une image RVB représentant des feuilles de vigne atteintes d'esca. Source : Carbon Bee AgTech (https://carbonbee-agtech.fr).

caméra est associée à un contenant et commande l'ouverture de la buse si et seulement si une cible est détectée. Le but *in fine* est de permettre aux agriculteurs d'appliquer des produits phytosanitaires de manière localisée et donc d'en réduire la quantité nécessaire. Cette réduction de produits a pour double bénéfice une diminution des coûts pour l'agriculteur ainsi qu'une limitation de l'impact chimique sur l'environnement.

(a) Vue d'ensemble du tracteur équipé de rampes. Les flèches jaunes indiquent la position de deux caméras.

(b) Vue rapprochée d'une caméra.

FIGURE I.3 – Un tracteur équipé de rampes axiales sur lesquelles sont montées plusieurs caméras Carbon Bee. Source : Carbon Bee AgTech.

Contraintes et verrous scientifiques

Procéder à une détection dans les conditions réelles d'utilisation de la caméra entraîne plusieurs contraintes. Premièrement, pour offrir un avantage agronomique significatif, la caméra doit pouvoir obtenir une information suffisamment riche afin de permettre une détection fine des cibles. En particulier, il est attendu que des maladies soient identifiées au stade le plus précoce possible. Ensuite, il faut que l'acquisition soit réalisée dans un laps de temps compatible avec la vitesse du support. L'utilisation attendue de cette caméra est que son analyse ne nécessite pas un ralentissement du déplacement de l'agent : on parle d'analyse en « temps réel ». En particulier,

dans le cas de l'épandage par un tracteur, Carbon Bee a calculé, en prenant en compte la vitesse de déplacement du tracteur, le temps d'activation de la buse et le temps nécessaire pour l'épandage du produit, que l'acquisition par la caméra et l'analyse qui suit doivent être réalisées en moins de 200 ms. En outre, la caméra doit être de taille relativement réduite et légère pour pouvoir être montée sur la variété de supports nécessaires aux diverses tâches de détection que propose l'entreprise.

Nous pouvons ajouter à ces contraintes techniques une contrainte économique. Les capteurs doivent être les moins onéreux possibles afin que l'offre de Carbon Bee soit compétitive sur le marché. Ce sont ces impératifs qui ont façonné la caméra de l'entreprise. Pour permettre à la fois l'acquisition d'une information riche et de respecter les impératifs de vitesse, de taille et de coût, l'entreprise a fait le choix d'inclure trois capteurs distincts dans la caméra (figure I.4) :

— un capteur RVB,

— un capteur IR,

— un capteur hyperspectral bas-coût.

FIGURE I.4 – La camera Carbon Bee vue de face. Les trois flèches jaunes indiquent les positions des objectifs des trois capteurs. Source : Carbon Bee AgTech.

Cependant, l'utilisation de ces capteurs de façon optimale est sujette à plusieurs verrous scientifiques.

— Le capteur hyperspectral choisi est encore peu étudié par la communauté des chercheur·euses en imagerie spectrale. Il s'agit du spectromètre imageur par tomographie (*Computed Tomography Imaging Spectrometer* en anglais, ou CTIS) [Descour and Dereniak, 1995]. Ce capteur, développé dans les années 1990, n'a à notre connaissance jamais été utilisé pour des applications industrielles avant son emploi par Carbon Bee. Cependant, l'intérêt s'y rapportant va croissant car ce capteur est robuste, bon marché et propose une acquisition rapide de l'information hyperspectrale [Salazar-Vazquez and Mendez-Vazquez, 2020]. L'utilisation du CTIS est cependant limitée par certains inconvénients du système. En particulier, les résolutions spatiale et spectrale des images sont réduites par rapport à des capteurs hyperspectraux plus conventionnels [Hagen et al., 2006] et l'acquisition de l'information hyperspectrale complète nécessite une étape de calcul.

— La caméra de Carbon Bee inclut d'autre part un capteur IR, une gamme de longueurs d'onde très étudiée en sciences végétales [Mahlein, 2016], en particulier dans le cadre de détection de maladies [Jones, 2004]. Ce capteur est utilisé par l'entreprise dans des contextes de vision difficiles, en particulier à des fins de segmentations de maladies dans des environnements agronomiques complexes tels que des canopées de feuilles enchevêtrées. À cette échelle apparaissent des problématiques de préparation des données d'apprentissage qui peuvent devenir rédhibitoires par rapport au gain attendu par ce procédé.

— Enfin, la présence de multiples capteurs ouvre la voie à la *fusion de données* [Baltrušaitis et al., 2018], c'est-à-dire la combinaison d'informations de plusieurs sources afin de mener à bien une tâche. Combiner plusieurs images est un procédé implanté depuis longtemps dans certains domaines de vision par ordinateur tels que la détection de silhouettes [Hwang et al., 2015] et qui se popularise plus récemment dans le cadre des sciences végétales [Mahlein, 2016]. Cependant, plusieurs difficultés surviennent dans le cas du système Carbon Bee, liées notamment à la nature différente des images acquises par les capteurs.

Il nous a paru pertinent de mener plusieurs études afin de tenter de lever ces verrous. Nous avons conduit ces travaux dans le cadre d'une application agronomique au fort impact économique : la détection de la tavelure du pommier. Il s'agit d'une maladie fongique affectant les pommiers, première cause de perte de production de pommes au niveau mondial [Bowen et al., 2011].

La thèse a été menée sous la direction de deux laboratoires :

— le Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS), équipe Imagine, à Lyon (69). Ce laboratoire possède une expertise dans les domaines de la vision par ordinateur et de l'apprentissage automatique, appliquée en particulier aux sciences végétales.

— le Laboratoire Angevin de Recherche en Ingénierie des Systèmes (LARIS), équipe « Information, Signal, Image et Sciences du Vivant », à Angers (49). Ce laboratoire possède une expertise concernant les instrumentations d'imagerie pour les sciences végétales. Le laboratoire a en outre à sa disposition plusieurs serres dans lesquelles diverses espèces de plantes et leurs réactions aux maladies sont étudiées, via un partenariat avec l'Institut de Recherche en Horticulture et Semences (IRHS).

Plan du manuscrit

Le manuscrit est organisé de la façon suivante :

— Le chapitre 1 présente les éléments théoriques et bibliographiques relatifs aux deux défis actuels de la détection de maladies de plantes : l'imagerie hors du domaine du visible et l'apprentissage profond.

— Les chapitres 2, 3 et 4 sont dédiés à une étude que nous avons menée concernant l'exploitation optimale des images d'un capteur CTIS dans un cadre d'apprentissage automatique. Plus précisément, nous nous sommes intéressés au cas d'une détection de lésions de tavelure à l'échelle de la feuille. En conséquence de la relative jeunesse de ce capteur CTIS, nous nous sommes tournés vers la simulation de données pour se donner les moyens d'analyser ses capacités, un recours très utilisé lors d'études de capteurs innovants [Spoelder, 1999]. Le chapitre 2 détaille le fonctionnement du CTIS et le positionne par rapport aux autres capteurs hyperspectraux existants. Le chapitre 3 présente les différents simulateurs que nous avons développés pour générer des signaux produits par ce capteur. Le chapitre 4 contient l'approche d'apprentissage que nous avons explorée pour exploiter les signaux du CTIS en contournant certains inconvénients de ce spectromètre.

— Dans le chapitre 5, nous nous sommes tournés vers un contexte agronomique plus proche des défis industriels de Carbon Bee, l'échelle de la canopée, et intéressés à la problématique du manque d'images annotées. Nous avons développé en réponse plusieurs simulateurs d'images IR inspirés des derniers développements dans la matière en sciences végétales.

— Enfin, le chapitre 6 présente plusieurs pistes de travail que nous avons explorées concernant la fusion des images des différents capteurs de la caméra. Nous nous sommes intéressés en particulier au cas de la combinaison d'images à l'information structurelle différente et aux problématiques de recalage provenant du décalage physique entre les différents capteurs.

Chapitre 1

Les défis d'aujourd'hui pour la détection de maladies de plantes

Dans ce chapitre, nous présentons les éléments théoriques et une bibliographie relative aux deux défis technologiques actuels dans le domaine de la détection de maladies végétales. Premièrement, nous nous intéressons à l'imagerie hors du domaine du visible qui permet l'obtention d'une large gamme d'informations concernant la santé d'une plante. Nous introduisons les notions de physique électromagnétique nécessaires avant de nous concentrer sur les apports de l'imagerie IR et hyperspectrale dans le domaine des sciences végétales. Deuxièmement, nous présentons les méthodes d'apprentissage profond qui ont révolutionné le champ de l'apprentissage automatique, en particulier dans le domaine de la vision par ordinateur. Nous nous attardons en particulier sur les difficultés spécifiques du domaine des sciences végétales. Enfin, nous présentons les travaux déjà menés sur notre cas d'étude : la détection de la tavelure du pommier.

Sommaire

1.1 Intérêt de l'imagerie invisible pour les plantes 8
1.1.1 Principe d'un spectre électromagnétique 8
1.1.2 Les plantes ont des spectres de réflexion complexes 9
1.1.3 L'imagerie invisible permet d'évaluer finement la santé d'une plante 10
1.1.4 Positionnement 11
1.2 Apprentissage profond pour les images de plantes 11
1.2.1 Apprentissage profond et réseaux de neurones 12
1.2.2 Spécificités du domaine de la détection de maladies de plantes 20
1.2.3 Positionnement 23
1.3 Tavelure du pommier 23
1.3.1 Une infection difficile à détecter et à soigner 23
1.3.2 De nombreuses imageries ont été employées pour sa détection 24
1.3.3 Positionnement 25
1.4 Conclusion 26

1.1 Intérêt de l'imagerie invisible pour les plantes

Nous commençons par présenter les différents éléments théoriques relatifs à l'imagerie hors du domaine du visible, en nous attardant en particulier sur l'impact de cette technologie dans le domaine des sciences végétales.

1.1.1 Principe d'un spectre électromagnétique

La lumière est une onde électromagnétique caractérisée par sa longueur d'onde, c'est-à-dire sa période spatiale, exprimée en mètres. La valeur de la longueur d'onde est inversement corrélée avec l'énergie que porte cette onde, exprimée en joules. La longueur d'onde d'une onde électromagnétique conditionne ainsi la façon dont l'onde peut être exploitée par les êtres vivants et les systèmes technologiques. Nous présentons quelques-unes des gammes qui ont un intérêt particulier pour les travaux de ce manuscrit, par longueur d'onde croissante :

— l'ultraviolet ;

— le domaine visible, qui correspond aux ondes que l'œil humain peut capter et que le cerveau humain peut interpréter. Chaque longueur d'onde correspond à une couleur spécifique pour le cerveau. Par opposition, les autres gammes de longueurs d'onde sont parfois regroupées sous le terme de domaine *invisible*.

— l'IR. Nous pouvons diviser cette gamme en plusieurs sous-gammes, suivant plusieurs nomenclatures. Dans ce document, nous adoptons la division de la Commission Internationale de l'Éclairage (CIE) [2] :

  — l 'IR-A, ou l'IR proche ;

  — l 'IR-B, ou l'IR court ;

  — l 'IR-C, ou l'IR thermique.

Les gammes de longueur d'onde correspondant aux différents domaines sont précisées figure 1.1.

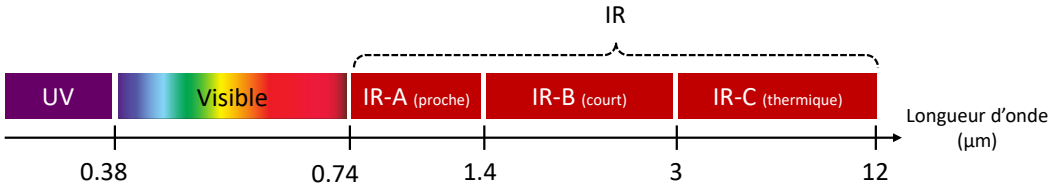


FIGURE 1.1 – Représentation des différentes gammes de longueurs d'onde entre l'ultraviolet et l'IR thermique, ordonnées en fonction de la longueur d'onde (échelle logarithmique).

On appelle *spectre* d'une scène l'ensemble des ondes électromagnétiques qui émanent de cette scène. Ce spectre est composé d'ondes de différentes longueurs, émises avec différentes intensités. On le représente comme une fonction traduisant l'intensité du rayonnement émis en fonction de la longueur d'onde. Sur la surface terrestre, le spectre de la plupart des objets est la somme de deux types de spectres. Tout d'abord, tout objet opaque et dont la température est supérieure au zéro absolu émet spontanément un spectre électromagnétique dit « thermique » qui dépend de cette température selon la loi de Planck [Planck, 2013]. Pour un objet à la température T, en notant $I(\lambda)$ l'intensité du spectre à la longueur d'onde λ, alors ce spectre se calcule comme

$$I(\lambda, T) = \frac{2k_1}{\lambda^5} \frac{1}{e^{\frac{k_2}{\lambda T}} - 1}, \tag{1.1}$$

où k_1 et k_2 sont des constantes dépendant de la constante de Planck, la constante de Boltzmann et la vitesse de la lumière. Plus la température de l'objet est élevée, plus cet objet émet dans des longueurs d'onde courtes. Les êtres et les objets à température ambiante sur Terre émettent un spectre

2. http://cie.co.at/eilv/580.

compris dans le domaine de l'IR thermique (qui en tire son nom). Le soleil, dont la température est d'environ 5800 K, émet un rayonnement thermique qui s'étend de l'IR thermique à l'ultraviolet, en passant par le visible.

Par ailleurs, tout objet non transparent exposé à la lumière renvoie une partie de cette lumière. Le facteur qui correspond à la proportion du rayonnement renvoyé par rapport à celui reçu s'appelle la *réflectance*. On appelle *spectre de réflexion* la réflectance exprimée en fonction de la longueur d'onde incidente. Le spectre de réflexion d'un objet dépend de la composition chimique et de la structure interne de l'objet. Le spectre perçu par un œil humain d'un objet à température ambiante et éclairé par la soleil correspond principalement au spectre de réflexion de cette lumière, plus intense que le spectre thermique.

1.1.2 Les plantes ont des spectres de réflexion complexes

Le spectre de réflexion des plantes est complexe car ce sont des organismes où divers mécanismes mènent à des régimes de réflectance bien distincts en fonction du domaine de longueur d'onde considéré [He et al., 2011]. La figure 1.2 présente un spectre de plante typique qui illustre ces différents régimes.

FIGURE 1.2 – Spectre de réflexion typique d'une feuille. Source : simulateur PROSPECT [Jacquemoud and Baret, 1990].

Nous pouvons distinguer plusieurs mécanismes biologiques en fonction de la gamme de longueur d'onde considérée. Dans le domaine visible, la forte concentration en pigments tels que la chlorophylle mène à une importante absorption de la lumière solaire [Blackburn, 2007]. Cette plage d'énergie est adéquate pour la génération de molécules carbonées via la photosynthèse. L'absorption est un peu moins forte dans le vert que dans le rouge et le bleu, ce qui explique la couleur des plantes. La lumière du domaine de l 'IR-A est énergétiquement peu intéressante pour les plantes. Les structures internes de celles-ci ont évolué de manière à ce que les multiples couches qui les composent mènent à une grande réflectance dans ce domaine, afin de ne pas provoquer un échauffement inutile [Gates, 2012]. Des altérations dans l'intégrité de ces couches internes peuvent provoquer une variation de la réflectance dans cette gamme [Mahlein, 2016]. Enfin, dans le domaine de l 'IR-B, plusieurs composants comme la cellulose, la lignine et d'autres protéines, mais aussi les molécules d'eau causent plusieurs spectres d'absorption indicatrices de la composition chimique de la plante [Elvidge, 1990].

Il apparait donc que de nombreuses informations concernant l'état physiologique de la plante sont accessibles dans le domaine invisible. Nous présentons maintenant les différentes applications existantes pour la détection de maladies dans ce domaine.

1.1.3 L'imagerie invisible permet d'évaluer finement la santé d'une plante

Les premières analyses en imagerie invisible se concentraient sur certaines gammes de longueurs d'onde spécifiques de l'IR. Les études plus modernes discrétisent pour la plupart l'information spectrale de façon plus fine.

Domaine infrarouge

Tout d'abord, des mesures dans l'IR-A été intégrées dans des calculs de valeurs appelées *indices*, qui correspondaient à des combinaisons arithmétiques de valeurs de réflectance à des longueurs d'onde différentes [Bannari et al., 1995]. Des capteurs IR ont ainsi été utilisés dans le domaine de la télédétection spatiale pour la segmentation de couverts végétaux à la surface terrestre [Rouse et al., 1974]. L'indice associé à ces études, nommé l'indice de végétation de différence normalisée (*Normalized Difference Vegetation Index* en anglais ou NDVI), est un ratio de luminosité entre une longueur d'onde de l'IR et une longueur d'onde du domaine visible proche du rouge. Le fort ratio des plantes entre la lumière réfléchie dans l'IR et celle dans le visible (figure 1.2) permet d'isoler la biosphère d'autres zones terrestres. Dans le cadre de la détection de maladies, une variation de l'IR proche réfléchi peut être signe d'une destruction des structures internes de la plante [Mahlein, 2016], en particulier des pigments [Peñuelas and Filella, 1998] à cause d'attaques parasitaires.

L'imagerie par IR-B a servi à différentes analyses quantitatives concernant la composition chimique fine des plantes. Plusieurs études se sont penchées sur la détermination de la quantité d'eau dans les plantes dans des conditions de stress hydrique grâce aux informations de ces longueurs d'onde [Buddenbaum et al., 2015; Kim et al., 2015]. D'autres se sont intéressées à la quantification des taux de nutriments absorbés tels que l'azote [Camino et al., 2018] ou d'autres composants d'intérêt des plantes tels que le tanin [Lehmann et al., 2015].

L'IR-C a aussi été beaucoup étudiée dans le cadre de la détection de maladies. La température est une mesure qui fournit des informations précieuses concernant les réactions d'une plante à un stress [Chaerle and Van Der Straeten, 2000]. La régulation de la température dans une plante s'effectue principalement par des pores, que l'on appelle des stomates, qui servent de voies d'échange d'eau et de gaz avec l'atmosphère. Ces stomates s'ouvrent et se ferment afin conserver ou d'évacuer l'eau afin de réguler la température de la plante [Kümmerlen et al., 1999]. En temps normal, cette régulation s'effectue en fonction de la quantité d'eau disponible dans la plante. Des travaux ont montré que certains parasites induisaient une fermeture de ces stomates similairement à l'effet d'un stress hydrique [Chaerle et al., 1999], provoquant un échauffement local de la plante. Des attaques parasitaires plus avancées peuvent mener à une percée des membranes cellulaires [Pennazio and Sapetti, 1982]. Le contenu aqueux des cellules est alors déversé, menant à une transpiration stomatique excessive et donc à un refroidissement local [Chaerle et al., 2001]. Il a été montré que l'imagerie thermique était adéquate pour détecter la présence de certaines attaques parasitaires, et notamment que les zones de variation de température coïncidaient avec les zones où les parasites s'implantaient [Chaerle et al., 2001]. Ces études montraient en particulier que la détection thermique pouvait précéder parfois de plusieurs jours l'apparition visible des parasites en surface de la plante, permettant ainsi un traitement plus anticipé des maladies.

Spectre complet

Les capteurs dits hyperspectraux, qui seront plus longuement détaillés au chapitre suivant, permettent d'acquérir pour un spectre donné l'intensité de chacune d'un grand nombre des longueurs d'onde qui le composent. En sciences végétales, la gamme spectrale ainsi étudiée couvre en général le domaine visible et une partie de l'IR-A. Les informations du spectre complet peuvent être exploitées pour des caractérisations plus fines que ne le permet l'information d'une réflectance IR ou visible seule. En particulier, les applications suivantes ont été possibles grâce à l'utilisation de capteurs hyperspectraux : la détection de stress hydriques ou d'attaques parasitaires significativement

plus tôt qu'en détection IR [Behmann et al., 2014] ; la classification de diverses espèces végétales [He et al., 2011], soit une analyse plus poussée que le classification binaire « plante/non plante » rendue possible par le NDVI ; la caractérisation biochimique d'espèces, telle que la composition pigmentaire [Ustin et al., 2004], l'analyse de stress hydrique multi-niveaux [Kim et al., 2011] ; la classification d'attaques parasitaires à un niveau fin, par exemple pour des maladies dont le spectre diffère peu du spectre sain [Bravo et al., 2003], pour distinguer plusieurs types de maladies [Rumpf et al., 2010] ou encore pour caractériser la date de la gravité de l'infection [Mahlein et al., 2012].

Méthodologiquement, les utilisations de l'information spectrale peuvent être séparées en deux catégories. La première possibilité est d'opérer une sélection de quelques longueurs d'onde parmi toutes celles offertes par l'imagerie hyperspectrale. Il est possible de sélectionner les longueurs d'onde pour créer des indices végétaux basés sur des connaissances biologiques de mécanismes spécifiques des plantes. Par exemple, les auteurs de [Rumpf et al., 2010] ont conduit une classification de maladies en choisissant comme caractéristiques de multiples indices très utilisés en sciences végétales donnant des indications sur les contenus en chlorophylle, en caroténoïdes, sur la biomasse en général. Il est aussi possible de procéder à la sélection automatique des longueurs d'onde les plus pertinentes par un algorithme statistique [Benoit et al., 2016]. Toutes les longueurs d'onde sont alors considérées dans un premier temps, et sont conservées uniquement celles qui contribuent le plus à l'application selon un certain critère. Par exemple, les auteurs de [Bravo et al., 2003] ont proposé une analyse de variance pour une sélection de longueurs d'onde « utiles » qu'ils fournissent ensuite en entrée à un algorithme de classification, pour des maladies affectant le blé. Les auteurs de [Delalieux et al., 2009b] ont testé exhaustivement tous les ratios de longueurs d'onde sur une gamme donnée afin d'évaluer leur impact pour la détection de tavelure par régression logistique.

La deuxième possibilité consiste à exploiter l'entièreté du spectre. Les spectres peuvent alors être condensés via une analyse en composante principale [Golhani et al., 2018] ou bien servir d'entrée tels quels à des algorithmes de classification de vecteurs à grande dimension. Les auteurs de [Mahlein et al., 2012] ont utilisé un tel algorithme appelé le *Spectral Angle Mapper* [Yuhas et al., 1992] pour la classification de plusieurs stades de maladies affectant la betterave sucrière. Les auteurs de [Qin et al., 2009] ont comparé des spectres de citrons sains avec ceux atteints de chancre afin de procéder à leur classification avec une variation de cet algorithme.

1.1.4 Positionnement

Le grand nombre d'études basées sur l'imagerie dans des domaines invisibles pour la détection de maladies de plantes montre à quel point cet axe de recherche est prometteur. Il faut cependant noter que ces recherches ont été conduites en grande majorité avec des capteurs qui ne concordent pas avec des contraintes industrielles comme celles de Carbon Bee. Les caméras utilisées pour l'imagerie thermique et l'imagerie hyperspectrale peuvent valoir jusqu'à plusieurs dizaines de milliers d'euros et ont des temps d'acquisition de l'ordre de la seconde. Le CTIS proposé par Carbon Bee est radicalement différent de ce type d'imagerie. Les particularités, promesses et difficultés du CTIS sont détaillées plus abondamment dans le chapitre 2.

1.2 Apprentissage profond pour les images de plantes

Nous présentons maintenant les bases théoriques relatives à un autre domaine qui a considérablement influencé les méthodes développées en détection de maladies végétales : l'apprentissage profond. Il existe de nombreuses ressources qui décrivent méticuleusement l'histoire, les attributs et les enjeux du champ de l'apprentissage profond. Nous nous limitons à décrire les blocs nécessaires à la compréhension du manuscrit. Nous nous attardons par la suite sur les enjeux liés à l'application de ces méthodes au domaine des sciences végétales.

1.2.1 Apprentissage profond et réseaux de neurones

Généralités sur l'apprentissage automatique supervisé

Nous rappelons d'abord les principes généraux du champ de l'apprentissage automatique, dont l'apprentissage profond est un sous-domaine. Nous nous concentrons ici sur le cas d'un algorithme de classification. D'autres tâches d'apprentissage automatique existent (segmentation, régression, système de recommandation, etc.) mais peuvent être dérivées à partir du cas de la classification. Le but d'un algorithme de classification est d'associer de façon automatique une *étiquette* à un objet. Nous nous plaçons dans le domaine de la vision par ordinateur, où l'objet est une image et l'étiquette peut être par exemple un mot qui décrit le contenu de l'image. Initialement, l'apport principal de ces algorithmes était une accélération de cette étiquetage lorsque le nombre d'objets en faisait une charge trop fastidieuse pour des humains. Aujourd'hui, certains de ces algorithmes ont progressé au point d'obtenir des performances surhumaines sur certaines tâches [He et al., 2015]. Nous nous limitons au cas des algorithmes dits « supervisés », c'est-à-dire ceux qui sont *entraînés* sur un ensemble d'images dont les étiquettes sont connues.

Nous détaillons à présent les étapes d'un tel apprentissage. Nous considérons pour illustration une classification binaire où nous cherchons à déterminer la présence ou l'absence de pommes dans des images (figure 1.3). Les étapes que nous décrivons sont valables pour n'importe quel algorithme d'apprentissage supervisé.

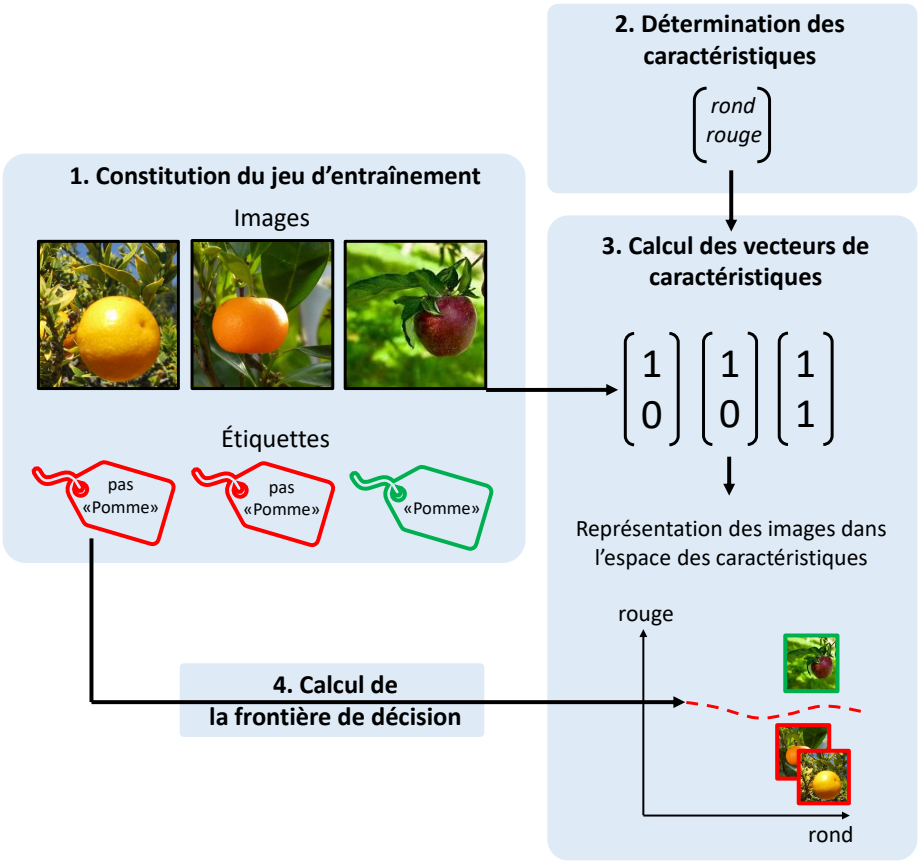


FIGURE 1.3 – Schéma représentant les étapes d'*entraînement* d'un algorithme d'apprentissage automatique pour une détection de pommes dans des images. Source des images de fruits : https://www.inaturalist.org.

1. Constitution d'un jeu d'entraînement : un ensemble d'images dont les étiquettes sont connues sont regroupées dans un jeu de données dit d'entraînement. On appelle *annotation* le processus d'attribuer des étiquettes aux images en premier lieu.

2. Détermination des caractéristiques : une caractéristique est une mesure que l'on fait sur les objets contenus dans l'image relativement à leur forme, leur couleur, leur gradient, leur longueur, etc. Dans l'application donnée en exemple, les caractéristiques retenues sont la présence de la couleur rouge (qui pourrait être calculée via une recherche de pixels dont la valeur est comprise dans une gamme représentant la couleur rouge) et de formes rondes (qui pourrait être calculée via un algorithme de détection de forme tel que la transformée de Hough [Duda and Hart, 1972]) dans l'image. Des caractéristiques plus élaborées ont été développées au fil des années, telles que des caractéristiques de textures d'Haralick [Haralick et al., 1973; Ramesh et al., 2018] ou les caractéristiques visuelles invariantes à l'échelle (*Scale-Invariant Feature Transform* en anglais, ou SIFT) [Lowe, 2004; Lavania and Matey, 2014].
3. Calcul des vecteurs de caractéristiques : les caractéristiques sont calculées sur chaque image du jeu d'entraînement, qui sont à l'issue de cette étape représentées chacune par un vecteur de caractéristiques. Dans l'exemple de la figure 1.3, nous indiquons pour chaque caractéristique sa présence ou son absence dans l'image par une valeur binaire. Nous pouvons alors nous représenter visuellement les différentes images dans l' « espace des caractéristiques ». Dans cet espace au nombre de dimensions égal au nombre de caractéristiques calculées, chaque image est représentée par un point dont la position est donnée par son vecteur de caractéristiques.
4. Calcul de la frontière de décision : les vecteurs de caractéristiques des images du jeu d'entraînement sont présentées à un algorithme d'apprentissage avec les étiquettes associées. Le but attendu de cette étape est que le lien se fasse entre les valeurs des caractéristiques des images et les étiquettes qui leur sont associées. Visuellement, le but de l'algorithme est de trouver l'hyperplan qui sépare dans cet espace de caractéristiques les groupes d'objets avec la même étiquette, que l'on appelle la *frontière de décision* (ligne pointillée rouge en bas à droite de la figure 1.3). Si les caractéristiques sont bien choisies, que le jeu d'entraînement est assez divers et que la *capacité* de l'algorithme, c'est-à-dire la complexité de l'hyperplan qu'il peut définir, sont suffisants pour la classification considérée, alors l'algorithme convergera vers cet hyperplan [Goodfellow et al., 2016].

Une fois l'algorithme entraîné, il peut être utilisé pour réaliser des *prédictions*, c'est-à-dire des propositions d'étiquettes sur un jeu d'images qu'on ne lui a jamais présenté. Visuellement, cela correspond à placer un point dans l'espace des caractéristiques et de lui attribuer une étiquette en fonction de sa position par rapport à la frontière de décision.

L'exemple d'algorithme d'apprentissage automatique supervisé le plus simple est celui des « k plus proches voisins » [Cover and Hart, 1967]. On présente d'abord à l'algorithme les vecteurs de caractéristiques et les étiquettes du jeu d'entraînement. Puis, pour déterminer l'étiquette d'un nouvel élément, on calcule les k plus proches éléments parmi ceux du jeu d'entraînement. Plusieurs possibilités existent pour calculer cette distance, la plus populaire étant la distance euclidienne entre les vecteurs de caractéristiques. L'étiquette la plus représentée parmi ces plus proches voisins est attribuée à ce nouvel élément [3].

En pratique, lorsque l'on souhaite entraîner un algorithme de classification à partir d'un jeu d'images, il est rare que l'on ait à disposition dans le même temps les images non étiquetées pour lesquelles on souhaiterait que l'algorithme propose une prédiction. Pour évaluer les performances de l'algorithme, la procédure est alors de séparer le jeu d'entraînement, dont on connait les étiquettes, en plusieurs sous-ensembles que l'on appelle des *blocs* (*sets* en anglais). On conserve typiquement entre 60 et 80% des images comme bloc d'entraînement, et on utilise le reste comme bloc dit de *test*. Ce bloc ne sert pas à entraîner l'algorithme, mais à évaluer les performances de ce

3. La frontière de décision est ici calculée de façon locale, élément par élément lors de la phase de prédiction, plutôt que globalement comme dans la figure 1.3

dernier sur des images hors de son bloc d'entraînement. La performance obtenue sur le bloc de test agit donc comme un indicateur de la performance de l'algorithme en déploiement sur de nouvelles images.

De plus, pour beaucoup de ces algorithmes, il est nécessaire de fixer des *hyperparamètres*, c'est-à-dire des paramètres concernant le fonctionnement de l'algorithme lui-même. Par exemple, la valeur k et la métrique de distance sont les hyperparamètres de l'algorithme des « k plus proches voisins ». Pour fixer ces hyperparamètres, on a souvent recours à une autre division du jeu, que l'on appelle bloc de *validation*. L'idée est alors, pour chaque combinaison d'hyperparamètres que l'on souhaite tester, d'effectuer un entraînement sur le bloc d'entraînement, puis d'effectuer une prédiction sur le bloc de validation. La performance atteinte sur ce bloc est alors un indicateur de la qualité des hyperparamètres pour cette tâche. L'ensemble d'hyperparamètres menant à la performance de validation la plus élevée est alors conservé. Pour entraîner un algorithme d'apprentissage, il est donc très courant de diviser le jeu d'images disponible en trois blocs : entraînement, validation et test.

Principes de l'apprentissage profond

L'apprentissage profond est un paradigme d'apprentissage automatique inspiré de l'anatomie du cerveau humain [Hubel and Wiesel, 1959]. Cet apprentissage est associé à une structure algorithmique que l'on appelle un *réseau de neurones*. Le neurone biologique est une cellule complexe, mais seul son fonctionnement basique a servi d'inspiration au neurone informatique [Rosenblatt, 1957] (figure 1.4).

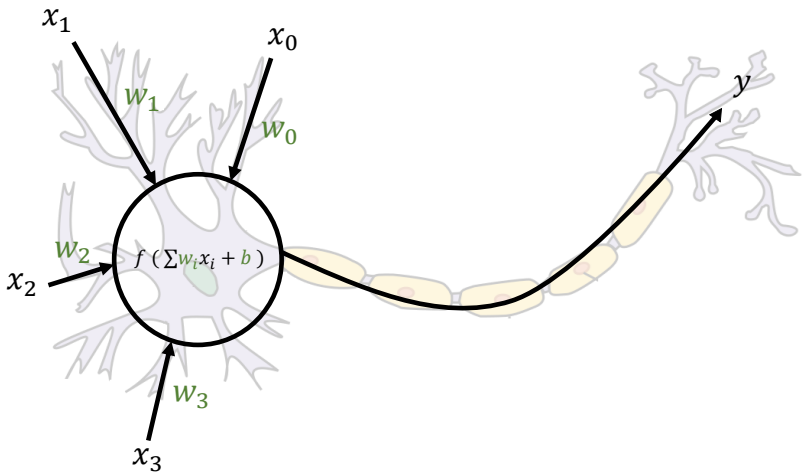


FIGURE 1.4 – Schéma d'un neurone informatique superposé à un schéma de neurone biologique. Les variables indiquées en vert sont les paramètres du neurone. Source de l'image du neurone biologique : https://simple.wikipedia.org/wiki/Neuron.

Un neurone biologique reçoit des signaux électriques d'autres neurones en amont via des points d'entrées. Ces signaux sont accumulés à l'intérieur du corps du neurone, et si leur somme dépasse un certain seuil, le neurone s'active et envoie à son tour un signal à des neurones en aval [Gerstner and Kistler, 2002]. Un neurone informatique modélise ces principes. Il accepte en entrée un nombre fixe de nombres réels (x_i dans la figure 1.4, représentant les signaux des neurones en amont) et produit en sortie une valeur réelle (y dans la figure 1.4, représentant le signal envoyé aux neurones en aval). Cette sortie est calculée à partir des entrées via l'équation

$$y = f(\sum_{i=0}^{n} w_i x_i + b) \tag{1.2}$$

qui représente l'accumulation de signaux électriques. Dans cette équation, les entrées x_i sont multipliées par des valeurs w_i que l'on appelle les *poids*, qui représentent la force de la connexion entre ce neurone et les neurones en amont. La fonction f est une fonction dite d'activation. Il s'agit d'une fonction non-linéaire croissante. Les premières implémentations de réseaux définissaient f comme une fonction seuil : f renvoyait 1 si son argument était strictement positif et 0 sinon, mais d'autres fonctions furent proposées au fil des années comme la fonction unité linéaire rectifiée (*Rectified Linear Unit* en anglais, ou ReLU) [Glorot et al., 2011]. Cette fonction représente l'activation du neurone si celui-ci a accumulé suffisamment de potentiel électrique. La valeur b, que l'on appelle le biais, représente l'appétence ou la résistance du neurone à s'activer. Les poids et le biais (souvent désignés collectivement sous le nom de « poids ») sont les paramètres du neurone : ce sont ces valeurs qui sont modifiées au cours d'un entraînement.

Un neurone informatique peut constituer à lui seul le support d'un algorithme d'apprentissage pour certaines tâches bien définies. En construisant un bloc d'entraînement composé de paires d'entrées réelles et des sorties binaires attendues, et en présentant séquentiellement ces exemples au neurone, il existe un algorithme d'entraînement qui définit comment modifier les poids de celui-ci afin de converger vers l'hyperplan attendu [Rosenblatt, 1957]. Cependant, la capacité d'un neurone unique est trop faible pour qu'il soit appliqué à d'autres cas que des cas « jouets ».

Les algorithmes d'apprentissage profond sont basés sur des ensembles de neurones interconnectés que l'on appelle des réseaux. On appelle *architecture* l'organisation selon laquelle les neurones sont reliés entre eux. Les premières architectures de réseaux s'appelaient les « perceptrons multi-couche » (*Multi-Layer Perceptron* en anglais, ou MLP). Dans un MLP, les neurones sont connectés à la fois parallèlement et séquentiellement selon une structure en *couches* (figure 1.5). La première couche est constituée d'un certain nombre de neurones qui prennent en entrée les données. Les sorties des neurones de cette couche servent d'entrée à une deuxième couche de neurones, et ainsi de suite, jusqu'à une dernière couche dont on identifie la sortie à la proposition faite par le réseau pour l'étiquetage de l'entrée. Cette structure en couches est inspirée de l'architecture neuronale du cerveau humain.

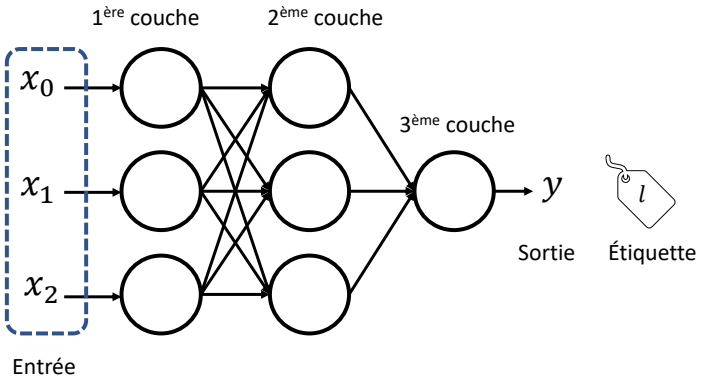


FIGURE 1.5 – Un MLP composé de trois couches. On identifie la sortie de la dernière couche y à la prédiction du réseau, que l'on compare à la véritable étiquette attendue l.

La phase d'entraînement d'un réseau se déroule de la façon suivante. Tous les paramètres sont initialisés aléatoirement. Un ensemble d'exemples issus du bloc d'entraînement, que l'on appelle un *lot*, est présenté au réseau. Le réseau calcule une sortie pour chacun de ces exemples, que l'on compare aux étiquettes. On en calcule une *fonction de coût* dont la valeur est corrélée à la distance entre les sorties et les étiquettes. Le but de l'entraînement est de réduire la valeur de cette fonction de coût, notée h, moyennée sur l'ensemble du bloc d'entraînement. Un algorithme que l'on appelle la « rétro-propagation d'erreur » [Rumelhart et al., 1986; LeCun et al., 1989] permet alors de calculer la contribution de chaque poids du réseau à la sortie proposée et donc à la fonction de coût pour cet

exemple. La connaissance de ces contributions permet de connaître la forme locale de la fonction de coût en fonction des valeurs des poids, c'est-à-dire la valeur du gradient

$$\frac{\partial h}{\partial w_i} \tag{1.3}$$

pour chaque poids w_i. Une itération d'un algorithme d'optimisation, tel que la descente de gradient, est alors appliquée pour modifier les valeurs des poids afin d'améliorer la performance du réseau sur ce lot d'exemples. Un nouveau lot d'exemples est présenté et le cycle recommence. L'entraînement se poursuit jusqu'à ce que la performance se stabilise ou selon d'autres critères d'arrêt. On dit alors que le réseau a *convergé* ou bien est entraîné. Le réseau peut ensuite être utilisé en prédiction pour produire des étiquettes sur des données qu'on ne lui a jamais présentées.

Avantages et inconvénients par rapport à l'apprentissage automatique traditionnel

Par rapport à d'autres structures algorithmiques, les réseaux de neurones ont deux grands avantages. Premièrement, leur structure basée sur l'empilement de fonctions non-linéaires leur octroie une énorme capacité. Il a été montré qu'un réseau avec deux couches seulement peut représenter n'importe quelle fonction mathématique liant des entrées et une sortie [Hornik et al., 1989]. Pour des raisons d'entraînement et de capacité de calcul, les architectures modernes comprennent toutefois un plus grand nombre de couches. Deuxièmement, les réseaux agissent à la fois comme algorithmes de classification et comme extracteurs de caractéristiques. En effet, ce sont, dans la grande majorité des cas, les données elles-mêmes (les valeurs des pixels dans le cas des images) qui sont fournies en entrée à un réseau et non des caractéristiques pré-définies (comme dans l'exemple de la figure 1.3). Les algorithmes de rétro-propagation et d'optimisation permettent d'exploiter la grande capacité du réseau en guidant les changements des poids des réseaux vers un ensemble des valeurs qui mène à une bonne performance de classification : en d'autres termes, les caractéristiques sont apprises spécifiquement pour la tâche considérée. Les caractéristiques extraites, simples si nous les considérons dans les premières couches, se complexifient peu à peu au fil des opérations faites par les couches. C'est cette dimension de couches empilées, qui permettent la création de caractéristiques complexes, à laquelle on fait référence lorsqu'on parle d'apprentissage « profond ». Cet empilement mène à des caractéristiques qui peuvent être extrêmement raffinées et subtiles, bien plus que ce qu'un humain ou un algorithme de calcul de caractéristiques classique peuvent réaliser.

La grande capacité des réseaux peut entraîner cependant un grand inconvénient : le *surapprentissage* (*overfit* en anglais) [Goodfellow et al., 2016]. Pour l'illustrer, plaçons-nous dans le cadre d'une application d'apprentissage automatique simple : la régression non-linéaire (figure 1.6). Considérons que nous avons n points d'entraînement qui sont des paires de nombres réels (x_i, y_i) et que nous souhaitons ajuster un polynôme de degré m à ces points, c'est-à-dire trouver les coefficients qui permettent au polynôme de passer au plus près de ces points. Si $m \geq n$, il existe forcément un polynôme qui soit exactement ajusté à ces données : un tel modèle a assez de capacité pour représenter complètement le jeu de données. L'optimisation des coefficients du polynôme mène donc à une fonction qui passe exactement par tous les points d'entraînement. Cette situation, alléchante sur le papier, est en fait souvent très fâcheuse. En effet, il faut se souvenir que, comme pour toute procédure de statistique inférentielle, le bloc d'entraînement ne constitue qu'un échantillon de la distribution que l'on cherche à ajuster. En conséquence, les points d'entraînement représentent la fonction génératrice des données mais avec un certain bruit. En optant pour un polynôme de degré élevé, l'ajustement se fera sur ce bruit plutôt que sur la véritable distribution. Par exemple, dans la figure 1.6, les dix points semblent provenir d'une distribution qui pourrait être ajustée par un polynôme de degré deux (courbe verte). L'utilisation d'un polynôme de degré dix (courbe marron) mène à un ajustement plus précis des points d'entraînement, mais nous constatons aisément que la capacité « excédentaire » du modèle a servi à ajuster le bruit. Une telle configuration d'entraînement mène souvent en outre à un modèle ajusté très éloigné de la fonction génératrice dans les plages

où il n'y a pas suffisamment de données d'entraînement (figure 1.6, tout à droite par exemple). Lors d'une phase de prédiction d'une nouvelle donnée, l'ajustement proposé risque de ne pas être satisfaisant. On parle alors de surapprentissage : l'algorithme a appris « par cœur » les données d'entraînement sans en tirer la « substantifique moelle » [Rabelais, 1534].

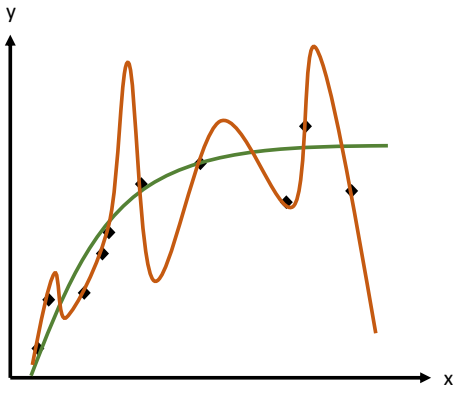


FIGURE 1.6 – Illustration sur un cas de régression non-linéaire du phénomène de surapprentissage. Les carrés noirs représentent les points d'entraînement. La courbe verte représente l'ajustement d'un polynôme au degré adéquat par rapport à la fonction génératrice du jeu de données, et la courbe marron celui d'un polynôme au degré trop important qui mène à un surapprentissage.

Les réseaux de neurones doivent faire face à cette difficulté. En effet, le nombre de poids d'un réseau est dans de nombreux cas bien supérieur au nombre de paramètres de l'hyperplan « optimal » permettant de séparer les données d'entrée selon leurs étiquettes. Des méthodes dites de *régularisation* sont alors implémentées pour combattre ce surapprentissage. Dans le cas « jouet » de régression polynomiale, la solution est de choisir un modèle de degré moindre, adapté au degré de la fonction génératrice. Dans le cas des réseaux de neurones, bien que cette idée fasse partie des méta-méthodes pour limiter le surapprentissage et soit même la clé de voûte de certaines architectures neuronales [Iandola et al., 2016; Tan and Le, 2019], nous souhaitons en général conserver la grande capacité de ces réseaux car c'est elle qui permet de générer des caractéristiques variées et sophistiquées. La solution pour obtenir un modèle utile est alors d'augmenter au maximum le nombre de données afin de contraindre la frontière de décision et d'éviter les zones « vides » dans lesquelles les prédictions du modèle risquent d'être aberrantes. La figure 1.7 illustre cette idée dans le cadre de la régression non-linéaire.

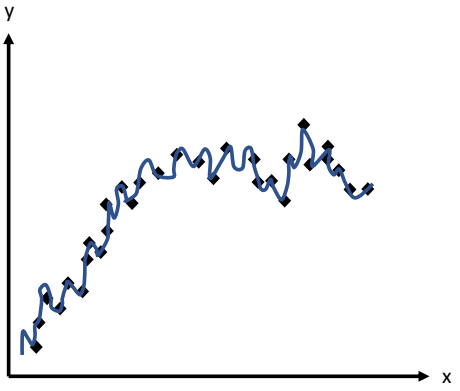


FIGURE 1.7 – Illustration sur un cas de régression non-linéaire de la stratégie d'obtenir un nombre important de données pour contraindre fortement un polynôme de degré élevé.

Les algorithmes d'apprentissage profond nécessitent donc pour donner des performances convenables un nombre de données extrêmement important [Warden, 2017]. De plus, il est néces-

saire que ces données soient annotées, c'est-à-dire qu'elles aient reçu une étiquette. En fonction de l'application, l'annotation peut être extrêmement chronophage ou coûteuse. Ceci est particulièrement vrai dans des domaines où cette annotation doit être réalisée par des expert ·e ·s tels que l'agronomie ou la médecine, et/ou pour des tâches plus fines telles que la segmentation.

Réseaux de neurones convolutifs pour la vision par ordinateur

Dans les applications où les données à traiter sont des signaux bidimensionnels tels que des images, une autre catégorie de réseaux est régulièrement employée : les réseaux de neurones *convolutifs* (*Convolutional Neural Networks* en anglais, ou CNN) [LeCun et al., 1989] (figure 1.8). Une image de dimension $d_1 \times d_2$ pixels est considérée comme une entrée de $d_1 \times d_2$ éléments. Plusieurs changements structuraux et conceptuels sont implémentés par rapport aux MLP. Ces changements sont motivés par la volonté de réduire le nombre de paramètres du réseau et transposer des concepts d'apprentissage automatique « classique » sur images (filtres, cascades, etc.) au domaine de l'apprentissage profond.

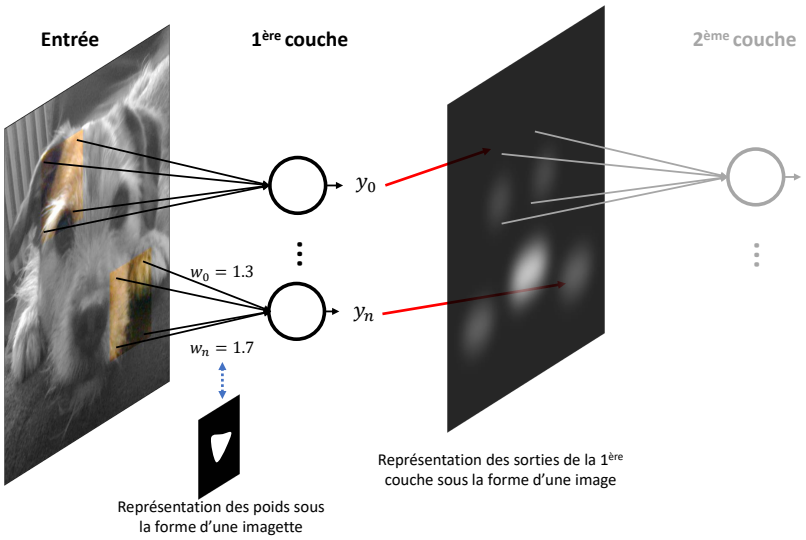


FIGURE 1.8 – Schéma de l'action des neurones dans un CNN. Source de l'image du chien : photo personnelle.

Premièrement, chaque neurone n'est connecté qu'à une portion des sorties des couches précédentes (zones colorées de l'image de la figure 1.8). On dit que les neurones ont un *champ de vision*. Un neurone de la première couche prend donc en entrée une petite portion de l'image centrée autour d'un pixel, que l'on appelle une imagette. Nous pouvons nous représenter l'ensemble des sorties des neurones de la première couche comme une nouvelle image, puisque chaque sortie a été calculée sur une imagette centrée sur une position spatiale dans l'image d'origine. Ainsi, l'analogie d'une entrée structurée en image peut être poursuivie pour les couches suivantes (image à droite de la figure 1.8). De plus, puisque chaque poids d'un neurone de la première couche est associé à un pixel de l'image d'entrée, nous pouvons nous représenter l'ensemble des poids d'un neurone par une imagette également, d'une taille égale au champ de vision du neurone (imagette en bas de la figure 1.8). On appelle alors l'ensemble des poids du neurone un *noyau*. Un neurone envoie un signal en sortie d'autant plus élevé que la somme des entrées pondérées par les poids associés est forte. Si nous représentons à la fois les poids et les entrées sous la forme d'images, alors nous pouvons nous représenter qu'un neurone produit une valeur forte en sortie si l'imagette qu'il « voit » correspond à son noyau.

Une contrainte supplémentaire des CNNs est qu'au sein d'une couche, tous les poids sont les mêmes d'un neurone à l'autre. Ainsi, la même imagette est recherchée dans toute l'image d'entrée à différentes positions, et la sortie de la couche correspond à une carte de chaleur pour la détection de

cette imagette. On peut dire par analogie avec le champ de la vision par ordinateur qu'une couche de neurones fonctionne à la manière d'un filtre appliqué à une image pour en détecter un motif (par exemple, le filtre de Sobel permet de détecter les bords d'une image [Sobel, 2014]). Dans la figure 1.8, la première couche recherche dans l'image d'entrée une forme arrondie (imagette du bas), et en conséquence les neurones qui s'activent sont ceux connectés à des zones qui correspondent à cette description, telles que la truffe, les pattes et les yeux du chien.

À la différence d'une opération de recherche de bords qui est réalisée avec un seul filtre, les CNNs correspondent à plusieurs filtres appliqués séquentiellement, c'est-à-dire que les filtres d'une couche donnée s'applique sur la sortie de la couche précédente, menant à un filtrage de plus en plus complexe, et donc à des caractéristiques recherchées de plus en plus élaborées [Yosinski et al., 2015]. Par exemple, la couche de neurones qui prend en entrée la sortie de la première couche de la figure 1.8 applique un filtre non pas sur les valeurs brutes des pixels de l'image mais sur des valeurs représentant la présence de formes arrondies. De plus, un CNN comprend une dimension de plus qu'un MLP : chaque couche d'un CNN est en fait constituée d'un certain nombre de filtres. On appelle cet ensemble de filtres une *couche convolutive*. L'image en sortie d'une couche de CNN est donc une image constituée d'autant de canaux que la couche comprend de filtres, et sert toute entière comme entrée à la couche suivante. On appelle une telle image tridimensionnelle une *carte de caractéristiques* (figure 1.9).

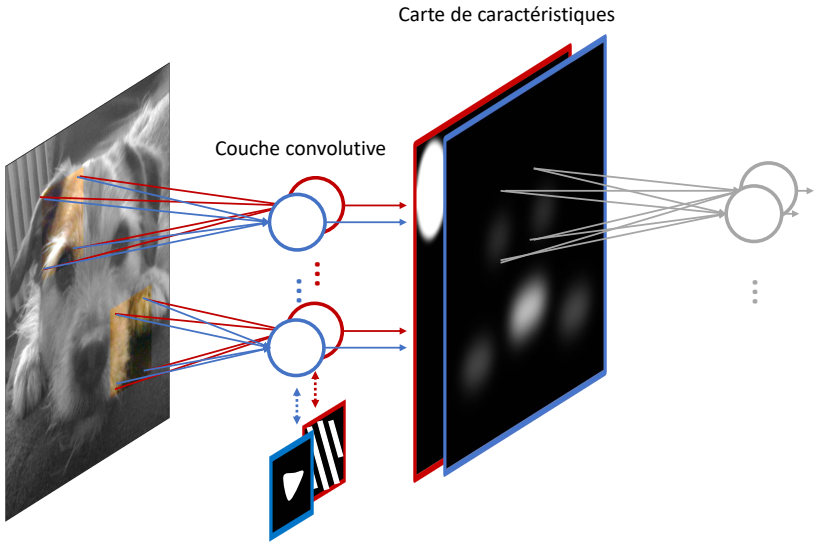


FIGURE 1.9 – Extension de la figure 1.8 pour un cas où la couche convolutive contient deux filtres.

Pour compléter le tableau d'un CNN, il est nécessaire de préciser qu'ils sont composés d'autres couches que les couches convolutives :

— des couches de « regroupement par maximum » (*max-pool* en anglais) sont insérées entre certaines couches convolutives. Cette couche réduit spatialement la taille de la carte de caractéristiques produite par la couche précédente, la plupart du temps par un facteur deux. Cette action permet mécaniquement d'augmenter le champ de vision des neurones de la couche convolutive suivante, puisque leur noyau s'applique à une imagette correspondant à une zone plus grande de l'image d'origine. Ainsi, à mesure que les caractéristiques extraites se complexifient, ces dernières sont calculées sur des sections de plus en plus importantes de l'image d'origine. Ces couches *max-pool* permettent une évolution harmonieuse entre un calcul de caractéristiques simples (bords, couleurs, etc.) sur des zones spatialement réduites à des caractéristiques complexes (forme d'objet plus large, etc.) sur des zones plus étendues.

— une couche de « regroupement par moyennage global » (*Global Average Pooling* en anglais, ou GAP) est parfois présente après toutes les couches convolutives et *max-pool*. Cette couche

réduit spatialement la taille de la carte de caractéristiques à une résolution spatiale fixe. Cette couche permet à l'utilisateur de fournir des images de tailles variables en entrée du réseau sans modifier l'architecture de ce dernier.

— des couches de MLP, donc composées de neurones qui prennent en entrée l'intégralité de l'image, sont présentes en général après toutes les couches convolutives et *max-pool* et l'éventuelle couche GAP. On les appelle dans le cadre d'un CNN des couches entièrement connectées (*Fully Connected* en anglais, ou FC). Puisque les neurones de ces couches sont connectés à la totalité des cartes de caractéristiques données en entrée, les caractéristiques provenant de l'ensemble de l'image sont ainsi regroupées pour permettre au réseau de proposer une étiquette en se basant sur la totalité de l'information disponible.

— enfin, après les couches FC, une couche de « fonction exponentielle normalisée » (*softmax*) applique une opération de normalisation aux sorties brutes du réseau afin que la somme des scores proposés pour les classes soit égale à un.

La figure 1.10 présente l'agencement de ces différentes couches dans un CNN simple.

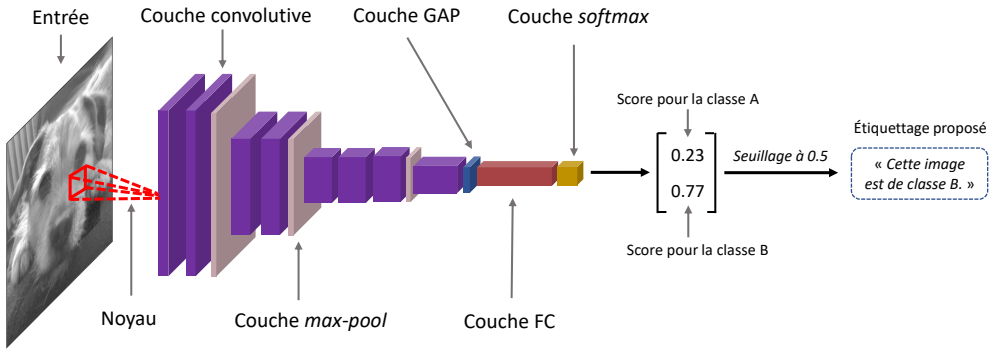


FIGURE 1.10 – Schéma d'un CNN simple. Nous conserverons le même code couleur pour toutes les figures présentant des réseaux de neurones dans ce manuscrit.

Les méthodes d'apprentissage profond ont révolutionné le champ de l'apprentissage automatique au début des années 2010 [Krizhevsky et al., 2012], permettant d'obtenir des progrès significatifs sur des tâches de classification considérées jusqu'alors comme extrêmement exigeantes [Alom et al., 2018]. Les progrès ont été particulièrement stupéfiants dans les domaines de vision par ordinateur [He et al., 2017; Zhu et al., 2017] et du traitement automatique du langage naturel [Vaswani et al., 2017; Brown et al., 2020]. Malgré les difficultés importantes du domaine, et notamment le besoin d'un très grand nombre de données afin de limiter le surapprentissage, les gains de performance possibles sont si substantiels par rapport aux méthodes d'apprentissage classiques que de nombreux acteurs de la recherche et de l'industrie, dont Carbon Bee, cherchent à implémenter prioritairement ces méthodes lors de l'étude d'un nouveau problème.

1.2.2 Spécificités du domaine de la détection de maladies de plantes

Nous discutons à présent les spécificités du domaine d'intérêt relativement à l'application des méthodes d'apprentissage profond.

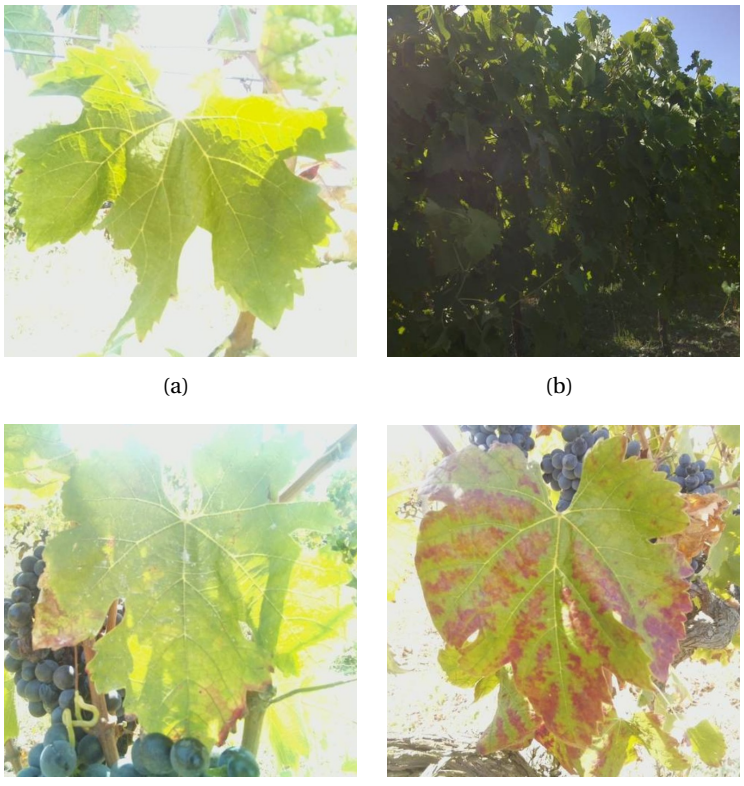
Les jeux de données sont restreints mais comportent une grande variabilité

Les techniques d'apprentissage profond ont été diffusées dans le domaine des sciences végétales comme dans de nombreux autres domaines à partir du milieu des années 2010. La détection de maladies d'une plante s'effectue principalement en analysant des images de ses feuilles [Singh et al., 2018]. Cependant, cette tâche présente plusieurs difficultés pour l'application des méthodes d'apprentissage profond. Tout d'abord, les jeux de données contiennent en général peu d'images. Ceci est dû en partie à la grande variabilité des plantes et des types de maladies, ce qui nécessite de

circonscrire strictement le cas d'étude [Ubbens et al., 2018]. De plus, le processus d'annotation doit très souvent être confié à un·e expert·e agronome du domaine. En effet, les symptômes visuels de la maladie recherchée sont souvent très proches de ceux d'autres maladies voire de ceux indiquant des mécanismes physiologiques sains au sein de la plante [Daughtry et al., 2000]. En conséquence, les jeux de données annotés disponibles sont restreints par rapport à ceux d'autres domaines comme la conduite autonome [Saleem et al., 2019].

Une autre difficulté est que la variabilité des symptômes possibles même pour un système plante-pathogène défini complique la tâche de classification. Les plantes et les agents pathogènes sont des organismes vivants, et leur signature visuelle évolue donc dans le temps. Au sein d'un même jeu voire d'une même image, plusieurs plantes peuvent être atteintes d'une même maladie présente à différent stades, parfois avec des signatures visuelles très différentes. De plus, les acquisitions des images sont très souvent réalisées à l'extérieur ou en serre : les conditions météorologiques et d'illumination mènent à une grande variabilité dans les jeux disponibles. La figure 1.11 illustre quelques-unes de ces variabilités.

Parmi les applications d'apprentissage profond, plus du tiers [Abade et al., 2020] des études publiées aujourd'hui utilisent au moins partiellement le jeu annoté PlantVillage [Hughes et al., 2015] pour mener à bien leur entraînement. Il s'agit d'un jeu libre d'environ 50 000 images RVB de feuilles atteintes de maladies. La tâche de classification associée est de déterminer la maladie dont sont atteintes les feuilles parmi les 38 présentes dans le jeu. L'importante proportion d'études s'appuyant sur ce jeu souligne le problème de l'étroitesse des jeux qui handicape l'application de techniques d'apprentissage profond au domaine.



(a) (b)

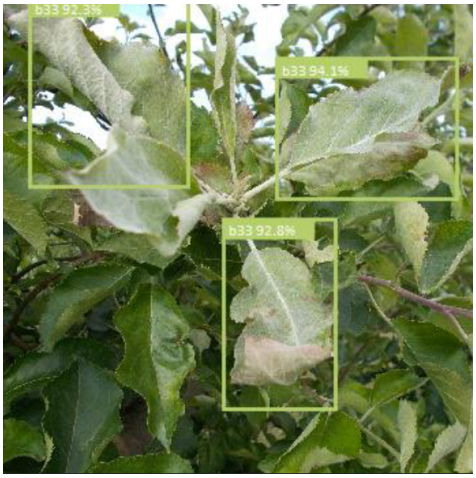
(c) (d)

FIGURE 1.11 – Quelques exemples d'une campagne d'acquisition d'images de vignes atteintes d'esca, acquises par Carbon Bee AgTech. Les sous-figures (a) et (b) illustrent la différence de luminosité qui peut exister entre des acquisitions en fonction de la position du soleil par rapport au capteur. Les sous-figures (c) et (d) illustrent les différences de signature de la maladie (taches brunes dans le cas de l'esca) en fonction de son stade de développement au moment de l'acquisition.

Les tâches étudiées concernent souvent une classification en conditions contrôlées

Par ailleurs, il est nécessaire de souligner que, plus encore que dans d'autres domaines, les jeux de données sont pour beaucoup en conditions dites *contrôlées*. Dans ces jeux, les acquisitions concernent des feuilles isolées sur un fond uni prises en intérieur avec une illumination homogène. Or les *conditions réelles* d'application de ces algorithmes, que ce soit en serre ou en champ, sont bien différentes : nombreuses feuilles superposées à distance variable, positionnement avec des angles arbitraires par rapport à la caméra, conditions lumineuses diverses et potentiellement variées au sein d'une même feuille, etc. La figure 1.12 illustre la différence entre les deux types de conditions.

(a) Une image, en conditions contrôlées, du jeu de données PlantVillage. Source : https://www.kaggle.com/emmarex/plantdisease.



(b) Une image, en conditions réelles, du jeu de données PlantDisease. Source : [Arsenovic et al., 2019].

FIGURE 1.12 – Comparaison entre un jeu en conditions contrôlées et un jeu en conditions réelles.

Par conséquent, les apprentissages réalisés sur des jeux en conditions contrôlées masquent une grande partie des difficultés du domaine lorsque le bloc de test est aussi acquis dans ces conditions. Les performances obtenues surestiment la capacité des modèles à effectuer leur tâche en conditions réelles [Ferentinos, 2018]. Les quelques études s'étant intéressées à cet écart de performance rapportent une baisse d'environ 30% de leur métrique de performance lorsque ces modèles sont utilisés en prédiction en conditions réelles [Mohanty et al., 2016]. PlantVillage est un exemple de jeu en condition contrôlées, et plus largement il est estimé que les deux tiers des études publiées à ce jour travaillent uniquement en conditions contrôlées [Abade et al., 2020]. Certain ·es chercheur ·euses, conscients de ces difficultés, cherchent à promouvoir l'utilisation de jeux comprenant au moins une certaine proportion d'images en conditions réelles. C'est le cas du jeu de données PlantDisease [Arsenovic et al., 2019], qui contient près de 80 000 images représentant 42 maladies, acquises en conditions réelles, pouvant notamment contenir plusieurs feuilles atteintes dans la même image.

Les architectures des réseaux employés sont empruntées à d'autres domaines

Malgré ces limitations, de nombreuses études ont porté sur l'application d'algorithmes d'apprentissage profond à des problèmes de détection de maladies [Singh et al., 2018]. Les plantes les plus étudiées sont les céréales (riz, maïs, blé, etc.) et les maladies les plus étudiées sont celles causées par des champignons [Abade et al., 2020]. Beaucoup de ces études consistaient simplement à appliquer un réseau de neurones constituant l'état de l'art au moment de l'étude au problème considéré [Kaur et al., 2019]. Par ailleurs, des réseaux dont l'architecture a été personnalisée pour la tâche considérée ont aussi été développés, en modifiant notamment le nombre de couches ou d'autres hyperparamètres architecturaux [Abade et al., 2020]. Certains réseaux ont des architectures

plus spécifiques encore au domaine, à l'image de PlantDiseaseNet [Arsenovic et al., 2019]. Cette architecture comprend un premier réseau servant à la détection d'instances de feuilles dans l'image suivi d'un deuxième dédié à la détection de maladie sur feuille isolée. L'immense majorité des études se cantonne au cas d'une classification binaire (présence ou absence de maladie) ou multiple (identification de la maladie), alors que des tâches plus complexes comme la segmentation avaient été explorées dans ce domaine par le biais de méthodes d'apprentissage classiques [Arivazhagan et al., 2013]. Or, une connaissance des positions et des tailles des lésions via la segmentation permettrait de proposer un diagnostic plus fin quant au stade d'évolution de la maladie. Conscients de l'enjeu d'une telle information, de nombreux ·ses chercheur ·euses ont proposés des approches pour extraire des résultats de segmentation à partir de réseaux destinés à la classification, telles qu'une analyse de saillance ou une visualisation des cartes de caractéristiques du réseau [Saleem et al., 2019].

1.2.3 Positionnement

Dans ce travail, nous avons implémenté des méthodes d'apprentissage profond pour mener à bien des classifications liées à notre cas d'étude. Nous avons employé pour la plupart des travaux présentés dans ce manuscrit une architecture existante, relativement simple et aux performances attestées dans le domaine de la vision par ordinateur ; mais nous avons aussi contribué aux méthodes en elles-mêmes en proposant une nouvelle architecture dédiée aux images du CTIS (chapitre 4). Nous nous démarquons significativement de l'état de l'art en ce qui concerne les jeux de données, puisque nous nous concentrons sur des imageries hors du domaine du visible, largement moins étudiées que les images RVB dans le cadre de la détection de maladies végétales. Nous menons par ailleurs une étude en conditions réelles, proche des contraintes industrielles rencontrées dans le domaine de l'agriculture de précision.

1.3 Tavelure du pommier

Enfin, nous présentons la maladie cas d'étude choisie ainsi que les différents travaux déjà menés pour sa détection.

1.3.1 Une infection difficile à détecter et à soigner

La tavelure du pommier est une maladie fongique causée par le champignon *Venturia inaequalis* affectant les arbres du genre *Malus*, qui regroupe les pommiers que nous pouvons trouver en Europe. Le champignon cause l'apparition de taches brunâtres et de zones nécrosées sur les feuilles et les fruits du pommier. Ces lésions ne compromettent pas directement la comestibilité du fruit mais favorisent l'installation de parasites secondaires qui peuvent, eux, provoquer le pourrissement du fruit. Même sans infection secondaire, les lésions sont suffisamment intrusives (déformations, flétrissures, crevasses) pour que les fruits produits perdent toute valeur marchande. À ce titre, il s'agit de la maladie affectant la pomme la plus grave en termes de coût économique au niveau mondial [Bowen et al., 2011].

Le cycle de reproduction de *V. inaequalis* est annuel. Les feuilles infectées au printemps tombent de l'arbre en automne, créant au pied de celui-ci un humus propice au développement du champignon. Au printemps suivant, des cellules reproductrices du champignon que l'on nomme ascospores sont relâchées dans l'air et infectent les feuilles à proximité. Cette étape intervient sous des conditions d'humidité spécifiques, ce qui rend difficile la prédiction de cette période et donc la bonne application d'un traitement phytosanitaire [Cuthbertson and Murchie, 2003]. L'attaque de la feuille par le parasite est représentée figure 1.13.

L'ascospore pénètre dans la couche protectrice externe de la feuille, appelée cuticule. Cette pénétration s'accompagne de signaux chimiques qui inhibent les mécanismes de défense de la

FIGURE 1.13 – Étapes d'infection d'une feuille de pommier par *V. inaequalis*. La feuille est représentée en vert et le champignon en marron. L'indication *JX* correspond au nombre de jours après l'infection. Source : image adaptée de Bowen et al. [2011].

feuille. Le champignon se développe alors en dessous de la cuticule, juste au-dessus de l'épiderme. La maladie est particulière en ceci que le champignon ne détruit pas les cellules de la couche épidermale. La souche croît ainsi sous la surface de la feuille en colonisant progressivement des zones autour du foyer d'infection pendant plusieurs jours, et est pendant ce temps quasiment invisible à l'œil nu [Oerke et al., 2011]. La sporulation survient autour du huitième jour : de nouvelles spores nommées conidies éclosent et percent la cuticule. Les spores se répandent alors vers d'autres zones de la feuille, voire vers d'autres feuilles. Même si l'infection est invisible à l'œil nu pendant la période « dormante », la présence du parasite est cause d'effets physiologiques importants dans ses zones d'implantation. En particulier, les micro-lésions de la cuticule réduisent la capacité de la feuille à retenir l'eau et causent une transpiration involontaire. Ceci mène à un refroidissement localisé des zones où le champignon se développe [Oerke et al., 2011].

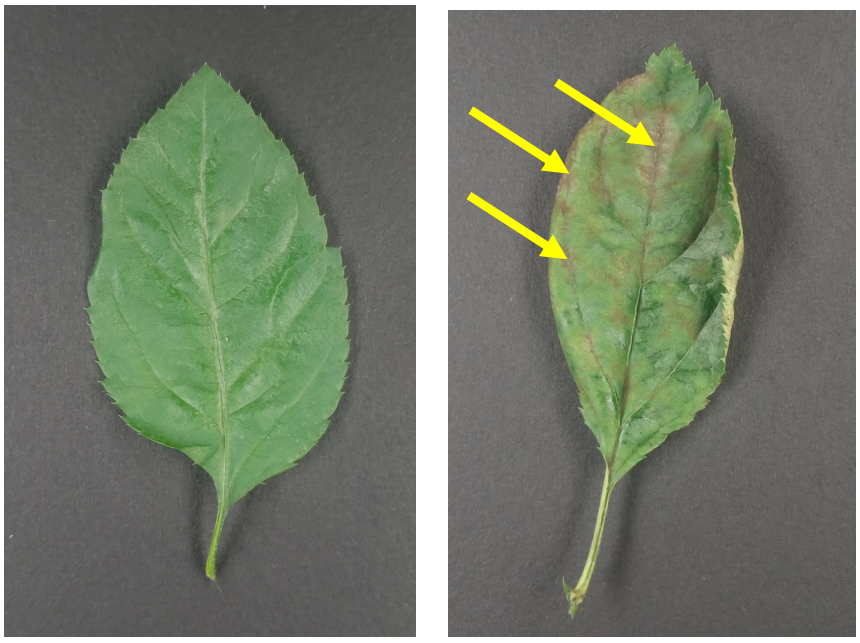
1.3.2 De nombreuses imageries ont été employées pour sa détection

Différents types d'imagerie sont pertinents pour détecter la tavelure. Une fois que la sporulation sur la feuille a commencé et que la lésion est visible à l'œil nu, des méthodes classiques basées sur l'imagerie RVB sont tout à fait adaptées. En effet, les lésions de tavelure ont une couleur très distincte des régions saines de la feuille : jaunâtre au début de l'infection, puis tirant progressivement vers le noir au fur et à mesure que les zones se nécrosent. La figure 1.14 présente une comparaison entre une feuille saine et une feuille *tavelée*, c'est-à-dire atteinte de tavelure, qui met en évidence la visibilité des lésions à l'œil nu.

Cependant, une infection ayant atteint cette phase détectable dans la lumière visible est déjà très avancée et difficile à traiter. Il est, de plus, très probable que les lésions détectées aient à ce moment là déjà causé l'infection, pour l'instant invisible, d'autres zones de la feuille. Pour des besoins de traitement automatique de la maladie, la détection doit se faire de façon plus précoce. Par conséquent, différents types d'imagerie ont été testés dans le but de réaliser une détection avant les symptômes visibles.

— l'IR proche : les auteurs de [Benoit et al., 2016] ont procédé à des acquisitions d'images de feuilles tavelées avec une caméra hyperspectrale à balayage sur la gamme [400-1000] nm. En suivant une méthode basée sur la théorie de l'information, ils ont déterminé automatiquement la longueur d'onde offrant le plus de contraste dans cette gamme : 760 nm. En nous basant sur les études réalisées sur d'autres parasites [Mahlein, 2016], il est possible que que ce contraste soit dû à la destruction des structures internes de la feuille par *V. inaequalis*.

— La thermographie : il a été montré que les lésions de tavelure causaient, par l'augmentation de la transpiration, des zones localement plus froides, visibles par thermographie jusqu'à quatre jours avant l'apparition de symptômes dans le visible [Oerke et al., 2011; Belin et al., 2013]. En raison de son cycle de vie en deux temps, dont le premier cause une chute locale des températures et le deuxième la nécrose des cellules infectées, la tavelure est une maladie « modèle » de l'étude thermographique [Bowen et al., 2011]. En effet, la chute de température est causée par une seul phénomène isolé dans le temps, contrairement à d'autres interactions



(a) Feuille saine.

(b) Feuille atteinte de tavelure, à J14. Les flèches indiquent la position de quelques unes des lésions.

FIGURE 1.14 – Comparaison en imagerie RVB entre une feuille saine et une feuille tavelée. Source : acquisitions réalisées au LARIS.

plante-pathogène où la concomitance de la mort des cellules et l'augmentation de la transpiration provoque des cycles d'augmentation et de diminution de la température difficilement interprétables.

— L'imagerie hyperspectrale : les auteurs de [Delalieux et al., 2009a] se sont intéressés à la détermination d'indices indicateurs de la présence de tavelure à partir d'acquisitions hyperspectrales dans le domaine visible jusqu'à l'IR-B pour des cultivars. Ils concluent sur la difficulté à déterminer des indices robustes au vieillissement de la feuille et aux différents stades de la maladie. Par exemple, certains indices basés sur l'absorption d'eau permettent de détecter une infection de tavelure dès le jour de l'infection, mais sont inutiles quelques jours plus tard. Les auteurs sont parvenus à déterminer des indices relativement robustes au temps, mais dont la qualité est cependant dépendante du cultivar.

— La fluorescence : une part de l'énergie lumineuse absorbée par la feuille est renvoyée sous la forme d'une onde électromagnétique à plus basse énergie que l'on appelle fluorescence. L'énergie solaire absorbée par la plante est partiellement consommée par la photosynthèse, mais également renvoyée par fluorescence et sous forme thermique. La mesure de fluorescence agit donc comme un indicateur pour mesurer l'énergie absorbée dédiée à la photosynthèse, indicateur de la bonne santé d'une plante. Dans le cas de la tavelure, cette imagerie n'a cependant pas montré une capacité de détection avant l'apparition de symptômes visibles [Belin et al., 2013].

1.3.3 Positionnement

Ainsi, la tavelure a été étudiée via une grande variété d'imageries, et des longueurs d'onde optimales pour la détection des lésions ont été déterminées pour plusieurs gammes spectrales. Cependant, cette caractérisation a nécessité l'usage de caméras hyperspectrales coûteuses qui permettaient une acquisition d'un grand nombre de longueurs d'onde. Dans ce travail, nous proposons d'étudier l'apport potentiel d'une imagerie bas-coût et rapide et qui ne nécessite pas de connaissance biologique *a priori* sur les longueurs d'onde optimales pour mener à bien la détec-

tion de lésions de tavelure. Nous exploitons la connaissance de la littérature pour positionner la performance de cette imagerie par rapport à celles déterminées comme optimales antérieurement.

1.4 Conclusion

L'imagerie invisible est utilisée depuis de nombreuses années à des fins de détection de maladies végétales. Les possibilités d'acquisition qu'elle permet ont été étendues ces dernières années via le développement de capteurs hyperspectraux. De plus, son déploiement a été accéléré par l'avènement des algorithmes d'apprentissage profond qui ont permis d'exploiter efficacement ces informations spectrales riches. Dans ce travail, nous nous inscrivons dans la continuité de ces deux axes porteurs dans le domaine de la détection de maladies végétales par imagerie. Cependant, contrairement aux paradigmes d'« expansion » que l'on retrouve à la fois dans le domaine de l'imagerie hyperspectrale via le développement de spectromètres onéreux à forte résolution spatiale comme spectrale, et dans le domaine de l'apprentissage automatique via le développement de réseaux de neurones toujours plus profonds et difficiles à entraîner, nous explorons dans ce travail la viabilité d'une approche à bas coût matériellement comme algorithmiquement. En particulier, nous présentons dans les chapitres suivants une étude menée sur l'exploitation optimale du signal d'un spectromètre à résolution spatio-spectrale réduite et à acquisition rapide : le CTIS.

Chapitre 2

Le CTIS : un capteur hyperspectral atypique à évaluer

L'innovation majeure de la caméra Carbon Bee est l'utilisation du capteur CTIS, encore méconnu dans la communauté d'imagerie hyperspectrale. Dans ce chapitre, nous commençons par positionner ce spectromètre par rapport aux autres capteurs hyperspectraux, en décrivant les principes de l'imagerie spectrale et les différents types de capteurs. Nous soulignons les particularités du CTIS par rapport aux capteurs existants : une capacité d'acquisition instantanée, une acquisition indirecte, et un coût réduit. Par la suite, nous décrivons en profondeur le CTIS en lui-même. Nous nous attardons particulièrement sur le lien fort que ce spectromètre partage avec le champ de la tomographie.

Sommaire

2.1 Principes de l'imagerie spectrale 28
2.1.1 Notions préliminaires 28
2.1.2 Capteurs hyperspectraux 29
2.2 Le CTIS : un capteur du domaine de l'imagerie computationnelle 33
2.2.1 L'acquisition d'une image destinée à une reconstruction 33
2.2.2 Un capteur relativement peu étudié 34
2.2.3 Un lien fort avec la tomographie 35
2.2.4 La reconstruction du signal CTIS est épineuse et approximative 39
2.3 Conclusion 42

2.1 Principes de l'imagerie spectrale

Nous présentons tout d'abord quelques éléments théoriques relatifs aux capteurs optiques. Par la suite, nous nous intéressons de façon plus approfondie aux capteurs hyperspectraux.

2.1.1 Notions préliminaires

Nomenclature d'un capteur

Un capteur est un composant électronique qui transforme une grandeur physique en une mesure quantifiée. Dans ce travail, nous nous intéressons aux capteurs de lumière, ou *imageurs*, qui convertissent un nombre de photons en une mesure. Nous nous concentrons en particulier sur les capteurs lumineux basés sur des circuits intégrés nommés « dispositifs à transfert de charge » (*Charge-Coupled Device* en anglais, ou CCD). Ces capteurs sont composés d'une matrice de détecteurs lumineux associés à un système électronique qui convertit pour chacun de ces détecteurs l'information lumineuse reçue en tension électrique, exprimée en volts, puis en un nombre flottant ou en entier. L'ensemble des valeurs acquises par ces détecteurs constitue l'image obtenue en sortie. Ces capteurs lumineux sont généralement accompagnés de *filtres* optiques. Il s'agit de matériaux fins qui ne transmettent que certaines gammes de longueur d'onde. Ainsi, un capteur dit IR est un capteur dont les filtres transmettent la lumière de la gamme IR et absorbent le reste. La figure 2.1 illustre le vocabulaire présenté dans cette section. Dans ce manuscrit, nous utiliserons le terme « capteur » pour désigner l'ensemble des éléments de cette figure.

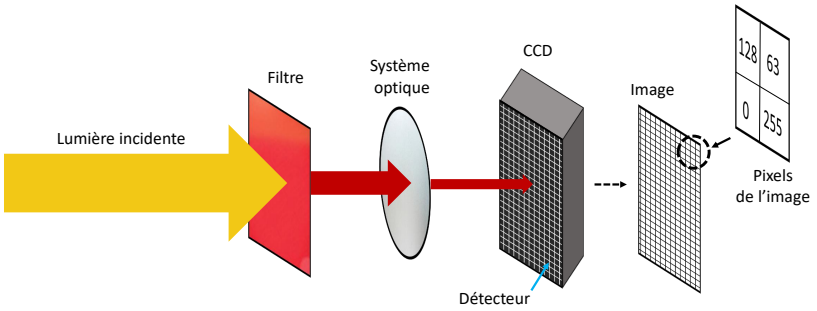


FIGURE 2.1 – La nomenclature des différents composants d'un capteur d'images.

Les capteurs multispectraux permettent d'acquérir différentes gammes de longueur d'onde

Lorsque nous observons une scène plane ou suffisamment éloignée, notre cerveau l'interprète comme une information à deux dimensions spatiales, accompagnée d'une information de couleur. Dans le cadre de l'imagerie hyperspectrale, il est pertinent de se représenter mentalement au lieu de cette information de couleur, qui n'existe que dans notre cerveau, une troisième dimension qui décrit le spectre de la scène. En effet, que ce soit par réflexion de la lumière solaire ou bien émission thermique, l'énergie lumineuse qui émane d'un objet est répartie dans une gamme de longueurs d'onde distinctes. Une représentation en trois dimensions d'une scène permet d'expliciter l'information spatiale de la scène sous la forme des deux premières dimensions x, y et l'information spectrale sous la forme de de la troisième dimension λ. Un élément (x_i, y_i, λ_i) de la scène correspond donc à l'intensité lumineuse d'un point spatialement défini à une longueur d'onde donnée. Cette représentation mentale d'une scène en tant qu'espace tridimensionnel permet de mieux appréhender le fonctionnement des différents types d'imageurs scientifiques existants (figure 2.2).

Plusieurs types de capteurs sont employés dans le domaine de la vision par ordinateur. Le plus simple de ceux-ci est celui dit *panchromatique* (figure 2.2, en haut à gauche). Il s'agit d'un capteur sans filtre, où chaque détecteur acquiert la somme spectrale de chaque point qui lui correspond spatialement. L'image acquise est en deux dimensions : chaque pixel (x_i, y_i) de l'image obtenue

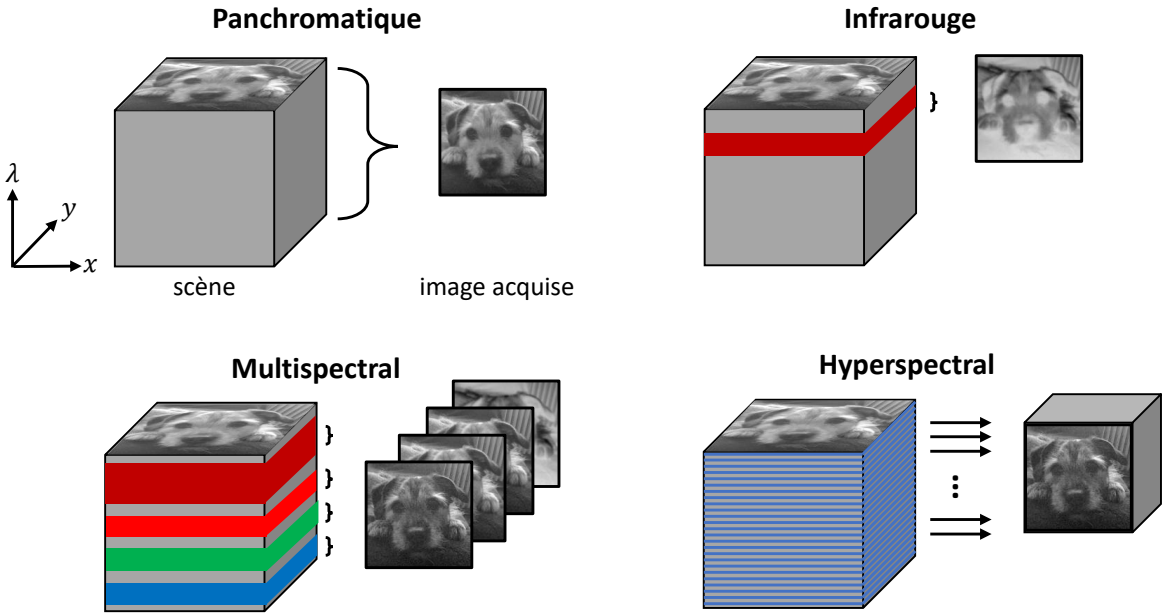


FIGURE 2.2 – Illustration des fonctionnements des différents types de capteur, en représentant la scène acquise comme une information tridimensionnelle.

contient une seule valeur. Les capteurs de photographie argentique qui produisent les photographies en « noir et blanc » sont panchromatiques, munis d'un filtre qui transmet uniquement les longueurs d'onde du domaine visible. Il est important de noter que l'absorption de la lumière incidente par chaque détecteur du CCD est pondérée par un coefficient qui dépend de la longueur d'onde de la lumière. Cette fonction de transfert, que l'on appelle la *sensibilité spectrale* du CCD, dépend de sa composition chimique. La plupart des CCDs utilisés dans les capteurs actuels sont en silicium, et leur sensibilité spectrale est une courbe en cloche centrée sur le domaine visible et le proche IR.

Les capteurs IR sont des capteurs panchromatiques (figure 2.2, en haut à droite) auxquels sont associés un filtre qui transmet une gamme de longueur d'onde dans le domaine IR. On dit que le capteur acquiert une *bande*, c'est-à-dire un ensemble restreint de longueurs d'onde contigües : l'information acquise par chaque détecteur correspond à une moyenne de l'intensité spectrale sur cette bande.

Les capteurs dits *multispectraux* (figure 2.2, en bas à gauche) acquièrent entre trois et quelques dizaines de bandes distinctes. Les capteurs RVB sont des exemples de capteurs multispectraux. Via des filtres de Bayer [Bayer, 1976], ils acquièrent des images contenant l'information de trois bandes spectrales du domaine visible, que notre cerveau combine pour produire une information de couleur [Grasset, 2020]. À ce titre, le système optique humain est également un capteur multispectral RVB. Pour des applications scientifiques, les bandes acquises correspondent souvent aux bandes RVB ainsi qu'une ou plusieurs bandes du domaine IR [Coffey, 2012]. Les capteurs multispectraux fonctionnent soit en acquérant séquentiellement les bandes en filtrant la lumière entrante successivement avec les différents filtres qui correspondent aux bandes que l'on souhaite acquérir, via par exemple une roue à filtres [Brauers et al., 2008] ; soit en utilisant un filtre composite comme celui de Bayer, composé de plusieurs micro-filtres rouge, vert et bleu qui alternent entre les détecteurs.

2.1.2 Capteurs hyperspectraux

Apparus beaucoup plus récemment que les capteurs multispectraux, les capteurs hyperspectraux acquièrent une information spectrale bien plus fine que ces derniers via un nombre très

important de bandes spectrales mesurées [Hagen and Kudenov, 2013] (figure 2.2, en bas à droite). Chacune représente une gamme de longueurs d'onde très restreinte, et les bandes sont contigües. Il est coutume de dire qu'un capteur hyperspectral acquiert un *cube* de données puisque la résolution spectrale de l'information acquise, de plusieurs dizaines à plusieurs centaines de bandes, est du même ordre de grandeur que sa résolution spatiale. Pour atteindre ce niveau de précision, les technologies optiques employées pour l'imagerie multispectrale, c'est-à-dire une multitude de filtres ou bien des filtres composites, ne sont pas viables pour un nombre de bandes beaucoup plus élevé. Les capteurs hyperspectraux utilisent à la place des éléments optiques permettant la décomposition spectrale de la lumière.

Les éléments dispersifs permettent de séparer finement l'information spectrale

Les éléments optiques dits *dispersifs* qui décomposent une lumière selon ses composantes spectrales. L'élément dispersif le plus connu [Kress, 2017] est le prisme. Ce bloc de verre taillé permet de décomposer spectralement la lumière puisque les rayons y empruntent un chemin différent en fonction de leur longueur d'onde. Un tel élément que l'on placerait entre une source de lumière et un CCD permettrait d'obtenir sur des détecteurs différents les intensités de différentes longueurs d'onde. Le terme « dispersif » est utilisé en optique comme synonyme de « décomposant spectralement ».

La majorité des capteurs hyperspectraux modernes incluent des éléments dispersifs que l'on appelle des *réseaux de diffraction*. Nous détaillons ici les réseaux de diffraction dits « à transmission », sachant qu'il en existe d'autres qui fonctionnent selon des principes similaires. Ces réseaux sont de petits matériaux en forme de rectangles plats, percés de multiples fentes espacées régulièrement. Les fentes sont de taille assez réduite pour causer le phénomène de diffraction, régi par la loi de Huygens-Fresnel [Huygens, 1920]. Dans le cadre d'un réseau de diffraction, cette loi stipule qu'au niveau des fentes, la lumière agit comme une source sphérique et se diffuse ainsi dans toutes les directions. La structure d'un réseau de diffraction associée à cette loi permet la décomposition de la lumière, comme l'illustre la figure 2.3.

Notons d la distance entre les fentes du réseau. Considérons deux rayons de longueur d'onde λ issus de deux fentes côte à côte, atteignant le même détecteur p (en bleu dans la figure 2.3). Nous admettons que la distance entre le réseau et le CCD est suffisamment grande pour considérer que ces rayons sont parallèles, formant un angle θ avec la normale du réseau. Les deux rayons ne parcourent pas le même chemin optique pour atteindre p. Par trigonométrie, nous pouvons calculer que leur différence de chemin D est égale à $d \sin\theta$ (en jaune sur l'agrandissement de la figure 2.3). Si cette différence de chemin est par ailleurs égale à $m\lambda$, $m \in \mathbb{N}$, alors les rayons sont en interférence constructive : le détecteur p reçoit l'information lumineuse correspondant à cette longueur d'onde. Pour toute autre longueur d'onde λ', la différence de chemin $D \neq m\lambda'$ est telle que les interférences sont partiellement destructives. Leur information est donc perdue pour ce détecteur p. Un détecteur donné reçoit donc l'information d'une longueur d'onde en particulier : la lumière est décomposée spectralement. Puisque les angles θ sont petits, si nous faisons l'approximation $\sin(\theta) \approx \theta$, nous pouvons remarquer de plus que les angles de dispersion maximale sont linéaires par rapport à λ. Si une longueur d'onde λ_1 est en interférence positive sur le détecteur p_1 défini par un angle θ_1, alors une longueur d'onde deux fois supérieure sera acquise sur un détecteur p_2 défini par un angle θ_2 approximativement deux fois plus grand que θ_1.

Si nous considérons, plutôt qu'un détecteur fixe, une longueur d'onde λ donnée, alors nous pouvons constate qu'il existe plusieurs angles de diffraction menant à une interférence constructive : tous ceux qui mènent à une différence de chemin égale à un multiple de λ. L'information correspondant à une longueur d'onde λ est donc acquise sur plusieurs détecteurs du CCD. Cette configuration mène à un ensemble de décompositions spectrales de la lumière incidente que l'on appelle des *ordres de diffraction*, (à droite de la figure 2.3). Par convention, ils sont numérotés selon

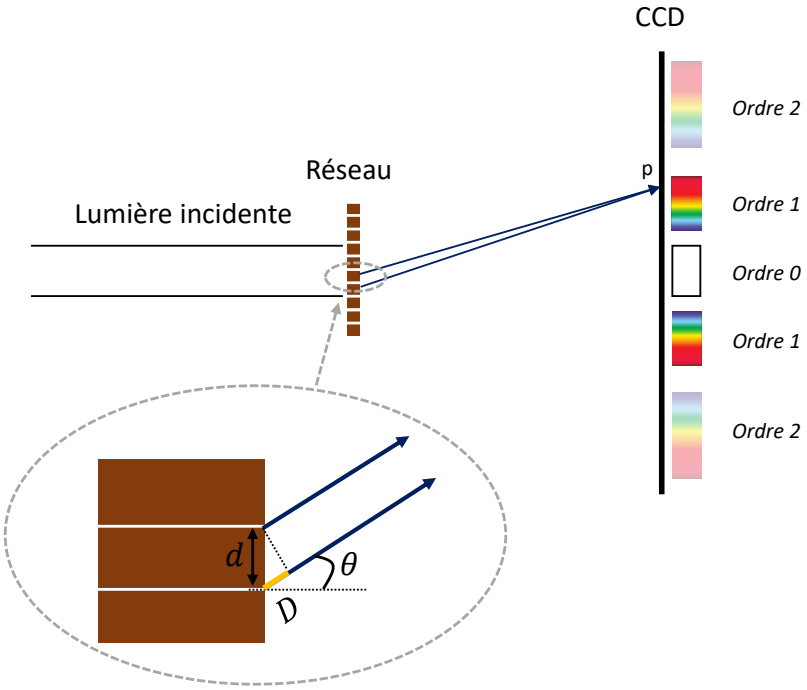


FIGURE 2.3 – Schéma de l'action d'un réseau de diffraction. La taille du réseau est exagérée à des fins d'illustration : il est en réalité de taille négligeable par rapport à la taille du CCD. Deux rayons lumineux issus de deux fentes côte à côte et atteignant le même détecteur sont représentés. La partie inférieure de la figure est un agrandissement de ces deux fentes.

les entiers naturels et ordonnés selon les angles de diffraction croissants. On distingue en particulier l'ordre 0 qui correspond à l'acquisition de la lumière non diffractée. La résolution spectrale, c'est-à-dire à quel point l'information spectrale est séparée spatialement sur le CCD, va en augmentant avec les ordres. Le nombre d'ordres qu'il est possible d'acquérir sur un CCD dépend de la taille de ce dernier et des conditions d'illumination. En effet, suivant les lois de la diffraction, la quantité d'énergie lumineuse va en décroissant selon les ordres, et certains ordres peuvent ainsi ne pas être visibles en conditions d'illumination défavorables.

Les capteurs à balayage acquièrent une scène en plusieurs itérations

La plupart des capteurs hyperspectraux intègrent un élément dispersif dans un système dit « à balayage » (*scanning* en anglais) [Hagen and Kudenov, 2013]. Ces capteurs acquièrent le cube hyperspectral de la scène en plusieurs itérations, en se « déplaçant » le long d'une dimension, spatiale ou spectrale. Parmi eux, les plus utilisés sont ceux à balayage spatial car ils permettent une décomposition plus importante que les systèmes à balayage spectral [Gat, 2000]. La figure 2.4 présente le fonctionnement d'un tel système dans le cadre d'un élément dispersif monté sur un support aéroporté se déplaçant selon un axe.

Dans ces systèmes, une acquisition concerne une « tranche » spatiale de la scène, c'est-à-dire une zone spatiale dont une des dimensions est négligeable par rapport à l'autre (représentée en couleur dans la « Scène » de la figure 2.4). Un élément dispersif décompose spectralement cette tranche. L'information ainsi créée est bidimensionnelle, et représente l'information spectrale de la tranche spatiale (« Acquisition » de la figure 2.4). Cette information est convertie par le capteur en une image, qui correspond à une tranche du cube. Afin d'acquérir le cube entier de la scène, il est nécessaire que soit l'objet étudié, soit le spectromètre se déplace. Les spectromètres de ce type étaient employés dans le domaine de la télédétection spatiale, où l'appareil reste fixe et où la rotation de la Terre permet le balayage dans la dimension spatiale orthogonale à la fente [Green et al., 1998].

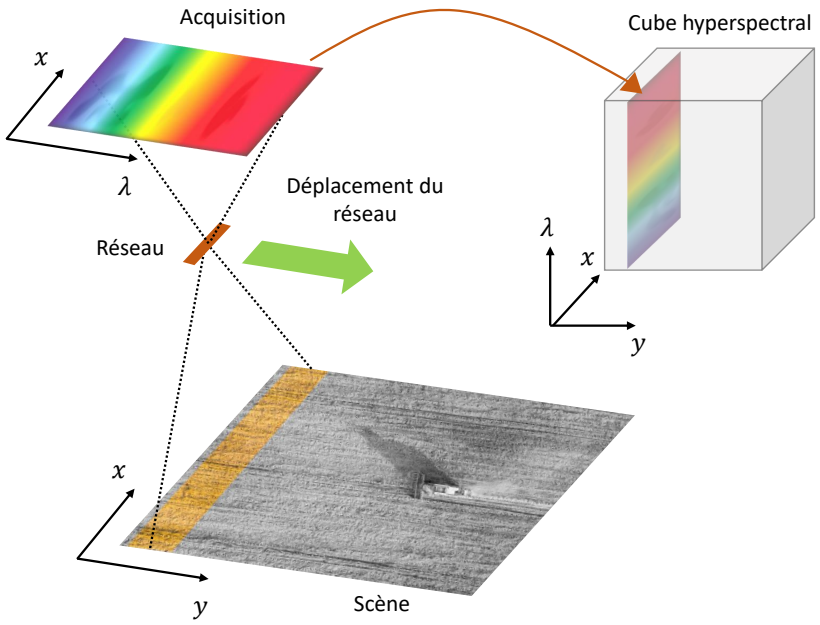


FIGURE 2.4 – Schéma illustrant le fonctionnement d'un capteur à balayage spatial. L'image bidimensionnelle acquise à un instant donné, en haut à gauche, constitue une tranche du cube hyperspectral, représenté en haut à droite. Pour acquérir tout le cube de la scène, il est nécessaire que le capteur se déplace, mouvement représenté par la flèche verte. Source de l'image du champ : https://shutterstock.com.

Les capteurs instantanés permettent une acquisition rapide de l'information

Une autre catégorie de spectromètres a vu son développement s'accélérer au cours des dernières décennies : les spectromètres *instantanés* (*snapshot* en anglais) [Hagen and Kudenov, 2013]. Ces systèmes permettent d'obtenir le cube entier d'une scène en une seule acquisition. Ils ont de nombreux avantages par rapport à leurs homologues à balayage. Premièrement, il n'est pas nécessaire que l'objet étudié ou que des éléments du système soient en mouvement, ce qui permet d'éviter des artefacts de déplacement et notamment le phénomène de flou [Hagen and Kudenov, 2013]. De plus, cette absence de mouvement des éléments du système le rend plus compact et robuste [Descour and Dereniak, 1995]. Par ailleurs, l'acquisition instantanée de la scène entière sans la filtrer spatialement ou spectralement permet une collecte de lumière bien plus importante que dans le cas des spectromètres à balayage [Hagen et al., 2012]. Enfin, cette catégorie de capteurs est adaptée à l'acquisition de l'information instantanée de scènes dynamiques comme il est courant de rencontrer en conditions extérieures.

Les auteurs de [Hagen and Kudenov, 2013] présentent et décrivent la liste exhaustive des différents capteurs hyperspectraux instantanés existants, dont le développement a commencé dans les années 1930. Le champ a par la suite beaucoup bénéficié des progrès du champ de l'*imagerie computationnelle* dans les années 1970. Ce terme désigne les systèmes d'imagerie indirects, où plusieurs acquisitions incomplètes de la scène nécessitent ensuite une étape de calculs faite par ordinateur. Ces techniques offrent l'avantage d'une plus grande capacité de représentation (par exemple dans le champ de l'imagerie à grande gamme dynamique [Reinhard et al., 2010]) voire la possibilité de former des images inaccessibles pour des imageurs classiques (par exemple dans le domaine de la *tomographie*, comme nous le détaillerons à la section 2.2.3). Les calculs peuvent aussi au moins partiellement relâcher les contraintes sur des systèmes optiques parfois difficiles et coûteux à manufacturer.

Certains capteurs hyperspectraux sont accessibles à un coût très bas

Alors que de nombreuses sociétés produisent des capteurs hyperspectraux dont les prix commencent à plusieurs dizaines de milliers d'euros, des études ont porté sur la possibilité de créer des capteurs à bas coût. Les articles dédiés préconisent l'utilisation de matériaux bon marché, voire à produire directement par des imprimantes 3D [Salazar-Vazquez and Mendez-Vazquez, 2020] (figure 2.5). Certains de ces capteurs sont à balayage et sont destinés en particulier au montage sur des drones [Uto et al., 2016; Sigernes et al., 2018], mais il existe aussi des recherches sur les capteurs instantanés, mieux adaptés à des applications véritablement bas-coût telles que des acquisitions à la main. Parmi ceux-ci, certaines solutions optent pour des conceptions originales, inspirés de capteurs instantanés développés plusieurs décennies en arrière [Mathews, 2008; Gao et al., 2010].

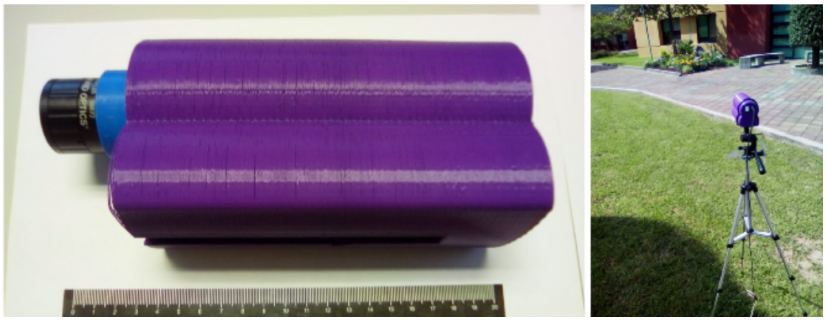


FIGURE 2.5 – Un capteur hyperspectral imprimé en 3D. Source : [Salazar-Vazquez and Mendez-Vazquez, 2020].

2.2 Le CTIS : un capteur du domaine de l'imagerie computationnelle

Le CTIS est le premier des capteurs hyperspectraux instantanés exploitant les capacités de l'imagerie computationnelle. Nous détaillons dans cette section le fonctionnement de ce spectromètre et la façon dont l'imagerie computationnelle permet l'acquisition instantanée de la scène.

2.2.1 L'acquisition d'une image destinée à une reconstruction

L'acquisition réalisée n'est pas directement un cube hyperspectral mais une image bidimensionnelle. Nous appelons cette image *l'image CTIS* et l'espace de ces images *l'espace de mesures*. Des calculs sont par la suite conduits sur cette image afin d'en produire le cube hyperspectral de la scène. On appelle ces calculs l'étape de *reconstruction* du cube. Ce *pipeline* général est illustré figure 2.6.

Le banc optique du CTIS est présenté figure 2.7. La lumière de la scène traverse d'abord une lentille d'objectif et un diaphragme de champ. Ces éléments ont pour conséquence une baisse de la résolution spatiale de la lumière acquise. Cette baisse permet de dégager une partie du CCD pour l'acquisition de l'information spectrale. Cette information est obtenue via le passage de la lumière dans un réseau de diffraction bidimensionnel, qui décompose l'information spectrale selon deux dimensions (figure 2.6, « Image CTIS »).

L'image acquise est ainsi partagée en plusieurs zones qui correspondent aux différents ordres de diffraction, que l'on appelle des *projections*. *L'ordre 0*, au centre, est la somme du cube selon la dimension spectrale. Il correspond à une acquisition que l'on obtiendrait via un capteur panchromatique à faible résolution spatiale. Les ordres supérieurs sont composés des différentes décompositions spectrales du cube. La figure 2.8 présente une image CTIS comprenant deux ordres, l'ordre 0 et l'ordre 1.

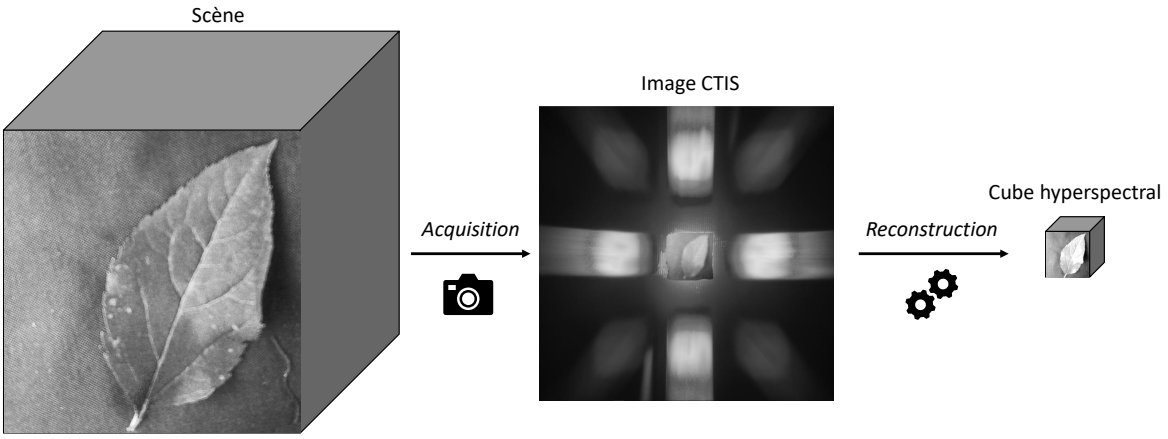


FIGURE 2.6 – *Pipeline* général d'une acquisition CTIS. Source des images « Scène » et « Image CTIS » : acquisition au LARIS avec la caméra Carbon Bee.

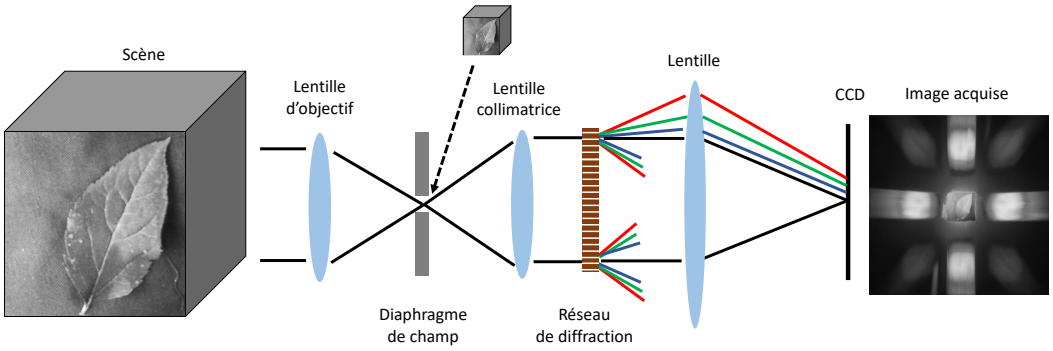


FIGURE 2.7 – Schéma du banc optique du CTIS. Le cube hyperspectral au sommet représente la lumière acquise au niveau du diaphragme de champ.

2.2.2 Un capteur relativement peu étudié

Le CTIS a été conçu indépendamment par les auteurs de [Okamoto and Yamaguchi, 1991] et de [Bulygin and Vishnyakov, 1992] avant d'être développé de manière plus approfondie par les auteurs de [Descour, 1994; Descour and Dereniak, 1995]. Ces derniers se sont notamment attachés à dépeindre les performances et les limitations du CTIS et à décrire précisément le protocole expérimental entier d'une acquisition, de la calibration du système à la reconstruction du cube. Ces mêmes auteurs ont par la suite popularisé l'utilisation du CTIS via des preuves de concept dans certaines gammes spécifiques de longueurs d'onde [Volin et al., 2001] et pour des applications militaires [Descour et al., 1995, 1997]. Les premières expériences abouties furent menées dans le champ de la biologie moléculaire, en particulier dans le cadre de l'imagerie par fluorescence [Volin et al., 1998; Ford et al., 2001a,b].

Le CTIS bénéficia au fil de ces expériences d'améliorations matérielles. L'augmentation progressive de la taille du CCD utilisé permit une meilleure résolution spatiale comme spectrale. De nouvelles structures d'éléments dispersifs permirent d'augmenter le nombre de projections acquises. Une série de travaux s'intéressa en particulier à ces éléments dispersifs, en générant des simulations d'images CTIS via des réseaux de diffraction théoriques [Hagen et al., 2006; Hagen and Dereniak, 2007, 2008]. D'autres travaux furent conduits sur l'amélioration de la calibration du système [Wilson et al., 1997]. La maturation de la technologie permit son application dans les champs de l'ophtalmologie [Johnson et al., 2007; Lee et al., 2010] et de l'astronomie [Hege et al., 2004; Smart and Kankelborg, 2018].

Il est à noter que pour tous ces travaux, les cubes produits par les CTIS ont servi pour des

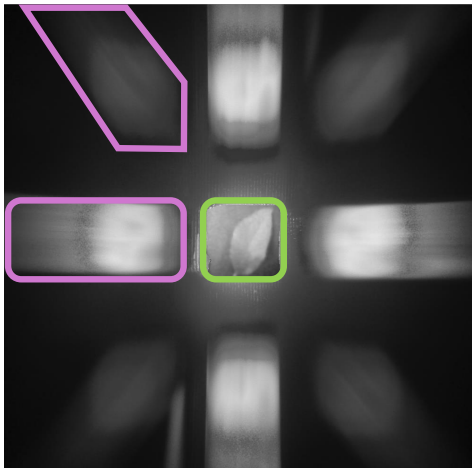


FIGURE 2.8 – Position des différents ordres sur une image CTIS. L'ordre 0 est encadré en vert. Toutes les autres projections, y compris celles encadrées en rose, appartiennent à l'ordre 1.

analyses qualitatives effectuées « à la main », par exemple pour une sélection manuelle de bandes spectrales pertinentes [Johnson et al., 2007] ou pour caractériser spectralement des objets [Hege et al., 2004]. Il n'y a jamais eu d'analyse automatisée à grande échelle des cubes produits comme cela a été fait pour des images issues d'autres capteurs hyperspectraux [Chen et al., 2016]. En vingt-cinq ans d'existence, le CTIS a donc été relativement peu utilisé dans des applications scientifiques. Cependant, grâce à ses qualités que sont une grande vitesse d'acquisition, une certaine robustesse et un prix bas par rapport à d'autres caméras hyperspectrales, le CTIS a été intégré cette dernière décennie dans plusieurs caméras hyperspectrales bas-coût innovantes [Johnson et al., 2007; Habel et al., 2012; Germain, 2019; Salazar-Vazquez and Mendez-Vazquez, 2020].

2.2.3 Un lien fort avec la tomographie

Pour mieux comprendre le fonctionnement, et donc les possibilités et les limitations, du CTIS,, il faut s'intéresser à l'action du réseau de diffraction qui est au cœur du système. Pour cela, il est utile de se représenter les projections de l'image CTIS dans le sens mécanique du terme, c'est-à-dire que chaque projection est ce que verrait du cube hyperspectral un capteur placé à une certaine position autour de ce cube. Nous rappelons que la troisième dimension du cube n'est en réalité pas une dimension spatiale, aussi cette illustration mécanique est-elle nécessairement indicative. Elle représente cependant exactement l'effet qu'a le réseau de diffraction sur le cube spatio-spectral (figure 2.9).

En ce sens, le principe optique du CTIS est le même que celui de la tomographie assistée par ordinateur (*computed tomography* en anglais, ou CT, domaine qui a donné le nom au CTIS). Il nous parait dès lors pertinent de détailler la procédure d'acquisition conduite en CT, car celle-ci donne les éléments pour comprendre plus en profondeur l'action de l'élément dispersif et la procédure de reconstruction du cube dans le cadre du CTIS.

L'acquisition de multiples signaux incomplets

Dans le domaine médical, le but d'une acquisition par CT est d'obtenir l'image d'une tranche d'un patient. Ce dernier est placé au centre d'une structure torique, que l'on appelle un scanner, où sont placés des émetteurs ponctuels de rayons X. Chaque émetteur est placé en face d'un récepteur. Au cours de l'acquisition, chaque émetteur envoie un rayon dans la direction de son récepteur, traversant ainsi le patient. La longueur d'onde des rayons est choisie de façon à ce que ces derniers soient absorbés différemment en fonction du type de matériau qu'ils traversent. En d'autres termes, les différents tissus et organes du corps humain ont des coefficients d'absorption aux rayons

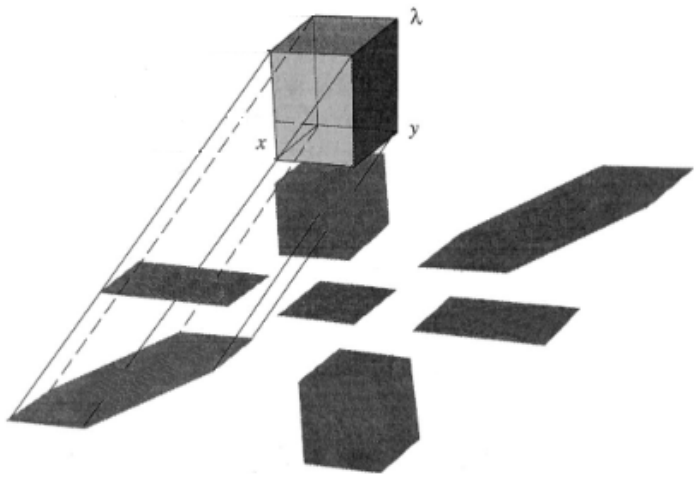


FIGURE 2.9 – Représentation des projections acquises (en bas) comme des projections mécaniques du cube hyperspectral (en haut). Source : [Descour and Dereniak, 1995].

différents, connus d'avance. Ainsi, l'intensité reçue par un récepteur donné est indicatrice de la composition des tissus que le rayon émis correspondant a traversé. Cependant, une mesure d'un récepteur donné ne fournit qu'une information partielle quant à ce chemin. En effet, un récepteur acquiert une seule valeur et permet donc uniquement de mesurer la quantité totale de l'énergie absorbée au cours du trajet du rayon. En notant I_0 l'intensité du rayon émis, I_1 celle du rayon acquis, et $\mu(x)$ le coefficient d'absorption à la position x du trajet du rayon, $x \in [0, L]$ nous avons l'équation

$$I_1 = I_0 \int_0^L \mu(x) dx. \tag{2.1}$$

Ainsi, la valeur acquise ne permet pas d'expliciter les coefficients $\mu(x)$ individuels du trajet du rayon. L'acquisition CT repose sur l'idée d'acquérir un grand nombre de rayons à l'information imparfaite et de rassembler les informations obtenues pour reconstruire la tranche entière. Il s'agit donc une technique d'imagerie computationnelle.

La disposition des émetteurs dans le scanner est illustrée figure 2.10. Un ensemble d'émetteurs positionnés côte à côte émettent un rayon vers des émetteurs qui leur font face. L'ensemble des mesures faites sur les récepteurs pour une position s'appelle une projection (figure 2.10, gauche). Au cours de l'acquisition, le scanner pivote sur son axe, permettant de déplacer les émetteurs et récepteurs. Les projections sont acquises régulièrement, permettant de couvrir une plage importante d'angles d'acquisition, noté θ sur la figure 2.10. À la fin de l'acquisition, les projections unidimensionnelles acquises sont parfois regroupées sous la forme d'une image bidimensionnelle appelée un *sinogramme* (figure 2.10, droite).

Les algorithmes de reconstruction permettent de retrouver le signal original

L'ensemble de ces projections et les angles avec lesquelles elles ont été acquises permet dans un deuxième temps la reconstruction de l'image. Un algorithme appelé la « rétro-projection filtrée » (*Filtered Back-Projection* en anglais, ou FBP) est l'algorithme de reconstruction implémenté aujourd'hui dans de nombreux scanners commerciaux [Pan et al., 2009]. Il consiste en un « ré-étalement » des projections dans le domaine spatial, où chaque projection est filtrée fréquentiellement afin d'éviter une surreprésentation des fréquences basses. Il a été prouvé que cet algorithme correspond mathématiquement à l'opération inverse de l'action du scanner, attendu des conditions idéales et un nombre de projections infinies [Al Hussani and Al Hayani, 2014]. Cependant, si cet algorithme est théoriquement parfait, il en va autrement dans la réalité. De nombreuses limitations expérimentales nous éloignent de la situation idéale où le FBP est une solution exacte. Certaines de ces

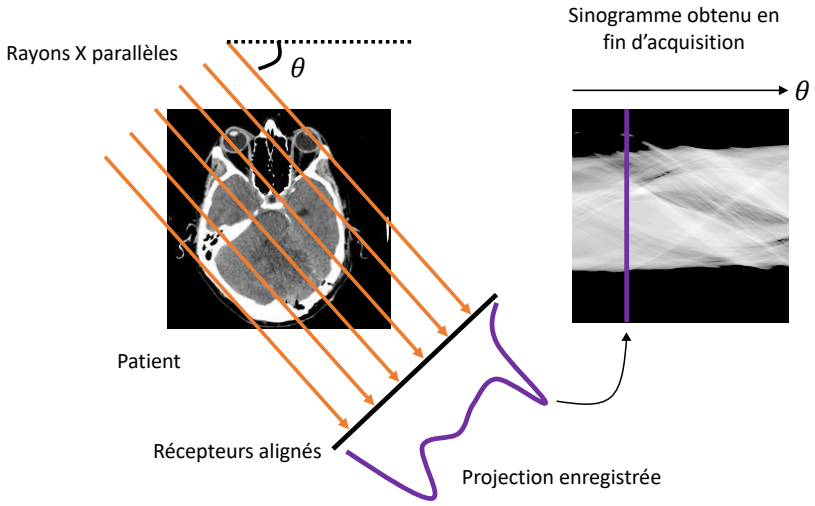


FIGURE 2.10 – Le principe de fonctionnement d'une acquisition CT. Source des images « Patient » et « Sinogramme » : [Siltanen, 2017].

limitations ont trait à la nature discrète de la réalité. En particulier, le nombre de récepteurs est discret et donc les projections le sont aussi. De même, le nombre d'angles d'acquisition est fini, et par conséquent le nombre de projections l'est aussi. De plus, certaines gammes d'angle peuvent être inaccessibles à cause de contraintes du système ou du patient.

De surcoît, il existe aussi des problématiques de *bruit* qui sont absentes du modèle idéal. Premièrement, lorsque le nombre de photons acquis par un récepteur est bas, il est nécessaire de prendre en compte le bruit statistique qui apparaît sous la forme de bruit de *grenaille* (*shot noise* en anglais). Ce bruit se traduit concrètement par une valeur aléatoire ajoutée ou multipliée à chaque pixel du sinogramme. Par ailleurs, au sein du patient, il peut se dérouler un phénomène dit de diffusion qui cause le changement de direction de photons. Un photon qui arrive à un récepteur ne provient donc pas nécessairement de l'émetteur associé par le chemin direct, mais peut provenir d'un autre émetteur via un chemin inconnu. De plus, de nombreux autres phénomènes causent des artefacts dans l'acquisition : le « durcissement du faisceau » causé par un rayon X trop poly-énergétique, des artefacts causés par des objets métalliques, des anneaux dus à une mauvaise calibration des récepteurs, des artefacts de flou causés par le mouvement de l'objet durant l'acquisition, etc. [Boas and Fleischmann, 2012].

En réponse à ces limitations des acquisition réelles, une autre famille d'algorithmes est régulièrement utilisée en reconstruction CT : les méthodes itératives. Ces méthodes se basent sur une description algébrique et discrétisée du problème. L'image originale que l'on cherche à acquérir (c'est-à-dire la tranche du patient) est décrite comme un vecteur f unidimensionnel d'une longueur égale au nombre de pixels N dans laquelle elle a été discrétisée, en listant les éléments qui les composent les uns à la suite des autres. L'ensemble des projections (c'est-à-dire le sinogramme) est décrit comme un vecteur g d'une longueur égale au nombre de projections fois le nombre de récepteurs, c'est-à-dire au nombre de mesures acquises M. L'action du scanner est supposée être une action linéaire entre la tranche du patient et le sinogramme, et est donc représentée par une matrice H de taille M × N. L'acquisition des projections est donc décrite par l'équation

$$g = \mathrm{H}f \tag{2.2}$$

où H et g sont connus. Le but de la reconstruction est de déterminer l'image originale f. La résolution directe de l'inverse de cette équation, c'est-à-dire l'équation

$$f = \mathrm{H}^{-1}g \tag{2.3}$$

est impossible car la matrice H est très grande, très éparse (c'est-à-dire qu'elle contient beaucoup de termes nuls), et non carrée : on dit que la matrice est très « mal conditionnée ». Cela signifie que lors de la résolution de l'équation 2.3, un bruit même faible dans la mesure de g peut mener à un bruit très grand dans la prédiction de f. Or, il est très probable que la mesure de g soit effectivement bruitée en conséquence des problématiques de bruit listées plus haut. Dans ce cas, les méthodes itératives qui prennent en compte la possibilité de bruit sont les plus adéquates. Il ne s'agit pas de résoudre l'équation 2.3, mais de trouver la meilleure solution à l'équation

$$g = \mathrm{H}f + s \tag{2.4}$$

où s est un vecteur de la même taille que g et qui représente le bruit. Le but est alors de trouver le vecteur f le plus vraisemblant sachant H et g. Le procédé le plus utilisé à cette fin est celui proposé par les auteurs de [Shepp and Vardi, 1982]. Dans cette étude, le bruit s est considéré comme du bruit de grenaille et les valeurs de f sont modélisées comme des variables aléatoires suivant une loi de Poisson centrée sur leur vraie valeur inconnue. Pour trouver le maximum de vraisemblance de f, l'algorithme itératif de l'espérance-maximisation (*Expectation-Maximization* en anglais ou EM) est employé.

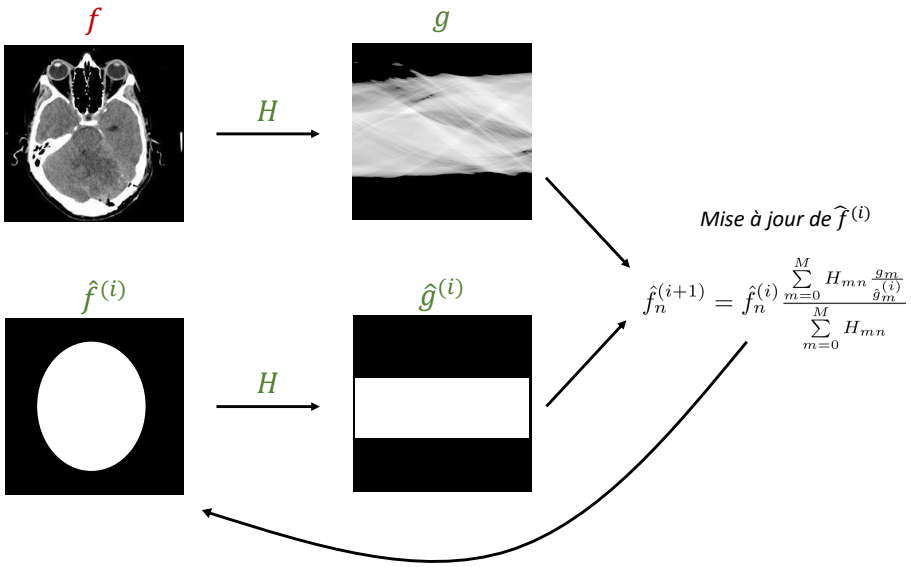


FIGURE 2.11 – Schéma du fonctionnement d'une itération de l'algorithme des auteurs de [Shepp and Vardi, 1982] pour la reconstruction d'une image CT. Les objets connus sont indiqués en vert et les inconnus en rouge. Les noms des différents objets sont explicités dans le texte.

Le déroulement de l'algorithme, dont une itération est illustrée à la figure 2.11, est le suivant. Nous notons i l'itération courante, en commençant à $i = 0$.

1. Proposer une estimation initiale de l'image f, notée $\hat{f}^{(0)}$.
2. Calculer le sinogramme de l'estimation actuelle de f : $\hat{g}^{(i)} = \mathrm{H}\hat{f}^{(i)}$. Nous rappelons que cette étape est possible car nous considérons H, l'action du scanner, comme connue.
3. Mettre à jour f en fonction de la différence entre ce sinogramme calculé $\hat{g}^{(i)}$ et le sinogramme mesuré g. En notant $\hat{f}_n^{(i)}$ chaque élément de f, $\hat{g}_m^{(i)}$ chaque élément de $\hat{g}^{(i)}$ et g_m chaque élément de g, alors mettre à jour chaque $\hat{f}_n^{(i)}$ selon l'équation

$$\hat{f}_n^{(i+1)} = \hat{f}_n^{(i)} \underbrace{\frac{\sum_{m=0}^{\mathrm{M}} \mathrm{H}_{mn} \frac{g_m}{\hat{g}_m^{(i)}}}{\sum_{m=0}^{\mathrm{M}} \mathrm{H}_{mn}}}_{\mathrm{T}}. \tag{2.5}$$

4. Reprendre à l'étape 2 jusqu'à un critère de convergence, par exemple une valeur seuil pour la différence entre g et $\hat{g}^{(i)}$.

Nous nous proposons d'expliciter l'équation 2.5. Premièrement, nous pouvons noter que la mise à jour de l'image f se fait pixel par pixel : chaque élément $\hat{f}_n^{(i+1)}$ dépend uniquement de $\hat{f}_n^{(i)}$ et pas d'autres éléments $\hat{f}_k^{(i)}, k \neq n$. Au cours d'une mise à jour, chaque élément $\hat{f}_n^{(i+1)}$ est calculé comme la multiplication de cet élément à l'itération précédente $\hat{f}_n^{(i)}$ par un terme noté T. Notons que T dépend uniquement de H et de g, qui ont des valeurs positives ou nulles : T est donc également positif ou nul. Pour calculer T, nous comparons le sinogramme véritable g au sinogramme calculé à l'itération courante $\hat{g}^{(i)}$. Cette comparaison est simplement une somme des divisions terme à terme des valeurs de g et $\hat{g}^{(i)}$. Chaque division est en outre pondérée par la participation de f_n à la valeur que prennent les éléments de g et de $\hat{g}^{(i)}$. Cette participation est la valeur H_{mn} (terme bleu, « normalisé » par le terme marron). Ainsi, si pour une position m donnée, l'estimation courante de $\hat{g}_m^{(i)}$ est loin de celle du véritable sinogramme g_m, et que f_n contribue fortement à g_m et à $\hat{g}_m^{(i)}$, alors le terme $H_{mn}\frac{g_m}{\hat{g}_m^{(i)}}$ influence fortement T. À l'inverse, par exemple dans un cas où $H_{mn} = 0$, cette différence ne contribue pas du tout à T. Ainsi, si les valeurs de $\hat{g}^{(i)}$ auxquelles $\hat{f}_n^{(i)}$ contribue sont en moyenne (c'est-à-dire en sommant sur toutes les positions m) trop grandes par rapport à g, alors T sera inférieur à 1 et $\hat{f}_n^{(i+1)}$ sera plus petit que $\hat{f}_n^{(i)}$. Nous pouvons tirer les conclusions inverses pour le cas où la contribution moyenne de $\hat{f}_n^{(i)}$ est trop faible par rapport au g attendu. Comme d'autres méthodes de maximum de vraisemblance, cet algorithme permet toujours de trouver un maximum local de vraisemblance, mais ne garantit pas la globalité de celui-ci. Comme il n'y a pas de stochasticité dans cet algorithme, le choix de $\hat{f}^{(0)}$ a une grande importance.

Il existe aussi des méthodes plus modernes de reconstruction CT basées sur les réseaux de neurones. Il est intéressant de constater que dans ces méthodes, les réseaux sont pensés de manière à reproduire les étapes d'algorithmes de reconstruction déjà existants, ou tout au moins les différentes parties du réseau y sont-elles identifiées. Les travaux de [Würfl et al., 2016] par exemple, cherchent à émuler l'algorithme de FBP. Ils identifient les couches convolutives de leur réseau au filtrage fréquentiel, les couches FC à la rétro-projection en elle-même et la fonction d'activation à la contrainte de positivité. Les travaux de [Hammernik et al., 2017] proposent deux réseaux, l'un identifié à la FBP et l'autre comme un algorithme de réduction d'artefacts.

Le CTIS est une opération de CT basée sur la lumière

Le CTIS fonctionne selon le principe du CT appliqué à un cube de lumière. Les similarités des deux procédés sont présentées dans le tableau 2.1. Notons que contrairement au CT , le CTIS permet d'obtenir *in fine* une image tridimensionnelle à partir de projections bidimensionnelles. Cependant, tous les concepts de la CT présentés dans la section précédente sont généralisables à des dimensions supérieures.

	CT	CTIS
Objet à acquérir	Tranche d'un patient (2D)	Scène vue comme un cube hyperspectral (3D)
Onde pénétrante	Rayons X	Lumière de la scène
Moyen de génération de projections	Émetteurs pivotant autour d'un axe	Élement dispersif
Image acquise	Sinogramme	Image CTIS

TABLEAU 2.1 – Comparaison entre le procédé d'acquisition du CT médical et celui du CTIS.

2.2.4 La reconstruction du signal CTIS est épineuse et approximative

La reconstruction du signal CTIS présente quelques difficultés supplémentaires par rapport à celle menée dans un cadre de CT médical. Premièrement, le nombre de projections est très limité par rapport à celui permis par un scanner. Leur nombre dépend de la structure de l'élément dispersif et de la taille du CCD, mais même les CTIS les plus récents ne dépassent pas une cinquantaine

de projections [Hagen et al., 2006], alors que les scanners en acquièrent des centaines. La figure 2.12 présente un exemple d'image CTIS à 25 projections.

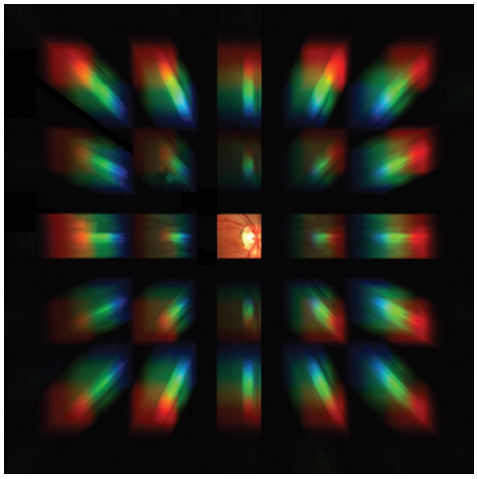


FIGURE 2.12 – Une image CTIS à 25 projections acquise dans un cadre d'étude ophtalmologique. Source : [Johnson et al., 2007].

Dans la plupart des systèmes CTIS, ce nombre est même encore plus réduit, en dessous de la dizaine. De plus, contrairement au scanner qui pivote autour du patient, la scène n'est acquise ici que d'un point de vue fixe. La diffraction permet de générer des projections dans des directions différentes mais il n'en reste pas moins qu'une part importante des angles de projection est impossible à acquérir : ceux qui définissent la demi-sphère qu'il serait possible d'acquérir si le capteur était placé « de l'autre côté » de la scène. Il est possible de faire le lien entre le nombre et la position des projections et la qualité attendue du signal reconstruit via le « théorème de la tranche centrale » [Bracewell, 1956] (figure 2.13). Dans un cadre de CT, ce théorème stipule que, pour une image bidimensionnelle donnée, les deux mesures suivantes sont équivalentes : (i) la transformée de Fourier d'une projection faite à un angle θ, et (ii) la tranche de la transformée de Fourier de l'image, à un angle θ et passant par le centre de l'image. Ainsi, le nombre de projections conditionne la proportion de fréquences de la scène originale que nous retrouvons dans le signal reconstruit. Ce théorème se généralise à plus haute dimension. Dans le cas du CTIS , la figure 2.14 illustre le faible remplissage de l'espace fréquentiel dû aux nombre réduit de projections.

Une autre difficulté du CTIS est que le phénomène de diffraction, qui permet la décomposition du cube et donc l'acquisition de projections, conduit à une division de l'intensité de la lumière incidente entre les ordres (section 2.1.2). Ainsi, certaines projections sont plus lumineuses que d'autres, et, sous des mauvaises conditions d'illumination, certains ordres peuvent ne pas apparaître.

Enfin, l'explicitation de la matrice H qui définit l'action du système peut être délicate. L'écrire *a priori* nécessite une connaissance fine du système optique, des phénomènes de diffraction et des aberrations optiques s'y déroulant. Il est alors recommandé de déterminer H de manière expérimentale en utilisant des sources lumineuses dont le spectre est connu [Descour and Dereniak, 1995]. Notons au passage que dans le cas du CTIS, H définit uniquement l'action du réseau de diffraction, c'est-à-dire la conversion entre le cube hyperspectral réduit au diaphragme de champ et l'image acquise (figure 2.7). L'action de la lentille d'objectif et du diaphragme de champ mène à une baisse de résolution spatiale qu'il est impossible de recréer à partir de l'algorithme de reconstruction.

Malgré ces difficultés, les études sur le CTIS reprennent pour réaliser la reconstruction les formalisations et algorithmes du domaine du CT. En particulier, l'algorithme EM est utilisé dès l'étude de [Descour and Dereniak, 1995]. Les mêmes auteurs ont implémenté plus tard la « technique de

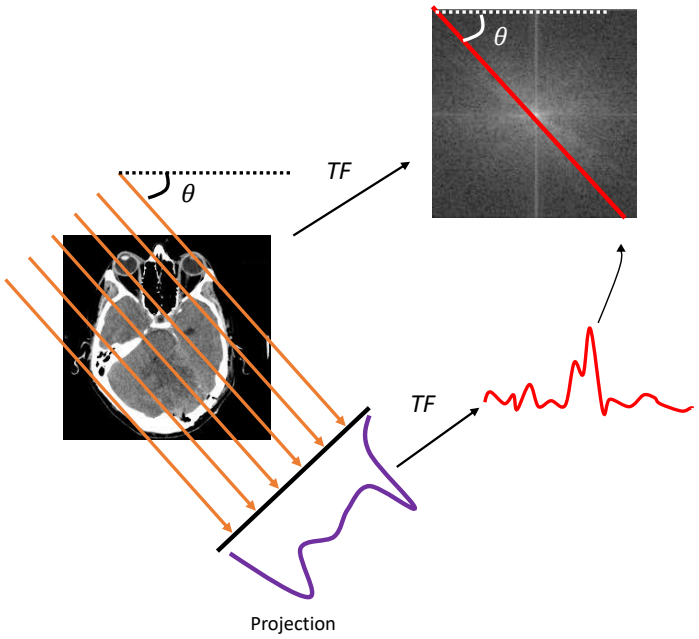
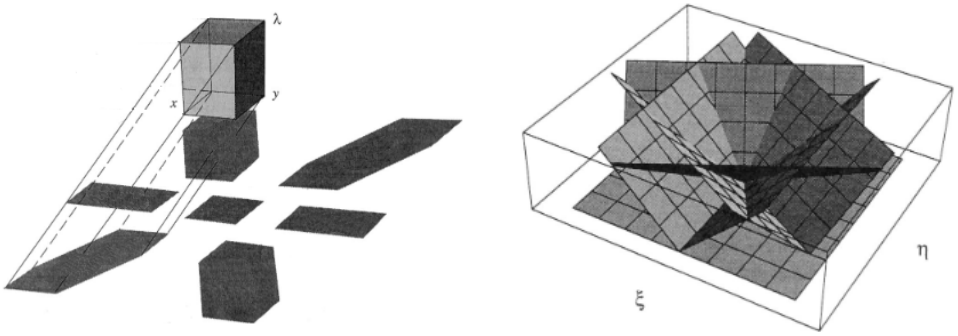


FIGURE 2.13 – Le théorème de la tranche centrale permet de faire le lien entre une projection et l'image dont elle provient dans le domaine fréquentiel. « TF » indique la transformée de Fourier.



(a) Domaine spatial (b) Domaine fréquentiel

FIGURE 2.14 – Illustration de la faible conservation de l'information fréquentielle d'une image lors de l'action d'un CTIS à faible nombre de projections. (a) Les projections du CTIS dans le domaine spatial et (b) la répartition de l'information dans le domaine fréquentiel d'après le théorème de la tranche centrale dans sa version tridimensionnelle (b). Source : [Descour and Dereniak, 1995].

reconstruction algébrique multiplicative » (*Multiplicative Algreabric Reconstruction Technique* en anglais, ou MART) [Gordon et al., 1970], et les études ultérieures ont utilisé soit l'algorithme EM [Descour et al., 1995, 1997; Hege et al., 2004; Johnson et al., 2006, 2007; Lee et al., 2010] soit le MART [Ford et al., 2001a,b; Hagen et al., 2006; Hagen and Dereniak, 2007]. La différence entre les deux algorithmes tient à la modélisation du bruit dans l'image. Il existe deux types de bruit que l'on peut choisir d'inclure dans le modèle du CTIS : le bruit de grenaille, décrit plus haut, et le bruit thermique lié aux fluctuations thermiques des éléments électroniques, suivant une loi normale. Dans la plupart des études, un des types de bruit est considéré comme tellement prépondérant dans l'image CTIS que l'autre est considéré comme négligeable. Dans le cas de l 'EM, c'est comme nous l'avons dit le bruit de grenaille qui est considéré comme dominant, tandis que dans le cas du MART, le bruit est considéré comme principalement de source thermique. Quelques études ont été conduites spécifiquement pour améliorer ces algorithmes. Nous pouvons citer en particulier la technique de reconstruction à espérance mélangée (*Mixed Expectation Reconstruction Technique*) [Garcia and Dereniak, 1999] qui prend en compte les deux sources de bruit sus-citées, arguant que

la prédominance d'un certain type de bruit pouvait varier en fonction de la localisation dans l'image acquise en fonction des conditions d'illumination. Cependant, cette initiative, comme d'autres [Hagen et al., 2007; Vose and Horton, 2007] n'ont jamais été implémentées dans des études publiées ultérieures. Il est aussi intéressant de noter que, bien que les algorithmes de reconstruction de CT médical aient grandement évolué depuis l'utilisation de l'algorithme EM, (cf. dernier paragraphe de la section 2.2.3) ces innovations n'ont jamais été portées au domaine du CTIS.

2.3 Conclusion

Le CTIS est un capteur hyperspectral, c'est-à-dire qu'il permet d'acquérir l'information d'un grand nombre de longueurs d'onde distinctes d'une scène. De surcoît, il s'agit d'un capteur instantané, une caractéristique utile pour les acquisitions en extérieur comme celles que mènent Carbon Bee. En contrepartie, une étape de calcul, identique à celle conduite en CT, est nécessaire afin de reconstruire l'information spectrale à partir du signal acquis par le spectromètre. Nous présentons dans le chapitre 4 des travaux que nous avons mené afin d'exploiter le signal produit par le CTIS qui permettent notamment de contourner les difficultés liées sa reconstruction. Nous avons conduit cette étude sur des données simulées, dont la création est présentée dans le chapitre suivant.

Chapitre 3

Des simulateurs pour évaluer le capteur CTIS

Afin d'évaluer la performance du capteur CTIS dans le cadre d'une application d'apprentissage automatique, nous avons procédé à une détection de tavelure à l'échelle de la feuille de pommier, c'est-à-dire en nous basant sur des images de feuilles isolées en conditions contrôlées. La feuille est parmi les organes d'une plante celle qui offre le plus d'indications visuelles quant à la santé de celle-ci, et elle fut à ce titre massivement étudiée dans des applications de vision par ordinateur [Martinelli et al., 2015; Khirade and Patil, 2015; Cerutti et al., 2013]. Aujourd'hui, les études portant sur des feuilles isolées sont pour beaucoup destinées à démontrer la faisabilité de nouvelles méthodes de détection [Belin et al., 2013] que celles-ci soient matérielles ou algorithmiques, et ce n'est que lorsque ces technologies sont arrivées à une maturité plus importante qu'elles sont évaluées à des échelles plus complexes (canopée, champ, etc.) [Wang et al., 2010]. Nous avons jugé que l'étude à l'échelle de la feuille, en conditions contrôlées, était adaptée au niveau de maturité technologique du capteur CTIS. Par ailleurs, comme cela est très courant pour des capteurs innovants, nous nous sommes tournés vers la simulation de données pour mener à bien cette étude [Spoelder, 1999]. La simulation permet en effet de générer un nombre illimité de données annotées à très faible coût. Elle peut permettre en outre d'affiner les analyses, par exemple en générant des configurations qu'il est rare de rencontrer dans le monde réel [Chawla et al., 2002]. Dans cette étude, nous avons exploité la flexibilité du procédé de simulation pour mener une analyse fine des différents stades d'infection de la tavelure. Ce chapitre présente la création de ces données simulées via le développement de deux nouveaux simulateurs.

Sommaire

3.1 Simulateur de cubes hyperspectraux de feuilles tavelées . . . 44
3.1.1 Notions préliminaires de traitement d'images . . . 44
3.1.2 Algorithme général . . . 44
3.1.3 Collecte des images des feuilles saines . . . 46
3.1.4 Distribution spatiale des lésions . . . 47
3.1.5 Acquisition expérimentale des spectres . . . 49
3.1.6 Exemple d'un cube simulé . . . 52
3.2 Simulateur de CTIS . . . 53
3.2.1 Modèle discrétisé de l'action du réseau de diffraction . . . 53
3.2.2 Intégration d'aspects matériels . . . 55
3.2.3 Algorithme général . . . 57
3.2.4 Détermination de la matrice d'action du système . . . 58
3.2.5 Simulateurs RVB et IR . . . 59
3.3 Création des jeux simulés . . . 60
3.4 Conclusion . . . 61

Nous avons créé un jeu de données simulées d'images CTIS de feuilles tavelées en couplant deux simulateurs distincts.

— D'une part, nous avons développé un simulateur « spectral » de cubes hyperspectraux de feuilles tavelées acquises en conditions contrôlées.

— De l'autre, nous avons produit un simulateur « optique » de CTIS qui reproduisait le fonctionnement de ce spectromètre.

Les cubes créés par le premier simulateur étaient fournis en entrée au deuxième, permettant de générer des images CTIS de feuilles tavelées. Nous présentons à présent plus en détail ces simulateurs. Tout le code nécessaire au travail présenté dans ce manuscrit relatif au traitement d'images a été réalisé en Python 3.6 avec la librairie OpenCV 4.1.0.

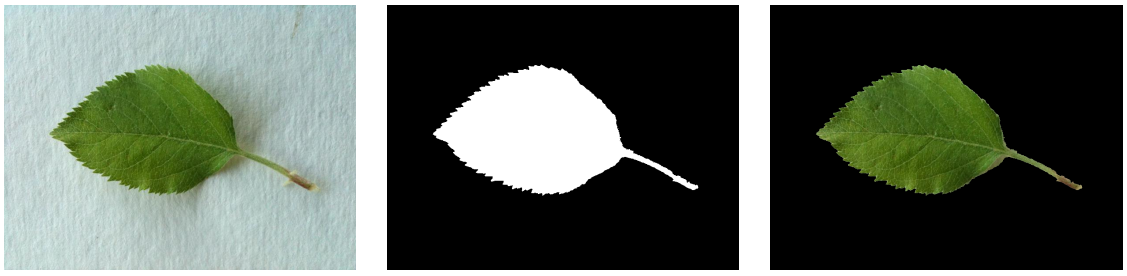
3.1 Simulateur de cubes hyperspectraux de feuilles tavelées

3.1.1 Notions préliminaires de traitement d'images

Dans tout le travail de ce chapitre, nous avons manipulé des images selon des opérations typiques du domaine du traitement d'images. Les images étant considérées pour un ordinateur comme de simples tableaux de nombres, il était tout d'abord possible de procéder à des opérations mathématiques standards telles que l'addition pixel à pixel et la multiplication de tous les pixels par un nombre. Nous avons traité des images encodées sur 8 bits, ce qui signifiait que les pixels peuvent prendre une valeur entière entre zéro (noir) et 255 (blanc). Dans le cas d'images RVB, chaque pixel stockait trois valeurs en 8 bits correspondant aux trois canaux de couleurs.

Nous nous sommes appuyés pour nos simulateurs sur l'opération de *seuillage*, qui consiste à mettre à 255 (blanc) les pixels d'une image si ceux-ci respectent un certain critère, et de mettre à zéro (noir) les autres. Un cas classique du seuillage est le seuillage binaire, où le critère de sélection des pixels est une comparaison de leur valeur avec une certaine valeur donnée. Les images (a) et (b) de la figure 3.1 illustrent le principe d'un seuillage basé sur la conservation de la couleur verte.

L'image (b) de cette figure est un exemple de *masque*, c'est-à-dire une image binaire représentant typiquement la localisation d'un certain objet dans l'image. On dit que l'on procède au *masquage* d'une image I par un masque M lorsqu'on conserve de I uniquement les pixels correspondant aux pixels blancs de M, et que l'on fixe les autres pixels à zéro. Les images (a), (b) et (c) de la figure 3.1 illustrent une telle opération de masquage.



(a) Image originale.

(b) Masque obtenu par seuillage de la couleur verte.

(c) Image (a) masquée par l'image (b).

FIGURE 3.1 – Principes du seuillage et du masquage. Source de l'image originale : [Kumar et al., 2012].

3.1.2 Algorithme général

Nous décrivons à présent le procédé général du simulateur de cubes hyperspectraux de cubes tavelés. Les cubes hyperspectraux sont des images tridimensionnelles dont deux dimensions repré-

sentent l'information spatiale de la feuille représentée, et la troisième l'information spectrale. Un tel cube peut être vu comme la concaténation selon l'axe des canaux d'images bidimensionnelles représentant l'aspect de la feuille aux différentes longueurs d'onde. Nous avons appelé *tranches spectrales* chacune de ces images.

Pour créer un cube donné, nous avons utilisé comme source une image de feuille saine. Nous avons dupliqué cette image autant de fois que le nombre de tranches spectrales que nous souhaitions simuler dans le cube. Nous avons généré une distribution spatiale de lésions de tavelure, c'est-à-dire un masque représentant les positions des lésions sur la feuille. Nous avons appliqué sur les tranches un contraste entre les zones saines et les zones tavelées. Ce contraste dépendait de la longueur d'onde, et était calculé selon des spectres mesurés expérimentalement. L'algorithme 3.1 présente cette procédure plus formellement. Cet algorithme est illustré figure 3.2 [4]. Les étapes indiquées en bleu dans l'algorithme et encadrées en bleu dans la figure sont détaillées dans les sections suivantes de ce chapitre. Notons que la sortie de cet algorithme était un cube, soit une image en trois dimensions. Nous désignons chacun des éléments de ces cubes par le mot *voxel*.

Algorithme 3.1 : Créer un cube hyperspectral à partir d'une image de feuille.

Entrées : une image RVB d'une feuille saine I, les spectres expérimentaux de zones saines et tavelées de feuilles R_S et R_T de longueur n_λ, la dimension spatiale du cube à générer d.

Convertir I en niveau de gris.
Redimensionner I à la dimension $d \times d$ pixels.
Créer par seuillage de la couleur verte un masque $M_{feuille}$ où les pixels blancs correspondent aux pixels appartenant à la feuille dans I.

/* Distribution spatiale des lésions de tavelure. */
Générer une distribution spatiale de lésions $M_{lésions}$ où les pixels blancs correspondent aux emplacements des lésions simulées.
Générer un masque de lésions de tavelure $M_{tavelure}$ en masquant $M_{lésions}$ par $M_{feuille}$.
Créer l'image M_{sain} comme le résultat de l'opération $M_{feuille}$ - $M_{tavelure}$.
Créer l'image $I_{tavelure}$ contenant uniquement les pixels tavelés de la feuille en masquant I par $M_{tavelure}$.
Procéder similairement pour obtenir I_{sain}.

/* Création des tranches spectrales. */
pour $\lambda \in [1, n_\lambda]$ **faire**
  Calculer la tranche spectrale du cube hyperspectral simulé à la longueur d'onde λ : $I_\lambda = M_{sain} . R_S[\lambda] + M_{tavelure} . R_T[\lambda]$.
fin

/* Création du cube. */
Concaténer les images I_λ suivant l'axe des canaux pour former le cube C.

Sortie : le cube hyperspectral C de la feuille I, avec des lésions de tavelure simulées, de dimension $d \times d \times n_\lambda$ voxels.

Nous détaillons à présent les étapes suivantes : collecte des images des feuilles saines, génération des distributions spatiales des lésions, acquisition expérimentale des spectres.

4. La quasi-intégralité des algorithmes de ce manuscrit sont au moins partiellement illustrés afin de faciliter la compréhension de ceux-ci. La figure correspondante sera indiquée entre parenthèses dans le texte après la première mention de l'algorithme.

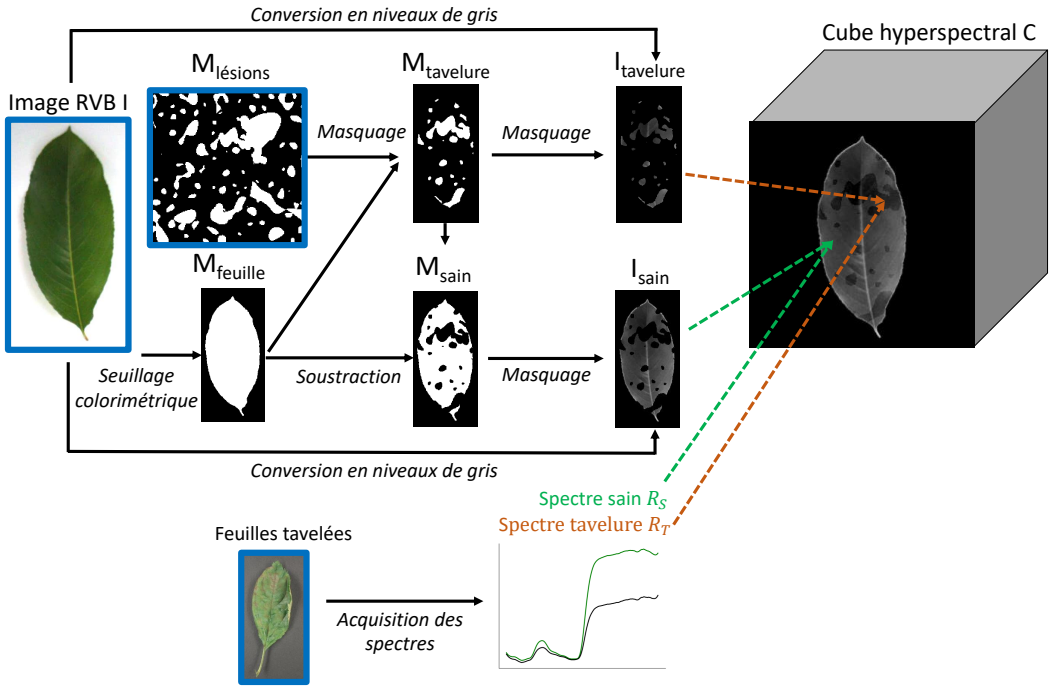


FIGURE 3.2 – Illustration de l'algorithme 3.1. Pour les étapes de masquage, la flèche étiquetée « masquage » indique le masque employé et la flèche vide l'image masquée. Pour l'étape de soustraction, la flèche étiquetée « soustraction » indique l'image de laquelle nous soustrayons et la flèche vide l'image que l'on soustrait. Les étapes encadrées en bleu sont détaillées dans les sections suivantes de ce chapitre. Source de l'image I : [Kumar et al., 2012].

3.1.3 Collecte des images des feuilles saines

Nous avons construit les cubes hyperspectraux à partir d'images de feuilles saines issues du jeu de données Leafsnap [Kumar et al., 2012]. Leafsnap est une application pour mobile qui propose la détermination automatique de l'espèce d'un arbre à partir de la photographie de l'une de ses feuilles. L'application, téléchargée plus d'un million de fois [5], permet ainsi la classification de 184 espèces présentes dans l'hémisphère Nord. Ses auteurs ont validé son fonctionnement sur un jeu de données éponyme créé pour l'occasion, pour lequel ils ont autorisé l'utilisation libre par la communauté scientifique. Le jeu a par exemple été téléversé sur Kaggle afin de servir de base à des compétitions d'apprentissage automatique [6].

Ce jeu de données comprend plus de 30 000 images RVB de feuilles acquises en conditions contrôlées : les feuilles sont isolées sur un fond uniforme clair, à l'intérieur. Environ la moitié d'entre elles ont été acquises via un scanner : les feuilles y sont aplaties. Nous nous sommes concentrés sur l'autre moitié du jeu qui contenait des images de feuilles posées sur un support, où l'acquisition avait été réalisée avec un capteur RVB standard. En effet, nous avons estimé que les repliements et les courbures naturelles de la feuille dans la profondeur étaient des caractéristiques précieuses pour s'approcher de l'aspect de feuilles en conditions réelles. Le jeu de données « champ » était subdivisé en plusieurs dossiers correspondant aux différentes espèces d'arbres. Parmi celles-ci, l'espèce *Malus pumila* (aussi connue sous le nom de *Malus domestica*) est celle à laquelle nous avons porté le plus grand intérêt car elle correspondait aux pommiers majoritairement présents en Europe et à ceux cultivés dans les serres de l'IRHS. Cependant, le nombre d'images de *Malus pumila* était trop faible pour conduire une tâche d'apprentissage profond. Nous avons donc élargi notre jeu de données à l'intégralité des feuilles du genre *Malus*. À l'issue de cette sélection, le jeu de données que nous avons conservé contenait 3000 feuilles. Nous avons désigné ce jeu comme le

5. https://play.google.com/store/apps/details?id=plant.identification.snap.
6. https://www.kaggle.com/xhlulu/leafsnap-dataset.

« jeu Leafsnap ». La figure 3.1 (a) présente une des images de ce jeu.

3.1.4 Distribution spatiale des lésions

Nous avons généré pour chaque feuille un masque de tavelure où les pixels blancs représentaient les positions des lésions ($M_{tavelure}$ dans la figure 3.2). En observant l'aspect des lésions sur des images expérimentales de feuilles tavelées [Oerke et al., 2011; Benoit et al., 2016], nous avons constaté que leurs formes pouvaient être approximées par des taches ovoïdes (figure 3.3). Ces formes s'expliquaient par le mode de développement de *V. inaequalis* : la densité du parasite était très forte à ses emplacements d'incursion sous le cuticule, mais son développement approximativement isotrope à partir de ces foyers menait à ces lésions en forme d'auréole.

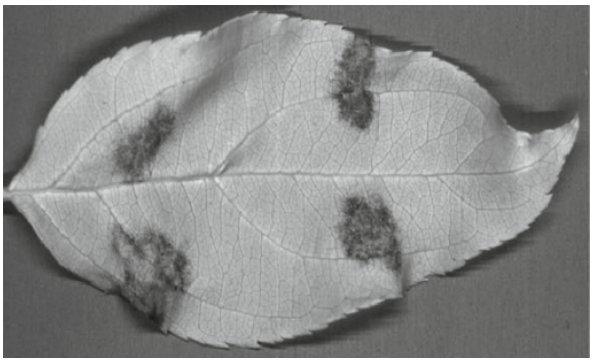


FIGURE 3.3 – Une acquisition IR d'une feuille tavelée à J14 après l'inoculation. Source : [Benoit et al., 2016].

Nous présentons maintenant le procédé de génération des lésions pour une feuille donnée. Le *pipeline* que nous avons mis au point est décrit dans l'algorithme 3.2 (figure 3.4). Il est adapté de travaux que nous avons mené pour une autre application en sciences végétales [Douarre et al., 2018a].

Algorithme 3.2 : Créer un masque de taches ovoïdes.

Entrées : la dimension de l'image à créer $d_1 \times d_2$ pixels, le diamètre du cercle du masque fréquentiel r, la valeur du seuillage binaire t.

Créer une image de dimension $d_1 \times d_2$ pixels dont les valeurs des pixels sont tirées indépendamment et uniformément dans l'intervalle $[0, 255]$.
Calculer la magnitude de la transformée de Fourier de cette image.
/* Conserver uniquement certaines fréquences de l'image. */
Masquer cette transformée de Fourier avec un masque noir à l'exception d'un cercle centré de rayon r.
Calculer la transformée de Fourier inverse du résultat.
Appliquer à l'image obtenue un seuillage binaire avec un seuil t.

Sortie : un masque de taches ovoïdes de dimension $d_1 \times d_2$ pixels.

De plus, nous avons constaté que les tailles des lésions suivaient pour beaucoup de feuilles une distribution multi-échelle. En effet, les lésions de tavelure s'étendaient avec le temps, et il était courant que des lésions d'âges différents soient présentes sur une même feuille. En conséquence, les lésions semblaient générées par un processus qui menait à différentes distributions de tailles, et il nous a paru important de respecter cette caractéristique. Afin de simuler cet effet, nous avons généré pour chaque feuille un masque composite « multi-échelle », somme de plusieurs masques de lésions « mono-échelle » générés selon l'algorithme 3.2 avec des distributions de tailles différentes

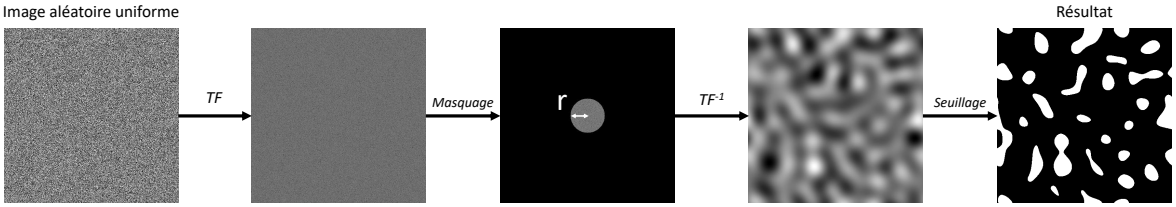


FIGURE 3.4 – Illustration de l'algorithme 3.2. « TF » indique l'opération de transformée de Fourier et « TF^{-1} » l'opération de transformée de Fourier inverse. Le masque employé pour l'opération de masquage n'est pas explicité dans cette figure.

(figure 3.5).

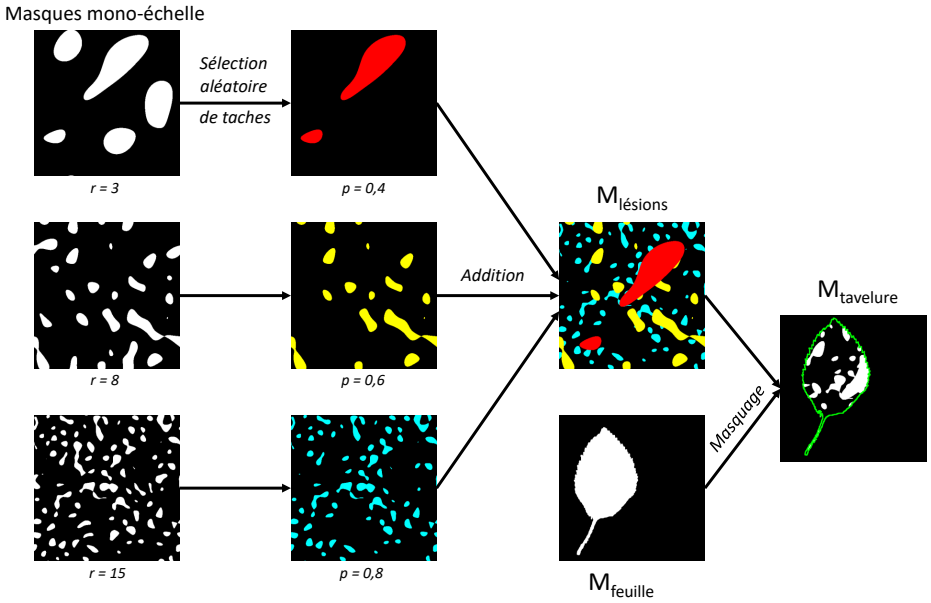


FIGURE 3.5 – Algorithme de génération de masques de lésions multi-échelle. Dans la première colonne, la valeur r indique le rayon du masque fréquentiel utilisé dans l'algorithme 3.2 pour générer le masque mono-échelle. Dans la deuxième colonne, la valeur p inique la proportion de taches conservées lors de l'étape de sélection de taches. Dans la deuxième et la troisième colonne, des couleurs ont été ajoutées à titre illustratif, mais les pixels colorés sont blancs dans les faits. Dans la quatrième colonne, la forme de la feuille, en vert, est indiquée elle aussi à titre illustratif.

Nous avons commencé par générer trois masques de lésions mono-échelle, correspondant à des tailles de lésions que nous avons identifiées comme « grandes », « moyennes », et « petites » (figure 3.5, première colonne). La distribution des tailles des taches dans les images mono-échelle dépendait du rayon r du cercle du masque fréquentiel, ainsi que de la valeur du seuil t dans l'algorithme 3.2. Nous avons fixé le seuil t à la valeur renvoyée par le seuillage automatique d'Otsu [Otsu, 1979] multipliée par un coefficient empiriquement posé comme 1,3 afin de se ramener à un processus paramétré par une seule valeur. Nous avons alors pu faire varier les tailles de taches générées en fixant différentes valeurs pour le rayon r du masque fréquentiel. Ensuite, pour chacun de ces masques mono-échelle, nous avons conservé aléatoirement une certaine proportion p des taches (figure 3.5, deuxième colonne). Les valeurs des paramètres r et p, fixées empiriquement, sont précisées figure 3.5.

Par la suite, nous avons combiné ces masques mono-échelle via une opération « OU logique » pour obtenir un masque de lésions multi-échelle noté $M_{lésions}$ (figure 3.5, troisième colonne). La figure 3.6 présente, pour un exemple de masque multi-échelle, la distribution de tailles des taches dans les trois masques mono-échelle qui le composent. À l'échelle du jeu de données, les aires

des taches variaient entre 10 et 70 000 pixels carrés, avec un mode à 28 pixels carrés. Nous avons par la suite masqué ce masque multi-échelle par le masque correspondant à la feuille considérée (figure 3.5, quatrième colonne). Nous avons appelé l'image obtenue $M_{tavelure}$.

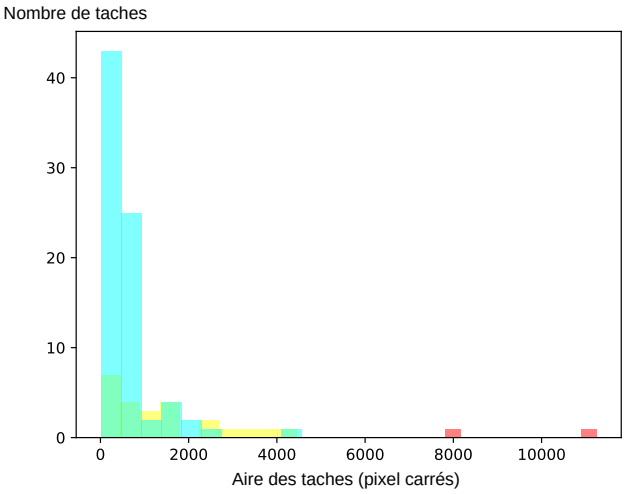


FIGURE 3.6 – Un histogramme des tailles des taches contenues dans chaque masque mono-échelle avant que ceux-ci ne soient combinés dans un masque multi-échelle. Le code couleur correspond à celui employé figure 3.5.

3.1.5 Acquisition expérimentale des spectres

Nous avons incorporé à la simulation des spectres expérimentaux de portions saines et tavelées de vraies feuilles. Il y avait à l'IRHS des cultures en serre de plants de *Malus pumila* qui avaient été inoculés avec *V. inaequalis*. Nous avons prélevé dix de ces feuilles quatorze jours après l'inoculation, une durée après laquelle les lésions de tavelure étaient extrêmement sévères [Oerke et al., 2011]. Nous avons acquis leurs cubes hyperspectraux via une caméra hyperspectrale à balayage. La caméra était de marque HySpex, modèle VNIR 1024 (figure 3.7), qui permettait d'acquérir 160 bandes spectrales sur une gamme de 400 à 1000 nm, correspondant donc aux gammes visible et IR-A. L'acquisition a été réalisée en intérieur, les lumières de la salle éteintes. La seule source de lumière était produite par la caméra HySpex elle-même. Les feuilles étaient posées sur un fond uniforme. Deux tranches spectrales d'une acquisition sont présentées figure 3.8.

Nous avons par la suite modifié la luminosité du cube acquis afin de simuler une acquisition en lumière extérieure au lieu de la lumière produite par la caméra HySpex. Lors de leur acquisition par cette caméra, les feuilles étaient accompagnées d'un « Spectralon ». Il s'agissait d'un matériau réfléchissant quasi-uniformément l'ensemble des longueurs d'onde qu'il recevait. Nous avons normalisé chaque tranche du cube grâce à ce Spectralon, c'est-à-dire que nous avons divisé l'intensité des pixels de chaque tranche par l'intensité moyenne du Spectralon à cette tranche, afin de simuler une illumination à intensité spectralement constante [7]. Puis, nous avons simulé une illumination extérieure en utilisant le spectre D65 [Schanda, 2007], un spectre produit par la CIE représentant une illumination standard en extérieur un jour de soleil en Europe. Nous avons normalisé ce spectre D65 à 1, puis pour chaque tranche du cube, nous avons multiplié l'intensité de tous les pixels par la valeur de ce spectre à la longueur d'onde correspondante.

7. Cette normalisation n'est pas celle représentée dans les sous-figures (b) et (d) de la figure 3.8. La normalisation qui est présentée dans cette figure est simplement une division de tous les pixels des images (a) et (c) par la valeur maximale des images puis une multiplication par 255 afin d'améliorer l'expérience de visualisation.

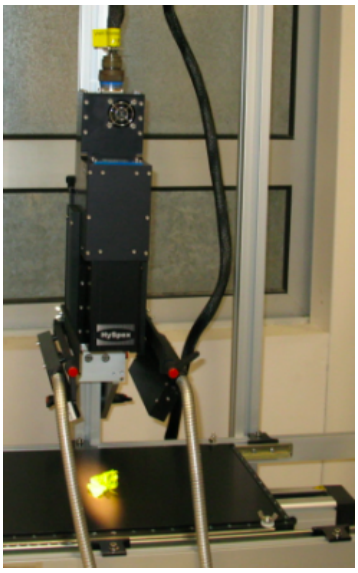
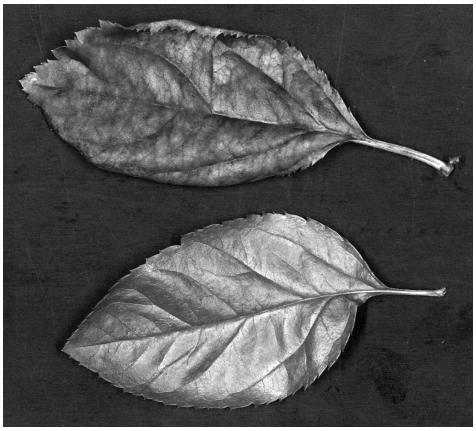
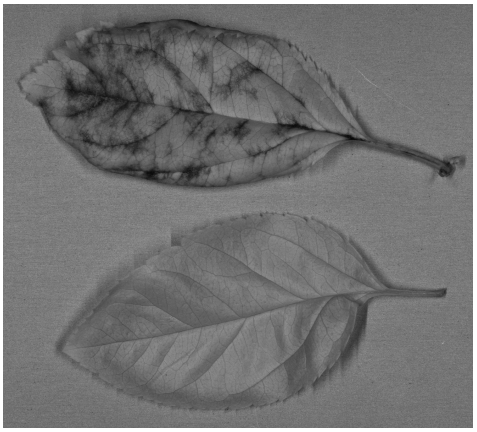


FIGURE 3.7 – La caméra HySpex du LARIS durant l'acquisition du cube hyperspectral d'une feuille. Source : LARIS.

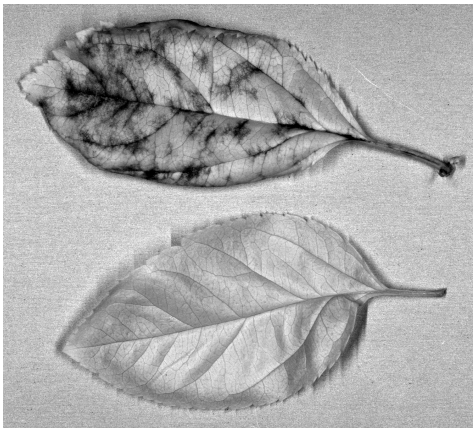
(a) Longueur d'onde : 560nm (couleur verte du domaine visible).



(b) Image (a) normalisée à des fins illustratifs.



(c) Longueur d'onde : 956nm (IR-A).

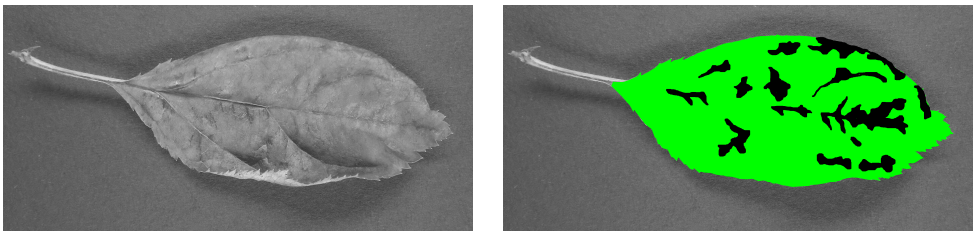


(d) Image (c) normalisée à des fins illustratifs.

FIGURE 3.8 – Deux tranches spectrales d'un cube hyperspectral acquis avec la caméra HySpex. La feuille située en haut est tavelée, celle en bas est saine (pour comparaison).

A ce stade, nous possédions dix cubes hyperspectraux de feuilles tavelées. Nous avons ensuite séparé les spectres correspondant aux zones saines et tavelées des feuilles. Pour une des tranches

de chacun des dix cubes hyperspectraux acquis, nous avons annoté les pixels correspondant aux zones tavelées (figure 3.9). Cette distinction entre zones saines et tavelées nous a permis, en moyennant tranche par tranche les valeurs de tous les pixels sains d'un côté, tavelés de l'autre, puis en moyennant ces résultats sur les dix feuilles, d'obtenir les spectres expérimentaux moyens des zones saines et tavelées, sur la gamme 400-1000nm (figure 3.10). Nous notons ces spectres de réflectance R_S et R_T respectivement.



(a) Image originale. (b) Image annotée.

FIGURE 3.9 – La tranche spectrale annotée d'un des dix cubes acquis. Les pixels verts de l'image (b) correspondent aux zones saines et les pixels noirs aux zones tavelées.

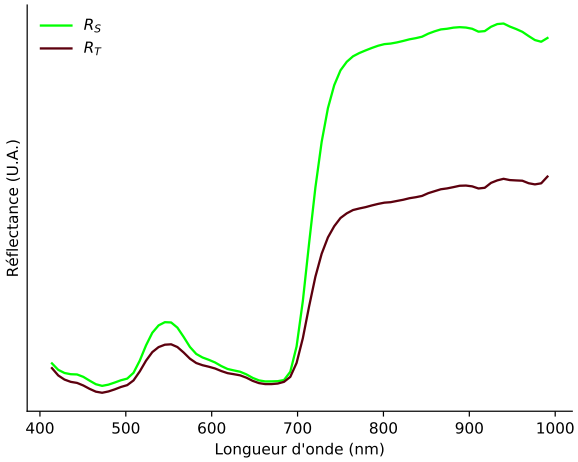


FIGURE 3.10 – Les spectres expérimentaux pour les zones saines et tavelées, R_S et R_T.

Représenter différents stades d'infection

Les feuilles dont nous nous sommes servies afin de déterminer ces spectres étaient très fortement infectées. Afin de pouvoir évaluer les algorithmes de détection de lésions sur des configurations plus difficiles, nous avons par ailleurs simulé des spectres représentant des infections à des stades intermédiaires, c'est-à-dire datant d'un nombre de jours inférieur à quatorze et par conséquent plus facilement traitables. Nous avons introduit une variable que nous appelons *sévérité*, qui prenait une valeur dans $[0,1]$ et qui représentait la gravité de l'infection. Nous avons généré des spectres intermédiaires $R_{sévérité}$ calculés à partir des spectres expérimentaux de la façon suivante :

$$R_{sévérité} = R_S + sévérité(R_T - R_S). \tag{3.1}$$

Ainsi, $R_0 = R_S$, $R_1 = R_T$, et les autres valeurs de sévérité permettaient de représenter des infections intermédiaires. Le spectre $R_{0,5}$ est tracé figure 3.11 pour illustration.

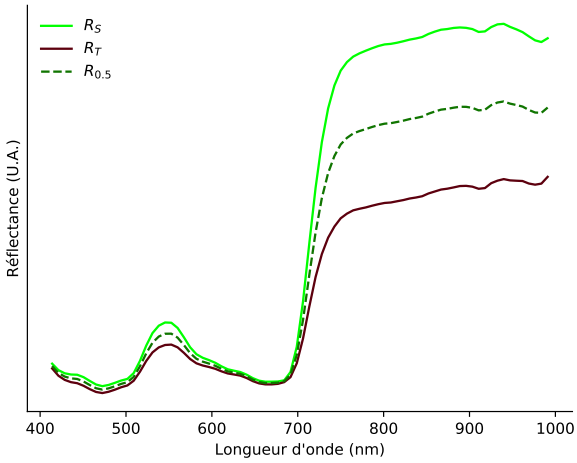
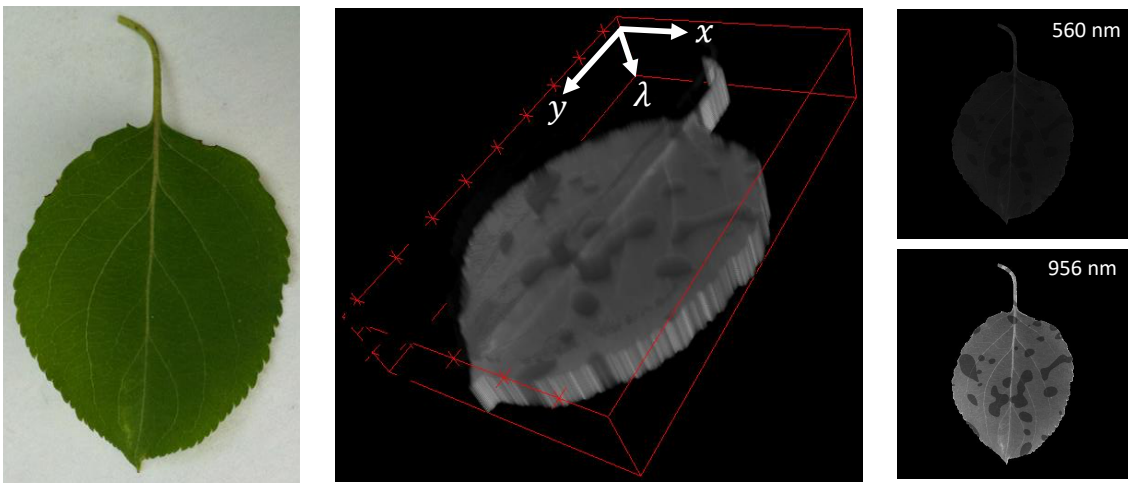


FIGURE 3.11 – Un spectre simulé de sévérité 0,5 ($R_{0,5}$), comparé aux spectres expérimentaux R_S et R_T.

3.1.6 Exemple d'un cube simulé

La figure 3.12 présente un exemple de cube généré avec le simulateur présenté dans cette section, pour une sévérité de 1. Ces cubes hyperspectraux constituaient les représentations spectrales des scènes dans lesquelles nous souhaitions procéder à une détection de lésions. Nous avons par la suite développé un simulateur de CTIS, qui, à partir d'un cube hyperspectral, en créait une image CTIS.



(a) Image originale du jeu Leafsnap.

(b) Visualisation 3D du cube, avec x et y indiquant les dimensions spatiales et λ la dimension spectrale.

(c) Deux tranches extraites du cube correspondant aux longueurs d'onde présentées figure 3.8.

FIGURE 3.12 – Exemple d'un cube hyperspectral généré avec le simulateur de cube hyperspectraux de feuilles tavelées.

3.2 Simulateur de CTIS

3.2.1 Modèle discrétisé de l'action du réseau de diffraction

L'action du réseau de diffraction au cœur du CTIS, qui créé des projections bidimensionnelles à partir d'une scène hyperspectrale tridimensionnelle, a été décrite au chapitre 2. Nous avons créé un simulateur basé sur une discrétisation de ce fonctionnement (figure 3.13 pour le cas de deux ordres). L'image CTIS, puisqu'acquise sur un CCD au nombre de détecteurs finis, était intrinsèquement de nature discrète, de dimension $d \times d$ pixels. Dans ce modèle, nous avons aussi discrétisé la scène hyperspectrale, que nous représentons sous la forme d'un cube hyperspectral de dimension $d \times d \times n_\lambda$ voxels. Ce cube servait d'entrée au simulateur.

L'ordre 0 de l'image CTIS était créé en sommant toutes les tranches spectrales de ce cube. Concernant les ordres supérieurs, une projection de l'image CTIS était constituée de l'ensemble des tranches spectrales où chaque tranche était décalée spatialement d'une valeur fixe par rapport à la précédente en suivant l'axe de la projection. Si nous nous représentons le cube hyperspectral comme un jeu de cartes à jouer parfaitement empilées, alors une projection d'un des ordres supérieurs correspondaient à ce même jeu partiellement étalé, comme lorsqu'un magicien demande au public de choisir une carte lors d'un tour.

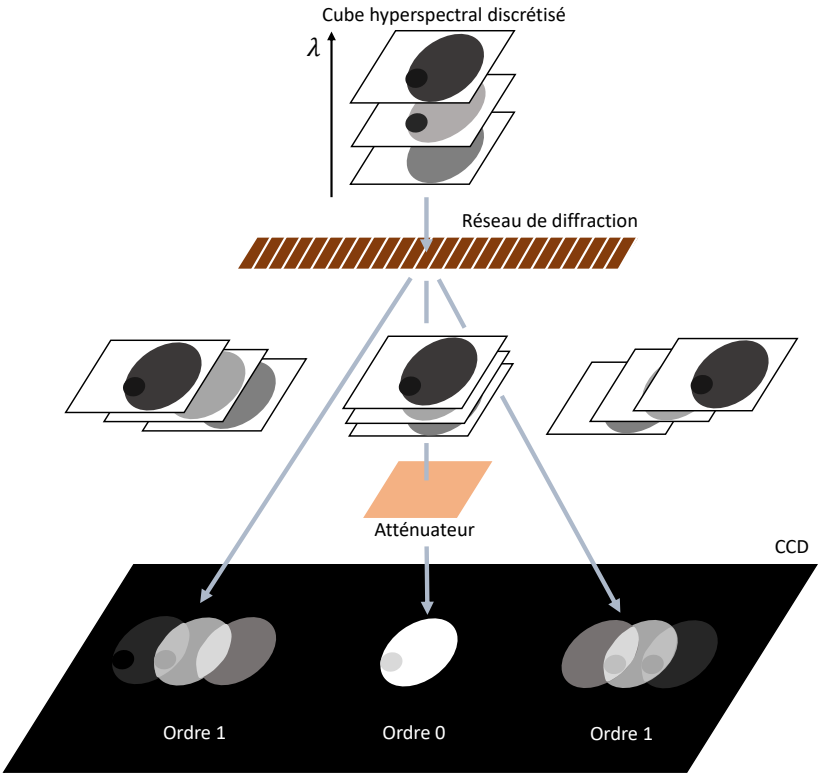
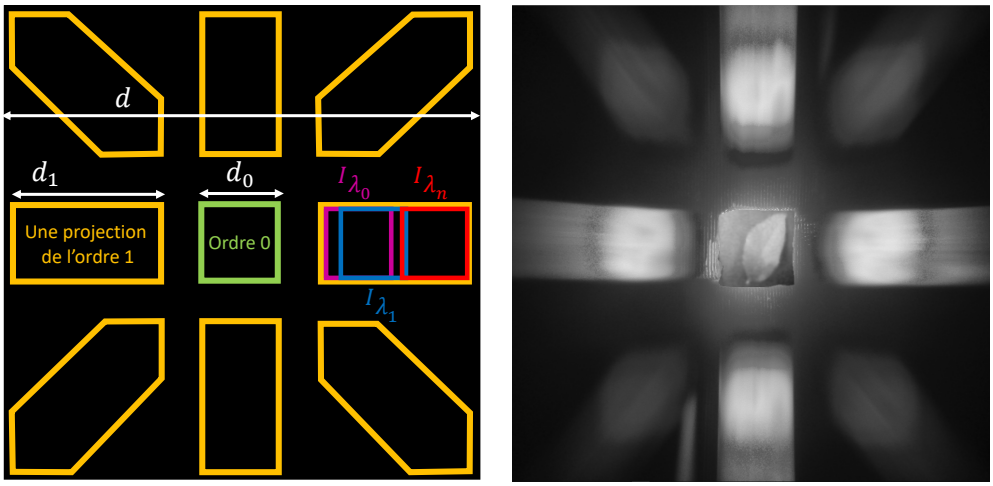


FIGURE 3.13 – Schéma représentant le modèle CTIS discrétisé. L'élément « atténuateur » est discuté à la section 3.2.2.

Ce modèle discret du CTIS permettait de mettre en évidence une limite du système. Dans une projection donnée d'un ordre supérieur, le décalage spatial entre deux tranches spectrales consécutives était très faible par rapport à la taille de ces tranches : il y avait un recouvrement important, et donc une perte d'information, entre les différentes tranches spectrales (figure 3.13,« CCD »). La présence de plusieurs projections dans des directions différentes permettait de partiellement compenser cette limitation lors de la reconstruction du cube à partir de l'image.

Nous avons reproduit ce fonctionnement discret en modélisant l'action du CTIS pour un cube donné en entrée. L'étape principale de ce modèle consistait à ajouter dans une image les tranches spectrales du cube aux localisations déterminées par le fonctionnement optique du système (figure 3.14). Ces localisations étaient déterminées par la *géométrie* du modèle, c'est-à-dire le nombre d'ordres acquis et la disposition générale des projections. Dans le système optique d'un CTIS, cette géométrie dépendait de la structure du réseau de diffraction employé. Nous avons basé notre simulateur sur un type de réseau proche de celui employé par Carbon Bee, proposant notamment deux ordres de diffraction (l'ordre 0 et l'ordre 1). Nous pouvons cependant constater figure 3.14 que la géométrie du modèle ne correspondait pas exactement à celle permise par la caméra Carbon Bee. Le réseau de diffraction utilisé dans cette dernière menait à une différence de luminosité entre les projections cardinales et diagonales de l'ordre 1. Nous avons préféré modéliser un réseau où toutes les projections au sein des ordres étaient de luminosité égale, en accord avec les réseaux étudiés dans la littérature (section 2.2.2).



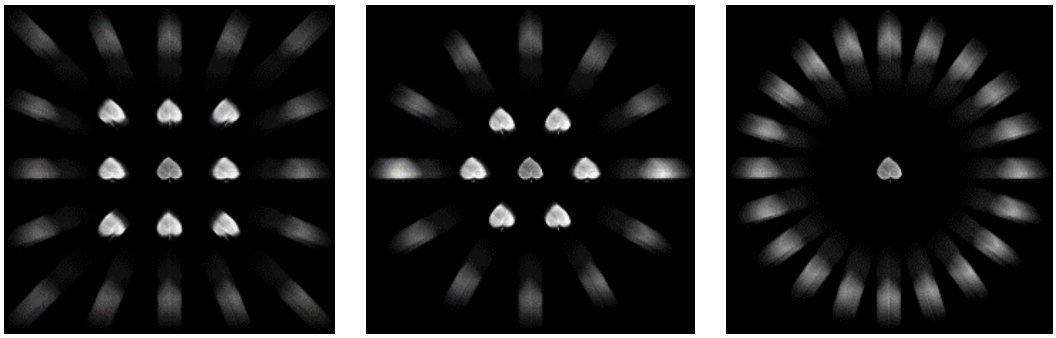
(a) Modèle. (b) Image acquise par le capteur Carbon Bee.

FIGURE 3.14 – Modèle de CTIS discret que nous avons implémenté, comparé à une acquisition CTIS du capteur Carbon Bee. Dans ce modèle, la position de l'ordre 0 est représentée en vert, et toutes les autres projections, représentées en jaune, appartiennent à l'ordre 1. Sur la projection de droite, nous avons explicité le placement de trois tranches spectrales pour illustration.

Il est intéressant de noter que cette littérature faisait état d'une grande variété de réseaux employés dans des systèmes CTIS [Hagen et al., 2006]. Nous avons intégré à notre modèle un paramètre régissant la géométrie obtenue, incluant ainsi la possibilité de simuler les géométries les plus fréquemment implémentées. Quelques exemples des sorties possibles du modèle sont présentés figure 3.15. Nous n'avons pas employé ces géométries pour la simulation d'images dans ce manuscrit.

Par ailleurs, nous avons inclus la possibilité de modifier la résolution spatiale de l'image obtenue (d_0 dans la figure 3.14), c'est-à-dire la taille de l'ordre 0 et de chacune des tranches spectrales dans l'ordre 1. Dans le système optique du CTIS, cette résolution est définie par le choix de la lentille d'objectif, du diaphragme de champ et de la distance entre ces deux éléments (figure 2.7).

La résolution spectrale, elle, c'est-à-dire la longueur des projections de l'ordre 1 (d_1 dans la figure 3.14), était fixe car conditionnée par les lois de la diffraction. Dans un montage CTIS, il était possible d'augmenter cette résolution « vers l'extérieur », c'est-à-dire en rapprochant les bords extérieurs des projections des ordres supérieurs vers les bords du CCD. Cette augmentation était dépendante du pouvoir dispersif du réseau de diffraction, et particulièrement de la distance entre ses fentes. Cependant, l'étirement de ces projections « vers l'intérieur », c'est-à-dire rapprocher le bord intérieur des projections vers le centre du CCD, était limité par la physique de la diffraction. En



(a) Géométrie : rectangulaire. Ordres : 3.

(b) Géométrie : hexagonale. Ordres : 3.

(c) Géométrie : radiale. Ordres : 2.

FIGURE 3.15 – Quelques exemples de géométries de CTIS que notre modèle pouvait générer. Nous avons suivi pour la description de ces géométries la nomenclature de l'étude de [Hagen et al., 2006]. Notre modèle correspondait à une géometrie rectangulaire avec deux ordres.

effet, au passage d'un réseau de diffraction, l'angle de dispersion de la lumière variait linéairement avec sa longueur d'onde (cf. section 2.1.2). Notons λ_{min} et λ_{max} les longueurs d'onde minimale et maximale acquises dans un cube hyperspectral en entrée du CTIS . Si la tranche spectrale correspondant à λ_{max} était située à une distance l_{max} du centre du CCD, alors par linéarité de la diffraction, la tranche λ_{min} était nécessairement située à une distance $l_{min} = \frac{\lambda_{min}}{\lambda_{max}} l_{max}$ du centre du CCD. Ainsi, la portion de l'image située entre le centre du CCD et l_{min} était nécessairement vide. Nous avons considéré dans notre simulateur CTIS que le réseau de diffraction utilisé était optimal et donc que les projections de l'ordre 1 étaient « étirées » au maximum vers l'extérieur. Dans ces conditions, la résolution spectrale de l'image CTIS d_1 était fixe pour une géométrie et une gamme spectrale $[\lambda_{min}, \lambda_{max}]$ donnée.

Dans toutes les simulations de ce manuscrit, la taille de l'image CTIS d était fixe. Ainsi, augmenter d_0 permettait une plus grande résolution spatiale mais menait par ailleurs à un recouvrement plus important entre les tranches spectrales dans les projections de l'ordre 1. En somme, l'aspect des projections était dans notre modèle entièrement déterminé par la géométrie des projections, la taille de l'ordre 0 et la gamme spectrale acquise.

3.2.2 Intégration d'aspects matériels

Le placement des tranches spectrales du cube aux différentes positions dans l'image CTIS, décrit dans la section précédente, constitue le cœur du simulateur que nous avons développé. Afin d'en améliorer le réalisme, nous avons incorporé par ailleurs des aspects liés au matériel (*hardware* en anglais) du capteur. Premièrement, nous avons modélisé la sensibilité spectrale du CCD (cf. section 2.1.1). La sensibilité spectrale varie peu d'un capteur à un autre, aussi nous sommes nous basés sur les travaux des auteurs de [Andor, 2020; Spring and Davidson, 2020] pour calculer celle d'un CCD typique, que nous avons notée S_T (figure 3.16). Dans la plupart des caméras vendues dans le commerce, un filtre de Bayer ainsi qu'un filtre anti-IR sont apposés au CCD afin de limiter la sensibilité « effective » au domaine du visible. Pour son intégration dans le cadre d'un CTIS, nous modélisons une sensibilité spectrale sans ces filtres. Nous avons intégré cette sensibilité spectrale au modèle en multipliant chacune des tranches du cube donné en entrée par la valeur $S_T[\lambda]$ correspondant à la longueur d'onde qu'elles représentaient, avant de les placer dans l'image CTIS.

Nous avons par ailleurs simulé le *gain* du capteur : il s'agissait d'une valeur traduisant l'efficacité du CCD à convertir des photons en tension électrique, indépendamment de la longueur d'onde. Nous avons introduit pour cela un paramètre g par lequel l'intégralité des pixels de l'image CTIS

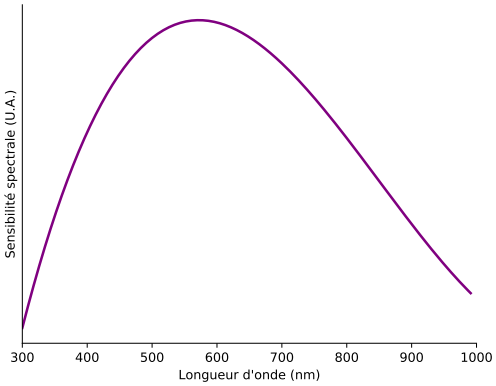
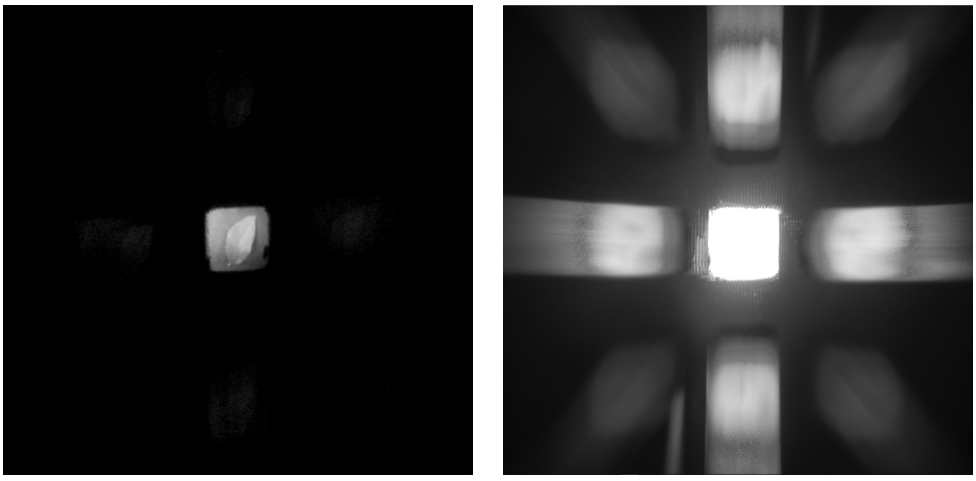


FIGURE 3.16 – Sensibilité spectrale du CCD modélisée d'après les données de [Andor, 2020; Spring and Davidson, 2020].

étaient multipliés. De plus, nous avons affiné la simulation de ce gain en introduisant dans notre modèle l'effet d'un *atténuateur* optique. Il s'agissait d'un filtre qui absorbait une quantité d'énergie lumineuse fixe, indépendamment de la longueur d'onde. Nous avons simulé son placement sur le trajet de la lumière qui se dirigeait vers l'ordre 0 (figure 3.13). En effet, puisque l'ordre 0 correspondait à la somme de toutes les longueurs d'onde, la projection concernée était toujours bien plus lumineuse que les projections de l'ordre 1 où les tranches spectrales n'étaient que partiellement superposées. En conséquence, lors de l'obtention d'une l'image CTIS par un système sans atténuateur, il y avait un grand risque que les deux ordres soient dans des dynamiques très différentes : soit l'ordre 0 était saturé, soit l'ordre 1 était trop faible (figure 3.17). Un atténuateur placé devant l'ordre 0 permettait de réduire l'intensité de la lumière correspondant à cet ordre, et d'utiliser ainsi un gain plus élevé qui permettait d'acquérir les deux ordres dans une dynamique similaire. Nous avons simulé cet atténuateur par un paramètre $a < 1$ par lequel tous les pixels correspondant à l'ordre 0 étaient multipliés.



(a) Ordre 1 sous-exposé (b) Ordre 0 sur-exposé

FIGURE 3.17 – Deux acquisitions réalisées avec un gain différent. En l'absence d'atténuateur, il est courant que les deux ordres ne puissent pas être acquis sur une même image.

Concrètement, pour un jeu de cubes donné en entrée du simulateur CTIS, nous avons implémenté un calcul automatique des valeurs a et g qui étaient identiques pour toutes les images CTIS obtenues. Ces valeurs étaient fixées de manière à ce que les deux ordres aient la plus grande gamme

dynamique possible, c'est-à-dire qu'en moyenne sur le jeu d'images créé, les pixels de chaque ordre soient compris dans une gamme le plus proche possible de [0,255]. Ce calcul automatique était proche de ce qui était réalisé dans la réalité. La plupart des caméras, même grand public, implémentent un gain automatique [Fowler, 2004] afin d'augmenter la gamme dynamique des images acquises. Quant à un atténuateur automatique de l'ordre 0, il fallait imaginer une atténuation numérique implémentée spécifiquement pour un capteur CTIS, où les positions des projections auraient été connues par le système. Un tel système était proche d'implémentations réelles, dans la mesure où certaines caméras hyperspectrales basées sur le CTIS faisaient intervenir l'utilisateur dans la définition des zones des ordres [Salazar-Vazquez and Mendez-Vazquez, 2020]. Cette normalisation automatique nous permettait en outre une grande souplesse quant aux opérations de normalisation dans le reste du *pipeline* : possibilité de traiter des cubes dans n'importe quelle dynamique, d'utilisation de spectres et de sensibilités spectrales non normalisés, etc.

3.2.3 Algorithme général

L'intégralité du modèle de CTIS prenant en compte les apports liés au matériel présentés à la section précédente est décrit dans l'algorithme 3.3 (figure 3.18). Cet algorithme prenait en entrée un cube hyperspectral de taille arbitraire, et simulait l'entièreté du banc optique CTIS (figure 2.7), y compris la baisse de résolution spatiale causée par la lentille d'objectif et le diaphragme de champ.

Algorithme 3.3 : Création d'une image CTIS à partir d'un cube hyperspectral.

Entrées : un cube hyperspectral C de dimension $d \times d \times n_\lambda$ voxels, la gamme spectrale acquise $[\lambda_{min}, \lambda_{max}]$, la géométrie des projections, la taille de l'image CTIS en sortie d, la résolution spatiale d_0, la sensibilité spectrale du CCD S_T, l'atténuation de l'ordre 0 a, le gain g.

Créer une image noire I_{CTIS} de dimension $d \times d$ pixels.

/* Lentille d'objectif et diaphragme de champ : création d'un cube hyperspectral spatialement réduit. */

pour $\lambda \in [1, n_\lambda]$ **faire**
  $Cs[:,:,\lambda] = C[:,:,\lambda]$ redimensionné spatialement à $d_0 \times d_0$ pixels.
fin

/* Ordre 0. */

Calculer la somme des tranches spectrales $I_0 = \sum_{\lambda=1}^{n_\lambda} Cs[:,:,\lambda] S_T[\lambda]$.
Multiplier I_0 par a.
Placer I_0 au centre de I_{CTIS}.

/* Ordre 1 : Pour chaque projection p de l'ordre 1, soient (x_{p0}, y_{p0}) les coordonnées du point de la projection le plus proche du centre de I_{CTIS}, et $[x_{p1}, y_{p1}]$ les coordonnées du point le plus éloigné. Ces coordonnées sont définies par la géométrie des projections ainsi que la gamme spectrale acquise. */

pour $\lambda \in [1, n_\lambda]$ **faire**
  Calculer la tranche spectrale λ : $I_\lambda = Cs[:,:,\lambda] S_T[\lambda]$.
  pour $p \in$ *Projections* **faire**
    Placer I_λ dans I_{CTIS} à l'emplacement $(x_{p0} + {}^{\lambda}/_{n_\lambda}(x_{p1} - x_{p0}), y_{p0} + {}^{\lambda}/_{n_\lambda}(y_{p1} - y_{p0}))$.
  fin
fin

/* Application du gain. */

Multiplier I_{CTIS} par g.

Sortie : l'image CTIS I_{CTIS} du cube C, de dimension $d \times d$ pixels.

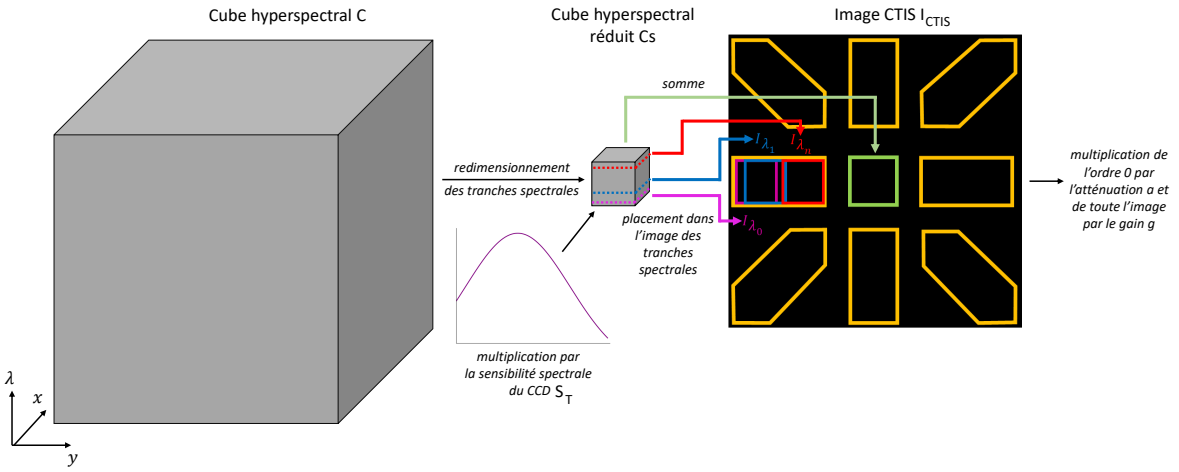


FIGURE 3.18 – Illustration de l'algorithme 3.3.

3.2.4 Détermination de la matrice d'action du système

L'image CTIS est une représentation indirecte du cube hyperspectral qu'il est par la suite courant de reconstruire. Cette reconstruction est basée sur la matrice H qui décrit l'action du réseau de diffraction (section 2.2.3). Nous avons intégré le calcul de cette matrice H à notre simulateur afin que les images CTIS générées puissent servir pour la reconstruction des cubes hyperspectraux.

Le réseau de diffraction caractérisé par H agissait sur le cube hyperspectral de la scène réduit spatialement par la lentille d'objectif et le diaphragme de champ, noté Cs. Nous avons écrit Cs et I_{CTIS} comme des vecteurs unidimensionnels, et défini H, de taille nombre de pixels de I_{CTIS} × nombre de voxels de Cs, selon l'équation suivante :

$$I_{CTIS} = HCs. \tag{3.2}$$

La figure 3.19 illustre l'action de H.

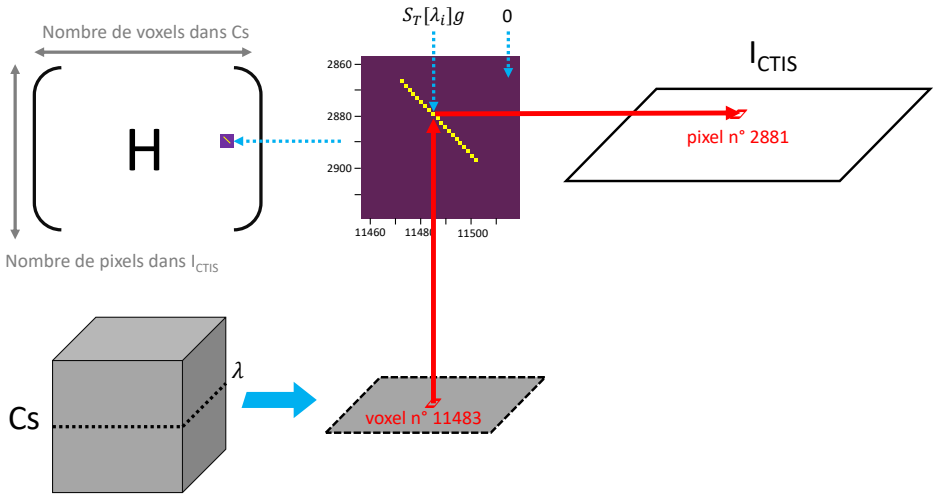


FIGURE 3.19 – Illustration de la signification de la matrice H. La matrice est représentée en haut à gauche, et un agrandissement d'une petite section de celle-ci est représenté en haut au centre sous la forme d'un carte de chaleur de couleur « plasma ». L'élément H[2881, 11483] représente la contribution du voxel 11483 de Cs, représenté en bas à gauche, au pixel 2881 de l'image I_{CTIS}, représentée en haut à droite.

Utiliser un modèle simulé nous a permis, contrairement à un CTIS réel, d'obtenir facilement H. En effet, les étapes de l'algorithme 3.3 pour convertir un cube hyperspectral réduit en image

CTIS étaient trivialement transposables en écriture matricielle. Pour les étapes qui nécessitaient de placer des éléments de Cs à des emplacements spécifiques dans I_{CTIS}, il suffisait de fixer à 1 les éléments de H correspondants. Pour les étapes qui consistaient à multiplier des éléments de Cs par des valeurs données, il suffisait de multiplier les éléments correspondants de H par ces valeurs. Notons que la matrice H dépendait uniquement des paramètres du modèle CTIS, mais qu'elle ne variait pas en fonction du contenu de Cs.

3.2.5 Simulateurs RVB et IR

Afin de comparer les performances basées sur des images CTIS à des imageries plus conventionnelles et notamment celles proposées par la caméra Carbon Bee, nous avons également implémenté des simulateurs de capteurs RVB et IR. L'image RVB simulée était composée de trois canaux correspondant aux trois sous-filtres d'un filtre de Bayer. Chaque canal est créé en sommant les tranches du cube pondérées par la sensibilité spectrale du CCD pour ce canal. Ce procédé est décrit plus formellement par l'algorithme 3.4.

Algorithme 3.4 : Création une image RVB à partir d'un cube hyperspectral.

Entrée : un cube hyperspectral C de dimension $d \times d \times n_\lambda$ voxels, les sensibilités spectrales pour chaque canal de couleur du CCD S_R, S_V et S_B, le gain g.

pour *canal* $\in \{R, V, B\}$ **faire**
  Calculer l'image $I_{canal} = \sum_{\lambda=1}^{n_\lambda} C[:,:,\lambda] S_{canal}[\lambda]$.
fin
Concaténer I_R, I_V et I_B suivant l'axe des canaux pour obtenir I_{RVB}.
Multiplier I_{RVB} par le gain g.

Sortie : l'image RVB I_{RVB} du cube C de dimension $d \times d \times 3$ pixels.

Nous avons modélisé les sensibilités spectrales d'un CCD pour chaque canal de couleur en nous basant sur les données expérimentales des auteurs de [Pagnutti et al., 2017] qui ont étudié les sensibilités d'un CCD associé à un filtre de Bayer. Nous avons généré les sensibilités correspondant aux canaux bleu, vert, rouge, comme des discrétisations de courbes gaussiennes générées selon les paramètres suivants : $S_B \sim \mathcal{N}(460, 30)$, $S_V \sim \mathcal{N}(540, 30)$, $S_R \sim \mathcal{N}(625, 30)$ (figure 3.20).

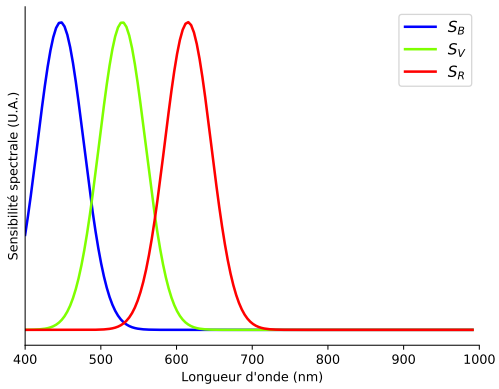


FIGURE 3.20 – Sensibilités spectrales d'un CCD avec filtre de Bayer modélisées d'après les données de [Pagnutti et al., 2017].

L'image IR simulée était composée d'un seul canal qui correspondait simplement à la tranche spectrale 760nm, une longueur d'onde montrée comme étant optimale pour la détection de tavelure

[Benoit et al., 2016]. Les images IR étaient en outre, comme les images RVB, pondérées par une valeur de gain qui permettait de les exprimer dans la plus grande dynamique possible.

3.3 Création des jeux simulés

Nous décrivons à présent la façon dont nous avons créé les jeux d'images simulées en couplant le simulateur de cubes de feuilles tavelées et ceux des capteurs. Nous avons d'abord créé un jeu de cubes hyperspectraux de feuilles tavelées à partir des 3000 images du jeu Leafsnap. Pour chaque image, nous avons tiré une valeur binaire aléatoire. Dans la moitié des cas, nous avons généré des cubes avec des lésions de tavelure, suivant l'algorithme 3.1. Pour l'autre moitié, nous n'avons pas généré de tavelure. Concrètement, la procédure était la même que l'algorithme 3.1, à la différence que l'image M_{tavelure} (figure 3.2) était fixée comme une image noire. Nous avons fixé la dimension spatiale des cubes (d dans l'algorithme 3.1) à 512 pixels, une longueur proche de la taille moyenne des images du jeu Leafsnap. Quant à la dimension spectrale de ces cubes (n_λ dans l'algorithme 3.1), nous avons discrétisé les spectres expérimentaux à 80 valeurs. Les cubes générés étaient donc de dimension $512 \times 512 \times 80$ voxels.

Tous les cubes étaient par la suite convertis en image CTIS en utilisant le simulateur éponyme. Les paramètres du simulateur, choisis pour être aussi proches que possibles du capteur présent dans la caméra Carbon Bee, sont précisés dans le tableau 3.1. Un exemple d'image CTIS simulée est présentée figure 3.21.

Paramètre	Valeur
d	512 pixels
d_0	60 pixels
λ_{min}	400 nm
λ_{max}	1000 nm
Géométrie	Rectangulaire, deux ordres

TABLEAU 3.1 – Paramètres fixés pour le simulateur CTIS.

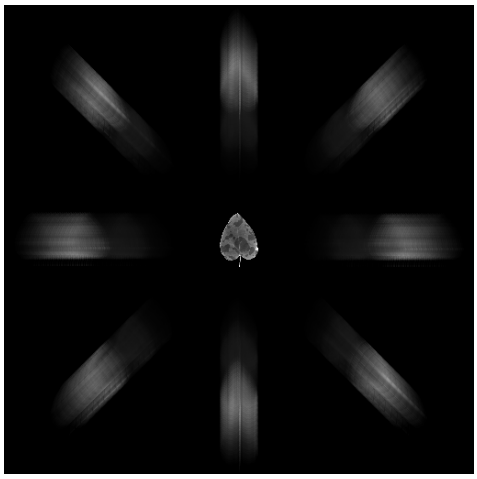


FIGURE 3.21 – Un exemple d'image généré en couplant le simulateur de cubes hyperspectraux de feuilles tavelées et le simulateur CTIS.

Conformément à l'action optique du CTIS, les tranches des cubes hyperspectraux subissaient un redimensionnement à la dimension $d_0 \times d_0$ pixels avant que leurs tranches ne soient placées dans l'image CTIS. Ce redimensionnement n'était pas sans conséquence pour la discriminabilité

des spectres sains et tavelés, en particulier dans l'ordre 0. La figure 3.22 présente la différence moyenne entre les spectres sains et tavelés au sein des cubes en fonction de la taille à laquelle ceux-ci étaient réduits. Nous pouvons observer que pour des valeurs de d_0 typiques des systèmes CTIS, c'est-à-dire entre 5 et 20% de la taille du CCD d (entre 25 et 100 pixels pour notre cas), cette discriminabilité diminuait fortement.

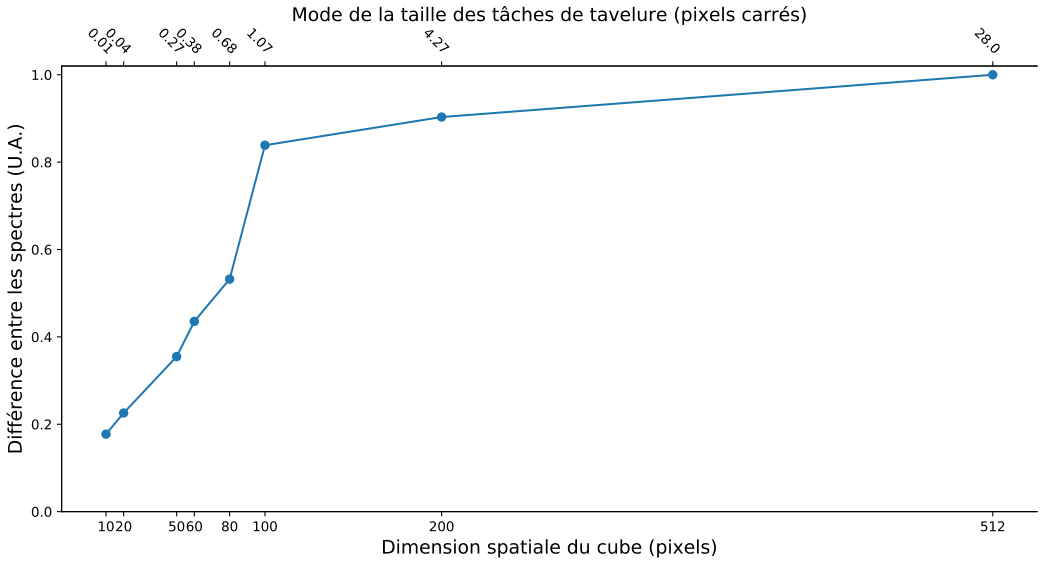


FIGURE 3.22 – Étude de la discriminabilité des spectres sains et tavelés. L'axe des abscisses du bas indique la dimension spatiale des cubes Cs. L'axe des abscisses du haut indique l'aire moyenne des taches de tavelure. L'axe des ordonnées indique la différence en norme 2 entre les spectres tavelés et sains, moyennée sur l'ensemble des cubes générés. La différence spectrale entre les cubes non redimensionnés, c'est-à-dire de taille 512 × 512 pixels, est fixée arbitrairement à 1.

Le jeu final était donc constitué de 3000 images CTIS de feuilles tavelées de dimension 512 × 512 pixels, associées chacune à une classe « saine » ou « tavelée ». La tâche d'apprentissage associée était donc une classification binaire. Afin de pouvoir évaluer la performance des algorithmes d'apprentissage dans des configurations plus ou moins difficiles, nous avons fait varier dans le simulateur de cubes hyperspectraux le paramètre *sévérité* (section 3.1.5) afin de générer plusieurs jeux de 3000 images avec des contrastes sain-tavelure différents. Nous avons noté $D_{CTIS}^{sévérité}$ les jeux ainsi créés. Nous avons créé de façon analogue les jeux $D_{RVB}^{sévérité}$ et $D_{IR}^{sévérité}$. La figure 3.23 résume les différents jeux créés. Les valeurs de sévérité que nous avons fixées pour la création de ces jeux sont : {0, 0,1, 0,12, 0,14, 0,16, 0,18, 0,20, 0,22, 0,25, 0.27, 0,3, 0,4, 0,5, 0,6, 0,7, 1}. Nous avons séparé chacun de ces jeux en trois blocs d'entraînement, validation et test selon une proportion de 60%, 20% et 20%. Nous avons noté D_{CTIS}, D_{RVB} et D_{IR} les ensembles de jeux correspondant à toutes les valeurs de sévérité.

3.4 Conclusion

Nous avons présenté dans ce chapitre deux simulateurs innovants. Le premier permettait une génération de cubes hyperspectraux de feuilles tavelées à partir d'images de feuilles saines et de spectres expérimentaux. Le principe d'un modèle d'imagerie est d'abstraire suffisamment le processus de génération d'images réalistes pour nous permettre de le paramétriser simplement. Cette paramétrisation permet alors de contrôler dans les images créées la variabilité qui est d'intérêt pour l'expérience que nous souhaitons mener. Ainsi, ce simulateur était volontairement simple, en particulier dans son processus de génération de lésions de tavelure. Nous y avons intégré l'information qui nous semblait être la plus cruciale pour l'évaluation d'un spectromètre, c'est-à-dire le contraste entre zones saines et tavelées, contraste qui dépendait de la longueur d'onde. Nous espérons ainsi

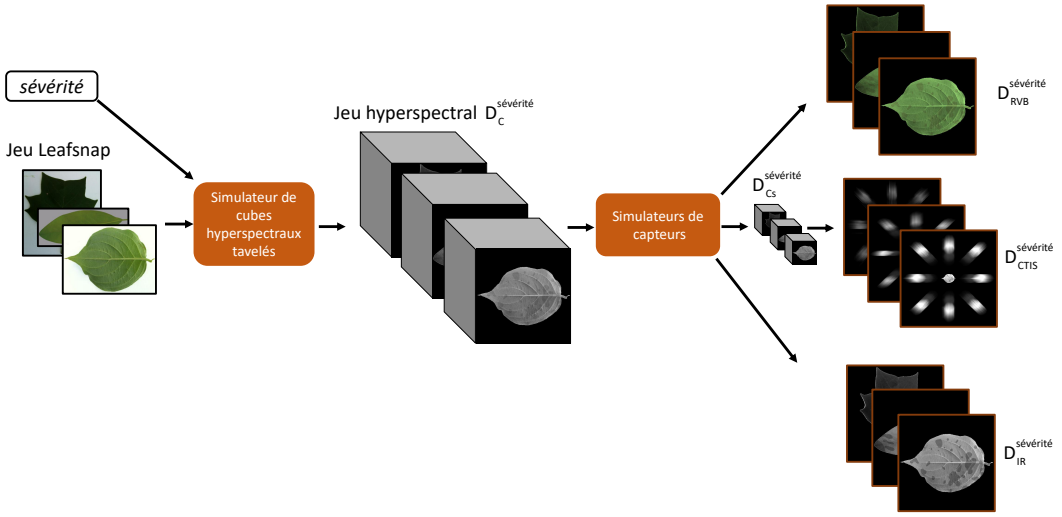


FIGURE 3.23 – Résumé des différents jeux simulés créés grâce aux deux simulateurs présentés dans ce chapitre.

que ce simulateur puisse être utilisé pour d'autres maladies de plantes dont les symptômes se manifestent par des taches foliaires, telles que le mildiou ou des rouilles [Mahlein et al., 2012].

Pour des simulations plus fines, nous pourrions étoffer le modèle en incluant certains traits biologiques à un niveau de réalisme plus élevé. En particulier, nous avons utilisé pour toutes les longueurs d'onde d'un cube donné une image unique de feuille correspondant à son acquisition dans le domaine visible. Ainsi, les tranches correspondant au domaine de l'IR étaient représentées par des images acquises en lumière visible. Nous pourrions enrichir le modèle d'informations texturales provenant de différentes longueurs d'onde, d'autant plus que nous possédions les cubes entiers de dix feuilles expérimentales. Il faut cependant noter que l'application de motifs provenant d'images réelles pour créer des images simulées constitue un champ de recherche en lui-même [Efros and Freeman, 2001; Kopf et al., 2007; Dong et al., 2019]. Dans la même veine, nous pourrions introduire une variation des formes des lésions de tavelure en fonction de la longueur d'onde représentée. Par ailleurs, la modélisation de la sévérité de l'infection était très simple et pourrait être significativement complétée. Nous avons simulé une progression de l'infection qui se traduisait par une évolution du spectre simultanément pour toutes les longueurs d'onde, ce que nous savions être une approximation de la réalité biologique. Pour s'approcher davantage des signatures visuelles de lésions de tavelure réelles, il faudrait suivre l'évolution des spectres de zones saines et tavélées sur des feuilles que nous étudierions entre leur inoculation et le J14. La simulation des distributions spatiales des lésions de tavelure pourrait aussi être affinée. Nous pouvons par exemple constater sur des images réelles des concentrations préférentielles du champignon au niveau des nervures (figure 5.15) de la feuille, par simple effet de gravité (figure 3.9). Nous pourrions aussi simuler l'effet « gaussien » de la concentration en agent infectieux. Enfin, pour rapprocher les cubes simulés de mesures réelles, nous pourrions intégrer une variabilité dans les spectres simulés pour un cas d'étude donné afin de traduire les différences qui existent entre les représentants distincts d'une même espèce. Certaines études comme celles de [Delalieux et al., 2009a] et de [Hu et al., 2008] ont présenté l'ensemble des acquisitions qui ont été réalisées pour les différentes instances d'un système hôte-pathogène donné, permettant ainsi de prendre la mesure de la variabilité spectrale inter-individus.

Concernant le simulateur CTIS, il s'agissait du premier modèle qui permettait la création d'images éponymes à partir de cubes hyperspectraux arbitraires. Nous pensons que ce simulateur représentait fidèlement les principes du spectromètre. Bien que nous ayons peu fait varier ses paramètres dans ce travail, nous avons volontairement intégré à ce simulateur la possibilité de

nombreuses personnalisations : géométrie des projections, sensibilité spectrale du CCD, résolution spatiale ; afin que ce simulateur puisse être librement utilisé et adapté par la communauté scientifique. Nous avons publié le code de ce simulateur en code source ouvert (cf. section Valorisations).

Il faut noter cependant que contrairement à tout système optique réel, les images qu'il permettait de générer étaient vierges d'aberrations optiques et autres imperfections de la réalité, comme l'illustre la figure 3.24. Il était donc possible que les images obtenues en l'état masquaient une partie de la difficulté d'exploitation des images CTIS réelles. Il aurait été bénéfique d'intégrer au moins partiellement ces irrégularités au fonctionnement du simulateur, ou tout au moins d'évaluer à quel point il était nécessaire d'intégrer chacune d'entre elles afin que des résultats d'apprentissage obtenus sur ces données simulées soient véritablement transposables en situation réelle. Nous avons mené à la section 4.5 et à l'annexe A des premiers travaux dans cette veine.

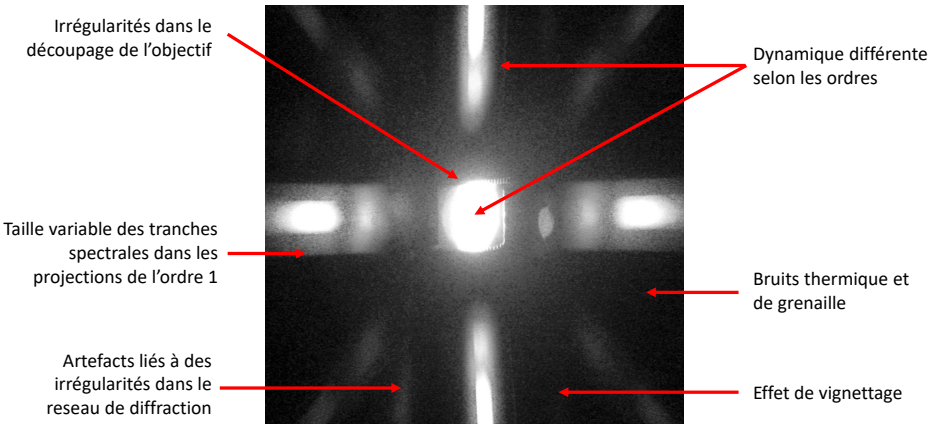


FIGURE 3.24 – Illustration des irrégularités que l'on observe dans des images CTIS réelles par rapport au modèle théorique que nous avons développé. L'acquisition présentée a été réalisée en conditions lumineuses faibles, ce qui a nécessité d'augmenter fortement le gain du capteur. Aussi le bruit observé est-il plus prégnant que dans d'autres images CTIS présentées dans ce manuscrit, telle que la figure 2.8. Source : acquisition avec la caméra Carbon Bee.

Nous avons généré grâce à ces deux simulateurs une quantité suffisante de cubes hyperspectraux de feuilles tavelées, qui de plus représentaient une gamme de difficulté suffisamment variée, pour qu'ils puissent servir de base d'évaluation de la viabilité d'un algorithme d'apprentissage automatique. Dans le chapitre suivant, nous présentons l'étude que nous avons conduite concernant l'exploitation du signal CTIS en nous basant sur les jeux créés dans ce chapitre.

Chapitre 4

Apprentissage comprimé sur images CTIS

Ce chapitre présente les apprentissages menés sur les signaux produits par le CTIS. Les systèmes d'imagerie computationnelle comme le CTIS permettent d'accroître les capacités des imageurs en exploitant la puissance de calcul des ordinateurs pour former une image enrichie à partir de mesures indirectes. Néanmoins, cette reconstruction du signal est souvent très chronophage et imprécise [Arce et al., 2013]. Nous avons commencé par implémenter cette reconstruction ainsi qu'une méthode d'apprentissage automatique travaillant sur les cubes reconstruits, en nous basant sur une littérature dédiée aux méthodes d'apprentissage pour l'imagerie hyperspectrale. Cependant, nous avons considéré qu'il existait aussi, grâce aux progrès récents des réseaux de neurones, une possibilité de mener des apprentissages directement dans l'espace de mesures du CTIS, c'est-à-dire en exploitant directement les images éponymes plutôt que les cubes reconstruits. Cette approche non-standard s'inscrivait dans le jeune champ de l'apprentissage comprimé (*compressed learning* en anglais). Nous avons ainsi mené dans ce chapitre plusieurs apprentissages de cette manière, d'abord avec un réseau de neurones générique, puis avec une architecture développée spécifiquement pour tirer parti des images CTIS de façon optimale. Nous avons comparé les résultats obtenus avec ceux de la voie « classique » ainsi qu'avec d'autres types d'imagerie.

Sommaire

4.1 Un cadre commun pour les expérimentations 66
4.1.1 Une architecture établie dans la communauté 66
4.1.2 Un protocole d'entraînement standard 67
4.2 Une performance de référence : l'apprentissage sur cubes reconstruits 69
4.2.1 Une reconstruction réussie mais chronophage 69
4.2.2 Une performance d'apprentissage de référence 71
4.3 L'apprentissage comprimé : une alternative viable pour le CTIS 74
4.3.1 Une tendance nouvelle en imagerie computationnelle 74
4.3.2 Des performances d'apprentissage proches de la référence 75
4.3.3 Une exploitation sub-optimale des entrées 76
4.4 Une architecture dédiée aux images CTIS 78
4.4.1 Des traitements spécifiques aux ordres de l'image 78
4.4.2 Une nette amélioration de la performance 81
4.4.3 Une étude par ablation met en lumière les apports de CTIS-Net 82
4.5 Les performances subsistent malgré l'ajout de bruit 85
4.6 Une plus grande polyvalence par rapport aux imageurs classiques 87
4.7 Une influence forte des paramètres optiques 88
4.8 Conclusion 89

4.1 Un cadre commun pour les expérimentations

Nous présentons dans ce chapitre de nombreux résultats d'apprentissages profonds menés grâce à des réseaux de neurones. Afin que les résultats obtenus soient comparables entre les expériences, nous avons fixé une méta-architecture de réseau standard sur laquelle toutes se sont basées ainsi qu'un protocole d'apprentissage identique pour toutes.

4.1.1 Une architecture établie dans la communauté

L'architecture du réseau de neurones sur laquelle nous nous sommes basés est celle du réseau VGG (*Visual Geometry Group*) [Simonyan and Zisserman, 2014b]. Nous avons choisi ce réseau d'une part car c'était un des réseaux les plus performants au moment de sa création. Cette architecture avait notamment remporté la catégorie « classification et localisation » du défi de reconnaissance visuelle à grande échelle d'ImageNet (*ImageNet Large Scale Visual Recognition Challenge* en anglais, ou ILSVRC) [Russakovsky et al., 2015], la tâche publique servant de point de comparaison entre les architectures (*benchmark* en anglais) la plus populaire dans le domaine de la vision par ordinateur [8]. D'autre part, son architecture était simple, composée uniquement de blocs de couches convolutives avec des activations ReLU, suivies de couches *max-pool*, ainsi qu'une série de couches FC. Bien qu'il existait des architectures de CNN proposées depuis la sortie de VGG qui obtenaient des performances supérieures sur les *benchmarks* les plus populaires [Szegedy et al., 2015; He et al., 2016; Tan and Le, 2019], cette simplicité nous paraissait être un avantage pour deux raisons. Premièrement, au moment des travaux présentés dans ce manuscrit, les couches qui composaient VGG avaient été étudiées suffisamment en profondeur (voir par exemple pour les couches convolutives les travaux de [Zeiler and Fergus, 2014], pour les activations ReLU les travaux de [He et al., 2015], pour les couches FC les travaux de [Montufar et al., 2014]) pour que les principes de fonctionnement de cette architecture nous semblent être solidement ancrés et validés par la communauté. Ces couches sont d'ailleurs toujours les piliers des architectures des réseaux créés aujourd'hui. Deuxièmement, cette architecture simple rendait les résultats que nous avons obtenu plus généralisables, car relativement agnostiques à l'ossature de l'architecture et ne dépendant pas de la présence de couches créées plus récemment et spécifiques à certaines architectures.

L'architecture de VGG [9] est présentée en figure 4.1. Tous les termes qui y sont employés ont été présentés à la section 1.2.1. Cette architecture était sujette à plusieurs hyperparamètres. Pour en fixer les valeurs optimales, nous avons effectué une recherche par grille (*grid search* en anglais), c'est-à-dire que nous avons mené des entraînements avec toutes les combinaisons possibles d'hyperparamètres sur des gammes prédéfinies. La combinaison qui permettait la performance la plus élevée sur les blocs de validation était retenue. Nous avons utilisé pour blocs de validation celui de D^1_{CTIS} et celui du jeu de cubes reconstruits D^1_{Cr} (présenté à la section 4.2.1). Les hyperparamètres optimaux étaient identiques pour les deux blocs. Le résultat de cette recherche en grille est présenté dans le tableau 4.1.

Comme l'indique la liste des hyperparamètres étudiés (première colonne du tableau 4.1), la recherche concernant certains d'entre eux n'a été que partielle. Nous n'avons pas fait varier le nombre de couches au sein de chaque bloc convolutif, en conservant par défaut celui de VGG. Par ailleurs, nous avons conservé le doublement du nombre de filtres à chaque bloc convolutif. Enfin,

8. http://image-net.org/challenges/LSVRC/2014/results.

9. Pour être précis, les auteurs de [Simonyan and Zisserman, 2014b] présentaient dans leurs travaux plusieurs variations architecturales de VGG. Nous avons utilisé la configuration dénotée « D » dans cette étude, et connue par la suite comme « VGG-16 », en référence au nombre de couches de poids qui composaient l'architecture. Cette configuration était celle qui avait été retenue comme l'architecture « standard » de VGG par la communauté. En particulier, il s'agissait, avec « VGG-19 », de celle que les auteurs avaient utilisé pour l'ILSVRC 2014 et qui était donc devenue *de facto* une architecture à laquelle les réseaux créés par la suite tels que ResNet [He et al., 2016] se sont comparés. De plus, VGG-16 a servi de base à d'autres architectures neuronales, et en particulier SegNet [Badrinarayanan et al., 2017], que nous présentons au chapitre 5. Dans tout ce manuscrit, nous avons écrit simplement « VGG » pour faire référence à VGG-16.

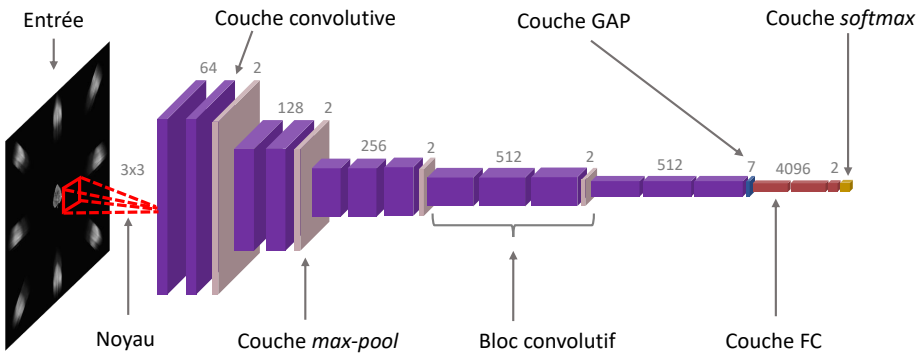


FIGURE 4.1 – Architecture du réseau VGG [Simonyan and Zisserman, 2014b]. Les nombres inscrits au-dessus des couches indiquent le nombre de filtres pour les couches convolutives (constant au sein d'un même bloc convolutif), le facteur de redimensionnement pour les couches *max-pool*, la dimension de sortie pour la couche GAP, le nombre de neurones pour les couches FC . La hauteur et la largeur des couches représentent les dimensions spatiales de leur entrée.

Hyperparamètre	Valeur dans VGG	Valeurs testées	Valeur optimale
Nombre de blocs convolutifs	5	[2-5]	3
Nombre de filtres dans la première couche convolutive	64	{32, 64, 128}	64
Nombre de neurones dans chaque couche FC	4096	{512, 1024, 2048, 4096}	1024
Taille des noyaux	3	{3,5,7}	3

TABLEAU 4.1 – Résultats de la recherche par grille des hyperparamètres architecturaux de VGG, sur les blocs de validation de D^1_{CTIS} et D^1_{Cr}.

nous avons conservé tel quel le nombre de couches FC.

Nous pouvons remarquer en comparant les valeurs des hyperparamètres de VGG et celles du réseau optimal (colonnes 2 et 4 du tableau 4.1) que ce dernier possèdait moins de paramètres que VGG, à la faveur d'un nombre réduit de blocs convolutifs et de neurones dans les couches FC. Cette différence s'expliquait par l'écart de complexité entre nos jeux de données et ceux pour lesquels les réseaux standards tels que VGG avaient été conçus. En effet, ces derniers avaient été créés de manière à être compétitifs pour une large gamme de tâches de vision. Pour juger de leur performance, ils avaient été appliqués à des tâches de classification génériques, comme l'ILSVRC, qui nécessitaient la reconnaissance de centaines de classes d'objets différentes dans des environnements variés. Comparativement, les tâches de classification que nous avons exploré dans ce travail étaient beaucoup plus cloisonnées. Il n'était donc pas surprenant que la frontière de décision pour ces tâches soit considérablement moins complexe que pour des tâches plus vastes, et que la capacité du réseau optimal soit plus réduite que celle de VGG. Nous précisons cependant qu'au cours de la recherche par grille, les écarts de performance entre le réseau optimal et d'autres architectures avec une capacité plus élevée étaient relativement faibles. Nous avons plutôt observé une saturation qu'un déclin des performances avec l'augmentation de la capacité, et le choix d'utiliser l'architecture optimale la plus réduite possible était en partie motivée par la réduction des temps d'entraînement des apprentissages. Nous avons noté VGGr (« r » pour « réduit ») l'architecture optimale pour les jeux de données étudiés (figure. 4.2).

4.1.2 Un protocole d'entraînement standard

Nous avons défini un protocole commun à tous les entraînements conduits dans ce manuscrit. Les réseaux étaient *pré-entraînés* sur l'ILSVRC, c'est-à-dire qu'avant tout entraînement, nous avons téléchargé les poids du réseau VGG entraîné sur cette tâche et ces poids servaient d'initialisation en lieu et place d'une distribution aléatoire. Cette technique, connue sous le nom d'apprentissage *par transfert* était très régulièrement implémentée dans les applications d'apprentissage profond

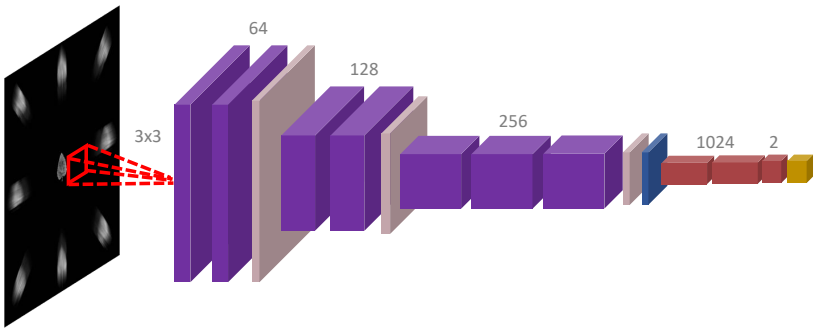


FIGURE 4.2 – Architecture de VGGr. Dans cette figure comme dans les suivantes, nous n'indiquons pas le facteur de redimensionnement des couches *max-pool*, ni la dimension de sortie de la couche GAP, qui sont fixés à deux et sept respectivement pour toutes les architectures employées dans ce manuscrit.

[Yosinski et al., 2014]. Elle permettait de tirer parti de caractéristiques apprises sur une tâche de vision générique, caractéristiques qui étaient supposées utiles comme « point de départ » pour d'autres tâches plus spécifiques [Sharif Razavian et al., 2014]. Parmi ces poids, nous avons conservé uniquement ceux correspondant aux couches convolutives présentes dans VGGr. Ainsi, les transferts de poids que nous avons réalisé n'était que partiel. En particulier, la différence du nombre de blocs convolutifs entre VGG et VGGr entraînait un changement de rôle pour les caractéristiques transférées. Par exemple, celles du troisième bloc de VGG, qui représentaient des caractéristiques intermédiaires pour la classification de l'ILSVRC, étaient directement fournies aux couches FC dans le cas de VGGr. Malgré le caractère incomplet de ce transfert, des expériences préliminaires que nous avions menées nous avaient convaincus de son utilité. Sans ce transfert, les résultats d'apprentissage étaient erratiques. En lançant plusieurs instances d'entraînements avec les mêmes hyperparamètres sur les mêmes jeux de données, nous obtenions tantôt des performances traduisant une classification partiellement réussie, tantôt des cas où le réseau échouait complètement à sa tâche. En d'autres termes, l'écart-type entre les performances était trop important pour tirer des conclusions des apprentissages. Le transfert de poids a permis de réduire fortement ces écarts-types de performance.

Au cours de l'entraînement proprement dit, les images étaient normalisées par une soustraction de leur moyenne et une division par leur écart-type. Elles étaient présentées par lot de 4 au réseau. Cette taille de lot relativement basse par rapport aux applications typiques d'apprentissage automatique s'expliquait par les limitations en mémoire que nous rencontrions à cause de la grande taille des données d'entrée lorsque nous avons traité des cubes hyperspectraux. Nous avons utilisé l'algorithme de descente du gradient stochastique [Kiefer et al., 1952] avec une inertie (*momentum* en anglais) de 0,99. La fonction de coût implémentée était l'entropie croisée [Goodfellow et al., 2016]. Nous avons implémenté les mécanismes de régularisation présentés dans les travaux de [Simonyan and Zisserman, 2014b] pour réduire le surapprentissage, c'est-à-dire des couches de décrochage (*dropout* en anglais) [Srivastava et al., 2014] à 50% entre chaque couche FC ainsi qu'une dégradation des poids (*weight decay* en anglais) avec un coefficient fixé à 5.10^{-4}. La performance du réseau était suivie sur le bloc de validation associé au bloc d'entraînement toutes les deux époques, une époque correspondant au passage de toutes les images à travers le réseau. L'entraînement était arrêté lorsque le coût sur le bloc de validation n'atteignait pas une valeur plus basse que sa valeur jusque-là minimale pendant 30 époques consécutives. Les valeurs des poids ayant mené au coût de validation le plus bas étaient alors conservées. Ce critère d'arrêt, que l'on appelle l'arrêt précoce (*early stopping* en anglais) était un des outils employés pour combattre le surapprentissage [Goodfellow et al., 2016].

Le bloc de test était alors présenté au réseau entraîné, et une métrique de classification était calculée. C'est à la valeur de cette métrique que nous faisons référence lorsque nous indiquons la « performance » du réseau dans ce manuscrit. Nous avons choisi pour métrique le coefficient de

corrélation de Matthews (*Matthews Correlation Coefficient* en anglais, ou MCC) [Matthews, 1975]. Dans un cas de classification binaire entre deux classes « A » et « B », en notant VP le nombre de vrais positifs dans une prédiction, c'est-à-dire le nombre d'objets de la classe « A » prédits comme tels, et que en définissant de façon analogue les valeurs FP, VN et FN, alors le MCC se calculait comme

$$\frac{\mathrm{VP.VN} - \mathrm{FP.FN}}{\sqrt{(\mathrm{VP}+\mathrm{FP})(\mathrm{VP}+\mathrm{FN})(\mathrm{VN}+\mathrm{FP})(\mathrm{VN}+\mathrm{FN})}}. \tag{4.1}$$

Nous avons choisi cette métrique car elle produisait des résultats cohérents dans le cas où les classes étaient déséquilibrées, un cas que nous avons rencontré dans l'étude présentée au chapitre 5. Nous entendons en particulier par « cohérent » la qualité qu'avait cette métrique de ne pas prendre des valeurs hautes lorsque les prédictions du réseau étaient absurdes à cause d'effets dûs aux déséquilibres d'effectif entre les classes. La configuration la plus illustre pour laquelle des métriques plus « classiques » pouvaient échouer de la sorte était la suivante : le cas d'une classification binaire sur un jeu de données comprenant 99% d'images appartenant à une classe « A » et 1% à une classe « B », où une prédiction attribuerait la classe « A » à tous les objets. Une métrique de performance « simple », comme par exemple un ratio entre images bien classées et le nombre d'images total, aurait pris une valeur de 99% pour une prédiction pourtant inutile. Des métriques telles que le score F1 ont été développées pour pallier ces défauts. Le MCC pouvait être vu comme un prolongement du score F1, qui corrigeait quelques cas pathologiques de ce dernier. Cette métrique prenait une valeur dans $[-1,1]$. Une valeur de 1 indiquait une prédiction parfaite tandis qu'une valeur de 0 indiquait une prédiction aléatoire. Des valeurs négatives signalaient une prédiction « inverse » (images de la classe « A » classées en majorité dans la classe « B »), mais nous n'avons observé cette gamme de valeurs dans aucune de nos expériences. En réponse à la stochasticité liée à l'initialisation des réseaux, tous les résultats présentés dans ce manuscrit sont les moyennes et écarts-types de dix répétitions des expériences lancées avec les mêmes hyperparamètres (*runs* en anglais).

Toutes les architectures ont été implémentées en Python 3.6 avec la librairie PyTorch 1.5.1. Tous les entraînements ont été menés sur une carte graphique Nvidia Titan RTX, avec les librairies CUDA 10.2 et cuDNN 7.6.

4.2 Une performance de référence : l'apprentissage sur cubes reconstruits

Pour évaluer l'apport possible d'une classification basée sur le CTIS, nous avons tout d'abord souhaité établir une performance de référence (*baseline* en anglais) par le biais d'un *pipeline* qui représentait l'état de l'art des analyses de signaux produits par un CTIS. Dans tous les travaux menés sur le CTIS, le cube hyperspectral est reconstruit à partir de l'image CTIS. Nous avons donc commencé par implémenter cette étape pour les jeux de données $\mathrm{D_{CTIS}}$.

4.2.1 Une reconstruction réussie mais chronophage

Nous avons implémenté pour chacun des jeux $\mathrm{D_{CTIS}}$ l'algorithme de reconstruction EM, (section 2.2.4) en suivant les directives de l'étude de [Descour and Dereniak, 1995]. Les reconstructions ont été réalisées sur un processeur Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz, commercialisé au même trimestre que les carte graphiques dont nous nous sommes servis pour meenr les apprentissages. Pour une reconstruction donnée, nous avons utilisé comme estimation initiale du cube un cube spatialement identique à l'ordre 0 de l'image CTIS et spectralement uniforme. Nous avons défini comme critère d'arrêt une différence inférieure à 1% entre deux itérations de reconstruction. En notant $\mathrm{Cr}^{(i)}$ la reconstruction du cube à l'itération i et Cr_n le voxel n du cube, alors le critère d'arrêt s'écrivait formellement

$$\frac{\sum_n |\mathrm{Cr}_n^{(i+1)} - \mathrm{Cr}_n^{(i)}|}{\sum_n \mathrm{Cr}_n^{(i)}} \leq 0,01. \tag{4.2}$$

L'algorithme de reconstruction convergeait en six itérations par image en moyenne, chaque itération durant environ 200 millisecondes. Bien que ce procédé pouvait paraître rapide, il fallait remettre en contexte ce temps avec le délai moyen de la prédiction d'un réseau, qui est de l'ordre de quelques dizaines de millisecondes. Ainsi, dans le cadre d'un apprentissage sur cubes reconstruits, le délai de prédiction d'un réseau sur une acquisition CTIS était dû en très grande partie à l'opération de reconstruction du cube.

Nous avons noté D_{Cr} les jeux de données constitués des cubes reconstruits selon ce procédé à partir des images des jeux D_{CTIS}. Pour juger de la qualité de cette étape, nous avons comparé chaque cube reconstruit Cr avec le « vrai » cube Cs associé, c'est-à-dire celui utilisé pour générer l'image CTIS à partir de laquelle Cr avait été reconstruit. Nous avons choisi l'erreur quadratique moyenne comme mesure d'erreur, conformément aux autres travaux qui évaluaient la qualité de la reconstruction de cubes CTIS [Hagen et al., 2006]. En notant N le nombre total du voxels dans ces cubes, alors l'erreur entre une paire de cubes Cr et Cs était calculée comme

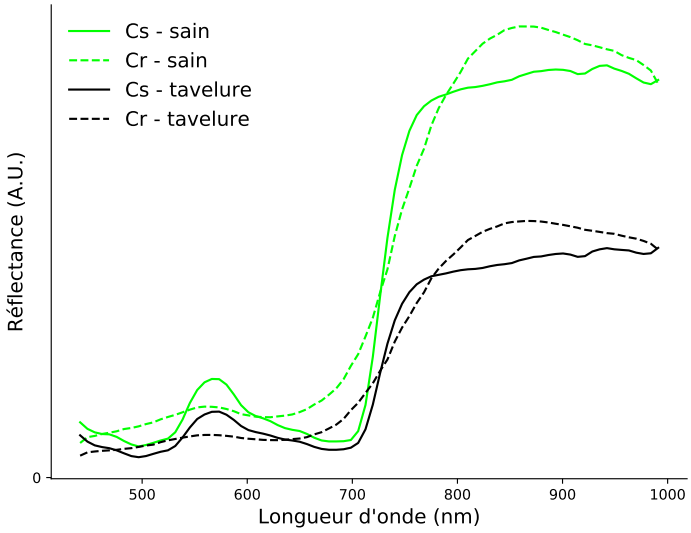
$$\text{erreur} = \sqrt{\frac{1}{\mathrm{N}} \sum_{n} \left(\frac{\mathrm{Cr}_n - \mathrm{Cs}_n}{\mathrm{Cs}_n} \right)^2}. \tag{4.3}$$

L'erreur moyenne sur les cubes reconstruits à partir du jeu D^1_{CTIS} était de 0,392 ± 0,137. Cette valeur était cohérente avec les résultats de la littérature correspondant à la reconstruction d'objets spatialement et spectralement complexes [Hagen and Dereniak, 2008]. Malgré l'écart-type relativement élevé de cette mesure, les forces et faiblesses du procédé de reconstruction étaient très similaires d'un cube à l'autre. Nous présentons les résultats pour une reconstruction typique dans la figure 4.3.

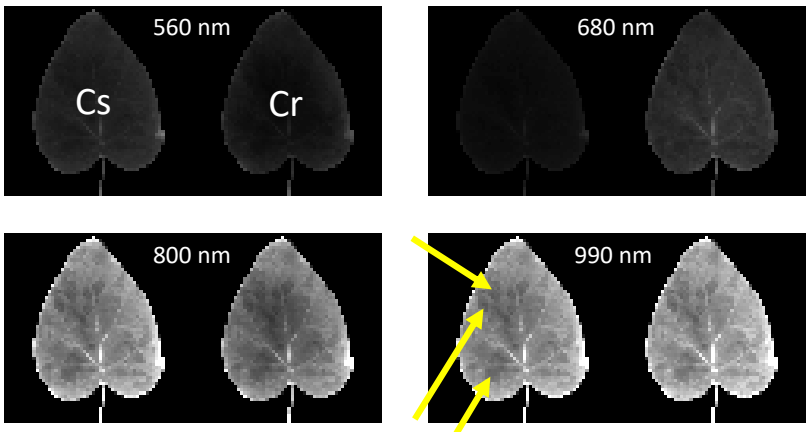
Nous pouvons voir dans cette figure que la reconstruction de l'information spatiale, c'est-à-dire la forme et l'aspect des feuilles ainsi que la position des taches de tavelure au sein d'une tranche donnée était très précise. Nous pouvons le constater en comparant les tranches de Cr et Cs deux à deux pour plusieurs longueurs d'onde (figure 4.3 (b)). Nous expliquons ce succès par deux raisons : d'abord parce qu'au sein d'un cube, la variation de l'information spatiale en fonction de la longueur d'onde était très faible ; ensuite parce que l'algorithme de reconstruction était initialisé avec un cube représentant spatialement l'ordre 0. Ainsi, avant même la première itération de l'algorithme, l'essentiel de l'information spatiale était présent dans le cube reconstruit. Nous avions tenté lors d'expériences préliminaires d'autres initialisations proposées dans la littérature, et en particulier l'utilisation de cubes issus de l'application d'une approximation de l'algorithme de FBP appliqué aux images CTIS [Descour and Dereniak, 1995]. Avec ces initialisations, l'information spatiale des cubes initiaux était appauvrie et la qualité de la reconstruction s'en ressentait grandement.

La reconstruction de l'information spectrale, c'est-à-dire les spectres moyens des zones saines et tavelées, était, elle, plus imprécise. Pour la mesurer, nous avons tracé pour le cube étudié les intensités moyennées des zones saines et tavelées par tranche (figure 4.3 (a)). Les spectres reconstruits suivaient approximativement la forme générale des spectres originaux, et en particulier représentaient fidèlement l'agrandissement de l'écart entre les deux spectres dans le domaine IR. Cependant, une grande part de la complexité des spectres originaux était perdue : nous n'y retrouvions ni la rapide croissance en réflectance au passage dans le domaine IR, ni l'incrément correspondant à la lumière verte.

Enfin, nous pouvons remarquer sur ces reconstructions des artefacts locaux sur certaines tranches. Ainsi, nous pouvons voir sur la tranche de Cr correspondant à la longueur d'onde 800 nm dans la figure 4.3 (b) une zone sombre au centre de la feuille par comparaison avec la tranche de Cs. Ces variations locales de luminosité étaient très fréquentes parmi les cubes reconstruits.



(a) Spectres correspondant aux zones saines de tavelées pour Cs et Cr.



(b) Comparaison à quatre longueurs d'onde différentes de tranches spectrales de Cr et Cs. Dans l'image correspondant à la longueur d'onde 990 nm, les flèches jaunes indiquent la position de quelques taches de tavelure.

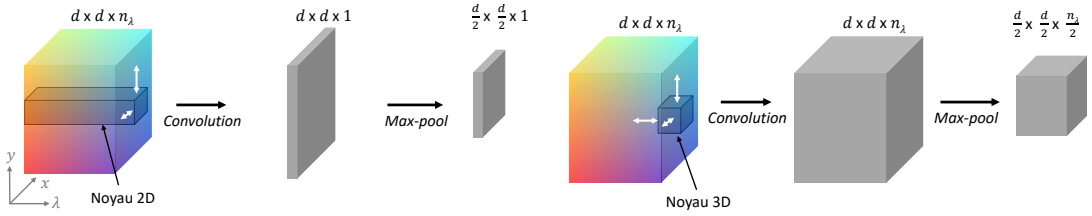
FIGURE 4.3 – Analyse de la qualité de reconstruction d'un cube Cr dont l'erreur de reconstruction par rapport au cube original Cs était de 0,354.

4.2.2 Une performance d'apprentissage de référence

Comme il a été discuté au cours du chapitre 2, le CTIS a été peu exploité dans le monde de l'industrie comme dans celui de la recherche, et il n'existait à notre connaissance aucune analyse automatique à grande échelle menée sur des acquisitions faites par un CTIS. Il nous était donc difficile de définir les modalités d'un apprentissage « classique » sur les cubes reconstruits. Nous avons implémenté une approche basique qui consistait à entraîner le réseau VGGr sur les jeux D_{Cr}, et nous nous sommes par ailleurs penchés sur la façon dont la communauté de l'apprentissage automatique traitait les images hyperspectrales.

Il existe un sous-champ de l'apprentissage automatique dédié aux images hyperspectrales car elles présentaient pour des réseaux des défis d'interprétation par rapport à leurs homologues RVB. Le nombre important de canaux qui les composaient nécessitait une plus grande compression de l'information entre l'espace image (jusqu'à plusieurs centaines de canaux) et l'espace de l'annotation (un canal) au cours de l'apprentissage [Yu et al., 2017]. Par ailleurs, ce nombre important

de canaux menait à un grand nombre de caractéristiques, et il était donc nécessaire de fournir aux réseaux un nombre conséquent d'images pour que l'apprentissage soit mené à bien [Donoho et al., 2000]. Enfin, ces images contenaient une information spatio-spectrale riche, mais prendre avantage des trois dimensions simultanément ainsi que de leurs interactions positives n'était pas trivial [Li et al., 2017b]. Cette dernière difficulté a orienté le développement des réseaux dédiés aux cubes hyperspectraux au cours de ces dernières années. Les premiers réseaux qui apprenaient sur des cubes hyperspectraux avaient une puissance de calcul limitée, et ne pouvaient pas traiter des cubes entiers en entrée. Les auteurs de [Hu et al., 2015] ont procédé à un découpage des cubes de dimension $d \times d \times n_\lambda$ voxels en d^2 vecteurs de dimension $1 \times 1 \times n_\lambda$ éléments qui étaient ensuite traités séparément. Dans ce procédé, seule l'information spectrale était prise en compte. Certain ·es chercheur ·euses ont par la suite intégré une part d'information spatiale en effectuant des découpages des cubes en imagettes spectrales de dimension $k \times k \times n_\lambda$ pixels, $k < d$ [Makantasis et al., 2015; Yu et al., 2017], ou bien en compressant l'information spectrale des cubes via des analyses en composantes principales afin de se ramener dans le cadre d'images de dimension $d \times d \times 3$ pixels et ainsi pouvoir utiliser des réseaux conçus pour des images RVB [Yue et al., 2015; Liang and Li, 2016]. Plus récemment, des réseaux de neurones à convolutions tridimensionnelles, ou *CNN 3D*, développés initialement à des fins d'analyse de vidéos [Tran et al., 2015], ont été appliqués à des cubes hyperspectraux en identifiant la dimension spectrale à celle du temps dans les vidéos [10] [Chen et al., 2016; Li et al., 2017b; Paoletti et al., 2018]. Lors de la convolution d'une image, les noyaux d'un CNN 3D se déplaçaient selon les dimensions spatiales mais aussi selon la dimension des canaux. Cela permettait à la dimension spectrale d'être considérée localement par les couches convolutive et avec un champ de vue variable à l'échelle du réseau, exactement comme sont considérées les dimensions spatiales. Par comparaison, les CNNs classiques « écrasaient » toute l'information spectrale de cubes hyperspectraux dès la première convolution. La figure 4.4 illustre cette différence.



(a) CNN standard. (b) CNN 3D.

FIGURE 4.4 – Comparaison entre (a) un CNN standard (2D) et (b) un CNN 3D. Dans les deux sous-figures sont représentés un cube hyperspectral en entrée, le résultat de l'action d'une couche convolutive à un seul filtre, et le résultat de l'action d'une couche de *max-pool*. Les flèches blanches autour des noyaux convolutifs indiquent le déplacement du noyau dans l'image au cours de la convolution. Les dimensions de chaque objet sont indiquées au-dessus de ceux-ci. Nous ignorons dans cette figure les modifications de dimensions par « effet de bord », c'est-à-dire le rognage des bords de l'image par l'action des noyaux convolutifs et l'éventuel prolongement qui est implémenté pour pallier cela.

Les CNN 3D travaillaient sur le cube hyperspectral dans son intégralité, permettant ainsi d'exploiter complètement les interactions spatio-spectrales. Les résultats sur les *benchmarks* hyperspectraux classiques montraient que ces approches constituaient l'état de l'art pour l'analyse de cubes hyperspectraux [Paoletti et al., 2018]. Il nous a dès lors paru pertinent d'intégrer cette troisième dimension convolutive dans l'architecture « de référence ». Nous avons implémenté un CNN 3D noté VGGr-3D, dont la seule différence avec VGGr était l'utilisation de noyaux convolutifs tridimensionnels (figure 4.5). L'hyperparamètre correspondant à la troisième dimension du noyau

10. Dans le cadre d'analyses de vidéos par CNN 3D, les vidéos sont considérées comme des objets à quatre dimensions $d_1 \times d_2 \times f \times c$, où d_1 et d_2 sont les dimensions spatiales de chaque image constituant la vidéo, f est le nombre de ces images et c est le nombre de canaux de chacune de ces images. Pour adapter l'utilisation de CNN 3D aux cubes hyperspectraux, nous avons considéré ces derniers comme des objets à quatre dimensions $d_1 \times d_2 \times f \times c$ où f était égal au nombre de tranches spectrales du cube et c était fixé à 1.

a été fixé à 3 après une recherche en grille concernant les valeurs 3, 5 et 7 sur le bloc de validation de D^1_{Cr}.

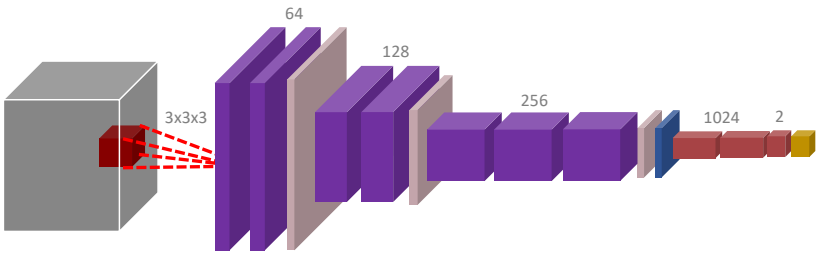


FIGURE 4.5 – Architecture de VGGr-3D. Hormis la dimension des noyaux, tous les hyperparamètres architecturaux sont identiques à ceux de VGGr (figure 4.2).

Nous avons donc mené pour chacun des jeux D_{Cr} un apprentissage avec deux réseaux : un réseau « naïf » VGGr et un réseau dont la structure était plus proche de l'état de l'art pour images hyperspectrales, VGGr-3D. Les performances sur la gamme de sévérité des jeux D_{Cr} sont présentées figure 4.6 Nous pouvons noter tout d'abord que l'action du paramètre de sévérité et la gamme de valeurs qu'il pouvait prendre étaient *a priori* pertinents pour notre étude car les performances de l'apprentissage varient d'une prédiction aléatoire [11] à une prédiction parfaite selon la sévérité du jeu. Nous avions donc à notre disposition une gamme de difficulté représentative de problématiques réelles et qui permettait de comparer les résultats d'autres méthodes à la performance de référence.

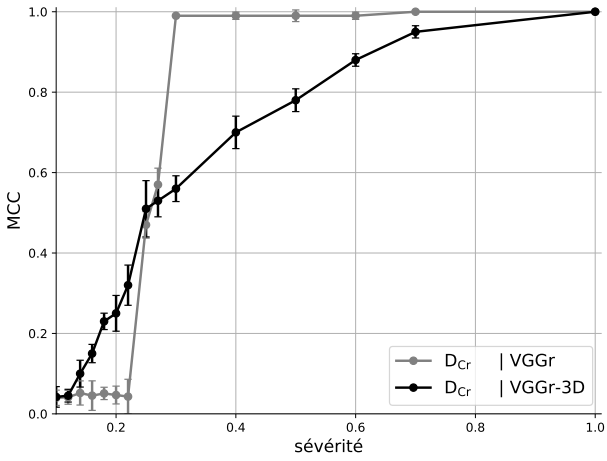


FIGURE 4.6 – Résultats des apprentissages sur les jeux D_{Cr} avec les architectures VGGr et VGGr-3D. Dans toutes les figures présentant les performances d'apprentissage dans ce manuscrit, nous employons une légende de la forme « jeu de données concerné | architecture employée ».

11. Pour les résultats de cette section ainsi que pour toutes les études de ce manuscrit, nous n'avons jamais obtenu un MCC de zéro, même dans les cas les plus difficiles. Il s'agissait d'un artefact du protocole d'apprentissage. En effet, même pour une configuration très difficile ou même impossible (par exemple les jeux correspondant à une sévérité de zéro) où le réseau proposait une prédiction aléatoire à chaque époque, il y avait des époques où le réseau était un peu meilleur que le hasard, par simple effet d'échantillonnage (comme il y avait aussi des époques où le réseau était un peu moins performant que le hasard). Or, la métrique de performance était définie comme celle calculée à l'époque où le coût de validation - globalement inversement corrélé aux MCC de validation et de test - était le plus bas. Cette définition menait donc à une sélection « artificielle » de MCC légèrement au-dessus de zéro, même pour un classifieur aléatoire. En prenant en compte cet effet, nous avons estimé qu'en suivant ce protocole, toute valeur de MCC en dessous de 0,06 était équivalente à une valeur de zéro.

La performance de reconstruction évoluait différemment en fonction de la sévérité selon l'architecture employée. L'architecture VGGr permettait une prédiction parfaite à partir d'une sévérité de 0,3 mais échouait pour les cas plus difficiles correspondant aux sévérités en dessous de 0,22. Utiliser l'architecture VGGr-3D menait à des performances plus linéaires en fonction de la sévérité, offrant des résultats meilleurs que VGGr pour des cas plus difficiles, mais en étant moins compétitif pour des cas plus simples. Nous avons considéré ces deux résultats comme la performance de référence, représentant les performances d'apprentissage « classique » sur les cubes reconstruits. Nous avons par la suite exploré nouvelle approche distincte de ce *pipeline* d'apprentissage.

4.3 L'apprentissage comprimé : une alternative viable pour le CTIS

L'approche que nous avons suivie s'inscrivait dans un courant développé cette dernière décennie au sein de la communauté de l'imagerie computationnelle : l'apprentissage comprimé. Nous présentons à présent un bref historique de ce champ ainsi que l'extension que nous lui avons proposé pour le CTIS.

4.3.1 Une tendance nouvelle en imagerie computationnelle

Dans le cas de nombreuses applications d'imagerie computationnelle, la reconstruction du signal original n'est pas une fin en soi, mais une étape intermédiaire pour effectuer une tâche de classification de la scène. Des chercheur ·euses se sont penchés sur la possibilité d'effectuer ces tâches directement dans l'espace de mesures, en s'affranchissant entièrement de l'étape de reconstruction.

Les premiers travaux dans ce sens ont eu lieu dans le domaine de l'acquisition comprimée (*compressed sensing* en anglais) [Candes et al., 2006]. Ce champ de l'imagerie computationnelle s'intéresse à la possibilité d'obtenir un signal (par exemple, une image) en l'acquérant avec un faible taux d'échantillonnage (autrement dit, avec un nombre réduit de mesures) puis de reconstruire au mieux le signal théorique que l'on aurait acquis avec un taux d'échantillonnage plus élevé. L'acquisition comprimée peut être considérée comme une extension des objectifs de la compression d'image. Dans ce dernier champ, les images sont acquises avec un taux d'échantillonnage élevé puis converties vers un autre domaine de représentation où l'essentiel de l'information qu'elles portent est contenu dans un nombre réduit de coefficients. Les autres coefficients portant le peu de l'information restante peuvent alors être éliminés sans dégrader significativement la qualité de l'image. Dans le champ de l'acquisition comprimée, l'objectif est de ne pas acquérir de mesures inutiles — celles que les algorithmes de compression écartent par la suite — en premier lieu. Les auteurs de [Candes et al., 2006] ont montré que sous certains conditions, ce procédé était possible, et ont développé le formalisme mathématique adéquat. L'acquisition comprimée se base sur l'acquisition d'un signal intermédiaire de taille restreinte qui sert à une reconstruction du signal original, et en ce sens s'inscrit pleinement dans le cadre de l'imagerie computationnelle. Plusieurs imageurs ont été développés suivant les principes de l'acquisition comprimée comme les caméras mono-pixel [Takhar et al., 2006], ou, dans l'imagerie hyperspectrale, l'imageur spectral instantané à ouverture codée (*Coded Aperture Snapshot Spectral Imager* en anglais, ou CASSI) [Wagadarikar et al., 2008].

Le champ de l'apprentissage comprimé est né lorsque les auteurs de [Calderbank et al., 2009] ont montré que dans un cadre d'acquisition comprimée, un algorithme d'apprentissage pouvait, sous certaines conditions, théoriquement obtenir une performance de classification aussi élevée lorsqu'appliqué directement dans l'espace de mesures que lorsqu'appliqué sur le signal reconstruit. Ce résultat était surprenant voire contre-intuitif car les images de l'espace de mesures des capteurs à acquisition comprimée n'étaient pas interprétables par des humains. L'apprentissage comprimé mettait en lumière l'existence d'une information dissimulée aux yeux des humains, mais exploitable

par des algorithmes spécialisés. Le champ s'est développé par la suite grâce à l'émergence de classifieurs de plus en plus performants. En particulier, les CNNs ont abondamment démontré dans les années qui suivirent leur capacité à extraire des caractéristiques complexes et pertinentes pour une grande variété de tâches de vision par ordinateur [Krizhevsky et al., 2012], y compris celles concernant des données que les humains ne pouvaient pas interpréter [Purwins et al., 2019]. Les travaux de [Lohit et al., 2016; Adler et al., 2016; Bacca et al., 2020] ont montré la possibilité d'apprentissage comprimé via des CNNs dans un cadre de classification d'images issues de caméras mono-pixel. Les travaux de [Ramirez et al., 2013] ont procédé similairement dans le CASSI. Plus récemment, les principes de l'apprentissage comprimé ont été appliqués dans des champs plus divers de l'imagerie computationnelle. Les auteurs de [Gao et al., 2019; De Man et al., 2019; Lee et al., 2019] ont montré la possibilité de réaliser des tâches de classification dans le domaine de la CT, c'est-à-dire en entraînant des algorithmes directement sur des sinogrammes. Dans ce travail, nous avons souhaité explorer la possibilité de réaliser un apprentissage comprimé dans l'espace de mesures du CTIS [12] (figure. 4.7). En effet, nous avons pu voir lors de l'étude de la voie « classique » que l'étape de reconstruction, en plus d'être extrêmement chronophage, menait à des approximations dans la reconstruction du cube. Une approche qui permettrait de s'affranchir de cette étape était donc prometteuse, en particulier dans des contextes industriels comme ceux de Carbon Bee.

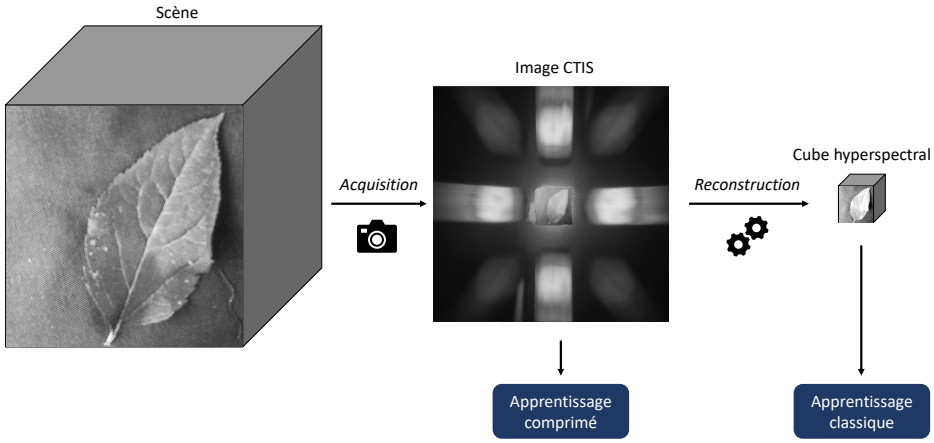


FIGURE 4.7 – Principe de l'apprentissage comprimé appliqué au CTIS : L'apprentissage est mené directement sur l'image CTIS, sans qu'il ne soit nécessaire de reconstruire le cube hyperspectral.

4.3.2 Des performances d'apprentissage proches de la référence

Nous avons mené un apprentissage comprimé adapté au CTIS, c'est-à-dire que nous avons conduit des apprentissages sur les jeux D_{CTIS} avec le réseau VGGr. Les résultats (figure 4.8) montraient que l'apprentissage comprimé était possible pour une gamme de sévérité proche de celle sur laquelle l'apprentissage classique fonctionnait : les performances étaient non-nulles pour les sévérités supérieures à 0,2 et augmentaient pour des valeurs de sévérité croissantes. L'apprentissage comprimé sous ces conditions était cependant moins performant que la performance de référence : les résultats étaient notamment inférieurs à ceux permis par l'utilisation de VGGr en apprentissage classique sur une large gamme de sévérité. Cependant, au vu des gains de temps qu'elle permettait, cette approche pouvait être viable sur certaines gammes de sévérité en fonction

12. Dans le cas d'apprentissages portant sur les espaces de mesures de la CT et du CTIS, la dénomination « apprentissage comprimé » était moins appropriée que dans le cadre de l'acquisition comprimée. En effet, les espaces de mesures concernés (sinogrammes et images CTIS) n'étaient pas à proprement parler des compressions du signal à reconstruire. En fonction des dimensions de ce signal et de celles des projections, il est même très courant que le nombre de mesures acquises soit plus important que le nombre d'éléments du signal reconstruit. Cependant, nous avons retrouvé dans notre application le principe d'apprentissage direct dans l'espace de mesures d'un système d'imagerie computationnelle, et nous avons donc conservé cette dénomination, même si la nature de l'espace de mesures était différent de celui de l'acquisition comprimée.

du compromis entre performance et vitesse d'exécution exigé par les conditions d'expérimentation.

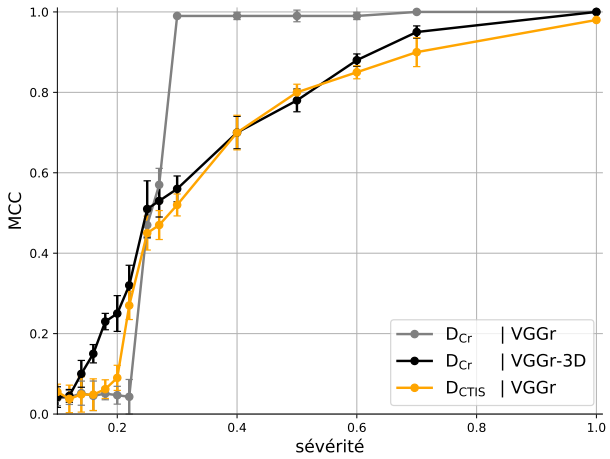


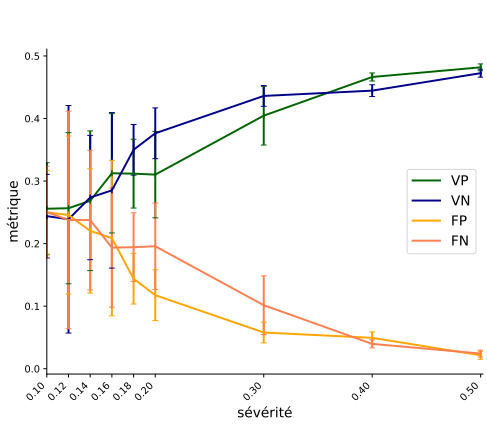
FIGURE 4.8 – Comparaison entre les résultats de l'apprentissage comprimé et la performance de référence.

Pour affiner l'analyse des performances de l'apprentissage comprimé, nous avons calculé la matrice de confusion et la courbe *Receiver Operating Characteristic* (ROC) pour les sévérités dans la gamme [0,10, 0,50] (figure 4.9). Les indicateurs statistiques de classification (sous-figure 4.9 (a)) indiquaient que le nombre de VP et celui de VN étaient proches quelle que soit la sévérité. Cela signifiait que le classifieur n'était pas biaisé vers un certain type d'erreur de classification, c'est-à-dire qu'il ne sous-détectait ni ne sur-détectait systématiquement la présence de tavelure. Les courbes ROC (figure 4.9 (b)) fournissent des indications sur le seuillage optimal à appliquer en sortie de classifieur. Nous avions implémenté un seuillage à 0,5 (figure 1.10), ce qui signifiait que nous n'avions pas incorporé d'*a priori* sur les biais du classifieur. En fonction des sévérités, le seuillage optimal, c'est-à-dire celui dont le point était le plus proche du point (0,1) dans l'espace de la courbe ROC, variait dans une gamme qui incluait cette valeur de 0,5. Lorsque la sévérité augmentait, les points sur les courbes se rapprochaient. Cela signifiait qu'à mesure que la tâche devenait plus facile, les prédictions proposées par le réseau devenaient de plus en plus nettes, et le choix du seuillage prenait de moins en moins d'importance. En vertu des résultats de ces deux analyses, nous avons considéré que le seuillage à 0,5 était un choix pertinent pour juger de la performance du réseau.

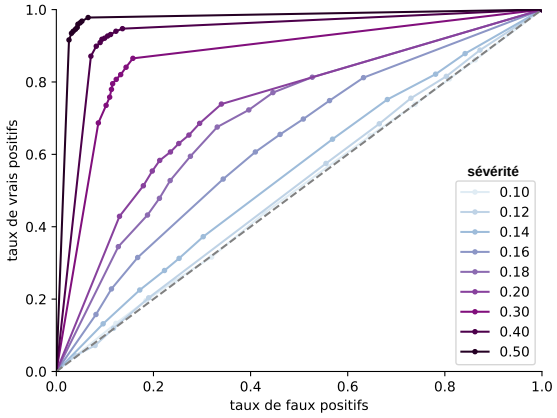
4.3.3 Une exploitation sub-optimale des entrées

En plus de ces analyses qui étaient génériques pour toute tâche de classification, nous avons mené une étude plus spécifique aux images CTIS qui concernait l'apport de chacun des ordres de diffraction. En effet, l'image CTIS pouvait être interprétée comme une image composite, où l'ordre 0 représentait l'information spatiale du cube hyperspectral, et l'ordre 1 l'information spectrale (figure 2.8). Nous nous sommes intéressés à l'apport de chacun de ces ordres pour la performance de la classification. Connaître cet apport était d'autant plus pertinent que les choix du matériel optique d'un CTIS pouvaient influencer de façon bénéfique la résolution d'un des ordres au détriment de l'autre (cf. section 3.2.1).

Pour étudier l'apport des ordres séparément, nous avons créé à partir de chacun des jeux D_{CTIS} deux jeux D_{CTIS0} et D_{CTIS1}. Dans les images des jeux D_{CTIS0}, les pixels correspondant à l'ordre 1 étaient mis à zéro afin de conserver uniquement l'ordre 0. Les jeux D_{CTIS1} étaient construits de façon analogue en mettant à zéro les pixels de l'ordre 1. Les résultats d'apprentissages sur ces deux types de jeux sont présentés figure 4.10. Nous constatons que dans notre cas d'étude, les



(a) Indicateurs statistiques de classification en fonction de la sévérité.



(b) Courbes ROC. Les courbes correspondent aux différentes sévérités (cf. légende). Sur une courbe donnée, chaque point représente le taux de faux positifs FP/(FP+VN) et le taux de vrais positifs VP/(VP+FN) lorsque la sortie du classifieur est seuillée à une certaine valeur. Les valeurs des seuils vont de 0 à 1 par pas de 0,1 : chaque pas correspond à un point de la courbe.

FIGURE 4.9 – Analyses des performances de l'apprentissage comprimé.

performances sur D_{CTIS} et D_{CTIS0} étaient très similaires, et supérieures à celles obtenues avec D_{CTIS1}. Cette proximité indiquait que le réseau se basait quasiment exclusivement sur l'ordre 0 pour mener à bien son apprentissage.

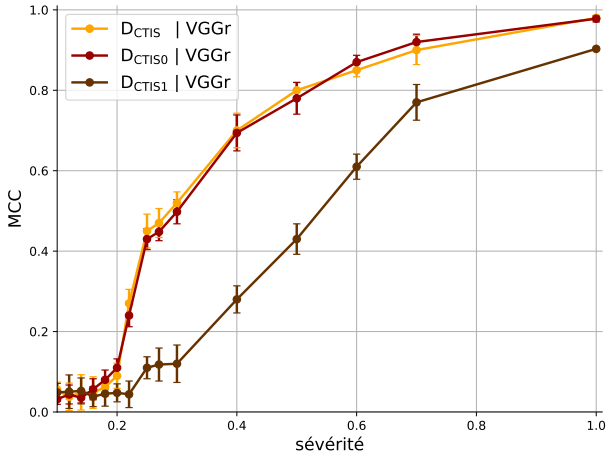


FIGURE 4.10 – Etude de l'apport de chacun des ordres pour les apprentissages menés sur D_{CTIS}.

Nous avons souhaité confirmer ce résultat en implémentant un algorithme de visualisation. De tels algorithmes visent à expliquer visuellement les décisions d'un réseau de neurones [Yosinski et al., 2015], qui sont à cause de leur grand nombre de paramètres souvent qualifiés de « boites noires » [Shwartz-Ziv and Tishby, 2017]. Nous nous sommes concentrés sur les visualisations destinées à mettre en lumière au sein d'une image d'entraînement les portions qui ont le plus contribué à l'attribution de sa classe [Springenberg et al., 2015]. Nous avons utilisé l'algorithme Grad-CAM [Selvaraju et al., 2017] qui constituait l'état de l'art pour ce type d'analyse à l'époque de nos travaux. L'algorithme de Grad-CAM s'appliquait sur un réseau entraîné et pour une image donnée. Il produisait un moyennage des cartes de caractéristiques issues du dernier bloc convolutif,

pondéré par les gradients provenant des neurones de la dernière couche FC. L'utilisation des cartes de caractéristiques permettait de produire comme visualisation une image ayant une structure spatiale similaire à l'image d'entrée, et les gradients des couches FC complétaient cette information en augmentant l'intensité des portions de ces cartes qui avaient eu un impact fort sur la prédiction finale. La figure 4.11 présente le résultat de l'application de Grad-CAM sur un réseau VGGr entraîné sur le jeu $D^{0,4}_{CTIS}$. Nous pouvons y constater que le réseau s'est effectivement concentré sur l'ordre 0.

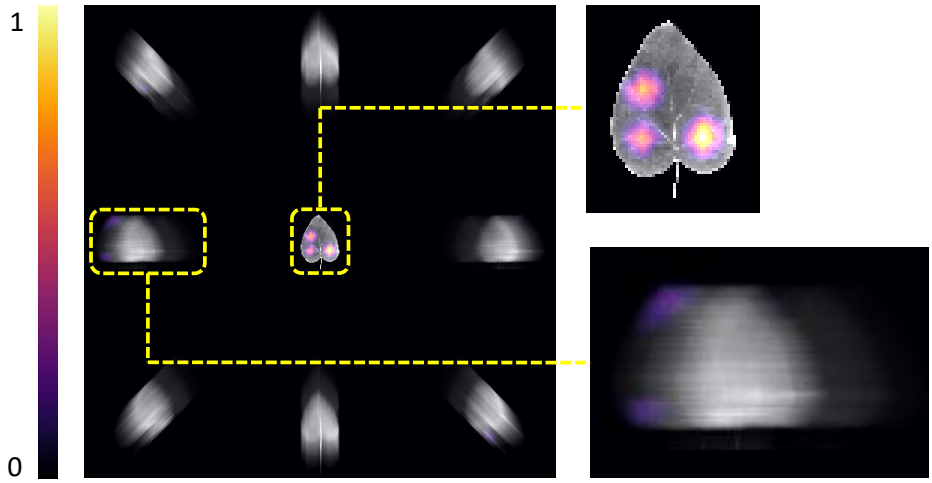


FIGURE 4.11 – Résultat de l'application de l'algorithme Grad-CAM sur un réseau VGGr entraîné sur le jeu $D^{0,4}_{CTIS}$. La carte de chaleur produite par l'algorithme est représentée en fausses couleurs (échelle à gauche) et superposée à l'image de $D^{0,4}_{CTIS}$ utilisée pour générer la visualisation. Des agrandissements de l'ordre 0 et d'une projection de l'ordre 1 sont proposés à droite de la figure.

Ces dernières expériences montraient que, bien que l'apprentissage comprimé était possible sur les images CTIS, le réseau ne tirait profit que d'une partie de l'information contenue dans ces images. Nous avons souhaité mieux tirer parti de l'intégralité de l'information disponible, convaincus que la séparation de l'information spatiale et spectrale pouvait être une chance plutôt qu'une entrave pour un apprentissage. À cette fin, nous avons développé une architecture neuronale spécifique dédiée aux images CTIS.

4.4 Une architecture dédiée aux images CTIS

4.4.1 Des traitements spécifiques aux ordres de l'image

La figure 4.12 présente l'architecture proposée, nommée CTIS-Net. Le but premier de cette architecture était de tirer parti des deux ordres du CTIS pour améliorer la qualité de la classification, un procédé connu sous le nom de fusion de données. Pour suivre la nomenclature de ce champ, nous avons désigné par *modalités* les deux ordres. Bien qu'exploiter l'image CTIS telle quelle avec VGGr, comme nous l'avons fait dans la section précédente, pouvait être considéré comme un cas de fusion très précoce de ces modalités, nous avons estimé que traiter les ordres séparément pouvait être utile au regard de l'étude de l'apport de chacun de ces ordres.

Nous avons donc implémenté une architecture « à deux *branches* » pour la fusion d'images [Eitel et al., 2015]. Une telle architecture prenait deux images en entrée. Nous avons fourni à cette architecture les bases de données D_{CTIS0} et D_{CTIS1} (figure 4.12, bloc a). Les modalités étaient ensuite traitées indépendamment par deux sous-réseaux que l'on nomme branches (figure 4.12, bloc c). Pour mener une comparaison la plus juste possible avec les performances de VGGr, nous avons utilisé la même architecture que les couches convolutives de VGGr pour chacune de ces branches. Les cartes de caractéristiques en sortie de chaque branche étaient ensuite concaténées selon l'axe

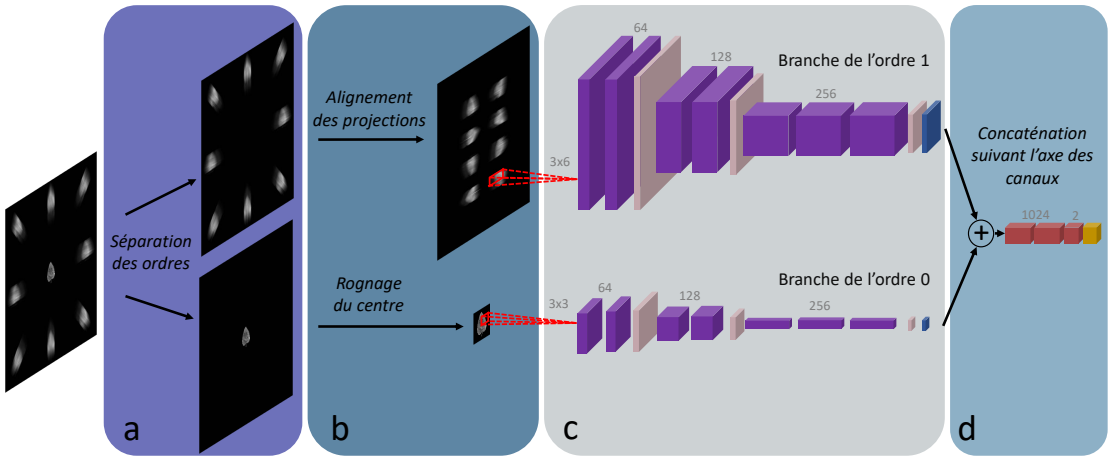


FIGURE 4.12 – Architecture de CTIS-Net. Bloc a : séparation des ordres. Bloc b : prétraitement des ordres. Bloc c : réseau en branches. Bloc d : concaténation et décision.

des canaux, c'est-à-dire que les deux cartes de dimension $7 \times 7 \times 256$ éléments étaient réunies en une carte de dimension $7 \times 7 \times 512$ éléments. Ce nouvel objet était par la suite traité par des couches FC. Nous avons utilisé la même architecture que les couches FC de VGGr pour ces couches (figure 4.12, bloc d).

La séparation des ordres d'une image CTIS, en plus de rendre possible une recherche personnalisée de caractéristiques pour chacun via une différenciation des couches convolutives, nous a permis d'implémenter des prétraitements spécifiques à chacun (figure 4.12, bloc b). Concernant l'ordre 1, l'information était répartie dans de multiples projections qui étaient étirées par la décomposition spectrale. Pour mieux exploiter cette information, nous avons intégré deux modifications à la branche de l'ordre 1. Premièrement, nous avons implémenté un prétraitement pour les images de D_{CTIS1} qui consistait en l'*alignement* des projections selon le même axe. Chaque projection était rognée et tournée de façon à ce que les axes des projections soient alignés à l'horizontale (figure 4.13). Cette opération était destinée à aider le réseau à apprendre des caractéristiques utiles pour toutes les projections, sans qu'il ne lui soit nécessaire d'acquérir une invariance à la rotation. Les positions de chaque projection dans cette nouvelle image et le choix d'un alignement horizontal plutôt que vertical étaient arbitraires car nous avons supposé que ces choix n'affecteraient pas les performances des apprentissages.

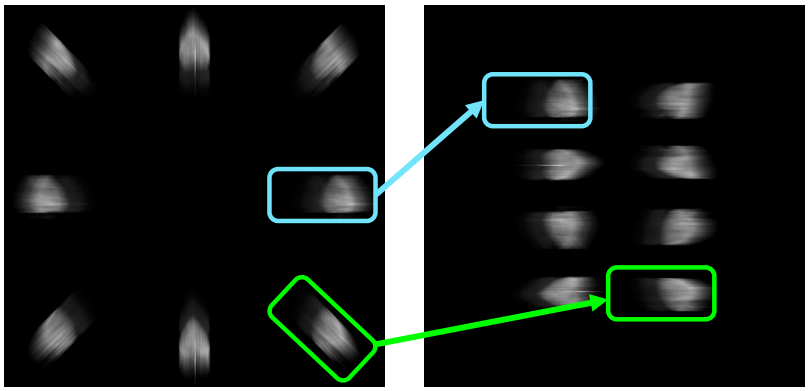


FIGURE 4.13 – Illustration du processus d'alignement mis en place pour les images de D_{CTIS1}. Cette étape correspond à l'étape « Alignement des projections » dans le bloc b de la figure 4.12.

Deuxièmement, inspirés par le réseau personnalisé que les auteurs de [Lee et al., 2019] ont développé pour traiter des sinogrammes issus de la CT, nous avons utilisé pour les couches convo-

lutives de la branche de l'ordre 1 des noyaux convolutifs rectangulaires au lieu de noyaux carrés. Ce changement était destiné à mieux correspondre aux formes des caractéristiques des projections de l'ordre 1. En particulier, le spectre d'une zone spatiale donnée d'une feuille était représenté dans l'image comme des lignes suivant l'axe des projections, c'est-à-dire l'horizontale après l'opération d'alignement. Nous avons fait l'hypothèse que des champs de vue neuronaux étendus dans la direction de la décomposition spectrale mènerait à une meilleure compréhension de ce spectre et serait ainsi bénéfique pour la prise de décision du réseau. Ainsi, nous avons utilisé des noyaux de dimension 3×6 plutôt que 3×3. Nous avons adapté le transfert de poids de l'ILSVRC en redimensionnant les noyaux convolutifs téléchargés de 3×3 à 3×6 par interpolation bilinéaire.

Quant à l'ordre 0, nous avons implémenté un rognage du centre des images de D_{CTIS0} afin de conserver uniquement les pixels de la projection correspondant à l'ordre 0. Cette opération a permis une accélération substantielle de l'entraînement, mais a également amélioré la résolution spatiale des cartes de caractéristiques après la couche GAP (figure 4.14). En effet, nous rappelons que la couche GAP avait pour effet de redimensionner les cartes de caractéristiques à une dimension fixe, dans notre cas 7×7 éléments. Une image de D_{CTIS0} de dimension 512×512 pixels menait, à l'issue des couches de *max-pool* de VGGr à des cartes de caractéristiques de dimension spatiale 64×64 éléments qui étaient ensuite réduites spatialement à 7×7 par la couche GAP, menant à une forte perte d'information (figure 4.14, haut). En rognant les images de D_{CTIS0} à une dimension 60×60 pixels, les cartes de caractéristiques en sortie des couches *max-pool* de la branche de l'ordre 0 étaient déjà de dimension spatiale 7×7 éléments et ne subissaient donc pas de dégradation de leur résolution spatiale (figure 4.14, bas).

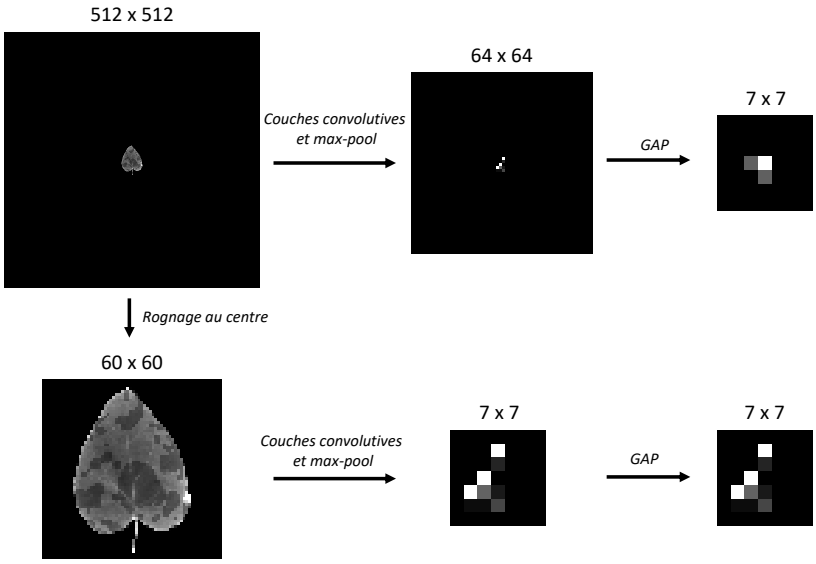


FIGURE 4.14 – Illustration de l'effet bénéfique d'un rognage au centre de l'ordre 0 vis-à-vis de la couche GAP. Les dimensions des objets sont indiquées au dessus d'eux.

L'entraînement de CTIS-Net suivait un protocole particulier. Les deux modalités que nous traitions étaient de nature différente, et nous avions observé lors d'expériences préliminaires une différence de temps de convergence de VGGr lorsqu'appliqué sur D_{CTIS0} ou sur D_{CTIS1}. La figure 4.15 présente les courbes de deux entraînements de VGGr, l'un portant sur D^1_{CTIS0} et l'autre sur D^1_{CTIS1}. Le réseau avait convergé en environ 30 époques sur l'ordre 0 et en 80 sur l'ordre 1. Cette différence de temps de convergence, qui était peut-être causée en partie par le transfert de poids de l'ILSVRC qui orientait l'entraînement vers une recherche de caractéristiques « spatiales », était délétère pour l'apprentissage avec CTIS-Net. En effet, au cours d'un entraînement, le réseau convergeait sur l'information de l'ordre 0 et ignorait l'information de l'ordre 1, même après 80 époques. Du point de vue de l'optimiseur, nous avons supposé qu'à partir de l'époque 30, les poids

du réseau constituaient un « minimum local » défini uniquement par l'information de l'ordre 0. En conséquence, les performances de CTIS-Net étaient identiques que l'on inclue l'ordre 1 ou bien que l'on mette tous les pixels de cet ordre à zéro. Cette problématique d'un réseau de fusion qui développait les caractéristiques d'une seule branche au détriment de l'autre était présente dans de nombreuses études portant sur la fusion de données [Wang et al., 2020a]. Afin de pallier cet effet, nous avons réalisé les entraînements de CTIS-Net « en deux temps » [Eitel et al., 2015]. Dans un premier temps, nous entraînions chaque branche individuellement en fournissant à l'autre branche un jeu d'images vides. Puis, nous créions une nouvelle instance de CTIS-Net dont les poids des couches convolutives étaient initialisées aux valeurs convergées à la du fin premier temps pour chaque branche. Cette distinction permettait au réseau d'extraire des caractéristiques de chaque modalité indépendamment du temps de convergence nécessaire afin d'être dans les meilleurs conditions pour les combiner lors du second temps.

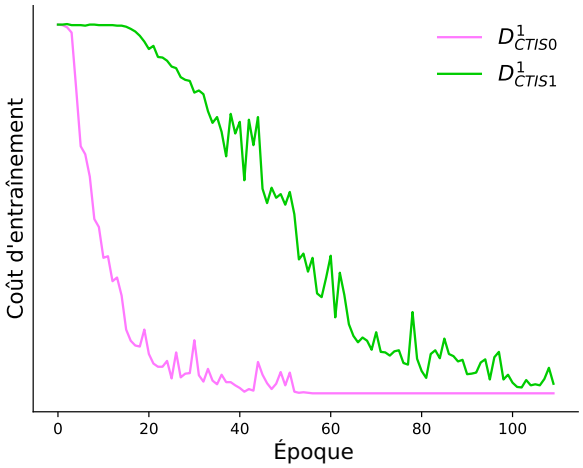


FIGURE 4.15 – Évolution du coût d'entraînement en fonction des époques pour deux entraînements portant sur D^1_{CTIS0} et D^1_{CTIS1} avec VGGr.

4.4.2 Une nette amélioration de la performance

L'utilisation d'une architecture spécifiquement conçue pour les images CTIS permettait une nette amélioration des performances par rapport à celle du réseau plus générique VGGr (figure 4.16). Parce qu'elle était constituée de deux branches, l'architecture de CTIS-Net contenait environ deux fois plus de paramètres que celle de VGGr. Pour mener une comparaison juste avec ce dernier, nous avons implémenté une version de VGGr où le nombre de filtres dans chaque couche convolutive était doublé par rapport à VGGr, que nous avons noté VGGr2. La comparaison des performances de CTIS-Net avec VGGr2 montrait que la progression permise par CTIS-Net n'était pas simplement due à une augmentation du nombre de paramètres. L'architecture CTIS-Net permettait en outre d'être proche des meilleurs résultats de la performance de référence et même de les dépasser sur certaines gammes de sévérité.

Afin de compléter la comparaison entre les différentes architectures, nos avons résumé dans le tableau 4.2 les spécificités des différentes expériences présentées jusqu'ici autres que leur performance, en soulignant en particulier les durées des entraînements et des prédictions. Nous confirmons par ces mesures que l'apprentissage comprimé permet un gain de temps substantiel par rapport à l'apprentissage classique, en particulier lors de la phase de prédiction.

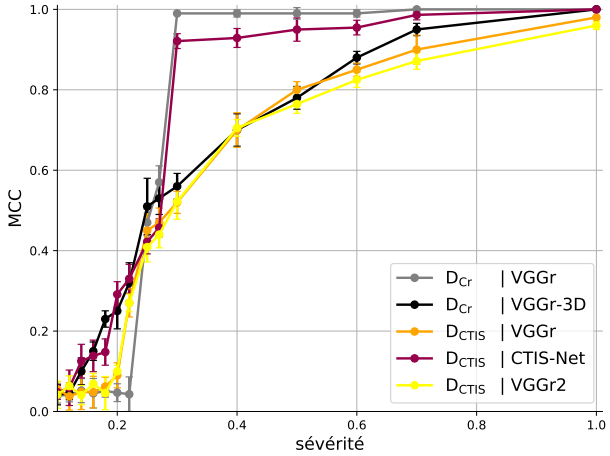


FIGURE 4.16 – Comparaison des résultats d'apprentissage comprimé avec l'architecture dédiée CTIS-Net et ceux avec l'architecture standard VGGr.

Jeu de données	Réseau	Taille des noyaux	Nombre de paramètres	Temps d'entraînement (heures, pour 1800 images*)	Temps de prédiction (ms, pour une image)
$D^{0,4}_{Cr}$	VGGr-3D	$3\times3\times3$	$1,9.10^7$	4,0 (+0,6)**	50 (+1200)**
$D^{0,4}_{Cr}$	VGGr	3×3	$1,6.10^7$	1,6 (+0,6)**	10 (+1200)**
$D^{0,4}_{CTIS}$	VGGr	3×3	$1,6.10^7$	1,6	10
$D^{0,4}_{CTIS}$	VGGr2	3×3	$3,4.10^7$	3,0	25
$D^{0,4}_{CTIS}$	CTIS-Net	3×3 & 3×6 ***	$3,2.10^7$	4,4	35

* Ce nombre correspond à la taille des blocs d'entraînement pour tous les jeux présentés dans ce chapitre.
** Le temps indiqué entre parenthèses correspond au temps de reconstruction nécessaire pour obtenir $D^{0,4}_{Cr}$ à partir de $D^{0,4}_{CTIS}$.
*** La branche de l'ordre 0 a des noyaux de taille 3×3 et celle de l'ordre 1 des noyaux de taille 3×6 (cf. figure 4.12).

TABLEAU 4.2 – Comparaison des caractéristiques des différentes expériences, mesurées sur la sévérité 0,4.

4.4.3 Une étude par ablation met en lumière les apports de CTIS-Net

Afin de déterminer la contribution de chacun des changements architecturaux entre VGGr et CTIS-Net au gain de performance observé, nous avons mené une étude par ablation sur cette dernière architecture.

Modifications de l'ordre 0

Pour commencer, nous avons évalué l'apport des modifications faites à l'ordre 0. Nous avons concentré notre étude sur la branche de l'ordre 0 de CTIS-Net en ignorant l'apport de l'ordre 1. Dans les faits, nous avons créé des jeux $D_{CTIS0\ rogné}$ composés d'images de l'ordre 0 rognées au centre, et nous avons entraîné un réseau VGGr sur $D_{CTIS0\ rogné}$ et D_{CTIS0}. Cette procédure était équivalente à éliminer la branche de l'ordre 1 de CTIS-Net. Les résultats indiquaient que le rognage permettait une performance légèrement supérieure dans la gamme de sévérité en dessous de 0,2 (figure 4.17). Nous avons également mesuré le temps que cette opération permettait de gagner, notamment à l'entraînement : une époque d'entraînement sur $D_{CTIS0\ rogné}$ dure huit secondes environ contre une minute trente environ sur D_{CTIS0}.

Modifications de l'ordre 1

Concernant les modifications portées à l'ordre 1, nous avons concentré notre étude sur la branche éponyme en comparant les performances de la branche de l'ordre 1 de CTIS-Net selon diverses configurations pour évaluer l'apport des noyaux rectangulaires et de l'alignement. Comme

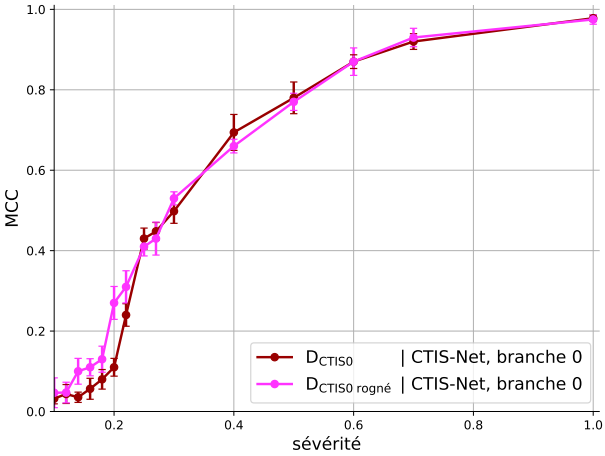


FIGURE 4.17 – Étude de l'apport du rognage de l'ordre 0.

pour l'étude de l'impact du rognage de l'ordre 0, nous avons entraîné dans les faits un réseau VGGr. Nous avons noté $D_{CTIS1\ aligné}$ les jeux constitués d'images des projections de l'ordre 1 « alignées ». Pour éviter une surabondance de résultats, nous avons concentré notre étude sur les jeux correspondant à la sévérité 0,4 (tableau 4.3). Les résultats étaient comparables pour d'autres sévérités.

Jeu de données	Réseau	Taille des noyaux	Nombre de paramètres	Temps d'entraînement (heures, pour 1800 images)	Temps de prédiction (ms, pour une image)	Performance (MCC)
$D^{0,4}_{CTIS1\ aligné}$	VGGr	3×3	$1,6.10^7$	1,6	7	$0,79_{\pm0,018}$
		6×3	$1,7.10^7$	2,9	14	$0,81_{\pm0,032}$
		3×6	$1,7.10^7$	3,1	14	$0,93_{\pm0,031}$
		6×6	$2,1.10^7$	6,0	23	$0,94_{\pm0,019}$
$D^{0,4}_{CTIS1}$	VGGr	3×3	$1,6.10^7$	1,6	7	$0,52_{\pm0,015}$
		6×3	$1,7.10^7$	2,9	14	$0,56_{\pm0,029}$
		3×6	$1,7.10^7$	2,9	14	$0,56_{\pm0,021}$
		6×6	$2,1.10^7$	5,8	23	$0,58_{\pm0,017}$
$D^{0,4}_{CTIS0\ rogné}$	VGGr	3×3	$1,6.10^7$	0,06	0,7	$0,72_{\pm0,008}$
		6×3	$1,7.10^7$	0,13	1,4	$0,62_{\pm0,016}$
		3×6	$1,7.10^7$	0,13	1,4	$0,63_{\pm0,015}$
		6×6	$2,1.10^7$	0,24	2,3	$0,51_{\pm0,016}$

TABLEAU 4.3 – Étude par ablation concernant les améliorations portées à la branche de l'ordre 1.

Pour juger de l'effet de l'alignement des projections, il faut comparer les résultats sur $D^{0,4}_{CTIS1\ aligné}$ à ceux sur $D^{0,4}_{CTIS1}$ (quatre premières lignes du tableau 4.3 et les quatre suivantes). Nous y constatons que ce procédé a eu un effet positif considérable sur la performance du réseau, quelle que soit la taille du noyau utilisé. Ces résultats donnaient aussi une indication sur l'effet des noyaux rectangulaires : la taille du noyau avait un effet positif substantiel lorsque le réseau travaillait sur les images alignées. Ce résultat nous paraissait cohérent car nous avions implémenté l'étirement des noyaux selon un axe pour bénéficier des caractéristiques des projections alignées au préalable selon ce même axe. Pour évaluer plus finement l'impact de la taille des noyaux, il faut comparer entre eux les résultats sur $D^{0,4}_{CTIS1\ aligné}$ (quatre premières lignes du tableau). Ces résultats montraient que c'était bien l'étirement du noyau dans le sens des caractéristiques alignées plutôt que le simple agrandissement de celui-ci qui permettait d'obtenir un gain de performance. Cette distinction était particulièrement visible en comparant les performances obtenues grâces à des noyaux de dimension 3×6 (ligne 3) par rapport à celles obtenues avec des noyaux 6×3 et 6×6 (lignes 2 et 4). Nous pouvons noter par ailleurs qu'augmenter la taille des noyaux augmentait de façon non négligeable le temps d'entraînement et de prédiction, justifiant d'autant plus l'utilisation de noyaux rectangulaires 3×6 par rapport à des noyaux 6×6 qui menaient à des performances sensiblement

similaires.

Par ailleurs, les résultats concernant le jeu $D^{0,4}_{\text{CTIS0 rogné}}$ (quatre dernières lignes) montraient qu'il était pertinent d'affecter des tailles de noyaux différentes en fonction de la branche comme c'était le cas dans CTIS-Net. En effet, la taille de noyau optimale pour l'ordre 0, 3×3, était différente de celle pour l'ordre 1 aligné, 3×6. Ce résultat était cohérent avec la recherche par grille que nous avions menée pour fixer les hyperparamètres de VGGr (section 4.1.1). En effet, puisque VGGr se concentrait exclusivement sur l'ordre 0 lorsqu'on lui fournissait une image CTIS (figure 4.10), il était logique que la taille de noyau optimale ait été trouvée à 3×3, au détriment de l'ordre 1 qui était ignoré dans cette recherche en grille.

Fusion des ordres

Pour terminer cette étude par ablation, nous avons évalué le bénéfice d'exploiter conjointement les jeux $D_{\text{CTIS0 rogné}}$ et $D_{\text{CTIS1 aligné}}$, c'est-à-dire de procéder à la fusion de ces ordres. Nous avons comparé les résultats des apprentissages sur chaque branche individuelle contre celui réalisé en utilisant CTIS-Net (figure 4.18).

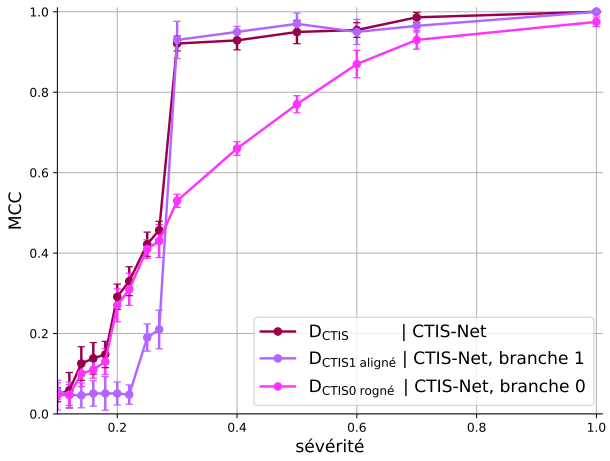


FIGURE 4.18 – Étude par ablation pour évaluer l'impact de la fusion des ordres.

Bien que la fusion des deux ordres n'abaissait pas la performance, elle ne permettait pas d'obtenir une performance meilleure que celle permise par la meilleure des branches du réseau considérée individuellement. Nous avons supposé que certaines limitations à une fusion plus efficace venaient des principes optiques du CTIS lui-même. En effet, les informations spatiale et spectrale d'un cube n'étaient pas aussi clairement séparées dans les ordres respectifs d'une image CTIS que l'on pourrait le croire à première vue. L'ordre 0 contenait une information moyennée du spectre du cube, et l'ordre 1 une information spatiale, certes rendue floue par la décomposition spectrale. L'information apportée par les deux modalités n'était donc pas aussi orthogonale que dans d'autres cas de fusion de données (par exemple, image et son [Meishvili et al., 2020], image et odeur [Korel et al., 2001] ou image et information mécanique [Ruiz-Altisent et al., 2006]) et par conséquent les modalités avaient ici une complémentarité limitée. De plus, nous avons fait l'hypothèse que notre cas d'étude exacerbait cette corrélation. En particulier, la différence de réflectance entre une zone saine et une zone tavelée d'une feuille était considérable pour une gamme significative de longueurs d'onde (figure 3.10). En conséquence, l'ordre 0 contenait une information spectrale importante.

Nous avons cependant considéré cette fusion comme utile pour plusieurs raisons. Premièrement, la modalité optimale pour mener un apprentissage sur un jeu de données pouvait varier d'un jeu à l'autre, sans qu'il ne soit nécessairement possible de la connaître *a priori*. Nous avons pu constater cet effet en faisant varier le paramètre « sévérité » des jeux (figure 4.18) mais nous avons aussi imaginé des cas où la modalité optimale pouvait varier en fonction du type de maladie à détecter. Nous nous attendions par exemple à ce que la fusion soit utile dans un cas de détection de multiples maladies ou bien un cas où de multiples stades de la même maladie étaient rassemblés dans un même jeu de données. Deuxièmement, nous supposions qu'il existait des cas où les informations spatiale et spectrale dans l'image CTIS étaient plus indépendantes que dans notre cas d'étude. Nous avons exploré dans l'annexe A un apprentissage où l'architecture CTIS-Net permettait véritablement de tirer parti de la fusion des deux ordres.

4.5 Les performances subsistent malgré l'ajout de bruit

Bien que nous ayons émulé avec soin les propriétés optiques du CTIS dans notre simulateur éponyme, les images que celui-ci permettait de produire et que nous avons exploitées dans ce chapitre manquent de caractéristiques distinctives d'images réelles, comme discuté à la section 3.4. Nous présentons dans cette section un premier travail destiné à améliorer le réalisme des images de D_{CTIS} via l'ajout de bruit. Similairement à l'étude par ablation portant sur l'ordre 1 (section 4.4.3), nous avons limité cette analyse au jeu $D^{0,4}_{CTIS}$.

Comme discuté à la fin de la section 2.2.4, il existe des études qui ont modélisé mathématiquement des distributions de bruits thermique et de grenaille sur des images à faible intensité lumineuse comme celles acquises par le CTIS. Une telle modélisation requérait cependant d'identifier et de séparer les deux sources principales de bruit dans les images. Devant la difficulté de cette opération pour les images CTIS réelles que nous avions à disposition, et après avoir constaté que le bruit présent dans ces images suivait une distribution de niveaux de gris proche d'une gaussienne, nous avons préféré suivre une approche dite « procédurale » [Dong et al., 2019]. Une telle approche consistait à créer une distribution de niveaux de gris et à en ajuster les caractéristiques par rapport à celles mesurées sur une distribution réelle, dites « caractéristiques modèle ». Pour évaluer ces caractéristiques dans le cas du bruit des images CTIS réelles, nous avons conduit des mesures dans des zones périphériques de ces images, à faible intensité lumineuse, où nous estimions que les deux sources de bruit susmentionnées étaient présentes.

Concernant le choix des caractéristiques à ajuster, nous nous sommes basés sur des travaux que nous avions mené auparavant pour une autre application en sciences végétales [Douarre et al., 2018a]. Nous avons d'abord mesuré deux caractéristiques liées à la distribution des niveaux de gris : les deux moments d'ordre 1 et 2, c'est-à-dire la moyenne et l'écart type du bruit. Nous avons noté $\mu_{\text{bruit réel}}$ et $\sigma_{\text{bruit réel}}$ ces mesures. Nous avons également calculé une caractéristique spatiale liée à la *texture* de ce bruit. Il existait de nombreuses manières de définir et de mesurer ce que l'on appelle la « texture » d'une image [Haralick et al., 1973; Howarth and Rüger, 2004]. Nous nous sommes basés sur la fonction d'autocorrélation [Jain et al., 1995] qui mesurait la concordance spatiale de pixels d'intensité similaire entre une image et cette même image décalée spatialement. Cette mesure donnait ainsi des indications sur la présence de structures répétées dans l'image et la taille typique de celles-ci. Plus formellement, l'autocorrélation A d'une image I de dimension $d \times d$ pixels prenait la forme d'une image de même taille et se mesurait comme suit :

$$\mathrm{A}[dx, dy] = \frac{\sum_{x=0}^{d} \sum_{y=0}^{d} \mathrm{I}[x, y].\mathrm{I}[x + dx, y + dy]}{\sum_{x=0}^{d} \sum_{y=0}^{d} \mathrm{I}^2[x, y]}. \tag{4.4}$$

Nous avons noté $\mathrm{A}_{\text{bruit réel}}$ la mesure que nous avons ainsi effectué.

Une fois ces mesures de caractéristiques modèle réalisées, nous avons créé pour chaque image du jeu $D^{0,4}_{\text{CTIS}}$ une image de bruit synthétique où chaque pixel était tiré indépendamment de la distribution $\mathcal{N}(\mu_{\text{bruit réel}},\sigma_{\text{bruit réel}})$. Puis nous avons ajusté l'autocorrélation de ces images en suivant l'algorithme 3.2 pour différentes valeurs de rayon r et en retenant la valeur qui menait à l'autocorrélation de l'image la plus proche de $A_{\text{bruit réel}}$. Nous avons choisi la somme des écarts pixel à pixel au carré comme mesure de distance entre l'autocorrélation de l'image et $A_{\text{bruit réel}}$. La figure 4.19 présente une comparaison visuelle entre le bruit réel et le bruit synthétique ainsi créé.

Nous avons par la suite créé le jeu $D^{0,4}_{\text{CTIS bruité}}$ en ajoutant une image de bruit synthétique à toutes les images de $D^{0,4}_{\text{CTIS}}$. La figure 4.20 présente un comparaison entre une image des deux jeux de données.

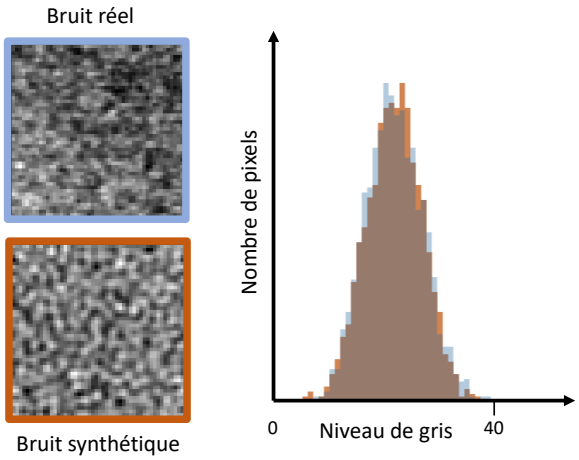


FIGURE 4.19 – Quelques illustrations concernant la génération du bruit synthétique. Gauche : des imagettes de dimension 50 × 50 pixels de bruits réel et synthétique sont présentées. Leur contraste et leur luminosité ont été augmentés à des fins de visualisation. Droite : les histogrammes de niveaux de gris des imagettes sont tracés sur un graphe commun, avec un code couleur correspondant aux encadrements des imagettes.

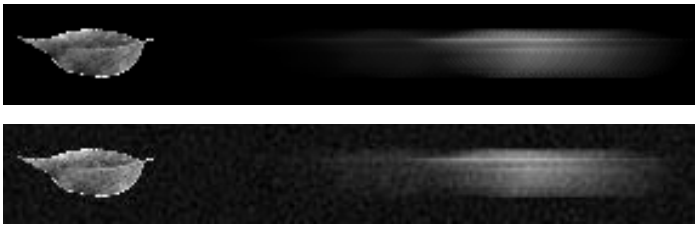


FIGURE 4.20 – Un extrait d'image de $D^{0,4}_{\text{CTIS}}$ (haut), comparé à la même image dans $D^{0,4}_{\text{CTIS bruité}}$ (bas).

Nous avons ensuite reconstruit à partir des images de $D^{0,4}_{\text{CTIS bruité}}$ les cubes hyperspectraux correspondants, constituant ainsi le jeu $D^{0,4}_{\text{Cr bruité}}$. Nous avons alors conduit à nouveau les principales méthodes de classification présentées dans ce chapitre sur ces deux jeux bruités (tableau 4.4). La performance était sans surprise amoindrie par rapport aux études menées sur les versions non bruitées des jeux. La performance d'apprentissage comprimé était toujours dans cette configuration proche de celle obtenue sur le jeu reconstruit, indiquant que cette approche pourrait effectivement être pertinente pour des applications réelles. L'utilisation de l'architecture CTIS-Net permettait d'augmenter encore cette performance, bien que de façon moins importante que dans le cas « propre ».

Jeu de données	Réseau	Taille des noyaux	Nombre de paramètres	Temps d'entraînement (heures, pour 1800 images)	Temps de prédiction (ms, pour une image)	Performance (MCC)
$D^{0,4}_{\text{CTIS bruité}}$	VGGr	3×3	$1,6.10^7$	1,6	6	$0,52_{\pm0,028}$
$D^{0,4}_{\text{Cr bruité}}$	VGGr-3D	$3\times3\times3$	$1,9.10^7$	4,0 (+0,6)	30 (+720)	$0,54_{\pm0,028}$
$D^{0,4}_{\text{CTIS bruité}}$	CTIS-Net	3×3 & 3×6	$3,2.10^7$	4,4	22	$0,57_{\pm0,030}$

TABLEAU 4.4 – Performances de classification sur données bruitées.

4.6 Une plus grande polyvalence par rapport aux imageurs classiques

La comparaison des résultats d'apprentissage comprimé avec celle d'apprentissage classique était pertinente dans le cadre de l'étude d'une imagerie computationnelle. Nous avons par ailleurs souhaité comparer les performances rendues possibles avec l'imagerie CTIS avec des imageurs plus standards. Nous nous sommes intéressés en particulier aux imageurs RVB et IR, imageurs qui étaient à la fois considérablement plus répandus et moins onéreux que n'importe quel capteur hyperspectral. Les performances qu'ils permettaient constituaient une référence quasiment *de facto* pour évaluer un nouveau type d'imageur spectral.

Ainsi avons-nous comparé les résultats obtenus sur les images de D_{CTIS} aux performances d'apprentissage obtenues avec les jeux D_{RVB} et D_{IR} (figure 4.21). Il en est ressorti que si l'apprentissage comprimé sur images CTIS offrait des performances comparables voire meilleures sur une courte gamme de sévérité par rapport au RVB, les deux imageries standards restent en général des alternatives plus performantes que le CTIS dans notre cas d'étude.

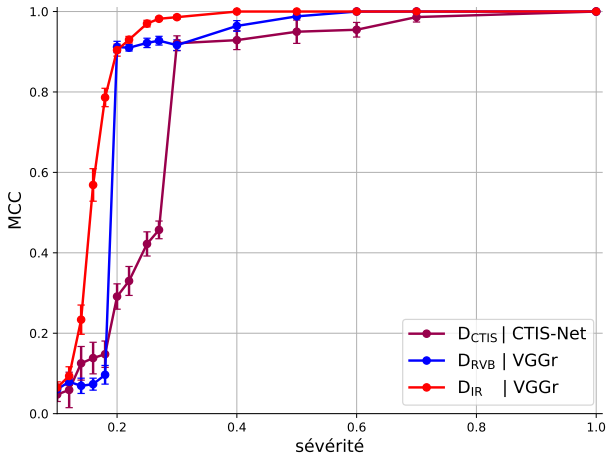


FIGURE 4.21 – Comparaison entre les résultats d'apprentissage comprimé sur image CTIS et ceux d'apprentissages sur les capteurs RVB et IR.

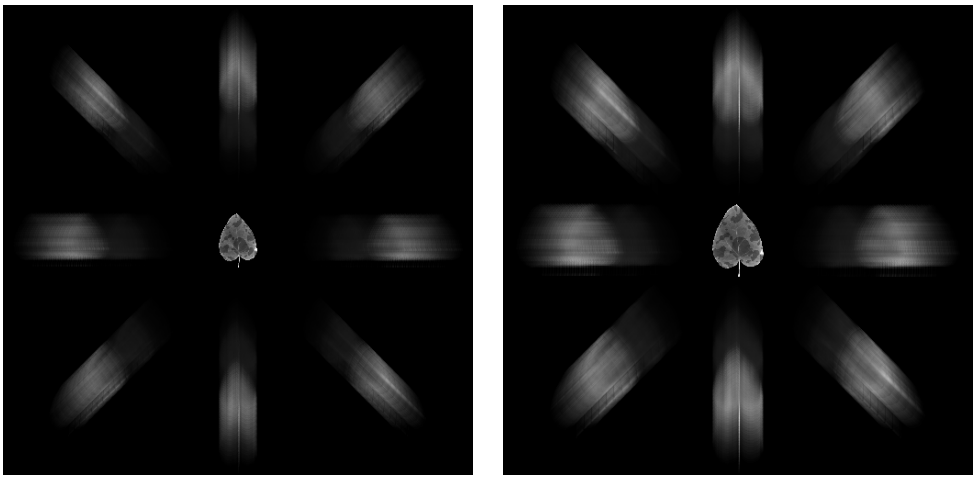
Il faut cependant noter que l'imagerie CTIS offre une plus grande polyvalence d'imagerie, en permettant d'obtenir de l'information dans une large gamme de longueurs d'onde. Autrement dit, les imageurs RVB et IR sont particulièrement appropriés lorsque l'on dispose d'information *a priori* sur les bandes spectrales pertinentes pour la classification attendue. Le cas de la tavelure répond à ces critères puisque les longueurs d'onde optimales sont connues dans la littérature. Nous avons intégré cette information dans le choix de la longueur d'onde employée dans le simulateur IR que nous avons implémenté (section 3.2.5). Il faut bien noter que l'obtention et la confirmation de cette information en premier lieu a nécessité plusieurs études [Delalieux et al., 2009b; Benoit et al., 2016] et l'usage de caméras hyperspectrales à balayage. Il existe de nombreuses applications où ces études n'ont pas été menées et de telles informations *a priori* sont inexistantes. Les imageurs plus

classiques utilisés « à l'aveugle » pourraient être dans ces cas beaucoup moins pertinents.

4.7 Une influence forte des paramètres optiques

Pour terminer l'étude de la valeur de l'apprentissage comprimé sur images CTIS, nous rappelons que la géométrie des images CTIS dépendait d'un certain nombre de paramètres liés aux choix du matériel optique (section 3.2.1). Nous avons intégré dans le simulateur CTIS nombre de ces paramètres, permettant d'obtenir théoriquement une grande variété d'images CTIS (figure 3.15).

La résolution spatiale d_0 était un des paramètres les plus importants du modèle. Ce paramètre était directement lié à la résolution spatiale du CTIS mais conditionnait également la qualité des projections spectrales (cf. section 3.2.1). Dans les cas limites du montage optique, une valeur de d_0 de 1 pixel menait à un ordre 0 inexistant et à un ordre 1 sans recouvrement spatial : en d'autres termes, à une résolution spatiale minimale et une résolution spectrale maximale. À l'inverse, une valeur de d_0 égale à la taille de l'image CTIS d maximisait la résolution spatiale mais ne permettait pas de décomposition spectrale. Des valeurs intermédiaires permettaient de faire varier le curseur entre résolutions spatiale et spectrale. En guise d'ouverture, nous présentons les principaux résultats obtenus dans ce chapitre en étudiant des images CTIS ayant un ordre 0 différent de 60 pixels. Pour fixer une nouvelle valeur, nous nous sommes basés sur une idée de « taille spatiale minimale » des lésions de tavelure. Pour le cas $d_0 = 60$ pixels, toute tache de tavelure ayant une aire inférieure à 15% de celle de la feuille sur laquelle nous la simulions n'était pas « discriminable » dans l'ordre 0 de l'image CTIS associée. Nous entendons par ce terme que l'aire de la tache était alors en deçà du pixel carré, et donc que le contraste causé par la lésion était nécessairement atténué car l'information lumineuse était moyennée avec celles des zones saines environnantes. Nous avons étudié le cas $d_0 = 80$ pixels qui permettait de diviser à peu près par deux cette taille spatiale minimale des lésions. La figure 4.22 présente l'aspect des images pour les deux valeurs de d_0. Nous pouvons y constater une taille plus grande de l'ordre 0, et donc une discriminabilité des lésions facilitée dans cet ordre, mais en contrepartie un recouvrement plus important des tranches au sein des projections de l'ordre 1.



(a) Une image CTIS avec $d_0 = 60$ pixels. (b) Une image CTIS avec $d_0 = 80$ pixels.

FIGURE 4.22 – Comparaison visuelle de l'impact du paramètre d_0, pour les valeurs 60 et 80 pixels.

Les résultats d'apprentissage pour les images créées avec $d_0 = 80$ pixels (figure 4.23) montraient tout d'abord que la valeur de d_0 avait un fort impact sur la qualité de la reconstruction : les apprentissages sur les cubes reconstruits par VGGr à partir d'images CTIS créées avec $d_0 = 80$ pixels étaient beaucoup plus performants que dans le cas $d_0 = 60$ pixels pour les sévérités les plus basses.

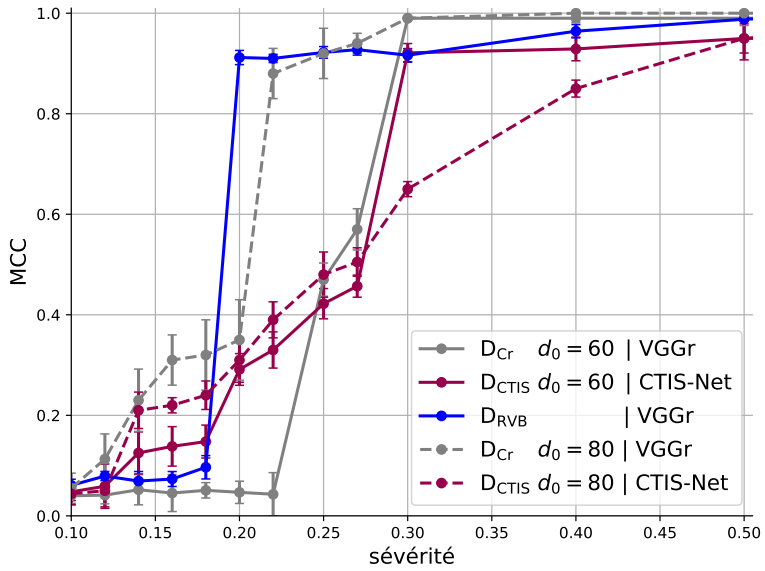


FIGURE 4.23 – Étude de l'impact du paramètre d_0 sur les performances des apprentissages. Contrairement à toutes les autres figures de ce chapitre, nous avons ici limité la gamme de sévérité à [0,1, 0,5] afin d'améliorer la visibilité des résultats, les performances au-delà étant similaires à celles obtenues à la sévérité 0,5.

Ce paramètre pouvait donc rebattre les cartes concernant la pertinence de la reconstruction de cubes par rapport à l'apprentissage comprimé. Il faut cependant noter qu'une reconstruction d'un cube de 80 pixels de large demandait en moyenne environ une fois et demie plus de temps qu'un cube de 60 pixels.

Concernant les performances de l'apprentissage comprimé, nous avons observé un double effet en comparant les deux courbes relatives aux performances de CTIS-Net. Nous savions par l'étude par ablation s'attachant à la fusion des ordres que dans notre cas cette architecture se basait sur un ordre ou sur l'autre en fonction de la valeur du paramètre « sévérité » (figure 4.18). En travaillant sur des images où l'ordre 0 était de taille 80 pixels au lieu de 60, nous avons constaté que les performances du réseau augmentaient pour les sévérités les plus faibles, pour lequel le réseau se basait sur l'ordre 0, et inversement pour les sévérités les plus élevées. Ces résultats étaient cohérents avec l'effet attendu d'augmenter la valeur de d_0 qui facilitait l'exploitation de l'information de l'ordre 0 au détriment de celle de l'ordre 1. Ainsi, avec une taille d'ordre 0 plus grande, l'imagerie CTIS devenait compétitive vis-à-vis de l'imagerie RVB dans la gamme de sévérité la plus faible. Il fallait cependant noter que les performances du CTIS dans cette gamme étaient loin d'être suffisantes pour être pertinentes industriellement. Cette expérience soulignait néanmoins que le compromis de résolution spatio-spectral conditionnait la performance de l'apprentissage sur des images CTIS par rapport à celles obtenues avec des imageurs plus classiques.

4.8 Conclusion

Dans ce chapitre, nous avons étendu pour la première fois le principe d'apprentissage comprimé à l'imagerie CTIS. En effet, nous avons montré, dans le cadre de l'application agronomique que nous avions définie, qu'il était possible pour un réseau de neurones générique d'effectuer une tâche de classification en se basant directement sur l'image CTIS (figure 4.8) plutôt que sur le cube reconstruit. De plus, nous avons mis au point une architecture dédiée à la structure des images CTIS, en nous basant en particulier sur une séparation des ordres afin de leur appliquer un traitement séparé. Les résultats obtenus étaient substantiellement meilleurs que ceux obtenus par

un réseau générique, et proches de ceux obtenus lors de l'approche classique, ce qui appuyait la possibilité de s'affranchir de l'étape chronophage de reconstruction pour analyser un signal CTIS. Le succès de l'apprentissage comprimé pouvait paraitre contre-intuitif *a priori* tant l'information spatiale de la scène semblait dégradée et son information spectrale brouillée dans une image CTIS. Ces résultats mettaient en évidence les différences d'interprétation entre le cerveau humain et les réseaux de neurones informatiques, et rappelaient que l'anthropomorphisme que nous attribuons parfois à ces algorithmes est bien souvent plus limité qu'on ne le croit.

Nous estimons que quelques améliorations auraient pu être apportées aux protocoles d'entraînement que nous avons établis (section 4.1.2). Il aurait en particulier été bénéfique d'inclure à la recherche par grille quelques hyperparamètres fixés arbitrairement tels que celui de l'optimiseur et celui des couches concernées par le pré-entraînement sur l'ILSVRC. De plus, nous considérons que les conditions sous lesquelles nous avons conduit les apprentissages sur les cubes reconstruits étaient légèrement moins favorables que celles relatives à l'apprentissage comprimé. En effet, la recherche par grille que nous avons menée pour fixer la dimension de profondeur des noyaux convolutifs VGG-3D était incomplète, puisque nous avons effectué une recherche par grille spécifique à ce paramètre (section 4.2.2) au lieu de réaliser une nouvelle recherche concernant tous les hyperparamètres de l'architecture en incluant ce dernier. Nous pensons cependant que cette approximation n'affaiblit pas significativement les conclusions que nous avons formulées dans ce chapitre.

Les conditions contrôlées des expérimentations de ce chapitre étaient pertinentes pour l'étude d'un système d'imagerie relativement nouveau. Cependant, dans le cadre d'applications industrielles telles que celles que mènent Carbon Bee, l'échelle pertinente pour la détection de maladies de plantes est bien souvent plus grande que celle de la feuille isolée. Les algorithmes destinés à une intégration dans un produit commercialisé sont appliqués à l'échelle d'un ensemble de plantes [Abdelghafour et al., 2020], voire du champ tout entier [Kerkech et al., 2020], et ce en conditions réelles. Ces algorithmes se basent sur des systèmes imageurs dont la pertinence est reconnue par la communauté des chercheur ·euses en sciences végétales, tels que les imageurs IR [Jones, 2004; Mahlein, 2016]. Dans le chapitre suivant, nous nous intéressons aux défis rencontrés dans le cadre d'une détection de tavelure sous ces conditions difficiles.

Chapitre 5

Simulation d'images pour alléger la charge d'annotation d'images réelles

Dans ce chapitre, nous nous sommes intéressés à la détection de lésions de tavelure en conditions industrielles. Nous avons pour cela acquis un jeu d'images représentant un ensemble de pommiers tavelés, en lumière IR. Nous avons associé à ce jeu une tâche de segmentation, qui est une opération alignée avec les objectifs industriels de l'agriculture de précision. Si la précision requise par cette tâche rendait la détection en elle-même plus ardue, il était par ailleurs particulièrement difficile d'annoter les données pour constituer un jeu d'entraînement en premier lieu. Nous nous sommes penchés sur ce défi qui nous a paru particulièrement impérieux. Nous croyions à la valeur des données simulées pour réduire cette charge d'annotation, et nous les avons employé ici en complément d'entraînements menés sur des données réelles et destinés à des prédictions également sur données réelles. Nous présentons dans ce chapitre des simulations innovantes spécifiques à notre application, basées sur les tendances les plus récentes en sciences végétales. Nous avons exploré par ailleurs les différentes façons d'intégrer ces données simulées aux entraînements conduits sur données réelles.

Sommaire

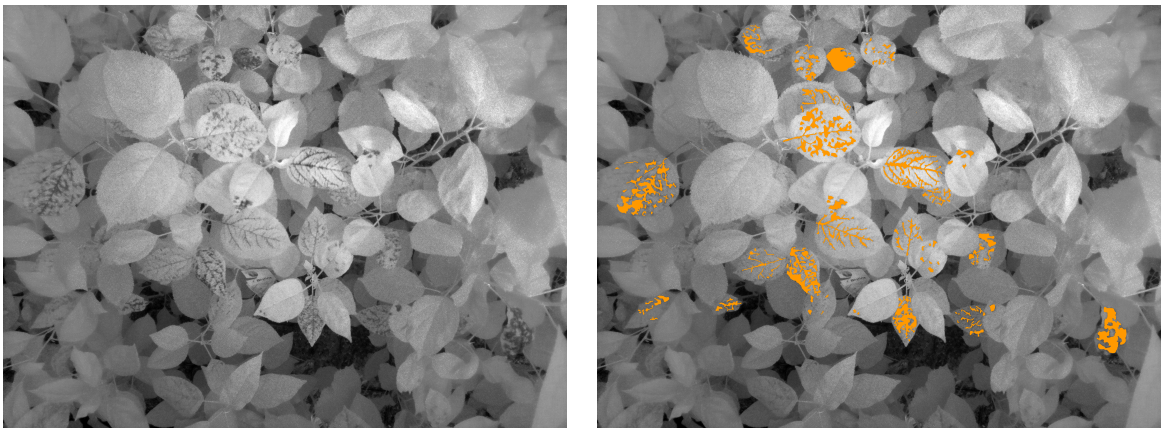
5.1 Une annotation chronophage à réduire 92
5.1.1 Un jeu difficile en conditions réelles 92
5.1.2 L'apport potentiel des données simulées 93
5.1.3 Une architecture spécifique pour effectuer une segmentation 94
5.1.4 Premiers résultats de segmentation : une performance « saturée » 96
5.2 Les différentes catégories de simulateurs 98
5.2.1 Augmentation d'images du jeu de données 98
5.2.2 Génération d'images à partir de modèles 100
5.3 Simulateurs implémentés pour notre cas d'étude 101
5.3.1 Déformation de données 101
5.3.2 GAN 103
5.3.3 Modèle de canopée 106
5.4 Résultats : des simulateurs efficaces 111
5.4.1 L'intégration des données simulées 111
5.4.2 L'impact des différents simulateurs 113
5.4.3 Les données simulées permettent d'aller au-delà des données réelles... 117
5.4.4 ... mais ne sont pas si « gratuites » que cela 118
5.5 Conclusion 119

5.1 Une annotation chronophage à réduire

5.1.1 Un jeu difficile en conditions réelles

Nous avons créé un jeu de données pour réaliser une détection de tavelure en acquérant des images de plants de *Malus pumilla* dans les serres de l'IRHS. Ces plants avaient été inoculés avec *V. inaequalis* quatorze jours avant l'acquisition. Les plants étaient placés proches les uns des autres, les feuilles formant une canopée. Cette structure était typique à la fois des environnements de recherche en sciences végétales et des plantations en champ. L'acquisition s'est déroulée en avril, à midi, sous conditions ensoleillées. Elle fut réalisée avec le capteur IR de la caméra Carbon Bee, portée manuellement environ un mètre au-dessus des plants, pointée vers le sol. Le gain de la caméra était réglé de façon automatique. Sept images de dimension 1944×2592 pixels furent acquises de cette manière, couvrant l'intégralité des plants disponibles, sans recouvrement entre les images. Nous avons appelé $D_{original}$ ce jeu de données. Nous avons tiré au hasard cinq de ces images pour constituer le bloc d'entraînement, une pour le bloc de validation et une pour le bloc de test de $D_{original}$.

Nous avons associé à ce jeu une tâche de segmentation des lésions de tavelure. Pour ce faire, nous avons procédé à une annotation en assignant à chaque pixel des images une étiquette « tavelure » ou « non tavelure ». Ainsi, bien que le nombre d'images du jeu pouvait paraître faible, chacun des pixels de ces images constituait en réalité une unité d'entraînement pour un algorithme d'apprentissage. Une image de $D_{original}$ et son annotation sont présentées figure 5.1.



(a) Image. (b) Annotation.

FIGURE 5.1 – (a) Une image du bloc d'entraînement de $D_{original}$. (b) L'annotation au niveau du pixel que nous avons réalisée pour cette image. Les pixels orange correspondent aux pixels étiquetés « tavelure », les autres à ceux étiquetés « non tavelure ». Dans tout ce chapitre, le contraste et la luminosité des images (réelles et simulées) présentant des canopées de feuilles ont été augmentés pour faciliter la visualisation.

En plus des difficultés communes aux jeux de données en condition réelles (illumination non homogène, feuilles orientées selon des angles variés, etc.), plusieurs caractéristiques de $D_{original}$ rendaient particulièrement ardue la tâche de segmentation qui lui était associée. Premièrement, le positionnement resserré des plants entraînait de nombreux recouvrements partiels entre les feuilles. Ensuite, l'échelonnage des feuilles selon leur position sur les tiges des plants entraînait des conditions d'illumination différentes pour les feuilles en fonction de leur distance à la source de lumière et des jeux d'ombres causés par les feuilles situées sur les couches supérieures. De plus, les différences d'illumination étaient accentuées par un effet dit de vignettage du capteur, c'est-à-dire que la zone centrale de l'image était plus lumineuse que sa périphérie [Zheng et al., 2008]. Concernant les lésions de tavelure, bien que leur localisation pouvait paraître aisée à l'œil nu, certains facteurs compliquaient cette détection. Les lésions étaient des structures de tailles

variables en fonction de la proximité de la feuille concernée au capteur, mais globalement de faible dimension. De plus, la frontière entre les lésions et les zones saines était souvent difficile à localiser. En effet, le procédé de prolifération du parasite (section 1.3.1) menait à une concentration variable d'agent en fonction de la proximité de la distance au point d'attaque originel. Ce comportement se manifestait visuellement par un contraste fort au centre de la lésion mais qui décroissait à mesure que l'on s'en éloignait. Enfin, certaines structures telles que les nervures des feuilles produisaient un contraste avec le reste de la feuille proche de celui créé par les lésions de tavelure.

Le jeu présentait par ailleurs une difficulté supplémentaire du point de vue de l'apprentissage en lui-même. Parmi les images de $D_{original}$, seuls 2% des pixels étaient étiquetés comme « tavelure ». De tels déséquilibres étaient néfastes pour les apprentissages : il était nécessaire pour une classification efficace que toutes les classes d'un jeu soient représentées de façon suffisante [Chawla, 2009]. Si une classe était excessivement sous-représentée, il y avait un fort risque que l'algorithme d'optimisation des poids du réseau mène à une frontière de décision qui « ignore » cette classe, car la prédiction erronée d'un faible nombre d'exemples n'était pas suffisamment pénalisante pour que d'autres configurations de poids soient explorées. Bien qu'il existe des méthodes pour modérer les problèmes engendrés par les jeux déséquilibrés [Yap et al., 2014], il est de bon aloi de mener des apprentissages sur des jeux les plus équilibrés possibles en premier lieu. Afin de réduire le déséquilibre des classes dans $D_{original}$, nous avons implémenté un type de sous-échantillonnage [Chawla, 2009] via un *tuilage* sélectif. Nous avons divisé les images de $D_{original}$ en imagettes, ou tuiles, de dimension 64×64 pixels, sans recouvrement [13]. Parmi les 6150 tuiles du bloc d'entraînement de $D_{original}$, nous avons conservé uniquement les 535 tuiles qui contenaient au moins un pixel étiqueté comme « tavelure ». Après ce tuilage, 10% des pixels du bloc étaient étiquetés comme tels. Nous appelons $D_{tuilé}$ le jeu constitué de ce bloc d'entraînement tuilé sous-échantillonné et des blocs tuilés complets de validation et de test.

5.1.2 L'apport potentiel des données simulées

Nous soulignons que le processus d'annotation de $D_{original}$ fut une entreprise particulièrement chronophage. D'une manière générale, l'annotation de données représente un coût considérable pour les entreprises qui implémentent des algorithmes d'apprentissage profond. Une technicienne était employée à plein temps à cette fin par Carbon Bee AgTech, et les ingénieur ·es agronomes de l'entreprise y consacraient aussi une partie de leur temps. Il était de plus nécessaire d'ajouter à ces coûts celui du processus d'acquisition des données brutes en premier lieu.

De nombreuses études se sont penchées sur la possibilité de réduire la charge d'annotation de données pour l'entraînement d'algorithmes d'apprentissage, que ce soit par le biais de méta-algorithmes tel que l'apprentissage actif [Settles, 2009; Nagasubramanian et al., 2020] ou bien via l'amélioration des moyens techniques d'annotation [Papadopoulos et al., 2014; Samiei et al., 2020]. Nous avons considéré pour notre part que la simulation de données pouvait ici aussi avoir un rôle à jouer. Bien que dans le champ de l'apprentissage automatique, certains apprentissages soient réalisés sur un bloc d'entraînement constitué uniquement de données simulées [Nikolenko, 2019], les performances de classification chutent la plupart du temps lorsque le réseau est utilisé en prédiction sur des images réelles [Tobin et al., 2017]. Cet écroulement de la performance est dû à « l'écart de la réalité » [Tremblay et al., 2018] entre les caractéristiques des images d'entraînement et celles de prédiction qui peuvent persistent malgré le soin amené à rendre les images simulées le plus réalistes possibles. Aussi dans de nombreuses études les données simulées sont-elles utilisées

13. Il est à noter que l'information d'une image pouvait être partiellement détruite au cours d'un tuilage. En particulier, le positionnement relatif d'objets dans l'image était perdu lorsque ceux-ci étaient répartis dans différentes tuiles. Cependant, la structure des images de $D_{original}$ limitait cet effet délétère. En effet, les images du jeu représentaient une multitude de plants inoculés séparément, dont le positionnement relatif n'amenait que peu information quant à la présence de tavelure. Nous avons donc considéré que le tuilage était une opération peu destructrice pour $D_{original}$ au vu de l'absence de structure globale dans les images du jeu.

en complément plutôt qu'en remplacement de données réelles pour mener à bien les entraînements. Dans ce cadre, les données simulées sont considérées comme des outils de régularisation [Goodfellow et al., 2016] et permettent d'augmenter la variabilité des caractéristiques présentes dans un bloc d'entraînement, en particulier dans le cas de classes déséquilibrées [Chawla et al., 2002].

Nous nous sommes alors posé la question suivante : des données simulées pouvaient-elles permettre de réduire la charge d'annotation tout en maintenant des performances proches de celles du jeu complet ? Pour répondre à cette question, nous avons d'abord évalué la performance de segmentation sur $D_{\text{tuilé}}$ ainsi que sur des versions de ce jeu aux blocs d'entraînement réduits. Nous présentons maintenant l'architecture du réseau employé et le protocole d'apprentissage que nous avons suivi pour mener à bien les segmentations.

5.1.3 Une architecture spécifique pour effectuer une segmentation

Les réseaux de neurones destinés à la classification, tel VGG que nous avons utilisé au chapitre 4, n'étaient pas adéquats pour réaliser une segmentation, car il s'agissait d'architectures conçues pour proposer un étiquetage unique pour l'intégralité de l'image qu'on leur fournissait en entrée. Nous nous sommes par conséquent tournés vers une autre catégorie d'architectures : les réseaux de segmentation. Nous présentons à présent un bref historique de ces réseaux ainsi que l'architecture retenue pour notre travail.

Les réseaux de classification ne permettaient pas de mener à bien des tâches de segmentation car ils compressaient l'information spatiale des caractéristiques par l'action des couches de *max-pool* et la détruisaient via les couches FC. Dans le cas d'une classification, cette perte d'information spatiale est sans conséquence puisque la réponse que l'on attend du réseau *in fine* est une valeur unique, mais pour réaliser une segmentation, cette information spatiale doit être conservée, ou plutôt reconstruite.

Le « réseau entièrement convolutif » [Long et al., 2015] fut le premier réseau de neurones destiné à la segmentation d'images. Ce réseau se basait sur l'architecture d'un réseau de classification mais, comme son nom l'indiquait, se passait de couches FC. Ces dernières étaient remplacées par une opération de redimensionnement afin de ramener les dimensions spatiales des cartes de caractéristiques à celles de l'image d'entrée. Ainsi, pour une image donnée en entrée de dimension $d_1 \times d_2$ pixels, la sortie de ce réseau était une carte de segmentation de dimension $d_1 \times d_2 \times n$ éléments avec n le nombre de classes différentes dans le bloc d'entraînement. Chaque pixel de la sortie représentait alors la probabilité d'appartenance aux classes de chacun des pixels de l'image d'entrée. La fonction de coût de l'entraînement était calculée en sommant les différences entre chacun des pixels de cette sortie et ceux de l'image annotée.

Les auteurs de ces travaux avaient implémenté le redimensionnement par une série de simples interpolations bilinéaires, mais aussi par des opérations paramétrisables, apprises par le réseau au cours de l'entraînement. Il s'agissait de couches convolutives dites « transposées » [Dumoulin and Visin, 2016] avec un pas (*stride* en anglais) supérieur à un, ce qui permettait de générer des sorties aux dimensions spatiales plus grandes que les entrées. Les auteurs de [Noh et al., 2015] ont implémenté cette idée de redimensionnement paramétrisable en utilisant un ensemble de couches convolutives classiques entrecoupées de couches de « dégroupement » (*unpool* en anglais). À l'inverse des couches de *max-pool*, les couches d'*unpool* augmentaient la taille des cartes de caractéristiques qu'on leur fournissait. L'architecture proposée, DeconvNet, était composée pour moitié d'un ensemble de blocs convolutifs identique à celui de VGG, et pour moitié du même nombre de blocs dits « déconvolutifs » constitués de couches convolutives et de couches d'*unpool*. Dans la première moitié du réseau, les caractéristiques de l'image étaient extraites et réduites spatialement jusqu'à une compression maximale au centre du réseau que l'on appelait l'espace

latent, comme l'aurait fait un réseau de classification. Dans la seconde moitié, cet espace latent était converti en une image aux mêmes dimensions spatiales que l'image d'entrée via l'action des blocs déconvolutifs. Les couches convolutives de ces blocs apprenaient à associer les caractéristiques de l'espace latent à des classes, et ce à différentes échelles de l'image, similairement à la partie convolutive du réseau. Les opérations d'*unpool* inversaient les opérations de *max-pool* auxquelles elles étaient associées, permettant ainsi de reconstruire couche par couche la localisation spatiale des caractéristiques. Le réseau U-Net [Ronneberger et al., 2015], contemporain à DeconvNet et très utilisé dans la communauté de l'imagerie, est basé sur des principes architecturaux similaires.

Dans ce travail, nous avons utilisé le réseau SegNet [Badrinarayanan et al., 2017], un réseau très proche de DeconvNet qui n'en différait que par l'architecture des couches liées à l'espace latent. SegNet proposait une réduction des dimensions spatiales de l'image moins importante que DeconvNet dans cet espace, ce qui le rendait d'après ses auteurs à la fois plus performant et plus aisé à entraîner. Son architecture est présentée figure. 5.2.

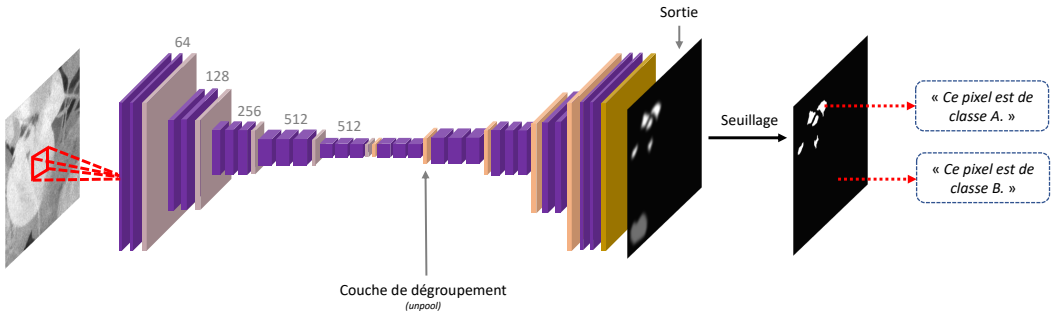


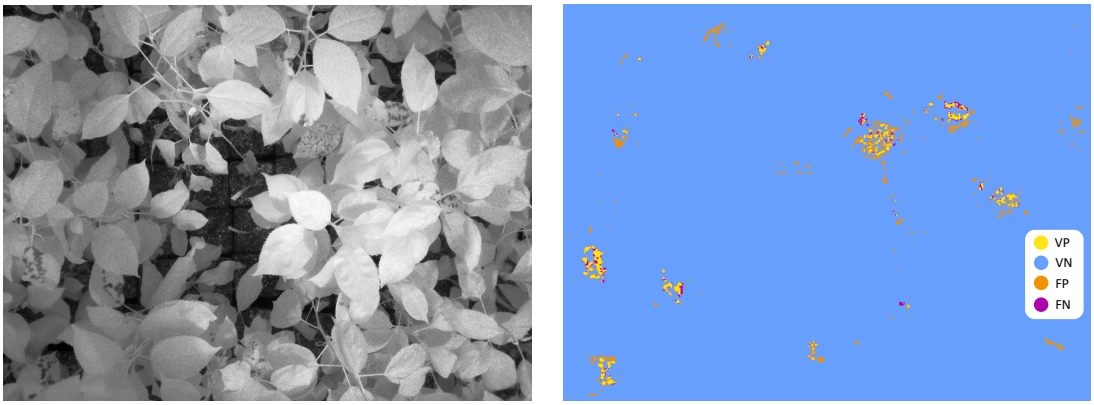
FIGURE 5.2 – Architecture de SegNet. L'architecture des bloc convolutifs est identique à celle de VGG, et celle des blocs « déconvolutifs » est calquée sur cette architecture.

Nous avons choisi de travailler avec cette architecture pour les mêmes raisons que nous avions sélectionné l'architecture VGG pour mener à bien les travaux présentés dans le chapitre 4 : au moment de la mise en place de ces travaux, SegNet nous paraissait proposer le meilleur compromis entre ses performances (il constituait l'état de l'art sur un certain nombre de *benchmarks* de segmentation) et son ancrage dans la communauté de l'apprentissage automatique. Depuis, plusieurs innovations ont été apportées au champ de la segmentation. Nous pouvons citer en particulier le réseau DeepLab [Chen et al., 2017] qui intégrait plusieurs modifications pour faciliter la reconstruction de l'information spatiale : un nouveau type de convolution pour augmenter le champ de vue des neurones [Holschneider et al., 1990] et la concaténation à chaque couche convolutive des résultats de plusieurs couches avec des tailles de noyaux différentes [Szegedy et al., 2015]. Ils proposaient en outre une opération d'affinage des sorties obtenues [Krähenbühl and Koltun, 2011].

Pour mener à bien l'entraînement, nous avons suivi quasiment le même protocole d'apprentissage que celui présenté à la section 4.1.2. Au vu du déséquilibre d'effectifs entre les deux classes qui persistait dans le jeu $D_{\text{tuilé}}$ malgré l'opération de tuilage sélectif, nous avons de plus implémenté un « équilibrage de classes » (*class balancing* en anglais). Cet équilibrage consistait à pondérer la fonction de coût d'entraînement avec des coefficients qui dépendaient de la classe du pixel sur laquelle cette fonction était calculée. Nous avons implémenté l'équilibrage par fréquence médiane [Badrinarayanan et al., 2017] pour établir ces coefficients en les fixant comme l'inverse de la fréquence relative de la classe concernée dans le jeu de données. Ainsi, le coût de prédiction erronée d'un pixel étiqueté « tavelure » était environ dix fois plus élevé que celui d'une prédiction erronée d'un pixel étiqueté « non tavelure ». Cet équilibrage permettait de compenser aux yeux de l'optimiseur le faible effectif des pixels étiquetés « tavelure » en leur conférant un fort impact sur la fonction de coût.

5.1.4 Premiers résultats de segmentation : une performance « saturée »

La segmentation du jeu $D_{tuilé}$ avec SegNet menait à une performance de 0,570±0,008. La figure 5.3 présente les résultats de la prédiction sur l'image de test. La figure 5.4 présente ces mêmes résultats sur une zone restreinte de l'image de test afin de permettre une visualisation plus fine.



(a) L'image dont l'ensemble des tuiles constituait le bloc de test de $D_{tuilé}$.

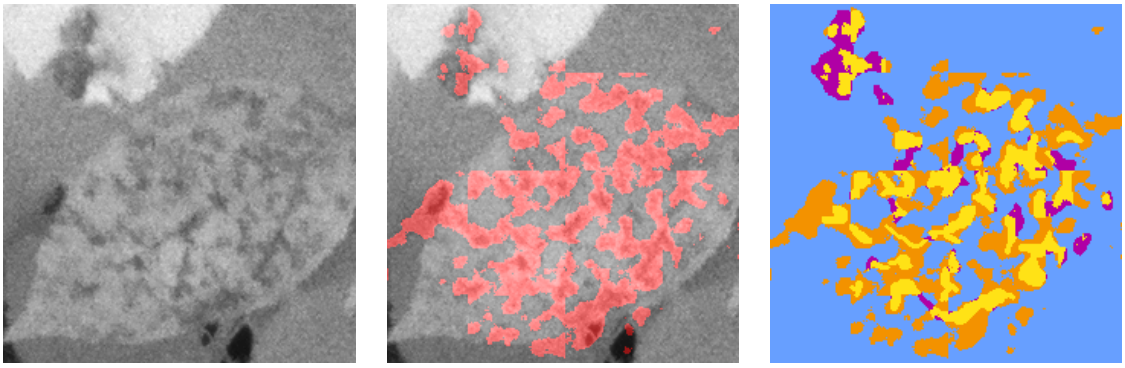
(b) Résultats de la prédiction du réseau à l'échelle de l'image : la couleur de chaque pixel dépend de s'il s'agit d'un VP, d'un VN, d'un FP ou d'un FN.

FIGURE 5.3 – Résultats de la prédiction sur le bloc de test de $D_{tuilé}$ par SegNet entraîné sur ce jeu de données.

La métrique de précision, c'est-à-dire le ratio $\frac{VP}{(VP+FP)}$, qui mesurait parmi tous les pixels prédits comme « tavelure » la proportion qui était étiquetée comme telle, était de 0,34. Un certain nombre de faux positifs étaient dus à des zones entières incorrectement prédites comme des lésions de tavelure, par exemple des jeux d'ombres sur les feuilles ou des portions du sol. Une autre partie était due à une sur-détection de tavelure autour des lésions véritables. Il est possible que l'équilibrage de classes ait été en cause pour ces faux positifs-ci. En effet, le coût artificiellement accru pour le réseau de prédire de façon erronée un pixel étiqueté « tavelure » a peut-être mené l'optimiseur à des choix excessivement « prudents » en particulier dans des zones indécises tels que les alentours des lésions. La métrique de rappel, c'est-à-dire le ratio $\frac{VP}{(VP+FN)}$ qui mesurait parmi tous les pixels étiquetés « tavelure » la proportion qui a été prédite comme telle était de 0,65. Les faux négatifs étaient regroupés dans des zones relativement peu étendues. Nous pouvons noter en particulier que le réseau n'avait commis aucun oubli à l'échelle de la feuille : pour toute feuille dont au moins certains pixels étaient étiquetés « tavelure », au moins certains de ces pixels étaient prédits comme tels. Ces résultats de prédiction pouvaient être considérés comme satisfaisants ou non suivant l'application qui leur était associée. Ils auraient traduit par exemple un apprentissage convenable pour une application de pulvérisation de produit phytosanitaire localisée, sans nécessité d'évaluer la gravité de l'infection.

Pour évaluer l'impact de la quantité de données annotées sur les performances de l'entraînement, nous avons mené une étude qui consistait à entraîner le réseau avec des portions réduites du bloc d'entraînement de $D_{tuilé}$ (figure 5.5). Plus précisément, nous avons commencé par créer à partir du bloc d'entraînement de $D_{tuilé}$ un bloc d'entraînement réduit en ne conservant aléatoirement que 5% des tuiles. Nous appelons $D_{tuilé}^{0,05}$ ce bloc d'entraînement réduit [14] associé aux blocs de validation et de test complets de $D_{tuilé}$. Nous avons ensuite créé à partir de $D_{tuilé}^{0,05}$ le jeu $D_{tuilé}^{0,10}$ en ajoutant à son bloc d'entraînement 5% des tuiles du bloc d'entraînement de $D_{tuilé}$ qui n'avaient pas été utilisées pour $D_{tuilé}^{0,05}$. Nous avons procédé de façon analogue pour la construction d'un ensemble

14. Nous soulignons pour éloigner toute confusion que nous reprenons dans ce chapitre la notation D_y^x employée au chapitre 4, mais que la valeur x indique ici la taille du bloc d'entraînement du jeu et non la valeur de « sévérité » qui paramétrisait les données simulées au chapitre précédent.



(a) Portion de l'image du bloc de test de $D_{tuilé}$.

(b) Pixels prédits comme « tavelure » par le réseau, en rouge, superposés à l'image (a).

(c) Résultat de la prédiction sur cette (a). Les couleurs sont identiques à celles de la sous-figure 5.3 (b).

FIGURE 5.4 – Agrandissement d'une partie de l'image du bloc de test, et résultats de la prédiction associés.

de jeux réduits imbriqués les uns dans les autres (figure 5.5, droite), jusqu'à la création de $D^{1}_{tuilé}$ qui correspondait à $D_{tuilé}$. Plus formellement, pour deux jeux $D^{p_1}_{tuilé}$ et $D^{p_2}_{tuilé}$ avec $p_1 < p_2$, alors $D^{p_1}_{tuilé} \in D^{p_2}_{tuilé}$. Pour tous les jeux $D^{p}_{tuilé}$, les blocs de validation et de test étaient identiques à ceux de $D_{tuilé}$. Les différents proportions p employées et le nombre de tuiles correspondant dans les blocs d'entraînement sont précisés dans le tableau 5.1.

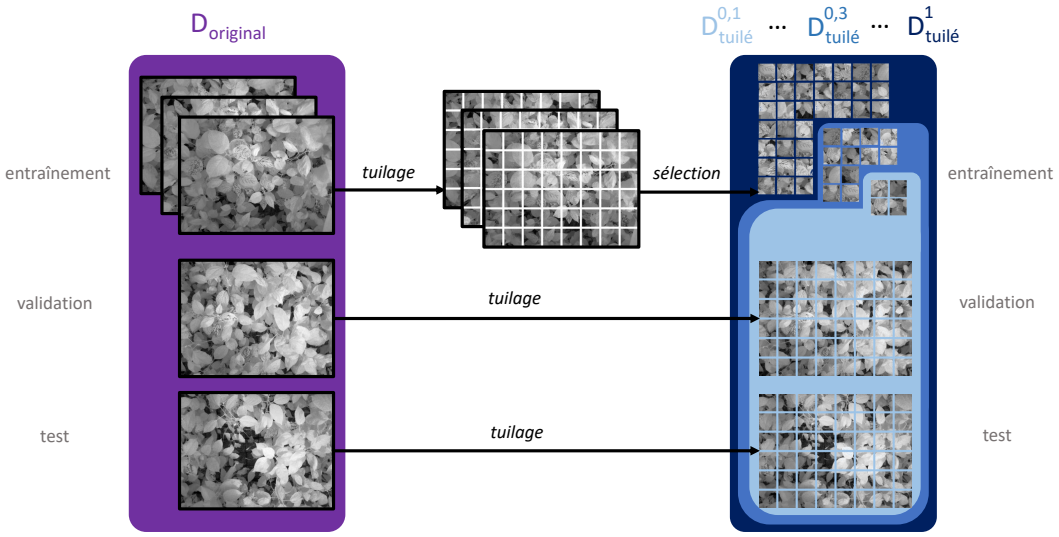


FIGURE 5.5 – Création des jeux $D^{p}_{tuilé}$ à partir de $D_{original}$.

Proportion conservée	0,02	0,05	0,1	0,2	0,3	0,4	0,5	0,75	1
Nombre d'images dans le bloc d'entraînement	11	27	54	107	161	214	268	401	535

TABLEAU 5.1 – Caractéristiques des jeux $D^{p}_{tuilé}$ créés.

Les performances d'apprentissage sur l'ensemble des jeux $D^{p}_{tuilé}$ sont présentées figure 5.6. Conformément aux principes de l'apprentissage automatique, la performance du réseau croissait lorsque le bloc d'entraînement était enrichi. Par ailleurs, nous pouvons observer qu'il était plus bénéfique pour la performance d'ajouter une quantité d'images donnée au bloc d'entraînement lorsque ce dernier était restreint que lorsqu'il était plus fourni. En d'autres termes, il y avait un effet de « saturation » de la performance en fonction du nombre de données disponibles pour

l'entraînement. En particulier, la taille du bloc d'entraînement de $D^1_{\text{tuilé}}$ correspondait à une configuration considérablement « saturée ». Cela indiquait donc que la quantité d'annotation que nous avions réalisée était suffisante pour notre application, et que des annotations supplémentaires ne semblaient pas pouvoir améliorer substantiellement les performances du réseau. Par contre, l'existence de cette performance « saturée » nous a amenés à nous pencher sur la possibilité d'annoter uniquement une portion réduite du jeu et d'intégrer à l'entraînement des données simulées afin d'atteindre cette performance saturée permise par l'annotation du jeu complet.

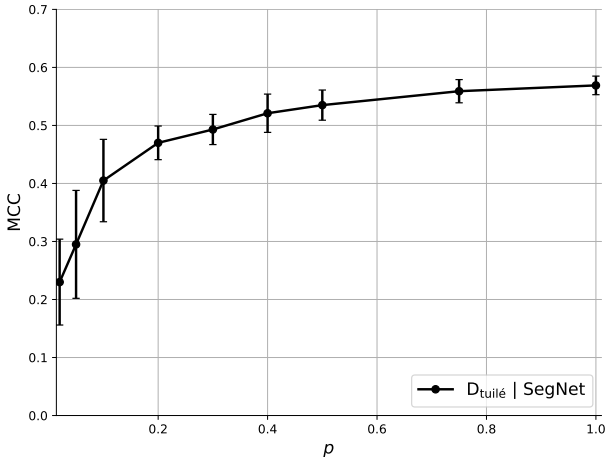


FIGURE 5.6 – Performance de SegNet sur les jeux $D^p_{\text{tuilé}}$.

Le reste de ce chapitre est consacré aux différentes méthodes de données simulées que nous avons implémentées à cette fin. En effet, il existe dans la littérature plusieurs familles de simulateurs de données, mais pas, à notre connaissance, d'étude comparative de l'apport de celles-ci, au moins dans le domaine des sciences végétales. Aussi avons-nous décidé d'implémenter chacune de ces familles de méthodes et de proposer pour la première fois une comparaison de leur apport sur un même cas d'usage de sciences végétales.

5.2 Les différentes catégories de simulateurs

Nous présentons maintenant une littérature concernant les différentes catégories de simulateurs d'images existants, en mettant l'accent sur leur utilisation dans le domaine des sciences végétales.

5.2.1 Augmentation d'images du jeu de données

Une première famille de simulateurs consiste à générer des données en modifiant des images du jeu de données concerné. On appelle cette action l'*augmentation* de données. On trouve deux classes de méthodes dans cette famille : celles basées sur des déformations des images et celles basées sur des réseaux génératifs.

Déformation d'images

L'implémentation la plus courante dans cette veine est appelée la *déformation* de données (*data warping* en anglais). Suivant les auteurs, ces techniques sont parfois aussi désignées sous le nom d'« augmentation de données ». Dans ce document, nous utiliserons plutôt ce terme pour toute méthode visant à créer des nouvelles données à partir des données initiales. La déformation de données consiste à appliquer des modifications géométriques simples aux images du jeu de données à augmenter : retournement horizontal ou vertical, rotation, déformation 3D, variation de

couleur, de contraste, etc. [Buslaev et al., 2020]. On trouve trace de ces déformations dès certaines des premières études d'apprentissage profond, mais ces techniques sont particulièrement mises en avant lors de l'étude séminale des auteurs de [Krizhevsky et al., 2012]. Dans cette étude, les déformations de données sont, avec d'autres techniques de régularisation, citées explicitement comme des implémentations nécessaires afin d'éviter le surapprentissage. Depuis, la déformation de données fait régulièrement partie des protocoles d'apprentissage profond, notamment lorsque les blocs d'entraînement sont de taille réduite [Wang et al., 2017]. Elle est intégrée à de nombreuses applications en sciences végétales, en adaptant les déformations appliquées aux spécificités du domaine [Pawara et al., 2017].

Réseaux génératifs

Au cours de cette dernière décennie, des techniques plus sophistiquées d'augmentation de données ont vu le jour, rendues possibles par l'emploi de réseaux de neurones. Une architecture particulièrement utilisée à cette fin est le réseau antagoniste génératif (*Generative Adversarial Network* en anglais, ou GAN) [Goodfellow et al., 2014] (figure 5.7).

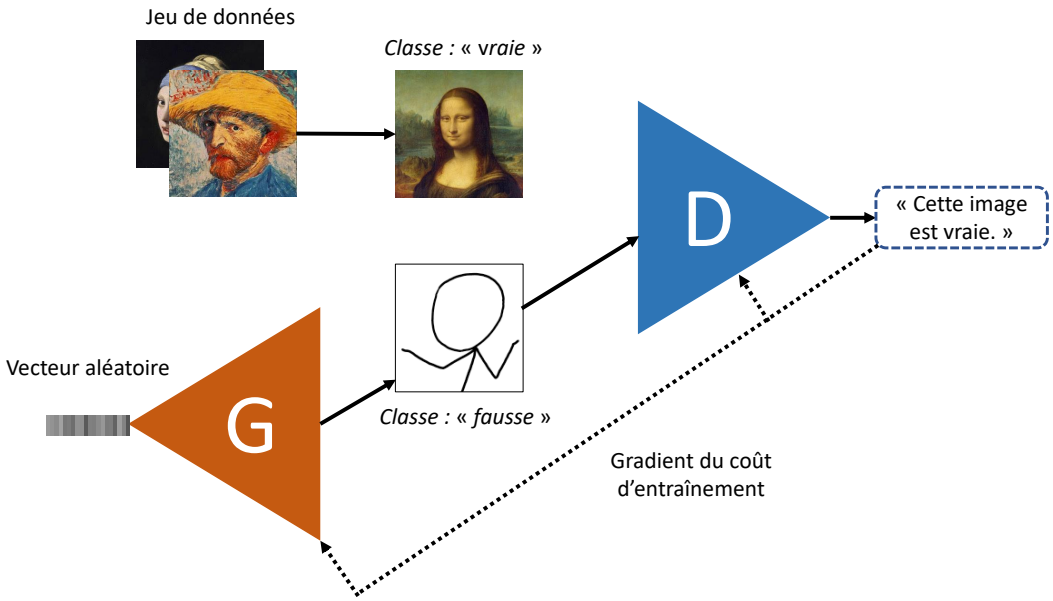


FIGURE 5.7 – Architecture et entraînement d'un GAN. Cette figure illustre le fonctionnement général d'un GAN : l'architecture des réseaux générateur et discriminateur, désignés respectivement par les lettres « G » et « D » n'est pas précisée. Nous représentons dans cette figure une itération de l'entraînement où une image du générateur, étiquetée « fausse », est présentée au discriminateur : le gradient de l'erreur de ce dernier remonte alors dans les deux réseaux. Lors d'autres itérations (non représentées), ce sont des images du jeu réel qui sont présentées au discriminateur. Source des images : articles wikipedia des tableaux concernés pour les images « vraies », `xkcd.com` pour l'image « fausse ».

On désigne par ce nom un ensemble de deux réseaux. Le premier est appelé « générateur » : il est utilisé pour créer une image à partir d'un vecteur aléatoire. Le second est appelé « discriminateur » : il est utilisé pour produire une valeur binaire à partir d'une image. Les deux réseaux sont entraînés conjointement de la façon suivante. On fournit au générateur un vecteur aléatoire qui change à chaque itération de l'apprentissage. On fournit au discriminateur des images provenant du jeu de données que l'on étiquette « vraies » et des images produites par le générateur que l'on étiquette « fausses ». Le discriminateur propose une étiquette « vraie » ou « fausse » pour chaque image, et est entraîné à distinguer les images du jeu de données de celles créées par le générateur. Sa fonction de coût associée vaut 1 s'il se trompe systématiquement et 0 s'il parvient à une distinction parfaite. Au début de l'entraînement, les deux réseaux sont initialisés avec des poids aléatoires. En conséquence, le générateur produit des images de « bruit », c'est-à-dire où chaque pixel prend une valeur aléatoire,

et le discriminateur ne sait pas distinguer les images vraies des fausses. Le gradient, qui résulte de l'erreur que le discriminateur commet, est propagé dans ce dernier. L'astuce des GANs est que ce gradient est aussi propagé dans le générateur, dont la fonction de coût est définie comme l'opposée de celle du discriminateur. En d'autres termes, le générateur est entraîné à « tromper » le discriminateur, et génère par conséquent des images de plus en plus proches des images du jeu de données. On emploie souvent pour décrire le fonctionnement d'un GAN l'analogie de l'association d'un faussaire qui peint des faux tableaux et d'un policier chargé de les repérer, tous deux s'améliorant à leur tâche au fil du temps [Goodfellow et al., 2014].

Lorsqu'un GAN a convergé, les images créées par le générateur sont en théorie indiscernables des images du jeu de données du point de vue du discriminateur. Il faut cependant noter que l'entraînement de deux réseaux en opposition est beaucoup plus instable que l'entraînement d'un seul réseau associé à une fonction de coût unique. Les GANs comptent parmi les architectures les plus difficiles à faire converger et des études entières ont été menées pour améliorer leur stabilité [Salimans et al., 2016; Arjovsky and Bottou, 2017]. En effet, pour qu'un GAN converge, il faut que les deux réseaux antagonistes progressent à la même vitesse. Il survient souvent le déséquilibre suivant : à un instant donné de l'entraînement, le discriminateur est bien meilleur à détecter les vraies images des fausses que le générateur ne l'est pour créer des fausses images. Si ce déséquilibre devient extrême, l'erreur du discriminateur est de zéro, et aucun gradient n'est fourni au générateur, qui cesse alors de progresser : le GAN est en « échec » [Arjovsky et al., 2017].

Lorsque les GANs ont convergé convenablement, ils sont exploités dans de nombreuses applications, y compris à des fins d'augmentation de données, afin de produire des images proches du bloc d'entraînement, mais qui pourraient compléter les manquements de celui-ci [Nikolenko, 2019]. Les versions initiales des architectures de GANs menaient cependant à une reproduction trop fidèle du bloc d'entraînement et n'apportaient pas un gain significatif à l'apprentissage [Arjovsky et al., 2017]. Une architecture développée en parallèle, le GAN conditionnel [Mirza and Osindero, 2014], permettait de produire des images en fonction d'une étiquette fournie par l'utilisateur. Cette architecture a été un pas significatif dans la génération d'images véritablement différentes du jeu de données. En parallèle, les travaux de [Gatys et al., 2016] ont montré qu'un réseau de classification pouvait être entraîné à séparer le contenu d'une image de son « style », c'est-à-dire des informations d'apparence multi-échelle sans notion de spatialité. Le « transfert de style neuronal » désigne l'application du style d'une image au contenu d'une autre. Cette idée a été intégrée dans des GANs [Isola et al., 2017] à des fins d'augmentation de données. L'architecture la plus utilisée aujourd'hui [Yi et al., 2019] dans cette veine est le CycleGAN [Zhu et al., 2017] qui permet d'appliquer le style des images d'un domaine aux images d'un autre, sans avoir eu besoin au cours de l'entraînement d'images appariées de chaque domaine. Dans le champ des sciences végétales, des GANs conditionnels simples ont montré des bons résultats, en particulier à des fins de phénotypage [Valerio Giuffrida et al., 2017; Zhu et al., 2020]. Récemment, des architectures basées sur le StyleGAN [Arsenovic et al., 2019] et le CycleGAN [Tian et al., 2019; Nazki et al., 2020] ont été employées à des fins de génération d'images de plantes malades.

5.2.2 Génération d'images à partir de modèles

À la différence des méthodes d'augmentation de données, une autre famille de simulateurs vise à créer des données à partir de sources externes au jeu de données concerné. Beaucoup d'entre eux ont été développés spécifiquement pour une tâche de reconnaissance d'un objet donné. Parmi les applications les plus étudiées, on retrouve des simulateurs de texte [Gupta et al., 2016], de visages [Richardson et al., 2016] et de silhouettes [Ragheb et al., 2008]. Les principes de fonctionnement de ces simulateurs sont très variés car ils reposent sur des *pipelines* personnalisés, adaptés au domaine d'application concerné. Par exemple, les auteurs de [Gupta et al., 2016] proposent une incrustation de textes dans des images déjà existantes, tandis que les auteurs de [Richardson et al., 2016] travaillent sur des modèles de visages déformables. Certains simulateurs sont volontairement

plus polyvalents, permettant la génération de multiples objets différents [Chang et al., 2015]. Dans le domaine des plantes, la génération d'objets synthétiques individuels a été facilitée par la présence de modèles de plantes antérieurs à l'ère de l'apprentissage profond [Prusinkiewicz and Runions, 2012]. Ces modèles existaient à des échelles variées, des cellules jusqu'aux écosystèmes [Pradal et al., 2008; Ubbens et al., 2018]. Ils étaient utilisés pour une grande variété d'analyses : étude de la croissance des plantes, de la distribution du poids des feuilles, des transferts d'eau, des études génotypiques, etc. [Prusinkiewicz, 2004]. En particulier, les L-systèmes [Lindenmayer, 1968] ont été une avancée mathématique majeure pour les modélisations à l'échelle de la plante. Il s'agit d'une grammaire itérative proposée pour modéliser le développement de plantes ramifiées. Des extensions multi-échelle de cette grammaire ont été proposées par la suite [Godin and Caraglio, 1998]. Des plantes ainsi simulées ont pu être employées pour améliorer les performances d'algorithmes d'apprentissage [Benoit et al., 2014; Isokane et al., 2018; Ubbens et al., 2018].

Pour d'autres domaines d'application, des simulateurs ont été développés dans le but de créer des environnements synthétiques, c'est-à-dire de véritables mondes virtuels tridimensionnels où des caméras virtuelles pouvaient acquérir des images à partir de points de vue variés. On trouve ce type de simulateurs principalement dans des applications de conduite autonome [Ros et al., 2016], de déplacement de robots [Handa et al., 2016] et de comptage de foules [Wang et al., 2019]. Ces simulateurs se basaient souvent sur des moteurs graphiques utilisés dans les jeux vidéo, par exemple celui de *Grand Theft Auto* [Wang et al., 2019] ou celui d'*Unreal* [Dosovitskiy et al., 2017]. Dans le domaine des sciences végétales, certaines études se sont penchées sur la génération de scènes tridimensionnelles de plantations accompagnées d'une caméra afin d'améliorer la performance de tâches de segmentation [Di Cicco et al., 2017; Barth et al., 2018].

5.3 Simulateurs implémentés pour notre cas d'étude

Nous avons pour notre cas d'étude exploré les différentes catégories de simulateurs de données qui constituaient l'état de l'art à l'époque des travaux, en les adaptant pour la problématique étudiée. Ainsi avons-nous implémenté des déformations de données spécifiques aux modalités d'acquisition des images de canopée. Nous avons par ailleurs adapté le fonctionnement d'un GAN afin que celui-ci puisse produire des images utiles dans le cadre d'une segmentation. Ces deux simulateurs augmentaient directement les images des jeux $D^{p}_{\text{tuilé}}$. Enfin, nous avons développé un simulateur « modèle » dédié qui permettait une génération de feuilles tavelées réparties en canopée, imitant ainsi les images de D_{original}. Cette section présente les détails de ces trois simulateurs.

5.3.1 Déformation de données

Afin d'adapter la déformation de données à notre tâche, nous avons repris des transformations employées très régulièrement dans les protocoles d'apprentissage profond. Nous avons basé notre sélection des transformations à appliquer en partie sur la littérature du domaine des sciences végétales [Pawara et al., 2017] mais surtout par rapport aux conditions d'acquisition des images du jeu de données. Nous avons choisi ces déformations car elles représentaient des transformations qui pouvaient relier deux images données du jeu. Les déformations de données servaient ainsi à introduire une variabilité que l'on supposait potentiellement présente dans le bloc de test (et pour le champ d'application véritable du modèle entraîné si celui-ci était déployé industriellement) mais pas nécessairement dans le bloc d'entraînement. En d'autres termes, les déformations implémentées n'étaient pas choisies aveuglément mais correspondaient à des connaissances *a priori* sur la structure des scènes acquises [15].

15. Il est intéressant de noter qu'il existe un paradigme plus récent de déformations de données, la randomisation de domaine [Tobin et al., 2017], dont le but n'est pas de représenter des transformations réalistes. Dans ce domaine, de nombreuses déformations irréalistes du point de vue des données d'entraînement sont appliquées afin d'aiguiller le réseau à se concentrer sur l' « essentiel » des données. Le but de ces déformations est que lors de la prédiction, les différences entre les images simulées et les images réelles soient identifiées comme simplement une autre variation

Le tableau 5.2 liste les déformations que nous avons sélectionnées pour ces raisons. La figure 5.8 présente des illustrations de l'effet de ces déformations pour une image de feuille. Les retournements, rotations étaient justifiés par la grande variabilité des positions des feuilles par rapport au capteur. Les redimensionnements traduisaient la distance variable des feuilles par rapport au capteur. Les transformations de perspective représentaient les différences d'angles d'acquisition des feuilles selon leur position par rapport au capteur. Le flou gaussien était justifié par les différences de netteté entre des feuilles présentes à différents niveaux de la tige acquises par un capteur à focale fixe.

Déformation	Paramètre associé	Valeur possibles du paramètre
Retournement	Direction	{horizontal, vertical}
Rotation	Angle	$\{\frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$
Agrandissement	Coefficient d'agrandissement	[0,8, 1,2]
Flou	σ du flou gaussien	{1, 2}
Perspective	Coordonnées destination	cf. texte

TABLEAU 5.2 – Déformations choisies avec leurs gammes de paramètres associées.

Pour une image donnée, chacune de ces déformations était réalisée avec une probabilité 0,5. Plusieurs de ces transformations pouvaient donc être combinées. Des paramètres, présentés tableau 5.2, étaient associés à ces déformations. A chaque application de ces déformations, les valeurs de ces paramètres étaient tirés au hasard. Lorsque les valeurs possibles étaient un ensemble fini d'éléments, la valeur du paramètre était tirée aléatoirement parmi ces éléments. Lorsque les valeurs possibles étaient un intervalle de valeurs, la valeur du paramètre était tirée aléatoirement uniformément dans cet intervalle.

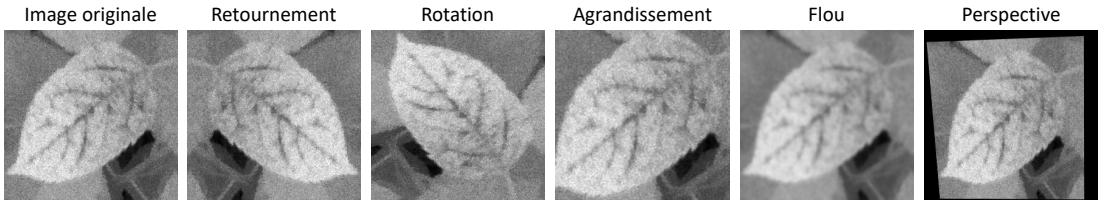


FIGURE 5.8 – Illustration des différentes déformations que nous avons implémentées.

La quasi-totalité de ces déformations constituaient des opérations basiques du domaine du traitement d'images. Nous explicitons ici seulement le procédé de changement de perspective. Nous avons procédé à cette transformation via une opération d'homographie. L'homographie est la transformation géométrique qui relie les images d'une même scène plane acquises par deux caméras différentes. Les homographies sont par conséquent souvent utilisées dans le domaine de la vision par ordinateur à des fins de recalage, en particulier pour des tâches de « cartographie et localisation simultanées » (*Simultaneous Localization And Mapping* en anglais) [Chatila and Laumond, 1985]. Pour notre application, nous nous sommes servis de l'opération d'homographie pour représenter un simple effet de perspective. Il y avait plusieurs manières de paramétriser l'opération d'homographie. Nous avons suivi la procédure recommandée par OpenCV [16] qui consistait à choisir dans l'image source quatre points d'intérêt (les « coordonnées source ») puis de spécifier les coordonnées que nous souhaitions que ces points d'intérêt occupent dans l'image après

aléatoire que le réseau a appris à ignorer. Nous n'explorons pas l'utilisation de données déformées dans ce cadre dans ce manuscrit.

16. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_geometric_transformations/py_geometric_transformations.html.

la transformation (les « coordonnées destination »). Une fois ces deux groupes de coordonnées précisés, OpenCV fournissait des fonctions capables de calculer automatiquement la matrice d'homographie qu'il était nécessaire d'appliquer à l'image source afin que les points d'intérêt de l'image transformée coïncident effectivement avec les positions spécifiées. Pour notre application, nous avons défini les coordonnées source comme les quatre coins de l'image originale de dimension $d \times d$, et les coordonnées destination comme des points tirés aléatoirement dans un cercle d'un rayon $\frac{d}{5}$ des coordonnées sources (figure 5.9).

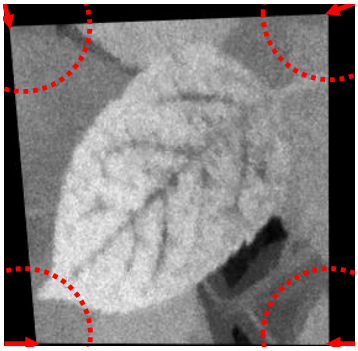


FIGURE 5.9 – Illustration de la transformation de perspective réalisée par homographie. Les flèches rouges relient les coordonnées source aux coordonnées destination.

5.3.2 GAN

Nous avons implémenté un GAN pour générer des nouvelles images à partir des images de $D^{p}_{\text{tuilé}}$. La différence avec un GAN classique est qu'afin de pouvoir exploiter ces images à des fins de segmentation, nous avons généré dans le même temps les annotations correspondant à ces images. En pratique, nous avons fourni au GAN des images à deux canaux, dont le premier correspondait à l'image IR et le second à l'annotation que nous en avions faite, et l'entraînions à produire les deux conjointement [Neff et al., 2017; Pollastri et al., 2020]. Ce protocole était inspiré d'une étude annexe des travaux de [Isola et al., 2017] où les auteurs montraient que les GANs étaient capables de générer des cartes de segmentation à partir d'images.

Nous avons utilisé comme architecture le GAN convolutif profond (*Deep Convolutional GAN* en anglais, ou DCGAN) [Radford et al., 2015]. Dans cette configuration, le générateur et le discriminateur étaient des CNNs, adaptés donc à la génération et la discrimination d'images. Le discriminateur était composé de couches convolutives au *stride* de deux qui réduisaient la dimension des images et permettaient ainsi de se passer de couches de *max-pool*. Le générateur suivait une architecture analogue, composé de couches convolutives transposées à *stride* de deux. Nous avons suivi le protocole du GAN Wasserstein [Arjovsky et al., 2017] pour l'entraînement, qui constituait à l'époque de ces travaux [17] l'état de l'art concernant l'entraînement de GANs. Pour améliorer davantage la stabilité de l'entraînement, nous avons implémenté des stratégies dédiées issues de la littérature [Chintala et al., 2016; Salimans et al., 2016]. Parmi celles-ci, la seule qui permit une amélioration notable de la stabilité de la convergence fut de doubler le nombre de filtres du générateur par rapport à celui du discriminateur. Cette modification était cohérente avec le cas pathologique décrit section 5.2.1 où le générateur échouait complètement à « tromper » le discriminateur. Par conséquent, nous avons conservé cette modification uniquement. L'architecture finale que nous avons utilisée est illustrée figure 5.10.

17. Les travaux présentés dans ce chapitre sont les premiers qui ont été menés au cours de cette thèse, au cours du premier semestre 2018.

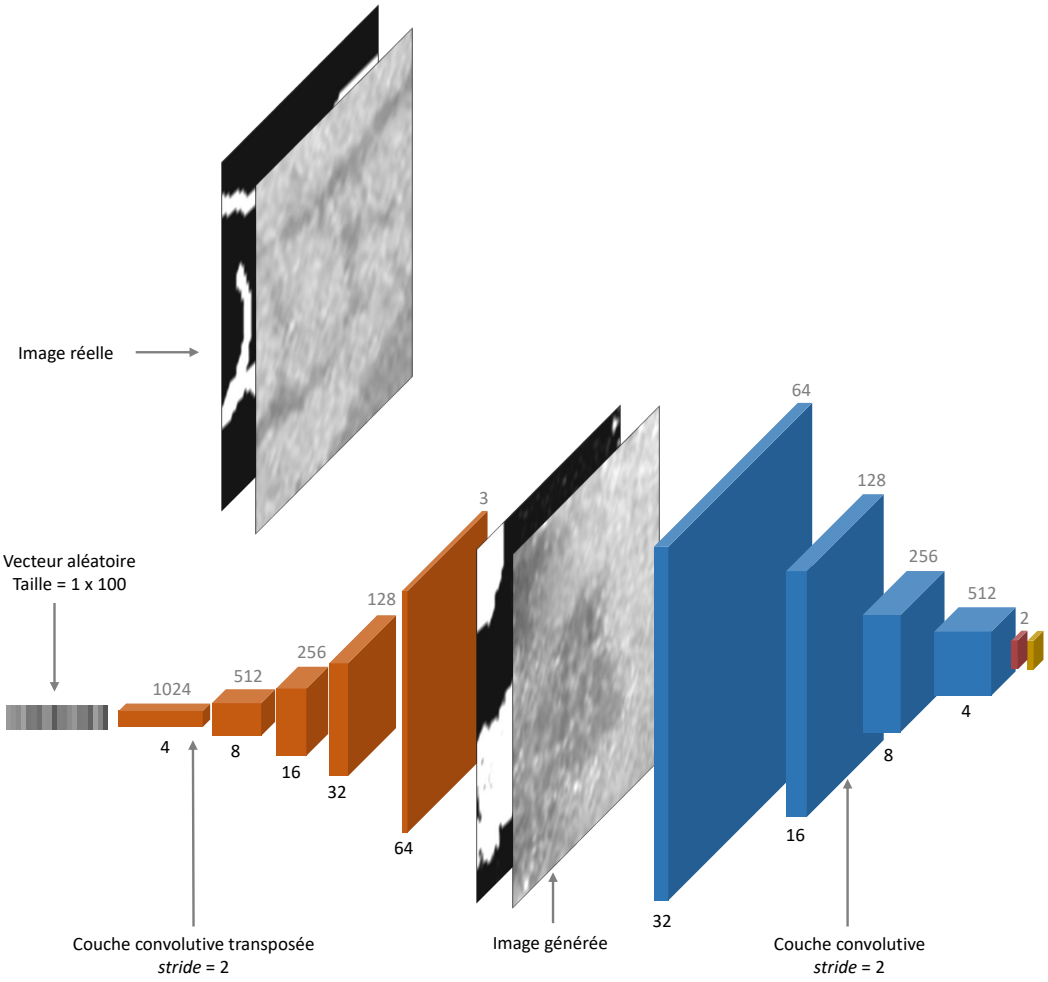
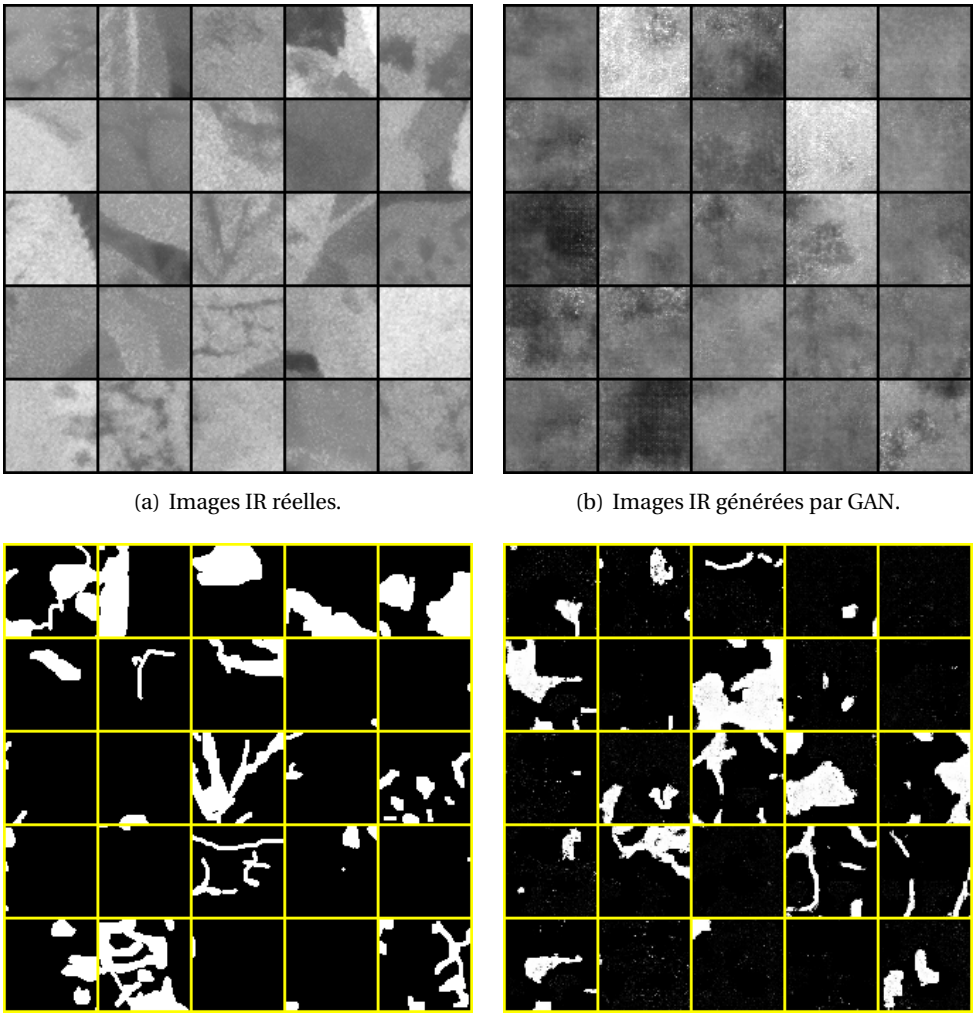


FIGURE 5.10 – Architecture du GAN utilisé. Les tailles des couches représentent les dimensions spatiales des cartes de caractéristiques qu'elles prennent en entrée. Puisque le pas de ces couches est fixé à deux, les dimensions de sortie des cartes ne sont pas les mêmes que celles d'entrée. Les dimensions spatiales des images en sortie de couches sont indiquées en dessous des couches concernées.

Au cours de l'entraînement, nous fournissions au GAN des images formées par la concaténation selon l'axe des canaux d'une image de $D^{p}_{tuilé}$ et de son annotation (figure 5.10, haut). Une image était donc de dimension $64 \times 64 \times 2$ pixels, et le GAN était contraint de créer des images de même taille. Le protocole d'entraînement, c'est-à-dire la normalisation des images, la mise en lots, la fonction de coût pour le discriminateur, etc., était identique à celui décrit pour nos expériences de classification (section 4.1.2). À la différence de l'entraînement d'un réseau de classification ou de segmentation cependant, l'évaluation de convergence du GAN n'était pas triviale car elle ne pouvait pas se mesurer par la valeur d'une fonction de coût unique ou d'une métrique de performance simple. Intuitivement, nous avions voulu nous baser sur l'erreur de classification du discriminateur puisqu'une valeur de 0,5 de celle-ci signifiait que ce dernier ne parvenait pas à séparer les images créées par le générateur de celles du jeu de données. Cependant, nous nous sommes aperçus que cela ne signifiait pas que ces images générées étaient effectivement proches de celles du jeu de données : le discriminateur pouvait à ce stade être encore mal entraîné à sa tâche et ne pas parvenir à prédire comme « fausses » des images extrêmement bruitées. Ainsi, la métrique de coût du discriminateur n'était pas un indicateur adéquat pour évaluer la convergence du GAN. Au moment de la réalisation de ces travaux, certaines métriques commençaient à être développées spécifiquement à cette fin telles que l'*Inception Score* [Salimans et al., 2016; Heusel et al., 2017], qui était basé sur la capacité d'un réseau annexe à classifier les images générées dans certaines classes prédéfinies de la même manière qu'il le ferait pour des images réelles. Pour notre travail,

nous nous sommes simplement basés sur une évaluation subjective de la proximité visuelle entre les images générées et les images de $D^{p}_{\text{tuilé}}$ [18]. Plus précisément, nous avons mené une pré-étude où nous avons entraîné un GAN *ad infinitum* et visualisé les images générées au fil des itérations. Cette étude nous a poussé à fixer le nombre optimal d'itérations à 100 000. Il est intéressant de noter qu'à ce moment de l'entraînement, la métrique de performance du discriminateur n'était pas de 0,5, mais variait plutôt entre 0,95 et 0,97. Ainsi, le générateur s'améliorait en parvenant à tromper, sur une proportion réduite d'images, un discriminateur très performant. Nous présentons figure 5.11 des exemples d'images générées par ce procédé.



(a) Images IR réelles.

(b) Images IR générées par GAN.

(c) Annotations réelles.

(d) Annotations générées par GAN.

FIGURE 5.11 – Exemples d'images générées par le GAN entraîné sur $D^{0,2}_{\text{tuilé}}$. Les images réelles étaient de dimension $64 \times 64 \times 2$ pixels. Les deux canaux de 25 de ces entrées tirées aléatoirement sont représentés dans les sous-figures (a) (canal « IR ») et (c) (canal « annotation »). Chaque image de la sous-figure (a) est associée à l'annotation située à la même position dans la sous-figure (c). La même représentation est utilisée pour présenter les images générées par le GAN, sous-figures (b) et (d).

18. Cette évaluation humaine de la qualité d' une image avait évidemment ses limites, d'autant plus lorsque, comme ici, les images générées étaient destinées non pas à être esthétiquement plaisantes ou réalistes aux yeux d'autres humains comme dans le domaine des *deep fakes* [Westerlund, 2019], mais à améliorer les performances d'apprentissage de réseaux. Or, il est connu, en particulier grâce aux études portant sur les exemples adversariaux, que des modifications portées à une image invisibles à l'œil humain pouvaient avoir un effet drastique sur la performance des réseaux qui la traitaient [Szegedy et al., 2013].

5.3.3 Modèle de canopée

Le simulateur « modèle » que nous avons développé permettait de créer des images de feuilles tavelées organisées en canopée, proches de celles du jeu $D_{original}$. Les simulateurs d'images de canopée contemporains à nos travaux qui généraient des scènes de plantations en trois dimensions étaient basés sur un grand nombre d'étapes et nécessitaient l'utilisation de logiciels sophistiqués tels que Blender [Di Cicco et al., 2017; Barth et al., 2018]. Nous avons volontairement implémenté un modèle plus simple, basé sur un script Python et une base d'images libre.

Il y avait deux hypothèses sous-jacentes fortes au fonctionnement de ce simulateur. Premièrement, nous avons considéré que le positionnement des feuilles des plants dans les images de $D_{original}$ pouvait être approximé par le placement itératif de feuilles les unes par dessus les autres, similairement au modèle dit de « feuilles mortes » [Lee et al., 2001]. Il a été prouvé que ce procédé simulait des propriétés statistiques proches de celles que l'on trouve dans des images naturelles, telles que l'invariance à l'échelle de plusieurs mesures statistiques [Ruderman and Bialek, 1993]. Deuxièmement, nous avons considéré que les lésions de tavelure pouvaient être caractérisées par un ensemble d'indicateurs statistiques simples.

L'algorithme 5.1 présente les étapes générales du modèle que nous avons développé (figure 5.12). Les étapes indiquées en bleu sont détaillées dans les sections suivantes de ce chapitre. Les étapes indiquées en violet sont identiques au simulateur de cubes tavelés hyperspectraux présenté à la section 3.1. En particulier, nous avons utilisé le jeu de données Leafsnap comme source d'images de feuilles saines, et l'algorithme de création de la distribution spatiale des lésions de tavelure était identique à celui présenté dans la section 3.1.4.

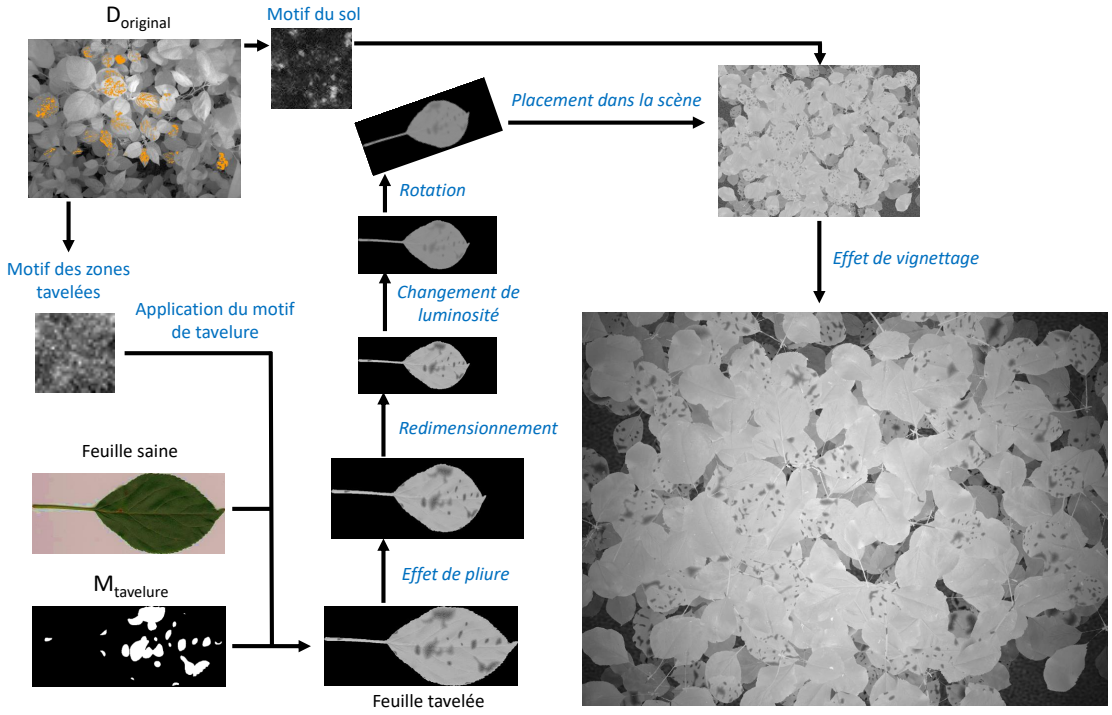


FIGURE 5.12 – Illustration de l'algorithme 5.1. Les étapes indiquées en bleu sont détaillées dans le texte.

Nous avons utilisé comme jeu d'images de canopées D le bloc d'entraînement de $D_{original}$. Nous avons fixé les dimensions de l'image de sortie égales à celles des images de $D_{original}$, c'est-à-dire $d_1 = 1944$ et $d_2 = 2592$. Nous avons fixé le nombre de plants n à 20 et la proportion de feuilles tavelées p à $\frac{1}{3}$ en concordance avec les images réelles. Nous détaillons à présent les différentes étapes de l'algorithme.

Algorithme 5.1 : Création d'une image de canopée de feuilles tavelées.

Entrées : un jeu d'images de feuilles saines RVB L, un jeu d'images de canopées D, les dimensions de l'image de sortie $d_1 \times d_2$ pixels, le nombre de plants n, la proportion de feuilles tavelées p.

Générer un motif de tavelure synthétique à partir des zones étiquetées comme « tavelure » dans les images de D.
Générer un motif de sol synthétique généré à partir des zones correspondant au sol dans les images de D.
Créer une image S vide de dimension $d_1 \times d_2$ pixels.
Remplir S du motif de sol synthétique.

nombre_de_plants = 0.
tant que *nombre_de_plants* < n **faire**

/* Création de la feuille tavelée. */
Tirer au hasard une image I du jeu L.
Convertir I en niveaux de gris.
Tirer un nombre aléatoire r dans [0,1].
/* Appliquer la tavelure. */
si $r < p$ **alors**
Générer un masque de lésions de tavelure $M_{tavelure}$.
Appliquer le motif de tavelure à l'endroit des lésions en utilisant $M_{tavelure}$ et le motif synthétique de tavelure.
fin
/* Déformation de la feuille. */
Simuler un effet de pliure pour I.
Calculer le placement de I dans S en fonction du nombre et de la position des feuilles déjà placées.
Redimensionner, modifier la luminosité et appliquer une rotation à I en fonction de son placement dans S.
Placer la feuille dans S.
Si un plant est terminé, incrémenter nombre_de_plants.
fin

Appliquer un effet de vignettage sur l'image.

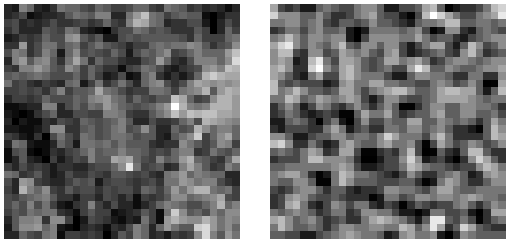
Sortie : une image de feuilles tavelées organisées en canopée, de dimension $d_1 \times d_2$ pixels.

Génération des motifs de tavelure et de sol

Contrairement au simulateur présenté à la section 3.1 où seule la variation moyenne d'intensité causée par les lésions sur la feuille était prise en compte, nous avons généré ici un motif de tavelure avec un niveau de réalisme plus élevé. Ce réalisme accru était guidé par la précision attendue dans la tâche de segmentation. Nous avons créé ce motif en suivant une approche procédurale similairement au procédé de création de bruit que nous avons implémenté à la section 4.5.

Pour obtenir le motif « modèle » duquel nous avons mesuré les valeurs des caractéristiques à ajuster, nous nous sommes concentrés sur les images du bloc d'entraînement de $D_{original}$. Nous avons rogné et conservé parmi celles-ci les zones étiquetées comme « tavelure » situées sur les feuilles appartenant à la couche supérieure de la canopée. Nous avons appelé l'ensemble de ces imagettes le jeu $D_{tavelure}$. Nous avons écarté de ce jeu toutes les imagettes dont les dimensions étaient inférieures à un carré de 10 pixels de côté. Nous avons calculé les moments d'ordre 1 et 2 ainsi que l'autocorrélation sur ces imagettes, calculé la moyenne de ces mesures, et ajusté un

bruit gaussien en conséquence. La figure 5.13 présente une comparaison d'un motif de tavelure réel et d'un motif synthétique ainsi créé. Nous avons généré un motif de sol en suivant la même procédure, en nous basant sur des imagettes de sol du bloc d'entraînement de $D_{original}$. Ce motif était utilisé pour remplir l'image simulée avant le placement des feuilles.



(a) Extrait de $D_{tavelure}$. (b) Tavelure synthétique.

FIGURE 5.13 – Comparaison entre les motifs de tavelure réel et synthétique. Le contraste et la luminosité ont été augmentés à des fins de visualisation.

Application du motif de tavelure

Nous avons généré pour chaque feuille une distribution spatiale des lésions $M_{tavelure}$ selon l'algorithme 3.2. Pour y appliquer le motif de tavelure, nous avons souhaité introduire un degré de réalisme supplémentaire par rapport au simulateur du chapitre 3 en modélisant l'effet « gradient » causé par le développement des lésions de tavelure dans le temps à partir d'un foyer d'infection. Nous avons modélisé cet effet en fixant la valeur des pixels appartenant à une lésion comme une somme pondérée des pixels du motif de tavelure et de ceux la feuille saine. La pondération de la texture de tavelure dans cette opération était décrite par une décroissance gaussienne qui dépendait de la distance du pixel au centre de cette lésion. L'ensemble du procédé est décrit dans l'algorithme 5.2 (figure 5.14). Nous avons fixé empiriquement la valeur de σ à 1.

Algorithme 5.2 : Création d'une feuille tavelée avec un effet « gradient ».

Entrées : une image de feuille saine I de dimension $d_1 \times d_2$ pixels, un motif de tavelure synthétique T de dimension $d_1 \times d_2$ pixels, la distribution spatiale de lésions $M_{tavelure}$ de dimension $d_1 \times d_2$ pixels, l'écart-type de la modulation spatiale gaussienne σ.

Créer par un seuillage de la couleur verte un masque $M_{feuille}$ où les pixels blancs correspondent aux pixels appartenant à la feuille dans I.

Appliquer pour chaque lésion de $M_{tavelure}$ individuellement une décroissance gaussienne, c'est-à-dire multiplier chaque pixel de la lésion par $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{l}{\sigma})^2}$, où l correspond à la distance de ce pixel par rapport au centre de masse de la lésion. Nous notons $M_{tavelure\ gradient}$ cette image.

Créer l'image $M_{sain\ gradient}$ en soustrayant $M_{tavelure\ gradient}$ à $M_{feuille}$.

Créer l'image tavelée par l'opération suivante : $I_{tavelure} = M_{sain\ gradient}.I + M_{tavelure\ gradient}.T$, où « . » désigne la multiplication pixel à pixel.

Sortie : une image de feuille tavelée avec un effet « gradient », de dimension $d_1 \times d_2$ pixels.

Effet de pliure

Les feuilles sont des organes dont la forme générale peut être considérée en première approximation comme un plan [Raabe et al., 2015], mais qui peuvent être en réalité significativement courbées [Rolland-Lagan et al., 2014] (figure 5.15).

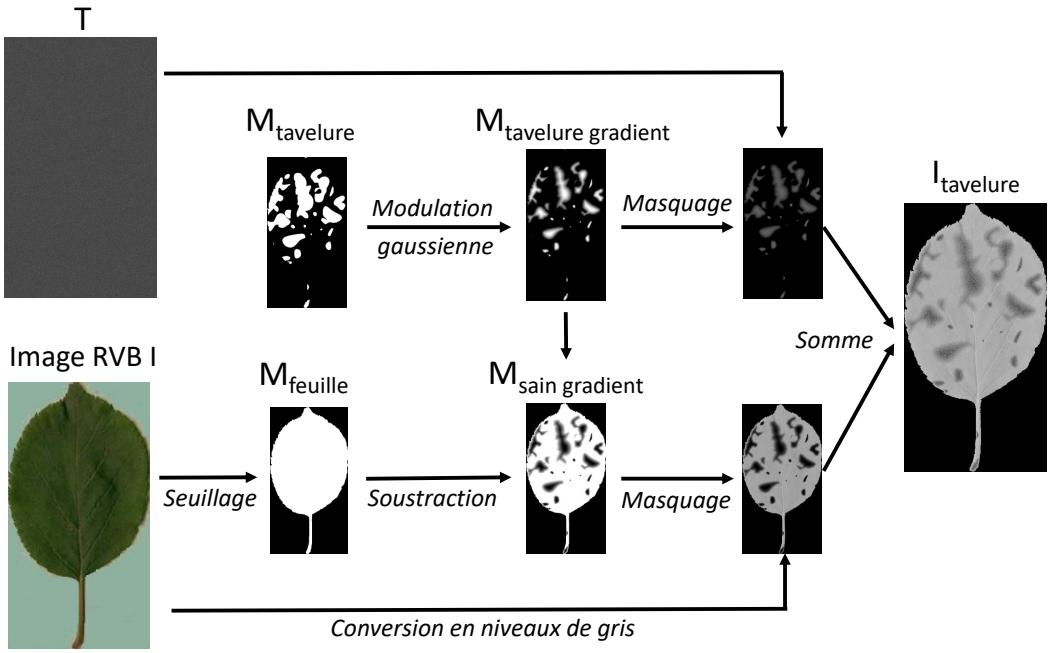


FIGURE 5.14 – Illustration de l'algorithme 5.2.

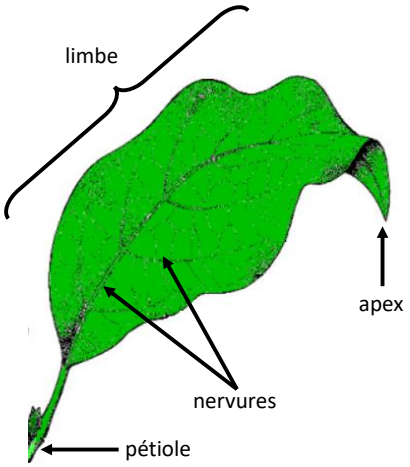


FIGURE 5.15 – Illustration d'une feuille courbée. Les principales structures anatomiques d'une feuille sont indiquées. Source : image adaptée de http://soutien67.free.fr/svt/vegetaux/reproduction.htm.

Dans les images de $D_{original}$, nous avions constaté que les feuilles étaient bombées sous le double effet de l'attache de leur pétiole à la tige du plant, orienté vers le ciel, et de la gravité qui entraînait le limbe vers le sol. Ainsi, une approximation plus correcte de la forme générale des feuilles du jeu était une concaténation de deux plans formant un angle entre eux. Puisque les images de Leafsnap représentaient des feuilles à plat posées sur un support, nous avons simulé cette pliure en appliquant une transformation de perspective à la moitié supérieure de la feuille.

Nous avons implémenté une homographie pour réaliser cette transformation de perspective (figure 5.16), selon le même procédé que celui suivi à la section 5.3.1. Pour une image de feuille donnée I, orientée vers le haut, nous considérions la moitié supérieure de cette image notée I_{haut}, de dimension $d_1 \times d_2$ pixels. Nous définissions un repère orthonormé dont l'origine était située en bas à gauche de I_{haut}, et nous associons aux quatre coins SO, SE, NE, NO de I_{haut} les coordonnées destination (0,0), $(0,d_2)$, $(\frac{3}{4}d_1, \frac{1}{4}d_2)$ et $(\frac{3}{4}d_1, \frac{3}{4}d_2)$. La matrice d'homographie était calculée et appliquée à I_{haut} pour obtenir l'image $I_{haut\ transformé}$. Nous remplacions alors la moitié supérieure de I par $I_{haut\ transformé}$.

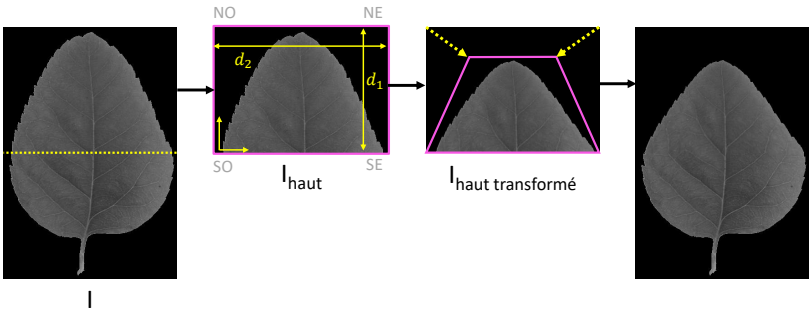


FIGURE 5.16 – Illustration de la transformation de perspective que nous appliquons aux feuilles. La dite transformation est effectuée entre les images I_{haut} et $I_{haut\ transformé}$.

Placement dans la scène

Afin de se rapprocher du positionnement spatial des feuilles entre elles dans les images de $D_{original}$, nous avons placé les feuilles simulées selon une distribution spatiale en « plants ». Nous avons placé des groupes de feuilles ensemble de façon à ce que les extrémités de leur pétioles soient à la même position, séparées par des rotations dont les angles simulaient la distribution des feuilles le long d'une même « tige », c'est-à-dire 2,4 radians [Zeng and Wang, 2009]. Nous avons simulé trois différents niveaux d'accroche le long de ces tiges. La taille et la luminosité des feuilles étaient modifiées en fonction du niveau d'accroche afin de simuler un effet de profondeur. Cette procédure est décrite plus formellement par l'algorithme 5.3.

Nous avons fixé les valeurs des paramètres de cet algorithme à celles précisées dans le tableau 5.3, en accord avec les images de $D_{original}$. La figure 5.17 illustre la création d'un plant. Comme spécifié dans l'algorithme, à l'échelle de l'image, nous avons d'abord ajouté les feuilles appartenant aux premiers niveaux de tous les plants, puis du deuxième, puis du troisième, afin de simuler la croissance simultanée des plants et les recouvrements qui s'en suivaient.

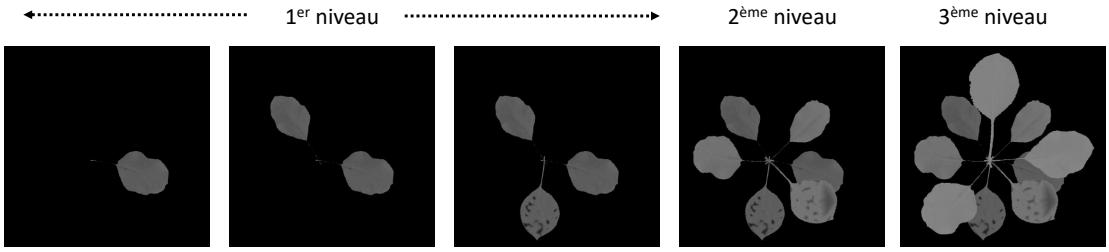


FIGURE 5.17 – Illustration de l'algorithme 5.3 pour un plant.

Effet de vignettage

Les images de $D_{original}$ souffraient d'un effet de vignettage. Nous avons représenté cet effet en appliquant une décroissance gaussienne de la lumière selon un procédé analogue à celui employé pour simuler un contraste « gaussien » des lésions. Plus formellement, nous avons multiplié tous les pixels de l'image par la valeur $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{l}{\sigma_v})^2}$ où l représentait la distance de ce pixel par rapport au centre de l'image. Nous avons fixé empiriquement la valeur de σ_v à 1300. La figure 5.18 illustre l'effet de vignettage appliqué.

Algorithme 5.3 : Placement des feuilles dans la scène.

Entrées : une image de la scène S de dimension $d_1 \times d_2$ pixels, un jeu d'images de feuilles L sur lesquelles des lésions de tavelure ont été simulées et où l'effet pliure a été appliqué, le nombre de plants n, les variations de luminosité $l = \{l_1, l_2, l_3\}$, les variations de taille $r = \{r_1, r_2, r_3\}$.

Générer n coordonnées correspondant aux « tiges » sous la forme $t_{plant} = (x_{plant}, y_{plant})$, avec x_{plant} et y_{plant} tirés indépendamment et uniformément dans [0,d_1] et [0,d_2], sous contrainte d'un critère de distance suffisant avec les tiges déjà existantes.

pour *plant* ∈ *[1,n]* **faire**
    Tirer un angle d'accroche initial α_{plant} aléatoirement dans [0,2π] radians.
fin

Nous modélisons chaque tige comme ayant trois niveaux de « points d'accroche » de feuilles. Nous remplissons S en ajoutant chaque niveau l'un après l'autre.
pour *niveau* ∈ *[1,3]* **faire**
    pour *plant* ∈ *[1,n]* **faire**
        Tirer le nombre de feuilles présent f à ce niveau pour ce plant, tiré uniformément dans [2,4].
        pour *feuille* ∈ *[1,f]* **faire**
            Tirer une feuille I du jeu L.
            Multiplier tous les pixels de I par l[niveau].
            Redimensionner I par un facteur r[niveau].
            Tourner I selon l'angle α_{plant}.
            Placer I de façon à ce que l'extrémité du pétiole de la feuille soit positionnée en t_{plant}.
            $\alpha_{plant} = \alpha_{plant} + 2{,}4$ radians.
        fin
    fin
fin

Sortie : une image S où ont été placées les feuilles sous forme de plants, de dimension $d_1 \times d_2$ pixels.

Nombre de plants n	20
Variations de luminosité l	{0,4, 0,7, 1}
Variations de taille r	{0,6, 0,8, 1}

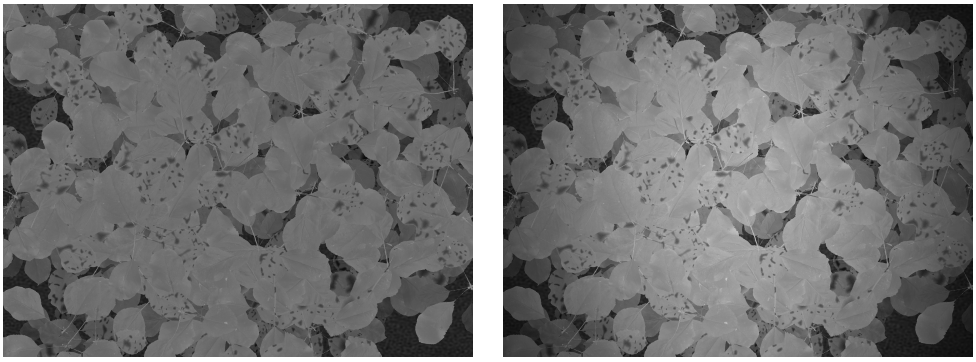
TABLEAU 5.3 – Valeurs des paramètres que nous avons utilisés pour l'algorithme 5.3.

5.4 Résultats : des simulateurs efficaces

5.4.1 L'intégration des données simulées

Les différents moyens d'intégration

Il existait plusieurs manières d'utiliser des données simulées en appui d'un entraînement sur données réelles. La voie la plus intuitive consistait à *rassembler* l'ensemble des données, c'est-à-dire créer un bloc d'entraînement contenant à la fois les données réelles et les données simulées. C'est ainsi que procédaient la plupart des études qui exploitaient des données simulées dans le domaine des sciences végétales [Valerio Giuffrida et al., 2017; Ubbens et al., 2018; Ward et al., 2018]. Nous avons proposé par ailleurs une autre voie, qui était bien moins souvent explorée dans le domaine des sciences végétales : l'utilisation des données simulées pour *pré-entraîner* des réseaux, voie suivie par exemple par les auteurs de [Barth et al., 2018]. Il s'agissait de réaliser un apprentissage



(a) Avant l'application de l'effet. (b) Après l'application de l'effet.

FIGURE 5.18 – Illustration de l'effet de vignettage.

par transfert (cf. section 4.1.2) en entraînant le réseau sur des données simulées dans un premier temps puis sur les données réelles dans un second temps. En ce sens, ce procédé était une extension des objectifs de l'apprentissage par transfert à partir de l'ILSVRC. Nous avons proposé ici de remplacer la tâche générique de vision qu'était l'ILSVRC par une tâche conçue spécifiquement pour être proche de la tâche réelle, en faisant l'hypothèse que ce pré-entraînement serait d'autant plus bénéfique pour l'apprentissage ainsi. Enfin, il existait même une troisième voie d'utilisation possible pour les données créées par déformation : la déformation « en ligne » (*online* en anglais) [Shorten and Khoshgoftaar, 2019]. Il s'agissait d'appliquer des déformations de données au bloc d'entraînement non pas avant l'entraînement proprement dit, mais au cours de celui-ci, en faisant varier les déformations appliquées à chaque époque [19].

Il n'existait pas à notre connaissance d'étude comparative de ces différentes façons d'intégrer les données simulées dans le champ des sciences végétales. Aussi avons-nous décidé d'implémenter chacune des voies proposées (rassemblement, pré-entraînement, en ligne) pour les différentes catégories de données simulées (déformation de données, GAN, modèle de canopée) afin de comparer les performances obtenues pour notre application. Afin de limiter le nombre d'expériences, nous avons dans un premier temps restreint cette étude au jeu $D^{0,2}_{\text{tuilé}}$, en faisant l'hypothèse que les tendances des résultats seraient extrapolables aux autres jeux $D^{p}_{\text{tuilé}}$. Nous détaillons à présent les protocoles d'entraînement suivis pour chacune des façons d'intégrer les données.

Protocoles d'intégration

Rassembler les données était une opération simple, mais dépendante d'un hyperparamètre important : la quantité relative de données simulées à ajouter au jeu de données réelles. Dans la littérature, cette quantité variait énormément selon les études. Nous avons trouvé des travaux où le nombre d'images simulées utilisé était significativement inférieur [Valerio Giuffrida et al., 2017], sensiblement égal [Ward et al., 2018], ou largement supérieur [Di Cicco et al., 2017] au nombre d'images réelles. Nous avons suivi la stratégie des auteurs de [Zhu et al., 2020] en menant plusieurs expériences avec les ratios « simulées sur réelles » suivants : 25, 100, 400 et 800%. Afin de ne pas surcharger les résultats, nous présentons uniquement dans les résultats d'apprentissage (tableau 5.4) les performances correspondant à la quantité de données optimale pour chaque expérience de « rassemblement ». Pour toutes ces expériences, nous avons intégré des données simulées uniquement au bloc d'entraînement du jeu, c'est-à-dire que les blocs de validation et de

19. Le terme *online* est aussi employé pour désigner un paradigme d'entraînement où de nouvelles données sont présentées régulièrement à l'algorithme qui apprend ainsi « en continu » [Fontenla-Romero et al., 2013]. Des méta-algorithmes d'augmentation de données inspirées de ce paradigme sont parfois désignées comme *online* [Tang et al., 2020]. Nous ne faisons pas référence à ces méthodes lorsque nous utilisons ce terme.

test restaient constitués uniquement d'images réelles.

Pour mener à bien des pré-entraînements à partir d'un simulateur donné, nous avons généré 3000 images à partir de celui-ci. Dans le cas des données issues du modèle de canopée, nous avons généré suffisamment d'images entières (c'est-à-dire de dimension 1944 × 2592 pixels) pour conserver 3000 tuiles contenant de la tavelure après l'opération de tuilage sélectif décrite section 5.1.1. Nous avons initialisé aléatoirement le réseau et l'avons entraîné dans un premier temps sur ces images simulées jusqu'à convergence. Puis, nous avons utilisé les poids obtenus comme initialisation du réseau pour un entraînement sur le jeu réel. Notons que dans ce protocole, contrairement à toutes les autres expériences, le transfert des caractéristiques de l'ILSVRC n'était pas implémenté.

Enfin, concernant la déformation de données en ligne, nous avons appliqué aux données réelles le *pipeline* de déformation décrit section 5.3.1 à chaque lot d'images présenté au réseau au cours de l'entraînement. Puisque ce *pipeline* était constitué d'une suite de six déformations dont chacune était appliquée avec une probabilité de 0,5, il y avait moins de 2% de chances qu'une image donnée subisse les mêmes déformations d'une époque à l'autre. De plus, ces déformations pouvaient varier en intensité selon la valeur de leur paramètre (par exemple, l'angle de la rotation), tiré aléatoirement pour chaque lot. Comme pour le rassemblement, la déformation en ligne était appliquée uniquement aux blocs d'entraînement des jeux étudiés.

5.4.2 L'impact des différents simulateurs

Les résultats des apprentissages sont réunis dans le tableau 5.4. Nous y présentons en sus les résultats d'apprentissage sans aucune donnée simulée, avec et sans transfert de l'ILSVRC. Nous avons considéré le résultat obtenu sans données simulées mais avec un transfert de l'ILSVRC comme la performance de référence. Nous commentons à présent l'impact des données simulées en fonction du simulateur employé.

Données simulées utilisées		Performance	
Aucune (sans transfert de l'ILSVRC)		0,424 ± 0,013	
Aucune (avec transfert de l'ILSVRC)		0,472 ± 0,009	
	Rassemblement	Pré-entraînement	En ligne
Déformation de données	0,559 ± 0,007	0,519 ± 0,006	0,574 ± 0,010
GAN	0,496 ± 0,011	0,494 ± 0,010	-
Modèle de canopée	0,509 ± 0,005	**0,602 ± 0,007**	-

TABLEAU 5.4 – Résultats des expériences concernant l'impact des données simulées sur l'entraînement sur $D^{0,2}_{\text{tuilé}}$.

Déformation de données

La déformation de données a permis une augmentation de performance substantielle par rapport à la performance de référence. En particulier, la configuration en ligne a permis une bond d'environ 10% de cette valeur. Ces résultats étaient cohérents avec l'omniprésence des déformations bien choisies dans les *pipelines* d'apprentissage modernes.

GAN

Contrairement aux autres catégories de simulateurs implémentées dans ce chapitre, les améliorations des performances permises par l'utilisation de données simulées par GAN étaient relativement faibles : environ 2% d'augmentation quelle que soit la manière dont ces données étaient

intégrées. Nous avons supposé que la faiblesse de ce résultat pouvait s'expliquer par les difficultés que nous avions rencontrées pour obtenir des images réalistes via un GAN.

Il était certes vrai que ces images, dont nous pouvons voir des exemples dans les figures 5.11 (b) et (d), avaient des qualités. Premièrement, le GAN parvenait à générer deux canaux qui étaient clairement identifiables aux canaux « IR » et « annotation » qui lui étaient fournis. En nous penchant sur le canal dédié aux annotations dans les images générées (sous-figure (d)), nous pouvons constater que ces images avaient une distribution de niveaux de gris proche d'une distribution binaire {0, 255} comme il était attendu, et que les structures représentées semblaient similaires à celles contenues dans les annotations réelles (sous-figure (c)). Concernant le canal correspondant aux images IR (sous-figure (b)), le GAN parvenait bien à générer des images dont la luminosité et les contrastes étaient similaires aux images IR réelles (sous-figure (a)), et nous y devinions même des formes plus sombres pouvant correspondre aux lésions ou au sol. Enfin, les canaux générés avaient une certaine cohérence entre eux puisque certaines structures dans les annotations générées semblent correspondre aux zones des « lésions » dans les images IR générées.

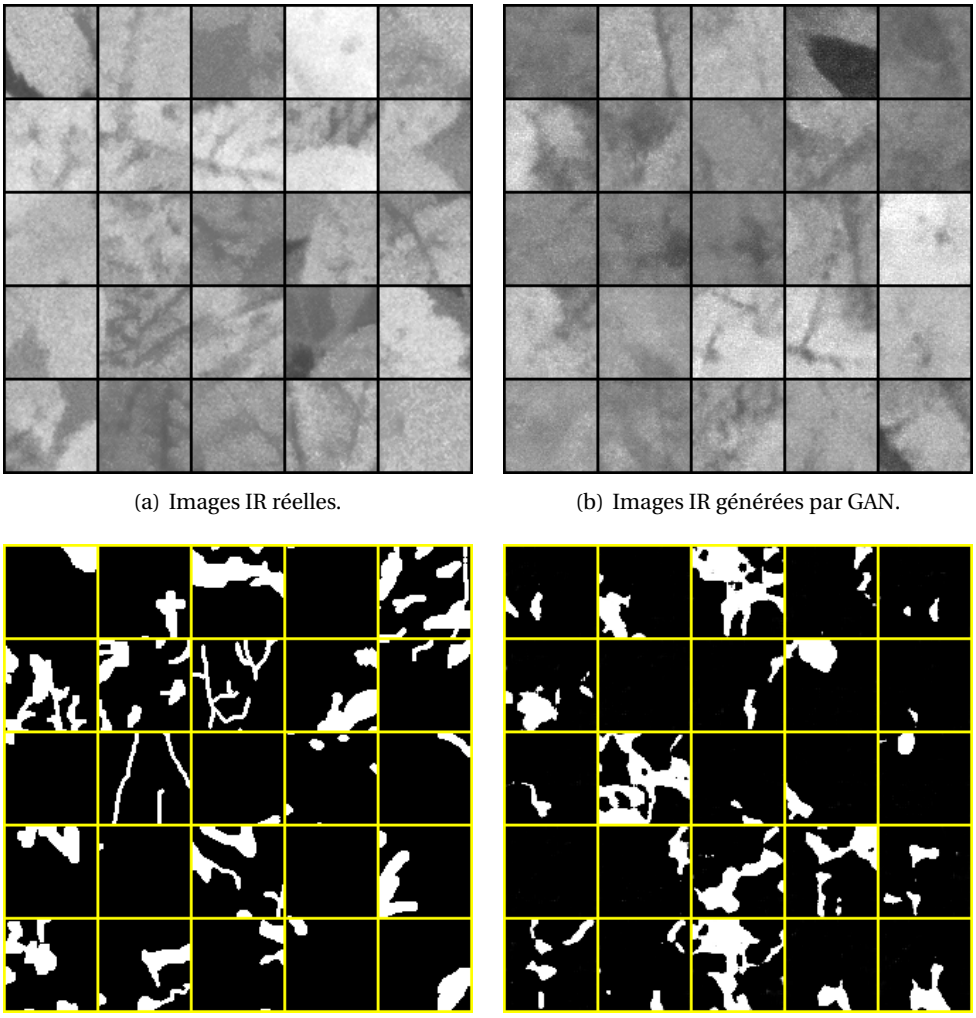
Cependant, l'information plus haut-niveau des images IR générées était médiocre. Les bords séparant les objets étaient flous, la texture des différentes zones y était approximative, les structures plus fines telles que les nervures ou bien des formes de lésions plus subtiles y étaient absentes, et de légers artefacts en « damier » [Odena et al., 2016] affectaient toute l'image. Ces images n'auraient pas passé le « test de Turing visuel » [Turing, 1950], c'est-à-dire qu'un humain ne les aurait pas confondues pas avec des images réelles. Même si la qualité esthétique de l'image pour un cerveau humain n'est pas strictement corrélée à sa pertinence du point de vue de l'entraînement d'un réseau de neurones (cf. note 18), nous avons fait l'hypothèse qu'elle en constituait un indicateur robuste.

Afin de disséquer les raisons de l'échec relatif du GAN, nous avons mené une expérience supplémentaire qui consistait à entraîner un GAN sur une variation de $D^{0,2}_{\text{tuilé}}$ où les annotations associées aux images IR avaient été permutées aléatoirement entre elles. En d'autres termes, les annotations n'étaient plus associées à l'image à partir de laquelle elles avaient été créées. Des exemples d'images générées par le GAN sur ce jeu de données sont présentés dans la figure 5.19. Nous constatons visuellement que dans cette configuration, en plus d'obtenir des canaux « annotation » de bonne qualité, les canaux « IR » générées avaient une qualité visuelle beaucoup plus satisfaisante. Ainsi, il était tout à fait possible pour le GAN de générer chacun des canaux séparément de façon satisfaisante. La pierre d'achoppement semblait être la création d'annotations *corrélées* aux images IR. Il semblait que pour réaliser des images réalistes, le GAN devait dans un premier temps réaliser une extraction de caractéristiques proche de celle demandée à un réseau de segmentation *bona fide*. Nous avons fait l'hypothèse que le cumul de cette tâche avec celle de la génération de données rendait la tâche trop difficile pour un GAN, ou *a minima* pour un GAN avec l'architecture utilisée dans cette étude.

Les difficultés de convergence du GAN employé pouvaient aussi s'expliquer par le faible nombre de données sur lequel celui-ci était entraîné. Cette dépendance à un grand nombre de données limitait l'intérêt de ce type d'architecture dans un cadre d'augmentation de données, où dans de nombreux cas le nombre de données initial était justement particulièrement faible.

Modèle de canopée

Le modèle de canopée a permis une augmentation de 12% de la performance dans le cas où les données étaient utilisées comme pré-entraînement du réseau. Cette augmentation démontrait que les images créées par le modèle étaient au moins partiellement réalistes du point de vue du réseau. Cependant, le gain permis par le rassemblement des images avec les données réelles était moins important, autour de 3%. Cette différence illustrait l'importance d'étudier les différentes façons



(a) Images IR réelles.

(b) Images IR générées par GAN.

(c) Annotations réelles mais décorrélées des images IR.

(d) Annotations générées par GAN.

FIGURE 5.19 – Exemples des images générées par un GAN entraîné sur des images de tavelure associées à des annotations qui leur étaient décorrélées. La présentation des données est identique à celle de la figure 5.11.

dont les données simulées pouvaient être intégrées à l'entraînement. Dans notre cas, les images issues du modèle de canopée semblaient être pertinentes pour mettre l'entraînement « sur de bons rails », comme le permettait le transfert de l'ILSVRC, mais étaient cependant trop différentes des données réelles pour que leur exploitation conjointe avec ces dernières soit très efficace.

Afin d'étudier plus en profondeur l'apport du simulateur, nous avons mené une étude par ablation. Le but de cette étude était d'évaluer l'apport de chacune des étapes du simulateur « modèle » au gain de performance observé. Pour ce faire, nous avons implémenté plusieurs versions du simulateur en omettant dans chacune d'entre elles d'exécuter une étape spécifique, tout en conservant les autres étapes inchangées. La liste des étapes que nous avons ainsi évaluées est présentée dans le tableau 5.5.

Pour chacun des simulateurs ainsi « amputés », nous avons généré un jeu de 3000 tuiles. Nous nous sommes servis de ces jeux pour pré-entraîner le réseau en lieu et place des images du jeu du modèle « complet », puis avons mené un entraînement sur $D^{0,2}_{\text{tuilé}}$. Les résultats sont présentés figure 5.20. La performance obtenue sans ablation (qui correspond à celle du modèle complet, rapportée dans le tableau 5.4) est indiquée à gauche du graphique. Plus la performance d'un apprentissage mené grâce à un simulateur « amputé » était basse par rapport à cette performance de

Étape omise	Changement implémenté pour omettre cette étape du simulateur
Genre	Les feuilles employées étaient remplacées par des feuilles de mûrier (*Broussonettia papyrifera*).
Structure	Les feuilles étaient placées en trois niveaux mais pas organisées en plants.
Luminosité	La luminosité globale des feuilles n'était pas modifiée en fonction de leur niveau sur la tige.
Taille	Les feuilles n'étaient pas redimensionnées en fonction de leur niveau sur la tige.
Courbure	Les feuilles ne subissaient pas de transformation de perspective pour simuler une pliure.
Tavelure (moments)	Le motif de tavelure était généré en fixant μ_{tavelure} et σ_{tavelure} de façon aléatoire pour chaque lésion.
Tavelure (texture)	Le motif de tavelure était généré sans modifier l'autocorrélation du bruit gaussien.
Vignettage	L'effet de vignettage n'était pas simulé.

TABLEAU 5.5 – Étapes étudiées lors de l'étude par ablation du modèle de canopée.

référence, plus nous pouvions considérer que l'étape associée était opportune.

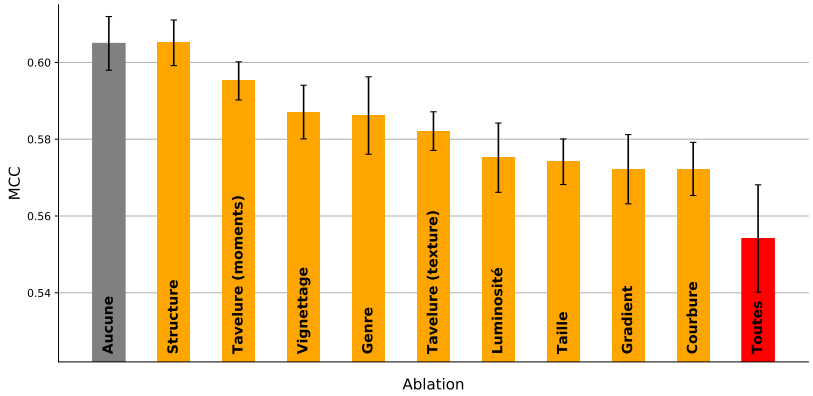


FIGURE 5.20 – Résultats de l'étude par ablation menée sur le modèle de canopée. Chaque barre correspond à la performance d'apprentissage sur $D^{0,2}_{\text{tuilé}}$ pré-entraîné sur des données issues du modèle de canopée dont une des étapes a été omise. L'étape omise en question est indiquée sur la barre. Nous présentons aussi la performance sans ablation (barre de gauche) ainsi que celle obtenue avec un simulateur qui cumule toutes les ablations (barre de droite).

Les résultats de cette étude montraient que l'effet « gradient » des lésions et la pliure des feuilles avaient l'impact le plus significatif parmi les étapes du modèle (+ 3%). À l'inverse, placer les feuilles dans une structure en plants semblait inutile. Nous nous sommes gardés cependant de généraliser excessivement ces résultats : ils traduisaient la pertinence des simulations telles que nous les avons implémentées, mais ne permettaient pas de valider ou d'infirmer l'utilité de tel ou tel trait de réalisme en général, même dans le cadre restreint de notre application. Par exemple, l'effet « gradient » des lésions de tavelure avait peut-être eu un impact positif sur la segmentation pour les raisons qui ont motivé son intégration au simulateur, c'est-à-dire un réalisme accru dû à la modélisation de la dispersion « gaussienne » de la concentration en agent infectieux. Mais il était aussi possible que le gain de performance ait été le résultat d'effets imprévus tels que la réduction de la taille effective des lésions, qui aurait compensé des tailles trop importantes de taches de tavelure dans notre modèle, ou bien la réduction du contraste des lésions, qui aurait mieux représenté certaines zones moins contrastées à cause des conditions d'illumination.

Nous avons mené une expérience supplémentaire qui a son intérêt : l'ablation de toutes les étapes listées dans le tableau 5.5 simultanément (barre rouge à droite dans la figure 5.20). Un exemple d'une image générée ainsi est présentée figure 5.21. Il était intéressant de noter que même dans ce cas, le pré-entraînement sur les images du modèle de canopée permettait une augmentation de performance de 8% par rapport à la performance de référence. Nous avons supposé, de la même manière qu'un pré-entraînement sur l'ILSVRC était utile pour toute tâche de vision [Sharif Razavian et al., 2014], un pré-entraînement sur une tâche des sciences végétales pouvait être utile pour d'autres tâches du domaine, et ce même si les informations de luminosité, contraste, structure, etc.,

de la scène étaient drastiquement différentes.

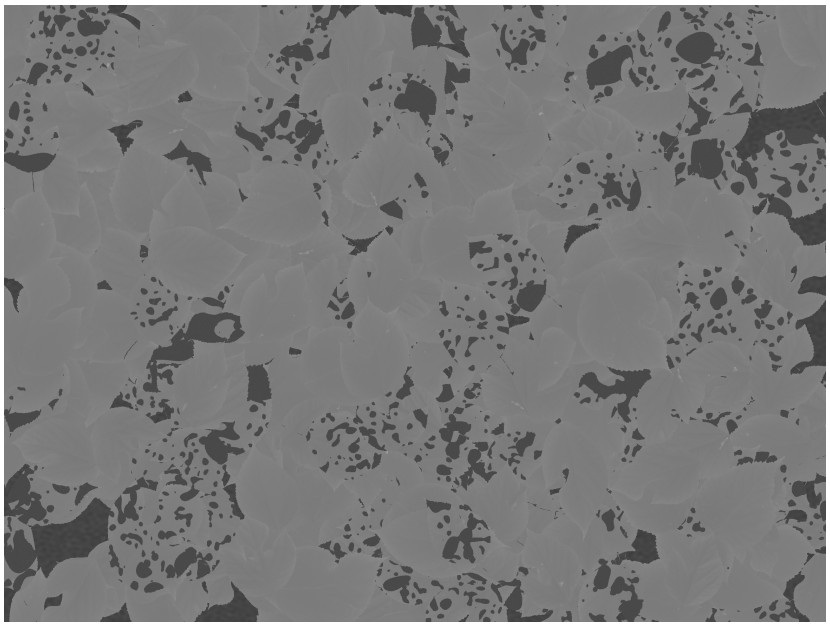


FIGURE 5.21 – Une image créée avec le modèle de canopée amputé de toutes les étapes listées dans le tableau 5.5.

Combinaison de données simulées

Les différentes façons d'exploiter les données simulées que nous avons étudiées agissaient à différentes étapes de l'entraînement : au cours de l'initialisation des poids, de la constitution du jeu ou bien des itérations d'apprentissage en elles-mêmes. Il était ainsi très aisé d'utiliser de façon conjointe certaines d'entre elles. Nous avons mis au point la procédure d'entraînement « optimale » selon les résultats du tableau 5.4 : nous avons pré-entraîné le réseau avec les données du modèle de canopée tout en implémentant une déformation de données en ligne. La performance de segmentation du réseau sur $D^{0,2}_{\text{tuilé}}$ était alors de 0,643 ± 0,005, soit une augmentation de 17% par rapport à la performance de référence. Ce résultat montrait que les effets bénéfiques de ces simulations pouvaient être ajoutés voire même mener à une interaction positive.

5.4.3 Les données simulées permettent d'aller au-delà des données réelles...

Nous présentons à présent les performances d'entraînement sur l'ensemble des jeux $D^{p}_{\text{tuilé}}$ dans ces conditions de simulation « optimales », en les comparant à celles obtenues sur la performance de référence (figure 5.22).

Nous pouvons constater qu'inclure des données simulées permettait un gain de performance sur l'ensemble des jeux $D^{p}_{\text{tuilé}}$. Ce gain de performance était d'autant plus important que les jeux étaient réduits (plus de 24% sur $D^{0,1}_{\text{tuilé}}$ par exemple), mais il était encore important même pour une quantité de données annotées où la performance était « saturée » (presque 10% sur $D^{1}_{\text{tuilé}}$ notamment). Nous avons trouvé ce résultat surprenant car nous nous attendions à ce que l'apport des données simulées tende rapidement vers zéro à mesure que la quantité d'images réelles utilisée augmentait. Il semblait que pour notre application, l'intégration de données simulées permettait non seulement de pallier le manque de vraies données mais amenait une véritable plus-value qui ne semblait pas être atteignable par l'ajout de données réelles, et ce même en quantité importante. Nous avons supposé que le pré-entraînement avait un effet particulièrement bénéfique à ce titre, en permettant à l'optimiseur de poids de démarrer la descente de gradient dans une configuration favorable et par la suite d'atteindre des minima de coût plus bas que ceux auxquels une initialisation

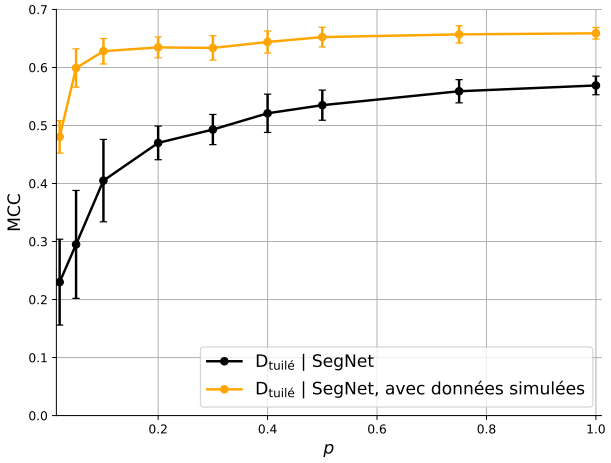


FIGURE 5.22 – Performance de SegNet sur les jeux $D^{p}_{tuilé}$, avec et sans l'utilisation de données simulées.

aléatoire des poids aurait permis d'accéder.

D'un point de vue « métier », la figure 5.22 montre qu'il était possible d'utiliser les données simulées pour réduire la charge d'annotation tout en conservant les mêmes performances d'apprentissage. Ainsi, dans notre cas d'étude, nous avons obtenu en annotant 5% du bloc d'entraînement de $D^{1}_{tuilé}$ et en intégrant des données simulées une performance d'apprentissage supérieure à celle permise par l'annotation du bloc d'entraînement entier de $D^{1}_{tuilé}$ sans données simulées [20].

5.4.4 ... mais ne sont pas si « gratuites » que cela

Il nous paraît important de tempérer les apports de la simulation en précisant que le gain de performance obtenu était moins « gratuit » qu'il n'en avait l'air au premier abord. En effet, l'utilisation de ces simulateurs entraînait plusieurs types de coûts. Il y avait d'abord le coût d'entraînement, c'est-à-dire l'allongement du temps d'entraînement dû à l'intégration des données supplémentaires. Rassembler des données simulées et des données réelles entraînait un délai proportionnel à la quantité de données simulées ajoutées. Quant à l'intégration de données simulées via un pré-entraînement, il était nécessaire de mener un entraînement préliminaire jusqu'à convergence sur les données simulées, mais les poids obtenus pouvaient ensuite être utilisés pour plusieurs entraînements sur données réelles, attendu que l'architecture du réseau restait identique.

Il fallait aussi considérer le coût de génération des données. Pour créer des données par GAN, l'implémentation du réseau en lui-même était rapide, mais la création de nouvelles données n'était possible qu'à l'issue d'un long et délicat entraînement du GAN. Un simulateur « modèle », au contraire, permettait de créer des images sans effort ni délai, mais c'était alors la création et l'implémentation du modèle en lui-même qui pouvait être très chronophage. Nous soulignons que dans ce dernier cas, le temps dont il était question n'était pas le temps d'une machine mais bien de celui d'un ·e développeur ·euse [21].

20. Cette comparaison n'était pas tout à fait juste dans la mesure où le modèle de canopée requérait pour générer un motif de tavelure synthétique les valeurs des caractéristiques calculées sur le jeu $D^{1}_{tuilé}$. L'entièreté du bloc d'entraînement de $D_{tuilé}$ était donc partiellement utilisé pour obtenir les résultats d'apprentissage sur $D^{0,05}_{tuilé}$. Nous avons cependant considéré que cette « fuite de données » était minime car les caractéristiques calculées sur les jeux réduits auraient sans doute pris des valeurs proches de celles calculées sur le jeu complet.

21. ... ce qui est rarement une alternative enviable, même d'un point de vue financier, comme le souligne l'adage « Le temps des développeurs est plus important que celui des machines. » (Source : https://blog.codinghorror.com/hardware-is-cheap-programmers-are-expensive/).

Ainsi, le gain d'annotation permis par les données simulées n'était pas aussi gratuit que les résultats de la figure 5.22 pouvaient le laisser penser. Une manière de mitiger ces coûts consistait à rendre les simulateurs les plus universels possibles, de manière à pouvoir utiliser les données simulées pour une large gamme d'applications. Nous n'avons pas au cours de ce travail tenté d'exporter ou d'adapter les simulateurs développés à d'autres applications de sciences végétales. Cependant, des résultats semblaient indiquer une certaine universalité des données générées, en particulier la pertinence de pré-entraînements du simulateur « modèle » réalisés sur des feuilles « simples », sans spécialisation excessive sur les images de notre cas d'étude (cf. étude par ablation de la section 5.4.2).

5.5 Conclusion

Nous nous sommes dans ce chapitre intéressés à l'apport possible des données simulées pour réduire la charge d'annotation nécessaire pour mener à bien une tâche de segmentation. Nous avons pour cela développé plusieurs nouveaux simulateurs de données dédiés à la tavelure du pommier, qui s'inscrivaient dans le courant des simulateurs développés ces dernières années dans le domaine des sciences végétales. En particulier, nous avons mis au point un simulateur « modèle » de canopée à partir de sources externes, selon un *pipeline* plus simple que d'autres simulateurs de la littérature. Par ailleurs, nous avons adapté un GAN afin de générer des images annotées utiles pour la segmentation.

De plus, contrairement à de nombreuses études travaillant sur les données simulées, nous nous sommes penchés sur les différentes façons dont ces données simulées pouvaient être intégrées dans le *pipeline* d'apprentissage. La comparaison que nous avons menée entre les différentes méthodes employées dans la littérature a souligné que ce choix d'intégration avait un impact fort sur la performance de segmentation. Les meilleurs résultats que nous avons obtenus provenaient d'une combinaison de différentes catégories de données simulées. Nous avons montré avec ces expériences qu'il était possible de réduire considérablement la charge d'annotation de la tâche à performance égale et même d'améliorer cette performance au-delà de l'apport de nouvelles données réelles. En particulier, nous sommes parvenus, grâce à l'utilisation de données simulées, à réduire de plus de 95% le nombre de données réelles à annoter à performance égale.

En outre, nous avons proposé dans ce chapitre un nouveau jeu de données annoté pour la segmentation de lésions de tavelure. Ce jeu était novateur selon plusieurs aspects par rapport à la plupart de ceux disponibles dans le domaine des sciences végétales. Premièrement, il s'agissait d'un jeu en conditions réelles, avec des caractéristiques qui rendaient une tâche de vision par ordinateur difficile. Il était associé à une tâche de segmentation, et pouvait ainsi servir à entraîner des algorithmes destinés à évaluer la sévérité des infections, ce qui était considérablement moins répandu que les jeux de classification de maladies. Enfin, il s'agissait d'un jeu en lumière IR, contrairement à la grande majorité des jeux disponibles qui sont constitués d'images RVB. Pour toutes ces raisons, les résultats positifs de segmentation que nous avons présenté dans la section 5.1.4 étaient significatifs car ils montraient pour la première fois la capacité d'un réseau de neurones à être performant sur cette tâche dans ces conditions difficiles. Selon nous, ce jeu présentait un intérêt pour la communauté des chercheur ·euses en sciences végétales et nous l'avons donc publié pour une utilisation libre (cf. section Valorisations).

Pour porter un regard critique sur le travail de ce chapitre, nous estimons que le modèle de canopée comportait des étapes que nous aurions pu enrichir. Similairement au simulateur de cubes hyperspectraux présenté au chapitre 3, l'acquisition en lumière IR d'une feuille était simulée par une simple conversion en niveaux de gris d'une feuille acquise par un capteur RVB. Par ailleurs, nous avons fixé un certain nombre de paramètres de ce simulateur en nous basant sur des simples observations visuelles des données réelles, mais sans argument biologique pour justifier leur valeur.

Nous affirmons à nouveau notre volonté présentée lors de la discussion à propos du simulateur de cubes hyperspectraux (section 3.4) de proposer un modèle relativement simple et facilement paramétrisable. Les gains de performance que ce modèle a permis pour une application sur données réelles appuient la viabilité de cette simplicité. Nous soulignons en particulier que, si les lésions de tavelure du pommier ont une texture complexe sur les images réelles, la simulation d'une petite partie des caractéristiques de cette texture (figure 5.13) a suffi à améliorer les performances de segmentation des lésions.

De plus, si le jeu de données réelles et la tâche que nous lui avons associée étaient représentatifs des défis de vision dans un cadre industriel d'agriculture de précision, nous pensons qu'il serait bénéfique de créer un « véritable » bloc de test, constitué d'acquisitions réalisées dans un autre contexte (autre serre, horaire d'acquisition différent, etc.). Bien que nous ayons pris soin de rassembler dans $D_{original}$ des images représentant des zones séparées de la serre sans recouvrement, il y avait tout de même une ressemblance entre les données du bloc d'entraînement et celles du bloc de test qui amplifiait artificiellement la capacité mesurée du réseau à généraliser sur des données « véritablement » nouvelles.

Les travaux de ce chapitre ont permis de montrer que les lésions de tavelure pouvaient être au moins partiellement détectées par un algorithme d'apprentissage sur des images IR, et ce même en conditions difficiles. Les travaux des chapitres 3 et 4 avaient par ailleurs montré la viabilité du CTIS pour réussir cette détection — certes en conditions contrôlées. Les résultats positifs de ces deux études nous ont poussés à nous intéresser aux possibilités offertes par la fusion de l'information de ces différents capteurs.

Chapitre 6

Vers une fusion spatio-spectrale

Lorsque plusieurs sources d'information complémentaires sont disponibles pour répondre à une question, il survient naturellement l'idée de les exploiter conjointement, de la même manière que le cerveau humain combine plusieurs entrées sensorielles pour créer un sens du monde qui l'entoure [Baltrušaitis et al., 2018]. Dans le champ de l'apprentissage automatique, l'action de combiner plusieurs sources de données afin de répondre à une problématique commune est désignée sous le nom de fusion de données. Nous avons présenté au chapitre 4 un travail dans cette optique lorsque nous avons décrit l'architecture CTIS-Net qui tirait parti des deux ordres de diffraction des images CTIS pour mener à bien une classification des images entières. L'étude de la fusion de données constitue en vérité un pan entier du champ de l'apprentissage automatique, et, comme nous y faisons mention au chapitre 4, des travaux s'y rattachant ont porté sur des modalités très variées dans des champs divers. Nous présentons dans ce chapitre des premiers travaux concernant la fusion de différentes modalités d'imagerie RVB, IR et CTIS, dans un objectif de classification. Nous nous attardons en particulier sur les difficultés que nous avons rencontrées concernant la fusion d'images aux structures très dissemblables, un problème relativement peu étudié dans la communauté de vision par ordinateur. Nous présentons des pistes de recherche se rapportant à cette problématique.

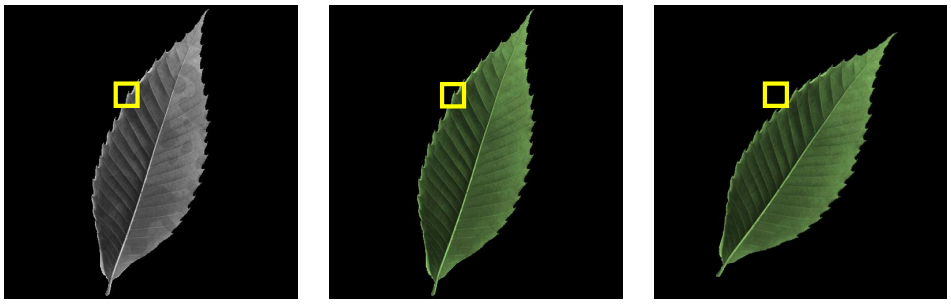
Sommaire

6.1 Un pan important de la vision par ordinateur 122
6.2 Un gain pour la fusion d'images alignées 123
6.3 Les difficultés de fusionner des imageries non standard 125
6.4 Conclusion 127

6.1 Un pan important de la vision par ordinateur

Dans le champ de la vision par ordinateur, les applications concernant la fusion de données sont nombreuses. Contrairement aux travaux présentés au chapitre 4, la fusion concerne dans la plupart des cas la combinaison d'informations provenant de deux capteurs distincts [Busemeyer et al., 2013]. Parmi les imageurs utilisés, on trouve quasiment systématiquement un imageur RVB en vertu de sa grande disponibilité dans le commerce. Les imageries le plus souvent employées en complément sont l'IR afin d'obtenir des informations thermiques [Ma et al., 2019], les cartes de profondeur pour des informations liées aux volumes de la scène [Kim et al., 2009] et dans le cas des vidéos, le flot optique afin d'intégrer des informations de mouvement [Simonyan and Zisserman, 2014a] [22]. Quant aux domaines d'application, la fusion de données est particulièrement étudiée dans le champ de la détection de silhouettes, notamment en conditions nocturnes [Hwang et al., 2015; Wagner et al., 2016], celui de la classification d'actions humaines à partir de vidéos [Karpathy et al., 2014; Simonyan and Zisserman, 2014a], celui de la segmentation urbaine à partir d'images aériennes [Liao et al., 2015; Yu et al., 2017] et celui de la conduite autonome [Kocić et al., 2018].

Concernant les méthodes employées, bien que le champ soit largement antérieur à celui de l'apprentissage profond, les réseaux de neurones sont depuis une décennie les algorithmes plébiscités pour réaliser l'opération de fusion d'images [Li et al., 2017a]. Ils sont utilisés en particulier lorsque les images proviennent de capteurs différents, mais sont malgré cela « alignées » entre elles. Nous entendons par ce terme qu'un pixel donné dans une modalité représente le même élément spatial de la scène que le pixel à la même position dans l'autre modalité (sous-figures (a) et (b) de la figure 6.1).



(a) (b) (c)

FIGURE 6.1 – Exemples d'images alignées (a et b) et désalignées (a et c). Nous avons encadré en jaune une zone située à la même position dans chacune des images. Source de l'image de feuille originale : [Kumar et al., 2012].

Les CNNs ont été appliqués à des problématiques de classification basées sur plusieurs images quelques années à peine après leurs premiers succès concernant les tâches de classification unimodales [Karpathy et al., 2014; Simonyan and Zisserman, 2014a]. Les réseaux employés adoptaient la structure « à deux branches » présentée au chapitre 4, où chaque branche traitait une des modalités avant que les cartes de caractéristiques ainsi créées ne soient concaténées puis traitées conjointement.

Un paramètre clé dans ce type d'architecture, que nous avons passé sous silence au chapitre 4, est la position de cette concaténation dans le réseau. En effet, son placement plus ou moins proche de l' « espace image » conditionne le niveau sémantique auquel la fusion des modalités s'effectue. Ce positionnement est une problématique amplement débattue dans les études traitant de la fusion, largement antérieure à l'utilisation de réseaux de neurones dans le domaine. Certaines études se concentrent même exclusivement sur ce choix [Gunes and Piccardi, 2005; Snoek et al.,

22. Pour être précis, le flot optique d'une vidéo ne provient pas d'un autre type d'imagerie mais résulte d'un calcul effectué sur la vidéo destiné à extraire le mouvement des objets entre les images qui la composent.

2005]. On qualifie la fusion de plus ou moins « tardive » en fonction de l'option choisie.

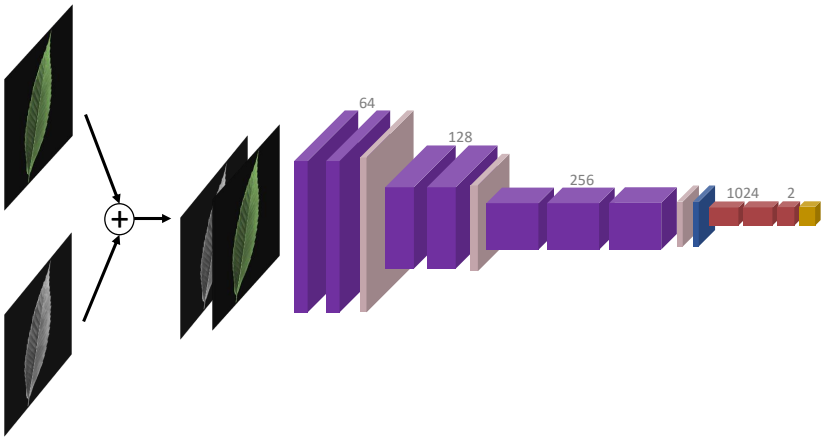
La fusion « précoce » consiste à concaténer les modalités entre elles avant de les traiter avec un réseau unique. Elle permet au réseau d'exploiter au maximum les interactions bénéfiques possibles entre les modalités [Paisitkriangkrai et al., 2015]. De plus, la simplicité architecturale permet des entraînements plus aisés. Cependant, cette fusion peut mener à un apprentissage qui se fait au détriment d'une des modalités qui aurait bénéficié à être traitée seule afin d'en extraire des caractéristiques pertinentes [Katsaggelos et al., 2015] (voir par exemple la différence des temps de convergence entre les deux modalités dans notre étude de CTIS-Net, illustrée en figure 4.15). La fusion « tardive », au contraire, consiste en un traitement individuel de chaque modalité avant de combiner les sorties de chaque branche. Cette combinaison peut être aussi simple qu'une moyenne des prédictions obtenues après la couche *softmax* [Simonyan and Zisserman, 2014a]. Un des avantages de cette fusion est que les caractéristiques issues de chaque branche sont dans un espace latent à structure similaire, ce qui permet une combinaison aisée des modalités. De plus, chaque modalité a pu être traitée de façon personnalisée et approfondie par le réseau [Katsaggelos et al., 2015]. Dans cette configuration en revanche, le réseau ne peut pas extraire les éventuelles interactions bénéfiques entre modalités qui peuvent émerger à des niveaux sémantiques moins élevés. Entre ces deux extrêmes, la flexibilité des architectures à deux branches permet de réaliser la fusion à n'importe quelle couche du réseau. On parle alors de fusion intermédiaire. Bien que le niveau de fusion optimal soit très dépendant de l'application et que certain ·es chercheur ·euses n'hésitent pas à l'inclure à la recherche par grille effectuée pour fixer les hyperparamètres architecturaux de leur réseau [Castro et al., 2020], un niveau de fusion régulièrement implémenté pour les CNNs se situe à la sortie des couches convolutives [Jing et al., 2017], comme il a été fait pour l'architecture de CTIS-Net.

6.2 Un gain pour la fusion d'images alignées

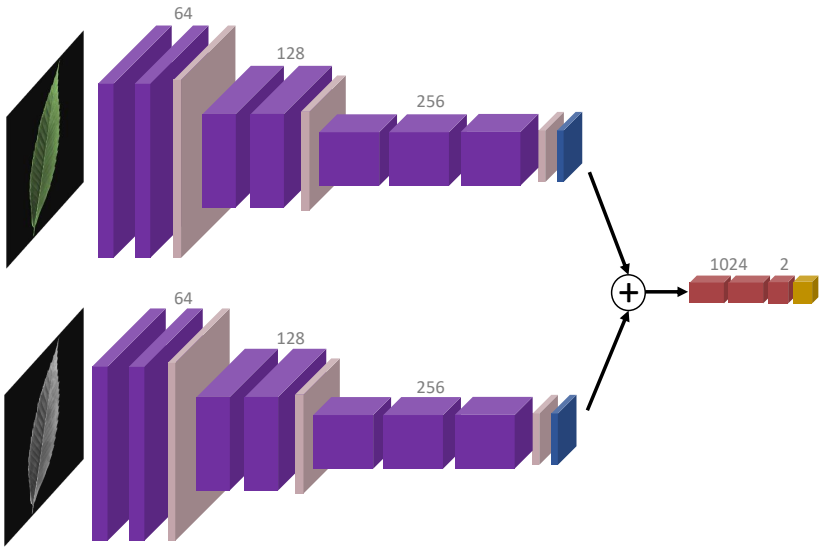
Pour étudier la possibilité de fusionner les images produites par la caméra Carbon Bee, nous avons implémenté trois réseaux de fusion notés $\text{VGGf}_{\text{précoce}}$, $\text{VGGf}_{\text{intermédiaire}}$ et $\text{VGGf}_{\text{tardif}}$. Il s'agissait de réseaux à deux branches basés sur VGGr (figure 4.2) qui ne différaient que par la position de concaténation des caractéristiques. Leur architecture est présentée dans la figure 6.2. Nous soulignons qu'en conséquence de leur structure à deux branches, ces architectures, en particulier $\text{VGG}_{\text{intermédiaire}}$ et $\text{VGG}_{\text{tardif}}$, comportaient un nombre de paramètres plus important que VGGr. Nous avons fait l'hypothèse que les éventuels gains de performance qu'ils permettraient proviendraient bien de l'exploitation conjointe des deux modalités, et non de la simple augmentation du nombre de paramètres, comme nous l'avions montré pour CTIS-Net (cf. tableau 4.2).

Nous nous sommes intéressés tout d'abord au cas d'images alignées. Un tel alignement était difficile à obtenir dans la réalité. Il était en effet nécessaire pour cela que le dispositif optique utilisé acquière les différentes modalités à travers le même objectif. Or, dans la grande majorité des cas, y compris celui de la caméra Carbon Bee, les montages optiques sont composés de plusieurs capteurs aux objectifs distincts. La scène n'est alors pas acquise d'un point de vue identique selon les capteurs, et les images résultantes ne sont pas exactement alignées. Un tel désalignement peut être délétère pour une analyse conjointe des modalités. En particulier, les CNNs sont moins efficaces en cas de décalage important car les noyaux de convolution ne s'appliquent alors pas sur les mêmes zones spatiales d'une modalité à l'autre. Dans ce cas, plus la concaténation envisagée est précoce, plus il est nécessaire d'aligner ces images en prétraitement [23], notamment par le biais d'algorithmes

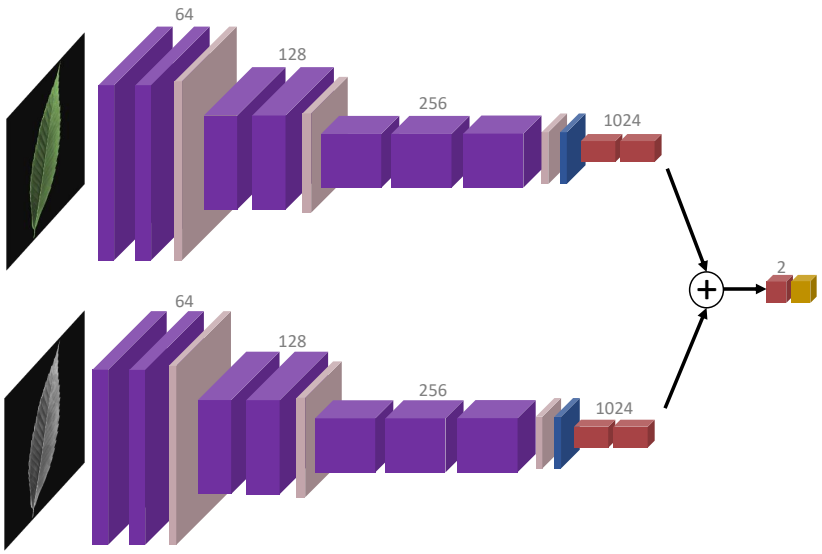
23. Il est à noter que les CNNs contenant de couches *max-pool* sont capables de « corriger » partiellement des désalignements entre des images. En effet, chacune de ces couches réduit les dimensions spatiales des cartes de caractéristiques. Il suffit par exemple d'une couche *max-pool* à coefficient de réduction de deux pour éliminer après cette couche un décalage d'un pixel entre deux modalités. À l'issue d'un grand nombre de couches *max-pool*, les cartes de caractéristiques produites peuvent être alignées même en cas de désalignement important entre les images fournies en entrée du réseau.



(a) $\text{VGGf}_{\text{précoce}}$



(b) $\text{VGGf}_{\text{intermédiaire}}$



(c) $\text{VGGf}_{\text{tardif}}$

FIGURE 6.2 – Les architectures de fusion utilisées dans ce chapitre.

de recalage. Nous présentons en annexe B des travaux que nous avons réalisés dans ce sens pour des acquisitions réelles de la caméra Carbon Bee.

Nous avons étudié la fusion des jeux simulés au cours du chapitre 3 (figure 3.23), afin de réaliser une classification binaire « feuille saine / feuille tavelée ». Les images de certains de ces jeux avaient l'avantage d'être parfaitement alignées entre elles. C'était le cas notamment des jeux D_{RVB} et D_{IR}. Nous nous sommes concentrés sur les jeux correspondant à une valeur de sévérité de 0,3 car les performances obtenues avec VGGr sur $D^{0,3}_{RVB}$ et $D^{0,3}_{IR}$ n'étaient ni parfaites (MCC de 1) ni « nulles » (MCC en dessous de 0,06) (figure 4.21). Étudier la fusion de ces deux jeux permettait donc d'identifier si cette opération était bénéfique, sans effet ou délétère. Nous avons rassemblé les jeux $D^{0,3}_{RVB}$ et $D^{0,3}_{IR}$ dans un nouveau jeu noté $D^{0,3}_{RVB+IR}$. Nous avons mené des entraînements sur ce jeu avec les différents réseaux VGGf, en suivant le protocole d'entraînement détaillé aux sections 4.1.2 et 4.4.1. Les résultats sont présentés dans le tableau 6.1. Ils indiquaient que la fusion précoce permettait de tirer parti des interactions bénéfiques entre l'image RVB et l'image IR. Cette interaction bénéfique n'était pas sans rappeler le succès des indices végétaux mentionnés à la section 1.1.3. Les NDVI sont en effet basés sur la combinaison arithmétique de plusieurs longueurs d'onde provenant du visible et de l'IR. Il est possible que l'exploitation conjointe de longueurs d'onde issues de ces deux domaines ait été bénéfique pour les réseaux d'une façon analogue.

Jeu de données	Réseau	Performance
$D^{0,3}_{RVB}$	VGGr	$0{,}91_{\pm 0{,}014}$
$D^{0,3}_{IR}$		$0{,}98_{\pm 0{,}005}$
$D^{0,3}_{RVB+IR}$	$VGGf_{précoce}$	$\mathbf{1{,}00}_{\pm 0{,}000}$
	$VGGf_{intermédiaire}$	$0{,}96_{\pm 0{,}025}$
	$VGGf_{tardif}$	$0{,}96_{\pm 0{,}020}$

TABLEAU 6.1 – Résultats des expériences de fusion des modalités RVB et IR. Dans tous les tableaux de ce chapitre, le meilleur résultat parmi les configurations testées est indiqué en gras.

Concaténer les modalités à un niveau plus tardif entraînait une performance légèrement inférieure par rapport à l'utilisation de la modalité optimale qu'était l'IR. Nous pouvons noter par ailleurs que l'écart-type de ces performances-ci était supérieur à celui observé dans le cas des modalités individuelles. Nous avons en effet observé deux cas différents selon les répétitions : soit les performances obtenues avec $VGGf_{intermédiaire}$ et $VGGf_{tardif}$ étaient comparables à celle obtenue avec VGGr sur le jeu $D^{0,3}_{IR}$, soit celles-ci étaient légèrement inférieures. Il semblerait donc que non seulement l'interaction bénéfique n'était pas possible à un niveau sémantique plus élevé, mais, de plus, que l'ajout d'une modalité sous-optimale pouvait perturber la convergence du réseau. En d'autres termes, le réseau pouvait ne pas être tout à fait capable « d'ignorer » une modalité s'il ne parvenait pas à intégrer bénéfiquement l'information de celle-ci.

6.3 Les difficultés de fusionner des imageries non standard

L'architecture neuronale à deux branches semblait être la structure naturelle permettant de fusionner deux images alignées. Elle permettait dans notre cas d'étude de bénéficier d'interactions positives qui émergeaient dans le cadre d'une fusion précoce. Cependant, il existe aussi de nombreux cas où les images à fusionner ne sont pas alignées car elles sont de natures différentes. Cette problématique d'incompatibilité entre modalités est très répandue dans le domaine de l'imagerie spectrale. En effet, il est courant pour les imageurs spectraux de produire des cubes à la résolution spatiale faible par rapport à celle permise par des imageurs non-spectraux [Shaw and Burke, 2003]

— c'est le cas pour le CTIS. Ces imageurs spectraux sont régulièrement associés à des capteurs à la résolution spatiale bien plus importante, notamment des capteurs RVB ou panchromatiques (figure 2.2) [Loncan et al., 2015], dans le but de fusionner les informations spectrales et spatiales. Cette tâche est particulièrement étudiée dans le domaine de la télédétection spatiale, c'est-à-dire l'acquisition d'images de la surface terrestre via des imageurs montés sur des satellites en orbite [Ghassemian, 2016]. Les chercheur ·euses du domaine s'intéressent en particulier à une opération nommée l'affinage panchromatique (*pansharpening* en anglais) [Loncan et al., 2015; Ferraris et al., 2017] qui consiste en la création à partir de ces deux modalités d'un cube spectral à haute résolution spatiale [24]. Il est en effet courant que la faible résolution spatiale des images spectrales compliquent l'interprétation de celles-ci car l'information spectrale de plusieurs éléments peut être confondue dans un pixel donné [Louargant et al., 2017]. De nombreuses méthodes ont été proposées à cette fin [Wei et al., 2015], basées en particulier sur la transformation des deux images dans un espace de représentation différent afin de les combiner plus aisément [Ben Hamza et al., 2005] ou bien par le biais de réseaux de neurones [Masi et al., 2016].

Une fusion suivant les objectifs du champ de l'affinage panchromatique était possible dans notre cas en combinant des images RVB, à forte résolution spatiale mais faible résolution spectrale, et de l'ordre 0 des images CTIS, à forte résolution spectrale mais faible résolution spatiale. Nous avons appliqué pour cela un post-traitement aux images de $D^{0,3}_{CTIS}$ qui consistait à isoler l'ordre 0 des images, puis à redimensionner cette imagette à la taille de l'image RVB (figure 6.3). Nous avons ainsi créé à partir du jeu $D^{0,3}_{CTIS}$ un nouveau jeu noté $D^{0,3}_{CTIS0r}$, puis le jeu $D^{0,3}_{RVB\,+\,CTIS0r}$. Les résultats d'entraînement sur ce dernier jeu sont présentés dans le tableau 6.2. Dans cette situation encore, la fusion précoce permettait un gain de performance par rapport à l'utilisation de modalités individuelles.

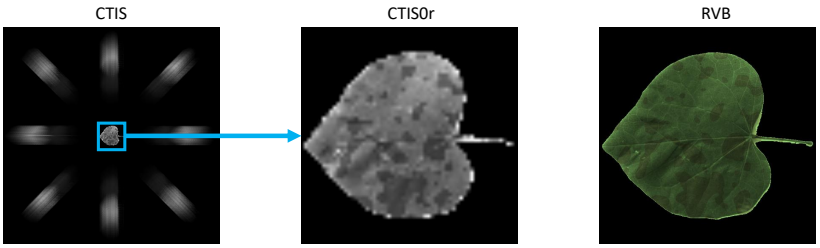


FIGURE 6.3 – Post-traitement proposé pour les images CTIS afin d'en créer une image alignée avec l'image RVB associée.

Jeu de données	Réseau	Performance
$D^{0,3}_{RVB}$	VGGr	$0{,}91_{\pm 0{,}014}$
$D^{0,3}_{CTIS0r}$		$0{,}49_{\pm 0{,}030}$
$D^{0,3}_{RVB+CTIS0r}$	$VGGf_{précoce}$	$\mathbf{0{,}94}_{\pm 0{,}017}$
	$VGGf_{intermédiaire}$	$0{,}87_{\pm 0{,}027}$
	$VGGf_{tardif}$	$0{,}90_{\pm 0{,}023}$

TABLEAU 6.2 – Résultats des expériences de fusion des modalités RVB et CTIS0r.

Cependant, cette expérience ne tirait pas parti de toute l'information spectrale portée par les images CTIS, c'est-à-dire les projections de l'ordre 1. La différence entre les images CTIS complètes

24. À proprement parler, il ne s'agit donc pas exactement d'une fusion de données dans le sens où nous l'avons définie dans ce chapitre. En effet, le but de l'affinage panchromatique n'est pas de tirer une information unique (comme une classe) des deux images mais de combiner celles-ci afin créer une image « améliorée ». Il s'agit donc d'un autre sous-domaine du champ de la multimodalité que l'on appelle l'*image enhancing* [Loncan et al., 2015].

et les images RVB et IR allait plus loin qu'un nombre de canaux différents ou des résolutions spatiales dissemblables. Les images CTIS étaient des images multiplexées où la distance entre deux pixels traduit tantôt une distance spatiale, tantôt une distance spectrale en fonction de la position des dits pixels dans l'image. Les performances sur le jeu $D^{0,3}_{RVB+CTIS}$, présentées dans le tableau 6.3, montraient que les réseaux à deux branches étaient sous-optimaux pour la fusion de deux images si structurellement différentes, et ce même en fusion tardive. Nous pouvons noter aussi que la fusion précoce ne permettait qu'une performance bien inférieure à celle permise par la modalité optimale. Cette performance faible était due à certaines répétitions où le réseau se concentrait uniquement sur la modalité CTIS malgré son caractère sous-optimal pour cette classification.

Jeu de données	Réseau	Performance
$D^{0,3}_{RVB}$	VGGr	**0,91**$\pm 0,014$
$D^{0,3}_{CTIS}$		$0,52 \pm 0,031$
$D^{0,3}_{RVB+CTIS}$	$VGGf_{précoce}$	$0,67 \pm 0,121$
	$VGGf_{intermédiaire}$	**0,91**$\pm 0,016$
	$VGGf_{tardif}$	$0,89 \pm 0,012$

TABLEAU 6.3 – Résultats des expériences de fusion des modalités RVB et CTIS.

Ainsi, la question de la fusion d'une image CTIS avec une image plus standard était à rapprocher d'études traitant de la fusion d'informations spatiales et spectrales qui sont véritablement structurellement différentes. On trouve ce cas en particulier dans le domaine de la sécurité alimentaire [Borràs et al., 2015] où de nombreux capteurs sont utilisés en parallèle pour juger de la qualité d'un aliment. Il est courant dans ce domaine qu'une image RVB soit associée à une information spectrale qui n'est pas représentée sous la forme d'une image, mais plutôt comme un vecteur unidimensionnel car provenant d'un spectromètre mesurant le spectre de l'aliment en une seule localisation. Les études de ce domaine suivent souvent des protocoles hybrides entre apprentissage profond et caractéristiques prédéfinies [Huang et al., 2011; Simonyan and Zisserman, 2014a; Sanaeifar et al., 2018]. Des caractéristiques telles que les moments et des informations de texture sont extraites des images RVB avant d'être concaténées avec les informations spectrales conservées telles quelles ou bien soumises à des algorithmes de réduction de dimension. Des réseaux de neurones sont ensuite employés pour proposer une prédiction à partir de ces caractéristiques.

6.4 Conclusion

Nous avons présenté dans ce chapitre des résultats d'apprentissage se basant sur la fusion de données. La fusion d'images parfaitement alignées, un cas rendu possible par l'utilisation de données simulées, a permis un gain de performance via l'exploitation d'architectures de fusion dites « précoces », c'est-à-dire où les deux modalités étaient exploitées conjointement dès le début de l'analyse. Ces gains étaient cohérents avec ceux observés dans la littérature, en particulier celle relative aux indices végétaux. Cependant, dans le cas d'images non alignées telles que les images RVB et CTIS, nous n'avons pas obtenu de gain de performance malgré l'implémentation de plusieurs types d'architectures de fusion.

Selon nous, la littérature concernant l'apprentissage automatique dans la sécurité alimentaire est riche d'enseignements pour la fusion d'informations spatiales et spectrales structurellement différentes. Il y est présenté des caractéristiques spatiales et spectrales variées, prétraitées afin d'être combinées efficacement par la suite. Selon nous, un travail important reste à mener dans le domaine de la fusion spatio-spectrale concernant le développement de telles caractéristiques

personnalisées. Nous estimons par ailleurs que les *pipelines* d'apprentissage spécifiques, comme nous avons proposé avec CTIS-Net pour les images CTIS et comme il existe déjà dans le domaine des sciences végétales [Louargant et al., 2018] auront aussi leur rôle à jouer.

Nous pensons que les travaux dans cette optique sont encore peu avancés car les signaux aux formats non-standards tels que les images CTIS sont encore peu exploités dans des *pipelines* d'apprentissage automatique, que cela soit dans un cadre de fusion ou non. Rappelons que le *pipeline* standard de l'imagerie CTIS mène à la génération d'un cube hyperspectral, dont la fusion avec une éventuelle image RVB est abondamment étudiée via les algorithmes d'affinage panchromatique. Cependant, il n'existe que peu de travaux portant sur la fusion de signaux véritablement non standards, tels que les signaux comprimés, avec des imageries classiques. Des progrès dans cette voie bénéficieraient pourtant au champ de l'apprentissage comprimé dans son ensemble, en permettant d'intégrer plus aisément les méthodes concernées en complément d'autres modalités. Ces avancées seraient d'autant plus fructueuses que d'autres modalités actuellement acquises conjointement avec les imageries RVB comme le lidar et le radar commencent tout juste à être étudiées dans le cadre de l'acquisition comprimée [Sher et al., 2018; Yang et al., 2019].

Conclusion et perspectives

Sommaire

Retour et perspectives pour les travaux effectués 130
Un jeu simulé pour le capteur CTIS 130
Apprentissage comprimé sur images CTIS 131
Simulation d'images pour alléger la charge d'annotation d'images réelles 132
Vers un monde plus sobre 134
Valorisations 136

Retour et perspectives pour les travaux effectués

Dans ce manuscrit, nous avons abordé plusieurs problématiques liées à l'acquisition d'images via le spectromètre RVB-IR-CTIS de l'entreprise Carbon Bee. Nous avons en particulier développé des simulateurs d'images innovants destinés à appuyer l'analyse de scènes complexes en lumière IR et à évaluer la viabilité du CTIS, un capteur hyperspectral jusqu'alors peu étudié.

Un jeu simulé pour le capteur CTIS

Afin d'étudier la viabilité d'analyses automatiques menées par apprentissage profond sur une problématique d'intérêt agronomique, nous avons conçu et implémenté plusieurs simulateurs d'images. D'une part, nous avons généré des cubes hyperspectraux de feuilles de pommier tavelées à partir d'images de feuilles saines. D'autre part, nous avons bâti un simulateur du système optique CTIS qui permettait de convertir des cubes hyperspectraux en images CTIS. Nous avons pu grâce au couplage de ces deux simulateurs procéder à la création d'images CTIS de feuilles tavelées visuellement proches d'acquisition réelles, et ce en nombre virtuellement illimité. Nous avons par ailleurs obtenu des premiers résultats encourageants concernant la transférabilité des résultats d'images simulées vers des images réelles (annexe A). Ces images simulées nous ont permis de mener des expériences encore inédites dans le champ de recherche de l'imagerie hyperspectrale.

Nous pensons que les travaux portant sur la simulation des cubes hyperspectraux de feuilles tavelées pourraient être étendus. Il est possible pour commencer de s'intéresser à d'autres maladies végétales que la tavelure. Il existe en effet de nombreuses infections affectant les plantes dont les spectres moyens ont été déterminés via l'utilisation de caméras hyperspectrales [Thomas et al., 2018]. Parmi ces maladies, certaines d'entre elles causent une déviation spectrale faible par rapport au spectre de la plante saine. Se circonscrire à ces cas permettrait de mettre davantage en valeur l'intérêt de l'imagerie hyperspectrale puisque celle-ci y apporte un vrai bénéfice par rapport à des imageries plus classiques (cf. section 4.4.3). Pour aller plus loin, il est aussi possible de se tourner vers des simulateurs spectraux tels que PROSPECT [Jacquemoud and Baret, 1990]. Ce simulateur permet de générer des spectres de feuilles en fonction de leur contenu en chlorophylle, caroténoïdes, contenu aqueux, etc. Nous nous sommes d'ailleurs servis de ce simulateur pour produire la figure 1.2) d'une plante saine, mais il peut aussi permettre de représenter des situations de stress telles que des carences de nutriments ou des sevrages en eau. Ces cas d'étude ont une importance agronomique et industrielle forte (cf. section 1.1.3), ainsi qu'un intérêt scientifique puisque les signatures spectrales des symptômes peuvent être subtiles et limitées à certaines gammes restreintes de longueurs d'onde.

Concernant le simulateur du système CTIS, nous avons implémenté une grande variabilité des paramètres optiques et spectraux (section 3.2), que nous n'avons toutefois pas exhaustivement explorée au cours de ce travail (voir par exemple les différentes géométries d'images CTIS possibles, figure 3.15). Nous espérons qu'ainsi ce simulateur pourra être utile à d'autres études portant sur le CTIS, même dans le cas de configurations optiques différentes de celles étudiées dans ce manuscrit. De plus, le découplage volontaire de ce simulateur par rapport au procédé de génération de cubes hyperspectraux en fait un outil agnostique aux cubes fournis en entrée, pertinent pour des cas d'étude divers. Enfin, nous avons pris soin d'implémenter le calcul automatique de la matrice d'action du système quels que soient les paramètres optiques choisis (section 3.2.4). Une perspective d'utilisation de ce simulateur pourrait être son usage à des fins d'évaluation des nouvelles stratégies de reconstruction des cubes hyperspectraux à partir de l'espace de mesures. En effet, de nombreuses innovations ont été apportées au processus de reconstruction dans le domaine du CT sans jamais être adaptées au champ connexe de l'imagerie CTIS (section 2.2.4). Dans ce manuscrit, nous n'avons pas proposé de contribution concernant cette reconstruction, mais nous estimons que cet axe n'est pas à négliger. Malgré les avantages de l'apprentissage comprimé pour certaines applications, l'acquisition d'un cube hyperspectral complet peut avoir de la valeur dans

d'autres, comme par exemple dans un cadre de recherche de longueurs d'onde optimales pour une détection.

Apprentissage comprimé sur images CTIS

Grâce aux deux simulateurs développés, nous avons mené pour la première fois un apprentissage comprimé réalisé sur des images CTIS dans le cadre d'une classification binaire. Nous avons pu montrer que les performances d'apprentissage étaient comparables à celles obtenues grâce à un apprentissage mené sur des cubes hyperspectraux reconstruits, et ce avec un temps d'entraînement et de prédiction considérablement réduits. Nous avons de plus conçu une nouvelle architecture neuronale spécifique aux images CTIS qui a permis d'améliorer les performances de classification par rapport à l'emploi d'une architecture générique. Par ailleurs, nous avons comparé les performances obtenues avec l'imagerie CTIS à celles permises par les imageurs RVB et IR. Si pour notre cas d'étude, ces imageurs standards étaient autant, voire plus, efficaces que l'imagerie CTIS, nous avons souligné la polyvalence du CTIS qui permettait d'étudier une large gamme de longueurs d'onde d'une scène sans *a priori* tandis qu'une utilisation des imageurs standards nécessitait une connaissance en amont des longueurs d'onde optimales. De surcoît, nous avons montré qu'un gain de performance était possible en exploitant conjointement le CTIS avec ces imageurs plus standards.

Nous voyons deux axes de recherche concernant l'apprentissage comprimé sur images CTIS. Le premier consiste à poursuivre la recherche d'architectures et de prétraitements spécialisés pour l'analyse de ces images. Il pourrait par exemple être bénéfique d'intégrer de façon structurelle une éventuelle information *a priori* sous la forme de longueurs d'onde connues comme optimales ou bien supposées comme telles. Puisque la position de ces longueurs d'onde dans les projections de l'ordre 1 sont fixes et connues d'avance, un simple prétraitement consistant à augmenter la luminosité des zones concernées dans l'image pourrait suffire pour mettre en avant cette information « aux yeux » du réseau (figure C.1, a). Nous pouvons aussi imaginer plutôt intégrer cette information à la structure du réseau via l'intégration de mécanismes d'attention [Vaswani et al., 2017]. Un autre prétraitement possible consisterait à approximativement superposer les différentes projections de l'ordre 1. Il s'agirait d'appliquer des rotations et des retournements à chaque projection afin de les aligner spectralement entre elles. Il serait alors possible de concaténer celles-ci selon l'axe des canaux afin de regrouper les informations spectrales et ainsi rendre l'exploitation des caractéristiques plus aisée pour le réseau (figure C.1, b) [25]. Une autre voie, compatible avec un tel repliement ou bien avec le procédé d'alignement que nous avons proposé pour CTIS-Net (figure 4.13), consisterait en l'implémentation de noyaux convolutifs aux formes non-standards. Nous pourrions imaginer des noyaux en forme de « + » ou bien un ensemble au sein d'une même couche de noyaux horizontaux et verticaux, ce qui permettrait d'intégrer les informations spatiale et spectrale des projections de façon plus décorrélée qu'un noyau carré (figure C.1, c). Une telle configuration n'est pas sans rappeler les opérations de convolutions séparables selon la profondeur (*depthwise separable convolutions* en anglais) [Chollet, 2017] développées à des fins de réduction du nombre de paramètres des réseaux afin de les implémenter sur des supports embarqués. Nous pourrions imaginer adapter la philosophie de cette implémentation afin d'améliorer l'exploitation d'une information spatio-spectrale.

Le deuxième axe consiste, plutôt qu'à rechercher des modifications du *pipeline* d'apprentissage spécifiques aux images CTIS, à examiner celles qui bénéficieraient à d'autres signaux comprimés. Il s'agit en d'autres termes d'analyser s'il existe des intuitions développées lors de l'étude de l'apprentissage comprimé sur images CTIS qui pourraient être transposables à d'autres signaux.

25. Nous qualifions cette superposition d'« approximative » car la superposition exacte des projections n'est pas possible. En effet, comme le montre la figure 3.13, les projections ne sont pas symétriques par rapport à l'ordre 0, comme nous pourrions le croire à première vue, puisque les tranches spectrales individuelles ne sont pas retournées par l'action du réseau de diffraction.

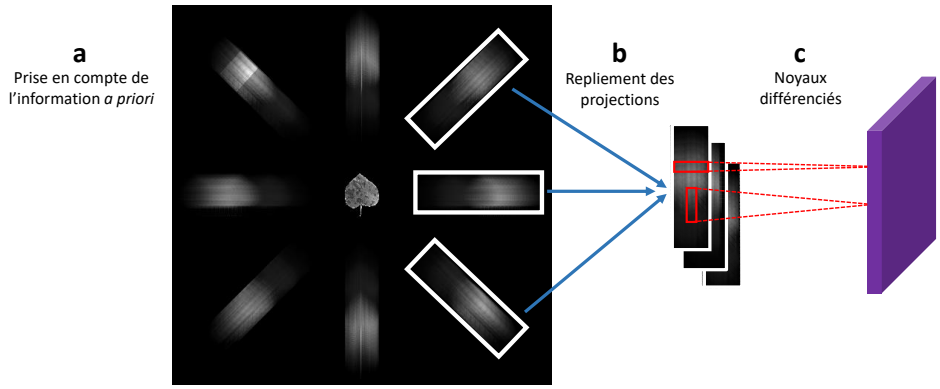


FIGURE C.1 – Illustrations de quelques perspectives de spécialisation pour un réseau dédié aux images CTIS, potentiellement complémentaires entre elles. Pour des raisons de clarté, la piste « a » n'est illustrée que pour la projection « nord-ouest » et la piste « b » n'est illustrée que pour les trois projections « est ».

Comme discuté à la fin du chapitre 6, nous pensons en particulier aux imageries lidar et radar. En plus d'être d'ores et déjà étudiées dans le cadre de l'acquisition comprimée (section 4.3.1), ces imageries intègrent une étape de reconstruction à partir de mesures indirectes (construction d'images à partir de mesures diélectriques pour les radars imageurs [Zhou et al., 2019], construction de surfaces à partir de nuages de points pour le lidar [Sampath and Shan, 2009]) qui en font des candidates désignées pour une étude par apprentissage comprimé.

Simulation d'images pour alléger la charge d'annotation d'images réelles

En plus des travaux exploratoires menés sur l'imagerie CTIS, nous avons approché le problème de détection de la tavelure dans un cadre plus industriel en travaillant sur une segmentation de lésions de tavelure dans des images réelles de canopées. Très vite, il est apparu que le temps d'annotation de données réelles constituait un goulot d'étranglement pour mener à bien un apprentissage, une difficulté commune à de nombreuses autres applications. Nous nous sommes tournés en réponse vers la génération d'images simulées venant en appui de l'entraînement. Nous avons adapté des méthodes à notre cas d'étude parmi les trois grandes familles de simulation qui constituaient l'état de l'art dans le champ de la vision par ordinateur. Nous avons comparé les gains de performance permis par ces méthodes, en étudiant pour chacune d'elles diverses façons d'intégrer les données simulées à l'apprentissage sur données réelles. Nous avons en particulier mis en avant les difficultés de convergence des solutions basées sur les GANs et souligné l'apport substantiel d'un pré-entraînement sur des images synthétiques issues d'un modèle développé spécifiquement pour notre cas d'étude.

Parmi tous les sous-domaines du champ de l'apprentissage automatique, les architectures des GANs sont sans doute celles qui ont le plus progressé entre le début des travaux présentés dans ce manuscrit et la rédaction de celui-ci. Revisiter les travaux réalisés concernant la simulation de données par GAN armés de ces nouvelles architectures permettrait sans doute d'atténuer les difficultés de convergence que nous avons rencontrées. En effet, il est notoire que ces architectures modernes sont capables de générer des images de plus grande taille et à un niveau de détail bien plus fin que les DCGANs, et ce de façon plus stable. Nous pourrions dans un premier temps simplement ré-implémenter le protocole que nous avons suivi dans cette étude en remplaçant le DCGAN par une architecture plus moderne telle que le StyleGAN. Mais les dernières avancées en transfert de style permettent d'imaginer d'autres protocoles où un CycleGAN pourrait apprendre à générer des symptômes de maladies sur des feuilles saines, comme le fait notre simulateur « modèle de canopée ». Resterait alors la question de l'adaptation de ces méthodes à une tâche de segmentation. Il apparait par ailleurs que les architectures conditionnelles telles que le GAN conditionnel (section 5.2.1) auraient sûrement et un fort intérêt pour améliorer la variabilité des

données produites. Nous pouvons imaginer en particulier générer des masques de tavelure selon un procédé proche de celui que nous avons implémenté pour le simulateur « modèle de canopée » (figure 3.4) et contraindre un GAN à générer des images de feuilles tavelées suivant ce masque.

Plus étonnant, le champ des déformation de données, c'est-à-dire l'application de transformations simples aux images du jeu initial, a également beaucoup progressé ces dernières années. Un sous-champ de l'apprentissage automatique appelé l'*automatic machine learning* en anglais [Yao et al., 2018], ou autoML, a été développé récemment. Ce champ regroupe les meta-algorithmes qui visent à optimiser les choix d'hyperparamètres, au sens large, d'un *pipeline* d'apprentissage automatique. Bien que l'autoML se concentre en particulier sur la recherche d'architectures neuronales [Zoph and Le, 2016], d'autres travaux du champ s'intéressent à toutes les autres étapes du *pipeline* d'apprentissage, de la collecte des données à l'évaluation des performances du réseau [He et al., 2021]. La déformation de données peut ainsi être optimisée via des méthodes destinées à sélectionner automatiquement les meilleures transformations et les valeurs de leurs paramètres à appliquer au vu du jeu de données [Cubuk et al., 2018]. Ces méthodes ont un apport en particulier lorsque les *a priori* structuraux de la scène à augmenter sont méconnus ou entachés d'incertitude. Il serait éclairant de comparer les déformations proposées avec ces méthodes par rapport à celles que nous avons choisies en nous basant sur la littérature.

Enfin, concernant le simulateur « modèle de canopée », nous pensons que la discussion initiée à la section 5.4.4 à propos de l'universalisation des simulateurs est essentielle. Plusieurs questions se cachent derrière cette notion d'universalité d'un modèle et nous estimons qu'une étude par ablation telle que celle menée à la section 5.4.2 permet d'y apporter des éléments de réponse. D'une part, une telle étude permet d'identifier, pour une tâche donnée, les traits de réalisme du simulateur qui sont les plus utiles pour améliorer la classification du jeu réel. Par exemple, dans notre cas d'étude, nous avions ainsi déterminé que la simulation de la courbure des feuilles constituait un trait de réalisme précieux (figure 5.20). Mais une telle étude est par ailleurs indicatrice, pour un simulateur donné, d'autres tâches qui pourraient bénéficier des images générées. En effet, les traits de réalisme n'amenant pas de gain de performance sur des données réelles, et donc écartables, permettent de partiellement généraliser l'usage du simulateur. Par exemple, dans notre cas d'étude, puisque nous avions conclu qu'il n'était pas nécessaire de simuler une structure en plants des feuilles, cela laissait supposer que ce trait n'était pas utile au réseau pour détecter des lésions de tavelure, et donc que des jeux de données présentant d'autres répartitions des feuilles pourraient bénéficier de ce simulateur. De plus, le gain de performance substantiel obtenu malgré une simplification extrême du simulateur suggérait un intérêt partiel de celui-ci pour des tâches variées dans le domaine des sciences végétales.

Ainsi, il serait enthousiasmant de se procurer d'autres jeux de données de segmentation du domaine des sciences végétales et d'explorer la transférabilité du simulateur à ces tâches. En particulier, nous pourrions imaginer créer un ensemble de simulateurs plus simples « contenus » dans le simulateur « complet », en « amputant » progressivement des étapes comme il a été fait au cours de l'étude par ablation, puis de mettre en place un pré-entraînement « hiérarchisé ». Nous entendons par là le *pipeline* consistant à mener, pour une tâche donnée, un premier pré-entraînement sur des images issues du simulateur le plus simple, puis d'enchaîner d'autres pré-entraînements sur des images issues de simulateurs de plus en plus en plus spécifiques si l'on constate que cela amène une augmentation des performances sur le jeu de données réelles étudié. Cette idée s'aligne avec la hiérarchisation sémantique des caractéristiques dans un CNN entraîné, où les premières couches apprennent des caractéristiques génériques à toute tâche de vision et les couches suivantes se spécialisent graduellement pour la tâche étudiée. Si une large gamme de jeux de données est disponible, nous pourrions même imaginer formaliser le niveau de précision optimal du simulateur pour un jeu donné en fonction d'une notion de « distance » au jeu pour lequel le simulateur a été initialement développé. Ainsi, le temps nécessaire pour le développement d'un

simulateur complet pourrait être davantage rentabilisé en créant un continuum de simulateurs variant entre universalité et spécificité. En outre, nous pensons que cette méta-procédure pourrait évidemment être appliquée à d'autres modèles hors du domaine des sciences végétales.

Vers un monde plus sobre

Les travaux que nous avons menés dans cette thèse s'inscrivent dans la tendance actuelle de l'apprentissage en peu d'exemples (*Few-Shot Learning* en anglais, ou FSL) [Wang et al., 2020b]. Ce paradigme a été développé récemment en réaction aux exigences démesurées de l'apprentissage profond en termes de nombre de données et de temps de calcul. Il regroupe l'ensemble des méthodes qui permettent de mener à bien un apprentissage profond avec un nombre de données annotées restreint. La revue de littérature de [Wang et al., 2020b] catégorise ces méthodes selon trois axes : augmenter les données afin de se ramener au cas de l'apprentissage profond classique, contraindre l'architecture du réseau pour limiter l'espace des frontières de décision à rechercher, aider l'optimiseur de poids à effectuer cette recherche plus efficacement. Les travaux de ce manuscrit se sont inscrits dans un de ces axes, que cela soit via l'implémentation de diverses méthodes de simulation de données, l'utilisation de réseaux de taille réduite ou encore l'implémentation de pré-entraînements sur des tâches afférentes afin de faciliter l'optimisation des poids [26].

De plus, il nous semble que l'apprentissage comprimé, paradigme auquel nous avons consacré une part importante de nos travaux, est compatible avec la philosophie plus large du FSL. En travaillant avec succès directement sur des signaux « intermédiaires » d'un capteur d'imagerie computationnelle et en évitant ainsi la reconstruction des signaux originaux, la génération de données elle-même a un coût limité. Nous considérons d'ailleurs que le système optique du CTIS lui-même est proche de cette sensibilité puisqu'il permet d'acquérir des images hyperspectrales pour une fraction du prix et du temps d'acquisition nécessaires aux caméras à balayage standards et est partie prenante du développement récent de caméras hyperspectrales à bas coût (section 2.1.2). Pour résumer en un mot le leitmotiv des travaux que nous avons menés pour réduire tous les coûts des *pipelines* d'apprentissage automatique, qu'ils soient liés à l'acquisition des données, leur traitement, leur annotation, ou bien au procédé d'apprentissage en lui-même, nous estimons que nous avons suivi une démarche de *sobriété*.

Pour conclure, nous souhaiterions arguer — et tout ce qui suit n'engage que l'auteur de ces lignes — que la sobriété prônée dans des domaines comme le FSL et l'apprentissage comprimé reflète une frugalité dont la nécessité s'impose de plus en plus nettement, dans le domaine de l'apprentissage automatique certes, mais plus largement dans la société humaine en général. Il est notoire que l'apprentissage profond est un champ de recherche gourmand en ressources. Les entraînements réalisés par les géants du domaine requièrent des parcs de cartes graphiques et une consommation électrique en conséquence. À cela s'ajoute le prix d'acquisition des données : nous pouvons par exemple citer les voitures autonomes de Tesla qui circulent uniquement afin de fournir des images destinées à entraîner des nouvelles voitures autonomes [27] [Karpathy, 2020]. De plus, les succès du domaine dopent le déploiement massif d'objets connectés [Shanthamallu et al., 2017] eux aussi énergivores. Comme l'accès à une source gargantuesque d'énergie sous la forme de charbon a permis l'avènement de la révolution industrielle il y a deux siècles, le développement de la puissance de calcul et des moyens d'acquisition massive de données a permis l'envol du champ de l'apprentissage profond voilà une décennie. Mais aurons-nous toujours les moyens de soutenir

26. Il faut noter que les méthodes développées récemment qui se revendiquent explicitement comme adoptant le paradigme FSL suivent des protocoles élaborés, tirant souvent partie d'un co-apprentissage sur des tâches connexes pour lesquelles on dispose de jeux de données réels annotés conséquents. Nous n'avons pas développé de telles méthodes, mais estimons malgré cela que nos travaux ont suivi les objectifs affichés du FSL.

27. … situation qui, pour le cycliste convaincu qu'est votre serviteur, incarne une certaine idée du neuvième cercle de l'Enfer.

nos ambitions ?

M. Jancovici, ingénieur spécialisé dans le domaine de l'énergie et du dérèglement climatique, souligne depuis des années le lien causal entre disponibilité de l'énergie fossile — en particulier, le pétrole — et PIB mondial [Jancovici, 2019]. Selon lui, aucune source d'énergie ne pourra remplacer à court ou moyen terme le pétrole, dont le pic de production est à présent derrière nous [Jancovici, 2020]. Par conséquent, il prévoit une décroissance inéluctable du PIB, qui se traduira à la fois par un déclin de la production industrielle et de l'investissement dans la recherche. Pour lui, il ne sera pas possible de conserver tout l'héritage technologique des deux derniers siècles. Il y aura donc un arbitrage à réaliser quant aux technologies que nous souhaiterons conserver, ainsi que celles dans lesquelles nous continuerons à investir en recherche et développement malgré un déclin continuel des ressources disponibles. Si ce scénario se vérifie, la question de la pertinence de l'intelligence artificielle (IA) dans nos sociétés pourra être posée, en particulier lorsque l'intérêt de l'application associée est discutable [Branwen, 2019], voire dangereuse [Korshunov and Marcel, 2018]. S'il n'est sans doute pas de bon ton de conclure un manuscrit traitant de l'IA par une remise en question de l'intérêt de l'intégralité du champ pour l'humanité, nous estimons que l'IA gourmande ayant basé sa renaissance sur la consommation massive d'énergie n'a pas nécessairement sa place dans le monde à venir, en particulier lorsque les applications qui en découlent ne permettent pas un progrès ou une augmentation du bonheur évident de l'humanité. Même dans le cas d'applications plus vertueuses qu'il pourrait être pertinent de pérenniser et développer — et nous considérons par exemple les travaux de Carbon Bee participant à la transition vers une agriculture plus verte comme tels — , nous espérons que la philosophie de sobriété dont nous avons brossé quelques traits dans ce manuscrit constituera un des piliers de cette IA du futur.

Valorisations

Publications

Les travaux menés dans ce manuscrit ont été publiés sous plusieurs formes. Pour valoriser la transdisciplinarité de ces travaux, nous avons pris soin de les présenter à plusieurs communautés scientifiques : imagerie computationnelle, vision par ordinateur pour sciences végétales, optique.

- **Journaux internationaux** :
  - *Computers and Electronics in Agriculture* pour les travaux concernant la simulation de données pour améliorer la détection de tavelure en conditions industrielles (chapitre 5) [Douarre et al., 2019c].
  - *Applied Optics* pour les travaux concernant l'apprentissage comprimé sur images CTIS avec VGG (chapitres 3 et 4) [Douarre et al., 2020a].
  - (en révision) *IEEE Transactions on Computational Imaging* pour les travaux concernant l'apport de l'architecture CTIS-Net à l'apprentissage comprimé sur images CTIS.
- **Conférence internationale** :
  - (en révision) *Optical Society of America Digital Holography and Three-Dimensional Imaging Topical Meeting* pour les travaux du chapitre 4 ainsi que l'étude de transférabilité entre images CTIS simulées et réelles (annexe A).
- **Ateliers internationaux** :
  - *Workshop on Machine Learning Assisted Image Formation* pour les premiers résultats des travaux des chapitres 3 et 4 [Douarre et al., 2019b].
  - *International Workshop on Image Analysis Methods in the Plant Sciences* pour les travaux concernant le recalage d'images réelles RVB et IR (annexe B) [Douarre et al., 2019a].
- **Dissémination nationale** :
  - Rencontres du végétal pour les travaux du chapitre 5 [Douarre et al., 2018b].
  - Réunion du Groupement de Recherche « Information, Signal, Images et Vision », journée à thème « Intelligence artificielle / Apprentissage machine pour l'agriculture » pour les travaux des chapitres 3 et 4 [Douarre et al., 2020b].
  - (à venir) Réunion du Groupement de Recherche « Information, Signal, Images et Vision », journée à thème « Imagerie Optique Non Conventionnelle » pour les travaux du chapitre 4.

Codes source et jeux de données

Le code du simulateur de CTIS présenté au chapitre 3 est disponible aux adresses https://gitlab.liris.cnrs.fr/cdouarre/ctis-simulator et https://github.com/CarbonBee/ctis-simulator.

Le jeu de données annoté d'images IR de canopées de feuilles tavelées étudié au chapitre 5 est disponible à l'adresse http://eidolon.univ-lyon2.fr/cdouarre_ScabIrDataset/dataset.zip.

Le jeu de données d'images CTIS simulées et réelles utilisé pour l'étude de transférabilité présentée à l'annexe A est disponible à l'adresse http://eidolon.univ-lyon2.fr/MNIST-CTIS-datasets.

Annexes

Sommaire

A Combler « l'écart de la réalité » 138
A.1 Cas d'étude 138
A.2 Entraînement et résultats 142
A.3 Conclusion 145
B Recalage d'images multimodales 146
B.1 Contexte : un décalage substantiel 146
B.2 Différentes familles de recalage 146
B.3 Application pour notre cas d'étude 148
B.4 Conclusion 152

A Combler « l'écart de la réalité »

Cette annexe présente les travaux exploratoires que nous avons menés afin d'évaluer la *transférabilité* des apprentissages réalisés sur des images CTIS simulées à des images CTIS réelles. Nous désignons par ce terme la différence de performance de classification entre l'utilisation des poids appris sur un jeu simulé pour une prédiction (i) sur des données simulées issues du même simulateur d'une part et (ii) sur des données réelles de l'autre. Une différence faible indique que la frontière de décision apprise sur les données simulées est pertinente pour proposer des prédictions pour les données réelles. Dans ce cas, cette haute transférabilité permet ainsi d'étendre au moins partiellement les conclusions tirées des apprentissages sur données simulées aux données réelles qu'elles représentent. Afin d'étudier spécifiquement la qualité de ce transfert pour les images CTIS indépendamment de l'objet acquis par le spectromètre, nous nous sommes placés dans le cas d'une tâche de classification « jouet », très simple en conditions simulées comme en conditions réelles.

A.1 Cas d'étude

Nous présentons dans cette section la constitution des jeux réel et simulé sur lesquels nous nous sommes basés pour mener l'étude de transférabilité.

A.1.1 Jeu réel

Pour constituer un jeu d'images CTIS réelles, nous avons commencé par éclairer avec une source lumineuse définie des transparents où des formes étaient imprimées de façon opaque. L'information lumineuse résultante à la sortie de ce transparent constituait la scène que nous souhaitions acquérir. Nous avons créé un montage optique reproduisant le CTIS de la caméra Carbon Bee (figure A.1) et avons acquis par ce biais les images CTIS de ces scènes. Les images acquises étaient de dimension 820 × 820 pixels, avec un ordre 0 de dimension 100 × 100 pixels.

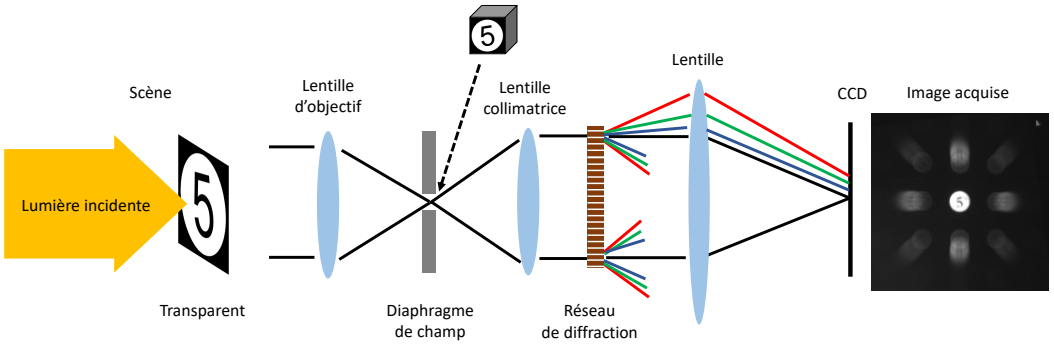


FIGURE A.1 – Schéma du banc optique utilisé pour l'acquisition d'un jeu d'images réelles CTIS .

Afin de créer une tâche de classification, nous avons fait varier les scènes acquises selon deux critères. Premièrement, la forme imprimée sur le transparent pouvait représenter soit le chiffre « 5 » soit le chiffre « 9 » [28]. Deuxièmement, l'éclairage employé pouvait provenir soit d'une lampe halogène à incandescence, soit d'une diode électro-luminescente (DEL). Seize transparents différents étaient disponibles représentant huit formes de « 5 » et huit formes de « 9 » à polices, tailles, et angles de rotation variés. Nous avons éclairé chacun d'eux avec une lampe à incandescence, puis avec une DEL. Nous avons ainsi créé un jeu de trente-deux images CTIS. Nous avons associé à ce jeu la tâche de classification qui consistait à reconnaître à la fois le chiffre représenté et le spectre

28. L'utilisation de chiffres en tant que formes était inspirée par le jeu de données MNIST [LeCun, 1998], un jeu « jouet » de classification d'images de chiffres manuscrits régulièrement étudié dans les applications d'apprentissage dans le domaine de la vision par ordinateur.

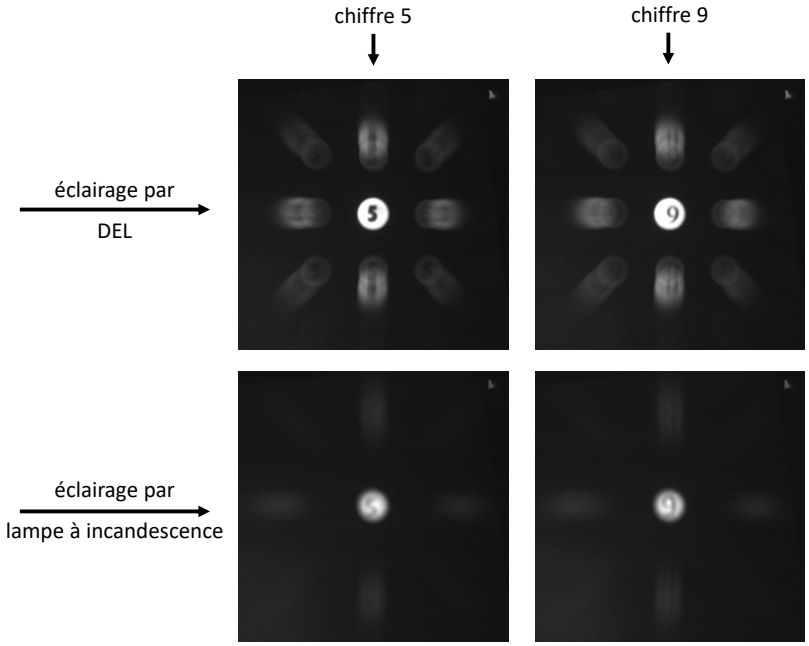


FIGURE A.2 – Exemples d'images de $D_{\text{CTIS réel}}$. Des exemples de chacune des quatre classes du jeu y sont présentés.

employé. Il s'agissait donc d'une classification à quatre classes que nous avons notées {« 5 DEL », « 5 INC », « 9 DEL », « 9 INC » }. Chaque classe comprenait huit images. Nous avons noté $D_{\text{CTIS réel}}$ ce jeu. La figure A.2 présente quelques images de ce jeu.

Nous pouvons observer dans les images de la figure A.2 plusieurs des « irrégularités » mises en avant figure 3.24. Nous y retrouvions en particulier un bruit présent dans toute l'image, une dilatation de la taille des projections de l'ordre 1 à mesure que l'on s'éloignait du centre de l'image ainsi qu'un flou notamment au niveau de l'ordre 0.

A.1.2 Jeu simulé

Nous avons créé un jeu simulé pour produire *in silico* des images proches de celles de $D_{\text{CTIS réel}}$. Nous avons d'abord reproduit sous la forme de cubes hyperspectraux les scènes acquises au cours de l'expérience réelle. Puis, nous avons généré des images CTIS à partir de ces cubes via le simulateur CTIS (section 3.2).

Afin de créer les cubes hyperspectraux, nous avons commencé par générer les spectres des sources lumineuses employées. Pour créer le spectre d'une DEL, nous nous sommes basés sur les données des travaux de [Singh, 2009]. Une lampe halogène, quant à elle, était suffisamment proche d'un corps noir pour que nous puissions calculer son spectre via la loi de Planck pour une température de 4000 K en suivant l'équation 1.1. Nous avons produit ces deux spectres pour la gamme de longueurs d'onde 400-1000nm afin de correspondre à la sensibilité spectrale du CCD réel utilisé (figure A.3). Nous avons discrétisé ces spectres à 80 valeurs.

Nous avons par la suite généré les formes imprimées sur les transparents. Nous avons créé 2000 images représentant les chiffres « 5 » et « 9 » en blanc sur fond noir selon différentes polices de caractère, épaisseurs de trait et tailles tirées aléatoirement. Les polices étaient celles proposées par OpenCV, sans que nous ne portions une attention particulière à leur correspondance avec celles employées dans le jeu réel. Nous avons également appliqué une légère rotation à ces chiffres, dont l'angle de rotation était tiré aléatoirement dans $[-\frac{\pi}{12}, \frac{\pi}{12}]$ radians. Nous avons par la suite combiné ces formes avec les spectres des lampes afin de générer des cubes hyperspectraux en

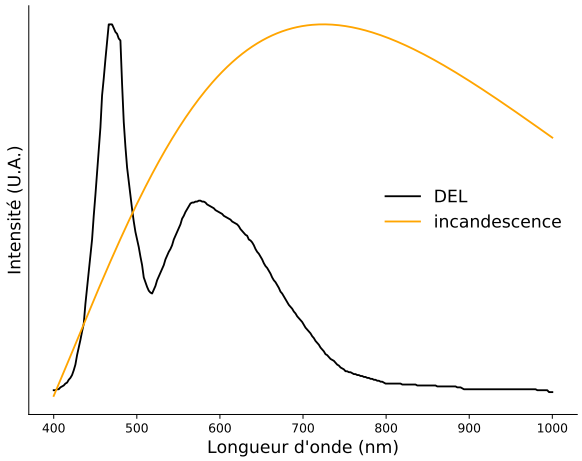


FIGURE A.3 – Spectres de DEL et de lampe à incandescence normalisés que nous avons utilisé dans cette annexe.

suivant l'algorithme A.1. Les étapes de cet algorithme indiquées en bleu sont illustrées à la figure A.4. Nous avons généré ainsi $n = 2000$ cubes composés de $n_\lambda = 80$ tranches spectrales.

Algorithme A.1 : Création de la base de cubes hyperspectraux simulés.

Entrées : Les spectres DEL et incandescence S_{DEL} et S_{INC}, les jeux d'images « 5 » et « 9 » $Chiffre_5$ et $Chiffre_9$, la dimension spatiale des cubes hyperspectraux à produire d.

n_λ = longueur(S_{DEL}) .
n = longueur($Chiffre_5$) .

pour C ∈ {5;9} **faire**
  pour $i \in [1, n]$ **faire**
    $I_C = Chiffre_C[i]$.
    Redimensionner I_C à la dimension $d \times d$ pixels.
    Tirer au hasard la source lumineuse L parmi {DEL, INC}.
    /* Création des tranches spectrales. */
    pour $\lambda \in [1, n_\lambda]$ **faire**
      Créer une image I_λ de dimension $d \times d$ pixels noire.
      Ajouter à I_λ un cercle centré, de rayon $\frac{d}{2}$, uniforme de valeur $S_L[\lambda]$.
      /* Action du transparent. */
      Soustraire l'image I_C à I_λ.
    fin
    /* Création du cube hyperspectral. */
    Concaténer les images I_λ selon l'axe des canaux.
  fin
fin

Sortie : Un jeu de n cubes hyperspectraux représentant des acquisitions de chiffres « 5 » et « 9 » éclairés en lumière DEL ou incandescence, de dimension $d \times d \times n_\lambda$ voxels.

Enfin, nous avons généré une image CTIS à partir de chacun des cubes ainsi créés. Nous avons utilisé le simulateur CTIS avec les paramètres précisés dans le tableau A.1. Nous avons ajouté un bruit gaussien aux images en suivant la procédure décrite à la section 4.5. Nous avons associé à chaque image CTIS une étiquette en fonction du chiffre et du spectre attribués. Nous avons appelé $D_{CTIS\ simulé}$ ce jeu (figure A.5).

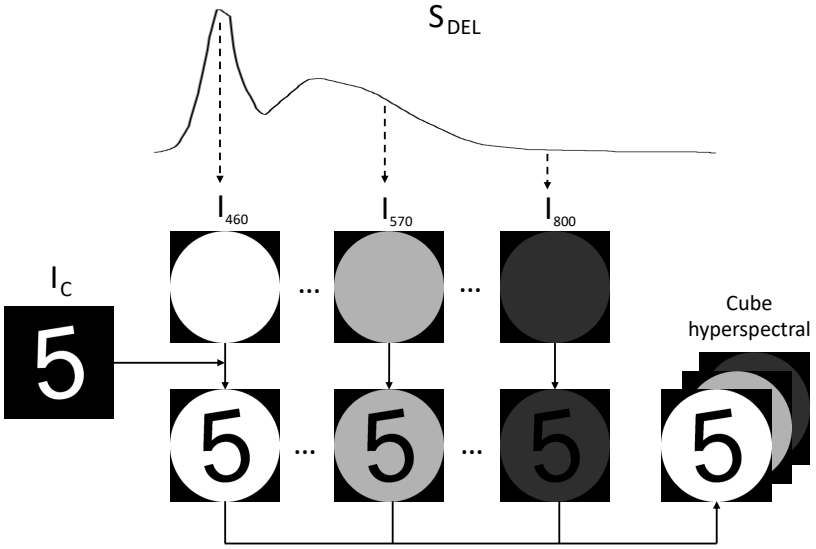


FIGURE A.4 – Illustration des étapes indiquées en bleu de l'algorithme A.1.

Paramètre	Valeur
d	820 pixels
d_0	100 pixels
λ_{min}	400 nm
λ_{max}	1000 nm
Géométrie	Rectangulaire, deux ordres

TABLEAU A.1 – Paramètres utilisés pour le simulateur CTIS pour la création de $D_{CTIS\ simulé}$.

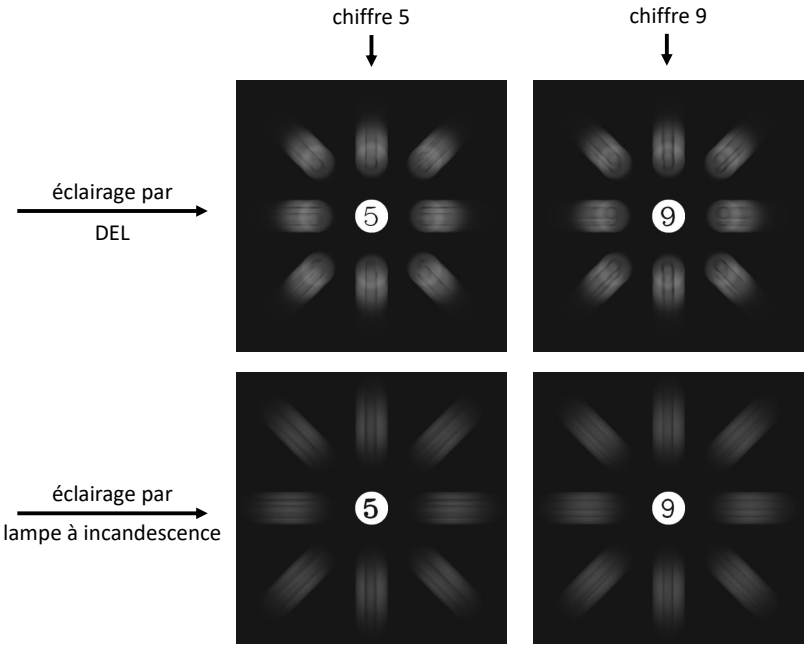


FIGURE A.5 – Exemples d'images de $D_{CTIS\ simulé}$. Des exemples de chacune des quatre classes du jeu y sont présentés.

A.2 Entraînement et résultats

A.2.1 Protocole d'entraînement

Pour tous les apprentissages présentés dans cette annexe, nous avons suivi un protocole d'entraînement identique à celui décrit aux sections 4.1.2 et 4.4.1 à deux différences près. Premièrement, nous avons modifié la métrique de performance, le MCC n'étant défini que dans un cas de classification binaire. Nous avons choisi à la place une métrique standard en classification multi-classes : le taux de classification correcte, qui correspondait au ratio entre le nombre d'images bien classées et le nombre d'images total. Cette métrique ne s'interprétait pas exactement de la même manière que le MCC. En particulier, une valeur positive de cette métrique n'indiquait pas nécessairement une classification même partiellement réussie, puisqu'une prédiction aléatoire menait à une performance égale à $\frac{1}{\text{nombre de classes}}$, soit 0,25 dans notre cas. Notons aussi, même si cela ne nous concernait pas dans cette étude, que, contrairement au MCC, cette métrique était peu adaptée à une analyse pour des classes aux effectifs déséquilibrés. L'autre changement apporté au protocole était que, cette étude annexe ayant un caractère plus exploratoire que celles menées dans le manuscrit, nous avons réalisé une seule répétition par apprentissage au lieu de dix.

A.2.2 Expériences préliminaires

Avant toute expérimentation concernant les données réelles, nous avons réalisé une expérience préliminaire concernant les données simulées uniquement. Plus précisément, nous avons divisé $D_{\text{CTIS simulé}}$ en blocs d'entraînement, de validation et de test, conduit un entraînement avec VGGr (figure 4.2) sur le bloc d'entraînement avant de réaliser une prédiction sur le bloc de test. La performance était égale à 1, ce qui signifiait que la prédiction était parfaite. Ce résultat a permis de confirmer que la tâche de classification était très aisée pour un réseau de neurones, et donc que toute performance en dessous de 1 lors de l'application du réseau sur des données réelles serait attribuable à un manque de transférabilité.

Afin d'étudier la transférabilité du savoir acquis sur les images simulées pour les images réelles, nous avons conduit un entraînement sur le bloc d'entraînement de $D_{\text{CTIS simulé}}$ jusqu'à convergence sur le bloc de validation de $D_{\text{CTIS simulé}}$, puis mené une prédiction sur $D_{\text{CTIS réel}}$. Les résultats obtenus indiquaient une transférabilité nulle. Cependant, nous nous sommes rendus compte que celle-ci s'améliorait considérablement lorsque nous utilisions pour la prédiction sur $D_{\text{CTIS réel}}$, à la place des poids convergés sur les données simulées, des poids provenant d'époques antérieures, avant la convergence. En d'autres termes, au cours de l'entraînement sur données simulées, le réseau semblait d'abord apprendre un savoir qui était transférable aux données réelles, avant de surapprendre [29] sur le jeu simulé. La figure A.6 illustre le comportement que nous observions.

La convergence basée sur le bloc de validation de $D_{\text{CTIS simulé}}$ n'était donc pas un bon critère d'arrêt pour une transférabilité optimale. Nous avons construit en réponse à cette observation le jeu $D_{\text{transférabilité}}$ constitué de l'intégralité du jeu $D_{\text{CTIS simulé}}$ en tant que bloc d'entraînement, de la moitié de $D_{\text{CTIS réel}}$ en tant que bloc de validation et de l'autre moitié en tant que bloc de test (figure A.7). La séparation de $D_{\text{CTIS réel}}$ a été réalisée en prenant en compte les classes des images de manière à obtenir autant d'images de chaque classe dans le bloc de validation de $D_{\text{transférabilité}}$ que dans celui de test. Ainsi, des images réelles servaient à évaluer si l'entraînement sur des images simulées avait convergé. Nous avons utilisé ce jeu pour conduire les expériences de transférabilité présentées dans les sections suivantes.

29. Le terme de « surapprentissage » est bien ici à comprendre comme « surapprentissage par rapport au bloc de test ». L'expérience préliminaire sur les données simulées a bien montré que le réseau était tout à fait capable d'apprendre sur données simulées sans surapprentissage rédhibitoire lorsque le bloc de test était lui aussi constitué de données simulées.

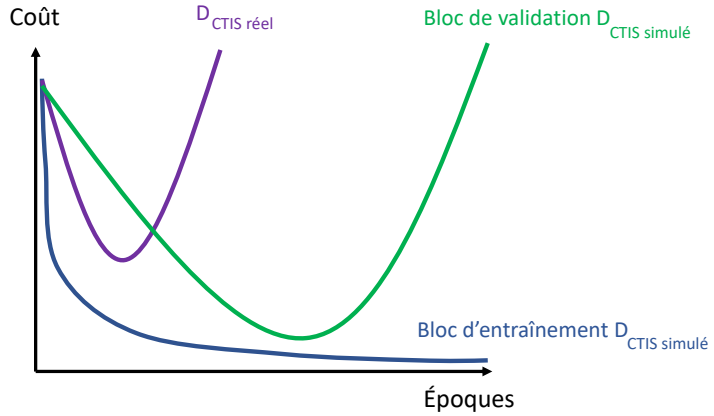


FIGURE A.6 – Illustration de l'évolution du coût au cours d'un entraînement sur le bloc d'entraînement de $D_{\text{CTIS simulé}}$ avec VGGr. Ces courbes sont à but illustratif et ne représentent pas des valeurs de coût mesurées au cours d'entraînements.

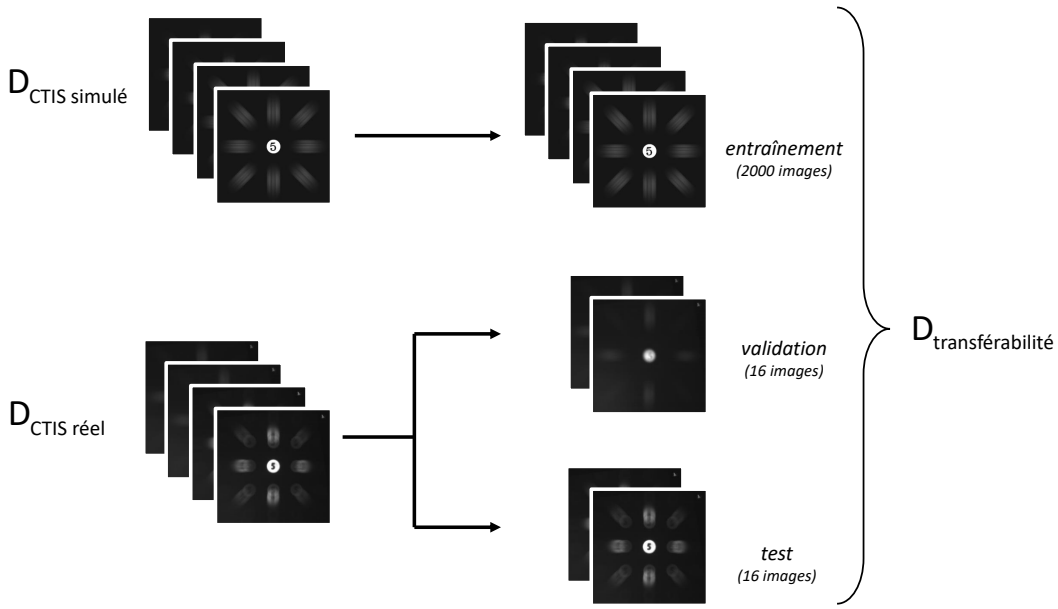


FIGURE A.7 – Création du jeu $D_{\text{transférabilité}}$ à partir des jeux $D_{\text{CTIS simulé}}$ et $D_{\text{CTIS réel}}$.

A.2.3 Étude de transférabilité avec VGGr

Nous avons mené un apprentissage sur $D_{\text{transférabilité}}$ avec VGGr. La performance de classification sur le bloc de test était de 0,5, ce qui indiquait une transférabilité partielle. La figure A.8 détaille ces résultats sous la forme d'une matrice de confusion associée à cette prédiction. Nous pouvons y voir que toutes les images éclairées en lumière DEL étaient correctement prédites comme telles (colonnes 1 et 3), et de même pour l'éclairage en lumière incandescente (colonnes 2 et 4). La sous-tâche concernant la discrimination « spectrale » était donc réussie. En revanche, la discrimination « spatiale », c'est-à-dire la détermination du chiffre représenté dans les images, avait échoué. Nous pouvons en effet constater que l'ensemble des images « 5 DEL » et « 9 DEL » avaient été prédites en « 5 DEL », et nous retrouvons un résultat similaire pour les classes INC. L'échec de la classification spatiale était surprenant car nous supposions avant de mener les apprentissages que la tâche de détermination du chiffre serait plus aisée pour le réseau que celle de la détermination spectrale. En effet, le chiffre employé apparaissait clairement dans l'ordre 0 des images de $D_{\text{transférabilité}}$, et la tâche paraissait ainsi du même ordre de difficulté que celle associée au jeu MNIST que l'on sait être relativement triviale pour les architectures récentes de réseaux de neurones (cf. note 28).

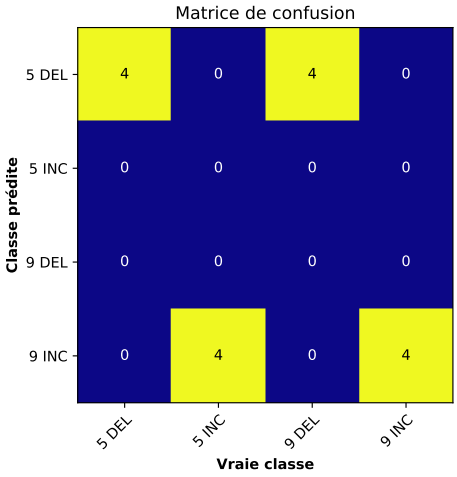
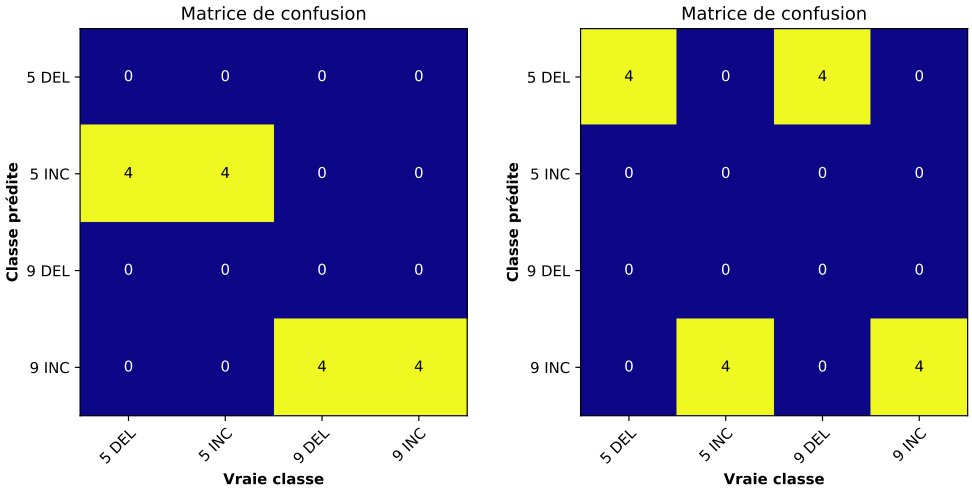


FIGURE A.8 – Matrice de confusion relative à la prédiction sur le bloc de test de $D_{\text{transférabilité}}$ avec VGGr.



(a) $D_{\text{transférabilité 0}}$.

(b) $D_{\text{transférabilité 1}}$.

FIGURE A.9 – Matrices de confusion relatives aux prédictions conduites sur les blocs de test de $D_{\text{transférabilité 0}}$ et $D_{\text{transférabilité 1}}$ avec VGGr. Les résultats sont identiques pour les prédictions conduites pour l'étude ablative de CTIS-Net.

Pour approfondir les résultats de cet apprentissage, nous avons créé les jeux $D_{\text{transférabilité 0}}$ et $D_{\text{transférabilité 1}}$ qui conservaient uniquement un des ordres des images. Nous avons créé ces jeux de la même manière que les jeux D_{CTIS0} et D_{CTIS1} présentés à la section 4.3.3, c'est-à-dire en mettant à zéro les pixels correspondant à l'ordre que nous ne souhaitions pas conserver, et ce dans les trois blocs du jeu. La performance sur chacun de ces jeux était de 0,5, mais les matrices de confusion (figure A.9) révélaient un apprentissage plus riche que l'expérience sur $D_{\text{transférabilité}}$. En effet, nous pouvons constater qu'en s'entraînant uniquement sur l'ordre 0, la sous-tâche « discrimination spatiale » était réussie, mais pas la « discrimination spectrale ». Nous pouvons tirer la conclusion inverse pour l'entraînement sur $D_{\text{transférabilité 1}}$. Ainsi, ces résultats montraient qu'une forte transférabilité était possible entre des images CTIS simulées et réelles. Néanmoins, la combinaison des deux informations de nature différente semblait perturber le transfert.

A.2.4 Étude de transférabilité avec CTIS-Net

En réponse aux résultats de l'expérience précédente, nous nous sommes intéressés au potentiel de l'architecture CTIS-Net (figure 4.12) sur ce jeu. Nous avons mené d'abord une étude préliminaire identique à l'étude par ablation conduite à la section 4.4.3, c'est-à-dire que nous avons conduit

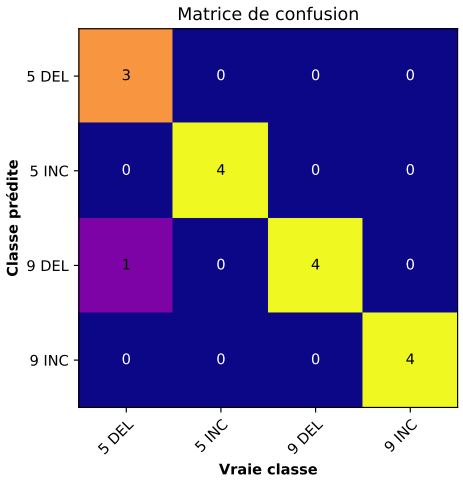


FIGURE A.10 – Matrice de confusion relative à la prédiction sur le bloc de test de $D_{\text{transférabilité}}$ avec CTIS-Net.

des apprentissages en utilisant uniquement une des branches du réseau. Les résultats étaient identiques à l'étude menée avec VGGr sur $D_{\text{transférabilité 0}}$ et $D_{\text{transférabilité 1}}$ (figure A.9) : l'apprentissage en exploitant uniquement l'information de l'ordre 0 permettait une discrimination spatiale parfaite mais la discrimination spectrale n'était pas possible (figure A.9, (a)), et inversement pour l'ordre 1 (figure A.9, (b)).

Cependant, lorsque nous avons mené un entraînement avec un CTIS-Net complet, qui intégrait donc l'information provenant des deux ordres, sur $D_{\text{transférabilité}}$, la performance était de 0,94, ce qui indiquait une transférabilité très haute. La matrice de confusion associée à cette expérience (figure A.10) confirmait ces résultats. Cette expérience soulignait l'intérêt de la fusion tardive et du protocole d'entraînement en deux temps suivi dans CTIS-Net pour combiner de façon optimale les informations spatiale et spectrale.

A.3 Conclusion

Les résultats de cette étude de transférabilité étaient encourageants car ils montraient pour un cas simple que le transfert d'un apprentissage réalisé sur des images CTIS simulées vers des images CTIS réelles était possible. Il n'y aurait donc pas d' « écart de la réalité » majeur dans le cas d'images CTIS. Par exemple, nous aurions pu craindre une certaine distribution spatiale de bruit qui aurait eu un grand impact sur l'apprentissage et qu'il aurait été très difficile de modéliser. Cette étude appuyait ainsi la validité de travailler sur des données simulées pour l'étude de ce spectromètre. Nous soulignons par ailleurs qu'il n'a pas été nécessaire pour la création des images simulées d'inclure des traits de réalisme approfondis pour que le transfert soit possible. Nous avons certes ajouté du bruit aux images, mais de manière *ad hoc* et sans nécessiter de modèle physique élaboré comme dans les travaux de certaines études [Garcia and Dereniak, 1999]. Nous n'avons inclus aucun mécanisme destiné à simuler les autres irrégularités listées à la section A.1.1.

Par ailleurs, ce cas jouet simple a permis de mettre en avant l'intérêt de la fusion tardive implémentée dans l'architecture CTIS-Net, en comparaison notamment avec l'utilisation du réseau standard VGGr sur l'image CTIS brute.

B Recalage d'images multimodales

B.1 Contexte : un décalage substantiel

Lorsque plusieurs capteurs distincts sont employés dans un cadre de vision par ordinateur multimodale, il est très courant que les différentes images obtenues lors d'une acquisition donnée ne soient pas alignées entre elles. Ce désalignement survient en particulier lorsqu'il existe un décalage physique entre les objectifs des capteurs qui acquièrent alors la scène selon des points de vue différents. On trouve un tel décalage dans la caméra Carbon Bee puisque les trois objectifs des capteurs RVB, IR et CTIS sont distants de quelques centimètres au sein du boitier (figure I.4). Le désalignement des images obtenues est d'autant plus fort que la distance entre la scène acquise et la caméra est faible par rapport à la distance entre les objectifs. Ainsi, le décalage est faible lors d'applications où la caméra est montée sur un drone et procède à des acquisitions plusieurs dizaines de mètres au-dessus du sol. Mais dans de nombreuses autres applications industrielles qui nécessitent que la caméra soit tenue à la main ou bien montée sur un tracteur, le désalignement peut poser problème. Comme discuté au chapitre 6, un tel désalignement est souvent néfaste pour les analyses multimodales et nécessite d'être corrigé.

Pour illustrer l'importance de ce désalignement dans ce cas, nous avons acquis un ensemble multimodal d'images de plants de pommiers avec la caméra Carbon Bee à l'IRHS. Les conditions d'acquisition étaient identiques à celles suivies pour la constitution du jeu $D_{original}$ présenté au chapitre 5. Nous avons acquis pour 50 positions différentes une image RVB et une image IR simultanément. Nous avons noté $D_{décalage}$ cet ensemble de 50 paires d'images (figure B.1). Nous pouvons voir dans cette figure que la distance entre la caméra et les plants était variable d'une paire d'images à l'autre en fonction de la taille des plants et de la hauteur de leur support. Nous pouvons y constater par ailleurs que les conditions de luminosité variaient en fonction de la localisation des plants dans la serre.

Le désalignement entre les images RVB et IR est illustré figure B.2, où nous avons souligné l'écart entre les positions de certaines feuilles spécifiques entre les deux images. Les décalages entre les encadrements jaunes et rouges de la sous-figure (b) indiquaient un désalignement suffisamment important pour nuire à l'apprentissage d'un réseau dans ses premières couches (cf. note 23). Ce désalignement pouvait être encore plus délétère dans le cas d'un tuilage comme celui que nous avons implémenté pour l'étude du chapitre 5. Aussi nous sommes-nous intéressés à procéder à un recalage des images qui viendrait en prétraitement d'un tel apprentissage.

B.2 Différentes familles de recalage

Dans le champ du traitement d'images, on appelle recalage le procédé permettant d'aligner deux images I_1 et I_2. Il s'agit plus formellement de trouver la transformation géométrique à appliquer à I_1 qui permet de maximiser une métrique de similarité entre I_1 transformée et I_2. Si les CNNs sont régulièrement employés pour trouver cette transformation [Haskins et al., 2020], il est nécessaire de leur fournir une quantité importante de paires d'images dont le décalage est connu. Dans notre cas, nous n'avions pas à notre disposition une telle base annotée, et nous nous sommes par conséquent tournés vers les méthodes de recalage issues du champ du traitement d'images. Il existe dans ce cadre deux familles d'algorithmes de recalage [Zitova and Flusser, 2003].

La première est basée sur la recherche de caractéristiques. Les caractéristiques les plus largement utilisées à cette fin sont celles issues de l'algorithme SIFT. Une caractéristique SIFT est définie par une position dans l'image, que l'on appelle un point saillant, et un vecteur de caractéristiques qui décrit le voisinage de ce point. Dans le cadre d'un recalage, ces caractéristiques sont calculées dans chacune des deux images, puis un appariement est réalisé entre les points saillants de chaque image en comparant leurs vecteurs descriptifs deux à deux et en associant les

FIGURE B.1 – Trois paires d'images de $D_{décalage}$, sans normalisation. Dans toutes les figures de cette annexe, les images RVB sont présentées dans la colonne de gauche et les images IR qui ont été acquises simultanément dans celle de droite.

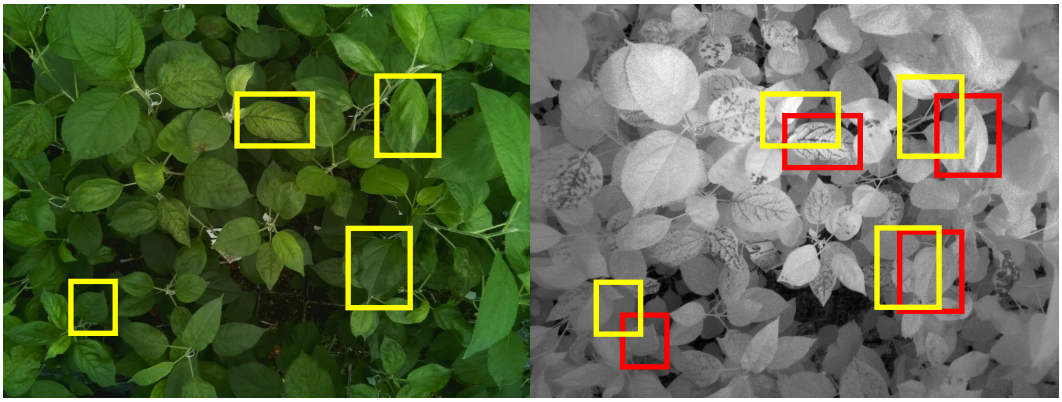


FIGURE B.2 – Illustration du désalignement entre les deux modalités sur une paire d'images de $D_{décalage}$. Dans l'image RVB, nous avons encadré quatre feuilles en jaune. Dans l'image IR, nous avons d'une part encadré ces quatre mêmes feuilles en rouge et d'autre part reproduit en jaune les encadrements de l'image RVB à la position qu'ils occupent dans cette dernière.

paires les plus similaires. L'objectif de cette étape est de trouver pour un élément donné dans I_1 sa position dans I_2 et réciproquement. La transformation géométrique jugée optimale est alors celle qui permet de ré-aligner spatialement au mieux ces paires d'éléments. La figure B.3 présente un exemple d'un tel recalage. Dans cette figure, les points saillants (ronds rouges) correspondant aux mêmes éléments (centre de la feuille, apex de celle-ci, aspérité dans le sol) ont des vecteurs de caractéristiques proches et sont appariés par la suite (flèches jaunes).

La deuxième famille concerne les recalages basés « sur l'intensité ». Ces méthodes recherchent la transformation géométrique qui permet de faire au mieux correspondre l'ensemble des pixels des deux images. Au départ de l'algorithme, I_1 est arbitrairement désignée comme « fixe », tandis que I_2

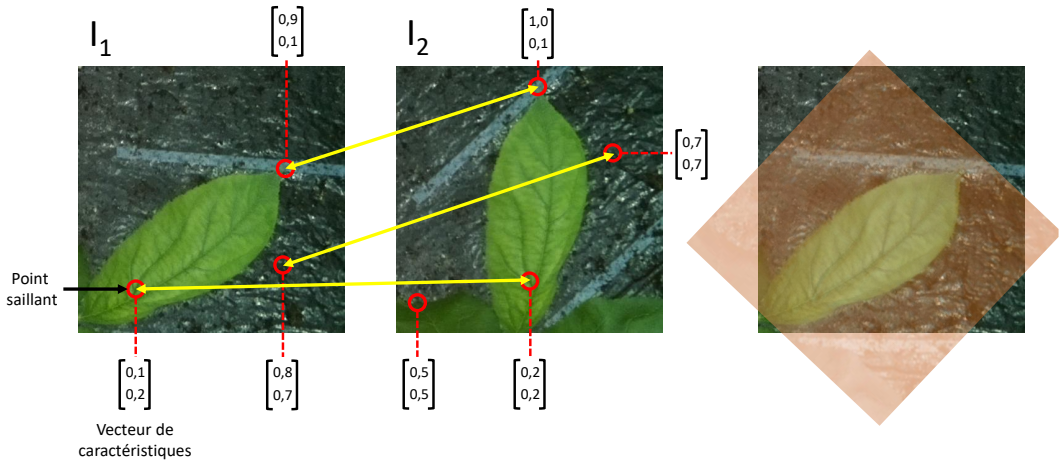


FIGURE B.3 – Illustration de l'algorithme de recalage SIFT. Le cas est illustratif : les deux images initiales sont très proches car séparées d'une simple rotation, seuls quelques points saillants sont représentés et ceux-ci ne sont décrits que par deux caractéristiques. La figure à droite présente l'alignement résultant. I_2 est représentée avec une transparence et une colorisation rouge en superposition à I_1.

est désignée comme « déformable ». Une transformation initiale est appliquée à I_2, et une métrique de similarité est calculée entre cette I_1 et I_2 ainsi transformée. Un algorithme d'optimisation est alors appliqué pour modifier la transformation afin de maximiser la métrique de similarité entre les deux images. Cette nouvelle transformation est appliquée à I_2, la métrique de similarité est calculée à nouveau, et l'algorithme se poursuit ainsi jusqu'à convergence de la métrique. La figure B.4 illustre un exemple d'un recalage par intensité.

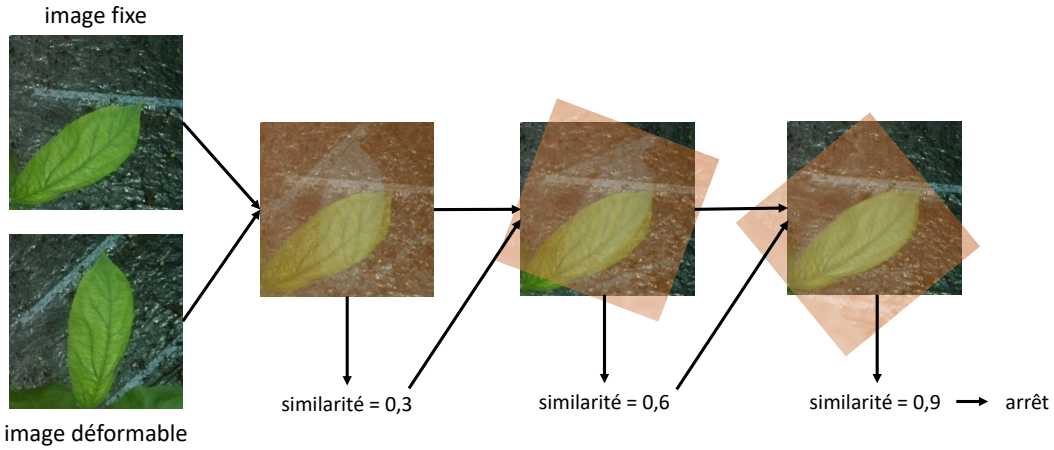


FIGURE B.4 – Illustration de l'algorithme de recalage par intensité. L'image déformable est représentée avec une transparence et une colorisation rouge en superposition à l'image fixe.

B.3 Application pour notre cas d'étude

Nous avons implémenté les deux catégories de recalage susmentionnées pour notre cas d'étude. Quelle que soit la famille choisie, il appartenait à l'utilisateur de fixer deux hyperparamètres des méthodes.

Premièrement, il était nécessaire de déterminer la catégorie de transformation géométrique recherchée (rotation, translation, etc.). Un algorithme de recalage a pour but de déterminer les paramètres de cette transformation qui sont optimaux pour aligner les deux images. Par exemple, si le type de transformation sélectionné est une rotation, l'algorithme de recalage permet de trouver l'angle de rotation optimal. Pour notre cas d'étude, nous avons choisi de fixer la transformation

recherchée comme étant une homographie. Comme expliqué au chapitre 5, l'homographie est la transformation qui existe entre deux images acquises par deux capteurs d'une même scène plane. Cette description était proche du protocole d'acquisition de $D_{décalage}$, mais pour être exact, il faut noter que les scènes acquises dans le jeu n'étaient pas planes. En effet, la distance entre l'objectif et les feuilles de pommier était variable en fonction de la hauteur d'attachement des feuilles sur les tiges des plants auxquels elles appartenaient. Pour être plus précis qu'une transformation par homographie, il aurait fallu idéalement s'intéresser à des transformations plus complexes, dites « non-rigides », capables de modéliser des déformations locales des images [Crum et al., 2004]. Cependant, ces méthodes étaient bien plus gourmandes en temps et en ressources calculatoires. De plus, le recalage n'aurait été, même avec ces méthodes, pas parfait, car l'échelonnage des feuilles sur l'axe des tiges menait à des occultations entre elles qui différaient en fonction de la modalité.

Il était par ailleurs nécessaire que l'utilisateur choisisse la métrique de similarité à maximiser. La métrique la plus standard était la corrélation pixel à pixel entre les images. En notant $d_1 \times d_2$ pixels la dimension des images, alors cette métrique se calculait comme

$$\text{corrélation} = \frac{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \mathrm{I}_1[x,y].\mathrm{I}_2[x,y]}{\sqrt{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \mathrm{I}_1[x,y]^2.\mathrm{I}_2[x,y]^2}}. \tag{6.1}$$

Ainsi, plus les pixels à la même position dans les deux images recalées avaient des valeurs proches deux à deux, plus la valeur de cette métrique était haute. Dans le cas d'images multimodales, cependant, cette mesure pouvait engendrer des résultats fallacieux. En effet, un objet donné de l'image pouvait apparaître de façon très différente d'une modalité à l'autre. Dans notre cas par exemple, les feuilles saines étaient beaucoup plus lumineuses en lumière IR qu'en lumière visible. Un recalage qui aurait aligné parfaitement les paires d'images RVB/IR aurait donc mené à une valeur faible de corrélation pixel à pixel.

Il existe d'autres métriques qui permettaient de prendre en compte au moins partiellement la multimodalité. Nous avons choisi parmi celles-ci le coefficient de corrélation amélioré (*Enchanced Correlation Coefficient* en anglais, ou ECC) [Evangelidis and Psarakis, 2008]. L'ECC de deux images, qui prenait des valeurs entre 0 et 1, se calculait ainsi :

$$\mathrm{ECC} = \frac{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \hat{\mathrm{I}_1}[x,y].\hat{\mathrm{I}_2}[x,y]}{\sqrt{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \hat{\mathrm{I}_1}[x,y]^2.\hat{\mathrm{I}_2}[x,y]^2}}, \tag{6.2}$$

où la notation $\hat{\mathrm{I}}$ signifiait

$$\frac{\mathrm{I} - \frac{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \mathrm{I}[x,y]}{d_1.d_2}}{\sqrt{\sum_{x=0}^{d_1}\sum_{y=0}^{d_2} \mathrm{I}[x,y]^2}}. \tag{6.3}$$

La normalisation des images (equation 6.3) permettait de s'affranchir partiellement des différences de luminosité causées par la multimodalité. D'autres métriques telles que l'information mutuelle [Shannon, 1948] qui prend une valeur haute lorsque tous les pixels de valeur p_1 dans I_1 sont alignés avec des pixels proches d'une valeur p_2 dans I_2 sans que les valeurs p_1 et p_2 aient de l'importance, auraient aussi été pertinentes [Wang et al., 2010].

Nous avons utilisé cette métrique comme valeur à optimiser pour le recalage par intensité, ainsi qu'en tant que métrique de performance pour évaluer la qualité du recalage à l'échelle de $D_{décalage}$ pour les deux familles de recalage. Plus précisément, nous avons désigné par « performance » la différence moyenne d'ECC entre toutes les paires de $D_{décalage}$ telles quelles d'une part, et après avoir appliqué l'algorithme de recalage sur ces paires de l'autre.

B.3.1 L'échec du recalage par caractéristiques

Nous avons implémenté un recalage par caractéristiques SIFT sur le jeu $D_{décalage}$. Ce recalage a mené à des résultats médiocres : les déformations proposées étaient aberrantes, et la performance était négative, c'est-à-dire que les paires originales étaient mieux alignées initialement qu'après l'étape de recalage. Après inspection, il s'avéra que, si de nombreux points saillants étaient bien calculés sur chacune des images, l'étape d'appariement de ces points était en échec. Un exemple d'appariement défaillant est proposé figure B.5.

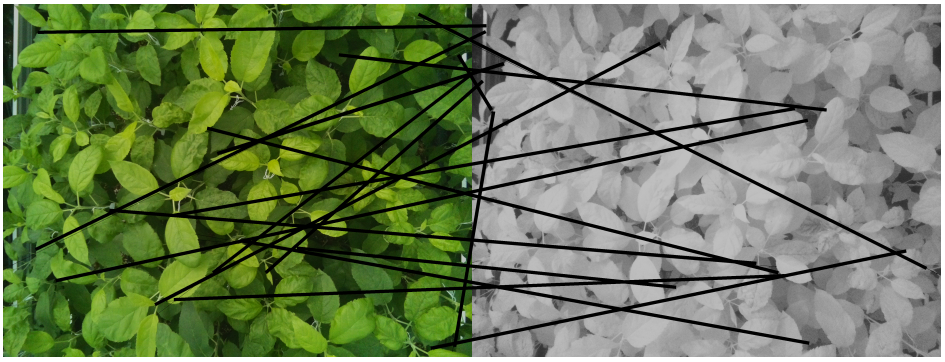


FIGURE B.5 – Appariement des caractéristiques SIFT sur une paire d'images de $D_{décalage}$. Les lignes noires tracées entre les deux images représentent chacune une paire de points saillants appariés. Ne sont représentées que les paires les plus fortement appariées, c'est-à-dire celles où la proximité entre les vecteurs descriptifs des points saillants était supérieure à une certaine valeur seuil.

Nous pouvons constater dans cette figure que les appariements étaient incorrects : la plupart des paires de points saillants ne correspondaient pas au même élément spatial. Un appariement correct aurait mené à des lignes noires à peu près parallèles entre elles. Selon nous, ce recalage par caractéristiques a échoué pour deux raisons. Premièrement, la multimodalité compliquait l'appariement de points saillants : le vecteur descriptif d'un même élément pouvait être différent selon la modalité d'acquisition. Deuxièmement, les feuilles étaient des structures très similaires les unes aux autres, ce qui pouvait facilement mener à des vecteurs descriptifs proches et donc à des appariements hasardeux.

B.3.2 Une adaptation du recalage par intensité

Nous avons par ailleurs implémenté un recalage par intensité. Nous avons désigné l'image IR comme l'image déformable, et nous avons initialisé la transformation à la transformation nulle, c'est-à-dire que l'image IR acquise telle quelle dans $D_{décalage}$ était utilisée comme image initiale. L'algorithme de descente de gradient était utilisé comme optimiseur. Ce recalage a permis d'obtenir une performance de 0,084, un résultat significativement meilleur que le recalage par caractéristiques. Cependant, nous avons constaté que pour de nombreuses images, l'algorithme de recalage convergeait en quelques itérations seulement, et que les transformations optimales renvoyées étaient alors extrêmement proches de la transformation nulle. Nous avons fait l'hypothèse que dans ces cas, l'optimiseur utilisé dans l'algorithme était rapidement pris dans un maximum local d'ECC. En effet, les images de $D_{recalage}$ comportaient de nombreuses fréquences hautes, causées non seulement par les bords des feuilles mais aussi par leur texture. Nous supposions qu'en conséquence, la forme de la fonction d'ECC selon les paramètres de l'homographie était hautement non-convexe, et donc difficile à optimiser.

En réponse à cette observation, nous avons implémenté un prétraitement sur les paires d'images afin de rendre cette fonction plus lisse. Nous avons appliqué un redimensionnement afin de réduire la taille des images, suivi d'une opération de flou gaussien. La transformation optimale était calculée sur ces images prétraitées, avant d'être appliquée aux images non prétraitées. La métrique

de similarité était alors calculée sur les images non prétraitées transformées. L'application de ce protocole, résumé figure B.6, a permis d'améliorer la performance du recalage. Celle-ci dépendait de l'intensité du redimensionnement et du floutage appliqués lors du prétraitement. Nous présentons dans le tableau B.1 la performance pour plusieurs valeurs des paramètres de ce prétraitement.

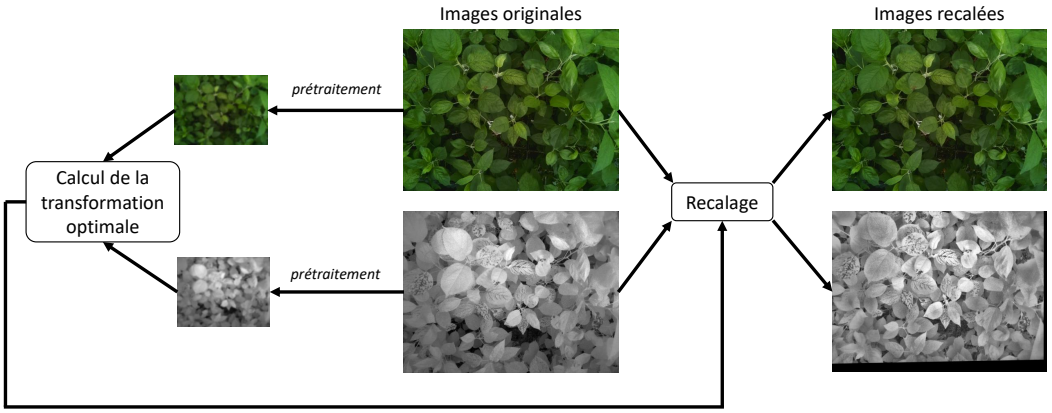


FIGURE B.6 – Illustration de l'algorithme de prétraitement que nous avons implémenté. Les images prétraitées (à gauche) servent à calculer la transformation optimale pour recaler les images originales (au milieu). La transformation calculée est appliquée aux images originales pour produire la paire d'images recalées (à droite).

	$\sigma = 0$	$\sigma = 2$	$\sigma = 4$
$r = 1$	0,084	0,094	0,100
$r = 0,5$	0,094	0,105	0,108
$r = 0,1$	0,109	0,115	**0,120**

TABLEAU B.1 – Performance de la méthode de recalage par intensité sur $D_{décalage}$. La valeur r indique l'échelle du redimensionnement appliqué à chaque dimension des images originales. La valeur σ indique l'écart-type de la fonction gaussienne utilisée dans l'opération de floutage.

Nous pouvons constater que les redimensionnements les plus drastiques et les floutages les plus intenses parmi la gamme essayée ont permis d'obtenir les meilleurs alignements. La figure B.7 illustre la qualité de l'alignement obtenu avec cette méthode.

FIGURE B.7 – Paire d'images de $D_{décalage}$ présentée figure B.2, alignée via un recalage par intensité avec le prétraitement proposé aux paramètres optimaux. Les encadrements ont la même signification que dans la figure B.2.

La figure B.8 présente un exemple de résultat de l'application du prétraitement avec les paramètres optimaux. Nous pouvons constater que la plupart des détails des images étaient effacés,

et que les seules informations de haute fréquence restantes provenaient des bords des feuilles particulièrement marqués, qui étaient des caractéristiques robustes pour un recalage multimodal. En effet, les détails estompés par le prétraitement pouvaient même être trompeurs : les structures internes des feuilles pouvaient correspondre à des taches de tavelure dans les images IR et à des nervures dans les images RVB par exemple. En d'autres termes, le prétraitement que nous avons proposé permettait de contrôler l'échelle de détail auquel le recalage s'effectuait.

FIGURE B.8 – Paire d'images présentée dans la figure B.2, après application du prétraitement décrit avec $d = 0,1$ et $\sigma = 4$.

B.4 Conclusion

Nous avons proposé dans cette annexe un protocole de recalage basé sur le recalage par intensité et adapté aux scène auto-similaires et aux nombreuses fréquences hautes comme les images de canopée. Nous estimons que grâce au recalage effectué ainsi, les deux modalités étaient suffisamment alignées pour qu'un apprentissage par CNN puisse être mené sur les informations conjointes RVB et IR. Cette fusion a montré son potentiel dans des conditions simulées pour une détection optimale de lésions de tavelure (tableau 6.1).

Références

Abade, A. S., Ferreira, P. A., and Vidal, F. d. B. (2020). Plant diseases recognition on images using convolutional neural networks : A systematic review. *arXiv preprint arXiv :2009.04365*.

Abdelghafour, F., Keresztes, B., Germain, C., and Costa, J.-P. D. (2020). In field detection of downy mildew symptoms with proximal colour imaging. *Sensors*, 20(16) :4380.

Adler, A., Elad, M., and Zibulevsky, M. (2016). Compressed learning : A deep neural network approach. *arXiv preprint arXiv :1610.09615*.

Al Hussani, M. T. and Al Hayani, M. H. A. (2014). The use of filtered back projection algorithm for reconstruction of tomographic image. *Al-Nahrain Journal for Engineering Sciences*, 17(2) :151–156.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esesn, B. C., Awwal, A. A. S., and Asari, V. K. (2018). The history began from alexnet : A comprehensive survey on deep learning approaches. *arXiv preprint arXiv :1803.01164*.

Andor, O. I. (2020). How to define the quantum efficiency of ccd cameras. https://andor.oxinst.com/learning/view/article/ccd-spectral-response-(qe). Accédé : 2020-11-13.

Arce, G. R., Brady, D. J., Carin, L., Arguello, H., and Kittle, D. S. (2013). Compressive coded aperture spectral imaging : An introduction. *IEEE Signal Processing Magazine*, 31(1) :105–115.

Arivazhagan, S., Shebiah, R. N., Ananthi, S., and Varthini, S. V. (2013). Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. *Agricultural Engineering International : CIGR Journal*, 15(1) :211–217.

Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv :1701.04862*.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223.

Arsenovic, M., Karanovic, M., Sladojevic, S., Anderla, A., and Stefanovic, D. (2019). Solving current limitations of deep learning based approaches for plant disease detection. *Symmetry*, 11(7) :939.

Bacca, J., Galvis, L., and Arguello, H. (2020). Coupled deep learning coded aperture design for compressive image classification. *Optics Express*, 28(6) :8528–8540.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet : A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12) :2481–2495.

Baltrušaitis, T., Ahuja, C., and Morency, L.-P. (2018). Multimodal machine learning : A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2) :423–443.

Bannari, A., Morin, D., Bonn, F., and Huete, A. (1995). A review of vegetation indices. *Remote sensing reviews*, 13(1-2) :95–120.

Barth, R., IJsselmuiden, J., Hemming, J., and Van Henten, E. J. (2018). Data synthesis methods for semantic segmentation in agriculture : A capsicum annuum dataset. *Computers and electronics in agriculture*, 144 :284–296.

Bayer, B. E. (1976). Color imaging array. US Patent 3,971,065.

Behmann, J., Steinrücken, J., and Plümer, L. (2014). Detection of early plant stress responses in hyperspectral images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 93 :98–111.

Belin, É., Rousseau, D., Boureau, T., and Caffier, V. (2013). Thermography versus chlorophyll fluorescence imaging for detection and quantification of apple scab. *Computers and electronics in agriculture*, 90 :159–163.

Ben Hamza, A., He, Y., Krim, H., and Willsky, A. (2005). A multiscale approach to pixel-level image fusion. *Integrated Computer-Aided Engineering*, 12(2) :135–146.

Benoit, L., Benoit, R., Belin, É., Vadaine, R., Demilly, D., Chapeau-Blondeau, F., and Rousseau, D. (2016). On the value of the kullback–leibler divergence for cost-effective spectral imaging of plants by optimal selection of wavebands. *Machine Vision and Applications*, 27(5) :625–635.

Benoit, L., Rousseau, D., Belin, É., Demilly, D., and Chapeau-Blondeau, F. (2014). Simulation of image acquisition in machine vision dedicated to seedling elongation to validate image processing root segmentation algorithms. *Computers and electronics in agriculture*, 104 :84–92.

Blackburn, G. A. (2007). Hyperspectral remote sensing of plant pigments. *Journal of experimental botany*, 58(4) :855–867.

Boas, F. E. and Fleischmann, D. (2012). Ct artifacts : causes and reduction techniques. *Imaging in medicine*, 4(2) :229–240.

Borràs, E., Ferré, J., Boqué, R., Mestres, M., Aceña, L., and Busto, O. (2015). Data fusion methodologies for food and beverage authentication and quality assessment–a review. *Analytica Chimica Acta*, 891 :1–14.

Bowen, J. K., Mesarich, C. H., Bus, V. G., Beresford, R. M., Plummer, K. M., and Templeton, M. D. (2011). Venturia inaequalis : the causal agent of apple scab. *Molecular Plant Pathology*, 12(2) :105–122.

Bracewell, R. N. (1956). Strip integration in radio astronomy. *Australian Journal of Physics*, 9(2) :198–217.

Branwen, G. (2019). This waifu does not exist. https://www.thiswaifudoesnotexist.net/. Accédé le 2021-02-03.

Brauers, J., Schulte, N., and Aach, T. (2008). Multispectral filter-wheel cameras : Geometric distortion model and compensation algorithms. *IEEE transactions on image processing*, 17(12) :2368–2380.

Bravo, C., Moshou, D., West, J., McCartney, A., and Ramon, H. (2003). Early disease detection in wheat fields using spectral reflectance. *Biosystems Engineering*, 84(2) :137–145.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv :2005.14165*.

Buddenbaum, H., Stern, O., Paschmionka, B., Hass, E., Gattung, T., Stoffels, J., Hill, J., and Werner, W. (2015). Using vnir and swir field imaging spectroscopy for drought stress monitoring of beech seedlings. *International Journal of Remote Sensing*, 36(18) :4590–4605.

Bulygin, T. V. and Vishnyakov, G. N. (1992). Spectrotomography : a new method of obtaining spectrograms of two-dimensional objects. In *Analytical Methods for Optical Tomography*, volume 1843, pages 315–322.

Busemeyer, L., Mentrup, D., Möller, K., Wunder, E., Alheit, K., Hahn, V., Maurer, H. P., Reif, J. C., Würschum, T., Müller, J., et al. (2013). Breedvision—a multi-sensor platform for non-destructive field-based phenotyping in plant breeding. *Sensors*, 13(3) :2830–2847.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., and Kalinin, A. A. (2020). Albumentations : fast and flexible image augmentations. *Information*, 11(2) :125.

Calderbank, R., Jafarpour, S., and Schapire, R. (2009). Compressed learning : Universal sparse dimensionality reduction and learning in the measurement domain. *Penn State University Report*.

Camino, C., González-Dugo, V., Hernández, P., Sillero, J., and Zarco-Tejada, P. J. (2018). Improved nitrogen retrievals with airborne-derived fluorescence and plant traits quantified from vnir-swir hyperspectral imagery in the context of precision agriculture. *International journal of applied earth observation and geoinformation*, 70 :105–117.

Candes, E. J., Romberg, J. K., and Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8) :1207–1223.

Castro, F. M., Marín-Jiménez, M. J., Guil, N., and de la Blanca, N. P. (2020). Multimodal feature fusion for cnn-based gait recognition : an empirical comparison. *Neural Computing and Applications*, pages 1–21.

Cerutti, G., Tougne, L., Mille, J., Vacavant, A., and Coquin, D. (2013). Understanding leaves in natural images–a model-based approach for tree species identification. *Computer Vision and Image Understanding*, 117(10) :1482–1501.

Chaerle, L., De Boever, F., Montagu, M. V., and Straeten, D. V. D. (2001). Thermographic visualization of cell death in tobacco and arabidopsis. *Plant, Cell & Environment*, 24(1) :15–25.

Chaerle, L., Van Caeneghem, W., Messens, E., Lambers, H., Van Montagu, M., and Van Der Straeten, D. (1999). Presymptomatic visualization of plant–virus interactions by thermography. *Nature biotechnology*, 17(8) :813–816.

Chaerle, L. and Van Der Straeten, D. (2000). Imaging techniques and the early detection of plant stress. *Trends in plant science*, 5(11) :495–501.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). Shapenet : An information-rich 3d model repository. *arXiv preprint arXiv :1512.03012*.

Chatila, R. and Laumond, J.-P. (1985). Position referencing and consistent world modeling for mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 138–145.

Chawla, N. V. (2009). Data mining for imbalanced datasets : An overview. In *Data mining and knowledge discovery handbook*, pages 875–886.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote : synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16 :321–357.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017). Deeplab : Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4) :834–848.

Chen, Y., Jiang, H., Li, C., Jia, X., and Ghamisi, P. (2016). Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10) :6232–6251.

Chintala, S., Denton, E., Arjovsky, M., and Mathieu, M. (2016). How to train a gan ? tips and tricks to make gans work. https://github.com/soumith/ganhacks. Accédé le 2021-03-15.

Chollet, F. (2017). Xception : Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258.

Coffey, V. C. (2012). Multispectral imaging moves into the mainstream. *Optics and Photonics News*, 23(4) :18–24.

Cooper, J. and Dobson, H. (2007). The benefits of pesticides to mankind and the environment. *Crop Protection*, 26(9) :1337–1348.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1) :21–27.

Crum, W. R., Hartkens, T., and Hill, D. (2004). Non-rigid image registration : theory and practice. *The British journal of radiology*, 77(S2) :S140–S153.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2018). Autoaugment : Learning augmentation policies from data. *arXiv preprint arXiv :1805.09501*.

Cuthbertson, A. and Murchie, A. (2003). The impact of fungicides to control apple scab (Venturia inaequalis) on the predatory mite Anystis baccarum and its prey Aculus schlechtendali (apple rust mite) in Northern Ireland Bramley orchards. *Crop Protection*, 22(9) :1125–1130.

Daughtry, C. S., Walthall, C., Kim, M., De Colstoun, E. B., and McMurtrey Iii, J. (2000). Estimating corn leaf chlorophyll concentration from leaf and canopy reflectance. *Remote sensing of Environment*, 74(2) :229–239.

De Man, Q., Haneda, E., Claus, B., Fitzgerald, P., De Man, B., Qian, G., Shan, H., Min, J., Sabuncu, M., and Wang, G. (2019). A two-dimensional feasibility study of deep learning-based feature detection and characterization directly from ct sinograms. *Medical Physics*, 46(12) :e790–e800.

Delalieux, S., Auwerkerken, A., Verstraeten, W. W., Somers, B., Valcke, R., Lhermitte, S., Keulemans, J., and Coppin, P. (2009a). Hyperspectral reflectance and fluorescence imaging to detect scab induced stress in apple leaves. *Remote sensing*, 1(4) :858–874.

Delalieux, S., Somers, B., Verstraeten, W., Van Aardt, J., Keulemans, W., and Coppin, P. (2009b). Hyperspectral indices to diagnose leaf biotic stress of apple plants, considering leaf phenology. *International journal of remote sensing*, 30(8) :1887–1912.

Descour, M. and Dereniak, E. (1995). Computed-tomography imaging spectrometer : experimental calibration and reconstruction results. *Applied optics*, 34(22) :4817–4826.

Descour, M. R. (1994). *Non-scanning imaging spectrometry*. PhD thesis, The University of Arizona.

Descour, M. R., Dereniak, E. L., and Dubey, A. C. (1995). Mine detection using instantaneous spectral imaging. In *Detection Technologies for Mines and Minelike Targets*, volume 2496, pages 286–304. International Society for Optics and Photonics.

Descour, M. R., Volin, C. E., Dereniak, E. L., Thome, K. J., Schumacher, A., Wilson, D. W., and Maker, P. D. (1997). Demonstration of a high-speed nonscanning imaging spectrometer. *Optics letters*, 22(16) :1271–1273.

Di Cicco, M., Potena, C., Grisetti, G., and Pretto, A. (2017). Automatic model based dataset generation for fast and accurate crop and weeds detection. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5188–5195.

Dong, J., Wang, L., Liu, J., Gao, Y., Qi, L., and Sun, X. (2019). A procedural texture generation framework based on semantic descriptions. *Knowledge-Based Systems*, 163 :898–906.

Donoho, D. L. et al. (2000). High-dimensional data analysis : The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000) :32.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). Carla : An open urban driving simulator. In *Conference on robot learning*, pages 1–16.

Douarre, C., Crispim-Junior, C., Gelibert, A., Rousseau, D., and Tougne, L. (2019a). A strategy for multimodal canopy images registration. In *7th International Workshop on Image Analysis Methods in the Plant Sciences*.

Douarre, C., Crispim-Junior, C., Gelibert, A., Tougne, L., and Rousseau, D. (2019b). When spectro-imaging meets machine learning. In *Workshop on Machine Learning Assisted Image Formation*.

Douarre, C., Crispim-Junior, C. F., Gelibert, A., Tougne, L., and Rousseau, D. (2019c). Novel data augmentation strategies to boost supervised segmentation of plant disease. *Computers and electronics in agriculture*, 165 :104967.

Douarre, C., Crispim-Junior, C. F., Gelibert, A., Tougne, L., and Rousseau, D. (2020a). On the value of ctis imagery for neural-network-based classification : a simulation perspective. *Applied optics*, 59(28) :8697–8710.

Douarre, C., Schielein, R., Frindel, C., Gerth, S., and Rousseau, D. (2018a). Transfer learning from synthetic data applied to soil–root segmentation in x-ray tomography images. *Journal of Imaging*, 4(5) :65.

Douarre, C., Tougne, L., Crispim-Junior, C., Gelibert, A., and Rousseau, D. (2018b). Data simulation to improve supervised segmentation of apple scab images. *Les Rencontres du Végétal, 10eme edition*.

Douarre, C., Tougne, L., Crispim-Junior, C., Gelibert, A., and Rousseau, D. (2020b). Apprentissage comprimé sur images hyperspectrales de feuilles de pommier atteintes de tavelure. In *Réunion du Groupement de Recherche Information, Signal, Images et Vision, journée à thème Intelligence artificielle / Apprentissage machine pour l'agriculture*.

Duda, R. O. and Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1) :11–15.

Dumoulin, V. and Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv :1603.07285*.

Efros, A. A. and Freeman, W. T. (2001). Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346.

Eitel, A., Springenberg, J. T., Spinello, L., Riedmiller, M., and Burgard, W. (2015). Multimodal deep learning for robust rgb-d object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 681–687.

Elvidge, C. D. (1990). Visible and near infrared reflectance characteristics of dry plant materials. *Remote Sensing*, 11(10) :1775–1795.

Evangelidis, G. D. and Psarakis, E. Z. (2008). Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10) :1858–1865.

Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145 :311–318.

Ferraris, V., Dobigeon, N., Wei, Q., and Chabert, M. (2017). Detecting changes between optical images of different spatial and spectral resolutions : a fusion-based approach. *IEEE Transactions on Geoscience and Remote Sensing*, 56(3) :1566–1578.

Fontenla-Romero, Ó., Guijarro-Berdiñas, B., Martinez-Rego, D., Pérez-Sánchez, B., and Peteiro-Barral, D. (2013). Online machine learning. In *Efficiency and Scalability Methods for Computational Intellect*, pages 27–54.

Ford, B. K., Descour, M. R., and Lynch, R. M. (2001a). Large-image-format computed tomography imaging spectrometer for fluorescence microscopy. *Optics Express*, 9(9) :444–453.

Ford, B. K., Volin, C. E., Murphy, S. M., Lynch, R. M., and Descour, M. R. (2001b). Computed tomography-based spectral imaging for fluorescence microscopy. *Biophysical Journal*, 80(2) :986–993.

Fowler, K. R. (2004). Automatic gain control for image-intensified camera. *IEEE Transactions on Instrumentation and Measurement*, 53(4) :1057–1064.

Gao, L., Kester, R. T., Hagen, N., and Tkaczyk, T. S. (2010). Snapshot image mapping spectrometer (ims) with high sampling density for hyperspectral microscopy. *Optics express*, 18(14) :14330–14344.

Gao, Y., Tan, J., Liang, Z., Li, L., and Huo, Y. (2019). Improved computer-aided detection of pulmonary nodules via deep learning in the sinogram domain. *Visual Computing for Industry, Biomedicine, and Art*, 2(1) :1–9.

Garcia, J. P. and Dereniak, E. L. (1999). Mixed-expectation image-reconstruction technique. *Applied optics*, 38(17) :3745–3748.

Gat, N. (2000). Imaging spectroscopy using tunable filters : a review. In *Wavelet Applications VII*, volume 4056, pages 50–64. International Society for Optics and Photonics.

Gates, D. M. (2012). *Biophysical ecology*. Courier Corporation.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423.

Germain, G. (2019). Dispositif de capture d'une image hyperspectrale. Brevet FR3071124.

Gerstner, W. and Kistler, W. M. (2002). *Spiking neuron models : Single neurons, populations, plasticity*. Cambridge university press.

Ghassemian, H. (2016). A review of remote sensing image fusion methods. *Information Fusion*, 32 :75–89.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323.

Godin, C. and Caraglio, Y. (1998). A multiscale model of plant topological structures. *Journal of theoretical biology*, 191(1) :1–46.

Golhani, K., Balasundram, S. K., Vadamalai, G., and Pradhan, B. (2018). A review of neural networks in plant disease detection using hyperspectral data. *Information Processing in Agriculture*, 5(3) :354–371.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Gordon, R., Bender, R., and Herman, G. T. (1970). Algebraic reconstruction techniques (art) for three-dimensional electron microscopy and x-ray photography. *Journal of theoretical Biology*, 29(3) :471–481.

Grasset, L. (2020). Dirty biology : comment créer une couleur? https://www.youtube.com/watch?v=wCMGxXgypS4. Accédé : 2021-03-12.

Green, R. O., Eastwood, M. L., Sarture, C. M., Chrien, T. G., Aronsson, M., Chippendale, B. J., Faust, J. A., Pavri, B. E., Chovit, C. J., Solis, M., et al. (1998). Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (aviris). *Remote sensing of environment*, 65(3) :227–248.

Gunes, H. and Piccardi, M. (2005). Affect recognition from face and body : early fusion vs. late fusion. In *2005 IEEE international conference on systems, man and cybernetics*, volume 4, pages 3437–3443.

Gupta, A., Vedaldi, A., and Zisserman, A. (2016). Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324.

Habel, R., Kudenov, M., and Wimmer, M. (2012). Practical spectral photography. In *Computer graphics forum*, volume 31, pages 449–458.

Hagen, N. and Dereniak, E. L. (2007). New grating designs for a ctis imaging spectrometer. In *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XIII*, volume 6565, page 65650N.

Hagen, N. and Dereniak, E. L. (2008). Analysis of computed tomographic imaging spectrometers. i. spatial and spectral resolution. *Applied Optics*, 47(28) :F85–F95.

Hagen, N., Dereniak, E. L., and Sass, D. T. (2006). Maximizing the resolution of a ctis instrument. In *Imaging Spectrometry XI*, volume 6302, page 63020L.

Hagen, N., Dereniak, E. L., and Sass, D. T. (2007). Fourier methods of improving reconstruction speed for ctis imaging spectrometers. In *Imaging Spectrometry XII*, volume 6661, page 666103.

Hagen, N. A., Gao, L. S., Tkaczyk, T. S., and Kester, R. T. (2012). Snapshot advantage : a review of the light collection improvement for parallel high-dimensional measurement systems. *Optical Engineering*, 51(11) :111702.

Hagen, N. A. and Kudenov, M. W. (2013). Review of snapshot spectral imaging technologies. *Optical Engineering*, 52(9) :090901.

Hammernik, K., Würfl, T., Pock, T., and Maier, A. (2017). A deep learning architecture for limited-angle computed tomography reconstruction. In *Bildverarbeitung für die Medizin 2017*, pages 92–97. Springer.

Handa, A., Patraucean, V., Badrinarayanan, V., Stent, S., and Cipolla, R. (2016). Understanding real world indoor scenes with synthetic data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4077–4085.

Haralick, R. M., Shanmugam, K., and Dinstein, I. H. (1973). Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6) :610–621.

Haskins, G., Kruger, U., and Yan, P. (2020). Deep learning in medical image registration : a survey. *Machine Vision and Applications*, 31(1) :1–18.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers : Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

He, K. S., Rocchini, D., Neteler, M., and Nagendra, H. (2011). Benefits of hyperspectral remote sensing for tracking plant invasions. *Diversity and Distributions*, 17(3) :381–392.

He, X., Zhao, K., and Chu, X. (2021). Automl : A survey of the state-of-the-art. *Knowledge-Based Systems*, 212 :106622.

Hege, E. K., O'Connell, D., Johnson, W., Basty, S., and Dereniak, E. L. (2004). Hyperspectral imaging for astronomy and space surveillance. In *Imaging Spectrometry IX*, volume 5159, pages 380–391.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30 :6626–6637.

Holschneider, M., Kronland-Martinet, R., Morlet, J., and Tchamitchian, P. (1990). A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297.

Hornik, K., Stinchcombe, M., White, H., et al. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5) :359–366.

Howarth, P. and Rüger, S. (2004). Evaluation of texture features for content-based image retrieval. In *International conference on image and video retrieval*, pages 326–334.

Hu, B., Levesque, J., and Ardouin, J.-P. (2008). Vegetation species identification using hyperspectral imagery. In *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, volume 2, pages II–299.

Hu, W., Huang, Y., Wei, L., Zhang, F., and Li, H. (2015). Deep Convolutional Neural Networks for Hyperspectral Image Classification. *Journal of Sensors*, 2015 :1–12.

Huang, X., Xin, J., and Zhao, J. (2011). A novel technique for rapid evaluation of fish freshness using colorimetric sensor array. *Journal of Food Engineering*, 105(4) :632–637.

Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3) :574.

Hughes, D., Salathé, M., et al. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv :1511.08060*.

Huygens, C. (1920). *Traité de la lumière*. Éditions Dunod.

Hwang, S., Park, J., Kim, N., Choi, Y., and So Kweon, I. (2015). Multispectral pedestrian detection : Benchmark dataset and baseline. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1037–1045.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet : Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv :1602.07360*.

Isokane, T., Okura, F., Ide, A., Matsushita, Y., and Yagi, Y. (2018). Probabilistic plant modeling via multi-view image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2906–2915.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134.

Jacquemoud, S. and Baret, F. (1990). Prospect : A model of leaf optical properties spectra. *Remote sensing of environment*, 34(2) :75–91.

Jain, R., Kasturi, R., and Schunck, B. G. (1995). *Machine vision*, volume 5. McGraw-hill New York.

Jancovici, J.-M. (2019). Co2 ou pib, il faut choisir. https://www.youtube.com/watch?v=Vjkq8V5rVy0. Conférence à Sciences Po. Accédé le 2021-02-03.

Jancovici, J.-M. (2020). Transition énergétique : pourquoi dit-on depuis 40 ans qu'il y a 40 ans de pétrole? https://jancovici.com/transition-energetique/petrole/pourquoi-dit-on-depuis-40-ans-quil-y-a-40-ans-de-petrole/. Accédé : 2020-11-13.

Jing, L., Wang, T., Zhao, M., and Wang, P. (2017). An adaptive multi-sensor data fusion method based on deep convolutional neural networks for fault diagnosis of planetary gearbox. *Sensors*, 17(2) :414.

Johnson, W. R., Wilson, D. W., and Bearman, G. (2006). Spatial-spectral modulating snapshot hyperspectral imager. *Applied optics*, 45(9) :1898–1908.

Johnson, W. R., Wilson, D. W., Fink, W., Humayun, M. S., and Bearman, G. H. (2007). Snapshot hyperspectral imaging in ophthalmology. *Journal of biomedical optics*, 12(1) :014036.

Jones, H. G. (2004). Application of thermal imaging and infrared sensing in plant physiology and ecophysiology. In *Advances in Botanical Research*, volume 41, pages 107–163.

Karpathy, A. (2020). Ai for full-self driving at tesla. https://www.youtube.com/watch?v=hx7BXih7zx8. 5th Annual Scaled Machine Learning Conference 2020. Accédé le 2021-02-03.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732.

Katsaggelos, A. K., Bahaadini, S., and Molina, R. (2015). Audiovisual fusion : Challenges and new approaches. *Proceedings of the IEEE*, 103(9) :1635–1653.

Kaur, S., Pandey, S., and Goel, S. (2019). Plants disease identification and classification through leaf images : A survey. *Archives of Computational Methods in Engineering*, 26(2) :507–530.

Kerkech, M., Hafiane, A., and Canals, R. (2020). Vddnet : Vine disease detection network based on multispectral images and depth map. *Remote Sensing*, 12(20) :3305.

Khirade, S. D. and Patil, A. (2015). Plant disease detection using image processing. In *2015 International conference on computing communication control and automation*, pages 768–771.

Kiefer, J., Wolfowitz, J., et al. (1952). Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3) :462–466.

Kim, D. M., Zhang, H., Zhou, H., Du, T., Wu, Q., Mockler, T. C., and Berezin, M. Y. (2015). Highly sensitive image-derived indices of water-stressed plants using hyperspectral imaging in swir and histogram analysis. *Scientific reports*, 5(1) :1–11.

Kim, Y., Glenn, D. M., Park, J., Ngugi, H. K., and Lehman, B. L. (2011). Hyperspectral image analysis for water stress detection of apple trees. *Computers and Electronics in Agriculture*, 77(2) :155–160.

Kim, Y. M., Theobalt, C., Diebel, J., Kosecka, J., Miscusik, B., and Thrun, S. (2009). Multi-view image and tof sensor fusion for dense 3d reconstruction. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pages 1542–1549.

Kocić, J., Jovičić, N., and Drndarević, V. (2018). Sensors and sensor fusion in autonomous vehicles. In *2018 26th Telecommunications Forum (TELFOR)*, pages 420–425.

Kopf, J., Fu, C.-W., Cohen-Or, D., Deussen, O., Lischinski, D., and Wong, T.-T. (2007). Solid texture synthesis from 2D exemplars. *ACM Transactions on Graphics*, 26(3) :2.

Korel, F., Luzuriaga, D., and Balaban, M. (2001). Objective quality assessment of raw tilapia (oreochromis niloticus) fillets using electronic nose and machine vision. *Journal of Food Science*, 66(7) :1018–1024.

Korshunov, P. and Marcel, S. (2018). Deepfakes : a new threat to face recognition ? assessment and detection. *arXiv preprint arXiv :1812.08685*.

Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in neural information processing systems*, 24 :109–117.

Kress, T. M. (2017). The dark side of the prism. In *Spinning Popular Culture as Public Pedagogy*, pages 63–77. Brill Sense.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems*.

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I., and Soares, J. V. B. (2012). Leafsnap : A computer vision system for automatic plant species identification. In *The 12th European Conference on Computer Vision*.

Kümmerlen, B., Dauwe, S., Schmundt, D., and Schurr, U. (1999). Thermography to measure water relations of plant leaves. *Handbook of computer vision and applications*, 3 :763–781.

Lavania, S. and Matey, P. S. (2014). Leaf recognition using contour based edge detection and sift algorithm. In *2014 IEEE International Conference on Computational Intelligence and Computing Research*, pages 1–4. IEEE.

LeCun, Y. (1998). The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/. Accédé : 2020-11-13.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4) :541–551.

Lee, A. B., Mumford, D., and Huang, J. (2001). Occlusion models for natural images : A statistical study of a scale-invariant dead leaves model. *International Journal of Computer Vision*, 41(1-2) :35–59.

Lee, H., Huang, C., Yune, S., Tajmir, S. H., Kim, M., and Do, S. (2019). Machine friendly machine learning : interpretation of computed tomography without image reconstruction. *Scientific reports*, 9(1) :1–9.

Lee, N., Wielaard, J., Fawzi, A., Sajda, P., Laine, A., Martin, G., Humayun, M., and Smith, R. (2010). In vivo snapshot hyperspectral image analysis of age-related macular degeneration. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 5363–5366.

Lehmann, J. R. K., Große-Stoltenberg, A., Römer, M., and Oldeland, J. (2015). Field spectroscopy in the vnir-swir region to discriminate between mediterranean native plants and exotic-invasive shrubs based on leaf tannin content. *Remote Sensing*, 7(2) :1225–1241.

Li, L., Zhang, Q., and Huang, D. (2014). A review of imaging techniques for plant phenotyping. *Sensors*, 14(11) :20078–20111.

Li, S., Kang, X., Fang, L., Hu, J., and Yin, H. (2017a). Pixel-level image fusion : A survey of the state of the art. *information Fusion*, 33 :100–112.

Li, Y., Zhang, H., and Shen, Q. (2017b). Spectral–spatial classification of hyperspectral imagery with 3d convolutional neural network. *Remote Sensing*, 9(1) :67.

Liang, H. and Li, Q. (2016). Hyperspectral imagery classification using sparse representations of convolutional neural network features. *Remote Sensing*, 8(2) :99.

Liao, W., Huang, X., Van Coillie, F., Gautama, S., Pižurica, A., Philips, W., Liu, H., Zhu, T., Shimoni, M., Moser, G., et al. (2015). Processing of multiresolution thermal hyperspectral and digital color data : Outcome of the 2014 ieee grss data fusion contest. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6) :2984–2996.

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development ii. simple and branching filaments with two-sided inputs. *Journal of theoretical biology*, 18(3) :300–315.

Lohit, S., Kulkarni, K., and Turaga, P. (2016). Direct inference on compressive measurements using convolutional neural networks. In *2016 IEEE International Conference on Image Processing*, pages 1913–1917.

Loncan, L., De Almeida, L. B., Bioucas-Dias, J. M., Briottet, X., Chanussot, J., Dobigeon, N., Fabre, S., Liao, W., Licciardi, G. A., Simoes, M., et al. (2015). Hyperspectral pansharpening : A review. *IEEE Geoscience and remote sensing magazine*, 3(3) :27–46.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

Louargant, M., Jones, G., Faroux, R., Paoli, J.-N., Maillot, T., Gée, C., and Villette, S. (2018). Unsupervised classification algorithm for early weed detection in row-crops by combining spatial and spectral information. *Remote Sensing*, 10(5) :761.

Louargant, M., Villette, S., Jones, G., Vigneau, N., Paoli, J.-N., and Gée, C. (2017). Weed detection by uav : simulation of the impact of spectral mixing in multispectral images. *Precision Agriculture*, 18(6) :932–951.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2) :91–110.

Ma, J., Ma, Y., and Li, C. (2019). Infrared and visible image fusion methods and applications : A survey. *Information Fusion*, 45 :153–178.

Mahlein, A.-K. (2016). Plant disease detection by imaging sensors–parallels and specific demands for precision agriculture and plant phenotyping. *Plant disease*, 100(2) :241–251.

Mahlein, A.-K., Steiner, U., Hillnhütter, C., Dehne, H.-W., and Oerke, E.-C. (2012). Hyperspectral imaging for small-scale analysis of symptoms caused by different sugar beet diseases. *Plant methods*, 8(1) :3.

Makantasis, K., Karantzalos, K., Doulamis, A., and Doulamis, N. (2015). Deep supervised learning for hyperspectral data classification through convolutional neural networks. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4959–4962.

Martinelli, F., Scalenghe, R., Davino, S., Panno, S., Scuderi, G., Ruisi, P., Villa, P., Stroppiana, D., Boschetti, M., Goulart, L. R., et al. (2015). Advanced methods of plant disease detection. a review. *Agronomy for Sustainable Development*, 35(1) :1–25.

Masi, G., Cozzolino, D., Verdoliva, L., and Scarpa, G. (2016). Pansharpening by convolutional neural networks. *Remote Sensing*, 8(7) :594.

Mathews, S. A. (2008). Design and fabrication of a low-cost, multispectral imaging system. *Applied optics*, 47(28) :F71–F76.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2) :442–451.

Meishvili, G., Jenni, S., and Favaro, P. (2020). Learning to have an ear for face super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1364–1374.

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv :1411.1784*.

Mohanty, S. P., Hughes, D. P., and Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7 :1419.

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932.

Nagasubramanian, K., Jubery, T. Z., Ardakani, F. F., Mirnezami, S. V., Singh, A. K., Singh, A., Sarkar, S., and Ganapathysubramanian, B. (2020). How useful is active learning for image-based plant phenotyping? *arXiv preprint arXiv :2006.04255*.

Nazki, H., Yoon, S., Fuentes, A., and Park, D. S. (2020). Unsupervised image translation using adversarial networks for improved plant disease recognition. *Computers and Electronics in Agriculture*, 168 :105117.

Neff, T., Payer, C., Stern, D., and Urschler, M. (2017). Generative adversarial network based synthesis for supervised medical image segmentation. In *Proc. OAGM and ARW Joint Workshop*.

Nikolenko, S. (2019). Synthetic data in deep learning. In *School-conference "Approximation and Data Analysis 2019"*, page 21.

Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528.

Odena, A., Dumoulin, V., and Olah, C. (2016). Deconvolution and checkerboard artifacts. *Distill*, 1(10) :e3.

Oerke, E.-C., Fröhling, P., and Steiner, U. (2011). Thermographic assessment of scab disease on apple leaves. *Precision agriculture*, 12(5) :699–715.

Okamoto, T. and Yamaguchi, I. (1991). Simultaneous acquisition of spectral image information. *Optics letters*, 16(16) :1277–1279.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1) :62–66.

Pagnutti, M. A., Ryan, R. E., Cazenavette, G. J., Gold, M. J., Harlan, R., Leggett, E., and Pagnutti, J. F. (2017). Laying the foundation to use raspberry pi 3 v2 camera module imagery for scientific and engineering purposes. *Journal of Electronic Imaging*, 26(1) :013014.

Paisitkriangkrai, S., Sherrah, J., Janney, P., Hengel, V.-D., et al. (2015). Effective semantic pixel labelling with convolutional networks and conditional random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 36–43.

Pan, X., Sidky, E. Y., and Vannier, M. (2009). Why do commercial ct scanners still employ traditional, filtered back-projection for image reconstruction ? *Inverse problems*, 25(12) :123009.

Paoletti, M., Haut, J., Plaza, J., and Plaza, A. (2018). A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS journal of photogrammetry and remote sensing*, 145 :120–147.

Papadopoulos, D. P., Clarke, A. D., Keller, F., and Ferrari, V. (2014). Training object class detectors from eye tracking data. In *European conference on computer vision*, pages 361–376.

Pawara, P., Okafor, E., Schomaker, L., and Wiering, M. (2017). Data augmentation for plant classification. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 615–626.

Pennazio, S. and Sapetti, C. (1982). Electrolyte leakage in relation to viral and abiotic stresses inducing necrosis in cowpea leaves. *Biologia plantarum*, 24(3) :218–225.

Peñuelas, J. and Filella, I. (1998). Visible and near-infrared reflectance techniques for diagnosing plant physiological status. *Trends in plant science*, 3(4) :151–156.

Planck, M. (2013). *The theory of heat radiation*. Courier Corporation.

Pollastri, F., Bolelli, F., Paredes, R., and Grana, C. (2020). Augmenting data with gans to segment melanoma skin lesions. *Multimedia Tools and Applications*, 79(21) :15575–15592.

Pradal, C., Dufour-Kowalski, S., Boudon, F., Fournier, C., and Godin, C. (2008). Openalea : a visual programming and component-based software platform for plant modelling. *Functional plant biology*, 35(10) :751–760.

Prusinkiewicz, P. (2004). Modeling plant growth and development. *Current opinion in plant biology*, 7(1) :79–83.

Prusinkiewicz, P. and Runions, A. (2012). Computational models of plant development and form. *New Phytologist*, 193(3) :549–569.

Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., and Sainath, T. (2019). Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2) :206–219.

Qin, J., Burks, T. F., Ritenour, M. A., and Bonn, W. G. (2009). Detection of citrus canker using hyperspectral reflectance imaging with spectral information divergence. *Journal of food engineering*, 93(2) :183–191.

Raabe, K., Pisek, J., Sonnentag, O., and Annuk, K. (2015). Variations of leaf inclination angle distribution with height over the growing season and light exposure for eight broadleaf tree species. *Agricultural and Forest Meteorology*, 214 :2–11.

Rabelais, F. (1534). *Gargantua*. Éditions Pocket.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv :1511.06434*.

Ragheb, H., Velastin, S., Remagnino, P., and Ellis, T. (2008). Vihasi : virtual human action silhouette data for the performance evaluation of silhouette-based action recognition methods. In *2008 Second ACM/IEEE International Conference on Distributed Smart Cameras*, pages 1–10.

Ramesh, S., Hebbar, R., Niveditha, M., Pooja, R., Shashank, N., Vinod, P., et al. (2018). Plant disease detection using machine learning. In *2018 International conference on design innovations for 3Cs compute communicate control*, pages 41–45.

Ramirez, A., Arguello, H., Arce, G. R., and Sadler, B. M. (2013). Spectral image classification from optimal coded-aperture compressive measurements. *IEEE Transactions on Geoscience and Remote Sensing*, 52(6) :3299–3309.

Reinhard, E., Heidrich, W., Debevec, P., Pattanaik, S., Ward, G., and Myszkowski, K. (2010). *High dynamic range imaging : acquisition, display, and image-based lighting*. Morgan Kaufmann.

Richardson, E., Sela, M., and Kimmel, R. (2016). 3d face reconstruction by learning from synthetic data. In *2016 fourth international conference on 3D vision (3DV)*, pages 460–469.

Rolland-Lagan, A.-G., Remmler, L., and Girard-Bock, C. (2014). Quantifying shape changes and tissue deformation in leaf development. *Plant physiology*, 165(2) :496–505.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The synthia dataset : A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243.

Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

Rouse, J., Haas, R., Schell, J., and Deering, D. (1974). Monitoring vegetation systems in the great plains with erts. *NASA special publication*, 351 :309.

Ruderman, D. and Bialek, W. (1993). Statistics of natural images : Scaling in the woods. *Advances in neural information processing systems*, 6 :551–558.

Ruiz-Altisent, M., Lleó, L., and Riquelme, F. (2006). Instrumental quality assessment of peaches : fusion of optical and mechanical parameters. *Journal of Food Engineering*, 74(4) :490–499.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088) :533–536.

Rumpf, T., Mahlein, A.-K., Steiner, U., Oerke, E.-C., Dehne, H.-W., and Plümer, L. (2010). Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance. *Computers and electronics in agriculture*, 74(1) :91–99.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3) :211–252.

Salazar-Vazquez, J. and Mendez-Vazquez, A. (2020). A plug-and-play hyperspectral imaging sensor using low-cost equipment. *HardwareX*, 7 :e00087.

Saleem, M. H., Potgieter, J., and Arif, K. M. (2019). Plant disease detection and classification by deep learning. *Plants*, 8(11) :468.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*, 29 :2234–2242.

Samiei, S., Rasti, P., Richard, P., Galopin, G., and Rousseau, D. (2020). Toward joint acquisition-annotation of images with egocentric devices for a lower-cost machine learning application to apple detection. *Sensors*, 20(15) :4173.

Sampath, A. and Shan, J. (2009). Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds. *IEEE Transactions on geoscience and remote sensing*, 48(3) :1554–1567.

Sanaeifar, A., Jafari, A., and Golmakani, M.-T. (2018). Fusion of dielectric spectroscopy and computer vision for quality characterization of olive oil during storage. *Computers and Electronics in Agriculture*, 145 :142–152.

Schanda, J. (2007). Cie colorimetry. *Colorimetry : Understanding the CIE system*, pages 25–78.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-CAM : Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision*, pages 618–626.

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3) :379–423.

Shanthamallu, U. S., Spanias, A., Tepedelenlioglu, C., and Stanley, M. (2017). A brief survey of machine learning methods and their sensor and iot applications. In *2017 8th International Conference on Information, Intelligence, Systems & Applications*, pages 1–8.

Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf : an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813.

Shaw, G. A. and Burke, H. K. (2003). Spectral imaging for remote sensing. *Lincoln laboratory journal*, 14(1) :3–28.

Shepp, L. A. and Vardi, Y. (1982). Maximum likelihood reconstruction for emission tomography. *IEEE transactions on medical imaging*, 1(2) :113–122.

Sher, Y., Cohen, L., Istrati, D., and Eisenberg, H. S. (2018). Low intensity lidar using compressed sensing and a photon number resolving detector. In *Emerging Digital Micromirror Device Based Systems and Applications X*, volume 10546, page 105460J.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1) :60.

Shwartz-Ziv, R. and Tishby, N. (2017). Opening the Black Box of Deep Neural Networks via Information. *arXiv :1703.00810 [cs]*. arXiv : 1703.00810.

Sigernes, F., Syrjäsuo, M., Storvold, R., Fortuna, J., Grøtte, M. E., and Johansen, T. A. (2018). Do it yourself hyperspectral imager for handheld to airborne operations. *Optics express*, 26(5) :6021–6035.

Siltanen, S. (2017). Formation of a ct sinogram. https://www.youtube.com/watch?v=q7Rt_OY_7tU. Accédé le 2021-02-03.

Simonyan, K. and Zisserman, A. (2014a). Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27 :568–576.

Simonyan, K. and Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*.

Singh, A., Ganapathysubramanian, B., Singh, A. K., and Sarkar, S. (2016). Machine learning for high-throughput stress phenotyping in plants. *Trends in plant science*, 21(2) :110–124.

Singh, A. K., Ganapathysubramanian, B., Sarkar, S., and Singh, A. (2018). Deep learning for plant stress phenotyping : trends and future perspectives. *Trends in plant science*, 23(10) :883–898.

Singh, S. C. (2009). *Basics of Light Emitting diodes, Characterizations and Applications*. Nova Scientific Publisher UK.

Smart, R. and Kankelborg, C. C. (2018). Machine learning techniques for computed tomography imaging spectroscopy of the solar atmosphere. In *AGU Fall Meeting Abstracts*.

Snoek, C. G., Worring, M., and Smeulders, A. W. (2005). Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402.

Sobel, I. (2014). An isotropic 3x3 image gradient operator. *Presentation at Stanford A.I. Project*, pages 271–272.

Spoelder, H. J. (1999). Virtual instrumentation and virtual environments. *IEEE Instrumentation & Measurement Magazine*, 2(3) :14–19.

Spring, K. and Davidson, M. (2020). Concepts in digital imaging technology : Quantum efficiency. http://hamamatsu.magnet.fsu.edu/articles/quantumefficiency.html. Accédé : 2020-11-13.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2015). Striving for Simplicity : The All Convolutional Net. *arXiv :1412.6806 [cs]*. arXiv : 1412.6806.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout : a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1) :1929–1958.

Strange, R. N. and Scott, P. R. (2005). Plant disease : a threat to global food security. *Annual review of phytopathology*, 43.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv :1312.6199*.

Takhar, D., Laska, J. N., Wakin, M. B., Duarte, M. F., Baron, D., Sarvotham, S., Kelly, K. F., and Baraniuk, R. G. (2006). A new compressive imaging camera architecture using optical-domain compression. In *Computational Imaging IV*, volume 6065, page 606509.

Tan, M. and Le, Q. (2019). Efficientnet : Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114.

Tang, Z., Gao, Y., Karlinsky, L., Sattigeri, P., Feris, R., and Metaxas, D. (2020). Onlineaugment : Online data augmentation with less domain knowledge. *arXiv preprint arXiv :2007.09271*.

Thomas, S., Kuska, M. T., Bohnenkamp, D., Brugger, A., Alisaac, E., Wahabzada, M., Behmann, J., and Mahlein, A.-K. (2018). Benefits of hyperspectral imaging for plant disease detection and plant protection : a technical perspective. *Journal of Plant Diseases and Protection*, 125(1) :5–20.

Tian, Y., Yang, G., Wang, Z., Li, E., and Liang, Z. (2019). Detection of apple lesions in orchards based on deep learning methods of cyclegan and yolov3-dense. *Journal of Sensors*, 2019.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497.

Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., and Birchfield, S. (2018). Training deep networks with synthetic data : Bridging the reality gap by domain randomization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 969–977.

Turing, A. (1950). Computing machinery and intelligence. *Mind*, 59(236) :433.

Ubbens, J., Cieslak, M., Prusinkiewicz, P., and Stavness, I. (2018). The use of plant models in deep learning : an application to leaf counting in rosette plants. *Plant methods*, 14(1) :6.

Ustin, S. L., Roberts, D. A., Gamon, J. A., Asner, G. P., and Green, R. O. (2004). Using imaging spectroscopy to study ecosystem processes and properties. *BioScience*, 54(6) :523–534.

Uto, K., Seki, H., Saito, G., Kosugi, Y., and Komatsu, T. (2016). Development of a low-cost hyperspectral whiskbroom imager using an optical fiber bundle, a swing mirror, and compact spectrometers. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(9) :3909–3925.

Valerio Giuffrida, M., Scharr, H., and Tsaftaris, S. A. (2017). Arigan : Synthetic arabidopsis plants using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2064–2071.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Volin, C. E., Ford, B. K., Descour, M. R., Garcia, J. P., Wilson, D. W., Maker, P. D., and Bearman, G. H. (1998). High-speed spectral imager for imaging transient fluorescence phenomena. *Applied optics*, 37(34) :8112–8119.

Volin, C. E., Garcia, J. P., Dereniak, E. L., Descour, M. R., Hamilton, T., and McMillan, R. (2001). Midwave-infrared snapshot imaging spectrometer. *Applied optics*, 40(25) :4501–4506.

Vose, M. D. and Horton, M. D. (2007). A heuristic technique for ctis image reconstruction. *Applied optics*, 46(26) :6498–6503.

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer vision : A brief review. *Computational intelligence and neuroscience*, pages 1–13.

Wagadarikar, A., John, R., Willett, R., and Brady, D. (2008). Single disperser design for coded aperture snapshot spectral imaging. *Applied optics*, 47(10) :B44–B51.

Wagner, J., Fischer, V., Herman, M., and Behnke, S. (2016). Multispectral pedestrian detection using deep fusion convolutional neural networks. In *European Symposium on Artificial Neural Networks*.

Wang, J., Perez, L., et al. (2017). The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Visual Recognition*, 11.

Wang, Q., Gao, J., Lin, W., and Yuan, Y. (2019). Learning from synthetic data for crowd counting in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8198–8207.

Wang, W., Tran, D., and Feiszli, M. (2020a). What makes training multi-modal classification networks hard? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12695–12705.

Wang, X., Yang, W., Wheaton, A., Cooley, N., and Moran, B. (2010). Efficient registration of optical and ir images for automatic plant water stress assessment. *Computers and Electronics in Agriculture*, 74(2) :230–237.

Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020b). Generalizing from a few examples : A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3) :1–34.

Ward, D., Moghadam, P., and Hudson, N. (2018). Deep leaf segmentation using synthetic data. *arXiv preprint arXiv :1807.10931*.

Warden, P. (2017). How many images do you need to train a neural network? https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/. Accédé : 2021-03-09.

Wei, Q., Bioucas-Dias, J., Dobigeon, N., and Tourneret, J.-Y. (2015). Hyperspectral and multispectral image fusion based on a sparse representation. *IEEE Transactions on Geoscience and Remote Sensing*, 53(7) :3658–3668.

Westerlund, M. (2019). The emergence of deepfake technology : A review. *Technology Innovation Management Review*, 9(11).

Wilson, D. W., Maker, P. D., and Muller, R. E. (1997). Reconstructions of computed-tomography imaging spectrometer image cubes using calculated system matrices. In *Imaging Spectrometry III*, volume 3118, pages 184–193.

Würfl, T., Ghesu, F. C., Christlein, V., and Maier, A. (2016). Deep learning computed tomography. In *International conference on medical image computing and computer-assisted intervention*, pages 432–440.

Yang, J., Jin, T., Xiao, C., and Huang, X. (2019). Compressed sensing radar imaging : fundamentals, challenges, and advances. *Sensors*, 19(14) :3100.

Yao, Q., Wang, M., Chen, Y., Dai, W., Li, Y.-F., Tu, W.-W., Yang, Q., and Yu, Y. (2018). Taking human out of learning applications : A survey on automated machine learning. *arXiv preprint arXiv :1810.13306*.

Yap, B. W., Abd Rani, K., Abd Rahman, H. A., Fong, S., Khairudin, Z., and Abdullah, N. N. (2014). An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets. In *Proceedings of the first international conference on advanced data and information engineering*, pages 13–22.

Yi, X., Walia, E., and Babyn, P. (2019). Generative adversarial network in medical imaging : A review. *Medical image analysis*, 58 :101552.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

Yosinski, J., Clune, J., Fuchs, T., and Lipson, H. (2015). Understanding neural networks through deep visualization. In *In ICML Workshop on Deep Learning*.

Yu, S., Jia, S., and Xu, C. (2017). Convolutional neural networks for hyperspectral image classification. *Neurocomputing*, 219 :88–98.

Yue, J., Zhao, W., Mao, S., and Liu, H. (2015). Spectral–spatial classification of hyperspectral images using deep convolutional neural networks. *Remote Sensing Letters*, 6(6) :468–477.

Yuhas, R. H., Goetz, A. F., and Boardman, J. W. (1992). Discrimination among semi-arid landscape endmembers using the spectral angle mapper (sam) algorithm. In *Proc. Summaries 3rd Annu. JPL Airborne Geosci. Workshop*, volume 1, pages 147–149.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833.

Zeng, L. and Wang, G. (2009). Modeling golden section in plants. *Progress in Natural Science*, 19(2) :255–260.

Zhang, N., Wang, M., and Wang, N. (2002). Precision agriculture—a worldwide overview. *Computers and electronics in agriculture*, 36(2-3) :113–132.

Zheng, Y., Lin, S., Kambhamettu, C., Yu, J., and Kang, S. B. (2008). Single-image vignetting correction. *IEEE transactions on pattern analysis and machine intelligence*, 31(12) :2243–2256.

Zhou, Y., Zhang, L., Xing, C., Xie, P., and Cao, Y. (2019). Target three-dimensional reconstruction from the multi-view radar image sequence. *IEEE Access*, 7 :36722–36735.

Zhu, F., He, M., and Zheng, Z. (2020). Data augmentation using improved cdcgan for plant vigor rating. *Computers and Electronics in Agriculture*, 175 :105603.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232.

Zitova, B. and Flusser, J. (2003). Image registration methods : a survey. *Image and vision computing*, 21(11) :977–1000.

Zoph, B. and Le, Q. V. (2016). Neural architecture search with reinforcement learning. *arXiv preprint arXiv :1611.01578*.