

RECONNAISSANCE AUTOMATIQUE DE PUCERONS PAR LA MÉTHODE DES FORÊTS ALÉATOIRES

Herearii Metuarea

Projet annuel encadré par

David ROUSSEAU, professeur

Université d'Angers, LARIS-INRAE

Amandine CORNILLE, chargée de recherche
ECLECTIC

Centre National de la Recherche Scientifique



TABLE DES MATIÈRES

1	Introduction	3
2	Origine du projet	4
2.1	Enjeux et motivation	4
2.2	Jeu de données	5
3	Revue de littérature	6
3.1	Préliminaire : image numérique	6
3.1.1	Image au niveau de gris	7
3.1.2	Image au niveau de couleur	7
3.2	Forêt aléatoire pour la segmentation	8
3.2.1	Principe de la classification de pixel	8
3.2.2	Forêt aléatoire pour la classification d'objet	11
3.3	Ligne de partage des eaux	13
3.3.1	Description de l'algorithme	13
3.3.2	Approche théorique	14
3.3.3	Algorithme de ligne de partage des eaux	17
3.3.4	Application	18
3.4	Résumé du traitement d'image sous Ilastik	19
4	Méthodologie	20
4.1	Matériel	20
4.2	Segmentation des images	20
4.3	Traitement du chevauchement en utilisant l'algorithme de ligne de partages des eaux	22
4.4	Nomenclature de l'évolution des pucerons pour la classification d'objets	22
4.5	Classification d'objets	23
4.6	Méthode de validation du modèle	23
5	Résultat	25
6	Discussion	26
7	Conclusion	26
8	Glossaire	27

1

INTRODUCTION

Le puceron cendré du pommier, *Dysaphis plantaginea*, est le puceron ravageur le plus nuisible de la culture de la pomme. Il cause chaque année des dégâts considérable sur son hôte, le pommier cultivé, *Malus domestica*. D'une part, sa salive détériore la qualité des feuilles et des fruits du pommier. D'autre part, l'exploitation de la sève par ces pucerons rend son hôte affaibli. Nuisible pour le pommier, le puceron cendré du pommier entraîne des pertes agricoles importantes chaque année. La connaissance du développement du puceron cendré du pommier sur son hôte et dans différents environnements permettra de concevoir un programme de lutte adapté contre cette espèce.

L'équipe ECLECTIC (Ecologie et génomique des interactions multi-espèces) a réalisé une étude cherchant à savoir si l'adaptation du puceron cendré du pommier est favorisée soit par son environnement local ou/et soit par son hôte, le pommier. Un test expérimental a été lancé dans lequel les pucerons de différentes origines (belges, françaises et espagnoles) sont infestés sur différentes variétés de pommier cultivé ont été greffées puis plantées l'année précédente en France, en Espagne et en Belgique. Ce test expérimental a pour objectif de déterminer le taux de croissance de chaque colonie infestée (nombre de pucerons par unité de temps) d'origine différentes (France, Belgique, Espagne) sur les différentes variétés de pommier cultivé (France, Belgique, Espagne) dans différents environnements (France, Belgique, Espagne). A la fin de ce test, les colonies de pucerons infestées sur chaque arbre, ont été récupérées et fixées dans de l'éthanol et ont été photographiées. L'objectif de mon projet a été de proposer une méthode objective et automatisée de détection et de comptage du nombre de pucerons par stades de développement à partir des photos fournies par l'équipe ECLECTIC.

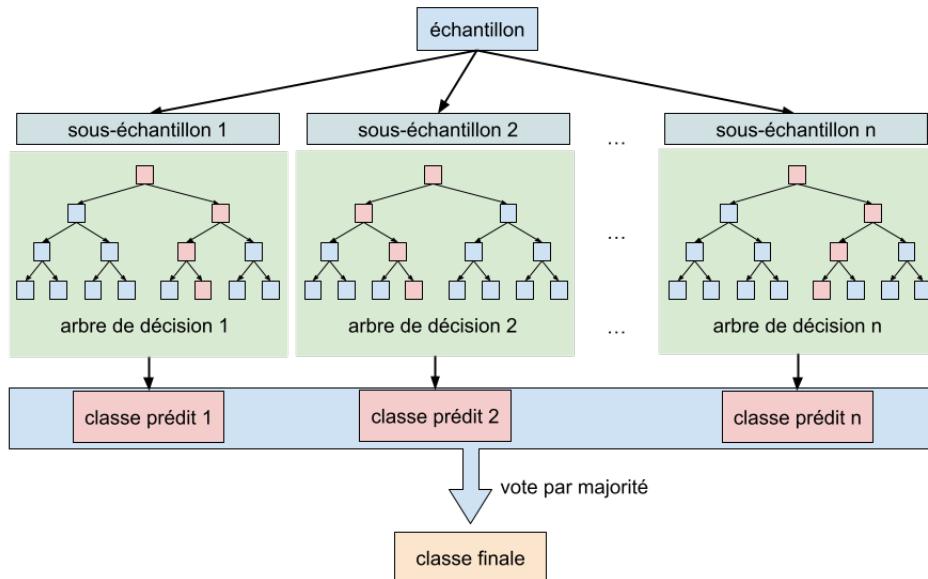


FIGURE 1. Illustration de la méthode des forêts aléatoires pour la classification

Notre approche utilise des méthodes de *machine learning*. Notre méthode de détection consiste à segmenter les images pour mettre en évidence les pucerons; puis à classer ces pucerons (Figure 1). Les étapes de segmentation et de classification s'appuient sur la méthode des forêts aléatoires pour la classification [2] [3]. Essentiellement, cette méthode consiste à entraîner sur chaque sous-échantillon de données tiré au hasard et avec remise un arbre décisionnel. Ensuite, nous obtenons plusieurs prédicteurs associés à différents arbres. Enfin,

on choisit le prédicteur qui apparaît le plus souvent parmi les prédicteurs qui apparaissent le plus souvent parmi les prédicteurs obtenus dans les différents arbres (figure 1).

La difficulté dans la résolution de la problématique est la présence d'objets qui se chevauchent dans certaines images. La figure 2 montre un exemple de chevauchement dans une image du projet.



FIGURE 2. Photographie illustrant les problèmes de disposition des pucerons rencontrés. À gauche, trois pucerons correctement séparés. À droite, deux pucerons se chevauchent.

Dans ce rapport, notre approche exploite deux outils du *machine learning* :

- les forêts aléatoires pour la classification [2] [3]
- la segmentation par ligne de partage des eaux [1]

Les forêts aléatoires pour la classification nous permettront de segmenter les images et de classifier les objets dans les images. La segmentation par ligne de partage des eaux contournera le problème de chevauchement des objets. L'agrégation des classes nous apportera des données quantitatives par classe.

Le rapport est organisé en quatre chapitres. Le premier chapitre présente le projet de recherche dans lequel ce rapport s'inscrit. Dans la suite, nous expliquons notre méthode en deux parties. En première partie, nous présentons le principe de la classification de pixel et d'objet par la méthode des forêts aléatoires. En deuxième partie, nous présentons la méthode de la segmentation par ligne de partage des eaux. Enfin, on présente la méthodologie suivie dans la collecte des données, le traitement et l'analyse des images. Puis, nous expliquerons les résultats quantitatifs et qualitatifs, puis nous conclurons.

2 ORIGINE DU PROJET

Dans cette partie, on présente l'origine du projet. En premier, on explique les enjeux pour soulever la problématique. Enfin, on présente la base de données.

2.1 ENJEUX ET MOTIVATION

L'équipe ECLECTIC (Ecologie et génomique des interactions multi-espèces) s'intéresse aux mécanismes écologiques et évolutifs à l'origine de la diversité des plantes, et de leurs pucerons associés, et à la manière dont ils évoluent et s'adaptent à leur environnement. Parmi ces recherches, on peut citer l'étude de l'impact de la domestication de l'hôte, le pommier, sur l'évolution de son ravageur, le puceron cendré du pommier *Dysaphis plantaginea*.

Le projet a pour objectif de comprendre si le puceron cendré du pommier (*Dysaphis plantaginea*) est adapté localement à son milieu et/ou à son hôte (le pommier, *Malus domestica*). Des pommiers de génotypes identiques d'origine française, belge et espagnole ont été plantés en France, en Belgique et en Espagne. L'hypothèse qui

vise à tester l'adaptation locale du puceron à son hôte ou à son environnement se formule comme suit : si une population de pucerons d'origine identique que le pommier et du lieu où s'est effectuée la transplantation se développe plus que les autres pucerons d'origines différentes alors il y a une adaptation locale de la population de pucerons à son hôte (figure 2).

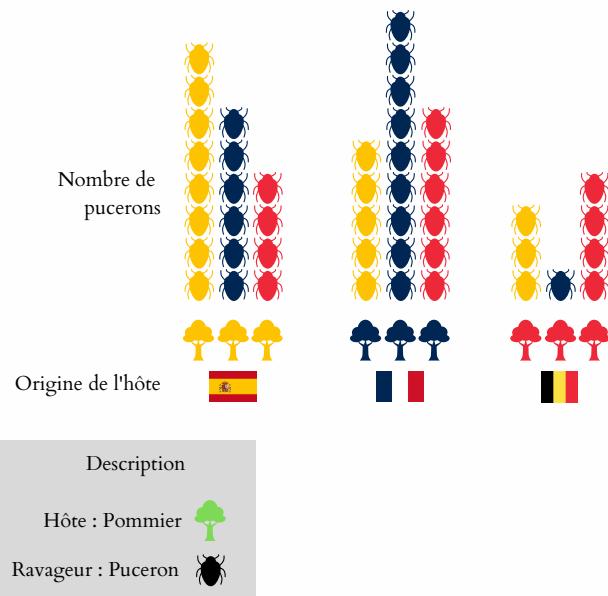


FIGURE 2. Résultat attendu s'il y a une adaptation du puceron à son hôte d'origine et à son environnement

En 2019, 28 génotypes de pommier ont été prélevés en Belgique, en France et en Espagne. Trois colonies de chaque origine géographique ont été prélevées. Les pommiers ont ensuite été infestés manuellement par pucerons de chaque origine sur les trois sites. Après 12-15 jours d'infestation, chaque colonie a été récupérée et rangée dans un flacon avec de l'éthanol (96 %). Le reste de l'expérience consiste à compter le nombre de pucerons dans chaque colonie. Le comptage manuel des pucerons est très coûteux en temps compte tenu des nombreuses colonies récoltées sur le terrain, et la détermination des stades peut être subjective (expérimentateur dépendant). On s'intéresse donc à une méthode objective de dénombrement de pucerons, et de leur stade de développement, efficace en termes de temps. Les résultats de cette méthode permettent d'obtenir des données quantitatives à savoir le nombre de pucerons au total et par stade de développement (adultes, larves, ailées). Ce résultat pourra en conséquence conclure sur les processus d'adaptation locale des pucerons à leur hôtes.

Notre approche exploite les outils du *Machine learning* pour contourner cela. Plus précisément, à partir d'images sur lesquelles figurent les colonies, on effectue une segmentation des images par la méthode des forêts aléatoires. Nous corrigéons les chevauchements d'objets par la segmentation de ligne par partage des eaux. Enfin, nous réalisons une classification des objets par forêt aléatoire.

2.2 JEU DE DONNÉES

Un total de 863 images de colonies de pucerons prises sous loupe binoculaire en boîte de Petri et éthanol ont été mises à disposition par l'équipe ECLECTIC. La taille du jeu de donnée brute vaut 4.98 Go. Ces images représentaient 98 colonies de pucerons :

- 470 images représentent 30 colonies de pucerons sous différents angles
- 393 images représentent 68 colonies de pucerons. Certaines colonies qui possède de nombreux pucerons ont été prises sur plusieurs images.

Le traitement du jeu de données brut consistait à éliminer les images en double. Ce traitement consistait aussi à concevoir une base de donnée qui rassemble des images qui représentent une seule colonie. Après traitement, le jeu de données est composé de 373 images. Dans notre projet, on utilise le *shallow learning* pour réaliser le traitement des images. Le tableau 1. montre la composition de notre échantillon d'entraînement et test.

TABLEAU 1. Description du jeu de donnée

Taille en pixel	Echantillon d'entraînement	Echantillon Test
2560×1920	22	351

Les fichiers se présentaient au format **.tiff**. Sous ce format, la qualité des images est préservée (compression et zone de transparence) rendant leur exploitation efficace pour une segmentation.

Les colonies de pucerons ont été prises à différentes échelles : 200µm, 100µm et 20µm. Ces échelles permettent de préserver la surface de chaque objet dans les images. Ainsi, la surface de chaque catégorie de pucerons est uniforme dans toutes les images.

3 REVUE DE LITTÉRATURE

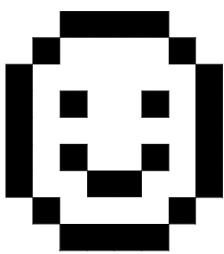
Cette section présente la classification de Pixel. Elle s'ouvre sur la définition d'une image matricielle. Ensuite, on énonce le principe du Pixel classification. On s'appuie sur [8], [9], [5], [6], [1].

Dans ce projet, on s'intéresse aux images matricielles à deux dimensions. Ci-dessous, une définition d'une image numérique.

3.1 PRÉLIMINAIRE : IMAGE NUMÉRIQUE

Une image numérique est un tableau bidimensionnel à n lignes et p colonnes. L'élément d'indice (y, x) est la couleur du pixel d'ordonnée y et d'abscisse x . L'origine $(0, 0)$ correspond au coin supérieur à gauche de l'image. Aussi, l'élément d'indice (y, x) est un entier. Chaque entier correspond à un niveau de couleur en pixel. La définition d'une image porte sur le nombre de pixels qui le compose. Une image de hauteur n pixels et de largeur p pixels contient $n \times p$ pixels au total.

La figure 4 présente un exemple avec une image monochrome. La dimension de l'image est 10 x 11 pixels. Sa matrice définit l'entier 0 pour la couleur noir ; l'entier 1 pour la couleur blanche. La matrice contient 10 colonnes et 11 lignes. Alors, l'image originale contient 110 pixels.



1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	1	1	1	1
1	1	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	0	1	0	1	1	0	1	0	1	1
1	0	1	1	1	1	1	1	0	1	1
1	0	1	0	1	1	0	1	0	1	1
1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	0	1	1	0	1	1
1	1	0	1	1	1	1	0	1	1	1
1	1	1	0	0	0	0	1	1	1	1

FIGURE 4. Exemple avec une image monochrome. A gauche, l'image originale. A droite, la matrice de l'image de gauche.

Source : lossendiere

En général, une image possède 256 niveaux de couleur et 256 niveaux de gris codés en 8 bits. Dans la matrice de taille $n \times p$, un entier est compris entre 0 et 255. L'intervalle 0 et 255 définit la clarté de la couleur. Un entier proche de 0 présente une couleur plus sombre ; un entier proche de 255 présente une couleur plus claire.

3.1.1 • IMAGE AU NIVEAU DE GRIS

Une image au niveau de gris présente une image en noir et blanc. L'intervalle [0, 255] présente le niveau de luminosité entre le noir et le blanc. Les entiers au milieu de [0, 255] présentent des niveaux de gris. Dans une image en niveau de gris, il y a 256 teintes possibles pour chaque pixel de l'image (Figures 5 et 6).

000 008 016 024 032 040 048 056 064 072 080 088 096 104 112 120 128 136 144 152 160 168 176 184 192 200 208 216 224 232 240 248 255

FIGURE 5. Niveau de gris

Source : Wikipedia

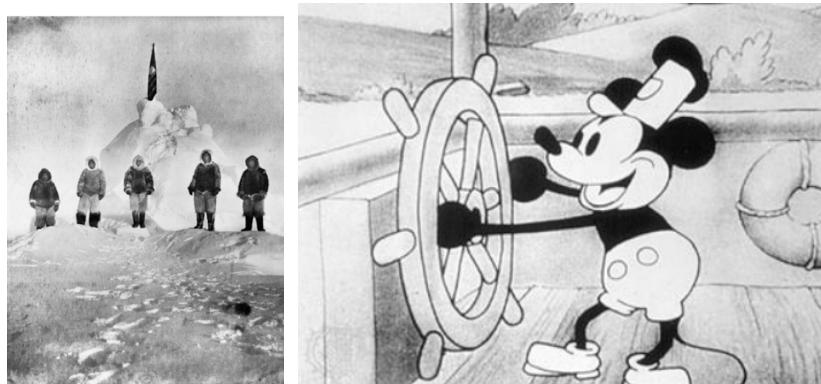


FIGURE 6. Exemples de photos en niveau de gris. De gauche à droite : Première photo prise au pôle Nord (1909) par Amiral Robert E. Peary ; Mickey Mouse dans l'épisode Steamboat Willie (1928)

3.1.2 • IMAGE AU NIVEAU DE COULEUR

Une image en couleur est une combinaison de trois matrices de taille identique donnant chacune l'intensité du rouge, du vert et du bleu (abrégé « RGB »). Pour chaque matrice, le nombre d'entiers est compris entre 0 et 255. Donc, chaque élément de la matrice d'une image en couleur est un triplet $(r, v, b) \in [0, 255]^3$ où r, v, b est associé au niveau de couleur rouge, vert et bleu.

$$(r, g, b) = \left\lfloor \frac{r + g + b}{3} \right\rfloor$$

Expression de la luminosité de la couleur dans le codage RGB

Dans une image en couleur, il est possible de représenter au plus 256^3 teintes possibles soit 16,7 millions de teintes (Figure 7). Dans une image en couleur, la différence de nuances entre deux couleurs est imperceptible à l'œil nu. Le système RGB permet une restitution exacte des couleurs visibles par l'homme (Figure 8).



FIGURE 7. De haut en bas, niveau de couleur pour Rouge (haut), Vert (milieu) et Bleu (bas)

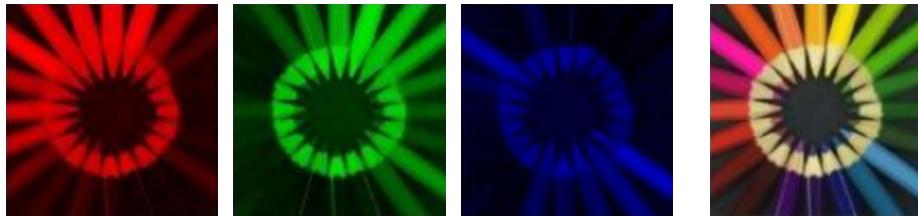


FIGURE 8. Exemple de trois images d'intensité rouge, vert et bleu combinées pour obtenir une image en couleur. A gauche, trois images d'intensité Rouge, Vert et Bleu ; A droite, le résultat de la combinaison des trois images d'intensité Rouge, Vert et Bleu.

Source : APMEP n°500

3.2 FORÊT ALÉATOIRE POUR LA SEGMENTATION

Le Pixel classification est un algorithme pour le traitement d'image qui permet de déterminer une ou plusieurs régions prédéfinies dans une image. L'algorithme de classification en pixel prend en entrée une image originale 2D ou 3D, puis retourne une image présentant des régions de segmentation. L'algorithme consiste à déterminer les *features* dans lesquelles chaque pixel sera assigné à une classe grâce à l'algorithme des forêts aléatoires pour la classification. L'algorithme est divisé en 4 grandes étapes (Figure 9).

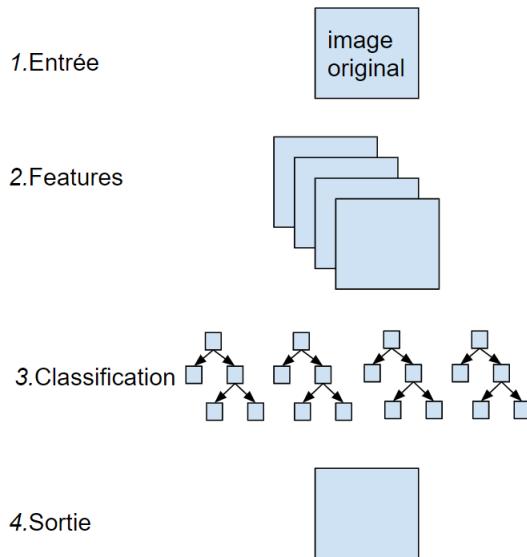


FIGURE 9. Illustration simple de l'algorithme de Pixel classification

3.2.1 • PRINCIPE DE LA CLASSIFICATION DE PIXEL

On considère une image matricielle 2D, plus précisément un tableau tridimensionnelle.

Prétraitement de l'image. Un prétraitement de l'image d'entrée vise à améliorer les caractéristiques de l'image à l'aide d'un filtre de convolution.

Définition 1. (Filtre de convolution)

Soit $A \in M_{n \times m}(\mathbb{N})$ une matrice définie par $A = (a_{i,j})$ et un noyau de convolution $C \in M_{k \times l}(\mathbb{N})$ défini par

$$C = (c_{i,j}).$$

Le filtre de convolution est la matrice $A * N$ définie par

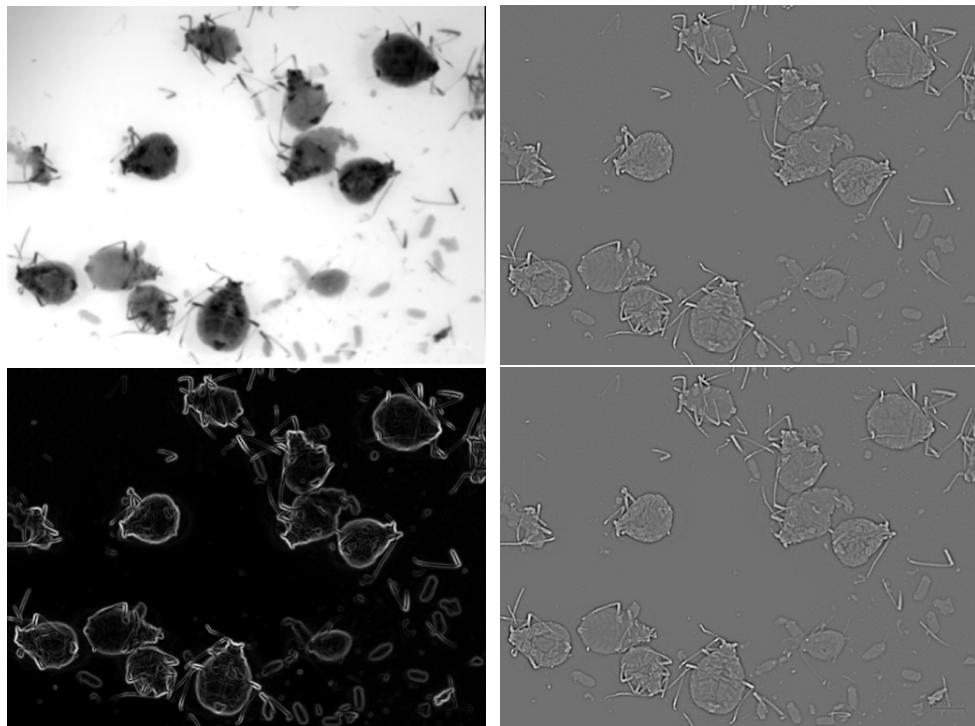
$$\forall(x,y), (A * N)(x,y) = \sum_i \sum_j a_{i,j} c_{x-i+1,y-j+1}$$

Chaque matrice couleur RGB se voit appliquer le filtre de convolution pour un noyau de convolution. Le type du filtre correspond au type du noyau de convolution. Chaque type de noyau de convolution permet de répondre à un problème parmi les trois problèmes suivant : la détection de bord, l'intensité des couleurs et la texture. La taille du noyau de convolution détermine le niveau de netteté de l'image. Ainsi, les images d'entrée prétraitées par le filtre de convolution forment les *features* qui seront utilisées pour la classification des pixels.

Le logiciel Ilastik effectue ce pré-traitement dans la rubrique Features selection. Pour chaque caractéristique à savoir la détection de bord (*edge*), l'intensité des couleurs (*intensity/color*) et la texture (*texture*), l'utilisateur annote un sigma (σ) qui définit la taille d'un noyau de convolution. Par défaut, le sigma (σ) est compris 0.3 et 10.0. Le logiciel Ilastik comprend cinq filtres (Tableau 2). La figure 10 présente le filtre de convolution de différents *features* appliquées à une image du projet ECLECTIC.

TABLEAU 2. Features utilisées dans Ilastik pour mettre en évidence le contour, l'intensité de la couleur ou la texture d'un objet dans une image.

Feature	Objectif
Filtre Gaussien	Couleur et intensité
Filtre Laplacien de gausse	Contour et objet
Valeur propre <i>structure tensor</i>	Objet filiforme
Valeur propre d'une hessienne	Objet filiforme
<i>Gradient magnitude</i>	Bord



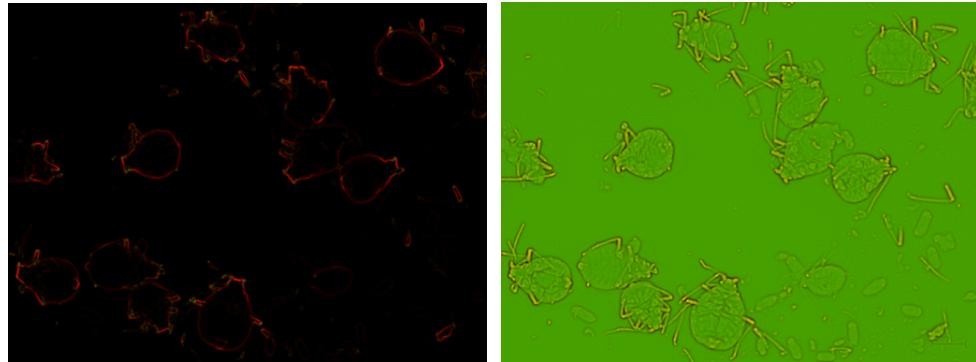


FIGURE 10. Quelques *features* appliquées à une image au niveau d'intensité rouge obtenu par la rubrique Features selection. [1] Filtre Gaussien ($\sigma = 5.0$) ; [2] Filtre Laplacien de gausse ($\sigma = 5.0$) ; [3] *Gaussian Gradient Magnitude* ($\sigma = 5.0$) ; [4] *Difference of Gaussians* ($\sigma = 5.0$) ; [5] Valeur propre de *Structure Tensor* ($\sigma = 5.0$) ; [6] Valeur propre de la hessienne d'une gaussienne ($\sigma = 5.0$). Sources : Snap-160 du projet ECLECTIC

Semantic Segmentation. On considère les différents filtres pour chaque matrice couleur RGB. Pour une matrice de couleur filtrée, on associe un pixel à une classe qui formera un échantillon d'entraînement. Les pixels qui n'ont pas été attribués à une classe forment un échantillon test. Sur cet échantillon test, on applique l'algorithme des forêts aléatoires. Dans chaque arbre de classification, la racine de l'arbre est la valeur d'un pixel et une feuille est une classe (Figure 11.1). Appliquée à chaque pixel de l'image, la méthode des forêts aléatoires permet d'obtenir en somme une segmentation de l'image en différente région. Les arbres de classification ne sont pas corrélés entre eux à cause de la randomisation par *bootstrap*. Par conséquent, la méthode de forêts aléatoires permet d'éviter le surapprentissage. Aussi, la rapidité du temps d'exécution des forêts aléatoires rend le traitement d'une image particulièrement rapide. L'erreur de prédiction d'un pixel est calculée à partir de la métrique *Out of bag error*.

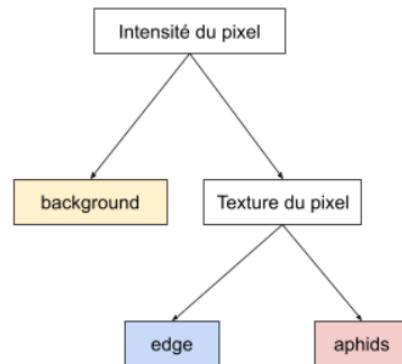


FIGURE 11.1. Illustration d'un arbre de classification



FIGURE 11.2. Annotation en trois couleurs des pixels d'entraînement.

Le logiciel Ilastik effectue la classification dans la rubrique training. L'utilisateur annote différentes régions de l'image par une couleur correspondant à une classe (Figure 11.2). Les pixels qui se trouvent dans les régions annotées constituent l'échantillon d'entraînement. Les pixels qui ne se trouvent pas dans une région annotée constituent l'échantillon test. La prédiction des classes pour l'échantillon test est réalisée par l'algorithme des forêts aléatoires avec la fonction Live Update. L'algorithme des forêts aléatoires est fourni par la librairie VIGRA de C++. Cet algorithme rassemble par défaut 100 arbres de classification. Les arbres de classification ne possèdent aucune profondeur. Sous Ilastik, la méthode permet particulièrement d'obtenir deux images :

- Simple Segmentation : image originale segmentée par région.
- Probability map : image où l'intensité de chaque pixel correspond à la probabilité d'appartenir fortement à une région

La figure 12 présente le Simple Segmentation et Probability map de Snap-160.

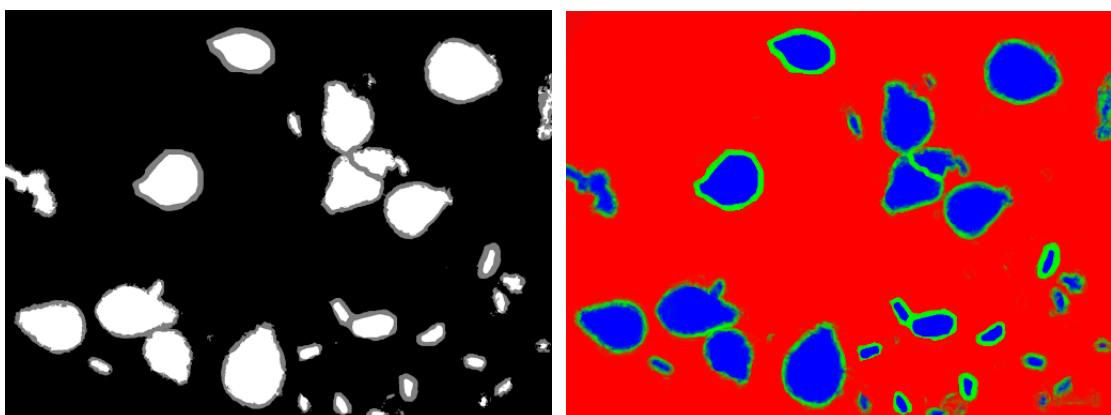


FIGURE 12. Sortie du Pixel Classification sous Ilastik. A gauche, simple segmentation ; A droite, pixel prediction map

Source : Snap-160 du projet ECLECTIC

3.2.2 • FORÊT ALÉATOIRE POUR LA CLASSIFICATION D'OBJET

La classification d'objet est un algorithme pour déterminer la classe de chaque objet. Elle prend en entrée une image 2D puis retourne la classe de chaque objet dans l'image 2D. L'algorithme consiste à entraîner le classifieur des objets puis de prédire la classe des autres objets grâce à l'algorithme des forêts aléatoires pour la classification.

Pré-traitement. Un prétraitement consiste à effectuer un seuillage et un lissage sur une image où les deux images d'entrées sont superposées. Ce prétraitement permet d'améliorer la qualité de représentation des objets segmentés en jouant sur le lissage et le seuillage de l'image. Le lissage est réalisé en sélectionnant correctement la taille du noyau de convolution qui permet d'écartier le bruit dans les images après application du filtre de convolution. Le seuillage est réalisé en conservant les niveaux de couleur sous un seuil correctement choisi.

Pour chaque objet segmenté, on calcule quelques mesures quantitatives à partir de sa surface. Ces mesures quantitatives forment les *features* qui seront utilisées pour la classification. Sous Ilastik, le lissage et le seuillage de l'image sont réalisés dans la rubrique Threshold and Sizer Filter. L'extraction des caractéristiques des objets segmentés est réalisée dans la rubrique Object Feature Selection. Ces caractéristiques sont réunies en trois grandes familles : Standard Object Features, Skeleton Features (2D only), and Convex Hull Features. Chaque famille détient trois groupes de *features* à savoir Location, Shape et Intensity Distribution. Ces features sont calculées avec la fonction ObjectExtractionApplet de la librairie Vigna de C++.

Classification. Pour un objet, on associe la surface en pixel qu'occupe cet objet, la forme de cette surface dans le tableau à une classe.

Plus largement, les objets associés à différentes classes forment un échantillon d'entraînement. Les objets non associés à une classe forment un échantillon test. Sur cet échantillon test, on applique l'algorithme des forêts aléatoires pour estimer l'attribution d'un objet à une classe. Dans chaque arbre de classification, la racine de l'arbre est la surface en pixel et une feuille est une classe. Appliquée à chaque surface d'objet dans l'échantillon test, la méthode des forêts aléatoires permet d'obtenir en somme une classification des objets.

Le logiciel Ilastik effectue la classification d'objet dans la rubrique Object classification. Après avoir entré l'image numérique originale et l'image segmentée (sortie de Pixel Classification), l'image peut effectuer le prétraitement dans la rubrique Threshold and Size Filter. L'utilisateur sélectionne les *features* sur un tableau dans la rubrique Object Feature Selection. L'entraînement des objets est réalisé dans la rubrique Training. L'utilisateur annote chaque objet par une couleur correspondant à une classe et entraîne l'échantillon test avec le bouton Live Test. La figure 13 présente les annotations d'entraînement et le résultat de la prédiction de classe pour une image en niveau de gris. En sortie, on obtient un tableau de données quantitatives associé aux objets d'une image.

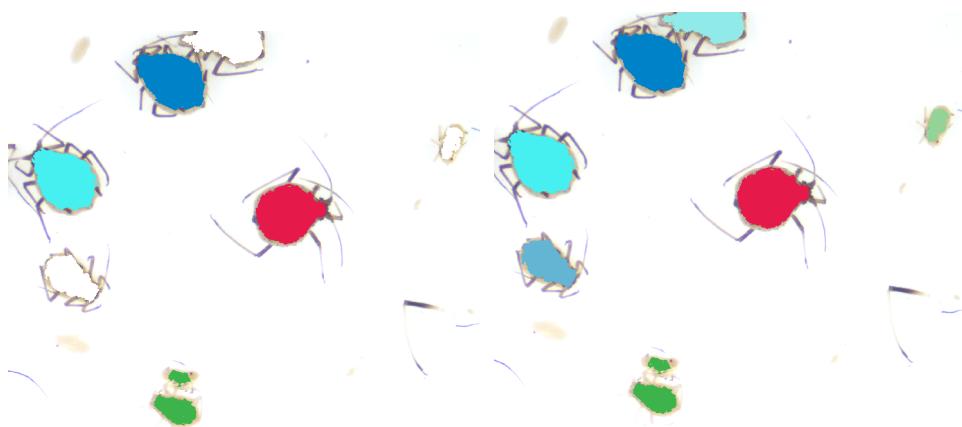


FIGURE 13. Sortie de Object classification sous Ilastik de la prédiction des classes
Source : Snap-160 du projet ECLECTIC

3.3 LIGNE DE PARTAGE DES EAUX

L'algorithme de partage des eaux est un algorithme de segmentation d'image. L'approche de cet algorithme exploite la théorie de la morphologie mathématiques. Bien que cet algorithme s'applique à tout types d'images, on se limite aux images de niveau gris. Cette partie s'ouvre sur une description non formelle de l'algorithme. Enfin, on présente quelques outils de la morphologie mathématiques. Puis, on présente l'algorithme.

On considère une image en niveau de gris.

3.3.1 • DESCRIPTION DE L'ALGORITHME

D'un point de vue de la théorie de la morphologie mathématiques, chaque niveau de gris de l'image est perçu comme un niveau d'une surface topologique. L'image est représentée par un relief avec un sommet, des reliefs et un bas relief. Le sommet correspond au niveau de gris très claire ; le bas relief correspond au niveau de gris très sombre. Ces bas relief sont appelés minimum régionaux.

La figure 14 présente un exemple de reliefs par les pixels d'une image.

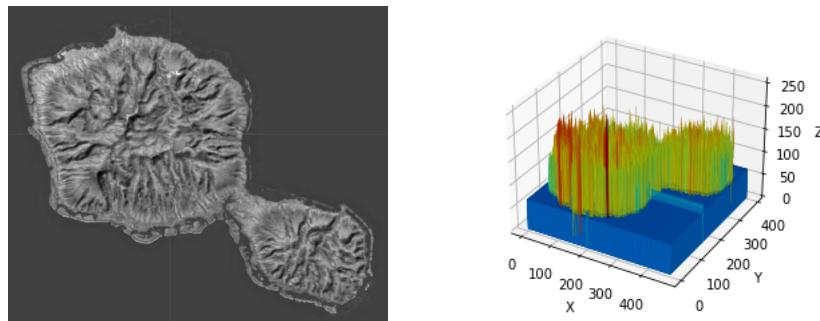


FIGURE 14. Illustration d'un relief formé par les pixels d'une image. L'image de droite représente les reliefs présentés par les pixels de l'image de gauche

Dans chaque minimum régionaux, on immerge de l'eau. Chaque vallée possède un bassin d'eau. Au fur et à mesure que l'eau monte, deux bassins d'eau risquent de se rencontrer. Pour éviter que deux bassins d'eau se recontrent, on fabrique sur le lieu de rencontre une digue d'une hauteur égale à la hauteur du plus grand sommet dans le relief. L'ensemble de ces digues est la ligne de partage des eaux.

La figure 15 illustre un relief présentant deux vallées (en vert) avec deux bassins d'eau (en bleu). Au fond de la vallée, le point rouge indique le bas relief. Le sommet du relief est indiqué par un point violet. La digue est la paroi en couleur beige.

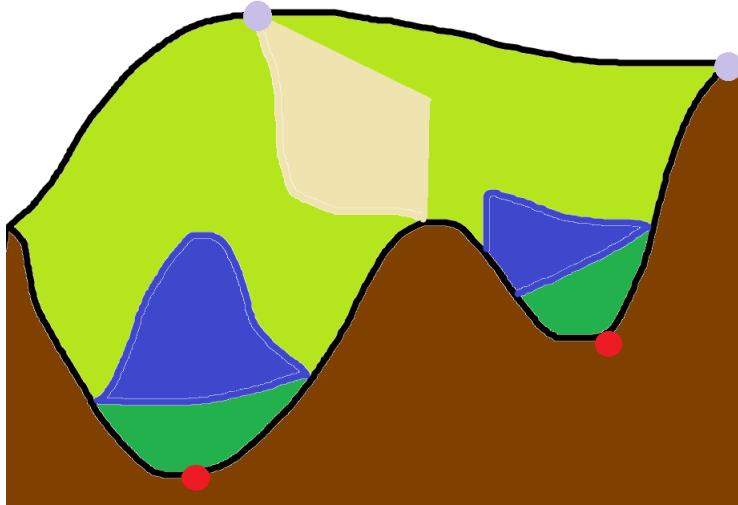


FIGURE 15. Croquis d'un relief

3.3.2 • APPROCHE THÉORIQUE

Dans cette section, on présente formellement l'algorithme de ligne en partages des eaux. Elle s'ouvre par une définition de quelques objets de la théorie de la morphologie mathématique. Enfin, on présente l'algorithme de la ligne de partage des eaux.

Préliminaire : première définition On considère une fonction f qui définit la valeur de nuance de gris en pixel dans un tableau bidimensionnelle.

Définition 1. On appelle ensemble des points de f supérieur à la hauteur i l'ensemble $X_i(f)$ définie par $X_i(f) = \{(x, y) \in \mathbb{Z}, f(x, y) \geq i\}$.

Définition 2. On appelle ensemble des points de f inférieur à la hauteur i l'ensemble $Z_i(f)$ définie par $Z_i(f) = \{(x, y) \in \mathbb{Z}, f(x, y) \leq i\}$.

On considère $X \subset \mathbb{Z}^2$.

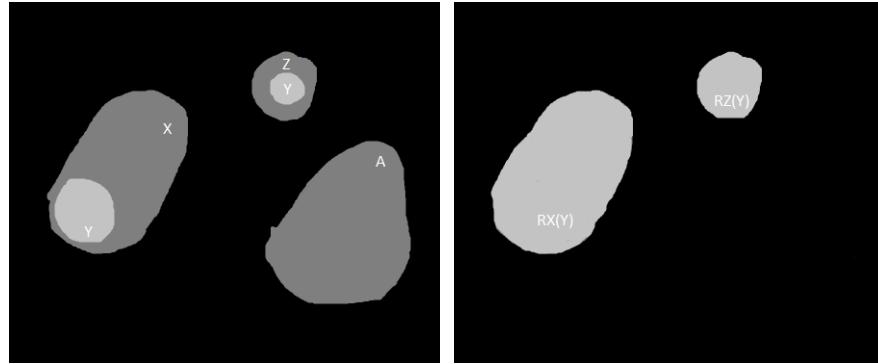
La distance géodésique entre x et y est la plus petite distance entre x et y dans X . On note $d_X(x, y)$ la distance géodésique de x et y sur X .

Définition 3. Soit $Y \subset X$.

On appelle reconstruction de l'ensemble X l'ensemble $R_X(Y)$ définie par

$$R_X(Y) = \{x \in X : \exists y \in Y, d_X(x, y) < +\infty\}$$

Autrement dit, la reconstruction de X par Y est l'étalement de Y dans toute la surface X . Comme on peut le constater dans la figure 16, chaque marqueur de Y peut reconstruire fidèlement à l'identique l'ensemble sur lequel il est situé. Cette reconstruction permet d'extraire les composantes connexes X et Z à l'aide des marqueurs Y .

FIGURE 16. Illustration de la reconstruction de X et Z par Y .

On suppose que l'ensemble Y est composé de n composantes connexes $Y_1, \dots, Y_n : Y = Y_1 \cup \dots \cup Y_n$

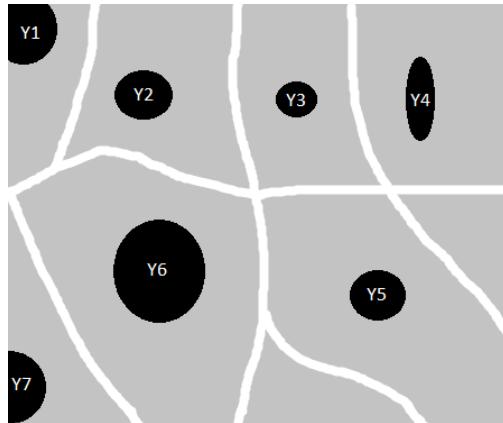
Définition 4. On appelle zone d'influence géodésique de Y_i sur X l'ensemble $z_X(Y_i)$ définie par

$$z_X(Y_i) = \{a \in X : \forall j \neq i, d_X(a, Y_i) < d_X(a, Y_j)\}$$

On note $IZ_X(Y)$ l'ensemble des zones d'influence géodésique $z_X(Y_i)$ pour chaque i :

$$IZ_X(Y) = \bigcup_i z_X(Y_i)$$

Cet ensemble correspond à un espace qui entoure Y_i dans X .

FIGURE 17. Illustration de zone d'influence géodésique (en gris) de 7 composantes connexes de Y .

Comme on peut le voir dans la figure 17, la zone d'influence géodésique (en gris) de Y_i rassemble les éléments qui sont le plus proche de Y_i et le plus éloigné pour les autres composantes connexes. Les éléments qui sont à égale de distance d'au moins deux composantes connexes se trouvent dans le squelette géodésique par zones d'influence.

Définition 5. On appelle squelette géodésique par zones d'influence (*geodesic skeleton by influence zones*), l'ensemble

$$SKIZ_X(Y) = X \setminus IZ_X(Y)$$

L'ensemble $SKIZ_X(Y)$ définit le bord de toutes les zones d'influences $z_X(Y_i)$.

Principe de l'algorithme On s'inspire de [1].

L'algorithme consiste à reconstruire les seuils successifs de f à l'aide des zones d'influence géodésiques.

Pour un petit entier $i_0 \in [\min(f), \max(f)]$, on suppose que $Z_{i_0}(f)$ est non vide c'est à dire que $Z_{i_0}(f)$ détient un ensemble de composantes connexes. Au niveau i_0 , ces composantes connexes forment un minimum régional de f .

Au niveau $i_0 + 1$, on obtient $Z_{i_0}(f) \subset Z_{i_0+1}(f)$. L'ensemble $Z_{i_0+1}(f)$ détient un ensemble de composantes connexes. On pose Z une composante connexe de $Z_{i_0+1}(f)$.

On étudie la relation de Z et les composantes connexes de $Z_{i_0}(f)$ en effectuant une distinction de trois cas sur la position de Z . La figure 18 présente trois cas rencontrés entre Z et les composantes connexes de $Z_{i_0}(f)$.

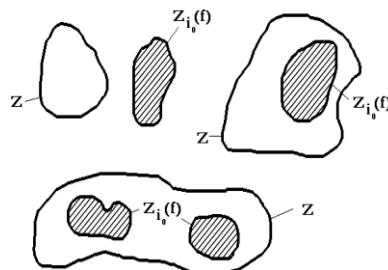


FIGURE 18. Illustration de la relation Z et des composantes connexes de $Z_{i_0}(f)$

1er cas On suppose que $Z_{i_0}(f) \not\subset Z$. On obtient $Z_{i_0}(f) \cap Z = \emptyset$.

Alors, Z est un minimum régional de f au niveau i_0 .

2e cas On suppose que $Z_{i_0}(f) \subset Z$ et $\text{Card}(Z_{i_0}(f) \cap Z) = 1$.

Alors, Z est un minimum régional de f au niveau $i_0 + 1$ issu de $Z_{i_0}(f)$.

3e cas On suppose que $Z_{i_0}(f) \subset Z$ et $\text{Card}(Z_{i_0}(f) \cap Z) \geq 2$.

Alors, Z est la réunion d'au moins une composante connexe de $Z_{i_0}(f) \cap Z$. On utilise la zone d'influence géodésique de $Z_{i_0}(f) \cap Z$ pour éviter de la réunion de $Z_{i_0}(f) \cap Z$. La zone d'influence géodésique appliquée à $Z_{i_0}(f) \cap Z$ permet d'obtenir le squelette géodésique par zone d'influences.

La figure 19 illustre le cas 1 et cas 2.

La figure 20 illustre le cas 3. La figure 20 montre dans 1 que Z contient deux composantes connexes $Z_{i_0}(f) \cap Z$ (1). La zone d'influence géodésique appliquée à $Z_{i_0}(f) \cap Z$ permet de déterminer le squelette géodésique par zone d'influence (2). Au niveau $i_0 + 1$, ces zones d'influences séparées par le squelette géodésique par zone d'influence forment chacune un minimum régional de $Z_{i_0}(f) \cap Z$.

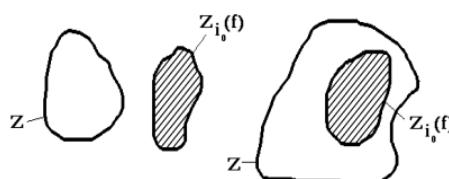


FIGURE 19. Illustration du cas 1 (à gauche) et cas 2 (à droite)

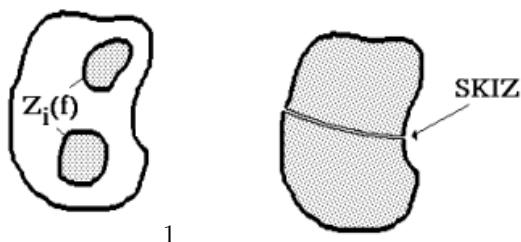
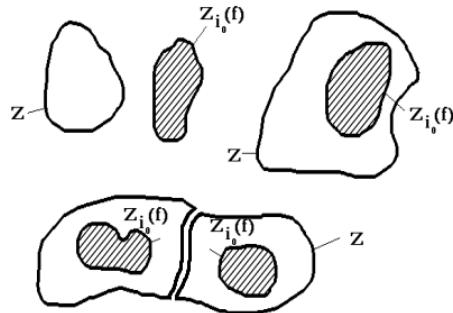


FIGURE 20. Illustration du cas 3.

La figure 21 montre le résultat de l'étude pour les 3 cas.

FIGURE 21. Illustration du résultat de l'étude de la relation de Z et des composantes connexes de $Z_{i_0}(f)$

Au niveau $i_0 + 1$, les composantes connexes de $Z_{i_0+1}(f)$ deviennent des minimum régional. On détermine par cette procédure les composantes connexes de $Z_{i_0+2}(f)$, $Z_{i_0+3}(f)$, ..., $Z_{\max(f)}(f)$.

Au terme de la procédure, le squelette géodésique par zone d'influence définit la ligne de partage des eaux comme le montre la figure 22.

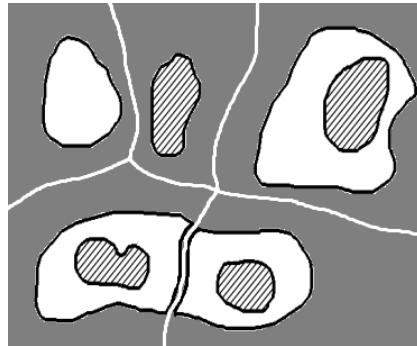


FIGURE 22. Résultat de la procédure de la ligne de partage des eaux

3.3.3 • ALGORITHME DE LIGNE DE PARTAGE DES EAUX

L'algorithme de ligne de partage des eaux est un algorithme de débordement (*flooding algorithm*) de S. Beucher et Ch. Lantuejoul.

On suppose que f est bornée. On pose $N = \max f$

Algorithme. (Ligne de partage des eaux)

ENTRÉE : Les minima de f à la hauteur 0.

SORTIE : Le squelette géodésique par zone d'influence

1. Affecter $m_0(f) \leftarrow$ les minima de f à la hauteur 0
2. Créer un tableau W_0 pour les minima de f
3. Pour un entier i parcourant de 1 à N .
 - (a) Créer $Z_i(f)$ et $Z_{i-1}(f)$
 - (b) Créer $R_{Z_i(f)}(Z_{i-1}(f))$
 - (c) Affecter $m_i(f) \leftarrow Z_i(f)/R_{Z_i(f)}(Z_{i-1}(f))$
 - (d) Créer $SKIZ_{Z_i(f)(W_{i-1})}$

(e) Affecter $W_i \leftarrow SKIZ_{Z_i(f)(W_{i-1})} \cup m_i(f)$

4. Renvoyer W_N^c

La quantité $m_i(f)$ est l'ensemble des minimum régionaux au niveau i . Elle est déterminé par la différence entre le niveau i et la reconstruction géodésique au niveau $i - 1$.

A la ligne 6, on définit le squelette géodésique par zones d'influence sur W_{i-1} .

3.3.4 • APPLICATION

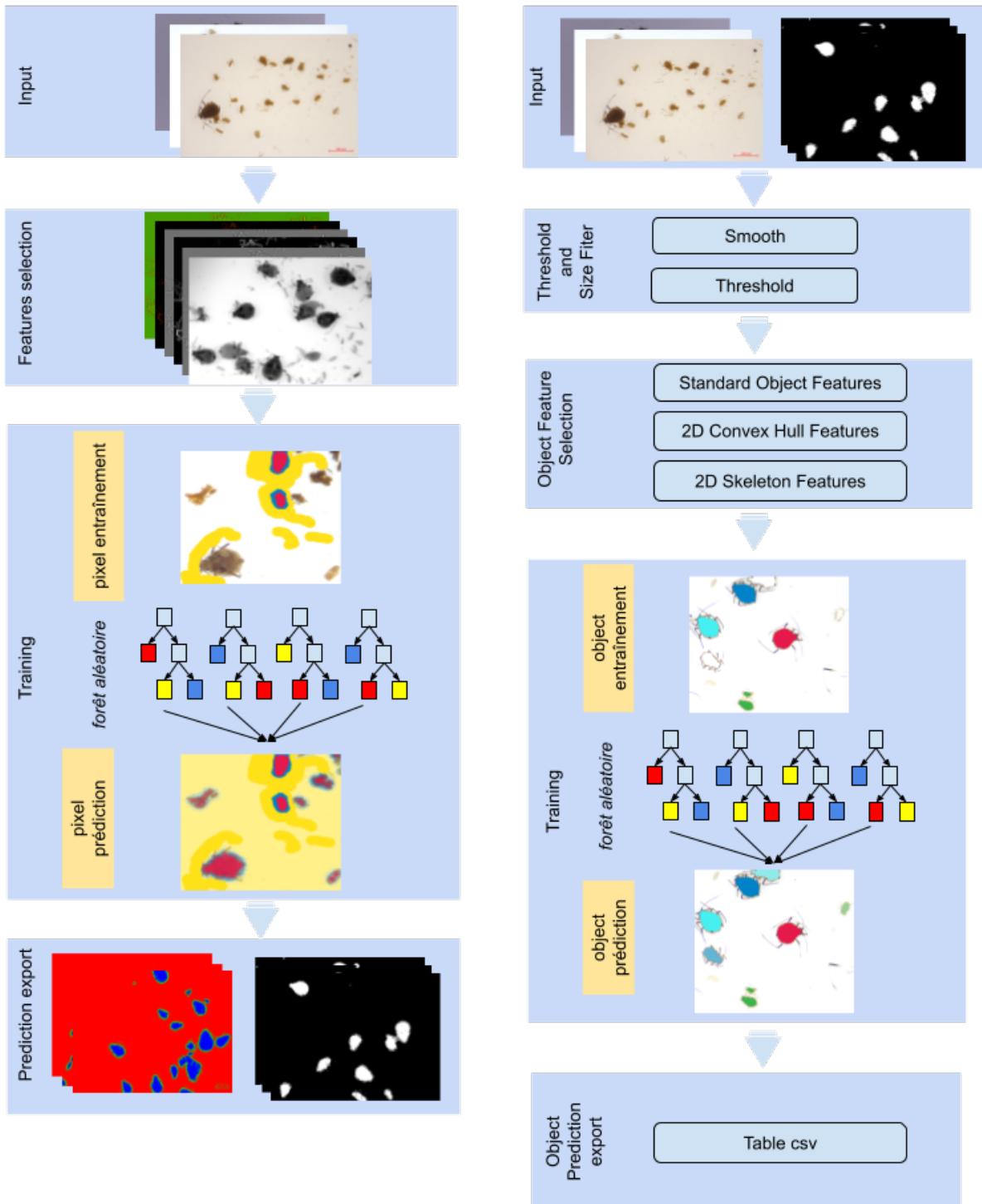
Certaines images du projet ECLECTIC présentent des pucerons qui se chevauchent. Ce chevauchement rend la qualité de la segmentation très faible. Ce problème de chevauchement des objets peut être surmonté par l'algorithme de ligne en partage des eaux.

On applique au *map prediction* de la sortie Ilastik, la segmentation par ligne de partage des eaux à l'aide de Imagej. Comme le montre la figure 22, (2) présente la mauvaise qualité de la segmentation sortie par Ilastik. On applique l'algorithme de segmentation par ligne de partage des eaux. Avec Imagej, on applique la fonction *Watershed* au *map prediction* sorti de Ilastik qu'on binarisée au préalable avec la fonction *Make binary* de Imagej. Le résultat de l'algorithme par ligne de partage des eaux est donné par la sortie de *Watershed* de Imagej (3).



FIGURE 23. Traitement du chevauchement d'objets dans l'image snap-178 du projet ECLECTIC.

3.4 RÉSUMÉ DU TRAITEMENT D'IMAGE SOUS ILASTIK



1. Classification de pixel

2. Classification d'objet

FIGURE 24. Représentation schématique de la classification de pixel et classification d'objet sous Ilastik.

4

MÉTHODOLOGIE

4.1 MATÉRIEL

Dans ce projet, nous faisons appel à deux outils informatiques pour résoudre le problème de chevauchement d'objet, et effectuer la segmentation d'images et la classification d'objets.

La segmentation d'images et la classification d'objets sont réalisées sur *Ilastik* [5]. *Ilastik* est un logiciel interactif dédié à l'analyse d'images. Son fonctionnement repose sur la notion de *shallow learning*. Essentiellement, *shallow learning* est un type d'apprentissage machine où l'apprentissage s'applique sur des *features* prédéfinies à partir des données brutes. Comme le montre la figure 25, on définit des *features* à partir des données brutes. Sur ces *features*, on applique un algorithme qui permet d'obtenir un modèle. Cette notion a la qualité d'éviter le sur-apprentissage en définissant quelques données en entrée.

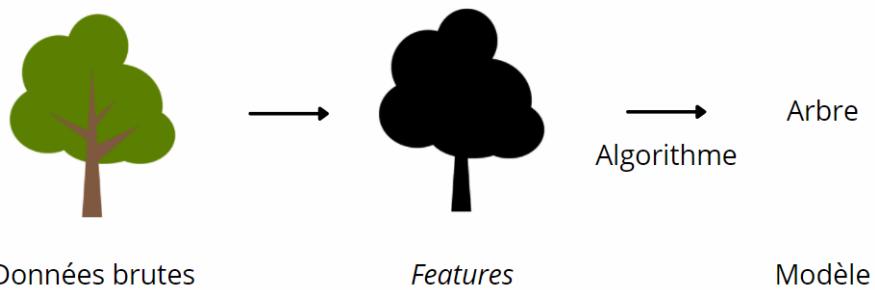


FIGURE 25. Illustration simple du fonctionnement du *shallow learning*

Le traitement du chevauchement d'objets est réalisé sur *Imagej* [7]. *Imagej* est un logiciel interactif de traitement et d'analyse d'images. Il propose de nombreux types de traitement et d'analyse d'image grâce aux algorithmes directement implantés dans le logiciel. Dans le traitement d'image, on peut citer la mesure d'un profil d'intensité lumineuse dans une image, la détection de contour, la binarisation d'une image par exemple. Dans le cadre de notre projet, on s'intéresse à l'algorithme de ligne de partage des eaux.

4.2 SEGMENTATION DES IMAGES

La segmentation est réalisée dans le *workflow* "pixel classification" sous *Ilastik*. En pré-traitement d'image, la sélection de *features* est basée sur l'intensité de couleur, la texture, le contour d'objet. Sur *Ilastik*, ce pré-traitement inclus la sélection de différents niveaux d'échelles (σ) pour chaque *feature*. La sélection de *features* permet d'obtenir des images filtrées (Gaussien, Laplace, Gradient Gaussien). Chaque échelle (σ) permet de mettre en évidence les détails des surfaces à différents niveaux. Par exemple, les σ grands rendent des filtres mettant en évidence des surfaces plus grandes ; alors que les σ petits rendent des filtres avec des surfaces plus détaillées comme le montre la figure 26.

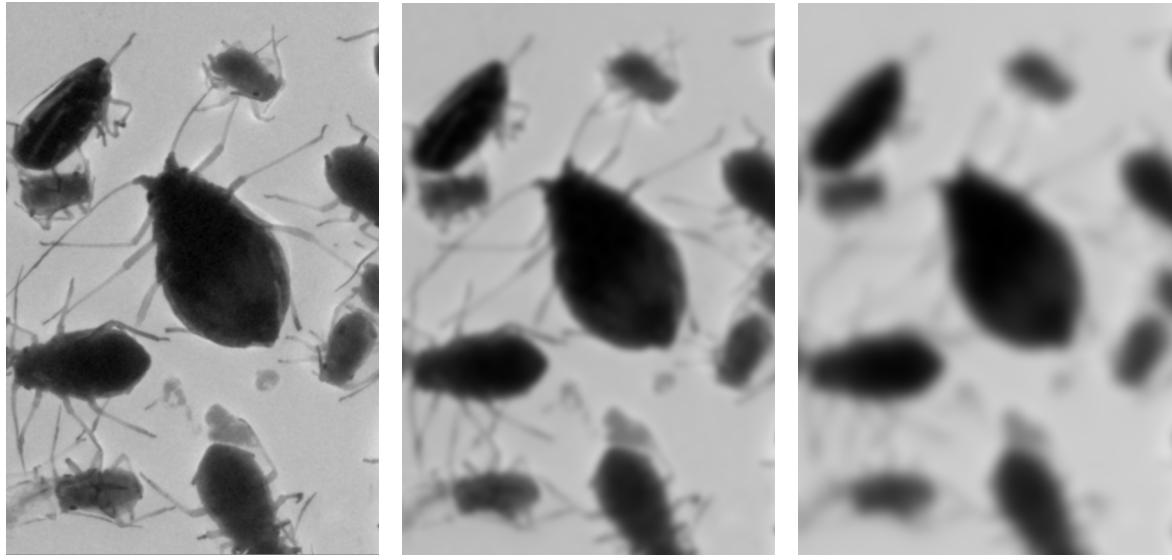


FIGURE 26. Résultats de différents sigma σ sous le filtre Gaussien dans une partie de l'image aphid-120-5 sous Ilastik.

En segmentation d'image, on définit trois classes de segmentation : background (arrière-plan), edge (bord) et aphid (puceron). La phase d'entraînement consiste à annoter les surfaces de différentes régions de l'image. La phase de prédiction consiste à prédire à chaque pixel une classe par la méthode forêt aléatoire. La figure 27 présente l'annotation des 3 couleurs (à gauche) et la prédiction des autres pixels (à droite).

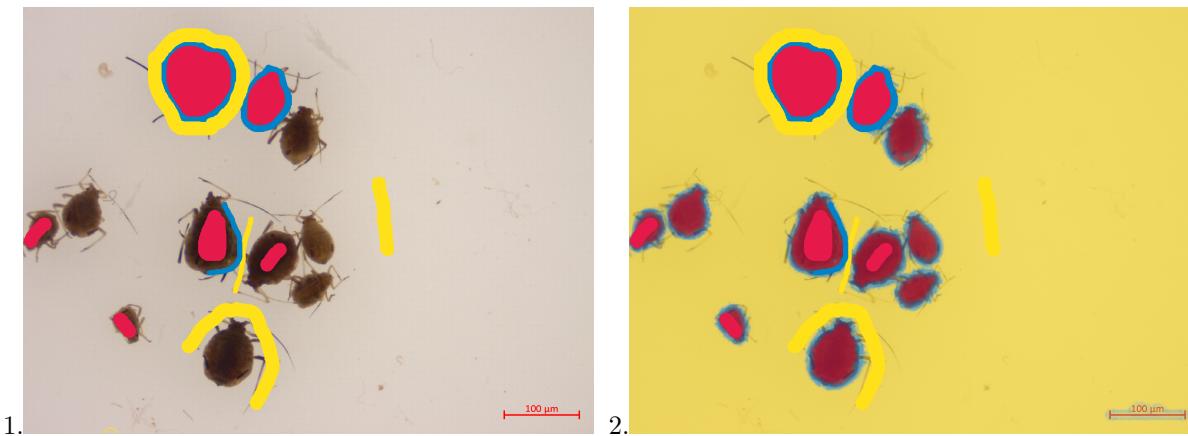


FIGURE 27. *Pixel classification.*

1. Annotation du puceron (rouge), de son contour (bleu) et de l'arrière plan (jaune)
2. Résultat de la prédiction de classes.

Le dataset a été séparé en 3 ensembles d'images. Les images prises à l'échelle $200\mu m$, $100\mu m$ et $20\mu m$ se distinguent par leur luminosité. On a effectué trois projets associés aux images à échelles $200\mu m$, $100\mu m$ et $20\mu m$. À chaque projet, on constitue un jeu d'entraînement en annotant une classe à chaque pixel ; puis on détermine les classes pour chaque pixel à tous les images de l'échantillon test avec la fonction *Batch processing*.

TABLEAU 3. Nombre d'images constitué dans l'échantillon d'entraînement et test pour chaque projet.

Echelle	Echantillon d'entraînement	Echantillon test
$20\mu m$	27	318
$100\mu m$	5	206
$200\mu m$	3	23

Le *Simple segmentation* et le *probability map* ont été exportés pour chaque image.

4.3 TRAITEMENT DU CHEVAUCHEMENT EN UTILISANT L'ALGORITHME DE LIGNE DE PARTAGES DES EAUX

Le traitement du chevauchement est réalisé dans la rubrique watershed sous Fiji [7].

Parmi les images traité dans le pixel classification, 43 images présentent des objets qui se chevauchent. On traite ce problème de chevauchement en suivant la procédure ci-dessous.

On prends en entrée le *probability map* où se trouve un chevauchement d'objets et on obtient une image segmenté à la fin du traitement.

Le traitement est composé en 4 étapes :

1. Binariser le *probability map* avec la fonction *Make binary*.
2. Remplissage des trous de la couleur de l'arrière plan avec la fonction *Fill holes*
3. Appliquer l'algorithme de partage des eaux avec la fonction *Watershed*
4. Exporter l'image avec la fonction *Export hdf5*

4.4 NOMENCLATURE DE L'ÉVOLUTION DES PUCERONS POUR LA CLASSIFICATION D'OBJETS

Chaque image présente les pucerons à différentes étapes de son développement. On identifie six classes qui sont les six étapes d'évolution du puceron :

- *Winged adult*
- *Apterous adult*
- *Nymph*
- *Larvae*
- *Small larvae/nymph*
- *Molt*

La figure 28 présente les 6 stades développement du puceron.



FIGURE 28. Différentes étapes d'évolution du pucerons cendré du pommier 1 – 6.

La morphologie des pucerons à chaque étape dans leur cycle de vie est importante pour l'identification visuelle dans le cadre de l'attribution des classes pour la partie classification d'objet.

Le *Winged adult* et *Apterous adult* ont un corps le plus large de la colonie. Le corps du *Winged adult* est de forme ellipsoïdale et le corps du *Apterous adult* est de la forme d'une goutte d'eau.

Le *Nymph* et *Larvae* ont un corps de taille moyen. Le corps du *Nymph* est de forme ellipsoïdale et le corps du *Larvae* est de la forme d'une goutte d'eau.

Les *Small larvae/nymph* ont le plus petit corps de la colonie. Leur morphologie est ellipsoïdale pour *Small larvae* et la forme d'une goutte d'eau pour *Small nymph*.

Enfin, *Molt* se caractérise seulement par une morphologie de forme irrégulière.

4.5 CLASSIFICATION D'OBJETS

La classification d'objets est réalisée dans le *workflow object classification* sur Ilastik. La procédure du classification d'objet contient 4 microprogrammes. En entrée de la procédure, on insère l'image et son masque de prédiction obtenu à l'issue du Pixel classification. Le tableau 4 indique le nombre d'image dans l'échantillon d'entraînement et l'échantillon test.

TABLEAU 4. Nombre d'images dans l'échantillon d'entraînement et l'échantillon test pour la classification d'objet

Echantillon	Echantillon d'entraînement	Echantillon test
Nombre d'images	22	351

Après avoir effectué le pré-traitement des images, on sélectionne tous les *features* disponibles des objets incluant l'enveloppe convexe et la taille en pixel de chaque objet par exemple.

Ensuite, on conçoit un jeu d'entraînement pour effectuer la prédiction des objets. On fabrique le jeu d'entraînement en annotant pour certains objets la classe qui lui est associée. On s'appuie sur la nomenclature de la section 4.4.

Le jeu test est constitué des objets qui n'ont pas été annotés. La prédiction de la classe pour chaque objet du jeu test est réalisé avec la méthode des forêts aléatoire avec la fonction *Live Test*. Cette fonction nous permet d'apercevoir la qualité du jeu d'entraînement. Nous pouvons corriger les défauts de la prédiction en concevant à nouveau un jeu d'entraînement et réalisé une prédiction des objets.

Lorsque la prédiction des objets est satisfaisante, on peut appliquer cette procédure à tous les images du jeu test avec la fonction *Batch Processing*.

A l'issue de la procédure, un tableau contenant 116 variables est exporté pour chaque image. Ce tableau donne une description les propriétés statistiques de l'intensité et la morphologie pour chaque objet détecté dans l'image.

4.6 MÉTHODE DE VALIDATION DU MODÈLE

La méthode de validation vise à déterminer la qualité moyenne de la prédiction d'un modèle [4]. La méthode consiste à appliquer le modèle à plusieurs sous-échantillons de l'échantillon d'entraînement et de calculer la moyenne des performances pour chaque sous-échantillon. Ces sous-échantillons sont fabriqués à partir d'un découpage aléatoire de l'échantillon d'entraînement comme le montre la figure 29.



FIGURE 29. Quelques méthodes de découpage d'un échantillon pour la validation

Nous choisissons la méthode *K-Fold* qui consiste à fabriquer K sous-échantillons et former un échantillon de validation d'un sous-échantillon différent des autres sous-échantillons. Par exemple dans le cas $K = 3$, si l'échantillon de validation est constitué de 30% de l'échantillon, cette constitution de l'échantillon de validation ne se répétera pas dans les 2 autres sous-échantillons comme le montre la figure 29.

Notre échantillon d'entraînement est composé de 249 images. Sur cet échantillon d'entraînement, on conçoit 10 sous-échantillons. Le tableau 5 montre le nombre d'images qui constituent l'échantillon d'entraînement et l'échantillon validation.

TABLEAU 5. Composition de l'échantillon d'entraînement et l'échantillon validation pour chaque sous-échantillon.

K	Echantillon d'entraînement	Echantillon validation
1	25	224
2	25	224
3	25	224
4	25	224
5	25	224
6	25	224
7	25	224
8	25	224
9	25	224
10	24	225

En classification supervisée, différentes mesures de performance sont utilisées pour évaluer la qualité de prédiction du modèle. Par exemple, on peut citer l'exactitude (ou *Accuracy*) qui est le rapport entre le nombre d'objets correctement prédits et le nombre d'objet total.

Nous choisissons la matrice de confusion pour sa qualité à s'adapter à un problème de classification à 6 classes. Cette matrice nous permet de déterminer des mesures pour chaque classe à partir de quatre tests : vrai positif, faux positif, vrai négatif, faux négatif. Ces mesures sont la spécificité, le rappel, la précision et Exactitude.

Pour obtenir la qualité moyenne de la prédiction du modèle, on calcule la moyenne de Exactitude donnée par chaque matrice de confusion obtenue à partir de la validation croisée *K - Fold*.

5

RÉSULTAT

La performance de la méthode des forêts aléatoires est 0.81. Plusieurs informations intéressantes ressortent du tableau 6.

Tout d'abord, la grande valeur de Precision des classes *Nymph/Larvae small, Molt* montre que le modèle prédit correctement ces classes. En revanche, la qualité du modèle pour la prédition de la classe *Larvae* est très faible puisque seulement 1.7% de la classe prédictive sont correctes. Quant aux classes *Nymph* et *Winged adult*, le modèle arrive à prédire correctement presque plus de 50% des objets comme le montre leur score Rappel. Parmi les prédictions de la classe *Winged adult*, 77% d'entre eux sont correctes.

Enfin, l'échantillon d'entraînement s'adapte bien au modèle pour les classes *Nymph/Larvae small, Molt, Nymph, Winged adult* et *Apterous adult* car ces classes détiennent un Rappel au-dessus de 0.5. En revanche, le modèle arrive moins à discriminer les objets de la classe *Larvae* au vu du score Rappel très faible.

TABLEAU 6. Matrice de confusion et mesures de chaque classe

Classe	Nymph/Larvae small	Molt	Nymph	Winged adult	Apterous adult	Larvae
Nymph/Larvae small	5079	147	192	24	15	90
Molt	394	2195	4	5	2	3
Nymph	144	0	451	129	57	46
Winged adult	9	0	41	304	36	4
Apterous adult	18	0	52	86	229	21
Larvae	64	48	59	11	10	34
Exactitude	0.89	0.93	0.92	0.96	0.97	0.96
Precision	0.91	0.84	0.54	0.77	0.56	0.15
Rappel	0.88	0.91	0.56	0.54	0.65	0.17
Spécificité	0.89	0.94	0.95	0.99	0.98	0.98
F1 score	0.89	0.93	0.71	0.70	0.78	0.29

En ligne, les classes vraies ; en colonne, les classes prédictives

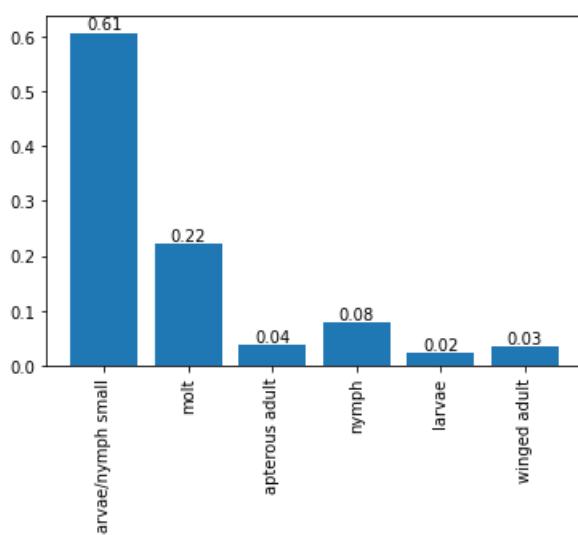


FIGURE 30. Distribution normalisée de l'échantillon d'entraînement par classe

De plus, la distribution de l'échantillon d'entraînement est déséquilibrée comme le montre la figure 30. Le

nombre d'objet de la classe *Larvae* est très faibles par rapport au reste des classes. Nous notons aussi à l'inverse que les objets de la classe *Larvae/Nymph small* sont très nombreux. On peut déduire une surestimation de leur score de Precision.

6 DISCUSSION

On rappel que ce projet vise à proposer une méthode objective de dénombrement de pucerons et leur stade de développement. Notre méthode porte sur les notions de traitement d'images à savoir la segmentation d'images et la classification d'objets. Nous avons réalisé la segmentation d'images en utilisant la méthode des forêts aléatoires puis l'algorithme de ligne de partage des eaux pour résoudre le problème de chevauchement d'objets. Puis, nous avons réalisé la classification d'objets en utilisant la méthode des forêts aléatoires.

La section 5. présente la performance de la méthode des forêts aléatoires pour la classification d'objet. La qualité de la prédiction des classes *Nymph/Larvae small*, *Molt*, *Nymph*, *Winged adult* et *Apterous adult* est bonne, bien que le score Precision des classes *Nymph* et *Apterous adult* reste très juste (proche 0.5) d'après le tableau 6.

Les classes *Nymph/Larvae small*, *Molt*, *Nymph*, *Winged adult* et *Apterous adult* sont d'excellent *classifier* à la lecture de leur score Precision, Rappel et Spécificité.

Par ailleurs, le score de Precision et la distribution de la classe *Larvae* et *Larvae/Nymph small* sont intéressants.

Tout d'abord, le nombre d'objets dans la classe *Larvae* est très faible dans l'échantillon d'entraînement et les objets dans la classe *Larvae/Nymph small* sont les plus nombreux. Par conséquent, il semble que le score Precision des classes *Larvae* et *Larvae/Nymph small* sont surestimés par ce déséquilibre.

Aussi, on observe d'une part que les scores très faibles des mesures Precision, Rappel et Spécificité de la classe *Larvae* suggèrent que *Larvae* est un mauvais *classifier*. D'autre part, les scores très forts des mesures Precision, Rappel et Spécificité de la classe *Larvae/Nymph small* suggèrent être un excellent classifier si on ne prends pas en compte sa forte distribution.

Le modèle semble être très sensible à la distribution d'une classe déséquilibrée. Le problème du déséquilibre de la distribution peut être résolu en effectuant deux corrections. D'une part, la solution serait d'augmenter le nombre d'objet de classe *Larvae* dans l'échantillon d'entraînement. D'autre part, la solution serait de réduire le nombre d'objet de la classe qui apparaît le plus fréquemment à savoir la classe *larvae/nymph small*.

7 CONCLUSION

Bien que le modèle des forêts aléatoires assure une qualité de prédiction de 81%, il a une difficulté à prédire les objets de la classe *Larvae*. Le modèle ne semble pas être optimale pour une classe *Larvae*.

Deux approches pourraient être exploitées.

Tout d'abord, on peut tenter de réduire le nombre d'objets pour la classe *larvae/nymph small* et augmenter le nombre de d'objets pour la classe *Larvae* pour améliorer le score de ces classes.

Enfin, une nouvelle approche sera d'exploiter la théorie du Deep Learning tel que les réseaux de neurones [10].

Toutefois, si on souhaite appliquer d'autres images qui n'ont pas appartenu au jeu d'entraînement sur notre modèle alors on doit s'attendre à ce que notre modèle prédit correctement les classes *larvae/nymph small, Molt, Nymph, Winged adult et Apterous adult*.

8 **GLOSSAIRE**

Terme	Définition
Bootstrap	Algorithme de rééchantillonage du jeu de données brute réalisé par un tirage aléatoire avec remise
Surapprentissage	Qualité du modèle parfaite liée à un apprentissage excessif de l'échantillon d'entraînement par le modèle

RÉFÉRENCES

- [1] Serge BEUCHER et Fernand MEYER. “The morphological approach to segmentation : the watershed transformation”. In : *Mathematical morphology in image processing* 34 (1993), p. 433-481.
- [2] Tin Kam Ho. “Random decision forests”. In : *Proceedings of 3rd international conference on document analysis and recognition*. T. 1. IEEE. 1995, p. 278-282.
- [3] Leo BREIMAN. “Random forests”. In : *Machine learning* 45.1 (2001), p. 5-32.
- [4] Sylvain ARLOT et Alain CELISSE. “A survey of cross-validation procedures for model selection”. In : *Statistics surveys* 4 (2010), p. 40-79.
- [5] Christoph SOMMER et al. “Ilastik : Interactive learning and segmentation toolkit”. In : *2011 IEEE international symposium on biomedical imaging : From nano to macro*. IEEE. 2011, p. 230-233.
- [6] Juergen GALL, Nima RAZAVI et Luc Van GOOL. “An introduction to random forests for multi-class object detection”. In : *Outdoor and large-scale real-world scene analysis*. Springer, 2012, p. 243-263.
- [7] David LEGLAND, Ignacio ARGANDA-CARRERAS et Philippe ANDREY. “MorphoLibJ : integrated library and plugins for mathematical morphology with ImageJ”. In : *Bioinformatics* 32.22 (2016), p. 3532-3534.
- [8] Gaetan BISSON. *Lecture notes in Informatique*. 2017.
- [9] Stuart BERG et al. “Ilastik : interactive machine learning for (bio) image analysis”. In : *Nature Methods* 16.12 (2019), p. 1226-1232.
- [10] Hamzan WADI. *Step By Step Neural Networks for Image Classification using Python GUI : A practical approach to understand the neural networks algorithm for image classification with project based example*. Turida Publisher.