MÉMOIRE :

Conception de solutions web de gestions et d'identification d'espaces de reptiles exotiques envahissants dans les Petites Antilles à partir de leur image numérique

Stéphane Bance

M2 Informatique

Responsabilité et encadrement du stage :

M. Jimmy Nagau
M. Emmanuel Biabiany

Responsable pédagogique :

M. Emmanuel Biabiany

Du 4 Janvier au 16 Juin 2022

Table des matières

1 Introduction **1**

2 Présentation de la structure d'accueil **2**
2.1 Université des Antilles . . . 2
2.1.1 Administration et personnels techniques . . . 2
2.1.2 Enseignements . . . 2
2.1.3 Recherche . . . 2
2.2 Le LAMIA . . . 2

3 Outils et documentation **3**
3.1 Python . . . 3
3.1.1 OpenCV . . . 3
3.1.2 PIL . . . 3
3.1.3 Panda . . . 3
3.1.4 PixelLib . . . 3
3.2 Weka . . . 3
3.3 html,css et js . . . 3
3.4 PWA . . . 3
3.5 État de l'art . . . 4
3.6 jeux de données fournis par les experts . . . 4

4 Processus d'extraction des formes pertinentes **5**
4.1 Réduction de la résolution . . . 5
4.2 Extraction des couleurs prédominantes . . . 5
4.3 Elimination des composantes connexes . . . 6
4.3.1 Composantes au bord . . . 6
4.3.2 Petites composantes connexes . . . 7
4.3.3 Composantes dans zones intermédiaires . . . 8
4.4 Fermeture . . . 8
4.5 Rectangles englobants . . . 9
4.6 Pistes abandonnées : . . . 10
4.6.1 Les réseaux de neurones . . . 10

5 Extraction de caractéristiques **13**
5.1 Extraction de la couleur . . . 13
5.1.1 Extraction RGB . . . 13
5.1.2 Extraction HSV . . . 14
5.2 Extraction de la texture . . . 15
5.2.1 LBP . . . 15
5.2.2 Application du LBP . . . 18
5.2.3 Taux homogénéité . . . 21
5.3 Dimensions de l'image . . . 21

6 Identification **22**
6.1 Agencement des données . . . 22
6.2 Résultat de la classification . . . 22
6.3 Bilan de la classification . . . 23

7 Conception de l'application **24**
7.1 Architecture de l'application . . . 24
7.2 Présentation de l'interface . . . 24

8 Conclusion **27**
8.1 Bilan des travaux 27
8.2 Analyse personnelle 27
Bibliographie28

Table des figures

1 Image avant traitement . . . 5
2 Image après traitement . . . 5
3 Image avant traitement . . . 6
4 Image avant traitement . . . 6
5 Image après traitement . . . 6
6 Image avec différentes valeur de k . . . 7
7 Image avec différentes valeur de k . . . 7
8 Effacement des composantes ayant moins de 500 pixels . . . 7
9 Exemples de lézard perdu . . . 8
10 Image avant et après traitement . . . 8
12 Effacement des composantes ayant moins de 500 pixels . . . 9
16 Résultat avec le modèle "deeplabv3_xception65_ade20k.h5" . . . 11
18 Résultat avec le modèle "deeplabv3_xception65_tf_dim_ordering_tf_kernels.h5" 11
20 Résultat avec le modèle "mask_rcnn_coco.h5" . . . 12
21 Gecko extrait quelconque . . . 13
22 Histogrammes couleurs du gecko . . . 13
23 Représentation du modèle HSV . . . 14
24 Histogrammes Teinte du gecko . . . 15
25 Schéma application du LBP . . . 16
26 Tableau des 36 paternes principaux . . . 16
27 Identifier à quels paternes principaux appartient le paterne . . . 17
28 code couleur . . . 18
29 Premier rendu du traitement LBP . . . 18
30 Rendu du traitement LBP avec filtre median . . . 19
31 Rendu du traitement LBP avec filtre median et seuil à 128 . . . 19
32 Rendu du traitement LBP avec filtre median et seuil à 64 . . . 19
33 Rendu du traitement LBP avec filtre median et seuil à 32 . . . 19
34 Rendu du traitement LBP avec filtre median et seuil à 16 . . . 19
35 Rendu du traitement LBP avec filtre median et seuil à 8 . . . 20
36 Rendu du traitement LBP avec filtre median et seuil à 4 . . . 20
37 Exclusion des paternes du fond et de bruit . . . 20
39 Image de gecko nain en fin de traitement . . . 20
41 Image de mabouia en fin de traitement . . . 21
42 Visualisation des 5 premières lignes des données . . . 22
43 Tableau de précision par classe de random tree . . . 22
44 Tableau de précision par classe de random forest . . . 22
45 Tableau de précision par classe de J48 . . . 23
46 Schéma fonctionnel du fonctionnement interne de l'application . . . 24
47 logo gecko+ . . . 24
48 Application page actualité . . . 25
49 Application page photo . . . 25
50 Application page actualité . . . 25
51 Application page actualité . . . 26

1 Introduction

Le traitement des animaux importés occupe un volume important dans le travail des douaniers. La diversité des espèces évaluées peut rendre délicates les procédures administratives dans le cadre des catégorisations. Parmi les espèces traitées figurent des reptiles exotiques envahissants qui peuvent constituer un facteur impactant pour la faune ou la flore s'il se retrouve en liberté. Afin de bien identifier et évaluer les risques que pourraient constituer les reptiles traités, nous souhaitons mettre en place un outil d'accompagnement des douaniers dans leur mission de protection de leur territoire. Au travers d'une application légère, il serait possible de fournir des informations précises sur une espèce dont on disposerait de sa photographie numérique.

Ce stage a pour but la mise en œuvre d'une chaîne de traitement qui permet l'identification automatique de deux types de lézards, le Lepidodactylus lugubris (le gecko nain) et le Hemidactylus mabouia. Nous dresserons dans un premier temps, un état de l'art des éléments permettant l'évaluation des reptiles. Dans un second temps à partir de ces éléments, nous souhaitons constituer, à partir des méthodes déjà mises en place, mais qui n'exploite pas suffisamment le domaine du numérique en vue d'une simplicité d'usage, à étudier des processus en traitement d'image qui permettrait la caractérisation des espèces en minimisant les distances intraclasses tout en maximisant les distances interclasses.<
Le processus visé se réalisera de la façon suivante :
Premièrement une capture de l'image avec un appareil doté d'un capteur photo(portable, tablette)
Exporter les données vers un serveur permettant la caractérisation et la mise en place du système d'information pour la gestion statistique et l'apprentissage des espèces.
Rapatrier vers le dispositif de l'utilisateur les éléments permettant d'identifier l'espèce du reptile soumis en lui apportant de l'infirmation riche sur ce dernier.

Dans ce mémoire, je décrirai d'abord la structure d'accueil ensuite je présenterai les outils et la documentation qui ont servi au projet, puis les méthodes utilisées pour extraire les zones d'intérêt ainsi que leurs caractéristiques afin de présenter notre étude pour la classification, ensuite, je montrerai l'application mise en œuvre dans le cadre de ce travail et enfin une conclusion.

2 Présentation de la structure d'accueil

Durant mon stage, j'ai été accueilli au **Laboratoire de Mathématiques Informatique et application (LAMIA)** de l'Université des Antilles (UA).

2.1 Université des Antilles

L'Université des Antilles s'organise autour de deux pôles universitaires régionaux autonomes : le Pôle Guadeloupe et le Pôle Martinique.

Sur ses deux pôles, l'Université assure des missions d'*enseignement* et de *recherche*, assistées par des *administratifs et des techniciens*.

2.1.1 Administration et personnels techniques

l'UA emploie 414 personnels administratifs, ingénieurs et techniciens (environ 200 agents pour l'administration centrale et 100 répartis sur chaque pôle)

2.1.2 Enseignements

L'UA délivre des diplômes de la Licence au Doctorat en passant par le Master dans de nombreux domaines (Sciences, Economie, Lettres, Sciences Humaines). Au total, cela représente :

— 484 enseignants-chercheurs (environ 240 pour chaque pôle)

— 12 000 étudiants (environ 7000 pour la Guadeloupe , 5000 pour la Martinique)

En informatique, cela représente, pour environ 20 enseignants-chercheurs, autour de 200 étudiants

2.1.3 Recherche

La recherche est structurée en laboratoires auxquels sont rattachés les enseignants chercheurs qui peuvent être amenés à, notamment, former de futurs chercheurs : les doctorants.

L'université compte ainsi au total :

— 17 laboratoires

— 320 doctorants

Pour ma part, j'ai effectué mon stage dans le laboratoire LAMIA que je vais maintenant présenter.

2.2 Le LAMIA

Le **Laboratoire de Mathématiques Informatique et Application (LAMIA)**, comme son nom l'indique, se concentre sur la recherche en informatique et en mathématiques.

Il compte une soixantaine de membres (Professeurs des Universités, Maitres de Conférences, ATER, Doctorants) répartis sur deux pôles (Guadeloupe et Martinique).

3 Outils et documentation

Au cours de mon stage, j'ai utilisé plusieurs logiciels qui m'ont donné la possibilité de traiter les images. d'extraire des caractéristiques et étudier les possibilités de classification de leur contenu.

3.1 Python

Python est un langage de programmation développé actuellement par la Python Software Foundation C'est avec celui-ci et ses diverses bibliothèques que j'ai pu implémenter les algorithmes qui ont servi durant ce stage.

3.1.1 OpenCV

OpenCV (pour Open Computer Vision) est une bibliothèque libre, développé par Intel spécialisé dans le traitement d'image. C'est avec son aide qu'est réalisée la majorité des opérations usuelles en traitement d'image.

3.1.2 PIL

PIL (pour Python Imaging Library) est aussi une bibliothèque libre de traitement d'image. Elle sert dans le traitement de suppression des couleurs dominantes.

3.1.3 Panda

Panda est une bibliothèque permettant la manipulation et l'analyse des données, développé par Wes McKinney. C'est avec celle ci que sont fait les jeux de données qui serviront dans les phases de test d'itentification des lézards.

3.1.4 PixelLib

PixelLib est une bibliothèque développée par Ayoolaolafenwa faite pour effectuer de la segmentation d'image et de vidéo via les réseaux de neurones. Celle-ci a servi dans une des idées d'extraction de lézards qui a été abandonnée.

3.2 Weka

Weka (pour Waikato envoronment for knowledge analysis en Français environnement Waikato pour l'analyse de connaissances) est une suite de logiciels d'apprentissage automatique développé par L'université de Waikato. Elle permet d'effectuer les tests d'identification entre les deux types de lézards.

3.3 html,css et js

Html (HyperText Markup Language), css (cascading style sheet) et JS (JavaScript) sont tous les 3 des langages servant dans la création de sites et d'applications web. Chacun ayant un but différent, html va créer la structure des pages, css modifie l'apparence du contenu et JS programme le comportement de la page. Dans le stage ils ont servi à créer l'application.

3.4 PWA

La technologie PWA (Progressive web app) permet de créer des applications Web multiplateformes, donnant à l'utilisateur les mêmes avantages qu'une application multiplateforme.

3.5 État de l'art

Avant de commencer les travaux, il est important de faire un état de l'art, pour savoir ce qui se fait actuellement, s'en inspirer. Une des premières documentations étudiées est une étude comparative des méthodes de classifications sur des images de serpents[5], elle différencie les méthodes holistiques (c'est à dire traditionnelles, tel que les k plus proches voisins, machine à vecteur de support,...) et la méthode des réseaux de neurones. Les méthodes holistiques présentaient de bons résultats mais celles avec les réseaux de neurones donnaient les meilleurs résultats. Une autre étude sur les serpents des Galàpagos[3] et une sur les lézards[4]. Toutes ensembles elles proposent des solutions au manque de données comme l'augmentation artificiel des images(rotation d'image, de-texturer et d'autre opération qui pour but de modifier les images pour les ajouter au jeu de données) ou utiliser du pre-learning (entrainer un modèle avec un autre jeu de données) et le dropout dans le cas des réseaux de neurones.

3.6 jeux de données fournis par les experts

Pour ce stage l'expert biologiste nous a fourni 360 photos de geckos nains et 136 photos de mabouias, ces images ne sont soumises à aucun protocole, il s'agit juste de lézards pris dans leur milieu naturel.

4 Processus d'extraction des formes pertinentes

4.1 Réduction de la résolution

Les images fournies possèdent souvent des résolutions au-delà de 3000 pixels de côté, ce qui joue sur le temps d'exécution des programmes, plus une image est grande plus le temps de traitement est long car les opérations sont effectuées pixel par pixel voire pour certains chaque pixel et leurs voisinages. Pour réduire ce temps d'exécution j'applique une réduction à l'aide de la méthode du médian vectoriel qui n'inclue pas de nouvelle couleur. Les images sont réduites d'un facteur de 3, jusqu'a une de leurs dimensions soi en dessous de 600 pixels.
La méthode du médian vectoriel s'effectue de la façon suivante :

1. Parcours de l'image pixel par pixel ;
2. Calculer les distances euclidiennes entre le pixel à analyser et son voisinage le plus proche ;
3. Ordonner de façons croissantes ces distances ;
4. La valeur de pixel retenu parmi les 8 voisins est celle dont la distance euclidienne entre le pixel central et le voisin est la distance médiane ;

4.2 Extraction des couleurs prédominantes

La méthode suivante permet de mettre en noir les pixels dont la couleur apparait le plus fréquemment avec un seuil placé à 5e-5, tout cela ayant pour objectif de faire ressortir les formes pertinentes.
On applique premièrement un filtre gaussien pour réduire le bruit puis on crée une matrice de 256x256x256 initialiser à 0, les coordonnées de cette matrice représentent les valeurs des composantes RVB (Rouge vert bleu) du pixel et ses valeurs représentent le nombre de pixels ayant ces valeurs RVB.
L'image est ensuite parcourue et la matrice est incrémentée aux coordonnées correspondantes, elle est ensuite de nouveau parcourue mais cette fois toutes les couleurs qui dépassent le seuil dans la matrice seront mises en noir.

FIGURE 1 – Image avant traitement

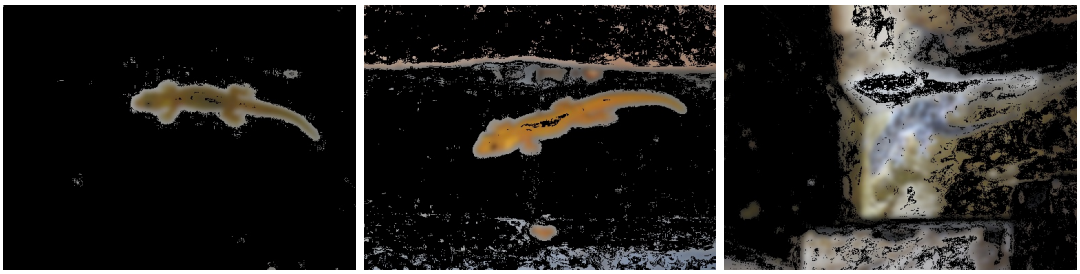


FIGURE 2 – Image après traitement

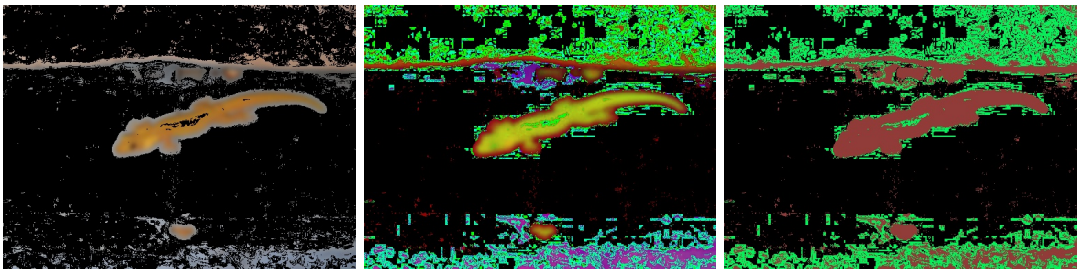
Dans de nombreux cas le lézard n'est pas la seule forme garder sur l'image, le traitement est plus efficace si le fond est homogène mais aussi si le lézard n'est pas camouflé dans son environnement.

4.3 Elimination des composantes connexes

En image une composante connexe est un groupe de pixels de couleurs identiques et en contact. Le traitement précédent n'a pas effacé tous ce qui n'était pas pertinent, il reste de nombreuses composantes connexes en dehors du lézard, qui doivent être supprimées.

4.3.1 Composantes au bord

Les premières composantes connexes qui seront effacées sont celles collées au bord, pour ce faire on commence par changer le repère colorimétrique RGB en HSV (Hue, Saturation, Value en français Teinte, Saturation, Valeur), la teinte représente la couleur, la saturation représente l'intensité de la couleur et la valeur est la brillance de la couleur, cette opération permet un plus grand écart entre les couleurs ce qui nous sera utile.
On effectue k-means est un algorithme de clustering, il permet d'analyser un jeu de données caractérisées par un ensemble de descripteurs, afin de regrouper les données similaires entre-elles. En image les données sont les pixels, elles sont rassemblées à l'aide de leurs composantes, le k-means attribuera au pixel du même groupe une nouvelle couleur, le nombre de groupe (donc de couleur dans l'image) est statué par la valeur de k.



(a) Image de base (b) Image HSV (c) Image K-means K = 3

FIGURE 3 – Image avant traitement

Le programme parcourt ensuite les bords de l'image, si un pixel qui n'est pas du fond est repéré alors on considère que c'est une forme qui subit une occlusion et donc tous les pixels dans cette composante sont supprimés.

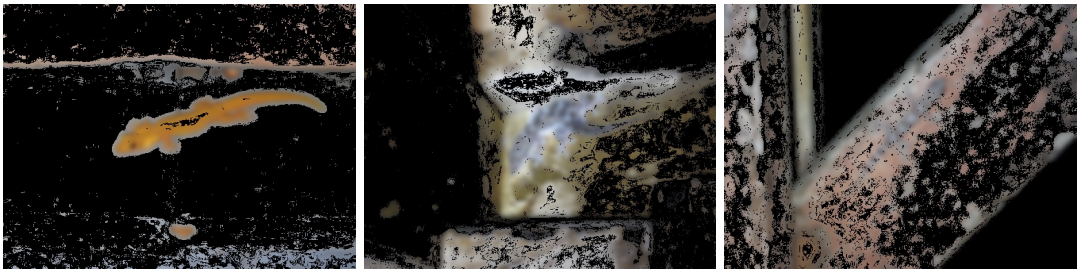
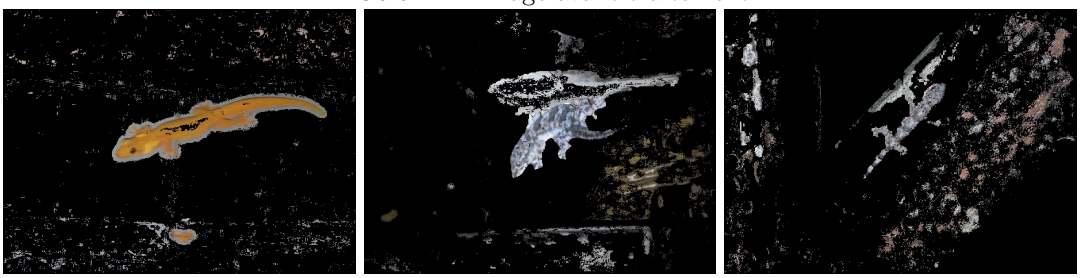


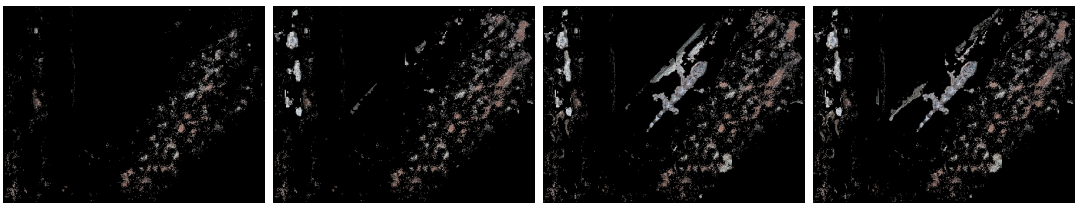
FIGURE 4 – Image avant traitement



(a) k = 2 (b) k = 3 (c) k = 4

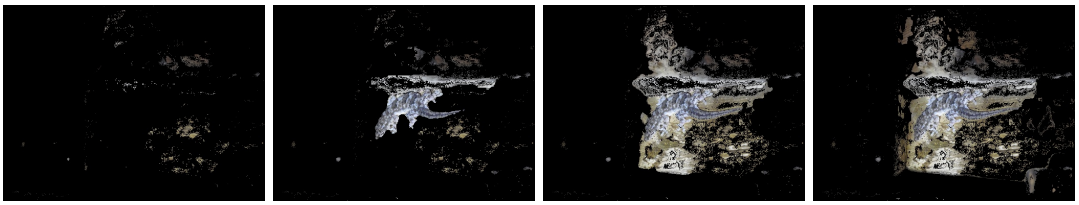
FIGURE 5 – Image après traitement

Les images en exemple nécessitent différentes valeurs de k, car certaines images ont besoin d'une valeur plus haute pour que la forme pertinente ne soit pas effacé avec les autres composantes. Mais prendre une valeur trop grande de k risque de garder et/ou modifier des éléments qui auraient été juste effacé.



(a) k = 2 (b) k = 3 (c) k = 4 (d) k = 5

FIGURE 6 – Image avec différentes valeur de k



(a) k = 2 (b) k = 3 (c) k = 4 (d) k = 5

FIGURE 7 – Image avec différentes valeur de k

4.3.2 Petites composantes connexes

D'autres composantes connexes sont effacées en fonction du nombre de pixels qui les composent, si ce nombre est inférieur à un seuil (ici 500 pixels) alors, la composante prend la couleur du fond.

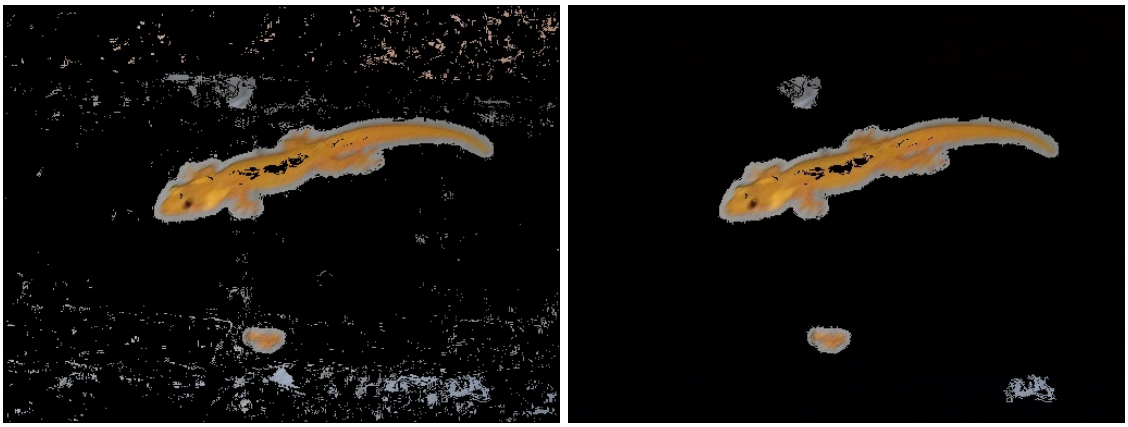


FIGURE 8 – Effacement des composantes ayant moins de 500 pixels

Malheureusement ce traitement entraine des pertes de certaines formes pertinentes qui étaient déjà trop petites sur l'image, elles sont confondues avec les éléments à effacer. Une des piste d'amélioration serait de ne plus se basé sur un seuil fixe pour chaque image, mais de générer plutôt un seuil adapté à chaque image qui se basera sur les tailles des autres composantes connexes et qui supprimerait les éléments les plus petits non pertinents.

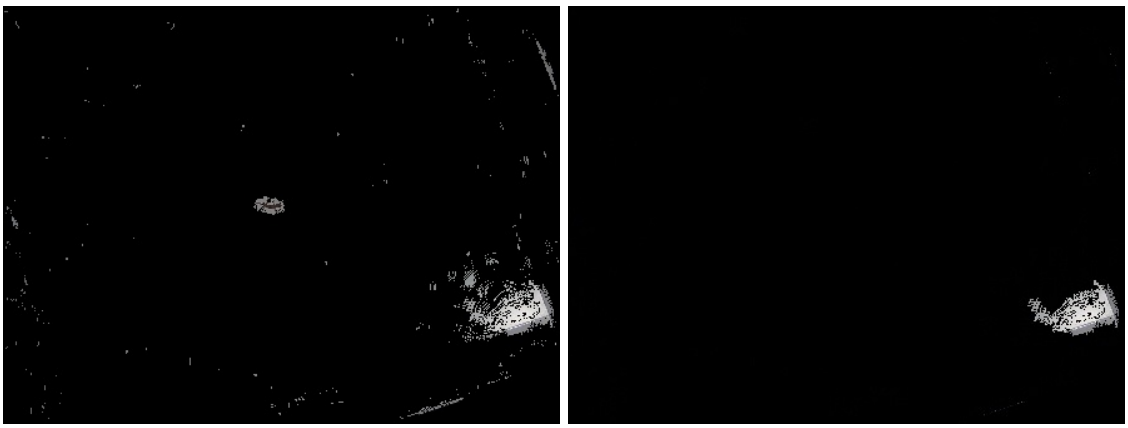


FIGURE 9 – Exemples de lézard perdu

4.3.3 Composantes dans zones intermédiaires

Pour exclure d'autres composantes plus grandes en déduit que l'utilisateur qui prendra la photo centrera le lézard dans l'image. Ainsi on a un cercle imaginaire dont le rayon est calculé avec le pourcentage de la dimension la plus petite de l'image. On utilise ensuite le centre des composantes connexes, si le centre de la composante ne se trouve pas dans le cercle, la composante sera effacé.

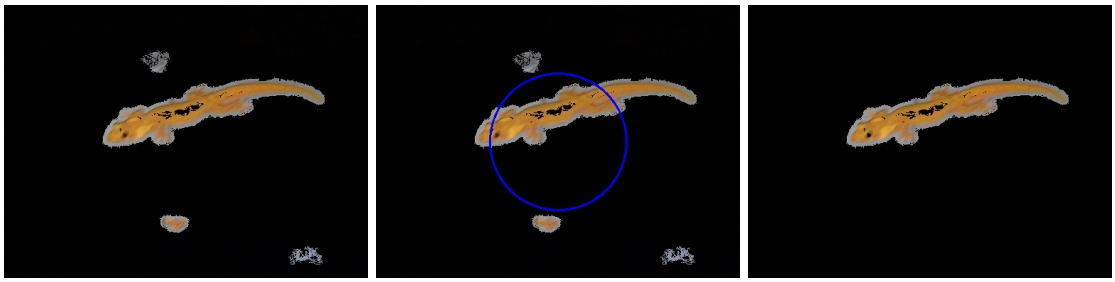


FIGURE 10 – Image avant et après traitement

4.4 Fermeture

Sur certaines formes pertinentes, on constate des trous, que l'on bouche à l'aide d'une méthode de morphologie mathématique, il s'agit d'une théorie et technique mathématique et informatique d'analyse de structure. La méthode utilisée sera la fermeture, qui est elle-même la combinaison de 2 méthodes de morphologie mathématique réalisée dans cet ordre : dilatation puis une érosion. Une dilatation est une opération qui consiste à étendre par les frontières une composante connexe alors que l'érosion consiste à réduire par les frontières.

Une forme (en bleu), sa dilatation morphologique (en vert), et son érosion morphologique (en jaune) par un élément structurant en forme de diamant.

La dilatation permet donc de combler les trous tandis que l'érosion permet de revenir à la forme d'origine.

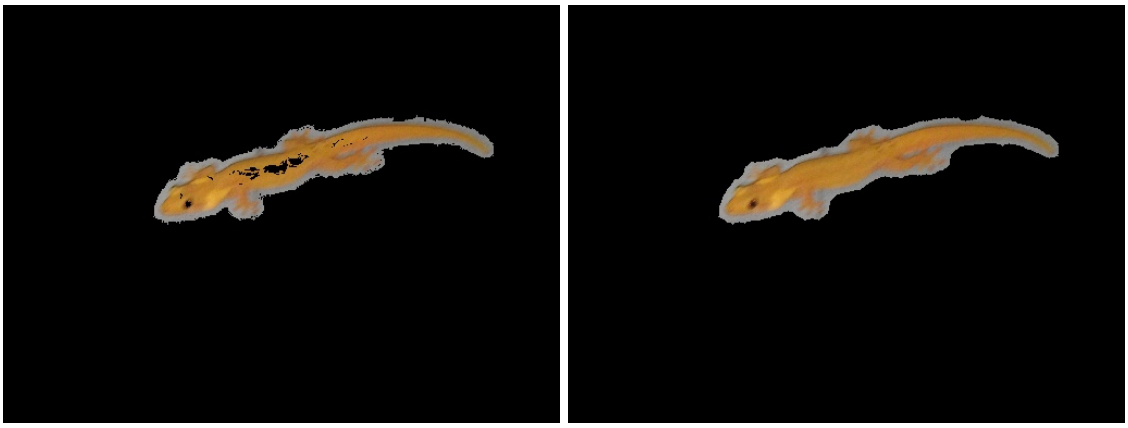
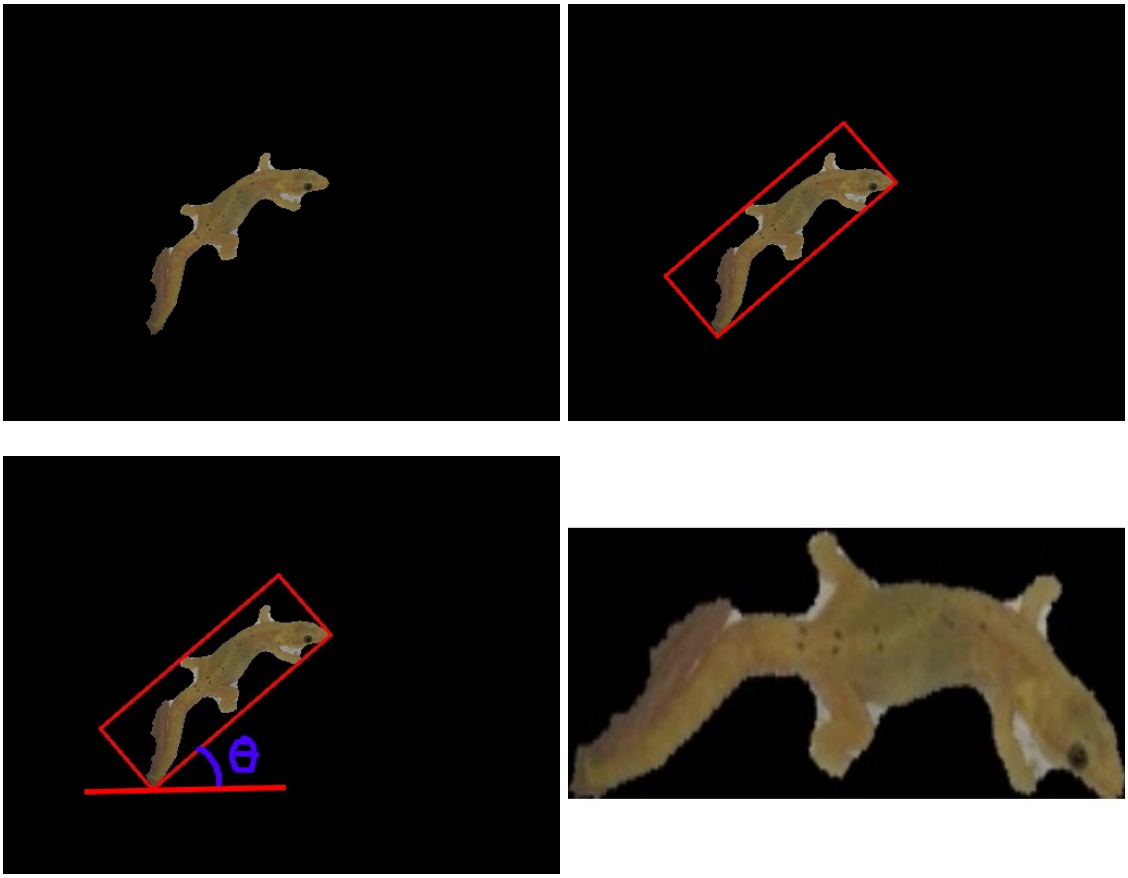


FIGURE 12 – Effacement des composantes ayant moins de 500 pixels

4.5 Rectangles englobants

En dernière étape on extrait tous les éléments restants dans chaque image, on récupère les éléments avec les rectangles englobants. Les rectangles englobants sont les plus petits rectangles qui comprennent tous les pixels de l'agrégat.

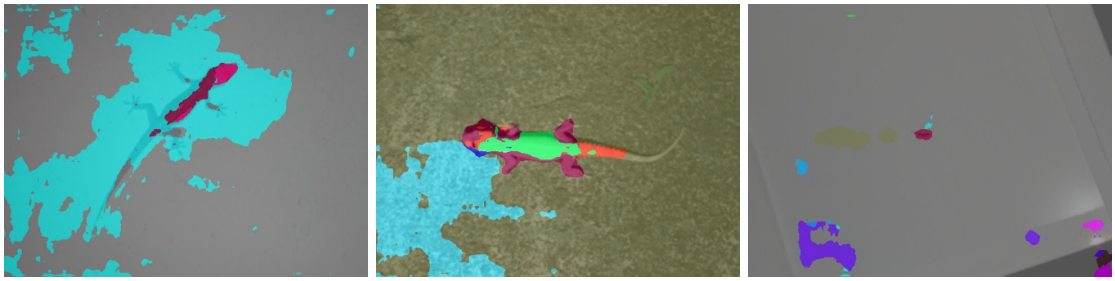


La forme est repérée, pour pouvoir extraire l'agrégat il faut que le rectangle soit à l'horizontale ou à la verticale, donc pour cela on tourne l'image d'un angle Θ, puis enfin on peut extraire la forme mais je vais toujours en sorte de l'extraire orienter horizontalement et toutes formes qui se rapprochent trop d'un carré (une différence entre la hauteur et la largeur de moins de 5%) seront considérées comme non pertinentes donc supprimées car on s'attend du lézard qu'il ait une forme allongée.

4.6 Pistes abandonnées :

4.6.1 Les réseaux de neurones

Cette méthode avait pour but de repérer les formes pertinentes en utilisant les réseaux de neurones grâce à la bibliothèque pixellib qui propose 3 modèles de réseaux adaptés à la segmentation d'image. On applique un filtre médian sur les images avant l'entrée dans les réseaux et en sortie on récupère l'image avec les éléments repérés.



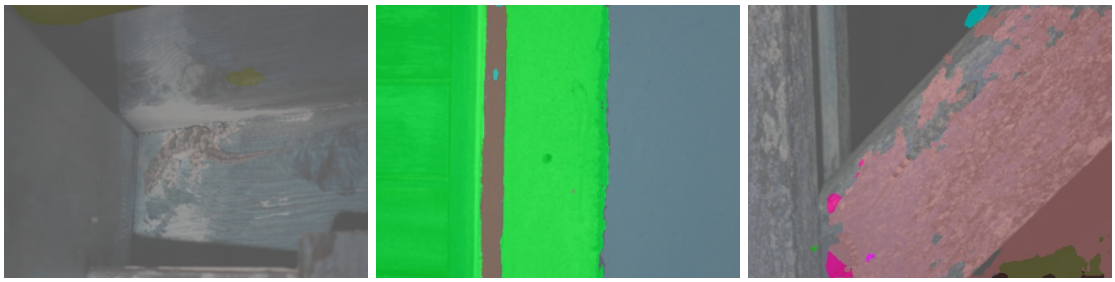


FIGURE 16 – Résultat avec le modèle "deeplabv3_xception65_ade20k.h5"

FIGURE 18 – Résultat avec le modèle "deeplabv3_xception65_tf_dim_ordering_tf_kernels.h5"

FIGURE 20 – Résultat avec le modèle "mask_rcnn_coco.h5"

Cette méthode a été abandonnée car les résultats étaient trop inconsistants, le lézard est plus souvent non repéré ou dans une zone trop large ou trop restreinte mais aussi le temps de traitement reste très important.

5 Extraction de caractéristiques

Pour pouvoir identifier le contenue dans l'image, il faut extraire des caractéristiques.Les caractéristiques choisies sont la couleur, la texture et les dimensions (ratio hauteur largeur)du lézard. Ses critères sont sélectionnés pour leurs pertinences, la couleur par exemple on observe plus souvent du jaune chez les geckos nains alors que chez les mabouias ça sera du blanc et du gris qui se démarquent. Pour la texture on remarque que les geckos nains en grande partie ils ne possèdent pas de motifs ou ils sont difficilement visibles, ainsi on s'attend à voir une homogénéité dans les textures. On espère se servir plus tard des dimensions dans le but d'effacer plus de formes non pertinentes qui auraient été considérées comme pertinentes.

5.1 Extraction de la couleur

On extrait les couleurs sur deux repères calorimétriques différent RGB et HSV, sur les deux repères on recherche la couleur prédominante de l'image.

5.1.1 Extraction RGB

Pour le RGB les trois composantes seront utilisées, l'information récupérer représente surtout un groupe de couleur dominante. Pour obtenir ses données on commence par réaliser un histogramme pour chaque composante RGB, un histogramme est un graphe de barre qui prend en abscisse la valeur du pixel et en ordonné le nombre de pixels qui ont cette valeur, cet histogramme ne tient pas compte du fond en noir.

FIGURE 21 – Gecko extrait quelconque

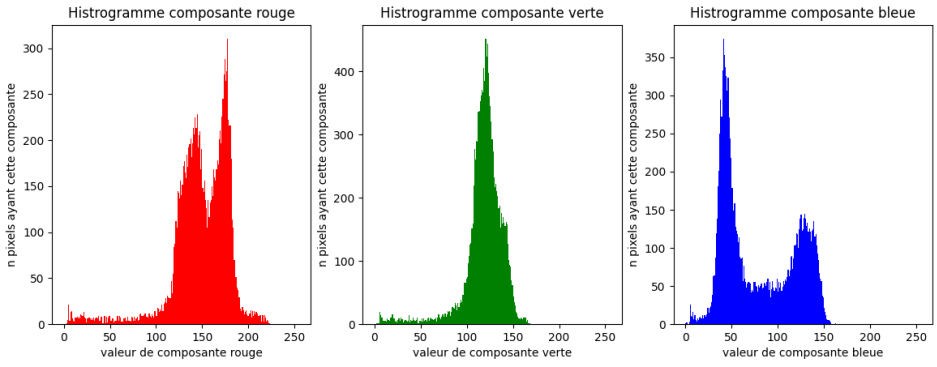


FIGURE 22 – Histogrammes couleurs du gecko

Chacune des composantes seront ensuite séparées en N groupes de même taille. Par exemple avec un $N = 10$, pour obtenir la taille d'un groupe on divise 256 par N donc, $256/10 = 25.6$ et donc pour les valeurs des groupes des composantes on obtient :

1. 0 - 25
2. 26 - 51
3. 52 - 76
4. 77 - 102

5. 103 - 128
6. 128 - 153
7. 154 - 179
8. 180 - 204
9. 205 - 230
10. 231 - 256

Maintenant ces groupes représentent les nouvelles valeurs de nos composantes et donc pour obtenir la couleur prédominante on teste toutes les combinaisons de groupe. Dans une matrice de dimensions N^3x4. Par exemple avec notre N= 10 voici les 11 premières lignes de la matrice :

groupe R	groupe G	groupe B	nombre de pixels
0 - 25	0 - 25	0 - 25	50
0 - 25	0 - 25	26 - 51	65
0 - 25	0 - 25	52 - 76	45
0 - 25	0 - 25	77 - 102	51
0 - 25	0 - 25	103 - 128	63
0 - 25	0 - 25	128 - 153	34
0 - 25	0 - 25	154 - 179	37
0 - 25	0 - 25	180 - 204	45
0 - 25	0 - 25	205 - 230	43
0 - 25	0 - 25	231 - 256	54
0 - 25	26 - 51	0 - 25	30

Il me reste plus que à sélectionner la combinaison avec le plus grand nombre de pixels, les données récupérées seront les numéros des groupes, c'est à dire dans le cas où la meilleure combinaison est R(128 - 153), G(52 - 76), B(0 - 25) on récupère les numéros des groupes donc 6, 3, 1. Ce procédé caractérise les informations récupérées sur l'image.

5.1.2 Extraction HSV

Dans l'extraction HSV, c'est surtout la composante H qui nous sera utile Hue la teinte en français, elle représente la couleur et elle est en degrés.

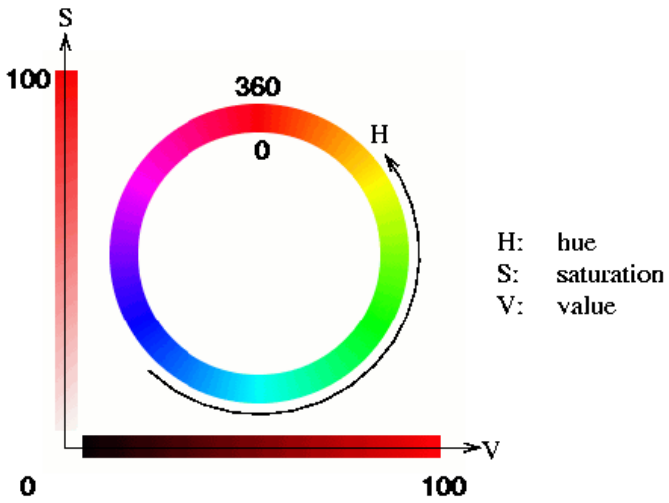


FIGURE 23 – Représentation du modèle HSV

Néanmoins la composante V sert à identifier les pixels du fonds et donc permet de les ignorer, si V = 0 c'est que le pixel est noir. La teinte est divisée en 6 angles de 60° mais comme on est limité par une variable de 1 octet donc des valeurs de 0 à 255, la limite est due à openCV en convertissant un image RGB en HSV, il n'alloue pas plus de bits aux cases de la nouvelle matrice donc elle reste avec le même nombre de bits que possède une image RGB.

1. $]30\,;\,90]° \Longrightarrow]21.335\,;\,64.005]$, Jaune
2. $]90\,;\,150]° \Longrightarrow]64.005\,;\,106.675]$, vert
3. $]150\,;\,210]° \Longrightarrow]106.675\,;\,149.345]$, cyan
4. $]210\,;\,270]° \Longrightarrow]149.345\,;\,192.015]$, bleu
5. $]270\,;\,330]° \Longrightarrow]192.015\,;\,234.685]$, magenta
6. $]330\,;\,360]° \cup [0\,;\,30]° \Longrightarrow]234.685\,;\,255] \cup [0\,;\,21.335]$, rouge

Ainsi on réalise l'histogramme des valeurs qui possèdent 6 barres pour les 6 angles et l'information que l'on récupère est le numéro de l'angle avec le plus grand nombre de pixels, les informations récupérées sur les deux repères colorimétriques seront utilisées pour que la couleur ait un plus grand poids dans la classification.

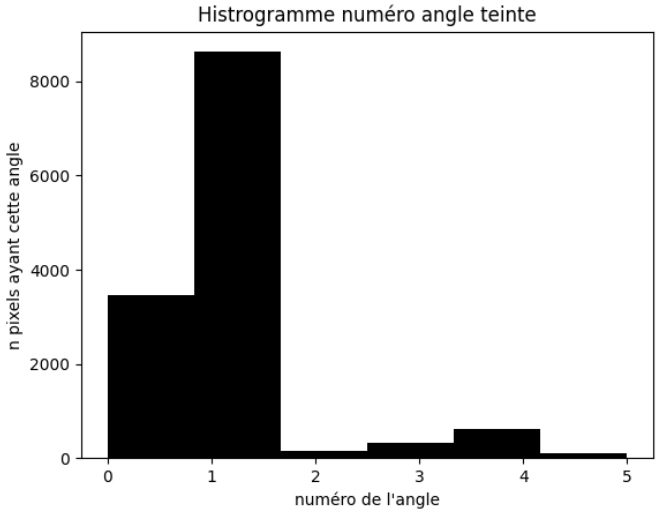


FIGURE 24 – Histogrammes Teinte du gecko

5.2 Extraction de la texture

Pour extraire les textures on utilise la méthode LBP (local binary pattern) qui permet d'identifier différents paternes de textures.

5.2.1 LBP

Local Binary Patterns propose une description des textures a un niveau local. Le traitement ce fait sur une image en niveau de gris, pour obtenir une texture on applique un seuillage sur le voisinage d'un pixel si le pixel voisin est plus grand il a une valeur de 1 sinon une valeur de 0, ensuite on récupère toutes les valeurs du voisinage pour avoir un mot binaire qui représente notre texture.

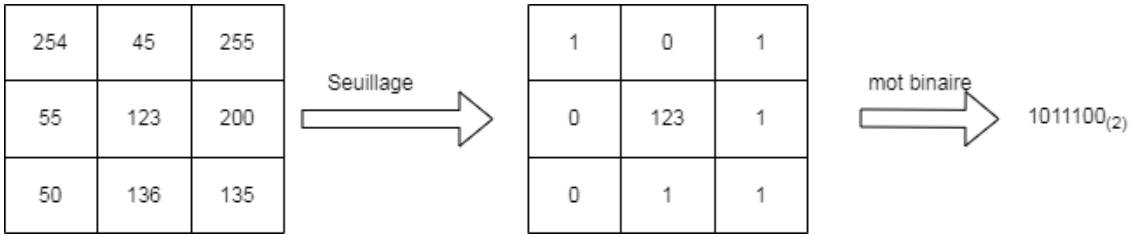


FIGURE 25 – Schéma application du LBP

Ce qui nous donne $2^8 = 256$ textures possibles, mais ce nombre peut être réduit car la rotation n'a pas encore été prise en compte. Par exemple les paternes 00000001 et 01000000 seront considéré comme étant identique car il suffit d'effectuer des rotations afin obtenir l'autre texture. On a ainsi 36 paternes qui nous permettent d'obtenir les 256 autres :

00000000	00000001	00000011	00000101	00000111	00001001
00001011	00001101	00001111	00010001	00010011	00010101
00010111	00011001	00011011	00011101	00011111	00100101
00100111	00101011	00101101	00101111	00110011	00110101
00110111	00111011	00111101	00111111	01010101	01010111
01011011	01011111	01101111	01110111	01111111	11111111

FIGURE 26 – Tableau des 36 paternes principaux

Maintenant que l'on a les 36 principaux paternes, il faut pouvoir savoir auxquelles appartiennent les 220 autres et on le fait avec la méthode suivante :

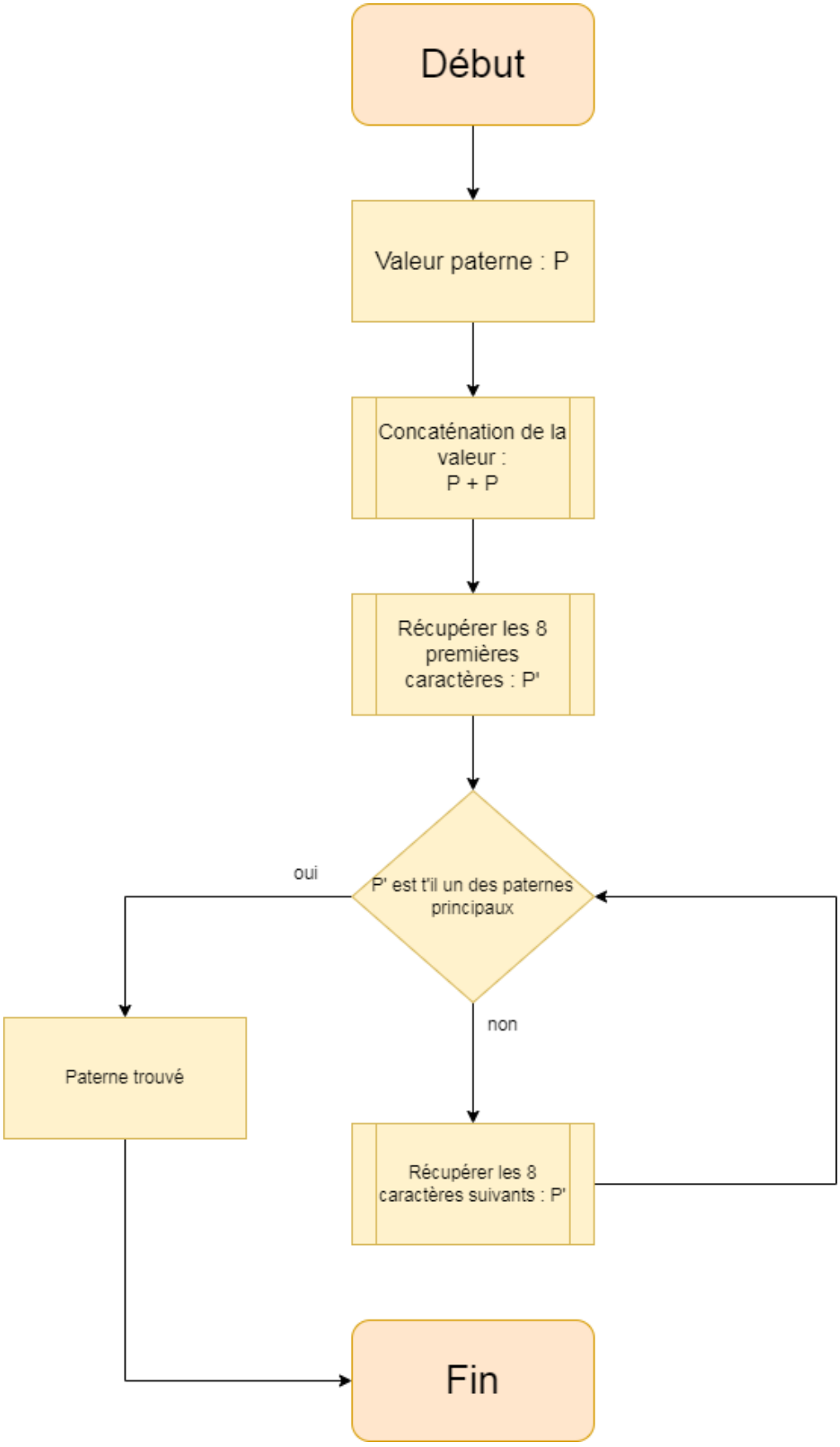


FIGURE 27 – Identifier à quels paternes principaux appartient le paterne

5.2.2 Application du LBP

Maintenant que tout est mis en place, on peut appliquer la procédure sur les images. Pour observer l'application sur une image, on associe à tous les principaux paternes une couleur, les pixels ayant ces paternes prendront ces couleurs.

00000000		00010001		00100111		00111111	
00000001		00010011		00101011		01010101	
00000011		00010101		00101101		01010111	
00000101		00010111		00101111		01011011	
00000111		00011001		00110011		01011111	
00001001		00011011		00110101		01101111	
00001011		00011101		00110111		01110111	
00001101		00011111		00111011		01111111	
00001111		00100101		00111101		11111111	

FIGURE 28 – code couleur

Les couleurs nous permettrons de déterminer aisément qu'elles sont les paternes prédominant dans l'image, lesquelles peuvent être ignorée. Les images suivantes sont les premiers tests avec ce traitement :

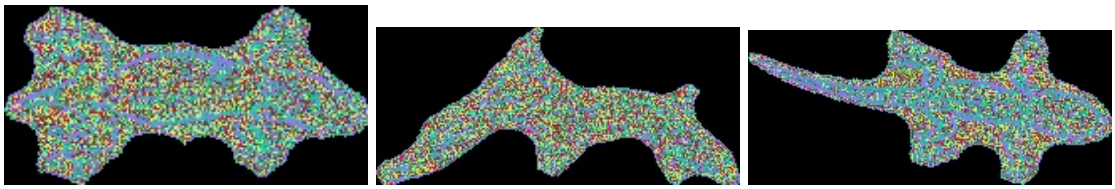


FIGURE 29 – Premier rendu du traitement LBP

On peut constater que les paternes sont trop diversifiés et on ne peut en retirer d'information pertinente. Le problème est dû au bruit encore important dans l'image. Donc on applique de nouveau un filtre médian afin de réduire la diversité de valeur.

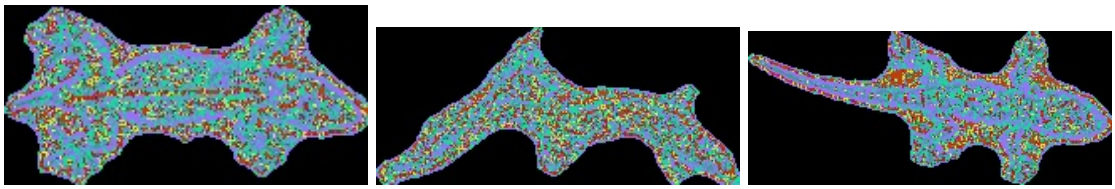


FIGURE 30 – Rendu du traitement LBP avec filtre median

Maintenant des paternes commencent à se dégager par rapport à d'autres. En additionnent le seuil qui déterminent les paternes à un autre seuil, on peut réduire la sensibilité de celui-ci, et réduire la présence de paterne non pertinent repérés. Donc on commence la recherche d'un bon seuil optimal :

FIGURE 31 – Rendu du traitement LBP avec filtre median et seuil à 128

FIGURE 32 – Rendu du traitement LBP avec filtre median et seuil à 64

FIGURE 33 – Rendu du traitement LBP avec filtre median et seuil à 32

FIGURE 34 – Rendu du traitement LBP avec filtre median et seuil à 16

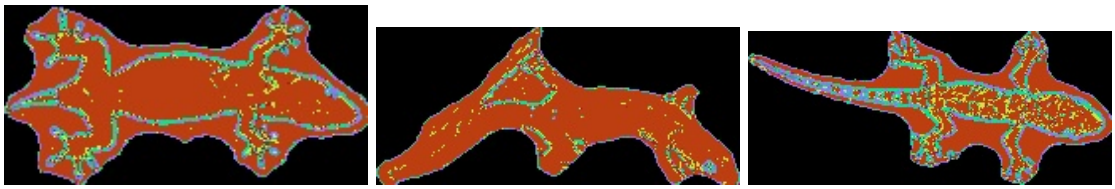


FIGURE 35 – Rendu du traitement LBP avec filtre median et seuil à 8

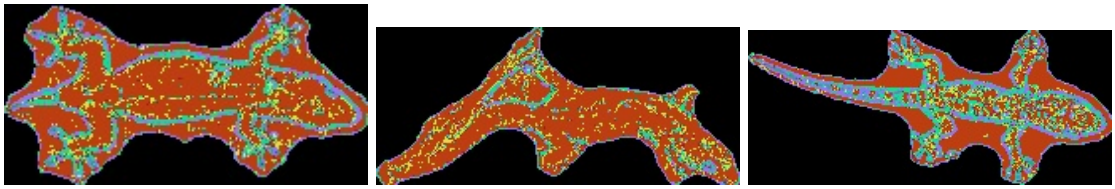


FIGURE 36 – Rendu du traitement LBP avec filtre median et seuil à 4

On constate qu'avec un seuil de 16 à 4, on obtient des paternes qui se distingues nettement, les seuils qui les précèdes (32 à 128) sont trop grand et donne surtout des paternes de fond. On peut aussi ignorer certains paternes inutile comme ceux désignant le fond.

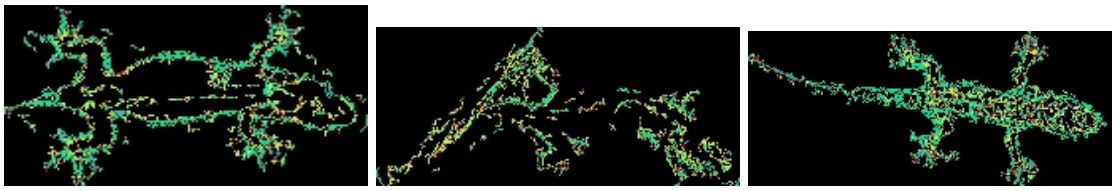


FIGURE 37 – Exclusion des paternes du fond et de bruit

On peut remarquer qu'en majorité le résultat obtenu sur les geckos nains, donne un lézard dont l'intérieur ne présente pas ou peu de détail.

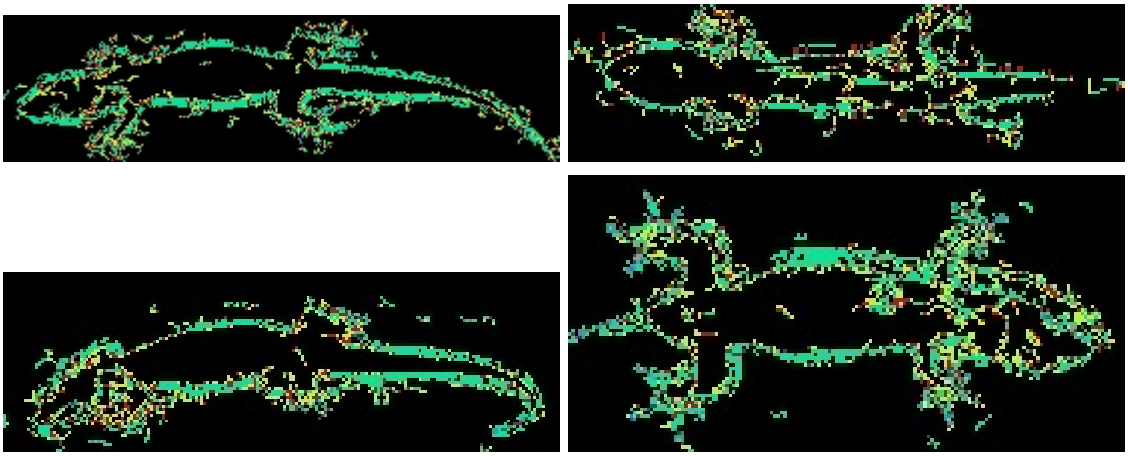


FIGURE 39 – Image de gecko nain en fin de traitement

Tandis que pour les mabouias c'est le contraire, ils tendent à avoir beaucoup plus de paternes en leur centre

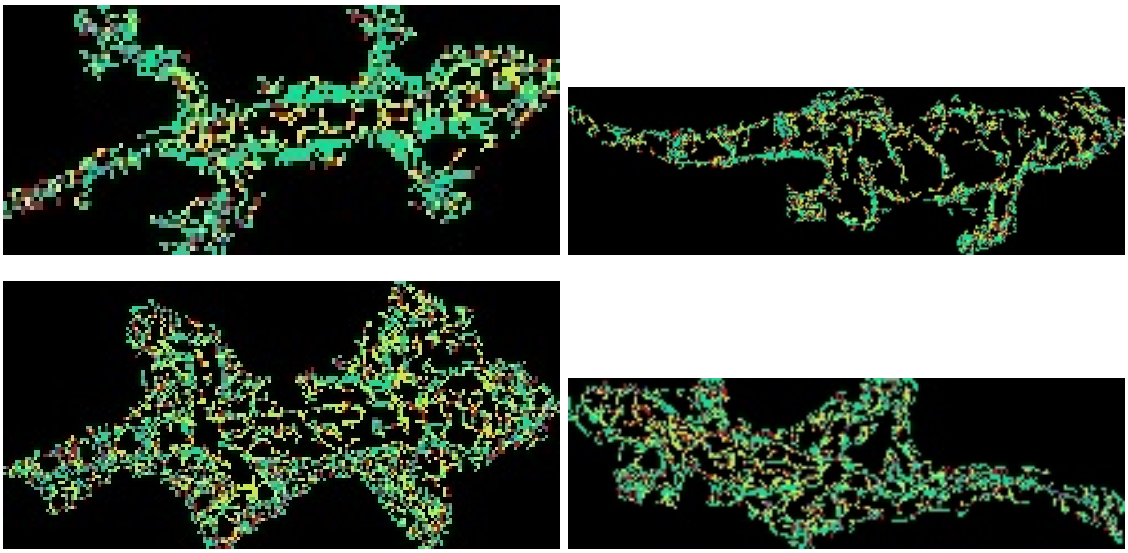


FIGURE 41 – Image de mabouia en fin de traitement

5.2.3 Taux homogénéité

Le taux d'homogénéité est un critère qui informe sur la proportion de paternes homogène dans une image. Comme nous le savons, un paterne est un mot binaire, ainsi un paterne homogène est décrit comme un paterne qui compte moins de 3 transitions entre les valeurs des bits.
Par exemple avec le mot 00011000 on constate deux transitions, une fois de 0 vers 1 et une autre de 1 vers 0 donc c'est un paterne homogène. Maintenant on peut calculer le taux d'homogénéité, en divisant le nombre de paternes homogène sur les paternes de l'image. C'est ce critère qui servira dans la classification.

5.3 Dimensions de l'image

Les informations pour la dimension sont récoltées très simplement, vu que tous les rectangles sont orientés horizontalement, on divise hauteur/largeur, c'est cette information qui sera utilisée dans la classification. Mais on espère l'utiliser aussi dans le but de faire une moyenne qui représentait le ratio d'un lézard typique dans le but de repérer moins d'éléments non pertinents.

6 Identification

6.1 Agencement des données

On utilise pour l'identification une base de 203 lézards, dont 141 éléments chez les geckos nains et 62 éléments chez les mabouias. On crée un jeu de donnée qui contiendra 8 types de données différente qui sont les suivants :

— ratio_h_l représente le ratio hauteur, largeur de l'image du lézard
— classe spécifie pour les tests si c'est un gecko nain ou un mabouia
— couleur_r est le groupe de composante rouge dominante de l'image
— couleur_g est le groupe de composante vert dominante de l'image
— couleur_b est le groupe de composante bleu dominante de l'image
— couleur_hsv est la teinte dominante de l'image
— homogene est la proportion de paterne homogène de l'image
— non_homogene est la proportion de paterne non homogène de l'image

ratio_h_l	classe	Couleur_r	Couleur_g	Couleur_b	Couleur_hsv	homogene	non_homogene
0.24803149606299213	gecko	11.0	10.0	10.0	1	0.9532194480946123	0.04678055190538765
0.3333333333333333	gecko	16.0	16.0	13.0	1	0.9622876087857439	0.03771239121425611
0.8265895953757225	gecko	15.0	14.0	12.0	1	0.9417920209287116	0.058207979071288427
0.3103448275862069	gecko	13.0	13.0	4.0	1	0.974785100286533	0.02521489971346705

FIGURE 42 – Visualisation des 5 premières lignes des données

6.2 Résultat de la classification

On a tenté 3 algorithmes différents de classification avec 66% de la base donnée en phase d'apprentissage, je vais les présenté de la moins précise à la plus précise sur notre jeux de données en fonction de nos critères retenus. On commence par la méthode random tree[2] est une méthode d'apprentissage basée sur l'utilisation des arbre de décision (un outil d'aide à la décision présentant l'ensemble des choix sous la forme d'un graphe arbre) comme un modèle prédictif, avec elle on obtient une précision de 70.9%.

```
=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                0,854    0,571    0,774      0,854    0,812      0,308    0,641     0,762     gecko
                0,429    0,146    0,563      0,429    0,486      0,308    0,641     0,415     mabouia
Weighted Avg.   0,725    0,442    0,709      0,725    0,713      0,308    0,641     0,657
```

FIGURE 43 – Tableau de précision par classe de random tree

On à la méthode randomforest (ou forêt d'arbre décisionnel) a été formellement proposée en 2001 par Leo Breiman[1], elle combine les concepts de sous-espaces aléatoires et de bagging. Elle effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différentes.On obtient avec cette méthode une précision de 75.7%

```
=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                0,896    0,524    0,796      0,896    0,843      0,415    0,804     0,909     gecko
                0,476    0,104    0,667      0,476    0,556      0,415    0,804     0,645     mabouia
Weighted Avg.   0,768    0,396    0,757      0,768    0,756      0,415    0,804     0,828
```

FIGURE 44 – Tableau de précision par classe de random forest

Et enfin la méthode J48 est une implémentation open source en java pour weka de l'algorithme C4.5 développé par Ross Quinlan[6], il s'agit d'un arbre de décision perfectionner.On obtient avec cette méthode une précision de 85,2%

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,917    0,286    0,880      0,917    0,898      0,650    0,820     0,869     gecko
                 0,714    0,083    0,789      0,714    0,750      0,650    0,820     0,670     mabouia
Weighted Avg.    0,855    0,224    0,852      0,855    0,853      0,650    0,820     0,808
```

FIGURE 45 – Tableau de précision par classe de J48

6.3 Bilan de la classification

On peut constater qu'aux vues des résultats, que les critères retenus sont pertinent pour la classification avec des performances largement mieux que le hasard qui aurait donné des précisions au alentour des 50%, mais on peut constater que notre pire résultat s'en éloigner très bien en étant à 70.9% et au plus haut en a une précision de 85.2%.

7 Conception de l'application

L'application sera réalisée à l'aide de la technologie PWA (Progrssive Web App). Elle permet un mode de fonctionnement hors ligne d'une application Web via la mise en cache des différents contenus (images, données, scripts, etc).

7.1 Architecture de l'application

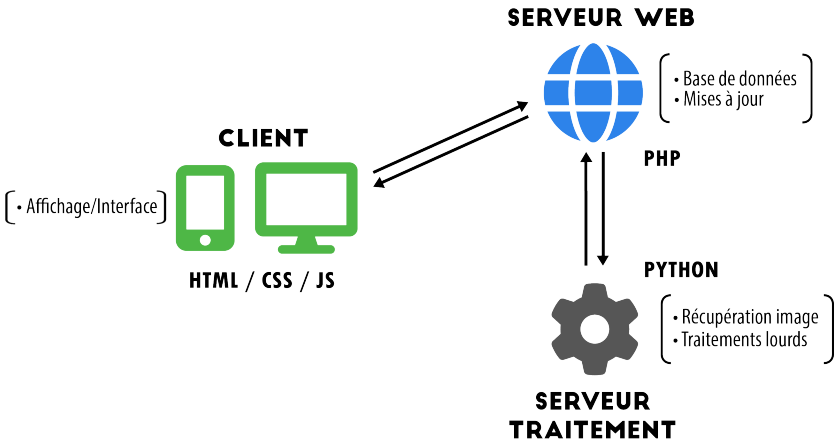


FIGURE 46 – Schéma fonctionnel du fonctionnement interne de l'application

Le client :
— Permet de prendre des photos
— Envoie les informations au serveur
— Visualiser la carte

Le serveur web :
— Collecte les images

Le serveur traitement :
— télécharge les images mises à disposition sur le serveur web
— Réalise l'extraction des formes pertinentes
— Réalise l'extraction des caractéristiques

7.2 Présentation de l'interface

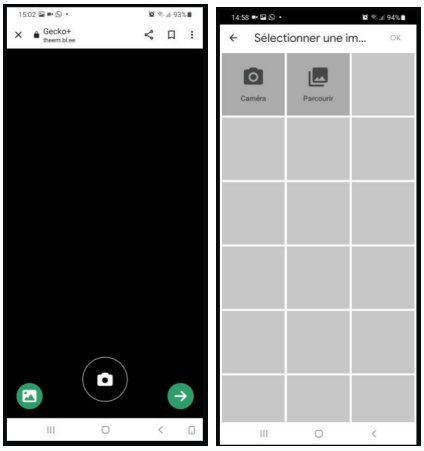
L'application web se nomme gecko+ et se compose de 4 pages principales qui sont photo, actualité, carte et configuration.

FIGURE 47 – logo gecko+

L'application s'ouvre sur la page des actualités qui montrera les derniers lézards pris en photo et analysés.

FIGURE 48 – Application page actualité

La page photo permet de prendre en photo les lézards mais aussi de rechercher ceux déjà présents dans la galerie afin de permettre leurs envoient au serveur.



(a) photo (b) galerie

FIGURE 49 – Application page photo

La page carte permet de visualiser une carte de la région pour un recensement des lézards par endroit.

FIGURE 50 – Application page actualité

La page des configurations permet de gérer les autorisations de l'application, de trouver les

informations la concernant et de la partager.

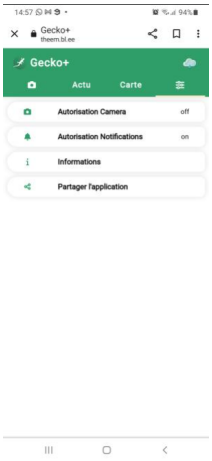


FIGURE 51 – Application page actualité

8 Conclusion

8.1 Bilan des travaux

Le but de mon travail durant ce stage, était la mise en place d'une application pour automatiser l'identification entre 2 types de lézards, le gecko nain et le mabouia. En programmant avec Python en utilisant les bibliothèques openCV, PIL j'ai pu extraire les zones d'intérêts pour ensuite extraire des caractéristiques qui ont servi plus à l'identification. Dans mes tests de classification avec le logiciel weka j'ai pu obtenir un résultat correct avec l'algorithme J48, on atteint une précision de 85,2%. Une des perspectives pour améliorer cette valeur sera la suppression de composantes connexes de petites tailles au niveau du traitement LBP. Malheureusement je n'ai pas pu faire l'encapsulation du code dans l'application donc celle-ci n'est pas encore totalement au point.

8.2 Analyse personnelle

Au cours de ce stage j'ai pu m'intéresser plus en détail au monde de la documentation scientifique en découvrant google scolar mais surtout en accomplissant un état de l'art sur les connaissances d'identification automatique des reptiles. J'ai pu mettre à l'épreuve mes compétences en Python, en traitement d'image, explorer plusieurs pistes pour atteindre un but précis, ne pas craindre de perdre du temps en s'intéressant à des idées qui finalement ne donnaient pas de résultat concluant. Et j'espère que ces nouvelles compétences me seront utiles dans mes futurs projets.

Références

[1] Leo Breiman. Random forests. *Machine Learning*, 45(1) :5–32, Oct 2001.

[2] Raphael Maree. *Classification automatique d'images par arbres de decision.* PhD thesis, University of Liege - Electrical Engineering and Computer Science, 2005.

[3] Anika Patel, Lisa Cheung, Nandini Khatod, Irina Matijosaitiene, Alejandro Arteaga, and Joseph Gilkey. Revealing the unknown : Real-time recognition of galápagos snake species using deep learning. *Animals*, 10 :806, 05 2020.

[4] Catarina Lopes Pinho. Using deep learning to classify images of wall lizards. 2021.

[5] Mahdi Rajabizadeh and Mansoor Rezghi. A comparative study on image-based snake identification using machine learning. *Scientific Reports*, 11 :19142, 09 2021.

[6] Steven L. Salzberg. C4.5 : Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16(3) :235–240, Sep 1994.