remote sensing

MDPI

Article

Détection des arbres dans des vergers d'agrumes par imagerie satellitaire et apprentissage profond

Mehdi El-Haylali [1], Yahya Zennayi [2] et Omkalthoum Amar[2]

[1] Ecole Centrale Casablanca, Casablanca, Maroc; mehdi.elhaylali@centrale-casablanca.ma
[2] Moroccan Foundation for Advanced Science, Innovation and Research, Rabat, Maroc.

Abstract: Dans cet article, nous présentons une nouvelle approche pour la détection des arbres dans les images satellitaires de vergers d'agrumes. L'approche se base sur « apprentissage profond » par des « réseaux de neurones à convolution » (CNN) incorporés dans l'algorithme de détection d'objets connu sous l'appellation de : « Yolo » (You Only Look Once). Les images satellitaires de très haute résolution utilisés pour l'entraînement et la validation sont issues du logiciel à accès libre « Google Earth Pro ». Les différents traitements appliqués aux images avant et après l'entraînement et la validation ont été programmés sous Python. Les résultats du test qui a été conduit à l'issue de l'entraînement du détecteur « Yolo » sont prometteurs, notamment le taux de détection des arbres individuels qui a atteint 92.6%. Ces résultats indiquent la faisabilité de la détection des arbres d'agrumes sur des images satellitaires par « apprentissage profond » et dans des vergers où les arbres ne sont pas forcément distribués en rangées.

Mots clés: Détection des arbres d'agrumes; Apprentissage profond; Imagerie satellitaire de très haute résolution; Traitement d'image.

1. Introduction

La gestion des grandes exploitations agricoles requiert une main d'œuvre importante, un effort et un temps considérables, sans oublier son coût et sa susceptibilité à l'erreur humaine. La technologie de télédétection par imagerie satellitaire ou aérienne fournit aux gestionnaires de fermes des données actualisées sur de vastes zones, constituant ainsi un outil efficace d'aide à la décision pour ces gestionnaires dans leurs pratiques de gestion agricole **[1]**. Par ailleurs, l'avènement -dans les deux dernière décennies- d'une imagerie satellitaire de très haute résolution spatiale (inférieure à 1m/pixel) ainsi que la large commercialisation des drones, ont permis aux images optiques de devenir une ressource précieuse dans la reconnaissance individuelle des arbres dans les grands vergers d'agrumes **[2]**. Pour atteindre l'objectif de cette étude, notre revue bibliographique se restreint sur la détection individuelle des arbres d'agrumes par imagerie multispectrale -satellitaire comme aérienne- ainsi que les méthodes et algorithmes de traitement d'image qu'y sont afférents.

Partant de la revue de littérature que nous avons menée sur les approches de détection d'arbres, il s'avère que les techniques utilisées peuvent être classées en trois catégories : *(1)* des techniques géométriques de traitement d'image comme « Canny Edge Detection » et la « Transformée Circulaire de Hough » **[3]**, *(2)* l'apprentissage automatique (Machine Learning) et en particulier les algorithmes de classification comme le « K-means » **[4]**, et finalement, *(3)* l'apprentissage profond (Deep Learning) avec le fameux Réseau de Neurones à Convolution (CNN) **[5, 6]**. Les meilleures performances ont été obtenues avec le CNN. Cependant, cette technique requiert une large base de données pour l'entraînement et une capacité de calcul importante. Les techniques d'apprentissage automatique et de traitement d'image traditionnelles donnent quant à eux de bons résultats (70-80% de précision), mais elles nécessitent des calibrations et ajustements manuels de certains paramètres. D'une manière

générale, toutes ces approches restent difficilement généralisables à tout type de verger. Le tableau ci-dessous (**Tableau 1**) présente de façon concise des informations clés liées aux articles étudiés dans le cadre de cette étude bibliographique.

Si les techniques d'apprentissage profond ont été appliquées dans le cadre de la détection des arbres dans des vergers d'agrumes, elles se sont restreintes à des types de vergers où les arbres sont distribués en rangées **[5, 6]**. L'objectif du présent article est d'élargir l'usage de cette technique aux vergers où les arbres ne sont pas forcément régulièrement arrangés. Cette étude a permis de démontrer la faisabilité de la détection d'arbres par Intelligence Artificielle sur des vergers non régulièrement plantés à condition de se doter d'une base de données d'images suffisante (de l'ordre du millier d'images) et d'appliquer les recommandations que nous avons proposées afin d'optimiser les résultats des détections.

Tableau 1. Étude bibliographique sur la détection des arbres d'agrumes.

Article	Outils et techniques	Type d'imagerie	Type de verger	Acquis par	date
UAV-Based High Throughput Phenotyping in Citrus Utilizing Multispectral Imaging and Artificial Intelligence	Convolutional Neural Network (CNN), YOLOv3, Pix4D software,	Multispectrale - 5 bandes	Uniquement pour les arbres distribués en rangées	Drone	2019
Automatic citrus tree extraction from UAV images and digital surface models using circular Hough transform	Python, Pix4D software, Open-CV Canny edge detection, Circle Hough Transform	- RGB - Digital surface models (DSMs) générés par Pix4D	Valable pour des arbres aléatoirement dispersés	Drone	2018
2-D delineation of individual citrus trees from UAV-based dense photogrammetric surface models	Orientation-based radial symmetry transform, Pix4D	RGB et DSM	Valable pour des arbres aléatoirement dispersés	Drone	2017
Detection and counting of orchard trees from VHR images using a geometrical-optical model and marked template matching [7]	Tempate Matching	RGB	Valable pour des arbres aléatoirement dispersés	Satellite (images Google Earth)	2016
Automatic detection and delineation of citrus trees from VHR satellite imagery	FRS transform, Hierarchical processing	Multispectrale (notamment les bandes proche infrarouge)	Valable pour des arbres aléatoirement dispersés	Satellite GeoEye-1	2015

A genetic algorithm for citrus tree counting and canopy diameter estimation	Algorithme génétique, K-means, érosion morphologique	multispectrale	Uniquement pour les arbres distribués en rangées	satellite Quickbird	2009

2. Matériel et méthodes

2.1. Base de données

Nous avons construit une base de données d'images satellitaires d'arbres d'agrumes à l'aide du logiciel Google Earth Pro. Nous avons extrait du logiciel des images de vergers provenant des États-Unis, de la Turquie et de la région de Sidi Slimane au Maroc (Figure 1). Ceci est dans le but d'avoir différents configurations et arrangements de vergers possibles. Ces images RGB de taille 1260×3840 ont été ensuite découpées en sous-images de taille 432×480, ce qui donne 8×5 sous-images pour chaque image originale (exemple : Figure 2). Etant donné que nous avons extrait 5 images du logiciel, nous avons fini par obtenir une base de données d'environ 40×6=240 images de taille 432×480 (H×L). Cette base de données servira par la suite pour l'entraînement du réseau de neurones à convolution.

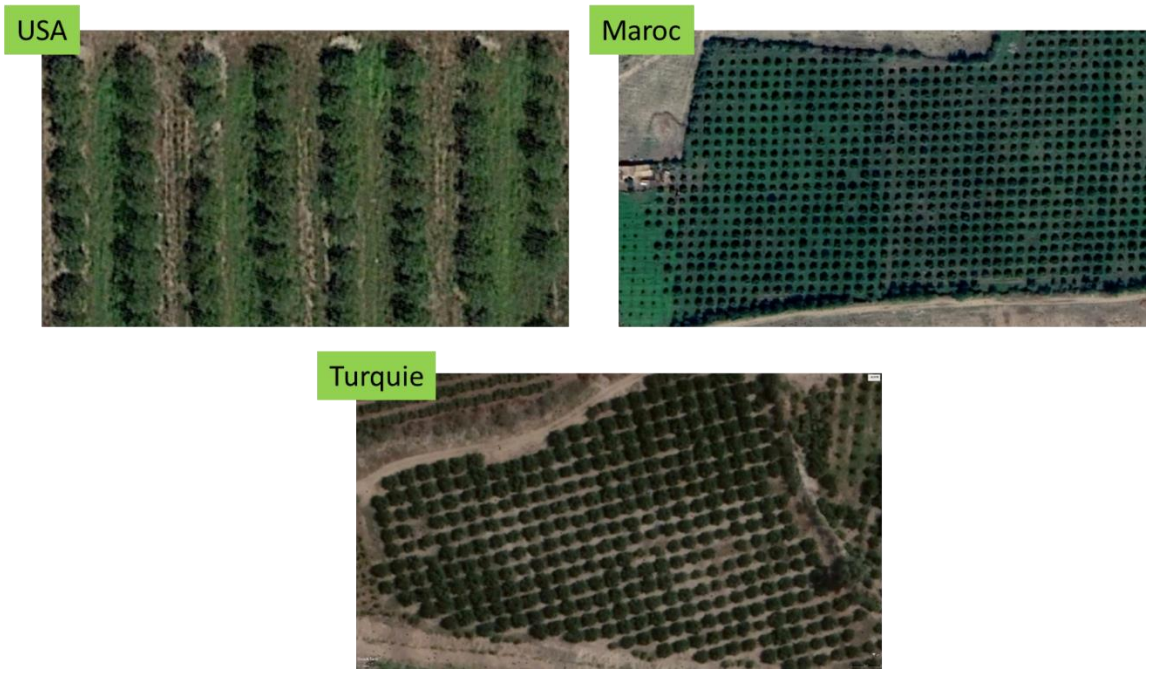

Figure 1.extrait de la base de données.

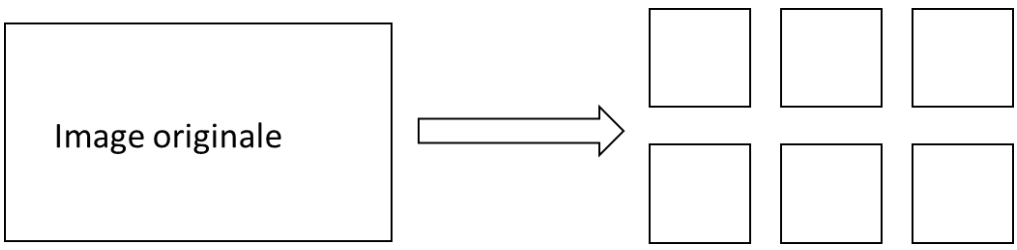

Figure 2.découpage d'une image.

2.2. *Annotation de la base de données*

L'opération d'annotation consiste à encadrer dans une image les différentes classes d'objets à détecter. Pour notre cas, la seule classe d'objet à détecter est la classe « arbre » (Figure 3). L'opération d'annotation a été effectuée en utilisant un script en Python.

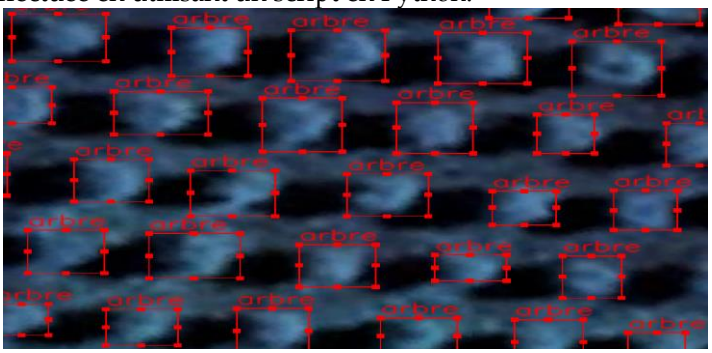

Figure 3. Annotation des arbres dans l'image

L'opération d'annotation donne lieu à la création d'un fichier .txt pour chaque sous-image. Chaque rectangle délimitant un arbre (bounding box en anglais) apparaissant dans l'image est représenté dans une ligne de ce fichier comme suit :

0 x y w h

Avec 0 c'est l'index de la classe « arbre », (x,y) la position absolue du rectangle, alors que w et h représentent respectivement la largeur et la hauteur absolues de ce rectangle (Figure 4).

```
1  0 0.005208333333333481 0.3287037037037037 0.26875 0.2361111111111111
2  0 0.2770833333333331 0.2708333333333335 0.20416666666666666 0.22685185185185186
3  0 0.5125000000000002 0.17245370370370372 0.275 0.24305555555555555
4  0 0.7791666666666668 0.08564814814814836 0.25833333333333336 0.25
5  0 0.8197916666666667 0.35185185185185164 0.26875 0.25
6  0 0.5677083333333335 0.44212962962962976 0.21041666666666667 0.2824074074074074
7  0 0.3104166666666668 0.5138888888888888 0.24583333333333332 0.22685185185185186
8  0 0.04062499999999991 0.5856481481481484 0.29375 0.25
9  0 0.1343749999999999 0.8622685185185186 0.23541666666666666 0.2569444444444444
10 0 0.3916666666666666 0.7789351851851851 0.2125 0.24305555555555555
11 0 0.6354166666666665 0.7268518518518516 0.22916666666666666 0.27314814814814814
12 0 0.8958333333333335 0.6354166666666665 0.24583333333333332 0.2708333333333333
13 0 0.9427083333333335 0.9155092592592591 0.22291666666666668 0.2569444444444444
14 0 0.7208333333333332 0.9988425925925926 0.2 0.22916666666666666
15
```

Figure 4. le contenu d'un fichier .txt.

2.3. *Entraînement du détecteur d'objets*

Afin de détecter les arbres d'agrumes dans une image, nous utilisons Yolo qui est un algorithme de pointe dans la détection d'objets [8]. L'usage de Yolo requiert, comme tout algorithme d' « apprentissage profond », la séparation de la base de données en deux catégories ; les « données d'entraînement » et les « données de validation ». À l'aide d'un script Python deux fichiers ont été créés « train.txt » et « test.txt » contenant respectivement les chemins vers les données d'entraînement et de validation. 80% de la base de données est allouée à l'entraînement alors que 20% est allouée à la validation.

Yolo étant basé sur des réseaux de neurones à convolution, il requiert également le choix d'une architecture pour ces réseaux. Pour notre étude, nous avons utilisé une architecture réduite, dans le sens où le nombre de couches constituant le réseau de neurones a été réduit. Ceci est pour deux raisons ; la première est qu'il s'agit de détecter une seule classe d'objet qui est l' « arbre », la deuxième réside dans le fait qu'une architecture réduite, réduit également le temps de calcul.

Après la séparation de la base de données et la détermination de l'architecture à utiliser par Yolo, nous avons lancé l'entraînement sur un ordinateur équipé d'un GPU. L'entraînement a duré 5h et a donné lieu à un fichier .weights contenant les poids finaux qui vont être utilisés par Yolo pour la détection des arbres dans une image quelconque.

2.4. Application du détecteur sur une image

Nous voulons détecter les arbres dans l'image d'un verger d'agrumes (*Figure 5*). Etant donné que Yolo ne peut être appliqué que sur des images dont la taille est la même que celle utilisée dans l'entraînement, à savoir 432×480, nous avons procédé au découpage de notre image originale en 8×5=40 sous-images. L'application de Yolo sur chacun de ces morceaux d'image donne lieu à une liste de vecteurs où chaque vecteur représente un rectangle de délimitation (bounding box) délimitant éventuellement un arbre. Ce vecteur s'écrit comme suit :

$$[\ x_b \quad , \quad y_b \ , \quad w_b \ , \quad h_b]$$

Avec (x_b, y_b) la position absolue du rectangle de délimitation dans la sous-image, alors que w_b *et* h_b sont respectivement la largeur et la hauteur absolues de rectangle. Ces informations contenues dans le vecteur sont alors transformées pour pouvoir tracer le rectangle de détection dans l'image d'origine de taille 1260×3840.

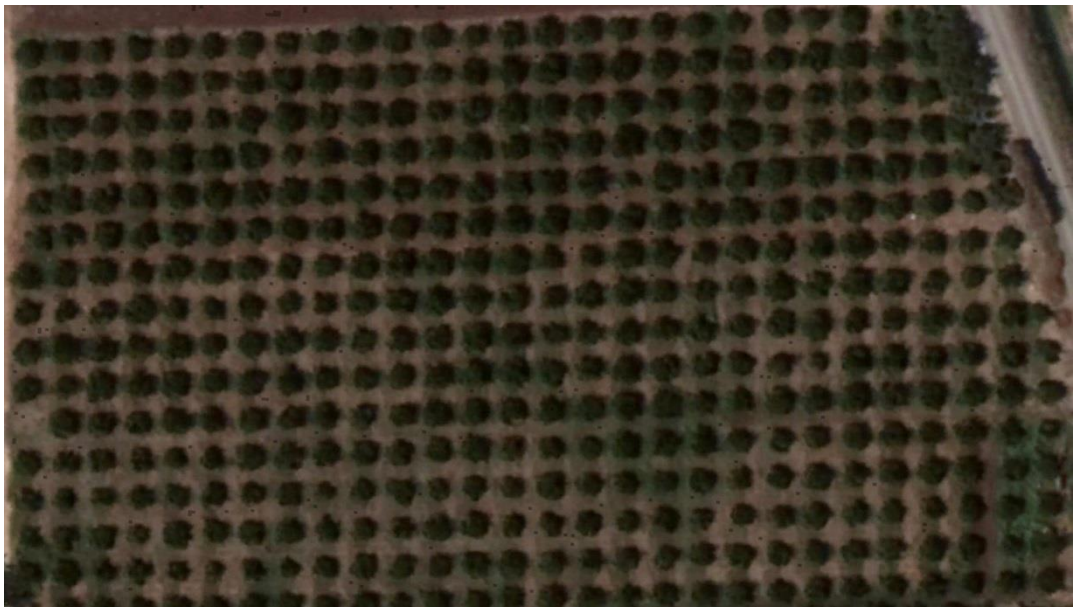

Figure 5. Image de verger d'agrumes (1260×3840).

Le diagramme de flux (Figure 6) illustre l'enchaînement des opérations liées à l'application de Yolo sur une image de grande taille.

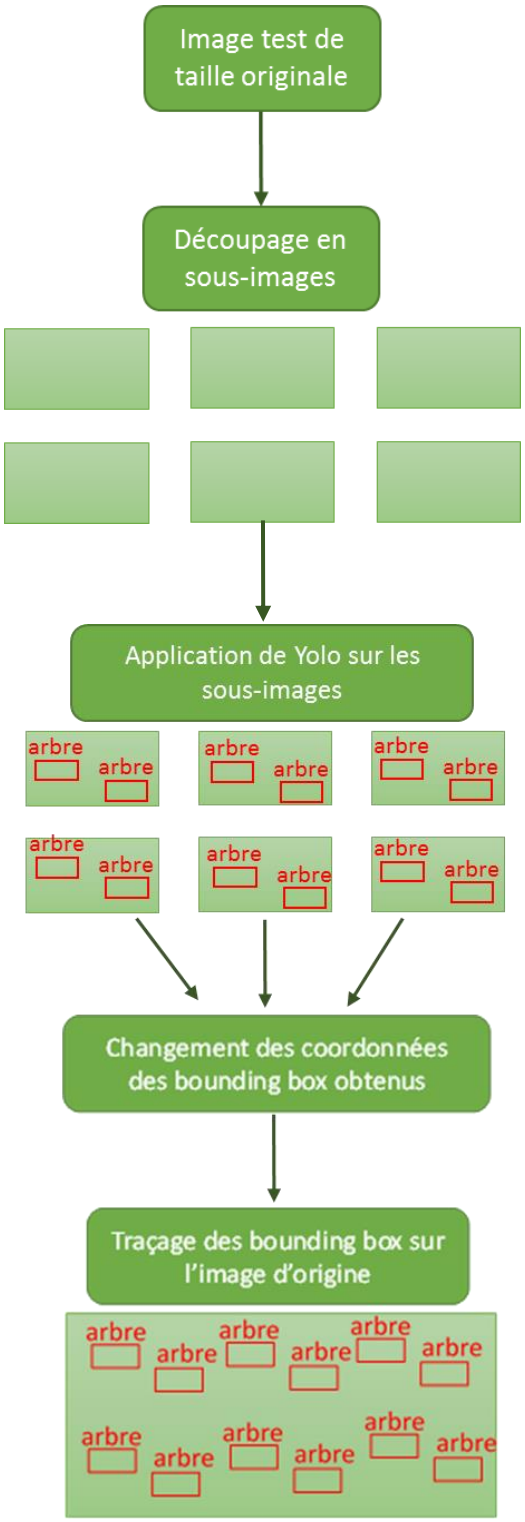

Figure 6. Diagramme de flux des opérations relatives à l'application de Yolo sur une image.

2.5. Evaluation de la détection

Pour évaluer la performance de la technique proposée sur les données d'entraînement et de validation, nous avons conduit un test de détection d'arbres dans l'image de verger ci-dessus (Figure 5). Nous prenons comme référence les arbres annotés (ou labellisés) précédemment dans ce verger. Nous appelons :

- TP (True Positive) les arbres correctement détectés,
- FN (False Negative) les arbres omis par l'algorithme,
- FP (False Positive) les fausses détections (aucun arbre dans le bounding box).

La précision (P) et le taux de détection (R) sont alors calculés par les formules (1) et (2). La précision (P) décrit l'exactitude des objets détectés et comment l'algorithme gère les FP ; le taux de détection (R) décrit la capacité du modèle à détecter la totalité des arbres dans le verger et comment il gère les FN.

$$P = TP/(TP + FP), \quad (1)$$

$$R = TP/(TP + FN), \quad (2)$$

3. Résultats

L'image résultante de l'application du détecteur d'arbres que nous avons conçu est présentée dans la Figure 7. Les rectangles en rouge sont ceux générés par Yolo, les rectangles en bleu ont été tracés manuellement pour pouvoir visualiser les arbres omis (FN). La croix sur un rectangle représente une fausse détection (FP), alors que le cas des rectangles rouges se trouvant adjacents sur un même arbre (Figure 8) est compté comme une bonne détection (voir 4. Discussion).

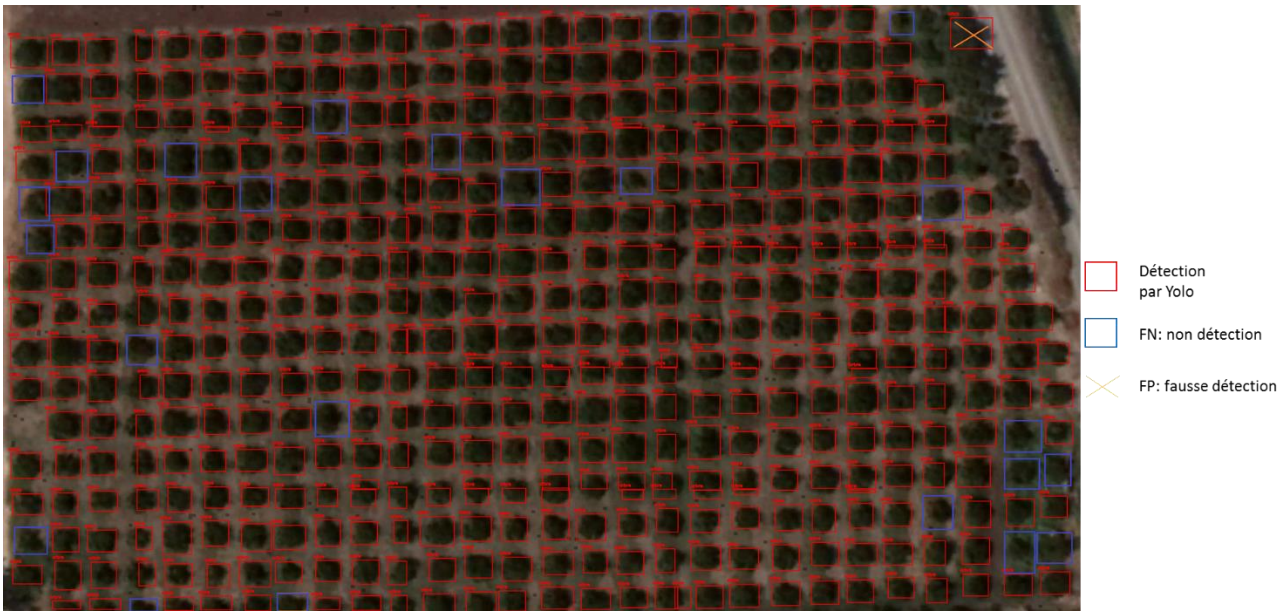

Figure 7. Résultat de détection.

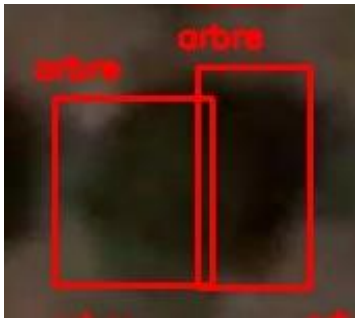

Figure 8. Double détection.

L'évaluation de ces résultats par les mesures de la section *2.5* est résumée dans le tableau ci-dessous. Il convient de noter que le modèle a généré une précision de 99.6% grâce à un nombre de fausse détection très petit (FP=1) par rapport au nombre d'arbres correctement détectés (TP=300). Il est également intéressant de souligner le taux de détection de 92.6% qui se traduit par le fait que le modèle a omis 24 arbres parmi les 324 arbres à détecter dans le verger.

Tableau 2. Mesures de précision.

Arbres labellisés	**TP**	**FN**	**FP**	**P**	**R**
324	300	24	1	99.6%	92.6%

4. Discussion

En dépit des résultats prometteurs, l'approche développée dans cet article présente certaines limites. Commençons par exposer le problème illustré dans la figure (Figure 8), et qui est une conséquence du découpage de l'image d'origine en fenêtres disjointes sur lesquelles est appliqué notre détecteur. Pour remédier à ce problème, nous recommandons l'utilisation d'une fenêtre glissante (sliding window) sur l'image d'origine de façon qu'il y ait un petit chevauchement entre deux fenêtres consécutives (Figure 9). Après application du détecteur, la zone de chevauchement présentera des arbres détectés deux fois avec deux rectangles de délimitation presque superposés (Figure 9). L'idée ici est d'éliminer un des rectangles dès qu'il y a un chevauchement supérieur à un certain seuil entre les deux rectangles.

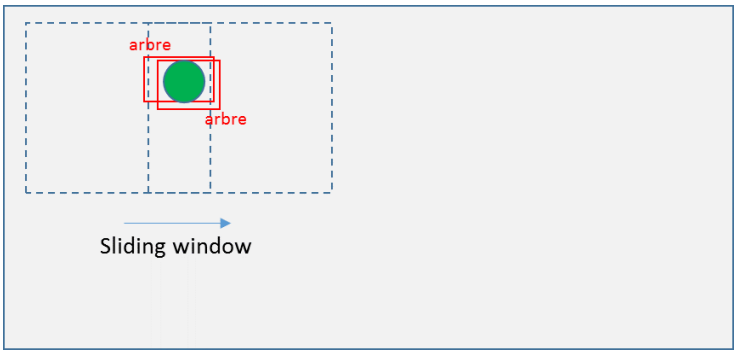

Figure 9. principe de la détection par fenêtre glissante.

Afin d'éviter le « surapprentissage » et garantir des performances similaires du modèle sur d'autres types d'images de vergers, il est indispensable d'utiliser une large base de données qui soit de l'ordre de milliers d'images. Cela est l'un des inconvénients les plus connus des algorithmes d'apprentissage profond, qui est le fait de nécessiter une large base de données pour l'entraînement.

Un prétraitement des images constituant la base de données peut s'avérer nécessaire dans le cas d'images bruitées. En effet, un taux de bruit élevé pourrait nuire à a phase d'entraînement.

Dernièrement, dans la phase d'annotation il est recommandé de créer un cadre carré autour de l'arbre labellisé et d'éviter au maximum les cadres rectangulaires. Ceci joue également dans la phase d'entraînement du modèle.

5. Conclusion

Une gestion précise et efficace des cultures d'agrumes requiert des technologies de détection des arbres individuels dans les vergers. Dans cet article, une approche basée sur de l'imagerie satellitaire de très haute résolution et l'Intelligence Artificielle (apprentissage profond) a été développée dans le but d'aider les gestionnaires de vergers à recenser et localiser les arbres, voire à estimer la densité de plantation de leurs exploitations ainsi que d'évaluer qualitativement leurs productivités et l'ampleur

des ressources humaines et logistiques à mettre en œuvre lors d'opérations comme l'épandage de fertilisants ou la cueillette des fruits. La technique proposée a estimé les arbres dans une image faisant partie de la base de données avec une précision de 92.6%, ce qui démontre la faisabilité et l'intérêt de l'approche développée. Néanmoins, il reste à procéder à des optimisations notamment pour remédier au problème de double détection, ainsi que d'augmenter significativement le nombre d'images utilisées dans l'entraînement afin que notre méthode soit généralisable sur d'autres bases de données. Comme l'exploitation de l'imagerie satellitaire pour les applications agricoles est de plus en plus répandue dans le monde, cette technique pourrait jouer un rôle crucial dans le traitement et l'analyse des cartes de vergers et présenter des informations précieuses aux gestionnaires et chercheurs sur le terrain.

Financement: Cette étude a été financée par "Les Domaines Agricoles" partenaire du ''Moroccan Foundation for Advanced Science, Innovatin and Research".

Références

[1] M. Lafaye, M. Lesage and P. Claquin, "Le recours aux satellites en agriculture : évolutions récentes et perspectives," Ministère de l'Agriculture, de l'Agroalimentaire, et de la Foret, Centre d'études et de prospective., 2014.

[2] A. Ozdarici-Ok, "Automatic detection and delineation of citrus trees from VHR satellite imagery", *International Journal of Remote*, 2015.

[3] D. Koc-Sana, S. Selimb, N. Aslanb and B. T. Sanc, "Automatic Citrus tree extraction from UAV images and digital surface models using circular Hough transform", *Computers and Electronics in Agriculture*, 2018.

[4] J. C. Neto and J. I. Miranda, "A genetic algorithm for citrus tree counting and canopy diameter estimation", 2009.

[5] M. Zortea, M. M. G. Macedo and A. B. Mattos, "Automatic Citrus Tree Detection from UAV Images based on Convolutional Neural Networks", in *3DGEO*.

[6] Yiannis Ampatzidis and Victor Partel, "UAV-Based High Throughput Phenotyping in Citrus Utilizing Multispectral imaging and Artificial Intelligence", *Remote Sensing*, 2019.

[7] M. F. G. Philippe Maillarda, "Detection And Counting Of Orchard Trees From V Images Using A Geo Metrical-Optical Model", in *XXIII ISPRS Congress*, Prague, 2016.

[8] R. Joseph and F. Ali, "YOLO", [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed Juin 2019].