Détection et suivi des caprins par deep learning

Djahlin Hervé Nikue Amassah, Bruno Emile, Sylvie Treuillet, Xavier Desquesnes

Détection et suivi des caprins par deep learning

D. NIKUE AMASSAH* B. EMILE*
X. DESQUESNES* S. TREUILLET*

* Laboratoire PRISME - Université d'Orléans

IUT de l'Indre, 2 avenue F. Mitterrand, 36000 Châteauroux, France
djahlin.nikue-amassah@etu.univ-orleans.fr

Résumé

Ce travail est réalisé dans le cadre du projet AniMov qui consiste à mettre en place un système de surveillance vidéo des comportements des animaux en situation d'élevage. L'objectif principal du projet est de fournir aux éleveurs un outil d'analyse capable de produire des indicateurs précis pour piloter l'alimentation et la reproduction mais aussi de détecter les cycles d'activité et les situations anormales. Notre système d'analyse automatique repose sur un ensemble de caméras placées dans l'enclos ou le bâtiment d'élevage et des algorithmes d'apprentissage machine. Le travail présenté dans cet article constitue la première partie de ce projet : la détection et le suivi des caprins à travers les vidéos. Nous avons mis en œuvre une méthode de suivi basée sur la détection. Pour la détection, nous avons utilisé YOLO v4, une architecture de détection à une étape, après une comparaison avec le modèle Faster R-CNN. Nous avons utilisé une base de 796 images dont 646 pour l'entraînement et 150 pour le test. Ensuite le filtre de Kalman, couplé avec l'algorithme hongrois, est appliqué pour le suivi. L'évaluation de notre méthode de détection sur les données de test nous donne une précision moyenne de 86.74% pour la classe chevre_debout et 90.56% pour la classe chevre_couche.

Mots Clef

Détection d'objets, Suivi d'objets multiples, apprentissage profond.

Abstract

This work is performed within the project AniMov, which consists in building a video surveillance system of animal behaviors in a livestock situation. The main objective of the project is to provide farmers with an analysis tool capable of producing precise indicators to control feeding and reproduction but also to detect activity cycles and abnormal situations. Our automatic analysis system is based on a set of cameras placed in the pen or farm building and machine learning algorithms. The work presented in this article is the first part of this project : detection and monitoring of sheep. We have implemented a monitoring method based on detection. For detection, we used YOLO v4, a one-step detection architecture, after a comparison with the Faster R-CNN model. We built a dataset of 796 images, 646 for training and 150 for testing. Then the Kalman filter, combined with the Hungarian algorithm, is applied for tracking. The evaluation of our detection method on the test data gives us an average accuracy of 86.74% for the "standing_goat" class and 90.56% for the "lying_goat" class.

Keywords

Object detection, Multiple object tracking, deep learning.

1 Introduction

Le projet AniMov vise à apporter aux éleveurs un outil d'analyse automatique de leur troupeau grâce à un système de vidéo-surveillance temps réel. Les éleveurs veulent connaître le comportement des animaux afin de disposer d'indicateurs précis pour piloter l'alimentation et la reproduction du troupeau.

L'observation permanente et directe des animaux par un humain n'est évidemment pas envisageable pour des raisons de coût mais également d'altération du comportement des animaux. La présence humaine influence le comportement animal et interdit bien souvent d'observer certaines situations spécifiques. C'est dans ce contexte que nous développons un système de vision permettant d'analyser automatiquement le comportement animal pour suivre les cycles d'activité et les situations anormales. Dans un premier temps, nous effectuons la détection et le suivi des animaux pour ensuite réaliser la reconnaissance des comportements. Le travail présenté dans cet article porte principalement sur la détection et le suivi des animaux, plus précisément des chèvres.

La détection et le suivi d'objets multiples restent un grand défi dans le domaine de la vision par ordinateur. Cette tâche est encore plus complexe lorsqu'il s'agit de suivre des troupeaux de chèvres dans des enclos. En effet, les chèvres étant très similaires, il est difficile de trouver des caractéris-

FIGURE 1 – Exemple de troupeau de chèvre en conditions d'élevage réelles

tiques pour les séparer efficacement par rapport à un suivi de personne où on pourrait utiliser des caractéristiques de couleur par exemple. La grande densité des chèvres dans les enclos augmente les cas d'occultations et peut entraîner des échecs de détection. La construction de bases d'entraînement et de test est également très couteuse par manque de bases vidéos existantes dans ce contexte.
La section 2 présente un état de l'art des méthodes de détection et de suivi des animaux en situation d'élevage par analyse vidéo. Dans la section 3 nous présentons notre méthode de suivi basée sur la détection dans les vidéos et dans la section 4 les résultats obtenus en comparaison à la vérité terrain étiquetée manuellement par l'homme.

2 Travaux existants

Au cours des deux dernières décennies, les chercheurs ont étudié une variété de méthodes et de technologies à base de caméras vidéo pour détecter et suivre les animaux en situation d'élevage. En utilisant l'imagerie couleur traditionnelle et la soustraction d'arrière-plan, les chercheurs ont conçu des méthodes de suivi dans des environnements contraints où les porcs marchent individuellement devant la caméra [2].
Les méthodes classiques de suivi d'objets multiples montrent des limitations dans le contexte d'un troupeau d'individus très similaire comme c'est le cas en situation d'élevage. Pour surveiller plusieurs animaux simultanément, il est nécessaire de les segmenter à la fois de l'arrière-plan et les uns des autres ; une tâche difficile compte tenu de leur tendance à se regrouper (voir figure 1). Kashiha et al. [4] ont proposé une méthode automatisée pour identifier les porcs marqués dans un enclos en utilisant des techniques de reconnaissance des formes. Dans un premier temps, une segmentation est effectuée à l'aide d'un filtre gaussien 2D (pour débruiter l'image) suivi du seuillage d'Otsu. Ensuite, les marques sur les porcs ont été extraites selon une méthode de segmentation similaire, qui a ensuite été utilisée pour identifier le motif sur la base d'une description de Fourier. Cette méthode exige que les animaux soient marqués individuellement, ce qui n'est pas toujours possible d'un point de vue commercial. Elle est également très sensible aux changements d'éclairage.
Pour cela, des informations sur la profondeur ont été introduites dans le suivi des animaux en utilisant des caméras de profondeur : suivi multiple en 3D. Cette technique est appliquée pour le suivi des porcs où la caméra de profondeur Kinect v2 a été utilisée pour l'acquisition des images [5]. La caméra placée verticalement requiert un calibrage manuel du système où l'utilisateur sélectionne des points d'angle définissant les limites de l'enclos, de la mangeoire, de l'abreuvoir, du tapis chauffant ainsi que la position de chaque porc. Les nuages de points 3D et un tracking d'ellipsoïdes sont utilisés avec un filtre de Kalman pour estimer la position et orientation de chaque porc. D'autres chercheurs ont également proposé un système de suivi en 3D en utilisant l'algorithme de croissance de région. Les porcs ont été suivis en reliant les détections dans les images consécutives. L'algorithme hongrois, comme décrit dans [6], a été utilisé pour l'association entre les images en effectuant une optimisation combinatoire de toutes les affectations de porc à porc.
Bien que les systèmes de vision existants basés sur des caméras vidéo de profondeur aient obtenu certains succès dans la détection et le suivi des animaux en élevage, ils présentent certains inconvénients. La caméra de profondeur Kinect a une portée de 4 m et un champ de vision (horizontal 58,5 degrés et vertical 45,6 degrés) limités. De plus,la précision des données de profondeur est très sensible à la position de la caméra [3].
Dans [3] les auteurs ont mis en place un logiciel de détection et de suivi des porcs à base de caméras couleurs 2D. Ils ont comparé les architectures de détection d'objets R-FCN, Faster R-CNN et SSD. Le Faster R-CNN et le R-FCN ont montrés de bonnes précisions de détection, mais ils sont moins rapides que le SSD. L'architecture SSD a été retenue pour la détection. Ensuite, pour le suivi, le filtre de corrélation discriminant (DCF : Discriminative Correlation Filters) est utilisé avec l'algorithme hongrois pour l'association des données.
J. Cowton et al. [1] ont proposé une méthode de localisation individuelle automatisée des porcs en utilisant le Faster R-CNN. Pour le suivi, ils ont évalué 2 méthodes : SORT et Deep SORT. La méthode SORT combine le filtre de Kalman et l'algorithme hongrois pour le suivi. Cette méthode ne requiert aucun apprentissage et peut être directement appliquée sur la sortie de la détection. Le Deep SORT, en plus du filtre de Kalman et de l'algorithme de hongrois, utilise une métrique d'association apprise pour déterminer si deux images consécutives contiennent le même porc. Cette métrique est apprise à l'aide d'un modèle CNN de 11 couches. Contrairement à SORT, Deep SORT dépend moins de l'exactitude des détections, bien qu'il exige toujours qu'elles soient de bonne qualité, car il utilise toujours partiellement le filtre de Kalman pour prendre des décisions d'association [1]. Leur méthode de suivi fut évaluée en utilisant le métrique MOTA (Multi-

Object Tracking Accuracy) [15]. Ils ont obtenu 95.1 % de MOTA avec SORT et 92.1 % avec Deep SORT.

D'autres chercheurs ont utilisé une architecture à base de R-CNN et LSTM pour le suivi des vaches [7]. La détection a été effectuée avec l'architecture de détection à 2 étapes Faster R-CNN et une architecture LRCN (Long-term Recurrent Convolutional Networks) pour le suivi. Dans le LRCN, Les caractéristiques visuelles des images vidéo d'entrée $\{f_1, f_2, ..., f_n\}$ sont extraites par un CNN pour être introduites dans une couche LSTM qui produit finalement une prédiction d'identité.
Dans [16], les auteurs nous présentent un résumé sur les méthodes à base du deep learning pour le suivi des personnes. Ces méthodes sont regroupées en deux catégories : les méthodes "online" (temps réel) et les méthodes "offline" (hors ligne). Malgré le fait que les deux composantes sont dépendantes l'une de l'autre, les travaux antérieurs conçoivent souvent les modules de détection et d'association de données séparément, qui sont entraînés avec des objectifs distincts. Par conséquent, il est impossible de rétro-propager les gradients et d'optimiser l'ensemble du système de suivi. Pour cela, les auteurs dans [17] proposent une méthode de détection et suivi multi-objets simultané à l'aide de réseaux de neurones de graphes (GNN). L'idée principale est que les GNN peuvent modéliser les relations entre des objets de taille variable dans les domaines spatial et temporel, ce qui est essentiel pour l'apprentissage de caractéristiques discriminantes pour la détection et l'association de données [17]. Les expérimentations ont été effectués sur les données de références 2DMOT15/MOT16/MOT17/MOT20 [1].
Les méthodes existantes de suivi par détection adoptent des heuristiques simples, telles que la similarité spatiale ou d'apparence. Ces méthodes, malgré leur performance, sont trop simples et insuffisantes pour modéliser des variations complexes, comme le suivi par occlusion [18]. Récemment, une approche de suivi d'objets multiples basées sur les réseaux "Transformer" a été proposée pour apprendre à modéliser la variation temporelle à long terme des objets. Dans [18], les auteurs proposent un réseau d'agrégation temporelle combiné à un entraînement multi-trame pour modéliser la relation temporelle à longue portée. Le réseau est construit à base de "transformer DETR" [19].

3 Méthode

3.1 Mise en oeuvre

La méthode mise en oeuvre pour le suivi des chèvres comprend 2 étapes : la détection et le suivi d'objets multiples.
Pour la détection nous avons entraîné et comparé 2 modèles : le Faster R-CNN et YOLO v4. Le Faster R-CNN est l'un des meilleurs modèles de détection en 2 étapes avec une architecture intégrant un réseau de propositions de régions (RPN) qui génère des régions d'intérêt (RoI) initiales pour l'apprentissage ultérieur. L'entraînement du Faster R-CNN se fait en 2 étapes principales : dans la première étape, les caractéristiques extraites par un CNN sont utilisées pour générer des régions à l'aide d'une fenêtre glissante sur la carte des caractéristiques, avec différentes échelles et rapport d'aspects. Dans la deuxième étape, les propositions de boîtes, ainsi que la carte de caractéristiques intermédiaire générées dans l'étape de proposition de RoI sont utilisées pour entraîner une somme de poids de coût de classification et de régression des boîtes englobantes [14]. Ce type d'architecture donne de bonnes précisions mais reste très lent pour un système temps réel.

Dans notre système, la détection est réalisée dans chaque image par le modèle YOLO v4 [8] qui est plus adapté pour un système temps réel. Il s'agit d'une architecture de réseau de neurone convolutif permettant la détection d'objets en une seule étape, ce qui le rend très rapide par rapport aux architectures de détection en 2 étapes. Avec YOLO v4 un seul CNN est appliqué sur l'image et prédit simultanément les boîtes englobantes, les scores de confiance et les probabilités de classe pour ces boîtes. Bien avant YOLO v4, il y a eu YOLO 9000, YOLO v2 et YOLO v3. Récemment, l'article YOLO v4 [8] a été publié et a montré de très bons résultats, en termes de rapidité et de précision, par rapport aux autres détecteurs d'objets.

Chaque objet détecté est identifié par un numéro (ID) puis un algorithme de suivi permet d'associer les ID image par image et générer les trajectoires de chaque animal. En raison du nombre important d'animaux dans les enclos, une chèvre peut être masquée par une autre dans quelques images et réapparaître ensuite. Par conséquent, le suivi doit être suffisamment robuste pour gérer une occultation partielle ou totale. D'autres problèmes tels que les mouvements brusques ou le bruit dans les séquences d'images, les changements d'apparence de l'objet et de la scène, l'éclairage, etc. doivent être traités lors de la conception d'un suivi visuel robuste. Dans notre cas, nous avons couplé le filtre de Kalman avec l'algorithme hongrois pour l'association des données. L'association des ID exploite des mesures de similarité basées sur l'intersection sur union (IoU) des boîtes englobantes. En effet, nous faisons l'hypothèse que si la boîte englobante de l'image courante chevauche la précédente, c'est probablement la même puisque les chèvres ne sont pas très mobiles en général. Cette approche rejoint un peu la méthode de suivi SORT [9]. La figure 2 représente le diagramme de notre méthode.

3.2 Jeu de données

Afin d'entrainer le modèle YOLO v4 et évaluer notre méthode, nous avons constitué une base d'images d'entraînement et de test. Les images sont extraites des vidéos de caméras de surveillance dans deux fermes dans l'Indre. Les

1. https ://motchallenge.net/data/MOT17/

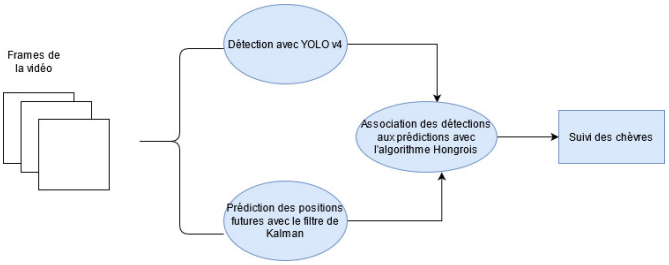

FIGURE 2 – Méthode proposée de suivi de chèvres en troupeau

FIGURE 3 – Annotation des images

caméras couleur haute résolution 1920x1080 pixels sont placées dans chaque angle de l'enclos pour avoir différents points de vue. Nous avons constitué un ensemble de 796 images de jour et de nuit. Les données d'entraînement et de test contiennent des exemples de tous les scénarios difficiles rencontrés.

Pour effectuer la détection des chèvres avec YOLO v4, une annotation des images avec des boîtes englobantes est nécessaire. Ces boîtes englobantes constituent notre vérité terrain. Pour l'annotation nous avons utilisé l'outil **LabelImg** (voir figure 3). L'annotation génère des fichiers txt (format yolo) pour chaque image contenant les coordonnées des boîtes englobantes. Nous avons considéré 2 classes : **chevre_debout** et **chevre_couche**. La base d'images est construite de façon à équilibrer la proportion de chèvres présente dans chaque classe.

TABLE 1 – Configuration de la base d'images

Nombre d'images total	796
Images d'entraînement	646
Images de test	150
Nombre de classes	2

3.3 Détection

La plupart des modèles de détection modernes nécessitent plusieurs GPU pour l'entraînement avec une grande taille de mini-batch, et le faire avec un seul GPU rend l'entraînement très lent et peu pratique. YOLO v4 résout ce problème en créant un détecteur d'objets qui peut être entraîné

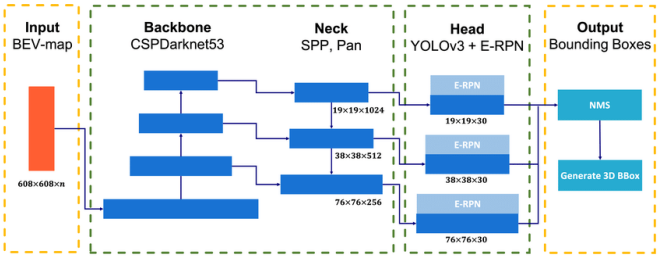

FIGURE 4 – Architecture de YOLO v4 [12]

sur un seul GPU avec une taille de mini-batch plus petite. L'architecture YOLO v4 comprend (figure 4) :

- Le réseau principal (backbone) **CSPDarknet53**, utilisé comme extracteur de caractéristiques et utilise les connexions Cross-Stage-Partial (CSP) avec une activation Mish ;
- Ensuite nous avons le **SPP (Spatial pyramid pooling)** et le **PANet (Path Aggregation Network)** qui sont des couches supplémentaires qui s'intercalent entre le réseau dorsal et la tête du détecteur. Elles sont utilisées pour extraire différentes cartes de caractéristiques à différentes échelles du réseau dorsal ;
- La dernière partie dénommée la tête du réseau comprend 2 sorties : une sortie pour la classification et une sortie pour la régression des boîtes englobantes.

Pour améliorer la précision de détection, YOLO v4 utilise deux nouvelles méthodes d'augmentation de données : la méthode **mosaïque** et la **SAT (Self-Adversarial Training)**. La méthode Mosaïque concatène 4 images d'entrée en une seule par rapport à la méthode CutMix qui elle ne concatène que 2 images. Cela réduit le besoin d'une grande taille de mini-batch.

Le principe de fonctionnement de YOLO v4 est le même que YOLO v3 [10]. La nouveauté se trouve dans l'utilisation des connexions Cross-Stage-Partial et une fonction d'activation Mish dans le réseau principal, l'ajout de blocs supplémentaires (SPP et PANet) et des techniques spécifiques d'augmentation de données lors de l'entraînement.

Pour l'entraînement de notre détecteur, nous avons utilisé l'apprentissage par transfert en chargeant les poids préentraînés du modèle YOLO v4 : yolov4.conv.137. Une fois les poids chargés, l'entraînement a été effectué avec toutes les couches du réseau.

La plupart des modèles de détection d'objets existants utilisent l'erreur quadratique moyenne de la régression des boîtes englobantes (pour prédire les coordonnées de la boîte englobante) comme fonction de coût. YOLO v4 utilise la fonction de coût CIoU qui introduit deux nouveaux concepts : la distance des centres des boîtes englobantes prédites de la vérité terrain et le rapport d'aspect. L'équation de la fonction de coût CIoU est la suivante :

$$L_{CIoU} = 1 - IoU + \frac{\varphi^2(b, b^{gt})}{c^2} + \alpha v \qquad (1)$$

$$\alpha = \frac{v}{(1 - IoU) + v} \quad (2)$$

b et b^{gt} correspondent respectivement au centre de la boîte englobante prédite et celui de la boîte englobante de la vérité terrain ; $\varphi(.)$ est la distance euclidienne ; c est la longueur de la diagonale de la plus petite boîte englobante couvrant les deux boîtes ; α est un paramètre de pondération positif, et v mesure la cohérence du rapport d'aspect par l'équation :

$$v = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2. \quad (3)$$

w, h et w^{gt}, h^{gt} correspondent respectivement aux dimensions (largeur et hauteur) des boîtes englobantes prédites et de la vérité terrain.

TABLE 2 – Hyperparamètres pour l'entraînement

Batch size	64
Subdivision	16
Taux d'apprentissage	0.001
Taille des entrées	416x416 pixels
Momentum	0.9
Decay	0.005
Nombre de classes	2

3.4 Suivi

Nous avons réalisé le suivi des chèvres en implémentant le filtre de Kalman pour l'estimation d'état et l'algorithme hongrois pour l'association des données. Le filtre de Kalman est une solution bayésienne pour prédire l'état d'un processus linéaire discret. Dans le suivi d'objet, il permet de prédire les positions futures de l'objet en fonction de la position actuelle. Il comprend 2 phases principales : la prédiction et la mise à jour. A la phase de prédiction, les informations de l'état précédent sont utilisées pour prédire l'état courant et à la mise à jour, les observations sont utilisées pour corriger la prédiction. Il existe d'autres filtres tels que le filtre à particule, le filtre à corrélation et le filtre Kalman étendu. Nous avons choisi d'implémenter dans un premier temps le filtre de kalman [11]. D'autres techniques de suivi bayésienne peuvent être trouver dans [20].

Des fonctions mathématiques permettent de détecter la moyenne et la covariance d'état : dans notre cas le vecteur d'état E_t est modélisé comme suit : $E_t(x_t, y_t, v_{xt}, v_{yt}, w_t, h_t)$ où x_t et y_t représentent les coordonnées du centre de la boîte englobante de l'objet, v_{xt} et v_{yt} sa vitesse et w_t, h_t sa largeur et sa hauteur à l'instant t. Ces paramètres sont initialisés au début par des valeurs aléatoires sauf la vitesse qui est fixée à 0 et sera ensuite estimée par le filtre de Kalman. Les autres paramètres sont mis à jour au fur et à mesure des détections. La covariance est initialisée par un nombre arbitraire et ajustée tout au long du traitement grâce aux mesures qui seront effectuées par le filtre de Kalman. Plus la covariance est élevée, plus l'incertitude est grande.

L'algorithme hongrois est utilisé pour associer les détections existantes aux prédictions effectuées par le filtre de Kalman. Nous avons une matrice qui nous indique la correspondance entre la détection et la prédiction Kalman. Nous avons utilisé la fonction linear_assignment() de la bibliothèque scipy qui implémente l'algorithme hongrois. Cet algorithme utilise un graphe bipartite pour trouver, pour chaque détection, le score d'association le plus bas dans la matrice.

4 Résultats expérimentaux

4.1 Détection

Tous les entraînements et évaluations ont été réalisés à l'aide d'un processeur Intel(R) Core(TM) i9-9900K CPU à 3.60GHz, de 2 GPU Nvidia Titan RTX et de 64 Go de RAM DDR3.

Nous avons comparé 2 architectures de détection d'objets : Faster R-CNN et YOLO v4. La valeur moyenne de la précision moyenne (mAP), la courbe PR et le nombre d'images par seconde (FPS) sont les métriques utilisées pour évaluer les 2 architectures de détection d'objets (tableau 3). La mAP (mean Average Precision) représente la valeur moyenne des précisions moyennes de chaque classe. La précision représente le pourcentage des détections correctes par rapport au nombre total de détection et le rappel représente le pourcentage des détections correctes par rapport à la vérité terrain.

Une prédiction est considérée comme vraie (TP : True Positive) si l'IoU est supérieur à un seuil donné (0.5 dans notre cas), et fausse (FP : False Positive) si c'est inférieur. La vérité terrain non détecté correspond au Faux Négatif (FN : False Negative) [13].

$$p = \frac{TP}{TP + FP}$$
$$r = \frac{TP}{TP + FN}$$

L'AP (précision moyenne) est calculée comme la moyenne des 11 valeurs de précision pour les valeurs de rappel = {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}.

TABLE 3 – Comparaison de YOLO v4 et Faster R-CNN

Categorie (field)	*YOLO v4*	*Faster R-CNN*
mAP@.5	88.65%	72.41%
chevre_debout AP@.5	86.74%	68.69%
chevre_couche AP@.5	90.56%	76.12%
FPS	3.49	0.95

La figure 5 nous montre la courbe PR (précision/rappel) de la classe "chevre_debout" à droite et à gauche la classe "chevre_couche", obtenue avec l'architecture YOLO v4 sur les données de test et la figure 6 celle obtenue avec l'architecture Faster R-CNN (classe "chevre_debout" à droite

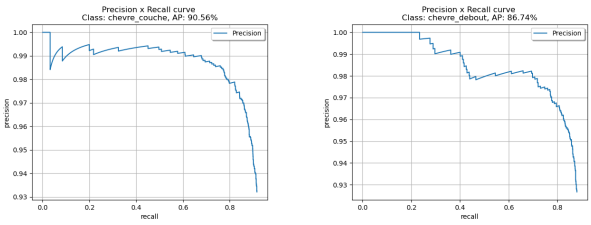

FIGURE 5 – Courbes PR avec YOLO v4

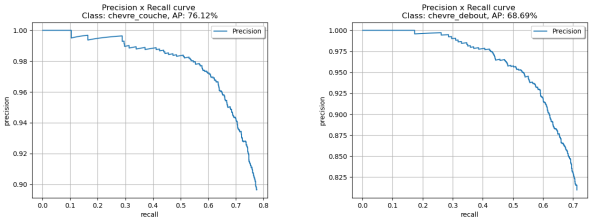

FIGURE 6 – Courbes PR avec Faster RCNN

et classe "chevre_couche" à gauche). D'après ces résultats, on peut observer que YOLO v4 a une performance de détection meilleure que le Faster R-CNN. YOLO v4 produit les meilleures boîtes englobantes de détection avec des scores de confiance élevés comme on peut le voir sur la figure 7.

4.2 Suivi

Nous avons testé notre méthode de suivi sur quelques vidéos de troupeau de chèvres et les résultats obtenus sont assez intéressants. Nous avons remarqué que la détection influence beaucoup le résultat de suivi. En effet, lorsque la détection échoue dans certains cas (quand les chèvres sont regroupés par exemple) l'ID suivi est soit perdu soit échangé avec un autre. Une meilleure détection produira un suivi plus robuste. Nous avons testé le suivi sur des vidéos de courtes durées (1 à 2 minutes). Sur la figure 8 (dans le cercle noir) nous pouvons observer un événement d'occultation où la chèvre ID 6 croise avec la chèvre ID 22. Dans l'image suivante, après l'occultation, les chèvres n'ont pas perdu leur ID (voir figure 9). Cela montre que

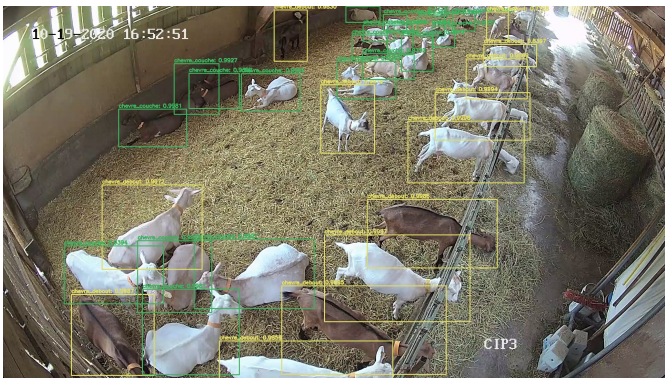

FIGURE 7 – Sortie de détection avec YOLO v4

FIGURE 8 – Suivi en cas d'occultation

FIGURE 9 – Suivi en cas d'occultation : image suivante

notre système de suivi arrive à gérer assez bien un cas d'occultation partielle. Nous ferons après une évaluation de notre système de suivi en utilisant les métriques MOT (Multiple Object Tracking) qui nécessite la construction de vérités terrain. La construction de la vérité terrain et l'implémentation d'une méthode pour évaluer notre détecteur sont un peu couteuses et nécessitent plus de temps. Dans les travaux à venir nous allons construire une base de vérité terrain qui contiendra des vidéos annotées afin d'évaluer quantitativement notre méthode et aussi de tester d'autres approches pour le suivi.

5 Conclusion et travaux à venir

Dans cet article, nous présentons notre méthode de détection et de suivi automatique d'objets multiples pour surveiller des caprins dans un environnement difficile. Afin de résoudre les principaux problèmes (par exemple, la variation de la lumière, les apparences similaires, et l'occultation), qui peuvent conduire de manière significative à des échecs de détection et de suivi, nous proposons une méthode qui couple un détecteur à base de réseau de neurones convolutif et un filtre linéaire Kalman. Après une comparaison entre YOLO v4 et le Faster R-CNN, nous avons retenu YOLO v4, une architecture de détection à une étape qui, avec des caractéristiques hiérarchiques de haut niveau sémantique provenant de plusieurs couches convo-

lutives et à des échelles différentes, produit une meilleure précision et un temps de détection plus rapide. Les coordonnées des boîtes englobantes issues de la détection sont ensuite suivies grâce au filtre de Kalman et l'algorithme hongrois est utilisé pour l'association des données afin de gérer les cas d'occultations. Lors de l'association, l'IoU des boîtes englobantes est utilisé comme mesure de similarité. Nous avons utilisé une base de 796 images dont 646 pour l'entraînement et 150 pour le test. L'évaluation de notre méthode de détection sur les données de test nous donne une précision moyenne de 86.74% pour la classe chevre_debout et 90.56% pour la classe chevre_couche.
Ces résultats montrent que notre système de détection est assez performant mais pourrait être amélioré. Ainsi, dans les travaux à venir, nous allons augmenter notre base d'images afin d'améliorer les performances de notre détecteur. Pour le suivi, nous prévoyons tester d'autres filtres pour l'estimation d'état et tester aussi les approches récentes de suivi à base de réseau de neurones.

Remerciements

Ce travail fait partie du projet FC9513/APR IR 2019 AniMOV Animal Movements Observation (ANIMOV), soutenu par la Région Centre-Val de Loire (France). Les auteurs tiennent à remercier le Conseil Régional du Centre - Val de Loire pour son soutien. Nous tenons également à remercier les partenaires de ce projet : l'Université d'Orléans, les laboratoires PRISME et INRAE, les sociétés tekin et ACTI'COM, le Pôle Capteurs et Automatismes et la chambre d'agriculture de l'Indre, le ZOOPARC de Beauval, l'association Beauval Nature pour la Conservation et la Recherche.

Références

[1] J. Cowton I. Kyriazakis J. Bacardit, Automated Individual Pig Localisation, Tracking and Behaviour Metric Extraction Using Deep Learning, *IEEE Access,* Vol. 7, pp. 108049–108060, 2019.

[2] N. M. Lind, M. Vinther, R. P. Hemmingsen, A. K. Hansen, Validation of a digital video tracking system for recording pig locomotor behaviour, *Journal of neuroscience methods*, Vol. 143(2), pp. 123–132, 2005.

[3] N.J.B. McFarlane, C.P. Schofield, Segmentation and tracking of piglets in images *Machine Vision and Applications,* Vol. 8, pp. 187-193, 1995.

[4] L. Zhang, H. Gray, X. Ye, L. Collins, N. Allinson, Automatic Individual Pig Detection and Tracking in Pig Farms *Sensors (Basel) 19-01188,* pp. 1-20, 2019.

[5] M. Mittek, E. T. Psota, L. C. Pérez, T. Schmidt, B. Mote, Health Monitoring of Group-Housed Pigs using Depth-Enabled Multi-Object Tracking, *In Proceedings of International Conference Pattern Recognition, Workshop on Visual observation and analysis of Vertebrate And Insect Behavior,* pp. 4, 2016.

[6] H.W. Kuhn, The hungarian method for the assignment problem, *Naval research logistics quarterly,* Vol. 2, pp. 83-97, 1955.

[7] W. Andrew, C. Greatwood, T. Burghardt, Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning, *In Proceedings of the IEEE International Conference on Computer Vision,* pp.2850-2859, 2017.

[8] A. Bochkovskiy, C. Wang, H. M. Liao, YOLOv4 : Optimal Speed and Accuracy of Object Detection, *Arxiv,* pp. 1-17, 2020.

[9] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple Online and Realtime Tracking, *Arxiv,* pp. 1-5, 2017.

[10] J. Redmon, A. Farhadi, YOLOv3 : An Incremental Improvement, *Arxiv,* pp. 1-6, 2018.

[11] M. Marrón, J.C. García, M.A. Sotelo, M. Cabello, D. Pizarro, F. Huerta, J. Cerro, Comparing a Kalman Filter and a Particle Filter in a Multiple Objects Tracking Application, *In IEEE International Symposium on Intelligent Signal Processing,* pp. 1-6, 2007.

[12] J. Miao, T. Hirakawa, T. Yamashita, H. Fujiyosh, 3D Object Detection with Normal-map on Point Clouds, *In 16th International Conference on Computer Vision Theory and Applications,* pp. 569-576, 2021.

[13] R. Padilla, S. L. Netto, E. A. B. da Silva, A Survey on Performance Metrics for Object-Detection Algorithms, *2020 International Conference on Systems, Signals and Image Processing (IWSSIP),* pp. 237-242, 2021.

[14] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN : Towards Real-Time ObjectDetection with Region Proposal Networks *Arxiv,* pp. 1-14, 2016.

[15] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, K.Schindler, MOT16 : A Benchmark for Multi-Object Tracking, *Arxiv,* pp. 1-12, 2016.

[16] Y. Xu, X. Zhou, S. Chen, F. Li, Deep learning for multiple object tracking : a survey, *in IET Computer Vision,* Vol. 13, pp. 355-368, 2019.

[17] Y. Wang, K. Kitani, X. Weng, Joint Object Detection and Multi-Object Tracking with Graph NeuralNetworks, *Arxiv,* pp. 1-8, 2021.

[18] F. Zeng, B. Dong, T. Wang, C. Chen, X. Zhang, Y. Wei, MOTR : End-to-End Multiple-Object Tracking with TRansformer, *Arxiv,* pp. 1-10, 2021.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-End Object Detection with Transformers, *Arxiv,* pp. 1-26, 2020.

[20] S. Padmavathi, S. Divya, Survey on Tracking Algorithms, *in nternational Journal of Engineering Research Technology (IJERT),* pp. 830-834, 2014.