Projet Recherche Innovation Master - Détection, *tracking*, et identification de joueurs sur des vidéos de football

Lucas Gonzalez–Leclercq
leclercq@telecom-paristech.edu

25 février 2020

Résumé

Ceci est le rapport de mon PRIM (Projet Recherche Innovation Master) avec l'entreprise *Footbar*, dans le cadre de mes études à Télécom Paristech. Le but de ce projet était de mettre au point un système de *tracking* et d'identification des joueurs sur des vidéos de football. Dans ce rapport, en m'appuyant sur la littérature existante, je décris ma démarche, mes expériences et leurs résultats, avant de dégager certains axes d'amélioration.

1 Introduction

La détection et le suivi (*tracking*) de personnes sur des vidéos est un problème classique en vision par ordinateur, et pour lequel on trouve dans la littérature une multitude d'approches et de modèles différents. En particulier, il existe deux principales approches : *tracking-by-detection* and *detection-by-tracking* [1]. Plus récemment, les auteurs de [7] utilisent un résau de neurones convolutionnel pour réaliser le *tracking* et la détection simultanément.

Dans ce projet, j'ai choisi l'approche *tracking-by-detection*, qui consiste à d'abord détecter les joueurs sur chaque image de la vidéo, sous la forme de *bounding boxes* délimitant un joueur. Cela m'a permis de séparer le système en trois étapes : la détection des joueurs, le rassemblement de ces détections

en des *tracklets* (fragments de trajectoires composés de détections successives d'un même joueur), et finalement, la recouvrement des identités via un *clustering* des *tracklets*. C'est également l'approche utilisée dans [8] et [13]. Cette séparation des étapes m'a permis de me concentrer sur la classification non supervisée des détections et de n'utiliser des modèles pré-entrainés que pour la génération des *features*.

Dans tout mon travail, j'ai utilisé le format du *MOT challenge* [9], qui est le suivant :

[frame id, object/player id, x, y, width, length, confidence, -1, -1, -1, appearance features]

2 Détection des joueurs

Dans un premier temps, j'ai utilisé le modèle *YOLO* (*You Only Look Once*) [14] pour générer les *bounding boxes* des joueurs sur chaque *frame*.

FIGURE 1: Détections générées par le modèle *YOLO* (*You Only Look Once*)

3 Extraction des *features*

3.1 Deep appearence vector de *DeepSort*

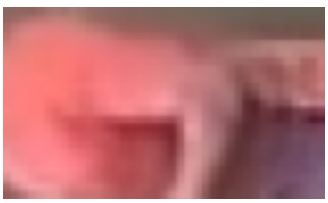
Pour la génération des *tracklets*, j'ai utilisé et modifié une implémentation de l'algorithme *DeepSort* [18]. Les auteurs de *DeepSort* appliquent un réseau convolutionnel pré-entraîné sur un *dataset* de re-identification de piétons, pour l'extraction de ce qu'ils appellent un *deep appearance descriptor* (*DAA*).

3.2 *LOMO*

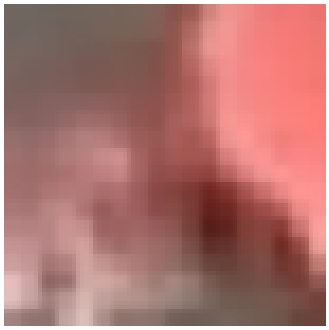
J'utilise aussi les représentations *LOMO* (*Local Maximal Occurrence Representation*) [10]. Ces représentations sont supposées être robustes aux changements de point de vue.

3.3 *OpenPose* + *LOMO*

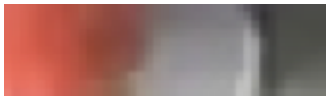
Grace à l'autre groupe, j'ai pu récupérer des estimations des poses en 2D prises par les corps des joueurs. Je comptais utiliser des rapports de longueurs de membres pour créer de nouvelles caractéristiques d'apparence mais je me suis rendu compte qu'il était impossible de déterminer des ratios constants avec une estimation seulement 2D (et non 3D). Toutefois, en m'inspirant de [2], j'ai calculé les représentations *LOMO* sur les parties du corps détectées par *OpenPose* [5], que j'ai associées à des détections avec l'algorithme Hongrois où ma fonction de coût était le nombre de *keypoints* contenus dans les *bounding box* des détections.



(a) Torse



(b) Tête



(c) Haut de jambe

FIGURE 2: Parties du corps extraites à partir des données d'*OpenPose*

4 Tracking

4.1 *DeepSort*

DeepSort [18] est un algorithme de *tracking* combinant une métrique d'apparence avec un modèle du mouvement des joueurs. D'un côté, la distance de *Mahalanobis* entre les nouvelles détections et des filtres de *Kalman* [17] associés aux *tracklets* existants donne de l'information utile pour les prédictions à court-terme. D'un autre côté, la mesure de similarité cosinus entre les DAA (3.1) (qui sont des vecteurs de la sphère unité) est utile pour ré-attribuer les bonnes identités après des occlusions ou des fragmentations sur le long terme, lorsque le mouvement du joueur est moins discriminatif.

Mon travail se base principalement sur l'implémentation open source de *DeepSort*, dont j'ai tenté de *fine-tuner* et de modifier certaines parties pour l'adapter à la problématique du projet [1].

Si chaque détection est caractérisée par un vecteur d'apparence, un *tracklet* est caractérisé par la collection des vecteurs d'apparence des détections qui lui sont associées. On notera qu'il ne s'agit donc pas de faire la moyenne des vecteurs d'apparence. Pour mesurer la distance detection-*tracklet*, on mesure le minimum des distances entre la représentation de la detection et les représentations de la collection de vecteurs d'apparence.

DeepSort utilise un filtre de *Kalman* [17] pour prédire la position future d'un joueur à partir de sa position et sa vitesse précédentes. Ceci permet d'imposer des contraintes spatio-temporelles en éliminant toute association détection-*tracklet* qui s'éloignerait trop de cette prédiction. Ensuite, l'algorithme élimine toute association détection-*tracklet* ayant une trop grande distance (cette fois-ci, au sens de la mesure d'apparence décrite au paragraphe précédent) entre la détection et le *tracklet*. Les couples (*détections*, *tracklets*) ayant franchi ces deux seuils d'admissibilités sont alors associés avec l'algorithme Hongrois appliqué à la matrice de similarité détections-*tracklets*.

1. Le code de mon projet est accessible à l'adresse `https://github.com/lucas64500/PRIM/tree/master/code`

DeepSort permet de générer des *tracklet*s mais ne permet pas de fixer leur nombre. Pour obtenir un nombre de *tracklet*s égal au nombre de joueurs présents sur la vidéo, j'ai donc implémenté un algorithme de *clustering* des *tracklet*s.

4.2 *Clustering* des *tracklets*

J'ai obtenu mes meilleurs résultats en implémentant un algorithme de type *constrained spherical k-means* (*CSP-kmeans*), que j'ai obtenu en combinant les idées de [15] et [4]. J'ai dû choisir certains hyper-paramètres de *DeepSort* pour que le *clustering* avec contraintes spatio-temporelles admette toujours une solution. Il y a beaucoup d'autres solutions pour ce *clustering*, que je n'ai pas eu le temps d'expérimenter et que je cite dans la partie 6.

Un des problèmes que j'ai rencontré est celui des classes non balancées : si un joueur a beaucoup de détections, il peut avoir plusieurs *clusters* qui lui sont associés, et cela se voit dans les résultats vidéos.

4.3 *DeepSort* modifié/limité

J'ai également implémenté une autre solution permettant de conserver l'aspect *online* de *DeepSort*. Dans cette implémentation, l'algorithme ne peut générer qu'un nombre de *tracklets* égal au nombre de joueurs présents sur la vidéo. Les détections ne franchissant pas le seuil d'admissibilité à un *tracklet* existant passent alors une deuxième vague de *matching* utilisant une moyenne pondérée de la distance de *Mahalanobis* et de la distance cosinus. J'ai apporté cette dernière modification dans le but de mieux capturer les variations d'apparence au sein d'une même identité, en associant des détections aux *tracklets* sur la base d'une métrique favorisant (uniquement pour cette seconde vague) l'information spatio-temporelle par rapport à l'information d'apparence.

4.4 Initialisation des centroïdes et des *tracklets*

Pour améliorer l'initialisation des *clusters* dans 4.2, j'ai remplacé l'habituelle méthode d'initialisation *k-means++* par une méthode qui initialise les centres avec les *tracklets* présents sur la *frame* comportant le plus grand nombre cumulé de détections (en comptant toutes les détections associées

aux *tracklets* présents sur cette *frame*, même ceux qui se trouvent sur des *frames* précédentes ou suivantes).

Pour améliorer la stabilité des *tracklets* dans 4.1 et 4.3, j'ai modifié le code de *DeepSort* de sorte que les *tracklets* soient caractérisés par les vecteurs d'apparence de leurs 100 premières détections plutôt que de leurs 100 dernières. Ceci constitue en quelque sorte l'initialisation des *tracklets*.

Cela a permis de considérablement améliorer la stabilité du *tracking* sur le long terme. Toutefois, cela rend l'algorithme moins susceptible d'apprendre des variations de l'apparence des joueurs. Imaginons que sur ces 100 premières détections d'un joueur, celui-ci soit de dos, et qu'entre la 101-ème et la 110-ème, il se retourne. Au vu de la modification apportée, les détections situées entre la 101-ème et la 110-ème ne voient pas leur vecteurs d'apparence ajoutés aux collections d'apparences des *tracklets* auxquels on associe ces détections. Et donc à la 111-ème *frame*, il se peut que le vecteur d'apparence correspondant au joueur de dos soit trop différent du vecteur d'apparence du même joueur de face pour que la mesure de similarité ne franchisse le seuil d'admissibilité nécessaire à une association detection-*tracklet*. Pour remédier à cela, j'ai implémenté plusieurs solutions où les vecteurs d'apparence ne sont ajoutés aux collections des *tracklets* que selon certaines règles temporelles et j'ai également modifié la métrique de distance detection-*tracklet*.

5 Évaluation et résultats

5.1 Annotation d'une vidéo

Pour évaluer objectivement les résultats, j'ai annoté une vidéo de 5 minutes en annotant les *bounding boxes* toutes les secondes. Les détections (*hypothesis*) sont alors associées à ces *bounding boxes* (*groundtruth*) sur la base de l'algorithme Hongrois appliqué à la matrice des distance *IOU (Intersection Over Union)*.

5.2 Métriques d'évaluation

A partir de ces associations, on peut calculer plusieurs métriques introduites dans [3]. On s'intéresse notamment aux mesures *MOTA* (*Multiple Object tracking Accuracy*), *MOTP* (*Multiple Object tracking Precision*), et *IDF1* (*Identification f1-score*). On introduit également la mesure de « pureté » d'un *tracklet* d'hypothèses comme étant la proportion de sa classe dominante, le « nombre d'IDs » (N_id) comme le nombre total d'IDs différents, et le « nombre moyen d'annotations différentes par *tracklet*» (N_a) comme le nombre moyen de joueurs différents et uniques parmi les annotations ayant été associées à un même *tracklet* (on fait la moyenne sur les différents *tracklets*).

TABLE 1: Évaluation des modèles présentés sur différentes métriques

Features	**Algorithme**	**Pureté**	**N_id**	**N_a**	**Recall**	**MOTA**	**MOTP**	**IDF1**
N/A	YOLO	N/A	N/A	N/A	0.570	N/A	N/A	N/A
DAA	DS	90.992	54	1.0	0.452	0.294	0.285	0.283
	DS + CSP-kmeans	80.336	10	2.7	0.452	0.307	0.285	0.426
	DS Limited	76.812	10	2.8	0.426	0.236	0.288	0.449
DAA + BODY *LOMO*	DS							
	DS + CSP-kmeans							
	DS Limited							
DAA + BODY PARTS *LOMO*	DS							
	DS + CSP-kmeans							
	DS Limited							

5.3 Résultats vidéo

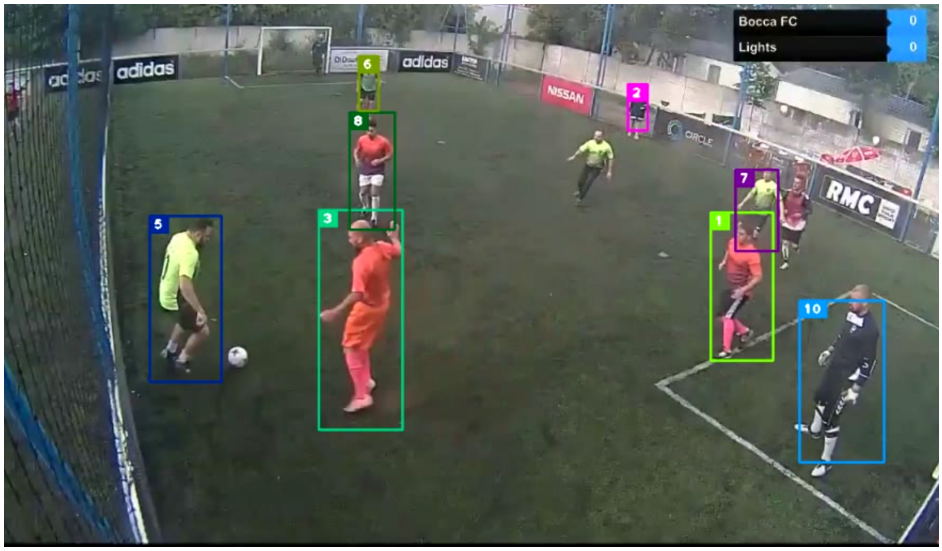


FIGURE 3: Visualisation du *tracking*

Exemple vidéo disponible sur ce lien : `https://drive.google.com/open?id=1M0REFRsYDq3lycTWSWjEpqdF06X2UAJj`.

5.4 *Fine-tuning*

Le choix de la bonne métrique à optimiser n'est pas un choix évident. On se confronte à la nécessité d'un compromis entre le *recall* et la précision.

En premier, puisque les mauvaises prédictions générées par *Deepsort* sont irréversibles et ne peuvent pas être rattrapées par le deuxième *clustering* de type *k-means*, il faut que *tracklets* soient les plus « purs » possibles, c'est à dire les plus continus possibles mais surtout, sans inversion d'ID. *Deepsort* est un *clustering* online, efficace et rapide, qui permet de faire cela si on l'autorise à générer autant de nouvelles identités que nécessaire. Le deuxième *clustering* a alors la charge de reconstituer les identités en fusionnant les *tracklets*.

Un bon *recall* permet d'exploiter au maximum les contraintes imposées au deuxième *clustering*, mais une mauvaise précision de *DeepSort* est irréversible, il s'agit donc de trouver un compromis.

Pour *fine-tuner DeepSort*, j'ai décidé d'optimiser sur les métriques de « pureté » et de « nombre moyen d'annotations différentes par *tracklet* » tout en essayant de maximiser le *recall*. Enfin, pour optimiser le *clustering* des *tracklets* ainsi que la version modifiée de *DeepSort*, je me focalise sur la métrique *IDF1*.

6 Axes d'amélioration

Voici quelques axes d'amélioration qui conserveraient la structure actuelle du projet et son approche générale. Les différentes parties du projet se succèdent et la qualité de l'une a un impact sur celle de la suivante. On peut donc considérer un axe d'amélioration pour chaque partie du système.

6.1 Qualité du détecteur

Un meilleur *recall* me semble important, et une précision moindre pour le détecteur ne me semble pas très grave dans la mesure où les *faux-positifs* ne seront probablement pas convertis en *tracklets* par le système. Un meilleur *recall* permettrait d'améliorer la qualité du *tracking* réalisé par *DeepSort* (notamment pour le filtre de *Kalman*) mais aussi pour améliorer le recouvrement de l'identité des joueurs. En effet, le *clustering* des *tracklet*s exploite beaucoup les contraintes temporelles (un même joueur ne peut pas apparaître deux fois sur la même image), et plus il y a de détections, plus il y a de contraintes à exploiter.

6.2 Qualité des *features*

Je pense que la manière la plus directe et la plus rapide d'améliorer le système actuel est d'intégrer de nouvelles *features* capturant mieux l'apparence des joueurs. Je pense qu'il faut se concentrer sur les approches non supervisées, car le *deep appearance vector* appris par *DeepSort* sur un dataset de ré-identification de piétons me semble suffisamment "*state of the art*" pour ne pas pouvoir vraiment être amélioré par des *features* d'apparence de nature similaire (construits en général par des réseaux de neurones convolutionnels). Les auteurs de [16] et [12] proposent chacun une méthode pour apprendre des vecteurs d'apparence de manière non supervisée.

6.3 Filtre de *Kalman*

DeepSort utilise un filtre de *Kalman* modélisant un mouvement à vitesse constante. Ceci n'est peut-être pas adapté à la modélisation du mouvement des joueurs qui changent souvent de direction, et parfois brutalement. Les auteurs de [6] proposent une amélioration du filtre de *Kalman* capable de traiter des mouvements non linéaires.

6.4 Clustering des *tracklets*

On pourrait améliorer cette partie avec un meilleur algorithme. Les auteurs de [13], [11] et [20] proposent des stratégies adaptées à ce problème mais que je n'ai pas eu le temps d'implémenter (et pour lesquels je n'ai pas réussi à faire fonctionner d'implémentation open source).

6.5 Semi-supervisation

Si l'on s'autorise à utiliser quelques annotations pour initialiser les *clusters* ou les *tracklets* de *DeepSort*, je suppose que les résultats pourraient être améliorés.

6.6 Autres approches

Tout comme les auteurs de [7] proposaient de traiter simultanément les problèmes de détection et de *tracking*, on pourrait avoir une approche visant à traiter simultanément les problèmes de *tracking* et d'identification. La version de *DeepSort* modifiée (voir 4.3) pour générer un nombre limité d'IDs fait partie de cette catégorie d'approches. Tout comme la stratégie proposée dans [19], dont les auteurs estiment qu'il est avantageux de traiter ces deux problèmes simultanément car chacun des problèmes génère de l'information utile et des contraintes pouvant être exploitées par l'autre, les deux modèles se "*boostrap*"-ant l'un l'autre.

7 Conclusion

Finalement, ce projet aura permis de développer un système de *tracking* et d'identification *end-to-end* de joueurs de football (ou d'autres sports) sur des vidéos. Ce système est entièrement non supervisé, et potentiellement *online*

(4.3). J'ai tenté de rassembler le code sous une forme facilement compréhensible et utilisable à la fois pour l'appliquer à de nouvelles vidéos, mais aussi pour poursuivre le développement, l'évaluation, et la sélection des modèles.

Personnellement, après plusieurs stages et projets davantage focalisés sur traitement du langage, j'ai pu me familiariser avec les techniques de vision par ordinateur que je n'avais fait que survoler en cours auparavant. J'en tire une des expériences les plus intéressantes et formatrices de mon passage à Télécom et de mon année d'application de l'X.

Références

[1] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. People-tracking-by-detection and people-detection-by-tracking. In *2008 IEEE Conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[2] Salwa Baabou, Behzad Mirmahboub, François Bremond, Mohamed Amine Farah, and Abdennaceur Kachouri. Person re-identification using pose-driven body parts. In *International conference on the Sciences of Electronics, Technologies of Information and Telecommunications*, pages 303–310. Springer, 2018.

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance : the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008 :1–10, 2008.

[4] Christian Buchta, Martin Kober, Ingo Feinerer, and Kurt Hornik. Spherical k-means clustering. *Journal of Statistical Software*, 50(10) :1–22, 2012.

[5] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose : realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv :1812.08008*, 2018.

[6] Xi Chen, Xiao Wang, and Jianhua Xuan. Tracking multiple moving objects using unscented kalman filtering techniques. *arXiv preprint arXiv :1802.01235*, 2018.

[7] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. *CoRR*, abs/1710.03958, 2017.

[8] Weina Ge and Robert T Collins. Multi-target data association by tracklets with unsupervised parameter estimation. In *BMVC*, volume 2. Citeseer, 2008.

[9] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015 : Towards a benchmark for multi-target tracking. *arXiv :1504.01942 [cs]*, April 2015. arXiv : 1504.01942.

[10] Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Z Li. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2197–2206, 2015.

[11] Yutian Lin, Xuanyi Dong, Liang Zheng, Yan Yan, and Yi Yang. A bottom-up clustering approach to unsupervised person re-identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8738–8745, 2019.

[12] Chunxiao Liu, Shaogang Gong, Chen Change Loy, and Xinggang Lin. Person re-identification : What features are important ? In *European Conference on Computer Vision*, pages 391–401. Springer, 2012.

[13] Adway Mitra, Soma Biswas, and Chiranjib Bhattacharyya. Temporally coherent crp : a bayesian non-parametric approach for clustering tracklets with applications to person discovery in videos. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, pages 801–809. SIAM, 2015.

[14] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once : Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[15] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *Icml*, volume 1, pages 577–584, 2001.

[16] Naiyan Wang and Dit-Yan Yeung. Learning a deep compact image representation for visual tracking. In *Advances in neural information processing systems*, pages 809–817, 2013.

[17] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.

[18] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[19] Baoyuan Wu, Siwei Lyu, Bao-Gang Hu, and Qiang Ji. Simultaneous clustering and tracklet linking for multi-face tracking in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 2856–2863, 2013.

[20] Shi Zhong. Efficient online spherical k-means clustering. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 5, pages 3180–3185. IEEE, 2005.