



Titre: Suivi multiobjet à l'aide de l'apprentissage automatique et de points
Title: au haut des objets

Auteur: Jacob Meilleur
Author:

Date: 2023

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Meilleur, J. (2023). Suivi multiobjet à l'aide de l'apprentissage automatique et de
Citation: points au haut des objets [Mémoire de maîtrise, Polytechnique Montréal].
PolyPublie. <https://publications.polymtl.ca/53464/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/53464/>
PolyPublie URL:

**Directeurs de
recherche:** Guillaume-Alexandre Bilodeau
Advisors:

Programme: Génie informatique
Program:

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

**Suivi multiobjet à l'aide de l'apprentissage automatique et de points au haut
des objets**

JACOB MEILLEUR

Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Mai 2023

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

**Suivi multiobjet à l'aide de l'apprentissage automatique et de points au haut
des objets**

présenté par **Jacob MEILLEUR**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Samuel KADOURY, président

Guillaume-Alexandre BILODEAU, membre et directeur de recherche

Tarek OULD-BACHIR, membre

DÉDICACE

*À tous mes amis, ma famille et mes collègues
merci pour tout...*

REMERCIEMENTS

Premièrement, j'aimerais remercier mon directeur de recherche, Guillaume-Alexandre Bilo-deau, pour m'avoir encadrer au cours de ma maîtrise. Il a toujours été en mesure de m'accorder du temps lorsque j'en avais besoin et de répondre à mes questions. De plus, il a su m'inculquer beaucoup de nouvelles connaissances. Deuxièmement, j'aimerais remercier ma famille pour m'avoir accompagné à chaque étape de mon parcours universitaire et pour m'avoir permis d'accomplir mes buts. Troisièmement, j'aimerais remercier mes collègues du LITIV pour leur aide lorsque j'avais des problèmes,, pour leur bonne compagnie et puisqu'ils m'ont aidé à approfondir mes connaissances techniques et scientifiques. Dernièrement, j'aimerais remercier le Conseil de Recherches en Sciences Naturelles et en Génie du Canada (CRSNG) pour leur généreux support financier.

RÉSUMÉ

Le suivi multiobjet est un domaine de recherche scientifique attrayant en raison des ses nombreuses applications académiques et commerciales. Cette tâche consiste à détecter les objets de classes prédéterminées à chaque trame d’une séquence vidéo, puis de faire l’association temporelle des détections afin de créer un *tracklet* pour chaque objet afin de modéliser son mouvement à travers la séquence vidéo. Le but est donc de concevoir une méthode qui parvient à créer les *tracklets* les plus fidèles à la réalité que possible. Les récentes avancées en apprentissage automatique ont permis de grandement améliorer les méthodes à l’état de l’art, mais il reste encore place à l’amélioration. En effet, un problème majeur qui doit toujours être surmonté est la gestion des occlusions. Les occlusions peuvent être catégorisées comme étant partielles ou complètes. D’abord, lorsqu’un objet est partiellement caché, il devient plus difficile à détecter à cause de la perte d’information que l’occlusion engendre. Il faut donc que notre méthode soit capable d’extraire des caractéristiques assez robustes afin d’être en mesure de détecter les objets même lorsqu’ils sont partiellement cachés. Puis, lorsqu’un objet est complètement caché, il devient impossible de le détecter avec les outils conventionnels, alors il est nécessaire d’adopter des stratégies spécifiquement conçues pour prédire son mouvement ou pour récupérer son *tracklet* une fois que l’objet redevient visible si l’on veut réussir à continuer de le suivre.

Ce mémoire présente une nouvelle méthode de suivi multiobjet qui utilise le haut des objets comme point-clé pour la détection afin de réduire le nombre de détections manquées causées par les occlusions partielles. Effectivement, les autres méthodes similaires utilisent le centre des objets comme point clé pour la détection, mais il est commun que ce point soit invisible à plusieurs reprises au cours des séquences vidéos utilisées pour évaluer le suivi multiobjet. Cependant, le haut des objets reste souvent visible lorsque le centre est caché. Ainsi, notre méthode permet de continuer à détecter certains objets qui auraient été perdus par une autre méthode utilisant le centre comme point clé. De plus, nous proposons une architecture simplifiée et plus intuitive qui ne requiert que la trame courante en entrée du réseau, contrairement à d’autres méthodes qui requièrent d’autres entrées supplémentaires. Notre méthode est d’abord composée d’un réseau neuronal convolutif servant d’extracteur de caractéristiques qui produit une carte d’attributs à partir de la trame courante. Cette carte d’attributs est ensuite utilisée par les têtes de prédiction du réseau afin d’effectuer la détection et le suivi simultanément. Certaines têtes de prédiction utilisent d’autres informations comme la carte d’attributs de la trame précédente ou les prédictions des autres têtes en addition à la carte d’attributs de la trame courante. Notre méthode parvient à obtenir des résultats compétitifs

avec les autres méthodes similaires sur deux bases de données standardisées. Généralement, notre méthode réussit à réduire le nombre de détections manquées mais commet plus d'erreurs d'association à cause de la simplicité de l'algorithme d'association temporelle utilisé.

ABSTRACT

Multi-object tracking has been a popular field of research in recent years because of its many academic and commercial uses. It consists of detecting objects of pre-defined classes at every frame of a video sequence and then associating them through time in order to create tracklets that model the trajectories of each object. The more the tracklets correspond to the actual trajectory of the objects, the better. Although, the field of multi-object tracking has existed for decades, recent advancements made in deep learning significantly improved the results achieved by tracking models. However, there are still many problems that need to be solved. One of the biggest problems that still plagues multi-object tracking is how to deal with partial and full occlusions. When an object is partially occluded, it becomes a lot harder to detect because of the loss of information incurred by the part of the object that is hidden. In these cases, it becomes extremely important that our method extracts robust and reliable features in order to be still able to detect the objects. When an object is fully occluded, it becomes impossible to detect it through conventional means, so it becomes necessary to use specially-crafted tools that can predict the movement of the object in order to follow it while it is fully occluded or that can link it back to its previous identity once it reappears.

This master’s thesis presents a novel method to perform multi-object tracking using the top of the objects as a keypoint for detection. Common multi-object tracking benchmarks include many crowded scenes in which it is often the case that an object will be partially occluded. However, when this occurs, the top of the object is generally still visible. Other similar methods that use the center of the object for detection will usually not be able to detect the occluded objects when these partial occlusions happen. Since our method uses the top of the objects for detection, it is still capable of detecting the partially occluded objects in many cases, which leads to a decrease in missed detections. Furthermore, we propose a simplified and more intuitive architecture where only the current frame is needed as input for the whole network, as opposed to many other methods that need multiple other inputs. Our method first extract a features map from the input image using a convolutional neural network. This feature map is then used by the various prediction heads of the network in order to perform detection and tracking jointly. Some prediction heads also require other inputs like the saved feature map of the previous frame or predictions from other heads. Our method manages to obtain competitive results with other similar tracker on two public benchmarks. Overall, our method manages to reduce the number of missed detections, but also commits more association errors.

TABLE DES MATIÈRES

DÉDICACE	iii
REMERCIEMENTS	iv
RÉSUMÉ	v
ABSTRACT	vii
TABLE DES MATIÈRES	viii
LISTE DES TABLEAUX	x
LISTE DES FIGURES	xi
LISTE DES SIGLES ET ABRÉVIATIONS	xiii
CHAPITRE 1 INTRODUCTION	1
1.1 Définitions et concepts de base	1
1.1.1 Suivi multiobjet	1
1.1.2 Concepts connexes	2
1.1.3 Mesures de distance et de similarité	3
1.2 Éléments de la problématique	5
1.3 Objectifs de recherche	8
1.4 Plan du mémoire	8
CHAPITRE 2 REVUE DE LITTÉRATURE	9
2.1 Détection d'objets	9
2.1.1 Détection en deux étapes	9
2.1.2 Détection en une étape	12
2.1.3 Détection sans ancrage	14
2.2 Suivi multiobjets	16
2.2.1 Suivi par détection	17
2.2.2 Détection et suivi simultané	18
2.3 Extractions de caractéristiques	21
2.3.1 Réseau neuronal convolutif	22
2.3.2 Réseau autoattentif	23

2.4 Métriques d'évaluation	24
CHAPITRE 3 DÉMARCHE DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE	29
CHAPITRE 4 ARTICLE 1 : TOPTRACK : TRACKING OBJECTS BY THEIR TOP	32 . . .
4.1 Abstract	32
4.2 Introduction	32
4.3 Related Work	34
4.4 Proposed Method	36
4.4.1 Top Heatmap	36
4.4.2 Object Size	39
4.4.3 Quantization offset	39
4.4.4 Displacement Offset	39
4.4.5 Data Association	40
4.5 Training	40
4.6 Experiments	41
4.6.1 Implementation Details	42
4.6.2 Results	42
4.6.3 Ablation Study	44
4.7 Conclusion	48
CHAPITRE 5 DISCUSSION GÉNÉRALE	49
CHAPITRE 6 CONCLUSION	54
6.1 Synthèse des travaux	54
6.2 Limitations de la solution proposée	55
6.3 Améliorations futures	55
RÉFÉRENCES	57

LISTE DES TABLEAUX

Tableau 4.1	Results of state-of-the-art methods on MOT17 dataset with private detections. Best results are in bold	43
Tableau 4.2	Results of state-of-the-art methods on MOT20 dataset with private detections. Best results are in bold	43
Tableau 4.3	Results of TopTrack on the MOT17 validation set using keypoint for detection. Best results are in bold	47
Tableau 4.4	Results of TopTrack on the MOT17 validation set using different matching algorithms. Best results are in bold	47
Tableau 4.5	Results of TopTrack on the MOT17 validation set using different seeds for training. Best results are in bold	47
Tableau 5.1	Comparaison de TopTrack et CenterTrack [1] par séquence de l'ensemble de test de MOT17 [2]. Les meilleurs résultats sont en gras . .	49
Tableau 5.2	Résultats de TopTrack sur l'ensemble de validation de MOT17 selon les bases de données utilisées pour l'entraînement. "CH" signifie l'utilisation de la base de données <i>CrowdHuman</i> . Les meilleurs résultats sont en gras	50
Tableau 5.3	Résultats de TopTrack sur l'ensemble de validation des piétons de KITTI Tracking [3]	51
Tableau 5.4	Résultats de TopTrack sur l'ensemble de validation des voitures de KITTI Tracking [3]	51

LISTE DES FIGURES

Figure 1.1	Exemple de matrice de confusion.	4
Figure 1.2	Image du jeu de données MOT17 [2] montrant des occlusions partielles et la taille variable des objets.	7
Figure 2.1	Exemples du fonctionnement de Selective Search avec des boîtes de différentes échelles [4] © 2014 IEEE.	10
Figure 2.2	Fonctionnement de RPN [5] © 2015 IEEE.	11
Figure 2.3	Pipeline de R-CNN [6] © 2014 IEEE.	11
Figure 2.4	Pipeline de Faster R-CNN [5] © 2017 IEEE.	12
Figure 2.5	Fonctionnement de YOLO [7] © 2016 IEEE.	13
Figure 2.6	Fonctionnement de SSD [8] © 2016 Springer International Publishing AG.	14
Figure 2.7	Pipeline de CornerNet [9] © 2019 Reproduit avec permission de Springer Nature.	15
Figure 2.8	Architecture de FairMOT [10] © 2021 Reproduit avec permission de Springer Nature	20
Figure 2.9	Fonctionnement de TrackFormer [11] © 2022 IEEE.	21
Figure 2.10	Pipeline de TransCenter [12] © 2022 IEEE.	21
Figure 2.11	Exemple de convolution. <i>Droits d'auteur</i> © Vincent Dumoulin, Francesco Visin [13].	22
Figure 2.12	Architecture générale des réseaux autoattentif [14].	24
Figure 2.13	Exemple de TP, FP et FN pour le calcul du score d'association de HOTA [15]	27
Figure 4.1	Overview of the architecture of TopTrack : An image at timestep t is fed to a Stacked Hourglass backbone to extract the feature map. This feature map is used to predict the size of the objects and the quantization offset. The feature map from the previous timestep $t - 1$ is saved and then used in conjunction with the current feature map to generate the top heatmaps. Finally, the current feature map, the current top heatmaps, the previous feature map and the previous top heatmaps are used to predict the tracking offset	37

Figure 4.2	Visualization of how top heatmaps are used to extract object position. The likelihood of a certain pixel being the position of the top of an object is represented by a Gaussian function. The positions are extracted by identifying the peaks of each Gaussian function	38
Figure 4.3	Comparison between TopTrack and CenterTrack on two frames of sequence 8 of MOT17 test set. Arrows show objects detected by TopTrack but missed by CenterTrack	44
Figure 4.4	Comparison between TopTrack and CenterTrack on one frame of sequence 7 and one frame of sequence 12 of MOT17 test set. Arrows show objects detected by TopTrack but missed by CenterTrack	45
Figure 4.5	Examples of shop mannequins being detected as objects by TopTrack. See ID 173 and 175 (left) and ID 27 and 41 (right)	45
Figure 4.6	Examples of reflections being detected as objects by TopTrack. See ID 56, 63 and 64 (left) and ID 16 and 22 (right)	46
Figure 4.7	Examples of a cartoon character being detected as an object by TopTrack (See ID 17)	46
Figure 5.1	Point-clés des piétons de la scène 02 de l'ensemble d'entraînement de MOT17 [2]	52
Figure 5.2	Point-clés des piétons de la scène 04 de l'ensemble d'entraînement de MOT17 [2]	53
Figure 5.3	Point-clés des piétons et des voitures de la scène 13 de l'ensemble d'entraînement de MOT17 [2]	53

LISTE DES SIGLES ET ABRÉVIATIONS

AssA	Association Accuracy
DetA	Detection Accuracy
FN	False Negative
FP	False Positive
GT	Ground Truth
HOTA	Higher Order Tracking Accuracy
IDF1	Identification F1
IDSW	Identity Switch
IoU	Intersection over Union
LocA	Localisation Accuracy
ML	Mostly Lost
MOTA	Multi-Object Tracking Accuracy
MOTP	Multi-Object Tracki Precision
MT	Mostly Tracked
NMS	Non-Maximum Suppression
PT	Partly Tracked
QLN	Query Learning Network
RPN	Region Proposal Network
SORT	Simple Online and Realtime Tracking
SSD	Single Shot MultiBox Detector
TN	True Negative
TP	True Positive
YOLO	You Only Look Once

CHAPITRE 1 INTRODUCTION

Ce projet de recherche se concentre sur le suivi multiobjet en ligne en utilisant l'apprentissage automatique. Être en mesure de suivre plusieurs objets simultanément dans une séquence vidéo et d'ainsi créer leur trajectoire à travers le temps est crucial pour plusieurs applications. Réussir à automatiser ce processus en utilisant les technologies de l'état de l'art comme l'apprentissage automatique apporte donc une très grande valeur au niveau académique et commerciale. Par exemple, dans le cas de la conduite automobile autonome, le suivi multiobjet est impératif à leur bon fonctionnement et serait en mesure de réduire le nombre d'accidents sur la route dans le futur. Bien que le suivi multiobjet est surtout connu à cause de la conduite autonome, il existe plusieurs autres applications qui en font usage. D'abord, les méthodes de suivi sont extrêmement utiles pour les systèmes intelligents d'analyse de contenu vidéo. Par exemple, ces méthodes peuvent assister des opérateurs humains lors de l'analyse de vidéos de surveillance. Elles peuvent aussi aider les biologistes à détecter et suivre différents animaux comme les oiseaux, les bancs de poissons, les abeilles ou même les cellules humaines. Une autre application intéressante du suivi multiobjet est la détection et l'identification des athlètes lors de la diffusion d'événements sportifs. Le suivi multiobjet est donc un domaine qui peut apporter beaucoup de valeur à plusieurs applications académiques et commerciales.

1.1 Définitions et concepts de base

1.1.1 Suivi multiobjet

L'objectif principal de cette tâche est de concevoir une méthode pour identifier les objets de classes prédéfinies présents dans une vidéo à l'aide d'un identifiant unique, puis d'attribuer le même identifiant aux mêmes objets tout au long de la vidéo. Cette tâche permet donc de déterminer la trajectoire parcourue par chaque objet au cours de la vidéo. Le présent projet de recherche se concentre sur le suivi *en ligne*, ce qui signifie qu'il n'est pas possible d'utiliser des informations futures ou la vidéo complète dès le départ. Les images sont plutôt fournies séquentiellement et en ordre chronologique. Il existe aussi des modèles *hors-lignes* qui utilisent toute l'information disponible dès le départ afin d'améliorer leurs performances. Ce type de modèle a son utilité, cependant il est impossible d'avoir accès à toute l'information d'avance pour plusieurs applications comme la conduite automobile autonome, d'où l'importance de faire la distinction entre les modèles en ligne et hors-ligne.

Étant donné qu'une vidéo n'est en vérité qu'une séquence d'images, le suivi en ligne se fait

une image à la fois. D’abord, un premier module va extraire les caractéristiques de l’image. Ces caractéristiques sont apprises par le modèle et servent à reconnaître les objets présents dans l’image. À l’aide de ces caractéristiques, un deuxième module sera en mesure de détecter les objets dans l’image. Enfin, un dernier module fera la correspondance entre les objets d’une image avec ceux de l’image précédente. Ce processus est répété pour chaque image formant la vidéo, après quoi le suivi est terminé. Si un même objet se fait attribuer plusieurs identifiants différents ou qu’il n’est pas détecté au cours de la vidéo, le suivi de cet objet a échoué. De plus, la détection d’un objet à un endroit où il n’y en a pas est aussi considéré comme une erreur. Des métriques standardisées se basant sur ces erreurs ont été développées pour mesurer la performance des méthodes de suivi multiobjet, ce qui facilite la comparaison des différentes méthodes. De plus, un autre moyen utilisé afin de mieux comparer les différentes méthodes qui sont développées dans ce domaine et d’assurer l’équité lors de leur évaluation est de créer des bases de données publiques pour l’entraînement et l’évaluation des méthodes. Les vérités de terrain des données d’évaluation ne sont pas disponibles publiquement afin d’éviter la tricherie. Les résultats des différentes méthodes sont affichés publiquement pour permettre de se comparer aux autres travaux.

1.1.2 Concepts connexes

Un concept important au suivi multiobjet est ce qui est généralement appelé un *tracklet*. Ce terme réfère à l’historique partiel de la position d’un objet, c’est-à-dire à une partie de sa trajectoire. L’objectif du suivi est donc de créer une trajectoire fidèle à la réalité pour chaque objet qui se trouve dans une vidéo fournie, mais il faut se rappeler qu’en général on obtiendra des *tracklets* puisque la méthode de suivi n’est pas capable de reproduire une trajectoire continue fidèle à la réalité.

La tâche de détection d’objet est intrinsèque au suivi multiobjet puisqu’il faut être en mesure de détecter les objets si l’on veut pouvoir les suivre. Une technique de détection d’objets communément utilisée par les méthodes de suivi multiobjet est de prédire une grande quantité de détections, puis de filtrer les mauvaises détections afin de ne garder que celles qui sont bonnes. *Non-Maximum Suppression* (NMS) [16] est un algorithme qui a été développé pour accomplir cette tâche. Il utilise l’indice de Jaccard, qui sera expliqué sous peu, afin de déterminer si plusieurs boîtes englobantes sont superposées, la supposition étant que si plusieurs boîtes englobantes ont un indice de Jaccard plus grand qu’un certain seuil pré-déterminé, elles correspondent au même objet. Dans un tel cas, NMS garde que la boîte englobante avec le plus grand score de confiance. Le détecteur d’objets apprend à attribuer correctement ce score de confiance lorsqu’il fait ses prédictions durant son entraînement. Le résultat final est

un ensemble de détections duquel les détections doubles ont été éliminées. Ensuite, il est souvent le cas que les méthodes vont aussi éliminer les détections restantes qui ont un score de confiance bas, même si elles n'ont aucun chevauchement avec une autre détection, puisque ce sont généralement des fausses détections.

Lors de l'étape d'association des données, une approche commune est de calculer un score à partir d'une mesure de distance ou de similarité entre les détections de la trame courante et les *tracklets* existants, puis d'utiliser un algorithme pour faire la correspondance entre les deux ensembles. Les méthodes qui seront présentées dans le chapitre 2 utilisent l'algorithme glouton ou l'algorithme hongrois [17]. Un algorithme est considéré glouton s'il prend le choix optimal lors de chacune de ses itérations sans se soucier du résultat global. Dans le contexte de l'association des données pour le suivi multiobjet, un algorithme glouton fait la correspondance une détection à la fois en associant la détection de la trame courante avec le *tracklet* ayant la plus petite distance ou la plus grande similarité dépendamment de la mesure choisie. En revanche, l'algorithme hongrois utilise un heuristique plus complexe que l'algorithme glouton afin de minimiser un coût global. En effet, il est parfois mieux de choisir un candidat non-optimal lors d'une itération afin d'avoir un meilleur candidat pour une itération future et d'obtenir un meilleur résultat global ainsi. D'abord, une matrice de coût carrée ($n \times n$) entre tous les éléments qui doivent être associés est créée. Par exemple, si le coût est la mesure de distance spatiale entre les détections de la trame courante et les détections de la trame précédente, alors l'algorithme hongrois va associer les détections afin de minimiser la distance totale.

Une façon habituelle de visualiser la performance d'un modèle est la matrice de confusion. Cette matrice met en relation dans une table, les valeurs véridiques et les valeurs prédites par le modèle afin de calculer un taux de vrai positif (TP), vrai négatif (TN), faux positif (FP) et faux négatif (FN). La Figure 1.1 présente une telle matrice.

1.1.3 Mesures de distance et de similarité

Il existe plusieurs mesures de distance qui sont fréquemment utilisées par les modèles de suivi multiobjets lors de l'étape d'association des objets. Il est donc important d'en prendre connaissance afin de bien comprendre le fonctionnement de la méthode proposée et de celles qui seront présentées lors de la revue de littérature. D'abord, la distance euclidienne est une mesure très populaire puisqu'elle représente la droite entre deux points, soit le chemin le plus court entre deux points. Soit deux vecteurs $p, q \in \mathbb{R}^n$, la distance euclidienne entre p et q est

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}. \quad (1.1)$$

		Prédiction	
		Positif	Négatif
Vérité de terrain	Positif	Vrai Positif	Faux Négatif
	Négatif	Faux Positif	Vrai Négatif

FIGURE 1.1 Exemple de matrice de confusion.

L'indice de Jaccard est une autre métrique importante dans le domaine du suivi multiobjet. Elle sert à quantifier le chevauchement entre deux formes. Par exemple, cette mesure peut être utilisée pour quantifier le chevauchement des boîtes englobantes rectangulaires A et B qui définissent la position des objets dans une image. Elle est calculée en faisant le rapport entre l'intersection et l'union des aires étudiées

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \quad (1.2)$$

L'indice de Jaccard peut aussi être calculé à partir des éléments de la matrice de confusion. Cette formulation est utilisée par certaines métriques de détection d'objets. La formule devient donc

$$J(A, B) = \frac{TP}{TP + FP + FN}. \quad (1.3)$$

Il est aussi possible de transformer l'index de Jaccard en une mesure de distance, la distance de Jaccard, en prenant son complément

$$d_J(A, B) = 1 - J(A, B). \quad (1.4)$$

Une troisième mesure fréquemment utilisée est la similarité cosinus. Cette mesure sert à quantifier la similitude entre deux vecteurs de nombres. Dans un espace vectoriel, la similarité cosinus représente le cosinus de l'angle entre deux vecteurs et produit une valeur dans l'intervalle $[-1, 1]$. Une similarité de 1 signifie que les vecteurs sont colinéaires, une similarité de 0 signifie que les vecteurs sont orthogonaux et une similarité de -1 signifie aussi que les

vecteurs sont colinéaires, mais opposés. La similarité cosinus se mesure par le rapport entre le produit scalaire et le produit des normes des vecteurs $\vec{x}$ et $\vec{y}$

$$S(\vec{x}, \vec{y}) = \cos \theta = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \|\vec{y}\|}. \quad (1.5)$$

Enfin, la distance de Mahalanobis est parfois utilisée et est similaire à la distance euclidienne, cependant elle se démarque par le fait qu'elle prend en compte la covariance des éléments afin d'attribuer moins de poids aux données qui ont une grande variance. Elle est donnée par

$$d_M(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x} - \vec{y})}. \quad (1.6)$$

On peut remarquer par la formule que si la matrice de covariance Σ est la matrice identité, la distance de Mahalanobis devient équivalente à la distance euclidienne.

1.2 Éléments de la problématique

Il existe plusieurs contraintes et problèmes spécifiques au domaine du suivi multiobjet qu'il faut surmonter afin d'accomplir la tâche avec succès. Une première contrainte est la rapidité de la méthode. En effet, plusieurs applications du suivi multiobjet nécessitent que la méthode ait une certaine rapidité. Par exemple, c'est le cas de la conduite autonome, où la méthode doit être assez rapide afin d'assurer un temps de réaction suffisant pour éviter un accident. Puis, une deuxième contrainte est la quantité de ressources computationnelles nécessaire pour le bon fonctionnement de la méthode. Effectivement, la méthode doit utiliser le moins de ressources possibles puisqu'une méthode consommant une grande quantité de ressources aura besoin d'ordinateurs puissants, ce qui augmentent le prix et la grosseur des systèmes qui vont utiliser cette méthode. De plus, généralement une méthode gloutonne sera plus lente, il y a donc un lien intrinsèque entre les deux premières contraintes. Dans ce travail, bien que la rapidité ne soit pas une contrainte qui doit être respectée puisqu'il n'y a aucune limite de temps imposée sur la génération des résultats sur les séquences vidéos des bases de données utilisées, avoir une méthode rapide est généralement désirable. Quant à la deuxième contrainte, les ressources requises pour notre méthode sont limitées par le matériel mis à notre disposition.

Un élément problématique commun aux méthodes faisant usage de l'apprentissage automatique est la généralisation. En effet, le modèle reçoit un nombre limité de données lors de son entraînement, mais il faudrait idéalement qu'il fonctionne bien dans tous les scénarios, même ceux qu'il n'a jamais vu auparavant. Il faut donc entraîner le modèle assez longtemps

et avec des données assez diverses pour qu'il soit capable d'effectuer la tâche pour laquelle il a été conçu, mais pas trop longtemps pour éviter le surapprentissage.

Les éléments problématiques spécifiques au suivi multiobjet peuvent être divisés selon les deux étapes qui composent la tâche de suivi, soit la détection et l'association. Les problèmes majeurs liés à la détection des objets incluent les occlusions partielles et complètes, le nombre variable d'objets et la variation de la taille des objets. Une occlusion partielle est définie comme un objet qui est seulement partiellement visible, alors qu'une occlusion est définie comme complète lorsque l'objet est complètement invisible. Généralement, comme le montre la Figure 1.2, une occlusion partielle ou complète peut survenir lorsqu'un objet en obstrue un autre. De plus, une occlusion partielle peut aussi survenir lorsqu'un objet entre ou sort de la scène. Il devient alors plus difficile de détecter un objet partiellement occlus et presque impossible de détecter un objet complètement occlus puisque la méthode de détection a moins ou aucune information sur cet objet. Dans le cas des occlusions complètes, il faudra généralement adopter d'autres stratégies comme de prédire le mouvement de l'objet ou de récupérer le *tracklet* une fois que l'objet redevient visible. Ensuite, le nombre variable d'objets présents dans la scène pose aussi un défi. En effet, comme le nombre d'objets à détecter n'est pas connu a priori et qu'il est variable d'une image à l'autre, il est impossible de déterminer le nombre de sorties que doit avoir le détecteur. Enfin, comme le démontre la Figure 1.2, une même trame peut contenir des objets avec des tailles radicalement différentes, alors il peut être difficile de bien capturer chaque objet malgré leur taille. En effet, les objets de petites tailles sont généralement plus difficiles à détecter puisque la méthode de détection a moins d'informations sur ces objets. De plus, la tâche de détection nécessite de localiser correctement les objets avec des boîtes englobantes alors le détecteur doit être en mesure de bien capturer la position des objets de toutes les tailles.

L'association des données utilise les informations sur le mouvement et/ou sur l'apparence des objets afin de créer les *tracklets* correctement. Les problèmes majeurs liés à ce processus incluent les occlusions complètes, les changements de mouvement, la dérive du modèle de représentation et les objets ayant des apparences similaires. Les occlusions complètes peuvent aussi être un problème lors de l'association des données parce que lorsqu'un objet redevient visible après une occlusion, il faut être en mesure de l'associer correctement au même *tracklet* qu'avant que l'occlusion se produise. Cela peut être particulièrement difficile si l'objet a bougé considérablement pendant qu'il était caché. Une deuxième difficulté est les changements de mouvements des objets. En effet, il se peut que des objets ne suivent pas un mouvement uniforme pour la totalité de la séquence vidéo utilisée, ce qui rend la prédiction du mouvement de ces objets plus difficile. Des changements de mouvement brusques peuvent aussi être le résultat d'une caméra mobile, comme dans le cas d'une caméra montée sur une voiture.



FIGURE 1.2 Image du jeu de données MOT17 [2] montrant des occlusions partielles et la taille variable des objets.

Un problème commun lorsque l'apparence d'un objet est utilisé pour le reconnaître est un phénomène appelé la dérive du modèle de représentation. Ce phénomène survient lorsque les caractéristiques utilisées pour décrire l'apparence de l'objet deviennent erronées, donc le modèle ne sera plus capable d'associer cet objet correctement en se basant sur son apparence. Lors du suivi multiobjet, il est nécessaire de mettre à jour régulièrement les caractéristiques utilisées pour décrire l'apparence des objets afin de prévenir une dérive du modèle puisque leur apparence peut changer au cours de la séquence vidéo. Effectivement, plusieurs scénarios fréquents vont causer un changement d'apparence tel que lors du passage d'un endroit éclairé vers un endroit sombre, lors d'une rotation et lors d'une occlusion partielle. Cependant, effectuer cette mise à jour peut aussi engendrer une dérive du modèle si elle est faite à un moment inopportun. Trouver une manière adéquate de mettre à jour les caractéristiques d'apparence des objets est donc un élément problématique important à surmonter. Finalement, il se peut que plusieurs objets aient une apparence similaire, alors utiliser les caractéristiques d'apparence peut causer des erreurs dans ce cas. Comme il sera expliqué plus tard, notre méthode se base uniquement sur la position et le mouvement des objets afin de les associer d'une trame à l'autre, de sorte que seuls les éléments problématiques qui affectent ces informations nous concernent.

1.3 Objectifs de recherche

L’objectif principal de ce projet de recherche est de développer une nouvelle méthode de suivi multiobjet en utilisant l’apprentissage automatique. L’approche favorisée pour la réalisation de ce projet est de se baser sur une méthode existante et d’y apporter des modifications afin d’améliorer ses performances. Les sous-objectifs sont :

- Trouver une stratégie afin d’obtenir des meilleurs performances lorsqu’il y a des occlusions et lorsque le mouvement des objets changent brusquement ;
- Conserver une architecture de réseau simple qui requiert moins d’information en entrée ;
- Évaluer et comparer les performances de notre méthode avec l’état de l’art sur des bases de données publiques.

1.4 Plan du mémoire

Le contenu du mémoire est divisé de la façon suivante. D’abord, le chapitre 2 présente la revue de littérature. Cette section a pour but de présenter les travaux connexes à celui-ci dans les domaines de la détection d’objets et du suivi multiobjet afin de mettre en contexte ce projet de recherche. De plus, le chapitre 2 présente aussi toute autre information jugée pertinente pour la bonne compréhension de ce travail, comme les métriques communément utilisées afin de juger les performances des méthodes de suivi multiobjet. Ensuite, le chapitre 3 détaille l’évolution du projet au cours de la maîtrise. Le chapitre 4 présente l’article que nous avons soumis. Le chapitre 5 sert à compléter la discussion des résultats présentée dans l’article afin de donner plus d’informations sur certains aspects. Enfin, le chapitre 6 conclut le travail et inclut une synthèse des travaux, les limitations de la solution proposée et des améliorations futures possibles.

CHAPITRE 2 REVUE DE LITTÉRATURE

Dans ce chapitre, nous allons expliquer les travaux en vision par ordinateur qui précèdent le présent projet de recherche et qui portent sur le domaine du suivi multiobjet afin d'avoir une bonne compréhension de l'historique et du contexte courant de ce domaine. D'abord, le domaine du suivi multiobjet se basent intrinsèquement sur les travaux provenant du domaine de la détection d'objets puisqu'il faut être en mesure de détecter les objets présents dans une image si l'on veut pouvoir les suivre. La revue de littérature explorera donc premièrement ce sujet afin d'établir une base de connaissances solides sur laquelle les autres concepts de suivi multiobjet pourront être bâtis. Ensuite, les différents concepts spécifiques au suivi multiobjet seront présentés. Enfin, une brève explication des principales façons d'extraire des caractéristiques à partir d'une image sera présentée. Seules les méthodes utilisant l'apprentissage automatique seront abordées puisque les méthodes qui les précèdent sont maintenant obsolètes et ne sont pas utiles pour bien comprendre ce qui sera présenté.

2.1 Détection d'objets

La détection d'objet est une tâche fondamentale de la vision par ordinateur de laquelle découle plusieurs autres tâches comme le suivi multiobjet, la segmentation d'instances, la détection de maladies et la conduite autonome. Étant donné les avancées technologiques importantes qui ont été apportées aux méthodes de détection d'objets à travers les années, c'est un domaine qui a une longue histoire et qui a fait l'objet de beaucoup de recherche. D'abord, les premières méthodes de détection d'objets ont été proposées avant l'ère des réseaux de neurones. Elles sont donc plutôt rudimentaires et se basent généralement sur l'extraction des caractéristiques d'un objet afin de pouvoir le retrouver dans une nouvelle image. Ensuite, avec l'introduction des réseaux de neurones profonds et de l'apprentissage automatique, plusieurs nouvelles méthodes faisant usage de ces puissantes technologies ont vu le jour. Ces nouvelles méthodes ont donné naissance à des nouvelles familles de détecteurs d'objets soit, les détecteurs en deux étapes, les détecteurs en une étape et les détecteurs sans ancrage.

2.1.1 Détection en deux étapes

Les détecteurs en deux étapes sont le premier type de détecteurs d'objets à avoir été proposés suite à l'introduction de l'apprentissage profond. L'idée générale est d'utiliser un premier réseau dont l'objectif est de proposer des régions d'intérêt dans l'image. Ces régions d'intérêt

correspondent à des propositions de boîtes englobantes dans l'image où la probabilité qu'un objet s'y retrouve est élevée. Cette première étape sert à restreindre la recherche d'objets à quelques régions de l'image afin de réduire le coût de calculs de la méthode. Ensuite, lors de la deuxième étape, un autre réseau analyse les régions proposées afin de déterminer si un objet s'y retrouve réellement et de le classifier correctement, le cas échéant.

Proposition de régions : Plusieurs méthodes existent afin de générer des propositions de régions. Les premiers modèles de la famille de modèles *R-CNN* [6, 18] utilisent l'algorithme *Selective Search* [4] qui fait usage de la segmentation sémantique afin de regrouper ensemble les pixels de l'image qui appartiennent à un même objet. Ce processus se déroule sur plusieurs itérations afin de raffiner la segmentation selon quatre critères laissés au choix de l'utilisateur, soit la couleur, la texture, la taille des groupes et le chevauchement entre deux groupes. Les boîtes englobantes sont ensuite générées à partir de chaque regroupement de pixels. La Figure 2.1 montre comment *Selective Search* réussit à détecter des objets de différentes tailles en variant la taille des groupes.



FIGURE 2.1 Exemples du fonctionnement de Selective Search avec des boîtes de différentes échelles [4] © 2014 IEEE.

Cependant, *Selective Search* a l'inconvénient d'utiliser beaucoup de ressources et d'être très lent en conséquence. Ainsi, *Faster R-CNN* [5] a introduit une nouvelle méthode de proposition de régions, appelée réseau de proposition de régions (RPN), afin d'accélérer la génération des régions d'intérêts et de leur boîtes englobantes respectives. D'abord, une carte d'attributs de l'image est extraite à l'aide d'un réseau neuronal convolutif. Ensuite, comme le montre la Figure 2.2, une fenêtre coulissante d'une grandeur de $(n \times n)$ passe séquentiellement à chaque position d'une grille sur la carte d'attributs afin de générer des boîtes d'ancrage en utilisant des dimensions pré-définies. Le nombre de boîtes d'ancrage et leurs dimensions sont des hyperparamètres qui sont parfois définis suite à une analyse statistique des données. Enfin, les boîtes d'ancrage sont envoyées à un classificateur et à un régresseur qui vont évaluer la

probabilité qu'un objet soit présent dans la boîte et comment modifier la position et la taille de la boîte pour mieux entourer l'objet respectivement. En utilisant l'information fournie par le classificateur et le régresseur, il est possible de générer les boîtes englobantes finales.

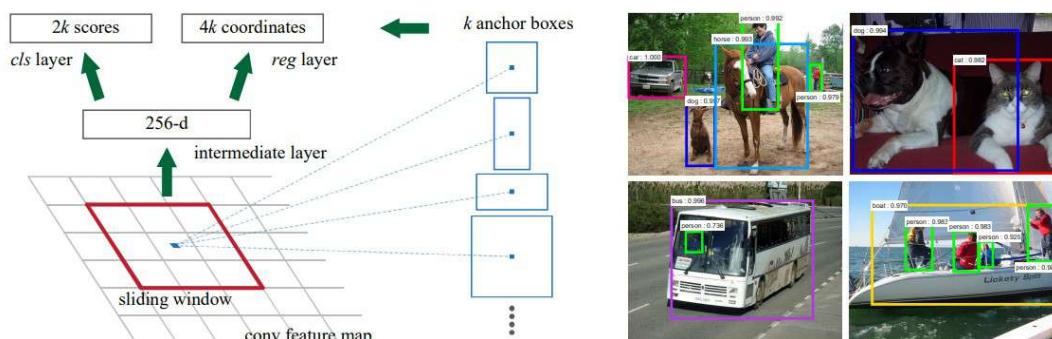


FIGURE 2.2 Fonctionnement de RPN [5] © 2015 IEEE.

R-CNN : Alors que les méthodes à l'état de l'art utilisaient principalement des point-clés *SIFT* [19] ou des histogramme de gradients orientés [20] afin de représenter les objets, *R-CNN* [6] propose plutôt d'utiliser un réseau neuronal convolutif pour extraire un vecteur de taille fixe qui servira de représentation pour les objets. La Figure 2.3 montre chaque étape de cette méthode. D'abord, l'algorithme *Selective Search* est utilisé afin d'extraire des régions d'intérêts et des boîtes englobantes candidates. Puis, chaque boîte est envoyée en entrée au réseau neuronal convolutif afin de générer les vecteurs représentant ce qui se trouve dans cette boîte. Durant le passage dans le réseau neuronal convolutif, un décalage sera aussi prédit afin que la boîte englobante finale entoure mieux l'objet détecté. En dernier, pour chaque classe, la probabilité que l'objet de cette classe soit présente est calculée pour chaque boîte en envoyant le vecteur à un SVM [21] entraîné pour reconnaître cette classe d'objet.

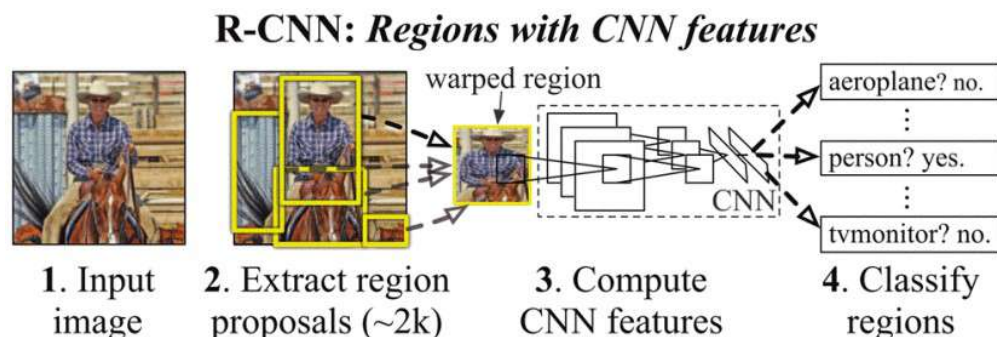


FIGURE 2.3 Pipeline de R-CNN [6] © 2014 IEEE.

Faster R-CNN : *R-CNN* a révolutionné le domaine de la détection d'objets, mais n'était

toujours pas assez rapide pour fonctionner en temps réel. En effet, un total de 2000 régions d'intérêts sont extraites pour chaque image et sont ensuite envoyées séquentiellement au réseau convolutif, ce qui est assez lent. Afin de grandement accélérer leur méthode, les auteurs de *Faster R-CNN* proposent, entre autres, de remplacer l'algorithme *Selective Search* par un Réseau de Proposition de Régions (RPN). Comme montré par la Figure 2.4, les régions extraites sont ensuite envoyées à une couche d'agrégation des régions d'intérêts (RoI pooling). Cette couche sert à normaliser la taille des régions d'intérêts qui ont été proposées à l'étape précédente. En effet, chaque région d'intérêt extraite a une grandeur différente des autres et toutes les régions extraites seront envoyées à des couches pleinement connectées afin de générer les prédictions finales du réseau. Ces couches pleinement connectées requièrent que les données qui leur aient été envoyées en entrée soient de la même grandeur, c'est pourquoi il est nécessaire de faire l'agrégation des régions d'intérêt. Finalement, la sortie des couches pleinement connectées représente la prédiction du modèle sur la classe de l'objet et la boîte englobante qui y correspond.

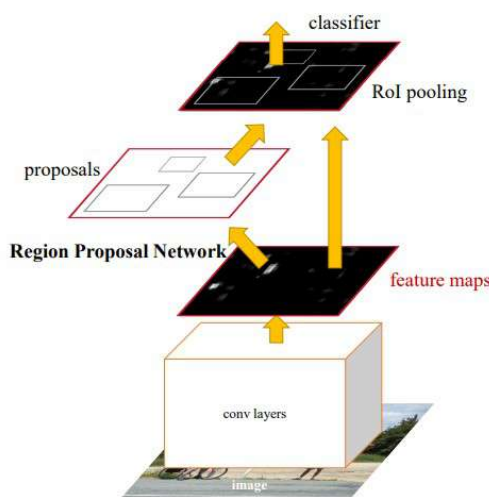


FIGURE 2.4 Pipeline de Faster R-CNN [5] © 2017 IEEE.

2.1.2 Détection en une étape

Les détecteurs en une étape représentent une évolution alternative des détecteurs d'objets et visent à significativement améliorer leur rapidité d'exécution afin d'avoir des méthodes capables de fonctionner en temps réel, au coût d'une légère baisse de précision. Alors que les détecteurs en deux étapes définissaient la détection d'objets comme un problème de classification, les détecteurs en une étape voient plutôt la détection d'objets comme un problème de régression où le modèle prédit les coordonnées de la boîte englobante et les probabilités de

chaque classe pour chaque objet directement à partir des pixels. En conséquence, l'étape de proposition de régions d'intérêt est complètement retirée et la détection d'objet se fait qu'en une seule étape de bout-en-bout et n'utilise qu'un seul modèle pour y arriver. Cela a comme effet d'éliminer un autre défaut des détecteurs en deux étapes, soit que chaque module du réseau doit être entraîné séparément.

You Only Look Once : *You Only Look Once* (YOLO) [7] est le premier détecteur d'objets en une étape à avoir été proposé. Il se démarque par sa grande rapidité, allant jusqu'à 150 images par seconde lors de son étape d'inférence. Il a aussi l'avantage d'être une méthode simple et généralisable. Comme le montre la Figure 2.5, les images sont redimensionnées à une taille de 448×448 pixels. Ensuite, elles sont envoyées comme entrée à un réseau neuronal convolutif qui prédit simultanément la position des boîtes englobantes, la probabilité de classe des objets et un score de confiance. Enfin, seules les détections avec score de confiance plus grand qu'un seuil choisi manuellement sont gardées. Au moment de sa publication, *YOLO* était capable d'atteindre une précision plus de deux fois plus grande que les autres méthodes faisant de la détection en temps réel. Cette première itération de *YOLO* est aussi la première méthode à ne pas faire usage d'ancrages pour aider à la prédictions de boîtes englobantes. Cependant, la prochaine itération de la méthode, *YOLO9000* [22], et les itérations subséquentes jusqu'à *YOLOX* [23] incorpore l'utilisation de boîtes d'ancrages afin d'atteindre de meilleurs performances.

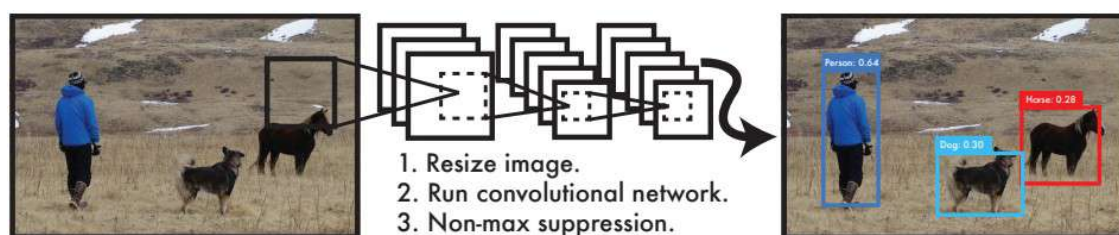


FIGURE 2.5 Fonctionnement de YOLO [7] © 2016 IEEE.

Single Shot MultiBox Detector : *Single Shot MultiBox Detector* (SSD) [8] est un détecteur d'objets en temps réel qui a été publié très peu de temps après *YOLO*. Il réussit à surpasser les résultats de *YOLO* en utilisant plusieurs carte d'attributs de différentes grandeurs afin de mieux détecter les objets de différentes tailles et en utilisant des ancres. La Figure 2.6 montre comment les ancres sont utilisées pour la détection, puis raffinées pour mieux décrire la localisation des objets. *SSD* se base sur l'architecture VGG-16 introduite par Simonyan et al [24]. Cette architecture consiste d'une série de couches de convolution afin de réduire progressivement la taille de la carte d'attributs. *SSD* ajoute ensuite de nouvelles couches de convolutions qui produisent chacune des détections et réduisent davantage la carte

d'attributs. Étant donné que ce modèle prédit une très grande quantité de détections, une dernière étape de NMS est nécessaire afin d'avoir les détections finales.

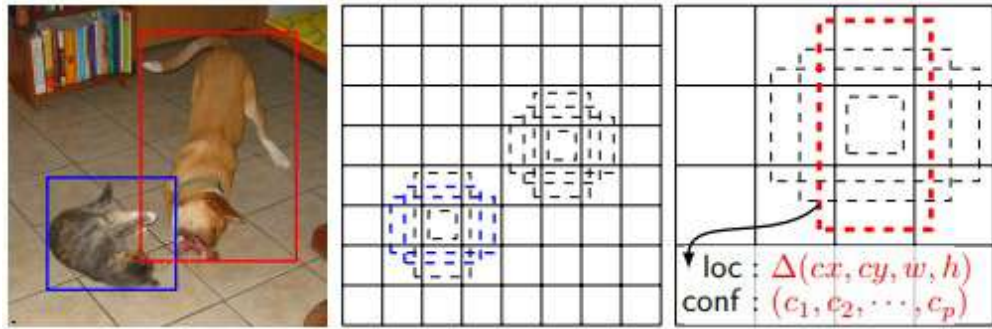


FIGURE 2.6 Fonctionnement de SSD [8] © 2016 Springer International Publishing AG.

2.1.3 Détection sans ancrage

Les méthodes de détection d'objets détaillées précédemment se basent principalement sur des ancres afin de détecter les objets correctement. Cependant, cette méthode de détection à plusieurs désavantages. En effet, afin de déterminer la taille des boîtes d'ancrage, une étude statistique est généralement faite sur le jeu de données. Cela affecte la capacité de généralisation du modèle puisque l'étude sera biaisée selon le jeu de données. De plus, dans le cas de l'utilisation d'un tel modèle pour une application réelle, il se peut que les données utilisées pour mener l'étude statistique ne soient pas représentatives des données que le modèle va traiter après l'entraînement. Enfin, l'utilisation d'ancres a un coût en terme des ressources requises pour utiliser le modèle et demandent aussi l'intégration d'une étape de post-traitement. C'est pourquoi un nouveau type de détecteurs qui ne font pas usage d'ancres a été inventé. Au lieu d'ajuster la position et la taille d'une boîte pré-définie, les détecteurs sans ancres détectent d'abord un ou plusieurs points caractéristiques sur les objets. Ensuite, la boîte englobante de l'objet est régressée à partir de ces points.

CornerNet : *CornerNet* [9] propose de détecter la boîte englobante d'un objet par une paire de points-clés, soit le coin supérieur-gauche et le coin inférieur-droit de la boîte. L'extraction des caractéristiques est premièrement faite avec un réseau Sablier. Ensuite, ces caractéristiques sont envoyées à un réseau neuronal convolutif qui va prédire les deux coins, par l'entremise de cartes de chaleur, et un plongement vectoriel pour chaque coin détecté. Ces cartes de chaleur correspondent à une collection de fonctions gaussiennes représentant la probabilité que le coin d'une boîte englobante s'y trouve. Il est donc facile d'extraire la position du coin à partir de chaque carte de chaleur, puisqu'il suffit de prendre la valeur maximale de chaque

fonction gaussienne. Les plongements vectoriels servent à associer les coins d'un même objet ensemble. Cela est fait en entraînant le modèle à prédire des plongements avec une petite distance si les coins appartiennent au même objet et une grande distance sinon. Le modèle prédit aussi un décalage au niveau des coins afin d'avoir une boîte englobante qui représente mieux la taille des objets. Enfin, en utilisant toutes les prédictions, les boîtes englobantes finales sont construites lors d'une étape de post-traitement. La Figure 2.7 présente un survol du pipeline de CornerNet.

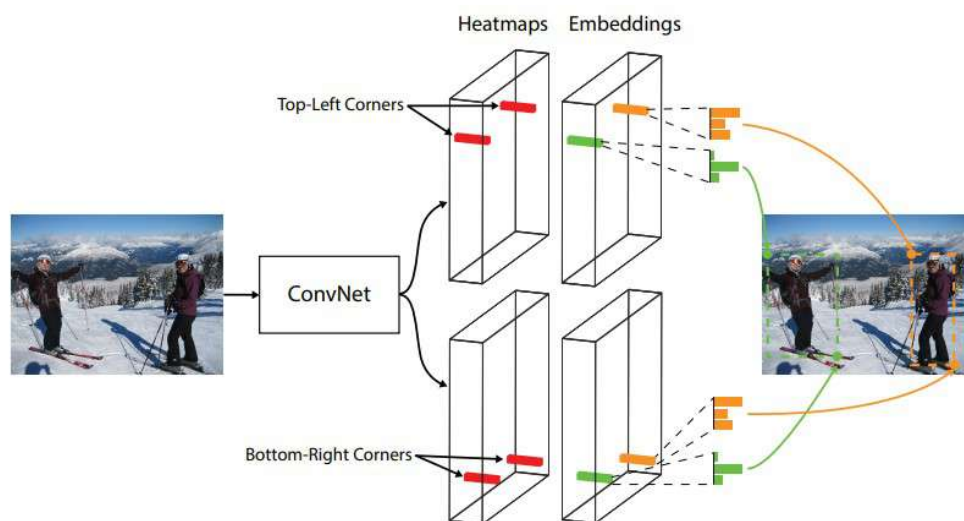


FIGURE 2.7 Pipeline de CornerNet [9] © 2019 Reproduit avec permission de Springer Nature.

Objects as points : *CenterNet* [25], le détecteur présenté par les auteurs de cet article, se base sur certains éléments de *CornerNet* et en introduit de nouveaux afin d'avoir un détecteur simple, rapide et qui obtient de bonnes performances. Comme *CornerNet*, *CenterNet* utilise une carte de chaleur, mais il détecte le point centre des objets au lieu de leurs coins. De plus, *CenterNet* utilise aussi un décalage au niveau du point centre afin de rectifier la perte d'information causée par le sous-échantillonnage inhérent aux réseaux neuronaux convolutifs. Cependant, contrairement à *CornerNet*, *CenterNet* est directement entraîné à prédire la taille de la boîte englobante à chaque pixel de l'image. Pour obtenir les boîtes englobantes finales, il suffit donc de simplement prendre les prédictions de la taille des boîtes englobantes aux positions correspondant au centre des objets et de déterminer les coordonnées des boîtes englobantes à partir de ces informations.

2.2 Suivi multiobjets

Le suivi multiobjets est une tâche relevant du domaine de la vision par ordinateur qui se base sur les travaux faits en détection d'objets. Alors que l'objectif des méthodes de détection d'objets est de détecter toutes les classes d'objets dans une image, le suivi multiobjet vise plutôt à suivre certaines classes d'objets prédéfinies dans une vidéo. Pour ce faire, il existe deux paradigmes avec lesquels on peut catégoriser les différentes méthodes de suivi : le suivi par détection et la détection et suivi simultané.

Différentes méthodes d'association d'objets sont utilisées afin de faire la correspondance des boîtes englobantes correctement et ainsi d'atteindre les meilleurs résultats. Une méthode naïve mais efficace est de simplement utiliser un algorithme glouton afin d'associer les objets d'une trame avec ceux qui sont les plus près dans la trame précédente. En effet, le mouvement d'une personne entre deux trames d'une vidéo à 30 images par secondes est extrêmement petit, alors un algorithme glouton peut être suffisant dans ce cas. Pour avoir un algorithme plus robuste, il est aussi possible de prédire le déplacement de chaque objet entre chaque trame afin d'ajuster la position de chaque objet détecté, puis d'utiliser l'algorithme glouton avec les nouvelles positions ajustées. Cependant, un tel algorithme n'est pas assez complexe pour gérer les cas les plus difficiles, comme lorsqu'un objet disparaît pendant plusieurs trames puis réapparaît à une position loin de sa position initiale. Dans un tel cas, l'algorithme glouton considérerait cet objet comme un nouvel objet, ce qui causerait une erreur. Pour alléger ce problème, il existe plusieurs méthodes basées sur des modèles statistiques qui permettent de prédire la position d'un objet même lorsqu'il n'est pas visible. Ainsi, il est possible de continuer d'associer cet objet à partir de sa position prédite. Les filtres de Kalman [26] sont un exemple d'un tel modèle statistique.

Plutôt que de faire des prédictions ou de se fier à la position des objets, il est aussi possible d'utiliser leur apparence. Les meilleures méthodes de ce type exploitent les capacités d'apprentissage des réseaux de neurones afin d'apprendre automatiquement des caractéristiques robustes qui serviront à représenter l'apparence des objets. Par la suite, il est possible d'utiliser ces caractéristiques lors de l'étape d'association afin de faire la correspondance des objets correctement en utilisant une métrique qui compare la similarité des objets. Utiliser l'apparence permet de récupérer un *tracklet* qui avait été perdu suite à une occlusion complète afin de créer une meilleure trajectoire finale. Cependant, il peut être difficile d'extraire des caractéristiques assez robustes pour discriminer correctement chaque objet et de correctement les mettre à jour au fur et à mesure que l'apparence des objets change. Généralement, les meilleures techniques d'association utilisent une combinaison de la position et de l'apparence des objets afin d'utiliser les forces des deux et de minimiser leurs faiblesses respectives.

2.2.1 Suivi par détection

Le paradigme de suivi multiobjet le plus commun est le suivi par détection. Ce paradigme inclut toutes les méthodes où les objets sont détectés dans chaque trame de la vidéo dans un premier temps, puis, dans un deuxième temps, les détections sont associées ensemble d'une trame à l'autre. La détection et l'association temporelle sont donc séparées en deux étapes distinctes.

Simple Online and Realtime Tracking : *Simple Online and Realtime Tracking* (SORT) [27] est une méthode de suivi de personnes simple mais efficace. Elle se base sur le détecteur d'objet *Faster R-CNN* afin de détecter les objets. Ensuite, pour l'étape d'association, cette méthode utilise un filtre de Kalman afin de prédire la position des objets d'une image à l'autre, puis compare ces prédictions avec les détections faites par *Faster R-CNN* à chaque trame. Cela permet de calculer la distance de Jaccard entre les prédictions et les détections, puis de faire une association des objets avec l'algorithme hongrois. Cette méthode permet aussi d'éliminer les mauvaises correspondances en contraignant la distance de Jaccard avec un seuil. En effet, si la distance entre la prédiction du filtre de Kalman et d'une détection est trop grande, il n'est pas possible que ce soit le même objet. Les objets restants sont donc considérés comme étant des nouveaux objets qui n'étaient pas présents auparavant.

Les auteurs de *SORT* ont ensuite itérés sur leur méthode avec l'introduction de *Deep-SORT* [28]. Les changements majeurs par rapport à la version précédente sont reliés à l'étape d'association. D'abord, un modèle de ré-identification de personne a été ajouté. Un tel modèle permet d'extraire une représentation vectorielle des objets dans la vidéo. Ces vecteurs peuvent ensuite être comparés d'une trame à l'autre à l'aide de la similarité cosinus afin d'obtenir une mesure de distance. La combinaison d'une mesure de distance sur la position et sur l'apparence des objets permet de faire une meilleur association. Un autre nouvel élément est le concept d'association en cascade, où les *tracklets* les plus récents sont associés avec les nouvelles détections en premier. Donc un *tracklet* qui n'a été associé avec aucune détection pendant plusieurs trames de suite sera attribué une moins grande priorité puisqu'il se peut que l'objet associé à ce *tracklet* soit disparu de la vidéo. L'association est donc séparée en deux étapes. Une première étape d'association est effectuée en utilisant l'apparence et la position des objets. Puis, pour les *tracklets* et les détections qui n'ont pas été associés avec lors de cette première étape, une deuxième étape d'association est faite en suivant la même méthode que dans *SORT* puisqu'il est possible qu'un ou plusieurs objets aient changés drastiquement d'apparence d'une trame à l'autre.

ByteTrack : *ByteTrack* [29] soulève le problème que les objets occlus auront tendances à avoir un score de confiance bas et seront complètement éliminés des détections par la

majorité des détecteurs puisque ces derniers utilisent un seuil sur le score de confiance minimal nécessaire pour qu’une détection soit considérée comme étant valide. Cela peut causer une augmentation des trajectoires fragmentées et des erreurs de suivi comme les faux négatifs et les échanges d’identifiants. *ByteTrack* propose donc une nouvelle méthode d’association qui peut être facilement ajoutée à un détecteur existant afin de créer un modèle de suivi multiobjet. D’abord, les détections avec un haut score de confiance sont associées en premier. Similairement à SORT, un filtre de Kalman est utilisé afin de prédire la position des objets dans l’image courante à partir de l’image précédente, puis l’algorithme hongrois avec l’indice de Jaccard comme distance sont utilisés afin de faire l’association. Ensuite, la même méthode d’association est faite à nouveau, mais cette fois avec les détections restantes, soit celles avec un score de confiance bas. Les détections avec un haut score de confiance qui n’ont pas été associées sont considérées comme étant de nouveaux objets, alors elles sont utilisées afin d’initialiser de nouveaux *tracklets*. Les détections avec un bas score de confiance qui n’ont pas été associées sont supprimées. Les *tracklets* qui n’ont pas été associés sont gardés pendant 30 trames, après quoi ils sont effacés.

2.2.2 Détection et suivi simultané

De façon similaire aux avancées qui ont été faites au niveau des détecteurs d’objets, les méthodes de suivi multiobjet du paradigme du suivi par détection sont plutôt lentes puisque le suivi est fait en deux étapes : la détection des objets, puis l’association temporelle. De plus, les différentes techniques d’association temporelle communément utilisées sont lentes et complexes. Afin d’avoir des méthodes de suivi plus simples et rapides, plusieurs méthodes proposant de faire la détection et le suivi simultanément ont fait leur émergence. Ce type de méthode se distingue par le fait qu’elles utilisent un seul système pour faire la détection et le suivi et qu’ils utilisent les informations acquises lors du suivi afin d’améliorer la détection dans les trames suivantes.

Tracking objects as points : *CenterTrack* [1] est la continuation du travail des auteurs de *CenterNet* et transforme ce dernier en une méthode de suivi multiobjet. *CenterTrack* prédit une carte de chaleur afin de détecter le centre des objets, ainsi qu’un décalage et la grandeur des boîtes englobantes, exactement comme *CenterNet* le fait. Cependant, *CenterTrack* prédit aussi le déplacement de chaque objet entre la trame courante et la trame précédente. De plus, alors que *CenterNet* n’utilise qu’une seule image pour faire ses prédictions, *CenterTrack* utilise aussi l’image précédente et la carte de chaleur prédite pour l’image précédente. De cette manière, *CenterTrack* est capable d’utiliser l’information de suivi afin d’améliorer les détections du modèle. Avec toutes les prédictions du modèle, *CenterTrack* est capable de

définir la position et la boîte englobante de chaque objet et de faire l'association des objets d'une image à l'autre. L'association est faite en soustrayant le déplacement prédit des objets de la position de leur centre afin d'avoir une prédiction de leur centre à la trame précédente et en calculant la distance euclidienne entre ces centres prédits et ceux de la trame précédente. Ensuite, un algorithme glouton utilise cette distance afin d'associer les détections et les *tracklets* existants ensemble. Enfin, les détections avec un score de confiance plus grand qu'un certain seuil qui n'ont pas été associées sont utilisées pour créer des nouveaux *tracklets* et les *tracklets* qui n'ont pas été associés sont supprimés.

FairMOT : *FairMOT* [10] propose une méthode qui tente d'adresser le déséquilibre qui peut survenir lorsqu'une méthode essaie d'optimiser conjointement la détection et la ré-identification. En effet, les méthodes qui utilisent des modèles séparés pour chacune des tâches sont lents et ne réussissent pas à fonctionner en temps réel et les méthodes qui utilisent un seul modèle pour les deux tâches performant généralement moins bien à cause d'une augmentation des erreurs faites lors de l'étape d'association. Effectivement, ces méthodes traitent généralement la ré-identification comme une tâche secondaire à la détection en utilisant pour la ré-identification les caractéristiques extraites pour la détection, ce qui explique partiellement le déclin en performance. Comme le montre la Figure 2.8, *FairMOT* vise à rectifier ce déséquilibre en traitant les deux tâches avec leurs propres branches qui fonctionnent en parallèle. La branche de détection est comme *CenterNet*. La branche de ré-identification correspond à une couche de convolution qui extrait les caractéristiques de ré-identification à chaque pixel à partir des caractéristiques des objets extraites de l'image. Pour l'étape d'association, comme pour l'extraction des boîtes englobantes dans *CenterNet*, il ne suffit que de prendre les caractéristiques de ré-identifications aux endroits où un centre a été détecté afin d'extraire les bonnes caractéristiques pour chaque objet. Ensuite, deux mesures de distance sont calculées. D'abord, à l'aide d'un filtre de Kalman, la position des objets à l'image courante est prédite à partir de leur position à l'image précédente et puis est comparée avec les détections de cette image à l'aide de la distance de Mahalanobis. Puis, une deuxième distance est calculée en prenant les caractéristiques de ré-identification aux pixels correspondant aux centres des objets à chaque image et en les comparant d'une image à l'autre avec la distance cosinus. Une distance finale est ensuite calculée en faisant une somme pondérée des deux distances précédentes. L'association entre les détections et les *tracklets* existants est ensuite faite avec l'algorithme hongrois. Une deuxième phase d'association est faite avec les détections et les *tracklets* qui n'ont pas été associés lors de la première phase. Elle consiste uniquement à faire l'association entre les détections et les *tracklets* restants avec l'indice de Jaccard et un seuil minimum de 0,5. Enfin, les détections toujours pas associées sont utilisées pour créer de nouveaux *tracklets* et les *tracklets* restants sont gardés en mémoire durant 30

trames, après quoi ils sont effacés.

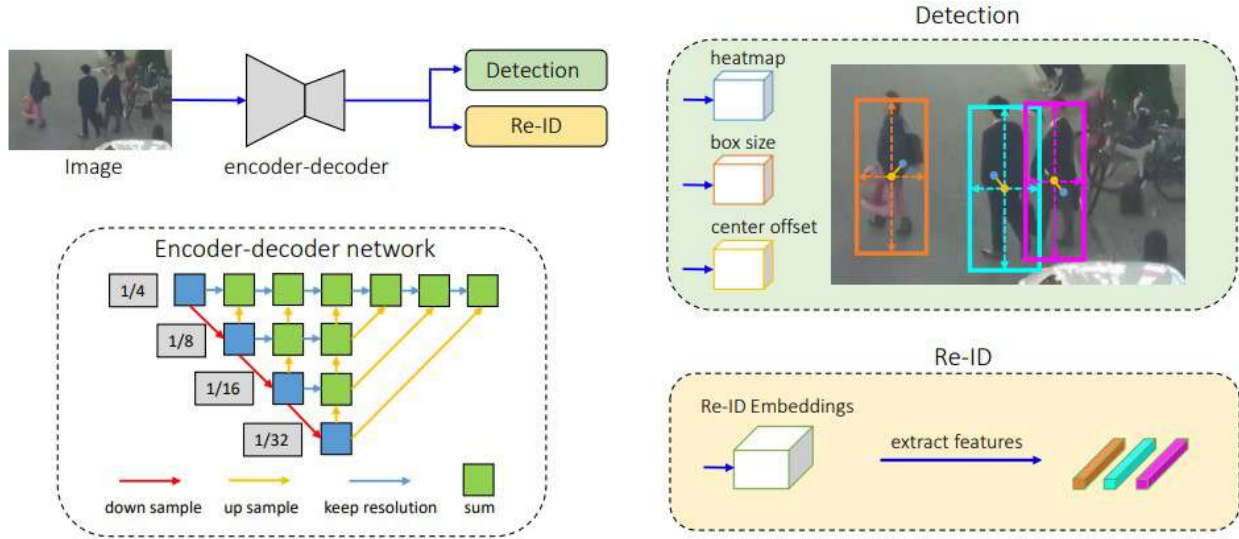


FIGURE 2.8 Architecture de FairMOT [10] © 2021 Reproduit avec permission de Springer Nature

TrackFormer : Malgré que, jusqu'à maintenant, la majorité des méthodes de suivi multiobjet utilisaient des réseaux neuronaux convolutifs, les réseaux autoattentifs ont réussi à faire leurs preuves aussi pour cette tâche. La Figure 2.9 montre un aperçu de l'architecture et du fonctionnement de *TrackFormer* [11]. D'abord, *TrackFormer* utilise aussi un réseau neuronal convolutif afin d'extraire les caractéristiques de l'image, puis un encodeur et un décodeur de réseau autoattentif afin de faire la détection et l'association simultanément. Pour la première image de la séquence vidéo, le décodeur ne prends que des requêtes d'objet fixes (carrés blancs dans la figure) qui sont apprises lors de l'entraînement. Dans le contexte des réseaux autoattentifs et des mécanismes d'attention, une requête est une représentation vectorielle qui définit le jeton qui sera comparé à tous les autres jetons. Pour les images subséquentes, les requêtes d'objets et de suivi prédites par le décodeur à l'image précédente sont combinées ensemble afin de permettre au modèle de raisonner sur les nouveaux objets et sur les objets qu'il a déjà vu. Comme le décodeur prédit directement les *tracklets*, aucune étape d'association supplémentaire n'est nécessaire.

TransCenter : *TransCenter* [12] est aussi une méthode qui se base sur les réseaux autoattentifs afin d'accomplir la tâche de suivi. Comme pour *CenterTrack*, les objets sont détectés à partir de leur point central à partir d'une carte de chaleur générée par le modèle. Cependant, les images en entrée sont envoyées directement à un encodeur, signifiant qu'aucun réseau neuronal convolutif n'est utilisé. La Figure 2.10 montre une visualisation du pipeline de traitement de *TransCenter*. L'encodeur extrait des *mémoires multiéchelles* à partir de la trame

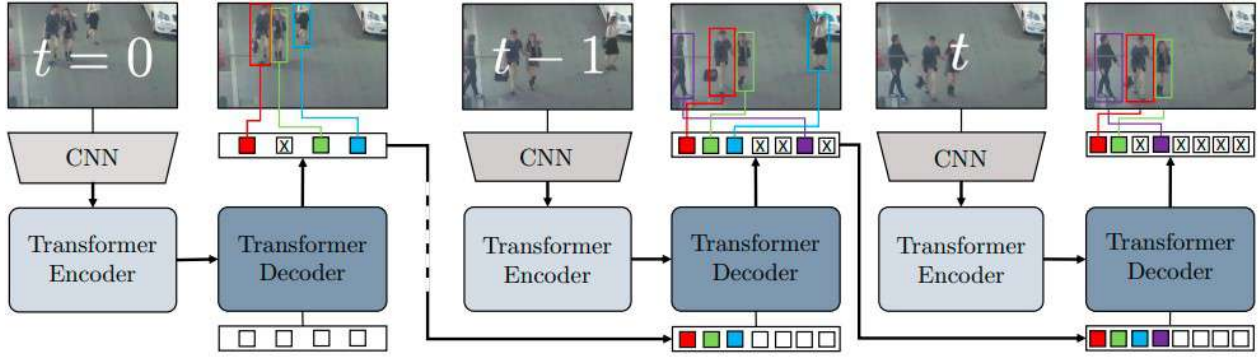


FIGURE 2.9 Fonctionnement de TrackFormer [11] © 2022 IEEE.

courante et de la trame précédente qui sont envoyées à un *Query Learning Network* (QLN). Ce nouveau module proposé par les auteurs apprend à générer des requêtes pour chaque objet détecté. Ces requêtes permettent au décodeur d'associer facilement chaque objet à sa trajectoire. Une *mémoire de détection* et une *mémoire de suivi* ainsi que des requêtes de détection et des requêtes de suivi sont créées par le QLN, puis envoyées vers le décodeur. Ce dernier utilise la *mémoire de détection* et les requêtes de détection pour extraire des caractéristiques de détections et utilise la *mémoire de suivi* et les requêtes de suivi pour extraire des caractéristiques de suivi. Les caractéristiques de détection sont utilisées pour prédire la carte de chaleur et la taille des boîtes englobantes, alors que les caractéristiques de suivi sont utilisées pour prédire le déplacement des objets d'une trame à l'autre.

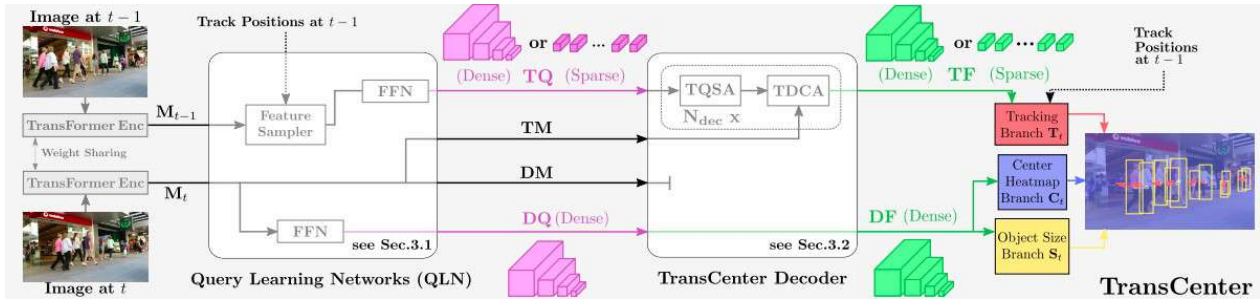


FIGURE 2.10 Pipeline de TransCenter [12] © 2022 IEEE.

2.3 Extractions de caractéristiques

Afin de pouvoir correctement détecter et suivre des objets, il est crucial que la méthode apprenne une représentation adéquate de ces objets. Cela est fait par l'extraction des caractéristiques des objets. Cette section vise à expliquer en détail les différentes manières dont l'extraction de caractéristiques est faite.

2.3.1 Réseau neuronal convolutif

Le but d'un tel réseau est d'extraire une carte d'attributs qui servira à représenter un objet. Ce type de réseau est généralement composé de trois types de couche : une couche de convolution, une couche de sous-échantillonnage et une couche entièrement connectée. Il peut y avoir plusieurs couches d'un même type de suite, cependant un réseau neuronal convolutif commence toujours par une couche de convolution et se termine par des couches entièrement connectées (perceptron multicouche) si l'on veut utiliser la carte d'attributs générée pour faire la tâche de classification ou régression finale. Sinon, le réseau se termine par une couche de sous-échantillonnage et ne génère que la carte d'attributs. La couche de convolution reçoit l'image en entrée, puis fait glisser un filtre d'une taille prédéterminer (généralement 1×1 , 3×3 ou 5×5) à travers l'image comme montré par la Figure 2.11. Ce processus s'appelle une convolution.

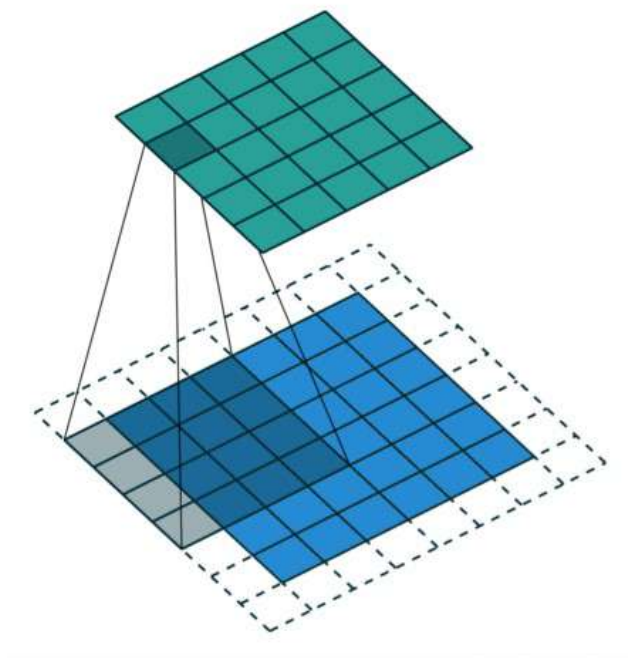


FIGURE 2.11 Exemple de convolution. *Droits d'auteur* © Vincent Dumoulin, Francesco Visin [13].

À chaque position de l'image, un produit scalaire est effectué entre les éléments du filtre et les éléments de l'image, créant ainsi une nouvelle matrice en sortie appelée carte d'attributs (matrice verte dans la Figure 2.11). Les éléments du filtre sont des poids appris par le modèle au cours de l'entraînement afin que ce dernier apprenne à créer une meilleure représentation de l'objet par lui-même. Il existe aussi d'autres hyperparamètres afin de modifier ce proces-

sus. D’abord, il est possible d’avoir plusieurs filtres différents afin d’avoir plusieurs cartes d’attributs à la sortie. Ensuite, il est possible de modifier la grandeur du pas par lequel le filtre se déplace. Enfin, si la taille de l’image et du filtre sont incompatible, il est possible d’ajouter un remplissage qui va ajouter des éléments avec la valeur de zéro à l’image (éléments en blanc dans la Figure 2.11). La couche de sous-échantillonnage suit le même processus que la couche de convolution, mais au lieu de faire un produit scalaire entre les éléments du filtre et de l’image, le filtre prends l’élément avec la plus grande valeur ou fait une moyenne des valeurs de tous les éléments de la carte d’attributs couverts par le filtre. Cette couche réduit l’information contenu dans l’image. Cependant elle permet aussi de réduire la complexité du réseau et de réduire les risques de surapprentissage. Finalement, le perceptron multicouche est une collection de neurones interconnectées dont le but est de faire la tâche de classification ou de régression finale du réseau, si requis.

2.3.2 Réseau autoattentif

Les transformeurs sont un type de réseau d’abord créé pour les tâches de traitement de la langue naturelle, mais qui ont éventuellement été adaptés pour les tâches relevant de la vision par ordinateur. Comme le montre la Figure 2.12, ce type de réseau se distingue par l’utilisation d’une pile d’encodeurs et de décodeurs, ainsi que par l’utilisation de différents mécanismes d’attention.

Les données en entrées doivent premièrement être transformées en jetons afin de pouvoir être utilisés par le réseau autoattentif. Cela peut être fait de plusieurs façon dans le cas de la vision par ordinateur. *Vision Transformer* [30] propose de séparer l’image en plusieurs tuiles, puis de faire une projection linéaire afin de les transformer en un vecteur 1D, alors que *TrackFormer* utilise directement les caractéristiques extraites par un réseau de neurones convolutif. Ensuite, un plongement de position est ajouté à chaque jeton afin d’incorporer de l’information tant qu’à la position de chaque jeton dans la séquence. Effectivement, le réseau autoattentif prend toute la séquence en même temps et traite chaque jeton individuellement dans aucun ordre précis. Le plongement de position vise alors à intégrer l’information de la position directement dans chaque jeton. La séquence peut ensuite être envoyée à l’encodeur qui consiste d’un mécanisme d’auto-attention suivi d’un perceptron multicouche. Le mécanisme d’auto-attention sert à quantifier la pertinence de chaque jeton en relation avec les autres jetons de la séquence, ce qui permet au réseau de raisonner à propos de la position des objets dans l’image. Généralement, plusieurs encodeurs sont utilisés un à la suite de l’autre. Le but de la pile d’encodeurs est de générer des encodages représentant les parties de la séquence qui sont importantes entre elles. Ces encodages sont ensuite envoyés à une pile de décodeurs. Les

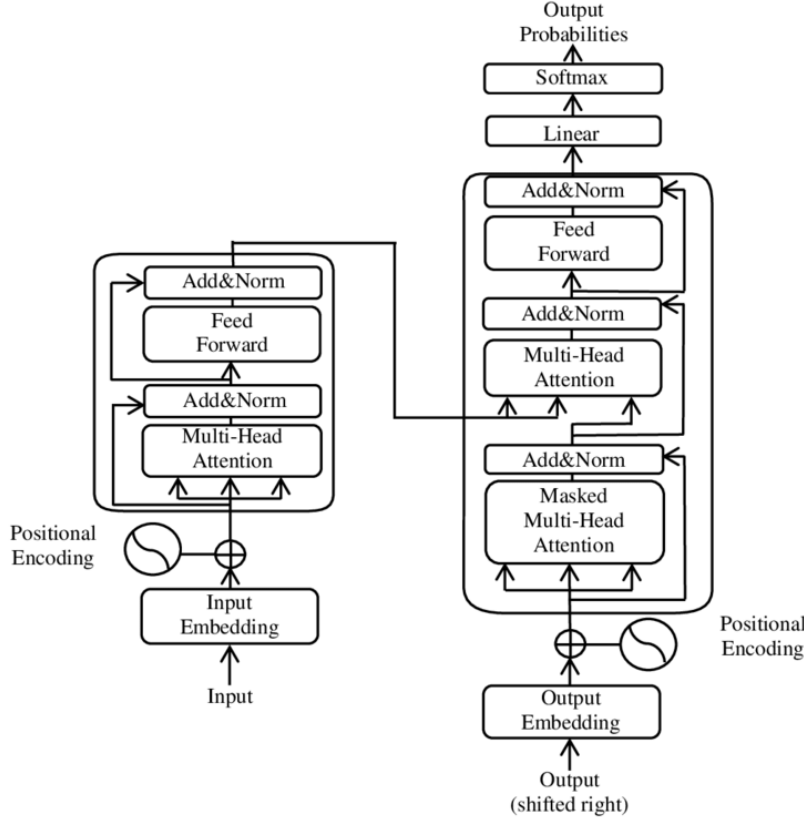


FIGURE 2.12 Architecture générale des réseaux autoattentif [14].

décodeurs font l'inverse des encodeurs et utilisent les encodages afin de générer une nouvelle séquence en sortie. Chaque décodeur consiste de deux mécanismes d'auto-attention et d'un perceptron multicouche. Le premier mécanisme d'auto-attention reçoit la séquence générée pour l'image précédente et sa sortie est envoyée au deuxième mécanisme d'auto-attention. Cela permet d'utiliser l'information des objets extraite de l'image précédente afin d'aider le modèle à faire l'association des objets d'une image à l'autre. Finalement, le perceptron multicouche génère la séquence finale.

2.4 Métriques d'évaluation

Afin de bien comprendre les résultats qui seront présentés au chapitre 4, il est important d'être familier avec les différentes erreurs qui peuvent survenir lors du suivi multiobjet et avec les métriques qui ont été créées afin de les mesurer. Les trois erreurs principales sont les faux positifs, les faux négatifs et les erreurs d'association (IDSW). Un faux positif survient si un objet est détecté à une position alors qu'il n'y en a pas, un faux négatif survient si

un objet présent dans la vidéo n'est pas détecté par la méthode et une erreur d'association survient lorsqu'un objet se fait assigner un identifiant différent que précédemment.

Plusieurs autres métriques peuvent ensuite être dérivées à partir des FP, FN et IDSW pour permettre une meilleure comparaison des méthodes et de leur capacité à accomplir les différents requis du suivi multiobjet. D'abord, la métrique la plus commune pour comparer les méthodes de suivi multiobjet est le *Multiple Object Tracking Accuracy* (MOTA) [31] qui mesure la quantité d'erreurs de détection et d'association qu'une méthode commet

$$MOTA = 1 - \frac{\sum_t FN_t + FP_t + IDSW_t}{\sum_t GT_t}, \quad (2.1)$$

où t représente l'index de la trame courante et GT représente le nombre d'objets dans la scène selon la vérité de terrain.

MOTA donne une bonne idée de la qualité d'une méthode puisqu'elle comptabilise le nombre d'erreurs qui sont commises, mais elle ne mesure pas à quel point la méthode est bonne pour localiser correctement les objets qui sont détectés. C'est pourquoi une autre métrique, le *Multiple Object Tracking Precision* (MOTP) [31], est nécessaire pour évaluer les capacités de localisation d'une méthode. Elle est calculée par

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}. \quad (2.2)$$

Pour tous les objets correctement détectés à la trame t , la distance de Jaccard d entre la position prédite de la détection et celle de la vérité de terrain est calculée pour chaque objet i et est divisée par le nombre d'objets correctement détectés c .

La précision et le rappel représente respectivement combien d'objets détectés sont pertinents et combien d'objets pertinents ont été détectés. Ces deux métriques sont calculées à partir des vrais positifs, soit les bonnes détections, des faux positifs et des faux négatifs

$$Précision = \frac{TP}{TP + FP}, \quad Rappel = \frac{TP}{TP + FN}. \quad (2.3)$$

Avec la précision et le rappel, il est possible d'obtenir le F-score. Dans le contexte du suivi multiobjet, la métrique s'appelle plutôt *IDF1* [32] puisqu'elle sert à mesurer la justesse de l'association des données et s'effectue donc seulement au niveau des identifiants qui sont attribués aux objets détectés. Cette métrique est donnée par

$$IDF1 = 2 \cdot \frac{Précision \cdot Rappel}{Précision + Rappel} = \frac{2TP}{2TP + FP + FN}. \quad (2.4)$$

La combinaison de MOTA, MOTP et IDF1 permet d’avoir une bonne représentation générale des capacités d’une méthode à accomplir chaque tâche du suivi multiobjet, soit la détection, la localisation et l’association. Cependant, avoir trois métriques complexifie aussi la comparaison des différents méthodes. La métrique *Higher Order Tracking Accuracy* (HOTA) [15] vise à évaluer les trois aspects du suivi en une seule métrique. Pour ce faire, un score pour chaque est premièrement calculé en utilisant l’indice de Jaccard. Ensuite, les trois scores sont combinés ensemble afin d’obtenir le score final.

Le score pour la localisation *Localization Accuracy* (LocA) est obtenu en prenant la moyenne des indices de Jaccard (Loc-IoU) pour toutes les paires de détections prédites et de vérités de terrain. correctement appariées

$$LocA = \frac{1}{TP} \sum_{c \in TP} Loc-IoU(c). \quad (2.5)$$

Le score pour la détection *Detection Accuracy* (DetA) est obtenu en prenant directement l’indice de Jaccard (Det-IoU) entre les détections prédites et les détections des vérités de terrain. La différence avec la localisation est que la localisation mesure à quel point la boîte englobante prédite est près de la vérité alors que la détection considère seulement si l’objet a été détecté ou pas. Pour déterminer quelles détections intersectent, on conditionne l’indice de Jaccard avec un seuil de localisation α . Cela permet de faire la correspondance entre les deux ensembles de détections pour calculer les éléments de la matrice de confusion nécessaires au calcul de l’indice de Jaccard. Puis l’algorithme Hongrois est utilisé puisqu’il est possible que plusieurs détections intersectent ensemble malgré le seuil. L’ensemble de détections qui ont été appariées correspond aux vrais positifs, les détections restantes de l’ensemble prédit sont les faux positifs et les détections restantes de l’ensemble des vérités de terrain sont les faux négatifs.

$$DetA = Det-IoU = \frac{TP}{TP + FP + FN} \quad (2.6)$$

Le score pour l’association *Association Accuracy* (AssA) mesure à quel point une méthode est capable de lier les détections d’un certain objet au même *tracklet* à travers le temps. Pour ce faire, une détection prédite est associée à une détection des vérités de terrain, puis l’indice de Jaccard (Ass-IoU) est calculé entre le *tracklet* complet de la détection prédite et le *tracklet* complet de la détection des vérités de terrain. Un TP est défini comme une détection qui est présente dans les deux *tracklets*. Les détections présentent dans seulement un des deux

tracklets sont soit des FP ou de FN.

$$Ass-IoU = \frac{TP}{TP + FP + FN} \quad (2.7)$$

Un exemple est donné à la Figure 2.13. Le carré rouge réfère à la détection prédite associée à une détection des vérités de terrain pour laquelle le score d'association sera calculée. Ensuite, toutes les détections des deux *tracklets* sont comparées afin de déterminer lorsqu'il y a des correspondances entre les deux afin de pouvoir calculer les TP, FP et FN.

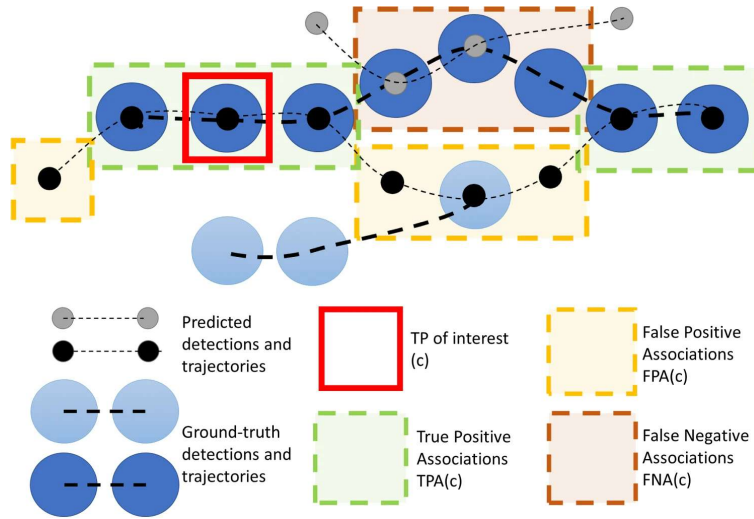


FIGURE 2.13 Exemple de TP, FP et FN pour le calcul du score d'association de HOTA [15]

Cette procédure donne le score d'association pour une seule paire de détections, alors pour avoir le score global, il faut faire la moyenne du score d'association pour toutes les paires de la base de données.

$$AssA = \frac{1}{TP} \sum_{c \in TP} Ass-IoU(c) \quad (2.8)$$

Étant donné que le score de localisation est pris en compte implicitement par les deux autres scores à cause du seuil de localisation α qui est utilisé lorsqu'on détermine une correspondance entre une détection prédite et une détection des vérités de terrain, le calcul final de *HOTA* est calculé en intégrant la moyenne géométrique du score de détection et d'association sur le seuil de localisation. En pratique, l'intégrale peut être approximée par une moyenne sur plusieurs scores de localisation uniformément distribués entre 0 et 1.

$$HOTA_{\alpha} = \sqrt{DetA_{\alpha} \cdot AssA_{\alpha}} \quad (2.9)$$

$$HOTA = \int_{0 < \alpha \leq i} HOTA_{\alpha} \quad (2.10)$$

La dernière métrique communément utilisée permet de savoir à quel point une méthode est en mesure de créer un *tracklet* complet pour les objets. Pour ce faire, le nombre de *tracklets* principalement suivis (MT) [31] et principalement perdus (ML) [31] sont calculés. Un *tracklet* est considéré principalement suivi s'il a capturé au moins 80% de la trajectoire réelle d'un objet et il est considéré principalement perdu s'il a capturé moins de 20% de la trajectoire réelle d'un objet. Parfois, une métrique intermédiaire, où le *tracklet* est considéré comme partiellement suivi (PT), est aussi calculée. Un *tracklet* correspond à cette catégorie s'il a capturé entre 20% et 80% de la trajectoire réelle d'un objet.

CHAPITRE 3 DÉMARCHE DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE

Lors de la revue de littérature initiale pour définir ce projet de maîtrise, l'article *TransCenter : Transformers with Dense Representations for Multiple-Object Tracking* [12] avait piqué notre intérêt. Les auteurs de *TransCenter* adoptent des techniques pour réduire les ressources computationnelles requises par leur méthode. Cependant, les réseaux autoattentifs restent de très gros réseaux qui demandent beaucoup de ressources lors de l'entraînement alors il était difficile dans notre cas de les utiliser. De plus, les méthodes basées sur les réseaux neuronal convolutifs réussissent encore à obtenir des résultats à l'état de l'art, alors il n'est pas nécessaire d'utiliser un réseau autoattentif si l'on veut avoir des résultats compétitifs. Nous avons donc ensuite tourné notre attention vers *CenterTrack* [1], puisque c'est une méthode simple, rapide et qui réussit à atteindre des bons résultats. De plus, cette méthode utilise l'apprentissage automatique pour prédire le déplacement des objets au lieu de se fier à un modèle statistique, ce qui permet de mieux suivre les objets qui changent leur mouvement brusquement. Selon leurs études d'ablation, où leur méthode est comparée aux filtres de Kalman [26] et au flux optique [33], leur méthode fonctionne mieux que les autres lorsque la fréquence d'images est plus basse, comme sur la base de données *KITTI Tracking* [3], puisque le déplacement des objets est plus grand d'une trame à l'autre que lorsque la fréquence d'image est plus grande. L'objectif est donc devenu d'itérer sur *CenterTrack* afin de l'améliorer. Les idées de départ étaient surtout centrées autour de l'architecture de la méthode. En effet, *CenterTrack* requiert en entrée l'image courante, l'image précédente et la carte de chaleur générée par le modèle à l'itération précédente. Cependant, plusieurs têtes de prédiction n'ont pas besoin de toutes ces informations alors il n'est pas nécessaire d'avoir autant d'entrées pour le réseau. Une autre idée, qui s'est avérée bénéfique, était de fournir la carte de chaleur générée durant l'itération courante à la tête de prédiction qui prédit le déplacement des objets en addition aux autres entrées.

En court de route, en évaluant les résultats de *CenterTrack* avec un outil de visualisation, nous avons remarqué que le modèle ne parvient souvent pas à détecter les objets lorsque leur point central est caché, même si le reste de l'objet est visible. Cela nous a donné l'idée d'essayer d'utiliser le haut des objets, la tête dans le cas d'une personne, comme point-clé pour la détection au lieu du centre puisque ce point-clé est généralement plus souvent visible. Par exemple, pour la base de données *MOT17* [2] où il faut suivre des piétons, il est assez rare que la tête d'une personne soit cachée mais pas le reste de son corps, alors que l'inverse est plus commun. Par contre, utiliser le centre comme point-clé a aussi certains avantages.

Effectivement, lors de la création des annotations, il est très facile de déterminer le point centre d'un objet à partir de sa boîte englobante, alors qu'il n'est pas évident a priori de trouver un point qui va adéquatement capturer le haut des objets dans tous les scénarios. Cet exemple est aussi vrai en inférence, lorsqu'il faut faire le processus inverse et déterminer les coordonnées de la boîte englobante d'un objet à partir de la position du point-clé utilisé pour la détection. Lorsque le point-clé est le centre, il suffit de prendre la moitié de la largeur et de la hauteur de la boîte englobante prédite tandis que lorsque le point-clé est le haut de l'objet, les proportions qu'il faut utiliser ne sont pas autant évidentes. C'est pourquoi la première approche adoptée a été d'utiliser les deux point-clés simultanément afin d'augmenter les détections faites à partir du centre avec celles faites à partir du haut des objets. Ainsi, la majorité de l'architecture resterait pareille, mais on serait en mesure de capturer certaines détections qui étaient manquées auparavant. En pratique, la méthode adoptée consistait à ajouter une tête de prédiction pour générer une carte de chaleur qui détecte le haut des objets et une tête de prédiction afin de déterminer la distance entre le centre et le haut des objets. En effet, étant donné que le reste de la méthode fonctionne en assumant que les détections représentent le point centre des objets, utiliser les détections faites avec le haut comme point-clé tel quel ne fonctionnerait pas. Il faut donc réussir à trouver le point centre des objets qui ont été détectés avec la nouvelle carte de chaleur afin que ces détections soient compatibles avec le reste de la méthode. Pour ce faire, les coordonnées du haut des objets étaient ajustées en utilisant la distance entre le centre et le haut qui a été prédite. Ensuite, il était devenu possible de combiner toutes les détections en un ensemble, de les utiliser pour extraire les boîtes englobantes et puis d'éliminer les détections doubles avec NMS [16]. Malheureusement, cette approche ne fonctionnait pas très bien pour deux raisons. D'abord, la prédiction de la taille des boîtes englobantes étaient souvent pas très bonne aux endroits où le centre des objets n'était pas visible. Puis la prédiction de la distance entre le centre et le haut des objets ajoutaient beaucoup de variance alors les nouvelles détections ajoutées avec cette approche ne représentaient souvent pas correctement la localisation des objets.

Afin d'éliminer ces deux problèmes, la deuxième approche consistait à créer deux branches qui exécutent en parallèle le processus complet de détection mais avec point-clé différent. De cette manière, il n'était plus nécessaire d'essayer de trouver le centre des objets détectés par leur haut puisque l'extraction des boîtes englobantes étaient faites avec le haut aussi. Ensuite, une fois les deux ensembles de détections combinées, les détections doubles pouvaient être éliminées en utilisant NMS. Selon des expériences sur la base de données *MOT17*, le haut des objets se trouve généralement à $\frac{1}{10}$ de la hauteur et à la moitié de la largeur de leur boîte englobante, et ce, peu importe leur taille. Il est donc devenu possible d'extraire les coordonnées des boîtes englobantes à partir de la position du haut des objets. Nous avons

spécifiquement choisi la base de données *MOT17* puisque c'est une base de données très populaire dans le domaine du suivi multiobjet et elle inclut une grande diversité au niveau des caractéristiques des objets. Cette approche a cependant l'inconvénient de grandement augmenter la grosseur du réseau et il était difficile de trouver un seuil adéquat lorsque NMS détermine quelles détections sont des doublons puisque dans un contexte de foule, il va inévitablement avoir plusieurs objets dont les boîtes englobantes vont se chevaucher. Cette méthode enlevait donc des vrais positifs lorsque le seuil était haut, gardait des faux positifs lorsque le seuil était bas et il ne semblait pas avoir de juste milieu adéquat. Nous avons aussi essayé d'utiliser d'autres méthodes plus complexes pour combiner les deux ensembles de détections et éliminer les détections doubles, mais le résultat était le même. Soit beaucoup de bonnes détections étaient éliminées, soit beaucoup de détections doubles étaient gardées causant beaucoup de faux positifs ou un peu des deux.

La dernière approche a donc été de se débarrasser complètement de la branche qui utilise le centre comme point-clé. En effet, cette branche était devenue obsolète puisque les raisons principales pour lesquelles le centre était utilisé comme point-clé originalement n'était plus applicables. Cette approche permet aussi de diminuer la grosseur du réseau à sa taille originale et de ne pas avoir à trouver une façon d'éliminer les détections doubles causées par la création de deux ensembles de détections.

CHAPITRE 4 ARTICLE 1 : TOPTRACK : TRACKING OBJECTS BY THEIR TOP

Jacob Meilleur, Guillaume-Alexandre Bilodeau

Soumis pour publication au journal Machine Vision and Applications le 11 avril 2023

4.1 Abstract

In recent years, the joint detection-and-tracking paradigm has been a very popular way of tackling the multi-object tracking (MOT) task. Many of the methods following this paradigm use the object center keypoint for detection. However, we argue that the center point is not optimal since it is often not visible in crowded scenarios, which results in many missed detections when the objects are partially occluded. We propose *TopTrack*, a joint detection-and-tracking method that uses the top of the object as a keypoint for detection instead of the center because it is more often visible. Furthermore, *TopTrack* processes consecutive frames in separate streams in order to facilitate training. We performed experiments to show that using the object top as a keypoint for detection can reduce the amount of missed detections, which in turn leads to more complete trajectories and less lost trajectories. *TopTrack* manages to achieve competitive results with other state-of-the-art trackers on two MOT benchmarks.

4.2 Introduction

Multi-object Tracking (MOT) is a hallmark task of computer vision that consists of detecting all the objects of predefined classes in each frame of a video and linking together the detections of the same object through time in order to create a trajectory for each object. Most state-of-the-art (SOTA) online trackers follow the tracking-by-detection paradigm, where the tracking task is divided into two subtasks : object detection and data association. Firstly, objects are detected in each frame and then the objects are associated with each other in a separate step using mostly motion and appearance cues extracted from each frame or group of frames. This method works well, but usually requires a high computational cost and often relies on complex data association methods. Because of this, recent works have introduced the joint-tracking-and-detection paradigm where the tracking task is reconciled into a single framework. These models manage to achieve competitive results while being both simpler and faster than models following the tracking-by-detection paradigm. *CenterTrack* [1] is such a model. It represents objects by their center point to perform detection and tracking and then regresses the bounding boxes from these center points in order to extract the position and size of every

object. *CenterTrack* demonstrated that tracking objects at the point-level simplifies two core components of MOT. Firstly, it allows for the representation of the candidate objects as a heatmap of points, which can then be fed as input to the network so that it can reason about every object in the frame jointly with those of the previous frame. Secondly, it simplifies data association, which can be performed with a simple displacement prediction that is conditioned on prior detections.

However, *CenterTrack* also has some drawbacks. Indeed, occluded objects remains one of MOT biggest problem to date, even with the introduction of the joint-tracking-and-detection paradigm. This is especially true for crowded scenes, where there are many partially or fully occluded objects. In these scenes, the center of the objects is often not visible, which can affect the performance of *CenterTrack* and other similar trackers, since they rely on the center point of the objects for detection. This is sometimes alleviated using the heatmap of the previous frame as input to the network to allow it to carryover some detections from one frame to the other, even if the object is not visible in the current frame. However, this is often not the case and detections are missed because the center of the objects is not visible while the rest is.

In this paper, we propose a model that uses the top of the objects as a keypoint for detection instead of the center because it is more often visible in these scenarios, which in turn allows for better detection results. Moreover, we propose a simpler and more intuitive architecture that does not rely on the generated heatmap as input for the whole network while training, but instead only uses it for tracking purposes. We also show how using a combination of the current frame, the previous frame, the current heatmap and the previous heatmap for tracking allows our method to better connect objects locally.

We evaluated our method on the MOT17 challenge benchmark [2] and the MOT20 challenge benchmark [34] and we show how our proposed method manages to better detect hard-to-track objects, resulting in more complete trajectories and less lost tracks. Our contributions are summarized as follows :

- We introduce *TopTrack*, an end-to-end trainable multi-object tracker that uses the top of the objects for detection and tracking. We show that using this keypoint instead of the center of the objects is beneficial for this task.
- We show that our method can achieve competitive tracking accuracy on popular MOT datasets : MOT17 and MOT20.

4.3 Related Work

Most SOTA trackers fall under the tracking-by-detection paradigm [29, 35–37], which is characterized by performing the object detection and data association in two distinct stages, and is often accomplished using two distinct models. An off-the-shelf object detector [5, 23, 38–40] is used in order to get candidate bounding boxes for each frame. Then, data association can be performed between the bounding boxes of each subsequent frame in order to link the identities of each object through time. The data association stage often makes use of motion and appearance cues in order to achieve better results. Commonly used motion cues are motion prediction models like Kalman filters [26, 41] and camera motion compensation [42]. Appearance cues include ReID features obtained from a pre-trained model [43–45] or learned jointly [10, 46, 47]. ReID features can also be augmented with other visual information like pose estimation [48].

Examples of tracking-by-detection methods include *SORT* [27] that uses *Faster R-CNN* [5] to perform object detection and uses Kalman filters [26] to predict the motion of each object. Data association is then performed between the prediction of the Kalman filters and the detector using Intersection over Union (IoU) as a distance metric and the Hungarian algorithm for matching [17]. *DeepSORT* [28] adds appearance information and cascade matching in the data association stage of *SORT* in order to improve results. *StrongSORT* [37] updates the components of *DeepSORT* to better performing ones and introduces a new global tracklet linking module and a Gaussian smoothing interpolation module to achieve SOTA results. *SST* [49] proposes a Deep Affinity Network, which models the appearance of the objects and does the data association by computing the affinity between the existing tracks and the detected object and then matching them using the Hungarian algorithm [17]. *C-BIOU* [35] uses detections from *YOLOX* [23] then performs cascade matching between existing tracklets and the detections. The size of the bounding boxes is enlarged by a constant to allow for the tracking of hard examples where the position of the object has changed a lot. The first matching uses a small enlargement factor, then the remaining unmatched tracks and detections are matched again using a large enlargement factor. *ByteTrack* [29] proposes a data association strategy that focuses on matching all detections extracted from an off-the-shelf object detector, even those with a low confidence score instead of ignoring them. This is done by first matching the detection with a high confidence score, determined by a fine-tuned threshold, with the existing tracks. Then, the detections with lower confidence scores are matched with the remaining tracks.

Many recent trackers [1, 10–12, 50–55] are using a novel paradigm, where the detection and tracking are performed jointly. These methods are characterized by their use of tracking

information in order to boost the detection results in future frames and by using a single model in order to make the predictions for both detection and tracking. Then, a post-processing step will use these predictions in order to obtain the final results. *TubeTK* [55] and *ChainedTracker* [53] pioneered the concept of joint-detection-and-tracking by proposing a completely end-to-end trainable pipeline. *TubeTK* proposes the concept of *bounding-tubes* to represent the spatiotemporal location of objects detected in a video. A *bounding-tube* is defined as a series of three bounding boxes of the same object from different frames. Tracks are broken down into a combination of tubes where the bounding box at each frame is defined as the middle bounding box of a tube in order to link tubes together. The three bounding boxes forming a tube do not have to be from consecutive frames, which allows the interpolation of the object location within a tube. Tubes are linked using IoU as a distance metric. *ChainedTracker* takes two adjacent frames as input and generates detection pairs of each object in both frames and then link each consecutive pair using IoU as a distance metric and then the Hungarian algorithm [17] for matching.

CenterTrack [1] learns to generate a center heatmap for object detection and an offset vector that represents the displacement from one frame to the next using the two consecutive frames and the center heatmap of the previous frame. It then regresses the bounding boxes from the center points of the objects. Data association is done using greedy matching between the position of the objects in the previous frame and the position of the predicted offset. *Tracktor* [50] exploits the bounding box regression module of *Faster R-CNN* to perform tracking directly from the information provided by the detector. Moreover, the model can be extended with a motion model and a re-identification algorithm to achieve better results. *FairMOT* [10] aims to reconcile the inherent bias in favor of the detection task when training a MOT model. Indeed, they noted three ways in which trackers following the joint detection-and-tracking paradigm are biased towards the detection task, which harms the global results. As a result, they use *CenterNet* [25] for their detection branch and add a parallel ReID feature embedding branch to generate robust appearance features that are then used during the data association step to improve their tracking results. *GSDT* [54] proposes to use graph neural networks to extract object features in order to model object-object relation within the frame. This is done by creating a graph where the nodes are the features of the detected objects and of the tracklets and the edges are formed between every detection node within a pre-defined spatial window of a tracklet node. Node features are first obtained using *CenterNet* and are then iteratively updated using the features of its neighbors. Finally, they are used for data association in conjunction with learned identity embeddings.

Several models using transformers for feature extraction [12, 51, 52] have also been proposed. These methods have shown to be accurate, but demand a very high amount of computational

resources. *TransCenter* [12] follows [1, 25, 54] by detecting object from their center point using a learned generated heatmap and introduces a query learning network (QLN) that learns dense queries in order to allow the transformer architecture to produce dense representations. *MOTR* [52] extends *DETR* [56] to the MOT task by changing *DETR* fixed-length object queries set to a track queries set that is dynamically updated and that has a variable length. Furthermore, a Tracklet-Aware Label Assignment strategy is introduced to better perform data association. *DETR* bipartite matching strategy is kept for new tracks, but existing tracks are matched according to their track query.

4.4 Proposed Method

We propose a method based on CNNs, making it more efficient and tractable than with Transformers. It uses a simpler and more intuitive architecture than *CenterTrack* [1] by not relying on the previous generated heatmap $\hat{H}^{t-1}$ as input for the whole network to facilitate training. Furthermore, we hypothesize that using the top instead of the center point for detection increases accuracy in crowded scenes since the head is more often visible than the center of the body which leads to fewer missed detections. Our proposed model uses a CNN backbone and adds other prediction heads on top of it in order to perform object detection and multi-object tracking. We call our proposed method, *TopTrack*.

As shown in Figure 4.1, *TopTrack* only takes the current image $I^t \in \mathbb{R}^{H \times W \times 3}$ as input. The saved previous feature map $F^{t-1} \in \mathbb{R}^{H \times W \times 3}$ and previous heatmap $\hat{H}^{t-1} \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times C}$ are also used by some network heads. The image I^t is first sent to a CNN backbone in order to extract the feature map $F^t \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times 3}$, with $R = 4$ being a downsampling factor.

Then, the feature map F^t is used in order to predict the size of the bounding box $\hat{S}^t \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times 2}$ for each object and a quantization offset $\hat{O}^t \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times 2}$. F^t and F^{t-1} are used to generate a top point heatmap $\hat{H}^t \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times C}$ for each pre-defined object class C . Finally, F^t , F^{t-1} , $\hat{H}^t$ and $\hat{H}^{t-1}$ are used to predict the displacement $\hat{D}^t \in \mathbb{R}^{\frac{H}{R} \times \frac{W}{R} \times 2}$ between frames of each object. Each network head of *TopTrack* makes a prediction at every pixel of the feature map. The position of the objects is extracted by identifying the peaks in the generated top point heatmap (see Figure 4.2). Then, the correct predictions from the other heads are extracted using the predicted position of the top point of each object.

4.4.1 Top Heatmap

A first network head generates a top point heatmap $\hat{H}^t$ using the feature maps F^t and F^{t-1} where each element $\hat{h} = [0, 1] \in \mathbb{R}$ represents the probability of a given pixel being the top

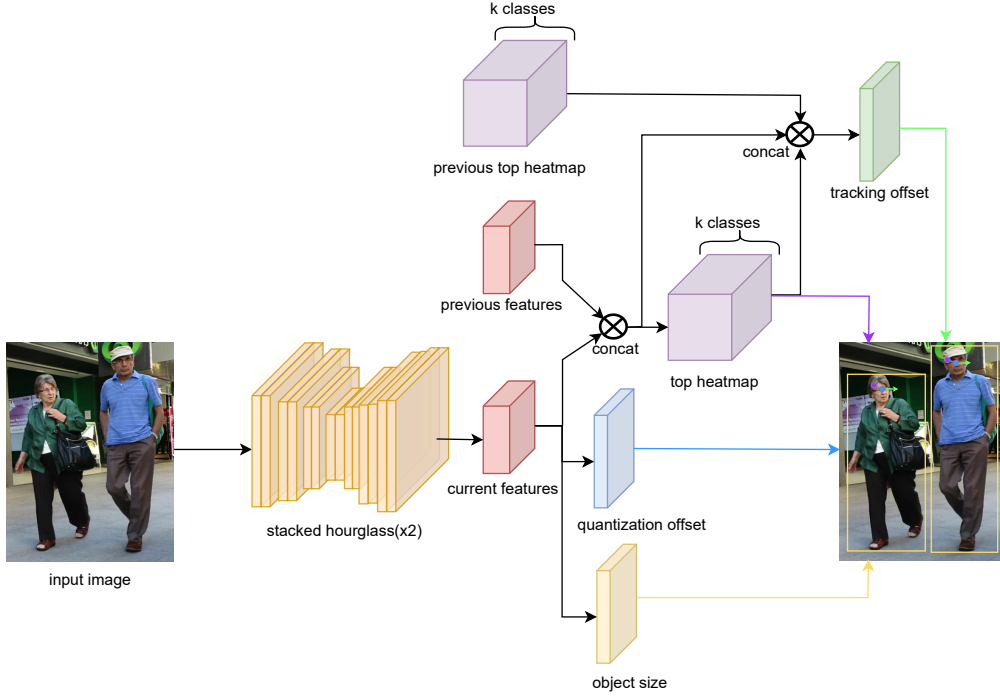


FIGURE 4.1 Overview of the architecture of TopTrack : An image at timestep t is fed to a Stacked Hourglass backbone to extract the feature map. This feature map is used to predict the size of the objects and the quantization offset. The feature map from the previous timestep $t - 1$ is saved and then used in conjunction with the current feature map to generate the top heatmaps. Finally, the current feature map, the current top heatmaps, the previous feature map and the previous top heatmaps are used to predict the tracking offset

point of an object at this position in F^t . Using F^{t-1} helps the detection capabilities of the network by providing additional context about the objects. Essentially, the heatmap is a collection of Gaussian distributions where the highest values (peaks) represent the position of the top of the objects. Extracting the location of every object simply requires taking the peak of each Gaussian distribution. In practice, an element is considered a peak if its value is greater or equal to its eight neighbors in order to account for overlapping objects. During inference, the top-100 peaks are extracted.

Since only the bounding box positions are given as annotations, we have to generate our own top position ground truths (GT) for training. According to our observations, the top point of an object is usually positioned at half the width and $\frac{1}{10}$ of the height of the bounding box. This is particularly true for humans, but this also captures the top area of other objects. Therefore, given a GT bounding box $b = (x_1, y_1, w, h)$, the ground truth top position is $p = (x, y)$ where $x = x_1 + \frac{w}{2}$ and $y = y_1 + \frac{h}{10}$. Taking the downsampling factor into account, the position of the top becomes $\tilde{p} = [\frac{p}{R}]$. The GT heatmap H is then created by turning the

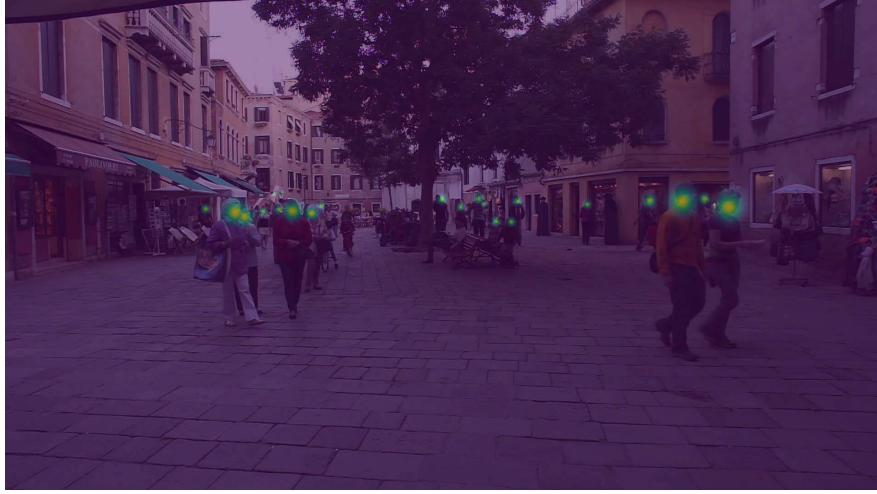


FIGURE 4.2 Visualization of how top heatmaps are used to extract object position. The likelihood of a certain pixel being the position of the top of an object is represented by a Gaussian function. The positions are extracted by identifying the peaks of each Gaussian function

GT top positions into Gaussian distributions using a Gaussian kernel. The value of the pixel H_{xy} at coordinates (x, y) in H is computed as $H_{xy} = \exp\left(-\frac{(x-\tilde{p}_x)^2 + (y-\tilde{p}_y)^2}{2\sigma_p^2}\right)$, where σ_p is the standard deviation [9]. A penalty-reduced pixel-wise logistic regression with focal loss [57] is used as the loss function of this network head [25]. Given a positive example ($H_{xy} = 1$), the corresponding loss is defined as

$$l = (1 - \hat{H}_{xy})^\alpha \log(\hat{H}_{xy}). \quad (4.1)$$

Otherwise, given a negative example, the corresponding loss is defined as

$$l = (1 - H_{xy})^\beta (\hat{H}_{xy})^\alpha \log(1 - \hat{H}_{xy}). \quad (4.2)$$

α and β are hyper-parameters of the focal loss, set to 2 and 4 respectively following [9, 25]. In (4.2), a penalty-reduction coefficient is added to the original focal loss equation so that positions near the GT are less penalized. The overall loss then becomes the sum of all examples normalized by the number of objects N and is given by

$$L_h = \frac{-1}{N} \sum_{xy} l. \quad (4.3)$$

4.4.2 Object Size

This network head uses $\hat{I}^t$ to predict the height and width of the bounding box $\hat{S}_i^t$ of every detected object i . It makes a prediction $\hat{b} = (\hat{b}_w, \hat{b}_h)$ at every pixel of $\hat{I}^t$, but only the pixels corresponding to the top of an object will be extracted using the generated heatmap $\hat{H}^t$. Given a predicted bounding box size $(\hat{b}_w, \hat{b}_h)$ and a top point (p_x, p_y) , the final bounding box coordinates can be computed as $b = (p_x - \frac{\hat{b}_w}{2}, p_y - \frac{\hat{b}_h}{10}, p_x + \frac{\hat{b}_w}{2}, p_y + \frac{9\hat{b}_h}{10})$. For training, the GT bounding boxes are provided under the format $b = (x_1, y_1, x_2, y_2)$. The GT size is then computed as $s = (x_2 - x_1, y_2 - y_1)$. The training is done using an L1 loss function conditioned on the size predictions extracted from the top point locations only. Let $\hat{S}_i$ be the predicted bounding box at the top position of object i , and s_i be the GT bounding box of object i , the loss is

$$L_{size} = \frac{1}{N} \sum_{k=1}^N |\hat{S}_i - s_i|. \quad (4.4)$$

4.4.3 Quantization offset

Downsampling the input image by a factor of R has the side effect of introducing a quantization error due to the discretisation of the top point position. As a result, it is possible for the top position of an object to be off by up to R^2 pixels, which can have a significant effect on both the detection and the data association tasks. In order to alleviate the quantization error, this network head predicts an offset $\hat{O}$ at each pixel of $\hat{I}^t$. Given the GT top point $p = (p_x, p_y)$ of an object downsampled by a factor of R and its discrete position $\tilde{p} = (\lfloor p_x \rfloor, \lfloor p_y \rfloor)$, the GT offset of object i is computed as $O_i = p - \tilde{p}$. The offset loss supervised at the detected top positions of each object i is

$$L_{off} = \frac{1}{N} \sum_{k=1}^N |\hat{O}_i - O_i|. \quad (4.5)$$

4.4.4 Displacement Offset

The model learns to predict the displacement $\hat{D}^t$ of each object i between the current frame I^t and the previous frame I^{t-1} . A prediction is made at each pixel of $\hat{I}^t$ and the correct prediction for each object is extracted using their top point position detected by the top heatmap $\hat{H}^t$. This predicted displacement can then be used for data association.

Given the GT position of object i in the current frame p_i^t and its GT position in the previous

frame p_i^{t-1} , the GT displacement is computed as $D_i^t = p_i^t - p_i^{t-1}$. The displacement loss is

$$L_D = \frac{1}{N} \sum_{k=1}^N |\hat{D}_i - D_i|. \quad (4.6)$$

4.4.5 Data Association

We used the same data association step as *CenterTrack*. It consists of a simple greedy association algorithm based on the distance $d = \hat{P}^{t-1} - \tilde{P}^{t-1}$ between the detected position of the top point $\hat{P}^{t-1} = \{p_i\}_{i=0}^{N-1}$ of each of the N objects in the previous frame I^{t-1} and the predicted position of the top point $\tilde{P}^{t-1} = \hat{P}^t - \hat{D}^t$ of each object based on their detected position $\hat{P}^t$ in the current frame and their predicted displacement $\hat{D}^t$. Detections in I^t are matched with the closest existing track. If a match was correctly made, the position of the track is updated with the position of the detection. Detections that remain unmatched are used to create new tracks and tracks that were not matched are deleted. This algorithm is local only, meaning that it does not allow for track rebirths or global association. Using the same association algorithm as *CenterTrack* ensures that the differences in our results are solely due to the changes in the architecture of the network and the usage of the top instead of the center point for object detection.

4.5 Training

TopTrack is end-to-end trainable by combining the losses of each network head into a global training objective. A scaling constant λ is used to adjust the weight on the overall training objective if needed. The training objective of the network becomes

$$L = L_h + \lambda_{size} L_{size} + L_{off} + L_D, \quad (4.7)$$

where the scaling constant λ_{size} is set to 0.1 for all experiments.

In order for the model to be trained properly, strong data augmentation techniques have to be used. Else, the heatmaps generated by the model, which are essential for successful tracking, are subpar. Also, if a video has a high frame rate, the object displacement between frames is very small, so the network could learn to repeat its previous predictions [1]. These issues are solved by using different data augmentation techniques and by adding common errors made at test time, which includes adding random noise in the GT heatmaps using a Gaussian distribution, removing GT detections to create false negatives and adding false detections to create false positives, flipping the input image horizontally, cropping the input image and

modifying its color. All these data augmentation techniques are applied randomly during the training process. Additionally, the previous heatmap H^{t-1} is provided to the model and it is modified using the same techniques as the input image I^t for consistency. Furthermore, during training, the exact previous image I^{t-1} is not always used. Indeed, it is possible to use another image temporally close to the current frame that is not the immediate previous image. I^{t-1} is chosen randomly from an interval of frames $[I^{t-k}, I^{t+k}]$, where $0 < k \leq 3$. This helps to alleviate overfitting to video framerate and unbalanced movement direction in the dataset. To train our model properly using this technique, the features of the previous image $\hat{I}^{t-1}$ have to be extracted alongside those of the current image instead of being able to save them at each timestep as in inference, because the same data augmentation has to be applied to both.

Following *CenterTrack* [1], *TopTrack* can be trained on static images. Although the previous frame I^{t-1} does not exist in this case, one can be simulated and used for training by randomly scaling and translating the current frame I^t .

4.6 Experiments

We used three datasets to train *TopTrack* : MOT17 [2], MOT20 [34] and CrowdHuman [58] and evaluated it on MOT17 and MOT20. These datasets do not provide official validation sets so the training sets were split in half in order to allow hyperparameter tuning and to perform the ablation study of section 4.6.3. The final results were obtained by training *TopTrack* on the whole training set of either MOT17 or MOT20 and submitting the results to the respective test server of each dataset for evaluation.

The MOT17 challenge dataset contains 14 video sequences, 7 for training and 7 for testing. Many considerations were taken when crafting this dataset in order to better evaluate MOT models. Firstly, in some sequences the camera is moving and in others it is static. It can also be positioned at a high, medium or low height to have different viewpoints. Moreover, sequences are filmed in different weather to account for different illumination conditions (sunny, cloudy, night). Finally, the crowd density and number of occlusions are much higher than in MOT15 [59]. MOT17 is an extension of MOT16 that provides public detections with more detectors and more precise annotations.

The MOT20 challenge is a more recent dataset created to tackle scenes that have denser crowds compared to its predecessor. It contains 4 training sequences and 4 test sequences, taken from 3 very crowded scenes that can go up to a mean density of 246 pedestrians per frame. The test set contains two sequences taken from the same scenes as the sequences from

the training set and two sequences that are completely new to the model. This dataset includes indoor and outdoor sequences, as well as daytime and nighttime settings. All sequences are filmed from an elevated point-of-view.

The CrowdHuman dataset is an object detection dataset that is used to pre-train *TopTrack*. It is a human detection benchmark that is characterized by its crowded scenes and high number of occlusions. Using strong data augmentation allows for the training of a MOT model even though it is a single image object detection dataset as described at the end of section 4.5.

Our model is evaluated using all the standard MOT metrics, such as the CLEAR MOT metrics (MOTA, MOTP, MT, etc) [31], IDF1 [32] and HOTA [15].

4.6.1 Implementation Details

TopTrack uses the *Stacked Hourglass* [60] backbone for feature extraction. Each hourglass module consists of 5 down and up-convolutional neural networks with skip connections [25]. Each network head consists of a 3×3 convolutional layer followed by a 1×1 convolutional layer to obtain the final prediction. All training was done using the Adam optimizer. The training was performed on a single RTX3090 GPU and an Intel Core i7-10700 CPU. Our model was first pretrained on the CrowdHuman [58] dataset for 140 epochs with a batch size of 8 and a learning rate of 2.5^{-4} . The starting learning rate is reduced by a factor of 10 after 90 epochs and once more after 120 epochs. We then further train the model on either the MOT17 dataset or MOT20 dataset to fine-tune it to the each respective benchmark. The MOT17 model was trained for an additional 70 epochs with a batch size of 32 and a learning rate of 1.25^{-4} reduced by a factor of 10 after 60 epochs. The MOT20 model was trained for an additional 90 epochs with a batch size of 32 and a learning rate of 2.5^{-4} reduced by a factor of 10 after 60 epochs.

4.6.2 Results

As we can see from table 4.1, our overall results are slightly lower than *CenterTrack* and other more recent trackers under the joint-tracking-and-detection paradigm. However, we did manage to significantly reduce the number of False Negative errors (missed detections), which supports the motivation of using object top information for better detection. Also, we managed to achieve the best score for the ML metric and a very high MT metric, meaning that our tracker manages to accurately track most trajectories for a significant amount of time and misses very few trajectories. Moreover, these trajectories that other trackers did not

TABLEAU 4.1 Results of state-of-the-art methods on MOT17 dataset with private detections. Best results are in **bold**.

Model	MOTA↑	HOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
CenterTrack [1]	67.8	52.2	64.7	34.6	24.6	18 498	160 332	3039
CTracker [53]	66.6	49.0	57.4	32.2	24.2	22 284	160 491	5529
FairMOT [10]	73.7	59.3	72.3	43.2	17.3	27 507	117 477	3303
GSDT_v2 [54]	73.2	55.2	66.5	41.7	17.5	26 397	120 666	3891
MOTR [52]	78.6	62.0	75.0	50.3	13.1	23 409	94 797	2619
SCSAN [61]	66.9	-	68.3	35.7	21.5	23 587	193 286	1868
SST [49]	52.4	39.3	49.5	18.3	34.0	25 423	234 592	8431
TransCenter [12]	76.4	-	65.4	51.7	11.6	37 005	89 712	6402
TubeTK [55]	63.0	48.0	58.6	31.2	19.9	27 060	177 483	4137
Ours	64.8	47.9	58.2	38.7	11.5	52 425	136 221	10 083

TABLEAU 4.2 Results of state-of-the-art methods on MOT20 dataset with private detections. Best results are in **bold**.

Model	MOTA↑	HOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
FairMOT [10]	61.8	54.6	67.3	68.8	7.6	103 440	88 901	5243
GSDT_v2 [54]	67.1	53.6	67.5	53.1	13.2	31 507	135 935	9878
MLT [62]	48.9	43.2	54.6	30.9	22.1	45 660	216 803	2187
TMM [63]	43.3	36.2	45.2	17.6	26.3	27 953	262 406	2965
TransCenter [12]	72.9	-	57.7	66.5	11.8	28 596	108 982	2625
Ours	46.3	26.8	27.6	20.0	22.0	28 363	226 089	23 227

manage to track are often the most difficult examples, which goes to show that there is value in using the top of the objects as a keypoint for detection. As it can be seen in figure 4.3 and 4.4, our trackers manages to detect and track many objects that other trackers cannot, like objects that are small and far away, that are partially occluded or that are sitting, which makes them a lot harder to detect.

Looking at the performance of *TopTrack* on the test sequences, a high amount of FP can be attributed to human-like objects. Figures 4.5, 4.6 and 4.7 show how our tracker detects shop mannequins, reflections from the crowd in windows and cartoon characters on advertisements. Furthermore, objects that are not evaluated by the benchmarks like seated people and cyclists are also considered as FP errors even though it is reasonable for our model to detect them. We are confident that many of these errors could be alleviated with better training using



FIGURE 4.3 Comparison between TopTrack and CenterTrack on two frames of sequence 8 of MOT17 test set. Arrows show objects detected by TopTrack but missed by CenterTrack

more data as described in [10] for example. The best models also all use many more datasets to train their model. As for the high amount of IDSW, this can mostly be attributed to the higher amount of detections from our tracker and the usage of a very simple data association strategy. Having more detections means there are more possibilities to make errors, especially for harder examples. Furthermore, like for *CenterTrack*, our data association strategy is local only and does not use appearance information and does not allow for track rebirths. We think that adopting a more robust association scheme would greatly reduce the amount of IDSW, which would improve the overall score of our tracker. These factors are even more relevant when looking at table 4.2. Indeed, the sequences of the MOT20 challenge have very dense crowds meaning it is very hard for a local data association scheme to be accurate.

4.6.3 Ablation Study

The most important aspect of our method to evaluate is the effect of using the top keypoint for detection instead of the center. To do so, we trained a model using each keypoint and present the results in Table 4.3. Using the top keypoint reduces the amount of FP, FN and increases IDF1, which results in a better MOTA score.

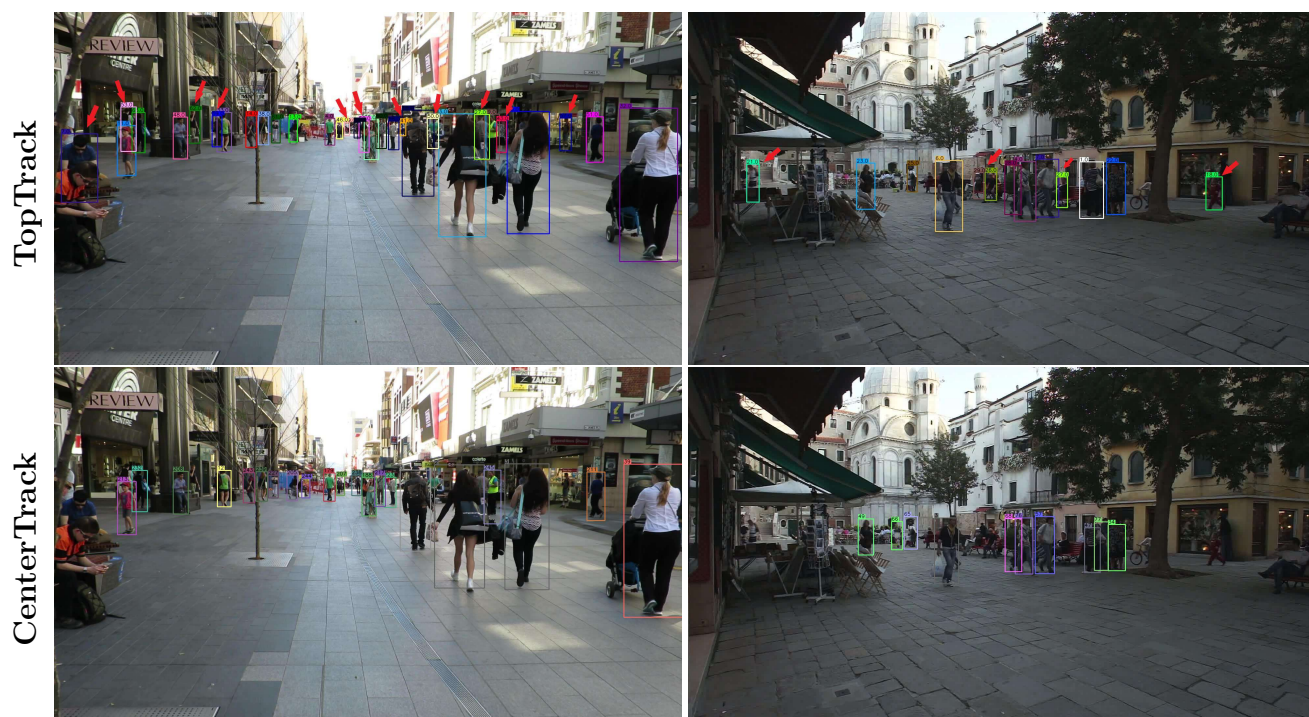


FIGURE 4.4 Comparison between TopTrack and CenterTrack on one frame of sequence 7 and one frame of sequence 12 of MOT17 test set. Arrows show objects detected by TopTrack but missed by CenterTrack

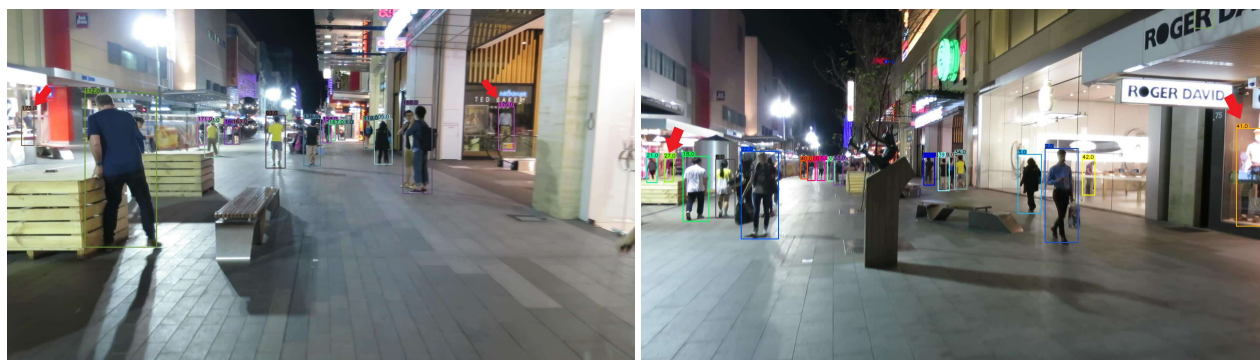


FIGURE 4.5 Examples of shop mannequins being detected as objects by TopTrack. See ID 173 and 175 (left) and ID 27 and 41 (right)

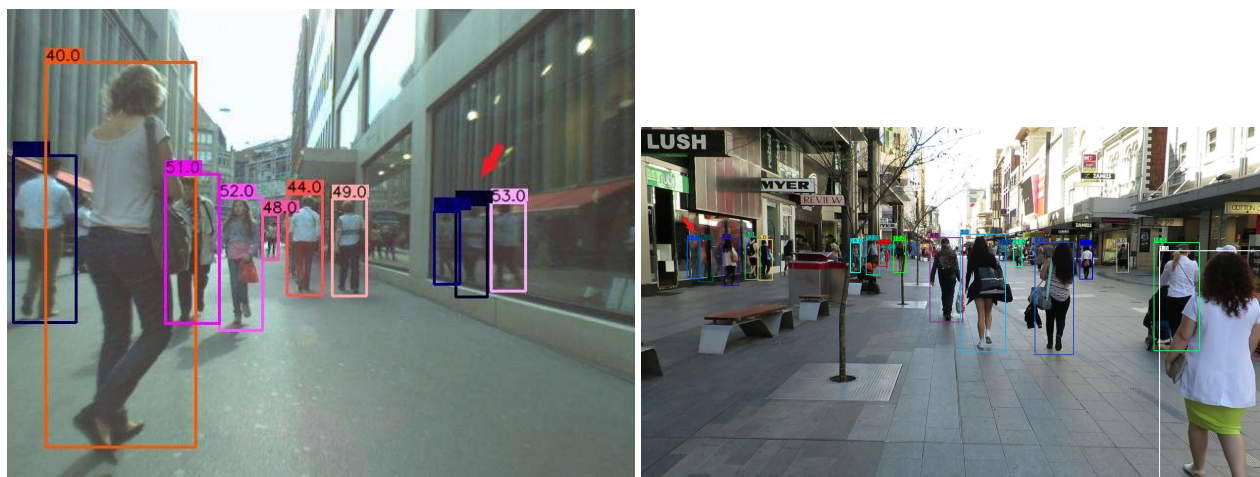


FIGURE 4.6 Examples of reflections being detected as objects by TopTrack. See ID 56, 63 and 64 (left) and ID 16 and 22 (right)



FIGURE 4.7 Examples of a cartoon character being detected as an object by TopTrack (See ID 17)

TABLEAU 4.3 Results of TopTrack on the MOT17 validation set using keypoint for detection. Best results are in bold

Keypoint	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
Center	58.0	60.2	29.8	21.5	5.9%	35.1%	0.9%
Top	60.7	61.5	30.1	21.8	4.7%	33.7%	0.9%

For data association, a common simple association scheme is the Hungarian algorithm. Table 4.4 shows how using it affects the results when compared to the greedy algorithm that we used. The main difference is a significant increase in IDSW when using the Hungarian algorithm and also a slight increase in FP and a slight decrease in FN. Overall, using the greedy algorithm gives better results.

TABLEAU 4.4 Results of TopTrack on the MOT17 validation set using different matching algorithms. Best results are in bold

Algorithm	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
Greedy	60.7	61.5	30.1	21.8	4.7%	33.7%	0.9%
Hungarian	60.5	53.5	32.4	20.9	5.3%	33.0%	1.2%

Finally, we tested our method using different training seeds. Seed 317 is the default used in *CenterNet* [25] and *CornerNet* [9]. The other seeds were chosen randomly. As it can be seen in Table 4.5, the default seed is the best one. Most other seeds however did not result in a significant decrease in accuracy except for seed 5. The model using this seed had a significant amount of FN more than the others. Globally, results are stable and not too sensitive to the choice of the seed.

TABLEAU 4.5 Results of TopTrack on the MOT17 validation set using different seeds for training. Best results are in bold

Seed	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
317	60.7	61.5	30.1	21.8	4.7%	33.7%	0.9%
310	60.2	60.7	28.0	27.7	4.5%	34.3%	0.9%
17	60.3	61.3	28.7	27.4	4.7%	33.9%	0.9%
5	58.0	59.4	27.1	26.8	4.8%	36.4%	0.8%
142	60.6	61.8	31.0	24.2	4.4%	34.2%	0.9%

4.7 Conclusion

In this work, we introduced an end-to-end trainable joint-detection-and-tracking model that uses the top of the object as a keypoint for detection and tracking instead of the more commonly used center point. We show how using this keypoint allows our tracker to better detect hard-to-track objects and we also show how using the generated heatmap of the current frame allows the tracker to better connect objects locally, which leads to a reduction in FN errors. Our model achieves good accuracy and competitive results on two popular MOT benchmarks.

Statements and Declarations

Funding This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), [funding reference number DGDND-2020-04633].

Conflict of interest The authors have no conflict of interests to disclose.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication All authors consent to the publication of this article.

Availability of data and materials All the datasets that were used for this work are freely available online on their respective official website.

Code availability The code used for this work is freely available online at the following URL : <https://github.com/TopTrack/TopTrack2023>

Authors' contributions Jacob Meilleur conducted the research project and wrote the first draft of the manuscript. Guillaume-Alexandre Bilodeau supervised the research project, revised the manuscript and secured the funding needed to conduct this work.

CHAPITRE 5 DISCUSSION GÉNÉRALE

Les résultats dans l'article sont présentés de façon globale, mais il est aussi pertinent de comparer les modèles selon les séquences de test de la base de données. En effet, chaque séquence a des caractéristiques différentes, ce qui affectent les performances des modèles différemment.

TABLEAU 5.1 Comparaison de TopTrack et CenterTrack [1] par séquence de l'ensemble de test de MOT17 [2]. Les meilleurs résultats sont en gras

CenterTrack	MOTA↑	HOTA↑	IDF1↑	FP↓	FN↓	IDSW↓
Séquence 01	56.7	40.6	48.7	270	2484	40
Séquence 03	85.2	61.4	78.2	2616	12 701	169
Séquence 06	62.2	40.1	47.2	515	3765	169
Séquence 07	52.0	37.4	45.0	985	6942	185
Séquence 08	30.3	32.1	33.4	139	14 488	92
Séquence 12	49.8	48.3	58.4	501	3809	42
Séquence 14	42.0	32.9	42.0	1140	9255	316
TopTrack	MOTA↑	HOTA↑	IDF1↑	FP↓	FN↓	IDSW↓
Séquence 01	37.7	32.1	35.2	1414	2530	76
Séquence 03	82.1	56.5	70.6	5688	12 682	343
Séquence 06	53.5	40.3	50.1	1514	3498	469
Séquence 07	52.4	35.0	41.7	2072	5598	380
Séquence 08	37.4	29.9	33.0	1577	11 368	290
Séquence 12	45.4	45.4	54.4	1531	3130	71
Séquence 14	35.0	31.5	38.8	3679	6601	1732

Comme montré par le Tableau 5.1, notre méthode réussit à réduire considérablement le nombre de FN pour presque toutes les séquences, au coût d'une augmentation des FP et des IDSW. Notre méthode a un très gros effet sur les séquences 07 et 08, ce qui nous permet d'atteindre un score de MOTA plus haut que *CenterTrack*. Ces séquences correspondent à des scènes urbaines d'une rue piétonne commerciale avec une caméra positionnée au niveau de la tête. Dans ce contexte, notre méthode est capable de faire usage de ces meilleures capacités de détection afin d'atteindre des meilleurs résultats. Les séquences 01 et 03 sont deux séquences où notre méthodes ne réussit pas à réduire significativement le nombre de FN. D'abord, la séquence 01 montre une plaza avec des piétons et des gens assis au milieu. Étant donné que la densité de cette scène est beaucoup plus petite que celle des scènes 07 et 08, il n'y a pas d'effet significatif sur le nombre de FN. De plus, notre méthode réussit à détecter les

personnes assises, mais ces personnes sont considérées comme des FP lors de l'évaluation de *MOT17*, ce qui affecte négativement les résultats de notre méthode, même si la détection de ces personnes est importante pour certaines applications du suivi multiobjet. Puis, la séquence 03 montre une rue piétonne très achalandée, cependant la caméra est positionnée haut dans les airs, alors il n'y a pas autant d'occlusions partielles où le centre est caché mais pas le haut que dans les séquences 07 et 08.

Certaines expériences que nous avons réalisées n'étaient pas nécessairement assez pertinentes pour les inclure dans l'article, mais elles peuvent quand même apporter des réflexions additionnelles importantes pour ce travail. D'abord, il est pertinent de vérifier comment le pré-entraînement du modèle affecte les résultats finaux. Le Tableau 5.2 présente les résultats obtenus par notre méthode sur l'ensemble de validation de la base de données *MOT17* [2] selon les bases de données utilisées pour l'entraîner. Nous avons créé l'ensemble de validation nous-même en séparant les données d'entraînement en deux moitiés puisque cette base de données n'inclut qu'un ensemble de données d'entraînement et un ensemble de données de test. Seul les annotations de l'ensemble d'entraînement sont fournies publiquement. On peut remarquer l'importance de pré-entraîner *TopTrack* sur la base de données *CrowdHuman* [58] afin d'obtenir de bons résultats. En effet, *CrowdHuman* contient 15000 images et 339565 objets uniques à détecter, ce qui est près de trois fois plus que *MOT17*. Ainsi, notre méthode est en mesure de mieux apprendre à extraire des caractéristiques robustes sur les objets qui doivent être suivis. De plus, ces résultats montrent l'efficacité de la méthode d'entraînement sur les images statiques détaillée dans l'article.

TABLEAU 5.2 Résultats de TopTrack sur l'ensemble de validation de MOT17 selon les bases de données utilisées pour l'entraînement. "CH" signifie l'utilisation de la base de données *CrowdHuman*. Les meilleurs résultats sont en gras

Base de données	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
CH	54.6	76.7	56.0	33.3	21.8	4709	19 152	603
MOT17	47.4	70.7	50.9	19.8	38.3	2152	25 700	510
CH+MOT17	60.7	79.2	61.5	30.1	21.8	2526	18 180	474

Nous avons aussi réalisé des expérimentations préliminaires sur la base de données *KITTI Tracking* [3] qui évalue le suivi multiobjet sur les piétons et les voitures. L'ensemble d'entraînement de cette base de données a aussi été divisé en deux moitiés afin de créer un ensemble de validation. Les résultats présentés dans les Tableaux 5.3 et 5.4 proviennent de cet ensemble de validation et non du serveur de test de *KITTI*. Aucun modèle n'a été testé sur le serveur de test puisque les autres modèles à l'état de l'art sont pré-entraînés sur des bases de données

de très grande taille, comme *NuScenes* [64], alors le temps d’entraînement était beaucoup trop long pour nous. Malgré tout, il est intéressant d’analyser les résultats de notre modèle sur une base de données qui contient plusieurs types d’objets. Le Tableau 5.3 présente les résultats de *TopTrack* et de *CenterTrack* sur les piétons. Malgré que ce soit des résultats préliminaires, nous pouvons voir que *TopTrack* obtient des résultats significativement meilleurs que *CenterTrack* pour toutes les métriques étudiées, mis à part les IDSW.

TABLEAU 5.3 Résultats de TopTrack sur l’ensemble de validation des piétons de KITTI Tracking [3]

Model	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
CenterTrack [1]	61.9	74.7	78.9	39.3	13.1	321	1375	14
TopTrack	65.8	77.4	82.0	45.2	11.9	224	1230	79

Le Tableau 5.4 présente les résultats de *TopTrack* et de *CenterTrack* sur les voitures. Pour cette classe d’objet, les deux modèles obtiennent des résultats similaires, avec *TopTrack* ayant moins de fausses détections, mais plus de IDSW. Cela signifie qu’utiliser le haut des objets comme point-clé fonctionne aussi bien que le centre dans les cas où les occlusions partielles sont plus rares. En effet, comme les séquences utilisées par la base de données *KITTI Tracking* sont capturées dans un contexte urbain, on ne retrouve pas les même occlusions partielles qui surviennent lors du suivi de piétons dans une foule dense où seul le haut des objets est visible. Les objets sont donc généralement complètement visibles ou complètement cachés, alors la position du point-clé utilisée pour la détection n’a pas un grand effet sur les résultats. Ce résultat est important puisque c’est une pièce d’évidence qu’utiliser le haut des objets n’est pas nuisible au suivi multiobjet de différents types d’objets même si la méthode est plus orientée vers le suivi des piétons. Nous pensons donc qu’utiliser le haut des objets comme point-clé fonctionne généralement au moins aussi bien que le centre sur des objets autres que les humains et peut mieux fonctionner que le centre dans les cas où il y a beaucoup d’occlusions partielles, mais que le haut des objets est visible.

TABLEAU 5.4 Résultats de TopTrack sur l’ensemble de validation des voitures de KITTI Tracking [3]

Model	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
CenterTrack [1]	84.5	83.2	93.4	83.5	2.8	626	1040	37
TopTrack	84.4	85.2	93.7	78.4	1.1	368	1134	210



FIGURE 5.1 Point-clés des piétons de la scène 02 de l’ensemble d’entraînement de MOT17 [2]

Le choix de positionner le haut des objets à $\frac{1}{10}$ du haut de leur boîte englobante a été fait en se basant sur des résultats quantitatifs et qualitatifs. D’abord, plusieurs hauteurs pour la position du point-clé avaient été choisies et les meilleurs résultats avaient été obtenus en utilisant une hauteur de $\frac{1}{10}$. Ensuite, des tests visuels ont été effectués dans différents scénarios afin de vérifier la robustesse de ce choix. Les Figures 5.1, 5.2 et 5.3 présentent ces résultats qualitatifs. Les Figures 5.1 et 5.2 montrent des piétons dans une aire publique. On peut y voir que tous les point-clés sont visibles, malgré certaines occlusions partielles à cause du grand nombre de personnes dans les scènes. Cependant, le point-clé de certains objets serait caché s’il était positionné au centre des objets au lieu de leur haut. La Figure 5.3 montre des piétons et des voitures dans un contexte urbain. Encore une fois, on peut voir que le haut des objets est visible même si les objets sont loin de la caméra et même lorsqu’il y a des occlusions partielles qui cachent le centre des objets.

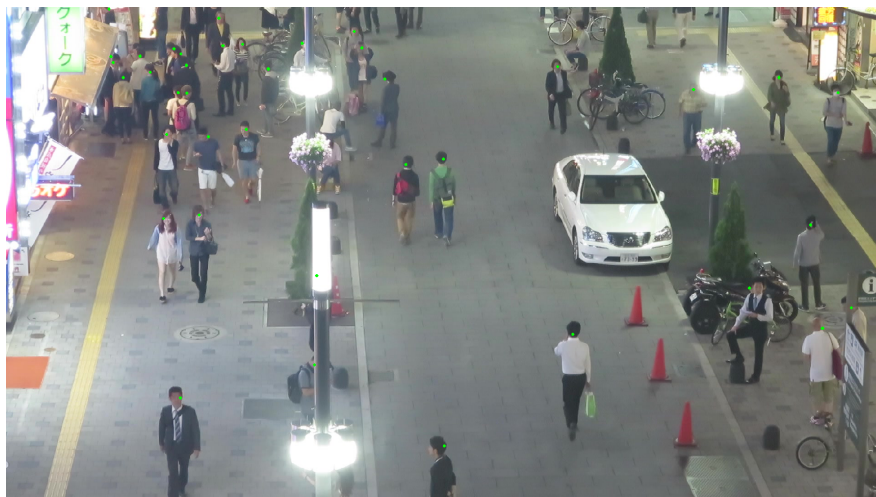


FIGURE 5.2 Point-clés des piétons de la scène 04 de l'ensemble d'entraînement de MOT17 [2]



FIGURE 5.3 Point-clés des piétons et des voitures de la scène 13 de l'ensemble d'entraînement de MOT17 [2]

CHAPITRE 6 CONCLUSION

Afin de conclure ce mémoire, ce chapitre inclut une synthèse des principaux travaux effectués au cours de ce projet de recherche, suivi d’une explication des principales limitations de la méthode proposée au chapitre 4 et se termine en proposant plusieurs améliorations qui pourraient être apportées dans le futur afin de l’améliorer.

6.1 Synthèse des travaux

Ce travail nous a permis de proposer une nouvelle méthode de suivi multiobjet afin d’essayer d’améliorer les méthodes existantes. Parmi les séquences vidéo utilisées pour mesurer les résultats des méthodes de suivi multiobjet, il est commun d’avoir des occlusions partielles et complètes. Les occlusions représentent un problème majeur puisqu’il devient beaucoup plus difficile de détecter et suivre un objet lorsqu’il est caché. La méthode proposée, *TopTrack*, réussit à suivre des objets qui sont manquées par d’autres méthodes en utilisant le haut des objets comme point-clé pour la détection. Nos expériences montrent qu’utiliser ce point-clé permet à notre méthode de continuer à suivre des objets qui sont partiellement cachés, en autant que leur haut soit toujours visible. Notre méthode envoie d’abord la trame courante à un extracteur de caractéristiques afin de créer une carte d’attributs. Cette carte d’attributs est ensuite utilisée par les têtes de prédiction du réseau afin de prédire une carte de chaleur, la taille des boîtes englobantes des objets, un décalage dû à la quantification de la position des objets et le déplacement des objets d’une trame à l’autre. La tête de prédiction qui génère la carte de chaleur reçoit aussi en entrée la carte d’attributs de la trame précédente. La carte de chaleur générée est composée de plusieurs fonctions gaussiennes afin d’effectuer la détection des objets. Chaque fonction gaussienne modélise la probabilité que le haut d’un objet se trouve à cette position. Alors que plusieurs méthodes emploient un modèle statistique pour prédire le mouvement des objets, nous utilisons plutôt une tête de prédiction faisant usage de l’apprentissage automatique afin de pouvoir mieux prédire les mouvements brusques et pour prendre en compte que les objets ne bougent pas nécessairement de façon uniforme tout au long de la séquence vidéo. Cette tête de prédiction reçoit en entrée la carte d’attributs de la trame courante et précédente, ainsi que la carte de chaleur de la trame courante et précédente. Notre méthode a été comparée avec les méthodes de suivi multiobjet à l’état de l’art sur deux bases de données et démontrent les bienfaits liés à l’utilisation du haut des objets pour la détection.

6.2 Limitations de la solution proposée

La plus grande limitation de notre méthode est la stratégie d’association temporelle que nous utilisons. Les détections sont associées à partir de la prédiction faite sur leur mouvement en utilisant un algorithme glouton, alors les cas plus complexes sont trop difficiles à associer pour cet algorithme et causent des erreurs d’association. Un exemple de cas complexe est lorsqu’un objet a grandement bougé alors qu’il était complètement caché. Un tel cas requiert d’utiliser l’apparence de l’objet pour l’associer correctement à son ancien *tracklet*. Une autre limitation est le manque de ressources de calcul. Effectivement, la tâche de suivi multiobjet est très complexe puisqu’il faut être en mesure de suivre les objets dans une multitude de scénarios afin d’obtenir des résultats à l’état de l’art. Par exemple, certaines séquences sont capturées par une caméra stationnaire et d’autres par une caméra en mouvement, certaines séquences sont capturées durant le jour et d’autres la nuit. De plus, les séquences incluent des objets avec des tailles et des apparences grandement différentes, alors la méthode doit être robuste à cette grande variabilité. Pour ce faire, il faut un modèle assez gros et il doit être entraîné sur une grande quantité de données ce qui augmentent significativement le temps d’entraînement. Il devient donc plus difficile d’essayer différentes configurations et d’affiner les hyperparamètres. Enfin, une limitation commune aux méthodes utilisant l’apprentissage automatique est l’explicabilité du modèle. Les bases de données et les métriques développées permettent d’évaluer le fonctionnement du modèle à haut niveau, cependant il est très difficile, voire impossible dans certains cas, de savoir avec certitude pourquoi une erreur a été commise à un endroit spécifique. Parfois un objet n’est pas détecté ou il se fait associé à un autre *tracklet* et il n’y a aucun moyen de savoir pourquoi. En conséquence, il est difficile de régler ces erreurs puisqu’on ne sait pas quels en sont les causes.

6.3 Améliorations futures

Plusieurs améliorations pourraient être apportées à la méthode proposée afin d’atteindre des résultats meilleurs. D’abord, l’amélioration future qui aurait le plus d’impact sur les résultats de *TopTrack* est d’utiliser une méthode d’association temporelle plus complexe afin de diminuer le nombre d’erreurs d’association. En effet, faire l’association temporelle en utilisant que les prédictions du déplacement des objets n’est pas suffisant dans les cas plus difficiles, alors utiliser un algorithme qui prend aussi en compte l’apparence des objets serait bénéfique. Additionnellement, incorporer un module qui performe l’association temporelle à long terme permettrait de récupérer des *tracklets* qui ont été perdus pendant un certain temps.

Puis, une deuxième amélioration possible serait d'utiliser plus de données d'entraînement. En effet, la majorité des autres méthodes à l'état de l'art utilisent beaucoup plus de données d'entraînement, allant parfois jusqu'à combiner six bases de données. Cela augmenterait significativement le temps d'entraînement du modèle, mais permettrait d'atteindre des meilleurs résultats.

Enfin, une troisième amélioration possible serait de changer l'extracteur de caractéristiques. Effectivement, l'extracteur de caractéristiques peut jouer un rôle significatif sur la performance d'un modèle, alors en essayer d'autres pourrait améliorer notre méthode.

RÉFÉRENCES

- [1] X. Zhou, V. Koltun et P. Krähenbühl, “Tracking objects as points,” dans *Computer Vision – ECCV 2020*, A. Vedaldi *et al.*, édit. Cham : Springer International Publishing, 2020, p. 474–490.
- [2] A. Milan *et al.*, “MOT16 : A benchmark for multi-object tracking,” *arXiv :1603.00831 [cs]*, mars 2016, arXiv : 1603.00831. [En ligne]. Disponible : <http://arxiv.org/abs/1603.00831>
- [3] A. Geiger, P. Lenz et R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” dans *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, p. 3354–3361.
- [4] J. Uijlings *et al.*, “Selective search for object recognition,” *International Journal of Computer Vision*, vol. 104, p. 154–171, 09 2013.
- [5] S. Ren *et al.*, “Faster r-cnn : Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, n° 6, p. 1137–1149, 2017.
- [6] R. Girshick *et al.*, “Rich feature hierarchies for accurate object detection and semantic segmentation,” dans *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, p. 580–587.
- [7] J. Redmon *et al.*, “You only look once : Unified, real-time object detection,” 2015. [En ligne]. Disponible : <https://arxiv.org/abs/1506.02640>
- [8] W. Liu *et al.*, “SSD : Single shot MultiBox detector,” dans *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, p. 21–37. [En ligne]. Disponible : https://doi.org/10.1007%2F978-3-319-46448-0_2
- [9] H. Law et J. Deng, “Cornersnet : Detecting objects as paired keypoints,” *International Journal of Computer Vision*, vol. 128, n°. 3, p. 642–656, Mar 2020. [En ligne]. Disponible : <https://doi.org/10.1007/s11263-019-01204-1>
- [10] Y. Zhang *et al.*, “FairMOT : On the fairness of detection and re-identification in multiple object tracking,” *International Journal of Computer Vision*, vol. 129, n°. 11, p. 3069–3087, sep 2021. [En ligne]. Disponible : <https://doi.org/10.1007/s11263-021-01513-4>
- [11] T. Meinhardt *et al.*, “Trackformer : Multi-object tracking with transformers,” dans *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, p. 8834–8844.

- [12] Y. Xu *et al.*, “Transcenter : Transformers with dense representations for multiple-object tracking,” *IEEE transactions on pattern analysis and machine intelligence*, 2021.
- [13] V. Dumoulin et F. Visin, “A guide to convolution arithmetic for deep learning,” 2018. [En ligne]. Disponible : <https://arxiv.org/abs/1603.07285>
- [14] Y. Jia, “Attention mechanism in machine translation,” *Journal of Physics : Conference Series*, vol. 1314, n^o. 1, p. 012186, oct 2019. [En ligne]. Disponible : <https://dx.doi.org/10.1088/1742-6596/1314/1/012186>
- [15] J. Luiten *et al.*, “HOTA : A higher order metric for evaluating multi-object tracking,” *International Journal of Computer Vision*, vol. 129, n^o. 2, p. 548–578, oct 2020. [En ligne]. Disponible : <https://doi.org/10.1007%2Fs11263-020-01375-2>
- [16] N. Bodla *et al.*, “Soft-nms — improving object detection with one line of code,” *2017 IEEE International Conference on Computer Vision (ICCV)*, p. 5562–5570, 2017.
- [17] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics (NRL)*, vol. 52, 1955.
- [18] R. Girshick, “Fast r-cnn,” dans *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, p. 1440–1448.
- [19] W. Burger et M. J. Burge, *Scale-Invariant Feature Transform (SIFT)*. London : Springer London, 2016, p. 609–664. [En ligne]. Disponible : https://doi.org/10.1007/978-1-4471-6684-9_25
- [20] N. Dalal et B. Triggs, “Histograms of oriented gradients for human detection,” dans *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, 2005, p. 886–893 vol. 1.
- [21] C. Cortes et V. N. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, p. 273–297, 1995.
- [22] J. Redmon et A. Farhadi, “Yolo9000 : Better, faster, stronger,” 2016. [En ligne]. Disponible : <https://arxiv.org/abs/1612.08242>
- [23] Z. Ge *et al.*, “Yolox : Exceeding yolo series in 2021,” 2021. [En ligne]. Disponible : <https://arxiv.org/abs/2107.08430>
- [24] K. Simonyan et A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014. [En ligne]. Disponible : <https://arxiv.org/abs/1409.1556>
- [25] X. Zhou, D. Wang et P. Krähenbühl, “Objects as points,” 2019. [En ligne]. Disponible : <https://arxiv.org/abs/1904.07850>
- [26] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME—Journal of Basic Engineering*, vol. 82, n^o. Series D, p. 35–45, 1960.

- [27] A. Bewley *et al.*, “Simple online and realtime tracking,” dans *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, p. 3464–3468.
- [28] N. Wojke, A. Bewley et D. Paulus, “Simple online and realtime tracking with a deep association metric,” dans *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, p. 3645–3649.
- [29] Y. Zhang *et al.*, “Bytetrack : Multi-object tracking by associating every detection box,” dans *Computer Vision – ECCV 2022*, S. Avidan *et al.*, édit. Cham : Springer Nature Switzerland, 2022, p. 1–21.
- [30] A. Dosovitskiy *et al.*, “An image is worth 16x16 words : Transformers for image recognition at scale,” 2020. [En ligne]. Disponible : <https://arxiv.org/abs/2010.11929>
- [31] K. Bernardin et R. Stiefelhagen, “Evaluating multiple object tracking performance : The clear mot metrics,” *EURASIP Journal on Image and Video Processing*, vol. 2008, n°. 1, p. 246309, May 2008. [En ligne]. Disponible : <https://doi.org/10.1155/2008/246309>
- [32] E. Ristani *et al.*, “Performance measures and a data set for multi-target, multi-camera tracking,” dans *Computer Vision – ECCV 2016 Workshops*, G. Hua et H. Jégou, édit. Cham : Springer International Publishing, 2016, p. 17–35.
- [33] B. D. Lucas et T. Kanade, “An iterative image registration technique with an application to stereo vision,” dans *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, ser. IJCAI’81. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1981, p. 674–679.
- [34] P. Dendorfer *et al.*, “Mot20 : A benchmark for multi object tracking in crowded scenes,” 2020. [En ligne]. Disponible : <https://arxiv.org/abs/2003.09003>
- [35] F. Yang *et al.*, “Hard to track objects with irregular motions and similar appearances? make it easier by buffering the matching space,” dans *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, January 2-7, 2023*. IEEE, 2023, p. 4788–4797. [En ligne]. Disponible : <https://doi.org/10.1109/WACV56688.2023.00478>
- [36] N. Aharon, R. Orfaig et B.-Z. Bobrovsky, “Bot-sort : Robust associations multi-pedestrian tracking,” 2022. [En ligne]. Disponible : <https://arxiv.org/abs/2206.14651>
- [37] Y. Du *et al.*, “Strongsort : Make deepsort great again,” *IEEE Transactions on Multimedia*, p. 1–14, 2023.
- [38] P. F. Felzenszwalb *et al.*, “Object detection with discriminatively trained part based models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, p. 1627–1645, 2010.

- [39] F. Yang, W. Choi et Y. Lin, “Exploit all the layers : Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2129–2137, 2016.
- [40] Z. Cai et N. Vasconcelos, “Cascade r-cnn : Delving into high quality object detection,” dans *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, p. 6154–6162.
- [41] Y. Du *et al.*, “Giaotracker : A comprehensive framework for mcmot with global information and optimizing strategies in visdrone 2021,” dans *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2021, p. 2809–2819.
- [42] G. D. Evangelidis et E. Z. Psarakis, “Parametric image alignment using enhanced correlation coefficient maximization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, p. 1858–1865, 2008.
- [43] H. Luo *et al.*, “Bag of tricks and a strong baseline for deep person re-identification,” dans *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019, p. 1487–1495.
- [44] Z. Zheng, L. Zheng et Y. Yang, “A discriminatively learned CNN embedding for person reidentification,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 14, n^o. 1, p. 1–20, dec 2017. [En ligne]. Disponible : <https://doi.org/10.1145%2F3159171>
- [45] A. Hermans, L. Beyer et B. Leibe, “In defense of the triplet loss for person re-identification,” 2017. [En ligne]. Disponible : <https://arxiv.org/abs/1703.07737>
- [46] L. Leal-Taixé, C. Canton-Ferrer et K. Schindler, “Learning by tracking : Siamese cnn for robust target association,” dans *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, p. 418–425.
- [47] Z. Wang *et al.*, “Towards real-time multi-object tracking,” dans *Computer Vision – ECCV 2020*, A. Vedaldi *et al.*, édit. Cham : Springer International Publishing, 2020, p. 107–122.
- [48] S. Tang *et al.*, “Multiple people tracking by lifted multicut and person re-identification,” dans *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, p. 3701–3710.
- [49] S. Sun *et al.*, “Deep affinity network for multiple object tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, n^o. 1, p. 104–119, 2021.
- [50] P. Bergmann, T. Meinhardt et L. Leal-Taixe, “Tracking without bells and whistles,” dans *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, oct 2019. [En ligne]. Disponible : <https://doi.org/10.1109%2Ficcv.2019.00103>

- [51] P. Sun *et al.*, “Transtrack : Multiple object tracking with transformer,” 2020. [En ligne]. Disponible : <https://arxiv.org/abs/2012.15460>
- [52] F. Zeng *et al.*, “Motr : End-to-end multiple-object tracking with transformer,” dans *Computer Vision – ECCV 2022*, S. Avidan *et al.*, édit. Cham : Springer Nature Switzerland, 2022, p. 659–675.
- [53] J. Peng *et al.*, “Chained-tracker : Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking,” dans *Computer Vision – ECCV 2020*, A. Vedaldi *et al.*, édit. Cham : Springer International Publishing, 2020, p. 145–161.
- [54] Y. Wang, K. Kitani et X. Weng, “Joint object detection and multi-object tracking with graph neural networks,” *2021 IEEE International Conference on Robotics and Automation (ICRA)*, p. 13 708–13 715, 2021.
- [55] B. Pang *et al.*, “Tubetk : Adopting tubes to track multi-object in a one-step training model,” dans *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [56] N. Carion *et al.*, “End-to-end object detection with transformers,” dans *Computer Vision – ECCV 2020*, A. Vedaldi *et al.*, édit. Cham : Springer International Publishing, 2020, p. 213–229.
- [57] T.-Y. Lin *et al.*, “Focal loss for dense object detection,” dans *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, p. 2999–3007.
- [58] S. Shao *et al.*, “Crowdhuman : A benchmark for detecting human in a crowd,” 2018. [En ligne]. Disponible : <https://arxiv.org/abs/1805.00123>
- [59] L. Leal-Taixé *et al.*, “Motchallenge 2015 : Towards a benchmark for multi-target tracking,” 2015. [En ligne]. Disponible : <https://arxiv.org/abs/1504.01942>
- [60] A. Newell, K. Yang et J. Deng, “Stacked hourglass networks for human pose estimation,” dans *Computer Vision – ECCV 2016*, B. Leibe *et al.*, édit. Cham : Springer International Publishing, 2016, p. 483–499.
- [61] X. Zhang *et al.*, “Head point positioning and spatial-channel self-attention network for multi-object tracking,” dans *2022 26th International Conference on Pattern Recognition (ICPR)*, 2022, p. 3500–3506.
- [62] Y. Zhang *et al.*, “Multiplex labeling graph for near-online tracking in crowded scenes,” *IEEE Internet of Things Journal*, vol. 7, n°. 9, p. 7892–7902, 2020.
- [63] Y. Chen *et al.*, “Appearance guidance attention for multi-object tracking,” *IEEE Access*, vol. 9, p. 103 184–103 193, 2021.

- [64] H. Caesar *et al.*, “nuscenes : A multimodal dataset for autonomous driving,” dans *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, p. 11 618–11 628.