`Neo4MOT` : suivi multi-objets dans un réseau de caméras à l'aide de graphes temporels multicouche

Pierre Lefebvre*, Ahmed Azough*, Nicolas Travers*

*De Vinci Higher Education, De Vinci Research Center, Paris, France
{prenom.nom}@devinci.fr
https://www.devinci.fr/research-center/

Résumé. La compréhension d'une vidéo s'appuie sur l'étude des interactions entre objets, lesquelles peuvent être représentées à l'aide de graphes de scène. Les occlusions entraînent toutefois des pertes d'information, rendant cruciale l'optimisation du suivi pour une représentation complète. L'approche multi-caméras représente une piste prometteuse en permettant de combiner plusieurs points de vue et en reliant les observations d'un même objet. Ainsi, nous présentons `Neo4MOT`, une méthode de suivi multi-objets multi-caméras (MCMOT) fondée sur un modèle de données de graphe temporel multicouche structurant les trajectoires, un réseau CNN pour l'extraction de caractéristiques visuelles, et un algorithme de graphe pour leur agrégation et l'association des trajectoires. La démonstration illustre son exécution sur les jeux de données CAMPUS, EPFL et PETS09, en comparant différents algorithmes de suivi (SORT, OC-SORT, ByteTrack) tout en permettant la visualisation des trajectoires avec Neo4j.

1 Introduction

Ces dernières années, le suivi multi-objets multi-caméras (MCMOT) est devenu un sujet de recherche majeur dans le domaine de la vision par ordinateur, en particulier pour génération de graphes de scène. En effet, une seule erreur de suivi peut fausser la représentation du contenu d'une scène et, par conséquent, l'interprétation des interactions et des actions qui s'y déroulent. MCMOT vise à suivre plusieurs objets en mouvement dans un réseau de caméras. Parmi les approches existantes, la méthode de *tracking-by-detection* est la plus répandue Amosa et al. (2023). Le processus de suivi commence par la détection des objets dans les images vidéo et l'attribution à chacun d'un identifiant unique. Cet identifiant est ensuite réattribué à chaque apparition successive du même objet, reliant ainsi les détections pour former une trajectoire continue représentant le parcours complet de l'objet. Les systèmes MCMOT trouvent des applications dans des domaines tels que la vidéosurveillance, la conduite autonome, la robotique, l'analyse vidéo et le sport, où le maintien de trajectoires d'objets précises est essentiel pour une prise de décision efficace.

MCMOT fait face à plusieurs défis, tels que la complexité des arrière-plans, l'incertitude des détections, les occlusions entre objets, les variations de vitesse et les trajectoires irrégulières. il peut être divisé en deux catégories : les méthodes en ligne Quach et al. (2021); Ristani

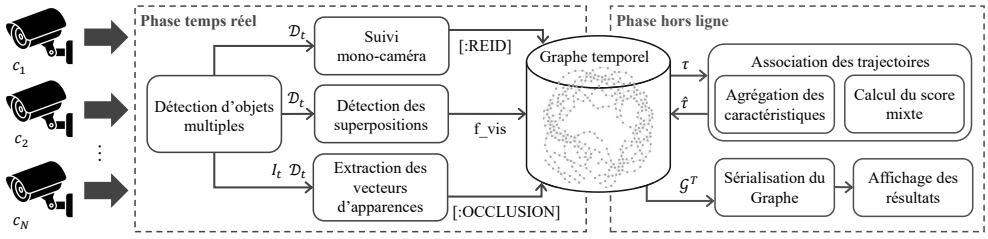

FIG. 1 – *Illustration de la méthode* `Neo4MOT`

et Tomasi (2018) et les méthodes hors ligne He et al. (2020); Liu et al. (2017); Hou et al. (2019). Les premières traitent les données en temps réel, tandis que les secondes les traitent de manière différée, en prenant en compte à la fois les détections passées et futures. Bien qu'elles reposent sur le principe du *tracking-by-detection*, la plupart de ces méthodes ne s'appuient pas explicitement sur des algorithmes de suivi mono-caméra (SCT). Elles sont donc peu susceptibles d'intégrer les avancées futures dans ce domaine. Le suivi mono-caméra regroupe des méthodes fondées sur des modèles de mouvement Bewley et al. (2016); Cao et al. (2023); Zhang et al. (2022); Yang et al. (2023), qui prédisent la position future des objets à partir de leurs positions et mouvements précédents, ainsi que des méthodes basées sur l'apparence Wojke et al. (2017); Du et al. (2023), qui exploitent les caractéristiques visuelles pour identifier et associer les objets entre les images successives.

Pour répondre à ces problématiques, nous proposons une approche modulaire combinant un suivi mono-caméra en temps réel et un post-traitement hors ligne basé sur les graphes pour le suivi multi-caméras, adaptée aux tâches d'investigation telles que l'analyse criminelle, qui nécessitent un traitement efficace de grands ensembles de vidéos. Notre méthode s'appuie sur un modèle de graphe temporel multicouche structurant les trajectoires, où les détections sont reliées par des relations de superposition au sein des couches et de ré-identification entre elles. Un modèle convolutif extrait les caractéristiques visuelles. Enfin, notre système reste compatible avec divers algorithmes de suivi mono-caméra, assurant flexibilité et extensibilité. Nos contributions se résument comme suit : i) un modèle de données en graphe temporel pour un stockage efficace des trajectoires ; ii) une méthode d'agrégation des caractéristiques visuelles indépendante de la longueur des trajectoires mitigeant les occlusions entre objets ; iii) une méthode d'association des trajectoire entre les caméra intégrant les caractéristiques visuelles fusionnées et les séquences de positions.

2 La méthode `Neo4MOT`

Nous proposons `Neo4MOT`, une méthode dédié au suivi multi-objets dans un réseau de caméras distribuées. Notre approche repose sur une chaîne de traitement modulaire de suivi mono-caméra permettant de suivre les objets dans chaque flux vidéo, sur un modèle de données en graphe structurant les trajectoires des objets précédemment détectés, ainsi que sur une phase de post-traitement visant à associer les trajectoires entre les différentes caméras par agrégation des caractéristiques visuelles stockées dans les nœuds.

2.1 Définition du problème

Soit un réseau de caméras connectées $\mathcal{C} = \{c_1,\ c_2,\ \ldots,\ c_N\}$ présentant un recouvrement partiel. Chaque caméra $c \in \mathcal{C}$ génère une séquence d'images $\mathcal{S}_c = \{I_t\}_{t=0}^{T}$, dans lesquelles un ensemble de détections $\mathcal{D}_t = \{d_i^t\}_{i=1}^{N_t}$ est obtenu à chaque instant discret t. Les détections successives d'une même entité sont regroupées en une trajectoire $\tau = \{d_i^{t_1}, d_j^{t_2}, \ldots, d_k^{t_L}\}$, de durée $\Delta t = t_L - t_1$. L'objectif est ensuite d'associer ces trajectoires afin de reconstruire, pour chaque entité, un parcours cohérent à travers l'ensemble des caméras du réseau.

2.2 Pipeline de Suivi

La première étape de notre méthode est un pipeline de suivi mono-caméra. Il consiste à ré-identifier les objets dans les images consécutives capturées par le réseau de caméras afin de former des trajectoires qui sont ensuite instanciées dans le graphe. Notre pipeline de suivi se décompose en quatre modules décrits ci-dessous.

Détection d'objets. Ce module repose sur un algorithme de détection d'objets et génère un ensemble de détections $\mathcal{D}_t$. Les détections sont ensuite mappées vers des nœuds de type `(:Detection)` dans le graphe. Chaque nœud possède un attribut `cam_id` identifiant la caméra à laquelle il appartient, un attribut `frame` correspondant au temps discret t, ainsi qu'un attribut `bbox` au format $[x_1, y_1, w, h]$, qui représentent respectivement les coordonnées du coin supérieur gauche et les dimensions de la boîte englobante de l'objet.

Suivi mono-caméra. Ce module repose sur un algorithme de suivi associant, d'une image à l'autre, les objets correspondant à une même entité afin de former un ensemble de trajectoires τ. Chaque caméra du réseau est associée à un modèle de suivi indépendant des autres. Les résultats du suivi sont mappés sous forme d'attributs de nœuds `pid` et de relations `[:REID]` entre les nœuds correspondant à une même entité à différents instants dans le graphe.

Détection de superpositions. Ce module détecte les occlusions entre les détections capturées par une caméra au sein d'une même image. En considérant deux détections d_i^t et d_j^t à l'instant t, nous définissons la fonction de recouvrement comme suit : $\Phi(d_i^t, d_j^t) = \frac{\mathcal{A}(\texttt{bbox}_j)}{\mathcal{A}(\texttt{bbox}_i \texttt{Ubbox}_j)}$, où $\mathcal{A}$ représente la surface couverte par une boîte englobante. Cette métrique quantifie le recouvrement spatial ρ de d_i^t sur d_j^t, variant de 0 (aucun recouvrement) à 1 (recouvrement total). Pour chaque paire de détections superposées dans une couche, deux relations non symétriques `[:OCCLUSION]` ainsi qu'une valeur de `superposition` correspondante sont créées.

Extraction des caractéristiques visuelles. Ce module extrait les caractéristiques visuelles f_i des détections à l'aide d'une modèle convolutif. Il génère l'attribut de nœud `f_vis`, permettant la ré-identification des objets entre les différentes caméras.

2.3 Modèle de données de graphe

Afin de formaliser l'instanciation des détections et des trajectoires au sein du graphe, nous introduisons un modèle de données à l'aide des définitions et propriétés suivantes. Notre graphe est un graphe de propriétés (LPG), et est composé de couches temporelles successives $\mathcal{L}_t$, correspondant à une scène observée selon différentes vues de caméras à un instant discret t.

Définition 1 (LPG temporel)**.** Soit $\mathcal{G}^T$ un graphe temporel définit par : $\mathcal{G}^T = (\mathcal{L},\ \mathcal{R}_{reid})$ avec $\mathcal{L}$ l'ensemble des couches temporelles, $\mathcal{R}_{reid}$ l'ensemble des relations de re-identification, et T le nombre total de couches temporelles.

Définition 2 (Couches temporelles)**.** Les couches temporelles $\mathcal{L}$ sont définies de la manière suivante : $\mathcal{L} = \{\mathcal{L}_t \mid t \in [1,\ T]\}$. Ainsi, $\mathcal{L}_t$ se réfère à une couche temporelle du graphe $\mathcal{G}^T$. $\mathcal{N}(\mathcal{L}_t)$ et $\mathcal{O}(\mathcal{L}_t)$ dénotent respectivement les ensembles de nœuds et les relations d'occlusions au sein d'une couche temporelle. De plus, les nœuds $n \in \mathcal{N}$ appartiennent à une unique couche temporelle : $\forall\ n \in \mathcal{N}(\mathcal{L}_t),\ \forall\ \mathcal{L}_{t'} \mid \mathcal{L}_{t'} \neq \mathcal{L}_t,\ n \notin \mathcal{N}(\mathcal{L}_{t'})$

Définition 3 (Relations d'occlusion)**.** les relations d'occlusion $o_{n \to n'}$ relient les nœuds n et n' au sein d'une même couche temporelle : $\mathcal{O}(\mathcal{L}_t) = \{o_{n \to n'} = (n,\ n',\ \rho) \mid n \in \mathcal{N}(\mathcal{L}_t)\ \wedge\ n' \in \mathcal{N}(\mathcal{L}_t)\ \wedge\ n \neq n'\ \wedge\ \rho = \Phi(n, n') \in\ ]0,\ 1]\}$ avec ρ la fonction de chevauchement entre deux nœuds $[0, 1]$. Par ailleurs, les relations d'occlusion ne sont pas symétriques : $o_{n \to n'} \neq o_{n' \to n}$.

Définition 4 (Relations de ré-identification)**.** Les relations de ré-identification $\mathcal{R}_{reid}$ relient les nœuds n et n' entre les couches temporelles $\mathcal{L}_t$ et $\mathcal{L}'_t$: $\mathcal{R}_{reid} = \{r_{n \to n'} = (n, n', s) \mid n \in \mathcal{N}(\mathcal{L}_t)\ \wedge\ n' \in \mathcal{N}(\mathcal{L}'_t)\ \wedge\ \mathcal{L}_t,\ \mathcal{L}_{t'} \in \mathcal{G}^T\ \wedge\ t < t'\}$ avec $r_{n \to n'}$ un lien de ré-identification $\in \mathcal{R}_{reid}$ entre deux nœuds n et n', et s le score de ré-identification $\in [0, 1]$. Les nœuds $n \in \mathcal{N}$ sont ré-identifiés au plus une fois.

Définition 5 (Trajectoire)**.** Une trajectoire dans le graphe est définie comme une séquence de nœuds distincts reliés par des relations de ré-identification : $\mathcal{T} = (n_0,\ n_1,\ ...,\ n_{k-1},\ n_k),\ k \in \mathbb{N}^* \mid \forall\ i \in \mathbb{N}^*,\ i \leq k,\ \exists!\ r_{i-1 \to i} \in \mathcal{R}_{reid}$

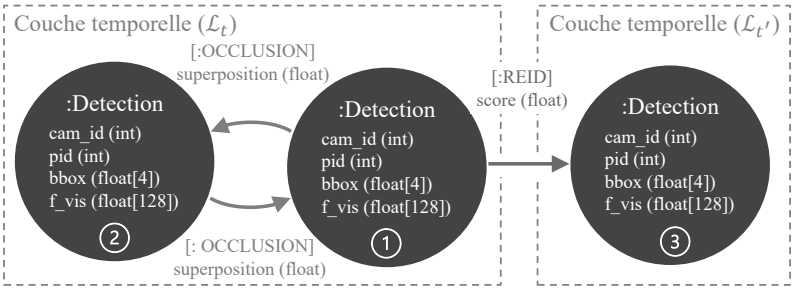

FIG. 2 – *Modèle de données de Graphe*

La figure 2 illustre notre modèle de données de graphe. Chaque nœud $n \in \mathcal{G}^T$ est de type `(:Detection)`. Les relations d'occlusion (déf. 3) sont typées `[:OCCLUSION]` et associées à une valeur de *superposition* au sein d'une couche temporelle (déf. 2). Enfin, les relations de ré-identification `[:REID]` (déf. 4) sont ajoutées entre des nœuds appartenant à des couches temporelles distinctes, tout en vérifiant la contrainte d'unicité.

Les nœuds ① et ② appartiennent à la même couche temporelle. Ils correspondent à des entités différentes et possèdent des attributs *pid* distincts. Les nœuds ① et ③ correspondent à une même entité capturée à des instants différents t et t'. Ils partagent donc le même attribut *pid* et forment un *tracklet* (l'ensemble des nœuds appartenant à une même chaîne formée par des relations de type `[:REID]` constitue alors une trajectoire).

2.4 Association des trajectoires

L'association hors ligne relie les trajectoires locales de différentes caméras en combinant apparence et déplacement spatial. Les descripteurs visuels extraits par le CNN sont agrégés selon $\Omega(\tau) = \frac{\sum_i f_i \mathbb{I}(\rho_i < \theta)}{\sum_i \mathbb{I}(\rho_i < \theta)}$, afin d'atténuer l'influence des détections partiellement occludées ($\rho_i \geq \theta$). Chaque détection est ensuite projetée dans un plan 2D à partir du centre du bord inférieur de sa boîte englobante via l'homographie H_c associée à la caméra. L'association entre trajectoires τ et τ' repose sur un score $S(\tau, \tau') = \alpha \cos(\bar{f}_\tau,\ \bar{f}_{\tau'}) + (1 - \alpha) \exp[-d(\tau, \tau')/\sigma]$, combinant similarité cosinus et distance spatiale entre trajectoires projetées. Cette formulation généralise le modèle de DeepSORT Wojke et al. (2017), où la cohérence de mouvement est modélisée par une distance de Mahalanobis. Les associations inter-caméras sont finalement sélectionnées par appariement biparti sous seuil de similarité.

3 Implémentation

Notre framework est basé sur différents modules pour la détection d'objets, la constructions des trajectoires, le mappage des données vers la base de données de graphe, la fusion des caractéristiques visuelles et l'association des pistes entre les caméras du réseau. L'accent est mis sur le module de détection, les méthodes de suivi implémentées, l'extraction des caractéristiques visuelles et la de fusion de ces dernière au sein des pistes.

Détection d'objets. Nous utilisons YOLOv11 [1], un algorithme de détection d'objets en une seule étape à l'état de l'art (SoTA). Les modèles YOLO détectent les objets en divisant l'image d'entrée en une grille, où chaque cellule permet la prédiction des boîtes englobantes, des classes et des scores de confiance. Un post-traitement par suppression des non-maxima (NMS) permet ensuite d'éliminer les doublons. YOLOv11 intègre des avancées récentes issues des versions précédentes, telles que le pooling spatial pyramidal pour l'extraction des caractéristiques, des mécanismes d'attention permettant de se concentrer sur les régions pertinentes de l'image, et la prédiction multi-échelle des caractéristiques visuelles afin d'améliorer la détection d'objets de différentes dimensions.

Algorithmes de suivi mono-caméra. Afin d'effectuer le suivi mono-caméra, nous ré-implémentons quatre méthodes à l'état de l'art : SORT Bewley et al. (2016), OC-SORT Cao et al. (2023), ByteTrack Zhang et al. (2022), et C-BiOU Yang et al. (2023). Chaque méthode peut être utilisé indépendamment pour chaque caméra.SORT repose sur la prédiction linéaire du mouvement et sur un appariement basé sur l'intersection sur l'union (IoU). OC-SORT améliore la gestion des occlusions grâce à une correction virtuelle des trajectoires. ByteTrack renforce la robustesse du suivi en séparant le traitement des détections fiables à score élevé de celles à score plus faible. Enfin, C-BIoU améliore les performances de suivi pour les mouvements irréguliers en utilisant un mécanisme de mise en mémoire tampon afin d'optimiser l'association des objets.

Extraction des caractéristiques visuelles. Notre modèle de ré-identification (ReID) est composé d'une architecture ResNet50 pour la partie convolution (backbone), suivie de deux couches entièrement connectées (FC) avec normalisation par batch et dropout afin d'éviter les problèmes de surapprentissage. Une fonction de perte contrastive basée sur la similarité cosinus évalue la distance entre les paires positives et négatives. Le réseau opère à deux niveaux au sein de notre framework. Le premier concerne l'extraction de caractéristiques, où des vecteurs de

1. YOLOv11 : `https://github.com/ultralytics/ultralytics`

dimension 128 sont extraits des images des objets durant la phase de suivi mono-caméra. Le second correspond au calcul de la similarité cosinus sur les vecteurs fusionnés afin d'associer les trajectoires lors de la phase multi-caméra. Nous entraînons notre modèle sur le jeu de données de référence Market-1501 pour 10 epochs, puis l'ajustons (fine-tuning) sur les ensembles d'apprentissage des jeux de données CAMPUS Kuo et al. (2010), EPFL Fleuret et al. (2007) et PETS09 Ferryman et Shahrokni (2009) pendant 10 epochs supplémentaires.

Association des trajectoires. Notre module d'association des trajectoires se décompose en deux étapes. La première est la fusion des caractéristiques visuelles des nœuds afin d'obtenir un vecteur unique pour chaque piste, excluant ainsi les objets avec un taux d'occlusions élevé. Cette étape est réalisée à l'aide de la requête Cypher suivante, implémentant notre fonction d'agrégation Ω.

```
MATCH (p:Person {pid: $pid})-[:REID*]->(pn:Person)
WHERE NOT EXISTS {
    MATCH (pn)<-[o:OCCLUSION]-(:Person)
    WHERE o.overlap > $treshold
}
WITH collect(pn.feature) AS vectors
WITH reduce(sumVector = [], vector IN vectors |
     [i IN range(0, size(vector) - 1) |
         coalesce(sumVector[i], 0) + vector[i]]
  ) AS summedVector, size(vectors) AS vectorCount
WITH [i IN range(0, size(summedVector) - 1) | summedVector[i] / vectorCount]
    AS averagedVector
RETURN averagedVector
```

Tout d'abord, nous effectuons une projection sur une piste entière pour un `$pid` donné en parcourant les liens de ré-identification `[:REID*]`. Ensuite, les vecteurs caractéristiques des noeuds sont collectés puis moyennés, ignorant ceux excédant un seuil de superposition `$threshold`. Les trajectoire et leurs vecteurs combinés sont ensuite passés à notre algorithme d'association implémenté en Python (voir la sous-section 2.4). Il convient de noter que, les liens d'occlusion `[:OCCLUSION]` n'étant pas symétriques, un lien peut être inférieur au `$threshold` dans un sens mais pas dans l'autre.

4 Démonstration

Notre scénario de démonstration se décompose en trois parties interactives portant sur la génération des trajectoires pour chaque caméra et leur association au sein du graphe.

Suivi mono-caméra en temps réel. La première étape de notre démonstration a pour objectif d'illustrer la génération des trajectoires et la modularité de notre méthode. Au cours de la présentation, les participants pourront sélectionner des vidéos issues de trois jeux de données (CAMPUS, EPFL et PETS09) ainsi que la méthode de suivi (SORT, OC-SORT, ByteTrack, C-BIOU) et la méthode de détection (YOLOv11 nano à large) pour chaque caméra. Les performances des algorithmes de détection et de suivi, ainsi que leurs avantages et leurs limites, seront ainsi discutées.

Suivi multi-caméra hors ligne. La deuxième étape adresse le suivi multi-caméra. Ce dernier repose sur la fusion des caractéristiques visuelles à l'aide de notre requête Cypher et sur notre méthode d'association des trajectoires. Une fois le graphe instancié, le suivi multi-caméra peu

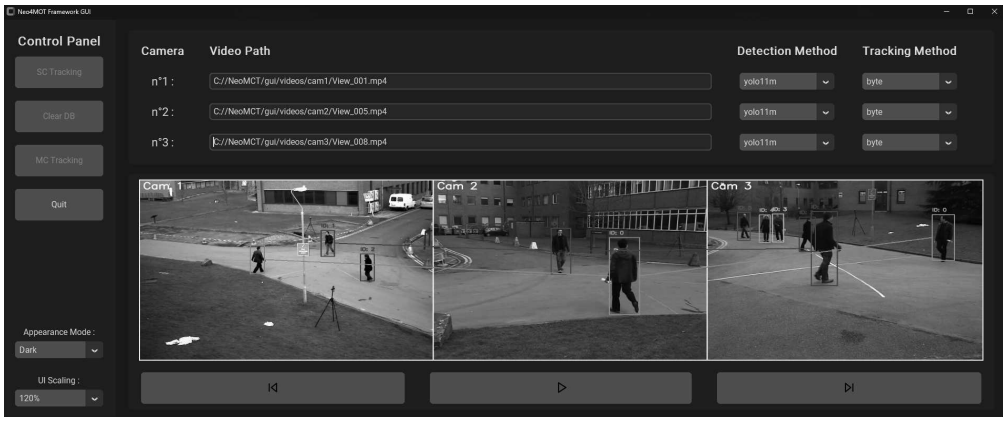

FIG. 3 – *Interface utilisateur de* `Neo4MOT` *avec les résultats de suivi pour les vues de caméras 001, 005 & 008 du jeu de données PETS09*

être effectué à partir de notre interface. Le public pourra visualiser les résultats du suivi, affichant jusqu'à trois caméras côte à côte, avec les détections, les identifiants et les liens d'association entre elles. Afin de faciliter la navigation au sein des vidéos, notre interface intègre un lecteur permettant soit une lecture accélérée offrant une vue d'ensemble des résultats de suivi, soit un défilement image par image afin d'examiner en détail certaines scènes spécifiques. Il convient de noter que notre méthode est extensible afin de gérer un plus grand nombre de caméras.

Fusion des caractéristiques. Dans cette dernière étape, les auditeurs interrogeront le graphe via l'interface Neo4j afin d'illustrer la fusion des vecteurs de caractéristiques au cœur de notre méthode de suivi multi-caméras. Le graphe sera d'abord visualisé dans son ensemble pour montrer la structuration des occlusions et des chaînes de ré-identification, puis plusieurs projections mettront en évidence des trajectoires spécifiques sous différentes vues. Les participants pourront ensuite exécuter la requête de fusion afin d'extraire les caractéristiques des trajectoires, constituer des paires positives (même individu) et négatives (individus distincts) et comparer ces dernier à l'aide d'une métrique de similarité (e.g. cosinus).

Remerciements. Ce travail de recherche a été financé par le ministère des Armées – Agence de l'Innovation de Défense (AID).

Références

Amosa, T. I., P. Sebastian, L. I. Izhar, O. Ibrahim, L. S. Ayinla, A. A. Bahashwan, A. Bala, et Y. A. Samaila (2023). Multi-camera multi-object tracking : a review of current trends and future advances. *Neurocomputing 552*, 126558.

Bewley, A., Z. Ge, L. Ott, F. Ramos, et B. Upcroft (2016). Simple online and realtime tracking. In *ICIP'16*, pp. 3464–3468. IEEE.

Cao, J., J. Pang, X. Weng, R. Khirodkar, et K. Kitani (2023). Observation-centric sort : Rethinking sort for robust multi-object tracking. In *IEEE/CVF'23*, pp. 9686–9696.

Du, Y., Z. Zhao, Y. Song, Y. Zhao, F. Su, T. Gong, et H. Meng (2023). Strongsort : Make deepsort great again. *IEEE Transactions on Multimedia 25*, 8725–8737.

Ferryman, J. et A. Shahrokni (2009). Pets2009 : Dataset and challenge. In *Workshop on performance evaluation of tracking and surveillance*, pp. 1–6. IEEE.

Fleuret, F., J. Berclaz, R. Lengagne, et P. Fua (2007). Multicamera people tracking with a probabilistic occupancy map. *IEEE transactions on pattern analysis and machine intelligence 30*(2), 267–282.

He, Y., X. Wei, X. Hong, W. Shi, et Y. Gong (2020). Multi-target multi-camera tracking by tracklet-to-target assignment. *IEEE Transactions on Image Processing 29*, 5191–5205.

Hou, Y., L. Zheng, Z. Wang, et S. Wang (2019). Locality aware appearance metric for multi-target multi-camera tracking. *arXiv preprint arXiv :1911.12037*.

Kuo, C.-H., C. Huang, et R. Nevatia (2010). Inter-camera association of multi-target tracks by on-line learned appearance affinity models. In *ECCV'10*, pp. 383–396. Springer.

Liu, W., O. Camps, et M. Sznaier (2017). Multi-camera multi-object tracking. *arXiv preprint arXiv :1709.07065*.

Quach, K. G., P. Nguyen, H. Le, T.-D. Truong, C. N. Duong, M.-T. Tran, et K. Luu (2021). Dyglip : A dynamic graph model with link prediction for accurate multi-camera multiple object tracking. In *IEEE/CVF'21*, pp. 13784–13793.

Ristani, E. et C. Tomasi (2018). Features for multi-target multi-camera tracking and re-identification. In *CVPR'18*, pp. 6036–6046.

Wojke, N., A. Bewley, et D. Paulus (2017). Simple online and realtime tracking with a deep association metric. In *ICIP'17*, pp. 3645–3649. IEEE.

Yang, F., S. Odashima, S. Masui, et S. Jiang (2023). Hard to track objects with irregular motions and similar appearances ? make it easier by buffering the matching space. In *IEEE/CVF'23*, pp. 4799–4808.

Zhang, Y., P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, et X. Wang (2022). Bytetrack : Multi-object tracking by associating every detection box. In *ECCV'22*, pp. 1–21. Springer.

Summary

Understanding video content relies on analyzing interactions between objects, which can be modeled using scene graphs. However, occlusions often cause information loss, making tracking optimization crucial for complete representation. The multi-camera approach offers a promising solution by combining multiple viewpoints and linking observations of the same object across cameras. We therefore present `Neo4MOT`, a multi-camera multi-object tracking (MCMOT) method based on a multi-layer temporal graph data model that structures object trajectories, a CNN for visual feature extraction, and a graph algorithm for feature aggregation and trajectory association. The demonstration showcases its execution on the CAMPUS, EPFL, and PETS09 datasets, comparing different tracking algorithms (SORT, OC-SORT, ByteTrack) and enabling trajectory visualization with Neo4j.