Étudiants : Allouchi Younes, Jeaidi Idriss
Matricule : 2231842, 2233451
Groupe : 9

Rapport de Vision par ordinateur INF6804

Travail Pratique #3

1. Présentation de la solution (Question 1)

Description de la solution

Pour la tâche de tracking d'objets multiples dans notre projet, nous avons choisi d'intégrer ByteTrack avec YOLOv8 comme modèle de détection d'objet. Cette méthode utilise les bounding boxes détectées temps réel de YOLOv8, pour ensuite utilser ByteTrack pour le suivi des objets à travers les frames d'une vidéo.

YOLOv8 est la dernière version stable du modèle "You Only Look Once", un modèle de détection d'objets extrêmement rapide et précis qui analyse les images en une seule passe, et donc prédit en une seule fois les boites englobantes ainsi que les classes associées. Le modèle utilise un CNN comme Backbone, et utilise aussi l'approche d'ancres abordées en cours. Plusieurs ancres sont placées dans l'écran, et pour chaque ancre plusieurs boites englobantes sont proposées. Ensuite, une fois que les boites englobantes les plus prometteuses et leur classe ont été déterminées par le modèle, il y'a une dernière étape de post processing pour supprimer les boites qui se chevauchent trop. Le Non-Maximum Suppression (NMS) est souvent utilisé pour le post processing.

ByteTrack prend en entrée les boites englobantes générées par YOLO, ainsi que les scores de confiances associés. Ensuite, pour effectuer l'association d'un même objet entre les frames, le modèle combine 2 approches, en plus d'un filtre de Kalman :

- ⇨ **IoU** : ByteTrack réalise une association de haut niveau basée sur le score IoU entre les boites englobantes détectées entre 2 frames consécutives. Cela est suffisant pour détecter suivre les objets qui n'ont pas trop bougés entre 2 frames.
- ⇨ **Réseau de neurones** : Pour les boites englobantes qui n'ont pas été associées suite à la première approche, ByteTrack extrait les features des objets qu'elles contiennent à l'aide d'un CNN, pour réaliser l'association même si l'IoU est bas. C'est une association de bas niveau qui se base sur l'apparence des objets.
- ⇨ **Filtre de Kalman** : Pour prédire la position et la vitesse des objets, ByteTrack intègre un filtre de Kalman. Ce filtre utilise un modèle mathématique pour estimer la position future et la vitesse d'un objet basé sur son état estimé actuel et les mesures de détection passées. Lorsqu'un objet est détecté, le filtre de Kalman est utilisé pour prédire sa position dans les frames suivantes, prenant en compte l'incertitude et le bruit possible des mesures. Le filtre de Kalman contribue à maintenir un suivi fluide et précis des objets,

même en cas d'occlusion partielle ou de mouvements rapides, en mettant à jour ses estimations avec chaque nouvelle mesure et en ajustant ses prédictions en conséquence.

Avantages :

- ⇨ **Précision et rapidité** : Yolo est un modèle très précis ce qui va nous être très utile étant donné la difficulté de la séquence vidéo que nous devons traité. De plus c'est un modèle très rapide, ce peut être utile lorsque l'on souhaite traiter des séquences vidéos en temps réel.
- ⇨ **Robustesse aux occlusions** : Grâce à son utilisation de 2 approches pour l'association, Bytetrack gère bien les cas où un objet est caché ou disparait temporairement du champ. En ne se basant pas seulement sur l'IoU, si un objet disparait et réapparait beaucoup plus loin, il pourra être associé grâce à l'association par features, qui reconnait si c'est le même objet grâce à son apparence.

Inconvénients :

- ⇨ **Inter dépendance** : Bytetrack dépend des prédictions de YOLO, ainsi une mauvaise prédiction de YOLO entrainera un mauvis suivi.

2. Difficulté de la séquence.

La séquence est complexe pour 2 raisons principales :

- ⇨ **Densité des objets :** La scène contient régulièrement plus d'une douzaine de tasses simultanément, ce qui crée un environnement dense et augmente la difficulté de différencier et de suivre chaque objet individuellement.
- ⇨ **Occlusions** : Les occlusions sont très fréquentes dans la séquences, souvent entre plusieurs tasses, et elles sont importantes, c'est-à-dire qu'il arrive que la majorité de la tasse soit cachée. Cela rend difficile la tâche de détection.
- ⇨ **Disparition temporaire** : Des interruptions dans la traçabilité des tasses surviennent lorsque celles-ci quittent temporairement le champ de vision et réapparaissent ultérieurement. Ces discontinuités dans la détection peuvent rendre difficile le suivi.
- ⇨ **Présence de répliques**: il y'a certaines tasses qui apparaissent en plusieurs exemplaires, ce qui peut induire en erreur le modèle de suivit, qui pensera qu'il s'agit de la même tasse et lui affectera donc le même identifiant.

3. Justification de la méthode utilisée (question 3)

Pour justifier le choix de notre approche, nous allons expliquer comment celle-ci peut répondre aux problématiques que nous avons mentionnées à la question précédente

- ⇨ **Densité des objets :** Grâce à son système d'ancres que nous avons mentionné dans la première question, YOLO est capable de détecter de multiples objets sans soucis. L'image est quadrillée et plusieurs ancres sont ajoutées, les boites englobantes qui en

découlent sont toutes analysées indépendamment, ce qui implique qu'avoir beaucoup d'objets n'impact pas le processus d'évaluation.

- ⇨ **Occlusions** : Comme nous l'avons mentionné durant l'explication des modèles, ByteTrack excelle particulièrement bien dans le suivi d'objets lors d'occlusions, étant donné que le modèle se base aussi sur les features des objets, pas seulement sur l'IoU, pour faire le suivi. Cela permet d'identifier une tasse même si celle-ci est partiellement cachée. De plus, YOLO a été entrainé sur un dataset contenant beaucoup de cas d'occlusions, le rendant plus robuste.
- ⇨ **Disparition temporaire** : Bytetrack garde les informations des objets récentes mêmes si ces derniers disparaissent de l'écran. Ainsi, lorsqu'un objet réapparait, il sera comparé à sa précédente version si celle-ci est toujours enregistre. De plus, même s'il apparait trop loin de l'endroit où il a disparu pour que l'IoU fasse l'association, alors l'association par features devrait pouvoir régler le problème. De plus, le filtre de Kalman continue de prédire sa trajectoire en se basant sur le modèle de mouvement estimé. Lorsque l'objet réapparaît, les prédictions du filtre peuvent être utilisées pour aider à la réidentification et la reprise du suivi.
- ⇨ **Présence de répliques**: Il s'agit du problème le plus difficile à résoudre. Il devrait poser des difficultés, mais ByteTrack se base aussi sur le mouvement des objets pour faire le suivi, pas seulement sur l'apparence et l'IoU. Ainsi, même si 2 objets identiques apparaissent en même temps sur des frames consécutives, leur mouvement devrait être différent ce qui devrait en théorie permettre de les différencier.

4. Implémentation (question 4)

Librairies utilisées

Nous avons pu implémenter notre approche à l'aide de 3 sources principales :

- ⇨ **ByteTrack** : Pour le modèle, nous avons cloner le github officiel de l'article, étant donné qu'il n'existe pas de librairies implémentant directement le modèle. Il a fallu isoler les fichiers intéressants et se plonger dans la documentation pour comprendre son fonctionnement, ses inputs et ses outputs.
- ⇨ **YOLOV8** : Nous avons utilisé la librairie « ultralitycs », qui permet de simplement utiliser plusieurs versions de YOLO.
- ⇨ **Supervision :** Une librairie qui facilite l'intégration entre YOLO et ByteTrack, et qui propose des fonctions pour la visualisation des boites englobantes.

Initialisation

- ⇨ **Chargement du modèle YOLO** : Nous instancions le modèle YOLO en utilisant le fichier pré-entraîné **yolov8n.pt**. Les informations du modèle sont obtenues par **model.info()**.
- ⇨ **Définition des constantes** :
  - **CLASS_NAMES_DICT** est un dictionnaire qui mappe l'identifiant de chaque classe à son nom.
  - **selected_classes** contient les identifiants de classe d'intérêt - ici, le numéro 41 correspondant aux tasses.

Préparation des images

⇨ **Extraction des images** : Une fonction **get_frames_list** est définie pour récupérer et trier les images de la séquence par ordre numérique.

Tracking et Annotation

⇨ **Initialisation de ByteTrack** : Un objet ByteTrack est créé avec des paramètres spécifiques pour l'activation du suivi, le buffer de perte de suivi et le seuil de correspondance minimum.
⇨ **Initialisation des Annotateurs** : Des instances pour annoter les boîtes (BoxAnnotator), les trajectoires (TraceAnnotator) et la zone de la ligne (LineZoneAnnotator) sont créées.

Boucle de traitement des images

⇨ **Traitement de chaque image** :
- Chaque image est lue et soumise au modèle YOLO pour obtenir des prédictions.
- Ces prédictions sont converties en détections compréhensibles par ByteTrack.
- Seules les détections appartenant aux classes sélectionnées sont retenues.
- ByteTrack met à jour le suivi avec les nouvelles détections.
- Les cadres annotés sont créés avec des informations de suivi telles que l'identifiant du tracker et la confiance.

Extraction des predictions

⇨ Les informations sur la position et l'identité de l'objet suivi sont extraites de chaque détection et stockées dans une liste **predictions**.
⇨ Il faut convertir le format <x,y,x,y> au format <x,y,w,h> demandé.

Stockage des résultats

⇨ **Écriture des prédictions dans un fichier texte** : Les données de suivi sont enregistrées dans un fichier **output.txt** que nous soumettrons ensuite sur Moodle.

6. Résultats (question 5)

Dans le but d'évaluer notre modèle, nous avons choisit d'utiliser le dataset MOT17, qui est un choix populaire pour les tâches de tracking multi-objets (MOT) en computer vision. Pour la métrique, nous avons utilisé HOTA comme indiqué dans l'énoncé, qui fonctionne comme suit :

HOTA est une métrique qui évalue à la fois la précision de la détection des objets (Detection Accuracy) et la précision de leur association au fil du temps (Association Accuracy). Pour calculer HOTA, on commence par faire correspondre les détections avec les objets de vérité terrain en utilisant un seuil d'Intersection sur Union (IoU), qui est modulé par un paramètre alpha variant de 0 à 1.

Ensuite, HOTA est obtenue en calculant la moyenne harmonique de deux sous-métriques : DetA, qui quantifie la précision des détections par rapport à la vérité terrain sans prendre en compte l'identité des objets, et AssA, qui évalue à quel point les identités des objets sont correctement associées sur l'ensemble des frames.

Un score HOTA élevé indique un bon équilibre entre une détection fiable et une association cohérente, ce qui est essentiel pour un système de suivi robuste. En ajustant le seuil alpha, nous pouvons contrôler la tolérance pour les correspondances de détection et ainsi étudier l'impact de la précision de localisation sur la performance globale de suivi.

Ainsi, voici un graphique récapitulatif des performances du modèles en fonction du treshold alpha, illustrant à la fois les scores HOTA, DetA, AssA et LocA, qui correspond au degré de chevauchement entre la boite englobante prédite et la vérité terrain :

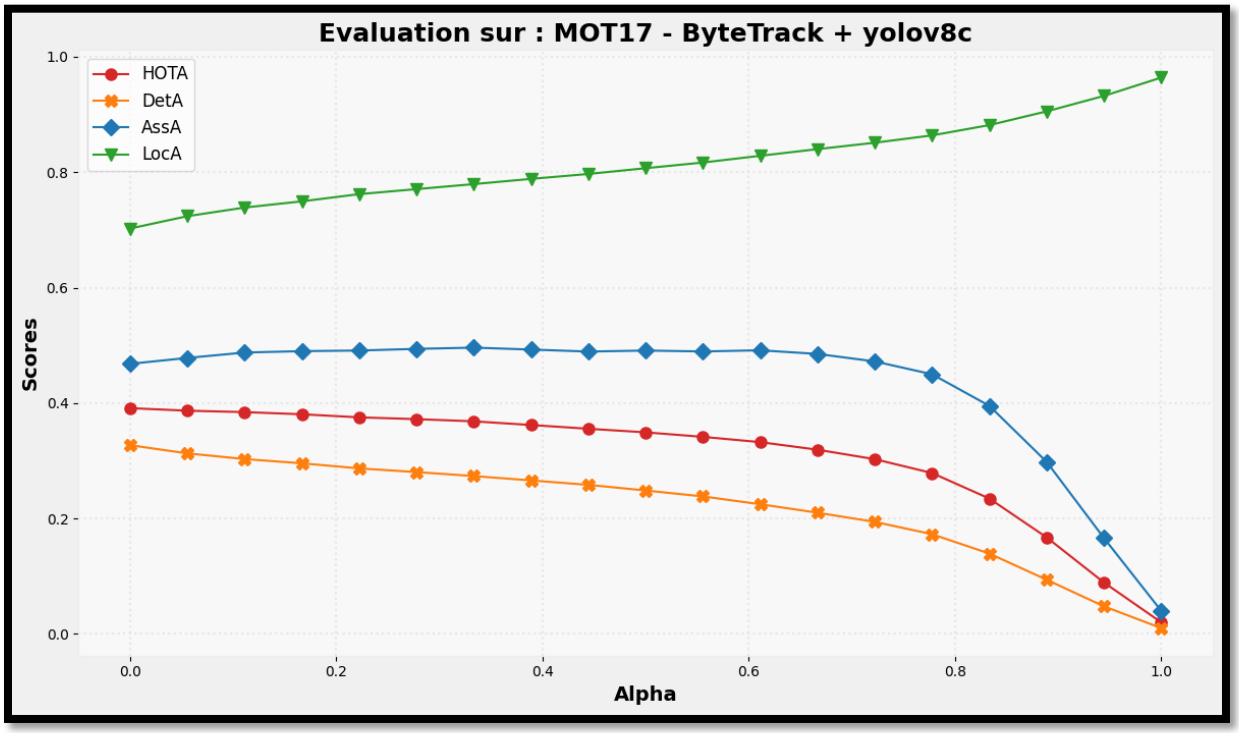


Figure 1 : Métriques en fonction d'alpha- séquence DPM02

Comme nous pouvons le voir, nous obtenons de bonnes performances. Si l'on choisit un alpha de 0.25, 0.5 et 0.75, voici les performances en détail

Alpha	DetA	AssA	LocA	HOTA
0.25	0.28	0.49	0.77	0.38
0.5	0.24	0.49	0.8	0.35
0.75	0.17	0.44	0.86	0.25

Tableau 1 : Performances précises – séquence DPM02

Plus précisément, nous avons évalué notre modèle sur la séquence DPM-02 en particulier du dataset MOT17, dataset qui est composé de plusieurs séquences où l'objectif filmées à l'extérieur. Intuitivement, le score HOTA diminue avec plus on augmente alpha, étant donné que la métrique devient de plus en plus stricte

Visualisation des résultats en image

Voici un exemple d'une image de la séquence DPM-04, accompagnées des prédictions de notre modèle et du ground truth :

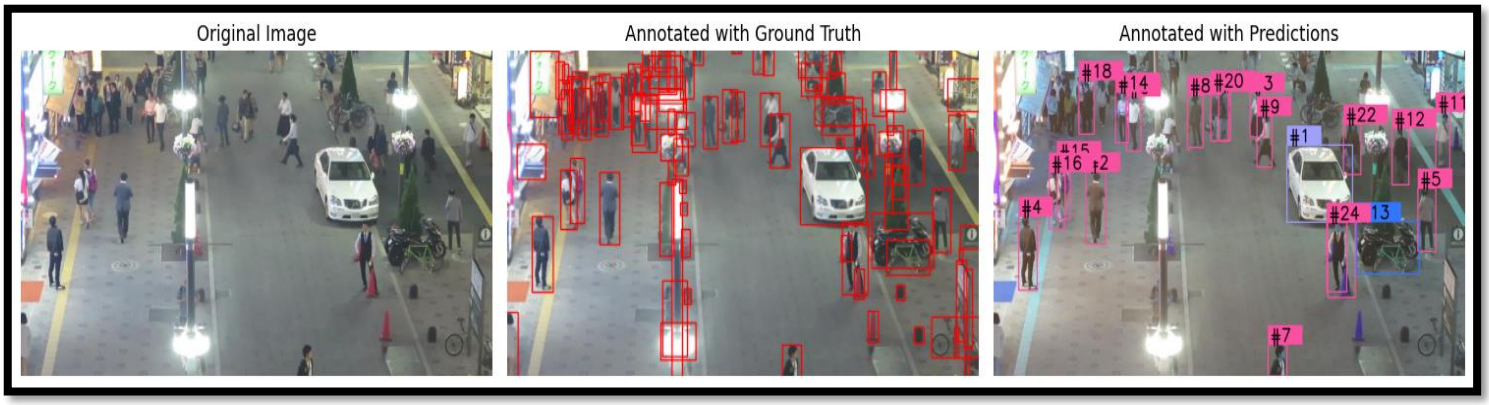


Figure 2 : Prédictions – MOT17-04-DPM

Notre modèle arrive a détecter une très bonne partie des objets de la scène, y compris certains objets qui ont de l'occlusion. Ainsi, montre bien que malgré le fait que le score DetA soit un légèrement plus bas que les autres, la capacité de détection de notre modèle est tout de même très bonne. Voici des exemples d'autres séquences du dataset MOT17 :

Figure 3 : Prédictions – MOT17-10-DPM

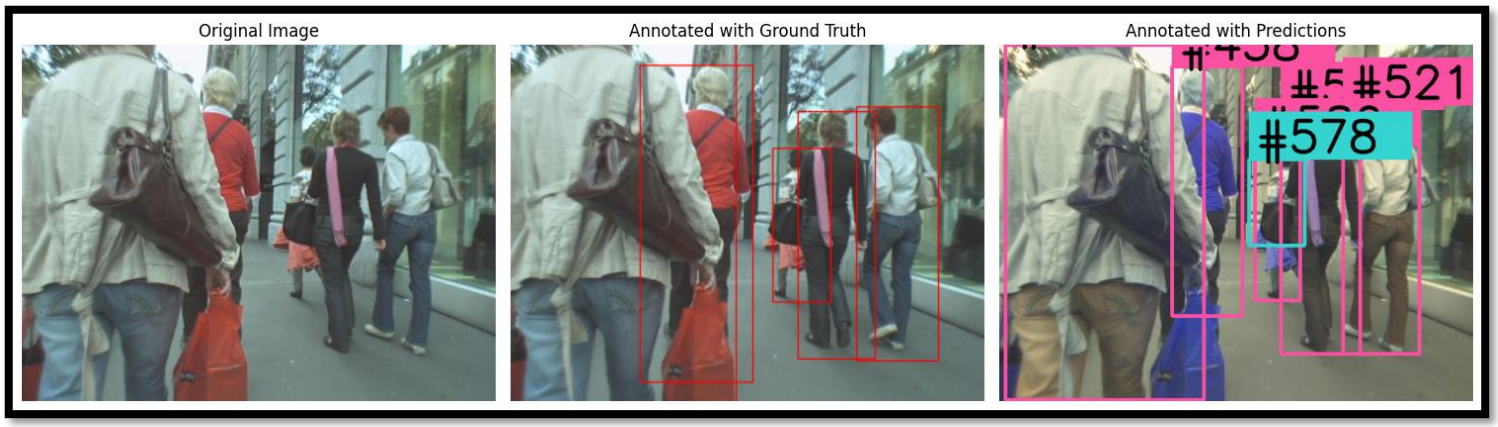


Figure 4 : Prédictions – MOT17-05-DPM

Pour conclure cette partie de présentation des résultat, voici les métriques moyennes sur chacune des séquences présentées ci-dessus :

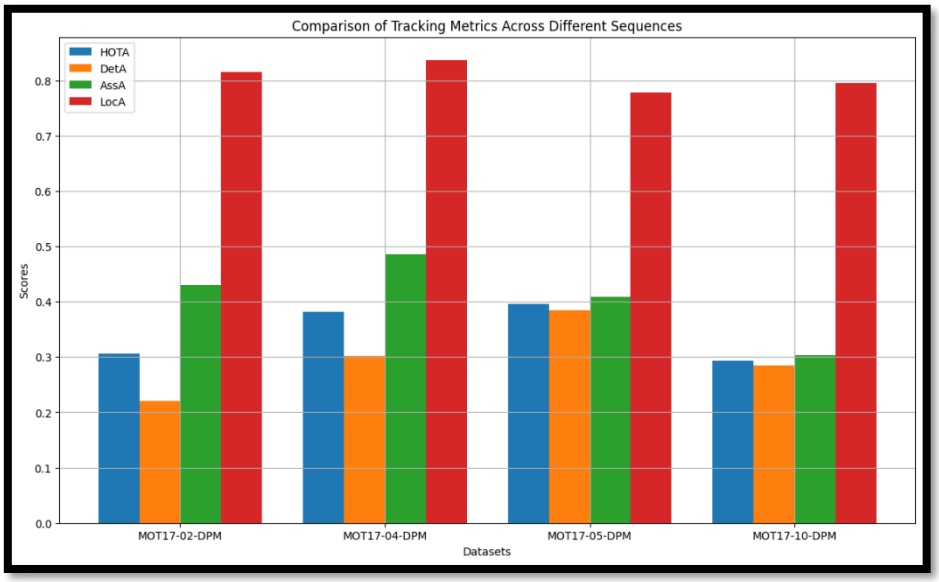


Figure 5 : Comparaison des métriques sur plusieurs séquences

Contrairement à la figure 1 qui montre les résultats précis, en fonction d'alpha, sur une seul séquence, nous observons ici les résultats plus généraux du modèle, afin de montrer que les performances observées plutôt sont consistantes sur plusieurs évaluations, illustrant la robustesse de notre modèle.

7. Discussion des résultats (question 7)

HOTA

Les forces de notre modèle

Localisation : Comme nous pouvons l'observer sur le tableau, tandis que sur la figure X, le score LocA est toujours très bon. La métrique LocA évalue, une fois qu'un objet a été detecté, la précision de cette détection. Elle ne dépend pas du nombre de prédiction, l'ensemble mesuré sont les boites englobantes prédites. Ainsi, un manque de détections n'impacte pas ce score. En somme, cela signifie que lorsque le modèle détecte un objet, la boite englobante qu'il dessine est très précise, étant donné que le score LocA est toujours aux alentours de 0.8

Association (AssA) : Avec des valeurs d'AssA atteignant 0.49 à alpha 0.25 et 0.5 et légèrement inférieures à 0.44 à alpha 0.75, le modèle montre une capacité robuste à maintenir des associations cohérentes entre les identités d'objets d'une image à l'autre. Cela est particulièrement remarquable dans des environnements complexes où les objets peuvent être nombreux et où leurs interactions peuvent compliquer le suivi, ce qui est le cas dans la séquence MOT17, qui est particulièrement complexe.

Robustesse globale (HOTA) : Le score HOTA est aussi assez impressionnant, surtout aux valeurs plus basses d'alpha, suggérant que le modèle réalise un équilibre compétent entre la détection précise et le suivi cohérent des objets au fil du temps. Un HOTA de 0.38 à alpha 0.25 et maintenu à 0.35 à alpha 0.5 montre que le modèle est capable de performances globales robustes en dépit des difficultés posées par la détection dans certains cas.

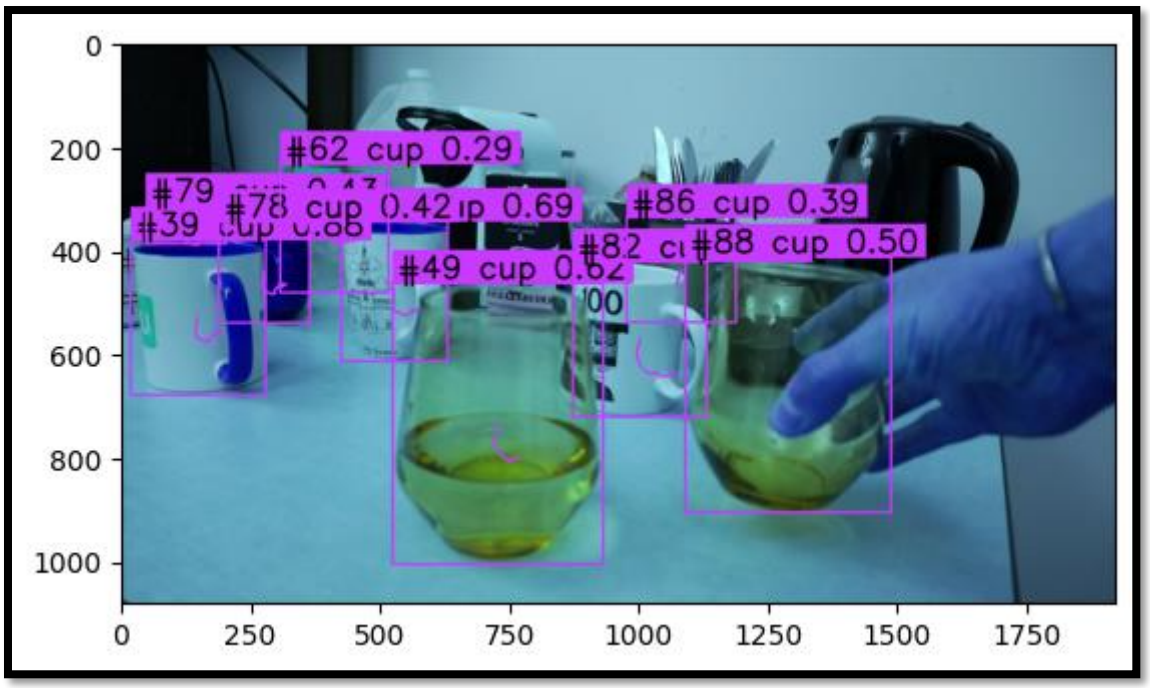
Les faiblesses de notre modèle

Détection (DetA) : La faiblesse la plus notable du modèle est reflétée par le score DetA qui diminue significativement avec l'augmentation de alpha. Un DetA de 0.28 à alpha 0.25 et qui descend à 0.17 à alpha 0.75 indique que le modèle pourrait manquer des détections, particulièrement dans des scénarios où la précision de la détection est mise à l'épreuve par des seuils de IoU plus élevés. Cette faiblesse peut provenir certainement liée à la complexité de la séquence, qui contient énormément d'occlusions et des objets très petits fondues dans le décor. Il y'a énormément de bruits.

Résultats par rapport aux défis de la question 2

Nous répondrons à cette partie à partir des résultats ainsi que des exemples de prédictions de notre modèle pour des situations illustrant les défis mentionnés.

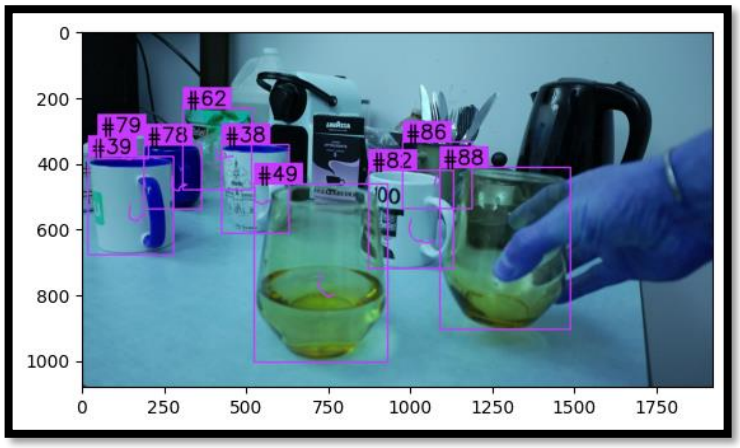
Densité des objets, occlusion et présence de répliques

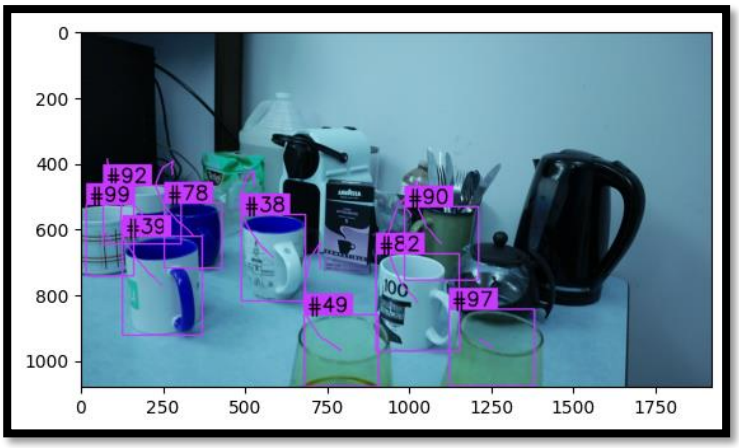


Densité : Même si le score DetA est moins bon que nos autres scores, il est bon dans l'absolue. Nous pouvons observer que notre modèle détecte toutes les tasses, malgré le grand nombre qui est présent sur l'image.

Occlusion : L'image choisie est aussi un bon exemple d'occlusion, ainsi notre modèle, comme prévue dans nos hypothèses, est robuste aux occlusions, ou en tout cas suffisamment pour la séquence de Moodle.

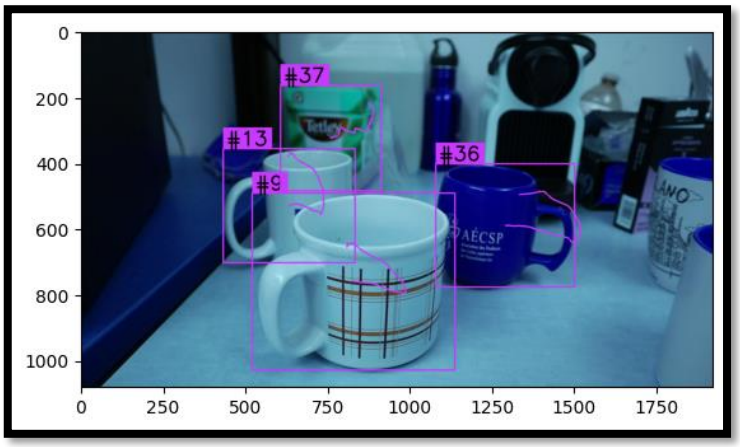
Présence de répliques : Il y'a deux exemplaires de la tasse verte au premier plan. Pour ce qui est de la détection, elles sont bel et biens détectées et identifiées comme 2 instances différentes. Cependant, le modèle a des difficultés à faire le suivi des 2 répliques en même temps. Comme nous pouvons le voir sur les 2 images précédentes, l'identifiant de la réplique de droite est différent dans les 2 frames, passant de **l'id 88** à **l'id 97** :

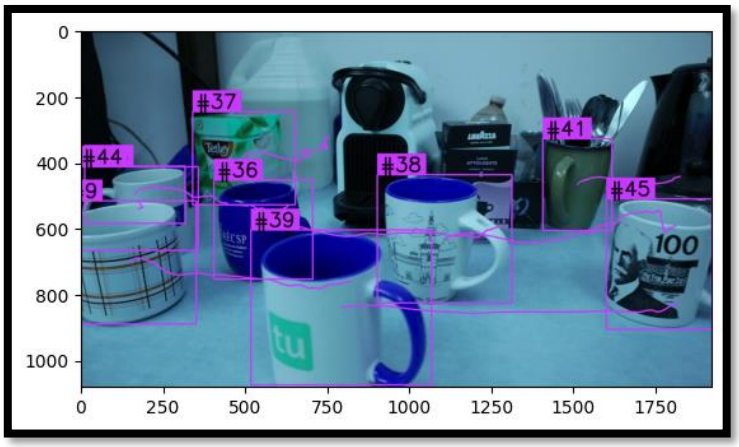




Suivi

Cependant, ce n'est que dans le cas précédent que le suivi a des difficultés. En cas général, le suivi a de très bonnes performances, comme le montre notre score AssA ainsi que les deux images suivantes, sur lesquels nous avons tracé le chemin parcouru par chaque objet tracké par notre modèle :





Le chemin tracé est correcte et les identifiants sont consistants.