



**POLYTECHNIQUE  
MONTRÉAL**  
UNIVERSITÉ  
D'INGÉNIERIE

INF6804

VISION PAR ORDINATEUR  
RAPPORT

---

## TP3 - Détection et suivi d'un objet d'intérêt

---

*Elèves :*

Marc ZHANG 2312403

Pierre CHAN KAN LEONG 2225665

*Enseignant :*

Guillaume-Alexandre

BILODEAU

## Table des matières

<b>1 Description de la solution</b>	<b>2</b>
<b>2 Identificcation des difficultés dans le dataset moodle</b>	<b>2</b>
<b>3 Justification de la méthode par rapport aux difficultés identifiées</b>	<b>3</b>
<b>4 Description du choix des outils utilisés dans notre implémentation</b>	<b>4</b>
<b>5 Présentation des résultats de validation</b>	<b>5</b>
5.1 Présentation du dataset utilisé . . . . .	5
5.2 Description des hyperparamètres . . . . .	5
5.3 Résultats avec les paramètres initiaux . . . . .	5
5.4 Amélioration des hyperparamètres . . . . .	7
5.5 Résultats avec les hyperparamètres améliorés . . . . .	8
<b>6 Résultats sur le dataset moodle</b>	<b>9</b>

## 1 Description de la solution

Dans le cadre de notre devoir, nous avons dû implémenter une méthode capable d'effectuer la détection et le suivi de multiples objets d'intérêt à travers une séquence vidéo. Le fonctionnement de ces traceurs d'objets multiples fonctionne en deux parties :

- Dans un premier temps, sur chaque frame de la vidéo, les objets sont d'abord détectés et placés dans des boîtes englobantes. Dans notre cas, nous utiliserons un modèle de deep learning basé sur les réseaux convolutifs. L'avantage de celui-ci est la robustesse, étant donné que ce genre de solution donne parmi les meilleurs résultats aujourd'hui. Cependant, il présente l'inconvénient d'exiger un temps de calcul important.
- Ensuite, chacun de ces objets est décrit en lui attribuant un identifiant unique et l'association de données est effectuée en associant ces objets avec ceux des frames précédentes pour réaliser un suivi. Dans notre cas, nous utiliserons l'algorithme SORT qui prendra en charge ces boîtes englobantes fournies par notre modèle de détection et qui effectuera le suivi de ces objets à travers les frames. SORT utilise principalement un filtre de Kalman pour prédire les mouvements des objets et un algorithme de correspondance pour associer les détections d'une frame à l'autre, permettant ainsi de construire des trajectoires. L'avantage de celui-ci est sa rapidité grâce à sa simplicité, qui le rend moins gourmand en ressources computationnelles. Le désavantage est sa robustesse, car SORT ne performe pas aussi bien que d'autres solutions existantes, comme par exemple DeepSORT, qui est une extension de SORT utilisant en plus des réseaux de neurones pour améliorer ses résultats.

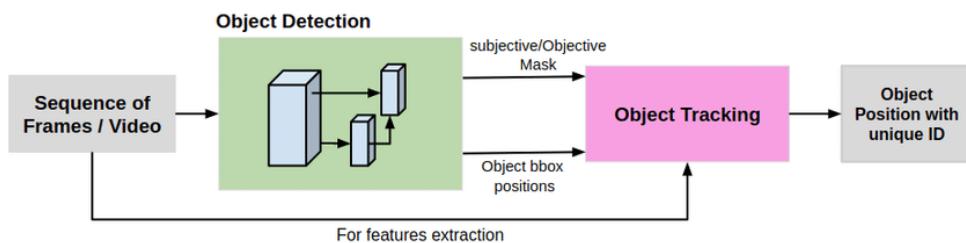


FIGURE 1 – Description de la pipeline d'un trackeur d'objet multiple

## 2 Identificcation des difficultés dans le dataset moodle

Dans le cas du dataset donné sur moodle à traiter, nous avons pu identifier plusieurs points pouvant conduire à des difficultés lors de la détection et du tracking :

- Sortie partielle ou occlusion partielle de l'objet du champ de la caméra : Sur certaines frames, nos tasses sont partiellement hors du champ de la caméra ou occultées par un autre objet au premier plan. Cela peut poser des problèmes, surtout au niveau de la détection, car cela peut rendre notre modèle de détection moins confiant sur la présence de l'objet désiré, ce qui pourrait le conduire à ne pas signaler sa présence. Le même problème peut affecter le tracking en remettant en question si l'objet est le même que celui de la frame précédente.
- Rotation de notre objet : Dans certaines parties de la vidéo, notre objet effectue une rotation, soit par manipulation directe, soit par un mouvement de la caméra

changeant l'angle sous lequel l'objet est filmé. Cela peut affecter les performances de notre solution, surtout au niveau du tracking de l'objet, car la rotation peut perturber notre système en le faisant identifier comme un nouvel objet et lui attribuer un nouvel identifiant.

- Sortie totale ou occlusion totale de l'objet du champ de la caméra : À certains moments de la vidéo, des objets sont totalement cachés par d'autres ou sortent entièrement du champ de la caméra. Cela peut poser des problèmes de tracking, car notre traqueur attribue un "âge maximal" à chaque objet suivi. Lorsqu'un objet sort du champ visuel de la caméra, cet âge commence à s'écouler, et si l'objet revient après que cet âge maximal soit dépassé, il est considéré comme nouveau par le tracker, ce qui peut mener à l'échec de son suivi.
- Changement du background par des mouvements de la caméra : Les mouvements de la caméra modifiant le fond de la vidéo peuvent perturber notre solution. En effet, la détection peut devenir instable, changeant l'éclairage ou causant des distorsions dans la vidéo, ce qui peut entraîner des faux positifs ou des faux négatifs.
- Présence d'objets transparents : La présence de verres transparents peut réduire nos performances. D'une part, au niveau de la détection, l'objet peut ne pas être détecté car il n'est pas considéré comme une tasse. D'autre part, au niveau du tracking, l'objet peut ne pas être bien suivi car il change constamment en fonction de ce qu'il contient (dans la vidéo, l'un est rempli tandis que l'autre est vidé) et de ce qui se trouve derrière lui.

## 3 Justification de la méthode par rapport aux difficultés identifiées

Dans un premier temps, parmi les difficultés potentiellement résolubles, nous avons :

- Les difficultés liées aux occlusions partielles ou aux sorties partielles du champ de la caméra : Du côté de la détection, les réseaux de neurones sont généralement efficaces pour identifier des objets même en présence de perturbations partielles comme les occlusions ou les objets partiellement hors champ, grâce à leur capacité à apprendre des caractéristiques robustes et discriminantes des objets d'intérêt. Du côté du suivi, le tracking peut gérer les occlusions temporaires, car le filtre de Kalman aidera à prédire la position de l'objet même lorsqu'il est partiellement visible, ce qui maintient l'identité de l'objet tant qu'il réapparaît dans un intervalle de temps acceptable.
- Les difficultés liées à la rotation de notre objet : Bien que les rotations puissent perturber le suivi en modifiant l'apparence de l'objet, les réseaux neuronaux, s'ils sont entraînés avec des données variées montrant des objets sous différents angles, peuvent généraliser et détecter efficacement l'objet malgré ces changements. Cela aide SORT à maintenir l'identité correcte à travers les rotations, à condition que la détection soit précise.

Ensuite, parmi les difficultés qui restent difficiles à traiter, nous avons :

- Sortie totale de l'objet du champ de la caméra / Occlusion totale : La performance face à cette difficulté repose sur le choix de l'hyperparamètre de l'âge maximal pour l'algorithme SORT. Un âge trop faible risque de faire perdre le suivi d'un objet lorsqu'il disparaît, mais un âge trop élevé augmente le risque de faux po-

sitifs et accroît la complexité de calcul due à la gestion d'un trop grand nombre d'identifiants.

- Changement du background par des mouvements de la caméra : Les mouvements de la caméra peuvent induire des variations significatives dans l'environnement de fond, affectant ainsi la précision de la détection. Les détecteurs basés sur les réseaux neuronaux peuvent être perturbés par ces changements, surtout s'ils n'ont pas été entraînés spécifiquement pour gérer de tels scénarios. Cela pourrait à son tour affecter la performance de SORT, car il dépend de détections précises pour un suivi efficace.
- Présence d'objets transparents : La détection d'objets transparents reste un défi pour les réseaux neuronaux convolutifs car ces objets peuvent facilement se confondre avec leur environnement en raison de leur transparence et des reflets, rendant leur détection inconsistante. Cela peut également compliquer le suivi, car SORT peut perdre le suivi d'un objet si celui-ci n'est pas bien détecté à chaque frame.

## 4 Description du choix des outils utilisés dans notre implémentation

Pour notre détection, nous allons utiliser le modèle basé sur les réseaux convolutifs YOLO, et plus précisément sa version 8 light<sup>1</sup>. L'un des points forts de ce modèle est sa performance, le classant très bien parmi les autres modèles basés sur les réseaux convolutifs. Un autre avantage de YOLO est sa rapidité, le rendant capable de calculs en temps réel, ce qui compense grandement l'un des désavantages de l'utilisation d'un réseau convolutif. Cependant, notre configuration est loin d'être suffisante pour le temps réel, d'où le choix de la version light pour accélérer au maximum le calcul au détriment des performances. Sur notre configuration, nous étions à un traitement de 2 à 3 frames par seconde.

Pour notre suivi, comme expliqué précédemment, nous utiliserons l'algorithme SORT<sup>2</sup>, dont les avantages et inconvénients d'implémentation sont déjà décrits dans la première partie.

La conception du pipeline est grandement inspirée de celle décrite dans un article<sup>3</sup> écrit sur le site medium.com par un utilisateur nommé Mosesdaudu. Chacune des deux parties de notre implémentation possède un ou plusieurs hyperparamètres dont nous détaillerons le choix plus tard. Le choix d'implémentation de ces deux algorithmes nous donne l'avantage d'être utilisable avec les ressources computationnelles à notre disposition. Cependant, elle est loin d'être aussi efficace que les meilleures solutions existantes, et des changements peuvent être effectués pour obtenir de meilleures performances au détriment de la vitesse de calcul, comme mentionné précédemment, par l'utilisation de DeepSORT au lieu de SORT, et l'utilisation de la version 8 complète de YOLO au lieu de sa version light.

Pour l'évaluation, nous utiliserons les métriques HOTA et HOTA(0) que nous décrirons plus tard et qui sont implémentées dans un outil nommé TrackEval<sup>4</sup>.

---

1. <https://github.com/ultralytics/ultralytics>

2. <https://github.com/abewley/sort>

3. <https://medium.com/@mosesdaudu001/object-detection-tracking-with-yolov8-and-sort-algorithm-363b>

4. <https://github.com/JonathonLuiten/TrackEval>

## 5 Présentation des résultats de validation

### 5.1 Présentation du dataset utilisé

Pour mesurer nos performances, nous utiliserons le dataset MOT20, qui est un benchmark bien connu dans le domaine du suivi d'objets multiples. La partie entraînement est composée de quatre vidéos dont nous possédons le ground truth, composé de piétons en mouvement.

### 5.2 Description des hyperparamètres

Notre implémentation possède quatre hyperparamètres au total : un pour la détection et trois pour le suivi :

- conf\_yolo : Ce paramètre correspond au seuil de confiance pour les détections générées par le modèle. Il définit la confiance minimale que le modèle doit avoir dans une prédiction pour que la détection soit incluse dans le processus de suivi. Un seuil de confiance plus élevé réduit le nombre de fausses détections (diminuant les faux positifs), mais peut aussi manquer des détections valides, surtout pour des objets difficiles à détecter ou partiellement occlus.
- max\_age : Représente le nombre maximal de frames consécutives pendant lesquelles un objet peut être non détecté (totalement occulté ou hors champ) avant que le tracker ne décide d'abandonner le suivi de cet objet. Un max\_age élevé permet de maintenir l'identité des objets même en cas d'occlusions ou de disparitions temporaires, mais peut aussi conduire à conserver des identifiants pour des objets qui ne sont plus dans la scène, comme discuté précédemment.
- min\_hits : Spécifie le nombre minimum de frames consécutives dans lesquelles un objet doit être détecté avant que le tracker ne commence à lui attribuer une trace officielle ou un identifiant. Cela aide à filtrer les fausses détections sporadiques qui peuvent survenir dans un seul frame. Un min\_hits plus élevé rend le système plus robuste face aux fausses alarmes, mais peut retarder l'initialisation du suivi pour de nouveaux objets, potentiellement en manquant de courtes apparitions valides.
- iou\_threshold : Utilisé pour déterminer si deux boîtes englobantes se chevauchent suffisamment pour être considérées comme le même objet dans le processus d'association des données. Il définit le seuil minimal de l'Intersection over Union pour accepter qu'une boîte prédite et une boîte détectée correspondent au même objet. Un seuil d'IoU plus bas peut augmenter le risque d'associations incorrectes, et un seuil trop élevé peut entraîner des échecs à associer correctement les boîtes à leurs objets en cas de changements rapides ou de mouvements.

### 5.3 Résultats avec les paramètres initiaux

Nous avons initialement choisi les hyperparamètres arbitrairement, en nous basant sur ceux utilisés par l'implémentation décrite dans l'article du site medium.com.

- conf\_yolo = 0.5
- max\_age = 20
- min\_hits = 1
- iou\_threshold = 0.3

	HOTA	LocA	DetA	AssA	HOTA(0)
MOT20-01	31.956	84.229	19.061	53.738	37.973
MOT20-02	24.007	84.815	17.15	33.721	28.349
MOT20-03	4.7259	81.842	1.7891	12.543	5.8064
MOT20-05	1.6146	85.27	0.36368	7.2248	1.8404
COMBINED	10.269	84.349	3.4036	31.103	12.095

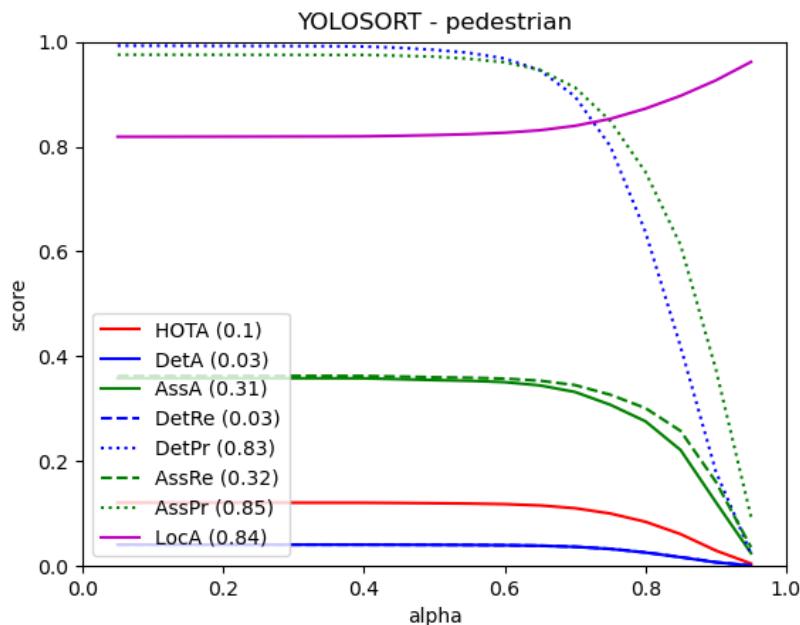


FIGURE 2 – Résultats avec nos paramètres initiaux

Voici les résultats obtenus sur notre dataset :

Notre score HOTA évalue la performance globale et est une combinaison de LocA, DetA et AssA. LocA évalue la précision des localisations effectuées, indiquant à quel point les positions estimées sont proches des positions réelles des objets. DetA évalue la précision des détections en elles-mêmes, sans prendre en compte l'association des identités. Elle mesure la capacité de l'algorithme à détecter tous les objets pertinents dans chaque trame. AssA évalue la précision de l'association des identités à travers les trames. Elle mesure la capacité de l'algorithme à maintenir les identités correctes des objets à travers les différentes trames. HOTA(0) est une mesure spécifique qui se concentre uniquement sur la précision de la localisation des objets détectés dans une seule image, sans prendre en compte leur suivi à travers le temps.

Pour chacune de ces mesures, la valeur va de 0 à 100, où 0 est un score nul et 100 un score parfait.

## 5.4 Amélioration des hyperparamètres

Pour notre recherche d'hyperparamètres, nous allons mesurer les performances en modifiant un seul hyperparamètre à la fois, tout en conservant les valeurs initiales pour les autres. Nous avons limité cette approche à la première vidéo en raison des contraintes de temps de calcul.

Voici les valeurs testées pour chaque hyperparamètre :

- conf\_yolo = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9
- max\_age = 5, 10, 15, 20, 25, 30, 35
- min\_hits = 1, 2, 3, 4
- iou\_threshold = 0.1, 0.2, 0.3, 0.4, 0.5

Voici la variation du score HOTA en fonction de la valeur de chaque hyperparamètre :

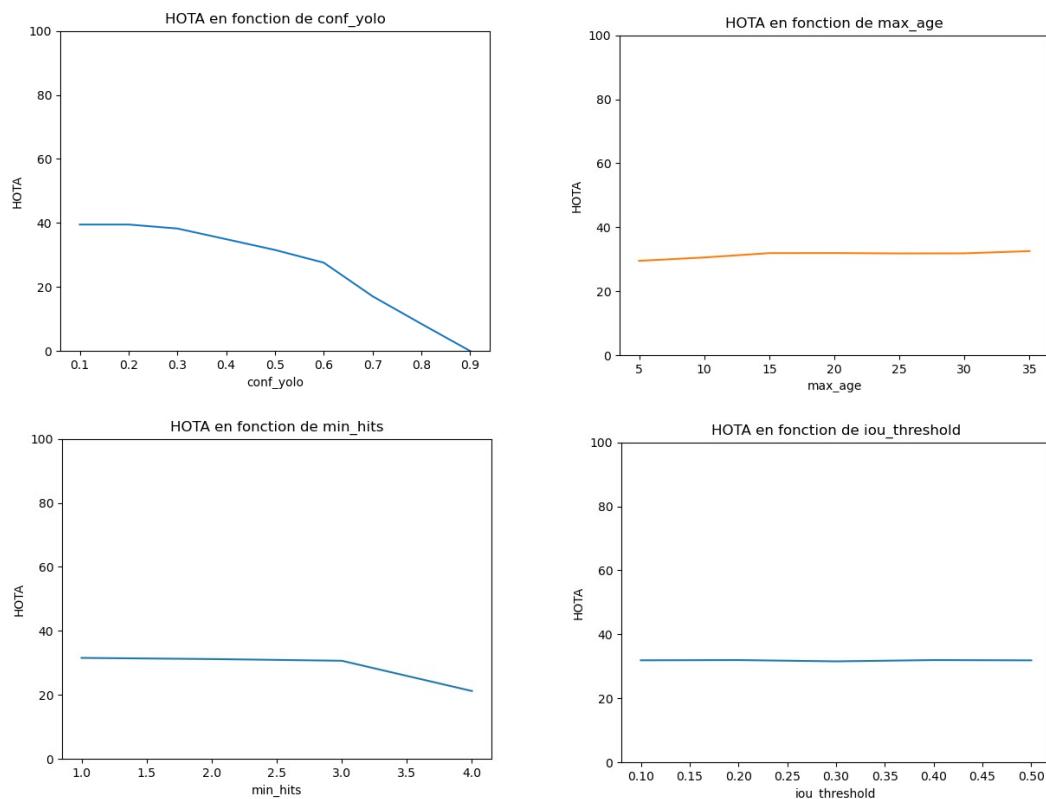


FIGURE 3 – HOTA de la valeur des différents hyperparamètres

Nous observons que la variation du score HOTA est relativement faible en ce qui concerne le max\_age et l'iou\_threshold. Cependant, la performance diminue lorsque le min\_hits est trop élevé, et nous notons également une baisse des performances à mesure que conf\_yolo augmente. En conclusion, nos hyperparamètres initiaux étaient généralement adéquats, et nous allons les conserver, à l'exception de conf\_yolo que nous réduirons à 0.3 pour optimiser les résultats.

## 5.5 Résultats avec les hyperparamètres améliorés

Voici nos résultats après avoir optimisé nos hyperparamètres :

	HOTA	LocA	DetA	AssA	HOTA(0)
MOT20-01	38.263	82.591	28.139	52.311	47.002
MOT20-02	29.118	83.301	24.636	34.627	35.403
MOT20-03	12.375	79.178	7.8745	19.521	16.043
MOT20-05	5.9957	83.36	3.0953	11.722	7.0518
COMBINED	14.415	82.113	7.8485	26.721	17.596

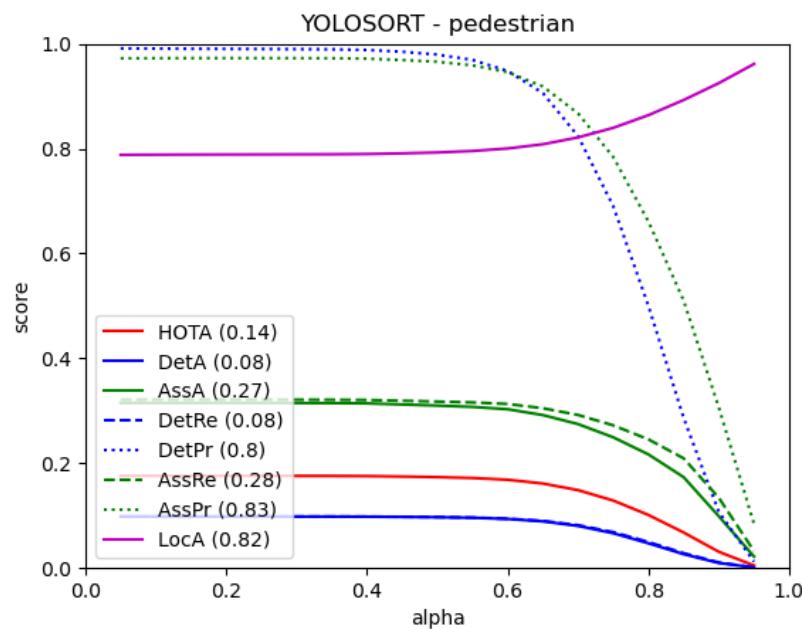


FIGURE 4 – Résultats avec nos paramètres mieux choisis

Ce que nous remarquons rapidement, c'est que nous obtenons de très mauvais résultats sur les vidéos MOT20-3 et MOT20-5. La principale différence de ces deux vidéos par rapport aux deux premières est qu'elles ont été prises le soir avec un éclairage très sombre et très fluctuant, en raison des lampadaires.

Lorsque nous examinons la moyenne des scores, nous observons que le score général est plutôt faible (14.4 sur 100). Cette performance médiocre est principalement due aux faibles scores de DetA et AssA, tandis que le LocA est très bon. Cela signifie que notre implémentation est très efficace pour localiser les objets — elle positionne bien les boîtes englobantes — mais elle est assez mauvaise pour détecter les objets, en manquant beaucoup, et elle a également des difficultés à suivre les objets de manière continue.

Enfin, nous remarquons que le score moyen de HOTA(0) est meilleur que le HOTA moyen, ce qui indique que nous avons une bonne localisation mais un suivi déficient. Le problème semble donc provenir davantage du tracker que du détecteur, même si, compte tenu de la valeur obtenue pour DetA, le détecteur n'est pas exempt de responsabilité.

## 6 Résultats sur le dataset moodle

Après la mise en place de la solution sur le dataset Moodle, plusieurs observations ont été faites :

La majorité des difficultés initialement imaginées ont été résolues, que ce soit les difficultés liées aux occlusions partielles ou aux sorties partielles du champ de la caméra, à la rotation de notre objet, à la sortie totale de l'objet du champ de la caméra ou à une occlusion totale, ainsi qu'au changement du background dû aux mouvements de la caméra.



FIGURE 5 – Détection de la tasse en fond malgré le mouvement de la caméra, la mauvaise mise au point, et l'occlusion partielle

Cependant, la difficulté liée à la transparence du verre n'a été que partiellement résolue. Notre solution est certes capable de détecter le verre malgré sa transparence, qui nous permet de voir ce qu'il y a à l'intérieur et derrière, mais cela reste instable avec des frames où un verre n'est plus détecté. De plus, la détection échoue totalement lorsque le verre est pris dans la main.



FIGURE 6 – Détection des verres transparents immobiles



FIGURE 7 – Perte de la détection d'un verre lorsqu'elle est prise dans la main

Enfin, nous avons rencontré un nouveau problème que nous n'avions pas initialement envisagé : l'apparition de faux positifs, avec une boîte à sachets de thé qui a été détectée comme étant un verre. Cela est dû à un seuil de détection trop bas pour notre détecteur, mais sans cela, d'autres verres n'auraient pas été détectés.



FIGURE 8 – Perte de la détection d'un verre lorsqu'elle est prise dans la main

Au final, notre solution fonctionne plutôt bien sur le dataset Moodle, même si elle reste assez instable avec des verres très flous en arrière-plan qui sont peu ou pas détectés, ou des verres qui perdent la détection parfois sur quelques frames. Cela laisse une grande marge pour des améliorations futures.