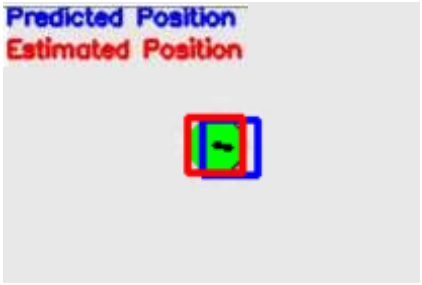
Junyi LI

Rapport ML-Based Visual Object Tracking

Exercice 1

Objectif : Implémenter le tracker d'un objet dans un espace 2D en utilisant un algorithme de détection d'objets existant et intégrer le filtre de Kalman pour un suivi lisse et précis.



Pour implémenter le filtre de Kalman, nous avons créé une classe contenant :

dt : temps pour un cycle utilisé pour estimer l'état (temps d'échantillonnage)
u_x, u_y : accélérations dans les directions x et y respectivement
std_acc : magnitude du bruit du processus
x_sdt_meas, y_sdt_meas : écarts-types de la mesure dans les directions x et y respectivement

Fonction de prédiction : predict(). Cette fonction fait la prédiction de l'estimation de l'état x^k et de l'erreur de prédiction Pk. Cette tâche appelle également le processus de mise à jour du temps (u) car il projette l'état actuel à l'étape de temps suivante.

Fonction de mise à jour : update(). Cette fonction prend les mesures zk en entrée (coordonnées x,y du centroïde des cercles détectés)

Résultats :
Nous pouvons remarquer le déplacement de la prédiction (bleu) comparé à la position actuelle (rouge).

Exercice 2

Objectif : Développer un tracker IoU simple sans utiliser d'information d'image. Étendre l'algorithme pour gérer le suivi d'objets multiples simultanément.

Etapes :
Charger les détections à partir d'un fichier texte
Calculer le score de similarité IoU pour chaque paire de boîtes
Associer les détections aux pistes de manière gloutonne en utilisant un seuil sigma_iou
Gérer les listes de correspondances, de détections non appariées et de tracks non appariées
Développer une interface pour vérifier les résultats du suivi.
Pour la fonction de prédiction, une simple différence de mouvement entre la frame actuelle et la frame précédente a été calculée.

Résultats :
Le suivi des individus est assez satisfaisant visuellement. Cependant, trop d'objets sont créés à cause de la superposition des boîtes sépare les tracks. Le nombre d'objets peut être réduit en augmentant le nombre de frames à garder avant suppression ou en diminuant sigma_iou afin de garder les tracks au lieu d'en recréer d'autres.

Le suivi IoU est une méthode simple mais efficace pour suivre des objets dans une séquence d'images. Il permet de créer et de mettre à jour des tracks en fonction de la superposition des boîtes englobantes. Il peut être utilisé pour le suivi d'objets multiples en attribuant des identifiants uniques aux objets. Cependant, la superposition des boîtes est un point faible de cette méthode.

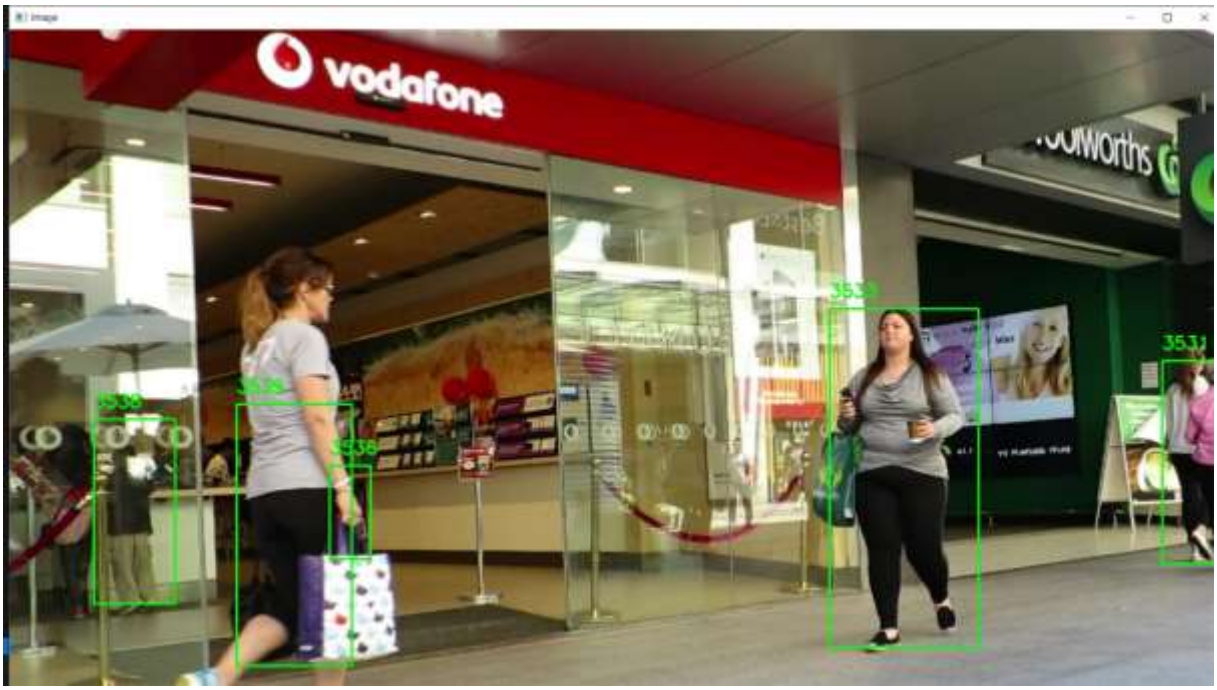
Exercice 3

Algorithme hongrois est un algorithme d'optimisation qui permet de trouver l'assignation optimale entre deux ensembles de données, en minimisant le coût total ou en maximisant le bénéfice total. Il est utilisé pour résoudre le problème d'association des objets entre deux images successives, en utilisant la matrice de similarité (IoU) comme critère de coût ou de bénéfice.

Exercice 4

Le filtre de Kalman de l'exercice 1 a été implémenté à notre suivi IoU. Sa fonction predict et update sont appelés lors des étapes.

Résultats :
Les résultats visuels sont satisfaisants. Le filtre de Kalman est beaucoup plus sensible aux mouvements, détectant plus d'objets subtils mais créant aussi de nombreux objets parasites. Augmenter sigma_iou réduit la sensibilité.



Exercice 5

Objectif : Étendre le suivi multi-objets (MOT) basé sur l'IoU (Intersection over Union) en utilisant des informations visuelles et d'autres améliorations.

Le modèle pré-entrainé de mobilenet_v2 a été utilisé pour capturer les features des images. Le résultat a été multiplié avec la matrice de similarité de l'IoU en utilisant une cosine similarity pour profiter des deux méthodes. Par soucis d'optimisation du code, il n'est que lancé sur le début de la vidéo.