# Traitement d'images

## Cours 5 - Détections et suivi d'objets

## Table des matières

# Introduction

La détection d'objets et le suivi d'objets sont des tâches importantes dans la vision par ordinateur. La détection d'image consiste à identifier et localiser si des objets spécifiques sont présents dans une image ou une séquence vidéo. Le suivi d'objets, consiste à suivre le mouvement d'un objet qui a été détecté dans une séquence d'images.

Les approches sont nombreuses et variées et leur invention et développement est **long**. Nous allons donc nous concentrer sur la découverte de quelques concepts clés de détection et de suivi d'objets mais également les utiliser.

# Bounding box

La détection d'un objet sera représentée par une « bounding box » qui est un rectangle qui entoure l'objet. Une « bounding box » est généralement définie par 4 paramètres, de deux manières différentes, la première :

- La coordonnée $x$ du coin supérieur gauche de la boîte
- La coordonnée $y$ du coin supérieur gauche de la boîte
- La largeur $w$ de la boîte
- La hauteur $h$ de la boîte

La deuxième :

- La coordonnée $x_{min}$ du coin supérieur gauche de la boîte
- La coordonnée $y_{min}$ du coin supérieur gauche de la boîte
- La coordonnée $x_{max}$ du coin inférieur droit de la boîte
- La coordonnée $y_{max}$ du coin inférieur droit de la boîte

Généralement, une « bounding box » doit être aussi proche que possible des bords de l'objet qu'elle entoure, tout en restant suffisamment grande pour inclure l'ensemble de l'objet.

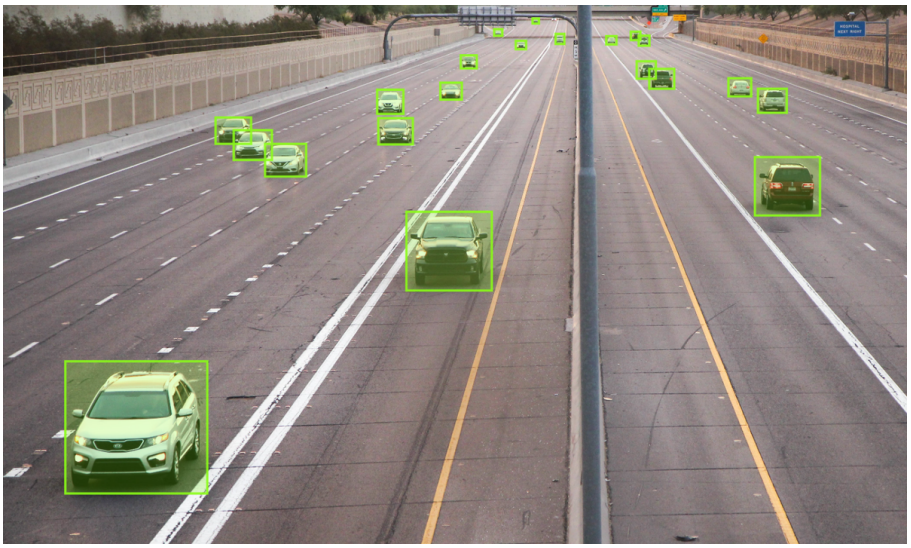

FIGURE 1 – Exemple de « bounding boxes » entourant des objets d'intérêt dans une image, ici des voitures.

Le concept de « bounding box » est largement utilisé, il est néamoins limité dans certains cas, comme par exemple quand l'orientation des objets n'est pas alignée avec les axes de l'image comme illustré dans la figure 2.

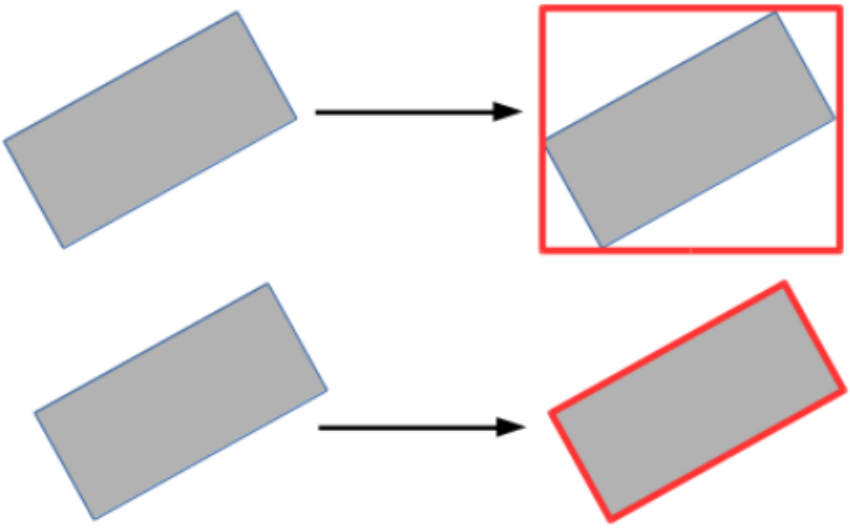

FIGURE 2 – Une limitation des « bounding boxes », dans l'image supérieure, la solution possible. Dans l'image inférieuse la solution idéale.

? Comment pourriez-vous faire pour adresser le problème illustré dans la figure 2 ?

# Fonctionnement général

## Détection d'objets

La détection d'objets prend en entrée une image

1. **Prétraitement de l'image** : Cette étape peut inclure des opérations telles que le redimensionnement, la normalisation des couleurs, ou la réduction du bruit pour améliorer la qualité de l'image.
2. **Extraction des caractéristiques** : Des caractéristiques pertinentes sont extraites de l'image pour aider à identifier les objets. Cela peut inclure des caractéristiques basées sur la couleur, la texture, les contours, ou des descripteurs plus avancés comme les SIFT, SURF, ou les caractéristiques apprises par des réseaux de neurones convolutifs (CNN).
3. **Classification** : Les régions d'intérêt dans l'image sont classifiées pour déterminer si elles contiennent des objets.
4. **Localisation** : Une fois qu'un objet est détecté, sa position est localisée dans l'image en utilisant des « bounding boxes » ou d'autres méthodes de localisation.
5. **Post-traitement** : Cette étape peut inclure des techniques telles que la suppression des doublons (Non-Maximum Suppression) pour éliminer les détections redondantes et améliorer la précision globale.

Les étapes 1,2 et 5 sont « optionnelles » .


FIGURE 3 – La suppression des doublons (Non-Maximum Suppression) permet de réduire le nombre de « bounding boxes » redondantes et/ou partielles.

### Suivi d'objets

Le suivi d'objet prend en entrée une image et la position d'un objet dans l'image précédente.

- **Initialisation** : Le suivi a besoin d'une position initiale, généralement fournie par une détection d'objet ou une annotation manuelle.
- **Prétraitement de l'image** : Comme pour la détection, cette étape peut inclure des opérations pour améliorer la qualité de l'image.
- **Extraction des caractéristiques** : Des caractéristiques pertinentes sont extraites de l'image pour aider à identifier et suivre l'objet. Cela peut inclure des caractéristiques basées sur la couleur, la texture, les contours, ou des descripteurs plus avancés.
- **Mise à jour de la position** : Le système utilise les caractéristiques extraites pour estimer la nouvelle position de l'objet dans l'image actuelle.
- **Gestion des occlusions et des disparitions** : Le système peut inclure des mécanismes pour gérer les occlusions temporaires ou les disparitions de l'objet.
- **Post-traitement** : Cette étape peut inclure des techniques pour affiner la position de l'objet et améliorer la précision globale du suivi.

Les étapes 2,4,5 et 6 sont « optionnelles » .

## Les difficultés du suivi d'objets

### Occlusion

Une occlusion se produit lorsqu'un objet d'intérêt est partiellement ou totalement caché par un autre objet dans la scène. Cela peut rendre le suivi difficile, car le système doit être capable de réidentifier l'objet une fois qu'il réapparaît.

### Changement d'échelle

Les objets peuvent apparaître plus grands ou plus petits dans l'image en fonction de leur distance par rapport à la caméra. Le suivi doit être capable de gérer ces changements d'échelle pour maintenir une identification précise de l'objet.

### Identity switching

L'identity switching se produit lorsque le système de suivi confond deux objets similaires et attribue incorrectement l'identité d'un objet à un autre. Cela peut se produire lorsque les objets se croisent ou se rapprochent l'un de l'autre.

### Arrêt du tracking

Le suivi d'un objet peut être interrompu pour diverses raisons, telles que la sortie de l'objet du champ de vision de la caméra, une occlusion prolongée, ou des changements drastiques dans l'apparence de l'objet. Le système doit être capable de gérer ces situations et de reprendre le suivi lorsque l'objet réapparaît.

### Exemple

Dans cette vidéo, nous pouvons observer une séquence difficile avec plusieurs personnes proches et qui se croisent.

Les trackers sont plus ou moins performants (robustes) face à ces difficultés.

## Métrique d'évaluation

### Précision et rappel

La précision et le rappel sont deux métriques couramment utilisées pour évaluer les performances des systèmes de détection et de suivi d'objets en vision par ordinateur. La précision (Precision) mesure la proportion de détections correctes parmi toutes les détections effectuées par le système. Elle est calculée comme le rapport entre le nombre de vraies positives (TP) et la somme des vraies positives et des fausses positives (FP) :

$$\text{Précision} = \frac{TP}{TP + FP} \tag{1}$$

Le rappel (Recall), quant à lui, mesure la proportion de détections correctes parmi toutes les instances réelles d'objets dans la scène. Il est calculé comme le rapport entre le nombre de vraies positives (TP) et la somme des vraies positives et des fausses négatives (FN) :

$$\text{Rappel} = \frac{TP}{TP + FN} \tag{2}$$

### F-mesure

La F-mesure (F1-score) est une métrique qui combine la précision et le rappel en une seule valeur pour évaluer les performances d'un système de détection et de suivi d'objets. Elle est particulièrement utile lorsque l'on souhaite équilibrer l'importance de la précision et du rappel. La F-mesure est calculée comme la moyenne harmonique de la précision et du rappel, selon la formule suivante :

$$F1 = 2 \cdot \frac{\text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}} \tag{3}$$

La F-mesure varie entre 0 et 1, où une valeur de 1 indique une performance parfaite, c'est-à-dire une précision et un rappel parfaits.

Nous avons vu la précision, le rappel et la F-mesure, il reste néamoins à définir ce qu'est un exemple bien classé ou non.

### L'intersection et l'union

L'intersection et l'union sont des concepts fondamentaux en mathématiques et en théorie des ensembles, et ils sont également largement utilisés en vision par ordinateur, notamment dans le contexte de la détection et du suivi d'objets. L'intersection (notée $A \cap B$) d'un ensemble $A$ et d'un ensemble $B$ est l'ensemble des éléments qui appartiennent à la fois à $A$ et à $B$. En d'autres termes, c'est la partie commune aux deux ensembles. L'union (notée $A \cup B$) d'un ensemble $A$ et d'un ensemble $B$ est l'ensemble des éléments qui appartiennent à $A$, à $B$, ou aux deux. En d'autres termes, c'est la combinaison de tous les éléments des deux ensembles, sans duplication. En vision par ordinateur, ces concepts sont souvent utilisés pour évaluer la qualité des détections d'objets. Par exemple, lorsqu'on compare une boîte englobante prédite (bounding box) avec une boîte englobante réelle (ground truth), on peut calculer l 'intersection et l'union de ces deux boîtes pour déterminer à quel point la prédiction est précise.

### IoU (Intersection over Union)

L'IoU (Intersection over Union) est une métrique couramment utilisée en vision par ordinateur pour évaluer la qualité des détections d'objets. Elle mesure le chevauchement entre une boîte englobante prédite (bounding box) et une boîte englobante réelle (ground truth). L'IoU est calculée en divisant l'aire de l'intersection des deux boîtes par l'aire de leur union. La formule mathématique de l'IoU est la suivante :

$$\text{IoU} = \frac{\text{Aire de l'intersection}}{\text{Aire de l'union}} \tag{4}$$

L'IoU varie entre 0 et 1, où une valeur de 1 indique une correspondance parfaite entre la boîte prédite et la boîte réelle, tandis qu'une valeur de 0 indique qu'il n'y a aucun chevauchement entre les deux boîtes. En pratique, une valeur d'IoU supérieure à un certain seuil (par exemple, 0,5) est souvent utilisée pour déterminer si une détection est considérée comme correcte (vraie positive) ou incorrecte (fausse positive).

## Utilisation de modèles existants

Nous allons utiliser la librairie `OpenCV` qui propose plusieurs méthodes de détection et de suivi d'objets.

Afin d'installer `OpenCV`, vous pouvez utiliser la commande suivante :

```
pip install opencv-python
```

La librairie `OpenCV` est rédigée en C++, mais dispose de « bindings » Python qui permettent de l'utiliser relativement simplement.

### Création et utilisation d'un tracker

Pour créer un tracker avec `OpenCV`, vous pouvez utiliser la fonction `cv2.TrackerMIL_create()` qui utilise la méthode MIL (Multiple Instance Learning) pour le suivi d'objets.

Voici un exemple de code :

```
import cv2
# Création du tracker MIL
```

```
tracker = cv2.TrackerMIL_create()
```

Dans ce cours nous supposerons que la séquence d'images est stockée dans un dossier et que le nom des images est cohérent et qu'il contient un numéro d'image.

Pour obtenir la liste des images dans le dossier, vous pouvez utiliser la bibliothèque `os` de Python comme suit :

```
import os
# Génére la liste d'image et trie par ordre alphabétique (i.e. numéro d'image)
images_list = sorted(os.listdir('sequences/Biker/'))

# Affiche la liste des images dans l'ordre
for images_list in images_list:
    print(images_list)
```

Dans cette librairie, un tracker doit être initialisé avec une image et une « bounding box » avant de pouvoir être utilisé pour suivre un objet dans une séquence d'images.

```
# Initialisation du tracker avec une image et une bounding box
images_list = sorted(os.listdir('sequences/Biker/img/'))
initial_frame = cv2.imread(images_list[0])

gt_bboxes = list(map(int, open('sequences/Biker/groundtruth.txt').read().strip().split('
    ,')))
# Définir la bounding box initiale (x, y, w, h)
initial_bbox = gt_bboxes[0]

# Initialiser le tracker avec l'image initiale et la bounding box
tracker.init(initial_frame, initial_bbox)
```

Une fois que cette initialisation est faite, vous pouvez utiliser la méthode `tracker.update()` pour mettre à jour la position de l'objet dans chaque image de la séquence.

```
for image_name in images_list[1:]:# Parcours des images à partir de la deuxième
    frame = cv2.imread(image_name)
    # Met à jour le tracker avec la nouvelle image
    success, bbox = tracker.update(frame)
\end{listing}

Afin de visualiser le suivi, vous pouvez dessiner la  ~bounding box~  sur chaque image
     en utilisant la fonction \texttt{cv2.rectangle()}.
\begin{lstlisting}[language=Python]
img = cv2.rectangle(img_cv2, box, (0, 0, 255), 2) # Dessine la bounding box en rouge
cv2.imshow("Tracking", img) # Affiche l'image avec la bounding box
cv2.waitKey(30) # Attend 30 ms avant de passer à l'image suivante
```