INF6804 - Vision par ordinateur

TP 2 : Segmentation vidéo

Antonin Tranchon
2096752

Nicolas Valenchon
2032097

22 mars 2021

Table des matières

Introduction

Le but de ce laboratoire est de travailler sur la détection de régions d'intérêt dans des séquences vidéos.

Nous allons comparer deux méthodes de recherche des régions d'intérêt : une méthode pour soustraction de l'arrière plan, et une par classification : la méthode Yolo.

Partie 1

1.1) Présentation des deux méthodes :

La méthode de soustraction d'arrière plan consiste à identifier des régions d'intérêt en recherchant ce qui se détache de l'arrière plan. La méthode que nous avons choisi d'implémenter consiste à calculer l'image d'arrière plan en faisant la moyenne des 5 premières images de la séquence vidéo, et ensuite en comparant chaque trames de la vidéo à cette moyenne. Les pixels qui seront différents entre la trame et l'image moyenne feront parti des objets en avant plan, et on pourra donc tracer un rectangle englobant autour d'eux.

La seconde méthode est une méthode de classification : nous avons décidé d'utiliser le détecteur Yolo (You Only Look Once). C'est une méthode de détection d'objets en temps réel, qui a permis d'obtenir de meilleurs performances qu'avec les structures précédents (R-CNN, ...). Pour une image, Yolo va diviser l'image en une grille en créant S x S cellules. Chaque cellule aura pour rôle de prédire un certain nombre de rectangles englobants, et un seuil va permettre d'éliminer les rectangles ayant une probabilité inférieure d'être un objet. De plus, Yolo est un classifieur (il va calculer un vecteur de probabilité, comme d'autres classifieurs) et permet donc d'identifier en plus de localiser les objets. Cette approche est assez novatrice, car elle ne parcourt qu'une fois l'image, ce qui lui permet d'être plus rapide que beaucoup d'autres méthodes.

1.2) Hypothèses de performance pour des cas spécifiques

Les deux méthodes ayant une approche différente, on peut s'attendre à ce que des configurations de vidéos soient plus optimales pour l'une ou l'autre.

- Pour une caméra fixe : la soustraction d'arrière plan sera probablement plus performante car elle va prendre les premières images pour calculer l'arrière plan, qui ne changera donc pas, puis elle va repérer chaque modification dessus. Le détecteur Yolo reconnaîtra les objets, mais risquera de détecter aussi des objets d'arrière plan, alors que ce n'est pas le but ici.
- Pour une caméra en mouvement : la soustraction d'arrière plan ne sera sûrement pas efficace ici. L'arrière plan va être calculé sur les premières images, mais si on déplace la caméra, alors l'arrière plan ne sera plus le même, et toutes les pixels de l'image seront détectées comme ayant changé. Il vaudra mieux utiliser le détecteur Yolo, qui détectera des objets de la même façon que si la caméra était fixe.
- Objet obstrué : si un objet est en partie caché, alors Yolo ne devrait pas être capable de le détecter correctement. Ce qui ne devrait cependant poser aucun problème à la soustraction d'arrière plan, car elle ne se base que sur les pixels et non sur l'objet.
- Image en mauvaise qualité : si la différence de pixel existe, même si l'objet n'est pas net, la soustraction d'arrière plan réussira à le détecter. En effet, la méthode étant purement

mathématiques, elle ne cherche pas à trouver un objet, mais juste une différence entre l'arrière plan et ce qui est affiché. Le détecteur Yolo aura plus de difficulté à reconnaître une modification, car c'est un classifieur qui recherche donc des objets sur lequel il est entraîné, une mauvaise qualité fera perdre ses caractéristiques à l'objet.

— Arrière plan en mouvement : Si l'arrière plan est changé, alors la soustraction d'arrière plan ne fonctionnera probablement pas, même si la caméra est fixe. Comme l'arrière plan est calculé sur les premières images, il faut qu'il ne soit pas modifié ensuite. Par exemple dans le cas d'une vidéo filmé à l'intérieur d'une voiture, devant une fenêtre. La vue en arrière plan va évoluer ce qui changera ce dernier. Il vaudra mieux utiliser le détecteur Yolo alors.

1.3) Description des expériences, bases de données et critères d'évaluation

Nous allons effectuer un traitement image par image d'une sélection de séquences vidéos ayant des caractéristiques différentes afin de comparer les deux méthodes.

Pour chaque image, on va calculer les zones d'intérêt avec la méthode de soustraction d'arrière plan et la classification Yolo, puis venir récupérer les coordonnées des régions d'intérêt identifiées dans le fichier *groundtruth* associé à l'image.

Afin de comparer les différentes approches, nous avons décidé de calculer un score entre les régions d'intérêt du groundtruth avec celles obtenues avec la soustraction d'arrière plan, puis celles obtenues avec le détecteur Yolo. Pour obtenir le score entre deux tableaux de régions d'intérêt, on parcourt l'ensemble de ces régions, deux à deux, et on calcule le rapport entre l'intersection et l'union. En sommant chacun de ses rapports, puis en le divisant par le nombre de rectangles englobants obtenues avec la méthode analysée (soustraction d'arrière plan ou détecteur Yolo) afin de pénaliser une méthode qui donnerait trop de rectangles englobants, on obtient le score entre les tableaux de régions d'intérêt entre deux images. On peut alors comparer les deux méthodes en calculant le score image par image, en tracer un graphique et les sommer pour obtenir le score total : plus le score est élevé, plus la méthode a bien détecté des régions d'intérêt équivalent à celles dans la Groundtruth.

Pour réaliser nos expériences, nous avons utilisé la banque de données CDNET. Pour pouvoir expérimenter nos différentes hypothèses, nous avons choisi certains jeux en particulier. Nous allons utiliser les jeux de données suivants :

— Pedestrians : c'est une caméra fixe filmant une rue où quelques piétons vont passer. Il y a un vélo dans le décor. On s'attend à ce que la soustraction d'arrière plan détecte uniquement le piéton, alors que Yolo détectera le piéton mais aussi les objets dans l'arrière plan.

— ZoomInZoomOut : la caméra est initialement placé sur un détail de l'image, puis elle fait un dézoom. On aperçoit alors le lieu en global, et une personne vient marcher dans le plan. On s'attend à avoir de très mauvais résultats pour la soustraction d'arrière plan, car c'est l'image initiale (zoomée) qui va être considérée comme arrière plan. Cependant Yolo devrait fournir des résultats corrects, si la qualité de l'image est suffisante.

— Canoë : L'arrière plan est ici une surface d'eau avec des légères vagues : il n'est donc pas parfaitement statique. Un canoë passe dans le plan. On s'attend à ce que Yolo soit plus performant, car il ne sera pas impacté par les mouvements de l'arrière plan. A l'inverse, la soustraction d'arrière plan ne devrait pas produire de très bons résultats.

— Office : C'est encore une caméra fixe dans un bureau. Une personne rentre mais est en partie obstruée par les meubles. On s'attend à ce que la soustraction d'arrière plan soit plus performante, car Yolo pourrait avoir plus de difficultés à retrouver l'objet.

1.4) Description des deux implémentations utilisées

Calcul et affichage des rectangles englobants

Les images de Grountruth, ainsi que celles obtenues par soustraction d'arrière plan, ne donnent pas directement des régions d'intérêt, mais plutôt des images binaires distinguant les pixels appartenant à l'avant plan. Pour rendre cela comparable avec Yolo et mettre en application notre métrique expliquée précédemment, il a fallu implémenter une fonction qui permette d'obtenir les rectangles englobants à partir d'une image binaire. Pour cela, nous avons utilisé une implémentation proposée dans ce lien `https://stackoverflow.com/questions/13887863/extract-bounding-box-and-save-it-as-an-image`. Elle se base sur les contours des objets (obtenus avec la librairie *cv2*).

La soustraction d'arrière plan

Pour implémenter la soustraction d'arrière plan, nous avons utilisé le code *TemporalAvgBGS.py* fourni dans le cours dans le lien suivant : `https://github.com/gabilodeau/INF6804/blob/master/TemporalAvgBGS.ipynb`. Pour une séquence vidéo, on va calculer ce qui sera considéré comme l'arrière plan. On commence déjà par charger les 5 premières trames de la vidéo en nuance de gris puis on calcule l'intensité moyenne de chaque pixel. Pour identifier l'avant plan sur une trame, on calcule la différence entre l'image en nuance de gris que l'on souhaite analyser et l'arrière plan moyen. On choisit un seuil qui permet d'identifier si le pixel fait partie de l'avant plan : nous avons choisi ce paramètre n à 40 car il permettait d'obtenir de bons résultats (il détectait suffisamment de pixels modifiés pour que l'on obtienne une reconstruction correcte de l'avant plan par la suite. Si le paramètre est plus faible, on en detectera trop et s'il est trop élévé, pas assez).

En prenant uniquement cette implémentation, on se retrouvait avec beaucoup de rectangles englobants assez petits, qui encadraient mal les objets à identifier et qui étaient parfois éparpillés sur toute l'image dans des zones sans objets. Nous avons donc décidé de rajouter des opérations morphologiques à notre traitement. Les opérations morphologiques sont des opérations effectuées sur des images binaires en se basant sur la théorie des ensembles. On va parcourir les pixels d'une image avec un élément structurant, et en fonction de l'opération effectuée, on va vérifier si l'intersection (ou l'union) entre l'élément structurant et le pixel et ces alentours est non nulle : on modifiera en conséquence la valeur du pixel.

Les figures 1, 2 et 3 permettent de se rendre compte de l'efficacité de certaines de ces opérations. Une fermeture (érosion suivie d'une dilatation) permet de retirer les traits noirs trop fins (voir le cercle vert), alors qu'une ouverture (dilatation puis érosion) va agrandir les traits noirs (voir le cercle rouge), ce qui va supprimer les petits espaces blancs.

Pour la soustraction d'arrière plan, l'algorithme obtenait beaucoup de petites zones. Nous avons donc d'abord appliqué une fermeture avec un élément structurant très petit. Cela permettait de relier certains points qui étaient proches (et d'éviter d'avoir trop de suppressions à l'étape suivante). On a ensuite fait une ouverture avec un élément structurant moyen : cela permettait de supprimer tous les points éparses considérés comme de l'avant plan mais qui n'étaient pas

FIGURE 1 – Image en noir et blanc sans traitement

FIGURE 2 – Image après fermeture avec élément structurant de taille 3x3

FIGURE 3 – Image après ouverture avec élément structurant de taille 3x3

intéressants. Comme la taille de cet élément structurant est supérieure à celui de la première fermeture, les pixels qui auront été agrandis ici seront bien supprimés. Enfin, nous appliquons une nouvelle fermeture, avec un grand élément englobant, afin de relier des zones proches entre elles et obtenir un grand rectangle englobant. Toutes ces fonctions font parties de la librairie d'*OpenCV*. Les figures 4 à 19 permettent de voir l'avant plan après chaque opération.

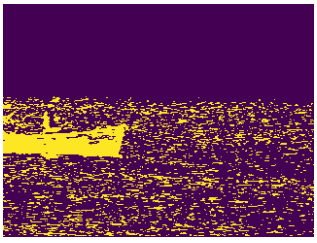

FIGURE 4 – Avant plan de l'image canoe 000891.jpg

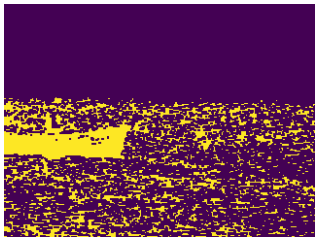

FIGURE 5 – Première fermeture sur 000891.jpg

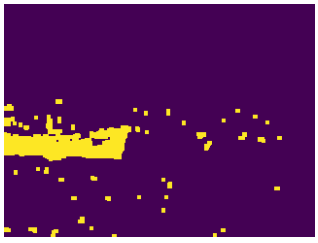

FIGURE 6 – Ouverture sur 000891.jpg

FIGURE 7 – Deuxième fermeture sur 000891.jpg

FIGURE 8 – Avant plan de l'image office 000620.jpg

FIGURE 9 – Première fermeture sur 000620.jpg

FIGURE 10 – Ouverture sur 000620.jpg

FIGURE 11 – Deuxième fermeture sur 000620.jpg

FIGURE 12 – Avant plan de l'image pedestrians 000345.jpg

FIGURE 13 – Première fermeture sur 000345.jpg

FIGURE 14 – Ouverture sur 000345.jpg

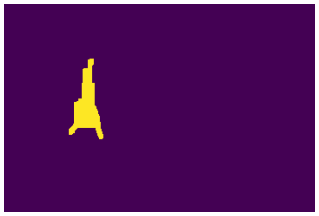

FIGURE 15 – Deuxième fermeture sur 000345.jpg

FIGURE 16 – Avant plan de l'image zoomInZoomOut 000512.jpg

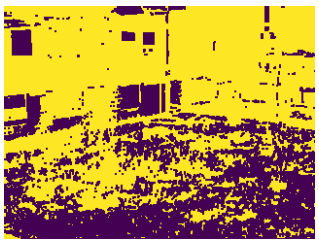

FIGURE 17 – Première fermeture sur 000512.jpg

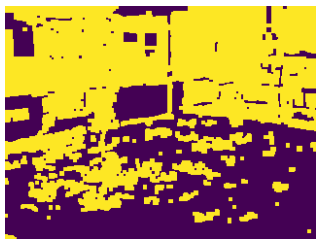

FIGURE 18 – Ouverture sur 000512.jpg

FIGURE 19 – Deuxième fermeture sur 000512.jpg

Nous avons donc obtenu un avant plan convainquant, sur lequel on peut calculer et afficher les rectangles englobants.

Le détecteur Yolov3

Nous avons utilisé l'implémentation décrite sur `https://github.com/gabilodeau/INF6804/blob/master/yolo_example.ipynb` pour implémenter le détecteur Yolov3, en prenant un modèle pré-entraîné. Nous avons conservé le paramètre $THRESHOLD = 0.3$ qui fonctionnait bien avec nos essais. Nous avons cependant fait quelques modifications pour adapter ce code à notre projet, afin de le transformer en une fonction qui renvoie les coordonnées des rectangles englobants tracés. De plus, pour rendre l'image comparable au Groundtruth, nous avons décidé de fusionner les rectangles englobant qui avaient une intersection non nulle. En effet, dans le Groundtruth, l'image étant binaire, si deux objets se touchent, alors l'extraction des rectangles englobant ne retournera qu'un seul rectangle pour ces deux objets. C'est ce que nous faisons donc avec Yolo.

Calcul du score

Comme les méthodes employées renvoient toutes les coordonnées d'un coin du rectangle englobant ainsi que sa hauteur et largeur, nous avons créé une fonction calculant le score, qui se base sur un rapport entre l'intersection et l'union de deux rectangles englobants. Pour chaque comparaison entre un rectangle obtenu par une des deux méthodes et un rectangle obtenu sur le Groundtruth, on calcule :

$$
\begin{aligned}
&rapport(rectangle_i, rectangle_j) : \\
&\qquad x1, y1, l1, h1 = rectangle_i \\
&\qquad x2, y2, l2, h2 = rectangle_j \\
&\qquad x = max(x1, x2) \\
&\qquad y = max(y1, y2) \\
&\qquad l = min(x1 + l1, x2 + l2) - x \\
&\qquad h = min(y1 + h1, y2 + h2) - y \\
&\qquad l = l \ si \ l > 0 \ sinon \ l = 0 \\
&\qquad h = h \ si \ h > 0 \ sinon \ h = 0 \\
&\qquad inter = l * h \\
&\qquad surface1 = l1 * h1 \\
&\qquad surface2 = l2 * h2 \\
&\qquad union = surface1 + surface2 - inter \\
&\qquad résultat = \frac{inter}{union}
\end{aligned}
\tag{1}
$$

On divise ensuite le score de chaque rectangle par le nombre de rectangle obtenus par la méthode à comparer plus 1 (pour ne jamais diviser par 0). Le score de la méthode pour l'image est donc :

$$
score = \frac{\sum_i^N \sum_j^M rapport(rectangle_i, rectangle_j)}{N + 1} \tag{2}
$$

Avec N le nombre de rectangles obtenus avec la méthode à comparer et M celui obtenus dans le Groundtruth.

1.5) Présentation des résultats de tests

Les figures 20 à 35 permettent de voir pour chaque jeu de données un exemple des régions d'intérêt calculées avec la soustraction d'arrière plan sans effectuer d'opérations morphologiques, avec opérations morphologiques, puis avec le détecteur Yolo. On montre aussi la région d'intérêt obtenue sur l'image Groundtruth, qui sert de référence pour le calcul du score.

FIGURE 20 – Rectangles englobants avec soustraction d'arrière plan (sans opération morphologique)

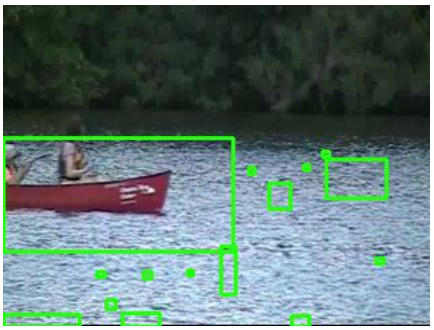

FIGURE 21 – Rectangles englobants avec soustraction d'arrière plan (avec opération morphologique)

FIGURE 22 – Détecteur Yolo

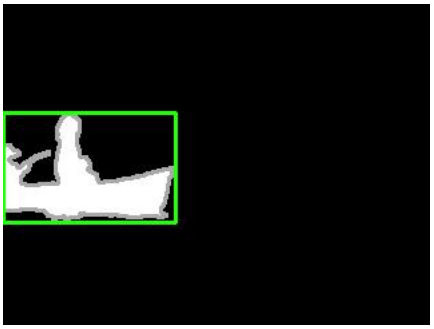

FIGURE 23 – Groundtruth

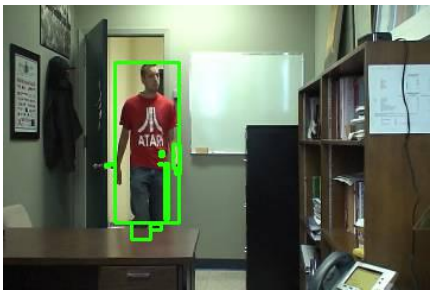

FIGURE 24 – Rectangles englobants avec soustraction d'arrière plan (sans opération morphologique)

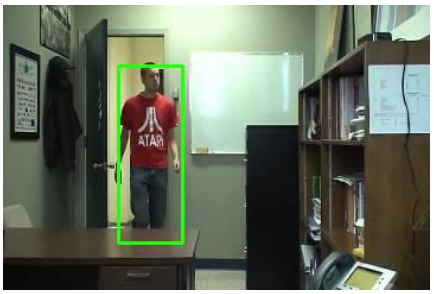

FIGURE 25 – Rectangles englobants avec soustraction d'arrière plan (avec opération morphologique)

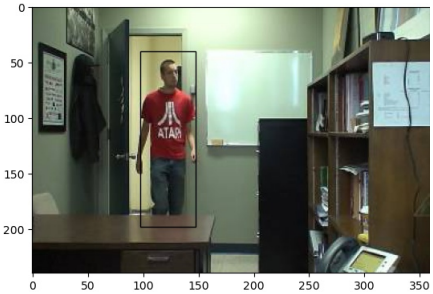

FIGURE 26 – Détecteur Yolo

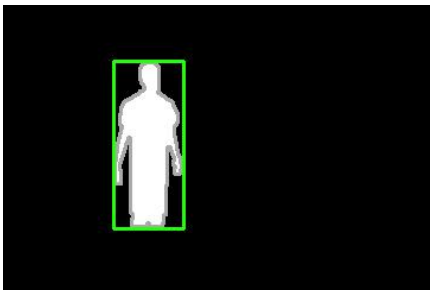

FIGURE 27 – Groundtruth

FIGURE 28 – Rectangles englobants avec soustraction d'arrière plan (sans opération morphologique)

FIGURE 29 – Rectangles englobants avec soustraction d'arrière plan (avec opération morphologique)

FIGURE 30 – Détecteur Yolo

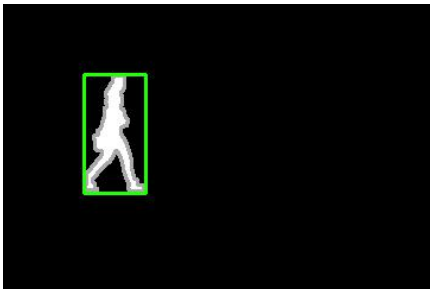

FIGURE 31 – Groundtruth

FIGURE 32 – Rectangles englobants avec soustraction d'arrière plan (sans opération morphologique)

FIGURE 33 – Rectangles englobants avec soustraction d'arrière plan (avec opération morphologique)

FIGURE 34 – Détecteur Yolo

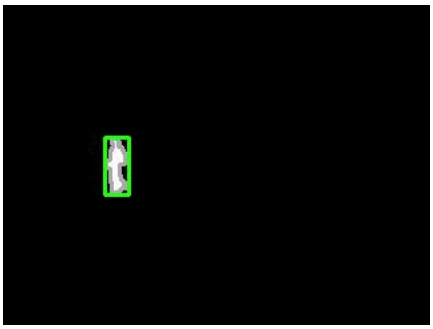

FIGURE 35 – Groundtruth

Jeu de données	Score Soustraction Arrière Plan sans opération morphologique	Score Soustraction Arrière Plan	Score détecteur Yolo	Ratio Yolo/Soustraction d'arrière plan
Canoe (Images 845 à 1000)	4.98×10^{-6}	2.03×10^{-4}	2.99×10^{-4}	1.47
Office (images 579 à 700)	8.15×10^{-4}	3.18×10^{-3}	2.04×10^{-3}	0.641
Pedestrians (images 307 à 400)	2.42×10^{-4}	3.06×10^{-3}	2.40×10^{-3}	0.783
ZoomIn-ZoomOut (images 500 à 600)	5.13×10^{-7}	3.66×10^{-5}	3.97×10^{-4}	10.835

Tableau 1 – Comparaison des scores pour chaque jeu de données

Les scores obtenus pour chaque méthode et chaque jeu de données sont affichés dans le tableau 1. Nous avons décidé d'utiliser uniquement des parties des différents jeu de données contenant des objets à détecter, et nous n'avons pas utilisé l'ensemble des datasets car le temps d'exécution devenait vite assez long, et une centaine d'image permet d'avoir un avis sur lequel des détecteurs est le meilleur dans cette situation.

Enfin, les figures 36 à 39 représentent le score obtenu à chaque trame étudiée sur une vidéo, avec en abscisse l'indice de l'image (0 étant la première image du sous dataset que nous avons sélectionné) et en ordonné le score.

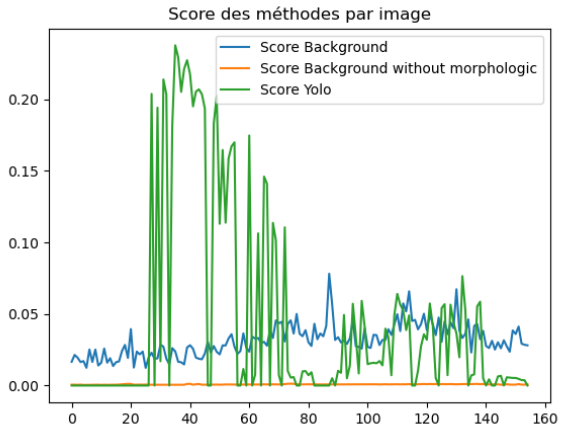

FIGURE 36 – Score par image pour le jeu de données canoe

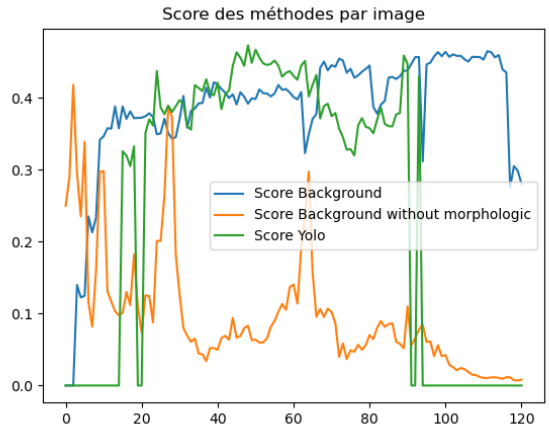

FIGURE 37 – Score par image pour le jeu de données office

1.6) Discussion des résultats et retour sur les hypothèses

Tout d'abord, on peut remarquer que les deux approches peuvent être efficaces, dépendamment du jeu de données. De même, on voit directement que les opérations morphologiques ont permis d'obtenir de biens meilleurs résultats sur la soustraction d'arrière plan. Sans ce traitement, le détecteur Yolo aurait toujours été le plus performant.

Pour le jeu de données Canoe, le détecteur Yolo est plus efficace que la soustraction d'arrière plan. En voyant les Figures 20 à 23, ce résultat était prévisible. Comme la soustraction d'arrière

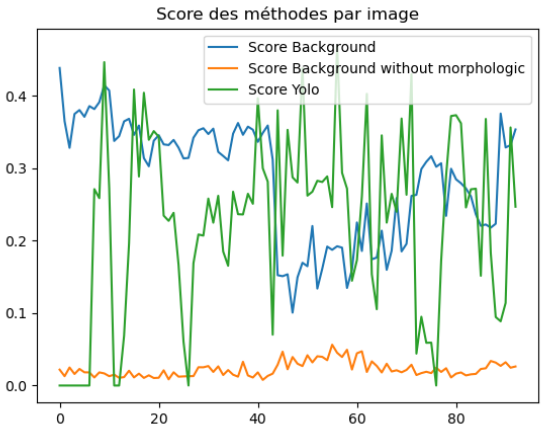

FIGURE 38 – Score par image pour le jeu de données pedestrians

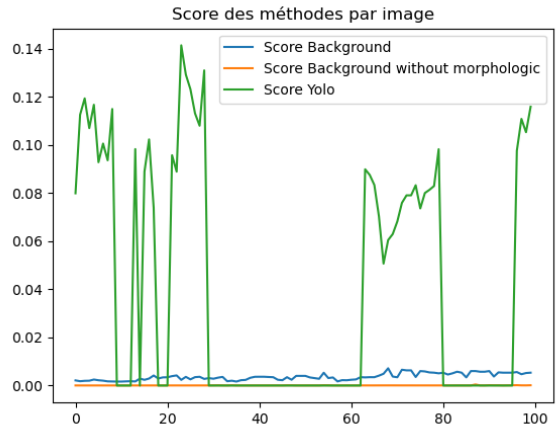

FIGURE 39 – Score par image pour le jeu de données zoomInZoomOut

plan calcule une différence d'intensités entre une moyenne calculée au début et la trame analysée. Hors, dans cette vidéo, l'arrière plan est constitué de vagues, et n'est pas fixe. La soustraction d'arrière plan va réussir à repérer le canoë qui passe dans le plan, mais il va aussi détecter des modifications de l'eau. Cela entraîne la création de plusieurs régions d'intérêt, et celle associée au canoë va être confondue avec certaines vagues (voir Figure 7). Comme le score est pénalisé si le détecteur trouve trop de régions d'intérêt, ou si l'union avec la groundtruth est trop élevée par rapport à l'intersection, cette méthode perd en efficacité. La figure 36 permet de constater que le détecteur Yolo est très efficace, mais seulement une fois que l'objet (ici le canoë) est suffisamment inclus dans l'image (on voit que le score reste longtemps à zéro).

Pour le jeu de données Office, c'est la soustraction d'arrière plan qui est la plus efficace. Comme c'est une caméra fixe, on pouvait s'attendre à ce résultat. La figure 37 permet de voir que lorsque la détection se fait bien avec Yolo, les scores sont à peu près équivalent entre la soustraction d'arrière plan (avec opération morphologique) et ce détecteur. Cependant, là où le score total de Yolo est fortement pénalisé, c'est qu'un objet apparaît mais qu'il est peu visible initialement. Comme nous l'avions supposé, Yolo arrive moins bien à détecter des objets obstrué, donc il ne les classifiera pas et ne créera donc pas de région d'intérêt. Dans le jeu de données Office, une personne passe devant la caméra : il apparaît progressivement, et certaines parties du corps sont obstruées. Le score total du détecteur Yolo est donc pénalisé sur ces premières trames, mais il reste très efficace avec une partie de la personne obstrué.

Pour le jeu de données Pedestrians, la soustraction d'arrière plan a été plus efficace. C'est encore une fois une caméra fixe. On peut voir sur la figure 38 que le détecteur Yolo a son score qui varie beaucoup. Ce détecteur perd de son score, car dans l'arrière plan il y a un vélo qui va être classifié par le détecteur. Mais le groundtruth ne le reconnaît pas comme région d'intérêt, car il fait partie de l'arrière plan, donc Yolo va être pénalisé en dessinant cette région d'intérêt. Comme ce vélo ne bouge pas et est présent dès les premières trames, il n'est pas vu comme région d'intérêt avec la méthode de soustraction d'arrière plan.

Le jeu de données zoomInZoomOut est une vidéo qui part d'une image zoomée, puis la caméra dézoom. Le plan de vue global se dévoile donc au fur et à mesure, puis quelqu'un passe dans le plan. Cette méthode n'est donc pas du tout adaptée par la soustraction d'arrière plan, car ce qui est considéré comme arrière plan est la moyenne des 5 premières images, qui sont ici

les images zoomées. Dès que l'image est dézoomée, la soustraction d'arrière plan ne peut plus fonctionner, et les régions d'intérêt tracées n'ont aucun sens (notamment avant les opérations morphologiques, figure 32). Parfois de très grandes régions vont être tracées, et vont englober la région du groundtruth a trouver. Cependant, l'union des deux régions va être largement supérieur à l'intersection, donc le score obtenu va être très faible. La figure 39 montre bien que le détecteur Yolo est plus efficace ici. On remarquera aussi que Yolo n'est pas non plus parfait, et que sur la Figure 34 le rectangle englobant de la personne est mal dessiné. Parfois certaines images de cette trame ne retournent même pas de rectangle englobant sur la personne. Cela est très probablement du à la mauvaise qualité de la zone contenant la personne. En effet, au plus l'objet à détecter est petit, au plus la résolution au sein du rectangle est faible. C'est pour cela que Yolo n'est pas parfait ici.

On a pu remarquer que la soustraction d'arrière plan ne cherche pas à trouver des objets dans l'image, mais fait seulement des calculs entre une image moyenne et l'image en cours de traitement. S'il aperçoit une différence, il va tracer un rectangle englobant, même si l'objet ne pourrait pas être classifié par un détecteur. Il faut donc que les cinq premières images représentent vraiment ce que sera l'arrière plan dans les trames suivantes, sinon tout le traitement sera faussé. À l'inverse, bien que cela ne soit pas le but dans ce laboratoire, Yolo va chercher à classifier les objets. Il pourra donc tracer des rectangles englobants uniquement s'il reconnaît un objet sur lequel il a été entraîné. Il sera invariant aux changements d'échelle (sauf si la qualité est trop réduite), mais il aura besoin de trouver des objets entiers pour les détecter. De plus, il ne sait pas faire la différence entre un objet d'arrière plan et d'avant plan, mais comme les objets d'arrière plan sont souvent assez petit, ils ne sont pas détectés par Yolo et le rendent donc assez efficace pour cette tache. Les deux méthodes étaient relativement rapides à s'exécuter, mais la méthode de soustraction d'arrière plan était la plus rapide des deux.

Conclusion

Ce laboratoire nous a permis de tester deux méthodes de détection de région d'intérêt. Tout d'abord, la soustraction d'arrière plan, qui calcule une différence d'intensité de pixels. Si cette méthode comme telle n'était pas concluante, couplée à d'autres outils de traitement d'images, elle permettait d'avoir de bien meilleurs résultats. Le détecteur Yolo permettait d'avoir des résultats bon lorsqu'il est capable de détecter un objet sur lequel il est entraîné, et s'il est bien visible sur l'image, même si cela n'implique pas forcement que l'objet en question fasse parti de l'avant plan.

Références

— Code du cours sur la soustraction d'arrières plans : `https://github.com/gabilodeau/INF6804/blob/master/TemporalAvgBGS.ipynb`
— Dessin des rectangles englobants : `https://stackoverflow.com/questions/13887863/extract-bounding-box-and-save-it-as-an-image`
— Implémentation de Yolo v3 : `https://github.com/gabilodeau/INF6804/blob/master/yolo_example.ipynb`
— Jeu de données CDNET : `http://changedetection.net/`