Rapport Projet de Fin d'étude Master 2

Option : TSI - IMOVI

Suivi mono-caméra d'objets (Personnes)

Réalisé par :

Mohamed Ameziane TOUIL
Abdeslam DJEMAI

Encadré par :

Mr. Rabah IGUERNAISSI

2020-2021

Table des matières

Table des figures

Liste des sigles et abréviations

Yolo : You Only Look Once.

SVM : Support vector machine.

LBP : Local Binary Patterns.

EOH : Edge Orientation Histogramme.

HOG : Histogramme Orienté du Gradient.

ROI : Region Of Interest.

RNA : Réseau de neurones artificiels.

CA : Contour Actif.

IOU : intersection over union

Résumé

Les techniques de suivi d'objets sont de plus utilisés dans la vision par ordinateur. Elles ont comme objectif principale de générer les trajectoires des objets et peuvent intervenir dans le cadre d'une seule caméra ou d'un réseau de caméras. Dans le suivi nous cherchons à assigner une identité unique à chaque objet présent dans la scène.

Les algorithmes d'apprentissage automatique sont souvent utilisés pour la détection des objets présents dans la scène vidéo à traiter. Les réseaux de neurones convolutifs sont parmi de ses algorithmes qui nous permettent de détecter les objets dans une scène, en identifiant des cadres délimitant (bounding boxes, en Anglais) l'objet présent dans le frame que nous pouvons utiliser comme outil d'identification avant d'effectuer le suivi. L'étape de suivi d'objets consiste à assigner une identité unique à un objet détecté dans les premiers frames de la scène. Plusieurs algorithmes sont proposés dans ce contexte. Les filtres de Kalman et les filtres particulaires sont les plus connus dans la prédiction des trajectoires futures des objets détectés dans les prochains frames d'une scène vidéo.

Dans le cadre de ce projet de fin d'études, nous avons comme objectif d'implémenter les solutions qui nous permettent d'effectuer un suivi de personnes dans une scène vidéo. Plus précisément, nous allons combiner les deux méthodes de suivi (filtres de Kalman et filtres Particulaires) avec un réseau de neurones convolutif(Yolo, MobileNet) pour la détection, ce qui nous permettra de faire une comparaison pour voir lequel s'adapte le mieux. De plus nous allons adapter un autre algorithme de détection (le SVM) que nous allons associer à un filtre particulier pour le suivi et voir la différence dans les résultats.

Mots clés : Détection d'objets, Suivi Mono-caméra d'objets, Réseaux de neurones artificiels, Yolo, MobileNet, Machine à vecteurs de support, Filtre de Kalman, Filtre Particulaire.

Abstract

Object tracking techniques are increasingly used in computer vision. Their main purpose is to generate the trajectories of objects, and can be used in the context of a single camera or a network of cameras. In tracking we look to assign a unique identity to each object present in the scene.

Machine learning algorithms are often used to detect objects in the video scene that will be processed. Convolutional neural networks are among those algorithms that allow us to detect objects in a scene, identifying bounding boxes that we can use as an identification tool before tracking. The object tracking step consists in assigning a unique identity to an object detected in the first frames of the scene. Several algorithms are proposed in this context. Kalman filters and particle filters are the best known in predicting the future trajectories of objects detected in the next frames of a video scene.

In this project, we aim to implement solutions that allow us to track people in a video scene. More precisely, we will combine the two tracking methods (Kalman filters and Particle filters) with a convolutional neural network (Yolo, MobileNet) for detection, which will allow us to make a comparison to see which one fits best. Furthermore we will adapt another detection algorithm (the SVM) that we will associate to a particular filter for tracking and see the difference in the results.

Keywords : Object detection, Mono-camera tracking, Artificial Neural Networks, Yolo, MobileNet, Support Vector Machine, Kalman Filter, Particle Filter.

Chapitre 1

Etat de l'art

1.1 Introduction

La détection d'objets consiste simplement à identifier et localiser tous les objets connus dans une scène [Mikhaylov et al, 2014, [MSME14]]. La détection d'objets dans les vidéos consiste à vérifier la présence d'un objet dans des séquences d'images et éventuellement le localiser précisément pour sa reconnaissance [Sethi et al, 2011, [SA11]].

Le suivi des objets a pour but de surveiller les changements spatiaux et temporels d'un objet au cours d'une séquence vidéo, y compris sa présence, sa position, sa taille, sa forme, etc. Cela se fait en résolvant le problème de correspondance temporelle, le problème de la correspondance de la région cible dans des images successives d'une séquence d'images prises à des intervalles de temps rapprochés.

Ces deux processus sont étroitement liés car le suivi commence généralement par la détection d'objets, tandis que la détection répétée d'un objet dans la séquence d'images suivante est souvent nécessaire pour aider et vérifier le suivi comme indiqué dans [Figure 1.1].

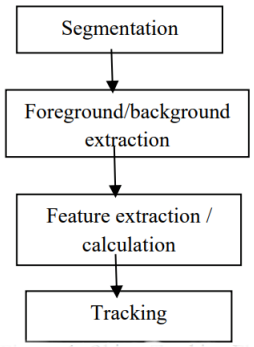

FIGURE 1.1 – Suivi d'objet [Mikhaylov et al, 2014, [MSME14]]

Plusieurs méthodes ont été proposées dans ce domaine d'étude. Nous trouvons la méthode HRIP-AC (Hybrid Region and Interest Points-based Active Contour) proposée par [Aitfares et al, [ABH+13]] qui vise à effectuer un suivi d'objets en se basant sur le Contours Actif (CA) de la personne, ROI (i.e. Région d'Intérêt), initialisé manuellement dans la première image F_0 dans une séquence vidéo F_f onnée de N images tel que $f = \{0, 1, 2, 3, \ldots, N-1\}$.

Une autre méthode pour un suivi 3D stochastique est proposée par [Boui et Marouane, 2018, [Bou18]]. Elle consiste à détecter la région d'intérêt (ROI) avec un algorithme de détection de personnes basé sur les descripteurs HOG dans les images omnidirectionnelles. La position initiale de la personne est initialisée grâce à cette détection. A partir de cette pose initiale, plusieurs positions peuvent être générées. Chaque position représente une particule respectant les possibilités de mouvement du corps humain qui permettra

de déterminer la pose de l'image suivante.

1.2 Détection d'objet

La détection de personnes dans les images statiques et les séquences vidéo est une tâche essentielle dans de nombreuses applications de vision par ordinateur. Ce problème, est l'un des plus difficiles en vision par ordinateur en raison du grand nombre de situations possibles [Boui et Marouane, 2018, [Bou18]].

Dans cette section, différentes approches de détection de personnes sont présentées. Ces méthodes utilisent principalement des approches basées sur l'apprentissage d'un classificateur pour la création d'un modèle de classe prédéfinie, utilisé par la suite dans la phase de détection.

1.2.1 Descripteur

1.2.1.1 Local Binary Patterns

Local Binary Patterns est un descripteur de caractéristiques utilisé en vision par ordinateur pour la correspondance de texture [Adrian Rosebrock, 2015, [Ros15a]]. La version originale de la caractéristique des motifs binaires locaux pour chaque pixel est basée sur un bloc de 3x3 pixels d'une image. Les pixels de ce bloc sont définis par la valeur du pixel central, multipliée par des puissances de deux, puis additionnées pour obtenir une étiquette pour le pixel central [Boui et Marouane, 2018, [Bou18]]. Après le calcul de masque LBP de l'image, un histogramme est calculé puis normalisé à partir de ce masque [Figure 1.2] [Adrian Rosebrock, 2015, [Ros15a]].

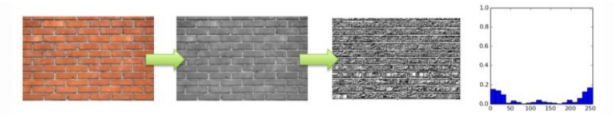

FIGURE 1.2 – Les étapes de L'algorithme LBP [Adrian Rosebrock, 2015, [Ros15a]]

1.2.1.2 Edge Orientation Histogramme

Les histogrammes orientés de bord (EOH), ont été proposés dans le but de coder les informations de silhouette et de contour, qui sont des caractéristiques importantes pour discriminer une personne dans les images [Boui et Marouane, 2018, [Bou18]]. Ces fonctionnalités permettent de conserver une invariance face aux changements globaux de luminosité, mais aussi de décrire des propriétés géométriques difficiles à capturer avec d'autres descripteurs [Boui et Marouane, 2018, [Bou18]].

1.2.1.3 Histogramme Orienté du Gradient

L'Histogramme Orienté du Gradient (HOG) est un descripteur basé sur la silhouette proposé par Dalal et Triggs dans [Dalal et Triggs, 2005, [DT05]] pour la détection des personnes. L'extraction des caractéristiques est plus complexe que dans les Histogrammes d'orientation du bord, améliorant les performances discriminatoires du descripteur tout en assurant un certain degré d'invariance [Figure 1.3]. [Boui et Marouane, 2018, [Bou18]].

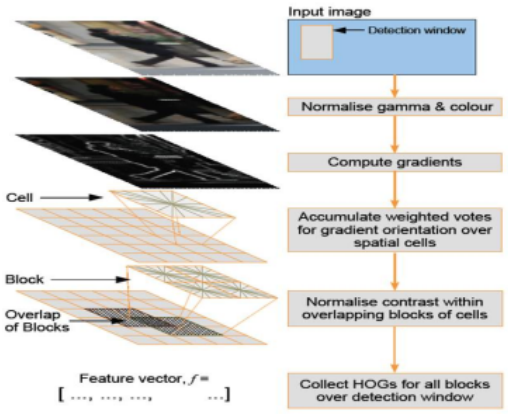

FIGURE 1.3 – Histogramme orienté du gradient pour la détection de personnes [Dalal et Triggs, 2005, [DT05]]

1.2.2 Classifieur

1.2.2.1 Machine à vecteurs de support (SVM)

Cette méthode est très couramment utilisée dans la détection des objets, notamment dans le domaine de la détection de personnes. Elle vise à déterminer un hyperplan séparateur entre les espaces des deux classes. L'idée est de maximisé la marge, c'est-à-dire la distance entre les frontières de séparation des échantillons les plus proches. Dans le cas d'un modéle non linéaire, l'algorithme transforme l'espace de représentation des données d'entrée en un espace de plus grande dimension, dans lequel il est probable qu'il existe une droite séparatrice linéaire [Figure 1.4] [Boui et Marouane, 2018, [Bou18]].

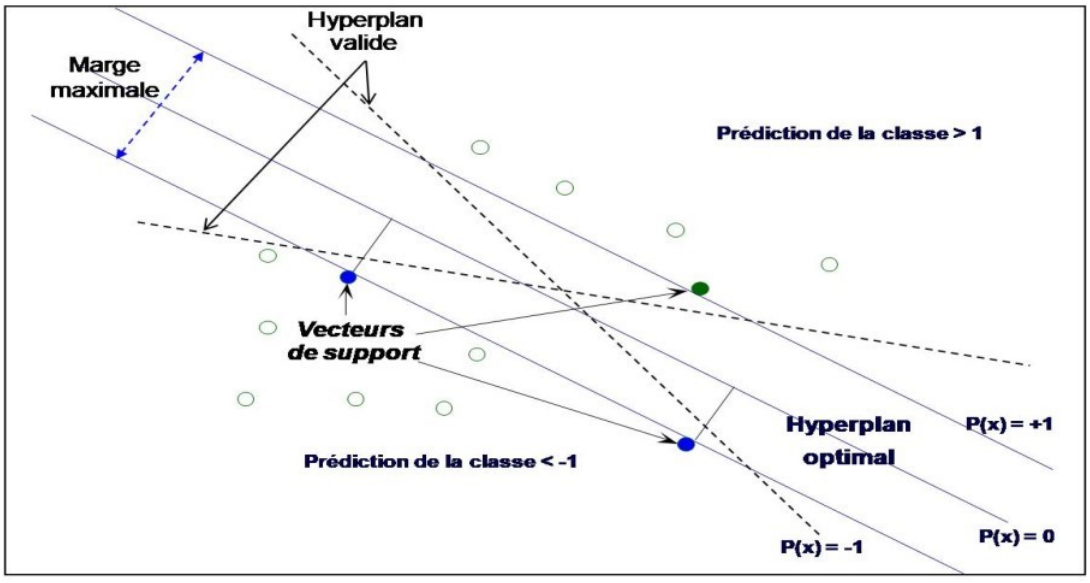

FIGURE 1.4 – SVM [Boui et Marouane, 2018, [Bou18]]

1.2.2.2 Réseau de neurones

Un réseau de neurones est un ensemble interconnecté d'éléments, d'unités ou de nœuds de traitement simples, dont la fonctionnalité est vaguement basée sur le neurone biologique. La capacité de traitement du réseau est stockée dans les forces de connexion inter-unités, ou poids, obtenues par un processus d'adaptation à, ou d'apprentissage à partir de, un ensemble de modèles d'entraînement [Simon Haykin, 2004, [Hay04]].

L'étude des réseaux de neurones artificiels est motivée par leur similitude avec le bon fonctionnement des systèmes biologiques, ce qui, en comparaison avec le système global, se compose de cellules nerveuses très simples mais nombreuses qui travaillent massivement en parallèle et (qui est probablement l'un des plus importants aspects) ont la capacité d'apprendre [Simon Haykin, 2004, [Hay04]].

1.2.2.3 You Only Look Once (YOLO)

La famille de modèle YOLO a été proposé par Joseph Redmon et al. Il s'agit d'un modèle simple de réseau neuronal convolutif qui permette des détections précises et surtout rapides [RONTEIX, 2017, [Fla17]].

Cet algorithme traite la détection d'objets comme un problème de régression, en prenant une image d'entrée donnée et en apprenant simultanément les coordonnées des boîtes englobantes et les probabilités des étiquettes de classe correspondantes [Adrian Rosebrock, 2018, [Ros18]].

1.2.2.4 MobileNet

MobileNet est un modèle proposé dans l'article [Howard et al, 2017, [HZC+17]] par des chercheurs de Google Brain en Avril 2017. L'idée derrière MobileNet comme son nom l'indique est de porter le Deep Learning sur des appareils mobiles, donc avec des contraintes de ressource mémoire, de calcul et énergétique [RONTEIX, 2017, [Fla17]].

Dans MobileNet, la convolution est remplacée par une Depthwise Separable Convolution. La Depthwise Separable Convolutions s'effectue en 2 étapes [Quantmetry, 2019, [Qua19]] :

— **Depthwise Convolution** : Consiste à appliquer un filtre sur chaque canal, contrairement à la convolution classique qui applique un filtre sur l'ensemble des canaux.
— **Pointwise convolution** : Consiste à combiner les sorties de la Depthwise Convolution

1.3 Suivi d'objet

Le suivi d'objets est le problème de la détermination (estimation) des positions et autres informations pertinentes des objets en mouvement dans les séquences d'images. Le suivi d'objet consiste à se verrouiller sur un ou plusieurs objets en mouvement en temps réel. Il comprend [Mikhaylov et al, [MSME14]] :

— Détection de mouvement
— Localisation de l'objet
— Segmentation des mouvements
— Suivi des objets

1.3.1 Filtre particulaire

La supériorité de la technologie des filtres à particules dans les systèmes non linéaires et non gaussiens détermine son large éventail d'applications [JOLION et Jean-Michel, [JOL]].

L'idée du filtre à particule est basée sur les méthodes de Monte Carlo, qui utilisent des ensembles de particules pour représenter des probabilités, et peuvent être utilisées dans n'importe quelle forme de modèle d'espace d'état. L'idée de base est d'exprimer sa distribution en extrayant des particules d'état aléatoires de la probabilité postérieure. Il s'agit d'une méthode d'échantillonnage à importance séquentielle [JOLION et Jean-Michel, [JOL]].

L'ensemble des particules pondérées a donc pour objectif d'estimer la densité de probabilité de l'état [Figure 1.5]. Et Pour estimer la densité de probabilité grâce aux particules pondérées, il faut trois étapes : propagation, pondération, et rééchantillonnage. En effet, il faut trouver le nombre de particules nécessaires, et la manière dont l'échantillonnage va être effectué. Ensuite, il faut une mesure associée aux observations de l'image pour déterminer le poids de chaque particule [Figure 1.5] [JOLION et Jean-Michel, [JOL]].

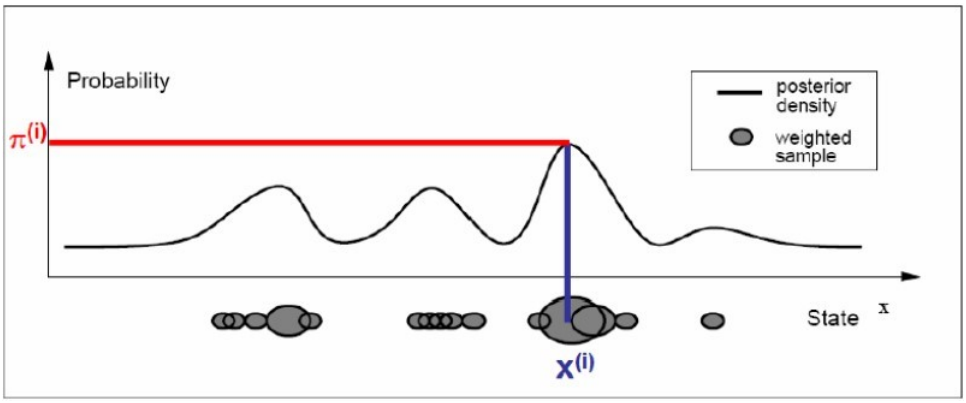

FIGURE 1.5 – Représentation des particules avec leurs probabilités [JOLION et Jean-Michel, [JOL]]

1.3.2 Filtre de Kalman

Le filtre de Kalman est un filtre récursif prédisant l'état courant du système à partir des Mesures précédentes si l'évolution dynamique du système est considérée comme linéaire. Le vecteur d'état X(t) est défini en fonction de X(t-1) et un terme correspondant à l'évolution dynamique du système [JOLION et Jean-Michel, [JOL]].

— **Intérêts** : obtenir une prédiction de l'état courant indépendamment des mesures obtenues et connaître la fiabilité du modèle de mouvement grâce à ces mesures.
— **Inconvénients** : sont la nécessité de la modélisation de l'évolution dynamique des objets de la scène, et sa sensibilité aux valeurs initiales à cause de son caractère itératif.

1.4 Conclusion

Durant ce chapitre, nous avons présenté quelque méthodes de détection qui figurent dans la littérature, et ça en définissant les descripteurs et les classifieurs qui sont utilisés pour performer cette étape de suivi d'objets. Comme nous avons aussi présenté deux méthodes de suivi les plus utilisées, le filtre de Kalman, et le filtre Particulaire. Et nous avons conclu ce chapitre en résumant deux des méthodes existantes qui permettent le suivi des objets en temps réel.

Chapitre 2

Conception

2.1 Introduction

Dans la vision par ordinateur les réseaux de neurones artificiels, et l'apprentissage automatique sont des outils très intéressants pour la détection d'obstacles avec une précision très élevée. Le but de ses algorithmes est de prédire des bounding boxes à partir d'une image. L'apprentissage automatique nous permet de localiser et de classifier ces obstacles en temps réel dans une image. Ces algorithmes n'incluent pas la notion de temps et de continué. Lors de la détection d'un obstacle, on suppose à chaque fois qu'il s'agit d'un nouvel obstacle.Pour y remédier à ce problème, des algorithmes de suivi ont été proposés, dans le but d'assurer la correspondance entre les différents objets détectés à travers le temps, et pour assurer un suivi d'objet précis et efficace.

A travers cette partie nous allons détailler trois méthodes inspirées de la littérature, que nous avons implémenté dans notre travail, afin de faire une comparaison entre les différents algorithmes de détection et de suivi, et tirer les avantages et les inconvénients de chacun d'eux.

2.2 Motivation

Le choix des détecteurs Yolo, MobileNet, SVM-HoG dans notre travail est inspiré par les différents travaux existants dans la littérature pour la détection et le suivi d'objet en temps réel, notamment les travaux exposés dans les articles (Sergio Canu, 2019, [Can19]), (Jeremy Cohen, 2019, [Coh19]), (Adrian Rosebrock, 2015, [Ros15b]), (Satya Mallick, 2016, [Mal16]), (DataTurks, 2018, [Dat18]).

Nous avons opté pour le filtre de Kalman et le filtre Particulaire, parce que l'utilisation de ces deux méthodes est très répondue dans le suivi des objets. Nous nous sommes inspiré par les articles (Jeremy Cohen, 2019, [Coh19]), (Nishagandhi, 2018, [nis18]), (Arulampalam, 2002, [AMGC02]), (Boui et Marouane, 2018, [Bou18]), pour la proposition de nos méthodes.

2.3 Détection et Suivi avec RNA-Kalman

Nous détaillons en premiere lieu la première solution que nous avons implémenté qui nous permet de détecter et suivre des personnes dans un flux vidéo en utilisant le modèle MobileNet pour la détection, l'algorithme Hongrois pour la correspondance, et le filtre de Kalman pour le suivi [Figure 2.1].

Cette solution a été inspiré par le travail de (Kyle Guan, 2015), qui consistait à faire un suivi des véhicules dans une circulation routière. Nous avons réadapté cette solution pour un suivi de personne en effectuant plusieurs modifications nécessaires.

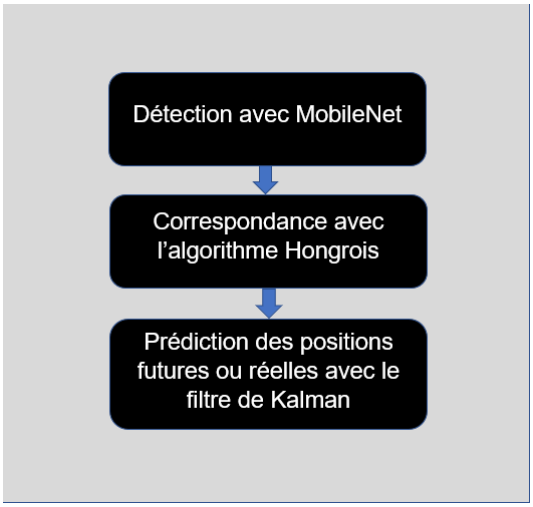

FIGURE 2.1 – Étapes de l'algorithme

2.3.1 Détection de personne avec MobileNet

Pour expliquer la conception de la solution que nous avons implémenté avec MobileNet nous allons prendre exemple similaire du RNA, il s'agit de YOLO. L'un des algorithmes les plus populaires pour la détection est YOLO (You Only Look Once). La sortie de l'algorithme consiste en une liste de bounding box de la forme [Classe, x, y, w, h, Confiance], où : **Classe** représente l'identifiant de la classe, **X** et **Y** représentent les coordonnées du centre de la bounding box, **W** et **H** représentent sa largeur et sa hauteur respectivement, et **Confiance** représente la précision de la détection.

Les bounding box sont utilisées pour compter le nombre d'obstacles de la même classe dans une foule. Cependant la limite s'agit du fait que les obstacles d'une classe sont de la même couleur et ne peuvent être séparés.

2.3.2 L'algorithme Hangrois (Kuhn-Munkres)

Aussi nommé algorithme Kuhn-Munkres, peut associer un obstacle d'une image à l'autre, sur la base d'un score qui serra calculé par l'une des méthodes suivantes :

- **IOU (Intersection Over Union)** : signifie que si la bounding box chevauche avec la précédente, c'est probablement la même.
- **Forme** : Si la forme ou la taille n'a pas trop variée entre deux boxes consécutives, alors le score augmente.
- **Convolution** : Avec un CNN à appliquer sur la box et comparer ce résultat avec celui d'une image antérieure, si les caractéristiques convolutées sont les mêmes, alors les objets ont le même aspect. Si présence d'occlusion partielle, les caractéristiques convolutives resteront en partie les mêmes et l'association restera.

Dans notre travail nous avons implémenté l'algorithme de Kuhn-Munkres, avec IOU pour le calcul de score, dans le but de détecter la correspondance entre la liste des box détectées à $t-1$ et celles détectées à t.

Dans ce qui suit, nous illustrons un exemple pour expliquer les étapes de l'algorithme implémenté.

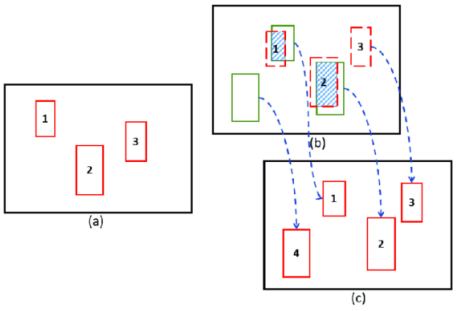

FIGURE 2.2 – Exemple explicatif de l'algorithme implémenté

Dans cet exemple [Figure 2.2], de la trame « a » à la trame « b », nous effectuons un suivi de deux obstacles (id 1 et id 2), en ajoutant une nouvelle détection (id 4) et en gardant la trace (id 3) ou cas où ça serait un faux négatif. Le processus d'observation est comme suit :

— Deux listes de box YOLO : une liste de tracking $(t-1)$ et une liste de détection (t).

— Passer dans les deux listes, et calculer le IOU, la forme, le score convolutif. Pour cet exemple considérons uniquement le IOU (nous pourrions aussi avoir une fonction de coût donnant de l'importance à chaque score, pour les convolutions, des mesures de distance cosinusienne pourraient être utilisées). Stockez les scores IOU dans une matrice [Figure 2.3].

  Dans le cas de chevauchement des bouding box, nous pouvons avoir deux ou plusieurs correspondances pour un même candidat. Dans ce cas, nous fixons la valeur maximale de l'IOU à 1 et toutes les autres à 0.

Detection/Tracking	Tracking 0	Tracking 1	Tracking 2
Detection A	IOU = 0	IOU = 0	IOU = 0
Detection B	IOU = 0.56	IOU = 0	IOU = 0
Detection C	IOU = 0	IOU = 0.77	IOU = 0

FIGURE 2.3 – Matrice IOU

— Ensuite nous utilisons une fonction *scickit-learn* appelée *lieanr_ assignment* qui implémente l'algorithme hongrois. Cet algorithme utilise un graphe bipartite pour trouver, pour chaque détection, la valeur de suivi la plus basse dans la matrice. Dans nos scores alors, nous remplacerons notre 1 par -1, le minimum serra trouver. Pour notre exemple, la matrice hongroise est illustrée dans la [Figure 2.4].

  Nous pouvons alors vérifier les valeurs manquantes dans notre matrice hongroise et les considérer comme des détections non appariées, ou des suivis non appariés.

ID Tracking	ID Detection
0	1
1	2

FIGURE 2.4 – Matrice hongroise

Nous allons alors obtenir à ce stade, une matrice indiquant quel élément de la liste de détection correspond à quel élément de la liste de tracking. À partir de de là, nous pouvons avoir des détections matchées, des détections non matchées, et des suivis non matchés.

2.3.3 Filtre de Kalman

Nous avons dans cette première solution utilisé les filtres de Kalman qui sont très populaires pour suivre les obstacles et prédire les positions actuelles et futures. Un filtre de Kalman est utilisé sur chaque bounding box dans le but d'estimer deux résultats importants : **La moyenne** qui est un vecteur composé des coordonnées du centre de la box (cx, cy), de la taille de la box (largeur, hauteur) et du changement de chacun de ces paramètres [Figure 2.5], et de **la covariance** qui est notre matrice d'incertitude dans l'estimation, nous allons la mettre à un nombre arbitraire et l'ajuster pour voir les résultats, plus le nombre est élevé, plus l'incertitude est grande.

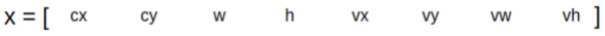

FIGURE 2.5 – vecteur de la moyenne

Pour chacun des boxes détecter nous associons un filtre de Kalman. Le fonctionnement des filtres de Kalman passe par deux étapes importantes :

— **Prédiction** : La phase de prédiction est une multiplication matricielle qui nous indiquera la position de notre box au temps t en fonction de sa position au temps $t-1$ [Figure 2.6].

$$x' = Fx + u$$
$$P' = FPF^T + Q$$

FIGURE 2.6 – Etape prédiction

Ou F est une matrice de transition d'état de taille [8x8]. Et la matrice Q est notre matrice de bruit, qui représente la confiance que nous accordons au système.

— **Mise à jour** : La phase de mise à jour est une étape de correction. Elle inclut la nouvelle mesure « z » et permet d'améliorer notre filtre. La mise à jour inclut deux étapes importantes :

  — La mesure d'une erreur entre la mesure (z) et la prédiction, ou z est le vecteur des coordonnées de la box retournée par le model de détection MobileNet.

  — Calculer un gain de Kalman (K). Le gain de Kalman est utilisé pour estimer l'importance de notre erreur. Nous l'utilisons comme facteur de multiplication dans la formule finale pour estimer un nouveau x [Figure 2.7].

$$y = z - Hx'$$
$$S = HP'H^T + R$$
$$K = P'H^TS^{-1}$$

$$x = x' + Ky$$
$$P = (I - KH)P'$$

FIGURE 2.7 – Equation mise à jour

Dont H[4x8] est notre matrice de mesure, et R est notre bruit de mesure.

Le filtre peut également être utilisé pour prédire au temps t+1 (prédiction sans mise à jour) à partir du temps t. Pour cela, il doit être suffisamment bon et avoir une faible incertitude.

2.4 Suivi d'objet en utilisant YOLO (You Only Look Once) et Les Filtre particulaire

Le suivi d'objets est souvent divisé en deux parties essentiels qui sont la détection et le suivi. Dans cette partie nous détaillons la deuxième solution que nous avons implémenté, nous permettant de faire de faire un suivi en suivant deux étapes :

— **Détection de personne :** Utilisation de le réseau de neurones profond YOLO.

— **Suivi de personne :**

— Filtre particulaire durant un certain nombre de frames.

— Réinitialisation de la détection. Puis faire une comparaison entre les nouveaux boxes détectés, et les boxes existants, en calculant la différence entre leurs histogrammes, afin de faire la correspondance entre les anciens boxes et les nouvelles, pour garder le suivi.

La [Figure 2.8] présente les étapes de L'algorithme implémenté :

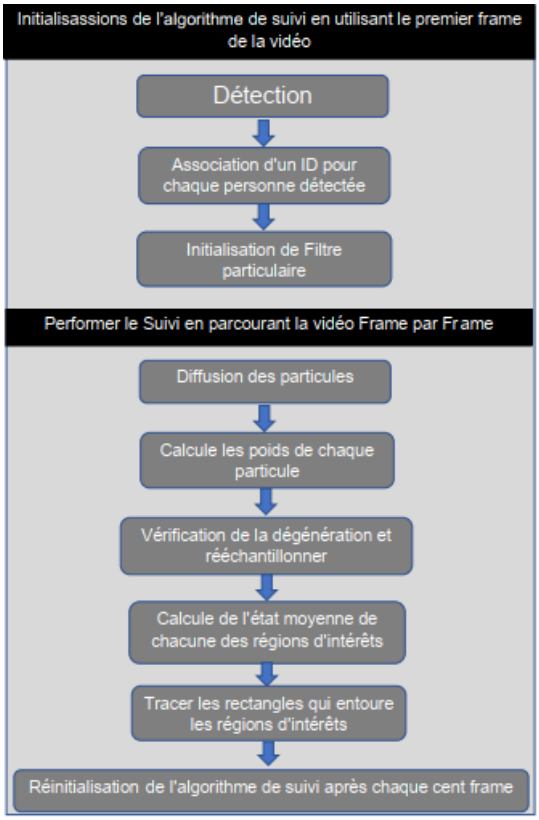

FIGURE 2.8 – Etapes de l'algorithme

2.4.1 Initialisation de l'algorithme

La partie initialisation de l'algorithme est performée sur le premier frame de la vidéo dans le but de détecter les régions d'intérêts (les régions ou se trouvent des personnes), et initialiser les particules de chacune des régions détectées.

2.4.1.1 Détection

Pour cette partie nous avons utilisé un modèle pré-entrainé, Yolo, utilisé pour la détection des personnes dans les frames vidéos.

Le principe de Yolo est d'appliquer un seul réseau de neurones à l'ensemble de l'image. Ce réseau divise l'image en régions et prédit les bounding boxes et les probabilités pour chaque région. Ces bounding boxes sont pondérées par les probabilités prédites.

Les étapes de l'algorithme :

- Lire le premier frame de la vidéo.
- Convertir l'image en blob : dans le but d'extraire des caractéristiques de l'image et de les redimensionner.
- Faire la détection en attribuant à Yolo les blobs comme entrée, et avoir comme sortie un vecteur avec :
  - ID de la classe à qui appartient l'objet détecté.
  - La position de l'objet détecté.
  - La précision de la détection.
- Pour tout objet détecté :
  - Vérifier si la précision de la détection > 0.8 et sa classe = « personne » :
    - Crée un vecteur « box » avec comme valeur :
      - Cordonnée x de début de rectangle qui englobe la région détectée
      - Cordonnée y de début de rectangle qui englobe la région détectée.
      - La largeur de rectangle.
      - La longueur de rectangle.
- Supprimer les boxes redondants qui représente la même région d'intérêt

2.4.1.2 Association des IDs

Dans cette étape de l'algorithme nous associons à chaque box un ID, ce qui fait que chaque personne détectée dans la scène aura un identifiant qui la diffère des autres personnes.

2.4.1.3 Initialisation de l'algorithme de filtre particulaire

L'étape d'initialisation de l'algorithme des filtres particulaire crée pour chaque région d'intérêt un ensemble de particule, et associe à ces particule un poids.

Les étapes de l'algorithme :

- Initialiser le nombre de particule.
- Pour chaque région d'intérêt :
  - La convertir en espace HSV.

— Calcule de son histogramme normalisé de la composante H.

— Initialiser la position initiale des particules qui sera le centre de la région.

— Attribuer à toutes les particules un poids initiale : 1/nombre_des_particule.

— Initialisation de pas à utiliser pour faire la diffusion des particules.

2.4.2 Le Suivi

Dans cette étape nous performons le suivi pour les différentes régions détectées en parcourant la vidéo frame par frame.

Les étapes importantes dans cette partie consistent à :
— Faire le suivi des personnes en appliquant le filtre particulaire [Figure 2.9].

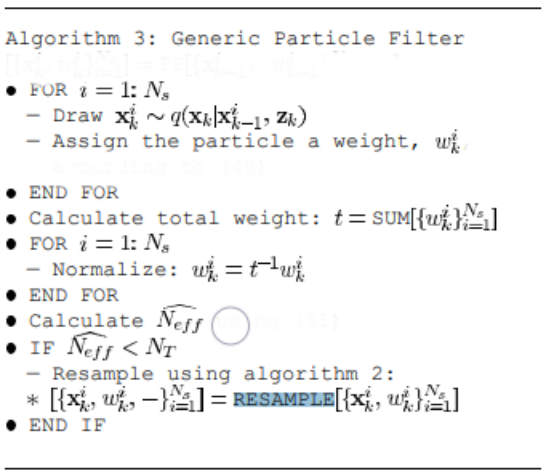

FIGURE 2.9 – Algorithme de filtre particulaire

— Réinitialisation de la détection après un certain nombre de frame dans le but de :

— Détecter les nouvelles personnes entrantes dans la scène.

— Éliminer les boxes des personnes qui ne figure plus dans la scène.

— Corriger le faux suivi de filtre particulaire qui peut se produire après un certain nombre de frames.

2.4.2.1 Diffusion des particules

Il est considéré comme la première étape de l'algorithme de filtre particulaire, après l'initialisation. Dans cette étape nous diffusons les particules en ajoutant des valeurs aléatoires qui suivent une loi uniforme.

2.4.2.2 Calcule de poids de chaque particule

Après la diffusion de particule nous calculons pour chaque particule un poids, qui définit son bon emplacement par rapport à la région d'intérêt initialisée lors de la détection.

Pour le calcul de poids nous passons par les étapes suivantes :

— Calcule de l'image de la rétro-projection, qui est un moyen d'enregistrer dans quelle mesure les pixels d'une image donnée correspondent à la répartition des pixels dans un modèle d'histogramme, dans notre cas l'histogramme est celui de la région d'intérêt calculée dans la phase d'initialisation. Le résultat de la rétroprojection est une matrice de la même taille que le frame, ou les pixels de la région d'intérêt ont des grandes valeurs par rapport aux autres pixels.

— Calcule d'un vecteur ou chaque case représente la valeur de pixel dans l'image de la rétro projection qui correspond au cordonnée x, y de particule.

— Normalisation de ce vecteur pour avoir un autre vecteur composé des poids des particules.

2.4.2.3 Ré-échantillonnage des particules

Cette étape a pour but de résoudre le problème du phénomène de dégénérescence, où après quelques itération, toutes les particules sauf une, auront un poids négligeable.

L'idée de base du ré-échantillonnage est d'éliminer les particules qui ont un faible poids et de se concentrer sur les particules de poids élevé.

Pour le ré-échantillonnage, nous suivons les étapes suivantes :

— Vérifier si N_{eff} tombe en dessous d'un certain seuil que nous choisissons, tel que $N_{eff} = 1/\sum(poids)$.

— Si c'est le cas nous appliquons le ré-échantillonnage [Figure 2.10].

Algorithm 2: Resampling Algorithm

$[\{\mathbf{x}_k^{j*}, w_k^j, i^j\}_{j=1}^{N_s}] = \text{RESAMPLE } [\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}]$

- Initialize the CDF: $c_1 = 0$
- FOR $i = 2$: N_s
  - Construct CDF: $c_i = c_{i-1} + w_k^i$
- END FOR
- Start at the bottom of the CDF: $i = 1$
- Draw a starting point: $u_1 \sim \mathrm{U}[0, N_s^{-1}]$
- FOR $j = 1$: N_s
  - Move along the CDF: $u_j = u_1 + N_s^{-1}(j-1)$
  - WHILE $u_j > c_i$
  * $i = i + 1$
  - END WHILE
  - Assign sample: $\mathbf{x}_k^{j*} = \mathbf{x}_k^i$
  - Assign weight: $w_k^j = N_s^{-1}$
  - Assign parent: $i^j = i$
- END FOR

FIGURE 2.10 – Algorithme de Rééchantillonnage

2.4.2.4 Calcule de l'état moyenne des particules

Le calcule de l'état moyenne des particules a pour but d'estimer la nouvelle position de la région d'intérêt prédite par le filtre particule. Le calcule est fait en faisant la somme de toutes les particules multipliées par leur poids.

2.4.2.5 Ré-initialisation de la détection

Le filtre particulaire permet juste de faire le suivi sur les personnes déjà détectées, d'où l'impossibilité de suivre des nouvelles personnes qui rentrent dans la scène, ou d'annuler le suivi d'une personne qui ne se trouve plus dans la vidéo. Cette étape est venue pour y remédier à ce problème, et aussi au problème

des position fausses qui peuvent être prédites par le filtre particulaire.

Cette partie passe par les étapes suivantes :

- Détection des régions d'intérêt dans le frame actuel.
- Pour chaque région détectée :
  - Calculer son histogramme.
  - Calculer la distance entre son histogramme et les différents histogrammes des régions détectées dans la phase de ré-initialisation précédentes (en utilisant la distance de Matusita), afin de vérifier s'il existe une correspondance entre cette région et l'une des régions existantes. Si la correspondance est vérifiée, alors :
    - Nous associons le même ID de la région qui lui correspond.
    - Sinon nous lui associons un nouveau ID.
- Initialisation de filtre particulaire

2.5 Suivi d'objet en utilisant HOG-SVM et Les Filtre particulaire

La troisième méthode que nous avons implémenté suit les mêmes étapes que celle de Yolo avec les Filtre particulaire, la seule différence est dans la partie de détection qui se base sur le descripteur HoG (Histogram of gradient) et l'Algorithme de classification SVM (Support Vector Machine).

La partie de détection se repose sur les étapes suivantes :

- Extraction des caractéristiques en utilisant l'algorithme de descripteur HoG.
- Extraction des caractéristiques en utilisant l'algorithme de descripteur HoG.
- Suppression des non maxima qui a pour but de supprimer les bounding boxes qui représentent une seule zone d'intérêt.

2.6 Conclusion

Dans cette partie nous avons expliqué les différentes solutions que nous avons proposé et implémenté dans notre travail. Nous avons présenté ces trois méthodes avec le principe de combinaison entre une méthode de détection d'objets et les méthodes de suivi, afin d'effectuer un suivi des personnes dans une scène vidéo.

Les étapes essentielles pour performer un bon suivi sont :

- Détection des personnes à un moment « t ».
- Faire la correspondance entre les personnes détectées et celle de la détection précédente à « t - 1 » .
- Application d'un algorithme de suivi pour la prédiction des positions futures.

Chapitre 3

Tests et évaluation

3.1 Introduction

Après avoir présenter la conception, il est nécessaire d'évaluer les performances de nos trois méthodes que nous avons implémenté.

Dans cette partie nous présentons les résultats obtenus dans les différentes méthodes implémentées [Figure 3.1], [Figure 3.2], [Figure 3.3], ainsi qu'une comparaison entre ces méthodes [Figure 3.4], [Figure 3.5] . De plus, nous effectuons une comparaison entre les méthodes de détections utilisées afin de tirer les points forts et les faiblesses de chacune.

3.2 Ressources matérielles

Pour l'exécution et les tests de notre application, nous utilisons un système d'exploitation 64 bits MacOS BigSur, tournant sur un PC portable MACBOOK PRO avec les caractéristiques suivantes :

— **Processeur** : 2 GHz Intel Core i5 quatre cœurs,
— **Mémoire RAM** : 16 Go 3733 MHz LPDDR4X,
— **Disque dur** : 512Go SSD,
— **Carte Graphique** : Intel Iris Plus Graphics 1536 Mo.

3.3 Discussion des résultats

Pour l'évaluation des trois méthodes que nous avons implémenté, nous avons calculé le nombre de frames ou le suivi ne diverge pas, la précision de la détection et la performance de suivi en temps réel.

3.3.1 MobileNet avec le Filtre de Kalman

Pour la première méthode que nous avons implémenté (MobileNet pour la détéction et Kalman pour le suivi) nous remarquons que :

— Le suivi diverge après un certain nombre de frames, et ça revient la méthode de correspondance utilisée qui se base sur IOU, alors lors de chevauchement des box qui est le cas lorsque deux personnes sont proches, l'algorithme le détecte comme si c'est une autre personne.

— Le suivi est un peu lent parce que pour chaque frame la détection est performée pour la mise à jour de Filtre de Kalman.

3.3.2 HoG-SVM avec le Filtre particulaire

Les résultats de notre deuxième méthode implémentée (HOG-SVM pour la détéction et Filtre Particulaire pour le suivi) nous constatons que :

FIGURE 3.1 – MobileNet avec le Filtre de Kalman

— Le suivi diverge rapidement, à cause des box détectées avec un fond considérable. La comparaison des histogrammes des box conduit à des fausses associations des ID.

— Le suivi est rapide parce que la détection est performée qu'après un certain nombre de frames et non pas pour chaque frame.

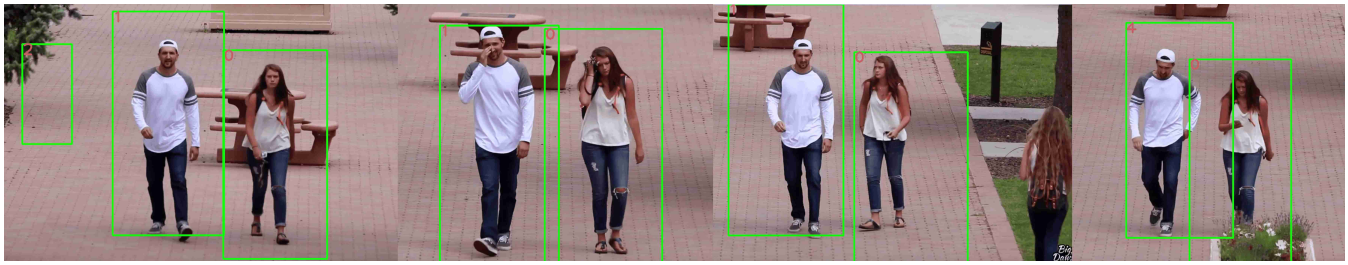

FIGURE 3.2 – HoG-SVM avec le Filtre particulaire

3.3.3 YOLO avec le Filtre particulaire

En ce qui concerne la troisième méthode que nous implémenté (YOLO pour la détection et Filtre Particulaire pour le suivi), nous avons constaté que :

— Cette méthode donne des résultats plus intéressants grâce à la bonne détection des personnes dans la scène, ce qui conduit à la convergence de suivi durant une longue durée.

— Le suivi est rapide parce que la détection est performée qu'après un certain nombre de frames et non pas pour chaque Frame.

FIGURE 3.3 – YOLO avec le Filtre Particulaire

Méthode	Précision de la détection	Nombre de frame ou le suivi est maintenu	Performance du suivi en temps réel
MobileNet avec le Filtre de Kalman	• Détections des personnes précise. • Les box ne contiennent pas trop de fond • Pas de fausse détection	• 360	• Un peu long
Yolo avec le Filtre particulaire	• Détections des personnes précise. • Les box ne contiennent pas trop de fond • Pas de fausse détection	• 600	• Rapide
HoG-SVM avec le Filtre Particulaire	• Détections de personne moins précise. • Les box contiennent trop de Fond • Des fausses détections sont obtenues sur certain Frame	• 200	• Rapide

FIGURE 3.4 – Comparaison des Méthodes sur une vidéo

SVM Filtre Particulaire		Nombre de personnes dans la vidéo	Nombre de personne bien détectées	Nombre de personne non détectées	Nombre de fausses détections	Nombre de personnes bien suivies	Exécution en temp réel
Vidéo 1	Frame 1	2	2	0	1	2	00 :02 :49
	Frame200	2	2	0	0		
	Frame600	3	2	1	1		
Vidéo 2	Frame1	7	4	3	1	2	00 :02 :02
	Frame200	9	7	2	3		
	Frame400	5	3	2	1		
Vidéo 3	Frame	2	2	0	1	3	00 :01 :00
	Frame	2	2	0	2		
	Frame	4	4	0	1		
YOLO Filtre Particulaire		Nombre de personnes dans la vidéo	Nombre de personne bien détectées	Nombre de personne non détectées	Nombre de fausses détections	Nombre de personnes bien suivies	
Vidéo 1	Frame 1	2	2	0	0	2	00 :00 :49
	Frame200	2	2	0	0		
	Frame600	3	3	0	0		
Vidéo 2	Frame1	7	6	1	0	4	00 :00 :45
	Frame200	9	7	2	0		
	Frame400	5	4	1	0		
Vidéo 3	Frame1	2	2	0	0	3	00 :00 :18
	Frame100	2	2	0	0		
	Frame200	4	3	1	0		
MobileNet Filtre Kalman		Nombre de personnes dans la vidéo	Nombre de personne bien détectées	Nombre de personne non détectées	Nombre de fausses détections	Nombre de personnes bien suivies	
Vidéo 1	Frame1	2	2	0	0	4	00 :00 :50
	Frame200	2	2	0	0		
	Frame600	3	3	0	0		
Vidéo 2	Frame1	7	6	1	0	6	00 :00 :46
	Frame200	9	6	3	0		
	Frame400	5	4	1	0		
Vidéo 3	Frame1	2	2	0	0	2	00 :00 :16
	Frame100	2	0	2	0		
	Frame200	4	3	1	1		

FIGURE 3.5 – Comparaison des Méthodes sur plusieurs vidéos

3.4 Perspectives pour améliorer les solutions

Les résultats d'exécution nous ont permis de même de tirer quelques limites de ces méthodes que nous avons mis en place que nous pouvons résumer dans les deux points suivants :

— Les performances des trois méthodes dégradent dans le cas ou plusieurs personnes se trouve dans la scène.

— Le suivi avec les filtres à particules diverge dans le cas d'un mouvement rapide, ce qui nous conduit à refaire l'initialisation de la détection après un certain nombre de frames réduit. Dans ce cas, nous perdons la contrainte de rapidité.

Dans le but d'améliorer les résultats que nous avons obtenus à travers les trois méthodes que nous avons implémenté, nous proposons quelques perspectives :

— Utilisation de l'algorithme Hongrois avec un calcule de score qui se base sur la convolution pour améliorer la méthode de suivi avec MobileNet et le Filtre de Kalman.

— Implémentation de la méthode Yolo sur un Framework d'apprentissage profond qui donne une détection avec un GPU au lieu de CPU, pour performer une détection en temps réelle. Dans ce cas la ré-initialisation de la détection après un petit nombre de frame, pour remédier au problème de divergence de filtres particulaires, n'influencera pas sur la performance de suivi en temps réel.

— Faire une extraction de fond avant la comparaison des zones détectées pour limiter la fausse attribution des ID après la détection.

— Faire une comparaison des descripteurs des régions d'intérêts, au lieu de comparer l'histogramme de la composante teinte de la région, dans la partie correspondance.

3.5 Conclusion

Dans ce chapitre nous avons exposer les tests des performances du processus de détection et de suivi de personne dans une vidéo. D'après les résultats trouvés, nous concluons que les méthodes implémentées ont des limites quand le nombre de personne dans la scène est important. La méthode de Yolo avec le filtre particulaire donne un meilleur résultat par rapport aux deux autres méthodes ou nous avons réussi à suivre deux personnes durant presque toute la vidéo.

En conclusion de ce projet de fin d'étude, nous avons mis en place les connaissances que nous avons acquis durant notre premier semestre au sein de notre formation. Nous avons mis en pratiques les différentes notions utilisées dans la vision par ordinateur pour la détection d'objets ainsi que dans le suivi de ces derniers. Nous avons alors réussi à implanter trois solutions et à comparer les résultats obtenus, ce qui nous a permis d'envisager des perspectives qui nous permettront d'améliorer nos solutions que nous avons proposé.

Bibliographie

[ABH+13] W Aitfares, EH Bouyakhf, A Herbulot, F Regragui, and M Devy. Hybrid region and interest points-based active contour for object tracking. *Appl Math Sci*, 7(118) :5879–5899, 2013.

[AMGC02] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2) :174–188, 2002.

[Bou18] Marouane Boui. *Détection et suivi de personnes par vision omnidirectionnelle : approche 2D et 3D*. PhD thesis, Université Paris-Saclay (ComUE), 2018.

[Can19] Sergio Canu. Yolo object detection using opencv with python. 2019. `https://pysource.com/2019/06/27/yolo-object-detection-using-opencv-with-python/`.

[Coh19] Jeremy Cohen. Tracking par computer vision. 2019. `https://medium.com/france-school-of-ai/tracking-par-computer-vision-90e5111cbb86`.

[Dat18] DataTurks. Efficient implementation of mobilenet and yolo object detection algorithms for image annotation. 2018. `https://medium.com/hackernoon/efficient-implementation-of-mobilenet-and-yolo-object-detection-algorithms\protect\@normalcr\relax-for-image-annotation-717e867fa27d`.

[DT05] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[Fla17] RONTEIX-JACQUET Flavien. La détection de tête en temps réel dans les scènes localement dense à l'aide des dernières techniques de deep learning. 2017. `https://flavienrj.github.io/docs/Rapport_stage_4A.pdf`.

[Hay04] Simon Haykin. *An introduction to neural networks*. the Taylor Francis e-Library, 2004.

[HZC+17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets : Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv :1704.04861*, 2017.

[JOL] Jean-Michel JOLION. Étude bibliographique suivi de personnes et trajectoire dans un réseau de caméras.

[Mal16] Satya Mallick. Histogram of oriented gradients. 2016. `https://learnopencv.com/histogram-of-oriented-gradients/`.

[MSME14] Dmitry Mikhaylov, Anton Samoylov, Peter Minin, and Alexey Egorov. Face detection and tracking from image and statistics gathering. In *2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems*, pages 37–42. IEEE, 2014.

[nis18] nishagandhi. Face-detection-and-tracking. 2018. `https://github.com/nishagandhi/Face-Detection-and-Tracking`.

[Qua19] Quantmetry. Mobilenet, optimisation de la convolution pour les réseaux de neurones embarqués. 2019. `https://www.quantmetry.com/blog/mobilenet-optimisation-de-la-convolution-pour-les-reseaux-de-neurones-embarques/`.

[Ros15a] Adrian Rosebrock. Local binary patterns with python et opencv. 2015. https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-pythonopencv/.

[Ros15b] Adrian Rosebrock. Pedestrian detection opencv. 2015. https://www.pyimagesearch.com/2015/11/09/pedestrian-detection-opencv/.

[Ros18] Adrian Rosebrock. Yolo object detection with opencv. 2018. https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/.

[SA11] Nandita Sethi and Alankrita Aggarwal. Robust face detection and tracking using pyramidal lucas kanade tracker algorithm. *International Journal of Computer Technology and Applications*, 2(5) :1432–1438, 2011.