République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A. Mira de Béjaïa
Faculté de Technologie
Département de Génie Electrique

Mémoire de Fin d'Etudes

En vue de l'obtention du diplôme de Master en Automatique et informatique industrielle

Thème

Génération automatique de synospis pour caméras de surveillance

Réalisé par

M. AISSIOU Saïd M. HEBBACHE Koceila

Devant le jury composé de

Examinateur : M. SAJDI
Examinatrice : M^{me} BELLAHSENE

Encadré par : M^{me}. MEZZAH

Promotion 2019 - 2020

Remerciements

En premier lieu, nous remercions le bon Dieu tout puissant de nous avoir donné la force et la volonté durant nos études et pendant réalisation de ce projet.

Nous voudrions aussi remercier notre encadreur M^{me} MEZZAH pour son aide et ses précieux conseils.

Comme nous tenons à remercier les membres du jury d'avoir accepté d'examiner et de juger notre modeste travail.

Un remerciement particulier à nos familles pour nous avoir soutenus, accompagnées et nous avoir permis d'en arriver là.

Enfin, nous remercions tous nos amis pour leur aide et leurs encouragements ainsi que tous ceux qui ont contribué de près ou de loin à la réalisation de ce projet.

Dédicaces

A mes parents et mes soeurs qui m'ont soutenu, encouragé et surtout supporté durant mes études.

A mes amis qui ont toujours su être présents.

AISSIOU Saïd

A mes parents, mes frères et ma petite soeur Thiziri qui m'ont toujours soutenu et encouragé durant mes études.

A mes amis qui ont toujours été présents.

HEBBACHE Koceila

Table des matières

Table des figures

Liste des abréviations

AVI	Audio Vidéo Interleave
EM	Esperance maximisation
GMM	Gaussian Mixtures Models
GUI	Graphic User Interface
R-CNN	Region Based Convolutional Neural Networks
RGB	Rouge,Ver,Bleu
YOLO	You Only Look Once

Introduction générale

Aujourd'hui, le déploiement de la vidéosurveillance continue de croître de façon exponentielle dans le monde entier, générant des heures et même des jours de vidéosurveillance qui restent sans analyse. L'examen des vidéos de surveillance est aussi difficile pour le gouvernement, les forces de l'ordre et d'autres organisations et entreprises que pour les utilisateurs grand public. En effet, bien que ces utilisateurs de vidéosurveillance puissent considérer la surveillance d'une entreprise ou d'une propriété privée, d'animaux domestiques, d'enfants, d'aînés et de gardiens comme critiques, ils ne disposent tout simplement pas du temps ou des ressources nécessaires pour visionner de grands volumes de séquences vidéo. Par conséquent, il existe un besoin croissant de technologies pouvant aider les opérateurs de vidéosurveillance à analyser les séquences vidéo, que ce soit en temps réel ou hors ligne, dans des scénarios d'enquête post-événement. En réponse à ce besoin aigu de technologie qui peut efficacement aider les utilisateurs à surmonter les défis de l'examen vidéo, plusieurs méthodes de génération automatique de synopsis (résumé vidéo) ont été mises au point et développées à la base des technologies de vision par ordinateur.

L'objectif de notre travail est de générer un synopsis vidéo permettant de réduire le temps d'analyse de flux dans une vidéo de surveillance. Les conditions adoptées pour notre projet considèrent des vidéos de caméra fixe et un flux peu dense à analyser hors ligne. La technique que nous avons utilisé repose sur la détection d'objets et le suivi de leurs trajets. La méthode de détection d'objets est réalisée avec un modèle de mixture de gaussienne pour extraire l'arrière-plan et le suivi de trajet est déterminé par les résultats de détection ou par prédiction en utilisant un filtre de Kalman.

Ce mémoire est organisé en 3 chapitres, le premier chapitre présente des généralités sur les systèmes de surveillances automatisés, leur but et leur architecture. Le deuxième chapitre est consacré à l'étude de différents algorithmes de détection d'objets avec des exemples d'applications. Le dernier chapitre expose le fond de notre travail, à savoir une interface graphique MATLAB permettant de générer un synospis vidéo à partir d'une vidéo choisie par l'utilisateur.

Chapitre 1

Généralités sur les systèmes de surveillance automatisés

1.1 Introduction

La vidéosurveillance consiste à placer des caméras de surveillance dans un lieu public ou privé afin de visualiser et/ou enregistrer en un endroit centralisé tous les flux de personnes au sein d'un lieu ouvert au public pour surveiller les allées et venues, prévenir les vols, agressions, fraudes et gérer les incidents et mouvements de foule. Au début des années 2000, les caméras font leur apparition en nombre important dans de nombreuses villes européennes.

1.2 L'apparition de la vidéosurveillance

La vidéosurveillance s'est développée d'abord au Royaume-Uni, en réponse aux attaques de l'IRA (Armée républicaine irlandaise en anglais Irish Republican Army). Les premières expériences au Royaume-Uni dans les années 1970 et 1980 ont conduit à des programmes de grande ampleur au début des années 1990. Ces succès conduisirent le gouvernement à faire une campagne auprès de la population, et lança une série d'installations de caméras. La British Security Industry Authority (BSIA) estime qu'il y a jusqu'à 5,9 millions de caméras de surveillance au Royaume-Uni dont 750 000 dans des endroits sensibles tels que les écoles, les hôpitaux. etc [1]

1.3 Les buts de la vidéosurveillance

Les raisons de l'installation de systèmes de vidéosurveillance sont diverses, toutefois la sécurité publique, la protection des biens mobiliers ou immobiliers et les attentats font office d'éléments phares dans la justification de la vidéosurveillance.

Cette menace qui a toujours été présente n'a jamais vraiment créé un sentiment d'insécurité, mais les attentats du 11 septembre 2001 ont changé la donne. Les gens ont pris conscience que personne n'était intouchable. Toutefois la mise en place de la vidéosurveillance ne peut s'expliquer uniquement par l'insécurité grandissante ou la protection des biens. Certaines autres raisons moins connues du grand public existent également. La mise en place de la vidéosurveillance permet une amélioration de la gestion des incidents ainsi qu'une augmentation de l'efficacité et de la rapidité d'intervention. Par exemple, dans la prévention du suicide ou encore lors d'accidents qui pourraient survenir sur la voie publique.

Elle permet ainsi indirectement, de maintenir les primes d'assurances à un niveau raisonnable. La surveillance des axes routiers sert à informer en temps réel les automobilistes sur les conditions du trafic.

Quelques affaires de crimes ont été résolues grâce aux enregistrements fournis par les caméras de surveillance. Par exemple, après les attentats du métro de Londres du 7 juillet 2005, les enregistrements des caméras de surveillance ont été utilisés pour identifier les poseurs de bombes, bien qu'il soit admis qu'ils n'aient pas été indispensables

1.4 Evolution de la vidéosurveillance

Les systèmes de surveillance vidéo de « première génération » ont commencé avec des systèmes de vidéosurveillance analogiques, qui consistaient en un certain nombre de caméras connectées à un ensemble de moniteurs via des commutateurs automatisés.

La supervision humaine étant coûteuse et inefficace en raison du déploiement généralisé de tels systèmes, ils sont plus ou moins utilisés comme un outil médico-légal pour enquêter une fois l'événement survenu.

En combinant la technologie de vision par ordinateur avec des systèmes de vidéosurveillance pour le traitement automatique des images et des signaux, il devient possible de détecter de manière proactive des événements alarmants plutôt que l'enregistrement passif.

Cela a conduit au développement de systèmes semi-automatiques appelés systèmes de surveillance de « deuxième génération », qui nécessitent un algorithme de détection et de suivi robuste pour l'analyse comportementale. Par exemple, le système de surveillance visuelle en temps réel W4 [2] utilise une combinaison d'analyse de forme et de suivi, et construit des modèles d'apparence des personnes afin de détecter et de suivre des groupes de personnes ainsi que de surveiller leurs comportements même en présence d'occlusion et dans des environnements extérieurs.

Actuellement, la recherche se base sur les algorithmes de vision par ordinateur robustes en temps réel et sur les algorithmes d'apprentissage automatique de la variabilité des scènes et des modèles de comportements. Le système de surveillance de troisième génération vise à concevoir de grands systèmes de surveillance distribués et hétérogènes pour la surveillance de zones étendues comme la surveillance des mouvements de véhicules militaires aux frontières, la surveillance des transports publics, etc. L'approche de conception habituelle de ces systèmes de vision consiste à construire un vaste réseau de caméras et de capteurs coopératifs pour agrandir le champ de vision.

1.5 Architecture des systèmes de vidéosurveillence

On présentera dans cette partie les composantes matérielles et logiciels des systèmes de surveillance. La figure 1.1 illustre l'architecture la plus courante des systèmes de vidéosurveillance :

- Acquisition : La scène est enregistrée par une caméra de surveillance qui peut être statique ou mobile.
- Compression : Les données acquises par la caméra de surveillance nécessitent un large espace de stockage. Pour y remédier on doit compresser ces données.
- Transfert : Les données vidéo compressées doivent être envoyées au centre de traitement à l'aide de plusieurs moyens de transmission.
- Analyse : l'objectif de cette étape est de traiter et d'analyser le flux vidéos reçu. Certains systèmes archivent simplement les séquences vidéo pour une durée limitée. Ils sont visionnés qu'en cas de besoin. Les systèmes de vidéosurveillance intelligents analysent automatique-

ment et en temps réel les scènes transmises et alertent l'opérateur en cas de d'anomalie.

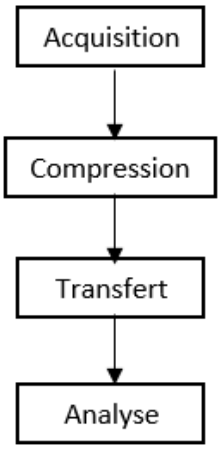

FIGURE 1.1 – Architecture des systèmes de vidéo surveillance.

1.6 Synopsis Vidéo

Les vidéos de surveillance enregistrées sont souvent trop longues à regarder et les approches automatisées d'analyse vidéo existantes, sont encore loin de donner des solutions satisfaisantes. L'opérateur doit prendre des décisions rapides et précises, mais dans la plupart des cas ceci est encore très difficile même pour les meilleurs systèmes d'analyse vidéo.

Les méthodes de résumé vidéo permettent une navigation plus efficace dans la vidéo, mais créent des résumés trop longs ou ambigu. La plupart des méthodes génèrent une description statique sous forme d'ensemble d'images clés. D'autres méthodes utilisent l'avance rapide (Timelapse), en supprimant les évènements non pertinents [3].

Le but de notre étude est de créer un synopsis afin d'améliorer l'efficacité de navigation dans les archives de vidéosurveillance, et de la rendre à la fois plus rapide et plus précise. Un synopsis vidéo présente simultanément plusieurs évènements se produisant à des instants différents ce qui permet de visionner des heures de séquences vidéo en quelques minutes [4][5][6][7]. La figure 1.2 illsutre le principe du synopsis vidéo.

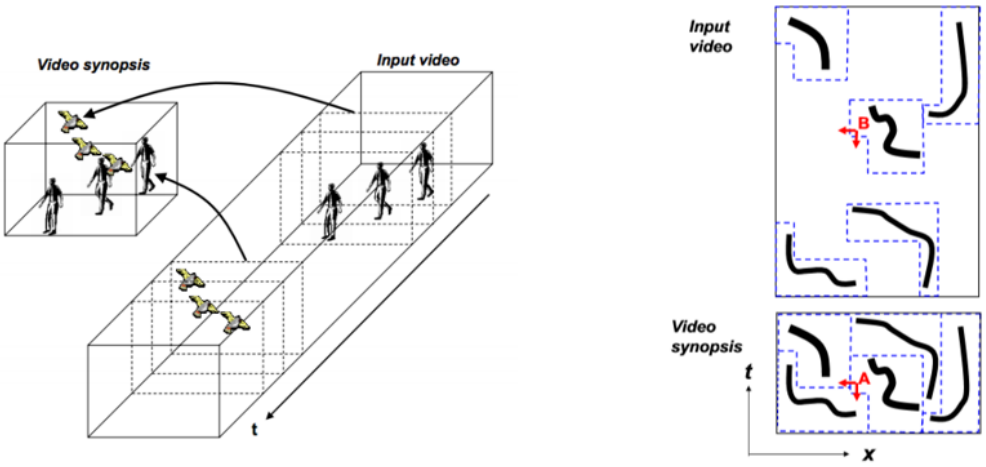

FIGURE 1.2 – Synopsis vidéo

A titre d'exemple, nous citons les solutions BriefCam qui est le principal fournisseur de l'industrie de la technologie vidéo synopsis pour l'examen et la recherche vidéo rapides, les alertes en temps réel et les informations vidéo quantitatives en exploitant principalement les techniques de deep learning [8].

FIGURE 1.3 – BriefCam Synopsis

1.7 Conclusion

Dans ce chapitre, nous avons présenté les buts, l'évolution et l'architecture des systèmes de vidéosurveillance automatisés ainsi que leurs domaines d'applications et leurs architectures.

Nous nous sommes aussi intéressé au synopsis vidéo, technique capable de réduire considérablement le temps nécessaire pour visionner et analyser une séquence de vidéosurveillance.

Chapitre 2

Algorithmes de détection d'objets

2.1 Introduction

Le présent chapitre sera consacré à l'étude de différents algorithmes de détection d'objets. On présentera leurs principes de fonctionnement ainsi que leurs différentes applications.

2.2 Principe d'extraction d'arrière-plan

Il existe plusieurs algorithmes de détection dont les étapes sont montrées dans la figure 2.1 [9]. Les quatres étapes principales d'un algorithme de soustraction d'arrière-plan sont le prétraitement, la modélisation d'arrière-plan, la détection de premier plan et la validation des données.

Le prétraitement consiste en une série de tâches de traitement d'image simples qui transforment la vidéo d'entrée brute en un format pouvant être traité par les étapes ultérieures. La modélisation d'arrière-plan utilise les nouvelles images pour calculer et mettre à jour un arrière-plan de référence. Ce modèle d'arrière-plan fournit une description statistique de l'ensemble de la scène d'arrière-plan. La détection du premier plan identifie ensuite les pixels de l'image qui ne peuvent pas être correctement expliqués par le modèle d'arrière-plan, et les sort sous forme de masque de premier plan. Enfin, la validation des données examine le masque candidat, élimine les pixels qui ne correspondent pas aux objets en mouvement réels et génère le masque de premier plan final.

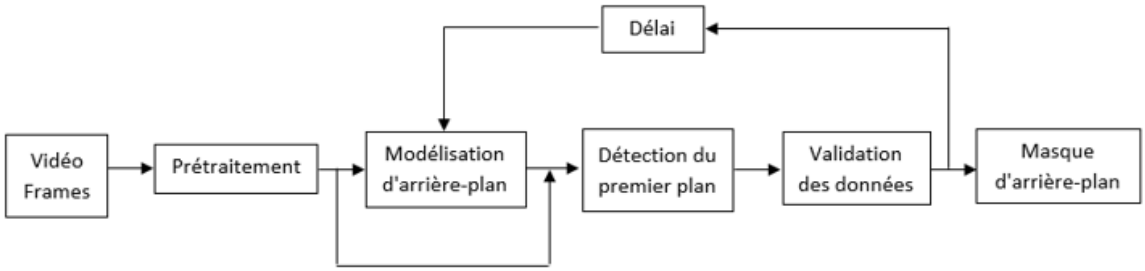

FIGURE 2.1 – Etapes d'extraction d'arrière-plan.

2.3 Détection d'objets

Nous nous intéressons dans notre projet à l'extraction des objets mobiles évoluant dans une séquence de vidéo surveillance. Le principe de détection de ces objets repose sur la soustraction entre une image dite image de référence et l'image courante. L'image de référence représente la scène statique de la vidéo.

Si l'image de référence n'est pas présente au début de la vidéo, il faut la construire et pour ce faire, on utilise tout simplement le moyennage de toutes les frames (images) de la vidéo :

$$f_{ref}(x,y) = \frac{1}{N}\sum_{i=1}^{N} f(x,y,i) \tag{2.1}$$

Avec f_{ref} l'image de référence, et $f(x,y,i)$ la valeur du pixel de la i^{me} image et N le nombre total d'images de la vidéo.

Au fil du temps, les éléments extérieurs subissent un changement ce qui provoque l'altération de l'arrière-plan de la vidéo. Il est donc nécessaire de réactualiser l'image de référence pour bien suivre l'évolutions de la luminosité et des ombres et dans ce cas-là, la méthode citée précédemment (moyennage des images) s'avère inefficace.

2.3.1 Modèle de mélange de Gaussiennes

2.3.1.1 Principe

L'algorithme GMM développé par Stauffer et Grimson [10] repose sur un mélange de gaussiennes adaptatives (fonction liée au phénomène aléatoire, dont la répartition faite au hasard obéi à la loi statistique gauss ou loi normale) permettant de modéliser l'arrière-plan de la séquence vidéo même en cas de changements des éléments extérieurs.

Chaque fois que les paramètres des gaussiennes sont mis à jour, les gaussiennes sont évalués à l'aide d'un simple heuristique pour évaluer les pixels susceptibles de faire partie de l'arrière-plan (valeurs les plus fréquentes). Les valeurs de pixels qui ne correspondent pas à l'un des gaussiennes de l'arrière-plan sont regroupés à l'aide de composants connectés formant l'avant plan (objet). Le processus de l'algorithme est décrit par l'organigramme de la figure 2.2.

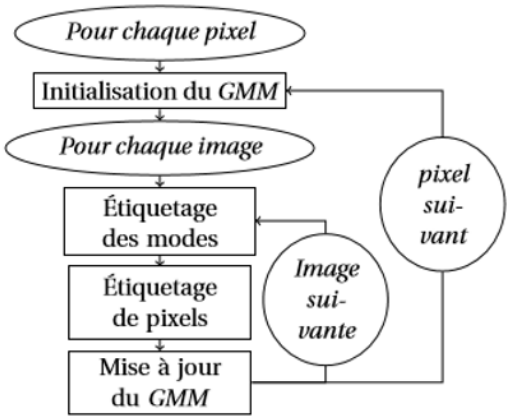

FIGURE 2.2 – Organigramme de l'algorithme GMM.

On peut décrire l'historique d'un pixel de coordonnées $\{x_0, y_0\}$ par :

$$\{x_1, ..., x_t\} = \{I(x_0, y_0, i : 1 \leq i \leq t\} \tag{2.2}$$

Avec I l'image de la vidéo et t le nombre total des images de la vidéo.

Le GMM associé au pixel p à l'image I est composé de K gaussiennes pondérées, et dans l'espace de couleur RVB chaque pixel se caractérise par son intensité dans R, G, et B. La fonction de densité d'un mélange de gaussiennes multivariées s'écrit alors :

$$F(x_t) = \sum_{i=1}^{N} \omega_{i,t} \eta(x_t, \mu_{i,t}, \Sigma_{i,t}) \tag{2.3}$$

Avec :

- $F(x_t)$: Probabilité de d'ovserver la valeur du pixel actuel.
- N : Nombre de distributions.
- $\omega_{i,t}$: L'estimation du poids du i^{me} mode gaussien dans le mélange à l'instant t
- $\mu_{i,t}$: La valeur moyenne du i^{me} mode gaussien dans le mélange à l'instant temps t.
- $\Sigma_{i,t}$: La matrice de covarience du i^{me} mode gaussien dans le mélange à l'instant t.
- η : Fonction de densité gausienne du i^{me} mode gaussien :

$$\eta(x_t, \mu_{i,t}, \Sigma_{i,t}) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_{i,t}|}} e^{-\frac{1}{2}(x_t - \mu_t)^T \Sigma^{-1} (x_t - \mu_t)} \tag{2.4}$$

Pour simplifier le calcul, la matrice de covariance écrit sous la forme :

$$\Sigma_{i,t} = \sigma_k^2 I \tag{2.5}$$

Ce qui suppose ici que les trois canaux de couleur RVB (Rouge, Vert, Bleu) sont de même variance. Quand ce n'est forcément pas le cas, l'hypothèse permet d'éviter une inversion de matrice coûteuse au détriment d'une certaine précision.

2.3.1.2 Etapes d'extraction d'arrière-plan avec l'algorithme GMM

Étape 1 : Initialisation du GMM

L'idéal est d'appliquer l'algorithme EM (esperance-maximisation) sur une partie de la vidéo, mais on peut aussi initialiser un unique mode par pixel (de poids 1), à partir des niveaux de la 1re image.

Étape 2 : Étiquetage des modes

Chaque mode gaussien est classé soit en arrière-plan ou en premier plan (objets). Plus le mode est fréquent et précis, plus il est probable qu'il caractérise les pixels de l'arrière-plan. Inversement, les éléments en mouvement introduits dans la scène sont représentés dans le modèle

par des gaussiennes de faibles poids et de fortes variances.

On peut alors approximer le modèle d'arrière-plan en retenant seulement les premiers modes gaussiens du mélange qui sont triés selon les valeurs croissantes de $\frac{\omega}{\sigma}$:

$$B = argmin_b(\sum_{k=1}^{b} \omega_k > T) \tag{2.6}$$

Étape 3 : Étiquetage des pixels

La troisième étape consiste à classer le pixel. Dans la plupart des méthodes, on attribue au pixel la classe du mode dont il est le plus proche, sous la contrainte suivante :

$$|x_t - \mu_{i,t}| \leq \gamma\sigma i, t \tag{2.7}$$

Où γ est un coefficient constant, à adapter pour chaque vidéo.

Étape 4 : Ajustement des paramètres

Il existe un modèle de mélange pour chaque pixel d'image, implémenter un algorithme EM exact sur une fenêtre de données récentes sera coûteux. Au lieu de cela, nous devons mettre en œuvre une approximation K-means en ligne. Cette dernière permet d'ajusté les paramètres du GMM suivant les équations suivantes :

$$\omega_{i,t} = (1 - \alpha)\omega_{i,t-1} + \alpha \tag{2.8}$$

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho x_t \tag{2.9}$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(x_t, \mu_t)^T(x_t - \mu_t) \tag{2.10}$$

$$\rho = \alpha\eta(x_t, \mu_t, \sigma_i) \tag{2.11}$$

2.3.1.3 Exemples d'application

Exemple d'application 1 :

La première vidéo testée est de format MPEG-4 couramment appelé mp4 et sa résolution est de 1280x720 pixels (aussi appelé 720P). La vidéo a été prise par une caméra fixe surveillant une route piétonnière [11].

La figure 2.3 illustre les objets détectés et leurs « bouding box » respectives. Ce terme désigne le rectangle contournant l'objet. Les bouding box des objets ont été obtenus en effectuant une analyse de blobs (Calcul des régions connexes).

La figure 2.4 illustre le masque : les pixels noirs représentent l'arrière-plan et les blancs les objets détectés (premier-plan).

FIGURE 2.3 – Détection d'objets avec l'algorithme GMM exemple 1.

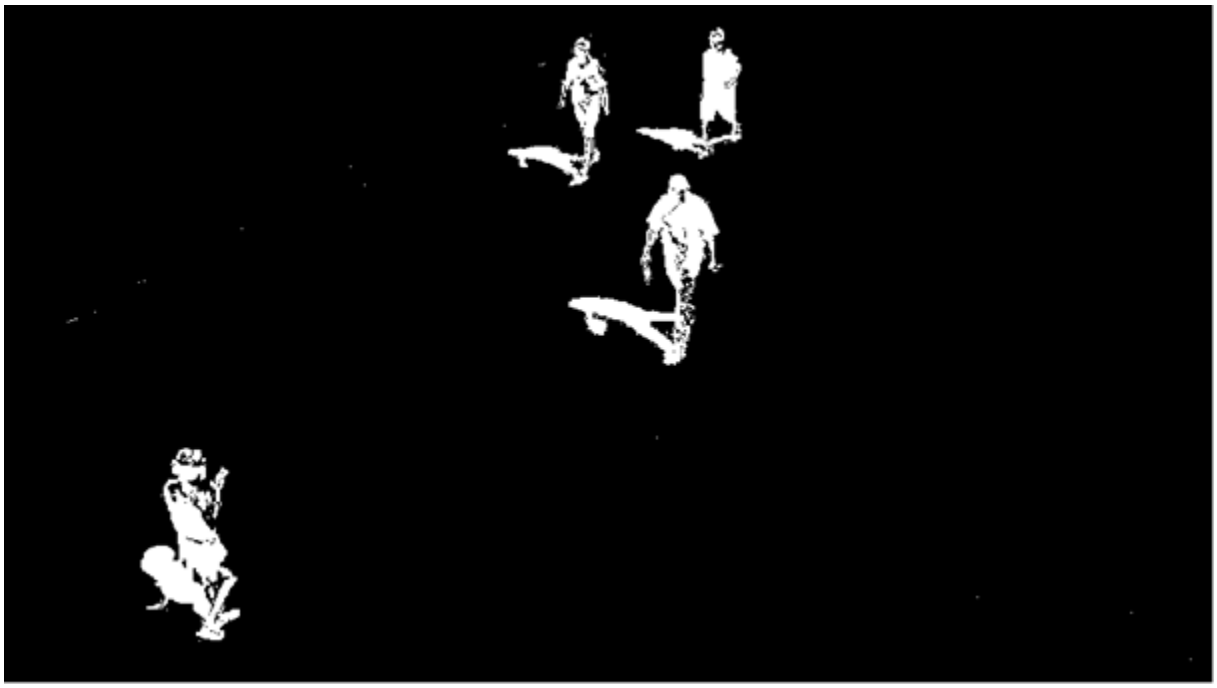

FIGURE 2.4 – Masque de premier plan exemple 1.

On remarque ici alors le présence d'un léger bruit, cela peut être arrangé en appliquant des opérations d'érosion et de dilatation. On obtiendra alors un résultat moins bruité. La figure 2.5 illustre le masque obtenu après l'application de ces deux opérations.

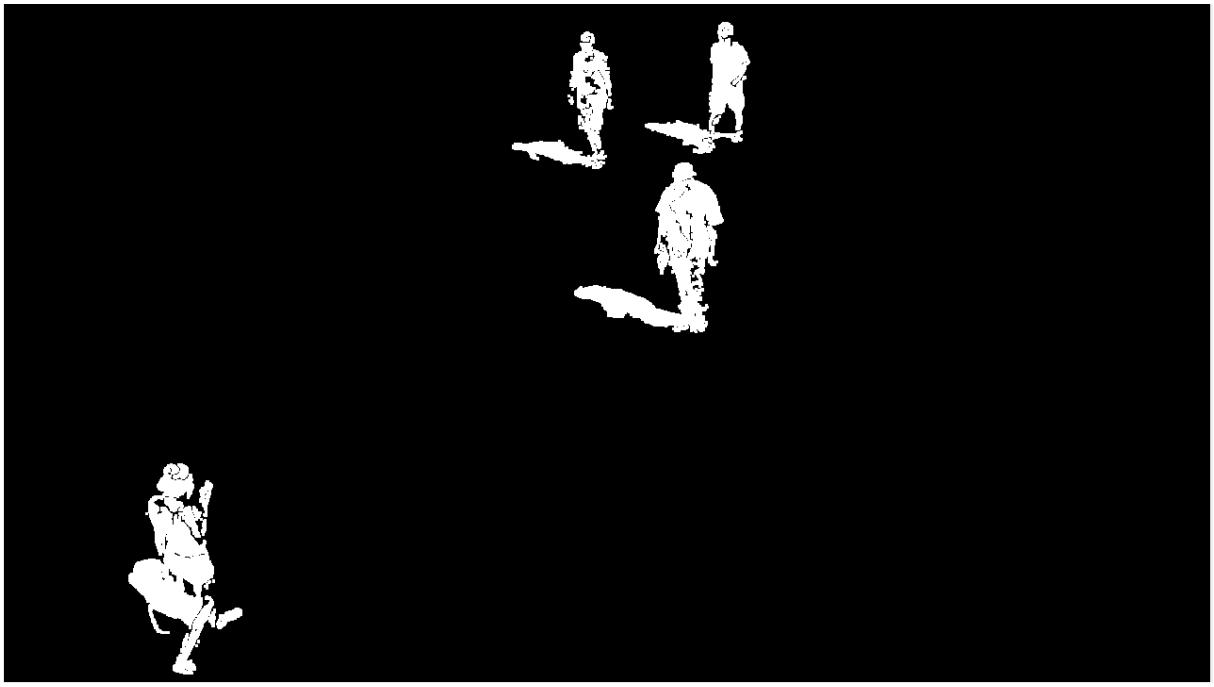

FIGURE 2.5 – Masque filtré.

Quand deux objets sont très proches l'un de l'autre, l'algorithme peut parfois les détecter en tant qu'un seul objet comme le l'illustre la figure 2.6. Cela peut être considéré comme étant un problème uniquement si on souhaite implémenter un compteur d'objets.

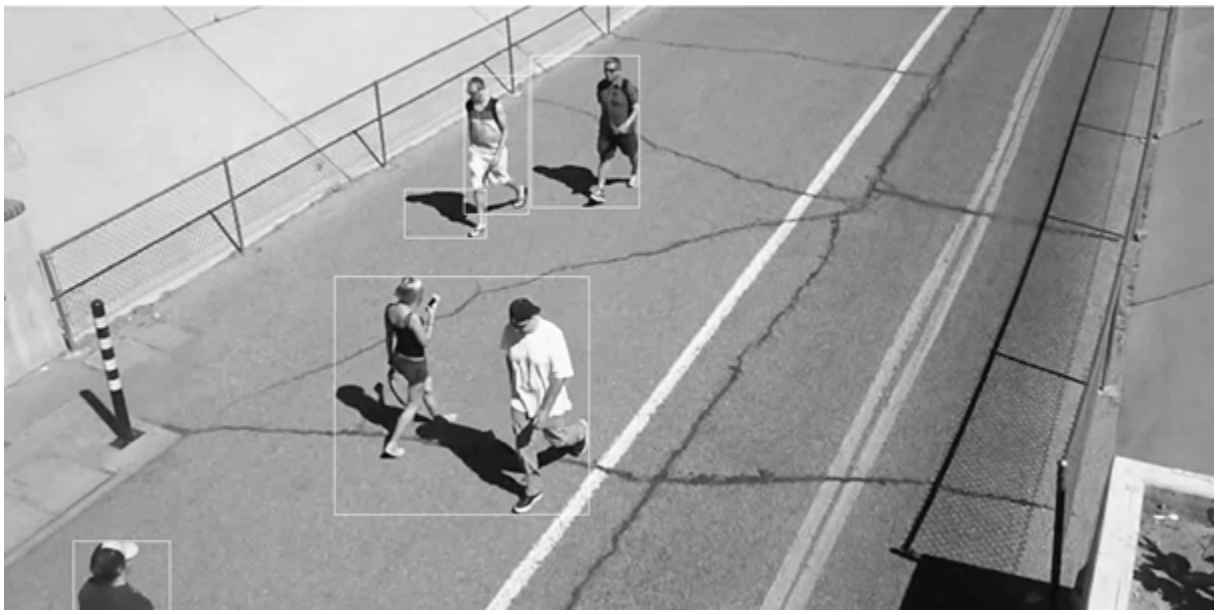

FIGURE 2.6 – Problème de bouding box.

Exemple d'application 2

Quant à la deuxième vidéo, on a utilisé une vidéo de format AVI (Audio Vidéo Interleave) de taille 360x640 pixels et c'est une vidéo de surveillance d'un trafic routier (exemples vidéos MATLAB). L'exécution a été beaucoup plus rapide en raison de sa taille. Les figures 2.7 illustre la détection d'arrière-plan et la figure 2.8 illustre le masque.

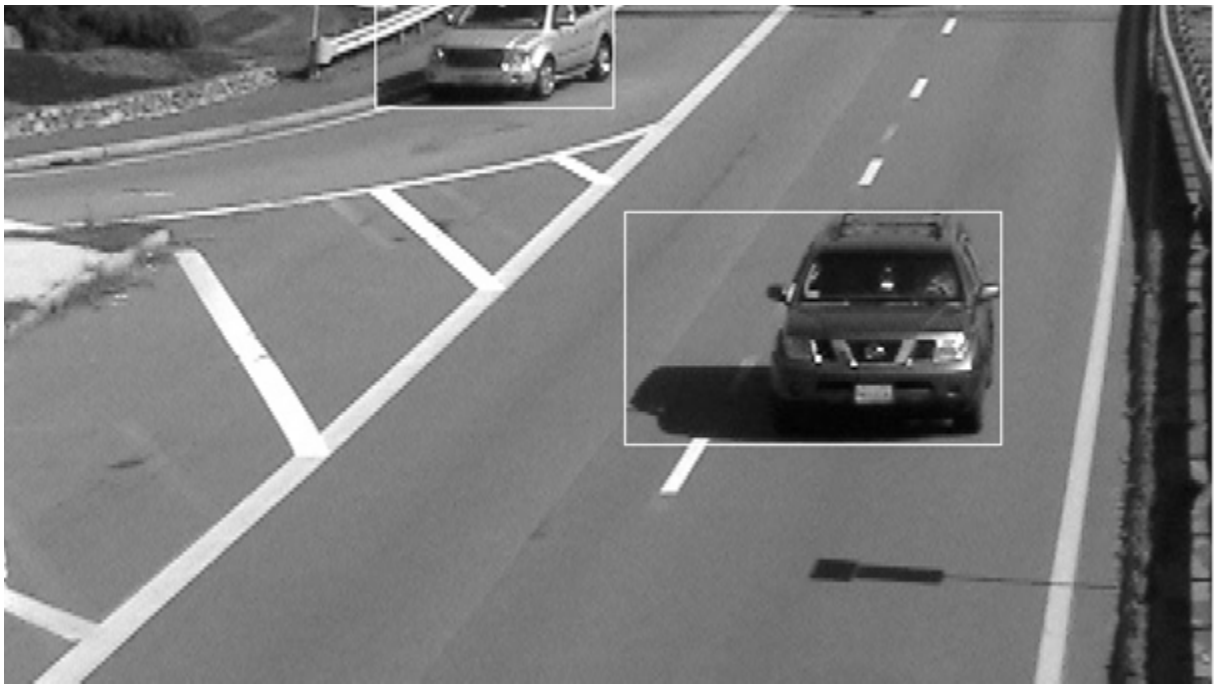

FIGURE 2.7 – Détection d'objets avec l'algorithme GMM exemple 2.

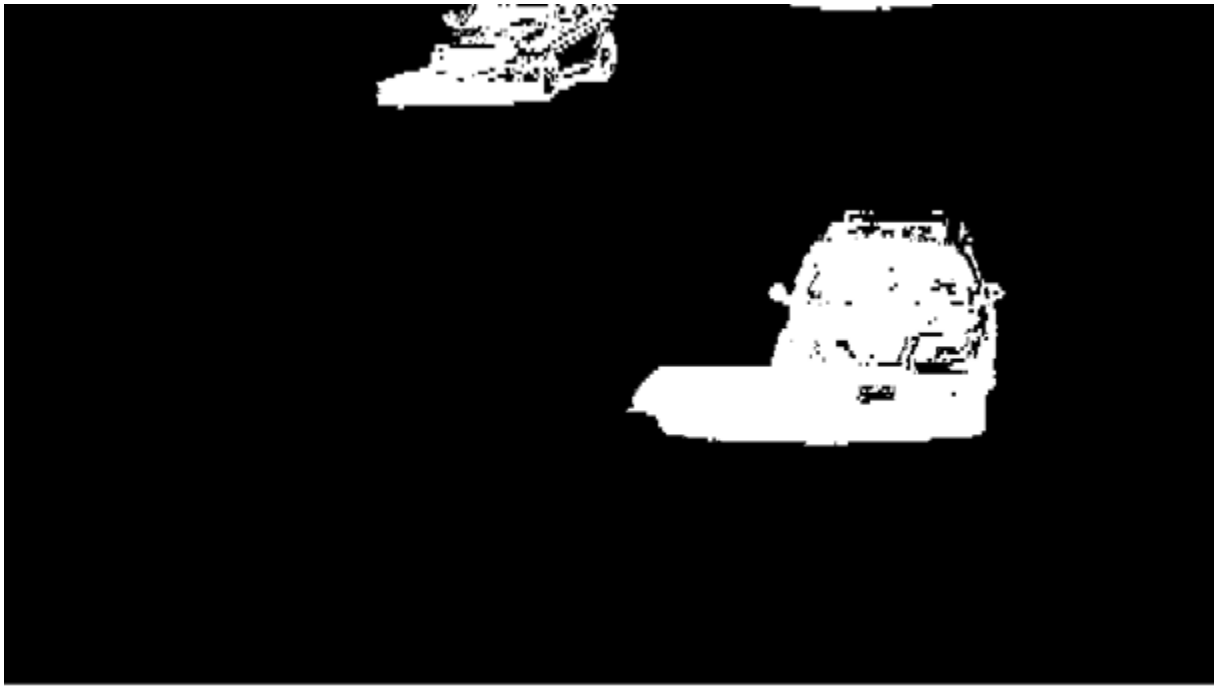

FIGURE 2.8 – Masque de premier plan exemple 2.

Exemple d'application 3

La vidéo utilisée est de format mp4 et de taille 640x360 pixels prise par une caméra fixe surveillant des passants dans un espace public (exemples vidéos MATLAB). Les figures 2.9 et 2.10 illustrent respectivement la détection d'arrière-plan et le masque. On remarque ici alors la

nécessité d'utiliser un algorithme de prédiction quand l'objet est derrière un obstacle. Ceci feras l'objet d'une étude à la fin de ce chapitre.

FIGURE 2.9 – Détection d'arrière plan avec l'algorithme GMM exemple 3.

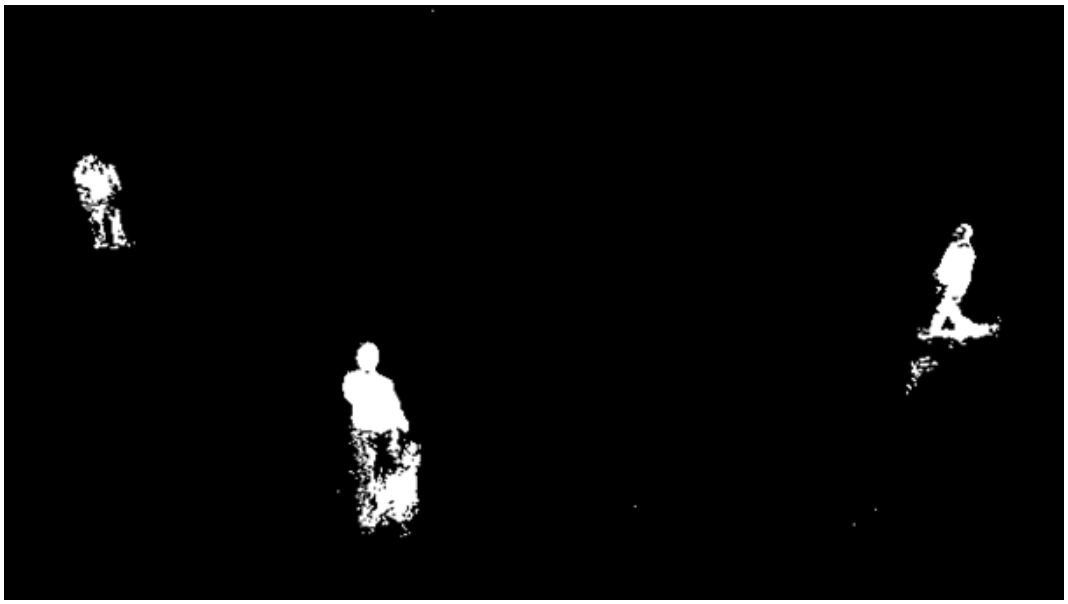

FIGURE 2.10 – Masque de premier plan exemple 3.

2.3.2 Flot Optique

2.3.2.1 Principe

Le flot optique décrit la direction et le taux de temps des pixels dans une séquence temporelle de deux images consécutives. Un vecteur bidirectionnel de vitesse dimensionnelle, portant des informations sur la direction et la vitesse du mouvement est attribué à chaque pixel de l'image.

Pour rendre le calcul plus simple et plus rapide, nous pouvons transférer les objets tridimensionnels du monde réel (temps + 3D) dans un cas (temps + 2D). Ensuite, nous pouvons décrire l'image au moyen de la fonction de luminosité dynamique 2D dépendante des coordonnées et du temps I (x, y, t) à condition qu'au voisinage d'un pixel déplacé, le changement d'intensité de luminosité ne se produit pas le long du champ de mouvement, nous pouvons alors utiliser l'expression suivante [12] :

$$I(x,t,y) = I(x+\delta x, y+\delta y, t+\delta t) \tag{2.12}$$

En appliquant les séries de Taylor sur la partie droite de (2.12), on obtient :

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + H.O.T \tag{2.13}$$

A partir de (2.12) et (2.13), en négligeant les termes d'ordres supérieurs (H.O.T) et après modifications, on obtient :

$$I_x.v_x + I_y.v_y = -I_t \tag{2.14}$$

Qui se traduit dans la représentation formelle du vecteur par :

$$\nabla I.\vec{v} = -I_t \tag{2.15}$$

Où ∇V est le gradient spatial de l'intensité de luminosité, v le vecteur de vitesse (flot optique) du pixel et It la dérivée dans le temps de l'intensité de luminosité.

L'équation (2.15), est appelée l'équation de contrainte de mouvement 2D (gradient de contrainte), et c'est l'équation la plus importante pour le calcul du flot optique. L'estimation du flot optique nécessite beaucoup de calcul.

A présent, il existe plusieurs méthodes pour sa résolution. Toutes ces méthodes partent de l'équation (2.12) qui suppose la conservation d'intensité de luminosité. La détermination du flot optique est résolue par le calcul des dérivées partielles du signal de l'image. Trois méthodes sont les plus utilisées, à savoir : Lucas-Kanade, Horn-Schunck et Gunnar Farnebäck. Ces méthodes supposent que la luminosité ne change pas au fil du temps.

2.3.2.1.1 La méthode de Lucas-Kanade

Lucas et Kanade ont introduit le terme d'erreur ρ_{LK} pour chaque pixel [13]. Celui-ci, selon la relation suivante, est calculé comme la somme des plus petits carrés pondérés de la contrainte de gradient (2.15) dans un voisinage proche du pixel :

$$\rho_{LK} = \Sigma_{x,y,\epsilon\Omega} W^2(x,y)[\nabla I(x,y,t).v + I_t(x,y,t)]^2 \tag{2.16}$$

Où Ω est le voisin du pixel, $W(x,y)$ sont les poids attribués aux pixels individuels Ω (généralement des coefficients gaussiens 2D). Pour trouver une erreur minimale, il est nécessaire de calculer la dérivation du terme d'erreur ρ_{LK} par des composantes individuelles de la vitesse et de mettre le résultat égal à zéro. Enfin, la forme matricielle de l'expression du flux optique est la suivante :

$$\dot{v} = [A^T W^2 A]^{-1} A^T W^2 b \tag{2.17}$$

Pour N pixels ($N = n^2$, pour n*n au voisinage de ω) et (Xi, Yi) Ω à l'instant t :

$$A = [\nabla I(x_1, y_1), ..., \nabla I(x_N, y_N)] \tag{2.18}$$

$$W = diag[W(x_1, y_1), ..., W(x_N, y_N)] \tag{2.19}$$

$$\vec{b} = -[I_t(x_1, y_1), ..., I_t(x_N, y_N)] \tag{2.20}$$

Nous obtiendrons donc la vitesse résultante pour un pixel comme solution du système (2.17). Au lieu du calcul des sommes, la convolution au moyen d'un filtre à gradient temporel gaussien ou différentiel est utilisée pour réduire la complication de l'algorithme [14].

2.3.2.1.2 La méthode de Horn-Schunck

La méthode de Horn et Schunck [15] est issue de de la méthode Lucas-Kanade (2.16). En plus de la contrainte de gradient (2.15), ils ont ajouté un autre terme d'erreur (appelé terme global de lissage) pour limiter les changements trop importants des composantes du flot optique (v_x, v_y) dans Ω. La minimisation de l'erreur totale ρ_{HS} est alors donnée par la relation :

$$\rho_{LK} = \int_D (\nabla I.\vec{v} + I_t) + \lambda^2 \left[\frac{\partial v_x}{\partial x}^2 + \frac{\partial v_x}{\partial y}^2 + \frac{\partial v_y}{\partial y}^2 + \frac{\partial v_y}{\partial x}^2 \right] d_x d_y \tag{2.21}$$

Où D (domaine) est la région de l'image entière, λ exprime l'effet relatif du deuxième terme d'erreur ajouté (typiquement $\lambda = 1.0$).

La méthode de Horn-Schunck est plus précise [14], mais pour un nombre relativement important d'itérations (en pratique, il y a 10 à 100 étapes), elle est plus lente.

La méthode de Lucas-Kanade est une méthode disséminée et locale, tandis que la méthode Horn-Schunck est dense et globale. Cette dernière suppose que le champ d'écoulement est globalement lisse (les vitesses voisines sont presque identiques). Alors que la méthode de Lucas-Kanade suppose que la vitesse est localement constante, et que les points voisins ont des déplacements semblables. La méthode de Lucas-Kanade produit moins de bruit par rapport à la méthode de Horn-Schunck. Le flot optique est utilisé pour détecter les objets en mouvement de façon indépendante en présence des mouvements de la caméra. Cependant, la plupart de ses méthodes exigent des calculs complexes difficiles à exécuter en temps réel. De plus, le flot optique est sensible au bruit de l'image.

2.3.2.1.3 La méthode de Gunnar Farnebäck

L'algorithme Farnebäck génère une pyramide d'images, où chaque niveau a une résolution inférieure par rapport au niveau précédent. Lorsqu'on sélectionne un niveau de pyramide supérieur à 1, l'algorithme peut suivre les points à plusieurs niveaux de résolution, en commençant au niveau le plus bas. L'augmentation du nombre de niveaux de pyramide permet à l'algorithme de gérer des déplacements de points plus importants entre les images. Cependant, le nombre de calculs augmente également. La figure 2.11 illustre une pyramide d'images à trois niveaux.

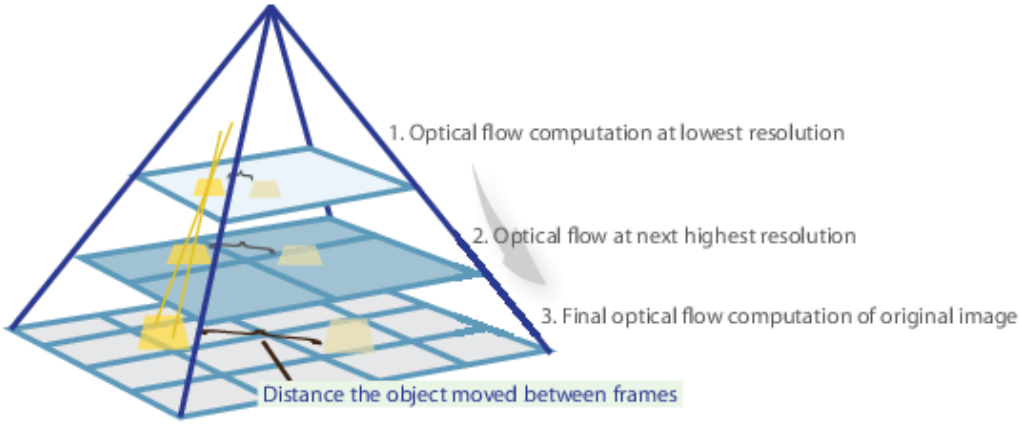

FIGURE 2.11 – Pyramide d'images à trois niveaux.

Le suivi commence au niveau de résolution le plus bas et se poursuit jusqu'à la convergence de l'algorithme. Les emplacements de points détectés à un niveau sont propagés en tant que points clés pour le niveau suivant. De cette façon, l'algorithme affine le suivi à chaque niveau. La décomposition pyramidale permet à l'algorithme de gérer de grands mouvements de pixels, qui peuvent être des distances supérieures à la taille du voisinage [16].

2.3.2.2 Exemples d'application

2.3.2.2.1 Méthode de Horn-Schunck

La figure 2.12 illustre l'application de la méthode de Horn-Schunck sur vidéo de surveillance de trafic routier (Matlab video sample). On remarque alors une bonne estimation du mouvement dans l'ensemble et un bruit quasi inexistant.

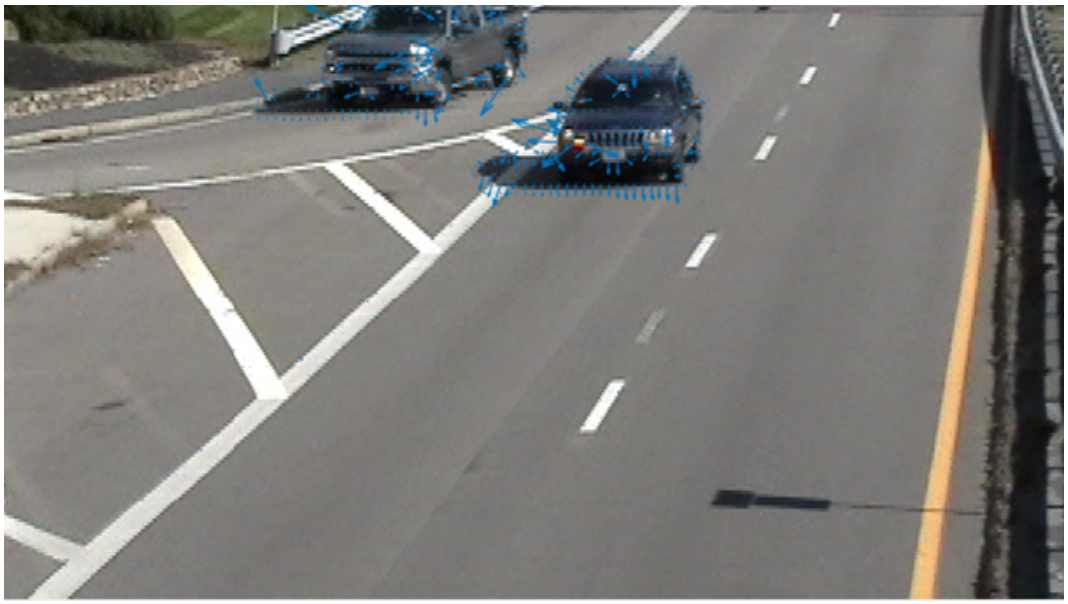

FIGURE 2.12 – Méthode de Horn-Shunck.

2.3.2.2.2 Méthode de Gunnar Farnebäck

La figure 2.13 illustre l'estimation du flot optique en utilisant la méthode de Gunnar Farnebäck. L'estimation est bonne dans l'ensemble mais on remarque la présence d'un léger bruit comparé à la méthode Horn-Schunck.

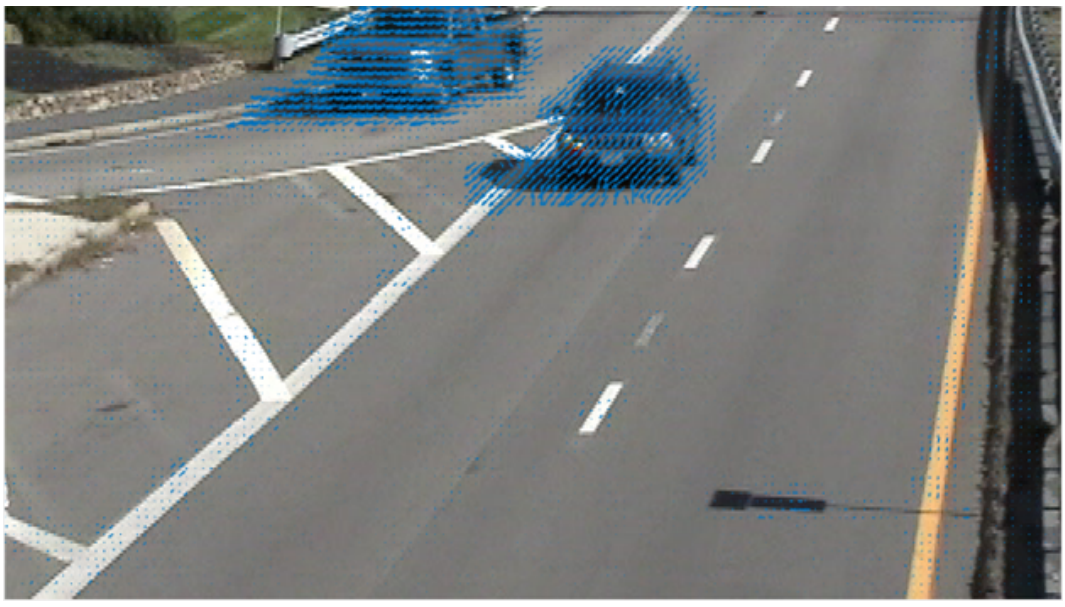

FIGURE 2.13 – Méthode de Gunnar Farnebäc.

2.3.2.2.3 Méthode de Lucas-Kanade

La figure 2.14 illustre l'application de la méthode de Lucas-Kanade et on remarque que c'est la moins efficace des trois méthodes.

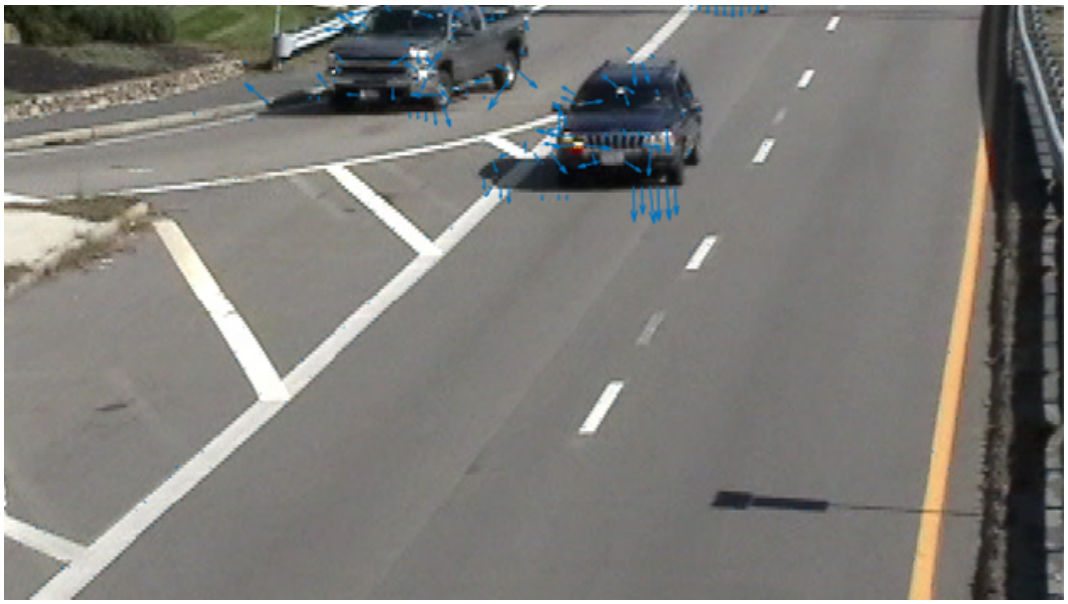

FIGURE 2.14 – Méthode de Lucas-Kanade.

2.3.3 Filtre de Kalman

2.3.3.1 Principe

Le suivi est l'estimation de la trajectoire d'un objet dans le plan image. Cette tâche requiert la localisation de chaque objet à partir d'une image à l'autre. Le suivi peut être fait en 2D, à partir d'une seule caméra, ou 3D, en combinant deux vues ayant une relation géométrique connue.

De nombreuses techniques de suivi prédisent la position de l'objet dans une trame à partir de ses déplacements observés dans les images précédentes. Chaque objet détecté doit être associé à son correspondant dans la trame suivante pour mettre à jour sa trajectoire, sinon une nouvelle trajectoire est créée. Le suivi de ces objets peut être difficile en raison de la complexité de leurs formes, leur nature non-rigide, leurs mouvements, des occlusions partielles ou complètes, des changements d'éclairage de la scène, etc.

En 1960, R.E. Kalman a publié son célèbre article décrivant une solution récursive au problème du filtrage linéaire à données discrètes [17]. Depuis, en grande partie grâce aux progrès du

numérique informatique, le filtre de Kalman a fait l'objet de plusieurs recherches et d'applications approfondies, notamment dans le domaine de la navigation autonome ou assistée.

Pour utiliser le filtre de Kalman il est nécessaire d'avoir une connaissance a priori sur la dynamique de la cible (objet) dans la vidéo. Pour mettre en équation ce filtre on utilise un vecteur de d'état qui représente l'ensemble minimal de variable permettant de mémoriser son passé. L'évolution de ce vecteur d'état soumis à un bruit aditif est décrit par :

$$x_{k+1} = \phi x_k + G u_k + v_k \tag{2.22}$$

Avec : ϕ : Matrice dynamique du système traduisant l'évolution temporelle du vecteur d'état. G : Matrice dynamique de commande représentant l'influence du vecteur de commande u_k sur le vecteur d'état. v_k : représente un bruit de mesure.

2.3.3.2 Etapes de l'algorithme

- Initialisation du filtre.
- Prédiction du vecteur d'état et de la covariance associé :

$$\hat{x}_{k+1\backslash k} = \phi . \hat{x}_{k\backslash k} \tag{2.23}$$

$$P_{k+1\backslash k} = \phi . P_{k\backslash k} . \phi^T + Q \tag{2.24}$$

$$\hat{Z}_{k+1\backslash k} = H . \hat{Z}_{k+1\backslash k} \tag{2.25}$$

- Actualisation du vecteur d'état et de la covariance :

  Calcul de l'erreur :

$$\gamma_{k+1} = Z_{k+1} - \hat{Z}_{k+1\backslash k} = Z_{k+1} - H . \hat{x}_{k+1\backslash k} \tag{2.26}$$

  Calcul du gain du filtre :

$$K_{k+1} = P_{k+1\backslash k} . H^T (R + H . P_{k+1\backslash k} H^T)^{-1} \tag{2.27}$$

Estimation du vecteur d'état :

$$\hat{x}_{k+1\backslash k+1} = \hat{x}_{k+1\backslash k} + K_{k+1}.\gamma_{k+1} \tag{2.28}$$

Estimation de la covariance sur l'état :

$$P_{k+1\backslash k+1} = (I - K_{k+1}.H)P_{k+1\backslash k} \tag{2.29}$$

2.3.3.3 Exemples d'application

L'algorithme de filtre de Kalman implique deux étapes, la prédiction et la correction (également appelée étape de mise à jour). La première étape utilise des états précédents pour prédire l'état actuel. La deuxième étape utilise la mesure actuelle, telle que l'emplacement de l'objet, pour corriger l'état. Le filtre de Kalman implémente un système d'état-espace linéaire à temps discret.

Exemple d'application 1 : Suivi l'emplacement d'un objet physique se déplaçant dans une direction :
- Générer des données synthétiques qui imitent l'emplacement 1-D d'un objet physique se déplaçant à une vitesse constante.
- Simuler les détections manquantes en définissant certains éléments comme vides.
- Créer un filtre Kalman 1-D à vitesse constante lorsque l'objet physique est détecté pour la première fois. Prédire l'emplacement de l'objet en fonction des états précédents.

La figure 2.15 illustre l'utilisation du filtre de Kalman pour prédire la direction d'un objet physique en mouvement.

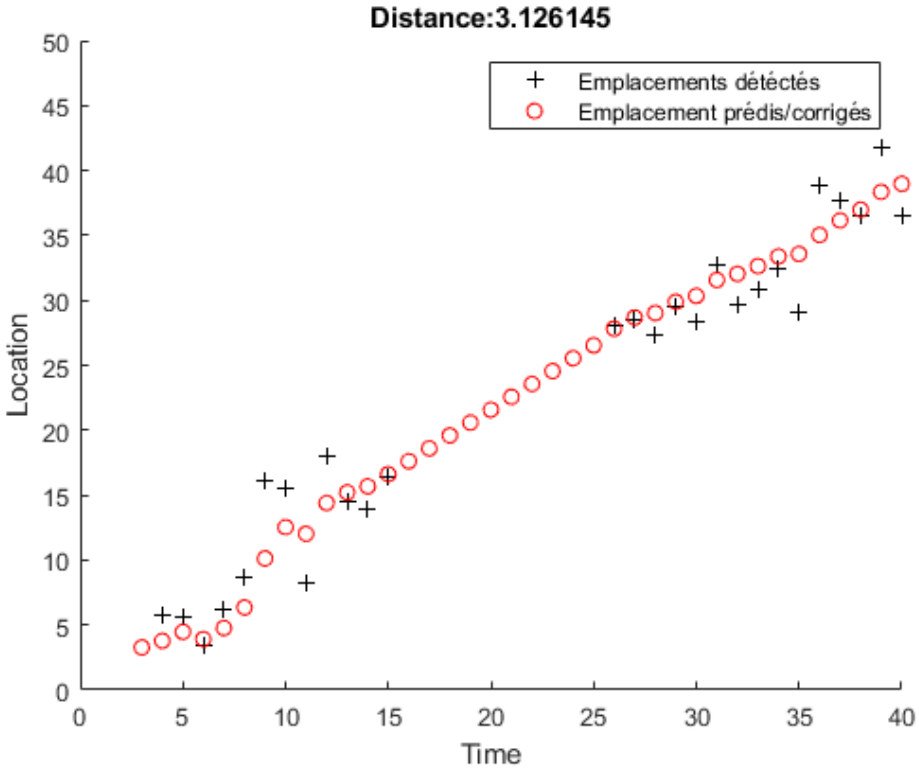

FIGURE 2.15 – Suivi d'un emplacement d'un objet physique.

Exemple d'application 2 : Suivi d'objets multiples et prédiction :

La figure 2.16 illustre l'utilisation du filtre de Kalman combiné à l'algorithme GMM détectant ainsi les objets en mouvement et la prédit leurs emplacements quand ils sont derrière un obstacle.

FIGURE 2.16 – atrium.mp4 filtre de Kalman.

2.3.4 Algorithmes de détection et de classification intelligents

Les humains regardent une image et savent instantanément quels objets se trouvent dans l'image, où ils se trouvent et comment ils interagissent. Le système visuel humain est rapide et précis, ce qui nous permet d'effectuer des tâches complexes comme la conduite avec peu de conscience. Des algorithmes rapides et précis pour la détection d'objets permettraient aux ordinateurs de conduire des voitures sans capteurs spécialisés, permettraient aux appareils d'assistance de transmettre en temps réel informations sur la scène aux utilisateurs humains et de libérer le potentiel pour des systèmes robotiques réactifs à usage général.

Dans cette section, nous présentons deux outils disponibles proposés récemment qui exploitent les réseaux de neurones de convolution pour la détection et la classification d'objets : Faster R-CNN et YOLO.

2.3.4.1 Faster R-CNN

L'algorithme Faster R-CNN créé par S.Ren [18] repose sur une détection entièrement faite avec des réseaux de neurones de convolution (Figure 2.17) :

- Un premier réseau de neurones de convolution (Region proposal Network) prend en entrée une image de taille quelconque et donne en sortie des régions dans lesquelles pourraient se trouver les objets à détecter.
- Un second réseau (Classifier Network) prend en entrée les régions proposées par le premier réseau et recherche si elles contiennent l'objet à détecter.
- La couche "feature map" permet l'identification des différentes caractèristiques présentes dans l'image telles que les bords, les lignes verticales, les lignes horizontales, les virages, etc. On obient grâce à ce réseau des cartes de caractéristiques appelées "feature maps".
- La couche "Rol pooling" prend en entrée plusieurs features maps et applique à chacune d'entre elles l'opération de pooling. L'opération de pooling consiste à réduire la taille des images, tout en préservant leurs caractéristiques importantes. La couche de pooling permet de réduire le nombre de paramètres et de calculs dans le réseau. On améliore ainsi l'efficacité du réseau et on évite le sur-apprentissage.

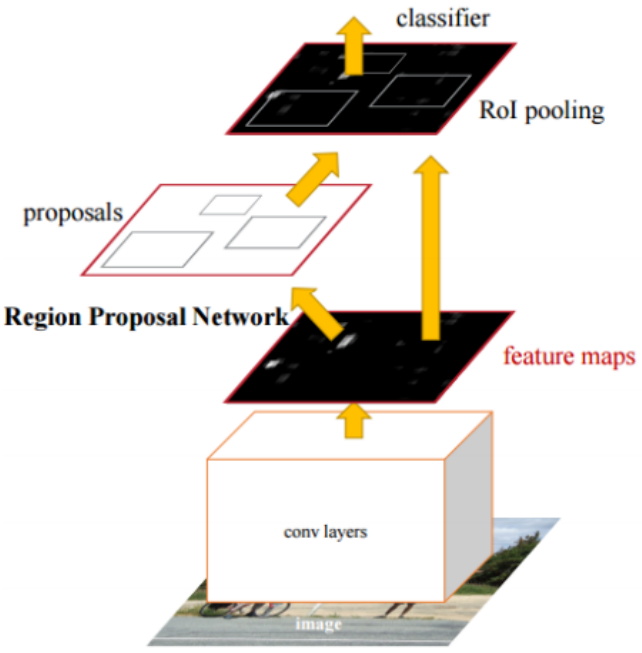

FIGURE 2.17 – Réseau de neurones de convolution.

La figure 2.18 illustre une comparaision du temps de détection des différentes versions l'algorithme R-CNN. En effet Faster R-CNN permet une détection et une classification très rapide ce qui lui permet d'être utilisé en temps réel.

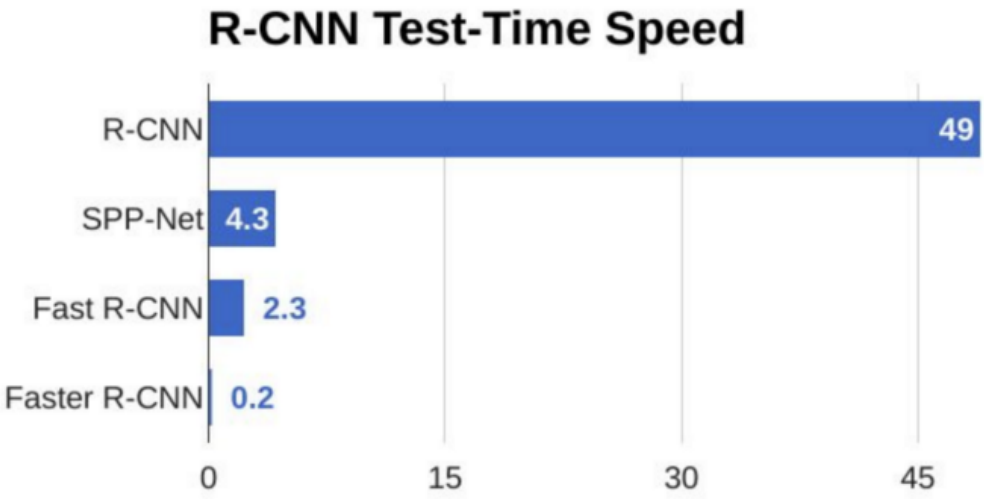

FIGURE 2.18 – Comparaison entre les différentes versions de R-CNN.

2.3.4.2 YOLO

Contrairement à l'algorithme Faster R-CNN qui utilise des régions pour localiser l'objet dans l'image, l'algorithme YOLO créé par Redmon and Farhadi [19] analyse l'image entière et sélectionne les parties de l'image qui ont de fortes probabilités de contenir l'objet. En effet, un seul réseau de neurones de convolution prédit les boîtes (bouding boxes) et les probabilités de classe pour ces boîtes.

L'algorithme YOLO fonctionne de la manière suivante : L'image en entrée est divisée en une grille SxS, et dans chacune des grilles, nous prenons m boîtes englobantes. Pour chacune des boîtes englobantes, le réseau génère une probabilité de classe et des valeurs de décalage pour la boîte englobante. Les boîtes ayant la probabilité de classe supérieure à une valeur de seuil sont sélectionnées et utilisées pour localiser l'objet dans l'image. La figure 2.19 illustre le processus de détection et de classification par l'algorithme YOLO :

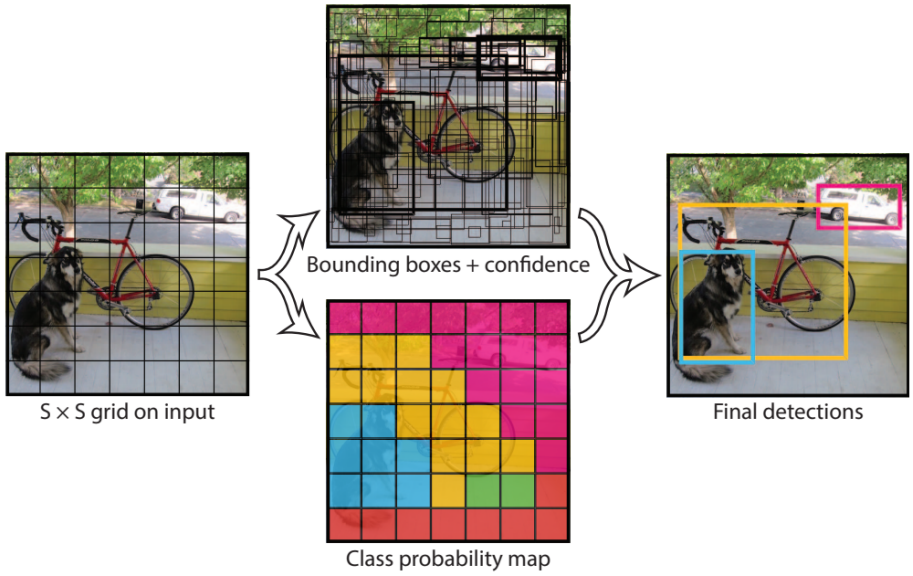

FIGURE 2.19 – Algorithme YOLO.

La figure 2.20 illustre les résultats impressionnants de la dernière version de YOLO, on peut remarquer que l'algorithme arrive à détecter et classifier les objets même avec une scène énormément encombrée. L'algorithme YOLO est aussi très rapide et il peut être également utilisé en temps réel.

FIGURE 2.20 – Détection et classification d'objet par YOLO.

2.4 Conclusion

Nous avons étudié dans ce chapitre les différents algorithmes d'extraction d'arrière-plan (de détection d'objets). Nous nous sommes intéressé aux mélange gaussien, au flot optique et à deux algorithmes de détection d'objets intelligents : YOLO et Faster R-CNN.

Nous avons aussi choisi d'étudier le filtre de Kalman, algorithme capable de prédire les trajectoires des objets détectés avec l'un des algorithmes cités précédemment.

Chaque algorithme a été suivi par un exemple d'application en utilisant la toolbox Computer Vision de Matlab.

Chapitre 3

Synopsis vidéo

3.1 Introduction

Ce chapitre sera consacré à l'assigniation des trajets et à la génération de différents types de synopsis vidéo. On présentera aussi une interface graphique capable de générer un synopsis à partir d'une vidéo choisie par l'utilisateur.

3.2 Assigniation de trajets

Pour construire une détection d'objet robuste, on a opté pour une approche heuristique permettant d'éliminer certains trajets d'objets détectés :

— Tout objet qui dure peu de frames est éliminé.

— Tout objet invisible pendant plus de 20 de frames est aussi éliminé.

Les objets détectés et leurs trajets sont stockés dans des variables, cela aide à séparer les objets et à les différencier pour ensuite les inclure dans le synopsis.

La figure 3.1 illustre les différents objets détectés ainsi que leurs id (identifiants).

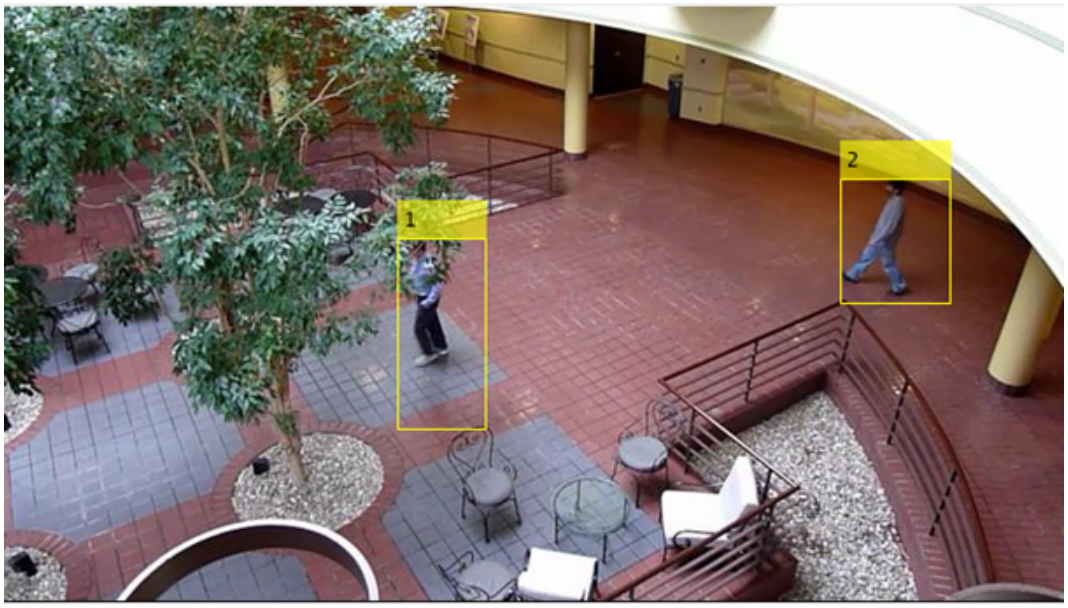

FIGURE 3.1 – Objets détectés avec id.

Après la sauvegarde de chaque objet et des informations relatives à ce dernier, on peut alors afficher les résultats du suivi (tracking en anglais).

La figure 3.2 illustre l'instant d'apparition (en terme de numéro de frame) et de disparition de chaque objet détecté et la figure 3.3 illustre la visibilité de chaque objet. Si l'objet existe ou est visible dans l'image en cours, la variable sera mise à « 1 » sinon elle sera mise à « 0 » pour un objet non existant ou invisible dans l'image en cours.

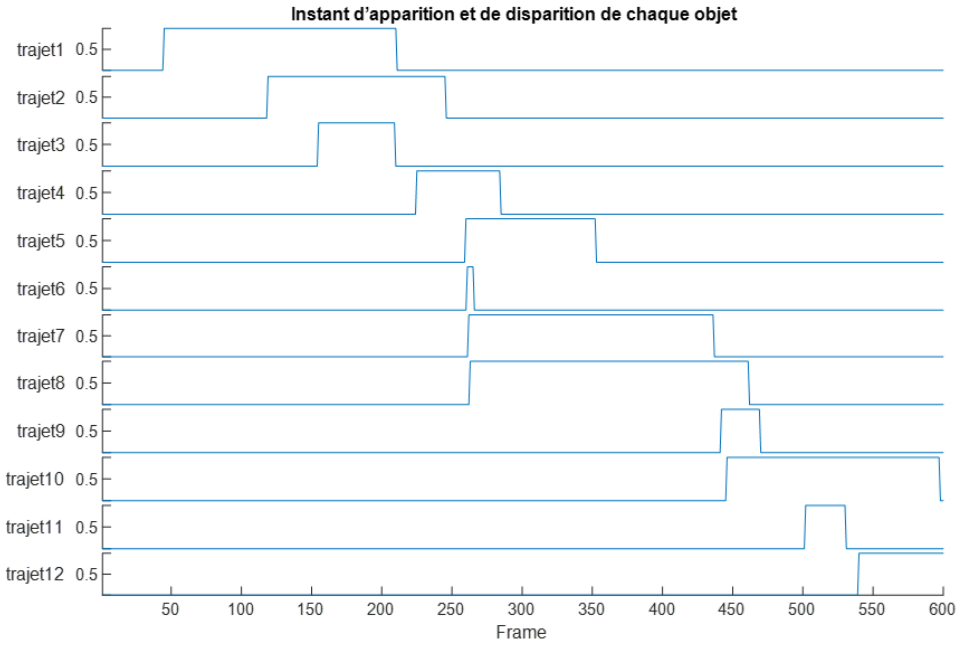

FIGURE 3.2 – Instant d'apparition et de disparition de chaque objet.

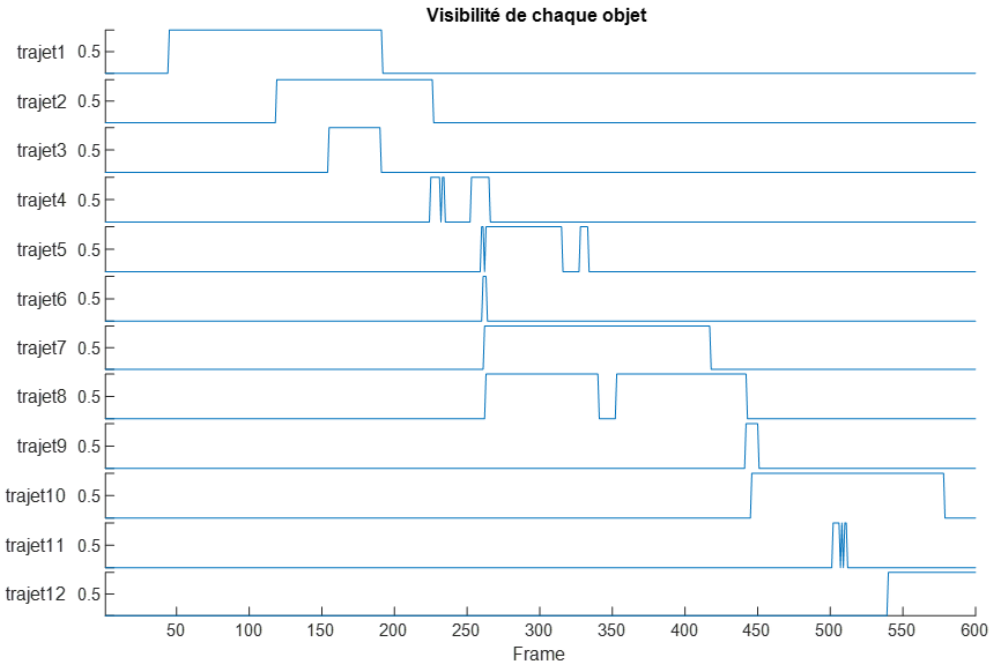

FIGURE 3.3 – Visibilité de chaque objet.

On peut également afficher les trajets des objets sans l'objet en question. La figure 3.4 illustre les trajets des objets ainsi que les trajets prédits en utilisant le filtre de Kalman. Le cercle vide correspond au trajet de l'objet et le cercle plein correspond au trajet prédit.

FIGURE 3.4 – Trajets et trajets prédits des objets détectés.

Quant à la figure 3.5, elle illustre uniquement les trajets fiables de chaque objet détecté à savoir les trajets respectant les conditions citées précédemment. La figure 3.6 illustre la visibilité de chaque trajet fiable.

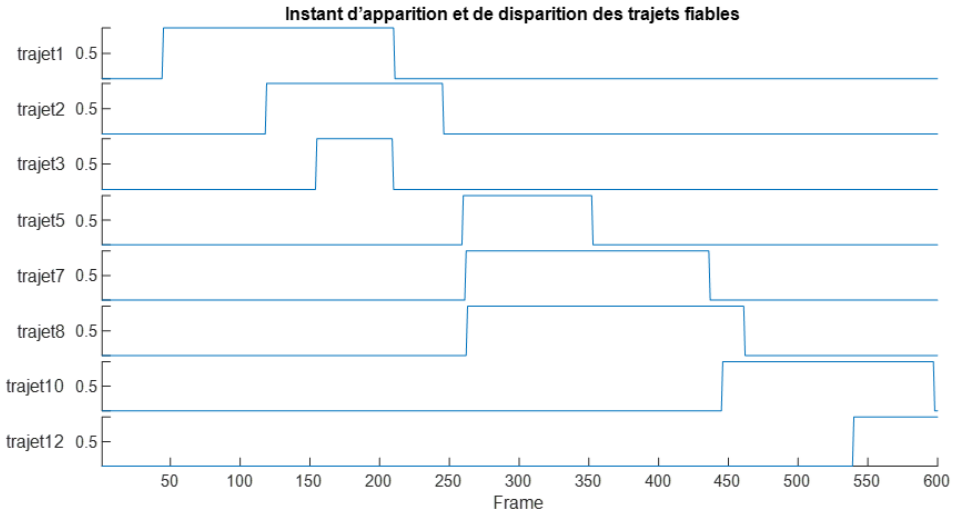

FIGURE 3.5 – Instant d'apparition et de disparition des trajets fiables.

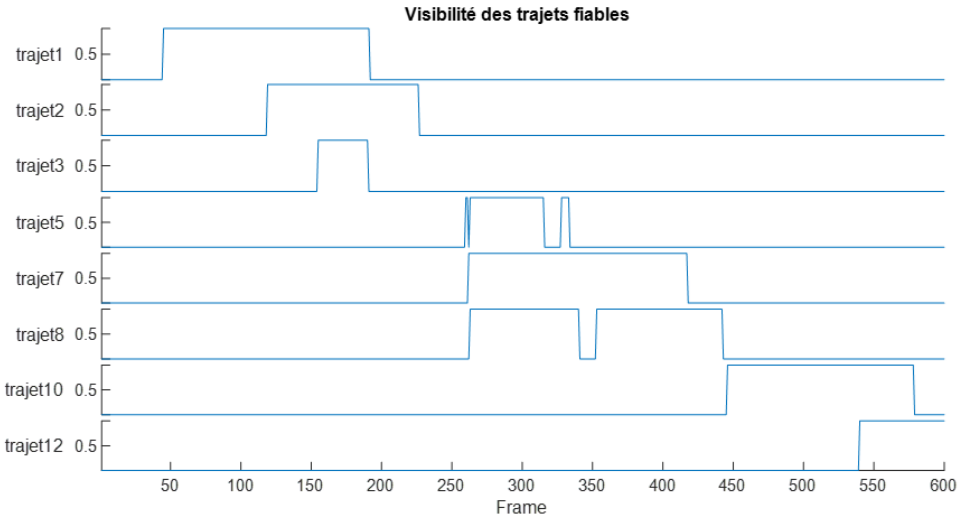

FIGURE 3.6 – Visibilité des trajets fiables.

3.3 Génération du synopsis

Une fois que tous les objets ont été détectés et séparés dans des variables, comme le montre la figure 3.7, le synopsis pourrait être construit dans le temps ou dans l'espace. L'approche de remplir le synopsis dans l'espace préserve l'ordre spatial des objets, le temps est également maintenu.

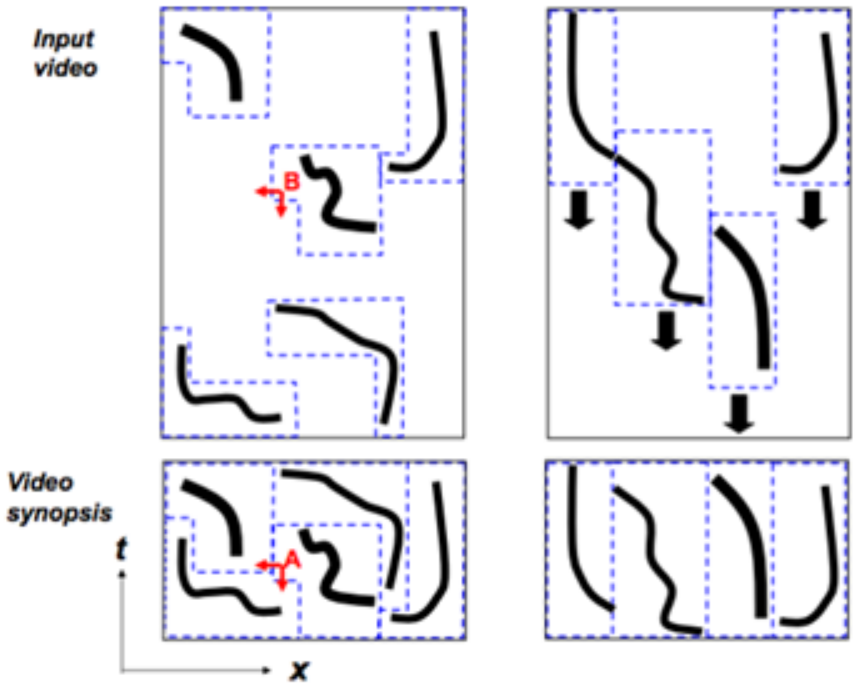

FIGURE 3.7 – Synopsis dans le temps (G) et synopsis dans l'espace (D).

La longueur du synopsis qu'on veut générer est déterminée par la plus longue activité dans la vidéo originale, en d'autres termes : elle est déterminée par la période de l'objet qui apparait pendant plus d'images. Il existe par contre d'autres types de synopsis qui ne respectent pas forcément ce principe.

La première image de la vidéo est utilisée comme image de référence pour construire le synopsis (on suppose que la première image ne contient aucun objet). Étant donné que le synopsis est construit uniquement pour une caméra fixe, les objets détectés peuvent être projetés au-dessus de l'arrière-plan de référence.

3.3.1 Synopsis par ségmentation

Le synopsis présenté dans cette partie est généré en utilisant une segmentation temporelle de la vidéo traitée : l'idée est de compresser la vidéo avec un ratio choisi. Si on utilise un ratio de 1/5 pour une vidéo de 20 secondes, on obtiendra un synopsis de 4 secondes, ce qui équivaut à un cinquième de la vidéo originale. Après la segmentation, on superpose tous objets détectés de chaque segment sur l'arrière-plan.

La figure 3.8 illustre le résultat obtenu en utilisant la méthode de segmentation ainsi.

FIGURE 3.8 – Synopsis par ségmentation.

Cependant, si un objet est présent dans deux segments il apparaîtra dans deux endroits différents dans le synopsis. L'objet entouré en blanc dans la figure 3.8 est en fait le même objet présent dans deux segments successifs.

3.3.2 Génération de synopsis d'un trajet spécifique

Il est possible de générer une vidéo contenant uniquement le passage d'un objet spécifique sélectionné. Cela peut aider à mieux visionner le comportement d'un seul objet.

La qualité du synopsis généré dépend énormément de la précision de détection et de l'assignation des trajets.

La figure 3.9 illustre une vidéo contenant la deuxième personne (le deuxième objet).

FIGURE 3.9 – Suivi d'un seul objet.

3.3.3 Synopsis par fusion

Cette méthode consiste à afficher sur un même arrière-plan tous les objets détectés présents sur la vidéo indépendamment du temps de leurs apparition. C'est la méthode la plus efficace car la durée de la vidéo générée par le synopsis dépend de la plus longue activité dans la vidéo originale.

La figure 3.10 illustre le synopsis généré par la fusion.

FIGURE 3.10 – Synopsis par fusion.

3.4 Ineterface Graphique

Pour conclure notre travail, on a opté pour la création d'une interface graphique MATLAB capable de générer un synopsis vidéo à partir d'une vidéo choisie par l'utilisateur et ce, en utilisant App Designer.

App Designer permet de créer des applications sans avoir à être un développeur de logiciel professionnel. Son utilisation est très facile car il suffit de glisser et de déposer les composants visuels pour concevoir une interface graphique (GUI) et utiliser l'éditeur intégré pour programmer rapidement son comportement.

App Designer est une alternative à l'ancien outil de création d'interface graphique GUIDE introduite depuis 2016 à MATLAB.

La figure 3.11 illustre les différents étéments d'une ineterface graphique App Designer.

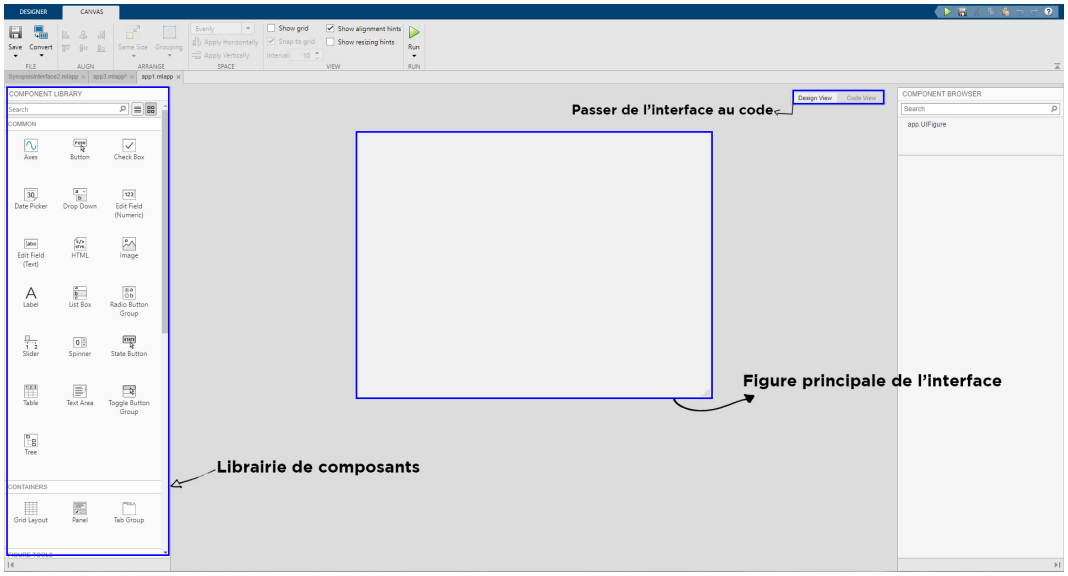

FIGURE 3.11 – App Designer.

La figure 3.12 illustre l'interface créée pour notre projet.

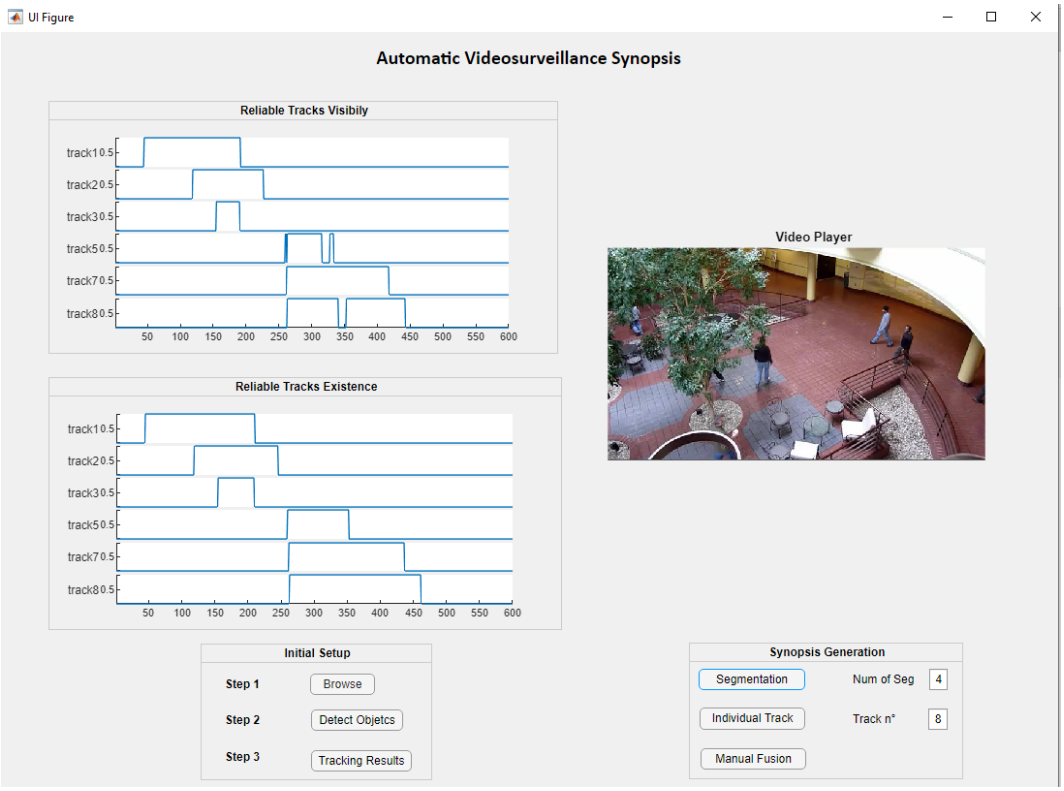

FIGURE 3.12 – Interface synopsis vidéo.

Elle est composée des éléments suivants :

- Partie réservée à l'affichage de la visibilité de chaque trajet (objet) fiable (Reliable Tracks Visibility).
- Partie réservée à l'affichage du temps d'apparition et de disparition de chaque trajet fiable (Reliable Tracks Existence).
- Un lecteur vidéo qui affichera le synopsis généré.
- Un bouton « Browse » (Parcourir) permettant de sélectionner un fichier vidéo.
- Un bouton « DetectObjets » qui appliquera l'algorithme GMM ainsi qu'un filtre de Kalman à la vidéo sélectionnée.
- Un bouton « Tracking Results » permettant d'afficher les résultats du suivi.
- Un bouton « Segmentation » qui génère un synopsis utilisant la méthode de segmentation, on peut aussi choisir le nombre de segments (Plus le nombre est élevé, plus la vidéo résultante est courte).
- Un bouton « Individual Track » générant une vidéo contenant l'objet souhaité parmi les objets détectés auparavant.
- Et enfin un bouton « Manual Fusion » qui génère un synopsis vidéo contenant tous les

objets détectés et commençant tous par le premier frame.

3.5 Conlusion

Nous avons présenté au cours de ce chapitre les différents types de synopsis qu'on a pu générer ainsi que les désavantages de certains. On a donc réussi à atteindre le but de notre projet : raccourcir le temps nécessaire pour visionner une longue séquence de vidéosurveillance.

Nous avons également créé une interface graphique capable de générer un synopsis à partir d'une vidéo choisie par l'utilisateur.

Conclusion générale et perspectives

Le synopsis vidéo a été proposé comme approche pour condenser l'activité dans une vidéo. Cette représentation permet d'analyser les activités dans les séquences vidéo de manière plus efficace et rapide.

Le choix de l'algorithme utilisé impacte énormément le temps qu'il faut pour générer le synopsis vidéo. Pour améliorer notre détection, nous avons choisi d'utiliser le filtre de Kalman permettant de prédire les trajectoires des objets détectés. L'activité dans les synopsis vidéo résultants est bien plus condensée que l'activité dans une vidéo ordinaire, mais lorsque l'objectif principal étant de visionner et d'analyser beaucoup d'informations en peu de temps, le synopsis vidéo répond à cet objectif.

Bien que nos implémentions du synopsis vidéo sont assez simples, des améliorations peuvent être ajoutées pour rendre le synopsis plus performant. Alors, en guise de perspective, on peut envisager l'utilisation d'un algorithme de détection et classification intelligent à l'instar d'un algorithme de détection classique. Ceci permettra de générer un synopsis par classe, c'est-à-dire qu'on aura la possibilité d'avoir une vidéo contenant le type d'objet souhaité (personne, véhicule, type de véhicule ... etc).

Bibliographie

[1] David Barret, *One surveillance camera for every 11 people in Britain, says CCTV survey*,The Telegraph, https ://www.telegraph.co.uk/technology/10172298/One-surveillance-camera-for-every-11-people-in-Britain-says-CCTV-survey.html, Consulté le 24/04/2020.

[2] I. Haritaoglu, D. Harwood, et L. S. Davis, *Real-time surveillance of people and their activities*, IEEE Transactions on Pattern Analysis and Machine Intelligence", vol. 22, no.8, pp.809–830, 2000.

[3] E. Bennett et L. McMillan, *Computational time-lapse video*, In SIGGRAPH 07, 2007.

[4] Rav-Acha, Alex, Yael Pritch, et Shmuel Peleg, *Making a long video short : Dynamic video synopsis*, IEEE Computer Society Conference vol. 1. IEEE, 2006.

[5] Rav-Acha, Alex, Yael Pritch, et Shmuel Peleg, *Clustered synopsis of surveillance video*, Sixth IEEE International Conference on. IEEE, 2009.

[6] Rav-Acha, Alex, Yael Pritch, et Shmuel Peleg, *Webcam synopsis : Peeking around the world*, IEEE 11th International Conference on. IEEE, 2007.

[7] Rav-Acha, Alex, Yael Pritch, et Shmuel Peleg, *Nonchronological video synopsis and indexing*, IEEE Transactions, 2008.

[8] BriefCam, *Transforming video into actionable intelligence*, https ://www.briefcam.com/, Consulté le 24/04/2020.

[9] Sen-Ching S. Cheung et Chandrika Kamath, *Robust Techniques for Background Subtraction in Urban Traffic Video*, EURASIP Journal on Applied Signal Processing, vol. 2005, pp. 2330-2340, 2005.

[10] C. Stauffer, et W. Grimson, *Adaptive background mixture models for real-time tracking*, Computer Vision and Pattern Recognition, vol. 2, pp. 246-252, 1999.

[11] BriefCam Syndex, *Pedestrian overpass - original video (sample)*, https ://www.youtube.com/watch ?v=aUdKzb4LGJIfeature=embtitleabchannel=BriefCam, Consulté le 02/06/2020.

[12] Sepehr Aslani et Homayoun Mahdavi-Nasab, *Optical Flow Based Moving Object Detection and Tracking for Traffic Surveillance*, International Journal of Eletrical, Computer, Energetic, Electronic and Communication Engineering vol 7, n9,2013.

[13] Bruce D.Lucas et Kanade Takeo, *An Iterative Image Registration Technique with an Application to Stereo Vision*, DARPA Image Understanding Workshop, pp. 121–130, 1981.

[14] Silar, Z., et M. Dobrovolny, *Comparison of two optical flow estimation methods using Matlab*, 2011 International Conference on. IEEE, 2011.

[15] B.K.P. Horn et B.G. Schunck., *Determining Optical Flow*, Artificial Intelligence, vol. 17, pp. 185–204, 1981.

[16] Farneback. G, *Two-Frame Motion Estimation Based on Polynomial Expansion*, 13th Scandinavian Conference on Image Analysis, pp. 363-370., 2003.

[17] R.E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*, Journal of Basic Engineering, pp. 35-45, 1960.

[18] Ren S., He K., Girshick R. et Sun J., Faster R-CNN : Towards real-time object detection with region proposal networks, Advances in Neural Information Processing Systems 28, pp 91–99, 2015.

[19] Redmon J. et Farhadi A., You Only Look Once : Unified, Real-Time Object Detection, University of Washington, 2015.

Résumé

Le visionnage et l'analyse de longues séquences de vidéosurveillance sont peu pratiques en raison des redondances spatio-temporelles, où certains intervalles de temps peuvent avoir aucune activité.

Ce travail consiste à générer un synopsis vidéo capable de fournir une représentation compacte tout en préservant toutes les activités essentielles de la vidéo originale. Nous présentons le synopsis vidéo comme une solution où les activités de la vidéo originale sont condensées en affichant simultanément plusieurs actions, indépendemment de leurs temps d'apparition. La vidéo générée est considéreblement plus courte, mais l'activité y est beaucoup plus dense.

Abstract

Browsing long video surveillance footage is inconvenientt due to spatio-temporal redundancies, where some time intervals may have no activity.

This work consists of generating a video synopsis capable of providing a compact representation while preserving all the essential activities of the original video. We present the video synopsis as a solution where the activities of the original video are condensed by simultaneously displaying several actions regardless of the original time of their apparition. The activity is shifted into a significantly shorter period, but the activity is much denser.