UNIVERSITÉ DE MONTRÉAL

SUIVI MULTIOBJET DE TRAFIC MIXTE URBAIN

JEAN-PHILIPPE JODOIN
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
DÉCEMBRE 2013

© Jean-Philippe Jodoin, 2013.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

SUIVI MULTIOBJET DE TRAFIC MIXTE URBAIN

présenté par : JODOIN Jean-Philippe
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. LANGLOIS J.M. Pierre, Ph.D., président
M. BILODEAU Guillaume-Alexandre, Ph.D., membre et directeur de recherche
M. SAUNIER Nicolas, Ph.D., membre et codirecteur de recherche
M. PAL Christopher J., Ph.D., membre

À Michel, Yolande, Alexandre et Marilyne.
Merci de votre soutien.

REMERCIEMENTS

En préambule de ce mémoire, je veux d'abord remercier mes codirecteurs de recherche Guillaume-Alexandre Bilodeau et Nicolas Saunier qui m'ont épaulé pour toute la durée de ma maîtrise, des demandes de bourses à la rédaction de mon mémoire. Vos expertises m'ont été précieuses dans le développement de ma solution et je n'y serais pas arrivé sans vous. Je voudrais également remercier le Fonds québécois de la recherche sur la nature et les technologies (FQRNT) et le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) pour leur financement qui m'a permis de me concentrer sur ma recherche au cours des deux dernières années.

Je remercie Frédéric Mailhot et Pierre Pontevia qui m'ont conseillé en plus de m'écrire une lettre de recommandation pour mes demandes de bourses. Je remercie également mes professeurs de l'Université de Sherbrooke et de l'École Polytechnique de Montréal pour tout ce qu'ils m'ont appris.

Je salue mes collègues Wassim Bouachir, Dorra Riahi, Tanushri Chakravorty et Pier-Luc Saint-Charles. Nos discussions m'ont été utiles en plus d'ajouter une dimension sociale qui peut parfois manquer en recherche.

Je remercie mes amis qui m'ont toujours inspiré par leur intelligence et leur ambition. Je remercie ma copine Marilyne Brosseau qui a enduré mes (trop) nombreuses heures de travail et qui m'a aidé à rester sain d'esprit. Je remercie mes parents et mon frère qui m'ont toujours supporté. Votre détermination, votre passion et votre courage m'inspirent chaque jour.

Plus spécialement, je remercie le personnel de l'Hôpital général juif sans qui je n'aurais probablement pas pu finir cette maîtrise. Finalement, je suis redevable à la société québécoise qui m'a permis de réaliser des études supérieures grâce à des subventions et des frais de scolarité accessible.

RÉSUMÉ

Notre recherche porte sur le suivi d'objets en milieux urbains. Elle vise l'extraction de la trajectoire des différents usagers de la route circulant à des intersections urbaines à des fins d'analyse de la sécurité routière. De nombreux accidents pourraient être évités chaque année en améliorant la géométrie et la signalisation des intersections en milieux urbains. Le problème est qu'il y a très peu de données concernant les accidents et aucune donnée concernant les situations dangereuses n'ayant pas engendré des accidents à une intersection. Il faut donc attendre qu'un ou plusieurs accidents graves surviennent afin d'avoir assez de données (via des rapports de police par exemple) pour améliorer la sécurité des intersections. Filmer les intersections et observer les interactions entre les usagers de la route afin d'identifier les problèmes n'est pas une solution viable vu le grand nombre d'heures de visionnement nécessaire pour observer une situation dangereuse. L'extraction automatique des trajectoires des usagers de la route à l'aide de la vision par ordinateur, suivie d'une analyse des trajectoires permettrait donc l'analyse d'une grande quantité de données et de localiser les interactions dangereuses entre ceux-ci. Les informations ainsi récoltées permettraient aux ingénieurs en transport de proposer des améliorations aux intersections avant qu'il n'y survienne des accidents. Cela pourrait donc permettre de réduire le nombre d'accidents routiers et les nombreux coûts sociaux économiques qui y sont rattachés.

Ce travail se concentre sur l'extraction des différents usagers de la route et leur suivi afin d'obtenir leur trajectoire. Il ne traite pas de l'analyse des interactions entre les différents usagers. Pour ce travail, trois objectifs ont été fixés. D'abord, il faut détecter les différents objets en mouvement de la séquence vidéo tout en filtrant les objets qui ne sont pas pertinents tels que les feuilles d'arbres ou encore les ombres d'oiseaux. Ensuite, le second objectif consiste à suivre les objets détectés de trame en trame afin d'obtenir la trajectoire empruntée par ceux-ci. Finalement, le dernier objectif est de valider la qualité des trajectoires extraites à l'aide de métriques standard et de comparer nos résultats avec ceux obtenus pour un autre algorithme.

Pour la détection des objets, de nombreuses méthodes sont documentées dans la littérature. Il y a les méthodes basées sur la soustraction d'arrière-plan qui permettent de détecter les différents objets en mouvement à l'aide des changements d'intensité des pixels. Il faut ensuite relier les pixels sous forme de blobs [1]. L'avantage de ce type de méthodes est qu'elles permettent de détecter tous les objets sans connaissance préalable de ceux-ci. Toutefois ces méthodes présentent une certaine sensibilité aux ombres et aux changements de luminosité, ce qui peut poser plusieurs problèmes de segmentation. Cette technique ne fonctionne qu'à

1. Ensemble de pixels interconnectés spatialement.

partir d'une caméra statique. De plus, les blobs seuls ne nous permettent pas de gérer la présence de plusieurs objets très près l'un de l'autre puisque ceux-ci sont détectés comme faisant partie du même blob. Une autre méthode populaire est l'utilisation d'un détecteur. Celle-ci a l'avantage de permettre l'utilisation d'une caméra en mouvement et de gérer les occultations. Celle-ci est d'ailleurs très populaire dans les approches de suivi par détection, principalement pour le suivi de piétons se déplaçant en groupe. L'inconvénient est qu'il est difficile de faire des détecteurs pour des objets qui changent de forme en fonction de l'angle ou du modèle. Puisque nous voulions supporter l'ensemble des utilisateurs de la route, nous avons utilisé une soustraction d'arrière-plan. Afin de réduire les problèmes de celle-ci, nous avons utilisé des opérateurs morphologiques et des filtres.

Pour le suivi d'objets, nous utilisons les blobs détectés précédemment. Dans un premier temps, nous tentons de mettre les blobs ensemble d'une trame à la suivante en utilisant un modèle de suivi. Pour ce modèle de suivi, plusieurs options s'offraient à nous comme l'histogramme de couleur des objets, les points caractéristiques, la forme, la position, le modèle de mouvement des objets, etc. Nous avons choisi un modèle basé principalement sur les points caractéristiques et la forme des objets. Les points caractéristiques ont été choisis puisqu'il s'agit d'un modèle très distinctif et qui n'impose aucune restriction sur le mouvement des objets. Afin de contrer les erreurs possibles, nous exigeons que les paires de points trouvés avec des techniques d'association de points caractéristiques répondent à certains critères spécifiques. Puisqu'il n'y a pas de points sur les bordures, nous utilisons également le recouvrement de blob comme mesure d'association. Pour le suivi d'objet, nous associons les blobs aux objets existants à l'aide du modèle de suivi de ceux-ci. Les divers problèmes du suivi sont gérés par une machine à état. Nous avons également développé une technique d'estimation de la position des objets dans des blobs sous-segmentés. Cette technique utilise les points caractéristiques et les observations a priori des objets pour estimer la position probable de l'objet courant.

Afin d'évaluer nos résultats, nous avons utilisé 5 vidéos, dont 4 vidéos urbaines réelles. Nous avons utilisé des métriques standards pour le suivi multiobjet afin d'obtenir une évaluation quantifiable de nos performances. Les résultats obtenus ont été comparés à un autre algorithme de suivi urbain. Ils démontrent que notre méthode proposée présente de nombreux avantages par rapport à l'autre algorithme tant au niveau de la précision, de la justesse et de la complétude du suivi. Notre méthode est d'ailleurs capable de suivre les objets indépendamment de leur type ou de leur taille en nécessitant peu ou pas d'ajustement de paramètres.

ABSTRACT

Our research focuses on the challenge of detecting and tracking multiple objects of various types in outdoor urban traffic scenes using a static video camera. The resulting system aims at collecting object trajectories for road safety analysis. Numerous crashes and dangerous situations could be avoided by changing intersection geometry and signalisation. The main problem is that there is very few available data related to crashes and none for dangerous situations. Road safety researchers must wait for casualties or accidents to happen in order to improve intersection safety. Using a video camera and manually watching thousands of hours of road users interaction is not really an option because of the high amount of resources required. The automatic extraction of user trajectories using a computer vision method combined with trajectory interaction analysis is a way to avoid this issue by enabling the analysis of a great amount of data at much lower cost. This would also allow the detection of dangerous interaction for which there are currently no data available. This information could help transport engineers solve critical issues in the road infrastructure before casualties happen. This could significantly reduce social-economic costs related to road casualties and accidents.

This work is about the detection of road users and tracking them in order to extract their trajectories. The trajectory analysis will not be covered by this work. In this work, three main objectives were defined. First, we want to be able to detect the various road users while avoiding uninteresting objects such as bird shadows and leaves flowing in the wind. The second objective is to follow the detected objects in order to be able to extract their trajectories. Finally, the last objective is to use standard metrics in order to validate the quality of extracted trajectories and compare them with another algorithm.

For object detection, numerous algorithms exist in the literature. Some methods are based on background subtraction which allows them to detect moving object using pixel intensity changes. The spatial connection of these pixels are then analysed in order to find blobs [2]. The advantage of this method is that it does not require any prior knowledge of the objects. The drawbacks are that the camera must remain static, shadows deform blobs and lighting changes can cause fragmentation issues in the blobs. It does not allow the detection of the presence of multiple objects inside one blob either. Another popular method is based on object detectors. This method does not require a static camera, there are no fragmentation issue and they do not detect shadows as part of the object. The inconvenient is that these methods handle with difficulty multiple object size and they require the training of a detector

2. Spatially connected group of pixels

for each object shape we want to detect. This works fine for the tracking of pedestrians inside a group, but the use of a detector to track cars is a different issue due to the numerous shapes of car available. Also, the shape of a car changes a lot depending on the angle of view, which means that multiple detectors are required for each car shape. In order to support all type of road users, we have decided to use background subtraction and morphological operators and filters in order to reduce the amount of fragmentation.

For object tracking, we use the blobs detected previously. We first track the blob from frame to frame. After that, we try to associate the blobs to existing objects. In order to track the blobs, we use a tracking model. Numerous features can be used in a tracking model such as color histogram, feature points, shape, position, motion model etc. We have chosen to use a model based on feature points and object shape. The feature points are the main part of the tracking model since they offer a very distinctive description of objects and they do not constraint the motion of objects. In order to solve the errors occurring during point tracking, the feature points require a certain amount of distinctiveness compared to others points and we require multiple points match to associate to blob observation. Since there are no points on image borders, we also use object shape (using blob overlap) to associate objects together. To improve object localization in case of partial occlusion, we also propose a bounding box estimation method.

The tracking results were evaluated using five videos. Four of those videos are real urban sequences. Those videos were annotated and we have then used standard metrics of multiple object tracking in order to quantify the performance of our algorithms. The results were compared to another state of the art urban tracking algorithm and our method is shown to have better results for precision and accuracy. This is mainly due to the capability of our algorithm to follow multiple objects of various type and size at the same time. Our results show balanced performance for the tracking of all road users.

TABLE DES MATIÈRES

DÉDICACE iii

REMERCIEMENTS iv

RÉSUMÉ v

ABSTRACT vii

TABLE DES MATIÈRES ix

LISTE DES TABLEAUX xii

LISTE DES FIGURES xiv

LISTE DES ANNEXES xviii

LISTE DES SIGLES ET ABRÉVIATIONS xix

CHAPITRE 1 INTRODUCTION 1
1.1 Définitions et concepts de base 2
1.2 Problématique 3
1.3 Objectifs 3
1.4 Aperçu de la méthode proposée 4
1.5 Contributions 4
1.6 Plan du mémoire 5

CHAPITRE 2 REVUE DE LITTÉRATURE 6
2.1 Suivi par blob 6
2.1.1 Explication de la méthode 6
2.1.2 Soustraction d'arrière-plan 6
2.1.3 Formation des blobs 9
2.1.4 Méthodes d'association des observations et modèle de suivi 9
2.2 Suivi par points 10
2.2.1 Explication de la méthode 10
2.2.2 Méthodes de détections de points caractéristiques 11
2.2.3 Méthodes existantes de descriptions de points et calcul de correspondance 14

2.2.4 Méthode de regroupement des points 18
2.3 Suivi par flux optique 18
2.4 Suivi par détection 19
2.4.1 Méthode générale 20
2.4.2 Application aux piétons 21
2.4.3 Généralisation pour gérer des points de vue multiples 22
2.4.4 Algorithmes de suivi 23
2.5 Application du suivi au domaine du transport 24
2.5.1 Application 24
2.5.2 Suivi sur autoroute 24
2.5.3 Suivi en milieu urbain 25
2.6 Gestion de la perspective 27
2.6.1 Correction de la perspective 28
2.6.2 Suivi multi caméras 28
2.7 Retour sur les méthodes 29

CHAPITRE 3 MÉTHODOLOGIE 31
3.1 Terminologie utilisée et définitions de base 31
3.2 Méthode développée 32
3.2.1 Aperçu de la méthode 33
3.3 Extraction des blobs 34
3.3.1 Soustraction d'arrière-plan 34
3.3.2 Analyse des composantes connexes 35
3.3.3 Filtrage des blobs 35
3.3.4 Calcul du modèle des blobs 38
3.4 Suivi 39
3.4.1 Suivi des blobs 39
3.4.2 Suivi des objets 40
3.4.3 États des objets 41
3.4.4 Gestion de la sous-segmentation 43
3.4.5 Gestion de la sursegmentation 45
3.4.6 Gestion des objets sortants 46
3.4.7 Gestion des objets perdus 48
3.4.8 Gestion des ombres projetées 48
3.4.9 Mise à jour du modèle 50
3.5 Paramètres de l'algorithme 50

CHAPITRE 4 RÉSULTATS ET DISCUSSION 51
4.1 Métriques 51
4.2 Outils développés 53
4.2.1 Outil de génération de vérité terrain 53
4.2.2 Outil d'évaluation des métriques 54
4.3 Résultats 55
4.3.1 Paramètres utilisés 56
4.3.2 Sherbrooke 56
4.3.3 Rouen 59
4.3.4 St-Marc 61
4.3.5 René-Lévesque 62
4.3.6 Atrium 64
4.3.7 Impact du seuil d'association de CLEAR MOT 67
4.3.8 Limitations de l'algorithme 67
4.3.9 Sensibilité des paramètres 69

CHAPITRE 5 CONCLUSION 71
5.1 Améliorations futures possibles 72

RÉFÉRENCES 74

ANNEXES 79

LISTE DES TABLEAUX

Tableau 3.1 Paramètres de l'algorithme 50

Tableau 4.1 Longueur maximum d'association par vidéo pour la métrique CLEAR MOT 55

Tableau 4.2 Paramètres utilisés pour chaque vidéo pour UT 56

Tableau 4.3 Paramètres utilisés pour chaque vidéo pour TI 57

Tableau 4.4 Métriques CLEAR MOT obtenues pour la vidéo Sherbrooke en utilisant un seuil d'association de 90 pixels. Les résultats sont rapportés pour tous les types d'objets (véhicules et piétons) ensemble et séparément. Les meilleurs résultats sont rapportés en gras. 58

Tableau 4.5 Métriques CLEAR MOT obtenues pour la vidéo Sherbrooke en utilisant un seuil d'association de 90 pixels. Les résultats sont rapportés pour tous les types d'objets (véhicules et piétons) ensemble et séparément. Les piétons se déplaçant en groupe sont ici considérés comme étant un seul objet. Les meilleurs résultats sont rapportés en gras. 58

Tableau 4.6 Métrique CLEAR MOT pour la vidéo Rouen calculée avec une distance maximale d'association de 164 px. Les meilleurs résultats sont rapportés en gras. 60

Tableau 4.7 Métrique CLEAR MOT pour la vidéo St-Marc calculée avec une distance maximale d'association de 113 px. Les meilleurs résultats sont rapportés en gras. 62

Tableau 4.8 Métrique CLEAR MOT pour la vidéo René-Lévesque calculée avec une distance maximale d'association de 24 px. Les meilleurs résultats sont rapportés en gras. 64

Tableau 4.9 Métrique CLEAR MOT pour la vidéo Atrium calculé avec une distance maximale d'association de 92 px. Les meilleurs résultats sont rapportés en gras. 65

Tableau 4.10 Résultats obtenus en utilisant les mêmes paramètres pour toutes les vidéos. ΔA et ΔP représente respectivement la différence entre les MOTA et MOTP obtenus pour les paramètres communs et les paramètres ajustés présentés précédemment. Pour ΔA un signe négatif signifie une baisse de qualité du suivi alors que pour ΔP, un signe positif présente une baisse de la précision du suivi. 70

Tableau A.1 Table *object* 79

Tableau A.2 Table *objects_type* 79
Tableau A.3 Types existant 80
Tableau A.4 Table *positions* 80
Tableau A.5 Table *objects_features* 80
Tableau A.6 Table *bounding_boxes* 81

LISTE DES FIGURES

Figure 2.1 Exemple de soustraction d'arrière-plan : a) Image originale b) Image obtenue après la soustraction d'arrière-plan 7

Figure 2.2 a) Image originale b) Image de référence c) Image après application de la méthode intuitive avec un seuil faible d) Image après application de la méthode intuitive avec un seuil élevé 7

Figure 2.3 Calcul des différences de gaussiennes 13

Figure 2.4 Voisinage d'un point considéré par FAST. En bleu, on a une ligne représentant les pixels plus pâles que le pixel central. Image tirée de Rosten et Drummond (2006). Reproduit avec permission. 14

Figure 2.5 Multiple octave d'un point avec interpolation des scores FAST. Figure tirée de Leutenegger *et al.* (2011). © 2011 IEEE, reproduit avec permission. 15

Figure 2.6 Plan d'échantillonnage du descripteur. a) SIFT b) BRISK c) FREAK. Pour b et c, les grands cercles représentent la gaussienne autour des échantillons (montrés par un point). a) Figure tirée de Lowe (2004) (figure 7, page 15), © 2004 Springer Science and Business Media, reproduite avec permission. b) Figure tirée de Leutenegger *et al.* (2011), © 2011 IEEE, reproduite avec permission. c) Figure tirée de Alahi *et al.* (2012), © 2012 IEEE, reproduite avec permission. 17

Figure 2.7 Exemple du flux optique 19

Figure 2.8 Exemple de point de vue supporté par un même détecteur de piéton. Modèle 3D tiré du jeu L.A. Noire, ©2011 Rockstar Games 21

Figure 2.9 a) Positionnement des points avec l'algorithme de Saunier et Sayed (2006) b) Points valides considérés dans le groupement c) Groupement de points résultants 26

Figure 2.10 a) Positionnement des points avec l'algorithme de Saunier et Sayed (2006) b) Groupement de points résultants 27

Figure 2.11 a) Vue en perspective obtenue par la caméra avec des annotations dessinées. b) Vue aérienne avec les annotations transformées à l'aide de la matrice d'homographie. L'image aérienne est tirée de Bing Maps, ©2013 Microsoft Corporation. 29

Figure 3.1 Problème d'une méthode à base de points. a) Trajectoire stationnaire non groupée b) Problème de groupement 32

Figure 3.2 Schéma bloc du système de suivi 34
Figure 3.3 **a)** Trame originale **b)** Trame après application du filtre gaussien **c)** Soustraction d'arrière-plan à partir de l'image a) **d)** Soustraction d'arrière-plan à partir de l'image b) **e)** Résultat de l'application d'une dilatation sur c) **f)** Résultat de l'application d'une dilatation sur d) **g)** Résultat de la suppression des blobs dont l'aire est inférieure à 200 px dans e) **h)** Résultat de la suppression des blobs dont l'aire est inférieure à 200 px dans f) 36
Figure 3.4 a) Un véhicule immobilisé depuis longtemps au temps $t = 0$. b) Le véhicule au temps $t = 10$. Celui-ci est encore présent dans le modèle d'arrière-plan, car le modèle ne s'est pas encore adapté à sa disparition. On a donc deux blobs dans les pixels d'avant-plan, un pour l'objet en déplacement et un blob «fantôme» à l'ancien emplacement du véhicule. c) Véhicule au temps $t = 10$, mais avec notre suppression des blobs statiques. Pour ce faire, nous calculons la variation interpixels de chaque blob, c'est-à-dire le pourcentage de pixels qui ont une variation suffisante d'intensité. Si ce pourcentage est suffisamment grand, alors le modèle d'arrière-plan est mis à jour avec les valeurs courantes de l'image pour ce blob ce qui élimine le blob «fantôme» des pixels d'avant-plan. 38
Figure 3.5 Points caractéristiques décrits avec FREAK en utilisant un voisinage de 22 pixels et 3 octaves. La zone hachurée verte est la zone sans points caractéristiques. 39
Figure 3.6 Correspondances de points trouvés entre l'image précédente et l'image courante (affichées sur l'image courante par un trait vert). **a)** Correspondances trouvées sans le test du ratio et le test de symétrie (sans seuil). **b)** Correspondances trouvées avec le test du ratio et le test de symétrie. 41
Figure 3.7 Résultat de la superposition de blobs concaves entre le temps t-1 et t. La partie grisée dans l'image à droite représente l'emplacement de A et B à t-1. On observe qu'il est possible de faire une mauvaise association avec ce type de méthode puisque les deux objets à t recouvrent chacun des objets à t-1. 41
Figure 3.8 Machine à état d'un objet 42
Figure 3.9 Exemple de sous-segmentation d'objets 44

Figure 3.10 Processus d'estimation des blobs : a) Position précédente des boîtes englobantes avec les points b) Blob sous-segmenté avec les points c) Boîte englobante estimée 45
Figure 3.11 Exemple de sursegmentation : a) Un piéton est séparé en 3 blobs distincts à cause d'une mauvaise soustraction d'arrière-plan. b) Un piéton entre dans la scène et son corps est divisé en trois blobs distincts. c) Un poteau cache la route et divise les objets en deux lors de leur passage. 46
Figure 3.12 Évolution des blobs de la sortie et l'entrée simultanée d'objets 47
Figure 3.13 Sortie des algorithmes de suppression des ombres : a) Fragment de l'image réelle étudiée. b) Résultat de la soustraction d'arrière-plan c) Méthode de chromacité d) Méthode géométrique e) Méthode physique f) Méthode à base de texture 49
Figure 3.14 Erreur de la méthode chromatique : a) Véhicule suivi b) La partie frontale du véhicule a été entièrement détectée comme une ombre c) La partie centrale du véhicule a été détectée comme une ombre alors le véhicule a été séparé en deux parties 49
Figure 4.1 Interface de l'outil de génération de vérité terrain 54
Figure 4.2 Une trame de chaque ensemble de données utilisées : a) Sherbrooke b) Rouen c) St-Marc d) René-Lévesque e) Atrium 55
Figure 4.3 Objet de référence pour chaque scène : a) Sherbrooke b) Rouen c) St-Marc d) René-Lévesque e) Atrium 56
Figure 4.4 Masque appliqué à la vidéo Sherbrooke 57
Figure 4.5 Problème de segmentation des objets (pour Traffic Intelligence (TI) et même scène gérée par Urban Tracker (UT) : a) et b) TI c) et d) UT . . 58
Figure 4.6 Piétons groupés pour l'ensemble de la séquence par UT 59
Figure 4.7 Exemple de résultats de UT pour la séquence Sherbrooke. 60
Figure 4.8 Problèmes de piétons groupés par UT 61
Figure 4.9 Exemple de résultats de UT pour la séquence Rouen 61
Figure 4.10 Problèmes de piétons groupés par UT 63
Figure 4.11 Exemple de résultats de UT pour la séquence St-Marc. 63
Figure 4.12 Masque appliqué à la scène René-Lévesque 64
Figure 4.13 Problèmes de véhicule lointain groupé par UT. On peut voir en c) que le groupe (visible en a) et b)) se brise lorsque le véhicule s'éloigne suffisamment. 65
Figure 4.14 Exemple de résultats de UT pour la séquence René-Lévesque 65

Figure 4.15 La partie sous l'escalier de l'atrium est très réflective. Cela déforme une partie des blobs pour UT. 66
Figure 4.16 Exemple de résultats obtenu à l'aide d'UT pour la vidéo Atrium 66
Figure 4.17 Impact du seuil d'association sur le MOTA et le MOTP de chaque ensemble de données. Pour la colonne de gauche, une valeur plus élevée représente un meilleur suivi alors que pour la colonne de droite, une valeur plus basse représente un suivi plus précis. 68
Figure B.1 Interface de l'annotateur 82
Figure B.2 Option d'importation 83
Figure B.3 Vidéo de résultat 84
Figure B.4 Panneau inférieur 85
Figure B.5 Panneau de contrôle de l'interpolation 85
Figure B.6 Panneau de contrôle vidéo 85
Figure B.7 Menu des options 86

LISTE DES ANNEXES

Annexe A Format Polytrack 79
Annexe B Manuel d'utilisation de l'annotateur 82
Annexe C Manuel d'utilisation de l'outil d'évaluation des métriques 87

LISTE DES SIGLES ET ABRÉVIATIONS

blob	Région d'une image formé par un ensemble de pixels connectés spatialement
OpenCV	Open Source Computer Vision. Il s'agit de la librairie la plus complète en vision par ordinateur à l'heure actuelle
trame	Une image d'une séquence vidéo
FREAK	Fast Retina Keypoint
SIFT	Scale-invariant feature transform
OpenCV	Open Source Computer Vision
AGAST	Adaptive and Generic corner detection based on the Accelerated Segment Test
RANSAC	RANdom SAmple Consensensus
FAST	Features from Accelerated Segment Test
BRISK	Binary Robust Invariant Scalable Keypoints
KLT	Kanade–Lucas–Tomasi
HoG	Histogrammes des gradients
SVM	Machine à support de vecteurs
DoG	Différences de gaussiennes
ViBe	Visual Background extractor
LCSS	Longuest Common Sub-Sequence
ICA	Independent Component Analysis
TI	Traffic Intelligence
UT	Urban Tracker

CHAPITRE 1

INTRODUCTION

Les accidents de la route ont été responsables de plus de 800 décès et de 90000 blessés en milieu urbain au Canada en 2010 selon le Gouvernement du Canada (2012). Ces accidents sont très coûteux pour la société autant au niveau humain qu'économique. La cohabitation difficile entre les différents usagers de la route est source de beaucoup de frustration en milieux urbains et bien que les usagers de la route soient directement responsables d'une partie de ces interactions dangereuses, une autre partie peut être attribuable à l'environnement dans lequel ceux-ci évoluent. L'étude des accidents et des conditions menant à des situations dangereuses permet d'améliorer la sécurité des infrastructures routières pour tous les usagers puisque les ingénieurs en transport peuvent alors changer la géométrie et la signalisation afin de réduire le risque d'accident. Malheureusement, très peu de données sont accessibles sur le sujet, puisque les sources de données d'accidents sont les rapports de police et comme il n'y a pas de rapport pour les interactions dangereuses sans accident, celles-ci ne sont pas recensées. La collecte de données est vitale à l'amélioration de la sécurité routière et pourrait permettre d'améliorer significativement la sécurité en milieu urbain pour les différents usagers de la route. L'acquisition de données vidéo est une méthode non intrusive de collecte d'information qui permettrait d'accumuler une grande quantité d'information sur les interactions entre les différents usagers de la route. Pour étudier les interactions entre les usagers de la route, il faut être en mesure de détecter les objets en mouvement et de les suivre pour la durée de leur passage dans le champ de vision de la caméra.

Le suivi d'objet est généralement une procédure simple pour un être humain. La connaissance d'une multitude de formes d'objets permet généralement à l'humain de deviner la position et la taille des objets mêmes lorsque ceux-ci sont partiellement cachés. Le problème se complexifie généralement lorsque la vitesse des objets est très élevée et lorsqu'il y a plusieurs d'objets en mouvement (par exemple pour suivre plusieurs individus dans une foule simultanément). L'analyse de longues séquences vidéo rend inévitablement l'humain moins alerte et il est possible que celui-ci ne remarque pas certaines informations importantes. De plus, il est difficile pour un humain de donner des données quantifiables précises comme la vitesse d'un véhicule. La grande quantité d'heures de vidéo à analyser pour voir des interactions dangereuses et des accidents rend cette tâche peu praticable pour une analyse par l'homme. Afin de réduire le nombre de séquences vidéo à visionner, le système Auto Incident Recording System (AIRS) a été conçu par Green *et al.* (2005). Ce système est constitué de deux caméras qui

enregistrent la circulation à une intersection. Le système utilise des microphones directionnels afin de détecter des bruits précurseurs d'accident tels que des bruits de klaxon ou de freins et des bruits indicateurs d'accidents comme du métal froissé. Lorsque ces sons sont détectés, les 4 secondes avant et après la détection du bruit sont enregistrées. L'analyse manuelle des vidéos résultante a permis de déterminer les problèmes les plus fréquents et d'améliorer l'intersection en allongeant un trottoir et en rajoutant de la signalisation pour réduire le nombre d'accidents de 5 types. Les auteurs ont ensuite analysé le nombre d'accidents pour les 3 années suivant les modifications effectuées. Ce faisant, ils ont réussi à faire passer la moyenne (sur 3 ans) du nombre d'accidents à l'intersection de 35 par année à 30 par année. Un certain nombre d'incidents n'ont pas été enregistrés par le système. Inversement, le système a décelé plusieurs accidents pour lesquels il n'y avait pas eu de rapport de police. L'inconvénient de ce système est qu'il a généré un très grand nombre de faux positifs ce qui a ralenti beaucoup l'analyse de vidéo par les humains. Il n'est également pas possible de quantifier le nombre d'interactions dangereuses qui n'ont pas été enregistrées par le système et par sa nature, ce système a beaucoup plus de chance de détecter les accidents impliquant plusieurs véhicules automobiles plutôt que ceux impliquant des piétons et des cyclistes. Toutefois, cette étude démontre tout de même qu'une meilleure connaissance des interactions entre les usagers de la route permet de proposer des améliorations qui réduisent le risque d'accident en milieu urbain.

1.1 Définitions et concepts de base

La vision par ordinateur permet l'analyse automatisée de séquences vidéo. Plus spécifiquement, le suivi d'objets est le champ d'application de la vision par ordinateur qui s'intéresse aux trajectoires empruntées par des objets en mouvement. Il s'agit d'une composante essentielle à beaucoup de systèmes utilisant la vidéo comme source d'information que ce soit pour assurer la surveillance d'un lieu, la sécurité des transports ou la supervision des opérations dans une usine. L'analyse par ordinateur présente des avantages importants par rapport à une analyse manuelle puisqu'elle permet l'analyse d'une grande quantité de données en plus de permettre le calcul de données quantitatives. Toutefois, des notions simples pour l'humain comme la détection, la description et le suivi d'objets deviennent très complexes une fois posées dans un contexte informatique. En effet, le suivi d'objets par ordinateur a pour information principale une grille de pixels (une image) de laquelle il faut extraire un ou plusieurs groupes de pixels d'intérêt représentant les objets et qu'il faut ensuite être capable de les suivre temporellement. La procédure générale peut donc être divisée en trois parties subséquentes :

1. La détection des objets.
2. La description des objets qui consiste à choisir une ou plusieurs propriétés des objets qui nous permettent de les identifier dans les trames subséquentes par exemple leur couleur, leur forme, leur texture ou leur position.
3. Le suivi des objets qui consiste à utiliser cette description afin de les retrouver dans les trames subséquentes.

1.2 Problématique

Le problème que nous désirons résoudre est de concevoir un système capable d'automatiser la collecte des trajectoires empruntées par les différents usagers de la route à des intersections urbaines à partir d'une de données vidéo. Il y a plusieurs défis dans la résolution de cette problématique. Premièrement, les caméras ne sont pas placées très hautes ce qui crée beaucoup d'occultation entre entre les véhicules. De plus, les véhicules peuvent s'arrêter sur de longues périodes (par exemple aux feux) ce qui peut les rendre difficiles à distinguer de l'arrière-plan si on utilise une méthode basée sur la soustraction d'arrière-plan. Les ombres peuvent également fortement déformer les objets ce qui peut avoir un impact important sur la position du centre de masse et occasionner la fusion de deux objets différents. Les mouvements sporadiques tels que le mouvement des feuilles d'arbres peuvent également être faussement détectés comme des objets d'intérêts. Notre problématique est différente de la problématique du suivi sur autoroute puisque le milieu urbain nous confronte à un bien plus grand nombre de types d'usager différents pouvant réaliser un ensemble de mouvements beaucoup plus complexes.

1.3 Objectifs

Notre objectif général consiste à développer un algorithme capable de suivre plusieurs objets différents et d'en enregistrer les trajectoires à partir d'une caméra vidéo. Pour ce faire, nous avons 3 objectifs spécifiques à atteindre :

1. Détecter les différents objets en mouvement dans la séquence vidéo et ignorer les objets non pertinents tels que les feuilles des arbres ou les ombres des oiseaux.
2. Extraire les trajectoires des objets en mouvement en calculant la correspondance entre les objets des différentes trames via une stratégie d'association de données et l'élaboration d'un modèle de suivi.
3. Valider la qualité des trajectoires extraites à l'aide de métriques standard pour le suivi multiobjet, et comparer nos résultats avec au moins un autre algorithme.

1.4 Aperçu de la méthode proposée

Nous utilisons des séquences vidéo urbaines ayant été filmées à l'aide d'une seule caméra statique. Nous effectuons ensuite un traitement sur chacune des trames de la vidéo.

La première étape de notre algorithme consiste à détecter les objets d'intérêts. Pour ce faire, nous commençons par réaliser une soustraction d'arrière-plan entre les trames. Cela nous permet de détecter les pixels dont l'intensité a changé et pour lesquels il y a probablement du mouvement. Nous utilisons ensuite une analyse en composantes connectées afin de déterminer les groupes de pixels qui sont connectés spatialement (les blobs). Nous filtrons ensuite les blobs qui sont dus à du bruit ou des changements de luminosité. Pour les blobs restants, un modèle de suivi est construit à l'aide des points caractéristiques situés à l'intérieur, de leur position spatiale, et de leur forme.

La seconde étape de l'algorithme consiste à associer les blobs temporellement d'une trame à la suivante. Pour ce faire, nous utilisons le modèle défini précédemment. D'abord nous calculons des paires de points caractéristiques correspondants d'une trame à la suivante. Ces paires de points sont ensuite filtrées afin d'éliminer les paires de points erronés. Les blobs ayant suffisamment de paires de points correspondants ou une superposition spatiale suffisante sont associés, ce qui nous permet de savoir s'il s'agit d'un lien direct, une fusion, une séparation ou encore un blob entrant ou quittant.

Finalement, la troisième étape consiste à réaliser le suivi des objets. Contrairement aux blobs, les objets sont suivis pour l'ensemble de leur passage dans la scène plutôt que d'une trame à l'autre. La première étape est donc d'associer les blobs aux objets et de gérer les différents types d'association. Ensuite, le modèle de suivi de chacun des objets sera mis à jour en fonction des associations de blob. Des mécanismes sont utilisés afin de gérer les problèmes d'occultation et de segmentation. Finalement, afin de gérer le cycle de vie des objets et gérer certains problèmes de segmentation, nous utilisons une machine à état.

1.5 Contributions

Notre contribution principale consiste à avoir créé un nouvel algorithme de suivi multiobjet conçu spécifiquement pour le milieu urbain. Celui-ci est capable de gérer des objets indépendamment de leur type ou de leur taille. Son utilisation ne nécessite aucune connaissance a priori de la scène (calibration de la caméra) et il ne nécessite que quelques paramètres intuitifs. Nous avons également proposé un algorithme pour estimer la position des boîtes englobantes en cas d'occultations partielles.

1.6 Plan du mémoire

Le second chapitre de ce mémoire fera une revue de littérature des systèmes de suivi existants et des diverses techniques employées dans notre projet. Le troisième chapitre traite de la méthodologie employée et des détails techniques de notre méthode. Le quatrième chapitre explique les détails de notre méthode d'évaluation ainsi que les résultats obtenus sur plusieurs vidéos. Le dernier chapitre conclut le mémoire et traite des travaux futurs.

CHAPITRE 2

REVUE DE LITTÉRATURE

Ce chapitre est divisé en trois parties. D'abord, les méthodes générales de suivi sont présentées. Plus spécifiquement, il sera question du suivi par blob, du suivi par point, du flux optique et du suivi par détection. Ensuite, il sera question des méthodes de suivi plus spécifiques à notre domaine d'application, le suivi des usagers du réseau routier. Pour cette section, le problème de suivi de véhicules sur autoroute sera d'abord abordé et ensuite les méthodes de suivi de circulation mixte en milieux urbains seront présentées. Finalement, les méthodes de gestion de la perspective seront abordées.

2.1 Suivi par blob

2.1.1 Explication de la méthode

Un blob est une région d'une image formée par un ensemble de pixels connectés spatialement. Les méthodes de suivi par blob consistent donc à trouver un blob d'intérêt dans une trame et de le retrouver ensuite dans les trames subséquentes. Puisque les objets d'intérêt sont habituellement les objets en mouvement dans la scène, la méthode consiste habituellement à appliquer une méthode de soustraction d'arrière-plan (tel que défini à la section 2.1.2) afin d'obtenir les pixels des objets en mouvement de la scène (voir par exemple la figure 2.1). Ensuite, afin de pouvoir trouver les blobs des objets désirés, on évalue les interconnexions spatiales entre les pixels en mouvement à l'aide d'une méthode d'analyse en composante connectée telle que celle définie par Chang *et al.* (2004). Il faut ensuite associer ces blobs de trame en trame à l'aide d'une propriété tels les histogrammes de couleur ou la proximité spatiale.

2.1.2 Soustraction d'arrière-plan

La soustraction d'arrière-plan est une étape importante dans beaucoup d'algorithmes de vision par ordinateur afin d'identifier les objets en mouvement. Cette méthode peut être utilisée lorsque la caméra est statique par rapport à une scène dans laquelle les objets d'intérêt sont en mouvement.

La méthode intuitive de soustraction d'arrière-plan nécessite une image courante et une image de référence sans les objets d'intérêt (l'arrière-plan). La méthode intuitive consiste



Figure 2.1 Exemple de soustraction d'arrière-plan : a) Image originale b) Image obtenue après la soustraction d'arrière-plan

ensuite à effectuer une soustraction entre les valeurs des pixels de l'image courante et de l'image de référence et d'appliquer un seuillage à l'image résultante. On obtient ainsi les pixels différents dans l'image ce qui représente les objets en mouvement. Un exemple du résultat d'une telle méthode est visible dans la figure 2.2. On peut toutefois remarquer plusieurs problèmes dus à cette méthode. Les objets ne sont pas correctement segmentés et plusieurs parties sont manquantes. Cela s'explique entre autres par une trop grande similarité entre la couleur des objets et celle de l'arrière-plan. De plus, on peut observer beaucoup de bruit dans l'image avec un seuil faible puisqu'il y a de légers mouvements de feuilles dans les arbres et le capteur de la caméra produit un certain niveau de bruit. De nombreux autres problèmes existent avec cette méthode. Cette méthode ne gère pas les ombres projetées des objets ni les changements de luminosité dans la scène. Si un objet arrête son déplacement, il sera toujours considéré comme étant en mouvement puisque l'objet ne fait pas partie de l'image de référence. De plus, une image de référence n'est pas toujours aisée à obtenir, car dans certains cas il y a toujours du mouvement.

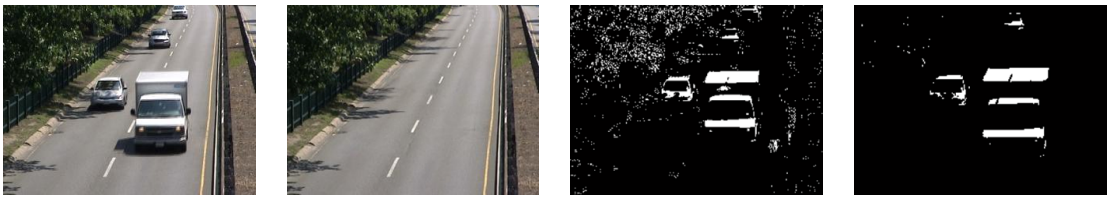


Figure 2.2 a) Image originale b) Image de référence c) Image après application de la méthode intuitive avec un seuil faible d) Image après application de la méthode intuitive avec un seuil élevé

De nombreuses méthodes de soustraction d'arrière-plan ont été conçues afin de résoudre un ou plusieurs de ces problèmes. Une partie de ces problèmes peuvent être résolus en créant un modèle de référence dynamique capable de se mettre à jour périodiquement. Ce type de

modèle permet de prendre en compte les changements de luminosité et les nouveaux objets s'arrêtant pour de longue période dans la séquence vidéo.

Une des méthodes les plus utilisées intégrant un modèle de référence dynamique est la soustraction d'arrière-plan à l'aide de la méthode mélange de distributions gaussiennes de Stauffer et Grimson (1999). Cette méthode consiste à maintenir à jour un modèle d'arrière-plan en modélisant chacun des pixels par un mélange de distributions gaussiennes. L'intérêt de cette méthode est que celle-ci permet de gérer les zones avec de petits mouvements comme les feuilles ou l'eau en prenant en compte la variance de la zone d'arrière-plan. L'utilisation d'un mélange de distribution de pixel permet de gérer plusieurs modèles d'arrière-plan ainsi que le bruit du capteur. L'utilisation de plusieurs modèles d'arrière-plan peut être utile dans une situation où une zone de l'image alterne fréquemment entre deux états par exemple si une branche d'arbre se déplace sous l'effet du vent devant un bâtiment. L'utilisation de multiples gaussienne permet ici de garder un modèle différent pour la branche et pour le bâtiment ce qui nous évite d'avoir un modèle changeant à chaque fois que les pixels de la branche et du bâtiment alternent. Les changements dus à la luminosité et aux nouveaux objets statiques sont également gérés, car ceux-ci sont intégrés graduellement au modèle. Le modèle étant mis à jour à l'aide des images, il n'est plus nécessaire de fournir une image de référence initiale. Toutefois, les objets dynamiques présents dans la première trame pourraient créer des objets fantômes pour un certain nombre de trames d'apprentissage, le temps que ceux-ci soient entièrement remplacés par des pixels de l'arrière-plan réel dans le modèle. D'autres problèmes subsistent dans cette méthode comme la mauvaise segmentation de l'avant-plan due à des couleurs similaires et les problèmes de distorsion des blobs de l'avant-plan due aux ombres et la méthode optimale dépend directement du problème à résoudre.

ViBe de Barnich et Van Droogenbroeck (2011) est une méthode très rapide basée sur l'échantillonnage de pixels aléatoire. La méthode travaille au niveau des pixels et utilise le modèle des pixels du voisinage du pixel (échantillonnés aléatoirement) pour décider si le pixel courant fait partie de l'avant-plan ou de l'arrière-plan. La méthode ne nécessite aucun changement de paramètres et est capable de gérer les changements d'arrière-plan et les objets dynamiques.

Goyette *et al.* (2012) propose un banc de test public contenant un ensemble de métriques appliqué à de nombreuses méthodes pour divers ensembles d'images contenant des ombres, des arrière-plans dynamiques, de la vibration au niveau de la caméra, des images thermiques, etc. Plusieurs méthodes capables de réduire l'impact des problèmes mentionnés plus haut sont évaluées sur ce banc de test comme ViBe de Barnich et Van Droogenbroeck (2011) et KDE de Nonaka *et al.* (2012). Un autre banc de test similaire a été réalisé par Antoine Vacavant et Lequièvre (2012), toutefois un nombre moins grand de méthodes y est évalué.

2.1.3 Formation des blobs

Une fois la soustraction d'arrière-plan réalisé, il faut associer les pixels d'avant-plan détecté sous forme de blobs en analysant les pixels connexes. Pour ce faire, il faut parcourir l'image et assigner une étiquette identique à tous les pixels connexes d'un même groupe. Une technique rapide d'analyse des composantes connexes est proposée par Chang *et al.* (2004). Elle consiste à d'abord détecter les contours en parcourant l'image. Ensuite, chacune des lignes à l'intérieur des contours est parcourue afin de trouver des contours internes à l'objet. Une fois les contours annotés, on étiquette ensuite l'ensemble des pixels situé à l'intérieur des contours.

2.1.4 Méthodes d'association des observations et modèle de suivi

Une fois les blobs d'intérêt extraits de la vidéo, il faut faire la correspondance entre ceux-ci de trame en trame et détecter lorsque des objets entrent dans la scène, sortent de la scène, entrent en état d'occultation ou se divisent. Pour faire la correspondance, il est possible d'utiliser la proximité spatiale entre les blobs de deux trames successives. Fuentes et Velastin (2001) proposent un système utilisant la superposition entre les blobs à l'instant précédent B_{t-1} et ceux de l'instant présent B_t. Cette technique permet de détecter les différents cas d'association, c'est-à-dire l'apparition des blobs (0 vers 1), la sortie des blobs (1 vers 0), la correspondance (1 vers 1), la fusion (m vers n avec m $>$ n), la division (m vers n avec m $<$ n) et tout mélange de fusion et de division. Ce système conserve l'information sur les groupes initiaux lorsque deux objets se rencontrent afin de restaurer l'identité des objets. Cette technique a l'avantage de fonctionner sur les images en noir et blanc puisqu'elle n'est pas basée sur les couleurs. De plus, elle est rapide et elle permet de gérer les scènes où il y a des fusions et des divisions entre les objets. Il faut toutefois que la vitesse des objets soit suffisamment lente par rapport à la vitesse de capture de la caméra pour qu'il y ait toujours superposition entre les blobs de deux trames consécutives. La position individuelle des objets n'est pas connue durant les occlusions. Haritaoglu *et al.* (1998) utilisent un système similaire de suivi par superposition, toutefois, au lieu de faire la correspondance entre B_{t-1} et B_t, ils font la correspondance entre le B_t et ${B^p}_t$, c'est-à-dire le blob prédit par le modèle de mouvement généré à l'aide des observations précédentes. Cela a pour effet que l'algorithme peut gérer certains cas où les objets vont plus rapidement par rapport à la fréquence de capture de la caméra.

Lorsque l'on a accès à une vidéo en couleur, une autre façon de faire le lien entre les blobs à travers le temps est d'utiliser des histogrammes de couleur. Pour ce faire, il faut calculer l'histogramme des couleurs de chacun des blobs et le comparer avec les histogrammes des blobs de la trame suivante. Ces histogrammes peuvent être calculés dans divers espaces de

couleurs tels que RGB ou HSV. Pour comparer les histogrammes de trame en trame, il faut ensuite utiliser une mesure de distance. Une bonne mesure de comparaison d'histogramme est la distance de Bhattacharyya. Celle-ci est d'ailleurs utilisée par Comaniciu *et al.* (2000) afin de trouver la position optimale d'une fenêtre dans l'algorithme Mean-Shift.

Une autre catégorie de méthode d'association des blobs sont les méthodes par graphe. Ces méthodes permettent de représenter spatialement et temporellement les données en créant un graphe des associations entre les blobs à travers le temps. Chia *et al.* (2006) modélise le graphe sur l'ensemble de la séquence ce qui permet d'utiliser les informations de blob futures afin de déterminer la séquence d'association optimale. Cette approche a pour limitation que le traitement du graphe doit se faire hors-ligne une fois que la vidéo complète a été acquise. De plus, la quantité de mémoire nécessaire peut devenir problématique si la vidéo est très longue. Torabi et Bilodeau (2009) ont plutôt proposé une approche en ligne et le graphe est mis à jour à chaque trame. Dans cette approche, un graphe contenant les évènements de type entré, sortie, fusion, division et correspondance sont créées avec dans chaque nœud les informations d'apparence du blob telles que son histogramme de couleur, sa taille et sa position. Un second graphe d'hypothèses est quant à lui utilisé afin de calculer les associations. Pour ce faire, la distance entre les histogrammes est utilisée comme poids entre les nœuds.

2.2 Suivi par points

2.2.1 Explication de la méthode

Le suivi par points consiste à suivre des caractéristiques locales d'une image de trame en trame. Les étapes habituelles consistent en la détection des points, leur description, la mise en correspondance des points d'une trame à l'autre et le regroupement des points en objets de plus haut niveau. Les caractéristiques des détecteurs et des descripteurs tels que leur niveau d'invariance aux rotations et aux changements d'échelle sont examinées dans la revue de littérature de Li et Allinson (2008). Le besoin de faire du traitement temps-réel sur mobile pour des applications comme la réalité augmentée a nécessité la création de méthodes moins exigeantes en termes de ressources. Ce besoin a amené la création de nombreux descripteurs binaires dans les dernières années. Ceux-ci sont comparés par Heinly *et al.* (2012) aux méthodes traditionnelles en termes de temps de traitement et résistance aux différents types de transformations. Dans les sections suivantes, des méthodes de détection, description et regroupement des points seront expliquées plus en détail.

2.2.2 Méthodes de détections de points caractéristiques

La détection des points consiste à trouver des points qui sont uniques dans l'image et qui peuvent être retrouvés dans une autre image avec le moins d'ambiguïté possible. Les points à éviter sont ceux situés sur des surfaces continues et sur les bordures, car ceux-ci ne sont pas uniques. Les coins leur sont préférés, car ils sont très caractéristiques. On définit un coin comme étant un point de l'image où l'intensité change fortement dans les deux dimensions. Les méthodes de détection de points de Moravec, Harris, SIFT, FAST, AGAST et BRISK seront expliquées ci-dessous.

Moravec

L'opérateur de Moravec (1980) consiste à considérer une petite fenêtre autour d'un point ainsi qu'une translation de cette fenêtre dans les 8 directions et de calculer l'équation 2.1 dans laquelle I(u,v) représente l'intensité d'un pixel à une position (u,v) dans l'image. Cette valeur représente la moyenne du changement d'intensité lorsque la fenêtre est déplacée de x,y. Cette valeur est faible lorsque les intensités sont approximativement constantes. Sur les contours, la fonction prendra de fortes valeurs pour les déplacements perpendiculaires au contour et de faibles valeurs pour ceux sur le contour. Pour un coin, la valeur sera très élevée dans toutes les directions. Un seuillage est ensuite appliqué pour obtenir les meilleurs points. Finalement, les valeurs qui ne sont pas des maximums locaux sont supprimées. Cette méthode permet donc trouver des coins, mais elle réagit également fortement aux diagonales ce qui est problématique, car les points sur les bordures ne sont pas caractéristiques dues à leur similarité aux autres points situés le long de la bordure.

$$E(x,y) = \sum_{u,v} |I(x+u, y+u) - I(u,v)|^2 \tag{2.1}$$

Harris

Le détecteur de point de Harris et Stephens (1988) est dérivé de l'opérateur Moravec. Ce détecteur tente de résoudre les problèmes majeurs de cet opérateur qui sont que l'opérateur réagit trop aux bordures, les réponses sont bruitées et la réponse de l'opérateur est anisotropique (dépendante de l'orientation) puisque seulement 8 directions sont vérifiées. Afin que l'opérateur soit moins affecté par le bruit, une fenêtre circulaire gaussienne est utilisée plutôt qu'une fenêtre binaire rectangulaire. Pour résoudre les problèmes d'anisotropisme, une nouvelle équation a été dérivée afin de couvrir toutes les variations. Finalement, pour réduire l'impact des bordures, on vérifie que l'intensité varie fortement dans toutes les directions et

non une seule. La méthode de Harris est significativement plus coûteuse en ressources que la méthode de Moravec, toutefois les points obtenus sont beaucoup plus caractéristiques.

SIFT

L'algorithme de Scale-invariant feature transform (SIFT) a été originalement publié par David G. Lowe en 1999 et détaillé d'avantage dans Lowe (2004). La méthode SIFT permet la détection et la description des points. Les points générés par cette méthode ont la caractéristique d'être invariants à la mise à l'échelle, la rotation et la translation en plus de présenter une certaine invariance aux changements d'illumination et aux transformations affines. La contribution majeure de cet article est principalement pour l'invariance au changement d'échelle qui n'était pas possible précédemment avec les points Harris. Pour détecter les points qui seront facilement localisables à de multiples échelles, la technique SIFT utilise la technique des différences de gaussiennes. Pour ce faire, un filtre flou gaussien est appliqué à l'image originale. On réapplique ensuite le filtre à l'image floue de multiple fois de façon à avoir 5 images floues. On effectue ensuite la soustraction entre les différents niveaux afin d'obtenir la différences de gaussiennes (DoG) (voir figure 2.3). Puisqu'il faut avoir cette information à plusieurs échelles, les images floues sont redimensionnées en divisant la taille de ceux-ci par deux et on calcule les DoG de plusieurs niveaux. Ces DoG à plusieurs échelles sont nommés octave. Afin de trouver les coins, il faut ensuite considérer l'entourage du point aux échelles adjacentes dans la différence de gaussienne et prendre les points qui représentent des minima ou maxima locaux. Pour éliminer les points situés sur les bordures, il faut calculer deux gradients perpendiculaires. Si les deux gradients sont grands, on a un coin. Si un seul des gradients est grand, nous avons une bordure. Finalement, si les deux gradients sont petits, nous sommes sur une surface plane. Nous ne gardons donc que les points qui ont deux grands gradients.

FAST

L'algorithme Features from Accelerated Segment Test (FAST) présenté par Rosten et Drummond (2006) vise à utiliser l'apprentissage automatique (*machine learning*) pour accélérer la détection de coins. Pour vérifier si un point p est un coin, l'algorithme utilise un cercle de rayon 3 autour du point p considéré (voir figure 2.4). S'il existe sur c un arc continu contenant plus de N pixels qui soient plus pâles ou foncés que p d'une différence t, alors il y a un coin au point p. Le nombre de points N utilisé dépend de la version de FAST. La meilleure version selon l'auteur est la version avec $N = 9$ (FAST-9) puisqu'il s'agit de la version offrant la meilleure répétabilité, c'est-à-dire celle qui aura le plus tendance à trouver

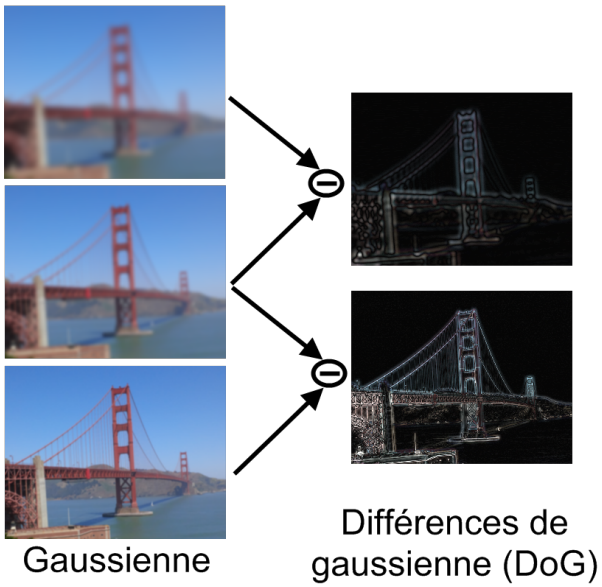


Figure 2.3 Calcul des différences de gaussiennes

les mêmes points dans 2 points de vue différents selon les tests effectués. Une fois les coins candidats trouvés, une seconde étape est exécutée afin d'éliminer les points adjacents qui ne sont pas des maximums locaux dans la zone délimitée par c. La figure 2.4 montre la zone considérée par l'algorithme.

Le point fort de cet algorithme est sa rapidité d'exécution par rapport à la différence de gaussiennes utilisé par SIFT et Harris. Pour obtenir une grande rapidité, l'algorithme utilise une série de tests afin de limiter au maximum le nombre vérifications nécessaires permettant d'éliminer des points candidats. Pour ce faire, un arbre de décision est généré par l'algorithme ID3. L'algorithme ID3 utilise un ensemble de données d'entrainement afin d'apprendre la configuration des coins et de pouvoir trouver une séquence efficace de comparaison pour éliminer le plus rapidement possible les points qui ne sont pas des coins. Par exemple, pour un cas où $N = 12$, si on évalue les points 1,9,5 et 13 alors il faut qu'au moins 3 de ces points répondent au critère pour qu'il soit possible d'avoir une séquence continue de longueur N. Dans Rosten et Drummond (2005), l'auteur a évalué qu'avec FAST-12, une moyenne de 3.8 px devait être évalué pour savoir s'il s'agissait d'un coin plutôt que les 16 px requis par la méthode naïve, ce qui représente une accélération considérable.

AGAST

L'algorithme Adaptive and Generic corner detection based on the Accelerated Segment Test (AGAST) de Mair *et al.* (2010) est une méthode optimisée pour générer des arbres de décision plus efficaces que la méthode utilisée par FAST. Les auteurs tentent ici de combler

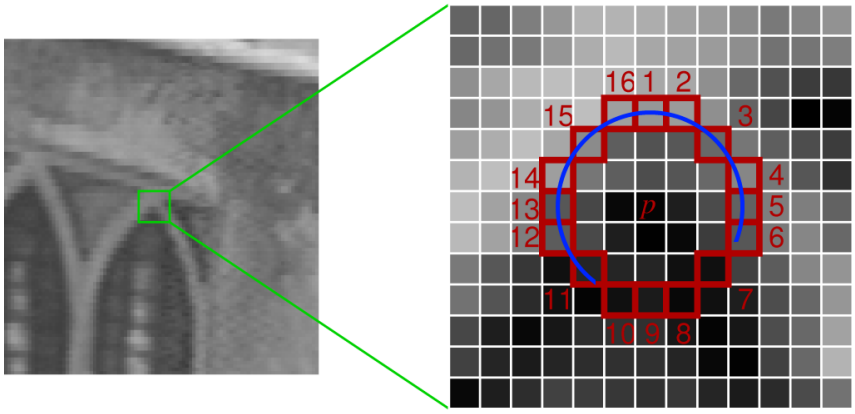


Figure 2.4 Voisinage d'un point considéré par FAST. En bleu, on a une ligne représentant les pixels plus pâles que le pixel central. Image tirée de Rosten et Drummond (2006). Reproduit avec permission.

certaines lacunes de FAST telles que la nécessité d'apprentissage pour obtenir de meilleurs résultats et le fait que les déplacements dans la scène tels que les rotations changent significativement la configuration des coins ce qui a pour effet de ralentir l'algorithme. Les auteurs proposent une méthode générique à deux arbres permettant d'obtenir des résultats plus rapidement et ne nécessitant pas d'apprentissage.

BRISK

La méthode Binary Robust Invariant Scalable Keypoints (BRISK) de Leutenegger *et al.* (2011) est une série d'algorithmes permettant de faire la détection et la description de points caractéristiques. L'algorithme de détection de BRISK est une généralisation à multiples échelles de l'algorithme AGAST vu précédemment. Pour ce faire, une pyramide à multiples échelles est construite. Cette pyramide est construite en échantillonnant l'image de façon à avoir n octaves, c'est-à-dire n images ayant été sous-échantillonnées par un facteur 2. De plus, des images intra octave sont créées entre les octaves de la pyramide. Pour chaque octave et intra octave, le détecteur FAST est utilisé. Pour chaque point avec un score suffisant, on prend le score obtenu à l'échelle inférieure et supérieure. On trace ensuite une parabole en prenant le score de chaque point et son échelle. On choisit ensuite le point de la parabole présentant le meilleur score afin de connaître son échelle exacte. Cette opération est représentée à la figure 2.5.

2.2.3 Méthodes existantes de descriptions de points et calcul de correspondance

La description des points est l'étape qui permet de trouver une signature de point qu'il sera possible de retrouver dans une autre image. Étant donné que les objets se déplacent, il

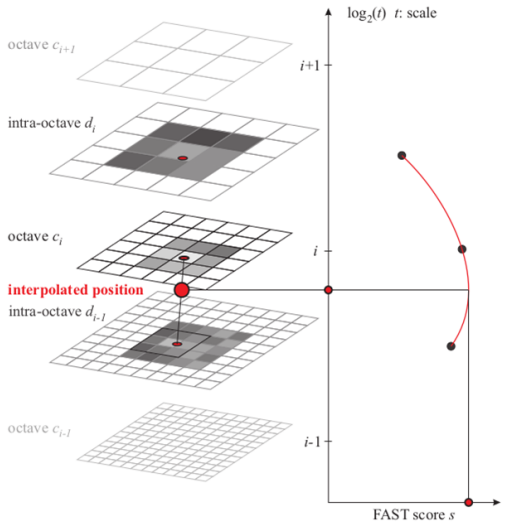


Figure 2.5 Multiple octave d'un point avec interpolation des scores FAST. Figure tirée de Leutenegger *et al.* (2011). © 2011 IEEE, reproduit avec permission.

est important que les descripteurs soient invariants à certaines transformations telles que la translation, la rotation et le changement d'échelle. L'invariance vient toutefois avec une perte de pouvoir discriminant. Il est donc important de bien choisir un descripteur en fonction des caractéristiques du problème à résoudre. Ce descripteur pourra ensuite être comparé à l'aide d'une mesure de distance à un autre descripteur ce qui permettra d'établir si un point est le même d'une image à l'autre malgré un déplacement de l'objet. Les méthodes de description SIFT, BRISK et FREAK seront expliquées ci-dessous.

SIFT

Comme vue dans la section précédente, la méthode des points SIFT permet de trouver de bons points invariants aux changements d'échelle. Pour être en mesure de retrouver des points malgré des rotations, il faut avoir une forme d'invariance à la rotation. Pour ce faire, la méthode SIFT calcule les gradients autour du point et utilise le gradient avec la magnitude la plus élevée. On utilise ensuite l'orientation de ce gradient comme point de référence pour tous les autres ce qui permet au descripteur d'avoir la même description, peu importe la rotation du point.

Le descripteur SIFT est un vecteur de 128 éléments. Ce vecteur représente un tableau de 4x4 histogrammes avec 8 classes contenant les orientations des gradients autour du point (le plan d'échantillonnage est visible à la figure 2.6.a). La magnitude des gradients est multipliée par une fonction gaussienne afin de donner plus de poids au centre du plan d'échantillonnage et cette magnitude est additionnée dans la classe de l'histogramme correspondant à son orien-

tation. Afin de réduire l'impact de la luminosité sur la description, le vecteur est normalisé. En effet, la magnitude des gradients varie proportionnellement à la luminosité, la normalisation a donc pour effet de tout remettre à la même échelle. Toutefois, puisque l'angle avec lequel la lumière frappe une surface a un grand impact sur la magnitude des gradients, un seuillage avec une valeur de 0.2 est appliqué sur le vecteur normalisé et une renormalisation est faite. Cela permet de limiter l'influence des grands gradients et d'augmenter l'emphase sur l'orientation plutôt que la magnitude des gradients. La valeur de 0.2 a été trouvée par les auteurs expérimentalement.

Afin de trouver la correspondance entre deux points SIFT, il faut calculer la distance euclidienne entre leurs descripteurs. Si la distance est inférieure à une certaine valeur, alors il s'agit vraisemblablement du même point. D'autres techniques peuvent être appliquées selon le domaine d'application afin d'améliorer la qualité des paires de points trouvées.

BRISK

La méthode BRISK vue précédemment décrit également une méthode pour décrire les points d'intérêt trouvés. Il s'agit comme SIFT d'une méthode invariante aux rotations et aux changements d'échelle, toutefois celle-ci est beaucoup plus rapide que SIFT pour la génération du descripteur ainsi que sa comparaison avec d'autres descripteurs. Le descripteur BRISK est un descripteur binaire de 512 bits calculé à l'aide d'une gaussienne pondérée sur un motif d'échantillonnage autour du point(voir figure 2.6.b). Afin d'avoir un descripteur invariant aux rotations, une direction caractéristique du descripteur est calculée à l'aide de comparaisons de valeurs d'intensité entre paires de points et les valeurs du descripteur sont ensuite calculées en fonction de ce référentiel. Ces valeurs sont le résultat de comparaisons d'intensité effectuées entre les paires de points échantillonnés. Pour comparer deux descripteurs BRISK, il faut calculer la distance de Hamming entre ceux-ci. Le nombre de bits identiques représente le degré de similarité entre les points. Cette opération est très rapide à effectuer sur les processeurs modernes x86, car elle peut être effectuée à l'aide d'un simple XOR entre les descripteurs suivis de l'appel à la fonction POPCNT (disponible avec le jeu d'instruction SSE4.2 sur processeur x86) qui permet de calculer rapidement le nombre de bits à 1 dans un nombre.

FREAK

Fast Retina Keypoint (FREAK) est une méthode de description de points caractéristiques par Alahi *et al.* (2012). Cette méthode est inspirée de la rétine humaine pour son plan d'échantillonnage. Ce plan est similaire à BRISK (voir 2.6.c), mais il y a une plus grande

densité de points vers le centre comme dans la rétine humaine. Le descripteur est composé de 64 octets qui représentent le résultat de différences de gaussiennes binaires. Pour que le descripteur soit invariant à la rotation, la même stratégie que BRISK est utilisée, toutefois un nombre moindre de paires de points est évalué (un algorithme d'optimisation a été utilisé afin de sélectionner les 512 paires de points les plus pertinentes). Le descripteur est conçu de façon à évaluer les détails grossiers d'abord (les cercles extérieurs) et ensuite les détails plus fins (approche *coarse-to-fine*) en utilisant une distance de Hamming. Cette approche permet d'éliminer 90% des points candidats après la comparaison des 16 premiers octets du descripteur, ce qui rend le calcul de comparaison extrêmement rapide.

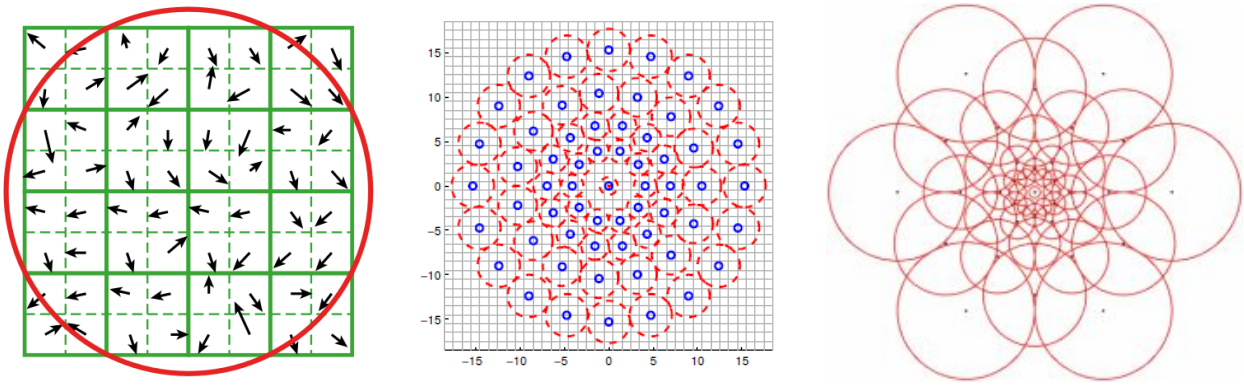


Figure 2.6 Plan d'échantillonnage du descripteur. a) SIFT b) BRISK c) FREAK. Pour b et c, les grands cercles représentent la gaussienne autour des échantillons (montrés par un point). a) Figure tirée de Lowe (2004) (figure 7, page 15), © 2004 Springer Science and Business Media, reproduite avec permission. b) Figure tirée de Leutenegger *et al.* (2011), © 2011 IEEE, reproduite avec permission. c) Figure tirée de Alahi *et al.* (2012), © 2012 IEEE, reproduite avec permission.

Méthode pour raffiner la qualité des correspondances

Le calcul des correspondances n'étant pas parfait, un nombre significatif de mauvaises correspondances seront calculées. Lowe (2004) décrit une série de tests à effectuer afin de filtrer un grand nombre de correspondances dans le domaine de la reconnaissance d'objets. Ces tests peuvent être également utilisés pour filtrer des mauvaises correspondances lors du suivi d'objets. Le premier des tests consiste à exiger un minimum de trois points caractéristiques correspondant à des points d'un même objet connu (appris préalablement à l'aide de points dans une scène annotée) puisque la plupart des mauvaises correspondances sont dues à des points d'arrière-plan ambigus et qu'il est plus probable qu'un ensemble de points soit correct que chaque point individuel. Le deuxième test à appliquer est le test du ratio. Ce test consiste à trouver la première et la deuxième meilleure correspondance (distance de descripteur la plus

faible) pour un point et de faire le ratio de leur résultat. Pour les bonnes correspondances, la deuxième meilleure correspondance a des fortes de chances d'être significativement moins bonne si les points sont vraiment caractéristiques alors que pour de mauvais points, celle-ci risque d'être beaucoup plus élevée. La deuxième meilleure correspondance sert de méthode d'estimation de la densité des mauvaises correspondances pour ce point. Pour le descripteur SIFT, l'auteur a calculé qu'un ratio de 0.8 permettait de filtrer 90% des mauvaises correspondances tout en éliminant seulement 5% des bonnes correspondances.

2.2.4 Méthode de regroupement des points

Une fois les points suivis de trame en trame, la majorité des applications de suivi nécessitent l'utilisation d'une technique afin de regrouper les points suivis en objet d'intérêt. Plusieurs informations peuvent être utilisées à cette fin comme la proximité spatiale et le mouvement commun. Beymer *et al.* (1997), proposent une mesure basée sur ces deux critères que nous appellerons mesure de similarité minimale maximale. Dans cette mesure, la proximité spatiale est vérifiée à l'aide d'un seuil de distance maximale entre deux points. Quant à lui, le mouvement commun des trajectoires p^a et p^b est validé par les équations (2.2) à chaque temps t commun des trajectoires et (2.3) avec s_{seg} un seuil de segmentation.

$$\Delta_p(t) = \sqrt{({p^a}_x(t) - {p^b}_x(t))^2 + ({p^a}_y(t) - {p^b}_y(t))^2} \tag{2.2}$$

$$max(\Delta_p) - min(\Delta_p) < s_{seg} \tag{2.3}$$

Antonini et Thiran (2006) proposent deux étapes de prétraitement des trajectoires d'une scène afin de pouvoir les regrouper. La première étape consiste à grouper les trajectoires de longueur similaire ensemble afin d'exploiter l'information temporelle. La deuxième étape consiste ensuite à grouper les trajectoires ayant des points de départ situés à proximité afin d'utiliser l'information spatiale.

2.3 Suivi par flux optique

Le flux optique est le mouvement perçu d'un objet causé par son mouvement relatif à l'observateur et la scène. Le flux optique peut être utilisé pour segmenter une image en fonction du mouvement des objets, stabiliser une vidéo, effectuer de la compression vidéo, etc. Contrairement à la soustraction d'arrière-plan, il est possible d'utiliser le flux optique sur des caméras en mouvement puisque le flux optique évalue le mouvement de toute la scène par rapport à la trame précédente et non par rapport à une image modèle statique. Il est donc possible d'isoler le mouvement de la caméra et le mouvement des objets individuels afin de

faire un suivi des objets. Le résultat obtenu par le calcul du flux optique entre les pixels de deux trames est un champ vectoriel comme montré à la figure 2.7.

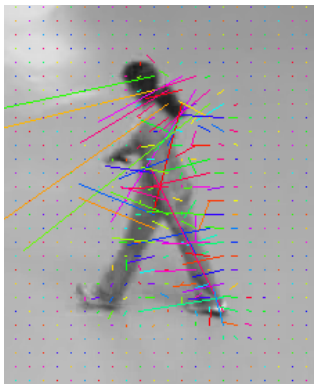


Figure 2.7 Exemple du flux optique

Une méthode populaire de calcul du flux optique est la méthode de Lucas et Kanade (1981). Cette méthode a été adaptée dans un algorithme de suivi par Shi et Tomasi (1994), la méthode Kanade–Lucas–Tomasi (KLT). Cette méthode utilise les valeurs propres minimales de matrices de gradients 2x2 afin d'obtenir de bons points caractéristiques. Ceux-ci sont ensuite suivis à l'aide de la méthode de Newton-Raphson qui consiste à minimiser les différences entre deux fenêtres. Lorsque suffisamment de points sont perdus par l'algorithme, une nouvelle recherche est faite pour trouver de nouveaux bons points caractéristiques et ceux-ci sont ajoutés à la liste des points à suivre.

2.4 Suivi par détection

Le suivi par détection consiste à utiliser un détecteur d'objets afin de localiser les objets d'intérêt dans une trame et d'ensuite lier temporellement ces détections de trame en trame. La méthode est similaire à celle du suivi par blob à l'exception qu'on remplace ici les blobs générés par soustraction d'arrière-plan par des détections. Les détecteurs d'objets sont beaucoup plus spécifiques que la soustraction d'arrière-plan. Alors que la soustraction d'arrière-plan se contente de trouver les pixels de l'image dont l'intensité a changé, les détecteurs vont chercher un objet avec une apparence particulière. Les détecteurs ont l'avantage de distinguer plusieurs objets mêmes si ceux-ci sont en occultation en plus d'ignorer les objets en mouvement inintéressants. Dans le contexte du suivi multiobjet, les inconvénients des détecteurs sont qu'il est difficile de générer un bon détecteur, c'est-à-dire un détecteur capable de détecter le plus grand nombre d'objets possible dans une scène sans générer trop de faux positifs. Pour qu'un détecteur fonctionne, il faut que les objets désirés aient tous des formes similaires. De plus, si l'objet a une forme différente selon le point de vue, alors il faut avoir

un détecteur pour chacun des points de vue considérés et être en mesure de choisir le point de vue approprié.

2.4.1 Méthode générale

Cascades de Haar

Une méthode classique de détection d'objet est la méthode des cascades de Haar de Viola et Jones (2001). Cette méthode originalement présentée pour la détection de visages a également été appliquée avec succès aux piétons et à plusieurs types d'objets. La méthode nécessite plusieurs milliers d'images positives et négatives desquelles les caractéristiques pseudo-Haar sont extraites afin de réaliser un apprentissage de la cascade des classifieurs du détecteur. À chaque étage de la cascade, un classifieur fort est entrainé à l'aide de la technique de *boosting* Adaboost qui sélectionne les caractéristiques pseudo-Haar et entraine des classifieurs faibles. La combinaison en cascade des classifieurs permet d'accélérer le traitement lors de la détection, car la plupart des données seront éliminées dès les premiers étages de la cascade. Cette accélération est importante puisqu'il y a un grand nombre de fenêtres à vérifier dans l'image. Une fois le détecteur entrainé, le processus de détection consiste à calculer l'image intégrale. L'image intégrale est une structure de données contenant en chacun de ses points la somme des pixels situés à gauche et au-dessus du point. Cela permet de calculer la somme des valeurs d'une zone rectangulaire très rapidement par la suite, car il suffit d'accéder à 4 valeurs pour connaître la somme plutôt que d'itérer au travers de l'image. Cette structure de donnée est ensuite utilisée pour accélérer le calcul des ondelettes pseudo-Haar dans toute l'image. On parcourt ensuite l'image à toutes les positions et les échelles désirées avec une fenêtre et on applique la cascade de décision afin de vérifier si la caractéristique répond positivement au classifieur. Si un des classifieurs rejette la caractéristique, alors la zone est rejetée et on décale la fenêtre avant de recommencer le processus. Une fois l'image complète parcourue, on combine les détections appartenant au même objet. Le grand défaut de cette méthode est le temps d'entrainement très long (qui peut dépasser une semaine) et la grande variété des paramètres d'entrainement qui sont optimisables uniquement que par essai-erreur. De plus, cette méthode ne peut pas directement être appliquée à un objet présentant des vues multiples différentes puisque les caractéristiques pseudo-Haar seront différentes.

Histogramme des gradients

Dalal et Triggs (2005) ont présenté une méthode efficace pour la détection de personnes basée sur les histogrammes des gradients (HoG). Cette méthode part du principe qu'on peut trouver l'objet en fonction de la distribution de ses gradients. Similairement à la mé-

thode de Viola et Jones (2001), cette méthode est également découpée en deux phases, une phase d'apprentissage et une phase de détection. Pour la phase d'apprentissage, un grand nombre d'échantillons d'images positives et négatives doivent être utilisés. Pour chacune de ces images, on calcule le descripteur HoG (un histogramme à 9 classes des orientations des gradients pondérés par leur intensité) et ces données sont apprises par une machine à support de vecteurs (SVM) linéaire. Pour la détection, on utilise ensuite une fenêtre glissante qu'on évalue à l'aide du SVM appris afin de classifier les données entre les classes positive et négative.

2.4.2 Application aux piétons

Une des applications des algorithmes de détection est pour la détection et le suivi des personnes. Le suivi de personnes dans une foule est très complexe avec des méthodes à base de soustraction d'arrière-plan à cause du grand nombre d'occultations entre les objets. De plus, le fait que les piétons soient des corps articulés rend plus difficile le groupement des points dans les méthodes à base de points caractéristiques. Puisque les détecteurs sont très efficaces pour détecter les objets de formes similaires et que ceux-ci sont fonctionnels même si plusieurs objets sont très rapprochés, leur application à la détection et au suivi de piétons est très prometteuse. En effet, la forme générale de chaque individu est similaire bien qu'il y est certaines variations physionomiques entre individus. Un piéton à la verticale a d'ailleurs une forme similaire pour tous les points de vue parallèles au plan (voir figure 2.8) ce qui évite d'avoir à créer un détecteur pour chaque orientation.



Figure 2.8 Exemple de point de vue supporté par un même détecteur de piéton. Modèle 3D tiré du jeu L.A. Noire, ©2011 Rockstar Games

Le développement rapide des méthodes de détection de piétons a amené Dollar *et al.* (2012) à faire une revue des méthodes courantes. On y constate que les méthodes courantes sont principalement dérivées de Dalal et Triggs (2005) et Viola et Jones (2001). Elles utilisent toutes des histogrammes de gradients ou des caractéristiques pseudo-Haar. Certaines

méthodes utilisent des versions légèrement modifiées de ces deux méthodes ou encore une combinaison des deux. Le classifieur est également basé sur un SVM ou sur une cascade de classifieurs en utilisant le *boosting*.

La détection peut également être appliquée avec succès aux véhicules. Toutefois, la forme du véhicule variant beaucoup avec les points de vue, celle-ci ne peut être utilisée avec succès que si les véhicules ont des formes similaires et qu'un seul point de vue est utilisé. Un bon exemple d'application est la détection de véhicules à partir d'un autre véhicule en mouvement. Ce type de détection peut permettre de s'assurer qu'un véhicule reste toujours à la même distance derrière un autre. De ce point de vue, la majorité des véhicules d'intérêt seront vus de derrière et ils auront des formes relativement similaires. La technique de Viola et Jones (2001) entrainée par Lee et Kanade (2007) pour la détection de véhicules a donné des résultats prometteurs pour ce type de détection.

2.4.3 Généralisation pour gérer des points de vue multiples

Les méthodes de détection ne peuvent pas facilement être utilisées si la forme des objets varie beaucoup en fonction des points de vue. Afin de résoudre ce problème, Viola *et al.* (2003) proposent d'utiliser plusieurs détecteurs combinés à un estimateur de pose afin de faire la détection de visage pour de multiples points de vue. Afin d'estimer la pose et connaitre le classifieur à utiliser, un arbre de décision a été entrainé avec un grand nombre d'images de chacune des classes. Le processus de détection se fait donc en deux étapes. D'abord on utilise l'estimateur de pose et ensuite on utilise le classifieur correspondant à la pose détectée pour la classification. Dans cet article, un total de 12 détecteurs ont été conçus (un pour chaque rotation de 30 degrés du visage).

Pour être en mesure d'appliquer la détection à des véhicules, il est essentiel d'utiliser de multiples détecteurs également puisque la forme varie fortement pour différents types de véhicule et pour différents points de vue. Ozuysal *et al.* (2009) utilisent un ensemble de données obtenues à l'aide de 20 voitures situées sur des plaques tournantes afin d'avoir des images annotées pour chacun des angles de véhicules. Afin de couvrir un grand nombre d'angles, une photo aux trois ou quatre degrés est prise ce qui fait un ensemble de données de 2000 images. La méthode proposée consiste à d'abord calculer le descripteur Daisy sur l'ensemble de l'image. Un classifieur bayésien est utilisé afin d'estimer la pose et de multiples SVMs sont ensuite entrainés avec ces descripteurs afin d'estimer les dimensions de la boîte englobante.

Vu le grand nombre de données requises pour la détection de véhicules, Kuo et Nevatia (2009) utilisent l'apprentissage non supervisé afin de déterminer les catégories de leur détecteur. L'algorithme regroupe les observations à l'aide de l'algorithme *Locally Linear Embedding*

(LLE) et du descripteur HoG. Une fois les catégories identifiées, un classifieur en cascade est généré à partir d'Adaboost. La méthode nécessite un temps d'apprentissage de deux jours et le temps de détection est de 1.5 seconde pour une image en 640x480 ce qui rend l'utilisation de cette méthode peu appropriée pour le suivi. Bien que la méthode utilise une structure en cascade afin de réduire le nombre de tests, le grand nombre de points de vue rend la méthode très coûteuse en ressource et complexe.

2.4.4 Algorithmes de suivi

Les algorithmes de détection vus précédemment n'ont d'intérêt pour le suivi que si leurs détections peuvent être mises en commun temporellement. De nombreuses approches ont été proposées à cette fin, principalement pour le suivi de piétons vu la grande difficulté de suivre individuellement les piétons d'un groupe avec les approches traditionnelles.

L'approche de Breitenstein *et al.* (2011) est une méthode utilisant un modèle de Markov d'ordre 1 (utilisant seulement l'observation courante et la dernière observation) pour le suivi. Pour chaque détection, un filtre de particules est initialisé et un modèle de l'objet est appris en ligne à partir des observations. Le degré de confiance en la détection est utilisé dans un algorithme d'association de type glouton (*greedy*).

Wang *et al.* (2013) utilisent une technique basée sur les détections afin de générer des courtes *tracklets* fiables, c'est-à-dire une association entre les détections de trames consécutives. La position, la taille et la couleur sont utilisées comme critères d'association. Une fois les tracklets générés pour l'ensemble de la vidéo, un graphe est généré à partir de ceux-ci et une procédure d'optimisation est utilisée afin de trouver les associations. Pour ce faire, un critère est défini afin de minimiser la distance moyenne parcourue de tous les objets

Andriluka *et al.* (2008) approchent le problème de la détection et du suivi différemment des autres méthodes de détection de piéton. Dans cette méthode, un modèle est conçu afin de détecter individuellement les parties articulées d'un piéton. Un modèle de déplacement des différents segments et joints du corps humain est ainsi produit afin de vérifier la qualité des détections et éliminer les faux négatifs. Une des limitations courantes de la méthode est que celle-ci fonctionne seulement pour des vues de côté de piéton, car le modèle est bidimensionnel. Cette méthode est donc difficilement utilisable pour le suivi à une intersection puisqu'il y a en général plusieurs directions de déplacement.

2.5 Application du suivi au domaine du transport

2.5.1 Application

Le domaine du transport est un domaine d'application de choix pour le suivi. Il peut servir à compter les véhicules, estimer leur vitesse afin de fournir des informations de circulation détaillées, détecter les accidents et analyser la sécurité. On peut également se servir d'informations de comptage pour optimiser la durée des feux de circulation en zone urbaine. Le suivi en zone urbaine est plus complexe que le suivi sur autoroute puisque dans le cas du suivi urbain, le nombre d'entrées/sorties des objets est inconnu. De plus, il y a plusieurs types d'objets à traiter, c'est-à-dire des objets articulés comme des piétons et des objets rigides comme des véhicules. Finalement, il y a plusieurs types de mouvements possibles pour la circulation en milieu urbain. La revue de littérature de Buch *et al.* (2011) sur le suivi urbain fait d'ailleurs la distinction entre les deux catégories ainsi qu'une comparaison d'un certain nombre de méthodes pour chacun des types de suivi. La prochaine section traitera d'abord du suivi sur autoroute et ensuite du suivi en milieu urbain.

2.5.2 Suivi sur autoroute

Le suivi sur autoroute est apparu bien avant le suivi en milieu urbain puisque la complexité y est réduite. Un des premiers systèmes est celui de Beymer *et al.* (1997). Pour chaque caméra de ce système, il faut définir une zone d'entrée et une zone de sortie. Une détection de coins est ensuite appliquée dans la zone d'entrée et on effectue ensuite le suivi de ces points. Une corrélation est effectuée pour trouver les correspondances entre les points. Un filtre de Kalman est utilisé pour filtrer les mauvais points et prédire la fenêtre de recherche utilisée pour la corrélation. Pour prendre en compte l'effet de perspective, on change la taille des coins à corréler selon la dernière position en coordonnée globale (une homographie est utilisée pour obtenir cette position) du point dans l'image. Les points sont ensuite groupés en fonction de leur mouvement commun à l'aide de la technique décrite à la section 2.2.4, la méthode similarité minimale maximale. Ce système a été implémenté entièrement au niveau matériel sur des DSP afin de faire le traitement en temps réel sur des autoroutes.

Batista *et al.* (2006) présentent une méthode dont la première étape est la détection d'objets basée sur une soustraction d'arrière-plan. Cette soustraction d'arrière-plan est capable de détecter les ombres et les reflets des lumières. La seconde phase de l'algorithme consiste à regrouper les pixels sous forme d'objets. Pour ce faire, le système divise les points de l'avant-plan en blocs 8x8. On doit ensuite calculer la corrélation entre les points de trame en trame afin de trouver les correspondances. Une matrice de correspondance est utilisée afin de permettre la détection des cas où un objet entre, fusionne, se sépare et sort de la

scène. Finalement, les blocs sont mis en commun sous forme d'objets à l'aide d'une fonction d'énergie.

Tamersoy et Aggarwal (2009) présentent également une méthode basée sur la soustraction d'arrière-plan (mélange de gaussiennes), toutefois l'approche pour le suivi est différente de la méthode précédente. Le système utilise la détection afin de trouver les véhicules dans les blobs de la soustraction d'arrière-plan. Pour ce faire, il utilise une approche d'apprentissage machine basée sur un SVM et le descripteur HoG. L'apprentissage est une étape qui peut être très longue, car il faut manuellement annoter des images de l'objet à détecter et des images qui ne contiennent pas l'objet à détecter. Pour résoudre ce problème, l'algorithme utilise les blobs de la soustraction d'arrière-plan pour faire l'apprentissage. Puisque les caméras utilisées sont très loin de la scène, la taille de véhicules est très similaire sauf pour les camions et les autobus. On prend donc les blobs ayant une taille similaire aux automobiles et on se sert de ceux-ci pour faire l'apprentissage automatique d'un classificateur de véhicules. Les blobs plus grands que des véhicules normaux sont considérés comme étant potentiellement en occultations. Afin de détecter les véhicules en état d'occultations dans les gros blobs, on utilise une fenêtre glissante à une seule échelle et on effectue le calcul du descripteur HoG. Ce descripteur est ensuite passé aux SVM afin que celui-ci le classifie et détermine s'il y a un véhicule dans la fenêtre. On obtient donc des détections dans les cas où les véhicules sont en occultation ou seul. On utilise alors un modèle de mouvement pour mettre en commun les détections. La principale limitation de cette méthode est qu'elle ne fonctionne pas pour les grands véhicules. L'auteur explique que ce mauvais comportement est dû au fait que l'entrainement n'a été effectué qu'avec des données provenant de véhicules de taille moyenne et que la fenêtre de détection est trop petite.

2.5.3 Suivi en milieu urbain

Le suivi en milieu urbain impose un certain nombre de difficultés supplémentaires par rapport au suivi sur autoroute, toutefois plusieurs méthodes urbaines sont des améliorations de méthode initialement conçues pour les autoroutes.

Song et Nevatia (2005) présentent une méthode qui utilise une soustraction d'arrière-plan afin de détecter les objets d'intérêt. Par la suite, on utilise des modèles 3D afin de générer des silhouettes de véhicules dans plusieurs angles de vues. On tente alors de trouver des correspondances entre les silhouettes et les blobs. Une fois les détections obtenues, on les associe à l'aide d'un modèle de mouvement et les objets sont ensuite suivis à l'aide d'un filtre de Kalman. Cette technique a pour avantage de gérer les occultations et de classifier les véhicules en même temps. L'inconvénient est qu'il est difficile d'avoir un modèle acceptable capable de gérer tous les véhicules et la technique ne permet pas le suivi des piétons.

L'approche de Saunier et Sayed (2006) est inspirée de la méthode Beymer *et al.* (1997) vue à la section 2.5.2. Toutefois, elle est généralisée au milieu urbain en éliminant la nécessité d'avoir des zones d'entrées et de sorties. La méthode effectue un suivi KLT afin de faire la détection et le suivi des points à travers le temps. Afin de filtrer les mauvais points, les trajectoires de longueur insuffisante et les points n'ayant pas un mouvement minimal sont éliminés. Cela permet de filtrer les bruits dus au feuillage et les points qui ne sont pas assez stables. Les trajectoires ainsi générées sont enregistrées dans un fichier et il est ensuite possible de faire le groupement en mode hors ligne à l'aide de la méthode de similarité minimale maximale. Le résultat de l'application de cette méthode d'association est visible à la figure 2.9. On peut y voir les points retenus pour le groupement ainsi que l'estimation de taille obtenue. Puisque le regroupement est basé sur le mouvement commun des points d'un objet, la méthode fonctionne mieux sur les objets rigides tels que les véhicules et moins bien sur les piétons. On peut d'ailleurs voir à la figure 2.10 la boîte englobante minimale obtenue pour les points suivis sur des piétons. Les points qui sont groupés sont ceux situés sur des parties se déplacent rigidement ce qui donne une mauvaise estimation de la taille des objets. De plus, puisque la taille des véhicules et des piétons est assez différente, il est difficile de trouver des seuils de distance permettant de suivre simultanément les piétons plus éloignés de la caméra et les véhicules ce qui est problématique pour les scènes urbaines à circulation mixte.

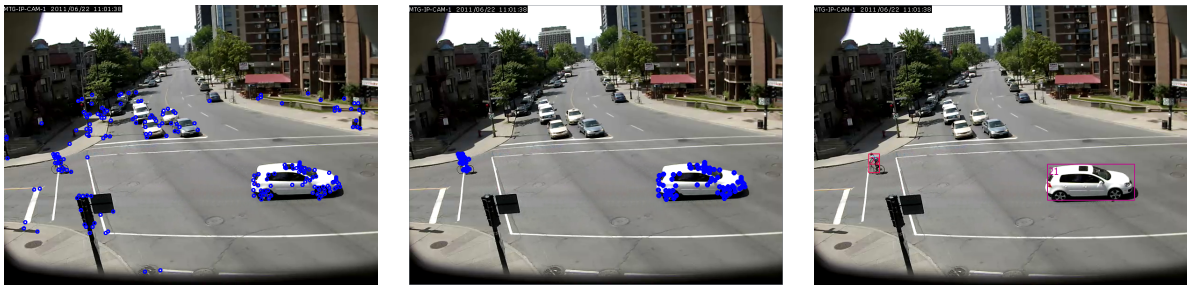


Figure 2.9 a) Positionnement des points avec l'algorithme de Saunier et Sayed (2006) b) Points valides considérés dans le groupement c) Groupement de points résultants

La méthode de Kim (2008) combine la soustraction d'arrière-plan et le suivi de points caractéristiques. La méthode est conçue de façon à ce que les points détectés servent d'indice pour éliminer les ombres, les mouvements mineurs comme les feuilles d'arbres et gérer certains changements d'illumination brusque. L'arrière-plan est utilisé comme indice pour la détection et le groupement des points détectés. La soustraction d'arrière-plan utilisée est une différence de trames suivie d'un seuillage. Une analyse en composantes connectées est ensuite effectuée afin d'éliminer les trous, remplir les petits objets et obtenir les blobs des objets. Une détection de points est effectuée en calculant les valeurs propres (eigenvalue) des pixels du blob de



Figure 2.10 a) Positionnement des points avec l'algorithme de Saunier et Sayed (2006) b) Groupement de points résultants

l'avant-plan. Les valeurs propres des points de l'arrière-plan sont également calculées et mises à jour dans une liste afin de les comparer avec les points futurs. Pour s'assurer de la validité des points de l'avant-plan, on compare leurs valeurs propres à ceux de l'arrière-plan et on élimine ceux qui correspondent à des points d'arrière-plan. Pour qu'un point caractéristique soit considéré comme valide, il doit être suivi sur au moins trois trames consécutives. Pour le groupement des points caractéristiques, la position, la direction du mouvement, l'historique de groupement et le blob de soustraction d'arrière-plan sont utilisés.

2.6 Gestion de la perspective

La perspective de la caméra a un impact important sur les suivis et les paramètres qu'il est possible d'extraire. En effet, la perspective a un impact sur le calcul de la vitesse de déplacement des objets et leur taille. Ainsi, un objet situé loin de la caméra sera beaucoup plus petit et il se déplacera beaucoup plus lentement qu'un objet plus proche. Afin d'avoir des données uniformes indépendamment de la position de l'objet dans l'image, il faut ramener les informations dans un référentiel où l'effet de la perspective n'est pas présent, c'est-à-dire une vue où la caméra pointe directement vers le sol (vue aérienne). De ce point de vue, les véhicules auront les mêmes dimensions partout dans l'image et un pixel représentera la même distance physique ce qui aura pour effet que nous pourrons calculer sa vitesse en unités physiques en multipliant simplement par le ratio pixels/mètres. Cette méthode ne fonctionne bien que si la scène est située sur un plan. Plus les points sont éloignés du sol, moins leur projection sera fiable puisqu'ils ne sont pas situés sur le plan.

2.6.1 Correction de la perspective

Pour convertir les coordonnées de la vue perspective vers la vue aérienne, on peut utiliser une matrice d'homographie qui a la forme de l'équation 2.4 pour faire une transformation linéaire d'un point p_1 situé dans un plan projectif π_1 à un point équivalent p_2 situé dans le plan projectif π_2 en appliquant l'équation 2.5.

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{2.4}$$

$$p_2 \sim Hp_1 \tag{2.5}$$

La matrice d'homographie peut être estimée à partir des positions d'au moins 4 points non colinéaires. Si on a 4 points, on peut trouver directement la solution unique ; si on en a plus de 4, alors il faut utiliser une fonction qui permet de minimiser l'erreur. L'approche la plus populaire pour cela est d'utiliser RANdom SAmple Consensensus (RANSAC) de Fischler et Bolles (1981). L'application de cette méthode au calcul de l'homographie est détaillée dans Hartley et Zisserman (2003). Pour ce faire, il faut prendre 4 paires de points aléatoires parmi les paires de points définis et calculer l'homographie. On calcule ensuite la projection des autres points à l'aide de cette homographie et on calcule l'erreur. On garde ensuite l'ensemble avec le plus de points correctement estimés. On utilise ensuite la méthode des moindres carrés sur les points correctement estimés afin d'obtenir la matrice d'homographie.

Une fois la matrice d'homographie calculée, on peut calculer la position d'un point dans un des deux plans, caméra ou au sol, et trouver le point correspondant dans l'autre plan. Un exemple de résultat obtenu est visible à la figure 2.11.

2.6.2 Suivi multi caméras

La possibilité de projeter un point de vue vers un autre point de vue rend possible l'utilisation de plusieurs caméras pour effectuer le suivi ce qui permet de réduire les problèmes d'occultations et dans certains cas, réduire l'erreur d'estimation de la position du barycentre de l'objet. La méthode de Atev *et al.* (2005) permet d'estimer la boîte englobante 3D d'un objet à partir d'une ou plusieurs vues ce qui permet d'estimer plus précisément s'il y a des collisions possibles entre plusieurs objets. La méthode de Dahlkamp *et al.* (2004) utilise un modèle paramétrique 3D projeté sur la scène afin de détecter l'orientation et la position des objets. La position du barycentre est ainsi connue précisément et on peut résoudre des problèmes d'occultations partielles. Le problème avec ce type d'approche est qu'il est très difficile

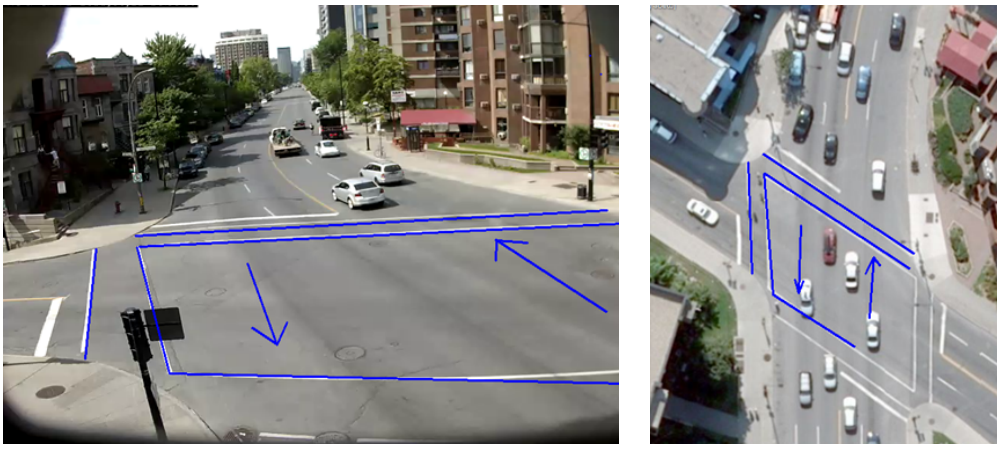


Figure 2.11 a) Vue en perspective obtenue par la caméra avec des annotations dessinées. b) Vue aérienne avec les annotations transformées à l'aide de la matrice d'homographie. L'image aérienne est tirée de Bing Maps, ©2013 Microsoft Corporation.

d'avoir un modèle 3D de tous les types de véhicules en circulation et d'usagers de la route. Cette approche est donc peu réaliste en pratique.

2.7 Retour sur les méthodes

Pour notre problématique, il est important de pouvoir suivre plusieurs types d'objets différents puisque les données visent à étudier l'interaction entre les différents usagers de la route. Il faut limiter le plus possible les interventions humaines dans le traitement puisque le système devra traiter un très grand nombre de données (quelques heures à plusieurs jours de vidéos). Un grand nombre de données est nécessaire afin de couvrir l'ensemble des interactions pouvant causer un accident. Afin d'obtenir une bonne précision dans la détection d'accidents et de quasi-accident, une bonne estimation de la taille des objets est nécessaire. Le système devra également être en mesure de gérer un grand nombre d'occultations puisque les caméras sont situées près du sol et qu'avec ce point de vue il y a beaucoup d'occultations entre les différents utilisateurs routiers. Voici les avantages et inconvénients de chaque méthode par rapport à ces critères.

Les méthodes à base de blobs ont pour avantage de fournir une bonne estimation de la taille des objets. La présence d'ombres et de reflets peut toutefois causer une surestimation de la taille des objets. De plus, certains problèmes peuvent causer une sursegmentation des objets, c'est-à-dire que plusieurs parties d'un même objet sont détectées comme étant des objets différents. Des occultations et des chevauchements peuvent également causer des problèmes de sous-segmentation. Le suivi de blobs par descripteurs globaux comme les histogrammes

ou par la forme a comme inconvénient que ceux-ci sont très sensibles à l'occultation partielle, ce qui arrive très fréquemment lors du suivi multi objet.

Pour leur part, les méthodes à base de points ont pour avantage de bien résister aux occultations puisqu'il ne faut en théorie qu'un seul bon point caractéristique pour associer deux observations. De plus, les points caractéristiques offrent une bien plus grande résistance aux changements de luminosité que les méthodes à base de couleur. La grande difficulté de la méthode demeure toutefois au niveau du groupement des points. Les trajectoires obtenues sont rarement complètes alors il faut générer des nouvelles trajectoires durant le suivi afin de garder le modèle à jour. Le groupement des points peut être fait en fonction du mouvement commun ou encore par proximité. Toutefois, la densité de points sur un objet n'est pas uniforme et le critère de mouvement commun peut ne pas fonctionner sur les objets non rigides.

Les méthodes à base de détection sont assez robustes et ont principalement été développées pour le suivi de visage et de piéton. De plus, elles peuvent être utilisées sur des flux vidéo provenant de caméras qui ne sont pas fixes. Cette méthode n'est toutefois pas appropriée lorsqu'on veut détecter plusieurs types d'objets dans plusieurs angles différents puisque le nombre de détecteurs nécessaire serait très élevé. De plus, pour les objets comme les véhicules qui ont des formes très différentes selon le point de vue et le modèle, il faudrait plusieurs détecteurs afin de couvrir les différents points de vue de chaque véhicule. Chaque détecteur nécessite un long apprentissage et la combinaison de plusieurs détecteurs augmente le nombre de faux positifs générés par ceux-ci. Étant donné le grand nombre de détecteurs nécessaires, ce type de méthode ne peut pas directement être utilisé pour notre application.

Notre méthode utilisera une combinaison des méthodes par blob et des méthodes par points. Les blobs seront utilisés comme indice de groupement et afin d'estimer la taille. Les points seront pour leur part utilisés afin de rendre la méthode résistante aux occultations et faciliter l'association des blobs. Les fusions et défusions des blobs seront gérées de façon à conserver l'historique des objets individuels qui les composent. Les points seront utilisés afin d'estimer la boîte englobante des objets durant les occultations.

CHAPITRE 3

MÉTHODOLOGIE

Le système créé dans le cadre de notre recherche, nommé Urban Tracker (UT), est un système complet de suivi conçu pour traiter des données réelles. Une partie importante du travail effectué consiste donc à gérer globalement les différentes imperfections pouvant survenir à chacune des étapes du suivi plutôt que d'essayer d'optimiser une étape en particulier. Les imperfections que nous voulons gérer sont principalement des problèmes de segmentation qui peuvent être dues à des changements de luminosité, du bruit dû au capteur de la caméra, des ombres, etc. Notre travail se distingue donc de plusieurs autres méthodes visant à améliorer une partie spécifique du processus de suivi par exemple l'association des données ou la détection des objets. Puisque nous désirons traiter un grand nombre de données avec un nombre minimal d'intervention humaine, le système doit également effectuer la détection des objets d'intérêt automatiquement et éliminer sans interventions humaines les objets suivis qui ne sont pas d'intérêt (par exemple les ombres dues à des oiseaux ou autres).

Notre méthode est une méthode en ligne, c'est-à-dire que les décisions de suivi sont prises de trame en trame plutôt qu'une fois l'entièreté des observations connut. Les avantages de procéder ainsi sont que le système peut fonctionner en temps réel (en admettant que les performances de la machine soient suffisantes) et que la méthode consomme peu de mémoire, car elle ne doit charger que les objets d'un nombre de trames limité en mémoire. Notre méthode possède la capacité de corriger certaines observations précédentes en fonction des observations courantes afin d'améliorer la qualité des pistes destinées à l'analyse. Par exemple, certains objets suivis sont éliminés avant l'enregistrement comme les ombres projetées et les feuilles d'arbres. En effet, notre méthode ne permet pas de déterminer avec certitude qu'il s'agit d'objets indésirables dans les premières trames, toutefois, il est possible de les identifier et les supprimer une fois que nous avons accumulé suffisamment d'information.

Ce chapitre explique la méthode ainsi que les algorithmes utilisés pour corriger les imperfections de chacune des étapes du suivi. Nous présenterons d'abord la terminologie utilisée dans le reste du chapitre et nous expliquerons ensuite notre méthode en détail.

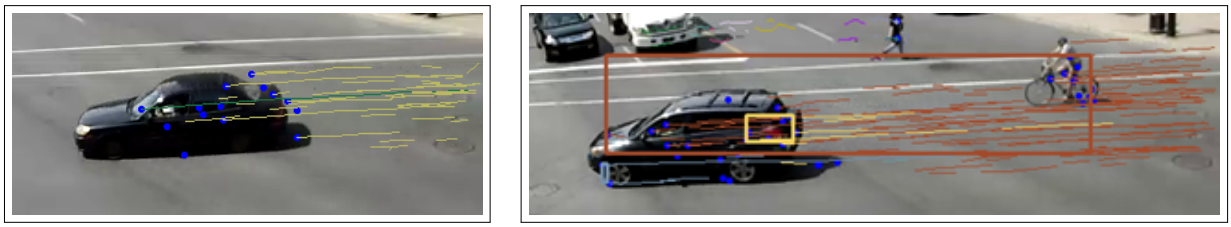
3.1 Terminologie utilisée et définitions de base

Afin de limiter l'ambigüité possible dans la description de notre méthode, il est nécessaire d'établir une terminologie de base qui sera utilisée tout au long du chapitre. Dans le contexte

de ce travail, une scène est un lieu filmé par une caméra vidéo. Un objet réfère à une structure informatique contenant les différentes observations d'un objet physique suivi dans une scène filmée. Un modèle de suivi est un ensemble de données permettant de décrire un objet telles que sa couleur, ses points caractéristiques, sa vitesse, etc. Une observation est le modèle de suivi d'un objet à un temps t. Un groupe d'objet est une structure contenant les observations d'un ensemble d'objets se déplaçant à proximité, mais qui maintiennent des modèles séparés pour chacun d'entre eux en plus d'un modèle de groupe. Une trajectoire est l'évolution spatiotemporelle d'un point caractéristique.

3.2 Méthode développée

Notre méthode est une combinaison d'une approche par blobs avec une approche par points caractéristiques. Les méthodes utilisant seulement des blobs ne sont pas en mesure de gérer les occultations partielles tandis que le défi à résoudre pour les méthodes par points est qu'il est difficile de grouper les points en un seul objet et de définir ses frontières. Les critères de groupement de points basés sur la rigidité des objets suivis ne permettent pas de gérer efficacement les objets de tailles différentes et il n'est pas rare qu'il y ait des problèmes avec certains points individuels lors du suivi tel que montré à la figure 3.1. Dans l'image 3.1.a), la trajectoire verte centrale n'est pas groupée avec les autres trajectoires (en jaune), car celui-ci a eu plusieurs mauvaises correspondances sur l'axe horizontal (puisqu'il n'est pas assez caractéristique). Dans la seconde image, on peut voir le type de problème de segmentation possible. Dans ce cas-ci, le véhicule et le vélo se sont déplacés à la même vitesse suffisamment longtemps pour être groupés alors que certains points du véhicule ont été groupés séparément de celui-ci. Il y a donc plusieurs types de problèmes pouvant survenir avec cette approche. Certains de ces problèmes peuvent être résolus par exemple en exigeant un nombre minimal de points par objets ou en contraignant davantage la distance maximum entre deux points d'un même objet. Toutefois, il est difficile de définir des contraintes fonctionnant pour des objets de tailles différentes.



a) b)

Figure 3.1 Problème d'une méthode à base de points. a) Trajectoire stationnaire non groupée b) Problème de groupement

Une autre approche envisagée pour le groupement consiste à faire de la détection d'objets et à utiliser les points situés dans les détections pour associer les détections de trame en trame. La détection est déjà utilisée pour le suivi de piétons et la qualité du suivi est bonne même en présence d'occultation assez sévère entre les objets (voir section 2.4.2). Cette approche est toutefois beaucoup moins pratique pour le suivi de véhicules puisque la forme d'un véhicule change énormément en fonction du modèle et de l'angle de vue. Il faut donc entrainer plusieurs détecteurs, c'est-à-dire un détecteur pour chaque angle de vue de chaque modèle de véhicule. Les méthodes actuelles de détection de véhicules assument généralement qu'il y a un nombre limité d'angles de vue (détection à partir d'une caméra fixe sur une autoroute, suivi de véhicule à partir d'un autre, etc.). Pour être en mesure de réaliser cette approche, il faudrait donc entrainer un détecteur pour les piétons et plusieurs détecteurs pour chaque type d'usagers de la route (puisque ceux-ci variant fortement selon l'angle et le type). Vue la grande variété du trafic routier, il a été décidé de ne pas utiliser cette approche puisqu'un grand nombre de détecteurs entrainerait également un grand nombre de fausses détections en plus d'un temps de traitement très long. De plus, les objets inconnus des détecteurs ne pourraient pas être suivis.

Puisque l'approche par détection n'est pas suffisamment généralisable pour notre problématique multiobjet et multiangle, nous avons décidé d'utiliser les blobs pour grouper les points. Les blobs permettent ici de détecter les objets indépendamment de leur type, leur taille ou de leur angle en plus de nous donner une estimation de leur taille. Cela nous donne donc la flexibilité de suivre des objets complètement inconnus du système. Puisque les blobs résistent mal aux occultations, nous avons décidé de nous servir des points caractéristiques pour le suivi. Les points nous servent également à estimer la position des objets lorsque les blobs sont en état d'occultation. L'usage de blobs entraine toutefois la contrainte qu'il faut que les objets soient séparés pendant une partie du suivi pour que ceux-ci ne soient pas surgroupés. Nous présenterons ces méthodes et la technique de suivi plus en détail dans les sections suivantes.

3.2.1 Aperçu de la méthode

La figure 3.2 présente une vue de haut niveau du système complet. Le module trames de vidéo fournit séquentiellement les trames de vidéo à l'entrée du système de suivi. Le premier bloc du système consiste à extraire les blobs des objets d'intérêt de la trame. Pour ce faire, le premier module du bloc est le module de soustraction d'arrière-plan. Ce module permet de déterminer quels sont les pixels représentant un objet en mouvement en utilisant les changements d'intensité entre la trame courante et les trames précédentes. Les pixels détectés sont ensuite passés au module d'analyse en composante connexe afin de déterminer

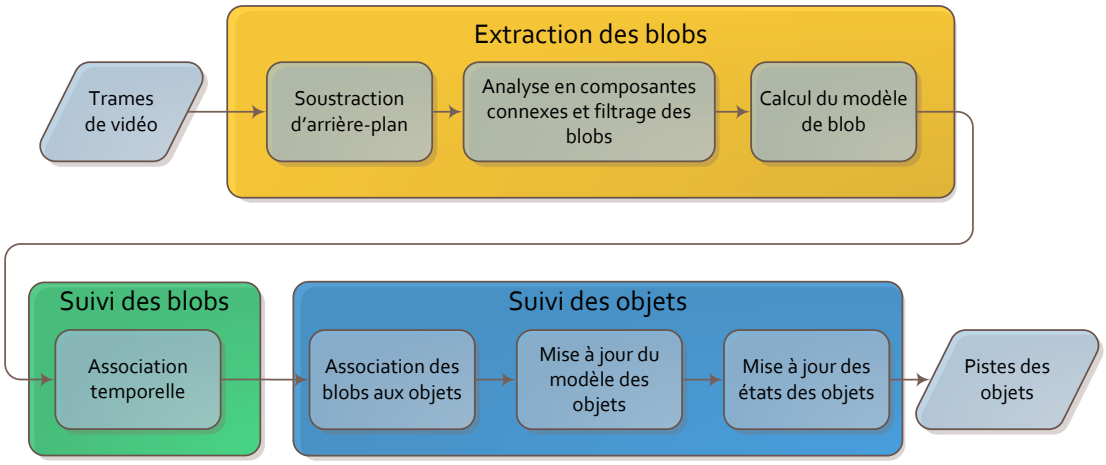


Figure 3.2 Schéma bloc du système de suivi

quels pixels sont connectés spatialement et de les regrouper sous forme de blob. Les blobs sont ensuite filtrés afin d'éliminer les blobs de trop petite taille et les blobs statiques. Une fois les blobs traités, leur modèle est calculé. Le processus de suivi est alors divisé en deux étapes. La première étape consiste à faire l'association entre les blobs de trame en trame. Les associations calculées seront utilisées par la deuxième partie du suivi, le suivi d'objets. Cette partie du suivi s'occupe de faire l'association entre les blobs suivis sur deux trames et les objets suivis sur l'ensemble de leur passage dans la scène. Selon les types d'association de blobs trouvés lors du suivi de blob, le module d'association des blobs aux objets crée, supprime, associe, et divise des objets ou des groupes d'objets. Le modèle de chacun de ces objets est ensuite mis à jour en fonction des associations faites avec les blobs. Afin de gérer le cycle de vie des objets et d'augmenter la qualité du suivi, une machine à états est utilisée. Ces états sont mis à jour à chaque trame en fonction des associations et de la position de l'objet dans la scène. Les objets sortant de la scène sont ensuite filtrés afin de déterminer s'il s'agit d'objets valides devant être enregistrés ou s'il s'agit de bruit devant être éliminé.

3.3 Extraction des blobs

3.3.1 Soustraction d'arrière-plan

La technique de soustraction d'arrière-plan que nous avons utilisée est la technique Visual Background extractor (ViBe) de Barnich et Van Droogenbroeck (2011). Cette technique a

été choisie puisqu'elle s'est bien classée lors du concours changedetection.net de Goyette *et al.* (2012) et elle est très rapide à l'exécution en plus de nécessiter une courte période d'apprentissage du modèle de référence. Afin de réduire le bruit dans l'avant-plan généré par la soustraction d'arrière-plan, un flou gaussien est appliqué sur l'image en prenant en compte une déviation σ de 1.5 pixel et une fenêtre de 5x5 pixels autour de chaque pixel de l'image. La description de ce filtre est donnée par l'équation 3.1 où x et y représentent la distance sur l'axe des abscisses et l'axe des ordonnées de la position du point par rapport au pixel central. Le résultat de ce filtrage est visible à la figure 3.3 b). On peut observer en comparant les figures 3.3 c) et d) que l'utilisation de ce filtre a un impact important sur la réduction du bruit après la soustraction d'arrière-plan. Afin de combler les trous dans les objets, un opérateur morphologique de dilatation (avec 4 itérations) suivi d'une érosion est appliqué sur la soustraction d'arrière-plan afin de combler les trous résultants et associer les groupes de pixels très proches afin de réduire les risques de sursegmentation. L'importance de filtrer le bruit est démontrée par les figures 3.3 e) et f) qui montrent le résultat de la dilatation avec et sans filtre gaussien. Les figures 3.3 g) et h) montrent la soustraction d'arrière-plan après suppression des objets trop petits.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.1}$$

3.3.2 Analyse des composantes connexes

Afin de créer des blobs à l'aide des pixels d'avant-plan, une technique d'analyse en composantes connexes est utilisée. La technique de Chang *et al.* (2004) tel qu'implémentée dans la librairie cvBlob de Linan (2012) a été utilisée. Cette technique est capable de trouver les connexions sur les 8 côtés d'un pixel ainsi que les contours du blob.

3.3.3 Filtrage des blobs

Afin d'éliminer les blobs de bruit restants, un filtre est appliqué afin de supprimer tous les blobs plus petits qu'une certaine taille. Ce paramètre, le paramètre «Taille minimale des blobs (T_m)» doit être ajusté selon l'aire minimum des objets que l'on désire suivre.

Certains blobs retournés par la soustraction d'arrière-plan sont dus à des effets de « ghosting », un problème qui survient lorsqu'un objet s'arrête pendant une longue durée et qu'il recommence ensuite à se déplacer. Étant donné qu'il fait alors partie du modèle d'arrière-plan de la méthode de soustraction d'arrière-plan, son absence est alors considérée comme étant un blob en lui-même. Puisque nous traitons des données d'intersections urbaines, des blobs de ce type ont tendance à se former dans l'intersection à chaque fois qu'un véhicule

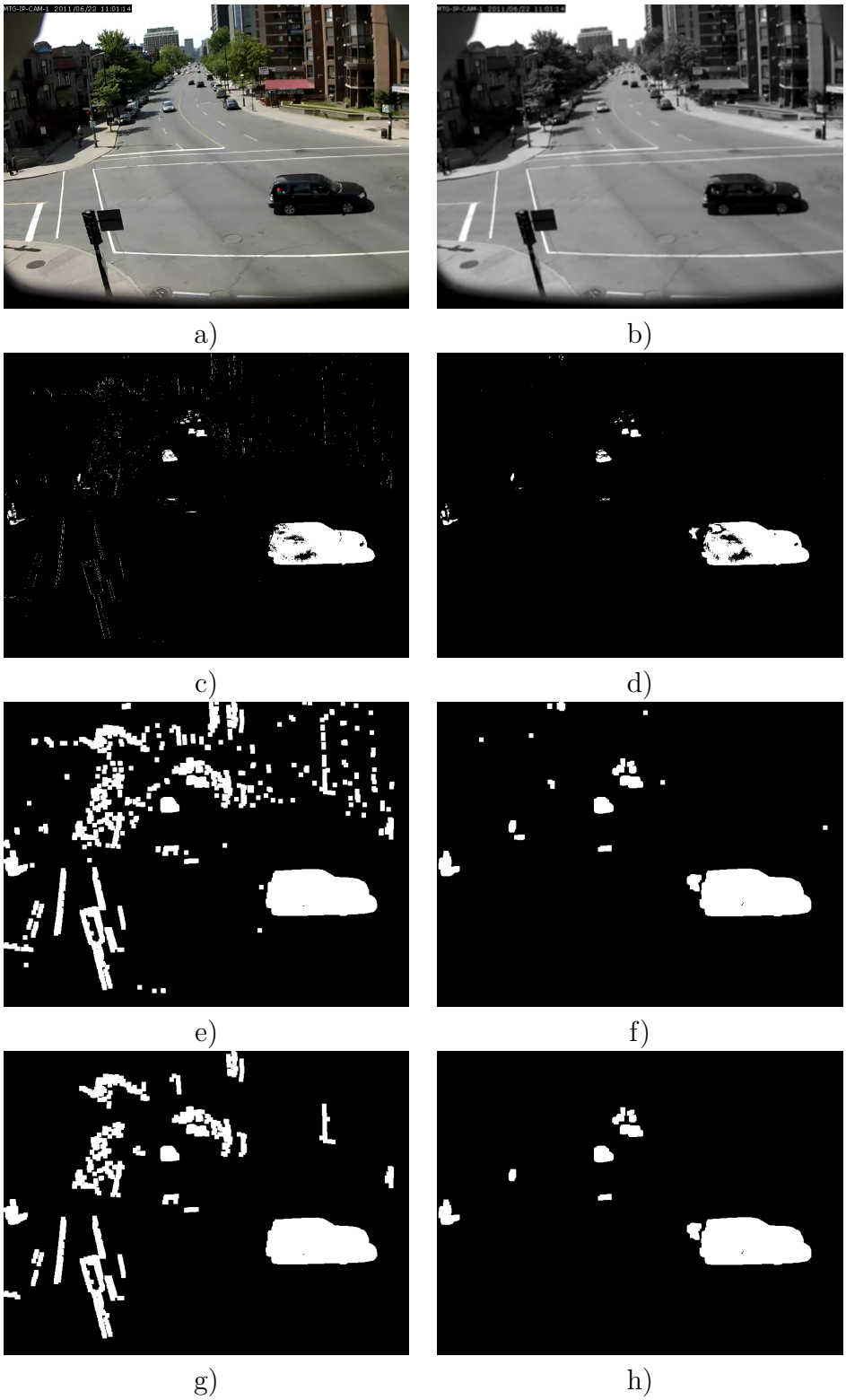


Figure 3.3 **a)** Trame originale **b)** Trame après application du filtre gaussien **c)** Soustraction d'arrière-plan à partir de l'image a) **d)** Soustraction d'arrière-plan à partir de l'image b) **e)** Résultat de l'application d'une dilatation sur c) **f)** Résultat de l'application d'une dilatation sur d) **g)** Résultat de la suppression des blobs dont l'aire est inférieure à 200 px dans e) **h)** Résultat de la suppression des blobs dont l'aire est inférieure à 200 px dans f)

s'arrête à un feu de circulation et à chaque fois qu'il repart. Afin de résoudre ce problème, nous avons modifié ViBe afin d'éliminer les blobs constitués de pixels statiques. Nous désirons donc rajouter une vérification afin de nous assurer que les blobs détectés sont bel et bien en mouvement. Afin de déterminer si des pixels ont changé entre deux images, nous prenons la valeur absolue de la différence entre l'image courante et la précédente. Nous comptons ensuite pour chaque blob le nombre de pixels ayant subi un changement d'intensité supérieur à σ_{cam}, l'écart type des pixels dus au bruit du capteur de la caméra. Une valeur de 4 pour σ_{cam} permet de gérer les scènes bruitées. Le nombre de pixels comptés est ensuite divisé par le nombre de pixels total du blob. Le paramètre «Variation interblob des pixels (V_p)» est utilisé afin de déterminer le nombre minimal de changements des pixels dans un blob pour que celui-ci ne soit pas considéré comme fantôme. Nous proposons une valeur de 10 %. Afin de l'éliminer, nous accédons alors au modèle d'arrière-plan maintenu par ViBe et nous le modifions afin que les pixels du blob fantôme dans le modèle aient maintenant les mêmes valeurs que ceux de l'image courante. Cela a donc pour effet de mettre à jour instantanément le modèle avec les valeurs courantes et d'éliminer le fantôme. Ce procédé est illustré à la figure 3.4 et le pseudo-code est donné à l'algorithme 1. Cette méthode ne nous permet pas de faire la distinction entre les fantômes et les objets qui s'arrêtent temporairement. Puisque nous voulons garder la même identité pour les objets qui s'arrêtent temporairement, une méthode a été développée afin de redonner la bonne identité aux objets s'étant arrêtés temporairement (à cause d'une lumière par exemple) qui recommencent à bouger. Celle-ci est décrite à la section 3.4.7.

Algorithme 1 Enlever les fantômes de la soustraction d'arrière-plan

I_{t-1}, l'image à $t-1$
I_t, l'image à t
I_{Label}, matrice indiquant l'appartenance des pixels à des blobs
$BlobDict$, dictionnaire permettant d'accéder aux attributs d'un blob par identifiant
$NbValidePx$, dictionnaire permettant de classer le nombre de pixels valides par identifiant
$\Delta_I = abs(I_t - I_{t-1})$
pour chaque coordonnée x,y dans I_{Label}
    $BlobId = I_{Label}(x, y)$
    si Existe($BlobId$) et $\Delta_I(x, y) > \sigma_{cam}$
        $NbValidePx[BlobId] = NbValidePx[BlobId] + 1$
pour *blob* in $BlobDict$
    si $NbValidePx[blob.id]/blob.aire < V_p$
        On met à jour le modèle ViBe avec les pixels de l'image courante pour *blob*

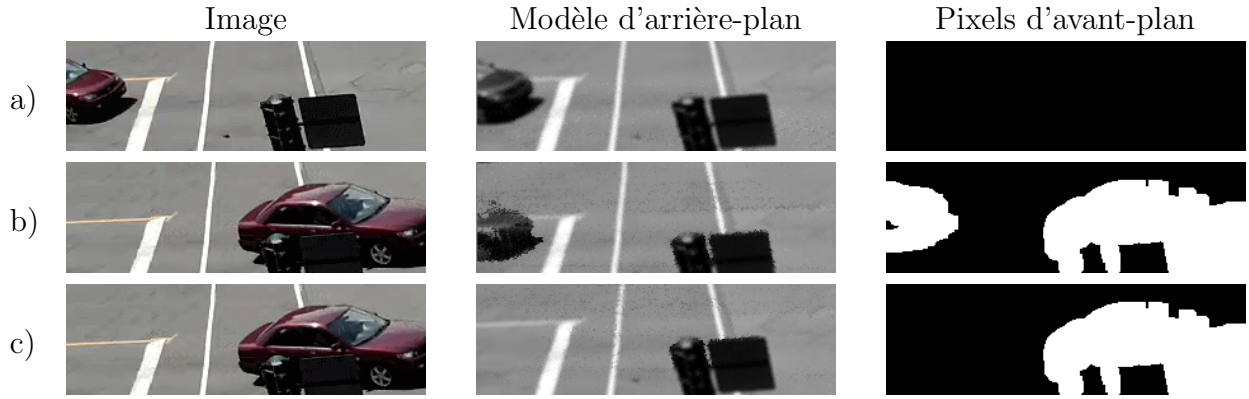


Figure 3.4 a) Un véhicule immobilisé depuis longtemps au temps $t = 0$. b) Le véhicule au temps $t = 10$. Celui-ci est encore présent dans le modèle d'arrière-plan, car le modèle ne s'est pas encore adapté à sa disparition. On a donc deux blobs dans les pixels d'avant-plan, un pour l'objet en déplacement et un blob «fantôme» à l'ancien emplacement du véhicule. c) Véhicule au temps $t = 10$, mais avec notre suppression des blobs statiques. Pour ce faire, nous calculons la variation interpixels de chaque blob, c'est-à-dire le pourcentage de pixels qui ont une variation suffisante d'intensité. Si ce pourcentage est suffisamment grand, alors le modèle d'arrière-plan est mis à jour avec les valeurs courantes de l'image pour ce blob ce qui élimine le blob «fantôme» des pixels d'avant-plan.

3.3.4 Calcul du modèle des blobs

Le modèle d'un blob est constitué de sa taille, sa position et des points caractéristiques situés à l'intérieur de celui-ci. Les points caractéristiques sont utilisés afin d'associer les blobs entre eux tout en permettant de gérer les occultations partielles. Le détecteur utilisé pour détecter les points est le détecteur de la méthode BRISK (décrit à la section 2.2.2). La description des points trouvés est réalisée à l'aide du descripteur FREAK (décrit à la section 2.2.3). Le descripteur FREAK a été choisi, car il est plus rapide que le descripteur BRISK à calculer et il est également invariant aux rotations et aux changements d'échelles. Le détecteur BRISK a été utilisé avec 3 octaves et un seuil de détection de 10. Le nombre d'octaves est la valeur par défaut recommandée de l'implémentation de OpenCV par Bradski (2000) pour gérer les changements d'échelles. Pour déterminer le seuil de détection, la valeur par défaut de OpenCV de 30 a été choisie comme référence. Puisque celle-ci ne générait pas suffisamment de points pour gérer correctement les occultations dans les autres parties de l'algorithme, celle-ci a été diminuée jusqu'à ce que les résultats soient considérés comme étant satisfaisants pour les scènes considérées. L'utilisation d'un seuil plus bas entraine inévitablement plus de mauvais agencements de points puisque ceux-ci sont moins caractéristiques. Toutefois d'autres mécanismes sont en place pour gérer les mauvaises paires de points (par exemple, l'utilisation d'un nombre minimal de points nécessaire pour associer deux blobs). Le descripteur FREAK

a pour sa part été utilisé avec les paramètres par défaut, c'est-à-dire un facteur d'échelle de 22 et 3 octaves afin d'être aligné sur BRISK. Il en résulte une bordure sans point caractéristique de 44 pixels sur la vidéo tel que visible à la figure 3.5. Le modèle doit gérer les blobs situés sur cette bordure différemment des autres blobs lors de l'association de ceux-ci puisqu'il ne peut pas utiliser les points.

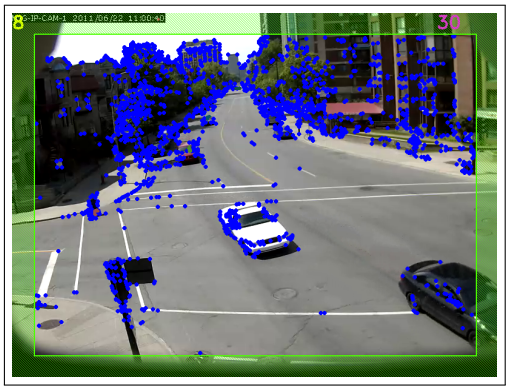


Figure 3.5 Points caractéristiques décrits avec FREAK en utilisant un voisinage de 22 pixels et 3 octaves. La zone hachurée verte est la zone sans points caractéristiques.

3.4 Suivi

Le suivi des objets s'effectue en deux phases. La première phase est de plus bas niveau et consiste à faire le suivi des blobs détectés à l'étape précédente d'une trame à la suivante. La deuxième phase est de plus haut niveau et consiste à faire le suivi des objets. Cette phase fait donc l'association des blobs aux objets et elle met à jour le modèle des objets ainsi que leur état interne.

3.4.1 Suivi des blobs

La première phase du suivi consiste à associer les blobs d'une trame à la suivante. Le but de cette étape est de faire le suivi des blobs qui peuvent ici représenter une partie d'objet, un objet ou même plusieurs objets. Pour chacune des trames, la compatibilité du modèle des blobs courants par rapport aux blobs précédents est vérifiée afin d'identifier les nouveaux blobs, les blobs sortants, les fusions et les divisions de blobs. Pour ce faire, le modèle de suivi défini à la section précédente est utilisé. D'abord, les correspondances entre les points caractéristiques situés à l'intérieur de B_{t-1} et B_t (les blobs aux temps $t-1$ et t respectivement) sont calculés à l'aide de la distance de Hamming de leurs descripteurs. Il n'y a pas de seuil minimum à cette distance de Hamming alors les points sont associés au point le plus proche. Enfin d'éliminer les mauvaises correspondances, nous utilisons plutôt les tests de symétrie

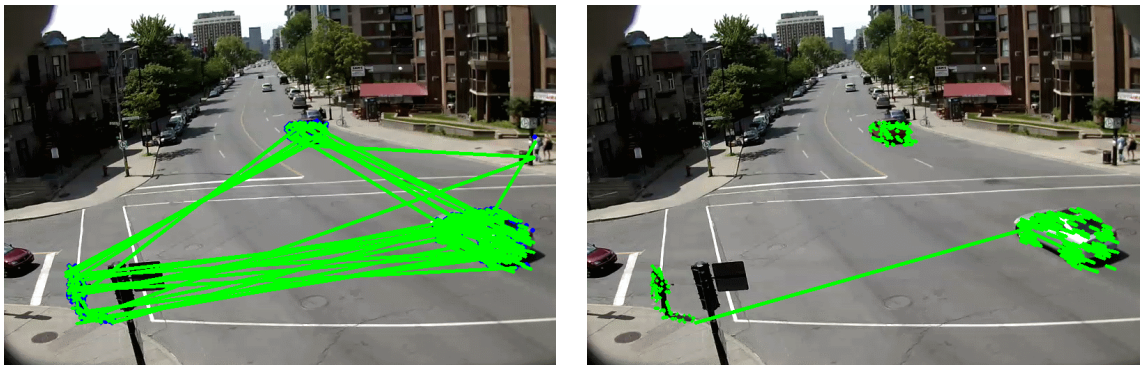
et de ratios décrit à la section 2.2.3. Cela permet de réduire significativement le nombre de mauvaises correspondances (voir figure 3.6) en utilisant un seuil qui est relatif aux autres points de l'image plutôt que fixe. Afin de gérer les mauvaises correspondances restantes, seuls les blobs ayant au moins 4 paires de points correspondants seront associés (réglable à l'aide du paramètre «Nombre minimal de pairs de points (N_m)»). Cette technique est inspirée de Lowe (2004) qui ne considère que les groupes de 3 correspondances puisque ceux-ci ont de bien plus faibles probabilités que les points individuels d'être erronés.

Puisqu'il y a une bordure sans points caractéristiques (voir figure 3.5), nous utilisons également l'information spatiale pour faire les associations. Pour tous les anciens et les nouveaux blobs qui n'ont pas été associés par les points caractéristiques, nous vérifions le recouvrement entre les blobs B_{t-1} non associés et les blobs B_t. Pour vérifier le recouvrement, il faut vérifier si les pixels du blob au temps $t-1$ sont en superposition avec ceux du blob au temps t. Cette méthode présume que les objets doivent se toucher entre 2 trames. Si la vitesse d'acquisition de la caméra est trop lente et que deux blobs dans des trames consécutives ne se superposent pas, il est possible de dilater les blobs par un certain pourcentage afin d'augmenter la tolérance. Cette méthode n'est toutefois pas parfaite lorsque plusieurs blobs sont à proximité. La figure 3.7 démontre ce problème. Dans cette situation, on observe que malgré le fait que le blob A et le blob B sont séparés au temps $t-1$ et t, il est possible que ceux-ci se recouvrent simultanément ce qui donnera que le blob A_{t-1} est devenu le blob A_t et le blob B_t et idem pour le blob B_{t-1} qui devient le blob A_t et le blob B_t également. Le système détecte donc deux fusions alors qu'il devrait détecter une correspondance directe entre le blob A_{t-1} et le blob A_t ainsi que le blob B_{t-1} et le blob B_t. Ceci explique pourquoi les points sont utilisés comme méthode principale de calcul des associations et que le recouvrement est utilisé seulement pour compenser les lacunes de cette méthode.

Les types d'associations détectées sont les associations 0 vers 1 (nouvel objet), 1 vers 0 (sortie d'objet), 1 vers 1 (suivi direct), 1 vers N (séparation d'objet) et N vers 1 (fusion). Puisque les associations se font de blob en blob, il n'y a pas d'association de type M à N.

3.4.2 Suivi des objets

Cette phase du suivi est celle qui fait l'association entre le suivi des blobs individuels et le suivi des objets. Cette partie de l'algorithme nous permet de gérer les problématiques liées à l'utilisation de données réelles provenant d'un environnement non contrôlé. Cela permet de gérer la sous-segmentation des blobs ainsi que la sursegmentation de ceux-ci. Elle permet également de gérer le cycle de vie complet d'un objet incluant ses entrées et sorties en utilisant la machine à état de la figure 3.8. Les transitions de cette machine à état sont quant à elles gérées par les associations des blobs.



a) b)

Figure 3.6 Correspondances de points trouvés entre l'image précédente et l'image courante (affichées sur l'image courante par un trait vert). **a)** Correspondances trouvées sans le test du ratio et le test de symétrie (sans seuil). **b)** Correspondances trouvées avec le test du ratio et le test de symétrie.

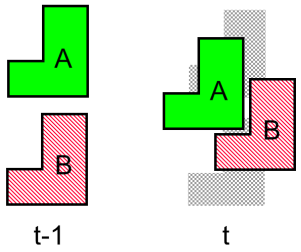


Figure 3.7 Résultat de la superposition de blobs concaves entre le temps t-1 et t. La partie grisée dans l'image à droite représente l'emplacement de A et B à t-1. On observe qu'il est possible de faire une mauvaise association avec ce type de méthode puisque les deux objets à t recouvrent chacun des objets à t-1.

3.4.3 États des objets

Pour une nouvelle paire de blobs, un nouvel objet est créé dans l'état *hypothèse* (transition a_0, association $0 \rightarrow 1$). Il est important d'avoir un état pour les nouveaux objets, car ceux-ci sont souvent incomplets ou fragmentés lors de leur entrée dans la scène (voir la figure 3.11 qui montre un objet concave entrant dans la scène) et il faut donc les traiter différemment des autres objets qui sont présents depuis plus longtemps dans la scène. Si nous ne trouvons pas d'associations pour un objet dans l'état hypothèse, il est directement supprimé (transition a_2, association $1 \rightarrow 0$). Cela permet de gérer les objets créés dus à des bruits parasitaires, changement de luminosité, ombre, vent sur les feuilles, etc. en définissant un nombre de trames minimum pour qu'un objet soit considéré comme étant objet réel et complet plutôt qu'un faux positif résultant du bruit. Si l'objet est suivi pour au moins 3 trames (paramètre «Nombre de trames hypothèse (N_h)»), alors il passe à l'état *normal* (transition a_1, association $1 \rightarrow 1$).

L'état *normal* est un état de suivi où l'objet est considéré comme étant seul, de taille complète et son suivi est stable. Si un objet dans l'état normal disparait de la scène (transition a_5, association $1 \rightarrow 0$), son état est changé à l'état *perdu*.

L'état *perdu* est un état utilisé pour accueillir les objets qui ont disparu en dehors des zones de sortie de la scène (la bordure). Cet état existe afin de nous permettre de réidentifier un objet perdu à cause d'une occultation complète avec un objet statique ou simplement parce qu'il s'est immobilisé temporairement. Si l'objet demeure dans l'état perdu pour un grand nombre de trames (à décider en fonction de la durée d'un feu de circulation par exemple), il est supprimé de la liste des objets suivis (transition a_{12}). Si le modèle de l'objet dans l'état perdu correspond au modèle d'un nouvel objet, alors ceux-ci sont fusionnés et retournés à l'état *normal* (transition a_{11}).

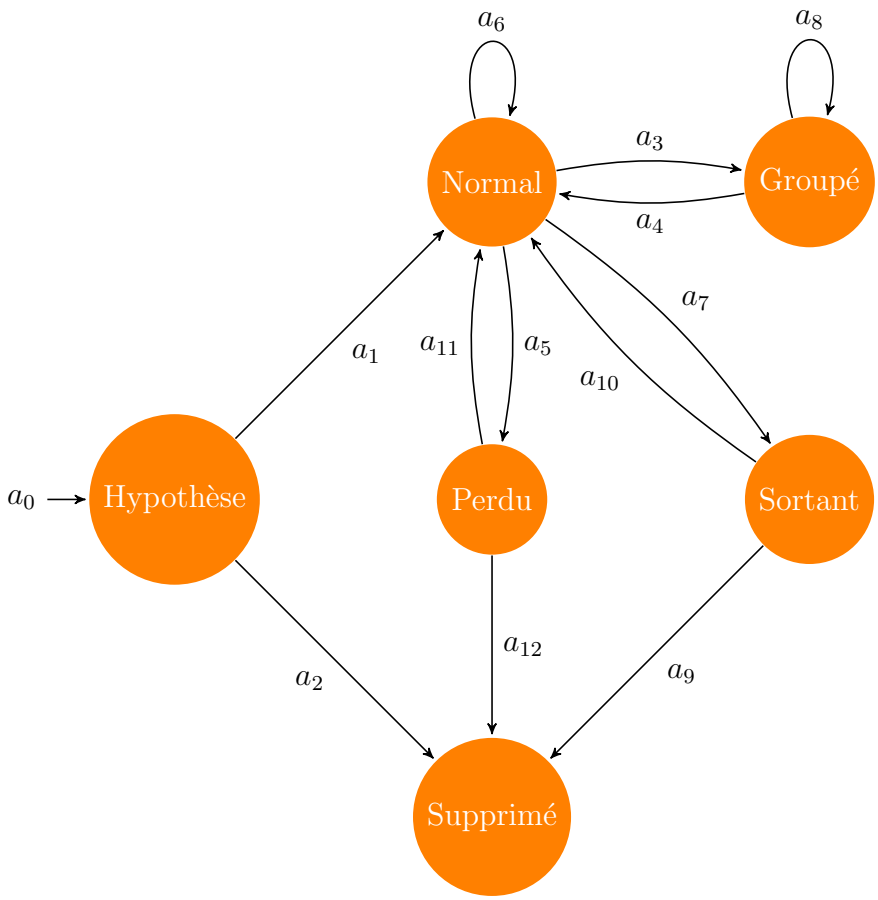


Figure 3.8 Machine à état d'un objet

Si plusieurs blobs associés à des objets existants se trouvent en état occultation mutuelle (transition a_3, association $N \rightarrow 1$), un groupe d'objets est créé avec les modèles de chacun des objets et chaque objet sera suivi individuellement dans le blob commun (plus de détail à la section 3.4.4). Le groupe d'objets sera dans l'état *groupé*. Cet état permet à l'algorithme

de mettre à jour les modèles des objets distincts à l'intérieur du groupe et d'évaluer leur taille à l'aide des points caractéristiques et des observations précédentes (voir section 3.4.4 plutôt qu'en utilisant la taille du blob. Si le groupe d'objet se sépare (association $1 \rightarrow N$), alors les objets séparés du groupe peuvent passer à l'état *normal* (transition a_4) ou rester à l'intérieur du groupe (transition a_8). S'il y a une association de blob $1 \rightarrow N$ pour un objet dans l'état normal, il est possible que ce soit un cas de sous-segmentation (plusieurs objets étant entré dans le même blob par exemple) ou bien un cas de sursegmentation. Si les deux blobs sont proches, un objet dans l'état hypothèse est créé avec le blob le moins similaire à l'objet. Sinon, ceux-ci sont séparés et on crée des objets dans l'état normal (transition a_6). On utilise ensuite les points des nouveaux objets pour reconstruire l'historique des objets individuels à partir de l'ancien.

Si un objet touche à la bordure de la scène, il change à l'état *sortant* (transition a_7). Cet état permet d'éviter de mettre un objet sortant de la scène dans la liste des objets perdus et de limiter les problèmes rencontrés lorsqu'un objet commençant à sortir de la scène entre en occultation avec un autre objet entrant dans la scène. Un objet dans l'état sortant qui n'a pas d'association avec des blobs plus récents passe automatiquement à l'état *supprimé* (transition a_9). Cet état permet d'indiquer qu'un objet doit être supprimé du système, car il n'est plus présent. Selon son état précédent et son modèle, celui-ci sera sauvegardé avant la suppression ou non. Si l'objet *sortant* ne touche plus à la bordure, une vérification de son modèle est effectuée afin de vérifier qu'il s'agit du même objet (tel qu'expliqué à la section 3.4.6). S'il s'agit du même objet, alors son état passe à l'état normal (transition a_{10}). Sinon, l'objet est supprimé et un nouvel objet est créé. Pour toutes les associations de type $1 \rightarrow 1$, le modèle de l'objet est simplement mis à jour et l'état ne change pas.

3.4.4 Gestion de la sous-segmentation

Un blob est considéré comme étant sous-segmenté s'il représente plusieurs objets. Cela peut être dû à un problème de la soustraction d'arrière-plan causant les pixels d'arrière-plan situés entre les blobs de plusieurs objets à être faussement détectés comme faisant partie de l'avant-plan, par exemple à cause d'une ombre comme à la figure 3.9a.

Une autre cause de sous-segmentation est que plusieurs objets sont en occultation comme à la figure 3.9 b) ce qui a pour effet qu'ils partagent un blob à plusieurs. Si les blobs correspondant à plusieurs objets distincts fusionnent à l'intérieur d'un même blob (transition a_3), ils seront ajoutés à un groupe d'objets et un modèle de suivi collectif sera créé. Ils seront également suivis individuellement en utilisant le modèle de chacun des objets bâti à partir des observations précédant la fusion. Ce processus est visualisable à la figure 3.10. Dans cette figure, on peut observer en a) le modèle avant l'occultation avec la boîte englobante du blob



a) b)

Figure 3.9 Exemple de sous-segmentation d'objets

associé et les points caractéristiques détectés. Dans la figure 3.10b) on observe les mêmes points situés aux mêmes endroits sur les objets, toutefois la boîte englobante est maintenant partagée entre les deux objets (dans un blob commun). Afin d'estimer la boîte englobante de chaque objet individuel dans la trame b), nous utilisons les points caractéristiques qui ont des correspondances avec ceux dans a). Pour ces correspondances, nous calculons la position relative du point au centroïde dans a). Cette position est ensuite soustraite de la position du point associé au blob dans b) afin d'obtenir un centroïde estimé. La largeur et la hauteur de chaque boîte sont également conservées. L'opération est recommencée pour chaque point dans b) ayant une correspondance dans un blob qui n'est pas en état d'occultation et on bâtit ainsi une liste de centroïde, de hauteur et de largeur. Une fois ces informations obtenues, nous prenons la valeur médiane de chaque liste que nous utilisons comme estimation. La valeur médiane est choisie afin d'éliminer le bruit pouvant se retrouver dans ces valeurs s'il y a eu une mauvaise observation. Le résultat obtenu est visible à la figure 3.10c). Un minimum de N_m correspondances de points est requis afin de faire cette estimation pour d'éviter de faire des estimations basées uniquement sur de mauvaises correspondances. S'il n'y a pas de correspondance pour un blob, mais qu'il y a une observation avant et après, une interpolation linéaire sera réalisée. Les boîtes englobantes estimées par les deux techniques précédentes sont ensuite comparées avec la boîte englobante du groupe et seule l'intersection des deux boîtes est gardée afin d'améliorer la qualité de l'estimation. Si le blob contenant le groupe d'objets se sépare, les objets qui le composent seront séparés à nouveau et ceux-ci seront associés aux nouveaux blobs à l'aide des points.

Il est possible que plusieurs objets pénètrent dans la scène dans le même blob. Si ceux-ci restent dans un blob commun pour l'ensemble du suivi, il ne sera pas possible de les séparer puisque nous manquerons d'informations. Ils seront donc suivis comme un seul objet.

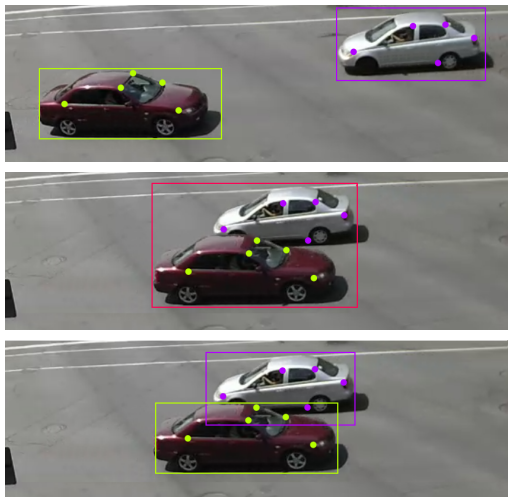


Figure 3.10 Processus d'estimation des blobs : a) Position précédente des boîtes englobantes avec les points b) Blob sous-segmenté avec les points c) Boîte englobante estimée

Toutefois, si ceux-ci se séparent alors qu'ils sont visibles dans la scène, de nouveaux objets seront créés (transitions a_6, a_8) et nous utiliserons le modèle de chacun des blobs afin de reconstituer l'historique du mouvement des objets individuel. Pour ce faire, nous utilisons la même technique que précédemment, toutefois nous utilisons comme modèle les blobs des objets une fois séparés et nous estimons les trames précédentes plutôt que la trame courante. Cette technique fonctionne bien pourvu que l'angle de vue de la caméra ne change pas trop et qu'il y a des points communs visibles avant et après la séparation.

3.4.5 Gestion de la sursegmentation

Il y a sursegmentation d'un objet lorsque plusieurs blobs représentent un même objet. Cela peut être dû à 1) une mauvaise soustraction d'arrière-plan (voir la figure 3.11a)), 2) un objet concave entrant la scène (voir la figure 3.11b)) ou encore un objet statique (intégré à l'arrière-plan) en occultation avec des objets en mouvement dans la scène comme un poteau situé plus près de la caméra que les objets d'intérêt (voir la figure 3.11c)). Pour le résoudre le cas 1) et 2), nous utilisons l'état Hypothèse afin d'accumuler suffisamment d'information avant de créer un objet permanent (dans l'état normal via la transition a_1). Ceci laisse le temps à la soustraction d'arrière-plan de se stabiliser. De plus, les objets dans l'état Hypothèse sont automatiquement fusionnés avec les objets à proximité (répondant au critère D_b présenté plus bas) ce qui évite la création d'objets séparés pour chaque sous-partie. Pour ce qui est la gestion plus spécifique du cas 2), les objets restent dans l'état Hypothèse tant et aussi longtemps qu'ils touchent à la bordure de la scène ce qui nous permet de gérer les cas de sursegmentation à l'entrée (voir figure 3.11). Pour le cas 3), la gestion de l'occlusion occasionnée par des objets

statiques, ceux-ci sont détectés comme étant une séparation de blob par la phase de suivi des blobs. Afin de gérer le cas pour les objets statiques de faible taille par rapport à l'objet suivi, nous ne séparons l'objet que si les blobs qui y sont associés sont suffisamment séparés. Pour vérifier cette condition, nous dilatons les blobs d'un facteur «Facteur de dilation des blobs (D_b)» et nous vérifions si les blobs sont en superposition. Nous utilisons 10 % par défaut pour ce paramètre. S'il y a une superposition entre les blobs, alors la séparation de l'objet est retardée puisque les objets sont trop proches pour juger qu'il s'agit d'objets distincts. Puisque le critère de séparation des blobs d'un objet est déterminé par la taille des objets, cela nous permet de gérer des objets de tailles différentes ce qui est très important pour la gestion de scènes urbaines.



a) b) c)

Figure 3.11 Exemple de sursegmentation : a) Un piéton est séparé en 3 blobs distincts à cause d'une mauvaise soustraction d'arrière-plan. b) Un piéton entre dans la scène et son corps est divisé en trois blobs distincts. c) Un poteau cache la route et divise les objets en deux lors de leur passage.

3.4.6 Gestion des objets sortants

La gestion des objets sortants est problématique puisqu'il est possible qu'un objet sortant fusionne avec un objet entrant (voir figure 3.12) et ceci n'est pas facilement détecté puisque l'objet entrant apparait dans un objet existant plutôt que dans un nouveau blob. L'absence de points sur la bordure vient compliquer davantage la détection de cette situation puisqu'il n'est pas possible de faire de vérification sur le modèle. Puisque cette situation est très fréquente aux intersections urbaines (dû au déplacement des véhicules dans les deux directions simultanément), nous avons introduit l'état sortant. Aussitôt qu'un objet touche à la bordure de la scène, son état change à l'état sortant. À partir de ce moment, il y a 3 situations

possibles : 1) l'objet quitte la scène, 2) l'objet ne quitte pas la scène et se déplace à nouveau vers le centre de la scène, 3) l'objet quitte la scène et son blob a été fusionné avec un objet entrant. La situation 1) est gérée par le fait qu'un objet sortant est supprimé s'il n'est plus détecté. Pour gérer efficacement 2) et 3), il est très important d'être capable de les distinguer. Si un objet dans l'état sortant ne touche plus la bordure de la scène, nous vérifions son identité. Si l'observation possède N_m points caractéristiques ou plus en commun avec l'objet avant qu'il commence à toucher à la bordure, alors nous sommes dans la situation 2). Sinon, nous sommes dans la situation 3), un nouvel objet est créé avec les nouvelles observations et l'ancien objet passe à l'état supprimé. Puisqu'il faut attendre que l'objet ne touche plus à la bordure pour déterminer s'il s'agit d'un nouvel objet, le moment exact où le blob a changé d'identité n'est pas connu. Afin de trouver ce moment et de corriger les trajectoires antérieures en conséquence, il faut passer au travers de tous les blobs dans l'état sortant enregistré aux trames précédentes pour l'objet et trouver un blob indiquant une transition. Afin de trouver ce blob, nous cherchons le blob sortant pour lequel l'aire est minimale puisque lorsque l'objet sort, l'aire de son blob diminue et lorsqu'un blob entre son aire augmente. En trouvant la valeur minimale de l'aire, nous pouvons estimer l'instant de transition (celui-ci variera selon la taille des objets en présence). Dans les situations non urbaines, il n'est parfois pas souhaitable ou nécessaire de faire cette vérification. Elle peut donc être désactivée avec le paramètre «Vérification de l'identité des objets sortant/entrant (V_{oes})», mais elle fortement recommandée pour le suivi urbain.

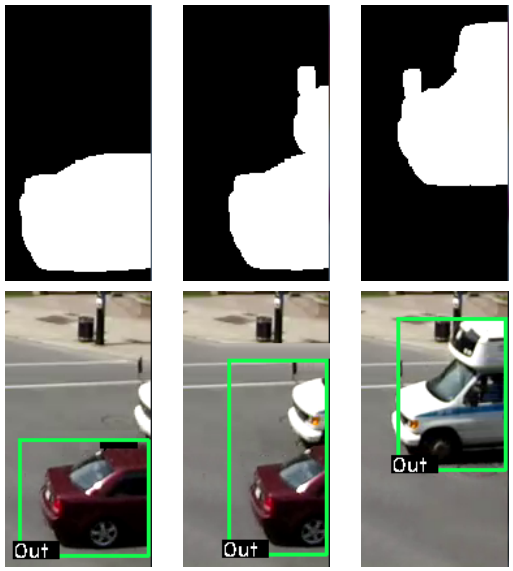


Figure 3.12 Évolution des blobs de la sortie et l'entrée simultanée d'objets

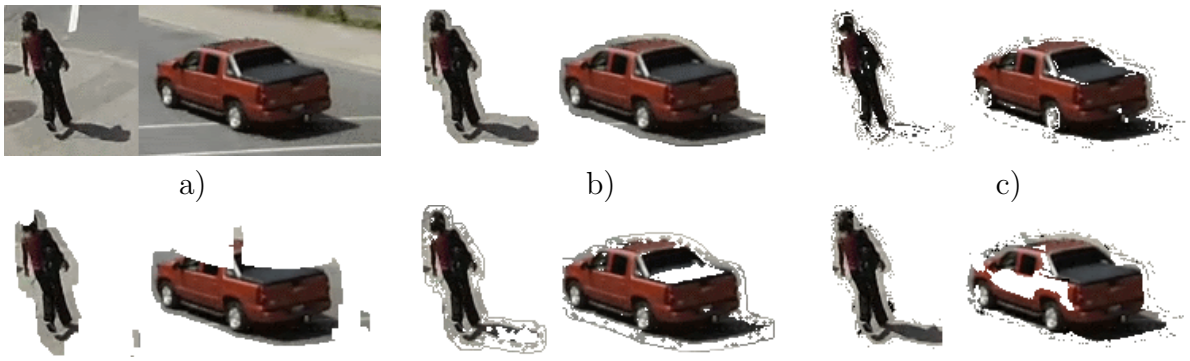
3.4.7 Gestion des objets perdus

Lorsqu'un objet s'immobilise, son état change à l'état *perdu* puisqu'il disparait de la soustraction d'arrière-plan. S'il se stationne, nous ne voulons plus le suivre puisqu'il sa trajectoire ne change plus. Toutefois, s'il s'arrête sur une courte période, nous voulons être en mesure de lui redonner la même identité. Les objets perdus sont donc conservés dans le système pour un certain nombre de trames «Nombre de trames de recherche (N_r)». Pour une scène urbaine, ce nombre de trames doit représenter un peu plus que la durée de la plus longue période d'attente possible à l'intersection. Il faut éviter de garder les objets plus longtemps pour éviter de trop ralentir le système puisque leur modèle est comparé avec tous les objets de la phase Hypothèse passants à l'état normal. Également, afin de limiter les opérations, nous limitons la zone de recherche à un rectangle 25% plus grand que la taille de la boîte englobant l'objet perdu à sa dernière observation. Cela permet d'améliorer les performances en limitant la zone de recherche. De plus, permettre l'association entre des objets trop éloignés augmenterait les risques d'erreur puisque les observations intermédiaires ne pourraient pas être simplement déduites.

3.4.8 Gestion des ombres projetées

Dans les scènes extérieures, les ombres ont un fort impact sur la soustraction d'arrière-plan. Elles peuvent déformer fortement les blobs d'objets déjà existants en plus de créer de nouveaux blobs de toute pièce. Plusieurs techniques existent pour détecter et supprimer les ombres d'une image, toutefois celles-ci ne sont pas toujours fiables et elles peuvent parfois classer les objets sombres comme étant des ombres. Des tests ont été effectués sur certaines images de notre ensemble de données à l'aide de techniques basées sur la géométrie, la texture et la couleur. Les résultats de ces tests effectués à l'aide de l'implémentation fournie par Sanin *et al.* (2012) sont présentés à la figure 3.13. On peut y voir que le résultat obtenu est intéressant pour la méthode de chromacité et dans une moindre mesure pour la méthode physique. Toutefois, de plus amples tests sur la méthode de chromacité ont démontré que la méthode génère de nombreux faux positifs ce qui créé de gros problèmes de segmentation tels que montrés à la figure 3.14. Il a donc été décidé de ne pas utiliser de technique de suppression des ombres pour éviter ces problèmes. Toutefois, les blobs contenant seulement une ombre comme les ombres d'oiseaux peuvent être détectés autrement. En effet, ceux-ci ne présentent que très peu de points caractéristiques. Il est donc possible de les éliminer en calculant le ratio entre le nombre total de points suivis et le nombre de trames pour lequel l'objet n'était pas sur la bordure. Pour les objets réels, il y aura un grand nombre de points suivi alors ce ratio sera élevé alors que pour les ombres projetées, il n'y aura que très peu de points. Il

est donc possible de définir un seuil en dessous duquel on considère que l'objet suivi est en réalité une ombre et on peut ainsi l'éliminer de la sortie de l'algorithme. Cette vérification est effectuée juste avant l'enregistrement ce qui veut dire que l'ombre sera suivie. Toutefois, l'analyse de l'ensemble de ces observations nous permettra de déduire qu'il n'est pas un objet d'importance. Puisque la bordure de l'image ne présente pas de points caractéristiques, les ombres s'étant déplacées pour l'ensemble du suivi sur la bordure ne seront toutefois pas éliminées par ce critère. La vérification des ombres n'est pas souhaitable si on désire suivre des objets bougeant peu et de couleur uniforme. Elle peut donc être désactivée avec le paramètre («Suppression des ombres d'oiseaux (S_o)») au besoin. Il est toutefois recommandé de l'activer pour les scènes urbaines.



a) b) c)

d) e) f)

Figure 3.13 Sortie des algorithmes de suppression des ombres : a) Fragment de l'image réelle étudiée. b) Résultat de la soustraction d'arrière-plan c) Méthode de chromacité d) Méthode géométrique e) Méthode physique f) Méthode à base de texture



a) b) c)

Figure 3.14 Erreur de la méthode chromatique : a) Véhicule suivi b) La partie frontale du véhicule a été entièrement détectée comme une ombre c) La partie centrale du véhicule a été détectée comme une ombre alors le véhicule a été séparé en deux parties

3.4.9 Mise à jour du modèle

La mise à jour du modèle consiste à ajouter à celui-ci les nouveaux points caractéristiques et les nouveaux blobs à chaque trame. Afin d'améliorer les performances, nous éliminons les points caractéristiques qui n'ont pas eu de correspondances après 3 trames. Nos tests nous démontrent que cette façon de procéder nous permet non seulement de réduire de beaucoup le nombre de comparaisons entre les points, mais en plus, les résultats sont légèrement meilleurs ainsi puisque qu'on réduit la probabilité qu'il y ait de mauvaises correspondances.

3.5 Paramètres de l'algorithme

Tableau 3.1 Paramètres de l'algorithme

Nom du paramètre	Valeur recommandée
Taille minimale des blobs (T_m)	50 px
Variance des pixels du capteur de la caméra (σ_{cam})	4
Variation interblob des pixels (V_p)	0.1
Nombre minimal de pairs de points (N_m)	4
Facteur de dilation des blobs (D_b)	0.1
Vérification de l'identité des objets sortant/entrant (V_{oes})	Oui
Nombre de trames de recherche (N_r)	1000
Suppression des ombres d'oiseaux (S_o)	Oui
Nombre de trames hypothèse (N_h)	4

Plusieurs paramètres ont été décrits dans les paragraphes précédents. Ceux-ci sont rapportés dans le tableau 3.1. Les paramètres les plus importants sont : 1) N_r, qui doit être ajusté en fonction de la durée des phases du feu de circulation, 2) T_m, qui doit être ajusté selon la taille minimale du plus petit objet de la scène. Ces paramètres peuvent être facilement décidés en fonction de la scène. Les autres paramètres décrits dans cette section ont également un impact qui est toutefois moins important que celui des deux paramètres précédents. Le système ne nécessite pas l'usage d'une homographie (elle n'est prise en compte à aucune des étapes de l'algorithme). Toutefois, si une homographie est fournie, celle-ci sera utilisée pour estimer la vitesse des véhicules et elle sera affichée lors du traitement.

Le prochain chapitre présentera les résultats obtenus par la méthode sur plusieurs ensembles de données urbaines et la méthode sera comparée à une autre méthode conçue pour le suivi en milieux urbains.

CHAPITRE 4

RÉSULTATS ET DISCUSSION

Ce chapitre présente les résultats obtenus à l'aide de notre méthode, Urban Tracker (UT). Il est divisé en trois sections. D'abord nous présenterons les métriques quantitatives intéressantes pour évaluer les algorithmes de suivi multiobjet. Ensuite, nous présenterons les outils développés afin de faciliter le calcul des métriques et la comparaison avec d'autres algorithmes existants. Finalement, nous présenterons les ensembles de données utilisés pour l'évaluation de la méthode ainsi que les résultats obtenus. Ces résultats seront comparés à ceux présentés dans Traffic Intelligence (TI) par Jackson *et al.* (2013) puisqu'il s'agit d'un algorithme créé pour les mêmes applications que nous et que celui-ci est disponible gratuitement en code libre.

4.1 Métriques

L'utilisation de métriques est cruciale afin d'évaluer les progrès de notre algorithme et de pouvoir comparer celui-ci de la façon la plus objective possible à d'autres algorithmes. Pour le suivi multiobjet, de nombreuses métriques sont utilisées et il n'y a pas encore de métrique qui a réussi à s'imposer comme standard d'évaluation des performances. Dans le cadre de cette recherche, deux ensembles de métriques ont été testés. D'abord, les métriques définies dans l'article de Yin *et al.* (2007) ont été testées puisqu'elles donnaient beaucoup d'informations (10 métriques) et que cette méthode était simple à implémenter. Nous avons toutefois découvert de grandes lacunes dans cette méthode, principalement au niveau de sa méthode d'association. En effet, avec cette méthode, une trajectoire est considérée comme ayant été correctement trouvée s'il y a 15% de la durée temporelle en superposition et qu'il y a suffisamment de superposition spatiale avec la vérité terrain (ground truth). Le problème est que chaque trajectoire est comparée individuellement avec la vérité terrain. Il n'y a donc pas de pénalités si un objet est fragmenté puisque chaque partie sera considérée comme ayant été suivie correctement. Il manque donc un mécanisme pour s'assurer que chaque trajectoire de l'algorithme ne peut être associée qu'à une seule trajectoire de la vérité terrain . Un autre problème avec cette méthode est que malgré le fait que le grand nombre de métriques permet d'avoir une vue détaillée de l'algorithme, ceux-ci viennent également complexifier la comparaison entre plusieurs algorithmes et il est difficile de savoir si l'algorithme s'améliore.

C'est pourquoi nous avons décidé d'utiliser CLEAR MOT de Bernardin et Stiefelhagen (2008) pour rapporter nos résultats.

Les métriques CLEAR MOT de Bernardin et Stiefelhagen (2008) furent conçues pour évaluer la précision et la performance du suivi multiobjet. La métrique peut gérer deux types de données, des barycentres et des boîtes englobantes. Il est préférable d'utiliser les boîtes englobantes pour l'évaluation puisqu'elles permettent d'évaluer la taille des objets également. Toutefois, puisque nous comparons notre méthode avec TI qui n'a pas comme sortie des boîtes englobantes, nous utiliserons la distance euclidienne entre les barycentres comme méthode d'association de données entre la vérité terrain et la sortie de notre algorithme. Afin d'associer les données pour une trame donnée, la méthode utilise l'algorithme hongrois de Munkres (1957) en utilisant la distance entre les sorties de l'algorithme et la vérité terrain comme poids pour optimiser les associations d'une trame donnée. Les associations ainsi formées sont suivies jusqu'à que le critère de distance maximale ne soit plus respecté. L'algorithme hongrois n'est évalué que sur les objets qui n'ont pas de correspondance dans la trame courante.

Les métriques calculées sur l'ensemble de la vidéo sont les suivantes :

Faux négatif (Misses ratio) $\overline{m}$ Pourcentage de détections manquées

Faux positif (False positive ratio) $\overline{fp}$ Pourcentage de détections qui correspondent à des faux positifs

Changement d'identité (Mismatches ratio) $\overline{mme}$ Pourcentage de détection qui correspondent à des changements d'identité

Multiple Object Tracking Accuracy (MOTA) Justesse du suivi, prends en compte $\overline{m}$, $\overline{fp}$, $\overline{mme}$.

Multiple Object Tracking Precision (MOTP) Distance moyenne entre le barycentre de la vérité terrain et la sortie de l'algorithme lorsqu'il y a correspondance.

Les deux métriques les plus importantes sont le MOTA et le MOTP puisque le MOTA prend en compte les trois autres métriques dans son calcul. Le MOTA est défini par l'équation 4.1, où g_t représente le nombre d'observations présentes dans la vérité terrain pour une trame t. m_t,fp_t et mme_t sont respectivement le m, le fp et le mme pour une trame t. La valeur maximale possible est de 1, toutefois le résultat peut également être inférieur à 0 s'il y a plus de faux positifs que de détections dans la vérité terrain. Pour ce qui est du MOTP, sa définition est donnée par l'équation 4.2 où d_t^i représente les distances entre les barycentres de la vérité terrain et ceux de l'algorithme à évaluer à un temps t pour un objet i.

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{4.1}$$

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t g_t} \tag{4.2}$$

Contrairement aux métriques de Yin *et al.* (2007), il n'est pas possible d'associer deux trajectoires d'un algorithme à une même vérité terrain ce qui donne des résultats plus cohérents. Les résultats sont également concis et simples à évaluer. Les inconvénients de cette méthode d'évaluation sont que l'optimisation par l'algorithme hongrois est effectuée par trame plutôt que pour l'ensemble des trames ce qui peut donner des résultats plus faibles que la réalité en cas de mauvaise association. Également, vu la méthode de calcul de MOTA, on peut remarquer que les changements d'identité sont très faiblement pénalisés ce qui peut être un problème pour certaines applications. Finalement, une autre lacune possible de cette méthode est qu'il faut définir un seuil maximal d'association au-dessus duquel les associations ne sont plus prises en compte. Il serait donc possible de trouver un seuil d'association plus avantageux pour une méthode qu'une autre et de ne rapporter que ce résultat. Afin de limiter cette influence, nous utiliserons la demi-longueur d'un objet typique situé au centre de chaque scène comme distance d'association maximum. Nous tracerons également les graphiques de MOTA et MOTP en fonction de la distance d'association pour chacune des vidéos de la section 4.3.7.

4.2 Outils développés

Quelques outils ont été développés afin de faciliter l'évaluation des métriques et le visionnement des résultats. Bien qu'il y ait beaucoup de recherche dans le domaine du suivi, il existe très peu d'ensembles de données annotées de scène urbaine. En effet, la majorité des chercheurs semble se concentrer sur le suivi de personnes, de visages ou d'une zone. Il y a peu d'ensembles de données disponibles pour le suivi multiobjet pour lesquels les objets sont de plusieurs types différents. Cela est d'autant plus vrai pour le suivi en milieux urbains. Des outils ont donc été développés afin de faire l'annotation de vérité terrain pour des vidéos et un nouveau format de données a été conçu pour stocker les résultats des algorithmes de suivi ainsi que les annotations. Les auteurs des métriques présentées à la section 4.1 ne fournissant pas d'implémentation, un outil d'évaluation des métriques a donc été conçu afin d'en faire l'évaluation.

4.2.1 Outil de génération de vérité terrain

L'outil de génération de vérité terrain (visible à la figure 4.1) est un ensemble d'outils pour manipuler les données de boîtes englobantes. Sa principale fonction est de permettre la création de données de suivi, leur enregistrement et leur chargement dans le format PolyTrack. Le format PolyTrack est une extension du format utilisé par TI afin de supporter

le suivi de boîte englobante. Sa spécification peut être trouvée à l'annexe A. L'outil permet également d'importer des données enregistrées dans de nombreux autres formats utilisés par des ensembles de données publique comme SQL, XML et texte. Parmi les formats gérés, on trouve entre autres ceux utilisés pour les conférences Performance Evaluation of Tracking and Surveillance (PETS 2001; 2009) ainsi que ceux de Context Aware Vision using Image-based Active Recognition (CAVIAR 2004) . Des annotations peuvent aussi être créées à partir de zéro ou encore en corrigeant la sortie d'un algorithme de suivi déjà existant. De plus, il est possible d'interpoler entre plusieurs annotations afin de réduire le temps nécessaire pour l'annotation de la vidéo. On peut également spécifier des noms, description et type pour chacun des objets considérés ce qui peut être utile lors de l'étude des interactions entre les différents types d'usager de la route ou encore pour évaluer le succès de méthodes de classification d'objets. L'outil permet d'ailleurs d'exporter les annotations afin de créer des ensembles de données d'entrainement pour des détecteurs d'objets. Il est également possible de l'utiliser afin d'exporter des vidéos de résultats avec certains paramètres pour l'affichage (boîte englobante et trajectoire de l'objet). Le manuel de l'outil d'annotation est disponible à l'annexe B.

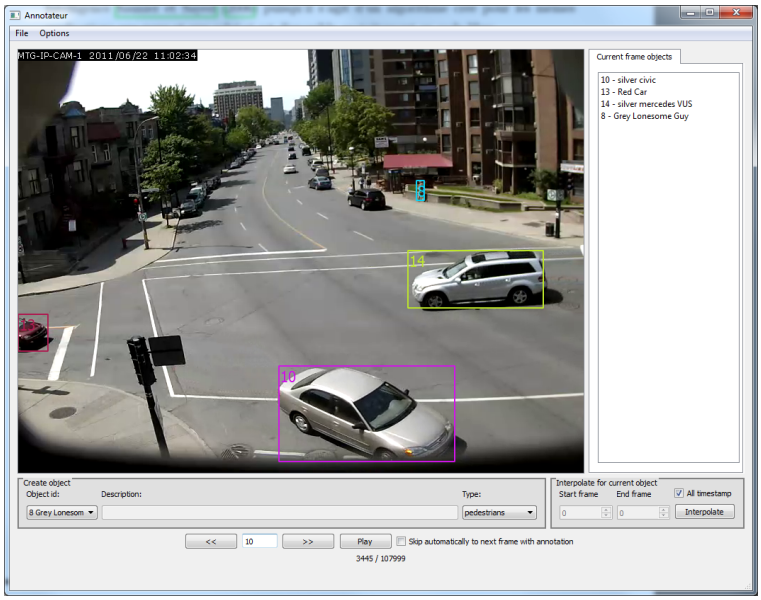


Figure 4.1 Interface de l'outil de génération de vérité terrain

4.2.2 Outil d'évaluation des métriques

L'outil d'évaluation des métriques est un outil en ligne de commande permettant d'évaluer les métriques définies à la section 4.1 en comparant les résultats d'un algorithme à la vérité

terrain. Les données doivent être dans le format PolyTrack. Le manuel de l'outil d'évaluation des métriques est disponible à l'annexe C.

4.3 Résultats



a) b) c) d) e)

Figure 4.2 Une trame de chaque ensemble de données utilisées : a) Sherbrooke b) Rouen c) St-Marc d) René-Lévesque e) Atrium

UT a été évalué sur 5 vidéos différentes qui ont été annotées à l'aide de l'outil d'annotation. Afin de nous positionner par rapport à l'état de l'art, nous comparons les résultats obtenus avec ceux de TI. Puisque TI est un projet ouvert en constant développement, il a fallu choisir une révision spécifique de l'algorithme à utiliser afin que les résultats soit reproductibles. La révision 8f8f437 [1] du 3 septembre 2013 a été choisie afin de se comparer à une version récente et stable de l'algorithme. Cette version de l'algorithme diffère peu de celui présenté dans Saunier et Sayed (2006). Quatre des séquences vidéo choisies pour l'évaluation sont de nature urbaine et prises à partir de différents points de vue. Une autre des séquences est composée de piétons s'entrecroisant à l'intérieur dans l'atrium du pavillon Mackay-Lassonde de l'École Polytechnique de Montréal. Cette séquence a été choisie afin de comparer la performance des algorithmes sur une séquence intérieure constituée d'objets de tailles similaires (des piétons). Une image de chacune des vidéos traitées est visible à la figure 4.2.

Les vidéos ayant été prises en des lieux différents avec de l'équipement différent, la taille des objets varie fortement. Il faut donc ajuster le seuil d'association de CLEAR MOT pour chaque ensemble de données. Pour ce faire, nous avons décidé de prendre le côté le plus long d'un objet typique de chaque scène lorsqu'il est au centre de l'écran et d'utiliser la moitié de cette longueur comme référence. Les points de référence utilisés sont illustrés à la figure 4.3 et les longueurs obtenues sont visibles au tableau 4.1.

Tableau 4.1 Longueur maximum d'association par vidéo pour la métrique CLEAR MOT

Sherbrooke	Rouen	St-Marc	René-Lévesque	Atrium
90 px	164 px	113 px	24 px	92 px

1. Disponible au https ://bitbucket.org/Nicolas/trafficintelligence/commits/8f8f437



a) b) c) d) e)

Figure 4.3 Objet de référence pour chaque scène : a) Sherbrooke b) Rouen c) St-Marc d) René-Lévesque e) Atrium

4.3.1 Paramètres utilisés

UT et TI utilisent tous deux des paramètres spécifiques à chaque scène afin de s'adapter à la scène. Ceux-ci ont été rapportés dans le tableau 4.2 pour UT et dans le tableau 4.3 pour TI. Les autres paramètres qui ne sont pas rapportés correspondent aux paramètres par défaut des algorithmes (tel que décrit dans la section 3.5 pour UT). Pour UT, le mode urbain consiste à activer V_{oes} et S_o, c'est-à-dire la vérification de l'identité des objets entrant/sortant et la suppression des ombres d'oiseaux. Ceux-ci n'ont pas été activés pour la séquence Atrium puisqu'il n'y a pas ce type de problème d'ombres à l'intérieur et dans cette scène il y a beaucoup de piétons qui vont sur les bordures et font des rotations sur eux-mêmes avant d'entrer. Notre critère de vérification étant assez sévère, un objet ayant une nouvelle identité était alors créé. Pour TI, l'homographie permet de ramener tous les objets sur un plan de façon à ce que la distance entre les points ne change pas à cause de la perspective. Certaines vidéos ont donc été traitées avec une homographie lorsque le résultat obtenu était meilleur. L'avantage de l'utilisation de l'homographie est qu'elle permet de définir des paramètres en mètres plutôt qu'en pixels.

Tableau 4.2 Paramètres utilisés pour chaque vidéo pour UT

Paramètres	Sherbrooke	Rouen	St-Marc	René-Lévesque	Atrium
N_r	1160	150	1160	900	150
D_b	0.1	0.7	0.1	0.2	0.4
T_m	300	380	300	50	280
Mode urbain (S_o et V_{oes})	oui	oui	oui	oui	non

4.3.2 Sherbrooke

La séquence Sherbrooke est une vidéo de l'intersection Sherbrooke et Amherst à Montréal filmée à partir d'un poteau à quelques mètres au-dessus du sol par Jean-Simon Bourdeau et Marilyne Brosseau. La résolution de la vidéo est de 800x600. Puisque la caméra n'est pas

Tableau 4.3 Paramètres utilisés pour chaque vidéo pour TI

Paramètres	Sherbrooke	Rouen	St-Marc	René-Lévesque	Atrium
Distance de connexion	4 m	25 px	10 px	10 px	120 px
Distance de segmentation	1.7 m	25 px	40 px	40 px	100 px
Nombre de points	1000	1000	1000	2000	1000
Homographie	oui	non	non	non	non

placée très haut, les occultations sont nombreuses entre les usagers de la route circulant dans l'intersection. Pour l'évaluation, une sous-séquence de 1001 trames à 30 trames/seconde avec 15 véhicules et 5 piétons a été annotée. La scène contient jusqu'à 7 objets simultanément. Puisque dans l'extrait traité deux piétons se promènent ensemble et qu'ils ne se séparent pas pour l'ensemble de la séquence, nous avons rapporté les résultats en les considérant individuellement et en groupe. En effet, pour les études de sécurité le nombre d'usagers dans un groupe est moins important que de bien détecter le groupe puisqu'il n'y a pas d'interactions dangereuses à l'intérieur du groupe. D'autres applications comme les comptages nécessitent toutefois un nombre exact de piétons. Afin de limiter le suivi à l'intersection, nous avons appliqué le masque visible à la figure 4.4. Les objets à l'extérieur de ce masque ne sont donc pas pris en compte.

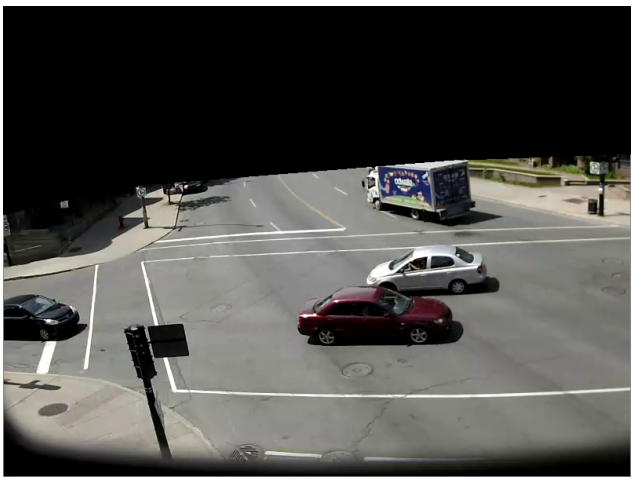


Figure 4.4 Masque appliqué à la vidéo Sherbrooke

Les résultats sont rapportés par type d'objets afin de montrer les performances obtenues pour chaque type d'objet dans le tableau 4.4.

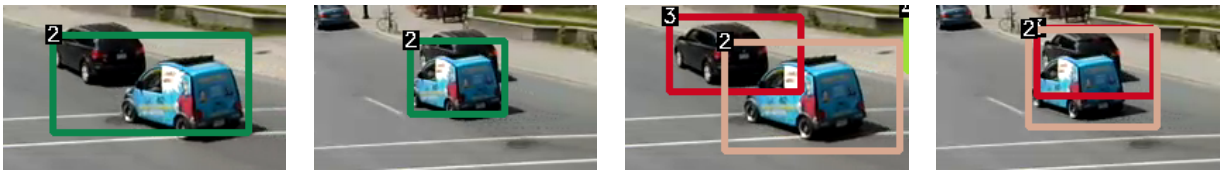
En observant les résultats pour les objets individuels, on remarque que notre méthode a un MOTA plus faible pour les piétons que pour les véhicules. Le tableau 4.5 montre le résultat si on groupe les piétons se déplaçant en groupe. Les résultats de ce tableau nous montrent que le MOTA plus faible pour les piétons est principalement dû au fait qu'un groupe de deux piétons est suivi comme étant un seul objet pour l'ensemble de la séquence

Tableau 4.4 Métriques CLEAR MOT obtenues pour la vidéo Sherbrooke en utilisant un seuil d'association de 90 pixels. Les résultats sont rapportés pour tous les types d'objets (véhicules et piétons) ensemble et séparément. Les meilleurs résultats sont rapportés en gras.

	Objets individuels	
Méthode	MOTA	MOTP
Véhicules		
TI (Jackson *et al.*, 2013)	0.8253	**7.42 px**
UT	**0.8928**	10.53 px
Piétons		
TI (Jackson *et al.*, 2013)	0.0141	11.98 px
UT	**0.6809**	**6.64 px**
Tous les objets		
TI (Jackson *et al.*, 2013)	0.3841	**7.54 px**
UT	**0.7771**	8.66 px

Tableau 4.5 Métriques CLEAR MOT obtenues pour la vidéo Sherbrooke en utilisant un seuil d'association de 90 pixels. Les résultats sont rapportés pour tous les types d'objets (véhicules et piétons) ensemble et séparément. Les piétons se déplaçant en groupe sont ici considérés comme étant un seul objet. Les meilleurs résultats sont rapportés en gras.

	Piétons groupés	
Méthode	MOTA	MOTP
Véhicules		
TI (Jackson *et al.*, 2013)	0.8253	**7.42 px**
UT	**0.8928**	10.53 px
Piétons		
TI (Jackson *et al.*, 2013)	0.0179	11.98 px
UT	**0.8764**	**6.78 px**
Tous les objets		
TI (Jackson *et al.*, 2013)	0.4328	**7.51 px**
UT	**0.8841**	8.72 px



a) b) c) d)

Figure 4.5 Problème de segmentation des objets (pour TI et même scène gérée par UT : a) et b) TI c) et d) UT

(voir figure 4.6). Cela n'est pas étonnant puisque nous n'utilisons pas de méthode de détection et que les blobs seuls ne nous permettent pas de segmenter les objets dans cette situation due à l'absence d'hypothèse liée à leur taille. En observant les résultats obtenus par TI,

on constate que l'algorithme n'a pas été en mesure de suivre les piétons et que le score de l'algorithme pour ceux-ci est quasiment nul. Cela s'explique par les limitations décrites dans la section 2.5.3, c'est-à-dire le fait que la méthode de groupement des points n'est pas adaptée au suivi d'objets de taille différente. Le score obtenu pour les véhicules par notre algorithme est également supérieur. Cela s'explique par le fait que TI n'a pas été en mesure de segmenter deux véhicules ayant pénétré la scène un à la suite de l'autre à des vitesses similaires. Du côté d'UT, puisque ceux-ci sont rentrés séparément, un modèle a été construit pour chacun des objets et ils ont pu être segmentés séparément. Cette situation est illustrée à la figure 4.5. On peut observer à la figure 4.5d) que même en présence d'une forte occultation partielle, UT a été capable de localiser l'objet grâce au mécanisme d'estimation de boîte englobante décrit à la section 3.4.4. Pour cette vidéo, les résultats obtenus par TI sont un peu plus précis (le MOTP est plus petit). Cela est en partie dû aux ombres qui déforment les objets ce qui a pour effet de déplacer le barycentre des objets pour UT.

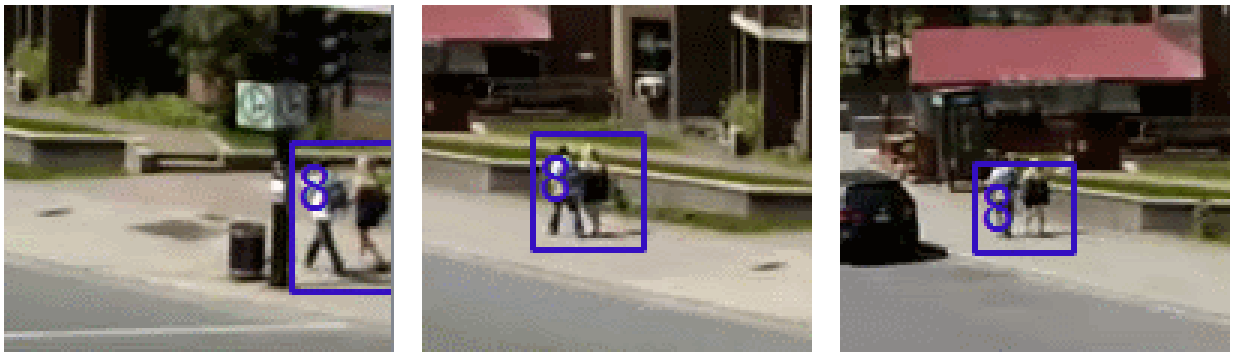


Figure 4.6 Piétons groupés pour l'ensemble de la séquence par UT

La figure 4.7 montre quelques extraits de la vidéo Sherbrooke avec le résultat obtenu par UT superposé sur la vidéo.

4.3.3 Rouen

La séquence vidéo Rouen que nous avons choisie contient 600 trames à 25 trames/seconde avec une résolution de 1024x576. La vidéo a été prise à partir d'une caméra située au 3e étage d'un immeuble par le laboratoire LEPSIS(IFSTTAR) Seize objets circulent dans la scène, c'est-à-dire 4 automobiles, un vélo et 11 piétons. La scène contient jusqu'à 8 objets simultanément. Puisque la majorité de l'activité se déroule sur une traverse piétonnière, il y a beaucoup d'occultations entre les piétons.

Le tableau 4.6 présente les résultats obtenus pour la vidéo Rouen. On observe que le MOTA et le MOTP sont meilleurs pour UT. Cela s'explique encore une fois par le fait qu'il s'agit d'un trafic mixte. Pour cette scène, les seuils de TI ont été optimisés afin de mieux

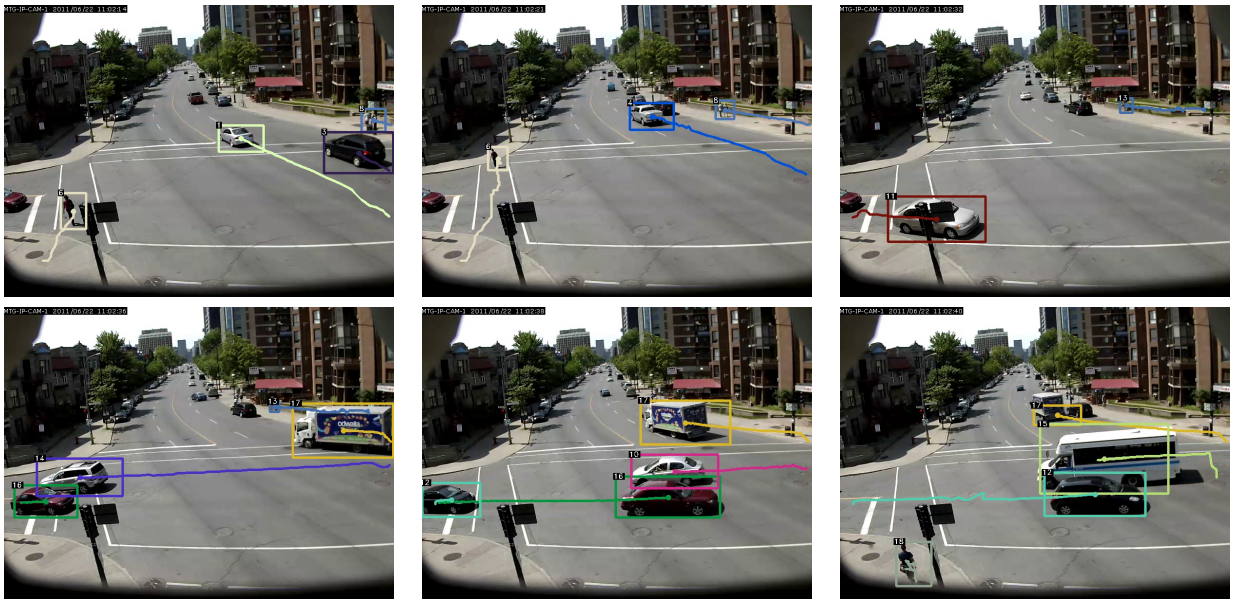


Figure 4.7 Exemple de résultats de UT pour la séquence Sherbrooke.

Tableau 4.6 Métrique CLEAR MOT pour la vidéo Rouen calculée avec une distance maximale d'association de 164 px. Les meilleurs résultats sont rapportés en gras.

	Objets individuels	
Méthode	MOTA	MOTP
Véhicules		
TI (Jackson *et al.*, 2013)	0.1853	66.69 px
UT	**0.8965**	**9.73 px**
Bicyclette		
TI (Jackson *et al.*, 2013)	0.8686	**13.11 px**
UT	**0.9270**	14.13 px
Piétons		
TI (Jackson *et al.*, 2013)	0.6467	20.04 px
UT	**0.8050**	**13.64 px**
Tous les objets		
TI (Jackson *et al.*, 2013)	0.5885	24.20 px
UT	**0.8234**	**13.09 px**

suivre les piétons (puisque ceux-ci sont plus nombreux), mais cela se fait au détriment des véhicules qui eux sont fragmentés en plusieurs objets ou pas du tout suivis. Pour ce qui est des piétons, plusieurs sont suivis correctement, toutefois certains piétons sont groupés entre eux. Pour ce qui est de UT, il y a quelques problèmes avec les piétons. En effet, certains piétons sont également groupés pour une partie du suivi (voir 4.8), toutefois cela affecte beaucoup moins d'observations que TI et les véhicules sont correctement suivis. Cela explique donc la différence au niveau du MOTA entre les méthodes. Pour ce qui est de la différence de score

du MOTP, elle peut facilement être expliquée par le fait que pour TI, la boîte englobante suivie peut être située sur un phare avant ou seulement une mince sous-partie de l'objet. Pour les piétons, c'est plutôt au niveau du torse que va se situer la boîte englobante de l'objet, ce qui a pour effet de décaler le barycentre. Pour UT, la précision est plus faible que dans la vidéo Sherbrooke à cause des erreurs de sous-segmentation entre les piétons et également à cause de la taille relative plus grande des objets. La figure 4.9 montre une partie des résultats obtenus pour UT. On peut observer que les objets sont suivis indépendamment de leur taille ou de leur type.

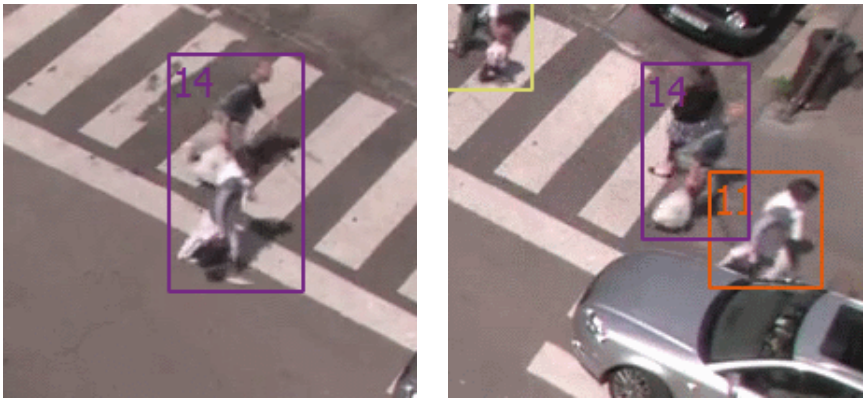


Figure 4.8 Problèmes de piétons groupés par UT



Figure 4.9 Exemple de résultats de UT pour la séquence Rouen

4.3.4 St-Marc

La séquence vidéo St-Marc est une séquence filmée en résolution 1280x720 à l'intersection Saint-Marc et Maisonneuve à Montréal. Elle a été fournie par Mohamed Gomaa Mohamed. Un extrait de 1000 trames à 29 images/seconde de cette vidéo a été annoté afin de faire

l'évaluation. La vidéo a été prise d'un point de vue similaire à celle de Rouen et on peut y observer 7 automobiles, 2 bicyclettes et 19 piétons. La scène contient jusqu'à 14 objets simultanément. Plusieurs des piétons se promènent en groupe pour l'ensemble de la séquence.

Tableau 4.7 Métrique CLEAR MOT pour la vidéo St-Marc calculée avec une distance maximale d'association de 113 px. Les meilleurs résultats sont rapportés en gras.

Méthode	MOTA	MOTP
Véhicules		
TI (Jackson *et al.*, 2013)	-0.1778	38.99 px
UT	**0.8887**	**10.9 px**
Bicyclette		
TI (Jackson *et al.*, 2013)	0.8952	7.46 px
UT	**0.9895**	**6.70 px**
Piétons		
TI (Jackson *et al.*, 2013)	0.6926	10.44 px
UT	**0.7216**	**5.97 px**
Tous les objets		
TI (Jackson *et al.*, 2013)	0.6018	14.58 px
UT	**0.7567**	**5.97 px**

Le tableau 4.7 présente les résultats obtenus sur cet ensemble de données. Les résultats de TI sont près des nôtres pour les piétons. Cela s'explique par le fait que les seuils de TI ont été ajustés pour les piétons ce qui lui permet de différencier une partie des piétons des groupes de piétons, toutefois cela a pour effet que les véhicules sont complètement fragmentés et les boîtes ne sont pas du tout alignées sur le barycentre de l'objet. La précision de TI est donc inférieure dans toutes les catégories et le MOTA des véhicules pour TI est négatif. Il est négatif à cause des faux positifs dus à la fragmentation et des faux négatifs qui sont présents en plus grande quantité que le nombre d'observations dans la vérité terrain. D'ailleurs, le seuil utilisé pour TI ici est très important puisque si on change le seuil de connexions de 10 pixels à 15 pixels, les piétons se groupent et on se retrouve alors avec un MOTA 20% inférieur. Pour UT, le groupement des piétons explique le MOTA plus faible que dans les autres ensembles de données. Les véhicules, les cyclistes et les piétons isolés sont correctement suivis. On observe toutefois quelques problèmes de segmentation et un changement d'identité dans un groupe de 4 piétons (voir figure 4.10). La figure 4.11 présente des résultats obtenus pour cette séquence par UT. On peut y voir certains des problèmes de groupement mentionnés plus haut.

4.3.5 René-Lévesque

La vidéo René-Lévesque a été filmée en 1280x720 à partir d'une tour du centre-ville de Montréal à plusieurs étages de hauteur. Elle a été fournie par Mohamed Gomaa Mohamed.

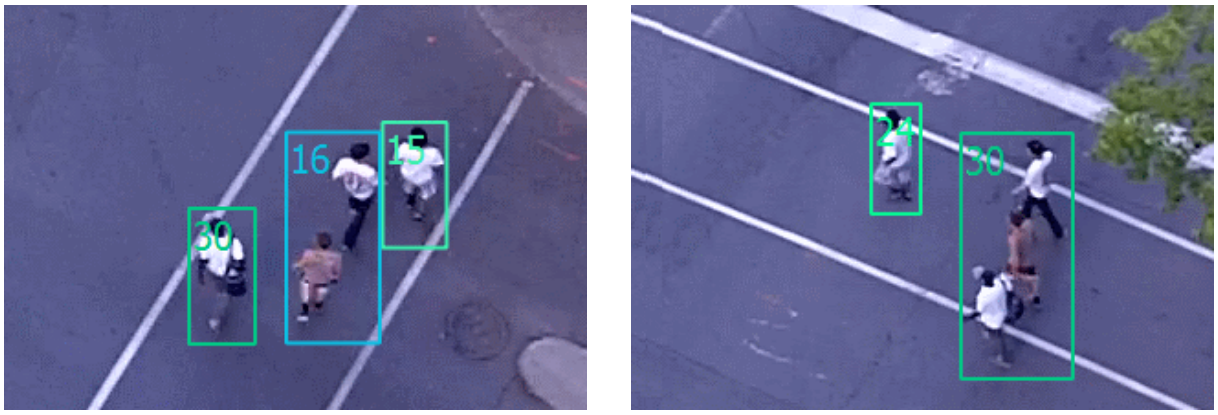


Figure 4.10 Problèmes de piétons groupés par UT



Figure 4.11 Exemple de résultats de UT pour la séquence St-Marc.

Elle offre une vue sur le boulevard René-Lévesque et elle couvre trois intersections. Nous avons décidé d'annoter les véhicules de deux intersections pour 1000 trames à 29 trames/seconde. La troisièmement intersection a été ignorée puisqu'elle est beaucoup trop éloignée. Les piétons n'ont pas été annotés puisque ceux-ci sont très petits et il aurait été difficile de les annoter avec notre outil puisqu'ils ne couvrent que quelques pixels. Le test se concentre donc uniquement sur les véhicules et les cyclistes circulant dans la scène dans la zone délimitée par le masque visible à la figure 4.12. Vue la hauteur à laquelle la séquence a été capturée, il n'y a que de faibles occultations, toutefois il y a beaucoup de véhicules se déplaçant simultanément à l'écran et certains objets sont très petits. La scène contient 29 automobiles et 2 cyclistes. La scène contient jusqu'à 20 objets simultanément.

Pour cette scène, les résultats sont rapportés au tableau 4.8. Le MOTA et le MOTP d'UT sont meilleurs que ceux de TI par une bonne marge malgré le fait que nous avons considéré seulement les véhicules et les vélos. Cela peut s'expliquer par le fait que les véhicules sont beaucoup plus loin, il y a donc beaucoup moins de points sur ceux-ci. C'est donc la raison

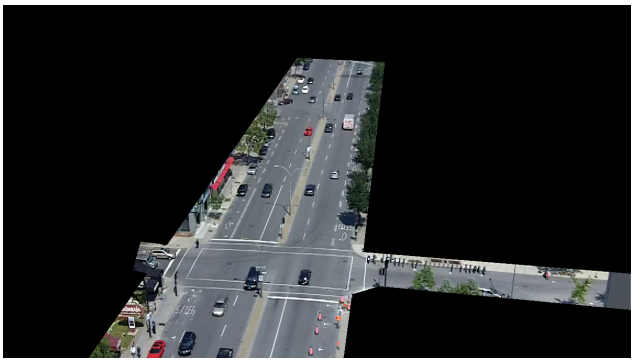


Figure 4.12 Masque appliqué à la scène René-Lévesque

pour laquelle nous avons mis des paramètres pour TI qui génèrent plus de points pour cette scène. La différence de précision est due à un problème de fragmentation pour TI qui a pour effet que la boîte ne couvre qu'une partie du véhicule qui n'est pas centrée. Un des problèmes pour les deux algorithmes est qu'il y a un vélo très petit qui est peu ou pas suivi pour une bonne partie de la scène. Sinon, il y a certains problèmes de fragmentation, principalement sur les objets lointains qui se divisent lorsque les objets s'éloignent les uns des autres (voir figure 4.13). La figure 4.14 montre quelques résultats obtenus pour UT.

Tableau 4.8 Métrique CLEAR MOT pour la vidéo René-Lévesque calculée avec une distance maximale d'association de 24 px. Les meilleurs résultats sont rapportés en gras.

Méthode	MOTA	MOTP
Véhicules		
TI (Jackson *et al.*, 2013)	0.5474	5.23 px
UT	**0.8038**	**3.02 px**
Bicyclette		
TI (Jackson *et al.*, 2013)	0.2321	3.14 px
UT	**0.2509**	**2.18 px**
Tous les objets		
TI (Jackson *et al.*, 2013)	0.5029	5.10 px
UT	**0.7267**	**2.97 px**

4.3.6 Atrium

La séquence Atrium est une séquence vidéo de 4540 trames à 30 trames/seconde ayant été prise à l'École Polytechnique de Montréal par le laboratoire LITIV. Cette séquence se déroule à l'intérieur et on peut y voir plusieurs individus se déplacer, déposer des objets et s'entrecroiser. Cette scène est intéressante puisqu'elle permet d'évaluer la performance de l'algorithme sur des piétons en plus de tester sa résistance aux occultations sur 3 objets simultanément.



a) b) c)

Figure 4.13 Problèmes de véhicule lointain groupé par UT. On peut voir en c) que le groupe (visible en a) et b)) se brise lorsque le véhicule s'éloigne suffisamment.



Figure 4.14 Exemple de résultats de UT pour la séquence René-Lévesque

Tableau 4.9 Métrique CLEAR MOT pour la vidéo Atrium calculé avec une distance maximale d'association de 92 px. Les meilleurs résultats sont rapportés en gras.

Méthode	MOTA	MOTP
Piétons		
TI (Jackson *et al.*, 2013)	0.6903	49.35 px
UT	**0.9359**	**7.93 px**

Les résultats obtenus pour cette séquence sont présentés dans le tableau 4.9. On observe que le MOTA obtenu pour cette scène est plus élevé que pour les scènes urbaines. Cela s'explique par le fait que la scène est moins complexe vu que la taille des objets ne varie pas et

qu'il y a un maximum de 4 objets dans la scène simultanément. Il y a toutefois des interactions complexes avec 3 objets simultanément et ceux-ci sont bien gérés par UT. La précision de UT est également supérieure à TI. Cela s'explique par le fait qu'afin d'augmenter le MOTA pour TI, nous avons utilisé des seuils de segmentation plus restrictifs pour le groupement de points caractéristiques. Les points sur des parties non rigides du corps ne sont pas groupés ce qui a pour effet que les groupes sont souvent au niveau du visage ou du haut du corps. Cela peut donc décaler le barycentre de 40 à 50 pixels. Le positionnement des points lors du groupement sur cette scène a été montré précédemment à la figure 2.10. La précision d'UT aurait toutefois pu être meilleure si la méthode soustraction d'arrière-plan était moins sensible aux réflexions (voir figure 4.15).

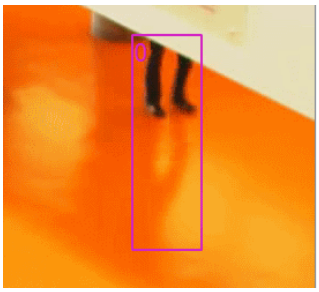


Figure 4.15 La partie sous l'escalier de l'atrium est très réflective. Cela déforme une partie des blobs pour UT.

Quelques exemples de suivi sont donnés à la figure 4.16.

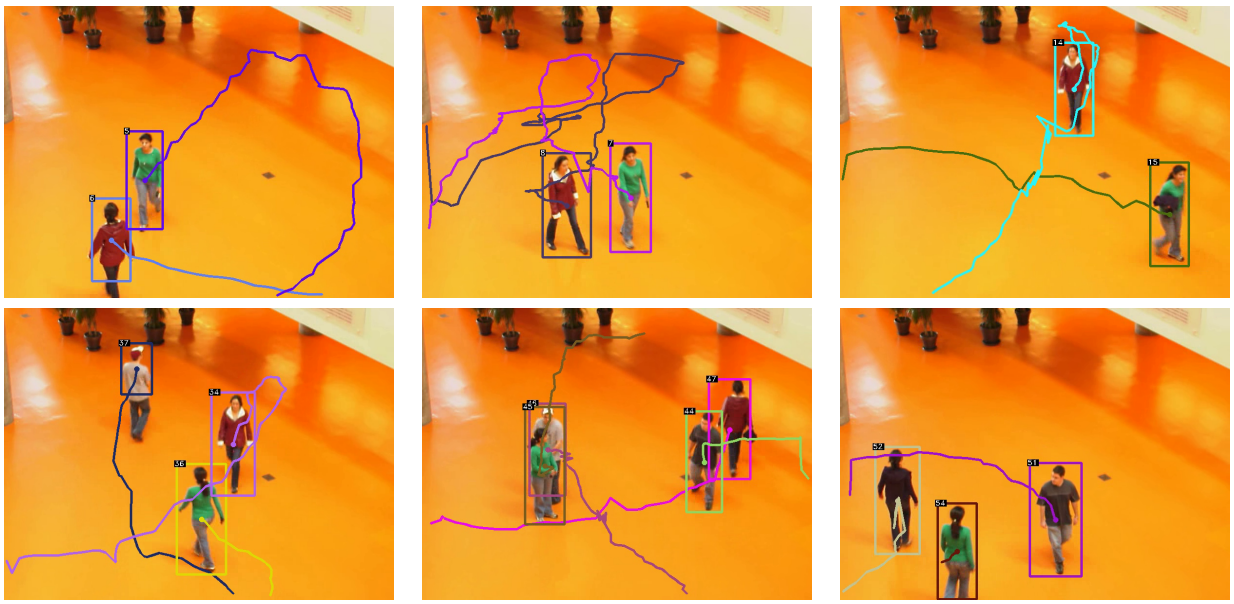


Figure 4.16 Exemple de résultats obtenu à l'aide d'UT pour la vidéo Atrium

4.3.7 Impact du seuil d'association de CLEAR MOT

Puisqu'il faut définir un seuil d'association lors du calcul des métriques CLEAR MOT, il est possible que le seuil utilisé biaise les résultats, principalement pour la comparaison entre deux méthodes pour lesquels les résultats obtenus sont similaires. Nous avons tracé une courbe pour chacun des ensembles de données avec le MOTA et le MOTP en fonction du seuil utilisé (figure 4.17). Pour le MOTA, la courbe la plus élevée représente une meilleure justesse alors que pour le MOTP, la courbe la plus basse correspond à l'algorithme le plus précis. On peut voir que la courbe MOTA d'UT est au-dessus de celle TI pour tous les seuils utilisés sauf dans le cas d'une distance d'association de moins de 9 pixels pour Sherbrooke. Cela s'explique par le fait que la précision de l'algorithme est inférieure à TI pour cette scène. Comme la précision moyenne de UT est près de 9 pixels dans cette scène, l'usage d'un seuil d'association inférieur à 9 pixels pénalise donc l'algorithme en double puisque pour chacune des observations il y aura un faux positif et un faux négatif. Cette caractéristique explique également pourquoi il est possible d'avoir des MOTA négatifs. Également pour Sherbrooke, on voit que la précision est légèrement inférieure à TI ce qui est représentatif de ce qui a été rapporté à la section 4.3.2.

Les graphiques nous permettent de constater que les résultats obtenus pour UT arrivent à un plateau plus rapidement que TI et que l'utilisation d'un seuil d'association plus élevé a peu d'impact. Cela est dû au fait que le système génère très peu de faux positif et des données plus précises que TI. L'usage d'un seuil plus élevé ne permet donc pas d'associer de mauvaises observations (ou des observations moins précises) avec les observations de la vérité terrain. Les courbes démontrent clairement que les seuils d'associations utilisés précédemment ne biaisent pas le portrait général des résultats que nous avons montré précédemment.

4.3.8 Limitations de l'algorithme

L'algorithme UT comporte certaines limitations connues. La plupart de ces contraintes sont dues à l'utilisation d'une soustraction d'arrière-plan. Tout d'abord, la soustraction d'arrière-plan peut seulement être utilisée avec des caméras statiques, et celles-ci doivent être le plus stables possible. Les changements brusques de luminosité peuvent également être problématiques, bien que puisque le modèle de l'arrière-plan est adaptatif, les changements plus graduels de luminosité sont correctement gérés. Les ombres des objets ne sont présentement pas éliminées faute d'avoir trouvé une méthode fiable. Puisque le critère de groupement est basé sur l'utilisation de blobs, il faut que les objets soient suffisamment séparés pour que leurs blobs correspondants ne se chevauchent pas pendant une partie du suivi pour que ceux-ci se constituent un bon modèle de l'objet. Cela n'est pas toujours le cas pour les piétons, tel

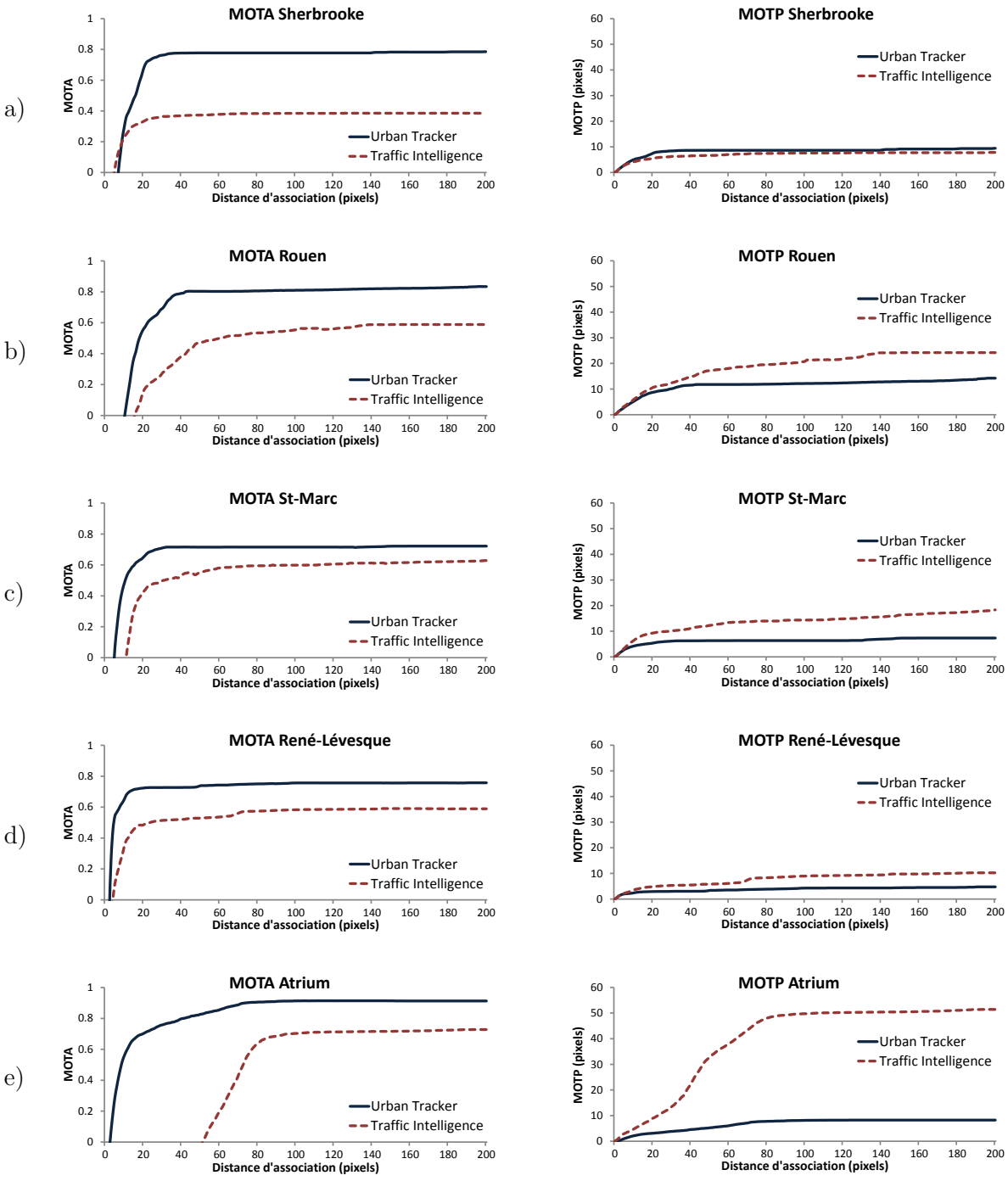


Figure 4.17 Impact du seuil d'association sur le MOTA et le MOTP de chaque ensemble de données. Pour la colonne de gauche, une valeur plus élevée représente un meilleur suivi alors que pour la colonne de droite, une valeur plus basse représente un suivi plus précis.

que montré pour la vidéo Sherbrooke (section 4.3.2). Il arrive parfois que certains véhicules partant en même temps d'une intersection à partir de deux voies différentes soient difficiles à segmenter. Toutefois, on retrouve également ce problème avec TI puisque le problème survient lorsque les véhicules se déplacent à la même vitesse.

4.3.9 Sensibilité des paramètres

Les résultats obtenus précédemment ont été générés avec des paramètres légèrement différents afin de prendre en compte la taille des objets, la durée des phases des feux aux intersections et le niveau de fragmentations des objets. Afin d'évaluer la sensibilité des algorithmes aux paramètres, nous avons traité tous les vidéos en utilisant les paramètres des autres vidéos. Le tableau 4.10 montre les résultats obtenus pour les 5 vidéos en utilisant les 5 ensembles de paramètres présentés a la section 4.3.1. On peut également y voir la différence entre les résultats obtenus à l'aide des paramètres communs et les paramètres optimisés. Les résultats des vidéos en utilisant les paramètres de Sherbrooke ne sont pas rapportés pour TI puisque les paramètres ont été calibrés en fonction d'une homographie et que l'homographie n'est pas disponible pour les autres scènes.

Ce tableau montre que le changement des paramètres a peu d'impact sur la précision de UT et que le MOTA varie peu également dans la plupart des cas, et ce malgré la grande variété des paramètres utilisés. Les paramètres par défaut décrits dans la section 3 devraient donc convenir à la plupart des scènes urbaines sans demander d'ajustement. Les colonnes de ΔA et ΔP montrent que TI est beaucoup plus dépendant des paramètres que UT, car la variation y est beaucoup plus extrême autant au niveau du MOTA que du MOTP. Le choix des paramètres est donc crucial pour TI. L'usage d'une homographie (si elle avait été disponible pour toutes les scènes) avec TI aurait également pu atténuer cet effet. Nous n'avons pas calculé les résultats pour TI avec les paramètres de Sherbrooke puisque celle-ci utilise une homographie qui ne serait pas applicable aux autres scènes.

Tableau 4.10 Résultats obtenus en utilisant les mêmes paramètres pour toutes les vidéos. ΔA et ΔP représente respectivement la différence entre les MOTA et MOTP obtenus pour les paramètres communs et les paramètres ajustés présentés précédemment. Pour ΔA un signe négatif signifie une baisse de qualité du suivi alors que pour ΔP, un signe positif présente une baisse de la précision du suivi.

Paramètres Sherbrooke								
	Urban Tracker				Traffic Intelligence			
Vidéo	MOTA	MOTP	ΔA	ΔP	MOTA	MOTP	ΔA	ΔP
Rouen	0.7697	14.28 px	**-5.37%**	**1.19 px**	N.D.	N.D.	N.D.	N.D.
St-Marc	0.7567	5.97 px	**0.00%**	**0.00 px**	N.D.	N.D.	N.D.	N.D.
René-Lévesque	0.7261	2.80 px	**-0.06%**	**-0.17 px**	N.D.	N.D.	N.D.	N.D.
Atrium	0.9144	7.75 px	**-2.15%**	**-0.18 px**	N.D.	N.D.	N.D.	N.D.

Paramètres Rouen								
	Urban Tracker				Traffic Intelligence			
Vidéo	MOTA	MOTP	ΔA	ΔP	MOTA	MOTP	ΔA	ΔP
Sherbrooke	0.7807	8.69 px	**0.36%**	**0.03 px**	0.3794	13.80 px	-0.47%	6.26 px
St-Marc	0.6952	7.07 px	**-2.03%**	**0.72 px**	0.4219	20.93 px	-18.0%	6.35 px
René-Lévesque	0.6741	3.04 px	**-5.26%**	**0.07 px**	0.3055	5.77 px	-19.7%	0.67 px
Atrium	0.8952	8.49 px	**-4.07%**	**0.56 px**	0.5360	52.05 px	-15.4%	2.70 px

Paramètres St-Marc								
	Urban Tracker				Traffic Intelligence			
Vidéo	MOTA	MOTP	ΔA	ΔP	MOTA	MOTP	ΔA	ΔP
Sherbrooke	0.7771	8.66 px	**0.00%**	**0.00 px**	-0.8599	19.90 px	-124.40%	12.36 px
Rouen	0.7697	14.28 px	**-5.37%**	**1.19 px**	0.4427	30.64 px	-14.58%	6.44 px
René-Lévesque	0.7261	2.80 px	**-0.06%**	**-0.17 px**	0.2056	5.95 px	-29.73%	0.85 px
Atrium	0.9144	7.75 px	**-2.15%**	**-0.18 px**	0.2838	54.87 px	-40.65%	5.52 px

Paramètres René-Lévesque								
	Urban Tracker				Traffic Intelligence			
Vidéo	MOTA	MOTP	ΔA	ΔP	MOTA	MOTP	ΔA	ΔP
Sherbrooke	0.7787	9.25 px	**0.16%**	**0.59 px**	-0.5029	17.29 px	-88.70%	9.75 px
Rouen	0.7122	15.59 px	-11.12%	**2.50 px**	0.5173	34.17 px	**-7.12%**	9.97 px
St-Marc	0.6781	7.74 px	**-7.86%**	**1.77 px**	0.3352	16.78 px	-26.66%	2.2 px
Atrium	0.9094	8.09 px	**-2.65%**	0.16 px	0.1398	47.32 px	-69.03%	**-2.03 px**

Paramètres Atrium								
	Urban Tracker				Traffic Intelligence			
Vidéo	MOTA	MOTP	ΔA	ΔP	MOTA	MOTP	ΔA	ΔP
Sherbrooke	0.7832	8.58 px	**0.61%**	**-0.08 px**	0.2735	26.71 px	-8.95%	19.17 px
Rouen	0.7906	13.54 px	**-3.28%**	**0.45 px**	0.2946	57.86 px	-35.71%	33.66 px
St-Marc	0.7115	6.58 px	**-4.52%**	**0.61 px**	0.2314	25.35 px	-64.43%	10.77 px
René-Lévesque	0.6862	3.71 px	**-4.05%**	**0.74 px**	-0.0425	9.18 px	-50.29%	4.08 px

CHAPITRE 5

CONCLUSION

Nos travaux ont visé à développer un système permettant le suivi des usagers de la route à une intersection routière. Ce système devait être en mesure de suivre les différents usagers de la route indépendamment de leur type, leur nombre, ou de leur grandeur. Le système conçu devait être en mesure de gérer les nombreuses occultations de ce genre de scène. Nous avions fixé trois objectifs à atteindre pour ce projet.

Le premier objectif était de détecter les différents objets en mouvement dans la séquence vidéo tout en ignorant les objets non pertinents tels que les feuilles des arbres et les ombres d'oiseaux. Pour ce faire, nous avons utilisé une technique de soustraction d'arrière-plan que nous avons modifiée afin d'éliminer rapidement les objets statiques. Pour éliminer les objets non pertinents, nous utilisons l'historique de leur suivi, le nombre de points caractéristiques présents dans les objets et leur taille. Cela nous permet de n'avoir que les objets d'intérêts circulant dans l'intersection (piétons, véhicules, cycliste) tout en éliminant les autres, tels que les feuilles d'arbres, les objets statiques, et les ombres d'oiseaux.

Notre second objectif consistait à extraire les trajectoires des différents objets en mouvement en calculant la correspondance entre les objets des différentes trames en utilisant une stratégie d'association de données et un modèle de suivi. Nous avons conçu un modèle de suivi basé sur les points caractéristiques, la position, et la forme du blob. Celui-ci nous permet d'extraire la trajectoire des usagers et d'estimer leur position même en présence d'occultations.

Notre dernier objectif était de valider la qualité des trajectoires extraites à l'aide de métriques standards et de comparer les résultats obtenus avec un autre algorithme. Nous avons conçu des logiciels afin d'évaluer les métriques CLEAR MOT qui sont les métriques les plus utilisées présentement pour le suivi multiobjet. Cinq vidéos représentant un grand nombre de situations urbaines et de point de vue différent ont été annotés. Les résultats obtenus pour ces séquences ont été comparés à ceux obtenus par Traffic Intelligence de Jackson *et al.* (2013), un algorithme de suivi en milieux urbains. Les résultats obtenus par notre algorithme Urban Tracker sont supérieurs autant au niveau de la précision que de la justesse du suivi. Cela s'explique par le fait que l'algorithme est capable de suivre des objets de tailles différentes simultanément ce qui lui permet de gérer le trafic mixte présent dans nos scènes d'intersection urbaine. De plus, l'algorithme est généralement plus précis grâce à l'utilisation de la soustraction d'arrière-plan pour l'estimation de la boîte englobante des objets.

Tel qu'énoncé dans l'introduction, notre contribution principale repose dans la conception d'un nouvel algorithme de suivi multiobjet conçu spécifiquement pour le milieu urbain. Cet algorithme est capable de gérer des objets indépendamment de leur type ou de leur taille et du fait que ceux-ci soient rigides ou non. Son utilisation ne nécessite aucune connaissance à priori de la scène et il nécessite peu ou pas d'ajustement de paramètres. Nous avons également proposé une procédure permettant d'estimer la position des boîtes englobantes en cas d'occultations partielles des objets. Finalement, nous avons développé des outils (qui seront publié en code ouvert) et annoté des vidéos afin de permettre aux autres chercheurs de comparer leurs algorithmes. Ces outils permettront également aux autres chercheurs d'annoter de nouvelles vidéos et d'exécuter notre algorithme sur celles-ci afin de comparer les résultats obtenus.

5.1 Améliorations futures possibles

Les faiblesses de notre algorithme proviennent directement de l'utilisation de la soustraction d'arrière-plan. Le premier problème est que les ombres peuvent déformer beaucoup les objets. Un certain nombre de techniques d'élimination d'ombres ont été essayées, mais celles-ci n'ont pas donné de bons résultats puisqu'elles avaient souvent pour effet d'enlever une bonne partie des objets considérés et de fragmenter les blobs. L'utilisation d'autres méthodes pourrait toutefois permettre d'améliorer la précision de l'algorithme. L'autre limitation de notre algorithme est qu'il faut que les objets se séparent durant le suivi pour que ceux-ci soient correctement suivis. Cela rend donc notre algorithme peu applicable aux scènes où le trafic est trop dense. De plus, cela nous empêche de suivre individuellement les objets se déplaçant en groupe (chaque piéton à l'intérieur d'un groupe par exemple). La connaissance du type des objets contenu dans un blob pourrait nous permettre d'appliquer des traitements particuliers par type d'objet afin de les segmenter. Pour identifier le type d'objet, des indices comme la taille, la vitesse de mouvement, ou la forme du déplacement pourraient permettre de classifier les objets par type. Pour les objets de type piétons, nous pourrions utiliser un détecteur de personne sur les blobs afin d'identifier les piétons individuels à l'intérieur d'un groupe. Un traitement similaire est d'ailleurs utilisé par Lascio *et al.* (2013) pour le suivi de piétons dans des blobs communs. Pour ce qui est des blobs de véhicules sous-segmentés, puisque la largeur d'un véhicule est similaire d'un véhicule à un autre, il pourrait être possible de segmenter ceux-ci. Certaines contraintes de rigidité comme ceux utilisés par Jackson *et al.* (2013) pourrait également être utilisés pour ces objets. Notre méthode pourrait donc bénéficier de contraintes de suivi basées sur le type des objets. Ce type de contraintes nécessitera toutefois sans doute un certain niveau de calibration de la scène afin d'assurer une taille

des objets qui soit indépendante de la perspective. Il faudra toutefois éviter de mettre des contraintes qui soient trop restrictives afin que le système reste assez générique pour suivre tous les types d'objets.

RÉFÉRENCES

ALAHI, A., ORTIZ, R. et VANDERGHEYNST, P. (2012). Freak : Fast retina keypoint. *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 510–517.

ANDRILUKA, M., ROTH, S. et SCHIELE, B. (2008). People-tracking-by-detection and people-detection-by-tracking. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. 1–8.

ANTOINE VACAVANT, THIERRY CHATEAU, A. W. et LEQUIÈVRE, L. (2012). A benchmark dataset for foreground/background extraction. *ACCV 2012, Workshop : Background Models Challenge*.

ANTONINI, G. et THIRAN, J. P. (2006). Counting pedestrians in video sequences using trajectory clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 16, 1008–1020.

ATEV, S., ARUMUGAM, H., MASOUD, O., JANARDAN, R. et PAPANIKOLOPOULOS, N. (2005). A vision-based approach to collision prediction at traffic intersections. *IEEE Transactions on Intelligent Transportation Systems*, 6, 416–423.

BARNICH, O. et VAN DROOGENBROECK, M. (2011). ViBe : A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20, 1709–1724.

BATISTA, J., PEIXOTO, P., FERNANDES, C. et RIBEIRO, M. (2006). A dual-stage robust vehicle detection and tracking for real-time traffic monitoring. *IEEE Intelligent Transportation Systems Conference, 2006. ITSC '06*. 528–535.

BERNARDIN, K. et STIEFELHAGEN, R. (2008). Evaluating multiple object tracking performance : the CLEAR MOT metrics. *J. Image Video Process.*, 2008, 1 :1 ?1 :10.

BEYMER, D., MCLAUCHLAN, P., COIFMAN, B. et MALIK, J. (1997). A real-time computer vision system for measuring traffic parameters. *, 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings*. 495–501.

BRADSKI, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

BREITENSTEIN, M. D., REICHLIN, F., LEIBE, B., KOLLER-MEIER, E. et VAN GOOL, L. (2011). Online multiperson tracking-by-detection from a single, uncalibrated camera. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33, 1820–1833.

BUCH, N., VELASTIN, S. et ORWELL, J. (2011). A review of computer vision techniques for the analysis of urban traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12, 920–939.

CHANG, F., JEN CHEN, C. et JEN LU, C. (2004). A linear-time component-labeling algorithm using contour tracing technique. *Computer Vision and Image Understanding*, 93, 206–220.

CHIA, A. Y. S., HUANG, W. et LI, L. (2006). Multiple objects tracking with multiple hypotheses graph representation. *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*. IEEE Computer Society, Washington, DC, USA, ICPR '06, 638–641.

COMANICIU, D., RAMESH, V. et MEER, P. (2000). Real-time tracking of non-rigid objects using mean shift. *IEEE Conference on Computer Vision and Pattern Recognition, 2000. Proceedings*. vol. 2, 142–149 vol.2.

DAHLKAMP, H., OTTLIK, A. et NAGEL, H.-H. (2004). Comparison of edge-driven algorithms for model-based motion estimation. *Proc. First International Workshop on Spatial Coherency for Visual Motion Analysis (SCVM'04*. Springer, 38–50.

DALAL, N. et TRIGGS, B. (2005). Histograms of oriented gradients for human detection. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*. vol. 1, 886–893 vol. 1.

DOLLAR, P., WOJEK, C., SCHIELE, B. et PERONA, P. (2012). Pedestrian detection : An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34, 743–761.

FISCHLER, M. A. et BOLLES, R. C. (1981). Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24, 381–395.

FUENTES, L. M. et VELASTIN, S. A. (2001). People tracking in surveillance applications. *In Proceedings of the 2nd IEEE International workshop on PETS*.

GOUVERNEMENT DU CANADA, T. C. (2012). Canadian motor vehicle traffic collision statistics : 2010. http ://www.tc.gc.ca/fra/securiteroutiere/tp-1317.htm. Liste des publications sur le site Web de la Sécurité routière.

GOYETTE, N., JODOIN, P., PORIKLI, F., KONRAD, J. et ISHWAR, P. (2012). Changedetection.net : A new change detection benchmark dataset. *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1–8.

GREEN, E., AGENT, K., PIGMAN, J., ADMINISTRATION, U. S. F. H., OF KENTUCKY TRANSPORTATION CENTER, U. et CABINET, K. T. (2005). *Evaluation of Auto Incident Recording System (AIRS)*. Research report (University of Kentucky Transportation Center). Kentucky Transportation Center, University of Kentucky.

HARITAOGLU, I., HARWOOD, D. et DAVIS, L. S. (1998). *W4 : Who ? When ? Where ? What ? A Real Time System for Detecting and Tracking People*.

HARRIS, C. et STEPHENS, M. (1988). A combined corner and edge detector. *In Proc. of Fourth Alvey Vision Conference*. 147–151.

HARTLEY, R. et ZISSERMAN, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge books online. Cambridge University Press.

HEINLY, J., DUNN, E. et FRAHM, J.-M. (2012). Comparative evaluation of binary features. *Proceedings of the 12th European conference on Computer Vision - Volume Part II*. Springer-Verlag, Berlin, Heidelberg, ECCV'12, 759–773.

JACKSON, S., MIRANDA-MORENO, L., ST-AUBIN, P. et SAUNIER, N. (2013). A flexible, mobile video camera system and open source video analysis software for road safety and behavioural analysis. Transportation Research Board Annual Meeting Compendium of Papers, 2013.

KIM, Z. (2008). Real time object tracking based on dynamic feature grouping with background subtraction. *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*. 1–8.

KUO, C.-H. et NEVATIA, R. (2009). Robust multi-view car detection using unsupervised sub-categorization. *Applications of Computer Vision (WACV), 2009 Workshop on*. 1–8.

LASCIO, R. D., FOGGIA, P., PERCANNELLA, G., SAGGESE, A. et VENTO, M. (2013). A real time algorithm for people tracking using contextual reasoning. *Computer Vision and Image Understanding*.

LEE, D. C. et KANADE, T. (2007). Boosted classifier for car detection. *Unpublished*.

LEUTENEGGER, S., CHLI, M. et SIEGWART, R. (2011). BRISK : binary robust invariant scalable keypoints. *2011 IEEE International Conference on Computer Vision (ICCV)*. 2548–2555.

LI, J. et ALLINSON, N. M. (2008). A comprehensive review of current local features for computer vision. *Neurocomputing*, 71, 1771–1787.

LINAN, C. C. (2012). cvblob. http ://cvblob.googlecode.com. Cvblob.

LOWE, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60, 91–110.

LUCAS, B. D. et KANADE, T. (1981). An iterative image registration technique with an application to stereo vision. *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'81, 674–679.

MAIR, E., HAGER, G. D., BURSCHKA, D., SUPPA, M. et HIRZINGER, G. (2010). Adaptive and generic corner detection based on the accelerated segment test. *European Conference on Computer Vision (ECCV'10)*.

MORAVEC, H. P. (1980). *Obstacle avoidance and navigation in the real world by a seeing robot rover*. Thèse de doctorat, Stanford, CA, USA. AAI8024717.

MUNKRES, J. (1957). Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5, 32–38.

NONAKA, Y., SHIMADA, A., NAGAHARA, H. et TANIGUCHI, R. (2012). Evaluation report of integrated background modeling based on spatio-temporal features. *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. 9–14.

OZUYSAL, M., LEPETIT, V. et FUA, P. (2009). Pose estimation for category specific multiview object localization. *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. 778–785.

PROJECT/IST 2001 37540, E. F. C. (2004). Caviar : Context aware vision using image-based active recognition. Caviar.

ROSTEN, E. et DRUMMOND, T. (2005). Fusing points and lines for high performance tracking. *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005.* vol. 2, 1508–1515 Vol. 2.

ROSTEN, E. et DRUMMOND, T. (2006). Machine learning for high-speed corner detection. *Proceedings of the 9th European conference on Computer Vision - Volume Part I*. Springer-Verlag, Berlin, Heidelberg, ECCV'06, 430–443.

SANIN, A., SANDERSON, C. et LOVELL, B. C. (2012). Shadow detection : A survey and comparative evaluation of recent methods. *Pattern Recogn.*, 45, 1684–1695.

SAUNIER, N. et SAYED, T. (2006). A feature-based tracking algorithm for vehicles in intersections. *The 3rd Canadian Conference on Computer and Robot Vision, 2006*. 59–59.

SHI, J. et TOMASI, C. (1994). Good features to track. *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on.* 593–600.

SONG, X. et NEVATIA, R. (2005). A model-based vehicle segmentation method for tracking. *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005.* vol. 2, 1124–1131 Vol. 2.

STAUFFER, C. et GRIMSON, W. E. L. (1999). Adaptive background mixture models for real-time tracking. *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.* vol. 2, 246–252.

TAMERSOY, B. et AGGARWAL, J. (2009). Robust vehicle detection for tracking in highway surveillance videos using unsupervised learning. *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, 2009. AVSS '09.* 529–534.

THE UNIVERSITY OF READING, U. (2001). Second ieee international workshop on performance evaluation of tracking and surveillance. http ://www.cvg.rdg.ac.uk/pets2001/. Pets2001.

THE UNIVERSITY OF READING, U. (2009). Eleventh ieee international workshop on performance evaluation of tracking and surveillance. http ://www.cvg.rdg.ac.uk/PETS2009/. Pets2009.

TORABI, A. et BILODEAU, G.-A. (2009). A multiple hypothesis tracking method with fragmentation handling. *Computer and Robot Vision, 2009. CRV '09. Canadian Conference on.* 8–15.

VIOLA, M., JONES, M. J. et VIOLA, P. (2003). Fast multi-view face detection. *Proc. of Computer Vision and Pattern Recognition.*

VIOLA, P. et JONES, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001.* vol. 1, I–511–I–518 vol.1.

WANG, S., SALVI, D., WAGGONER, J. et TEMLYAKOV, A. (2013). A graph-based algorithm for multi-target tracking with occlusion. *Proceedings of the 2013 IEEE Workshop on Applications of Computer Vision (WACV).* IEEE Computer Society, Washington, DC, USA, WACV '13, 489–496.

YIN, F., MAKRIS, D. et VELASTIN, S. A. (2007). Performance Evaluation of Object Tracking Algorithms. *10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2007), Rio de Janeiro, Brazil.*

ANNEXE A

Format Polytrack

Le format Polytrack est une extension du format SQLite utilisé par Traffic Intelligence de Jackson *et al.* (2013). Il est d'ailleurs compatible avec ce format. Polytrack est un format de d'enregistrement de données de suivi conçu pour le suivi multiobjet et multitype 2D à base de points ou de boîtes englobantes. Celui-ci est composé des tables suivantes :

La table *objects* (table A.1) contient l'ensemble des informations se rapportant à un objet, c'est-à-dire son nom, son type et sa description.

Tableau A.1 Table *object*

objects			
nom du champs	type	propriété	description
object_id	nombre entier	clé	identifiant de l'objet
road_user_type	nombre entier	N/A	identifiant du type de l'objet
description	texte	optionnel	description textuelle de l'objet

La table *objects_type* (table A.2) est quand à elle utilisé afin de fournir une version texte de chaque type. Il est possible de définir de nouveaux types via cette table. Cette table est optionnelle et il n'est pas nécessaire de la définir si le type désiré existe déjà dans le tableau A.3.

Tableau A.2 Table *objects_type*

objects_type			
nom du champs	type	propriété	description
road_user_type	nombre entier	clé	identifiant du type de l'objet
type_string	texte	N/A	description textuelle du type

Les autres tables dépendent du type d'enregistrement (trajectoire de points ou boîtes englobantes). Pour les méthodes à base de points, il faut utiliser les table *positions* et *objects_features*. La table *positions* (table A.4) permet de reconstruire les trajectoires complètes des points et la table *objects_features* (table A.5) permet de faire l'association entre les trajectoires d'un point et leur regroupement dans un même objet.

Les méthodes qui enregistrent directement des boîtes englobantes utilisent plutôt la table *bounding_boxes* A.6 qui contient les quatres coins de la boîte englobante de chaque objet pour chaque trame.

Tableau A.3 Types existant

Type existant	
road_user_type	*type_string*
0	inconnu
1	automobile
2	piétons
3	motocyclette
4	vélo
5	autobus
6	camion

Tableau A.4 Table *positions*

positions			
nom du champs	type	propriété	description
trajectory_id	nombre entier	clé	identifiant de trajectoire
frame_number	nombre entier	clé	numéro de la trame
x_coordinate	nombre flottant	N/A	coordonnée x du point
y_coordinate	nombre flottant	N/A	coordonnée y du point

Les coordonnées des boîtes englobantes sont sauvegardées dans le point de vue de l'image alors que les points sont eux enregistré en coordonné planaire et il faut leur appliquer une transformation à l'aide d'une homographie afin d'obtenir les coordonnées dans le point de vue de l'image.

Tableau A.5 Table *objects_features*

objects_features			
nom du champs	type	propriété	description
object_id	nombre entier	clé	identifiant de l'objet
trajectory_id	nombre entier	clé	identifiant de trajectoire

Tableau A.6 Table *bounding_boxes*

bounding_boxes			
nom du champs	type	propriété	description
object_id	nombre entier	clé	identifiant de l'objet
frame_number	nombre entier	clé	numéro de la trame
x_top_left	nombre flottant	N/A	coordonnée x du coin en haut à gauche
y_top_left	nombre flottant	N/A	coordonnée y du coin en haut à gauche
x_bottom_right	nombre flottant	N/A	coordonnée x du coin en bas à droite
y_bottom_right	nombre flottant	N/A	coordonnée y du coin en bas à droite

ANNEXE B

Manuel d'utilisation de l'annotateur

L'outil d'annotation permet d'annoter des vidéos à l'aide de boîtes englobantes. Il permet également de visionner des résultats enregistrés en «PolyTrack». L'outil permet également d'utiliser le résultat des annotations pour générer des ensembles de données d'entrainement à des fins de test. Il est ensuite possible d'exporter les vidéos avec les annotations superposés.

L'outil gère plusieurs sortes d'importateur et exporte dans un format de fichier commun. D'autres importateurs/exportateurs peuvent être écrit aisément en C++.

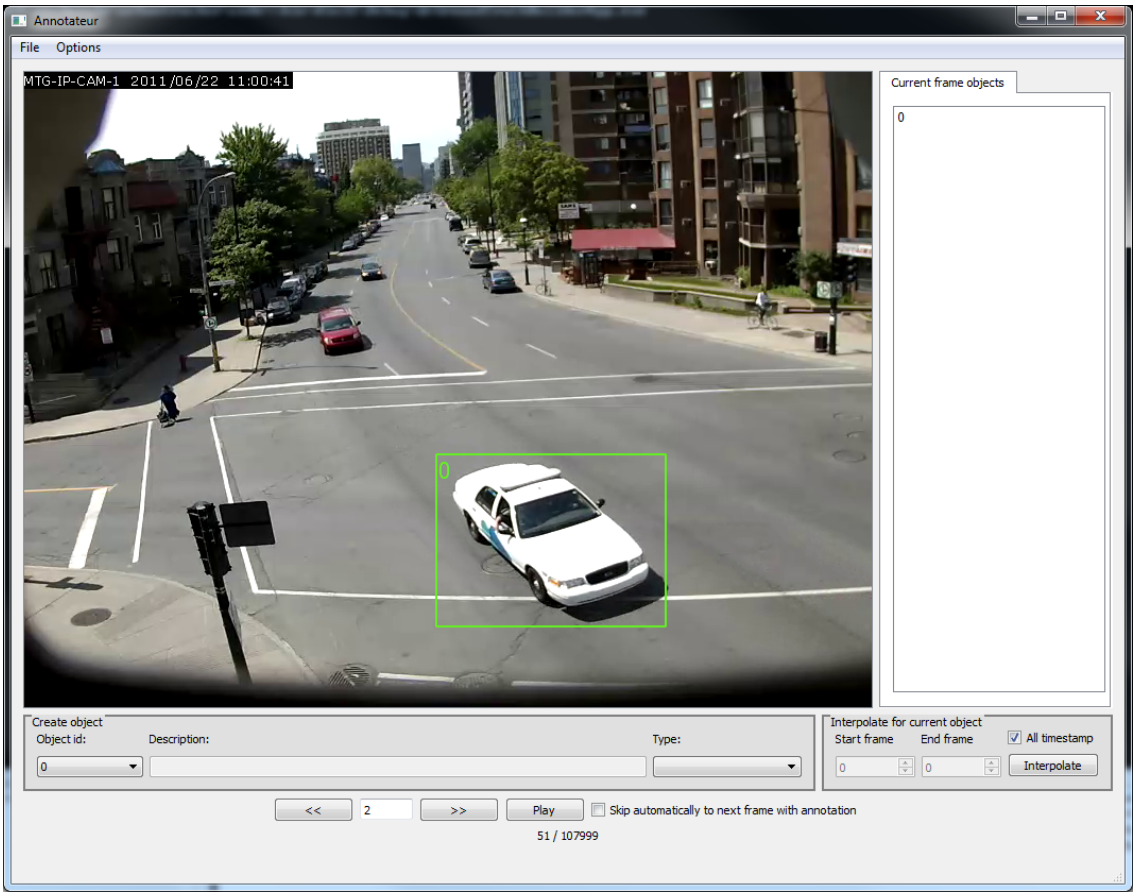


Figure B.1 Interface de l'annotateur

B.0.1 Importation

Avant d'importer, il faut d'abord charger une vidéo avec «File/Load video». On peut ensuite importer des fichiers avec «Load ground truth». Le format commun est PolyTrack.

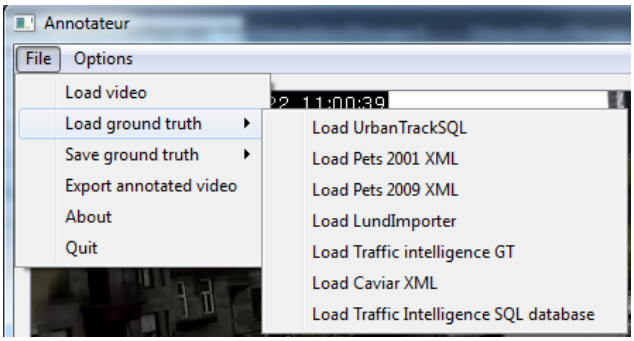


Figure B.2 Option d'importation

Il s'agit également du format d'exportation. Certains autres formats XML/SQL/TXT sont également supportés. D'autres importateurs peuvent être créés au besoin.

B.0.2 Exportation

Le programme enregistre seulement en PolyTrack pour l'instant. Il s'agit d'un format sqlite.

B.0.3 Sauvegarde de vidéo de résultat

Il est possible de sauvegarder la vidéo avec les annotations ensuite et de les rejouer dans un éditeur vidéo.

B.0.4 Création d'objets

Pour créer une boîte englobante, il suffit de faire un clic gauche dans l'interface et de glisser. Pour annuler, il suffit de faire une boîte de taille inférieure à 3x3 pixels. Il est possible de supprimer les boîtes avec le bouton supprimer du clavier. Le champ Object id indique dans quel objet sera créé la nouvelle boîte englobante. Si «New Object» est sélectionné, alors un nouvel objet est créé (dernière option de la liste «ObjectId»). Lorsqu'une boîte englobante est sélectionnée, il est alors possible d'y assigner une description et un type. L'option «New object type» permet de créer un nouveau type.

B.0.5 Interpolation des observations

Il est possible de faire une interpolation linéaire entre un certain nombre d'observations. Il suffit de tracer deux boîtes englobantes associées à un même objet à 2 trames différentes et de cliquer sur le bouton «Interpolate». Les boîtes englobantes interpolées seront marquées avec un «-I» à côté de leur nom. Si celles-ci sont modifiées (déplacées ou redimensionnées), alors le I disparaitra et celle-ci sera considéré comme une boîte positionnée manuellement.

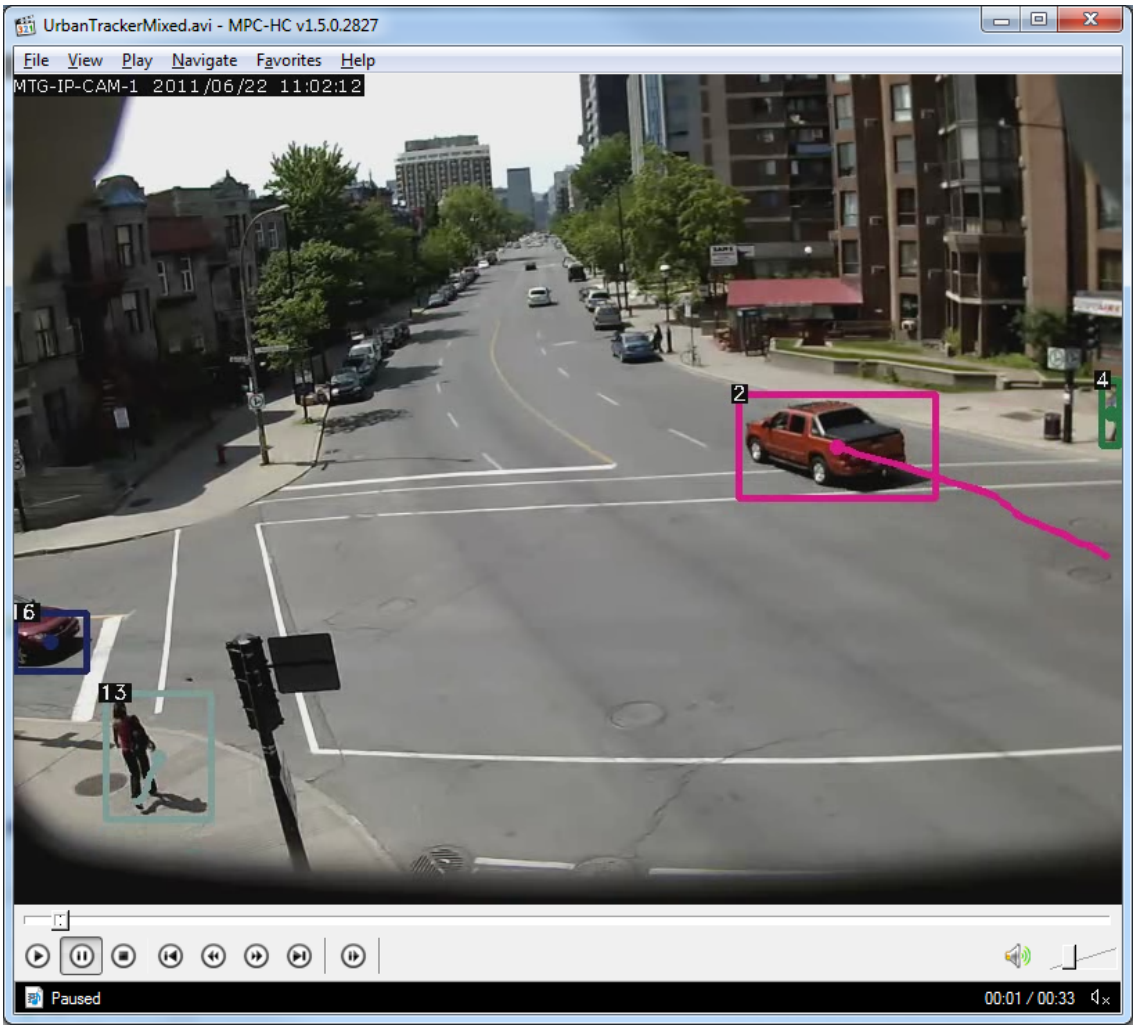


Figure B.3 Vidéo de résultat

Recliquer sur le bouton ‘Interpolate aura alors pour effet de refaire l’interpolation en utilisant cette nouvelle boîte de contrôle.

Attention : Lors de la sauvegarde de la base de donnée, l’information sur l’interpolation sera supprimée et toute les boîtes seront considérées comme ayant été placées manuellement.

B.0.6 Contrôle vidéo

La boîte de texte permet de définir le pas (*step*), c’est-à-dire de combien de trames à la fois on veut avancer ou reculer la vidéo.

Le bouton «<<» permet de reculer de N trames (step) la vidéo. Le bouton «>>» permet d’avancer de N trames (step) la vidéo. Le bouton Play permet de jouer la vidéo à 30 FPS L’option «Skip automatically. . . » permet de n’afficher que les trames avec des annotations. On ira donc chercher la prochaine trame avec une annotation.

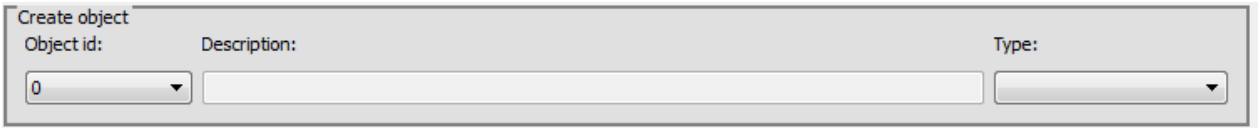


Figure B.4 Panneau inférieur

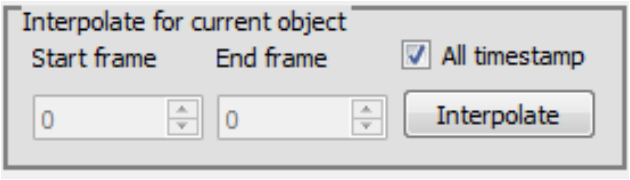


Figure B.5 Panneau de contrôle de l'interpolation

B.0.7 Options

Add object type : Permet d'ajouter un type d'objet (fonction identique à ce qui est fait lorsqu'on sélectionne l'option «New object type» du menu «Type»).

Merge objects : Cette option permet de fusionner plusieurs objets ensemble. Les observations seront alors toutes fusionnées au même objet.

Change id : Cette option permet de changer l'identité des objets vers un autre objet. Il est possible de choisir la durée temporelle.

Delete range : Cette option permet de supprimer les observations de certains objets pour un interval de temps.

Apply homography : Cette option permet d'appliquer une matrice homographique à des observations.

Apply Mask : Cette option permet d'appliquer un masque à une vérité terrain. Il suffit de charger un fichier image avec des pixel blanc (zone d'intérêt) et des pixels noirs.

Count number of box : Permet d'afficher le nombre total de boîtes englobantes dans la barre d'état.

Le menu extraction contient des options pour extraire des échantillons positif et négatif à partir des observations annotées.

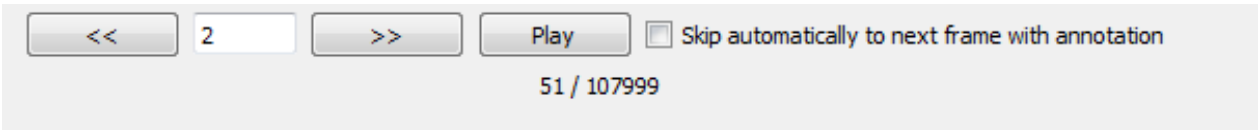


Figure B.6 Panneau de contrôle vidéo

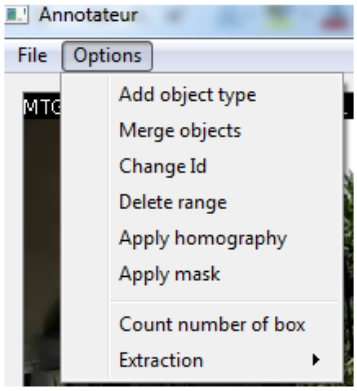


Figure B.7 Menu des options

ANNEXE C

Manuel d'utilisation de l'outil d'évaluation des métriques

L'outil d'évaluation des métriques est un outil en ligne des commandes qui lit des fichiers au format Polytrack. Voici les arguments qu'il prend en entrée :
1er argument : Chemin vers la vérité de terrain (*ground truth*)
2e argument : Chemin vers les données du suiveur (*tracker*)
3e argument : Mode d'association des données ABSDIST ou BBOVERLAP (évaluation par distance de centroïd ou superposition de boîtes)
4e argument : Seuil d'association, Si le 3e arg est ABSDIST, alors distance en pixel. Si BBOVERLAP, alors valeur entre 0 et 1
Arguments optionnel :
5e argument : chemin vers un fichier pour enregistrer les résultats.
6e argument : Mettre noprompt si on veut que le programme se ferme automatiquement à la fin

Il est possible pour l'outil de sortir des fichiers de résultat en format CSV. Voici un exemple de sortie : Format des fichiers de résultats :
NB_GT_TRACKS ;53
NB_SYSTEM_TRACKS ;76
CDT ;29
FAT ;38
TDF ;24
TF ;15
IDC ;1
LT ;10.4138
CTM ;0.322773
CDT ;29
TMEMT ;43.7471
TMEMTD ;15.3074
TCM ;0.517275
TCD ;0.00849958
MOTA ;0.433373
MOTP ;44.8625

MISSES_RATIO ;0.400949
FP_RATIO ;0.163899
MME_RATIO ;0.400949