Titre: Title:	Suivi multiobjet en situation urbaine à l'aide de la programmation par contraintes
Auteur: Author:	Alexandre Pineault
Date:	2019
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Pineault, A. (2019). Suivi multiobjet en situation urbaine à l'aide de la programmation par contraintes [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/4165/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Suivi multiobjet en situation urbaine à l'aide de la programmation par contraintes

ALEXANDRE PINEAULT
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Décembre 2019

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Suivi multiobjet en situation urbaine à l'aide de la programmation par contraintes

présenté par **Alexandre PINEAULT**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Thomas HURTUT, président
Guillaume-Alexandre BILODEAU, membre et directeur de recherche
Gilles PESANT, membre et codirecteur de recherche
Nicolas SAUNIER, membre

REMERCIEMENTS

Pour les nombreux conseils qu'ils m'ont partagés tout au long de cette recherche, les révisions minutieuses qu'ils ont réalisées pour ce mémoire et pour toutes autres rédactions réalisées au cours de ma maîtrise ainsi que pour le soutien financier qu'ils m'ont offert, je remercie sincèrement mon directeur de recherche Guillaume-Alexandre Bilodeau et mon codirecteur Gilles Pesant. J'exprime également ma reconnaissance envers tous mes collègues de laboratoire, mais plus particulièrement à Hugues Perreault et Hui-Lee Ooi qui m'ont permis non seulement d'apporter certaines améliorations à ce travail de recherche, mais aussi d'étendre mes connaissances dans le domaine du suivi multiobjet au travers de discussions. Je termine en remerciant Thomas Hurtut, président du jury, et Nicolas Saunier, membre externe, pour la révision de ce mémoire.

RÉSUMÉ

Ce mémoire présente une nouvelle approche pour effectuer le suivi de plusieurs objets utilisant la programmation par contraintes (CP) pour transformer une série de détections d'usagers de la route en un ensemble de trajectoires. Au meilleur de nos connaissances, il s'agit d'une première tentative de combiner le suivi d'objets et la programmation par contraintes. De nombreuses applications pourront bénéficier des avancées concernant la formation de trajectoires par le suivi multiobjet, notamment dans le domaine de la sécurité routière où l'identification de sites potentiellement dangereux permettrait d'éviter de futurs accidents de la route.

Sur une note plus technique, exprimer le problème du suivi multiobjet dans un cadre CP demande tout d'abord la formulation d'un modèle exprimé à l'aide d'un ensemble de variables à domaine fini et de contraintes s'appliquant sur ces variables. Pour que la modélisation soit un succès, il faut à la fois que les contraintes correspondent le mieux possible à la définition du problème, soit d'associer chaque détection d'objet à son identité trame après trame en se basant sur des images, mais également que ces contraintes soient suffisamment faciles à propager de telle sorte que la résolution prenne le moins de temps possible. De manière appliquée, les contraintes de notre modèle restreignent les déplacements considérés comme étant acceptables pour un même objet d'une trame à l'autre. Des contraintes préservant l'apparence de chaque objet tout au long de leur trajectoire ont également été ajoutées au modèle.

Une grande partie du travail du suivi multiobjet consiste à corriger des erreurs commises lors de l'étape de détection des objets d'intérêt. Certains de ces objets demeurent quelques fois non identifiés alors que d'autres fois, ce sont des parties de l'arrière-plan telles que des panneaux de signalisation qui sont identifiés comme étant d'intérêt. Une méthode de filtrage corrigeant ces anomalies a donc été développée et appliquée sur l'ensemble des détections avant même de commencer la phase d'association. Cette méthode implique l'utilisation de techniques de suivi d'un seul objet (SOT) servant à prédire l'ajout de nouvelles détections.

Cette méthode de suivi a été testée sur un premier ensemble de données portant sur le suivi d'usagers de la route (piétons et véhicules motorisés) ainsi que sur un deuxième portant sur le suivi de piétons uniquement. Bien que les résultats du suivi de piétons n'atteignent pas le niveau de l'état de l'art, ils demeurent compétitifs face aux méthodes utilisant des caractéristiques similaires aux nôtres. Toutefois, les performances obtenues sur les séquences avec les usagers de la route sont compétitives face aux autres méthodes constituant l'état de l'art de cette application. Comme il s'agit d'une nouvelle méthode, plusieurs expériences

ont été réalisées afin de déterminer que chaque module développé permet l'amélioration de la qualité du suivi.

ABSTRACT

This thesis introduces a new method for multiple object tracking (MOT) using constraint programming (CP). More specifically, the goal of this work is to solve the data association phase where sets of individual detections are combined together to form trajectories. To the best of our knowledge, this is the first attempt to use CP in a visual tracking context. Since trajectories are the building blocks of numerous applications, especially in the field of road safety, identifying dangerous road crossings may lead to faster urban planner interventions. These interventions should reduce the mortality caused by road accidents.

Tracking multiple objects at the same time through the use of constraint programming requires first to define a CP model. A CP model is made of finite domain variables and constraints restraining the domain possibilities in order to obtain a solution at the end having only one possible value per variable. To be successful, a model needs to correctly represent the problem, i.e. match each road user detection to an identity shared across multiple frames. It also needs to be easy to propagate, otherwise, the solver will not be able to find a solution in an acceptable amount of time. The constraints defined in this work address the motion of each objects by forbidding unlikely movements, but also make sure to preserve the general appearance between detections of a same trajectory.

Correcting the detector mistakes is an important task of MOT algorithms. Many errors of different types (e.g., missing objects of interest or identifying background regions as road users) are made before the data association step, causing a significant drop in the tracking results. A filtering method is proposed in this thesis in order to improve the set of detections given as input to the CP model. The main part of this process is the use of visual object tracking (VOT) techniques to predict where detections are missing and adding new boxes to fill in these gaps.

This tracking method was applied on two different datasets. The first one, UA-DETRAC, is exclusively considering road vehicles such as cars, bus, taxis and trucks, while the second, 2D MOT 2015 focuses on pedestrian tracking. State-of-the-art performances where not achieved for pedestrian tracking, but the results were outperforming many methods using similar features as used in this work. However, the tracking performances were better for the vehicles dataset; this method was able to outperform a state-of-the-art algorithm from the UA-DETRAC benchmark. In addition, as this is an attempt to perform tracking using new techniques, more results detailing the performance of individual parts of our process will be provided with this thesis.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

CP	Constraint Programming
MOT	Multiple Object Tracking
HOG	Histogrammes de gradients

CHAPITRE 1 INTRODUCTION

Le suivi de plusieurs objets à la fois ou suivi multiobjet constitue l'un des nombreux défis du domaine de la vision par ordinateur. Reconnaître et identifier un objet est une tâche loin d'être triviale pour un programme informatique et qui pourtant s'effectue de manière naturelle chez de nombreuses espèces vivantes. Il y a également plusieurs applications dans lesquelles le suivi multiobjet peut s'avérer utile. Beaucoup de villes importantes utilisent aujourd'hui des systèmes de caméras installées dans l'espace public à des fins de sécurité ou d'analyse du trafic routier. Or, employer des surveillants humains pour traiter les informations visuelles provenant de tels systèmes relève de l'impossible dû au grand nombre de ressources nécessaires pour couvrir entièrement le réseau de caméras. De plus, les opérateurs humains perdent rapidement leur concentration lorsqu'ils surveillent des écrans. Pour ces raisons, développer une méthode informatisée en mesure de mettre à profit ces séquences vidéos en produisant les trajectoires des différents usagers de la route est utile pour ensuite détecter de potentielles situations problématiques. De plus le traitement automatique permet de diminuer la surcharge cognitive des opérateurs humains.

Obtenir les trajectoires des différentes entités se partageant la route peut servir au niveau de la prévention d'accidents. Certaines intersections peuvent présenter un risque plus grand dû à une conception problématique et utiliser ces trajectoires afin de détecter la fréquence à laquelle se produisent des comportements anormaux ou inattendus peut contribuer à corriger la situation. Une autre manière de tirer parti de la formation de ces trajectoires est de les regrouper en classes selon le type de comportement observé (un virage à droite ou à gauche par exemple). Ensuite, on peut utiliser la répartition des différentes trajectoires parmi ces classes pour déterminer les comportements les plus fréquents et apporter des correctifs aux intersections.

Au niveau technique, le suivi multiobjet est un problème connu depuis plusieurs années. Une approche répandue est de diviser le problème en deux phases distinctes à exécuter de manière séquentielle, soit la détection d'objets d'intérêt et l'attribution d'une identité propre à toute entité préalablement détectée. La figure 1.1 illustre ce processus.

Au cours de la première phase, un détecteur repère les régions de l'image qui contiennent une entité d'intérêt et appose un rectangle englobant délimitant soit les frontières de la région de l'image contenant l'entité en question ou soit l'ensemble des points caractéristiques identifiés. Un tel traitement est effectué sur toutes les images d'une séquence vidéo. Au terme de ce processus, on obtient une séparation claire entre les objets d'intérêts et l'arrière-plan

ou l'environnement, qui correspond aux parties de l'image où aucun objet d'intérêt n'est représenté.

Dans la deuxième phase, chaque objet trouvé à la première phase se fait attribuer une identité propre, c'est-à-dire un identifiant indiquant la trajectoire à laquelle appartient la détection. Pour les trames subséquentes, il faut retrouver les objets qui correspondent aux bonnes identités tout en tenant compte que de nouvelles entités peuvent faire leur entrée ou sortir à tout moment. À la fin du processus, la combinaison de la position du de l'objet dans l'image, de son identité ainsi que de son identifiant de trame de toutes les détections affiliées à une même entité constituent sa trajectoire. On peut approcher la phase d'attribution des identités en divisant la problématique en trois sous-problèmes interreliés :

1. **Modèle de mouvements** : À cette étape, on cherche à utiliser les informations basées sur le positionnement des objets, souvent approximé par les coordonnées du centre de la boîte englobante. En accumulant ces positionnements dans une même trajectoire, la direction et la vitesse de chaque objet peuvent être calculées et intégrées au modèle de mouvement. Un suivi basé sur un modèle simple serait de favoriser les associations entre deux détections de trames consécutives selon leur proximité.
2. **Modèle d'apparence** : Un moyen de s'assurer qu'une trajectoire contienne toujours le même objet d'intérêt est de se servir des caractéristiques visuelles présentes à l'intérieur de la boîte englobante fournie par le détecteur. On peut ainsi extraire les contours, la répartition des couleurs, certaines textures locales et autres descripteurs d'apparence et ainsi s'assurer que l'objet suivi ne change pas brusquement d'apparence, ce qui pourrait révéler une erreur commise lors du processus d'identification.
3. **Association des données** : Finalement, une fois que tous les objets détectés ont été décrits spatialement et visuellement, il faut apposer une étiquette d'identité sur chacun d'entre eux. À cette étape, on mesure la similarité entre les détections de trames différentes selon les modèles de mouvements et d'apparence et on établit, globalement, quelles sont les meilleures associations à faire.

Aucune méthode n'est en mesure, à l'heure actuelle, de produire correctement les trajectoires de tous les objets pour une séquence vidéo filmant une route passante. Plusieurs situations viennent compliquer la tâche du processus de suivi d'objets multiples.

1.1 Les défis du suivi multiobjet

Avant de présenter ces défis, les notions d'erreurs de types faux positif et faux négatif doivent être définies clairement. Comme le problème étudié dans ce mémoire est la phase d'associa-

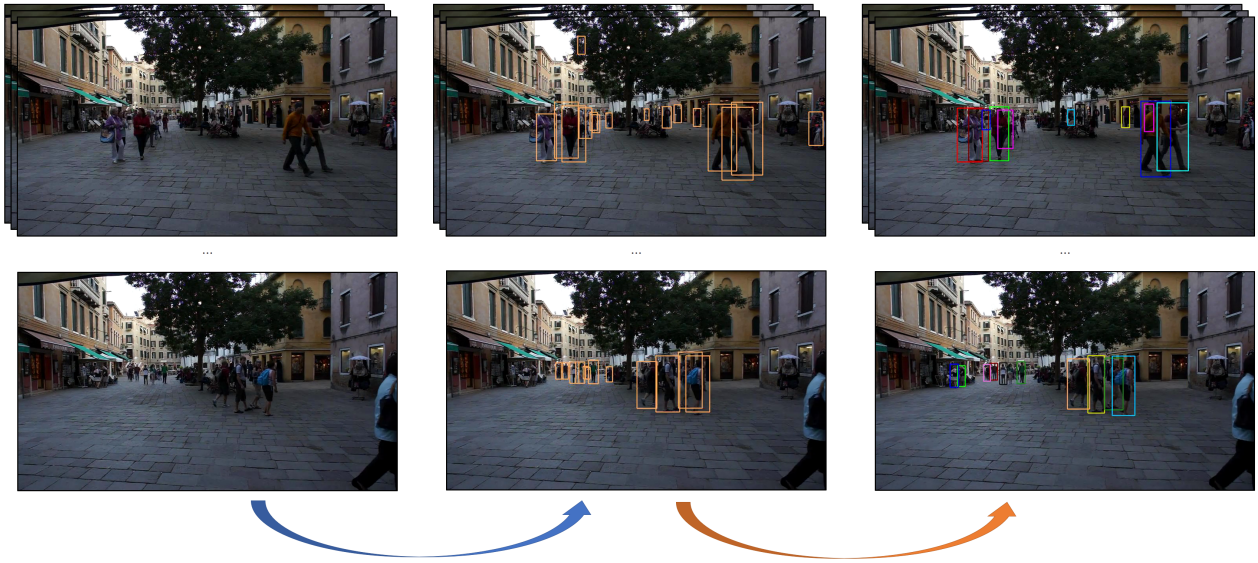


Figure 1.1 Illustration du processus de suivi multiobjet «Tracking-by-detection»(la flèche bleue indique le processus de détection et la flèche orange, le processus d'association)

tion et non de détection, un faux positif correspond à l'action d'utiliser un objet non-valide dans une trajectoire (erreur au niveau de l'association) et non au fait de prendre une partie d'arrière-plan pour un objet d'intérêt (erreur de détection). De cette manière, le détecteur peut fournir une détection en trop ; si elle est correctement filtrée, on ne sera pas en présence d'un faux positif. Le même principe s'applique pour les faux négatifs.

Une première catégorie de défis concerne la détection d'objets d'intérêts, soit la détection d'usagers de la route tels que les piétons, les voitures et autres véhicules motorisés dans le cadre de cette recherche. Voici les principales situations qui diminuent les performances des détecteurs :

- **Occlusions** : Selon l'angle de la caméra par rapport au sol et l'environnement, il peut arriver qu'un piéton soit caché par une autre partie de l'image, qu'il s'agisse d'un second piéton ou d'un objet de l'environnement. Ce phénomène a pour effet de réduire la partie visible du piéton, ce qui altère les caractéristiques utiles à la reconnaissance, par exemple la forme et la distribution des couleurs. En cas extrême, l'objet est complètement absent de l'image et la détection est impossible.
- **Déformations de l'objet** : Il peut arriver dans certaines situations qu'un objet change de forme. Pour se déplacer un piéton déplace ses jambes et ses bras, ce qui modifie non seulement son apparence, mais aussi les proportions du rectangle englobant qui le circonscrit. Pour bien performer, les détecteurs doivent s'assurer de ne pas être trop

sensibles à ces déformations. Il peut également arriver que deux objets se combinent pour n'en former qu'un seul et inversement, par exemple une personne qui retire son chapeau ou qui prend sa valise.

- **Variations de la pose** : Les piétons, comme la plupart des usagers de la route, sont en déplacement la majorité du temps. L'orientation de ces entités par rapport à la caméra est donc changeante, forçant les méthodes de détections à être robustes à ce genre de transformation pour éviter de ne pas détecter trop d'objets. Dans le cas d'un piéton, il s'agit de savoir le reconnaître de face, de dos et selon d'autres angles intermédiaires.
- **Variations d'échelle** : Lorsqu'un objet proche de la caméra s'éloigne, sa taille diminue dû au phénomène de perspective. Ce type de variations ajoute une difficulté aux détecteurs qui ne peuvent pas utiliser des textures ou des motifs de taille fixe pour reconnaître un piéton. De plus, une baisse du nombre de pixels survient lors de l'éloignement d'un objet, ce qui diminue la précision des caractéristiques de ce même objet.
- **Variations d'illumination** : La couleur perçue d'un objet varie en fonction de la source lumineuse qui l'éclaire. Cet effet est particulièrement présent lors de vidéos tournées dans un environnement extérieur où le déplacement du soleil et des nuages environnants peuvent modifier l'intensité lumineuse ambiante.

Une fois l'étape de détection complétée, il faut maintenant établir l'identité de chaque piéton identifié afin d'en obtenir la trajectoire. Comme le détecteur travaille de manière indépendante à la méthode d'identification, toutes les caractéristiques des objets d'intérêts doivent être à nouveau calculées. La seule information partagée est le pourcentage de confiance du détecteur qu'une détection représente en effet un piéton. Pour cette raison, certaines des difficultés mentionnées plus tôt viennent aussi poser problème, mais d'une manière différente. Il faut aussi noter qu'aucun détecteur n'est pas parfait ce qui ajoute plusieurs défis supplémentaires :

- **Fausse détection** : Le premier type d'erreurs commises par le détecteur est d'identifier une partie de l'arrière-plan comme étant un objet d'intérêt (voir la figure 1.2 a). Cette erreur a un effet direct sur la qualité du suivi puisque s'il n'est pas retiré de la liste des détections réelles, un identifiant lui sera attribué ce qui ajoutera une erreur de type faux-positif au résultat de suivi. Une situation de dérive peut également survenir si cette fausse détection se retrouve dans une trajectoire qui suivait un objet valide précédemment. Dans ce cas, la trajectoire pourrait ne pas être en mesure de retrouver l'objet en question et ainsi dériver en commençant le suivi d'un autre objet par la suite.
- **Détection manquante** : À l'opposé des fausses détections, un détecteur peut confondre un véritable objet d'intérêt pour une partie de l'environnement et ainsi ne pas lui apposer de boîte englobante. Cette situation provoque une coupure dans la trajectoire de

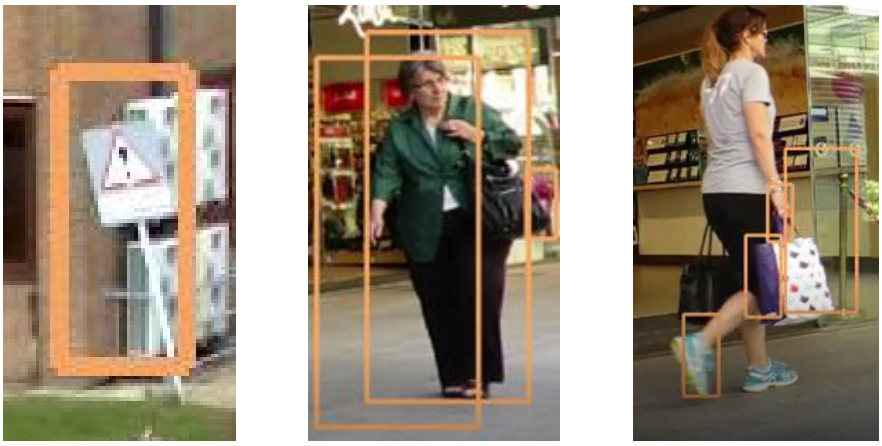
l'objet et si la méthode de suivi n'est pas en mesure de ré-identifier cet objet, il sera traité comme un nouvel objet et la trajectoire finale sera morcelée en deux parties distinctes. En plus des effets de morcellement, une absence de détection cause directement une erreur de type faux négatif.

- **Objet détecté en double** : Un détecteur peut parfois produire par erreur plusieurs boîtes englobantes pour un même piéton (voir la figure 1.2 b). Le nombre d'erreurs de type faux positif augmente donc si aucun mécanisme pour supprimer les doublons n'est en place dans la stratégie de suivi. En revanche, en utilisant un tel mécanisme, il existe un risque de supprimer une véritable boîte dans une situation où deux personnes marchent côte à côte par exemple.
- **Fragmentation** : L'effet de fragmentation se produit lorsqu'un objet d'intérêt est identifié à l'aide de plusieurs sous-boîtes (voir la figure 1.2 c). Contrairement, aux doubles détections, chaque sous-boîte ne couvre pas l'entièreté du piéton, mais plutôt certains membres. En plus d'augmenter le risque d'ajouter de faux positifs, ajouter l'une des sous-boîtes à la trajectoire d'un objet peut poser problème, puisque le modèle d'apparence de l'objet à une trame donnée se base uniquement sur une fraction de l'information disponible. Ainsi, il peut être difficile de reconnaître l'objet complet à son retour la trame suivante.

En plus des difficultés qu'elles apportent au processus de détection, les occlusions sont un défi à gérer pendant la phase d'association. En pire cas, l'objet n'est pas détecté et l'occlusion agit comme une détection manquante. Cependant, il peut arriver qu'un objet masqué soit correctement détecté, ce qui signifie qu'une boîte aura l'aspect de ce qui est à l'avant-plan au lieu de celui de l'objet lui-même. De cette façon, l'identité de l'objet ne sera probablement pas reconnue.

Il faut aussi considéré que les transformations présentées précédemment (illumination, échelle et déformation) impactent aussi le processus de suivi. Ces transformations altèrent l'apparence de l'objet à suivre, ce qui le rend plus difficile à identifier trame après trame.

En plus de ces difficultés inhérentes au problème du suivi multiobjet, certaines séquences vidéos peuvent être plus complexes à traiter que d'autres. Généralement, les vidéos que nous observons ont un taux de trames se situant autour de 30 trames par secondes. Il peut toutefois arriver que certaines séquences aient un taux de trames beaucoup plus faible par souci d'économie de ressources. On dispose ainsi de moins d'information pour faire le suivi, ce qui se traduit par le choix de modèles de mouvement et d'apparence plus souples (de plus grandes distances séparant deux détections consécutives d'un même objet sont tolérées, ainsi qu'une plus forte variation de l'apparence). Il peut également arriver que la caméra ne soit



(a) Fausse détection (b) Détection multiple (c) Fragmentation

Figure 1.2 Types d'erreurs pouvant être commises par le détecteur

pas toujours fixe. Elle peut, par exemple, être positionnée sur un piéton en déplacement.

Il existe donc une variété de situations qui peuvent poser problème à l'exercice de la deuxième phase du suivi. On constate néanmoins que les conséquences de toutes ces situations se résument à trois différents types d'erreurs de base qui surviennent lors de la phase d'association des données en considérant que les objets ont été identifiés par un détecteur imparfait :

1. Associer une détection au mauvais objet ou combiner deux trajectoires différentes en une seule ;
2. Ne pas associer une détection à un objet alors que celui-ci est présent dans la scène (faux négatif) ;
3. Associer une détection représentant une partie de la scène qui n'est pas un objet d'intérêt à une trajectoire (faux positif).

1.2 Nature combinatoire du problème d'associations en suivi

Malgré la possibilité d'avoir des détections manquantes ou en trop, il est possible de représenter cette problématique comme étant le placement d'identifiants de détections dans un tableau à deux dimensions où chaque ligne représente l'identité apposée et où chaque colonne indique la trame dans laquelle se retrouve les détections utilisées. On se retrouve donc en présence d'un problème combinatoire ou l'objectif général recherché est d'obtenir des associations détections/identifiants les plus proches possibles de celles produites manuellement qui constituent la vérité terrain. Or, comme celle-ci n'est pas disponible lors de la résolution, des objectifs secondaires devront être définis. Par exemple, minimiser les variations d'apparence

ainsi que les déplacements improbables.

Une approche adaptée à ce type de problématique est la programmation par contraintes (CP) qui utilise un ensemble de variables à domaine fini ainsi que plusieurs contraintes restreignant le domaine de ces variables. En plus, cet outil permet la définition d'un objectif d'optimisation, si nécessaire. Un solveur pourra ensuite explorer l'ensemble des combinaisons variables/valeurs respectant chaque contrainte en favorisant certaines combinaisons selon la manière de parcourir les solutions possibles qui a été définie. Une brève description de la CP ainsi que des contraintes de haut niveau utilisées dans ce travail seront présentées dans la section 2.3.

1.3 Objectif du mémoire

L'objectif général de cette recherche est de trouver comment tirer avantage des mécanismes offerts par la CP pour effectuer un suivi sur les différents usagers de la route dans plusieurs séquences vidéos en milieu urbain. L'étape de détection des objets d'intérêt a déjà été effectuée, ce qui signifie que la méthode développée prendra en entrée les boîtes englobantes correspondant aux détections, en plus de l'ensemble des images sur lesquelles le suivi doit être effectué. À son terme, le processus produira une liste mise à jour de boîtes englobantes qui auront toutes été associées à un objet d'intérêt.

Voici les grandes étapes qui ont conduit au développement de la méthode présentée dans ce mémoire :

1. Identification des caractéristiques utilisables ;
2. Conception et prototypage du processus à l'aide d'un modèle CP de base ;
3. Conception de filtres sur les entrées et les sorties de la partie CP ;
4. Amélioration du modèle d'apparence ;
5. Définition d'une stratégie de parcours des solutions possibles pour accélérer la résolution ;
6. Optimisation des paramètres de la méthode selon l'ensemble des données à traiter.

1.4 Plan du mémoire

Tout d'abord, les prochaines pages serviront à présenter les différentes méthodes de suivi multiobjet dont celles formant l'état de l'art actuellement. Par la suite, la démarche utilisée lors de cette recherche sera présentée et suivie par un article soumis à la revue *IEEE Transactions on Intelligent Transportation Systems*. La discussion générale qui suivra portera sur

les résultats intermédiaires obtenus lors du développement de cette méthode. Finalement, une conclusion clôturera ce mémoire en rappelant les objectifs de recherche et en indiquant les prochaines étapes pour de futurs travaux sur le sujet.

CHAPITRE 2 REVUE CRITIQUE DE LA LITTÉRATURE

Le suivi multiobjet est un sujet toujours à l'étude et de nouvelles méthodes sont continuellement proposées pour améliorer la qualité du suivi. Les nouvelles avancées en matière d'intelligence artificielle ont fait de cette technologie un outil efficace pour approcher cette problématique au cours des dernières années. Les méthodes à base de réseaux neuronaux convolutifs forment présentement l'état de l'art et ont augmenté significativement la précision et l'exactitude du suivi. Les travaux présentés à la section 2.2 illustrent quelques manières de mettre à profit ces nouvelles technologies.

Avant de présenter ces méthodes de suivi, un préambule sur la détection d'objets permettra de faire la différence entre quelques types de détecteurs qui ont été utilisés dans ce travail. Seulement ensuite, plusieurs exemples de solutions seront détaillées pour chacune des trois composantes du problème du suivi multiobjet. Finalement, la dernière section comportera des notions de base utiles pour le lecteur qui n'est pas familier avec la programmation par contraintes.

2.1 Détection d'objets d'intérêts

Bien que cette recherche ne porte pas sur la partie de détection d'objets, plusieurs détecteurs seront utilisés avec la méthode de suivi développé. Une distinction est généralement effectuée entre les détecteurs qui utilisent des caractéristiques définies à la main tels des points caractéristiques ou des statistiques sur les contours et couleurs présentes et les détecteurs utilisant l'apprentissage profond d'un bout à l'autre du processus de détection. Dans le second cas, les caractéristiques les plus discriminantes sont apprises au cours de la phase d'entraînement et non pré-sélectionnées.

Voici un bref aperçu des deux méthodes qui ont fourni des détections au cours de ce travail :

- **Fast Feature Pyramids** : Il s'agit de la méthode qui a été utilisée pour générer les détections publiques du défi 2D MOT 2015 [1]. Pour obtenir les boîtes englobantes autour des objets d'intérêt, cette méthode calcule le gradient selon plusieurs échelles ou tailles d'images pour en faire des histogrammes (voir le descripteur HOG à la section 2.2.2 pour plus de détails à ce niveau). Combiné à des caractéristiques basées sur la couleur, il est possible d'utiliser n'importe quel modèle de classification supervisé. La machine à support de vecteurs (SVM) permettant de distinguer les vecteurs caractéristiques de piétons de ceux de l'environnement [2] est utilisée comme classificateur dans le cas du

2D MOT 2015.

- **Region proposal with Convolutional Neural Network (R-CNN)** : Cette approche populaire a permis de grandes améliorations à son arrivée dans le domaine de la détection d'objets. Le processus se divise en deux étapes. La recherche de régions (R) où pourraient potentiellement se trouver des objets d'intérêt et l'extraction de caractéristiques à l'aide des réseaux neuronaux convolutifs (CNN) [3]. Plusieurs améliorations ont été apportées à cette technologie de manière à la rendre plus efficace à la fois au niveau du temps d'opération que pour la précision des résultats [4, 5].

2.2 Méthodes de suivi

La présente section présentera les caractéristiques des différentes méthodes pour effectuer du suivi multiobjet. Tel que mentionné dans le chapitre précédent une méthode de suivi peut être divisée en trois composantes. Ces trois composantes seront étudiées séparément.

2.2.1 Modèles de mouvements

Comme le suivi s'effectue à partir de séquences vidéos qui ont bien souvent un taux de trames élevé, aucun objet en déplacement n'est en mesure de franchir une grande distance entre deux trames successives. Ainsi, utiliser la distance pour identifier un objet dans une trame ultérieure permet d'éliminer un certain nombre de possibilités puisqu'on ne considère qu'une sous-région de l'image. Évidemment, il y a moyen de faire beaucoup plus à partir des informations pouvant être extraites de la position et la taille des détections à associer.

Les modèles de positionnement les plus simples se contentent de prédire que la prochaine position d'un objet sera la même que celle occupée lors de sa dernière apparition. On peut ensuite utiliser cette prédiction pour accorder une priorité aux détections de la trame courante les plus proches de la position référence de la trame précédente [6].

Une manière de faire plus réaliste est de considérer que l'objet se déplace linéairement, à vitesse constante. Il s'agit d'une approche fréquemment utilisée, malgré le fait qu'elle ne modélise pas tout à fait le véritable comportement d'usagers de la route qui peuvent interrompre brusquement leur trajectoire par un virage, un arrêt ou un changement de direction. Cette approche a été utilisée dans différents travaux [7, 8].

Pour prendre en comptes ces variations de trajectoires, l'utilisation d'un filtre de Kalman a été étudiée [9]. Le filtre de Kalman gère la position et la vitesse comme étant deux variables aléatoires décrites généralement pas des équations du mouvements à chaque pas de temps suivant chacune une distribution normale. On peut ensuite calculer la matrice de covariance

pour estimer à la fois la position et la vitesse réelle de chaque détection. La méthode par filtre de corrélation (KCF) peut aussi être utilisé pour le suivi [10]. Après avoir analyser une région de l'image à suivre, l'objectif est de retrouver celle-ci dans les trames subséquentes. Afin d'y arriver, un filtre est estimé à l'aide de l'image originale. Par la suite, un sous-échantillon de régions d'image est aléatoirement formé et chaque région est multipliée avec le filtre. Plus le résultat de l'opération est élevé, plus la probabilité d'avoir trouvé la bonne région de l'image est élevée. Cette méthode est généralement utilisée pour le suivi d'un seul objet.

Présentement, les méthodes les plus performantes pour modéliser le mouvement sont celles produisant des prédictions basé sur des techniques d'apprentissage. L'idée d'exploiter les forces de l'apprentissage par renforcement pour effectuer des prédictions sur la position a été étudiée à plusieurs reprises [11, 12]. La manière de procéder est d'utiliser un réseau neuronal convolutif qui utilise la sous-région de l'image contenue dans une boîte englobante et sa position initiale comme donnée d'entrée. Le réseau doit être en mesure de prédire les variations des dimensions de la boîte englobante pour la trame suivante, ce qui correspond à prédire la position de l'objet. Ce module est intégré à un algorithme d'apprentissage par renforcement et l'entraînement vient accroître la précision du modèle de mouvements [12].

Certaines méthodes font le choix de combiner plusieurs modèles de mouvements selon le type de séquences vidéos qui doit être traité. Par exemple, en présence de séquences avec caméras en déplacement, une méthode pour compenser le mouvement de la scène à l'aide de transformations affines est appliquée. Concernant les autres types de séquences, y compris pour celles à faible taux de trames par seconde, l'hypothèse que l'objet se déplace à vitesse constante est considérée [13].

2.2.2 Modèles d'apparence

Si le modèle de mouvements permet de donner une idée de la probabilité d'appartenance d'une détection à une certaine trajectoire, le modèle d'apparence a pour principale fonction la description de l'objet et à ce titre, se base sur toutes caractéristiques visuelles disponibles pour faire une description discriminante, mais suffisamment robuste pour reconnaître l'objet dans différentes conditions. Il existe plusieurs manières de décrire l'apparence d'un objet dans une image. On peut se baser sur sa forme ou son contour, les textures discernables sur sa surface ainsi que les principales couleurs le composant.

Historiquement, outre les informations provenant de la boîte englobante (taille de l'objet), les manières traditionnelles de représenter un objet étaient la conception manuelle de descripteurs qui utilisaient chacun un seul type de caractéristiques.

L'histogramme de couleurs est un exemple de descripteur qui se base uniquement sur la répartition des couleurs dans une image. Une méthode de suivi peut se baser sur plusieurs de ces descripteurs pour produire une description de plus grande qualité. Plusieurs approches utilisent une variété de combinaisons pouvant inclure les histogrammes de couleurs [14, 15], mais également la matrice de covariance et les histogrammes de gradients :

- **Histogrammes de couleur** : Il s'agit d'une manière simple de déterminer la couleur présente dans une région d'intérêt de l'image. La première étape pour obtenir l'histogramme est la déclaration d'un tableau où chaque case est associée à une valeur de pixel possible. Chaque case est initialisée à zéro. Pour chaque pixel de la région de l'image à évaluer, on incrémente de un la case du tableau associée à la valeur du pixel en question de manière à obtenir le compte de chaque occurrence des valeurs de pixels se retrouvant dans la région. Pour les images couleurs, qui possèdent généralement trois canaux, ce traitement est effectué trois fois, ce qui signifie qu'un histogramme par canal est produit. Dans ce cas, les valeurs des pixels correspondent aux valeurs des R, G, et B qui leurs sont associées. Pour éviter d'être trop précis dans sa description et donc peu robuste aux différentes variations dans l'image, un histogramme peut être quantifié, c'est-à-dire que les valeurs individuelles sont regroupées en intervalle de telle sorte que le résultat est un histogramme plus approximatif.
- **Histogrammes des orientations du gradient (HOG)** : Une manière de représenter la forme ou les contours d'un objet d'intérêt est de calculer l'histogramme des orientations du gradient de la région de l'image délimitée par sa boîte englobante. La première étape pour obtenir une description est de calculer le gradient à l'aide de convolutions entre chaque pixel de la sous-région et un noyau de convolution comme celui de Sobel. Ensuite, des regroupements carrés de pixels sont formés et chaque valeur d'orientation du gradient est comptabilisée et pondérée selon la force de celui-ci. Après une normalisation entre regroupements de pixels, un vecteur décrivant la forme de chaque région de la détection est obtenu. La figure 2.1 permet de visualiser le résultat du calcul du gradient pour quelques détections.
- **Matrice de covariances** : Cette technique de description est utilisée pour modéliser les textures se retrouvant dans une image [16]. Le descripteur C_k est défini par l'équation suivante où z_k correspond à un vecteur contenant les deux dérivées premières et trois dérivées secondes de l'image par rapport à x et y ainsi que le vecteur moyen μ de la région à décrire :

$$C_k = \frac{1}{n-1} \sum_{k=1}^{n} (z_k - \mu)(z_k - \mu)^T \tag{2.1}$$

Figure 2.1 Exemples de détections (images de gauche) et d'une représentation de leurs gradients (images de droite)

Plus récemment, les méthodes par apprentissage profond ont, ici aussi, surpassé la création manuelle de descripteurs en termes de résultats. De manière générale, cela signifie qu'un réseau de neurones est entraîné à découvrir quelles sont les caractéristiques les plus utiles pour reconnaître une identité de manière à former un vecteur de caractéristiques profondes. Ces caractéristiques sont obtenues à l'aide d'un réseau neuronal convolutif. Typiquement, cette architecture utilise des couches successives de convolutions sur une région d'image qui baisse en résolution à chaque couche. Cela a pour effet d'amasser des caractéristiques à différents niveaux. Un vecteur contiendra la concaténation de tous les résultats obtenus aux différentes couches pour décrire la région d'intérêt. Plusieurs méthodes récentes utilisent ces caractéristiques [17, 18].

Une approche gagnant en popularité à l'heure actuelle pour caractériser un objet est d'utiliser des caractéristiques de méthodes de ré-identification pour le décrire [13, 19]. Dans cette situation, l'objectif est de reconnaître un objet d'intérêt, bien souvent humain, à partir de différentes caméras réparties sur un territoire. Il ne faut pas perdre de vue la personne, particulièrement lorsqu'elle quitte la région couverte par une caméra pour passer à une zone couverte par une autre. Les caractéristiques extraites par les méthodes de ré-identification sont dans la même catégorie que les caractéristiques profondes ; la similarité est établie en comparant deux vecteurs de descriptions. Plusieurs méthodes utilisant cette approche en tant que modèle d'apparence seront présentées prochainement. Une approche testée a été d'entraîner le réseau de neurones siamois TriNet à l'aide des détections de référence (*ground truth*) incluses dans l'ensemble de données *MOT17 tracking*. Lors de l'apprentissage, une technique pour bonifier l'ensemble des données disponibles pour l'entraînement est de transformer les détections existantes à l'aide de renversement horizontaux et de recadrages et de les ajouter en tant que nouvelles détections sur lesquels s'exercer [13].

2.2.3 Association des données

Une fois les descriptions de tous les objets d'intérêts calculées, il faut maintenant les utiliser pour décider si une association entre deux de ces objets est valide ou non de manière à former en ensemble de trajectoires dans lequel chaque détection ne se retrouve qu'une seule fois. La figure 2.2 illustre les possibilités d'association ainsi que le résultat valide pour ce sous-problème (en jaune). Les méthodes classiques, sans apprentissage, se divisent en deux classes. L'une contient les algorithmes gloutons, peu avare en temps de calculs, mais qui peinent à atteindre une performance de suivi satisfaisante dans les situations où la densité d'objets est trop élevé, par exemple une foule où les occlusions sont fréquentes. L'autre classe contient les approches qui considèrent chaque association possible, ce qui augmente la probabilité de faire de bons choix, mais qui transforme la tâche d'association en un problème combinatoire de la classe de complexité NP difficile [20].

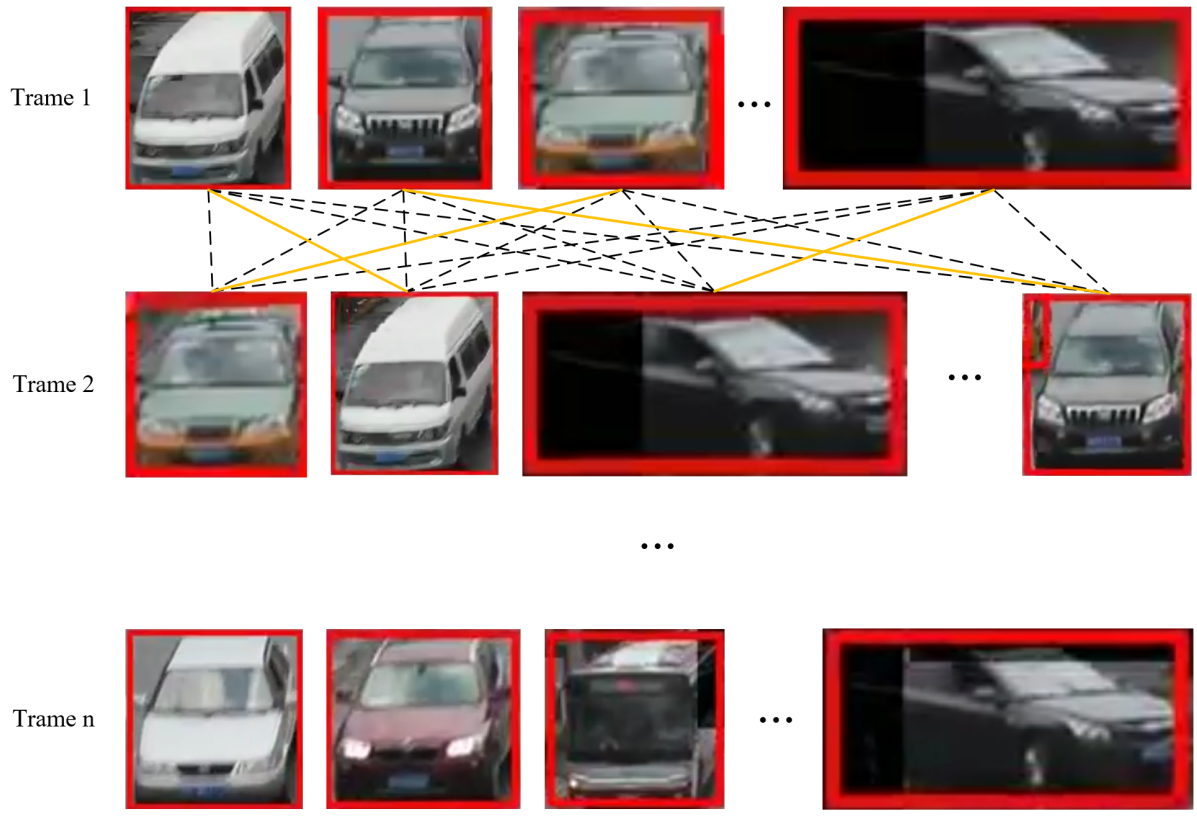


Figure 2.2 Illustration du problème d'association des données. Les objets peuvent sortir et entrer dans la scène d'où le fait que les objets de la trame n ne correspondent pas tous à ceux des deux premières trames.

Une méthode simple pour effectuer ces associations est l'utilisation de l'algorithme hongrois. De manière générale, le problème résolu par cet algorithme glouton est la recherche des

meilleures associations entre deux ensembles de telle sorte que la somme des coûts pour chaque association soit minimale. La résolution se fait essentiellement en prenant les associations qui sont les plus avantageuses localement en premier, à condition qu'il n'en existe qu'une seule. Puis, la taille du problème est réduite en éliminant les valeurs déjà intégrées à la solution finale et l'étape précédente est effectuée à nouveau [21]. Plus spécifiquement, dans le cas du suivi multiobjet, il s'agit de construire une matrice listant les détections d'une première trame dans une dimension et celles de la suivante dans l'autre dimension.

L'approche statistique nommée *Joint Probabilistic Data Association Filter (JPDAF)* sélectionne les associations à faire selon des probabilités calculées entre chaque paires possibles des détections de trames consécutives tout en prenant en compte la possibilité que certaines détections soient manquantes ou en trop. Cette technique a été utilisée dans plusieurs travaux dont les suivants [22, 23]. La version Monte-Carlo utilise un filtre de particules qui permet d'estimer la probabilité qu'une détection soit à une certaine position à une trame t sachant l'historique de la trajectoire. Il est ensuite possible de construire la distribution des probabilités pour faire un choix [24].

De plus en plus, certaines méthodes arrivent à un résultat satisfaisant en appliquant des méthodes de suivi d'un seul objet (SOT) dans une situation de MOT. Le problème du suivi d'un seul objet se résume à donner une séquence d'images, en plus d'une seule boîte englobante manuellement formée pour indiquer sur la première trame quel est l'objet d'intérêt. En présence de plusieurs objets, le défi avec cette approche n'est plus de résoudre une suite de graphes bipartis, mais de suivre individuellement chaque objet présent. Historiquement, cette manière de procéder était problématique ; les objets d'intérêts étant similaires, les méthodes de suivi individuel étaient sujettes à les confondre et il devenait difficile de s'assurer que chaque objet était bel et bien suivi. Pour pallier à cette faiblesse, les méthodes de suivi d'un seul objet ont été appliquées dans un contexte restreint, soit pour séparer les objets d'intérêt de l'arrière-plan et pour bien identifier les différences entre chaque objet [25]. Avec ces informations, il est possible d'établir un système de prédictions (tel que décrit dans les modèles de mouvement) qui sera intégré à une méthode d'association des données considérant tous les objets à la fois [19].

Finalement, l'utilisation de réseaux de neurones récurrents (RNN) s'est avérée efficace pour l'association de données [20] en suivi multiobjet. Cette méthode utilise des cellules LSTM qui sont en mesure de retenir certaines informations provenant des trames traitées plus tôt, ce qui permet au réseau de neurones non seulement d'avoir la détection courante en entrée, mais également des réminiscences des précédentes tentatives d'association effectuées par le réseau. Plus précisément, le RNN est en mesure d'entreposer des informations basées sur

les précédentes associations effectuées, ce qui permet d'obtenir des probabilités d'association entre trajectoires et détections en se basant sur les informations de la trame courante, mais aussi l'état caché qui est calculé et mis à jour au fur et à mesure du processus de suivi.

2.3 Concepts de base en programmation par contraintes

De manière plus formelle, la programmation par contraintes est avant tout une méthode pour résoudre les problèmes de satisfaction de contraintes. Ce type de problème se formule à l'aide d'un triplet $\langle X, D, C\rangle$ où X représente un ensemble fini de variables, D, un ensemble fini de valeurs possibles pour ces variables et C, un ensemble fini de contraintes s'appliquant sur ces variables. L'objectif est de trouver la valeur à associer à chaque variable de manière à satisfaire l'ensemble des contraintes définies et obtenir une solution.

En plus des problèmes de satisfaction de contraintes, la CP contient des mécanismes pour supporter les problèmes d'optimisation. Ainsi, une fonction s'ajoute au triplet $\langle X, D, C\rangle$ pour représenter l'objectif devant soit être minimisé, soit maximisé. Le problème se définit maintenant ainsi : $P = \langle X, D, C, f\rangle$ où $f : D(x_1) \times D(x_2) \times \cdots \times D(x_n) \to \mathbb{R}$ est la fonction objectif, trouver une affectation

$$\tau : x \in X \mapsto v \in D(x)$$

telle que

$$\forall c \in C \quad \tau(X(c)) \in c$$

$$f(\tau(X)) \text{ est minimale.}$$

On transforme souvent P en $P' = \langle X \cup \{z\}, D', C \cup \{z = f(X)\}, min(z)\rangle$ où z est une variable de coût qu'on cherche à minimiser.

Une fois le problème modélisé, le solveur CP génère un arbre contenant les différentes solutions possibles au problème. Chaque contrainte est automatiquement propagée par le solveur selon le niveau de cohérence qui lui est associé de manière à filtrer les valeurs du domaine qui violent le principe de cohérence locale. Comme plusieurs contraintes peuvent s'appliquer sur la même variable, il est souvent nécessaire de considérer l'ensemble des contraintes pour retirer chaque incohérence, c'est-à-dire les valeurs du domaine qui ne sont acceptés que dans un sous-ensemble strict des contraintes. Différents niveaux de cohérence peuvent être appliqués pour effectuer le filtrage dont les principaux sont ceux-ci :

- **Cohérence d'arcs** : Ici, chaque contrainte binaire (s'appliquant sur deux variables) est examinée de manière à s'assurer que pour toute valeur assignée à la première variable, il

en existe au moins une possible pour la seconde et inversement. Dans le cas contraire, la valeur sans équivalent est filtrée. Par exemple, dans le cas de la contrainte arithmétique suivante : $x + y = 15, D(x) = 1, 2, 3, 4, 5, D(y) = 9, 10, 11$, y ne peut pas prendre la valeur 9, car aucune valeur du domaine de x ne permet de satisfaire la contrainte.

- **Cohérence d'arcs généralisée ou cohérence de domaine** : Ce type de cohérence est utile aux contraintes non-binaires telles que *All-Different* et *Regular*. Le principe de la cohérence d'arcs est appliqué sur des hyper-arcs où pour chaque valeur que peut prendre une variable, il doit exister au moins une possibilité de valeurs pour toutes les autres variables de l'hyper-arc.
- **Cohérence de bornes** : Il peut arriver qu'une variable contienne un domaine trop large pour qu'il soit efficace de noter si chaque valeur de son domaine est possible ou non. Dans ces situations, la cohérence de bornes permet de filtrer les valeurs en travaillant avec les valeurs minimales et maximales pouvant être prises par la variable plutôt que l'ensemble des valeurs possibles ou impossibles.

Une fois qu'on obtient les domaines filtrés pour toutes les variables et qu'aucune propagation n'est possible, le solveur doit prendre une décision, c'est-à-dire retirer une valeur du domaine ou fixer une variable en retirant une valeur qui pourrait faire partie d'une solution. Ce type de décision se définit comme étant un branchement dans l'arbre des solutions possibles. Trouver une stratégie de branchement efficace est donc impératif pour obtenir une solution au problème rapidement.

L'arbre des solutions est la représentation utilisée par un solveur CP pour les possibilités d'assignation variable/valeur qui sont chacune représentée par un noeud de l'arbre. Une solution au problème correspond à un chemin de la racine jusqu'à une feuille où chaque variable est assignée à une valeur ne violant aucune contrainte. Tous les parents directs d'un noeud indiquent des assignations ajoutées préalablement à la solution. Lorsque le solveur se retrouve dans un noeud sans issue, l'assignation du noeud en question est filtrée de l'arbre des solutions. Il est alors possible de faire une remontée d'un niveau, ce qui signifie annuler la dernière assignation effectuée, ou choisir un tout autre chemin à partir de la racine. Une stratégie de branchement est la référence utilisée pour prendre ce type de décision, c'est-à-dire guider le solveur dans son parcours de l'arbre des solutions.

À l'heure actuelle, modéliser un problème en utilisant la programmation par contraintes n'est pas une tâche triviale qui peut être effectuée par un non-spécialiste. Plusieurs manières différentes existent pour modéliser un problème et elles ne sont pas toutes équivalentes. En effet, il faut considérer que certaines contraintes se propagent mieux que d'autres et doivent être privilégiées lorsque possible, alors que d'autres, au niveau de cohérence plus faible,

doivent être évitées. Il faut également être attentif au choix de variables puisque utiliser un nombre de variables plus grand que nécessaire augmente le nombre de possibilités et ainsi augmente inutilement la taille de l'arbre de solutions.

Une situation particulièrement problématique lorsqu'on modélise un problème par la CP est d'inclure des symétries dans le modèle. Des symétries sont présentes lorsque le choix de variables est effectué de telle sorte que plusieurs combinaisons de variables/valeurs conduisent à une même solution. Par exemple, dans le problème des n-reines où l'on cherche à placer n reines dans un échiquier de taille $n \times n$ de telle sorte qu'aucune reine ne soit menacée par les autres, n'importe quelle permutation entre chaque reine sur le plateau ne produit pas de nouvelle solution (voir la Figure 2.3). La présence de symétries est problématique puisque le solveur peut dépenser beaucoup d'énergie dans une zone de l'arbre contenant un grand nombre de solutions erronées symétriques au lieu d'explorer des parties plus intéressantes de l'arbre de solutions.

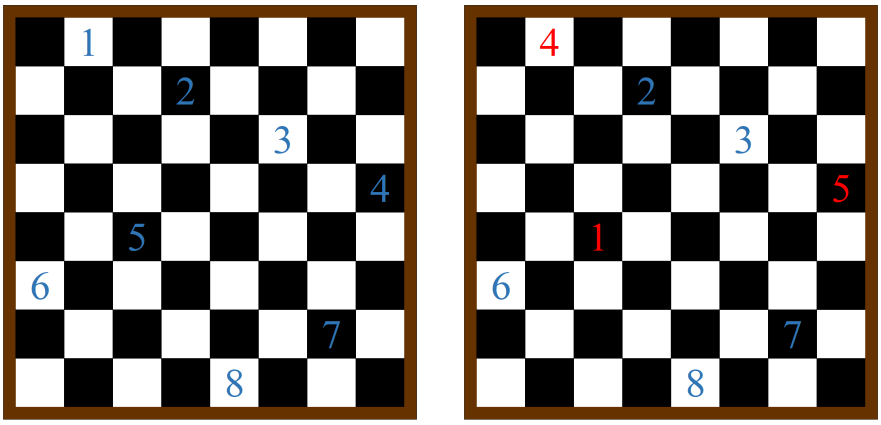


Figure 2.3 Exemple de deux solutions symétriques pour le problème des n-reines. Chaque chiffre correspond à l'identité d'une reine. Toute permutation de ces identités produisent une seule et même solution.

Pour mieux comprendre le fonctionnement de la programmation par contraintes, le lecteur est référé à l'article suivant [26].

2.3.1 Contraintes de haut niveau

Un des avantages offerts par la CP est l'utilisation de contraintes de haut niveau dites *globales* qui permettent de modéliser rapidement un problème, tout en offrant un niveau de cohérence plus global que celui qui serait obtenu par l'utilisation d'une décomposition en contraintes plus simples pour arriver au même résultat.

Voici la description de quelques-unes de ces contraintes de haut niveau qui sont utilisés dans cette recherche :

1. ***All-Different*** : Cette contrainte s'applique à un tableau de variables et stipule que chaque variable doit être fixée à une valeur distincte de celle des autres variables. Ainsi, il est impossible de retrouver deux fois la même valeur dans l'ensemble de variables spécifiées. C'est la contrainte typique du problème du Sudoku, où se retrouve une contrainte *all-different* pour chaque ligne, colonne et secteur (bloc carré de neuf cases). Formellement, voici les équations de cette contrainte pour un tableau de variables x de taille n :

$$x_i \neq x_j \qquad \forall\ 1 \leq i < j \leq n \tag{2.2}$$

   Considérer les variables par séquences de paires peut être problématique au niveau de la propagation. Prenons l'exemple suivant où x a trois cases et où chacune a un domaine de $\{1, 2\}$. Chaque sous-équation de la contrainte peut être respectée :

   (a) $x_0 \neq x_1$: Si x_0 prend la valeur 1, x_1 prendra la valeur 2 et inversement ;

   (b) $x_1 \neq x_2$: Si x_1 prend la valeur 1, x_2 prendra la valeur 2 et inversement ;

   (c) $x_0 \neq x_2$: Si x_0 prend la valeur 1, x_2 prendra la valeur 2 et inversement.

   Néanmoins, il est évident qu'aucune solution n'existe à ce problème puisque le domaine est trop restreint et ce, même si la contrainte *All-Different* n'est pas en mesure de le déterminer d'elle-même. Dans cet exemple, il faudra tester tous les branchements possibles pour réaliser l'absence de solution existante. Cette faiblesse a toutefois été traitée à l'aide d'un nouvel algorithme de filtrage assurant la cohérence d'arcs généralisée [27].

2. ***Inverse*** : Pour deux tableaux de variables CP $\langle x_1, x_2, \ldots, x_m \rangle$ et $\langle y_1, y_2, \ldots, y_n \rangle$ de domaines $[1, m]$ et $[1, n]$ respectivement, la contrainte *inverse* s'assure de respecter les équations suivantes :

$$x_i \in [1, n] \Rightarrow y_{x_i} = i \qquad \forall\ 1 \leq i \leq m \tag{2.3}$$

$$y_j \in [1, m] \Rightarrow x_{y_j} = j \qquad \forall\ 1 \leq j \leq n \tag{2.4}$$

   Elle peut être utilisée de manière asymétrique ; les valeurs du domaine de la fonction qui sont plus grandes que le nombre de variables spécifiées dans la fonction inverse sont ignorées par la contrainte. Sa principale fonction est de maintenir la cohérence lorsque deux représentations complémentaires sont exprimées dans un modèle CP, c'est-à-dire que les indices et les valeurs du tableau x se retrouvent inversés dans le tableau y

3. ***Regular*** : Ces contraintes servent à exprimer l'appartenance d'une suite de caractères à un langage régulier. Il s'agit donc du mécanisme qui permet d'intégrer les automates à états finis dans la programmation par contraintes. Pour une certaine suite de variables $\langle x_1, x_2, \ldots, x_n \rangle$ et un automate $\mathcal{A}$, la contrainte *Regular* s'assure que toutes les valeurs assignées à ces variables correspondent à un mot reconnu par l'automate $\mathcal{A}$ [28]. Cette contrainte peut être adaptée aux problèmes d'optimisation et devenir ainsi la contrainte *CostRegular*. Pour ce faire, une matrice de coûts prend en compte les combinaisons entre variables et valeurs, en plus de l'état courant dans lequel se situe l'automate. Tous les coûts observés pour une solution sont additionnés et le résultat de cette somme peut être utilisé comme objectif de minimisation [29].

2.3.2 Stratégies de branchement

La sélection d'une stratégie de branchement a une grande influence sur le temps de résolution du problème. On peut aussi évaluer ce paramètre selon le nombre total de branchements effectués lors de la résolution puisque qu'une bonne stratégie conduira à une bonne solution plus rapidement. Un échec se définit par une situation dans laquelle le solveur teste un branchement qui conduit à un sous-espace de l'arbre de recherche ne contenant aucune solution.

Les branchements s'effectuent toujours selon un parcours en profondeur de l'arbre de solutions. Ainsi, tant qu'une association variable/valeur conduit à un sous-arbre de solutions en contenant au moins une, cette dernière n'est pas remise en question. Une manière fréquente de sélectionner les branchements à effectuer est de spécifier manuellement l'ordre selon lequel tester les variables ou les valeurs. Le cas échéant, on parle d'un ordre lexicographique.

Voici quelques stratégies de recherche qui ont été utilisées dans le cadre de cette recherche :

- **Plus petit domaine** : Utilisée comme stratégie par défaut dans les premières versions de la librairie ILOG, elle est généralement en mesure de donner une solution assez rapidement pour la plupart des modèles CP. La taille du domaine des variables permet de déterminer celles qui seront testées en premier, les petits domaines étant privilégiés. Quant au choix des valeurs, celui-ci dépend d'un ordre lexicographique favorisant les petites valeurs en premier.
- **Lexicographique** : La stratégie de branchement lexicographique combine l'utilisation d'un ordre lexicographique pour le choix des variables et pour celui des valeurs. Cette stratégie peut s'avérer utile lorsque les caractéristiques du problème à résoudre sont plus utiles que les informations dont dispose le solveur pour se diriger dans l'arbre de

solutions.

- **Heuristique de densité maximale pondérée de solutions (*CostMaxSD*)** : Dans cette stratégie heuristique, les branches de l'arbre contenant le plus grand nombre de solutions sont mises à l'avant pour des contraintes avec coûts comme *CostRegular*. Il faut tout d'abord effectuer le filtrage du domaine à l'aide d'un graphe où chaque noeud est une solution partielle et chaque arc est une association variable/valeur. Si en partant de la solution vide, il n'existe aucun chemin menant à une solution complète en passant par un arc sélectionné, cet arc peut être retiré du graphe. L'avantage d'utiliser un tel graphe est la possibilité de pouvoir calculer le nombre de solutions existantes pour chaque choix d'associations effectué. Cependant, en présence d'un problème d'optimisation, chaque solution n'est pas équivalente. On introduit les coûts au calcul de solution en ne considérant que les solutions dont le coût total est dans une certaine marge du coût minimal obtenu en traversant le graphe pour compter le nombre de solutions. Lorsque le filtrage sera complété et qu'un branchement devra être effectué, le solveur a ainsi suffisamment d'information pour choisir l'association de laquelle découlera le plus grand nombre de solutions avantageuses [30].

Une stratégie d'élimination statique de symétries utilisée dans cette recherche provient d'un article de Van Hentenryck et Michel [31]. Ce dernier présente une stratégie de branchement pour le problème d'affectation de commandes à des dalles de capacité fixe dans une aciérie. L'objectif est de remplir à pleine capacité chacune de ces dalles et ainsi d'utiliser un nombre minimal de dalles. Pour ce faire, un branchement lexicographique est fait sur les plus petites commandes qui peuvent seulement être assignées aux dalles précédemment utilisées en plus d'une seule nouvelle, ce qui permet d'éviter les symétries inutiles où chaque dalle inutilisée peut être interchangée avec les autres sans modifier pour autant la valeur de la solution.

CHAPITRE 3 DÉMARCHE DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE

Dans cette section sera présentée l'évolution du développement de la méthode tout au long de la recherche. Comme il n'y a qu'un seul article et que celui-ci a été écrit vers la fin de ce projet, à lui seul il résume l'ensemble des choix de cette méthode qui ont eu les meilleurs résultats une fois toutes les améliorations apportées. Au lieu de suivre chronologiquement toutes les évolutions de la méthode, les améliorations sont présentées selon la raison les motivant.

3.1 Réflexions initiales sur le problème

Après quelques recherches sur les sujets du suivi et de la programmation par contraintes, l'objectif était de produire un prototype de méthode de suivi afin d'en savoir plus sur la facilité de modélisation du problème. Voici les trois décisions à prendre qui étaient les plus importantes à cette première étape :

1. Quel solveur CP utiliser ?
2. Quel ensemble de données utiliser ?
3. Quelles caractéristiques utilisées pour faire les associations ?

Le choix du solveur utilisé dans le cadre de cette recherche s'est porté sur celui contenu dans la librairie d'optimisation ILOG offerte par IBM principalement pour les deux raisons suivantes :

- La présence des contraintes de haut-niveau telle que *AllDifferent* et *Regular*, en plus des stratégies de branchements basées sur le dénombrement comme *CostMaxSD* qui ont été identifiées comme potentiellement utiles. Certaines contraintes et ces stratégies de branchement sont toutefois disponibles grâce à des ajouts apportés à la librairie originale.
- La simplicité dans la manière d'exprimer les contraintes. Beaucoup de tâches, par exemple la gestion des propagateurs de chaque contrainte, sont gérées de manière automatique par le solveur. Comme il s'agit d'un problème nouveau pour la CP, la facilité de modélisation est un avantage pour construire des prototypes.

Parallèlement au choix du solveur, il était nécessaire de choisir quelles bases de données seraient utilisées. La plupart se présentent sous la forme de concours (*benchmark*) où sont accessibles deux ensembles de séquences, l'une dédiée à l'entraînement de la méthode et l'autre servant uniquement à évaluer la méthode. Pour s'assurer du respect des objectifs des deux

ensembles, ces bases de données donnent habituellement les résultats d'associations corrects uniquement pour l'ensemble d'entraînement. Autrement pour l'ensemble de test, seules les images de chaque trame et une ou plusieurs séries de détections dites publiques sont fournies. Une autre possibilité est de générer les détections soi-même.

Cependant, malgré la portée de se projet qui se veut une méthode de suivi pour tous les usagers de la route, aucune base de données satisfaisante combinant à la fois piétons et véhicules motorisés n'a été trouvée. Le choix initial a ainsi été le défi 2D MOT 15 inclus dans le MOTChallenge et qui ne traite que de piétons [1]. Pour chaque détection voici les informations fournies par le détecteur :

1. Le numéro de trame ;
2. Les coordonnées du coin supérieur gauche de la boîte englobante ;
3. La longueur et la largeur de la boîte ;
4. Le pourcentage de confiance du détecteur que l'objet est bien un piéton.

Finalement, nous avons réfléchi au choix des caractéristiques. Avant d'extraire certaines caractéristiques de l'image, les informations de position et de taille de chaque objet semblaient plus simples à utiliser pour construire un premier modèle. La manière la plus directe pour exprimer des contraintes sur ces renseignements était de les exprimer sous la forme de contraintes dures où la différence de position entre deux détections de trames consécutives d'une même trajectoire ne peut excéder un seuil à déterminer expérimentalement. En ce qui concerne la position, la formulation formelle de cette modélisation se trouve à la section 4.4.1. L'utilisation de la taille de boîte comme caractéristique a été abandonnée. Les résultats étaient moins bons lorsque les contraintes de taille étaient activées.

Une fois toutes ces décisions prises, il a été possible de modéliser le problème d'associations d'objets et ainsi d'obtenir un premier prototype.

3.2 Diminution du nombre de mauvaises associations

Avec un modèle CP qui ne considère que la position de chaque détection, il est difficile d'obtenir des résultats satisfaisants particulièrement au niveau des fausses associations. Pour pallier ce problème, l'approche a été de restreindre davantage les possibilités d'association par l'usage de contraintes supplémentaires ainsi que la promotion d'associations plus probables, ce qui peut être accompli à l'aide de coûts pour chaque association ainsi que par l'ajout d'un objectif d'optimisation.

La première manière d'étoffer le modèle a été le calcul des histogrammes de couleurs pour

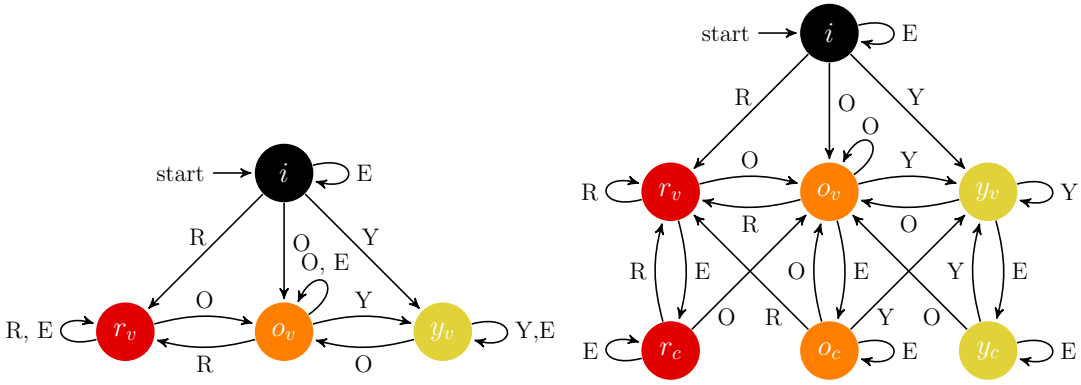
chaque détection et son intégration dans des contraintes dures de la même façon qu'avec la position. Cependant, comparer chaque case du vecteur histogramme pour chaque association testée s'est avéré trop coûteux en temps de calcul pour que cette approche soit viable, y compris avec quantification de l'histogramme.

De plus, à ce stade, les occlusions étaient difficilement gérées par la CP ; les contraintes de positions étaient simplement élargies en multipliant le seuil original par un facteur proportionnel au nombre de trames manquantes entre deux détections réelles consécutives. Pour éviter les cas où le solveur préférerait alterner absences de détection et détections réelles, des contraintes supplémentaires ont été ajoutées. Pour chaque trajectoire, le nombre de paires de trames consécutives où se retrouve une transition (soit détection vers occlusion ou occlusion vers détection) est additionné. L'objectif de minimisation est défini comme étant le maximum de ces nombres de transitions par trajectoire.

Cet objectif de minimisation a toutefois été écarté au profit d'un autre, plus général, qui se base sur l'apparence des détections à l'aide de la contrainte *Regular*. En appliquant un automate sur chaque trajectoire (voir section 4.4.1), il est possible de préserver l'apparence générale de chaque objet tout au long du processus de suivi. Toutefois, un premier modèle d'automate précédent celui présenté dans l'article utilisait une stratégie différente pour gérer les occlusions. Dans le cas d'une transition d'une détection vers une occlusion le coût appliqué est le même pour les deux automates. Or, pour le premier automate (A1), la transition correspondra à une boucle sur le même état. L'effet immédiat est que lors d'une absence répétée de plusieurs détections consécutivement, le coût d'occlusion sera appliqué à chaque transition individuelle, contrairement au second automate (A2) où ce coût ne sera appliqué qu'à la première transition. Une comparaison de ces deux modèles d'automate est illustrée sur la figure 3.1 à l'aide d'un exemple réduit à seulement trois couleurs.

Considérer seulement un ensemble de couleurs réduit permet d'éviter d'utiliser un trop grand nombre de possibilités (255 valeurs possibles pour trois canaux distincts), ce qui augmenterait la complexité de calcul et ralentirait la résolution. De plus, grouper les couleurs en classes a l'avantage de mieux gérer les variations de luminosité ; une même couleur générique sera attribuée à une détection pour des variations faibles. Ainsi, les états possibles de l'automate se baseront sur un petit ensemble de classes de couleurs. L'exemple de la figure 3.1 en utilise 3.

Pour définir les classes de couleurs à utiliser par l'automate, l'algorithme de Lloyd-Max a été utilisé. Il s'agit d'un algorithme pour résoudre le problème des K-Moyennes où, pour un ensemble de points répartis d'une façon quelconque, il faut trouver la meilleure séparation de ces points en k classes distinctes. Plus la séparation est réussie, plus la somme des distances



(a) Premier automate (A1)

(b) Deuxième automate (A2)

Figure 3.1 Comparaison entre les deux modèles d'automate de couleurs à l'aide d'un exemple réduit. Trois classes de couleurs pour représenter l'apparence sont possibles sont possibles : jaune (y), orange (o) et rouge (r). Les transitions correspondent à la couleur de la prochaine détection d'une trajectoire, soit jaune (Y), orange (O) et rouge (R). Dans le cas où aucune détection ne se retrouve dans la trajectoire à un moment donné, la transition d'absence (E) est utilisée. Un état initial sans description de l'apparence est utilisé pour amorcer chaque trajectoire.

entre le centre de ces classes et chaque point lui appartenant sera faible. Afin de résoudre ce problème, voici les étapes proposées par l'algorithme de Lloyd-Max :

1. Placer aléatoirement chaque centre de classe dans l'espace occupé ;
2. Pour chaque point, calculer les distances entre le point en question et chaque centre de classes et assigner le point au centre le plus près ;
3. Une fois tous les points associés, changer la position du centre de chaque classe sur le centroïde formé à partir de tous les points de la classe ;
4. Répéter les deux dernières étapes jusqu'à ce que toutes les associations ne changent plus pour deux itérations consécutives [32].

La mise en place d'un objectif de minimisation a permis d'obtenir des solutions de meilleure qualité, mais au prix d'un temps de calcul beaucoup grand. Par ailleurs, la taille du problème, considéré dans son ensemble, est énorme ce qui peut affecter le processus d'optimisation négativement. Cette situation a été considérée par le système de résolution par blocs individuels expliqué plus en détails dans la section 4.4.5 de l'article.

Suite à une analyse des résultats obtenus selon différents seuils de distance entre positions sur un ensemble de séquences à la fois, il a été possible de réaliser qu'un seuil unique ne pouvait pas être la solution idéale. En effet, certaines séquences réagissaient mieux à une valeur plus élevée de seuil contrairement à d'autres séquences qui étaient mieux traitées avec un faible

seuil. Pour tenir compte de cette réalité, un seuil variable a été introduit en se basant sur les distances maximales entre les associations minimales de boîtes englobantes. La stratégie a été d'effectuer les associations minimales pour deux trames et se baser sur la valeur maximale de distance p_{max} obtenue. Il faut toutefois faire face à deux problèmes lors du calcul de cette valeur :

1. La valeur d'association maximale peut se trouver non seulement pour chaque paire consécutive de trames, mais également pour chaque paire non consécutive dûe à la possibilité d'occlusion ou de détection manquante. Il faut faire l'ajustement du seuil (facteur d'expansion) le cas échéant. Par exemple, pour associer deux détections des trames 0 et 2, et laisser un trou à la trame 1, la valeur du seuil sera doublée.
2. Les associations minimales ne sont pas nécessairement les bonnes à faire, même si elles ont plus de chances d'être valides.

Ainsi, en échantillonnant le nombre de paires de trames, l'équation pour obtenir le seuil τ propre à la séquence vidéo à l'aide des paramètres k_1 (seuil minimal) et k_2 (facteur d'expansion) déterminés expérimentalement est la suivante :

$$\tau = k_1 + k_2 p_{max} \tag{3.1}$$

Même si certaines de ces modifications ont permis l'amélioration du processus d'association, le modèle développé rend difficile la réduction du nombre d'erreurs de types faux-positif et faux-négatif résultant de l'utilisation de mauvaises détections fournies par le détecteur, c'est-à-dire supprimer les détections en trop et ajouter les détections manquantes puisqu'il est exigé que chaque détection présentée au modèle CP soit utilisée dans une trajectoire.

3.3 Augmenter la qualité des détections utilisées et des trajectoires produites

Afin d'être en mesure de supprimer ces détections en trop et d'ajouter celles manquantes, il a été nécessaire d'utiliser certaines techniques pour encadrer la partie CP de part et d'autre.

Une première technique de filtrage se basant sur l'indice de confiance directement produit par le détecteur fut de retirer les détections de chaque trame dont la probabilité de ne pas représenter un objet d'intérêt est trop élevée. Avec ce filtre, comme la CP n'a pas à considérer ces détections en trop, un effet incident devrait être la diminution du nombre de mauvaises associations, en plus du nombre de faux positifs.

Malgré l'efficacité observée pour la mesure précédente, plusieurs fausses détections subsistaient. Une deuxième étape qui a été considérée se base sur l'intuition que certaines régions

de la scène ont une plus grande probabilité de contenir des détections que d'autres selon l'angle de caméra utilisé. Avec l'action de la gravité, la disposition des obstacles dans le paysage, et la forme des routes et passages piétonniers, il est facile de concevoir que certains espaces sont presque impossibles d'accès tel qu'illustré sur la figure 3.2. Trouver ces régions d'images pourrait donc permettre de non seulement éliminer les détections de faible confiance, mais également d'éliminer celles trop improbables de par leur position dans l'image. Pour ce faire, l'hypothèse formulée fut de considérer que les régions où se déroule le plus de mouvements sont celles où les usagers de la route peuvent circuler. Calculer le flot optique de chaque trame permet justement de connaître l'amplitude du mouvement de chaque pixel. Or, malgré plusieurs tentatives non fructueuses pour tenter de seuiller les mouvements superflus, la présence de trop de bruits provenant de sources naturelles (l'action du vent dans les feuilles des arbres, changements d'illuminations, etc) a rendu impossible de séparer les régions avec fortes probabilités de retrouver un objet d'intérêt des autres régions de l'image.

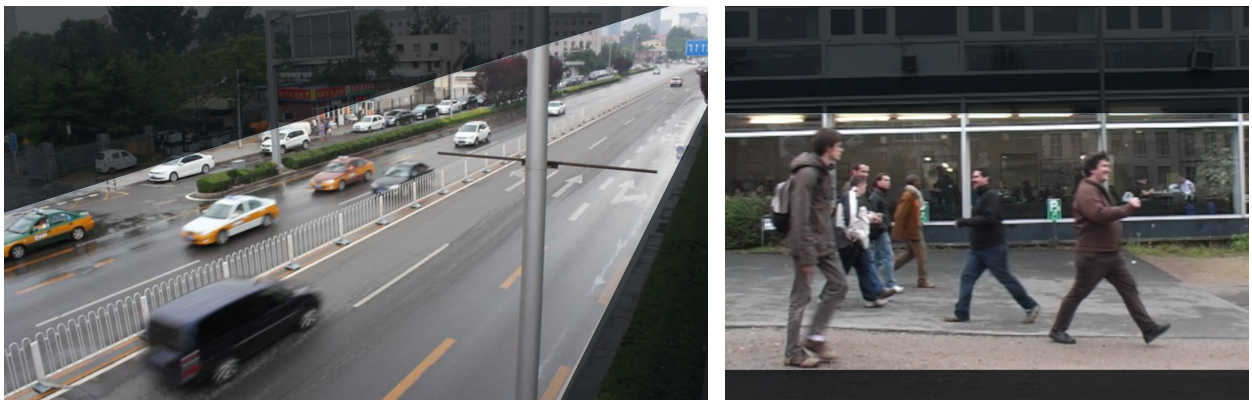


Figure 3.2 Zones mortes (régions obscurcies) de deux scènes où la présence d'usagers de la route est fortement improbable

Les deux dernières techniques permettent d'améliorer la qualité des détections en retirant les moins pertinentes. Il peut néanmoins arriver que de véritables détections soient touchées par ces filtrages. Afin de contrer cet effet, une méthode d'ajout de détections utilisant le suivi d'un seul objet a été développée et appliquée sur les détections. Il s'agit de la dernière étape avant l'utilisation de la CP. L'explication de cette méthode se trouve à la section 4.4.3 de l'article.

Bien que l'amélioration des détections fournies à la CP ait permis une amélioration significative du processus d'association, plusieurs erreurs peuvent être plus difficiles à gérer en amont. Les trajectoires produites par la CP sont souvent incomplètes par la faute des occlusions ou de détections manquantes. Une approche testée a donc été de remplir ces trous à l'aide de l'interpolation linéaire. Ici aussi, la méthode résultante est décrite dans l'article, à

la section 4.4.4.

Toutes ces techniques de filtrage et de prédictions, combinées aux contraintes d'optimisation CP utilisées ont permis une nette amélioration des résultats en réduisant le nombre d'erreurs de suivi. Cependant, le temps nécessaire pour obtenir des solutions était trop grand pour le nombre de trames que la présente méthode pouvait gérer.

3.4 Optimisation du temps de résolution

Toutes les bases de données de suivi multiobjet contiennent des séquences dont le nombre de trames se situent généralement entre 500 et 3000. Bien que ce nombre soit petit comparé aux dizaines d'heures de vidéo en environnement urbain qu'on peut veut pouvoir traiter, la combinatoire produite est très grande pour une approche par CP qui considère les possibilités exhaustivement. Même si la résolution par sous-séquence permet de réduire le nombre de trames à traiter parallèlement par le solveur, l'utilisation de sous-séquences trop courtes sera néfaste pour les résultats. En effet, la CP offre des mécanismes comme le *backtracking* qui permettent de revenir sur les décisions prises afin d'appliquer des corrections. Ces outils perdent de leur portée s'ils sont utilisés sur une sous-instance trop petite du problème.

Le premier réflexe lorsque la résolution prend trop de temps est d'aller examiner la stratégie de branchements définie. Celle établie par défaut a été remplacé par un ordre lexicographique de variables à examiner (voir section 4.4.2 de l'article pour plus de détails) :

1. Les variables associant trajectoires et détections, selon l'identifiant de trames, puis de trajectoires (ordre croissant).
2. Les variables de l'automate de couleurs, dans le même ordre que précédemment.

La stratégie de branchements par heuristique de densité maximale pondérée de solutions a été testée en utilisant les coûts de la contrainte *costRegular* pour effectuer la pondération.

Une des sources de complexité affectant le temps de résolution en CP est la présence de symétries inhérentes à la structure du problème. Le modèle développé dans ce travail en comporte un grand nombre, principalement dû au fait que la numérotation des trajectoires n'est pas un élément caractéristique d'une solution : toutes permutations de trajectoires produit une seul et même résultat de suivi. Pour éviter au solveur de chercher des solutions équivalentes, une représentation alternative du problème a dûe être utilisée pour réduire la complexité du problème. Cette transformation a été un changement majeur qui a permis de réduire le temps de résolution. Or, comme à ce stade la formulation du modèle CP n'a pas été présentée, j'invite le lecteur à se rendre à la section 4.4.1 pour connaître les détails de cette transformation.

3.5 Ajustements finaux

Une fois la méthode définitivement complétée, voici les différents tests effectués dans le but d'évaluer les performances de notre méthode et de comparer celle-ci à d'autres solutions proposées dans la littérature.

La soumission étant très réglementée pour tous les défis offerts par le MOTChallenge (maximum de quatre soumissions, 72 heures d'attente entre chaque soumission), il était difficile d'évaluer chaque amélioration apportée au processus à l'aide d'un ensemble de tests. Cette difficulté a été contournée à l'aide d'un second ensemble de données : UA-DETRAC [33] portant sur les véhicules routiers et excluant les piétons du suivi. Un ajustement des paramètres a été effectué ; les différents usagers de la route ne se comportent pas de la même façon et ne se déplacent pas à la même vitesse. Le principal avantage d'utiliser cet ensemble de données provient du fait que la vérité terrain est disponible pour les ensembles d'entraînement et de test.

Même si notre méthode n'est pas basée sur l'apprentissage automatique, l'intention derrière la séparation des séquences en deux ensembles distincts a été respectée, c'est-à-dire que seul l'ensemble d'entraînement a été utilisé pour l'ajustement de paramètres.

CHAPITRE 4 ARTICLE 1: TRACKING ROAD USERS USING CONSTRAINT PROGRAMMING [1]

Abstract

In this paper, we aim at improving the tracking of road users in urban scenes. We present a constraint programming (CP) approach for the data association phase found in the tracking-by-detection paradigm of the multiple object tracking (MOT) problem. Such an approach can solve the data association problem more efficiently than graph-based methods and can handle better the combinatorial explosion occurring when multiple frames are analyzed. Because our focus is on the data association problem, our MOT method only uses simple image features, which are the center position and color of detections for each frame. Constraints are defined on these two features and on the general MOT problem. For example, we enforce color appearance preservation over trajectories and constrain the extent of motion between frames. Filtering layers are used in order to eliminate detection candidates before using CP and to remove dummy trajectories produced by the CP solver. Our proposed method was tested on a motorized vehicles tracking dataset and produces results that outperform the top methods of the UA-DETRAC benchmark.

4.1 Introduction

In this paper, we address the multiple object tracking (MOT) problem in the context of traffic scenes. A very common approach to tackle the MOT problem uses the tracking-by-detection paradigm, which divides the task into two smaller ones: the detection of objects in the video frames and the association of these detections between frames to form trajectories for the objects of interest [34–37]. It is illustrated in Figure 4.1. The resulting trajectories can represent data about road users that is useful to solve higher-level problems such as improving security in our streets and improving traffic flows for more eco-friendly mobility. For example, tracking road users can help design more secure roads by analyzing users' behaviour and finding anomalies in their trajectories before incidents actually happen [38].

This work focuses on the data association step of MOT methods to improve their performance for traffic scenes. We assume that our road user detections come from an off-the-shelf object detector. The main challenges of data association are: 1) handling occlusions which occur

1. A. Pineault, G.-A. Bilodeau et G. Pesant, "Tracking Road Users using Constraint Programming", article soumis à la revue IEEE Transactions on Intelligent Transportation Systems, novembre 2019.

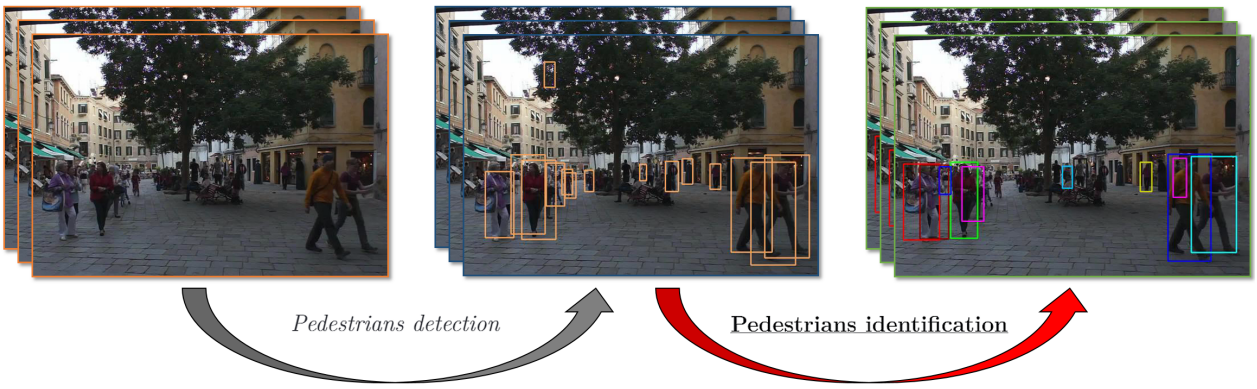


Figure 4.1 Solving a MOT situation involving pedestrians using the Tracking-by-detection paradigm. Pedestrians are first detected and then they are assigned a unique identifier, represented here by the various colors.

when a tracked entity is hidden completely or partially by other objects such as other road users or untracked scene elements (e.g. urban furniture), 2) managing incoming and outgoing objects of interest, which means that a trajectory may not last the whole video sequence, and 3) taking into account the imperfect performance of the detector, such as bounding boxes that do not bound perfectly an object, missing objects, or false detections (see Figure 4.2).

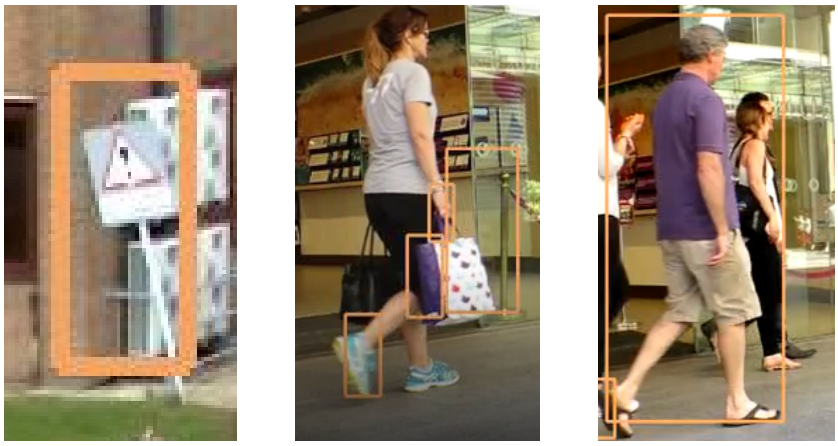


Figure 4.2 Examples of detection errors. a) False detection, b) Object fragmentation, and c) Missed detection (the person behind) caused by an occlusion.

The data association step can be viewed as a combinatorial problem between current tracks (a track is a trajectory being built by the tracker) and new detections. For this reason, the use of constraint programming (CP) is an interesting option to explore since CP has long been used to solve various combinatorial problems [39]. As a first step, CP states a

formal model of the problem to solve using finite-domain variables and constraints on them. It then explores the combinatorial space of solutions using tree search guided by variable- and value-selection heuristics and applies sophisticated inference at each node of the search tree in order to prune infeasible branches. To every variable in the model is associated a finite set called its domain: the variable can take any value in that domain. Constraints on the variables forbid certain combinations of values. Each type of constraint encapsulates its own specialized inference algorithm that filters out inconsistent values from the domains of variables. We refer the interested reader to [26] for a more detailed exposition of CP.

This paper presents a MOT method that produces trajectories by matching recurring objects together using a set of constraints that use the center position and the color of detections. Our main contribution is a road user tracker that uses a CP model that is adapted to the MOT problem. Some filters are defined before and after the CP solving phase. Our method is compared with others based on data association strategies with similar simple features as the ones we are using. We show that in the case of simple features, our CP-based data association method gives better results compared with a state-of-the-art road user tracker on the UA-DETRAC benchmark. Using additional constraints, the proposed CP-based data association can be extended to include other features.

4.2 Background and Related Work

4.2.1 Road User MOT

A simple method to track road users is to use only the proximity between detections in consecutive frames to make decisions. It is possible to do so by computing the intersection over union metric that corresponds to the overlap area proportion between two bounding boxes, then choosing associations based on the highest overlap [6]. This method has proven to be fast and was able to achieve a competitive accuracy on the UA-DETRAC benchmark.

Another approach is to use kernelized correlation filters (KCF) [10] where a filter is estimated in such a way that when applied to the region of interest to track, a strong response is produced. The goal is to find the image region that produces the most similar response in the subsequent frames. This technique was combined with a background subtraction process to track road users [40]. Although successful, this method lacks a proper data association process to better handle occlusions. A Kalman filter is also often used to help association tof road users by predicting missing detections [41]. Another work [42] uses background subtraction, and manages occlusion situations by considering vehicles travelling together in only one track. A state machine manages the combination of tracklets when road users

merge together or choose different paths. It uses keypoints as an appearance model and does data association only on two consecutive frames. Finally, another approach is to track keypoints using optical flow, and grouping these using a common motion criteria [43]. The shortcoming of this method is that objects are not localised very well because it is based on sparse keypoints. Furthermore, keypoint detection and tracking rely on motion. Stopped road users are not tracked.

4.2.2 General MOT

State-of-the-art performance in multiple object tracking is currently obtained with methods using deep neural networks to extract many automatically-learned features [35, 44]. One of these methods [44] is using a convolutional neural network (CNN) to build two classifiers: one for separating object categories and another for instance classification which will differentiate objects of a same category. In this approach, the previous classifiers are connected to a particle filter which, combined with the appearance model, makes a prediction of where the object will appear next. The idea of using object categories in tracking was also exploited in the work of Ooi et al. [41] in road user tracking. More recently, re-identification features are gaining popularity to describe an object of interest in multiple object tracking situations [13, 19].

Another paper using recurrent neural networks (RNNs) has focused more on interactions between objects in order to disambiguate matches [35]. A spatio-temporal appearance model, where a CNN takes bounding boxes as input and outputs features to a LSTM capable of memorizing long-term feature dependencies, is added to a motion feature extractor and an interaction feature extractor. The last part is made of a grid representing where nearby detected objects are located and a second LSTM module.

There are more and more methods that start using visual object tracking (VOT) techniques in a MOT context. In the VOT problem, there is only one given detection in the first frame, that states which object is to be tracked. Thus, VOT method uses strategies to differentiate the foreground from the background. This approach is problematic in MOT situations since there are many similar foreground objects and it becomes difficult to ensure that all objects are tracked since two trackers initialized on two different objects may end up following the same after a few frames. A possible way to mitigate the risk of these situations is to use a VOT for each object to make predictions about the position of the object in the next frame. Then these predictions can be incorporated in a data association method, where they are considered in a more global context [19, 25].

These state-of-the-art methods focus mainly in correcting missing and spurious detections

using a prediction method and defining robust features for data association. They do not study how to make the data association itself. They show that a strong appearance model and predicting where an object should appear next to filter object detections are two important components of a MOT system. However they use a limited data association strategy, as data association decisions are not taken over several frames by combining several observations of the objects in a video. Occlusions are better resolved over several frames. Therefore, a more robust data association method is always desirable.

Typical data association methods used in tracking are the Hungarian algorithm, the joint probabilistic data-association filter (JPDAF), and the minimum-cost flow algorithm. The Hungarian algorithm finds a maximum cost matching in a bipartite graph where the costs are on edges [45]. As the assignment problem can be formulated as a bipartite graph, this algorithm is one way to get the solution. It was used in several works. For example, the Hungarian algorithm can be combined with tracklets (incomplete track parts) to make the associations [36]. In this work, the tracklets are ranked according to a confidence metric considering occlusions, number of frames covered and how well the detections fit. Online detections are matched to existing tracklets with high confidence using the Hungarian algorithm maximizing the affinity level. The next step is to globally match these new associations with low level detections using once again the Hungarian algorithm.

JPDAF is a statistical method that can track objects based on what is the most likely outcome for each trajectory. It considers any detection available, but also the possibility of a missing object or a false detection. It was used several times for the MOT problem, for example, in the work of [22] and [23].

The min-cost flow algorithm combines a cost function with a greedy algorithm in order to obtain the required tracking associations [37]. The goal here is to formulate the data association as a minimum-cost flow problem, then to compute shortest paths on the flow network with detections at each frame as nodes, from the first appearance of an object to its last appearance in the scene. This process helps ensure that the resulting trajectories are as smooth as possible.

4.3 CP Background

In this section we present some useful background about constraint programming. There are three high level constraints used by our approach:

1. `AllDifferent`: Applied to a set of CP variables $\{x_1, x_2, \ldots, x_n\}$, this constraint states that it is not permitted that two variables x_i and x_j share the same value:

$$x_i \neq x_j \qquad \forall\, 1 \leq i < j \leq n \tag{4.1}$$

2. `Inverse`: Given two arrays of CP variables $\langle x_1, x_2, \ldots, x_m \rangle$ and $\langle y_1, y_2, \ldots, y_n \rangle$, the following equations must hold:

$$x_i \in [1, n] \Rightarrow y_{x_i} = i \qquad \forall\, 1 \leq i \leq m \tag{4.2}$$

$$y_j \in [1, m] \Rightarrow x_{y_j} = j \qquad \forall\, 1 \leq j \leq n \tag{4.3}$$

   This constraint is useful to maintain consistency between dual representations.

3. `Regular` and `CostRegular`: These express a constraint as membership to a regular language. Given a sequence of variables $\langle x_1, x_2, \ldots, x_n \rangle$ and an automaton $\mathcal{A}$, each solution to the constraint corresponds to an assignment of values to these variables that spell out a word recognized by $\mathcal{A}$ [28]. `CostRegular` is the optimization version of the latter constraint: it takes as additional parameters the costs associated with each combination of variable, value, and automaton state, as well as a cost variable equal to the sum of individual costs in a solution [29].

The inference algorithm associated with each of these constraints removes every inconsistent value in the domain of each variable.

4.4 Methodology

Our road users tracking approach can be summarized as follows. First, detections are obtained for every video frame using a road user detector. A VOT is also applied to predict the position of the objects detected in the previous frame, in the new one. These detections are then filtered based on their confidence and redundancy. Detections can be redundant if a predicted box matches a detection in the bouding boxes initial set. The predicted bounding boxes should be ignored in this case. Then, the detections are used to instantiate the CP model and the model is solved by forming tracks with the detections. Finally, unused tracks are removed before outputting the final result (a set of trajectories). Our method is detailed in the following and a high-level view is shown on Figure 4.3.

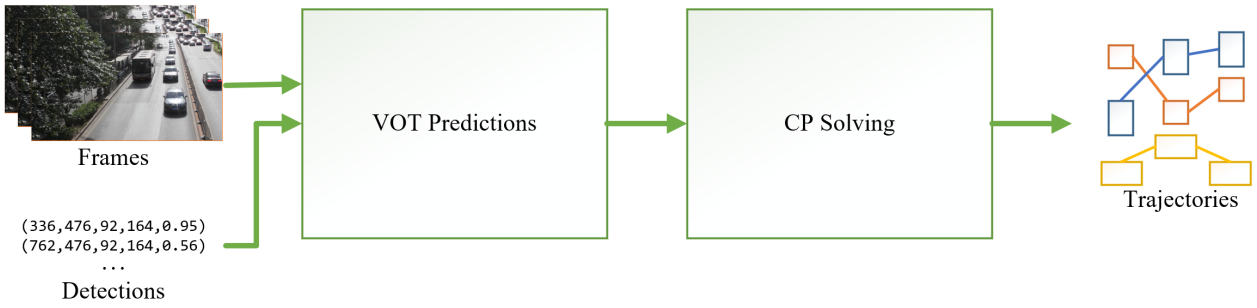


Figure 4.3 Diagram showing the interactions of the high-level modules of our method.

4.4.1 Constraint Programming Model

Main Variables

In order to associate tracks to detections, variables

$$t_{ij} \in \{1, 2, \ldots, \tau\} \qquad \forall\, 1 \leq i \leq m,\ 1 \leq j \leq n_i \tag{4.4}$$

identify to which track (out of τ available ones) detection j at frame i belongs, where m is the number of frames and n_i the number of detections at frame i.

Such a representation is convenient to avoid unnecessary symmetries since it uses the smallest number of possible variables: one per detection. However, referring to consecutive detections of a given track is not possible before we obtain the solution; there is no way to know the indices of the variables that will be part of a same track before the solving step. Since we need this to express some of our constraints, a second array of variables is defined,

$$d_{ik} \in \{1, 2, \ldots, \tau\} \qquad \forall\, 1 \leq i \leq m,\ 1 \leq k \leq \tau \tag{4.5}$$

identifying the detection assigned to track k at frame i. Whenever $\tau > n_i$ for some frame i, values greater than n_i do not represent actual detections — they are however necessary because we will require that detections be uniquely associated to tracks (see Equations 4.7 and 4.8).

To ensure that these two dual representations remain consistent with each other, we link them using an `inverse` constraint

$$d_{i\star} = \texttt{inverse}(t_{i\star}) \qquad \forall\, 1 \leq i \leq m \tag{4.6}$$

relating each variable to its counterpart in the other array. the star symbol ($\star$) is used to indicate the selection of a subarray.

Frame Consistency

The next step is to ensure that it is impossible to find a specific track or detection more than once at a given frame. This is a common requirement for which CP provides an `AllDifferent` constraint. Such constraints have been specified for this purpose using the same variables as the previous equations.

$$\texttt{AllDifferent}(t_{i\star}) \qquad \forall\, 1 \leq i \leq m \tag{4.7}$$

$$\texttt{AllDifferent}(d_{i\star}) \qquad \forall\, 1 \leq i \leq m \tag{4.8}$$

Position Constraints

After the model structure is set, constraints on features such as position can be considered. First, an array of integers is created for each dimension in a frame (x and y coordinates). These arrays, indexed by frame and then by detection ID, contain the center of each detection bounding box. It is then possible to formulate a constraint restricting the distance between two consecutive detections in a track by stating that the one-dimensional distance along the X and the Y axes separating them may not be greater than thresholds λ_x and λ_y :

$$|x_{i,d_{ik}} - x_{i+1,d_{i+1,k}}| \leq \lambda_x \quad \forall\, 1 \leq i < m, 1 \leq k \leq \tau \tag{4.9}$$

$$|y_{i,d_{ik}} - y_{i+1,d_{i+1,k}}| \leq \lambda_y \quad \forall\, 1 \leq i < m, 1 \leq k \leq \tau \tag{4.10}$$

The same process has been tested using object scale indicators (width and height) but this feature did not help improve the solution.

Appearance Model

An appearance model is useful for minimizing the risk of mismatch between two nearby objects. Our proposed appearance model assigns a color class label to each detection available. The color classes are obtained by clustering observed object color histograms in several videos. Clustering is performed with K-Means using the Bhattacharya distance between the two color histograms. The learned classes are then used during tracking. A detection in a

frame is assigned the color class label of the nearest cluster.

A `CostRegular` constraint is applied to each track preventing the association of two objects with contrasting colors. Possible transitions between color classes for a given object at each frame is governed by a state machine. Transitions between similar color classes are allowed but a cost is applied based on the Bhattacharya distance between the class centers. The solver will use the sum of all these costs as a minimization objective.

$$c_{ik} = \texttt{colour}(d_{ik}) \tag{4.11}$$

$$\texttt{CostRegular}(c_{\star k}, \mathcal{A}, C, a_k) \qquad \forall\, 1 \leq k \leq \tau \tag{4.12}$$

$$\texttt{min}\left(\sum_{k=1}^{\tau} a_k\right) \tag{4.13}$$

In these equations, c_{ik} represents the list of color class labels for the track detections acting as transition variables between the automaton's states. $\mathcal{A}$ and C correspond respectively to the automaton and the cost matrix for every possible transition. Variable a_k represents Track k's total cost computed by the regular constraint.

Each color class is represented by two states, depending on whether the object is currently being occluded or not (e.g. red object and red occluded object). Using different states for occlusions allows to set a cost for transitions from a valid detection to an absence of detection while preserving the object appearance even when it is occluded for an extended duration (see Figure 4.4). The cost are based on the Bhattacharya distance between each color class histograms.

4.4.2 Solving Strategy

Variable Selection

As stated in the introduction, CP explores the combinatorial space of solutions by branching in a search tree that enumerates all possible solutions. To solve our model, we first branch on the t_{ij} variables since it is the main array containing the smallest number of variables necessary to specify a solution whereas in the case of the d_{ik} variables, all possible tracks are instantiated in case they are needed. To allow the state machine to handle occlusions or the absence of a detection for a given track, it is necessary to branch on the c_{ij} variables. The lexicographic order specified will make sure that branching will always at first be made on t_{ij} variables, then on c_{ij} variables. Without including c_{ij} variables in the branching order, some

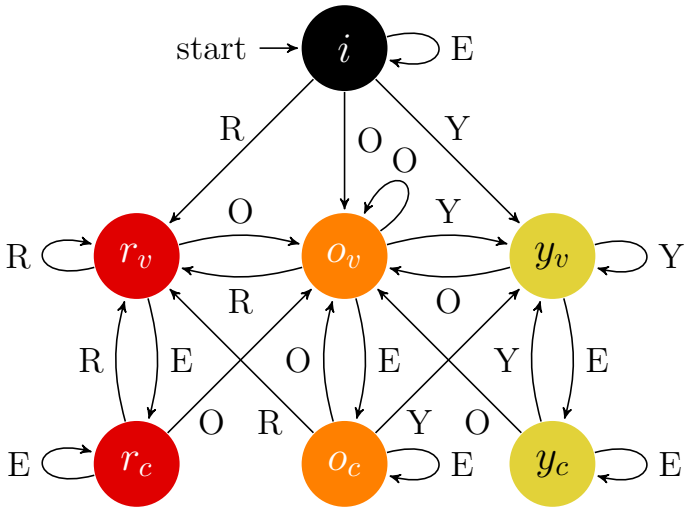


Figure 4.4 Small state machine example illustrating the doubled states: visible (v) and occlusion (c) for three color classes: red (d), orange (o) and yellow (y). Transition values correspond to the detection color (R, O, Y) or empty (E) when there is no detection.

subconstraints of the *CostRegular* might be ignored in the event of occlusions (no t_{ij} linked to the c_{ij}). In both cases, variables are considered frame by frame, then detection identifier by detection identifier in ascending order.

Value Selection

Since the t_{ij} variables are fixed by the solver from the first to the last frame, the use of prediction becomes an interesting option to guide the search. By computing every distance between possible pairs of consecutive detections, it is possible to obtain a ranked list that the solver can use to branch on closest detections first for each t_{ij} except for the last frame. This way, we minimize the distance between each consecutive pair of objects without altering the model objective oriented towards the appearance preservation.

The inherent symmetries are addressed in the branching to control the number of opened tracks available as candidate for t_{ij} variables. The domain of each of these variables is reduced to all previously used tracked index plus one other that may be opened if necessary. This strategy was applied before to the Steel Mill Slab Design Problem [31].

4.4.3 Pre Solving Computations

As mentioned in the introduction, detectors are prone to miss objects of interest. To compensate this weakness, we developed a pretracking method using an VOT to add new detections,

using this process:

1. We initialize a KCF tracker [10] for each given detection of the first frame. With KCF, the image is correlated with a filter that is learned for each road user. The object is localized based on the strength of the filter response. It is a very fast tracker.
2. For the next frames, one by one, all KCF trackers make a prediction about the next bounding box of the object. For each prediction that is not redundant with a detection of the current frame, the bounding box is kept. For all detections in the current frame that do not match a prediction, a new tracker is initialized.
3. Each created tracker has a lifespan that allows it to make predictions for a fixed small number of frames. It is assumed that the tracker is reliable only on a small time window. To avoid adding detections based on a false detection, predicted boxes are only added to the existing detections if there is a match between an existing detection and the tracker prediction in a subsequent frame.

Figure 4.5 illustrate many possible situations. For trajectory 1, a tracker initialized in the the first frame add two detections (frames 2 and 3). This is the ideal way to manage a two frames occlusion; two boxes are added when the object is hidden, then the tracker response match again the object and reinitialized itself since is prediction life is over. Trajectories 2 and 3 illustrate cases where we assume that the tracker is mistaken. If a VOT tracker is initialized on a false detection, no predicted boxes should be kept (trajectory 2). For the third trajectory, the tracker is able to match a real detection, but produces afterward another detection that remains unmatched. To avoid the addition of too many detections that would fall into the false positives category, the last detection is simply discarded.

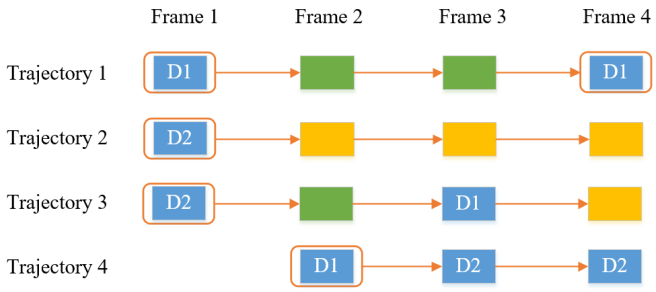


Figure 4.5 Illustration of the detection prediction process where blue boxes are detections from an object detector, yellow and green boxes are respectively discarded and added detections using KCF. The orange frames indicate the initialization of a tracker.

4.4.4 Post Solving Computations

Once the solver has completed the associations, another type of filtering is required due to how the model is built. The model is unable to eliminate detections entirely — the only choice that it has is to put each of them in dummy tracks, which will have to be deleted afterward. To delete them, a filter removes every track that contains fewer than β_D detections from the object detector, a parameter set to a fixed value for all sequences.

After the suppression of the unwanted tracks, it is possible to fill gaps in the remaining tracks. A small state machine searches for places where there are γ_D or less consecutive missing detections and fills these empty spots by adding new detections using linear interpolation to find the correct center position and size of the bounding box for each one. This has to be done carefully because a too large gap probably means that an identity change has happened. Therefore, filling missing detections in these situations would probably mean that false detections are added.

4.4.5 Batch Resolution Process

Since this model uses optimization to find the best solution to the MOT problem applied to large instances, the time required to find a solution is much longer than the one obtained by simply solving the satisfaction problem. Moreover, the number of computations required to find the optimal solution grows exponentially in the size of the problem instance which makes the minimization process intractable to complete in practice.

This situation was addressed by dividing a video sequence into multiple batches of κ frames. Except for the first batch, to insure that the transition between batches stay as smooth as possible, the first β frames of the trajectories were taken from the end of the previous batch. Frames containing no detection were ignored by this batch system, which means that a batch is not always constituted of consecutive frames. This is also valid for the overlapping section of the batch. If a sequence would contain too many empty consecutive frames, the next batch will start at the first frame containing at least a detection and will be be completely independent of the previous one. The values that gave the best trade-off between accuracy of the global solution and computation time was 30 for κ and 5 for β.

4.5 Experimental Setup

4.5.1 Evaluation Metrics

To evaluate our method, we use the standard MOT metrics that account for three types of errors at each frame i: 1) Identity switches (IDS_i) or the number of mismatches, 2) the missing detections or false negatives (FN_i) which are the number of objects that are not tracked, and 3) the false positives (FP_i) which stands for the number of detections representing no object of interest. With these values, it is possible to compute the *MOTA* score [46]

$$MOTA = 1 - \frac{\sum_i (FP_i + FN_i + IDS_i)}{\sum_i N_i} \tag{4.14}$$

where N_i is the total number of ground-truth objects per frame and it is summed throughout the entire video. The matching between tracking results and ground truth is done by measuring the intersection over union (IoU) of the two boudning boxes. If the overlapped area is higher than a certain percentage, the match is considered valid.

A second metric known as *IDF1* balances the recall and precision for each trajectory into a single value. This is done by dividing the total number of accurately matched detections to the average number of ground truth and given detections [47]. Finally, mostly tracked trajectories (MT) is the ratio of ground truth that possess a matching hypothesis for at least 80% of their life span and mostly lost trajectories (ML) is the number of times this ratio is under 20%. Fragmentation ($Frag$) indicates how many times any trajectory got interrupted over its length.

4.5.2 Dataset

We tested our method on the UA-DETRAC tracking challenge [33]. This dataset contains 60 training sequences and 40 for the testing set. Only motorized vehicles such as cars, buses and trucks are tracked. We did not submit our results on the benchmark due to technical reasons (challenge website down, evaluation toolkit not cross-platform) to obtain the PR metrics. Instead, the code of the second best methods (IOU tracker) [6] was downloaded, executed and tested using the usual MOTA metric (the code of the best method was not functionnal. Besides, it was only marginally better than the second one). Detections were filtered with the same parameters for both our method and the competing one. We used the same source detection from R-CNN.

We used a second dataset: the 2D MOT 2015 challenge [48] to discover if our method could be extended from vehicules to pedestrian tracking. This dataset provides twenty-two video sequences divided equally in a training set and a testing set. There are many additional difficulties compared to UA-DETRAC such as moving cameras, variable frame rates (sometimes 7-10 FPS) and less elevated camera angles that are more prone to occlusions.

4.5.3 Constraint Programming Solver

The constraint programming solver used in this paper is IBM ILOG CP (version 1.6) augmented with our own implementation of the `CostRegular` constraint. All developed code is available on GitHub [2].

4.6 Results

Three series of experiments were conducted to evaluate how well our proposed method performs compared to others. First, the quality of our tracking approach is compared to a state-of-the-art method on the UA-DETRAC dataset. Second, we extend our method to the 2D MOT 2015 pedestrian dataset. Third, an ablation study evaluates the isolated performance of the components of our method.

For all experiments, the minimal overlap value between the detection and the object areas for true positive was fixed to 50% in the MOTA computation. No parameter optimization was done using sequences from any testing sets.

4.6.1 UA-DETRAC

To make sure that the way to compare the methods is fair, the same detection confidence parameter was used every time. Since the post-solving computations were not improving results for UA-DETRAC sequences, it was turned off for the results of this section.

Table 4.1 present the tracking results obtained by our method (CP) for all sequences of the test set. The MOTA was computed globally by considering the total number of errors and objects. The R-CNN public detections were used for both the IOU tracking method and ours (CP). These results show that our method is able to outperform IOU in every situations and globally. The MOTA is affected by the difficulty level, especially by the hard sequences where the false negative errors are higher than in any other sequence groups. Weather conditions and exterior luminosity also impact tracking performance. While tracking tends to be easier

2. Link will be added upon acceptance

Table 4.1 Results on the UA-DETRAC test dataset. Bold indicates best MOTA result. The first category group (Easy, Medium and Hard) differentiates sequences according to the difficulty level while the second category group (Sunny, Cloudy, Rainy and Night) is about illumination and weather conditions. All groups of sequences are mutually exclusive in their category, but a single sequence can be found in multiple categories. Higher MOTA is better as are lower IDS, FN and FP

Sequences	CP + R-CNN				IOU + R-CNN			
(per category)	*MOTA*	*IDS*	*FN*	*FP*	*MOTA*	*IDS*	*FN*	*FP*
Easy	**64.39**	3840	31675	6833	60.66	103	40824	5861
Medium	**63.70**	11533	74538	15159	58.36	310	98117	17683
Hard	**48.38**	10316	126451	6704	45.58	215	142372	8686
Sunny	**90.65**	1748	21118	3131	64.09	35	24348	2980
Cloudy	**76.42**	6392	51383	7766	69.04	221	61163	7784
Rainy	**58.42**	9961	94828	10793	35.63	126	112843	13254
Night	**71.24**	7588	65335	7006	41.98	246	82959	8212
Global	**57.52**	25689	232664	28696	53.51	628	281313	32230

in sunny weather (highest results), rain looks like the most difficult environmental factor to deal with.

Our strategy that helps to improve the quality of detections seem to have a high impact on the global results since the number of false positive and false negative errors are lower than those obtained by the IOU method for the same detections. However the number of ID switches is significantly higher. The main reason explaining this situation comes from the CP model itself. While the objective of the `CostRegular` constraint includes the minimization of the number of occlusions found in open trajectories, the branching strategy always tries to use already open trajectories first. Thus, the solver will implicitly prioritize compact solutions. Therefore, while our solution reduces significantly false positives and false negatives by handling better occlusion situations, it may result in a higher number of ID switches. Note that even if the number of available trajectories is determined before the resolution process, we made sure that there was always at least one unused trajectory.

In order to give more details about the performance of our method, the results obtained on a subset of the testing set are presented in Table 4.2. The five best and worse sequences in term of MOTA are presented. The distribution of errors made by the two compared methods is still similar to what was observed in the global results, especially regarding the number of IDS and FN mistakes.

To understand why there is an important variability between sequences in this dataset, Figure 4.6 presents extracted frames of videos from Table 4.2. Our method performed best

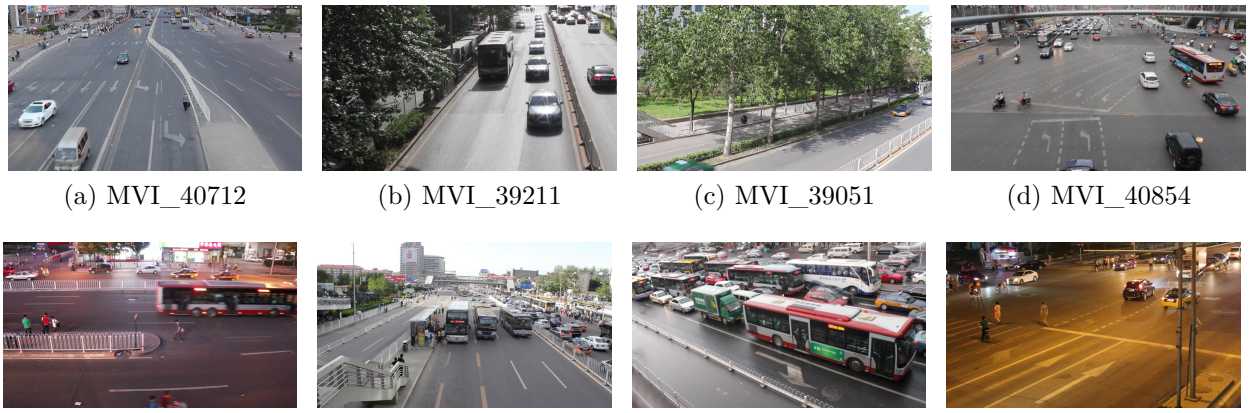
Table 4.2 Detailed results on the sequences in which we obtained the five highest and lowest MOTA values (UA-DETRAC, testing set). Bolface means best MOTA result. Higher MOTA is better as are lower IDS, FN and FP

Sequences	CP + R-CNN				IOU + R-CNN			
(Name)	*MOTA*	*IDS*	*FN*	*FP*	*MOTA*	*IDS*	*FN*	*FP*
MVI_40712	**83.97**	449	2726	613	82.49	30	3763	336
MVI_39211	**78.96**	28	846	37	78.00	0	866	51
MVI_39051	**77.98**	71	306	161	75.62	2	471	119
MVI_40854	77.24	485	3222	867	**78.57**	10	3835	460
MVI_39271	**76.50**	189	1350	459	74.17	1	1710	477
MVI_40763	**36.50**	309	5971	33	30.62	3	6741	115
MVI_40792	**33.40**	537	7044	470	29.60	3	8161	332
MVI_40863	**31.83**	937	20966	392	29.43	6	21370	525
MVI_39501	30.51	140	2973	902	**33.62**	4	3118	687
MVI_40761	**22.22**	629	15655	36	18.93	4	16839	117

on the images from the first row while the worst results were obtained on the second row. The sunny weather condition seems to be a critical factor to improve the results, but we think the camera angle is playing an important role. Tracking while facing the traffic with a good elevation angle looks like the optimal setup to achieve best performance with our method. The same applies to IOU tracker indicating that detections could be of lesser quality in those setups. No sequence in a rainy environment are among the least successful sequences even if it was the category with the worst results globally, which is counterintuitive. We assume the explanation comes from the camera angles used; side perspectives and low elevation angles cause difficult tracking situations. This is logical since both factors increase the probability of occlusion. Examples supporting this hypothesis are in the MVI_40761 sequence (Figure 4.6e), where the bus is hiding the two most distant lanes for over half the frame and in the MVI_39501 sequence (Figure 4.6f) where even with a front view, tall vehicles are able to hide many others behind them.

4.6.2 2D MOT 2015

Table 4.3 compares our method with other approaches. The first two rows, *HWDPL* [44] and *AMIR15* [35], are to show the current performance of state-of-the-art methods that use multiple learned features, learned cost function and prediction in future frames. The last four entries include our method and three classic approaches using similar simple features as ours: *DP NMS* (Min-cost flow) [37], *TC ODAL* (Hungarian algorithm) [36] and *JPDA OP*, a JPDAF implementation from the MOT challenge website [48].



(a) MVI_40712 (b) MVI_39211 (c) MVI_39051 (d) MVI_40854

(e) MVI_40761 (f) MVI_39501 (g) MVI_40863 (h) MVI_40792

Figure 4.6 Filming perspective for different video sequences on UA-DETRAC sequences. The first row contains sequences in which our method performed the best while the second row contains the worst ones.

Table 4.3 Tracking results on the 2D MOT 2015 benchmark.

Method	*MOTA*	*IDF*1	*MT*	*ML*	*FP*	*FN*	*IDS*	*Frag*
HWDPL [44]	38.5	47.1	8.7%	37.4%	4005	33203	586	1263
AMIR15 [35]	37.6	46.0	15.8%	26.8%	7933	29397	1026	2024
CP (ours)	17.0	13.3	2.5%	58.4%	4872	43170	2973	3077
TC ODAL [36]	15.1	0.0	3.2%	55.8%	12970	38538	637	1716
DP NMS [37]	14.5	19.7	6.0%	40.8%	13171	34814	4537	3090
JPDA OP	3.6	7.5	0.4%	96.1%	1024	58189	29	119

These results show that there are still many improvements to be made before we can match the performance of the state-of-the-art approaches, but it also shows that our method surpasses many classical approaches using similar features, even with our method using only a simple appearance model considering only ten color classes and the tendency to make many ID switch mistakes due to the CP model.

The results are also significantly lower than those obtained on the UA-DETRAC benchmark. Tracking pedestrians instead of road vehicles comes with its own challenges. People move more slowly than motorized vehicles, but their general appearance is more variable because of the leg-and-arm complex movements required to walk. The 2D MOT 2015 dataset was created before the rise of deep learning, therefore the noise level contained in the set of public detections is higher. The frame rate is not the same for all sequences (from 10 to 30).

4.6.3 Ablation study

After comparing the method to others available, the next step was to check the performance of individual modules. We made this evaluation on a subset of the training set of UA-DETRAC, that includes 22 videos. Table 4.4 reports the MOTA values obtained after three removals:

1. At first, the problem was converted to a strict constraint satisfaction problem (CSP). The `CostRegular` constraint is kept, but the trajectory costs are no longer optimized.
2. Then, the `CostRegular` is completely removed. No appearance model is used at this point.
3. Finally, the pre solving computations based on an VOT technique are deactivated.

Table 4.4 Results on a subset of sequences from UA-DETRAC with different configurations of our method. Each removal is valid for the current row and all the row below.

Setup	*MOTA*	*IDS*	*FN*	*FP*	*Exec (s)*
All	80.23	11048	39831	8722	53747.70
CSP	80.01	11699	39835	8726	1180.25
No Regular	80.01	11710	39835	8726	1100.41
No Presolve	78.88	12909	43626	7126	246.25

Table 4.4 confirms that each module helps improve the quality of the tracking, even if the increase is sometimes only small. Without the optimization process, the `CostRegular` is not removing many ID switch errors. The addition of an optimization objective allows a significant drop in the number of ID switch mistakes while maintaining approximately the same number of false positives and negatives. This indicates that a simple motion model

is able to make mostly correct associations since forbidding associations based on vehicle appearances produces almost identical results.

The pre solve process allows to eliminate a greater number of false negatives while not increasing too much the number of false positives, thus increasing the global MOTA value. It makes sense since this module is mainly creating new detections.

The resolution time values were provided (in seconds) for each execution since it is an important factor to assess the quality of a CP model. All computations were done in a virtual machine running Ubuntu 19.10, either on a Intel i7-2600 processor from 2011, or a 2018 i9-9900k processor for the ablation study.

Heavy computations are required to obtain the best results with this method. As MOT is a problem that often requires to be solved in real-time; video surveillance cameras are filming constantly, taking hours to solve a five-minute video sequence is not always a possibility. The ablation study presented in Table 4.4 confirms that even without considering the appearance model and using single object tracking predictions to add detections, the tracking performance is not dropping that much. The number of frames in the sequences equal 35495, which means that real-time performance is achieved since all UA-DETRAC sequences are at 30 frames per second, even with the presolve process and the automaton activated. This is an advantage provided by our CP method; the problem is solved in a larger context with real time performance compared to the IOU tracker which is fast, but considers the association problem only two frames at a time.

4.7 Remaining Challenges

4.7.1 Computation times

MOT is a problem where the computation time is limited since completing the tracking in real-time is desired in many situations. Our approach using CP and all its components is not close to this level of efficiency. However, as we have shown, removing some component can allow achieving real-time with a small performance hit. Processing video sequences that contain often more than a thousand frames increases the number of possible combinations before obtaining a solution. The total number of objects throughout the complete sequence is also impacting the computation since it directly affects the domain of the main branching variables. As a solution to this, we applied the data association to blocks of consecutive frames. Recall that usually data association is performed only on two consecutive frames. Considering more frames allows taking better long-term decisions. It is therefore not mandatory to perform the data association with all the frames, if at least, the associations are solved with

bigger batches. Finally, the complexity of the appearance model (i.e. the number of possible appearance values) affects directly the time required to test each variable-value combination.

4.7.2 Appearance Model

The appearance model is important to indicate the presence of occlusions. The precision with which we describe detections will impact the tracking accuracy. One approach may be to use vectors to describe the object (histogram of oriented gradients, deep features). However, they are difficult to include in a CP model without doing clustering first which may reduce the precision of object descriptions. As said before, a complex model will also slow down the resolution process.

4.8 Conclusion

This paper introduced a novel data association method using constraint programming. More precisely the center position of objects and their colors are used as main features to investigate if CP can be a valid approach to solve this problem in the context of road user tracking. Our result show that CP can help improve tracking as we outperform the IOU tracker on UA-DETRAC, which is one of the best tracker on this road user MOT dataset. Our strategy of using object position and a simplified color model proves to be well suited for vehicle tracking. There are still improvements required to be able to solve large instances and to capitalize better on road user appearance. Future work will include developing a more complex model that considers a larger number of features and optimizing the search strategy in order to improve run-time and performance on pedestrian tracking.

Acknowledgements

This research was supported by an IVADO fundamental research grant.

CHAPITRE 5 DISCUSSION GÉNÉRALE

La principale discussion de la méthode finale se trouve dans la section 4 de l'article présenté dans ce mémoire. Cette discussion sera donc orientée davantage vers les différentes étapes intermédiaires ainsi que les raisons expliquant les décisions prises.

Avant d'obtenir les résultats présentés dans l'article, plusieurs évaluations intermédiaires ont été nécessaires pour déterminer quelles étaient les forces et faiblesses de cette méthode de suivi de manière à remplacer les éléments problématiques. Cette section présente les différents résultats obtenus au cours de ces expériences.

Tous les tests ont été effectués sur les ensembles d'entraînement que ce soit de l'ensemble de données 2D MOT 2015 ou UA-DETRAC. Dans plusieurs situations, la vérité terrain a été utilisée. Les détections fournies, contrairement à la vérité terrain, contiennent beaucoup d'erreurs ce qui rend l'isolation des variables à étudier plus difficile. Un problème également intéressant à étudier pour évaluer la robustesse d'une méthode de suivi est d'utiliser une version de la vérité terrain réduite, c'est-à-dire où des boîtes englobantes ont été retirées aléatoirement à intervalles réguliers. Il est ainsi possible de voir si la CP est en mesure de placer les occlusions aux bons endroits et si les méthodes d'ajouts de détections fonctionnent bel et bien.

Avant de poursuivre, les résultats sont présentés à titre de comparaison entre plusieurs situations, en contexte fixe et ne sont pas nécessairement ordonnés chronologiquement. Il est donc normal que les résultats témoins entre plusieurs expériences ne correspondent pas aux mêmes valeurs. Pour tous les tableaux de cette section, les valeurs en caractères gras correspondent aux meilleurs résultats.

5.1 Seuil de position

Dans les contraintes de position de notre modèle, faire varier le seuil de distance de positions a pour effet de changer le nombre d'associations possibles dans chaque trajectoire. Avec un seuil plus faible, la probabilité de faire la bonne association est plus grande dans la majorité des situations. D'une part, un même objet bouge généralement très peu d'une trame et d'autre part, moins de détections environnantes sont considérées. Par contre, en cas de seuils trop faibles, l'association correcte à effectuer pourrait être éliminée. À l'inverse un seuil plus élevé réduit le risque que la bonne association à faire soit rejetée, mais augmente les possibilités de faire une mauvaise association.

Les premiers tests effectués pour tenter de trouver une valeur de seuil optimale ont été d'exécuter la méthode complète sur l'ensemble des séquences d'entraînement du 2D MOT 2015, puis de noter les performances obtenues. Pour l'ensemble de ces tests utilisant les détections de la vérité terrain, l'automate de couleur a été activé et la limite de temps accordé pour trouver une solution maximale a été fixée à 20 minutes pour chaque séquence, considérée en un seul bloc. Le tableau 5.1 permet de voir une récapitulation de ces résultats, pour des valeurs de seuils de 30, 50 et calculées dynamiquement tel que décrit à la section 3.2.

Tableau 5.1 Résultats de la méthode selon différents seuils de position sur les séquences d'entraînement du 2D MOT 2015 (résolution en 20 minutes max).

Séquences	SEUIL FIXE = 30					SEUIL FIXE = 50					SEUIL DYNAMIQUE					
	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)	τ
Venice-2	**99.63**	2	0	0	1206.84	99.26	4	0	0	1214.17	98.52	6	0	2	1247.31	34
KITTI-17	91.90	19	1	12	1206.30	**95.95**	14	0	2	1208.03	**95.95**	14	0	2	1208.56	51
KITTI-13	85.71	9	1	6	1205.71	**92.86**	3	0	5	1208.66	**92.86**	4	0	4	1209.01	51
ADL-Rundle-8	95.87	14	0	7	1205.42	**97.05**	13	0	2	1212.58	95.87	16	0	5	1209.49	35
ADL-Rundle-6	**100.00**	0	0	0	1205.42	99.26	2	0	0	1208.57	**100.00**	0	0	0	1209.22	38
ETH-Pedcross2	**96.25**	9	0	6	1205.59	94.75	12	0	9	1207.92	**96.25**	9	0	6	1208.08	24
ETH-Sunnyday	**98.79**	1	0	2	1206.54	93.95	11	0	4	1208.39	**98.79**	1	0	2	1209.12	32
ETH-Banhof	88.58	19	0	14	1204.73	**88.93**	19	0	13	1215.50	88.58	19	0	14	1208.58	34
PETS09-S2L1	**100.00**	0	0	0	1205.44	99.10	2	0	0	1208.62	**100.00**	0	0	0	1212.32	32
TUD-Campus	91.60	15	0	7	1205.29	**92.37**	15	0	5	1208.46	91.60	17	0	5	1209.38	35
TUD-Stadmitte	97.00	8	0	3	1206.26	95.64	13	0	3	1208.14	**98.09**	5	0	2	1208.68	26

Les résultats obtenus permettent de constater que l'utilisation d'un seuil fixe ne permet pas de maximiser le résultat sur l'ensemble des séquences. Le plus grand écart de résultats se situe au niveau des séquences KITTI-13 et KITTI-17 où la différence de MOTA est la plus marquée. Ces deux séquences sont parmi celles ayant le plus bas taux de trames par seconde (10 tps). De plus, pour les vidéos KITTI, la caméra est installée sur une voiture en mouvement, ce qui implique que l'association doit considérer le mouvement des objets d'intérêt, les piétons, en plus du mouvement de la voiture, ce qui augmente les distances entre associations valides et ainsi la hausse des performances pour un seuil plus élevé. Le seuil dynamique permet de prendre en compte les caractéristiques comme le taux de trames et la vitesse de la caméra puisque ceux-ci affectent directement la position détectée de chaque piéton.

Ensuite, afin d'optimiser notre méthode pour l'ensemble de données UA-DETRAC, une nouvelle batterie de tests a été effectuée sur un échantillon de cinq séquences pour cinq différentes valeurs de seuils fixes et une dynamique (voir le tableau 5.2).

Contrairement aux résultats du 2D MOT 2015 où le seuil dynamique offrait les meilleures performances en permettant de s'ajuster aux mouvements de caméras et aux différents taux de trames, les résultats sur DETRAC sont les plus élevés avec un choix de seuil fixé à 35. Pour ce jeu de données, les principales raisons qui expliquent les variations entre les séquences,

Tableau 5.2 Résultats de la méthode selon différents seuils de position sur un sous-échantillon des séquences d'entraînement d'UA-DETRAC

Séquences	τ					
	25	30	35	40	50	Dyn
MVI_39801	60.54	60.93	60.99	**61.08**	60.99	61.01
MVI_39861	39.31	39.52	39.81	40.14	**40.18**	38.68
MVI_40191	**90.60**	90.56	90.53	90.46	90.31	90.53
MVI_40192	**90.00**	89.99	89.96	89.94	89.88	89.90
MVI_40201	87.61	88.17	**88.38**	88.32	88.27	88.21
Global	86.84	86.93	**86.94**	86.91	86.91	86.87

particulièrement entre MVI_39861 et MVI_40191 où la valeur de seuil optimale diffère d'un facteur deux, proviennent du positionnement de la caméra. Même si celle-ci est toujours fixe et avec un angle plus grand par rapport au sol, différentes configurations sont possibles. Placer la caméra au-dessus de la route et la diriger vers les véhicules approchant en est une. Placer la caméra en arrière d'un trottoir piétonnier et capter les véhicules par le flanc en est une autre. Tout dépendant du positionnement et de l'orientation de la caméra, l'effet de perspective agira de telle sorte que le déplacement de deux véhicules se déplaçant à la même vitesse peut avoir une progression marquée par un nombre différent de pixels entre deux trames consécutives. Quoiqu'il en soit, dans ce test, on constate que l'utilisation d'un seuil dynamique donne des résultats tout à fait acceptables sans l'intervention d'un utilisateur.

5.2 Automate de couleurs

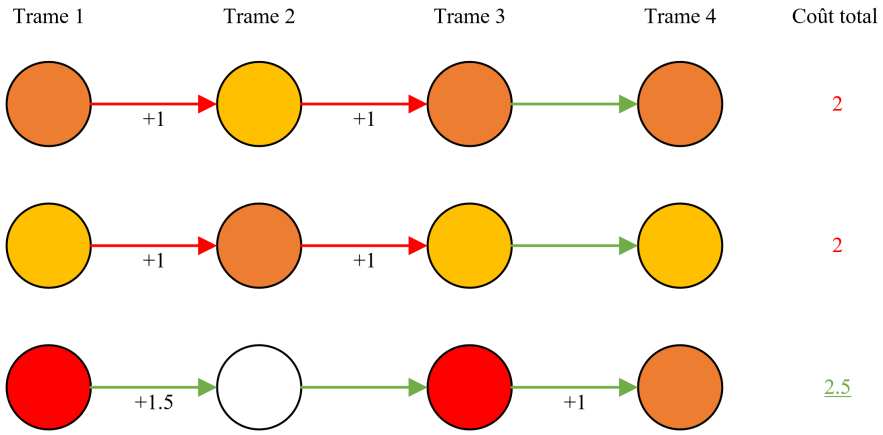
Dû à des mauvais réglages de paramètres, l'idée d'utiliser un automate comme modèle d'apparence a été abandonné une fois, avant de finalement être adoptée. En effet, les performances du suivi peuvent facilement être inférieures à celles obtenues sans modèle d'apparence lorsque :

1. Les occlusions ou détections manquantes ne sont pas traitée adéquatement. Comme il est impossible de calculer l'histogramme de couleurs d'une détection manquante, l'automate doit être en mesure de bien définir le coût d'utiliser une occlusion comme valeur de transition. S'il est trop bas, le solveur CP préférera utiliser ces transitions trop souvent et les trajectoires résultantes seront éparses et fractionnées.
2. Les transitions acceptées entre classes de couleurs sont soient trop permissives, soit trop restrictives. Lorsque trop permissif, en plus d'être plus long à résoudre dû à l'augmentation des possibilités, il est possible de se retrouver avec des trajectoires peu probables. À l'inverse, interdire un trop grand nombre de transitions augmente l'utilisation de la transition occlusion, ce qui augmente le nombre de trajectoires inutilement et affecte négativement le suivi.

En considérant ces deux dernières situations, le coût pour une détection manquante a été fixé à la même valeur que le coût maximal accepté pour toutes transitions. La valeur p_a correspondra à ce coût pour les prochains résultats.

De plus, choisir la bonne manière de combiner les coûts de chaque trajectoire individuelle en un seul objectif de minimisation peut faire une différence. La première hypothèse énoncée pour gérer la combinaison des coûts à été d'utiliser une fonction minmax. En minimisant la valeur maximum, le solveur peut déterminer la trajectoire qui est problématique et y apporter des modifications. Ainsi, on s'assure que la pire trajectoire n'est jamais trop mauvaise, ce qui assure une certaine qualité de suivi minimale pour toutes les trajectoires. En pratique, il s'est avéré que ce choix pose problème dans plusieurs situations pour le problème de suivi. La situation illustrée à la figure 5.1 se produit fréquemment. L'automate se voit obligé d'utiliser l'occlusion comme valeur de transition puisqu'une détection est manquante, ce qui augmente le coût associé à la trajectoire. Dès lors, le solveur ne verra aucun bénéfice à ajuster les trajectoires dont le coût total est inférieur au coût de transition vers l'occlusion et ce, peu importe la simplicité de la modification à apporter pour corriger la situation.

Figure 5.1 Exemple de situation où l'utilisation du maximum comme fonction d'agrégation est problématique. Les cercles correspondent aux détections disponibles. Chaque cercle est coloré selon la classe de couleurs identifiée par le modèle d'apparence. Les flèches en rouge représentent les transitions qui auraient pu facilement être corrigées pour obtenir un suivi de meilleur qualité. Or, comme la troisième trajectoire est celle ayant un coût maximal, le solveur ne cherche pas à améliorer les deux premières, même en présence d'une situation évidente où une seule permutation serait suffisante.



Remplacer la fonction d'agrégation de la valeur maximum par celle effectuant la somme des coûts de trajectoires permet d'éviter cette situation. Toutefois, la rétro-propagation est plus ardue pour le solveur puisqu'il est difficile d'identifier une trajectoire plus problématique que les autres. Moins de solutions seront trouvées en un même intervalle de temps.

Le tableau 5.3 résume les différents tests effectués pour obtenir l'automate le plus efficace possible. On y constate que l'automate 2 utilisant la somme comme fonction d'agrégation où p_a est fixé à 2.5 est la combinaison obtenant la meilleure qualité de suivi. Les autres coûts de transitions sont des nombres décimaux variant entre 0.5 et 5. Ceux-ci sont obtenus en calculant la distance de Bhattacharyya chaque vecteurs constituant une classe de couleurs. Les deux automates de la figure 3.1b sont utilisés dans ces tests et sont indiqués par les étiquettes A1 et A2 dans le tableau.

Tableau 5.3 Comparaisons des résultats pour différents réglages de l'automate de couleurs. A1 et A2 représente les deux automates de la figure 3.1b. Sols : nombre solutions trouvées. (vérité terrain, 2D MOT 2015)

Séquences	**A1 (Max)**				**A2 (Max)**				**A2 (Somme)**	
	$p_a = 1$		**$p_a = 2$**		**$p_a = 1$**		**$p_a = 2$**		**$p_a = 2.5$**	
	MOTA	Sols	MOTA	Sols	MOTA	Sols	MOTA	Sols	MOTA	Sols
Venice-2	92.59	89	96.11	33	97.22	16	98.52	10	**99.63**	3
KITTI-17	-	0	79.24	17	-	0	77.98	125	**91.90**	3
KITTI-13	68.75	4	71.43	1	72.32	9	82.14	56	**85.71**	31
ADL-Rundle-8	69.69	26	86.42	53	71.06	21	91.34	112	**95.87**	1
ADL-Rundle-6	94.82	8	91.85	31	95.93	4	96.30	12	**100.00**	10
ETH-Pedcross2	89.25	28	89.25	28	**97.50**	4	**97.50**	4	96.25	7
ETH-Sunnyday	92.34	15	90.32	21	96.77	3	96.77	7	**98.79**	5
ETH-Banhof	88.58	1	91.70	101	**95.85**	4	95.85	23	88.58	12
PETS09-S2L1	89.24	5	89.24	5	92.83	2	92.83	2	**100.00**	1
TUD-Campus	73.28	1	85.88	81	69.85	56	89.31	98	**91.60**	3
TUD-Stadmitte	74.93	40	92.92	38	83.38	27	92.64	68	**97.00**	2
Global	**83.72**	**217**	**88.82**	**409**	**87.88**	**146**	**92.53**	**517**	**95.71**	**78**

Comme les boîtes englobantes sont de forme rectangulaire mais qu'un usager de la route possède une forme qui lui est propre, une proportion non-négligeable d'arrière-plan est considérée lors de la description d'une détection. Pour vérifier à quel point ces pixels d'arrière-plan faussent la description de l'objet, des tests sur certaines parties d'histogrammes, soit la moitié du haut et du bas ont été effectués. Ces séparations ont du sens pour des piétons puisque si une personne se démarque par son chandail ou son pantalon, l'apparence sera fortement influencée par ce caractère discriminant, contrairement à un histogramme sur la région complète de l'image. Cette hypothèse n'a toutefois pas été confirmée comme le tableau 5.4 permet de le constater. Cela est probablement dû au fait que la moitié haut ou bas de la personne sera plus sensible à un mauvais positionnement du rectangle englobant.

Finalement, des tests supplémentaires ont été réalisés sur une partie de l'ensemble d'entraînement d'UA-DETRAC. Les tableaux 5.5 et 5.6 affichent les performances de suivi obtenus avec et sans automate pour une résolution par bloc versus d'un seul trait. En plus d'offrir de meilleurs temps de résolution, la résolution par blocs permet également de bonifier la valeur ajoutée de l'automate en augmentant la qualité du suivi. La raison qui semble la plus probable

Tableau 5.4 Résultats selon la variation de la région utilisée pour le calcul de l'histogramme (détections de références, 2D MOT 2015)

Séquences	Histogramme Complet				Demi Histograme (Haut)				Demi Histograme (Bas)			
Trames 1-50	MOTA	IDS	FN	FP	MOTA	IDS	FN	FP	MOTA	IDS	FN	FP
Venice-2	**99.63**	2	0	0	**99.63**	2	0	0	**99.63**	2	0	0
KITTI-17	**92.91**	15	1	12	92.66	17	1	11	85.06	33	0	26
KITTI-13	**87.50**	7	1	6	79.46	9	0	14	85.71	6	0	10
ADL-Rundle-8	**95.87**	14	0	7	95.47	16	0	7	**95.87**	14	0	7
ADL-Rundle-6	**100.00**	0	0	0	98.52	4	0	0	99.26	2	0	0
ETH-Pedcross2	**96.25**	9	0	6	**96.25**	9	0	6	95.75	8	1	8
ETH-Sunnyday	**98.79**	1	0	2	97.18	3	0	4	**98.79**	1	0	2
ETH-Banhof	88.24	20	0	14	88.24	20	0	14	**91.00**	15	0	11
PETS09-S2L1	**100.00**	0	0	0	**100.00**	0	0	0	**100.00**	0	0	0
TUD-Campus	**91.60**	15	0	7	**91.60**	15	0	7	**91.60**	15	0	7
TUD-Stadmitte	97.00	8	0	3	**97.55**	6	0	3	**97.55**	6	0	3

est que de considérer une instance plus petite du problème de suivi permet une résolution locale plus en profondeur, ce qui apporte plus de bénéfice que de considérer le problème dans son ensemble. Les temps de résolutions sont également plus faibles, mais surtout plus équilibrés lors de la résolution par blocs. Dans cette situation, le temps de résolution maximal fixé s'applique à chaque bloc individuellement. Or, appliquer une limite de temps sur une séquence complète de longueur variable favorise les séquences plus courtes au détriment de celles plus longues. D'ailleurs, la résolution globale ne permet pas toujours de trouver une solution dans les délais fixés comme pour les séquences MVI_40191 et MVI_40192 dans le tableau 5.5.

Tableau 5.5 Comparaison entre les résultats avec l'automate de couleurs vs sans sur UA-DETRAC (résolution en un coup). Délai maximal de résolution : 3600 secondes

Séquences	SANS AUTOMATE					AVEC AUTOMATE				
	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)
MVI_39801	62.54	146	1623	49	30.99	**62.58**	144	1623	49	3650.62
MVI_39861	43.73	76	1257	14	17.56	**43.94**	71	1257	14	3637.81
MVI_40191	**90.90**	557	2208	729	226.60	AUCUNE SOLUTION				
MVI_40192	**90.10**	507	844	1439	189.70	AUCUNE SOLUTION				
MVI_40201	**89.17**	237	473	473	79.92	**89.17**	237	473	473	3716.33
Global	**87.46**	1523	6405	2704	544.76	76.11	452	3353	536	11004.76

5.3 Stratégies de branchement

Au niveau de la stratégie de branchement, deux options supplémentaires ont été testées : un ordre lexicographique défini dans l'article et une heuristique de densité maximale pondérée de solutions [30]. Dans le premier cas, il s'agit d'une stratégie très spécifique au problème, où la priorité de toute association variable/valeur est ordonnée selon un critère imposé au

Tableau 5.6 Comparaison entre les résultats avec l'automate de couleurs vs sans sur UA-DETRAC (résolution par blocs multiples) Délai maximal de résolution : par bloc

Séquences	SANS AUTOMATE					AVEC AUTOMATE				
	MOTA	IDS	FN	FP	Exec	MOTA	IDS	FN	FP	Temps (s)
MVI_39801	62.72	137	1623	49	30.36	**63.09**	119	1623	49	1653.53
MVI_39861	43.73	76	1257	14	18.14	**44.74**	52	1257	14	892.84
MVI_40191	91.17	453	2208	729	213.89	**91.27**	416	2207	728	1744.72
MVI_40192	90.45	410	844	1439	184.92	**90.57**	378	842	1437	984.74
MVI_40201	89.42	210	473	473	87.56	**89.48**	203	473	473	2311.71
Global	87.74	1286	6405	2704	534.88	**87.88**	1168	6402	2701	7587.54

solveur. Dans le deuxième cas, il s'agit d'une stratégie générique choisissant de brancher sur l'association variable/valeur apparaissant dans la plus grande proportion de bonnes solutions aux contraintes individuelles. Comme la stratégie par défaut du solveur, fixer les variables au plus petit domaine d'abord, converge trop peu rapidement vers une solution, elle a été écartée d'emblée. Le tableau 5.7 explicite les temps de calcul obtenus pour certaines séquences d'entraînement du 2D MOT Challenge (détections de référence). Les deux automates de couleurs (A1 et A2) ont été testés avec chaque stratégie. Les performances MOTA ont été exclus des résultats celles-ci ne sont pas représentatives des résultats actuels ; des ajustements ont été effectués à l'automate de manière subséquente.

Tableau 5.7 Temps de calcul en secondes pour deux stratégies de branchement : LexicoDyn décrite à la section 4.4.2 et CostMaxSD, à la section 2.3.2. Les deux ont été testées avec chaque automate de couleurs (résolution en 20 minutes max)

Séquences	Lexico Dyn		Cost Max SD	
	A1	A2	A1	A2
Venice-2	1208.11	1208.56	1215.55	1209.14
KITTI-13	0.38	1208.09	1209.55	1209.05
ADL-Rundle-8	1207.88	1207.89	1216.37	1208.54
ADL-Rundle-6	0.97	1.05	1228.82	1.18
ETH-Pedcross2	1208.07	1.07	1211.66	1208.39
ETH-Sunnyday	1207.35	0.74	1209.35	2.11
ETH-Banhof	0.35	0.68	0.74	0.72
PETS09-S2L1	1207.54	1.02	1210.54	0.84
TUD-Campus	0.50	1208.40	0.86	1207.93
TUD-Stadmitte	0.73	1208.07	1209.64	1205.96
Total	6041.88	6045.55	9713.09	7253.86

Les temps de calcul observés permettent de constater que la stratégie de branchement par heuristique de densité maximale pondérée de solutions est rarement la plus performante et ce, pour les deux automates de couleurs. La séquence PETS09-S2L1 où l'automate A2 et la stratégie Cost Max SD est la combinaison la plus rapide fait exception, mais l'amélioration

apportée est négligeable dans le contexte global. Il est difficile d'en connaître la raison exacte ; il faudrait examiner les branchements faits pour chaque stratégie durant l'heure de travail qui a été accordée. Toutefois, il est possible d'affirmer que lorsque le temps de résolution est sous les 1200 secondes, la solution optimale a été trouvée. Ainsi, l'hypothèse la plus probable est que la solution optimale ne se situe pas toujours dans une branche comportant une grande quantité de solutions. Autrement, la branche en question de l'arbre de solutions aurait été favorisée par la seconde stratégie de branchements. L'automate n'ayant pas encore été défini sous sa forme actuelle à cette étape, la qualité du suivi a été exclue des résultats.

Des tests supplémentaires ont été effectués pour déterminer l'impact des prédictions de valeurs basées sur la distance sur la qualité de la solution. Sans ces prédictions, le solveur choisit les valeurs à tester selon un ordre croissant, où les valeurs les plus proches de zéro sont priorisées. Dans le cas du suivi, cela signifie que le solveur testera toujours d'abord les plus petits identifiants de trajectoires, ce qui représente un choix arbitraire puisque pour la première trame, les détections sont fixées aux trajectoires de manière aléatoire. Les tests à l'aide des détections de référence et sans optimisation ont été réalisés sur les séquences d'entraînement de 2D MOT 15. Pour chaque séquence, la méthode disposait de 60 secondes pour produire un suivi des 50 premières trames. Cette quantité de trames arbitraire est un intermédiaire entre un nombre trop petit de trames où les résultats seraient difficilement généralisable et un nombre trop grand qui ne permettrait pas à l'automate d'optimiser suffisamment le suivi dans les délais impartis. Selon le tableau 5.8, la majorité des séquences (6 sur 11) ont vu leur efficacité de suivi augmenter suite à l'ajout des prédictions alors que seulement trois séquences ont vu une baisse de performance.

Tableau 5.8 Résultats obtenus avec la stratégie de branchement lexicographique avec et sans prédictions selon la distance (résolution en une minute max)

Séquences	**Sans prédictions**					**Avec prédictions**				
Trames 1-50	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)
Venice-2	99.63	2	0	0	61.83	**100.00**	0	0	0	62.02
KITTI-17	94.94	19	1	0	61.05	**96.20**	15	0	0	61.08
KITTI-13	**91.96**	7	2	0	61.00	90.18	10	1	0	61.03
ADL-Rundle-8	97.05	15	0	0	61.73	**100.00**	0	0	0	61.90
ADL-Rundle-6	**100.00**	0	0	0	76.64	**100.00**	0	0	0	61.96
ETH-Pedcross2	**97.25**	11	0	0	60.88	97.00	12	0	0	60.95
ETH-Sunnyday	**99.60**	1	0	0	60.96	99.19	2	0	0	60.96
ETH-Banhof	91.70	24	0	0	60.93	**97.92**	6	0	0	60.93
PETS09-S2L1	**100.00**	0	0	0	61.28	**100.00**	0	0	0	61.04
TUD-Campus	93.89	16	0	0	60.85	**98.86**	3	0	0	61.05
TUD-Stadmitte	97.82	8	0	0	60.84	**99.18**	3	0	0	60.90

5.4 Évaluation de la robustesse aux détections

Dans le but de tester la robustesse de la méthode d'ajout de détections proposée par ce travail, une série de tests supplémentaires a été réalisée. La vérité terrain réduite permet de simuler des situations où le détecteur ne serait pas parvenu à détecter certains usagers de la route. Deux paramètres ont permis de contrôler le nombre de détections retirées :

1. **La fréquence d'omission (f)** : Les détections sont retirées à chaque f trames ;
2. **Le nombre de détections à retirer (r)** : À chaque trame identifiée, un nombre égal à r détections est retiré.

Le tableau 5.9 permet de visualiser les métriques de performance obtenues selon la variation des deux paramètres de retrait de détections pour l'ensemble des séquences d'entraînement du 2D MOT 15. Une limite de temps de une heure a été attribuée au solveur CP. Il est possible de noter qu'en utilisant la vérité terrain complète, le MOTA obtenu est de 98.67 ($IDS = 47$, $FN = 1$ et $FP = 0$).

Tableau 5.9 Résultats sur la version réduite de la vérité terrain des séquences d'entraînement de 2D MOT 2015 selon deux paramètres : 1. à chaque f trame une série de plusieurs détections (r) est retirée. (résolution en une heure max)

		f = 10 trames					f = 5 trames					f = 3 trames				
		MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)
	2	93.66	120	109	0	3464.10	89.07	177	218	0	3556.89	82.82	249	372	0	3378.26
r	3	91.78	137	160	0	3412.50	85.70	196	321	0	3493.47	77.84	250	551	0	3489.70
	5	89.35	135	250	0	3361.73	80.33	205	506	0	3396.61	68.95	254	868	0	3442.61

En analysant les précédents résultats (tableau 5.9), il est possible de constater que non seulement le nombre d'erreurs de type faux-négatif augmente en fonction de l'augmentation du nombre de détections retirées, mais que le nombre de mauvaises associations augmente également. L'effet sur les mauvaises associations est plus marqué lorsque la fréquence d'omission augmente par rapport à une augmentation du nombre de détections omises par trames. La conclusion est donc que le modèle CP peut avoir de la difficulté à gérer les occlusions de courte durée, mais plus fréquentes dans le temps. Plusieurs raisons peuvent expliquer ce comportement :

1. Les prédictions pour la stratégie de branchement CP sont seulement faites pour deux trames consécutives. Or, en présence d'une occlusion, la valeur de distance n'est pas disponible. Ainsi, le solveur teste les valeurs de trajectoires en ordre croissant pour la variable en question.

2. L'utilisation de seuils sur l'écart de positions des détections fonctionne de telle sorte qu'une distance maximale entre deux détections de trames consécutives soit permise. En situation d'occlusion avec une durée d'une seule trame, ce seuil est doublé, ce qui peut laisser place à une situation où un trop grand nombre de candidats sont disponibles pour l'association.

5.5 Efficacité du pré-traitement

Un élément important de la méthode proposée est le pré-traitement effectué sur les détections à l'aide d'une méthode de SOT (voir section 3.3 pour plus de détail). Ce module a d'abord été testé sur une version réduite des détections parfaites pour l'ensemble d'entraînement du 2D MOT 2015. Les résultats sont présentés dans le tableau 5.10. Malgré l'ajout de quelques erreurs de type faux-positif, cette approche permet de combler près de la moitié des trous ajoutés par la réduction des détections, ce qui diminue fortement le nombre de mauvaises associations.

Tableau 5.10 Efficacité du pré-traitement sur le 2D MOT 2015 à l'aide de la vérité terrain réduite ($f = 5$, $r = 3$, résolution en 5 minutes max)

Séquences	**Sans pré-traitement**					**Avec pré-traitement**				
Trames 1-100	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)
Venice-2	91.10	27	60	0	308.62	**97.65**	7	16	0	326.05
KITTI-17	82.60	51	60	0	305.26	**85.58**	50	37	5	309.39
KITTI-13	71.18	37	46	0	305.02	**71.53**	37	44	1	306.09
ADL-Rundle-8	88.39	48	60	0	315.54	**95.59**	21	17	3	317.54
ADL-Rundle-6	87.98	15	60	0	310.55	**93.27**	7	35	0	328.88
ETH-Pedcross2	87.13	43	60	0	311.07	**90.50**	40	34	2	319.07
ETH-Sunnyday	82.63	23	59	0	306.71	**87.71**	24	34	0	306.50
ETH-Banhof	84.01	53	61	0	314.97	**92.15**	28	28	0	308.69
PETS09-S2L1	82.22	33	60	0	318.84	**83.56**	35	51	0	305.60
TUD-Campus	77.99	34	45	0	339.79	**85.52**	24	28	0	308.39
TUD-Stadmitte	88.20	22	60	0	313.88	**92.23**	18	36	0	309.05
Global	85.51	386	631	0	3450.24	**90.57**	291	360	11	3445.25

Après avoir constaté que le pré-traitement permet de combler certaines détections manquantes, une série de tests supplémentaires a été conduite sur les mêmes séquences du 2D MOT 2015, mais cette fois en utilisant les détections publiques. Le défi est plus grand, puisque le risque de se baser sur une fausse détection pour en ajouter de nouvelles est présent. De plus, les détections manquantes ne sont plus réparties de manière uniforme ; les petits objets peuvent être ignorés pendant plus d'une trame. Le nombre de trames considérées a été diminué puisque résoudre le problème du suivi multiobjet avec des détections imparfaites demande plus de calculs. Le tableau 5.11 permet de constater qu'encore une fois,

le pré-traitement augmente la valeur du MOTA en diminuant le nombre de mauvaises associations et de faux-négatifs. Les mêmes conclusions ont été tirées des résultats présentés dans le tableau 5.12 qui présente les résultats sur cinq séquences provenant de l'ensemble d'entraînement d'UA-DETRAC.

Tableau 5.11 Efficacité du pré-traitement sur le 2D MOT 2015 à l'aide des détections publiques (résolution en 5 minutes max)

Séquences	**Sans pré-traitement**					**Avec pré-traitement**				
Trames 1-100	MOTA	IDS	FN	FP	Temps (s)	MOTA	IDS	FN	FP	Temps (s)
Venice-2	**33.15**	12	310	39	305.09	**33.15**	12	306	43	309.92
KITTI-17	12.91	30	314	0	302.61	**16.96**	27	301	0	305.14
KITTI-13	**20.54**	10	77	2	306.70	**20.54**	10	77	2	303.29
ADL-Rundle-8	16.93	20	332	70	305.53	**17.52**	16	323	80	308.57
ADL-Rundle-6	33.70	13	144	22	305.54	**36.67**	11	137	23	309.44
ETH-Sunnyday	22.58	2	190	0	306.79	**25.00**	2	184	0	303.22
ETH-Banhof	**31.49**	7	185	6	306.73	31.14	7	184	8	303.45
PETS09-S2L1	**73.09**	15	30	15	304.17	68.16	15	32	24	304.20
TUD-Campus	40.46	29	126	1	305.02	**43.89**	27	116	4	303.99
TUD-Stadmitte	64.03	14	114	4	303.61	**65.40**	15	104	8	305.36
Global	30.05	152	1822	159	3051.79	**31.02**	142	1764	192	3056.59

Tableau 5.12 Efficacité du pré-traitement sur DETRAC, sans la contrainte *regular*

Séquences	**Sans pré-traitement**					**Avec pré-traitement**				
	MOTA	IDS	FN	FP	Exec	MOTA	IDS	FN	FP	Exec
MVI_39801	60.99	157	1710	26	7.15	**62.54**	146	1623	49	30.99
MVI_39861	39.81	98	1341	2	5.71	**43.73**	76	1257	14	17.56
MVI_40191	90.53	602	2394	641	53.51	**90.90**	557	2208	729	226.60
MVI_40192	89.96	550	954	1327	36.95	**90.10**	507	844	1439	189.70
MVI_40201	88.38	301	567	401	10.74	**89.17**	237	473	473	79.92
Global	86.94	1708	6966	2397	114.05	**87.46**	1523	6405	2704	544.76

CHAPITRE 6 CONCLUSION ET RECOMMANDATIONS

Ce travail présente une nouvelle approche de suivi multiobjet basée sur la programmation par contraintes. Au mieux de nos connaissances, il s'agit de la première tentative effectuée pour exprimer ce problème à l'aide de la CP. Les résultats obtenus sur les séquences provenant d'UA-DETRAC démontre sa compétitivité dans le domaine du suivi de véhicules routiers. Toutefois, tel que démontré par les résultats obtenus sur l'ensemble de données 2D MOT 2015, cette approche ne parvient pas à atteindre des performances au niveau de l'état de l'art pour le suivi de piéton.

6.1 Synthèse des travaux

Voici un résumé des réalisations effectuées au cours de ce travail de recherche :

1. Un modèle CP proposant une solution au problème d'association des données du suivi multiobjet a été présenté. Ce modèle se base principalement sur la distance entre les boîtes englobantes détectées ainsi que sur la répartition des couleurs pour chacune d'entre elles.
2. Une stratégie de branchement adaptée au problème a été développée pour accélérer la résolution et augmenter la qualité du suivi final. En plus d'intégrer un système de prédiction des associations les plus probables, cette stratégie brise les nombreuses symétries inhérentes au problème de suivi.
3. Un processus d'amélioration des détections a été ajouté au pipeline de résolution. En plus de filtrer les détections dont la validité est remise en question, d'autres sont ajoutées à l'aide d'une technique de suivi d'un seul objet appliquée sur chaque détection séparément.
4. Des comparaisons avec les méthodes de l'état de l'art dans deux domaines d'application : le suivi de véhicules routiers et le suivi de piétons ont été réalisées.

6.2 Limitations de la solution proposée

Le cadre offert par la programmation par contraintes impose certaines limites sur la manière de représenter le problème. La première provient du fait qu'il n'est pas possible d'ajouter des variables supplémentaires au cours de la résolution. Pour le suivi multiobjet, le nombre maximal de trajectoires doit donc être déterminé à l'avance, ce qui peut être problématique

dans une situation où trop peu de trajectoires sont disponibles pour le nombre d'usagers de la route faisant leur apparition dans la scène. Dans un tel scénario, aucune solution ne peut être trouvée, même partielle, et la résolution doit être recommencée avec un plus grand nombre de trajectoires possibles. Une deuxième limite imposée par la CP provient du fait que certaines des contraintes disponibles ne se propagent pas suffisamment pour permettre leur utilisation dans le cadre de ce projet. Par exemple, l'utilisation de contraintes du type : SI x ALORS y serait en mesure de simplifier la logique du modèle CP exprimé au cours de cette recherche.

À l'heure actuelle, la méthode de suivi développée permet d'atteindre des performances optimales en utilisant une très grande quantité de temps, même pour une taille de voisinage restreinte, c'est-à-dire le nombre de trames considérées en une seule résolution CP. Il faut également tenir compte du fait qu'une limite de temps est accordée à chaque processus de résolution. Comme il n'est pas possible d'obtenir une solution optimale dans aucune situation, il est nécessaire de faire un compromis : sacrifier la taille du voisinage pour obtenir des solutions mieux optimisées localement avec un temps de résolution plus court, ou choisir un grand voisinage et résoudre le problème avec une vue d'ensemble plus large, au prix d'un temps de résolution beaucoup plus grand pour un même niveau d'optimisation.

Bien que le suivi de piétons et de véhicules ait été réalisé de manière indépendante, combiner les deux à la fois pourrait se révéler problématique. Comme l'utilisation du seuil dynamique de distance (voir la section 3.2) ne s'est révélé la meilleure stratégie que sur les tests réalisés sur les piétons et que ces derniers n'ont pas le même type de trajectoire que des automobiles, il pourrait être ardu de trouver les bons paramètres pour optimiser le suivi pour des objets avec des vitesses hétérogènes. Ainsi, il faudrait soit appliquer la méthode sur chaque classe d'objets d'intérêt, soit se contenter d'un suivi général de moins bonne qualité.

6.3 Améliorations futures

Comme il s'agissait d'un premier travail combinant CP et suivi multiobjet, la majeure partie du travail a été de produire un modèle CP robuste, dont les contraintes s'adaptent bien aux caractéristiques de la problématique, soit être en mesure de formuler des contraintes se basant sur l'apparence et le déplacement de chaque objet d'intérêt qui soient faciles à propager par le solveur. Les modèles d'apparence et de mouvement formulés sont très simples pour une méthode de suivi dans l'état de l'art actuel et pourraient être grandement améliorés.

Dans le cas du modèle de mouvement, la priorité est de ne pas trop contraindre durement le nombre d'associations possibles de telle sorte que les bonnes associations entre trajectoires

et détections soient rejetées. L'amélioration la plus pertinente à ce niveau serait de parfaire l'ordre dans lesquels les détections candidates sont testées pour chaque variable CP de trajectoires. En utilisant certaines informations provenant du suivi préliminaire appliqué avant la CP pour ajouter des détections comme la position de l'objet dans la trame suivante, il serait possible d'estimer une vitesse et de se baser sur la position future au lieu de la dernière position observée dans la trajectoire.

Comme le modèle d'apparence utilisé ne se base que sur des informations simples, c'est-à-dire la couleur, beaucoup d'améliorations sont possibles à ce niveau. Utiliser des descripteurs plus précis sur la couleur en considérant la répartition de celle-ci, mais aussi intégrer d'autres catégories d'informations comme les contours et les textures. Les caractéristiques profondes qui offrent de l'information à plusieurs échelles sur chaque région d'intérêt sont celles qui semblent offrir le plus de potentiel étant donné les progrès observés pour les classificateurs à apprentissage profond. Il faut toutefois considérer que l'utilisation d'un vecteur ayant trop d'éléments peu ralentir encore plus la résolution sans un algorithme de regroupements adéquat, par exemple celui des K-moyens.

La méthode d'ajouts de détections précédent la résolution CP pourrait également être bonifiée de plusieurs façons. En effet, l'utilisation de techniques de suivi d'un seul objet offre davantage de perspective que seulement l'ajout de détections. Tel que mentionné, les informations sur la position des détections utilisées à cette étape pourrait permettre de préciser le modèle de mouvement. De manière encore plus poussée, tout le suivi pourrait être conservé et donné au solveur en tant que solution initiale du problème. Le solveur pourrait ensuite considérer les options les plus avantageuses à l'aide du modèle de mouvement et d'apparence.

RÉFÉRENCES

[1] L. Leal-Taixé *et al.*, "Motchallenge 2015 : Towards a benchmark for multi-target tracking," 2015.

[2] P. Dollar *et al.*, "Fast feature pyramids for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, nº. 8, p. 1532–1545, août 2014. [En ligne]. Disponible : https://doi.org/10.1109/TPAMI.2014.2300479

[3] R. Girshick *et al.*, "Rich feature hierarchies for accurate object detection and semantic segmentation," dans *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, p. 580–587.

[4] K. He *et al.*, "Mask r-cnn," dans *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, p. 2980–2988.

[5] S. Ren *et al.*, "Faster r-cnn : Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, nº. 6, p. 1137–1149, June 2017.

[6] E. Bochinski, T. Senst et T. Sikora, "Extending iou based multi-object tracking by visual information," dans *IEEE International Conference on Advanced Video and Signals-based Surveillance*, Auckland, New Zealand, nov. 2018, p. 441–446. [En ligne]. Disponible : http://elvera.nue.tu-berlin.de/files/1547Bochinski2018.pdf

[7] A. Andriyenko et K. Schindler, "Multi-target tracking by continuous energy minimization," dans *CVPR 2011*, June 2011, p. 1265–1272.

[8] W. Choi et S. Savarese, "Multiple target tracking in world coordinate with single, minimally calibrated camera," dans *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos et N. Paragios, édit. Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, p. 553–567.

[9] M. He *et al.*, "Fast online multi-pedestrian tracking via integrating motion model and deep appearance model," *IEEE Access*, vol. PP, p. 1–1, 07 2019.

[10] J. F. Henriques *et al.*, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, nº. 3, p. 583–596, March 2015.

[11] B. Chen *et al.*, "Real-time 'actor-critic' tracking," dans *ECCV*, 2018.

[12] L. Ren *et al.*, "Collaborative deep reinforcement learning for multi-object tracking," dans *ECCV*, 2018.

[13] P. Bergmann, T. Meinhardt et L. Leal-Taixé, "Tracking without bells and whistles," dans *The IEEE International Conference on Computer Vision (ICCV)*, 03 2019.

[14] C. Kuo, C. Huang et R. Nevatia, "Multi-target tracking by on-line learned discriminative appearance models," dans *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, p. 685–692.

[15] B. Yang et R. Nevatia, "An online learned crf model for multi-target tracking," dans *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, p. 2034–2041.

[16] O. Tuzel, F. Porikli et P. Meer, "Region covariance : A fast descriptor for detection and classification," dans *ECCV (2)*, 2006, p. 589–600. [En ligne]. Disponible : https://doi.org/10.1007/11744047_45

[17] L. Leal-Taixé, C. Canton-Ferrer et K. Schindler, "Learning by tracking : siamese cnn for robust target association," *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR). DeepVision : Deep Learning for Computer Vision.*, 2016.

[18] B. Uzkent et Y. Seo, "Enkcf : Ensemble of kernelized correlation filters for high-speed object tracking," dans *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2018, p. 1133–1141.

[19] W. Feng *et al.*, "An online learned crf model for multi-target tracking," dans *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[20] A. Milan *et al.*, "Online multi-target tracking using recurrent neural networks," dans *AAAI*, 2016.

[21] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, n°. 1-2, p. 83–97, 1955.

[22] S. H. Rezatofighi *et al.*, "Joint probabilistic data association revisited," dans *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, p. 3047–3055.

[23] Y. Bar-Shalom, F. Daum et J. Huang, "The probabilistic data association filter," *IEEE Control Systems Magazine*, vol. 29, n°. 6, p. 82–100, Dec 2009.

[24] A. G. Daronkolaei, S. Shiry et M. B. Menhaj, "Multiple target tracking for mobile robots using the jpdaf algorithm," dans *19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007)*, vol. 1, Oct 2007, p. 137–145.

[25] P. Chu *et al.*, "Online multi-object tracking with instance-aware tracker and dynamic model refreshment," dans *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Jan 2019, p. 161–170.

[26] G. Pesant, “A constraint programming primer,” *EURO J. Computational Optimization*, vol. 2, n°. 3, p. 89–97, 2014. [En ligne]. Disponible : https://doi.org/10.1007/s13675-014-0026-3

[27] J. Régin, “A filtering algorithm for constraints of difference in csps,” dans *Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1.*, 1994, p. 362–367. [En ligne]. Disponible : http://www.aaai.org/Library/AAAI/1994/aaai94-055.php

[28] G. Pesant, “A regular language membership constraint for finite sequences of variables,” dans *Principles and Practice of Constraint Programming – CP 2004*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2004, p. 482–495.

[29] S. Demassey, G. Pesant et L.-M. Rousseau, “A cost-regular based hybrid column generation approach,” *Constraints*, vol. 11, n°. 4, p. 315–333, Dec 2006. [En ligne]. Disponible : https://doi.org/10.1007/s10601-006-9003-7

[30] G. Pesant, “Counting-based search for constraint optimization problems,” dans *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI’16. AAAI Press, 2016, p. 3441–3447. [En ligne]. Disponible : http://dl.acm.org/citation.cfm?id=3016100.3016386

[31] P. Van Hentenryck et L. Michel, “The steel mill slab design problem revisited,” dans *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, L. Perron et M. A. Trick, édit. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008, p. 377–381.

[32] S. Lloyd, “Least squares quantization in pcm,” *IEEE Transactions on Information Theory*, vol. 28, n°. 2, p. 129–137, March 1982.

[33] L. Wen *et al.*, “DETRAC : A new benchmark and protocol for multi-object detection and tracking,” *arXiv CoRR*, vol. abs/1511.04136, 2015.

[34] D. Riahi et G.-A. Bilodeau, “Online multi-object tracking by detection based on generative appearance models,” *Computer Vision and Image Understanding*, vol. 152, p. 88 – 102, 2016.

[35] A. Sadeghian, A. Alahi et S. Savarese, “Tracking the untrackable : Learning to track multiple cues with long-term dependencies,” dans *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, p. 300–311.

[36] S. Bae et K. Yoon, “Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning,” dans *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, p. 1218–1225.

[37] H. Pirsiavash, D. Ramanan et C. C. Fowlkes, “Globally-optimal greedy algorithms for tracking a variable number of objects,” dans *CVPR 2011*, June 2011, p. 1201–1208.

[38] N. Saunier, T. Sayed et K. Ismail, “Large-scale automated analysis of vehicle interactions and collisions,” *Transportation Research Record*, vol. 2147, n°. 1, p. 42–50, 2010. [En ligne]. Disponible : https://doi.org/10.3141/2147-06

[39] M. Wallace, “Practical applications of constraint programming,” *Constraints*, vol. 1, n°. 1/2, p. 139–168, 1996.

[40] Y. Yang et G.-A. Bilodeau, “Multiple object tracking with kernelized correlation filters in urban mixed traffic,” *2017 14th Conference on Computer and Robot Vision (CRV)*, p. 209–216, 2016.

[41] H. Ooi *et al.*, “Multiple object tracking in urban traffic scenes with a multiclass object detector,” dans *ISVC*, ser. Lecture Notes in Computer Science, vol. 11241. Springer, 2018, p. 727–736.

[42] J. Jodoin, G. Bilodeau et N. Saunier, “Tracking all road users at multimodal urban traffic intersections,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, n°. 11, p. 3241–3251, Nov 2016.

[43] N. Saunier et T. Sayed, “A feature-based tracking algorithm for vehicles in intersections,” dans *The 3rd Canadian Conference on Computer and Robot Vision (CRV’06)*, June 2006, p. 59–59.

[44] L. Chen *et al.*, “Online multi-object tracking with convolutional neural networks,” dans *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, p. 645–649.

[45] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Res. Logist. Quart.*, vol. 2, p. 83 – 97, 03 1955.

[46] K. Bernardin et R. Stiefelhagen, “Evaluating multiple object tracking performance : The clear mot metrics,” *EURASIP Journal on Image and Video Processing*, vol. 2008, n°. 1, p. 246309, May 2008. [En ligne]. Disponible : https://doi.org/10.1155/2008/246309

[47] E. Ristani *et al.*, “Performance measures and a data set for multi-target, multi-camera tracking,” dans *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II*, 2016, p. 17–35.

[48] L. Leal-Taixé *et al.*, “MOTChallenge 2015 : Towards a benchmark for multi-target tracking,” *arXiv :1504.01942 [cs]*, avr. 2015, arXiv : 1504.01942. [En ligne]. Disponible : http://arxiv.org/abs/1504.01942