Simuler le trafic routier à partir de données réelles

Vers un outil d'aide à la décision

Alexandre Bonhomme, Philippe Mathieu, Sébastien Picault

Univ. Lille, CNRS, Centrale Lille, UMR 9189 – CRIStAL (équipe SMAC)
Centre de Recherche en Informatique Signal et Automatique de Lille
F-59000 Lille, France
prenom.nom@univ-lille1.fr

RÉSUMÉ. *La simulation microscopique de trafic routier est depuis longtemps un domaine d'application privilégié des systèmes multi-agents (SMA). L'approche centrée individus permet en effet d'intégrer la variété comportementale nécessaire pour rendre compte de situations réelles. Plus récemment, des bases de données géographiques fournissent des informations environnementales précises dans des formats ouverts, offrant ainsi la possibilité de réaliser facilement des simulateurs de trafic à base d'agents, informés en direct de modifications de conditions de circulation. En utilisant ces données et grâce à l'adaptativité des SMA, il est ainsi possible de construire des systèmes d'aide à la décision capables d'intégrer des changements environnementaux et comportementaux de manière immédiate. Ainsi, des scénarios construits sur des hypothèses différentes en matière d'acteurs, de comportements, d'environnement et de flux peuvent être comparés. Nous décrivons ici un processus permettant la réalisation d'un tel outil.*

ABSTRACT. *Microscopic simulations of road traffic are a typical application domain for Multi-Agent Systems. Indeed, the individual-based approach allows to take into account the diversity of behaviors so as to consider real situations. More recently, geographical databases provide environmental information under open formats, which offers the opportunity to design agent-based traffic simulators which can be continuously informed of changes in traffic conditions. The use of such data, together with the adaptability of MAS, allows the realization of decision support systems that are able to integrate environmental and behavioral modifications in a direct way; and compare various scenarios built from different hypothesis in terms of actors, behaviors, environment and flow. We describe here a process which leads to the development of such a tool.*

MOTS-CLÉS : *simulation multi-agent, SIG, trafic routier, générateur de trafic.*

KEYWORDS: *multi-agent simulation, GIS, road traffic, traffic generator.*

DOI:10.3166/RIA.30.329-352

1. Introduction

La simulation de trafic est un domaine où l'approche centrée individus a prouvé très tôt ses avantages (Espié *et al*., 1994 ; Champion *et al*., 1999), notamment pour évaluer l'impact de décisions individuelles sur l'état macroscopique de la circulation routière ou, plus récemment, pour sa capacité à intégrer une variabilité statistique dans la violation des normes par les conducteurs (Lacroix *et al*., 2013).

Une difficulté majeure dans la conception d'un simulateur de trafic tient à la diversité des objectifs souhaités : étude de flux, ergonomie de l'habitacle, effet de modifications de l'infrastructure routière sur le comportement des conducteurs... l'espace concerné et les échelles de temps peuvent changer considérablement ce qui conduit en général à des modèles *ad hoc* (c.à.d. conçus pour un usage particulier), qui sont difficiles à modifier ou adapter (Bazzan, Klügl, 2014).

En fait, dans un domaine comme le transport, la multiplicité des objectifs et des échelles pourrait plutôt être prise en compte comme un cadre général de conception d'outils de simulation, afin de répondre à ces différentes questions à l'aide d'une plate-forme homogène composée de modèles facilement modifiables s'appuyant sur des connaissances explicites. Ainsi, il nous semble possible de construire des bibliothèques de modèles de comportements qui pourraient être réutilisées dans différents contextes et à des fins différentes, sélectionnées selon des hypothèses de simulation et des scénarios d'utilisation spécifiques.

En outre, afin d'assurer un maximum de fiabilité dans les résultats de simulations, il est nécessaire de configurer l'environnement et les comportements des agents à partir de données réelles. En ce qui concerne le comportement des agents, nous avons déjà proposé des méthodes dans ce sens aussi bien pour le transport (Lacroix *et al*., 2013) que pour d'autres domaines (Mathieu, Picault, 2013). Dans cet article, nous montrons que les comportements peuvent être couplés à un environnement construit à partir de données cartographiques réelles, et paramétrés par (ou comparés avec) des informations de flux routiers réels.

Dans ce qui suit, nous présentons en particulier *TrafficGen* (Bonhomme *et al*., 2016), un simulateur de trafic expérimental que nous avons développé comme preuve de concept pour évaluer notre approche. Il n'a pas pour objectif d'entrer en compétition en performance ou en qualité avec des outils professionnels spécialisés comme ArchiSim, SCANeR II ou Synchro studio, mais plutôt de démontrer et d'illustrer la manière de construire un outil d'aide à la décision efficace permettant d'effectuer facilement une mise à jour continue des données, la mise en place aisée de modifications des modèles utilisés ou une comparaison pertinente de différents scénarios.

Cet article est organisé comme suit. Tout d'abord, nous présentons l'état de l'art en simulation de trafic routier, puis nous donnons un aperçu des données ouvertes disponibles pour ce type de simulation, leurs sources et leurs structures. Ensuite, nous décrivons le modèle de comportements suivi dans ce travail puis le workflow que nous utilisons pour l'acquisition, le filtrage et la structuration des données afin de les

intégrer dans *TrafficGen*. Pour finir, nous analysons les résultats des simulations obtenues avec cet outil. A travers cette présentation, nous expliquons par ailleurs comment construire un modèle multi-agent modulaire, facilement extensible et modifiable. Nous illustrons notre propos à l'aide d'une expérience classique, avant de conclure sur les perspectives de ce travail.

2. État de l'art

2.1. Les plateformes de simulation

Il existe à l'heure actuelle une grande diversité de simulateurs de trafic fondés sur des systèmes multi-agents ou des automates cellulaires. Cependant, la plupart de ces applications ont un objectif commercial et ne fournissent donc que très peu de détails de conception. Nous pouvons par exemple citer Synchro Studio [1] avec SimTraffic, Aimsun [2], PTV Vissim [3], Paramics [4] ou encore TransModeler [5] parmi les plus connus.

En marge de ces systèmes, nous trouvons des logiciels d'entreprise comme ArchiSim (Espié *et al*., 1994), SCANeR II (Champion *et al*., 1999) ou Megaffic (Osogami *et al*., 2012) qui reposent également sur le paradigme multi-agent (Chen, Cheng, 2010 ; Bazzan, Klügl, 2014). Néanmoins, ces outils sont dédiés à des questions spécifiques issues de l'IFSTTAR, Renault et IBM, telles que des études ciblées sur la psychologie du conducteur, l'ergonomie de l'habitacle du véhicule ou encore des simulations microscopiques massives. Leurs architectures sont optimisées (en termes de logiciels d'infrastructure et de matériel), les comportements sont dédiés et à peine extensibles. De manière identique, la plateforme DIVAs (Al-Zinati *et al*., 2013) (initiative de l'Université du Texas à Dallas) est plus centrée sur les problèmes de vision de l'agent. D'autres travaux se concentrent sur la question de l'évitement de collisions (Bourbakis, 1997) dans les véhicules autonomes (Bourbakis, Findler, 2001).

Enfin, certaines alternatives *open source* existent, comme MATSim (Raney, Nagel, 2006), SUMO (Krajzewicz *et al*., 2012) ou MITSIMLab (Yang, 1997). Seuls MATSim et SUMO sont basés sur une architecture à base d'agents. Tous deux permettent l'utilisation de données géographiques (SIG). Ils ont aussi l'avantage d'être capables de simuler un grand nombre de véhicules, au prix d'un manque de flexibilité des mécanismes de simulation. Les véhicules ne possèdent qu'un petit nombre de comportements, réductibles à un modèle de suivi de véhicule ainsi qu'un mécanisme de changement de voie.

1. http://www.trafficware.com/products/planninganalysis-software
2. http://www.aimsun.com/
3. http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim/
4. http://www.paramics-online.com/
5. http://www.caliper.com/transmodeler/

Ces comportements simples et naïfs sont également fournis dans la plate-forme polyvalente GAMA (Taillandier *et al*., 2012). Dans ces différents cas, la modification et la facilité d'extension du modèle comportemental restent les principaux problèmes.

2.2. Modèles de comportements

Dans la plupart des simulateurs de trafic, les efforts de modélisation portent essentiellement sur les véhicules, qui sont en général *les seuls agents* existant dans le système. Toute la complexité du trafic est modélisée par la perception, la cognition, les seules capacités d'actions de ces agents, ainsi que leurs comportements et leurs interactions mutuelles. En outre, la majorité des travaux de modélisation essayent de résoudre des problèmes dans des domaines variés : la perception (i.e. perception active (Bourgois *et al*., 2012), voies virtuelles (Bonte *et al*., 2007), cônes de vision (Kuiper, Wenkstern, 2014)), la psychologie du conducteur (Mandiau *et al*., 2008), leur rapport aux normes (Lacroix *et al*., 2009 ; Doniec *et al*., 2008) ou encore l'utilisation de la théorie des jeux pour gérer les conflits dans les carrefours (Mandiau *et al*., 2008).

Bien que la majorité de ces modèles aient fait l'objet de validations séparées, la combinaison de leurs hypothèses sous-jacentes pose évidemment d'importantes questions d'évaluation. Un modèle psychologique précis du conducteur ne garantit pas que la population de véhicules simulés soit plus réaliste au niveau macroscopique. Pour concevoir un outil qui permet de tester et de comparer différents scénarios ou hypothèses, le problème n'est pas de choisir et mettre en œuvre une théorie psychologique particulière ou une autre, mais plutôt de permettre à l'utilisateur :

1. d'accéder *explicitement* (et de manière compréhensible) au modèle de comportement que décrivent ces hypothèses,
2. de choisir ceux à utiliser dans un scénario,
3. de construire facilement des expériences, sans avoir à réécrire l'ensemble du code correspondant.

Avant d'expliquer comment nous mettons en œuvre cette approche dans *TrafficGen*, il est tout d'abord necessaire de décrire le problème de la récupération d'informations cartographiques pour la simulation.

3. Formats de données ouverts

Depuis quelque temps, un nombre croissant d'initiatives visant à fournir des données géographiques dans différents formats ouverts se sont développées, fournissant une opportunité pour, d'une part, construire *à la volée* et mettre à jour des environnements multi-agents pour la simulation de trafic et, d'autre part, reproduire l'état du trafic en temps réel dans une simulation (ou comparer les conséquences d'une hypothèse particulière sur le monde réel). Nous présentons dans cette section certains de ces formats et leur utilisation.

3.1. Données cartographiques

Au cours des quinze dernières années, de nouveaux services en ligne de cartographie et de routage se sont développés. Parmi eux, le projet *OpenStreetMap* (OSM [6]), créé en 2004 au University College de Londres, est une initiative participative basée sur la contribution des utilisateurs (comme Wikipedia). Les éléments fournis par OpenStreetMap sont extrêmement diversifiés (détails et fonctions des bâtiments, limitations de vitesse, informations touristiques, etc.) et par ailleurs ce format de données est ouvert.

Nous pouvons aussi mentionner *OpenDrive* [7], qui est lui aussi un format ouvert, développé cette fois par un groupement d'entreprises privées en 2005 (VIRES Simulationstechnologie GmbH). Ce format se concentre plus spécifiquement sur les caractéristiques de la route (géométrie de la chaussée, pentes, altitude, nature du revêtement, etc.) et a été développé afin de normaliser la simulation de prototypes de véhicules sur des circuits de test.

3.2. Informations de flux

Outre ces informations cartographiques, une simulation doit aussi être capable de prendre en compte des flux de véhicules réalistes. Bien qu'il soit possible d'utiliser des générateurs de flux (i.e. MNTG (Mokbel *et al*., 2013)) qui génèrent des véhicules avec des itinéraires scriptés à partir d'un plan prédéfini, l'usage de données issues du trafic réel est évidemment bien plus pertinent, que ce soit pour calibrer les flux de véhicules simulés ou pour valider les résultats fournis par la simulation. La disponibilité de ces données est en constante augmentation aussi bien à l'échelle nationale que urbaine.

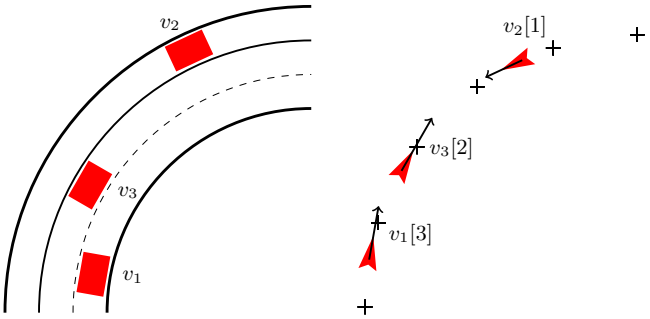
Par exemple, le Centre national de la circulation de l'information français (« Bison Futé ») offre librement et en temps réel (toutes les 6 min.) des données sur les principales routes du réseau routier national et les principales villes Françaises [8]. Ces données utilisent le format Datex II [9] (i.e. le format d'échange européen pour les informations de trafic routier), qui indique les événements (accident de voiture, embouteillage, etc.), mais également les indicateurs de trafic pour un certain nombre de points importants du réseau : débit (nombre de véhicules par période de mesure), taux d'occupation du tronçon routier (rapport entre la longueur de l'ensemble des véhicules sur une section et la longueur de cette section) et vitesse moyenne des véhicules. Cette information permet par exemple la reconstitution d'une population de véhicules simulés, qui possède des caractéristiques semblables. De même, la plupart des grandes villes du monde possèdent des systèmes de mesure statiques : par exemple, en France,

6. http://wwww.openstreetmap.org
7. http://www.opendrive.org
8. http://diffusion-numerique.info-routiere.gouv.fr
9. http://www.datex2.eu

SIREDO [10] s'appuie sur des boucles magnétiques placées sous la route, qui fournissent des informations toutes les 15 minutes (débit, vitesse, etc. moyenne) ainsi que chaque heure (taux d'occupation). En outre, des mesures ponctuelles avec des tuyaux pneumatiques placés sur la route complètent bien souvent ces informations.

3.3. Création de l'infrastructure routière

Une route peut être représentée comme une séquence de nœuds qui connaissent leurs prédécesseurs et successeurs possibles. Les véhicules se déplacent alors le long d'une ligne droite, d'un nœud à l'autre, et récupèrent quand ils arrivent sur un nœud les différentes routes possibles. Dans *TrafficGen*, les véhicules ont une position logique sur la ligne axiale de la route : la visualisation de la route ou la position des véhicules sur sa voie ne sont en fait qu'une représentation graphique comme le montre la figure 1.



(a) représentation graphique (b) représentation interne

Figure 1. (a) Section de route avec un véhicule par voie. Représentation dans (b) TrafficGen : les carrefours sont des nœuds, chaque flèche est un vecteur de vitesse d'un agent véhicule, le nombre entre crochets indique le numéro de voie du véhicule

Ainsi, l'initialisation de l'environnement de simulation implique la lecture des nœuds et des routes à partir d'une base de données, de manière à les instancier et construire le graphe définissant la topologie du réseau routier simulé. Néanmoins, certains nœuds sont uniquement des points d'inflexion de la route (une seule sortie possible), tandis que d'autres sont des jonctions plus complexes qui fournissent plusieurs directions potentielles (i.e. carrefour) et impliquent aux véhicules de faire un choix. En outre, des données issues de systèmes d'information géographiques (SIG, i.e. OpenStreetMap ou OpenDrive) fournissent des indications sur les infrastructures routières qui doivent être réifiées dans la simulation (panneaux indicateurs par exemple, stops, etc.) pour améliorer le réalisme du comportement des véhicules. Plus généralement, la capacité d'intégrer d'autres éléments du réseau routier (i.e. piétons, cycles, voies

10. *Système Informatisé de REcueil de DOnnées* : http://www.transport-intelligent.net/produits-services/article/siredo

de bus, lignes de tramway, etc.) implique d'associer des éléments du SIG avec des entités situées dans l'environnement correspondant. Enfin, afin de tester des scénarios pertinents, la simulation doit inclure des outils de génération de véhicules (capable de reproduire les flux de données réelles) ainsi que des outils de mesure à placer en différents points du réseau.

Compte tenu de la grande diversité de ces entités et du besoin de les étendre en fonction des enjeux et des objectifs de modélisation, *nous préconisons une séparation claire entre les aspects déclaratifs et procéduraux du modèle, mais aussi une approche où toutes les entités sont traitées de façon homogène*. Par conséquent, nous préconisons l'usage de l'approche orientée interactions « IODA » (Kubera *et al*., 2010 ; 2011) qui stipule que chaque entité est un agent et que chaque comportement doit être défini comme une règle générique entre agents (appelé une *interaction*). Cela conduit à la conception séparée des agents et de bibliothèques d'interactions réutilisables, qui sont traitées par un moteur de simulation générique. Concrètement, les agents utilisés dans *TrafficGen* sont décrits ci-dessous (nous présentons la modélisation de leur comportement dans la section suivante). Un dictionnaire assure la correspondance entre les éléments SIG et les agents.

Les **nœuds** ont d'autres nœuds comme accointances; ils sont reliés par des **liens** représentant les routes, dotés d'attributs spécifiques (e.g. le nombre de voies, voies mono-directionnelles ou bi-directionnelles, limites de vitesse, etc.).

Les **véhicules** ont une vitesse désirée (qu'ils cherchent à atteindre) et une vitesse instantanée qu'ils adaptent en fonction de la situation. Ils calculent également une vitesse moyenne (sur les derniers cycles de simulation). Comme illustré sur la figure 1 tous les véhicules se déplacent le long de l'axe de la route (à savoir un lien entre deux nœuds) et sont affectés logiquement à une voie. La perception du véhicule est personnalisable; par défaut, chaque véhicule a un cône de vision à l'avant et un autre à l'arrière. A chaque nœud, il choisit un nouveau nœud de destination et mémorise d'où il vient. Il est également possible de spécialiser les véhicules en « sous-espèce » avec d'autres capacités de perception ou d'action (i.e. les voitures, les cycles...).

Les **carrefours** sont créés quand un nœud possède plus de deux accointances. Ils gèrent l'accès à des voies pour éviter ou détecter les collisions. Différentes possibilités ont été proposées pour ce faire (Mandiau *et al*., 2008 ; Tlig *et al*., 2012); comme comportement par défaut, nous utilisons un sémaphore pour contrôler l'accès à chaque voie. Actuellement, dans l'approche IODA l'agent Carrefour ne possède pas de mécanismes de régulation : à la place, ceux-ci sont exprimés par des règles (*interactions*), de sorte que différents mécanismes de la littérature puissent être mis en œuvre facilement puis assignés aux carrefours.

Les feux de circulation et autres panneaux routiers sont créés à partir des informations SIG (lorsqu'elles sont disponibles) et réifiés par des agents appartenant à une famille spécifique.

Les **générateurs** sont en charge de la création de véhicules avec des caractéristiques (vitesse, flux, destination, propriétés du véhicule...) qui reflètent soit des données réelles soit des lois de probabilité.

Les **sondes** mesurent et sauvegardent les caractéristiques des véhicules qui passent à proximité. Elles peuvent être personnalisées de manière à surveiller les fonctionnalités considérées comme pertinentes dans chaque scénario de simulation.

Des **gestionnaires d'événements** de plusieurs types permettent d'étendre les actions qui peuvent être effectuées sur les véhicules ou autres agents, selon des scénarios spécifiques; en particulier des réducteurs de vitesse peuvent imposer des limites de vitesse temporaires à tous les véhicules dans un périmètre circulaire (travaux, accidents, marchés...).

Ces trois dernières familles d'agents ne font évidemment pas partie des données SIG. Générateurs et sondes peuvent être placés sur des points de mesure réels (pour générer des flux réalistes de véhicules ou comparer les flux simulés avec ceux issus des données réelles) ou sur des nœuds arbitrairement choisis pour l'expérimentation. Les gestionnaires d'événements peuvent ainsi être placés partout sans contrainte d'existence de nœuds : ils peuvent être utilisés pour évaluer l'impact de mesures d'urgence, telles qu'une politique locale de réduction de vitesse en réponse à des pics de pollution, par exemple.

4. Modèle de comportement

La simulation de trafic routier implique la mise en œuvre de comportements complexes. Une partie de cette complexité, notamment tout ce qui touche à la manière dont les conducteurs obtiennent des informations issues de l'environnement, peut élégamment être déléguée à des agents qui réifient les dispositifs routiers (carrefours, panneaux de signalisation, feux, etc.), comme suggéré par Gibson dans sa *théorie des affordances* (Gibson, 1979). En effet, notre objectif n'est pas de reproduire un modèle psychologiquement réaliste du conducteur, mais de veiller à ce que les véhicules se déplacent d'une manière compatible avec les connaissances et observations, tout en maintenant une certaine diversité comportementale. Ainsi, nous utilisons *l'approche orientée interactions* IODA (Kubera *et al*., 2011) afin de représenter différents scénarios et déléguer une partie de la connaissance des véhicules à d'autres types d'agents.

4.1. Une approche orientée interactions pour la modélisation de scénarios

Ayant pour objectif de concevoir des outils pour l'aide à la décision, il est important de permettre à l'utilisateur de construire des scénarios, de les réviser facilement, de pouvoir les comparer et d'évaluer leurs résultats. Par exemple : *Il est 10 heures à Paris, 80 % des conducteurs sont des personnes âgées, 20 % sont des cadres dynamiques. Ils arrivent sur le périphérique à un taux de 50 véhicules/min. Une alerte de pollution est déclenchée sur le centre de Paris. Quelle sera la conséquence sur la densité de*

la circulation dans les avenues principales d'une vitesse limitée à 50 km/h sur le périphérique ?

Pour tenir compte de tous ces aspects (le sous-réseau étudié, les spécifications individuelles des véhicules, leur flux d'entrée), l'architecture du simulateur doit permettre des associations modulaires de différentes familles d'agents afin de différencier les comportements génériques.

Notre proposition est d'utiliser la représentation matricielle de l'approche IODA (Kubera *et al.*, 2011), dans laquelle nous indiquons quelles interactions peuvent être effectuées entre chaque paire de familles d'agents (tableau 1). Par exemple, quand un véhicule A (*source*) perçoit un véhicule B (*cible*) devant lui, il peut *dépasser*, *décélérer* ou exécuter un *freinage d'urgence* en fonction de leur vitesse et de la distance qui les sépare.

Tableau 1. Extrait simplifié de la matrice d'interactions décrivant le modèle comportemental dans TrafficGen. Cette matrice indique quelles interactions peuvent être réalisées par certains agents (sources) sur d'autres agents (targets) selon leurs familles : e.g. les véhicules (cars) peuvent doubler (Overtake), ralentir (SlowDown) ou freiner fortement (EmergencyBrake) en présence d'autres véhicules (cars)

Sources \ Targets	∅	Cars	Nodes	Crossroads	…
Cars	Accelerate Forward Fallback	**Overtake** **SlowDown** **EmergencyBrake**	Cross	Enter Exit	…
Nodes					…
Crossroads		SelectDirection GivePriority			…
Lights		Stop			…
Generators	CreateVehicle				…
Probes		Count			…
⋮	⋮	⋮	⋮	⋮	

Si nous décrivons un algorithme « naïf » (tableau 2) pour la prise de décision entre ces trois comportements, nous observons l'imbrication de plusieurs concepts distincts : en particulier, la définition des actions à effectuer, et un processus de décision complexe. Nous pouvons également observer que la notion de priorité entre les comportements n'est pas explicitement spécifiée, d'où un manque de modularité dans les comportements. En fait, une simple tentative de modification de la priorité d'une interaction nécessite de changer l'ordre des « if » en cascade de manière à prendre en compte les conditions d'exécution et la distance entre les agents.

Au contraire, avec l'approche IODA, notre exemple nous conduit à définir les associations appropriées dans la matrice d'interactions (tableau 3) et trois interactions indépendantes (tableau 4). Cette représentation permet une séparation claire entre la définition des actions et leurs conditions à travers les *interactions*. Il permet aussi une planification explicite des comportements (par le biais des priorités) et une « garde de distance » indiquée explicitement dans la matrice d'interactions.

Tableau 2. Algorithme « naïf » pour la sélection d'actions d'un véhicule, s'appuyant sur les caractéristiques perçues chez un véhicule voisin (target)

```
1: if target:distance ≤ 10 ∧ not-in-crossroad? ∧ target:in-front? ∧ target:same-direction? ∧
   target:same-lane? ∧ not-road-end? then
       // Overtake(30; 10)
2:     if target:too-close? ∧ target:too-slow? ∧ has-left-lane? ∧ left-lane-free? then
3:         save-target-overtaking
4:         go-to-left
5:         forward
       // EmergencyBrake(20; 5)
6:     else if target:distance ≤ 5 ∧ target:emergency-distance? then
7:         emergency-braking
       // DecelerateAvoidCar(10; 10)
8:     else if target:too-close? ∧ not-stopped? then
9:         slowdown
10:        forward
11:    end if
12: end if
```

Ces interactions précisent les comportements en tant que règles de type conditions/actions (comme en STRIPS) en utilisant des primitives abstraites (tableau 4). Ces primitives peuvent conduire à une implémentation différente pour chaque famille d'agent en fonction de sa structure et de ses capacités. Grâce à cette approche, une modification du modèle consiste uniquement à modifier les interactions affectées aux familles d'agents dans la matrice, ainsi que leurs priorités et leurs gardes de distance.

*Tableau 3. Représentation des interactions possibles entre véhicules dans le tableau 1 à l'aide de IODA-NetLogo. Les champs représentent respectivement les données suivantes: le type de l'agent **source** (e.g. `cars`); le nom de l'**interaction** (e.g. `Overtake`); la **priorité** de cette interaction (e.g. `30`); le type de l'agent **cible** (e.g. `cars`); la **garde de distance** (e.g. `10`)*

`cars`	`Overtake`	`30`	`cars`	`10`
`cars`	`EmergencyBrake`	`20`	`cars`	`5`
`cars`	`DecelerateAvoidCar`	`10`	`cars`	`10`

Cette approche facilite grandement la modification des modèles. La séparation entre les connaissances relatives aux entités et leur comportement, nous permet de changer le modèle, même durant la simulation. En effet, dans l'approche IODA, la matrice d'interactions est réifiée et relue à chaque prise de décision des agents, de sorte qu'une simple modification de la matrice d'interactions permet de modifier l'ensemble des comportements disponibles au pas de temps suivant. Il est d'ailleurs possible, si cela est nécessaire, d'écrire des interactions qui ont pour effet de modifier la matrice.

*Tableau 4. Implémentation des interactions définies dans le tableau 3. Les **déclencheurs** indiquent les motivations de l'agent à réaliser l'interaction, les **conditions** décrivent les pré-requis des actions et les **actions** sont exécutées en séquence*

```
INTERACTION Overtake
    TRIGGER target:in-front? target:same-direction? target:same-lane?
            target:too-close? target:too-slow?
    CONDITION not-in-crossroad? not-road-end? has-left-lane? left-lane-free?
    ACTIONS save-target-overtaking go-to-left forward
END

INTERACTION EmergencyBrake
    TRIGGER target:in-front? target:same-direction? target:same-lane?
            target:emergency-distance?
    CONDITION not-in-crossroad? not-road-end?
    ACTIONS emergency-braking
END

INTERACTION DecelerateAvoidCar
    TRIGGER in-front? target:same-direction? target:same-lane?\
                                             target:too-close?
    CONDITION not-in-crossroad? not-stopped? not-road-end?
    ACTIONS decelerate forward
END
```

En outre, les interactions (qui sont des règles génériques) peuvent être réutilisées dans différents contextes. Par exemple, les interactions définies pour les dépassements de véhicules sont transposables à des deux-roues voire à des piétons, dans la mesure où le comportement d'ensemble est le même, et que les modalités particulières dans la réalisation de ces interactions sont implémentées au sein des primitives spécifiques à chaque famille d'agents.

4.2. Délégation de politiques à d'autres agents

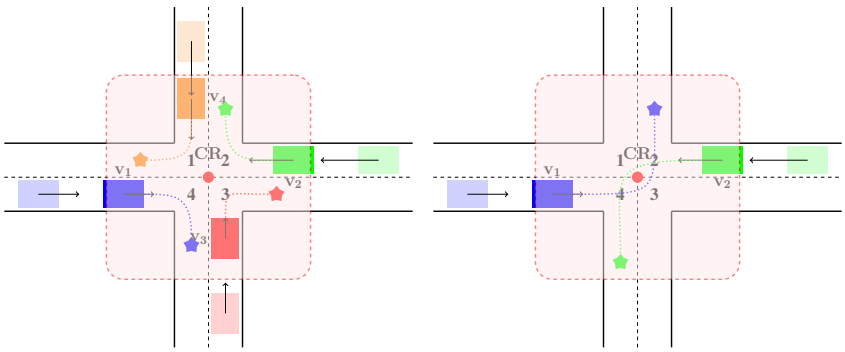
En outre, l'approche centrée interactions nous permet également de réduire la complexité de la connaissance des conducteurs, et d'utiliser des politiques alternatives pour représenter différentes hypothèses de comportement. Nous abordons ci-dessous deux problèmes classiques : la coordination dans les carrefours et les véhicules obstacles.

Les comportements de coordination des véhicules arrivant à un carrefour peuvent être mises en œuvre à l'aide de nombreux modèles (Dresner, Stone, 2007 ; Mandiau *et al.*, 2008 ; Tlig *et al.*, 2012). Lorsque les véhicules intègrent l'un de ces modèles dans leurs règles de décision, non seulement leur complexité augmente, mais ce choix est difficile à modifier, e.g. afin d'évaluer des modèles alternatifs.

Dans l'approche centrée interactions, ces différents modèles de coordination peuvent être mis en œuvre comme des interactions entre un agent *carrefour* et les véhicules entrants. Puis, afin de tester un modèle ou un autre, le concepteur n'a plus alors qu'à

choisir dans la bibliothèque d'interactions disponibles laquelle mettre dans la matrice d'interactions.

Nous montrons dans la figure 2 deux situations qui peuvent être traitées avec un agent Carrefour. Les agents Carrefour sont construits durant la phase d'initialisation de l'environnement, lorsque qu'un nœud est relié à au moins trois routes. Ils sont décomposés en n quadrants (1 par route). Ils sont dotés d'un halo de perception et peuvent interagir avec des véhicules entrants, demander leur direction et détecter d'éventuels conflits en fonction des quadrants impactés par la direction prise par chaque véhicule. Dans la première situation (a), quatre véhicules arrivent à peu près au même moment, avec l'intention de tourner à droite : de cette manière le carrefour anticipe qu'aucun conflit n'est en mesure de se produire puisque tous les véhicules utilisent des quandrants distincts, et permet à tous les véhicules d'entrer : i.e. les agents véhicules sont en mesure d'effectuer l'interaction `Enter` sur le carrefour. Dans le second cas (b), deux véhicules arrivant sur des voies opposées ont l'intention de tourner à gauche. Il y a donc conflit puisque le véhicule v_1 doit traverser les quadrants 4, 3 et 2 tandis que v_2 doit passer par 2, 1 et 4 : ainsi, l'agent Carrefour doit décider à quel véhicule accorder l'autorisation d'entrer : i.e. le carrefour effectue l'interaction `GivePriority` sur l'un des véhicules.



(a) Situation 1 (b) Situation 2

Figure 2. Deux situations où plusieurs véhicules arrivent à un carrefour (CR), qui peut interagir avec eux au sein de son halo de perception. (a) : chaque véhicule souhaite tourner à droite, donc les carrefours peuvent laisser les véhicules entrer sans risque; (b) : deux véhicules ont l'intention de tourner à gauche, le carrefour doit alors appliquer une politique définie par l'utilisateur pour gérer leur coordination

Selon la mise en œuvre de `GivePriority`, plusieurs stratégies peuvent être mises en œuvre et comparées. Par exemple :

1. Code de la route : les véhicules doivent céder le passage aux véhicules venant de la droite. Bien sûr, cette méthode peut conduire à une impasse lorsque toutes les voies sont occupées. Il est donc souvent utile de la coupler avec une autre politique.

2. FIFO : la priorité d'un véhicule dépend de son instant d'arrivée (le premier arrivé entre en premier).

3. Round-Robin : Le carrefour donne la priorité à chaque voie dans un ordre circulaire, de manière à équilibrer le temps d'attente moyen sur chaque voie.

En outre, les carrefours peuvent bien sûr être réglementés par des feux de circulation. Dans ce cas, ces derniers sont également introduits dans l'environnement comme des agents, ce qui réduit là encore considérablement la complexité de la tâche de coordination pour ces carrefours. Les agents Feux de circulation sont alors en charge de l'arrêt des véhicules (qui peuvent être paramétrés pour réagir plus ou moins efficacement) dans un rayon donné.

Un autre problème classique consiste à prendre en compte les véhicules ayant une vitesse très faible, ou même un véritable obstacle, sur une route à deux voies. Dans ce cas (figure 3), le véhicule v_0 est juste en face du véhicule v_1 avec $speed(v_0) << speed(v_1)$. Bien sûr, comme expliqué ci-dessus, le véhicule v_1 est capable de percevoir v_0, de réduire sa propre vitesse ou d'effectuer un freinage d'urgence, en particulier si d'autres véhicules arrivent en même temps sur les voies opposées (e.g. v_2). Cette situation est susceptible d'ammorcer un embouteillage. Mais, dans les cas réels, les véhicules ne resteront pas dans leur voie et essayeront de dépasser le véhicule v_0. En raison de la très faible vitesse de cet obstacle et des véhicules derrière, il est probable que l'interaction existante `dépassement` ne soit pas vraiment appropriée pour traiter ce problème. Ainsi, il est utile de prévoir un mécanisme qui permet à tous les véhicules à proximité de l'obstacle de détecter le problème et d'appliquer une politique adaptée.

Pour ce faire, quand un véhicule perçoit un autre véhicule devant lui, avec une vitesse beaucoup plus petite que sa propre vitesse, il crée un agent *obstacle* lié au véhicule le plus lent. Cet agent possède un halo de perception spécifique et est en charge de l'interaction avec les véhicules à proximité afin de les coordonner. Encore une fois, plusieurs stratégies peuvent être mises en œuvre :

1. Code de la route : par défaut, le véhicule derrière un obstacle doit attendre jusqu'à ce que la voie opposée soit libre.

2. FIFO : le véhicule qui est plus proche de l'obstacle augmente sa priorité

3. Round-Robin : chaque voie envoie un véhicule à tour de rôle.

4. Communication/négociation : les véhicules coincés peuvent utiliser une coordination tacite (i.e. à travers les feux de croisement) pour céder le passage.

Comme pour l'agent *carrefour* dont l'intérêt est de simplifier la régulation de l'ensemble des véhicules selon une politique choisie librement (et ainsi éviter des mécanismes de négociation complexes entre les véhicules), l'agent *obstacle* sert de médiateur pour la coordination entre les véhicules des deux files. Il avertit non seulement le prochain véhicule de sa propre voie mais également les véhicules en face, qui de ce fait adaptent leur vitesse.

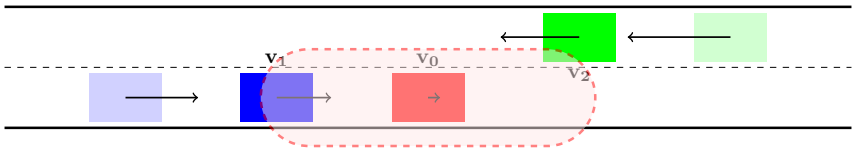


Figure 3. Partie de route avec un véhicule roulant à une vitesse très lente (par rapport à d'autres véhicules), ici v_0. Dans TrafficGen il est détecté, couplé avec un agent obstacle*, dynamiquement doté d'un comportement spécifique de manière à gérer les véhicules entrants dans son halo de perception (v_1 et v_2) selon une politique définie par l'utilisateur*

Utiliser le principe d'affordance n'est évidemment pas obligatoire, mais, comme nous pouvons le voir, il aide fortement à la conception des comportements modulaires, permet de tester plusieurs alternatives dans les politiques utilisées, et réduit la complexité des agents véhicules. Lorsque des artefacts apparaissent dans la simulation, on leur demande simplement de gérer la complexité supplémentaire qu'ils induisent en affectant le moins possible les agents existants.

5. La plateforme *TrafficGen*

Comme une preuve de concept, nous avons développé un outil expérimental nommé *TrafficGen* (Bonhomme *et al.*, 2015), basé sur l'extension IODA [11] pour la plateforme multi-agents NetLogo. NetLogo (Wilensky, 1999) fournit aussi une extension native GIS et une extension SQL externe [12]. Nous avons tout d'abord testé la capacité de cette plateforme à importer des cartes géographiques de différentes zones (e.g. figures 4, 5 et 6), puis implémenté quelques comportements « classiques » comme l'évitement de collisions ou des modèles de déplacements (figure 6).

5.1. OpenStreetMap

Dans cette partie, nous nous concentrons sur le format OSM, en expliquant comment nous construisons un réseau routier dans la simulation à partir de données de ce type. Notre approche peut donc être facilement transposable à d'autres réseaux de transport, comme indiqué dans la section 5.2.

Les données OSM sont structurées sous forme d'un fichier XML composé de trois niveaux d'éléments. Les **nœuds** (`<node>`) sont des points d'intérêt sur la carte, identifiés par un ID et des coordonnées GPS. Comme tous les éléments OSM ils peuvent encapsuler des balises Clé/Valeur **tags** (`<tag k = "..." v = "..." />`) fournissant des informations complémentaires. Les **chemins** (`<way>`) sont des séquences de

11. http://cristal.univ-lille.fr/SMAC/projects/ioda/ioda_for_netlogo/
12. https://code.google.com/p/netlogo-sql/

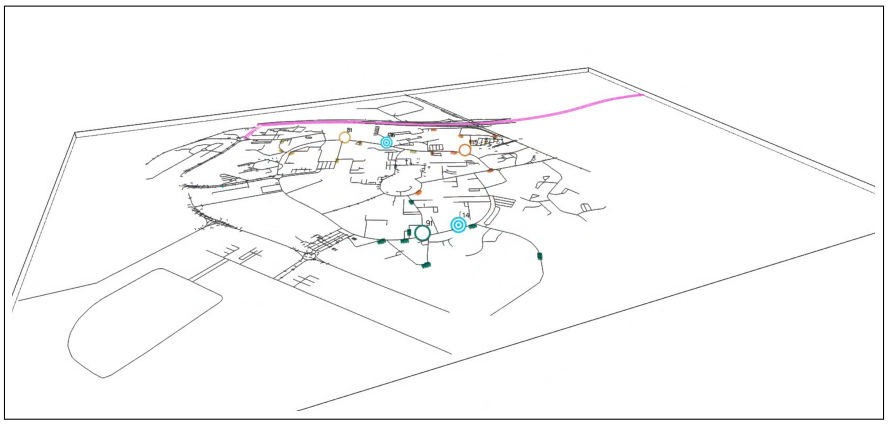


Figure 4. Le campus de l'université Lille 1 dans TrafficGen

nœuds (`<nd ref = "..." />`) qui peuvent aussi bien représenter une courbe ouverte (e.g. une rue) qu'un polygone. Les chemins possèdent également des balises Clé/Valeur (e.g. route à sens unique, nombre de voies, rond-point, etc.). Les **relations** (`<relation>`) sont des entités logiques impliquant des nœuds et des chemins (*<membre type = "nodeway" ref role = "..." = "..." />*) avec notamment des rôles (e.g. les lignes de métro sont décrites en connectant des chemins et des nœuds spéciaux que sont les stations). Chaque relation peut également être dotée de balises spécifiques.

Comme nous pouvons le voir, cette information est de bas niveau et peu structurée. Le système de balises permet d'associer de multiples informations à un élément, comme : la limitation de vitesse, les voies circulaires, l'éclairage public, etc. Dans le cas des réseaux routiers, la balise la plus importante est annotée par la clé `highway`; la valeur associée indique la nature de l'élément correspondant (à savoir le type de route : `primary, secondary, résidential` ...).

Pour construire un réseau routier à partir d'un fichier OSM, les données doivent tout d'abord être filtrées pour ne conserver que l'information pertinente sur la structure du réseau souhaité. Beaucoup d'outils peuvent être utilisés (e.g. *Osmosis* [13], *JOSM* [14], *Merkaartor* [15]) pour conserver ou exclure les nœuds et chemins en fonction de leurs balises (i.e. dans notre cas nous ne conservons que les balises routières). Ensuite, nous recommandons d'injecter les données restantes dans une base de données relationnelle, pour faciliter l'accès structuré aux propriétés des éléments OSM (à savoir les tags). PostgreSQL et l'extension PostGIS sont particulièrement bien adaptés pour gé-

13. http://wiki.openstreetmap.org/wiki/Osmosis
14. http://wiki.openstreetmap.org/wiki/Josm
15. http://wiki.openstreetmap.org/wiki/Merkaartor

rer cela. Un outil comme *osm2pgsql* [16] effectue cette opération en une seule ligne de commande.

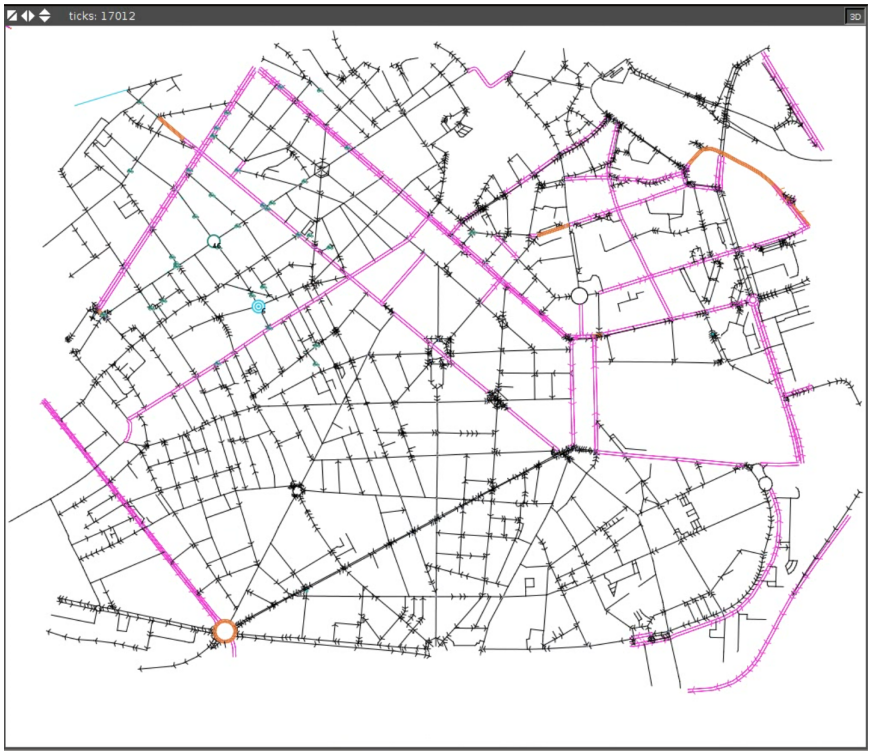


Figure 5. Une partie du centre ville de Lille (France) chargé dans TrafficGen

Pourtant, une limitation majeure de cette approche vient du nombre de nœuds impliqués dans la description des infrastructures routières issues d'OpenStreetMap. Selon l'échelle de la simulation, le niveau de détail peut souvent être trop élevé et diminue considérablement la vitesse du simulateur. Ainsi, afin de traiter de grandes cartes (à savoir un grand nombre de nœuds), il est nécessaire d'appliquer des algorithmes de simplification. Le domaine de la généralisation cartographique fournit un grand nombre de méthodes pour le faire. Dans *TrafficGen* nous avons implémenté l'algorithme Visvalingam-Whyatt (Visvalingam, Wyatt, 1993) à la fois simple et efficace. Basé sur une mesure d'aires de zones triangulaires, cet algorithme nous permet de diviser par 4 à 5 le nombre de nœuds (en fonction du niveau de zoom), sans affecter visuellement la forme de routes. Par exemple, en prenant la carte présentée à la figure 5, le fichier OSM original contient 82 852 nœuds; après avoir filtré les nœuds pertinents pour la circulation routière, la carte importée ne contient plus que 2 839

16. http://wiki.openstreetmap.org/wiki/Osm2pgsql

nœuds; enfin, après l'application de l'algorithme Visvalingam-Wyatt, le nombre de nœuds tombe à 700-800 en fonction du niveau de détail souhaité.

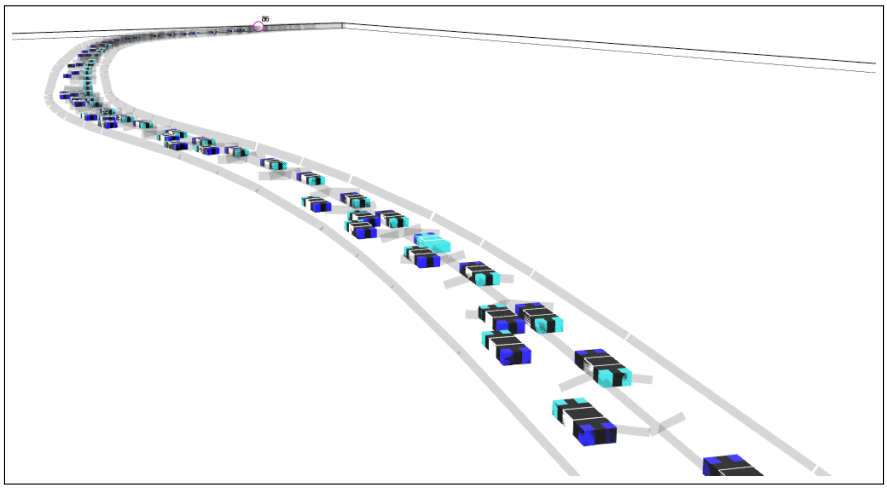


Figure 6. Simulation de trafic sur l'autoroute A23 dans TrafficGen

5.2. OpenDRIVE et les autres réseaux de transport

Afin de démontrer la puissance de notre solution face à différents formats de données, nous l'avons appliquée également au format OpenDrive. Le format OpenDrive est composé de trois parties distinctes : les routes, les contrôleurs (i.e. feux et limitations de vitesse dynamiques) ainsi que les jonctions (e.g. carrefour, rond-point, etc.), qui sont représentés par les marqueurs `road`, `controller` et `junctions`. Malheureusement, il n'existe pas de moyen simple pour analyser le format OpenDrive. Nous avons donc développé un plug-in dédié (non décrit ici) pour charger ce type de données dans notre simulateur.

A l'heure actuelle, nous sommes concentrés sur l'implémentation des routes. Les routes sont composées de trois couches, qui permettent différents niveaux de détail. Nous n'abordons ici que la première couche (l'ensemble des spécifications peut être trouvé sur le site OpenDrive [17]), qui est dédiée à la géométrie des routes. Cette couche est composée d'un ensemble de routes (i.e. marqueur `small road`), chacun possédant une liste des sections avec une géométrie spécifique (i.e. marqueur `geometry`). Dans OpenDrive 1.3 quatre types de géométries peuvent être utilisés : les lignes droites (`line`), les courbes (`arc`), la spirale (`spiral`) et les polynômes du troisième degré (`poly3`). Leurs différences se situent principalement dans les coefficients de courbure qui sont respectivement zéro, constante non nulle et linéaire, comme indiqué dans la figure 7.

17. http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.3D.pdf

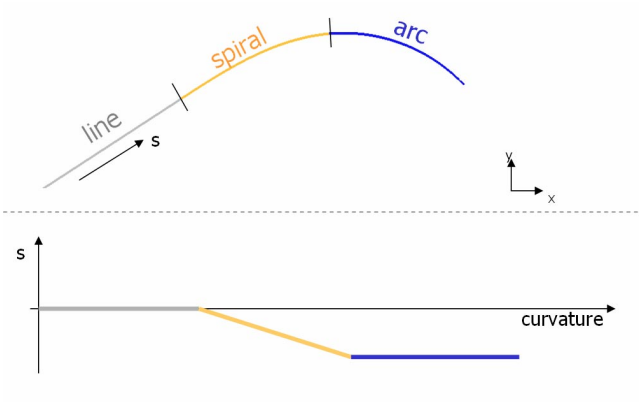


Figure 7. Extrait de la spécification OpenDrive illustrant les différentes géométries. La courbe du haut montre le rendu géométrique de chaque géométrie en fonction du coefficient de courbure (graphique du bas)

Dans ce format, chaque géométrie possède uniquement une coordonnée (généralement un point de départ), une longueur et les informations sur la courbure. Ainsi, les nœuds agents doivent être extrapolés en fonction de la géométrie souhaitée. Les nœuds sont ensuite connectés ensemble pour construire des routes (figure 8). Il est important de noter que ce format a un avantage majeur sur une solution comme OSM dans le cadre de simulations multi-échelles (Picault, Mathieu, 2011), puisque le niveau d'interpolation peut être réglé à l'échelle souhaitée.

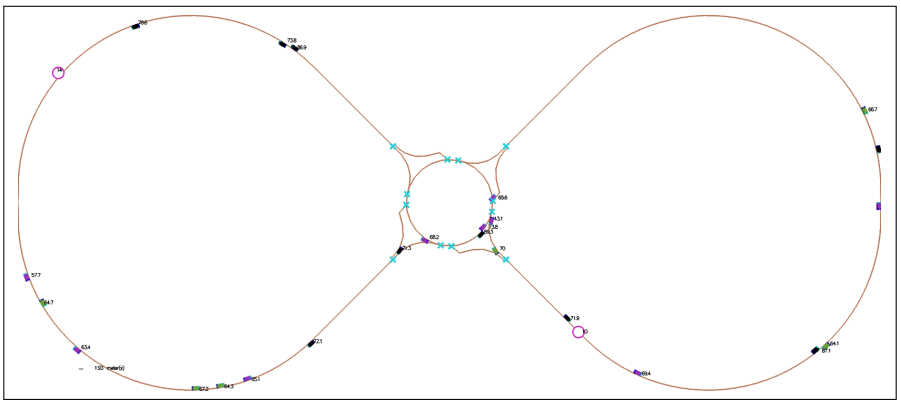


Figure 8. Un fichier OpenDrive chargé dans TrafficGen

5.3. Fonctionnalités actuelles

Dans cette version de TrafficGen, toutes les infrastructures routières fournies par OpenStreetMap (i.e. avec la clé `highway`), peuvent être manipulées par le simulateur. Les sens uniques comme les routes bi-directionnelles, sont pris en compte, avec un nombre quelconque de voies. Les cartes dans le format OpenDrive sont construites en utilisant uniquement les éléments d'information 2D. Les limites de vitesse, les stops et les panneaux indicateurs sont également pris en compte.

Actuellement, tous les véhicules ont la même taille. Tous les véhicules peuvent effectuer les mêmes interactions. Néanmoins, la réalisation de ces interactions implique des paramètres qui peuvent être réglés pour reproduire plusieurs types de conducteurs, basés sur l'analyse de données réelles (Lacroix *et al.*, 2013). En outre, les véhicules peuvent être dotés de points de destination (là encore à partir de données réelles enregistrées), qu'ils vont alors essayer de joindre à l'aide de l'algorithme de plus court chemin D*-Lite (Koenig, Likhachev, 2005).

Tous les véhicules simulés sont en mesure d'accélérer ou de décélérer en fonction de leur vitesse désirée, de la présence d'autres véhicules, en fonction des limites de vitesse et de la proximité de carrefours. Ils sont bien sûr capables de dépasser les véhicules plus lents.

L'utilisateur de TrafficGen dispose dans l'interface de boutons permettant d'interagir avec les agents dans leur environnement, qu'il s'agisse de véhicules ou d'agents d'infrastructure. Ces outils permettent notamment, durant l'exécution de la simulation et avec une prise en compte immédiate dans les calculs de plus courts chemins, d'ouvrir ou de fermer des routes dynamiquement, de changer les caractéristiques d'une route, de lancer des événements qui réduisent la vitesse dans une zone donnée, d'ajouter des sondes pour mesurer des paramètres du véhicule, et d'ajouter plusieurs types de générateurs à tout nœud de la carte (soit à partir sur les lois statistiques soit à partir de données réelles enregistrées).

Nous avons également testé notre méthode sur d'autres réseaux de transport. Par exemple, nous avons utilisé OpenStreetMap pour obtenir des données de différents réseaux de métro français (e.g. le métro de Lyon, figure 9). Afin de compléter la création du réseau, les agents nécessaires ont été associés à des éléments OSM (i.e. trains, gares, etc.), les interactions supplémentaires ont été définies (parfois en utilisant des primitives d'interaction existantes), et la matrice d'interactions correspondante a été ré-écrite.

Bien évidemment, en fonction du niveau de finesse dans les comportements ainsi que des objectifs de la simulation, de nombreuses autres fonctionnalités pourraient être introduites sous forme de nouveaux agents ou de nouvelles interactions. Par exemple, on peut décider si nécessaire de prendre en compte la largeur des voies ou l'espace disponible dans les carrefours et ainsi tester leur adéquation avec les rayons de braquage des véhicules. Ou encore, on peut mettre en place des mécanismes de calcul et de prise en compte de voies virtuelles pour les deux-roues (Bonte *et al.*, 2006). En-

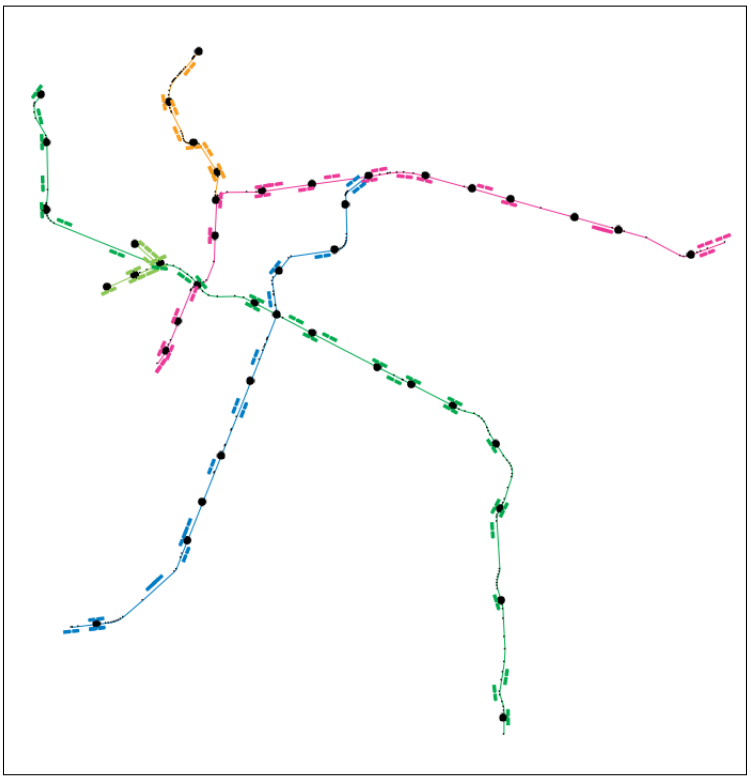


Figure 9. Une simulation du métro de Lyon (France) dans TrafficGen

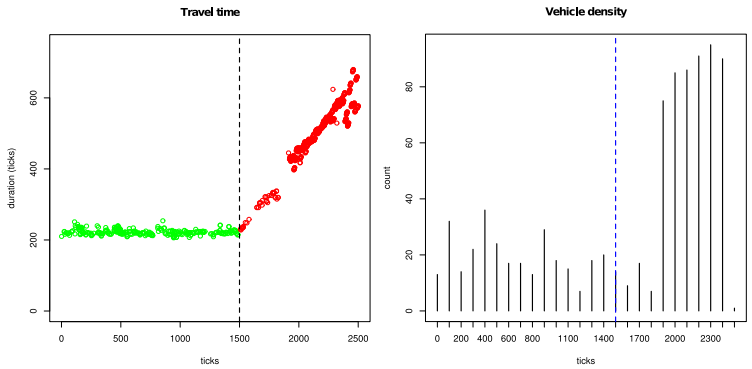
fin, on peut introduire des agents d'observation destinés à détecter l'engorgement des routes et rétroagir sur les calculs de plus courts chemins.

5.4. Exemple expérimental

Pour illustrer notre approche, nous montrons une expérience simple, basée sur la carte du centre-ville de Lille (figure 5), dans laquelle nous avons placé un point de mesure permettant d'avoir le temps de voyage (par rapport à un point en amont) de chaque véhicule ainsi que la densité (nombre de véhicules par période de 100 cycles de simulation) de véhicules observés (figure 10). Ceux-ci sont créés par un agent Générateur utilisant un intervalle aléatoire suivant une loi de Poisson (avec un paramètre $\lambda = 5$) et une vitesse dictée par une loi normale (avec une vitesse moyenne de 90 km/h et un écart type de 10 km/h).

Au cycle de simulation 1500, une limitation de vitesse à 50 km/h est imposée par un agent réducteur de vitesse dans un périmètre de 800 m autour du point de mesure. Comme nous pouvons le voir dans la figure 10a, la limitation augmente progressive-

ment le temps de voyage des véhicules, mais néanmoins la vitesse ne se stabilise pas, ce qui indique la formation d'un embouteillage. Le graphique des densités (figure 10b) confirme cette observation : après une faible diminution (en raison du ralentissement des voitures avant le point de mesure), la densité augmente considérablement.



(a) (b)

Figure 10. Evolution du temps de trajet (a) et de la densité (b) (nombre de véhicules observés tous les 100 cycles). Au cycle 1 500 de la simulation, une limitation de vitesse est placée dans un rayon de 800 m autour du point de mesure, ce qui a pour conséquence la création d'un embouteillage

Pour réaliser ce type d'expérience, il suffit de placer un générateur, une sonde et un gestionnaire d'évènements dans l'environnement étudié. Les comportements des conducteurs implémentés dans *TrafficGen* ont été abondamment cités dans la littérature et la plupart ont déjà été validés expérimentalement. Il nous reste maintenant à valider l'intégration de flux de données (travail en cours) et reproduire une population d'agents à partir de ces informations globales. Nous envisageons aussi de fusionner des informations nationales (e.g. *Bison Futé*) ainsi que des informations issues de communautés urbaines, en utilisant les techniques présentées par (Mandiau *et al.*, 2008) ou (Lacroix *et al.*, 2013).

6. Conclusion et perspectives

Les simulateurs sont maintenant couramment reconnus comme des outils essentiels pour l'aide à la décision. Le transport, d'une manière générale, et le trafic routier en particulier sont de plus en plus étudiés, aussi bien à un niveau macroscopique (équations de flux, variables aggregées, etc.) qu'au niveau microscopique (approche centrée individus). Durant ces dernières années, la quantité de données réelles s'est considérablement accrue. Dans cet article, nous avons présenté une méthode d'intégration de ces données modulaire et hautement paramétrable, à l'aide d'un modèle de comportements pour simulation multi-agent intelligible et facilement modifiable. Par ailleurs, notre approche est facilement extensible à d'autres réseaux de transport (e.g. trams,

bus, métro, vélo, piétons, etc.) avec très peu d'effort de codage, et à d'autres formats de données GIS comme OpenDRIVE.

Bien que la qualité des données ouvertes puisse être discutable (Girres, Touya, 2010), leur usage facilite grandement la réalisation de simulations de trafic en fournissant des données gratuites, corrigées et mises à jour constamment. L'approche IODA aide le concepteur à élaborer son modèle de comportements dans un processus incrémental en séparant clairement la structure et les capacités des agents à l'aide de règles comportementales abstraites qui rendent les modèles facilement modifiables ou extensibles. L'ensemble des éléments de l'infrastructure routière nécessaire au bon fonctionnement du système sont modélisés et implémentés par des agents, et chaque comportement d'agent est basé sur des règles génériques réutilisables (les interactions).

Ces propriétés permettent une grande flexibilité pour la conception et le test de scénarios qui, de ce fait, peuvent couvrir un large spectre d'études, allant du choix du réseau routier jusqu'à l'étude des détails du comportement des agents.

Nous travaillons actuellement sur l'acquisition de flux d'informations dans les différents réseaux qui sont, en général, bien moins strandardisés que celui du réseau routier et sont très souvent fournis comme de simples plannings. Néanmoins, la généralisation de notre approche devrait faciliter la conception « à la demande » de simulations, afin de répondre à des scénarios variés.

Bibliographie

Al-Zinati M., Araujo F., Kuiper D., Valente J., Wenkstern R. (2013). DIVAs 4.0: A multi-agent based simulation framework. In *IEEE/ACM 17th Int. Symp. on Distributed Simulation and Real Time Applications*, p. 105-114.

Bazzan A., Klügl F. (2014, 1). A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, vol. FirstView, p. 1–29.

Bonhomme A., Mathieu P., Picault S. (2015). TrafficGen: a flexible tool for informing agent-based traffic simulations with open data. In Y. Demazeau, K. S. Decker, J. Bajo Pérez, F. de la Prieta (Eds.), *Proc. of the 13th Int. Conf. on Practical Applications of Agents and Multi-Agent Systems (PAAMS'2015)*, vol. 9086, p. 259–262. Springer. (Demonstration paper)

Bonhomme A., Mathieu P., Picault S. (2016). A versatile multi-agent traffic simulator framework based on real data. *International Journal on Artificial Intelligence Tools*, vol. 25, n° 1, p. 20.

Bonte L., Espié S., Mathieu P. (2007). Virtual lanes interest for motorcycles simulation. In *5th European Workshop on Multi-Agent Systems (EUMAS'07)*, p. 580–596. ATIA.

Bonte L., Mathieu P., Espié S. (2006). Modélisation et simulation des usagers des deux-roues motorisés dans archisim. In V. Chevrier (Ed.), *14e journées francophones sur les systèmes multi-agents (jfsma)*, p. 31–44. Hermès.

Bourbakis N. (1997). A traffic priority language for collision-free navigation of autonomous mobile robots in dynamic environments. *Trans. Sys. Man Cyber. Part B*, vol. 27, n° 4, p. 573–587.

Bourbakis N., Findler M. (2001). Smart cars as autonomous intelligent agents. In *IEEE 13th Int. Conf. on Tools with Artificial Intelligence (ICTAI'2001)*, p. 25–32. IEEE.

Bourgois L., Saunier J., Auberlet J.-M. (2012). Towards contextual goal-oriented perception for pedestrian simulation. In F. J., A. Fred (Eds.), *Proceedings of the 4th International Conference on Agents and Artificial Intelligence (ICAART)*, p. 197–202. SciTePress.

Champion A., Mandiau R., Kolski C., Heidet A., Kemeny A. (1999). Traffic generation with the SCANeR™ simulator: towards a multi-agent architecture. In *Driving Simulation Conference*, p. 311–324.

Chen B., Cheng H. (2010). A review of the applications of agent technology in traffic and transportation systems. *Trans. Intell. Transport. Sys.*, vol. 11, n° 2, p. 485–497.

Doniec A., Mandiau R., Piechowiak S., Espié S. (2008). A behavioral multi-agent model for road traffic simulation. *J. Eng. Appl. of AI*, p. 1443–1454.

Dresner K., Stone P. (2007). Sharing the road: Autonomous vehicles meet human drivers. In *20th Int. Joint Conf. on Artificial Intelligence (IJCAI'2007)*, p. 1263–1268.

Espié S., Saad F., Schnetzler B., Bourlier F., Djemane N. (1994). Microscopic traffic simulation and driver behaviour modelling: the ARCHISIM project. In *Conf. Road Safety in Europe and Strategic Highway Research Program (SHRP)*, p. 22–31.

Gibson J. (1979). *The ecological approach to visual perception*. Hillsdale.

Girres J.-F., Touya G. (2010). Quality assessment of the french OpenStreetMap dataset. *Transactions in GIS*, vol. 14, n° 4, p. 435–459.

Koenig S., Likhachev M. (2005). Fast replanning for navigation in unknown terrain. *Transactions on Robotics*, vol. 21, n° 3, p. 354–363.

Krajzewicz D., Erdmann J., Behrisch M., Bieker L. (2012, December). Recent development and applications of SUMO - Simulation of Urban MObility. *Int. J. on Advances in Systems and Measurements*, vol. 5, n° 3–4, p. 128–138.

Kubera Y., Mathieu P., Picault S. (2010). Everything can be agent! In W. van der Hoek *et al.* (Eds.), *9th Int. Joint Conf. on Auton. Agents and Multi-Agent Systems (AAMAS)*, p. 1547–1548. IFAAMAS.

Kubera Y., Mathieu P., Picault S. (2011). IODA: An interaction-oriented approach for multi-agent based simulations. *J. of Auton. Agents and Multi-Agent Systems*, vol. 23, n° 3, p. 303–343.

Kuiper D., Wenkstern R. (2014). Agent vision in multi-agent based simulation systems. *J. of Auton. Agents and Multi-Agent Systems*, p. 1–31.

Lacroix B., Mathieu P., Kemeny A. (2009). Generating various and consistent behaviors in simulations. In Y. Demazeau *et al.* (Eds.), *Proceedings of the 7th International conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'2009)*, vol. 55, p. 110–119. Springer.

Lacroix B., Mathieu P., Kemeny A. (2013). Formalizing the construction of populations in multi-agent simulations. *J. Eng. App. of AI*, vol. 26, n° 1, p. 211–226.

Mandiau R., Champion A., Auberlet J.-M., Espié S., Kolski C. (2008). Behaviour based on decision matrices for a coordination between agents in a urban traffic simulation. *Appl. Intell.*, vol. 28, n° 2, p. 121–138.

Mathieu P., Picault S. (2013). From real purchase to realistic populations of simulated customers. In Y. Demazeau *et al*. (Eds.), *11th Int. Conf. on Practical Applications of Agents and Multi-Agent Systems (PAAMS)*, vol. 7879, p. 216–227. Springer.

Mokbel M., Alarabi L., Bao J., Eldawy A., Magdy A., Sarwat M. *et al*. (2013). MNTG: An extensible web-based traffic generator. In *13th Int. Symp. on Spatial and Temporal Databases (SSTD)*, vol. 8098, p. 38-55. Springer.

Osogami T., Imamichi T., Mizuta H., Morimura T., Raymond R., Suzumura T. *et al*. (2012). *IBM Mega Traffic Simulator*. Rapport technique. IBM.

Picault S., Mathieu P. (2011). An interaction-oriented model for multi-scale simulation. In T. Walsh (Ed.), *22nd Int. Joint Conf. on Artificial Intelligence (IJCAI'2011)*, p. 332–337.

Raney B., Nagel K. (2006). An improved framework for large-scale multi-agent simulations of travel behavior. *Towards better performing European Transportation Systems*, p. 305–347.

Taillandier P., Vo D.-A., Amouroux E., Drogoul A. (2012). GAMA: A simulation platform that integrates geographical information data, agent-based modeling and multi-scale control. In N. Desai *et al*. (Eds.), *Principles and Practice of Multi-Agent Systems*, vol. 7057, p. 242–258. Springer.

Tlig M., Buffet O., Simonin O. (2012). Cooperative behaviors for the self-regulation of autonomous vehicles in space sharing conflicts. In *IEEE 24th International Conference on Tools with Artificial Intelligence (ICTAI)*, p. 1126-1132. IEEE.

Visvalingam M., Wyatt J. (1993, 5). Line generalization by repeated elimination of points. *Cartographic Journal*, vol. 30, n° 1, p. 46–51.

Wilensky U. (1999). *Netlogo.* Consulté sur http://ccl.northwestern.edu/netlogo/

Yang Q. (1997). *Simulation laboratory for evaluating dynamic traffic management systems*. Thèse de doctorat, Massachusetts Institute of Technology.