

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

ALGORITHME ÉCOÉNERGÉTIQUE POUR LA RECONSTRUCTION DE
CHEMINS ET LA DÉTECTION DU MOYEN DE TRANSPORT LORS DE
DÉPLACEMENTS

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN INFORMATIQUE

PAR

JONATHAN MILOT

JANVIER 2019

UNIVERSITÉ DU QUÉBEC À MONTRÉAL
Service des bibliothèques

Avertissement

La diffusion de ce mémoire se fait dans le respect des droits de son auteur, qui a signé le formulaire *Autorisation de reproduire et de diffuser un travail de recherche de cycles supérieurs* (SDU-522 – Rév.07-2011). Cette autorisation stipule que «conformément à l'article 11 du Règlement no 8 des études de cycles supérieurs, [l'auteur] concède à l'Université du Québec à Montréal une licence non exclusive d'utilisation et de publication de la totalité ou d'une partie importante de [son] travail de recherche pour des fins pédagogiques et non commerciales. Plus précisément, [l'auteur] autorise l'Université du Québec à Montréal à reproduire, diffuser, prêter, distribuer ou vendre des copies de [son] travail de recherche à des fins non commerciales sur quelque support que ce soit, y compris l'Internet. Cette licence et cette autorisation n'entraînent pas une renonciation de [la] part [de l'auteur] à [ses] droits moraux ni à [ses] droits de propriété intellectuelle. Sauf entente contraire, [l'auteur] conserve la liberté de diffuser et de commercialiser ou non ce travail dont [il] possède un exemplaire.»

REMERCIEMENTS

Je souhaite tout d'abord remercier le professeur Éric Beaudry, pour sa direction et ses conseils tout au long de ma maîtrise.

J'aimerais aussi remercier mes collègues étudiants du GDAC (Jean, Mathieu, Jaël et Éric) que j'ai côtoyés durant mes études supérieures pour le support moral, l'aide et l'amitié qu'ils m'ont apportés.

Je remercie aussi mon raton pour tout, tout simplement.

Finalement, je remercie aussi l'Université du Québec à Montréal et le Fonds de recherche du Québec – Nature et technologies pour l'appui financier au travers des bourses de recherche qu'ils m'ont octroyées durant mes études.

TABLE DES MATIÈRES

LISTE DES FIGURES	ix
LISTE DES TABLEAUX	xi
RÉSUMÉ	xiii
INTRODUCTION	1
CHAPITRE I	
ÉTAT DE L'ART SUR LA RECONSTRUCTION DE CHEMINS	7
CHAPITRE II	
RECONSTRUCTION DE CHEMINS	11
2.1 Méthode	12
2.1.1 Définitions	14
2.1.2 Données utilisées	14
2.1.3 Initialisation	16
2.1.4 Évolution	17
2.1.5 Arrêt de la simulation	21
2.1.6 Reconstruction des chemins	23
2.2 Évaluation	23
2.2.1 Simulation de trajet	24
2.2.2 Construction d'une méthode de référence	26
2.2.3 Comparaison de la méthode de référence et de notre méthode	27
2.3 Consommation énergétique	31
2.3.1 Méthodes pour mesurer la consommation énergétique	31
2.3.2 Expérience sur la consommation énergétique	34
2.4 Conclusion	35
CHAPITRE III	

ÉTAT DE L'ART SUR LA DÉTECTION DU MOYEN DE TRANSPORT	37
3.1 Préparation des données	37
3.1.1 Cueillette des données	37
3.1.2 Prétraitement des données	44
3.1.3 Segmentation des traces	45
3.2 Détection du moyen de transport	48
3.2.1 Biljecki <i>et al.</i> (2013)	48
3.2.2 Bolbol <i>et al.</i> (2012)	51
3.2.3 Byon et Liang (2014)	53
3.2.4 Gonzalez <i>et al.</i> (2010)	55
3.2.5 Jahangiri et Rakha (2015)	56
3.2.6 Lari (2015)	58
3.2.7 Nitsche <i>et al.</i> (2014)	58
3.2.8 Reddy <i>et al.</i> (2010)	60
3.2.9 Shin <i>et al.</i> (2015)	60
3.2.10 Stenneth <i>et al.</i> (2011)	62
3.2.11 Xiao <i>et al.</i> (2015a)	63
3.2.12 Yu <i>et al.</i> (2014)	64
3.3 Consommation énergétique	68
3.4 Conclusion	69
CHAPITRE IV	
DÉTECTION DU MOYEN DE TRANSPORT	71
4.1 Modèle	72
4.1.1 Définitions et hypothèses	72
4.1.2 Filtre à particules	73
4.1.3 Stratégie de rééchantillonnage	73
4.1.4 Changement de moyen de transport	78

4.2	Expérimentations	79
4.2.1	Données du modèle	79
4.2.2	Collecte des données GPS	80
4.3	Résultats	81
4.3.1	Détection du moyen de transport	82
4.3.2	Reconstruction de chemins	83
4.3.3	Consommation énergétique	85
	CONCLUSION	89

LISTE DES FIGURES

Figure	Page
2.1 Évolution d'une simulation des déplacements possibles	13
2.2 Vitesses sur les segments de routes principaux de la ville de Montréal	17
2.3 Création de nouvelles particules dans les grappes	19
2.4 Différence entre les chemins de deux particules	20
2.5 Évolution du score de la simulation de la figure 2.1 sur 90 secondes.	22
2.6 Exemple de résultats de l'approche proposée par rapport à ceux de la réimplémentation de Li <i>et al.</i> (2016)	29
2.7 Consommation énergétique d'une charge de travail incrémentale du processeur	33
3.1 Exemple du processus de classification de Bolbol <i>et al.</i> (2012) . .	52
3.2 Réseau de neurones pour la détection multimodale utilisé par Byon et Liang (2014)	54
3.3 Réseau de neurones pour la détection voiture/non-voiture utilisé par Byon et Liang (2014)	55
3.4 Schéma de l'approche de Nitsche <i>et al.</i> (2014)	59
3.5 Schéma de l'approche de Reddy <i>et al.</i> (2010)	62
3.6 Profils d'accélération calculés par Shin <i>et al.</i> (2015) pour le train/ tram, la voiture et l'autobus	63
3.7 Schéma du réseau de neurones de Xiao <i>et al.</i> (2015a)	65
4.1 Discrétisation de $\mathcal{N}(0, 1)$ à un intervalle de σ sur l'intervalle $[-2\sigma, 2\sigma]$ avec les particules ainsi créées montrées sur une arête avec leur vrai- semblance associée	75

4.2	Modélisation de la consommation énergétique selon le fréquence GPS pour différents λ	76
4.3	Exemple de l'évolution de $Score(\mathcal{X}_t)$ pour trois différentes valeurs de λ sur une période de 60 secondes	77
4.4	Exemple d'une possible confusion sur le véritable arrêt d'autobus pris par l'utilisateur	84

LISTE DES TABLEAUX

Tableau	Page
0.1 Résumé sommaire de certaines approches de détection du moyen de transport	4
2.1 Comparaison des résultats de Li <i>et al.</i> (2016) et de notre implémentation	27
2.2 Erreurs moyennes de l'approche proposée et de Li <i>et al.</i> (2016) pour différentes fréquences GPS.	30
2.3 Énergie consommée selon la fréquence de lectures GPS	34
3.1 Techniques de cueillette de données pour la détection du moyen dans transport dans la littérature	42
3.2 Matrice de confusions de Jahangiri et Rakha (2015) avec un kNN	57
3.3 Matrice de confusions de Jahangiri et Rakha (2015) avec un SVM	57
3.4 Matrice de confusions de Jahangiri et Rakha (2015) avec une forêt d'arbres de décisions	57
3.5 Résultats obtenus par Lari (2015)	58
3.6 Résultats obtenus par Nitsche <i>et al.</i> (2014)	60
3.7 Résultats obtenus par Reddy <i>et al.</i> (2010)	61
3.8 Matrice de confusion de Shin <i>et al.</i> (2015)	61
3.9 Résultats obtenus par Stenneth <i>et al.</i> (2011)	64
3.10 Matrice de confusion du réseau de neurones de Xiao <i>et al.</i> (2015a)	64
3.11 Matrice de confusion du SVM de Yu <i>et al.</i> (2014)	65
3.12 Tableau récapitulatif des articles étudiés sur la détection du moyen de transport	66

4.1	Données GPS collectées	81
4.2	Précision de l'algorithme selon différents λ	82
4.3	Matrice de confusion pour la détection du moyen de transport avec $\lambda = 0,05$	84
4.4	Consommation d'énergie selon la fréquence du GPS	86

RÉSUMÉ

Ce mémoire propose un algorithme permettant de reconstruire l'itinéraire d'une personne à l'aide de la puce GPS intégrée à son téléphone intelligent. Plus particulièrement, il s'agit de déterminer les segments de routes qu'elle a empruntés ainsi que les moyens de transport utilisés pour se déplacer. L'aspect novateur de l'algorithme est qu'il module la fréquence du GPS selon la configuration du réseau de transport pour chaque lecture GPS. Ainsi, la fréquence du GPS est choisie de sorte à obtenir un compromis intéressant entre la consommation d'énergie du téléphone et la précision de l'algorithme. Pour ce faire, un modèle de déplacement en voiture a été dans un premier temps développé comme preuve de concept afin de ne considérer que la reconstruction de chemins. Dans un second temps, le modèle a été étendu à un réseau de transport complet (voiture, marche, transport en commun) afin d'inclure la détection du moyen de transport. Avec ces modèles, il est possible d'estimer l'évolution de l'espace d'état de croyance de l'utilisateur et d'utiliser un filtre à particules pour estimer l'état de croyance de l'utilisateur tout au long de son déplacement. Des expérimentations démontrent la plus grande précision et la plus petite consommation d'énergie de l'algorithme proposé comparativement aux méthodes vues dans la littérature scientifique.

INTRODUCTION

Depuis une dizaine d'années, la popularité croissante des téléphones intelligents a fait émerger de nombreuses applications exploitant divers capteurs, dont le GPS, l'accéléromètre et le gyroscope. Reconnaissance d'activité (Cheng *et al.*, 2017), reconstruction des chemins empruntés (Chung et Shalaby, 2005; Li *et al.*, 2016) et détection des moyens de transport empruntés (Byon et Liang, 2014; Shin *et al.*, 2015; Stenneth *et al.*, 2011) sont des exemples de leur utilisation. D'un point de vue de la recherche, ces applications facilitent la collecte et le prétraitement de données sur les individus participant à des études portant sur leurs activités quotidiennes (Jankowska *et al.*, 2015; Patterson et Fitzsimmons, 2016).

Particulièrement, lors d'études sur le trafic en urbanisme, on cherche souvent à connaître les chemins empruntés et les moyens de transport utilisés par les habitants d'une région lors de leurs déplacements. Traditionnellement, cette collecte de données se faisait à la main (Murakami *et al.*, 1997). Les participants à ces études devaient donc manuellement noter différentes informations, telles que les points et les heures de départ et d'arrivée, les moyens de transport, etc. de leurs déplacements. Cette méthode est cependant sujette à l'erreur humaine et ne permet pas de déterminer avec certitude les chemins empruntés. Pour pallier ce problème, Patterson et Fitzsimmons (2016) ont proposé l'utilisation d'une application pour automatiser le processus de collecte des données. La puce GPS des téléphones intelligents des participants est utilisée afin de déterminer leur position durant leurs déplacements. Dans l'application de Patterson et Fitzsimmons (2016), les lectures GPS sont faites à une fréquence fixe de 60 secondes, puis un algorithme de reconstruction de chemins est appliqué sur celles-ci. Les utilisateurs

doivent ensuite répondre à un court questionnaire afin de préciser, entre autres, le ou les moyens de transport utilisés. Cependant, ces applications de reconstruction de chemins présentent des inconvénients majeurs.

Tout d’abord, il faut savoir que l’utilisation des services de géolocalisation sur téléphones intelligents est couteuse en énergie. L’utilisation de la puce GPS en continu (c’est-à-dire à une fréquence de 1 Hz) consomme 143,1 mW sur les modèles HTC Dream et Google Nexus One (Carroll et Heiser, 2010). Selon nos expérimentations, la consommation est plutôt de 439,3 mW pour un Samsung Galaxy S8 et de 397,44 mW pour un Zenphone 4 Max. La durée de la batterie de ces téléphones intelligents était alors de 8 à 10 heures lors d’une utilisation normale (réception cellulaire et Wifi activés, navigation 4G lors de déplacements en transport en commun, etc.) lorsque la géolocalisation en continu était activée durant les déplacements.

Faire des lectures GPS en continu est donc difficilement acceptable pour les utilisateurs, puisque cela réduit l’autonomie de leur téléphone intelligent à moins d’une journée. Il faut donc diminuer la fréquence de lectures GPS pour préserver la batterie des téléphones intelligents. Cependant, une diminution de la fréquence diminue la précision des algorithmes de reconstruction de chemins. Il existe donc une concurrence entre la précision de l’algorithme et la durée de vie de la batterie d’un téléphone intelligent. En d’autres mots, il faut rechercher un compromis idéal entre la qualité de l’estimation et la consommation d’énergie.

Typiquement, ce compromis est déterminé de manière empirique. C’est exactement ce que Patterson et Fitzsimmons (2016) font. Comme mentionné plus tôt, leur application effectue des lectures GPS aux 60 secondes et obtient un taux d’erreur de 20 % sur les chemins estimés. Comparativement, on peut citer Li *et al.* (2016) qui propose un autre algorithme de reconstruction de chemins ar-

rivant à une erreur de 6.5 % avec des lectures aux 20 secondes. Cependant, ces erreurs ne sont que des moyennes sur tous les chemins empruntés durant leurs expérimentations. Il est évident qu'il est plus facile de reconstruire un chemin lorsque le déplacement est fait dans un environnement avec une faible densité de routes. Le taux d'erreur varie donc selon la configuration routière pour une même fréquence GPS. Conséquemment, le compromis idéal entre la consommation énergétique et la précision est variable selon la configuration routière dans laquelle évolue l'utilisateur. Par exemple, un plus grand nombre de lectures GPS sont nécessaires au centre-ville de Montréal que sur l'Autoroute 20 à la hauteur de Saint-Louis-de-Blandford pour obtenir une précision comparable sur le chemin reconstruit.

La tâche se complexifie encore plus lorsqu'on tente de détecter le moyen de transport utilisé durant les déplacements d'un utilisateur, ce que ne font pas Patterson et Fitzsimmons (2016) et Li *et al.* (2016). Plusieurs approches pour déterminer le moyen de transport d'utilisateurs ont été proposées, mais sans reconstruire le chemin emprunté (Stenneth *et al.*, 2011; Bolbol *et al.*, 2012; Byon et Liang, 2014; Xiao *et al.*, 2015a; Dabiri et Heaslip, 2018). La détection du moyen de transport est souvent présentée comme un problème de classification pouvant être résolu à l'aide d'algorithmes d'apprentissage machine. Durant les déplacements, des lectures GPS sont faites à de très hautes fréquences (lectures aux 1 seconde (Byon et Liang, 2014; Xiao *et al.*, 2015a) à 15 secondes (Stenneth *et al.*, 2011)). Une fois les déplacements terminés, les données recueillies sont envoyées en entrée à des réseaux de neurones (Byon et Liang, 2014; Xiao *et al.*, 2015a), des réseaux de neurones à convolution (Dabiri et Heaslip, 2018), des forêts d'arbres décisionnels (Stenneth *et al.*, 2011) ou encore des SVM (Bolbol *et al.*, 2012) préalablement entraînés. La précision moyenne de ces algorithmes varie de 83.8 % (Byon et Liang, 2014) à 94.4 % (Xiao *et al.*, 2015a). Le tableau 0.1 présente un résumé de ces approches.

Celles-ci présentent toutes la même problématique. Bien que ces précisions soient jugées acceptables par les auteurs de ces méthodes, l'intense sollicitation de la puce GPS fait en sorte que ces approches sont difficilement utilisables à grande échelle, puisqu'elle viderait la batterie de ses utilisateurs en moins d'une journée.

Tableau 0.1: Résumé sommaire de certaines approches de détection du moyen de transport

Approche	GPS (s)	Algorithme	Marche (%)	Voiture (%)	Bus (%)	Précision moyenne (%)
Stenneth <i>et al.</i> (2011)	15	Forêts d'arbres décisionnels	98.9	80.8	93.0	93.8
Bolbol <i>et al.</i> (2012)	60	Machines à vecteurs de support	93.8	88.5	58.3	88.0
Byon et Liang (2014)	1	Réseaux de neurones	95.0	72.0	84.0	83.8
Xiao <i>et al.</i> (2015a)	1	Réseaux de neurones	98.5	94.2	88.4	94.4
Dabiri et Heaslip (2018)	1-5	Réseaux de neurones à convolution	95.7	67.4	81.1	84.8

Dans ce mémoire, nous nous intéressons donc à proposer une approche permettant (1) la reconstruction de chemins et (2) la détection du moyen de transport, le tout dans une optique d'efficacité énergétique. L'objectif est donc de proposer des approches ayant une précision au pire similaire à celles présentes dans la littérature, mais qui consomme moins d'énergie que ces dernières. Ce mémoire se divise en deux principales parties.

Premièrement, un algorithme simple de reconstruction de chemins effectués en voiture est proposé (chapitres 1 et 2) comme preuve de concept. Son aspect novateur réside dans sa capacité à adapter dynamiquement la fréquence à laquelle les lectures GPS sont effectuées afin d'être énergétiquement efficient. Basé sur l'algorithme de filtre à particules, il simule les déplacements possibles de l'utilisateur et évalue pour chaque seconde simulée une erreur moyenne sur le chemin reconstruit.

Il cherche ainsi à déterminer l'instant t le plus tard où l'erreur demeure acceptable. Une nouvelle lecture GPS est faite à cet instant t et le processus de simulation recommence. Pour ce faire, un modèle de déplacement alimenté par des données ouvertes de la Ville de Montréal¹ et la carte d'*OpenStreetMap*² a été développé. Les expérimentations réalisées démontrent que cet algorithme obtient une erreur moyenne sur le chemin estimé de 3.65 % en faisant une lecture aux 68.1 secondes en moyenne. Ceci se traduit par une réduction de 19.35 points de pourcentage sur l'erreur comparativement à des méthodes faisant des lectures aux 60 secondes et une réduction de 44 % sur l'énergie consommée comparativement aux approches ayant une erreur moyenne similaire à la nôtre, mais faisant des lectures aux 30 secondes et moins.

Deuxièmement, l'algorithme de reconstruction de chemins est formalisé et étendu à la détection du moyen de transport (chapitres 3 et 4). Il implémente un filtre à particules (Thrun *et al.*, 2005) afin d'estimer l'espace d'états du téléphone intelligent (chemin emprunté et moyen de transport pour chaque segment de route) dans le futur. Une composante temporelle est ajoutée dans l'évaluation des états de l'utilisateur afin de pondérer l'erreur estimée à la consommation énergétique. Le modèle de déplacement est enrichi avec les données ouvertes des horaires de la

1. Ville de Montréal. «Temps de parcours sur des segments routiers (historique)». <http://donnees.ville.montreal.qc.ca/dataset/> [consulté le 17 avril 2018]

2. OpenStreetMap. «Planet dump». <https://wiki.openstreetmap.org/wiki/Planet.osm> [consulté le 6 octobre 2017]

STM³ et du RTL⁴ afin de permettre les déplacements en transport en commun. Les expérimentations réalisées démontrent que l'approche proposée atteint une précision de 96.3 % sur la détection du moyen de transport tout en consommant 15.0 % d'énergie de moins que les meilleures approches existantes. Pour une précision similaire aux autres approches, la réduction de la consommation d'énergie varie de 37.5 % à 56.0 %.

3. Société de Transport de Montréal. «Développeurs». <http://www.stm.info/fr/a-propos/developpeurs> [consulté le 26 avril 2018]

4. Réseau de Transport de Longueuil. «Archive de fichiers GTFS». <http://www.rtl-longueuil.qc.ca/fr-CA/donnees-ouvertes/fichiers-gtfs/> [consulté le 26 avril 2018]

CHAPITRE I

ÉTAT DE L'ART SUR LA RECONSTRUCTION DE CHEMINS

La grande consommation d'énergie de la géolocalisation dans les applications mobiles est un problème connu¹ et les approches proposées pour le pallier sont variées.

Tel que mentionné plus tôt, Patterson et Fitzsimmons (2016) cherchent à connaître les chemins empruntés par la population du grand Montréal. Ils fixent une fréquence constante de 60 secondes et obtiennent une erreur moyenne de 20 % sur les chemins estimés. Cependant, aucune analyse ni justification n'est faite quant à cette fréquence. Il n'est pas mentionné si d'autres fréquences ont été testées ou si une erreur moyenne de 20 % est acceptable pour leurs besoins. Une certaine optimisation de la consommation d'énergie est faite avec différents stratagèmes pour démarrer et arrêter automatiquement la collecte de données au courant de la journée. Ainsi, la collecte s'arrête si le téléphone intelligent se connecte à un réseau Wifi «connu» ou si l'utilisateur est considéré comme immobile par les services de reconnaissance d'activité fournis par Android. La collecte recommence ensuite si l'utilisateur se déplace de 100 mètres ou plus selon la localisation par Wifi/tours cellulaire. La métrique utilisée pour évaluer la consommation d'énergie est que leur application *«would not affect a user's charging schedule, and in*

1. Android Developers. «Location strategies». <https://developer.android.com/guide/topics/location/strategies> [consulté le 6 septembre 2017]

particular that a user could make it through the day without having to charge their phone» (Patterson et Fitzsimmons, 2016). Ils jugent leur optimisation suffisante, puisqu'un grand nombre d'utilisateurs ont participé à leur étude pendant plusieurs jours.

La fréquence des lectures GPS a une influence directe sur la qualité des chemins estimés. Li *et al.* (2016) étudient la variation du taux d'erreur selon plusieurs fréquences constantes sur leur algorithme. Ce dernier consiste à associer les lectures GPS à des noeuds de la carte d'*OpenStreetMap*, la liste des noeuds ainsi obtenue donnant le chemin emprunté. Une liste de noeuds candidats est déterminée pour chaque lecture GPS selon une distance maximale, puis cette liste est triée par distance de la lecture GPS. En traitant les lectures GPS de manière chronologique, un noeud leur est associé de sorte à être voisin du noeud associé à la lecture GPS précédente. Si l'algorithme rencontre une lecture GPS n'ayant aucun noeud candidat étant voisin du noeud précédent, la distance maximale de sélection des noeuds candidats est augmentée et l'association des noeuds aux lectures GPS recommence du début. Li *et al.* (2016) font varier leur fréquence de lectures GPS d'une lecture aux 20 à 300 secondes et trouvent que le taux d'erreur se stabilise à 23 % après 120 secondes.

L'idée de faire varier la fréquence pendant les déplacements n'est pas nouvelle. Nakagawa *et al.* (2013) s'intéressent à détecter le moment où un utilisateur entre ou sort de zones géographiques prédéterminées. Ils proposent une approche faisant varier la fréquence selon la distance des utilisateurs des zones géographiques et leurs vitesses de déplacement. Plus un utilisateur se rapproche d'une zone prédéterminée, plus la fréquence augmente. S'il en est loin, la fréquence diminue. La fréquence est aussi modulée en fonction de la vitesse d'approche des zones géographiques. Leur technique se base uniquement sur les distances à vol d'oiseau et ne considère pas les routes.

Paek *et al.* (2010) proposent un cadriciel, *RaPS*, permettant de déterminer quand démarrer et arrêter le GPS. Bien qu'ils ne cherchent pas à reconstruire le déplacement d'un utilisateur, l'ensemble de leurs techniques est intéressant. Ils utilisent l'historique des emplacements de l'utilisateur pour estimer sa vitesse et ne démarrer le GPS que si l'incertitude sur la position dépasse un certain seuil. De plus, ils utilisent l'accéléromètre et le réseau *Bluetooth* pour préciser cette incertitude. Le *celltower-RSS blacklisting* est utilisé pour déterminer si le GPS est indisponible (ex. : à l'intérieur).

Duribreux *et al.* (2014) essaient d'apprendre les déplacements de ses utilisateurs afin de réduire le nombre de lectures GPS nécessaires pour suivre une personne. Ainsi, si leur modèle prédit correctement la prochaine position de l'utilisateur, le temps avant la prochaine lecture augmente. Dans le cas d'une erreur (prédiction à plus de 30 mètres), ce temps diminue. Bien que leur méthode diminue le nombre total de lectures GPS effectuées, son coût total en énergie demeure supérieur. Leur algorithme consomme en effet plus d'énergie que l'économie obtenue grâce à la réduction du nombre de lectures GPS. Un autre désavantage de leur méthode est qu'elle ne peut fonctionner que pour des chemins déjà effectués dans le passé.

Fang et Zimmermann (2011) proposent une méthode permettant d'ajuster la fréquence selon l'état de l'utilisateur : T_1 : *City-driving State*, T_2 : *Highway-driving State* et T_3 : *Stopped State*. Ils utilisent ensuite un algorithme de reconstruction de chemins basé sur les modèles cachés de Markov. Ils arguent que ceci permet de ne pas utiliser de distance maximale pour sélectionner les segments de routes candidats telle que Li *et al.* (2016) utilisent. Durant leurs expérimentations, ils font varier la fréquence de lectures GPS de 5 à 30 secondes pour leur comparaison. Ils utilisent $T_1 = 5$ et font varier T_2 avec les valeurs [5, 10, 15, 20, 25, 30]. Des erreurs moyennes d'environ 3.8 % pour une lecture aux 25 secondes et de 2.5 % pour une lecture aux 30 secondes sont obtenues.

L'étude de l'état de l'art met en évidence le problème suivant : soit la fréquence de lectures GPS est relativement basse (> 60 secondes) et le taux d'erreurs est de plus de 20 % (Patterson et Fitzsimmons, 2016; Li *et al.*, 2016), soit elle est grande et le taux d'erreur est acceptable (Fang et Zimmermann, 2011). Ceci est principalement dû au problème de *arc-skipping*, un problème qui survient lorsque qu'il existe un ou plusieurs segments de routes non détectés entre les lectures GPS. Fang et Zimmermann (2011) ont choisi délibérément une fréquence plus grande pour éviter ce problème. La méthode conventionnelle pour pallier ce problème est de relier par le plus court chemin les lectures GPS (Li *et al.*, 2016; Fang et Zimmermann, 2011), ce qui induit nécessairement des erreurs, puisqu'un utilisateur n'utilise pas forcément le chemin le plus court entre deux points géographiques à cause du trafic ou de ses préférences de déplacement, par exemple.

CHAPITRE II

RECONSTRUCTION DE CHEMINS

La première approche proposée vise à reconstruire le chemin d'un utilisateur fait en voiture. À partir des lectures GPS précédentes, elle consiste à simuler les déplacements de l'utilisateur afin d'estimer l'évolution possible de son état (position). Cette estimation permet de calculer le prochain instant où il sera intéressant de faire une lecture GPS. Cette méthode permet d'obtenir un compromis entre la consommation énergétique de l'algorithme et sa précision.

Comme l'état initial de l'utilisateur (position et vitesse) n'est pas directement observable, mais est estimé à partir d'un capteur externe (GPS), une incertitude existe. La transition d'un état vers un autre, c'est-à-dire la nouvelle position d'un utilisateur après un certain temps, présente aussi une incertitude, puisque l'utilisateur peut faire varier sa vitesse de déplacement entre deux lectures GPS. De plus, la position sur un segment de route est une variable continue. Ainsi, alors qu'une incertitude sur la position sur un unique segment de route peut facilement être modélisée en une dimension par une loi gaussienne, elle est d'autant plus difficile à modéliser dans un graphe (réseau routier) où la position est une variable continue sur les arrêtes du graphe. Les filtres à particules (ou *Sequential Monte Carlo Filter*) (Cappe *et al.*, 2007) est une technique permettant de traiter ce type de problème, à savoir des problèmes à domaine continu et dynamique, et à ob-

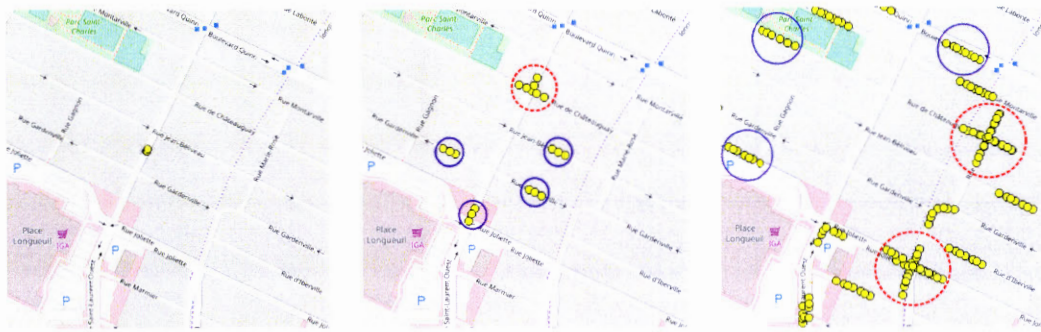
servabilité partielle avec des variations aléatoires (du bruit). Dans le domaine de la planification, il permet notamment de briser le fléau de la dimension dans les POMDPs (Silver et Veness, 2010) et servent à représenter les états de croyance dans le *robot motion planning* (Bai *et al.*, 2010). L’approche présentée dans ce chapitre se base donc sur cet algorithme.

2.1 Méthode

Concrètement, la méthode effectue une première lecture GPS afin de connaître la position et la vitesse initiale du téléphone. Une simulation des déplacements possibles est alors effectuée à partir de cette position à l’aide de particules afin d’évaluer les positions possibles de l’utilisateur dans le futur. Éventuellement, certaines positions pourront s’expliquer par plusieurs chemins différents. Si une lecture GPS était effectuée à ces positions, il faudrait alors déterminer quel chemin serait le plus vraisemblable. Cette situation induit un degré d’ambiguïté sur le chemin estimé. Lorsqu’un certain seuil d’ambiguïté est atteint dans la simulation, elle est arrêtée et l’état de la simulation présentant le meilleur compromis entre le temps avant la prochaine lecture (et donc de consommation d’énergie) et l’incertitude sur le chemin estimé est choisi. La prochaine lecture GPS se fait lorsque ce temps sera atteint dans la réalité. De plus, simuler les déplacements permet de pallier le problème de *arc-skipping*, tel que présenté dans le chapitre 1. Avec cette technique, les segments de routes empruntées entre deux lectures GPS sont connus. Surtout, il est possible de faire une lecture GPS avant qu’il y ait une multitude de segments de routes candidats pour les relier.

La figure 2.1 montre l’évolution d’une simulation après 1, 10 et 30 secondes. On remarque que dans la figure 2.1c, il n’y aurait pas d’ambiguïté sur le chemin emprunté si une lecture GPS était près d’un des groupes de particules encadrés

en bleu avec un trait plein. Cependant, si une lecture GPS était près d'un des groupes de particules encerclés en rouge avec un trait pointillé, il serait difficile de déterminer quelle grappe représenterait le chemin emprunté par l'utilisateur. C'est ce type de situation que l'algorithme cherche à éviter.



(a) Simulation après 1 seconde. Il existe deux grappes de deux particules ayant tous la même position.

(b) Simulation après 10 secondes. Encerclées en bleu avec un trait plein, quatre grappes de trois particules. Encerclées en rouge avec un trait pointillé, une grappe étant en train de se subdiviser en trois autres grappes.

(c) Simulation après 30 secondes. Encerclées en bleu avec un trait plein, des exemples de grappes de 6 à 8 particules. Encerclées en rouge avec un trait pointillé, plusieurs grappes se chevauchant.

Figure 2.1: Évolution d'une simulation des déplacements possibles

Les prochaines sous-sections détaillent les différents aspects de l'approche : définitions, données d'entrées, initialisation de la simulation, évolution de la simulation et son arrêt, et la reconstruction de chemins.

2.1.1 Définitions

- **Carte.** Une carte est représenté par un graphe $G = (V, E)$ où V est un ensemble de noeud et E , un ensemble de segment de routes. De plus, $\forall e \in E, e = (v_1, v_2)$ où $v_1 \in V$ et $v_2 \in V$.
- **Noeud.** Un noeud est une position géographique (latitude, longitude) servant à définir des segments de route. Une liste de segments entrants et une liste de segments sortants sont attachées à chaque noeud.
- **Segment de route.** Un segment de route est un segment sur lequel une particule peut se déplacer. Il s'agit des arêtes du graphe représentant la carte. Il se caractérise par deux noeuds déterminant ses extrémités, une vitesse maximale permise et un sens de circulation (unidirectionnel ou bidirectionnel).
- **Particule.** Une particule est un état possible dans lequel peut se trouver le capteur GPS, soit celui d'un téléphone intelligent dans notre cas. Elle se caractérise par sa position géographique (latitude et longitude), sa direction, sa vitesse et sa vraisemblance.
- **Grappe de particules.** Une grappe de particules est un ensemble de particules ayant le même chemin. Elle représente l'étendue de l'incertitude sur la position de l'utilisateur s'il emprunte le chemin représenté par la grappe. Par exemple, dans la figure 2.1b, on identifie facilement quatre grappes de trois particules, ainsi que la subdivision d'une grappe en trois autres.

2.1.2 Données utilisées

Pour réaliser les simulations, nous utilisons les données d'*OpenStreetMap*, un projet communautaire de cartographie offrant ses données sous licence libre. *OpenS-*

treetMap nous permet de connaître la configuration du réseau routier, les types de routes et les limites de vitesse. Le modèle divise aussi les segments de routes en deux catégories : les segments résidentiels, à savoir ceux étant de type *residential*, *road* ou *living_street* dans la terminologie d'*OpenStreetMap* et les non résidentiels.

La Ville de Montréal utilise un réseau de capteurs *Bluetooth* pour calculer le temps de trajet sur certains segments routiers¹ dont les résultats sont présentés dans la figure 2.2. Les capteurs fonctionnent avec tout dispositif électronique utilisant la norme *Bluetooth* et sont installés à des intersections de routes stratégiques. Les capteurs à des intersections adjacentes forment ainsi des segments sur lesquels il est possible de calculer le temps de passage (et donc la vitesse) des individus le traversant. Ces données étant ouvertes, il est possible de s'en servir pour déterminer la vitesse moyenne des déplacements. Nous utilisons ces données pour construire un modèle de déplacement à priori pour nos simulations. Cependant, puisque les capteurs *Bluetooth* ne peuvent identifier le moyen de transport du dispositif électronique, il est important de d'abord nettoyer les données. Trois moyens de transport sont représentés dans ces données :

- **À pied.** On remarque deux pics pour les valeurs ≤ 5 km/h. Puisque la vitesse de marche des piétons varie de $0,96 \pm 0,18$ m/s ($3,46 \pm 0,65$ km/h) à $1,37 \pm 0,19$ m/s ($4,93 \pm 0,68$ km/h) selon les endroits et le sexe des individus (Chandra et Bharti, 2013), on présuppose que ces pics correspondent à des piétons. On retire donc toutes vitesses sous les 5,61 km/h (moyenne maximale plus un écart type) de notre analyse.
- **Vélo.** Les cyclistes roulent en moyenne $15,61 \pm 6,28$ km/h (Thompson *et al.*, 1997), mais comme ce moyen de transport ne représente que 1,7 % des

1. Ville de Montréal. «Temps de parcours sur des segments routiers (historique)». <http://donnees.ville.montreal.qc.ca/dataset/> [consulté le 17 avril 2018]

déplacements dans la région de Montréal², nous les considérons négligeables et ne cherchons pas à les enlever.

- **Voiture.** C'est le moyen de transport dont on veut le profil. Cependant, comme notre modèle ne tient pas compte du trafic, on retire les lectures ayant été fait pendant les heures de pointe, soit entre 6 h à 9 h et entre 15 h et 18 h.

Une fois ce nettoyage effectué, on trouve une vitesse moyenne de $28,60 \pm 9,49$ km/h. C'est la vitesse à priori de déplacement des utilisateurs dans notre modèle en ville. Puisqu'aucune donnée n'est prise sur une autoroute, les limites de vitesse de ces dernières seront simplement utilisées.

2.1.3 Initialisation

La méthode nécessite en entrée une vitesse V_0 , un écart-type sur la vitesse σ_V et une position C . La projection de la position C sur le segment de route le plus près est considérée comme la position de départ. À partir de cet position, une grappe de particules est créée par direction de circulation permise. Chacune des grappes contient initialement deux particules : P_1 et P_2 . Les vitesses des particules sont fixées à $P_1^{Vitesse} = V_0 + \sigma_V$ et $P_2^{Vitesse} = V_0 - \sigma_V$. Cet écart de vitesse sert à simuler l'incertitude sur la vitesse moyenne de l'utilisateur, puisqu'il pourra modifier sa vitesse selon les autres voitures sur la route, sa propension à rouler sécuritairement, la signalisation routière, etc.

2. Statistics Canada. «Proportion of workers commuting to work by car, truck or van, by public transit, on foot, or by bicycle, census metropolitan areas (2011)». <http://www12.statcan.gc.ca/nhs-enm/2011/as-sa/99-012-x/2011003/tbl/tbl1a-eng.cfm> [consulté le 20 avril 2018]

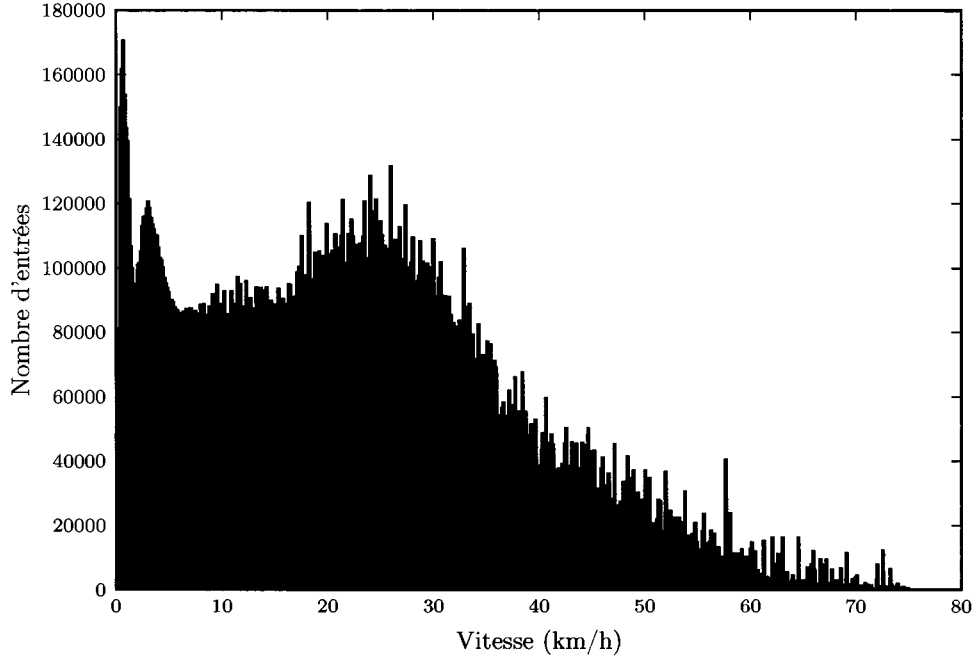


Figure 2.2: Vitesses sur les segments de routes principaux de la ville de Montréal

2.1.4 Évolution

On laisse la simulation s'exécuter. Les particules se déplacent et se multiplient afin d'explorer tous les déplacements possibles. Les conditions d'arrêt de la simulation sont décrites dans la section 2.1.5.

Le temps dans la simulation est discrétisé en pas de Δt seconde (en pratique, $\Delta t = 0,1$). À chaque pas de simulation, la position de chaque particule est mise à jour. Durant cette mise à jour, deux situations peuvent survenir :

1. Une particule P arrive au bout de son segment au noeud $N_{destination}$. Cette particule se divise alors en $n - 1$ particules, une particule sur chacun des n segments sortant de $N_{destination}$ sauf $P_{segment}$, puisqu'il est généralement

impossible ou illégale de faire demi-tour à une intersection. Il est possible de voir ce phénomène dans la figure 2.1b.

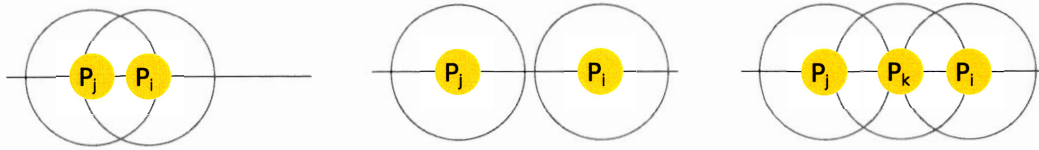
2. Une particule P n'arrive pas au bout de son segment. Dans ce cas, elle avance de $P^{Vitesse} \times \Delta t$ unités vers sa destination.

Une fois les particules mises à jour, trois actions sont effectuées successivement :

(1) Ajouter des particules aux grappes. Une grappe représente l'incertitude sur une position possible d'un téléphone intelligent. Au départ, on crée les grappes avec seulement deux particules. Puisque l'incertitude sur la position du téléphone intelligent est faible, ces deux particules sont suffisantes pour la représenter. Cependant, comme cette incertitude augmente avec le temps, il est nécessaire d'ajouter de nouvelles particules aux grappes au fil de la simulation pour conserver une granularité acceptable. Une nouvelle particule est donc créée lorsque la distance entre deux particules successives d'une même grappe est de plus de 20 mètres. La distance de 20 mètres est choisie puisque la marge d'erreur sur la position d'une lecture GPS est en moyenne d'une dizaine de mètres selon nos expérimentations. La figure 2.3 illustre ce processus.

(2) Mettre à jour le degré de vraisemblance des particules. Toutes les particules n'ont la même vraisemblance. Un degré de vraisemblance est associé à chaque particule selon deux facteurs : sa position dans sa grappe de particules et les types de routes qu'elle a empruntées. Comme l'incertitude sur une position est modélisée avec une loi gaussienne, on attribue le degré de vraisemblance avec cette loi. Ainsi, la vraisemblance d'une particule P_x dans une grappe G est défini par l'équation (2.1), où W est le degré de vraisemblance, x est la position de la particule dans la grappe, $\mu = \frac{|G|}{2}$, $\sigma = \frac{|G|}{5}$ et $|G|$ est le nombre de particules dans la grappe considérée.

De plus, si le chemin parcouru par la particule ne contient aucun segment de voie



(a) Deux particules successives d'une même grappe proches l'une de l'autre. Les cercles gris représentent la précision qu'aurait une lecture GPS effectuée à la position des particules. On remarque que les cercles représentant la marge d'erreur sur la position se chevauchent.

(b) Après un certain temps, comme $P_i^{Vitesse} > P_j^{Vitesse}$, un écart se crée entre les cercles représentant la marge d'erreur sur la position de P_i et P_j .

(c) On crée une nouvelle particule P_k de vitesse $\frac{P_i^{Vitesse} + P_j^{Vitesse}}{2}$. Avec la création de P_k , il n'y a plus d'espace entre les cercles représentant la marge d'erreur sur la position des particules.

Figure 2.3: Création de nouvelles particules dans les grappes

résidentielle, sa vraisemblance est multipliée par 20. En effet, si un utilisateur est en train de se déplacer sur une route non résidentielle (boulevard, voie principale, autoroute, etc.), il y a de fortes chances qu'il reste sur ce type de route.

$$W = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad (2.1)$$

(3) Calculer le score de l'état de la simulation. À chaque pas de simulation, un score global de l'état courant est calculé. Un score pour chaque particule est d'abord calculé. Il représente l'erreur sur le chemin estimé si la particule représente l'état réel du téléphone intelligent. En effet, si une particule P_i est la position réelle de du téléphone intelligent et qu'il existe des particules appartenant à une autre

grappe (ayant donc des chemins différents) à une distance inférieure à la marge d'erreur d'une lecture GPS, il est alors impossible de déterminer exactement que P_i est l'état réel à partir d'une lecture GPS. Il faudrait alors en choisir une de manière arbitraire et, si les chemins diffèrent considérablement entre les différentes particules, l'erreur sur le chemin estimé peut être grande. Nous reprenons une formule adaptée de Li *et al.* (2016) pour comparer les chemins de deux particules P_i et P_k , à savoir l'équation (2.2), où $|d_k^+|$ est la longueur de la fraction du chemin de P_k supplémentaire au chemin de P_i et $|d_k^-|$ la longueur de la partie manquante (voir figure 2.4). Le calcul du score d'une particule est donné par l'algorithme 1.

$$D = \frac{|d_k^+| + |d_k^-|}{|d_i|} \quad (2.2)$$

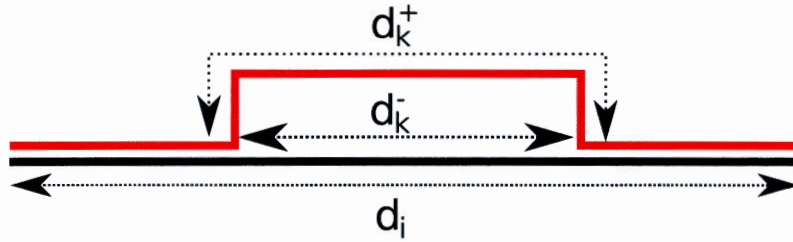


Figure 2.4: Différence entre les chemins de deux particules

Le score d'un instant donnée de la simulation (son état à un instant i) est alors défini par l'équation (2.3) et donne une valeur comprise entre 0 et 1 (1 étant aucune ambiguïté et 0 étant une ambiguïté totale).

$$Etat_i^{score} = \frac{\sum_{P \in Particles} P.likelihood \times P.score}{\sum_{P \in Particles} P.likelihood} \quad (2.3)$$

À chaque pas de simulation, le score de la simulation est calculé et gardé en

Algorithm 1 Calcul du score d'une particule

```

1: function SCORE(Particule  $p$ , Ensemble de particules  $P$ )
2:    $likelihood_{Total} = 0$ 
3:    $error_{Total} = 0$ 
4:   for Particle  $P_i$  in  $P$  do
5:     if  $P_i.distance(p) < GPS_{Accuracy}$  AND  $P_i \neq p$  then
6:        $error_{Total} += p.PathDifference(P_i)$ 
7:        $likelihood_{Total} += P_i.likelihood$ 
8:   if  $likelihood_{Total} > 0$  then
9:     Return  $1 - (error_{Total}/likelihood_{Total})$ ;
10:  else
11:    Return 1;

```

mémoire. La figure 2.5 présente l'évolution du score lors d'une simulation durant 90 secondes. On remarque que le score diminue plus la simulation avance, un comportement auquel on s'attend. En effet, plus le temps passe, plus le nombre de chemins possibles se recoupant augmente, augmentant l'ambigüité. La forte baisse observée à partir de 50 secondes est due à la collision de plusieurs grappes de particules sur des routes principales, c'est-à-dire des particules ayant un degré de vraisemblance important. Par la suite, le score de la simulation continue de descendre et il est peu probable qu'il remonte. Après 50 secondes, il y a tout simplement trop de chemins possibles se recoupant.

2.1.5 Arrêt de la simulation

Une simulation peut en théorie continuer de manière infinie, mais il arrive un moment où il est inutile de continuer. Deux critères d'arrêt sont utilisés. Le premier est une simple limite fixe de temps. Si une simulation est démarrée sur une route

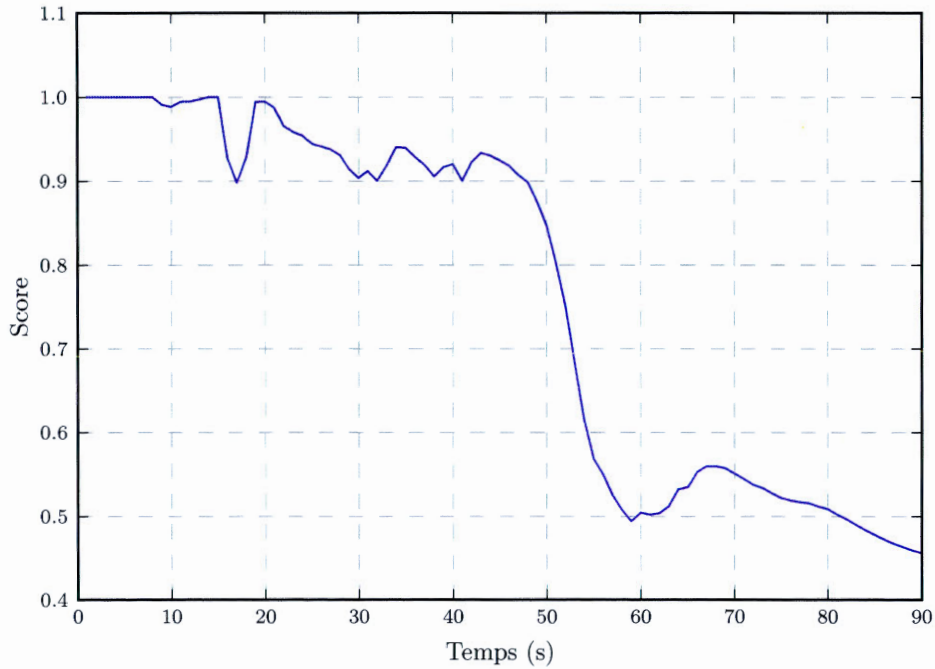


Figure 2.5: Évolution du score de la simulation de la figure 2.1 sur 90 secondes.

ayant très peu d'intersections, comme une autoroute ou une route de campagne, il peut se passer un temps excessivement long avant que plusieurs grappes de particules se forment et se rencontrent. Une limite de cinq minutes est donc fixée aux simulations puisqu'au delà de ce temps, la consommation d'énergie du GPS devient marginale. Le deuxième critère est d'arrêter la simulation lorsqu'on est confiant d'avoir rencontré l'état présentant le compromis entre la précision et l'économie d'énergie le plus intéressant. Il a été déterminé de manière empirique qu'une simulation ayant atteint un score de 0,65 et moins n'offrira pas de meilleur compromis dans le futur. Le dernier temps associé à un score de 0,75 ou plus est alors sélectionné et est appelé t_{Best} . On cherche ainsi à obtenir une erreur moyenne d'au plus 25 % sur la reconstruction de chemins, une erreur considérée acceptable dans la littérature (Li *et al.*, 2016; Patterson et Fitzsimmons, 2016).

Une fois la simulation arrêtée, une nouvelle est commencée avec la nouvelle position GPS de l'utilisateur à l'instant t_{Best} .

2.1.6 Reconstruction des chemins

Un premier traitement est effectué en amont sur la carte. Afin de limiter la consommation énergétique de l'algorithme, les temps avant les prochaines lectures sont précalculés pour chaque noeud sur un serveur à l'aide de la vitesse à priori déterminée dans la section 2.1.4. La carte et les temps sont ensuite téléchargés sur le téléphone intelligent.

Durant les déplacements, la fréquence de lectures GPS est donc déterminée par le temps associé au noeud le plus près du segment le plus près de la lecture GPS. En plus des positions GPS calculées pendant le déplacement, les vitesses de l'utilisateur associées aux positions GPS sont aussi conservées.

Une fois le déplacement de l'utilisateur terminé, les données collectées (position et vitesse) sont alors transférées à un serveur. Des simulations sont refaites pour chaque lecture GPS, mais, cette fois-ci, avec la vitesse enregistrée lors du déplacement. La durée en temps simulé de chacune de ces simulations correspond à l'intervalle de temps avant la prochaine lecture GPS. À la fin de chaque simulation, la particule la plus près de la lecture GPS suivante est sélectionnée. La concaténation des chemins des particules sélectionnées après chaque simulation donne le chemin emprunté par l'utilisateur durant son trajet.

2.2 Évaluation

L'évaluation de notre approche se fait en selon deux critères : la précision et la consommation énergétique.

L'évaluation de la précision de notre méthode se divise en deux parties :

1. **Construction d'une méthode de référence.** Afin d'effectuer une comparaison équitable (la précision des algorithmes peuvent varier selon l'environnement dans lequel ils ont été testés, les chemins estimés, etc.), une réimplémentation de la méthode de Li *et al.* (2016) a été faite. Nous comparons la précision obtenue de cette réimplémentation avec des trajets simulés dans l'état du New Jersey, USA (lieu où Li *et al.* (2016) a effectué son évaluation), avec les résultats de Li *et al.* (2016). Nous démontrons ainsi que les résultats sont similaires et concluons que notre réimplémentation est fidèle à leur implémentation. Ainsi, une comparaison avec notre méthode à fréquence dynamique est équitable.
2. **Comparaison de la méthode de référence à notre méthode.** Une fois que nous avons une méthode de référence qu'on juge pertinent avec notre réimplémentation de Li *et al.* (2016), on compare la précision de celle-ci avec notre méthode à fréquence dynamique. Évidemment, pour que la comparaison soit intéressante, les mêmes chemins sont utilisés pour les deux algorithmes. De plus, les trajets seront créés artificiellement dans la région de Montréal. Ceci nous permettra d'ensuite mesurer la consommation énergétique de ces deux algorithmes sur le terrain dans la section 2.3 en effectuant de réels trajets similaires à ceux créés artificiellement.

2.2.1 Simulation de trajet

La simulation se fait par la sélection de deux points aléatoires dans une carte d'une distance minimale de 10 km. Une limite minimale est fixée pour éviter les chemins trop courts et trop faciles à reconstruire. On calcule ensuite le plus court chemin entre ces deux points avec l'algorithme de Dijkstra. Ceci nous donne le trajet sur

lequel le déplacement est simulé. Lors du déplacement, deux variables importantes sont à simuler, l'incertitude sur les lectures GPS et la vitesse de déplacement.

Lectures GPS. Une lecture GPS comporte toujours une incertitude, et l'estimation de la précision horizontale est une mesure de cette incertitude. Elle est définie comme étant «*the radius of 68 % confidence. In other words, if you draw a circle centered at this location, and with a radius equal to the accuracy, there is a 68 % probability that the true location is inside the circle*»³. Comme cette incertitude a un impact sur la précision des méthodes de reconstruction de chemins, il est important de la simuler. On génère une précision horizontale avec une loi normale, c'est la distance à laquelle sera la lecture GPS de l'endroit à partir duquel on fait la lecture, puis on applique une rotation aléatoire sur 2π radian. Formellement, on obtient :

Soit C , la coordonnée à laquelle on souhaite faire une lecture GPS, G , la lecture GPS qu'on souhaite effectuer, $\theta \sim U[0, 2\pi[$ et $X \sim \mathcal{N}(C.longitude, Accuracy)$. Alors,

$$\begin{bmatrix} G.longitude \\ G.latitude \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X \\ 0 \end{bmatrix} + \begin{bmatrix} C.longitude \\ C.latitude \end{bmatrix} \quad (2.4)$$

Dans nos simulations, la variable *Accuracy* a été fixée à 10 mètres. La conversion de mètre en longitude se fait avec la formule d'Haversine⁴.

3. Android Developers. «Location Class». <https://developer.android.com/reference/android/location/Location>. [consulté le 8 août 2018]

4. Wikipedia, The Free Encyclopedia. «Haversine formula». https://en.wikipedia.org/wiki/Haversine_formula. [consulté le 18 avril 2018]

Vitesses. On utilise le modèle construit avec les données de la ville de Montréal⁵. Cependant, comme les données ne sont recueillies que sur des segments de routes de ville (avec feux de circulation et arrêts aux intersections), ce modèle n'est pas valide pour tous les types de routes, tels que les autoroutes ou autres voies rapides. À défaut d'avoir un modèle pour ces derniers types de routes, on utilisera la limite de vitesse permise sur ces routes comme vitesse de déplacement. Ainsi :

Soit V , la vitesse de déplacement simulée sur le segment de route R . Alors, si $R_{VitesseLimite} \geq 80$ km/h, on a que

$$V = R_{VitesseLimite} \quad (2.5)$$

Sinon,

$$V \sim \mathcal{N}\left(\frac{28,6}{50} \times R_{VitesseLimite}, \frac{9,49}{50} \times R_{VitesseLimite}\right) \quad (2.6)$$

2.2.2 Construction d'une méthode de référence

En se basant sur la description de la méthode de Li *et al.* (2016), nous avons réimplémenté le plus fidèlement possible leur méthode. Cent trajets ont ensuite été simulés dans l'état du New Jersey, USA (la région où leur donnée ont été collecté), puis estimés avec cette réimplémentation. Les résultats sont présentés dans le tableau 2.1. La différence moyenne entre les deux implémentations est de 4,0 %, les différences les plus notables étant les chemins reconstruits avec un taux d'échantillonnage de 20 secondes (6,5 % contre 11,8 %) et ceux de 300 secondes (23,6 % contre 35,4 %). Ces différences s'expliquent par le fait que des chemins

5. Ville de Montréal. «Temps de parcours sur des segments routiers (historique)». <http://donnees.ville.montreal.qc.ca/dataset/> [consulté le 17 avril 2018]

différents ont été utilisés. Cependant, à défaut d’avoir accès aux données de Li *et al.* (2016), on juge que la différence est acceptable et que notre réimplémentation est fidèle à l’originale. Comparer cette implémentation à notre méthode est donc pertinent.

Tableau 2.1: Comparaison des résultats de Li *et al.* (2016) et de notre implémentation

Taux d’échantillonnage (s)	Erreur de Li <i>et al.</i> (2016)	Erreur de notre implémentation	Différence (%)
20	0,065	0,118	44,9
30	0,129	0,132	23,0
40	0,178	0,146	-24,9
50	0,172	0,155	-10,9
60	0,180	0,161	-11,8
70	0,202	0,180	-12,2
80	0,205	0,215	4,6
90	0,202	0,198	-2,0
120	0,235	0,217	-8,3
180	0,256	0,308	16,8
240	0,258	0,299	13,7
300	0,236	0,354	33,3
Erreur moyenne	0,193	0,207	4,0

2.2.3 Comparaison de la méthode de référence et de notre méthode

Afin d’obtenir une variété intéressante de types de chemins (zones urbaines et résidentielles, autoroutes, etc.), la zone de la Rive-Sud de Montréal (ville de Brossard, Longueuil, St-Hubert et Boucherville) a été choisie. Ceci représente 37 070 noeuds

et 43 939 segments de routes, pour un total de 2 305 kilomètres de routes. Deux catégoriques catégories de chemins ont été simulées : (1) simple, soit le plus court chemin entre deux points tels que défini dans la section 2.2.1 et (2) erratique, des chemins où 20 % des segments de routes ont été enlever du réseau routier (différents pour chaque déplacement simulé). L'intérêt des chemins erratiques est de simuler des chemins moins prédictibles pour les méthodes et de représenter des situations où un utilisateur devrait éviter certains segments de routes à cause du trafic ou de travaux routiers. En tout, 200 trajets ont été évalués par catégorie de chemin. Les résultats de notre méthode à fréquence dynamique et de la réimplémentation de Li *et al.* (2016) sont présentées dans le tableau 2.2.

La figure 2.6 présente un exemple des résultats obtenus par les deux méthodes évalués, le nôtre et Li *et al.* (2016). Dans cet exemple, notre approche a effectué des lectures aux 39,1 secondes en moyenne. En conséquence, une fréquence fixe de lectures aux 40 secondes a été retenue pour l'autre méthode pour la comparaison. En regardant la portion de chemin sur l'autoroute, on constate que la méthode à intervalle fixe effectue deux fois plus de lectures GPS pour reconstruire le même chemin ($\langle B1, B2, B3, B4, B5, B6 \rangle$ vs. $\langle A1, A2, A3 \rangle$), augmentant inutilement sa consommation énergétique. Sur la portion de chemin en ville, un cas typique d'*arc-skipping* est visible dans le chemin reconstruit par la méthode de Li *et al.* (2016) entre les lectures B7 et B8 et les lectures B8 et B9. Notre approche permet de pallier ce problème en adaptant sa fréquence de lectures GPS. Clairement, on voit l'avantage d'adapter la fréquence des lectures GPS selon la configuration routière.

Notre méthode fait une lecture en moyenne aux $68,1 \pm 8,75$ secondes pour un taux d'erreur moyen de 3,65 % sur des chemins simples. Comparativement, la réimplémentation de la méthode de Li *et al.* (2016) n'arrive jamais à faire mieux qu'une erreur moyenne de 11,56 % avec une lecture GPS aux 20 secondes. Pour une fréquence similaire de lectures aux 70 secondes, l'erreur moyenne est alors de

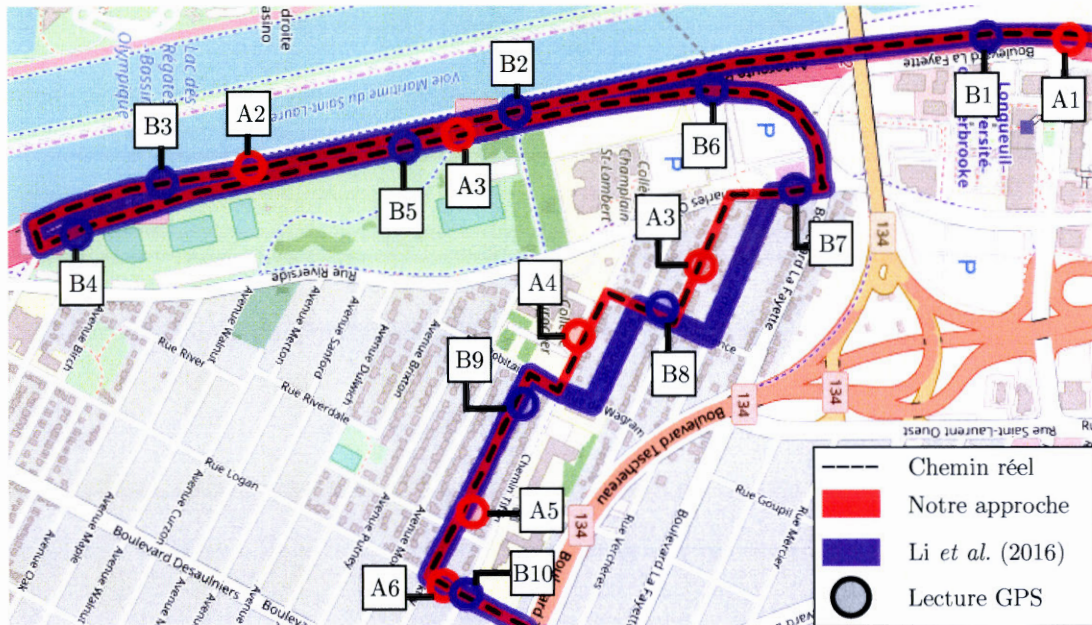


Figure 2.6: Exemple de résultats de l'approche proposée par rapport à ceux de la réimplémentation de Li *et al.* (2016)

17,35 %. Dans le cas des chemins erratiques, l'erreur argumente à 9,1 % pour une lecture aux $43,4 \pm 8,26$ secondes. Notre méthode accordant plus d'importance aux segments de route principale, il est normal que l'erreur augmente lorsqu'il y a plusieurs détours. Néanmoins, notre méthode performe toujours mieux que celle de Li *et al.* (2016) dont l'erreur la plus faible est de 29,25 % pour une fréquence de lecture aux 20 secondes. Notre approche est donc plus précise que n'importe quelle fréquence et situation que Li *et al.* (2016).

Nous comparons aussi notre méthode à Fang et Zimmermann (2011), même si la comparaison est moins intéressante. En effet, les fréquences évaluées ne dépassent jamais les 30 secondes. Il est donc difficile d'évaluer comment se comporte leur méthode avec des fréquences plus basses. Néanmoins, on peut observer qu'ils obtiennent une erreur d'environ 3,8 % pour une fréquence de lectures aux 25 secondes et de 2,5 % pour une fréquence de lectures aux 30 secondes. Leur approche obtient

Tableau 2.2: Erreurs moyennes de l'approche proposée et de Li *et al.* (2016) pour différentes fréquences GPS.

Méthode	Fréquence GPS (s)	Erreur sur les chemins simples	σ (Chemins simples)	Erreur sur les chemins erratiques	σ (Chemins erratiques)
Notre implémentation de Li <i>et al.</i> (2016)	20	0,1155	0,1564	0,2925	0,3504
	30	0,1220	0,1576	0,3079	0,3572
	40	0,1343	0,1632	0,3492	0,3636
	50	0,1412	0,1679	0,3665	0,3592
	60	0,1512	0,1611	0,3918	0,3796
	70	0,1734	0,1830	0,4220	0,3970
	80	0,1710	0,1847	0,4497	0,3981
	90	0,1821	0,1839	0,4707	0,4005
	120	0,2286	0,2075	0,5328	0,4287
	180	0,2843	0,2407	0,6550	0,4778
	240	0,3505	0,2685	0,7639	0,5119
	300	0,4010	0,2820	0,8438	0,5649
Notre méthode	68,1	0,0365	0,0633	-	-
	43,4	-	-	0,0911	0,0827

donc des résultats similaires en terme de précision, mais avec une fréquence plus de deux fois plus élevée. Il est donc raisonnable de conclure que notre approche est plus écoénergétique que la leur.

On trouvait dans la littérature que soit la fréquence de lectures GPS était relativement petite (> 60 secondes) et le taux d'erreur dépassait les 20 % (Patterson et Fitzsimmons, 2016; Li *et al.*, 2016), soit elle était grande et le taux d'erreur était acceptable (Fang et Zimmermann, 2011) à cause du problème de *arc-skipping*. Les

résultats obtenus avec notre méthode à fréquence dynamique démontrent qu'il est possible de pallier ce problème avec la simulation des déplacements possibles, c'est-à-dire d'utiliser une petite fréquence de lectures (en moyenne, puisqu'elle est dynamique avec notre méthode) et obtenir une forte précision dans les chemins estimés.

2.3 Consommation énergétique

Bien que notre méthode se montre aussi ou plus performant que les approches vues dans la littérature, nous tenons à démontrer son côté écoénergétique, puisqu'il s'agit d'un enjeu important en développement mobile. Plusieurs méthodes existent pour calculer la consommation énergétique des applications de téléphones intelligents.

2.3.1 Méthodes pour mesurer la consommation énergétique

Ampèremètre externe. Cette méthode consiste à brancher un ampèremètre entre la batterie et le téléphone intelligent. Elle est très précise et utile lorsqu'on tente de mesurer de petites différences de consommation, comme dans Hecht *et al.* (2016) où est analysée la consommation énergétique de *Internal Getter/Setter* des classes en Java. Elle peut cependant se révéler couteuse et devenir encombrante lorsqu'on doit l'utiliser lors de déplacements en voiture. C'est la méthode utilisée par Duribreux *et al.* (2014) et Paek *et al.* (2010).

Modèle énergétique. Un modèle de consommation énergétique est construit, comme PETra (Di Nucci *et al.*, 2017), et est ensuite utilisé pour évaluer les applications. Ces modèles permettent d'obtenir une estimation de la consommation énergétique selon des formules statistiques. Cependant, ils demandent souvent une calibration fine et ne sont pas toujours maintenus. Ce fut l'approche utilisée par

Nakagawa *et al.* (2013) qui a utilisé un modèle de consommation énergétique du GPS.

API Android BatteryManager. Le système d'exploitation Android possède une API, à savoir *BatteryManager*⁶, permettant d'avoir accès à des informations concernant la batterie (état, niveau de chargement, etc.) Parmi ceux-ci, on retrouve la consommation énergétique instantanée du téléphone intelligent. Le guide Android sur la mesure de la consommation énergétique nous met en garde : «*even with identical test environments, measurements are not guaranteed to be of high absolute accuracy*»⁷. Cependant, il ajoute que «*most inaccuracies specific to the fuel gauge and sense resistor are consistent between test runs, making comparisons between identical devices useful*». De plus, «*for best results, perform device power measurements during long-running A/B tests that use the same device type with the same fuel gauge and same current sense resistor*».

Puisqu'un ampèremètre est difficilement transportable en déplacement et que les modèles énergétiques sont complexes à mettre en oeuvre, nous utilisons l'API Android *BatteryManager* pour mesurer la consommation énergétique de notre méthode.

Une application a donc été développée afin de mesurer la consommation énergétique du téléphone intelligent où les valeurs retournées par l'API *BatteryManager* sont sauvegardées toutes les 10 secondes. Il faut savoir que l'API *BatteryManager* ne retourne pas une valeur instantanée, mais plutôt une valeur sauvegardée en interne qui est mise à jour périodiquement. Il s'agit d'une valeur moyenne de

6. Android Developers. «BatteryManager Class». <https://developer.android.com/reference/android/os/BatteryManager> [consulté le 27 avril 2018]

7. Android Developers. «Measuring Device». <https://source.android.com/devices/tech/power/device> [consulté le 25 avril 2018]

l'énergie consommée depuis sa dernière mise à jour. Comme le calcul de positions GPS se fait de manière courte, mais intense, en activité du processeur, il est important de s'assurer que l'API *BatteryManager* mesure bien l'énergie de ces micropics d'activité. À cette fin, nous construisons une expérimentation nous permettant de s'assurer du comportement de l'API *BatteryManager* lors de micropic d'activité du processeur. L'expérimentation se déroule sur une heure et, à chaque i -ème seconde de l'expérimentation, nous faisons calculer au processeur des divisions de points flottants pendant $\frac{i}{3600}$ secondes. Ceci permet d'obtenir une charge de calcul incrémentale répartie en micro-pic d'intensité. La figure 2.7 présente les résultats de cette expérimentation. On voit que la mesure de la consommation énergétique suit une progression relativement constante, confirmant que l'API *BatteryManager* mesure bien la consommation de micropics d'activité du processeur similaire à ceux d'un calcul de position GPS.

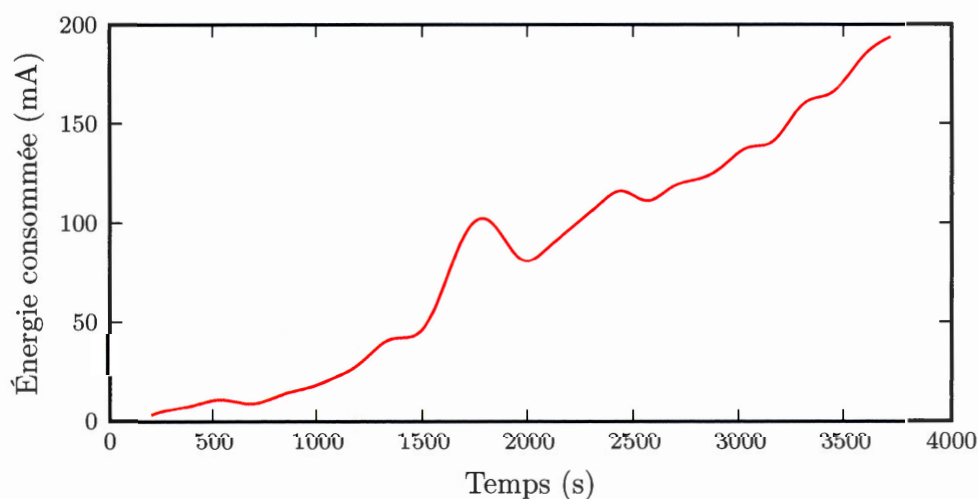


Figure 2.7: Consommation énergétique d'une charge de travail incrémentale du processeur

2.3.2 Expérience sur la consommation énergétique

La mesure de la consommation énergétique est effectuée sur des téléphones intelligents identiques de modèle ZenFone 4 Max d'Asus (modèle n° ZC554KL). Deux séries de tests sont effectuées. D'abord, nous mesurons l'énergie consommée par des lectures GPS à fréquences fixes de lectures, soit 1, 20, 40, 60, 80 et 100 secondes. Ensuite, nous mesurons l'énergie consommée par notre méthode. Avant chaque expérimentation, les téléphones intelligents sont pleinement chargés, mis en mode «avion» et toutes les applications non essentielles sont fermées manuellement. Leurs configurations sont donc identiques entre les tests afin que la différence de consommation énergétique ne soit attribuable qu'au nombre différent de lectures GPS effectuées. Ces tests ont été faits sur une période de deux mois pendant nos déplacements hebdomadaires (travail, loisirs, etc.) afin d'obtenir une variété de conditions de déplacement (trafic, type de trajet, etc.) et météorologiques (ensoleillé, nuageux, pluie, etc.). Le tableau 2.3 présente les résultats de nos tests.

Tableau 2.3: Énergie consommée selon la fréquence de lectures GPS

Taux d'échantillonnage (s)	Consommation d'énergie (mW)	Temps (heures))
1	397,44	10,60
20	231,39	10,27
40	134,87	10,41
60	104,95	10,14
80	78,89	10,55
100	65,10	10,43
Notre approche	97,48	10,33

Comme on s'y attend, plus la fréquence de lectures GPS est élevée, plus la consom-

mation énergétique est élevée. Elle passe de 397,44 mW pour une fréquence de lecture aux 1 seconde à 65,10 mW pour une fréquence de lecture aux 100 secondes. Quant à notre méthode à fréquence dynamique, elle présente une consommation moyenne de 97,48 mW. En faisant une interpolation linéaire entre la consommation d'une lecture aux 40 et aux 60 secondes, on trouve que notre méthode consomme l'équivalent d'une fréquence fixe de 65,7 secondes. C'est une réduction de 75,25 % comparativement à une méthode utilisant une fréquence de lecture aux 1 seconde et de 57,87 % comparativement à une fréquence de lectures aux 20 secondes. Ce résultat est semblable à la fréquence de lectures moyenne de 68,1 secondes trouvée à l'aide de simulations. La différence entre les trajets effectués pour la mesure de la consommation d'énergie et ceux simulés explique ce mince écart.

2.4 Conclusion

Dans ce chapitre, nous avons démontré que faire varier la fréquence GPS d'un téléphone intelligent permettait d'économiser de l'énergie en faisant moins de lectures GPS sans sacrifier la précision des chemins reconstruits. En fait, notre approche obtient même de meilleurs résultats que ceux vus dans la littérature. Ainsi, simuler les déplacements possibles pour déterminer le prochain instant quand faire une lecture GPS est une approche intéressante.

Ayant démontré qu'une approche avec une fréquence dynamique de lectures GPS peut obtenir de meilleurs résultats sur l'estimation de chemins en voiture tout en consommant moins d'énergie que les approches vues dans la littérature, nous souhaitons ensuite généraliser la méthode de sorte à inclure plusieurs moyens de transport dans le modèle de déplacement. Ainsi, seulement estimer le chemin effectué par un utilisateur ne sera pas suffisant, il faudra aussi déterminer le moyen de transport qu'il aura utilisé pour se déplacer. Une analyse des méthodes déjà

existantes pour détecter le moyen de transport d'un utilisateur à partir de son téléphone intelligent est d'abord présenté dans le chapitre 3, puis l'approche que nous proposons est ensuite présenté dans le chapitre 4.

CHAPITRE III

ÉTAT DE L'ART SUR LA DÉTECTION DU MOYEN DE TRANSPORT

Comme discuté dans l'introduction, la détection du moyen de transport dans la littérature scientifique se fait généralement avec des algorithmes d'apprentissage automatique. L'idée générale derrière ces approches est de collecter un maximum de données (position, temps, vitesse, direction, accélération, etc.) à une grande fréquence (lectures GPS aux 1 à 30 secondes) tout en enregistrant manuellement le moyen de transport courant. On laisse ensuite la tâche aux algorithmes d'apprentissage automatique de trouver des liens entre les différentes données recueillies et les moyens de transport. Ce chapitre présente en détail les différentes approches vues dans la littérature pour détecter le moyen de transport, c'est-à-dire la cueillette des données, leurs prétraitements, et le choix des algorithmes d'apprentissages et leur entraînement.

3.1 Préparation des données

3.1.1 Cueillette des données

La première étape de la détection du moyen de transport consiste à collecter les données qui serviront d'entrées aux algorithmes. Cette section présente les différents jeux de données utilisés par les études analysées. On y retrouve, lorsque décrit, les tailles d'échantillon, la durée dans le temps de la cueillette, l'endroit où

la récolte a été faite, l'appareil servant à collecté les données, les types de données collectées et la fréquence de collecte. Le tableau 3.1 présente un récapitulatif de cette sous-section.

Biljecki *et al.* (2013) ont utilisé deux jeux de données totalisant 17,5 millions de points collectés dans les Pays-Bas, constituant ainsi plus d'un demi-million de kilomètres de déplacements effectués. Selon eux, cela couvre adéquatement les différents scénarios possibles afin de tester rigoureusement la robustesse de leur approche. Le premier jeu de données a été récolté par Bohte et Maat (2009) et contient des données pour chaque jour de la semaine d'un millier de répondants collectées grâce à un GPS Logger avec une puce SiRFStarII avec un taux de lecture aux 6,5 secondes. Le deuxième jeu de données vient du projet de Van Der Spek *et al.* (2009) et contient des données concernant des piétons. L'information a été recueillie grâce à une puce SiRFStarIII avec un taux de lecture aux 5 secondes.

Bolbol *et al.* (2012) ont collecté eux-mêmes leurs données avec 81 participants dans la ville de Londres en 2010 et 2011. L'expérimentation s'est faite en deux phases afin de tenir compte de la variation saisonnière des conditions routières. Les lectures GPS ont été faites aux 60 secondes, mais le type d'appareil utilisé pour la récolte n'est pas précisé.

Byon et Liang (2014) ont quant à eux utilisé des GPS Logger et des iPhone 4 pour construire leur jeu de données dans le *Greater Toronto Area*. Durant l'été 2011, ils ont collecté un total de 100 heures de jour de semaine réparties 50/50 entre les deux types d'appareils à une fréquence de 1 Hz.

Gonzalez *et al.* (2010) ont enregistré 114 trajets à Tampa, Floride (38 en autobus, 38 à la marche et 38 en voiture) en utilisant des Motorola i870 et i580. La position GPS a été enregistrée aux 4 secondes.

Jahangiri et Rakha (2015) ont demandé à 10 employés du *Virginia Tech Transportation Institute* de transporter un Galaxy Nexus ou un Nexus 4 pour la récolte de données. Les données recueillies étaient l'accélération et de direction du téléphone. Au total, 25 heures de déplacements ont été enregistrées. Chaque employé a collecté 30 minutes de temps de trajet pour chaque moyen de transport étudié. Les données ont été récoltées à une fréquence d'environ 25 Hz, mais cette fréquence n'avait aucune garantie d'être constante. Un rééchantillonnement à une fréquence de 100 Hz a été effectué avec une interpolation linéaire.

Lari (2015) ont récolté leurs données dans la ville de Téhéran, Iran, sur une période de deux semaines en décembre 2013 grâce à une application pour téléphone intelligent. Trente-cinq étudiants ont participé à l'étude : 25 hommes et 10 femmes, âgés de 19 à 25 ans, pour un total de 245 418 lectures GPS. Les données recueillies étaient la position, la précision, l'accélération, la vitesse et la direction du téléphone intelligent.

Nitsche *et al.* (2014) ont utilisé des Samsung I-9000 Galaxy S, HTC Desire HD et HTC Desire Z pour collecter leurs données. Quinze volontaires de la région métropolitaine de Vienne, Autriche, ont ainsi collecté pendant deux mois leurs déplacements quotidiens avec à une application enregistrant leurs positions GPS à une fréquence de 1Hz. Au total, 355 heures de déplacements ont été collectées.

Reddy *et al.* (2010) ont demandé à 16 individus, âgés de 20 à 45 ans, de collecter 15 minutes de trajet pour chacun des cinq moyens de transport visés par leur approche pour construire leur jeu de données. Chaque participant portait six Nokia n95 situés sur le bras, sur la taille, la poitrine, la main, dans une poche de pantalon et dans un sac. Les téléphones cellulaires enregistraient l'accélération à une fréquence de 32Hz et la position (fréquence GPS à 1Hz fréquence Wifi à 0,33Hz et fréquence GSM à 1Hz). Au total, ils ont récolté 120 heures de données, à rai-

son de 1,25 heure de données par position de cellulaire (6) par individus (16). De plus, Reddy *et al.* (2010) ont demandé à un volontaire présent dans la première expérience d'utiliser leur système de classification tout en annotant les moyens de transport utilisés durant ses déplacements de tous les jours pendant quatre semaines afin de tester leur modèle sur la durée au quotidien.

Shin *et al.* (2015) ont utilisé 495 ensembles d'échantillons segmentés de trajets situés dans la ville de Zurich, Allemagne. De ces ensembles, 78 (correspondant à 30 utilisateurs), choisis avec une technique d'échantillonnage simple aléatoire, ont été utilisés pour une calibration fine. La collecte s'est faite avec un téléphone cellulaire et recueillait le temps, l'accélération et la position estimée par les tours cellulaires (GSM) à une fréquence de 1 Hz.

Les données utilisées par Stenneth *et al.* (2011) ont été récoltées par 6 individus, 3 hommes et 3 femmes, sur une période de 3 semaines. Trois types de téléphones cellulaires ont été utilisés (HP IPAQ PDA, Samsung Galaxy, iPhone 3G) afin d'enregistrer la position, la vitesse, l'accélération des téléphones lors des déplacements des participants toutes les 15 secondes.

Xiao *et al.* (2015a) ont collecté leurs données à partir de téléphones mobiles (Android et iOS) dans la ville de Shanghai, Chine, d'octobre 2013 à février 2014 avec la participation d'individus recrutés par internet. L'application développée enregistrait ainsi le temps, la position, la vitesse, la direction, le nombre de satellites et la précision à toutes les secondes. Au total, 102 répondants ont complété l'expérience pour un total équivalent à 535 jours de données. Des 102 répondants, 83 ont fait l'expérience pendant 5 jours et 19, pendant 6 à 8 jours. Les données collectées se décomposent en 820 segments pour la marche, 148 pour le vélo, 326 pour la voiture et 360 pour l'autobus.

Xiao *et al.* (2015b) ont collecté leurs données d'octobre 2013 à juillet 2014. La

première étape a été de demander à 30 individus connus des auteurs de participer à l'étude. La durée de l'étape n'est pas spécifiée. La deuxième étape a été de recruter plus de personnes comme décrit plus tôt (voir Xiao *et al.* (2015a)). Le nombre total de participants n'est pas spécifié, mais Xiao *et al.* (2015b) espérait atteindre les 1000 répondants d'ici la fin de l'année 2015.

Yu *et al.* (2014) ont récolté 8311 heures de transport de 2012 à 2014 à l'aide de 150 étudiants, et 74 employés et stagiaires de la compagnie HTC. La collecte s'est faite avec une application Android pour téléphone HTC One. Ils ont porté une attention particulière à ce que leur échantillon couvre suffisamment les genres (60 % hommes et 40 % femmes) et les âges possibles (20 à 63 ans).

Tableau 3.1: Techniques de cueillette de données pour la détection du moyen dans transport dans la littérature

Approche	Taille d'échantillon	Durée	Lieu	Dispositif de collecte	Données collectées	Fréquence
Biljecki <i>et al.</i> (2013)	1000 répondeurs 17 millions de points GPS	-	Pays-Bas	GPS logger avec puce SiRFStarII	Position GPS	$\sim \frac{1}{6,5}$ Hz
Bolbol <i>et al.</i> (2012)	81 participants	2 semaines	Londres, Royaume-Uni	-	Position GPS	$\frac{1}{60}$ Hz
Byon et Liang (2014)	5 personnes	100 heures de jours de semaines	Greater Toronto Area	GPS data logger, iPhone4	Position GPS	1 Hz
Gonzalez <i>et al.</i> (2010)	114 trajets	-	Tampa, Floride	Motorola i870 et i580	Position GPS	$\frac{1}{4}$ Hz
Jahangiri et Rakha (2015)	10 employés de <i>Virginia Tech Transportation Institute</i>	25 heures (30 min. / personne / moyen)	Blacksburg, Virginia	Galaxy Nexus, Nexus 4	Gyroscope, accélération	~ 25 Hz
Lari (2015)	25 hommes et 10 femmes (étudiants entre 19 et 25 ans)	2 semaines	Teheran, Iran	Téléphones équipés d'un GPS	Temps, vitesse, précision, direction, position par GPS	Inconnue
Nitsche <i>et al.</i> (2014)	15 volontaires	335 heure sur 2 mois	Vienne, Autriche	Samsung I-9000, Galaxy S, HTC Desire HD et HTC Desire Z	Vitesse, accélération, changement de direction par unité de temps par GPS	1 Hz (GPS) 100 Hz (Accéléromètre)

Reddy <i>et al.</i> (2010)	16 individus (20 à 45 ans)	15 minutes / moyen / personnes	-	Nokia n95	Accélération, position, précision, vitesse	1 Hz (GPS) 32 Hz (Accéléromètre)
Shin <i>et al.</i> (2015)	495 trajets	-	Zurich, Suisse	Nexus S, Nexus Galaxy, Samsung Galaxy S I9000 et Galaxy S2 I9100	Accélération et position estimé par GSM	1 Hz
Stenneth <i>et al.</i> (2011)	6 individus (3 hommes et 3 femmes)	3 semaines	-	HP IPAQ PDA, Samsung Galaxy, iPhone 3G	Vitesse, accélération, position, direction	$\frac{1}{15}$ Hz
Xiao <i>et al.</i> (2015a)	102 répondant 535 journées	Octobre 2013 à Février 2014	Shanghai, Chine	Android et iOS	Timestamp, position, vitesse, direction	1 Hz
Xiao <i>et al.</i> (2015b)	30 proches des chercheurs 1000 visés pour fin 2015	Octobre 2013 à juillet 2014	Shanghai, Chine	Android et iOS	Timestamp, position, vitesse, direction	1 Hz
Yu <i>et al.</i> (2014)	150 étudiants, 74 employés et stagiaires	8311 heures	-	HTC One	Accéléromètre, Magnétomètre	30 Hz

3.1.2 Prétraitement des données

Biljecki *et al.* (2013) considèrent que des données sont manquantes lorsqu'il a un espace de plus de 30 secondes entre deux lectures GPS (P_1 et P_2). Ils tentent donc d'estimer ce qui se passe entre ces deux points. Leur système regarde d'abord si P_1 et P_2 se trouvent tous les deux à proximité d'infrastructures liées au transport, comme une station de métro. Si c'est le cas, le moyen de transport correspondant à ces infrastructures est assigné au segment reliant P_1 et P_2 . Dans le cas où P_1 et P_2 seraient proches de plusieurs infrastructures différentes (arrêt de bus, station de train, etc.), la vitesse moyenne pour aller de P_1 à P_2 et le moyen de transport précédent prévalent pour le choix du moyen de transport. Sinon, leur système analyse la vitesse moyenne du segment manquant et celles des 20 prochains points du segment.

Lari (2015) définissent des limites de vitesse dures pour différents moyens de transport : si la vitesse est > 35 m/s alors le moyen de transport est *véhicule*, sinon si la vitesse est $< 5,5$ m/s, le moyen de transport est *marche*. Ceci leur permet de nettoyer les données GPS brutes récoltées en excluant les lectures ne respectant pas les règles. Ils n'est cependant pas clair comment le trafic, une situation où un véhicule peut avoir une vitesse de moins de 5,5 m/s, est pris en considération. Ce cas de figure peut induire une erreur dans la détection du moyen de transport.

Nitsche *et al.* (2014) appliquent un filtre de Kalman aux traces GPS. Ils calculent ainsi une trajectoire plus douce en combinant les positions calculées par un GPS et à partir des tours cellulaires avec un modèle de déplacement linéaire assumant une accélération suivant une loi gaussienne avec une moyenne à 0.

Reddy *et al.* (2010) implémentent une étape de filtrage de bruit. Les lectures GPS jugées «invalides» en se basant sur la précision (position, direction et vitesse), la

dilution de la précision¹ sur le temps et les variations de la vitesse du signal sont simplement ignorées. Le filtrage utilise aussi les données venant de l'accéléromètre : si trop peu de données sont reçues pour calculer une fréquence, les données sont aussi exclues de l'ensemble d'apprentissages.

Shin *et al.* (2015) reconnaissent que les individus peuvent parfois marcher dans un véhicule (autobus, train) et filtrent donc les accélérations associées à la marche dans les segments classés comme véhicule. Dans le même ordre d'idées, lorsque l'activité *stop-walking* est détectée, les données de l'accéléromètre sont ignorées jusqu'à ce que la position de l'individu soit «suffisamment» différente.

Xiao *et al.* (2015a) et Xiao *et al.* (2015b) détectent les pertes de signal par une absence de données dans la trace GPS. Si la perte de signal dure moins de 10 secondes, une interpolation linéaire est faite pour recréer les données GPS non reçues. Si la perte de signal est de plus de 10 secondes, aucune analyse n'est tentée pour cette période de temps.

Aucune mention de prétraitement des données n'est faite par Bolbol *et al.* (2012), Byon et Liang (2014), Gonzalez *et al.* (2010), Jahangiri et Rakha (2015) et Yu *et al.* (2014).

3.1.3 Segmentation des traces

Une fois les traces récoltées et traitées, elles doivent maintenant être divisées en segments plus petits. Ce sont ces segments qui seront classifiés grâce aux algorithmes de détection de monde de transport. On souhaite obtenir des segments suffisamment petits pour qu'ils ne contiennent qu'un seul moyen de transport. D'un autre côté, les segments doivent être suffisamment longs pour contenir une

1. *Geometric Dillution of Precision*

quantité d'information suffisante pour déterminer le moyen de transport. La stratégie généralement utilisée est d'évaluer les points susceptibles d'être des changements de moyen de transport afin d'effectuer la segmentation.

Biljecki *et al.* (2013) tentent de trouver les points de transitions potentiels en se basant sur les arrêts de déplacement plus long qu'un certain seuil et sur les pertes de signal induites par l'entrée dans un autobus ou une station de train. Le seuil du temps d'arrêt ou de perte de signal doit être soigneusement choisi afin d'éviter la sursegmentation ou la sous-segmentation. Biljecki *et al.* (2013) ont déterminé qu'un temps d'arrêt de 12 secondes et une perte de signal de plus de 30 secondes donnaient les meilleurs résultats. À noter que Biljecki *et al.* (2013) considèrent un arrêt comme étant un intervalle de temps où la vitesse de déplacement ne dépasse pas 2 km/h. Biljecki *et al.* (2013) reconnaissent que leur méthode ne parvient pas à détecter quelques transitions très rapides, tel qu'une personne montant dans un autobus partant immédiatement.

Bolbol *et al.* (2012) soulignent que la perte de signal GPS dû aux activités intérieures cause de longues pertes de signal sans mouvement. En conséquence, une segmentation des traces GPS est effectuée selon ces pertes de signal. La méthode précise pour détecter ces pertes de signal n'est cependant pas évoquée. Bolbol *et al.* (2012) utilisent par la suite une fenêtre mobile pour faire la détection de moyen de transport, rendant inutile un processus de segmentation des traces en segments de moyen de transport unique.

Nitsche *et al.* (2014) considèrent que les changements de moyen de transport se produisent lorsqu'un individu reste immobile pendant une certaine période ou lorsqu'il commence ou arrête de marcher. Ainsi, un nouveau segment est détecté lorsque la vitesse descend sous un seuil ou lorsque l'accéléromètre enregistre de hautes amplitudes indiquant que l'individu est peut-être en train de marcher. Si

aucun de ces événements ne se produit deux minutes après le début d'un segment, un nouveau segment est tout de même détecté. La vitesse maximale de déplacement pour détecter un nouveau segment n'est cependant pas précisée. Shin *et al.* (2015) observent que la marche peut facilement être différencié des autres moyens de transport. En effet, l'accélération moyenne de la marche est de $12,4 \text{ m/s}^2$, tandis que l'accélération des autres activités varie de $0,0$ à $0,9 \text{ m/s}^2$. En conséquence, ils basent la segmentation des trajets sur la détection de la marche. Un point de départ de marche est détecté si une accélération de 7 m/s^2 est enregistrée pendant au moins 10 secondes. Un arrêt est détecté si l'accélération moyenne sur 5 secondes est de moins de 7 m/s^2 , puis si cette condition est satisfaite pendant un autre 55 secondes, ce qui correspond au temps minimal entre deux passages d'autobus ou de train selon Shin *et al.* (2015).

Xiao *et al.* (2015a) déterminent que la fin d'un segment se caractérise par un regroupement de plusieurs points près les uns des autres, par un changement brusque de direction s'il n'y a pas de perte de signal ou par une perte de signal d'une certaine durée. Les participants à l'étude sont invités à corriger la segmentation de leurs trajets si nécessaire après les avoir enregistrées.

Yu *et al.* (2014) ont testé des tailles de fenêtres de 256, 512, 1 024 et 2 048 lectures de données pour leur segmentation. La taille de 512 a été choisie, car elle offrait le meilleur compromis entre la précision (90,76 % pour un SVM avec un polynôme de degré 3), le temps de réponse (17,1 secondes, 8,7 secondes en considérant un chevauchement de 50 %) et l'utilisation de la mémoire (4 kb).

Dans le cas de Gonzalez *et al.* (2010) et de Lari (2015), la segmentation des données est présumée et n'est pas traitée dans leurs articles. Aucune information n'est donnée sur la segmentation des trajets par Byon et Liang (2014), Jahangiri et Rakha (2015), Reddy *et al.* (2010) et Stenneth *et al.* (2011).

3.2 Détection du moyen de transport

Cette section présente en détails les différents algorithmes existants pour faire de la détection de moyen de transport. Le tableau 3.12 présente un résumé de cette section.

3.2.1 Biljecki *et al.* (2013)

Biljecki *et al.* (2013) ont décidé d'utiliser un système expert pour «leur maturité et leur capacité à résoudre d'autres problèmes» (Biljecki *et al.*, 2013, p. 10) combiné à la logique floue. La logique floue permet d'introduire le concept d'incertitude représenté par une valeur numérique $CF(h, e)$ où CF est le facteur de certitude par rapport à une conclusion selon l'hypothèse h et l'observation e . Par exemple, si une section de route à une limite de vitesse de 118 km/h, une règle selon laquelle un déplacement de 118 km/h est fait en voiture est créée : $CF_{car}^{max.speed}(118 \text{ km/h}) = 1,0$. Par contre, il est possible de voir des déplacements avec des vitesses supérieures aux limites. La possibilité que ces déplacements soient effectués en voiture ne devrait pas être ignorée, alors une règle avec une certitude plus basse est créée : $CF_{car}^{max.speed}(138 \text{ km/h}) = 0,7$.

Une fonction $f(e)$ est déterminée de manière empirique à partir du jeu de données pour chaque moyen de transport afin de déterminer CF : $CF = f(e)$. Ainsi, chaque observation doit être utilisée pour chaque moyen de transport considéré. Par exemple, en continuant l'exemple précédent, on obtiendrait en considérant la vitesse maximale des trains : $CF_{train}^{max.speed}(118 \text{ km/h}) = 0,4$. En conséquence, chaque règle du système détermine un ensemble de facteurs de certitude, une pour chaque moyen de transport :

IF (max.speed is 118 km/h) THEN

$$CF_{car}^{max.speed} = 1,0, \quad CF_{train}^{max.speed}(118 \text{ km/h}) = 0,4, \quad \dots$$

Dans le cas où plusieurs règles s'appliquent, le CF final est déterminé par un CF conjonctif :

IF (max.speed is 55 km/h)

$$\text{THEN } CF_{tram}^{max.speed} = 0,85$$

IF (average proximity to tram network is 4933 m)

$$\text{THEN } CF_{tram}^{prox} = 0$$

$$\rightarrow CF_{tram} = \min(0,85; 0) = 0$$

Il est facile de voir qu'une seule règle peut complètement éliminer la possibilité d'un moyen de transport. Ainsi, le système expert de Biljecki *et al.* (2013) fonctionne en éliminant les moyens non probables en leur assignant des CF égaux à 0 pour chaque observation qui contredirait ce moyen de transport.

De manière plus formelle, pour chaque moyen de transport m des N moyens de transport considérés, le système de classification contient k fonctions de certitude f_m^i , où k est le nombre de variables utilisées pour faire la prédiction. Pour chaque segment, on passe les k variables dans leur fonction de certitude respective pour chaque moyen de transport afin de calculer les certitudes $CF_m^i = f_m^i(i)$. Après ce calcul, le système considère le plus petit CF pour chaque moyen de transport comme CF final. Le moyen avec le plus haut CF est considéré comme le résultat final. Sous forme d'équation, cela donne :

$$\begin{array}{ccccccc}
CF_1^1 = f_1^1(i_1) & CF_1^2 = f_1^2(i_2) & \dots & CF_1^k = f_1^k(i_k) & \Rightarrow & CF_1 = \min(CF_1^1, \dots, CF_1^k) \\
CF_2^1 = f_2^1(i_1) & CF_2^2 = f_2^2(i_2) & \dots & CF_2^k = f_2^k(i_k) & \Rightarrow & CF_2 = \min(CF_2^1, \dots, CF_2^k) \\
\vdots & \vdots & & \vdots & & \\
CF_n^1 = f_n^1(i_1) & CF_n^2 = f_n^2(i_2) & \dots & CF_n^k = f_n^k(i_k) & \Rightarrow & CF_n = \min(CF_n^1, \dots, CF_n^k)
\end{array}$$

Un des avantages de ce système est d'obtenir une hiérarchie des résultats et de pouvoir proposer une classification alternative (par exemple, en donnant le deuxième meilleur moyen).

Afin de distinguer les moyens de transport semblables (voiture, autobus, tramway, etc.), Biljecki *et al.* (2013) utilisent des données GIS (*Geographic Information System*) pour améliorer les CF de ces moyens. Par exemple, si un segment est loin d'une ligne d'autobus, le système de classification rejette le moyen «autobus» en lui assignant un CF de 0. Par contre, s'il y a un trajet d'autobus dans la marge d'erreur de la lecture GPS, la classification devient ambiguë. Afin de résoudre ce problème, le système utilise la localisation d'arrêts d'autobus et de tramway, et la connaissance du précédent moyen de transport. En effet, si un segment commence près d'un arrêt d'autobus/tramway (moins de 20 mètres), il y a de fortes chances que le moyen de transport utilisé soit l'autobus/tramway. De plus, un trajet ne peut être classé comme autobus/tramway que si le moyen précédent était la marche. Les arrêts d'autobus/tramway sont généralement près d'intersection (qui sont prônés à être détectés comme de nouveaux segments), ce qui pourrait introduire une mauvaise classification sans cet ajustement.

Au final, Biljecki *et al.* (2013) obtiennent une précision de 91,7 % avec leur méthode et près de 100 % lorsque l'on tient compte du deuxième meilleur moyen de transport prédit. De plus, leur système permet de distinguer entre 10 moyens de transport différents : marche, voiture, train, métro, bicycle, tramway, autobus,

traversier, bateau et avion à l'aide des informations suivantes : Vitesse moyenne, 95e percentile de vitesse, proximité des infrastructures de train, tramway, routes, lignes d'autobus et de métro.

3.2.2 Bolbol *et al.* (2012)

Bolbol *et al.* (2012) ont construit leur cadriciel avec un *SVM*. Un des avantages des *SVM* selon eux est qu'ils peuvent être facilement entraînés directement sur les données brutes récoltées sans avoir besoin d'une phase d'extraction de caractéristiques, permettant de conserver la structure des données (par exemple, la façon dont une voiture se déplace : déplacement rapide, arrêt à un feu de circulation, puis réaccélération). Leur méthode utilise une fenêtre mobile d'une largeur de trois lectures GPS ($\{P_1, P_2, P_3\}$). Ainsi, chaque fois que la fenêtre bouge d'un point, une classification est faite.

Une fois la classification terminée, une étape de postprocessing est effectuée. En effet, comme la classification se fait à l'aide d'une fenêtre mobile, certains points GPS appartiendront à des fenêtres contigües de différents moyens de transport. On identifie donc toutes les paires de fenêtres $\{f_1, f_2\}$ adjacentes ayant des moyens de transport différents, notés a et b . Le moyen a est alors assigné à P_1 et P_2 et le moyen b à P_3 de f_1 . La même chose se produit pour f_2 , où le moyen a est assigné à P_1 et le moyen b à P_2 et P_3 (voir figure 3.1 pour un exemple visuel).

Bolbol *et al.* (2012) appliquent ensuite un processus de vérification validant la vraisemblance des transitions de moyen de transport en se basant sur les statistiques compilées par Zheng *et al.* (2008). Ils pointent que pratiquement tous les moyens de transport sont suivis par le moyen *marche*. Ainsi, leur algorithme regroupe alors les segments en différentes *étapes* séparés par la marche. Cependant, cet algorithme fait en sorte que certaines étapes peuvent contenir des segments

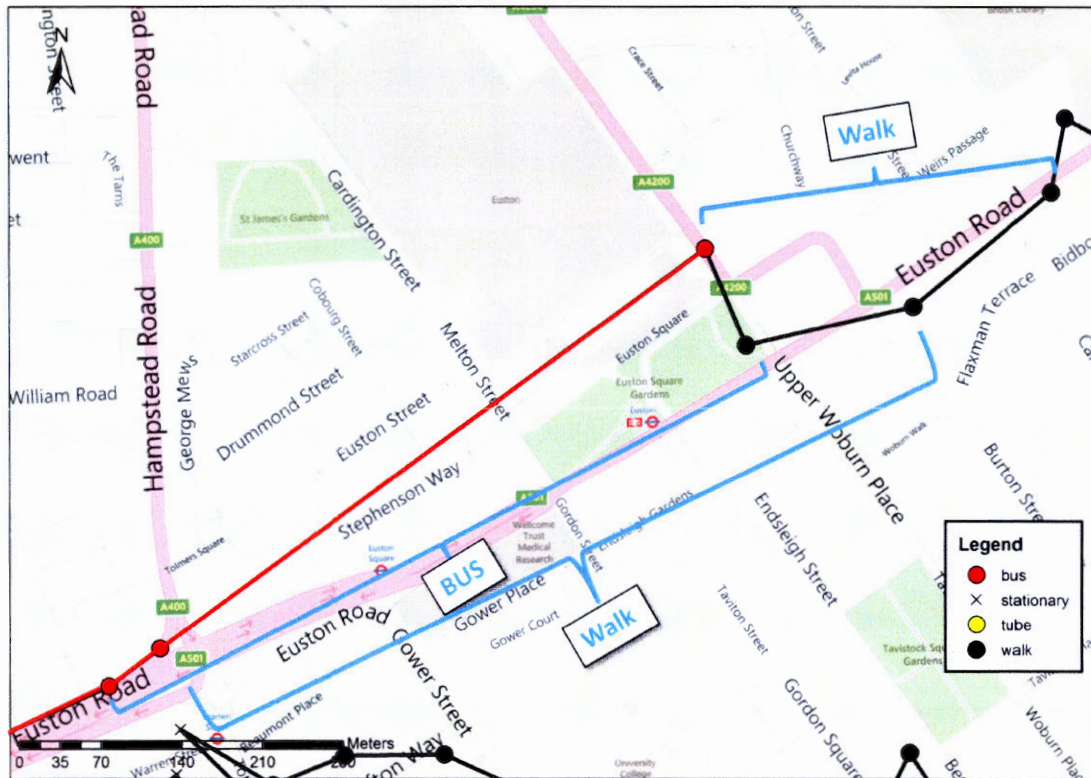


Figure 3.1: Exemple du processus de classification de Bolbol *et al.* (2012)

de différents moyens de transport. Si c'est le cas, le moyen de transport dominant (c.à.d qui représente plus de 66 % des segments) est assigné à l'ensemble des segments de cette étape.

Il faut noter qu'une attention particulière a été apportée à la détection du moyen *métro*. Ce dernier étant sous-terrain, la géolocalisation est impossible pendant ces segments. Un seuil de temps/distance est donc déterminé statistiquement pour identifier les potentiels segments *métro*. Toutes les pertes de signal qui débutent et finissent près de station de métro et qui ne dépassent pas un seuil de temps/distance sont alors détectées comme était fait en métro.

Tout le processus précédemment décrit est appliqué deux fois sur les traces GPS, une fois avec les données d'accélération et une fois avec les données de vitesses. En

comparant les résultats des deux itérations de l'algorithme, Bolbol *et al.* (2012) notent que l'accélération est plus performant pour la plupart des moyens de transport, sauf pour la *marche* et le *vélo*, atteignant une précision supérieure de 14 points de pourcentage en moyenne. Ces résultats sont utilisés pour faire l'intégration des deux itérations de l'algorithme en un seul résultat.

Au final, Bolbol *et al.* (2012) obtiennent une précision de 88 %, où les moyens *voiture*, *train* et *marche* sont très bien discriminés, tandis que les moyens *vélo*, *métro* et *autobus* auraient besoin d'amélioration selon eux. L'utilisation des données en lien avec les réseaux de transport, des autobus et des métros dans l'algorithme est suggérée.

3.2.3 Byon et Liang (2014)

L'étude effectuée par Byon et Liang (2014) vise à comparer la performance de réseau de neurones à prédire des moyens de transport utilisés selon plusieurs critères : GPS traditionnels vs GPS de téléphone intelligents, variation du taux d'échantillonnage et la variation de la densité du trafic. Deux réseaux de neurones, un pour la prédiction multimodales (voir figure 3.2) et un autre pour la prédiction voiture/non-voiture (voir figure 3.3), se composent chacun d'une couche d'entrée, d'une couche cachée et d'une couche de sortie. La prédiction se faisant segment par segment, ils prennent en entrée l'ensemble de l'information du segment analysé, c'est-à-dire la vitesse, l'accélération, le nombre de satellites et la lecture de la boussole (si la donnée provient d'un téléphone intelligent) de chaque point GPS du segment.

Lors de la détection multimodale, le réseau de neurones utilisant les données d'un GPS traditionnel obtient une précision moyenne de 75,8 %, tandis que celui

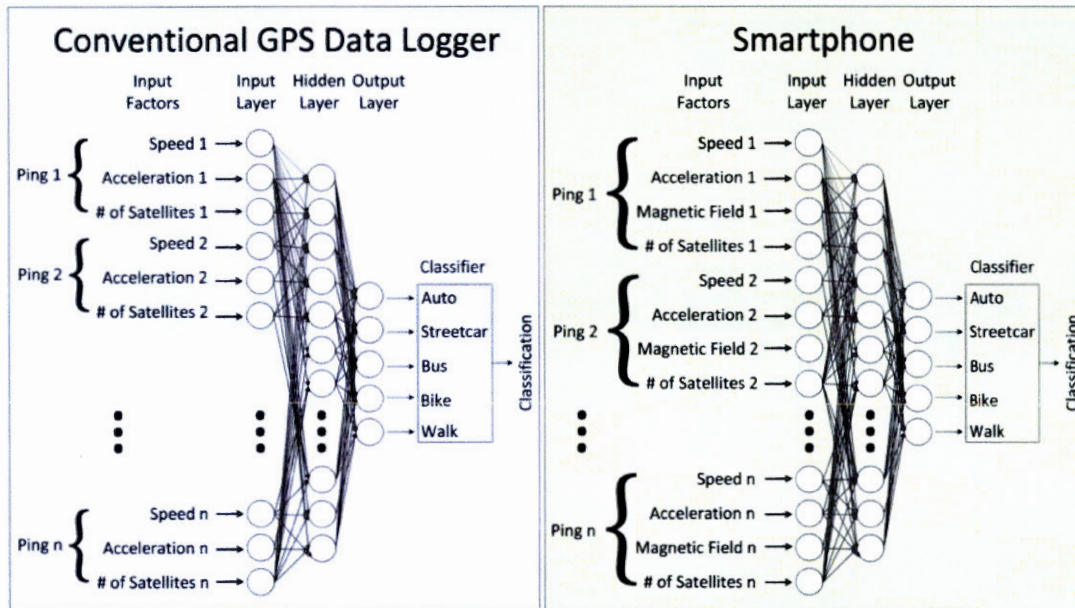


Figure 3.2: Réseau de neurones pour la détection multimodale utilisé par Byon et Liang (2014)

avec les données GPS de téléphones intelligents obtient une précision moyenne de 83,8 %.

Lors de la détection voiture vs non-voiture, Byon et Liang (2014) découvrent qu'en général la classification des segments est d'environ 10 points de pourcentage plus précis avec les données issues de téléphones intelligents qu'avec ceux de GPS traditionnels, indiquant que l'ajout de la boussole aide considérablement. C'est d'ailleurs la seule étude vue dans la littérature à prendre en considération cette valeur. Plus précisément, la détection du moyen *voiture* à une précision de 67 % avec un GPS traditionnel contre 74 % avec un téléphone intelligent lors de l'heure de pointe (16 h -19 h) contre 55 % et 64 % respectivement en dehors de l'heure de pointe (10 h -14 h). La plus haute précision lors de l'heure de pointe est due, selon Byon et Liang (2014), à l'augmentation du nombre de freinages et d'accélération brusques caractéristique en heure de pointe.

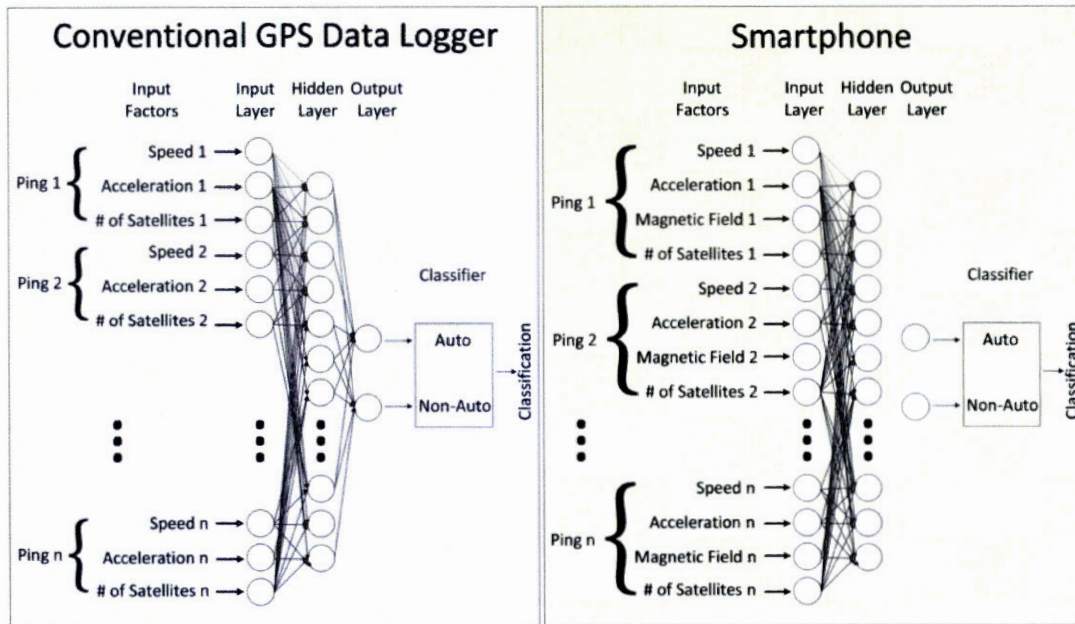


Figure 3.3: Réseau de neurones pour la détection voiture/non-voiture utilisé par Byon et Liang (2014)

En ce qui a trait à la variation de la fréquence d'échantillonnage, Byon et Liang (2014) testent plusieurs combinaisons possibles de la longueur des segments (5, 10, 15 et 20 minutes) et du nombre de lectures GPS en entrée de leur réseau de neurones (2, 5, 10, 15, 20) pour les appareils GPS traditionnels et les téléphones intelligents. Évidemment, plus le segment est long ou plus le nombre de lectures est élevé, plus la précision est bonne. La précision de la détection du moyen *voiture* passe de 53 % lors de 2 lectures sur 5 minutes à 92 % lors de 20 lectures sur 20 minutes pour les téléphones intelligents et de 54 % à 82 % pour les mêmes conditions pour les GPS traditionnels.

3.2.4 Gonzalez *et al.* (2010)

Gonzalez *et al.* (2010) utilisent un perceptron multicouche pour détecter le moyen

de transport. Ils utilisent en entrée la vitesse, l'accélération et la précision estimée. L'apprentissage est fait avec un *10-fold cross-validation*, c'est-à-dire que les lectures GPS sont divisées en 10 ensembles de taille égale et qu'un des ensembles est utilisé comme ensemble de tests tandis que les neuf autres sont utilisés comme données d'apprentissage. Ce processus est répété dix fois, chaque fois avec un ensemble de tests différent. Plusieurs paramètres sont étudiés, mais les meilleurs résultats sont atteints avec un taux d'apprentissage de 0,1 et 300 époques. Avec ces paramètres, une précision de 91,23 % est atteinte sur la détection du moyen de transport.

3.2.5 Jahangiri et Rakha (2015)

Jahangiri et Rakha (2015) étudient trois algorithmes différents : SVM, arbres décisionnels et kNN. Les SVM et les arbres de décisions sont les plus performants dans leur état de l'art, ce qui explique pourquoi ils les choisissent. Ils ajoutent l'algorithme des kNN à leur étude comme référence puisqu'il s'agit d'une méthode simple. La sélection des paramètres d'entrée est faite avec une approche du *minimum redundancy maximum relevance* (mRMR). Cette approche permet de sélectionner les variables étant les plus pertinentes tout en minimisant la redondance d'information entre elles. Ainsi, des 180 variables récoltées par Jahangiri et Rakha (2015), 80 sont conservées pour les algorithmes d'apprentissage. Ce nombre de 80 a été empiriquement choisi après avoir expérimenté différentes valeurs. Les tableaux 3.2, 3.3 et 3.4 présentent leurs résultats obtenus.

En moyenne, le kNN atteint une précision de 91,23 %, le SVM, 94,25 % et finalement la forêt d'arbres de décisions, 95,16 %.

Tableau 3.2: Matrice de confusions de Jahangiri et Rakha (2015) avec un kNN

Réel Déecté	Vélo (%)	Voiture (%)	Marche (%)	Course (%)	Bus (%)	Précision (%)
Vélo	93,31	2,33	2,30	0,99	2,96	91,72
Voiture	2,73	84,07	0,77	0,38	11,74	84,32
Marche	2,37	0,20	96,51	1,50	0,25	95,72
Course	0,07	0,00	0,15	97,00	0,05	99,71
Bus	1,51	13,40	0,27	0,13	85,01	84,79
Recall	93,31	84,07	96,51	97,00	85,01	

Tableau 3.3: Matrice de confusions de Jahangiri et Rakha (2015) avec un SVM

Réel Déecté	Vélo (%)	Voiture (%)	Marche (%)	Course (%)	Bus (%)	Précision (%)
Vélo	95,73	0,78	1,71	0,56	1,08	96,07
Voiture	0,83	91,72	0,13	0,15	7,75	91,04
Marche	1,70	0,48	97,16	1,27	0,74	95,91
Course	0,46	0,30	0,75	97,82	0,32	98,12
Bus	1,48	6,83	0,25	0,25	0,20	90,11
Rappel	95,73	91,72	97,82	97,82	90,11	

Tableau 3.4: Matrice de confusions de Jahangiri et Rakha (2015) avec une forêt d'arbres de décisions

Réel Déecté	Vélo (%)	Voiture (%)	Marche (%)	Course (%)	Bus (%)	Précision (%)
Vélo	95,47	1,46	2,73	0,97	2,29	93,06
Voiture	0,37	93,84	0,12	0,05	4,47	94,93
Marche	2,93	0,13	96,23	1,59	0,12	95,24
Course	0,03	0,00	0,40	96,81	0,00	99,55
Bus	1,19	4,57	0,73	0,58	93,12	93,02
Recall	95,47	93,84	96,23	93,81	93,12	

3.2.6 Lari (2015)

Lari (2015) utilisent un algorithme de forêt d'arbres aléatoires (Breiman, 2001). Des forêts avec 120, 140, 160 et 180 arbres ont été évaluées pour des erreurs moyennes de 4,18 %, 4,19 %, 4,16 % et 4,17 % respectivement. Naturellement, la forêt avec 160 arbres a été choisie comme meilleur algorithme. L'accélération, le changement de direction et la précision des lectures GPS ont été utilisés pour la classification. Ils n'utilisent pas le temps ni la géolocalisation, arguant que ces caractéristiques rendent les données trop uniques pour les algorithmes d'apprentissage machine.

Le tableau 3.5 présente la matrice de confusion obtenue avec la forêt de 160 arbres aléatoires.

Tableau 3.5: Résultats obtenus par Lari (2015)

Réel Détecté	Bus	Voiture	Marche	Précision (%)
	Bus	2967	367	40
Voiture	12	37176	871	97,78
Marche	3	757	7100	90,33

3.2.7 Nitsche *et al.* (2014)

Les données utilisées par Nitsche *et al.* (2014) pour la classification sont :

- le 5e percentile, le 50e percentile et le 95e percentile de l'accélération, de la vitesse, de la décélération et du changement de direction par unité de temps tel que calculé par le GPS.
- la déviation standard de la magnitude de l'accéléromètre.

- la densité spectrale de l'accéléromètre pour les fréquences $\frac{i\omega}{128}$ Hz pour $i = 1, \dots, 64$ et le fréquence de $\omega = 50$ Hz.

Les données collectés par Nitsche *et al.* (2014) sont un espace de 77 dimensions, en plus de contenir du bruit. Ils choisissent donc un schéma de classification décrit dans Ho (1998). Cette approche permet la sélection aléatoire de sous-espaces et entraîne un ensemble de classificateurs sur ces sous-espaces. Ainsi, selon une fenêtre de temps donnée, les probabilités à postériori des différents moyens de transport sont en premier lieu pondérées selon ces classificateurs. En deuxième lieu, un modèle caché de Markov discret (DHMM) combine les probabilités instantanées liées aux moyens de transport (les états cachés) aux probabilités transitions entre ceux-ci. La classification des moyens de transport correspond alors à simplement sélectionner le moyen de transport le plus probable pour chaque instant selon les données observées et les probabilités du modèle de transition. Les calculs des probabilités ne sont cependant pas détaillés. La Figure 3.4 présente un schéma de cette approche.

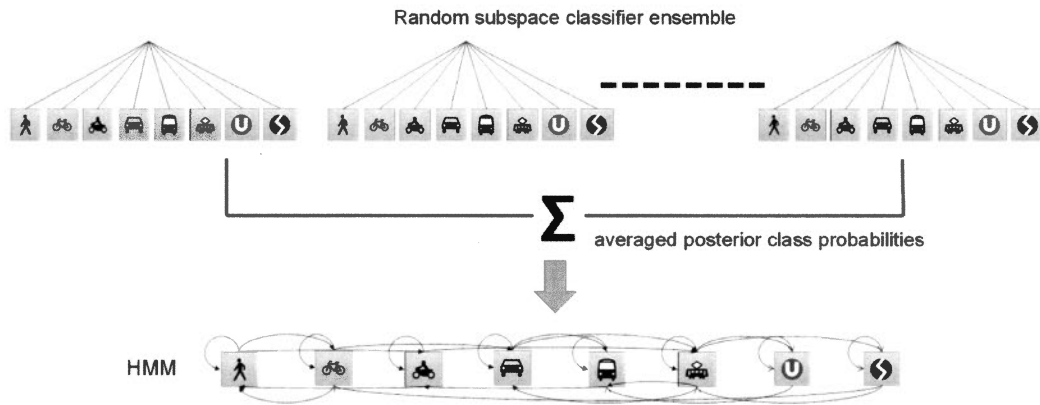


Figure 3.4: Schéma de l'approche de Nitsche *et al.* (2014)

Ils atteignent une précision variant de 65 % (train, métro) à 95 % (vélo). Le tableau 3.6 présentent ces résultats. Malheureusement, aucune matrice de confusion n'est fournie par Nitsche *et al.* (2014).

Tableau 3.6: Résultats obtenus par Nitsche *et al.* (2014)

Moyen	Marche	Vélo	Motocyclette	Voiture	Bus	Tram	Métro	Train
Précision	0,82	0,95	0,80	0,77	0,81	0,72	0,75	0,75
Rappel	0,72	0,88	0,91	0,79	0,74	0,83	0,76	0,78

3.2.8 Reddy *et al.* (2010)

Reddy *et al.* (2010) ont expérimenté six algorithmes de classification : arbre de décision (DT), méthode de Monte-Carlo cinétique (KMC), classification naïve bayésienne (NB), réseau de neurones (NN), machine à support vectoriel (SVM) et modèle de Markov caché discret (DHMM). Pour effectuer l'apprentissage et déterminer la précision de ces algorithmes, un *10-fold cross-validation* est employé, où chaque *fold* contient un nombre égal de trajets pour chaque moyen de transport. Ils trouvent que le classificateur le plus performant est l'arbre de décision avec une précision de 91,3 %. Cependant, en combinant l'arbre de décision avec un modèle de Markov caché discret, la précision atteint 93,7 %. Le tableau 3.7 présente en détail la précision obtenue avec chaque classificateur. Ainsi, leur classificateur final se compose de deux classificateurs combinés. Ce système est présenté dans la figure 3.5.

3.2.9 Shin *et al.* (2015)

Pour rappel, Shin *et al.* (2015) découpe leurs données en segments séparés par des phases de marche. Comme la marche est le seul moyen de transport a présenté une accélération supérieure à 1 m/s^2 , cette étape est très fiable (100 % de précision). Une fois ces données subdivisées, il reste à déterminer le moyen de transport pour chaque segment avec une accélération de moins de 1 m/s^2 .

Tableau 3.7: Résultats obtenus par Reddy *et al.* (2010)

Algorithmes \ Moyen	Immobile	Marche	Course	Vélo	Véhicule à moteur	Moyenne
DT	0,950	0,876	0,955	0,845	0,939	0,913
KMC	0,540	0,810	0,985	0,456	0,989	0,756
NB	0,884	0,881	0,935	0,756	0,713	0,834
NN	0,964	0,873	0,933	0,848	0,927	0,909
SVM	0,907	0,888	0,959	0,816	0,978	0,910
DHMM	0,892	0,900	0,943	0,805	0,776	0,863
DT-DHMM	0,955	0,924	0,964	0,879	0,962	0,937

Afin de détecter le moyen de transport autre que la marche, Shin *et al.* (2015) construisent un profil d'accélération pour le train/tramway, la voiture et l'autobus. La figure 3.6 montre ces profils. Ils trouvent que ces profils sont facilement linéairement séparables et fixent donc des intervalles d'accélération pour déterminer les différents moyens de transport. Plusieurs intervalles sont testés, mais ceux présentant les meilleurs résultats sont : $train/tram = [0; 0,072]$, $voiture = [0,072; 0,290]$, et $autobus = [0,290; 1,0]$. Avec cette approche, Shin *et al.* (2015) atteignent une précision moyenne de 82,05 %. Le tableau 3.8 montre la matrice de confusion quant à l'approche de Shin *et al.* (2015).

Tableau 3.8: Matrice de confusion de Shin *et al.* (2015)

Détecté \ Réel	Marche (%)	Train/Tram (%)	Voiture (%)	Bus (%)
Marche	100,00	0,00	0,00	0,00
Train/Tram	0,00	88,46	11,54	0,00
Voiture	0,00	3,85	8,77	15,38
Bus	0,00	3,85	19,23	76,92

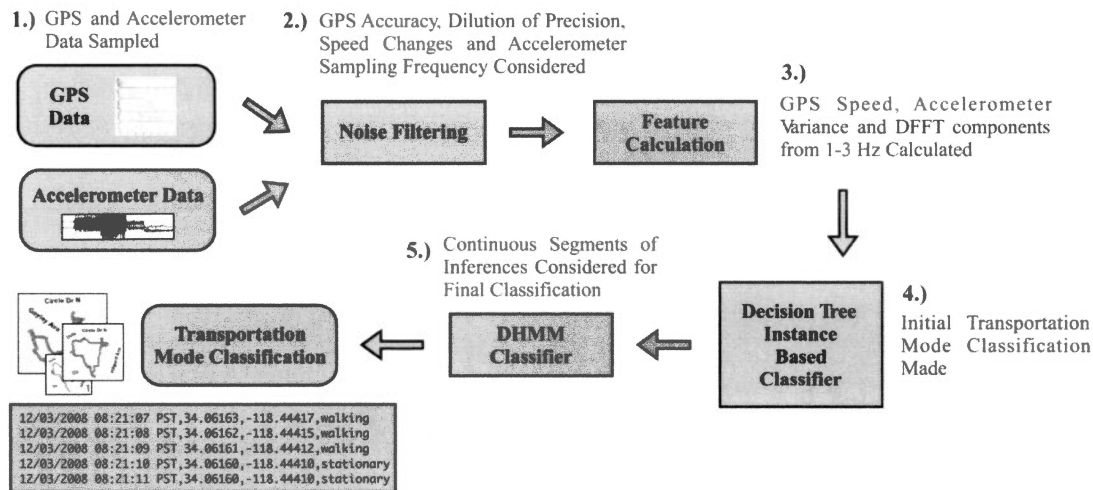


Figure 3.5: Schéma de l'approche de Reddy *et al.* (2010)

3.2.10 Stenneth *et al.* (2011)

L'approche de Stenneth *et al.* (2011) est celle se basant le plus sur les données géospatiales. Ils utilisent la position des autobus en temps réels, leur permettant ainsi de calculer la distance moyenne de l'utilisateur d'un autobus en tout temps. Aussi, les chemins de fer et les arrêts d'autobus sont intégrés dans leur modèle. Ainsi, la distance moyenne du bus ou du chemin de fer le plus proche, l'accélération moyenne, le changement de direction moyen et la vitesse moyenne sont utilisés par différents algorithmes d'apprentissage automatique pour détecter le moyen de transport de l'utilisateur. Stenneth *et al.* (2011) utilisent un classique *10-fold cross-validation* pour l'apprentissage. Les moyens de transport détectés sont le train, l'autobus, immobile, la marche, la voiture et le vélo. Les algorithmes d'apprentissage automatique évalués sont la classification naïve bayésienne, le réseau bayésien, l'arbre de décision, la forêt d'arbres de décision et le perceptron multicouche. La meilleure précision moyenne est atteinte avec la forêt d'arbres

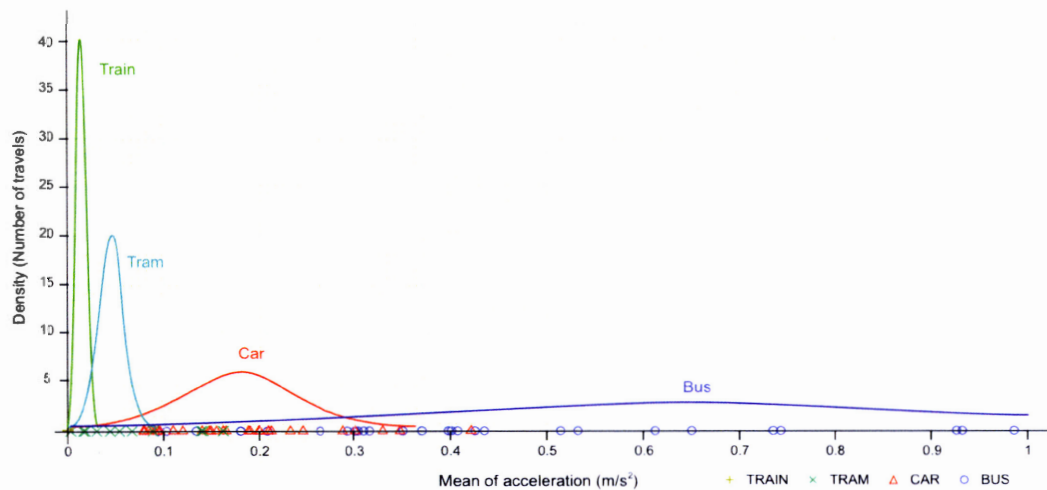


Figure 3.6: Profils d'accélération calculés par Shin *et al.* (2015) pour le train/tram, la voiture et l'autobus

de décision avec une précision de 93,7 %. Le tableau 3.9 présente les résultats de Stenneth *et al.* (2011).

3.2.11 Xiao *et al.* (2015a)

Xiao *et al.* (2015a) utilisent un réseau de neurones pour effectuer la détection du moyen de transport (voir figure 3.7). La nouveauté de leur approche est qu'ils utilisent l'optimisation par essaims de particules (Eberhart et Kennedy, 1995) pour entraîner leur réseau de neurones. Ils arguent que cette technique d'optimisation permet de réduire les chances que l'algorithme se coince dans un maximum local lors de l'apprentissage du réseau de neurones. En entrée, leur réseau de neurones prend la distance parcourue lors du déplacement, la vitesse et l'accélération moyenne, la vitesse médiane, le 95e percentile de vitesse et le taux de vitesse basse, c'est-à-dire le taux de lectures avec une vitesse de moins de 1 m/s. En moyenne, leur approche atteint une précision de 93,54 %. Le tableau 3.10 présente les résultats obtenus.

Tableau 3.9: Résultats obtenus par Stenneth *et al.* (2011)

	Classification naïve bayésienne (%)	Réseau bayésien (%)	Arbre de décision (%)	Forêt d'arbres de décision (%)	Perception multicouche (%)
Train	98,3	96,8	96,8	98,4	89,1
Bus	85,0	88,3	88,9	88,3	83,3
Immobile	100,0	100,0	100,0	100,0	96,7
Marche	96,7	94,7	94,7	96,8	86,5
Voiture	78,2	85,4	85,1	87,5	67,3
Vélo	88,9	88,7	85,5	88,92	75,0
Moyenne	91,7	92,5	92,2	93,7	83,3

Tableau 3.10: Matrice de confusion du réseau de neurones de Xiao *et al.* (2015a)

Détecté \ Réel					Précision (%)
	Marche	Vélo	Bus	Voiture	
Marche	598	2	3	2	98,84
Vélo	10	107	3	5	85,70
Bus	4	1	230	8	94,26
Voiture	3	1	8	253	95,47
Moyenne					93,54

3.2.12 Yu *et al.* (2014)

Comme données d'entrée pour leur algorithme, Yu *et al.* (2014) utilisent les données récoltées à l'aide de l'accéléromètre et du magnétomètre. Yu *et al.* (2014) comparent trois algorithmes : arbre de décision, *AdaBoost* (Freund et Schapire, 1997) et SVM. Ils concluent que c'est le SVM qui a la meilleure précision pour détecter le moyen de transport avec une précision moyenne de 90,76 %. Les deux

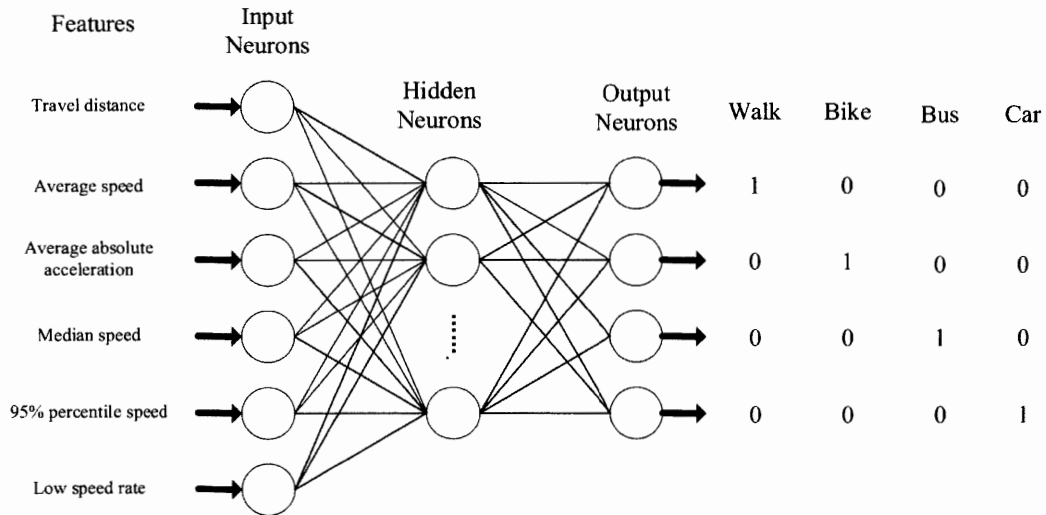


Figure 3.7: Schéma du réseau de neurones de Xiao *et al.* (2015a)

autres algorithmes n'avaient obtenu que 84,81 % (Arbre de décision) et 87,16 % (AdaBoost). Le tableau 3.11 présentent leurs résultats pour le SVM.

Tableau 3.11: Matrice de confusion du SVM de Yu *et al.* (2014)

Détecté \ Réel						Précision (%)
	Immobile	Marche	Course	Vélo	Véhicule	
Immobile	6699	22	0	2	409	93,93
Marche	466	9859	25	450	119	90,29
Course	0	99	3846	5	1	97,34
Vélo	150	537	0	5196	202	85,39
Véhicule	375	89	1	287	5837	88,59
Moyenne						90,76

Tableau 3.12: Tableau récapitulatif des articles étudiés sur la détection du moyen de transport

Approche	Meilleur algorithme	Données utilisées	Moyens de transport détectés	Meilleur précision (%)
Biljecki <i>et al.</i> (2013)	Système expert et logique floue	Vitesse moyenne, 95 percentile de vitesse, proximité des infrastructures de train, tramway, routes, lignes d'autobus et de métro	Marche, voiture, train, métro, bicycle, tramway, autobus, traversier, bateau et avion	91,7 (près de 100 % si on tient compte du deuxième moyen prédit)
Bolbol <i>et al.</i> (2012)	SVM	Vitesse et accélération	Marche, bicycle, voiture, autobus, train, métro	88
Byon et Liang (2014)	Réseau de neurones	Vitesse, accélération, nombre de satellites, boussole	Voiture, Tramway, Autobus, vélo, marche Voiture vs non-voiture	83,8
Gonzalez <i>et al.</i> (2010)	Perceptron multicouches	Vitesse, accélération et précision estimée	Marche, Voiture, Bus	91,23
Jahangiri et Rakha (2015)	kNN, SVM, Forêt d'arbres de décision,	Accéléromètre, gyroscope	Vélo, Voiture, Marche, Course, Bus	91,23 (kNN) 94,25 (SVM) 95,16 (Forêt d'arbres de décision)
Lari (2015)	Forêt d'arbres de décision	Accélération, changement de direction et précision	Marche, Voiture, Bus	95,84
Nitsche <i>et al.</i> (2014)	Modèle caché de Markov discret	Vitesse, accélération, décélération, changement de direction, densité spectrale de l'accéléromètre	Marche, vélo, motocyclette, voiture, bus, tram, métro, train	75,88

Approche	Meilleur algorithme	Données utilisées	Moyens de transport détectés	Meilleur précision (%)
Reddy <i>et al.</i> (2010)	Arbre de décision et modèle de Markov caché discret	Vitesse GPS, accéléromètre	Immobile, marche, course, vélo, véhicule à moteur	93,7
Shin <i>et al.</i> (2015)	Basé sur des règles	Accéléromètre	Marche, Train/tram, voiture, autobus	82,05
Stenneth <i>et al.</i> (2011)	Forêt d'arbres de décision	Accéléromètre, distance des autobus / arrêts d'autobus / chemins de fer	Train, autobus, immobile, marche, voiture, vélo	93,7
Xiao <i>et al.</i> (2015a)	Réseau de neurones et optimisation par essais de particules	Distance parcourue, Vitesse moyenne, accélération moyenne, vitesse médiane, 95e percentile de la vitesse	Marche, Vélo, Bus, Voiture	93,54
Yu <i>et al.</i> (2014)	SVM	Accéléromètre, magnétomètre	Immobile, marche, course, vélo, véhicule	90,76

3.3 Consommation énergétique

Bolbol *et al.* (2012) remarquent que la plupart des études utilisent des fréquences de collecte GPS de 1 Hz et se demandent si une telle granularité est nécessaire, mais ils ne creusent pas leur réflexion plus loin.

Byon et Liang (2014) reconnaissent qu'une hausse de la fréquence de récolte réduit la durée de vie de la batterie des téléphones intelligents et souhaitent donc trouver la fréquence la plus basse permettant une précision satisfaisante. Il n'y a cependant pas d'effort déployé à adapter un algorithme à cette réalité, tel que la mise en place d'une fréquence d'échantillonnage dynamique.

Gonzalez *et al.* (2010) envoient sur un serveur toutes les données collectées par les téléphones intelligents. Ils développent une méthode pour ne transférer que les données pertinentes à la détection du moyen de transport, arguant que cela économisera l'énergie de la batterie du téléphone intelligent. Aucune réflexion n'est faite quant à la consommation de lectures GPS aux 4 secondes.

Jahangiri et Rakha (2015) choisissent de ne pas utiliser le GPS, car il consomme trop d'énergie.

Lari (2015) reconnaissent que les applications utilisant le GPS sont très énergivores, mais ne poussent pas la réflexion plus loin.

Reddy *et al.* (2010) note qu'utiliser leur classificateur fait en sorte que la batterie d'un Nokia n95, le téléphone utilisé lors de leur expérimentation, a une durée de vie de 8,27 heures. Ils se justifient en rappelant que cela représente significativement moins d'énergie que les applications de divertissement présentes sur ce téléphone.

Shin *et al.* (2015) sont conscients de l'importance de proposer une solution éco-énergétique. C'est d'ailleurs pourquoi il décide de n'utiliser que l'accéléromètre à une fréquence de 1 Hz pour détecter le moyen de transport.

Stenneth *et al.* (2011) réduise la fréquence de leur capteur GPS afin d'économiser de l'énergie, mais aucune justification ni analyse n'est faite quant à la fréquence finalement choisie.

Yu *et al.* (2014) sont très conscient de la consommation d'énergie lorsqu'on tente de détecter le moyen de transport. Ainsi, toute leur approche est influencée par l'économie d'énergie. C'est d'ailleurs pourquoi il n'utilise que l'accéléromètre et le magnétomètre des téléphones intelligents. L'utilisation du gyroscope améliore leurs résultats, mais ils ont mesuré que ce dernier consomme 10 fois plus que l'accéléromètre. Une analyse poussée de la consommation d'énergie de leur algorithme est présentée. Ainsi, il compare la consommation du SVM selon son noyau. Ils comparent des noyaux polynomiaux de degré 1 à 6 et conviennent qu'un polynôme de degré 3 est celui offrant le meilleur compromis entre la consommation d'énergie et la précision de leur algorithme.

Aucune considération concernant la consommation d'énergie n'est faite par Biljecki *et al.* (2013).

3.4 Conclusion

Comme il a été présenté dans ce chapitre, les enjeux liés à la conception et l'implémentation d'algorithmes permettant la détection moyen de transport à l'aide de téléphone intelligent sont nombreux. Il faut collecter les données, les traiter et les classer tout en consommant le moins possible la batterie du téléphone intelligent. De nombreuses approches ont été analysées pour finalement constater que les algorithmes d'apprentissage automatique sont les plus utilisés pour cette

tâche, à savoir SVM, réseau de neurones, arbre de décision, forêt d'arbres de décision, perceptron multicouches, modèle caché de Markov, etc. La précision de ces algorithmes est variable, allant de 75,88 % à 93,7 %. De plus, alors que pratiquement tous les auteurs cités reconnaissent que la consommation énergétique de ces applications peuvent causer problème, seul Yu *et al.* (2014) poussent leur analyse jusqu'à mesurer la consommation d'énergie de leur approche. Il est important de noter aussi que, de toutes les approches analysées dans ce chapitre, aucune ne tente de reconstruire le chemin de l'utilisateur.

Dans le prochain chapitre, une approche permettant de détecter le moyen transport et de reconstruire le chemin emprunter par l'utilisateur sera présentée. Contrairement à ce que plusieurs auteurs ont affirmé (Jahangiri et Rakha, 2015; Shin *et al.*, 2015), il sera démontré que la puce GPS peut être utilisée pour détecter le moyen de transport tout en limitant la consommation d'énergie de l'approche, à condition que ce dernier soit utilisé de manière intelligente.

CHAPITRE IV

DÉTECTION DU MOYEN DE TRANSPORT

L'objectif de la méthode présentée est d'estimer les segments de routes et les moyens de transport utilisés par un utilisateur durant ses trajets quotidiens à l'aide de la puce GPS d'un téléphone intelligent tout en minimisant la consommation énergétique de l'algorithme. Pour arriver à cette fin, un filtre à particules est utilisé afin d'estimer l'état du téléphone intelligent selon les lectures GPS faites à une fréquence dynamique, c'est-à-dire un algorithme qui change la fréquence de lectures GPS du téléphone intelligent selon la configuration du réseau routier et de l'état qu'il estime. L'originalité de cette méthode est d'effectuer les lectures GPS à des moments stratégiques, contrairement aux autres méthodes qui ont une fréquence de lectures GPS fixes. Le schéma général de la méthode proposée va comme suit :

1. Faire une lecture GPS pour estimer l'état s du téléphone intelligent.
2. Simuler l'évolution de S jusqu'à un instant T dans le futur.
3. Déterminer le temps t donnant un compromis optimal entre la précision et la consommation d'énergie sur $[0, T[$.
4. Après que le temps t se soit écoulé dans la réalité, faire une lecture GPS et retourner à 2.

4.1 Modèle

4.1.1 Définitions et hypothèses

L'espace dans lequel évolue le téléphone intelligent est modélisée à l'aide d'un graphe $G = (V, E)$. Un noeud est défini comme $v = (Latitude, Longitude)$ et une arête comme $e = \{v_{from}, v_{to}, Speed_{Limit}, TransportModesAllowed\}$. Une arête représente une façon de se déplacer entre deux noeuds comme un segment de route, un tunnel de métro, une voie ferrée, etc. L'état du téléphone intelligent au temps t est défini par un tuple $s(t) = \{P, T, c, \bar{V}\}$ où P est la position courante du téléphone intelligent définie comme un point sur une arête $e \in E$; T , le moyen de transport courant; c , la vitesse à laquelle se déplace le téléphone intelligent et $\bar{V}$, le chemin emprunté par le téléphone intelligent jusqu'à maintenant.

L'état $s(t)$ du téléphone intelligent n'est pas directement entièrement observable, mais est estimé à partir de données obtenues par un capteur externe (GPS). À noter, le GPS ne peut estimer que les variables P et c de s , mais pas T et encore moins $\bar{V}$. Comme ce capteur n'est pas parfait et contient du bruit, $s_P(t)$, la position du téléphone intelligent à l'instant t , est modélisé par une distribution gaussienne $\mathcal{N}(x, \sigma^2)$, où x est la projection de la coordonnée obtenue par le GPS sur e , le segment de route le plus près et σ , la précision de la lecture GPS. Il est important de noter que la distribution de la position du téléphone intelligent peut déborder sur d'autres segments de routes permis selon le moyen de transport si x est près de l'extrémité de e .

L'évolution de $s_P(t + \Delta t)$ est donnée par l'équation (4.1). Puisque s_c peut varier de manière continue durant Δt (dû au trafic, à la dénivellation de la route, etc.), il existe une incertitude sur la distance réelle parcourue entre t et $t + \Delta t$ même si $s_c(t)$,

la vitesse du téléphone intelligent à l'instant t , est connue. On considère qu'une distribution gaussienne est une approximation suffisante de cette incertitude.

$$s_P(t + \Delta t) = s_P(t) + \Delta t \times \epsilon, \text{ où } \epsilon \sim \mathcal{N}(s_c(t), \sigma^2) \quad (4.1)$$

Formellement, l'algorithme proposé permet de déterminer la liste de sommets et les moyens de transport $\bar{V} = \langle \{v_0, T_0\}, \{v_1, T_1\}, \dots, \{v_{n-1}, T_{n-1}\}, \{v_n, T_n\} \rangle$ empruntés durant un déplacement, où T_i est le moyen de transport utilisé pour atteindre le sommet v_i du graphe G .

4.1.2 Filtre à particules

Puisque l'évolution de l'état s ne peut être modélisée par des fonctions paramétriques (le téléphone intelligent peut être à différentes positions après Δt , chacune avec des incertitudes, des chemins et des moyens de transport différents), un filtre à particules est utilisé pour estimer la distribution à posteriori $bel(s_t)$, où $bel(s_t)$ est l'état de croyance à l'instant t . L'algorithme général d'un filtre à particules est présenté dans l'Algorithme 2 (Thrun *et al.*, 2005). Le but d'un filtre à particules est d'approximer $bel(s_t)$ à l'aide d'un (large) ensemble de particules $\mathcal{X}_t$ de taille M généré aléatoirement à partir des données de contrôle u_t et des observations des capteurs z_t . Dans la problématique présentée, u_t est le temps écoulé entre deux lectures GPS et z_t , les données obtenues par la puce GPS.

4.1.3 Stratégie de rééchantillonnage

Un des enjeux des filtres à particules est de déterminer quand effectuer un rééchantillonnage de $\mathcal{X}_t$ (Cappe *et al.*, 2007). Après un certain temps, la distribution de $\mathcal{X}_t$ peut devenir trop grossière pour approximer adéquatement $bel(s_t)$. Ainsi, un

Algorithm 2 The particle filter algorithm

```

1: function PARTICLE FILTER( $\mathcal{X}_{t-1}, u_t, z_t$ )
2:    $\bar{\mathcal{X}} = \mathcal{X}_t = \emptyset$ 
3:   for  $m = 1$  to  $M$  do
4:     sample  $s_t^{[m]} \sim p(s_t|u_t, x_{t-1}^{[m]})$ 
5:      $w_t^{[m]} = p(z_t|x_t^{[m]})$ 
6:      $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$ 
7:   for  $m = 1$  to  $M$  do
8:     draw  $i$  with probability  $\propto w_t^{[i]}$ 
9:     add  $s_t^{[i]}$  to  $\mathcal{X}_t$ 
10:  return  $\mathcal{X}_t$ 

```

rééchantillonnage est nécessaire. Un rééchantillonnage est l'action de générer un nouvel ensemble de particules à partir de l'ensemble déjà existant (ligne 7 à 9 de l'algorithme 2).

Dans le cas présent, un rééchantillonnage signifie effectuer une nouvelle lecture GPS z_t et de consommer de l'énergie à cette fin, ce que nous cherchons à éviter. Deux méthodes sont utilisées pour déterminer quand effectuer la prochaine lecture GPS et éviter de gaspiller de l'énergie.

Premièrement, pour éviter que la distribution de $\mathcal{X}_t$ ne devienne trop grossière avec le temps, tous les états possibles du téléphone intelligent sont discrétisés avec des particules au lieu d'effectuer un échantillonnage d'un nombre M fixe de particules à partir de $p(s_t|u_t, x_{t-1}^{[m]})$ (Algorithme 2, ligne 4). Plutôt que de calculer la densité de probabilité de $bel(s_t)$ à partir de la densité des particules, un poids est associé à chaque particule (voir Figure 4.1). Quand un état s atteint le noeud v_{end} de son arête courante, il se subdivise sur toutes les arêtes sortantes de v_{end} .

Ceci assure que tous les états possibles soient adéquatement couverts par l'espace d'état.

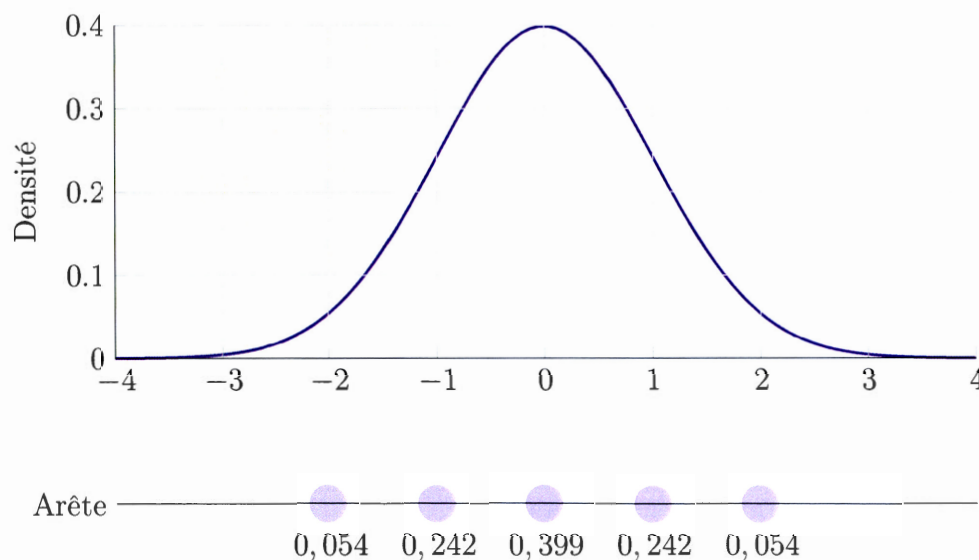


Figure 4.1: Discrétisation de $\mathcal{N}(0, 1)$ à un intervalle de σ sur l'intervalle $[-2\sigma, 2\sigma]$ avec les particules ainsi créées montrées sur une arête avec leur vraisemblance associée

Deuxièmement, pour éviter des lectures GPS inutiles et le gaspillage d'énergie, un score est calculé pour $\mathcal{X}_t$ (équation (4.2)). Ce score représente le compromis entre la consommation d'énergie et la précision moyenne (autant sur le chemin reconstruit que sur le moyen de transport) qu'obtiendrait l'algorithme si une lecture GPS était effectuée à l'instant t , où t est le temps écoulé depuis la dernière lecture GPS. L'équation (4.2) se décompose en deux parties : (1) l'équation (4.3) et (2) l'équation (4.4). La première partie évalue la précision estimée de l'algorithme selon l'espace d'état de croyance, tandis que la deuxième partie pondère cette précision selon la quantité d'énergie consommée. À noter que l'équation (4.4) n'est pas la quantité d'énergie consommée à proprement parlé, mais plutôt un score défini sur l'intervalle $[0; 1]$ représentant l'importance qu'on accorde à la consommation

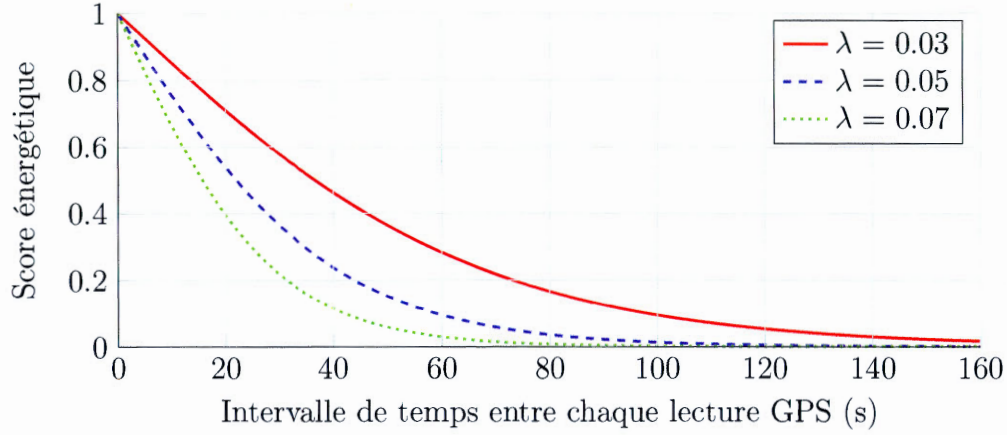


Figure 4.2: Modélisation de la consommation énergétique selon le fréquence GPS pour différents λ

d'énergie. Le paramètre λ de l'équation (4.4) permet justement d'ajuster cette importance (voir figure 4.2). Ainsi, une plus grande valeur λ met la préférence sur l'économie d'énergie, tandis qu'une valeur λ plus petite met la préférence sur des résultats plus précis au détriment de l'énergie. La Figure 4.3 montre un exemple de l'évolution de $Score(\mathcal{X}_t)$ pour différentes valeurs de λ . La prochaine lecture GPS (et le rééchantillonnage) est faite à l'instant t maximisant $Score(\mathcal{X}_t)$. Ainsi, dans l'exemple de la Figure 4.3, la prochaine lecture GPS pour $\lambda = 0,03$ serait fait à $t = 21$ secondes, alors que pour $\lambda = 0,05$ et $\lambda = 0,07$, la prochaine lecture GPS serait faite à $t = 44$ secondes.

$$Score(\mathcal{X}_t) = Précision - Score \text{ Énergétique} \quad (4.2)$$

$$Précision = \sum_{x \in \mathcal{X}_t} \left(P(x) \times x_{acc} \right) \quad (4.3)$$

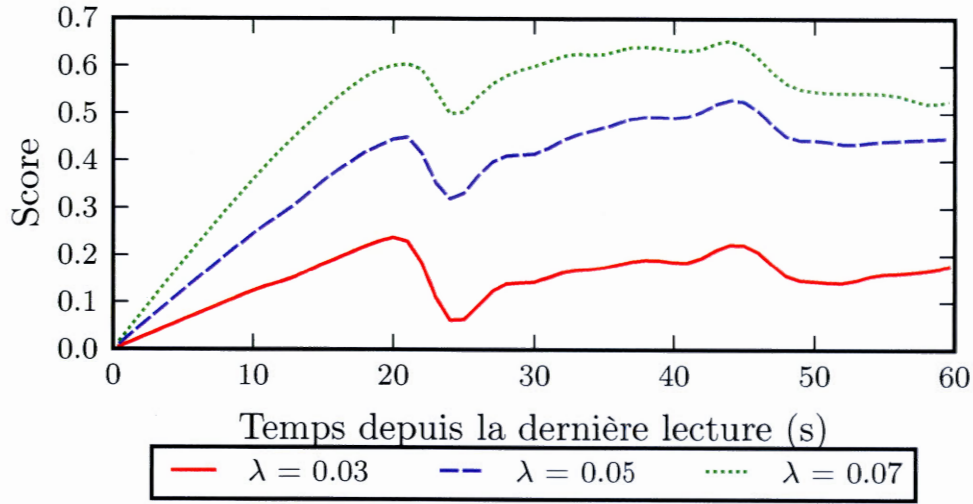


Figure 4.3: Exemple de l'évolution de $Score(\mathcal{X}_t)$ pour trois différentes valeurs de λ sur une période de 60 secondes

$$Score \text{ Énergétique} = \frac{2}{1 + e^{\lambda t}} \quad (4.4)$$

$$Score(\mathcal{X}_t) = \sum_{x \in \mathcal{X}_t} \left(P(x) \times x_{acc} \right) - \frac{2}{1 + e^{\lambda t}} \quad (4.5)$$

L'équation (4.6) définit x_{acc} , soit la précision estimée de x , où x est la particule présentement considérée, $\bar{\mathcal{X}}_t$ est l'ensemble des particules se trouvant à une distance inférieure à l'erreur de la lecture GPS de x . L'équation (4.6) représente l'erreur moyenne sur l'état estimé si x était l'état réel du téléphone intelligent et une lecture GPS était faite. Puisque les lectures GPS ont une incertitude sur la position calculée, il est possible que le véritable état du téléphone intelligent soit mélangé pour n'importe quelle particule de $\bar{\mathcal{X}}_t$, des particules ayant des positions similaires à x , mais qui peuvent avoir des chemins ou des moyens de transport qui diffèrent grandement de x . Ensuite, $|x'_{Path+}|$ est la portion de chemin que x'

possède, mais pas x et $|x'_{Path-}|$, la portion de chemin que x' n'a pas, mais que x possède.

$$x_{acc} = 1 - \frac{\sum_{x' \in \tilde{\mathcal{X}}_t} \left(P(x') \times \frac{|x'_{Path+}| + |x'_{Path-}|}{|x_{Path}|} \right)}{\sum_{x' \in \tilde{\mathcal{X}}_t} P(x')} \quad (4.6)$$

L'étape de rééchantillonnage est simple. $Score(\mathcal{X}_t)$ est calculé pour l'intervalle $[0, T[$, où T est une limite de temps après laquelle il n'y aura pas de meilleur compromis. L'instant t offrant le meilleur compromis entre consommation d'énergie et précision est retenu ($t = \max(Score(\mathcal{X}_t))$). Ensuite, une lecture GPS est faite à cet instant t . Puisque les particules sont une discrétisation de l'espace d'état du téléphone intelligent, les particules situées dans un rayon de $2 \times GPS_{Accuracy}$ sont simplement conservées au lieu de générer de nouvelles particules (Algorithme 2, Ligne 8), où $GPS_{Accuracy}$ est la rayon en mètre représentant un intervalle de confiance de 68 % sur la position GPS calculée donnée par le capteur GPS¹. Dans l'éventualité où aucune particule n'est présente dans ce rayon, ce dernier est graduellement étendu par pas de $GPS_{Accuracy}$. Le poids de chaque particule est ensuite pondéré selon leur distance respective de la lecture GPS.

4.1.4 Changement de moyen de transport

Durant un trajet, l'utilisateur peut changer de moyen de transport. Il faut tenir compte de cette possibilité dans le modèle. Ainsi, le changement de moyen de transport est permis dans deux circonstances dans le modèle : à des noeuds spéciaux et durant le rééchantillonnage.

1. Android Developers. «Android Location Class». <https://developer.android.com/reference/android/location/Location> [consulté le 8 août 2018]

1. **Noeuds spéciaux.** Certains changements de moyen de transport ne peuvent survenir qu'à des noeuds prédéfinis, tel que la *marche* vers l'*autobus* à un arrêt de bus ou le *métro* à une station de métro. En conséquence, le modèle génère des changements de moyen de transport aux noeuds correspondants à des arrêts de bus ou des stations de métro. Quand un état s atteint un de ces noeuds v , une copie s' de s avec un moyen de transport différent dicté par v est alors ajouté à l'espace d'état.
2. **Durant le rééchantillonnage.** D'autres changements de moyen de transport peuvent survenir pratiquement n'importe où, tel que la *marche* vers la *voiture* et inversement. Puisque ces changements ne surviennent en général qu'au plus une ou deux fois durant un trajet, ils ne sont considérés que lors du rééchantillonnage. Selon les résultats obtenus, l'erreur ainsi induite sur la détection du moyen de transport est négligeable comparativement à l'explosion combinatoire qui résulterait si ces changements de moyen de transport étaient considérés à chaque moment (c'est-à-dire chaque seconde) du trajet. Ainsi, après chaque rééchantillonnage, un nouvel état possible est ajouté à $\mathcal{X}_t$ pour chaque moyen de transport qui n'est pas déjà présent dans $\mathcal{X}_t$ et qui est permis par le segment de route auquel la lecture GPS a été associé. Typiquement, ceci se traduit par l'ajout d'un seul état avec la *marche* ou la *voiture* comme moyen de transport.

4.2 Expérimentations

4.2.1 Données du modèle

Pour les expérimentations, les données de la carte ont été obtenues du projet communautaire *OpenStreetMap*, tandis que les horaires et les trajets des différents transports en commun (autobus et métro) ont été obtenus directement à partir des

sites des sociétés de transport qui couvrent la région métropolitaine de Montréal (Société de Transport de Montréal, Réseau de Transport de Longueuil et Société de Transport de Laval). Les arrêts de bus et les stations de métro ont été intégrés aux données d'*OpenStreetMap* en les connectant au noeud le plus proche du segment de route le plus proche.

Pour obtenir une estimation de la vitesse de déplacement pour le moyen de transport *voiture*, les données collectées par la ville de Montréal² ont encore été utilisées. Pour rappel, une vitesse moyenne de $28,60 \pm 9,49$ km/h avait été trouvée (section 2.1.2). Pour le moyen de transport *marche*, la vitesse moyenne variait de $0,96 \pm 0,18$ m/s ($3,46 \pm 0,65$ km/h) à $1,37 \pm 0,19$ m/s ($4,93 \pm 0,68$ km/h) (Chandra et Bharti, 2013). Pour ce qui est de la vitesse de déplacement des transports en commun, elle est calculée à partir de l'horaire de ces derniers.

4.2.2 Collecte des données GPS

Durant deux mois, les déplacements quotidiens de l'auteur ont été enregistrés à l'aide d'une application Android sur un Asus Zenphone 4 Max (modèle ZC554KL). Une attention particulière a été accordée pour que tous les trajets ainsi enregistrés soient différents, que ça soit en lien avec les segments de routes empruntées ou l'heure à laquelle ils ont été effectués. Ceci permet d'obtenir une variété dans les données concernant les conditions météo, l'intensité du trafic et les horaires des transports en commun, tous des facteurs pouvant influencer les résultats de l'algorithme. Durant ces déplacements, la position géographique du téléphone intelligent a été enregistrée à une fréquence de 1 Hz par le GPS pendant que le moyen de transport courant était manuellement spécifié (*marche, voiture, autobus*

2. Ville de Montréal. «Temps de parcours sur des segments routiers (historique)». <http://donnees.ville.montreal.qc.ca/dataset/> [consulté le 17 avril 2018]

et *métro*). Un résumé des données ainsi recueillies est présenté dans le Tableau 4.1. Peu de lectures GPS ont été enregistrées pour le moyen de transport *métro*, mais cela est normal. Le signal GPS ne pénètre généralement pas suffisamment sous la surface pour être capté lorsque le téléphone intelligent se trouve dans une ligne de métro. Cependant, il arrive que les stations de métro soient suffisamment près de la surface pour permettre quelques lectures GPS. Par exemple, durant les expérimentations, des lectures GPS ont pu être effectuées à la station Berri-UQAM.

Tableau 4.1: Données GPS collectées

Moyen de transport	Temps d'utilisation (heures)	Nombre de lectures GPS
Marche	8,03	28 514
Voiture	11,94	42 768
Autobus	5,13	17 290
Métro	2,06	390
Total	27,16	88 962

4.3 Résultats

Avec les données GPS recueillies, l'algorithme a été évalué avec différents λ pour analyser son comportement selon différents compromis entre précision et énergie consommée. Évidemment, bien que les données aient été recueillies à une fréquence de 1 Hz, seules les lectures GPS qui auraient été réellement faites par l'algorithme sont utilisées. Par exemple, si au temps t l'algorithme calcule un temps optimal pour la prochaine lecture de $t + 30$ secondes, seules les lectures GPS enregistrées au temps t et $t + 30$ sont considérées et non pas les 29 autres entre ces deux dernières.

Comme attendu, un λ plus élevé signifie une fréquence GPS plus basse. Avec

$\lambda = 0,01$, une lecture GPS est faite en moyenne aux 21 secondes, tandis qu'avec $\lambda = 0,09$, les lectures sont faites aux 66 secondes en moyenne. Les résultats de l'algorithme sont présentés dans le tableau 4.1. L'analyse est divisée entre la précision de la détection du moyen de transport, la précision de la reconstruction de chemins et l'économie d'énergie atteinte avec l'algorithme.

Tableau 4.2: Précision de l'algorithme selon différents λ

λ	Fréquence moyenne (s)	Marche	Voiture	Autobus	Métro	Précision moyenne de la détection du moyen de transport	Erreur moyenne sur la reconstruction de chemins
0,01	20,9	0,996	0,988	0,936	0,932	0,963	0,023
0,03	32,5	0,920	0,982	0,871	0,902	0,919	0,028
0,05	48,6	0,908	0,969	0,806	0,890	0,893	0,031
0,07	59,2	0,876	0,934	0,772	0,885	0,867	0,039
0,09	66,0	0,859	0,932	0,758	0,876	0,856	0,041

4.3.1 Détection du moyen de transport

La précision de la détection du moyen de transport est similaire à celles trouvées dans les autres approches. Elle varie de 96,3 % avec $\lambda = 0,01$ à 85,6 % avec $\lambda = 0,09$. Les résultats pour $\lambda = 0,01$ sont même légèrement meilleurs en moyenne qu'aucune autre approche citée, la plus près utilisant le GPS ayant une précision moyenne de 94,4 % (Xiao *et al.*, 2015a). De plus, cette précision est atteinte avec une fréquence GPS moindre que celles trouvées dans l'état de l'art. Avec $\lambda = 0,01$, les lectures GPS sont faites en moyenne aux 21 seconds, tandis que Xiao *et al.* (2015a) utilisaient une fréquence de 1 Hz. Bolbol *et al.* (2012), qui avait la plus basse fréquence GPS (60 secondes), atteint une précision moyenne de 88,5 % pour la *voiture* et 58,3 % pour l'*autobus* (Bolbol *et al.*, 2012). En comparaison, l'approche présentée dans ce mémoire atteint une précision de 93,2 % pour la

voiture et 75,8 % pour l'*autobus* pour une lecture moyenne aux 66 secondes avec $\lambda = 0,09$.

Pour montrer comment l'algorithme confond parfois les moyens de transport, le tableau 4.3 présente une matrice de confusion sur la détection des moyens de transport pour $\lambda = 0,05$. La plus faible précision est pour l'*autobus*. Ce moyen de transport est faussement détecté comme étant en *voiture* 5,5 % du temps, ce qui est compréhensible puisque les autobus se déplacent sensiblement aux mêmes vitesses que les voitures. Cependant, la *voiture* n'est confondu avec l'*autobus* que 2,1 % du temps. Alors que les autobus partagent pratiquement toujours la route avec les voitures (sauf quelques rares segments de routes réservés à ceux-ci), les voitures voyagent souvent sur des segments de routes sans lignes d'autobus. Ainsi, ceci diminue la probabilité de les confondre pour des autobus. Les moyens de transport *autobus* et *métro* peuvent être confondu avec la marche lorsque l'algorithme n'arrive pas à déterminer avec précision l'arrêt de bus ou la station de métro pris par l'utilisateur. Il arrive qu'il soit équivalent en terme de temps d'arriver à un arrêt, atteindre le bus un long moment, puis le prendre que d'arriver à un arrêt, continuer au second à pied et prendre directement l'autobus à ce moment. La figure 4.4 illustre ce phénomène. Plus de lectures GPS pourraient être faites pour prévenir cette confusion, mais ceci augmenterait la consommation énergétique de l'algorithme.

4.3.2 Reconstruction de chemins

Les véritables chemins empruntés lors de la collecte de données GPS ont été retrouvés en deux temps. Un algorithme simple de reconstruction de chemins (prendre le segment de route le plus près de chaque lecture GPS) a été utilisé sur les traces, puis les chemins ainsi reconstruits ont été corrigés à la main. Ces

Tableau 4.3: Matrice de confusion pour la détection du moyen de transport avec $\lambda = 0,05$

Réel Détecté	Marche	Voiture	Autobus	Métro
Marche	0,908	0,030	0,049	0,013
Voiture	0,001	0,969	0,021	0,009
Autobus	0,048	0,055	0,806	0,091
Métro	0,048	0,018	0,044	0,890

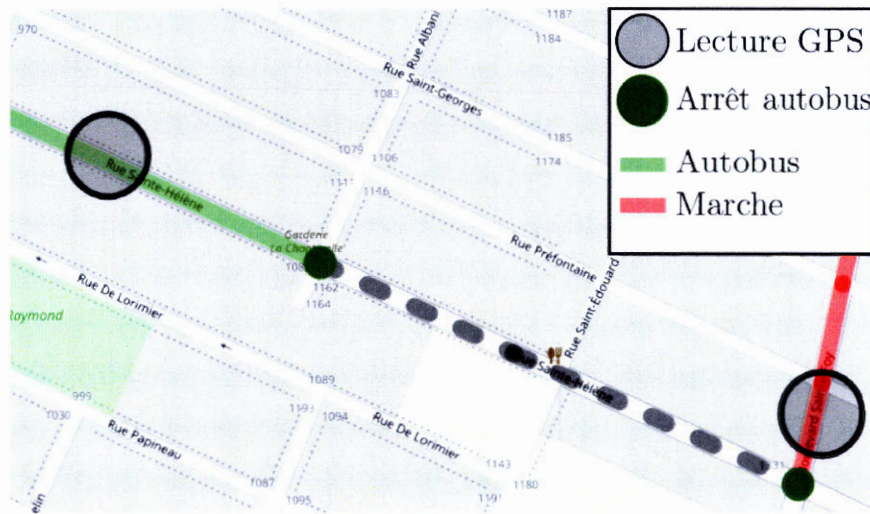


Figure 4.4: Exemple d'une possible confusion sur le véritable arrêt d'autobus pris par l'utilisateur

chemins ont ensuite été comparés à ceux trouvés par l'approche proposée. L'erreur sur les chemins reconstruits a été calculé avec l'équation (4.7) (Newson et Krumm, 2009; Li *et al.*, 2016), où P_{Real} est le chemin réel, $P_{Estimated}^+$ est la fraction de chemin estimé en plus comparé à P_{Real} et $P_{Estimated}^-$, la fraction en moins.

$$E = \frac{|P_{Estimated}^+| + |P_{Estimated}^-|}{|P_{Real}|} \quad (4.7)$$

La précision de l'approche proposée varie de 2,3 % avec $\lambda = 0,01$ à 4,1 % avec $\lambda = 0,09$ pour des lectures aux 21 secondes et 66 secondes respectivement. Pour des fréquences similaires aux approches trouvées dans la littérature, la précision de l'approche proposée est plus élevée. Cependant, comme la reconstruction de chemins a été traitée en profondeur dans le chapitre 2 (avantages d'une fréquence dynamique, sources possibles d'erreurs, etc.), il n'en sera pas discuté d'avantage ici.

4.3.3 Consommation énergétique

Pour démontrer que l'approche proposée consomme moins d'énergie, sa consommation énergétique a été mesurée sur le terrain. Pour ce faire, l'API *BatteryManager* a été utilisée³. Avant chaque test, une attention particulière a été accordée pour s'assurer que la batterie était chargée au maximum et que toutes autres applications aient été fermées. Le Wifi a aussi été désactivé, mais pas la 4G. Exécuter l'algorithme directement sur le téléphone intelligent consomme plus d'énergie que l'énergie économisée en faisant moins de lectures GPS. Ainsi, le téléphone intelligent doit communiquer à un serveur se chargeant d'effectuer les calculs. À des fins de comparaison, l'énergie consommée par des fréquences fixes de lectures GPS a été mesurée. Le tableau 2.3 présente les résultats obtenus.

Évidemment, pour des fréquences moyennes de lectures GPS équivalentes, l'approche proposée consomme plus qu'une fréquence fixe (286,28 mW contre 272,49

3. Source Android. «Measuring Device Power». <https://source.android.com/devices/tech/power/device> [consulté le 25 avril 2018]

Tableau 4.4: Consommation d'énergie selon la fréquence du GPS

Méthode	λ	Fréquence moyenne (s)	Consommation d'énergie (mW)	Précision moyenne de la détection du moyen de transport	Erreur moyenne sur la reconstruction de chemins
Fréquence GPS fixe	-	1	336,82	-	-
	-	20	272,49	-	-
	-	40	172,67	-	-
	-	60	146,74	-	-
	-	80	114,02	-	-
Fréquence GPS dynamique	0,01	22,3	286,28	0,963	0,023
	0,03	29,6	210,50	0,919	0,028
	0,05	49,1	170,79	0,893	0,031
	0,07	55,2	151,26	0,867	0,039
	0,09	72,8	148,18	0,856	0,041

mW pour une lecture aux ~ 20 secondes, par exemple), parce qu'elle nécessite d'envoyer et recevoir des requêtes HTTP sur le réseau 4G. Cependant, pour une précision plus grande ou équivalente, la consommation énergétique est moindre. La plupart des approches utilisent des fréquences de 1 Hz pour une consommation de 336,82 mW. La précision de la détection du moyen de transport associée varie de 84 % à 95 %. À titre de comparaison, l'approche proposée ne consomme que 286,28 mW pour une précision de 96,3 % ou 210,50 mW pour une précision de 91,9 %. C'est une réduction de 15,0 % en terme de consommation d'énergie pour une meilleure précision (3,3 % plus précis) et une réduction d'énergie consommée de 37,5 % pour une précision équivalente. De plus, la précision de l'approche proposée demeure acceptable avec des fréquences encore plus basses. Avec une lecture moyenne aux 72,8 secondes, elle ne consomme que 148,18 mW pour une

précision de 85,6 %, soit une diminution de 56,0 % sur l'énergie consommée pour une précision comparable à Xiao *et al.* (2015a).

CONCLUSION

Dans ce mémoire, nous nous sommes intéressés à la reconstruction de chemin et à la détection du mode de transport à l'aide du capteur GPS d'un téléphone intelligent. Un objectif important a été de trouver un compromis intéressant entre l'énergie consommée par l'approche et sa précision. L'idée principale derrière l'approche proposée est de faire les lectures GPS à une fréquence dynamique afin de ne consommer de l'énergie que lorsque pertinent, contrairement aux méthodes vues dans la littérature qui utilisent des fréquences fixes indépendamment de la pertinence et de l'information apportées par une nouvelle lecture GPS. Le projet a été réalisé en deux temps.

Premièrement, une approche simple a été proposée pour reconstruire les chemins effectués avec un seul mode de transport, la voiture. Elle se base sur la modélisation et la simulation de tous les déplacements possibles de l'utilisateur dans un réseau routier. Le calcul de la fréquence dynamique du GPS se fait selon une estimation de l'espace d'état du téléphone intelligent. L'évolution de cet espace permet de trouver un moment jugé opportun où faire une lecture GPS. Pour évaluer l'approche, des simulations ont été faites pour comparer sa précision à une implémentation d'une méthode de référence. Cela a permis de démontrer sa supériorité en terme de précision. Aussi, des expérimentations ont été faites pour montrer sa plus petite consommation d'énergie.

Deuxièmement, cette approche a été étendue afin d'inclure la marche et le transport en commun (bus et métro) dans le modèle de déplacement. La façon de déterminer la fréquence dynamique du GPS a aussi été modifiée de sorte à cal-

culer un compromis entre précision et consommation estimées. Ce compromis est d'ailleurs paramétrable. L'approche permet de favoriser la précision ou la conservation de l'énergie selon ses préférences. Des expérimentations ont été réalisées pour déterminer la précision de l'approche et sa consommation énergétique pour, encore une fois, démontrer sa supériorité comparativement à l'état de l'art. Ainsi, l'approche proposée était 3,3 % plus précise pour une économie de 15.0 % d'énergie consommée. Pour des précisions comparables, l'approche proposée atteignait alors une économie de 37.5 %.

Les approches proposées dans ce mémoire reposent fortement sur l'estimation de l'évolution de l'espace d'états de l'utilisateur au cours du temps. Un modèle a été développé, mais il pourrait être encore amélioré. Par exemple, il considère que l'utilisateur se déplace à une vitesse relativement constante et ne prend pas en compte l'élévation des routes, le trafic ou les feux de circulation. Alors que l'élévation des routes et les présences de feux de circulation sont relativement faciles à obtenir, les données reliées au trafic (tendance historique ou en temps réel) sont plus difficiles à obtenir. Des modèles existent pour estimer le trafic et son évolution dans un réseau routier (Anantharam *et al.*, 2016; Achar *et al.*, 2018; Lv *et al.*, 2018). Leur intégration améliorerait la prédiction du modèle de déplacement présenté, réduisant l'incertitude sur la position du téléphone intelligent après un certain temps et augmentant la précision de l'approche.

RÉFÉRENCES

- Achar, A., Sarangan, V., Regikumar, R. et Sivasubramaniam, A. (2018). Predicting vehicular travel times by modeling heterogeneous influences between arterial roads. Dans *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Anantharam, P., Thirunarayan, K., Marupudi, S., Sheth, A. et Banerjee, T. (2016). Understanding city traffic dynamics utilizing sensor and textual observations. Dans *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Bai, H., Hsu, D., Lee, W. S. et Ngo, V. A. (2010). Monte carlo value iteration for continuous state POMDPs. Dans *Proceedings of the International Workshop on the Algorithmic Foundation of Robotics*, 175–191.
- Biljecki, F., Ledoux, H. et van Oosterom, P. (2013). Transportation mode-based segmentation and classification of movement trajectories. Dans *Proceedings of the International Journal of Geographical Information Science*, volume 27, 1–23.
- Bohte, W. et Maat, K. (2009). Deriving and validating trip purposes and travel modes for multi-day GPS-based travel surveys : A large-scale application in the Netherlands. Dans *Transportation Research Part C : Emerging Technologies*, volume 17, 285–297.
- Bolbol, A., Cheng, T., Tsapakis, I. et Haworth, J. (2012). Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification. Dans *Computers, Environment and Urban Systems*, volume 36, 526–537.

- Breiman, L. (2001). Random forests. Dans *Machine learning*, volume 45, 5–32.
- Byon, Y.-J. et Liang, S. (2014). Real-time transportation mode detection using smartphones and artificial neural networks : Performance comparisons between smartphones and conventional global positioning system sensors. Dans *Proceedings of the Journal of Intelligent Transportation Systems*, volume 18, 264–272.
- Cappe, O., Godsill, S. et Moulines, E. (2007). An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. Dans *Proceedings of the IEEE*, volume 95, 899–924.
- Carroll, A. et Heiser, G. (2010). An analysis of power consumption in a smartphone. Dans *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, 21–21.
- Chandra, S. et Bharti, A. K. (2013). Speed distribution curves for pedestrians during walking and crossing. Dans *Procedia - Social and Behavioral Sciences*, volume 104, 660–667.
- Cheng, W., Erfani, S. M., Zhang, R. et Ramamohanarao, K. (2017). Markov dynamic subsequence ensemble for energy-efficient activity recognition. Dans *Proceedings of the EAI International Conference on Mobile and Ubiquitous Systems : Computing, Networking and Services*, 282–291.
- Chung, E.-H. et Shalaby, A. (2005). A trip reconstruction tool for GPS-based personal travel surveys. Dans *Transportation planning and technology*, volume 28, 381–401.
- Dabiri, S. et Heaslip, K. (2018). Inferring transportation modes from GPS trajectories using a convolutional neural network. Dans *Transportation Research Part C : Emerging Technologies*, volume 86, 360–371.

- Di Nucci, D., Palomba, F., Prota, A., Panichella, A., Zaidman, A. et De Lucia, A. (2017). PETrA : A software-based tool for estimating the energy profile of Android applications. Dans *Proceedings of the IEEE/ACM International Conference on Software Engineering Companion*, 3–6.
- Duribreux, J., Rouvoy, R. et Monperrus, M. (2014). An energy-efficient location provider for daily trips [Research Report]. Dans *RR-8586, INRIA*, p. 18.
- Eberhart, R. et Kennedy, J. (1995). A new optimizer using particle swarm theory. Dans *Proceedings of the International Symposium on Micro Machine and Human Science*, 39–43.
- Fang, S. et Zimmermann, R. (2011). EnAcq : Energy-efficient GPS trajectory data acquisition based on improved map matching. Dans *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 221–230.
- Freund, Y. et Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Dans *Journal of computer and system sciences*, volume 55, 119–139.
- Gonzalez, P., Weinstein, J., Barbeau, S., Labrador, M., Winters, P., Georggi, N. et Perez, R. (2010). Automating mode detection for travel behaviour analysis by using global positioning systems-enabled mobile phones and neural networks. Dans *IET Intelligent Transport Systems*, volume 4, p. 37.
- Hecht, G., Moha, N. et Rouvoy, R. (2016). An empirical study of the performance impacts of Android code smells. Dans *Proceedings of the International Conference on Mobile Software Engineering and Systems*.
- Ho, T. K. (1998). The random subspace method for constructing decision forests.

Dans *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 20, 832–844.

Jahangiri, A. et Rakha, H. A. (2015). Applying machine learning techniques to transportation mode recognition using mobile phone sensor Data. Dans *Proceedings of the IEEE Transactions on Intelligent Transportation Systems*, volume 16, 2406–2417.

Jankowska, M. M., Schipperijn, J. et Kerr, J. (2015). A framework for using GPS data in physical activity and sedentary behavior studies. Dans *Proceedings of the Exercise and Sport Sciences Reviews*, volume 43.

Lari, A. (2015). Automated transportation Mode detection using smart phone applications via machine learning : Case study mega city of Tehran. Dans *Proceedings of the Transportation Research Board Annual Meeting*, volume 6147.

Li, X., Yuan, F. et Lindqvist, J. (2016). Feasibility of duty cycling GPS receiver for trajectory-based services. Dans *Proceedings of the IEEE Annual Consumer Communications & Networking Conference*.

Lv, Z., Xu, J., Zheng, K., Yin, H., Zhao, P. et Zhou, X. (2018). LC-RNN : A deep learning model for traffic speed prediction. Dans *Proceedings of the International Joint Conference on Artificial Intelligence*, 3470–3476.

Murakami, E., P Wagner, D. et M Neumeister, D. (1997). Using global positioning systems and personal digital assistants for personal travel surveys in the united states. Dans *Proceedings of the International Conference on Transport Survey Quality and Innovation*.

Nakagawa, T., Yamada, W., Doi, C., Inamura, H., Ohta, K., Suzuki, M. et Morikawa, H. (2013). Variable interval positioning method for smartphone-based

- power-saving geofencing. Dans *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 3482–3486.
- Newson, P. et Krumm, J. (2009). Hidden markov map matching through noise and sparseness. Dans *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 336–343.
- Nitsche, P., Widhalm, P., Breuss, S., Brändle, N. et Maurer, P. (2014). Supporting large-scale travel surveys with smartphones - A practical approach. Dans *Proceedings of the Transportation Research Part C : Emerging Technologies*, volume 43, 212–221.
- Paek, J., Kim, J. et Govindan, R. (2010). Energy-efficient rate-adaptive GPS-based positioning for smartphones. Dans *Proceedings of the International Conference on Mobile Systems, Applications, and Services*, 299–314.
- Patterson, Z. et Fitzsimmons, K. (2016). DataMobile : Smartphone Travel Survey Experiment. Dans *Proceedings of the Transportation Research Record Journal of the Transportation Research Board*, volume 15, 35–43.
- Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M. et Srivastava, M. (2010). Using mobile phones to determine transportation modes. Dans *Proceedings of the ACM Transactions on Sensor Networks*, volume 6, 1–27.
- Shin, D., Aliaga, D., Tunçer, B., Arisona, S. M., Kim, S., Zünd, D. et Schmitt, G. (2015). Urban sensing : Using smartphones for transportation mode classification. Dans *Computers, Environment and Urban Systems*, volume 53, 76–86.
- Silver, D. et Veness, J. (2010). Monte-carlo planning in large POMDPs. Dans *Proceedings of the Advances in Neural Information Processing Systems*, 1–9.
- Stenneth, L., Wolfson, O., Yu, P. S. et Xu, B. (2011). Transportation mode detection using mobile phones and GIS information. Dans *Proceedings of the ACM*

SIGSPATIAL International Conference on Advances in Geographic Information Systems, p. 54.

Thompson, D. C., Rebolledo, V., Thompson, R. S., Kaufman, A. et Rivara, F. P. (1997). Bike speed measurements in a recreational population : Validity of self reported speed. Dans *Injury Prevention*, volume 3, 43–45.

Thrun, S., Burgard, W. et Fox, D. (2005). *Probabilistic Robotics*. The MIT Press.

Van Der Spek, S., Van Schaick, J., De Bois, P. et De Haan, R. (2009). Sensing human activity : GPS tracking. Dans *Sensors*, volume 9, 3033–3055.

Xiao, G., Juan, Z. et Gao, J. (2015a). Travel mode detection based on neural networks and particle swarm optimization. Dans *Information*, volume 6, 522–535.

Xiao, G., Juan, Z. et Zhang, C. (2015b). Travel mode detection based on GPS track data and Bayesian networks. Dans *Computers, Environment and Urban Systems*, volume 54, 14–22.

Yu, M.-C., Yu, T., Wang, S.-C., Lin, C.-J. et Chang, E. Y. (2014). Big data small footprint. Dans *Proceedings of the VLDB Endowment*, volume 7, 1429–1440.

Zheng, Y., Liu, L., Wang, L. et Xie, X. (2008). Learning transportation mode from raw GPS data for geographic applications on the web. Dans *Proceedings of the International Conference on World Wide Web*, p. 247.