Détection et agrégation d'anomalies dans les données issues des capteurs placés dans des smartphones

van Khang Nguyen

université PARIS-SACLAY

université PARIS-SACLAY

TELECOM SudParis

NNT : 2019SACLL021

Thèse de doctorat

Détection et agrégation d'anomalies dans les données issues des capteurs placés dans des smartphones

Thèse de doctorat de l'Université Paris-Saclay
préparée à Télécom SudParis

Ecole doctorale n°580 Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Réseaux, information et communications

Thèse présentée et soutenue à Évry, le 17 décembre 2019, par

VAN KHANG NGUYEN

Composition du Jury :

Véronique VÈQUE Professeur, Université Paris-Sud	Présidente
Hacène FOUCHAL Professeur, Université de Reims Champagne-Ardenne	Rapporteur
Selma BOUMERDASSI MdC HDR, CNAM	Rapporteur
Thi Mai Trang NGUYEN MdC HDR, Université Pierre et Marie Curie	Examinatrice
Éric RENAULT Professeur, ESIEE	Directeur de thèse
Viet Hai HA MdC, Université de Hué, Vietnam	Co-encadrant de thèse

Remerciements

Je tiens à remercier en premier lieu mon directeur de thèse, Professeur Éric Renault de l'ESIEE Paris, pour sa disponibilité, ses qualités pédagogiques et scientifiques, son enthousiasme et sa sympathie. Son orientation, ses précieux conseils et ses nombreux encouragements m'ont effectivement aidé à conduire cette thèse à la fin. Je lui adresse ma gratitude pour tout cela.

J'adresse de chaleureux remerciements à mon co-encadrant de thèse, M. Viet Hai Ha, Maître de conférences de l'Université de Hué, Vietnam, pour son attention de tout instant sur mes travaux, pour ses conseils avisés. Son énergie, sa confiance et ses encouragements ont été des éléments moteurs pour moi.

Je voudrais remercier Mme Selma Boumerdassi, Maître de conférences HDR au CNAM et M. Hacène Fouchal, Professeur à l'Université de Reims Champagne-Ardenne pour l'honneur qu'ils m'ont fait de participer à mon jury de thèse en qualité de rapporteur, pour le temps consacré à la lecture de cette thèse, et pour les suggestions et les remarques judicieuses qu'ils m'ont indiquées.

J'associe à ces remerciements Mme Véronique Vèque, Professeur à l'Université Paris-Sud et Mme Thi Mai Trang Nguyen, Maître de conférences HDR à l'Université Pierre et Marie Curie, pour l'honneur qu'ils m'ont fait en acceptant d'examiner mon travail.

J'adresse de sincères remerciements au Professeur Alain Sibille, Télécom ParisTech, Directeur adjoint de l'ED STIC, responsable du Pôle 2 et les membres de l'école doctorales pour m'avoir accepté au sein de l'école et pour leur assistance apporté tout au long de ma thèse.

Je suis très honoré de remercier Mme Pascale Minet du Centre de Recherche Inria de Paris et Mme Joanna Tomasik, Professeur à CentraleSupélec pour leur participation à mon jury mi-parcours et pour toutes les suggestions et les remarques judicieuses qu'elle m'ont faites.

Je désire grandement remercier Professeur Ruben Milocco de l'Université de Comahue, Argentine, pour les suggestions intéressantes. J'ai pris un grand plaisir à travailler avec lui pour une partie importante de cette thèse.

Je désire remercier tous les membres de Télécom Sudparis pour leur enthousiasme, leur sympathie, leur amitié. J'ai eu beaucoup de plaisir à travailler avec eux.

Je tiens à remercier mes amis, mes collaborateurs pour leur soutien, leur encouragement et leur aide tout le long de ma thèse.

Enfin, je remercie les membres dans ma familles et de ma belle-famille pour leur soutien, tout au long de ces années, sans lesquels je ne serais pas là aujourd'hui. Un merci spécial à ma chère épouse pour avoir assumé la lourde charge de famille afin que je puisse réaliser mon rêve.

Résumé

Aujourd'hui, les *smartphones* sont devenus extrêmement populaires et équipés de nombreux capteurs tels qu'une caméra, un microphone, un capteur GPS, un accéléromètre, un magnétomètre, etc. Les utilisateurs de *smartphones* se connectent de plus en plus souvent à l'Internet. Par conséquent, il existe de nombreuses recherches sur les systèmes de capteurs à base de *smartphones* dans de nombreux domaines d'application.

Nos recherches visent à construire un système de capteurs à base de *smartphones* pour détecter les anomalies environnementales. Nous avons développé un tel système et améliorons les composants principaux : méthode de détection des anomalies par le *smartphone* et méthode de synthèse des données issues des anomalies au centre de calcul.

Pour les méthodes de détection des anomalies, nous nous appuyons sur les recherches actuelles de détection des anomalies routières. En fait, une méthode efficace, économe en énergie et répondant bien à différents types d'environnements et périphériques est nécessaire. Par conséquent, nous avons proposé une méthode de détection des anomalies basée sur le test des valeurs aberrantes en statistique. Les algorithmes améliorés ont une faible complexité et utilisent moins de mémoire. Nos expériences sur des données réelles montrent que nos méthodes s'adaptent bien aux conditions différentes dans le cas de la détection d'anomalies routières.

L'agrégation des données des anomalies n'a pas fait l'objet de beaucoup de travaux. Les recherches actuelles ne s'arrêtent qu'à des méthodes de regroupement simples basées sur la distance. Nous avons donc construit une méthode d'intégration des données permettant de trouver plus précisément l'emplacement de ces anomalies. Nous avons développé deux algorithmes. Le premier réalise une mise en cluster simple et fonctionne de manière cumulative. Cet algorithme décompose les données afin d'accroître la vitesse et la précision pour la localisation des anomalies. Le second permet de trouver des anomalies en se basant sur la recherche des modes de densité de probabilité. Nous avons également proposé un moyen d'attribuer une pondération aux emplacements trouvés pour évaluer la fiabilité. Notre expérience de simulation a confirmé l'efficacité de cet algorithme.

Table des matières

Table des figures

Liste des tableaux

Abréviations

WSN	Wireless Sensors Network
SCS	Système de capteurs de smartphones
EDI	Environnement de développement
MCS	Mobile Crowdsensing System
P^2	Pothole Patrol
SVM	Support Vector Machines
WPD	Wavelet Packet Decomposition
PCA	Principal component analysis
RoSDS	Road pavement Anomaly Detection System
FFT	Fast Fourier Transform
SWT	Stationary Wavelet Transform
ROC	Receiver Operating Characteristic
BSSDF	Binary Semi-structured Data Format
EM	Espérance-maximisation
KDE	Kernel Density Estimator

1 Introduction

1.1 Contexte

Un réseau de capteurs sans fil (*Wireless Sensors Network* – WSN) est composé de capteurs géographiquement dispersés qui fonctionnent ensemble pour surveiller des conditions physiques ou environnementales. Aujourd'hui, le WSN est identifié comme l'une des technologies les plus importantes et les plus répandues. Le WSN est déployé dans de nombreux domaines tels que [55] : la surveillance de l'environnement, les applications militaires, le suivi des véhicules, les bâtiments intelligents, les soins de santé, le suivi des animaux, etc.

Les *smartphones* sont de plus en plus populaires et puissants. Actuellement, environ 3,3 milliards de personnes dans le monde utilisent des *smartphones*, et ce nombre devrait atteindre environ 3,8 milliards en 2021 [62]. La puissance des *smartphones* est également améliorée avec des puces multicœurs et plusieurs gigaoctets de mémoire. De plus, les *smartphones* aujourd'hui contiennent de nombreux capteurs différents tels qu'une caméra, un microphone, un capteur GPS, un accéléromètre, un capteur de proximité, un capteur de lumière ambiante, un magnétomètre, un baromètre, un capteur d'humidité de l'air, un thermomètre, etc. À côté de l'augmentation de la capacité des *smartphones*, les coûts d'accès au réseau sont de plus en plus bas. Le WiFi gratuit est également disponible dans de nombreux endroits permettant aux utilisateurs de téléphones d'être facilement connectés à l'Internet. Toutes les conditions ci-dessus ouvrent une grande opportunité de profiter des *smartphones* pour construire des systèmes de capteurs.

Les principaux avantages d'un système de capteurs à base de *smartphones* (SCS) par rapport aux WSN sont :

- *le coût extrêmement bas des équipements.* Tirer parti des capteurs sur les *smartphones* conduit à des SCS ne nécessitant pas de coût pour les capteurs. Les coûts requis pour un SCS sont principalement le coût des logiciels, des équipements centraux et des équipements supplémentaires, le cas échéant ;
- *les smartphones possèdent une grande puissance de calcul.* Chaque *smartphone* est un ordinateur doté de capacités informatiques parallèles pouvant aider à partiellement traiter des données capturées. Cela réduit la quantité de données échangées et le traitement au niveau des serveurs centraux ;
- *la programmation pour smartphones est pratique, les outils de développement sont gratuits et puissants.* Pour les *smartphones* Android, il est possible de programmer en Java sur l'environnement de développement (EDI) Eclipse ou Android Studio [17]. Les applications pour iPhones peuvent être programmées avec Objective-C ou Swiff sur l'EDI XCode [46]. Les langages de programmation java, Objective-C et Swiff supportent la programmation orientée objet, la documentation est complète et disponible en ligne [19, 53, 64].
- *les smartphones peuvent se connecter à l'Internet.* Les *smartphones* peuvent non seulement se connecter les uns aux autres, comme des réseaux ad-hoc, mais également se connecter directement à l'Internet . De plus, avec le faible coût de l'accès à l'Internet et la large couverture WiFi, les utilisateurs de téléphones sont très souvent connectés à l'Internet. La connexion Internet permet aux données des *smartphones* d'être envoyées directement au centre. Par conséquent, le modèle de transfert des données devient plus simple et l'architecture de réseau du SCS devient également plus simple qu'un WSN traditionnel ;
- *le nombre de smartphones est très important.* Des milliards de *smartphones* dans le monde sont potentiellement exploitables. De nombreuses recherches ont été menées sur la construction de systèmes de capteurs avec la participation de nombreuses personnes utilisant un *smartphone.* Les résultats de l'analyse des données de l'ensemble du système apporteront de nombreux avantages que les participants peuvent exploiter. Un tel système s'appelle le système *crowdsensing* [23].

Cependant, l' utilisation de *smartphones* comme système de capteurs présente également de nombreuses difficultés :

- *les données capturées ne sont pas complètes ni uniformes.* La collection des données dépend de l'emplacement du participant. Les participants peuvent démarrer ou arrêter le programme de collecte de données sur leur téléphone en fonction de la situation. Par conséquent, les échantillons prélevés sur les téléphones mobiles sont généralement distribués de manière aléatoire dans l'espace et dans le temps et sont incomplets. En outre, la qualité des *smartphones* est très variable. Cela se traduit par une précision des données différente. L'inégalité dans la qualité et la distribution des données constitue également un défi majeur, qui nécessite des

méthodes d'analyse adaptatives basées sur la réalité de la qualité et de la densité des données. Par exemple avec un système de détection de nids-de-poule. Il y a plus de rapports de nid-de-poule autour du point A que du point B. Cependant, la certitude de l'existence d'un nid-de-poule au point A peut être inférieure à celle du point B, car elle dépend également de la qualité des rapports et de la densité de la détection à chaque point ;

- *il est difficile d'évaluer la fiabilité des données.* Les participants peuvent positionner leurs dispositifs par inadvertance de manière à enregistrer des mesures incorrectes. Par exemple, un participant stocke le téléphone dans son sac acquérant des informations sur le bruit urbain. En cas de détection d'anomalies routières (nid-de-poule, dos-d'âne, etc), le fait de laisser le téléphone à n'importe quelle orientation rend également difficulté pour la détection. En outre, les participants peuvent involontairement ajouter de fausses informations. Par exemple, faire un cri en participant à la collecte d'informations sur le bruit urbain, ou faire basculer le téléphone en cas de détection des anomalies routières. Cette difficulté nécessite une analyse fiable et la suppression des fausses données des participants. C'est aussi un gros défi ;
- *il est nécessaire de réserver des ressources du smartphone pour le participant.* L'utilisation principale des téléphones mobiles doit être réservée aux activités habituelles des participants, telles que passer des appels, accéder à l'Internet, écouter de la musique, jouer, etc. Les participants se porteront volontaires pour fournir des données si ce processus n'alourdit pas le téléphone en prenant beaucoup de ressources en CPU et en mémoire. En particulier ce processus ne consomme pas beaucoup de batterie et de réseau. Par conséquent, le mécanisme de collecte de données doit être construit de manière à économiser de l'énergie. Les algorithmes d'analyse des données dans les *smartphones* doivent également être peu complexes et utiliser le moins de mémoire possible ;
- *il est nécessaire de préserver la confidentialité des utilisateurs.* Sans aucun mécanisme de protection approprié, les téléphones portables sont transformés en espions miniatures, révélant éventuellement des informations privées sur leur propriétaire. De nombreux utilisateurs sont conscients des conséquences possibles et peuvent donc être réticents à contribuer aux campagnes de détection. La réticence des utilisateurs à contribuer diminuerait l'impact et la pertinence des campagnes de détection déployées à grande échelle.

Les avantages du SCS attirent les chercheurs dans ce domaine. L'application de SCS est diversifiée dans de nombreux domaines tels que l'environnement, la société et les soins de santé (voir la Sec. 2.1).

Cependant, il reste encore de nombreux défis pour tirer parti des énormes ressources qui peuvent capturées par des *smartphones* . Les systèmes SCS d'aujourd'hui ne sont pas encore complètement développés, ils n'ont donc pas été largement appliqués dans

la vie. Pendant ce temps, le nombre de *smartphones* augmente, la capacité des *smartphones* s'améliore constamment, les capteurs des *smartphones* sont de plus en plus diversifiés, la connexion à l'Internet à partir de *smartphones* est également de plus en plus fréquente. Afin de profiter de cette énorme ressource, il est nécessaire d'effectuer des recherches pour améliorer le système, de la méthode de détection à l'échange de données, de l'économie de la consommation des resources, du *smartphone* au mécanisme approprié d'agrégation et d'analyse des données suitabes aux caractéristiques d'un SCS.

Dans ce contexte, nos recherches visent à contribuer à l'amélioration des méthodes de détection et à l'agrégation des données sur le SCS. Dans le cadre de cette thèse, nous étudions le SCS selon le type de crowensensing appliqué à la détection d'anomalies environnementales et qui peut s'appliquer à la détection des anomalies routières, la détection des accidents, la détection des sons anormaux, etc.

1.2 Contribution de la thèse

Nos recherches sont axées sur l'amélioration du SCS afin de détecter les anomalies dans l'environnement. La thèse contribue à améliorer certaines étapes du système.

1) Au niveau de l'architecture du système, nous avons analysé plusieurs modèles de types SCS pour construire un modèle de base du système SCS afin de détecter les anomalies. Notre modèle permet de convertir un certain nombre de composants du téléphone au centre afin d'optimiser la consommation de ressources sur le téléphone. Nous proposons également un format BSSDF qui permet d'envoyer des données semi-structurées au centre avec une taille réduite. Les données à envoyer au centre contenant de nombreux longs chiffres, la taille de la description sous forme binaire sera réduite. De plus, BSSDF utilise un minimum de balises, ce qui réduit considérablement la taille des données par rapport au type de données semi-structuré XML.

2) Pour la détection des anomalies, nous proposons une méthode de détection basée sur la caractérisation des valeurs aberrantes en statistique. Cette méthode ne permet pas d'identifier les anomalies sur la base de paramètres fixes, mais plutôt sur le principe de comparaison des données capturées avec les données immédiatement avant et après celles-ci dans un échantillon de données continu. Donc, cette méthode permet de développer des algorithmes qui s'adaptent aux conditions environnementales réelles et à la qualité de l'équipement. En même temps, notre méthode nécessite moins de mémoire et moins de calcul.

Nous avons effectué des expériences sur la détection des anomalies routières avec des

données collectées dans la ville de Hué au Vietnam, afin de valider notre approche. Nous proposons également une méthode de traitement avec l'orientation aléatoires du téléphone et éliminons les actions de l'utilisateur appliquées en cas de détection des anomalies routières.

Nos expériences montrent que nos algorithmes améliorés donnent de meilleurs résultats que les algorithmes originaux. En particulier, lorsque des algorithmes améliorés sont appliqués à différents ensembles de données déférents (collectés avec le véhicule, collectés par scooter et toutes les données collectées), les résultats sont presque constants. Cela prouve que notre méthode répond bien aux conditions réelles.

3) Pour synthèse des données, nous proposons une méthode en deux étapes :

- Les données sont regroupées géographiquement de manière cumulative. Le but de cette classification est de scinder les anomalies rapportées en zones non liées afin d'accélérer la prochaine étape.
- La deuxième étape est effectuée sur chaque zone de données générée à partir de la première étape. Nous proposons un algorithme qui permette de trouver les positions les plus plausibles des anomalies. Cette méthode repose sur la recherche du nombre maximal de points de densité de probabilité. Nous proposons également une formule de pondération des points trouvés pour évaluer la fiabilité. Nous effectuons également des expériences de simulation pour évaluer les résultats. Les résultats de nos expériences montrent que notre algorithme est très efficace.

1.3 Organisation du document

Le document est organisé en 6 chapitres. Le premier chapitre donne un aperçu du contexte de la recherche, de nos motivations et de nos objectifs, ainsi que de nos contributions.

Le chapitre 2 décrit les domaines d'application de SCS et les recherches en cours liées à nos recherches, notamment les architectures typiques des systèmes SCS de type crowensensing et les méthodes de détection des anomalies routières à l'aide de *smartphones* .

Le chapitre 3 présente le modèle du système SCS pour la détection d'anomalies environnementales, le mécanisme de fonctionnement et la structure des données échangés entre les *smartphone* et centre.

Le chapitre 4 décrit des méthodes pour améliorer les algorithmes de détection d'anomalies. Des expériences sur des données réelles pour la détection d'anomalies routières sont également présentées dans ce chapitre.

Le chapitre 5 présente les besoins, le manque de moyens efficaces pour synthétiser les données des anomalies et notre solution. Ce chapitre décrit les algorithmes d'agrégation de données ainsi que les méthodes et les résultats de la simulation.

Le dernier chapitre conclut ce document en présentant un résumé de nos recherches, les contributions de la thèse ainsi que de futures perspectives.

1.4 Publications liées à la thèse

- Van Khang Nguyen, Éric Renault, and Ruben Milocco. "Event Aggregation for Smartphone-based Road-Anomaly Detection". In : *The 8th IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*. November 26-28, 2019.
- Van Khang Nguyen, Éric Renault, and Ruben Milocco. "Environment monitoring for anomaly detection system using smartphones". In :*Sensors* 19.18 (2019), p. 3834.
- Van Khang Nguyen and Éric Renault. "Cooperative Sensing and Analysis for a Smart Pothole Detection". In : *2019 15th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE. 2019, pp. 1785–1790.
- Van Khang Nguyen, Éric Renault, and Viet Hai Ha. "Road Anomaly Detection Using Smartphone : A Brief Analysis". In : *International Conference on Mobile, Secure, and Programmable Networking (MSPN)*. June 18–20, 2018. Springer LNCS 11005, pp. 86–97.

2 État de l'art

2.1 Applications basées sur les capteurs des smartphones

Prenant avantage des possibilités offertes par les *smartphones*, les systèmes basés sur les capteurs des *smartphones* constituent un domaine d'intérêt émergent pour les chercheurs. Ces systèmes ont été étudiés et largement appliqués dans de nombreux domaines. Cette section présente quelques domaines typiques ainsi que quelques exemples de recherches correspondantes.

2.1.1 Surveillance de l'environnement

La surveillance de l'environnement est l'un des domaines d'application les plus courants utilisant un système de capteurs basé sur *smartphones*.

Surveillance du bruit et de l'ambiance

Le microphone des *smartphones* peut être utilisé pour mesurer le niveau de bruit et donner un aperçu de la nature des événements contextuels. Dans les systèmes NoiseTube [40], Ear-Phone [56] et SoundOfTheCity [58], les niveaux de bruit sont utilisés pour surveiller la pollution par le bruit. Les données audio sont combinées aux données GPS pour créer une carte de la pollution sonore en temps réel.

Surveillance des conditions de route et de circulation

De nombreuses recherches utilisent les microphones, les caméras ou les accéléromètres avec capteur GPS pour surveiller la qualité de la route, les feux de circulation ou les conditions de circulation. Dans Nericell [44], l'accéléromètre, le microphone et le capteur GPS sont utilisés pour détecter et localiser les conditions de circulation et les conditions de la route, comme par exemple les nids-de-poule, les bosses ou les coups de frein. L'application intègre les informations fournies sur la rugosité de la surface des routes, le bruit environnant et les conditions de circulation dans des cartes de circulation accessibles au public.

Le système Vtrack [66] utilise le GPS des *smartphones* pour suivre et estimer les retards de circulation sur les routes. Alors que dans SignalGuru [34], la vidéo de la caméra est collectée pour identifier et analyser le temps d'attente aux feux de circulation. À partir de là, les horaires des feux de circulation peuvent être prédits avec précision.

Le système CrowdITS [1] est un système de transport intelligent pour aider les autorités responsables des transports et les conducteurs de véhicules à prendre des décisions informatives et fournir une expérience de conduite sécuritaire et de loisirs. Le CrowdITS utilise des capteurs GPS en conjonction avec la saisie manuelle d'événements tels que les congestions, les constructions, les fermetures, les pannes, les collisions, et les incidents. Le système WreckWatch [71], et le système de Chris Thompson et al. [67], utilisent l'accéléromètre et le capteur GPS des *smartphones* pour détecter les accidents de voiture et autres incidents afin de réduire la congestion globale du trafic et augmenter l'efficacité des interventions d'urgence.

2.1.2 Soins et santé

Concernant la surveillance de la santé personnelle, les téléphones mobiles sont utilisés pour surveiller l'état physiologique et la santé des patients ou des participants à l'aide de capteurs. Par exemple, le système DietSense [57] assiste les participants qui souhaitent perdre du poids en documentant leurs choix alimentaires au moyen d'images et d'échantillons sonores. Les professionnels de la santé qui effectuent des analyses sur l'état des patients peuvent parcourir et annoter rapidement les données des participants. Le système StressSense [38] permet de reconnaître le stress de la voix humaine à l'aide d'un *smartphone*.

Dans E-FallD [7], et dans le système de détection de chute de Yi He et al. [28], l'accéléromètre du *smartphone* est principalement utilisé pour détecter les chutes des utilisateurs (des patients ou des personnes âgées).

Un certain nombre de recherches utilisent l'accéléromètre des *smartphones* pour détecter le symptôme *freezing of gait* (mouvements pas à pas) qui est un déficit de la marche courant dans la maladie de Parkinson avancée [8, 50].

2.1.3 Environnement social

Les systèmes basés sur les capteurs des *smartphones* s'appliquent aussi à plusieurs facteurs liées à l'environnement social, comme par exemple certaines recherches sur l'utilisation des capteurs de *smartphones* pour enrichir le contenu partagé sur les médias sociaux, tels que les blogs et les réseaux sociaux [42, 43]. Sur la base des informations multimodales (accélération, audio, images, localisation) capturées par le téléphone mobile, les informations de contexte sont déduites sous diverses dimensions, notamment l'humeur de l'utilisateur, sa localisation et ses habitudes, ainsi que des informations sur l'activité en cours et l'environnement.

Le système PEIR *(Personal Environmental Impact Report)* permet aux utilisateurs d'utiliser leur téléphone portable pour déterminer leur exposition aux polluants environnementaux [43]. Un module de détection installé sur le téléphone détermine l'emplacement actuel de l'utilisateur, ainsi que des informations sur le mode de transport utilisé et transfère ces informations à un serveur central. En retour, le serveur fournit aux utilisateurs des informations sur l'impact environnemental de leurs déplacements.

LineKing est un système permettant de surveiller et de prévoir les temps d'attente des lignes de café [6]. LineKing se compose d'un composant placé sur le *smartphone* et qui fournit une détection automatique et précise du temps d'attente. Les résultats de nombreux *smartphones* sont stockés dans le cloud pour fournir aux participants une estimation précise du temps d'attente.

2.2 Modèles SCS pour la détection des nids de poule

Il existe actuellement un certain nombre de modèles de système différents pour SCS. Nous avons choisi d'analyser les architectures de système SCS décrivant les composants du système et les mécanismes de fonctionnement. Nous nous concentrons principalement sur les modèles de systèmes SCS pour la détection des anomalies environnementales. La plupart de ces systèmes s'appliquent à la détection des anomalies routières. Quelques modèles typiques sont présentés ci-dessous.

Jakob Eriksson et al. ont proposé le système Pothole Patrol (P^2) [21] pour détecter les nids de poule. Le système est construit sur un modèle centralisé (voir la Fig. 2.1). Selon ce modèle, les quatre premiers paramètres proviennent du GPS et le vecteur d'accélération de l'accéléromètre à trois axes. Ces deux flux de données sont combinés par la composante *Location Interpolator* qui effectue une interpolation des données GPS. La composante *Pothole Detector* filtre le flux de données combiné pour détecter les nids-de-poule. Lorsque la connectivité réseau est disponible, les anomalies détectées sont téléchargées automatiquement sur un serveur central, qui conserve une base de données des nids de poule. La composante *Clustering* du serveur effectue un partitionnement sur les anomalies détectées et applique une taille de cluster minimale, produisant ainsi le résultat final du système : une série de nids de poule de confiance et de sévérité variables.

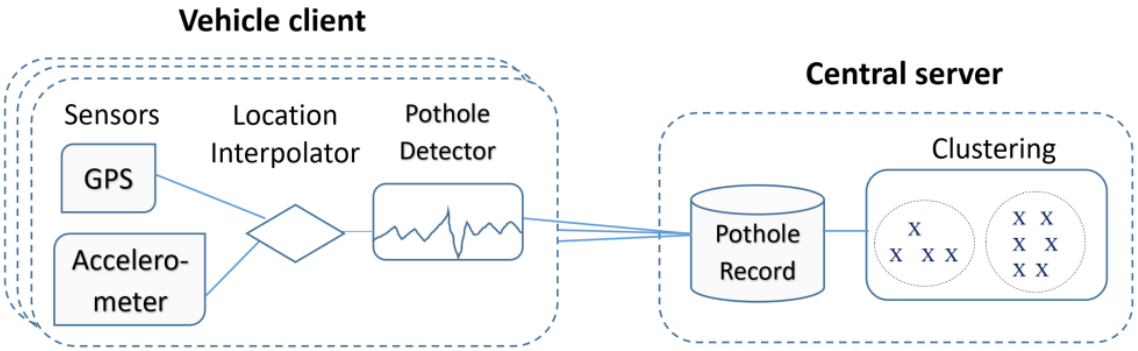


FIGURE 2.1 – Architecture du système P^2.

Girts Strazdins et al. ont proposé un réseau de capteurs à base de *smartphones* Android pour la surveillance de surfaces routières [63]. Les auteurs ont mis au point un modèle permettant de collecter des données à partir de capteurs sur les *smartphones* et de les envoyer au centre afin de détecter les nids-de-poule (voir la Fig. 2.2).

La liste des capteurs utilisés inclut le GPS pour déterminer la position actuelle et l'accéléromètre pour détecter les nids de poule. Une fois que les données du capteur sont reçues, elles sont traitées et stockées dans la base de données synchronisée périodiquement avec la base de données du serveur principal afin que les deux aient une information à jour. Une interface utilisateur permet de démarrer/arrêter le processus d'échantillonnage et de voir les informations de débogage. Les résultats de la détection sont sauvegardés dans la base de données des nids-de-poule. Cette architecture permet aux auteurs de tester des algorithmes de détection de nids-de-poule. Les auteurs envisagent à l'avenir d'utiliser un programme de détection des nids-de-poule qui fonctionnerait comme un service en arrière-plan.

Zhaojian Li et al. ont utilisé une architecture Véhicule-à-Cloud-à-Véhicule comme dans la Fig. 2.3 [63]. Une fois les anomalies détectées, leurs positions, obtenues par

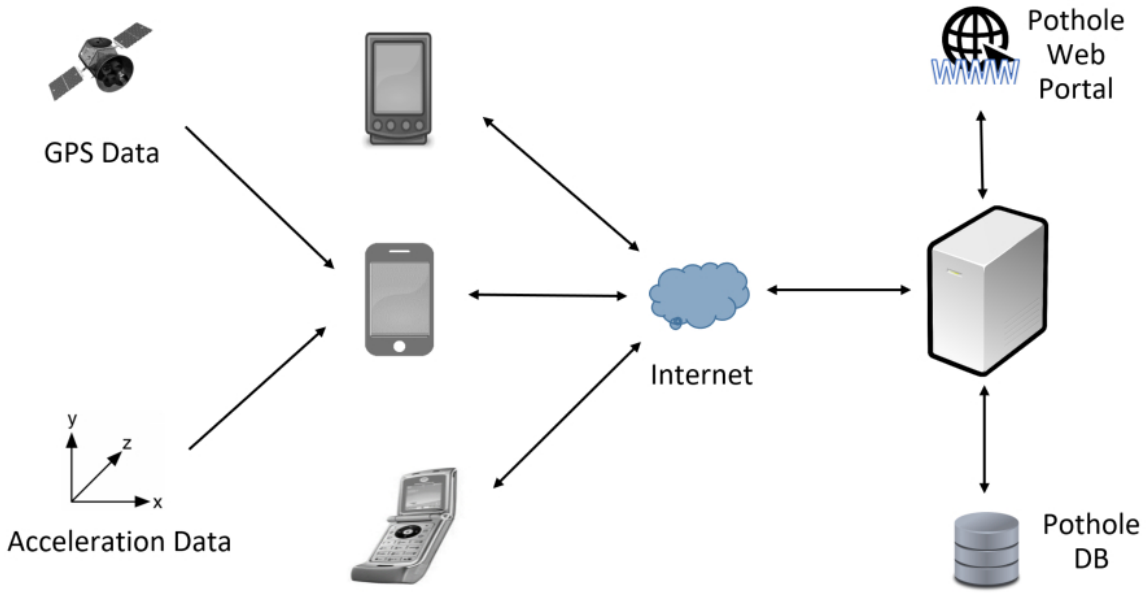


FIGURE 2.2 – Architecture du système proposé par G. Strazdins et al.

exemple à partir des coordonnées GPS *(Global Positioning System)*, sont envoyées au nuage, où un module de mise en cluster est implémenté pour traiter les rapports d'anomalies bruts. Les clusters avec un score de crédibilité élevé sont stockés dans une base de données cloud où leurs positions peuvent ensuite être diffusées à d'autres véhicules et agences routières. Les clusters avec un score de crédibilité faible sont stockés dans une mémoire tampon et ne sont pas partagés.

Raghu K. Ganti et al. ont présenté l'architecture typique du système Mobile Crowdsensing (MCS) [23] dont le schéma est présenté en Fig. 2.4. C'est le modèle général pour les systèmes de capteurs à base de samrtphopnes. Par conséquent, les fonctions ne sont pas définies spécifiquement.

La fonction principale sur les *smartphones* est *Localized Analytics*. Certains capteurs tels que GPS, accéléromètre, microphone et caméra sont disponibles sur les appareils mobiles. Le système d'exploitation (OS) permet aux applications d'accéder aux capteurs et d'en extraire des données de détection brutes. Toutefois, selon la nature des données brutes et les besoins des applications, les lectures physiques effectuées par les capteurs peuvent ne pas être adaptées à la consommation directe des applications. Parfois, certaines analyses locales effectuant certains traitements primitifs des données brutes sur le périphérique sont nécessaires. Ils produisent des résultats intermédiaires, qui sont envoyés au *Back-end server* pour traitement et consommation ultérieur. Par exemple, dans une application de détection de nids-de-poule, une analyse locale détermine les nids-de-poule potentiels à partir des données du capteur d'accélération sur trois axes.

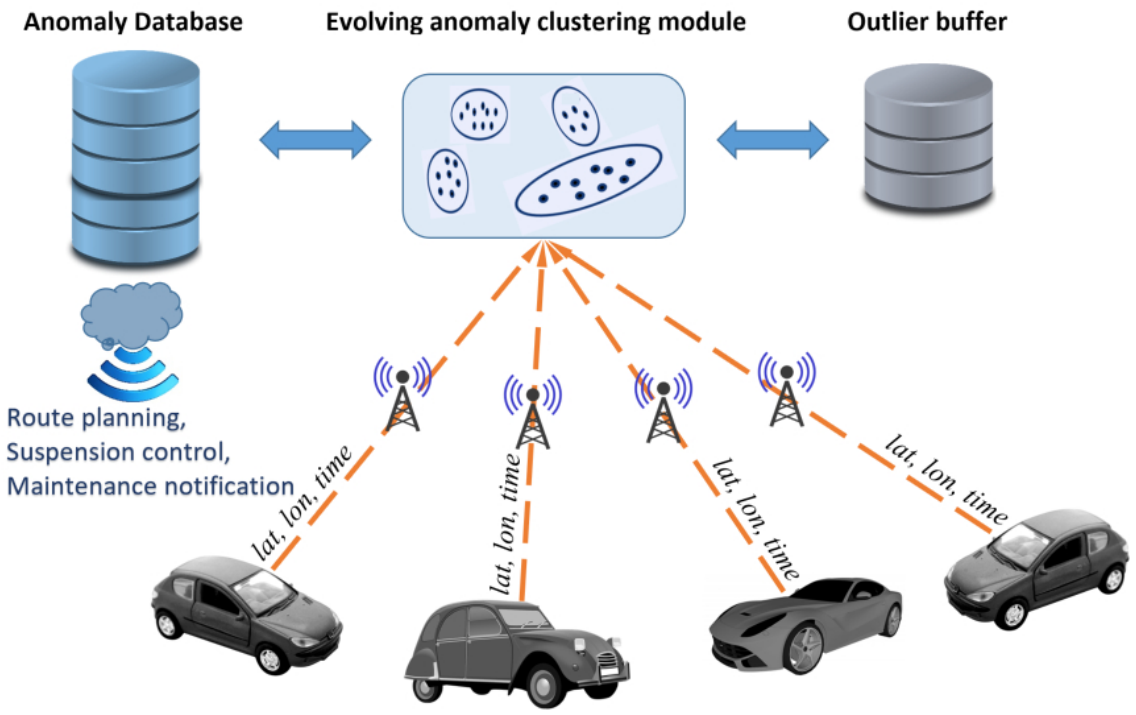


FIGURE 2.3 – Architecture du système Véhicule-à-Cloud-à-Véhicule proposé par Zhaojian Li et al.

Le composant *Agregate Analytics* analyse les données collectées des appareils mobiles pour identifier des motifs spatio-temporels *(spatio-temporal patterns)*. Par exemple, l'autorité de transport d'une ville peut être intéressée par la distribution spatiale des points chauds du trafic autour du réseau routier et par l'évolution de la distribution sur différentes périodes. Ces informations peuvent les aider à mieux coordonner les feux de signalisation pour faciliter la circulation en fonction de l'heure du jour et à mieux planifier l'expansion future des routes à long terme afin de réduire les encombrements.

En général, les modèles sont relativement similaires. L'utilisation de *smartphones* permet une connexion à l'Internet et l'envoi d'informations au centre. Par conséquent, les modèles courants utilisent des serveurs centraux pour stocker les résultats. Cela simplifie la structure du système. Le transport et l'intégration des données sont également beaucoup plus simples que les réseaux de capteurs conventionnels. Nous analysons les caractéristiques de ces modèles afin de pouvoir créer des modèles pour les cas où le système détecte des anomalies environnementales d'utilisation du *smartphone*.

L'architecture du système P^2 semble être l'architecture de base de la détection des anomalies basée sur les données de l'accéléromètre et des capteurs GPS. Nous nous appuyons sur ce modèle pour développer davantage de fonctions afin d'améliorer le système et améliorer l'efficacité et de réduire la consommation de ressources sur les

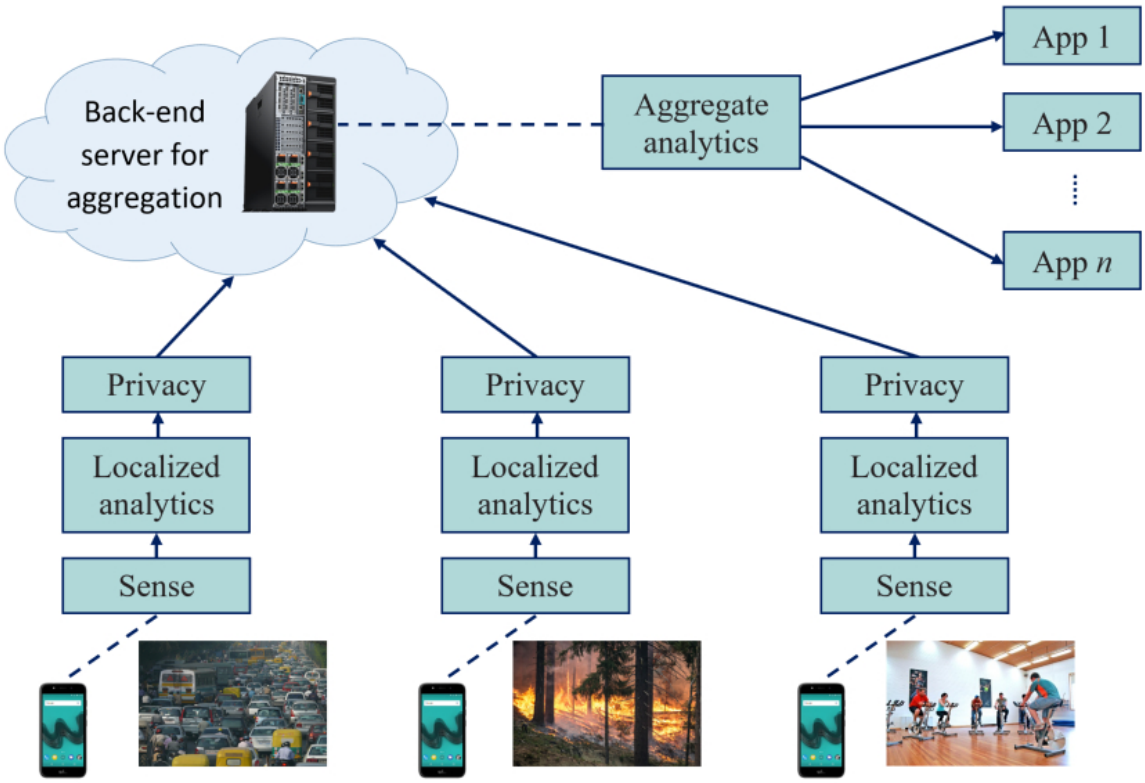


FIGURE 2.4 – Architecture du système MCS selon R.K. Ganti et al.

smartphones. Cependant, selon nous, le composant *Location Interpolator* peut s'intégrer après la détection d'anomalies. Effectuer une interpolation GPS avant la détection augmente les traitements inutiles et augmente la quantité de données. Cela oblige le programme à utiliser davantage de ressources sur les *smartphones*. Nous recommandons donc d'abord d'utiliser les informations GPS pour déterminer les conditions de déclenchement de la détection. Chaque fois qu'une anomalie est détectée, le programme interpole pour trouver la position de l'anomalie.

Dans le modèle de Girts Strazdins et al., la détection des anomalies est effectuée au centre. Ce modèle est utile pour tester des algorithmes. Toutefois, si il est appliqué dans la pratique, ce modèle n'est pas pratique car les données envoyées au centre sont trop volumineuses. L'envoi de ces données volumineuses augmente non seulement les coûts au niveau du réseau, mais consomme également de la mémoire et la batterie du *smartphone*. L'opération d'envoi des données sollicite beaucoup la batterie du téléphone. De plus, le téléphone n'est pas toujours connecté au centre. Lorsque le *smartphone* n'est pas connecté, le programme sur le téléphone doit stocker temporairement des données. La quantité de données provenant des capteurs peut être importante. L'installation d'un élément permettant une détection préliminaire des anomalies dans le *smartphone* permettrait de réduire la quantité de données envoyées au centre, en

exploitant davantage les ressources informatiques sur le *smartphone*. Bien entendu, l'exploitation de la puissance de calcul des téléphones doit être exploitée correctement pour optimiser le système.

L'architecture du système proposé par Zhaojian Li et al. est similaire au système P^2. Sa particularité est que les auteurs ont proposé de séparer les deux bases de données d'anormalies. La base de données *Anomaly database* a pour but de stocker les anomalies confirmées à partager avec des applications. La base de données *Outlier buffer* sert à stocker les anomalies incertaines. Cette division est due aux caractéristiques de la méthode de classification proposée par les auteurs, qui peuvent être mises à jour régulièrement, et ne convient pas à d'autres méthodes de classification. Il aurait probablement été plus pertinent de sauvegarder les données des anomalies confirmées dans une base de données propre et partagée avec les applications. Les données brutes devant toujours être stockées dans une base de données commune pour une future mise en cluster.

Le système MCS de Raghu K. Ganti et al. est un système polyvalent permettant d'exploiter les capteurs sur les *smartphones*. Par conséquent, il ne contient pas d'information détaillée sur les traitements. Les fonctions n'ont pas été spécifiées. En réponse au système de détection d'anomalies, ce modèle doit être personnalisé.

2.3 La détection des anomalies routière utilisant le smartphone

La détection des anomalies routières est basée sur une variété d'informations telles que l'audio, l'image, la vidéo, le laser 3D ou les données d'accélération [33]. L'approche basée sur des données obtenues depuis les accéléromètres, également appelée méthode basée sur la vibration, est la plus couramment utilisée dans les systèmes à base de *smartphones* pour la détection des anomalies routières. Les méthodes actuelles pour la détection des anomalies routières peuvent être classées en deux catégories : les méthodes *basés sur un seuil* et les méthodes *basés sur une classification* [47].

2.3.1 Les méthodes basés sur un seuil

La caractéristique commune des méthodes basées sur un seuil est l'utilisation d'algorithmes simples pour déterminer les nids-de-poule selon le principe suivant : si un composant ou un dérivé des composants d'une accélération mesurée dépasse un seuil

spécifique alors une anomalie est détectée (voir la Fig. 2.5). Quelques recherches typiques de cette classe sont présentées ci-dessous.

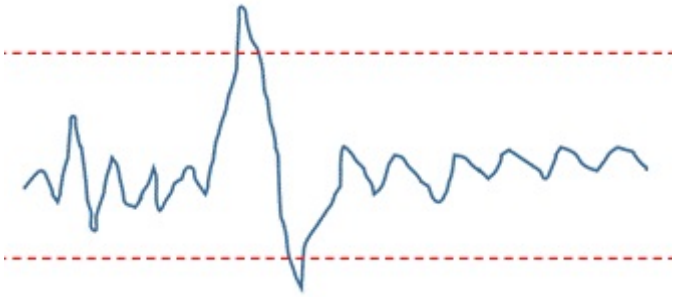


FIGURE 2.5 – Exemple pour la détection basée sur un seuil.

Le système *Pothole Patrol*

Jakob Eriksson et al. ont proposé un système appelé *Pothole Patrol* (P^2) [21]. Afin de détecter les anomalies routières, ils utilisent des accéléromètres et des capteurs GPS installés dans les taxis. Les informations collectées à partir des capteurs sont :

$<$ `temps, location, vitesse, entête, accélération 3 axes` $>$

La détection est effectuée par un véhicule embarquant un ordinateur. Les résultats (nids-de-poule) sont envoyés à un centre de contrôle via un réseau WiFi. Le serveur calcule les grappes de nids-de-poule pour éviter les analyses faussement positives.

L'algorithme de détection de nid-de-poule proposé dans P^2 comprend cinq filtres. Les informations reçues des capteurs sont d'abord segmentées en fenêtres de 256 échantillons. Les filtres testent ensuite chaque fenêtre pour rejeter les types d'événements autres que les nids-de-poule (voir la Fig. 2.6). Ces filtres sont :

- **Speed** : ce filtre rejette les fenêtres dans lesquelles la vitesse est trop petite ;
- **High-pass** : cette étape rejette les composantes basse fréquence de la mesure de l'accélération sur les axes x et z. Ce filtre supprime les événements tels que l'accélération, le virage et le freinage ;
- **z-peak** : cet important filtre rejette toutes les fenêtres dont la composante z de l'accélération est inférieure au seuil t_z ;
- **xz-ratio** : cette étape permet de rejeter les événements affectant les deux côtés d'un véhicule, tels que les passages à niveau, les ralentisseurs et les joints de dilatation. Une fenêtre est rejetée si le pic de la composante x de l'accélération dans une sous-fenêtre du pic de la composante z est inférieur à un facteur t_x multiplié par le pic de la composante z. À titre d'exemple, la taille de sous-fenêtre utilisée est 32 ;

- **speed vs z -ratio** : cette étape supprime les fenêtres dans lesquelles un pic sur l'axe z de l'accélération est inférieur au facteur t_s multiplié par la vitesse.

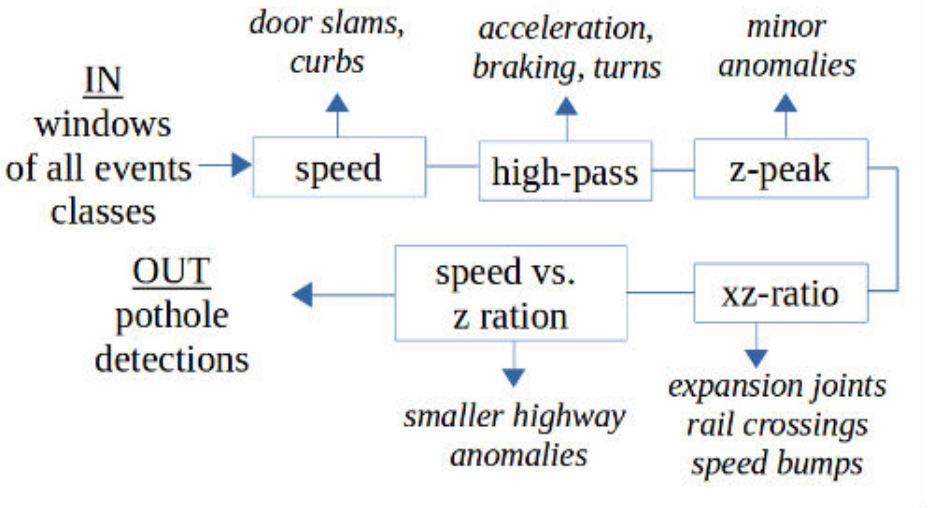


FIGURE 2.6 – La détection du système P^2.

Afin de détecter les paramètres tz, t_x et t_s, les auteurs utilisent des exemples de données pour entraîner le détecteur de nids-de-poule.

Le système de surveillance routière Nericell

Prashanth Mohan et al. ont présenté un système de surveillance des conditions de la route [44]. La détection est effectuée sur le *smartphone*. Le système utilise des capteurs de type accéléromètre, microphone, radio GSM et GPS pour détecter des nids-de-poule, des dos d'âne, des coups defreins et des klaxons. Les auteurs ont proposé également d'utiliser les angles d'Euler pour représenter l'orientation de l'accéléromètre et une méthode d'estimation des angles en fonction de la gravité, des actions de freinage ou des accélérations du véhicule. Deux algorithmes sont utilisés pour déterminer les anomalies :

- **Détection du freinage** : soit a_x l'accélération sur l'axe x. L'algorithme consiste à calculer $\overline{a_x}$ sur une fenêtre glissante. Lorsque $\overline{a_x}$ est supérieur au seuil T, un événement de freinage est détecté. La taille de la fenêtre utilisée par les auteurs est de 4 secondes ;
- **Détection des anomalies routières** : Nericell ne fait pas la distinction entre les types de anomalie routière telles que les nids-de-poule et les ralentisseurs. À grande vitesse (c'est-à-dire lorsque la vitesse est supérieure à 25 km/h), les auteurs utilisent la méthode z-peak [21] : une anomalie est détectée lorsque

a_z est supérieur à un seuil donné. À faible vitesse, les auteurs proposent un filtre appelé z-sus. Avec z-sus, une anomalie est détectée lorsqu'il y a un creux soutenu, par exemple lorsque a_z est inférieur au seuil T pendant au moins 20 ms.

L'approche proposée par Mednis et al.

Mednis et al. ont proposé quatre algorithmes pour la détection des anomalies routières en temps réel en utilisant les *smartphones* [41]. Tous ces algorithmes sont basés sur les données d'accélération sur les trois axes :

- **Z-THRESH** est similaire à l'algorithme z-peak utilisé dans [21] et [44]. Les événements sont détectés lorsque l'amplitude de la valeur z de l'accélération dépasse un seuil spécifié ;
- **Z-DIFF** : des événements sont détectés lorsque la différence entre deux valeurs consécutives est supérieure à un seuil spécifique ;
- **Z-STDEV** : l'écart-type sur une petite fenêtre glissante est calculé. Lorsque cet écart-type est supérieur à un seuil spécifique, un événement est détecté ;
- **G-ZERO** : un événement est détecté lorsque la valeur des trois axes est inférieure à un seuil spécifique.

L'approche proposée par Astarita Vittorio et al.

Dans cette approche [69], les auteurs ont appliqué une méthode de détection des anomalies routières aux dispositifs mobiles basées sur le signal d'accélération et le signal GPS. Les auteurs ont proposé d'appliquer une ré-orientation partielle pour recalculer la valeur verticale de l'accélération (a_z). La fréquence des données GPS étant de 1 Hz et celle des données d'accélération de 5 Hz, les auteurs ont regroupé les données de l'accéléromètre par groupes de 1 seconde en calculant a_{z_min}, a_{z_max} et a_{z_avg}.

Les auteurs proposent un algorithme de détection, nommé dans cette thèse DVA-THRESH, basé sur l'impulsion d'accélération verticale définie par $DVA = a_{z\_max} - a_{z\_min}$.

Les données de l'accéléromètre ont également été filtrées au moyen d'un algorithme de post-traitement afin d'éliminer les composantes de basse fréquence dans le signal en raison du bruit de fond :

$$DVA = \begin{cases} 0 & \text{si } DVA \leq DVA^{st} \\ DVA - DVA^{st} & \text{si } DVA > DVA^{st} \end{cases}$$

où DVA^{st} est l'anomalie maximale acceptable dans un état stationnaire. Afin de filtrer les enregistrements non anormaux, DVA est filtré comme suit :

$$DVA = \begin{cases} 0 & \text{si } DVA \leq DVA^{th} \\ DVA & \text{si } DVA > DVA^{th} \end{cases}$$

où DVA^{th} est un ensemble de référence de valeurs DVA. Les expériences permettent de déterminer la valeur optimale de DVA^{th}.

2.3.2 Les méthodes basées sur la classification

Pour les méthodes basées sur la classification, les caractéristiques sont d'abord extraites des données collectées par plusieurs méthodes. Ensuite, une classification est appliquée à ces caractéristiques pour les classer afin de détecter les anomalies routières (voir la Fig. 2.7).

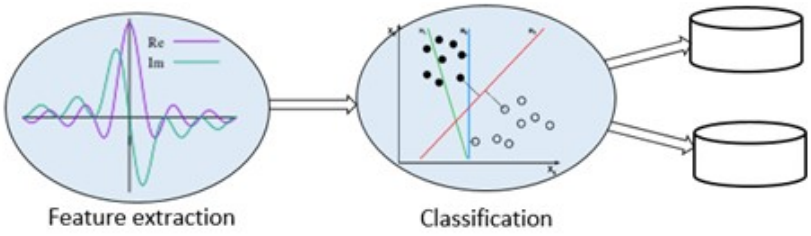


FIGURE 2.7 – Le principe des méthodes basées sur la classification.

L'approche de Mikko Perttunen et al.

La méthode de détection des anomalies proposée par Perttunen et al. [51] comporte plusieurs étapes complexes. Les données (signaux GPS et accélération) passent d'abord par un prétraitement pour retirer les valeurs aberrantes du GPS, et des filtres de Kalman sont appliqués pour réduire le bruit. Ensuite, les signaux sont encadrés en utilisant une fenêtre glissante pour extraire certaines caractéristiques. Nombre d'entre elles sont extraites du signal d'accélération : moyenne, variance, écart-type, amplitude sur les axes x, y et z, corrélation entre les trois dimensions... Une transformation de Fourier rapide est également appliquée pour incorporer des informations provenant de fréquences spécifiques. Les auteurs appliquent également une méthode de suppression de dépendance linéaire comme dans [65] pour supprimer la dépendance des caractéristiques sur la vitesse. Cette méthode est basée sur l'information mutuelle (introduite par Shannon [61]). Enfin, ils appliquent la méthode *Support Vector Machines* (SVM) [16] pour classifier les fenêtres entre anomalie et non-anomalie.

L'approche de Fengyu Cong et al.

Fengyu Cong et al. ont proposé d'appliquer la décomposition de paquets d'ondelettes (WPD – *Wavelet Packet Decomposition*) et SVM aux données d'accélération [15]. WPD est utilisé pour l'extraction des caractéristiques. Le signal d'accélération est segmenté en fenêtres de 114 échantillons. Ensuite, un WPD utilisant les ondelettes de Daubecheis [18] est appliqué pour extraire les caractéristiques. La deuxième étape est la sélection des caractéristiques. Enfin, la méthode SVM est utilisée pour classer les caractéristiques afin de détecter les anomalies. Afin de réduire le calcul du classifieur, une sélection d'un sous-ensemble des caractéristiques est effectuée. Les auteurs ont testé quatre méthodes de sélection : la sélection en avant [2], la sélection en arrière [2], un algorithme génétique [29] et l'analyse en composantes principales (PCA) [2].

- La sélection en avant commence par un ensemble des caractéristiques sélectionnées vide. La première caractéristique est sélectionnée en testant chaque caractéristique avec la classification pour choisir laquelle donne le meilleur résultat. Ensuite, l'algorithme continue en comparant quelle caractéristique non sélectionnée restante donne le meilleur résultat avec les caractéristiques précédemment sélectionnées. La procédure se termine lorsqu'il n'y a plus de caractéristique qui augmente de bon résultat ou lorsque le nombre de caractéristiques sélectionnées est égal à un maximum prédéfini ;
- La sélection en arrière est similaire à la sélection en avant à ceci près qu'elle commence par toutes les caractéristiques. À chaque tour, l'algorithme teste toutes les caractéristiques restantes et supprime la caractéristique qui diminue le plus le résultat jusqu'à ce que le nombre de caractéristiques atteigne la valeur souhaitée ;
- Les algorithmes génétiques sont couramment utilisés pour chercher une solution potentielle aux problèmes d'optimisation en utilisant une simple structure de données ressemblant à un chromosome, inspirée par l'évolution [35]. Dans cette recherche effectuée par Fengyu Cong et al., un chromosome est un vecteur binaire, la taille du vecteur est le nombre de caractéristiques. La valeur du $i^{ième}$ élément dans le vecteur est 1 si la $i^{ième}$ caractéristique est sélectionnée et 0 sinon. La population initiale des chromosomes est choisie au hasard dans l'ensemble de données d'origine. La reproduction se produit par croisement, mutation et sélection. Les chromosomes qui représentent une meilleure solution potentielle au problème ont de meilleures chances de sélection. Le processus se termine quand aucun meilleur résultat n'est trouvé ou que l'on est limité par le temps de calcul ;
- L'analyse en composante principale a pour but de transformer des variables (les caractéristiques dans ce cas) corrélées en nouvelles variables décorrélées les unes des autres par la décomposition en valeurs propres de la matrice de covariance

des variables. Le nombre des nouvelles variable est plus petite que dans celui des variables originales.

Le système RoSDS

Fatjon Sera et al. ont proposé le système de détection d'anomalies routières (*Road pavement Anomaly Detection System* – RoSDS) [60] basé sur *smartphones*. Les auteurs combinent les données de vitesse extraites des données GPS pour effectuer la démodulation des données d'accélération afin de séparer les données de vitesse des données d'accélération. Comme d'autres méthodes basées sur la classification, RoSDS extrait les caractéristiques depuis les données. Les caractéristiques extraites sont :

- pour le domaine temporel : la moyenne, la variance, l'écart-type, le carré moyen, la moyenne des valeurs absolues, le coefficient de corrélation entre les valeurs sur les axes, les angles d'inclinaison, etc.
- pour le domaine de fréquence : les caractéristiques sont extraites à l'aide d'une transformée de Fourier rapide (*Fast Fourier Transform* ou FFT).
- les caractéristiques sont extraites par *Stationary Wavelet Transform* (SWT) [45]. Le SWT permet aux auteurs de décomposer les signaux accélérés en représentation temps-fréquence. Les auteurs ont réalisé des expérimentations pour sélectionner le meilleur type d'ondelettes et niveau de résolution. Les caractéristiques extraites des résultats de la décomposition sont : la moyenne, la variance, l'écart-type et l'énergie.

Enfin, la méthode SVM est appliquées pour classer les caractéristiques. La classification comprend deux étapes : la première a pour but d'identifier les fenêtres anormales et la seconde consiste à classer le type d'anomalie.

La méthode Wolverine

Wolverine [4] est une méthode introduite par Ravi Bhoraskar et al. consistant à utiliser des capteurs de *smartphones* pour surveiller l'état des routes et détecter les anomalies routières. Une méthode de détection de choc et une méthode de détection de freinage basées sur le classificateur SVM sont proposées . Les données de l'accéléromètre réorientées sont établies dans des fenêtres d'une durée de 1 seconde (soit 50 échantillons). Plusieurs caractéristiques sont calculées à partir de ces fenêtres : la moyenne et l'écart-type sur trois axes (μ_x, μ_y, μ_z, σ_x, σ_y et σ_z). Ces valeurs sont utilisées pour détecter les anomalies. La détection du freinage est basée sur ces valeurs et sur les trois autres valeurs :

$$\delta_x = \max a_x - \min a_x \qquad \forall a_x \in \text{fenêtre}$$

$$\delta_y = \max a_y - \min a_y \qquad \forall a_y \in \text{fenêtre}$$
$$\delta_z = \max a_z - \min a_z \qquad \forall a_z \in \text{fenêtre}$$

Ces caractéristiques sont ensuite classées en utilisant SVM pour déterminer l'état du véhicule. Afin de créer des données étiquetées pour la formation SVM, les auteurs ont utilisé l'algorithme de classification K-means pour classer les caractéristiques en deux classes lisse ou bosselée (pour la détection des anomalies) et freinée ou non (pour la détection du freinage).

2.3.3 Comparaison des méthodes de détection des anomalies

Cette sous-section présente une analyse des méthodes ci-dessus dans le but d'examiner leur pertinence sur *smartphone*. Faute de moyen technique, il n'a pas été possible d'expérimenter chacune de ces solutions. De fait, la comparaison est faite de manière qualitative en termes de ressources nécessaires, de capacité d'adaptation aux différentes conditions de route, de type de véhicule et de vitesse. Le Tab. 2.1 résume les résultats obtenus.

Mémoire et CPU

Cette première partie est consacrée aux besoins en terme de ressources car ceci détermine la consommation d'énergie du *smartphone*. Cependant, il faut noter que la plupart des méthodes de détection nécessitent un entraînement. L'entraînement doit être réalisé une fois et peut être effectué sur ordinateur. Il n'est donc pas nécessaire de mentionner les ressources nécessaires à la l'entraînement ou aux expériences. En conséquence, seule la détection après la formation, pouvant être exécutée sur un *smartphone*, est prise en compte.

Méthodes basées sur un seuil

Tous ces algorithmes nécessitent peu de mémoire et de ressources informatiques. Comme les algorithmes utilisent des données directes recueillies à partir de capteurs, la fenêtre courante est la seule devant être stockée en mémoire. De plus, la complexité des algorithmes basés sur un seuil est basse.

- Le *système* P^2. Supposons que n soit le nombre d'échantillons d'accélération capturés sur l'ensemble de la route et devant être surveillés. Le signal est segmenté en fenêtres de taille k. Chaque fenêtre traverse cinq filtres. Pour les filtres *speed*, *High-pass*, *z-peak* et *speed vs. z-ratio*, le détecteur doit parcourir les fenêtres une seule fois. Cependant, pour le filtre *xz-ratio*, le programme recherche d'abord le

Tableau 2.1 – Résumé des méthodes basées sur les caractéristiques adaptées aux *smartphones*.

Méthode	Complexité	Mémoire[1]	Conditions[2]	Orientation[3]
Méthodes basées sur seuil				
Pothole Patrol [21]	basse	basse	faible	faible
Nericell [44]	basse	basse	faible	bonne
Algorithme Z-THRESH Z-DIFF, STDEV(Z) [41]	basse	basse	faible	bonne
Algorithme G-ZERO[41]	basse	basse	faible	moyenne
Méthode de Astarita Vittorio and al. [69]	basse	basse	faible	bonne
Méthodes basées sur la classification				
Méthode de Mikko Perttunen and al. [51]	haute	haute	bonne	moyenne
Méthode de Fengyu Cong and al. [15]	haute	moyenne	moyenne	moyenne
RoSDS [60]	haute	moyenne	moyenne	moyenne
Wolverine [4]	moyenne	moyenne	moyenne	bonne

[1] Utilisation de la mémoire — [2] Adaptation à des conditions réelles — [3] Adaptation à une orientation aléatoire

pic sur l'axe z de l'accélération, puis, pour chaque pic de z, il recherche un pic sur l'axe x de l'accélération dans une sous-fenêtre $\triangle w$. Comme la taille de $\triangle w$ est de 32 échantillons seulement, la complexité du filtre de *z-peak* est de $O(k)$. En conséquence, la complexité de l'algorithme devient $O(n)$;

- Dans le système Nericell, la détection du freinage utilise une fenêtre glissante. Ce détecteur parcourt toute la fenêtre pour chaque échantillon. En conséquence, la complexité globale est $O(k \times n)$ où n est le nombre total d'échantillons d'accélération et k est la taille de la fenêtre. Pour la détection des anomalies, pour les cas *z-peak* et *z-sus*, le détecteur parcourt les données une seule fois. Ainsi, la complexité du détecteur est $O(n)$. Lorsque deux détecteurs sont appliqués, la complexité totale devient $O(n \times k)$. Les auteurs ont choisi une taille de fenêtre de 4 secondes, c'est-à-dire $k = 4 \times f$, où f est la fréquence de l'accéléromètre ;
- Quant aux quatre algorithmes proposés par Artis Mednis et al., la complexité des trois algorithmes Z-THRESH, Z-DIFF et G-ZERO est égale à $O(n)$ parce qu'il parcourt la fenêtre une seule fois . Pour STDEV(Z), la complexité est $O(n \times k)$ où k est la taille de la fenêtre, car l'écart-type doit être calculé sur

une fenêtre glissante. Toutefois, la complexité de STDEV(Z) peut être réduite à $O(n)$ si l'écart-type peut être calculé progressivement ;

- L'algorithme proposé par Astarita Vittorio et al. est aussi de l'ordre de $O(n)$ car le détecteur ne doit parcourrir les données d'accélération qu'une seule fois.

Méthodes basées sur la classification

Contrairement aux algorithmes précédents, ces méthodes nécessitent beaucoup plus de ressources. De plus, tous les algorithmes partagent un aspect commun : avant les traitement, les données des capteurs sont divisées en fenêtres. Ensuite, une méthode d'extraction de caractéristiques, telle que FFT ou WPD est appliquée. Une fois les caractéristiques sélectionnées, une classification à l'aide de SVM est effectuée.

- Mikko Perttunen et al. [51] ont utilisé plusieurs outils tels que les filtres de Kalman, les FFT, les méthodes de suppression de dépendance linéaire et SVM ;
- Dans l'approche de Fengyu Cong et al. [15], WPD est utilisé pour extraire les caractéristiques, puis une méthode de sélection est appliquée et un SVM est finalement exécuté ;
- Les methodes FFT, SWT et SVM sont aussi utilisées dans RoSDS [60] ;
- Wolverine [4] est la méthode la moins intensive en terme de calculs pour l'ensemble des méthodes basées sur la classification. Neuf caractéristiques sont calculées à partir des fenêtres et classées à l'aide de SVM.

En bref, ces méthodes nécessitent plus de mémoire et de calcul. Par conséquent, si ces méthodes étaient déployées sur un *smartphone*, elles consommeraient beaucoup plus d'énergie que les précédentes. De plus, le programme de détection doit s'exécuter en arrière-plan. Si le programme utilise trop de ressources, celles-ci peuvent être hiérarchisées et/ou nettoyées par le système d'exploitation.

Le temps de traitement est également un facteur à prendre en compte. En règle générale, pour qu'un algorithme puisse s'exécuter en temps réel sans perte de données, le temps de traitement d'un bloc de données est inférieur au temps qui sépare deux arrivées de bloc. De plus, le programme de détection des anomalies ne devrait pas occuper une grande partie du processeur du *smartphone*.

Adaptation aux conditions réelles

L'un des principaux défis des algorithmes de détection d'anomalies de la route basés sur les vibrations est que l' amplitude et la fréquence des vibrations dépendent des

conditions de la route, de la vitesse, du type de véhicule et de la qualité du système de suspension, etc.

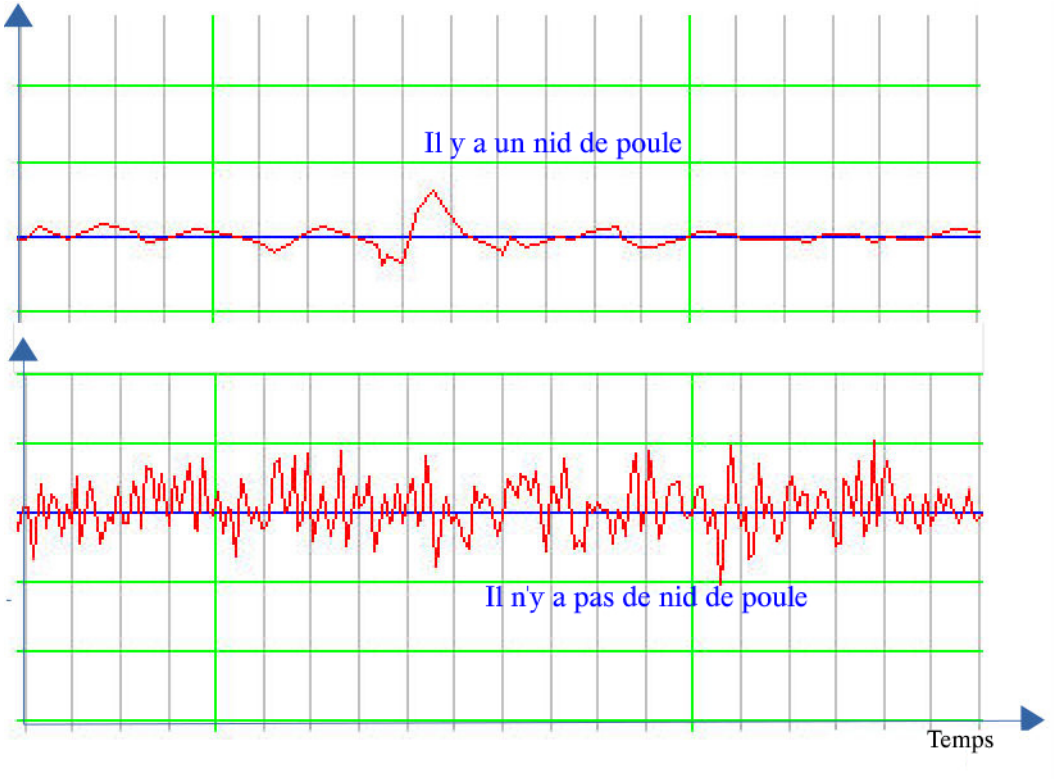


FIGURE 2.8 – Exemples pour deux conditions de route différentes.

Analyse des algorithmes basés sur des seuils

Les seuils utilisés par ces algorithmes sont fixes et les données captées sont comparées aux valeurs de pointe ou à l'écart-type des valeurs verticales des données d'accélération pour détecter les anomalies. Bien que ces seuils soient déterminés à partir d'expériences ou d'apprentissages automatiques, ils ne peuvent pas être appliqués à toutes les situations dans lesquelles l'amplitude et l'écart type de l'accélération dépendent de la vitesse, du type de route et du type de véhicule. Comme indiqué sur la Fig. 2.8, l'amplitude de l'accélération (composante verticale de l'accéléromètre) lorsqu'une voiture pénètre dans le nid-de-poule peut-être plus petite que dans d'autres cas où par exemple une autre voiture ne traverse pas de nid-de-poule mais roule sur une route caillouteuse. Dans les deux cas, les voitures roulent à une vitesse environ 7 m/s. En conséquence, un seuil dynamique, adaptable à la qualité de la route, serait beaucoup plus approprié pour ces algorithmes. Celui-ci peut être calculé/adapté régulièrement en fonction des conditions de la route.

Analyse des méthodes basées sur la classification

Les classifications basées sur de nombreux attributs permettent une meilleure identification des nids-de-poule dans différentes conditions réelles.

Cependant, pour classifier avec une bonne précision, les données d'entraînement doivent être une combinaison de différents types de routes et de véhicules à différentes vitesses. En fait, il est pratiquement impossible de créer un ensemble complet de données étiquetées. En outre, la formation doit être effectuée dans un centre de données et les paramètres téléchargés ultérieurement sur le *smartphone* pour économiser de l'énergie.

Adaptation à l'orientation aléatoire

En cours de fonctionnement, le *smartphone* ne peut pas être fixé sur un support de voiture ou de moto. En conséquence, l'orientation peut être orientée de manière aléatoire ou, pire encore, l'orientation peut changer tout le temps lors de la détection. Ainsi, les trois composantes de l'accélération doivent être calculées en fonction des axes de coordonnées attachés au véhicule. Cet ajustement appelé réorientation est également un défi important aujourd'hui.

Des outils ont été utilisés pour déterminer l'orientation d'un *smartphone*, tels que les angles d'Euler [44, 69] ou les matrices de rotation [4]. L'idée générale pour déterminer l'orientation d'un *smartphone* repose sur la détermination de deux directions : la direction verticale et la direction du mouvement. La direction verticale peut être déterminée à l'aide de la gravité. Il y a plusieurs approches pour déterminer la direction du mouvement. Dans [44], la direction du mouvement est déterminée lors d'un freinage quand le véhicule roule sur une droite. Dans ce cas, le vecteur d'accélération tourne vers la direction du mouvement. Une autre façon de déterminer la direction du mouvement consiste à utiliser les capteurs magnétiques : la combinaison du vecteur magnétique et de la direction du véhicule (basée sur le GPS) permet de déterminer l'angle de déviation du *smartphone* par rapport au véhicule [4].

En fait, il est difficile de déterminer l'orientation du *smartphone*. Par exemple, l'utilisation du freinage du véhicule pour déterminer la direction du véhicule à partir du *smartphone* peut ne pas être simple. Si le *smartphone* pivote mais que le véhicule ne change pas de vitesse en même temps ou s'il ne se déplace pas en ligne droite, il est impossible de déterminer la direction du véhicule à partir des données d'accélération. Si la direction du mouvement est déterminée à l'aide d'un capteur magnétique, la difficulté principale réside dans le fait que de nombreux *smartphones* ne comportent pas un tel capteur magnétique. De plus, il est également important de garder à l'esprit

qu'un capteur magnétique peut produire des résultats inexacts lorsqu'il est situé près d'un objet ferrique, ce qui est souvent le cas dans un véhicule.

2.4 Conclusion

Dans ce chapitre, nous avons présenté une vue d'ensemble des applications de SCS et avons donné plus de détails sur les aspects liés aux problèmes étudiés : architectures de SCS, méthodes de détection des anomalies routières, etc.

Les recherches actuelles montrent que l'application du SCS est très répandue dans de nombreux domaines. Pour un même but, l'approche peut être très différente et utiliser des types de capteurs variés. Cela rend difficile la construction d'un modèle commun. Par conséquent, nous visons à construire un système basé sur les SCSs pour la détection des anomalies routières avec la possibilité d'élargir l'application à d'autres types d'anomalies.

À partir de l'analyse de l'architecture des systèmes SCS il apparaît que chaque modèle présente des caractéristiques propres selon l'approche des auteurs contenant des fonctions de base mais incomplètes selon nos besoins en matière de SCS pour la détection d'anomalies. Sur cette base, nous avons construit un modèle plus approprié pour guider l'étude détaillée de chaque composant du système. Ce modèle est présenté au Chap. 3

Comme indiqué au Chap. 1, l'une des difficultés du SCS est que les données dépendent des conditions réelles de l'équipement et de l'environnement. L'une des applications les plus touchées par ces conditions est la détection des anomalies routières, en particulier la planéité de la route, le type de véhicule impliqué, la vitesse du véhicule et la qualité du capteur du téléphone. Ceci est également l'uns des applications les plus communes aujourd'hui concernant la détection des anomalies. Par conséquent, nous étudions les méthodes de détection des anomalies routières. L'analyse des études nous a permis de constater que les méthodes actuelles ne répondent pas bien aux conditions réelles. Les méthodes basées sur une classification sont plus prometteuses mais nécessitent des données d'apprentissage complètes. De plus, ces méthodes prennent beaucoup de ressources réseaux. Les méthodes basées sur un seuil utilisent moins de ressources mais sont moins adaptées aux conditions réelles. Par conséquent, nous proposons d'améliorer la méthode à base de seuils afin de trouver une méthode de détection des anomalies légère qui réponde bien aux conditions réelles d'installation sur les *smartphones* (voir le Chap. 4).

3 Architecture du SCS pour la détection d'anomalies

3.1 Introduction

Les applications utilisant les capteurs des *smartphones* deviennent de plus en plus populaires. Par conséquent, un modèle de système de capteurs de *smartphones* devient nécessaire pour les étudier. Sur la base d'un modèle commun, nous pouvons déterminer quelles études sont nécessaires, quelles améliorations peuvent être apportées et ce qui est nécessaire pour développer le système. Également à partir de ce modèle, nous développons des modèles plus spécifiques pour des cas précis.

Contrairement aux systèmes de capteurs conventionnels, les systèmes de capteurs de *smartphones* présentent l'avantage de la connexion. Les *smartphones* d'aujourd'hui sont facilement connectés à l'Internet *via* le réseau wifi ou la 4G / 5G avec un coût de connexion bas. Il est donc possible de créer un modèle client-serveur simple, dans lequel les téléphones agissent en tant que clients pouvant se connecter directement à des serveurs sur accessible *via* l'Internet.

Sur la base des avantages et des inconvénients des modèles actuels analysés en Sec. 2.2, nous avons construit un modèle commun de détection des anomalies basé sur l'utilisation de *smartphones*. Nous proposons quelques modifications et ajouts pour tirer parti des fonctionnalités des *smartphones* tout en réduisant la consommation de leur ressources.

3.2 Architecture du système

Nous analysons l'architecture des systèmes actuels et proposons un modèle pour la détection les anomalies. L'architecture du modèle constitue la base de nos études. Nous

nous référons aux composants en prenant en compte leur corrélation dans un système. Bien entendu, il n'existe pas de modèle complet pour tous les types d'applications. L'architecture du système que nous avons construit, comme illustré à la Fig. 3.1 , met uniquement en évidence le cœur du système que nous développons. Nous n'avons pas mentionné la sécurité, la gestion des droits ou la technologie appliquée. De plus, en fonction de l'application, la structure du système doit être modifiée pour être plus adaptée. En fonction de l'échelle, il est possible de créer des systèmes de serveur distribués.

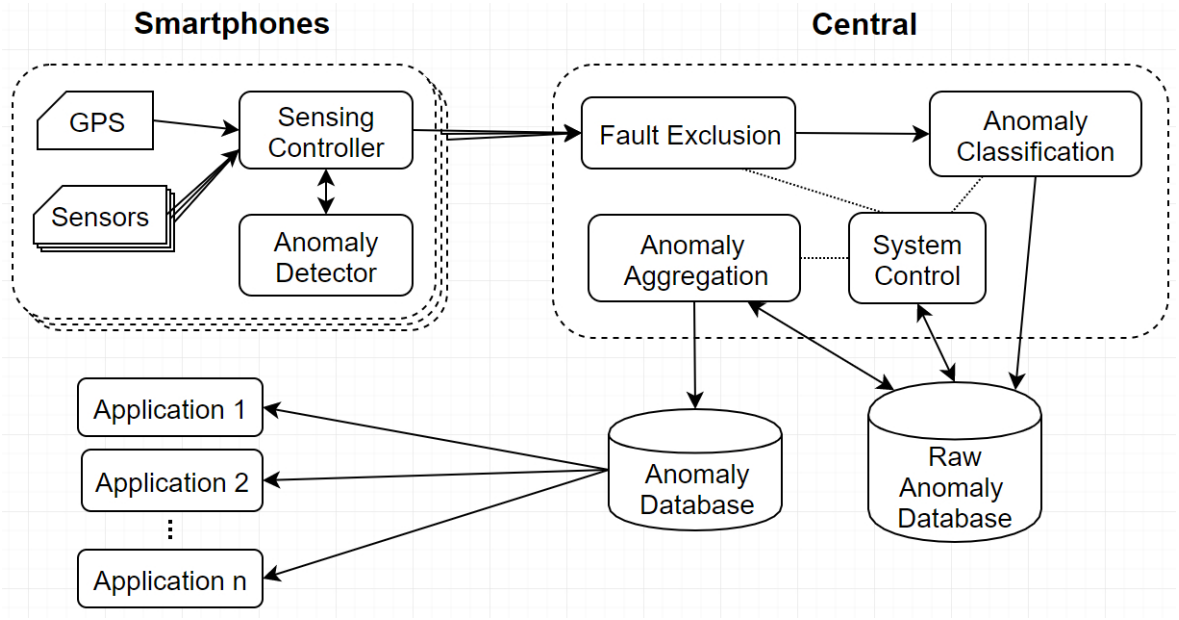


FIGURE 3.1 – Architecture du système de capteurs de *smartphones* pour la détection des anomalies.

Dans notre système de détection d'anomalies, nous avons divisé la tâche de détection en plusieurs étapes. Certaines de ces étapes sont gérées au niveau du serveur afin de réduire la consommation électrique des *smartphones*. La fonctionnalité des composants du système est décrite ci-dessous :

- le composant *Sensing Controller* est responsable du contrôle des capteurs, de la détection et de l'envoi des données afin de garantir la surveillance de l'environnement et de réduire la consommation d'énergie sur les *smartphones*. Ce composant transmet les données des capteurs au composant *Anomaly Detector* sous certaines conditions, par exemple lorsque le *smartphone* atteint une vitesse supérieure à une certaine valeur lors de la surveillance et des conditions de trafic. Ensuite, ce composant reçoit les anomalies et les envoie au serveur lorsque le *smartphone* est connecté à l'Internet ;
- *Anomaly Detector* est le composant du smartphone qui détecte initialement les anomalies ;

- le composant *Fault Exclusion* est destiné à éliminer les fausses anomalies causées par les actions de l'utilisateur. Ce composant peut être vu comme une partie détachée du *Anomaly Detector*. La séparation entre les deux composants, *Anomaly Detector* et *Fault Exclusion*, est destinée à répartir le traitement entre le client et le serveur afin d'optimiser la consommation de ressources des *smartphones* ;
- le composant *Anomaly Classification* classe les anomalies, ce qui permet également d'éliminer les anomalies moins fiables. La position des anomalies trouvées avec un poids supérieur à une valeur spécifiée est stockée dans *Anomaly Database* ;
- le composant *Anomaly Aggregation* regroupe les rapports d'anomalies de plusieurs *smartphones* pour localiser les anomalies et calculer un coefficient de confiance associé à chaque anomalie ;
- le composant *System Control* gère les paramètres des composants centraux. La gestion des paramètres est possible *via* l'interface utilisateur et le calcul automatique. En outre, ce composant est chargé de gérer la mise à jour et la suppression des anciennes données périodiquement ;
- les bases de données *Raw Anomaly Database* et *Anomaly Database* peuvent utiliser des systèmes de gestion de données ou des données en nuage. *Raw Anomaly Database* enregistre les anomalies détectées et classées. Ces données sont la base pour trouver la position la plus probable des anomalies. Ces positions sont stockées dans *Anomaly Database* pour servir à l'exploitation de l'application.

3.3 Les traitements sur smartphone

Le programme de détection sur le *smartphone* est divisé en composants distincts, qui ne fonctionnent que lorsque cela est nécessaire pour limiter la consommation de la batterie. Sur la Fig. 3.1, seuls deux composants sont présents : *Sensing Controller* et *Anomaly Detector*. Pourtant, selon les cas, il est possible que le composant *Fault Exclusion* fonctionne sur les *smartphones* si cela est plus efficace. De plus, le composant *Sensing Controller* comporte un module de communication qui ne fonctionne que lorsque le téléphone est connecté à l'Internet et qu'il y a un besoin d'échanger des paramètres ou que des anomalies sont à envoyer au centre.

Le composant *Sensing Controller* est activé en permanence dans le *smartphone* et contrôle les capteurs. Des capteurs minimum sont utilisés périodiquement pour suivre l'état de la détection. Lorsque cela est nécessaire, ce composant active tous les capteurs nécessaires à la détection. En même temps, il active également le composant *Anomaly Detector* pour effectuer une détection des anomalies. Les données sur les anomalies

détectées sont stockées dans une mémoire tampon. Le suivi de la connectivité et de l'envoi de ces données est effectué par le composant *Sensing Controller*.

Par exemple, en cas de détection des anomalies routières, le capteur GPS est activé après certains intervalles pour déterminer la vitesse du téléphone. Si cette vitesse est supérieure à un seuil défini, l'accéléromètre est activée et la détection est également déclenchée. Lorsque la vitesse est inférieure au seuil, le programme revient à son état d'origine. Il est à noter que la surveillance pour l'envoi de données est toujours effectuée si nécessaire.

Le programme de détection sur le *smartphone* fonctionne sur le principe d'activer seulement les composants et les capteurs lorsque c'est nécessaire. Ce mécanisme permet de réduire la consommation des processeurs, de la mémoire et de la batterie du téléphone. De plus, les algorithmes de détection doivent être peu complexes et nécessiter moins de mémoire. Étant donné que les *smartphones* doivent hiérarchiser les fonctions du propriétaire, ils ne concernent pas uniquement les systèmes de capteurs.

3.4 Organiser et échanger des données d'anomalie

3.4.1 Base de données et structure de donnée d'échange

Les systèmes de détection d'anomalies peuvent évoluer avec le temps. Les structures des différents types d'anomalies varient et peuvent changer avec le temps. Nous avons donc besoin d'une base de données avec des structures dynamiques. De plus, nous voulons ajouter facilement de nouvelles structures ultérieurement. Par conséquent, nous avons choisi le format de données XML. Le format XML utilisé est populaire et pris en charge par la plupart des langages de programmation et des systèmes de gestion de données. Avec le format de données XML, nous pouvons stocker et envoyer des anomalies dans différentes structures. Au passage, les données XML peuvent être transformées en d'autres types de données, telles que des données relationnelles, des données orientées objet, en fonction des besoins .

Cependant, l'un des inconvénients de XML est l'augmentation de la taille des données lors du formattage du texte avec des balises open et close. Pour surmonter cet inconvénient, il existe différents formats pour XML, le plus courant étant JSON (*JavaScript Object Notation*) [30]. JSON est conçu pour être un langage d'échange de données lisible par l'homme, et facile à analyser et a utiliser [48] par les machines. Ce langage augmente la vitesse d'envoi des données. Il est particulièrement bien adapté aux applications Web [70]. Dans certains systèmes de capteurs à base de *smartphones*,

JSON est également utilisé comme format pour les données échangées entre un *smartphone* le datacenter [1, 71]. JSON raccourcit les symboles de formatage permettant de stocker des données plus petites que XML, mais faciles à lire par les humains. Voici un exemple simple en JSON :

```
{
    "contact" : {
        "nom" : "Le NGUYEN",
        "phone" : "07-52-34-56-78",
        "adresse" : {
            "numetvoie" : "37 Rue de l'Orge",
            "codepostal" : "91000",
            "ville" : "Evry"
        }
    }
}
```

Notre préoccupation est que l'envoi de données du *smartphone* au centre de calcul consomme trop de bande passante réseau et de batterie en utilisant XML. Nous nous appuyons donc sur JSON pour créer une structure de données permettant d'envoyer des données semi-structurées sur le réseau avec une plus petite taille. Les données envoyées en ligne n'ont pas besoin d'être lues et comprises par l'homme. La taille des données numériques binaires est inférieure à celle des texte littéraires. Les données sur les anomalies contiennent de nombreuses valeurs numériques. Nous définissons donc une structure binaire pour représenter les données semi-structurées à envoyer sur le réseau. Pour faciliter la description, nous appelons ce format BSSDF (*Binary Semi-structured Data Format*). Sur le *smartphone*, les données contenues dans les objets sont converties directement dans le format BSSDF. Ceci permet de conserver le type des données. À la réception, les données sont lues à partir du format binaire et peuvent être converties au format XML ou en d'autres types de données, le format BSSDF étant difficile à manipuler et difficilement lisible par l'homme.

3.4.2 Structure du format de données BSSDF

BSSDF est une structure binaire dans laquelle les noms des balises ne sont utilisés qu'une seule fois dans un nœud. Aucune balise d'ouverture ou de fermeture et aucun caractère <, \ et > n'est requis. De plus, pour économiser de la mémoire, la définition d'alias — plus courts — pour remplacer les balises longues et répétitives est autorisée. Les utilisateurs peuvent définir des alias à une ou deux lettres tant qu'il n'y a pas de duplication dans les balises de données envoyées.

Dans la structure de BSSDF, l'écriture et la lecture doivent être effectuées de manière séquentielle. La fin d'une balise, d'un nœud, d'un attribut ou d'une chaîne est spécifiée par un octet spécial. La terminaison d'un attribut numérique est déterminée par la taille du type de données. Le type de données d'un attribut est spécifié par un octet associé.

Types de données

Le BSSDF actuel prend en charge les types de données suivants :

Tableau 3.1 – Les type de données de BSSDF.

Num	Type	Description
1	`string`	Chaîne de caractères Unicode UTF-8
2	`byte`	Nombre entier en complément à deux sur 8 bits signé
3	`short`	Nombre entier en complément à deux sur 16 bits signé
4	`int`	Nombre entier en complément à deux sur 32 bits signé
5	`long`	Nombre entier en complément à deux sur 64 bits signé
6	`float`	Nombre réel virgule flottante IEEE 754 32 bits
7	`double`	Nombre réel virgule flottante IEEE 754 64 bits
8	`logic`	Nombre entier 8 bits, 0=faux, 1=vrai

Format de balise

Une balise est stockée sous la forme d'une séquence d'octets se terminant par un code de ENDTAG (valeur 0). Le dernier caractère avant le ENDTAG représente le type de la balise, comme indiqué dans le Tab. 3.2.

Pour une balise de type `string`, la lecture de sa valeur se termine par un caractère spécial. Nous utilisons actuellement la valeur caractère `NULL` (0) pour terminer une chaîne. La valeur d'un attribut d'un autre type est stockée immédiatement après la balise d'attribut et est lue en fonction de sa taille.

Par exemple, le type *Anomaly* comprend le champ *Time* (`long`), le champ *Type* (`byte`) et le champ *Position* qui est une structure composée de trois champs : *Latitude* (`double`), *Longitude* (`double`) et *Accuracy* (`float`). Ainsi, un enregistrement d'anomalie a une valeur de *Type* = 2, *Latitude*=48,817456, *Longitude*=2,419799 et *Accuracy*=4,5 est converti en BSSDF en la structure représentée sur la Fig. 3.2. Sur cette figure, les valeurs représentant les caractères sont affichées en bleu, tandis que les

Tableau 3.2 – Les caractères représentent le type de balise dans BSSDF.

Num	Caractère	Description
1	`<`	Début d'un nœud
2	`>`	Fin d'un nœud
3	`&`	Définition d'un alias
4	`*`	Nom de balise correspondant à l'alias avant celui-ci
5	`.`	Fin du document XML
6	`[`	Début d'un tableau
7	`|`	Fin de la structure ou d'un élement d'un tableau
8	`]`	Fin d'un tableau
9	`T`	Attribut de type `string`
10	`L`	Attribut de type `logic`
11	`1..6`	Attributs de type `byte/short/int/long/float/double`

valeurs en rouge indiquent des valeurs numériques. La quantité d'information totale nécessaire au stockage ou à l'envoi de ces données est de 97 octets.

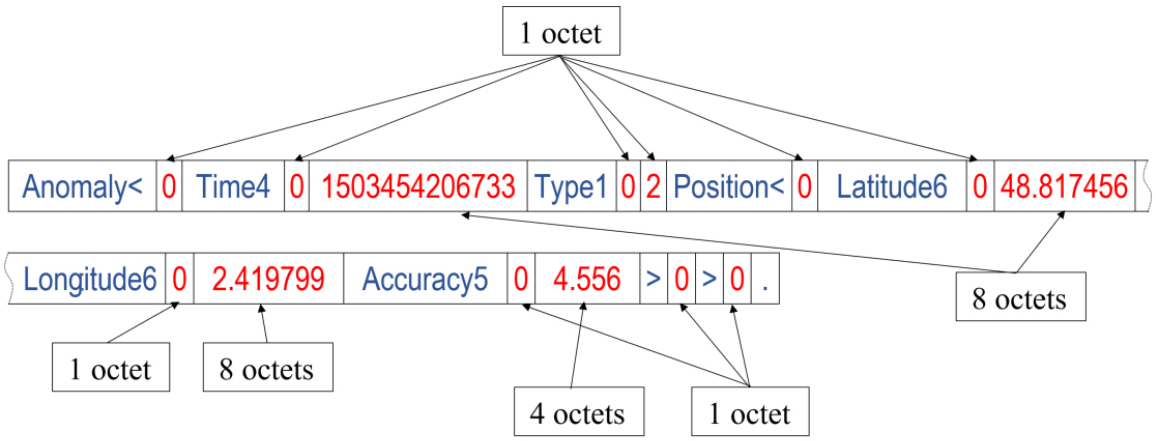


Figure 3.2 – Exemple de BSSDF

Si ces données étaient envoyées au serveur en utilisant le format XML (Cf ci-dessous), 168 octets seraient utilisés (sans compter la première ligne et les sauts de ligne).

```
<?xml version="1.0" encoding="UTF-8"?>
    <Anomaly>
        <Time>1503454206733</Time>
        <Type>2</Type>
        <Position>
```

```
            <Latitude>48.817456</Latitude>
            <Longitude>2.419799<Longitude>
            <Accuracy>4.556</Accuracy>
        </Position>
    </Anomaly>
```

En utilisant JSON pour ces données (Cf ci-dessous), la taille est réduite mais reste à 126 octets sans compter les sauts de ligne.

```
{
    "Anomaly" : {
        "Time" : "1503454206733",
        "Type » : 2,
        "Position" : {
            "Latitude" : "48.817456",
            "Longitude" : "2.419799",
            "Accuracy" : "4.556"
        }
    }
}
```

En fait, les valeurs numériques telles que le temps, les coordonnées géographiques, les valeurs d'accélération sur trois axes, etc sont répétées plusieurs fois dans les données. Par exemple, la valeur numérique 1503454206733 (le temps) prend 8 octets dans BSSDF mais 13 octets si elle est au format texte, comme dans XML ou JSON. Par conséquent, BSSDF contribue à réduire significativement la taille des données.

Utilisation des alias

Si une balise est longue et répétitive, elle peut être remplacée par un alias de taille plus courte. Par exemple, le nom *Longitude* peut être remplacé par un alias *L*. Pour ce faire, au début du document, il suffit de déclarer simplement des balises alias telles que "L&" et "Longitude*" immédiatement après. Ensuite, les balises de nom "L" sont automatiquement remplacées par le nom "Longitude".

Utilisation de tableau

L'autre méthode pour réduire la taille des envois lorsqu'il existe plusieurs objets de même structure et contigus consiste à utiliser un tableau.

Pour utiliser un tableau, il faut commencer par une balise avec uniquement le caractère "[", suivie des balises représentant la structure de l'objet, y compris les attributs, mais sans valeur associée. Une balise "|" doit être ajoutée ensuite. Les données des attributs de la structure élément peuvent être ensuite ajoutées et présentées dans l'ordre de la structure. La fin des données de chaque élément du tableau est une balise "|", sauf pour le dernier élément qui se termine par une balise "]".

Avantages et inconvénients de BSSDF

Comme les données des capteurs telles que l'heure, les coordonnées et les données d'accélération contiennent de nombreux chiffres, l'envoi en binaire réduit considérablement la quantité de données à transmettre. En outre, le BSSDF conserve le type de données, et le serveur peut les convertir en XML , en objets, ou en structures.

Cependant, BSSDF n'est pas facile à lire par les humains. Il n'est également pas pratique d'envoyer des données à partir de XML. Nous ne l'utilisons que pour envoyer des données des *smartphones* au centre afin d'économiser de l'énergie et de la bande passante. À l'avenir, nous élargirons les types de données pour prendre en charge d'autres types de données telles que le son.

3.5 Agrégation et exploration de données

Nous proposons que certaines opérations de détection soient effectuées au niveau du *datacenter*, en particulier l'application du *machine learning* pour classer et éliminer les faux résultats (le composant *Anomaly Classification* du schéma 3.1 en page 28). Les données sont ensuite enregistrées dans la base de données *Raw Anomaly Database* pour l'agrégation. Selon notre méthode d'agrégation (Cf Chap. 5), un processus de partitionnement est exécuté de manière cumulative lorsque des données arrivent dans la base de données. Ces données sont ensuite analysées pour trouver les positions les plus probables des anomalies. Ces positions sont pondérées et sélectionnées pour être transférées dans la base de données *Anomaly Database* afin de servir les applications. La localisation des anomalies est effectuée par un algorithme basé sur

le *Mean Shift* [22]. Le composant *System Control* est chargé de démarrer le composant *Anomaly Aggregation* pour exécuter cet algorithme régulièrement ou en cas de modification des *clusters* de données.

Les applications d'exploration de anomalies peuvent être destinées aux administrateurs ou aux utilisateurs. Il peut s'agir d'applications pour les *smartphones*, d'applications sur l'ordinateur ou d'applications Web. Cela dépend de l'objectif du système.

Un exemple courant est le système de détection d'anomalie. Pour ce système, les applications destinées aux agences gouvernementales pour gérer, réparer et surveiller la détérioration des routes sont essentielles. En outre, une applications utilisés sur *smartphone* et adressée aux conducteurs est également très importante. De plus, il est possible de combiner le programme d'application et le programme de détection pour que les utilisateurs participent ensemble au système de capteurs.

3.6 Notre système plate-forme expérimentale

Afin de mener à bien nos travaux de recherche, nous avons développé une plateforme de collecte et de traitement des données pour les *smartphones* Android, avec un logiciel serveur pour la réception des données et un logiciel pour la conduite des expériences. Tous les logiciels ont été développés en java. La Fig. 3.5 décrit le système. Le logiciel sur *smartphones* remplit de nombreuses fonctions . Sur le diagramme de la Fig. 3.5, elles sont présentées de manière fonctionnelle.

Logiciel embarqué sur smartphone

Le logiciel de collecte des données et de test de la détection a été développé pour système Android dans la mesure où les *smartphones* utilisant ce système sont très populaires, variés et généralement peu onéreux. Le programme peut être exécuté en avant-plan ou en arrière-plan. Des vues du programme sont illustrés à la Fig. 3.4. Le logiciel sur le *smartphone* peut fonctionner en trois modes : le mode acquisition des données des capteurs, le mode détection et le mode test.

En mode acquisition de données , le logiciel peut diviser les données (GPS et accélération) en parties et les envoyer au serveur en utilisant le format BSSDF (voir la Sec. 3.4.2) ou stocker les données au format binaire sur le *smartphone*.

En mode détection, le logiciel embarqué effectue l'acquisition des données et la détection des anomalies routières. Afin d'économiser la batterie, le programme bascule entre différents états, comme illustré à la Fig. 3.5. Chaque fois qu'une anomalie est

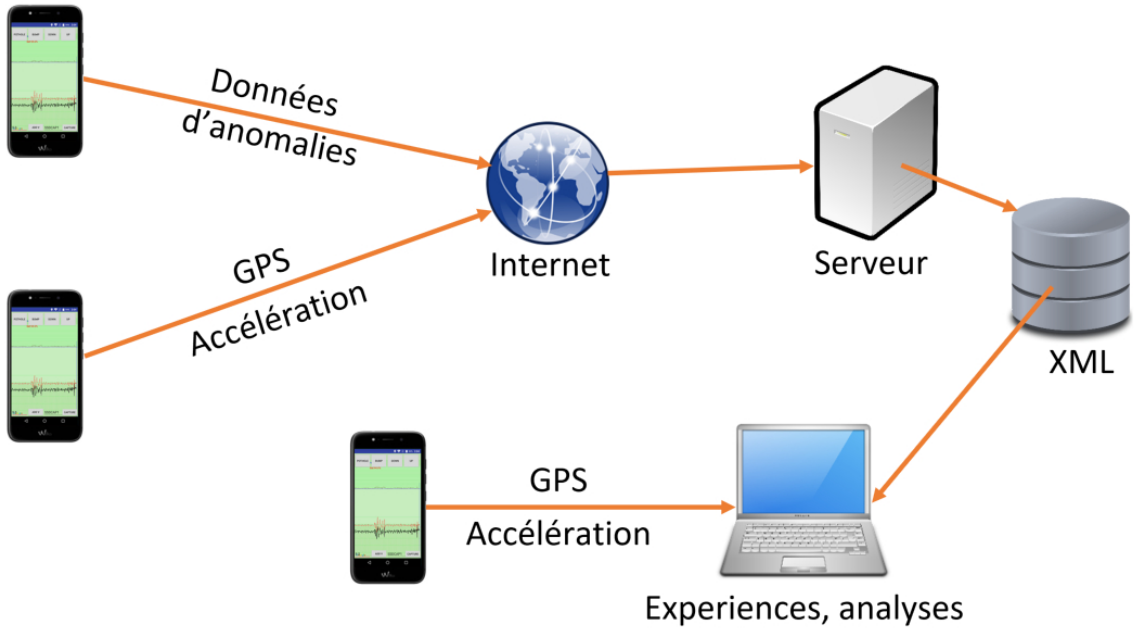


FIGURE 3.3 – Système de capteurs de *smartphones* développé dans le cadre de notre étude.

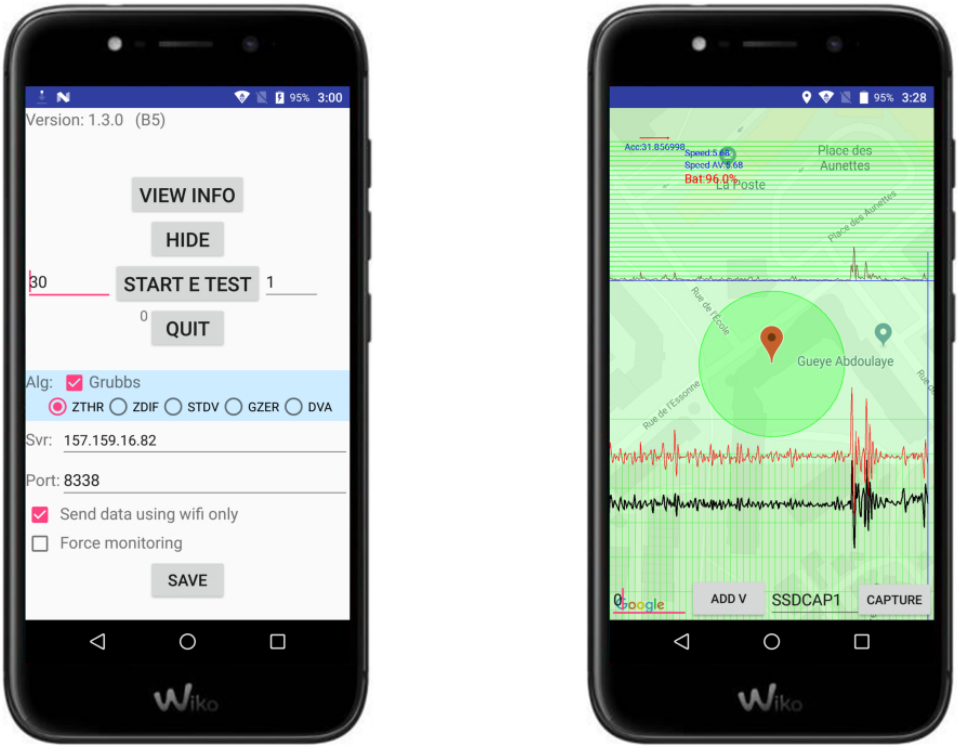


FIGURE 3.4 – Vues du logiciel sur *smartphone*.

détectée, sa position et son segment de données GPS d'accélération sont extraits, stockés temporairement et envoyés au serveur sous format BSSDF lorsque le téléphone est connecté à l'Internet.

Comme son nom l'indique, le mode test est ouvert pour les tests. Par exemple, lors des développements, un mécanisme permet au programme de lire séquentiellement d'anciennes données du capteur sauvegardées dans un fichier, et ainsi remplacer les données réelles fournies par le capteur pour simuler une situation avec des anomalies. De cette façon, il devient possible d'étudier le processus de détection pour divers algorithmes et considérer par exemple la consommation de la batterie.

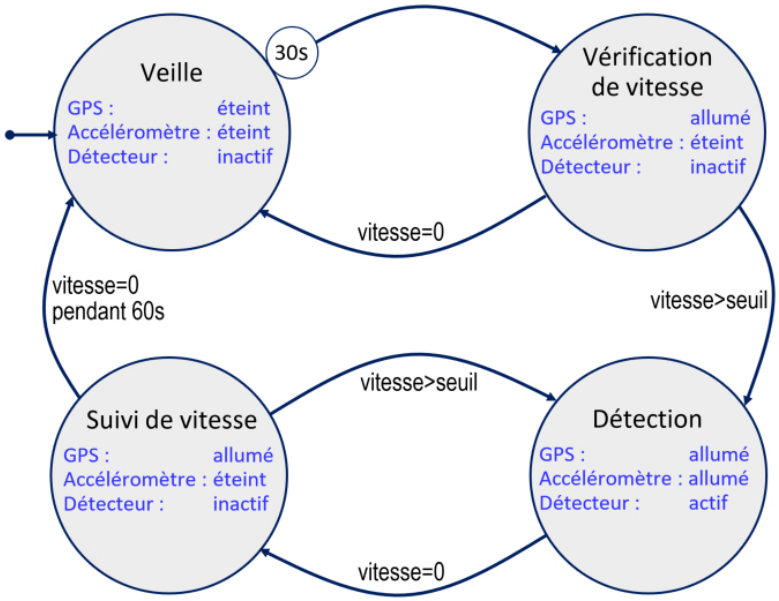


FIGURE 3.5 – Passage entre les états du logiciel sur les *smartphones*.

Le serveur

Notre serveur actuel vise à recevoir les données BSSDF des téléphones et à les stocker sous forme de fichiers XML. Ce programme ne fonctionne qu'en mode ligne de commande (voir la Fig. 3.6).

```
vanlong@tranpc03: ~/dev
[2019/10/09 03:34:57] 1 events saved in /home/vanlong/dev/java/SSDataStation/bin/database/ssd-47c935ad.xml
[2019/10/09 03:35:04] Connection 120 at Socket[addr=/46.193.64.17,port=37272,localport=8338]
[2019/10/09 03:35:04] 1 events saved in /home/vanlong/dev/java/SSDataStation/bin/database/ssd-47c935ad.xml
[2019/10/09 03:35:09] Connection 121 at Socket[addr=/46.193.64.17,port=37273,localport=8338]
[2019/10/09 03:35:09] 1 events saved in /home/vanlong/dev/java/SSDataStation/bin/database/ssd-47c935ad.xml
[2019/10/09 03:35:13] Connection 122 at Socket[addr=/46.193.64.17,port=37274,localport=8338]
[2019/10/09 03:35:13] 2 events saved in /home/vanlong/dev/java/SSDataStation/bin/database/ssd-47c935ad.xml
```

FIGURE 3.6 – Serveur en fonctionnement dans un terminal sous Ubuntu.

Le logiciel expérimental

Le logiciel expérimental a été développé pour remplir les fonctions suivantes : la visualisation et l'ajustement des données ; l'exécution d'expériences telles que la comparaison de méthodes de détection d'anomalies et d'expériences sur les algorithmes d'agrégation de données (voir la Fig. 3.7). Les résultats expérimentaux sont présentés dans l'interface et générés sous forme de textes pour analyse par d'autres outils tels que MATHLAB ou le langage R.

FIGURE 3.7 – Interface du logiciel de test.

3.7 Conclusion

Le but de ce chapitre est de proposer un modèle général pour les systèmes de capteurs sur *smartphones* pour la détection des anomalies. L'architecture du système permet de développer des fonctions et de prendre en compte leurs relations au sein du système. Sur la base de ce système, nous avons développé, amélioré les fonctions de détection, l'agrégation des données et les mécanismes d'économie d'énergie.

Nous avons analysé l'architecture de certains systèmes actuels, à savoir les systèmes de détection d'anomalies et l'architecture générale d'un système de capteur *smartphone*. Sur la base des avantages et des inconvénients de ces systèmes, nous avons reconstruit l'architecture du système de détection des anomalies. Nous avons proposé de séparer la détection en plusieurs étapes, qui peuvent être distribuées entre le *smartphone* et le centre pour limiter la consommation de ressources sur le téléphone.

Comme le système évolue constamment, l'utilisation de structures de données semi-structurées telles que des bases de données XML facilite la mise à jour du système et l'amélioration de la compatibilité. Nous avons proposé également le format BSSDF qui permet d'envoyer des données semi-structurées au centre de calcul et qui peut être converti au format XML ou objet. Le BSSDF minimise les marqueurs. De plus, BSSDF stocke les données sous format binaire. Pour les données des capteurs et des anomalies, les valeurs de temps, de coordonnées, d'accélération etc. sont des nombres assez longs. Le format binaire permet de réduire signficativement la taille des données à envoyer. Cela permet d'économiser aussi sur le coût réseau et la batterie du téléphone.

Dans ce chapitre, nous présentons également le système que nous avons développé pour les expérimentations. Ce système démontre nos méthodes de collecte de données et d'expériences.

À l'avenir, nous allons continuer à améliorer le modèle de notre système, en utilisant par exemple l'apprentissage automatique pour améliorer la qualité de la détection. En outre, pour les systèmes de grande taille, un modèle distribué peut être utilisé pour réduire la charge du serveur.

4 Amélioration des méthodes de détection des anomalies

4.1 Introduction

De nombreuses études sur les capteurs présents sur les *smartphones* ont été décrites dans la littérature [3, 27, 32]. Les domaines d'application courants sont les suivants : surveillance de la santé des personnes, calcul de l'impact sur l'environnement, détection des chutes, détection de la marche, surveillance de la circulation, surveillance du bruit et de l'ambiance, etc.

Il est possible de regrouper des domaines tels que la surveillance des conditions de route et de circulation, la surveillance du bruit et la détection des ambiances et des chutes sous le chapeau des systèmes de surveillance et de détection des anomalies. À l'avenir, de nombreuses autres applications pourront être exploitées grâce aux systèmes de surveillance et de détection des anomalies basés sur les données de capteurs tels que les capteurs de lumière ambiante, les magnétomètres, les baromètres, les capteurs d'humidité de l'air et les thermomètres. Le traitement de base des systèmes de surveillance et de détection des anomalies est le suivant : a) les données des capteurs sont collectées régulièrement ; b) ces données contiennent des anomalies telles que des anomalies de la route, des chutes ou des sons inhabituels ; c) s'il est nécessaire de calculer les conditions environnementales, telles que la planéité de la chaussée ou le bruit, en fonction du type d'applications, toutes les données ou uniquement la partie restante des données sont analysées ; d) tous les résultats finaux requis sont envoyés au serveur pour traitement ultérieur.

Dans ce chapitre, nous proposons une application des tests statistiques des valeurs aberrantes à la détection des anomalies environnementales. Dans la section suivante, nous nous concentrons uniquement sur l'amélioration et l'expérimentation de méthodes de détection des anomalies routières basées sur les données d'accélération (les méthodes basées sur la vibration). Cependant, le principe de cette méthode peut

également être appliquée à d'autres types d'anomalies.

4.2 Améliorations de la détection des anomalies

4.2.1 Nécessité d'améliorer la détection

Comme montré dans l'analyse de la Sec. 2.3, les méthodes de détection actuelles, basées sur des seuils, ne répondent pas bien aux conditions réelles telles que la non-planéité de la chaussée, la vitesse, le type de véhicule, la qualité du système de suspension, le type de *smartphone*, etc. Les méthodes basées sur une classification sont trop exigeantes en termes de ressources. Dans le but de construire une méthode de détection des anomalies légère et qui réponde toujours bien aux conditions réelles, nous avons amélioré des méthodes basées sur des seuils afin qu'elles répondent mieux aux conditions réelles. Le terme *conditions réelles* implique une combinaison de nombreux facteurs différents. La construction de paramètres adaptés à tous les facteurs est très compliquée. Par conséquent, nous proposons une solution commune consistant à détecter les anomalies en appliquant des méthodes statistiques pour détecter des données aberrantes (outliers) par rapport aux données de mesure antérieures et postérieures. Ces données peuvent être considérées dans les mêmes conditions si la vitesse n'a pas trop changé. Les changements de vitesse peuvent être vérifiés à l'aide des données GPS.

Nous avons choisi d'appliquer la méthode de test Grubbs. Il s'agit d'une méthode de test des valeurs aberrantes très populaire pour les jeux de données univariés. En particulier, l'application de cette méthode permet de réduire la complexité des calculs.

Cette solution a été appliquée et validée sur des algorithmes de détection d'anomalies routières. Cependant, cette idée est suffisamment générale pour également être utilisée pour détecter des anomalies basées sur des types de données tels que l'accélération, le son ou le magnétisme.

4.2.2 Le test de Grubbs

Le test de Grubbs [26] est un test statistique utilisé pour détecter les valeurs aberrantes dans un jeu de données univarié.

Le test suppose que la distribution des données suive une loi normale. Le test de Grubbs est défini pour les hypothèses suivantes :

- H_0 (hypothèse nulle) : il n'y a pas de valeur aberrante dans l'échantillon ;
- H_1 (hypothèse alternative) : il y a au moins une valeur aberrante dans l'échantillon.

Pour le test bilatéral, cela signifie qu'une valeur est aberrante si elle est trop grande ou trop petite. La statistique du test de Grubbs est définie comme suit :

$$C = \frac{max_i|x_i - \overline{x}|}{\sigma}$$

où $\overline{x}$ et σ désignent respectivement la moyenne et l'écart-type de l'échantillon.

Dans le cas d'un test unilatéral, cela signifie de ne tester qu'un seul cas : les valeurs trop petites sont aberrantes (test unilatéral à gauche) ou seule les valeurs trop grandes sont aberrantes (test unilatéral à droite). La statistique du test de Grubbs est définie dans les deux cas : pour le test unilatéral à gauche

$$C = \frac{\overline{x} - min_i(x_i)}{\sigma}$$

et pour le test unilatéral à droite

$$C = \frac{max_i(x_i) - \overline{x}}{\sigma}$$

Soit α le niveau de signification (*significance level*) du test, c'est-à-dire la probabilité qu'une valeur aberrante existe lorsque l'hypothèse nulle est vraie. Pour le test bilatéral, l'hypothèse H_0 est rejetée si :

$$C > G_{N,\alpha} = \frac{N-1}{\sqrt{N}} \sqrt{\frac{t^2_{\alpha/(2N),N-2}}{N-2+t^2_{\alpha/(2N),N-2}}}$$

où $t^2_{\alpha/(2N),N-2}$ désigne la valeur critique supérieure de la t-distribution avec des degrés de liberté $N-2$ et un niveau de confiance de $\alpha/(2N)$. Pour le test unilatéral, $\alpha/(2N)$ est remplacé par α/N [24].

4.2.3 Application du test de Grubbs pour la détection

Nous avons appliqué la méthode Grubbs aux cinq algorithmes basés sur un seuil suivants : Z-THRESH, Z-DIFF, STDEV(Z), G-ZERO et DVA-THRESH.

Afin d'appliquer le test de Grubbs, nous avons comparé les composants des données d'accélération selon la formule de Grubbs au lieu de les comparer à un seuil fixe. Les données du capteur d'accélération sont fournies sous forme d'une séquence notée

$< t^i, a_x^i, a_y^i, a_z^i >$ où t^i est le temps réel et a_x^i, a_y^i, a_z^i sont les trois composantes du vecteur d'accélération à l'instant t^i. Chacun des algorithmes ci-dessus utilise une valeur calculée ou extraite à partir des données pour comparer avec le seuil. Nous appelons cette valeur *valeur comparative*. Par exemple, la valeur comparative de l'algorithme Z-THRESH est a_z^i et celle de l'algorithme Z-DIFF est $a_z^i - a_z^{i-1}$. Notre méthode consiste à calculer et retenir les N dernières valeurs comparatives dans un vecteur X : l'échantillon de la méthode Grubbs. Le Tab. 4.1 montre la dernière valeur comparative et la condition pour qu'une anomalie soit détectée pour chaque algorithme après amélioration. La formule de test bilatéral est appliquée à Z-THRESH et le test unilatéral est appliqué aux autres algorithmes.

Tableau 4.1 – La dernière valeur valeur comparative correspondant aux algorithmes.

Algorithme	**Valeur X_N**	**Condition d'anomalie**
Z-THRESH	a_z^{last}	$X_i > \overline{X} + \sigma * G_{N,\alpha}$ *or* $X_i < \overline{X} - \sigma * G_{N,\alpha}$
Z-DIFF	$\lvert a_z^{last} - a_z^{last-1} \rvert$	$X_i > \overline{X} + \sigma * G_{N,\alpha}$
STDEV(Z)	$\text{stddev}(a_z^{last-winSize}..a_z^{last})$	$X_i > \overline{X} + \sigma * G_{N,\alpha}$
G-ZERO	$\max(a_x^{last}, a_y^{last}, a_z^{last})$	$X_i < \overline{X} - \sigma * G_{N,\alpha}$
DVA-THRESH	$(a_{z\_max} - a_{z\_min})^{\text{dernière sec}}$	$X_i > \overline{X} + \sigma * G_{N,\alpha}$

Pour la détection des anomalies en temps réel, lorsqu'un nouvel ensemble de données d'accélération $(t^i, a_x^i, a_y^i, a_z^i)$ arrive, le programme met à jour l'échantillon X en ajoutant la nouvelle valeur X_{new} et en supprimant la valeur la plus ancienne X_{old}. La moyenne $\overline{X}$ et l'écart-type σ de l'échantillon X sont également recalculés. Pour éviter d'augmenter considérablement le temps de traitement et afin de réduire la complexité algorithmique du programme, $\overline{X}$ et σ sont recalculés cumulativement. Comme $\sigma = \sqrt{\overline{X_i^2} - \overline{X}^2}$, où $\overline{X}$ est la moyenne et $\overline{X^2}$ est la moyenne des carrés des valeurs dans X, le programme peut recalculer $\overline{X}$ et σ sans utiliser de boucle en mettant à jour $\overline{X}$ et $\overline{X^2}$ à l'aide des formules suivantes :

$$\overline{X} = \overline{X} + \frac{X_{new} - X_{old}}{N}$$

$$\overline{X^2} = \overline{X^2} + \frac{X_{new}^2 - X_{old}^2}{N}$$

Avec ces optimisations, l'application du test de Grubbs n'augmente pas de manière significative la complexité de calcul. Ainsi, les méthodes basées sur des seuils ont une

complexité de $O(n)$ même en utilisant le test de Grubbs.

4.2.4 Adaptation à l'orientation aléatoire

Comme l'orientation du *smartphone* est aléatoire, les trois composantes des données d'accélération doivent être calculées en fonction des axes de coordonnées rattachés au véhicule (voir la Fig. 4.1).

FIGURE 4.1 – Le système de coordonnées de la voiture et du téléphone ne coïncide pas.

Comme analysé en Sec. 2.3, la détermination des axes X et Y est très complexe et parfois impossible. En revanche, la détermination de la direction Z est toujours possible par gravité. Or, les algorithmes Z-THRESH, Z-DIFF, STDEV(Z) et DVA-THRESH n'ont besoin que de la composante a_Z des données de l'accéléromètre. Par conséquent, l'amélioration des algorithmes présentés ci-dessus est une solution tout à fait réalisable.

Pour calculer la composante verticale a_Z des données d'accélération, nous proposons une méthode simple consistant à calculer la projection des données d'accélération sur la verticale déterminée par la pesanteur. Chaque donnée d'accélération est un vecteur. Supposons que $\overrightarrow{u}(u_x, u_y, u_z)$ soit une valeur de l'accélération lue quand le téléphone est immobile. En théorie, $\overrightarrow{u} = -\overrightarrow{g}$, où $\overrightarrow{g}$ est la pesanteur. Supposons que $\overrightarrow{a}(a_x, a_y, a_z)$ soit une valeur lue en mouvement. Les composantes $u_x, u_y, u_z, a_x, a_y, a_z$ sont en coordonnées xyz sur le *smartphone*. Ensuite, la composante verticale a_Z de $\overrightarrow{a}$ est calculée par la formule suivante :

$$a_Z = |\text{proj}_{\overrightarrow{u}} \overrightarrow{a}| = \frac{\overrightarrow{a}.\overrightarrow{u}}{|\overrightarrow{u}|} = \frac{a_x u_x + a_y u_y + a_z u_z}{\sqrt{u_x^2 + u_y^2 + u_z^2}}$$

Dans [4] et [44], les auteurs proposent d'identifier $\overrightarrow{u}$ une fois que le téléphone est à l'état stationnaire. Sachant que la rotation initiale du téléphone peut changer pendant le déplacement du véhicule, nous proposons de remplacer $\overrightarrow{u}$ par une valeur estimée de $\widehat{\overrightarrow{u}}$ calculée comme la valeur moyenne des données d'accélération sur une période de temps autour de la position à prendre en compte, lorsque le véhicule roule à une vitesse relativement constante. Lorsque le téléphone est fixe et que la voiture tourne régulièrement, le vecteur d'accélération oscille autour de la position de $\overrightarrow{u}$.

4.2.5 Exclusion des actions de l'utilisateur

Un autre élément très important, souvent omis dans la littérature est l'exclusion des actions entreprises par l'utilisateur du *smartphone*. Prashanth Mohan et al. [44] sont capables de détecter les interactions des utilisateurs en recherchant les touches enfoncées, les événements de souris et/ou les appels téléphoniques. Cependant, l'utilisateur peut déplacer le *smartphone* sans taper de touche et/ou toucher l'écran. Afin de surmonter ce problème, il semble nécessaire de prendre en compte les variations du vecteur d'accélération pour identifier les mouvements inhabituels du dispositif.

Nous suggérons d'ajouter une méthode pour éliminer les cas où le téléphone change d'angle ou tombe. Après avoir détecté des anomalies, les données sont analysées avant, pendant et après l'événement pour voir si la direction du vecteur accélération a été modifiée. Si le cosinus de l'angle de changement dépasse une valeur spécifiée, cette anomalie est exclue. Notons que seule l'amplitude maximale de l'accélération doit être prise en compte car il existe de très grandes fluctuations qui inversent la direction du vecteur d'accélération, auquel cas les valeurs intermédiaires peuvent avoir des écarts de racine très importants. En outre, une anomalie est également éliminée lorsque le *smartphone* tombe. La chute d'un *smartphone* est déterminée par la valeur d'accélération moyenne qui devient inférieure à une valeur spécifiée.

4.2.6 Expériences et résultats

Les proposition présentées dans la section précédente ont été implémentées et validées lors d'expérimentations en vraie grandeur.

Collection des données

Afin de collecter les données pour nos expériences, nous avons développé un logiciel pour *smartphone* qui lit les données de deux capteurs : l'accéléromètre et le capteur

GPS. Le logiciel a été installé sur un *smartphone* Wiko WIM Lite fixé sur un véhicule. L'application enregistre en permanence trois types de données différents : l'accélération, la vitesse et la position. La vitesse et la position sont fournies par le capteur GPS. Les informations collectées sont fournies sous forme d'un ensemble de trois séquences :

< `temps, accélération 3-axes` >

< `temps, location` >

< `temps, vitesse` >

Les données ont été recueillies sur un segment routier d'environ 40 km de rues dans la ville de Hue au Vietnam qui contient de nombreux nids-de-poule (voir les Fig. 4.2 et 4.3). Ce segment de route est composé de plusieurs parties de planéité différentes. Nous avons roulé à différentes vitesses allant de 10 km/h à 50 km/h. Nous avons utilisé deux types de véhicules : une voiture Mazda 3 et un scooter Honda Lead 125 (voir la Fig. 4.4).

FIGURE 4.2 – Parcours de collecte de données dans la ville de Hué.

De plus, afin d'évaluer l'efficacité des algorithmes, des informations supplémentaires sont nécessaires pour établir le *ground truth*, c'est-à-dire de marquer les anomalies routières pour les comparer a posteriori aux résultats du détecteur. De ce fait, quatre boutons ont été ajoutés dans l'application de collection de données pour marquer quatre types d'anomalie routières différentes : nid-de-poule, dos-d'âne, rampant et saillie (voir la Fig. 4.5). Comme, nos expériences ne détectent que des anomalies,

FIGURE 4.3 – Quelques images d'anomalies routières à Hué.

FIGURE 4.4 – Types de véhicule utilisés pour collecter des données à Hué : Mazda 3 et Honda Lead 125.

sans distinguer leur type, lorsque la voiture rencontre une anomalie routière, l'opérateur appuie sur le bouton correspondant au type d'anomalie pour marquer son emplacement. Nous espérons, dans le cadre de travaux ultérieurs, avoir la possibilité de discriminer entre les différents types d'anomalies. La position des anomalies est marquée en temps réel et est sauvegardée avec les autres informations. Au cours des

expériences sur ce segment routier, 217 anomalies routières ont été marquées avec la voiture et 219 anomalies ont été marquées avec le scooter. Nous avons développé un programme permettant d'analyser et reproduire ces expériences. Ce programme permet également d'ajuster le *ground truth* sur ordinateur après la collection des données (voir la Fig. 4.6).

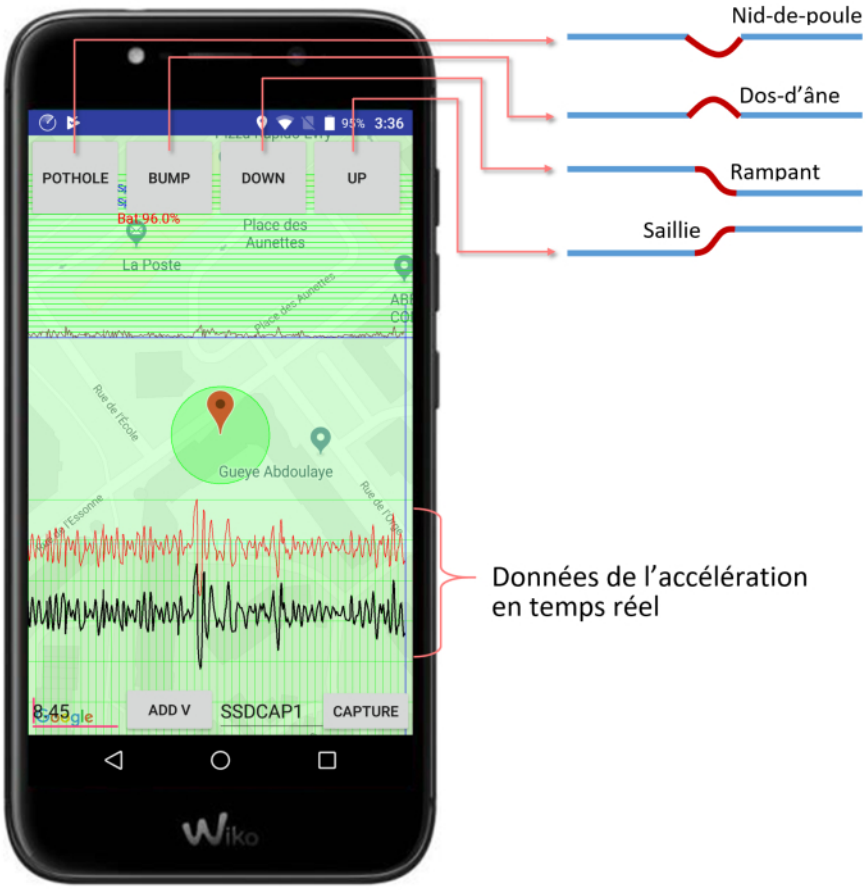


FIGURE 4.5 – Vue de l'application de collecte des données avec les boutons permettant de marquer les anomalies.

Protocole expérimental

Lors de nos expériences, les données d'accélération ont été traitées comme une séquence de données continue, afin de simuler une détection d'anomalie en temps réel. Une fois qu'une anomalie a été détectée, l'étape suivante consiste à séparer le segment contenant les données de l'accélération associé à l'anomalie (voir la Fig. 4.7).

La comparaison entre les anomalies détectées et les *ground truth* est basée sur le temps mais pas sur la position GPS. Lorsqu'une anomalie est détectée, le segment de données correspondant est comparé aux données de *ground truth*. S'il y a une

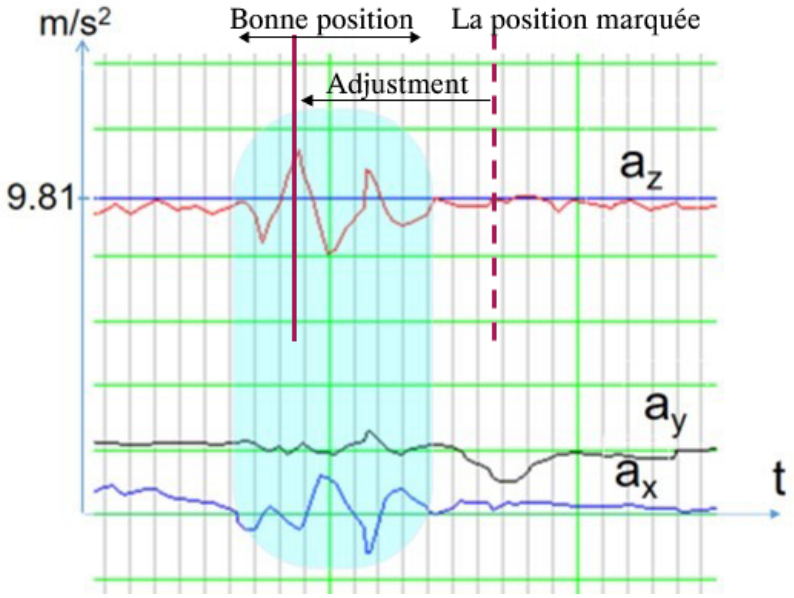


FIGURE 4.6 – Ajustment du *ground truth*.

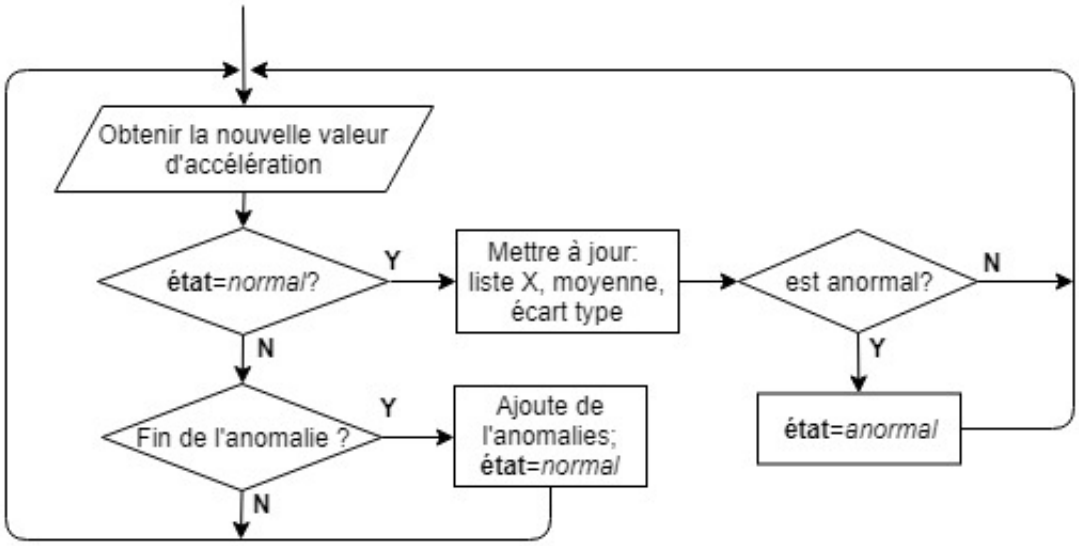


FIGURE 4.7 – Traitement des données d'accélaration.

anomalie, il s'agit d'un *True Positive* (TP), sinon c'est un *False Positive* (FP). Une anomalie non détectée est un *False Negative* (FN). Dans cette expérience, le nombre de conditions négatives détectées, également connu sous le nom de *True Negative* (TN), ne peut pas être déterminé. En effet, les données ont été traitées comme une séquence continue et n'ont pas été coupées en fenêtres. Par conséquent, le nombre de cas négatifs est indénombrable.

Cinq algorithmes différents ont été appliqués sur le même processus de détection d'anomalie. Pour chaque algorithme. Le seuil a été automatiquement incrémenté dans

la plage appropriée et le programme de test a compté les valeurs TP, FN et FP. Ensuite, le test de Grubbs a été ajouté à ces algorithmes avec un ajustement du niveau de signification α. La taille de l'échantillon pour le test de Grubbs que nous avons utilisée est $N = 100$. Les tests montrent que cette valeur est assez bonne. Lorsque N est trop petit, la méthode Grubbs réduit la précision ; en revanche si N est trop grand, elle réduit l'adaptabilité aux conditions de la route et à la vitesse.

Base d'évaluation : courbe de précision-rappel et F-mesure

Les résultats expérimentaux peuvent être résumés de manière appropriée dans une matrice de confusion comme celle présentée au Tab. 4.2 :

Tableau 4.2 – Matrice de confusion.

	Vrai condition positive	Vrai condition négative
Condition prédite positive	TP	FP
Condition prédite négative	FN	TN

Certains indices peuvent être utilisés pour évaluer des algorithmes sur la matrice de confusion. L'évaluation peut être généralisée sur le domaine variable du paramètre (la valeur de seuil pour l'algorithme d'origine et la valeur α pour la version améliorée de l'algorithme avec l'application du test de Grubbs) ou sur un cas concret, en fonction du paramètre. Une vue générale sur le domaine de paramètre est le plus souvent réalisée à l'aide de la courbe ROC *(Receiver Operating Characteristic)* [52] et de la courbe précision-rappel [5]. Cependant, la courbe ROC nécessite la valeur de TN. Nous utilisons donc uniquement la courbe précision-rappel basée sur Précision et Rappel qui sont calculées à partir de TP, FP et FN. L'indice *Average Precision* [73] basée sur cette courbe est également utilisée pour évaluer quantitativement les algorithmes. En outre, un autre indice très populaire, la F-mesure, est également utilisé pour comparer les résultats des algorithmes. Un aperçu de ces indices est présenté ci-dessous.

Les indices que nous utilisons sont basés sur deux valeurs : la *précision* (ou valeur *prédictive positive*) et le *rappel* (ou *sensibilité*). À partir des valeurs de la matrice de confusion, la précision (P) et le rappel (R) sont calculés comme suit [25] :

$$P = \frac{TP}{TP + FP} \quad \text{and} \quad R = \frac{TP}{TP + FN}$$

Le F_1 *score*, également appelé *F-mesure*, est la moyenne harmonique de la précision et du rappel [59] :

$$F_1 = 2.\frac{P.R}{P+R}$$

Une autre mesure mérite d'être notée et prise en compte dans le cadre de cette analyse. La zone située sous la courbe précision-rappel, également appelée précision moyenne (*Average Precision – AP*) [73], est utile pour identifier l'efficacité globale d'un algorithme :

$$AP = \int_0^1 P(R)\, dR$$

Les valeurs P et R étant des valeurs discrètes, la précision moyenne peut être réduite à une simple somme. Soient R_i et P_i, respectivement la précision et le rappel correspondant au $i^{ème}$ paramètre ($i = 1..n$), AP devient :

$$AP = \sum_{i=1}^{n} (R_i - R_{i-1}) P_i$$

Résultats et analyses

Pour les algorithmes améliorés, nous avons réalisé des expériences avec une variable α variant dans l'intervalle (0,1). Quant aux algorithmes originaux, avant les expériences officielles, nous avons effectué une expérience pour sélectionner l'intervalle approprié pour le seuil. Nous utilisons trois ensembles de données différents : les données collectées parla voiture, les données collectées par le scooter et l'ensemble de toutes les données collectées (données agrégées de ces deux ensembles).

Évaluation de l'efficacité

Notre première expérience visait à comparer les algorithmes améliorés et les algorithmes originaux en termes d'efficacité. Cette expérience a été réalisée sur toutes les données collectées.

L'analyse des résultats sur les courbes de précision-rappel a montré que les algorithmes donnaient de meilleurs résultats lorsqu'ils étaient améliorés avec le test de Grubbs. La Fig. 4.8 montre les courbes précision-rappel des cinq algorithmes originaux et des cinq algorithmes améliorés. Idéalement, s'il n'y avait pas d'erreur, P et R devraient être égaux à 1, c'est-à-dire que plus une courbe est proche du coin supérieur droit, meilleur est le résultat. Comme le montre la Fig. 4.8, les graphes des algorithmes améliorés sont tous au-dessus des courbes associées aux algorithmes originaux et plus proches

du coin supérieur droit.

La Fig. 4.9 montre les valeurs de précision moyenne des algorithmes pour les deux cas : avec et sans l'application de la méthode Grubbs. Notez que certaines courbes ne pouvant pas être entièrement dessinées de 0 à 1 avec la variable R, la surface sous les courbes a été calculée dans l'intervalle [0.2, 0.8] pour R. La précision moyenne calculée des algorithmes améliorés est nettement supérieure à celle des algorithmes d'origine.

La Fig. 4.10 montre les graphiques de F-mesure obtenus à partir des expériences. On peut voir que toutes les courbes sont maximisées pour une valeur du seuil ou de α.

La Fig. 4.11 permet de comparer la valeur maximale de F-mesure entre les deux versions des algorithmes. Ce graphique à barres montrent également que l'application du test de Grubbs donne de biens meilleurs résultats en considérant l'indice de F-mesure. L'algorithme Z-THRESH reste la meilleure solution lorsque le test de Grubbs est appliqué, suivi de STDDEV(Z) et de DVA-THRESH. Les résultats détaillés lorsque la F-mesure est optimale sont présentés dans le Tab. 4.3 et le Tab. 4.4.

Tableau 4.3 – Valeur optimale de F-mesure pour les algorithmes originaux.

Algorithm	Seuil	TP	FN	FP	R	P	F1
Z-THRESH	3.70	224	212	171	0.51	0.57	0.539
Z-DIFF	4.01	234	202	260	0.54	0.47	0.503
STDEV(Z)	2.08	209	227	196	0.48	0.52	0.497
G-ZERO	7.33	268	168	417	0.61	0.39	0.478
DVA-THRESH	6.67	209	227	128	0.48	0.62	0.541

Tableau 4.4 – Valeur optimale de F-mesure pour les algorithmes améliorés.

Algorithm	α	TP	FN	FP	R	P	F1
Z-THRESH	0.9886	325	111	113	0.75	0.74	0.744
Z-DIFF	0.9998	279	157	187	0.64	0.60	0.619
STDEV(Z)	0.9987	305	131	172	0.70	0.64	0.668
G-ZERO	0.9621	259	177	157	0.59	0.62	0.608
DVA-THRESH	0.9606	314	122	222	0.72	0.59	0.646

Évaluation de l'adaptation aux conditions réelles

Nous avons mené l'expérience suivante avec l'algorithme Z-THRESH, à la fois sur la version originale et sur la version améliorée. Chaque algorithme était testé sur trois ensembles de données différents : les données collectées par voiture, les données collectées par scooter et l'ensemble de toutes les données collectées.

Les courbes précision-rappel de la Fig. 4.12 montrent que :

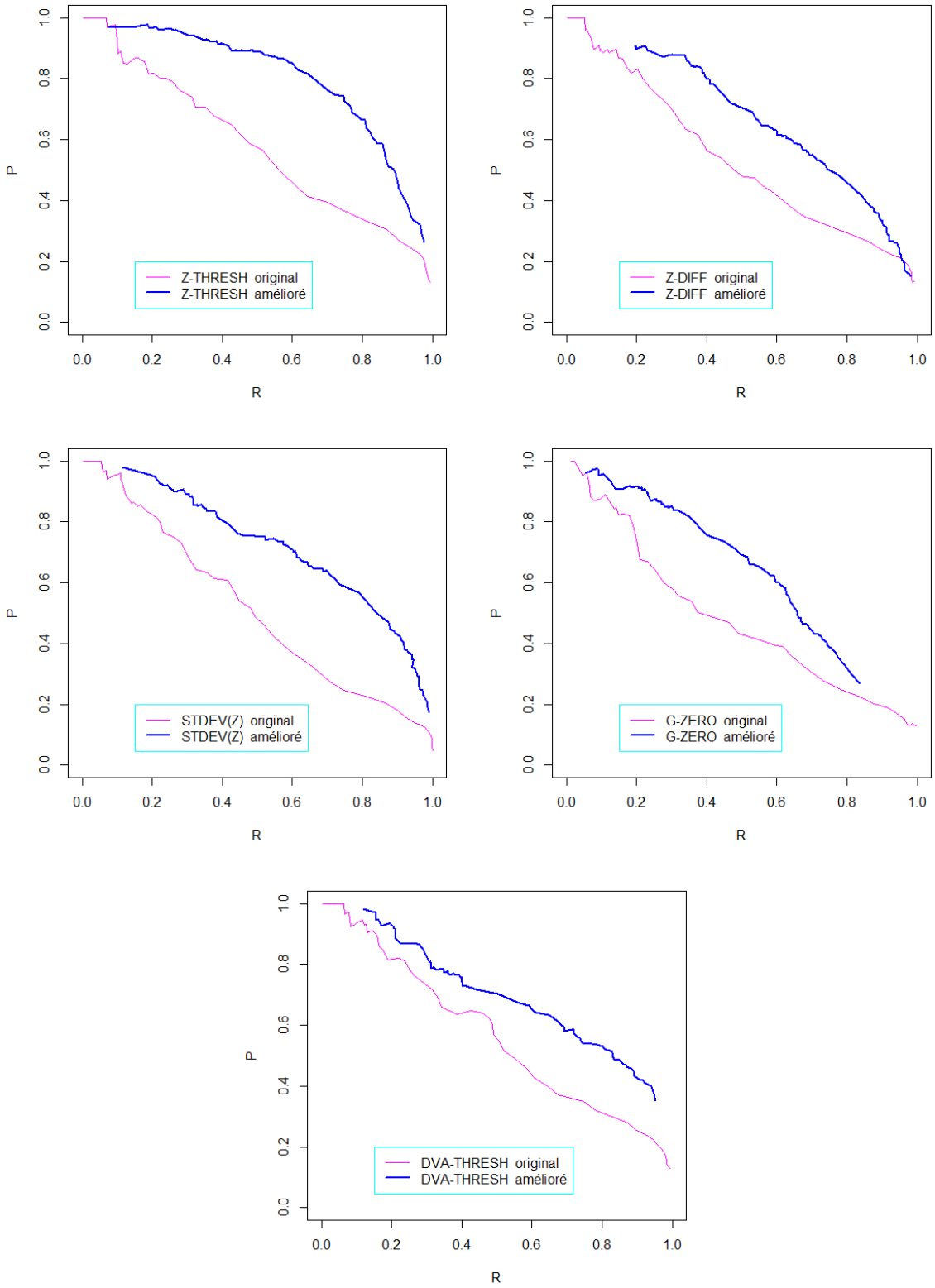


FIGURE 4.8 – Les courbes précision-rappel correspondent aux deux versions des cinq algorithmes.

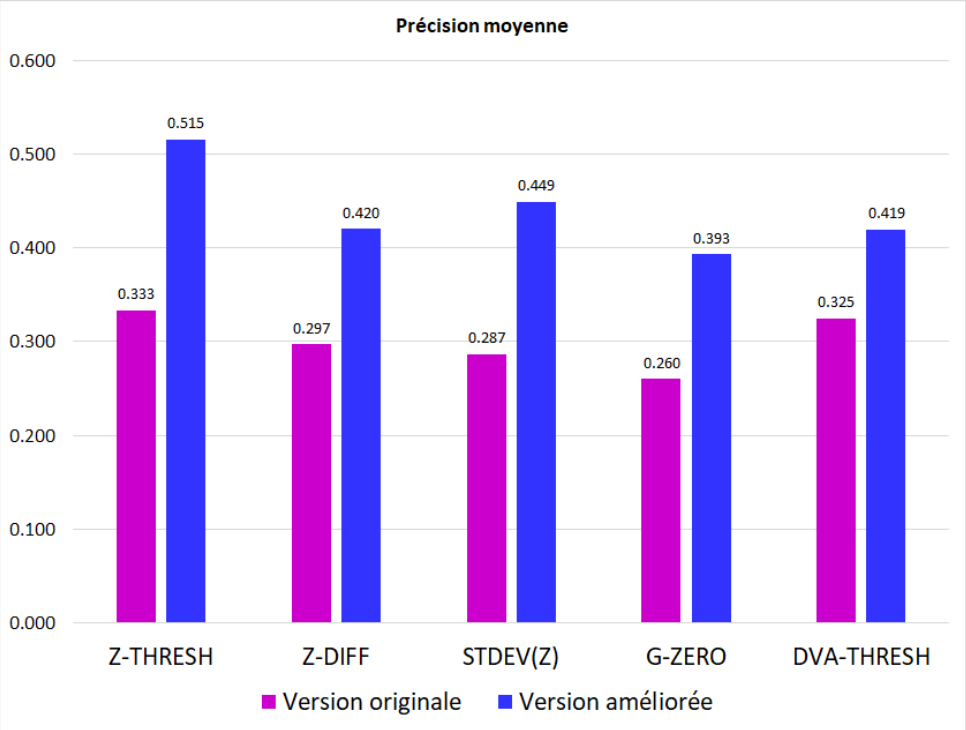


FIGURE 4.9 – Les valeurs de précision moyenne des algorithmes avec R dans l'intervalle [0.2, 0.8].

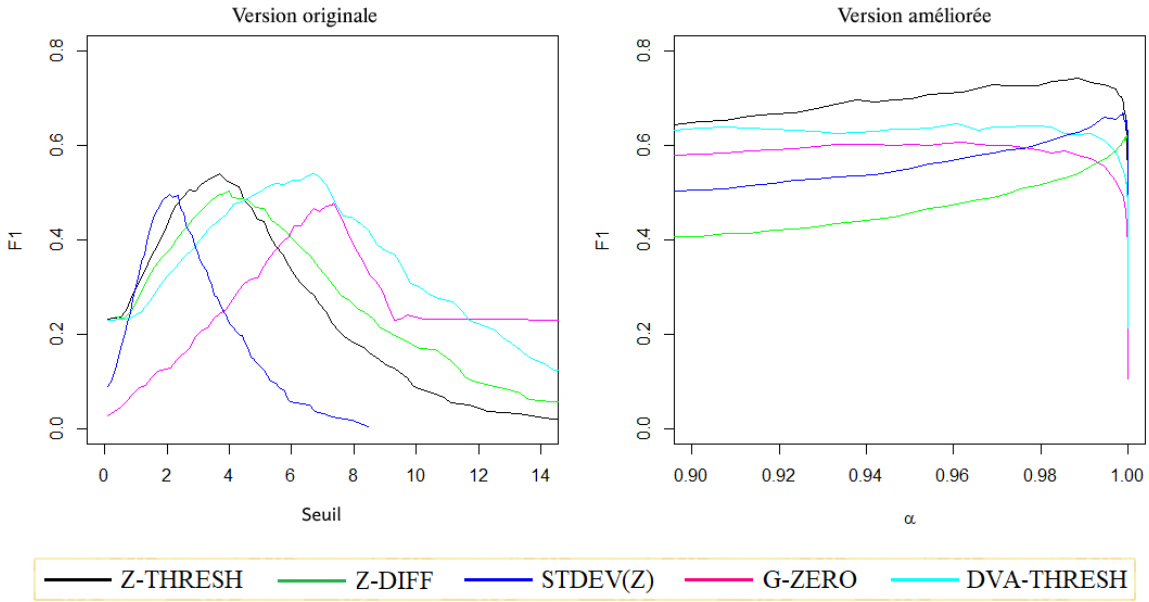


FIGURE 4.10 – Les graphiques de F-mesure.

- la performance de l'algorithme est considérablement réduite lorsque les données sont plus diverses, ce qui correspond à la courbe magenta (données totales) ;
- lorsque la méthode Grubbs est utilisée, les résultats sont meilleurs et tous les

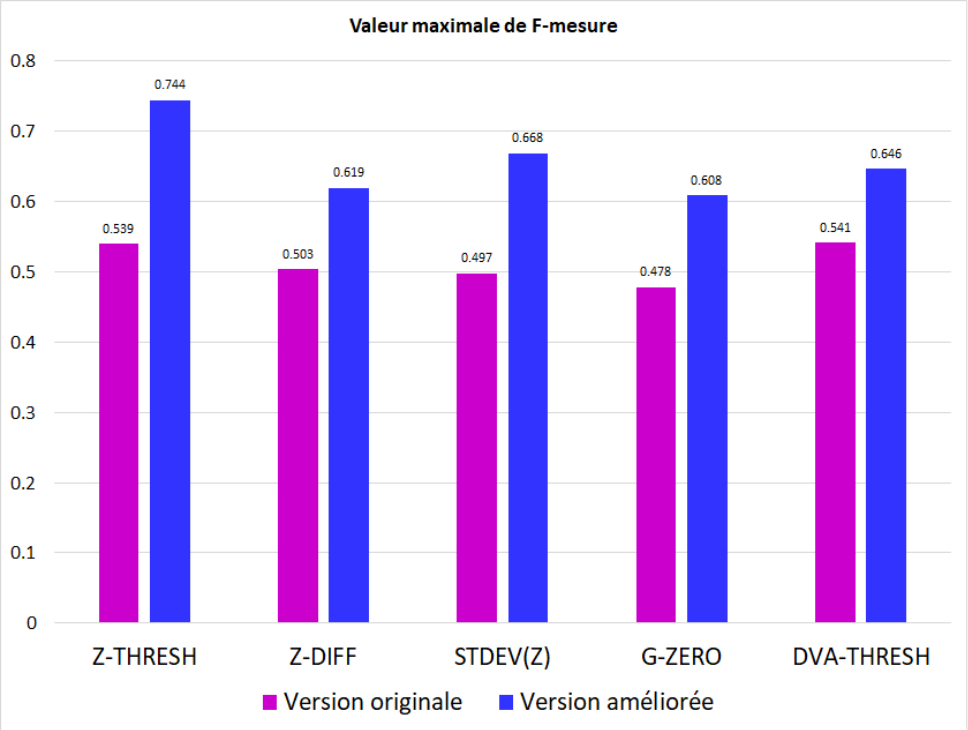


FIGURE 4.11 – Valeur maximale de la F-mesure.

ensembles de données fournissent des résultats similaires. Cela démontre que l'algorithme amélioré est très peu affecté par les conditions réelles.

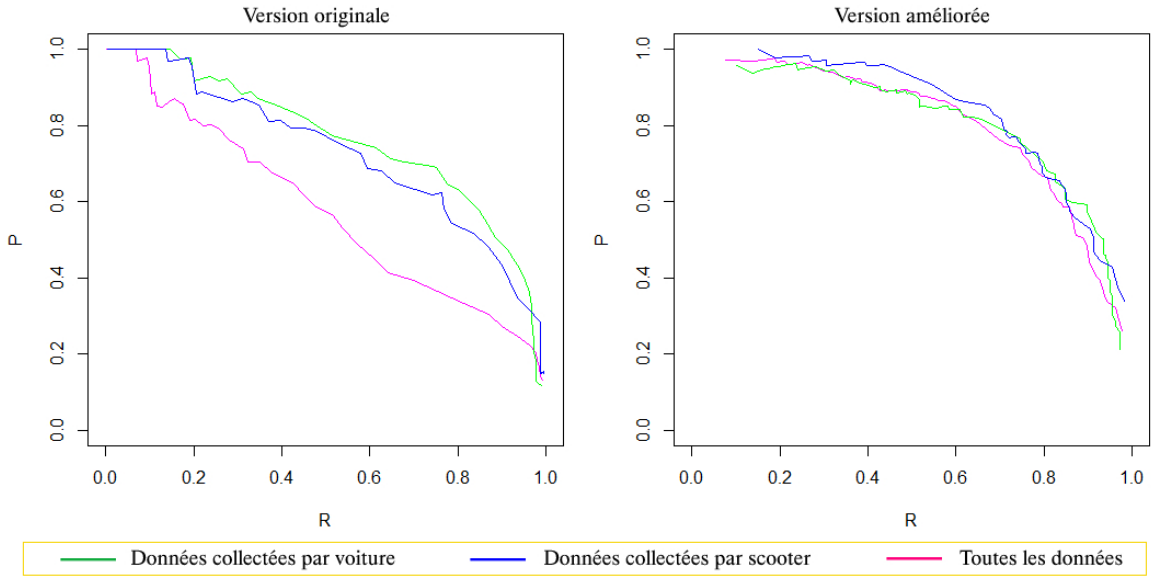


FIGURE 4.12 – Les courbes précision-rappel de l'algorithme Z-THRESH sur les trois ensembles de données.

La Fig. 4.13 considère l'indice F-mesure. Pour l'algorithme original, les fonctions de F-mesure atteignent un maximum à des valeurs de seuil très différentes pour le cas

avec la voiture et le cas avec le scooter. Au contraire, lorsque le test de Grubbs est utilisé, la fonction de F-mesure atteint son maximum à des valeurs assez proches de α.

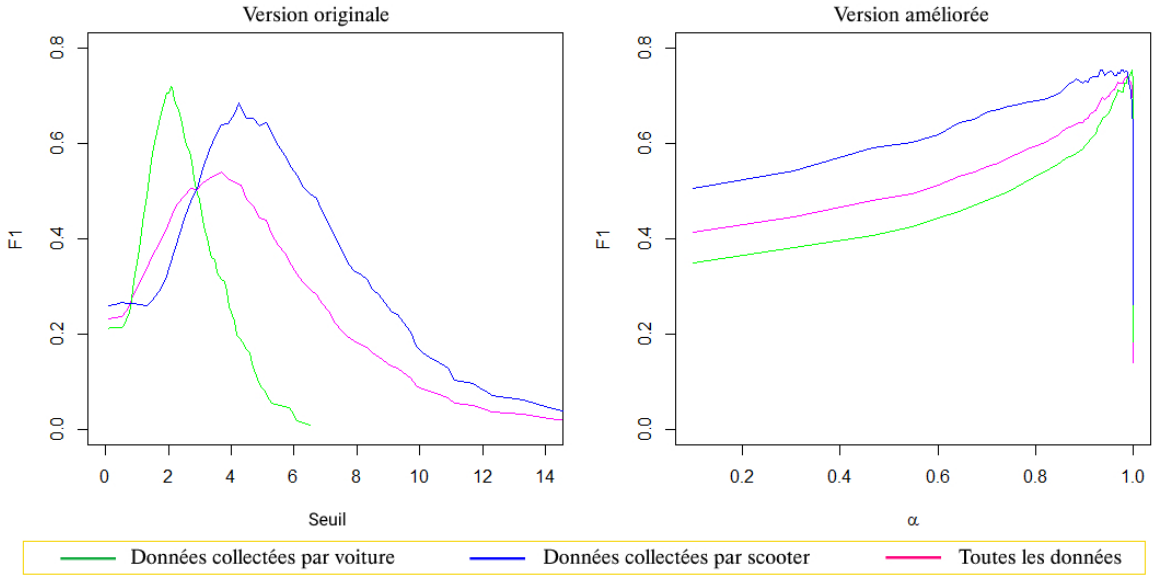


FIGURE 4.13 – Les graphiques de F-mesure de l'algorithme Z-THRESH sur les trois ensembles de données.

En résumé, l'application du test de Grubbs à des algorithmes basés sur un seuil permet de créer des algorithmes avec une complexité minimale et nécessitant peu de mémoire. Nos expériences montrent que les algorithmes améliorés permettent d'atteindre des résultats élevés et s'adaptent bien aux conditions réelles, ce qui convient parfaitement aux applications pour *smartphones*.

4.3 Conclusion

L'objectif de ce chapitre était d'identifier un algorithme de détection des anomalies en temps réel pouvant s'adapter à de nombreuses conditions réelles et nécessitant le moins de ressources possible sur les *smartphones*. Nous avons commencé par examiner les méthodes de détection des anomalies routières. À partir des études actuelles, nous avons proposé une méthode améliorée de détection de ces anomalies pouvant être appliquée à la détection d'autres types d'anomalies.

Nous avons exploré les différentes méthodes de détection des anomalies routières qui ont été publiées jusqu'à présent et présenté à la fois leurs avantages et leurs inconvénients. Nous avons montré que les méthodes basées sur un seuil nécessitent le moins de

ressources mais sont moins adaptées aux conditions réelles des routes et des véhicules. Ainsi, nous avons suggéré d'utiliser un seuil dynamique pour ces méthodes. Pour les méthodes basées sur une classification, les données nécessaires à l'apprentissage sont généralement assez volumineuses.

Sur la base de ces études, nous avons proposé d'appliquer le test statistique de Grubbs aux algorithmes de détection basés sur un seuil pour détecter et localiser les anomalies routières en recherchant les valeurs aberrantes dans une fenêtre glissante de données des capteurs. De cette manière, les anomalies comme les nids-de-poule sont détectées sur la base de l'analyse d'un échantillon de données dans des conditions réelles. Par conséquent, cette méthode est capable de s'adapter à différentes conditions pratiques. Nous avons également montré comment utiliser la méthode de test Grubbs sans augmenter la complexité de l'algorithme. La validation expérimentale a été réalisée à partir de données réelles collectées à l'aide d'une voiture et d'un scooter dans les rues de Hue au Vietnam. En outre, nous avons également proposé de traiter les données d'accélération sous forme de séquences continues, sans les découper en fenêtres comme le font d'autres travaux de recherche. Pour les expériences, nous avons proposé de marquer le *ground truth* avec le temps. Les résultats ont été analysés sur la base de la courbe de rappel de précision et de l'indice de F-mesure montrant que notre méthode donne de meilleurs résultats et s'adapte très bien lorsque différents types de véhicule sont utilisés.

Notre méthode a été appliquée avec succès à la détection de nids-de-poule. Cependant, elle peut aussi être appliquée à la détection d'autres types d'anomalies et basée sur certains types de signaux tels que le son, le magnétisme ou l'accélération.

5 Agrégation de données des anomalies dans le SCS

5.1 Introduction

L'agrégation des données dans un système de capteurs composé de *smartphones* est un problème délicat. Le fonctionnement des systèmes de capteurs de *smartphones* permet la collecte d'informations à partir de nombreux endroits. Dans le cas d'un système de capteurs composé de *smartphones* pour la détection d'anomalies environnementales (Cf le Chap. 3), les informations envoyées au datacenter sont l'état et les anomalies de l'environnement. Ces données proviennent de nombreux *smartphones* largement dispersés. Une anomalie peut être détectée plusieurs fois à partir de différents *smartphones* et à différents moments. Lorsqu'une anomalie est détectée et que ses informations sont envoyées au centre, nous appelons ces informations une *observation* de cette anomalie ou une *anomalie rapportée*. En raison de l'erreur des capteurs et de l'algorithme de détection, quelques observations ne suffisent pas pour confirmer l'existence d'anomalies, il est donc nécessaire d'agéger les données obtenues depuis de nombreux *smartphones* afin d'améliorer la précision. Ce chapitre focalise principalement sur l'agrégation des données d'anomalies.

En fait, l'agrégation de données est l'une des étapes critiques du système. Cette étape complète la détection pour augmenter la précision. Comme les données GPS contiennent des erreurs, les observations de la même anomalie sont réparties dans une région. De plus, il y a des anomalies proches les unes des autres, ce qui implique que leurs observations sont mélangées dans l'espace. Le processus de l'agrégation des anomalies permet de trouver la position exacte des anomalies. De plus, l'agrégation des données a pour objectif de calculer la fiabilité des positions trouvées. Le résultat de cette étape, en plus d'être utilisé pour des applications routières, peut également être utilisé pour réajuster les paramètres de détection.

Actuellement, l'application la plus populaire nécessitant l'agrégation de données est le

système de détection d'anomalies routières. Certains travaux de recherche ont porté sur les méthodes d'agrégation des données d'anomalies. Jakob Eriksson et al. [21] ont notamment proposé une méthode de regroupement dans laquelle deux anomalies sont regroupées au sein d'un *cluster* si leur distance est inférieure à une valeur donnée Δd. Cependant, la distance maximale entre les anomalies dans un *cluster* est limitée à une valeur donnée Δt. Selon les auteurs, un événement est valide s'il possède au moins k éléments dans le *cluster* auquel il appartient. Zhaojian Li et al. [37] ont proposé une méthode de partitionnement des anomalies basée sur la distance de Mahalanobis [39]. L'idée principale pour le partitionnement est la suivante : un événement avec les coordonnées x est défini dans le *cluster* C si la distance de Mahalanobis $D(x, C)$ est inférieure à une valeur donnée de p. La distance de Mahalanobis est calculée à l'aide de l'équation :

$$D(x, C) = \sqrt{(x - \mu_C)^{\mathrm{T}} \Sigma_C^{-1} (x - \mu_C)}$$

où μ_C est l'espérance pondérée et Σ_C est la matrice de covariance pondérée de C. Les auteurs ont proposé d'utiliser l'heure de l'événement comme paramètre pour calculer μ_C et Σ_C comme suit :

$$\mu_C = \frac{\sum_{k=1}^{K_C} f(t - t_{Ck}) x_{Ck}}{\sum_{k=1}^{K_C} f(t - t_{Ck})}$$

$$\Sigma_C = \frac{\sum_{k=1}^{K_C} f(t - t_{Ck})(x_{Ck} - \mu_C)(x_{Ck} - \mu_C)^{\mathrm{T}}}{\sum_{k=1}^{K_C} f(t - t_{Ck})}$$

où t_{Ck} et x_{Ck} sont l'heure et les coordonnées du point de données k^{th} du *cluster* C, et t est l'heure actuelle. La fonction f est une fonction exponentielle de la forme $f(\tau) = \alpha^{-\lambda\tau}$ où $\alpha > 1$ et $\lambda > 0$ sont deux scalaires positifs.

L'inconvénient commun des deux méthodes ci-dessus est qu'elles ne s'arrêtent qu'au regroupement, il n'y a pas de calcul de la position des anomalies ni d'évaluation de la fiabilité. En outre, la méthode de Jakob Eriksson ne précis e pas comment supprimer les anomalies d'un cluster si la distance maximale dans ce *cluster* dépasse la valeur autorisée. Par conséquent, nous proposons une méthode d'agrégation de positions d'anomalies dans laquelle l'essentiel est un algorithme basé sur le *mean shift*.

5.2 Définition de problème

Les données des anomalies rapportées contiennent de nombreuses informations, notamment une localisation GPS avec une valeur de précision. Ici, nous considérons uniquement les coordonnées au sol en fonction des deux coordonnées, longitude et latitude. Par exemple, les données d'anomalies routières contiennent les informations clés suivantes :

- *time* : le moment de l'anomalie ;
- *type* : le type d'anomalie (nid-de-poule ou dos d'âne) ;
- *position* : longitude et latitude ;
- *accuracy* : la précision du GPS.

L'agrégation des anomalies peut s'effectuer de manière égale sur les différents types d'anomalie. Dans la suite du document, les algorithmes décrits sont appliqués aux nids-de-poule seulement. L'adaptation aux autres types d'anomalie pouvant se faire très facilement, elle n'est pas présentée dans cette thèse.

La position d'une anomalie comprend les coordonnées géographiques (la longitude et la latitude) et la *précision*, qui correspond à l'erreur horizontale estimée par le GPS en mètres. Ainsi, lors du calcul de la distance entre les points, les coordonnées sont converties en mètres.

Les positions GPS peuvent être considérées comme des données gaussiennes distribuées autour de la position réelle [68, 31, 20] et la précision horizontale du GPS sur le *smartphone* est la précision RMS (Root Mean Squared) en deux dimensions [20] :

$$RMS = \sqrt{\sigma_x^2 + \sigma_y^2}$$

où σ_x^2 et σ_y^2 sont les variances sur les axes. Comme les données GPS sur le *smartphone* ne contiennent pas les détails σ_x et σ_y, on peut prendre approximativement $\sigma_x \approx \sigma_y \approx \textit{précision}/\sqrt{2}$. Dans le cadre de nos recherches, nous avons utilisé une valeur pour la variance $\sigma^2 = \sigma_x^2 = \sigma_y^2$ pour chaque point de données. En outre, le calcul de l'erreur GPS est basé sur l'hypothèse que les composantes de l'erreur sont indépendantes [36]. Nous assumons donc que la position GPS sur le plan est distribuée selon la loi normale avec la matrice de covariance $\Sigma = \text{diag}(\sigma^2, \sigma^2)$.

En raison de l'évolution de la technologie, le type précision des GPS peut changer. Le calcul de la variance devra peut-être être ajusté. Cependant le problème de l'agrégation des anomalies reste fondamentalement inchangé.

Les positions des anomalies rapportées peuvent être considérées comme des observations d'un modèle de mélange gaussien à deux variables dans lequel les variances

correspondant à chaque point d'observation sont connues. Il est à noter que, comme différentes observations d'une même anomalie peuvent correspondre à différentes variances, une anomalie ne correspond pas à un, mais à plusieurs gaussiennes du même couple d'espérances (μ_x, μ_y). Trouver la position des anomalies reviens à estimer les points (μ_x, μ_y). En d'autres termes, il s'agit d'un problème d'espérance-maximisation (EM). En outre, il est également nécessaire de calculer un poids correspondant à chaque point trouvé pour évaluer la fiabilité.

Notre solution consiste à construire un algorithme EM basé sur l'algorithme *mean shift* pour trouver les points maximum de la fonction d'estimation de la densité de probabilité. Cependant, la localisation des anomalies étant dispersée à travers le monde, nous devons d'abord diviser les données en *cluster* non inter-connectées afin d'accroître la précision et réduire le temps de calcul de l'algorithme EM.

5.3 Partitionnement simple de données des anomalies

L'algorithme de partitionnement simple est le pré-traitement du processus de l'agrégation des données d'anomalies. Les anomalies peuvent être réparties sur une vaste zone, de sorte que ses observations sont également divisées en différentes régions. Par exemple, avec le système de détection des anomalies routières, les anomalies rapportées peuvent être très volumineuses et distribuées dans le monde entier. Notre algorithme de partitionnement permet de traiter le fractionnement des données afin d'obtenir des avantages en termes de ressources informatiques ainsi que d'efficacité d'agrégation des données.

Le partitionnement avant la localisation des anomalies apporte plusieurs avantages. Le premier est une réduction significative du temps de calcul. L'algorithme permettant de trouver la position des anomalies par la méthode *mean shift* présente une complexité polynomiale [11, 12, 9]. Comme $a_1^n + a_2^n + .. + a_m^n < (a_1 + a_2 + .. + a_m)^n$ où $a_1, a_2, .., a_n$ et n sont des entiers positifs et $n > 1$, lorsque l'algorithme *mean shift* est appliqué à de petits groupes indépendants, le nombre de calculs est très réduit. Le deuxième avantage est de réduire les besoins en ressources du système. Lorsque de grandes tailles de données sont traitées en une seule fois, le système nécessite beaucoup de calculs et de mémoire. Inversement, lorsque le traitement est effectué successivement sur une partie des données, les besoins en ressources sont moins importants. Un autre avantage du partitionnement est une augmentation de la précision pour la location des anomalies. Comme l'algorithme trouve des anomalies sur la fonction d'estimation de la densité de probabilité, lorsqu'il y a peu d'anomalies (ce qui ne signifie

pas qu'il y a moins d'observations pour chaque anomalie), la fonction donne des résultats plus précis. De plus, en traitant les données d'une petite zone géographique il est possible de la voir comme une surface plane. En outre, notre classification est implémentée cumulativement, c'est-à-dire que chaque fois qu'une nouvelle anomalies reportée arrive, le système met simplement à jour les *clusters* sans avoir besoin de tout recalculer.

Afin de construire l'algorithme de *partitionnement simple*, considérons d'abord quelques définitions. En supposant que p_i et p_j soient deux points correspondant aux anomalies reportées, nous appelons ces deux points *éloignés* si la distance qui les sépare est supérieure à une valeur spécifique d_{ij}. Ceci est défini par la fonction *far* :

$$\text{far}(p_i, p_j) = \begin{cases} \text{vrai} & \text{si } \|p_i - p_j\| > d_{ij} \\ \text{faux} & \text{autrement} \end{cases}$$

À partir de la fonction *far*, nous définissons la fonction *connected*(C, p) pour déterminer si un point p est proche du *cluster* C :

$$\text{connected}(C, p) = \begin{cases} \text{vrai} & \text{si } \exists\ q \in C \ /\ \text{far}(p,q)=\text{faux} \\ \text{faux} & \text{autrement} \end{cases}$$

La valeur d_{ij} doit être choisie suffisamment grande pour que, dans le cas où p_i et p_j ne soient pas dans le même *cluster*, la probabilité que ces deux points appartiennent à la même anomalie réelle soit très petite. Dans l'hypothèse d'une distribution gaussienne (comme mentionnée plus haut), la probabilité qu'une observation tombe en dehors du cercle de rayon r pour le centre correspondant à la position de l'anomalie est $\exp\left(-\frac{r^2}{2\sigma^2}\right)$, où σ^2 est la variance. Soit n le nombre d'observations. Il vient que la probabilité qu'une seule observation tombe en dehors de ce cercle, selon la distribution binomiale, est de $C_n^1 \exp\left(-\frac{r^2}{2\sigma^2}\right)^1 \left(1 - \exp\left(-\frac{r^2}{2\sigma^2}\right)\right)^{n-1}$ qui est inférieur à $n * \exp\left(-\frac{r^2}{2\sigma^2}\right)$. Par conséquent, si $p_i - p_j > d_{ij}$ et $p_j \in C$, la probabilité que p_i appartienne à une anomalie du *cluster* C est inférieure à $\delta = N_{max} * \exp\left(-\frac{d_{ij}^2}{2\sigma_i^2}\right)$, où N_{max} est le nombre maximum de rapports possibles pour une anomalie. La valeur d_{ij} peut être sélectionnée en fonction de la valeur souhaitée de δ. En fait, nous devons également nous assurer que p_j n'appartient pas à une anomalie du *cluster* de p_i. Nous proposons donc la formule suivante :

$$d_{ij} = \max(\sigma_i, \sigma_j)\sqrt{-2\ln\left(\frac{\delta}{N_{max}}\right)} \tag{5.1}$$

Par exemple, si $N_{max} = 10000$ et $\delta = 0.0001$, alors $d_{ij} \approx 6 * \max(\sigma_i, \sigma_j)$. La valeur N_{max} peut être prédite à partir des données rapportées. Il est important de noter

que N_{max} est mis à jour, mais ne se développe pas indéfiniment dans le temps car le système supprime périodiquement les anciens rapports.

La formule (5.1) garantit uniquement que la valeur de d_{ij} est « assez grande », et n'est pas optimale pour la mise en *cluster*. L'objectif est que cette mise en *cluster* simple soit effectuée rapidement de manière cumulative. Cette technique de classification est plus détaillée en Sec. 5.4.

L'algorithme de partitionnement des données est exécuté de manière cumulative, c'est-à-dire qu'il est appelé lorsqu'un nouveau point de données est ajouté.

La mise à jour des *clusters* est présentée dans Algorithme 1.

Algorithme 1 Mise à jour des *cluster*.

INPUT: $C(C_1, C_2, .., C_K)$ {liste des *clusters* actuelles}, p {nouveau point}

OUTPUT: C {nouvelle liste des *clusters*}

$C^* \leftarrow \emptyset$ {C^* consiste à stocker toutes les *clusters* actuels dont p est proche}

for $k \in \{1, .., K\}$ **do**

  if $connected(C_k, p)$ **then**

    $C^* \leftarrow C^* \cup \{C_k\}$

  end if

end for

{Prochaine étape : ajouter p dans l'union des *clusters* dans C^* pour obtenir un nouveau *cluster*}

$C \leftarrow C \backslash C^*$

$C_{new} \leftarrow \left(\bigcup_{i=1}^{|C^*|} C_i^* \right) \cup \{p\}$

$C \leftarrow C \cup \{C_{new}\}$

5.4 L'algorithmes d'agrégation des données

5.4.1 *Mean shift* à bande passante variable

Le *mean shift* (MS) est une technique d'analyse non paramétrique de l'espace-fonctionnalités (en anglais *feature-space*) permettant de localiser les maxima d'une fonction de densité (mode) basé sur les idées proposées par Keinosuke Fukunaga et Larry D. Hostetler [22]. Aujourd'hui, *mean shift* a été dérivé dans plusieurs versions différentes.

Les deux versions populaires de *mean shift* sont le *mean shift* à bande passante (bandwidth) fixe et le *mean shift* à *bande passante* variable [14]. Cette section présente un aperçu de la version de MS appliquée dans ces travaux : le *mean shift* à *bande passante* variable.

Mean shift est basée sur l'estimateur de la densité du noyau (*Kernel Density Estimator* - KDE). Soit p_i, $i = 1..n$ un ensemble de points dans l'espace $\mathcal{R}^d$, et les bandes passantes associées h_i. L'estimateur de densité calculé en un point donné p avec le *noyau* (*kernel profile*) $k(x)$ est donné par [54] :

$$\hat{f}(p) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_i^d}k(\left\|\frac{p-p_i}{h_i}\right\|^2) \tag{5.2}$$

La formule (5.2) est l'estimateur de la densité du noyau à bande passante variable. Si $h_1 = h_2 = ... = h_n = h$ alors (5.2) devient l'estimateur de la densité du noyau avec la bande passante fixée.

Un noyau $k(x)$ est une fonction qui remplit les conditions suivantes [11] :

1) k est non négatif.
2) k est non croissante, c'est-à-dire $k(a) \geq k(b)$ si $a < b$.
3) k est continue par morceaux et $\int_0^\infty k(x)dx < \infty$

Quelques noyaux sont communément utilisés [72, 11] :

- Noyau plat :

$$k(x) = \begin{cases} 1 & \text{si } x \leq 1 \\ 0 & \text{si } x > 1 \end{cases}$$

- Noyau d'Epanechnikov :

$$k(x) = \begin{cases} 1\text{-}x & \text{si } x \leq 1 \\ 0 & \text{si } x > 1 \end{cases}$$

- Noyau gaussien :

$$k(x) = \exp(-\frac{1}{2}x)$$

Depuis l'estimateur de la densité (5.2), on peut définir l'estimation du gradient de densité en tant que gradient de l'estimation de la densité du noyau, nous avons :

$$\begin{aligned}\hat{\nabla} f(p) &\equiv \nabla \hat{f}(p) = \frac{2}{n}\sum_{i=1}^{n} \frac{p-p_i}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right) \\ &= \left[\frac{2}{n}\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right)\right] \times \left[\frac{\sum_{i=1}^{n} \frac{p_i}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right)} - x\right] \end{aligned} \tag{5.3}$$

où $g(x) = k'(x)$.

Le dernier crochet dans (5.3) représente le vecteur de *mean shift* à bande passante variable [54] :

$$m(p) = \frac{\sum_{i=1}^{n} \frac{p_i}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right)}{\sum_{i=1}^{n} \frac{1}{h_i^{d+2}} g\left(\left\|\frac{p-p_i}{h_i}\right\|^2\right)} - x \tag{5.4}$$

Comme le vecteur de *mean shift* est proportionnel au gradient de la fonction $\hat{f}(p)$, il indique le point avec la densité estimée la plus élevée. L'idée de la méthode *mean shift* est la suivante : à partir de n'importe quel point p, le vecteur *mean shift* conduit ce point au maximum dans la direction du vecteur de gradient. Cette itération se termine lorsque le vecteur *mean shift* converge vers le vecteur zéro, la dernière position de p est un mode [13]

5.4.2 Identification des anomalies par un algorithm basé sur mean shift

Compte tenu des caractéristiques des données sur les anomalies rapportées, nous avons trouvé que l'algorithme *mean shift* est une option efficace pour résoudre le problème. Dorin Comaniciu a montré que *mean shift* à bande passante variable est un estimateur adaptatif du gradient normalisé [12]. Carreira-Perpinan et Miguel A. ont également confirmé que, quand le noyau est Gaussien, mean-shift est un algorithme espérance-maximisation [10]. En ce qui concerne le problème de l'agrégation des anomalies, on peut voir qu'il s'agit d'un modèle de mélange gaussien. Ce modèle est assez compliqué à calculer car différentes observations correspondent à une précision différente. En théorie, chaque observation différente correspond à un gaussien. Cependant, cette complexité peut être facilement résolue par un *mean shift* à bande passante variable dans lequel chaque point de données est combiné à une valeur de bande passante adaptée. De plus, l'algorithme *mean shift* résout simultanément deux problèmes : le problème de recherche de mode pour localiser les anomalies et le problème de

classification pour calculer la fiabilité des modes identifiés. Le noyau approprié pour ce modèle est évidemment un noyau gaussien.

Un des avantages du problème d'agrégation des anomalies est que l'écart-type associé à chaque point de données est connu (voir la Sec. 5.2). La valeur de la bande passante appropriée semble être cet écart-type. Cependant, l'estimateur de la densité n'est pas une densité de probabilité exacte. De plus, en raison de la confusion sur les données d'observation, le *mean shift* ne donne pas toujours des résultats exacts. La bande passante doit être ajustée selon les caractéristiques des données et les objectifs de l'algorithme pour augmenter l'efficacité. Par conséquent, nous proposons d'utiliser la bande passante $h_i = \lambda\sigma_i$, où λ est un paramètre expérimental ajusté autour de 1 pour obtenir les meilleurs résultats pour chaque situation.

Maintenant, le vecteur *mean shift* doit être recalculé pour le problème d'agrégation des anomalies. Le noyau gaussien utilisé est :

$$k(x) = \exp(-\frac{1}{2}x)$$

donc :

$$g(x) = k'(x) = -\frac{1}{2}\exp(-\frac{1}{2}x) \tag{5.5}$$

La position d'une anomalie est définie sur un espace bidimensionnel, en remplaçant $d = 2$, $h_i = \lambda\sigma_i$ et (5.5) dans (5.4), on obtient le vecteur *mean shift* suivant :

$$m(p) = \frac{\sum_{i=1}^{n} \frac{p_i}{\lambda^4\sigma_i^4} \exp(-\frac{1}{2}\left\|\frac{p-p_i}{\lambda\sigma_i}\right\|^2)}{\sum_{i=1}^{n} \frac{1}{\lambda^4\sigma_i^4} \exp(-\frac{1}{2}\left\|\frac{p-p_i}{\lambda\sigma_i}\right\|^2)} - p \tag{5.6}$$

Sur la base de ce vecteur *mean shift*, nous avons construit un algorithme basé sur celui présenté dans [9] pour trouver les points maximum de la densité (Cf l'Algorithme 2). L'algorithme assigne un *poids* pour chaque point maximum trouvé. Ces poids sont stockés dans le système pour évaluer sa fiabilité. Un mode (point maximum de la densité) est traité comme une anomalie si le poids associé dépasse un certain seuil. Nous pouvons également améliorer l'algorithme en éliminant les modes qui suivent un certain critère lié au poids.

Il est recommandé d'utiliser la fonction de pondération $w(p)$ comme une proportion de $\hat{f}(p)$ avec le noyau gausien et la bande passante σ_i :

$$\mathrm{w}(p) = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} \exp\left(-\frac{\|p-p_i\|^2}{2\sigma_i^2}\right) \tag{5.7}$$

Algorithme 2 Identification des anomalies

INPUT:

$p_1, p_2, .., p_n$ {La liste des points (positions d'anomalies reportées)}

$\sigma_1, \sigma_2, .., \sigma_n$ {La liste des écarts-types des points correspondants}

λ {Le paramètre de bande passante}

ε, ρ {Les paramètres de tolérance}

OUTPUT: Q, w {La liste des modes et la liste des poids}

$Q \leftarrow \emptyset$

for $i \in \{1, .., n\}$ **do**

$p \leftarrow p_i$

repeat

$$m(p) \leftarrow \frac{\sum_{j=1}^{n} \frac{p_j}{\lambda^4 \sigma_j^4} \exp(-\frac{1}{2} \left\| \frac{p-p_j}{\lambda \sigma_j} \right\|^2)}{\sum_{j=1}^{n} \frac{1}{\lambda^4 \sigma_j^4} \exp(-\frac{1}{2} \left\| \frac{p-p_j}{\lambda \sigma_j} \right\|^2)} - p$$

$p \leftarrow p + m(p)$

until $|m(p)| < \varepsilon$

$q = nearestPoint(Q, p)$ {*nearestPoint(Q,p)* renvoie le point dans Q le plus proche de p}

if $\exists q$ **and** $\|q - p\| < \rho$ **then**

$w(q) \leftarrow w(q) + \frac{1}{\sigma_i^2} \exp\left(-\frac{\|q-p_i\|^2}{2\sigma_i^2}\right)$

else

$w(p) \leftarrow \frac{1}{\sigma_i^2} \exp\left(-\frac{\|p-p_i\|^2}{2\sigma_i^2}\right)$

$Q \leftarrow Q \cup \{p\}$

end if

end for

Cette proposition est basée sur la propriété suivante :

Propriété 1. *Soit p la position d'une anomalies, et soient $p_1, p_2, .., p_n$ les observations de cette anomalie avec les écarts-types $\sigma_1, \sigma_2, ..\sigma_n$ respectivement. En considérant que les coordonnées GPS reportées sont distribuées selon la loi normale avec des coordonnées indépendantes, l'espérance mathématique de la fonction w calculée au point p est :*

$$\mathrm{E[w(p)]} = n\frac{\overline{(\sigma_i^{-2})}}{2} \tag{5.8}$$

Démonstration. Tout d'abord, considérons le cas où il n'y a qu'une observation p_i. Il est possible de choisir p comme l'origine du système de coordonnées. Soit (x, y) les coordonnées de p_i. L'espérance de la fonction w est fournie par :

$$\begin{aligned}
\mathrm{E[w_i(p)]} &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \mathrm{w_i(p)} f(x,y)dxdy \\
&= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \frac{1}{\sigma_i^2}\exp\left(-\frac{x^2+y^2}{2\sigma_i^2}\right)\frac{1}{2\pi\sigma_i^2}\exp\left(-\frac{x^2+y^2}{2\sigma_i^2}\right)dxdy \qquad (5.9)\\
&= \frac{1}{2\pi\sigma_i^4}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\exp\left(-\frac{x^2+y^2}{2\sigma_i^2}\right)dxdy
\end{aligned}$$

où $f(x, y)$ est la fonction de densité de probabilité d'une distribution normale bivariée. L'intégrale de l'équation dans (5.9) peut être calculée en coordonnées polaires :

$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\exp\left(-\frac{x^2+y^2}{\sigma^2}\right)dxdy = \int_{0}^{+\infty}\int_{0}^{2\pi}\exp(-\frac{r^2}{\sigma^2})\, r\, dr\, d\theta = \pi\sigma^2$$

d'où :

$$\mathrm{E[w_i(p)]} = \frac{1}{2\sigma_i^2} \qquad (5.10)$$

À partir de la formule (5.10), dans le cas de n observations indépendantes, l'espérance de la fonction w est la suivante :

$$\mathrm{E[w(p)]} = \sum_{i=1}^{n}\mathrm{E[w_i(p)]} = \sum_{i=1}^{n}\frac{1}{2\sigma_i^2} = n\frac{\overline{(\sigma_i^{-2})}}{2}$$

□

La propriété 1 montre que l'espérance de la fonction w est directement proportionnelle au nombre d'observations d'anomalie. En outre, plus l'erreur de rapport est petite, plus la valeur de la fonction w est grande. En conséquence, la fonction w peut être utilisée comme une fonction de pondération pour évaluer la fiabilité des modes. Pourtant, il est à noter que l'équation (5.8) ne s'applique que dans le cas d'une anomalie. En effet, il peut y avoir plusieurs anomalies, donnant lieu à plusieurs modes. Il est nécessaire de partitionner les données par mode pour appliquer la fonction de pondération.

Le poids d'un mode n'est calculé que sur des points situés dans son *bassin d'attraction*. Un bassin d'attraction, qui est utilisé comme un cluster de l'algorithme *mean shift*, est une région dans laquelle tous les points ont une trajectoire (définie par les vecteurs *mean shift*) menant au même mode [49] (Cf Fig. 5.1).

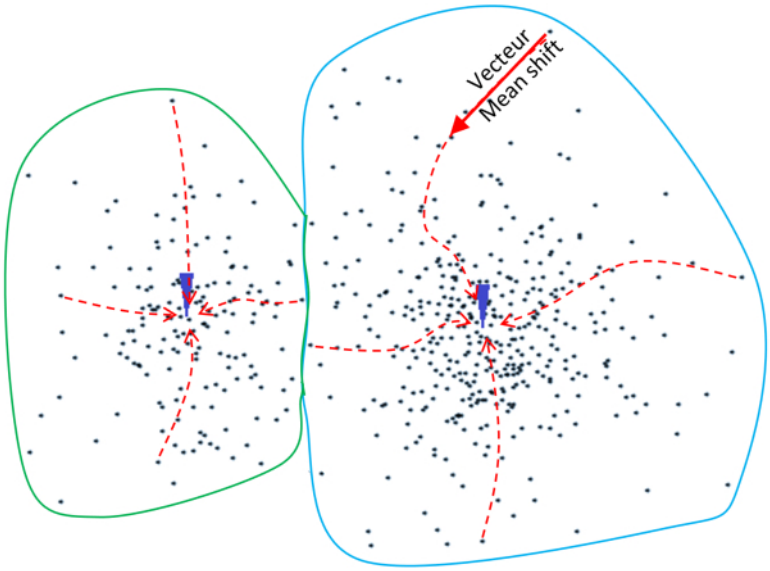


FIGURE 5.1 – Exemple avec deux bassins d'attraction.

Dans l'algorithme 2, deux paramètres de tolérance (ε et ρ) sont utilisés. Dans l'algorithme *mean shift*, le décalage avec le vecteur *mean shift* s'arrête lorsque la distance de décalage est inférieure à une tolérance positive [9] (ce processus est effectué avec la boucle `repeat` ... `until` dans l'algorithme 2). La valeur de tolérance ε est choisie en fonction de l'application et est spécifique. Plus ε est élevé, plus le programme est rapide. Cependant, la précision de la localisation des modes est réduite. Avec cette condition d'arrêt, le programme peut donner plusieurs points de convergence autour d'un mode réel. Pour résoudre ce problème, un paramètre ρ est ajouté et l'algorithme doit assurer :

1) qu'il n'y a pas deux modes tel que la distance entre eux soit inférieure à ρ ;
2) que si un point de convergence est supprimé, il doit y avoir un autre point de convergence sélectionné afin que la distance entre eux soit inférieure à ρ.

Le paramètre ρ doit être suffisamment grand pour que le programme ne produise qu'un seul point pour chaque mode, et suffisamment petit pour empêcher la fusion de différents modes réels en un seul. Dans nos expériences de simulation, nous avons d'abord choisi ε puis fait une expérience pour choisir un ρ convenable.

5.4.3 Simulation et résultats

L'objectif de nos expériences est de tester l'efficacité de notre solution et de déterminer la valeur des paramètres λ sur le résultat de l'algorithme 2, afin d'attirer un choix approprié. Nous avons expérimenté des données réparties dans de petites zones,

d'environ 100 mètres carrés. En effet, l'algorithme 2 est appliqué aux partitions créées par l'algorithme 1.

Principe

Nous avons effectué la simulation comme suit. À partir d'un ensemble de points choisis, $P = \{p_1, p_2, .., p_r\}$ représentant la position des anomalies, le programme de simulation génère le jeu de données de cette manière : pour chaque point p_i, k points $P_i = \{p_{i1}, p_{i2}, .., p_{ik}\}$ représentant les observations de l'anomalie i sont générés. Les points p_{ij} sont générés aléatoirement selon la distribution gaussienne avec la variance $\sigma^2_{xij} = \sigma^2_{yij} = \sigma^2_{ij}$ autour du point p_i. La valeur de σ_{ij} est également une valeur aléatoire entre σ_{min} et σ_{max}. Les valeurs de σ_{min} et σ_{max} sont 2,12 et 8.48 respectivement, ce qui correspond à la précision du GPS qui est comprise entre 3 et 12 m.

L'ensemble des $k \times r$ points générés sert de données test. La liste des modes obtenue est comparée à l'ensemble de points d'origine P pour évaluer les résultats.

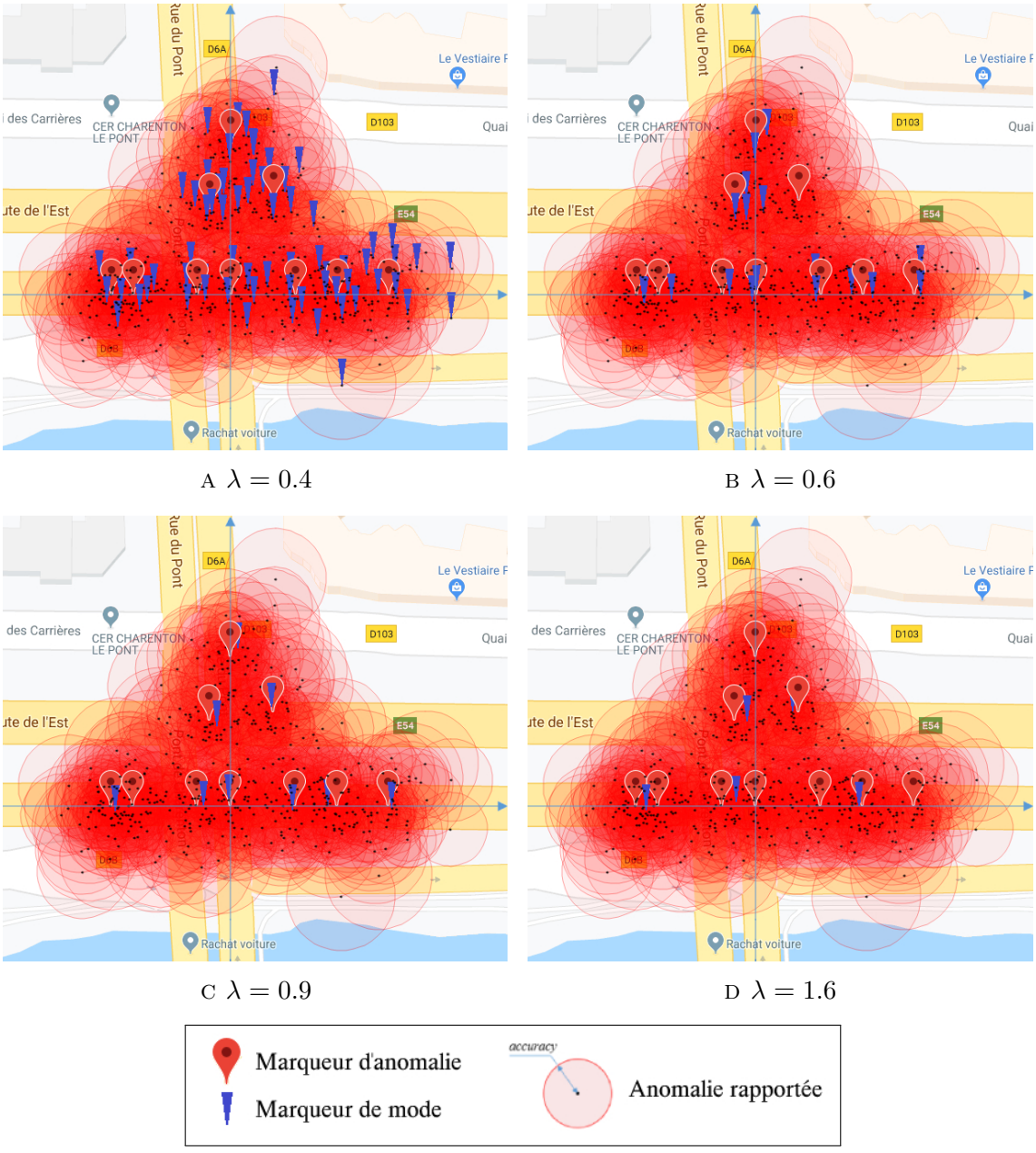
Au cours de l'expérience, nous avons d'abord observé l'effet du paramètre sur les résultats afin de sélectionner les paramètres et procéder à l'obtention des résultats officiels. Nous avons expérimenté le cas d'un mode, $\lambda = 0.9$, $\varepsilon = 0.01$, avec un nombre de points variant : 20, 50, 100, 1000. Le programme produit des points avec une distance moyenne et un écart-type, calculé sur 1000 essais, respectivement de $(0.008, 0.0123)$, $(0.012, 0.0147)$, $(0.014, 0.0154)$ et $(0.013, 0.0078)$. Nous choisissons le paramètre $\rho = 0.1$, qui est une valeur assez grande afin d'assurer que ces points ne sont comptés que pour un seul mode. D'autre part, cette valeur du ρ est très petites par rapport à l'erreur GPS. Nous avons donc choisi la paire de paramètres$\varepsilon = 0.01$ et $\rho = 0.1$ pour ces expériences.

Expérience visuelle

La première expérience a pour but de visualiser les résultats de manière visuelle. À travers ce résultat, on peut voir l'effet du paramètre λ sur le résultat.

Cette expérience est conduite sur un ensemble d'anomalies P constitué de 10 points d'anomalies imaginaires. L'axe vertical est dans la direction sud-nord. L'axe horizontal est dirigé d'ouest en est. L'unité de longueur est le mètre. Les dix points sont aux coordonnées (-23,0), (-28,0), (-8,0), (0,0), (15,0), (25,0), (37,0) , (0,35), (-5,20) et (10,22). Pour chacun de ces points, nous avons généré $k = 50$ points de données. Nous avons étudié les résultats avec des valeurs différentes de λ. Les résultats sont illustrés

sur la carte géographique. L'origine des coordonnées est au point (lat=48,817456 nord, lon=2,419799 est).



A $\lambda = 0.4$ B $\lambda = 0.6$

C $\lambda = 0.9$ D $\lambda = 1.6$

FIGURE 5.2 – Résultats visuels de la simulation

Les résultats correspondant au cas où λ est égal à 0.4, 0.6, 0.9 et 1.6 sont illustrés à la Fig. 5.2. À travers cette figure, on peut voir que lorsque la bande passante est trop petite, l'algorithme 2 reçoit trop de points maxima. Ensuite, plus la bande passante est large, plus le nombre de points maxima est petit, les maxima les plus proches se regroupent pour finir par ne faire qu'un seul. Avec les valeurs λ appropriées, par exemple. $\lambda = 0.9$ dans notre expérience , les résultats sont très bons à l'exception des deux points très proches. Ils sont situés à une distance de 5 m et le programme

les identifie comme un seul et unique point situé au centre. Dans le cas spécifique de l'identification des anomalies de la route, ce résultat ne devrait pas s'améliorer davantage. Cependant, ceci ne constitue qu'une difficulté théorique dans la mesure où d'un point de vue purement pratique l'opérateur qui sera mandaté sur place pourra voir de visu les nids-de-poule à reboucher et agir en conséquence.

Évaluation de l'effet des paramètres λ sur l'efficacité de l'algorithme

L'expérience suivante consiste à évaluer la dépendance de l'efficacité algorithmique sur les paramètres λ. Pour cela, il est nécessaire d'abord de définir une fonction objectif.

Soit P l'ensemble des anomalies initiales. Soit Q l'ensemble des modes obtenus à partir de l'algorithme. Soient p et q deux points tels que $p \in P$ et $q \in Q$. La fonction objectif est définie :

$$T(P,Q) = |\{p : p \in P \wedge \text{found}(p,Q)\}| - |\{q : q \in Q \wedge \neg \text{ matched}(q,P)\}|$$

où

$$\text{found}(p,Q) = \begin{cases} \text{vrai} & \text{si } \exists\ q^* \in Q \text{ tel que } \|p - q^*\| < \omega \\ \text{faux} & \text{sinon} \end{cases}$$

et

$$\text{matched}(q,P) = \begin{cases} \text{vrai} & \text{si } \exists\ p^* \in P \text{ tel que } \|p^* - q\| < \omega \\ \text{faux} & \text{sinon} \end{cases}$$

Nous avons choisi $\omega = 4$ mètres pour ces expériences. Cette valeur appliquée à la recherche des anomalies routières est acceptable. On peut ajuster ω pour l'adapter à l'application et à la précision du GPS. Avec une hypothèse sur l'erreur du GPS comme indiquée au début de la Sec. 5.4.3, nous avons effectué des expériences avec des ensembles de deux points et avons déterminé que, si la distance entre les deux points est inférieure à 4 mètres, la probabilité d'identifier les deux anomalies est inférieure à 10%.

Les Fig. 5.3 et 5.4 montrent les résultats des expériences pour les 10 points originaux comme dans la première expérience. Dans cette expérience, 6 valeurs de k sont utilisées et λ prend successivement les valeurs de 0,5 à 1,5 par pas de 0,025. Pour chaque paire (k, λ), la fonction objectif est moyennée sur 100 tentatives. Les graphiques permettent de dessiner certaines des propriétés suivantes :

- en général, plus les données sont volumineuses, meilleur est le résultat. Avec $k \geq 50$, le résultat du programme est très bon, près de 10 (alors que le nombre de maxima à trouver est justement de 10) ;

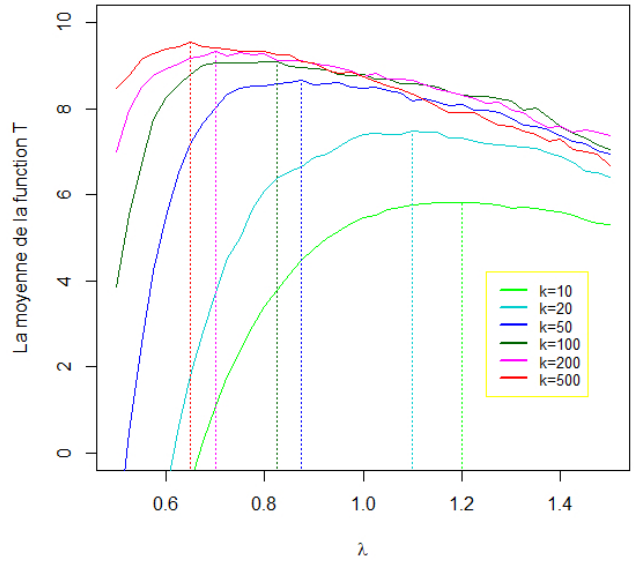


FIGURE 5.3 – Moyenne de la fonction objectif selon λ.

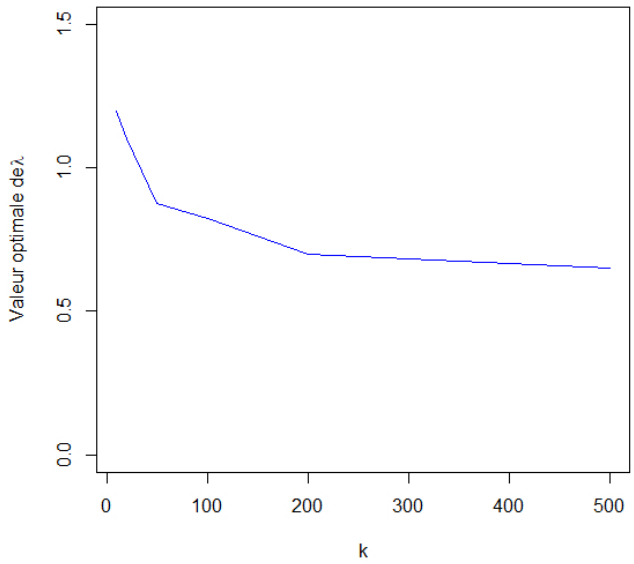


FIGURE 5.4 – Valeur optimale de λ selon k.

- plus le jeu de données est grand, plus la valeur λ requise est petite pour que l'algorithme obtienne le résultat optimal (Cf Fig. 5.4) ;
- plusieurs expériences sur des données réelles sont nécessaires pour estimer les bonnes valeurs de λ en fonction du nombre total de points n. Selon cette expérience, λ devrait être choisi dans la plage allant de 0,7 à 0,9. Comme cette valeur est assez petite, le programme aboutit généralement à identifier plus d'anomalies qu'il n'y en a effectivement, en particulier lorsque la taille de l'ensemble de données est petite. Cependant, cela peut être amélioré en éliminant les points de faible poids pour éliminer les points maximum éloignés de l'anomalie réelle.

Les résultats ci-dessus ne changent pas beaucoup si le nombre de rapports correspond

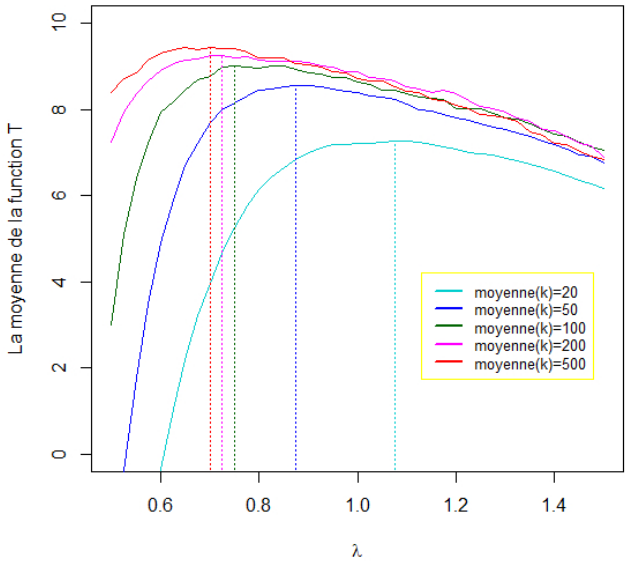


FIGURE 5.5 – Moyenne de la fonction objectif selon λ dans le où le nombre de rapports est aléatoire.

à des anomalies différentes. Nous avons également effectué une expérience avec k aléatoire en fonction de chaque anomalie à condition que $k \geq 10$ et que le nombre moyen de rapports reste inchangé. Les résultats présentés à la Fig. 5.5 montrent que les conclusions ci-dessus restent correctes.

Évaluation de l'effet du paramètre λ sur la capacité de notre solution à différencier des anomalies proches

La valeur de λ affecte également la capacité à distinguer les anomalies proches les unes des autres. Nous avons conduit l'expérience avec $k = 50$ sur l'ensemble P qui contient deux points avec des distances différentes. Les résultats sont illustrés à la Fig. 5.6.

Il souligne qu'une bande passante petite permet de différencier les anomalies proches les unes des autres, mais réduit les résultats pour des distances entre anomalies plus grandes. À partir des résultats du test indiqués sur les graphiques, on peut choisir le paramètre λ en fonction de la quantité des données et de la densité de l'événement. On peut également choisir une valeur harmonique pour λ comprise entre 0,7 et 0,9.

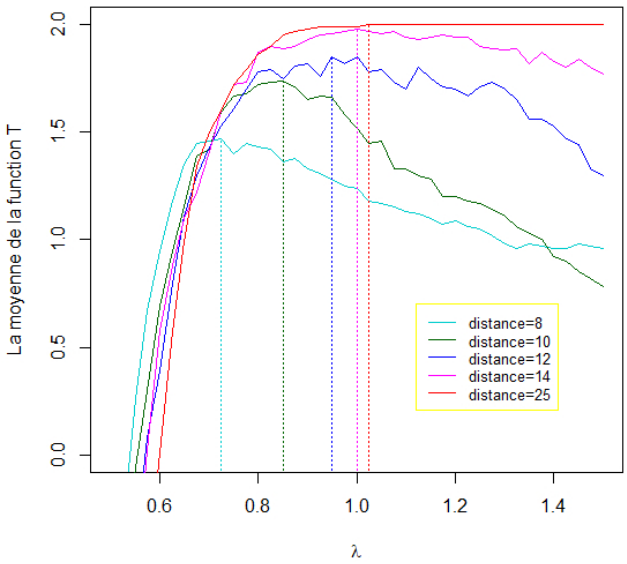


FIGURE 5.6 – Résultats expérimentaux sur deux points avec des distances différentes.

5.5 Conclusion

Dans ce chapitre, nous proposons une méthode permettant d'identifier des anomalies environnementales à partir des données rapportées par un système de capteurs basé sur des *smartphones*. Notre méthode se déroule en deux étapes. Premièrement, il s'agit d'un regroupement simple basé sur la distance entre les observations. Le but de cette classification est de fractionner les données en parties non liées. Cela augmente la rapidité et la précision du processus de localisation des anomalies. L'algorithme de regroupement fonctionne de manière cumulative. Cela signifie que les *clusters* sont mis à jour chaque fois que de nouvelles données arrivent au centre. Ensuite, un algorithme basé sur le *mean shift* détermine la position des anomalies et attribue à chaque point un poids comme base d'évaluation de la fiabilité. Nous avons choisi d'appliquer l'algorithme *mean shift* à bande passante variable avec le noyau gausien pour résoudre ce problème. Nous avons également mené des expériences de simulation. Ces expériences montrent que l'algorithme donne de très bons résultats. Des expériences ont été également menées pour différentes tailles et distances moyennes entre points afin de déterminer comment sélectionner les paramètres du système.

Nos principales contributions dans ce chapitre ont consisté à élaborer un algorithme de partitionnement et à personnaliser l'algorithme *mean shift* pour l'adapter à l'agrégation des positions des anomalies. Nous proposons également une fonction de pondération et prouvons que cette fonction est appropriée pour évaluer la fiabilité.

6 Conclusion et perspectives

6.1 Conclusion

À travers cette thèse, nous avons étudié comment améliorer certaines étapes des systèmes de capteurs à base de *smartphones* (SCS) pour la détection d'anomalies environnementales. Nos améliorations incluent l'architecture du système, les méthodes de détection d'anomalies et les algorithmes d'agrégation de données.

Au Chap. 2, nous avons discuté des recherches en cours liées aux améliorations susmentionnées, dans lesquelles, la Sec. 2.2 est consacrée à l'analyse de systèmes types aux niveaux organisationnel et opérationnel. Nous trouvons ce qui est acceptable et ce qui ne l'est pas pour les systèmes SCS pour la détection des anomalies. À partir de ces analyses, nous avons construit une architecture de système SCS pour la détection des anomalies environnementales au Chap. 3. Ce chapitre présente le modèle de fonctionnement et la disposition des composants tels que l'économie des ressources sur les *smartphones*. Nous avons également proposé le langage de format BSSDF pour envoyer des données semi-structurées depuis un *smartphone* vers le datacenter. Le format BSSDF limite l'utilisation des balises et stocke des données binaires. Par conséquent, la taille des données sous format BSSDF est inférieure à celle de XML et même de JSON, car la propriété data contient de nombreuses valeurs numériques longues (coordonnées géographiques, durée en millisecondes, données accélérées avec de nombreuses décimales, etc). Nous avons implémenté avec succès BSSDF pour notre système de test. Le fonctionnement de ce système est également présenté dans ce chapitre.

En ce qui concerne la méthode de détection d'anomalies, nous avons étudié la méthode de détection des anomalies routières. L'analyse et l'évaluation des méthodes actuelles sont présentées en Sec. 2.3. Nous avons classé les méthodes existantes en deux catégories : les méthodes basées sur un seuil et les méthodes basées sur une classification. Les

méthodes basées sur un seuil sont simples, peu complexes et consomment peu de mémoire. Les méthodes basées sur une classification nécessitent beaucoup de ressources sur le téléphone. Les méthodes actuelles sont mal adaptées aux mauvaises conditions environnementales et les matériels, notamment celles basées sur un seuil. Nous avons trouvé une méthode permettant d'améliorer les algorithmes basés sur un seuil pour créer des algorithmes qui s'adaptent mieux aux conditions réelles. Nous avons présenté cette méthode au Chap. 4. Notre approche a consisté à appliquer les méthodes de test des valeurs aberrantes en statistique à des échantillons de données courts-continus. Nous avons appliqué le test de Grubbs à cinq algorithmes pour détecter des anomalies routières. Les algorithmes améliorés ont une complexité en $O(n)$.

Pour l'application du SCS à la détection des anomalies de la route, nous avons également proposé une méthode permettant de recalculer la composante verticale du vecteur d'accélération dans le cas où le *smartphone* est placé dans une orientation autre que verticale. Nous avons également proposé une méthode pour détecter les actions du participant en suivant l'origine de la rotation du vecteur d'accélération.

Nous avons effectué des expériences sur des données collectées par voiture et scooter sur 40 km de rues dans la ville de Hué au Vietnam pour valider la méthode d'amélioration des algorithme de détection des anomalies routières. Les résultats montrent que nos méthodes donnent d'excellents résultats et s'adaptent très bien aux conditions réelles. Les résultats expérimentaux avec trois jeux de données (collectés avec le véhicule, collectés par scooter et toutes les données collectées) sont très similaires. Dans le même temps, pour les algorithmes originaux, lorsque les données varient, les résultats expérimentaux diminuent de manière significative.

Au Chap. 5, nous avons étudié les méthodes actuelles d'agrégation de données d'anomalies. Grâce à l'analyse, nous avons pu constater que ces méthodes ne font qu'arrêter le regroupement des données, sans rechercher l'emplacement des anomalies avec la précision nécessaire. Nous avons proposé une méthode d'agrégation de données en deux étapes. Premièrement, les données d'anomalies sont regroupées de manière cumulative par un simple algorithme de regroupement basé sur la distance. Nous avons ensuite appliqué l'algorithme *mean shift* pour trouver les maxima des densités de probabilité. Ces points sont les positions les plus probables des anomalies. Nous avons formulé et également prouvé la validité de la fonction de pondération afin de nous permettre d'évaluer la fiabilité des positions trouvées. Nous avons également effectué des expériences de simulation pour prouver l'exactitude de cet algorithme. Les résultats des expériences visuelles et quantitatives ont montré que notre algorithme de localisation inhabituelle est très fiable.

6.2 Perspectives

Sur la base du système que nous avons construit et des méthodes d'amélioration que nous avons proposées, voici quelques directions de recherche envisageable pour la suite.

L'extension de l'application des méthodes de détection des anomalies a permis de dégager des recommandations sur la détection d'autres types d'anomalies. Par exemple, des sons anormaux (tels que des accidents, des explosions, etc.) produisent également des valeurs aberrantes dans les données, en termes d'intensité ou de fréquence (voir la Fig. 6.1). L'utilisation de tests de valeurs aberrantes dans les statistiques peut donc aider à détecter des valeurs sonores anormales dans l'environnement. Cette méthode de détection d'anomalies peut également être appliquée pour détecter des modifications anormales du champ magnétique, de la lumière, etc. dans l'environnement servant aux applications futures.

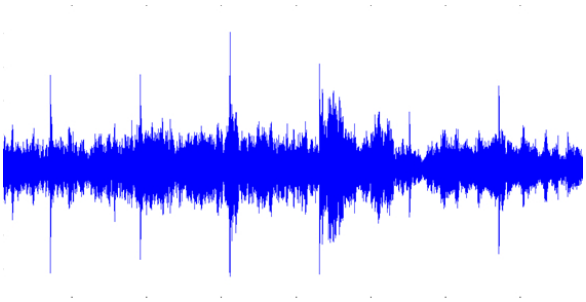


FIGURE 6.1 – Les ondes sonores montrent que les anomalies peuvent être détectées rapidement en puisant dans l'amplitude.

Une autre direction de recherche consiste à développer des méthodes supplémentaires pour détecter les anomalies par apprentissage automatique dans le *datacenter*. C'est un système central capable de classifier les anomalies, d'éliminer les fausses anomalies et de s'entraîner à ajuster les paramètres. L'apprentissage automatique du système peut être basé sur l'affirmation ou le rejet d'un responsable concernant l'existence d'anomalies et les résultats de l'agrégation des données.

Les expériences d'agrégation de données d'anomalies sur des données réelles est également une direction qui semble intéressante. Pour ce faire, un nombre important de volontaires est nécessaire dans certaines zones où se trouvent de nombreuses anomalies. En outre, l'application de notre méthode d'agrégation de données à d'autres types de données constitue également un axe de recherche car la position des données est basée sur des capteurs GPS. Ce qu'il faut, c'est ajouter les paramètres appropriés et ajuster les algorithmes en fonction de chaque cas. Par exemple, l'agrégation d'informations sur une explosion non seulement sur la densité des observations, mais également sur l'intensité sonore obtenue pour déterminer son emplacement de l'explosion.

En terme d'économie de batterie des téléphones, des recherches supplémentaires sont nécessaires pour établir de manière optimale la répartition des composants logiciels entre le *smartphone* et le *datacenter*. En outre, l'étude utilise le capteur à une certaine période. Par exemple, le GPS est activé pour obtenir les coordonnées à certains intervalles plutôt que pour une utilisation continue. Les données intermédiaires seront interpolées. La question est le temps nécessaire pour équilibrer précision et économie de batterie.

En outre, les systèmes de capteurs peuvent utiliser des informations supplémentaires provenant d'autres sources pour augmenter la précision. Par exemple, le système de détection d'aberrations routières sur route pourrait utiliser les informations des cartes routières et des informations sur la densité de trafic sur les routes pour évaluer la fiabilité et ajuster les résultats. Par exemple, un nid-de-poule rapporté et situé loin des routes doit être rejeté, ou si deux nids-de-poule ont le même nombre de rapports, celui qui a la densité de véhicule la plus élevée est le moins fiable, etc.

Bibliographie

[1] Kashif Ali et al. « Crowdits : Crowdsourcing in intelligent transportation systems ». In : *2012 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE. 2012, p. 3307–3311.

[2] E Alpaydin. *Introduction to Machine Learning, 2nd edn. Adaptive Computation and Machine Learning*. 2010.

[3] Siddharth Arora et al. « Detecting and monitoring the symptoms of Parkinson's disease using smartphones : a pilot study ». In : *Parkinsonism & related disorders* 21.6 (2015), p. 650–653.

[4] Ravi Bhoraskar et al. « Wolverine : Traffic and road condition estimation using smartphone sensors ». In : *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*. IEEE. 2012, p. 1–6.

[5] Kendrick Boyd, Kevin H Eng et C David Page. « Area under the precision-recall curve : point estimates and confidence intervals ». In : *Joint European conference on machine learning and knowledge discovery in databases*. Springer. 2013, p. 451–466.

[6] Muhammed Fatih Bulut, Murat Demirbas et Hakan Ferhatosmanoglu. « Lineking : Coffee shop wait-time monitoring using smartphones ». In : *IEEE Transactions on Mobile Computing* 14.10 (2014), p. 2045–2058.

[7] Yabo Cao, Yujiu Yang et WenHuang Liu. « E-FallD : A fall detection system using android-based smartphone ». In : *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*. IEEE. 2012, p. 1509–1513.

[8] Marianna Capecci et al. « A smartphone-based architecture to detect and quantify freezing of gait in Parkinson's disease ». In : *Gait & posture* 50 (2016), p. 28–33.

[9] Miguel A Carreira-Perpinán. « A review of mean-shift algorithms for clustering ». In : *arXiv preprint arXiv :1503.00687* (2015).

[10] Miguel A Carreira-Perpinan. « Gaussian mean-shift is an EM algorithm ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.5 (2007), p. 767–776.

[11] Yizong Cheng. « Mean shift, mode seeking, and clustering ». In : *IEEE transactions on pattern analysis and machine intelligence* 17.8 (1995), p. 790–799.

[12] Dorin Comaniciu. « An algorithm for data-driven bandwidth selection ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.2 (2003), p. 281–288.

[13] Dorin Comaniciu et Peter Meer. « Mean shift analysis and applications ». In : *Proceedings of the Seventh IEEE International Conference on Computer Vision*. T. 2. IEEE. 1999, p. 1197–1203.

[14] Dorin Comaniciu, Visvanathan Ramesh et Peter Meer. « The variable bandwidth mean shift and data-driven scale selection ». In : *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. T. 1. IEEE. 2001, p. 438–445.

[15] Fengyu Cong et al. « Applying wavelet packet decomposition and one-class support vector machine on vehicle acceleration traces for road anomaly detection ». In : *International Symposium on Neural Networks*. Springer. 2013, p. 291–299.

[16] Corinna Cortes et Vladimir Vapnik. « Support-vector networks ». In : *Machine learning* 20.3 (1995), p. 273–297.

[17] Ian F Darwin. *Android Cookbook : Problems and Solutions for Android Developers*. " O'Reilly Media, Inc.", 2017.

[18] Ingrid Daubechies. « Ten Lectures on Wavelets (Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992) ». In : *Google Scholar* (1992).

[19] *Documentation for app developers*. `https://developer.android.com/docs`. Accessed : 2019-09-10.

[20] Christopher Drane, Malcolm Macnaughtan et Craig Scott. « Positioning GSM telephones ». In : *IEEE Communications magazine* 36.4 (1998), p. 46–54.

[21] Jakob Eriksson et al. « The pothole patrol : using a mobile sensor network for road surface monitoring ». In : *Proceedings of the 6th international conference on Mobile systems, applications, and services*. ACM. 2008, p. 29–39.

[22] Keinosuke Fukunaga et Larry Hostetler. « The estimation of the gradient of a density function, with applications in pattern recognition ». In : *IEEE Transactions on information theory* 21.1 (1975), p. 32–40.

[23] Raghu K Ganti, Fan Ye et Hui Lei. « Mobile crowdsensing : current state and future challenges ». In : *IEEE Communications Magazine* 49.11 (2011), p. 32–39.

[24] FAA Garcia. « Tests to identify outliers in data series, Pontifical Catholic University of Rio de Janeiro ». In : *Industrial Engineering Department, Rio de Janeiro, Brazil* (2012).

[25] Cyril Goutte et Eric Gaussier. « A probabilistic interpretation of precision, recall and F-score, with implication for evaluation ». In : *European Conference on Information Retrieval*. Springer. 2005, p. 345–359.

[26] Frank E Grubbs. « Procedures for detecting outlying observations in samples ». In : *Technometrics* 11.1 (1969), p. 1–21.

[27] Mohammad Habib et al. « Smartphone-based solutions for fall detection and prevention : challenges and open issues ». In : *Sensors* 14.4 (2014), p. 7181–7208.

[28] Yi He, Ye Li et Shu-Di Bao. « Fall detection by built-in tri-accelerometer of smartphone ». In : *Proceedings of 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics*. IEEE. 2012, p. 184–187.

[29] John H Holland. « Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence ». In : *Ann Arbor, MI : University of Michigan Press* (1975), p. 439–444.

[30] *Introducing JSON[EB/OL]*. `https://www.json.org/`. Accessed : 2019-09-10.

[31] Mattias Johansson. *Estimering av GPS pålitlighet och GPS/INS fusion*. 2013.

[32] Salil S Kanhere. « Participatory sensing : Crowdsourcing data from mobile smartphones in urban spaces ». In : *2011 IEEE 12th International Conference on Mobile Data Management*. T. 2. IEEE. 2011, p. 3–6.

[33] Taehyeong Kim et Seung-Ki Ryu. « Review and analysis of pothole detection methods ». In : *Journal of Emerging Trends in Computing and Information Sciences* 5.8 (2014), p. 603–608.

[34] Emmanouil Koukoumidis, Li-Shiuan Peh et Margaret Rose Martonosi. « SignalGuru : leveraging mobile phones for collaborative traffic signal schedule advisory ». In : *Proceedings of the 9th international conference on Mobile systems, applications, and services*. ACM. 2011, p. 127–140.

[35] John R Koza. « Genetic programming ». In : (1997).

[36] Richard B Langley et al. « Dilution of precision ». In : *GPS world* 10.5 (1999), p. 52–59.

[37] Zhaojian Li et al. « A new clustering algorithm for processing GPS-based road anomaly reports with a mahalanobis distance ». In : *IEEE Transactions on Intelligent Transportation Systems* 18.7 (2017), p. 1980–1988.

[38] Hong Lu et al. « Stresssense : Detecting stress in unconstrained acoustic environments using smartphones ». In : *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM. 2012, p. 351–360.

[39] Prasanta Chandra Mahalanobis. « On the generalized distance in statistics ». In : National Institute of Science of India. 1936.

[40] Nicolas Maisonneuve et al. « NoiseTube : Measuring and mapping noise pollution with mobile phones ». In : *Information technologies in environmental engineering*. Springer, 2009, p. 215–228.

[41] Artis Mednis et al. « Real time pothole detection using android smartphones with accelerometers ». In : *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. IEEE. 2011, p. 1–6.

[42] Emiliano Miluzzo et al. « CenceMe–injecting sensing presence into social networking applications ». In : *European Conference on Smart Sensing and Context*. Springer. 2007, p. 1–28.

[43] Emiliano Miluzzo et al. « Sensing meets mobile social networks : the design, implementation and evaluation of the cenceme application ». In : *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM. 2008, p. 337–350.

[44] Prashanth Mohan, Venkata N Padmanabhan et Ramachandran Ramjee. « Nericell : rich monitoring of road and traffic conditions using mobile smartphones ». In : *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM. 2008, p. 323–336.

[45] Guy P Nason et Bernard W Silverman. « The stationary wavelet transform and some statistical applications ». In : *Wavelets and statistics*. Springer, 1995, p. 281–299.

[46] Matt Neuburg. *IOS 9 Programming Fundamentals with Swift : Swift, Xcode, and Cocoa Basics*. " O'Reilly Media, Inc.", 2015.

[47] Van Khang Nguyen, Éric Renault et Viet Hai Ha. « Road Anomaly Detection Using Smartphone : A Brief Analysis ». In : *International Conference on Mobile, Secure, and Programmable Networking*. Springer. 2018, p. 86–97.

[48] Nurzhan Nurseitov et al. « Comparison of JSON and XML data interchange formats : a case study. » In : *Caine* 9 (2009), p. 157–162.

[49] Umut Ozertem, Deniz Erdogmus et Robert Jenssen. « Mean shift spectral clustering ». In : *Pattern Recognition* 41.6 (2008), p. 1924–1938.

[50] L Pepa et al. « Smartphone based fuzzy logic freezing of gait detection in parkinson's disease ». In : *2014 IEEE/ASME 10th International Conference on Mechatronic and Embedded Systems and Applications (MESA)*. IEEE. 2014, p. 1–6.

[51] Mikko Perttunen et al. « Distributed road surface condition monitoring using mobile phones ». In : *International Conference on Ubiquitous Intelligence and Computing*. Springer. 2011, p. 64–78.

[52] David Martin Powers. « Evaluation : from precision, recall and F-measure to ROC, informedness, markedness and correlation ». In : (2011).

[53] *Programming with Objective-C*. `https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html`. Accessed : 2019-09-10.

[54] Dorin Comaniciu Visvanathan Ramesh et Peter Meer. « The variable bandwidth mean shift and data-driven scale selection ». In : *Eigth International Conference on Computer Vision*. 2001, p. 438–445.

[55] SR Jino Ramson et D Jackuline Moni. « Applications of wireless sensor networks—A survey ». In : *2017 International Conference on Innovations in Electrical, Electronics, Instrumentation and Media Technology (ICEEIMT)*. IEEE. 2017, p. 325–329.

[56] Rajib Kumar Rana et al. « Ear-phone : an end-to-end participatory urban noise mapping system ». In : *Proceedings of the 9th ACM/IEEE international*

conference on information processing in sensor networks. ACM. 2010, p. 105–116.

[57] Sasank Reddy et al. « Image browsing, processing, and clustering for participatory sensing : lessons from a DietSense prototype ». In : *Proceedings of the 4th workshop on Embedded networked sensors*. ACM. 2007, p. 13–17.

[58] Lukas Ruge, Bashar Altakrouri et Andreas Schrader. « Soundofthecity-continuous noise monitoring for a healthy city ». In : *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. IEEE. 2013, p. 670–675.

[59] Yutaka Sasaki et al. « The truth of the F-measure ». In : *Teach Tutor mater* 1.5 (2007), p. 1–5.

[60] Fatjon Seraj et al. « RoADS : A road pavement monitoring system for anomaly detection using smart phones ». In : *Big data analytics in the social and ubiquitous context*. Springer, 2014, p. 128–146.

[61] Claude Elwood Shannon. « A mathematical theory of communication ». In : *Bell system technical journal* 27.3 (1948), p. 379–423.

[62] *Statista, Smartphone Users Worldwide 20146–2021. Statista Inc., Hamburg, Germany*. `https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide`. Accessed : 2019-09-10.

[63] Girts Strazdins et al. « Towards vehicular sensor networks with android smartphones for road surface monitoring ». In : *2nd International Workshop on Networks of Cooperating Objects (CONET'11), Electronic Proceedings of CPS Week*. T. 11. 2011, p. 2015.

[64] *Swift.org - Documentation*. `https://swift.org/documentation/`. Accessed : 2019-09-10.

[65] N Tanaka, H Okamoto et M Naito. « Detecting and evaluating intrinsic nonlinearity present in the mutual dependence between two variables ». In : *Physica D : Nonlinear Phenomena* 147.1-2 (2000), p. 1–11.

[66] Arvind Thiagarajan et al. « VTrack : accurate, energy-aware road traffic delay estimation using mobile phones ». In : *Proceedings of the 7th ACM conference on embedded networked sensor systems*. ACM. 2009, p. 85–98.

[67] Chris Thompson et al. « Using smartphones to detect car accidents and provide situational awareness to emergency responders ». In : *International Conference on Mobile Wireless Middleware, Operating Systems, and Applications*. Springer. 2010, p. 29–42.

[68] CCJM Tiberius et Kai Borre. « Are GPS data normally distributed ». In : *Geodesy Beyond 2000*. Springer, 2000, p. 243–248.

[69] Astarita Vittorio et al. « Automated sensing system for monitoring of road surface quality by mobile devices ». In : *Procedia-Social and Behavioral Sciences* 111 (2014), p. 242–251.

[70] Guanhua WANG. « Improving data transmission in web applications via the translation between XML and JSON ». In : *2011 Third International Conference on Communications and Mobile Computing*. IEEE. 2011, p. 182–185.

[71] Jules WHITE et al. « Wreckwatch : Automatic traffic accident detection and notification with smartphones ». In : *Mobile Networks and Applications* 16.3 (2011), p. 285–303.

[72] Kuo-Lung WU et Miin-Shen YANG. « Mean shift-based clustering ». In : *Pattern Recognition* 40.11 (2007), p. 3035–3052.

[73] M ZHU. *Recall, precision and average precision. Department of Statistics and Actuarial Science, University of Waterloo*. Rapp. tech. Waterloo Working paper, 2004.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Détection et agrégation d'anomalies dans les données issues des capteurs placés dans des *smartphones*

Mots clés : système de capteurs de smartphones, détection d'anomalies, agrégation d'anomalies, crowdsensing

Résumé : Aujourd'hui, les *smartphones* sont devenus extrêmement populaires et équipés de nombreux capteurs tels qu'une caméra, un microphone, un capteur GPS, un accéléromètre, un magnétomètre, etc. Les utilisateurs de *smartphones* se connectent de plus en plus souvent à l'Internet. Par conséquent, il existe de nombreuses recherches sur les systèmes de capteurs à base de *smartphones* dans de nombreux domaines d'application.
Nos recherches visent à construire un système de capteurs à base de *smartphones* pour détecter les anomalies environnementales. Nous avons développé un tel système et améliorons les composants principaux: méthode de détection des anomalies par le *smartphone* et méthode de synthèse des données issues des anomalies au centre de calcul.
Pour les méthodes de détection des anomalies, nous nous appuyons sur les recherches actuelles de détection des anomalies routières. En fait, une méthode efficace, économe en énergie et répondant bien à différents types d'environnements et périphériques est nécessaire. Par conséquent, nous avons proposé une méthode de détection des anomalies basée sur le test des valeurs aberrantes en statistique. Les algorithmes améliorés ont une faible complexité et utilisent moins de mémoire. Nos expériences sur des données réelles montrent que nos méthodes s'adaptent bien aux conditions différentes dans le cas de la détection d'anomalies routières.
L'agrégation des données des anomalies n'a pas fait l'objet de beaucoup de travaux. Les recherches actuelles ne s'arrêtent qu'à des méthodes de regroupement simples basées sur la distance. Nous avons donc construit une méthode d'intégration des données permettant de trouver plus précisément l'emplacement de ces anomalies. Nous avons développé deux algorithmes. Le premier réalise une mise en cluster simple et fonctionne de manière cumulative. Cet algorithme décompose les données afin d'accroître la vitesse et la précision pour la localisation des anomalies. Le second permet de trouver des anomalies en se basant sur la recherche des modes de densité de probabilité. Nous avons également proposé un moyen d'attribuer une pondération aux emplacements trouvés pour évaluer la fiabilité. Notre expérience de simulation a confirmé l'efficacité de cet algorithme.

Title : Detection and aggregation of anomalies in data from smartphone sensors

Keywords : smartphone sensing, anomaly detection, anomaly aggregation, crowdsensing

Abstract : Today, smartphones have become extremely popular and equipped with many sensors such as camera, microphone, GPS sensor, accelerometer, magnetometer, etc. Smartphone users connect more and more often to the Internet. As a result, there is a great deal of research into smartphone sensor systems in many areas of application.
Our research aims at building a smartphone sensor system to detect environmental anomalies. We built the system model and improved the main components: anomaly detection method for the smartphone and method of synthesis of anomaly in the data center.
For anomaly detection methods, we rely on current research for the detection of traffic anomalies. In fact, there is a need for an efficient method that responds well to different environments and devices but consumes few resources. Therefore, we propose a method for anomaly detection based on the test of outliers in statistics. The improved algorithms have low complexity and use less memory. Our experiments on real data show that our methods adapt well to the different conditions in the case of the detection of road anomalies.
The aggregation of anomaly data has not received much attention. Current research only stops at the simple grouping method based on distance. We developed a data integration method that makes it possible to find more precisely the location of anomalies. We implemented two algorithms. A simple clustering algorithm that works cumulatively. This algorithm decomposes the data to increase speed and accuracy for anomaly location. Another algorithm makes it possible to find anomalies based on the search for probability density modes. We also propose a way to assign a weight to the found locations to assess reliability. Our simulations confirmed the effectiveness of this algorithm.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France