Topological Data Analysis and Statistical Learning for measuring pedestrian activities from inertial sensors

Bertrand Beaufils

université
PARIS-SACLAY

Topological Data Analysis and Statistical Learning for measuring pedestrian activities from inertial sensors

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 574, École doctorale de mathématiques Hadamard (EDMH)
Spécialité de doctorat: Mathématiques aux interfaces
Unité de recherche: Université Paris-Saclay, CNRS, Laboratoire de mathématiques d'Orsay, 91405, Orsay, France.
Référent: : Faculté des sciences d'Orsay

Thèse présentée et soutenue à Orsay, le 18/06/2020, par

Bertrand BEAUFILS

Composition du jury:

Elisabeth Gassiat Professeure des Universités, Université Paris-Saclay	Présidente
Charles Bouveyron Professeur des Universités, Université Côte d'Azur	Rapporteur
Fabrice Rossi Professeur des Universités, Université Paris-Dauphine	Rapporteur
Mathilde Mougeot Professeur des Universités, ENSIIE	Examinatrice
Frédéric Chazal Directeur de Recherche, INRIA Saclay	Directeur
Bertrand Michel Professeur des Universités, Ecole centrale de Nantes	Codirecteur
Marc Grelet Ingénieur de Recherche, Sysnav	Invité

sysnav NAVIGATION TECHNOLOGIES

Inria

CENTRALE NANTES

Thèse de doctorat

NNT: 2020UPASS107

université PARIS-SACLAY

ÉCOLE DOCTORALE
Physique et ingénierie : électrons, photons, sciences du vivant (EOBE)

Titre: Méthodes topologiques et apprentissage statistique pour l'actimétrie du piéton à partir de données de mouvement

Mots clés: Apprentissage statistique, Analyse topologique des donnéees, Reconnaissance d'activité, Séries temporelles multivariées, Unité de mesure inertielle

Résumé: Cette thèse s'intéresse à la détection de mouvements spécifiques à partir du dispositif ActiMyo développé par la société Sysnav, système de capteurs inertiels miniatures bas-coût pouvant se porté à la cheville et au poignet. En particulier, une approche d'apprentissage statistique supervisé vise à détecter les foulées dans les enregistrements cheville. Ce premier travail, combiné avec un algorithme breveté par l'entreprise Sysnav, permet de reconstruire la trajectoire du piéton. Cette trajectoire est ensuite utilisée dans une nouvelle méthode d'apprentissage supervisé pour la reconnaissance d'activité qui est une précieuse information notamment dans un contexte médical. Ces deux algorithmes proposent une approche innovante basée sur l'alignement des signaux inertiels et l'extraction d'intervalles candidats qui sont ensuite classés par l'algorithme de Gradient Boosting Trees. Le manuscrit présente également une architecture de réseaux de neurones combinant des channels de convolution et d'analyse topologique des données pour la détection de mouvements caractéristiques de la maladie de Parkinson tels que les tremblements et crises de dyskinésie.

Title: Topological Data Analysis and Statistical Learning for measuring pedestrian activities from inertial sensors

Keywords: Machine Learning, Topological data analysis, Activity recognition, Multivariate time series, Inertial measurement unit

Abstract: This thesis focuses on the detection of specific movements using ActiMyo, a device developed by the company Sysnav. This system is composed by low-cost miniature inertial sensors that can be worn on the ankle and wrist. In particular, a supervised statistical learning approach aims to detect strides in ankle recordings. This first work, combined with an algorithm patented by Sysnav, allows to compute the trajectory of the pedestrian. This trajectory is then used in a new supervised learning method for the activity recognition, which is valuable information, especially in a medical context. These two algorithms offer an innovative approach based on the alignment of inertial signals and the extraction of candidate intervals which are then classified by the Gradient Boosting Trees algorithm. This thesis also presents a neural network architecture combining convolutional channels and topological data analysis for the detection of movements representative of Parkinson's disease such as tremors and dyskinesia crises.

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

Remerciements

L'environnement dans lequel j'ai évolué durant presque quatre ans chez Sysnav a été idéal. L'expérience de l'entreprise dans l'accompagnement de thèse CIFRE et la disponibilité de mes directeurs de thèse et leur investissement m'ont permis de m'épanouir et d'avancer sereinement dans mon travail.

Je remercie tout d'abord Bertrand, qui a su me conseiller dans mon parcours scolaire et professionnel, toujours avec beaucoup de bienveillance. C'est grâce à toi que j'ai découvert le monde de l'apprentissage statistique et je t'en serai éternellement reconnaissant. Merci également à Frédéric qui a bien voulu m'accepter parmi ses thésards et me faire confiance alors qu'on ne s'était jamais rencontré. Votre aide tout au long de ces années a été précieuse. Vous m'avez beaucoup apporté d'un point de vue technique et vous m'avez également toujours soutenu lors de passages plus délicats. Vous vous êtes rendus disponibles à chaque fois que j'en avais besoin malgré vos emplois du temps chargés. Je mesure la chance de vous avoir eus en tant qu'encadrants, merci.

Un thèse CIFRE ne peut être une réussie sans la qualité d'accueil de l'entreprise. Je suis sincèrement reconnaissant envers David et Marc qui m'ont fait confiance et m'ont laissé carte blanche pour développer les solutions. Marc qui m'a impressioné par sa capacité à appréhender les problématiques de machine learning jusqu'à être force de proposition. Ton professionalisme, ta qualité d'encadrement, ta vitesse d'esprit en plus de ta gentillesse m'ont permis de travailler dans un cadre privilégié.

Je remercie également Charles Bouveyron et Fabrice Rossi pour m'avoir fait l'honneur de rapporter ma thèse. Merci pour votre lecture attentive, pour vos remarques constructives ainsi que pour l'intérêt que vous portez à mes travaux. Merci à Elisabeth Gassiat et Mathilde Mougeot pour avoir accepté de participer à mon jury ainsi que pour vos questions et remarques très intéressantes en fin de soutenance.

Pour conclure, je souhaite remercier mes proches et ma famille. Merci à mes parents et à mon frère pour leurs encouragements durant ces quatre années, surtout pendant ce Noël 2019 agité par la rédaction du manuscrit. Je suis maintenant le plus haut diplômé de la famille, qui l'eût cru ! J'ai une pensée émue pour mes grands parents, j'aurais adoré partager ce moment avec vous. Enfin je remercie Maeva pour son amour. Merci de m'avoir soutenu et d'avoir accepté de mettre de côté notre vie sociale à certains moments de cette aventure.

Contents

I Introduction (Français) 11

1 Contexte général et enjeux majeurs de la thèse 13
- 1.1 Unité de Mesure Inertielle pour la reconstruction de trajectoire 13
- 1.2 Unité de Mesure Inertielle pour des applications médicales . . 14

2 ActiMyo: système de capteurs magnéto-inertiels 14
- 2.1 IMU à composants liés 15
- 2.2 Calibration des capteurs inertiels 15

3 Variables cliniques pour la Dystrophie Musculaire de Duchenne . . . 15
- 3.1 Présentation de la maladie 15
- 3.2 Tests en hôpital 16
- 3.3 Variables à domicile avec l'ActiMyo 16

4 Navigation à l'estime pour le piéton avec un IMU sur la chaussure . . 16
- 4.1 Détection des instants de vitesse nulle basée sur des valeurs seuil 17
- 4.2 Apprentissage automatique pour la détection des vitesses nulles 18
- 4.3 Reconstruction de trajectoire à partir des instants de vitesse nulle 18
- 4.4 Innovation Sysnav : navigation à l'estime pour un IMU positionné à la cheville 19
- 4.5 Reconnaissance d'activité (AR) 22

5 Variables cliniques pour la maladie de Parkinson 22
- 5.1 Présentation de la maladie 23
- 5.2 Détection automatique d'évènements parkinsoniens 23
- 5.3 Détection automatique des tremblements et crises de dyskinésie avec l'ActiMyo positionné à la cheville 23

6 Contributions générales de la thèse 24
- 6.1 Apprentissage statistique supervisé pour la détection de foulées et des évènements parkinsoniens 24
- 6.2 Extraction d'intervalles candidats basée sur le contact au sol . 25
- 6.3 Extraction d'intervalles candidats de fouléees basée sur une pseudo-vitesse 26
- 6.4 Reconnaissance d'activité à partir de la trajectoire reconstruite 27
- 6.5 Réseau de neurones pour la détection des évènements parkinsoniens 27

I Introduction (English) 29

1 General context and main issues of the thesis 31
- 1.1 Inertial Measurement Unit for Trajectory Reconstruction 31
- 1.2 Inertial Measurement Unit for Healthcare Application 32

2 ActiMyo: magneto-inertial sensors system 32
- 2.1 Strapdown IMU 33

2.2 Inertial sensors calibration . . . 33
3 Clinical outcomes for Duchenne Muscular Dystrophy . . . 33
3.1 Presentation of the disease . . . 33
3.2 Hospital tests . . . 34
3.3 Home variables with ActiMyo . . . 34
4 Introduction to PDR with shoe-mounted IMU . . . 34
4.1 Threshold-based Zero Velocity Update Detection . . . 35
4.2 Machine Learning for Zero Velocity Update Detection . . . 36
4.3 Trajectory Reconstruction from Zero Velocity Update Detection 36
4.4 Sysnav Inovation: PDR for ankle-mounted IMU . . . 37
4.5 Human Activity Recognition (HAR) . . . 39
5 Clinical outcomes for Parkinson disease . . . 40
5.1 Presentation of the disease . . . 40
5.2 Automatic detection of Parkinson's events . . . 40
5.3 Automatic detection of ankle tremors and dyskinesia with ActiMyo . . . 41
6 General Contributions and Thesis Outline . . . 41
6.1 Supervised Learning for stride detection and Parkinson's events 42
6.2 Candidate stride interval extraction based on ground contact . 42
6.3 Candidate stride interval extraction based on pseudo-speed . 43
6.4 Activity recognition from computed stride trajectory . . . 44
6.5 Neural Network for Parkinson's events detection . . . 44

II Supervised learning for stride detection and Parkinson's events detection 47
1 Introduction to Statistical supervised learning . . . 49
1.1 Empirical Risk Minimization . . . 49
1.2 Bias-Variance Trade-Off in Machine Learning . . . 50
1.3 Risk Estimation . . . 51
2 Ensemble methods . . . 52
2.1 Introduction to Boosting Algorithm through Adaboost . . . 52
2.2 Gradient Boosting Trees . . . 54
3 Neural Networks . . . 55
3.1 Transfer function for MLP . . . 55
3.2 Parameters Optimization through Backpropagation . . . 57
3.3 Convolutional Neural Networks . . . 58
3.4 Topological Data Analysis Channel . . . 62
4 Applications to ActiMyo data . . . 67
4.1 Databases description . . . 68
4.2 Candidate intervals extraction for stride detection . . . 72
4.3 Activity Recognition from computed trajectory . . . 74
4.4 Parkinson's event detection with sliding window . . . 75

III Candidate stride interval extraction based on ground contact 79
1 Intervals Extraction Method . . . 81
2 Features engineering . . . 83
2.1 Sensors alignment . . . 83
2.2 Swing modelling . . . 85
2.3 Functional data analysis . . . 91
3 Intervals classification with GBT . . . 91

3.1 Cross-validation performance . . . 91
3.2 False negative rate in MOCAP . . . 92
4 Contributions . . . 94

IV Candidate stride interval extraction based on pseudo-speed 97
1 Terrestrial Reference Frame Computation . . . 99
2 Pseudo-speed Computation . . . 101
2.1 Pseudo-speed norm extrema for interval extraction . . . 102
3 Features engineering . . . 104
3.1 Speed estimation at extracted interval boundaries . . . 105
3.2 Pseudo-trajectory and sensors alignment . . . 105
3.3 Functional data analysis . . . 107
4 Intervals classification with GBT . . . 107
4.1 Cross-validation performance . . . 107
4.2 False Negative Rate in MOCAP . . . 108
4.3 False Positive Rate . . . 109
5 Contributions . . . 109

V Activity Recognition from Computed Trajectory 111
1 Trajectory Reconstruction of the Detected Strides . . . 113
2 Trajectory reconstruction performance . . . 115
2.1 Stride length performance in MOCAP . . . 115
2.2 Performance in Uncontrolled Environment . . . 116
3 Activity Recognition of the Detected Strides with Machine Learning from the Computed Trajectory . . . 118
3.1 Cross-validation performance with GBT . . . 118
3.2 Algorithm Overview . . . 119
3.3 Activity Recognition in Controlled Environments . . . 119
3.4 Activity Recognition for One Healthy Child Recording in Uncontrolled Environment . . . 121
4 Contributions . . . 123

VI Parkinson's event detection with sliding window 125
1 Data Background . . . 127
1.1 Parkinson's event dataset and HAPT dataset . . . 127
1.2 Sliding Window approach . . . 128
2 Architecture description and contribution of the channels . . . 128
2.1 Learning protocol . . . 129
2.2 Convolutional Neural Networks . . . 130
2.3 Handcrafted Features Channel . . . 131
2.4 Topological Data Analysis Channel . . . 133
3 Parkinson's statistics for home recordings . . . 135
4 Contributions . . . 135

VII General Conclusions 137
1 Summary of main contributions with machine learning approaches . 139
1.1 Stride detection from candidate intervals extraction . . . 139
1.2 Activity Recognition from the computed trajectory . . . 140

1.3 Dyskinesia and tremor detection 140
2 Extensions to industrial applications 141
2.1 Beacons for trajectory reconstruction 141
2.2 Stride detection algorithm in embedded system 142
3 Future work: Domain Adaptation for Activity Recognition 142
3.1 Joint Distribution Optimal Transportation 144

Conventions and Notations

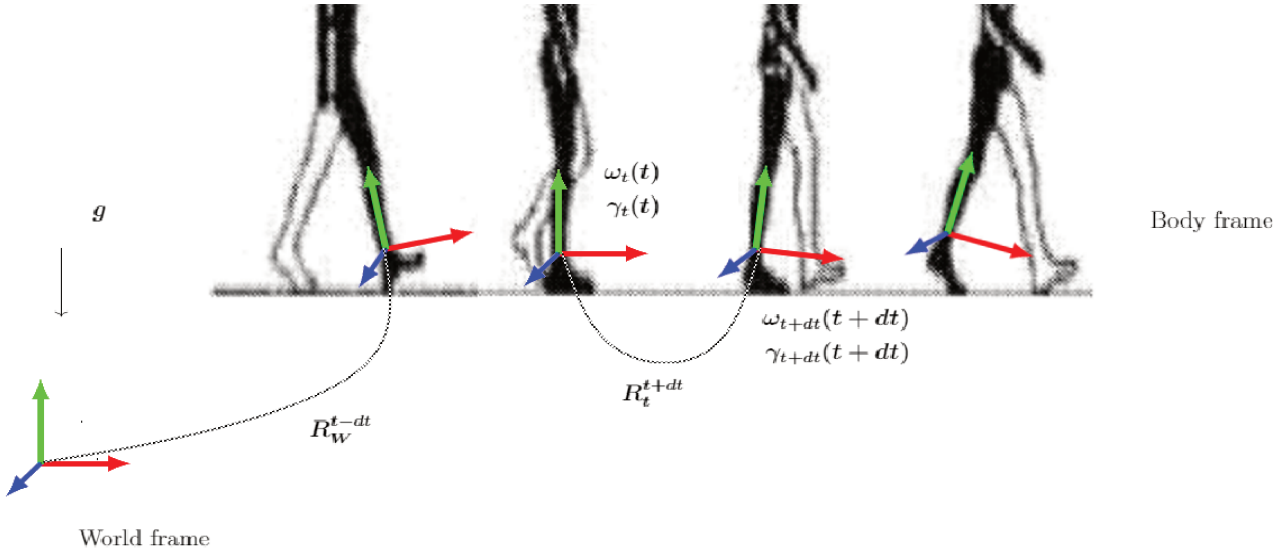


Figure 1: Reference coordinate frames in the general problem with associated typical measurements.

Reference frames convention and rotation matrix parametrizations

The World coordinate are defined such that the gravity vector $\boldsymbol{g} \simeq (0, 0, -9.81)^T$. The reference frame in which the quantity is expressed is noted in index: W stands for the worlds gravity-aligned reference frame, t for the current body reference frame at time t. For rotations, we use the convention that $\boldsymbol{R_t^{t+dt}}$ transforms a quantity from body frame at time $t + dt$ to body frame at time t and the rotation matrix $\boldsymbol{R_W^T}$ transforms a quantity from body frame at time t to world frame by left multiplying it. The figure 1 summarizes the chosen notations.

Matrix notations and operations

For a matrix $\boldsymbol{X} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_j, \ldots, \boldsymbol{X}_n) = (X_{ij})$, $\boldsymbol{X}_j$ denotes the column j vector and X_{ij} the element of the row i and column j.
The multiplication operation between $\boldsymbol{X} = (X_{ij})$ taking values in $\mathbb{R}^{m\times n}$ and $\boldsymbol{Y} = (Y_{ij})$ taking values in $\mathbb{R}^{n\times p}$ is noted $\boldsymbol{XY} = \boldsymbol{Z}$ and is given by:

$$\forall i, j : Z_{ij} = \sum_{k=1}^{n} A_{ik}B_{kj} = A_{i1}B_{1j} + A_{i2}B_{2j} + \ldots + A_{in}B_{nj}$$

The cross product between $\boldsymbol{X} = (X_1, X_2, X_3)$ and $\boldsymbol{Y} = (Y_1, Y_2, Y_3)$ taking both values in $\mathbb{R}^3$ is noted $\boldsymbol{X} \times \boldsymbol{Y}$ and is given by:

$$\boldsymbol{X} \times \boldsymbol{Y} = \begin{pmatrix} X_2Y_3 - X_3Y_2 \\ X_3Y_1 - X_1Y_3 \\ X_1Y_2 - X_2Y_1 \end{pmatrix}$$

Symbols	
$\boldsymbol{g}$	gravity vector
$\boldsymbol{R}$	Rotation matrix
$\boldsymbol{v}$	Speed
$\boldsymbol{a}$	Specific acceleration
$\boldsymbol{\omega}$	Angular velocity measured by gyrometer
$\boldsymbol{\gamma}$	Global acceleration measured by accelerometer: $\boldsymbol{\gamma} = \boldsymbol{a} + \boldsymbol{g}$
$\hat{\boldsymbol{X}}$	Estimated version of quantity $\boldsymbol{X}$

Part I

Introduction (Français)

Ce premier chapitre vise à contextualiser les enjeux majeurs de cette thèse et de présenter une vue d'ensemble des algorithmes existants qui ont été intiallement développés pour répondre à ces problématiques. En Section I.1 nous décrivons les Unités de Mesure Inertielle (IMUs) et leur utilisation pour la reconstruction de trajectoire et des applications médicales. En particulier, nous présentons en Section I.2 le dispositif ActiMyo développé par Sysnav, destiné à être porté à la chevillet ou au poignet. En Section I.5 et Section I.3 nous décrivons les variables calculées par Sysnav visant à caractériser l'évolution de la santé des patients pendant les études cliniques pour deux maladies : Parkinson et la Dystrophie Musculaire de Duchenne (DMD). Les variables cliniques DMD sont basées sur la reconstruction de trajectoire du dispositif positionné à la cheville. En Section I.4 nous introduisons les notions de reconstruction de trajectoire pour un piéton équipé d'un IMU posé directement sur la chaussure. Cette approche a été adaptée au dispositif ActiMyo qui est positionné à la cheville. Que ce soit pour les variables cliniques DMD ou Parkinson, Sysnav a développé des algorithmes montrant plusieurs limitations pour lesquelles nous apporteront des solutions au long de cette thèse. En Section I.6 nous décrivons brièvement notre travail, divisé en trois applications utilisant du Machine Learning : la détection de foulées, la reconnaissance d'activité et la détection d'évènements parkinsoniens.

1 Contexte général et enjeux majeurs de la thèse

Dans cette section nous présentons deux types d'application utilisant des Systèmes de Mesure Inertiels (IMUs). En Section I.1.1 nous décrivons les avantages d'utiliser les IMUs pour la reconstruction de trajectoire, répondant notamment à plusieurs limitations de la Géolocalisation et Navigation par un Système de Satellites (GNSS) qui apparaissent dans les bâtiments par exemple. Ces dispositifs sont également de plus en plus utilisés dans un contexte médical pour mesurer l'évolution de la santé des patients à domicile. De part leur petite taille, il est en effet facile de les intégrer dans la vie de tous les jours.

1.1 Unité de Mesure Inertielle pour la reconstruction de trajectoire

L'émergence de la Géolocalisation et Navigation par un Système de Satellites (GNSS) dans les années 2000 a changé la perception du grand public de la géolocalisation. Alors que les systèmes GNNS sont beaucoup utilisés, dans de nombreuses situations ils ne parviennent pas à fournir une position précise à cause d'une mauvaise réception de signal (tunnels, parking sous-terrain, en forêt, à l'intérieur des bâtiments etc.). Cependant être capable de géolocaliser avec précision peut s'avérer vital pour les travailleurs isolés, pompiers ou des applications militaires.

Dans ce contexte les Unités de Mesure Inertielle (IMUs) peuvent être utilisés dans un Système de Navigation Inertielle (INS). L'avantage majeur de ces dispositifs est qu'ils ne dépendent d'aucune infrastructure externe une fois que le point d'initialisation est fixé. Ceci les rend particulièrement robustes aux perturbations extérieures ou sabotages, à l'inverse des architectures basées sur la GNSS. C'est pour cette raison que les INS sont largement utilisés dans des applications où une précise géolocalisation est critique comme pour les avions, bateaux, sous-marins, missile, fusées, satellites. Ils ont été de plus en plus développés pour ces applications industrielles et militaires au début des années 1950. La trajectoire est estimée par "navigation à l'estime" : un signal IMU (accélération et vitesse

angulaire) est intégré afin de calculer la position actuelle à partir d'une position initiale connue. Pour ce genre d'application, cette méthode demande d'utiliser des capteurs à haute précision qui sont très volumineux et très cher. Il est impossible d'en équiper un piéton. Dans ce cas, pour estimer la position avec des capteurs inertiels légers et peu coûteux comme ceux des téléphones portables par exemple, des méthodes différentes doivent être implémentées.

1.2 Unité de Mesure Inertielle pour des applications médicales

D'autres travaux à partir de données inertielles sont également entrepris dans un contexte médical. L'utilisation des INS en ingénierie biomédicale s'est rapidement popularisée pour l'analyse de mouvement. Les IMUs sont discrets et simples à porter à l'aide d'une sangle, les rendant faciles à intégrer dans diverses technologies portatives destinées aux applications de santé et bien-être à domicile. En effet ils sont capables de mesurer et de suivre les activités pour permettre ensuite d'évaluer la qualité de vie d'un individu. Ils peuvent être utilisés par exemple pour la détection de chute et de symptômes de patients souffrant de troubles du mouvement. Bien que la médecine promeut ce genre d'innovation, les études cliniques qui évaluent l'efficacité de nouveaux traitements sont réalisées dans des environnement très contrôlés. Jusqu'à 2019, aucun système n'avait été approuvé par l'Agence Européenne du Médicament (EMA) comme outil pour mesurer l'évolution de la santé des patients.

Dans ce contexte Sysnav a développé un système de capteurs magnéto-inertiels destinés à être porté à la cheville et au poignet. Ce dispositif a pour but de permettre le calcul de variables de trajectoire pour des patients souffrant de troubles du mouvement. Par exemple il est pertinent d'étudier la trajectoire de la cheville pour des maladies provoquant une perte progressive de la marche.

2 ActiMyo: système de capteurs magnéto-inertiels

L'ActiMyo est composé d'un accéléromètre et d'un gyromètre enregistrant les données (respectivement γ et ω) dans son propre référentiel (qu'on appelle référentiel "body") avec une fréquence de 130 Hz et une autonomie sur batterie de 16 heures. L'accéléromètre mesure à la fois l'accélération produite par le mouvement (a) mais mesure également en continue l'accélération de la gravité terrestre. Les capteurs inertiels commencent à enregistrer dès que la dispositif est retiré de son boîtier. Lorsqu'il est remis en place, les données sont transférées sur un serveur cloud auquel nous avons accès.

(**a**) (**b**) (**b**)

Figure 2: ActiMyo connecté à la station (**a**), positionné à la cheville (**b**) et au poignet (**c**).

2.1 IMU à composants liés

Le système de mesure inertiel utilisé dans ce dispositif appelé "à composants liés" est décrit en [60]. Dans ce type de IMU, l'accéléromètre et le gyromètre sont fixés de manière rigide sur une plate-forme solide, contrairement à une plate-forme gyrostabilisée où l'orientation de l'accéléromètre est stabilisée dans le référentiel inertiel. Ceci a une implication majeure : l'attitude n'est pas donnée directement par le gyroscope mais doit être intégrée à partir de la vitesse angulaire. Cela est fortement corrélé aux problèmes d'attitude et de position et représente une la majeure difficulté de la navigation inertielle. Cependant, de nos jours, les systèmes de navigation inertielle à composants liés suscitent un intérêt particulier. En effet ils sont plus faciles à construire mécaniquement, sont plus petits et plus légers. On les retrouve par exemple dans l'ensemble des smartphones du marché.

2.2 Calibration des capteurs inertiels

Les capteurs inertiels triaxiaux de l'ActiMyo sont conçus pour émettre un signal proportionnel à la quantité d'intérêt (vitesse angulaire pour le gyromètre et accélération pour l'accéléromètre). Cependant, leurs modèles doivent être calibrés individuellement avant de les utiliser comme mesures d'accélération ou de vitesse de rotation. De plus, la combinaison des trois axes n'est jamais totalement orthogonale en raison de la précision de l'assemblage mécanique. L'étalonnage peut également être influencé par des facteurs environnementaux comme la température et ces effets doivent également être pris en compte. Sysnav a développé un module interne qui calcule une erreur de compensation de calibration (facteur d'échelle, biais, non orthogonalité, etc.) pour chaque système fabriqué.

ActiMyo a été conçu pour être facile à utiliser pour les patients. Il nécessite peu de manipulation et peut être connecté directement à la station d'accueil afin que les données collectées chaque jour soient automatiquement envoyées à un serveur dédié pour une analyse hors ligne.

3 Variables cliniques pour la Dystrophie Musculaire de Duchenne

Dans cette section, nous présentons l'impacte de la maladie dans la vie quotidienne pour des patients souffrant de la dystrophie musculaire de Duchenne (Section I.5.1) ainsi que les variables "officielles"qui sont calculées à l'hôpital pour mesurer les symptômes pendant les études cliniques (Section I.3.2). Cependant, ces tests effectués à l'hôpital présentent de nombreux inconvénients ue Sysnav vise à surpasser avec l'ActiMyo (Section I.3.3).

3.1 Présentation de la maladie

la dystrophie Musculaire de Duchenne est une maladie génétique qui provoque une perte progressive de la fonctionnalité des fibres musculaires ce qui affecte la mobilité. Les dernières années ont vu une augmentation de l'intérêt des sociétés pharmaceutiques pour le lancement de recherche et développement dans de nouveaux traitements. On peut penser que de nouvelles thérapies permettront de ralentir la progression de la maladie afin de maintenir des bonnes conditions

de vie. Les études cliniques ciblent des patients myopathes qui sont suffisamment âgés pour être évalués de manière fiable et qui ont encore suffisamment de fibre musculaires pour démontrer une bénéfice sur la durée usuelle d'un essai clinique (autour d'un an).

3.2 Tests en hôpital

Durant les dernières années, des dizaines de milliards de dollars ont été investis pour créer des nouvelles molécules dans des laboratoires afin de développer des traitements innovants. Mais un aspect n'a jamais changé : les méthodes de mesures pour évaluer l'efficacité du traitement (variables cliniques). À ce jour, les variables cliniques consistent à chronométrer le temps minimal du patient pour monter marches (test des marches), le temps minimal du patient pour courir 10 mètres (test des 10 mètres de course) ou la distance maximale parcourue sur 6 minutes de marche. Le problème principal de ces variables cliniques est qu'elles peuvent être biaisées par la motivation du patient. En effet les résultats peuvent être impactés par la condition physique du jour sans pour autant être corrélés avec l'état de santé général. De plus, les erreurs humaines peuvent apparaître dans la prise de mesure. Par exemple, le test des 4 marches est parfois effectué en moins de deux secondes et la mesure prise sur le même test par deux docteurs peut varier de manière significative. Enfin, ces tests sont usants et parfois compliqués à réaliser pour des patients qui ne vivent pas proche de l'hôpital.

3.3 Variables à domicile avec l'ActiMyo

Il y a un fossé entre l'investissement mis dans les traitements comparés au besoin de définir des nouvelles variables cliniques plus robustes. Le but de l'ActiMyo est de calculer des variables cliniques corrélés avec les tests précédemment mentionnés mais dans un environnement non contrôlé : c'est à dire à domicile, à l'école, sur n'importe quel lieu du quotidien. Il est important de calculer des variables qui traduisent la véritable activité du patient sur plusieurs mois, et non pas sur des tests ponctuels réalisés en hôpital. Les variables proposées correspondent à la longueur des foulées, la vitesse des foulées et la distance des foulées [32, 46]. Afin de relier ces résultats aux tests des 4 marches et des 10 mètres de course, nous avons besoin de Reconnaissance l'Activité (AR).

Pour la longueur des foulée calculée à part d'IMU, la navigation à l'estime pour le piéton a montré qu'on pouvant atteindre des bonnes précisions dans de nombreuses applications industrielles. Dans ce qui suit, nous allons introduire les bases de la navigation à l'estime pour le piéton avec un IMU positionné sur la chaussure ainsi que l'algorithme innovant développé par Sysnav qui adapte la méthode utilisée pour un IMU sur la chaussure.

4 Navigation à l'estime pour le piéton avec un IMU sur la chaussure

Le principe de la navigation inertielle peut être résumée de manière très simple : estimer l'évolution de la position, l'orientation et la vitesse d'un objet dans un référentiel inertiel. Cette approche repose essentiellement sur l'intégration des accélérations et des vitesses angulaires pour estimer la position suivante

au cours du temps. Contrairement aux systèmes dépendant d'infrastructure extérieure comme le map matching, Wi-Fi [84], l'identification par radiofréquence [80], ou la technologie à bande ultralarge [41], Les IMUs peuvent être déployés rapidement et facilement (Figure 3).

Figure 3: PERSY (Predestrian Reference SYstem) développé par IFSTTAR.

Cependant, tous les IMUs à composants liés sont soumis à une dérive et les intégrations accumulent rapidement des erreurs importantes. Pour résoudre ce problème, la technique zero velocity update technique (ZUPT) [43, 5, 26, 74] est souvent utilisée. Le but de cette méthode est de détecter quand le ppied est complètement à plat par rapport au sol. En Section I.4.1 et Section I.4.2 nous introduisons deux principes pour détecter ces instants : les méthodes basées sur des valeurs seuil prédéfinies et celles utilisant de l'apprentissage automatique. À partir de cette détection, l'intégration des données inertielles se fait uniquement lorsque le pied est en mouvement en l'air (Figure 4) au lieu d'être réalisée sur l'ensemble de la durée de l'enregistrement. La procédure est décrite en Section I.4.3. Cependant, cette techniques n'est pas directement ultilisable avec

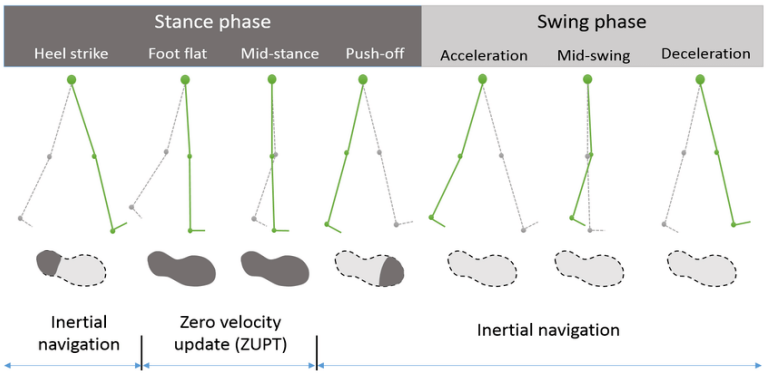


Figure 4: Les instants de vitesse nulle sur un cycle de marche.

l'ActiMyo car il est porté à la cheville et non sur le pied. C'est pourquoi Sysnav a développé une nouvelle méthode inspirée du ZUPT et présentée en Section I.4.4 mais qui comporte tout de même certaines limitations. En Section I.4.5 nous décrivons comment utiliser la trajectoire reconstruire de la cheville pour reconnaître l'activité pour une application médicale.

4.1 Détection des instants de vitesse nulle basée sur des valeurs seuil

Plusieurs études dans la littérature ont proposé d'optiiser des valeurs seuil pour les données inertielles (accélérations proche de 1 g et petites valeurs de vitesses angulaires) [58, 42, 1]. Un limitation bien connue de ces approches est qu'elles ne parviennent pas à être efficaces face à la grande variété de déplacements et

mouvements. Il est vrai que ces approches fonctionnent bien dans des situations de marche classique mais échouent pour des pas atypiques comme dans des escaliers ou des piétinements. Dans [69], plusieurs systèmes inertiels portés à la cheville ou poignet ont été testées pour détecter les foulées et la distance parcourue pendant la marche, dans des escaliers et durant des activités du quotidien en intéreieur. Tous les dispositifs ont montré une baisse de performance lorsque la marche était ralentie, en ratant un nombre important de foulées. Les mauvaises performances pour le calcul des distances ont aussi été démontrées pour des marches lentes ou des escaliers. Les approches récentes visent à améliorer la détection des foulées en adaptant les valeurs seuil vis à vis des accélérations observées [51, 77] ou de la fréquence du signal [83]. Cependant, modéliser une détection de foulées précise durant les phases d'escaliers ou la course reste un challenge [40].

4.2 Apprentissage automatique pour la détection des vitesses nulles

Dans le but de répondre aux limitations des approches basées sur des valeurs seuil, des récents travaux utilisent l'apprentissage automatique dans des détections de foulées avec une approche de fenêtre glissante [76, 6]. Ces méthodes consistent à construire une fonction de prédiction qui fournit une sortie binaire pour chaque échantillon de l'enregistrement : 1 si l'échantillon correspond à un instant de vitesse nulle, 0 sinon. La principal inconvénient de cette approche est que ces fonctions présentent un bon taux de détection mais avec beaucoup de fauux positifs lorsque le dispositif est en mouvement. Cela peut engendrer des erreurs importantes dans la reconstruction de la trajectoire. Pour maintenir des performances acceptables, les algorithmes de la littérature ajuste la sortie de la fonction de prédiction, par exemple en éliminant les instants de vitesse nulle prédits sans grande confiance (valeur seuil optimisée). Ce genre de compensation montre des bons résultats quand on sait que le piéton est en train de marche mais n'est pas robuste à la grande variété des activités du quotidien. En effet plusieurs foulées sont détectées à tort lorsque le pied bouge alors que la personne est assise ou fait du vélo etc. L'étude présentée en [69] montre que l'ensemble des dispositifs testés pour la détection des foulées comportent beaucoup de faux positifs pendant des activités basiques effectuées à domicile comme la lecture, les jeux de cartes etc.

4.3 Reconstruction de trajectoire à partir des instants de vitesse nulle

Dans le but de calculer la trajectoire, la stratégie qui consiste à intégrer les accélérations et vitesses angulaires enregistrées par les capteurs inertiels dérivent rapidement. La méthode de détection des instants de vitesse nulle limite les erreurs en nécessitant d'intégrer uniquement pendant les foulées détectées, en faisant l'hypothèse que la vitesse nulle apparaît au début et à la fin de la foulée. La détection des instants de vitesse nulle est incluse dans un filtre de Kalman étendu [57] avec une approche de navigation à l'estime (Figure 5). Ce filtre permet de réduire significativement les erreurs au cours du temps. Le filtre peut par exemple inclure un état à 6 degrés de liberté pour la vitesse et l'attitude. D'autres états peuvent être ajoutés comme la position, le biais du capteur etc. Il permet aussi de mesurer la confiance des états estimés.

La méthode de détection des instants de vitesse nulle (ZUPT) améliore la qualité de reconstruction de trajectoire mais repose sur une contrainte assez forte : le placement du dispositif sur la chaussure. Cet emplacement est très sensible

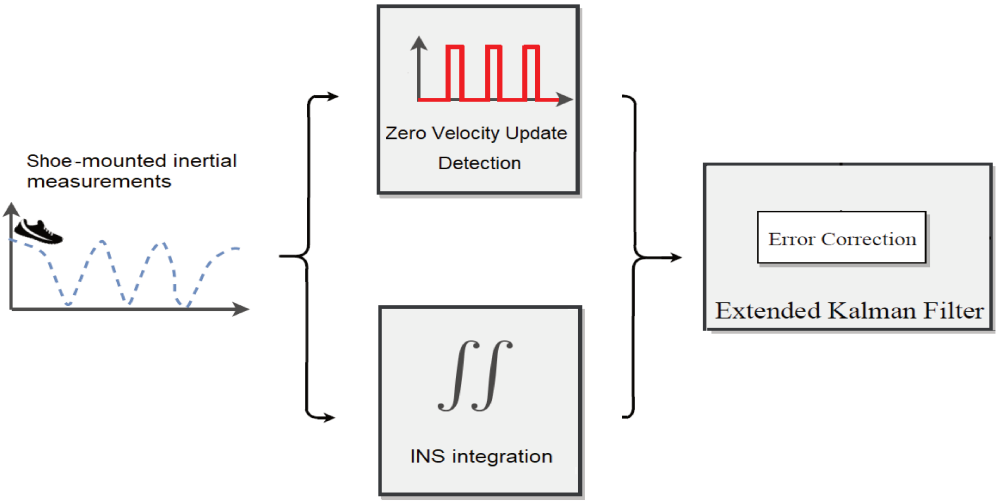
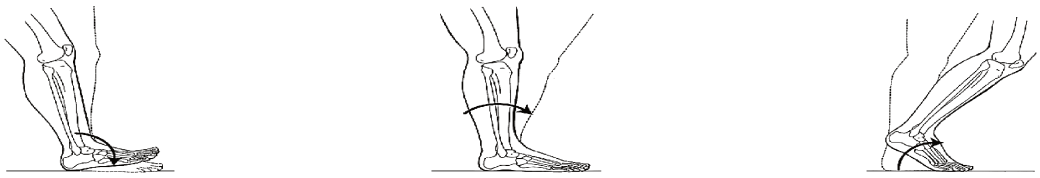


Figure 5: Détection des instants de vitesse nulle combinée avec la navigation à l'estime dans un filtre de Kalman étendu.

aux chocs provoqués par le contact du pied avec le sol et le système reste visible et compliqué à installer dans la vie de tous les jours. Par exemple, pour des enfants atteints de myopathie qui doivent se rendre à l'école, cela peut engendrer des moqueries de la part des autres élèves. Enfin, lorsque le piéton est en train de courir, ces instants de vitesse nulle lorsque le pied est au sol sont quasi inexistants ce qui empêche les corrections apportées par le filtre de Kalman. Le système ActiMyo est lui positionné à la cheville, ce qui permet notamment de passer le pantalon par dessus, le rendant invisible. Mais cela a une conséquence importante : la cheville n'est pas à vitesse nulle lorsque le pied est au sol.

4.4 Innovation Sysnav : navigation à l'estime pour un IMU positionné à la cheville

Comme les instants de vitesse nulle ne sont pas observés durant la phase de pied au sol (Figure 4), Sysnav a adapté le principe du ZUPT en développant un modèle basé sur le bras de levier pour estimer la vitesse de la cheville au début et à la fin de la foulée, ce qui permet de calculer les intégrations uniquement entre ces deux instants également. Quand le pied est à plat au sol, nous supposons que la cheville est en rotation pure autour du talon. Ainsi, la vitesse de la cheville est estimée par le produit vectoriel entre le vecteur "talon-cheville" dans un référentiel terrestre noté $\boldsymbol{r}_W = (0, 0, r)^T$ et les viteses angulaires (données par le gyromètre). Cette procédure est déccrite en [2] et illustrée en Figure 6.



(**a**) (**b**) (**c**)

Figure 6: Les trois moments de rotations pendant la phase d'appuie : (**a**) rotation du talon, (**b**) rotation de la cheville, (**c**) rotation de l'avant-pied.

Cette procédure requière encore une fois de détecter quand la foulée intervient dans les enregistrements. Pour cette tâche, Sysnav a dans un premier

temps développé une approche similaire à celles présentées en Section I.4.1 basées sur la détection du balancement du pied (Figure 4) et une combinaison de critères sur les données inertiielles pour déterminer l'instant de contact entre le pied et le sol (accélérations proche de 1 g et des vitesses angulaires inférieures à un certain seuil. À titre d'exemple, la Figure 7 représente les données inertielles enregistrées par l'ActiMyo pendant une série de 5 foulées. Dans le référentiel

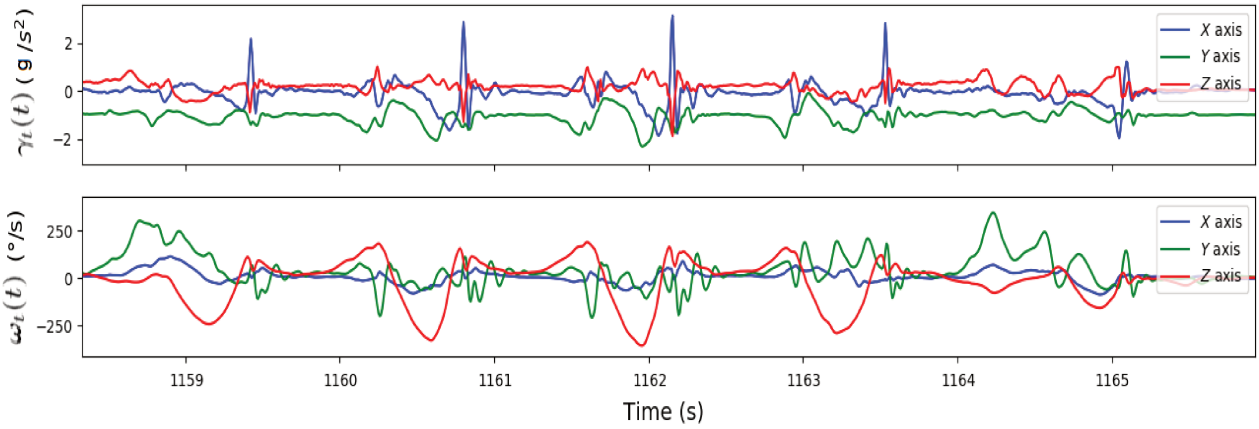


Figure 7: Données inertielles dans le référentiel du dispositif pendant la marche.

du système "body", le balancement duu pied est visible selon l'axe Z avec des valeurs négatives des vitesses angulaires et des accélérations sur Y proches de 1 g après un pic qui correspond au contact du peid avec le sol. Cette approche indique grossièrement où nous pouvons estimer la vitesse de la cheville avec le modèle bras de levier, ce qui a été breveté et décrit en [82] par Dorveaux E., Jouy A., Grelet M., Vissiere D. and Hillion M. La difficulté comparée à la méthode ZUPT est de trouver le moment optimal où la vitesse est égale au produit vectoriel entre $\boldsymbol{r}_W$ et $\boldsymbol{\omega}_W$: $\boldsymbol{v}_W = \boldsymbol{\omega}_W \times \boldsymbol{r}_W$. Soit une initialisation au temps t où la matrice de rotation entre le référentiel body du système et le référentiel terrestre $\boldsymbol{R}_W^t$ est connue, on peut exprimer les données inertielles au temps $t + \Delta t$ dans le référentiel terrestre grâce à l'intégration des vitesses angulaires. Puis lorsque l'accélération spécifique mesurée $\gamma_W(t + \Delta t) - \boldsymbol{g}_W$ est proche de l'accélération spécifique attendue qui est définie par la dérivée de $\boldsymbol{\omega}_W(t + \Delta t) \times \boldsymbol{r}_W$, l'instant $t + \Delta t$ est considéré comme le moment optimal et nous pouvons mettre à jour la vitesse dans le filtre étendu de Kalman avec le modèle de navigation à l'estime (Figure 8) en suivant le même principe que celui présenté en Figure 5.

Cette méthode montre des bons résultats pour des phases de marche classique mais montre des limitations pour des pas atypiques comme lors des escaliers ou piétinements. Afin de proposer une illustrations de ces problèmes, nous avons demandé à un utilisateur de montrer des escaliers et passer par une couloiir étroit où il doit réaliser des petites foulées de piétinement avant de revenir au point de départ. La trajectoire reconstruite par l'algorithme de Sysnav est représentée en Figure 9. Cela montre bien que lors des foulées atypiques dans le couloir notamment, aucun pas n'a été détecté car la trajectoire reste inchangée pendant plusieurs dizaines de secondes. Au final, l'erreur de reconstruction de trajectoire est proche de 2 mètres.

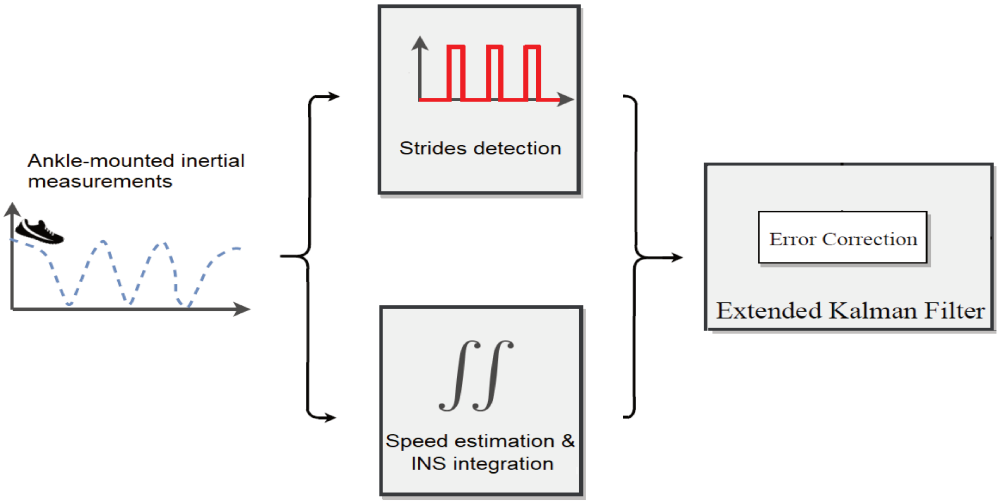


Figure 8: Détection des instants de vitesse bras de levier combinée avec la navigation à l'estime dans un filtre de Kalman étendu.

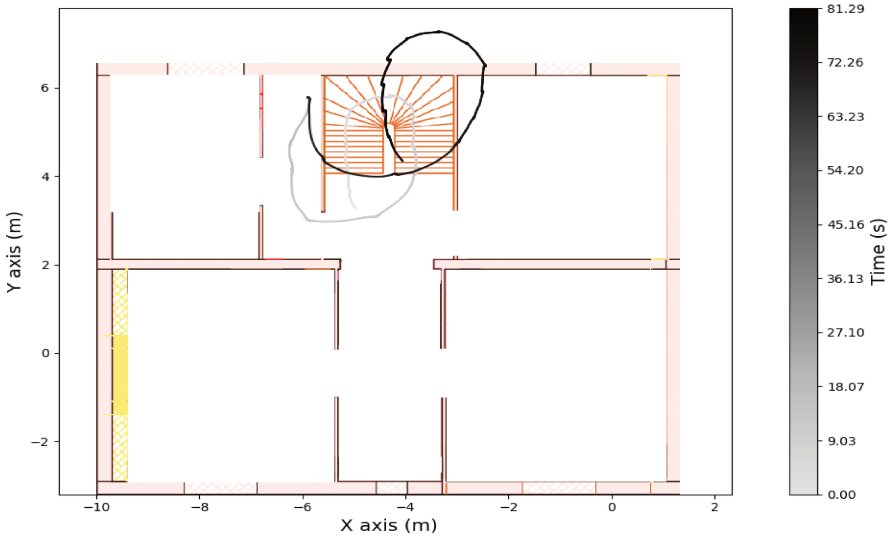


Figure 9: Trajectoire calculée avec l'algorithme Sysnav : une détection des foulées basée sur des valeurs seuil inertielles.

Cette performante n'est pas satisfaisante pour les cas d'application industrielle présentée en Section I.1.1. Par exemple les pompiers font de nombreux pas atypique pendant leurs missions (escaliers etc.) et il est essentiel de ne pas accumuler plusieurs mètres d'erreur en moins de une minute. Même conclusion pour les application médicales et notamment les variables cliniques présentées en Section I.3.3 qui sont basées sur la distribution de la trajectoire des foulées. À domicile, les enfants réalisent de nombreuses foulées atypiques lors des activités du quotidien. Les manquer biaise les résultats et empêche une étude clinique fiable. De plus, en pratique les capteurs ne sont pas toujours portés à la cheville, et il est important de ne pas détecter de foulées sur ces moments. Ne serait ce que lorsque le dispositif est installé ou retirer de la cheville pour le reposer sur sa station, ou lorsqu'il est porté à la main, mis dans la poche ou un sac à dos etc. Ces situations sont ordinaires et provoquent de nombreuses erreurs.

4.5 Reconnaissance d'activité (AR)

Ce travail vise à être appliqué pour calculer des statistiques pertinentes durant des études cliniques, reliées aux différents tests déjà en place et présentés en Section I.3.2 (test des 4 marches, 10 mètres de course, 6 minutes de marche). Durant les dix dernières années, le reconnaissance d'activité a constitué un sujet de recherche important dans un contexte médical. Alors que les solutions à base d'analyse d'image sont très intrusives [62], l'émergence des accéléromètres et gyromètres dans les objets connectés du quotidien (smartphones) permet d'effectuer des analyse de données inertielles pour la reconnaissance d'activité. La plupart des articles de la littérature utilise des algorithmes à fenêtre glissante combinée avec des chaînes de Markov [75] ou de l'apprentissage automatique [53, 78, 85]. Cependant, ces méthodes sont généralement non robustes aix phases de transition. En effet, elles montrent souvent des erreurs au début ou à la fin des activités, quand la fenêtre superpose la fin d'une activité et le début de la nouvelle. Ces algorithmes montrent aussi des mauvaises performances lorsque la taille de la fenêtre est trop petite et ne contient pas assez d'information. De plus, ils ne sont pas adaptés à la détection des foulées car dans le quotidien on peut changer rapidement et régulièrement de types de foulée (escaliers avec des plateforme par exemple). Le travail réalisé dans cette thèse utilise la trajectoire reconstruite des foulées détectées, information précieuse, afin de reconnaître l'activité. Comme les utilisateurs peuvent avoir des âges et physiques différents, nous n'appliquons pas des techniques basées sur des valeurs seuil de la longueur ou vitesse de la foulée. Nous préférons construire un algorithme d'apprentissage automatique.

D'autres travaux traitant des données inertielles sont appliqués dans un contexte médical. Par exemple dans [64], un dispositif porté à la chaussure est utilisé pour analyser la démarche et détecter automatiquement la maladie de Parkinson en utilisant des algorithmes d'apprentissage automatique. Cependant leur étude présente plusieurs limitations : elle est basée sur un petit jeu de données qui a été construit dans un environnement contrôlé. Les utilisateurs marchent 10 mètres quatre fois à un rythme confortable et en ligne droite sans obstacle. La segmentation est donnée par une approche valeur seuil sur le gyromètre et des techniques classiques de traitement du signal (moyenne, variance, maximum, minimum). Cette approche ne serait robuste pour la détection de foulées dans des enregistrement à domicile, ils ont d'ailleurs admis que le modèle avait des difficultés à généraliser sur d'autres jeux de données.

5 Variables cliniques pour la maladie de Parkinson

Parmi les troubles neurodégénératifs courants liés à l'âge, la maladiie de Parkinson est la deuxième plus présente. En France on compte plus de 200 000 personnes atteint de la maladie. Des traitements sont en cours de développement et plusieurs présentent des résultats encourageants mais l'analyse de leur efficacité reste un véritable challenge, dûe notamment à la complexité des symptômes. En Section I.5.1 nous présentons rapidement les symptômes et le traitement Levodopa. En Section hyperlinkAutomatic detection of Parkinson's eventsfrI.5.2 nous présentons les algorithmes à la pointe de l'état de l'art pour la détection des évènements parkinsoniens. Puis nous concentrons notre étude en Section I.5.3 sur les algorithmes utilisant l'ActiMyo développés par Sysnav pour la détection des tremblements et crises de dyskinésie, ainsi que leurs limitations.

5.1 Présentation de la maladie

Actuellement, le traitement le plus suivi est basé sur la substance Levodopa qui est connue pour atténuer les symptômes [39], comme les tremblements bradykinésie et le "gel de la marche". Cependant ce médicament a un fort inconvénient : il provoque des crises de dyskinésie (see Figure 10). Ces effets secondaires appraissent de manière impromtue et ne suivent pas de modèle très clair.

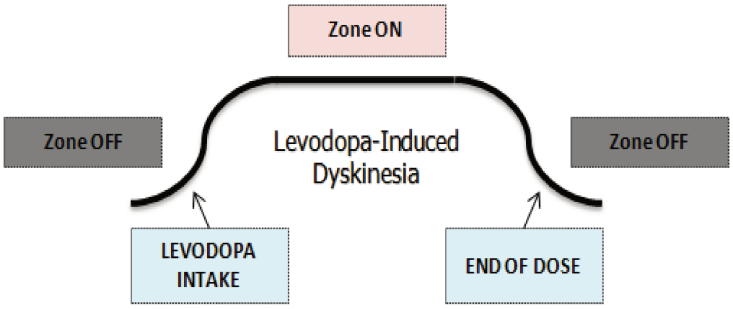


Figure 10: Effets secondaires du traitement Levodopa.

5.2 Détection automatique d'évènements parkinsoniens

Beaucoup d'articles ont été rédigés sur la détection des crises de dyskinésie et des tremblements. Plusieurs techniques ou signaux d'entrée ont été testées, comme les signaux d'électromyographie ou signaux d'électroencéphalogramme, ainsi que l'utilisation de la vision ou même de capteurs audio. Cependant, la plupart des articles se limitaient soit à un environnement contrôlé, soit à l'utilisation des dispositifs uniquement par des personnes atteintes de la maladie [56], au gel de la marche [55, 52], à la détection des tremblements [4, 49] et bradykinésie [50], l'activité physique duu patient [71], l'apparition des crises de dyskinésie lors d'activités bien spécifiques [44], ou à la corrélation avec le score UPDRS donné par les médecins [63]. D'autres articles essaient de caractériser les états ON-OFF en détectant les crises de dyskinésie selon des valeurs seuil du spectre des signaux [45]. À notre connaissance, aucun algorithme ne parvient à généraliser en dehors d'un milieu contrôlé. Il semble aussi qu'aucun article ne prétend obtenir une détection fiable et précise des crises de dyskinésie.

5.3 Détection automatique des tremblements et crises de dyskinésie avec l'ActiMyo positionné à la cheville

Le système ActiMyo a été porté à la cheville par 14 patients atteint de la maladie de Parkinson pendant un preuve de concept en 2014 et durant une étude clinique de 2017, toutes les deux traitant avec la Levodopa. Ces enregistrement ont été utilisés pour développer un algorithme de détection automatique de deux types d'évènements : les tremblements et crises de dyskinésie.

Dans un premier temps l'algorithme décrit en Section I.4.4 est utilisé pour détecter quand une foulée intervient dans les enregistrement. Cela permet d'exclure tous les intervalles de temps où le patient est en train de marcher. Les intervalles sans activité sont aussi exclus. Les tremblements ont été caractérisés par des mouvements répétés en fréquence : un pic empirique dans des valeurs de fréquences doit être observé dans un axe des accéléromètres et gyromètres. Tandis que les crises de dyskinésie sont détectées à la cheville en procédant par

élimination en ne gardant uniquement les phases avec des mouvements suffisament importants mais n'étant pas associés à des déplacements (marche, course etc.).

Cette approche a été testée pendant des enregistrements effectués en hôpital, où les patients doivent réaliser des actions particulières pour la mesure du score UPDRS. Les algorithmes ont montré des résultats encourageants dans ce genre d'environnement contrôlé mais ne parviennent pas à généraliser suur des enregistrements à domicile face à la grande variété de mouvements. En effet, une comparaison des détections peut être faite à partir des enregistrements de volontaires, non atteints par la maladie, qui ont accepté de porté le dispositif. Ils ont souvent le même âge, les conjoint(e)s par exemple et servent de patient "contrôle" pour les étude. Nous avons pu observer beaucoup de faux positifs dans la détection des évènements parkinsoniens.

6 Contributions générales de la thèse

Dans la section précédente, nous avons décrit les algorithmes développés par Sysnav pour la reconstruction de trajectoire et la détection des évènements parkinsoniens (respectivement en Section I.4.4 et Section I.5.3). Nous avons montré plusieurs limitations que nous visons à résoudre dans le travail de cette thèse avec des algorithmes d'apprentissage statistique supervisé. En Section I.6.1 nous introduisons les concepts de l'apprentissage supervisé et comment l'utiliser aux données enregistrées par l'ActiMyo pour la tâche de détection des foulées (Section I.6.2 and Section I.6.3), activity recognition in Section I.6.4 et la détection des évènements parkinsoniens en Section I.6.5. Ces travaux sont basés sur des algorithmes innovants dont nous présentons les principales idées dans ce qui suit.

6.1 Apprentissage statistique supervisé pour la détection de foulées et des évènements parkinsoniens

De part la complexité de notre cadre d'application, la problématique de notre étude est difficile à décrire avec des modèles déterministes, et donc nous avons adopté des stratégies basée sur le machine learning. Cette première section vise à introduire les principaux concepts de l'apprentissage supervisé pour les séries temporelles. Ces méthodes sont appliquées aux données inertielles de l'ActiMyo : accélérations et vitesses angulaires selon trois axes. Traditionnellement pour la détection d'évènements dans les séries temporelles, l'approche par fenêtre glissante est utilisée pour sélectionner des intervalles qui sont ensuite classifiés par la fonction de prédiction, auparavant calculées par apprentissage statistique supervisé. Nous appliquons cette technique pour la détection des crises de dyskinésie mais développons une approche innovante pour la détection des foulées basée sur une extraction d'intervalles candidats. Le principe est de sélectionner des intervalles pertinents qui pourraient correspondre à des foulées et sélectionner ceux qui sont réellement des foulées via le machine learning. En effet, de nombreux intervalles sont sélectionnés à tort quand l'utilisateur bouge sa cheville mais sans être en train de marcher ou lorsque le dispositif n'est pas porté à la cheville (dans la poche ou dans un sac à dos). L'extraction d'intevalles pertinents permet de réduire le nombre de classifications par la fonction de prédiction et réduit la complexité statistique.

Considérons un vecteur aléatoire $(\mathbf{X}, Y)$ à valeurs dans $\mathbb{R}^p \times \mathcal{Y}$ dont la distribution de probabilité $P_{\boldsymbol{X},Y}$ est inconnue. Les algorithmes d'apprentissage supervisé vise à estimer le lien entre les covariables $\boldsymbol{X} = (X_1, \ldots, X_u, \ldots, X_p)$ (qui correspondent aux données inertielles des intervalles dans notre cas) et une variable à prédire Y (évènements parkinsoniens par exemple). L'estimation de la fonction de prédiction f définie sur $\mathbb{R}^p$ et à valeurs dans $\mathcal{Y}$ repoose sur un jeu de données $\mathcal{D}_n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_i, Y_i), \ldots, (\boldsymbol{X}_n, Y_n)\}$ de n couples indépendants et identiquement distribués suivant la distribution $P_{\boldsymbol{X},Y}$. Un estimateur $\hat{f}$ de f permet de prédire une nouvelle valeur Y_{n+1} à partir d'une nouvelle observation $\boldsymbol{X}_{n+1}$.

En pratique $\boldsymbol{X}_i$ correspond aux données du gyromètre et accéléromètre de l'intervalle i (fourni par la fenêtre glissante ou l'extraction d'un intervalle candidat). Les algorithmes d'apprentissage supervisés nécessitent que pour tout i, $\mathbf{X}_i$ soit à valeurs dans $\mathbb{R}^p$ avec p un entier fixé. Une approche par fenêtre glissante permet d'extraire des intervalles à taille fixe mais l'approche innovante d'extraction d'intervalles candidats n'assure pas d'obtenir une taille constante. Pour répondre à cette problématique, on peut calculer des variables depuis les données inertielles $\boldsymbol{X}_i$ (comme la moyenne, la variance etc.). Cette procédure est appelée features engineering. Au final, cela permet de calculer $\boldsymbol{Z}_i$ à valeurs dans $\mathbb{R}^d$ quel que soit la taille de l'intervalle extrait. Nous observons alors un couple $(\boldsymbol{Z}, Y)$ à valeurs dans $\mathbb{R}^d \times \mathcal{Y}$ où d correspond au nombre de features calculées. Les algorithmes d'apprentissage statistique supervisé peuvent alors être utilisés quelle que soit la méthode d'extraction d'intervalles.

6.2 Extraction d'intervalles candidats basée sur le contact au sol

Le dispositif ActiMyo doit être positionné à la cheville ou au poignet, comme illustré en Figure 11. Dans cet emplacement par défaut, les capteurs enregistrent les données inertielles dans le référentiel défini par un axe Z aligné avec la jambe et l'axe X aligné avec le pied. Cependant nous avons observé que le système pouvait être placé à l'envers et pouvait tourner autour de la cheville au cours de l'enregistrement. Typiquement, en Figure 7, l'axe Y est aligné avec la jambe (accélérations proches de -g quand le pied est au sol). C'est pourquoi à ce stade

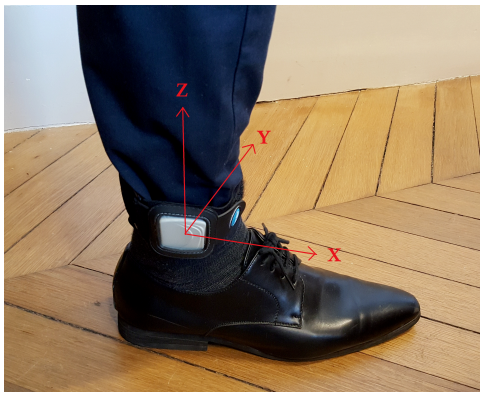


Figure 11: Placement du dispositif par défaut.

nous n'avons aucune information de la direction des différents axes des données inertielles. Pour répondre à ce problème en Chapitre III, les normes des signaux sont utilisée. Nous avons observé que les foulées produisent un pic dans la norme des accélérations quand le pied entre en contact avec le sol. Puis le début et la fin de l'intervalle sont définis par des minima locaux de la norme du gyromètre autour des pics d'accélération. En effet, quand le pied entre en contact avec le sol, les vitesses angulaires de la cheville sont plus faibles que quand le pied est en

phase de balancement. Cependant cette approche extrait également des intervalles qui ne correspondent pas à des foulées quand par exemple l'utilisateur est en train de faire du vélo. The but est alors de sélectionner parmi les intervalles extraits ceux qui sont réellement des foulées. Nous appliquons une méthode d'apprentissage statistique supervisé pour répondre à ce problème grâce à une base de donnéees que nous avons préalablement construites.

La procédure de features engineering repose sur le calcul d'une rotation appliquée sur les données inertielles afin de travailler dans un référentiel commun entre les enregistrements. Cette étape clef de notre algorithme est basée sur un ensemble de modèles géométriques à trois dimensions des vitesses angulaires calculés depuis la base de données. Ensuite le balancement du pied vers l'avant, qui intervient juste avant le contact avec le sol, est étudié. Il fournit des précieuses informations pour la fonction de prédiction calculé à partir de l'algorithme Gradient Boosting Trees (Section II.2.2).

Ce premier algorithme de détection des foulées montre des résultats encourageants et des améliorations comparé à la méthode existante développée par Sysnav et introduite en Sectioin I.4.4 tout particulièrement pour les foulées atypiques. Cependant, il montre également des limitations lors de pas chassés rapides ou des descentes rapides d'escaliers. De plus, cet algorithme est coûteux en temps de calcul car le nombre de pics d'accélération peuvent être créés par de nombreux mouvements qui ne correspondent pas à des foulées. En conséquence le calcul des features et l'appel de la fonction de prédiction sont effectués de nombreuses fois en parcourant les enregistrement.

6.3 Extraction d'intervalles candidats de fouléees basée sur une pseudo-vitesse

En Chapitre IV, l'idée principale repose sur le fait que dans un référentiel inertiel, l'intégration des accélérations pendant une période Δt est égale à la différence de la vitesse de la cheville (quelques mètres par seconde pour un piéton) ce qui est petit devant l'intégration de la gravité. À partir des intégrations des vitesses angulaires pendant t et $t + \Delta t$, nous pouvons calculer les matrices de rotation $\boldsymbol{R}_t^{t+dt}, \boldsymbol{R}_t^{t+2dt}, \ldots, \boldsymbol{R}_t^{t+\Delta t}$. Cela permet par composition des matrices de rotation à projeter les accélération dans le référentiel body du système du temps t. En effet par définition on a $\boldsymbol{R}_t^{t+2dt} = \boldsymbol{R}_t^{t+dt}\boldsymbol{R}_{t+dt}^{t+2dt}$. Puis l'intégration de $\boldsymbol{\gamma}_t$ entre t et $t + \Delta t$ est donné par :

$$\begin{aligned}
\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{\gamma}_t(u)du &= \frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{a}_t(u)du + \boldsymbol{g}_t \\
&= \frac{\boldsymbol{v}_t(t+\Delta t) - \boldsymbol{v}_t(t)}{\Delta t} + \boldsymbol{g}_t \\
&\simeq \boldsymbol{g}_t,
\end{aligned}$$

où $\boldsymbol{g}_t$ est une constante. En supprimant la gravité des données accéléromtètre projetées dans le référentiel terrestre qui est aligné avec la gravité identifiée, nous pouvons calculer une pseudo-vitesse $\hat{\boldsymbol{v}}_W$ qui est une features importante pour détecter le début et la fin des foulées. Cette procédure qui vise à obtenir les données dans un référentiel terrestre, est appelée filtre d'attitude. La quantité de vitesse calculée est appelée pseudo-vitesse $\hat{\boldsymbol{v}}_W$ car ce n'est pas la quantité utilisée dans la reconstruction de trajectoire construite par le filtre de Kalman étendu qui est plus précise. Par contre son calcul est très rapide.

Cet algorithme vise à aligner les capteurs dans un référentiel commun (référentiel terrestre) afin de tirer profit des trois dimension de l'accéléromètre

et gyromètre, principe similaire à celui présenté auparavant en Section I.6.2. Cependant, la pseudo-vitesse calculée est plus robuste dans les situations critiques et fournit, par intégration, une pseudo-trajectoire qui apparaît comme une variable clef pour la décision de la fonction Gradient Boosting Trees (Section II.2.2). En effet, cette approche extrait des intervalles qui ne correspondent pas à des foulées quand par exemple l'utilisateur tient le dispositif dans sa main le temps de l'installation. La sélection des foulées réelles parmi les intervalles extraits est fournie par la fonction de prédicion.

6.4 Reconnaissance d'activité à partir de la trajectoire reconstruite

Sysnav a développé un algorithme introduit en Section I.4.4 qui permet de calculer la trajectoire pour chaque foulée détectée. Cet algorithme nécessite une estimation précise des débuts et fins des foulées avant d'appliquer l'estimation de la vitesse du système par le modèle de bras de levier. L'estimation de la vitesse est intégrée dans un filtre de Kalman étendu avec l'intégration des accélérations, fournissant la trajectoire reconstruite.

Pendant les études cliniques, la reconnaissance d'activité est une précieuse information pour évaluer la santé des patients souffrant de troubles du mouvement. Nous concentrons notre travail sur trois activités particulières en relation avec les variables cliniques usuelles pour la myopathie : les escaliers, la marche, et la course. Cependant, définir la frontière entre la course et la marche rapide au travers de la trajecctoire est un véritable challenge. En effet, la différence d'âge entre les patients en étude clinique peut être assez grande, et leur démarche très différente. De plus, détecter les escaliers pour des patiients souffrant de myopathie s'avère difficile car ils ont tendance à monter les marches une par une ce qui aboutit à une faible différence d'altitude notamment. C'est pourquoi en Chapitre V nous avons mise en place un algorithme d'apprentissage supervisé pour construire une fonction de classification qui reconnaît l'activité des foulées via leur trajectoire avec un calcul des features basé l'analyse des données fonctionnelles.

6.5 Réseau de neurones pour la détection des évènements parkinsoniens

Le Chapitre VI présente une approche de deep-learning innovante et générique pour des problématiques issues de données inertielles. Au travers de ce travail, nous concentrons notre attention sur la détection des tremblements et des crises de dyskinésie, en parallèle de la reconnaissance d'activité. Notre travail est basé sur un réseau multi-channel, utilisant en particulier des réseaux de neurones à convolution et un channel d'analyse topologique des données (TDA) pour le traitement de séries temporelles multivariées. L'analyse topologique des données est un domaine récent qui a émergé de divers travaux de géométrie computationnelle notamment, visant à fournir des méthodes mathématiques, statistiques et algorithmiques bien définies pour exploiter les structures géométriques et topologiques dans les données.

Part I

Introduction (English)

This first chapter aims to contextualize the main problems of the thesis and to give an overview of the existing algorithms that have been developed to overtake them. In Section I.1 we describe the Inertial Measurement Units (IMUs) and their use for trajectory reconstruction and healthcare application. In particular we present the ActiMyo device developed by Sysnav in Section I.2, designed to be worn at the wrist and at the ankle. In Section I.5 and Section I.3 we describe the variables computed by Sysnav aiming to characterize the evolution of the health condition of patients during clinical studies for two diseases: Parkinson and Duchenne Muscular Dystrophy (DMD). The DMD clinical outcomes are based on the trajectory reconstruction of the device worn at the ankle. In Section I.4 we introduce the notions of trajectory reconstruction for a pedestrian with shoe-mounted IMU that have been adapted to the ActiMyo device. Whether for DMD or Parkinson's clinical variables, Sysnav developed algorithms with several limitations that we will try to overtake in this thesis. In Section I.6 we briefly describe the content of our work divided into three machine learning applications: the stride detection, the activity recognition and the Parkinson's event detection.

1 General context and main issues of the thesis

In this section we present two types of application using Inertial Measurements Units (IMUs). In Section I.1.1 we describe the advantage of using IMUs for trajectory reconstruction which overtakes several limitations of Global Navigation Satellite Systems (GNNS) in indoor positioning as an example. These devices are also increasingly used in a medical context to measure the living conditions of patients at home. Indeed their small size makes them particularly easy to integrate into daily life situations.

1.1 Inertial Measurement Unit for Trajectory Reconstruction

The emergence of Global Navigation Satellite System (GNSS) receivers in the 2000s changed the perception of navigation. While they are commonly used in outdoor environments, in many situations they fail to produce accurate localization due to poor reception (e.g., tunnels, indoor parking, forests, inside buildings, etc.). However, reliable localization may be vital in situations where a continuous position estimate is needed to ensure safety, for instance for isolated workers, firefighters or military applications.

In this context, Inertial Measurement Units (IMUs) can be utilized as part of an Inertial Navigation System (INS). The main advantage of these systems is that they do not rely on external references once the initialization point has been given to the INS, which makes them highly robust to external disturbances or sabotage, especially compared to architecture based localization systems, such as the GNSS. For this reason, INS are extensively used in products relying critically on their position estimate such that aircraft, boat, submarines, missile, spaceship, satellites. Being critical for these military and industrial applications, INS have been extensively developed after the beginning of the second half of the 20th century. The trajectory is estimated by dead reckoning technique: an IMU signal (acceleration and rotation speed) is integrated to predict one's current position by using a previously determined position. It is known that for this kind of applications, using classical navigation method requires to use high precision sensors that are too bulky and too expensive to be carried by a pedestrian. To be

able to estimate a trajectory with light and low cost inertial sensors such as those used in mobile phones, for example, different methods hast to be implemented.

1.2 Inertial Measurement Unit for Healthcare Application

Other works with inertial wearable devices are applied in a medical context. The use of INS in medical engineering has been rapidly spreading for human motion tracking and analysis. IMUs are noninvasive and relatively easy to integrate into a wearable strap, making them appropriate sensors to utilize in a diverse variety of portable technologies aimed at home healthcare and wellness applications since they are capable of assessing and monitoring activities and subsequently enabling the evaluation of an individual's quality of life. They can be used for instance in activity monitors, fall detection and symptoms detection of patient suffering from pathologies associated with movement disorders. Although medicine promotes this kind of innovation, clinical trials that evaluate the effectiveness of new treatment take place in a highly regulated environment. Until 2019, no system has been approved by the European Medicine Agency (EMA) as representative of the physical conditions of the patient and can be used in clinical studies. This topic has thus become a major issue.

In this context Sysnav developed ActiMyo, a system to be worn at the ankle and the wrist, based on magneto-inertial sensors to compute relevant trajectory measures for patients suffering from movement disorders. In particular, for pathologies involving difficulties of walking, it is crucial to precisely analyse the trajectory of the ankle-mounted device.

2 ActiMyo: magneto-inertial sensors system

Actimyo is composed of one accelerometer and one gyrometer recording data (respectively γ and ω) in its own reference frame (called body reference frame) at 130 Hz with about 16 hours of battery autonomy. The accelerometer is measuring both the linear acceleration due to motion ($\boldsymbol{a}$) and the acceleration caus ed by gravity ($\boldsymbol{g}$): $\gamma = \boldsymbol{a} + \boldsymbol{g}$. The inertial sensors start recording data when the device is taken from its case. When put back, the recorded data are transferred to a cloud server we have access to.

(**a**) (**b**) (**b**)

Figure 12: ActiMyo connected to the docking station (**a**), worn at the ankle (**b**) and the wrist (**c**).

2.1 Strapdown IMU

The IMU used in this device is a strapdown kind described in [60]. In this kind of IMU, the accelerometer and gyrometer are rigidly attached on a solid platform, in contrast to a gyrostabilized platform where accelerometer orientation is stabilized in the inertial frame. This has one major implication: the attitude is not given by the gyroscope directly, but has to be integrated from the angular velocity. This strongly correlates the problems of attitude and position and one of the reasons for the difficulty of doing inertial navigation. However nowadays, Inertial Navigation Strapdown systems are of particular interest. The majority of IMUs are strapdown montage because they are easier to build mechanically, are smaller and lighter, the best example being the smartphones.

2.2 Inertial sensors calibration

The triaxial inertial sensors in ActiMyo are designed to output a signal proportional of the quantity of interest (angular velocity for gyrometer and acceleration for accelerometer). However, their models have to be calibrated individually before using them as measurements of acceleration or rotational velocity. Moreover, the combination of the three axis is never totally orthogonal because of the precision of the mechanical assembly. The calibration can also be influenced by environmental factors such as the temperature, this effects also have to be taken into account. Sysnav has developed an intern module that computes one unique calibration compensating errors (scale factor, biais, nonorthogonality etc.) for each manufactured system.

ActiMyo has been designed to be easy to use for patients. It requires little manipulation and can be connected directly to the docking station so that the data collected each day is automatically sent to a dedicated server for offline analysis.

3 Clinical outcomes for Duchenne Muscular Dystrophy

In this section, we present the effects on patients life, suffering from Duchenne muscular dystrophy (Section I.3.1) and the "official" variables calculated at hospital for measuring symptoms during clinical trials (see Section I.3.2). However, these hospital tests have several drawbacks that Sysnav wants to overtake with the ActiMyo device (see Section I.3.3).

3.1 Presentation of the disease

Duchenne muscular dystrophy (DMD) is a genetic disorder that results in a progressive loss of functional muscle fibers and weakness affecting mobility. The past several years have seen increased interest by biopharmaceutical companies in conducting research and development into novel treatment agents for DMD. There is hope that new therapies will slow disease progression and maintain quality of life. The clinical trials target DMD patients who are old enough to be assessed reliably and still have sufficient muscle fibers available for therapeutics to demonstrate a measurable benefit over the typical 1-year clinical trial duration.

3.2 Hospital tests

Over the last 50 years, tens of billions of dollars have been invested to bring new molecules from the laboratory for develop the most innovative drugs. But one aspect has never changed: how to judge the effectiveness of treatments, by variables called "endpoint". The existing measured variables are the time taken by the patient to climb 4 steps (4-stairs test), the time for running 10 meters (10 meters run test) or the distance traveled when the patient is asked to walk for 6 minutes around two cones in the hallway of a hospital (6 minutes walk test). The main problem is these clinical outcome measures can be biased by the motivation of the patient. The results can indeed be impacted by the fitness condition of the day without being correlated with patient health. In addition, human errors may appear in the measurement. For instance, the 4-stairs test can be performed in less than two seconds and the measures taken by two different doctors can vary consistently. Finally it is tiring and sometimes complicated for patients to regularly go to the hospital.

3.3 Home variables with ActiMyo

There is a gap between the investment in the therapies compared to effort for designing new endpoints to judge the efficacy of treatments in clinical trials. The goal of ActiMyo is to compute new outcome measures correlated with the previous mentioned tests in uncontrolled environments: at home, at school, playing outside or resting inside etc. It is important to compute outcomes that describe the actual daily activity during months compared to punctual measures performed at hospital. The proposed variables are aiming at the motor function domain and include the stride length, stride velocity and distance walked [32, 46]. In order to connect these results to 4-stairs test or 10 meters run test, Activity Recognition (AR) must be applied.

For stride length estimation with IMU, Pedestrian Dead Reckoning (PDR) has been shown to yield positioning accuracy adequate for many end applications. In the following we will introduce the basics of PDR algorithm with shoe-mounted IMU and the innovative Sysnav algorithm that adapted the technique for an ankle-mounted device.

4 Introduction to PDR with shoe-mounted IMU

The problem of inertial navigation can be summarized in a very simple way: to estimate the evolution of the position, orientation and speed of an object it is attached to, with respect to another reference frame. This approach mainly relies on integrating the linear acceleration and the angular velocity data to yield position updates over time. Unlike infrastructure-dependent localization systems such as map matching, Wi-Fi [84], radio frequency identification [80], or ultra-wideband [41], body-mounted IMUs are lightweight and can be rapidly and easily deployed (see Figure 13).

However, all strapdown IMUs are subject to drift, and the integrations rapidly accumulate large errors. To overcome this issue, the zero velocity update technique (ZUPT) [43, 5, 26, 74] is traditionally used. The goal of this method is to detect when the foot is flat on the ground and stationary relative to the surface. In Section I.4.1 and Section I.4.2 we introduce two kinds of detection: respectively threshold-based and with machine learning. Knowing these time moments, integration of the inertial data is required only during the intervals between footfalls

Figure 13: PERSY (Predestrian Reference SYstem) developed by IFSTTAR.

(see Figure 14) instead of along an entire trajectory. The procedure is described in Section I.4.3. However, this technique is not directly usable to the ActiMyo

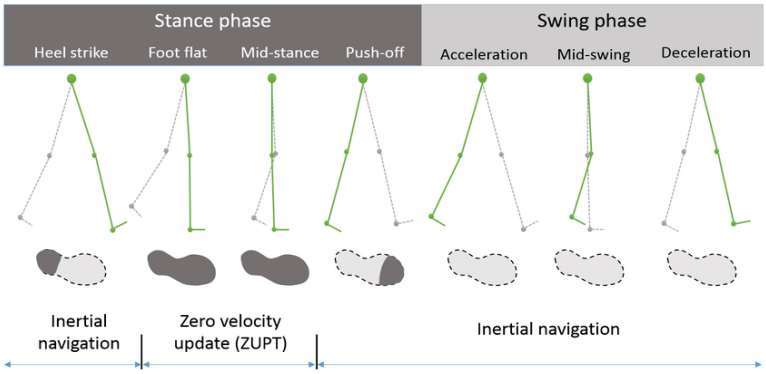


Figure 14: Walking gait cycle and measurement update cycle.

device as it is worn on the ankle and not on the foot. Sysnav develop a new inspired ZUPT approach presented in Section I.4.4 that presented several limitations. Finally in Section I.4.5 we described how to take advantage of the ankle reconstructed trajectory in a medical activity recognition application.

4.1 Threshold-based Zero Velocity Update Detection

Several studies in the literature have proposed to tune thresholds on the inertial data (linear velocity close to one g and small values of the angular velocity) [58, 42, 1]. A known limitation of these fixed-thresholds-based detectors is they fail to perform reliably across a variety of gait motions. These methods show good results for classical gait, but fail for atypical strides such as stairs and small steps. In [69], several inertial devices worn at the ankle or wrist have been tested for estimating steps and travelled distance during walking, stairs, and simulated household activities. Every tracker showed a decreasing accuracy with slower walking speed, which resulted in a significant under-counting of steps. Poor performance in travelled distance estimation was also evident during walking at low speeds and climbing up/down stairs. Recent approaches aim to improve detection by implementing adaptive techniques that are dependent on velocity [51, 77] or gait frequency [83]. However, modeling zero-velocity detection during motions such as stair climbing and crawling, while maintaining accurate detection during walking and running, is fundamentally challenging [40].

4.2 Machine Learning for Zero Velocity Update Detection

In order to overtake the limitations of threshold-based ZUPT detection, recent works use machine learning in stride detector with sliding windows approach [76, 6]. These methods consist of building a prediction function that outputs binary zero velocity classifications for every sample of the recording. The main drawback of this approach is that classifiers present a good detection rate but with many false positives corresponding to instants when the device is in motion and detected as zero velocity. This can induce large errors in the trajectory reconstruction. In order to maintain good performances, the algorithms in the literature adjust the classifier outputs, for example by removing the detected zero-velocity samples with insufficient confidence (under a tuned threshold). These kinds of compensation based on hand-tuned thresholds algorithms show good results when it is known that the pedestrian is walking, but are not robust enough for the complexity of daily human gait motion and in many real-life situations. Indeed, several foot movements in sitting position and bicycling for example are wrongly detected as strides. The review in [69] shows that every tracker tested in the study as a step detector presented many false positives during basic home activities such as writing, reading, and playing cards.

4.3 Trajectory Reconstruction from Zero Velocity Update Detection

In order to compute the trajectory, the strategy consisting of the integration of the linear acceleration and angular velocity data from the unit rapidly accumulates large errors due to IMU drifts. The Zero Velocity Update technique limits the errors by computing the integration only during detected strides, assuming that zero velocity is observed at the beginning and the end. The Zero Velocity Update Detection is fused in an extended Kalman filter [57] with a dead-reckoning motion model which is the process of calculating one's current position from the INS integration by using a previously determined position (see Figure 15). This allows to significantly reduce error growth over time. This filter may for example

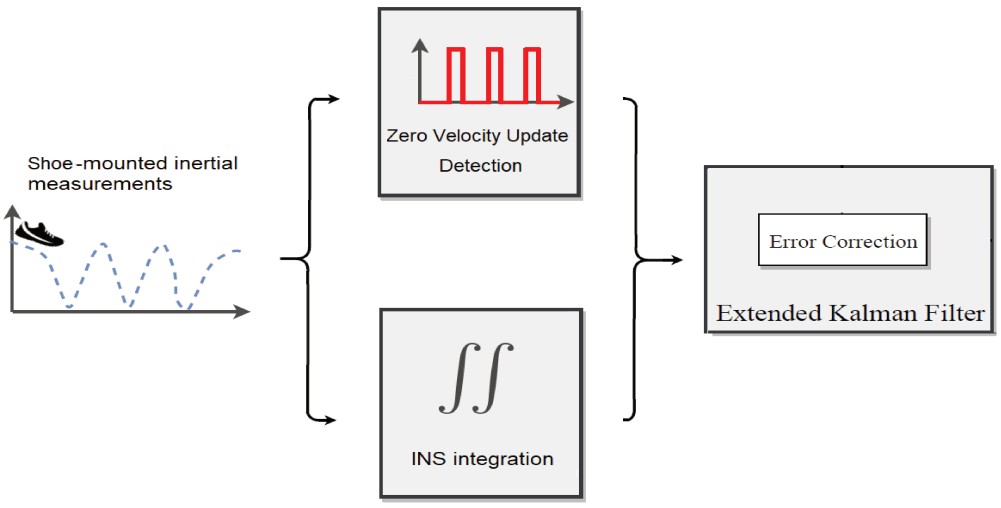


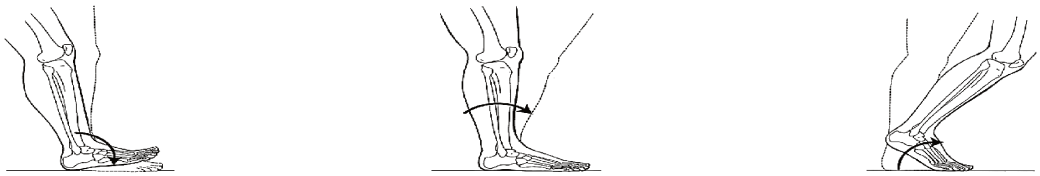
Figure 15: Zero Velocity Update Detection combined with dead reckoning in an extended Kalman filter.

include a state with 6 degrees of freedom for speed and attitude. Other states can be added such as position, sensor bias, etc. The filter also gives a measure of the confidence of estimated states.

The ZUPT method improves the quality of the motion estimation but leads to additional problems due to the device placement on the shoe. This makes it very sensitive to shocks and uncomfortable because it has to be incorporated to the shoe. For instance in clinical trials for Duchenne muscular dystrophy, it is difficult for children subjected to mockery at school to wear a system visible to all. Finally, during running phases, the immobile duration of the foot is reduced or even nonexistent which prevents the correction of the states in the Kalman filter. On the contrary, the ActiMyo system worn at the ankle can be easily hidden under a trouser but it leads to one major consequence: zero velocity is never observed in the inertial data when the foot is on the ground.

4.4 Sysnav Inovation: PDR for ankle-mounted IMU

As zero velocity is not observed during the stance phase (see Figure 14) Sysnav adopted an ZUPT-inspired method by developing a model based on lever arm to estimate the ankle speed at the beginning and the end of a stride and compute the integration in between. When the foot is flat on the floor, we assume the ankle is in rotation around the heel. Thus, the ankle speed is estimated by the cross-product between the vector "heel-ankle" in World frame noted $\boldsymbol{r}_W = (0, 0, r)^T$ and the angular velocity (given by the gyrometer). This process is described in [2] and illustrated in Figure 6.



(**a**) (**b**) (**c**)

Figure 6: The three foot rockers during stance phase: (**a**) heel rocker, (**b**) ankle rocker, (**c**) forefoot rocker.

This procedure still requires to detect when a stride occurs in the recording. For this task, Sysnav first developed an approach similar to those presented in Section I.4.1 based on the swing detection (see Figure 14) and a combination of criteria on the inertial data to determine the contact between the heel with the ground (acceleration close to one g and values of the angular velocity under a fixed threshold). As an example, the Figure 7 represents the inertial data recorded by ActiMyo during a series of 5 strides. In this body reference frame, the swing phase is visible in the Z axis with negative angular velocity values and the acceleration in Y axis is close to one $-g$ after the peak corresponding to the foot contact with the ground. This approach gives an rough indication where we can estimate the speed by the lever arm model which has been patented and described in [82] by Dorveaux E., Jouy A., Grelet M., Vissiere D. and Hillion M. The difficulty compared to ZUPT method is to find the optimum moment where the speed is equal to the cross product between $\boldsymbol{r}_W$ and $\boldsymbol{\omega}_W$: $\boldsymbol{v}_W = \boldsymbol{\omega}_W \times \boldsymbol{r}_W$. Given an initialization at time t when the rotation matrix between the body frame and world frame $\boldsymbol{R}_W^t$ is known, one can express the inertial data at time $t+\Delta t$ in World frame thanks to angular velocity integration. Then when the measured specific acceleration $\boldsymbol{\gamma}_W(t + \Delta t) - \boldsymbol{g}_W$ is close to the expected specific acceleration that is the derivative of $\boldsymbol{\omega}_W(t + \Delta t) \times \boldsymbol{r}_W$, the instant $t + \Delta t$ is considered as optimum and we update the speed in an extended Kalman filter with a dead-reckoning motion model (see Figure 8) following the same principle presented in Figure 15.

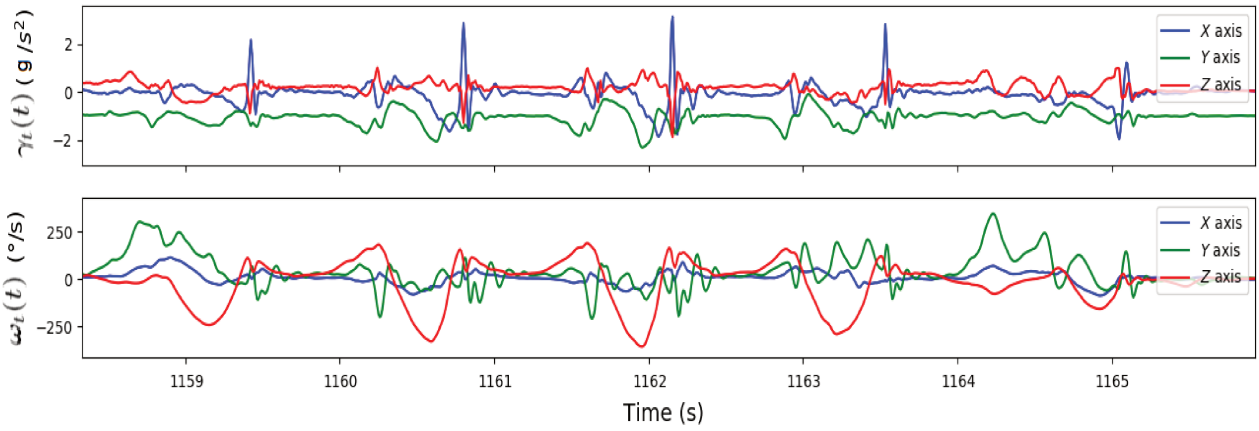


Figure 7: Inertial data in body frame during walking.

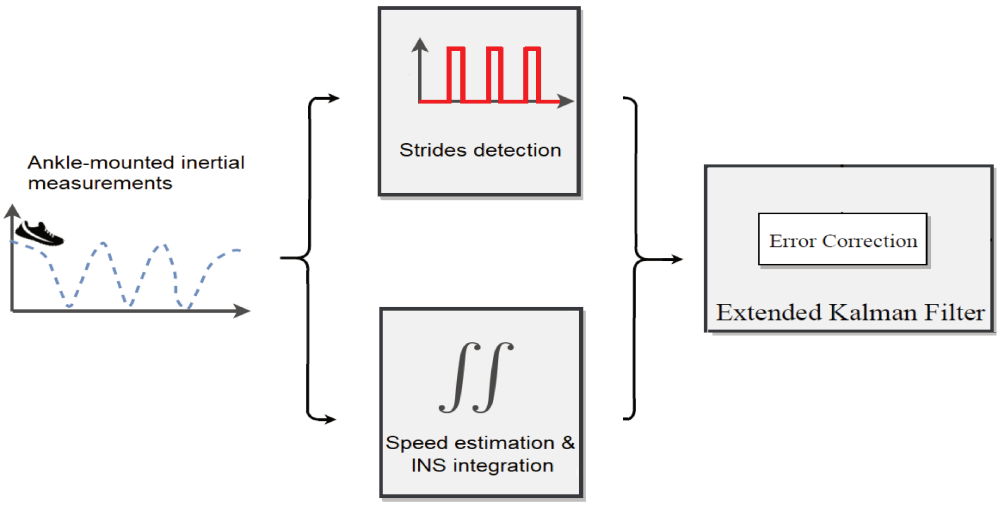


Figure 8: Lever arm Update Detection combined with dead reckoning in an extended Kalman filter.

This overall method shows good results for classical gait but it tends to fail for atypical strides such as stairs and small steps. In order to give an illustration of these limits, a device user was asked to climb stairs and get in a small corridor where he performs small steps before going back. The computed trajectory by the Sysnav algorithm is plotted in Figure 9. It illustrates that during the atypical strides in the corridor, no stride is detected as the trajectory stays on the same point for dozens of seconds. In the end, the error is about two meters.

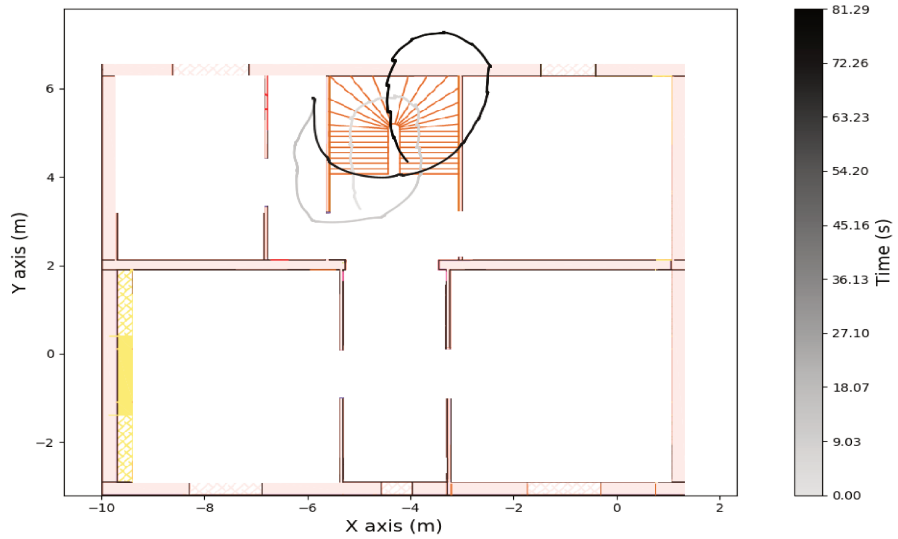


Figure 9: Computed trajectory with Sysnav algorithm: a strides detection based on inertial thresholds.

This performance is not satisfying for industrial applications presented in Section I.1.1. For instance firefighters perform a lot of atypical strides during missions (stairs etc.) and it is vital not to reach several meters error in less than one minute to have a reliable continuous position estimation. The same goes for the medical variables presented in Section I.3.3 that are based on strides trajectory distribution. At home, children perform many atypical strides during daily activities. Missing them skews the results and prevent a reliable clinical study to measure the evolution of patients' health. In addition, the sensors are often recording while the ActiMyo device is not worn at the ankle. For example, during the required time to install or uninstall the system from its docking station, when it is carried by hand, put in a pocket or a backpack. These situations are common and produce lot of errors.

4.5 Human Activity Recognition (HAR)

An important application of this work in the medical context is to compute relevant statistics during clinical trials related to the medical tests we mentioned in Section I.3.2 (walking test, stairs test, and running test). In the past decade, HAR has become an important field of research in the health-care context. While vision-based techniques work with intrusive equipment [62], with the emergence of accelerometers and gyrometers in connected objects in daily life (wearable sensors, smartphones), inertial data analysis for Activity Recognition (AR) appears as an important challenge for precision medicine. Most of the papers in the literature use HAR algorithms based on a fixed-size sliding window combined with hidden Markov model [75] or machine learning [53, 78, 85]. However, these methods are generally not very efficient at transition times. Indeed, they often show errors at the beginning or at the end of activities, when the window overlaps the end of one activity and the beginning of the next. Bad predictions also occur when the window length is too short to provide the best information for the recognition process. In addition, these methods are not adapted when we need to detect individual strides, while activities can change quickly in many daily situations (e.g., stairs with platforms). The work presented in this thesis takes advantage of the computed trajectory of the detected strides, which is precious information for activity recognition. As device wearers have various ages and heights, we do not adopt a threshold approach on the length or speed of

strides, which would be not robust to the gait variety. Instead, we built a machine learning algorithm that is able to recognize the stride activity given its trajectory.

Other works with inertial wearable devices are applied in a medical context. For example in [64], a device worn on the shoe is used for gait analysis and automatic classification of Parkinson's diseases using machine learning. Their study has several limitations: it uses a small data set that has been built in a controlled environment. The wearers walked 10 meters four times at their comfortable speed and in an obstacle-free environment. The stride segmentation is given by a tuned threshold on gyrometer and usual signal processing techniques (mean, variance, maximum, minimum). This approach would not be robust to detect strides for home recordings, and they admit the model may have difficulty in generalizing on different data sets.

5 Clinical outcomes for Parkinson disease

Among the common age-related neurodegenerative disorders, Parkinson's disease is currently ranked second. Due to our aging population, its frequency is increasingly growing. France only fosters more than 200 000 sick people. Treatments are being developed and some prove to be efficient but the quantification of their efficiency remains a real challenge, inherently due to the complexity of the symptoms. In Section I.5.1 we briefly describe the symptoms and the Levodopa treatment. In Section I.5.2 we present the state of the art on Parkinson's event detection. Then we will focus in Section I.5.3 on the algorithms with ActiMyo developed by Sysnav for the detection of tremors and dyskinesia, and their limitations.

5.1 Presentation of the disease

Nowadays, the most commonly prescribed treatment is based on Levodopa, a drug known to ease the symptoms of the sufferers [39], such as tremors, bradykinesia or freezing of gait. However this drug has a main drawback: induced dyskinesia crises (see Figure 10). Those side effects are totally unpredictable and unintended movements, and do seemingly not follow obvious patterns.

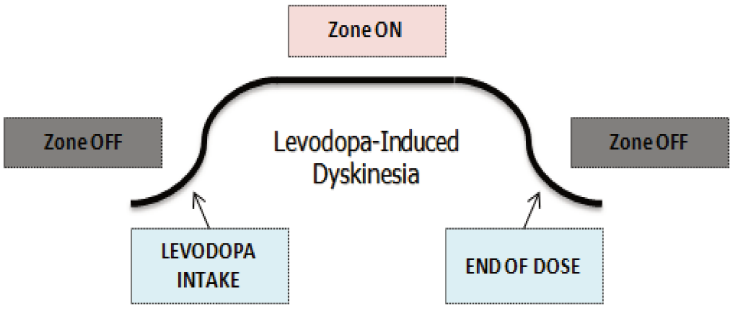


Figure 10: Leodopa side effects.

5.2 Automatic detection of Parkinson's events

Many papers have been written about the detection of dyskinesia crises and tremors. Multiple techniques and inputs have been tested, such as electromyography signals or electroencephalogram signals, along with the use of vision,

unimodal wearable sensors or even audio sensors. However, most papers were restrained either to a controlled environment, the characterization of parkinsonian patients [56], the detection of freezing of gait periods [55, 52], the detection of tremors [4, 49] and bradykinesia [50], the physical activity of patient [71], the limited detection of dyskinesia crises in specific life activities [44], or to a correlation with the UPDRS [63] score given by doctors. Others tried the characterization of ON-OFF states, while detecting dyskinesia based on a thresholding of the spectrum [45]. Nonetheless, some papers [72] implemented a way of doing automatic detection in everyday life environments by learning patterns. To the best of our knowledge, none did really extend their detection out of controlled areas, to test their robustness and generalization ability. It also seems like none did claim to provide an accurate detection of the beginning and the end of a dyskinesia crisis.

5.3 Automatic detection of ankle tremors and dyskinesia with ActiMyo

The system ActiMyo has been worn at the ankle by 14 patients suffering from Parkinson disease during a proof of concept in 2014 and a clinical study in 2017, both dealing with the Levodopa drug. These recordings have been used to develop an automatic detection of two parkinsonian events: tremors and dyskinsia.

First, the algorithm described in Section I.4.4 is used to detect when a stride occurs in the recording. It allows to exclude intervals during which the patient is walking. Motionless intervals are also excluded as they could not correspond to these two events. Tremors have been characterised by repeated movements in frequencies: a peak in a tuned range of frequencies has to be detected in one of the axes of the accelerometer and gyrometer. Then dyskinesia are detected in the ankle proceeding by elimination keeping only phases with a long and strong enough move but which is not associated to walking or tremors. When the ankle moves, it usually implies lifting the foot before returning it rapidly to the floor, eg when walking. So looking at the acceleration and angular velocity at the ankle, for most regular moves, a peak in acceleration is regularly noticeable. So if the amplitude remains large enough during the studied interval, a candidate dyskinesia is considered. If there is a tremor or normal looking leg movements in the phase, then the phase is discarded as not a dyskinesia. The remaining phases are the dyskinesia output.

This approach has been tested during hospital recordings where they are asked to realize specific actions to attest for their UPDRS score. It provides promising result in this kind of controlled environment but fail to perform reliably across a variety of gait motions during home recordings. Indeed, an available comparison was provided by healthy volunteers, who were merely partners of those patients that accepted to wear the device as well. All of them were matching age, and could thus serve as controls for the study. We observed a a lot of false symptom detections.

6 General Contributions and Thesis Outline

In the previous sections we described the algorithms developed by Sysnav for trajectory reconstruction and Parkinson's envent detection (respectively Section I.4.4 and Section I.5.3). They had several limitations we want to overtake in the work of this thesis with supervised statistical learning algorithms. In Section I.6.1

we introduce the concepts of supervised statistical learning and how to adapt this approach to the data recorded by ActiMyo for stride detection (see Section I.6.2 and Section I.6.3), activity recognition in Section I.6.4 and Parkinson's event detection in Section I.6.5. These works are based on innovative algorithms whose we present the main ideas in the following.

6.1 Supervised Learning for stride detection and Parkinson's events

Due to the complexity of our application framework, the problem in our study is difficult to describe with simple deterministic models, and thus we adopted a machine learning approach. This first section aims to introduce the general concepts of statistical supervised learning for times series. These methods are applied to ActiMyo recordings that contains inertial data: acceleration and angular velocity in three axes. Traditionally, for event detection in time series, the sliding window approach is used to select intervals that are classified by the prediction function previously built with supervised learning. We adopt this technique for tremor and dyskinesia detection and develop an innovative approach for stride detection based on candidate intervals extraction. The principle is to select relevant intervals that may correspond to stride and select among them the true strides by machine learning. Indeed, many intervals are wrongly selected when the wearer is moving its ankle but not walking or even when the device is not worn at the ankle (in a pocket, backpack). This intervals exaction allows to reduce the number of classifications by the prediction function and reduce the statistical complexity.

Let's consider a random vector $(\mathbf{X}, Y)$ taking values in $\mathbb{R}^p \times \mathcal{Y}$ whose the probability distribution $P_{\boldsymbol{X},Y}$ is unknown. Supervised learning algorithms aim to estimate the link between the covariates $\boldsymbol{X} = (X_1, \ldots, X_u, \ldots, X_p)$ (corresponding to the inertial data of an interval in our case) and a variable to be predicted Y (Parkinson's event for example). The estimation of the prediction function f defined on $\mathbb{R}^p$ and taking values in $\mathcal{Y}$ relies on a dataset $\mathcal{D}_n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_i, Y_i), \ldots, (\boldsymbol{X}_n, Y_n)\}$ of n independent and identically distributed couples following the $P_{\boldsymbol{X},Y}$ distribution. One estimator $\hat{f}$ of f allows to predict a new output value Y_{n+1} knowing an new observation $\boldsymbol{X}_{n+1}$.

In practice $\boldsymbol{X}_i$ corresponds to the gyrometer and accelerometer data of the interval i (given by sliding window or candidate intervals extraction). The supervised learning algorithms require that for all i, $\mathbf{X}_i$ takes its values in $\mathbb{R}^p$ with p a fixed integer. A sliding window approach allows us to extract fixed-size interval but the innovative candidate stride extraction does not ensure the same size for all intervals. To overtake this issue, one can compute relevant variables from the inertial data $\boldsymbol{X}_i$ (such as the mean, standard deviation etc.). This procedure is called the features engineering process. Finally it allows to compute $\boldsymbol{Z}_i$ taking values in $\mathbb{R}^d$ regardless of the interval size. We observe now a couple $(\boldsymbol{Z}, Y)$ taking values in $\mathbb{R}^d \times \mathcal{Y}$ where d corresponds to the number of computed features. The supervised learning algorithms can thus be adapted to any interval extraction approach.

6.2 Candidate stride interval extraction based on ground contact

The ActiMyo system should be worn at the ankle, as illustrated in the Figure 11. In this default placement, the sensors record the inertial data in the reference frame defined by the Z axis aligned with the leg and the X axis aligned with the foot. However we observed that the device may be worn upside down and may turn around the ankle during the recording. Typically, in Figure 7, the Y axis is aligned

with the leg (acceleration close to -g when the foot is on the ground). Thus at this

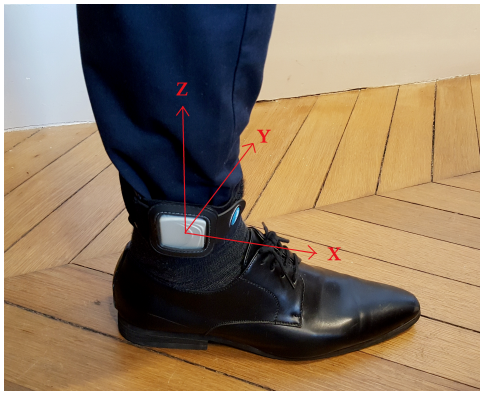


Figure 11: Default device placement.

stage, we have no information about the direction of the different axes of inertial data. To overtake this issue in Chapter III, the norms of the signals are used. We observed that strides induce a peak in the norm of the acceleration when the foot touches the ground. Then the beginning and the end of the interval are defined by local minima of the gyrometer norm around the acceleration peak. Indeed, when the foot is on the ground, the angular velocity of the ankle is lower than during the swing phase. However this approach also extract intervals that do not correspond to stride when for example the user is bicycling. The goal is to select among these intervals which ones are true strides. We adopt a statistical learning approach to answer this problem thanks to a database we have built.

The features engineering process relies on the calculation of a rotation applied on the inertial data in order to work in the same reference frame for all records. This key stage is based on fitting the gyrometer data with 3D reference geometric patterns computed from the database. Then the forward swing movement that occurs just before the foot contact with the ground is studied, providing precious information for the prediction function built from the Gradient Boosting Trees algorithm (see Section II.2.2).

This first stride detection algorithm shows promising performance improvement compared to the existing method developed by Sysnav introduced in I.4.4 especially for atypical strides. However it shows limitations in situations such as fast side stepping and prompt descent of stairs. In addition, this algorithm is computationally expensive as the number of acceleration peaks are in practice caused by a lot of device movements that do not correspond to strides. Consequently the features and the classifications by the prediction function are computed a lot of times along the recording.

6.3 Candidate stride interval extraction based on pseudo-speed

In Chapter IV, the main idea lies in the fact that in an inertial reference frame, the integration of the accelerometer data during a period Δt is equal to the difference of the ankle speed (a few meters per second for a pedestrian) that is small compared to the integration of the gravity. With angular velocity integration during t and $t + \Delta t$, we can compute the rotation matrices $\boldsymbol{R}_t^{t+dt}, \boldsymbol{R}_t^{t+2dt}, \ldots, \boldsymbol{R}_t^{t+\Delta t}$. It allows by composition of the rotation matrices to project the acceleration in the body frame of time t. Indeed by definition $\boldsymbol{R}_t^{t+2dt} = \boldsymbol{R}_t^{t+dt}\boldsymbol{R}_{t+dt}^{t+2dt}$. Then the integration of

$\boldsymbol{\gamma}_t$ during t and $t + \Delta t$ is given by:

$$\begin{aligned}\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{\gamma}_t(u)du &= \frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{a}_t(u)du + \boldsymbol{g}_t \\ &= \frac{\boldsymbol{v}_t(t+\Delta t) - \boldsymbol{v}_t(t)}{\Delta t} + \boldsymbol{g}_t \\ &\simeq \boldsymbol{g}_t,\end{aligned}$$

with $\boldsymbol{g}_t$ a constant. By removing the gravity from the accelerometer data projected in the world frame that is aligned with the identified gravity, we can compute a pseudo-speed $\hat{\boldsymbol{v}}_W$ which is a good feature to detect the beginning and the end of strides. This procedure aims to have a representation of the data in a world frame which is called attitude filter. The computed speed quantity is called pseudo-speed $\hat{\boldsymbol{v}}_W$ because it is not the quantity used for trajectory reconstruction given by the extended Kalman filter that is more accurate. The advantage of this described attitude filter is its computation efficiency.

This algorithm aims to align the sensors in a common reference frame (world frame) in order to take advantage of the three dimensions of the accelerometer and gyrometer, as the previous one briefly described in Section I.6.2. However the computed pseudo-speed is more robust in critical situations and provides, by integration, a pseudo-trajectory that appears to be a key variable for the Gradient Boosting Trees decision function (see Section II.2.2). Indeed, this approach extracts intervals that are not strides when for example the wearer is moving the device in the hand during the required time to install or uninstall the system from the docking station. The selection of the true stride intervals among them is given thanks to this decision function.

6.4 Activity recognition from computed stride trajectory

Sysnav developped an algorithm introduced in Section I.4.4 that allows to compute the trajectory of each detected stride. This algorithm requires a precise estimation of the stride start and stride end before applying a speed estimation of the device based on a lever arm model. Then, the speed estimation is fused with inertial integration in an extended Kalman filter, providing the computed trajectory.

During clinical studies, activity recognition is precious information to evaluate the health of patients suffering from movement disorders. In this work, we focus on three activities related to the primary outcomes for DMD: stairs, walking, and running. However, defining the difference between running and fast walking regarding the trajectory is a challenging task. Indeed, the age difference of patients in clinical studies can be very large, and their gaits very dissimilar. Moreover, detecting stairs is difficult for patients suffering from DMD who can hardly take them and go up the stairs one by one (the difference in the computed altitude is small). Thus, in Chapter V we adopted supervised machine learning algorithm to build a classifier that recognizes the activity of the performed stride given its computed trajectory with features engineering process based on functional data analysis.

6.5 Neural Network for Parkinson's events detection

Chapter VI presents an innovative and generic deep-learning approach for issues based on inertial data recordings. Throughout this work, we focused our attention on the detection of tremors and dyskinesia, concurrently to the issue

of activity recognition. The models have been built in order to precisely detect the time boundaries of such Parkinson's events or activities. Our work is based on multi-channel networks, using in particular Convolutional Neural Networks and one Topological Data Analysis (TDA) channel for multivariate time series that improved the performances. Topological Data Analysis is a recent field that emerged from various works in applied topology and computational geometry, aiming at providing well-founded mathematical, statistical and algorithmic methods to exploit the topological and underlying geometric structures in data (see Section II.3.4). The proposed methodology involves converting the multivariate time series to point cloud and applying methods for findings topological structure that allow to compute features for our Neural Network. Our approach proved its efficiency on activity recognition and Parkinson's event detection, with scores reaching the performances of state of the art methods.

Part II

Supervised learning for stride detection and Parkinson's events detection

Due to the complexity of our applications, the problem we study is difficult to describe with simple deterministic models, and thus we adopted machine learning approaches. In this thesis, machine learning techniques are used at three different stages: for stride detection, activity recognition and Parkinson's events detection. In Section II.1 we describe the principles of supervised statistical learning [81, 35] and in particular the Gradient Boosting Trees (Section II.2) and the Neural Network algorithms (Section II.3). We then explain in Section II.4 how these methods can be applied to our setting.

1 Introduction to Statistical supervised learning

Let's consider a random vector $(\mathbf{X}, Y)$ taking values in $\mathbb{R}^p \times \mathcal{Y}$ whose the probability distribution $P_{\mathbf{X},Y}$ is unknown. Supervised learning algorithms consists of defining an efficient prediction rule between the covariates $\mathbf{X} = (X_1, \ldots, X_u, \ldots, X_p)$ and a variable to be predicted Y. Namely a prediction function f defined on $\mathbb{R}^p$ with values in $\mathcal{Y}$. For instance, for stride interval detection, $\boldsymbol{X}$ is a vector inertial data quantity and $\mathcal{Y} = \{-1, 1\}$, with 1 for a stride and -1 otherwise. The error for a prediction rule f is given by:

$$R(f) = \mathbb{E}_{\boldsymbol{X},Y} \ell(f(\boldsymbol{X}), Y),$$

where ℓ is a loss function. The best prediction function f^*, called Bayes estimator, is then the one that minimizes the previous expression of R, called risk, for the set of measurable functions in $\mathcal{C}$ defined on $\mathbb{R}^p$ with values in $\mathcal{Y}$:

$$f^* \in \underset{f \in \mathcal{C}}{\operatorname{argmin}} R(f).$$

However, as the joint distribution $P_{\boldsymbol{X},Y}$ is unknown we can not compute directly f^*. The technique lies in the estimation of such a function from a dataset $\mathcal{D}_n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_i, Y_i), \ldots, (\boldsymbol{X}_n, Y_n)\}$ of n independent and identically distributed couples following the $P_{\mathbf{X},Y}$ distribution.

1.1 Empirical Risk Minimization

From the dataset $\mathcal{D}_n$ the empirical risk noted $\hat{R}$ is given by:

$$\hat{R} = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{X}_i), Y_i).$$

The function that is zero everywhere except at $\boldsymbol{X}_i$ taking the value Y_i is a minimizer of the empirical risk. This kind of function is not satisfying as it would achieve poor generalization performance (overfitting). Let consider a binary classification problem with $Y_i \in \{-1, 1\}$. A new observation $\boldsymbol{X_{n+1}}$ that do not belongs to the dataset $\mathcal{D}_n$, would be always wrongly classified zero. Hence, we choose the $\hat{f}$ that minimizing the empirical risk over some class of functions $\mathcal{F}$, such as parametric models, histogram classifiers, decision trees or linear/polynomial functions, etc.:

$$\begin{aligned} \hat{f} &\in \underset{f \in \mathcal{F}}{\operatorname{argmin}} \hat{R}(f) \\ &= \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{X}_i), Y_i). \end{aligned}$$

To justifiy this empirical risk minimization method, we need to know how similar the $R(f)$ and $\hat{R}(f)$ are. When $\mathcal{F}$ contains only finite number of classifiers, say N classifiers, then for bounded loss ℓ Hoeffding's inequality shows that with probability at least $1-\delta$ for all f in $\mathcal{F}$:

$$|R(f) - \hat{R}(f)| \leq \sqrt{\frac{logN + log\frac{2}{\delta}}{2n}},$$

Now let consider $\mathcal{F}$ contains infinite number of classifiers. As long as its Vapnik-Tchervonenkis (VC) dimension d [81] is strictly smaller than n, the VC theory tells us that with probability $1-\delta$ we have:

$$|R(f) - \hat{R}(f)| \leq \sqrt{\frac{d\left(log\frac{2n}{d}+1\right) - log\frac{\delta}{4}}{n}}.$$

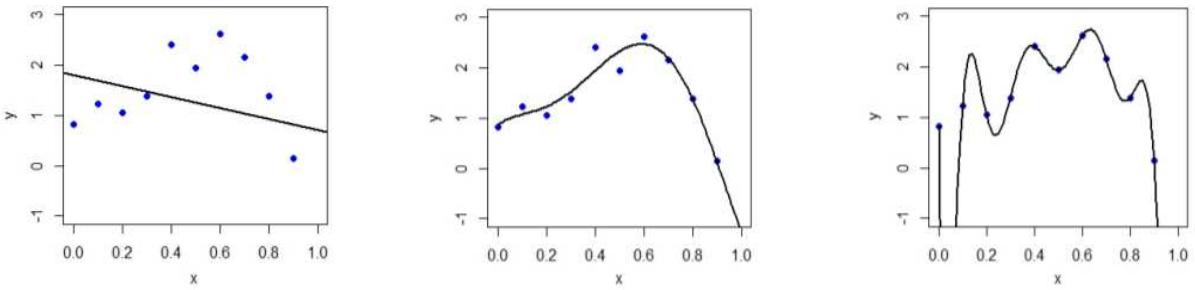
1.2 Bias-Variance Trade-Off in Machine Learning

The performance of $\hat{f}$ based on the empirical risk minimization depends on the class $\mathcal{F}$. One of the main goals of statistical analysis of supervised learning algorithms is to understand how the excess risk defined by the difference between $R(f^*)$ and $R(\hat{f})$ depends on the sample size n, on the complexity of the class $\mathcal{F}$ and on the underlying complexity of the prediction problem itself. Indeed, the excess risk can be expressed by:

$$R(\hat{f}) - R(f^*) = \left(R(\hat{f}) - R(f^*_{\mathcal{F}})\right) + \left(R(f^*_{\mathcal{F}}) - R(f^*)\right),$$

where $f^*_{\mathcal{F}} \in \underset{f\in\mathcal{F}}{\operatorname{argmin}} R(f)$. The first term in the equation, corresponding to the variance, measures the estimation error in the $\mathcal{F}$. The second term, corresponding to the bias, measures the error based on the restriction the class $\mathcal{F}$. Consequently, the more the complexity of $\mathcal{F}$ is big, the more the variance is large and the biais is small. The class $\mathcal{F}$ has to be chosen carefully to reach the best trade-off between the bias and the variance. Namely, the class $\mathcal{F}$ has to be not too large to avoid over-fitting but also not to restrictive in order to approach f^*.

The phenomenon can be illustrated for a polynomial regression. In this case, the complexity of $\mathcal{F}$ corresponds to the polynomial degree. By increasing the degree, the prediction error decreases until zero, meaning the observations of $\mathcal{D}_n$ are all exactly predicted by the interpolation. But in Figure 12, it shows also that the predictions performed outside the set $\mathcal{D}_n$ can reach extreme values and thus large errors.



(**a**) (**b**) (**c**)

Figure 12: Polynomial regression in two dimensions: (**a**) degree 1, (**b**) degree 5, (**c**) degree 10.

One solution to find the best $\hat{f}$ with good bias-variance trade-off compromise is to add a penalization into the empirical risk minimization. Namely, we can consider:

$$\underset{f \in \mathcal{F}}{\operatorname{argmin}} \{\hat{R}(f) + pen_\lambda(f)\},$$

where $pen_\lambda(f)$ allows to penalize the complexity of the models. One challenge is to tune the parameter λ that could yield to a strong overfitting. An alternative method is cross-validation that shows good model selection performances in (almost) any framework. Nevertheless, universality has a price: compared to procedures designed to be optimal in a specific framework, the model selection performances of cross-validation can be less accurate, while its computational cost is higher.

1.3 Risk Estimation

The simplest method to estimate the risk consists in using an independent subset that did not participate to the estimation of the model. It requires to build three sets respectively called learning, validation and test set.

- The training set aims to find the best prediction function for a fixed model (for example a model of fixed polynomial degree).
- The validation set is used to compare models (for example polynomial regression with several degrees).
- The test set allows to compare the best prediction functions of the different tested models.

However this solution requires a large initial dataset size n. In practice, the labelling for the three tasks we consider in this thesis (stride detection, activity recognition, Parkinson's events detection) is costly in time and effort. Consequently the total size of observations with label is small. Several strategies have been proposed to estimate the risk with low bias in this situation. The most popular is cross-validation. The risk estimation is iteratively computed before average in order to reduce the variance and increase the precision. The K-fold cross-validation is described in the Algorithm 1. It consists in randomly splitting the dataset into K subsets and then they iteratively are considered as validation set whereas the $K-1$ others constitute the training set.

Algorithm 1: K-fold cross-validation

Input : K random subsets
Output: Estimation by cross-validation

1 **foreach** $k \in [\![1, K]\!]$ **do**
2 | Consider the k^{th} subset as test set.
3 | Estimate the model on the $K-1$ subsets remaining.
4 | Compute the error on the k^{th} subset.
5 **end**
6 Mean of K errors.

The choice of the K value is generally taken between 5 and 15. If K is small ($K = 5$), the variance is lower but the bias increases. This choice corresponds

again to a trade-off and its optimization is to complicated or requires too many observations to be in practice performed, hence the default value equals to 10. This technique can be repeated several times in order to get the variance of the resulting error score. It is called the Monte Carlo cross validation.

2 Ensemble methods

The initial idea of Freund and Schapire in [73] described the original algorithm of AdaBoost (Adaptative Boosting) for the prediction of a binary variable. Several studies have been published that adjust this algorithm to other situations: multi class, regression etc. These tests have shown that this kind of algorithm reduce drastically the variance but also the bias of prediction. This algorithm have been considered as the best method *of-the-shell* meaning that does not require a long pre-treatment of data neither a fine tuning of parameters during learning. Nonetheless, the evolution towards new versions (extrem gradient boosting) with better performances lead to grow the number of tuned parameters.

Boosting algorithm adopts the same general principle of bagging: building a family of models that are finaly aggregated thought a weighted mean of estimations/votes. But it is different in terms of building the family that is in this case recursively: each model is an adaptative version of the previous ones, by given more weight to observations wrongly estimated for the next estimation. This section aims to describe boosting algorithms, from AdaBoost (Section II.2.1) to Gradient Boosting Trees (Section II.2.2).

2.1 Introduction to Boosting Algorithm through Adaboost

Let describe the original version of boosting for a binary problem with δ the prediction function with values in $\{-1, 1\}$. It is also possible to adapt this version to a regression, namely with a real target values.

The weights of each observation are initialized to $\frac{1}{n}$ for the estimation of the first model and then change for each iteration. The weight of an observation is noted w_i is unchanged if it is correctly classified. In the opposite, it grows proportionally to the error made by the model. The final aggregation of the predictions for a $\boldsymbol{X}$ in $\mathbb{R}^p$ is a sum of the different models weighted by their performance: $\sum_{m=1}^{M} c_m \delta_m(\boldsymbol{X})$.

This kind of algorithm is used with decision tree (CART) as weak learner (decision tree with accuracy just a little bit better than random guessing). Several applications have shown that if the weak learner is a trivial decision tree with only two leafs, AdaBoost reaches better performance compared to a sophisticated tree for a same amount of computational cost: same number of leafs in the sophisticated tree as number of AdaBoost iteration. The goal of boosting algorithm is to find an approximation of f such as:

$$\hat{f}(\boldsymbol{X}) = \sum_{m=1}^{M} c_m \delta(\boldsymbol{x}; \beta_m),$$

where c_m is a parameter depending on the quality of the weak learner of iteration m, δ the weak learner with β_m parameters fonction of $\boldsymbol{x}$. With ℓ the loss function, the goal is to solve at each stage:

$$(c_m, \beta_m) = \underset{(c,\beta)}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(y_i, \hat{f}_{m-1}(\boldsymbol{x}_i) + c\delta(\boldsymbol{x}_i; \beta)).$$

The estimation $\hat{f}_m$ given by $\hat{f}_m(\boldsymbol{x}) = \hat{f}_{m-1}(\boldsymbol{x}) + c_m\delta(\boldsymbol{x};\beta_m)$ is then an amelioration of the previous iteration.

With Adaboost, the loss function is defined by $\ell(y, f(\boldsymbol{x})) = \exp[-yf(\boldsymbol{x})]$. Namely, the goal is to solve:

$$\begin{aligned}(c_m, \beta_m) &= \underset{(c,\beta)}{\operatorname{argmin}} \sum_{i=1}^{n} \exp[-y_i(\hat{f}_{m-1}(\boldsymbol{x}_i) + c\delta(\boldsymbol{x}_i;\beta))] \\ &= \underset{(c,\beta)}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^{m-1} \exp[-y_i c\delta(\boldsymbol{x}_i;\beta))],\end{aligned}$$

with $w_i^{m-1} = \exp[-y_i\hat{f}_{m-1}(\boldsymbol{x}_i)]$ independent of c and β. The solution to this optimization problem is computed in two steps: finding the best weak classifier δ_m (namely finding β_m) and then the optimization of the parameter c_m.

$$\beta_m = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} w_i^{m-1}\mathbb{1}\{y_i \neq \delta(\boldsymbol{x}_i;\beta)\}, \tag{1}$$

$$c_m = \frac{1}{2} log \frac{1-\hat{\epsilon}_m}{\hat{\epsilon}_m}, \tag{2}$$

where $\hat{\epsilon}_m$ is the weighted classification error defined by:

$$\hat{\epsilon}_m = \frac{\sum_{i=1}^{n} w_i^{m-1}\mathbb{1}\{\delta_m(\boldsymbol{x}_i) \neq y_i\}}{\sum_{i=1}^{n} w_i^{m-1}}. \tag{3}$$

The weights w_i for i in $[\![1, n]\!]$ are updated with:

$$w_i^m = w_i^{m-1} \exp[-c_m\mathbb{1}\{\delta_m(\boldsymbol{x}_i \neq y_i\}]. \tag{4}$$

We described in this section the boosting algorithm for Adaboost (pseudo-code in Algorithm 2), with exponential loss function. Other loss functions can be used such as $\ell(y, f(\boldsymbol{x}) = log_2(1 + \exp[-2yf(\boldsymbol{x})]$ for LogitBoost. It depends on the dataset $\mathcal{D}_n$, if it presents outliers for example.

Algorithm 2: AdaBoost algorithm

Input : Training set $\mathcal{D}_n$,
Weights initialization $w^0 = \{w_i = \frac{1}{n}; i = 1, \ldots, n\}$,
Number of iteration M.

Output: Prediction function $\hat{f}_M$, estimation of f.

foreach $m \in [\![1, M]\!]$ **do**
  Estimation of δ_m on the training set weighted by w^{m-1} (Equation 1).
  Compute the error $\hat{\epsilon}_m$ (Equation 3).
  Compute c_m (Equation 2).
  Update the weights (Equation 4).
end
Resulting prediction function $\hat{f}_M(\boldsymbol{x}) = sign[\sum_{m=1}^{M} c_m\delta_m(\boldsymbol{x})]$.

Assuming the trees at each iteration are weak learners (a little bit better than random guessing) it is possible to prove that the training error of AdaBoost's final prediction function $\hat{f}_M$ decreases to zero. Furthermore, we can measure the complexity of the final hypothesis using the VC-dimension which can be computed

[30]. By analyzing both the complexity and training fit of the final hypothesis, one can immediately apply the VC theory to obtain a bound on its generalization error. As noted above, we expect training error to drop very quickly, but at the same time, the VC-dimension of the final hypothesis is increasing roughly linearly with the number of iterations M. Thus, with a large value of M, the final prediction function becomes overly complex and leads to overfitting behavior.

2.2 Gradient Boosting Trees

With the same adaptive approach, Freidman proposed in [31] a familiy of Gradient Booting Models (GBM) with a loss function ℓ that is convex and differentiable. Based on the same idea of Adaboost, the goal of Gradient Boosting Trees algorithm is to build a sequence of models that combined together estimate better and better f, but the main innovation is that we use the gradient of ℓ estimated by a regression tree at each step (to avoid overfitting). The previous presented model $\hat{f}_m(\boldsymbol{x}) = \hat{f}_{m-1}(\boldsymbol{x}) + c_m\delta(\boldsymbol{x};\beta_m)$ is now changed in a gradient descent:

$$\hat{f}_m(\boldsymbol{x}) = \hat{f}_{m-1}(\boldsymbol{x}) - \beta_m \sum_{i=1}^{n} \nabla_{f_{m-1}} \ell(y_i, f_{m-1}(x_i)).$$

Despite finding the best weak learner δ_m in Adaboost, the optimization problem is based on finding the descent step. This requires computation of the derivatives $\nabla_{f_{m-1}}\ell(y_i, f_{m-1}(x_i))$, for $i \in \{1,\dots,n\}$. The quantities $r_{i,m} := -\nabla_{f_{m-1}}\ell(y_i, f_{m-1}(x_i))$ are also called pseudo residuals. Now, the problem is that a naive gradient descent approach would yield a predictor that could only be computed on the training set. The solution is to fit a weak learner in the direction of the gradient. The Algorithm 3 summarizes the different steps for a regression problem for regression but it can be adapted to a classification problem.

Algorithm 3: Gradient Boosting Trees algorithm

Input : Training set $\mathcal{D}_n$,
Prediction function initialization $\hat{f}_0 = \underset{\beta\in\mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(y_i, \beta)$
Number of iteration M.

Output: Prediction function $\hat{f}_M$, estimation of f.

1 **foreach** $m \in [\![1, M]\!]$ **do**
2 Compute the pseudo residuals $r_{i,m} := -\nabla_{f_{m-1}}\ell(y_i, f_{m-1}(x_i))$; $i = 1,\dots,n$.
3 Fit a regression tree δ_m to the couples $(\boldsymbol{x}_i, r_{i,m})_{i=1,\dots,n}$.
4 Compute β_m by solving: $\underset{\beta\in\mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{n} \ell(y_i, f_{m-1}(\boldsymbol{x}_i) + \beta\delta_m(\boldsymbol{x}_i))$.
5 Update the prediction function $\hat{f}_m(\boldsymbol{x}) = \hat{f}_{m-1}(\boldsymbol{x}) + \beta_m\delta_m(\boldsymbol{x})$.
6 **end**
7 Return $\hat{f}_M$.

Friedman also proposed in [31] a version named Stochastic Gradient Boosting including a random under-sampling inspired from bagging methods. An other proposition consists in add a learning rate coefficient ν that penalizes the update of the new model in the aggregation:

$$\hat{f}_m(\boldsymbol{x}) = \hat{f}_{m-1}(\boldsymbol{x}) + \nu\beta_m\delta_m(\boldsymbol{x}).$$

A small value of ν requires to increase the number of trees M but in practise it generally improves the final result. These two parameters have to be tuned by cross validation on the training set. More recetnly Chen and Guestrin [20] proposed XGBoost algorithm (Extreme Gradient Boosting). It is a specific implementation of the Gradient Boosting Trees algorithm which uses more accurate approximations to find the best tree model:

- Computing second-order gradients, namely second partial derivatives of the loss function (similar to Newton's method), which provides more information about the direction of gradients and how to get to the minimum of the loss function.
- Advanced regularization (L1 and L2), which improves model generalization.

3 Neural Networks

Deep learning is a set of learning methods attempting to model data with complex architectures combining different non-linear transformations. These techniques have enabled significant progress in the fields of sound and image processing, including facial recognition, speech recognition, computer vision, language processing, text classification etc. There exist several types of architectures for neural networks: Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNN), recurrent neural networks (LSTM). This section aims to introduce the reader MLP principles (Section II.3.1), CNN (Section II.3.3) and their parameters optimization through backpropagation (Section II.3.2). These supervised learning models are used in Parkinson's event detection with an innovative channel for inputs that we introduce in Section II.3.4.

3.1 Transfer function for MLP

An artificial neuron is a function h of the input $\boldsymbol{x} = (x_1, \ldots, x_p)$ weighted by a vector of connection weights $w = (w_1, \ldots, w_p)$, completed by a neuron bias b and associated to an activation function ϕ. Namely, the output is defined by $h(\boldsymbol{x}) = \phi(\boldsymbol{w^T x} + b)$. Several activation functions can be considered, the most popular are the identity function ($\phi(x) = x$), sigmoid function ($\phi(x) = \frac{1}{1+\exp(-x)}$), hyperbolic tangent function noted "tanh" ($\phi(x) = \frac{\exp(2x)-1}{\exp(2x)+1}$), threshold function ($\phi_\beta(x) = \mathbb{1}\{x \geq \beta\}$) and the Rectified Linear Unit activation function noted "ReLu" ($\phi(x) = max(0, x)$). The Figure 13 illustrates the activation function of one artificial neuron where $\sum = \boldsymbol{w^T x}$.

A multilayer perceptron (or neural network) is a structure composed by several hidden layers of neurons where the output of a neuron of a layer becomes the input of a neuron of the next layer (without backfeed loop). On last layer, called ouput layer, different activation function are applied depending on the type of problems (regression or classification). The Figure 14 represents a neural network with three input variables, one output variable and two hidden layers. The parameters of the architecture are the number of hidden layers and of neurons in each layer. The activation functions are also chosen by the user for each layer. Usually for regression task, the last layer is defined by one neuron with the identity activation function. For binary classification, the output neuron is defined with the sigmoid activation function whereas for a classification problem of M classes, the last layer contains M neurons (one neuron per class) with a

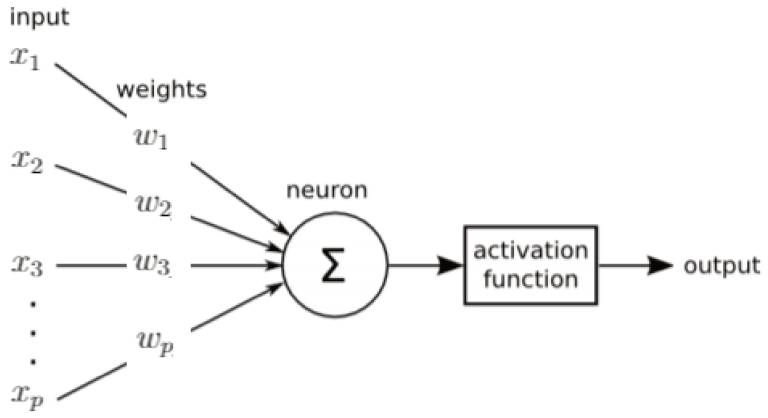


Figure 13: Schematic representation of an artificial neuron.

softmax activation function. This M values sum is equal to 1 and correspond to a posteriori probabilities ($P(Y = i/X)$) per class. Let $a_j = \sum_j + b_j$ the value to be activated by the neuron j of the last layer, we have:

$$softmax(a)_i = \frac{\exp(a_i)}{\sum_{j=1}^{M} \exp(a_j)}.$$

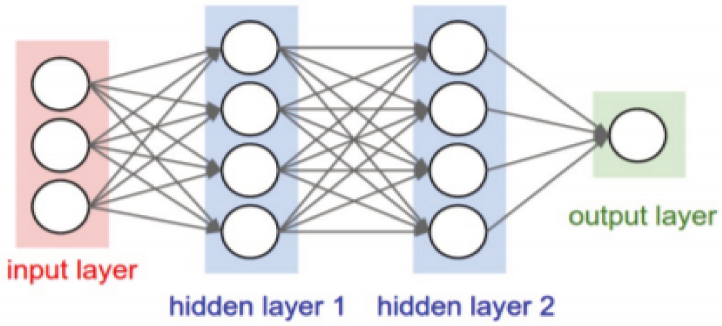


Figure 14: Schematic representation of basic neural network.

Let us summarize the mathematical formulation of a multilayer perceptron with L hidden layers. the variable k indicates the considered layer and j the neuron of the layer. We note $a^{(k)}(\boldsymbol{x})$ the input of the layer k and $h^{(k)}(\boldsymbol{x})$ the output of the layer k. We set $h^{(0)}(\boldsymbol{x}) = \boldsymbol{x}$ and $a^{(k)}(\boldsymbol{x}) = \boldsymbol{x}$. Then for $k \in [\![1, \ldots, L]\!]$ (L hidden layers):

$$\begin{aligned} a^{(k)}(\boldsymbol{x}) &= \boldsymbol{b}^{(k)} + \boldsymbol{W}^{(k)} h^{(j-1)}(\boldsymbol{x}), \\ h^{(k)}(\boldsymbol{k}) &= \phi(a^{(k)}(\boldsymbol{x})). \end{aligned}$$

For $j = L + 1$ (output layer):

$$\begin{aligned} a^{(L+1)}(\boldsymbol{x}) &= \boldsymbol{b}^{(L+1)} + \boldsymbol{W}^{(L+1)} h^{(L)}(\boldsymbol{x}), \\ h^{(L+1)}(\boldsymbol{x}) &= \Phi(a^{(L+1)}(\boldsymbol{x})) := f(\boldsymbol{x}, \boldsymbol{\theta}). \end{aligned}$$

We keep the notation ϕ the activation function and Φ is the output layer activation function depending on the objective task. (softmax for multiclass classification). At each step, $\boldsymbol{W}^{(k)}$ is a matrix with number of rows equals to the number of neurons in the layer k and number of columns equals to the number of neurons in the layer $k - 1$.

3.2 Parameters Optimization through Backpropagation

The set of parameters to be tuned is noted $\boldsymbol{\theta}$, composed of the weights and biases. As usual, the estimation is obtained by minimizing a loss function with a gradient descent algorithm. For multi-class classification problem with M number of classes, we consider loss function defined by:

$$\ell(\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{X},Y}[\sum_{j=1}^{K} \mathbb{1}_{Y=j} \ln f(\boldsymbol{X}, \boldsymbol{\theta})].$$

In order to estimate the parameters $\boldsymbol{\theta}$, we use a training sample $(\boldsymbol{X}_i, Y_i)_{1 \leq i \leq n}$ and we minimize the empirical risk:

$$\hat{R} = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{X}_i), Y_i).$$

We can add a regularization term to penalize the empirical risk, that leads to minimize the following empirical loss:

$$L_n(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{X}_i), Y_i) + \lambda \Omega(\boldsymbol{\theta}).$$

Usually, L2 regularization is used:

$$\begin{aligned} \Omega(\boldsymbol{\theta}) &= \sum_k \sum_i \sum_j (\boldsymbol{W}_{ij}^{(k)})^2 \\ &= \sum_k ||\boldsymbol{W}^{(k)}||_F^2, \end{aligned}$$

where $||\boldsymbol{W}||_F$ denotes the Frobenius norm of the matrix $\boldsymbol{W}$. Only the weights are penalized here, not the biaises. This penalization is used because we can compute its gradient during the gradient descent algorithm by the following equation:

$$\nabla_{\boldsymbol{W}^{(k)}} \Omega(\boldsymbol{\theta}) = 2\boldsymbol{W}^{(k)}.$$

In order to minimize the empirical loss $L_n(\boldsymbol{\theta})$, a stochastic gradient descent is used. It computes the gradient for the loss function at each step of the algorithm but only on a subset B of cardinality m (called a batch) taken randomly without replacement. A large m value allows to speed up the algorithm but it leads to poor generalization. An iteration over all the training samples is called an epoch. The number of epochs is a parameter to be fixed by the user. The total number of iterations N equals the number of epochs times the sample size n divided by m. This procedure is called batch learning and is presented in Algorithm 4. As

we can compute the gradient of the loss function, the gradient vector of the loss function indicated the direction of a growing error. Thus to decrease the loss value, we need to get the inverse direction of the gradient. The learning rate τ can be fixed by the user, its choice is crucial for the convergence of the SGD algorithm. Variations of the algorithm have been proposed to update its value during the learning. Intuitively, at first we an take a large value to speed up the process until decreasing the value when the system is close to a solution.

The parameters update consists in an iterative algorithm that modify the weights for each neuron, called backpropagation algorithm that has

Algorithm 4: Stochastic Gradient Descent (SGD) algorithm

Input : Initialization of $\boldsymbol{\theta} = (\boldsymbol{W}^{(1)}, \boldsymbol{b}^{(1)}, \ldots, \boldsymbol{W}^{(L+1)}), \boldsymbol{b}^{(L+1)}$
Number of epochs
Batch size m
Number of iterations N.
Output: Optimization of $\boldsymbol{\theta}$

for N *iterations* **do**
  Parameters update: $\boldsymbol{\theta} = \boldsymbol{\theta} - \tau \frac{1}{m} \sum_{i \in B} [\nabla_{\boldsymbol{\theta}} \ell(f(\boldsymbol{X}_i, \boldsymbol{\theta}, Y_i) + \lambda \nabla_{\boldsymbol{\theta}} \Omega(\boldsymbol{\theta})]$
end
Return $\boldsymbol{\theta}$.

been introduce in [68]. It is summarized in Algorithm 5 with no regularization term to simplify the equations, but it can be easily added.

Algorithm 5: Backpropagation algorithm

Input : the values of the current fixed weights of iteration r :
$\boldsymbol{\theta}^{(r)} = (\boldsymbol{W}^{(1,r)}, \boldsymbol{b}^{(1,r)}, \ldots, \boldsymbol{W}^{(L+1,r)} =, \boldsymbol{b}^{(L+1,r)}))$.
Output: Update of $\boldsymbol{\theta}$

Forward pass:
Compute the predicted values $f(\boldsymbol{X}_i, \boldsymbol{\theta}^{(r)})$.
Store all the intermediate values $\left(a^{(k)}(\boldsymbol{X}_i), h^{(k)}(\boldsymbol{X}_i) = \phi(a^{(k)}(\boldsymbol{X}_i))\right)_{1 \leq k \leq L+1}$.
Backpropagation algorithm:
Compute the output gradient $\nabla_{a^{(L+1)}(\boldsymbol{x})} \ell(f(\boldsymbol{x}), y)$
foreach $k \in \{L+1, L, \ldots, 1\}$ **do**
  Compute the gradient at the hidden layer k :
    $\nabla_{\boldsymbol{W}^{(k)}} \ell(f(\boldsymbol{x}), y) = \nabla_{a^{(k)}} \ell(f(\boldsymbol{x}), y) h^{(k-1)}(\boldsymbol{x})^T$,
    $\nabla_{\boldsymbol{b}^{(k)}} \ell(f(\boldsymbol{x}), y) = \nabla_{a^{(k)}} \ell(f(\boldsymbol{x}), y)$.
  Compute the gradient at the previous layer $k-1$:
    $\nabla_{h^{(k-1)}(\boldsymbol{x})} \ell(f(\boldsymbol{x}), y) = (\boldsymbol{X}^{(k)})^T \nabla_{a^{(k)}} \ell(f(\boldsymbol{x}), y)$,
    $\nabla_{a^{(k-1)}} \ell(f(\boldsymbol{x}), y) = \nabla_{h^{(k-1)}(\boldsymbol{x})} \ell(f(\boldsymbol{x}), y) \odot (\ldots, \phi(a^{(k-1)}(x)_j)^T, \ldots)^T$,
  where $\odot$ is the element-wise product.
end
Return $\boldsymbol{\theta}$.

For some types of data, especially for images, multilayer perceptrons are not well adapted. Indeed, they are defined for vectors as input data, hence, to apply them to images, we should transform the images into vectors, losing the spatial information.

3.3 Convolutional Neural Networks

The Convolutional Neural Networks (CNN) introduced in [48] are widely used for image classification, object recognition etc. They have revolutionized image processing as they act directly on matrices or even tensors for images with three RGB color chanels. Indeed a black and white picture is defined by a matrix with values corresponding to the pixel intensity. But a colored image has three chanels, namely, we have three values for each pixel that represent the lel of red, green and blue.

Kernel

As a image, the kernel is a matrix but with generally smaller dimensions (in $\mathbb{R}^{3\times3}$). It allows to capture the characteristics of an image (color, luminosity, bounds etc.). It is applied iteratively step by step along the considered image (see Figure 15) by computing the convolution. At each kernel position, we get the convolution

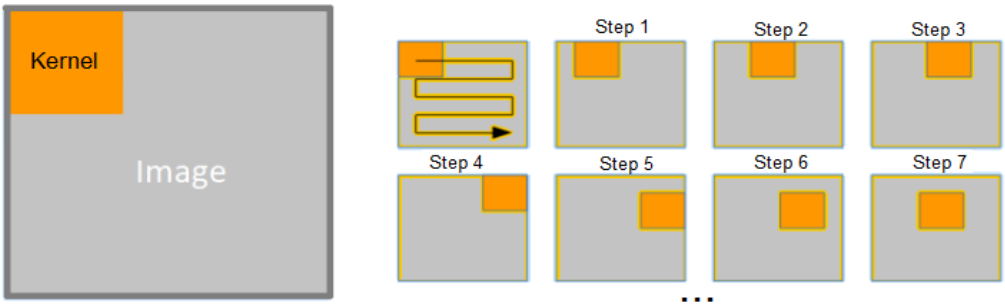


Figure 15: Schematic representation of a kernel applied on an image.

between the kernel and the part of the image that is currently treated (see Figure 16). For 2-dimensional signals such as images, we consider the 2D-convolutions:

$$(\boldsymbol{K} * \boldsymbol{I})_{ij} = \sum_{m,n} \boldsymbol{K}_{mn} \boldsymbol{I}_{i+n,j+m}.$$

The kernel moves by a number s of pixels, called step. When the step is small

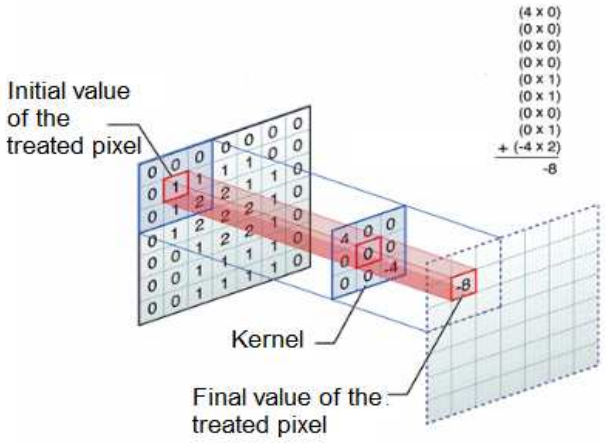


Figure 16: Example of a convolution between the kernel and one part of the image.

we get redondant information. Sometimes, zero padding is used that is a margin of size p adding zero values around the image in order to control the size of the ouput. Let consider an image in $\mathbb{R}^{W_0\times H_0\times C_0}$ where W_0 denotes the width, H_0 the height and C_0 the number of channels (typically $C_0 = 3$ for RGB images). With a kernel in $\mathbb{R}^{k\times k}$, the size of the ouput is in $R^{W_1\times H_1\times C_1}$ and we have:

$$\begin{aligned} W_1 &= \frac{W - k + 2p}{s} + 1 \\ H_1 &= \frac{H_0 - k + 2p}{s} + 1. \end{aligned}$$

The value of C_1 corresponds to the number of kernels that we used, called depth. This defines the number of characteristics we want to detect. The Figure 17 shows two different transformations with two kernels traditionally used

for bounds detection and blurring. These parameters have to be tuned empirically regarding the type of data. Taking small value of k or a number of C_1 too large will usually lead to over-fitting. On the contrary, using few kernels with big size may not provide sufficient details from the treated image. The strength of a convolution layer consists in learning the kernels that are the most useful for the task.

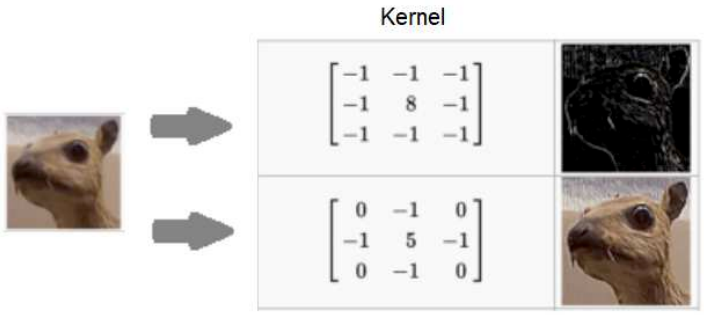


Figure 17: Example of one convolution layers provided by two different kernels.

Architecture

The convolution operations are combined with an activation function ϕ, usually the Relu activation function (see Section II.3.1). Another powerful tool called pooling layer is used in CNNs. It allows to reduce the size of an image by keeping the main relevant characteristics. The most common used method is max pooling, consisting in reducing the image and retaining the largest values of pixels. Like the convolutional layer, pooling layers act on small part of the image and slide along all the image step by step. In practice, we consider a window of size 2×2 or 3×3, over which we take the maximum value to define the output layer (see Figure 18). Other pooling methods exist such as average pooling that computes the

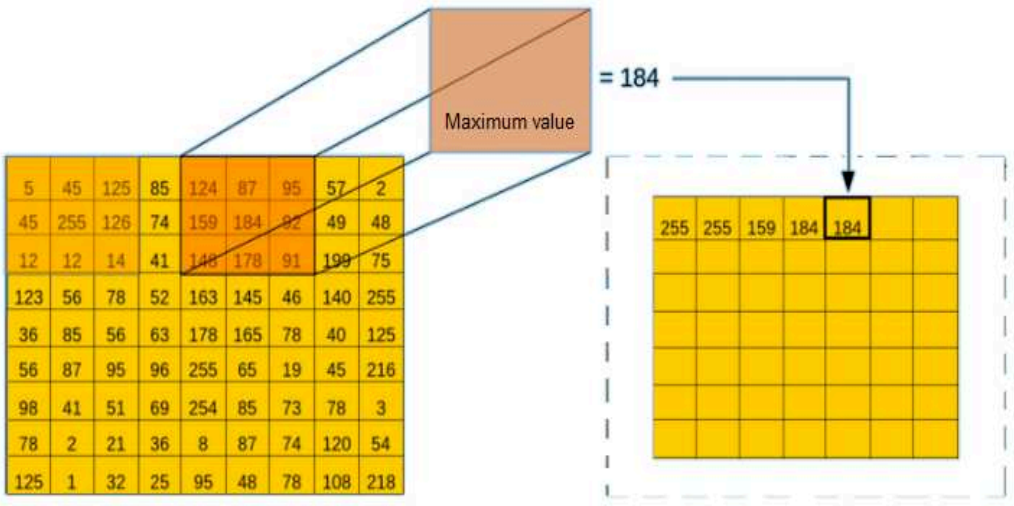


Figure 18: Example of one max pooling layer.

mean of the window, stochastic pooling that keeps one value under probability. Reducing the dimension is also achievable by the convolutional layer, considering a step size s larger than 1 and without zero padding. But the main advantage of pooling layer is that it is less sensitive to small translations of the input images.

The particularity of these two layers is that their outputs can be directly considered as new images which can be treated as the initial input ones. At each

stage, the outputs become more and more complex. In the most classical CNN, we chain several times a convolution layer followed by a pooling layer and we add at the end fully connected layers (MLP). The famous LeNet network, proposed in [47], developed to digit recognition, is composed only on few layers and few filters due to the computer limitations at that time. The Figure 19 shows in details the resulting outputs of each stage.

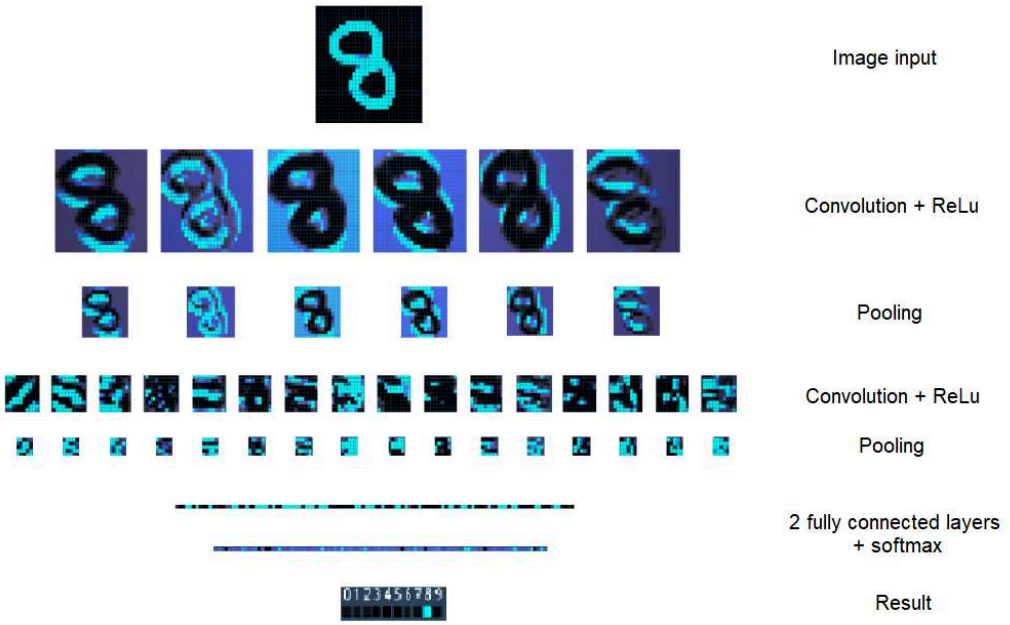


Figure 19: Example of the layers outputs for digit recognition with LetNet network.

Learning and tuning the whole model

The hyperparameters to be tuned by the user are the number of kernels for each convolutional layer, their size and the type of pooling to apply. For fully connected layers, the number of layers and number of neurons is crucial. To a higher level, one has to chose the basic architecture, namely the number of convolutional layers etc. The learning phase is similar to the described one in Section II.3.2. We need a data base of labelled images. They are passed through the CNN and the resulting errors inform about the quality of the weights and biases of each neuron in the convolutional of fully connected layers. They are updated through backpropagation in order to reduce the errors. The elements that appear rarely into the images are skipped with small weight value whereas relevant patterns that are found regularly in the set of images are combined with high weight value.

Beyond images

Although CNNs were used at first for images classification, their success has motivated research to adapt them to other types of data. For multivariate time series of W_0 dimensions, we can split the overall signal into smaller parts with smaller fixed duration H_0. We obtain matrix in $\mathbb{R}^{W_0 \times H_0}$ where the lines correspond the W_0 variables data and the columns correspond to the time samples. Keeping the notations introduce in Section II.3.3, we can consider this matrix as an image with pixels indicating the intensity of the signal values. The higher is the signal data, the higher is the pixel value. Typically, with inertial data in three dimensions provided by ActiMyo recordings, we can build "images" in $\mathbb{R}^{3 \times H_0}$. For

1D signals, one method is to consider a split time duration equals to $W_0 \times H_0$ and build the matrix with the first line corresponding to the H_0 first sample values, then the interval $[H_0, 2 \times H_0]$ correspond to the second line of the matrix etc. However, An alternative version of 2D CNNs called 1D Convolutional Neural Networks have been developed for 1D signals (for example in ECG classification [70]). Several studies have shown that for certain applications 1D CNNs are advantageous and thus preferable to their 2D counterparts in dealing with 1D signals due to several reasons. Indeed rather than matrix operations, the backpropagation require simple array operations, reducing the computational complexity. In addition recent studies show that 1D CNNs with relatively shallow architectures (small number of hidden layers and neurons) are able to reach good performances when 2D CNNs usually require deeper architectures to handle the same scores.

Other techniques allow to extract relevant information from multivariate time series that feed the last fully connected layer in the architecture presented above. One can compute variables from signal processing techniques such as maximum, mean, standard deviation, root mean square, inter-quantile range, Fast Fourier Transform, 3^{rd} and 4^{th} order moments, auto-correlation and correlations between signals etc. In this thesis, we developed a framework for analyzing multivariate time series using topological data analysis (TDA) methods. The proposed methodology involves converting the multivariate time series to point cloud data and exploit the underlying geometric structures through persistent homology. The experimental results on "DB1" and "DB2" has shown that it allow to compute features that are missed by traditional signal processing techniques. In the following section, we briefly introduce the basics of persistence used in this thesis. We refer the reader to [28, 17, 13] for more details.

3.4 Topological Data Analysis Channel

Topological Data Analysis (TDA) refers to a collection of methods for findings topological structure in data ([14, 19]). The input is a dataset drawn from a probability measure supported on an unknown set $\mathbb{X}$. The ouput is a collection of data summaries that are used to describe the topological features of $\mathbb{X}$. Homology, or more precisely persistent homology ([12]), appears as a fundamental tool for TDA. It associates to any topological space $\mathbb{X}$, a family of vector spaces (the so-called homology groups) $H_k(\mathbb{X})$, $k = 0, 1, \ldots,$ each of them encoding topological features of $\mathbb{X}$. The k^{th} Betti number of $\mathbb{X}$, denoted β_k, is the rank of $H_k(\mathbb{X})$ and represents the number of k-dimensional features of $\mathbb{X}$: for example, β_0 is the number of connected components of $\mathbb{X}$, β_1 the number of independent cycles or "tunnels", β_2 the number of "voids", etc. (see [36]). Persistent homology ([29, 86, 7]) provides a framework and efficient algorithms to encode the evolution of the homology of families of nested topological spaces indexed by a set of real numbers that may often be seen as scales, such as the union of growing balls, or nested family of simplicial complexes built on top of the data. The obtained multiscale topological information is then represented in a simple way as a barcode or persistence diagram, providing relevant information about the data ([22, 16]). Persistence diagrams coming as sets of intervals are not well-suited for direct use to standard machine learning method and need to be converted into a vector representation to design a neural network architecture that can take them as input features. Several approaches and methods have been recently proposed to achieve this task (see [38, 79, 3, 25, 15]). In this work, we use the functional

summary called a persistence landscape ([13]). These landscapes are the data summaries that we focus on this section and are used to compute inputs of the last fully connected layer in the neural net architecture.

Simplicial complexes

To compute the persistent homology from a set of data we need to construct a set of structures called simplicial complexes. We use the point cloud $\{\boldsymbol{x}_\alpha$ in a metric space (Euclidean space in practice) as the vertices of a combinatorial graph whose edges are determined by proximity (vertices within some specified distance ϵ). We complete the graph to a simplicial complex, a space built from simple pieces (simplicies) identified combinatorially along faces. The choice of how to fill in the higher dimensional simplices of the proximity graph allows for different global representations. In the following we present two methods for doing so given a collection of points $\{\boldsymbol{x}_i\}$ in a Euclidean space $\mathbb{E}^n$:

- The Cech complex, $\mathcal{C}_\epsilon$, is the abstract simplicial complex chose k-simplices are determined by unordered $(k+1)$-tuples of points $\{\boldsymbol{x}_i\}_0^k$ whose closed $(\frac{\epsilon}{2})$-ball neighborhoods have a point of common intersection.

- The Vietoris-Rips complex, $\mathcal{R}_\epsilon$, is the abstract simplicial complex whose k-simplices correspond to unordered $(k+1)$-tuples of points $\{\boldsymbol{x}_i\}_0^k$ which are pairwise within distance ϵ.

Note that these two complexes are related by $\mathcal{R}_\epsilon \subseteq \mathcal{C}_\epsilon \subseteq \mathcal{R}_{2\epsilon}$. In Figure 20 we represent a set of points completed to a Cech complex and Vietoris-Rips complex (Figure 21).

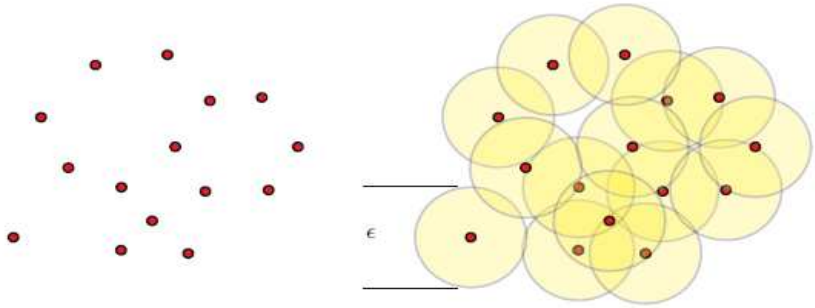


Figure 20: A fixed set points with proximity parameter ϵ (source: [33]).

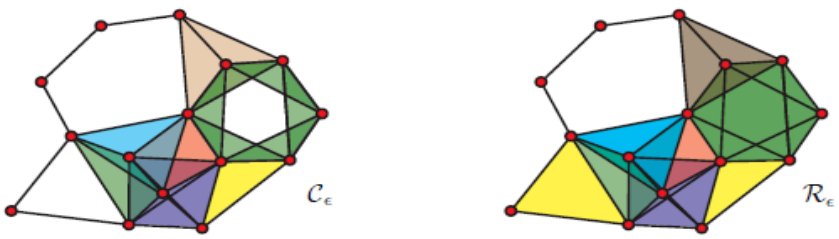


Figure 21: Cech complex $\mathcal{C}_\epsilon$ (left) and Vietoris-Rips complex (right) computed from the point clouds in Figure 20 (source: [33]).

Filtrations

Converting a point cloud data set into a global complex (whether Vietoris-Rips, Cech, or other) requires a choice of parameter ϵ. For a value of ϵ sufficiently small the complex is a discrete set, for ϵ sufficiently large the complex is a single high-dimensional simplex. The optimal choice for ϵ which best captures the topology of the dataset is a challenging task. Consider the point cloud data set and a sequence of Vietoris-Rips complexes as illustrated in Figure 22. This point cloud is a sampling of points on a planar annulus. From the figure, it appears that an ideal choice of ϵ is difficult: by the time ϵ is increased so as to remove small holes from within the annulus, the large hold distinguishing the annulus from the disk is filled in.

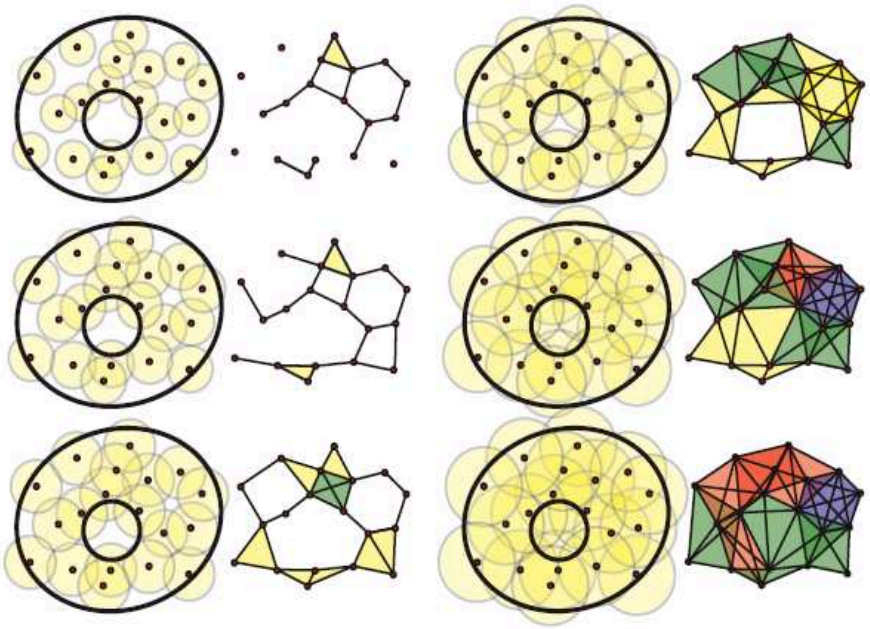


Figure 22: A sequence of Vietoris-Rips complexes for a point cloud data set representing an annulus (source: [33]).

Each family described above is non-decreasing with ϵ : for any $\epsilon \leq \alpha$, there is an inclusion of $\mathcal{R}_\epsilon$ in $\mathcal{R}_\alpha$, and similarity for the Cech complex. These sequences of inclusions are called filtrations.

Examining the homology for one specific value of ϵ is insufficient. The counts of the number and types of holes appearing at each parameter value, namely Betti numbers, is not relevant. Indeed one requires a means of declaring which holes are relevant and which can be ignored. The Figure 23 shows the effect of outliers. Adding just a few outliers to a point cloud may dramatically change the topology of its offsets.

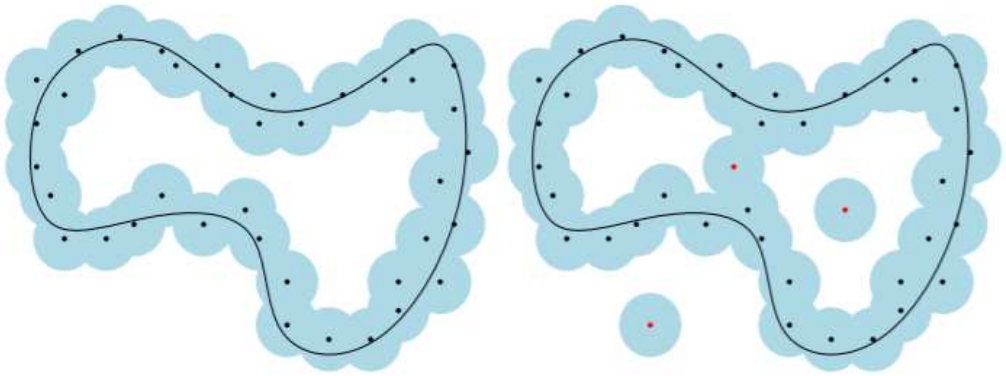


Figure 23: Outliers effect on the topology.

In the following, we let $Filt_\epsilon(\mathbb{X})$ denotes a filtration of the metric space $\mathbb{X}$ corresponding to one of the parameterized complexes defined above.

Persistence diagrams

We saw that the topology of $Filt_\epsilon(\mathbb{X})$ changes as ϵ increases: new connected components can appear, existing connected components can merge, cycles and cavities can appear or be filled etc. Persistent homology tracks these changes, identifies features and associates an interval or lifetime (from birth b to death d) to them. For instance, a connected component is a feature that is born at the smallest ϵ such that the component is present in $Filt_\epsilon(\mathbb{X})$, and dies when it merges with an older connected component. A barcode allows a graphical representation of this procedure. It is a collection of horizontal line segments in a plane whose horizontal axis corresponds to the parameter ϵ and whose vertical axis represents an (arbitrary) ordering of homology generators. Equivalently, the lifetime of a feature can be represented as a point in the plane with coordinates (b, d). The obtained set of points (with multiplicity) is called the persistence diagram $D(Filt(\mathbb{X}))$ (and we will abuse terminology slightly by denoting it $D_\mathbb{X}$). Note that the diagram is entirely contained in the half-plane above the diagonal since death always occurs after birth.

As an example, in Figure 24 we consider the filtration given by a union of growing balls centered on the finite set of points:

a) For the radius $\epsilon = 0$, the union of balls is reduced to the initial finite set of point, each of them corresponding to a 0-dimensional features, namely a connected component. An interval is created for the birth for each of these features at $\epsilon = 0$.

b) Some of the balls started to overlap resulting in the death of some connected components that get merged together. The persistence diagram keeps track of these deaths, putting an end point to the corresponding intervals as they disappear.

c) New components have merged giving rise to a single connected component and all the intervals associated to a 0-dimensional feature have been ended, except the one corresponding to the remaining components. Two new 1-dimensional features have appeared resulting in two new intervals (in blue) starting at their birth scale.

d) One of the two 1-dimensional cycles has been filed, resulting in its death in the filtration and the end of the corresponding blue interval.

e) All the 1-dimensional features have died, it only remains the long (and never dying) red interval. As in the previous stages, the final barcode can also be equivalently represented as a persistence diagram where every interval (b, d) is represented by the point of coordinate (b, d) in $\mathbb{R}^2$.

Intuitively the longer is an interval in the barcode, or equivalently the farther from the diagonal is the corresponding point in the diagram, the more persistent ,and thus relevant, is the corresponding homological feature across the filtration. Notice also that for a given radius ϵ, the k-th Betti number of the corresponding union of balls is equal to the number of persistence intervals corresponding to k-dimensional homological features and containing ϵ. So, the persistence diagram can be seen as a multiscale topological signature encoding the homology of the union of balls for all radii as well as its evolution across the values of ϵ.

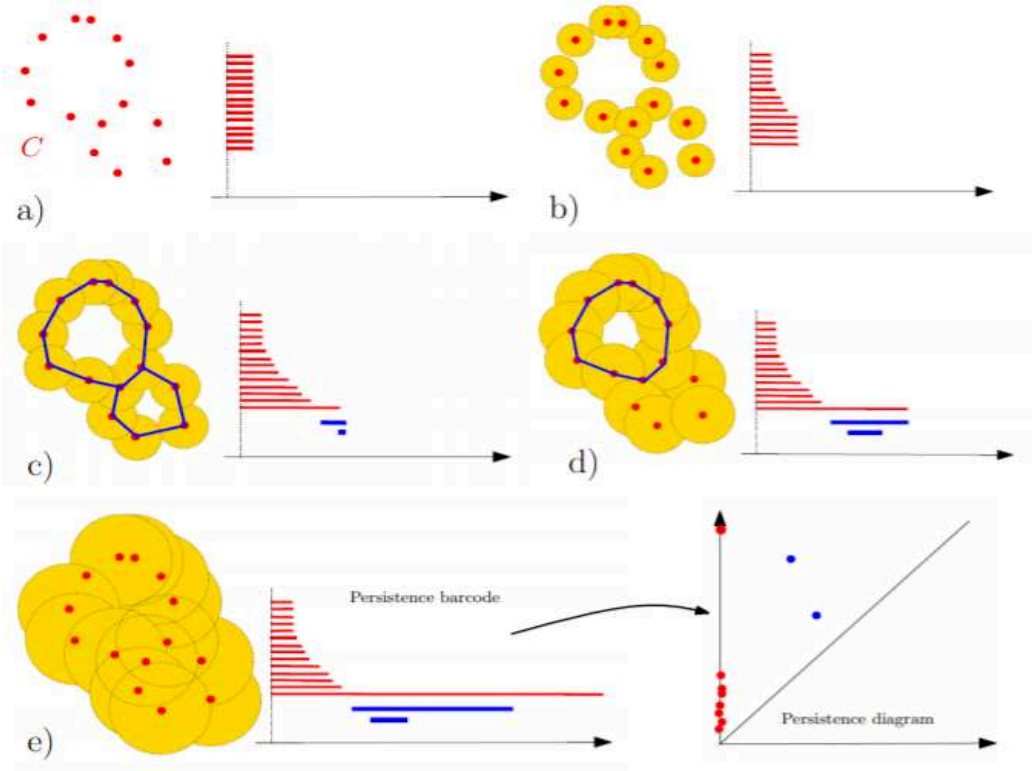


Figure 24: The sublevel set filtration of the distance function to a point cloud and the "construction" of its persistence barcode as the radius of balls increases and its persistence diagram.

To avoid technical difficulties, we restrict our attention to diagrams D such that $(b, d) \in [0, T] \times [0, T]$ for all $(b, d) \in D$, for some fixed $T > 0$. We denote $\mathcal{D}_T$ the space of all such persistence diagrams and we endow it with a metric called the bottleneck distance d_b. Given two persistence diagrams, the bottleneck distance is defined as the infimum of the δ for which we can find a matching between the diagrams, such that two points can only be matched if their distance is less than δ and all points at distance more the δ from the diagonal must be matched.

A fundamental property of persistence diagrams proven in [18] is their stability. The Hausdorff distance between two compact subsets X, Y of a metric space $(\mathbb{X}, \rho)$ is $H(X, Y) = \max\{\max_{x\in X} \min_{y\in Y} \rho(x, y), \max_{y\in Y} \min_{x\in X} \rho(x, y)\}$. If $\mathbb{X}$ and $\tilde{\mathbb{X}}$ are two compact metric spaces, then one has:

$$d_b(D_{\mathbb{X}}, D_{\tilde{\mathbb{X}}}) \leq 2d_{GH}(\mathbb{X}, \tilde{\mathbb{X}}), \tag{5}$$

where $d_{GH}(\mathbb{X}, \tilde{\mathbb{X}})$ denotes the Gromov-Hausdorff distance.

Persistence Landscapes

The persistence landscape, introduced in [13] is a collection of continuous, piecewise linear functions $\lambda : \mathbb{Z}^+ \times \mathbb{R} \to \mathbb{R}$ that summarizes a persistence diagram. To define the landscape, consider the set of functions created by tenting each point $p = (x, y) = \left(\frac{b+d}{2}, \frac{d-b}{2}\right)$ representing a birth-death pair (b, d) as follows:

$$\Lambda_p(t) = \begin{cases} t - b & t \in \left[b, \dfrac{b+d}{2}\right], \\ d - t & t \in \left[\dfrac{b+d}{2}, d\right], \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

We obtain an arrangement of piecewise linear curves by overlaying the graphs of the functions $\{\Lambda_p(t)\}_p$. The persistence landscape is a summary of this arrangement. Formally, the persistence landscape of the persistence diagam D is the collection of functions

$$\lambda_D(k, t) = \underset{p}{\text{kmax}}\, \Lambda_p(t), \quad t \in [0, T], k \in \mathbb{N}, \tag{7}$$

where kmax is the k^{th} largest value in the set. In particular 1max is the usual maximum function. In Figure 25, we use the rotated axes to represent a persistence diagram D. A feature $(b, d) \in D$ is represented by the point $\left(\frac{b+d}{2}, \frac{d-b}{2}\right)$ (pink). Namely, the x-coordinate is the average parameter value over which the feature exists, and the y-coordinate is the half-life of the feature. The cyan curve is the landscape $\alpha(1, .)$.

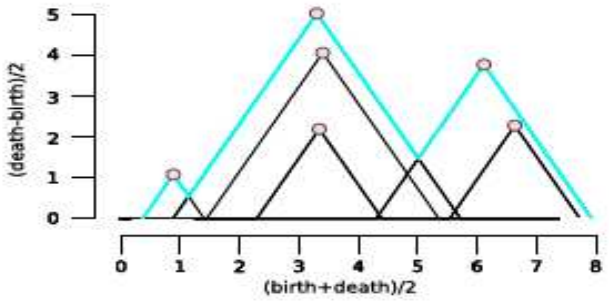


Figure 25: Example of persistence landscapes.

This representation is convenient for machine learning algorithm as the persistence landscapes can be used as 1D signals. In Chapter VI we will see how we use them as inputs of our Neural Network for Parkinson's event detection.

4 Applications to ActiMyo data

The work of this thesis includes three uses of supervised machine learning algorithms: stride detection, activity recognition and Parkinson's event detection. In the previous sections we introduced the mathematical tools and the main supervised statistical learning algorithms used for these three tasks. They require labeled databases that we present in Section II.4.1.

For stride detection, we have seen in Section I.4.3 that threshold based algorithms do not perform well for atypical strides (stairs, small steps...). In addition,

in Section I.4.2 the state of the art machine learning algorithms only achieve good performance when it is known that the wearer is walking but do not perform well in uncontrolled environments, namely at home. These methods consist of building a prediction function that outputs binary zero velocity classifications for every sample of the recording. In this thesis we develop an innovative approach for stride detection based on candidate intervals extraction. The following Section II.4.2 describes this task that can be sum up in two stages:

- Extract candidate intervals from the ActiMyo data that may correspond to strides.
- Select among the extracted intervals the true strides by machine learning with a binary classification function.

This procedure allows to compute the trajectory of each detected stride that is a main feature for activity recognition (Section II.4.3). Indeed, faced to the variety of gait motions, we adopted supervised machine learning algorithm to build a classifier that recognizes the activity of the performed stride from the ActiMyo data and its computed trajectory that greatly helps the decision.

In Section II.4.4 we present our sliding window approach for Parkinson's event detection with machine learning. We also present our use of topological data analysis (TDA) introduced in Section II.3.4, a recent field aiming to exploit the topological and underlying geometric structures in data. Applying this newly spreading method to our problematic seemed relevant when looking at the 3D point clouds representation of inertial data. It will feed one channel in our Neural Network merged with CNNs for tremors and dyskinesia detection.

4.1 Databases description

Our three supervised learning tasks require labelled datasets to compute the corresponding prediction functions (see Section II.1). In the following, we describe the recordings that have been used to build the databases for the model selection and recordings that have not been used in the learning but still with ground truth for the validation of the three algorithms.

Parkinson's event dataset "DB1"

The system ActiMyo has been worn at the ankle by 14 patients suffering from Parkinson disease during a proof of concept in 2014 and a clinical study in 2017, both dealing with the Levodopa. This drug aims to ease the symptoms such as tremors but has a main drawback: induced dyskinesia crises (Section I.5.1). Those patients underwent a Levodopa test at the hospital, while wearing the sensors at the ankle and the wrist, so that every record made could be annotated by the doctors through real-time observation. The resulting labelled database per patient is represented in Figure 26. The first observation is that all patient do not show all Parkinson's events, whether on the ankle or on the wrist. Among the labelled events, we focused our effort for tremor detection and more actively on dyskinesia crisis detection. Ten of the patients did wear the devices at home as well, enabling us to gather the corresponding data. An available comparison was provided by healthy volunteers, who were merely partners of those patients that accepted to wear the device as well. All of them were matching age, and could thus serve as controls for the study.

One major problem we faced is the annotation confidence. Typically we realized for example that some of tremors events labelled on wrist could occur only

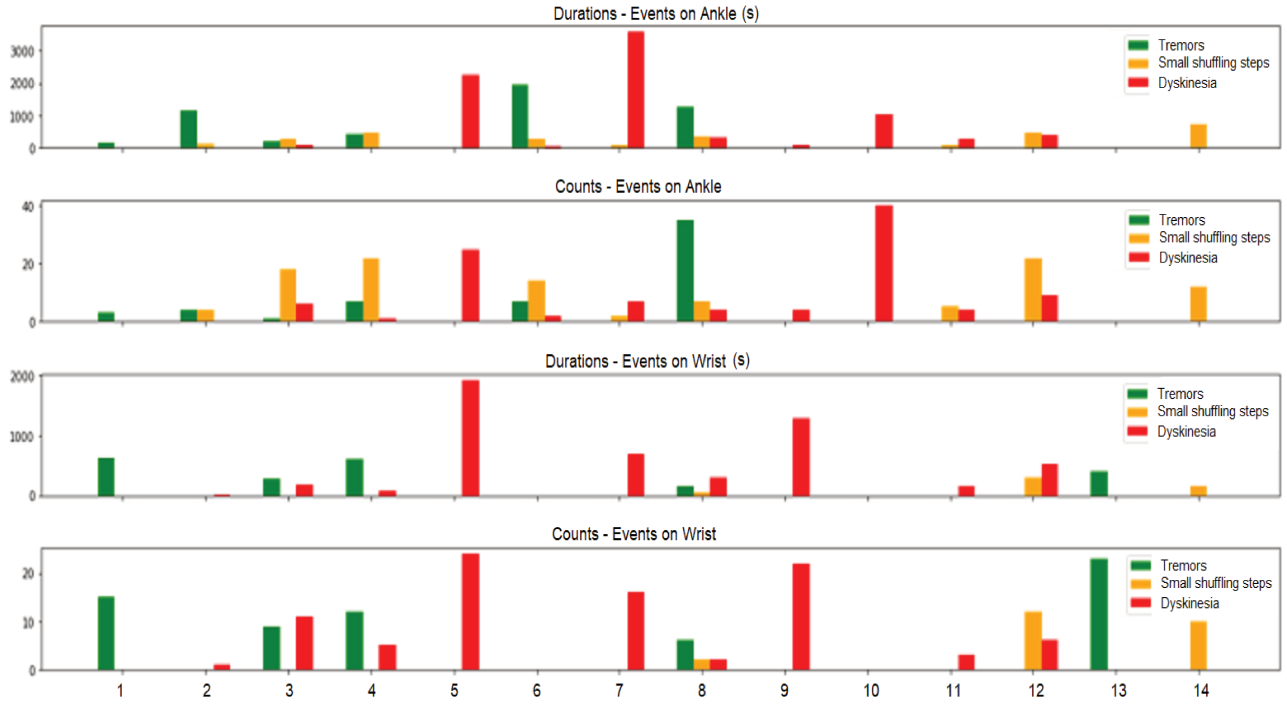


Figure 26: Patient-relative labelled database.

to the fingers. Consequently they are not recorded by the sensors worn at the wrist and it is not relevant to build a supervised learning approach aiming to detect the event from these sensors signals. The same situation may also appear for dyskinesia events. An other kind of error we have detected is when small pauses occur in long time duration events. These pauses are not considered and still annotated with the same label. In addition, the non-annotated events during hospital recordings can not be considered as non Parkinson's events with 100% confidence. Thus recordings that we consider as labelled "other", namely non Parkinson's events, are only provided by the controls at home. It was not possible to re-watch hours of video recording a second time with a doctor and an ActiMyo sensors expert from Sysnav in order to correct the errors in the labelled database.

Among the encountered problematics of ground truth, this database presents two particularities that add difficulty for the supervised learning approach. The amount of daily home recordings from the controls are enormous (several months of studies) whereas the total tremors and dyskinesia crises duration is several hours. Thus this database is large (heavy to learn with supervised learning algorithms) and unbalanced, namely proportions between the targeted classes are very different. Indeed, one daily recording of 8-9 hours with ActiMyo 130 Hz frequency provides almost 4 millions of data points in three dimensions for acceleration and angular velocity (both ankle and wrist). Hence, several months of clinical study correspond to billions of labelled data points recorded by controls. Instead, the annotated events by the doctors represent barely 2 millions of annotated data points on the ankle and wrist. This motivated us to build our algorithm on another database built for activity recognition from inertial sensors, presenting the same unbalanced particularity but with smaller amount of data.

HAPT Dataset "DB2"

This open-source dataset is a newest version of the UCI HAR Dataset. The data is made of 30 volunteers who were gathered within an age range of 19-48 years,

to perform a set of six basic activities (standing, sitting, lying, walking, upstairs and downstairs). Inertial data about those activities and their transition were recorded by a smartphone placed on their waist, sampling the 3-axial acceleration and the 3-axial angular velocity at a frequency of 50 Hz. As one may notice in Tabular 1 and Tabular 2, this database is also linked to imbalance learning due to the poor amount of gathered events for transition activities. In addition, the activity recognition task is very similar to Parkinson's event detection as we can consider tremors or dyskinesia as particular activities of patients that we want to detect among all the other daily activities. The total recordings duration is about 4 hours which makes it easier to test different approaches compared to Parkinson's database. With 50 Hz frequency, it corresponds to 720 thousands recording points versus billions for dataset "DB1".

	Walking	**Upstairs**	**Downstairs**	**Sitting**	**Standing**	**Laying**
Ratio	15%	14%	13%	14%	16%	18%

Table 1: Basic Activities Ratios of 4 hours recording.

	Stand to Sit	**Sit to Stand**	**Sit to Lie**	**Lie to Sit**	**Stand to Lie**	**Lie to Stand**
Ratio	1%	1%	1%	1%	1%	1%

Table 2: Transition Activities Ratios of 4 hours recording.

Co-workers at Sysnav "DB3"

A dozen of co-workers of various ages and heights were filmed practicing several activities while wearing the system at the ankle and the wrist at Sysnav company under video control. At the ankle, the device has been carefully fixed with the default placement (see Figure 11), namely the X axis of the body frame is aligned with the foot and the Z axis is aligned with the leg. This precision is important for our first algorithm presented in Chapter III. With synchronisation between the video and ActiMyo data time scales, it is easy to annotated when a stride occurs in the ankle recordings and precise the activity of the performed stride among "atypical steps", "walking", "running", "upstairs" and "downstairs". This dataset will be used to build algorithms for stride detection and activity recognition both. These recordings contain about 6000 strides balancedly distributed among the activities. Depending on the activity, one stride corresponds to approximately 130 recorded points by ActiMyo. Hence, these recordings require a labelling process with video control of almost one million points.

Co-workers in Motion Capture (MOCAP) environment "DB4"

A group of seven people wore the ActiMyo with infrared markers during MOCAP sessions in a 25 m^2 room (Figure 27). Several cameras were set in order to film the whole scene. They broadcast infrared radiation that was reflected by the markers. This allowed the camera to record the position of the markers with sub-millimeter accuracy. We faced three issues here. Firstly the tracking system samples the data at 240 Hz whereas ActiMyo sensors samples at 130 Hz. Secondly the triangulation of the markers position do not correspond to the position of the ActiMyo sensors. Finally, the data provided by the optical tracking system

Figure 27: ActiMyo device in MOCAP environment.

are recorded in a terrestrial reference frame defined by the user at the beginning of the motion trial that we can not easily project in the ActiMyo data reference frames. C.-Y. Chesneau developped in [21] the smoothing algorithm providing the synchronization between ActiMyo gyrometer data angular velocities recorded by the tracking system markers and the estimation of the inertial sensors of the device compared to the triangulation point of the markers. This allows us to study the evolution of the device altitude in the terrestrial reference frame in ActiMyo time scale, that is an important feature to detect when the foot was on the ground regarding altitude minima. The Figure 28 represents the altitude and the inertial data of the ActiMyo device during 5 walking strides after applying the smoothing algorithm. Then it is easy to give a precise estimation of time intervals when the foot touches the ground. They correspond to periods when the MOCAP altitude is stable and close to 15 cm on the graph. Indeed, during walking the ankle (ActiMyo position) reaches its smallest altitude when the foot is on the ground.

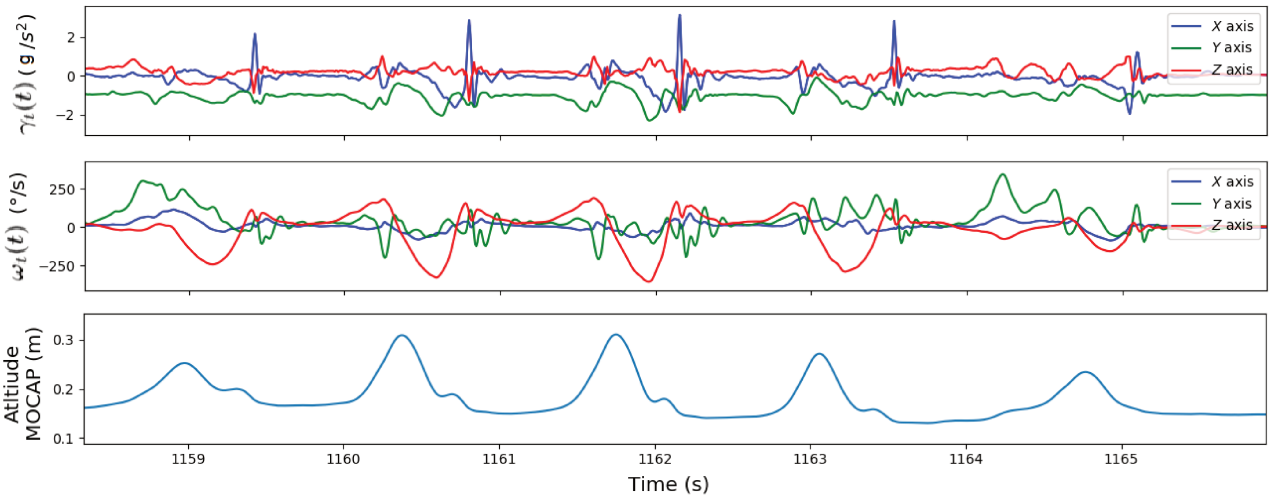


Figure 28: Inertial data in body frame with synchronized MOCAP altitude during walking.

We asked the seven wearers to perform three different walking paces, small steps, and side steps (both sides). In order to observe left and right turns and straight lines, the wearers had to follow a loop path in both directions and a figure-eight-shaped reference trajectory. These sets of recordings will be used for evaluating our stride detection algorithms regarding the rate of missing strides. They represent more than 100 thousands recording points that have been an-

notated thanks to the MOCAP altitude defining when the foot is on the ground. In addition, we can compare the difference between the computed stride length with the ground truth (MOCAP stride length) for each detected stride.

DMD hospital tests "DB5"

During hospital tests for DMD clinical trials (see Section I.3.2), several patients were wearing the ActiMyo system at the ankle under video control. We have access to the recordings of 10 patients performing the 10 meters run test and 6 minutes walk test. Moreover, 8 patients performed the 4-stairs test. As one goal of this thesis is to compute statistics on strides trajectory and strides activity on home recordings of patients suffering from Duchenne muscular dystrophy, this dataset is particularly important. It will allow us to test our stride detection and activity recognition algorithms on patients recordings with ground truth. All recordings together provide hundreds of walking strides, more than one hundred of running and upstairs strides. With 130 Hz frequency, one stride length can vary between few dozens of points to more than several hundreds. They have been manually labelled with the video control.

4.2 Candidate intervals extraction for stride detection

In this thesis, the stride detection task results in a binary classification problem. Indeed, we developed an approached based on candidate intervals extraction that may correspond to stride. The stride selection among them is given by a binary classifier that has been computed with a supervised learning algorithm. In the following we introduce the notations defining the extracted intervals and their labelling procedure that allow to build a dataset for supervised learning algorithms.

Interval extraction

We develop two algorithms for candidate stride interval extraction. To define the beginning and the end in the intervals, the first approach described in Chapter III uses the inertial data norm whereas the second one in Chapter IV is based on a pseudo-speed norm computation. However due to the complexity of our application framework, the problem in our study is difficult to describe with simple deterministic models for both extraction methods. Indeed the combination of criteria to apply on the inertial norm or pseudo-speed norm have to be sufficiently wide to detect all types of strides (atypical strides, running, stairs etc.). Consequently, when these two extraction algorithms are launched on the recordings from the database "DB3", it leads to build a family of candidate intervals with several intervals that may not correspond to strides. They are defined by one start and one end, and we note the family of intervals $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1), \ldots, (\hat{s}_i, \hat{e}_i), \ldots, (\hat{s}_n, \hat{e}_n)\}$[1]. In the following we will consider a generic $\hat{\mathcal{I}}_{DB3}$ without introducing notations defining the extraction method but the number of extracted intervals can change along with. Some of the intervals in the family $\hat{\mathcal{I}}_{DB3}$ correspond to real strides, with correct start and end times tying in moments when the foot is on the ground. Others come from movements that are not strides and that we want to exclude (wrist movements, bicycling etc.).

[1]Note to the reader that $\hat{\mathcal{I}}$ does not defined an estimation here although conventionally we use hat for estimated version of a quantity.

Ground truth annotation

With the video control, we can note precisely when the foot is in contact with the ground during the stance phases in "DB3". Each stride starts with a stance phase before the swing phase and finishes with another stance phase (see Figure 29). We note $\mathcal{I}_{DB3} = \{(s_{1,1}, s_{1,2}, e_{1,1}, e_{1,2}), \ldots, (s_{n,1}, s_{n,2}, e_{n,1}, e_{n,2})\}$ the family of strides

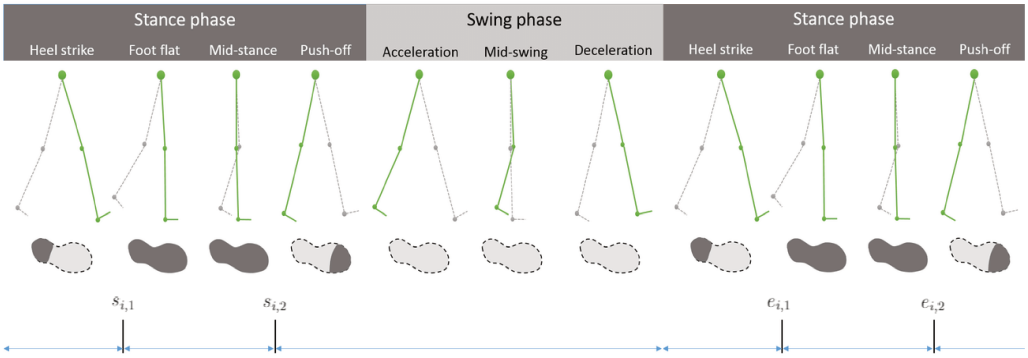


Figure 29: Walking gait cycle and measurement update cycle.

($n \approx 6000$) with $(s_{i,1}, s_{i,2})$ and $(e_{i,1}, e_{i,2})$ delimiting the first and last stance phase of the stride i. Then we affected a label $y_j^s = 1$ (exponent notation s for stride detection) to the interval j in $\hat{\mathcal{I}}_{DB3}$, if and only if there exists a stride i in $\mathcal{I}_{DB3}$ for which $\hat{s}_j \in [s_{i,1}, s_{i,2}]$ and $\hat{e}_j \in [e_{i,1}, e_{i,2}]$. If not, we affected a label $y_j^s = -1$ meaning the interval j do not correspond to a stride ($y_j^s = -1$ for wrist recordings intervals). Finally we have a family of intervals $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, y_1^s), \ldots, (\hat{s}_i, \hat{e}_i, y_i^s), \ldots, (\hat{s}_n, \hat{e}_n, y_n^s)\}$. At this point, one extraction methods aims to build a familiy such as for each stride i of I_{BD3} there exists an extracted interval with the start in $[s_{i,1}, s_{i,2}]$ and the end $[e_{i,1}, e_{i,2}]$ while having the smallest family size n possible, namely with only a few intervals labelled -1.

Labelled database for learning

Our approach needs to compute a prediction function $\hat{f}^s$ from Gradient Boosting Trees algorithm that allows to predict the target y_i^s from the inertial data of the extracted interval i. We saw in Section II.1 that this machine learning algorithm requires the variables to be the same for all inputs. However these two interval extractions do not ensure the extracted intervals length to be equal for all intervals.

To overtake this issue we compute the same number of relevant variables from the inertial data of each interval. On the one hand, this procedure called features engineering process has the advantage of no longer having different inputs dimension. On the other hand, computing relevant variables helps the GBT algorithm to reach better prediction performances. During the features engineering process of the two extraction methods described respectively in Section III.2 and in Section IV.3, the main idea is to project the inertial data in a reference frame with the same definition for each interval. Indeed the device may be worn upside down and may turn around the ankle during the recording leading to different axes definition. In the default placement, the sensors record the inertial data in the reference frame defined by the Z axis aligned with the leg and the X axis aligned with the foot. However in Figure 7, the Y axis is aligned with the leg (acceleration close to -g when the foot is on the ground). One could only compute features from the norm of the signals but it misses a lot of relevant information

contained in each axis. In Section III.2, the sensors alignment is based on the computation of a rotation matrix fitted with geometric patterns built from the database. In Section IV.3, we compute a terrestrial reference frame with gravity identification that allows to compute a pseudo-trajectory. Then the data are projected to have the Z axis aligned with the gravity and X axis is defined by the beginning and the end of the pseudo-trajectory.

From the features engineering procedure, the family of extracted intervals is now defined by $\{(\boldsymbol{x}_1^s, y_1^s), \ldots, (\boldsymbol{x}_n^s, y_n^s)\}$ where $\boldsymbol{x}_i^s \in \mathbb{R}^{p^s}$ with p^s the number of computed features. With the same notations introduced in this Section, we can tune the parameters of the GBT algorithm by 10-fold cross validation (Algorithm 1). This provides a prediction function $\hat{f}^s$ that can be applied for a new extracted interval (after features computation). The overall procedure is sum up in the following Algorithm 6. Then for the recordings in "DB4" for example, we can detect the strides with Algorithm 7.

Algorithm 6: Learning a binary prediction function for interval classification

1 Compute the intervals extraction algorithm on the recordings of "DB3" and affect a label with video control for each extracted intervals: 1 if it is a true stride, -1 if not. It provides a family of extracted intervals:
$\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, y_1^s), \ldots, (\hat{s}_i, \hat{e}_i, y_i^s), \ldots, (\hat{s}_n, \hat{e}_n, y_n^s)\}$.
2 Compute features engineering process with sensors alignment:
$\hat{\mathcal{I}}_{DB3} = \{(\boldsymbol{x}_1^s, y_1^s), \ldots, (\boldsymbol{x}_n^s, y_n^s)\}$ with $x_i^s \in \mathbb{R}^{p^s}$.
3 Tune the parameters of GBT algorithm with 10-fold cross-validation providing the prediction function $\hat{f}^s$ defined on $\mathbb{R}^{p^s}$ taking values in $\{-1, 1\}$.

Algorithm 7: Detecting strides in "DB4" recordings

1 Compute the intervals extraction algorithm providing a family of candidate intervals:
$\hat{\mathcal{I}}_{DB4} = \{(\hat{s}_1, \hat{e}_1), \ldots, (\hat{s}_n, \hat{e}_n)\}$.
2 Compute the features for each extracted intervals:
$\hat{\mathcal{I}}_{DB4} = \{(\hat{s}_1, \hat{e}_1, \boldsymbol{x}_1^s), \ldots, (\hat{s}_i, \hat{e}_i, \boldsymbol{x}_i^s), \ldots, (\hat{s}_n, \hat{e}_n, \boldsymbol{x}_n^s)\}$ with $\boldsymbol{x}_i^s \in \mathbb{R}^{p^s}$.
3 Apply the prediction function $\hat{f}^s$:
$\hat{\mathcal{I}}_{DB4} = \{(\hat{s}_1, \hat{e}_1, \boldsymbol{x}_1^s, \hat{y}_1^s), \ldots, (\hat{s}_i, \hat{e}_i, \boldsymbol{x}_i^s, \hat{y}_i^s), \ldots, (\hat{s}_n, \hat{e}_n, \boldsymbol{x}_n^s, \hat{y}_n^s)\}$ with $\hat{f}^s(\boldsymbol{x}_i^s) = \hat{y}_i^s$.
4 Exclude the intervals classified as non stride, namely $\hat{y}_i^s = -1$.

The beginning and the end of the detected strides are given by $\{\hat{\mathcal{I}}_{DB4} \mid \hat{y}_i^s = 1, i = 1, \ldots, n\} := \tilde{\mathcal{I}}_{DB4}$. In order to simplify the notation in the following, we introduce $\{\hat{\mathcal{I}}_{DB3} \mid \hat{y}_i^s = 1, i = 1, \ldots, n\} := \tilde{\mathcal{I}}_{DB3}$ and $\{\hat{\mathcal{I}}_{DB3} \mid y_i^s = 1, i = 1, \ldots, n\} := \bar{\mathcal{I}}_{DB3}$. Namely, The symbol $\bar{\mathcal{I}}_{DB3}$ represents the detected strides that are true strides and $\tilde{\mathcal{I}}_{DB3}$ represents all the detected strides. Consequently $\bar{\mathcal{I}}_{DB4} \subset \tilde{\mathcal{I}}_{DB4}$ and a perfect stride classification on "DB4" leads to $\bar{\mathcal{I}}_{DB4} = \tilde{\mathcal{I}}_{DB4}$.

4.3 Activity Recognition from computed trajectory

Sysnav developped an algorithm introduced in Section I.4.4 that allows to compute the trajectory of each detected stride that is a main feature for activity recognition. This algorithm requires a precise estimation of the stride start and

end before applying a speed estimation of the device based on a lever arm model. Then, the speed estimation is fused with inertial integration in an extended Kalman filter, providing the computed trajectory. This algorithm has been launched on the database "DB3" from the annotation of the strides indices of the family $\bar{\mathcal{I}}_{DB3}$. With video control, we affected a label y^a (exponent a for activity) defining the activity of the performed stride among "atypical strides" (label 1), "walking" (label 2), "running" (label 3), "upstairs" (label 4) and "downstairs" (label 5). Thus we have a family of intervals corresponding to strides $\bar{\mathcal{I}}_{DB3} = \{(\bar{s}_1, \bar{e}_1, y_1^a), \ldots, (\bar{s}_i, \bar{e}_i, y_i^a), \ldots, (\bar{s}_n, \bar{e}_n, y_n^a)\}$ where $y_i^a \in \{1, 2, 3, 4, 5\}$.

The goal is to compute a prediction function $\hat{f}^a$ from Gradient Boosting Trees algorithm, for multi-class classification task, that allows to predict the target y_i^a from the inertial data and the computed trajectory of the stride intervals in $\bar{\mathcal{I}}_{DB3}$. We faced the same problem as in Section II.4.2, the length of the strides is not a constant whereas the GBT algorithm requires the input variables to be the same for all observations. In addition to the features computed from the inertial data based on functional data analysis, relevant variables are computed from the trajectory that allow to enrich considerably the features for the activity recognition task. It provides a set of observations $\{(\boldsymbol{x}_1^a, y_1^a), \ldots, (\boldsymbol{x}_n^a, y_n^a)\}$ with $\boldsymbol{x}_i \in \mathbb{R}^{p^a}$ for the GBT algorithm. The overall procedure is sum up in the following Algorithm 8. Then for the recordings in "DB5" for example, we can apply the activity recognition with Algorithm 9.

Algorithm 8: Learning a prediction function for activity recognition

1 Affect a label defining the activity for the family of extracted strides in "DB3": $\bar{\mathcal{I}}_{DB3} = \{(\bar{s}_1, \bar{e}_1, y_1^a), \ldots, (\bar{s}_i, \bar{e}_i, y_i^a), \ldots, (\bar{s}_n, \bar{e}_n, y_n^a)\}$ where $y_i^a \in \{1, 2, 3, 4, 5\}$
2 Compute the trajectory from the strides in $\bar{\mathcal{I}}_{DB3}$.
3 Compute the features from inertial data and computed trajectory based on functional data analysis. it provides the set: $\{(\boldsymbol{x}_1^a, y_1^a), \ldots, (\boldsymbol{x}_n^a, y_n^a)\}$ with $\boldsymbol{x}_i^a \in \mathbb{R}^{p^a}$.
4 Tune the parameters of GBT algorithm with 10-fold cross validation providing the prediction function $\hat{f}^a$ defined on $\mathbb{R}^{p^a}$ taking values in $\{1, 2, 3, 4, 5\}$.

Algorithm 9: Activity recognition in "DB5" recordings

1 Compute the strides detection with Algorithm 7
2 Compute the trajectory reconstruction for the detected strides $\tilde{\mathcal{I}}_{DB5} = \{(\tilde{s}_1, \tilde{e}_1), \ldots, (\tilde{s}_i, \tilde{e}_i), \ldots, (\tilde{s}_n, \tilde{e}_n)\}$
3 Compute the features for each detected strides: $\tilde{\mathcal{I}}_{DB5} = \{(\tilde{s}_1, \tilde{e}_1, \boldsymbol{x}_1^a), \ldots, (\tilde{s}_i, \tilde{e}_i, \boldsymbol{x}_i^a), \ldots, (\tilde{s}_n, \tilde{e}_n, \boldsymbol{x}_n^a)\}$ with $\boldsymbol{x}_i^a \in \mathbb{R}^{p^a}$.
4 Apply the prediction function $\hat{f}^a$: $\tilde{\mathcal{I}}_{DB5} = \{(\tilde{s}_1, \tilde{e}_1, \boldsymbol{x}_1^a, \hat{y}_1^a), \ldots, (\tilde{s}_i, \tilde{e}_i, \boldsymbol{x}_i^a, \hat{y}_i^a), \ldots, (\tilde{s}_n, \tilde{e}_n, \boldsymbol{x}_n^a, \hat{y}_n^a)\}$ with $\hat{f}^a(\boldsymbol{x}_i^a) = \tilde{y}_i^a$.

4.4 Parkinson's event detection with sliding window

The parkinsonian events detection was one of the most difficult tasks of this thesis. As far as we know, there is no robust algorithm for detecting dyskinesia crises and tremors from inertial home recordings. Indeed, faced to the variability of daily movements, especially on the wrist, it is impossible to describe with deterministic models these events. Thus, we adopted a machine learning approach.

For this task, the database described in 4.1, also presented difficulties because it was built from doctors annotations who are not expert in inertial systems and in a very controlled environment at the hospital with specific movements. As a result, tagged events are not always captured by the sensors and distort the learning phase.

Therefore we chose to build our supervised machine learning model on the HAPT dataset (see 4.1) which has similar characteristics: an unbalanced inertial database for movement recognition. Indeed, we have access to a lot of daily control recordings, without parkinsonian event, compared to the dyskinesia crises and tremors tagged at the hospital. The HAPT dataset includes basic activities (walking, standing, sitting etc.) but also transition activities underrepresented such as standing to lying. In addition, it is not absurd to construct the detection of parkinsonian events in parallel with activity recognition because a dyskinesia crisis or tremor can be considered as an activity to be detected between all daily activities.

In order to build features for the machine learning tasks, we applied the classical method of the 50% overlapping sliding window on both datasets presented above. Multiple time window widths were tested. Our choices were determined by the minimum length recorded for Parkinson's event annotations, which is 6 seconds. The time window of 3 seconds appeared to be the most efficient for the ankle, while a width of 5 seconds performed better for the wrist. As a consequence, each dataset is composed of split inertial signals, matched with their corresponding annotated events. For Parkinson's envent detection task we are confronted to multi-class classification among "other", "tremor" and "dyskinesia". Moreover, the issue of imbalance learning appeared rapidly, as dyskinesia crises and tremors are minority events. To partially deal with it, we oversampled the minority events thanks to a bigger overlap than for the rest of the signal. In Figure 30 we illustrate the procedure. The same goes for the HAPT dataset, us-

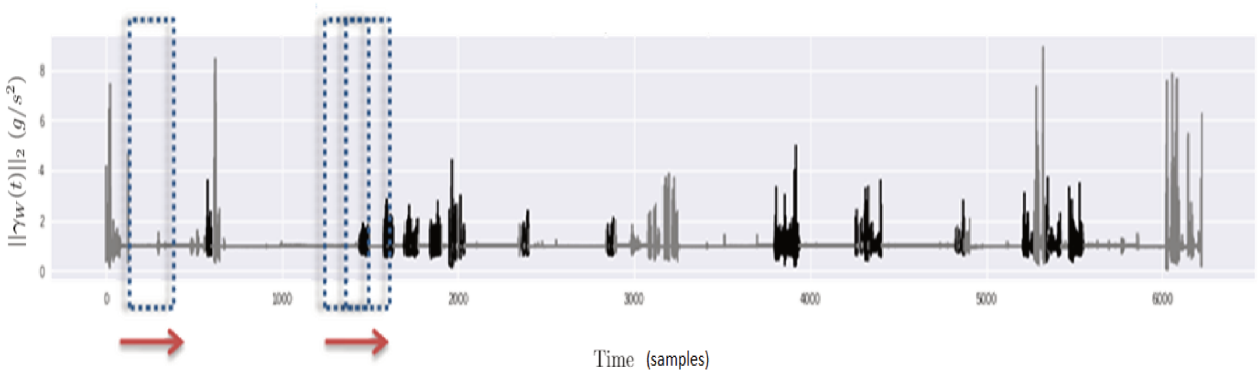


Figure 30: Sliding window with overlap (annotated events in black)

ing the annotation directly given for each recorded inertial signals with a sliding window size equals to 2.56 seconds with 50% overlap. Finally we have access for each labelled window to the acceleration and angular velocity in three dimensions recorded in the body frame of the device.

As it is difficult to model the physics of the sensors for these both tremors and dyskinesia among the variety of daily movements, we used a neural network with channels of different natures: CNN for the raw inertial data, dense network for the handcrafted features resulting from signal processing and one channel computed from Topological Data Analysis (TDA).

Topological Data Analysis Features

It is a recent field that emerged from various works in applied topology and computational geometry (see Section II.3.4). It aims at providing well-founded mathematical, statistical and algorithmic methods to exploit the topological and underlying geometric structures in data based on persistence homology. It is a method aimed at computing topological features of a space at different spatial resolutions. By construction, those features are more likely to represent true characteristics of the underlying space (rather than artifacts of sampling, noise, or particular choice of parameters) as they are intrinsically linked to the spatial relationships of the data points.

To compute the persistence homology of a space, it must first be represented as a nested family of simplicial complex (basically a graph made of a set of points and their relationships: lines and triangles in a two-dimensional space). This family is called a filtration. Generally, the construction of this filtration is based on the definition of a distance function, whose values are used to index the complexes in the family.

The theory of persistence homology allows us to uniquely represent the persistence homology of a filtered simplicial complex with a persistence barcode or persistence diagram. A barcode diagram represents each persistence generator with a horizontal line beginning at the first filtration level where it appears, and ending at the filtration level where it disappears, while a persistence diagram plots a point for each generator with its x-coordinate the birth time and its y-coordinate the death time.

In practice, the 3 dimensions raw signals are converted into point cloud with a sliding window approach and then we build a ball of increasing radius around those points until finding some intersections (birth time). Pursuing this method leads to cover some points and destroy some of the previously created structures (death time). Finally it ends up with components (date of birth and death), which are represented through persistence diagram and barcode.

One problem is that one window does not ensure to compute the same amount of topological features. To overtake this issue we compute the persistence landscapes that is a convenient representation for machine learning algorithm as they can be used as 1D signals. In Chapter VI we will see how we use them as inputs of our Neural Network for Parkinson's event detection.

Part III

Candidate stride interval extraction based on ground contact

This chapter aims to present a first stride detection algorithm. It is based on the extraction of candidate intervals from the inertial norms recorded by the ActiMyo device. Among these intervals, some are true strides and others correspond to movements that are not strides. Thanks to the labeled databases presented in Section III.4.1, we can compute a prediction function calculated by Gradient Boosting Trees algorithm (see Section II.2.2). It allows, with relevant features calculation (see Section III.2), to select from the set of candidate intervals the true strides. In Section III.3, we present its learning performances and the overall algorithm performances on recordings in motion capture environment. The work of this chapter has been published in [9], IEEE conference paper.

1 Intervals Extraction Method

The ActiMyo device records along time the acceleration and the angular velocity in its body reference frame with 130 Hz frequency (respectively $\gamma_t(t)$ and $\omega_t(t)$). The device should be worn at the ankle as illustrated in the Figure 31. In this default placement, the sensors record the inertial data in the reference frame defined by the Z axis aligned with the leg and the X axis aligned with the foot. However we observed that the device may be worn upside down and may turn around the ankle during the recording. Typically, in Figure 7, the Y axis is aligned with the leg (acceleration close to -g when the foot is on the ground).

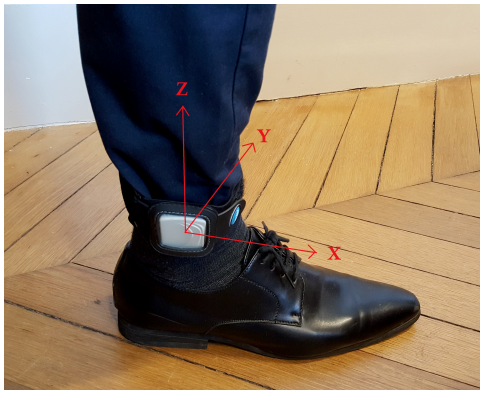


Figure 31: Default device placement.

To overtake the unknown device position that does not allow the interpretation of the inertial data between the three axes, we decided to study the norm of $\gamma_t(t)$ and $\omega_t(t)$. We observed that strides induce a peak in the norm of the acceleration when the foot touches the ground. Then the beginning and the end of the interval are defined by local minima of the angular velocity norm or when the acceleration norm is close to 9.81 (the gravity value in m/s^2). Indeed, when the foot is on the ground, the angular velocity of the ankle is lower than during the swing phase and the specific ankle acceleration is close to zero: the accelerometer sensors record only the gravity. However, the choice among the minima of the angular velocity norm is challenging. In addition, during non walking activities such as bicycling, the same type of patterns in the inertial norm happens. To illustrate the complexity of our application we represent in Figure 32 the end of an ankle recording (inertial norms in blue). The user performed three strides highlighted in green before removing the device and carrying it in his hands (in red). In the red part, the intervals extraction algorithm selected several intervals as it faced lot of acceleration peaks surrounded by angular velocity minima or acceleration values close to 9.81. The Figure 33 shows two extracted intervals, one corresponding

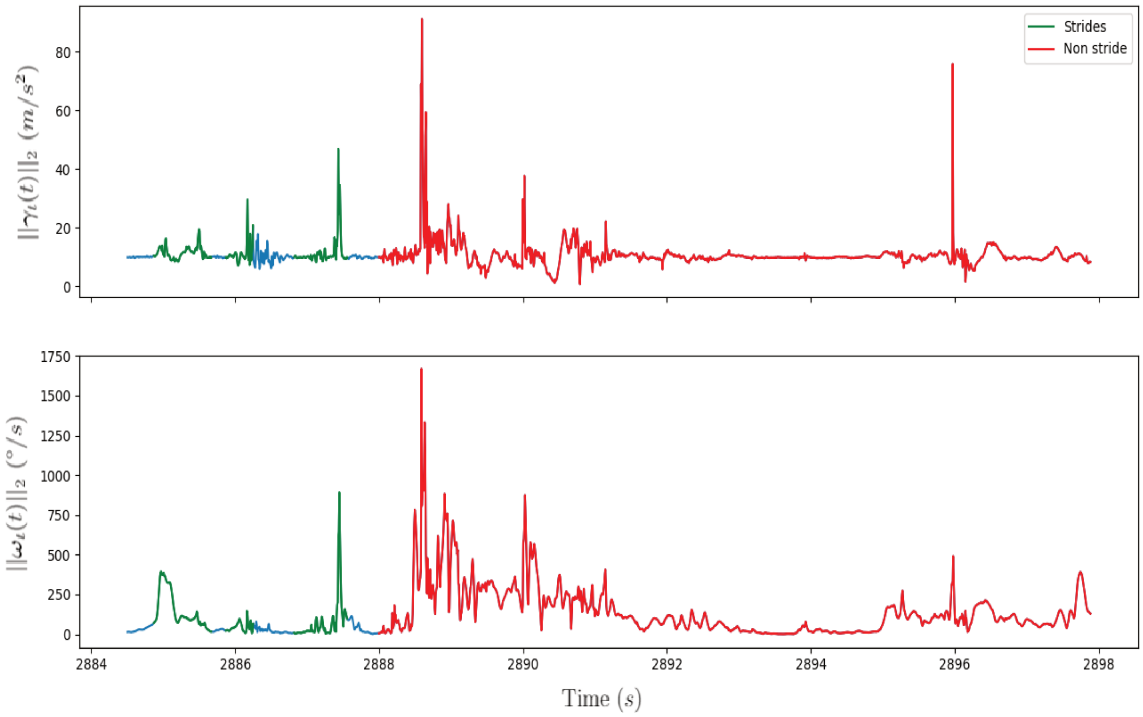


Figure 32: Example of three strides followed by hand manipulation.

to a stride and the other correspond to movements when the device is handle by hand. The signals norm present very similar patterns and it is difficult to

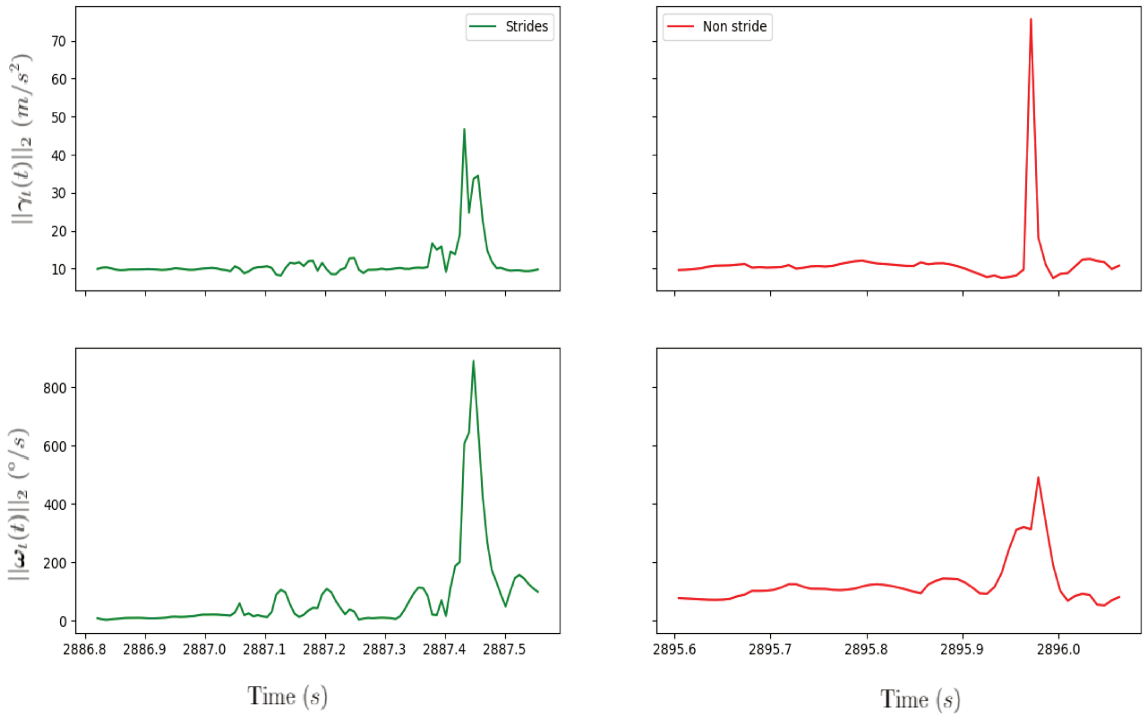


Figure 33: Example of two extracted intervals.

find deterministic criteria to distinguish these two. In addition, regarding the Figure 32, the norms show different forms between the three strides. Face to the variety of gait motion and due to the complexity of our application framework we adopted a machine learning approach.

The goal is to compute relevant features from the extracted intervals to build a function with Gradient Boosting Trees (GBT) algorithm (see Section II.2.2) able

to classify the considered interval: if it is a stride or not. One could only compute features from the norms of the signals but it misses a lot of relevant information contained in each axis. That is why during the features engineering process the main idea is to project the inertial data in a reference frame with the same definition for each interval.

2 Features engineering

The database "DB3" (see Section II.4.1) contains a dozen of co-workers recordings of various ages and heights practicing several activities under video control. At the ankle, the device has been carefully fixed with the default placement, ensuring that the system was placed as defined in Figure ??. With synchronisation between the video and ActiMyo data time scales, the performed strides have been annotated among "atypical steps", "walking", "running", "upstairs" and "downstairs".

Activity	Atypical steps	Walking	Running	Upstairs	Downstairs
Label y^a	1	2	3	4	5

Table 3: Activity labels definition.

From these records, the gyrometer data of $\bar{\mathcal{I}}_{DB3}$ (extracted intervals that are true strides, see Section II.4.2) were used to define a reference pattern $\mathbf{\Omega}^{y^a}$ in $\mathbb{R}^{3\times n}$ for each activity. In Section III.2.1, the goal is to fit the gyroscope data of a new candidate stride interval to each reference pattern by allowing a rotation. Then, if the considered extracted interval is a stride, the swing phase (see Figure 14) is visible on the Y axis of the gyrometer data. The aim of Section III.2.2 is to model the forward swing by computing a 1D reference pattern for the five considered activities presented in Table 3. The variables computed along the previous stages are combined with features based on functional data analysis (Section III.2.3).

2.1 Sensors alignment

Let $\mathbf{\Omega}^{y^s} = (\mathbf{\Omega}_1^{y^s}, \ldots, \mathbf{\Omega}_n^{y^s})$ with $\mathbf{\Omega}_i^{y^s} \in \mathbb{R}^3$ the 3D reference pattern data of size n on the three gyrometer axes and $(\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_N)$ the gyroscope data of a new candidate stride interval. The first step is to bring the observed data to the same number of samples as the reference pattern by a cubic spline interpolation [24] on each axis. Let $\hat{\boldsymbol{\omega}} = (\hat{\boldsymbol{\omega}}_1, \ldots, \hat{\boldsymbol{\omega}}_n)$ be the vector of the interpolated data. We want to compute the rotation matrix $\boldsymbol{R}^*$ that minimizes $\sum_{i=1}^n w_i ||\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \mathbf{\Omega}_i^{y^s}||_2^2$ with $\boldsymbol{R}$ a rotation matrix in $\mathbb{R}^{3\times 3}$. The coefficients $w_1, \ldots, w_n$ are the weights given to the samples of the pattern ($\sum_{i=1}^n w_i = 1$). In practice we set empirically higher weight values to the samples in the middle of the stride to avoid side effects. Indeed, data in the end can be noisy by the contact of the foot with the ground. Moreover, the foot movement on the ground during the stance phase is less specific to the activity than during the swing phase. Hence, the weights are set close to zero when the foot is on the ground and increase linearly over time until the foot is in the air before decreasing linearly with time when the foot approaches the ground.

Property 1. — *Given $\boldsymbol{\omega}_i \in \mathbb{R}^3$ and $\boldsymbol{\Omega}_i^{y^s} \in \mathbb{R}^3$ for all i in $[\![1,n]\!]$. The solution of the problem*

$$\boldsymbol{R}^* = \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I}_3, \det(\boldsymbol{R})=1}{\operatorname{argmin}} \sum_{i=1}^{n} w_i ||\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \boldsymbol{\Omega}_i^{y^s}||_2^2,$$

is given by

$$\boldsymbol{R}^* = \boldsymbol{V}\boldsymbol{U}^T,$$

where $\boldsymbol{I}_3$ is the identity matrix in $\mathbb{R}^{3\times3}$, $\boldsymbol{V}$ and $\boldsymbol{U}$ are the unitary matrices of the decomposition into singular values of $\hat{\boldsymbol{\omega}}\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T$, and $\boldsymbol{W} = diag(w_1,\ldots,w_n)$ with $\sum_{i=1}^{n} w_i = 1$.

Proof 1. —

$$\begin{aligned}
||\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \boldsymbol{\Omega}_i^{y^a}||_2^2 &= (\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \boldsymbol{\Omega}_i^{y^a})^T(\boldsymbol{R}\hat{\boldsymbol{\omega}}_i\boldsymbol{\Omega}_i^{y^a}) \\
&= (\hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T - (\boldsymbol{\Omega}_i^{y^a})^T)(\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \boldsymbol{\Omega}_i^{y^a}) \\
&= \hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a} - (\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i + (\boldsymbol{\Omega}_i^{y^a})^2 \\
&= (\hat{\boldsymbol{\omega}}_i)^2 - \hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a} - (\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i + (\boldsymbol{\Omega}_i^{y^a})^2
\end{aligned}$$

Therefore,

$$\underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmin}} \sum_{i=1}^{n} w_i ||\boldsymbol{R}\hat{\boldsymbol{\omega}}_i - \boldsymbol{\Omega}_i^{y^a}||_2^2 = \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmin}} -\sum_{i=1}^{n} w_i \quad (\quad \hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a} \quad - \quad (\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i \ ).$$

In addition for all i in $[\![1,n]\!]$ we have $\hat{\boldsymbol{\omega}}_i^T$ in $\mathbb{R}^{1\times3}$, $\boldsymbol{R}^T$ in $\mathbb{R}^{3\times3}$ and $\boldsymbol{\Omega}_i^{y^a}$ in $\mathbb{R}^{3\times1}$. Thus $\hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a}$ is a scalar and $\hat{\boldsymbol{\omega}}_i^T\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a} = (\hat{\boldsymbol{\omega}}_i^t\boldsymbol{R}^T\boldsymbol{\Omega}_i^{y^a})^T = (\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i$.
We can write,

$$\begin{aligned}
\boldsymbol{R}^* &= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmin}} -2\sum_{i=1}^{n} w_i(\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} \sum_{i=1}^{n} w_i(\boldsymbol{\Omega}_i^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_i \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} w_1(\boldsymbol{\Omega}_1^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_1 + \ldots + w_n(\boldsymbol{\Omega}_n^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_n \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} Tr(\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}})
\end{aligned}$$

Indeed,

$$\begin{aligned}
\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T\boldsymbol{R}X &= \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_n \end{pmatrix} \begin{pmatrix} (\boldsymbol{\Omega}_1^{y^a})^T \\ \vdots \\ (\boldsymbol{\Omega}_n^{y^a})^T \end{pmatrix} \boldsymbol{R}\left(\hat{\boldsymbol{\omega}}_1,\ldots,\hat{\boldsymbol{\omega}}_n\right) \\
&= \begin{pmatrix} w_1(\boldsymbol{\Omega}_1^{y^a})^T \\ \vdots \\ w_n(\boldsymbol{\Omega}_n^{y^a})^T \end{pmatrix} \left(\boldsymbol{R}\hat{\boldsymbol{\omega}}_1,\ldots,\boldsymbol{R}\hat{\boldsymbol{\omega}}_n\right) \\
&= \begin{pmatrix} w_1(\boldsymbol{\Omega}_1^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_1 & & \\ & \ddots & \\ & & w_n(\boldsymbol{\Omega}_n^{y^a})^T\boldsymbol{R}\hat{\boldsymbol{\omega}}_n \end{pmatrix}
\end{aligned}$$

Hence $\sum_{i=1}^{n} w_i (\boldsymbol{\Omega}_i^{y^a})^T \boldsymbol{R}\hat{\boldsymbol{\omega}}_i = Tr((\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T)(\boldsymbol{R}\hat{\boldsymbol{\omega}})) = Tr((\boldsymbol{R}\hat{\boldsymbol{\omega}})(\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T)$.
Let the decomposition into singular values of $\hat{\boldsymbol{\omega}}\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T : \hat{\boldsymbol{\omega}}\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, *therefore we have*

$$\begin{aligned}
\boldsymbol{R}^* &= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} Tr(\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T \boldsymbol{R}\hat{\boldsymbol{\omega}}) \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} Tr(\boldsymbol{R}\hat{\boldsymbol{\omega}}\boldsymbol{W}(\boldsymbol{\Omega}^{y^a})^T) \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} Tr(\boldsymbol{R}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T) \\
&= \underset{\boldsymbol{R}\boldsymbol{R}^T=\boldsymbol{I_3},\ det(\boldsymbol{R})=1}{\operatorname{argmax}} Tr(\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U})
\end{aligned}$$

However the matrices $\boldsymbol{V}^T$, $\boldsymbol{R}$ *and* $\boldsymbol{U}$ *are orthogonal matrices. If we write* $\boldsymbol{O} = \boldsymbol{V}^T\boldsymbol{R}U$ *we can deduce that the matrice* $\boldsymbol{O}$ *is orthogonal too*:

$$\begin{aligned}
(\boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U})^{-1} &= \boldsymbol{U}^{-1}\boldsymbol{R}^{-1}(\boldsymbol{V}^T)^{-1} \\
&= \boldsymbol{U}^T\boldsymbol{R}^T\boldsymbol{V} \\
&= (\boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U})^T
\end{aligned}$$

We write $\boldsymbol{O} = (\boldsymbol{O}_1, \boldsymbol{O}_2, \boldsymbol{O}_3)$. *As* $\boldsymbol{O}$ *is orthogonal, we have for all* i *in* $[\![1,3]\!]$ $\boldsymbol{O}_i^T\boldsymbol{O}_i = 1$ *which is equivalent to* $\sum_{j=1}^{3} O_{ij}^2 = 1$. *Hence for all* i *and* j *in* $[\![1,3]\!]$ $|O_{ij}| \leq 1$, *in particular* $|O_{ii}| \leq 1$.
By writing $\boldsymbol{\Sigma} = diag(\sigma_1, \sigma_2, \sigma_3)$ *we have,*

$$\begin{aligned}
Tr(\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U}) &= Tr(\boldsymbol{\Sigma}\boldsymbol{O}) \\
&= \sum_{i=1}^{3} \sigma_i O_{ii} \\
&\leq \sum_{i=1}^{3} \sigma_i
\end{aligned}$$

Thus the maximum of $Tr(\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U})$ *is reached for all* i *in* $[\![1,3]\!]$ $O_{ii} = 1$. *As* $\boldsymbol{O}$ *is orthogonal, it is equivalent to consider* $\boldsymbol{0} = \boldsymbol{I}_3$.

$$\begin{aligned}
\boldsymbol{O} = \boldsymbol{I}_3 &\Leftrightarrow \boldsymbol{V}^T\boldsymbol{R}\boldsymbol{U} = \boldsymbol{I}_3 \\
&\Leftrightarrow \boldsymbol{R} = (\boldsymbol{V}^T)^{-1}\boldsymbol{U}^{-1} \\
&\Leftrightarrow \boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T
\end{aligned}$$

Thus the solution is given by:

$$\boxed{\boldsymbol{R} = \boldsymbol{V}\boldsymbol{U}^T}$$

We compute this rotation matrix for the five 3D reference patterns to be sure that at least one good alignment has been computed. In the following of this Section 4.2.2, we assume that the rotated data of the extracted strides are in the default reference frame defined in Figure ??.

2.2 Swing modelling

We saw (see Figure 29) that the cycle of a stride is divided into two phases: swing and stance. During the swing phase, moving the foot forward creates a distinctive pattern in the Y axis (with default device placement assumption) of the angular

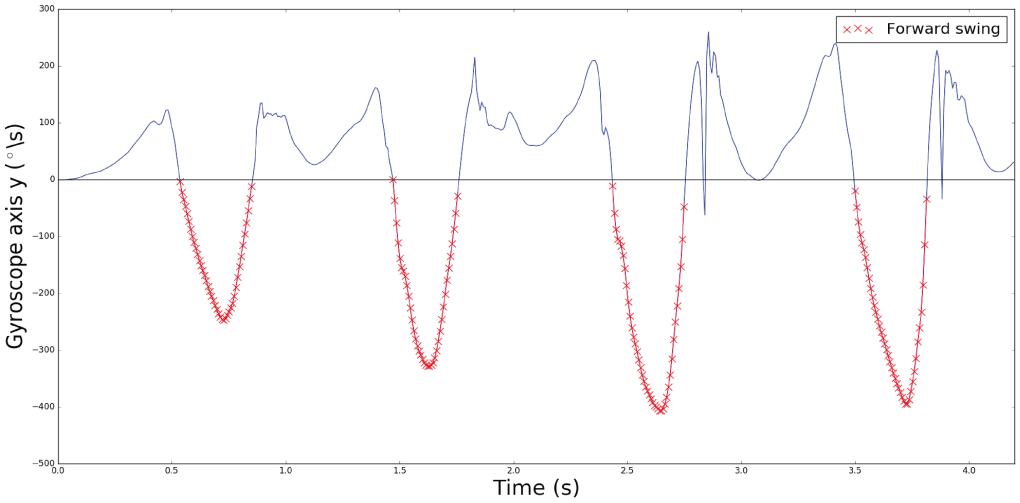


Figure 34: Example of forward swing patterns during walking with default device placement.

velocity (Figure 34). In this thesis we call forward swing the sequence where the values remain negative. The aim of this section is to model the forward swing. We want to compute a 1D reference pattern that defines the gyrometer data on the Y axis of the forward swing for the five considered activities presented in Table 3: "atypical strides", "walking", "running", "upstairs" and "downstairs".

Let N^{y^a} the number of strides of each activity y^a from the database "DB3" (Section II.4.1). From these records with default device placement, we note $h_l^{y^a}$ the observed function associated to the gyroscope data on the Y axis of the l^{th} forward swing among N^{y^a} forward swings for the activity y^a. We note h^{y^a} the unknown function associated to the reference pattern of the activity y^a, defined on the interval $[0, 1]$ taking values $\mathbb{R}$. We assume that the 1D reference pattern we want to compute can be approached with an error $\epsilon_l^{y^a}$ by all the $h_l^{y^a}$ functions of the same activity by multiplying them with a real coefficient $a_l^{y^a}$:

$$h^{y^a} = a_l^{y^a} \times h_l^{y^a} + \epsilon_l^{y^a}.$$

By observing the swing patterns of several activities from different wearers, we noticed that they provide the same shape for one given activity but with different ranges of values. Indeed a child who is running, swings forward his foot with less angular velocity than an adult. We assume that the functions belong to a function space E with its norm $||.||$. The observations are the functions $h_l^{y^a}$ and we want to estimate the quantities $\hat{h}^{y^a}$ and $\hat{a}_l^{y^a}$ which are computed by least squares minimization under constraints $(\mathcal{P})$ for all y^a in $\{1, 2, 3, 4, 5\}$ and for all l in $[\![1, N^{y^a}]\!]$:

$$\hat{h}^{y^a} = \underset{h^{y^a} \in E, ||h^{y^a}||=1}{\operatorname{argmin}} \sum_l ||\hat{a}_l^{y^a} h_l^{y^a} - h^{y^a}||, \tag{8}$$

$$\hat{a}_l^{y^a} = \underset{a_l^{y^a} \in \mathbb{R}_+^*}{\operatorname{argmin}} ||a_l^{y^a} h_l^{y^a} - \hat{h}^{y^a}||. \tag{9}$$

To solve the problem $(\mathcal{P})$, we consider an orthonormal basis $(e_1, \ldots, e_p)$ for the norm $||.||$. In practice, we use Lagrange polynomials [11] but other basis can be selected such as Fourier basis. We note:

$$\hat{h}^{y^a} = \sum_{u=1}^{p} \hat{\beta}_u e_u, \tag{10}$$

$$h_l^{y^a} = \sum_{u=1}^{p} \alpha_{lu} e_u. \tag{11}$$

Property 2. — *Given* $\mathbf{\Lambda}_l = (\alpha_{l1}, \ldots, \alpha_{lp})^T$ *and the symetric matrix* $\boldsymbol{A}$ *defined by* $A_{ij} = \sum_l \frac{\alpha_{l,i}\alpha_{l,j}}{||\Lambda_l||_2^2}$*, the solution of the problem* $(\mathcal{P})$ *is given by*:

$$\hat{h}^{y^a} = \pm \sum_{u=1}^{p} \theta_{pu} e_u, \tag{12}$$

and

$$\hat{a}_l^{y^a} = \frac{\sum_{u=1}^{p} \alpha_{lu}\hat{\beta}_u}{\sum_{u=1}^{p} \alpha_{lu}^2}, \tag{13}$$

where $\boldsymbol{\theta}_p$ *is the eigenvector of* $\boldsymbol{A}$ *associated to its greater eigenvalue.*

Proof 2. — *The definition of* $\hat{a}_l^{y^a}$ *is given by the minimization of the expression* $||a_l^{y^a} h_l^{y^a} - \hat{h}^{y^a}||$ *that is equivalent to minimize* $||a_l^{y^a} h_l^{y^a} - \hat{h}^{y^a}||^2$*. Considering the notations of Equation 10 and Equation 11 we have*:

$$\begin{aligned} ||a_l^{y^a} h_l^{y^a} - \hat{h}^{y^a}||^2 &= \sum_{u=1}^{p} (a_l^{y^a}\alpha_{lu} - \hat{\beta}_u)^2 \\ &= \sum_{u=1}^{p} (a_l^{y^a})^2(\alpha_{lu})^2 + (\hat{\beta}_u)^2 - 2a_l^{y^a}\alpha_{lu}\hat{\beta}_u \\ &= (a_l^{y^a})^2 \sum_{u=1}^{p} (\alpha_{lu})^2 - 2a_l^{y^a} \sum_{u=1}^{p} \alpha_{lu}\hat{\beta}_u + \sum_{u=1}^{p} (\hat{\beta}_u)^2. \end{aligned}$$

By derivating with respect to $a_l^{y^a}$ *we have*:

$$2a_l^{y^a} \sum_{u=1}^{p} (\alpha_{lu})^2 - 2\sum_{u=1}^{p} \alpha_{lu}\hat{\beta}_u \;>\; 0,$$

that is equivalent to:

$$a_l^{y^a} \;>\; \frac{\sum_{u=1}^{p} \alpha_{lu}\hat{\beta}_u}{\sum_{u=1}^{p} (\alpha_{lu})^2}.$$

Thus the solution of Equation 9 is given by:

$$\boxed{\hat{a}_l^{y^a} = \frac{\sum_{u=1}^{p} \alpha_{lu}\hat{\beta}_u}{\sum_{u=1}^{p} \alpha_{lu}^2}.}$$

The estimator $\hat{a}_l^{y^a}$ *is well defined because its denominator* $\sum_{u=1}^{p} \alpha_{lu}^2$ *is equals zero if and only if for all* u *in* $[\![1, p]\!]$ *we have* $\alpha_{lu} = 0$*. By equivalence of the norms in finite dimension, such a function is the null function defined in* $[0, 1]$ *(if we consider in particular the norm* $||.||_2$*). However the studied functions have strictly negative values (negative angular velocity values on the* Y *axis between two zero crossings).*
The solution of Equation 8 is the function $\hat{h}^{y^a}$ *that minimizes the expression* $||\sum_{l=1}^{N^{y^a}} ||\hat{a}_l^{y^a} h_l^{y^a} - h^{y^a}||^2$*. Considering the notations of Equation 10 and Equation 11*

we have:

$$\begin{aligned}
\sum_{l=1}^{N^{y^a}} ||\hat{a}_l^{y^a} h_l^{y^a} - h^{y^a}||^2 &= \sum_{l=1}^{N^{y^a}} \left((\hat{a}_l^{y^a})^2 \sum_{u=1}^{p} \alpha_{lu}^2 - \hat{a}_l^{y^a} \sum_{u=1}^{p} \alpha_{lu}\beta_u + \sum_{u=1}^{p} \beta_u^2 \right) \\
&= N^{y^a} \sum_{u=1}^{p} \beta_u^2 + \sum_{l=1}^{N^{y^a}} (\hat{a}_l^{y^a})^2 \sum_{u=1}^{p} \alpha_{lu}^2 - 2 \sum_{l=1}^{N^{y^a}} \hat{a}_l^{y^a} \sum_{u=1}^{p} \alpha_{lu}\beta_u \\
&= N^{y^a} \sum_{u=1}^{p} \beta_u^2 + \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\left(\sum_{u=1}^{p} \alpha_{lu}^2\right)^2} \sum_{u=1}^{p} \alpha_{lu}^2 \\
&\quad -2 \sum_{l=1}^{N^{y^a}} \frac{\sum_{u=1}^{p} \alpha_{lu}\beta_u}{\sum_{u=1}^{p} \alpha_{lu}^2} \sum_{u=1}^{p} \alpha_{lu}\beta_u \\
&= N^{y^a} \sum_{u=1}^{p} \beta_u^2 + \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\left(\sum_{u=1}^{p} \alpha_{lu}^2\right)^2} - 2 \sum_{l=1}^{N^{y^a}} \frac{\sum_{u=1}^{p} \alpha_{lu}\beta_u}{\sum_{u=1}^{p} \alpha_{lu}^2} \\
&= N^{y^a} \sum_{u=1}^{p} \beta_u^2 - \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\sum_{u=1}^{p} \alpha_{lu}^2}.
\end{aligned}$$

Thus we want to minimize the expression $N^{y^a} \sum_{u=1}^{p} \beta_u^2 - \sum_{l=1}^{N^{y^a}} \frac{\sum_{u=1}^{p} \alpha_{lu}\beta_u}{\left(\sum_{u=1}^{p} \alpha_{lu}^2\right)^2}$ *under the constraint* $||h^{y^a}|| = 1$ *that is equivalent to* $\sum_{u=1}^{p} \beta_u^2 = 1$*. With the notation in the canonical basis of* $\mathbb{R}^p$*,* $\boldsymbol{\Lambda}_l = (\alpha_{l1}, \ldots, \alpha_{lp})^T$*,* $\boldsymbol{X} = (\beta_1, \ldots, \beta_p)^T$ *and* $\hat{\boldsymbol{X}} = (\hat{\beta}_1, \ldots, \hat{\beta}_p)^T$ *we have:*

$$\begin{aligned}
\hat{\boldsymbol{X}} &= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmin}} N^{y^a} \sum_{u=1}^{p} \beta_u^2 - \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\sum_{u=1}^{p} \alpha_{lu}^2} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmin}} N^{y^a} - \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\sum_{u=1}^{p} \alpha_{lu}^2} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmax}} \sum_{l=1}^{N^{y^a}} \frac{\left(\sum_{u=1}^{p} \alpha_{lu}\beta_u\right)^2}{\sum_{u=1}^{p} \alpha_{lu}^2} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmax}} \sum_{l=1}^{N^{y^a}} \frac{< \boldsymbol{X}, \boldsymbol{\Lambda}_l >_2^2}{||\boldsymbol{\Lambda}_l||_2^2} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmax}} \sum_{l=1}^{N^{y^a}} \frac{\boldsymbol{X}^T \boldsymbol{\Lambda}_l \boldsymbol{\Lambda}_l^T \boldsymbol{X}}{||\boldsymbol{\Lambda}_l||_2^2} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmax}} \boldsymbol{X}^T \sum_{l=1}^{N^{y^a}} \frac{\boldsymbol{\Lambda}_l \boldsymbol{\Lambda}_l^T}{||\boldsymbol{\Lambda}_l||_2^2} \boldsymbol{X} \\
&= \underset{||\boldsymbol{X}||_2^2=1}{\operatorname{argmax}} \boldsymbol{X}^T \boldsymbol{A} \boldsymbol{X}.
\end{aligned}$$

The matrix $\boldsymbol{A}$ *in* $\mathbb{R}^{p \times p}$ *is defined for all* i *and for all* j *in* $[\![1, p]\!]$ *by* $A_{ij} = \sum_{l=1}^{N^{y^a}} \frac{\alpha_{li}\alpha_{lj}}{||\boldsymbol{\Lambda}_l||_2^2}$*. It is a symetric matrix with values in* $\mathbb{R}$*, thus its eigenvectors are defined in* $\mathbb{R}$*. We note* $\lambda_1 \leq \ldots \leq \lambda_p$ *its eigenvalues. As* $\boldsymbol{A}$ *is symetric, it is diagonalizable in the orthonormal basis of its eigenvectors* $(\boldsymbol{w}_1, \ldots, \boldsymbol{w_p})$ *associated to* $\lambda_1, \ldots, \lambda_p$*. The*

expression of $\boldsymbol{X}$ in the basis $(\boldsymbol{w}_1, \ldots, \boldsymbol{w_p})$ is given by $\boldsymbol{X} = \sum_{u=1}^{p} \beta_u^w \boldsymbol{w}_u$. Then we have:

$$\begin{aligned} \boldsymbol{X}^T \boldsymbol{A} \boldsymbol{X} &= \sum_{u=1}^{p} \lambda_u (\beta_u^w)^2 \\ &\leq \sum_{u=1}^{p} \lambda_p (\beta_u^w)^2 \\ &\leq \lambda_p \sum_{u=1}^{p} (\beta_u^w)^2 \\ &\leq \lambda_p. \end{aligned}$$

Indeed, $||\boldsymbol{X}||_2^2 = 1$ is equivalent to $\sum_{u=1}^{p} (\beta_u^w)^2 = 1$. But if we take $\boldsymbol{X} = \pm \boldsymbol{w}_p$, we have $||\boldsymbol{X}||_2^2 = 1$ and

$$\begin{aligned} \boldsymbol{X}^T \boldsymbol{A} \boldsymbol{X} &= \boldsymbol{w}_p^T \boldsymbol{A} \boldsymbol{w}_p \\ &= \alpha_p \boldsymbol{w}_p^T \boldsymbol{w}_p \\ &= \alpha_p. \end{aligned}$$

We can deduce that $\boldsymbol{w}_p$ is solution and thus:

$$\boxed{\hat{h}^{y^a} = \pm \sum_{u=1}^{p} \boldsymbol{w}_{pu} e_u.}$$

We chose $\hat{h}^{y^a}$ among these two solutions in order to have positive values for $\hat{a}_l^{y^a}$, namely $\frac{\sum_{u=1}^{p} \alpha_{lu} \hat{\beta}_u}{\sum_{u=1}^{p} \alpha_{lu}^2} > 0$.

From the above, we want to express the functions defined in $[0,1]$ taking values in $\mathbb{R}$ in a finite orthonormal basis. Let assume that the functions h^{y^a} and $h_l^{y^a}$ belong to the space $\mathbb{R}[X]$, the basis composed by the polynomials of Lagrange fulfil the criteria and the expression of a function is easy. Let assume that the space E of the studied functions h^{y^a} and $h_l^{y^a}$ are the set of polynomials defined on $[0,1]$, with real coefficients and with degrees smaller or equal to d. Let the regular subdivision of $[0,1]$: $0 = \frac{0}{d} = x_0 < \frac{1}{d} = x_1 < \ldots < x_d = \frac{d}{d} = 1$. The Lagrange polynomials associated to these points are the polynomials defined for all t in $[0,1]$ and for all u in $[\![0,d]\!]$, $l_u(t) = \prod_{z=0, d \neq u}^{d} \frac{t - x_z}{x_u - x_z}$. They constitute a basis of the space E. In addition, for the scalar product $< P, Q >= \sum_{i=0}^{d} P(x_u) Q(x_u)$, this basis is orthonormal. In this basis we write the functions $h_l^{y^a}$ for all t in $[0,1]$ by the following:

$$h_l^{y^a}(t) = \sum_{u=0}^{d} h_l^{y^a}(x_u) l_u(t).$$

However we saw that the gyrometer sensors record data with 130 Hz frequency. We note $t_{l,1}^{y^a} < t_{l,2}^{y^a} < \ldots < t_{l,n}^{y^a}$ the n recording times during the l^{th} forward swing of the database "DB3" (see Section II.4.1) performed with the activity y^a. As in this database the ActiMyo device has been set carefully with the default placement, the forward swing values associated with the recordings time are given by the Y axis of $\boldsymbol{\omega}_t$. We note them $\omega_{l,1}^{y^a}, \ldots, \omega_{l,n}^{y^a}$. We consider the n couples $(0, \omega_{l,1}^{y^a}), (\frac{1}{n}, \omega_{l,2}^{y^a}), \ldots, (1, \omega_{l,n}^{y^a})$. this subdivision does not necessarily correspond to the one defined by the x_u. Thus, we compute a cubic splines interpolation ([24]) on the couples $(0, \omega_{l,1}^{y^a}), \ldots, (1, \omega_{l,n}^{y^a})$ that estimates the values of $h_l^{y^a}(x_u)$. The quality of

Lagrange polynomials interpolation depends on the d value (the more d is large better is the interpolation). However considering a d value too large would be too much time consuming for the computation of the eigenvectors of $\boldsymbol{A}$ (see Property 2) with $\mathcal{O}(d^3)$ dimension complexity. In practice, $d = 99$ leads to sufficient precision. For illustration we represent in Figure 35 the forward swing for a walking stride with the n recordings times (resized in $[0, 1]$) and its interpolation by Lagrange polynomials in dimension 100.

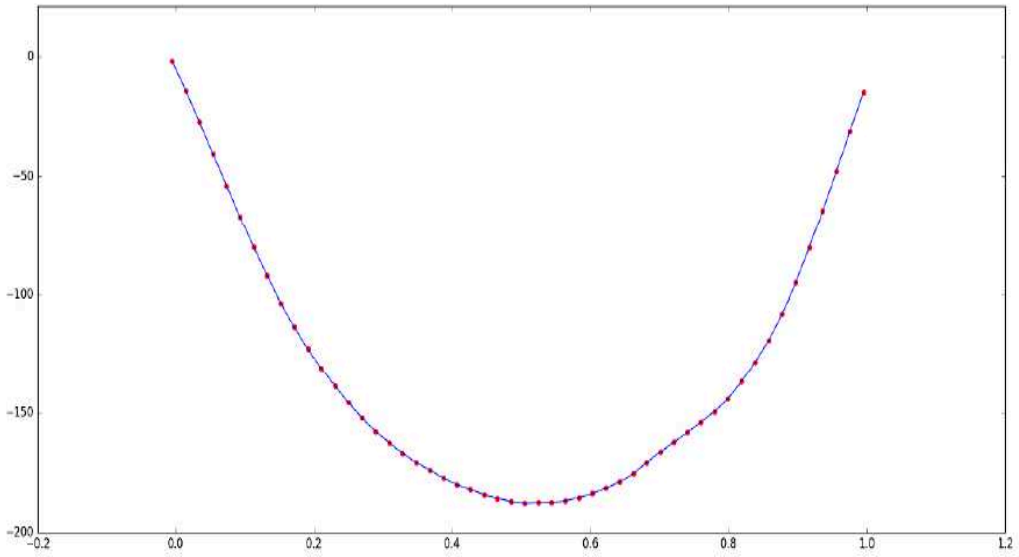


Figure 35: Example of a walking forward swing with Lagrange interpolation (blue) on n recording points $\omega_{l,i}^{y^a}$ (red).

The 5 functions $\hat{h}^{y^a}$ are computed once for all and the procedure is sum up in the following Algorithm 10.

Algorithm 10: Computation of the 1D forward swing pattern for activity y^a.

---

1 Compute the matrix $\boldsymbol{A}$ such as for all i, j in $[\![0, 99]\!]$:

$$A_{ij} = \sum_{l=1}^{N^{y^a}} \frac{h_l^{y^a}(x_i) h_l^{y^a}(x_j)}{\sum_{u=0}^{d} \cdot \left(h_l^{y^a}(x_u)\right)^2}$$

2 Compute the eigenvalues (in $\mathbb{R}$) of $\boldsymbol{A}$.

3 Consider the greatest eigenvalue associated with the eigenvector $\boldsymbol{\theta}_d \in \mathbb{R}^{d+1}$.

4 Compute the 1D reference pattern for the activity y^a define in $[0, 1]$:

$$\hat{h}^{y^a}(t) = -|\sum_{u=0}^{99} \theta_{du} l_u(t)|.$$

For a an extracted interval, after sensors alignment with the method presented in Section III.2.1, if negative values are detected in the Y axis of gyrometer data between two zero crossings, we compute the function g defined in $[0, 1]$ with values in $\mathbb{R}$ in the Lagrange polynomials basis:

$$g(t) = \sum_{u=0}^{99} g(x_u) l_u(t).$$

Then for each class y^a we compute the multiplier coefficient a^{y^a} (Equation 13) and compare $a^{y^a} g(t)$ it to the 1D reference pattern $\hat{h}^{y^a}$. If the residuals $||a^{y^a} g(t) - \hat{h}^{y^a}||_2$ are small for one 1D reference pattern, it means that g probably corresponds to the forward swing of a stride of activity y^a. On the other hand, it the residuals are large for every 1D reference patterns, it means the extracted interval probably does not correspond to a stride.

2.3 Functional data analysis

From the norm and each axis of both acceleration and angular velocity, features were computed in the time and frequency domains: maximum, mean, standard deviation, root mean square, inter-quantile range, Fast Fourier Transform, 3^{rd} and 4^{th} order moments, auto-correlation and correlations between signals etc.

Following the strategy explained in this section for each element of our extracted intervals algorithm, 2695 features are computed. The overall procedure is described in a pseudo-code in Algorithm 11. At this stage, we want to build a binary classifier that decides if one extracted interval is a stride or not.

Algorithm 11: Features computation

```
Compute the frequency domain features from the inertial data norms.
Compute the rotations (Property 1).
foreach activity rotation do
    Compare the rotated angular velocity with the 3D reference pattern
    if negative values on y gyroscope axis then
        foreach 1D reference pattern do
            Compute the multiplying coefficient (Equation 13).
            Compare the resulting multiplied negative values with the 1D reference
             pattern.
        end
    end
end
```

3 Intervals classification with GBT

This section describes the performance of the GBT classifier on the extracted intervals from the database "DB3" (Section III.3.1) and the performance of the overall stride detection algorithm during experimental tests in Motion Capture environment "DB4" (Section III.3.2). As a reminder, the descriptions of "DB3" and "DB4" are given in Section II.4.1.

3.1 Cross-validation performance

A dozen of co-workers of various ages and heights were filmed practicing several activities while wearing the system at the ankle and the wrist at Sysnav company under video control, composing the database of recordings "DB3". From this database, the extraction algorithm built the family $\hat{\mathcal{I}}_{DB3}$ with 6213 intervals that do not correspond to strides (label -1) and 5964 stride intervals divided into 5

different activities (label 1): "atypical step" that includes small step, side step, backward walking etc., "normal walking", "running", "upstairs" and "downstairs". We computed the features following the Algorithm 11 for each element in $\hat{\mathcal{I}}_{DB3}$ and then we launched the 10-fold cross-validation method testing several set of GBT parameters. We focused the tuning procedure for important parameters that can lead to overfitting such as the number of trees, the learning rate, the maximum depth of the trees and subsampling proportion. The cross-validation results of the best parameters are presented in the following confusion matrix (Table 4).

	Predicted -1	Predicted 1
Actual -1	6195	18
Actual 1	19	5945

Table 4: Confusion matrix.

The global error is about 0.3%. The distribution of the false negatives is presented in the Table 6.

	Atypical steps	Walking	Running	Upstairs	Downstairs
FN	$\frac{19}{1319}$	$\frac{0}{1473}$	$\frac{0}{1146}$	$\frac{0}{1024}$	$\frac{0}{1002}$

Table 5: False negatives distribution.

Our algorithm made no mistake on the "walking", "running", "upstairs" and "downstairs" strides. On "atypical" strides, it achieved a detection error rate of 1.5%. They are the most difficult kind of strides to detect because the associated 3D and 1D patterns are less specific. Consequently the sensors alignment procedure sometimes leads to an wrong orientation, namely different to the default one. Moreover a small step can be performed without a forward swing which makes the features engineering in Section III.2.2 useless.

The performance of the cross-validation depends on the interval extraction algorithm. Indeed, the extraction procedure may not select true strides in the "DB3" recordings that would have been wrongly classified by the GBT prediction function and then would have deteriorated the cross-validation score. In order to better analyze the performances of our algorithm, in the following we study the results of our stride detector in MOCAP environments "DB4".

3.2 False negative rate in MOCAP

A group of seven people wore the ActiMyo with infrared markers during MOCAP sessions in a 25 m^2, composing the databse of recordings "DB4". Several cameras were set in order to film whole scene. They broadcast infrared radiation that was reflected by the markers. This allowed the camera to record the position of the markers with sub-millimeter accuracy. We launched the stride detector algorithm (Algorithm 7), with stage 2 corresponding to the features engineering process described in Algorithm 11, on the recordings of "DB4". It provides a family of intervals detected as strides $\tilde{\mathcal{I}}_{DB4} = \{(\tilde{s}_1, \tilde{e}_1), \ldots, (\tilde{s}_i, \tilde{e}_i), \ldots, (\tilde{s}_n, \tilde{e}_n)\}$. With the MOCAP altitude (see Figure 28), we can note precisely when the foot is in contact with the ground during the stance phases in "DB4". We note $\mathcal{I}_{DB4} = \{(s_{1,1}, s_{1,2}, e_{1,1}, e_{1,2}), \ldots, (s_{n,1}, s_{n,2}, e_{n,1}, e_{n,2})\}$ the family of strides with $(s_{i,1}, s_{i,2})$

and $(e_{i,1}, e_{i,2})$ delimiting the first and last stance phase of the stride i. In this section we focus on the False Negative (FN) rate, namely the strides in $\mathcal{I}_{DB4}$ that are not detected by our stride detector. We consider that a stride i in $\mathcal{I}_{DB4}$ is not detected if there is no element j in $\tilde{\mathcal{I}}_{DB4}$ such as $\tilde{s}_j \in [s_{i,1}, s_{i,2}]$ and $\tilde{e}_j \in [e_{i,1}, e_{i,2}]$. The final results are presented in Table 6.

	Slow Walking Total - FN	**Medium Walking Total - FN**	**Fast Walking Total - FN**	**Small Steps Total - FN**	**Side Steps Total - FN**
Wearer 1	291 - 0%	279 - 0%	216 - 0%	88 - 0%	287 - 2.1%
Wearer 2	306 - 0%	261 - 0%	195 - 0%	67 - 0%	265 - 4.5%
Wearer 3	294 - 0%	219 - 0%	198 - 0%	107 - 4.7%	143 - 5.8%
Wearer 4	297 - 0%	267 - 0%	228 - 0%	145 - 2.1%	301 - 1.3%
Wearer 5	273 - 0%	249 - 0%	213 - 0%	65 - 30.8%	246 - 0%
Wearer 6	345 - 0%	339 - 0%	327 - 0%	90 - 5.2%	150 - 3.8%
Wearer 7	342 - 0%	246 - 0%	240 - 0%	48 - 10.4%	200 - 48.5%
Total	2148 - 0%	1860 - 0%	1617 - 0%	610 - 6.2%	1592 - 8.3%

Table 6: False negative rates in MOCAP sessions.

All walking strides were detected. We can see in Figure 36 that the MOCAP dataset presents diversified stride lengths and stride durations. This means that our detection achieved 100% accuracy for walking phases with various paces. Some of the walking strides may appear very small, but this corresponds to half turns. Indeed, the foot ends very close to the starting point of the stride. Our algorithm did not detect all atypical strides (small steps and side steps). It's consistent with the cross-validation results presented in Table 6 but the amount of missing strides is much greater than the one estimated in Table 6. This phenomenon comes from the fact that the extraction algorithm based on ground contact detection does not select several stride intervals that consequently are missed. For small steps, the ground contact could be too small to be marked by a peak in the acceleration and thus it is not considered by the interval extraction algorithm. For side steps, the instants when the foot is on the ground do not always satisfy the combined criteria on the acceleration norm and angular velocity norm. Thus the interval extraction algorithm do not propose correct start and end in this situation. These limitations will be described in the following section.

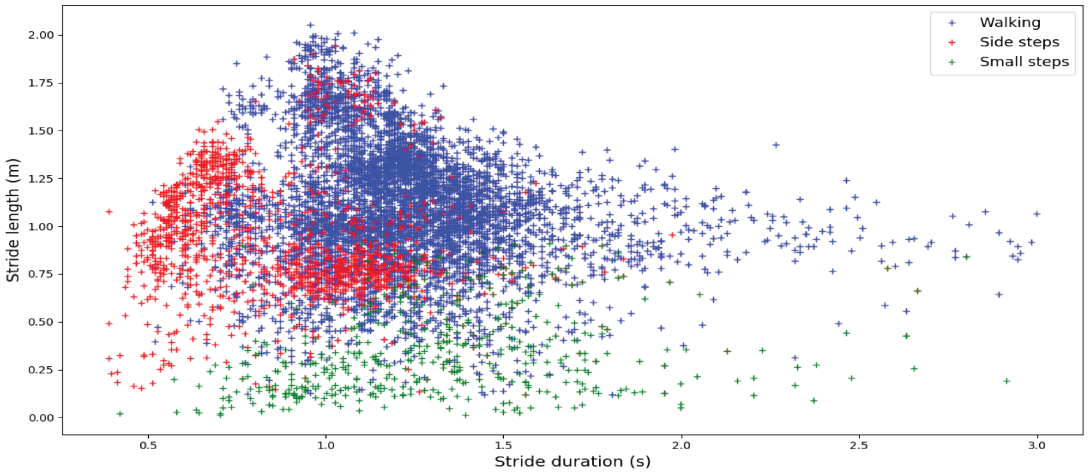


Figure 36: Stride length as a function of stride duration during MOCAP sessions.

4 Contributions

This section described an algorithm that allows to detect when a stride occcurs with its start and end times from inertial sensors worn at the ankle. This work is divided in four main stages:

- The selection of candidate intervals that may correspond to strides.
- The calculation of a rotation applied on the data in order to work in the same frame for all records. This step is built on fitting the gyroscope data with 3D geometric patterns.
- The extraction of the forward swing on the gyroscope axis y. These data are then fitted with 1D reference patterns.
- The binary classification of the intervals using the Gradient Boosting Trees algorithm with features computed along the previous steps.

For normal walking it has shown good results achievable with existing algorithms. But the method described in this section also has a good sensitivity for atypical strides such as small steps, side steps and backward walking contrary to most algorithms proposed in the literature. In Table 7 and Table 8 we present the atypical strides detection rates for the threshold based algorithm developed by Sysnav and for the machine learning (M.L.) approach described in this Section, both launched on the "DB4" recordings.

	Small steps		
	Total	**Threshold based detection**	**M.L. Section III**
Wearer 1	88	75%	100%
Wearer 2	67	68.7%	100%
Wearer 3	107	63.6%	95.3%
Wearer 4	145	55.2%	97.9%
Wearer 5	65	10.8%	69.2%
Wearer 6	90	40.7%	94.8%
Wearer 7	48	39.6%	89.6%
Total	610	52.9%	93.8%

Table 7: Detection rate for small steps.

	Side steps		
	Total	**Threshold based detection**	**M.L. Section III**
Wearer 1	287	50.7%	97.9%
Wearer 2	265	56.5%	85.5%
Wearer 3	143	52.9%	84.2%
Wearer 4	301	59.7%	98.7%
Wearer 5	246	36.1%	100%
Wearer 6	150	51.4%	96.2%
Wearer 7	200	44.4%	51.5%
Total	1592	50.6%	91.7%

Table 8: Detection rate for side steps.

For small steps and side steps our method allows to respectively increase the average detection rates from 52.9% to 93.8% and from 50.6% to 91.7%. The gain is significant but observing the mean of the detection rates hides several limitations.

Indeed for the wearer 5 the detection of small steps (69.2%) is not satisfying. For these strides, the foot contact with the ground was very light. The Figure 37 shows a sequence of non detected small steps. Each step are described by a variation in MOCAP altitude between 15 cm and 20 cm and are delimited by a stable altitude corresponding to periods where the foot is on the ground with a constant ankle altitude. We can see some variations (less than 0.1 g) during the strides

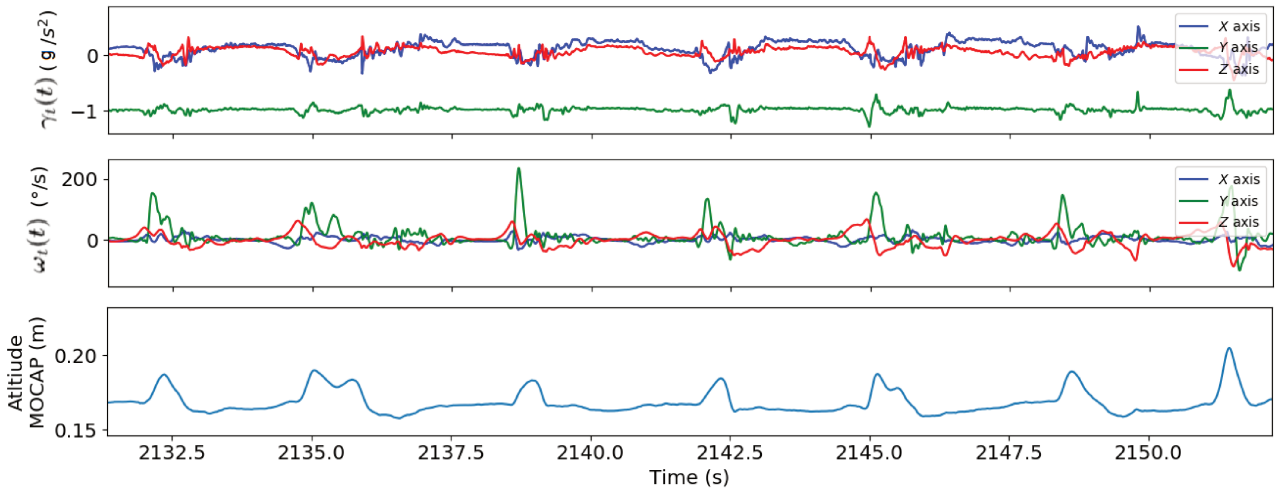


Figure 37: Inertial data in body frame with synchronized MOCAP altitude during missed small steps.

but here the values of the acceleration are particularly small. As a comparison, during normal walking (see Figure 38) the ground contact is characterized by a peak close to 2 g. Consequently the interval extraction algorithm presented in

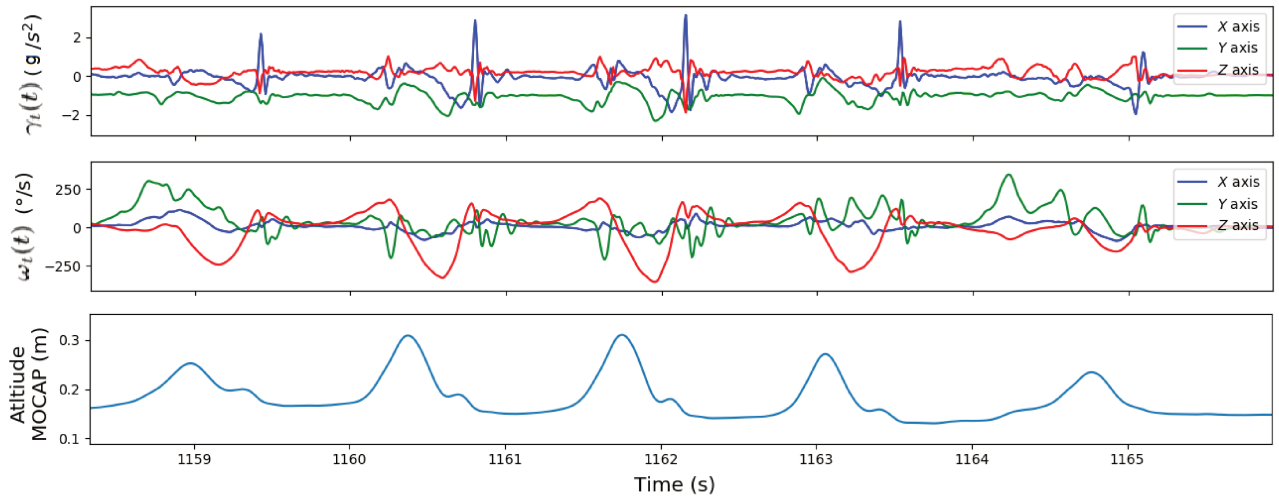


Figure 38: Inertial data in body frame with synchronized MOCAP altitude during walking.

this section did not detect it and did not propose any corresponding start and end. One could consider a smaller threshold value for the acceleration peak but it would lead to extract much more intervals, increase the statistical complexity of our binary classification problem and increase the overall computational cost of our stride detector. As a reminder, it is aimed to be launched on daily home

recordings and this overall algorithm is already expensive (about 12 hours for one day recording).

Moreover, the stride detector do not perform well for the side steps of wearer 7. The Figure 39 illustrates a sequence of non detected side steps. In this situation, the interval extraction algorithm do not select correct start and end. When the foot is on the ground the acceleration norm values are not close to one g and the angular velocity values are always large. The same phenomenon appeared for fast downstairs and it is difficult to capture some patterns compared to normal walking.

Facing these limitations that we did not manage to overtake with the described extraction algorithm, we developed a new extraction method which is not directly based on inertial data this time, but based on the computation of a pseudo-speed that is presented in the following Section IV.

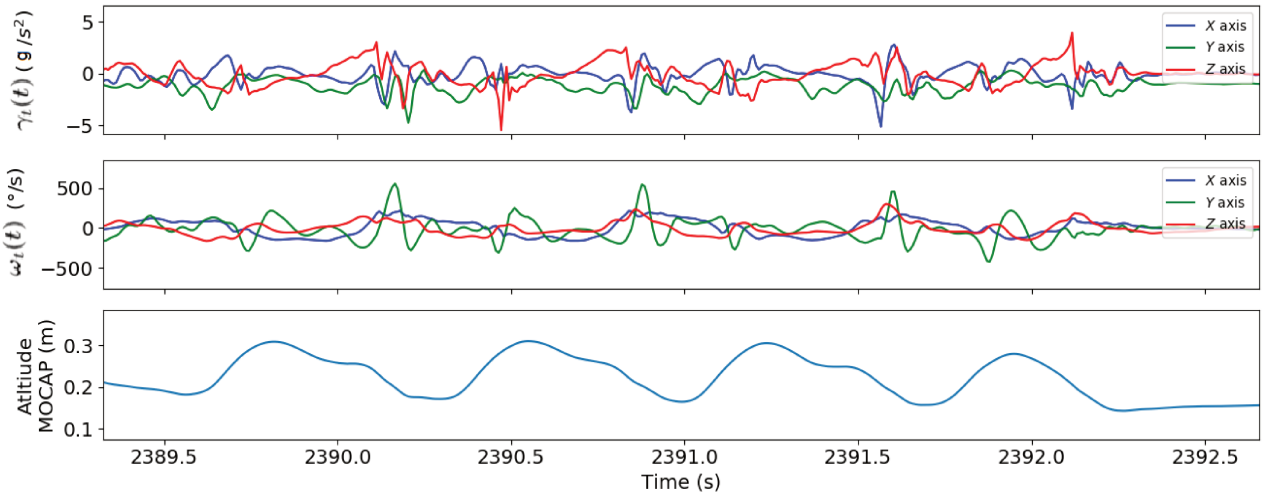


Figure 39: Inertial data in body frame with synchronized MOCAP altitude during missed side steps.

Part IV

Candidate stride interval extraction based on pseudo-speed

The ActiMyo device is a strapdown IMU (see Section I.2), thus the inertial data are recorded in its reference frame that we called body frame. Consequently, the rotation matrix that allows to express the acceleration (including the gravity) and the angular velocity in a world frame is not given directly. The general problem of reference coordinate frames was introduced in Figure 1 with the associated notations. One major consequence is that the speed of the device is difficult to estimate. However, we believe that this feature would be particularly relevant for detecting strides. Indeed when the foot is on the ground, namely during the stance phase, the speed of the ankle is small compared to the swing phase, whatever the activity. Even if during running for example, the ankle speed can reach more than 4 m/s when the foot is on the ground, it is still much greater during the swing. The idea is that studying the minima around the maxima of the ActiMyo speed would give the beginning and the end of the strides. But still, it requires to estimate the speed in the world frame.

In this chapter we propose a method removing the gravity from the acceleration and computing a terrestrial reference frame (Section IV.1). Then the integration of the projected linear velocity leads to the ankle pseudo-speed computation, noted $\hat{v}_W$ (Section IV.2). This is a rough estimation of the ActiMyo speed that is less accurate than the one computed in the extended Kalman filter (see Chapter V) but it allows to proposed relevant candidate stride intervals and shows fast computation.

Thanks to the labeled databases presented in Section II.4.1, we can compute a prediction function calculated by Gradient Boosting Trees algorithm (see Section II.2.2). It allows, with relevant features calculation (see Section IV.3), to select from the set of candidate intervals the true strides. In Section IV.4, we present its learning performances and the overall algorithm performances on recordings in motion capture environment. The work of this chapter has been published in Sensor journal [10] and in [8] IEEE conference paper.

1 Terrestrial Reference Frame Computation

The main idea of the terrestrial reference frame computation lies in the fact that in an inertial reference frame, the integration of γ is equal to the difference of the ankle speed (a few meters per second for a pedestrian) that is small compared to the integration of the gravity. At any time t in $[0, T]$, the device records the acceleration and angular velocity data (respectively $\boldsymbol{\gamma}_t(t)$ and $\boldsymbol{\omega}_t(t)$ in $\mathbb{R}^3$) in the body reference frame of the system. For all u in $[t, t+\Delta t]$ we compute the rotation matrix $\boldsymbol{R}_u^t$ between the body reference frame at time t and u by angular velocity integration. The matrix $\boldsymbol{R}_u^t$ is the solution of Equation (14):

$$\frac{d\boldsymbol{R}_u^t}{du} = -\boldsymbol{R}_u^t \; Skew(\boldsymbol{\omega}_u(u)), \tag{14}$$

with $\boldsymbol{R}_t^t = \boldsymbol{I}_3$ and the Skew operator defined for all vectors $\boldsymbol{n}$ in $\mathbb{R}^3$, $\boldsymbol{n} = (n_x, n_y, n_z)^T$:

$$Skew(\boldsymbol{n}) = \begin{pmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{pmatrix}.$$

Let $\boldsymbol{a}_u(u)$ be the acceleration of the ActiMyo system without gravity $\boldsymbol{g}_u$: $\boldsymbol{\gamma}_u(u) = \boldsymbol{a}_u(u) + \boldsymbol{g}_u$. Then, the mean of the recorded acceleration projected in the body reference frame at time t, on an interval Δt, is given by:

$$\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t \boldsymbol{\gamma}_u(u) du = \frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t (\boldsymbol{a}_u(u) + \boldsymbol{g}_u) du.$$

We assume that for a sufficiently small Δt, the integration of the angular velocity produces no error. As a result, $\boldsymbol{R}_u^t \boldsymbol{g}_u$ is a constant $\boldsymbol{g}_t$ on this interval. We have:

$$\begin{aligned}\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t \boldsymbol{\gamma}_u(u)du &= \frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t(\boldsymbol{a}_u(u)+\boldsymbol{g}_u)du \\ &= \frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t \boldsymbol{a}_u(u)du + \boldsymbol{g}_t.\end{aligned}$$

Let $\boldsymbol{v}_t(u)$ be the speed of the ankle in the reference frame at time t for all u in $[t, t+\Delta t]$. From the equation above we can write:

$$\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t \boldsymbol{\gamma}_u(u)du = \frac{\boldsymbol{v}_t(t+\Delta t)-\boldsymbol{v}_t(t)}{\Delta t} + \boldsymbol{g}_t.$$

For a sufficiently long duration of integration Δt, we assume that the speed difference of the ankle, between ΔT and t, divided by Δt is small relative to the gravity:

$$\frac{\boldsymbol{v}_t(t+\Delta t)-\boldsymbol{v}_t(t)}{\Delta t} \ll \boldsymbol{g}_t. \tag{15}$$

Thus, we can deduce the following equation:

$$\frac{1}{\Delta t}\int_t^{t+\Delta t} \boldsymbol{R}_u^t \boldsymbol{\gamma}_u(u)du \approx \boldsymbol{g}_t. \tag{16}$$

The assumption in Equation (15) is valid for large Δt. However, this approach requires that the mean of the acceleration in an inertial reference frame is computed. Due to the integration drift with time (caused by lack of precision and noise of inexpensive sensors in ActiMyo), if Δt is too large, we have no guarantee that $\boldsymbol{R}_u^t$ provides a rotation between the body reference frame at time u and t. In practice, we found a compromise by setting $\Delta t = 15$ s.

Thanks to Equation (16) we can identify the gravity in the body reference frame at time t : $\boldsymbol{g}_t$. If the angular velocity integration did not produce any error, for all $t > 0$, $\boldsymbol{g}_t$ would be equal to $\boldsymbol{R}_{t+dt}^t \boldsymbol{g}_{t+dt}$. In practice, we observe variations due to the integration drift. For all $t > 0$ we can correct this by computing the rotation matrix $\boldsymbol{R}_{g_t}$ that aligns $\boldsymbol{g}_t$ over time. We introduce the vector $\boldsymbol{a}$ as follows:

$$\boldsymbol{a} = \lim_{dt\to 0} \frac{\boldsymbol{g}_t/||\boldsymbol{g}_t|| \times \boldsymbol{R}_{t+dt}^t \boldsymbol{g}_{t+dt}/||\boldsymbol{R}_{t+dt}^t \boldsymbol{g}_{t+dt}||}{dt}.$$

Then the rotation matrix $\boldsymbol{R}_{g_t}$ is the solution of the following equation:

$$\frac{d\boldsymbol{R}_{g_t}}{dt} = -\boldsymbol{R}_{g_t}\, Skew(\boldsymbol{a}). \tag{17}$$

As a result, we can project the inertial data to have $\boldsymbol{g}$ constant and equal to $\boldsymbol{g}_{t=0}$. Finally, we define and project the data into a terrestrial reference frame W by considering the vector $-\frac{\boldsymbol{g}_0}{||\boldsymbol{g}_0||}$ as the new Z_W axis and arbitrarily choosing X_W and Y_W axes in order to build an orthonormal basis. The overall procedure is described in the pseudo-code Algorithm 12.

We now have access to the acceleration of the ankle for all t by removing the gravity ($\approx$ 9.81 m/s) from the Z_W axis:

$$\hat{\boldsymbol{a}}_W(t) = \boldsymbol{\gamma}_W(t) - \begin{pmatrix}0\\0\\9.81\end{pmatrix}. \tag{18}$$

Algorithm 12: Terrestrial reference frame computation with gravity identification.

Input : Recording of the system worn at the ankle
Output: Inertial data projected in a terrestrial reference frame with gravity identification

foreach $t \in [0, T]$ **do**
  Computation of $\boldsymbol{R}_u^t$ by gyrometer integration during Δt.
  Integration of $\boldsymbol{R}_u^t \boldsymbol{\gamma}_u(u)$ during Δt that we assume to be close to gravity.
  Alignment of the computed gravity with the previous ones.
end
Alignment of the gravity with the Z_W axis of the final terrestrial reference frame and projection of the inertial data.

The advantage of this attitude filter is the efficiency of its computation. This characteristic is necessary as we use $\hat{\boldsymbol{a}}_W$ to compute a pseudo-speed introduced in Section IV.2 that is one of the main features in our stride detector. It allows to extract a family of candidate intervals that may correspond to stride and provides, by integration, a pseudo-trajectory that appears to be a key variable for the Gradient Boosting Trees decision function. Indeed, this approach extracts intervals that are not strides when for example the wearer is moving the device in the hand during the required time to install or uninstall the system from the docking station.

2 Pseudo-speed Computation

In the previous section, we described the projection of the inertial data recorded by the device into a terrestrial reference frame. In this procedure, the gravity is removed from the acceleration, which can be integrated to compute the pseudo-speed of the ankle during the recording with an unknown initial condition. The first step of our algorithm is to detect phases of inactivity where we assume the ankle velocity as null. Let $\{(t_1^0, t_1^1), \ldots, (t_i^0, t_i^1), \ldots, (t_n^0, t_n^1)\}$ the n detected couples of inactivity instants with the ankle in motion in between. We can integrate $\hat{\boldsymbol{a}}_W$ between t_i^0 and t_i^1 chronologically and in the reverse-time direction to compute what we call respectively forward speed ($\hat{\boldsymbol{v}}_W^f$) and backward speed ($\hat{\boldsymbol{v}}_W^b$). We introduce here their general expression between two instants a and b, with $a < b$:

$$\begin{cases} \hat{\boldsymbol{v}}_W^f(t; a, b) = & \displaystyle\int_0^{t-a} \hat{\boldsymbol{a}}_W(a+u)du + \hat{\boldsymbol{v}}_W^f(a; a, b), \\ \hat{\boldsymbol{v}}_W^b(t; a, b) = & \displaystyle\int_0^{b-t} \hat{\boldsymbol{a}}_W(b-u)du + \hat{\boldsymbol{v}}_W^b(b; a, b). \end{cases} \tag{19}$$

In particular, the instants t_i^0 and t_i^1 are defined as moments where the ankle is motionless, so we assume $\hat{\boldsymbol{v}}_W^f(t_i^0; t_i^0, t_i^1) = 0$ and $\hat{\boldsymbol{v}}_W^f(t_i^1; t_i^0, t_i^1) = 0$. As a result we have:

$$\begin{cases} \hat{\boldsymbol{v}}_W^f(t_i^0, t_i^1)(t) = & \displaystyle\int_0^{t-t_i^0} \hat{\boldsymbol{a}}_W(t_i^0+u)du, \\ \hat{\boldsymbol{v}}_W^b(t_i^0, t_i^1)(t) = & \displaystyle\int_0^{t_i^1-t} \hat{\boldsymbol{a}}_W(t_i^1-u)du. \end{cases} \tag{20}$$

Since the integration drift accumulates errors over time, we make the assumption that for all t in $[a, b]$, the closer t is to b, the more $\hat{\boldsymbol{v}}_W^f(t; a, b)$ produces errors and on

the opposite, the closer t is to a, the more $\hat{\boldsymbol{v}}_W^b(t;a,b)$ produces errors. That is why we compute the pseudo-speed $\hat{\boldsymbol{v}}_W(t;a,b)$ as a weighted mean between a and b :

$$\hat{\boldsymbol{v}}_W(t;a,b) = \hat{\boldsymbol{v}}_W^f(t;a,b)\frac{b-t}{b-a} + \hat{\boldsymbol{v}}_W^b(t;a,b)\frac{t-a}{b-a}. \tag{21}$$

We note t_0 is the first index of inactivity detected and t_{n+1} is the last one. For all $t < t_0$ we can only compute the backward speed, as we do not know the initial condition for $t = 0$:

$$\hat{\boldsymbol{v}}_W^b(t;0,t_0) \;=\; \int_0^{t_0-t} \hat{\boldsymbol{a}}_W(t_0-u)du. \tag{22}$$

In contrast, for all $t > t_{n+1}$ we can only compute the forward speed between t_{n+1} and T because we do not have any information on the speed of the ankle at the end of the recording:

$$\hat{\boldsymbol{v}}_W^f(t;t_{n+1},T)(t) \;=\; \int_0^{t-t_{n+1}} \hat{\boldsymbol{a}}_W(t_{n+1}+u)du. \tag{23}$$

We can now define the pseudo-speed $\hat{\boldsymbol{v}}_W$ during the entire recording. The pseudocode of $\hat{\boldsymbol{v}}_W$ calculation is presented in Algorithm 13. For all t in $[0,T]$ we have:

$$\hat{\boldsymbol{v}}_W(t) = \begin{cases} \hat{\boldsymbol{v}}_W^b(t;0,t_0) & \text{if } \; t < t_0, \\ \hat{\boldsymbol{v}}_W(t;t_i^0,t_i^1) & \text{if } \; t_i^0 < t < t_i^1, \;\; \forall i \in [\![1,n]\!], \\ \hat{\boldsymbol{v}}_W^f(t,t_{n+1},T) & \text{if } \; t_{n+1} < t, \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

Algorithm 13: Pseudo-speed computation.

Input : Inertial data projected in a terrestrial reference frame with gravity removed
Output: Pseudo-speed

1 Detection of inactivity.
2 Definition of intervals with device in motion: $\{(t_1^0,t_1^1),\ldots,(t_i^0,t_i^1),\ldots,(t_n^0,t_n^1)\}$.
3 Backward integration between t_0 and t_1^0 (Equation 22).
4 Forward integration between t_n^1 and t_{final} (Equation 23).
5 **foreach** *interval* $[t_i^0,t_i^1]$ **do**
6 | Forward and backward integration between t_i^0 and t_i^1 (Equation 20).
7 | Weighted mean of the forward speed and backward speed (Equation 21).
8 **end**

2.1 Pseudo-speed norm extrema for interval extraction

As inertial criteria (acceleration norm close to one g and local minima of angular velocity) described in Chapter III for extracting candidate stride intervals, the norm of $\hat{\boldsymbol{v}}_W$ is a good variable to detect the start and the end of a stride during walking. This characteristic is illustrated in Figure 40, the stride segmentation is given by minima around a maximum. Yet, while the previous extraction algorithm has shown its limits in situations such as fast side stepping, prompt descent of stairs or during small steps with soft foot contact, the pseudo-speed norm criteria

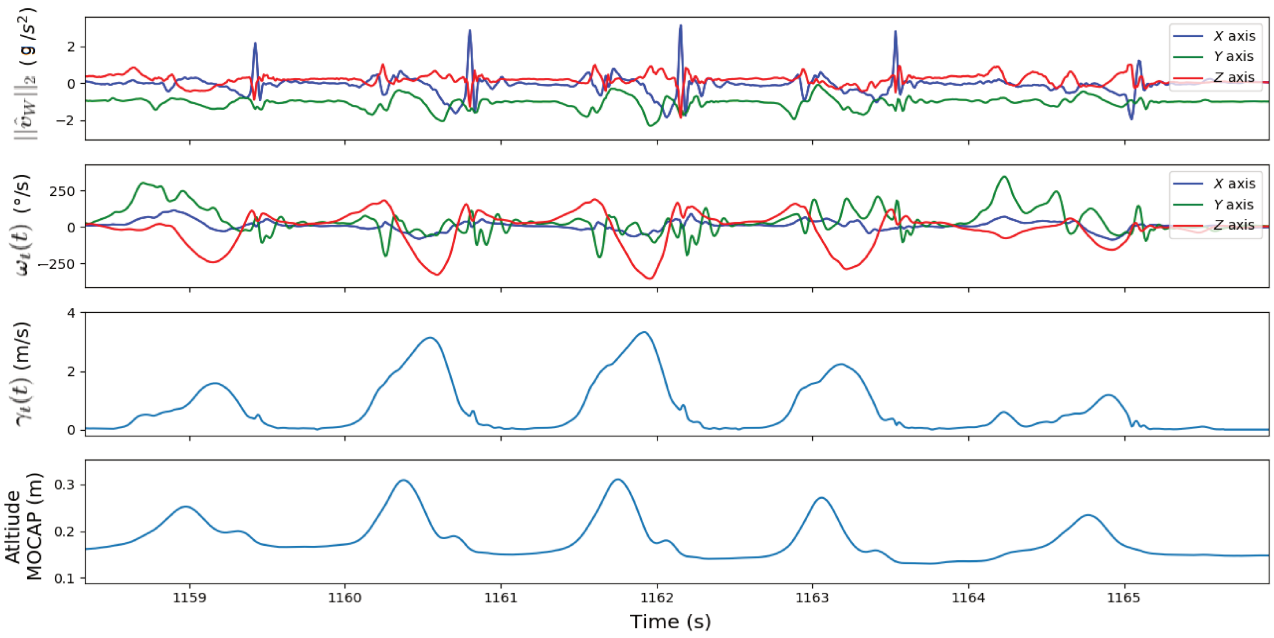


Figure 40: Pseudo-speed norm illustration during walking.

overtake this issue (see Figure 41 and Figure 42). These graphs correspond to the same periods represented in the previous chapter, leading to difficult strides detection from the inertial data. The pseudo-speed norm shows here the same pattern than the MOCAP altitude, providing small speeds when the foot is on the ground and higher values when the foot is in the air. Hence it allows to visually detect the strides.

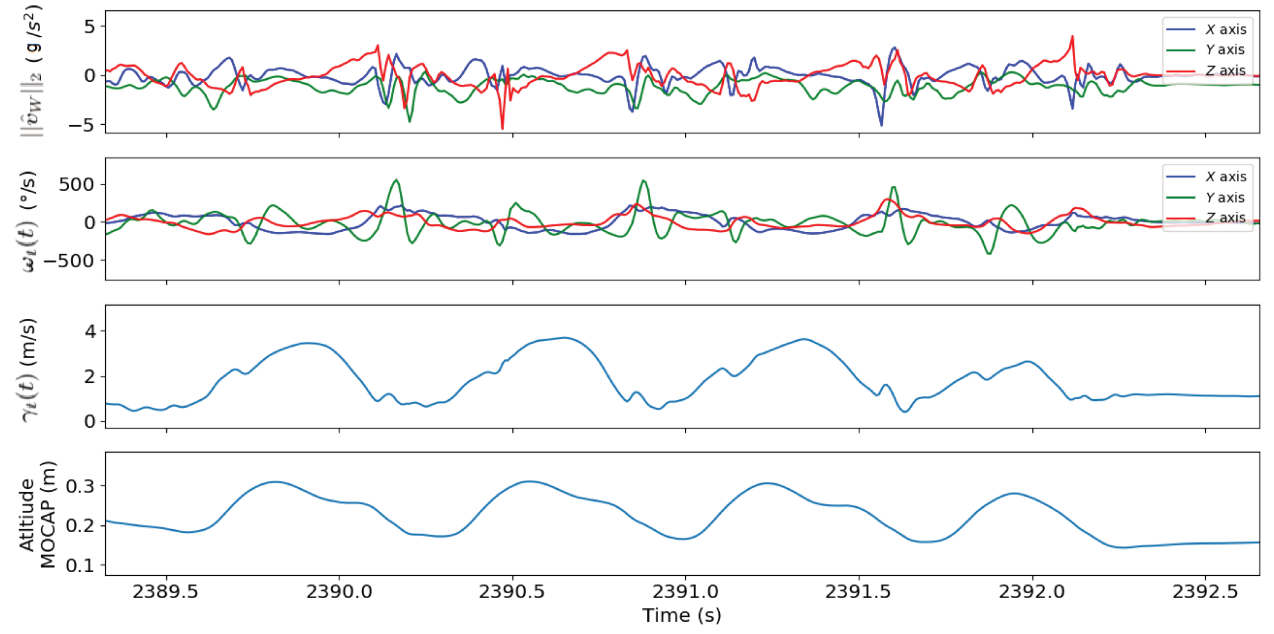


Figure 41: Pseudo-speed norm illustration during fast side steps.

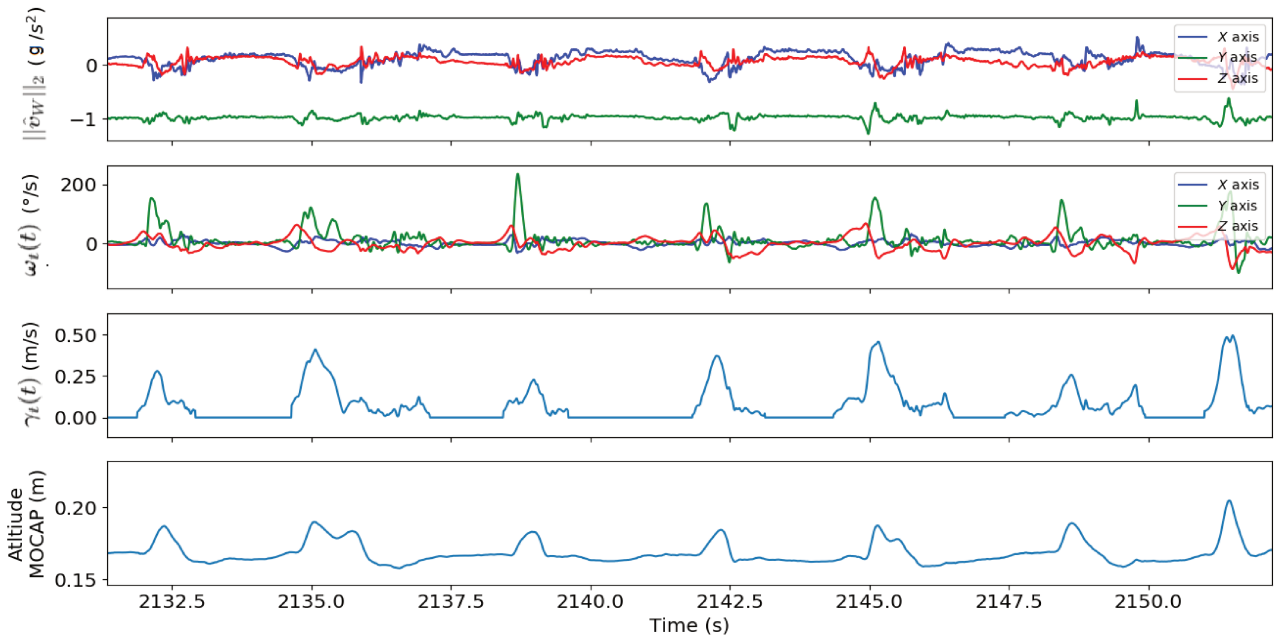


Figure 42: Pseudo-speed norm illustration during soft small steps.

However, by following this extraction procedure based on $\hat{v}_W$ norm criteria minima, many intervals are wrongly extracted when the wearer is moving their ankle but not walking (for instance during bicycling). Moreover, The algorithm is designed to be used in the daily life context during clinical trials. Thus, in many situations the sensors are recording while the ActiMyo device is not worn at the ankle. For example, during the required time to install or uninstall the system from the case, when it is carried by hand, put in a pocket or a backpack. These situations are common and may produce several interval extractions. The goal is now to select among these intervals which ones are true strides. We adopt a statistical learning approach to answer this problem, building a GBT classifier (see Section II.2.2) from relevant features we describe in the following Section 3.

3 Features engineering

Proceeding by the same method introduced in Section III.2, we launched the interval extraction algorithm based on pseudo-speed norm on the "DB3" recordings (see Section II.4.1). They are provided by a dozen of co-workers of various ages and heights who were filmed practicing several activities while wearing the system at the ankle and the wrist at Sysnav company under video control. With synchronisation between the video and ActiMyo data time scales, the performed strides have been annotated -1 while the non stride extracted intervals are labelled with the label 1. Our database contains about 6000 positive intervals and about 6000 negative intervals. As the labeling is a time-consuming task, the number of labeled intervals is relatively small compared to other supervised problems. As a consequence, in order to help the GBT algorithm to provide a classifier with good performance it is crucial to extract relevant information from the data we have access to for each interval.

For all recordings, the inertial data and pseudo-speed are projected in the terrestrial frame W where the Z_W axis is aligned with gravity and the two other axes (X_W and Y_W) are set arbitrarily (see Section IV.1). This orientation is dependent to each recording and is a difficulty from a statistical learning point of view. Indeed, the same stride performed with different X_W and Y_W corresponds to different sit-

uations to learn. In the following we present a new rotation matrix around Z_W that projects any interval in a terrestrial reference frame invariant to the initial X_W and Y_W axes. It is based on a speed estimation at the end and the start of the extracted interval (see Section IV.3.1), then the alignment procedure is described in Section IV.3.2. Finally we present in IV.3.3 the computed features from functional data analysis techniques on the time series we have access to for each considered interval.

3.1 Speed estimation at extracted interval boundaries

With the notations introduced in Section II.4.2, the family $\hat{\mathcal{I}}$ of one recording represents the set of n intervals extracted from the algorithm described in Section IV.2.1. The intervals are defined by one start and one end: $\hat{\mathcal{I}} = \{(\hat{s}_1, \hat{e}), \ldots, (\hat{s}_i, \hat{e}_i), \ldots, (\hat{s}_n, \hat{e}_n)\}$. Inspired by the Sysnav algorithm developed for trajectory reconstruction of an ankle-mounted IMU pedestrian in Section I.4.4, we assumed that during the beginning and the end of a stride, when the foot is flat on the floor, the ankle is in rotation around the heel. Then, if the ith interval is a true stride we assumed that the ankle speed at $\hat{s}_i$ and $\hat{e}_i$ is given by a lever arm model:

$$\begin{cases} \hat{\boldsymbol{v}}_W(\hat{s}_i) = & \boldsymbol{\omega}_W(\hat{s}_i) \times \begin{pmatrix} 0 \\ 0 \\ r \end{pmatrix}, \\ \\ \hat{\boldsymbol{v}}_W(\hat{e}_i) = & \boldsymbol{\omega}_W(\hat{e}_i) \times \begin{pmatrix} 0 \\ 0 \\ r \end{pmatrix}, \end{cases} \tag{25}$$

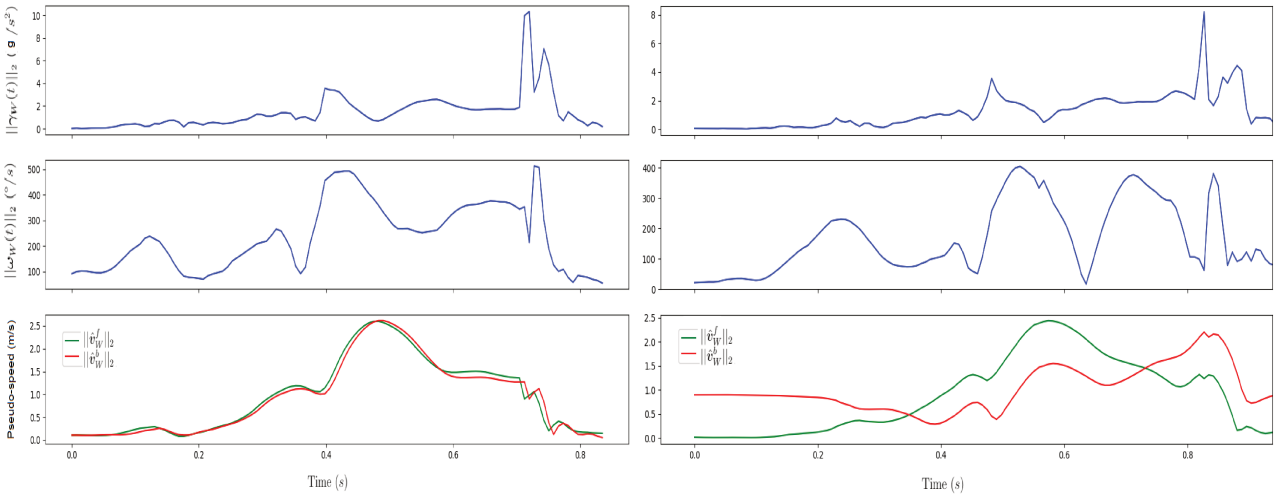
with r the device's height relative to the ground. In practice, we set the value of r to 8 cm. From Equation (19) we can compute the forward speed $\hat{\boldsymbol{v}}^f_W(t; \hat{s}_i, \hat{e}_i)$ and the backward speed $\hat{\boldsymbol{v}}^b_W(t; \hat{s}_i, \hat{e}_i)$ for t in $[\hat{s}_i, \hat{e}_i]$.

If the interval i is a stride, these two speeds are close because we integrate the acceleration during a short period so that the drift stays small. However if i is a non stride interval, $\hat{s}_i$ and $\hat{e}_i$ do not necessarily correspond to the ankle rocker. That is why we observed differences in the residuals $|\hat{\boldsymbol{v}}^b_W(t; \hat{s}_i, \hat{e}_i) - \hat{\boldsymbol{v}}^f_W(t; \hat{s}_i, \hat{e}_i)|$ for t in $[\hat{s}_i, \hat{e}_i]$ as features. The residuals can indeed be much larger for movements that are not strides. This phenomenon is illustrated in Figure 44 where the computed forward speed norm $||\hat{\boldsymbol{v}}^f_W||_2$ (in green) and backward speed norm $||\hat{\boldsymbol{v}}^b_W||_2$ (in red) have been represented for two extracted intervals that are characterized with very similar inertial norms patterns. The interval on the left is a stride and the interval on the right correspond to a device movement that is not a stride. We can see that the gap between the forward and backward speed is larger for the non stide interval.

3.2 Pseudo-trajectory and sensors alignment

Then, thanks to Equation (21), we compute $\hat{\boldsymbol{v}}_W(t; \hat{s}_i, \hat{e}_i)$ from $\hat{\boldsymbol{v}}^f_W(t; \hat{s}_i, \hat{e}_i)$ and $\hat{\boldsymbol{v}}^b_W(t; \hat{s}_i, \hat{e}_i)$ on each studied interval. By integrating this pseudo-speed, we compute a pseudo-trajectory in the terrestrial reference frame W, starting from the origin $(0, 0, 0)$ and ending in $(\hat{x}_{\hat{e}_i}, \hat{y}_{\hat{e}_i}, \hat{z}_{\hat{e}_i})^T$:

$$\begin{pmatrix} \hat{x}_{\hat{e}_i} \\ \hat{y}_{\hat{e}_i} \\ \hat{z}_{\hat{e}_i} \end{pmatrix} = \int_{\hat{s}_i}^{\hat{e}_i} \hat{\boldsymbol{v}}_W(u; \hat{s}_i, \hat{e}_i) du. \tag{26}$$



(a) (b)

Figure 44: Example of forward speed norm and backward speed norm for an extracted stride interval (**a**) and an extracted non stride interval (**b**).

We consider a new terrestrial reference frame W_i with the Z_{W_i} axis still aligned with gravity but with X_{W_i} defined by $\frac{(\hat{x}_{\hat{e}_i}, \hat{y}_{\hat{e}_i}, \hat{z}_{\hat{e}_i})^T}{||(\hat{x}_{\hat{e}_i}, \hat{y}_{\hat{e}_i}, \hat{z}_{\hat{e}_i})^T||}$. We note that $\boldsymbol{R}_{W_i}$ is the rotation matrix that projects the data from the reference frame W to the reference frame W_i. For one stride interval i, we plot the trajectories in W and W_i (Figure 45). As we align the end of the pseudo-trajectory with the X_{W_i} axis, the value on Y_{W_i} of the end point is null. The body frames W_i are not the same for all i and for all recordings, but they have the same building specifications. The 3D interval data we have access to (trajectory, pseudo-speeds, residuals, acceleration, and angular velocity) in W_i are independent of the initial position of the sensors. By proceeding in this way, we drastically reduced the complexity of the supervised learning problem.

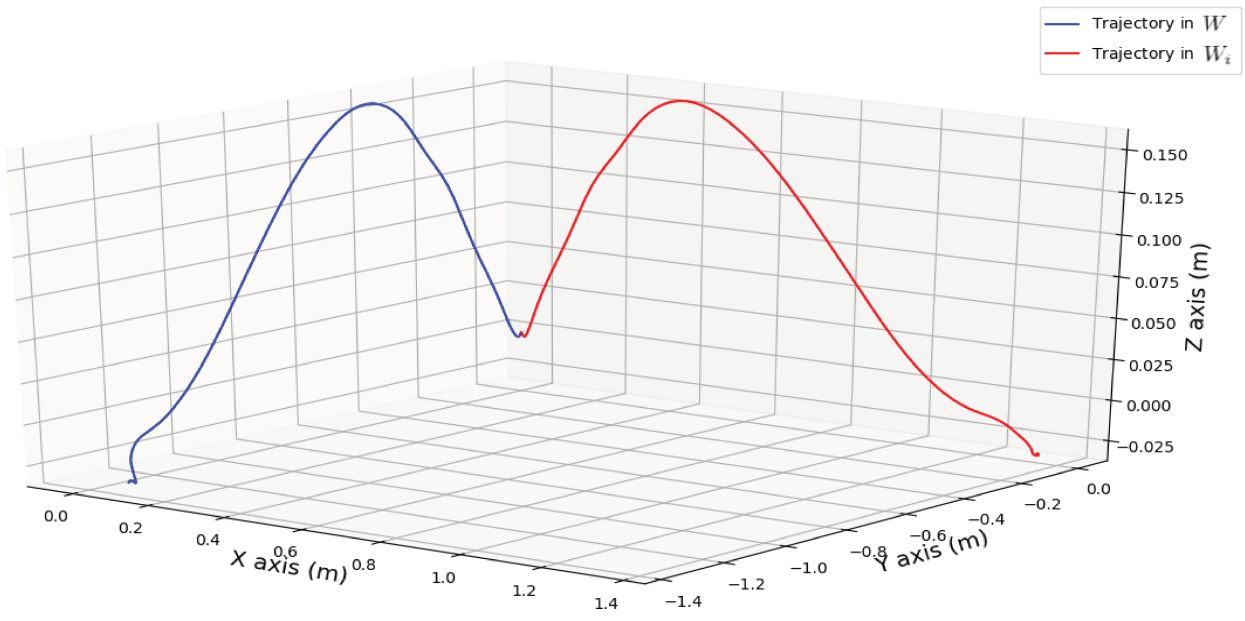


Figure 45: Example of a computed pseudo-trajectory in W and W_i.

3.3 Functional data analysis

The GBT algorithm requires the observations features of the learning set $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, y_1^s), \ldots, (\hat{s}_i, \hat{e}_i, y_i^s), \ldots, (\hat{s}_n, \hat{e}_n, y_n^s)\}$ to be the same variables for all i. However, the size of the extracted intervals $[\hat{s}_i, \hat{e}_i]$ are not all the same. As a consequence, the 3D interval data (pseudo-speed norm, pseudo-trajectory, acceleration, angular velocity etc.) projected in the terrestrial reference frame W_i are not directly suitable for the GBT algorithm. To face this problem, we computed features for each axis from signal processing techniques in time and frequency domains such as maximum, mean, standard deviation, root mean square, inter-quantile range, Fast Fourier Transform, 3^{rd} and 4^{th} order moments, auto-correlation and correlations between signals etc. In the end, the number of features was equal to 1657. This allow us to apply the GBT algorithm to compute the classifier that will identify the strides among the extracted intervals. Its performance is evaluated in Section IV.4.

4 Intervals classification with GBT

The following section describes the performance of the GBT classifier on the extracted interval from the database "DB3" (Section IV.4.1) and the performance of the overall stride detection algorithm during experimental tests in Motion Capture environment "DB4" (Section IV.4.2). As a reminder the definition of these databases are described in Section II.4.1. We also present results in challenging situations, namely when the device is moving but not during walking, such as bicycling, car ride, carried in a pocket etc. (Section IV.4.3).

4.1 Cross-validation performance

In the database "DB3", a dozen of co-workers of various ages and heights were filmed practicing several activities while wearing the system at the ankle at Sysnav company under video control. From this database, the extraction algorithm based on pseudo-speed norm (Section IV.2.1) and the features engineering process (Section IV.3) built the family of intervals $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, \boldsymbol{x}_1^s), \ldots, (\hat{s}_i, \hat{e}_i, \boldsymbol{x}_i^s), \ldots, (\hat{s}_n, \hat{e}_n, \boldsymbol{x}_n^s)\}$ with $\boldsymbol{x}_i^s \in \mathbb{R}^{1657}$. The video control allowed us to affect a label to each extracted interval leading to a annotated family $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, \boldsymbol{x}_1^s, y_1^s), \ldots, (\hat{s}_n, \hat{e}_n, \boldsymbol{x}_n^s, y_n^s)\}$ with the label variable y^s that takes the value 1 is the interval is a stride, -1 if not. The set $\hat{\mathcal{I}}_{DB3}$ contains 6213 stride interval and 6085 intervals extracted from device movements that are not strides. We launched the 10-fold cross-validation method testing several set of GBT parameters (see Section II.2.2). We focused the tuning procedure for important parameters that can lead to overfitting such as the number of trees, the learning rate, the maximum depth of the trees and subsampling proportion. The cross-validation results of the best parameters are presented in the following confusion matrix (Table 9).

	Predicted -1	Predicted 1
Actual -1	5852	233
Actual 1	128	6085

Table 9: Confusion matrix.

The mean error is around 3%, namely ten times more than the cross-validation score reached in Chapter III. However this score depends on the difficulty of the set of extracted intervals from the recordings in "DB3". In this Section, the extraction method based on the pseudo-speed norm only selects a few non strides interval when the wearer is walking (whatever the activity). Thus many device movements coming from challenging situations (hand-carried, bicycling etc.) labelled -1 that looked like true strides from inertial data point of view were included in the learning set. As a result, the final score was slightly deteriorated but it led to a more robust classifier. In order to better analyze the performances of our algorithm, in the following we study the results of our stride detector in controlled environments, meaning we know where strides occurred in the recordings. Two kinds of error were considered: the false negatives (FNs) corresponding to missing strides, and the false positives (FPs) corresponding to non-stride intervals wrongly classified with the label 1.

4.2 False Negative Rate in MOCAP

We launched the stride detector algorithm on the "DB4" recordings. They are providing by a group of seven people who wore the ActiMyo with infrared markers during MOCAP sessions in a 25 m^2 room. Several cameras were set in order to film the whole scene. They broadcast infrared radiation that was reflected by the markers. This allowed the camera to record the position of the markers with sub-millimeter accuracy. The stride detector algorithm provides a family of intervals detected as strides $\tilde{\mathcal{I}}_{DB4} = \{(\hat{s}_1, \tilde{e}_1), \ldots, (\tilde{s}_i, \tilde{e}_i), \ldots, (\tilde{s}_n, \tilde{e}_n)\}$. With the MOCAP altitude (see Figure 28), we can note precisely when the foot is in contact with the ground during the stance phases in "DB4". We note $\mathcal{I}_{DB4} = \{(s_{1,1}, s_{1,2}, e_{1,1}, e_{1,2}), \ldots, (s_{n,1}, s_{n,2}, e_{n,1}, e_{n,2})\}$ the family of strides with $(s_{i,1}, s_{i,2})$ and $(e_{i,1}, e_{i,2})$ delimiting the first and last stance phase of the stride i. In this Section we focus on the False negative rate, namely the strides in $\mathcal{I}_{DB4}$ that are not detected by our stride detector (FN). We consider that a stride i in $\mathcal{I}_{DB4}$ is not detected if there is no element j in $\tilde{\mathcal{I}}_{DB4}$ such as $\tilde{s}_j \in [s_{i,1}, s_{i,2}]$ and $\tilde{e}_j \in [e_{i,1}, e_{i,2}]$. The final results are presented in Table 10.

	Slow Walking Total - FN	**Medium Walking Total - FN**	**Fast Walking Total - FN**	**Small Steps Total - FN**	**Side Steps Total - FN**
Wearer 1	291 - 0%	279 - 0%	216 - 0%	88 - 0%	287 - 0.7%
Wearer 2	306 - 0%	261 - 0%	195 - 0%	67 - 0%	265 - 2.6%
Wearer 3	294 - 0%	219 - 0%	198 - 0%	107 - 4.7%	143 - 3.5%
Wearer 4	297 - 0%	267 - 0%	228 - 0%	145 - 0.7%	301 - 0%
Wearer 5	273 - 0%	249 - 0%	213 - 0%	65 - 7.7%	246 - 0%
Wearer 6	345 - 0%	339 - 0%	327 - 0%	90 - 1.1%	150 - 0.8%
Wearer 7	342 - 0%	246 - 0%	240 - 0%	48 - 8.3%	200 - 0.7%
Total	2148 - 0%	1860 - 0%	1617 - 0%	610 - 2.3%	1592 - 1.1%

Table 10: False negative rates in MOCAP sessions.

All walking strides with various stride lengths and stride durations (see Figure 36) are detected. This means that our detection achieved 100% accuracy for walking phases with various paces. Our algorithm did not detect all atypical strides, but shows very good results while most existing methods described in the literature do not detect them. In particular, our stride detector do not presents weakness whatever the wearer.

4.3 False Positive Rate

A good score for the false negative rate is important, but in return it may lead to an increased false positive rate. We have to pay attention to this type of error, as our algorithm is designed for daily evaluation of the physical conditions of subjects suffering from pathologies associated with movement disorders. Falsely detected strides could deteriorate the statistics during clinical trials. We tested our algorithm on several typical situations that may produce errors: when the ActiMyo system is worn at the ankle but the wearer is not walking (e.g., sitting on a chair and moving the ankle, bicycling, in a car) and when the device is not worn at the ankle (e.g., carried in the hand, in a backpack, in a pocket). The results of several tests are presented in Table 11.

Movement	Walking	Sitting	Bicycling	Car Ride	Hand-Carried	Backpack	Pocket
FP average per hour	0	0	1.7	0	10.1	0	0.1

Table 11: False positive (FP) rates during daily activities.

Our stride detector produced almost no mistakes during these situations. We can see in Table 11 that the most difficult situation is when the ActiMyo device is manipulated in the hand. Some hand movements may look a lot like strides (displacing the system on a table) that even an expert can hardly differentiate by looking at the data we have access to. With 130 Hz frequency, one hour of ActiMyo recording corresponds to 468 thousands data points. The number of extracted non stride intervals depends on the activity intensity. But as an example, one hour of "Hand-Carried" movements can lead to thousands of extracted intervals. Thus, reaching dozens of wrongly predicted intervals still remains a good performance.

5 Contributions

This section described a stride detector algorithm for ActiMyo recordings. The first step of our algorithm consists of a procedure that removes gravity from the linear acceleration. It allows the computation of a pseudo-speed in a terrestrial reference frame that finally provides a family of candidate intervals that may correspond to strides. Some of these intervals are true strides, while others come from recorded movements that are not strides and we want to exclude. We use a gradient boosting trees algorithm to choose the intervals that we consider as true strides. The stride detection given by the GBT classifier showed 100% stride detection success for more than 5600 walking strides and about 98% detection success for more than 2000 atypical strides such as small steps and side steps. In Table 12 and Table 13 we present the atypical strides detection rates from "DB4" recordings for the threshold based algorithm developed by Sysnav and for the two machine learning (M.L.) approaches described in this thesis (Chapter III and Chapter IV).

	Small steps			
	Total	**Threshold based detection**	**M.L. Chapter III**	**M.L. Chapter IV**
Wearer 1	88	75%	100%	100%
Wearer 2	67	68.7%	100%	100%
Wearer 3	107	63.6%	95.3%	95.3%
Wearer 4	145	55.2%	97.9%	99.3%
Wearer 5	65	10.8%	69.2%	92.3%
Wearer 6	90	40.7%	94.8%	98.9%
Wearer 7	48	39.6%	89.6%	91.7%
Total	610	52.9%	93.8%	97.7%

Table 12: Detection rate for small steps.

	Side steps			
	Total	**Threshold based detection**	**M.L. Chapter III**	**M.L. Chapter IV**
Wearer 1	287	50.7%	97.9%	99.3%
Wearer 2	265	56.5%	85.5%	97.4%
Wearer 3	143	52.9%	84.2%	96.5%
Wearer 4	301	59.7%	98.7%	100%
Wearer 5	246	36.1%	100%	100%
Wearer 6	150	51.4%	96.2%	99.2%
Wearer 7	200	44.4%	51.5%	99.3%
Total	1592	50.6%	91.7%	98.9%

Table 13: Detection rate for side steps.

We can see that the limitations observed with the previous M.L. algorithm in Chapter III are much better handled by the algorithm of this chapter. These improvements are mainly due to the new interval extraction procedure that allows to extract the stride intervals even for small steps with light foot contact (see Figure 42) and also during fast side steps (see Figure 41).

Moreover, our stride detector showed its robustness by presenting no error for several critical daily situations such as bicycling, sitting in a car or on a chair, and walking with the device in a backpack or pocket while existing methods in the literature do not consider them. Indeed, they evaluate false positive errors only when the device is correctly worn (on the foot) and the wearer is walking (still generating FP). As our algorithm aims to compute outcomes based on stride trajectory for daily home recordings during clinical, it is important not to produce false positives for stride detection which would distort the medical findings.

The proposed clinical variables are aiming at the motor function domain and include the stride length, stride velocity and distance walked. In order to connect these results to the DMD hospital tests (4-stairs test or 10 meters run test) we want to recognize the activity of each detected strides. In Chapter V we present the algorithm developed by Sysnav that allows to compute the trajectory of a pedestrian wearing the ActiMyo device. This computed trajectory turned out to be a main feature for activity recognition that we describe in Section V.3.

Part V

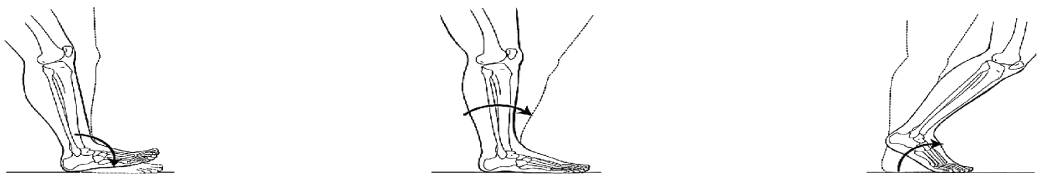
Activity Recognition from Computed Trajectory

In order to compute the trajectory, the strategy consisting of the integration of the linear acceleration and angular velocity data from the unit rapidly accumulates large errors due to Inertial Measurement Units (IMUs) drifts. In Section I.4 we presented the ZUPT technique, which limits the errors by computing the integration only during detected strides, assuming that zero velocity is observed at the beginning and the end. This approach is valid for foot-mounted devices. However, in this thesis we consider an ankle-mounted system, ActiMyo (see Section I.2). The speed of the ankle may reach more than 4 m/s when the foot is on the ground during running. As a result, the zero-velocity assumption is not applicable with our device. In this chapter, we first present in Section V.1 the ZUPT-inspired innovation for PDR with ankle-mounted IMU. Then, we evaluate the algorithm performances in Section V.2 through the strides length thanks in a motion capture environment and through the overall trajectory reconstruction in an uncontrolled environment.

The computed trajectory is a main feature for activity recognition. For example, regarding the relative altitude between the start and the end of a stride, one could imagine being able to detect stairs. In this work, we focus on three activities related to the primary outcomes for patients suffering from Duchenne Muscular Dystrophy (DMD) : stairs, walking, and running (see Section I.3.2). However, defining the difference between running and fast walking regarding the trajectory is a challenging task. Indeed, the age difference of patients in clinical studies can be very large, and their gaits very dissimilar. Thus, we adopted a supervised machine learning algorithm to build a classifier that recognizes the activity of the performed stride given its computed trajectory (Section V.3). The work of this chapter has been presented in a poster session at the International Conference on Indoor Positioning and Indoor Navigation (IPIN) 2018.

1 Trajectory Reconstruction of the Detected Strides

As zero velocity is not observed during the stance phase Sysnav adapted the Zero Velocity Update Technique (ZUPT, see Section I.4.4) by developing a model based on lever arm to estimate the ankle speed at the beginning and the end of a stride. Then the integration of inertial data is computed only in between. When the foot is flat on the floor, we assume the ankle is in rotation around the heel. Thus, the ankle speed is estimated by the cross-product between the vector "heel-ankle" in World frame noted $\boldsymbol{r}_W = (0, 0, r)^T$ and the angular velocity (given by the gyrometer). This process is illustrated in Figure 46.



(**a**) (**b**) (**c**)

Figure 46: The three foot rockers during stance phase: (**a**) heel rocker , (**b**) ankle rocker , (**c**) forefoot rocker.

This procedure still requires to detect when a stride occurs in the recording. For this task we use the stride detector described in Chapter IV that achieved very good performances in several challenging environments. The difficulty compared to ZUPT method is to find the optimum moment where the speed is equal to the

cross product between $\boldsymbol{r}_W$ and $\boldsymbol{\omega}_W$: $\boldsymbol{v}_W = \boldsymbol{\omega}_W \times \boldsymbol{r}_W$. Given an initialization at time t when the rotation matrix between the body frame and world frame $\boldsymbol{R}_W^t$ is known, one can express the inertial data at time $t + \Delta t$ in World frame thanks to angular velocity integration. Then when the measured specific acceleration $\boldsymbol{\gamma}_W(t+\Delta t) - \boldsymbol{g}_W$ is close to the expected specific acceleration that is the derivative of $\boldsymbol{\omega}_W(t + \Delta t) \times \boldsymbol{r}_W$, the instant $t + \Delta t$ is considered as the optimum. Then we update the speed in an extended Kalman filter with a dead-reckoning motion model (see Figure 47) to significantly reduce error growth over time. This filter may for example also include a state with 6 degrees of freedom for speed and attitude. Other states can be added such as position, sensor bias, etc. The filter also gives a measure of the confidence of estimated states. This overall procedure has been patented by Dorveaux E., Jouy A., Grelet M., Vissiere D. and Hillion M. in [82] and is not detailed here. As an example, we represented in Figure 48 the computed trajectory during a walking phase.

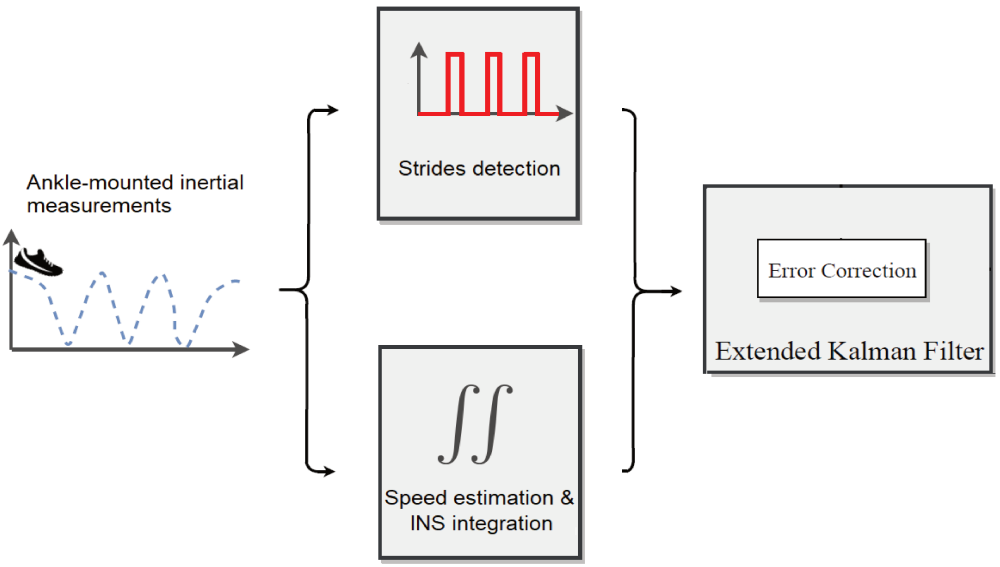


Figure 47: Zero Velocity Update Detection combined with dead reckoning in an extended Kalman filter.

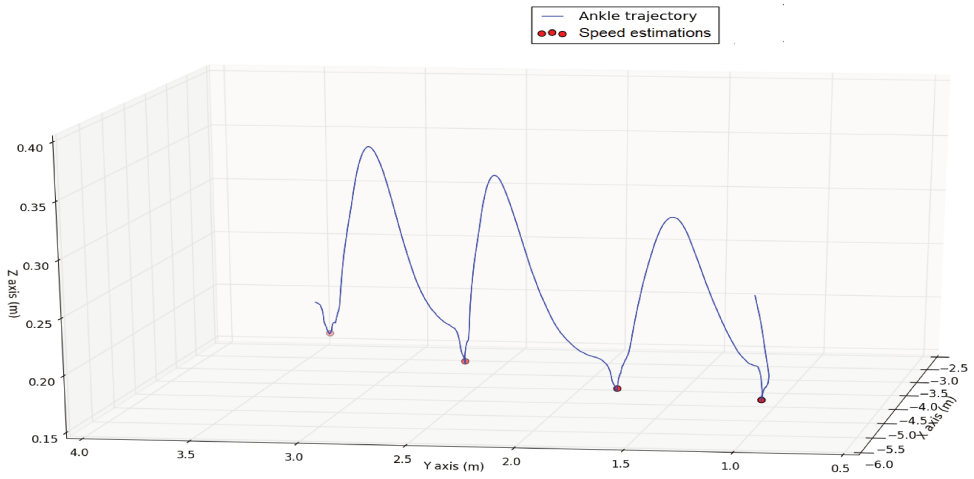


Figure 48: Example of the computed ActiMyo trajectory.

2 Trajectory reconstruction performance

For the evaluation of the trajectory reconstruction algorithm, we used the recordings of "DB4" performed in the MOCAP environment that provides the ground truth ActiMyo trajectory. We also validated the algorithm with a test performed in an uncontrolled environment. A co-worker was wearing the system during several hours at Sysnav free to move in the office as he wanted.

2.1 Stride length performance in MOCAP

We launched on the database "DB4" (See Section II.4.1) our stride detector presented in Section IV. The "DB4" recordgins a providing by a group of seven people wore the ActiMyo with infrared markers during MOCAP sessions in a 25 m^2 room. Several cameras were set in order to film the whole scene. They broadcast infrared radiation that was reflected by the markers. This allowed the camera to record the position of the markers with sub-millimeter accuracy. We saw in Section IV.4.2 that this algorithm allows to detect 100% of the walking strides and more than 98% of the atypical strides. Then we applied the trajectory reconstruction algorithm in order to compare the results with the trajectory provided by the MOCAP infrared infrastructure. As one of the clinical variables for Duchenne Muscular Dystrophy (DMD) is based on daily statistics of stride lengths (see Section I.3.3), we studied the difference between the computed stride length and the ground truth in absolute value. The results are presented in Table 14 and Table 15.

	Slow walking		**Medium walking**		**Fast walking**	
	Mean (m)	Std (m)	Mean (m)	Std (m)	Mean (m)	Std (m)
Wearer 1	0.024	0.038	0.020	0.022	0.029	0.036
Wearer 2	0.026	0.039	0.016	0.018	0.025	0.034
Wearer 3	0.023	0.021	0.028	0.024	0.036	0.023
Wearer 4	0.028	0.020	0.028	0.021	0.023	0.022
Wearer 5	0.061	0.091	0.025	0.020	0.032	0.029
Wearer 6	0.018	0.016	0.024	0.024	0.043	0.039
Wearer 7	0.014	0.026	0.014	0.012	0.023	0.044
Total	0.028	0.048	0.022	0.021	0.032	0.034

Table 14: Absolute computed stride length error for walking phases.

	Small steps		**Side steps**	
	Mean (m)	**Std (m)**	**Mean (m)**	**Std (m)**
Wearer 1	0.027	0.070	0.044	0.106
Wearer 2	0.039	0.048	0.071	0.168
Wearer 3	0.057	0.082	0.070	0.177
Wearer 4	0.025	0.024	0.048	0.056
Wearer 5	0.049	0.082	0.022	0.046
Wearer 6	0.074	0.061	0.133	0.170
Wearer 7	0.039	0.055	0.053	0.116
Total	0.048	0.069	0.056	0.129

Table 15: Absolute computed stride length error for atypical strides.

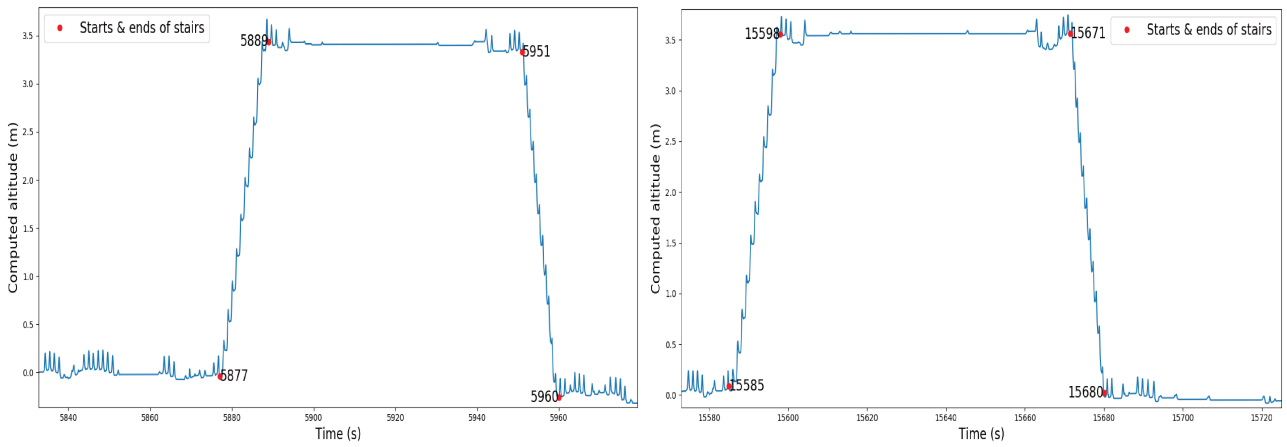
Our algorithm achieved similar performance to existing methods [37, 34] for normal walking but also achieved good results for atypical strides (around 5 cm

of absolute mean error) that are not even studied in the literature. Still, our algorithm produced more error for these kinds of strides compared to walking ones. This can be explained by the lever arm model that is less consistent for the ankle speed estimation at the beginning and end of strides.

2.2 Performance in Uncontrolled Environment

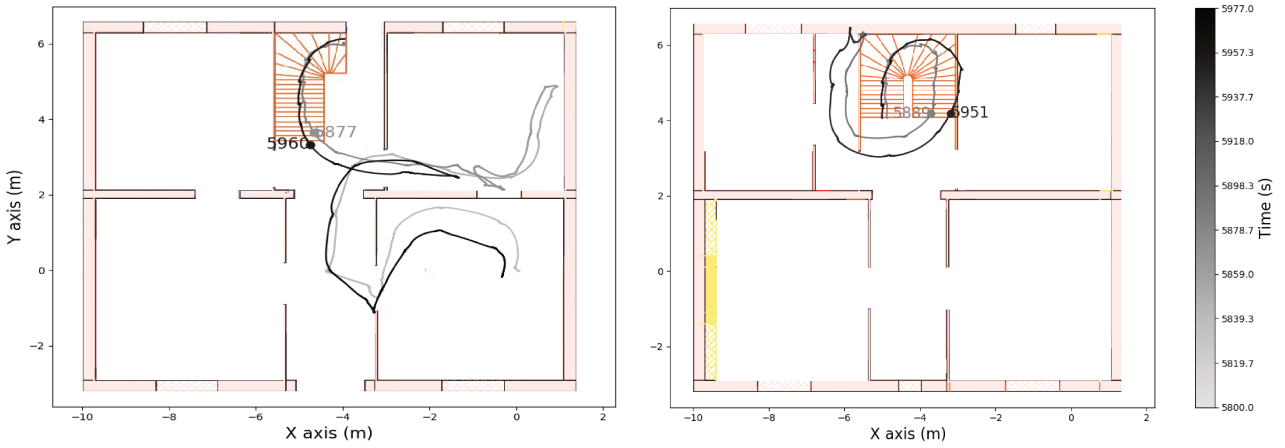
In this section, we validate the trajectory reconstruction in an everyday life situation. For this experiment, an office worker wore the system for 5 hours and a half. The aim was to test the stride detector and trajectory reconstruction algorithm for strides performed naturally, including small steps. During this experiment, the wearer was mostly sitting on his office chair. These periods are also interesting because the ankle did not remain inactive and it is important that no stride was wrongly detected.

The recording contains three walking phases, including upstairs and downstairs in the first and last walking phases whose we have represented the computed altitude in Figure 49. From this graph we can detect when the wearer was walking on the stairs. In Figures 50 and Figure 52 we plot the computed trajectory in two dimensions on the maps of the ground floor and first floor depending on the altitude evolution. We added markers indicating the beginning and end of the detected stairs from Figure 49. The colormap defines the time over the considered walking period: the darker the grey, the more time that had elapsed. In addition, the starting point was initialized with the coordinates $(0, 0, 0)$ and the computed trajectory was rotated to have the correct first direction.



(**a**) (**b**)

Figure 49: Computed altitude during (**a**) the first walking period and (**b**) the third walking period.



(**a**) (**b**)

Figure 50: Computed trajectory during the first walking period on (**a**) the ground floor and (**b**) the first floor.

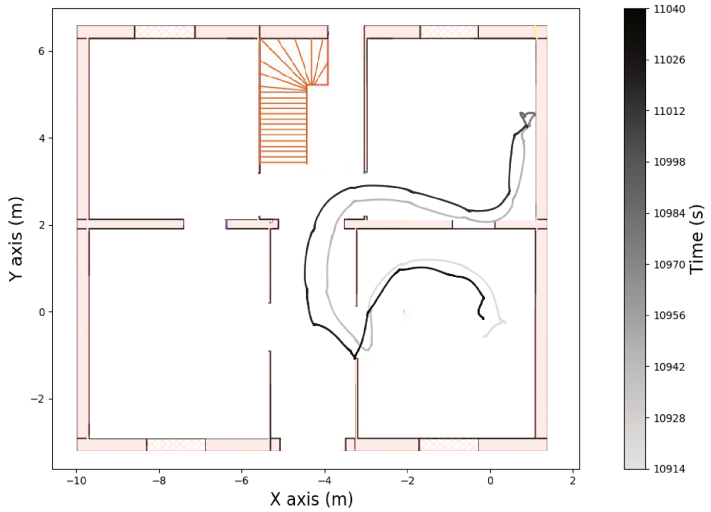
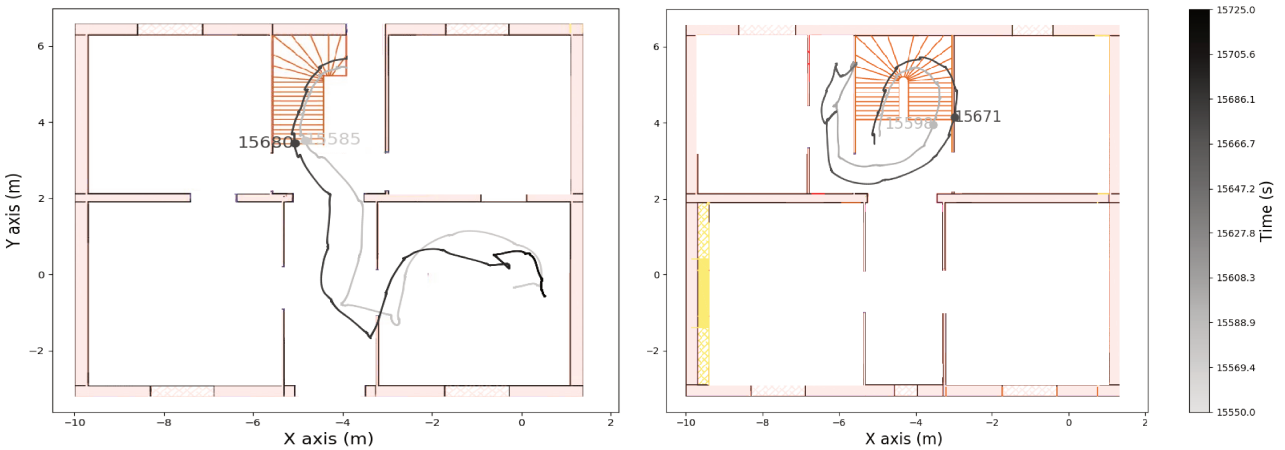


Figure 51: Computed trajectory during the second walking period on the ground floor.



(**a**) (**b**)

Figure 52: Computed trajectory during the third walking period on (**a**) the ground floor and (**b**) the first floor.

This experiment illustrates the good performance of the trajectory reconstruction in a difficult environment with narrow areas, small rooms, and corridors. In

this context, the computed trajectory almost never crossed the walls and we could identify which room the wearer was in at any given time or when he was taking the stairs. In Figure 49 we can see a difference in the computed altitude of 3.4 m for both stairs phases that were composed of 21 stair treads of 15.4 cm height. The true altitude of the first ground is 3.234 m, so the altitude mean error was less than 1 cm for each stair tread. In addition, the starting points of the second and third walking periods correspond to the ending point of the previous ones. This means that no stride was wrongly detected when the wearer was on his chair and moving his ankle.

The computed altitude allowed to detect when the wearer was walking on stairs during the recording. More generally, the computed trajectory of a stride is a relevant feature to recognize the activity. In the following, we describe a supervised learning approach for activity recognition based on the computed trajectory of the detected strides.

3 Activity Recognition of the Detected Strides with Machine Learning from the Computed Trajectory

During clinical studies, activity recognition is a precious information to evaluate the health of patients suffering from movement disorders. In this work, we focus on three activities related to the primary outcomes for Duchenne Muscular Dystrophy (DMD): stairs, walking, and running (See I.3.2). However, defining the difference between running and fast walking regarding the trajectory is a challenging task. Indeed, the age difference of patients in clinical studies can be very large, and their gaits very dissimilar. Moreover, detecting stairs is often more difficult than in Section V.2.2. Some patients suffering from DMD can hardly take them and go up the stairs one by one (the difference in the computed altitude is small). Thus, we adopted supervised machine learning algorithm to build a classifier that recognizes the activity of the performed stride given its computed trajectory.

3.1 Cross-validation performance with GBT

We launched the stride detector (see Algorithm 7) in "DB3" recordings with the interval extraction approach described in Chapter IV. The "DB3" database is composed by recordings performed by a dozen of co-workers of various ages and heights were filmed practicing several activities while wearing the system at the ankle at Sysnav company under video control (See Section II.4.1). The stride detector provides a family of classified intervals $\hat{\mathcal{I}}_{DB3} = \{(\hat{s}_1, \hat{e}_1, \boldsymbol{x}_1^s, \hat{y}_1^s), \ldots, (\hat{s}_i, \hat{e}_i, \boldsymbol{x}_i^s, \hat{y}_i^s), \ldots, (\hat{s}_n, \hat{e}_n, \boldsymbol{x}_n^s, \hat{y}_n^s)\}$ where $\boldsymbol{x}_i^s$ defines the features computed following the procedure described in Section IV.3 and $\hat{y}_i^s$ defines the prediction (1 if it is a stride, -1 if not). This allows to compute the trajectory with the algorithm presented in Section V.1.

Keeping the notations introduced in Section II.4.2, the family $\tilde{\mathcal{I}}_{DB3}$ defines the extracted intervals classified as strides ($\hat{y}_i^s = 1$). With the video control, we keep among the elements in $\tilde{\mathcal{I}}_{DB3}$ the intervals that are true strides and we affected a label y^a defining the activity of the performed stride among "atypical strides" (label 1), "walking" (label 2), "running" (label 3), "upstairs" (label 4) and "downstairs" (label 5). Thus we have a family of intervals corresponding to strides $\bar{\mathcal{I}}_{DB3} = \{(\bar{s}_1, \bar{e}_1, \boldsymbol{x}_1^s, y_1^a), \ldots, (\bar{s}_i, \bar{e}_i, \boldsymbol{x}_i^s, y_i^a), \ldots, (\bar{s}_n, \bar{e}_n, \boldsymbol{x}_n^s, y_n^a)\}$ where $y_i^a \in \{1, 2, 3, 4, 5\}$ (see Table 16).

Activity	Atypical Stride	Walking	Running	Uptairs	Downstairs
Label	1	2	3	4	5

Table 16: Label definitions for activity recognition.

One could at this stage compute a prediction function from GBT algorithm (See Section II.2.2), for multi-class classification in order to predict the target y_i^a from $\boldsymbol{x}_i^s$. However we wanted to take advantage of the computed trajectory that turned out to be a main feature for the activity recognition task. As you can see in Figure 48 the algorithm described in Section V.1 computes the trajectory in an arbitrary reference but we can extract information by considering the relative evolution. This technique also provides the speed in the three dimensions and the angle evolution of the device, which have characteristic patterns according to the activity performed. In the end, we computed 510 extra features for each interval in $\bar{\mathcal{I}}_{DB3}$ leading to a set $\{(\boldsymbol{x}_1^a, y_1^a), \ldots, (\boldsymbol{x}_n^a, y_n^a)\}$ with $\boldsymbol{x}_i^a \in \mathbb{R}^{p^a}$.

We tested several supervised learning algorithms with several hyperparameters for multi-class classification (five classes for five activities). Once again, GBT provided the best results using the 10-fold cross-validation. The confusion matrix is presented in Table 17.

	Predicted 1	**Predicted 2**	**Predicted 3**	**Predicted 4**	**Predicted 5**
Actual 1	1138	14	0	0	0
Actual 2	17	1185	0	2	2
Actual 3	0	0	1334	0	0
Actual 4	0	2	0	1098	0
Actual 5	0	0	0	0	1155

Table 17: 10-fold cross-validation results of the GBT classifier for activity recognition.

The global score was about 99.4%. The difference between ”atypical stride” and ”walking” is difficult to define, especially for a small forward step. As even the labelling decision by the video viewer is difficult, it is not surprising that most errors were between these two classes.

3.2 Algorithm Overview

We can now compute the entire algorithm for activity recognition on clinical study recordings and especially on the recordings on "DB5" (See Section II.4.1) containing several episodes of walking, running and stairs performed by DMD patients at hospital. The overall algorithm (see Algorithm 14) is described in pseudo-code.

Step 14 of Algorithm 14 is important to counter the integration drift if the ankle movement period $[t_i^0, t_i^1]$ is large. In that case, the weighted mean of forward speed and backward speed (see Equation 20 and Equation 21) may not overtake the integration errors for t far from t_i^0 and t_i^1. With the pseudo-speed update, if strides are detected, the weighted mean is computed for smaller and smaller intervals and overcomes the integration drift.

3.3 Activity Recognition in Controlled Environments

In this section we first validate the activity recognition algorithm on the "DB5" recordings" containing hospital Duchenne Muscular Dystrophy (DMD) tests with

Algorithm 14: Activity recognition algorithm.

Input : Recording of the ActiMyo system worn at the ankle
Output: Activity recognition

Projection of the inertial data in a terrestrial reference frame W (Section 1).
Detection of inactivity times $\{(t_1^0, t_1^1), \ldots, (t_i^0, t_i^1), \ldots, (t_n^0, t_n^1)\}$.
Pseudo-speed computation (Section 2).
foreach *ankle movement period* i **do**
    Extraction of the set of candidate stride intervals $\hat{\mathcal{I}}$ ((Section 2.1).
    foreach *interval* j *in* $\hat{\mathcal{I}}$ **do**
        Features computation (Section 3).
        GBT binary classification for stride detection.
        if *interval classified as a stride* **then**
            Trajectory reconstruction (Section 1)
            Computation of the trajectory features.
            GBT multi-class classification for activity recognition.
            Pseudo-speed update: $\hat{\boldsymbol{v}}_W(t) = \hat{\boldsymbol{v}}_W(t, \hat{e}_j, t_i^1)$, $\forall t \in [\hat{e}_j, t_i^1]$ (Equation (21))
            Update of the candidate intervals extraction posterior to $\hat{e}_j$
        end
    end
end

video control. The four-stairs test consists of DMD patients climbing four stairs as quickly as possible. During one session, they performed the test several times with the ActiMyo device at the ankle. We studied the performance of our activity recognition algorithm (see Algorithm 14) by counting the number of true stairs strides that were missing (FN). One FN stride was either classified as another activity or not detected at all. The results are presented in Table 18.

	Patient 1	**Patient 2**	**Patient 3**	**Patient 4**
Total	4	15	10	7
FN	0	0	0	0
	Patient 5	**Patient 6**	**Patient 7**	**Patient 8**
Total	16	20	16	15
FN	9	1	0	0

Table 18: Activity recognition for DMD recordings: false negative rates during four-stairs tests.

The algorithm performed well for all patients except one (patient 5). This can be explained by the difficulty this patient had in climbing the stairs. In Figure 53 we show the video recording during one stairs step every half second. The patient took 5 seconds to climb it for a small altitude variation. This kind of stride does not exist in the learning dataset built from "DB3" recordings where stairs were performed by healthy adults co-workers. Thus, it is not surprising that the GBT prediction function for activity recognition classified the stride as "atypical stride". In addition, several patients performed a run in a 10-m hospital corridor with the ActiMyo device at the ankle. We adopted the same evaluation, by counting the number of missing running strides defined as classified as another activity or not detected (FN). The results are presented in Table 19.

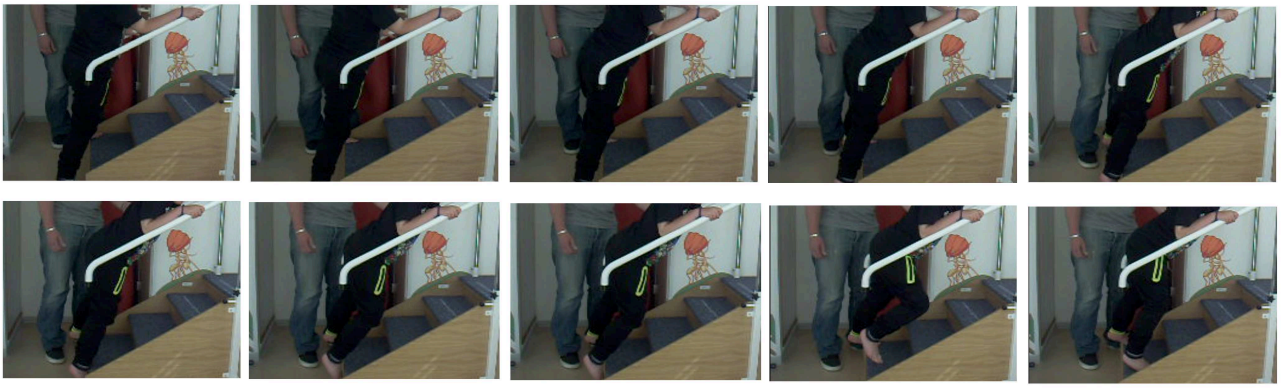


Figure 53: Example of one stairs step progression every half second of patient 5.

	Patient 1	Patient 2	Patient 3	Patient 4	Patient 5
Total	26	18	13	18	14
FN	0	0	2	0	0
	Patient 6	**Patient 7**	**Patient 8**	**Patient 9**	**Patient 10**
Total	32	12	14	21	22
FN	0	0	0	2	1

Table 19: Activity recognition for DMD recordings: false negative rates during four-stairs tests.

The success rate was about 97.4% less than the 99.4% presented in Table 17. This is due to the fact that most of the labeled strides for activity recognition were performed by Sysnav employees. As a result, the GBT classifier for AR achieved better results for adult recordings. Moreover, the classification of DMD activities may be a more challenging task from a statistical point of view.

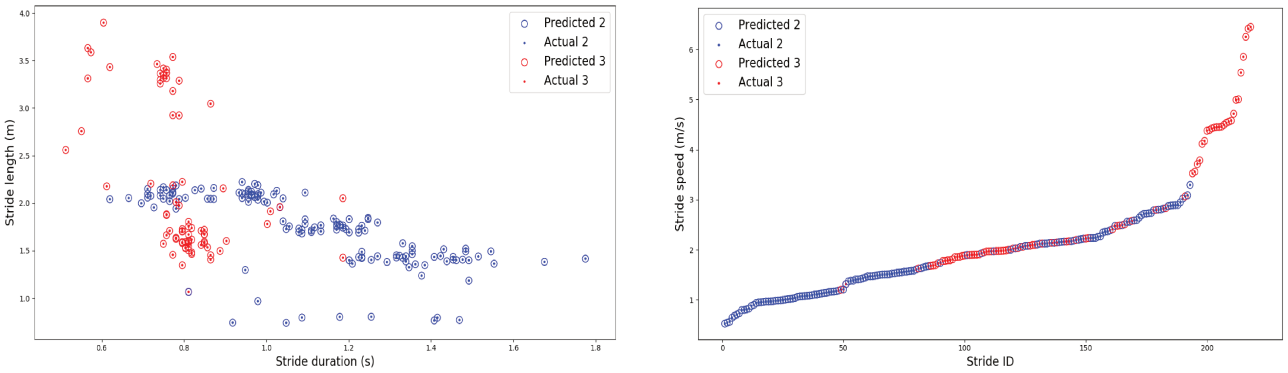
During the 6-minute walk tests, DMD patients performed hundreds of strides. Our algorithm produced no error for these walking phases. Several running strides were detected but validated by the video control. This again shows that this kind of test is not relevant to measure the health of the patients because those who run while they have to walk skew the results.

We validated our algorithm during another test with a Sysnav employee that performed 139 walking strides and 79 running strides with various speeds. The classification results are presented in Figure 54. We can see that several running strides speed were below walking strides. Nevertheless, the GBT predictions for activity recognition produced only one error (ID = 51, stride duration ≈ 0.8 s, stride length ≈ 1 m). This shows the benefits of having a supervised learning approach rather than setting empirical thresholds for the stride length, duration, or speed that would not work here.

In this section, we validated our algorithm for activity recognition in controlled environments. The wearers were asked to perform the activities we wanted to test. However, our algorithm was designed to be applied for home recordings during clinical trials. In uncontrolled environments, the activity recognition is more challenging. In the following section we present the stairs detection for one home recording.

3.4 Activity Recognition for One Healthy Child Recording in Uncontrolled Environment

The child of one coworker agreed to wear the ActiMyo device at home for one day as a DMD patient would do during a clinical trial. He is not affected by the



(a) (b)

Figure 54: Activity recognition (AR) results associated with the distribution of (**a**) the strides length/duration and (**b**) the strides speed.

disease but this recording is interesting to study because he was living without any constraint and the stairs were partially annotated by our coworker. The house is composed of two staircases: one flight of stairs with two sets of seven steps and a small set of stairs comprising three steps. We denote them respectively as "main stairs" and "small stairs". The activity recognition results are presented in Table 20. Every stairs event corresponded to strides classified as "downstairs" or "upstairs" by the GBT function prediction. The number of strides can vary depending on the first foot starting the stairs (wearing the ActiMyo device or not) or if the wearer climbs or descends the steps stairs two in a row. Moreover, we had already observed that the first or last stairs step were difficult to recognize because the altitude variation is small and the foot forward swing is as large as a walking stride. In conclusion, the fact that stairs strides were detected for every stairs event confirmed that our activity recognition algorithm performs well even in an uncontrolled environment (but of course we cannot report with 100% confidence that no stride was missing).

Event (number of appearances)	**Number of detected stairs strides per event**
Climbing main stairs (3)	7–6–8
Descending main stairs (2)	8–6
Climbing small stairs (4)	2–1–1–2
Descending small stairs (1)	2

Table 20: Activity recognition: detected stairs strides associated with event annotation for one healty child.

The AR algorithm is ready to be applied to DMD recordings during clinical trials. The goal is to compute relevant clinical outcomes based on detected stairs strides and running strides. Inspired by the four-stairs test in hospital, one could compute the duration of climbing four stairs. Moreover, doctors have the intuition that the number of running strides would be interesting to study. The ability to run would indeed be the first noticeable physical capacity to be lost for DMD patients. Not being biased by the controlled environment, statistics over a long period in a home environment would be more representative than classical hospital tests (see Section I.3.2).

4 Contributions

This section described the overall algorithm for inertial sensors worn at ankle that enables trajectory computation and activity recognition for each detected strides in the recordings. Our approach is divided into four main stages with two machine learning predictions (see Figure 55).

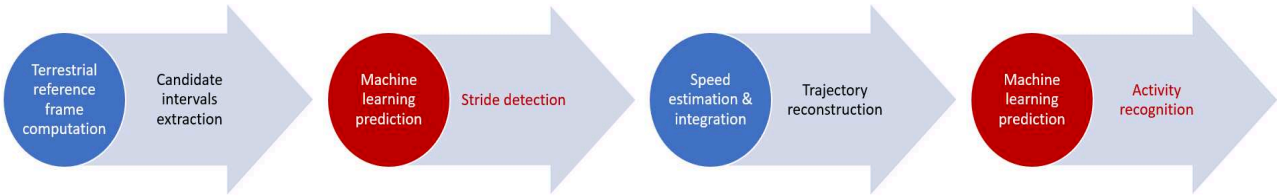


Figure 55: The four main algorithm stages with machine learning uses (red).

The first step of our algorithm is described in Chapter IV. It consists of a procedure that removes gravity from the linear acceleration, allowing the computation of a pseudo-speed in a terrestrial reference frame that finally provides a family of candidate intervals that may correspond to strides. Some of these are real strides, while others come from recorded movements that are not strides and we want to exclude. We use a gradient boosting trees algorithm to choose the intervals that we consider as real strides.

From the stride detection, trajectory reconstruction is computed with an inspired ZUPT technique (see Section I.4). As the ankle speed is not null when the foot is on the ground, we used a speed estimation based on a lever arm model that is fused with inertial integration in an extended Kalman filter. It achieved around 3 cm absolute mean error for the walking stride length and about 5 cm for atypical strides. In the literature, existing methods [37, 34] achieved similar performances for walking strides but do not even study atypical strides (as they faced difficulties to detect them). Our algorithm aims to be applied for daily recordings during clinical trial. In home situations, a majority of small steps are performed. In Section V.2.2, we showed the good performance of the trajectory reconstruction in a difficult environment with narrow areas, small rooms, corridors and stars. In this context, the computed trajectory almost never crossed the walls of the house and regarding the altitude we could easily identify when the wearer was taking the stairs.

More generally we use the computed stride trajectory to recognize the activity with a machine learning approach which was robust to the gait variety. Indeed, the age difference of patients in clinical studies can be very large, and their gaits very dissimilar. Our algorithm performed well for adult recordings (more than 99% success) and also recordings of patients suffering from Duchenne Muscular Dystrophy (more than 97% success), which is a challenging task. This original approach allows classification of the detected strides into five main labelled activities: "atypical stride", "walking", "upstairs", "downstairs" and "running". We believe that our methodology is ready to be applied to home recordings over long periods to compute clinical outcomes related to hospital tests (four stairs, 10-m run) in clinical trials.

Part VI

Parkinson's event detection with sliding window

This chapter presents an innovative and generic deep-learning approach for issues based on inertial data recordings. Throughout this chapter, we focused our attention on the detection of parkinsonian symptoms, concurrently to the issue of activity recognition (see Section VI.1). The models have been built in order to precisely detect the time boundaries of specific movements (tremors, dyskinesia, activities) with a sliding window approach. Our work is based on multi-channel networks (Section VI.2), using in particular Convolutional Neural Networks and one Topological Data Analysis channel that improved the performances (see Section VI.2.4). Finally in Section VI.3 we launch our algorithm for Parkinson's event detection with home recordings of several patients allowing to compute interesting statistics.

1 Data Background

In this section we describe the dataset that allowed us to build our neural network model. The HAPT dataset firstly described in Section VI.4.1 and reintroduced in Section VI.1.1 was used to optimize our neural network architecture that has been adapted for Parkinson's event detection through a sliding window approach presented in Section VI.1.2.

1.1 Parkinson's event dataset and HAPT dataset

The Parkinson's event dataset "DB1" is composed by 13 patients who underwent tests at the hospital while wearing the ActiMyo device at the ankle and the wrist. The recordings have been annotated by the doctors through real-time observation. We described in Section VI.1.1 the issues related to the confidence in the labeling. As an example, we noticed that pauses in a long diskynesia crisis are often not considered and are wrongly annotated the same as the rest of the event. Among the encountered problematics of ground truth, this database presents two particularities that add difficulty for the supervised learning approach. The amount of healthy daily home recordings from the controls are enormous (several months of studies) whereas the total tremors and dyskinesia crises duration is several hours. Thus this database is large (heavy to learn with supervised learning algorithms) and unbalanced, namely proportions between the targeted classes are very different. Indeed, one daily recording of 8-9 hours with ActiMyo 130 Hz frequency provides almost 4 millions of data points in three dimensions for acceleration and angular velocity (both ankle and wrist). Hence, several months of clinical study correspond to billions of labelled data points recorded by controls. Instead, the annotated events by the doctors represent barely 2 millions of annotated data points on the ankle and wrist. This motivated us to build our algorithm on another database of activity recognition from inertial sensors, presenting the same unbalanced particularity but with smaller amount of data.

The open-source HAPT dataset is a newest version of the UCI HAR Dataset. The data is made of 30 volunteers who were asked to perform a set of six basic activities (standing, sitting, lying, walking, upstairs and downstairs). Inertial data about those activities and their transition were recorded by a smartphone placed on their waist, sampling the 3-axial acceleration and the 3-axial angular velocity at a frequency of 50 Hz. This database is also linked to imbalance learning due to the poor amount of gathered events for transition activities. In addition, the activity recognition task is very similar to Parkinson's event detection as we can consider tremors or dyskinesia as particular activities of patients that we want

to detect among all the other daily activities. The total recordings duration is about 4 hours which makes it easier to test different approaches compared to Parkinson's database. With 50 Hz frequency, it corresponds to 720 thousands recording points versus billions in "DB1".

1.2 Sliding Window approach

In order to build features for the machine learning tasks, we applied the classical method of the 50% overlapping sliding window on both datasets presented above. Multiple time window widths were tested. Our choices were determined by the minimum length recorded for Parkinson's event annotations, which is 6 seconds. The time window of 3 seconds appeared to be the most efficient for the ankle, while a width of 5 seconds performed better for the wrist. As a consequence, each dataset is composed of split inertial signals, matched with their corresponding annotated events. For Parkinson's envent detection task we are confronted to multi-class classification among "other", "tremor" and "dyskinesia". Moreover, the issue of imbalance learning appeared rapidly, as dyskinesia crises and tremors are minority events. To partially deal with it, we oversampled the minority events thanks to a bigger overlap than for the rest of the signal. The sliding window moves slowly on the Parkinson's event to cover more intervals. In the end, the oversampled minority events are not exactly the same but can be very similar. In Figure 56 we illustrate the procedure. The same goes for

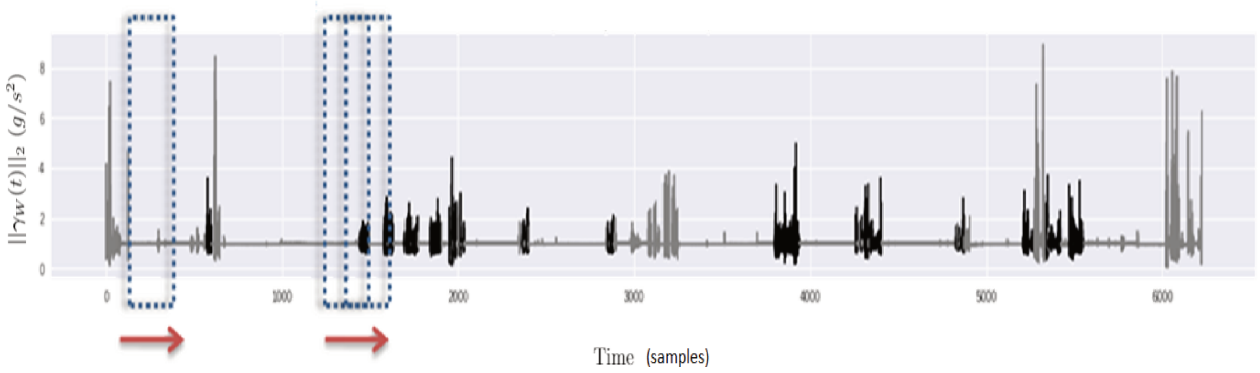


Figure 56: Sliding window with overlap (annotated events in black)

the HAPT dataset, using the annotation directly given for each recorded inertial signals with a sliding window size equals to 2.56 seconds with 50% overlap.

Finally we have access for each labelled window to the acceleration and angular velocity in three dimensions recorded in the body frame of the device. In the following section, we describe how we use this set of labelled inertial intervals for the two considered multi-class classification tasks: Activity Recognition and Parkinson's event detection.

2 Architecture description and contribution of the channels

In this section we describe the construction of the architecture of our neural network which was done iteratively. At first we used in convolutional neural networks the preprocessed inertial data given by the sliding window (Section VI.2.2). This first model presented encouraging results that we have iteratively improved with

other features (see Section VI.2.3), especially the innovative ones calculated from topological data analysis in Section VI.2.4.

2.1 Learning protocol

For Parkinson's event detection, our methodology has to be robust enough to address two problematics. Firstly, the generalization of the detection of events of partially known patients. Secondly, the generalization of the detection of events to unknown patients, while lowering the amount of false positives. The notion of knowledge refers here to the learning process, inevitably based on some of the annotated dyskinesia crises. Both issues modeled our methodology of training, validation and testing. As a consequence, a protocol of cross-validation was implemented in order to validate the robustness of our models, even with little information to train on. We illustrate our approach in Figure 57, inspired by the leave-one-out protocol. Each step of our cross-validation consisted in setting

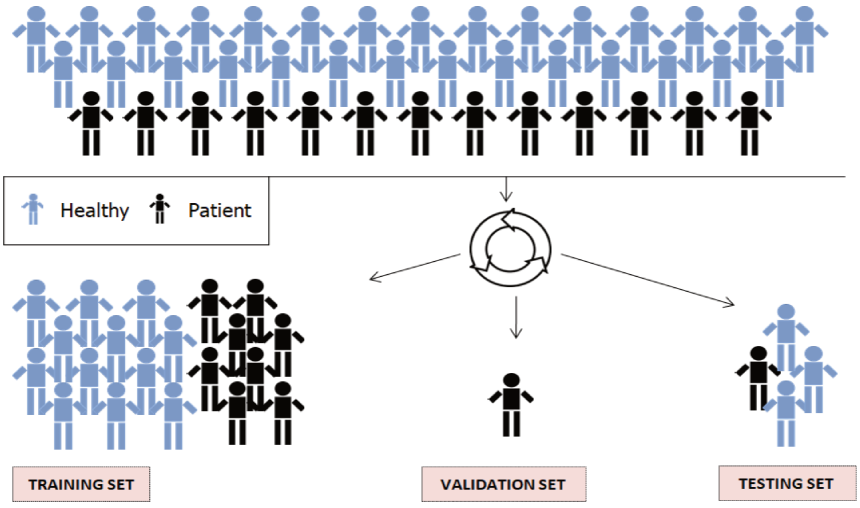


Figure 57: Train-test-validation splits for Parkinson's event detection.

aside one patient out of the 13, along with recordings of 3 healthy volunteers people matching patients ages for validation. On the other side, our training set was composed of the rest of the healthy records, mixed with all except one event for each remaining parkinsonian patient. The remaining events constituted our validation set used to stop the learning procedure when the network begins to overfit. This process of selection was repeated for each parkinsonian patient. Finally, each step was characterized by a training, testing and validation accuracies. Concerning activity recognition, one would notice that the HAPT Dataset is inherently separated between a training and a testing set, both constituted of different individuals. We use the traditional 10-fold cross validation to tune the hyperparameters of our machine learning model.

Each inertial time serie has been preprocessed using a second order high-pass Butterworth filter of 0.5 Hz, inspired by the literature [65]. The scaling consisted then in a reduction to a range of -1 to 1, before translating the mean to zero without modifying the standard deviation. The preprocessing values used are based on the training set, and spread to the testing and validation sets, to keep a training mean of zero for each input.

2.2 Convolutional Neural Networks

Although Convolutional Neural Networks were used at first for images classification, their success has motivated research to adapt them to other types of data. Keeping the notations introduced in Section 3.3, we can consider the windows of the 3D time series as images with pixels indicating the intensity of the signal values. The higher is the signal data, the higher is the pixel value. In addition, an alternative version of 2D CNNs called 1D Convolutional Neural Networks have been developed for 1D signals.

We ended up with a multi-channels network. The underlying idea is to create convolutional channels depending on their signal input, and merging them in a unique dense network aiming to exploit the concatenated feature maps. Thus we built a 2D Convolutional Channel for the triaxial acceleration, as well as for the triaxial angular velocities; a 1D Convolutional Channel for both the normed acceleration and angular velocity. The outputs of these 4 convolutional channels are then merged in a fully connected dense network. The convolutional channels included two layers of convolution with pooling. An outreach representation of the model is given in Figure 58. The final output is a vector of dimension 3 in

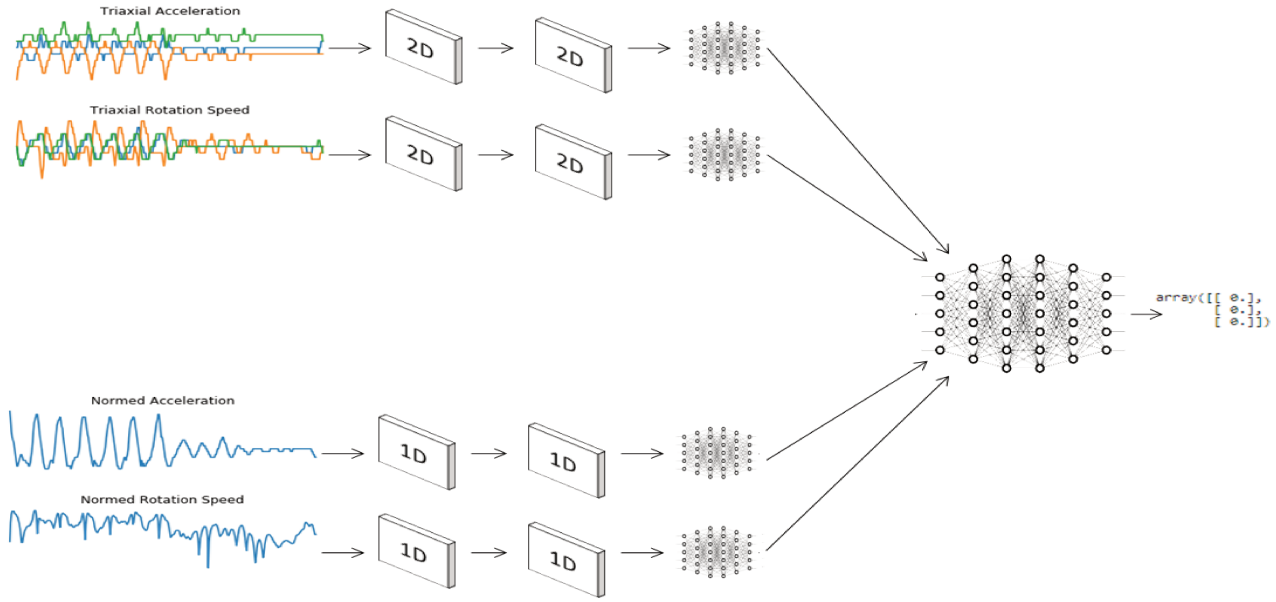


Figure 58: Multi-Channels Network Representation.

the case of Parkinson's event detection as the task is to classify the considered window into "other", "tremor" and "dyskinesia". For the HAPT dataset, the output vector dimension is equal to the number of activities.

The hyperparameters of the model to be tuned by are the number of kernels for each convolutional layer, their size and the type of pooling to apply. For fully connected layers, the number of layers and number of neurons is crucial. Also for the learning phase, the batch size has to be fixed but the number of epochs is chosen regarding the performances on the validation set during the learning. For Parkinson's event detection we have hundred of thousands of labelled windows. The training procedure is heavy and it is time consuming to tune the hyperparameters for this task. We decided to focus our attention on the HAPT dataset to overtake this issue. As we saw in Section VI.1, it is relevant to consider this multi-classification problem to tune the architecture of the model and then compute the optimization through Backpropagation with Parkinson's inputs. Indeed, the HAPT dataset also deals with inertial data for specific movement detection

and has the same imbalance in classes proportion, but with much smaller overall number of tagged intervals.

The Table 21 represents the performances of the tuned model on the testing set of the HAPT dataset. We specified here the mean accuracy on both basic activities and postural transitions as these last are underrepresented. Indeed, the imbalance learning characteristic is one of the main reasons we chose to consider this dataset.

	Basic Activities	**Postural Transitions**
Mean accuracy	97.2%	86.35%

Table 21: Testing scores for HAPT dataset with CNN channels model.

Our model achieved more than 97.2% mean accuracy on the basic activities that is achieved by the best result of the literature for this dataset. It also presented good performances for the transition activities with more than 86% accuracy that most of the paper in the literature do not even consider in their validation. Still, these classes are more difficult to detect as they are in minority compared to basic ones.

For the Parkinson's event detection task, the model quickly achieved 100% performance for the "tremor" class for both wrist and ankle recordings. As the results in Table 22 we focused on the results including "dyskinesia" and "other" classes.

	Sensitivity	**Specificity**
Ankle	41.2%	95.8%
Wrist	21%	90.6%

Table 22: Testing scores for dyskinesia detection with CNN channels model.

The difficulty behind the detection of dyskinesia crises is the difference between the controlled environment at the hospital and the home environment of those patients, as well as the generalization to new patients. Knowing the validation accuracy had a mean of 99.8% for every step of our cross-validation protocol, we may observe in Figure 59 the intrinsic difficulty of generalizing the dyskinesia detection of the studied patients to a patient for which the model was blind. Even if we use a validation set to stop the learning phase when the validation performance starts decreasing, it seems that the model still leads to a kind of overfitting.

In the following we add new channels in order to improve the results of this model keeping the same strategy: tuning the hyperparameters on the HAPT dataset and then compute the optimization through backpropagation with Parkinson's inputs.

2.3 Handcrafted Features Channel

Inspired by the literature concerning the detection of symptoms of Parkinson's patients [61], we built statistical features such as mean, correlation, kurtosis etc. We also computed signal theory features such as spectral power, harmonics, Fourier transform, Wavelet transform etc. We computed features from the freezing index [54], which has been proved to be strongly correlated to the presence of freezing of gait. We computed features based on the chaos theory and

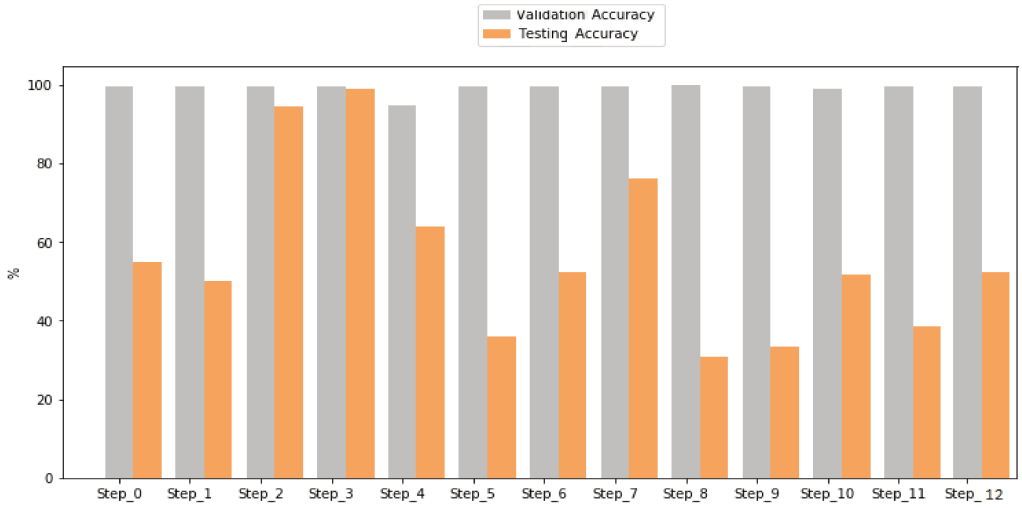


Figure 59: Testing and validation accuracies during cross validation with CNN channels model for dyskinesia detection on the wrist.

more specifically, the evolution of dynamic systems such as the Lyapunov and Hurst coefficients [66, 27, 67]. Our construction was mainly oriented towards what dyskinesia were inherently: unintended movements.

A new channel composed by a dense network of handcrafted features is thus added to the previous architecture. It leads to the following model described in Figure 60. Table 23 represents the performances of the tuned model on the

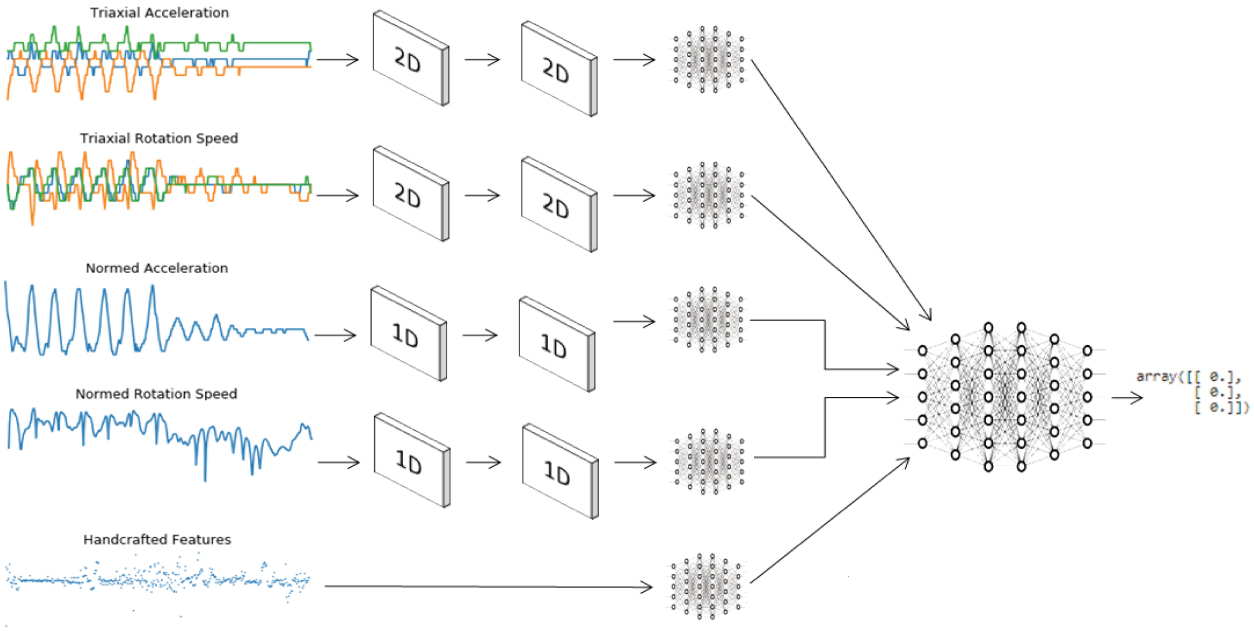


Figure 60: Multi-Channels Network Representation.

testing set of the HAPT dataset. This tuned model has also been tested for the dyskinesia crises detection and we represented the results in Table 24.

	Basic Activities	**Postural Transitions**
Mean accuracy	97.3%	88.15%

Table 23: Testing scores for HAPT dataset with CNN and handcrafted features channels model.

	Sensitivity	Specificity
Ankle	41.3%	96.5%
Wrist	24.7%	91.1%

Table 24: Testing scores for dyskinesia detection with CNN and handcrafted features channels model.

We observe an improvement for the activity recognition and for the detection of dyskinesia. We can also see the same phenomenon as before, namely that the detection of dyskinesias crises is more difficult for wrist recordings. this can be explained by the variety of movement which is greater at the wrist than at the ankle. Indeed it is rarer to perform erratic movements like dyskinesia crises with the leg, while on the wrist, tooth brushing is very similar to what can be a dyskinesia regarding the inertial signals.

2.4 Topological Data Analysis Channel

We also considered the use of topological data analysis (TDA) to improve our characterization of dyskinesia crises. It is a recent field introduced in Section II.3.4 that emerged from various works in applied topology and computational geometry, aiming at providing well-founded mathematical, statistical and algorithmic methods to exploit the topological and underlying geometric structures in data, that is often represented with point cloud. Applying this newly spreading method to our problematic seemed all the more relevant when looking at the 3D representation of angular velocity for one tremor event (Figure 61) and dyskinesia event (Figure 62). In practice, the 3 dimensions raw signals are converted

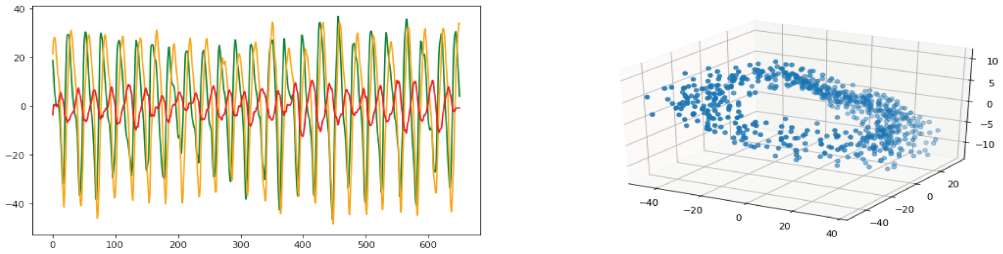


Figure 61: Angular Velocity during one tremor event.

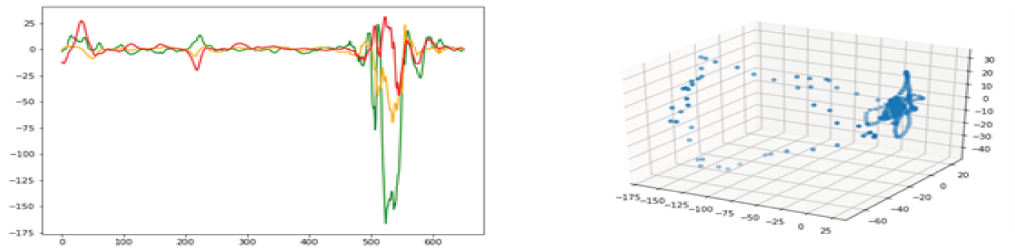


Figure 62: Angular Velocity during one dyskinesia event.

into point cloud with the sliding window approach. Then, thanks to GHUDI software (`https://gudhi.inria.fr/` and `https://github.com/GUDHI`) developed by Inria, we compute the Vietoris-Rips filtrations introduced in II.3.4. It allows to compute the persistence diagrams (see Section II.3.4 and Figure 63). One problem is that one window does not ensure to compute the same amount of topological

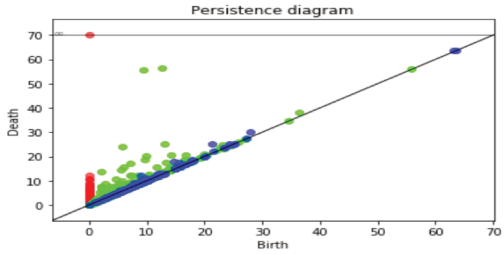


Figure 63: Example of one persistence diagram from Vietoris-Rips filtrations.

features. To overtake this issue we compute the persistence landscapes (see Section 3.4 and Figure 64) that are a convenient representation for machine learning algorithm as they can be used as 1D signals. However, we observed no real improvement considering new convolutional channels from the landscapes. A possible reason could be the redundancy of the landscapes. We rather developed a new method, based on the construction of persistence silhouettes, equivalent to a linear combination of the previously built landscapes. Tuning the weights of this linear combination is a difficult task. Thus we create a network aiming at finding the best coefficients for each landscapes, converting a set of landscapes into a 1D signal. Finally we add a 1D convolutional channel on the silhouette to the previous architecture presented in Figure 60.

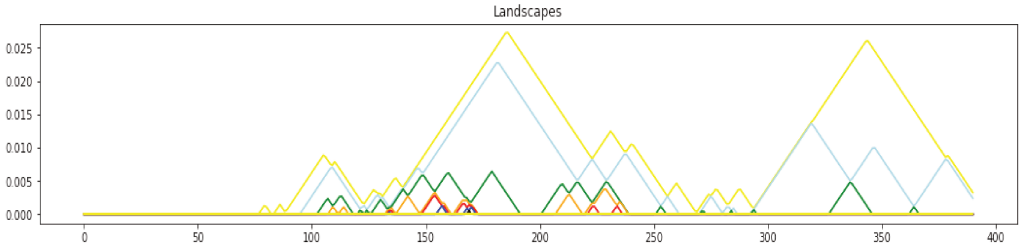


Figure 64: Example of persistence landscapes.

The Table 25 represents the performances of the tuned model on the testing set of the HAPT dataset.

	Basic Activities	**Postural Transitions**
Mean accuracy	98.4%	90.2%

Table 25: Testing scores for HAPT dataset with CNN, handcrafted features and TDA channels model.

As we can see, the addition of this channel enabled us to greatly improve our score. Concerning the results obtained on the majority classes, we achieve an accuracy score of 98.4% which put us above state of the art performances using an approach based on inertial data. Comparing this model with the previous one implies that TDA improved also the mean accuracy of the minority classes by more than 2%, finally achieving 90.2% of mean accuracy whereas most of the papers in the literature do not consider these classes. We implemented the same kind of model for the parkinsonian problematic, whose results are presented in Table 26.

	Sensitivity	**Specificity**
Ankle	41.5%	96.9%
Wrist	31.4%	90.8%

Table 26: Testing scores for dyskinesia detection with CNN, handcrafted features and TDA channels model.

We observed that the TDA channel has also increased the performance of detection of dyskinesia crises. In particular, we have a gain of almost 7% for sensitivity score on the wrist without decreasing too much the specificity score.

3 Parkinson's statistics for home recordings

To quantify our ability to detect dyskinesia crises at home without ground truth, our statistical results are indicative. However, one main goal had to be fulfilled: reducing the amount of false positives on healthy people. For an entire recording, we apply the same sliding window approach with overlap presented in Section VI.1.2. Hence, one data point is extracted several times and is associated to several a posteriori probabilities predictions of the neural network. They are averaged to compute the final prediction. We assign the predicted class that corresponds to the highest mean of the probabilities. The Table 27 represents the statistics of detected events in home recordings.

	Wrist	**Ankle**
Hospital	14.3%	13.5%
Home	3.6%	9.5%
Healthy	0.2%	0.1%

Table 27: Parkinson's event rates.

It is promising to detect tremors and dyskinesias in home recordings for patients and only little for controls (healthy people). According to these results, there are many more events in hospitals than at home. Indeed the protocol at the hospital is programmed to induce dyskinesia or tremors in order to record them.

4 Contributions

In this chapter we have presented an architecture of neural networks constructed and tuned iteratively for activity recognition (HAPT dataset) and for the detection of Parkinson's events. The use of the HAPT dataset is justified by the fact that it is a multi-class classification problem on specific movement detection with inertial data and it has the same characteristic as our Parkinson's database: it is unbalanced. Indeed, we have access to a very large number of home control records (healthy people) and very few parkinsonian events tagged in the hospital. The HAPT dataset is composed of a set of basics activities which represents almost 95% of the global dataset (completed by the postural transitions which are therefore the underrepresented activities). The small size of the HAPT dataset compared to the parkinsonian dataset allowed us to test several architectures and quickly tuned the structure of the convolutional networks and the dense networks present in our model. The final architecture is then used for the backpropagation learning phase on the parkinsonian data.

The architecture of our network is composed of 1D convolutional network for the inertial norms and 2D convolutional networks for the three axial acceleration

and angular speed, extracted with a sliding window (with overlaps). We also compute handcrafted features from chaos theory and signal processing techniques creating a dense network channel. Finally we use the topological data analysis theory introduced in II.3.4 to compute silhouettes that can be considered as 1D signal in our network. All the channels are merged in a dense network providing a posteriori probabilities for the considered multi class classification. For each sample, they are averaged (considering all the crossing windows) providing the final prediction.

At each iteration we estimated the performances of the model which allows us to assess their contribution. The results are represented in Table 28 for the activity recognition with scores reaching the performances of state of the art methods, and above. In Table 29 and Table 30 are described the results for respectively the dyskinesia detection on ankle and wrist recordings.

	Basic Activities	**Postural Transitions**
CNNs	97.2%	86.35%
+ Handcrafted Features	97.3%	88.15%
+ TDA	98.4%	90.2%

Table 28: Mean accuracy on the testing HAPT set.

	Sensitivity	**Specificity**
CNNs	41.2%	95.8%
+ Handcrafted Features	41.3%	96.5%
+ TDA	41.5%	96.9%

Table 29: Testing scores for dyskinesia detection on ankle recordings.

	Sensitivity	**Specificity**
CNNs	21%	90.6%
+ Handcrafted Features	24.7%	91.1%
+ TDA	31.4%	90.8%

Table 30: Testing scores for dyskinesia detection on wrist recordings.

Thus this architecture is modular and allows adding relevant channels which improve the results. In particular, the innovative channels TDA has greatly improved the sensitivity for the detection of dyskinesia on the wrist. In addition, the model achieved 100% accuracy for tremor detection task.

This final model has been applied on home recordings of patients and controls who are healthy people (see Table 31). It is promising to detect more Parkinson's event for patients recordings but as we do not have the ground truth is it hard to conclude. Still, our model detects false positive events that may be eliminated with new channels in a future work.

	Wrist	**Ankle**
Hospital	14.3%	13.5%
Home	3.6%	9.5%
Healthy	0.2%	0.1%

Table 31: Parkinson's event rates.

Part VII

General Conclusions

In this chapter, we first describe the general context of the thesis, the main issues that arise from it and the developed algorithms that propose methods to overtake them with machine learning approaches (Section VII.1). In Section VII.2 we describe two industrial applications which were inspired by our work but requiring some specific modifications or adaptations. Finally in Section VII.3, we introduce the general principles of domain adaptation which has shown interesting results for activity recognition.

1 Summary of main contributions with machine learning approaches

In this section, we briefly describe the main works of this thesis. The challenges were guided by the medical use of the inertial system Actimyo in clinical studies for Duchenne Muscular Dystrophy (DMD) and Parkinson's disease. The device has been developed to be worn at the wrist and the ankle by patients suffering from movement disorders in order to evaluate their health condition along clinical trials. The DMD clinical variables proposed by Sysnav are based on the trajectory and the activity recognition (see Section VII.1.2) of strides in ankle home recordings. The developed algorithms for the calculation of these variables depend initially on the detection and precise segmentation of the stride indices in the records (Section VII.1.1). For Parkinson, we aim to detect movements characteristic of the disease, that is to say tremors and dyskinesia crises, in both wrist and ankle records (Section 1.3). All the solutions proposed can hardly be relied on robust deterministic models from the inertial data. This is why the proposed algorithms are all based on a prediction functions built beforehand by statistical supervised learning procedures dedicated to the considered task.

1.1 Stride detection from candidate intervals extraction

Stride detection with precise segmentation of the start and end indices is a key step in our thesis. Indeed, the trajectory reconstruction algorithm that we introduced in Section V.1 aims to estimate the speed of the ActiMyo around these indices with a model that is valid only when the foot is on the ground. Thus, poor segmentation, for example a detected end of stride when the foot is in the air, can lead to large errors in trajectory reconstruction. On the opposite, not detecting a stride immediately causes as much error as the distance traveled by this stride.

In this thesis, we adopted an innovative approach which consists in selecting candidate stride intervals from the inertial recordings. Some of the selected intervals are true strides, others are movements of the ActiMyo device resulting for example from a manipulation by hand or from activities without strides like bicycling. The choice among the candidate stride intervals is given by a prediction function computed by a binary gradient boosting trees algorithm (stride or not stride). As the annotation is time consuming, because it has to be done manually, we have to compute relevant features for this classification task and in particular robust at any positioning of the ActiMyo sensors around the ankle. During the thesis we developed two extraction methods presented in Chapter III and Chapter IV. The last one which leads to the best performances is based on the identification of the gravity in the acceleration recorded by the ActiMyo accelerometer. It allows to project the inertial data into a terrestrial reference frame

and to compute a pseudo-speed that is used for the extraction of relevant candidate intervals and is one of the main features for the gradient boosting trees learning. ://www.overleaf.com/project/5e1200f610a3b3000132f758 This innovative approach for stride detection has improved the performance of the existing algorithm developed by Sysnav that presented good detection for classic walking strides but suffered from limitations for atypical strides (small steps, side steps, fast stairs) with only a detection rate of 50%. Our algorithm detects more than 98% of atypical strides and is also much more robust in situations where the system ActiMyo is in motion that does not correspond to strides.

1.2 Activity Recognition from the computed trajectory

We introduced in Section V.1 the principles of trajectory reconstruction for an inertial system worn at the ankle. This block developed by Sysnav takes as input the stride indices detected by our algorithm described in Chapter IV. We validated it in a motion capture room which allows a precise evaluation of the error produced for each stride length. With an error about 3 cm for walking and about 5 cm for atypical strides, we are able to compute the trajectory of a pedestrian in three dimensions, even in narrow and challenging places.

The trajectory is a key variable for recognizing the activity of detected strides. Indeed, we have access to the trajectory, in particular the altitude, but also the speed which a priori allows to differentiate running strides and stairs strides. However, we want to apply this activity recognition algorithm for children suffering from Duchenne Muscular Dystrophy. These children are at very different stages of the disease and present a wide variety of gait motion. It is difficult to develop a deterministic model. We have therefore adopted a machine learning approach aiming in particular to recognize running strides and stairs strides which are precious information in a medical context. The features computed from the trajectory and the inertial data allows the gradient boosting trees algorithm to compute a robust prediction function for healthy adult and children recordings (more than 99% success), but also for DMD patient recordings (more than 97% success).

1.3 Dyskinesia and tremor detection

The Levodopa drug aims to ease the symptoms of Parkinson's disease, such as tremors, but has a main drawback: induced dyskinesia crises. Thus, the detection of these two events is essential to measure the quality of life of patients. In this context, the ActiMyo device was used in clinical trials that we can use to build our algorithm. Faced to the variety of movement (especially on the wrist) it is impossible to simply describe the dyskinesia. We opted for a machine learning approach with a sliding window that chronologically extracts time intervals of a few seconds, and which we aim to classify among the following events: "tremors", "dyskinesia", "other".

The database build from the recordings of clinical trials is highly unbalanced (few tremors and dyskinesia annotated compared to the amount of healthy home recording) and very large (hundreds of home recordings). This last characteristic makes the tuning phase of a machine learning model very time consuming. We decided to proceed with the development of our model from the HAPT dasaset, composed by inertial recordings for activity recognition. It has the particularity of presenting the same problem of imbalance learning (with under-represented transition activities) but is much smaller. In addition, the detection of tremors

and dyskinesia can be considered as recognition of activity among all other daily activities.

We ended up with a neural network composed by several relevant channels merged in a dense network, the overall architecture is described in Chapter VI. During the procedure, we added iteratively the channels in order to measure their respective contribution. The convolutional networks for the inertial time series and the handcrafted features channel inspired by signal processing techniques allowed to achieve results better than the literature for the HAPT dataset. In the end, we build a topological data analysis (TDA) channel that remarkably improved the scores, especially for the underrepresented classes. The principles of TDA introduced in Section Topological Data Analysis ChannelII.3.4 aims to capture geometric structures into point cloud. In our case, we applied this techniques on angular velocity data extracted by the sliding window. the validation of the parkinsonian model has shown 100% success for the tremor detection but limitations for the generalization of dyskinesia detection on recordings of patients who have not been used for the learning. On the other hand, for patients already seen in the learning base, we obtain around 98% of accuracy. Even if we do not have the ground truth for home recordings, we tested false positives on control recordings (healthy people). It is encouraging to detect less than 2 minutes of Parkinson's events throughout the day while for patients it detects several tens of minutes (which is coherent).

2 Extensions to industrial applications

In this section we describe two applications that were inspired by the work developed in this thesis. In Section VII.2.1, the algorithm for trajectory reconstruction using the stride detection of Chapter III is extended to beacon data. In Section VII.2.2, we describe the adaptations of the algorithm described in Chapter IV to online constraints in an embedded system.

2.1 Beacons for trajectory reconstruction

In Chapter III, we presented a stride detection algorithm that allows to reconstruct the pedestrian trajectory thanks Sysnav methods introduced in Section V.1. The advantage of these algorithms is that they do not require any additional infrastructure-dependent localization systems to compute the trajectory. However, it is represented in an arbitrary reference frame whose the definition of the axes is unknown in the pedestrian environment.

A beacon is a small system that continuously sends a radio signal incorporating its identifier. A device equipped with a receiver is then able to "see" the beacon when it enters its emission radius. Depending on the strength of the received signal, we can then estimate the distance between the receiving device and the beacon. Their advantage in industrial applications is that they are not expensive and can be installed quickly in any environment. However, the signals are often disturbed by the walls, machines, etc. Thus computing the trajectory only as a function of the intensity of the signals received by a triangulation method for example is not reliable.

The device ActiMyo has therefore been adapted so that it can read the signals sent by beacons. The proposed solution is to firstly compute the trajectory without using the beacon information. Then we modify the trajectory in order to correspond with the beacon data only when the strength of the received signal

is large. That is to say when we are sure that the pedestrian is very close to the transmitting beacon. In practice we consider the signals to modify the trajectory which indicate a position less than one meter from the beacon.

The proposed algorithm consists in processing the beacon data chronologically. The principle is to apply a rotation between two successively detected beacon so that the resulting trajectory fit at best with their position. Then we force the trajectory capturing the beacons to enter into the corresponding emission zone (applying as little modification as possible on the initial trajectory). This overall algorithm is currently used by an industrial Sysnav client in order to optimize worker movements between machines.

2.2 Stride detection algorithm in embedded system

The stride detection algorithm described in Chapter IV is used offline. That is to say, the ActiMyo recordings are first loaded into a cluster and then the algorithm is launched. It is therefore not suitable for online use which would consist of processing inertial signals directly when they are recorded. In many industrial applications, this functionality is requested, in particular for isolated workers, firefighters or military, whose position must be known "in live" to ensure safety.

In this context, Sysnav modified the ActiMyo device to develop an embedded system. The goal is to adapt our stride detection algorithm to launch it on this system. The principle of this algorithm remains the same: identifying the gravity in the accelerations recorded by the sensor, computing a pseudo-speed that allows to extract candidate stride intervals, then selecting among them the intervals that are true strides. The offline identification of the gravity could be directly transcribed for online algorithm. However we had to restrict the number of extracted candidate stride intervals to reduce the number of prediction function launches. In addition, the procedure for computing the features presented in Section IV.3 and the prediction of the gradient boosting trees function requires too much resources for the embedded microprocessor. We opted for the construction of a neural network with 2D Convolutional Neural Networks channels (with only two layers) for each 3D times series that need no more computation at this stage (acceleration, angular velocity, pseudo-speed). The three channels are finally merged in a last fully connected layer.

These modifications provide a good compromise between execution time and performance. This work was supported by the French Délégation Générale de l'Armement (DGA) and by ANR project TopData ANR-17-MALN-0003. It is currently used in the final solution of Sysnav competing for the MALIN challenge.

3 Future work: Domain Adaptation for Activity Recognition

The work described along this thesis has resulted in performances good enough to be used in several solutions proposed by Sysnav. However during our research we have been studying the domain adaptation by optimal transport which showed interesting results for the activity recognition. In the following we introduce this work that deserves to be studying further.

In this thesis, the work for activity recognition aims to be used in a medical context, that is to say for home recordings of patients suffering from Duchenne Muscular Dystrophy (DMD). We saw in Section V.3 that the algorithm achieved good performance except for one patient with a very particular gait motion for

climbing stairs. More generally, the database used to learn the prediction function by machine learning is mainly composed of healthy adults recordings. The performed strides are therefore not representative of the gait motion of DMD patients. As a result, The learning process can lead to a prediction function that is not entirely suitable for DMD recordings. As it is difficult to have access to annotated data for DMD recordings, we wanted to transform the learning phase regarding to the observed gait motion strides whose we need to recognize the activity.

This general problem is very present in industrial supervised learning tasks. The observations on which the prediction is applied (called target observations) can be disturbed compared to those which were used for learning (called source observations). This can be explained for example by a difference in sensors, recording conditions, etc. Consequently the decision rules learned on the source features do not work on the target features. This phenomenon is particularly observed in the image classification tasks. In Figure 65 we represent the tagged pictures taken in ideal conditions (white background, well framed). However in practice, the prediction function will be applied for images in "real conditions", that is to say not well framed and with a background that varies.

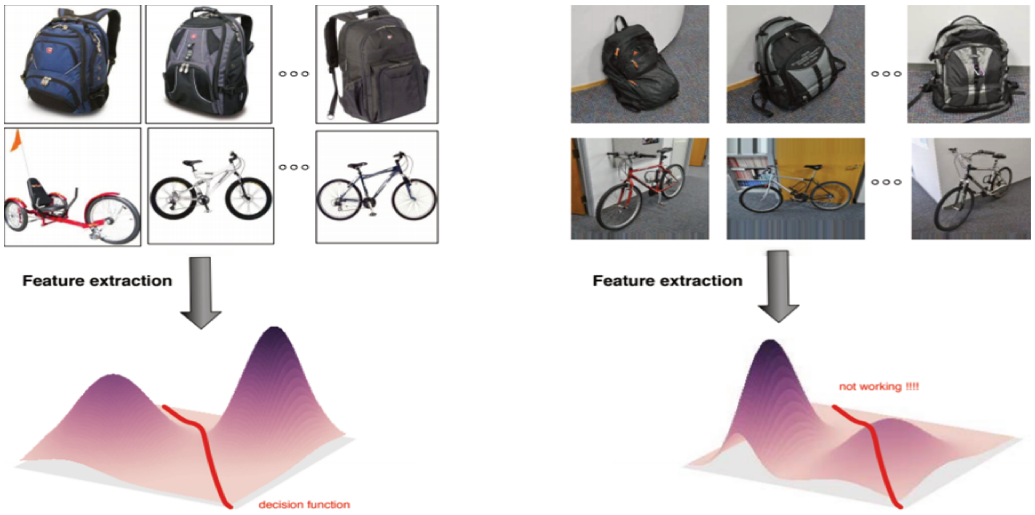


Figure 65: Example of features distortion between the source (left) and the target (right) (source: `https://remi.flamary.com/cours/tuto_otml.html`).

In our application settings for activity recognition, we built a database composed mostly of healthy adult recordings and only a few DMD recordings. For an experiment, this DMD recordings have been removed on the learning database and are considered as the target observations. The resulting prediction function performances are presented in Table 32.

	Atypical stride	**Walking**	**Running**	**Upstairs**	**Downstairs**
Score	99.6%	30.98%	58.37%	80.55%	65.20%

Table 32: Performances of the activity recognition with a source of healthy adult recordings and a target of DMD recordings.

Thus, what we observed in Section V.3 appears to be a more general problem. We cannot easily extend the prediction function learned on healthy adult recordings to DMD recordings. It therefore requires a wide variety of DMD recordings to for good performance in recognizing activity during clinical studies. However,

having access to more DMD recordings tagged in the hospital requires a medical team to perform the tests (6 minutes walk, 4 steps, 10 meters of running). In addition, we can make the assumption that the bias observed between adult recordings and DMD recordings may also be present between hospital (annotated recordings) and home (clinical study recordings). In the following, the goal is to modify the prediction function learned on the source so that it is more efficient on the target. Our research has led us to study domain adaptation through optimal transport.

Optimal transport (OT) was introduced by Monge in 1781. The goal was to transport the mines production to the factories as efficiently as possible. In our case, we want to transport the observations from the source to the target (transport T defined and taking values on Ω) while minimizing the energy required for this transport (cost function $c(\boldsymbol{x}, T(\boldsymbol{x}))$. This problem is illustrated in Figure 66 and can be expressed with the following equation:

$$T_0 = \underset{T}{\operatorname{argmin}} \int_{\Omega_s} c(\boldsymbol{x}, T(\boldsymbol{x}))d\mu(x),$$

where Ω_s and Ω_t note respectively the source and the target measurable space of features, μ_s and μ_t the marginal distributions of the source and the target. One can noticed that in this example, the labels of the source observations are

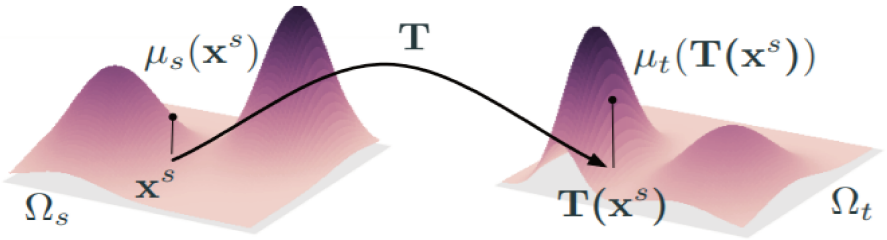


Figure 66: Example features spaces transport between the source (left) and the target (right) (source: `https://remi.flamary.com/cours/tuto_otml.html`).

not considered in the optimal transport. We believe that the labels would be a relevant information for the transport in classification task. For example, one wants to penalize the situation where two observations of different classes are transported close to each other.

3.1 Joint Distribution Optimal Transportation

Our research has brought us to the work of N. Courty et al. [59]. This paper deals with the unsupervised domain adaptation problem, where one wants to estimate a prediction function f in a given target domain without any labeled sample by exploiting the knowledge available from a source domain where labels are known. This work makes the following assumption: there exists a nonlinear transformation between the joint feature/label space distributions of the two domain $\mathcal{P}_s$ (source) and $\mathcal{P}_t$ (target) that can be estimated with optimal transport. They propose a solution of this problem that allows to recover an estimated target $\mathcal{P}_t^f = (X, f(X))$, where X denotes the features, by optimizing simultaneously the optimal coupling and f. They find a function f that predicts an ouput value given an input x (features) and that maximizes the optimal transport loss between the joint source distribution $\mathcal{P}_s$ and the estimated target joint distribution $\mathcal{P}_t^f$ depending on f. The method is denoted as JDOT for "Joint Distribution Optimal

Transport". As an example we represent in Figure 67 a resulting domain adaptation with JDOT in two dimension for a binary classification problem (in green the decision rule). One can notice that the source observations are not completely

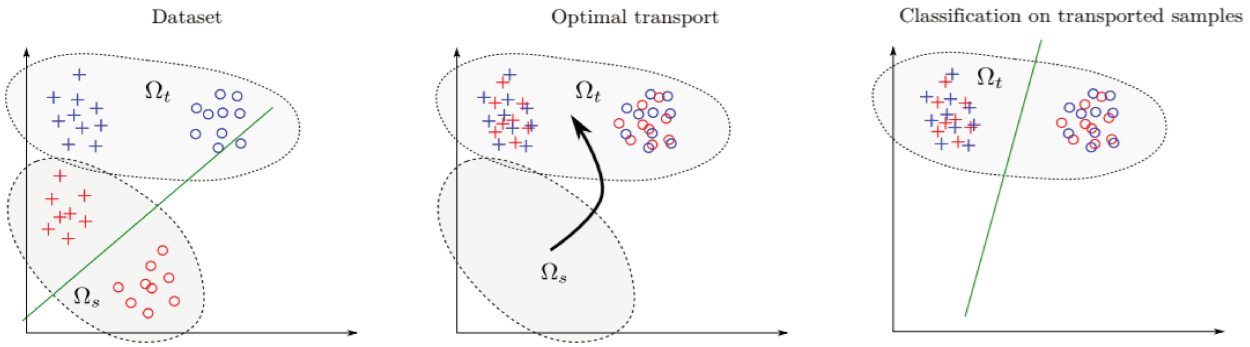


Figure 67: Example of domain adaptation with JODT in two dimensions for a binary classification problem.

superimposed on the target observations. This is due to an entropic regularisation used in the optimal transport introduced by Cuturi in [23]. It allows to divide the mass of a source observation over several target observations, reducing the overfitting risk and speeds up the computation time of the optimization process. We give an illustration in Figure 68. Note that fixing the value for penalization fac-

Figure 68: Example of optimal transport with entropic regularization in two dimensions.

tor of entropic regularization and labels penalization in JDOT is a difficult choice depending on the considered data.

We applied this work for the activity recognition learning task. We first considered the previous example of adapting adult strides to DMD strides. The results are presented in Table 33.

	Atypical strides	Walking	Running	Upstairs	Downstairs	Mean
Score without JDOT	99.6%	30.98%	58.37%	80.55%	65.20%	67.32%
Score with JDOT	91.66%	86.27%	86.97%	87.77%	85.83%	87.74%

Table 33: Performances of the activity recognition with a source of healthy adult recordings and a target of DMD recordings with and without JDOT.

The use of JDOT has therefore improved activity recognition performance on average by more than 20% and shows that it would be useful in our application framework. As another example, we considered a particular DMD patient for whom, if we exclude his strides from the learning base, we obtain the following predictions (Table 34).

	Predicted walking	Predicted running	Predicted upstairs	Predicted downstairs
Actual walking	18	13	1	0
Actual running	1	31	0	0
Actual upstairs	1	1	30	0
Actual downstairs	1	1	0	30

Table 34: Confusion matrix after learning the source and predicting on the target.

We can count 19 errors essentially between running and walking. In the following we will study if a domain adaptation between the source and the target with JDOT can improve this score. We obtain the confusion matrix presented in Table 35.

	Predicted walking	**Predicted running**	**Predicted upstairs**	**Predicted downstairs**
Actual walking	31	0	1	0
Actual running	0	29	2	1
Actual upstairs	0	0	32	0
Actual downstairs	1	3	1	27

Table 35: Confusion matrix after applying JDOT algorithm.

Thus, optimal transport allowed to achieve 9 errors from 19 errors, by removing all the errors between walking and running. However it can be noted that no error was initially made between climbing and descending stairs before transport, but here one false descent was transported on the observations of stair climbing. In addition, JDOT requires the proportion of strides in each class (namely in each activity) to be identical in the source and the target. This is one of the main drawbacks of JDOT agorithm in our application framework. Indeed we can organize the learning base for activity recognition so that it is balanced, however it is very likely that the DMD home recording on which we want to apply JDOT is not exactly composed of the same stride number of walking, running and stairs etc.

A difference in decision rule between source and target observations can be observed in many industrial applications. The algorithm for optimal transport introduced by Cuturi in 2013 with entropy regularization greatly accelerated the calculation of optimal transport. However it is still time consuming in large dimension such as our application settings.

The JDOT algorithm allows to take into account the labels of the source in the transport and has shown particularly interesting results in our framework for activity recognition (adaptation of healthy adult strides to DMD strides). However, this algorithm requires a complicated choice for the parameters of regularization and labels penalization. In addition, it requires the proportion of the classes in the target and the source to be the same. From JDOT algorithm, We started research for the estimation of classes proportion in the target (in order to adjust them on the source) and for the selection of variables based on the quality of transport and learning to improve the prediction on the target. It could also provide a dimension reduction easing the use of optimal transport algorithm in JDOT that is time consuming in large dimension. These works could be continued in the future.

Bibliography

[1] Norrdine A., Kasmi Z., and Blankenbach J. Step detection for zupt-aided inertial pedestrian navigation system using foot-mounted permanent magnet. *IEEE Sensors Journal*, 16.

[2] Z.O. Abu-Faraj, G.F. Harris, P.A. Smith, and S. Hassani. Human gait and clinical movement analysis. *Wiley Encyclopedia of Electrical and Electronics Engineering*.

[3] Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, and Lori Ziegelmeier. Persistence images: a stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18(8):1–35, 2017.

[4] Samà A. Ahlrichs C. Is 'frequency distribution' enough to detect tremor in pd patients using a wrist worn accelerometer. *Proceedings of the International Conference on Pervasive Computing Technologies for Healthcare*.

[5] Sabatini A.M. Quaternion-based strap-down integration method for applications of inertial sensing to gait analysis. *Medical and Biological Engineering and Computing*, 43.

[6] Wagstaff B. and Kelly J. Lstm-based zero-velocity detection for robust inertial navigation. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[7] Sergey Barannikov. The framed morse complex and its invariants. In *Adv. Soviet Math.*, volume 21, pages 93–115. Amer. Math. Soc., Providence, RI, 1994.

[8] Grelet M. Beaufils B., Chazal F. and Michel B. Robust pedestrian trajectory reconstruction from inertial sensor. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[9] Grelet M. Beaufils B., Chazal F. and Michel B. Stride detection for pedestrian trajectory reconstruction: a machine learning approach based on geometric patterns. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[10] Grelet M. Beaufils B., Chazal F. and Michel B. Robust stride detector from ankle-mounted inertial sensors for pedestrian navigation and activity recognition with machine learning approaches. *Sensors*, 2019.

[11] J-P Berrut and L. N. Trefethen. Barycentric lagrange interpolation. *SIAM (Society for Industrial and Applied Mathematics) Review*, 2004.

[12] Jean-Daniel Boissonnat, Frédéric Chazal, and Mariette Yvinec. *Geometric and Topological Inference*, volume 57. Cambridge University Press, 2018.

[13] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *arXiv preprint 1207.6437*.

[14] Gunnar Carlsson. Topology and data. *Bull. Amer. Math. Soc.*, 46(2):255–308.

[15] Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. Perslay: A simple and versatile neural network layer for persistence diagrams. In *Proc. AISTATS 2020*, 2020.

[16] Cohen-Steiner David Glisse Marc Guibas-Leonidas J. Chazal, Frédéric and Steve Y. Oudot. Proximity of persistence modules and their diagrams. *In Proc. 25th Annu. Symp. Comp. Geom.*, pages 237–246.

[17] De Silva Vin Chazal, Frédéric and Steve Oudot. Persistence stability for geometric complexes. *Geom. Dedicata*, pages 1–22.

[18] Frédéric Chazal, Vin de Silva, Marc Glisse, and Steve Oudot. *The structure and stability of persistence modules*. SpringerBriefs in Mathematics. Springer, 2016.

[19] Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *arXiv preprint arXiv:1710.04019*, 2017.

[20] Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[21] C.-Y. Chesneau. Magneto-inertial dead-reckoning in inhomogeneous field and indoor applications. *Automatic. Communauté Université Grenoble Alpes*.

[22] Edelsbrunner Herbert Cohen-Steiner, David and John Harer. Stability of persistence diagrams. *Discrete Comput. Geom.*, 37(1):103–120.

[23] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transportation distances. *Advances in Neural Information Processing Systems*, 26, 2013.

[24] H. P. Dikshit and P. Powar. Discrete cubic spline interpolation. *Numerische Mathematik*, 40:71–78, 1982.

[25] Meryll Dindin, Yuhei Umeda, and Frédéric Chazal. Topological data analysis for arrhythmia detection through modular neural networks. *arXiv preprint arXiv:1906.05795*, 2019.

[26] Foxlin E. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer Graphics and Applications*, 25.

[27] Ruelle D. Ciliberto S. Eckmann J.P., Kamphorst S.O. Liapunov exponents from time series. *Physical Review A*, 34(6).

[28] Herbert Edelsbrunner and John Harer. Computational topology: An introduction. *AMS*.

[29] Letscher D. Edelsbrunner, H. and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533.

[30] Schapire R.E Freund, Y. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55.

[31] J. H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38.

[32] T.; Denis S.; Grelet M.; Lilien C.; Gargaun E.; Lilien C.; Moreaux A.; Dorveaux E.; Vissiè D.; Servais L. Gasnier, E.; Gidaro. Assessment of lower limbs in fshd: The actimyo as a new outcome for home-monitorin. *Neuromuscular Disorders*.

[33] Robert Ghrist. Barcodes: The persistent topology of data. 2007.

[34] Julius Hannink, Cristian F. Pasluosta Thomas Kautz, Jens Barth, Samuel Schülein, Karl-Günter Gaßmann, Jochen Klucken, and Bjoern M. Eskofier. Stride length estimation with deep learning. *IEEE EMBS*, 2017.

[35] R.; Friedman J. Hastie, T.; Tibshirani. The elements of statistical learning. *Springer New York*.

[36] A. Hatcher. Algebraic topology. *Cambridge Univ. Press*.

[37] Ngoc-Huynh Ho, Phuc Huu Truong, and Gu-Min Jeong. Step-detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone. *Sensors*, 2016.

[38] Christoph D Hofer, Roland Kwitt, and Marc Niethammer. Learning representations of persistence barcodes. *Journal of Machine Learning Research*, 20(126):1–45, 2019.

[39] Jankovic J. Parkinson's disease : clinical features and diagnosis. *Journal of Neural Neurosurg Psychiatry*.

[40] Rantakokko J., Emilsson E., Stromback P., and Rydell J. Scenario-based evaluations of high-accuracy personal positioning systems. *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*.

[41] Swzyslo S.and Schroeder J., Galler S., and Kaiser T. Hybrid localization using uwb and inertial sensors. *Proceedings of the IEEE International Conference on Ultra-Wideband (ICUWB)*.

[42] Carrera J.-L., Zhao Z., Braun T., and Li Z. A real-time indoor tracking system by fusing inertial sensor, radio signal and floor plan. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[43] Abdulrahim K., Moore T., Hide C., and Hill C. Understanding the performance of zero velocity updates in mems-based pedestrian navigation. *International Journal of Advancements in Computing Technology*, 5.

[44] Gielen S.C. Keijsers N.L., Horstink M.W. Ambulatory motor assessment in parkinson's disease. *Movement Disorders*.

[45] Gielen S.C. Keijsers N.L., Horstink M.W. Automatic assessment of levodopa-induced dyskinesias in daily life by neural networks. *Movement Disorders*.

[46] A.; Moraux A.; Dorveaux E.; Annoussamy M.; Gasnier E.; Hogrel J.-Y.; Voit T.; Vissière D Le Moing, A.-G.; Seferian and L. Servais. A movement monitor based on magneto-inertial sensors for non-ambulant patients with duchenne muscular dystrophy: A pilot study in controlled environment. *Instute of Myology*.

[47] Bengio Y. LeCun Y., Bottou L. and Haffner P. Gradient-based learning applied to document recognition. *IEEE Communications magazine*, 27:41–46.

[48] Boser B. Denker J. Graf H. Guyon I. Henderson D. Howard R. LeCun Y., Jackel L. and Hubbard W. Handwritten digit recognition : Applications of neural networks chipsand automatic learning. *Proceedings of the IEEE*, 86:2278–2324.

[49] Grundfest W. LeMoyne R., Mastroianni T. Wireless accelerometer configuration for monitoring parkinson's disease hand tremor. *Advances in Parkinson's Disease*, 02.

[50] Xuan M. Gu Q. Xu X. Kong D. Zhang M. Long D., Wang J. Automatic classification of early parkinson's disease with multi-modal mr imaging. *PloS one*, 07.

[51] Ren M., Pan K., Liu Y., Guo H., Zhang X., and Wang P. A novel pedestrian navigation algorithm for a foot-mounted inertial-sensor-based system. *Sensors*, 16.

[52] Möller J.C. Stiasny-Koslter K. Eggert K.M. Krüger H.P Ellgring H. Macht M., Kaussner Y. Predictors of freezing in parkinson's disease : A survey of 6620 patients. *Movement Disorders*.

[53] A.M. Mannini, A.; Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10.

[54] Calatroni A. Gazit E. Hausdorff J.M. Tröster G. Mazilu S., Blanke U. The role of wrist-mounted inertial sensors in detecting gait freeze episodes in parkinson's disease. *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*.

[55] Tröster G. Mazilu S., Blanke U. Gait, wrist, and sensors : Detecting freezing of gait in parkinson's disease from wrist movement. *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*.

[56] Tarvainen M. Karjalainen P. Iudina-Vassel I. Airaksinen O. Karjalainen P.A. Meigal A.I., Rissanen S. Analysis of surface emg signal morphology in parkinson's disease. *Physiological Measurement*.

[57] Ribeiro M.I. Kalman and extended kalman filters: Concept, derivation and properties.

[58] Castaneda N. and Lamy-Perbal S. An improved shoe-mounted inertial navigation system. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*.

[59] Amaury Habrard Alain Rakotomamonjy Nicolas Courty, Rémi Flamary. Joint distribution optimal transportation for domain adaptation. *NIPS Los Angeles*.

[60] Savage P. Stapdown analytics. *Strapdown Associates*.

[61] Sankar R. Perumal Shyam V. Gait and tremor assessment for patients with parkinson's disease using wearable sensors. *The Korean Institute of Communications and Information Sciences*.

[62] R. Poppe. A survey on vision-based human action recognitiont. *Image and Vision Computingy*.

[63] Brokaw E.B. Mera T.O. Mari-Z.K. Burack M.A. Pulliam C.L., Heldman D.A. Continuous assessment of levodopa response in parkinson's disease using wearable motion sensors. *Transactions on Biomedical Engineering*.

[64] S.; Ali H. Rastegari, E.; Azizian. Machine learning and similarity network approaches to support automatic classification of parkinson's diseases using accelerometer-based gait analysis. *Proceedings of the 52nd Hawaii International Conference on System Sciences*.

[65] Wang L. Raza S.B., Patterson R.P. Filtering respiration and low-frequency movement artefacts from the cardiogenic electrical impedance signal. *Medical and Biological Engineering and Computing*.

[66] Moorman J.R. Richman R. Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology-Heart and Circulatory Physiology*, 278(6).

[67] De Luca C.J. Rosenstein M.T., Collins J.J. A practical method for calculating largest lyapunov exponents from small data sets. *Physica D: Nonlinear Phenomena*, 65(1).

[68] Hinton G.E. Williams R.J. Rumelhart, D.E. Learning internal representations by error propagations. *Parallel distributed processing: explorations in the microstructure of cognition*, 1:138–362.

[69] Tedesco S., Sica M., Ancillao A., Timmons S., Barton J., and O' Flynn B. Accuracy of consumer-level and research-grade activity trackers in ambulatory settings in older adults. *PLOS ONE*, 14.

[70] R. Hamila M. Gabbouj S. Kiranyaz, T. Ince. Convolutional neural networks for patient-specific ecg classification. *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*.

[71] Vingerhoets F. Burkhard P. Blanc Y. Dehollain C. Aminian K. Salarian A., Russmann H. An ambulatory system to quantify bradykinesia and tremor in parkinson's disease. *Proceedings of the International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine*.

[72] Romagosa J. Rodriguez-Martin D. Català A. Cabestany J. Pérez-Martinez D.A. Rodriguez-Molinero A. Samà A., Pérez-Lopez C. Dyskinesia and motor state detection in parkinson's disease patients with a single movement sensor. *Proceedings of the International IEEE EMBS*.

[73] Y. Freund; R.E. Schapire. Experiments with a new boosting algorithm. *Machine Learning : Proceedings of the thirteenth International Conference*.

[74] Bamberg S.J., Benbasat A.Y., Scarborough D.M., Krebs E.E., and Paradiso J.A. Gait analysis using a shoe-integrated wireless sensor system. *IEEE transactions on information technology in biomedicine*, 21.

[75] T.; van Laerhoven K.; Schiele-B. Stikic, M.; Huynh. Adl recognition based on the combination of rfid and accelerometer sensing. *Proceedings of the 2nd International Conference on Pervasive Computing Technologies for Healthcare*.

[76] Park S.Y., Ju H., and Park C.G. Actions for military drill using foot-mounted imu. *Proceedings of the Indoor Positioning and Indoor Navigation (IPIN)*.

[77] Walder U., Bernoulli T., and Wang P. Context-adaptive algorithms to improve indoor positioning with inertial sensors. *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*.

[78] V.; Kulic D. Um, T.T.; Babakeshizadeh. Exercise motion classification from large-scale wearable sensor data using convolutional neural networks. *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*.

[79] Yuhei Umeda. Time series classification via topological data analysis. *Transactions of the Japanese Society for Artificial Intelligence*, 32(3):D–G72_1, 2017.

[80] Renaudin V. Uwb and mems based indoor navigation. *Journal of Navigation*, 32.

[81] V.N. Vapnik. The nature of statistical learning theory. *Springer-Verlag*.

[82] M.; Dorveaux E.; Jouy A.; Grelet M. Vissiere, D.; Hillion. Method for estimating the movement of a pedestrian. *U.S. Patent Application 15/766,296*.

[83] Tian X., Chen J., Han Y., Shang J., and Li N. A novel zero velocity interval detection algorithm for self-contained pedestrian navigation system with inertial sensors. *Sensors*, 16.

[84] Chen Y. and Kobayashi H. Signal strength based indoor geolocation. *Proceedings of the IEEE International Conference on Communications*.

[85] M.N.; San P.P.; Li X.L.; Krishnaswamy S. Yang, J.B.; Nguyen. Deep convolutional neural networks on multichannel time series for human activity recognition. *Proceedings of the IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence*.

[86] A. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete Comput. Geom.*, 32(2):249–274.