Laboratory of Movement analysis and Measurement- EPFL

Final Report

Algorithm for detection of walking on stairs using data recorded with a low-back worn inertial measurement unit

Student
Paco Mermoud

Professor
Kamiar Aminian

Superviser
Dr. Anisoara Ionescu

Abstract

This paper explores two classification approaches to detect subtle gait changes while walking on a flat plane and when walking on stairs. This is done by using machine learning and an inertial sensor worn close to the body's CoM (lower back/ waist). Walking on stairs represents a challenging bio-mechanical effort compared to walking on a flat surface. Its detection can help prevent fall injuries in people with physical impairment by identifying their ability to walk on stairs and potential risk areas. The first approach takes advantage of feature selection using three traditional filter methods; T-test (TT), Maximum Relevance Minimum Redundancy (MRMR) and chi square test (CST). The second approach takes advantage of feature reduction using a dimensionality reduction algorithm; principal component analysis (PCA). The resulting models of the two approaches were trained and tested using a single inertial measurement unit worn in the lower back from the dataset of Luo and al. and the K-folds and leave-one-out cross-validation algorithms.

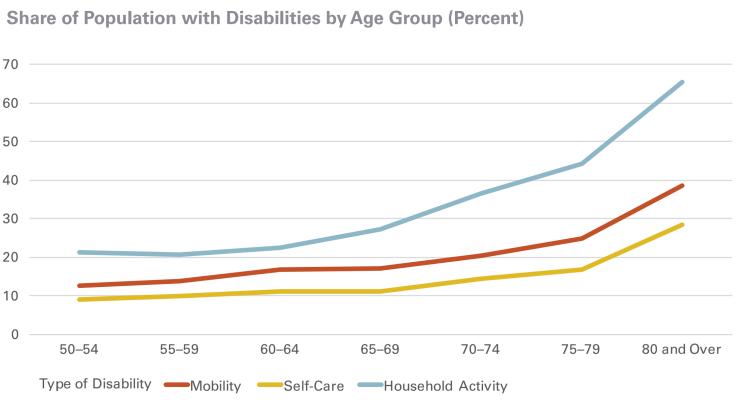
January 14, 2022

Contents

1 Introduction

The rise of machine learning has benefited many fields including healthcare. Gait detection and analysis have been developed and applied to many areas such as sports, security and ergonomics with promising results. One specific application concerning the detection of particularly energy-intensive areas which is a major issue for people with reduced mobility. In fact, walking on stairs represent one of the most challenging bio-mechanical effort for this part of the population and comport a high risk of fall injuries. Indeed, a significant amount of the people presenting a locomotion disability are the elderly people 1.0.1 and this number may even increase as life expectancy increases. Therefore, the identification of the walking ability and potential hazardous location have the potential to significantly reduce the number of present and future accidents.

One good step toward this goal is the walking detection of stairs and flat ground using only a small and non-invasive device as an IMU.



Source: JCHS tabulations of University of Michigan, 2014 Health and Retirement Survey.

Figure 1.0.1

2 Methodology

This work used the data of Luo and al. [14] which regroups signals of several inertial measurement units (IMUs) worn in different locations, used in different scenarios with several participants. The dataset is first analysed and its main characteristics are described.

Then the feature extraction of every signal of the dataset is performed for each scenario and participants with two different time windows of 1 [s] and 6,4 [s]. Every feature used in the context of this work are explained and defined.

Subsequently, the application of the first approach (i.e. feature selection) is described. The three filter methods used for ranking were the T-test (TT), the Maximum Relevance Minimum Redundancy (MRMR) and the chi square test (CST). Their influences on the rank of each feature is explained. The methodology to select the optimised number of features is presented.

Thereafter, the application of the second approach (i.e. feature reduction) is described and the principle component analysis (PCA) algorithm is explained step-by-step.

Finally several models are created with the MATLAB Classification Learner App using the features of the two previous approaches. Each model was tested using two different cross-validation algorithms called K-fold and Leave-one-out method.

2.1 Dataset

The dataset of Luo and al. was developed to advance machine learning for the recognition of small, but significant changes in gait mechanics. The data was collected with wearable motion sensors attached at several locations of the participant's body. The participants walked on different terrains and at varying inclines such as grass, slopes, steps or flat surfaces.

2.1.1 Participant

The dataset regrouped thirty healthy, young participants with no reported fall injuries in the past two years. Their anthropometry information is provided in tab. 2.1.1.

Participant	Age	Sex	Height (cm)	Body mass (kg)
1	28	F	154.5	49.1
2	24	F	158.6	54.1
3	22	F	167	53.6
4	22	F	166	56
5	23	F	168.2	61.4
6	33	M	175	99
7	27	M	184	75.3
8	18	M	187	82.3
9	22	F	162.1	53.6
10	19	F	162	61.7
11	28	M	180	70.4
12	18	M	177.9	81
13	22	F	174.2	58.6
14	19	F	66.5	67.3
15	19	M	181	72.4
16	31	M	176	101.2
17	19	F	173	73.9
18	30	M	165.4	82.9
19	32	F	165	53
20	22	F	167.1	74.8
21	19	M	169	73.9
22	22	M	178.5	80.2
23	24	F	179.6	61.6
24	26	F	174.6	62.5
25	22	F	157	61.6
26	22	M	175.6	66
27	22	M	192.7	85.9
28	22	M	180	91.1
29	26	M	172	78.1
30	22	M	188	84.4
Summary	23.5 (4.2)	15M, 15F	169.3 (21.5)	70.9 (13.9)

Figure 2.1.1: 30 participants with their anthropometry informations

2.1.2 Scenarios

Each participant was instructed to walk at their normal pace in nine different scenarios (see tab. 2.1.4):

- Flat even
- Up stairs
- Down stairs
- Slope up
- Slope down
- Grass
- Banked left
- Banked right
- Cobble stone

Each scenario was performed six times per participant with three additional calibration trials totalling fifty seven trials (see tab. 2.1.2).

Trial number (#)	Walking surface condition	Sample duration (s) Mean (standard deviation)
1–3	Calibration (CALIB)	19.29 (3.14)
4–9	Flat even (FE)	13.55 (2.19)
10–15	Cobble stone (CS)	16.12 (1.93)
16,18,20,22,24,26	Upstairs (StrU)	12.48 (1.17)
17,19,21,23,25,27	Downstairs (StrD)	11.84 (1.42)
28,30,32,34,36,38	Slope up (SlpU)	22.70 (1.89)
29,31,33,35,37,39	Slope down (SlpD)	22.77 (2.22)
40,42,44,46,48,50	Bank left (BnkL)	16.06 (1.90)
41,43,45,47,49,51	Bank right (BnkR)	16.29 (1.67)
52–57	Grass (GR)	14.48 (1.52)

Figure 2.1.2: 57 trials with their corresponding scenarios

2.1.3 Data acquisition and storage

The data collection was done with IMU sensors placed on the participant at six different locations (see fig. 2.1.3):

- centered on posterior level of L5/S1 joint i.e. the **trunk**
- centered on the **wrist** on the dorsal forearm
- centered on both the anterior **thighs**
- centered 5 [cm] above the bony processes of both ankles i.e. the **shins**

Figure 2.1.3: Location of the six IMUs (trunk, wrist, right/left thighs, right/left shins)

The data was acquired at a sampling frequency of *100 [Hz]* and the whole dataset is composed of 10'260 text files (30 participant * 57 trials * 6 sensors). All the data is organised in a folder named 'input_data_SD'. This main folder contains thirty sub-folders which regroups the data by participant numbers 1 to 30. Each participant's sub-folder contains all data acquired by the participant in text files named

systematically as '#-000_00B432**.txt', where '#' represents the trial number (see tab. 2.1.2) and '**' represents the sensors location (see tab. 2.1.5).

Figure 2.1.4: 9 scenarios (Flat even, Cobble stone, Up/Down-stairs, Slope up/down, Bank left/right, Grass)

Orange Sensor number/**	Sensor location
CC.txt	Trunk
95.txt	Wrist
93.txt	Right thigh
8B.txt	Left thigh
9B.txt	Right shank
B6.txt	Left shank

Figure 2.1.5: Table for sensor locations of each trial based on last 2 digits of filenames

Each text file organise's the sensor's output under a label similar to tab. 2.1.6. The axis of the sensors are defined as the X-axis, being the vertical direction, and the Y-axis, being the medio-lateral direction, and the Z-axis, being the anterior-posterior direction.

Labels	Unit	Description
PacketCounter	*N/A*	Packet counter, value will be same if data frames were recorded at the same time (increase 1 unit per data frame)
SampleTimeFine	*N/A*	Not recorded in this study
Acc_X	m/s^2	Acceleration in the vertical direction (w/gravity)
Acc_Y	m/s^2	Acceleration in the medio-lateral direction (w/gravity)
Acc_Z	m/s^2	Acceleration in the anterior-posterior direction (w/gravity)
FreeAcc_X	m/s^2	Acceleration in the vertical direction (w/o gravity)
FreeAcc_Y	m/s^2	Acceleration in the medio-lateral direction (w/o gravity)
FreeAcc_Z	m/s^2	Acceleration in the anterior-posterior direction (w/o gravity)
Gyr_X	*rad/s*	Rate of turn along the vertical direction
Gyr_Y	*rad/s*	Rate of turn along the medio-lateral direction
Gyr_Z	*rad/s*	Rate of turn along the anterior-posterior direction
Mag_X	*a.u.*	3D magnetic field in the vertical direction
Mag_Y	*a.u.*	3D magnetic field in the medio-lateral direction
Mag_Z	*a.u.*	3D magnetic field in the anterior-posterior direction
VelInc_X	*m/s*	Delta_velocity (dv) in the vertical direction
VelInc_Y	*m/s*	Delta_velocity (dv) in the medio-lateral direction
VelInc_Z	*m/s*	Delta_velocity (dv) in the anterior-posterior direction
OriInc_q0	*N/A*	Delta_quaternion (q0)
OriInc_q1	*N/A*	Delta_quaternion (q1)
OriInc_q2	*N/A*	Delta_quaternion (q2)
OriInc_q3	*N/A*	Delta_quaternion (q3)
Roll	*deg*	Euler angles in XYZ Earth fixed type (roll)
Pitch	*deg*	Euler angles in XYZ Earth fixed type (pitch)
Yaw	*deg*	Euler angles in XYZ Earth fixed type (yaw)

Figure 2.1.6: Data stored in .txt files (all variables are with dimension n x 1)

2.1.4 Remarks

In this section several additional information was added to complete the description of the dataset.

Missing Data
The data acquisition was not perfect and some data is missing and replaced by Nan. Nevertheless the data missing rate remained low with each sensor location and walking surface reaching a maximum of 0.93 % and 0.66 % respectively (see

tab. 2.1.7a and 2.1.7b).

Sensor location	Missing rate Mean (standard deviation)
Trunk	0
Wrist	0.13% (0.13%)
Right thigh	0.19% (0.18%)
Left thigh	0.93% (4.08%)
Right shank	0.08% (0.08%)
Left shank	0.06% (0.05%)

(a) missing rate by sensor locations

Sensor location	Missing rate Mean (standard deviation)
Calibration (CALIB)	0
Flat even (FE)	0.17% (0.26%)
Cobble stone (CS)	0.36% (1.67%)
Upstairs (StrU)	0.59% (3.04%)
Downstairs (StrD)	0.66% (3.03%)
Slope up (SlpU)	0.02% (0.05%)
Slope down (SlpD)	0.10% (0.20%)
Bank left (BnkL)	0.16% (0.23%)
Bank right (BnkR)	0.30% (0.42%)
Grass (GR)	0.12% (0.17%)

(b) missing rate by scenario

Figure 2.1.7: Missing rate

Comparison within each dimension

As seen previously, the dataset can be decomposed in terms of:

- ◇ *Participants*
- ◇ *Scenarios*
- ◇ *Trials*
- ◇ *IMU position*

For each term, the norm of the angular velocity and the linear acceleration were plotted and analysed to better understand the composition of the dataset.

- ◇ *Participants*

  The dataset is mainly composed of young participants with no reported neurological or musculoskeletal conditions. In the context of the project, it gives a good start for implementing the method but a dataset with older people is required to validate the results.

  In fig. 2.1.8, the resulting linear acceleration and angular velocities are pretty similar between participants. This is promising since a model needed to find common embedded information between each participant's gait.

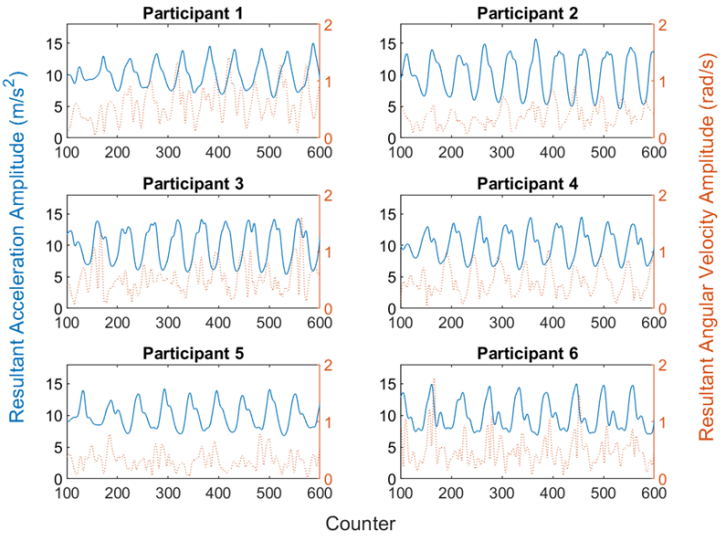


Figure 2.1.8: Comparison between participants walking on flat ground with an IMU worn on the trunk

◇ *Trials*
The more trials and the more data compiled, the more complete the dataset will be. In fact, a large number of samples are required in order to improve the generalisation property of the dataset. Indeed, the more different situations encountered during the training, the more the model will learn to detect the correct class with similar situations.

In fig. 2.1.9, the resulting linear acceleration and angular velocities are pretty similar between trials. This proves the repeatability of the experiment.

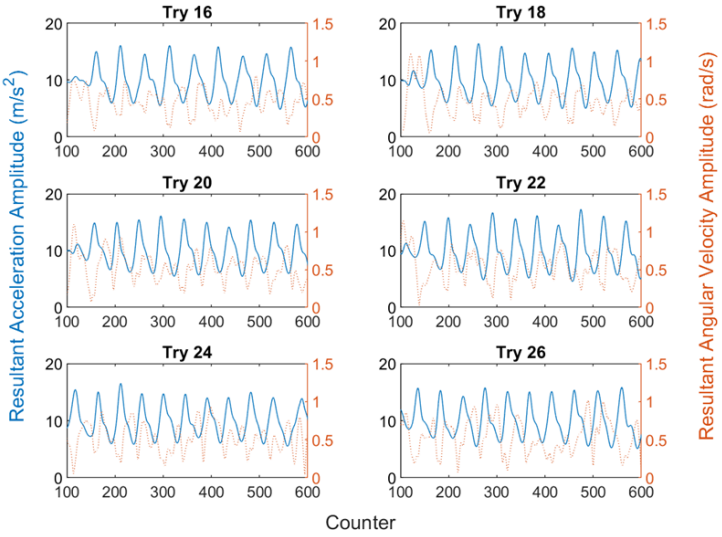


Figure 2.1.9: Comparison between trials for upstairs walking with an IMU worn on the trunk

◇ *Scenarios*
Multiple isolated scenarios allow us to choose the degree of complexity and variety of the dataset.

In fig. 2.1.10, the resulting linear acceleration and angular velocities look different in terms of range, amplitude or mean compared to the two previous comparisons. This visual difference between scenarios is promising since the models will use them and others even more to properly detect the subtle gait difference between scenarios.

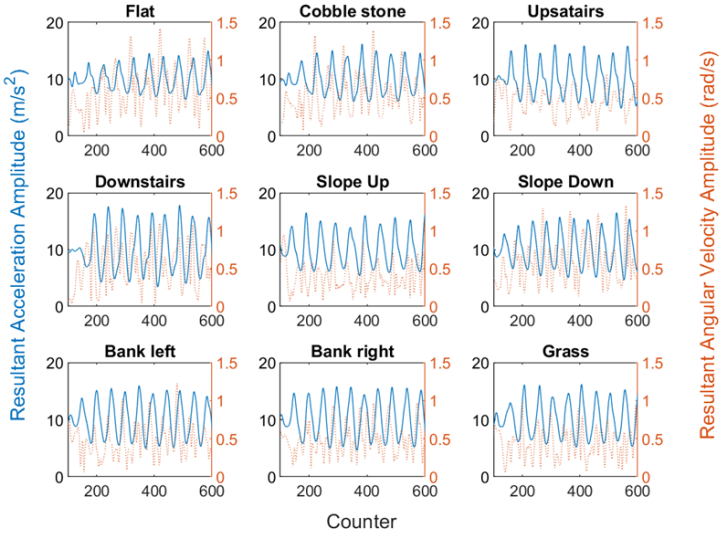


Figure 2.1.10: Comparison between scenarios for participant N°1 with an IMU worn on the trunk

◇ *IMU position*
Different positions give different information. In fig. 2.1.11, the resulting linear acceleration and angular velocities are very dependant to the IMU position. The cycles follow different patterns containing different information.

In fig. 2.1.11, the trunk looks more poor in information than the shanks and the thighs. However, it remains the main focus of this project and largely succeeds the interest of the research community. This is mostly due to its non-invasive location near the CoM.

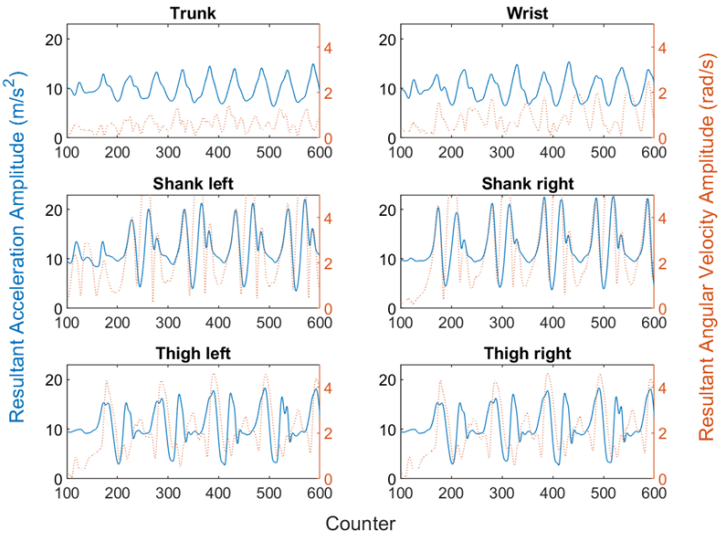


Figure 2.1.11: Comparison between IMU position for participant N°1 walking on flat ground

Outliers

During the testing of the dataset, several participants (3,4,8,16,18 and 21) presented significantly lower performance in terms of accuracy, precision and recall. After a deeper analysis of their data, only participant 18 was identified with an abnormal signal (See fig. 2.1.12) and was considered an outlier for the rest of the work.

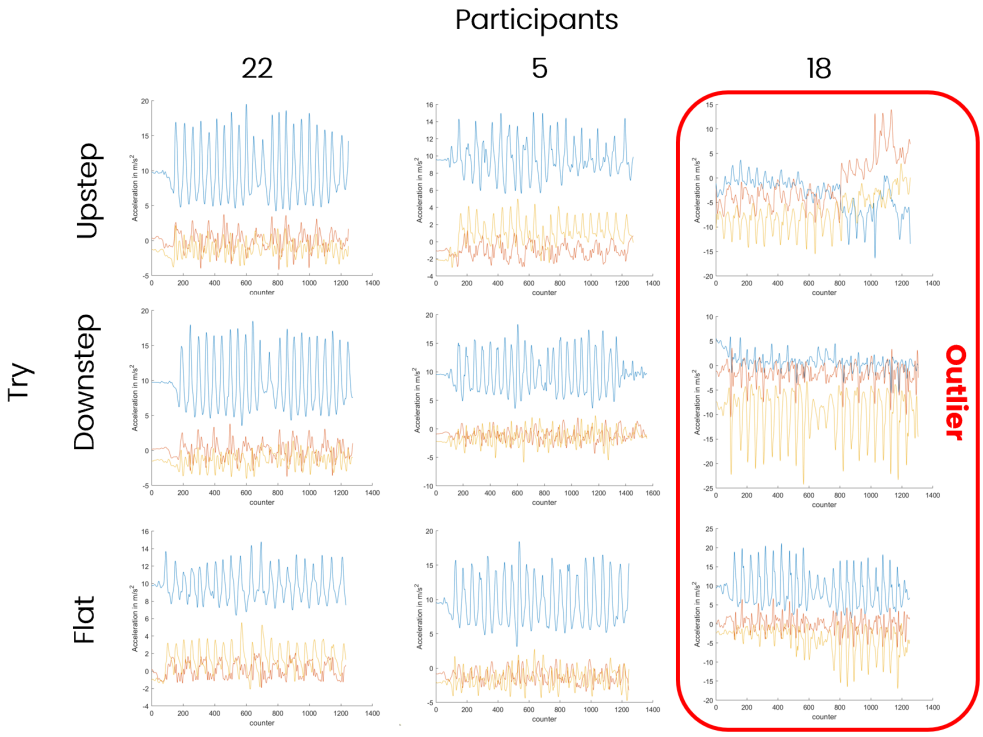


Figure 2.1.12: Plots of the signal for a random trial of participants 22, 5 and 18 for flat, downstep and upsteps.

2.2 Feature extraction

The feature extraction was applied for each model. The dataset of a model regrouped :

- an IMU position
- a set of selected scenarios
- all the corresponding trials
- all the participants

The set of scenarios represent the different classes that the model had to predict. All the corresponding signals for a given model passed through the three following steps.

- Pre-processing
- Sampling
- Features calculation

2.2.1 Pre-processing

The signals of the dataset were preprocessed with a 2nd order butterworth low pass filter with a cut-off frequency of 3 Hz (see fig. 2.2.1). This was a necessary step to filter out the noise of the signal due to the sensitivity and the offset errors of the sensors but also with the user's trembles.

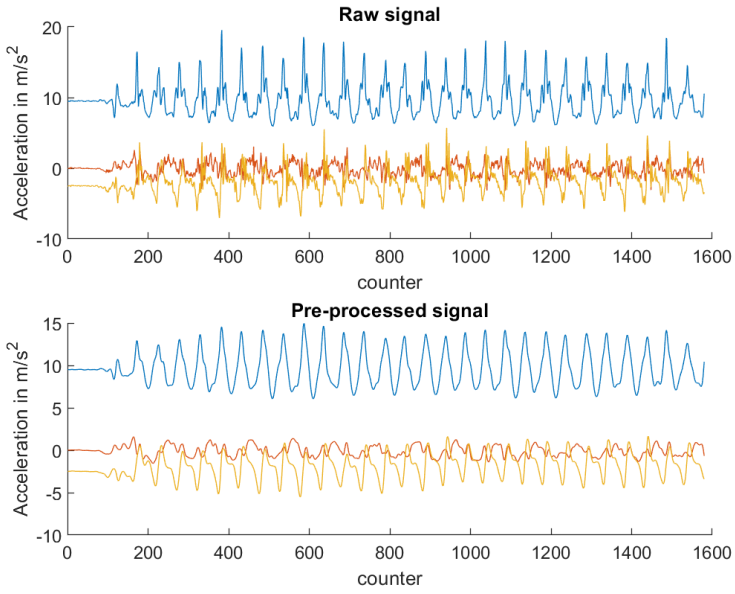


Figure 2.2.1: Raw and pre-processed signal of try 4 of participant N°1 on trunk

2.2.2 Sampling

Every signal were sampled into a chunks set . Every chunk was attributed to a class and a participant. The class attributed corresponded to the scenario of the sampled signal.

To get the chunks set of a signal, it was first cut at the beginning with an offset of 2s (see red lines in fig. 2.2.2). Then a time window was moved all along the signal. An overlap of fifty percent was chosen between two consecutive windows (see yellow lines in fig. 2.2.2). This continued until the window went beyond the signal size.

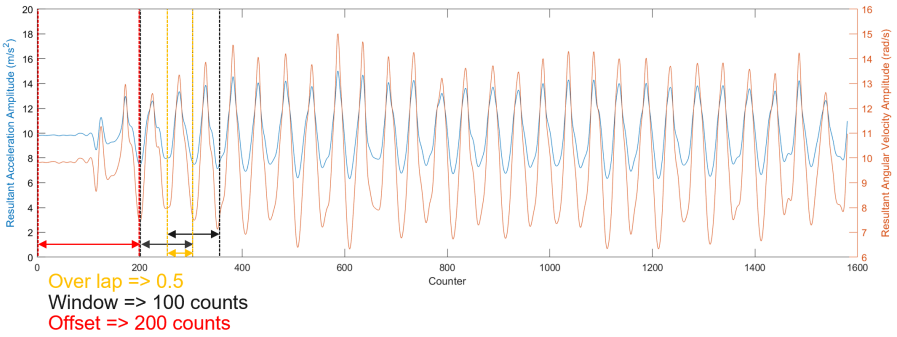


Figure 2.2.2: Sampling

At the end of the sampling, the chunks constituted a labeled dataset decomposed into scenario and participant number.

2.2.3 Features calculations

Each chunk contains meaningful characteristics and properties related to its class label. The features are used to uncover this information and help the classification capacity of the model.

Each chunk is a signal with N element as follows:

$$signal = [x_1,\ \dots\ ,\ x_{i-1},\ x_i,\ x_{i+1},\ \dots\ ,\ x_N] \qquad \text{(Eq. 2.1)}$$

The feature belonged to two categories :

- Time domain
- Frequency domain

Each of them underline different type of meaningful information. Every feature calculated was added to an array of size $N_f\ x\ N_{chunk}$ with N_f representing the number of features and N_{chunk} the total number of chunk extracted from every tries and

every participant. Their details of calculation are described bellow.

Time domain

◇ *Mean*
The mean value is the summation of values of all the data points of the signal in the window, divided by the window size N. It is defined as

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \text{(Eq. 2.2)}$$

◇ *Mean Absolute Value*
The means absolute value is the summation of absolute values of all the data points of the signal in the window, divided by the window size N. It is defined as

$$MAV = \frac{1}{N}\sum_{i=1}^{N} |x_i| \qquad \text{(Eq. 2.3)}$$

◇ *Harmonic Mean*
The harmonic mean is defined as

$$HM = \frac{N}{\sum_{i=1}^{N} \frac{1}{x_i}} \qquad \text{(Eq. 2.4)}$$

◇ *Variance*
The variance of data measures the spread of the data around the mean. It is defined as

$$Var = \sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2 \qquad \text{(Eq. 2.5)}$$

◇ *Standard deviation*
The standard deviation is the square root of the variance. It is defined as

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2} \qquad \text{(Eq. 2.6)}$$

◇ *Root mean squared*
The root mean square (RMS) is defined as

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \qquad \text{(Eq. 2.7)}$$

◇ *Median*
To obtain the median, the vector's signal was sorted in increasing or decreasing order and then the th term was found as follow:

$$m = \begin{cases} \left(\frac{N+1}{2}\right)^{th} term, \ if \ N \ is \ odd \\ \frac{\left(\frac{N}{2}\right)^{th} term + \left(\frac{N}{2}+1\right)^{th} term}{2}, \ if \ N \ is \ even \end{cases} \quad \text{(Eq. 2.8)}$$

The th element represents the median of the signal.

◇ *Median absolute deviation*
The median absolute deviation (MAD) is defined as

$$MAD = median(|signal - m|) \quad \text{(Eq. 2.9)}$$

◇ *Inter-Quartile Range*
The inter-quartil range (IQR) is defined as

$$IQR = Q1 - Q2 \quad \text{(Eq. 2.10)}$$

With Q1 and Q2 being respectively the first and the third quartile.

$$Q1 = \begin{cases} \left(3 \cdot \frac{N+1}{4}\right)^{th} term, \ if \ N \ is \ odd \\ \frac{\left(3 \cdot \frac{N+1}{4}\right)^{th} term + \left(\frac{N}{2}+1\right)^{th} term}{2}, \ if \ N \ is \ even \end{cases} \quad \text{(Eq. 2.11)}$$

◇ *Minimum*
The minimum (min) represents the lowest value of all the data points of the signal in the window. It is defined as

$$min = min(x_i), \ 0 \leq i \leq N-1 \quad \text{(Eq. 2.12)}$$

◇ *Maximum*
The maximum (max) represents the highest value of all the data points of the signal in the window. It is defined as

$$max = max(x_i), \ 0 \leq i \leq N-1 \quad \text{(Eq. 2.13)}$$

◇ *Range*
The range is the difference between the highest and the lowest value of all the data points of the signal in the window. It is defined as

$$range = max - min \quad \text{(Eq. 2.14)}$$

◇ *Skewness*
Skewness (SK) measures the asymmetry of the distribution of the data points around the mean and is defined as

$$SK = \frac{E(x-\mu)^3}{\sigma^3} \quad \text{(Eq. 2.15)}$$

with μ the mean and σ the standard deviation of the signal calculated before.

◇ *Kurtosis*
Kurtosis (KT) is a descriptor of the shape of the distribution of the data points and is defined as

$$KT = \frac{E(x-\mu)^4}{\sigma^4} \quad \text{(Eq. 2.16)}$$

with μ the mean and σ the standard deviation of the signal calculated before.

◇ *Zero Crossing Rate*
The zero crossing rate (ZCR) is the number of times the signal crosses zero and changes its signs. The ZCR is initialised as zero and is incremented when :

$$\{x_i > 0 \; and \; x_{i+1} < 0\} \; or \; \{x_i < 0 \; and \; x_{i+1} > 0\}, \; 1 \leq i \leq N \quad \text{(Eq. 2.17)}$$

◇ *Slope Sign Change*
The slope sign change (SSC) is the number of times the slope of the signal changes its sign. SSC is initialised as zero and incremented if :

$$\{x_i > x_{i-1} \; and \; x_i > x_{i+1}\} \; or \; \{x_i < x_{i-1} \; and \; x_i < x_{i+1}\}, \; 1 \leq i \leq N \quad \text{(Eq. 2.18)}$$

◇ *Cross-Correlation Coefficient*
The cross-correlation coefficient measures the linear dependence between two axis. The Pearson correlation coefficient is defined as

$$\rho(W,V) = \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{W_i-\mu_w}{\sigma_w}\right)\cdot\left(\frac{V_i-\mu_v}{\sigma_v}\right) \quad \text{(Eq. 2.19)}$$

where μ_w and σ_w are the mean and standard deviation of the W-axis, respectively, and μ_v and σ_v are the mean and standard deviation of the V-axis.

Frequency domain

For the frequency domain, some additional preprocessing steps were necessary to extract the features. First the mean was subtracted from the signal:

$$signal_{zm} = signal - \mu \quad \text{(Eq. 2.20)}$$

And then a 'hann' window (see fig. 2.2.3) was applied to the previous results:

$$signal_{zm-hann} = hann \cdot signal_{zm} \quad \text{(Eq. 2.21)}$$

To finally apply a fast fourrier transform (fft):

$$signal_{fft} = |fft\,(signal_{zm-hann})\,| = [x_1^{fft},\ \ldots,\ x_{i-1}^{fft},\ x_i^{fft},\ x_{i+1}^{fft},\ \ldots,\ x_{\frac{N}{2}}^{fft}] \quad \text{(Eq. 2.22)}$$

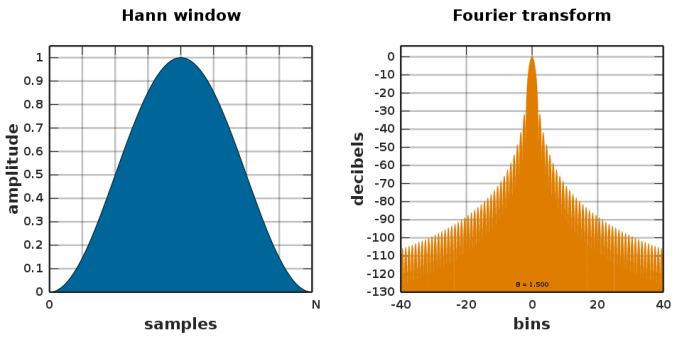


Figure 2.2.3: hann window in the time and frequency domain

The frequency bins of the signal was derived as:

$$f_{bins} = [0,\ \dots\ ,\ N/2] \cdot \frac{f_s}{N} = [f_1,\ \dots\ ,\ f_{i-1},\ f_i,\ f_{i+1},\ \dots\ ,\ f_{\frac{N}{2}+1}] \qquad \text{(Eq. 2.23)}$$

With f_s the sampling frequency (100 Hz) and N the size of the signal.

◇ *Spectral centroid*
The spectral centroid indicates where the centre of mass of the spectrum is located. It is defined as

$$spec_\mu = \frac{\sum_{i=1}^{\frac{N}{2}+1} f_i \cdot x_i^{fft}}{\sum_{j=1}^{\frac{N}{2}+1} x_j^{fft}} \qquad \text{(Eq. 2.24)}$$

◇ *Spectral spread*
The spectral spread measures the spread of the spectrum around the spectral centroid. It is defined as

$$spec_\sigma = \sqrt{\frac{\sum_{i=1}^{\frac{N}{2}+1} (f_i - spec_\mu)^2 \cdot x_i^{fft}}{\sum_{j=1}^{\frac{N}{2}+1} x_j^{fft}}} \qquad \text{(Eq. 2.25)}$$

2.3 Approach I - Feature Selection

This first approach targets the selection of the best features to maximise the classification performance for a given model. It first ranked every feature with a score and selected the right number to keep to optimise performance and computational cost. This approach is categorised as a " filter method" since it selected the feature independently of the model.

2.3.1 Ranking

Every feature contains a different amount of useful information. To maximise the performance of a model while maintaining reasonable computational power, it is important to identify a subset of features that are relevant for predicting a response. Therefore, the features were scored with three tests :

- Minimum Redundancy Maximum Relevance Test (MRMRT)
- Chi-Square Test (CST)
- T-Test (TT)

Minimum Redundancy Maximum Relevance Test (MRMRT)
This test is aimed at choosing the best subset of features while taking into account the redundancy and the correlation between them.

The MRMR score S_{mrmrt} of each feature represents the confidence of the algorithm that this features is relevant for the prediction (see fig. 2.3.1).

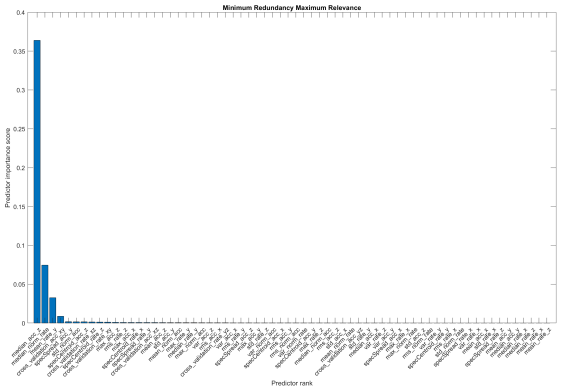


Figure 2.3.1: Example of a MRMR test plot

The final score is derived as follows:

$$F_{mrmr} = \begin{cases} 0, \ if \ S_{mrmrt} < 0.01 \\ a, \ if \ S_{mrmrt} > 0.01 \end{cases} \qquad \text{(Eq. 2.1)}$$

With a the number of feature with $S_{mrmrt} > 0.01$.

Chi-Square Test (CST)
This test is aimed at selecting features which are highly dependent on the response. In fact the Chi-Square test was used to test the independence between the predictor variable and the response variable.

The higher the Chi- Square test score S_{cst}, the more the hypothesis of independence is rejected and the more likely the feature should be selected (see fig. 2.3.2).

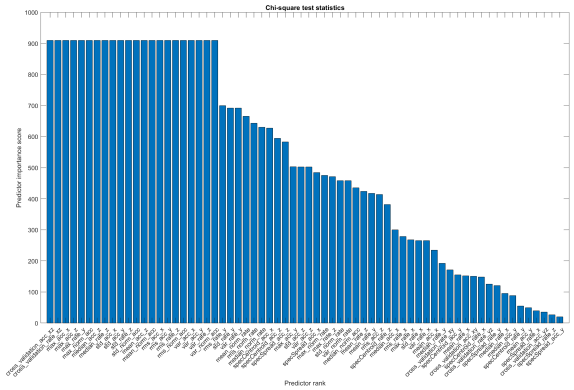


Figure 2.3.2: Example of a Chi-Square Test plot

The final score was calculated in three steps. Firstly the features were sorted in decreasing order in function of their score.

$$features_{cst} = [f_{best},\ \dots\ ,\ f_i,\ \dots\ ,\ f_{worst}] \qquad \text{(Eq. 2.2)}$$

Secondly a final score inversely proportional to their index number was attributed to each feature.

$$features_{cst} = [f_{best},\ \dots\ ,\ f_i,\ \dots\ ,\ f_{worst}] = [N_f,\ \dots\ , 1] \qquad \text{(Eq. 2.3)}$$

Hence the feature with the best score was sorted in the first position and received a score equal to the total number of features N_f, and the worst feature was sorted in the last position with a score equal to one.

Thirdly the feature vector is normalised by the total number of features and multiplied by the maximum score of MRMR (i.e. a).

$$F_{cst} = features_{cst} \cdot \frac{a}{N_f} \qquad \text{(Eq. 2.4)}$$

The final score of the Chi-Square Test F_{cst} was comprised between zero and a.

T-Test (TT)
This test determined if there was a significant difference between the means of the different classes. It was applied to each possible pair of classes as shown with this three classes example:

- Flat - Upstep (FU)
- Downstep - Upstep (DU)

- Flat - Downsten (FD)

For each pair, the p-value of the t-test was calculated and the cumulative distribution function (CDF) of the p-values were calculated (see fig. 2.3.3). A small p-value indicates that there was a significant difference between the mean of the two classes for a specific feature. Hence the features with a small p-value had strong discrimination power.

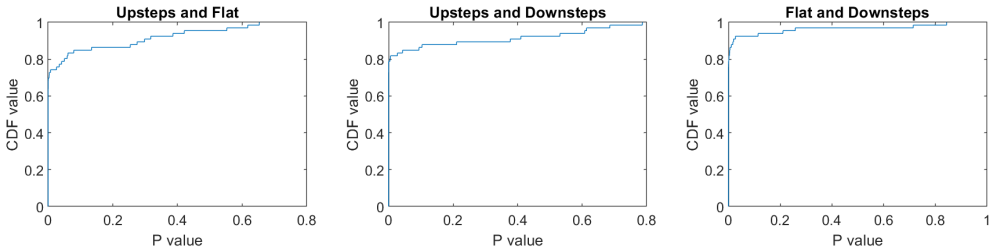


Figure 2.3.3: Example of the cumulative distribution function of the p-values between each pair of classes

The final score was calculated for each pair in three steps similarly as the Chi-Square Test. Firstly the features were sorted in ascending order in function of their p-value:

$$feature_{tt} = [p_{best},\ \dots\ ,\ p_i,\ \dots\ ,\ p_{worst}] \qquad \text{(Eq. 2.5)}$$

Secondly a final score inversely proportional to their index number was attributed to each feature.

$$features_{tt} = [p_{best},\ \dots\ ,\ p_i,\ \dots\ ,\ p_{worst}] = [N_f,\ \dots\ , 1] \qquad \text{(Eq. 2.6)}$$

Hence the feature with the smallest p-value was sorted in the first position and received a score equal to the total number of features N_f, and the worst feature with the biggest p-value was sorted in the last position with a score equal to one.

Thirdly the feature vector was normalised by the total number of features N_f and possible pair of classes N_p and multiplied by the maximum score of MRMR (aka a).

$$F_{tt}^i = features_{tt} \cdot \frac{a}{N_f \cdot N_p} \qquad \text{(Eq. 2.7)}$$

The final score of the T-Test F_{tt} was the sum of these three steps on each pair of classes and was comprised between zero and a:

$$F_{tt} = \sum_{i=1}^{N_p} F_{tt}^i \qquad \text{(Eq. 2.8)}$$

Final score The features final score were derived from the sum of the three previous tests as follow:

$$Score_{final} = F_{mrmrt} + F_{cst} + F_{tt} \quad \text{(Eq. 2.9)}$$

The features were ranked according to their score in decreasing order (see fig. 2.3.4).

$$rank = [feature_{best},\ \dots\ ,\ feature_i,\ \dots\ ,\ feature_{worst}] \quad \text{(Eq. 2.10)}$$

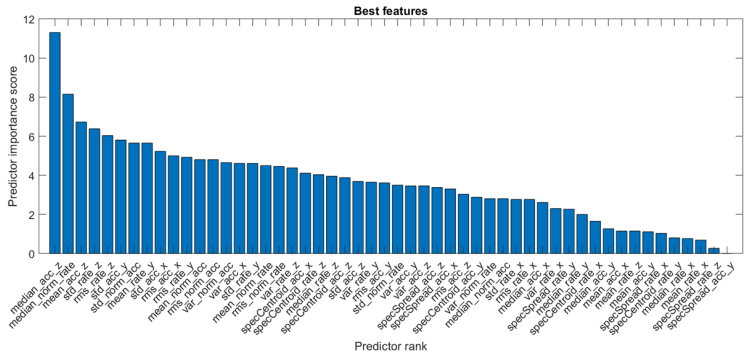


Figure 2.3.4: Feature ranked in decreasing order in function of their final score

2.3.2 Selection

The number of features selected influenced both the computational cost and the performance of the model. An excess of features can lead to overfit and a deficit to poor performance. Therefore the features selected had to represent an informative, discriminating and independent subset.

A sequential feature selection in a wrapper fashion was applied to find the optimal number of features. This approach only took into account the score of the feature and did not consider eventual redundancy or interaction between them.

The method consisted in selecting a subset of features by sequentially adding them one by one while trying to minimise the miss-classification error (MCE). This method was a cycle of 4 steps.

First, a feature was added to the set of features initially empty. Second, all the samples of the dataset corresponding to one of the features of the set were added and separated into a training (70% of the samples) and a testing set (30% of the samples). Each class was equally separated between the training and testing set. Third, the training set was used to train a standard Quadratic Discriminant Analysis (QDA). Fourth the testing set was used to calculate the corresponding MCE of the previously trained model.

$$MCE = \frac{\#\ of\ miss - classified\ sample}{total\ sample\ \#} \qquad \text{(Eq. 2.11)}$$

The cycle was repeated until the subset of features contained every possible feature. A solution which optimised both the computational cost and the overall performance was selected from the resulting graph (see fig. 2.3.5 for an example).

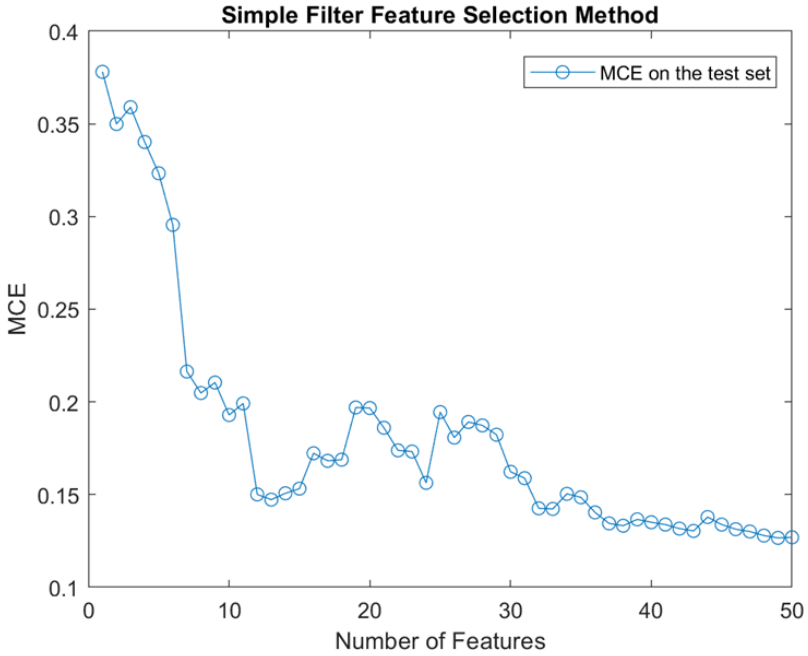


Figure 2.3.5: Resulting MCE curve of the sequential feature selection. In this example a good compromise in term of accuracy and computation is 13 features

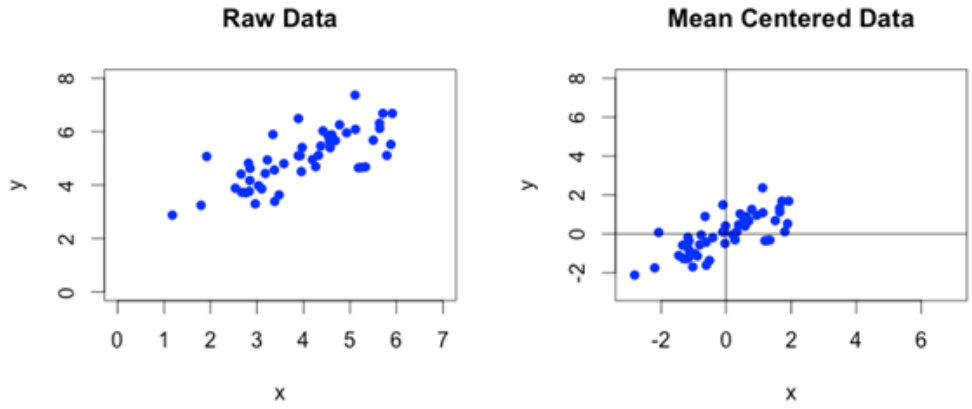
The final subset of selected features were of size $k \times N_{chunk}$.

2.4 Approach II - Feature Reduction

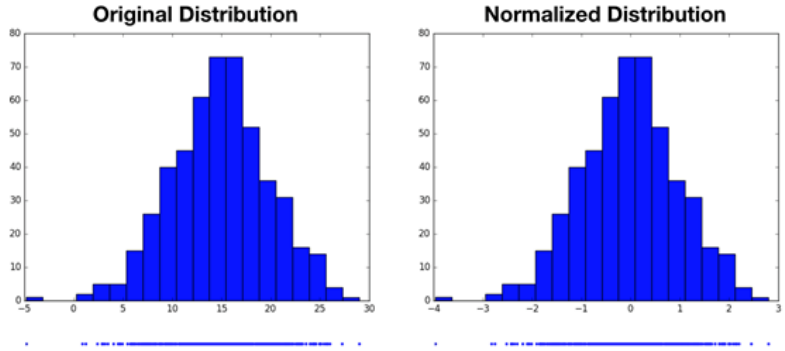
This second approach tried to summarise information contained in all the features in a reduced dimensionality. It first centered and normalised the features and then apply a principle component analysis (PCA). This approach is also categorised as a " filter method" since it selected the feature independently of the model.

2.4.1 Normalisation

The normalisation of the data was a prerequisite of the PCA. In fact, the features had to be centered and normalised (see fig 2.4.1).



(a) Centring



(b) Normalisation

Figure 2.4.1: Centring and normalisation

A standard scaler was applied to each feature set of the dataset :

$$Z_i = \frac{x_i - \mu_i}{\sigma_i} \qquad \text{(Eq. 2.1)}$$

with x_i the set, μ_i the mean, σ_i the standard deviation and Z_i the resulting set of feature i.

2.4.2 PCA

The principle composant analysis (PCA) is a well-known dimensionality reduction technique which projects the data onto a new orthogonal basis of smaller dimensions (see fig. 2.4.2).

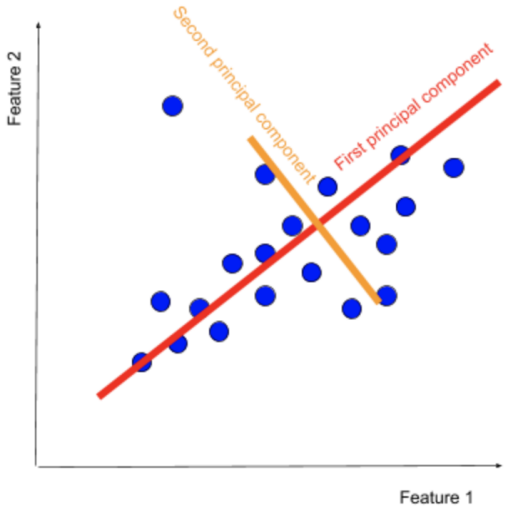


Figure 2.4.2: Example of PCA in 2D

The algorithm was applied in 5 steps:

- ***Step 1:*** Standardise the dataset
- ***Step 2:*** Compute the covariance matrix
- ***Step 3:*** Compute the eigenvalues and eigenvectors of the covariance matrix
- ***Step 4:*** Sort the eigenvalues and their corresponding eigenvectors
- ***Step 5:*** Pick k eignevalues and form a matrix eigenvectors
- ***Step 6:*** Transform the original matrix

Step 1: Standardise the dataset
The first step concerned the normalisation of the data and was already applied in the previous section with a standard scaler. It was essential to obtain meaningful results as it balanced every feature to the same order of magnitude.

Step 2: Compute the covariance matrix
The terms of the covariance matrix were calculated with:

$$Cov(z_i, z_j) = \frac{\sum_{k=1}^{N}(z_i k - \mu_i) \cdot (z_j k - \mu_j)}{N - 1} \qquad \text{(Eq. 2.2)}$$

with μ_i and μ_j equal to zero due to the fact the features data were normalised.

Step 3: Compute the eigenvalues and eigenvectors of the covariance matrix

To retrieve the eigenvectors V and the eigenvalues Λ of the covariance matrix, the following equation was resolved :

$$(Cov - \lambda I)v = 0 \qquad \text{(Eq. 2.3)}$$

with λ the associate eigenvalue of the eigenvetor v of the covariance matrix.

Step 4: Sort the eigenvalues and their corresponding eigenvectors
In the context of PCA, an eigenvector represents a direction or axis and the corresponding eigenvalue represents variance along that eigenvector. The higher the eigenvalue, the higher the variance along that eigenvector. As a result the eigenvalues and their corresponding eigenvector were sorted in decreasing order to have stronger variances first.

Step 5: Pick k eigenvalues and form a matrix eigenvectors
The sum of every eigenvalue was computed :

$$\lambda_{tot} = \sum \lambda \qquad \text{(Eq. 2.4)}$$

and divided from each eigenvalue to obtain the explained variance ratio which represents the percentage of variance attributed to each eigenvector. The first k eigenvectors were selected so the sum of their explained variance ratio was greater than an explained variance chosen beforehand.

Step 6: Transform the original matrix
The k eigenvectors were multiplied to the original feature matrix. The resulting matrix of size $k\ x\ N_{chunk}$ represented the reduced dataset of features used to optimise the models of approach II.

2.5 Classification

The dataset extracted from Approach I and II were used to train learning algorithm which, once trained, became models. As it is suggested by the name, a model tries to learn the link between an input and its response. In our case the input is a chunk of a signal processed by one of the two approach and the response is the actual class of this signal. The perfect model would be capable to predict the class of any input given to it correctly.

To access the capability and the performance of a model, it needs to be tested by data not encountered during the training. This help to minimise the effect of overfitting and proves the generalisation capability of a model. This was done with two different methods of cross-validation.

2.5.1 Learning algorithm

The resulting dataset of the two previous approaches were tested with all the following supervised algorithms from the MATLAB Classification Learner app:

- Decision trees
- Discriminant analysis
- Support vector machines
- Logistic regression
- Nearest neighbours
- Naive Bayes
- Kernel approximation
- Ensembles
- Neural networks

The algorithms were chosen in function of their performance in the Classification Learner app which uses a k-fold cross-validation. Each model was then exported into a function.

2.5.2 Cross-validation

Each model was tested with two different cross-validation methods:

- K-fold
- Leave-one-out

This was an important step to test the predictive performance of every model. Indeed, the evaluation of performance had to be done with unknown new data to properly evaluate if a model was under-fitted or over-fitted or well generalised.

At the end of each cross validation method, the confusion matrix was computed for each participant. The total confusion matrix was the addition of all the confusion matrix per participant (see fig. 2.5.1).

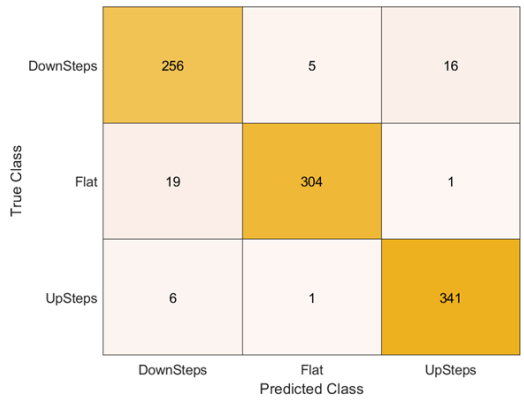


Figure 2.5.1: Example of confusion matrix for all participant

From the total confusion matrix, three metrics were computed to fully evaluate the effectiveness of a model.

First the precision which indicates the ability of the classifier not to label positive a sample that is negative. It is define as

$$Precision = \frac{TP}{TP + FP} \tag{Eq. 2.1}$$

with TP and FP being respectively the true positive and the false positive.

Second the recall which indicates the ability of the classifier to find all the positive samples. It is define as

$$Recall = \frac{TP}{TP + FN} \tag{Eq. 2.2}$$

with TP and FN being respectively the true positive and the false negative.

Third the F1-measure which is a combination between the two. It helps to find a good compromise between this two important metrics. It is define as

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{Eq. 2.3}$$

This metrics were also computed for the the confusion matrix per participant to evaluate the actual performance of the model per participant. The variance and the mean per metric was also computed. (See fig. 2.5.2)

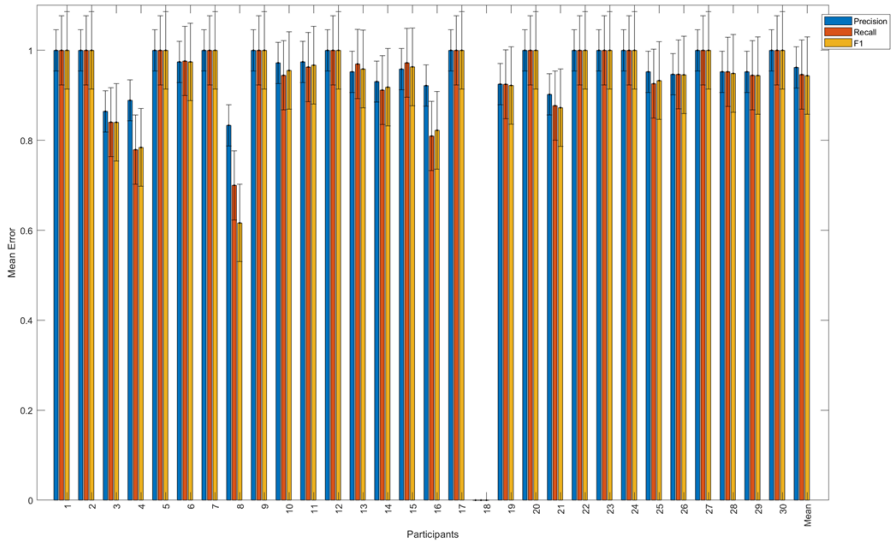


Figure 2.5.2: Example of metric per participant

K-fold
The K-fold method is a simple and very popular cross-validation technique largely used in the machine learning field. The main reason is that it ensures that all observations from the original dataset can appear in the training set, but also in the set used for testing. This is particularly effective when input data is limited.

The method consists of first performing a random separation of the data set into K folds. K refers to the number of groups into which the sample will be divided. K should not be too high or too low and is usually chosen between 5 and 10 according to the size of the dataset. The higher the K value, the less likely the model will be biased. Not to mention that a too large variance can lead to over-fitting. With a lower value, the approach simply joins the Train-Test Split method.

Then training of the learning algorithm is performed with K-1 folds (K minus 1), then the model is validated using the remaining K-fold. The same process is repeated until all K-folds are used in the testing set. The average of the recorded scores is then calculated, which represents the performance metric of the model.

Leave-one-out
The leave-one-out method is an extension of the previously described k-fold method. The difference lies in the way the dataset is separated. Instead of performing a random separation, the dataset is separated by participants. In the context of the dataset used for this report, there were thirty participants which make K=30 each K representing all the sample of a specific participant. The rest of the method is the same as the k-fold.

The training of the learning algorithm is performed with $N_{participants}$-1 (every participant minus 1), then the model is validated using the remaining participants. The same process is repeated until all participants are used in the testing set. The average of the recorded scores is then calculated which represents the performance metrics of the model.

3 Results and discussion

For this section, the optimal model for each approach is detailed. The two models are named :

- ◇ *trainClassifier_approach1_w640* **for Approach I (i.e. feature selection)**
- ◇ *trainClassifier_approach2_w640* **for Approach II (i.e. feature reduction)**

Then a new model is developed from approach I for comparison purposes with approach II. The new model's name is *trainClassifier_approach1_w640_comparison*.In fact, this new model selects a number of feature equivalents to the dimension of the reduced dataset of features from model *trainClassifier_approach2_w640*. Therefore, both models can be compared on equal footing.

Finally, four additional models were created with approach II to explore the influence of the time window, additional classes and different IMU positions:

- ◇ *trainClassifier_approach2_w100*
  for a different time window (1 [s])
- ◇ *trainClassifier_approach2_w640_MoreClasses*
  for additional classes (Up and down slopes)
- ◇ *trainClassifier_approach2_w640_wrist* **and** *trainClassifier_approach2_w640_shankR*
  for different IMU position (Wrist and right Shank)

3.1 Models

- ◇ *trainClassifier_approach1_w640*
  This first model used the approach I methodology (i.e. feature selection). The time window was fixed to 6.4 [s] and the model was trained with the following dataset :
  - All participants beside participant 18
  - IMUs positioned on the trunk.
  - Three distinct classes; Flat, Up stairs and down stairs.
  - All trials from the selected classes and IMU position.

  The results of the MRMRT clearly highlight five features (see fig. 3.1.1):
  - zero crossing rate (angular velocity z)
  - maximum (acceleration x)
  - mean (angular velocity y)
  - minimum (norm angular velocity)

– cross correlation (acceleration y and z)

All the rest of the features have a really bad score. The CST obtains a less discriminant result with a score slowly going down (see fig. 3.1.1).

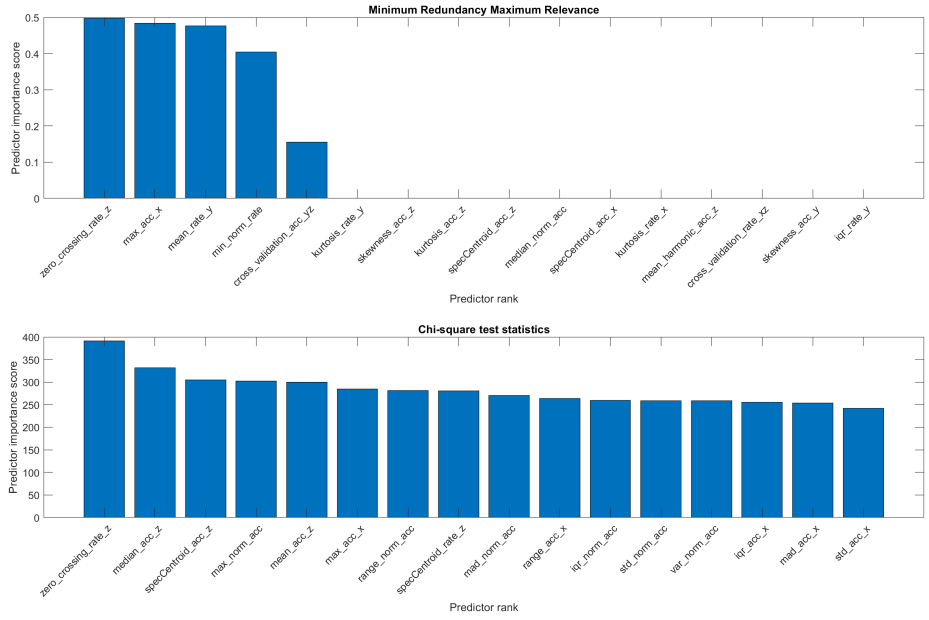


Figure 3.1.1: Plot of the MRMR and CST score for the first 16 features

The cumulative distribution of the TT for each pair of class's looks very similar. All of them indicate that at least 60% of the features have a p-value close to zero meaning that they have strong discrimination power (see fig. 3.1.2).

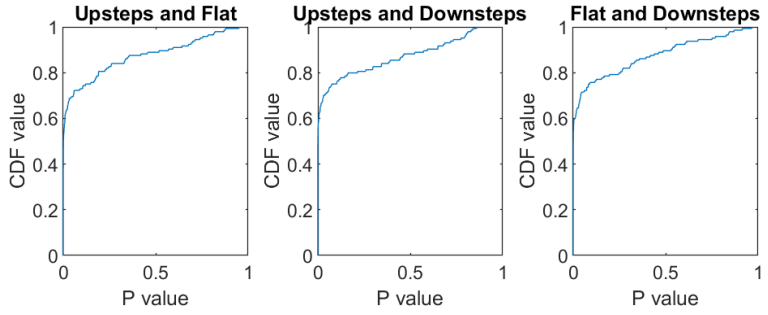


Figure 3.1.2: Plot of the p-value obtain by the TT for the three combination of classes

The final ranking score clearly indicates two features i.e. the zero crossing rate (angular velocity z) and the maximum (acceleration x). Then the score slowly goes down from the third feature (see fig. 3.1.3).

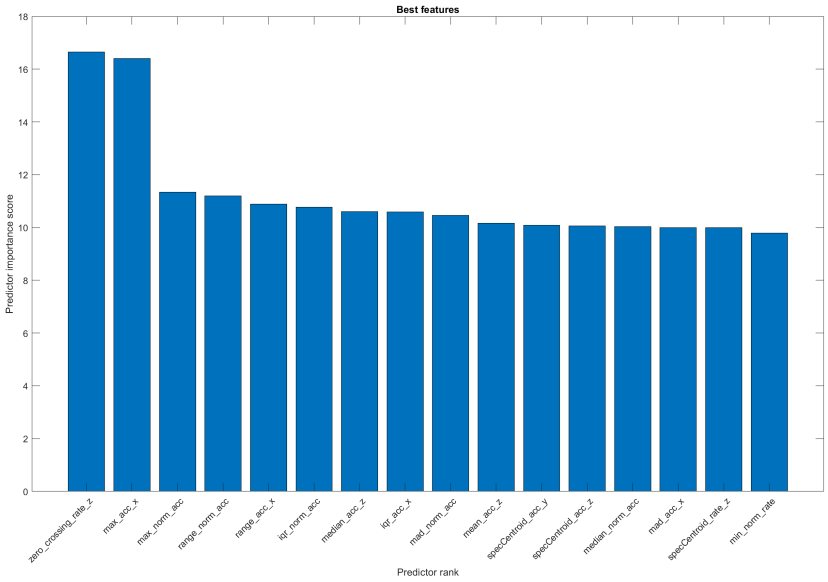


Figure 3.1.3: Plot of the final ranking score for the first 16 features

The feature ranking shows the best score for the linear acceleration in the X-axis, the angular velocity around the Z-axis and the norm of the acceleration. That indicates, as expected, that the scenario such as stair walking can be strongly distinguish by its movement in the vertical direction and its rotation around the anteroposterior axis (respectively the linear acceleration on the X-axis and the angular velocity around the z-axis). The norm seems to be able to take into its advantage these two movements; a strong asset for the feature selection.

The resulting MCE curve of the sequential feature selection indicates a plateau starting from the thirteen features which indicates an MCE of 0.06 (6 %). Therefore thirteen was selected as a good compromise between performance and computational cost.

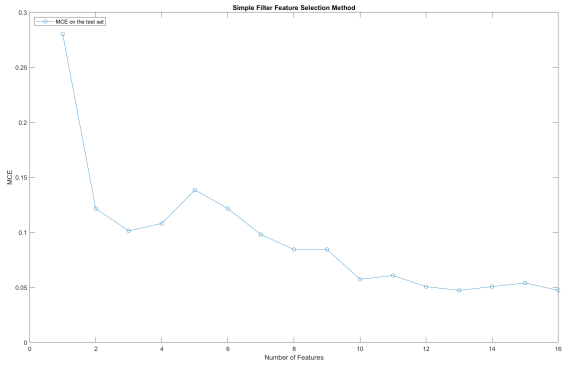


Figure 3.1.4

The thirteen features were sent to the MATLAB Classification App and the best model obtained was a Gaussian SVM which achieved a k-fold cross-validation accuracy of 97.7%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 90.37%. This decrease indicates that a minority of the participants obtained a lower performance as demonstrated later.

The confusion matrix of the leave-one-out method shows that the most false positives are obtained for the 'Flat' class (see fig. 3.1.5b and 3.1.5a). This can be explained by the fact that flat walking is an in-between of the two other classes.

Class	Precision	Recall	F1
Flat	85%	97%	91%
DownSteps	89%	91%	90%
UpSteps	99%	85%	91%

(a) Metrics

(b) Confusion matrix

Figure 3.1.5: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model struggles to accurately classify the gait of participants 8, 15, 17 and 22 (see fig. 3.1.6). However the model shows good results for all the rest of the participants showing its generalisation property.

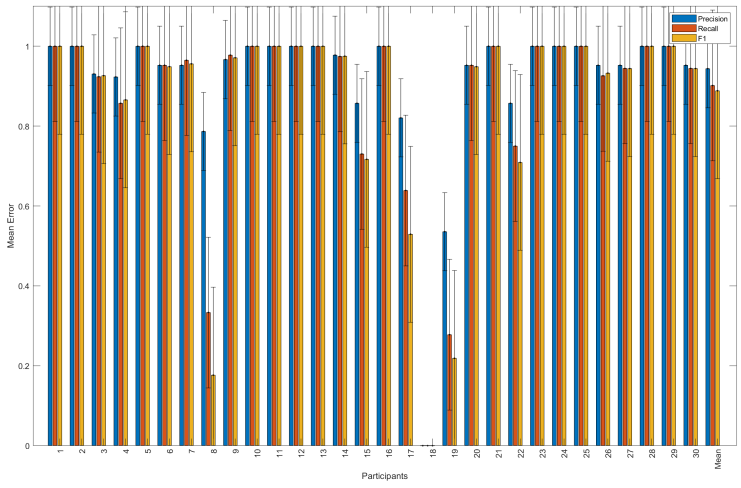


Figure 3.1.6: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach2_w640*

This second model used the approach II methodology (i.e. feature reduction). The time window was fixed to 6.4 [s] and the model was trained with the following dataset :

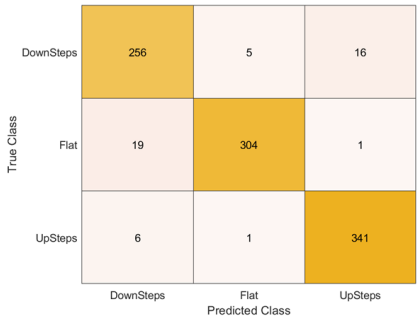
- All participants beside participant 18
- IMUs positioned on the trunk.
- Three distinct classes; Flat, Up stairs and down stairs.
- All trials from the selected classes and IMU position.

The feature reduction gave a resulting feature space dimension of thirty-two. This was then sent to the MATLAB Classification App and the best model obtained was a Polynomial SVM which achieved a k-fold cross-validation accuracy of 99.1%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 95.48%.

The confusion matrix of the leave-one-out method shows an overall positive results with good performances (see fig. 3.1.7b and 3.1.7a). However the metrics tend to be systematically lower for the class 'down steps'. This can be due to more variability in the gait when walking down steps. Indeed, when walking down stairs, the impact of your feet on each step can jam the measurement and make it difficult for a model to follow a pattern.

Class	Precision	Recall	F1
Flat	94%	98%	96%
DownSteps	92%	91%	92%
UpSteps	98%	95%	97%

(a) Metrics



(b) Confusion matrix

Figure 3.1.7: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model struggles to accurately classify the gait of participants 2, 4, 8 and 21 (see fig. 3.1.8). However the model shows good results for all the rest of the participants showing its generalisation property.

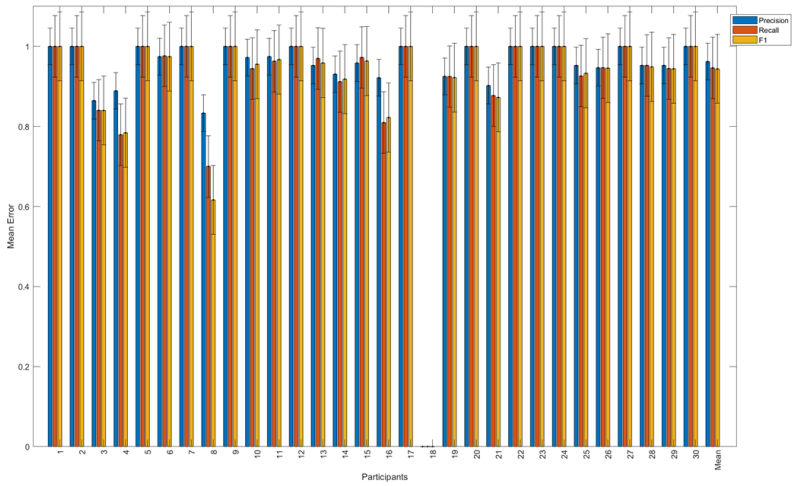


Figure 3.1.8: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach1_w640_comparison*

This third model used the approach I methodology (i.e. feature selection) and was developed to compare the two approaches on equal footing. The time window was fixed to 6.4 [s] and the model was trained with the following dataset :

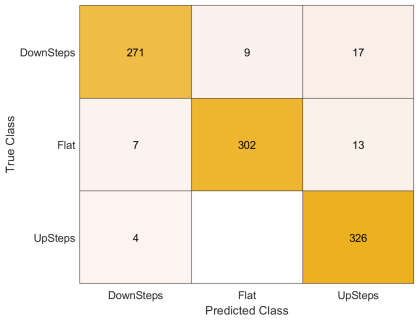
- all participants beside participant 18
- IMUs positioned on the trunk.
- Three distinct classes; Flat, Up stairs and down stairs.
- All trials from the selected classes and IMU position.

The methodology of the first approach was applied but the number of features selected was equal to thirty-two to be equal to the dimension of the feature space of the previous model. This feature was then send to the MATLAB Classification App and the best model obtained was a Cubic SVM which achieved a k-fold cross-validation accuracy of 98.6%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 94.71%.

The confusion matrix of the leave-one-out method shows an overall positive results with good performances (see fig. 3.1.9b and 3.1.9a). However there is a lot of false positives in class 'down step' from the class 'up step'. Indeed, these two classes present symmetry in their gait and therefore are hard to differentiate.

Class	Precision	Recall	F1
Flat	94%	97%	95%
DownSteps	91%	96%	94%
UpSteps	99%	92%	95%

(a) Metrics



(b) Confusion matrix

Figure 3.1.9: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicate that the model struggles to accurately classify the gait of participants 2, 4, 8, 19, 21 and 28 (see fig. 3.1.10). This result is really similar to the previous models. However the model shows some significant progress for participants 8 and 19 compared to the first model presented. Overall the metrics shows the generalisation ability of the model.

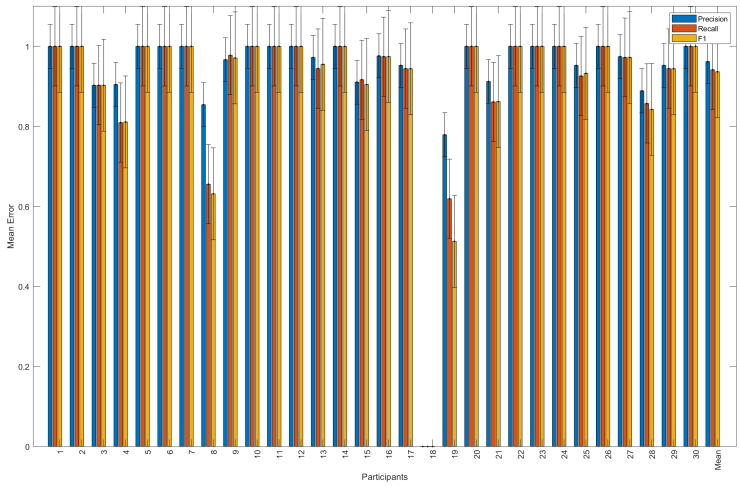


Figure 3.1.10: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach2_w100*

This fourth model was developed to compare the influence of the time window and uses the approach II methodology (i.e. feature reduction). The time window was fixed to 1 [s] and the model was trained with the following dataset :

- All participants beside participant 18
- IMUs positioned on the trunk.
- Three distinct classes; Flat, Up stairs and down stairs.

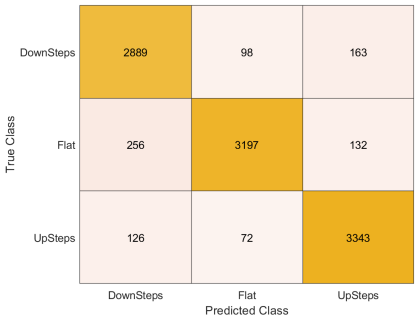
– All trials from the selected classes and IMU position.

The feature reduction gave a resulting feature space of dimension fourty-one and it was sent to the MATLAB Classification App. The best model obtained was a Quadratic SVM which achieved a k-fold cross-validation accuracy of 96.7%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 91.52%.

The confusion matrix of the leave-one-out method shows an overall positive result with good performance (see fig. 3.1.11b and 3.1.11a). However the metrics tends to be systematically lower than the previous model with a precision down to 89% for the class 'flat' and a recall down to 88% for the class 'down step'.

Class	Precision	Recall	F1
Flat	89%	95%	92%
DownSteps	92%	88%	90%
UpSteps	94%	92%	93%

(a) Metrics



(b) Confusion matrix

Figure 3.1.11: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model struggles to accurately classify the gait of participants 8, 15, 16 and 28 (see fig. 3.1.12). However the model shows good results for all the rest of the participants showing its generalisation property.

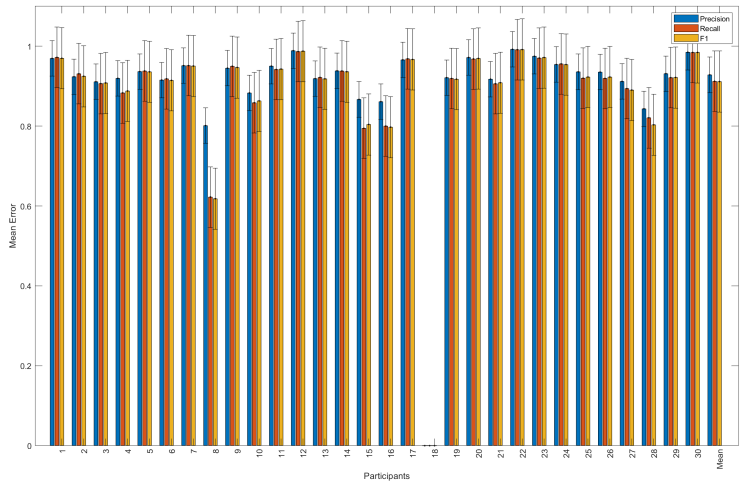


Figure 3.1.12: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach2_w640_MoreClasses* k-fold: 98.6 LOO : 88.635453 33 features quadratic SVM

This fifth model was developed to compare the influence of additional classes and uses the approach II methodology (i.e. feature reduction). The time window was fixed to 6.4 [s] and the model was trained with the following dataset :

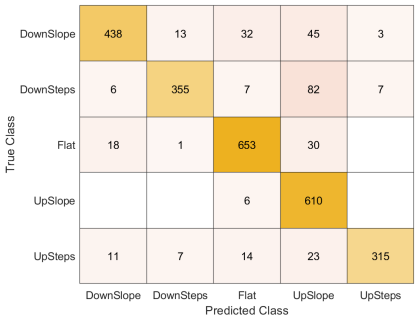
- All participants beside participant 18
- IMUs positioned on the trunk.
- Five distinct classes; Flat, Up stairs and down stairs, Up and down slope.
- All trials from the selected classes and IMU position.

The feature reduction gave a resulting feature space of dimension thirty-three and was sent to the MATLAB Classification App. The best model obtained was a Quadratic SVM which achieved a k-fold cross-validation accuracy of 98.6%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 88.64%.

The confusion matrix of the leave-one-out method shows an overall positive result despite the two additional classes (see fig. 3.1.13b and 3.1.13a). Indeed the classes 'slope up' and 'slope down' are very similar to the other class which significantly increase the difficulty of the classification.

Class	Precision	Recall	F1
UpSteps	85%	97%	91%
DownSteps	78%	94%	85%
DownSlope	83%	93%	87%
Flat	93%	91%	92%
UpSlope	99%	77%	87%

(a) Metrics



(b) Confusion matrix

Figure 3.1.13: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model struggles to accurately classify the gait of participants 1, 3, 4, 6, 8, 14, 15, 16, 17 and 28 (see fig. 3.1.14). The variance of each metric is high showing the difficulty of generalisation of the model.

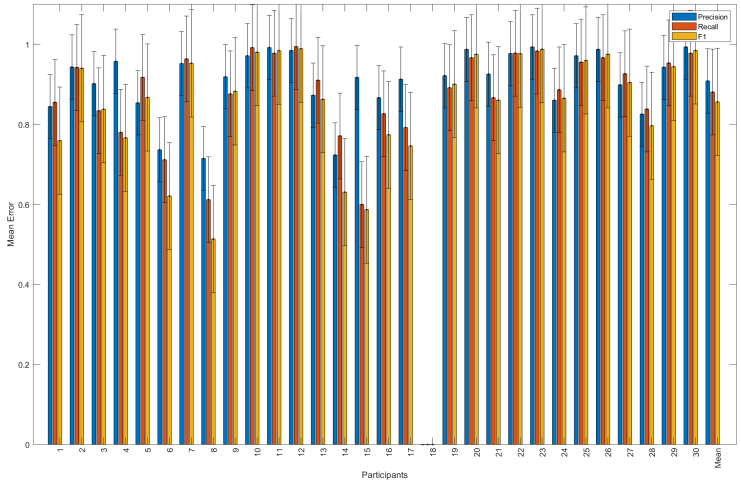


Figure 3.1.14: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach2_w640_wrist*

This sixth model was developed to compare the influence of the IMU position and uses the approach II methodology (i.e. feature reduction). The time window was fixed to 6.4 [s] and the model was trained with the following dataset :

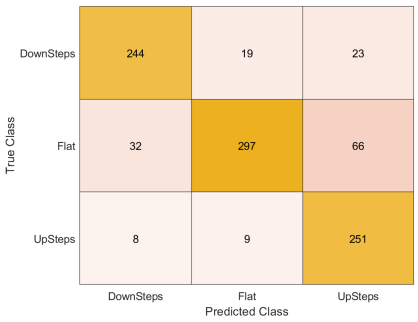
- all participants beside participant 18
- IMUs positioned on the wrist.
- Three distinct classes; Flat, Up stairs and down stairs.
- All trials from the selected classes and IMU position.

The feature reduction gave a resulting feature space of dimension thirty and was sent to the MATLAB Classification App. The best model obtained was a Quadratic SVM which achieved a k-fold cross-validation accuracy of 96.8%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 83.51%.

The confusion matrix of the leave-one-out method shows an overall positive results (see fig. 3.1.15b and 3.1.15a). However the metrics tends to be systematically lower than the other model. Indeed, the wrist is not an optimal position because it does not does not necessary follow a fixed pattern every time a participant walk on stairs or casually walk on flat ground.

Class	Precision	Recall	F1
Flat	76%	91%	83%
DownSteps	85%	86%	85%
UpSteps	93%	74%	83%

(a) Metrics



(b) Confusion matrix

Figure 3.1.15: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model struggle to accurately classify the gait of participants 1, 2, 3, 8, 9, 14, 19, 21, 22 and 28 (see fig. 3.1.16). The variance of each metric is high showing the difficulty of generalisation of the model.

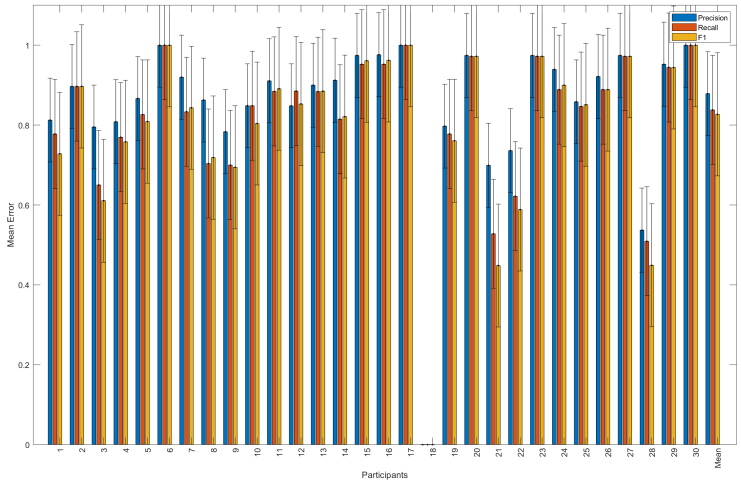


Figure 3.1.16: Precision, recall and F1 measure for every participant and the mean

◇ *trainClassifier_approach2_w640_shankR* This seventh and last model was also developed to compare the influence of the IMU position and uses the approach II methodology (i.e. feature reduction). The time window was fixed to 6.4 [s] and the model was trained with the following dataset :

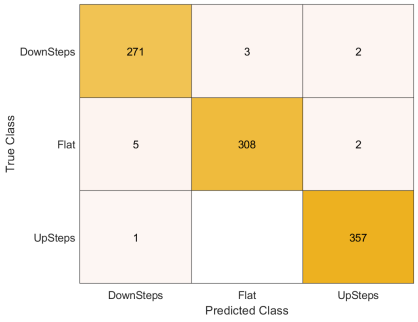
- all participants beside participant 18
- IMUs positioned on the right shank.
- Three distinct classes; Flat, Up stairs and down stairs.
- All trials from the selected classes and IMU position.

The feature reduction gave a resulting feature space of dimension twenty-nine was sent to the MATLAB Classification App. The best model obtained was a Quadratic SVM which achieved a k-fold cross-validation accuracy of 99.2%. The model was then tested with the leave-one-out cross validation and achieved an accuracy of 98.68%.

The confusion matrix of the leave-one-out method shows an overall excellent result with very few misclassifications (see fig. 3.1.17b and 3.1.17a). Indeed the metrics confirm the high performance of this model. This demonstrates the high influence of the IMU position on the results.

Class	Precision	Recall	F1
Flat	98%	99%	98%
DownSteps	98%	98%	98%
UpSteps	100%	99%	99%

(a) Metrics



(b) Confusion matrix

Figure 3.1.17: Metrics and confusion matrix from the Leave-one-out cross-validation

Subsequently, the metrics per participant indicates that the model successfully generalised to all the participant.

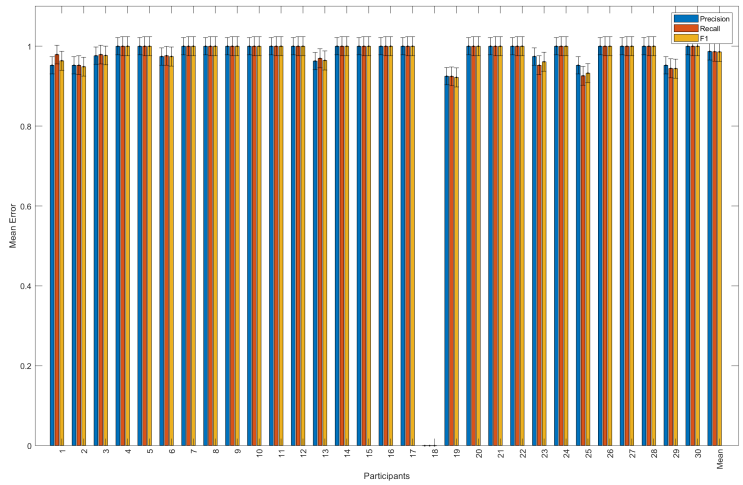


Figure 3.1.18: Precision, recall and F1 measure for every participant and the mean

3.1.1 Comparison of the approaches

As said previously, the two models *trainClassifier_approach2_w640* and *trainClassifier_approach1_w640_comparison* benefited from the same number of features and therefore can be compared on an equal footing.

The difference in term of k-fold and leave-one-out cross-validation are not significant between the two approaches with respectively 99.1% and 95.48% for the first approach, 98.6% and 94.71% for the second. Moreover,the two approaches struggle with almost the same participants. Overall both approaches perform equally.

However, they differ in their advantages and disadvantages. In fact, the first approach: feature selection expresses explicitly the best features. The second approach: feature reduction obtain an set of undefined mix of the input feature. The first approach is then more suitable to analyse the features selected in order to improve the future features choice. On the other hand, approach II chooses the dimension of the feature set more simply than approach I. This is important to simplify the implementation of the approach.

3.1.2 Exploration of variables

The implementation of the different algorithm used in this project requires making choices along the way. In this section, the influence of three variable are explored and analysed:

- *Time window*
- *More classes*
- *Different IMU position*

Time window
The two models *trainClassifier_approach2_w640* and *trainClassifier_approach2_w100*

are completely similar and only differ in term of the time window they used (respectively 6.4 [s] and 1 [s]).

There is a small difference in term of k-fold and leave-one-out cross-validation between the two approaches with respectively 98.6% and 94.71% for the first model, 96.7% and 91.52% for the second. Even if this difference of performance is low, the time window increases the accuracy while reducing the computational cost. Indeed, a smaller time window requires calculation at a higher frequency. Moreover, a small time window can possibly not incorporate enough of a signal to properly determined a feature in case of a slow gait with a large cycle time.

More classes
The two models *trainClassifier_approach2_w640* and *trainClassifier_approach2_w640_MoreClasses* are completely similar and only differ in terms of the number of classes they used (respectively 3 and 5 classes).

The k-fold cross-validation is equal for both model reaching 98.6% of accuracy. However, the addition of the two new classes results in a more significant loss of performance with the second leave-one-out cross-validation methods reaching 94.71 of accuracy for the 3-classes model and 88.64 of accuracy for the 5-classes.

This can be explained by the fact that the two additional classes 'up slope' and 'down slope' are in between walking flat and walking up or down stairs. This significantly increased the complexity of the problem. Moreover, the definition of a slope is not an hard boundary, in the context of this report it means flat with an inclination of 8.33% (i.e. 4-5 degree). Overall, the results are encouraging and demonstrate that the method can by applied to model with even more classes.

Different IMU position
The three models *trainClassifier_approach2_w640*, *trainClassifier_approach2_w640_wrist* and *trainClassifier_approach2_w640_shankR* are completely similar and only differ in term of the IMU position they used (respectively the trunk, wrist and right shank).

The k-fold cross-validation accuracy for the three model was respectively 99.1% for the trunk, 96.8% for the wrist and 99.2% for the right shank. Moreover the leave-one-out cross-validation accuracy for the three model was respectively 95.48% for the trunk, 83.51% for the wrist and 98.68% for the right shank.

This clearly demonstrates the importance of the IMU position. The worst performance is reached with the wrist. This is explained by the fact that when a person moves in one of the scenario, their wrist does not necessarily follow a constant pattern.

On the other hand, the right shank reached the best score of all models. In fact, the shank moves more than the trunk during locomotion and therefore embed bigger patterns easier to identify with features.

However, the trunk still achieved fine results and remains a good position candidate.

3.2 Conclusion

All the results show that it is nowadays possible to classify stairs, slope and flat walking with the help of a single IMU.

This can change the life of millions of people with mobility problems if this method was improved and used to help people with mobility problems. For example the model could be used to map the energy expenditure of every location.As a result, applications such as Google maps would have the possibility to take into account the mobility of its user and let people with disabilities avoid some dangerous situations such as stair climbing or energetic exhaustion. Moreover, the risk of fall injuries can be reduced as well by the identification of the walking ability and potential hazardous locations of a patient.

Subsequently, the approaches described in this report can be improved in several ways. First, the dataset used to train and test the models were composed of healthy young people. A larger dataset with people of all ages, with or without mobility problems is needed to increase the robustness of the model.

Second, the trunk is a good placement in terms of comfort but other location should be evaluated since it has a great impact on the performance of the model.

Third, a lot of variables such as time window, number of features, number of classes, etc. have to be optimised.

Finally, the method and the features presented are a good start and it is now important to build on these foundations to improve the global performance.

Following, the two approaches can benefit from each other if the feature selection was applied to choose the feature set to reduce. This could possibly lead to better or equivalent result with a smaller dimension of feature.

In conclusion, it is now imperative to test the developed model with a bigger and more complete dataset and optimise each step of the two approaches in order to have a final and ready-to-use algorithm.

Signatures

Name	**S**ignature
MERMOUD Paco	

4 Code availability

The dataset and the code of Luo and al. is publicly available in this GitHub repository: `https://github.com/UF-ISE-HSE/UnevenWalkingSurface`.

The code for both approaches analysed during this paper is available in this GitHub repository: `https://github.com/PacM-6610/Gait_classification_using_single_IMU`.

5 References

References

[1] N. Giladi A. Weiss M. Brozgol and J. M. Hausdorff. "Can a single lower trunk body-fixed sensor differentiate between level-walking and stair descent and ascent in older adults?" In: *ResearchGate* (2016). DOI: 10.1016/j.medengphy.2016.07.008.

[2] BIN YIN ALBERTO G. BONOMI ANNELIES H.C. GORIS and KLAAS R. WESTERTERP. "Detection of Type, Duration, and Intensity of Physical Activity Using an Accelerometer". In: *Medicine Science in Sports Exercise* (2009). DOI: 10.1249/MSS.0b013e3181a24536.

[3] M. Arif and A. Kattan. "Physical Activities Monitoring Using Wearable Acceleration Sensors Attached to the Body". In: *Plos one* (2015). DOI: https://doi.org/10.1371/journal.pone.0130851.

[4] Brian Beers. *P-Value*. 2022. URL: https://www.investopedia.com/terms/p/p-value.asp.

[5] Tri-Nhut Do et al. "Personal Dead Reckoning Using IMU Mounted on Upper Torso and Inverted Pendulum Model". In: *IEEE Sensors Journal* 16.21 (2016), pp. 7600–7608. DOI: 10.1109/JSEN.2016.2601937.

[6] K. Khoshelham F. Gu A Kealy and J. Shang. "User-Independent Motion State Recognition Using Smartphone Sensors". In: *MDPI* (2015). DOI: https://doi.org/10.3390/s151229821.

[7] Sampath Kumar Gajawada. *Chi-Square Test for Feature Selection in Machine learning*. 2019. URL: https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223.

[8] Illapha Cuba Gyllensten and Alberto G. Bonomi. "Identifying Types of Physical Activity With a Single Accelerometer: Evaluating Laboratory-trained Algorithms in Daily Life". In: *IEEE Transactions on Biomedical Engineering* 58.9 (2011), pp. 2656–2663. DOI: 10.1109/TBME.2011.2160723.

[9] Adam Hayes. *T-Test*. 2021. URL: https://www.investopedia.com/terms/t/t-test.asp.

[10] Yu-Liang Hsu et al. "Human Daily and Sport Activity Recognition Using a Wearable Inertial Sensor Network". In: *IEEE Access* 6 (2018), pp. 31715–31728. DOI: 10.1109/ACCESS.2018.2839766.

[11] Asif Iqbal et al. "Wearable Internet-of-Things platform for human activity recognition and health care". In: *International Journal of Distributed Sensor Networks* 16.6 (2020), p. 1550147720911561. DOI: 10.1177/1550147720911561. eprint: https://doi.org/10.1177/1550147720911561. URL: https://doi.org/10.1177/1550147720911561.

[12] Insik Jo, Sangbum Lee, and Sejong Oh. "Improved Measures of Redundancy and Relevance for mRMR Feature Selection". In: *Computers* 8.2 (2019). ISSN: 2073-431X. DOI: 10.3390/computers8020042. URL: https://www.mdpi.com/2073-431X/8/2/42.

[13] Xiaofang Kong et al. "Application of Stabilized Numerical Integration Method in Acceleration Sensor Data Processing". In: *IEEE Sensors Journal* 21.6 (2021), pp. 8194–8203. DOI: 10.1109/JSEN.2021.3051193.

[14] Y. Luo, S.M. Coppola, and P.C. Dixon. "A database of human gait performance on irregular and uneven surfaces collected by wearable sensors". In: *Sci Data 7* (2020). DOI: https://doi.org/10.1038/s41597-020-0563-y.

[15] K. Radecka M. Janidarmian A. Roshan Fekr and Z. Zilic. "A Comprehensive Analysis on Wearable Acceleration Sensors in Human Activity Recognition". In: *MDPI* (2017). DOI: https://doi.org/10.3390/s151229821.

[16] MathWorks. *Introduction to Feature Selection*. 2022. URL: https://www.mathworks.com/help/stats/feature-selection.html.

[17] MathWorks. *Select Features for Classifying High-Dimensional Data*. 2022. URL: https://www.mathworks.com/help/stats/selecting-features-for-classifying-high-dimensional-data.html.

[18] L. Elgert S. Hellmers T. Kromke and A. Heinks. "Stair Climb Power Measurements via Inertial Measurement Units - Towards an Unsupervised Assessment of Strength in Domestic Environments". In: *ResearchGate* (2018). DOI: 10.5220/0006543900390047.

[19] Boaz Shmueli. *Multi-Class Metrics Made Simple, Part I: Precision and Recall*. 2019. URL: https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2.

[20] Jeen-Shing Wang and Fang-Chen Chuang. "An Accelerometer-Based Digital Pen With a Trajectory Recognition Algorithm for Handwritten Digit and Gesture Recognition". In: *IEEE Transactions on Industrial Electronics* 59.7 (2012), pp. 2998–3007. DOI: 10.1109/TIE.2011.2167895.

6 Appendix

Laboratory of Movement analysis and Measurement- EPFL
Prof. Kamiar Aminian
Autumn 2021– Semester Project (X credits)
Section: Microtechnics

Student: **Paco MERMOUD**

Subject: Algorithm for detection of walking on stairs using data recoded with a body worn inertial measurement unit

Walking on stairs is a common activity of daily living that is important for functional mobility and independence. Stair walking presents a greater biomechanical challenge relative to walking on level ground because the body center-of-mass (COM) must be raised during ascent and lowered during descent during single limb support while maintaining forward progression and proper foot placement. The automatic detection of walking on stairs (ascend/descent) and ability to distinguish walking on stairs from level walking using data recorded with body worn inertial sensors (accelerometers, gyroscopes) and dedicated data algorithms is important for:

(1) the functional assessment of people with physical impairment (e.g., elderly, Parkinson's, post-stroke…) by *identification of ability to walk on stairs*;

(2) the robust analysis of gait parameters (e.g., speed, stride length, cadence) by *separation of walking on flat from walking on stairs* (gait analysis algorithms should be applied on walking on level ground)

The objective of this project is therefore to develop an algorithm for detection of walking on stairs and separation from walking on flat using data recorded with an inertial sensor worn close to body CoM (lower back/ waist).

The project essentially involves the following steps:

- Literature review, improve background on processing data from wearable inertial sensors
- Record data for development and validation of algorithms (e.g., few subjects monitored during level walking, up/down stairs, IMUs fixed on various body segments, ground truth (synchronized video) for validation)
- Algorithm development and performance assessment
- Write the project report

Available Data and Software:

- Open source database including data recorded with body worn inertial sensors during various daily activities (including walking on ground level and stairs)
- Matlab software
- Matlab functions for detection of level walking using body worn IMU devices (Copyright LMAM, EPFL)

References:

[1] Do, Tri Nhut, et al. "Development of a Wireless Displacement Estimation System Using IMU-based Device."

]2] Do, Tri-Nhut, et al. "Personal dead reckoning using IMU mounted on upper torso and inverted pendulum model." *IEEE Sensors Journal* 16.21 (2016): 7600-7608.

[2] Weiss, Aner, et al. "Can a single lower trunk body-fixed sensor differentiate between level-walking and stair descent and ascent in older adults? Preliminary findings." *Medical engineering & physics* 38.10 (2016): 1146-1151.

[4] Kong, Xiaofang, Wenguang Yang, and Baoming Li. "Application of Stabilized Numerical Integration Method in Acceleration Sensor Data Processing." *IEEE Sensors Journal* 21.6 (2021): 8194-8203.

August, 2021

Dr. Anisoara Ionescu **Prof. Kamiar Aminian**