TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Computer Systems

Janno Sepajõe 205923IACB

Adding functionality of automatic gait cycles detection to gait analysis software

Bachelor's thesis

Supervisor: Jeffrey Andrew Tuhtan
Tenured Associate Professor

Co-supervisor Cecilia Monoli
PhD

Tallinn 2023

TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond
Arvutisüsteemide instituut

Janno Sepajõe 205923IACB

Kõnnitsüklite automaatse tuvastamise funktsiooniaalsuse lisamine kõnnianalüüsi tarkvarale

Bakalaureusetöö

Juhendaja:	Jeffrey Andrew Tuhtan Kaasprofessor tenuuris
Kaasjuhendaja:	Cecilia Monoli PhD

Tallinn 2023

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Janno Sepajõe

15.05.2023

Abstract

Gait analysis is a scientific field that focuses on the systematic study of human walking. It is mainly used in physical therapy, bioengineering and several branches of medicine. By analysing human walking patterns, also called gait cycles, researchers and healthcare workers can find out about possible problems in the person's physical condition.

The primary goal of this thesis was to enhance the functionality of the existing gait analysis software [1], which was developed in an earlier group project, by adding automatic gait cycle detection. The chosen approach to achieve the goal was to design an algorithm that can detect gait cycles and remove all the incorrect peaks. Additionally, as a potential future alternative to the algorithm-based solution, the possibility of using Matrix Profile was explored.

Two different versions of the algorithm were created for gait cycle detection. The first version used the existing peak detection method already used by the software. The second version used a modified version of the method with adjusted parameters for peak detection. The accuracy of both algorithms was tested, and it was found that the second version achieved the highest accuracy. It achieved an accuracy of 90.63% when detecting gait cycles in real-life datasets.

Research on Matrix Profile concluded that although it is possible to detect gait cycles with it, it is not suitable for the gait analysis software without prior rework of the software.

Based on the test results, the second algorithm was implemented in the gait analysis software, adding the functionality of automatic gait cycle detection to it.

This thesis is written in English and is 25 pages long, including 4 chapters, 22 figures and 4 tables.

Annotatsioon

Kõnnitsüklite automaatse tuvastamise funktsiooniaalsuse lisamine kõnnianalüüsi tarkvarale

Kõnnianalüüs on teadusvaldkond, mis keskendub inimese kõndimise süstemaatilisele uurimisele. Seda kasutatakse peamiselt füsioteraapias, biotehnoloogias ja mitmetes meditsiiniharudes. Analüüsides inimeste kõndimismustreid, mida nimetatakse ka kõnnitsükliteks, saavad teadlased ja tervishoiutöötajad teada võimalikest probleemidest inimese füüsilises seisundis.

Lõputöö põhieesmärk oli täiustada varasemas rühmaprojektis välja töötatud olemasoleva kõnnianalüüsi tarkvara [1] funktsionaalsust, lisades sinna automaatse kõnnitsükli tuvastamise. Eesmärgi saavutamiseks valiti algoritmi loomine, mis suudaks tuvastada kõnnitsükleid ja eemaldada kõik ebaõiged tipud. Lisaks uuriti potentsiaalse hilisema alternatiivina algoritmipõhisele lahendusele Matrix Profile'i kasutamise võimalust.

Algoritmist loodi kõnnitsükli tuvastamiseks kaks erinevat versiooni. Esimeses versioonis kasutati olemasolevat tippude tuvastamise meetodit, mida tarkvara juba kasutas. Teises versioonis kasutati meetodi modifitseeritud versiooni tippude tuvastamiseks kohandatud parameetritega. Testiti mõlema algoritmi täpsust ja leiti, et teine versioon saavutas suurima täpsuse. See saavutas reaalsetes tingimustes kogutud andmestike kõnnitsüklite tuvastamisel 90,63% täpsuse.

Matrix Profile'i uuringus jõuti järeldusele, et kuigi sellega on võimalik kõnnitsükleid tuvastada, ei sobi see kõnnianalüüsi tarkvara jaoks ilma olemasoleva tarkvara eelneva ümberehituseta.

Testitulemuste põhjal rakendati kõnnianalüüsi tarkvarasse teine algoritm, lisades sellega automaatse kõnnitsükli tuvastamise funktsionaalsuse.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 25 leheküljel, 4 peatükki, 22 joonist, 4 tabelit.

List of abbreviations and terms

S-G	Savitzki-Golay
IMU	Inertial measurement unit
GUI	*Graphical user interface*
RMSE	*Root mean square error*
UI	*User interface*

Table of contents

List of figures

List of tables

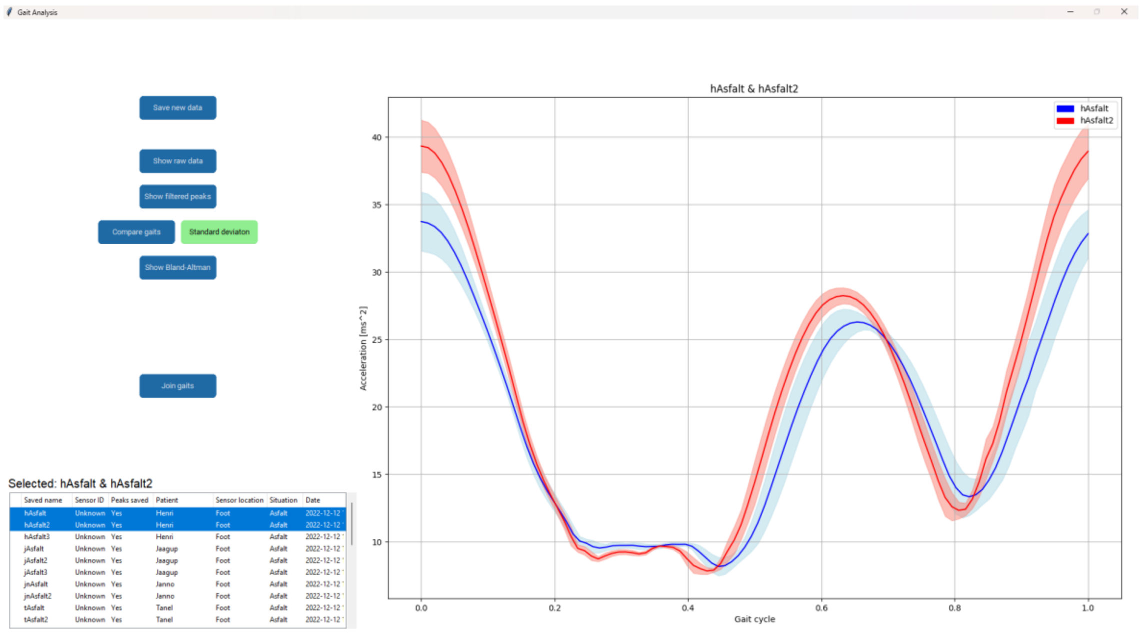
1 Introduction

Gait analysis is a scientific field that focuses on the systematic study of human walking. It is mainly used in physical therapy, bioengineering and several branches of medicine. By analysing human walking patterns, also called gait cycles, researchers and healthcare workers can find out about possible problems in the person's physical condition [2].

During the autumn semester of 2022, I took part in a group project which continued the work of Andrii Boryshkevych's master's thesis which was on the topic of automated gait event detection with the help of wearable sensors. Gait analysis is mostly done either by the researcher or healthcare worker having to physically observe the patient walking or by using expensive infrared cameras in a laboratory environment to record light-reflecting markers that are placed on the patient's body. Andrii's work explored using a TinyTag motion and pressure sensor for collecting the data for gait analysis, which is cost-effective and enables the data to be collected outside in real-life conditions [3].

The group project [1] was based on investigating data collected with the TinyTag sensor in different real-life conditions and developing software to make the analysation of gait cycles faster and easier. The user experience for using Andrii's solution was not very user-friendly. The data collected by the sensor had to be manually modified using Excel every time and there was no graphical user interface. The result of the work done by the group project was software with GUI and different functionalities (see Figure 1.1):

1. Save the collected datasets to the local database for future usage, no need to modify the data, data works straight from the sensor.
2. Manually select the gait cycles by removing peaks on the graph that don't belong to the gait cycles.
3. Compare gait cycles from different datasets.
4. Join datasets together to compare with other joined datasets.
5. Compare 2 datasets using the Bland-Altman plot.



Selected: hAsfalt & hAsfalt2

Saved name	Sensor ID	Peaks saved	Patient	Sensor location	Situation	Date
hAsfalt	Unknown	Yes	Henri	Foot	Asfalt	2022-12-12
hAsfalt2	Unknown	Yes	Henri	Foot	Asfalt	2022-12-12
hAsfalt3	Unknown	Yes	Henri	Foot	Asfalt	2022-12-12
jAsfalt	Unknown	Yes	Jaagup	Foot	Asfalt	2022-12-12
jAsfalt2	Unknown	Yes	Jaagup	Foot	Asfalt	2022-12-12
jAsfalt3	Unknown	Yes	Jaagup	Foot	Asfalt	2022-12-12
jnAsfalt	Unknown	Yes	Janno	Foot	Asfalt	2022-12-12
jnAsfalt2	Unknown	Yes	Janno	Foot	Asfalt	2022-12-12
tAsfalt	Unknown	Yes	Tanel	Foot	Asfalt	2022-12-12
tAsfalt2	Unknown	Yes	Tanel	Foot	Asfalt	2022-12-12

Figure 1.1 Screenshot of the gait analysis software.

Automatic gait cycle detection instead of manual user selection of the gait cycles was one of the possible goals for the group project, but the group ran out of time before attempting to achieve it. I chose to continue that task alone as a topic for my bachelor’s thesis, because I was already familiar with the topic, and I wanted to add the missing functionality to the software.

The thesis is divided into four parts. The introduction consists of an overview of gait analysis and the software for it and a problem statement. The implementation part describes the process of implementing solutions for the gait detection problem. In the Accuracy section, the accuracy of both versions of the algorithm is tested. The summary concludes the thesis, and its results and suggests possible future continuations for the thesis.

1.1 Problem statement

Current research aims to enhance the functionality of the existing software by adding automatic gait cycle detection. The software currently requires the user to manually remove all the non-valid peaks from the graph (see Figure 1.2). While this method may suffice for a small number of datasets, it becomes highly inefficient and time-consuming when dealing with hundreds of datasets. Therefore, developing an automated solution is essential to optimize the user's time utilization.

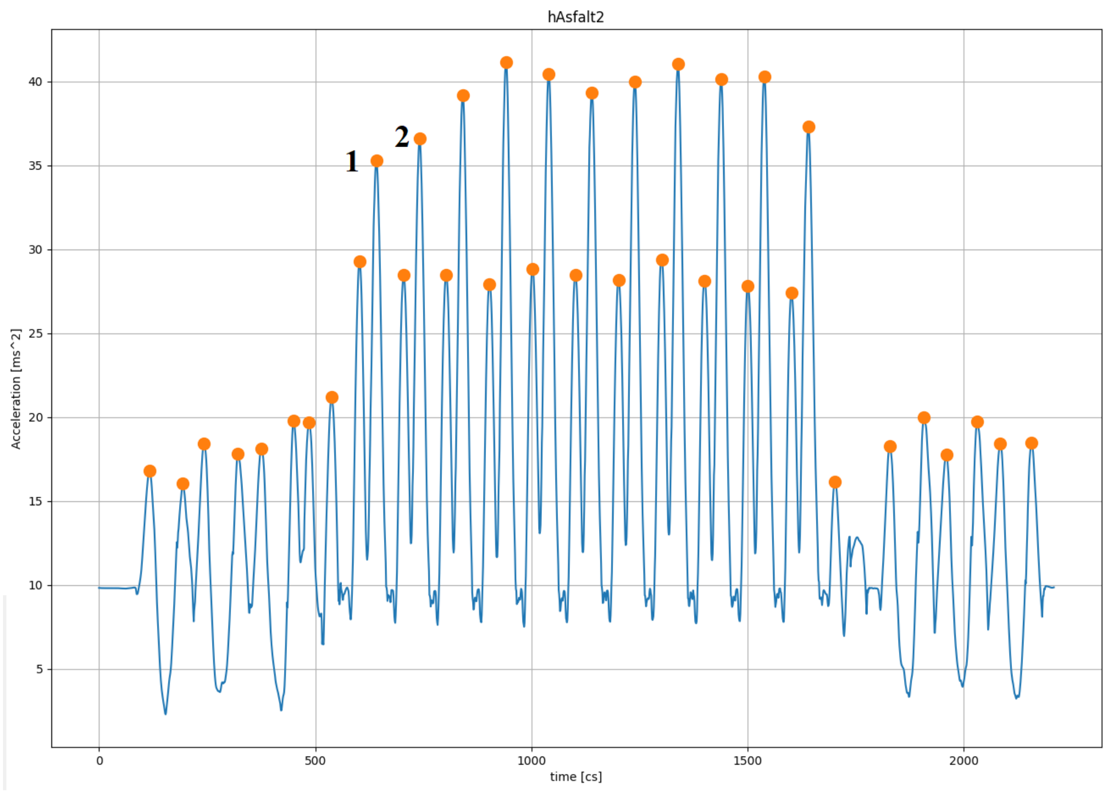


Figure 1.2 Graph of a dataset needing user's manual peak selection.

1 – First heel-strike, 2 – Second heel-strike

This research aims to address the following issues:

1. Developing an automated detection method for human walking gait using IMU time series data captured from the shank, knee, and foot.
2. Comparing the detection accuracy of different methods based on RMSE and phase shift in walking gait.

To achieve automation, two alternative approaches will be explored. The first approach involves designing an algorithm that can identify the peaks that correspond to the gait cycles and remove all the other invalid peaks from the dataset. The second approach involves exploring Matrix Profile as a potential solution to identifying the gait cycles.

2 Implementation

This chapter describes the implementation of the different solutions for automatic gait cycle detection.

2.1 Overview

During the development of the gait analysis software, the student team collected datasets to test its functionalities. For this research, the same datasets will be used for testing the solutions for automatic gait cycle detection. A total of 74 datasets were gathered, involving 4 individuals under 2 different conditions, on asphalt and in snow. Each person took 3 tests per ankle, knee and shank sensor placement in both conditions. Outwalk protocol was used for all the data collection, which required the participants to raise their legs to 90 degrees 3 times before starting to walk and after finishing walking [4], resulting in the data having not only gait walk cycles but also the Outwalk protocol recorded in it. Some of the datasets were too inconsistent for a user to detect the correct peaks in them, and thus these datasets will be excluded from the tests.

To utilize the dataset for other functionalities within the software, users are required to select and save the correct peaks that correspond to the gait cycles. When the user clicks on any peak, multiple options become available to him (see Figure 2.1). The easiest way for users to select the correct peaks is by selecting the peaks that correspond to the first and last gait events in the cycle, all the peaks before the first and after the last event are then removed. That is minimally 2 peak selections for every dataset, if there are some incorrect peaks inside the cycles, the user will have to remove those peaks manually increasing the total amount of interaction needed.

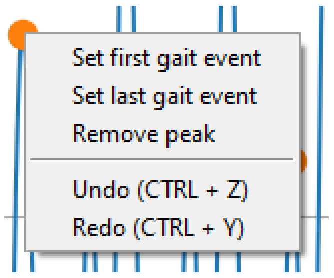


Figure 2.1 Options available to the user when clicking on a peak.

The objective of this research is to eliminate the user’s need to select the gait cycles for datasets. When the user loads a dataset into the software, the gait cycles would automatically be detected and selected. If there was any mistake in the automatic detection, the users can still manually modify the peaks using the available options.

When a dataset is selected for gait cycle selection, the S-G filter is used to smoothen the dataset. The results vary depending on the window length value chosen. Based on the findings of Andrii’s master’s thesis, 39 was concluded as the best window length value that suits datasets from all 3 sensor locations, it is the default value that is used in the software [3]. Users can change the value of the filter’s window length in the UI if they wish to do so. Once the dataset has been smoothed, the peaks are found by using the find_peaks method from an open-source Python library called Scipy [5].

One full gait cycle is defined from a heel strike to the following heel strike (see Figure 1.2) [6]. In the dataset displayed in Figure 1.2, there are a total of 10 gait cycles.

2.2 Algorithm based solution

A possible solution to automating the process of finding gait cycles in the software is to develop an algorithm capable of removing all the incorrect peaks. Every time a new dataset is saved into the software, the algorithm would be run on it to remove all the incorrect peaks in it.

2.2.1 Removal of Outwalk protocol peaks

The Outwalk protocol peaks are usually located lower on the graph than the gait cycle peaks meaning they have smaller acceleration values. Based on that, those peaks can be removed from the dataset by using the average value of all the peaks. The average value of all the peaks is found and all the peaks that have a value less than 80% of the average value are removed from the dataset (see Figure 2.2). Different percentages of the average value were tested, starting from 100% and going lower, but 80% gave in overall the best results. It is a high enough value to remove Outwalk protocol peaks in most cases, unless the peaks have a very similar value as gait cycle peaks, and it is low enough in order not to accidentally remove any gait cycle peaks.

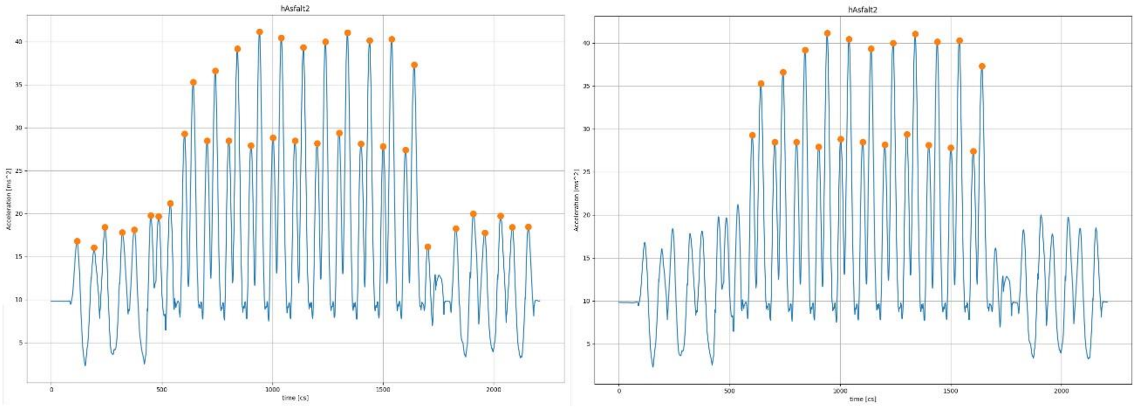


Figure 2.2 Comparison of the dataset before and after Outwalk protocol peaks removal.

In certain datasets, some of the Outwalk protocol peaks have values like those of gait cycle peaks. Therefore, the current function designed to remove the peaks based on their value can fail to eliminate all the Outwalk protocol peaks (see Figure 2.3). This can be attributed to both types of peaks having overlapping values, which makes it harder to distinguish between them using this approach.

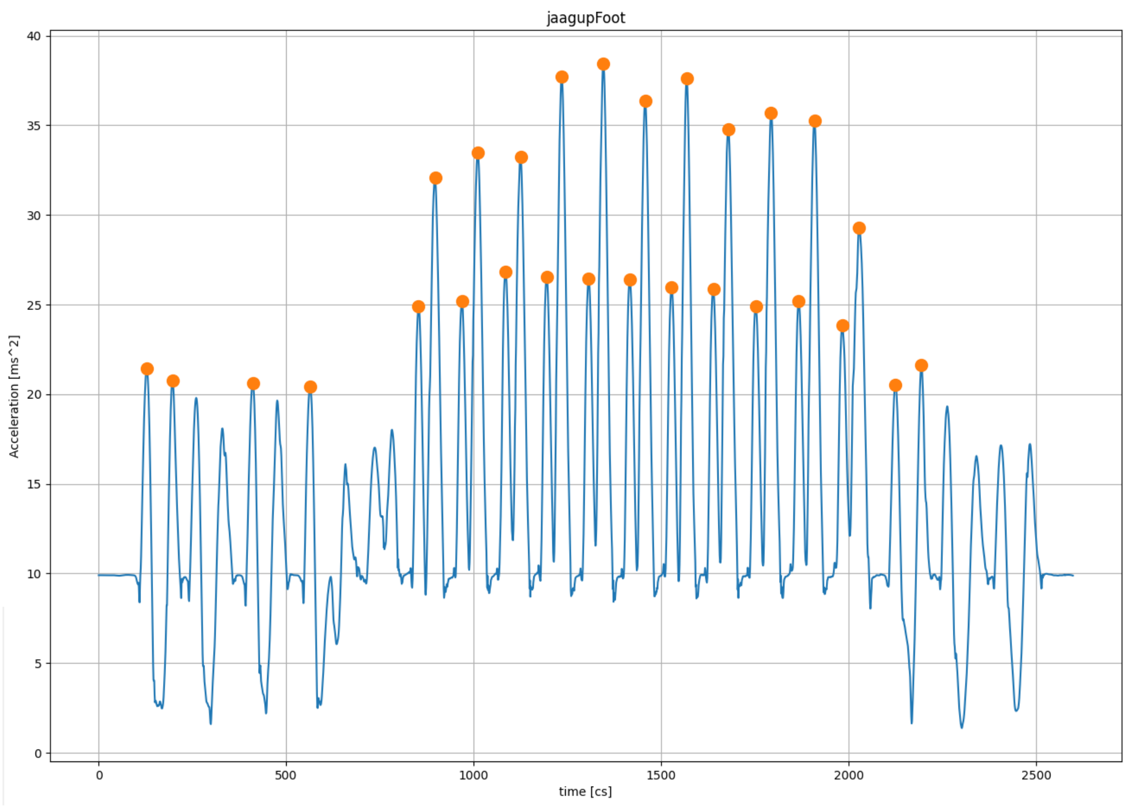


Figure 2.3 Example of Outwalk protocol peaks that the algorithm failed to eliminate.

2.2.2 Removal of duplicate peaks

In some rare instances, datasets may contain multiple peaks occurring at the same location. This can disrupt the functionality of the software if these duplicate peaks are kept. To address this issue, a function was created, which can remove any peaks that are duplicated. It goes through all the peaks, checking if the next peak is closer than 15 centiseconds. If duplicate peaks are detected, the values of the peaks are compared and the peak with a higher acceleration value will be kept. A peak with a higher value is kept over the one with a lower value because the higher value means it is closer to the actual peak of the cycle.

2.2.3 Removal of invalid peaks during gait cycles

Having removed most of the invalid peaks from the dataset by now, the remaining invalid peaks in the dataset include those within the gait cycles and the Outwalk protocol peaks that were not eliminated. Gait cycles are characterized by a specific pattern: a high-value peak followed by a low-value peak followed by another high-value peak. Taking that information into consideration we can confirm the validity of the three peaks belonging to a gait cycle if they follow that rule and remove all the invalid peaks.

To validate the gait cycle peaks a function was implemented. It iterates through the peaks and compares the value difference between the three peaks to the value of the average peak difference between all remaining peaks. If the difference in values between the peaks is too small, it means that there is an invalid peak among them.

Different percentages of average value were tried to see which one worked the best. Starting from 100% of the average value and lowering it by 10% every time. Anything over 70% removed a big part of the correct gait cycles. 40% had fewer correct peaks removed, but it was still too big of a value to compare against. Comparing the 3 peaks' value difference to 30% of the average value difference worked the best, it removed the incorrect peaks inside the gait cycles but did not remove any of the correct gait cycles (see Figure 2.4).

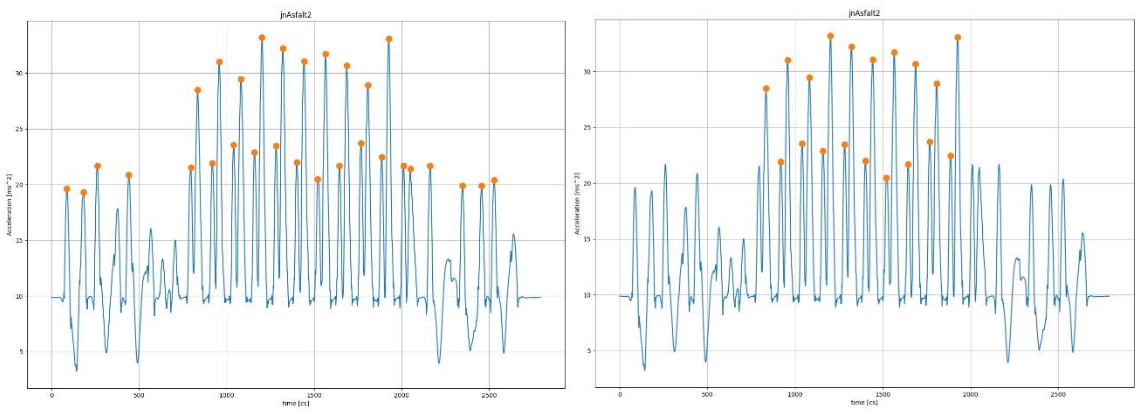


Figure 2.4 Comparison of a dataset before and after removing invalid peaks.

2.2.4 Best S-G window length value

Although 39 is the suggested value for the window length value for the S-G filter [3], in certain cases other window length values can make it easier to distinguish gait cycles. A function for finding the best window length value was implemented, which iterates through all the possible window length values and determines the best one.

The best window length value is the one that has the least amount of gait cycles missed. In other words, the more correct gait cycles detected, the more optimal the window length value is. Since window length values over 49 introduce too dramatic changes into the raw data [3], the function iterates through window length values ranging from 39 to 49.

The function returns the window length value that has the highest number of peaks, and that window length value will be used to detect the gait cycles.

2.2.5 Different sensor placements

Up until now, the algorithm has been tested using datasets where the sensor was located on the foot during the data collection. When applying this solution to datasets involving sensors located either on the shank or on the ankle, the results were very similar compared to the previous findings. However, few of the new datasets had a problem where the Outwalk protocol peaks were as high as, if not higher than, some of the gait cycle peaks. Therefore, the algorithm failed to remove them (see Figure 2.5).

To address this issue, another function was implemented in the algorithm that detects the start and end of the gait cycles, removing any peaks that occur before or after that since they don’t belong to the gait cycles. It does that by looking for the first peak in the time-series data whose value is over the average value of the remaining peaks. Using the average value of remaining peaks had few occurrences where it resulted in one gait cycle going missing from the beginning, lowering the average value to 95% of the original value fixed that issue. The function uses similar logic to detect the end of gait cycles. Once the beginning of gait cycles is detected, it starts looking for a peak whose value is under the average value of the remaining peaks. Using the base average value of peaks had the same issue as using a base average value when looking for the beginning. If the average value is too high, it can cause one gait cycle to go missing at the end of the gait cycles. Lowering the average value was tried to find the most suitable percentage of base value that worked the best, which ended up being 77% of the original average value of the remaining peaks.

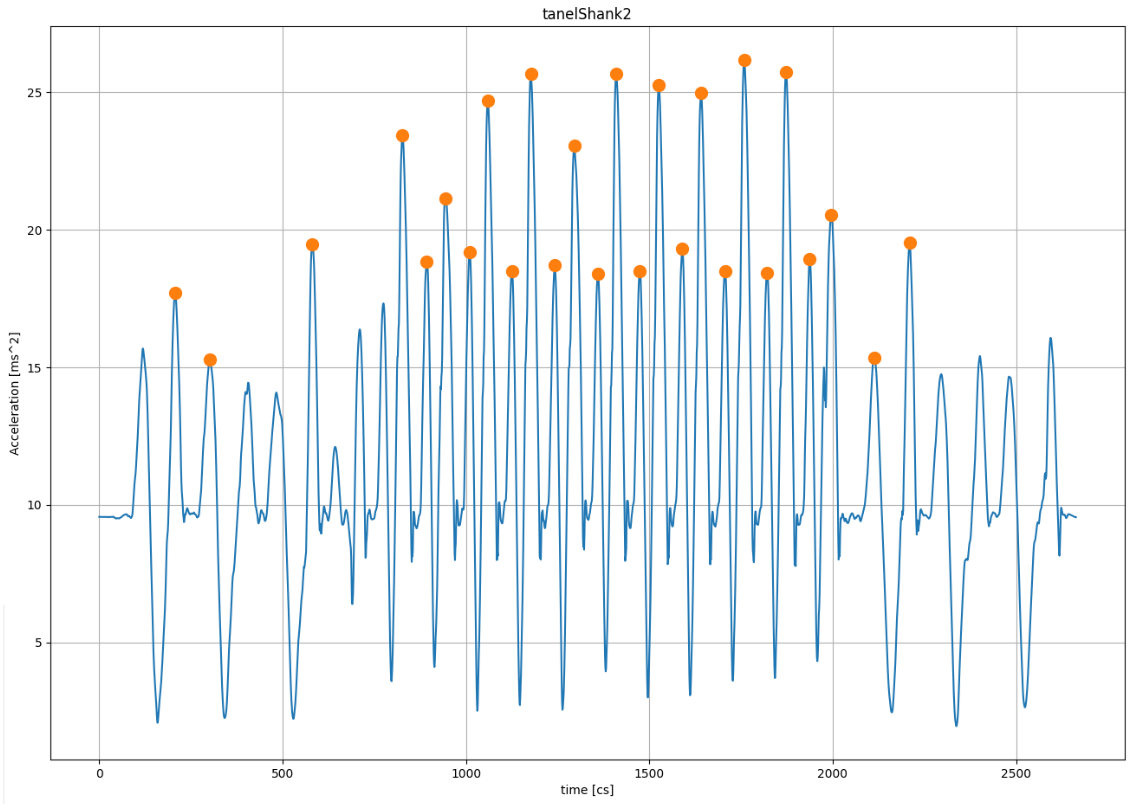


Figure 2.5 Shank dataset graph showing the invalid peaks that the algorithm failed to eliminate.

2.3 Using Scipy to remove invalid peaks

The software is using only one parameter with the find_peaks method from Scipy. It is possible to change the parameters used in the find_peaks method, potentially reducing the number of peaks that need to be removed and increasing the accuracy of gait cycle detection.

Not all the parameters in the method are useful in this case. We are interested in the parameters that are beneficial for removing the peaks caused by noise and handling multiple peaks in the same location. The following parameters, along with their respective functionalities, will be used [5], [7]:

1. Height: it is the only parameter that is used when the software uses the method currently. By setting the height value, any peaks that are lower than that value will be ignored by the method.
2. Distance: it is the minimum horizontal distance between the peaks, in our case minimum time. If peaks are located too close to each other, only the one with the highest vertical value will be kept. This is beneficial for removing multiple peaks in one location.
3. Prominence: it is the minimum prominence value a peak must have. The prominence of a peak measures how much it stands out from the surrounding baseline of the signal. It can help remove peaks caused by noise.

2.3.1 Finding suitable parameter values

In Andrii's master's thesis, a height value of 12 or 13 is selected depending on the sensor's location [3]. Different height parameter values were tried and increasing the height parameter to 16 did not affect any gait cycle peaks while removing some of the Outwalk protocol peaks (see Figure 2.6).

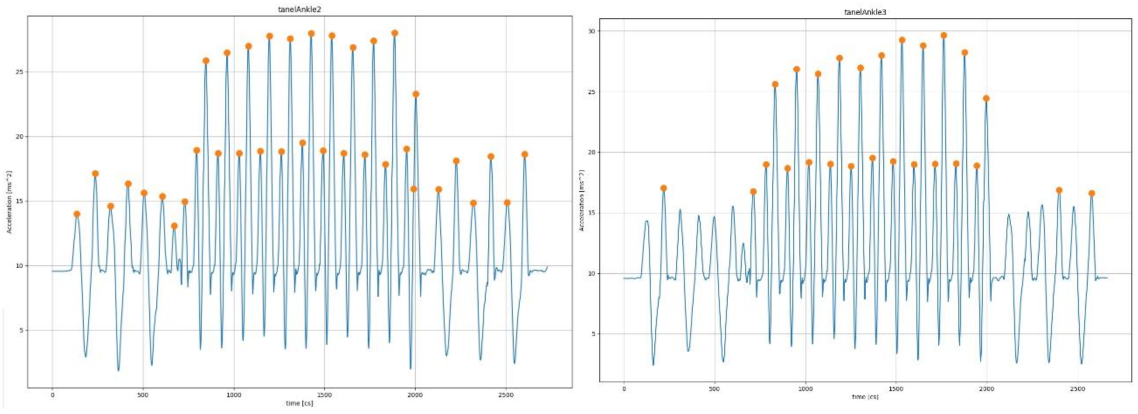


Figure 2.6 Comparison of height parameter values of 13 and 16.

Using the same distance parameter value of 15 centiseconds as is used in the algorithm can remove the occurrences of multiple peaks in the same area (see Figure 2.7).

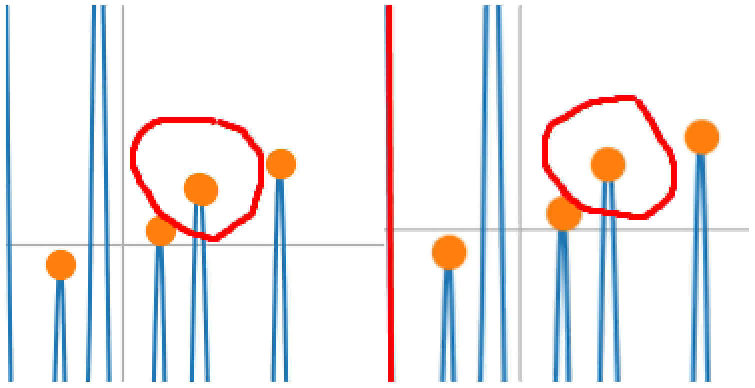


Figure 2.7 Picture showing the additional peak in the same area being removed by the method.

Increasingly larger prominence values were tried. When the value is too small, it does not affect the peaks. When the value is too high, it also removes the gait cycle peaks (see Figure 2.8). The largest prominence value that didn't affect any gait cycles was 9, therefore that will be the prominence value that is used.

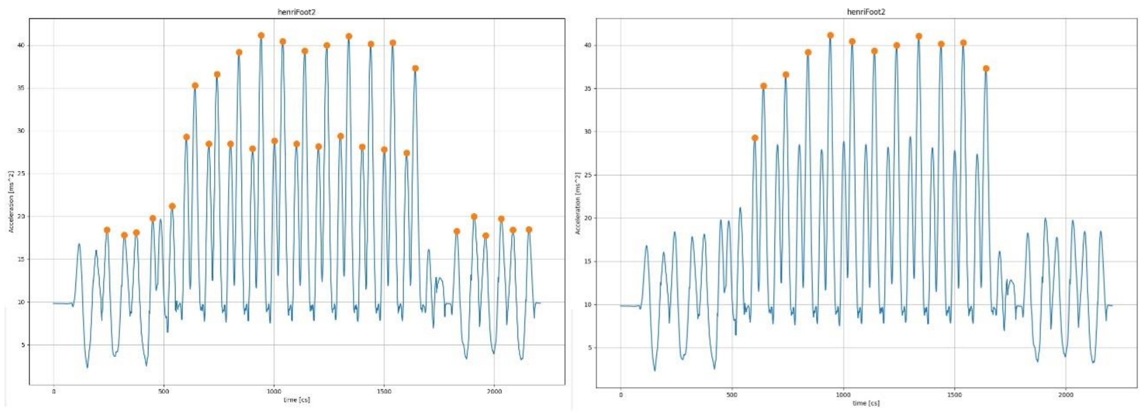


Figure 2.8 Comparison of 2 different prominence values of 9 and 17.

2.3.2 Combining the algorithm and modified find_peaks method

By using the find_peaks method with modified parameters, instead of the one in use in the software, the accuracy of gait cycle detection should go higher. Some of the algorithm's methods and what the parameters are doing are overlapping, so they could be disabled. The distance parameter is taking care of the multiple peaks in the same location problem, so the algorithm does not need to remove them anymore. When testing this new combination on datasets where the algorithm was struggling with removing the invalid peaks, it managed to remove them successfully (see Figure 2.9).

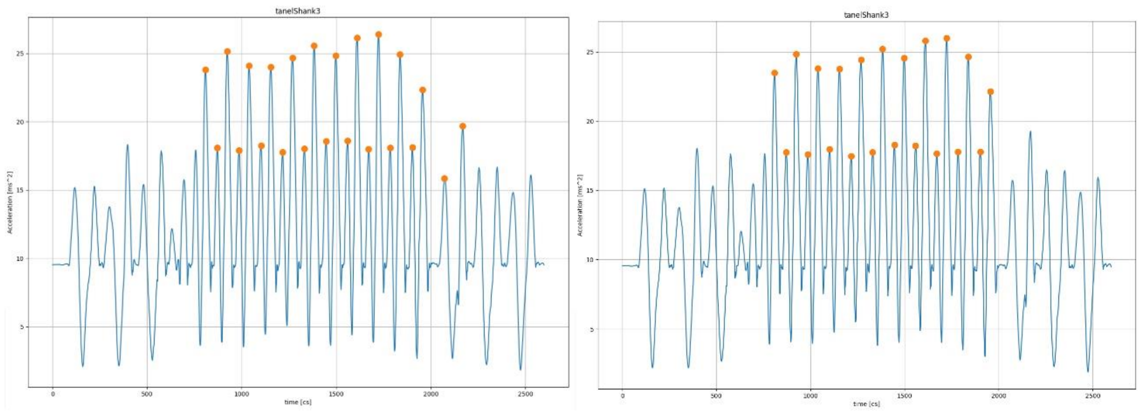


Figure 2.9 Comparison of the detected gait cycles by the two versions of the algorithm.

2.4 Matrix profile

After finishing exploring the algorithm-based solution for gait detection automation, I had some time left over. I decided to investigate using Matrix Profile as a possible alternative solution to the problem, for future usage.

The Matrix profile is a data structure used for time series analysis developed by Eamonn Keogh and Abdullah Mueen. It can be used to find repeated patterns (motifs) and anomalies (discords) in time-series data [8]. In this case, we are interested in its ability to find the motifs, which would be the gait cycles.

2.4.1 Implementation

Using Matrix Profile consists of three steps [9]:

1. Computing the Matrix Profile
2. Discovering the repeating patterns or anomalies
3. Visualizing the results

To complete the first 2 steps, a Python library called Stumpy is used. A method called stump from stumpy is used for computing the Matrix Profile, it needs the time-series data as an input and a window size. The window size is different for every project depending on the motif sizes [10], [11].

To discover all the repeating patterns, each motif's similarity to the nearest neighbour is checked and the ones that are under the similarity threshold are selected. When visualizing those motifs, a problem becomes visible. The Matrix profile is also detecting Outwalk protocol cycles since they are also repeating patterns (see Figure 2.10).

To fix the problem, there was going to have to be another check when selecting the motifs. Looking at the graph, one way to do it would be to check for the highest part of the motif. Gait cycle motifs have always a higher maximum value compared to the Outwalk protocol cycles, so it is a good way to eliminate them. After calculating the maximum value for each motif, only the motifs that have a maximum value over 25 are selected. This only keeps the gait cycle motifs (see Figure 2.11).

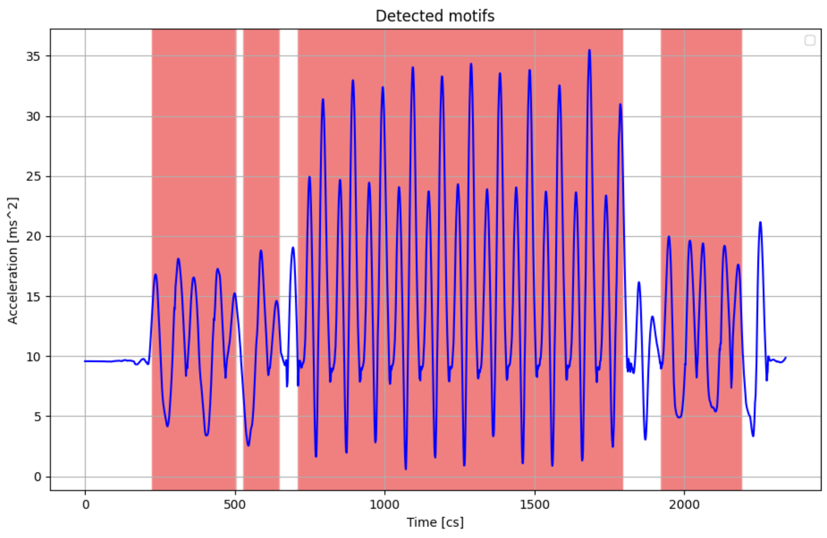


Figure 2.10 Graph showing Matrix Profile detecting Outwalk protocol cycles.

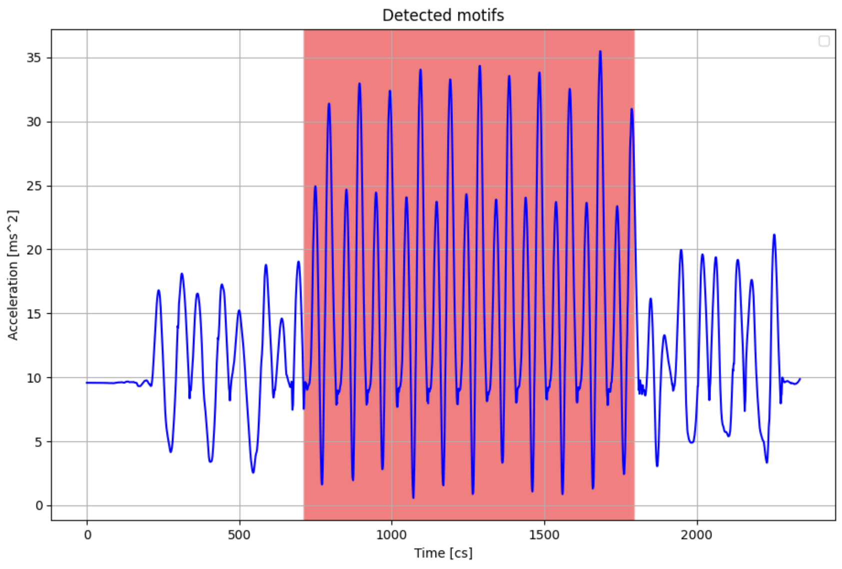


Figure 2.11 Graph of Matrix Profile detecting only gait cycle motifs.

2.4.2 Compatibility with the software

The software's all functionalities depend on the datasets having peakes. Although Matrix Profile can successfully detect the gait cycle motifs, to make that information usable for the software, peaks would have to be added to the motifs.

When the find_peaks method from Scipy is used on the motifs, although the need for removing Outwalk protocol peaks is gone, the need for removing incorrect peaks that can occur during gait cycles remains. To remove those, the peak detection algorithm would have to be run on the dataset, making the usage of Matrix Profile trivial because the algorithm can remove the Outwalk protocol peaks on its own.

For the usage of Matrix Profile to make sense, the software would have to be reworked to not depend on the peaks anymore. This is a possible continuation of the software project.

3 Accuracy

This chapter talks about how the accuracy of the solutions was tested and the results of them.

3.1 Accuracy based on the available real-life datasets

As previously mentioned, 74 datasets are available for testing the solutions, with half of them collected on asphalt and the other half in snow. However, most of the datasets that were collected in the snow are very inconsistent (see Figure 3.1). These inconsistencies make it challenging for both the algorithm and human users to accurately identify gait cycles. Therefore, the snow datasets will not be used for testing. That leaves 36 datasets that can be used to test the accuracy. Among the four total participants involved in the data collection, two of them had a few unusable datasets that were collected on the asphalt (see Figure 3.2), those datasets will also not be used.

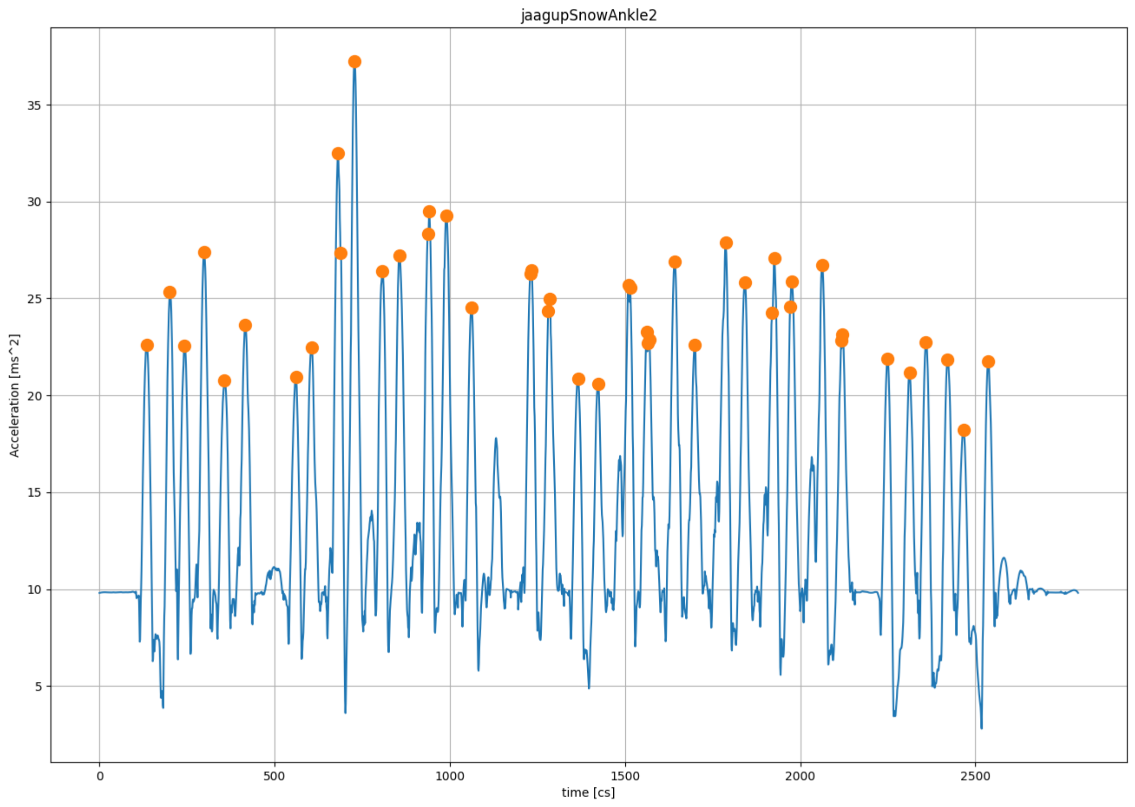


Figure 3.1 Example of a dataset collected in snow that is not suitable for testing the solutions.

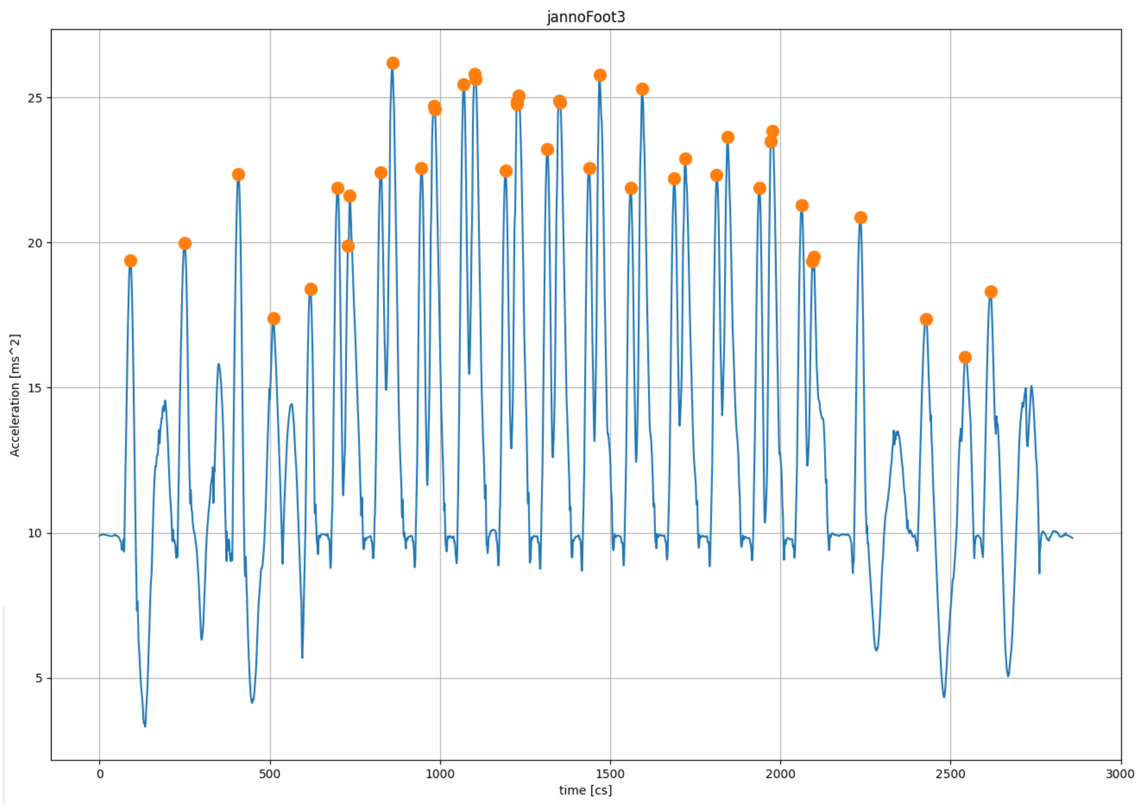


Figure 3.2 Example of a dataset collected on asphalt that is not used for testing.

3.1.1 Algorithm without modified find_peaks

The algorithm was applied to all available datasets for testing purposes. The accuracy of the algorithm's results was evaluated by comparing the number of detected gait cycles with the number that a user would identify from each specific dataset.

Without searching for the best S-G window length value, the algorithm accurately detected the correct gait cycles with 25 datasets out of 32, resulting in an accuracy of 78.13% (see Table 3.1). Out of the 7 incorrect detections, only 3 included incorrect peaks not being removed and the other 4 were missing a gait cycle.

When a gait cycle is not detected, the other detected gait cycles are still usable in the software meaning that user interaction is not needed for the dataset to be usable. If incorrect peaks are included in the dataset along with the gait cycle peaks, it will make the dataset inaccurate and unusable in the software, resulting in the need for user manual interaction.

	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**
Ankle	3/3 datasets	2/3 datasets	3/3 datasets	3/3 datasets
Foot	3/3 datasets	3/3 datasets	1/2 datasets	3/3 datasets
Shank	3/3 datasets	1/3 datasets	0/1 dataset	0/2 datasets
Correct detections	9/9 datasets	6/9 datasets	4/6 datasets	6/8 datasets
Correct detections in total for all the subjects			25/32 datasets	

Table 3.1 Algorithm accuracy per sensor location without finding the best S-G window length value.

When the algorithm searches for the best possible S-G window length value, it accurately detects 26 datasets out of 32, which is 81.25% (see Table 3.2). That is a better result than when not finding the window length value.

	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**
Ankle	3/3 datasets	3/3 datasets	3/3 datasets	3/3 datasets
Foot	3/3 datasets	3/3 datasets	1/2 datasets	3/3 datasets
Shank	3/3 datasets	1/3 datasets	0/1 dataset	0/2 datasets
Correct detections	9/9 datasets	7/9 datasets	4/6 datasets	6/8 datasets
Correct detections in total for all the subjects			26/32 datasets	

Table 3.2 Algorithm accuracy per sensor location with finding the best S-G window length value.

3.1.2 Combination of algorithm and modified find_peaks

The algorithm exhibits a notable increase in accuracy when using the modified parameters for the find_peaks method. Without searching for the best possible S-G window length value, the algorithm detected the gait cycles in 28 out of 32 datasets correctly (see Table 3.3) resulting in an accuracy of 87.5%, which is almost a 10% increase in accuracy compared to the accuracy without modified parameters. Out of the 4 incorrectly detected datasets, only one had incorrect peaks included in the detection.

	Subject 1	Subject 2	Subject 3	Subject 4
Ankle	3/3 datasets	2/3 datasets	3/3 datasets	3/3 datasets
Foot	3/3 datasets	3/3 datasets	1/2 datasets	3/3 datasets
Shank	3/3 datasets	2/3 datasets	1/1 dataset	1/2 datasets
Correct detections	9/9 datasets	7/9 datasets	5/6 datasets	7/8 datasets
Correct detections in total for all the subjects			28/32 datasets	

Table 3.3 Algorithm accuracy per sensor location without finding the best S-G window length value when using modified find_peaks.

There is also an increase in accuracy when the algorithm searches for the best S-G window length value. The algorithm detected gait cycles accurately in 29 datasets out of 32 datasets, which is 90.63% (see Table 3.4). There is again almost a 10% increase in the accuracy compared to the results where the best S-G window length value was used without the modified find_peaks method.

	Subject 1	Subject 2	Subject 3	Subject 4
Ankle	3/3 datasets	3/3 datasets	3/3 datasets	3/3 datasets
Foot	3/3 datasets	3/3 datasets	1/2 datasets	3/3 datasets
Shank	3/3 datasets	3/3 datasets	0/1 dataset	1/2 datasets
Correct detections	9/9 datasets	9/9 datasets	4/6 datasets	7/8 datasets
Correct detections in total for all the subjects			29/32 datasets	

Table 3.4 Algorithm accuracy per sensor location with finding the best S-G window length value when using modified find_peaks.

3.2 Comparison of detection accuracy based on RMSE and phase shift

To compare the accuracy based on RMSE and phase shift a ground truth dataset is needed. Using a Python library called NumPy it is possible to generate a perfect sinusoid to act as the ground truth dataset [12]. The generated sinusoid consists of 4 gait cycles resulting in the expected number of peaks always being exactly 9 (see Figure 3.3). Anything over or under 9 means that an incorrect peak has been detected or a correct one has been removed.

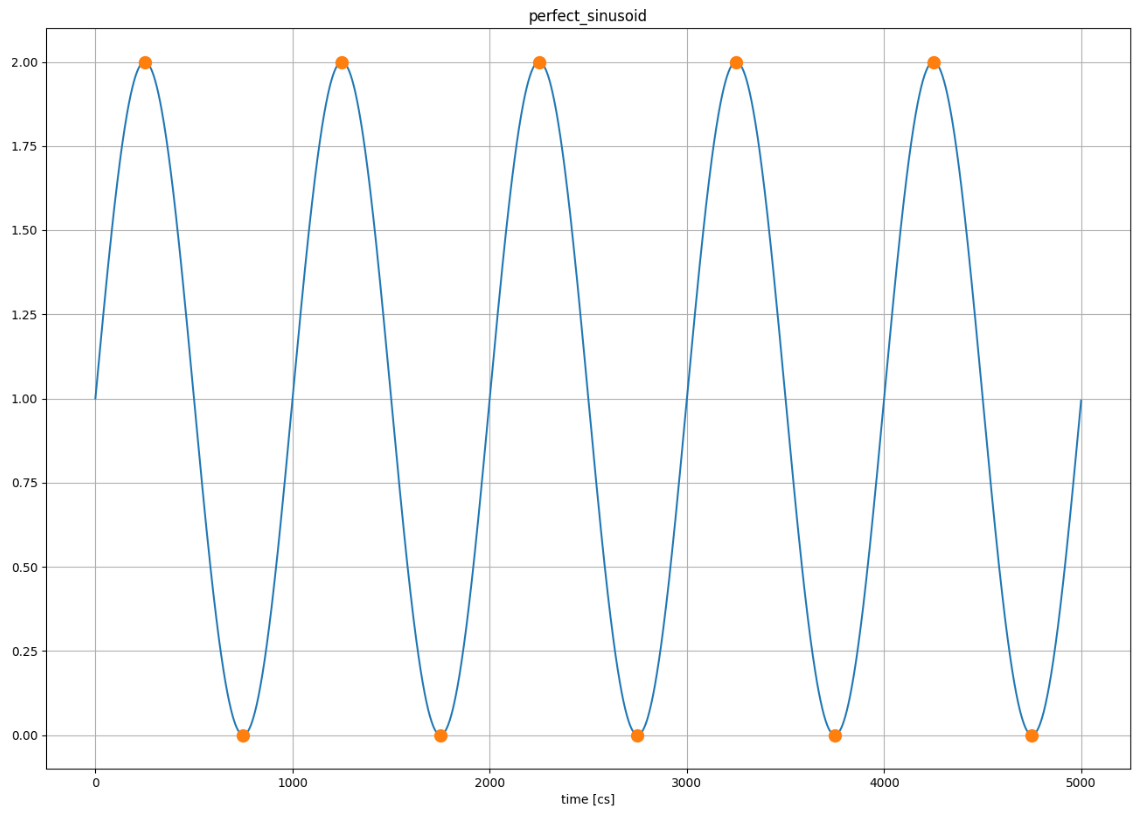


Figure 3.3 Graph showing the generated perfect sinusoid with 4 gait cycles.

Different random noise is added to the perfect sinusoid using Gaussian distribution with a zero-mean value. The standard deviation is systematically increased to see the effect on gait cycle detection. In addition, an increasingly larger phase shift will be added to the sinusoid to test the algorithm's resilience to such variations. The standard deviation values range from 0.01 to 1, with an increment of 0.01, while phase shift values range from 1 to 100, with an increment of 1.

The results will be shown as a graph of calculated RMSE values, where the x-axis is the standard deviation value or phase shift value or both and the y-axis is the RMSE value (see Figure 3.4, Figure 3.5). RMSE value was calculated based on 9 being the expected value and the number of peaks with different amounts of noise as the actual values.

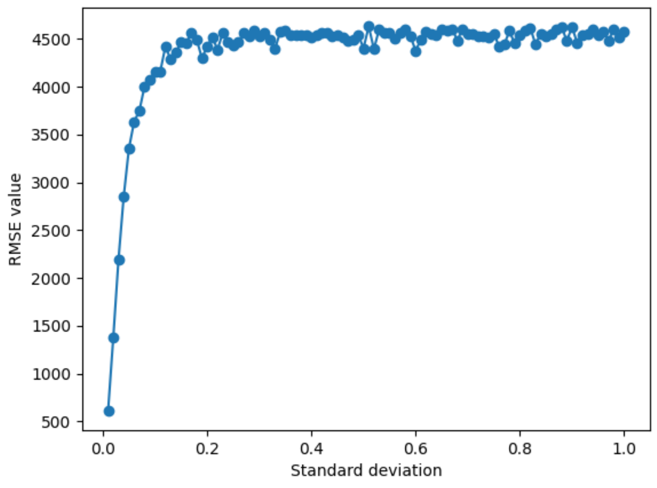


Figure 3.4 Graph showing the RMSE values with different standard deviations when there is no peak removal applied.

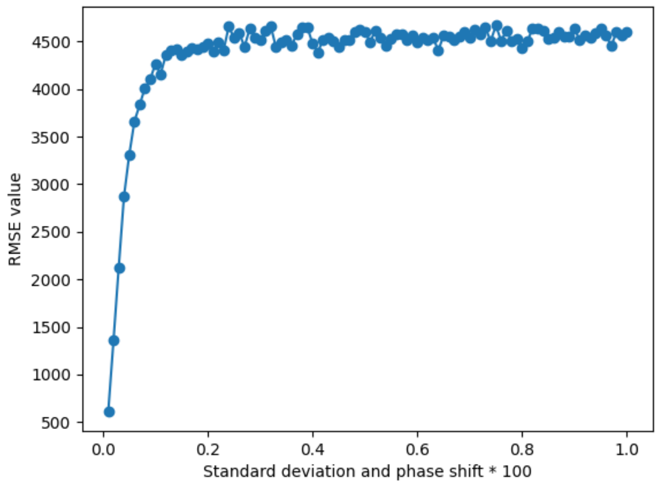


Figure 3.5 Graph showing RMSE values with different standard deviations and phase shift added when there is no peak removal applied.

3.2.1 Algorithm without modified find_peaks

The algorithm showed good accuracy in real-life dataset gait cycle detection, but it struggled with the noisy ground truth dataset. It only showed correct gait cycles when there was no random noise applied to the sinusoid (see Figure 3.6). That was probably caused by the algorithm being based on real-life datasets where noise like in this test did not occur, therefore the algorithm is not capable of handling this amount of random noise. Adding phase shift on top of random noise made minimal changes to the result (see Figure 3.7).

Compared to the results with no peak removal applied, the algorithm was able to remove a large amount of the invalid peaks, but constantly was not removing a similar number of invalid peaks.

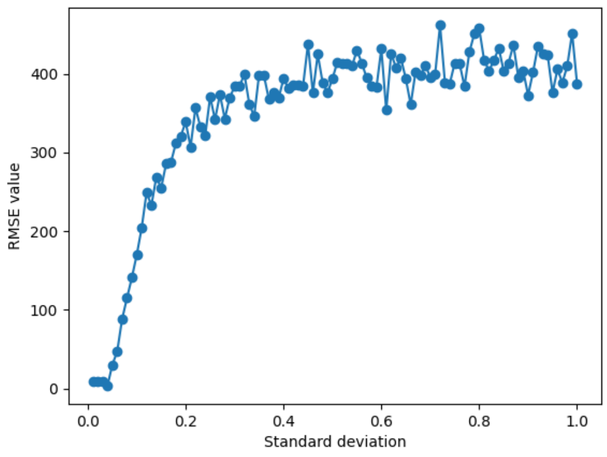


Figure 3.6 Graph showing the RMSE values with different standard deviations when the gait cycle detection algorithm is applied.

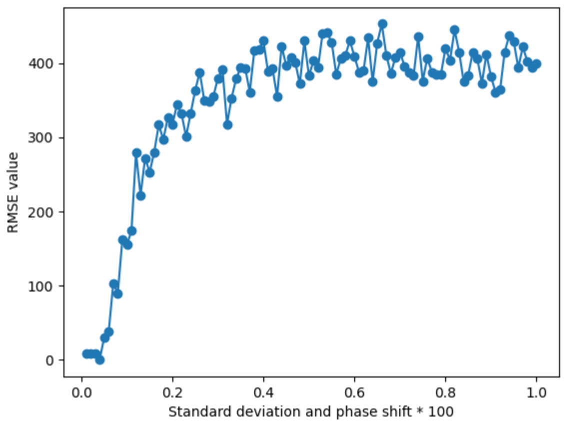


Figure 3.7 Graph showing RMSE values with different standard deviations and phase shift added when the gait cycle detection algorithm is applied.

3.2.2 Combination of algorithm and modified find_peaks

The algorithm with the modified find_peaks method was able to handle the noise better than the counterpart (see Figure 3.8). The RMSE value stayed near 0 until the standard deviation of 0.25 after which it started to rise until reaching and staying in its worst state from near 0.9 to a standard deviation of 1. Adding phase shift on top of random noise resulted in minimal changes to the RMSE values, like the previous results.

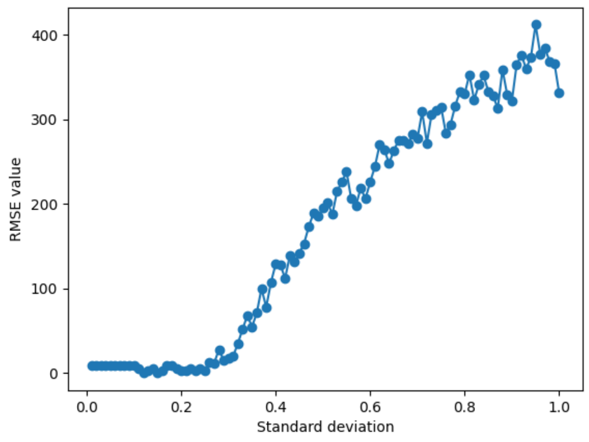


Figure 3.8 Graph showing the RMSE values with different standard deviations when the gait cycle detection algorithm with modified find_peaks is applied.

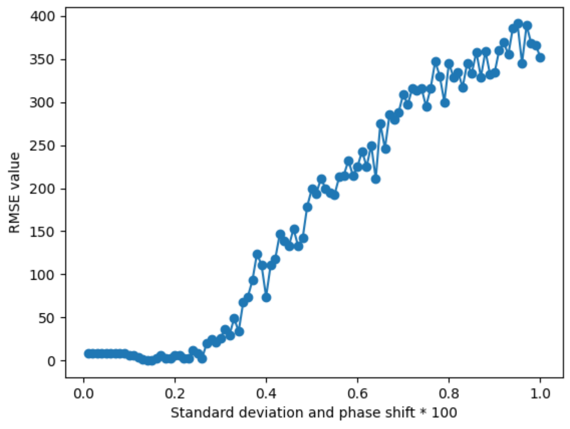


Figure 3.9 Graph showing RMSE values with different standard deviations and phase shift added when the gait cycle detection algorithm with modified find_peaks is applied.

3.3 Results

The tests conducted on real-life datasets show that finding the best S-G window length value does influence the accuracy of gait detection. When you increase the window length value the value difference between the top and bottom peaks of the gait cycles becomes bigger. The difference in accuracy is caused by the gait cycles that were missed when using the default S-G window length value becoming more prominent and easier to detect when using the most optimal window length value.

Without searching for the best S-G filter window length value, the algorithm achieved an accuracy of 78.13% (25 out of 32 datasets), while when the algorithm searched for the best S-G window length value, it achieved a higher accuracy of 81.25% (26 out of 32 datasets).

By using the find_peaks method's parameters in addition to relying on the algorithm for eliminating invalid peaks, the number of peaks is reduced before the algorithm detects correct peaks them. When there are fewer peaks to determine whether they belong to the gait cycle or not, the chance of incorrect detections is also reduced.

The algorithm demonstrates nearly a 10% increase in accuracy both without and with the best S-G filter window length value when using modified find_peaks method parameters. When the algorithm searched for the best S-G window length value and used the modified find_peaks method parameters, it achieved the highest accuracy of 90.63% (29 out of 32 datasets).

Based on the tests involving inducing random noise to a perfect sinusoid, the first solution was able to eliminate some of the invalid peaks caused by the random noise but never was near the correct number of peaks. In contrast, the second solution was effective at eliminating nearly all the invalid peaks up to a standard deviation value of 0.25, beyond which its ability to eliminate the invalid peaks started diminishing.

Based on the results obtained from both tests, the solution that yields the highest accuracy involves a combination of the algorithm searching for the best S-G window length value and using modified parameters for the find_peaks method.

4 Summary

The primary goal of this thesis was to enhance the functionality of the existing gait analysis software by adding automatic gait cycle detection. The existing software required manual selection of gait cycles, which was time-consuming and inefficient when large amounts of datasets are involved. The approach to solving the problem was designing an algorithm to detect gait cycles and remove all the incorrect ones. In addition, Matrix Profile was investigated as a possible future alternative to the algorithm-based solution.

Two different versions of the algorithm were developed. One that is using the method that the software was using before to find the peaks in the time-series data, and the other one is using the method with modified parameters to find the peaks. The accuracy of both algorithms was tested. The best accuracy in detecting gait cycles in real-life datasets was by the algorithm that is using peaks method with modified parameters and has the best S-G window length value finding enabled. It achieved the highest accuracy of 90.63% (29 out of 32 datasets). Using ground truth data and random noise to compare the two algorithms, the RMSE showed that the second algorithm has better resistance to random noise. Therefore, since both tests had the second algorithm as the more accurate one, that was the one chosen to be implemented into the software.

Due to some time being left over after the completion of the algorithm-based solution to the goal of this thesis, Matrix Profile was researched. It was investigated whether Matrix Profile could be a possible alternative in the future to the algorithm for detecting gait cycles. The investigation concluded that it is possible to detect gait cycles with Matrix Profile by finding the motifs of the gait cycles. The gait analysis software is currently designed to use peaks in all its functionalities, which results in the found motifs of gait cycles not being usable in the software. The software would have to be reworked to make the functionalities of the software not be depending on peaks, for Matrix Profile to be a possible solution for automatic gait detection. This could be a future continuation of this software project.

Another potential continuation for the project could be to find a way to incorporate an alert system into the software that tells the user if a dataset needs manual editing. One approach that could be investigated is asking the user whether all the datasets they are entering have the same number of steps or in other words gait cycles in them. The most occurring number of gait cycles would be the correct number of gait cycles. It would need to be researched, whether this is possible and how accurately it could tell the user when they need to manually edit the dataset.

In conclusion, all the issues stated in the problem statement were solved and the main goal of this thesis was met. An algorithm that can detect gait cycles and remove invalid peaks was developed and automatic gait cycle detection was implemented to the gait analysis software.

4 Kokkuvõte

Käesoleva lõputöö esmane eesmärk oli täiustada olemasoleva kõnnianalüüsi tarkvara funktsionaalsust, lisades sinna automaatse kõnnitsükli tuvastamise. Olemasolev tarkvara nõudis kõnnitsüklite käsitsi valimist, mis oli aeganõudev ja ebaefektiivne, kui tegemist on suure hulga andmekogumitega. Probleemi lahendamisele läheneti algoritmi väljatöötamisega, mis suudaks kõndimistsükleid tuvastastada ja valed tipud eemaldada. Lisaks uuriti Matrix Profile'i kui võimalikku hilisemat alternatiivi algoritmipõhisele lahendusele.

Algoritmist töötati välja kaks erinevat versiooni. Üks, mis kasutab meetodit, mida tarkvara varem kasutas aegridade andmete tippude leidmiseks, ning teine, mis kasutab tippude leidmiseks modifitseeritud parameetritega meetodit. Testiti mõlema algoritmi täpsust. Reaalelu andmehulkade kõnnitsüklite tuvastamisel oli parim täpsus algoritmil, mis kasutab muudetud parameetritega tippude leidmise meetodit ja millel on olemas parim S-G akna pikkuse väärtuse leidmine. See saavutas suurima täpsuse 90,63% (29 andmestikku 32-st). Kasutades kahe algoritmi võrdlemiseks perfektset sinusoidi ja juhuslikku müra, näitas RMSE, et teisel algoritmil on parem vastupanu juhuslikule mürale. Seega, kuna mõlemas testis oli täpsem teine algoritm, valiti see tarkvarasse lisamiseks.

Kuna selle lõputöö eesmärgi saavutamisel jäi algoritmipõhise lahenduse valmimisel veidi aega üle, vaadati lisaks Matrix Profile'i. Uuriti, kas Matrix Profile võiks tulevikus olla võimalik alternatiiv kõnnitsüklite tuvastamise algoritmile. Uurimisel jõuti järeldusele, et Matrix Profile'iga on võimalik kõnnakutsükleid tuvastada, leides kõnnitsüklite motiivid. Kõnnianalüüsi tarkvara on praegu loodud kasutama tippe kõigis oma funktsioonides, mistõttu leitud kõnnitsüklite motiivid ei ole tarkvaras kasutatavad. Tarkvara tuleks ümber töötada, et tarkvara funktsioonid ei sõltuks tippudest. See võib olla selle tarkvaraprojekti tulevane jätk.

Projekti teine potentsiaalne jätk võiks olla tarkvarale hoiatussüsteemi lisamine, mis teataks kasutajale, kui andmestik vajab käsitsi redigeerimist. Üks võimalik lähenemisviis, mida võiks uurida, on kasutajalt küsida, kas kõigis hetkel sisestatavates andmekogumites on sama arv samme või teisisõnu kõnnitsükleid. Kui jah, siis kõige sagedamini esinev kõnnitsüklite arv oleks õige kõnnitsüklite arv. Seda tuleks põhjalikult uurida, kas see on võimalik ja kui suur oleks selle täpsus kasutajale teada andmises, millal on vaja andmestikku käsitsi redigeerida.

Kokkuvõttes said kõik probleemipüstituses välja toodud probleemid lahendatud ja töö põhieesmärk täidetud. Töötati välja algoritm, mis suudab tuvastada kõnnitsükleid ja eemaldada valed tipud, ning automaatne kõnnitsükli tuvastamine rakendati kõnnianalüüsi tarkvarasse.

Bibliography

[1] T. Lentso, J. Sepajõe and J. Vaht, "Gait Analysis software Github repository," [Online]. Available: https://github.com/DxMickey/Gait-Analysis-Project.

[2] W. MW, Gait analysis: an introduction, Butterworth-Heinemann, 2014.

[3] A. Boryshkevych, "Automated gait event detection with the help of wearable sensors" [Online]. Available: https://digikogu.taltech.ee/et/Item/2213c011-2d05-468e-86cb-a2902551d0df

[4] A. F. P. G. M. R. A. C. A. F. Andrea Giovanni Cutti, "'Outwalk': a protocol for clinical gait analysis based on inertial and magnetic sensors," *Medical & biological engineering & computing ,* no. 48, p. 17–25, 2010.

[5] "Scipy," [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.find_peaks.html. [Accessed 10 04 2023].

[6] J. Kawalec, Mechanical Testing of Orthopaedic Implants, Woodhead Publishing, 2017, pp. 231-253.

[7] "Python Guides," [Online]. Available: https://pythonguides.com/scipy-find-peaks/. [Accessed 10 04 2023].

[8] "Towards Data Science," [Online]. Available: https://towardsdatascience.com/introduction-to-matrix-profiles-5568f3375d90. [Accessed 15 04 2023].

[9] "Towards Data Science," [Online]. Available: https://towardsdatascience.com/how-to-painlessly-analyze-your-time-series-f52dab7ea80d. [Accessed 16 04 2023].

[10] "STUMPY," [Online]. Available: https://stumpy.readthedocs.io/en/latest/Tutorial_STUMPY_Basics.html#Analyzing-Motifs-and-Anomalies-with-STUMP. [Accessed 16 04 2023].

[11] "STUMPY," [Online]. Available: https://stumpy.readthedocs.io/en/latest/api.html#stumpy.stump. [Accessed 16 04 2023].

[12] "NumPy," [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.sin.html#numpy.sin. [Accessed 02 04 2023].

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Janno Sepajõe

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Adding functionality of automatic gait cycles detection to gait analysis software”, supervised by Jeffrey Andrew Tuhtan and co-supervised by Cecilia Monoli
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

15.05.2023

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.