Master’s thesis in Electrical Engineering

Implementing real-time step detection algorithm in EyesWeb environment

June 2011

Authors:
Jonas STANDAERT
Wouter SPEYBROUCK

Supervisors:
Dr. Jörgen NORDBERG
Aylar SEYRAFI
The authors of this thesis give the permission to BTH to copy this work partially or fully on loan to any person, organization or company. The authors of this thesis cannot be held responsible in any way for any consequences for the use of the results of this work, either directly or indirectly.

Wouter Speybroeck & Jonas Standaert
Karlskrona, June 2011

This thesis was an examination. The errors discovered during the defense were not corrected. Use as a reference in publications is allowed after a favorable opinion of BTH.
Abstract

Many methods are used for measuring and defining the presence of Parkinson’s disease. One important aspect of the disease is gait disturbance, which can be discovered in the early stages of the disease. This disturbance can be measured using three-dimensional accelerometers, which are present in the Shimmer device. These devices are light-weighted wireless sensor nodes which can be easily mounted on the patient’s limbs. Data captured by the device can be streamed to a computer for real-time processing. The Matlab algorithms for step detection are converted to C++ code so they can be used in the EyesWeb environment. Whereas Matlab code can only serve post-processing, the Shimmer’s Bluetooth datastream can now be processed in real-time, since the EyesWeb environment supports real-time execution. Hence the Visual Studio environment is needed to design the algorithms into C++ based EyesWeb blocks. This real-time detection algorithm can be applied easily in small and inexpensive lab environments to calculate and obtain the Parkinson’s disease severity parameters. Hereby conclusions can be made to prevent the progress of the disease and to treat Parkinson’s disorder in early stages.

Keywords: Shimmer - Parkinson’s disease - Gait analysis - C++ - EyesWeb
Acknowledgment

First we would like to thank our supervisor Dr. Jörgen Nordberg and co-supervisor Aylar Seyrafi, without their applied knowledge and confidence this thesis would have been impossible. We want to thank them for their support and succession during our thesis work. We want to express our gratitude to them for the necessary research work.

Also we want to thank Blekinge Tekniska Högskola (BTH) for the opportunity they have given us to complete our studies at their campus. We would also like to thank Katholieke Hogeschool (KAHO) Sint-Lieven for the education and knowledge they taught us during the past four years, we also like to thank them for the unique chance they gave us to go on this unforgettable and life enriching Erasmus project.

Finally, we would like to thank especially our parents, without their financial and moral support, this experience would not have been possible.

Wouter Speybrouck & Jonas Standaert
Karlskrona, June 2011
Contents

1 Introduction

2 Parkinson's disease
  2.1 Primary parkinsonism
    2.1.1 Classification
    2.1.2 Characteristics
    2.1.3 Stages of Parkinson's disease
    2.1.4 Causes
    2.1.5 Diagnosis
  2.2 Gait Analysis
    2.2.1 Definition
    2.2.2 Gait cycle
    2.2.3 Why
    2.2.4 Methods of analysis
    2.2.5 Equipment and techniques
  2.3 Tremor
    2.3.1 Definition
    2.3.2 Causes
    2.3.3 Characteristics
    2.3.4 Categories
    2.3.5 Unified Parkinson's Disease Rating Scale (UPDRS)
    2.3.6 Measurements
    2.3.7 Existing measurement methods
  2.4 Resolution

3 Hardware
  3.1 Shimmer
  3.2 USB Reader
  3.3 Block diagram
    3.3.1 Microcontroller MSP430F1611
    3.3.2 Triple Axis Accelerometer MMA7260Q
    3.3.3 Radio communication
    3.3.4 MicroSD Flash Storage

4 Software
  4.1 BSL430 Application
  4.2 Bluetooth Pairing of the Shimmer
4.3 Displaying Shimmer data in the EyesWeb environment ................. 30
4.4 Creating blocks ........................................................................... 32
  4.4.1 Creating a new project .......................................................... 32
  4.4.2 Creating a new block ............................................................. 33
  4.4.3 Configuring the new block ...................................................... 34
4.5 Implementation of a new block in EyesWeb ................................. 36

5 Step detection algorithm .................................................................. 37
  5.1 Algorithm for 3D accelerometer signals ..................................... 37
  5.2 Developed algorithm for step detection ...................................... 39
    5.2.1 Characteristics in the acceleration of the ankle ..................... 40
  5.3 Implementation of the step detection algorithm in real-time ......... 41
    5.3.1 Real-time C++ implementation .......................................... 41

6 Results ............................................................................................... 45
  6.1 EyesWeb ...................................................................................... 45
  6.2 Post-processing versus real-time ............................................... 48
  6.3 Evaluation .................................................................................. 49

7 Conclusion and future work .............................................................. 50
  7.1 Conclusion ................................................................................. 50
  7.2 Future work ............................................................................... 51

A StepDetection.h ................................................................................. 56

B StepDetection.cpp ............................................................................. 58

C Step detection flowchart ................................................................. 65

D EyesWeb visual output ................................................................. 66

E Output file plot ................................................................................. 67

F R code used for plots ................................................................. 68
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Dopamine, a neural messenger</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>The gait cycle</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>Acceleration and rotation in the Carthesian coordinate system</td>
<td>16</td>
</tr>
<tr>
<td>2.4</td>
<td>The accelerometer principle</td>
<td>17</td>
</tr>
<tr>
<td>2.5</td>
<td>Vibrating wheel gyroscope</td>
<td>18</td>
</tr>
<tr>
<td>3.1</td>
<td>Shimmer Wireless Sensor Unit/Platform</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>USB Reader: Dock for programming and charging the Shimmer</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>Block diagram of the Shimmer baseboard interconnections and integrated devices</td>
<td>25</td>
</tr>
<tr>
<td>3.4</td>
<td>Microcontroller MSP430F1611</td>
<td>25</td>
</tr>
<tr>
<td>3.5</td>
<td>Triple Axis Accelerometer MMA7361</td>
<td>26</td>
</tr>
<tr>
<td>3.6</td>
<td>Bluetooth module in the Shimmer</td>
<td>27</td>
</tr>
<tr>
<td>3.7</td>
<td>Zigbee module in the Shimmer</td>
<td>27</td>
</tr>
<tr>
<td>3.8</td>
<td>MicroSD card socket on the Shimmer baseboard</td>
<td>28</td>
</tr>
<tr>
<td>4.1</td>
<td>Shimmer Windows Bootstrap Loader with pre-defined bootstraps</td>
<td>29</td>
</tr>
<tr>
<td>4.2</td>
<td>Application Shimmer Graphics</td>
<td>30</td>
</tr>
<tr>
<td>4.3</td>
<td>EyesWeb library</td>
<td>31</td>
</tr>
<tr>
<td>4.4</td>
<td>Implementation of the Shimmer in the EyesWeb Environment</td>
<td>31</td>
</tr>
<tr>
<td>4.5</td>
<td>New project dialog window in Visual Studio</td>
<td>33</td>
</tr>
<tr>
<td>4.6</td>
<td>New User Catalog for EyesWeb XMI</td>
<td>33</td>
</tr>
<tr>
<td>4.7</td>
<td>Add a new class to the created project</td>
<td>34</td>
</tr>
<tr>
<td>4.8</td>
<td>Add Class: EyesWeb Block template</td>
<td>34</td>
</tr>
<tr>
<td>4.9</td>
<td>Generic information about the block</td>
<td>35</td>
</tr>
<tr>
<td>4.10</td>
<td>Pin information of the block</td>
<td>35</td>
</tr>
<tr>
<td>4.11</td>
<td>Implementation of a created block in the EyesWeb environment</td>
<td>36</td>
</tr>
<tr>
<td>4.12</td>
<td>Representation of the output</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Flowchart of different steps of gait detection algorithm</td>
<td>38</td>
</tr>
<tr>
<td>5.2</td>
<td>Whole signals acquired from walking test</td>
<td>39</td>
</tr>
<tr>
<td>5.3</td>
<td>Gait detection result</td>
<td>40</td>
</tr>
<tr>
<td>5.4</td>
<td>X, y and z are the three axes of the sensor</td>
<td>40</td>
</tr>
<tr>
<td>5.5</td>
<td>Principle of a rotating window containing 100 values</td>
<td>42</td>
</tr>
<tr>
<td>5.6</td>
<td>Flowchart of the real-time step detection algorithm</td>
<td>44</td>
</tr>
<tr>
<td>6.1</td>
<td>Representation of the total EyesWeb program</td>
<td>46</td>
</tr>
<tr>
<td>6.2</td>
<td>Results plots</td>
<td>47</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Nowadays a lot of research is already done to detect whether a person has Parkinson’s disease or not. A wealth of methods are available to discover the different stages of the disease. A well known rating scale for Parkinson’s disease is the Unified Parkinson’s Disease Rating Scale (UPDRS), explained in section 2.3.5. This scale is able to give a precise view on stage of the disease after a clinical examination and a questionnaire. As the elderly are more susceptible to have Parkinson’s disease the tests of the UPDRS may become inconvenient or impossible, therefore a search for easy measurements was started. Wireless technology offers a whole new kind of measurement possibilities. Vibration, acceleration, tilt, etc. can now be measured on a free-moving body or body part. Without the wires used in wired measurements the patient’s movements are less restricted and can be recorded even outside the lab environment.

This thesis will present a way of detecting Parkinson’s disease using accelerometer signals from a patient’s feet. Using wireless technology gait data is recorded and can be used as an input for a gait detection algorithm. Chapter 2 will give the background information needed about Parkinson’s disease stages, characteristics, the gait cycle, wireless measurements and much more. Chapter 3 will be used to explain the device used. The Shimmer’s specifications will be given and the possibility to stream data real-time will be explained. How software should be installed and what versions need to be used will be given in chapter 4. This chapter will also include explanation for programming the Shimmer device and a small tutorial for working with the EyesWeb environment. How to make your own EyesWeb functionality in Visual Studio is given in the same chapter. An existing step detection will be analysed in chapter 5 after that the new real-time algorithm that was written in C++ will be described. Accelerometer signals that have been processed by the real-time algorithm are shown in chapter 6. Here characteristics of the gait are reviewed based on the acceleration signals. Finally we will resume what is accomplished in this thesis in chapter 7. The last chapter will also make some suggestions how the developed gait detection algorithm can be used in future works.
Chapter 2

Parkinson’s disease

2.1 Primary parkinsonism

2.1.1 Classification

Parkinson’s disease is the second most common neurodegenerative disorder after Alzheimer’s disease. Parkinson’s disease is a disorder that should not be confused with parkinsonism or Parkinson’s syndrome. The term parkinsonism\[1\] denotes a wide range of neurological diseases that are characterized by tremor, hypokinesia, rigidity, and postural instability. Parkinson’s disease is the most known disease out of four different types of parkinsonism. Parkinson’s disease is a disorder where the functionality of the central nervous system slowly decreases, the disease is characterized by movement-related symptoms, mostly the slowness and difficulty of movements and walking.

The four types of parkinsonism\[2\] in order of commonness are:

1. Primary parkinsonism (idiopathic parkinsonism)
2. Secondary parkinsonism (acquired, symptomatic parkinsonism)
3. Heredodegenerative parkinsonism
4. Multiple system degeneration (parkinsonism plus syndromes)

The most common type of parkinsonism is Parkinson’s disease, also called PD, paralysis agitans or idiopathic parkinsonism. The term idiopathic is used to indicate the unknown cause of the disease. The second type of parkinsonism is caused by the side-effect of certain medicines, a different nervous system disorder, or another illness. Heredodegenerative parkinsonism refers to the hereditary types of parkinsonism like Huntington’s disease, Wilson’s disease and others. Multiple system degeneration or the so-called parkinsonism plus syndromes present the greatest diagnostic challenge, characteristics will be presented in the next section.
2.1.2 Characteristics

The characteristics of Parkinson’s disease can be summarized by saying: tremor, rigidity and bradykinesia. Still there are other and more specific characteristics. Presented in table 2.1 are the characteristics used for distinguishing Parkinson’s disease from parkinsonism plus syndromes.

Differential diagnosis of Parkinson’s disease and the parkinsonism plus syndromes

Table of characteristics used to distinguish Parkinson’s disease from Parkinson plus syndromes. Symptoms with a + are present in the disease mentioned in the column. A ± is assigned to a symptom that could occur in the disease and a - for a characteristic that is mostly not present.

Table 2.1: Characteristics for Parkinson’s disease (PD) and parkinsonism plus syndromes

<table>
<thead>
<tr>
<th></th>
<th>PD</th>
<th>PSP</th>
<th>SDS</th>
<th>SND</th>
<th>OPCA</th>
<th>CBGD</th>
<th>ADP</th>
<th>DPCG/ALS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bradykinesia</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>±</td>
<td>+</td>
<td>±</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Rigidity</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>±</td>
<td>+</td>
</tr>
<tr>
<td>Gait disturbance</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>±</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Tremor</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>+</td>
</tr>
<tr>
<td>Ataxia</td>
<td>-</td>
<td>-</td>
<td>±</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>±</td>
<td>-</td>
</tr>
<tr>
<td>Dysautonomia</td>
<td>±</td>
<td>±</td>
<td>+</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>±</td>
</tr>
<tr>
<td>Dementia</td>
<td>±</td>
<td>±</td>
<td>+</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Dysarthria/dysphagia</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>+</td>
<td>+</td>
<td>±</td>
<td>±</td>
<td>+</td>
</tr>
<tr>
<td>Dystonia</td>
<td>±</td>
<td>±</td>
<td>+</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Eyelid apraxia</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>±</td>
<td>-</td>
<td>±</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Limb apraxia</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>±</td>
<td>-</td>
</tr>
<tr>
<td>Motor neuron disease</td>
<td>-</td>
<td>-</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>±</td>
<td>+</td>
</tr>
<tr>
<td>Myoclonus</td>
<td>±</td>
<td>-</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>±</td>
<td>-</td>
</tr>
<tr>
<td>Neuropathy</td>
<td>-</td>
<td>-</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Oculomotility disturbance</td>
<td>-</td>
<td>+</td>
<td>±</td>
<td>-</td>
<td>+</td>
<td>±</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Orthostatic hypotension</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Sleep abnormal</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Asymmetric find</td>
<td>+</td>
<td>+</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Levodopa response</td>
<td>+</td>
<td>±</td>
<td>±</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Symptoms that are typical for Parkinson’s disease are bradykinesia (slowness of movement), rigidity (stiffness of the muscles), gait disturbance (problems during walking), tremor (in hands and other body parts), assymmetric movements (like the swing of the arms during walking) and levodopa response (this will be explained later in section 2.1.5).

Some symptoms are possible to manifest on patients indicating Parkinson’s but do not occur on every patient. These are dysautonomia (autonomic dysfunction, malfunction of the nervous system),
dementia (cognitive dysfunction), dysarthria (speech difficulty), dysphagia (swallowing impairment), myoclonus (sudden twitching of the muscles), orthostatic hypotension (low blood pressure), abnormal sleep, micrographia (small cramped handwriting) and dystonia. Dystonia is the abnormal tonicity of muscle, characterized by prolonged, repetitive muscle contractions that may cause twisting or jerking movements of the body or a body part.

Characteristics that may but will probably not occur are ataxia (uncontrolled muscular movement), eyelid and limb apraxia (inability to perform normal movements), motor neuron disease (degeneration of the neurons in the spinal cord to coordinate muscular movement), neuropathy (weakness of the nerves) and oculomotility disturbance (eye and eyelid movement)\[4\].

### 2.1.3 Stages of Parkinson’s disease

Two scientists, Margaret M. Hoehn and Melvin D. Yahr, described in their work\[5\] five stages of Parkinson’s disease\[6, 7\]. These stages are already for a long time used to make a specific treatment for every stage. Nowadays stage 0, stage 1.5 and 2.5 have already been added, but these will not be discussed in this work.

#### Stage one
- Mild unilateral symptoms
- Presence of tremors in one of the limbs

#### Stage two
- Bilateral symptoms
- Problems walking or maintaining balance

#### Stage three
- Rather severe symptoms
- Inability to walk straight
- Slowing of physical movements

#### Stage four
- Severe symptoms
- Rigidity and bradykinesia are often visible
- Tremors or shakiness may lessen for unknown reasons

#### Stage five
- Inability to control physical movements
- Inability to stand or walk during this stage
- One-on-one nursing care needed
2.1.4 Causes

Parkinson’s disease is caused by the progressive impairment of neurons (nerve cells) in the substantia nigra, which is a brain structure in the midbrain. Normal neurons produce dopamine. Dopamine serves as a chemical messenger allowing communication through the nerves to control muscular movement. The lack of dopamine biosynthesis in these neurons cause abnormal nerve functioning, and thus loss in the ability to control body movements\[8, 9\]. How dopamine basically works is displayed in figure 2.1. Decreased dopamine production does not result directly in movement disorders but will only become outward when dopamine production is significantly lower than it should be.

![Figure 2.1: Dopamine, a neural messenger](image)

Scientists are still looking for the “why” of Parkinson’s disease. The cause of the low dopamine production is the cell death in the substantia nigra, but it is unknown why these cells lose functionality. Three main caused are being proposed\[10\]:

- Aging
- Genetic
- Environmental

Aging

According to the work of Loneneke M L de Lau conclusions can be made that the risk to develop Parkinson’s disease is 0,3% for the general population. The elderly (people of age 60+) have three times as much risk to develop the disorder\[11\]. The elderly are thus more vulnerable to PD but still more information is needed to know more about specific causes.

Genetic

Although Parkinson’s disease is believed not to be a hereditary disease scientists try to prove a genetic link among PD patients. It is said certain families share a gene that leads to Parkinson’s disease. The fact about these genes is that only 5% of the patients have them and having this gene only gives the
person a slightly higher risk for the disorder. Large epidemiological studies demonstrate that people with an affected first-degree relative, such as a parent or sibling, have a two-to-three fold increased risk of developing Parkinson’s disease, as compared to the general population.

The genes identified to date include: PARK1, DJ-1 (PARK7), Pink1 (Park6), dardarin (DRDN), Tau, Irlk2, parkin, uchl-1, park3, park9, park10, park11[10].

Environmental

Many environmental factors have been proposed for being the origin of Parkinson’s disease, but the studies of these factors only show a slightly higher risk for Parkinson’s disease[11]. Yet none of the studies could prove any environmental factor to be the cause of disease.

To date, epidemiological research has identified rural living, well water, herbicide use and exposure to pesticides, as factors that may be linked to Parkinson’s disease. Though, these environmental factors are not useful in diagnosing the cause of Parkinson’s disease in individual people. However, they have been helpful in studying laboratory models of Parkinson’s disease.

Additionally, a synthetic narcotic drug called MPTP (1-methyl 4-phenyl 1, 2, 3, 6-tetrahydropyridine) can cause immediate and permanent parkinsonism if injected. This drug can appear in a street herion called MPPP (also called Super Demerol). MPTP is a side-product of the MPPP synthesis and is converted in the body to MPP+ (1-methyl-4-phenypyridinium) that is extremely toxic to the dopaminergic neurons in the substantia nigra. Those neurons stimulate the basal ganglia which in turn are responsible for conscious control of movement. When these are not working as they should, the person can no longer move his body properly.

2.1.5 Diagnosis

How Parkinson’s disease is distinguished among other neurologic diseases will depend on which symptoms the patients suffers from. Signs of bradykinesia, rigidity and tremor will point out the disease is part of the parkinsonism disorder group. Other more specific signs should sort out whether the patient suffers from Parkinson’s disease or not.

Doctors will look for the resting tremor (tremor that is heaviest when the body part is in the rest position) and other muscular malfunction. The patient could have an uncommon face expression (so-called masklike facies), stiffness in the muscles, asymmetrical arm swing, short steps and others. Muscular blocking is a common feature that keeps the muscle from contracting or extending. This can occur in the arm, on that moment the patient will not be able to bend the elbow (the lead pipe rigidity). A muscular block could also occur when a person is walking, the patient will then “freeze” for a time up to one minute. PD patients will have difficulty raising from a chair, or maintaining their balance[10].

6
Levodopa response

For the diagnosis of Parkinson’s disease it is impossible not to mention levodopa. L-Dopa or levodopa is a psychoactive drug that is a precursor to neurotransmitters like dopamine. So if a person suffering from the neurodegenerative disease PD takes levodopa his symptoms will temporarily cease. L-Dopa has been used effectively for over 30 years as treatment for Parkinson’s disease. It is also helpful for distinguishing Parkinson’s disease from other conditions like the various types of parkinsonism.

Diagnostic confidence

Clinical diagnosis of Parkinson’s disease is based on the identification of several cardinal motor signs as it is mentioned in section 2.1.2. These characteristics are used to rate a patient with the Unified Parkinson’s Disease Rating Scale, which is explained in section 2.3.5.

There are three different levels of diagnostic confidence: Definite, Probable, and Possible. The first two are based on clinical criteria alone. Neurologic examination is needed for the diagnosis of Definite PD[12].

2.2 Gait Analysis

2.2.1 Definition

Gait Analysis[13] is the evaluation of the manner or style of walking usually done by observing the human as it walks in a straight line. It is the systematic study of human motion. This study can be done on 2 ways, the first observation is just by using the eyes and brains. Another way is to use instrumentation for measuring body movements, body mechanics and activity of the muscles.

2.2.2 Gait cycle

A gait period[14] can be seen as the period of time from one heel strike to the next heel strike of the same foot. After years of analysis, they came to the conclusion the gait can be divided in 2 phases: the stance and the swing phase. The total gait cycle[15] can be seen in figure 2.2.

The stance phase is about 60% of the time and is subdivided into the next 5 sequential steps:

- Initial contact: double limb stance and weight acceptance (0-10%)
- Loading response: early stance (10-25%)
- Mid stance (25-37%)
Figure 2.2: The gait cycle

- Terminal stance (37-50%)
- Pre-swing: double limb stance and weight release (50-60%)

The second part of the cycle is the swing phase and it contains next subdivisions: 3 swing stages:

- Early swing (60-75%)
- Middle swing (75-87%)
- Late swing (87-100%)

**Stance phase**

The stance phase begins when the heel of the forward makes contact with the ground and ends when the toe of the same limb leaves the ground. The first stance subphase is the initial contact. Where the heel strikes the ground, so both limbs are in contact with the ground. The ankle is at a neutral position. The stance knee begins to flex slightly and is close to full extension. Finally the hip flexion occurs. The next stage, called the loading response, happens under impact of the body weight and the maximum impact loading occurs. And the weight shifts to the supported leg. The foot arrives flat at the ground. During the next phase, mid stance, the double stance also ends and so the foot in the back lifts of the ground, the foot moves rapidly into pronation. In its phase the other foot is in his swing phase, so the body weight is totally on the supported foot. Now the balance is critical, because there is only one fixed point. Initially the foot remains pronated, but then re-supinates and so a turn around the ankle occurs and the body is pushed in front. In the next step, terminal stance, the heel lifts of the ground and when this movement is not under control, there is the feeling of falling forward. But now the other feet strikes the other side and the weight turns to the other leg again. So the balance changes again and the supported limb discharges. Here the final phase of the stance starts, the body
is balanced in double stance, the limb rapidly unloads again. The toe leaves the ground. This part is called the pre-swing, because at this point the body starts thinking about the swing phase of the supported leg.

Swing phase

The next phase will start, this is called the swing phase[14]. This phase is obviously shorter than the stance phase, it contains 40% of the total gait cycle. This phase starts when the toe of the supported limb leaves the ground and ends when the heel of the same limb makes contact with the ground again. This phase is divided in three parts:

- Initial swing
- Middle swing
- Terminal swing

In the beginning of the initial swing the toe has just left the floor and the limbs makes an acceleration, where the leg withdraws, and so the angle of the knee changes from 30° to 60°, also the position of your hip changes from a vertical position till an angle of 20° is reached. During the mid swing subphase the tibia reaches the vertical position and the angle of the knee and hip changes again. In the last phase of the gait cycle the foot is preparing for initial contact with the floor again. So the knee and hip stop their flexion. The gait cycle is ended now and starts again with the initial stance.

2.2.3 Why

The reason of the study and analysis of the gait will be discussed below. There are 3 reasons to study the gait, first is to make diagnosis and treatments for different diseases. It can also help to treat people and to make them walk again after an accident. It can also be used to optimise the gait, what is really interesting for athletes, in order to improve their performance. It can be used to distract the age, health and other characteristics.

Shock absorption

Important aspects of an efficient gait are shock absorption[14][16] and energy conservation. The most important reason is that 60% of the body weight is loaded in less than 20 milliseconds so a good shock absorption is really necessary. Altered joint motion or absent muscle forces can lead to abnormal gait and problems of the joints. The shock absorption will no longer be optimal. The lower joints collect the quick load of the weight. The effect on the joints is decreased due to the absorptive work by the pretibial muscles, this effect is accompanied by a delay in the forefoot contact. During the first contact the knee is also responsible for the absorption of shocks. A repositioning of the knee takes care of the stability of the knee, but it leads to more force on the contact. So the body makes a compromise between shock absorption and knee flexion, the eccentric quadriceps is the most important part during the phase of loading response. The knee catches the most of the shock, the hip stays unloaded due to the contraction of the abductor muscle.
Energy conservation

The gait is associated with metabolic costs. These costs are relatively low when people performing a free speed level walking, the speed someone chooses minimizes the metabolic work. Unlike running where there is a relative big metabolic cost. The metabolic cost during walking depends on the unit of time, the walking velocity and the travelled distance. During walking the energy costs to travel a prescribed distance increases, unlike the oxygen cost, that is saved by decreasing the walking velocity. But the energy cost cannot be changed substantially every unit of time, it is just a process that happens step by step. The walk velocity will decrease when getting older and the energy cost increases three times. [14]

2.2.4 Methods of analysis

The most common way to analyse the gait is to characterize most gait pathologies, so you can distract the gross abnormalities in walking. [14]

Motion

The first way to analyse the gait was just observation by using eyes and brains, now the research of the motion[14] is an expansion by using extra tools, like cameras and markers. In this expansion the three joints of each leg is observed and analysed individually. Commonly the normal walk motion is in a sagittal plane and there are also some subtle rotations, but not only only these movements can be observed, also other planes must be observed, because every rotation and translation has an influence on the gait. The disadvantage of this method is that a certain plane has to be in the vision of the camera. Another disadvantage is that the rotations of the joints cannot be clearly displayed when they are twisted compared to the plane of the camera. The solution for the last problem is to have a three dimensional motion analysis.

The position of the joints and their changes can be studied by fixing markers on the skin. A clear view of a joint can be taken by using at least three markers for every joint. The evolution and movements of a marker can be registered using different cameras, when using the right algorithm a lifelike representation can be made for each single step of the gait cycle.

The angular velocity and the acceleration of the limbs can be inferred. The second way to get the data is using an electrogoniometer[17], who can be used to measure angles. The advantage is the easy way of use and you can directly use the data, unlike video where the data has to be edited before it can be used.

External forces

During the stance period of the gait the feet performs a three dimensional pressures on the surface [14]. These pressures change during the different phases of the gait cycle. Electric pressure sensors in the surface can be used to check how the pressure divides themselves in the three dimensions. On
several moments: In the beginning, in the middle and at the end of the gait cycle, both feet make contact with the ground and the body weight is redistributed and the points of pressure will change abruptly. This analysis can be used to compute the pressure and the loads on the joints. Also tremor can be found and analysed.

**Dynamic electromyography**

It is not possible to measure muscle contractions directly, an electromyogram (EMG)\[14\] can be used to measure the activity of the muscles. This way of analysis cannot be seen as a standard part of gait analysis, but contains important information about timing and intensity of the signals. The force of muscles cannot be estimated based on relative intensity of the signal. When there are isometric contractions, a linear relation between EMG intensity and the power output can be found. This relation can be expressed in absolute voltage or percentage. Due to normalisation of the force output of the muscles there is the possibility to compare the relative intensity of the muscles. Another technique that can be used is the percentage comparison to the maximum force output of the muscles compared to the other forces. This is a useful technique when the contractions does not really achieve extreme values, otherwise the result should not indicate the normal changes very clearly and are limited visible, considering the peak values are seen as 100%.

**Mechanical and metabolic efficiency**

Mechanical work is the amount of energy transferred by a force acting through a distance, compared to a relative angular velocity. When people want to execute forces with their legs, a contraction of the muscles must occur. What matches with a change of energy and a metabolic cost. But mechanical work not the same as a metabolic cost, since some certain movements does not perform mechanical work, but there is a metabolic cost present, for example pushing a walk. Otherwise a mechanical movement entails a metabolic cost. The most energy expenditure are determined by indirect calorimetry and separating O$_2$. The inclusion of volume O$_2$ indicates the output performance, what depends from person to person, athletes will absorb more quickly oxygen in their blood. A comfortable walk gait normally matches with a low consumption of oxygen for healthy people, for other people this might ask a bigger effort and so a higher rate of the consumption of oxygen. When the walking speed increases, the body needs more oxygen and the heart will start beating faster to divide the oxygen over the whole body. \[14\]

### 2.2.5 Equipment and techniques

Different techniques and equipments to analyse the gait will be explained in this section. These techniques may include only cameras or can also include markers, which are placed on the skin of the patient. \[18\]
Chronophotography

Chronophotography[^19] is a set of photographs of a moving object, taken from a certain object or action, to look and analyse what happens step by step. One single camera can be used to shoot all the images, or multiple cameras can be used so the time step gets smaller and a more detailed description can be made. There are different ways of presentation, the pictures can be placed side by side, to have a good overview over the process or a part of the process, but there is also the opportunity to make some kind of movie of the sequential steps, to see the differences between two sequential images. The original purpose was to help scientist study motion objects of humans and animals. But now there are new techniques to use like a video, which yield a higher accuracy.

Cine film or video

By using a video camera[^18], a high accuracy of motion and also the analysis of every single step of the motion process can be made. Multiple cameras can be used to become different points of view this allows a three dimensional view of the gait process. Software programs can be used to measure the joint angles and velocities. In this way conclusions of the process that occurs can easily be drawn.

Passive marker systems

Another method to analyse the gait cycle is using passive markers[^18], which are placed on the skin around the joints. The advantage of this methods is the high accuracy using multiple cameras to measure the movement of each marker exactly. The data of the different markers may be sent to a computer and then create a three dimensional view of the markers. Using an algorithm the movements of the joints can be computed to analyse the gait.

Active marker systems

Active optical system[^18] does not only reflect an incoming signal, but also they send out their own signals using light-emitting diodes, each marker contains his own frequency. A camera captures the signals from the markers and can process them, the way of working is the same as passive markers. The frequency of a marker can be seen as an ID-tag and so it is easy to know from which marker a signal comes. The disadvantage of these markers is their energy consumption, so an energy source must be provided.

Inertial (cameraless) systems

Sensors are placed on different places of the limb of the observed people. These sensors capture the data from their accelerations and movements, every action is saved. These data can be transmitted to the computer and can be used to draw some conclusions. The advantage is that people can just wear it during the day and do not have to be in the lab, where they are most of the time concentrated on their

[^19]: Multiple cameras can be used so the time step gets smaller and a more detailed description can be made.
[^18]: Software programs can be used to measure the joint angles and velocities.
walk. During a free walk the movements are more free and a realistic pattern of the gait will be arise. It is also possible to capture signals from a longer period (a few hours) and the analysis can be more relevant. These sensors consume energy, which should be as low as possible and a energy source must be present. The data of the signals must be captured and saved on a memory card.  

2.3 Tremor

2.3.1 Definition

Tremor is defined as an unintentional, somewhat rhythmic, muscle movement involving to-and-fro movements (oscillations) of one or more parts of the body. Although tremor cannot be seen as a life-threatening disease, it can make daily tasks hard to perform. Parkinsonian tremor is characterized by the twitching of the thumb and index finger in a motion that is referred to as ‘pill rolling’.

2.3.2 Causes

Tremor is generally caused by problems in the cerebellum, this part of the brain takes care of the control of the muscles. These problems can be caused by strokes, traumatic brain injury and neurodegenerative diseases. Other causes which can increase tremor are the excessive use of drugs, alcohol and other poisons. Tremor can also be inherited.

2.3.3 Characteristics

The most significant characteristic is a rhythmic shaking in the hands, arms, head, legs or trunk. Hereby daily tasks can become difficult, like writing, drawing, holding and controlling utensils.

Tremor can be classified in four classes. The most common tremor related to PD is resting or static tremor, that occurs when the body is in a total rest position and every body part is supported against gravity. This is most of the time seen with patients with Parkinson’s disease. Another type of tremor is the action tremor, this occurs during or short after an effort of a person. Postural tremor is tremor under influence of gravity, the muscles wants to constitute a reaction against this force. Kinetic tremor can occur when someone is performing a specific task, especially when precision skills are necessary.

2.3.4 Categories

Tremor can be subdivided in different classes depending on the cause, clinical features or origin. The most important categories will be discussed below.
Essential tremor

The most common type of tremor is essential tremor. This tremor begins mostly at the ends of the body and can be accompanied by a mild disturbed gait. This tremor can be mild and non-progressive, the tremor frequency can decrease when getting older, but the severity may increase, another reason can be stress, fever, exhaustion and low blood sugar. So daily tasks can become more difficult to perform. Essential tremor can be family related, children of parents with essential tremor have 50% more chance to have the tremor themselves.

Parkinsonian tremor

The cause of parkinsonian tremor is the damage of structures within the brain, which are responsible for the control of movements. This is a subdivision of the resting tremor. The most visible effect of the tremor is the ‘pill-rolling’, this is a circular movement of the tips of the thumb and the index finger. The parkinsonian tremor starts generally around the age of 60, it begins at one side of the body and will later become bilateral.

Dystonic tremor

Dystonic tremor occurs with people affected by dystonia, this neurological movement disorder is caused by sustained muscle contractions due to twisting and repetitive movements or abnormal postures. The tremor occurs irregularly and can be prevent by total rest of the body. The tremor can be reduced by touching the affected body parts or muscles.

Cerebellar tremor

Cerebellar tremor is present at the end of a targeted action, the controlled effort causes the tremor. Damage of the cerebellum, is the result of stroke, tumor or disease as multiple sclerosis or inherited degenerative disorder. The tremor is present when a person is active or is maintaining a particular posture.

Psychogenic tremor

Psychogenic tremor can happen when a person is at rest or during postural or kinetic movement. The characteristics can be very varied but mostly covers onset and remission. The tremor can increase by stress and change in tremor direction, but can also be decreased by distracting the person. Psychogenic tremor is mostly combined by conversion disorder or other psychiatric diseases.
Orthostatic tremor

This tremor occurs immediately after standing up and is visible as a contraction of the muscles in the legs and trunk. The patient shakes uncontrollably and the body has a tensed position. No other signs are present and the shaking stops when the patient is in a more relaxed position. Orthostatic tremor may also occur on patients who have essential tremor.

Physiologic tremor

This tremor has no clinical importance and cannot be observed by the eyes. The most important factors of this disease are strong emotions, physical exhaustion, heavy metal poisoning, stimulants, alcohol withdrawal and fever. It is mostly not caused by a neurological disease but by reaction on above factors. The detection of the tremor can made visible by placing a piece of paper on the hands of the patient with arms outstretched. This tremor can mostly be prevented by avoiding the causes.

2.3.5 Unified Parkinson’s Disease Rating Scale (UPDRS)

The Unified Parkinson’s Disease Rating Scale is a rating scale used for measuring the severity of Parkinson’s disease. The rating scale includes ratings for behaviour, mood, tremor, cognitive abilities and much more. It consists of five parts.

Part I Evaluation of Mentation, behavior and mood.

Part II Self evaluation of the activities of daily life (ADLs).

Part III Clinician-scored motor evaluation.

Part IV Hoehn & Yahr stating of severity of Parkinson’s disease.

Part V Schwab and England ADL scale.

Each part is subdivided in disabilities related to Parkinson’s disease. These impairments are rated on a scale from zero to five. Zero is assigned to a patient that is not affected with the symptom, five assigns a severe presence of the disability.

Part one describes the state of the patient concerning intellect and mood. Memory loss is rated with the Intellectual Impairment score, dreaming and hallucinations with a Thought Disorder score. Also Depression and Motivation/Initiative scores are located in this part.

In Part two the activities of daily living (ADL) are being rated with a score from zero to five. These activities are speech (understandability), salivation, swallowing, handwriting, cutting food and handling utensils, dressing, hygiene, turning in bed and adjusting bed clothes, falling, freezing when walking, walking, tremor and sensory complaints.
Part three is the Clinical motor examination. The patient is being rated on muscular movement. Scores are given to speech, facial expression, tremor at rest, action or postural tremor of hands, rigidity, finger taps, hand movements, rapid alternating movements of hands, leg agility, arising from chair, posture, gait, postural stability, body bradykinesia and body hypokinesia.

All these ratings are evaluated by interview and clinical observation. Part four and five are different kinds of rating scales, since they use different scores. The Hoehn and Yahr scale is divided into five stages, the patient is assigned a stage linked to the Parkinson’s disease severity, stage one is the lightest form of Parkinson’s disease.

The Schwab & England activities of daily living scale assigns a percentage to the independence of the patient. A patient receives the score of 100% when he or she is able to perform activities of daily living (ADL) without any difficulty, slowness or impairment.

2.3.6 Measurements

Identifying tremor can be done in several ways. For example visual observation can done on a stretched arm, less distinct tremor can still be noticed by placing a sheet of paper on the patient’s hand when stretching the arm. But as always technology offers us the most precise ways for measurements. The accelerometer and gyroscope are two devices that can be used to measure tremor. With this technology acceleration and rotation can be measured[22, 23].

Accelerometer

Description

Accelerometers are devices to detect and measure proper acceleration. This equipment is thus able to measure any acceleration, gravity also taken into account. For mathematical purposes acceleration in space is divided in three-axis acceleration, this is shown in figure 2.3a. An accelerometer can measure acceleration in 1 direction, but devices containing three of these accelerometers (for acceleration
Electromechanical accelerometers are based on Newton’s first law of motion, the law of acceleration. A force is thus needed to induce an acceleration (a) to a certain mass (m). This is stated in equation 2.1.

\[ F = m \frac{dv}{dt} = ma \quad (2.1) \]

The second law of physics on which accelerometers are based is Hooke’s law of elasticity, given in equation 2.2, this law states that the extension of a spring is in direct proportion with the load applied to it as long as the extension does not exceed the material’s elastic limit. In this equation \( k \) is the spring rate and \( x \) is the contraction or extension of the spring, compared to its resting position.

\[ F = -kx \quad (2.2) \]

\[ C = \varepsilon_r \varepsilon_0 \frac{S}{d} \quad (2.3) \]

Accelerometer devices hold a movable mass hung up on springs. On acceleration the mass will try to maintain his original speed so the movable part can induce an extension or contraction of the springs. This variable contraction/extension can be converted into a variable capacitance since capacitance is inverse related to the distance (d) between the plates of a capacitor, as shown in equation 2.3, also the capacitance is directly proportional with the area of overlap of the two plates (S), the relative static permittivity (\( \varepsilon_r \)), which is related to the used material and the electric constant (\( \varepsilon_0 \)). The principle of the accelerometer is shown in figure 2.4, this is a simplified version of the real equipment. The variable capacitance can be converted to a voltage or directly to a digital value.

Gyroscope

Description

Gyroscopes are devices to detect and measure rotation. These devices are able to measure change in orientation, whether it is roll, pitch or yaw. A mechanical gyroscope consist of a spinning wheel
with a certain mass. The axis of this wheel is free to turn around the two other axes, but due to the principles of conservation of angular momentum the wheel will try to maintain his original orientation, the gyroscope will thus have to turn around the two joints when orientation is changed. Gyroscopes used for angular measurements are most of the time based on this principle.

**Principle**

A conventional spinning wheel gyroscope is based on the principle of conservation of angular momentum and the variable value of a capacitor discussed in the section about the accelerometer. This kind of gyroscope is able to measure rotations around two axes, other than the axis of the wheel. Though Microelectromechanical systems (MEMS) gyroscopes do not have a spinning wheel. Most MEMS gyroscopes fall into these four categories: tuning-fork gyroscopes, oscillating wheels, Foucault pendulums and wine glass resonators. Those gyroscopes take advantage of the Coriolis effect, this is the effect on moving objects when viewed from a rotating reference frame. With the Coriolis effects moving objects seem to be deflected from their path.

A **Tuning-fork gyroscope** contains two masses that are oscillating in opposite directions. When the device rotates the Coriolis force will cause this vibration to become outward. These oscillations can be measured by a variety of mechanisms. For example they can be measured by a variable capacitance as mentioned in the section about accelerometers.

**Oscillating wheel gyroscopes** contain a wheel that is driven to vibrate about its axis of symmetry. Conservation of angular momentum will cause to tilt the wheel when changing the orientation, this can be detected with capacitive electrodes under the wheel. This can be seen in figure 2.5. With two extra capacitive nodes rotations around two axes can be measured.

The **Wine glass resonator** is also known as hemispherical resonant gyroscope (HRG). Wine glass resonators contain a resonant ring or hemisphere that vibrates. Again the Coriolis effect makes it possible to measure the change in orientation.

**Foucault pendulum** gyroscopes are based on a vibrating rod that is typically oriented perpendicular to the chip plane. They are similar to the conventional Foucault pendulum that takes advantage of the Coriolis force (swinging pendulum in the rotating earth reference frame).
2.3.7 Existing measurement methods

Wired measurements

Wired measurements for quantifying Parkinson’s disease severity exist but encounter some limitations. This kind of measurements are able to record action tremor and resting tremor in a lab environment. The limitations of the environment is that a patient can only be observed for a short time, thereby it is harder to observe the patient’s activities of daily living.

Nano17 6-axis force/torque sensors

Observations have already been done with a new kind of protocol called Advanced Sensing for Assessment of Parkinson’s disease (ASAP)\textsuperscript{24}. This protocol measures the grip force of a patient trying to follow a sinusoidal force target wave. A different target wave could be used but measurements remain the same: force and torque are measured each in 3 directions. Test were performed under three different cognitive loads: the first test of one minute the patient just tries to follow the target wave, the second minute the patient needs to count down from 100 to 1 in the same time and the third minute the patient counts down from 100 by 3 while following the sine wave.

The measurements were summarized in 3 variables. The tremor integral (area under the spectral density curve between 2 and 8 Hz) was calculated first, for the other two variables the data was filtered with a low-pass second-order dual-pass Butterworth filter with a cutoff frequency of 2 Hz. The remaining variables were the root-mean-square error (RMSE) between target wave and subject’s force response and the lag between the target waveform and the force response. In total 36 predictor variables were obtained for each individual (2 hands × 2 waveforms × 3 cognitive load conditions × 3 summary variables). These variables were used in different kinds of regression methods to approximate the UPDRS scale. The mean absolute prediction error for each regression method is given below:

- Principal component analysis with the Kaiser criterion: 7.06 ±1.37 UPDRS
- Lasso regression: 4.57 ±0.84 UPDRS
- Ridge regression: 3.58 ±0.69 UPDRS

Digitising Tablet

An other measurement method is represented by a digitising tablet\textsuperscript{25}. The aim of this research was to analyse the micrographia of PD patients. With the equipment used accurate two-dimensional information about the tremor in the upper limbs could be recorded at a relative low cost. Most tablets provide at least 200 lines per inch accuracy at a sample rate of 50 Hz. This sample rate is yet high enough to meet the Nyquist theorem (states that the sample rate should be at least twice the maximum spectral frequency) for a tremor that not exceeds 12 Hz. The digitising tablet is also able to determine the location of the pen when the pen is above the surface within the distance of 2.5 cm. This is an interesting feature as parkinsonian tremor could cause the pen to be lifted from the surface.

Data was processed with a Fast Fourier Transform (FFT) so as to determine the spectrum of the sampled data. A Hann window function was implemented to avoid start-up transients.
Wireless measurements

Wireless measurements provide some more flexibility as the patient’s movements are less restricted by cables or by the dimensions of the device. Still observation of a patient tend to be done in a lab environment and therefore only quite short observations are possible. Action tremor, resting tremor, gait etc. can than be measured when the patient is concentrating on the limbs being used. Some devices can be worn during the day and provide a realistic view on the Parkinson’s disease severity.

G-Link Wireless Accelerometer Node

The G-Link Wireless Accelerometer Node developed by Microstrain consists of MEMS accelerometers able to measure in three dimensions. With a memory of 2 MB up to 1 million samples can be recorded with a sample rate between 32 and 2048 Hz. After recording data can be sent wireless to a computer for post-processing.

With the device mounted on the dorsum of the hand tremors in the upper limbs can be recorded. The mass of the device (47 g) is enough to notice the device on the hand but does not restrict the patient’s movements. The trapezoid method for integration was used for calculating the mean acceleration of the device normalized to gravity.

Shimmer

The Shimmer (Sensing Health with Intelligence, Modularity, Mobility, and Experimental Reusability) used in 2007 also contained a triaxial MEMS accelerometer. The Shimmer contains a couple of interesting features. Bluetooth connection can be made with the device so data can be streamed in real-time or sent for post-processing, a MicroSD slot makes it possible to record data for a much more than a single day and the mass of approximately 22 g makes this piece of technology ultra-wearable.

The accelerometer data was high-pass filtered with a cutoff frequency of 1 Hz to remove gross orientation changes, and low-pass filtered with a cutoff frequency of 15 Hz to remove high frequency noise. Features like intensity, modulation, rate, periodicity, and coordination of movement were extracted for dyskinesia and bradykinesia. Clustering has been done using the Davies-Bouldin clustering evaluation index and the optimal window length was determined 5 seconds for dyskinesia and 6 seconds for bradykinesia.

Kinesia Technology

Kinesia Technology came up with a device consisting of two pieces. The wrist-worn module (85 g) recording sensor data at 128 Hz is able to send this data via Bluetooth and the finger-worn sensor (12 g) is able to produce tri-axial accelerometer and gyroscope signals.

The patient was observed for resting, postural and kinetic tremor performing different tasks. The gyroscopes and accelerometers collected a total of six signals. Each signal was first processed into quantitative variables describing tremor severity and those variables where given as inputs to another Matlab algorithm that was used to output a final score correlated to the UPDRS scores. The raw data was band-pass filtered using a second-order Butterworth filter with cutoff frequencies 3 and 10 Hz. Parameters for each kinematic channel included peak power intensity, peak power frequency, root mean square (RMS) of angular velocity, and RMS of angle.
Every patient was questioned after the lab tests for the comfort and ease of the device and 55% would wear the device in public.

**iPhone wireless accelerometer application**

The iPhone 3G with the mass of 133 g has a lot of technological features to offer. With the robust and scalable software package applications can be developed to access the built-in three dimensional accelerometer subsystem[29]. The iPhone can be mounted on the dorsum of the hand to capture the data. Realization of this application made spectral analysis very easy. Spectral analysis has been done using the Blackman window, with one Parkinson’s disease patient predominant frequencies were discovered in the range of 5.3 Hz, 7.7 Hz and 10.4 Hz.

**Autonomous Sensing Unit Recorder**

The only sensors the ASUR or Autonomous Sensing Unit Recorder has are gyroscopes. The device weighs 50 g and with a memory of 64 MB it can record up to 14 hours with a sample rate of 200 Hz. During tests a two-dimensional configuration of the gyroscopes was used in order to minimize energy consumption (30 mW for each gyroscope) and patient movements were recorded for periods of 3 to 5 hours while moving freely[30].

For detecting tremor the gyroscope signals were filtered with a first degree infinite impulse response (IIR) filter with a cutoff frequency of 0.25 Hz before they were divided into windows of three seconds. The spectrum of these windows was analysed and if the dominant frequency was between 3.5 and 7.5 Hz and the spectral density value higher than a certain threshold value the window was reported as tremor.

Bradykinesia was detected using different parameters: the average velocity of the hand, the percentage of time the hand was moving and the average rotation of the hand. Several window sizes from 5 up to 45 minutes were studied.

The UPDRS subscores of resting tremor and action tremor were used for evaluating the tremor quantification algorithm. The summation of the UPDRS subscores of finger tapping, hand movement and rapid alternate movements of hand were used for the bradykinesia quantification algorithm.

### 2.4 Resolution

After gathering the background information about the different characteristics from Parkinson’s disease, an easy to measure and clearly visible characteristic must be chosen to detect PD. The gait cycle satisfies this requirements. This cycle is divided in two phases, the stance and the swing phase. Each of these phases is subdivided in several steps. Having this knowledge an accurate and reliable method can be used to collect the data from the gait cycle. This can be done by measuring the acceleration of the legs. Since the acceleration is most pronounced in the lower part of the legs measurements are done as close as possible to the feet. The Shimmer is a small and light-weight device which can be easily mounted on the ankle of the foot. The Shimmer has the possibility to measure acceleration and store the data locally on a microSD card, hereby it is possible to do measurements outside the lab environment and is it easier to record accelerometer signals during daily activities. This data can be
used for post processing. The device can also send this data immediately to a computer using Bluetooth where the data can be processed in real-time. The use of a real-time Bluetooth data stream is the purpose of this thesis.
Chapter 3

Hardware

3.1 Shimmer

The Shimmer\cite{31}, contains an three-dimensional accelerometer. The Shimmer is a low-power device that provides a lot of functionality needed for wireless sensing. The design is made so it is comfortable wearable for sensing applications. The device contains an on-board microcontroller, the necessary information can be sent wireless to the computer using Bluetooth. It has also the possibility to store the data locally on a microSD card. But before this a program must be loaded on the Shimmer so the desired information can be collected. A picture of the Shimmer is depicted in figure\ \textit{3.1}

\textit{Features and benefits}

- Small size (53×32×25mm)
- Light weight (baseboard and battery 15g; with enclosure 22g)
- Stylish, functional enclosure with wearable straps
- \textit{Highly configurable}: Can be programmed to meet specific application, with configurable sensitivity, sampling rate, transmission rate and frequency and communication protocols.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{shimmer.png}
\caption{Shimmer Wireless Sensor Unit/Platform}
\end{figure}
3.2 USB Reader

The USB Reader is a docking station that is used for programming and charging the Shimmer. A USB cable is used to connect the Shimmer to any PC or laptop, now the dock can be used as a single unit charger. When the dock is plugged in to a computer the drivers will automatically be loaded so connection between the computer and Shimmer is made possible right away. When the Shimmer contains a microSD card, the flash storage can be mounted as it were a USB flash key.

The docking station has two light-emitting diodes, a pinhole reset and user button. One of indicators will blink amber when the Shimmer is being programmed and it will light up yellow when serial data is being transferred. The other indicator on the USB reader is used to indicate the microSD host status. The docking station is displayed in figure 3.2.

3.3 Block diagram

The core element of the Shimmer is the microcontroller MSP430F1611 which is responsible for the operation of the total device. The CPU configures and controls various integrated peripherals through I/O pins. Also an 8 channel 12 bit analog-to-digital converter is integrated, so signals gathered by the accelerometer can be captured. There is an external expansion present which allows communication to and from the baseboard using the docking station. A microSD Flash socket is built in on the board so additional storage can be provided. The Shimmer also has three light-emitting diodes for display purposes, like indicating his status. The platform is also equipped with a Bluetooth and 802.15.4 radio module so wireless data streaming can be provided. Figure 3.3 illustrates a block diagram of the Shimmer baseboard interconnections and integrated devices. In the next subsection the most important parts of the Shimmer will be discussed.

3.3.1 Microcontroller MSP430F1611

The core element of the Shimmer is the Texas Instruments MSP430F1611 microcontroller[32]. This microcontroller incorporates a 16-bit RISC CPU, peripherals and a flexible clock system. The interconnection uses a von-Neumann common memory address bus and a memory data bus. The modern CPU has the opportunity to memory-map analog and digital peripherals. The most important
characteristic of the MSP430F1611 is the possibility for mixed-signal applications, like the three-dimensional accelerations. The microcontroller contains a watch crystal so timestamps can be given to the signals. The clock system is also very flexible what allows the microcontroller to use an ultralow power consumption in the stand-by mode. Also high performance signal processing is made possible, what requires more power. The CPU integrated in the microcontroller incorporates features specially designed for modern programming techniques as calculated branching, table processing and the use of high-level languages as C. The CPU uses sixteen 16-bit registers, four of them are used for dedicated functions and the others have a general use. Flash memory is available to load a desired program on the microcontroller, this is bit-, byte-, and word-addressable and programmable. The flash memory module has an integrated controller to control the programming and erase operations. So the MSP430F1611 allows to program desired functions to gather information. The low power modes make the microcontroller optimised to use it in portable measurement devices, like the Shimmer. The microcontroller is depicted in figure 3.4.

### 3.3.2 Triple Axis Accelerometer MMA7260Q

Apart from the core of the device it has a triple axis accelerometer, depicted in figure 3.5. The operation of the accelerometer is already explained in section 2.3.6 and will not be discussed here anymore. Here some more specifications will be discussed. The sensor requires a very low amount of power an has a g-select input which switches the accelerometer between ±1.5 g and ±6 g measurement ranges. Other features include a sleep mode, signal conditioning, a one-pole low pass filter, temperature compensation, self test and zero-g detection, which detects linear freefall. The zero-g offset and sensitivity are set by factory. The sensor works between a power range of 2.2 V and 3.6 V and
consumes 500 µA of current. Every axis has his own analog output, which is connected to an analog-to-digital converter input of the microcontroller. Where the signal is converted to a digital signal and can be used for transmission.

### 3.3.3 Radio communication

The ability of the Shimmer to communicate as a wireless platform is one of the key functions. Two wireless solutions are integrated in the Shimmer, Bluetooth and 802.15.4. Both radio communications cannot operate simultaneously. Which connection is used depends on the requirements. The most important characteristics are shown in [Table 3.1](#).

<table>
<thead>
<tr>
<th>Metric</th>
<th>802.15.4</th>
<th>Bluetooth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Consumption</td>
<td>Better</td>
<td>Worse</td>
</tr>
<tr>
<td>Agility/Connection Speed</td>
<td>Better</td>
<td>Worse</td>
</tr>
<tr>
<td>Prebuilt application</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>Good</td>
<td>Poor</td>
</tr>
<tr>
<td>Range</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>Mesh Implementations</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Ability to customize</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>FCC Modular Certification</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Rate</td>
<td>Worse</td>
<td>Better</td>
</tr>
</tbody>
</table>

**Bluetooth**

Bluetooth is a low-cost, low-power, robust, short-range wireless communication protocol. A low-cost transceiver microchip is available in the device, so a short range (1 to 100 metres, depending on the power class) can be provided. Up to eight Bluetooth devices are able to communicate together using different channels, this occurs in a network called piconet, what is a point to multipoint network.
from one master and seven slave devices. Bluetooth uses 1 MHz channels to transmit the data. The Shimmer platform uses the Roving Networks RN-42. This is a perfect short range Bluetooth module that uses only 26µA in his sleep mode while it is still discoverable and connectable, so a low power consumption can be provided. The module contains a full version 2 Bluetooth Protocol Stack. The RN-42 has a range of more than 10 metres and the transmitted power can be adjusted depending on the application distance. In figure 3.6 the placing of the Bluetooth module can be found as EU20.

802.15.4 Radio

802.15.4 is maintained by the IEEE 802.15 working group and is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks. The focus of the standard is to become a low-cost, low-speed ubiquitous communication between devices. Also a low power consumption is on of the specifications. The range of use is limited to 10-meter communication and a transfer rate to 250 kbit/s is available, but also a lower transfer rate can be provided considering the result of a lower power consumption. Guaranteed time slots are reserved, which is an important feature for real-time applications. In figure 3.7 the placing of the 802.15.4 Radio module can be found as EU21.
3.3.4 MicroSD Flash Storage

A microSD card socket is present on the Shimmer baseboard, this is to incorporate extra memory resources, with a capacity limited to 2 GB. This is provided to store more data when the Shimmer is not streaming and also to make sure no data will get lost. This is necessary when the Shimmer is worn during daily tasks or when there could be network disturbance. Figure 3.8 shows the location of the socket on the baseboard, so where a microSD card can be placed.
Chapter 4

Software

In this chapter will be discussed which software is necessary to load a program on the Shimmer, to visualize the captured data on two ways: Shimmer display and EyesWeb Environment. Another section in this chapter will explain how to develop a new eyesWeb functionality in Microsoft Visual Studio 2005, and how to implement those blocks in the EyesWeb environment.

4.1 BSL430 Application

A Boot Strap Loader (BSL) is necessary to load a program on the Shimmer to capture the specific data from the different components of the Shimmer, the recommended BSL for the Shimmer is BSL430.exe which is available on the website of shimmer-research [33]. First the Shimmer is connected to the PC using the Shimmer docking station. The BSL application will automatically detect which COM port the Shimmer is connected to. A wide range of pre-defined bootstraps is available, figure 4.1, these bootstraps can be easily loaded on the Shimmer by clicking the ‘program’ button. After a few seconds the Shimmer is programmed and can be unplugged from the Dock. Now the bootstrap program can be executed by the Shimmer and this could involve reading the accelerometer sensors. The bootstrap used to capture the desired data from the sensors is called AccelGyro shimmer2r, this bootstrap can be found in the list of pre-defined bootstraps.

Figure 4.1: Shimmer Windows Bootstrap Loader with pre-defined bootstraps
4.2 Bluetooth Pairing of the Shimmer

After the installation of the desired bootstrap on the Shimmer, the captured data of the device can be send to the computer using Bluetooth pairing[44]. In this part will be discussed how the Shimmer device can be connected to the computer and how to visualise the captured data.

Adding a new device to the computer using Bluetooth consists of following steps. If you already added the Shimmer once before, it can be found in the list of ‘Devices and Printers’ in the start menu, so this section can be skipped. If not the Shimmer can be appended. The Shimmer can be found in the list of available Bluetooth devices, the Shimmer is named “RN42-xxxx”, where xxxx is a unique number. Now select the Shimmer and pair it using the ‘pairing via entering pairing code’, which has a default code of 1234. After completion of this process, the Shimmer is now successfully connected to the computer and the data can be sent to the PC.

Now the received data can be visualised to make sure the device is working properly. The used application ‘Shimmer Graphics’ can be downloaded from the website of Shimmer-research[33]. The graphical interface of the application can be seen in figure 4.2. To connect the Shimmer the corresponding COM-port must be selected. This port can be found in the properties of the Shimmer (‘Devices and Printers’). This port must also be selected in ‘Shimmer Graphics’ > Tools > Bluetooth link and select the desired COM-port. Now connection can be made to the Shimmer and displaying the data can start. Also some easy operations (like filtering) can be done on the data, but for more fine-grained operations the EyesWeb environment must be used, this will be discussed in the next section.

![Figure 4.2: Application Shimmer Graphics](image)

4.3 Displaying Shimmer data in the EyesWeb environment

In the previous section an easy way to display the data of the Shimmer is discussed, but the problem is that the necessary adjustments and calculations on the data cannot be done. A good and graphic alternative is using the ‘EyesWeb Visual Environment’, which can be downloaded on [35], now ‘BioMOBIUS_setup_2.0.exe’[36] must also be downloaded, this program creates a workspace
where the Shimmer is implemented and is automatically added to the EyesWeb Environment. After the installation of both programs EyesWeb can be launched. The application will open with a clean sheet. The necessary blocks can be found in the library, on the left hand side of the application (figure 4.3). First the Shimmer block has to be added, which can be found in the library > BioMOBIUS > Hardware > Shimmer. Now the block can be dragged to the workspace, in the properties of the block the serial port must be set and also other parameters can be changed. Now 4 buttons (library > BangGenerator) can be added to do following actions:

- **Connect**: Make connection with the Shimmer
- **Start**: Start the data stream from the Shimmer
- **Stop**: Stop the data stream
- **Disconnect**: Disconnect the Shimmer

The result of this process can be seen in figure 4.4. The properties are also visible, here the serial port and other properties can be set.

![Figure 4.3: EyesWeb library](image)

![Figure 4.4: Implementation of the Shimmer in the EyesWeb Environment](image)
After the implementation of the Shimmer, the data must be acquired\[34\]. The received data must be made ready for adaptation, in the library in the section > Math > Matrix, the block ‘MatrixGetItem’ is found. This block extract an item from the matrix with the data capture from the accelerometer. In total there are four blocks needed to capture all data (x, y and z from accelerometer and a timestamp). All those blocks have the same parameters. On this data an algorithm should be applied, but this requires the development of new functions, this must be done by creating new blocks using Visual Studio, this procedure be discussed in the next section.

### 4.4 Creating blocks

To implement the existing post processing algorithm in real time it is necessary to create the various functions in own EyesWeb blocks \[37\]. Each block implements some specific piece of functionality and algorithm. To create those blocks on an easy way the next software packages must be used:

- Microsoft Visual Studio 2005 \[38\]
  - EyesWeb XMI SDK 5.0.2.3 \[36\]

#### 4.4.1 Creating a new project

Microsoft Visual Studio is used to create the functionalities in C++, after the development of the blocks Visual Studio is used to compile the functionalities so they can be used in EyesWeb Visual Environment. The appropriate version of EyesWeb SDK is also required for block development, this SDK can be downloaded from the developers section on the BioMOBIUS website \[36\].

Visual Studio 2005 must be installed first before installing the Eyesweb SDK 5.0.2.3 in order for the correct insertion of the block wizard and the catalog wizards in Visual Studio, otherwise it would not be possible to create own blocks. Important to mention is that no other versions of Visual Studio can be used since Visual Studio 2005 is the only version that collaborates with the EyesWeb SDK 5.0.2.3. After the software is successfully installed the first block can be developed. A new project can be created in Visual Studio as shown in figure 4.5.
The used project type is Visual C++, where a template ‘EyesWeb User Catalog’ should be available if the wizard is correctly installed.

The next step is to create the catalog, in the first step some fields need to be filled in like: name, description, some more information about the author and the company (figure 4.6). This information gives the author of the block the opportunity to provide some information and license status.

Development and configuration of the new catalog DLL are done automatically by Visual Studio. Now the EyesWeb project is created and a new block can be added.

4.4.2 Creating a new block

After the creation of the catalog a new block can be added to this catalog. After opening the project ‘MyFirstProject’, right click on the project and add a new class, these steps are shown in figure 4.7.
In the next step the ‘EyesWeb Block’ template must be chosen to create a new block and a name for the block must be given, this step is shown in figure 4.8.

After executing previous steps the block is created, the next section will run through the different steps to configure the block.

### 4.4.3 Configuring the new block

After creating a new block, the blocks will be provided from some more information like the name of block, a description, the input, parameter and output pins. This step will be shown automatically after the creation of the block.

The first step of the configuration is the ‘Block Definition’.

- Block name
- Block description
• Libraries: list of libraries which the blocks belongs to
• Type of activation: selection between a periodic or a reactive activation

In figure 4.9 can be seen that a block can be made periodic or reactive, this will make a difference for the execution of the block its execution. A periodic block will execute on fixed time intervals whereas a reactive block will read its inputs whenever one of them is updated.

![Figure 4.9: Generic information about the block](image)

In the next steps different in- and output pins can be defined, as well as parameter pins. Figure 4.10 shows how multiple input pins can be added. These pins have a label, a type and an optional description. Different types of input can be chosen, like int, double, boolean, buffer and others. Parameter and output pins can be configured in a similar way.

![Figure 4.10: Pin information of the block](image)

Now the block is totally configured and pressing the ‘Finish’ button will cause Visual Studio to generate standard code for the block. After this process a block ‘shell’ is available with all its input, parameter and output pins defined with their types. The block its functionalities and algorithm can be implemented by override the methods Init(), Execute() and Done() in the .cpp file of the created block. A bitmap image is linked to each block, it can also be edited in Visual Studio and will be shown when the block is used in the Eyesweb environment. The last step is to build the block in Visual Studio. The ‘Build Solution’ button can be found under ‘Menu’ or can be triggered by pressing F7.
4.5 Implementation of a new block in EyesWeb

The block created in the previous section is ready to be used in the EyesWeb environment. First the EyesWeb environment has to be restarted, considering the new block is loaded at startup of the program. Now the block can be found in the chosen library. This block can be added to the patch created in section 4.3. Figure 4.11 shows three instances of the new block in the EyesWeb environment.

The data has undergone the desired changes and is ready to be displayed. There are different possibilities of representation. First the data can be displayed as values using ScalarDisplay from the library Math. Another method is to display the data as a graph using ScalarGraph, which can be found in the same part of the library. Both methods are shown in figure 4.12.

Finally after all the desired blocks are implemented, the program is ready to do his task, by capturing data and doing the desired calculations on the gathered data. The program can be started using the Start-button in the system toolbar or the shortcut F5.
Chapter 5

Step detection algorithm

This chapter presents the method for step detection based on accelerometer signals. The signals are received from two Shimmer device that is mounted on the ankles of a patient.

5.1 Algorithm for 3D accelerometer signals

When three-dimensional accelerometer signals can be recorded from a person’s ankles these signals can be given as input to an algorithm for step detection. How this is possible and what characteristics of the gait are necessary to consider will be explained in this section. The algorithm is developed by Jonghee Han (et al.) and is able to determine if a patient is walking and whether the limb is in swing or stance phase.

The first part of the algorithm is to distinguish if the patient is walking or not, in this first step the gait signal is divided into blocks of one second. Once detected if the patient is moving, the blocks of one second can be divided into ten smaller blocks of 0.1 second each. These blocks are used to distinguish if the limb is in his swing phase, otherwise the limb is said to be in stance phase. After the detection of the swing phase, a simple peak detection will be done on the detected swing phase. After the detection of all the peaks, the non-gait peaks will be removed. The different steps of this procedure in the x-direction can be seen in the flowchart in figure 5.1. The same procedure can be followed for the other two directions. In the last step of the flowchart the gait peaks must be detected, now the x- and y-acceleration must be compared to distinguish peaks occurring on the same time. In this flowchart $X_{th}$ represents the threshold value, used to compare with the standard deviation values of the different blocks, the standard deviation of the blocks is represented as $X_{std}$.

The consecutive steps of the algorithm are represented below:

1. Stop/moving discrimination: The total gait signal is divided into blocks with a length of one second. The standard deviation of the block is calculated and compared to a threshold value which is 25% of the standard deviation of the total recorded signal. If the standard deviation of the one second block is lower than the threshold the block is determined as ‘stop’. Otherwise the block will be classified as ‘moving’.
Threshold calculation for X axis (Xth)

\( X_{stdv} > X_{th} \)
: blocks 1 sec

yes
- Moving

\( X_{stdv} > X_{th} \)
: blocks 0.1 sec

yes
- Swing phase

no
- Stance phase

Positive peak detection

Removal of non-gait peaks

Figure 5.1: Flowchart of different steps of gait detection algorithm
2. **Stance/swing discrimination:** This step only applies to blocks that were classified as ‘moving’ by the previous step of the algorithm. The gait signals of the moving phase are now divided into 10 smaller blocks (each block now contains data of 0.1 second). Again the standard deviation of each block is calculated and compared to the same threshold as mentioned in the previous step. Values above the threshold indicate a block in the swing phase, values under the threshold will cause the block to be classified as stance phase.

3. **Positive peak detection:** During the swing phase a simple peak detection method will be applied. Detected peaks will be selected if they occur simultaneously in the vertical and horizontal acceleration signal.

4. **Removal of non-gait peak:** Finally the amplitude of the selected peaks has to be compared to a threshold $T_{st}$, peaks with a higher value are accepted. The peaks with lower value are reconsidered by comparing the shape of the x and y-accelerations near a peak, if they are similar, those peaks are accepted, all other peaks will be removed. For the peaks that remain, if the interval between two consecutive peaks is less than 0.5 seconds, one of the two peaks is removed. The peaks which remain now are determined as gait peaks.

The visual representation of a similar test [39] is shown in figure 5.2, here L/R is used to distinguish the left (L) and the right (R) leg. Accelerations, foot pressure and video image could be compared synchronously. Since data is captured simultaneously. Figure 5.3 also displays a similar test, here an automatic algorithm is made to detect peaks, in the figure the thin boxes marked by ‘G’ are the automatically detected peaks. Three dimensional accelerations of ankles, foot pressure and video recorded images which were acquired simultaneously. As the signals were recorded at the same time the synchronized signals could be compared easily.

### 5.2 Developed algorithm for step detection

The gait of a patient can already be measured using a two-dimensional accelerometer, but these two directions are not able to cover all movement. Two-dimensional data can be used to give a quite accurate view on the leg movement as the legs move in a more or less vertical plane. For a more precise view on movements in physical space a third axis is necessary, the Shimmer device can satisfy this need with the three-dimensional accelerometer. The device is mounted on the ankles and the measurements can be done in three dimensions, depicted in figure 5.4, these indicate the x, y and z
axes of the sensor, which are equivalent to horizontal, vertical and transverse direction respectively. Still it has to be kept in mind that the device its coordinate system is moving and rotating along with the lower part of the legs.

5.2.1 Characteristics in the acceleration of the ankle

Characteristic peaks can be determined in the acceleration of a single gait cycle. In the x-direction, according to figure 5.4 a smooth positive peak can be detected in the swing phase, especially during the start of the swing phase. When the toe leaves the ground a sharp positive peak can be recognized. In the y-direction something similar can be seen, a smooth positive peak can be detected in the swing phase and some positive peaks take place near to the toe-off event. The characteristics of these peaks can be different when recording a patient with an abnormal gait. Gait problems can be characterized by peaks with different amplitude and time. Needless to say is that locomotion is necessary to be able to record gait signals. The step detection algorithm is developed to use with patients in early stages of a disease and for sequential observation of a person.
5.3 Implementation of the step detection algorithm in real-time

A new real-time algorithm was elaborated based on the same algorithm as mentioned before. Chapter 4 describes how EyesWeb can be used for processing and visualising data in real-time. First the Shimmer has to be configured to capture the data, this data will be sent to the EyesWeb environment where it can be used as an input for a function block including the algorithm.

First the signals of the Shimmer must be saved and used as input for the algorithm. For each block that is filled with data (1 second or 0.1 second block) calculations for the standard deviation can be done. When values are present for the total standard deviation the threshold can be set and stop/moving and stance/swing can be distinguished.

The standard deviations $\sigma$ for the blocks and also for the whole signal are calculated with the same well know formula:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\chi_i - \mu)^2}$$  \hspace{1cm} (5.1)

In formula [5.1] are the received sample values from the Shimmer written as $\chi$, $N$ is the number of values and $\mu$ is the average or expected value of $\chi$ in the range of $N$ numbers.

As the calculation is made after receiving every value the average of received samples values is used. The formula for the parameter $\mu$ is thus:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} \chi_i$$  \hspace{1cm} (5.2)

After detecting the swing phase peak detection can be done and non-gait peaks should be removed.

5.3.1 Real-time C++ implementation

A flowchart from the real-time algorithm is represented in figure 5.6, every step of the flowchart is explained extensively in the following sections.

Sample rate and block size

The first thing needed is the sample rate of the Shimmer because the size for every block needs to be set. The Shimmer is configured to send values every 5 milliseconds so 200 values are received every second. Block sizes are set to 200 and 20 values, respectively for one and 0.1 second. The total array for received values is set to 12,000, because of this limitation data can only be recorded and used of the last 60 seconds, the reason for this limitation is explained in the paragraph about the rotating window.
Figure 5.5: Principle of a rotating window containing 100 values

A counter is used to identify each received value. An index is computed out of this counter to make sure the values are not saved out of the bounds of the total array. Based on this counter for every block can be checked if the block is filled and thus calculations on the received values can be started.

Threshold calculation

The threshold value that will be later used to define different phases of the gait is based on the total standard deviation. The total standard deviation is a parameter that will relate to the way of walking of a patient, the value will be lower for patients with a shuffling gait. In the post-processing algorithm the threshold value was 25% of the total standard deviation. The real-time algorithm was used with different values for the threshold. Trying values in the range of 20 to 33% of the total standard deviation the best value is fixed to 20%.

Rotating window

As mentioned many times before the threshold value is derived from the total standard deviation. Calculating the total standard deviation is possible for a post-processing algorithm as this only has to be done once, this is somewhat different for a real-time algorithm. In real-time the total standard deviation has to be recalculated so the value is up to date. The more the standard deviation is updated the more precise this value will be but more computation time will be used from the computer. The algorithm will update the value every second and it is possible to change the frequency of the update.

Also for a real-time algorithm calculations will become heavier every time a new value is received. Calculations on an array containing one minute of acceleration signals will not be that hard but problems may arise when recording for a long time, the algorithm may have problems to do calculation in the given time. That is why the rotating window is implemented, for the total standard deviation only the last 60 seconds of data are used. Actually the ‘total standard deviation’ is not the right way to name that value no longer, but the same name is kept for clarity. The value is also a good approximation for the total standard deviation.

How a rotating window works is shown in figure 5.5. The array is able to hold 100 values, when less than 100 values are received the array will hold all the values. However when all places in the array are taken values will have to be deleted in order to save new values, in a rotating window oldest values will be deleted first. The window used in the real-time algorithm contains 60 seconds of data and will thus contain 12000 values.
Moving and swing phase detection

After every second a block is filled and will contains 200 (new) values, as the index of an array starts with zero the first values will have index 0 to 199. A for-loop is used to calculate the average of these values, a second loop will then use this average to compute the standard deviation. The obtained value can than be compared to the threshold and a boolean ‘move’ is set true if the value is greater than the threshold. When the value is lower the algorithm detects the person is not walking (‘stop’ phase) and sets the boolean ‘move’ false.

After every 0.1 second a smaller block is filled with 20 values. When the previous one second block was defined as moving calculation of the standard deviation can be executed. Here the boolean ‘swing’ is set true if the standard deviation exceeds the threshold. In the stance phase the boolean ‘swing’ is set to false.

Peak detection

When a 0.1 second block is detected as swing peak detection will be done. The algorithm will check 0.1 second blocks for a high standard deviation. When this standard deviation is higher than the total standard deviation this block will be defined as a peak, in this way only a small part of the swing phase can be detected as a peak.

Output to file

The final step of the real-time algorithm is to write the data to an output file. When the algorithm is started the file must be made, a file name can be provided in the EyesWeb environment and this file name should be different for each accelerometer stream. After opening the file the column headers will be written on the first line, then data can be written and each new sample value will cause a new line to be written in the file. When execution is stopped the file will be closed. Values are written in a text file and are separated by tabs. It is now possible to open the file in a spreadsheet environment or to read the file with different programs like Matlab or R.
Figure 5.6: Flowchart of the real-time step detection algorithm
Chapter 6

Results

This chapter will discuss the results obtained from the real-time algorithm implemented in the EyesWeb environment. Information will be presented of gathering, output and inferring the information. Furthermore the difference between post-processing and real-time implementation will be explained. At the end of this chapter the recorded gait and the real-time implementation will be evaluated.

6.1 EyesWeb

In figure 6.1 the representation of the total process is visualised. First the Shimmer-block is included, to receive the input signals from the Shimmer. The output data of this block is a matrix, for each acceleration (x, y and z) the values are gathered so they can be used as input for the block which includes the algorithm. In this block the data is gathered and the desired calculations are done. Then the stop and moving phase are determined, as well as stance and swing and the peaks are detected. This data, together with the input value is the output of the created block and will now be submitted to a screen so it can be visualised. During the block is executed the data will be sent to a file, this data contains:

1. Time stamp
2. Input value
3. Stop/moving discrimination (0/1)
4. Stance/swing discrimination (0/1)
5. Peak detection (0/1)
6. Standard deviation of block of last 0.1 second
7. Standard deviation of block of last 1 second
8. Standard deviation of the total signal
Figure 6.1 represents the results of the real-time gait analysis. The algorithm is able to detect the moving phase quite good, the x-direction shows the maximum delay this ‘moving’ detection can have, being one second. During the moving phase swing or stance are detected by the algorithm.

As explained in section 2.2.2 the swing phase lasts for 40% of the gait cycle. The measured result does not match totally with the theoretical information, this is because the acceleration is not directly related to the stance and swing phase. During the initial contact in the stance phase there is still an acceleration of the ankle, the same occurs during the pre swing. The swing phase contains 40% of the total gait cycle, the acceleration of the ankle is approximately 60% of the gait cycle. Another issue that should be kept in mind is the constant movement of the limb during the mid swing, so the accelerometer will not detect an acceleration. This constant speed of the limb can be detected in every direction on some of the gait cycles. Still the distinction between swing and stance can easily be made.

The first peak in the x-direction occurs during the beginning of the swing phase, here a smooth positive peak can be seen. The second peak is caused by the moment when the foot makes contact with the ground again at the end of the swing phase, this is the only peak is detected by the algorithm as it is a lot sharper than the first peak. In the y-direction two peaks are detected. The first peak in the beginning of the swing phase is again a smooth positive peak, when the toe leaves the ground. The second peak is shown when the heel makes the ‘initial’ contact with the ground, at the end of the swing phase. It is evident that the x- and y-direction are the most important and give the best results since the acceleration is most pronounced in these directions when walking, the lateral movement is rather limited.

Table 6.1 represents an example of 25 values for the parameters and standard deviations of the algorithm output for the x-direction, the same output is available for the two other axes. The first value of
Figure 6.2: Results plots
each column contains the name of the output variable. This data can be used as input in R or Matlab for displaying and further processing.

Table 6.1: Data output in the x-direction

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Input value</th>
<th>Moving</th>
<th>Swing</th>
<th>Peak</th>
<th>SD 0.1 sec</th>
<th>SD 1 sec</th>
<th>SD Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>62973</td>
<td>2139</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>209.914</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63133</td>
<td>2033</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>209.914</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63293</td>
<td>1960</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63453</td>
<td>1930</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63613</td>
<td>1920</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63773</td>
<td>1910</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>63933</td>
<td>1890</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64093</td>
<td>1884</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64253</td>
<td>1872</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64413</td>
<td>1840</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64573</td>
<td>1807</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64733</td>
<td>1832</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>64893</td>
<td>1827</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>65053</td>
<td>1819</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>65213</td>
<td>1786</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>65373</td>
<td>1816</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>65533</td>
<td>1816</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>157</td>
<td>1807</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>317</td>
<td>1783</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>477</td>
<td>1779</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>637</td>
<td>1804</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>797</td>
<td>1835</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>281.553</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>957</td>
<td>1818</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>45.0132</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>1117</td>
<td>1815</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>45.0132</td>
<td>217.517</td>
<td>255.177</td>
</tr>
<tr>
<td>1277</td>
<td>1786</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>45.0132</td>
<td>217.517</td>
<td>255.177</td>
</tr>
</tbody>
</table>

6.2 Post-processing versus real-time

The post-processing and real-time algorithms are quite similar, since they have to implement the same functionalities for detecting stop/moving, stance/swing and peak. Still there are several differences.

The most common similarities are the use of 1 second blocks and 0.1 second blocks to determine stop/moving and stance/swing. In both methods the standard deviation of the blocks is calculated and they are compared with a threshold value to discriminate moving and the different phases of the gait cycle. Peaks are only detected during the swing phase, but here some different techniques are used.

The main difference between real-time and post-processing is the calculation of the standard deviation. In post-processing the total signal is already available and so this can be used to calculate the total standard deviation. In real-time only the signal up to that moment is available and can be used for the calculation. Because of that the standard deviation is updated every second. The more values are received, the heavier the calculations would become, using a rotating window keeps the calculations quite light. Only the samples of the last 60 seconds are used to calculate the total standard deviation.
Another difference is the calculation of the threshold value, in post-processing 25% of the total standard deviation is used to obtain the threshold, in real-time this value doesn’t fit the algorithm, different percentages of the total standard deviation were used, and a good result was obtained with 20%.

The third difference is a delay of 0.1 second in real-time between input and output. The smallest block of 20 samples (0.1 second) causes the output to be delayed, calculations can be done after the smallest block is filled.

Finally peak detection was done on the 0.1 second blocks that were defined as swing. Peak detection gave a high output when the standard deviation of the small block was higher than 100% of the total standard deviation.

### 6.3 Evaluation

The developed algorithm allows in real-time to determine if a person is walking or not, and can also determine the stance and the swing phase. During the swing phase peaks can be detected. The x-direction supplies a good result since the swing phase is mostly detected as a continuous block, the flaw of the x-direction is that only the sharpest peak can be detected at phase of the initial contact. The smooth peak in the beginning of the swing phase is not detected. The y-direction shows less continuous blocks but two peaks can be detected, the smooth sinusoidal peak during the swing phase and the sharp peak at phase of the initial contact. The z-direction give the least reliable signal for swing and peak detection but it can be used to support the conclusions from x- and y-directions.

Another conclusion that can be made is the difference between the phases of the gait cycle and the accelerations. The stance phase contains 60% of the gait cycle and the swing 40%. The acceleration in x- and y-direction can be determined as 60% of the gait cycle, looking at the subdivisions of the stance phase, it is clear that in the beginning, during the initial contact, when the heels strikes the ground there is still an acceleration of the ankle, where the Shimmer is mounted. Something similar is visible at the end of the stance phase, during pre swing, here the heel leaves the ground and the accelerations starts already.

During the swing phase there is a constant movement and no acceleration can be detected, this can cause some difficulties, because the algorithm will cannot determine this as a swing phase, but this is only a small part in the middle of the swing phase. The algorithm is thus functioning properly in distinguishing the stop and moving phase and can also discriminate stance and swing phase and can detect peaks.
Chapter 7

Conclusion and future work

7.1 Conclusion

Parkinson’s disease is a neurodegenerative disorder that is characterised by tremor, hypokinesia, rigidity, and postural instability. Problems with speech, swallowing and gait are also strongly related to Parkinson’s disease. The severity of each symptom reveals the progress of the disease. Because measuring every symptom is quite time expensive measurements on one characteristic are done to determine the state of the disease.

With the help of MEMS sensors like accelerometers it is possible to measure the speed and duration of movements. The Shimmer device used in this study contains a three-dimensional accelerometer. The device is small (53×32×25mm) and able to send data wireless using Bluetooth or Zigbee. The long-life battery makes it possible to record accelerometer signals for prolonged periods. Advantages of small wireless devices are that they do not restrict the movements of a patient that much and that there is no need for a hassle with cables to transfer data.

Recorded data can be processed in the EyesWeb environment for step detection using an algorithm coded in C++ code. The algorithm was developed for data post-processing by Jonghee Han (et al.). The real-time algorithm is made in Visual Studio to follow the same steps as the post-processing algorithm. First steps were based on the standard deviation of the received sample values for acceleration over each axis (x, y and z). The algorithm was able to detect whether the patient was walking or not, and if the person was performing his gait the algorithm would detect the stance and swing phase for each limb. As forward and upward acceleration are most pronounced during the activity of walking these two data streams are used for later steps for the algorithm. The original algorithm implemented several ways to distinguish non-gait peaks from the true gait peaks.

The output of the real-time algorithm that was a product of this master thesis included standard deviations of 0.1 and 1 second blocks, the total standard deviation is also saved every time. For the peaks time of occurrence and amplitude are saved as the acceleration peaks for Parkinson’s disease patients tend to be lower as the disease progresses. From this output file several aspects of the gait can be analysed.
The step detection algorithm was developed to use with patients suffering from Parkinson’s disease. Gait abnormalities was always a well known characteristic of Parkinson’s disease, but when this characteristic is clearly visible this kind of technology is no more needed. The measurements with the Shimmer along with the algorithm have the aim to help on a different level. The step detection algorithm for discovering Parkinson’s disease in early stages, measuring progress of the disease over a long timespan, measuring the temporal or longtime effect of medicine.

7.2 Future work

Until now the C++ real-time algorithm is able to distinguish moving/stop for a patient, stance/swing for a limb and peaks in the swing phase. Peak amplitude and time of occurrence are also written in the output file. Data can now be analysed for a lot of gait characteristics: average step speed, average stance/swing ratio, average amplitude of acceleration peaks, etc.

More EyesWeb functionality can be developed to support the non-gait peak removal, this should include the check for peaks in x- and y-direction at the same time. An other opportunity is to delay the output to file with one second, in this way the peak occurrence (and peak amplitude) with the timestamp on which they occur could be written out. Peaks should be analysed carefully as the amplitude will tend to be lower as Parkinson’s disease progresses.

Gait analysis can be an important way to monitor a patient suffering from Parkinson’s disease. Recurring tests can show the difference in gait characteristics objectively. This kind of monitoring could also be used on the swallowing and respiration of a person. Different kind of Shimmer devices are able to perform such measurements.
Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADL</td>
<td>Activities of daily life</td>
</tr>
<tr>
<td>ADP</td>
<td>Alzheimer’s disease with parkinsonism</td>
</tr>
<tr>
<td>ASAP</td>
<td>Advanced Sensing for Assessment of Parkinson’s disease</td>
</tr>
<tr>
<td>ASUR</td>
<td>Autonomous Sensing Unit Recorder</td>
</tr>
<tr>
<td>BSL</td>
<td>Boot Strap Loader</td>
</tr>
<tr>
<td>CBGD</td>
<td>Corticobasal Ganglionic Degeneration</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>EMG</td>
<td>Electromyogram</td>
</tr>
<tr>
<td>HRG</td>
<td>Hemispherical resonant gyroscope</td>
</tr>
<tr>
<td>ID</td>
<td>Identifier</td>
</tr>
<tr>
<td>IIR</td>
<td>Infinite Impulse Response</td>
</tr>
<tr>
<td>L-DOPA</td>
<td>Levodopa</td>
</tr>
<tr>
<td>MEMS</td>
<td>Microelectromechanical systems</td>
</tr>
<tr>
<td>MPPP</td>
<td>1-methyl-4-phenyl-4-propionoxy-piperidine</td>
</tr>
<tr>
<td>MPP+</td>
<td>1-methyl-4-phenpyridinium</td>
</tr>
<tr>
<td>MPTP</td>
<td>1-methyl-4-phenyl-1,2,3,6-tetrahydropyridine</td>
</tr>
<tr>
<td>OPCA</td>
<td>Olivopontocerebellar Atrophy</td>
</tr>
<tr>
<td>PD</td>
<td>Parkinson’s disease</td>
</tr>
<tr>
<td>PDCG/ALS</td>
<td>Parkinsonism-Dementia-Amyotrophic Lateral Sclerosis Complex of Guam</td>
</tr>
<tr>
<td>PSP</td>
<td>Progressive Supranuclear Palsy</td>
</tr>
<tr>
<td>RISC</td>
<td>Reduced Instruction Set Computer</td>
</tr>
<tr>
<td>RMS</td>
<td>Root-Mean-Square</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root-Mean-Square Error</td>
</tr>
<tr>
<td>RN</td>
<td>Roving Networks</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Developer’s Kit</td>
</tr>
<tr>
<td>SDS</td>
<td>Shy-Drager syndrome</td>
</tr>
<tr>
<td>SHIMMER</td>
<td>Sensing Health with Intelligence, Modularity, Mobility &amp; Experimental Reusability</td>
</tr>
<tr>
<td>SND</td>
<td>Strialonigral Degeneration</td>
</tr>
<tr>
<td>TRANSCEIVER</td>
<td>Transmitter/Receiver</td>
</tr>
<tr>
<td>UPDRS</td>
<td>Unified Parkinson’s Disease Rating Scale</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
</tbody>
</table>
Bibliography


53


# Appendix A

StepDetection.h

```cpp
#pragma once

class CStepDetection : public Eyw::CBlockImpl
{
public:
    CStepDetection(const Eyw::OBJECT_CREATIONCTX* ctxPtr);
    ~CStepDetection();

    static const int SR = 200; // Sample rate

    // Define window sizes, indexes and bounds
    // A: Small window (0.1 sec)
    // B: Medium window (1 sec)
    // C: Large window (60 sec)
    static const int windowSizeA = SR / 10;
    static const int windowSizeB = SR;
    static const int windowSizeC = SR * 60;
    static const int numberOfAwindows = windowSizeB / windowSizeA;
    static const int numberOfBwindows = windowSizeC / windowSizeB;
    int windowIndexA;
    int windowIndexB;

    int count; // Counter for input sample values
    int inputValue; // Acceleration value (int) received from the Shimmer
    int timestamp;

    // Array containing last received sample values
    double inputValues[windowSizeC];

    // Outputs that need to be saved until next update
    int output1, output2, output3;

    // standard deviation of 1 and 0.1 sec windows (B and A windows resp.)
    double stdDevB[numberOfBwindows];
    double stdDevA[numberOfAwindows * numberOfBwindows];

    // total standard deviation (last samples up to 1 minute)
    double stdDevTot;
};
```
bool move; // true if previous B block was 'moving' phase (false = stop)
bool swing; // true if previous A block was 'swing' phase (false = stance)
bool peak; // true if previous A block was 'peak' (false = non peak)

// Data output to file
ofstream fout;
string FileName;

protected:
  virtual void InitSignature(); // should also initialize layout and private data
  virtual void CheckSignature();
  virtual void DoneSignature();

  // Actions
  virtual bool Init() throw();
  virtual bool Start() throw();
  virtual bool Execute() throw();
  virtual void Stop() throw();
  virtual void Done() throw();

private:
  // Block param pins
  Eyw::string_ptr _pParamFilename;
  // Block input pins
  Eyw::int_ptr _pInSignal;
  Eyw::int_ptr _pInTimestamp;
  // Block output pins
  Eyw::int_ptr _pOutMoveStop;
  Eyw::int_ptr _pOutStanceSwing;
  Eyw::int_ptr _pOutPeak;
  Eyw::int_ptr _pOutInputSignal;

};
Appendix B

StepDetection.cpp

```cpp
#include "StdAfx.h"
#include "StepDetection.h"
#include "Signature.h"
#include "resource.h"

using namespace Eyw;

// Signature
static block_class_registrant blockClassInfo(
    block_class_registrant::block_id( "StepDetection" )
    .begin_language( EYW_LANGUAGE_US_ENGLISH )
    .name( "Step Detection" )
    .description( "Step detection based on accelerometer signals" )
    .libraries( "Shimmer" )
    .bitmap( IDB_STEPDETECTION_BITMAP )
    .end_language()
    .begin_authors()
    .author( EYW_SHIMMERALGORITHM_CATALOG_AUTHOR_ID )
    .end_authors()
    .begin_companies()
    .company( EYW_SHIMMERALGORITHM_COMPANY_ID )
    .end_companies()
    .begin_licenses()
    .licence( EYW_SHIMMERALGORITHM_LICENSE_ID )
    .end_licenses()
    .default_factory< CStepDetection >()
);

// Identifiers
#define PARAMETER_FILENAME "filename"
#define INPUT_SIGNAL "signal"
#define INPUT_TIMESTAMP "timestamp"
#define OUTPUT_MOVESTOP "moveStop"
#define OUTPUT_STANCESWING "stanceSwing"
#define OUTPUT_PEAK "peak"
#define OUTPUT_INPUTSIGNAL "inputSignal"
```

CStepDetection::CStepDetection( const Eyw::OBJECT_CREATIONCTX* ctxPtr )
{
  pInSignal=NULL;
  pInTimestamp=NULL;
  pOutMoveStop=NULL;
  pOutStanceSwing=NULL;
  pOutPeak=NULL;
  pOutInputSignal=NULL;

  schedulingInfoPtr->SetActivationEventBased( true );
  schedulingInfoPtr->GetEventBasedActivationInfo();
  ->SetActivationOnInputChangeED( INPUT_SIGNAL, true );
  schedulingInfoPtr->GetEventBasedActivationInfo();
  ->SetActivationOnInputChangeED( INPUT_TIMESTAMP, true );

  windowIndexA = 0;
  windowIndexB = 0;
  output1 = 0, output2 = 0, output3 = 0;
  count = 0;
  stdDevTot = 0;
  inputValue = 0;
  move = false;
  swing = false;
  peak = false;

}

CStepDetection::~CStepDetection()
{
}

void CStepDetection::InitSignature()
{
  pParamFilename = Eyw::Cast<Eyw::IString*>(
    SetParameter(Eyw::pin::id(PARAMETER_FILENAME)
      .name("filename")
      .description("Output file name")
      .type<Eyw::IString>()
    )->GetDatatype() );
  SetInput(Eyw::pin::id(INPUT_SIGNAL)
    .name("signal")
    .description("accelerometer 1-axis input")
    .type<Eyw::IIInt>()
  );
  SetInput(Eyw::pin::id(INPUT_TIMESTAMP)
    .name("timestamp")
    .description("Sample time")
    .type<Eyw::IIInt>()
  );
  SetOutput(Eyw::pin::id(OUTPUT_MOVESTOP)
    .name("moveStop")
    .description("Move (high) or stop (low) discrimination")
    .type<Eyw::IIInt>()
  );
SetOutput(Eyw::pin::id(OUTPUT_STANCESWING)
  .name("stanceSwing")
  .description("Swing (high) or stance (low) discrimination")
  .type<Eyw::IInt>());
SetOutput(Eyw::pin::id(OUTPUT_PEAK)
  .name("peak")
  .description("Acceleration peak detection")
  .type<Eyw::IInt>());
SetOutput(Eyw::pin::id(OUTPUT_INPUTSIGNAL)
  .name("inputSignal")
  .description("Original accelerometer signal")
  .type<Eyw::IInt>());
_pParamFilename->setValue("output.txt");

void CStepDetection::CheckSignature()
{
  _pParamFilename = get_parameter_datatype<Eyw::IString>(PARAMETER_FILENAME);
  _signaturePtr->GetInputs()->FindItem(INPUT_SIGNAL);
  _signaturePtr->GetInputs()->FindItem(INPUT_TIMESTAMP);
  _signaturePtr->GetOutputs()->FindItem(OUTPUT_MOVESTOP);
  _signaturePtr->GetOutputs()->FindItem(OUTPUT_STANCESWING);
  _signaturePtr->GetOutputs()->FindItem(OUTPUT_PEAK);
  _signaturePtr->GetOutputs()->FindItem(OUTPUT_INPUTSIGNAL);
}

void CStepDetection::DoneSignature()
{
  _pParamFilename=NULL;
}

// Actions
bool CStepDetection::Init() throw()
{
  try
  {
  
  _pInSignal = get_input_datatype<Eyw::IInt>(INPUT_SIGNAL);
  _pInTimestamp = get_input_datatype<Eyw::IInt>(INPUT_TIMESTAMP);
  _pOutMoveStop = get_output_datatype<Eyw::IInt>(OUTPUT_MOVESTOP);
  _pOutStanceSwing = get_output_datatype<Eyw::IInt>(OUTPUT_STANCESWING);
  _pOutPeak = get_output_datatype<Eyw::IInt>(OUTPUT_PEAK);
  _pOutInputSignal = get_output_datatype<Eyw::IInt>(OUTPUT_INPUTSIGNAL);

  return true;
  }
  catch (...) 
  {
    return false;
  }
}
bool CStepDetection::Start() throw() {
    try {
        FileName = _pParamFilename->GetValue();

        // Open file for appending
        fout.open(FileNane.c_str(), ios::app);
        // assert(!fout.fail());
        // Send column headers to file
        fout << "Timestamp" << "\t" << "Input value" << "\t" << "moving" << "\t" << "Swing" << "\t" << "Peak" << "\t" << "SD 0.1 sec" << "\t" << "SD 1 sec" << "\t" << "SD Total" << endl;
        return true;
    }
    catch(...) {
        return false;
    }
}

bool CStepDetection::Execute() throw() {
    try {
        // Read and save input value
        inputValue = _pInSignal->GetValue();
        timestamp = _pInTimestamp->GetValue();
        int index = count%windowSizeC;
        inputValues[index] = inputValue;

        // Calculate stdDevTot
        int period = 1; // Once every [period] seconds
        if(index%(windowSizeB*period)==(windowSizeB*period)-1) {
            // Define buffersize by 'count' if not yet filled
            int bufferSize = (count<windowSizeC)?count:windowSizeC;
            double sum = 0;
            double dev = 0;
            for(int i=0;i<bufferSize;i++) {
                sum += inputValues[i];
            }
            for(int i=0;i<bufferSize;i++) {
                dev += pow(inputValues[i]-sum(bufferSize),2.0);
            }
            stdDevTot = sqrt(dev/bufferSize);
        }

        // Calculate STDDEV for 1 sec window (B) that has been filled
        if(index%windowSizeB==windowSizeB-1) {

    }
windowIndexB = index / windowSizeB;

double sum = 0;
double dev = 0;
for(int i = windowIndexB * windowSizeB; i < (windowIndexB + 1) * windowSizeB; i++){
    sum += inputValues[i];
}
for(int i = windowIndexB * windowSizeB; i < (windowIndexB + 1) * windowSizeB; i++){
    dev += pow(inputValues[i] - sum / windowSizeB, 2.0);
}
stdDevB[windowIndexB] = sqrt(dev / windowSizeB);

// Define 1 sec block as move/stop
if (stdDevB[windowIndexB] > 0.20 * stdDevTot) {
    move = true;
} else {
    move = false;
}
output1 = move ? 150 : 0;

// Calculate STDDEV for 0.1 sec window (A) that has been filled
if (move) {
    if (index % windowSizeA == windowSizeA - 1) {
        windowIndexA = index / windowSizeA;
        double sum = 0;
        double dev = 0;
        for(int i = windowIndexA * windowSizeA; i < (windowIndexA + 1) * windowSizeA; i++) {
            sum += inputValues[i];
        }
        for(int i = windowIndexA * windowSizeA; i < (windowIndexA + 1) * windowSizeA; i++) {
            dev += pow(inputValues[i] - sum / windowSizeA, 2.0);
        }
        stdDevA[windowIndexA] = sqrt(dev / windowSizeA);
    }
    // Define 0.1 sec window (A) as stance/swing
    if (stdDevA[windowIndexA] > 0.2 * stdDevTot) {
        swing = true;
    } else {
        swing = false;
    }
    output2 = swing ? 100 : 0;
} else {
    swing = false;
    output2 = swing ? 100 : 0;
}

if (move) {
    if (index % windowSizeA == windowSizeA - 1) {
        // Define 0.1 sec window (A) as peak/nonpeak
        if (stdDevA[windowIndexA] > stdDevTot) {
            peak = true;
        } else
peak = false;
output3 = peak ? 50 : 0;
}
else {
    peak = false;
    output3 = peak ? 50 : 0;
}

// Output stop/moving
_pOutMoveStop->SetValue(output1);
// and tell EyesWeb we have an output
_pOutMoveStop->SetCreationTime(_clockPtr->GetTime());

// Output stance/swing
_pOutStanceSwing->SetValue(output2);
_pOutStanceSwing->SetCreationTime(_clockPtr->GetTime());

// Output peaks
_pOutPeak->SetValue(output3);
_pOutPeak->SetCreationTime(_clockPtr->GetTime());

// Output original signal
_pOutInputSignal->SetValue(inputValue/10);
_pOutInputSignal->SetCreationTime(_clockPtr->GetTime());

// Send data to file
fout << timestamp << "\t"
    << inputValue << "\t"
    << move << "\t"
    << swing << "\t"
    << peak << "\t"
    << stdDevA[windowIndexA] << "\t"
    << stdDevB[windowIndexB] << "\t"
    << stdDevTot << endl;
    count++;
}
catch (...) {
    return true;
}

void CStepDetection::Stop() throw ()
{
    try {
        // Reset values when EyesWeb patch is stopped
        count = 0;
        stdDevTot = 0;
        inputValue = 0;

        // Close file
        fout.close();
        // assert(!fout.fail());
void CStepDetection::Done() throw()
{
    try
    {
        pInSignal = NULL;
        pInTimestamp = NULL;
        pOutMoveStop = NULL;
        pOutStanceSwing = NULL;
        pOutPeak = NULL;
        pOutInputSignal = NULL;
    }
    catch (...)
    {
    }
    catch (...)
    {
    }
}

Listing B.1: StepDetection.cpp
Appendix C

Step detection flowchart

start recording

updated input value

200 values received? yes calc total std dev

no

200 values received? yes calc 1s std dev

no

std dev > th yes moving

no stop

moving? yes 20 values received? yes calc 0.1s std dev

no

std dev > th yes swing

no stance

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

calc 1s std dev

std dev > th yes moving

no stop

moving? yes 20 values received? yes calc 0.1s std dev

no

std dev > th yes swing

no stance

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

detect peaks

write data to file

detect peaks

calc 0.1s std dev

yes

no

swing

stance

dete
Appendix D

EyesWeb visual output

(a) X axis output display, test 1

(b) X axis output display, test 2

(c) Y axis output display, test 1

(d) Y axis output display, test 2

(e) Z axis output display, test 1

(f) Z axis output display, test 2
Appendix E

Output file plot

![Graphs showing acceleration of X, Y, and Z axes with original signal, moving, swing, and peak markers.](image-url)
Appendix F

R code used for plots

```r
# sample 32500 to 34300

# Plot options
begin=32500
end=34300
plotwidth=12
plotheight=4
maxaccel=3500

# X output

# Read data as 8 columns of numerics
data<-'scan("x-2011-06-16-18u30.txt",what=list(0,0,0,0,0,0,0,0))
# Put column names
names(data)<-'c("Timestamp","Input","Moving","Swing","Peak","STDV-A","STDV-B","STDV-C")

pdf(file="x-32500-to-34300.pdf",width=plotwidth,height=plotheight)
# Trim off excess margin space (bottom, left, top, right)
par(mar=c(4.2, 4.1, 0.6, 0.2))

# Plot acceleration and add lines for move & swing & peak
plot(data$Input[begin:end], ylab="Acceleration of X axis",xlab="Number of sample", type="l",col="navy",lwd=2. ylim=c(0,maxaccel),cex.axis=0.75)
lines(data$Moving[begin:end]*900,col="darkorange",lwd=2)
lines(data$Swing[begin:end]*600,col="gold",lwd=2)
lines(data$Peak[begin:end]*300,col="olivedrab",lwd=2)
# Put legend in upper left corner
legend(0,maxaccel,"c("Original signal","Moving","Swing","Peak")

# Turn off device driver (to flush output to PDF)
dev.off()
```

Listing F.1: xplot.R