UNIVERSITY OF TWENTE.

MIRA - Institute of Biomedical Technology and Technical Medicine

Medisch Spectrum Twente

Department of Pulmonary Medicine

Master thesis

Sleep Apnoea Detection Using Small & Cheap Sensors

J.R. Benistant, BSc

Supervisors
Drs. M.M.M. Eijsvogel, MD
Dr. ir. F.H.C. de Jongh
Drs. P.A. van Katwijk

Graduation committee
Prof. dr. S.A. van Gils (Chair)
Drs. M.M.M. Eijsvogel, MD
Dr. ir. F.H.C. de Jongh
Drs. P.A. van Katwijk
Prof. dr. ir. M.J.A.M. van Putten, MD

May 24, 2016

J.R. Benistant, BSc
Sleep Apnoea Detection Using Small & Cheap Sensors
Master thesis, May 24, 2016

University of Twente
MIRA - Institute of Biomedical Technology and Technical Medicine
Drienerlolaan 5
7522 NB and Enschede, The Netherlands

Medisch Spectrum Twente
Department of Pulmonary Medicine
Koningsplein 1
7512 KZ and Enschede, The Netherlands

Abstract

Sleep apnoea syndrome (SAS) is a condition in which the breathing decreases or even ceases during the sleep. We differentiate two types of SAS; obstructive and central sleep apnoea syndrome. This may lead to short term problems but there are also ample long term consequences. It would be desirable to treat all patients suffering from SAS. This however, would require a total population screening, since not everyone with SAS is aware of it.

We would like to introduce a tool that is cheap enough to be used in large quantities but also small enough to not interfere in the patient's sleep. Besides, this tool could use automated analysis methods of existing software to report back the results. The main purpose of such a tool would be to cost-efficiently screen for SAS in large populations.

We have designed a sensor capable or registering the respiratory effort and pressure signal. Our sensors use Bluetooth low energy and connect to a storage device. Our data is pre-processed and analysed in existing sleep apnoea analysing software. In this pilot study we have included 10 patients in the period November-December 2015. We used 4 accelerometers positioned lateral on the thoracic and abdominal respiratory inductance plethysmography (RIP) bands, a pressure sensor connected to a nasal cannula, a pulse oximeter and laptop microphone during our registrations.

We found an agreement of 87.1% of the registered events with the gold standard, the polysomnography (PSG). Our pressure signal showed a very good linear correlation (r^2=0.9451) with the PSG's pressure signal. Our respiratory effort signal has a high linear correlation (r^2=0.8381) with the PSG's RIP band and our registered oxygen desaturation index has a maximum deviation of 1 event/hour. Our analysis would have scored 7 out of 9 patients in the right sleep apnoea category.

We have successfully created a system that is applicable in clinical setting. The analysis is already done by a sleep specialist and applying the sensors is easier than connecting a patient to the PSG sensors. With this study we have shown that it is possible to reliably determine SAS by using cheap electronics, a laptop and existing analysing software.

Contents

List of Acronyms

1 Tick	1/10.000.000 Second
AASM	American Academy of Sleep Medicine
AHI	Apnoea-Hypopnoea Index
BASIC	A programming language
BMI	Body Mass Index
BPM	Beats Per Minute
BTLE	Bluetooth Low Energy - Wireless communication with minimal power
CPAP	Continues Positive Air Pressure - Treatment option for SAS
CSAS	Central Sleep Apnoea Syndrome
dB FS	Decibel Full Scale
dB SPL	Decibel Sound Pressure Level
ECG	Electrocardiography (registers electrical activity of the heart)
EEG	Electroencephalography (registers electrical activity of the brain)
EMG	Electromyography (registers muscle activities)
EOG	Electrooculography (registers eye movement)
Flashing	Placing new code on a electronic chip
GUI	Graphical User Interface
Interpreter	A computer program that can run un-compiled code
MRD	Mandibular Repositioning Device
ODI	Oxygen Desaturation Index
OSAS	Obstructive Sleep Apnoea Syndrome
Pairing	Setting up an initial connection between two Bluetooth devices
PCB	Printed circuit board
PG	Polygraphy (a simplified PSG)
PLMD	Periodic Limb Movement Disorder
pOSAS	Positional Obstructive Sleep Apnoea Syndrome
PSG	Polysomnography (current gold standard for diagnosing sleep apnoea syndrome)
RIP band	Respiratory Inductance Plethysmography band
RLS	Restless-Leg Syndrome
SAS	Sleep Apnoea Syndrome
SpO_2	Arterial oxygen saturation measured at the fingertip

1 General Introduction

1.1 Sleep Apnoea Syndrome

Sleep apnoea syndrome is a condition in which the breathing decreases or even ceases during the sleep [25]. We differentiate two types of sleep apnoea syndrome; obstructive sleep apnoea syndrome (OSAS) in which the pathophysiological increased collapsibility of the upper airways stands central and central sleep apnoea syndrome (CSAS) in which the breaching control centre, the brains, do not provide an impulse to the respiratory muscles [48]. This may lead to short term problems like fatigue and decreased concentration [16]. There are however also ample long term consequences for sleep-disordered breathing: systemic hypertension, increased cardiovascular risk, developing of insulin resistance and diabetes mellitus, a higher prevalence of psychiatric comorbid conditions (e.g. dementia), increased cancer and all-cause mortality [28, 33, 46, 48, 49].

1.1.1 Types of Sleep Apnoea Syndrome

A difference in pathophysiology exist between OSAS and CSAS. In OSAS, the increased collapsibility of the upper airways causes a restriction in airflow and thus restrict the breathing. This condition is most common in obese, male patients. Large neck circumference, narrow airways, smoking and the consumption of alcohol are well-known risk factors to this condition [48]. A common subset is positional OSAS (pOSAS) in which the occurrence of OSAS depend on the position. In supine position the symptoms of OSAS are present, while normal sleeping is seen in left or right position. Depending on the criteria used, 23% till 65% of the OSAS group has pOSAS [6, 24]. In CSAS however, the cause is of neurological nature. Approximately 4% of the sleep apnoea syndrome (SAS) group has CSAS only, while most CSAS patients have a combination of central and obstructive sleep apnoea. Amongst systolic heart failure patients, a prevalence of 30%-40% is seen [18, 27]. The signal to the breathing muscles is reduced or even missing for a couple of seconds till a minute. This type of apnoea is less common and associated with some degree of heritability. Cheyne-Stokes breathing, a specific breathing pattern seen in CSAS patients, is most common in elder men with chronic congestive heart failure. It is seen in male patients who have had a stroke but rarely seen in women. A mix of obstructive and central SAS is possible as well [13, 17, 22].

1.1.2 Diagnosticate

Diagnostics of SAS are complicated by two factors: first the identification by the doctor and secondly the comprehensive and intensive form of diagnosis. Symptoms associated with OSAS may not always be straight forward: daytime sleepiness or fatigue, concentration problems, restlessness during sleep, snoring, trouble getting up in the mornings all indicate sleep problems but on their own would not identify SAS [16]. Having overweight, having a large neck circumference or narrowed airway, a high blood pressure, diabetes, smoking, morning headaches, waking with a sore or dry throat or even being of the male gender also could indicate having SAS [11, 42, 48]. The non-specific symptoms may not indicate SAS on their own, but the combination could. Often it is the partner who complains about snoring and suggesting to see a doctor. People with non-specific symptoms not always have a partner to recognize snoring and to send the doctor in the direction of SAS.

The second problem lays in the method of diagnostics. The current gold standard for diagnosing SAS is polysomnography (PSG) [4]. This is displayed in figure 1.1. This method is expensive and requires the patient (mostly) to spend the night in the hospital.

The PSG records many signals: thoracic and abdominal effort, nasal pressure, blood saturation and heartbeat with a pulse oximeter, nose/mouth breathing with a

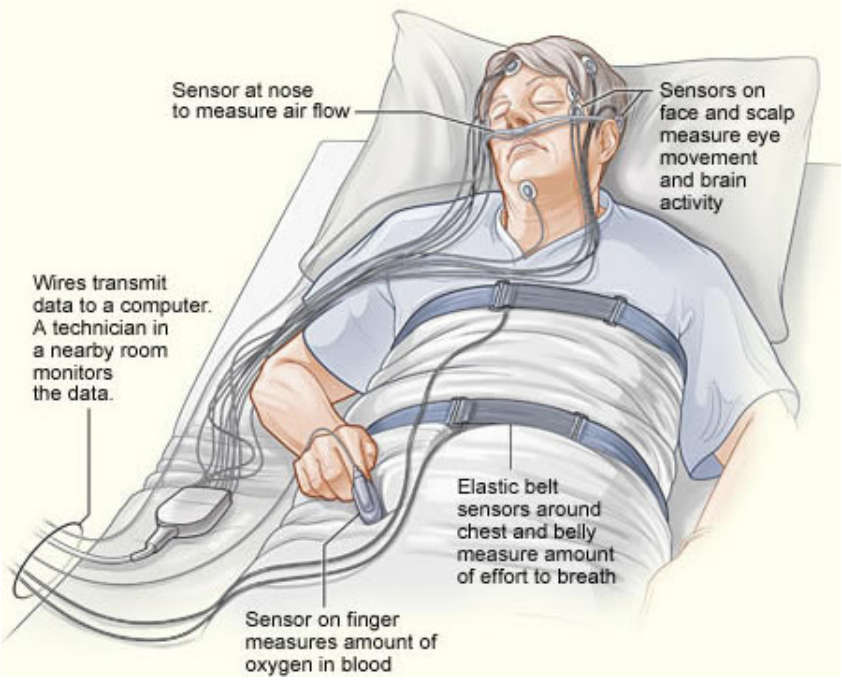


Figure 1.1: Representation of a patient undergoing a polysomnography. This image gives an impression of the large amount of different sensors the patient is connected to during the polysomnography.

pressure sensor and or thermistor, body position, video and audio recording, EEG, EOG and EMG. Having to sleep in the hospital and being attached to many wired sensors does not add to the sleep comfort of the patient. Complaints have been heard that the Respiratory Inductance Plethysmography (RIP) bands press in the side when sleeping on the left or right side. The large registration equipment does not allow the patient to sleep well in prone position. This all makes that the PSG is an uncomfortable method of diagnostics. The many sensors do however register enough information, to be able to diagnose the problem accurately. The PSG method is expensive and time consuming for both patient and sleep specialist. It is thus logical that a stripped version, polygraphy (PG), of this method is used in many hospitals to diagnose SAS. PSG is used only when PG cannot provide a definite answer. The PG uses many of the same sensors as PSG, but does not record EEG, EOG, EMG, video and has no thermistor. The hardware is also portable so that the patient can do the registration in his own bed at home. Even though less sensors are attached, it still can provide enough information for diagnosing SAS. Patients with other symptoms, like restless-leg syndrome (RLS), periodic limb movement disorder (PLMD), insomnia, bruxism and more cannot be diagnosis with this method and still require a PSG. The PG is just like the PSG, not comfortable due to its size. The discomfort likely causes a distortion in the patient's sleep, meaning that the recorded night is not the best representation of average patient's night.

Both the PG ans PSG can provide enough information to detect SAS and its severity. The analysis provides several numbers which are used in establishing, and diagnosing the type of SAS. The most important value is the apnoea-hypopnoea index (AHI). This number gives the average amount of apnoea's and hypopnoea's per hour. The criteria for apnoea and hypopnoea are defined in the American Association for Sleep Medicine (AASM) 2013 guidelines [40], respectively as:

- a flow limitation of 90% for $\geq$ 10 seconds
- a flow limitation of 30-90% for $\geq$ 10 seconds followed by a desaturation of $\geq$3% or an arousal.

The SAS can be classified by severity. An AHI <5 indicates no SAS, between 5-15 is classified as light SAS, 15-30 is moderate SAS and an AHI >30 is defined as severe SAS. Besides the AHI, the oxygen desaturation index (ODI) and supine AHI/ODI are important in diagnosing SAS, to distinguish the different types. E.g. a patient who has almost all apnoea's in supine position, but near to zero in left or right position, would be classified as pOSAS.

A couple of decades ago, the prevalence of SAS in the western world was low. In 1990, the percentage of Dutch overweight (body mass index (BMI)>25) was 35%. This number increased to 50.3% in 2014. The percentage of Dutch obese (BMI>30) people increased from 6.2% in 1990 till 13.6% in 2014 [7]. The awareness as well

as the risks associated with SAS have become more known during the past decades. With an increasing overweight population, the risk on SAS increases as well. In 2013, over 40.000 sleep registrations (combination of PG and PSG) were done in the Netherlands. Using the prevalence of OSAS in 2011, a quick sum learns that at least 235.000 people with undiagnosed and untreated sleep apnoea are out there [23]. It would take many years to catch up with the current diagnostic capacity. An expected increase of prevalence due to increasing BMI would even make catching up take longer.

1.1.3 Treatment

In the Netherlands different treatment options exist for treating SAS. The most common treatment type is by using a continues positive air pressure (CPAP) device. This device offers a continuous positive air pressure to a face mask, to keep the airways open. This treatment option is used for the moderate and severe OSAS patients, as well as the CSAS patients [42]. A less invasive treatment option for OSAS, is the mandibular repositioning device (MRD). This bracket pushes the lower mandible forwards and with it the tongue. This causes the oral cavity to become bigger, allowing air through more easily and removing obstructions [47]. pOSAS used to be treated by applying a tennis-ball on the back, preventing the patient to sleep in supine position. More recent techniques wake the patient using a vibration mechanism when the patient lays in supine position. Once the patient changes position to the left or right side, the vibration stops allowing the patient to continue sleeping [15].

In some cases it is possible to treat OSAS using surgery. The most common procedure is ubulopalatopharyngoplasty, i.e. performing surgery on the uvula and/or removing the almonds and increasing the oral cavity size. To increase the airflow in the nasal

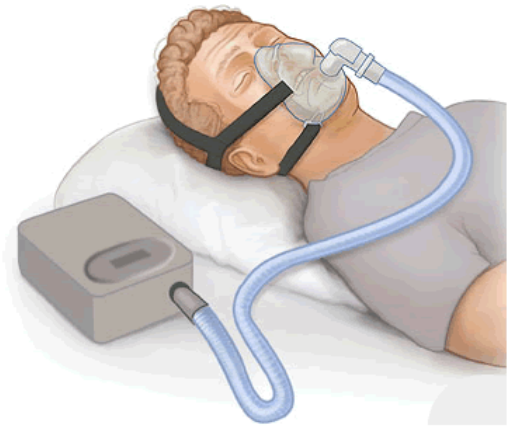


Figure 1.2: A patient using a CPAP device with full-face mask. The device provides a positive air pressure, keeping the upper airways open while sleeping.

cavity, removing polyps from the nose, opening the sinuses or straighten the nasal septum could all help in lowering the amount of apnoea's [25].

1.2 Screening Tools

It would be desirable to treat all patients suffering from SAS. This however, would require a total population screening, since not everyone with SAS is aware of it. The easiest method of screening is by using a questionnaire. The downside of this method is that the current developed questionnaires, like the STOP-bang and Berlin questionnaire have good test characteristics for in-hospital use (i.e. sleep clinic or preoperative risk screening), but are only sparsely tested in primary care with low specificity as outcome. A recently published questionnaire, the Philips questionnaire [14] (a combination of the STOP-Bang questionnaire, Berlin questionnaire and Epworth sleepiness scale) has shown good results for non-hospital setting, but is still awaiting validation in primary care setting. The Philips questionnaire is a good start to detect SAS, but does not have a high enough specificity to rule it out.

In addition to the questionnaire, nightly data is needed to assess the presence of SAS and if so, the type. The current method of PG is expensive, time consuming and interferes with the patient's sleep. Possibly the pulse oximeter alone can establish the presence of SAS, though this technique would not be able to determine the type.

What if we could provide a simple and cheap alternative for diagnosing SAS? By introducing a tool that is cheap enough to be used in large quantities but also small enough to not interfere in the patient's sleep. Besides, this tool could use existing automated analysing software to report back the results. If the tools can be used in home setting and issued by general practitioners, it would not increase the workload of the specialised sleep centre. The main purpose of such a tool would be to cost-efficiently screen for SAS in large populations.

1.3 Technological Challenge

We face two challenges in developing such a tool. First we need a tool that is capable of measuring the right amount of signals, to be able to diagnose SAS. The biggest challenge in this, is to use as much as possible existing techniques, yet providing a small enough sensor to not disturb the patient during its sleep. It should be possible to perform the recordings in a in-home situation, allowing the patient to sleep in the comfort of his own room. The tool must ensure a high quality, continuous signal

during the whole sleeping period. The total cost should remain low to allow our tool to be widely used.

The second challenge is the signal analysis. Our recorded data need to be interpreted in some way. If we can translate our data to the same outcome as found on a PG report, the sleep specialist can use the information without the need of training. Getting our signals from raw data to a report will be challenging. This process should be done in a way that the end user does not require training. This can be achieved by automating the process completely, or making use of existing software for the analysis of the data. This later requires the data to be available in a specific format.

1.3.1 Ultimate Goal

In summary, the ultimate goal of this research is to mimic the current polygraphy using small & cheap sensors only and to reliably determine sleep apnoea syndrome in an as much as possible automated way.

1.3.2 Approach and Study Focus

All in all, this complex project contains quite some technological challenges. It is hereby essential to first identify which signals are mandatory for the diagnosing of SAS. Once we are acquainted with the required signals, it is possible to search for existing electronic sensors we can use.

Several options for replacing the current SAS diagnosis technique have been described in literature [1, 2, 5, 9, 12, 19, 26, 29, 30, 31, 32, 35, 36, 37, 38, 41, 43, 44]. We elaborate on two different upcoming trends in this field of research: (1) contact-free methods (2) simple sensors. In the contact-free group we see methods using sensors that are placed on or around the bed that monitor the sleep during the night. An example is the Beddit® system which is placed on the mattress, under the sheets. It uses piezoelectric elements to accurately detect movement of the body like respiration, heartbeat and snoring. This system communicates with a smart phone to store, analyse and visualize the data. Another recent research has turned a smart phone into a sonar system to accurately detect movement around it. This method can keep track of multiple subjects during the night and shows great similarity with the AHI recorded by PSG [32]. Furthermore many apps use the microphone for sound analysis (mostly snoring) or build-in accelerometer to detect movement on the mattress [31, 43]. In the second group one can find simple sensors for detecting SAS, some are capable of determining the type, some only the presence of SAS.

Most of the researchers chose to use an accelerometer, sometimes accompanied by a gyroscope or even a pressure sensor [9, 19, 38]. Another example is the use of HealthPatch® for the prediction of AHI [44]. Besides using accelerometers, the SleepStrip® uses a thermistor to detect breathing. In combination with a build-in algorithm it is able to detect SAS with a low sensitivity and specificity [36].

In accordance with the AASM guidelines, we need at least the following four signals: arterial oxygen saturation, cardiac variable and two respiratory variables [40].

As part of this study we will designed a sensor that can measure multiple types of data. We will first test these sensors in laboratory conditions to make sure it is safe to use, and if it is able to obtain the data correctly. In addition, we performed a prospective pilot study in the intended target group to investigate the correlation between the gold standard and our method.

1.4 Research Question and Study Objectives

1.4.1 Research Question

The following research question was proposed:

Is it possible to develop an inexpensive, non-invasive measurement system which can correctly and accurately determine the presence of SAS?

1.4.2 Study Objectives

Primary Objectives

To answer this research question, the primary study objective was:

- To reliably determine sleep apnoea syndrome using cheap electronics, a smart phone and existing analysing software.

Secondary Objectives

Secondary objectives included:

- To find which signals and sensors are mandatory for the diagnosing of SAS and its type.

- To validate and compare our system with the current gold standard: PSG.
- To make a system that is applicable in clinical setting that automatically provides clinical outputs.

2 Sensor Design

2.1 Sensor Design

As mentioned in the introduction, we face many technological challenges in finding the right sensor for the job. Prior to being able to understand what sensor we are looking for, we need to know what information is mandatory to obtain the diagnoses SAS. Many hospitals use a Type III portable monitoring device according to the AASM 2013 guidelines. This should at least measure four physiologic variables (arterial oxygen saturation, cardiac variable and two respiratory variables). Some devices also record one or more of the following signals: microphone for snoring, light detector and/or body position sensor [40]. Research has shown that a sole pulxe-oximeter signal has a high positive predictive value for SAS [21]. Using this signal only, it is not possible to discriminate between OSAS, CSAS and pOSAS. To distinguish the first two, an airflow signal is mandatory. To detect pOSAS, a body position sensor is needed. The guidelines also mention respiratory effort sensors. Because obstructive and central apnoeas are not always clearly distinguishable in the airflow signal, the respiratory effort signals can aid and be decisive. As will be explained later, the pulse oximeter registers not only the arterial blood oxygen saturation, but also the heart rate. Even though this signal has less clinical importance, it can be obtained without extra effort and used in the diagnostics. Our sensor should thus at least record the four physiologic variables as well as the body position.

Arterial oxygen saturation This physiologic variable can be measured by taking a blood sample. It would however be an invasive method, and thus not recommended for overnight use. Both PSG as PG setup use a pulse oximeter for recording the arterial oxygen saturation, better known as SpO_2. This stands for 'peripheral capillary oxygen saturation' and is mostly recorded at the fingertip. This method makes use of two light sources: 660 nm and 910nm. These frequencies are respectively absorbed by de-oxygenated haemoglobin and oxygenated haemoglobin. The ratio of light transmitted through the finger, indicates the oxygenation level. Many SpO_2 sensors exist, however the Nonin wristox® is used in the Medisch Spectrum Twente and considered one of the market leaders in the field of wrist worn pulse oximeters.

Cardiac variable This physiologic variable is the heartbeat of the user over time. An electrocardiography (ECG) device is capable of recording this variable by attaching at least 2 leads to the body positioned around the heart. The electric pulse of the

heart is detected by the leads and recorded for analysis. Peak detection software can then find the QRS-complex and calculate the heartbeat [45]. Using the full ECG signal, it is possible to detect SAS using the ECG-derived respiratory signal and the ECG inter-beat intervals [8]. Another example is the use of the HealthPatch® for SAS detection [44]. Early research shows this patch to be able to detect SAS to a certain degree, making it a good sensor to use in our setup. This sensor does have a downside: it is disposable. This would not help in keeping the cost low of the final tool. We found an alternative version, the AmpStrip™, on the crowd funding website indigogo. This little sensor is capable of registering the heartbeat and contains a body temperature sensor and accelerometer as well. More importantly, it is not disposable and uses rechargeable batteries. Even though the funding criteria were met, the developers decided to withhold on sending out the product. Instead, they continued developing it for medical use (instead of the initial sport usage). Since we are only interested in the heartbeat, we can obtain this information from the pulse oximeter.

Two respiratory variables The respiratory variables are the effort done by the thorax and abdomen for breathing and flow detection near the nose or mouth. There are multiple ways to detect the respiratory effort. In the introduction we have seen a method using sonar in a smartphone [32]. This is however a very complex way but does show great potential when further researched. Another method often seen in literature is the use of MEMS accelerometers [1, 3, 5, 38]. An accelerometer registers the local acceleration of the object to which it is attached. It does this for three axis, X, Y and Z. This xyz-vector has a length of 1G in rest, which is the gravitational force. If the sensor moves or rotates, the angle and length of the vector change as well. Some research uses both accelerometer and gyroscopes to increase the precision of position detection [3, 19, 26]. The PSG and PG system however make use of RIP bands. These bands are connected to an oscillator with specific frequencies. The bands contain a zig-zagged wire of which the self-inductance changes while stretching or shrinking. This alters the recorded frequencies after which the input frequencies are compared to the output and converted to a digital respiration waveform of which the amplitude is proportional to the inspired breath volume. This technique is very reliable, however requires large electronic components and two bands, around the thorax and abdomen [10]. Especially the bands are uncomfortable for the patient. We will focus on an accelerometer based sensor to obtain the respiratory effort variables. If positioned correctly, we can use these values to detect the body position as well. During our search for such a sensor, we found many unfit for the job. Some sensors claim to be able to register position within a couple of mm accuracy, but ask top dollar for this. Other sensors, like the motion capture sensors of Xsense are too large and have an unnecessary high sample frequency or even special wireless transmitting frequencies. A sensor we found that meets our criteria, was the RealTag-CC2541 sensor (see figure 2.1). A small printed circuit boards (PCB) with build-in

pressure sensor, accelerometer, on-off switch and Bluetooth 4.0 low energy (BTLE) adapter. The sensor is powered by a CR2032 battery and very compact. Once we obtained this sensor, we found out that the sensor was unstable. The sensor stops transmitting data when it is exposed to a quick acceleration. The second respiratory variable is flow detection. In PSG this is done with a nasal cannula with pressure sensor and a thermistor in front of the mouth. A nasal cannula with pressure sensor is a reliable method to register breathing. The recordings do need to be corrected for difference in temperature. Almost all pressure sensors have a thermometer installed to correct the pressure reading for temperature. A thermistor changes its resistance based on the temperature, which fluctuates due to warm air being exhaled, and cold air being inhaled. This method however requires the thermistor to be right in front of the mouth / nose. If the thermistor is not properly positioned, or moved during sleep, it is possible that it does not register the breathing. For us, the easiest method is to use a temperature corrected pressure sensor in combination with a nasal cannula.

We do need a reliable sensor for our measurements, thus we decided to design our own sensor for this purpose. We think that the respiratory effort is best measured using two sensors per RIP band. Two sensors can register more information than one. If one sensor gets blocked, the other sensor might still be able to obtain the respiratory effort data. Furthermore, if one sensor fails, we have a backup. In total we will use four accelerometers to detect the respiratory effort: two positioned on the thorax and two positioned on the abdomen. Furthermore we will use a pressure sensor which is connected to the nasal cannula to register the pressure (and thus flow) during the night.

Figure 2.1: RealTag CC2541 sensor with accelerometer, pressure sensor and Bluetooth low energy transmitter. This compact sensor has the size of a 2-euro coin and is powered by a small battery on the back.

Body position Since we use an accelerometer based sensor for the detection of the respiratory variables, it allows us to detect the body position as well. This does however require us to position the sensors on specific places. Besides thoracic/abdominal position, literature learns that the head position compared to the thorax might be of

influence on OSAS as well [20]. Since our sensors will be small, we can position one sensor on the forehead to record this extra information.

The sensors are designed in collaboration with the Biomedical Signals and Systems group of the University of Twente.

2.2 Bluetooth Chip

A Bluetooth chip can be bought from store, but requires an antenna and some small components to transmit the data. A ready made PCB with pre-assembled chip as well as an antenna exist and can be integrated with ease in another PCB design. The chip used in this project however is special. Some chips can be reprogrammed using a hardware interface, so that the software, compiled for that specific chip, can be stored in the memory of the chip and executed upon request. This process is referred to as flashing the chip (placing new software in a chip) and is required each time the program changes. This has two downsides: (1) When you are testing new software on the chip, you need to connect the chip to the computer. The chip has a width and length of only a couple of millimetres, making this a difficult task. (2) The software needs to be 'compiled' for this specific chip. The program that can compile software often costs thousands of euro's.

The Bluetooth chip used in this project however, the CC2540 chip, was 'hacked' by T. Wilkinson Ph.D. He created a program to run on the chip, that is able to interpret BASIC code. Since the BASIC code runs on the chip itself, it does not need to be flashed each time a change is made in the code. Nor does it require the expensive compiler. The BASIC code can be uploaded to the chip using a BTLE interface. With this program came another advantage. Normally the Bluetooth chip is used only for sending information and uses little of its capacity. The program however allows us to use the full potential of the chip, and connect sensors to the remaining pins.

2.2.1 Bluetooth Low Energy Sensors

Normal breathing frequency in sleep ranges from 8 to 25 min^{-1} [39]. This corresponds to a frequency of 0.13Hz to 0.41Hz. The Nyquist frequency would thus be $2 \cdot 0.41 \approx 1$Hz, which is the absolute minimum sample frequency we should use to record airflow and thoracic/abdominal movement. For body position recording, one sample per second (1Hz) would suffice as well. If a quick movement or inhaling/exhaling happens, we would like to register that. A sample frequency of 10Hz should suffice for our goal. For testing purposes it would be nice if the sensor can handle 50Hz or higher. The sensors should also be accurate with little noise. An important

aspect is that the sensors can communicate with our Bluetooth chip. More about this topic can be found in Appendix box 10.1. Lastly, it is appreciated if the sensors do not cost more than a couple of euro's each.

Taking these requirements into consideration, two sensors meet all criteria: BMP180 and BMA180. Combined with some small electronics (resistors, capacitors and an on/off switch), a PCB was created. Schematics of this sensor, as well as the PCB design can be found in the Appendix box 10.2 & 10.3.

Figure 2.2: Our custom made printed circuit board containing an on/off switch, pressure sensor, accelerometer and Bluetooth transmitter. On the left top, a white connector for a nasal cannula is shown with the pressure/temperature sensor underneath. The left bottom shows the Bluetooth board. Below this board the accelerometer is positioned. This first design used a CR2032 battery. The box size is 40 x 40 x 20 mm (L x W x H).

2.2.2 Bluetooth Low Energy Sensors Programming

As explained in section Bluetooth Chip, the Bluetooth chip is programmed with BASIC code. This code can be uploaded to the sensor using a Bluetooth connection. This method, instead of connecting the sensor to the pc each time, makes debugging the software easier. The BASIC code is executed on the Bluetooth chip when it is powered on. Using simple BASIC code, we can read the accelerometer and/or pressure/temperature sensor value. To understand these values, we need to read the calibration values once from the sensors. This process is further explained in Appendix box 10.4. A timer in the chip reads this value each 0.1 seconds and sends its information over the BTLE connection to a receiving device. We have written BASIC code to read just one of the sensors, or both sensors at the same time. More information about this process is included in Appendix box 10.5 & 10.6.

2.2.3 Pulse Oximeter

In this study we used the Nonin 4100 (Nonin 4100 Pulse Oximeter, Nonin Medical, Plymouth, Minnesota) pulse oximeter. This device, as shown in figure 2.3, is capable of recording both the pulse oximeter and heart rate. We use data format #7, which sends the information with a sample frequency of 3Hz. The device is battery powered and wrist worn, making it an ideal device to use. The device comes with two type of sensors. One is made of medical grade silicon and can be slipped on the finger. The other sensor needs to be attached using a plaster. We chose to use the later, since it is more likely to stay on the finger during the night.

Figure 2.3: The Nonin 4100 pulse oximeter with flexiwrap sensor. This device is wrist worn and transmits its data over Bluetooth. The flexiwrap sensor is positioned around the tip of the finger using a special plaster.

3 Software Design

3.1 Software

Even if our sensors work as expected, and register data during the night, we still need a device to acquire that data and store it for further analysis. We have acquired software to connect and read the Nonin 4100 sensor, however this software is for testing purposes only and not open source: it is unclear how the program behaves in edge situations, like connection loss or data transmitting failures. For our accelerometer / pressure sensors, no default software exists. We should thus design and create this by our self. In this process we can design the software in a way that it fulfils all our requirements for our intended usage. A requirement is that we need to be able to 'find/detect' our sensors and connect to them. Another requirement is that the software must 'start' the sensor, if the sensor failed to initialize itself. For this we need to send a 'start' command to the sensor. It is not unlikely that a sensor loses its connection during the night. For example, when the signal is blocked by the patient. The software must be able to handle this by trying to reconnect to the sensor. The data received must be stored on the device, in a format we can easily read in an application like Matlab for further analysis. We need visual feedback from the software that all sensors are working as expected. A small graphical user interface (GUI) could provide the needed information to the end-user. Last but not least, if our recording device has a microphone, we could use it to record the sound volume in the room. This information might be usable to identify snoring. These requirements bring us to the following list:

- Search and display all available sensors
- Connect to a sensor
- Start the sensor
- Reconnect to a sensor, if the connection is lost
- Receive and store data from the sensor
- Display the sensors' information
- Record the sound volume

The first problem we face, is what device to use? Current mobile phones do support Bluetooth and BTLE. This means that it is possible to use a smart phone for the above requirements. A smart phone is however not a very practical solution when the code needs to be debugged (fixing errors in the source code). A convenient solution

would be to use a windows based computer. Once the software is fully operational and working, it is still possible to convert it later to a smart phone device.

Search and display all available sensors

Windows based operating systems have a security policy that denies Bluetooth sensors to 'just' connect. On top of that, not all types of Bluetooth devices can connect to a windows based operating system. This security policy means that Bluetooth devices first need to pair with the computer, prior to being allowed to communicate with it. The pairing process is rather simple, but must be done manually. The Bluetooth chip is programmed with a pairing code, which needs to be filled in on the computer. This code consists of 4-6 digits. If the chip does not have a pairing code, it cannot connect to a windows based operating system using the default Bluetooth drivers. After the sensor is paired, the computer can attempt to connect, to 'see' if the sensor is in range. The communication with the Bluetooth sensor is done using special protocols and profiles. More information on this topic can be found in Appendix box 10.7.

The Nonin 4100 device uses a slightly different method. This sensor does not use BTLE, but Bluetooth version 1.1. This older version is still commonly used and has the advantage of being more stable and having a larger range (up to 100m in open field). The downside however is that it has a larger energy consumption and needs two AA batteries for a full night measurement. Just like with the BTLE sensors, the Bluetooth sensor first needs to be paired. The revamped version, the Nonin 3150, is smaller and uses two AAA batteries. Unfortunately these sensors were used in regular PG's and unavailable for our research. The Bluetooth sensor needs a different software connection than our BTLE sensors. This is described in more details in Appendix box 10.8.

Connecting to a sensor

Once a sensor is selected in the GUI, the software will try to connect to that sensor. If the sensor is turned off, the software keeps trying to connect, even though it will fail each time. Once the sensor comes online, the connection will succeed and create a connection. In setting up this connection, a couple of events take place. The most important one is that the software tells the sensor to 'notify' each time new data is available. This way the software does not have to ask (polling) the sensor for new data, meaning that less information is transmitted and thus less energy is needed. The implementation of this method can be found in Appendix box 10.9.

The Nonin 4100 device uses a slightly different technique, and thus different implementation. The software does not ask for new data, but keeps the connection open and polls the device for new information so now and then. This method requires more transmission power, but also creates a more stable signal with a higher throughput. When the connection is created, the software tells the Nonin 4100 device in what format it should send its information. We chose to use format #7, which provides the SpO_2 and heart rate signal with 3Hz. More information as well as code can be found in Appendix 10.10.

Reconnect to a sensor

It is of the utmost importance that a sensor, once the connection is lost, tries to reconnect. There can be different reasons for a connection loss: signal blocked by the patient, sensor out of range (i.e. patient is going to the toilet), battery empty or sensor crashing. BTLE has a lower power output, meaning that the distance it works with is lower than the range of a normal Bluetooth device. This has an advantage, namely it lasts longer with one battery, but also the downside that the signal can easily be blocked by the patient. Not all of the connection lost reasons as listed above, can be solved. An empty battery for example cannot be fixed during the night. A crashing sensor requires the sensor to be reset (turned off and on again) and is also not fixable during the night. A description of the implementation of how the timers try to reconnect, can be found in Appendix box 10.11.

The process for the Nonin device is a bit different. It uses a connection over a virtual serial port, of which the software knows if it is still open. If the serial port connection is closed, it can try to reconnect to it. This process uses only one timer. This timer will attempt to reconnect on a regular base and is stopped as soon as the connection is recreated.

Receive and store data

The implementation of storing data might sound simple, but can be a tricky business when using multiple threats. Important to understand is that the software knows what data it receives, since it knows what sensor sent the data. Depending on the information, it will be stored in a certain format for later analysis. We use three different formats: (1) accelerometer data, (2) pressure & temperature data, (3) saturation & heart rate data. The data is saved with its corresponding timestamp (in Ticks) in a file as binary value to reduce the file size. The problems and solutions

we encountered while implementing this, are described in Appendix box 10.12 & 10.13.

Record sound volume

The sound volume is a parameter which is not mandatory according to the AASM guidelines. We do however use a device with build-in microphone, which we can use to record the sound volume. For the diagnosis of bruxism, or talking while sleeping, we need the actual sound during the night. Recording sound in uncompressed format requires quite a bit of storage space. For snoring it is not important to know the actual sound. It is enough to know the sound volume the patient produces. With a microphone we can record a sounds volume. If our microphone is calibrated, we can calculate the actual decibel sound pressure level (dB SPL) as shown in equation 3.1.

The sound volume n in dB SPL is defined as: [34]

$$n = 20 \cdot log_{10}\left(\frac{P_{eff}}{P_0}\right) dB\ SPL \tag{3.1}$$

in which P_{eff} is the effective sound pressure in Pascal (Pa) and P_0 the reference sound pressure (Pa).

SPL is a logarithmic measure of the effective pressure of a sound relative to a reference value. To calculate this, we need to know the reference value and calibration value of our microphone. The reference value for air is $P_0 = 2 \cdot 10^{-5}$ Pa, but the calibration factor of our microphone is unknown. We can better calculate the sound volume in decibel full scale (dB FS) as shown in equation 3.2 This equation uses the Root Mean Square (RMS) of a part of the sound recording. The length of the sample can vary from a couple of milliseconds till the whole length of a sound file. For live dB FS calculation, short samples of 10-100ms are commonly used. The RMS is calculated using equation 3.3. The outcome in dB FS over time, tells us if the patient snores. Higher values indicate noise in the room, thus snoring and low values indicate no snoring. The software implementation is shown in the Appendix box 10.14.

$$n = 20 \cdot log_{10}\left(RMS\right) dB\ FS \tag{3.2}$$

$$RMS = \sqrt{\frac{\sum_{i=1}^{n} \left(sound\,(i)^2 \right)}{n}} \tag{3.3}$$

Graphical user interface

The graphical interface should contain the basics to aid in performing the research. Using the GUI, it should be possible to connect to a sensor, see some form of logging for problem solving and last but not least, it should be possible to turn off the monitor i.e. when there is no hardware on/off button. To make the software a bit more user friendly, it should provide some feedback to the user, it should display the sensors' values and show if the sensors are connected.

In figure 3.1, the GUI as used in our research is shown. In the left top box, multiple sensors (both accelerometer and pressure) are displayed. The user can select the checkboxes of the sensors to which the software should connect. If the connection is established successfully, the box turns green. Right of this box, the logs are shown. Basic information about problems are displayed for problem solving. On the right top, all connected sensors are displayed. The accelerometer sensors have 4 checkboxes which can be used to indicate where that sensor is positioned on the body. The last received value is displayed, so that the user can see if the connection is actually working. The right bottom shows the pressure signal over time. This is useful for problem solving: if only the numeric value is shown, it is difficult to understand if the recorded data is the breathing of the patient or just noise. Finally, the left bottom side shows the connected Nonin device, its values and a button to turn off the screen.

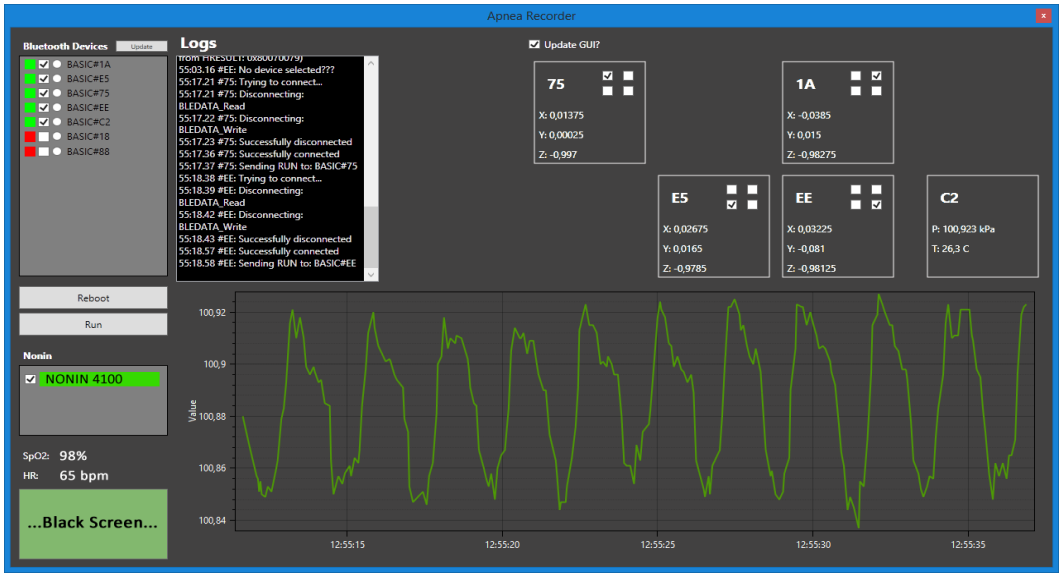


Figure 3.1: The software Graphical user interface used during our research. It allows the user to select which sensors to connect to, and provides feedback to the user.

4 Signal Processing

Our recorded signals need to be processed before they can be analysed. The processing is needed to make the data available in such a format that it can be understood by other programs or analysed using our own methods. This process can be divided into 4 steps: (1) loading data, (2) synchronising the signals, (3) pre-processing, (4) exporting and analysis. Each of these steps are further explained in the sections below.

4.1 Loading Data

In subsection 'Receive and store data' we explained how the data is stored in a file as binary information. Because we know the storage format, we can read this information using another program. We use Matlab (Matlab, The MathWorks Inc., R2015b) because it offers us many tools, like working with arrays, applying filters, plotting signals, etc. In Appendix box 10.12 we explained that the data might not always be stored in time-chronological order. Another problem that might appear is that two data points have the same timestamp. This latter problem is not easily solved, since we can't know for sure what signal was measured earlier. A possible way of solving this problem is by using the wave of the signal, and determine which point is most likely to have arrived first. Because we do not expect this problem to occur often, we assume that the latter data point was in fact recorded later and add a millisecond to the timestamp of that data point to prevent further errors.

Another issue is that there are two methods to interpreted the data: (1) an assumption could be that the sensor is perfect, and records its information with an exact sample frequency of e.g. 10 Hz and sends it in chronological order to the PC. (2) Another interpretation, is that the signal arrival time is each time, a perfect X nanoseconds after measuring it. This interpretation means that the sample frequency of the sensor is not exactly 10 Hz, but varies during the measurement. The first interpretation would make more sense, however the sensor does not provide an increasing ID per measurement, nor does it send a timestamp. It is thus unknown at what exact time a value was measured. The only time we know for sure, is the exact arrival time. We thus use that time as the recorded timestamp of the sensor. The consequence is that the time between two data points is not equal, but variable.

We need to solve this problem before we can further processes and analyse it. The first mentioned problem is solved by sorting the data points based on the timestamp. A gap-finding algorithm is used to detect gaps of more than one seconds, and fills those points with 0 (missing data). Using the linear interpolant method, the data points are turned into a function which can provide the data points for each given time within the provided time-interval. Using a continuous time array, we can calculate the data points and obtain a continuous dataset with equal time in-between. This process is shown in Appendix box 10.15.

The PSG data is exported to a European Data Format (EDF) file, which is a standardised format and known to Matlab.

4.2 Synchronising the signals

To answer our secondary objective, 'to validate and compare out system with the current gold standard: PSG', we need to be able to compare the signals. We have recorded many signals on one computer system, but the PSG data is recorded on a different system. The time on both computers differ (seconds to minutes), meaning that the signals are not in sync. A comparison would not be valid if we did not synchronise the data over time.

The easiest method for synchronisation our data, is by finding the time difference between the two computers and add that to one of the signals. We might be able to synchronise them with an accuracy of a couple of seconds this way. The PSG is however recorded using multiple computers, and it is unclear which computer recorded which patient. A more appropriate method is using signal analysis methods. Assuming we have two signals that have recorded the same parameter during the night, we can calculate the lag of one signal to get a perfect alignment. Both the PSG equipment and our sensor, recorded the pressure value at the end of one nasal cannula attached to the patient. If our system is capable of recording the pressure with approximately the same accuracy as the PSG, we can use this signal to calculate the time offset.

The first step is to visually synchronise both signals, so that they are only a couple of seconds till minute apart. This step is not mandatory, but gives an understanding of the offset we are looking for when the signals are aligned automatically. The shifting can be done using distinctive peaks in the beginning of the signals. The next step is to automate the synchronising process so that the signals are perfectly aligned. Section 4.1 describes how an interpolation function is created. Using this function, we can calculate the value of a data point for every given timestamp within our interval. We

can re- and up-sample our data to any sample frequency we like. The theoretical accuracy can even be 1 Tick. To find the time offset with an accuracy of 10ms, a sample frequency of 100Hz is needed. We used a sample frequency of 4000Hz. This would result in a theoretical accuracy of 250μs. The true accuracy might be a bit worse than that, since we use the interpolant function to generate the missing data. Although 4000 Hz might be a (too) high sample frequency for our signals it was easily applicable (in terms of handling by the computer) and the synchronising frequency might be lowered depending on our results. The implementation is explained in Appendix box 10.16.

4.3 Pre-processing

Some signals need to be processed before they can be used for analysis. We aim in this study to use an external program for the analysis of our results, Noxturnal v4.4.2 (Nox Medical, Reykjavik, build 14926) which is commonly used for analysing PG. One important processing step, is to apply the time synchronisation to all our recorded signals, so that the signals can easily be compared with the PSG outcome.

4.3.1 BPM / Saturation

The beats per minute (BPM) and saturation signal are sent by the Nonin 4100 device. This device has an internal algorithm to calculate the BPM as well as the blood oxygen saturation (SpO_2). The '4-beat pulse rate average' and '4-beat SpO_2 average' signals are used. These signals are shifted over time to synchronise, but not altered in Matlab.

4.3.2 Pressure signal

The pressure value is calculated based on the pressure reading of the sensor, the temperature reading and quite a few static calibration values. The temperature value also needs to be calculated using the calibration values. This process is further explained in Appendix box 10.17. We are however not interested in the absolute pressure value in kPa, but in the fluctuation. We apply a 5^{th} order Butterworth infinite impulse response (IIR) low-pass filter with cut-off frequency of 1Hz and high-pass filter of 0.1 Hz. The first one removes high frequency noise, the second one removes the offset.

4.3.3 Body Position

Both PSG and PG register the body position using one accelerometer, placed at the centre of the abdomen or thorax. In this study we used four accelerometers, however all positioned more lateral compared to the PSG. The mean of two thoracic signals, gives the signal centred between the two sensors. This also goes for the abdominal sensors, resulting in two signals medial on the body. To calculate the body position, the mean needs to be taken once more between these two signals, to get the body position in the middle of the four sensors. Adding them all and dividing by four will however not suffice. It is likely that one (or more) of the sensors failed during the night, e.g. when the patient lays on his side. The sensors don't start at the exact same time, but started a couple of minutes after another.

In Matlab this problem can be solved using arrays. To calculate the mean of two thoracic sensors, we use a 2-dimensional array, filled with zeroes with a width of 2 (sensor left & sensor right) and a length that is long enough to hold all data. For the known time points, the array is filled with data from the sensors after applying the interpolant function on it. For the unknown time points, the array remains 0. A second array with the exact same shape, keeps track of the positions that are filled with a value in the first array, and stores a 1 on those positions. To calculate the mean, the sum of each column of the first array is taken and divided by the sum of each column of the second array. This method is repeated for the abdominal sensors and finally repeated for those two mean values to obtain the body position in the centre of the four sensors on the body. The implementation is shown in Appendix box 10.18. This method only works if the orientation of the sensors are the same.

For the body position, we are not interested in little movements of the accelerometers. We are however interested in the 'offset' value, which indicates what vector of the sensor was recording the highest acceleration and thus indicating the position of our patients. The signal we calculated above, was first filtered using a 2^{nd} order Butterworth IIR low-pass filter with a cut-off frequency of 0.1 Hz. This removes all noise as well as the breathing signal, but leaves the offset in tact.

A final step of processing is required to make Noxturnal understand our data. The Nox T3 system, the system used for PG has the accelerometer rotated compared to us. We have to swap the values of the Y and Z axis to make the data valid for Noxturnal.

We performed a little test to see what vector values should be considered to indicate a certain position: supine, left, right, prone and upright. We created our own body position script around these testing values. The script first checks if the Z-axis is

above a certain value (indicating supine position), or below a value (indicating prone position). Then it overrules this outcome if the X-axis is above or below a certain value (left / right). Finally, it checks if the Y-axis is above or below a certain value, which indicates that the patient is sitting or standing (or upside down). We use this technique to calculate the patient's position for each moment in time.

4.3.4 Respiratory effort

Both PSG and PG use the RIP technique to record the respiratory effort. As explained in section 2.1, we use accelerometers to do this. Our sensors are positioned on the body, at the mid-clavicular line left and right, attached to the thoracic and abdominal RIP band. Figure 4.1 shows the expected results for the accelerometer and RIP band in different positions. When the person breathes in, the red box moves towards the orange box. If the sensor is positioned perfectly in the centre of the body, and inspiration causes no rotation of the sensor, and the patient lays in supine position, the sensor will move perfectly over the Z-axis and we should not see any signal in the X-axis and Y-axis. If the patient moves to his side and keeps breathing perfectly over one axis, we would still see all the signal in the Z-axis, due to the fact that the sensor rotated with the body. The gravity however is now seen in the X-axis. A more realistic example is the third image in which the sensor is positioned on the mid-clavicular line. The sensor will rotate around its own axis while inhaling, shifting the data signal from mostly the Z-axis, to the X-axis and back when exhaling. Depending on how the patient lays and how the sensor is positioned, the signal can shift to the Y-axis as well.

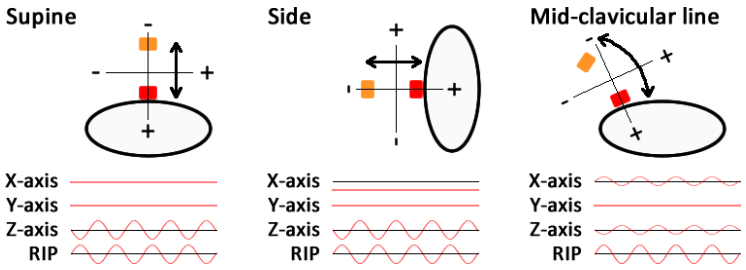


Figure 4.1: An accelerometer positioned on a body

We have tried two techniques to calculate the RIB band values based on the accelerometers. For each step the beginning is the same. The start and stop time of the signal are sought for. Using this, a new time series with equal interval is created. This time series and the interpolant function for our X-, Y- and Z-axis are used to calculate a 10Hz signal. The same method is applied to the PSG RIP band data: first an interpolant function is calculated, then the same time series are applied to calculate the corresponding values with a 10Hz sample frequency. Based on

our sensor information, the body position is calculated as explained in subsection 4.3.3.

The first method assumes that the signal is in all axis and that the sum of these signals returns RIP signal we seek. We use the same accelerometer signal for two different purposes. The first is to extract the breathing effort information, the second is to find the acceleromer's global orientation during the registration. For the first signal, we apply a a 2^{nd} order Butterworth IIR filters, highpass 0.1Hz and lowpass 0.5Hz. The orientation signal is filtered with a 2^{nd} order Butterworth IIR filters, lowpass 0.1Hz and smoothed over 10 samples. If in the global orientation of the sensor has X-axis > 0, the data signal is inverted for those points. This also applies for Y-axis > 0 and Z-axis < 0. The sum is taken of the X-, Y- and Z-axis. This process is applied on both the left and right sensor data. Finally, the sum is taken of the left and right sensor and returned as the RIP signal. This process is completed for the thoracic sensors and for the abdominal sensors. The signal returned, has the unit m/s^2 for a signal of which we say is the representation of volume difference.

A more logical method is to use the accelerometer signals and calculate the integrate twice. This would give us a value in 'meters', which is the x, y and z position of the sensor over time. While integrating, we don't know the value of the constant variable 'C'. The signals are integrated twice using the Matlab function 'cumtrapz'. Those results are than processed the same way as described above, with an orientation signal to help in adding the numbers. The outcome has a unit in meters and indicates the distance the sensor travels when breathing in/out.

4.3.5 Sound Volume

Subsection 3.1 explains how the sound volume in dB is calculated. This signal is not further pre-processed.

4.4 Exporting and Analysing

4.4.1 Export

We use Noxturnal to analyse the data, as if it was PG data. For this to work, we need to provide the data in a format Noxturnal can handle. For Noxturnal, each signal needs to be stored in its own file. The data is stored in binary form, but the file itself contains a header with additional information about the signal. Amongst other, it contains information like the sample frequency, type of signal, unique identifiers and

signal length. We created a script per file format that creates the header and stores it along with the data in a file, readable for Noxturnal.

4.4.2 Analysis

Our data will be analysed by a sleep specialist with many years of experience in the field of analysing PGs. The PSG times are leading in our analysis. Our data is analysed between the start sleep time and wakening time as defined in the PSG. The overall score of our results will be compared with the PSG results. The PSG itself is analysed using the BrainRT+ (O.S.G. Bvba, Belgium) software package. In this analysis we only use the PG components of the PSG, i.e. we do not include an event that is scored as a result of an arousal in the EEG signal. Furthermore, we will compare each scored event in the PSG and our data set with another. Besides the scored results, we will compare the registered signals.

Pressure Signal

Our pressure sensor is connected to the same nasal cannula and should thus record the same data. Because we designed this sensor our self. First we will visually compare both signals over a large time, but also zoomed in on specific areas. This already gives a good impression about how alike both signals are. We will also plot the data in a Bland Altman plot. Furthermore we will compare the signal's lag over time.

Blood Oxygen Saturation

Both the PSG and our tool use gold standard equipment for the measurement of the blood oxygen saturation. Nevertheless, we have the signals perfectly aligned over the exact same time span. We have created a little script that is capable of detecting 3% desaturations. The script first finds all local peaks. Between those peaks, it searches for the minimum value. If the subtraction of peak minus dale value is 3 or higher, it marks those points and adds one to the counter. This method is applied on both our signal and the PSG signal. The PSG signal is recorded with 5Hz, while our signal is 3Hz. Both should be more than sufficient to correctly detect oxygen desaturations. We will also do a visual comparison, both over the full time set and zoomed in on a couple of minutes of data.

Respiratory Effort

We have described two methods we used to analyse our respiratory effort. These results will be visually analysed over a large and small dataset.

5 Methods

5.1 Protocol

5.1.1 Study Population

Patients were asked by one of two sleep specialised lung physicians for participation in this research. Those who agreed were contacted by the researcher for further information and were provided a patient information letter. The patients were given at least two days to inform themselves about the research and were asked on the day of the in-hospital PSG for participation. For this pilot study we will try to include at least 10 patients in the period November-December 2015.

Inclusion Criteria

- The patient must at least be 18 years of age;
- The patient must be forwarded by a sleep specialised lung physician for in-hospital PSG.

Exclusion Criteria

- There are no exclusion criteria.

5.1.2 Methods

Patients were seen by a sleep technician in the afternoon for connecting the EEG electrodes. During this process the patient was offered the opportunity to ask more questions regarding the study, and asked to participate in the study. The same day, the patient was seen by the sleep technician in the evening, to attach all PSG sensors. Once the patient was connected to all sensors of the PSG equipment, we attached our sensors as well.

The recording device, a laptop with power adapter, was positioned opposite of the patient on a table. The power socket used to power the laptop, did not interfere with the PSG equipment. The pressure sensor was connected using a T-junction to the original PSG nasal cannula. The patient was asked to sniff, to make sure

both the PSG system and ours was able to detect the disturbance in flow. A Nonin 4100 pulse-oximeter was attached around the left arm of the patient. The Nonin FlexiWrap sensor was taped around the index or ring finger, depending on which one was free and not used by the PSG system. Finally the accelerometer sensors were attached to the RIP bands around the patient. Two sensors were positioned on the mid-clavicular line on the thoracic RIP band and two on the same line on the abdominal RIP band as shown in figure 5.1. The sensors were attached using medical grade tape and all positioned in the same direction with the on off-switch facing the patient's head. Finally the sensors were checked in the GUI, to see if all sensors worked. Sensors that could not connect were replaced by other sensors. The build-in microphone was aimed towards ($\pm$ 3.5 meters away) the patient and the screen, wifi, audio volume were turned off to not disturb the patient during the measurement. All data was recorded on an encrypted hard disk. The next morning, the sensors were removed together with the PSG sensors.

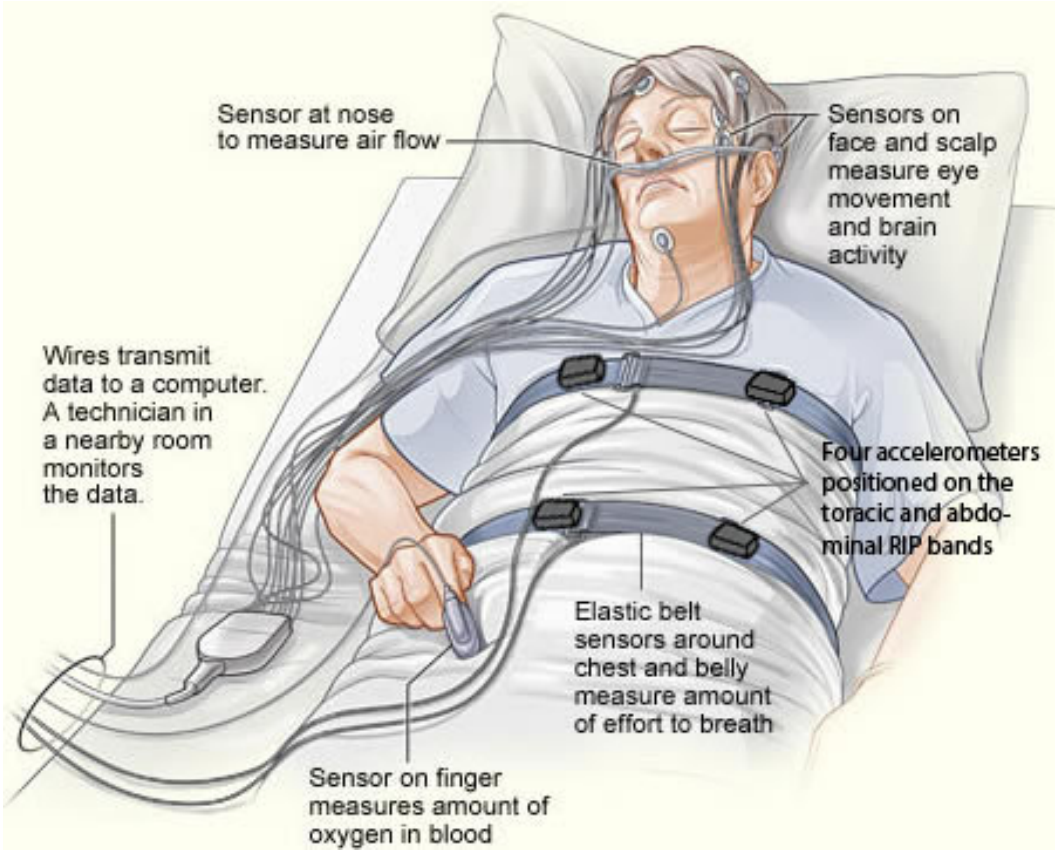


Figure 5.1: The four accelerometers are attached to the thoracic and abdominal bands using medical tape, after the patient is fully connected to the polysomnography equipment. The patient is also attached to an extra pulse oximeter and a pressure sensor is connected to the end of the already existing nasal cannula using a T-junction connector.

The medical ethics committee Twente approved this pilot study as a non-scientific medical research, but did unfortunately not approve the positioning of the head accelerometer without a full medical ethics review application.

5.2 Data Acquisition

Our data is acquired using BTLE sensors with a sample frequency of 10Hz, as explained in previous sections. The pulse oximeter uses Bluetooth and its data is sampled with 3Hz. All PSG sensors are wired and sampled with a frequency up to 250Hz. The PSG data set is exported as EDF file and loaded in Matlab for analysis.

5.2.1 Statistics

Statistical analysis is done using SPSS (Version 23.0. Armonk, NY: IBM Corp). Results are shown in the format of mean$\pm$standard deviation for normal distributed data. Furthermore we will use paired t-test for normally distributed paired data to look for significant mean differences, plot paired data in a correlation graph with the corresponding coefficient of determination (r^2) to understand the correlation. Finally we will look for consistent bias by plotting the data in a Bland Altman plot.

6 Results

6.1 Patients

A total of 10 patients were approached in the period of November-December 2015, and all patients agreed to participate in the study. In one registration, the saturation signal was lost after only two hours of registration, of which only half an hour of sleep data. This patient is excluded from our analysis. In one registration the recording ended prematurely. This was due to a mistake of the researcher: the laptop was not plugged in to the socket. That measurements lasted for slightly more than 5 hours, of which a part in the beginning when the patient was still awake. In all other registrations, the sensors started before the sleep start time and lasted till after the wakening time. The patient characteristics are displayed in table 6.1. The sleep start and stop time as defined in the PSG report, are used for our analysis. Data registered prior to the start of sleeping and after wakening of the patient, are discarded. Data is reported as mean±std.

Table 6.1: Patient characteristics.

pt. #	Gender	Age (year)	Weight (kg)	Length (m)	BMI (kg/m^2)	Sleep time (h)
1	F	31	92	179	28.7	8:40
2	M	55	96	180	29.6	4:31
3	M	38	86	172	29.1	8:29
4	M	39	76	184	22.4	8:29
5	M	19	93	185	27.2	9:28
6	M	43	102	190	28.3	8:16
7	M	43	98	178	30.9	8:26
8	F	55	98	171	33.5	9:31
9	F	40	92	180	28.4	8:08
	6M/3F	40.3±11.1	179.9±6.0	92.6±7.7	28.7±3.0	8:13±1:28

6.2 Polysomnography vs Our data

Our data is analysed using Noxturnal and analysed by a sleep specialist. We did however not use all of the recorded signals in this analysis. The sound volume signal hardly changed over the entire night and we assume that the microphone was not able to detect the snoring sounds properly due to the distance to the patient. The RIP band signals were also not used in the analysis because the pre-processing of these signals was not yet working properly at the time of analysis. We did use the

body position, flow signal, heart rate and saturation signal. The sound volume was available as well, but did not show a lot of variation.

The PSG results of our patients and the results of our data are shown in table 6.2. A quick look shows that the obtained results are rather similar. Only one patient (#9) has an AHI and ODI difference larger than 1 event/hour. Two patients are categorised in a group lower (no SAS in stead of light SAS) compared to the PSG (patient #7 and #9). The absolute difference in Apnoea-Hypopnoea events is 2.8±3.8 for all registrations, for the oxygen desaturation events this is 3.3±3 for all registrations.

Table 6.2: The results of the polysomnography and our research are listed per patient. The numbers are shown as index (Apnoea-Hypopnoea Index (AHI)/ Oxygen Desaturation Index (ODI)) which are compensated for the sleep time. We also show the total numbers of events during the registration.

	Apnoea Hypopnoea						Oxygen desaturation					
	AHI			Total events			ODI			Total events		
pt. #	PSG	Our	diff	PSG	Our	diff	PSG	Our	diff	PSG	Our	diff
1	0.6	0.6	0.0	5	5	0	0.5	0.6	-0.1	4	5	1
2	7.7	8.2	-0.5	35	37	-2	7.5	8.2	-0.7	34	37	3
3	1.4	1.3	0.1	12	11	1	1.4	1.3	0.1	12	11	1
4	2.7	2.0	0.7	23	17	6	2.5	2.0	0.5	21	17	4
5	1.0	0.7	0.3	9	7	2	0.8	0.7	0.1	8	7	1
6	11.0	10.9	0.1	91	90	1	11.1	10.9	0.2	92	90	2
7	5.3	4.6	0.7	45	39	6	5.3	4.6	0.7	45	39	6
8	2.0	1.9	0.1	19	18	1	2.0	1.8	0.2	19	17	2
9	5.7	4.4	1.3	46	36	10	5.7	4.4	1.3	46	36	10

We have compared each reported event in the PSG dataset, with the reported events in our dataset. We also have done this the other way around, by comparing each event in our analysed data with the events in the PSG dataset. Table 6.3 shows the results of this comparison. The first column shows the patient number. The second column shows how many events in the PSG and our dataset were a perfect match. This means that the same desaturation is scored at the same timestamp in both sets.

The 3^{rd}-6^{th} column show how many events are extra found in the PSG set, compared to our set. In this, the first column "2%" indicates how many events were correctly scored in the PSG set, but only had a 2% desaturation in our signal. The second column "False" shows how many events are scored, but should not have been in the PSG set. These events do show a 3%+ desaturation, but this is caused by a short peak (fluctuation) in the saturation signal. These event are scored by an automated mechanism, but should have been removed in the manual check. The third column "Mov." are the amount of events scored during or directly after a rotation of the patient. These events as well should have been removed in the manual check of the

registration. The last column "Not Our" are events that are correctly scored, but no trace of them can be found in our signals.

The 7^{th}-10^{th} column show how many events are found extra in our set, compared to the PSG set. In this the "2%" column is the same as for PSG, though now we found a 3%+ desaturation in our signals, but only 2% in the PSG signals. The second column "2 = 1" indicates that two separate events in our data are marked as just one long event in the PSG set. The PSG set does however show two separate events. To correctly compare the sets, the amount of events in the PSG signal should be increased by this number. The "In PSG" column indicates how many events are scored in our set and are clearly visible in the PSG set, but are not scored. To correctly compare the data, the amount of events in the PSG signal should be increased by this number. Lastly, the column "Sig Mis." indicates that an event found in our set, was at a timestamp at which no saturation data was available in the PSG set. Looking at the results, it is not unlikely that the signal would have been in the PSG data as well. It is however impossible to say for sure, thus we will exclude these events from our analysis.

Table 6.3: All events listed and categorised to a group that explains how it differs from the other signal.

		Oxygen desaturation - differences in events							
		Extra events in PSG				Extra events in our Data			
pt. #	Exact match	2%	False	Mov.	Not Our	2%	2 = 1	In PSG	Sig Mis.
1	4 (80.0%)	-	-	-	-	1	-	-	-
2	30 (81.1%)	1	1	-	2	3	4	-	-
3	9 (75.0%)	-	2	1	-	1	-	1	-
4	12 (57.1%)	6	-	3	-	-	-	2	3
5	7 (87.5%)	-	1	-	-	-	-	-	-
6	82 (89.1%)	9	-	1	-	1	1	5	1
7	37 (76.2%)	8	-	-	-	-	-	2	-
8	14 (73.7%)	1	2	1	1	-	-	3	-
9	35 (76.1%)	7	4	-	-	-	-	1	-
sum	230 (80.7%)	32	10	6	3	6	5	14	4

2%: a desaturation of 3%+ was found in one signal, but showed only 2% desaturation in the other signal
False: event should not have been scored as an event
Mov.: event is scored after patient turns around; should not have been scored
Not Our: correctly scored event, no trace of it in our dataset
2 = 1: 2 desaturations in our dataset, are scored as one long desaturation in the PSG set
In PSG: correctly scored in our dataset but not in PSG; should have been scored
Sig Mis.: the PSG signal is missing at the time of the event

The corrected numbers are displayed in table 6.4. On average the ODI difference between the PSG signal and our data has become lower. This would mean a drop in the AHI values as well. The maximum difference is 1 ODI. The amount of matching events is 87.1% after the correction. The absolute difference in oxygen desaturation events is 3.5 ± 3.3.

Table 6.4: The results of the polysomnography (corrected for errors) and our results listed per patient. The numbers are shown as index (Apnoea-Hypopnoea Index (AHI)/ Oxygen Desaturation Index (ODI)) which are compensated for the registration time. We also show the total numbers of events during the registration and the total amount of matching events.

	Corrected Oxygen desaturation values						
	ODI			Total events			
pt. #	PSG	Our	diff	PSG	Our	diff	Matching Events
1	0.5	0.6	-0.1	4	5	1	4 / 5 (80.0%)
2	8.2	8.2	0	37	37	0	34 / 37 (91.9%)
3	1.2	1.3	0.1	10	11	1	10 / 11 (90.9%)
4	2.4	1.7	0.7	20	14	6	14 / 20 (70.0%)
5	0.7	0.7	0	7	7	0	7 / 7 (100%)
6	11.7	10.9	0.8	97	90	7	88 / 97 (90.7%)
7	5.6	4.6	1.0	47	39	8	39 / 47 (83.0%)
8	2.0	1.8	0.2	19	17	2	17 / 19 (89.5%)
9	5.3	4.4	0.9	43	36	7	36 / 43 (83.7%)

6.3 Pressure Signal

Figure 6.1 shows twice 1 minute of pressure signal at two different start times. The blue line shows the pressure registered by the PSG system, the red line shows the pressure registered by our sensor. The signals are scaled to the same height. Signals of other patients show similar visual correlation.

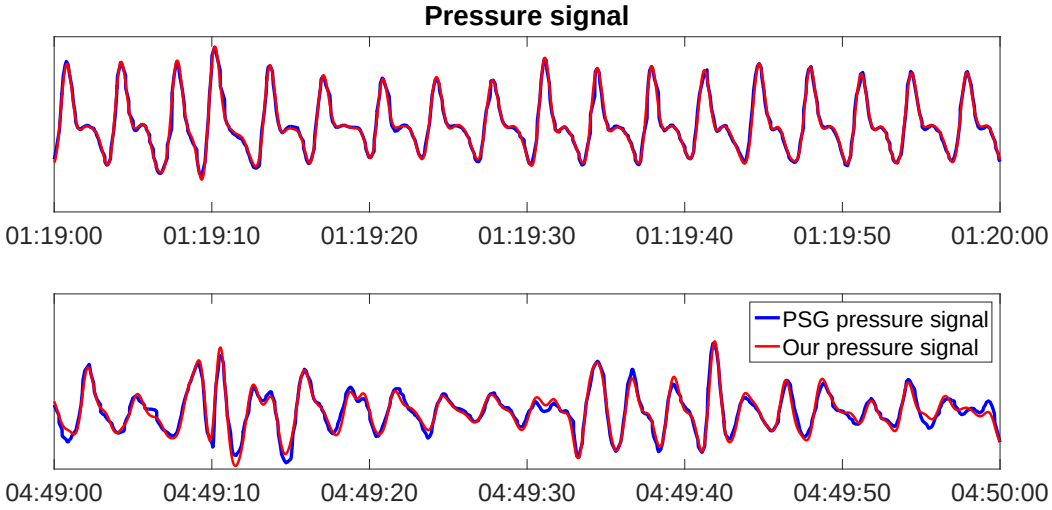


Figure 6.1: Pressure data of the PSG system (blue) and our sensor (red). Two times one minute of data is plotted at two random times in the signal.

Figure 6.2 shows the correlation of a 10 minute sample of the pressure signal as well as the Bland Altman plot. A correlation coefficient of 0.9451 is found, which

indicates that a strong positive linear relationship exists between both signals. The Bland Altman plot shows the points both above and under the zero line. This suggests that there is no consistent bias between the signals. The p-value of a paired t-test is shown as well. The result, p = 0.0016, indicates that there is no significant mean difference between the two paired samples. This all suggests that the signals are the same.

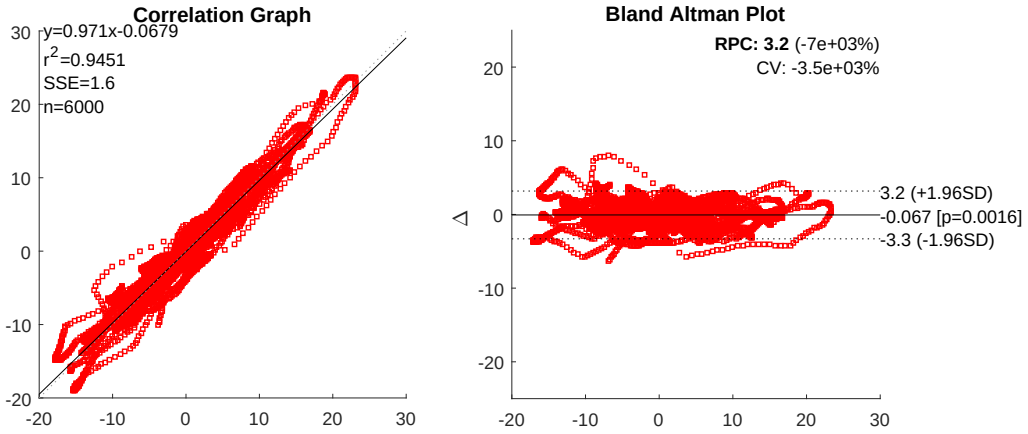


Figure 6.2: On the left side the Correlation graph with the calculated coefficient of determination (r^2). On the right side the Bland Altman plot of 600 seconds (10Hz) of PSG vs Our pressure signal.

We have calculated the time offset of the pressure signal over time, using data with a length of 60 seconds. Figure 6.3 shows the offset over time. All signals have a more decreasing offset. If we assume that the PSG pressure signal is indeed sampled with exactly 250Hz, then this means that our pressure signal is missing more points over time. The delay is very small, less than 0.25 second in 8 hours of data. This is an accepted error. Patient #2 stops around 3:30. This is due to the empty laptop battery. Some patients show a hiccup at the end: this happens because the patient stops breathing through the nasal cannula and both the PSG and our pressure sensor: correlating two noise signals results in more noise.

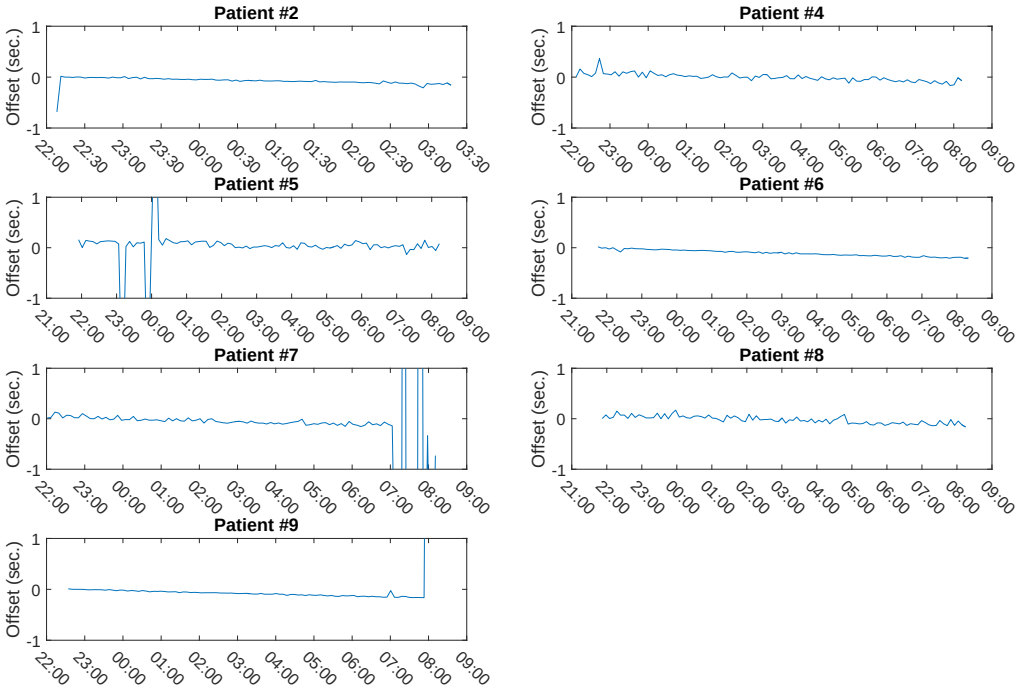


Figure 6.3: The offset in seconds, calculated at 100 evenly divided points during the night. Note that the laptop turned down early for Patient #2.

6.4 Saturation Signal

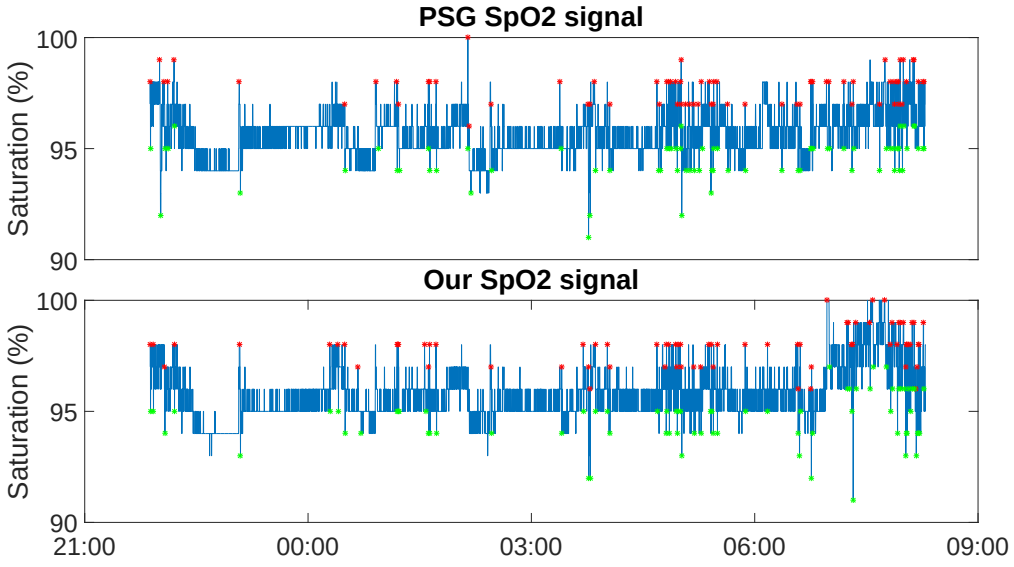


Figure 6.4: Saturation signal of the PSG system and our sensor. An automated process has marked all maximum and minimum values if there was a 3% desaturation.

The saturation signals are both registered with medical approved equipment. We would thus expect some sort of correlation. In table 6.3 we see that quite some

amount of desaturations in one signal, which are not seen in the other signal. This could indicate a difference in the sensitivity of the sensors but can just as well be the cause of two different sample rates (5Hz for the PSG system, 3Hz for our system). When we plot the signals over time, as shown in figure 6.4, there are similarities visible between the signals. If we zoom in on a couple of minutes, as seen in figure 6.5, small differences can be seen. An automated script found an ODI of 6.8 for the PSG signal, and an ODI of 7.0 for our signal. Note that this process is far from optimized. This is not as expected, since the PSG system has more desaturations which were not seen in our signal.

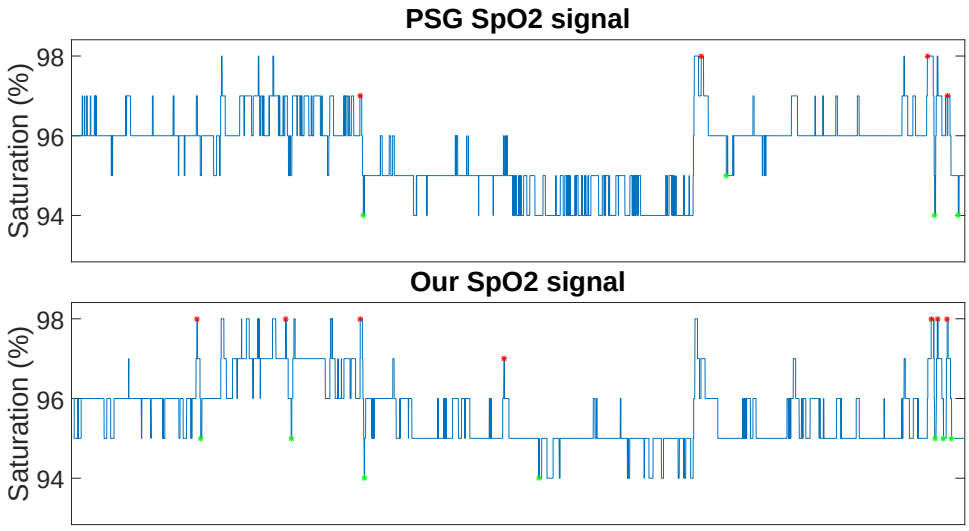


Figure 6.5: Saturation signal of the PSG system and our sensor. When zoomed in, small differences can be seen between both signals.

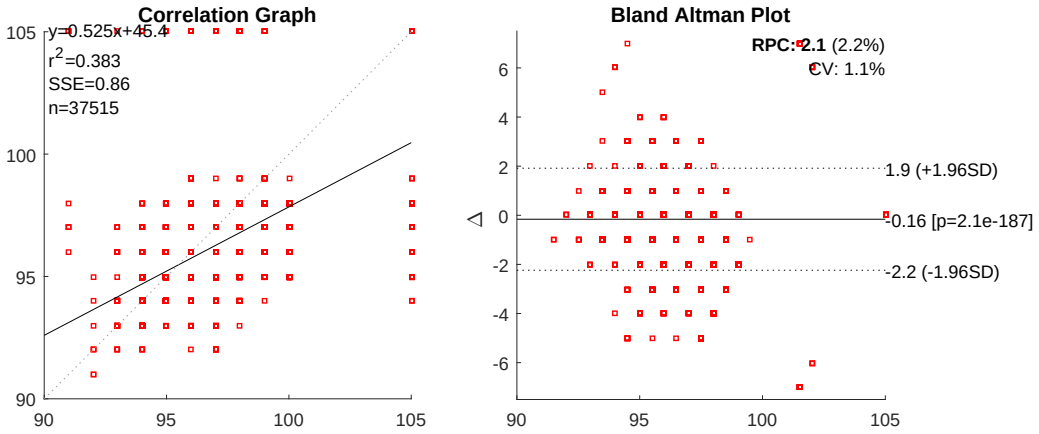


Figure 6.6: On the left side the Correlation graph with the calculated coefficient of determination (r^2). On the right side the Bland Altman plot of the entire SpO2 signal with a sample frequency of 1 Hz.

The correlation graph in figure 6.6 shows that there is a poor linear correlation between the SpO_2 signal measured with the PSG system, and the SpO_2 signal measured with the Nonin 4100. The Bland Altman plot indicates that there is no significant mean difference between the signals ($p \ll 0.01$) and that no consistent bias is present. The correlation graph does show a very wide plot, meaning that a value of 100% in one signal, could correlate to 99%, 98%, 97%, 96% or even 95%.

6.5 Respiratory Effort

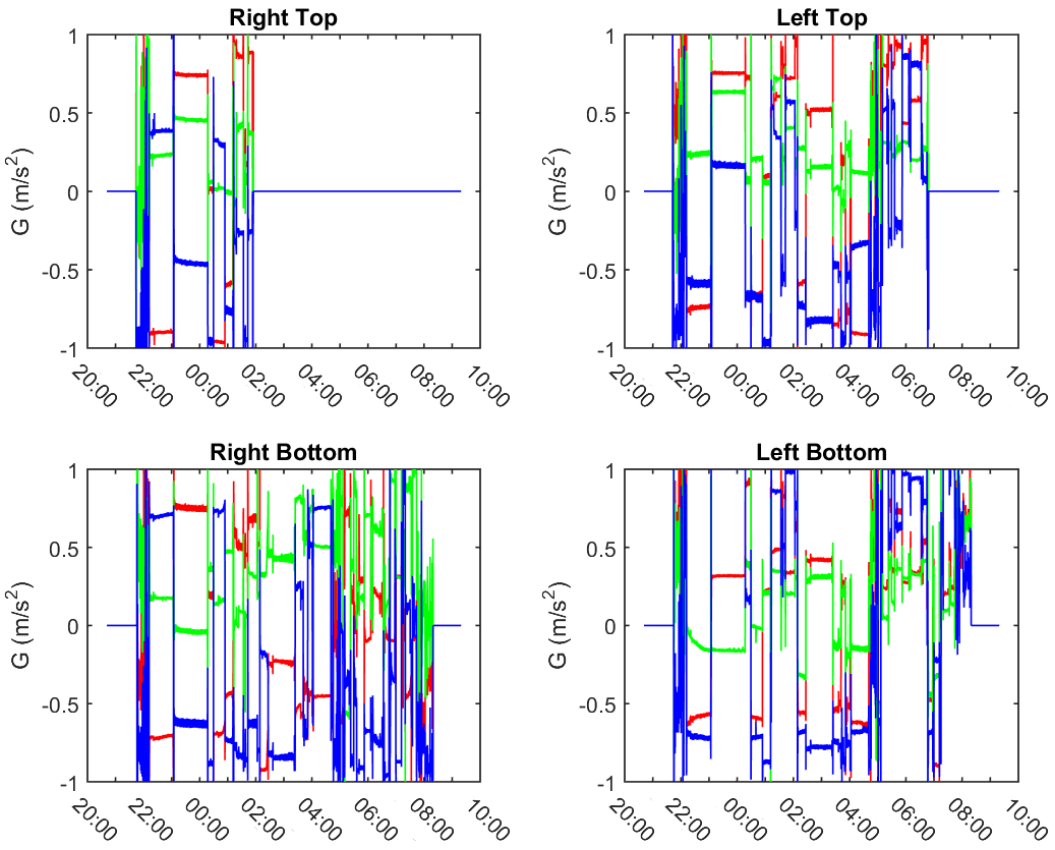


Figure 6.7: Gravitational acceleration (m/s^2) signal of patient #6 of the entire night. X-axis (Red), Y-axis (Green), Z-axis (Blue). The signal of the Right top sensor (thoracic sensor) stops working after a couple of hours into the registration.

Figure 6.7 shows the accelerometer data of all four sensors over the entire night. A loss in signal is clearly visible in the right top sensor. Despite the loss of data, enough data is available to calculate the respiratory effort using the left sensor. Figure 6.8 shows the respiratory effort calculated using the abdominal sensors (left bottom and right bottom). In this figure, the red line shows the PSG RIP band signal and the blue line is the calculated respiratory effort based on two of our accelerometers. In this signal we can see periods in which the blue line has a large amplitude e.g. around 23:30, and areas in which the blue line has a small amplitude e.g. around 1:00. The strength of the signal depends on the sleeping position of the patient. A

stronger amplitude is found for supine position than left or right. In supine position both sensors can move freely, while one might get blocked in left or right position.

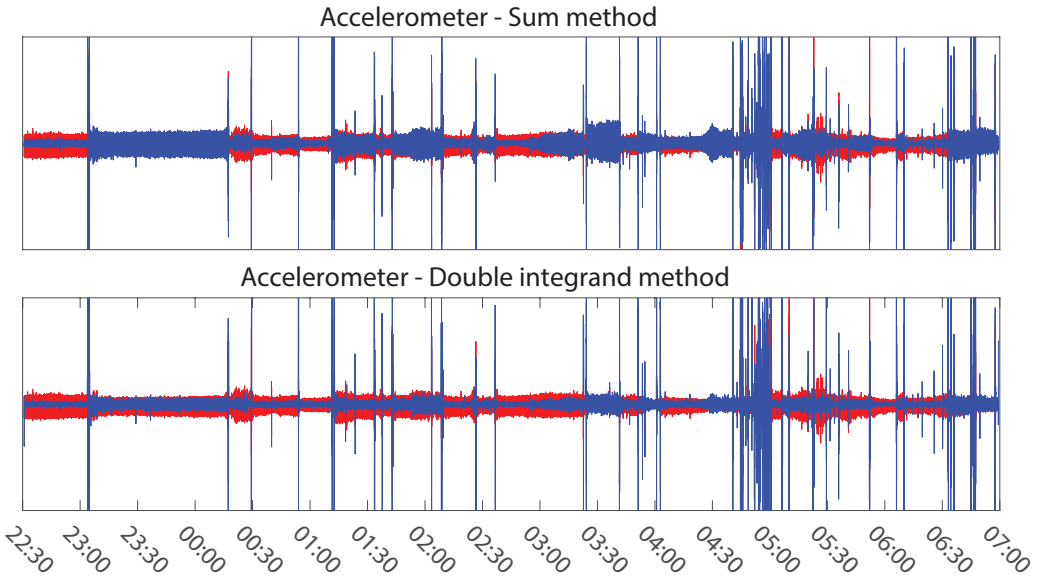


Figure 6.8: The calculated abdominal respiratory effort signal over an entire night. The red line shows the PSG RIP band values, the blue line is the calculated respiratory effort signal based on our accelerometers.

If we have a closer look as the data, by visualising approximately 2 minutes of signal, we can see the correlation between our calculated signal and the RIP band signal. This is shown in figure 6.9. As was clearly seen in figure 6.8, the signal around 23:20 has a high amplitude. The signal around 1:00 however has a low amplitude. In figure 6.10 we show the area around 1:00. The signal around 1:00 is scaled so, that it has the same amplitude as the RIP band. Just like around 23:20, both methods follow the signal rather good, whereby the sum method seems to follow the RIP band signal slightly better. We have not scored central apnoeas in our data, since the respiratory effort data was missing during the analysis. We have however compared the data of some of the central apnoeas scored in the PSG data, with our respiratory effort and pressure data. Based on our data, the events (now scored as normal apnoea) would have been scored as central apnoea.

Figure 6.11 shows the correlation graph as well as the Bland Altman plot of the sum method. A correlation coefficient of 0.8381 is found, which indicates that a strong positive linear relationship exists between the signals. The p-value indicates that there is a significant mean difference between both signals. The Bland Altman plot shows the signal above and under the zero line which would indicate that there is no consistent bias.

Figure 6.12 shows the correlation graph and the Bland Altman plot of the double integer method. A correlation coefficient of 0.6846 is found, which indicates that

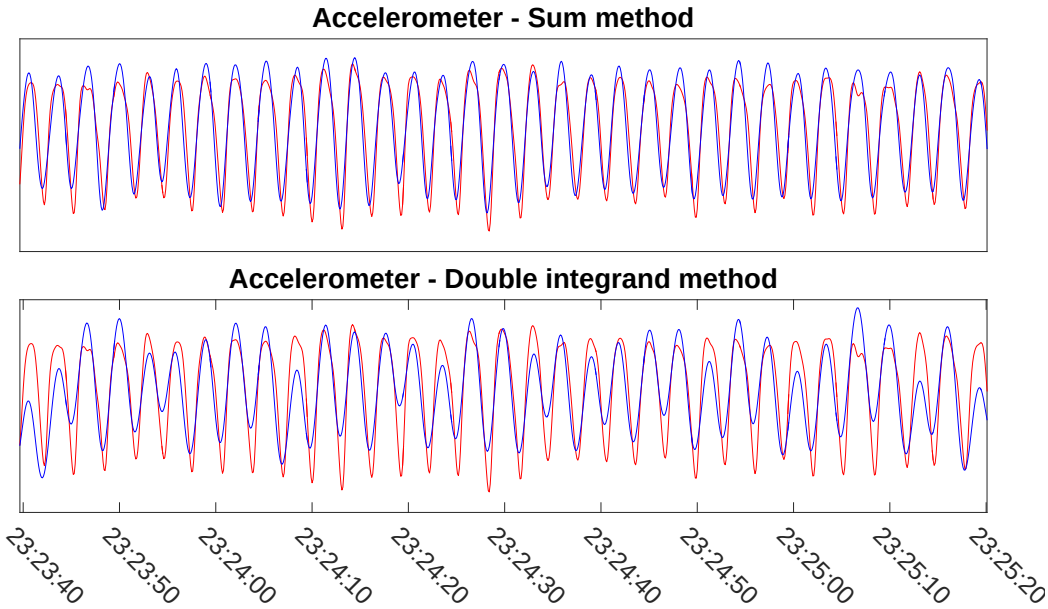


Figure 6.9: A close-up of the pressure signal of figure 6.8. The signals are scaled to have the same amplitude. During this, the patient lays on the left side.

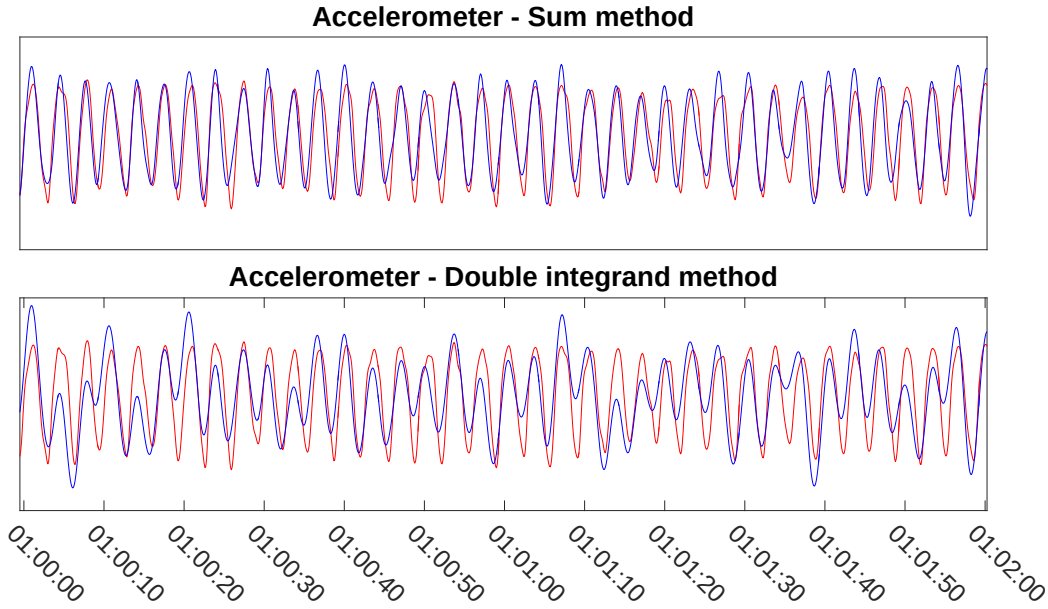


Figure 6.10: A close-up of the pressure signal of figure 6.8. The signals are scaled to have the same amplitude. During this, the patient lays in supine position

a moderate positive linear relationship exists between the signals. The p-value is 0.00015 and indicates that there is no significant mean difference between both signals. The Bland Altman plot shows that there is no indication for a consistent bias.

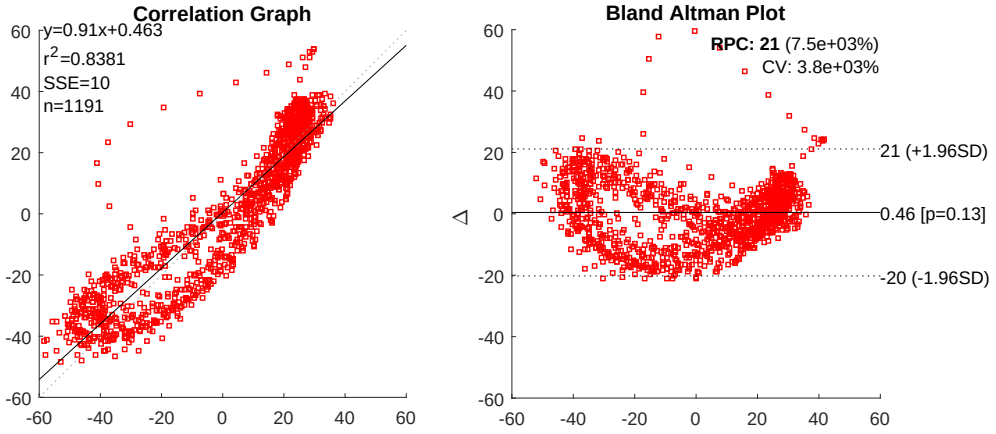


Figure 6.11: On the left side the Correlation graph with the calculated coefficient of determination (r^2). On the right side the Bland Altman plot of 119 seconds (10Hz) of RIP band vs Our "sum method" RIP band.

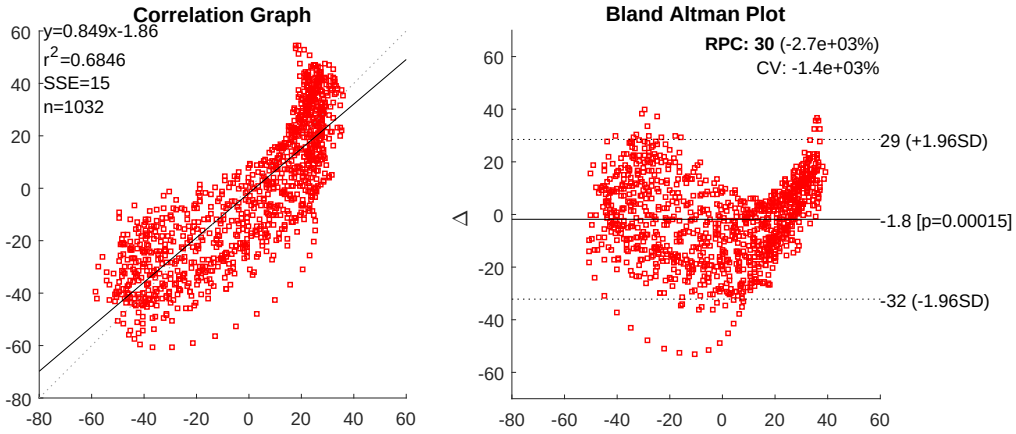


Figure 6.12: On the left side the Correlation graph with the calculated coefficient of determination (r^2). On the right side the Bland Altman plot of 103 seconds (10Hz) of RIP band vs Our "sum method" RIP band.

6.6 Body Position

Although we have calculated the body position our self, we have not done a one on one comparison of our calculated values with body position found in the PSG data. The Noxturnal software used our pre-processed accelerometer data to do the body position calculate as well. This was visually compared with the PSG body position and showed a very good match. For each rotation found in the PSG data, e.g. going from supine to left, a same rotation was found in the Noxturnal report within 10 seconds of the PSG time.

7 Discussion

The aim of our study was to determine if it is possible to reliably determine sleep apnoea syndrome using cheap electronics, a smart phone and existing analysing software. For this we have explored which signals were mandatory to record: the flow signal, respiratory effort, saturation and heart rate. We have build a sensor and used existing equipment to record these essential signals; first in laboratory setting and later in a prospective clinical study.

After start up problems with patient #1-4, we have been able to record all these signals during the entire night. We have also successfully transformed this raw data into an understandable format for a well known sleep apnoea analysing program. Finally a sleep specialist was able to analyse our data and score the apnoeas and hypopnoeas with it. With that, we have successfully completed our objective to make a system that is applicable in clinical setting and which automatically provides clinical outputs.

When we compare our system with the gold standard, the PSG, we find an agreement of 87.1% of the registered events. Our pressure signal shows a very good linear correlation (r^2=0.9451) with the PSG's pressure signal. Our respiratory effort signal has a high linear correlation (r^2=0.8381) with the PSG's RIP band and our registered ODI has a maximum deviation of 1 event/hour. Our analysis would have scored 7 out of 9 patients in the right sleep apnoea category. 2 patients were scored in a category too low, because our score was respectively 0.9 and 1.0 desaturation/hour lower than the PSG. This brought our findings at an ODI of 4.4 and 4.6, while the border between the two groups lays at 5. Overall, our patients scored lower than the PSG. The most important factor for this, is the saturation signal. The PSG system uses a 5Hz wired sensor while we used a 3Hz wireless sensor. The correlation graph shows a wide plot, meaning that a value in the PSG signal can correlate to many values in our signal. In the worst case it can correlate to a value that is $\pm$5%. If we look at the scored desaturations, they show a good resemblance. This means that even though the signals might differ at some points in value, they do register the desaturations correctly. We expect that the problem of scoring the patient in the wrong group, will remain if patients with higher AHI's are measured. Our system will most likely score them in a group lower if they are near the border of ODI=15 or ODI=30. Even though the label can be different for these patients, the treatment option will most likely be the same.

Our objective was to make a system that is applicable in clinical setting. The sensors were each time connected by the researcher. We have thus not tested if any of the PSG technicians were able to connect our system. However, it seems that it is less work than connecting a patient to the PSG sensors, or even the PG sensors. The system was designed to be user friendly and the GUI provides feedback to the user if anything goes wrong. We have currently connected the sensors to the RIP bands, but they can be connected via a plaster to the skin with the same ease. Doing so could even provide better signals of the thoracic and abdominal movement. Our data is automatically pre-processed and stored in a different file format, so that it can be used in Noxturnal. This process is already automatic and requires no extra knowledge: just select the patient and press run. The data itself was analysed by a sleep specialist with many years of experience in the field of scoring sleep apnoea. This latter process is already part of a clinical process and also does not require extra knowledge or training.

The PSG data was analysed by a different sleep specialist. Even more experienced than the one who analysed our data. He did however not analyse the PSG as thorough as we did analyse our data. The PSG data did contain mistakes which we have been able to correct for. It would however have been better to use the same sleep specialist for analysing both datasets.

7.1 Patients

A total of 10 patients were approached in the period of November-December 2015, all patients agreed to participate in the study. None of the patients stopped prematurely with the study, and none of them was extra discomforted by the four extra sensors attached during the PSG registration. Even though the patients were not discomforted by the sensors, they did have a lot of equipment on that did bother them during the registration. Due to the great discomfort the PSG registration already brings along, the extra discomfort of the sensors might not have been noticed. It is thus unclear if the sensors would discomfort in the situation of home screening. In this study we did not have the time to reduce the size of the sensors even more, though the used battery was able to power the sensor for more than one night: which means that a smaller battery with a lower capacity can be used to reduce the overall sensor size. A smaller sensor would likely interfere even less with the patients sleep comfort, resulting in a better and more natural sleep registration.

We initially wanted to include 30 patients. Due to administrative requirements and the move of the hospital, we have been able to include 10 patients only. Table 6.1 shows the patient characteristics of which the age and BMI are normally distributed.

A larger amount of patients could have resulted in a wider variety of AHI, allowing us to test our system with higher AHI values as well. In the period November-December 2015 all patients sent by a pulmonologist were recruited, but patients sent for a PSG by a neurologist were not. This patient group mostly have neurological reasons and would thus less likely have medium or severe sleep apnoea syndrome. We were unable to continue our inclusion because the new PSG room was not ready in January 2016.

7.2 Sensors

After some start up problems, our sensors have done a rather good job. Some sensors, like in figure 6.7, stopped working in the middle of the night. This does limit the data registration to one sensor, which is enough to calculate the respiratory effort. Some sensors (not visualised) lost connection for a few minutes before reconnecting. The software is interpreted on the chip itself. It might increase the stability of the sensor if the software is pre-compiled and installed on the chip. This process is time consuming while debugging, but now that we have proven the system to work, it might add to the trustworthiness of the system.

The Nonin 4100 has failed in the fourth registration. After only a few hours of registration it stopped transmitting data. It is unclear what caused this problem or why the sensor did not reconnect. We can only hypothesize that the batteries did not work properly, or that the sensor cable got loose.

7.3 Signals

7.3.1 Pressure

The pressure signal shows a very good correlation with the PSG signal. We only used a 10Hz sample frequency, while the PSG system used 250Hz. Figure 6.3 shows the time offset of the pressure signal over time, which decreases. There are two possible explanations for this: the PSG signal has a higher sample frequency than 250 Hz, or we miss some data. The second option is most likely since the PSG system uses a wired pressure sensor. The total delay over a 8 hour registration is negligible, yet it is present. This problem might be fixed when he sensor itself sends the timestamp of a registration. The problem of this solution is that BTLE might not have sufficient bandwidth to send the full timestamp with each sample. Another solution might be to increase the stability of the sensor, by programming it with pre-compiled code,

in stead of a BASIC script. This method however requires expensive devices and compilers to be bought.

The pressure signal has an offset in the beginning. Using cross-correlation, the signals can easily be synchronised over time. The method used, calculates the interpolant function of both signals, and produces a 4.000 Hz sample of 60 seconds in length. The method used, first tries to remove all noise by removing all data below 0.1 Hz and above 1Hz. Normal breathing frequencies are between 0.1 and 0.4Hz. Below that the patient is most likely holding breath and above 0.33Hz the patient has tachypnea. Technically the filter could be closer to the upper limit, by taking a cut-off value of 0.5 Hz. The FFT of the signal did however not show much signal in this band, indicating that it only contains little noise and could not harm the outcome. Since the noise is filtered out of the signal, prior to it is being converted to an interpolant function, it does not exist in the output either. The interpolant function uses the 'linear' interpolant. This is the default method used by the Matlab 'griddedInterpolant' function. Other options exist for calculating the interpolant function, of which the 'spline' could be a good alternative. The signal is round, meaning that the spline interpolant would likely to better predict the values in between points. This method however has a downsize, it uses more memory and computation time to operate. The difference in output signal, between linear and spline is very little and not worth the extra time waiting. The calculated lag in the pressure signal using linear and spline would be small enough to not have a clinical value.

The interpolant function is used to create a 4.000Hz signal for both the PSG pressure readings and our pressure readings. Each sample is thus 0.25ms apart. The cross-correlation function 'xcorr', can calculate the amount of lags required to obtain the maximum correlation. The amount of lags, multiplied by 2.500 is the amount of lag in Ticks (2.500 Ticks = 0.25ms). This value is used to correct the offset of all signals recorded by our sensors, to synchronise them in time with the PSG measurements. The accuracy of the synchronised signals is thus 0.25ms. Clinically there is no reason at all to do this, and it is questionable if the value actually can be calculated with this precision due to a lack of information in our 10Hz signal. The signal however is a constant sinus which mean that the signal to predict does not include difficult components. For our research, there are two reasons to synchronise the signals: To use the PSG pressure signal for patient #1 and #3 when analysing the results, and to be able to compare the signals with another. If our tool is used in clinical setting, there is no need to synchronise the signals since we don't need to compare it with e.g. the PSG signals.

7.3.2 Accelerometers

Different methods have been tested, to mimic the RIP band signals.

- The Sum method
- Double integrand method

Figure 6.8 shows the respiratory effort calculated using these two methods. Overall, the sum method has a higher linear correlation with the RIP band signal. Visually, the sum method also looks more like the RIP band signal. But the units of this signal are m/s^2, which is a weird unit for a respiratory effort signal. We would expect a length or volume as unit. The displacement in position is obtained using the double integrand method. Though this required two unknown constants 'C' to know the actual position of the sensor.

The Sum method did show a good correlation and provided the right information to make the signal of clinical importance. In clinical setting, the RIP band signals are used to determine activity of the thorax and abdomen prior to a desaturation. For this clinical application, our data is useful. It has been able to differentiate between normal apnoeas and central apnoeas. The accelerometer data was also able to be used to calculate the body position, meaning that we can differentiate between supine AHI and non-supine AHI required for pOSAS.

In literature, good results are found using accelerometers as well. These studies however use high sample frequencies. Morillo et al. used a sample frequency of 8192Hz and used the accelerometer signal for snoring detection as well [29].

Rendón et al. researched multiple positions on the neck and chest for the detection of snoring, breathing and heart rate [41]. The position we chose, approximately 10a and 10b in their research, show good signal powers for breathing detection and hear rate but weak signal power for snoring registration. For our purpose, we have chosen the right position.

Pierleoni et al. used the BioHarness^{TM}, which contains an accelerometer and pressure sensor (respectively 18Hz and 50Hz) using a Bluetooth 2.1 connection [38]. They designed a method for the automated detection of apnoeas, but have not validated this technique in a clinical trail against PSG data.

8 Conclusion

We have created a system that is applicable in clinical setting. The analysis is already done by a sleep specialist and applying the sensors is easier than connecting a patient to the PSG sensors. We have learned what signals are mandatory to obtain, in order to diagnose sleep apnoea and its specific type. These signals are the respiratory effort, nasal flow, blood saturation meter and body position. The heart rate is obtained with the blood saturation meter but is not a mandatory signal for diagnosing sleep apnoea. We have validated our system with the current gold standard, the PSG. 87.1% of the scored events agreed with the PG events scored using the PSG data. Our pressure sensor shows a high linear correlation (r^2=0.9451) with the PSG's pressure sensor. Our accelerometer based respiratory effort signal also shows a good linear correlation with the RIP band signal of the PSG (r^2=0.8381). The saturation signal does not correlate well and many scored Oxygen desaturation events in the PSG were not visible in our signal. The ODI and AHI found using our sensors were within 1 event/hour of the PSG results. With this study we have shown that it is possible to reliably determine sleep apnoea syndrome by using cheap electronics, a laptop and automatic processing techniques.

9 Recommendations

In this research we have shown that our proof of concept works. We recommend to further develop the sensor to an even smaller, more stable system. This can be achieved by making a separate sensor for the pressure signal and accelerometer, which are now both on one sensor. Using a smaller battery could significantly reduce the sensor's size. To improve the data receiving, the sensor should send an increasing unique id, or timestamp with each registration. This will ensure that the data recording order remains available. Once a new sensor is developed, a new clinical trail can be executed. We recommend to attach the sensor directly to the body, instead of attaching it to the RIP bands. We recommend to test the new sensor first with PSG patients, in a controlled environment. We also recommend to further analyse the positioning of the sensors. If the sensors work stable, one on the thorax and one on the abdomen might suffice. The software can be further developed and eventually rewritten for iPhone, android or windows phone. The microphone should be positioned close the patient, 3.5 meter is too far away.

Bibliography

[1] Mamoun Al-mardini. „A Framework For Screening And Classifying Obstructive Sleep Apnea Using Smartphones“. PhD thesis. American University of Sharjah, 2013, pp. 1–110 (cit. on pp. 8, 12).

[2] Shamma Alqassim, Madhumeta Ganesh, Shaheen Khoja, et al. „Sleep apnea monitoring using mobile phones“. In: *2012 IEEE 14th International Conference on e-Health Networking, Applications and Services (Healthcom)*. IEEE, 2012, pp. 443–446 (cit. on p. 8).

[3] Matej Andrejasic and Igor Poberaj. „Mems accelerometers.“ PhD thesis. University of Ljubljana, 2008, pp. 1–17 (cit. on p. 12).

[4] David J Bruinvels, Jos M Rooijackers, Erik Stigter, and Daniëlle N Visser. *Richtlijn: obstructieve slaapapneusyndroom en werk*. Utrecht, 2012 (cit. on p. 4).

[5] Christie L. Bucklin, Manohar Das, and Sam L. Luo. „An inexpensive accelerometer-based sleep-apnea screening technique.“ In: *Proceedings of the IEEE 2010 National Aerospace & Electronics Conference*. IEEE, 2010, pp. 396–399 (cit. on pp. 8, 12).

[6] R. D. Cartwright. „Effect of sleep position on sleep apnea severity.“ In: *Sleep* 7.2 (1984), pp. 110–4 (cit. on p. 3).

[7] Centraal Bureau voor de Statistiek. *Gezondheid, leefstijl, zorggebruik en -aanbod, doodsoorzaken; vanaf 1900.* 2016 (cit. on p. 5).

[8] P. de Chazal, Conor Heneghan, Elaine Sheridan, et al. „Automated processing of the single-lead electrocardiogram for the detection of obstructive sleep apnoea.“ In: *IEEE Transactions on Biomedical Engineering* 50.6 (2003), pp. 686–696 (cit. on p. 12).

[9] Zhenyu Chen, Mu Lin, Fanglin Chen, et al. „Unobtrusive sleep monitoring using smartphones.“ In: *Proceedings of the ICTs for improving Patients Rehabilitation Research Techniques*. New York, New York, USA: IEEE, 2013, pp. 1–14. arXiv: 1407.5910 (cit. on pp. 8, 9).

[10] J. Dall'Ava-Santucci and A Armanganidis. „Respiratory inductive plethysmography.“ In: ed. by Salvador Benito and Alvar Net. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991. Chap. Respirator, pp. 121–142 (cit. on p. 12).

[11] R J Davies, N J Ali, and J R Stradling. „Neck circumference and other clinical features in the diagnosis of the obstructive sleep apnoea syndrome.“ In: *Thorax* 47.2 (1992), pp. 101–5 (cit. on p. 4).

[12] Parastoo Kh Dehkordi, Marcin Marzencki, Kouhyar Tavakolian, Marta Kaminska, and Bozena Kaminska. „Validation of respiratory signal derived from suprasternal notch acceleration for sleep apnea detection.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2011 (2011), pp. 3824–7 (cit. on p. 8).

[13] Danny J. Eckert, Amy S. Jordan, Pankaj Merchia, and Atul Malhotra. „Central sleep apnea: Pathophysiology and treatment.“ In: *Chest* 131.2 (2007), pp. 595–607. arXiv: NIHMS150003 (cit. on p. 3).

[14] Michiel M Eijsvogel, Sytske Wiegersma, Winfried Randerath, et al. „Obstructive Sleep Apnea Syndrome in Company Workers: Development of a Two-Step Screening Strategy with a New Questionnaire.“ In: *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine* 12.4 (2016), pp. 555–64 (cit. on p. 7).

[15] Michiel M. Eijsvogel, Rinse Ubbink, Janita Dekker, et al. „Sleep position trainer versus tennis ball technique in positional obstructive sleep apnea syndrome.“ In: *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine* 11.2 (2015), pp. 139–47 (cit. on p. 6).

[16] Maxine Hardinge. „Obstructive sleep apnoea.“ In: *Medicine* 36.5 (2008), pp. 237–241 (cit. on pp. 3, 4).

[17] S. Javaheri. „Central sleep apnea.“ In: *Clinics in chest medicine* 31.2 (2010), pp. 235–48 (cit. on p. 3).

[18] Shahrokh Javaheri. „Sleep disorders in systolic heart failure: a prospective study of 100 male patients. The final report.“ In: *International journal of cardiology* 106.1 (2006), pp. 21–8 (cit. on p. 3).

[19] Christoph Kalkbrenner, Philipp Stark, Guy Kouemou, Maria-Elena Algorri, and Rainer Brucher. „Sleep monitoring using body sounds and motion tracking.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2014 (2014), pp. 6941–4 (cit. on pp. 8, 9, 12).

[20] Ellen R van Kesteren, J Peter van Maanen, Anthony A J Hilgevoord, D Martin Laman, and Nico de Vries. „Quantitative effects of trunk and head position on the apnea hypopnea index in obstructive sleep apnea.“ In: *Sleep* 34.8 (2011), pp. 1075–81 (cit. on p. 14).

[21] Ken M Kunisaki, Oksana A Bohn, Erin E Wetherbee, and Thomas S Rector. „High-resolution wrist-worn overnight oximetry has high positive predictive value for obstructive sleep apnea in a sleep study referral population.“ In: *Sleep & breathing* 20.2 (2016), pp. 583–7 (cit. on p. 11).

[22] R. Loddenkemper, G.J. Gibson, Y. Sibille, and B. Lundbäck. „The European Lung - white book“. In: *Principles and Practice of Sleep Medicine*. erswhitebook2013, 2013, pp. 270–281 (cit. on p. 3).

[23] Long Alliantie Nederland. *Longziekten - feiten en cijfers 2013*. 2013, p. 139 (cit. on p. 6).

[24] M. Jeffery Mador, Thomas J. Kufel, Ulysses J. Magalang, et al. „Prevalence of positional sleep apnea in patients undergoing polysomnography.“ In: *Chest* 128.4 (2005), pp. 2130–2137 (cit. on p. 3).

[25]Atul Malhotra and David P White. „Obstructive sleep apnoea.“ In: *Lancet (London, England)* 360.9328 (2002), pp. 237–45 (cit. on pp. 3, 7).

[26]Seyed Mohammad and Reza Haji. „Design of the wireless devices to record body position for sleep apnea.“ In: *9th Symposium on Advances in Science & Technology (9thSASTech 2014), Mashad, Iran*. 9th SASTech 2014. 2014, pp. 29–33 (cit. on pp. 8, 12).

[27]Farouk Mookadam, Andrew D Calvin, and Virend K Somers. „Prevalence and management of central sleep apnea in heart failure patients.“ In: *Current heart failure reports* 5.4 (2008), pp. 233–7 (cit. on p. 3).

[28]Michael Morgenstern, Janice Wang, Norman Beatty, et al. „Obstructive sleep apnea: an unexpected cause of insulin resistance and diabetes.“ In: *Endocrinology and metabolism clinics of North America* 43.1 (2014), pp. 187–204 (cit. on p. 3).

[29]D Sánchez Morillo, J L Rojas Ojeda, L F Crespo Foix, D Barbosa Rendón, and A León Leon. „Monitoring and analysis of cardio respiratory and snoring signals by using an accelerometer.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2007 (2007), pp. 3942–5 (cit. on pp. 8, 51).

[30]Daniel Sánchez Morillo, Juan Luis Rojas Ojeda, Luis Felipe Crespo Foix, and Antonio Léon Jiménez. „An accelerometer-based device for sleep apnea screening.“ In: *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society* 14.2 (2010), pp. 491–9 (cit. on p. 8).

[31]Hiroshi Nakano, Kenji Hirayama, Yumiko Sadamitsu, et al. „Monitoring sound to quantify snoring and sleep apnea severity using a smartphone: proof of concept.“ In: *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine* 10.1 (2014), pp. 73–8 (cit. on p. 8).

[32]Rajalakshmi Nandakumar, Shyamnath Gollakota, and Nathaniel Watson. „Contactless sleep apnea detection on smartphones.“ In: *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '15*. New York, New York, USA: ACM Press, 2015, pp. 45–57 (cit. on pp. 8, 12).

[33]F. Javier Nieto, Paul E. Peppard, Terry Young, et al. „Sleep-disordered breathing and cancer mortality: results from the Wisconsin Sleep Cohort Study.“ In: *American journal of respiratory and critical care medicine* 186.2 (2012), pp. 190–4 (cit. on p. 3).

[34]A. van Oosterom and T.F. Oostendorp. *Medische fysica*. 3th editio. Maarssen: Elsevier gezondheidszorg, 2008, p. 45 (cit. on p. 20).

[35]Kenny P Pang and David J Terris. „Screening for obstructive sleep apnea: an evidence-based analysis.“ In: *American journal of otolaryngology* 27.2 (2006), pp. 112–8 (cit. on p. 8).

[36]Kenny P Pang, Thomas A Dillard, Amy R Blanchard, et al. „A comparison of polysomnography and the SleepStrip in the diagnosis of OSA.“ In: *Otolaryngology–head and neck surgery : official journal of American Academy of Otolaryngology-Head and Neck Surgery* 135.2 (2006), pp. 265–8 (cit. on pp. 8, 9).

[37]D H Phan, S Bonnet, R Guillemaud, E Castelli, and N Y Pham Thi. „Estimation of respiratory waveform and heart rate using an accelerometer.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2008.Figure 2 (2008), pp. 4916–9 (cit. on p. 8).

[38]Paola Pierleoni, Luca Pernini, Alberto Belli, and Lorenzo Palma. „Real-time apnea detection using pressure sensor and tri-axial accelerometer“. In: *IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*. IEEE, 2014, pp. 513–516 (cit. on pp. 8, 9, 12, 51).

[39]M Pradella. „Breathing frequency in sleep related respiratory disturbances.“ In: *Arquivos de neuro-psiquiatria* 51.2 (1993), pp. 227–35 (cit. on p. 14).

[40]MD R. B. Berry and MEd R. Brooks. *The AASM manual for the scoring of sleep and associated events*. 2013 (cit. on pp. 5, 9, 11).

[41]David Barbosa Rendón, Juan L Rojas Ojeda, Luis F Crespo Foix, Daniel Sánchez Morillo, and Miguel A Fernández. „Mapping the human body for vibrations using an accelerometer.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2007 (2007), pp. 1671–4 (cit. on pp. 8, 51).

[42]Tracey D Robinson and Iven H Young. „Sleep apnoea“. In: *Medicine* 32.1 (2004), pp. 91–95 (cit. on pp. 4, 6).

[43]Scott A Sands and Robert L Owens. „Does my bed partner have OSA? There's an app for that!“ In: *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine* 10.1 (2014), pp. 79–80 (cit. on p. 8).

[44]Nandakumar Selvaraj and Ravi Narasimhan. „Automated prediction of the apnea-hypopnea index using a wireless patch sensor.“ In: *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* 2014 (2014), pp. 1897–900 (cit. on pp. 8, 9, 12).

[45]L. Thomas Sheffield. *The only EKG book you'll ever need*. 1989 (cit. on p. 12).

[46]J R Stradling and J H Crosby. „Predictors and prevalence of obstructive sleep apnoea and snoring in 1001 middle aged men.“ In: *Thorax* 46.2 (1991), pp. 85–90 (cit. on p. 3).

[47]G.E. de Vries, P.M. Meijer, B. Stegenga, et al. „Obstructief slaapapneusyndroom (OSAS): definitie, diagnostiek en behandeling.“ In: *Bijblijven* 27.1 (2011), pp. 51–58 (cit. on p. 6).

[48]Terry Young, Paul E. Peppard, and Daniel J. Gottlieb. „Epidemiology of obstructive sleep apnea: a population health perspective.“ In: *American journal of respiratory and critical care medicine* 165.9 (2002), pp. 1217–39 (cit. on pp. 3, 4).

[49]Terry Young, Laurel Finn, Paul E Peppard, et al. „Sleep disordered breathing and mortality: eighteen-year follow-up of the Wisconsin sleep cohort.“ In: *Sleep* 31.8 (2008), pp. 1071–8 (cit. on p. 3).

Acknowledgements

After years of studying, I am grateful for being able to complete this process. I would like to thank my family for being there for me. To help me to never give up, provide support and guidance and aiding me in reaching my goals.

During my graduation, I have had the pleasure of working with and learning from great people. I would like to thank Frans de Jong, Michiel Eijsvogel, Emanuel Citgez and Paul Hendriks, whith whom I have had many interesting conversations during my graduation process. I would like to express my sincere gratitude to Paul van Katwijk, whom has helped molding me during my study, to the person I currently am.

I would also like to thank Ed Droog, whom has devoted a part of his time to create the sensors I needed to complete this research.

Finally I would like to thank my friends and collogues for all the interesting discussions, chitchats and useful comments. It has been a true pleasure working with you all.

Appendix

10

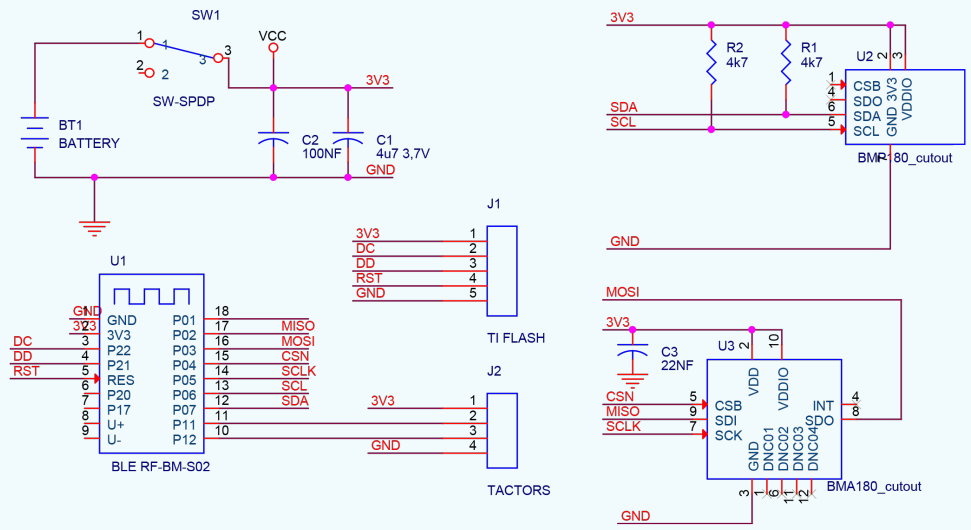
List of Acronyms

ATT	Attribute Protocol
C#	C-sharp - A programming language
COM port	Communication port
Core	Unit that reads and executes program instructions
EDF	European Data Format
GATT	Generic Attribute Profile
GUID	Globally Unique Identifier
IIR	Infinite Impulse Response
MAC address	Unique software address of a hardware device
Pollen	Asking for information (samples) with a regular interval
RFCOMM	Radio Frequency Communication - a bluetooth protocol
SPP	Serial Port Profile
Thread	A method to run tasks simultaneously on a CPU
UUID	Same as GUID

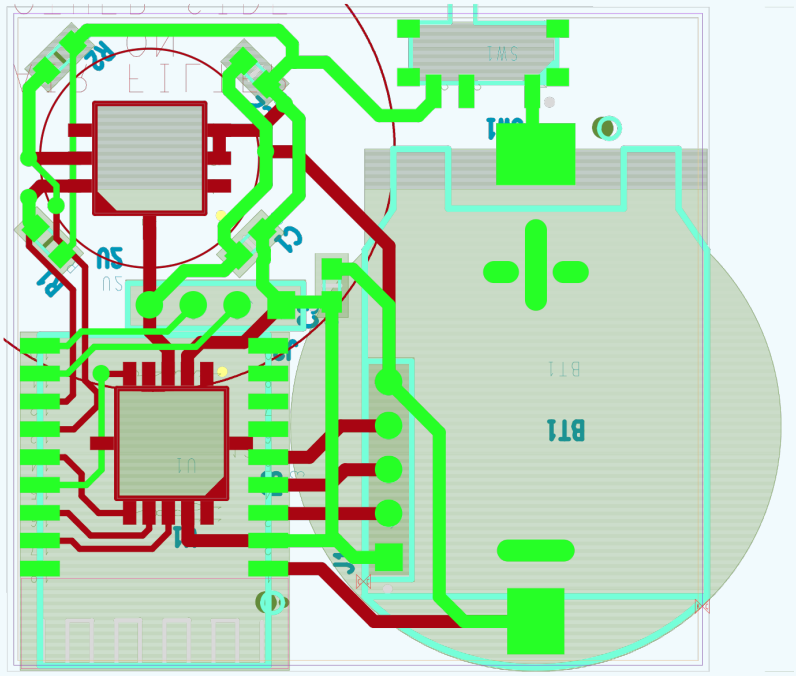
Box 10.1: CC2540 communication interfaces

Some electronic components, like our CC2540 'Bluetooth' chip, can communicate with other electronics. This communication can be done using two different interfaces: Inter-Integrated Circuit (I^2C) and Serial Peripheral Interface (SPI). Both interfaces are commonly used in electronics. Due to the effort of T. Wilkinson Ph.D, we can use the full potential of the CC2540 chip and program it to use these interfaces to communicate with other chips. This means that we do not need a second chip for 'reading' the sensors, but can use the capacity of the Bluetooth chip. This keeps the power consumption low and reduces the required size of the PCB, allowing us to make a small sensor. In section Bluetooth Low Energy Sensors, the requirements of the sensors are described. Many sensors met the criteria and we chose the PMB180 (pressure sensor) and BMA180 (accelerometer sensor) for our project. Both sensors, can communicate using both I^2C and SPI. They are also both manufactured by Bosch, are easy to obtain and have a full manual available online to aid in integrating in our design.

Box 10.2: Electric schematics of our sensor design



Box 10.3: PCB design



Box 10.4: BlueBasic code for detecting calibration parameters

The pressure sensor has its calibration values stored within the chip. These calibration values are put in the sensor by the manufacturer. The calibration values are required to convert the sensor's output to an understandable value, like kPa and degrees Celsius. The accelerometer outputs an integer which needs to be divided by a static number to obtain the acceleration value. The static number depends on the settings in the chip itself. By default, these settings allow values in the range of ±2G. Altering could increase the accuracy to ±1.5G or ±1G. We chose to not alter the accelerometer nor pressure settings. The settings must be exported once for each sensor.
The below code first makes a connection with the accelerometer chip, to obtain its ID. This connection is done using the SPI interface (line 20-80). After this, the chip connects to the pressure sensor over the I^2C interface (line 90). It then tells the chip what register to read (line 110), in this case the sensor ID, and read 1 byte of information to be stored in variable 'M'. It then reads the calibration values (line 140-150). The information is printed over the Bluetooth connection (line 170) and received on any receiving device.

```
1  NEW
2  5 A = 0
3  20 PINMODE P0(2) INPUT PULLUP
4  30 DIM I(2)
5  40 PINMODE P0(4) OUTPUT
6  50 SPI MASTER 0, 3, MSB 1
7  60 I(0) = 0X80
8  70 SPI TRANSFER P0(4) I(0)
9  80 PRINT "ID accel sensor: ", I(1)
10 90 I2C MASTER P0(6) P0(7) PULLUP
11 100 DIM M(1)
12 110 I2C WRITE 0XEE, 0XD0, READ M
13 120 DIM P(11)
14 130 PRINT "ID pressure sensor: " M(0)
15 140 DIM M(22)
16 150 I2C WRITE 0XEE, 0XAA, READ M
17 155 PRINT "pressure sensor calibration data"
18 160 FOR N = 0 TO 21
19 170 PRINT(M(N) << 8) + M(N + 1)
20 180 N = N + 1
21 190 NEXT N
22 250 DIM I(2)
23 260 P0(4) = 1
24 270 I(0) = 0X80
25 430 END
```

Box 10.5: Reading pressure sensor

The below code reads the pressure sensor data. This code sets a timer at 10Hz (line 100), to run the code from line 200 onwards. Using the I^2C interface it first reads the temperature value (line 240). The chip then reads the pressure value (line 270). The pressure value reading has a delay of 30ms. This is used to obtain a more average (and thus less noisy) signal. Finally the data is transmitted over the Bluetooth low energy connection (line 280).

```
1  NEW
2  10 I2C MASTER P0(6) P0(7) PULLUP
3  100 TIMER 0, 100 REPEAT GOSUB 200
4  110 RETURN
5  200 DIM P(3)
6  210 DIM T(2)
7  220 I2C WRITE 0XEE, 0XF4, 0X2E
8  230 DELAY 5
9  240 I2C WRITE 0XEE, 0XF6, READ T
10 250 I2C WRITE 0XEE, 0XF4, 0XF4
11 260 DELAY 30
12 270 I2C WRITE 0XEE, 0XF6, READ P
13 280 PRINT (T(0) << 8) + T(1)," ",(((P(0) << 16) +(P
       (1) << 8) + P(2)) >> 5)
14 290 RETURN
```

Box 10.6: Reading accelerometer sensor

The code below reads the accelerometer sensor data. This code first sets up a connection using SPI (line 10-50). It then sets a timer at 10Hz (line 100), to run the code from line 200 onwards. Using SPI it first reads the memory of the accelerometer sensor (line 200-220). Finally the data is transmitted over the Bluetooth low energy connection (line 230).

```
1  NEW
2  10 PINMODE P0(2) INPUT PULLUP
3  30 PINMODE P0(4) OUTPUT
4  40 SPI MASTER 0, 3, MSB 1
5  50 P0(4) = 1
6  100 TIMER 0, 100 REPEAT GOSUB 200
7  110 RETURN
8  200 DIM U(10)
9  210 U(0) = 0X80
10 220 SPI TRANSFER P0(4) U
11 230 PRINT (((U(4) << 8) + U(3)) >> 2)," ",(((U(6) <<
       8) + U(5)) >> 2)," ",(((U(8) << 8) + U(7)) >> 2)
12 240 RETURN
```

Box 10.7: Sensor protocols & Profiles

Bluetooth low energy uses the Generic Attribute Profile (GATT). This profile is build on top of the Attribute Protocol (ATT) and establishes common operations as well as a framework for the data transport. The GATT protocol defines two roles: The Server and Client role. GATT are mandatory in Bluetooth Low Energy (BTLE) since it is used for discovering services. Using the server implementation of GATT, a list can be obtained of all paired BTLE devices. Each BTLE sensor contains a Globally Unique Identifier (GUID, also refered to as UUID) which is used to identify a sensor. The GUID is also used to identify a GATT service on a sensor, which is unique for that service, but available on each sensor that contains that service. Using C-sharp (C#) code, it is possible to find all paired sensors that contain a certain service GUID. The resulting list can be looped through to store all sensors with a unique ID (d.Id in the below code) in a custom class (BLEDevices). The graphical user interface (GUI) uses this list to display all available sensors.

```
1   Guid Guid_service = new Guid("25FB9E91-1616-448D-B5A3-F70A64BDA73A");
2
3   var deviceList = await DeviceInformation.FindAllAsync(
4           GattDeviceService.GetDeviceSelectorFromUuid(
5                   Guid_service
6           )
7   );
8
9   foreach (var d in deviceList)
10  {
11          /* Check if the sensor exists in the "BLEDevices" list */
12          if (d.Id.Contains("BTHLEDevice") && !BLEDevices.Any(x => x.id == d.Id))
13          {
14                  /* Only store sensors that are not in the BLEDevices list */
15                  BLEDeviceObject tmp = new BLEDeviceObject();
16                  tmp.id = d.Id; /* unique ID */
17                  tmp.name = d.Name; /* Sensor Name */
18                  BLEDevices.Add(tmp);
19          }
20  }
```

Box 10.8: Nonin protocol & Profile

The Nonin 4100 pulse oximeter uses the Serial Port Profile (SPP). This profile communicates via the Radio Frequency Communication (RFCOMM) protocol and mimics a serial port connection. The software interface should thus connect to an emulated COM port. The BT radio that communicates with the Nonin pulse oximeter, has a MAC address. This MAC address is used to set up a serial port connection. The 'BluetoothClient' uses this connection which is then used by the 'BluetoothComponent' to actually search for in-range devices. If the 'BluetoothComponent' finds a device, it will be handled just like the BTLE sensors: each unique device is stored in a custom class (BLDevices) and displayed in the GUI.

```
1  /* Get Laptop Bluetooth radio */
2  BluetoothRadio myRadio = BluetoothRadio.PrimaryRadio;
3  if (myRadio == null) {
4          return; /* No BT radio found */
5  }
6
7  /* MAC address of laptop Bluetooth device */
8  localEndpoint = new BluetoothEndPoint(myRadio.LocalAddress, BluetoothService.SerialPort);
9
10 /* Client is used to manage connections */
11 localClient = new BluetoothClient(localEndpoint);
12
13 /* Component is used to manage device discovery */
14 localComponent = new BluetoothComponent(localClient);
15
16 /* Find devices */
17 localComponent.DiscoverDevicesAsync(255, true, true, true, true, null);
18 localComponent.DiscoverDevicesProgress += new EventHandler<DiscoverDevicesEventArgs>(
       ↪ component_DiscoverDevicesProgress );
```

Box 10.9: Connect to sensor

The user interface provides a list of all available sensors. Once a sensor is selected, the software tries to connect to that sensor. To do so, it uses the already stored sensor information in the custom class 'BLEService'. The BTLE sensor contains a couple of GATT Characteristics. A GATT Characteristic is an object that is pre-defined for a single task. Many Characteristics are standardised so that they can easily be used between sensor manufacturers. The receiving side (mostly an app on a smartphone) can than use all sensors that use the same characteristics without having to customize the code for new sensors. This BTLE sensor has 2 characteristics that are important. One can be read and set to notify, the other characteristic can be written to. The software creates an event, that is raised once the value of the 'read characteristics' of the BTLE sensor changes. In order for this value to change, the BTLE sensor needs to notify the software, and is thus set to Notify. For the BTLE sensor to actually start working, a "RUN" command is send to the other 'write characteristics'. Once this process is completed, the function "BLEData_ValueChanged" is executed each time the sensor has new data available. This function will further process the received information.

```
1  /* Setup a new connection */
2  BLEService = await GattDeviceService.FromIdAsync(id);
3
4  /* Store the GATT Characteristic that can be read from */
5  BLEData_ReadNotify = BLEService.GetCharacteristics( Characteri stics_ReadNotify ).First();
6
7  /* Add an event that runs once the value changes */
8  BLEData_ReadNotify.ValueChanged += BLEData_ValueChanged;
9
10 /* Store the GATT Characteristic that can be written to */
11 BLEData_Write = BLEService.GetCharacteristics( Characteristics_Write ).First();
12
13 /* Set the BTLE to Notify */
14 await BLEData_ReadNotify.WriteClientCharacteristicConfigurationDescriptorAsync(
       ↪ GattClientCharacteristicConfigurationDescriptorValue.Notify );
15
16 /* Send the RUN command to the BTLE sensor */
17 byte[] buffer = Encoding.ASCII.GetBytes("RUN" + "\r");
18 IBuffer ibuffer = buffer.AsBuffer();
19
20 /* Asynchronious send the RUN command to the sensor. */
21 await BLEData_Write.WriteValueAsync(ibuffer);
```

Box 10.10: Connect to Nonin

The Nonin 4100 uses an emulated serial port. A connection is created by connecting to the BT DeviceAddress over the serial port. Once connected, this connection holds a stream over which data can be written to, and received from the Nonin 4100 pulse oximeter. The Nonin device is configured to use "format #7", meaning that the plethysmography signal is sent with a resolution of 16 bit in stead of 8 bit and the SpO_2 and heart rate information is sent with 3Hz. The below code shows how the connection is set up, and how the format is set to #7.

```
1   /* Custom class to hold and handle Bluetooth Objects */
2   BLDeviceObject curBLDevices;
3
4   /* Laptop BT client is used to manage the connection */
5   LpBtClient = new BluetoothClient(localEndpoint);
6
7   /* Setup a new connection */
8   LpBtClient.Connect( curBLDevices.DeviceInfo.DeviceAddress, BluetoothService.SerialPort );
9   NetworkStream stream = LpBtClient.GetStream();
10
11  /* Send 2 bytes, to set the device to format #7 */
12  stream.Write(new byte[2] { 0x44, 0x37 }, 0, 2);
13  while (true)
14  {
15          data = new byte[1];
16          stream.Read(data, 0, data.Length);
17          if (data[0] == 6)
18          {
19                  break; /* Continue when the device responds with OK */
20          }
21  }
22
23  /* Start listning and reading data from the stream */
24  readNoninData();
```

Box 10.11: Reconnecting to a sensor

The BTLE sensor does not maintain an open connection, like the serial port of the Nonin device. When the serial port connection closes, both devices will know. The BTLE connection however does not provide this information if the sensor has stopped. This means that the laptop does not know if the sensor is still connected, it only knows when information was received for the last time. The solution for this problem is rather simple: a timer. C# allows the usage of multiple timers. A timer is an object that executes a funtion after a given time. The timer can be started, stopped but also reset. The software starts a timer that executes a "reconnect" function after 5 seconds. Each time however that data is received, the timer is being reset. Thus, only if no data was received for 5 seconds, the software will attempt to reconnect. Once the reconnection process has started, the timer is set to 100 seconds to give the BTLE time to reconnect. If the first reconnection attempt fails, a new one is started after 100 seconds until new data is received and the timer is set to 5 seconds again.

Box 10.12: Receiving and storing data

In order to understand the storage mechanism and the problems that arise with it, it is necessary to understand the basics of a computer. A computer exists of multiple components: A screen, Central Processing Unit (CPU), Hard Disk Drive (HDD), Random Access Memory (RAM) and a motherboard to connect them all. Modern CPU's have multiple cores within them so that one CPU chip, can actually have 2, 4, 6 or 8 cores. Each core can perform a task independently from the other cores. Some manufacturers have 3-cored CPU's, which actually is a 4-core CPU with one broken core. Each core can execute a task parallel to another core. Intel introduced the hyper-threading technique in 2002. This technique allows two tasks to run semi-parallel in one core. The resources a core has is shared over 2 threads. If one thread is on hold (e.g. waiting a few nanoseconds) the other thread can use the available resources.

The software is designed, so that each sensor-code can be executed on any thread. The laptop has a quad-core CPU with hyper-threading technique, meaning that a total of 8 threads are available. In total, there are 4 accelerometers, 1 pressure sensor, 1 pulse oximeter, 1 sound recorder and a GUI active during the registration. When data arrives, an event is raised as explained in section 'Connecting to a sensor'. The thread on which that event is executed, depends on what thread is available. This thread saves the arrival time in 'Ticks' of the data and will attempt to save the results to a file. Ticks is a time format, just like seconds, minutes, etc that is used by computers to define time with a high accuracy. One second is equal to 10.000.000 Ticks.

The data storing comes with a problem. Only one thread can write to the file at the same time. The threads however run parallel to each other, making it possible that two threads try to access the file at the same time. This problem is solved by placing a lock on the file. During the lock, only that thread can access the file (write to it) and all other threads who try to access it wil hold until the lock is removed. Since each sensor data is stored in its own file, this situation is not common. It is possible that sensor 1 sends package 1 which is handled by thread 1. However, due to a delay in the arrival of package 1, i.e. due to connection problems, package 2 is received also at the same time but handled by thread 2. The arrival timestamp of package 2 will likely be a few hundred or thousand Ticks higher, indicating that it indeed was received later. The computer can however still give thread 2 priority in accessing the file over thread 1, meaning that the package was received later. It had a higher timestamp, but is saved earlier to the datafile. A bigger problem occurs if two packages are received by two threads, at the exact same timestamp. This could be caused by connection problems.

Box 10.13: Receiving and storing data

The code below places a lock on the file, so that only one treat can access it at the same time. It stores the data in binary format, so that the file size is reduced to a minimum.

```
/* Lock access to the file, until the writing procedure is done */
lock (this)
{
        using (var stream = new BinaryWriter(new FileStream(FolderName + @"\ " + name + ".txt",
            ↪ FileMode.Append)))
        {
            /* 16 bytes */
            stream.Write(new byte[] { 187, 255 }); /* Write identification code */
            stream.Write(ticks, 0, ticks.Length); /* Write timestamp, Ticks */
            stream.Write(TP[0]); /* Write temperature value */
            stream.Write(TP[1]); /* Write pressure value */
        }
}
```

Box 10.14: Recording sound volume

The code below is called when the buffer of the audio recorder is filled with 50ms of data. The root-mean-square is calculated over the buffer and used to calculate the sound volume in dB Full Scale. This value is stored for analysis.

```
double sum = 0;
for (var i = 0; i < waveInEventArgs.Buffer.Length; i = i + 2)
{
        double sample = BitConverter.ToInt16(waveInEventArgs.Buffer, i) / 32768.0;
        sum += (sample * sample);
}
double rms = Math.Sqrt(sum / (waveInEventArgs.Buffer.Length / 2));
var decibel = 20 * Math.Log10(rms); /* in dB Full Scale */
```

Box 10.15: Ordering the samples and filling gaps

The Matlab code below shows how samples, saved in the wrong order, are rearranged to become time-successive sorted. The interpolant function is then used to generate data points that are missing. Earlier code (not shown here) makes sure the missing points will get a value of '0'.

```
% Handle duplicate timestamps.
T1(end) = T1(end) + 1; % same timestamp at end bugfix....
i = find( diff( T1 ) == 0 ); T1(i+1) = (T1(i) + T1(i+2)) /2;
[T1, I] = sort( T1 );
D1 = D1(I);

% Create the interpolant function F
Fx = griddedInterpolant( double( T1 ), double( D1 ), 'linear' );
```

Box 10.16: Signal synchronisation

The first step is to synchronise the data manually, by checking distinguishable peaks. The next step is to find the highest start-time of both signals. Prior to this moment, only one signal is available and thus that data cannot be synchronised. Ten minutes after that, a window of 180 seconds is taken. 180 seconds is enough, because of the manual synchronisation we have done. The first 10 minutes are excluded, since the Bluetooth sensor might not be connected yet to the nasal cannula. The start time (eq. 10.1), stop time (eq. 10.2) and sample frequency (eq. 10.3) are calculated. TS is the Timestamp array, corrected for manual synchronisation.

$$T_{start} = \left(max\left(TS_{PSG}\left(1\right), TS_{BLT}\left(1\right)\right) + 600\right) sec. \tag{10.1}$$

$$T_{stop} = \left(T_{start} + 180\right) sec. \tag{10.2}$$

$$fs = 4000Hz \tag{10.3}$$

With these variables, the data points are calculated using the interpolant function. For visual confirmation, both signals are scaled to the same size using equation 10.4, in which F is an array with all data points.

$$F_{PSG} = F_{PSG} \cdot \frac{max\left(F_{BLT}\right) - min\left(F_{BLT}\right)}{max\left(F_{PSG}\right) - min\left(F_{PSG}\right)} \tag{10.4}$$

The offset is then calculated using the cross-correlation. This mathematical method produces an estimate of the correlation between two random sequences for each lag m as shown in equation 10.5.

$$C\left(m\right) = E\left[F_{PSG}\left(n+m\right) \cdot conj\left(F_{BLT}\left(n\right)\right)\right] \tag{10.5}$$

The maximum value for C is sought and its corresponding index is multiplied to obtain the lag in ticks. This lag is used to correct the timestamp of the non-PSG signals.

$$Lag = \left(index\left(max\left(C\right)\right) \cdot 1/4000Hz \cdot 10.000.000\right) Ticks \tag{10.6}$$

Box 10.17: Calculating the true temperature and pressure value

The pressure value is calculated using the pressure reading of the sensor, the temperature reading of the sensor and many static calibration values which are read once from the sensor.

```
Constant calibration variables:
short AC1, AC2, AC3, B1, B2, MB, MC, MD
unsigned short AC4, AC5, AC6

UT = Raw temperature reading of the sensor
UP = Raw pressure reading of the sensor

X1 = (UT - AC6) * AC5 / 2^15
X2 = MC * 2^11 / (X1 = MD)
T = (B5 + 8) / 2^4 /* True temperature value */


B6 = B5 - 4000
X1 = (B2 * (B6 * B6 / 2^12 )) / 2^11
X2 = AC2 * B6 / 2^11
X3 = X1 + X2
B3 = ((AC1 * 4 + X3) << oss + 2) / 2^2
X1 = AC3 * B6 / 2^13
X2 = (B1 * (B6 * B6 / 2^12 )) / 2^16
X3 = ((X1 + X2) + 2) / 2^2
B4 = AC4 * (unsigend long)(X3 + 2^15) / 2^15
B7 = ((unsigned long)UP - B3) * (50000 >> oss)
if (B7 < 0x80000000) {
        p = (B7 * 2) / B4
}else{
        p = (B7 / B4) * 2
}
X1 = (p / 2^8 ) * (p / 2^8 )
X1 = (X1 * 3038) / 2^16
X2 = (-7357 * p) / 2^16
p = p + (X1 + X2 + 3791) / 24 /* True pressure value */
```

Box 10.18: Calculating the body position

In our research we have used four accelerometers. If they have continuously recorded a signal, we could just take the mean of all four sensors to calculate the body position in the centre of those sensors, which is the lower sternum. Our sensors did however not record all night, making the calculation a bit more complicated. To correctly calculate the body position we use the following method. First the sum is calculated of the thoracic, and abdominal signals (Sum 1 Thorax in the below example). Then the amount of values that added to that sum is calculated (Sum 2 Thorax in below example). Finally Sum 1 is divided by Sum 2, to obtain the correct values. This process is repeated for the abdomen, and finally for the mean thorax/abdomen signals. The Matlab implementation is added in the below code.

```
                Xaxis: First array
                0 0 0 1 2 3 2 1 2 3 2 1 0 0 0 0 0 /* Signal Left Thorax */
                0 0 3 2 3 4 2 3 4 2 3 4 2 0 0 0 0 /* Signal Right Thorax */
                0 0 3 3 5 7 4 4 6 5 5 5 2 0 0 0 0 /* Sum 1 Thorax */

                Astore: Second array
                0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 0 /* has value? Left */
                0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 /* has value? Right */
                0 0 1 2 2 2 2 2 2 2 2 2 1 0 0 0 0 /* Sum 2 Thorax */
```

```
% Time array
Ticks = (TicksStart:1000000:TicksEnd)';

% Array to hold all values
Xaxis = zeros( length( Ticks ), 4 );

% Array to store where a value is set
Astore = zeros( length( Ticks ), 4 );

for i = 1:4
ii = find( Ticks > signal{ i }.Ticks(1) & Ticks < signal{ i }.Ticks(end) );
Xaxis( ii(1):(ii(1)+length(ii)-1), i ) = signal{ i }.Fx( double( Ticks( ii ) ) );
Astore( ii(1):(ii(1)+length(ii)-1), i ) = ones( 1, length( ii ) );
end

% Calculate Thoracic signal
r = sum( Daxis(:,1:2), 2 );
Xaxis(:,1) = (sum( Xaxis( :, 1:2 ), 2 ) ./ r );
Astore(:,1) = r ~= 0;

% Calculate Abdominal signal
r = sum( Daxis(:,3:4), 2 );
Xaxis(:,2) = (sum( Xaxis( :, 3:4 ), 2 ) ./ r );
Astore(:,2) = r ~= 0;

% Calculate final signal
r = sum( Daxis(:,1:2), 2 );
Xaxis = (sum( Xaxis( :, 1:2 ), 2 ) ./ r );
```