Internal Report 2013–10 August 2013

Universiteit Leiden
Opleiding Informatica

A Mobile Smart Care platform

Home spirometry by using the smartphone microphone

Bas van Stein

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

A Mobile Smart Care Platform
Home spirometry by using the smart phone microphone

Bas van Stein

August 22, 2013

Abstract

Mobile health (mHealth) applications are increasingly seen as cost-effective alternatives for hospital care. In this paper, it is investigated whether a smart phone microphone-based application can replace a more common differential-pressure spirometry sensor. Moreover, we investigate the steps that are needed to implement an application that uses a sensor to mimic another sensor, for which its high cost and its mobility limitations prohibit its use. Nowadays with the extended use of smart phones, people carry a set of sensors with them that can be used in many situations. Using these sensors for measurement (for which the sensors were initially not intended) gives us a huge variety of possibilities and might change the way mHealth and mobile phones are seen completely.

The spirometry sensor is used for patients with chronic obstructive pulmonary disease and due to the way in which the health care market functions, very expensive. A smart phone application that can replace this sensor would be a cost-effective alternative. By using the smart phone's built-in microphone and sound analysis algorithms we create a prototype of such an application. The application is tested among a group of patients and a control group to see if the outcome is acceptable, in order to use the application as a replacement of differential-pressure spirometry sensors.

Contents

1 Introduction

Mobile health (mHealth) smart phone applications are increasingly seen as a cost-effective alternative for hospital care. *mHealth* or *eHealth* (health care that uses electronic technology) can be explained as health care *independent of your location* by means of technology. Usually, mHealth applications focus on the management of specific diseases, often chronic ones such as *diabetes mellitus*, *chronic obstructive pulmonary disease* (COPD), *asthma*, and *heart failure*. Smart-phone-like devices are ideal platforms for mHealth applications as smart phones are the cheapest, smallest, most powerful general purpose computers available, that can communicate with the environment through wireless networks. Typically, mHealth applications are used to monitor the condition of the patient through regular measurements and computer-generated questions. This results in patient's data that are interpreted in the smart phone by an intelligent system, resulting in a prediction of whether the condition is stable, improving or worsening. It is also possible to determine treatment actions automatically that would have a positive effect on the well-being of the patient.

To be cost effective it is necessary, that mHealth applications make use of cheap sensors for the purpose of measurements of disease parameters. For pulmonary diseases, one of these sensors is a spirometer. Unfortunately, spirometers are very expensive, not so much because of the costly nature of their components, but rather because of the way the health care market functions. There is now scientific evidence that the microphone of the smart phone can also be used to measure lung function [6]. If this is true, then this could result in a major breakthrough in the management of COPD and other lung diseases because of low costs and great availability.

Many researchers try to obtain information with a limited set of sensors and more specifically, try to use sensors to measure different parameters from what they are designed for. With current technology, you see more and more applications that try to use the smart phone's sensors to their advantage, whether the sensors are meant for it or not. Apps are created to do things that nobody expected to be possible, by using the sensors of the smart phone to their full utility. The difficult part of this though, is that we have to convert the sensor's output to the expected output of the parameters we want to measure. We investigated such a process for the smart phone's microphone: How can we interpret the sound captured by the smart phone in such a way so that we can measure the function of the human lungs? This thesis is not the only result of the project, we also implemented a working system that can perform a spirometry test by only using the smart phone's built-in microphone.

In this master thesis at the Leiden Institude of Advanced Computer Science (LIACS), we investigated whether a smart phone *microphone-based* interpretation algorithm can replace the more common and much more expensive *differential-pressure* spirometry sensor. The latter is part of an experimental mHealth application for the management of COPD that has been developed at Radboud University [16]. To be useful, the application has to be reliable and proven to work "acceptable" within particular bounds.

In Section 2 an introduction to spirometry and mobile health care is given, to explain the purpose of the application. In the section that follows (Section 3), the research questions of this master thesis are discussed. In Section 4 the algorithms used in the final program will be mentioned and explained. A detailed overview of the implementation will be given in Section 5.1. In the sections that follow, we explain how we calibrate the app in order to

work with the microphone as a spirometer. Following, the results of the program will be given. Finally the results are analyzed and a conclusion is presented.

2 Biomedical background: mHealth and spirometry

In the following subsections we explain the biomedical background, required to understand the project and results.

2.1 mHealth

Mobile health care becomes more and more popular since more things are nowadays possible with respect of hardware and software. Mobile health is also becoming very important to diagnose illnesses as quickly as possible. Due to *mHealth*, massive health tests are possible and more people can be diagnosed at an early stage of their disease. With most chronic, but also non-chronic diseases, an early diagnosis is vital. Mobile health can help in diagnosing a disease in a patient, and moreover, help in monitoring the patient's condition. With mobile health care the patient can perform several tests at home by himself. The test results then usually are transferred to a web portal that the doctor can access. If the patient's condition becomes worse or unstable, the treatment can be adjusted. In this way a patient does not have to visit the hospital regularly to check his status and only needs to visit the doctor if he is in need. Both the patient and the hospital benefit from using mHealth; the patient because his disease is monitored which gives a safe feeling and other opportunities for a personalized treatment and the patient does not need to visit the hospital as frequently as without mHealth. The hospital benefits because the doctor can monitor more patients at the same time and can view the condition of his patients in a structured way, which provides better overview of the patients progress. Nowadays, hospitals use specialized mobile devices that are lent to patients to perform measurements. These devices are usually expensive due to the health care industry as well as the sensors that are used. More ideally would be to use more general purpose devices with a limited set of sensors, that can be used to perform many health tests and are able to diagnose many diseases. A smart phone could be a good candidate for that, since the modern smart phone has a wide range of sensors inside. If we can use those sensors to create mHealth applications, then we could serve more patients, and also healthy people, with these applications. This would lead to a faster diagnosis of the diseases. In the end, this will save a lot of money on treatment, and will improve the patients life. One of the diseases that might benefit from mHealth is COPD, as well as other chronic lung diseases. These diseases are most often diagnosed and monitored by means of spirometry.

2.2 Spirometry

Spirometry [18] is one of the most common pulmonary function tests. Pulmonary function tests are a group of tests, that measure how well the lungs inhale and exhale air. But also, how well they move gases, such as oxygen, from the atmosphere into the body's blood circulation. The volume of air and the speed of the air that can be inhaled and exhaled are measured by spirometry. Many devices have been created over the past time to perform these tests. These devices are described in the subsection below, a detailed explanation of

the parameters of the possible tests is given subsequently, as well as an explanation of how spirometry tests are performed and why a smart phone application is preferred.

2.2.1 History and devices

The earliest known history of the concept of spirometry starts around 129–200 AD (Roman Empire). Claudius Galen, a Greek doctor and philosopher, made a boy breathe in and out of a dried bladder from an animal like a pig, and discovered that after a period of time, the volume of gas did not change [14].Many devices since then have been created to perform spirometry tests.

One of the earliest volumetric spirometers is the *Water bell* by Hutchinson (Figure 1). The spirometer works with a calibrated bell, inverted in water, that captures the exhaled air from the lungs. This spirometer is still used today, although with some slight modifications.

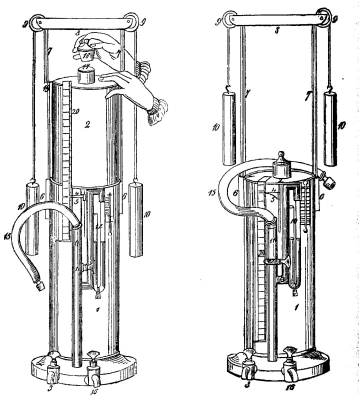


Figure 1: Water bell, an early spirometer

Modern spirometers use turbines that spin when air comes in or have sensors that measure the airflow passing, like anenometers. These devices are usually big and are located in hospitals, as they have many sensors and can capture the air to analyze the oxygen and carbon dioxide levels. There exist several kinds of mobile spirometers as well, but they are quite expensive.

2.2.2 Measurements

When a spirometry test is performed the patient is asked to breathe into the device. Depending on the kind of spirometry test this can be either normal breathing, deep breathing and completely exhaling or exhaling with force. In Figure 2 one can see the patient first breathing normally (*Tidal Volume*) and then taking a deep breath (filling the *Inspiratory Reserve Volume*(IRV)) and exhaling as much air as possible, emptying first the IRV and the Tidal Volume and then the *Expiratory Reserve Volume* (ERV); together we call them the Vital Capacity. There is always a volume of air left in the lungs, this is called the *Residual Volume* (RV).

A spirometer measures a few parameters to analyze the patient's status. Parameters that are most common to measure are:

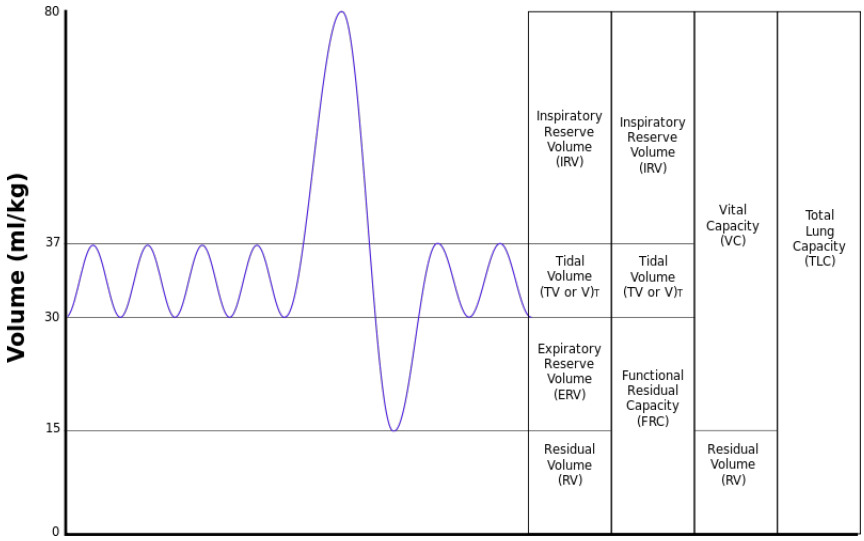


Figure 2: Lung volumes

- Forced Vital Capacity (FVC)
- Peak Expiratory Flow (PEF)
- Forced Expiratory Volume (FEV)
- Forced Expiratory Flow 25–75% ($FEF_{25–75\%}$)

These measurements are well documented and standardized. The devices that perform spirometry tests have to be certified and used in standardized ways [10, 2]. FVC is the volume of air, that can be blown out after full inspiration. FVC is measured in liters and is the most basic maneuver in spirometry tests. PEF is defined as the fastest speed (liters per second) exhaled by the patient. FEV is the volume of blown out air in a given time. Usually FEV_1 is used; FEV_1 is the amount of air exhaled in the first second. Normally this is around 80% of the amount of FVC. If the patient's lungs are obstructed this percentage will be a lot lower. $FEF_{25–75\%}$ is the volume in liters divided by the time in seconds of the middle half of the total expiration (by volume) (Figure 3). The values of $FEF_{25–75\%}$ and FEV_1 are generally close in patients with COPD.

A diagnosis can be made based on these five parameters. Normal values for the parameters that are known for all ages are shown in Figure 4. Ideally our program should be able to capture all these parameters. The normal values are used by our application to determine the condition of the user. By using a small number of input fields for the age and gender of the user, we can determine which normal values should be applied.

Chronic obstructive pulmonary disease (COPD) is the occurrence of chronic bronchitis or emphysema, diseases of the lungs. These diseases make the airways narrow over time and create large air gaps by destroying lung tissue, which causes shortness of breath. Usually COPD is detected in a late stage because the symptoms are not recognized earlier, by

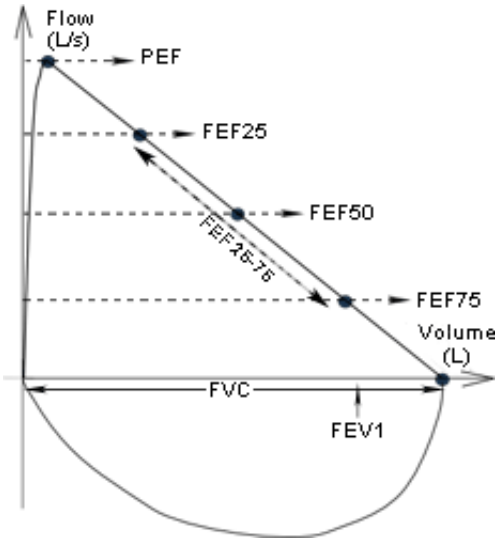


Figure 3: Example plot output of a spirometer

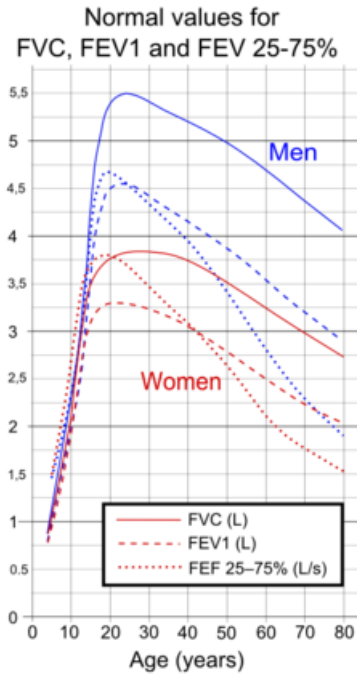


Figure 4: Normal values for all ages and genders [15]

the patient, as COPD. It has been shown that spirometry at home and in large groups,

significantly improves early detection of COPD [1, 19]. In order to apply spirometry at home and on a large scale, it is important that spirometry can be applied at a low cost. An application for smart phone devices could be such a low-cost alternative.

2.2.3 Techniques

Some techniques (sensors) used in *volumetric* and *flow measuring* spirometers are:

- **Water bell**: Calibrated inverted bell, filled with water, that captures gas.
- **Turbine**: Rotates because of incoming air, used to measure airflow.
- **Ultrasound**: Ultra sonic sound to scan the airways.
- **Anenometer**: Sensor mostly used in weather stations, the speed of rotation can be used to measure quantity of air passing.

We can split spirometry into two groups: Closed-circuit spirometry and open-circuit spirometry [9]. Closed-circuit spirometry is when a person breathes into a pre-filled container. The exhaled air passes trough a canister of soda lime. The lime absorbs the carbon dioxide and the oxygen passes back into the container. The amount of oxygen removed is recorded. In this paper we mainly talk about closed-circuit spirometry. This type of spirometry is used in hospitals and in mobile health care specifically because of small portable devices that can be used for it. It is called closed-circuit spirometry, because the air is measured in a closed environment, like a turbine or a container.

Open-circuit spirometry is useful for measuring energy expenditure during physical activity, because the patient does not need to breath in a big machine and can therefore perform physical activity like running. Some common methods of open-circuit spirometry are *portable spirometry* and *computerized instrumentation*. The person breathes in ambient air and the exhaled air passes only trough a gas meter. In portable spirometry the person carries the equipment usually in a backpack form. The gas meter in portable spirometry also takes a sample of the ambient air to use when analyzing the oxygen consumption. In computerized instrumentation, the expired air is continuously sampled, while measuring air flow, and the gas mixture's composition.

Modern spirometry devices are usually working with computerized instrumentation. The sensors are attached to a processing unit that records and analyzes the parameters mentioned in Section 2.2.2. A portable spirometer that can be used at home like the one in Figure 5c, costs around $1500. A mobile application on a phone that can do the same measurements only costs a few euros plus the cost of the device, which is around $300, depending on the device of choice.

3 Research challenges

This project was inspired by a project of the Radboud University, in the Netherlands. A similar experimental mHealth application is developed to facilitate health care for COPD patients. This is done by means of regular measurements with sensors and a questionnaire [16, 17]. This application uses a blue-tooth sensor interface (MOBI from Twente Medical Systems) which connects a custom micro-spirometer and a pulse-oximeter to the smart

(a) usb-based spirometer. (b) desktop spirometer. (c) turbine based spirometer.

Figure 5: Some modern spirometry tools

phone. The raw spirometer data is being read and processed by the smart phone to obtain FEV_1 values (Figure 6).

Figure 6: Scheme of the system setup [17]

The data being captured from the sensors are then, together with the data from a questionnaire, further processed by means of a Bayesian network model [7]. A Bayesian network is a probabilistic graphical model represented as an acyclic directed graph consisting of vertices (V) and edges (E), where V represents the random variables of interest and E the dependencies between them.

With this "intelligent" model the likelihood of the health status of the patient is calculated. The health status is presented to the patient, but is also uploaded by means of a mobile internet connection (or WIFI) to a database, accessible to the doctor of the patient.

This application is basically a small spirometer with low-cost sensors and using the smart phone general purpose processor to do the calculations and user interaction. Our project is to create a similar system but without any sensor that is not already in the smart phone. If we accomplish this, which was not expected, people are able to do a spirometry test by just using their smart phone and nothing else, which makes the application extremely portable. The research questions that we are interested in, are the following:

- How can we use a sensor to measure a parameter for which the sensor is not intended in the first place?
- What are in generally the steps needed to implement such application?
- What problems may arise when using a sensor to replace other sensors?
- Is it possible to do a spirometry test using only a smart phone?
  The huge difference between a spirometer and microphone indicates that it would be hard to achieve this goal. However, it was expected to be likely that the microphone would at least give some indication of lung function. The extent to which such microphone-based measurement would be useful was therefore another question to be investigated.
- How can we measure that the replacement sensor works reasonable well?
- Which could be the limitations of measuring with a different sensor?
- What are the possible advantages of the use of another sensor?
- In which other situations can it be useful to use a replacement for a sensor?

If we succeed in answering these questions, healthcare can be changed dramatically (especially mHealth), by creating new opportunities to develop cheap solutions in order to monitor and diagnose patients on a massive scale. In the subsections below we explain how we answer the above mentioned questions.

3.1 How to replace a sensor by another sensor?

To measure, a parameter that is normally not measured by a sensor, we can distinguish several steps. First of all, we need to choose the sensor we will be using. To use a sensor in a way for which it is not built, one should first be sure that the sensor is able to sense that, what is related to the parameter to be measured. For example; if we want to measure water temperature, a light sensor will not help us, since it can not measure something related to temperature (unless water changes color of the way it filters light when changing temperature). In order to choose the correct replacing sensor we can ask ourself a couple of questions: Is the replacing sensor able to sense information related to the parameter we want to measure? Is the replacing sensor available? Does the replacing sensor need dependencies? Would the replacement sensor give an advantage over the original sensor? Most of the time we would like to use a different sensor if the sensor we want to use is not available in the enviroment we want to apply it, or is more expensive. In our case we choose the microphone to replace a differential pressure spirometer sensor. The microphone is able to sense sound, since sound is basically moving air with pressure differences and the volume of the sound

is related to the speed which the air passes, so the microphone might be able to sense the speed of air passing it. The microphone is available in every Android smart phone and does not need any other sensor or dependency different than the phone itself. The advantage of using the microphone, rather than being a far cheaper solution, is also that the application can be used for spirometry on a massive scale without the need of custom equipment.

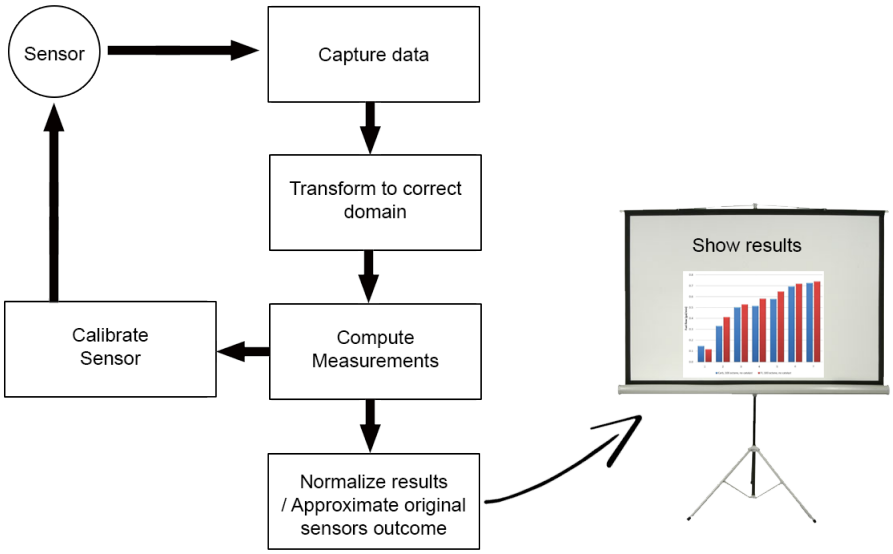


Figure 7: Steps needed for implementation

As long as the advantages of using a different sensor are strong enough and the replacing sensor is available and able to sense related information, we can continue to the next step. The second step in the process is to transform the raw data captured by our sensor to the domain of the sensor we want to mimic. The microphone captures sound data that are in an amplitude-time domain. What we whish to measure, i.e., liters of air over time, can be expressed as the amplitude of certain frequencies over time. The frequencies we are interested in, are the frequencies made by passing air. Thus, our second step is to transform our audio data to a frequency power spectrum. Not to lose the notion of time, we use multiple time windows of the audio data. Windows (pieces of the data) of 0.1 seconds are used, which are half overlapping. We choose 0.1 seconds because we need a minimum amount of samples to perform the Fourier Transform. The overlap there, is to make sure we do not lose frequency information between the windows.

If the transformation to the, more or less correct, domain is done, we can extract the measurements we are interested in. This is the third step in our process. Extracting the information, in our case, is mapping the powerspectrum over time, to liters air over time and adding these values to obtain FEV_1 and $FEF_{25–75\%}$. The mapping can be created by calibrating the sensor to mimic the outcome of the sensor we want to mimic. Calibration

can be done, for instance, by testing the sensor and the original sensor multiple times and comparing the output of the tests. After calibration we need to make sure that we handle noise and possible errors in our process. This can be done by funcion approximation techniques, filters and tresholds and more testing. The entire implementation process can be seen as the scheme in Figure 7.

3.2 Problems that may arise

In implementing software, problems are always around the corner. When replacing a sensor by another sensor you might find out that the range of the sensor is not good enough to measure what the original sensor measures and this, unfortunately, can not be solved easily. Another problem might be, that you have to deal with a lot of noise. Either white noise from the environment or noise that is there because you measure something related, but not exactly the same as what the original sensor is able to measure. White noise can be dealt with, by applying a noise filter or certain tresholds. In our case, we deal with white noise by adding the low frequencies together and ignoring the high frequencies. Non-white noise, however, is much more difficult to deal with and the solution differs at each application. For our case this problem is dealt with by using a function approximation algorithm. This algorithm approximates the function we can observe as an outcome of the spirometer sensor. By approximating this function with the use of our raw data we get more smooth and better results.

3.3 Measure quality

When we have implemented our application with a sensor that replaces another sensor we want to be sure that the application works sufficiently well. How can we measure the quality of our measurements? An option is to test the application and the original sensor together under the same circumstances. In our case this means testing the application and the sensor with a group of people. If exactly the same circumstances can not be created for both sensors then multiple runs are needed per test to give an approximation of the same circumstances. The quality of the application can then be seen as the error, or average error, that the application has in perspective to the original sensor. In our case, we need to test the application, not only to healthy people but also to patients with a lung disease like COPD. Patients can breathe in a way entirely different from healthy subjects. To create a clinical application we often have to deal with standards and certifications. In our case the spirometer is a certified clinical device, that is not allowed to have a higher error than 3%. If we want to certify our application as a clinical application we have to make sure we can guarantee this error rate. However, in our project, it is not our goal to create a clinical spirometer, but rather a handy alternative in cases that we do not have acces to a clinical one.

3.4 Limitations and advantages

An important question is what the advantages are, and more importantly, the possible limitations of this replacement. The advantages can be the lower cost, faster development, more mobility or a combination of them. However, there can also be advantages that are not obvious at the beginning. We can, for example, think of other measurements, that we can

perform with our sensor, that might improve our application. In our case this might be the measuring of the high frequencies of the sound when the patient exhales, so that we can see if the patient is wheezing, that might come from an obstruction in the lungways. It is also an advantage that in our case we use the smart phone as our general purpose processor. The smart phone has also other sensors that we can use in combination with the microphone. We can use the GPS location of the patient, for example, to inform the doctor of a patient's location in case of emergency, or we can use the Wifi or mobile internet connection to share the patient's data with the doctor of his/her preference. On the other hand, there are also limitations, of course. A limitation might be, that it is not possible to measure everything that the original sensor is able to measure. For example, we can not measure the oxygen and carbon dioxide levels in the air that is exhaled, with the microphone. An important limitation is also that the measurement might not be good enough in all cases. That is why a replacement sensor can be used in home-care for patient's daily checkup, but should not be used as a tool to give the final diagnosis or as a tool to use in hospitals.

3.5 Other situations and uses

We can for example, measure heart rhythm by using the microphone as a stethoscope. Or we can detect the frequency with which the user coughs or the amount of it per day to see if his/her condition improves or becomes worse. With the use of the camera on modern smart phones, we might be able to analyse skin diseases by taking a photo on various parts of the body. Another use of the camera might be the automatic detection of the result of a pH strip test or other test that uses color for showing the result [8]. Testing pupil reflections or measuring the size of the pupil might also be done by using the camera. Another idea is to detect or even predict epileptic attacks by using accelerometers [4].
Now we only referred to mHealth applications, but of course, there are many more fields that can benefit from the use of a sensor in different ways. Not only it is extremely useful to be able to measure things with different sensors, but it is even more useful to measure multiple things by using only a selected set of sensors. A smart phone, in that case, is perfect because it has a wide range of sensors and is handy in size and usability. The smart phone's processor can be used to do the IO handling and processing of the data and the smart phone can also handle the user interaction. If we can use the smart phone's sensors in multiple ways to create mHealth applications, we are able to make a device that can diagnose and control multiple diseases with low cost and massive scalability.

4 Methods and algorithms

To use the smart phones microphone as a replacement for the existing spirometer sensors we need to analyze the sound captured by the microphone. In order to process and interpret the sound data we used a couple of existing algorithms and bundle them together to obtain our final results. We first analyzed the sound data by frequency analysis, by using multiple *Fast Fourier Transforms* with overlapping time windows. The results of the Fourier Transforms are smoothed by regression lines with the use of *polynomial regression*.

From the plots we calculate a series of measurements, that are also used in clinical spirometers. We calculate FEV_1 and $FEF_{25–75\%}$. The parameters then will be evaluated along with some information of the patient. A diagnosis can be based on the parameters

and the patient's age and gender. We can warn the user if the values become worse than what they were in the past or if the user is not a COPD patient and the values tell us that he/she might suffer from COPD, we can warn the user and suggest a visit to a doctor. This way we can apply mass home-spirometry in an easy and effective way. For simplicity, we only remember the last known test results to compare with, though in the future this could be done with a local database so we can analyze the complete history of the patient.

4.1 Fast Fourier Transform

The *Fast Fourier Transform* (FFT) is used to compute the *discrete Fourier transform* (DFT) and its inverse. It is a commonly used algorithm in *sound analysis*, basically transforming the amplitude time space into a frequency power spectrum. By using the FFT on sound samples we can analyze which frequency is loudest, for example. If we split a sound sample in time windows of, for instance, 0.01 seconds, we can track the frequencies over time. The discrete Fourier transform is defined as follows:

$$DFT: X_k = \sum_{n=0}^{N-1} X_n e^{-(2\pi i/N)nk}$$

Here n stands for the descretized notion of time. There are many FFT algorithms available nowadays, all with different properties and time complexity. The most commonly used variant is the version from *Cooley-Tukey* [3]. This version of the FFT runs in $O(N \log(N))$ time and is therefore suitable for our cause. The DFT of size N, where N is a power of two, is computed by first computing the DFT of the even-indexed inputs and of the odd-indexed inputs. Then these results are combined. Splitting the inputs is done in this way recursively. The intermediate results of the algorithm are stored to gain speed by re-using these results for multiple DFT outputs. In pseudo code the *Cooley Tukey* algorithm is given as Algorithm 1.

Algorithm 1 *FFT_Cooley_Tukey*(x,N,s)

Require: array x of size N and step s.
  if $N = 1$ **then**
    $X_0 \leftarrow x_0$
  else
    $X_0, X_1, \ldots, X_{N/2-1} \leftarrow FFT\_Cooley\_Tukey(x, N/2, 2s)$
    {DFT of $(x_0, x_{2s}, x_{4s}, \ldots)$}
    $X_{N/2}, X_{N/2+1}, \ldots, X_{N-1} \leftarrow FFT\_Cooley\_Tukey(x+s, N/2, 2s)$
    {DFT of $(x_s, x_{s+2s}, x_{s+4s}, \ldots)$}
    for $k = 0$ to $N/2 - 1$ **do**
      $t \leftarrow X_k$
      $X_k \leftarrow t + \exp(-2\pi ik/N) * X_{k+N/2}$
      $X_{k+N/2} \leftarrow t - \exp(-2\pi ik/N) * X_{k+N/2}$
    end for
  end if
  return X

In our application we use the FFT to determine the "loudest" frequencies and to track

these frequencies over time. By assuming that the air being blown directly into the microphone produces the loudest frequencies, we can determine the amount of air exhaled over time. The resulting plot of the power spectrum of this particular frequency band over time then needs to be mapped to liters per second.

4.2 Function approximation

To deal with noise and errors in the transformation we can use a function approximation method to approximate the desired outcome. There are multiple ways to accomplish this. We can use regression, deterministic or non-deterministic curve fitting, neural networks, genetic programming and many more.
For our application, the algorithm has to deal with several limitations; the main memory is small (around 19MB), and the computing power can be small. Furthermore, we want the algorithm to finish in a few seconds. These limitations make neural networks and genetic programming not a good choice. The difference between deterministic and non-deterministic curve fitting, is that in deterministic curve fitting we find a function that touches every data point. Non-deterministic does not need to touch every point, but searches a line in between the points, like regression. Regression seems to be a good solution since it is rather lightweight and fast.

4.2.1 Regression options

Now that we choose for a regression algorithm we still have many options what kind of regression we take:

- Linear regression [13]
- Polynomial regression (every degree) [11]
- Using Taylor expansion
- Orthogonal polynomials [12]

We can define our choice by looking at the expected output of a spirometer. The function that a spirometer gives as output is not a linear function and so the first option can be excluded. A higher polynomial function might work, but the question then still is which degree? A polynomial function approximation can also be created by using Taylor expansion. Tailor expansion uses local data (one data point) to calculate the entire function. This can only be done if the function is already known. In our case this is not preferred since different people can breath in an entirely different way, especially when they have a lung disease. Using orthogonal polynomials can be useful since they have some "extra" features. However, we choose to use a polynomial function by applying a polynomial regression algorithm that we explain in the subsection below. There we also explain what degree we took and why.

4.2.2 Polynomial regression

Polynomial regression is a (modeling) technique to model a n^{th} degree polynomial to fit a nonlinear, or if $n = 1$ linear, relationship of x and y values. The goal of regression is to

model a dependent variable y in terms of an independent variable x. Such a model can be a polynomial function like:

$$p(x) = a_0 + a_1 x + a_2 x^2 + \varepsilon$$

Where ε is a random variable with a mean of 0 and standard deviation σ. We used polynomial regression to fit a curve to the data points obtained from our Fourier analysis.

For polynomial regression over points x and y, we first create a matrix like the matrix below (for degree 3):

$$\begin{bmatrix} \sum {x_i}^0 & \sum {x_i}^1 & \sum {x_i}^2 & \sum {x_i}^3 \\ \sum {x_i}^1 & \sum {x_i}^2 & \sum {x_i}^3 & \sum {x_i}^4 \\ \sum {x_i}^2 & \sum {x_i}^3 & \sum {x_i}^4 & \sum {x_i}^5 \\ \sum {x_i}^3 & \sum {x_i}^4 & \sum {x_i}^5 & \sum {x_i}^6 \end{bmatrix}$$

Here $\sum {x_i}^k$ stands for the sum of all x values, and this number to the power k.
Lets call this matrix X, and then need to solve $Xa = Y$ where Y is the vector:

$$\begin{bmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \\ \sum x_i^3 y_i \end{bmatrix}$$

This matrix equation is solved by means of a *Gauss-Jordan* elimination [5]. The resulting matrix A, contains the polynomial coefficients we want to use for our plot.

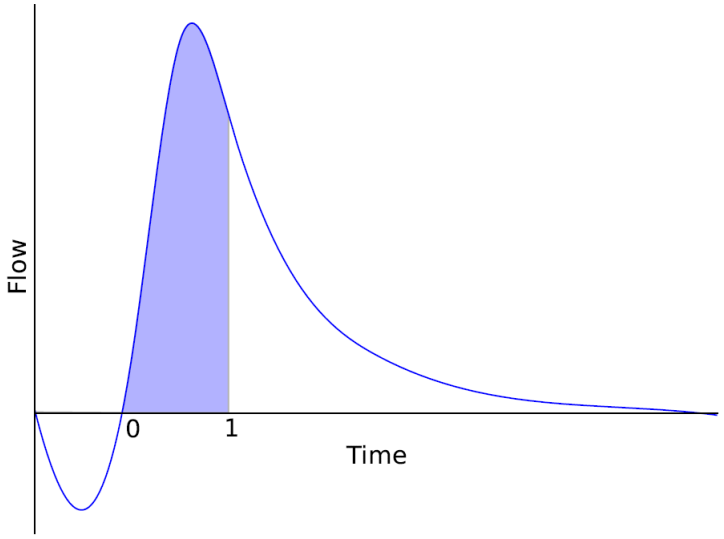


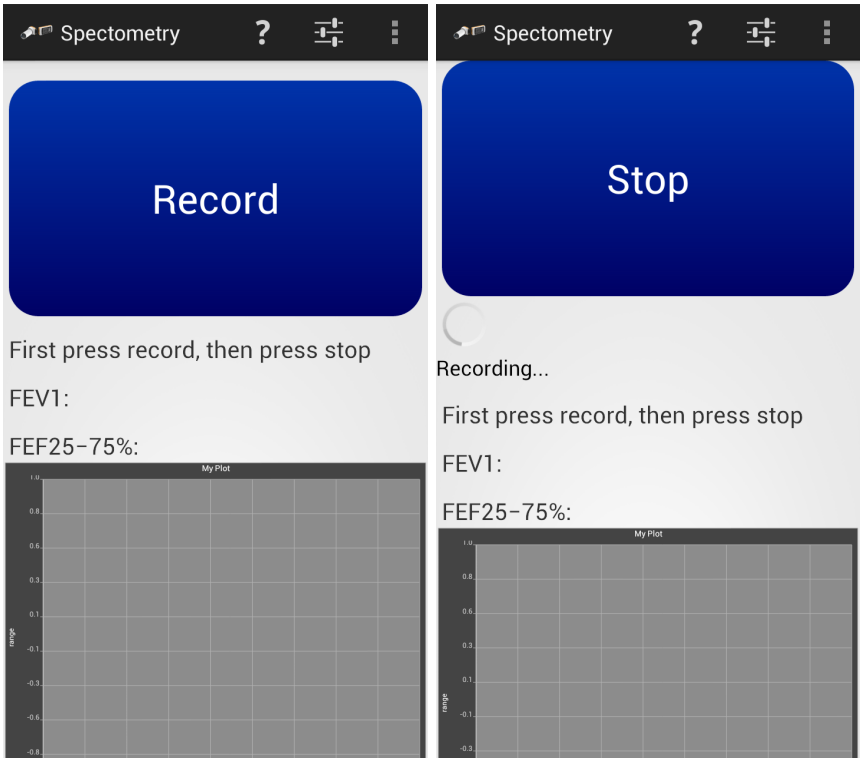
Figure 8: Expected plot from a spirometer [17]

We use a polynomial regression line of the fourth degree in our application. This line regresses the air flow beginning at time zero. Time zero is the time the patients starts

to breath out. We use a fourth degree polynomial, because the usual plot we observe in spirometers (Figure 8) first slightly increases to a top value (PEV) and later on the curve decreases in a parabolic way. When our polynomial decreases to zero we discard the rest of the regression line (it goes below zero and then up again).

5 The application

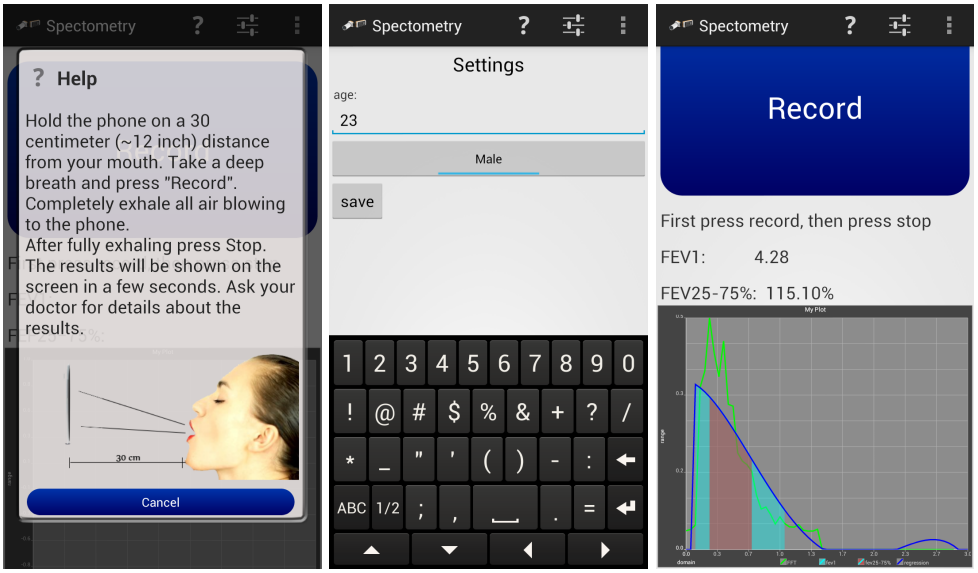
The design of the application (that is called SpiroApp from now on) is simple, to ensure that, e.g., people that are not able to see well can still use the application easily.



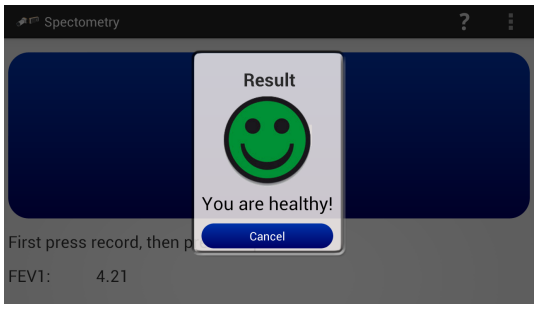
(a) View when you open the app (b) After pressing Record

Figure 9: Screen shots of the application

There is only one big button (Figure 9a) that starts a lung function test and when this button is pressed, the button changes to a "stop" button (Figure 9b) to stop the test (once the patient is done breathing out). When the stop button is pressed, the information is analyzed and the result will be shown below the button (Figure 10c) . We show two parameters, FEV_1 and $FEF_{25-75\%}$. The latter we show as a percentage of FEV_1, this is, because it does not matter in what way the patient breathes, how far the microphone is in distance and at which direction the patient breathes. In other words, $FEF_{25-75\%}$, devided



(a) How to use the app (b) Settings for the user (c) Result display



(d) Result display with smiley

Figure 10: Screen shots of the application (continue)

by FEV_1 times 100, is a good way to see if a patient is healthy or not since it is more stable than FEV_1 on its own.

The app has several screens for the user to make the app more intuitive. There is a help screen, which can be accessed by pressing the question mark from the menu, that explains how the patient should use the application (Figure 10a). After a test has been performed a pop-up display informs the user whether the result is “good”, “maybe bad” or “bad” (Figure 10d). The app also has a settings screen (Figure 10b). This screen is used to gather important data from the patient, like age and gender. The information is stored in the device’s local storage and can be used to diagnose the patient’s condition.

5.1 Implementation

In this section we will explain in detail the implementation of our application. First of all we use the microphone to record a piece of audio. The length of the audio footage depends on the user. The user presses "record" to start recording and "stop" to finish the recording. After the recording is done our "main" algorithm starts. We did not include all code from the main algorithm since this is not needed to understand how the algorithm works.

Listing 1: Reading audio footage

```
new Thread(null,new Runnable(){
public void run(){
        //Get recorded file from path
        File file = null;file = new
            File(Environment.getExternalStorageDirectory().getAbsolutePath()
            +"/spectometry/test"+(testnummer-1)+".wav");
        WavFile defile;
        double[] totalfft = null; //will contain the raw audio data first

        FileInputStream in = null;
        try { //Open audio file and read raw data
                in = new FileInputStream( file );
                defile = WavFile.openWavFile(in);
                Log.v("file","file opened: channels:" + defile.getNumChannels() + "
                    rate" + defile.getSampleRate() );
                totalfft = new double[(int) defile.getNumFrames()];
                defile.readFrames(totalfft,(int) defile.getNumFrames());
                defile.close();
                in.close();
        } catch (FileNotFoundException e) {
                e.printStackTrace();
        } catch (IOException e) {
                e.printStackTrace();
        }
```

After we recorded the audio and saved it to the local file storage, we retrieve the WAV file and read the raw data (Listing 1). We do all the calculations in a separate thread, so that the user interface thread is not blocked and the user will see a loading dialog. The recorded audio will be temporarily saved in an array of type double named *totalfft*, and we also use some help variables that are not shown in the explanation.

Listing 2: Initialize FFT

```
DoubleFFT_1D fft = new DoubleFFT_1D(1024);
//Create FFT algorithm instance of size 1024
Number[] magnitude = new Number[512];
final Number[] magnitudeglobal = new Number[512];
final Number[] loudest = new Number[(transformed.length / 2205)+1];
//2205 rates is exactly 0.05 seconds.
final Number[] freq = new Number[512];
int loudestfreq = 0; //var to store loudest frequency
```

```
fft.realForward(sound); //global fft to determine loudest freq
```

With the raw data we initialize an FFT class with $N = 1024$, and with this FFT instance we calculate the discrete Fourier transform of the complete audio file (Listing 2). This we use to see which frequency band is important and loudest. Next we use time windows of 0.1 seconds and an overlap of 0.05 seconds to create a power-spectrum over time. We apply for each window a Fourier transformation and add the low frequencies together.

Listing 3: FFT time windows

```
int timewindownummer = -1;
int teller = 0;
//FOR all timewindows 0.05 seconds do a Fourier transform, windows are overlapping.
    (so 0.1 sec windows, 0.05 sec overlap.)
for (int timewindow = 0; timewindow < transformed.length - WINDOW_SIZE; timewindow
    += WINDOW_SIZE/2)
{
        timewindownummer ++;
        double[] copytrans = new double[WINDOW_SIZE];
        //Copy window in copytrans array
        for (int j=0; j<WINDOW_SIZE; j++)
        {
                copytrans[j] = transformed[j + timewindow];
        }

        fft.realForward(copytrans); //perform FFT on window
        // Calculate the magnitudes of each frequency and add
        //them to the loudest array. Not only the loudest frequency is
        // added to reduce noise
        try {
                loudest[teller] = 0;
                for (int i=0; i< 512 ; i++){
                        magnitude[i] = Math.sqrt(copytrans[i*2]*copytrans[i*2] +
                            copytrans[i*2+1] * copytrans[i*2+1]);
                        freq[i] = (fs * i) / 1024;
                        loudest[teller] = loudest[teller].doubleValue() +
                            magnitude[i].doubleValue();
                }
                if (loudest[teller].doubleValue() < treshold)
                {
                         loudest[teller] = 0; //skip data that we don't care about
                         teller--;
                }
                teller++;
        } catch (Exception e) {
                e.printStackTrace();
        }
}
```

After calculating all overlapping windows (Listing 3), we perform a polynomial regression on the outcome. This regression, as explained earlier, is done to reduce noise and

approximate the function expected from a clinical spirometer. We then calculate FEV_1 and $FEF_{25-75\%}$ by adding the corresponding function values together (Listing 4).

Listing 4: Measurements

```
j=0;
start = false;
for (int i=0;i<20 && j<yvalues.length;j++){
        if (yvalues[j] != null && yvalues[j].doubleValue() > 0)
                start = true;
        if (yvalues[j] != null && start){
                fev1 += yvalues[j].doubleValue();
                fev1array[i] = yvalues[j].doubleValue();
                fev1timearray[i] = j * 0.05;
                i++;
        }
}
//calc fev25-75
double totalloudest = 0;
for (int i=0;i<yvalues.length;i++){
        if (yvalues[i] != null)
         totalloudest += yvalues[i].doubleValue();
}
double temp = 0;
double fev2575d = 0;
double fevtime = 0;
final Number[] fev25array = new Number[yvalues.length];
final Number[] fev25timearray = new Number[yvalues.length];

for (int i=0;i<yvalues.length;i++){
        if (yvalues[i] != null)
        {
                temp += yvalues[i].doubleValue();

                if (temp>totalloudest/4 && temp<totalloudest/4*3){
                 fev2575d += yvalues[i].doubleValue();
                 fev25array[i] = yvalues[i].doubleValue();
                 fevtime += 0.05;
                 fev25timearray[i] = i*0.05;
                }
        }
}
```

As last step for our results, we apply a linear transformation model that calibrates our app's output to the output of a spirometer (Listing 5).

Listing 5: Mapping

```
//linear model that improves our accuracy.
//is a mapping to test data gathered using another spirometer.
fev1 = 0.000128 * fev1 + 0.71;
```

```
//create final vars to use in UI
final double fevfinal = fev1;
final double fev2575final = ((0.000128 * fev2575d + 0.71) / fevtime);
```

After this transformation the data are ready to be shown to the user and several plots are made. Based on the two most important parameters, FEV_1 and $FEF_{25-75\%}$, we show whether the user is "healthy", in an "uncertain condition" or most likely "not healthy". This is done, in our case, for now, by putting basic thresholds and is subject for further research.

6 Experiments

In the following subsections we use our application to further improve it by means of calibration and the standardization of some enviroment variables. We also test our application by comparing the results to the results of a clinical spirometer.

6.1 Standardization

In order for the app to run correctly and to parse the sensor's output we need to standardize the way users use the application. For example, when a user breathes into the microphone with a four centimeter distance the output will be much higher than if the user uses a ten centimeter distance to the microphone. The user can also breath on the screen or at the side of the smart phone instead of directly into the microphone, and all these variables may affect the outcome of the test and have to be taken care of. To understand how important these variables are we did some tests to see the difference. For each angle at the smart phone and for several distances we did three tests, and for each three tests we took the average. As we can see from Table 1, the most stable side to blow is the front side. So we let the user blow at the screen of the smart phone, holding the phone at eye height. The advantage of this position is also that the user is able to see the screen and that he/she knows which buttons to press.

Angle	Distance (in cm)	AVG Outcome	Standard deviation
mic	30	7.53	0.99
right	30	2.22	0.78
left	30	1.99	0.73
front	30	4.31	0.20

Table 1: Test results for different blow sides

Next we did tests for the front side of the phone but with distances ranging from forty down to ten centimeters (Figure 11). For each distance six tests by one person are performed. More than forty centimeters is not preferable since the user needs to hold the phone and preferably still be able to read. We can see that if you come too close to the screen the results become less stable; this can be explained by the fact that the further you go from the phone, the less important is the angle, since the air that hits the phone will hit a larger area.

Angle	Distance (in cm)	AVG Outcome	Standard deviation
front	40	2.64	0.72
front	35	3.18	0.99
front	30	4.48	0.33
front	25	4.82	0.98
front	20	5.44	0.91
front	15	6.51	1.56
front	10	8.67	1.08

Table 2: Test results for different blow distances

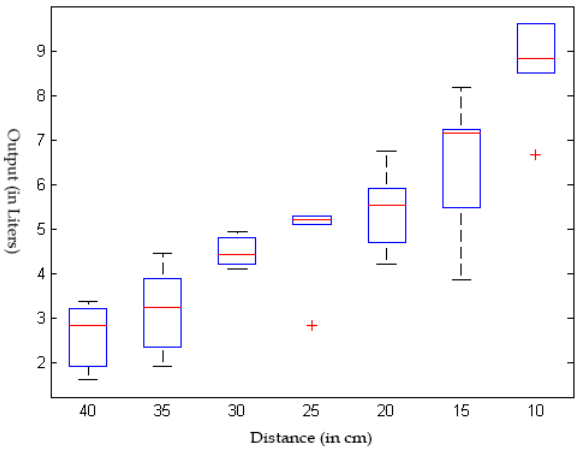


Figure 11: Boxplot of the test results (Table 2) per distance

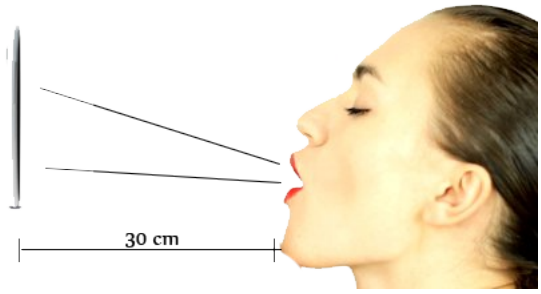


Figure 12: Standardized use of application

The thirty centimeter distance will be chosen for the rest of our experiments and calibration (Figure 12).

Test	Radboud	SpiroApp
1	3.98	4.31
2	1.86	1.30
3	0.71	0.14

Table 3: Calibration test results

6.2 Calibration

The next step in the development process is the calibration of the application. The sensor output of the smart phone (microphone) needs to be calibrated in such a way that it works like a spirometer sensor. This done by using a mobile spirometer that is used in the Radboud project [16]. The spirometer is used three times on a person and the average value of the tests is compared to the average value of using our application three times (Table 3).

We notice that the high values are a bit too high and the lower values are a bit too low for our app. This suggests a linear transformation model that we can calculate using our test values. Notice that the results from the tables are the output values of the microphone's amplitudes, divided by 6000, which was done as the first calibration action. The transformation model is made by the original numbers, so the above times 6000:

$$y = 1.28 \cdot 10^{-4}x + 0.71$$

Including this transformation into the program code, leads to more accurate results.

To ensure better results, a group of test patients and a control group should be used to test the application. However, it is already clear that it is possible to create an application that can measure the lung function of a patient by only using the microphone that is built-in to the smart phone.

6.3 Results

We tested the application with a number of healthy people. The tests are done by performing a spirometry test three times with a Clinical certified spirometer (*microlive* PEV/FEV1 meter) and three times with our SpiroApp. The averages of these test sets are then used to compare the application's performance.

The results of the tests are in Table 4 and Figure 13. One can see that the error of our application does generally not exceed 0.5 liters (Figure 14), with the exception of three small outliers and one bigger outlier. This is acceptable for self-care and to use as indication. For a more precise measurement the patient should always take a test by using a clinical spirometer.

Test results are highly dependent on the standardized way of testing. When the distance to the microphone is different, the results will be different. Due to time limitations and the scope of this project we were only able to test one COPD patient and one patient with Asthma. The results for these tests are given in Table 5, and these tests are also included in both Figures 13 and 14.

While performing tests with the COPD patient, we noticed that testing by using the application on the phone, was much more easy for the patient than testing with the clinical

Test	Clinical Spirometer	SpiroApp	gender	age
1	4.30	4.02	male	23
2	3.23	2.76	female	24
3	4.73	4.57	male	29
4	5.13	5.36	male	30
5	3.77	3,21	male	21
6	2.19	2.62	female	31
7	2.13	2.33	female	64
8	3.40	3.71	male	58
9	3.68	4.31	male	29
10	3.98	3.79	male	24
11	4.49	5.95	male	24
12	4.78	5.48	male	39

Table 4: FEV_1 results for our application and a clinical spirometer

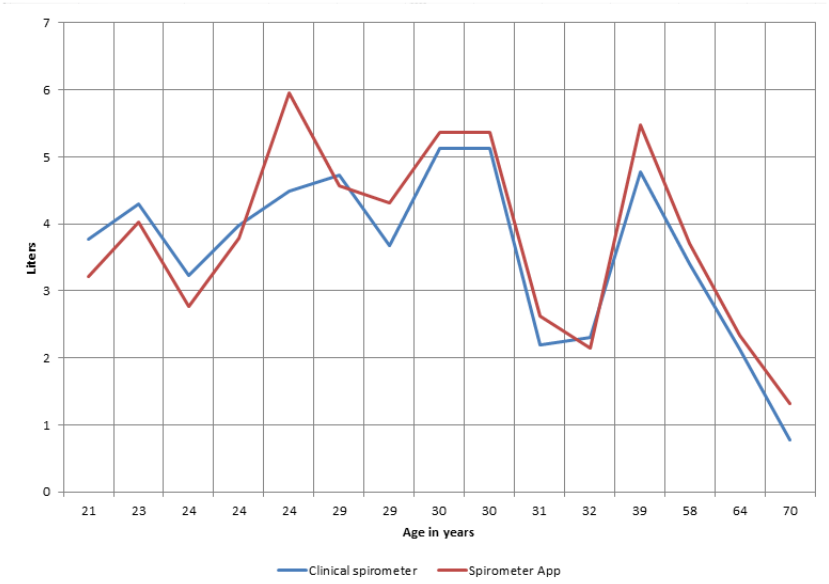


Figure 13: Test results per age

Test	Clinical Spirometer	SpiroApp	gender	age	disease
1	0.78	1.32	male	70	COPD
2	2.30	2.14	female	32	Asthma

Table 5: Test results

spirometer. This is due to the fact that for the smart phone application, the patient does

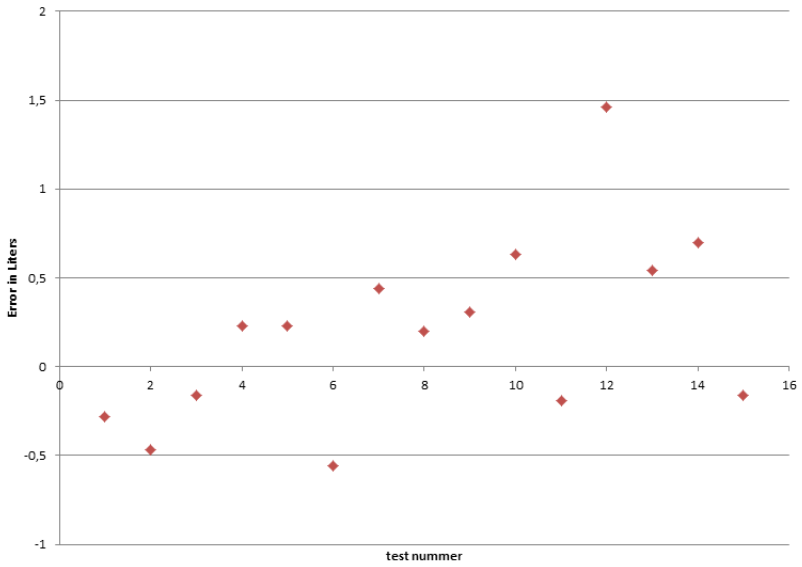


Figure 14: Error of tests in liters

not need to blow in a tube. The tube gives a certain resistance that the patient has to deal with.

7 Related work and limitations

This work is inspired by two projects, the Radboud University project [16, 17] as we already discussed in Section 3 and the *SpiroSmart* project from the University of Washington [6]. *SpiroSmart* is an application for the iPhone in which they claim to do a spirometer test. The goal of this thesis was to verify that this is indeed possible.

Our application, able to do a spirometry test, has a few limitations that spirometers at hospitals do not have. First of all, the application can only measure air that is exhaled, since inhaling is inaudible. Inhaled air, however, is almost never used to diagnose a patient. What we see in tests, noise is not really a problem unless it is very loud and in a low frequency range. Overall, patients should be able to perform a test, while people are talking, so giving instructions during a test should still be possible, though not preferred.

The application has been tested and is made for the HTC One X with Android 4.1.2. The application is tested on several other devices and it is clear that the results vary per device. The results are dependent on the microphone quality of the phone, the housing of the phone, the size of the phone and also the software that handles the audio on the phone. These dependencies can partly be solved by programming, by handling different sample rates of audio quality for instance, but some of the dependencies can only be taken care of by testing it on the specific device and calibrating the application again. This is outside the scope of this project, but would be subject of further research if the application comes to

the market. This application is *not* a replacement of a clinical *ERS certified device*. To get ERS certification we need to prove, among other things, that the error rate never exceeds 3% [10]. This is probably not feasible.

8 Conclusion and further research

In conclusion, an application has been created to test lung functions by using only the microphone of a smart phone. The application works reasonably well and can be used in home care. This *mHealth* application uses the built-in microphone to catch the sound of the patient's breath. When the recording stops, the application calculates several lung parameters and these are shown to the user. The application has been tested with some health subjects and this worked great. The outcome of the application is compared to the outcome of the application made at the Radboud University (Section 3) and to the outcome of a clinical certified spirometer. We have shown that the application works and that it can be used for a patient's daily self check. This a very cheap solution in comparison with mobile spirometers and can be used on a wide scale.

In comparison with the application from *SpiroSmart* [6], our application has some advantages and differences. *SpiroSmart* can not be used when people are talking, as stated in their article; our application, however, has no problem with this kind of noise. Making it possible to perform a spirometry while a doctor or relative speaks encouraging words. The *SpiroSmart* application is designed to work at "arm-length", meaning that the user has to keep the smart phone at arm-length distance to perform the test. This distance is in our oppinion not good since the distance will be different for each user. Our application works with a distance set to 30 centimeters. The problem with this distance is that it needs a ruler or other distance measurement device to be correct. Another difference between *SpiroSmart* and our application is that *SpiroSmart* does the calculations in real time. This has the advantage that the patient sees his results in real time. Our application calculates the results after the test is performed in less than a second, which is strictly not in real time, but has the advantage of using more powerful, compute-intensive algorithms that may lead to better results. The results from the *SpiroSmart* shown in their article are comparable with the results from our application and are between -30% and 30%. By personalizing *SpiroSmart*, the article showed an error rate below 10%, and for our application this is probably possible as well. For this a good standardization and calibration procedure is vital. Furthermore, we have proven that it can be extremely useful to use sensors and more specific the sensors offered by mobile devices to perform tests and algorithms normally run on different types of sensors. In this way we can create all in one solutions that would otherwise be very costly.

Sensors in smart phones or other multi purpose devices can be used on a large scale in a cost effective way and are ideal for home and internet health care. With more then 400 million Android devices in the world a spirometry application can be sold on the market and everybody would be able to test their lungs. With such massive scale spirometry, we can in the near future threat lung diseases fast and detect them in an early stage. We have proven that a sensor can be used to do measurements the sensor is not intentionally made for. The steps needed to implement such program are distinguished and explained. Something not many people expected is proven; a spirometry test can be performed by using only the built-in microphone of a smart phone!

The app needs further development. A group of test patients, both healthy and with COPD, is needed to test and calibrate the app even more. This is also needed to see if a patient exhales in a way that the application still understands. The app can be improved by keeping history data of the patient and the patients gender/name and doctor information in case of emergency. An online portal can be used to monitor the patient from a distance. This is also done in the Radboud University project and can be used again for this app. Furthermore, a questionaire or other measurements done by the smart phone can improve the diagnosis tool even more. Other ways of using smart phone sensors might be examined in order to create even more mHealth apps or apps used in other fields, like data security by using the camera to perform an iris scan. The possibilities seem to be endless and researching these options might provide us in the near future with an all in one, multiple disease diagnosing, multi tool.

Acknowledgements

I would like to express my gratitude to Peter Lucas and Walter Kosters, my research supervisors, for their useful critiques of my work. I would also like to thank Bas Dirkson for useful tips and an open minded conversation. Further more, I would like to extend my thanks to everyone that helped me testing the application. Special thanks to Tom Groentjes, who worked double hours at our company to give me the time I needed finishing my project. Thanks as well to trimethylxanthine, that kept me awake in the hours most needed. Finally, I would like to thank my fiance, Eftychia Thomaidou, for her love, support and encouragement during my project.

References

[1] J. Buffels, J. Degryse, J. Heyrman, and M. Decramer. Office spirometry significantly improves early detection of COPD in general. *Chest Journal*, 125(4):1394–1399, 2004.

[2] B.R. Celli, W. MacNee, A. Agusti, A. Anzueto, B. Berg, A.S. Buist, P.M.A. Calverley, N. Chavannes, T. Dillard, and B. Fahy. Standards for the diagnosis and treatment of patients with copd: a summary of the ATS/ERS position paper. *European Respiratory Journal*, 23(6):932–946, 2004.

[3] J.W. Cooley and J.W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation journal*, pages 297–301, 1965.

[4] K. Cuppens, L. Lagae, B. Ceulemans, S. Van Huffel, and B. Vanrumste. Detection of nocturnal frontal lobe seizures in pediatric patients by means of accelerometers: A first study. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6608–6611. IEEE, 2009.

[5] J.F. Epperson. *An introduction to numerical methods and analysis*. Wiley-Interscience, 2007.

[6] E.C. Larson, M. Goel, G. Boriello, S. Heltshe, M. Rosenfeld, and S.N. Patel. Spirosmart: Using a microphone to measure lung function on a mobile phone. 2012.

[7] P.J.F. Lucas, L.C. van der Gaag, and A. Abu-Hanna. Bayesian networks in biomedicine and health-care. *Artificial Intelligence in medicine*, 30(3):201–214, 2004.

[8] A.W. Martinez, S.T. Phillips, E. Carrilho, S.W. Thomas III, H. Sindi, and G.M. Whitesides. Simple telemedicine for developing regions: Camera phones and paper-based microfluidic devices for real-time, off-site diagnosis. *Analytical Chemistry*, 80(10):3699–3707, 2008.

[9] W.D. McArdle, F.I. Katch, and V.L. Katch. Exercise physiology, 5th ed. 2001.

[10] M.R. Miller, J.A.T.S. Hankinson, V. Brusasco, F. Burgos, R. Casaburi, A. Coates, R. Crapo, P. Enright, C.P. Van Der Grinten, P. Gustafsson, et al. Standardisation of spirometry. *European Respiratory Journal*, 26(2):319–38, 2005.

[11] John Neter, William Wasserman, Michael H Kutner, et al. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.

[12] P. Neval and M.E.H. Ismail. *Orthogonal Polynomials: Theory and Practice:[Proceedings of the NATO Advanced Study Institute on Orthogonal Polynomials and Their Applications]*, volume 3. Springer, 1990.

[13] G.A.F. Seber and A.J. Lee. *Linear regression analysis*, volume 936. John Wiley & Sons, 2012.

[14] E.A. Spriggs et al. The history of spirometry. *British Journal of Diseases of the Chest*, 72(3):165–166, 1978.

[15] S. Stanojevic, A. Wade, J. Stocks, J. Hankinson, A.L. Coates, H. Pan, M. Rosenthal, M. Corey, P. Lebecque, and T.J. Cole. Reference ranges for spirometry across all ages a new approach. *American Journal of Respiratory and Critical Care Medicine*, 177(3):253–260, 2008.

[16] M. Van Der Heijden, B. Lijnse, P. Lucas, Y. Heijdra, and T. Schermer. Managing copd exacerbations with telemedicine. *Artificial Intelligence in Medicine*, pages 169–178, 2011.

[17] M. van der Heijden, P. JF Lucas, B. Lijnse, Y. Heijdra, and T. Schermer. An autonomous mobile system for the management of copd. *Journal of Biomedical Informatics*, pages 1–11, 2013.

[18] J.B. West. *Pulmonary pathophysiology: The essentials*, pages 1–12. Wolters Kluwer, 7th edition, 2005.

[19] J. Zieliński and M. Bednarek. Early detection of COPD in a high-risk population using spirometric screening. *Chest*, 119(3):731–736, 2001.