Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Low Power Gesture Recognition

Semester Project

Student: Severin Mutter

Supervisors:
Pascal Bissig
Prof. Dr. Roger Wattenhofer

March 2014 – June 2014

Abstract

This thesis describes the concept and the design of a small, handheld and low power prototype for gesture recognition to control functions of a mobile phone. The motion tracking is done with a sensor chip which includes a 3-axis accelerometer and a 3-axis gyroscope. Over a Bluetooth connection, the control chip on the device sends the measured data to the mobile phone. We compare two algorithms to perform gesture recognition, the dot product algorithm and the Hidden Markov Model.
The outcome of this project is a working prototype which can be used for gesture recognition.

TABLE OF CONTENTS

1. INTRODUCTION

As the size of mobile phones is growing the last years, possibilities to control them without pulling them out of the pocket are in great demand. If for example someone only wants to change the song or to stop the music, doing this without the phone in the hand would be much easier.

Tools like the headphones which can start and stop a phone call or control the music player are already on the market. Other examples are smart watches that can control messages, phone calls and more.

For this Semester Project, we tried to find out a method to control our device with a much smaller gadget like a key chain with an integrated Bluetooth platform. We focused on a low power chip as it could not only be used to control mobile phones but also could be integrated in other devices without the need of any other additional energy than the battery. For the future, a combination of the tool with televisions or laptops could be interesting.

The concept is that we measure the movement of this gadget via sensors and then to send this data over a Bluetooth connection to the mobile phone as can be seen in Figure 1.0:

Fig. 1.0: Concept of the project: The sensors measure the movement of the key chain and the data is sent to the mobile phone over a Bluetooth connection.

The data we get will be processed and gestures then get recognized. With these gestures it should be possible to control functions of the mobile phone. For example, we think it would be interesting to unlock the mobile phone by doing a gesture or if by doing a specific movement a phone number would be automatically called.

First the report focuses on the concept and the design of the hardware. After that, the function of the gesture recognition is described and the results of their implementation are discussed. At the end of the report, there is an outlook for future work on this project.

1.1 Contributions

One contribution of the project is the design of a prototype with low power components on it. A Bluetooth Low Energy system-on-chip is used in combination with a sensor chip. The data of the sensor measurements will get recorded and the gestures get recognized by using an algorithm. The two algorithms applied for this task are the dot product algorithm, which is implemented in a mobile phone application, and the Hidden Markov Model. We compare these two and discuss the results of their gesture recognition.

2 HARDWARE

For our system, we need suitable hardware. As the prototype should fit into a key chain, the sensor chip and the Bluetooth platform should be as small as possible. One other point is that the hardware should have low power requirements. In the next chapters, our decision to use the sensor MPU6050 and the NRF51822 is described. Furthermore, the concept, the design and the implementation of the prototype are shown.

2.1 SENSORS

There are many different possibilities to measure the motion and the orientation of a device. Today gyroscopes and accelerometers are often used in devices like tablets or mobile phones. For the gesture recognition prototype we use these two sensors in combination for tracking the motion of the prototype, so in the following chapters we will have a closer look at them. With these two sensor types and both of them with 3-axis, six degrees of freedom (DOF) are covered.

Tracking gestures with these sensors can be done by calculating the motion and orientation with the measured movements of the device. A global position is not needed, as a gesture depends only on the change and not on the exact global starting point. If we record the sensor data over time, we can recognize this motion as a certain gesture.

2.1.1 GYROSCOPE

Gyroscopes are able to measure angular velocity. They are used for gaming, navigation, image stabilization and other applications. First they were only available as mechanical-inertial spinning devices consisting of rotors, axles and gimbals. But nowadays they can be bought as Micro-Electro-Mechanical Systems (MEMS) in very small chip sizes. In such devices, the Vibrating Structure Gyroscope (CVG) is implemented. A mass that oscillates at high frequency is used in such a VSG and not a spinning wheel. The secondary vibration caused by the Coriolis force is then measured by a pickoff. [1]

Although gyroscopes have many advantages like low jitter, high sampling frequency readout and low measurement noise, they have the big disadvantage that when we integrate the angular rates the result will drift. So if they are lying still on the table, due to the drift, the angles will change over time. The drift is caused mostly by the gyro bias, which is unknown at the beginning. Later the gyro bias instability is the main reason for it. Resets of the sensor after a time are therefore necessary. Also algorithms which take the drift into account or highly accurate sensors can help. [1]

For our system this becomes only a problem, if we measure long gestures. For gestures which take around five seconds, the drift of the gyros does not affect the possibility to track the motion of the device.

2.1.2 Accelerometer

The measurement principle of modern acceleration sensors is based on a mass attached to a spring with constant k. If a force is attached to the mass, the displacement of the mass from its center position is measured by using Newton's and Hooke's law:

Newton's law: $F = ma$

Hooke's law: $F = kx$

Acceleration: $a = \frac{kx}{m}$

A problem is that the spring only behaves linearly around its zero point. Therefore a closed-loop system is used, which has a forcer and an electromagnetic displacement pickoff. The device measures the force which is necessary to hold the mass at the same place. Today it is possible to build accelerometers as so called Micro-Electro-Mechanical Systems (MEMS), which allows very small chips. [1]

One drawback of an accelerometer is that the gravitational force results in the measurement of the gravitational acceleration. As this acceleration is not causing any motion relative to the earth's surface, it is not useful for recognizing gestures and leads to wrong calculations. To get rid of this problem it is useful to know the orientation of the device. With this information, the gravitational part can be substituted from the measured data and only the wanted acceleration caused by motion is left.

2.1.3 Sensor MPU6050

We decided to use a combination of accelerometers and gyros, as we hoped to overcome the drawbacks of the accelerometers and the gyros by combining them and to cover all degrees of freedom. We found the chip MPU6050 from "Invensense", which includes a 3-axis gyroscope and a 3-axis accelerometer. The chip has an integrated Digital Motion Processor (DMP) which is capable of 9-axis "MotionFusion" algorithms. This means that it is possible to get processed and calculated data out of the chip, which already took into account the gravitational part of the acceleration and substituted it to get the part of acceleration that we want. [2] This would simplify the gesture recognition application, as we would not have to deal with raw sensor data at all.

The MPU6050 is designed for low power and high performance requirements, so fits very well for our purpose. As it uses the MEMS technology, the chip is small and cheap, both characteristics required four our project. [2]

For the first steps with this device, we were able to use a board from "Sparkfun" and connect it with our Bluetooth platform.

2.2 System-on-a-chip: NRF51822

As Bluetooth platform we use the System-on-a-chip NRF51822 from "Nordic Semiconductor". It supports Bluetooth Low Energy (BLE) protocol stacks. Also it has programmable I/O-pins which we use to communicate over the TWI-interface with the sensor chip MPU6050. We found the device well suited as it comes with a given protocol stack called "SoftDevice", which can be downloaded and programmed on the chip. It then can be used for the whole BLE communication. Nordic Semiconductor also supports software for the TWI-interface. This helps us to configure the sensors and to get the data out of them. [3]

2.3 PROTOTYPE

The concept of the prototype was to build a small print with the NRF51822 and the MPU6050 on it. As “Nordic Semiconductor” offers reference designs, we tried to stick close to their designs. The basic organization can be seen in Figure 2.0. We need one crystal for the Bluetooth communication, a JTAG to program our custom board and a battery to power everything. The communication with the MPU6050 works over the TWI protocol. The two interfaces will be described in more details in the Section 2.3.1.

A picture of the final prototype with the components on it is shown in Figure 2.1

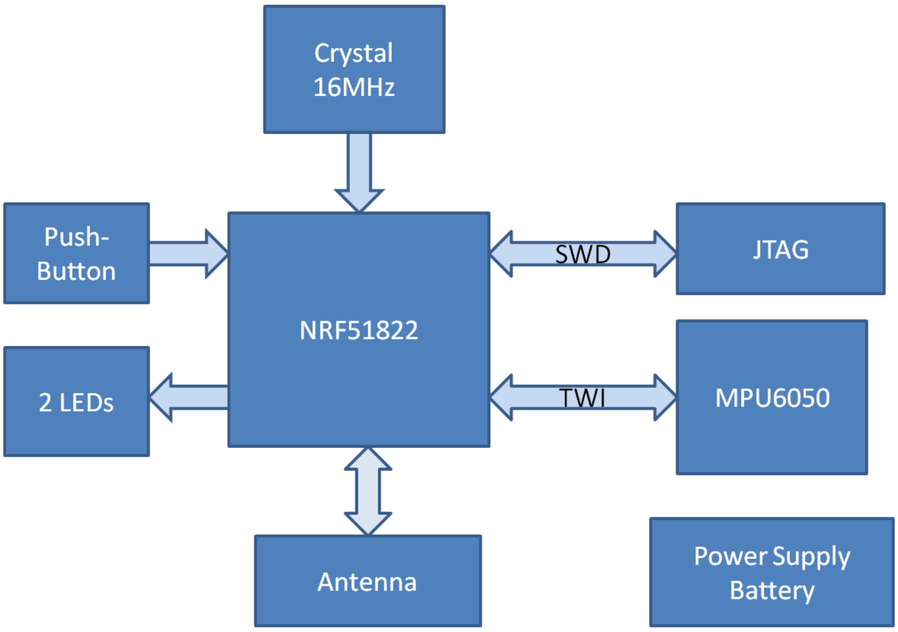

Fig. 2.0: Block diagram of the prototype

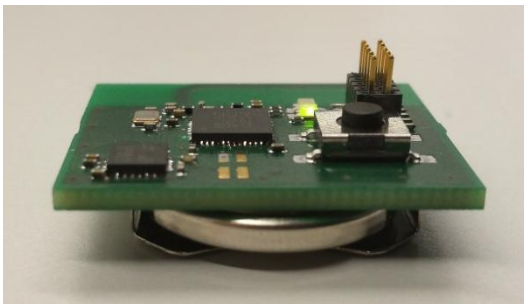

Fig. 2.1: The final prototype with the NRF51822 in the middle, the MPU6050 on the left side, the button, the LEDs and the connector to program the custom board. Below the board is a 3 V battery.

2.3.1 SCHEMATICS

The schematic is close to the one of the Evaluation Kit. The two LEDs are for testing if the device is working and if the TWI interface with the MPU6050 is initialized correctly. The button starts the Bluetooth communication with the mobile phone. One difference is that we only use one external crystal with 16MHz to keep it simple. The other crystal can be programmed internally.

A TWI Interface is used for a serial data transmission in asynchronous mode and for data transmission between two devices. It needs the two signals Serial Data (SDA) and Serial Clock (SCL) for the communication. One device is the master, the other is the slave. [4] In our case the master is the NRF51822 and the slave is the MPU6050.

The circuit of the antenna is exactly like the one of the Kit. For the connections of the MPU6050 we used the reference designs provided by the producer of the chip "InvenSense" and connected the necessary signals to the NRF51822. As Power Supply we only need one 3V battery.

To program our custom board we use the J-Link LITE CortexM from the company "Segger". Therefore we have a Serial Wire Debug interface over a 10-pin connector. The Serial Wire Debug interface works actually over the two pins SWDIO and SWCLK and is compatible with all ARM processors. [5]

For further developments, four more General Purpose I/Os are conducted to a connector.

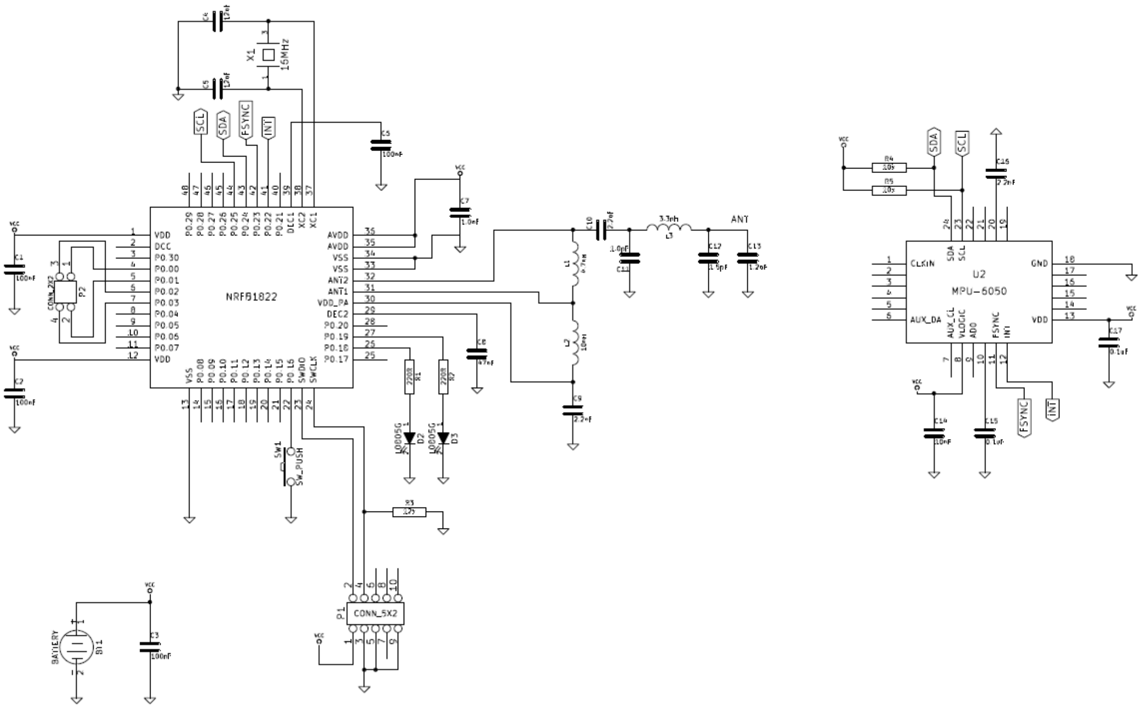

Fig. 2.2: Schematic of the prototype with its three parts, the Bluetooth platform NRF51822, the sensor MPU6050, and the power supply.

2.3.2 LAYOUT

For ensuring that the antenna is working correctly, we designed this part of the layout as close as possible to the reference design. A conductive path serves as antenna as can be seen in Figure 2.3. Under it is a restricted area where nothing else is. If for example the battery would be below the antenna, the electromagnetic radiation in this direction would decrease and this could lead to bad Bluetooth connections between the prototype and the mobile phone.

The size of the design is given by the size of the antenna and the battery, which is on the bottom side. For good connectivity of the ground planes some more vias connect them.

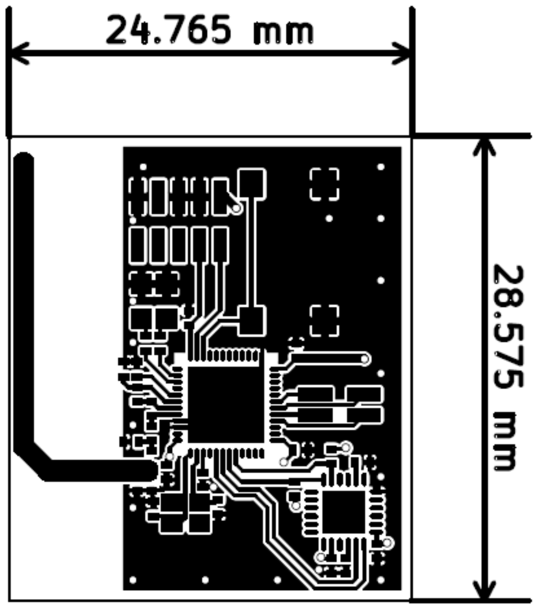

Fig 2.3: Layout Copper top with a ground plane and the antenna on the left side, where no components and no conductive paths are placed to reach a high performing electromagnetic radiation in all directions

2.3.3 *SOFTWARE*

The software on the prototype is based on the provided example codes of “Nordic Semiconductor”. The task of the program is to communicate with the MPU6050, to receive the data from it, to prepare the data for sending and to handle the whole Bluetooth connection with the mobile phone on the client’s side.

Our sensor data is transferred in a Bluetooth service. This service contains two characteristics, one for the gyro data and one for the accelerometer data, both with the timestamp of the measurement added by the control platform. These values are given to the GATT server, which is on top of the Bluetooth low energy protocol and stands for Generic Attribute Protocol. As notification is on for both characteristics, the master is informed whenever a value at the slave is changed. In our case the master is our mobile phone and the slave is the motion tracking system.

3. Gesture Recognition

Gesture recognition tries to identify human gestures with the help of mathematical algorithms using a computing device. [6] In our case, we try to classify gestures by analyzing the measured data. If we find out that the device made a specific gesture, this should lead to a certain answer of the mobile phone, which is our computing device.

3.1 Algorithms

To test algorithms for our gesture recognition, we need first of all a set of a gesture with many samples. With some of these slightly different gestures, a training set is chosen and given to the algorithm. At this point, the algorithm should know how a certain gesture should look like and therefore have built a model of the gesture. The rest of the gesture instances are then used as a test set. This means that we give the algorithm with its model the test set to check if it will recognize the gestures correct.

The two algorithms we used, the dot product algorithm and the Hidden Markov Model, are explained in the next chapters.

3.1.1 Dot product algorithm

Our simplest gesture recognition algorithm makes use of the dot product of two vectors. The normalized dot product is defined as follows:

$$\frac{x \cdot y}{\|x\| \cdot \|y\|} = \frac{x_1 * y_1 + x_2 * y_2 + \cdots}{\|x\| \cdot \|y\|} = cos(\theta)$$

It can be used to calculate the angle between two vectors. If two vectors point in the same direction, the angle between them is small and therefore the result of the dot product is close to 1. Otherwise, if for example, the vectors show in orthogonal directions, the dot product would become 0. For an obtuse angle its result would become negative. [7] In our case, one of the vectors is the gesture which we have to test and the other one is from our model set of the gestures. We calculate this for the coordinates x, y and z of the accelerometer data or the gyro data and then add the results. With the result, we can decide if this gesture was made, because then the vectors should be quite similar and the dot product should result in a number close to 3.0.

3.1.2 Hidden Markov Model

Hidden Markov Models (HMM) are a common choice to model and classify time signals. A HMM is a statistical model and based on the Markov Chain. The significant advantage over the dot product algorithm is that the HMM can as well deal with gestures which differ in duration and shape. [8] As human gestures can differ in time, this is a huge benefit in gesture recognition. When a gesture is made slower or faster than the model, the gesture can still be recognized.

We use a given HMM library in MATLAB and train it with a set of gestures out of our records. Then we use this model with our test set and calculate the likelihood that a test gesture corresponds to the given ones. As in the dot product algorithm, we can choose if we compare the accelerometer data or the data of the gyros.

3.2 Results

To evaluate the functionality of our two algorithms for gesture recognition, we test both under the same conditions with the given test sets and compare them.

To test our system, we measure the sensor data with 20 Hz and estimate that a gesture takes five seconds at most. So we get 100 values for each sensor type per record. As gestures we have chosen a lying eight and a circle as can be seen in Figure 3.0:

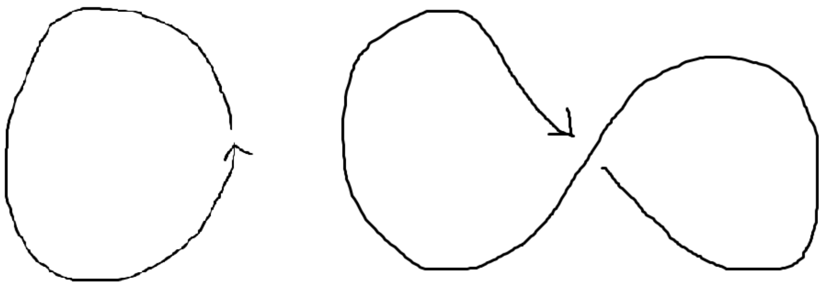

Fig. 3.0: The tested gestures, a circle and a lying eight. The starting point is the open end without an arrow and the end point is the arrow.

We recorded 30 samples of each gesture. The model set of the algorithms holds five of them. The other 25 samples are then tested with both model sets. Additionally we recorded ten instances of imprecise gestures of the two and test them as well with the models.

3.2.1 Gesture recognition with the dot product algorithm

The gesture recognition with the dot product can recognize the two gestures of our test. If we test a set of gestures with the two model sets, the outcome is always the right gesture. This means that a circle is never interpreted as a lying eight and vice versa. The calculation of the dot product delivers numbers between 1.9 and 2.9.

The imprecise lying eight and the imprecise circle are again interpreted right. For inexact or wrong gestures this is maybe not desired, as the user is asked to do the gestures close to the exact ones. This can be achieved by inserting a threshold at which the gesture is still recognized. If the result of the dot product is below this threshold, the gesture will not be recognized. In Figure 3.1 and 3.2, it is shown depending on the threshold, how many imprecise gestures would be recognized as true ones and how many precise gestures would be taken as false ones. To get a good value for both failure cases, the value of the threshold should be chosen where the diagonal of the axes crosses the blue curve. This cross point gives similar small values for both rates.

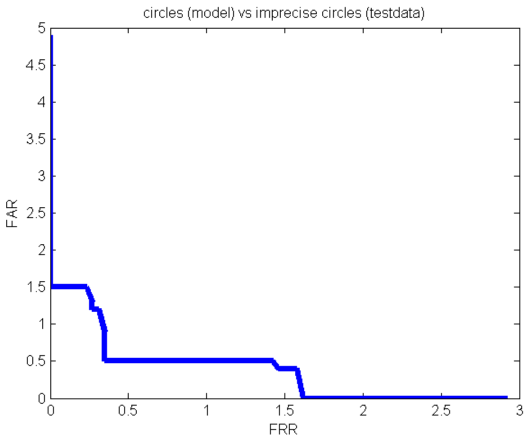

Fig.3.1: Diagram of the FRR-FAR-function depending on the threshold value of the circle gesture. FAR stands for False Accept Rate and describes how many false gestures are accepted. FRR means False Reject Rate and shows how many right gestures are wrongly not accepted.

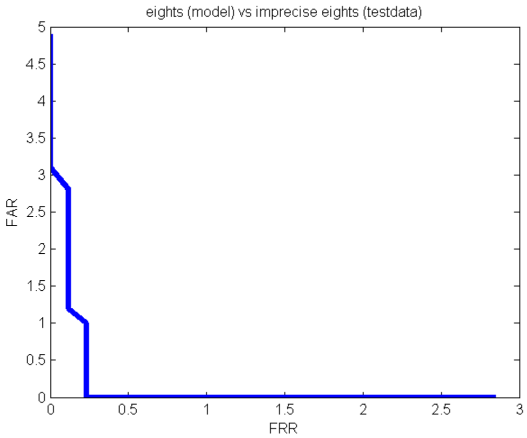

Fig. 3.2: FRR-FAR-function of the lying eight gesture for the dot product with imprecise gestures as false ones and precise gestures as true ones

3.2.2 Gesture Recognition with the Hidden Markov Model

The gesture recognition with the HMM results as the one with the dot product in the right gesture, if the task is only to choose between the two gestures. As can be seen in Figure 3.3 for the circle and 3.4 for the lying eight, it is possible to reach a low False Accept Rate and as well a low False Reject Rate with a certain threshold. The threshold should be chosen where the rates are similar. It can be found as cross point of the diagonal of the axes and the blue curve. These diagrams show the outcome, when we use the model set with the test set of the other gesture.

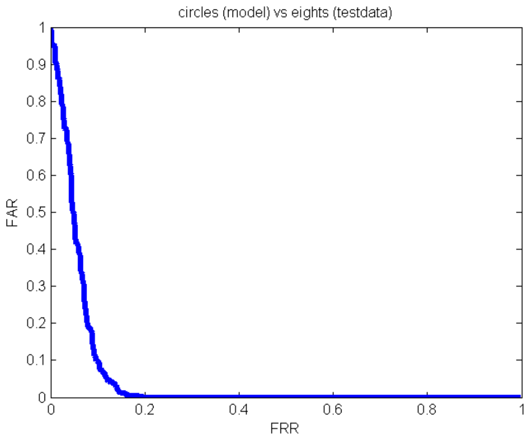

Fig. 3.3: FRR-FAR-function of the circle gesture for the Hidden Markov Model with the test data from the lying eight.

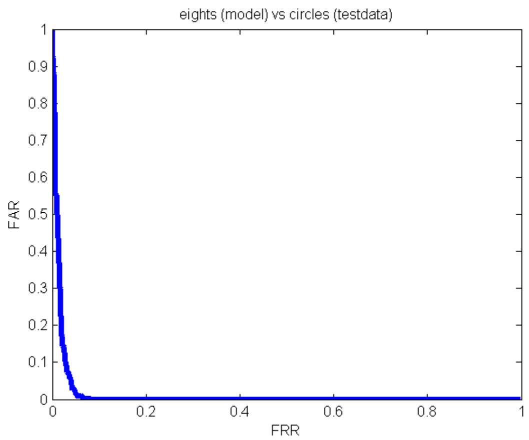

Fig. 3.4: FRR-FAR-function of the lying eight gesture for the Hidden Markov Model with the test data from the circle.

As for the dot product, if it is wanted that a gesture has to be done exactly, we can compare the imprecise gestures with the models, which can be seen in Figure 3.5 and 3.6. They show that it should be possible to distinguish between an exact gesture and an imprecise one with low False Reject Rate and low False Accept Rate.

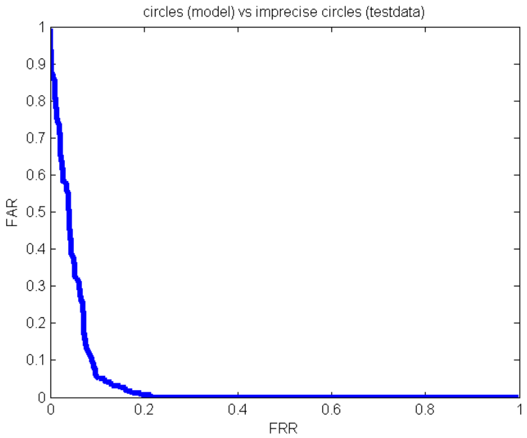

Fig. 3.5: The FRR-FAR-function of the circle gesture when the imprecise circle is used as test data.

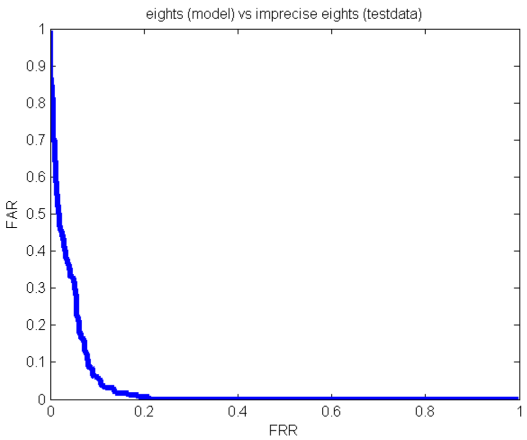

Fig. 3.6: The FRR-FAR-function of the lying eight gesture when the imprecise lying eight is used as test data.

3.2.3 Discussion

For the first tests, we used the accelerometer data for the two algorithms. The results were not as we hoped. Even with only two different gestures, the outcome was sometimes wrong. One reason for the bad results was that in the case of the accelerometer, the gravity plays a big role. We tried to hold the device stable in its orientation, but a small change leaded to big differences of the measured data.

The results when we used the data of the gyros were more robust against small differences in the gesture samples and the whole gesture recognition worked with both algorithms. The dot product algorithm is interesting for implementing on the prototype as it is simple. On the other hand, the HMM is suited for gestures which can differ in time. If the task of the device is only to choose between different gestures, the gesture recognition would work perfectly with both algorithms.

The tests showed also that it could be possible to use the device as an authentication tool, as imprecise gestures can be separated of correct gestures. So if the user defines a gesture that only he can do exactly, it should work to unlock the phone with a gesture. It should then be hard for other people to repeat the gesture exactly enough that it is accepted.

4. FUTURE WORK

The work showed that it should be possible to control a mobile phone with such a motion tracking device. Surely we can think about some future work based on this one. Here are some points that we suggest for further process on the subject:

Use the DMP of the sensor MPU6050

As mentioned in chapter 2.1.3, the device MPU6050 has an integrated Digital Motion Processor. It would be really nice to use it, as the motion tracking would reach a higher accuracy and the processing of the data would get much simpler. With this option, the whole problem with the acceleration caused by the gravitational force would be gone, as the DMP can calculate the independent acceleration values by combining them with the measured gyro data.

Implement a gesture recognition algorithm on the NRF51822

At the moment the algorithms are implemented in the application of the mobile phone or in MATLAB. A next step could be to run one algorithm on the prototype. If so, less data would have to be sent to the phone than now. The App would become simpler and the Bluetooth traffic would decrease. It would even be possible to implement a Human Interface Device protocol on the prototype and therefore the App could go away completely. Examples for HID devices are the mouse and the keyboard. If we would implement the HID protocol on the device, it could be used as a keyboard and a specific gesture could stand for a key. This would allow for example to write letters in the air, which could write something on the computer or mobile phone. Also video game controllers use this protocol, so a combination with our device and a video game could be interesting. [9] The prototype then could be used with every device which supports the HID.

Write an App with more functionality

Our app is for receiving the raw sensor data and for checking which gesture was made. To write an App with more functionality would be an interesting further development.

We think about an App, where the user can record his own gestures and then choose what he wants to happen when he makes this gesture by associate the gesture to intents. In Android, intents are used to open new activities, to use services, to communicate between activities or to give responsibilities to other applications. So it could be possible to navigate between activities by making gestures. [10] This would make our device a powerful tool to control the mobile phone in a personalized way. For example, the user could define a gesture which would start a phone call to a number, which could be again defined by a gesture. With this function and in combination with headphones, it would not be necessary to take the phone out of the pocket to make a phone call. Or another interesting function could be, when a gesture would automatically send a predefined message to somebody. As can be seen, we can think about many interesting functions and the App would make it possible that the user could add his own ideas of functions that he wants to use with our device.

Use the gesture recognition prototype with other devices

Our prototype can be used with any other device which is capable of Bluetooth Low Energy. As example, the board could control some functions of a laptop or a television. There are already televisions which can be controlled by human gestures. With our device, this could be possible as well.

5. BIBLIOGRAPHY

[1] ”Motion Tracking Systems”, Dominic Jud, Andreas Michel. Last Access: 26.06.2014 [online].
URL: http://students.asl.ethz.ch/upl_pdf/308-report.pdf

[2] “MPU-6000 and MPU6050 Product Specification”, Invensense. Last Access: 26.06.2014 [online].
URL: http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A-00v3.4.pdf

[3] “nRF51822”, Nordic Semiconductor. Last Access: 26.06.2014 [online].
URL: https://www.nordicsemi.com/eng/Products/Bluetooth-R-low-energy/nRF51822/%28language%29/eng-GB

[4] “I2C Interface or TWI“, EngineersGarage. Last Access: 26.06.2014 [online].
URL: http://www.engineersgarage.com/tutorials/twi-i2c-interface

[5] “Serial Wire Debug”, ARM. Last Access: 26.06.2014 [online].
URL: http://www.arm.com/products/system-ip/debug-trace/coresight-soc-components/serial-wire-debug.php

[6] “Gesture Recognition“, Techopedia. Last Access: 26.06.2014 [online].
URL: http://www.techopedia.com/definition/618/gesture-recognition

[7] “The dot product”, Math Insight. Last Access: 26.06.2014 [online].
URL: http://mathinsight.org/dot_product

[8] “Multi-touch Interaction with Gesture Recognition“, Espen Solberg Nygård. Last Access: 26.06.2014 [online].
URL: http://www.divaportal.org/smash/get/diva2:347990/FULLTEXT01.pdf

[9] “What is a Human Interface Device”, wiseGEEK. Last Access: 27.06.2014 [online].
URL: http://www.wisegeek.com/what-is-a-human-interface-device.htm

[10] “Android Intent”, Codelearn.org. Last Access: 27.06.2014 [online].
URL: http://www.codelearn.org/android-tutorial/android-intent

6. APPENDIX

6.1 TASK DESCRIPTION

The Semester Project contains the following tasks:

Define the hard- and software concept

Before starting with the building of the motion tracking device, the hardware had to be evaluated. The decision, which Bluetooth platform and which sensors should be on the prototype had to be made. The software concept was about thinking which task should be implemented on which device, on the motion tracking system or on the mobile phone. We decided to implement the algorithms on the mobile phone.

Design and build motion tracking system

After the first steps with the Evaluation Kit and the awareness that it works with the chosen hardware, the Schematic and the Layout had to be drawn. We used the open source program “KiCad” for this task and gave the board to “PCB Pool” to produce. The components had to be ordered and we did the pick and place by ourselves.

Record gesture data

With the working prototype we then started to record some gestures. For starting we made a set of 25 instances of one gesture. With this we are able to use a reference set of 5 instances and test the other 20 ones with them.

Design and evaluate gesture recognition algorithms

As algorithms, there are three different opportunities that we took into account. The easiest one is the gesture recognition with the dot product. The hardest one is the Hidden Markov Model. For this two we made statistics about how good they worked. The third one is somewhere between this two and could be interesting for future projects. It is called Dynamic Time Warping.

Implement gesture recognition algorithms on hardware prototype

It is possible to implement the gesture recognition algorithm on the hardware, but we decided for simplicity to do it on the mobile phone in java. We implemented the dot product on it. The Hidden Markov Model got tested in MATLAB.

Write companion smartphone App

The smartphone application is able to connect to our prototype, to receive the raw sensor data and to process this data. Gesture sets can be recorded and gestures made controlled with these.