Evaluation of Threshold-based Fall Detection on Android Smartphones

Tobias Gimpel, Simon Kiertscher, Alexander Lindemann, Bettina Schnor and Petra Vogel
Department of Computer Science, University of Potsdam, August-Bebel Str. 89, Potsdam, Germany

Keywords: Fall Detection, Development of Assistive Technology, Sensors-based Applications.

Abstract: This paper evaluates threshold-based fall detection algorithms which use data from acceleration sensors that are part of the current smartphone technology. Different detection algorithms are published in the literature with different threshold values. This paper presents the evaluation of 5 different algorithms which are suited for Android smartphones. In contradiction to prior work, our experiments indicate that the Free Fall detection Phase is necessary for a low False Positive Rate. Further, we present an empirical evaluation of currently available fall detection apps in the Google Play store.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

The probability of falls increases for older people. Approximately, about 33 % of older persons fall unintentionally each year (Mellone et al., 2012). Falls with the loss of consciousness are of special harm. Therefore, several groups are working on automatic fall detection, for example (Bourke et al., 2007; Sposaro and Tyson, 2009; Dai et al., 2010; Fudickar et al., 2014).

In this paper, we will only consider fall detection systems using Android based smartphones due to their widespread use and availability. Current smartphones are equipped with a tri-axial accelerometer sensor which periodically collects a vector of axis-specific acceleration data. Typically, this is used to re-orient the screen as a user moves the device. But this sensor has also potential to be used for fall detection. Smartphone operating systems such as Android aim to save energy and typically sample with low rates (between 20 and 100 Hz). Prior studies have shown that these sampling-rates are suited for fall detection (Mehner et al., 2013; Fudickar et al., 2014).

An optimal fall detection algorithm should combine a high sensitivity with a high specificity. The sensitivity is a measure for the number of correctly detected falls:

$$Sensitivity = \frac{TruePositives}{\text{Number of all falls}}$$

where $TruePositives$ is the number of correctly detected falls.

The specificity measures the rate of $TrueNegatives$, i.e. the percentage of correctly classified non-fall situations:

$$Specificity = \frac{TrueNegatives}{\text{Number of all non} - \text{falls}}$$

A high specificity means a low number of false alarms. This is also a very important metric for the usability of the system.

We analyzed the already published algorithms for fall detection and discovered that there exist different variations of the algorithms which differ for example in the calculation of the orientation change and in their threshold settings. Since these algorithms were not compared before, we implemented and tested these different algorithms. As a result from these tests, we delivered new threshold values which perform better than the ones published before.

In September 2014, there are already about 13 apps available in the Google play store which claim to perform fall detection. The second part of this work is a survey about the quality of those applications.

The remainder of the article is structured as follows: In Section 2 related work is discussed. The basic algorithm of threshold-based fall detection is introduced in Section 3. Section 4 presents the evaluated algorithms and parameter settings. The results of our evaluation are shown in Section 5. Further, we tested current available fall detection applications from the Google App Store. The results are presented in Section 6. The article ends with a conclusion.

Gimpel T., Kiertscher S., Lindemann A., Schnor B. and Vogel P..
Evaluation of Threshold-based Fall Detection on Android Smartphones.
DOI: 10.5220/0005280805980604
In *Proceedings of the International Conference on Health Informatics* (HEALTHINF-2015), pages 598-604
ISBN: 978-989-758-068-0
Copyright © 2015 SCITEPRESS (Science and Technology Publications, Lda.)

2 RELATED WORK

Several groups have already investigated fall detection on smartphones.

In 2010, Dai et al. proposed the PerFallD prototype for the first available Android smartphone, the G1 phone (Dai et al., 2010). The detection algorithm uses four thresholds to detect an impact and a stable phase and the change of orientation. The chosen thresholds are not applicable for current Android versions anymore, but the study already shows the feasibility of smartphone based fall detection. For the evaluation, data from 450 falls were collected with different directions (forward, lateral and backward), different speeds (fast and slow) and in different environment (living room, bedroom, kitchen and outdoor garden). Further, data of activities of daily living (ADL) including walking, jogging, standing and sitting were gathered. Fall and ADL data were provided by graduate students from 20 to 30 years old. The authors present evaluations for wearing the smartphone in a shirt pocket, on the belt, or in the pant pocket. The presented evaluation shows that PerFallD achieves the best performance when the device is attached with the belt with an average False Negative value being 2.67 % and the False Positive value being 8.7 %. While this is a promising result, no evaluation with ADLs of older people is given. Further, the authors report that the system keeps running about 33.5 h until the battery is exhausted which also confirms the usability of these devices for fall detection.

Dai et al. compare their system with a commercial product provided by Brickhouse (Brickhouse,) . This fall detector consists of a teleassist base and a portable sensor. Since the base device needs to be installed indoors and the signal transmission distance between the sensor and the base is limited, the system is only useful in this environment. The experiments reported in (Dai et al., 2010) show that this system has a high false negative rate in backward falls (29.9 %) and a high false positive rate (21.9 %).

In (Hwang et al., 2012), a smartphone running Android 2.3.3 was used to record 100 Hz bandwidth signals from the three-axis acceleration sensor. The saved data were processed to detect falls using Matlab 7.0 not on the device itself, but on a PC. Based on data from 200 experimental falls, obtained by fastening a smartphone to a belt worn around the waist, an overall detection rate of 95% was achieved, corresponding to direction-specific rates of 94% for forward falls, 100% for backward falls, 94% for leftward falls and 92% for rightward falls. Based on the experimental results of 6 ADL scenarios, the threshold for acceleration was established at 2.2 g and the threshold for angular displacement was set at 50°.

Since Android based smartphones use sampling-rates of maximum 100 Hz, different sampling rates were investigated in a trace-driven simulation ((Fudickar et al., 2014)). The results confirm that low sampling rates of at least 50 Hz can be used and have a sensitivity of 99% for the collected fall records. Further, the specificity of the threshold-based fall-detection algorithm was tested with ADLs of **elderly people**. In the simulation, no false positive occurred. The threshold settings were taken from (Karth, 2012).

Mehner et al. (Mehner et al., 2013) also published results from experiments with threshold based fall-detection on smart phones which confirm that the lower sampling rates (such as 50 Hz) that are supported by Android are uncritical. Their results indicate that the exclusion of the free-fall detection phase may increase the detection accuracy by 27 % from 56% with free-fall detection to 83% without free-fall detection. Overall the proposed algorithm achieved a maximal sensitivity of 83% and a specificity of 100% in the presented experiments. Results from real falls and ADLs from elderly people are not presented.

3 THRESHOLD BASED FALL DETECTION

For a tri-axial accelerometer, fall situations are characterized by multiple sequential phases. The accelerometer collects a vector (x,y,z) of the axis-specific acceleration forming together the acceleration. The length of the acceleration vector corresponds to the power of the acceleration. Since we use *gravity* as our metric, we have to divide the length of the acceleration vector by the power of the gravity on earth which is $1g = 9,81\frac{m}{s^2}$ [1]. This leads to equation (1):

$$|\overrightarrow{v}| = \frac{\sqrt{x^2+y^2+z^2}}{9,81} \tag{1}$$

This value is the so-called **G-Value** used in the threshold based fall detection algorithm.

The different phases of the fall detection algorithm are shown in Figure 1. The fall detection algorithm starts with the *FreeFall Test Phase*. Within a fall situation, the falling body experiences zero-gravity.

[1]The exact gravity appeals on a body (or mass in general) depends on where you are on earth. This is due to gravity anomalies and the fact that the earth is not a perfect sphere. One can say in general, that gravity is stronger near the poles. The differences in gravity is still small enough to set a general value for gravity in a fall detection application

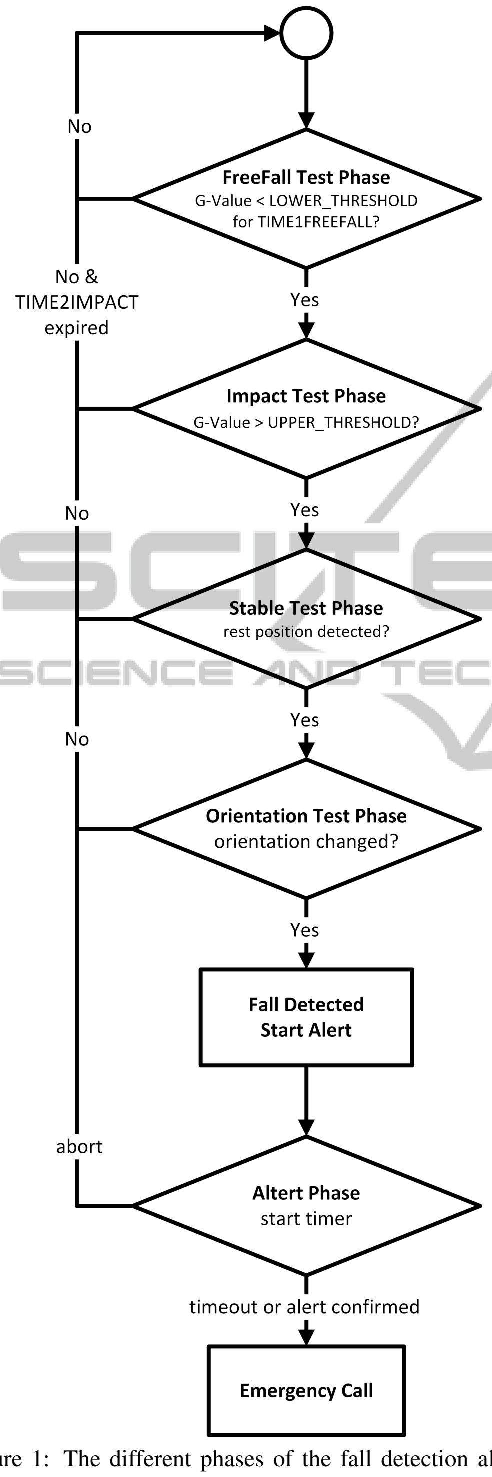

Figure 1: The different phases of the fall detection algorithm.

The free fall is the initial event of each fall situation and therefore has to be recognized first. To detect a free fall, the gravity value must drop below the free fall threshold LOWER_THRESHOLD for a minimal duration of TIMER-FREEFALL. If this is not the case and the threshold is exceeded again before TIMER-FREEFALL has expired, the algorithm stays in the *FreeFall Test Phase*. Other algorithms (such as (Bourke et al., 2007; Mehner et al., 2013; Fudickar et al., 2014)) exclude a minimal duration for free fall detection since without the free fall detection phase they observed a higher fall detection rate in their experiments.

Once a free fall was detected, the *Impact Test Phase* starts in which the algorithm tries to detect an impact, which is given if the G-Value exceeds the so-called UPPER_THRESHOLD. The maximal duration between free fall and impact recognition may not exceed the duration defined in TIMER-IMPACT. If this period expires and no impact was detected, the algorithm resets and starts again with the *FreeFall Test Phase*.

In a fall situation the impact is followed by a stable phase, thus the algorithm switches into the *Stable Test Phase*. During this Phase, the G-Values has to drop below the VARIATION threshold (1st value of the *Stable A* row of Table 2), for a minimal duration of TIMER-NOMOVEMENT (2nd value of the *Stable A* row of Table 2). All this must happen within another time window called TIMER-STABLE (3rd value of the *Stable A* row of Table 2) to pass this phase. Mehner et al. (Mehner et al., 2013) divide this phase into two sub-phases (depicted by the *Stable B* row of Table 1 and 2). The first sub-phase is only for omitting fluctuations in the G-Value which may occur right after an impact and its duration is equal to TIMER-NOMOVEMENT. The second sub-phase tests, if the VARIATION threshold is exceeded by the G-Value within the time of that sub-phase which is equal to the TIMER-STABLE parameter. If it is exceeded the *Stable Test Phase* has failed and the algorithm resets.

The correct distinction between a fall situation and an *Activity of Daily Life (ADL)* is one of the most important features of a fall detection algorithm, since even a small false positive rate will make the algorithm useless for real world scenarios. To distinguish a fall situation from an ADL, some researcher propose an *Orientation Test Phase* (Karth, 2012; Mehner et al., 2013). During this phase it is tested, if the device, and thus its carrier has changed its orientation significantly. This can be done in different ways as depicted in Table 1 and 2 by the *A*, *B* and *C* in the *Orientation* row.

- Version *A*, from (Karth, 2012), takes the last acceleration vector prior to a possible fall situation and compares it with the acceleration vector given after the *Stable Test Phase*. If the difference is 45° or more, the fall is verified.

- Version *B*, from (Mehner et al., 2013), compares the mean of the last 100 values before the fall situation with the mean of the first 100 values after the detected fall. This is done separately for every axis. If one of the axis has a deviation of 0,4g or greater, the fall is verified.
- Version *C* which is proposed in this paper combines Version *A* and *B* and compares the mean acceleration vector $\overrightarrow{v}$ of the last 100 acceleration vectors given before the fall situation occurred with the mean acceleration vector $\overrightarrow{w}$ of the last 100 acceleration vectors given after *Stable Test Phase* is passed. The angle is calculated as:

$$\alpha = \arccos \frac{\overrightarrow{v} * \overrightarrow{w}}{|\overrightarrow{v}| * |\overrightarrow{w}|} \tag{2}$$

If the angle is greater than 66°, the fall is verified.

After a fall was detected by passing all previous phases, the *Alert Phase* begins. In this phase, a timer is started and the device informs the user that a fall was detected. Typically the user can confirming or aborting the alert, so an emergency call can be either initiated or prevented. In case the timer expires and no action was chosen, an emergency call is initiated. It often happens to elder people that they will not lose its consciousness after a fall, but will still be unable to interact with the device. If this is the case, the timer approach will work fine due to an activation of the hands-free phone system. Alternatively or in addition to an emergency call, a SMS message, TCP message or other message systems can sent alerts to a configured phone number, IP address etc.

4 EVALUATION ENVIRONMENT

While fall detection with smartphones seems to be a promising approach and different groups have published work in this direction, an analysis with real falls of elderly people is unrealistic and still missing.

In (Fudickar et al., 2014), results from a simulation with ADLs of elderly people was presented. It was shown that fall detection is also possible with the low sampling rates used on smartphones. But the simulated fall detection algorithm was adapted for a special sensor, the ADXL345, which is not used in current smartphone technologies.

Therefore, we decided to repeat the experiment for Android smartphones. The following steps were executed:

1. implementation of the fall detection algorithm (as described in Section 3) for Android smartphones,
2. integration of the new implemented fall detection algorithm into the simulation testbed,
3. collecting falls with an Android smartphone
4. collecting ADLs from elderly people with an Android smartphone.

During the implementation phase, we considered the already published algorithms and discovered the different variations presented in Section 3. Since it was still unclear which algorithm and parameter setting will give the best results, we implemented and tested the following configurations for Android smartphones:

1. **Karth FF:** The parameter values for thresholds and timer settings were taken from (Karth, 2012).
2. **Karth:** The same as **Karth FF**, but the Free Fall Phase is omitted.
3. **Mehner FF:** The parameter values for thresholds and timer settings were taken from (Mehner et al., 2013).
4. **Mehner:** The same as **Mehner FF**, but the Free Fall Phase is omitted.
5. **Gimpel:** The fall detection algorithms from (Karth, 2012) and (Mehner et al., 2013) were implemented by Tobias Gimpel in (Gimpel, 2014). He also developed a combination of the former two settings.

An overview of the different variants and the relating parameter settings are given in the Tables 1 and 2.

5 RESULTS

The sensitivity and specificity of the threshold based fall detection algorithms is evaluated with the fall sets and ADLs that are described in the following subsections.

5.1 Fall Set and Sensitivity

Since, the risk of injuries in case of older test persons is obviously much too high, the *fall set* was recorded by three test persons between 23 and 55 years old.

The fall set covered frontal falls, falls to the left and to the right. The test persons used two types of smartphones: HTC Desire 816 and Sony Xperia V. The device was put into a funny bag at the hip in front.

The first test person (29 years old) simulated 30 falls: 10 frontal, 10 to the left side, and 10 to the right side (prob29). The second test person (23 years old) simulated 12 falls: 4 frontal, 5 to the left side, and 3 to the right side (prob23). The third test person (55

Table 1: Different versions of the fall detection algorithm.

	Karth FF	Karth	Mehner FF	Mehner	Gimpel
Free Fall	X		X		X
Impact	X	X	X	X	X
Stable A	X	X			X
Stable B			X	X	
Orientation A	X	X			
Orientation B			X	X	
Orientation C					X

Table 2: Parameter settings of the different algorithms.

	Karth	Mehner	Gimpel
Free Fall	0,75g/30ms	0,5625g/30ms	0,75g/30ms
Impact	2g/500ms	2,3g/300ms	2g/500ms
Stable A	0,4375g/1sec/3,5sec	-	0,2g/1sec/3,5sec
Stable B	-	0,4g/1sec/1sec	-
Orientation A	45°	-	-
Orientation B	-	0,4g	-
Orientation C	-	-	66°

years old) simulated 10 falls: 4 frontal, 3 frontal left, and 3 frontal right (prob55). The different fall characteristics and number of falls are due to the different *fall capabilities* of the test persons. All falls recorded were critical falls, with loss of consciousness (where the test person did not move after the falls).

We used the data from our simulated falls as input to test the algorithms and parameter settings described in Section 4. The results are shown in Table 3. In the column NOF the number of falls is given. The detection rate using the `Karth FF, Karth, and Gimpel` settings are very good. The `Mehner` settings show an inconsistent behavior. For all three test persons, the algorithm without Free Fall Phase has a higher sensitivity, which correlates with prior published results in (Mehner et al., 2013). But for the falls from prob29, the sensitivity for both variants is only about 50 %. While these variants have a good detection rate for falls ahead, they have difficulties to detect falls to the left or right side.

5.2 ADLs of Elderly People

To evaluate the specificity of the algorithms, we recorded the acceleration characteristics of ADLs of two test persons, 55 resp. 72 years old. As before, the test persons were equipped with a fanny pack, worn at the test persons' hip in which the device was carried.

The considered recording duration per test person ranged from 11 (prob72) to 286 (prob55) hours. The older test person carried the device during one day, while the younger test person carried it several days during her daily activities covering typical daily activities including eating lunch, walks, activities at work and also homework.

The maximal recorded acceleration in the cropped recordings was at 3.17 g.

Table 4 shows the results for the different versions and parameter settings. The most robust versions are `Mehner with the Free Fall Detection Phase` (no False Positives) and `Gimpel` (only 2 False Positives). Further, the `Karth FF` version including the Free Fall Phase also performs better than the simple `Karth` version. But both `Karth` version show much higher False Positives than the other settings. Obviously, the Fall detection Phase is important for a low False Positive Rate.

5.3 Interpretation of the Results

While (Mehner et al., 2013) proposed the exclusion of the Free-Fall Detection step to enhance the sensitivity, our experiments with ADLs show that the free-fall detection phase is important for a low False Positive Rate. Therefore, we consider in the following only those algorithms which use this phase: `Karth FF, Mehner FF`, and `Gimpel`. From these, `Mehner FF` has a significantly worse sensitivity (see Table 3). The sensitivity of `Kart FF` and `Gimpel` is comparable. Since regarding the specificity, `Gimpel` is much better than `Kart FF` in our experiments, we conclude that the settings from `Gimpel` are a good compromise.

Table 3: Sensitivity: Detected Falls.

	fall direction	NOF	Karth FF	Karth	Mehner FF	Mehner	Gimpel
prob29	overall	30	30	28	13	15	26
	ahead right	5	5	4	5	5	4
	ahead left	5	5	5	5	5	4
	left side	10	10	10	3	4	10
	right side	10	10	9	0	1	8
prob23	overall	12	10	9	3	10	9
	left side	5	4	4	1	5	3
	right side	3	2	2	2	2	3
	ahead	4	4	3	0	3	3
prob55	overall	10	9	10	2	8	9
	ahead right	3	3	3	1	2	3
	ahead left	3	3	3	0	3	3
	ahead	4	3	4	1	3	3

Table 4: Specificity: Results from ADLs.

	duration	Karth FF	Karth	Mehner FF	Mehner	Gimpel	max g	min g
prob55	286 h	24	57	0	5	2	3.179	0.061
prob72	11 h	0	3	0	1	0	2.555	0.041

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

6 EVALUATION OF APPs FROM THE GOOGLE APP STORE

The second part of this work is a survey about useful fall detection applications in the Google play store. In September 2014, the search for *fall detection*, within the Google play store, results in 22 hits for applications in different categories (e.g. medicine, health & fitness, lifestyle and social). 13 out of these 22 applications are related to real fall detection and thus were considered in the following. 2 out of the 13 remaining applications were commercial applications and available for 4 € and 120 € respectively. Due to the high price, the 120 € application was not tested any further.

First, we tested their user interface and the usability. The following characteristics of an application resulted in an exclusion for further tests:

- a failed or impossible installation
- no reaction of the application after installation
- the need to register for a phone call in a foreign country
- the phone call destination is not obvious

Only 8 applications passed these initial tests and were further tested regarding their sensitivity and specificity.

The specificity was tested with a fixed set of activities of daily living (ADL) like walking around, climbing stairs or sitting down on a chair. All this was done at varying speeds in a time window of about 10 minutes per application. During these tests, the smartphone was in a trousers' pocket.

The sensitivity of the applications was tested with 10 falls in forward direction performed by two test persons (23 resp. 55 years old). During the fall tests, the smartphone was worn in a belt bag in front.

The results of the tests are shown in Table 5. The *FP* column indicates observed false positives within the ADL tests. Columns *prob23* and *prob55* refer to the 23 and 55 years old test persons. 50 % of the applications did not pass the ADL test and indicated falls while executing. Regarding the sensitivity, the best application was *iCare Personal Emergency Alert* with a detection rate of 70 %, followed by *Fade: fall detector* and *SecureMe Active* with 60% detection rate each. All other applications had a very low sensitivity even for the easiest category of falls in forward direction, or low specificity.

7 CONCLUSIONS

Automatic fall detection is one of the most important assistive technologies for elderly to lead a self determined life. An old person who lives alone is supported by the automatic fall detection system and feels safer.

Table 5: Test results for applications from the Google play store.

name	published in app store by	FP	prob23	prob55
T3LAB Fall Detector	T3LAB-Technology Transfer Team	no	1/5	3/5
iCare Personal Emergency Alert	Fansoft Labs	no	5/5	2/5
Smart Fall Detection	Hamideh Kerdegari	no	0/5	0/5
Emergency Fall Detector	Socaplaya21	no	0/5	0/5
Fall Detector	Spantec GmbH	yes	0/5	0/5
Fade: fall detector	ITER S.A.	yes	3/5	3/5
iFall: Fall Monitoring System	FSU Mobile Solution	yes	0/5	2/5
SecureMe Active	Orion Systems	yes	2/5	4/5

Threshold-based fall detection on smartphones has shown to be a promising approach. In this study, we evaluated different parameter settings and also tested fall detection applications from the Google play store.

The results of our experiments are:

1. We propose a variant of the known threshold-based fall detection algorithms which performs better than the algorithms published before in the literature. The so-called `Gimpel` variant has a high sensitivity (44 of 52 falls detected) and a high specificity (only 2 false positives in 297 h ADLs). Regarding sensitivity, the `Karth FF` algorithm performed better (49 of 52 falls detected), but suffers from a lower specificity (24 false positives).

2. While (Mehner et al., 2013) proposed the exclusion of the Free-Fall Detection step to enhance the sensitivity, our experiments with ADLs show that the free-fall detection phase is important for a low false positive rate.

3. The tests with the applications from the Google App Store show that the currently available applications have to be considered with care. Only one application performed satisfying in our tests (no false positive and 4 of 10 falls detected).

4. From the experiences with the applications from the Google App Store, we conclude that collecting fall data is a critical factor for the evaluation: different test persons tend to have different simulated fall characteristics.

In the next step, we will investigate the usability of our system in cooperation with a nursing home. While younger people are used to the smartphone technology, this may be a burden for elderly people.

REFERENCES

Bourke, A., O'Brien, J., and Lyons, G. (2007). Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. *Gait & Posture*, 26:194–199.

Brickhouse. Brickhouse alert. http://www.brickhousealert.com/personal-emergency-medical-alarm.html; accessed October, 2014.

Dai, J., Bai, X., Yang, Z., Shen, Z., and Xuan, D. (2010). PerFallD: A pervasive fall detection system using mobile phones. In *PerCom Workshops'10*, pages 292–297.

Fudickar, S., Lindemann, A., and Schnor, B. (2014). Threshold-based fall detection on smart phones. In *HEALTHINF 2014, 7th International Conference on Health Informatics*, Angers, France.

Gimpel, T. (2014). Anpassung eines Algorithmus zur aktiven Sturzerkennung für Android-Smartphones. Bachelor thesis, University of Potsdam.

Hwang, S.-Y., Ryu, M.-H., Yang, Y.-S., and Lee, N.-B. (2012). Fall Detection with Three-Axis Accelerometer and Magnetometer in a Smartphone. In *International Conference on Computer Science and Technology*, pages 65–70.

Karth, C. (2012). Fusion von Sensordaten zur robusten Sturzdetektion im Bereich Assisted Living. Master's thesis, University of Potsdam.

Mehner, S., Klauck, R., and Koenig, H. (2013). Location-independent fall detection with smartphone. In *Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '13, New York, NY, USA. ACM.

Mellone, S., Tacconi, C., Schwickert, L., Klenk, J., Becker, C., and Chiari, L. (2012). Smartphone-based solutions for fall detection and prevention: the FARSEEING approach. *Zeitschrift für Gerontologie und Geriatrie*, 8:722–727.

Sposaro, F. and Tyson, G. (2009). iFall: an Android application for fall monitoring and response. *Conference proceedings: Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2009:6119–6122.