Bogdan DZIADAK[1], Maciej CZUMAK[2]

Institute of Theory of Electrical Engineering, Measurement and Information Systems Warsaw University of Technology (1), Warsaw University of Technology (2)

doi:10.15199/48.2025.03.48

Evaluating the effectiveness of algorithms used for human fall detection

Abstract. *The article presents research on the implementation and testing of three different algorithms used for fall detection. Two of the algorithms use a time-based waveform analysis and monitoring of fixed values of acceleration and angular velocity. The most effective algorithm is also based on the acceleration value but the fall decision is made by a classifier using SVM. This makes it possible to achieve an algorithm efficiency of 97%.*

Streszczenie. *Artykuł przedstawia badania nad implementacją i testowaniem trzech różnych algorytmów wykorzystywanych do wykrywania upadków. Dwa pierwsze algorytmy wykorzystują analizę przebiegów czasowych i monitorowanie stałych wartości przyspieszenia i prędkości kątowej. Najskuteczniejszy algorytm jest również oparty na wartości przyspieszenia, ale decyzja o upadku jest podejmowana przez klasyfikator wykorzystujący SVM. Dzięki temu możliwe jest uzyskanie skuteczności algorytmu na poziomie 97%. (**Ocena skuteczności algorytmów stosowanych do wykrywania upadków ludzi**)*

Keywords: Fall detector, algorithm, SVM.
Słowa kluczowe: Wykrywanie upadków, algorytm, klasyfikator SVM.

Introduction

According to the World Health Organization (WHO), the most common cause of injuries among people over the age of 60 is falls [1]. Falls are defined as unintended events that result in a person coming to rest at ground level due to a loss of balance. Up to 20 percent of falls among the elderly lead to severe brain injuries [2]. When a person over the age of 70 years old suffers a hip fracture, mortality can occur in 30 percent of cases. Only the arrival of assistance within less than 30 minutes provides a chance to save the injured person. However, summoning help is not always possible, as falls often occur in homes where individuals live alone. Modern communication technologies and micro-measurement systems are useful in such situations, allowing help to be called remotely without the participation of the monitored individual. This enables older adults to maintain independence without the need for constant supervision by a caregiver.

In the context of fall detection, methods vary depending on the type of sensor used [3 - 5]. Three main groups can be identified:

Personal Detectors These are integrated monitoring systems based on accelerometers, gyroscopes, magnetometers, and barometers [6 - 9]. They take the form of wristbands, watches, or keychains. Some of these functions can also be incorporated into smartphones. Fall detection algorithms implemented in personal detectors analyze sensor readings over several-second time windows [10].

Ambient Detectors - The difference between environmental detectors and personal detectors lies in the sensor placement, which is located in the user's surroundings. Placed at floor level, accelerometers can detect vibrations [11]. The operating principle of the algorithm is similar to that of personal detectors. However, a significant drawback of this solution is the inability to distinguish between a person falling, loud stomping, or dropping an object. Therefore, environmental detectors are most commonly used in combination with visual detectors.

Visual Detectors – These use camera images to detect falls [12]. During real-time analysis, the algorithm identifies the human silhouette in the room. The detected individual is outlined with a rectangle, and their limbs are also recognized. The algorithm classifies the activity performed by the person into one of several predefined categories. While the individual is standing or sitting, the sides of the rectangle surrounding them are longer than its base. In the event of a sudden reversal of this relationship due to a fall, the algorithm analyzes the positioning of the person's limbs and determines whether a fall has occurred. The biggest drawback of this solution is its cost.

The article will present research on evaluating the effectiveness of algorithms in human fall detection. For this purpose, a fall detector with appropriate communication infrastructure was built. Three algorithms were developed :

- threshold algorithm
- threshold algorithm with additional time window analysis
- an algorithm using artificial intelligence.

Each of the algorithms was implemented on the previously built detector subjected to a series of tests to assess the correctness of operation during falls, but also during typical daily activities.

Implementation

The personal detector was based on the Arduino Nano RP2040 Connect. It is a controller with a built-in 3-axis accelerometer and 3-axis gyroscope, as well as WiFi and Bluetooth communication modules. The detector was in the form of a wristband fastened to the wrist. It is presented in Figure 1.

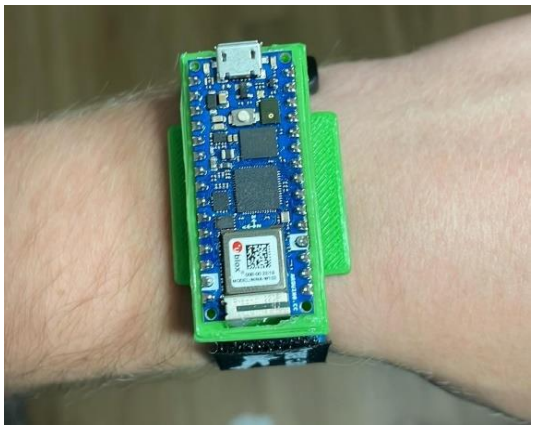

Fig. 1. Personal fall detector on the wrist. Prototype version (photo: M. Czumak).

The data collected by the controller, i.e. measurements of acceleration and values from the gyroscope, were

transmitted via Bluetooth and the MQTT protocol to a server built on a Raspberry Pi 4B. Such a structure allowed archiving the data during the conducted tests. It also provide the possibility to implement advanced calculation procedures, which was the case with the A3 algorithm. Regardless of whether the decision regarding to fall was made directly by the detector or the server, the message about the occurrence of a fall was transmitted to the custodian through the server

Algorithms

A1 – Threshold algorithm

The algorithm [6] continuously calculates the angular velocity and the length of the acceleration vector |a|, in the three axes. The decision to fall is made in two phases. In the first phase, it is checked whether threshold values have been exceeded:

- LFT - (Low Fall Threshold) the threshold of acceleration values, which may indicate the beginning of a fall,
- HFT - (High Fall Threshold) the threshold of the acceleration value, after which there is a high probability that a fall has occurred,
- HG - (High Gyroscope) threshold value from the gyroscope indicating a contraction of the detector's allowable angular velocity.

When the above conditions are met the first flag is set and the second phase of the detection process begins, namely checking whether the acceleration modulus |a| remains below the STH (Standing Threshold) value after a fall. This indicates whether the user is able to move independently after a fall. If there is no movement or it is below the threshold, the second flag is set and information about the fall is sent to the server. The server, in turn, informs the designated caregiver or staff that a fall has occurred.

Changes in acceleration during the fall are shown in Figure 2, where the phases of the algorithm are also marked.

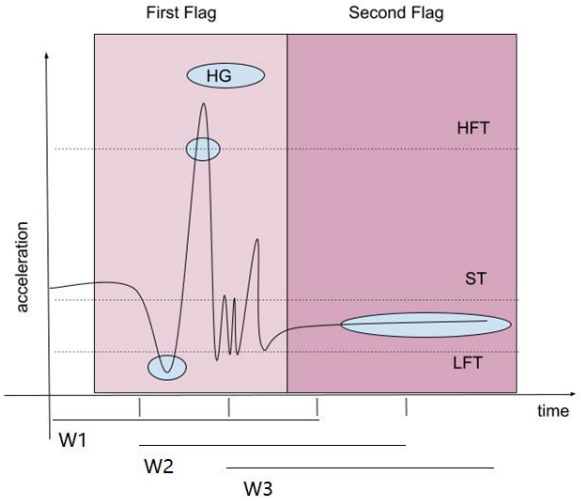

Fig. 2. Acceleration during the fall

A2 – Threshold algorithm with additional time window analysis.

The operation of the second algorithm [13] is based on similar assumptions to the one described earlier. The data are continuously taken from the accelerometer, the modulus of the acceleration vector in 3 axes is calculated, and finally the data are placed in a single row of a matrix with two rows and six hundred and twenty-four columns. This number comes from the fact that the algorithm analyzes the sensor data in three-second windows - this gives the equation:

$$n = 3s * 208Hz = 3s * 208\ /s = 624 \quad (2)$$

The second row of the matrix contains the accelerometer's acceleration values in the x-axis - located along the user's forearm. The course of the fall can be divided into three phases:

- The pre-fall phase,
- The phase of the fall,
- The post-fall phase.

Three three-second time windows shifted by 1 second are analyzed simultaneously (windows W1, W2, W3 are marked on Figure 2). The first two seconds of the time window are analyzed in terms of exceeding the HFT threshold. After that, the sum of the variances of the module acceleration vectors |a| and value of acceleration along the wrist a_x is calculated for the time covering the entire fall. When the value is greater than the threshold, a fall is detected. As before, the decision regarding to fall is sent to the server, which starts the rescue procedure.

A3 – Algorithm using data analysis and machine learning

In this algorithm, we used a classifier based on Support Vector Machines SVM. In order to develop this algorithm, it was necessary to choose a suitable learning dataset. One of the largest available collections of records of human falls and activities is - developed by staff at the University of Rennes – FallAllD [14]. It consists of 26420 records of runs collected using three data loggers worn by fifteen different subjects at the waist, wrist and neck. The waveforms are recorded using an accelerometer, gyroscope, magnetometer and barometer. They can be used for devices involved in detecting falls, or recognizing human activity. Among the recorded activities, there are thirty-four different types of falls, or activities like clapping, moving up stairs, waving, or getting out of bed.

A key step before learning a machine algorithm is to parameterize the signal [15]. This process, called feature extraction, allows the input signal to be mathematically described in an unbiased way. Its result is a set of signal parameters - that is, a numerical expression of features. Parameters, due to the basis of their determination, can be divided into:

- Temporal parameters,
- Spectral parameters,
- Formant parameters,
- Time-frequency parameters.

For signal analysis and feature extraction, the hctsa (highly comparative time-series analysis) framework and the catch22 algorithm were used - fast extraction of the 22 most significant signal time parameters.

From the database, 2279 records were extracted with measured values from an accelerometer worn on the wrist at various life activities including falls. Using a catch22 algorithm, features were extracted from each of the 2279 records. A function in MATLAB -fitcsvm - was used to train the SVM classifier. It is part of the Statistics and Machine Learning toolbox. The best results were achieved for the radial RBF kernel function.

The operation of the algorithm is shown in Figure 3. It can be described as follows Data are continuously taken from the accelerometer, the modulus of the acceleration vector in the 3 axes. After obtaining 4160 samples (equivalent to 20 seconds), feature extraction of the collected time waveform is carried out using the catch22 algorithm. The extracted features are fed to the SVM model running on the server, which decides whether a fall has occurred. If a fall is detected, the caregiver is notified *via* Telegram messenger.

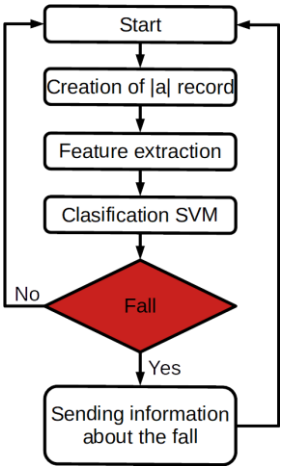

Fig. 3. Flow chart of algorithm using data analysis and machine learning

Tests and results

During the tests, it was checked whether the algorithms would detect a fall in situations where one occurred and whether a false alarm would not be raised during normal activity. The scenarios were repeated 10 times for each algorithm, making it possible to determine, their effectiveness in each situation. This knowledge allows weaknesses in the system's performance to be detected and possibly correct. For example, one of the biggest problems for fall detectors using the threshold algorithm is false detection of a fall during the activity of moving upstairs. Tests of the detector were carried out in an environment replicating real operating conditions. The person testing the detector's performance in practice was a 24 year-old man, with a height of 196 cm and a weight of 102 kg. The detector was placed on his left wrist. The tester's task was to play out the scenarios described in the following paragraphs. Each situation was tested 10 times for each algorithm. The result of each test was a score of 0 (incorrect classification) or 1 (correct classification). The results are summarized in Table 1. To best compare the algorithms with each other, test scenarios were developed. Situations were chosen to reflect real-world conditions of fall detector use. The focus was on circumstances that may occur in daily life. Using this approach, the functionality of the detector can be effectively evaluated, both in terms of sensitivity to falling situations and minimizing false alarms during routine activities of daily life. The following situations were simulated:

1. Fall,
2. Fall and rise,
3. Walking around an apartment,
4. Walking with a stumble,
5. Sitting down,
6. Up and down stairs,
7. Clapping hands.

In order to determine the effectiveness of the three algorithms tested, a study was conducted that covered 7 different scenarios that would occur in the daily life of a person using such a device. This was translated into 70 test trials for each algorithm, for a total of 210 trials

The Threshold algorithm was wrong in 9 of the 70 trials. This gave the correctness of its decision at 87%. The Threshold algorithm with additional time window analysis made incorrect classification in 4 trials, which translates into a 94% success rate. In contrast, the algorithm using data analysis and machine learning indicated an incorrect verdict in only 2 cases. This resulted in high efficiency of 97%.

Table 1. Result of performer tests

Tested activities		Series									
		1	2	3	4	5	6	7	8	9	10
A1	Fall										
	Fall and rise										
	Walking around an apartment										
	Walking with a stumble										
	Sitting down										
	Up and downstairs										
	Clapping hands										
A2	Fall										
	Fall and rise										
	Walking around an apartment										
	Walking with a stumble										
	Sitting down										
	Up and downstairs										
	Clapping hands										
A3	Fall										
	Fall and rise										
	Walking around an apartment										
	Walking with a stumble										
	Sitting down										
	Up and downstairs										
	Clapping hands										

Correct clasification — Incorrect classification

In 3 of the 7 scenarios tested, all algorithms showed 100% correct decisions. These were scenarios numbered 3, 4 and 6, i.e. moving around the apartment, moving upstairs and marching with momentary loss of balance. These fall into the category of scenarios during which the test person is in constant motion. This explains the lack of false alarms in the 1st and 2nd algorithms, i.e. those that base their operation on the thresholds-crossing readings from the accelerometer and gyroscope. The reliance on thresholds explains why algorithms 1 and 2 were wrong in scenarios 1, 5, and 7, i.e., falling and lying motionless, sitting on a chair, and clapping hands. In the case of falling and lying motionless, false alarms occurred when the person was lying on the ground after falling - moving. This resulted in one of the conditions not being met, thus causing the fall to go undetected. On the other hand, fall detection in situations of sitting on a chair or clapping was due to poor scenario planning.

After performing the actions of sitting down or clapping, the test subject remained in one position and did not move for about 10 seconds. This could have triggered the inactivity threshold to activate and detect a fall. Still, the A2 algorithm performed better than A1, most likely because it did not use gyroscope data. In contrast to its reliability in other scenarios, the A3 algorithm did not perform well in the fall and immediate rise scenario. Twice a fall was detected, even though the test subject got up. It is possible that this was due to by the user's rate of lifting off the ground.

It is also necessary to analyze the space complexity of the algorithms, that is, the amount of memory needed to perform the task, expressed as a function of the amount of data [3]. This parameter is very important in the implementation of algorithms on microcontrollers, which operate on much smaller memory resources than computers. Greater use of RAM will translate into higher current consumption and lower battery life of the device. RAM occupancy was estimated from a sub-function using the start and end pointers of the allocation memory stack. The ranking is as follows:

1. Threshold algorithm - 16% of the microcontroller's RAM used,
2. Algorithm using data analysis and machine learning - 18% of the microcontroller's RAM consumed, for data acquisition only.
3. Threshold algorithm with additional time window analysis - 21% of microcontroller RAM consumed.

Conclusion

The article presents research verifying the effectiveness of algorithms for human fall detection. For this purpose, a detector was built using a 3-axis accelerometer and gyroscope, on which 3 algorithms were implemented and tested. Seven typical activities of daily life were tested. The most effective turned out to be an algorithm using machine learning and a classifier based on Support Vector Machines with efficiency 97%. However, it should be noted that this method consumes the most computing resources, but part of the calculations can be performed on the server.

***Authors**: dr inż. Bogdan Dziadak,, Institute of Theory of Electrical Engineering, Measurement and Information Systems, Warsaw University of Technology ul. Koszykowa 75 00-662 Warszawa, E-mail bogdan.dziadak@pw.edu.pl; mgr inż. Maciej Czumak Warsaw University of Technology, ul. Koszykowa 75 00-662 Warszawa,*

REFERENCES

[1] World Health Organization - Falls. : https://www.who.int/news-room/fact-sheets/detail/falls. (01.08.2024)
[2] PZH - Narodowy Instytut Zdrowia Publicznego, Urazy i upadki osób starszych. .: https://pacjent.gov.pl/zapobiegaj/urazy-i-upadki-osob-starszych. (01.08.2024)
[3] Wang X., Ellul J., Azzopardi, G., „Elderly Fall Detection Systems: A Literature Survey", *Frontiers in Robotics and AI*, t. 7, 2020), ISSN: 2296-9144. DOI: 10.3389/frobt.2020.00071
[4] Mubashir M., Shao L., Seed, L.. A survey on fall detection: Principles and approaches, *Neurocomputing*, *100 (2013)*, 144-152.
[5] Noury, N., Fleury, A., Rumeau, P., Bourke, A. K., Laighin, G. O., Rialle, V., & Lundy, J. E. Fall detection-principles and methods. (2007) *29th annual international conference of the IEEE engineering in medicine and biology society* (pp. 1663-1666). IEEE.
[6] Dziadak B., Czumak M., „Personal fall detector using MEMS sensors", *23rdInternational Conference on Computational Problems of Electrical Engineering*, (2022), s. 1–4.
[7] Kau, L., Chen, C., A Smart Phone-Based Pocket Fall Accident Detection, Positioning, and Rescue System, *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 44-56, (2015) doi: 10.1109/JBHI.2014.2328593.
[8] Kołodziej M., Majkowski A., Czop W., Tarnowski P., Rak R. J. Sawicki D., „Fall Detection Using a Smartphone", *IEEE 21st International Conference on Computational Problems of Electrical Engineering*, (2020), s. 1–4. DOI: 10.1109/CPEE50798.2020.9238691.
[9] T. Kose Y. Terzioglu, K. A. i Akin, T., A single-mass self-resonating closed-loop capacitive MEMS accelerometer, *2016 IEEE SENSORS*, (2016), pp. 1-3, doi: 10.1109/ICSENS. 2016.7808711.
[10] Hsieh S. T., & Lin C. L.. Fall detection algorithm based on MPU6050 and long-term short-term memory network. *2020 International Automatic Control Conference (CACS)* (2020) pp. 1-5. IEEE.
[11] Werner F., Diermaier J., Panek P., Fall detection with distributed floor-mounted accelerometers: An overview of the development and evaluation of a fall detection system within the project eHome, *2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, (2011), pp. 354-361,
[12] Lin, B.-S. i in., Fall Detection System With Artificial Intelligence-Based Edge Computing, *IEEE Access*, t. 10, s. 4328–4339, (2022). DOI: 10.1109/ACCESS.2021.3140164.
[13] Saleh, M., Georgi, N., Abbas, M. I Jeannès, R. L. B., „A Highly Reliable Wrist-Worn Acceleration-Based Fall Detector", *2019 27th European Signal Processing Conference (EUSIPCO),* (2019), s. 1–5. DOI: 10.23919/EUSIPCO.2019.8902563
[14] Saleh, M., Abbas, M., & Le Jeannes, R. B. FallAllD: An open dataset of human falls and activities of daily living for classical and deep learning applications. *IEEE Sensors Journal*, (2020) 21(2), 1849-1858.
[15] Free-lectures Parametryzacja sygnału i ekstrakcja cech https://pre-epodreczniki.open.agh.edu.pl/openagh-permalink_view.php?moduleId=2074&link=facccf589548c1e880246556fba8eab3 (01.08.2024)