Real-time Alarm Model Adaptation Based on User Feedback

Violeta Mirchevska[1], Boštjan Kaluža[2], Mitja Luštrek[2], and Matjaž Gams[2]

Abstract. This paper presents real-time adaptation of alarm detection models in a remote health monitoring system based on user feedback. Real-time adaptation enables systems to fine-tune to the needs and preferences of the user and changes in the environment. This way, the system performance is improved in terms of technical accuracy and subjective user wishes. Two types of alarm detection models were used: (1) models in the form of rules created by domain expert and (2) models induced by machine learning. The problem of adaptation for the rule-based models is defined as Markov decision process. Machine-learning models are adapted by rebuilding the model every time new data is obtained. We tested the adaptation capabilities of the two types of alarm detection models based on their accuracy and time-to-alarm (needed length of possibly critical activity, such as lying on the ground, which causes the models to raise an alarm). Both types of models achieved 90% alarm detection accuracy. The rule-based models decreased time-to-alarm when user-triggered alarms were raised and increased it when the user indicated false alarm. We did not observe this process for the machine-learning models.

1 INTRODUCTION

Remote health monitoring is gaining attention in developed countries. One of its main goals is to allow users, particularly the elderly, to continue living at home instead of being admitted to a nursing home or hospital. On the one hand, this increases independence and quality of life, and on the other hand, it significantly reduces elderly-support costs. Many research projects address the problem of design and development of systems that can intelligently, continuously and pervasively monitor the health conditions of its user.

In order for remote health monitoring to achieve its goals, it must be highly reliable, which means that it must detect all major health problems without raising too many false alarms. Given the diversity of users, it is difficult to develop a system that will suit each user in each possible circumstance from the start. Therefore, real-time system adaptation based on user feedback is an important topic of research.

We have studied the problem of real-time system adaptation based on user feedback in Confidence [1], ubiquitous care system to support independent living of the elderly. This system is able to detect emergency situations such as falls, which result in the user lying/sitting at an inappropriate place (e.g. on the ground). The user of the system may provide feedback concerning the detection of such situations through portable device. Emergency-situation detection is performed by rule-based alarm detection agents incorporating domain knowledge and machine-learning alarm detection agents. In this paper we present and compare the adaptation capabilities of the both agent types to user feedback.

[1] Result d.o.o., Bravničarjeva 11, 1000 Ljubljana, Slovenia, e-mail: violeta.mircevska@result.si

[2] Jožef Stefan Institute, Department of Intelligent Systems, Jamova 39, 1000 Ljubljana, Slovenia

The paper is organized as follows: Section 2 presents related health monitoring systems and adaptation capabilities in the intrusion detection domain. In Section 3 we present Confidence. The alarm detection system, which constitutes a module in the Confidence system, is presented in detail. Section 4 describes the procedure for adapting the rule-based and machine-learning alarm detection agents. Section 5 presents the experiments for evaluating the adaptation of these agents as well as achieved results. Section 6 concludes the paper and states open problems for future work.

2 RELATED WORK

Remote health monitoring systems mainly focus on fall detection in domestic environment. Several systems were introduced able to detect falls and trigger an alarm. Zang et al. [2], for example, designed a SVM-based fall detector using a waist-worn accelerometer. Similar functionalities were presented by Kangas et al. [3] and also by Bourke et al. [4] using gyroscope, Fu et al. [5] using video, and Lustrek and Kaluza [6] using a localization system. There are also commercially available products, for example, by AlertOne [7] and Zenio [8]. Cesta et al. [9] proposed an agent-based approach for detection of inconsistencies in the execution of daily activities and reacting to exogenous events. All these systems constructed a model from learning data that remained unchanged after deployment at a user, meaning that the model was not able to adapt to the user and the environment.

On the other hand, there were several successful applications of adaptive incremental learning in intrusion detection. Pietraszek [10] introduced an approach for reducing false alarms by having a human analyst review the alarms and mark them as true or false, after which a new model was built by machine learning. NIDES system [11] generates user profiles that are constantly aged, meaning that new behavior is compared to the most recently observed one. Cannady [12] demonstrated the use of reinforcement learning method for adaptation according to user feedback, while Hossain and Bridges [13] proposed a fuzzy association rule system for adaptive anomaly detection.

We were unable to find any paper related to adaptive learning in remote care which was partly motivation for our work. Adaptation seems a necessity if one wants to build a truly user-friendly system. We propose methodology for adaptation of both machine-learning-based and rule-based systems. The latter is particularly relevant in the field of remote health monitoring, since a large number of remote health monitoring systems found in literature are based on expert knowledge.

3 ALARM DETECTION IN CONFIDENCE

We have designed the Confidence system as a multi-agent system since agents are flexible, adaptive and robust. Figure 1 presents the architecture of Confidence. The user is tracked using *sensor agents* part of a localization system. The raw data is filtered by *filtering agents* and the posture of the user is recognized using *posture*

recognition agents. History of user posture and movement is used for alarm detection by *alarm detection agents*. When alarming situation is detected, it is reported to the user by *communication agents* either over a portable device that is part of the system or by a computer screen. In this paper we study real-time adaptation of alarm detection agents based on user feedback. Unusual movement such as limping is also detected in Confidence by tracking a variety of statistics (e.g. walking signature) by *statistics agents*. In this paper we will not present the statistics agents since they are independent of the alarm detection agents.

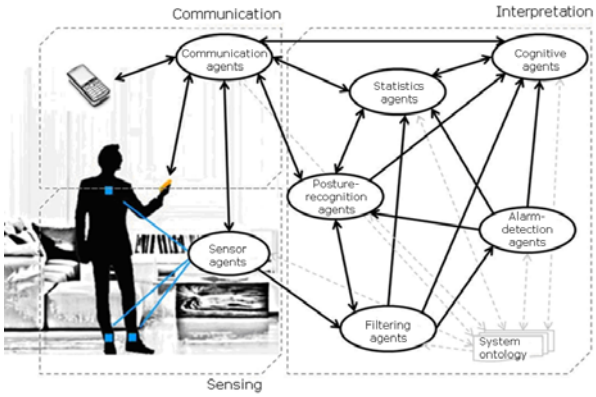

Figure 1. Architecture of Confidence.

For localization, we selected the commercially available Ubisense system [14], which can determine positions of a set of wearable tags using radio technology. Based on studies of posture recognition accuracy that can be achieved with different tag number and placement configurations [15], we placed wearable tags on the following parts of user's body: chest, belt (optional), left and right ankle. In a typical open-environment, the localization accuracy of Ubisense is about 15 cm, but in practice it may drop to 200 cm or more. What is more, sensor data is not necessarily available at each moment in time. Therefore, filtering agents were developed [16] in order to tackle the problems with the Ubisense system.

User's posture is classified as one of the following 5 postures: standing, sitting, lying, on all fours and intermediate states (falling, going down and standing up). Posture recognition was performed using rule-based and machine-learning agent, whose predicted posture is combined using merge agent in order to obtain final posture classification.

The rule-based posture recognition agent is used to reduce the gap between real-world and data captured in training phase. It consists of a set of rules for posture classification. Attributes that need to be included in a rule condition were determined by using knowledge extracted from decision trees and human know-how [17]. Two approaches for determination of limits for each condition in the rules were considered: (1) genetic algorithms and (2) approach which computes information gain for each attribute included in a particular rule in order to determine the most suitable rule condition limit. In this paper we use the information gain approach in order to determine the condition limits of the posture recognition expert knowledge agent.

The machine-learning posture recognition agent contains posture recognition knowledge captured in training data. It takes 10 successive sets of attributes computed for successive time frames as an attribute vector with the prevailing posture in those frames as the class value. We have tested a variety of classifiers, including C4.5, Naïve Bayes, SVM, kNN, Bagging and AdaBoost [6]. Based on these studies, we selected machine-learning posture recognition agent to use the random forest ensemble classifier since it offers the highest classification accuracy.

The posture-recognition-merge agent takes into account the ability of the rule-based and machine-learning posture recognition agent to correctly classify each posture of interest. This ability can be easily determined by exploiting classification's confusion matrix from each of the two agents. In particular, for each posture of interest we compute its probability to be the true posture using the predictions by both the rule-based and machine-learning agent taking into account their ability to correctly classify each posture of interest. The posture with the highest probability is considered as true. This way the strengths of each agent are exploited when determining true user posture.

Table 1. Confusion matrix of joint classification using random forest and expert rules.

	Standing	sitting	lying	intermediate state	on all fours
Standing	72	9	0	19	0
sitting	7	83	0	3	7
lying	2	21	63	8	6
intermediate state	36	19	6	37	2
on all fours	4	11	9	21	55

The confusion matrix of the posture-recognition-merge agent is shown in Table 1. The left column shows the label of the correct posture, and the top row shows the assigned label. Except intermediate state and on all fours, all basic postures are detected reliably. Most errors appear on transitions from one state to another. These errors do not influence system performance since they introduce just time shift.

Alarm detection agents raise alarms when falls and sudden health problems of the user are detected. These are situations which are reflected by person lying/sitting at an inappropriate place (e.g. on the ground) for a prolonged period of time. Similarly to activity recognition, we considered a rule based and machine-learning based approach for alarm detection.

Rule-based alarm detection agents contain rules created by domain experts concerning situations in which an alarm should be raised. We considered four types of alarm situation (1) falling detected and person lying/sitting immovable at inappropriate place, (2) falling detected and person lying/sitting movable at inappropriate place, (3) person lying/sitting immovable at inappropriate place and (4) person lying/sitting movable at inappropriate place. Each rule-based alarm detection agent captures one type of alarm situation which it gives as output.

Machine-learning alarm detection agents contain knowledge about alarm situations that are induced from available data concerning such situations. The alarm detection module in Confidence contains two such agents, one containing SVM classifier, the other C4.5 classifier. They obtain percentage of all observed user's postures and movement in periods of 5 s, 10 s and 15 s as input. These intervals are suitable for our experiments because we considered a reasonable period of lying/sitting at inappropriate place after which an alarm should be raised somewhere between 5 and 15 s. In real life a longer period might

make more sense, in which case the intervals used in attributes should be lengthened. The output of the machine-learning alarm detection agents, unlike the rule-based agents, is only the presence or absence of emergency situation, without any information about its type.

When an alarm is raised, the communication agents notify the user either over a portable device that is a part of the system or by a computer screen. The user may (1) confirm the alarm (explicitly by pressing a button or implicitly by doing nothing), meaning it is true alarm and the user needs help, (2) inhibit the alarm, which means that this is a false alarm and (3) suspend alarm, meaning that the situation was actually a true alarm, but the user can help himself/herself. Additionally, the user may trigger alarm by himself/herself.

4 ADAPTIVE ALARM MODEL

Users behave differently and what might be alarm-worthy for one is normal for another. To give two extreme examples, one user might never voluntarily lie or sit on the ground because of a physical disability which prevents him/her from getting up again, whereas another might exercise regularly on the living room carpet. Our goal is to automatically adapt the alarm model to such differences, fine-tuning it to the needs and preferences of a particular user. This section presents the adaptation mechanisms of the rule-based and machine–learning alarm detection agents.

4.1 Rules adaptation

The problem of adapting the rule-based agents, whose agent function is represented in the form of a rule, is defined as a Markov decision process (MDP). Each rule can be thought of as a point in N-dimensional space, where N is the number of adjustable parameters in the rule. For example, the space of the rule »IF *person lying on the ground* **P**% *in* **T** *seconds* THEN *alarm*« is two dimensional, with one dimension representing the set of possible percentage (P) values and the other representing possible time interval (T) values (Figure 2). The goal state of the rule-based agents is dynamically changing as user's needs change. This adds uncertainty to the reward received for each agent action. Available agent actions in each state are all combinations of parameter value changes. Therefore, agent actions, unlike in problems usually modeled with MDP, are deterministic. The rule-based agents must determine optimal parameter values under uncertainty about their goal state.

The initial state of each rule-based agent is set to values predefined by domain expert. Returning to our example agent (Figure 2), using domain knowledge its agent function is initialized to »IF *person lying on the ground* **90**% *in* **8** *seconds* THEN *alarm*«.

The reward matrix is initialized to zero for all states (Figure 2a). Without any feedback from the user, we do not have any information whether there exists a state which better suits user's needs. The agent does not change its state as long as there is no state with better utility than the utility of the current state. For example, because all states have zero utility at the beginning, the agent does not change its state, i.e. rule parameters. When an alarm is raised and the user responds with false alarm, the reward of the current state and all states that are dominated by it is reduced by penalty amount *pa*, in our case -2. Then the utility of being in the current state is reassessed. When the reward matrix is updated after the first false alarm (Figure 2b), it is easy to see that the current parameter values are no longer optimal. The agent chooses randomly one of the states with best possible utility. In the example in this figure, after the first false alarm the agent function is changed to »IF *person lying on the ground* **100**% *in* **9** *seconds* THEN *alarm*«. When the user triggers alarm by himself/herself, the reward of all states which are dominated by the current state is incremented by reward amount *ra*, in our case 1 (Figure 2c). Again, the utility of being in the current state is reassessed. After the user triggered alarm, the agent function of the example agent is changed to »IF *person lying on the ground* **100**% *in* **8** *seconds* THEN *alarm*«. Notice how the initial state is avoided because of the negative reward received during the first false alarm. This way the rule-based agents autonomously adapt their parameter values to achieve optimal utility in an environment with non-deterministic goal state.

4.2 Machine learning adaptation

The agent function of the machine learning agents is adapted by re-inducing the alarm detection classifiers every time new alarm situation examples are obtained. User triggered alarms and false alarms provide new true or false alarm examples. In the case of a user triggered alarm, the instance at the point when the alarm is triggered by the user and all following instances that dominate that instance with respect to the amount of person lying at an inappropriate place are added as true alarm examples. In the case of a false alarm, all the instances that were classified as true alarms are added as false alarm examples to the dataset. In order to escape classifier bias in case of unbalanced dataset, after each addition, the weight of the database instances is updated in order to bring the ratio of true alarm to false alarm to neutral examples (includes standing, sitting and lying on the bed) to 40 to 30 to 30. A new machine learning classifier for alarm detection is built on this dataset

.

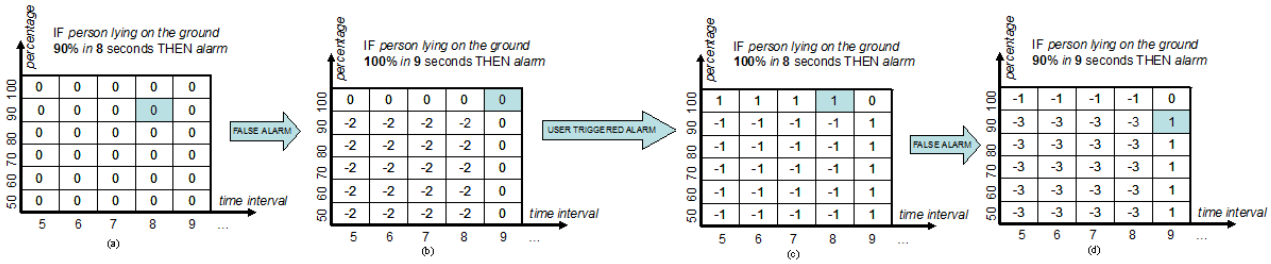

Figure 2 Rule-based agent adaptation

5 EXPERIMENTS

To design and test performance of Confidence a room resembling a studio apartment equipped with a bed, few chairs and tables was set up. The apartment was divided into five regions: kitchen, living room, toilet, sleeping area (bed) and corridor.

5.1 Data

In order to test the adaptation capabilities of the rule-based and machine-learning alarm detection agents we created a set of true and false alarm scenarios. True alarm scenarios include (1) user falling quickly and then lying on the ground moving for 15 s, (2) user falling quickly and then lying immovable for 15 s, (3) user falling slowly and then lying moving for 15 s and (4) user falling slowly and than lying immovable for 15 s. The cases (1) and (2) represent tripping; (2) results in an injury that prevents movement. The cases (3) and (4) represent falling due to dizziness or fainting. In all true alarm scenarios the user raised alarm after lying on the ground for 7 seconds. Three types of false alarm scenarios were also recorded: (1) user on all fours on the ground for 10 s, (2) user on all fours for 5 s, then lying on the ground for 5 s and (3) user lying on the ground for 10 s. The user is moving in all three cases. These scenarios may represent the user searching for things under the table or bed. They differ from the true alarm scenarios by the length the user stays on the ground and in some cases the amount of movement.

Each true and false alarm scenario was recorded five times for the purpose of training. Additionally, all scenarios were recorded five times for testing.

5.2 Results

We tested the adaptation capabilities of the rule-based and machine-learning alarm detection agents based on two measures: alarm detection accuracy and time-to-alarm. Each test started with agents that are not able to recognize any true alarm situation in the test sequences. The training sequences were provided to the agents one by one. After each new training sequence, the agents were adapted and their alarm detection accuracy and time-to-alarm was tested on each of the five test sequences. As previously stated, in all true alarm scenarios, the user triggered alarm was raised after 7 seconds. Designing true alarm scenarios this way, we intended to force the classifiers to reduce their time-to-alarm to 7 seconds or less. However, if the agents fire alarm in the true alarm scenarios within 7 seconds, they will also fire alarms in the false alarm scenarios since they last for 10 seconds. Designing the false alarm situations this way, we intended to force the classifiers to require longer period of person lying on the ground (more than 10 seconds) before they raise an alarm. Obtaining information how the alarm detection accuracy of the agents changes over time provides information how good the agents can distinguish true from false alarm scenarios. Measuring the time-to-alarm, on the other hand, provides information whether the agents separate true and false alarm sequences by the period of user lying on the ground, which we consider most relevant for this task.

In order to test how agent's alarm detection accuracy changes after each adaptation step, three test runs were executed. In the first, all true alarm scenarios were presented to the alarm detection agents first, followed by all false alarm scenarios. In the remaining two test runs, the true and false alarm training scenarios were given randomly to the agents. By adapting the agents by various orders of training scenarios, we gain information not only how their alarm detection accuracy improves, but also how much it depends on particular order of training scenarios.

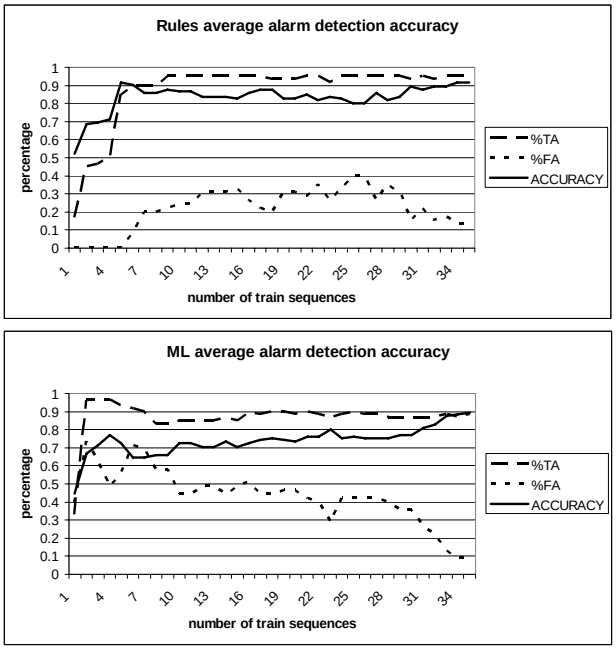

Figure 3 Average rule-based and machine learning agent alarm detection capabilities. *%TA* presents the percentage of correctly detected true alarms. *%FA* presents the percentage of alarms raised for false alarm test scenario examples. *ACCURACY* presents the overall achieved accuracy.

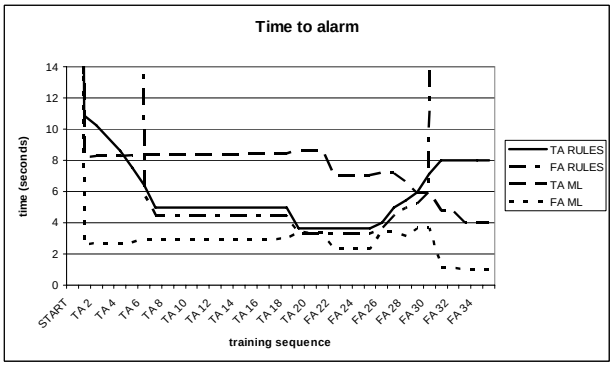

Figure 4 Average rule-based and machine-learning agent time-to-alarm. *TA RULES* and *TA ML* present average time-to-alarm for the true alarm test scenario examples for the rule-based and machine learning agent respectively. *FA RULES* and *FA ML* present their time-to-alarm for the false alarm test scenario examples.

Figure 3 presents how alarm detection accuracy of the rule-based and machine-learning based agents changes after each adaptation step. The *%TA* line presents the percentage of correctly raised true alarms in the test alarm sequences. The *%FA* line presents the percentage of incorrectly raised false alarms in the test alarm sequences. The *ACCURACY* line presents the overall alarm detection accuracy of the alarm detection agents. From this graph we can see that both rule-based and machine-learning alarm

detection agents can adapt to reliably detect the alarms presented in the test sequences, reaching an accuracy of 90%.

We tested how agent's time-to-alarm changes by presenting all true alarm scenarios to the agents first, followed by all false alarm scenarios. Our intention was to: (1) teach the agents to raise alarm when the user has lied on the ground for a period of 7 seconds or less after all true alarm scenarios are presented and (2) increase time-to-alarm to 10 seconds or more when the agents are presented all false alarm examples.

Figure 4 presents how agent's time-to-alarm changes in such test run. The *TA RULES* line presents average time-to-alarm for the true alarm examples in the test sequences for the rule-based alarm detection agent, whereas the *FA RULES* line presents its time-to-alarm for the false alarm examples. Time-to-alarm for the true and false alarm examples of the machine-learning alarm detection agent is presented with the lines *TA ML* and *FA ML*, respectively. Rule-based agent's time-to-alarm follows our expectations. It falls to around 4 seconds after the rules are presented with all true alarm training scenarios, then raises to 8 seconds after the classifier receives the false alarm ones. The machine-learning agent, on the other hand, does not follow our expectations. Its time-to-alarm stayed fairly constant when the true alarm training scenarios were presented to the agents, then started to fall when false alarm training scenarios were presented to it. The time-to-alarm fell to 4 seconds in the end. This means that machine-learning agents learned to separate true from false alarm scenarios not according to the length of person lying on the ground, but by some other attribute. As future work we need to test if the machine-learning agents have created alarm detection classifiers that work in the general case and whether they are not over-fitted to these particular scenarios.

6 CONCLUSIONS AND FUTURE WORK

We presented adaptation of rule-based and machine-learning alarm detection agents based on received user feedback in Confidence, ubiquitous care system to support independent living. Markov decision process formalism was used for adapting the rule-based agents, whereas machine learning was adapted by re-inducing the alarm detection classifiers every time new data is obtained.

Experimental results show that both agents are able to adapt their agent function according to provided user feedback. On average they both achieved alarm detection accuracy of 90% on the performed test runs. The rule-based agent followed our expectations with respect to the length of user lying on the ground (which we considered as the most relevant for this task) needed in order to fire alarm. User triggered alarms influenced their time-to-alarm to be reduced, where as false alarms influenced their time-to-alarm to be enlarged. On the other hand, machine-learning alarm detection agents did not follow our expectations. True alarms did not influence them by reducing their time-to-alarm. What is more, time-to-alarm was reduced when these agents were presented with false alarm training sequences. This means that the machine-learning agents found attribute other than the period of user lying on the ground for separating true from false alarm examples.

As future work we need to test whether the machine-learning alarm detection agents designed in this way work well in the general case and are not over-fitted to the particular scenarios used in this test. We also need to develop more intelligent way of choosing best rule-based agent state when the agent faces a situation with more than one state with the same highest utility.

ACKNOWLEDGMENTS

This research is partly financed by the European Union, European Social Fund, partly by the Slovenian Research Agency under the Research Programme *P2-0209 Artificial Intelligence and Intelligent Systems*, and partly from the European Community's Framework Programme FP7/2007–2013 under grant agreement No. 214986. We would like to thank Domen Marinčič, Rok Piltaver, Boža Cvetkovič, Damjan Kužnar and Blaž Strle for suggestions, discussion and help with programming.

REFERENCES

[1] Confidence. http://www.confidence-eu.org, 2010-05-19.

[2] T. Zhang, J. Wang, L. Xu, and P. Liu, 'Fall detection by wearable sensor and one-class SVM algorithm', *Lecture Notes in Control and Information Sciences*, pp. 858–863, (2006).

[3] M. Kangas, A. Konttila, I. Winblad, and T. Jamsa, 'Determination of simple thresholds for accelerometry-based parameters for fall detection', *Proceedings of the 29th Annual International Conference of the IEEE, Engineering in Medicine and Biology Society*, pp. 1367–1370, (2007).

[4] A. K. Bourke and G. M. Lyons, 'A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor', *Medical Engineering & Physics* 30, pp. 84 – 90, (2008).

[5] Z. Fu, E. Culurciello, P. Lichtsteiner, and T. Delbruck, 'Fall detection using an address-event temporal contrast vision sensor', *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 424–427, (2008).

[6] M. Lustrek, and B. Kaluza, 'Fall detection and activity recognition with machine learning', *Informatica* 33(2), pp. 197–204, (2009).

[7] AlertOne Services, Inc. iLife™ Fall Detection Sensor. http://www.falldetection.com, 2010-05-19.

[8] Zenio, 'Zenio Fall Detector', http://www.zenio.be/product/8.html, 2010-05-19.

[9] A. Cesta, and F. Pecora, 'Integrating intelligent systems for elder care in RoboCare', W. C. Mann and A. Helal (Eds): *Promoting Independence for Older Persons with Disabilities*, IOS Press, pp. 65-73, (2006).

[10] T. Pietraszek, 'Using adaptive alert classification to reduce false positives in intrusion detection', *Proceedings of the Symposium on Recent Advances in Intrusion Detection*, pp. 102-124, (2004).

[11] T. Lunt, 'Detecting intruders in computer systems', *Proceedings of the Symposium on Computer Security, Threats, and Countermeasures*, pp. 110–121, (1990).

[12] J. C. Georgia, 'Next generation intrusion detection: autonomous reinforcement learning of network attacks', *Proceedings of the 23rd National Information Systems Secuity Conference*, (2000)

[13] M. Hossain, and S.M. Bridges, 'A framework for an adaptive intrusion detection system with data mining', *Proceedings of the 13th Annual Canadian Information Technology Security Symposium*, (2001)

[14] Ubisense. http://www.ubisense.net, 2010-05-19.

[15] M. Lustrek, B. Kaluza, E. Dovgan, B. Pogorelc, and M. Gams, 'Behavior analysis based on coordinates of body tags', *Proceedings of the European Conference on Ambient Intelligence*, pp. 14–23, (2009).

[16] B. Kaluža, and E. Dovgan, 'Glajenje trajektorij gibanja človeškega telesa zajetih z radijsko tehnologijo', *Proceedings of the 13th International Multiconference Information Society* vol. A, pp. 97-100, (2009).

[17] V. Mirchevska, M. Lustrek, I. Velez, N. G. Vega, and M. Gams, 'Classifying Posture Based on Location of Radio Tags', *Ambient Intelligence Perspectives II*, pp. 85 – 92, (2009).