Using conventional cell phones to recognize basic physical movements

Oxana Tormashova, ETSETB UPC

***Abstract*--Human activity recognition system via wearable sensors can provide valuable information about person's life-style and can help to monitor human activities in order to improve person's interaction with environment. In current work we research possible approaches to development of an accelerometer sensor based system for human activity recognition. The movements are described with accelerometer data, gathered by means of sensor embedded in mobile phone. For each type of movements a feature vector is created and is further processed by a classification system to generate a classification rule. The result rule is used to recognize a particular human activity. The purpose is to develop a program that implements a human activity classification system for a mobile phone with sufficient accuracy.**

***Index Terms*--Accelerometer, Human activity recognition, Adaboost, Gaussian Models, KNN, KMeans.**

I. Introduction

IN modern time of widely spread mobile devices people do NOT imagine their life without mobile devices, such as PDAs, mobile phones, and other portable personal electronic devices. These devices are equipped with different types of sensors as a standard component of the control. The information from these sensor can be used in systems that are capable of automatic recognition and classifying specific physical activities of human beings. The aim of activity recognition system is to recognize the activity of its user based on previously monitored and analyzed data about behavior of the person, and take necessary actions in response. It may help improving the performance of healthcare monitoring devices or promoting the development of advanced human-machine interfaces.

The acceleration sensor, i.e. accelerometer, in particular, now among the standard features in most mobile phones and entertainment devices, can be used as a source of information about its owner's activity and motion. An accelerometer measures proper acceleration of a device and returns a real-time measurement of acceleration along the x-, y- or z-axis to be used as a human motion detector. Analysis of acceleration signals enables recognition of different type of human motion activities such as walking, running, standing up, etc, which is a rich source of context information for a mobile application. Accelerometers have been widely accepted due to their compact size, their low-power requirement, low cost, non-intrusiveness and capacity to provide data directly related to the motion of people.

Most of the systems proposed in the literature utilize data from an accelerometer in combination with data from other sensing devices like gyroscopes or EMG sensors in order to enhance the system's performance. It increases the number of required sensors for the classification system. The additional limitation is the number, location and nature of sensors that people will tolerate. That's why development of a single sensor system for human activities recognition is a challenging but promising research area. Activity recognition requires a technique that can achieve required level of reliability of recognition while being used under the conditions of daily living.

The aim of the current work is to research a possibility of building a recognition system based solely on data from a single 3-axis accelerometer and to develop a software that implements the algorithm of movement recognition on a mobile phone.

For a general analysis the signal processing flow is shown in a following figure:

Fig. 1. Signal process flow for accelerometer data.

II. Approaches to Human Activity Recognition

There is a bulk of human motion analysis literature. Many techniques have already been proposed for activity recognition in specific environment (e.g. laboratory) using the cooperation of several sensors. Three mainly utilized approaches for activity recognition are: video based, environmental sensor based and wearable sensor based.

A. Video based systems

This type of system use video cameras to obtain information about human activities. These systems works good enough in the specific environment, but have a significant drop of accuracy in real home settings [1]. Changes of the scene, variable light, increasing number of people in the scene etc, provides a significant challenges while processing of the information. The other limitation of such systems is that they are tied to a specific location and can't be used for monitoring activities of a person outside of

this location. In addition, the cost of utilized sensor such as microphones and camera are high enough.

B. Environmental sensor based systems

The aim of these system is to monitor the interaction between users and their home environment [1, 2]. They use a set of ambient sensors distributed through the person's living environment. The system monitor the occupants of the home all day. The data gathered by these sensors can be used in order to intelligently adapt the environment to persons needs. Like the video based systems environmental systems depend on location and can't monitor outside of the living environment. They also are infrastructure dependent.

C. Wearable sensor based systems

Such systems utilize wearable during normal daily activity sensors. They continuously monitor bio-mechanical and physiological data of the person independent of his location. This type of sensors can be attached to a person by means of clothes, jewelry or worn as independent wearable devices. They can measure physical parameters that can not be measured by other types of sensor systems and thus are well suited for collecting data of human activities in order to classifier and recognize activity patterns. In addition this type of sensors are low-priced and do not invade as actively peoples privacy as video sensors for example.

There are different types of body-attaches sensors, such as electromechanical switches, goniometers, accelerometers, gyroscopes, pedometers, and actometers, that can be used to monitor human movements in free-living environment. Of all these accelerometers were considered to be a useful tool for collecting physical information about human movements. By means of accelerometers it's possible to gather information about both frequency and intensity of movement. Some type of accelerometers can in addition measure body tilt.

Accelerometers are miniature and low cost and is useful for development of small lightweight, portable systems that can be worn by person during day without too much inconvenience for the person.

In this work we try to develop and implement in programming code a movement recognition system based on a single tri-axial accelerometer sensor embedded in a mobile phone.

III. Related Work in Activity Recognition using Accelerometer

Most studies on the use of wearable devices in monitoring have used multiple sensors, typically accelerometers fixed to specific places on the body, usually a subset of the thighs, wrists, arms, sternum, waist and lower legs[3, 4, 5]. The limitation of such approach is number and location of these devices.

A smaller number of studies have investigated the use of a single accelerometry device attached at the waist, sternum or back[6, 7]. A single sensor can be integrated into a mobile device, such as cell phones or wristwatches, and is more comfortable for the person in daily use. In order to compensate for one sensor it can be located in different places on the person.

Although the use of a larger number of sensors is likely to provide a higher accuracy of movement classification, such a system may be uncomfortable and inconvenient during long-time monitoring of daily activities. In [13] was shown that placing accelerometer at only two locations (either the hip and wrist or thigh and wrist) did not affect activity recognition scores significantly (less than 5%) when compared to a system with five sensors.

The researches on movement classification from accelerometer data implemented wide variation of classification algorithm. In order to implement classification algorithm the software has to learn to recognize and associate activity patterns. This is called machine learning.

There are two approaches to such techniques[15]: supervised or unsupervised. In case of supervised technique there is a set of activity data previously labeled and used to “train” classification algorithm. Once the training phase is complete, the classifier is able to assign an activity label to an unknown example of sensor data. With unsupervised approaches all the sensor data are passed to the algorithm which automatically identifies a number of states or data clusters, each of which may correspond to a particular activity.

The range of classification methodologies can be divide into the following types[15]:

A. Threshold-based classification

The idea of the threshold-based classification is to compare obtained feature value with the specified threshold in order to decide whether a particular activity is being performed. Due to its implementation this approach is useful for discrimination between static postures and transitions between them using angles derived from accelerometers placed on different parts of body. Some of the researches have also applied threshold-based classification to the problem of differentiation between static postures and dynamic activity. In addition threshold-based classification has been successfully applied to the detection of falls. A range of features can be used in order to detect fall accident. The most common characteristic used to identify the presence of a fall is the rapid deceleration which occurs as the faller contacts the ground. The value of the threshold depends on location of the sensor. It is also possible to combine different thresholds in order to achieve better accuracy during classification[16, 17].

B. Hierarchical methods

A hierarchical classification scheme construct a binary decision structure which is consists of of a number of consecutive nodes. At each node a decision rule is applied to input features. The result is either a final classification or a

transition value for feather classification process. The disadvantage of this approach is that the decision rules have to be determined before the testing based of manual inspection and analysis of the training data.

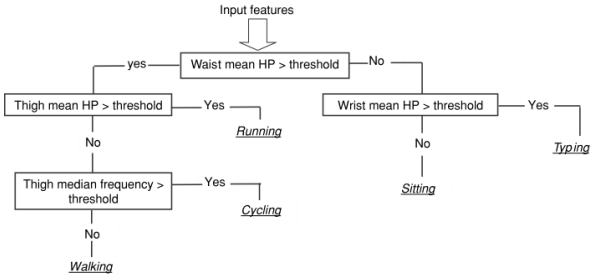

Fig. 2. Example of the hierarchical classification scheme.

Many researches utilized hierarchical in combination with other classification schemes [7 , 19].

C. *Decision trees*

The decision tree approach is similar to hierarchical scheme. The difference is that decision trees use strict algorithms in order to define a set of rule for classification and thus automate the process. These algorithms examine features one at a time to determine the most suitable ones for discrimination of the activities and develop a set of rules that will be later used in a classification system.

Decision trees has been applied to solve a wide range of classification problems[21, 22, 23, 46]. One of the most thorough studies was carried out in [20]. The authors combined time and frequency features and used five sensors to recognize 20 activities with an accuracy of 86%.

D. *K-nearest neighbor*

K-nearest neighbor algorithm (k-NN) is a method for classifying objects by constructing multidimensional feature space, in which each dimension corresponds to a different feature. First training data is placed in a feature space based on closure metric.

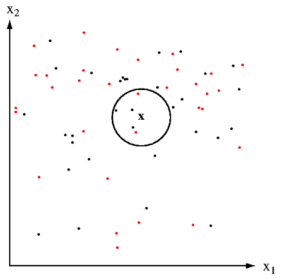

Fig. 3. Example of the KNN classification.

Then a new data is classified by defining the majority of the k-nearest neighbors which correspond to a given activity. The k value is varied from 1 to a small percentage of training data and is usually selected empirically.

E. *Artificial neural networks*

Artificial neural network (ANN) is a mathematical model or computational model that represents complex relationships between its inputs (independent variables) and outputs (dependent variables). ANN is an adaptive system that characterized by some form of optimization process that permits to obtain the output values for given set of inputs. First ANN is run over the training set, after that it can then be used to obtain the outputs for any set of inputs.

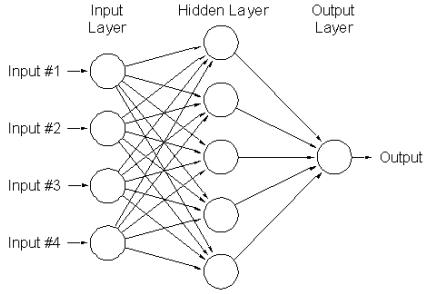

Fig. 4. An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in the human brain.

In case of activity classification ANN inputs usually are the sensor data and outputs - different classes of activities.

ANNs have been widely used within the field of human movement research.

One of the most common ANNs is referred to as a multilayer feed-forward neural network or Multi-Layered Perceptron (MLP) described in [30, 31]. In it input and outputs are interconnected through hidden layers, the flow of information through the network is controlled by the weighting of the links between the nodes and the transfer function within each node. This type of network is trained by iteratively optimizing the weights in order to accurately produce the desired training outputs from the corresponding inputs. MLP was used in many researches, such as [32, 33, 34] .

An alternative to the feedforward ANN is the probabilistic neural network[35]. This type of network assume to have a example patterns for classification stored in memory and thus reduce time of training.

Spiking or pulsed neural networks (PNN) [36, 37] work with much larger number of binary input that normal ANN and behave relatively well with accelerometer-base data[38].

F. *Support vector machines*

Support vector machines (SVMs) [24] establish a popular machine learning method which is based on finding optimal separating decision hyperplanes between classes with the maximum margin between patterns of each class. In short it constructs for given example a classification between two classes, making it a non-probabilistic binary linear classifier. For a set of training data this algorithm is implemented on each of the examples marking it as belonging to one of two categories. The result model is used for assigning test data to a specific category.

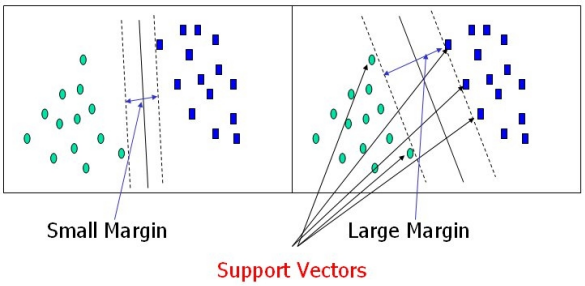

Fig. 5. A two-dimensional example of SVM. The SVM analysis attempts to find a best 1-dimensional hyperplane (i.e. a line) that separates the cases based on their target categories. The vectors (points) that constrain the width of the margin are the support vectors.

SVMs have only been applied in a small number of activity classification studies[6, 25, 26]. Three studies have used SVM techniques to differentiate between simulated falls and other activities. In [27] data from accelerometer were combine with data from microphone in order to differentiate between falls, walking and running. In [28, 29] data from from a tri-axial accelerometer embedded in a mobile phone were used for a fall recognition.

G. Naive Bayes and Gaussian mixture models

The Bayesian classifier is based on the estimated conditional probabilities or likelihoods of the signal patterns available from each activity class. The probability of unknown example being generated by a specific activity is estimated by means of likelihood function.

With a naive Bayes classifier, the input features are assumed to be independent of each other. Under this assumption it is possible to present likelihood function for each class as a mixture of n simple probability density functions, where n is the number of features. In reality the assumption of independence between features is often violated but this approaches is popular due to its simplicity and ease of implementation[43, 44, 45]. The approach was studied in many researches but sometimes produce unstable results. In [3] authors suggested that the reason for this poor performance may have been the violation of assumptions that acceleration features can be considered conditionally independent and modeled by a normal distribution.

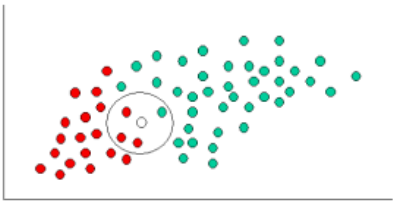

Fig. 6. Example of Bayesian classifier. The likelihood is measured by drawing a circle around X(white circle) which encompasses a number (to be chosen a priori) of points irrespective of their class labels. Then a new object (white circle) is classified as a RED one, based on maximum likelihood value.

A Gaussian mixture model (GMM) [10, 31, 9] is very similar to Bayes classifier. However, the likelihood function is to be of unknown shape and functional form and thus approximated by a weighted mixture of Gaussian functions.

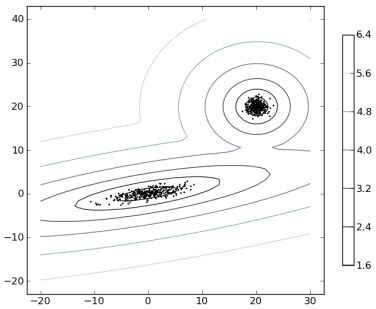

Fig. 7. Two-component Gaussian mixture model: data points, and equi-probability surfaces of the model.

To improve the quality of GMM it is usual to fit these models many times with different parameters and choose the best result, as measured by the likelihood or some other external criterion.

One of the most popular approximate inference algorithms that is used to calculated parameters and mixing coefficient of Gaussian components is the expectation-maximization algorithm (EM). Expectation-maximization is a well-fundamented statistical algorithm to get around this problem by an iterative process and it's the fastest algorithm for learning mixture models.

But there are also other approaches. For example in [39] time-domain features were used to construct separated GMMs for a number of movements. For training instead of EM they employed a statistical estimate proposed in the field speech recognition. Classification of test data was achieved by selecting the GMM (activity) with the highest probability of having produced that particular set of features.

H. Fuzzy logic

Fuzzy logic is derived from fuzzy set theory and allows mapping from set of inputs to one or more outputs utilizing a set of if–then statements called rules. Fuzzy logic allows input data to have a partial membership in multiple sets. During the training process a training data first is assigned to fuzzy sets. Then the rules is applied to produce a corresponding output. In case of activity classification the example with the maximum membership is selected.

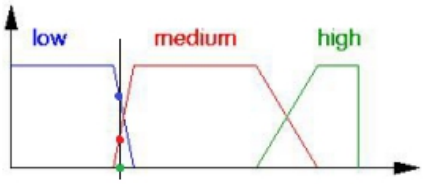

Fig. 8. Example of fuzzy logic membership functions

Although fuzzy logic should be more suitable for dealing with real-world problems than normally used hierarchical or decision tree classification schemes it's been applied to a limited number of activity classification problems f.e. to identify different static postures or to differentiate between

different movements [40, 49], to identify sit-to-stand and stand-to-sit transitions [41], to identify falls [42].

I. Markov chains and hidden Markov models

For specific movements, some transitions between activities are more likely to occur than others. For such problem can be used hidden Markov models (HMM), that represents Markov chains with unknown (or hidden) state of the model at any given time. The states can only be determined from observable parameters which depend on the state.

For classification purpose the features obtained from sensor data are defined as observable parameters and different activities correspond to the state of the model. States in a HMM can correspond to more than one activity. As previously describe techniques HMM is firstly trained with example data. Then it can then be used to determine witch sequence of state transitions is the most likely could have resulted from an observed sequence of features. HMMs are trained by determining state transitions along with the probabilities that each possible set of observations (features) will be observed for a given state.

HMMs is one of the most popular approaches in literature[47, 48, 50]. It can be used as a single classifier just like a part of a two-stage classification scheme (e.g in combination with boosting algorithm[47]).

J. Combining different classifiers

Recently meta-stage schemes for classification analysis has gained popularity. They improve performance of single classifiers by combining their output using different techniques. These include majority voting (where the class with the majority of votes is accepted), stacked generalization (which trains the base classifiers and then uses their predictions as data to a new learning stage) or boosting (which assigns weights to the training patterns to combine the performance of weak classifiers). In [23] five base-level classifiers in a boosted scheme for eight common activities were studied. In general, when an inter-subject design was used, the boosted SVM was shown to outperform other meta-level classification schemes.

AdaBoost is a type of adaptive boosting that adapt multiple weak classifiers to create a single more reliable one.

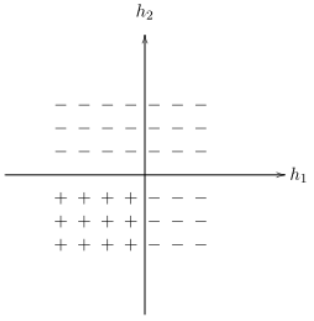

Fig. 9. Example of AdaBoost with two weak hypothesis. It combines two learners, h1 and h2, in order to obtain a “good” learner.

Neither h1 nor h2 is a perfect learner. Initially AdaBoost chooses the one that classifies more data correctly. Then the data is re-weighted to reduce the impact of the misclassified data[51]. This process continues and at each step the weight of each week learner among other learners is determined. The obtained at the end of iterations combination of “weak” learners will be a “good” learner, that classifies the given data more accurately that each learner separately.

IV. Classification Procedure

Classification process consists of training phase and test phase.

The training phase includes processing the labeled set of movement data and generating a classification rule for each movement as the result. The training were carried out using whole learning data, generating classification rule for each type of movement.

Test phase were conducted in two steps: for the set of data that were previously included for generating classification rule in training phase; and for independent set of movement data, that were not used for generation of the classification rules. Accuracy was calculated as follows:

$$Accuracy = \frac{number\ of\ correctly\ recognized\ movements}{number\ of\ movements}$$

Test data is automatically labeled in order to evaluate the recognition accuracy of the test phase.

V. Acceleration Data

For the current work was used a smart-phone with Android OS and a a built-in tri-axial accelerometer. The accelerometer records linear acceleration information along x, y, z axis that is applied to a device itself.

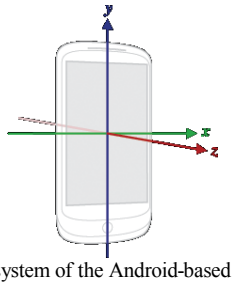

Fig. 10. The coordinate-system of the Android-based mobile phone. It is defined relative to the screen of the phone in its default orientation. The axes are not swapped when the device's screen orientation changes.

When the mobile phone is placed in the upright position the y-axis acceleration direction reflects the up and down body movement, forward movement was along the z-axis and side movement was assigned to the x-axis.

In the experiment performed in this research, the movement data collection the phone were held in the hand while moving.

During collection the change of acceleration was measured while the subject was repeating postures such as standing, sitting, walking, falling, going upstairs and downstairs. The acceleration data was collected during a short directed routine performed by a person and was recorded by a mobile phone

application in a text file with corresponding title in a following format: time-stamp, x acceleration, y acceleration, z acceleration.

The set of instructions for data collection can be summarized as following:

1. Select movement type.
2. Press the button to start the testing.
3. Wait a few seconds before the recoding starts.
4. Make the motion.
5. Press stop to end recording.

Different physical activities results in different patterns in data provided by acceleration sensors, and thus can be classified accordingly. The example of gathered data captured by acceleration sensors over time (samples) can be seen on the following figures:

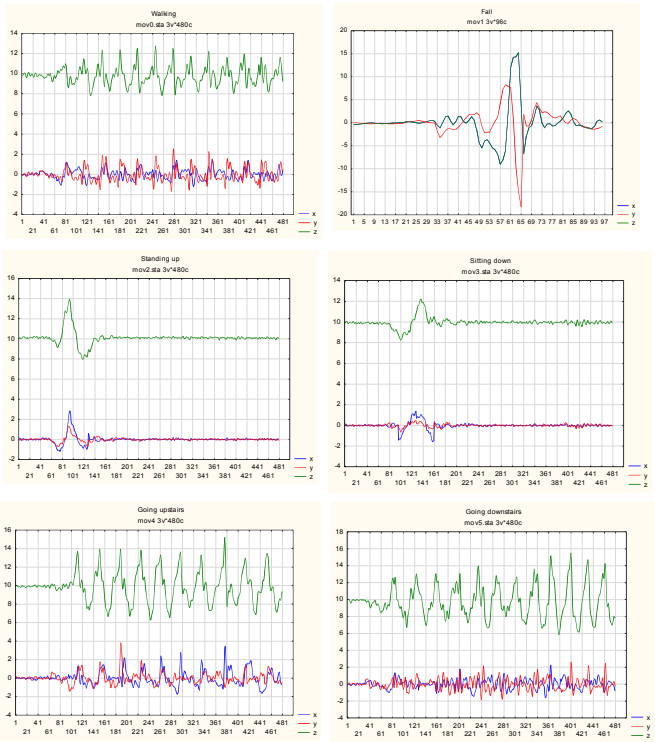

Fig. 11. The acceleration values for x, y, z direction for each observed movement.

Different activities has been placed on the figures. As can be seen, there is a significant difference in pattern of acceleration for different activities.

For the purpose of and in order to remove to remove the dependence on phone screen angle gathered data was preprocessed by calculating in addition the angle of inclination with respect to the horizontal plane for gathered data.

Data processing of training data was performed offline, after a recording had been completed. The result of this was the classification of training data that can be used later in mobile application in order to classify the new movement in real-time.

VI. Windowing techniques

Most classification methods before processing sensor data divide the sensor signal into smaller time segments by means of different windowing techniques. Then classification algorithm is applied for each window data separately. The information then combined in order to generate an activity profile for the whole signal.

In activity monitoring three windowing techniques are used: sliding windows, event-defined windows and activity-defined windows.

In case of sliding widow the signal is divided into small windows of fixed length without gaps between them. As a variation of sliding window exist overlapping sliding widow approach.

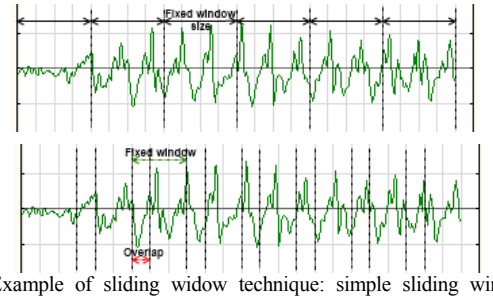

Fig. 12. Example of sliding widow technique: simple sliding window and overlapping sliding window.

This approach doesn't require preprocessing of the sensor signal and is very simple in implementation, thus can be used in real-time applications. Due to simplicity of this approach it's used by the most of movement recognition systems. In this work exactly this approach was used.

In event-defined approach the signal is preprocessed in order to locate specific events as points for and defined successive windows. As events may not be uniformly spaced in time, the window size is not fixed.

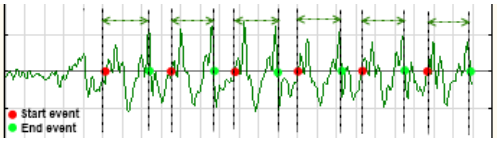

Fig. 13. Example of event-defined widow technique

In case of activity-defined windows the times when activities changes is determined. These points define the definition of sensor data into windows, each of of which correspond to a different activity.

Fig. 14. Example of activity-defined widow technique

VII. Feature selection and extraction

For the purpose of automatic classification of acceleration data it should be preprocessed into a subset of feature variables with high information content.

In order to moderate rapid and dramatic change in gathered accelerometer values a simple Kalman filter was implemented. The data was processed through filter before extraction of the features. The parameters of the filter are the following: observation H = 1, state transition F = 1, noise covariances Q = 5 and R = 5 , the control-input model B = 0 and initial covariance = 0.1

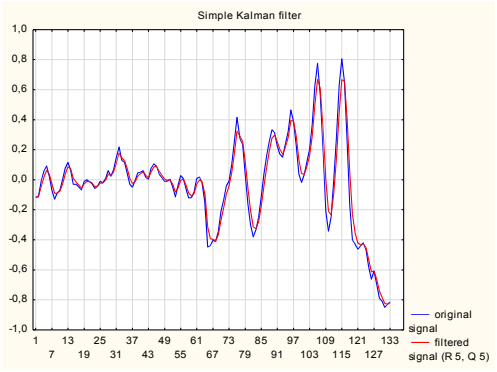

Fig. 15. Original x-acceleration and filtered signal. The chosen parameters for Kalman fiter: H=1, F=1, R=5, Q=5, initial covariance = 0.1.

In addition to x, y and z accelerometer signals were calculated a magnitude signal as follows:

$$\bar{s}_{res}(\bar{t}) = \sqrt{\bar{s}_x(t)^2 + \bar{s}_y(t)^2 + \bar{s}_z(t)^2}$$

Time domain features are obtained directly from accelerometer signals and usually represent statistical values. In time domain for each acceleration and magnitude signals were calculated: minimum and maximum values, mean values, standard deviation, correlation between axis, root mean squared acceleration, interquartile range and zero crossings (number of sign changes in the segment).

For frequency domain features obtained from accelerometer data transformed into the frequency domain using a fast Fourier transform (FFT). The aggregated FFT signals and energy were calculated in order to characterize the spectral distribution.

In the dataset, the acceleration data stream from one movement was divided using a sliding window approach into overlapping rectangular windows 2.5 seconds each. Over each window a vector of the previously described features was calculated. The segmentation is done for all three acceleration signals x, y, z.

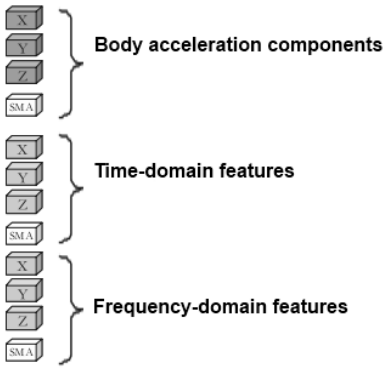

Fig. 16. Types of features used in current study.

The segments at this stage are still represented as time series. Then for each segment one feature vector is created that consists of all previously described features.

VIII. Methods used in study

In this work were used three methods of classification in order to specify the most suitable one for movement recognition.

A. K-Means Clustering Algorithm

K-means is a very simple and fast algorithm for clustering data. It computes the geometric center of samples belonging to the same category in training set. The test sample is classified by finding the nearest center.

It requires very little resources for training phase and thus is a possible candidate for a classification algorithm that can be used in a real-time movement recognition system. In movement recognition systems with wearable sensors K-means algorithm mostly used for filtering or combining data before processing it through main classifier.

In [53] K-means algorithm was used to recognize touch gesture types in unsupervised analysis. The other study [54] used k-means to construct a compressed dataset for the SVM classifier in order to reduce redundant information in the original data set.

B. k-Nearest Neighbors Classification

The second algorithm that were implemented in current study is k-Nearest Neighbors classification. It is one of the most well-known and widely used nonparametric pattern classification methods. There are two basic problems that are yet to be resolved by the research community and can be viewed as serious disadvantages. The first issue is the selection of the best number of neighbors to consider. The second issue is the computational and the storage issue. The traditional KNN algorithm requires the storage of the whole training set which may be an excessive amount of storage for large data sets and leads to a large computation time in the classification stage.

Despite its shortcomings k-Nearest Neighbors is one of the most popular classification algorithms as it's simple in implementation, effective and fast.

C. AdaBoost Classifier

The general idea of boosting algorithm is to try to build a “good” learning algorithm based on a group of ”weak” classifiers. The one of the most popular machine learning algorithms AdaBoost was proposed by Freund and Schapire in 1995 [11].

As AdaBoost is an algorithm that works under supervising, a training set of data has to be label previously. This labeled data set should contain both correct samples labeled as “+1” and incorrect samples labeled as “ − 1.” One of the main ideas of the algorithm is to maintain a distribution or set of weights over the training set. For each

example x_i the weight of this distribution is defined. Initially, all weights are set equally, but on each round, the weight of incorrectly classified examples increases and thus decrease its impact on the result strong classifier.

An individual weak classifier is simple and easy to implement. Its classification accuracy is relatively low. To improve accuracy a strong classifier is obtained as a combination of weak classifiers. The strong classifier will have a higher classification accuracy than each weak classifier.

The implementation of AdaBoost is simple and depends on choice of "weak" classifier. As a weak classifier in this work we use a simple threshold-based classification. The algorithm of it is the following:

Given: Input sequence of N feature vectors $x(x_1..x_N)$, where each $x \in R^d$, number C of clusters to partition the data set and maximum number of iterations T.

1. Fixed one of the features. Select and order values for a chosen feature in training dataset.
2. Set threshold.
3. Make hypothesis of the correct location of the feature regarding selected threshold.
4. Find the threshold that produce minimum error for the current feature under that assumption.
5. Make inverse hypothesis.
6. Select the threshold with minimum error for the current feature.
7. Repeat.

The general algorithm of AdaBoost is the following [12]:

Given: $(x_1, y_1), \ldots, (x_m, y_m)$
where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$.
- Update:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where Z_t is a normalization factor (chosen so that D_t+1 will be a distribution).

Output the final hypothesis:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

After obtaining the weak hypothesis h_t AdaBoost chooses the parameter α_t. This parameter measures the importance of hypothesis. This value is larger the smaller is error for the hypothesis.

The final hypothesis H is a weighted majority vote of the T weak hypotheses where a_t is the weight assigned to h_t.

The result of running AdaBoost over training data is a set of threshold for specific features for each movement. To classify test data, for each example each hypothesis is verified and the one with the minimum error is selected.

D. Gaussian Mixture Models

Gaussian Mixture Models (GMMs) are parametric representation of probability density functions, based on a weighted sum of multivariate Gaussian distribution. The Gaussian classifier produces a mixture of class-conditional probability density $p(x|\lambda_i)$ for each class λ_i under assumption that it has a a Gaussian distribution.

Each cluster for training data is interpreted as a hyperplane in a high dimensional space and is modeled as a GMM with specific parameters. The aim of clustering is to obtain parameters of the cluster hyperplanes that maximize a likelihood function of data memberships. GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm.

A Gaussian mixture model is a weighted sum of M component Gaussian densities as given by the equation,

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{M} w_i \, g(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i),$$

where x is a D-dimensional continuous-valued data vector (i.e. features), w_i, i = 1 … M, are the mixture weights (mixing coefficients), and $g(x|\mu_i, \Sigma_i)$, i = 1 … M, are the component Gaussian densities. Each component density is a D-variate Gaussian function of the form,

$$g(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x}-\boldsymbol{\mu}_i)\right\},$$

with mean vector μ_i and covariance matrix Σ_i. The mixture weights satisfy the constraint that

$$\sum_{i=1}^{M} w_i = 1$$

The complete Gaussian mixture model is parametrized by the mean vectors, covariance matrices and mixture weights from all component densities. These parameters are collectively represented by the notation,

$$\lambda = \{w_i, \ \boldsymbol{\mu}_i, \ \Sigma_i\} \qquad i = 1, \ldots, M.$$

The example of the GMM is shown on the following figure.

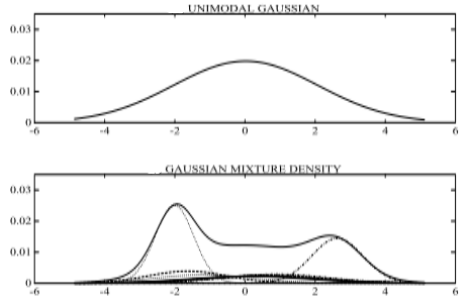

Fig. 17. Types of Gaussian Mixture Models.

The covariance matrices, Σ_i, can be full rank or constrained to be diagonal. The choice of model configuration (number of components, full or diagonal covariance matrices, and parameter tying) is often determined by the amount of data available for estimating the GMM parameters and how the GMM is used in a particular biometric application.

According to [10] because the component Gaussian are acting together to model the overall feature density, full covariance matrices are not necessary even if the features are not statistically independent. The linear combination of diagonal covariance basis Gaussians is capable of modeling the correlations between feature vector elements. The effect of using a set of M full covariance matrix Gaussians can be equally obtained by using a larger set of diagonal covariance Gaussians.

The algorithm implemented in current work are the following:

Given: Input sequence of N feature vectors $x(x_1..x_N)$, where each $x \in R^d$, number C of clusters to partition the data set, number K of Gaussians for each cluster and the maximum number of iterations T.

1. C = 0;
2. Initialize iteration t = 0:
   Initialize Gaussian parameters: means μ_i (obtained from data by running K-means over it), covariances Σ_i and $w_i = 1/N$ for i =1...N. One for each Gaussian k.
3. *E step.* For each point x_i determine its assignment score to each Gaussian k:

$$\gamma(z_{nk}) = \frac{w_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} w_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

$\gamma(z_{nk})$ is called a "responsibility": how much is this Gaussian k responsible for this point x_i

4. *M step.*

Given scores, adjust μ_i, w_i, Σ_i for each Gaussian k.

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

Mean of Gaussian k:

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$$

Covariance matrix of Gaussian k:

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^{\text{T}}$$

Mixing Coefficient (weights) for Gaussian k:

$$w_k^{\text{new}} = \frac{N_k}{N}$$

5. Evaluate log likelihood. If likelihood or parameters converge, stop. Else go to Step 3 (E step).

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, w) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} w_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

6. Next cluster C.

The idea is to run a back-end EM-Gaussian classifier over data for each specified movement. The initial parameter for each GMM was obtained from K-means algorithm. The result of the classification is a set of GMMs that describe each class. Then this set of GMMs is used for recognition of movements. For each example in given test data a likelihood function for each type of movement is calculated and the class that provide a maximum likelihood is selected.

IX. Test Results

In current study each of the selected classification algorithms was developed and tested in the NetBeans IDE 7.1.2 environment, Java 1.6 platform and Windows 7 operating system. Tests were made for the teaching and the recognition phase separately.

On mobile platforms Android 4.1 the software was developed in order to gather data for observed movements and test the results. There were observed six types of basic movements, such as: walking, falling, standing up, sitting down, going upstairs and going downstairs.

Training data consisted of files with stored accelerometer information for each observed movement. Test data consisted of two sets: data that were included in training set and data that wasn't used during training phase respectively.

The tests were conducted for features obtained from original data and from filtered data. As filter was used simple Kalman filter described previously in VII part.

The result aggregated accuracy for each classification algorithm is presented in the Tables I, II, III. Here K-means algorithm was tested in two modes: for each tested sample separately and for all tested samples taken together. KNN algorithm was tested for different parameter of neighbors taken into account. And GMM algorithm was tested with seven initial number of Gaussians in the model.

TABLE I
Accuracy of Classification Algorithms for All Features

	All features			
	Without Kalman filter		With Kalman filter	
Algorithm	Accuracy for test files excluded from training	Accuracy for test files included in training	Accuracy for test files excluded from training	Accuracy for test files included in training
K-Means	0,37	0,53	0,29	0,48
K-Means All	0,38	0,54	0,3	0,49
KNN5	0,66	0,86	0,66	0,8
KNN9	0,68	0,73	0,64	0,77
KNN11	0,74	0,75	0,66	0,77
AdaBoost	0,72	0,9	0,83	0,88
GMM5	0,2	0,44	0,38	0,51
GMM9	0,44	0,64	0,42	0,6
GMM11	0,46	0,66	0,42	0,64
GMM13	0,35	0,44	0,55	0,62
GMM17	0,5	0,53	0,09	0,1
GMM19	0,18	0,06	0,4	0,4
GMM23	0,44	0,48	0,25	0,28

TABLE II
ACCURACY OF CLASSIFICATION ALGORITHMS FOR FEATURES IN TIME-DOMAIN

	Time domain features: mean, stdev, correlation			
	Without Kalman filter		With Kalman filter	
Algorithm	Accuracy for test files excluded from training	Accuracy for test files included in training	Accuracy for test files excluded from training	Accuracy for test files included in training
K-Means	0,22	0,26	0,2	0,35
K-Means All	0,23	0,27	0,21	0,36
KNN5	0,38	0,62	0,33	0,6
KNN9	0,38	0,51	0,37	0,51
KNN11	0,35	0,55	0,33	0,46
AdaBoost	0,68	0,73	0,57	0,6
GMM5	0,61	0,6	0,33	0,3
GMM9	0,29	0,42	0,4	0,31
GMM11	0,33	0,35	0,46	0,46
GMM13	0,4	0,46	0,61	0,64
GMM17	0,35	0,46	0,51	0,51
GMM19	0,5	0,46	0,4	0,64
GMM23	0,5	0,51	0,48	0,64

TABLE III
ACCURACY OF CLASSIFICATION ALGORITHMS FOR FEATURES IN FREQUENCY-DOMAIN

	Frequency domain features: energy, FFT			
	Without Kalman filter		With Kalman filter	
Algorithm	Accuracy for test files excluded from training	Accuracy for test files included in training	Accuracy for test files excluded from training	Accuracy for test files included in training
K-Means	0,35	0,53	0,27	0,48
K-Means All	0,36	0,54	0,28	0,49
KNN5	0,68	0,86	0,68	0,82
KNN9	0,7	0,82	0,7	0,75
KNN11	0,7	0,77	0,7	0,71
AdaBoost	0,6	0,82	0,7	0,75
GMM5	0,22	0,42	0,4	0,57
GMM9	0,42	0,62	0,38	0,53
GMM11	0,29	0,64	0,4	0,62
GMM13	0,33	0,68	0,4	0,62
GMM17	0,42	0,73	0,29	0,55
GMM19	0,27	0,68	0,35	0,51
GMM23	0,27	0,48	0,4	0,68

The best results in all cases is produced by AdaBoost classification algorithm. Because of the selected weak learner the more features are introduced – the best results we obtain. The introduction of the filter improves the results of classification. The classifier performance of 83% for filtered data not used in training phase is decent enough for gathered accelerometer data.

The size of training data affect performance of AdaBoost classifier quite markedly. The more data were utilized in training phase the better result will be achieved for recognition of new activity.

TABLE IV
ACCURACY OF ADABOOST CLASSIFICATION FOR DIFFERENT NUMBER OF TRAINING DATA

	All features			
	Without Kalman filter		With Kalman filter	
Algorithm	Accuracy for test files excluded from training	Accuracy for test files included in training	Accuracy for test files excluded from training	Accuracy for test files included in training
AdaBoost (half of training data)	0,59	0,73	0,68	0,73
AdaBoost (all training data)	0,72	0,9	0,83	0,88

GMM algorithm that were implemented in current study works better with time-domain features. But the achieved accuracy is too low to be utilized as a movements classifier. The reason for this may have been the assumptions made earlier while implementing of the algorithm in order to reduce the complexity of implementation (we assume that the covariance matrix for each Gaussian is diagonal which simplifies the implementation significantly).

Another subject that affect the performance of GMM classifier is that it is very sensitive to initial conditions. So it's necessary to run train phase with different parameters in order to find the best model that describes observed movements.

K-Means, KNN and GMM algorithms provide better results without filtering. It is not clear exactly why this is. My guess is that maybe some of the filtered data allow to produce a stronger identifying feature for different movements and thus affect the accuracy of the classifier.

KNN algorithm with number of neighbors equal to 9 provided the best result in frequency domain. It's has accuracy of 70%. But in case when only time-domain features are utilized there is a significant drop in accuracy.

Tables V and VI present how exactly movements were recognized during the test phase for KNN9 and AdaBoost classifiers respectively.

TABLE V
KNN9 CLASSIFIER: DISTRIBUTION OF RECOGNIZED MOVEMENTS

	Walk	*Fall*	*Stand up*	*Sit down*	*Stairs up*	*Stairs down*
Walk	0,92	0	0	0	0,08	0
Fall	0	0,6	0,4	0	0	0
Stand up	0	0	0,2	0,8	0	0
Sit down	0	0	0,4	0,6	0	0
Stairs up	0,22	0	0,11	0	0,66	0
Stairs down	0	0	0	0	0,6	0,4

TABLE VI
ADABOOST CLASSIFIER: DISTRIBUTION OF RECOGNIZED MOVEMENTS

	Walk	*Fall*	*Stand up*	*Sit down*	*Stairs up*	*Stairs down*	Undentified
Walk	0,76	0	0	0	0,08	0,12	0,04
Fall	0,2	0,4	0,2	0	0	0	0,2
Stand up	0	0,2	0,6	0,2	0	0	0
Sit down	0	0	0,6	0,2	0	0	0,2
Stairs up	0,11	0	0	0	0,88	0	0
Stairs down	0	0	0	0	0,4	0,6	0

Despite of the good result that provide KNN algorithm it's not very practical for mobile recognition system as it requires the storage of all training data for recognition purpose. In case of AdaBoost training data is only necessary for obtaining classification rules.

X. FUTURE WORK AND CONCLUSIONS

The classifier performance of 83% achieved with Adaboost algorithm is the best result we were able to obtain for gathered data. It is noticeable that the introduction of the filter for the accelerometer signal before obtaining features helps to improve classifier performance. To further improve results in future we can try different filter parameters. Another possible way to improve accuracy of the implemented AdaBoost is introduction of new features such as Cepstral Coefficients, Spectral Entropy, etc.

It can also be promising to see more into GMM algorithm and try to implement a more generalized version of it.

In present study for KNN and K-means algorithms an Euclid distance were used as a metric for the distance

between samples. In future studies it'll be useful to explore the usage of other metrics for improvement of algorithm performance.

The current study investigated accelerometer-based activity recognition algorithms. An experimental study was carried out for activity classification in everyday life by using accelerometer embedded into mobile phone. Different classification techniques were implemented and evaluated in order to determine the best classifier for activity recognition.

XI. References

[1] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in Pervasive Computing. Springer Berlin / Heidelberg, 2004, pp. 158–175.

[2] T. van Kasteren, A. Noulas, G. Englebienne, and B. Krose, "Accurate activity recognition in a home setting," in UbiComp '08: Proceedings of the 10th international conference on Ubiquitous computing. New York, NY, USA: ACM, 2008, pp. 1–9.

[3] L. Bao and S. S. Intille, "Activity Recognition from User-Annotated Acceleration Data," PERVASIVE 2004, Berlin Heidelberg, 2004.

[4] J. B. Bussmann, W. L. Martens, J. H. Tulen, F. C. Schasfoort, H. J. van den Berg-Emons, and H. J. S. Behavior, "Measuring Daily Behaviour using ambulatory accelerometry: The Activity Monitor," Behavior Research Methods, Instruments & Computers, vol. 33, pp. 349-356, 2001.

[5] F. Foerster, M. Smeja, and J. Fahrenberg, "Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring," Computers in Human Behavior, vol. 15, 1999.

[6] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity Recognition from Accelerometer Data," 12th National Conference on Artificial Intelligence, 2005.

[7] M. J. Mathie, B. G. Celler, N. H. Lovell, and A. C. F. Coster, "Classification of basic daily movements using a triaxial accelerometer," Medical and Biological Engineering and Computing, vol. 42, pp. 670-687, 2004.

[8] K. Kiani, C. J. Snijders, and E. S. Gelsema, "Computerized analysis of daily life motor activity for ambulatory monitoring," Technology and Health Care, vol. 5, pp. 307-318, 1997.

[9] M. Sung and A. Pentland, "Minimally-Invasive Physiological Sensing for Human-Aware Interfaces," HCI International, 2005.

[10] Douglas Reynolds, "Gaussian Mixture Models", MIT Lincoln Laboratory, [Online]. Available: http://www.ll.mit.edu/mission/communications/ist/publications/0802_Reynolds_Biometrics-GMM.pdf

[11] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1):119-139, August 1997.

[12] Robert E. Schapire, "A Brief Introduction to Boosting", AT&T Labs, Shannon Laboratory, Florham Park, NJ [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.4706&rep=rep1&type=pdf

[13] L. Bao and S. Intille. Activity Recognition from User-Annotated Acceleration Data. Proc. Pervasive, 1-17, Vienna, Austria, 2004.

[14] Marco Ramoni, Paola Sebastiani, Paul R. Cohen "Multivariate Clustering by Dynamics", AAAI-00 Proceedings, 2000, Available: http://www.aaai.org/Papers/AAAI/2000/AAAI00-097.pdf

[15] Stephen J Preece, John Y Goulermas "Activity identification using body-mounted sensors—a review of classification techniques", Physiol. Meas, 2009. [Online]. Available: http://iopscience.iop.org/0967-3334/30/4/R01

[16] Chen J, Kwong K, Chang D, Luk J and Bajcsy R 2005 Wearable sensors for reliable fall detection 27th Annual Conf. of the IEEE Engineering in Medicine and Biology Society (Shanghai) pp 3551–4

[17] Bourke A K, O'Brien J V and Lyons G M 2007 Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm Gait Posture 26 194–9

[18] Mathie M J, Coster A C, Lovell N H and Celler B G 2003 Detection of daily physical activities using a triaxial accelerometer Med. Biol. Eng. Comput. 41 296–301

[19] Ermes M, Parkka J, Mantyjarvi J and Korhonen I 2008 Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions IEEE Trans. Inf. Tech. Biomed. 12 20–6

[20] Bao L and Intille S S 2004 Activity recognition from user-annotated acceleration data Pervasive Computing (Lecture Notes in Computer Science vol 3001) (Berlin: Springer) pp 1–17

[21] Ermes M, Parkka J, Mantyjarvi J and Korhonen I 2008 Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions IEEE Trans. Inf. Tech. Biomed. 12 20–6

[22] Parkka J, Ermes M, Korpipaa P, Mantyjarvi J, Peltola J and Korhonen I 2006 Activity classification using realistic data from wearable sensors IEEE Trans. Inf. Technol. Biomed. 10 119–28

[23] Ravi N, Dandekar N, Mysore P and Littman M 2005 Activity recognition from accelerometer data, Proceedings of the 7th Innovative Applications of Artificial Intelligence Conference (CA) pp 11–8

[24] Cristianini N and Shawe-Taylor J 2000 An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods (Cambridge: Cambridge University Press)

[25] Krause A, Ihmig M, Rankin E, Leong D, Gupta S, Siewiorek D, Smailagic A, Deisher M and Sengupta U 2005 Trading off prediction accuracy and power consumption for context-aware wearable computing Proc. of the 9th IEEE International Symposium on Wearable Computers pp 20–6

[26] Huynh T and Schiele B 2006a Towards less supervision in activity recognition from wearable sensors Proc. of the 10th IEEE International Symposium on Wearable Computers (Montreaux) pp 3–10

[27] Doukas C and Maglogiannis I 2008 Advanced patient or elder fall detection based on movement and sound data IEEE Pervasive Health Conference and Workshops 2008 (Tampere, Finland)

[28] Zhang T, Wang J, Liu P and Hou J 2006a Fall detection by embedding an accelerometer in cellphone and using KFD algorithm Int. J. Comput. Sci. Netw. Secur. 6 277–84

[29] Zhang T, Wang J, Xu L and Liu P 2006b Fall detection by wearable sensor and one-class SVM algorithm Intell. Comput. Signal Process. Pattern Recognit. 345 858–63

[30] Bishop C M 1999 Neural Networks for Pattern Recognition (Oxford: Oxford University Press)

[31] Haykin S 1999 Neural Networks; a Comprehensive Foundation (Englewood Cliffs, NJ: Prentice-Hall)

[32] Zhang K, Sun M, Lester D K, Pi-Sunyer F X, Boozer C N and Longman R W 2005 Assessment of human locomotion by using an insole measurement system and artificial neural networks J. Biomech. 38 2276–87

[33] Baek J, Lee G, Park W and Yun B J 2004 Accelerometer signal processing for user activity detection Knowledge-Based Intelligent Information and Engineering Systems, Pt 3, Proceedings pp 610–7

[34] Mantyjarvi J, Himberg J and Seppanen T 2001 Recognizing human motion with multiple acceleration sensors IEEE International Conference on Systems, Man, and Cybernetics (Tucson) pp 747–52

[35] Specht D F 1990 Probabilisitic neural networks Neural Netw. 3 109–18

[36] Gerstner W and Kistler W 2002 Spiking Neuron Models (Cambridge: Cambridge University Press)

[37] Maass W and Bishop C M 2001 Pulsed Neural Networks (Cambridge, MA: MIT Press)

[38] van Laerhoven K and Gellersen H W 2004 Spine versus porcupine: a study in distributed wearable activity recognition Proc. 8th Int. Symposium on Wearable Computers (Arlington) pp 142–50

[39] Allen F R, Ambikairajah E, Lovell N H and Celler B G 2006 An Adapted Gaussian Mixture Model Approach to Accelerometry-Based Movement Classification Using Time-Domain Features 28 IEEE EMBS Annual International Conference

[40] Lee S and Mase K 2002 Activity and location recognition using wearable sensors IEEE Perv. Comp. 1 25–32

[41] Salarian A, Russmann H, Vingerhoets F J G, Burkhard P R and Aminian K 2007 Ambulatory monitoring of physical activities in patients with Parkinson's disease IEEE Trans. Biomed. Eng. 54 2296–9

[42] Boissy P, Choquette S, Hamel M and Noury N 2007 User-based motion sensing and fuzzy logic for automated fall detection in older adults Telemed J. E. Health 13 683–93

[43] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," Pervasive Computing, pp. 158–175, 2004.

[44] O. Brdiczka, P. Reignier, and J. L. Crowley, "Detecting individual activities from video in a smart home," in Proceedings of 11th International Conference on Knowledge-Based & Intelligent Information & Engineering Systems (KES), ser. LNCS, vol. 4692. Springer, 2007, pp. 363–370.

[45] T. van Kasteren and B. Krose, "Bayesian activity recognition in residence for elders," in 3rd IET International Conference on Intelligent Environments, 2007. IE 07, Sept. 2007, pp. 209–212.

[46] U. Maurer, A. Smailagic, D. Siewiorek, and M. Deisher, "Activity recognition and monitoring using multiple sensors on different body

positions," in International Workshop on Wearable and Implantable Body Sensor Networks, 2006. BSN 2006, April 2006, pp. 4–116.
[47] P Rashidi, Diane J. Cook, Lawrence B. Holder, Discovering Activities to Recognize and Track in a Smart Environment
[48] Jonathan Lester, Tanzeem Choudhury, Nicky Kern, Gaetano Borriello and Blake Hannaford, A Hybrid Discriminative/Generative Approach for Modeling Human Activities, 2005,
[49] Nikolaos Gkalelis, Anastasios Tefas and Ioannis Pitas, Human movement recognition using fuzzy clustering and discriminant analysis, 2008
[50] Jonathan Lester, Tanzeem Choudhury , and Gaetano Borriello, A Practical Approach to Recognizing Physical Activities, Pervasive 2006, pp. 1 – 16
[51] R.E. Schapire. The boosting approach to machine learning: an overview. MSRI Work-shop on Nonlinear Estimation and Classification, 2002.
[52] MacQueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1. (1967) 281–297.
[53] "Unsupervised K-means Feature Learning for Gesture Recognition with Conductive Fur" Available: http://www.cs.ubc.ca/~nando/540b-2011/projects/19.pdf
[54] J. Wang, X. Wu, and C. Zhang, "Support vector machines based on k-means clustering for real-time business intelligence systems," Int. J. Business Intell. Data Mining, vol. 1, no. 1, pp. 54–64, 2005.