www.fit4work-aal.eu

FIT4WORK

SELF-MANAGEMENT OF PHYSICAL AND MENTAL FITNESS OF OLDER WORKERS

CO-FUNDED BY

AAL PROGRAMME

The National Centre for Research and Development

GOBIERNO DE ESPAÑA MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

ZonMw

uefiscdi EXECUTIVE AGENCY FOR HIGHER EDUCATION, RESEARCH, DEVELOPMENT AND INNOVATION FUNDING
INNOVATION AND CREATIVITY

REPUBLIC OF SLOVENIA
MINISTRY OF HIGHER EDUCATION, SCIENCE AND TECHNOLOGY

PARTNERS

PSNC

AWF The Eugeniusz Piasecki Poznan University of Physical Education

"Jožef Stefan" Institute

UNIE KBO

SGS

Teamnet transforming technology

OSM

Data Analysis

Ambient Assisted Living Joint Programme project no. AAL-2013-6-060

Deliverable 3.2.1, version 1.0

Deliverable 3.2.2, version 1.1

Lead author: Mitja Luštrek, Jožef Stefan Institute

Co-authors: Božidara Cvetković, Jožef Stefan Insitute
Martin Gjoreski, Jožef Stefan Insitute
Vito Janko, Jožef Stefan Insitute

Published on 29th of April, 2016 (deliverable D3.2.1), on 27th of March, 2017 (deliverable D3.2.2)

Current revision published on 29th of December, 2017

The Fit4Work project is co-financed though the AAL Joint Programme by:

- European Commission
- National Centre for Research and Development, Poland
- Ministry of Industry, Energy and Tourism, Spain
- Executive Agency for Higher Education, Research Development and Innovation Funding, Romania
- Ministry of Higher Education, Science and Technology, Slovenia
- The Netherlands Organisation for Health Research and Development (ZonMW), The Netherlands

Table of contents

1. Introduction

Fit4Work aims at monitoring and improving heathy lifestyle of the elderly workers. To do that the developed system monitors the users' physical activity, mental stress and the ambient conditions at the workplace. The system recommends additional physical workout in terms of daily activities or functional fitness exercises if the user was static during the day and the stress relief exercises in case stress was detected. The ambient condition monitoring results in recommendation on how to improve the ambient parameters. Figure 1.1 presents that concept schematically.

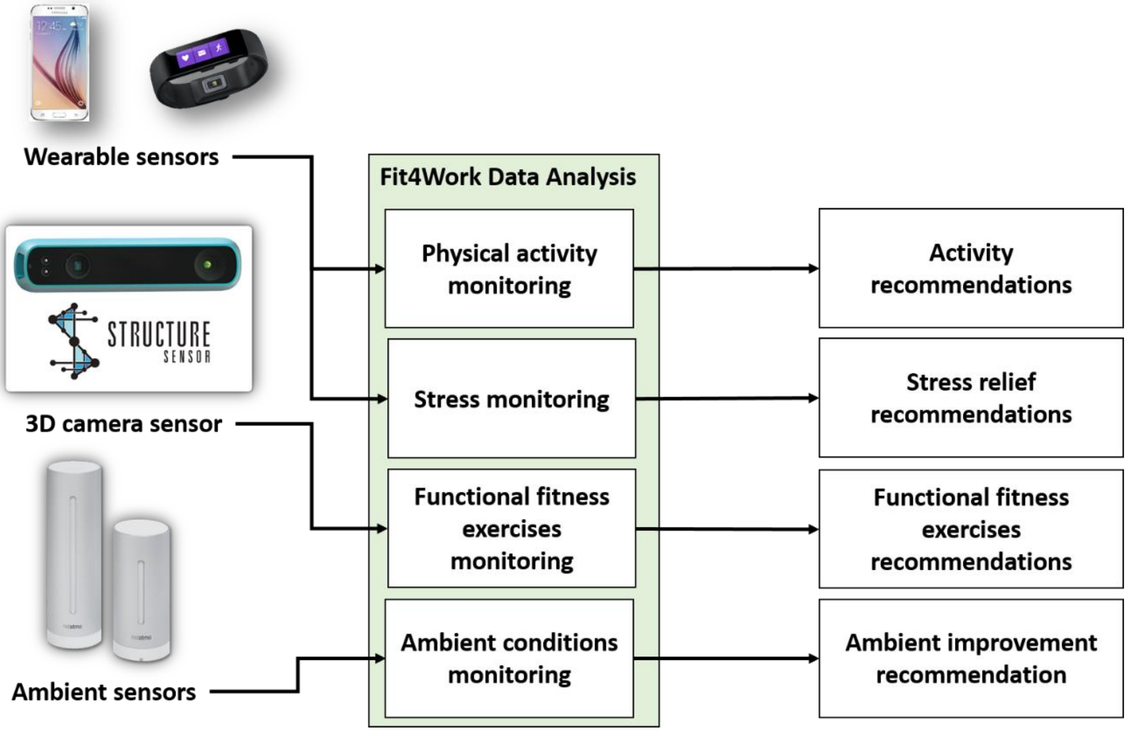
Wearable sensors
3D camera sensor
STRUCTURE SENSOR
Ambient sensors
Fit4Work Data Analysis
Physical activity monitoring
Stress monitoring
Functional fitness exercises monitoring
Ambient conditions monitoring
Activity recommendations
Stress relief recommendations
Functional fitness exercises recommendations
Ambient improvement recommendation

Figure 1.1. Input and output of the Fit4Work data analysis module.

The current document contains contents related to the initial phase of the prototype development and to the final version of the component as developed within the project. The initial version of the data analysis prototype (i.e. deliverable D3.2.1) was extended and updated to increase efficiency of data processing and provide more accurate results, thus forming the current version of the modules (D3.2.2).

2. Physical Activity Monitoring

The objective of the physical activity monitoring data analysis is to recognize the users' activities and to estimate the expended energy while active. These tasks are most often tackled with use of consumer devices in form of wristband or armband for tracking fitness activities, which correlate the amplitude of measured acceleration and measured heart rate with the expended energy. More advanced devices also include additional sensors for measuring skin and near-body temperatures, galvanic skin response, etc. These devices are either very expensive or somewhat less accurate and have limited number of activities which can be recognized (walking, running and rest).

We have seen from our previous research that activity monitoring can be accurately performed utilising only acceleration sensor embedded in an average smartphone, which most people already have and an optional heart-rate monitor [1]. Wearing a heart-rate monitor on the chest is often impractical, so in Fit4Work we decided to replace it with a simple and more comfortable wristband with embedded accelerometer and several bio sensors, one of which is the heart-rate. One of the essential goals is also to give the user freedom to carry the smartphone wherever they want (trousers pocket, torso pocket and bag) and to wear it whenever they want. This also applies to the wristband. Therefore, the method should be able to detect the presence of the devices, location and orientation of the device (in case of the smartphone) and adapt the activity monitoring accordingly.

2.1. Method

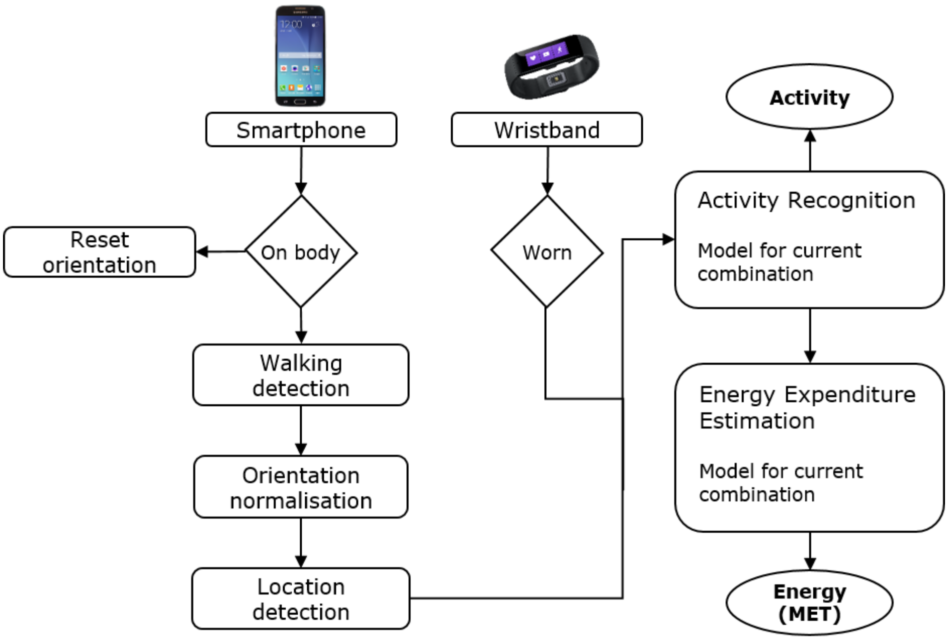

Figure 2.1 Activity monitoring method

The Fit4Work physical activity monitoring method is composed of six tasks and its workflow is presented in Figure 2.1. The devices we use are the accelerometer equipped smartphone and the Microsoft Band 2 [2] wristband capable of measuring acceleration, heart rate, barometer, galvanic skin response, ambient light and estimates the skin temperature and RR interval. First, we check whether each of the two devices is present on the patient's body. Second, if the smartphone is on the body, we wait for a walking period. Walking is detected in a location- and orientation-independent manner. It is a prerequisite for recognizing the location, and for recognizing and normalizing the orientation of the smartphone. Third, when a 10-second period of walking is detected, we normalize the orientation and detect the location of the smartphone. If the smartphone ceases to be on the body because the patient has taken it out of the pocket or bag, the information on the orientation and location is no longer valid and we have to wait until the next period of walking. Fourth, depending on the devices which are on the body and the location in which the smartphone is, we invoke the appropriate classification model for recognition of patient's elementary activity. Finally, depending on the patient's elementary activity, the devices which are on the body and the location in which the smartphone is, we invoke an appropriate regression model for estimating patient's energy expenditure.

2.1.1. Device presence detection

The detection of the presence of the devices is done by using simple heuristics.

Smartphone is considered present on the body unless one of the following is true:

- The screen is on – in this case it is probably in use and its relation to the user's body is unpredictable.
- There is an ongoing call – this is excluded for simplicity, since it is true only infrequently.
- The smartphone is still – when the smartphone is on the body, the accelerometer in it registers small movements even if the user is not moving. This is measured with the number of peaks in the acceleration signal that exceeds a threshold (feature *ACPeakCount* described in Section 2.1.3), and should be zero for the smartphone to be considered still.

The Microsoft Band 2 wristband can self-report whether it is being worn with the "Band Contact" signal.

2.1.2. Orientation normalization

The normalization of the orientation is based on the assumption that the average acceleration during walking corresponds to the Earth's gravity. The first step in the normalization is thus the detection of walking, which is described in the next subsection. The orientation is then normalized using the quaternion rotation transformation. Let $\overrightarrow{a_{walk}} = (x_{walk}, y_{walk}, z_{walk})$ be the acceleration vector consisting of the average accelerations along the three accelerometer axes during 10 seconds of walking. Furthermore, let $\overrightarrow{a_{preferred}} = (0{,}9.81{,}0)$ be the preferred acceleration vector, which would be obtained if the smartphone's longest side was perfectly aligned with the gravity. The rotation between these two vectors is represented with the quaternion matrix R, such that $\overrightarrow{a_{preferred}} = R * \overrightarrow{a_{walk}}$. To compute R, we adopted the approach by Tundo et al. [3]. The procedure is as follows. Firstly, we compute the axis-angle pair $\overrightarrow{a_{pair}}$ as the cross product between the preferred vector and the walking vector, i.e. $\overrightarrow{a_{pair}} = \overrightarrow{a_{preferred}} \times \overrightarrow{a_{walk}}$. Second, we normalize $\overrightarrow{a_{pair}}$ by dividing it by its magnitude giving us $\overrightarrow{a_{pair\_norm}}$, which is needed for the quaternion construction. Third, we use the dot product to compute the angle α between the vectors $\overrightarrow{a_{preferred}}$ and $\overrightarrow{a_{walk}}$:

$$\alpha = arccos\left(\frac{\overrightarrow{a_{preffered}} \cdot \overrightarrow{a_{walk}}}{\left\|\overrightarrow{a_{preffered}}\right\|\left\|\overrightarrow{a_{walk}}\right\|}\right)$$

Finally, the quaternions and the matrix R are calculated as follows:

$$q_0 = \cos\left(\frac{\alpha}{2}\right)$$

$$q_1 = \sin\left(\frac{\alpha}{2}\right)\overrightarrow{a_{pair\_norm}}.x$$

$$q_2 = \sin\left(\frac{\alpha}{2}\right)\overrightarrow{a_{pair\_norm}}.y$$

$$q_3 = \sin\left(\frac{\alpha}{2}\right)\overrightarrow{a_{pair\_norm}}.z$$

$$R(q_0, q_1, q_2, q_3) = R = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_0q_2+q_1q_3) \\ 2(q_1q_2+q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_0q_1+q_2q_3) & 1-2(q_1^2+q_2^2) \end{bmatrix}$$

Once the matrix R is computed, each sensed acceleration vector ($\overrightarrow{a_{original}}$) can be normalized to the preferred orientation in real time, thus creating the normalized acceleration vector $\overrightarrow{a_{normalized}}$ using the formula:

$$\overrightarrow{a_{normalized}} = R * \overrightarrow{a_{original}}$$

2.1.3. Detection, recognition and estimation

In this section we address four tasks – walking detection, smartphone location detection, elementary activity recognition and the energy expenditure estimation. They are done in essentially the same way using machine learning. First, the stream of acceleration data is segmented into windows. Two-second windows were chosen experimentally for walking detection, smartphone location detection and elementary activity recognition, as a compromise between the accuracy on longer activities (which is better with longer windows) and the ability to recognize short activities. Ten-second windows are required for energy-expenditure estimation, since this requires more information on the dynamics of movement. Then, a number of features are computed for each window, forming a feature vector. The feature vector computed from training data are fed into a machine-learning algorithm which outputs a task-specific predictor, which may be a classification or a regression model. Classification was used for walking detection, smartphone location detection and elementary activity recognition, since the output of the model is a discrete class. Regression was used for energy-expenditure estimation, since the output is a real number. When new data is obtained, the same features are computed and fed into the classification/regression model, which outputs the activity, location or estimated energy expenditure. The Support Vector Machines algorithm as implemented in the Weka machine-learning suite [4] was used to train all of the classification models, since it outperformed C4.5 decision trees and Naïve Bayes in our experiments and its size is suitable to be used on a smartphone (see Section 2.2.2). The regression models were trained using the Support Vector Regression algorithm, since it outperformed linear regression, multi-layer perceptron and REPTree on the energy-expenditure estimation task (see Section 2.2.4).

We calculated 90 features from the acceleration data from each device, three features from wristbands heart-rate monitor and three features from wristbands galvanic skin response sensor.

The features characterize the intensity and variation of the acceleration, distribution of the acceleration across each window, correlations between the axes of a single accelerometer and both accelerometers, the orientation of the accelerometers etc. The features characterizing the accelerometer orientation are low-pass filtered to attenuate the signal component related to the movement of the user, while the features characterizing the movement of the user are band-pass filtered to eliminate the gravity on one side and high-frequency noise on the other side.

Since computing a large number of features can be computationally too expensive for the smartphone, and irrelevant features may decrease the classification accuracy, feature selection was performed for each classification model. A wrapper approach was used: the features were ranked by gain ratio as implemented in the Weka machine-learning suite, they were added to a subset starting with the most informative one, a classification or regression model was cross-validated using each feature subset, and the subset with the highest classification accuracy or lowest mean absolute error was selected as the final one for the model.

After feature selection, the following features remained. Low-pass filtered features have the prefix DC, band-pass filtered prefix AC and non-filtered prefix NF.

- *AvgHeartRate* – The average heart rate within the window ($\overline{hr}$).
- *MaxHeartRate* – The average heart rate within the window (hr_{max}).
- *AvgGalvanicSkinRespose* – The average heart rate within the window ($\overline{gsr}$).
- *Activity* – The most prevalent activity within the window (*act*).
- *ACAbsoluteArea, NFAbsoluteArea* – The area under the absolute acceleration along the x, y, and z axes (A_x, A_y, A_z). Let *T* represent the length of the window and x_t represent the acceleration along the x axis at time *t*.

$$A_x = \int_{t=0}^{T} |x_t| \, dt$$

  A_y and A_z are computed similarly, using y_t and z_t in the formula instead of x_t.

- *ACTotalArea, DCTotalArea* – The sum of areas under the absolute acceleration along the x, y, and z axes: $A = A_x + A_y + A_z$.

- *ACMean, DCMean* – The mean value of the acceleration along the x, y, and z axes ($\bar{x}_{mean}$, $\bar{y}_{mean}$, $\bar{z}_{mean}$; only the equation for the x axis is shown).

$$\bar{x}_{mean} = \frac{1}{N} \cdot \sum_{i=1}^{N} x_i$$

- *ACArea, DCArea* – The sum of the acceleration along the x, y, and z axes ($\bar{x}$, $\bar{y}$, $\bar{z}$; only the equation for the x axis is shown).

$$\bar{x} = \sum_{i=1}^{N} x_i$$

- *ACAbsMean* – The absolute mean value of the acceleration along the x, y, and z axes ($\bar{x}$, $\bar{y}$, $\bar{z}$; only the equation for the x axis is shown).

$$a_{ix} = \sqrt[2]{x_i^2}$$

$$\bar{a}_x = \frac{1}{N} \cdot \sum_{i=1}^{N} a_{ix}$$

- *ACAveargeVectorLength* – The average length of the acceleration vector ($\bar{a}$).
- *ACAveargeVectorLengthPow* – The average of the squared length of the acceleration vector ($\bar{a}_{sqr}$).

$$\bar{a}_{sqr} = \frac{1}{N} \cdot \sum_{i=1}^{N} {a_i}^2$$

- *ACTotalMagnitude* – Sum of area under the length of the acceleration vector (A_a), computed as the integral of the acceleration within the window of all used device.

$$A_a = \int_{t=0}^{T} a_t \, dt$$

- *ACAreaSum* – The sum of the areas under the absolute acceleration along the x, y, and z axes (*A*):

$$A = A_x + A_y + A_z$$

- *ACAreaSumSqr* – The squared area under the absolute acceleration along the x, y, and z axes (A_x^2, A_y^2, A_z^2), and the squared area under the length of the acceleration vector (A_a^2).

- *ACPeakCount* – The number of times the movement of the length of the acceleration vector changes direction, i.e., stops increasing and starts decreasing or vice versa (*cn*).

- *ACPeakSum* – The sum of the values at which the changes (peaks) occur.

- *ACVariance* – The variance of the acceleration along x, y and z axes and the variance of the length of the acceleration vector (${\delta_x}^2$, ${\delta_y}^2$, ${\delta_z}^2$, ${\delta_a}^2$; only the equation for the x axis is shown, the other three are the same except that *y, z* and *a* are substituted for *x*):

$${\delta_x}^2 = \frac{\sum_{i=1}^{N}(x_i - \overline{x})^2}{N}$$

- *ACTotalKineticEnergy* – The integral of the change in the kinetic energy due to translation along the x, y and z axes over the window (Wk_x, Wk_y, Wk_z) In the following equation *m* represents the mass of the observed person.

$$Wk_{total\_x} = \int_{t=0}^{T} \frac{m(x_t dt)^2}{2}$$

$$Wk_{total} = Wk_{total\_x} + Wk_{total\_y} + Wk_{total\_z}$$

- *ACMeanKineticEnergySignal* – The integral of the change in the kinetic energy due to translation along the x, y and z axes over the window (Wk_x, Wk_y, Wk_z) In the following equation *m* represents the mass of the observed person.

$$Wk_x = \frac{1}{n}\int_{t=0}^{T} \frac{m(x_t dt)^2}{2}$$

- *ACMeanKineticEnergy* – The sum of the integrals of the changes in kinetic energy over the window (*Wk*):

$$Wk = Wk_x + Wk_y + Wk_z$$

- *ACVelocity* – The change in the velocity of the accelerometer along the x, y and z axes over the window (v_x, v_y, v_z):

$$v_x = \int_{t=0}^{T} x_t dt$$

- *ACSkewness* – The asymmetry of the probability distribution of a signal values about its mean $\overline{x}$, $\overline{y}$, $\overline{z}$ (s_x, s_y, s_z):

$$s_x = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i -)^3}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2\right)^{3/2}}$$

- *DCPostureDistance* – The mean distances between x, y and z axes (*distance*$_{xy}$, *distance*$_{xy}$, *distance*$_{yz}$; only the distance between x and y is shown):

$$distance_{xy} = \overline{x} - \overline{y}$$

- *ACCoefficientVar* – The index of dispersion along the x, y and z axes and the index of dispersion of the length of the acceleration vector (d_x, d_y, d_z, d_a):

$$d_x = \frac{{\delta_x}^2}{\overline{x}}$$

- *ACCorrelation* – The correlation between all pairs of axes of the accelerometer (r_{xy}, r_{xz}, r_{yz}), computed as the Pearson product-moment correlation coefficient:

$$r_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

- *ACAmplitude* – The difference between the maximum and the minimum acceleration along the x, y and z axes and the difference between the maximum and the minimum length of the acceleration vector ($diff_x$, $diff_y$, $diff_z$, $diff_a$):

$$diff_x = M_x - m_x$$

- *ACQuartileSignal* – The three points that divide the acceleration data along the x, y and z axes within the window into four equal groups, each group comprising a quarter of the data. The first quartile $Q1$ is defined as the middle number between the smallest number and the median of the data set. The second quartile $Q2$ is the median of the data. The third quartile $Q3$ is the middle value between the median and the highest value of the data set. ($Q1_x$, $Q2_x$, $Q3_x$, $Q1_y$, $Q2_y$, $Q3_y$, $Q1_z$, $Q2_z$, $Q3_z$)
- *ACQuartileMagnitude* – Quartile values of the length of the acceleration vector ($Q1_a$, $Q2_a$, $Q3_a$).
- *ACInnerQuartileRange* – The difference between the first and third quartile values along the x, y and z axes ($Qdiff_x$, $Qdiff_y$, $Qdiff_z$).
- *ACInnerQuartileRangeMagnitude* – The difference between the first and third quartile values of the length of the acceleration vector ($Qdiff_a$).
- *ACMeanCrossRate* – The number of times the acceleration along the x, y and z axis crosses the mean acceleration value ($cross_x$, $cross_y$, $cross_z$).

Walking detection classifies the user's activity into walking or anything else. Single classification model was trained, which is used in case the smartphone is perceived as a present device. Since the orientation was not normalized yet at this step, we used orientation-independent features, five of which were left after feature selection.

Location detection classifies the smartphone's location into the breast pocket, trousers pocket or bag. When using only the smartphone, the classification model started with 90 acceleration features, 16 of which were left after feature selection.

Elementary-activity recognition classifies the users's activity into lying, sitting, standing, walking, running, cycling or transition. If the smartphone is in the bag, it classifies it into walking, running, cycling or other, since a bag/backpack is not often carried during such static activities, and if it is, they cannot be distinguished anyway. Classification models were trained for the smartphone alone in each of the locations and for the wristband alone. The classification models started with all 90 (smartphone or wristband alone) and ended with 8–48 of them after feature selection.

Energy-expenditure estimation estimates the user's energy expenditure in MET (Metabolic Equivalent of Task, 1 MET corresponds to the energy expended at rest) using regression models. We trained four regression models, one for each location of the smartphone alone and one regression model for the wristband alone. The regression models started with 90 features (smartphone or wristband alone) ended with 11–22 features after feature selection.

2.2. Evaluation

2.2.1. Dataset

The dataset contains data of 10 elderly healthy volunteers. Each volunteer was wearing four smartphones at four locations (trousers pocket, jacket pocket, bag and torso pocket), and two heart-rate equipped wristbands (Microsoft Band 2 and Empatica) one on each wrist. The volunteers were also equipped with the indirect calorimeter Oxycon Mobile to measure the reference energy expenditure, and BodyMedia SenseWear armband. SenseWear is a high-quality commercial device to measure energy expenditure, which was used as a benchmark. Figure 2.2 presents a volunteer during recording.

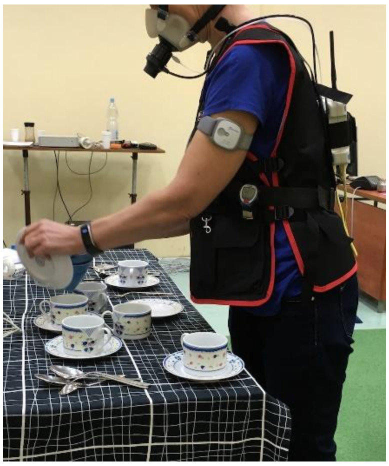

Figure 2.2. Volunteer during recordings.

Volunteers were performing predefined set of activities from the scenario presented in Deliverable 3.4 in Section 2.1. We obtained approximately 110 minutes of data from each sensor for each volunteer.

Evaluation was performed using leave-one-person-out cross-validation, which means that all classification and regression models were trained on the recordings of 9 volunteers and tested on the recording of the final one, with the procedure repeated once for each volunteer.

2.2.2. Walking detection evaluation

The walking detection evaluation was done in isolation on data from the presented dataset in which we relabelled the data of other activities then walking as other and created a binary classification problem. A comparison of different machine-learning algorithms for walking detection preferred C4.5 algorithm for final classification model. The results per device are presented in Table 2.1 after feature selection which returned 6 features to be used for wristband model and 9 features for the smartphone mdoel. We can see that the accuracy for walking detection with the wristband (W) is 90% and with the smartphone (S) is 91%. We also show the Kappa and F-score measure for the task.

Table 2.1 Evaluation of walking detection

Device	Accuracy	Kappa	F-score
W	90%	0.76	0.89
S	91%	0.83	0.91

2.2.3. Location detection evaluation

The location detection classification model was also done in isolation. It was tested with three locations trousers pocket, torso pocket and bag. The preferred machine-learning algorithm is the C4.5. The accuracy of the model after feature selection is 91% and the number of features in the model is 11.

2.2.4. Activity-recognition evaluation

Activity recognition was evaluated in seven combinations. When single device is used (wristband and smartphone in three locations) and when combination of devices is used (wristband + smartphone at three locations). A comparison of different machine-learning algorithms for activity recognition preferred Random Forest for the final models training. The accuracy results per orientation are presented in Table 2.2. Since different activities are recognised with different devices we show confusion matrices in Figure 2.3 per combination in which we see the set of the activities and misclassifications.

Table 2.2. Evaluation of the activity recognition [%].

	Single device				Combination		
	W	T	J	B	W+T	W+J	W+B
Accuracy	80	92	80	95	89	85	89
Kappa	0.75	0.89	0.70	.90	0.87	0.80	0.83
F-score	0.75	0.78	0.65	0.92	0.89	0.83	0.71
Number of activities	6	6	5	4	8	8	5

2.2.5. Energy expenditure estimation evaluation

The performance of the regression models for energy-expenditure estimation was measured using mean absolute error (MAE). Let EE_{true} represent the real expended energy as measured by the Cosmed indirect calorimeter and $EE_{predicted}$ represent the energy expenditure estimated by our regression models. The mean absolute error over a set of n examples equals:

$$MAE = \frac{1}{n}\sum\left|EE_{true} - EE_{predicted}\right|$$

The energy-expenditure estimation was tested on the presented dataset. A comparison of different machine-learning algorithms for energy-expenditure estimation preferred the Random Forest regression. The results per single device and combination are presented in Table 2.3.

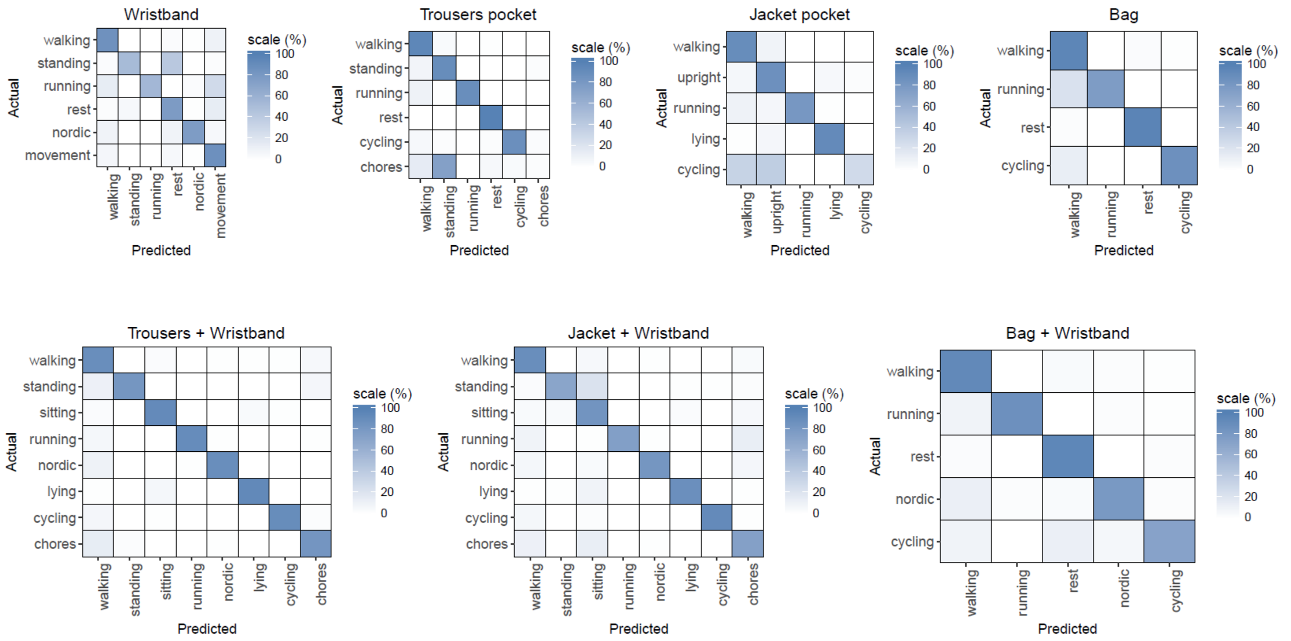

Figure 2.3. Confusion Matrices for activity recognition.

Table 2.3. The mean absolute error [MET] of the energy expenditure per combination. O_w – wristband without orientation, O_s – smartphone without orientation, W – wristband, T – trousers, J-jacket, B-bag, W+T – wristband and trousers, W+J – wristband and jacket, W + B – wristband and bag. Last column is the consumer device SenseWear dedicated for estimation of energy expenditure.

	Single device						Combination			SenseWear
	Ow	Os	W	T	J	B	W+T	W+J	W+B	
Accuracy	0.94	1.12	0.64	0.67	0.72	0.75	0.59	0.71	0.57	1.03

One can see in Table 2.3 that the lowest MAE (0.59 MET) is obtained when the user wears the wristband and the smartphone in the trousers pocket. The highest error (1.12 MET) is obtained when the orientation of the smartphone is not known, thus the location is not known. The movements that are able to occur per location confuse the energy-expenditure estimation. Since this model is used only in the beginning while the procedure is detecting the location, the error is not very crucial for the overall estimation. We also compared our method against the SenseWear device (last column in Table 8). We can see that all models except the smartphone without orientation outperform the estimation by the consumer device. The true values as measured by the calorimeter (blue) and estimated values with the wristband model for each 10 second (pink) is presented in Figure 2.4.

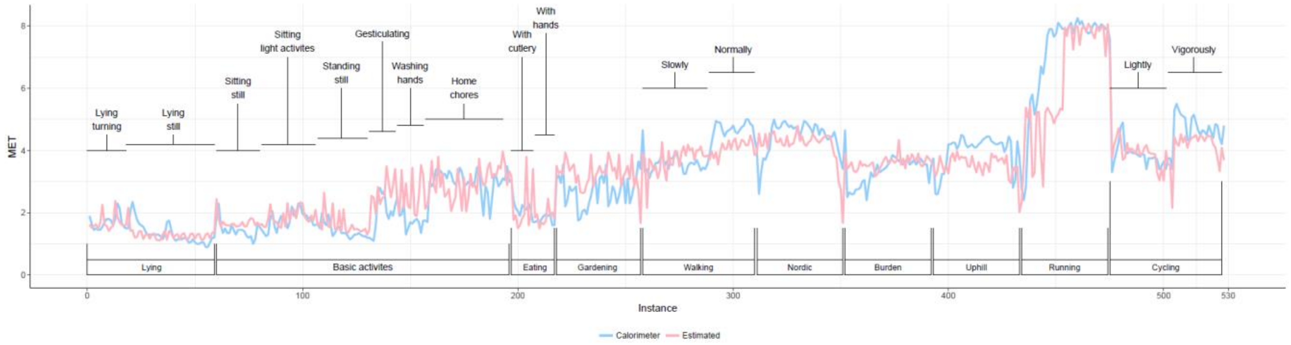

Figure 2.4 True vs predicted estimated energy expenditure when wristband is used.

3. Functional Fitness Exercises Monitoring

Functional exercises are aimed to maintain or improve the physical capabilities of fit4work users according to their daily working activities. These kinds of exercises are focused on the user' flexibility and range of movement and will be monitored using mobile mocap devices.

The most common MOCAP systems are composed by at least 4 high precision cameras and a set of markers placed on strategic parts of the body allowing the recognition of a human body and also tracking his/her limbs segments. These kinds of capture environments have high accuracy (under the millimeter) but also high complexity, are really expensive and needs to set them up properly calibrated and the markers position has a large influence on the skeleton tracking. Kinect-like devices allow computing this skeleton tracking without markers but with a lower precision.

The term "Kinect-like devices" refer to those digital CMOS sensor cameras, able to register the depth distance of the pixels in the resulting captured image within the field of view (FOV) of the camera. A light source emits a dot matrix wavelength of 830nm constant amplitude, invisible to the human eye, which is filtered and dispersed by a pseudo-random pattern.

The correlation between the source pattern of light and its reflection recorded with the camera provides the distance of the reflecting objects in the scene. Figure 3.1 shows the two images obtained from Kinect in a single shot. First is a RGB image of the scene, while the second is the colour representation of Depth values.

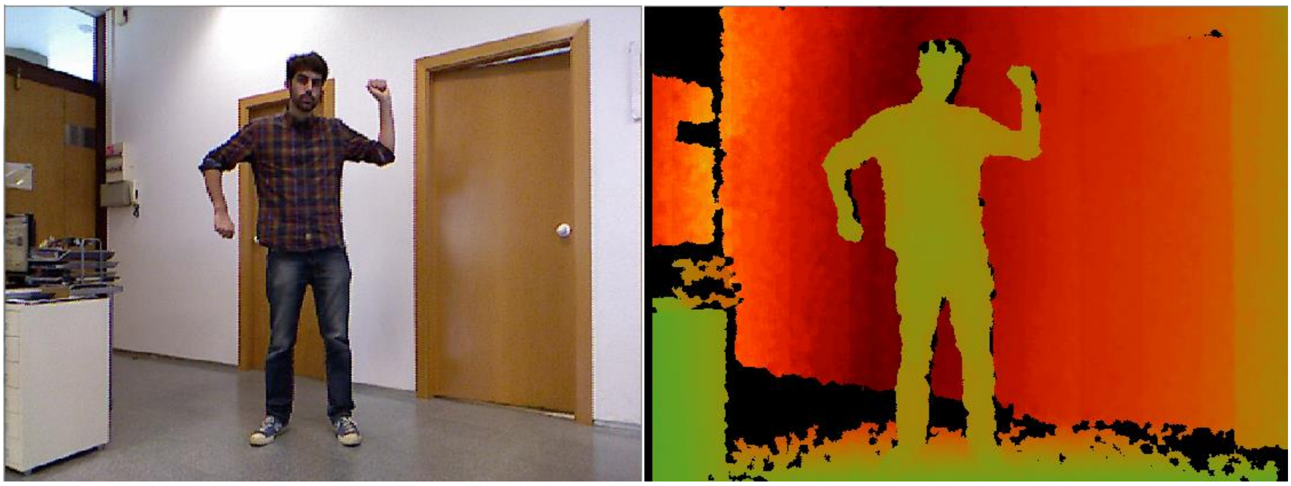

Figure 3.1 RGB image and Color representation of depth information

Adding a sense of depth to the image simplifies one of the more complex tasks in computer vision algorithms: segmentation. Having the distance of image pixels allows to the computer analyses the objects by removing the background and other elements in the scene avoiding awkward segmentation algorithms based on colors and gradients. Figure 3.2 shows a 3d representation of captured images using depth information associated to each pixel.

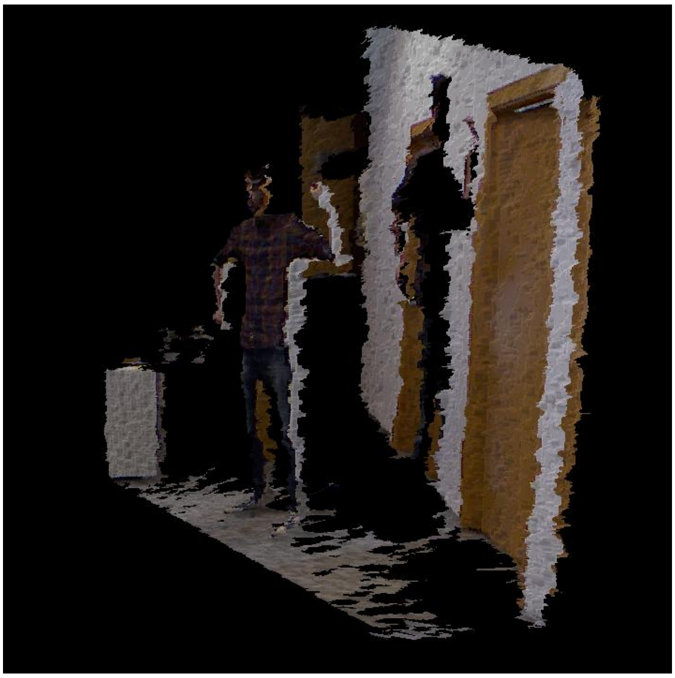

Figure 3.2 3D scene

The analysis of objects in movement according to the size and shape determines the potential users (persons) in the image. Once the user is segmented from the image, Skeleton tracking algorithm needs to be initiated by holding a defined pose. Morphological matching algorithms converge the user's shape to the anthropometric model and allow having 3D coordinates for each joint by means of OpenNI[20] libraries. The anthropometric model is commonly constructed by segments representing the different parts of the body. These segments are connected through the joints of the skeleton tracking (see Figure 3.3). OpenNI algorithms are based on joints position estimation and don't allow to compute proximal-distal axis rotations.

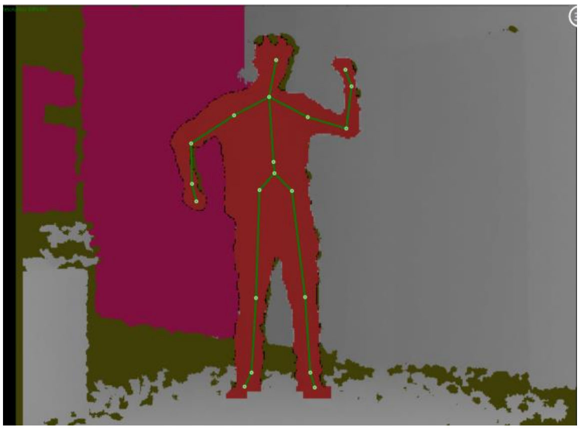

Figure 3.3 Skeleton tracking algorithm overlayed

The physical exercises are series of movements described by the position of all joints involved in the movement. Each joint is depicted in relation to its reference joint. (F.I. elbow is described in reference of shoulder joint.) The algorithms will gather and classify the joints coordinates in order to assess the exercise in performance.

Deliverable D4.3.1/D4.3.2 discusses the construction of motion capture module (i.e. the 3D sensing module) in details.

4. Stress Monitoring

The objective of the Fit4Work stress monitoring module is to continuously monitor user's stress level and to provide overview of stressful events to the user. Additionally, the output of the stress monitoring module is used by a stress-relief module for suggesting appropriate exercises at appropriate time. For the stress monitoring module we developed a machine learning method which is applied on data collected via sensor-equipped device worn on the wrist. Recently, similar approaches have been proposed for stress detection [11][12][13], but all of them are utilizing chest belts which differ from our goal for stress detection using a wrist device. The device provides raw data from bio-sensors (blood volume pulse, heart rate, R-R intervals, galvanic skin response and skin temperature) and acceleration sensors (e.g., accelerometers). This raw data is then processed using signal processing techniques in order to provide numerical features relevant for stress.

4.1. Method

Figure 4.1 presents the method used for developing the stress monitoring module. The method consists of three main machine learning components: a laboratory stress detector, an activity recognizer, and a context-based stress detector which provides the final output. The laboratory stress detector and the activity recognizer uses raw data provided by the user's wrist device, and the context-based stress detector uses the output provided by the laboratory stress detector and the activity recognizer.

The laboratory stress detector is a machine learning classifier that distinguishes stressful vs. non-stressful events in 4-minute data windows with a 2-minute overlap. For each data window, features for stress detection are computed from the raw data provide by the user's wrist device. From each physiological signal (BVP, HR ST and GSR), statistical and regression features are computed: mean, standard deviation, 2 computed with an algorithm for peak detection [14]. For the RR signal, we use features obtained through heart-rate-variability analysis in the frequency and time domain. These features are fed into a classifier trained with the Random Forest algorithm, which was chosen experimentally.

The activity recognition (AR) classifier is a ML classifier that uses the accelerometer data to recognize the user's activity: sitting, walking, running, and cycling. It is based on our previous approach for AR [15]. The classifier outputs an activity every 2 seconds. When aggregating these activities over the data window of 4 minutes, each activity is changed into an activity level (e.g., lying = 1, walking = 3, running = 5) and averaged over the window. The average activity level is passed as a feature to the context-based stress detector.

The context-based stress detector is developed to distinguish between genuine stress in real life and the many situations which induce a similar physiological arousal (e.g., exercise, eating, hot weather, etc.). As features, it uses the distribution of the last 10 outputs of the laboratory stress detector, the previous output of the context-based detector, and context features: whether there was any high-level activity in the last 20 minutes, the hour of the day, the type of the day – workday/weekend, etc. It classifies every 20 minutes as stressful or non-stressful. The context-based stress detector was trained with the SVM algorithm, which was again chosen experimentally.

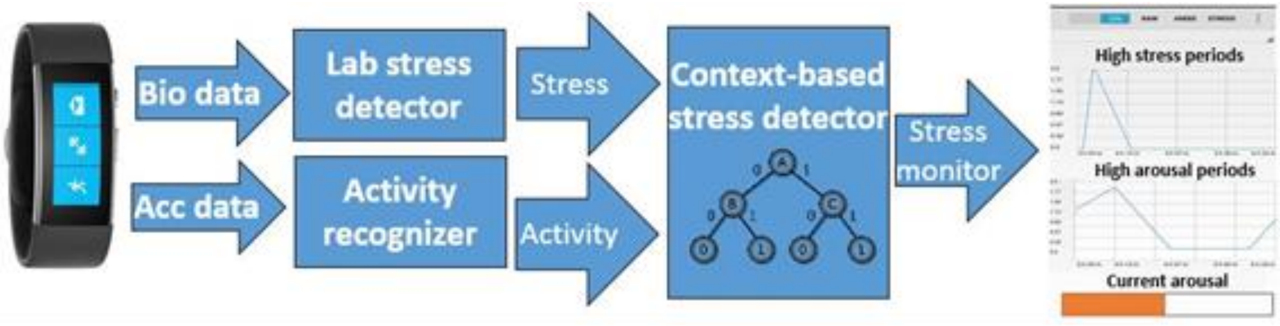

Figure 4.1.The stess monitorring module

4.2. Data

The data used for developing the stress monitoring module consist of laboratory and real-life data. For collecting the laboratory data we used standardized stress-inducing experiments. Additionally, baseline (no-stress) data was recorded on a separate day. During the experiment, there were no movement constraints, making it as close as possible to real life. As suggested by Hovsepian et al. [16], the laboratory data was used to build machine-learning models (laboratory stress detector) for distinguishing stress vs. no-stress. On the other hand, the real-life data was gathered on ordinary days where five subjects were wearing the wrist device, and were keeping track of their stressful events. This data was used for evaluation only. The following two subsections provide detailed description of the data in this study.

4.2.1. Laboratory data

For collecting the laboratory data, a web application was developed in collaboration with psychologists. The application implements a variation of the stress-inducing method presented by Dedovic et al. [16]. This method is validated with cortisol analysis in three different studies. The main stressor is solving a mental arithmetic task under time and evaluation pressure (see Figure 4.2). In short, a series of randomly generated equations were presented to subjects, who provide answers verbally. The time given per equation was dynamically changing. For each two consecutive correct answers the time was shortened by 10%, and for each two consecutive wrong answers the time was increased by 10%. By varying the time, the subjects are pushed to their limits for executing mental arithmetic tasks. Each session consisted of three series of equations with increasing difficulty: easy, medium and hard. Each series of equations lasted for five minutes. For motivation, a reward was promised to the top three participants. After each stage, the participant was shown false ranking score, positioning him/her in the top five, and this way motivating him/her to try harder in the next stage and try to win the award. The application is available on-line: http://dis.ijs.si/thestest/.

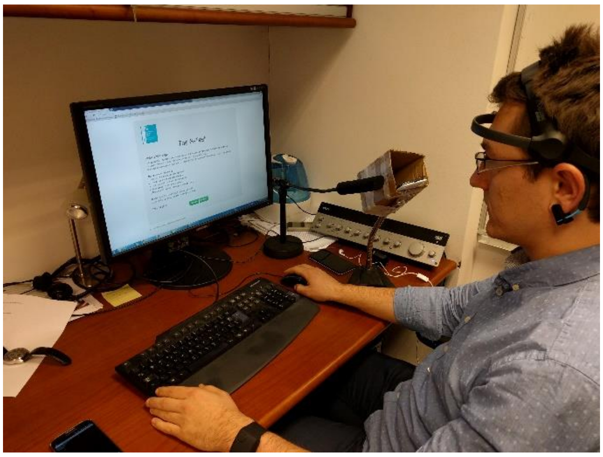

Figure 4.2. Collecting laboratory data for stress monitorring.

Four Short STAI-Y anxiety questionnaires (STAI-Y) [19] were filled by each participant: before the experiment, and after the easy, medium and hard session. Table 4.1 presents a summary of the questionnaire answers.

Table 4.1. Laboratory data for stress monitorring module – questionnaires summary

Type	Before	Easy	Medium	Hard
STAI score	10.95	13.33	14.05	13.81

The STAI questionnaire reflects the degree to which the subjects have a feeling of unease, worry, tension, and stress. Its value range is from 0 to 24, with 24 being the highest level of anxiety. The answers of the STAI questionnaire were used for subject-specific labelling of the data. For each subject, the period before answering the STAI questionnaire in which they achieved the lowest score is labelled as low stress, and for each +3 STAI points, the stress label is increased by one, thus we get no stress (baseline data), low stress (lowest STAI score), medium stress (lowest STAI score +3) and high stress (lowest STAI score +6). In the final experiments the medium and high stress were merged because only two subjects achieved a high level of stress, so we had three degrees of stress: no stress, low and high.

4.2.2. Real-life data

For the real-life data we used a combination of stress log and Ecological Momentary Assessment (EMA) prompts [17] implemented on a smartphone [18]. The subjects had to answer 4-6 EMA prompts at random periods of the day, and in the case of a stressful situation, they were logging the start, duration and level of stress on a scale from 1 to 5 (1-no stress, 2-low stress, 3 and 4 – high stress, there were no 5). The answers of the EMA prompts and the stress log were used to label the real-life data. Table 4.2 presents an overview of the data.

Table 4.2. Data overview. Duration in minutes.

Data	# participants	No Stress	Low	High
Lab	21	840	356	368
Real	5	73000	4200	2500

4.3. Evaluation

The evaluation the method was performed on the real-life data. Because labeling stress is quite subjective [13] and it is almost impossible to strictly define starts and ends of stressful situations, we used a technique that splits the stream of real-life data into discrete events. Each event had a minimum length of one hour. If there was a stressful situation in the event (labeled by the user), the event's duration was extended to capture the stressful situation plus one hour before and after the situation. By this, we are allowing for a labeling lag of one hour. The 55 days of the real-life data was split into nearly 900 events, each lasting at least an hour. Table 4.3 presents the confusion matrix for the event-based evaluation using leave-one-subject-out (LOSO) cross-validation. The accuracy achieved by the stress detecion method (for distinguishing stressful vs. non-stressful events) is 92%.

Table 4.3. Confusion matrix for leave one user out evaluation on the real-life data.

	No Stress	Stress
No Stress	790	23
Stress	51	63
Recall	97%	55%
Precision	94%	73%
F1 score	96%	63%
Accuracy	92%	

Additionally, Figure 4.3 depicts the output of the stress monitoring method for the real-life dataset with LOSO evaluation. On the x-axis is the day, on the y-axis is the hour of the day, the black stripes label to which subject belongs the data, and the colored squares correspond to the false positive (FP), false negative (FN), true positive (TP) and true negative events (TN). From the figure it can be seen that subject 1 (S1) has many FN events, and subject 2 (S3) has many FP events compared to the rest of the subjects.

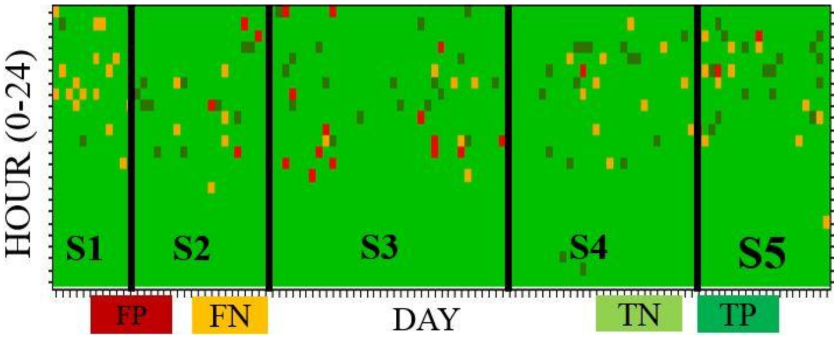

Figure 4.3. Output with LOSO evaluation.

Figure 4.4 present the intensity of stress for the real-life dataset with LOSO evaluation. Real labels are presented at top, while the predictions of the method are at the bottom. On the x-axis is the day, on the y-axis is the hour of the day, and the color corresponds to the intensity of stress.

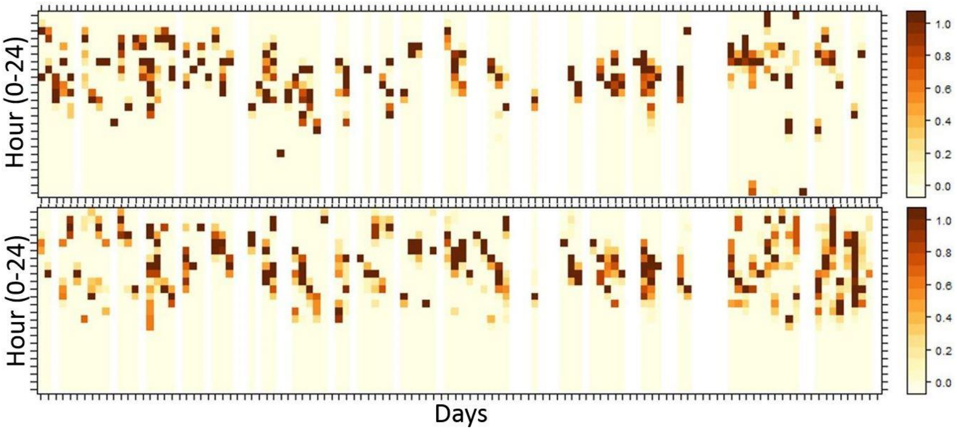

Figure 4.4. Real labels (top) and predictions of the method (bottom) with LOSO evaluation. Color presents intensity of stress.

The experimental results show that there is still room for improvement, but they are encouraging for such a challenging problem. For now, the context-based stress detector receives information from the laboratory detector and from the activity recognizer. Additional context information can be provided from other components that recognize events which induce similar physiological arousal to a stress event (e.g., exercise, eating, hot weather etc.). Because stress is quite subjective and perceived differently by different subjects, we also plan to implement personalization to allow to the general model to adapt to new users. The need for personalization was confirmed by the visualization in Figure 4.3, where it can be seen that distribution and the type of the classification errors (e.g., FP vs. FN) is subject-specific.

5. Ambient Conditions Monitoring

The objective of the Fit4Work ambient conditions monitoring module is to detect the state of the environment and recommend the appropriate actions to the user which will improve the ambient quality. The module for ambient conditions monitoring is presented in Figure 5.1. In this deliverable we focus on the sensing component. Ontology and q-rating are presented in deliverable D3.4 while the simulator and recommendations in deliverable D3.3.1/D3.3.2.

In comparison with the methodology in the previous deliverable D3.2.1, we created models with the heavier use of expert knowledge and mathematical modelling, and achieved better results both in terms of accuracy as well as comprehensibility of the prediction.

The Fit4Work ambient conditions monitoring sensing component is composed of hardware sensors and virtual sensors. The hardware sensors measure and return raw parameter values, while virtual sensors values are estimated from raw parameter values, since they cannot be sensed directly (ex. state of a device, we have no hardware sensor for).

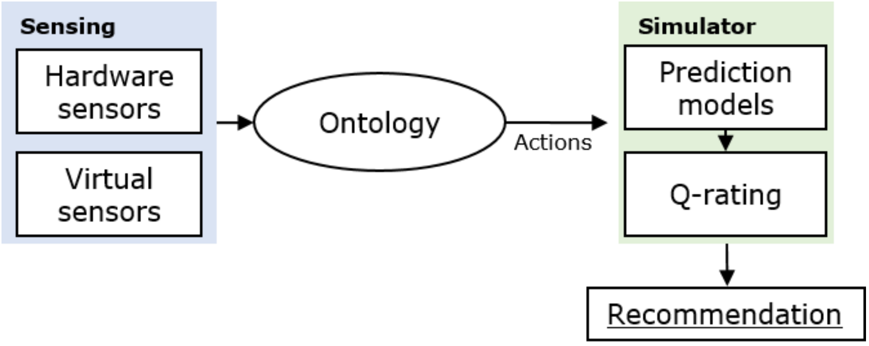

Figure 5.1. The ambient conditions monitoring and recommendation system.

5.1. Hardware sensors

There is a growing market of commercial devices with integrated environmental sensors, which can monitor environment quality [5][6][7]. In Fit4Work we decided to use the commercial weather station NetAtmo^{TM} [7] composed of an indoor and outdoor module (Figure 5.2). The indoor module measures the indoor temperature, humidity, CO_2 concentration and noise, while the outdoor module measures the outdoor temperature, humidity, and air pressure. The NetAtmo measurements occur every ten minutes, so we define ten minutes as time of one step for data acquisition and data analysis.

5.2. Virtual sensors

Virtual sensors refer to values that are not directly measured. Instead, their value is derived from the measured data and then later used to help derive some other value. In our setting, there are five virtual sensors that affect the ambient parameters: the windows state, air conditioner state, heater state, humidifier state and number of people in the office (occupancy estimation). Virtual sensors serve instead of physical sensors, which would increase the cost of the system and be more difficult to install than the used weather station. We have observed in the related work that correlation between parameters such as temperature, humidity and CO_2 values and the occupancy state or window state is high [8][10], so it can be modelled. For example, Han et al. [10] report 0.96 root mean squared error for estimating the number of occupants solely from CO_2 concentration using Hidden Markov Models, while others used additional sensors but report a similar error [9]. We did not come across any relevant methods for estimating the state of other virtual sensors.

Occupancy estimation is calculated from the raising CO_2 levels, and the humidifier state is tied to humidity data, so those two will be explored in the corresponding Sections 2.2 and 2.4. The remaining three can be determined with simple heuristics as described below.

The window state detection

Windows were modelled in a binary fashion: they are either open or they are closed. In a real office there might be many windows, some of them open, some closed, some perhaps half-open at any time; but lacking any knowledge about the window quantity or size, predicting their state more accurately is almost impossible.

An effect of opening the window is reflected on all three ambient parameters, but only in the case of CO_2 is the effect consistent. Whenever a window is opened, CO_2 falls drastically, whenever it is closed it starts to rise again. This allows us to make a simple heuristic: a.) if the CO_2 is falling faster than some threshold, window was opened; b.) if CO_2 keeps increasing, the window is closed; c.) if neither of those is happening, assume the last known state. Thresholds can be determined by looking at the data history and find such values that would generate predictions, where windows are opened/closed few times a day, as would realistically be the case.

The air conditioner state detection

Again we assume binary outcome - the air conditioner is either on or off - additionally we assume that the temperature set on it is constant, or at least is changing infrequently. The distinguishing pattern of air conditioning is one of temperature inside decreasing while the temperature outside is higher than inside. Since the temperature naturally tries to equalize itself with its surroundings and since all other factors (people, computers, etc..) only serve to warm the office, it is reasonable to conclude that such a temperature drop was caused by the air conditioner. After a while of the air conditioner working, the temperature will converge to value that can be stored for later predictions. If the temperature starts rising again, the air conditioner is assumed to be turned off.

The heater state detection

The same assumptions and methods are used here as with the air conditioner, except in reverse: the heater is on if the inside temperature rises significantly more than expected from the outside temperature, etc.

The light estimator

We have additionally developed light estimator to correct the light measurements of the used smartphone. The light estimator utilises only the stream of data from the smartphones light sensor to form a feature vector. It is composed of three regression models build on the raw light data from the smartphone light sensor to estimate the current light in lux. The estimator chooses one of the models according to the light measurement as measured by the smartphone. One model is used for the light interval between 0 and 400 lux, second model for the light interval between 401 and 900 lux and third for the light interval above 900 lux.

Figure 5.2. NetAtmo indoor and outdoor module.

5.3. CO_2 prediction

Intuitively, CO_2 level inside the office is increasing linearly with respect to the number of people present, but at the same time it tries to equalize itself with the outside CO_2 level. The bigger the difference between outside and inside, the faster it moves from one side to another. If window is opened, the same happens, only to a significantly larger degree. This can be encapsulated in the following equation.

$$C_{n+1} = C_n + \alpha(C_{out} - C_n) + \beta p$$

C_n = CO_2 inside at timestamp *n*
C_{out} = CO_2 outside
p = the number of people in the room
α = the coefficient of diffusion speed (between 0 and 1) — small for closed windows, big for open ones
β = how much a single person raises CO_2 in a given time unit

Using all the labeled data, the α and β are mostly trivial to compute using linear regression. Using them results in an almost perfect match between the predicted and real values. In Figure 5.3 we plot a scenario where we know the initial CO_2 level and all future windows states and all future numbers of people, and we are able to predict CO_2 level two days in advance. This strongly signifies that the model captures the real-life behavior of CO_2, and it is only a matter of determining the correct coefficients.

Calculating the coefficients for a given office without the labeled data, however, is a challenging task as the above formula has 5 unknowns - α when windows are closed, α when windows are opened, window state, β and number of people p. Furthermore, these coefficients can behave very similarly: CO2 level in a room with many people and open window can be close to CO_2 level in a room with closed windows and few people. The first improvement is to combine the two variables β and p into one - γ, as we never need those two individually and are only interested in their product. This shortens the formula to:

$$C_{n+1} = C_n + \alpha(C_{out} - C_n) + \gamma$$

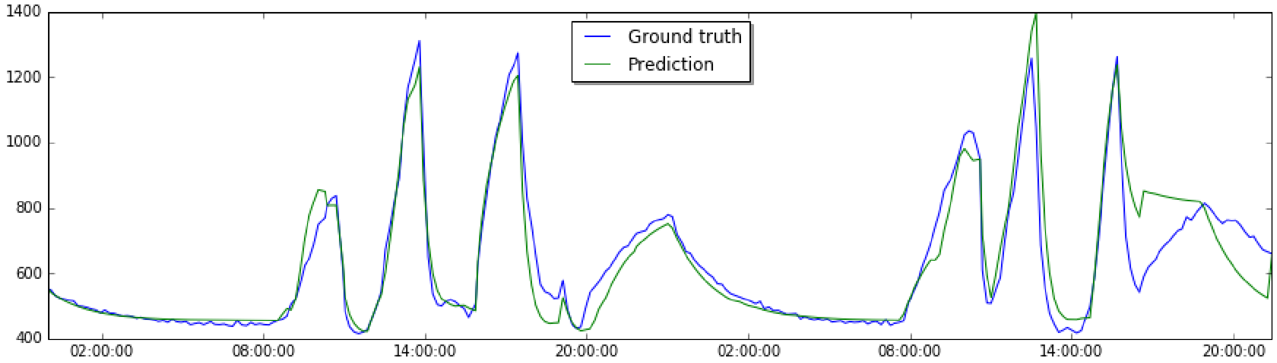

Figure 5.3. CO_2 prediction and real CO_2 values

This formula can be rewritten in an analytical way (listed below) so it can predict an arbitrary time step instead of only steps of integer size (10 minutes). A simple explanation of this formula goes as follows - CO_2 always converges to a value L. The number of people in the office dictates this limit, while the value α dictates how fast we approach this limit. The inverse of this formula will also be useful and can be trivially computed using some basic algebra.

$$C_n = L + (C_0 - L)(1 - \alpha)^n$$

$$L = C_{out} + \frac{\gamma}{\alpha}$$

Determining the window state is described in Section 5.2. If we know the α value for the current window state, the γ value becomes the only unknown in the formula and can be determined with a simple linear regression, using last three data points. Since γ - correlates with the number of people in the office, it must be recalculated for every prediction. The α value on the other hand is dependent on the office heat insulation level, office size and windows size, and is therefore a constant. We can therefore estimate the α value by trying different values on the past two weeks of data and then select the one that has the lowest error rate when predicting – this is possible since when predicting on the past data, we already know what CO_2 value will be reached.

5.4. Temperature prediction

We used the same base formula as in Section 5.3 for the inside temperature prediction. This model however, had to be made more complex because of two factors.

First, the temperature does not converge towards the outside one, but goes towards some function of the outside temperature instead. For example, even if the outside temperature is below zero, the temperature in the office never went below 10 degrees, even without heating. There are several reasons for this behavior, including the heat of the building itself, and the fact that building is warming and cooling at different rates than the exterior when the external temperature changes. This is dealt by calculating a function from last two weeks of data that models the expected inside temperature as a linear function of the outside temperature. The calculation is made during rest days, when no one is in the office, reducing the noise in the data. This calculated value then replaces the value C_{out} in the base equation.

Second, we have to account for both air conditioning and heating. The detection of their state is described in Section 5.4 In the same section it is also described how to collect the limiting temperature value these devices generate. If either device is on, the corresponding limiting value replaces L in the base equation.

5.5. Humidity prediction

Humidity values did not exhibit any obvious pattern, so we used the same methodology with machine learning as described in deliverable D3.3.1 and paper [21].

The features were extracted from the historic data of all parameters from previous 20 time steps and from the current measurements of the same features. All used features are:

- Last measured values of indoor parameters
- Last measured values of outdoor parameters
- Estimated number of occupants
- Estimated window state
- “First derivate” of each parameter, calculated through the last n time steps (n=3, 5, 20) with least square linear regression
- “Second derivate” of each parameter, calculated with the last n time steps (n=3, 5, 20) with least square linear regression, giving us the speed of dynamics of the parameter
- Number of time steps since the last window action. The dynamics of parameter changes faster right after the action was taken and then asymptotically approaches the new equilibrium value.

5.6. Evaluation of Virtual Sensors

To evaluate the virtual sensors, we collected the dataset in real office environment. The devices used for the data collection and for interaction with the office occupants is presented in Figure 5.4.

5.6.1. Datasets

Hardware and virtual sensors

We have equipped three offices, A (43 m^2), B (27 m^2), and C (20 m^2) with NetAtmo, humidifier, window state sensors and a smartphone application which was used for self-reporting the occupancy, labelling the state of the devices (e.g., humidifier is on or off) and later on for receiving the recommendations from the Fit4Work system. During the working hours (on work days between 9.00 and 17.00), the average number of occupants per office was: 2.6 ± 1.5 (max 9) in A, 2.0 ± 0.9 (max 7) in B and 1.6 ± 0.9 (max 7) in C.

We started the collection of the data on 2016-01-16 and the collection is still in progress. For each office we collected the raw data from the devices, state of the devices, recommended action and user interaction.

Light estimator dataset

The dataset for light estimator was collected in different offices with different light conditions. We have collected 250 measurements of light intensity as measured by smartphone (Samsung galaxy s6) and at the same time as measured by reference light meter.

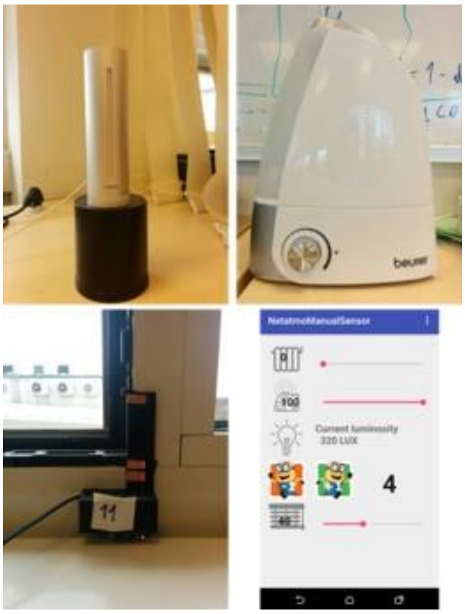

Figure 5.4. Devices used for the data collection. Indoor NetAtmo module (top left), humidifier (top right), window state sensor (bottom left), and application interface for self-reporting the occupancy, labelling the state of the used devices and receiving the receiving the recommendations (bottom right).

5.6.2. The Results

The task of the system was to predict the humidity, temperature and CO_2 values in the office for the next 30 minutes. System could only learn from past, unlabelled data from the same office, simulating a real-life deployment scenario. While the system determined the state of virtual sensors, it performance on this task was tested only indirectly, via the performance on the humidity, temperature and CO_2 values prediction. Mean absolute error (MAE) was calculated for each of the ambient parameters and results are listed in Table 5.1, where they are compared against our previous results, reported in paper [21].

Table 5.1. MAE when predicting ambient parameters, 30 minutes in advance

Parameter	Error	Previous error [21]
CO_2 [ppm]	43	79
Temperature [°C]	0.36	0.5
Humidity [%]	0.74	0.74

For light estimator we evaluated all three models separately with 10-fold-cross validation. The results are presented in Table 5.2. The correlation between the light as measured by smartphone and the reference light meter is high when the light is in normal interval (around 500 lux) and low in case of really bright environment. The results show that the error increases with the brightness of the room.

Table 5.2. Results of the evaluation of the light estimation with linear regression model.

Light estimator	MAE [lux]	RMSE [%]
From 0 lux to 400 lux	42	53
From 401 lux to 900 lux	84	110
From 901 lux and above	2052	3621

6. Discussion

In this deliverable we present the final data analysis and final methods for physical activity monitoring, functional fitness exercises, stress monitoring and ambient conditions monitoring. Methods for physical activity monitoring utilises data from smartphone and Microsoft Band 2 wristband and stress monitoring utilises bio-sensors from the same wristband. The functional fitness exercises and ambient condition monitoring utilise external devices. To recognise the execution of functional fitness exercise, we use the 3D scanner used for augmented reality. To retrieve the environment data we utilise the NetAtmo weather station.

7. Bibliography

[1] B. Cvetković, V. Janko and M. Luštrek, "Demo abstract: Activity recognition and human energy expenditure estimation with a smartphone," *Pervasive Computing and Communication, 2015 IEEE International Conference on*, St. Louis, MO, 2015, pp. 193-195. doi: 10.1109/PERCOMW.2015.7134019

[2] Microsoft Band 2, https://www.microsoft.com/microsoft-band

[3] M. D. Tundo, E. Lemaire and N. Baddour, "Correcting Smartphone orientation for accelerometer-based analysis," Medical Measurements and Applications Proceedings (MeMeA), 2013 IEEE International Symposium on, Gatineau, QC, 2013, pp. 58-62. doi: 10.1109/MeMeA.2013.6549706

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1, 2009.

[5] Cubesensors, https://cubesensors.com/

[6] Foobot, http://foobot.io/

[7] NetAtmo, https://www.netatmo.com/en-US/product/weather-station

[8] A. Bruce-Konuah. 2014. Occupant window opening behaviour: the relative importance of temperature and carbon dioxide in university office buildings. Ph.D Dissertation. The University of Sheffield, UK.

[9] L. M. Candanedo, V. Feldheim. 2016. Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models. Energy and Buildings, 112, 28-39.

[10] Z. Han, R. X. Gao, Z. Fan. 2012. Occupancy and indoor environment quality sensing for smart buildings. In Instrumentation and Measurement Technology Conference (I2MTC), 882-887.

[11] J. Ramos, J. H. Hong, A. K. Dey, “Stress recognition: a step outside the lab”, Proceedings of the International Conference on Physiological Computing Systems, 2014.

[12] A. Muaremi, A. Bexheti, F. Gravenhorst, B. Arnrich, G. Troster, “Monitoring the Impact of Stress on the Sleep Patterns of Pilgrims using Wearable Sensors,” IEEE-EMBS Int. Conference on Biomedical and Health Informatics, pp. 3-6, 2014.

[13] K. Hovsepian et al. “cStress: Towards a Gold Standard for Continuous Stress Assessment in the Mobile Environment”, UBICOMP, JAPAN, 2015.

[14] G.K. Palshikar, ”Simple Algorithms for Peak Detection in Time-Series”. Ii the Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence, June 2009

[15] H. Gjoreski, S. Kozina, M. Gams, M. Luštrek, Álvarez-García JA, Hong JH, Ramos J, Dey AK, Bocca M, Patwari N. Competitive Live Evaluation of Activity-recognition Systems. IEEE Pervasive Computing, Vol:14 , Issue: 1, pp. 70 – 77 (2015).]

[16] K. Dedovic, R. Renwick, N. K. Mahani, and V. Engert, “The Montreal Imaging Stress Task : using functional imaging to investigate the ...,” vol. 30, no. 5, pp. 319–325, 2005.

[17] J. M. Smyth and A. A. Stone, “Ecological Momentary Assessment Research In Behavioral Medicine,” pp. 35–52, 2003.

[18] B. Cvetković, V. Mirchevska, V. Janko, M. Luštrek, "Recognition of high-level activities with a smartphone", Proc.UbiComp/ISWC'15, 2015.

[19] C. D. Spielberger, “State-Trait Anxiety Inventory,” Anxiety, vol. 19. p. 2009, 1987.

[20]OpenNI, http://openni.ru/

[21] F, Martin, A. Gradišek, B. Cvetković, M. Luštrek, "An intelligent system to improve THC parameters at the workplace." Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct. ACM, 2016.