CONFIDENCE

UBIQUITOUS CARE SYSTEM TO SUPPORT INDEPENDENT LIVING

Small or medium-scale focused research project

DELIVERABLE **D3.2-3**

Report on Performance of Indoor Reconstruction and Interpretation Systems, Indoor Reconstruction and Interpretation System Prototypes Specification and Test Plan

Contract number:	214986
Project acronym:	CONFIDENCE
Project title:	UBIQUITOUS CARE SYSTEM TO SUPPORT INDEPENDENT LIVING

Deliverable number:	**D3.2**
Nature:	Report
Dissemination level:	PU – Public
Report date:	2009-02-02

Author(s):	Mitja Luštrek, Boštjan Kaluža, Erik Dovgan, Matjaž Gams, Igone Velez
Partners contributed:	JSI, CEIT
Contact:	Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia +386 1 4773380, mitja.lustrek@ijs.si

The CONFIDENCE project was funded by the European Commission under the 7th Framework Programme (FP7) –theme 3 “Information & Communication Technologies”.

ICT 1-7.1 ICT & Ageing

CONFIDENCE

UBIQUITOUS CARE SYSTEM TO SUPPORT INDEPENDENT LIVING

Small or medium-scale focused research project

Reviewed by:	**Approved by:**	
Andrés García Alonso External Reviewer CEIT	Igone Vélez CONFIDENCE Project Coordinator CEIT	Pablo de Lecea Head of CONFIDENCE Project Management Office CEIT

INDEX

1. INTRODUCTION

This deliverable reports on the performance of the reconstruction and interpretation subsystem on simulated data. In the absence of the Confidence hardware for the acquisition of tag coordinates, the data was recorded by an infrared motion capture system. The persons whose behaviour was recorded were healthy young volunteers imitating the elderly under the supervision of a physician. The reconstruction subsystem recognizes the user's posture and movement as one of six activities. Various methods for reconstruction are described and their performance compared. The final result is the reconstruction accuracy with respect to the level of sensor noise and the number of tags to be worn by the user. The interpretation subsystem recognizes falls and interprets walking as normal or abnormal. The accuracy of these two tasks is reported with respect to the level of sensor noise and the number of tags as well.

The deliverable also presents the architecture of the prototype of the reconstruction and interpretation subsystem and a plan for testing its performance. The testing will consist of two stages: the first stage will take place before the prototype of the Confidence hardware is ready and will mostly utilize the data already available; in the second stage, new data will be recorded with the Confidence hardware.

2. DATA ACQUISITION

We acquired 815 recordings of behaviours of interest performed by five healthy young volunteers imitating the elderly under the supervision of a physician (to ensure the authenticity of the behaviours). The recordings took place in two phases. Reconstruction subsystem development only utilized the first-phase data, while the interpretation subsystem was developed with the help of the second-phase data as well.

First phase

For the purpose of recognizing activities, which is what the reconstruction subsystem does, each behaviour was split into the following activities: walking/standing, sitting, lying, the process of sitting down, the process of lying down and falling.

- 45 recordings of falling, consisting of walking/standing normally, falling and lying (the volunteer tried to fall differently each time).
- 30 recordings of lying down, consisting of walking/standing normally, lying down and lying.
- 30 recordings of sitting down, consisting of walking normally, sitting down and sitting.
- 90 recordings of walking normally (30 straight, 30 in circle, 30 with a stop).
- 30 recordings of walking with a burden.
- 30 recordings of walking limping.

Second phase

- 165 recordings of falling (11 types of falls).
- 70 recordings of lying down (4 different ways).
- 25 recordings of sitting down and standing up.
- 25 recordings of walking normally.
- 150 recordings of walking abnormally (25 limping due to pain in the back, 25 limping due to pain in the leg, 50 dizzily, 25 with hemiplegia, 25 with Parkinson's disease).
- 25 recordings of standing up abnormally (with Parkinson's disease).
- 25 recordings of lying abnormally (with epilepsy).
- 75 recording of other abnormal behaviours (25 starting walking with Parkinson's disease, 25 chorea, 25 agitation).

The recordings consisted of the coordinates of 12 tags worn on shoulders, elbows, wrists, hips, knees and ankles. Since the Confidence hardware for the acquisition of tag coordinates is not available yet, the coordinates were acquired with the Smart infrared motion capture system [11]. The Smart system adds negligible noise to the coordinates (under 1 mm) and samples them with 60 Hz. The use of the Smart system allowed us to control the total amount of noise by adding varying degree of Gaussian noise to the raw coordinates. The standard deviation of the added noise ranged from 0 to 13.08 cm horizontally and 16.32 cm vertically. The maximum noise corresponds to triple the noise measured in the Ubisense real-time location system [13], which is similar to the hardware being developed in the Confidence WP2. We hereafter refer to the levels of noise in multiples of the measured Ubisense noise (reference noise). The coordinate sampling rate was reduced from 60 Hz to 10 Hz, which is the frequency planned for Confidence. The noise was smoothed with Kalman filter [8].

3. RECONSTRUCTION PERFORMANCE

The reconstruction subsystem recognizes the user's activity as one of the following: walking/standing, sitting, lying, the process of sitting down, the process of lying down and falling. This is accomplished by machine learning: a classifier is trained that recognizes the activity from a one-second interval of the user's behaviour (other durations were tried, but one second proved most suitable). The attribute vector for machine learning is a concatenation of the attributes belonging to the ten snapshots of the user's posture in that interval. Six attribute sets briefly described in Subsection 3.1 were considered. More detailed descriptions of the attributes can be found in the Deliverable D3.1 [7]. Experimental comparison of the attributes and 11 machine learning algorithms is presented in Subsection 3.2 and [6]. The removal of some spurious activities is described in Subsection 3.3 and the final results are in Subsection 3.4.

3.1. Attributes for machine learning

3.1.1. Reference attributes

The reference coordinate system is immobile with respect to the environment. The reference attributes consist of the z coordinates, the velocities and the z components of velocities of all the tags in each of the ten snapshots of the user's posture within the one-second interval to be classified. The x and y coordinates were omitted because the location where an activity takes place is not important. Additional attributes are the absolute distances and the distances in the z direction between all pairs of tags.

3.1.2. Body attributes

The body coordinate system is affixed to the user's body. It enables the observation of the x and y coordinates of the user's body parts, since these coordinates no longer depend on the location in the environment.

We considered four variants of the body coordinate system differing in two characteristics. First, the coordinate system may be either fully affixed to the body or it may use reference z coordinates, since z coordinates are typically not dependent on the location in the environment. Second, it may be computed for each snapshot in the one-second interval separately or it may be computed for the first snapshot in the interval and the coordinates in the remaining snapshots expressed in the coordinate system of the first snapshot.

The attributes common to all the variants of the body coordinate system are the x, y and z coordinates, the absolute velocities and the angles of movement of all the tags. Additional attributes describe the relation between the body and reference coordinate systems: the z coordinate, the absolute velocity and the angles of movement of the origin of the body coordinate system expressed in the reference coordinate system, and the orientation of the body coordinate system with respect to the reference one. The additional attributes appear only once per attribute vector in the first-snapshot variant of the body attributes, since there is only one body coordinate system in that case.

3.1.3. Angle attributes

These are the angles between adjacent body parts: the shoulder, elbow, hip and knee angles and the angle between the lower and upper torso. The shoulder, hip and torso have three degrees of freedom, so their angles are represented by quaternions (mathematical constructs often used for describing 3D rotations); the remaining two joints have only one degree of freedom each, so their angles are represented by scalars.

3.2. Machine learning algorithms and experiments

The data for machine learning were prepared as follows. Sections of the first-phase recordings described in Section 2 were first manually labelled with the activities of interest. The recordings were then split into overlapping one-second intervals (one interval starting every one-tenth of a second). Afterwards, the attributes described in Subsection 3.1 were extracted from these intervals. This resulted in 5,760 attribute vectors consisting of 240–2,700 attributes each (depending on the combination of attributes used). An activity was assigned to each attribute vector. Finally, these vectors were used as training data for 11 machine learning algorithms: C4.5 decision trees, RIPPER decision rules, Naive Bayes, 1-, 3-, 5- and 10-Nearest Neighbour, Support Vector Machine, Random Forest, Bagging and Adaboost M1 boosting. The algorithms were implemented in the Weka machine learning suite [15] and ran with the default parameter values.

Machine learning experiments proceeded in three steps. In the first step we compared the classification accuracy of the six single attribute sets: reference, every-snapshot body with body z, every-snapshot body with reference z, first-snapshot body with body z, first-snapshot body with reference z and angle. We used three levels of noise: none, reference noise and reference noise × 2. The results of the first step of experiments are shown in Table 1. The accuracy was computed with ten-fold cross validation. The accuracy of the best attribute set for each algorithm is in bold type; the accuracy of the best algorithm for each attribute set is on gray background.

Attribute set / **Algorithm**	Reference	Every-snapshot body with body z	Every-snapshot body with reference z	First-snapshot body with body z	First-snapshot body with reference z	Angle
	No noise					
C4.5 decision trees	**94.1**	92.8	93.7	92.9	93.2	91.8
RIPPER decision rules	**93.1**	91.4	92.8	92.0	93.0	90.9
Naive Bayes	89.5	88.7	**90.6**	86.8	88.2	76.7
1-Nearest Neighbour	**96.9**	89.9	81.2	86.6	84.0	96.7
3-Nearest Neighbour	**97.1**	92.0	82.8	88.1	85.1	96.9
5-Nearest Neighbour	**96.5**	92.0	83.8	88.0	85.1	96.2
10-Nearest Neighbour	**95.1**	91.9	84.9	87.9	85.8	94.1
Support Vector Machine	**97.7**	94.4	95.0	94.1	94.3	90.5
Random Forest	**97.0**	96.5	96.8	96.0	96.0	96.8
Bagging	**95.9**	95.3	95.7	95.4	94.9	94.5
Adaboost M1 boosting	**97.7**	94.9	95.3	94.7	94.7	94.4
	Reference noise					
C4.5 decision trees	**90.1**	88.4	89.9	88.9	90.0	80.8
RIPPER decision rules	87.5	84.7	88.1	86.2	**88.6**	80.0
Naive Bayes	83.9	79.1	**84.0**	81.0	82.2	78.2
1-Nearest Neighbour	**94.6**	69.8	76.0	70.1	71.5	92.7
3-Nearest Neighbour	**95.3**	74.6	79.7	73.4	74.7	93.3

5-Nearest Neighbour	**94.9**	76.9	81.7	76.3	75.5	93.1
10-Nearest Neighbour	**93.8**	76.7	83.2	77.9	75.1	91.5
Support Vector Machine	**96.3**	87.2	91.6	89.9	91.1	87.2
Random Forest	**93.9**	90.5	93.4	91.9	93.2	90.5
Bagging	**93.6**	91.8	93.3	92.3	93.5	89.1
Adaboost M1 boosting	**93.2**	92.0	93.1	92.1	92.9	88.4
	Reference noise × 2					
C4.5 decision trees	85.3	81.9	**85.6**	83.8	85.4	68.7
RIPPER decision rules	82.7	79.5	84.1	81.0	**84.6**	70.5
Naive Bayes	**81.2**	75.0	**81.2**	77.9	80.1	77.1
1-Nearest Neighbour	**91.0**	61.8	69.8	61.6	64.1	83.4
3-Nearest Neighbour	**91.6**	64.6	73.2	62.9	69.0	85.4
5-Nearest Neighbour	**91.3**	67.4	74.9	65.8	70.7	85.3
10-Nearest Neighbour	**90.6**	68.2	75.1	67.0	70.3	84.7
Support Vector Machine	**94.6**	85.5	88.4	86.2	89.9	82.2
Random Forest	**89.9**	84.8	89.3	86.7	89.6	80.2
Bagging	**91.2**	88.8	90.9	89.8	90.3	81.4
Adaboost M1 boosting	90.5	88.3	**90.7**	88.9	90.5	79.2

Table 1. Classification accuracy for all the algorithms and all single attribute sets.

For the second step of machine learning experiments, we retained the best machine learning algorithms and the best attribute sets. To rank them, we compared the classification accuracies of all pairs of algorithms and all pairs of attribute sets. Table 2 shows the number of comparisons in which a given algorithm statistically significantly ($p < 0.05$) wins over another algorithm, minus the number of comparisons where it loses. The accuracies of the algorithms selected for the second step are on grey background; the accuracy of the best algorithm is in bold type. If we take into account all the attribute sets, the Nearest Neighbour algorithms perform substantially worse than the other algorithms selected for the second step. However, using reference and angle attributes, they work comparably to the top performers, so we retained one of them. We chose 3-Nearest Neighbour, because it performs best with reference and angle attributes, even though 5-Nearest Neighbour gives better results if all the attribute sets are taken into account.

Table 3 shows for the attribute sets what Table 2 shows for the machine learning algorithms. The difference is that only the results for the algorithms selected for the second step are included, since the performance with the other algorithms should not affect our decision. Considering that the second step was to consist of combining the attribute sets, we wanted to retain all the sets with any chance of performing well in combination. Thus we discarded only two redundant attribute sets: each-snapshot body attributes with body z (outperformed by the very similar first-snapshot variant) and the first-snapshot body attributes with reference z (outperformed by the each-snapshot variant).

Algorithm	**Wins – losses**		
	No noise	**Reference noise**	**Reference noise × 2**
C4.5 decision trees	–9	–4	–9
RIPPER decision rules	–14	–15	–16
Naive Bayes	–43	–29	–23
1-Nearest Neighbour	–24	–29	–35
3-Nearest Neighbour	–7	–13	–18
5-Nearest Neighbour	–9	–10	–7

10-Nearest Neighbour	–18	–15	–12
Support Vector Machine	25	23	33
Random Forest	**52**	31	17
Bagging	26	**32**	**38**
Adaboost M1 boosting	21	29	32

Table 2. The number of wins – losses of every algorithm against the others.

Attribute set	**Wins – losses**		
	No noise	**Reference noise**	**Reference noise × 2**
Reference	16	19	20
Every-snapshot body with body z	1	–12	–13
Every-snapshot body with reference z	0	11	11
First-snapshot body with body z	–6	–8	–10
First-snapshot body with reference z	–5	6	9
Angle	–6	–16	–17

Table 3. The number of wins – losses of every single attribute set against the others taking into account the algorithms selected for the second step of machine learning experiments.

After selecting the best algorithms and attribute sets, we proceeded with the second step of machine learning experiments, in which we tried combinations of attribute sets. Table 4 shows the classification accuracy for the four algorithms we retained and the combinations of the attribute sets we retained. The accuracy of the best combination of attributes for each algorithm is in bold type; the accuracy of the best algorithm for each combination of attributes is on gray background.

Attribute set combination **Algorithm**	Reference + first-snapshot body with body z	Reference + every-snapshot body with reference z	Reference + angle	First-snapshot body with body z + angle	Every-snapshot body with reference z + angle	Reference + first-snapshot body with body z + angle	Reference + every-snapshot body with reference z + angle
	No noise						
3-Nearest Neighbour	96.6	96.2	**97.1**	94.6	91.9	96.6	96.3
Support Vector Machine	96.7	96.9	**97.7**	95.6	95.5	96.9	96.9
Random Forest	96.9	97.0	**97.2**	96.6	96.9	96.9	97.0
Bagging	96.0	96.0	96.1	95.6	95.7	**96.2**	96.0
Adaboost M1 boosting	**95.6**	**95.6**	95.5	95.2	95.3	**95.6**	95.5
	Reference noise						
3-Nearest Neighbour	92.2	90.6	**95.4**	85.0	84.6	92.7	91.2
Support Vector Machine	94.9	95.4	**96.5**	91.3	92.5	95.0	95.5
Random Forest	**94.5**	94.2	94.1	92.6	93.5	94.4	94.0
Bagging	**94.2**	94.1	93.7	93.1	93.4	**94.2**	94.1
Adaboost M1 boosting	**93.8**	93.7	93.2	92.5	93.3	**93.8**	93.7

	Reference noise × 2						
3-Nearest Neighbour	89.2	87.0	**91.5**	76.7	79.4	89.9	88.1
Support Vector Machine	93.4	93.5	**94.7**	87.8	89.6	93.7	93.8
Random Forest	91.1	90.2	89.9	87.6	89.2	**91.2**	90.4
Bagging	92.3	92.3	91.4	89.9	91.0	92.3	**92.4**
Adaboost M1 boosting	**92.2**	91.9	90.7	89.4	90.8	**92.2**	92.0

Table 4. Classification accuracy for the retained algorithms and combinations of attribute sets.

Table 4 shows that the Support Vector Machine algorithm with the reference + angle attributes is the most accurate combination. However, for the final verdict we decided to tune the parameters of the algorithms, which was carried out in the third step of the machine learning experiments. We only retained three algorithms for this step: Support Vector Machine on the reference + angle attributes, 3-Nearest Neighbour on the same attributes and Bagging on the reference + first-snapshot body with body z + angle attributes. We discarded the Random Forest and Adaboost M1 algorithms, because they are similar to Bagging (all three are ensemble methods on decision trees) and achieved worse classification accuracy on average over the three noise levels.

Changing the complexity parameter in the Support Vector Machine algorithm from $c = 1.0$ to $c = 0.2$ increased the classification accuracy at reference noise from 96.5 % to 96.9 %, which is a small but statistically significant ($p < 0.05$) improvement. Tuning of the Nearest Neighbour algorithm did not yield any improvement whatsoever. Tuning the Bagging algorithm was more productive: increasing the number of iterations from 10 to 30 and using C4.5 instead of Fast Decision Tree Learner as the base algorithm increased the accuracy by 0.7, 1.2 and 1.0 percentage points at the three noise levels. However, the highest accuracy still belonged to Support Vector Machine, which is what we used henceforth.

3.3. Removing spurious activities

The reconstruction of the user's activity can be improved by taking into account the continuity of activities, e.g., the user's activity cannot switch between walking and sitting down every one-tenth of a second. The activity recognition classifier made this kind of mistakes more often than it misclassified longer intervals of an activity.

To remove spurious activity transitions, the activities returned by the activity recognition classifier were smoothed with Hidden Markov Model (HMM) [10]. The classifications returned by the activity recognition classifier were considered the observations in the HMM and the manually labelled true activities were considered the hidden states. The observation of the activity O in the hidden state corresponding to the activity S indicates a correct classification if $O = S$ and an incorrect one otherwise.

To use a HMM, its parameters need to be learned. This was done by the Baum-Welch algorithm [2] as implemented in the Jahmm toolbox [5]. The algorithm assigns probabilities to state transitions and observations in each state. These probabilities corresponded to the probabilities of transitions between activities and the activities returned by the activity recognition classifier (correct and incorrect) given each true activity.

The actual reduction of spurious activity transitions was accomplished by the Viterbi algorithm [14]. The algorithm computes the most likely sequence of states given a sequence of observations. This corresponds to the most likely sequence of true activities given a sequence of activities returned by the activity recognition classifier.

The reduction in the number of spurious activities with respect to the noise level after smoothing with HMM is shown in Figure 1. The noise is expressed in multiples of our reference noise. The experimental results were obtained with ten-fold cross validation.

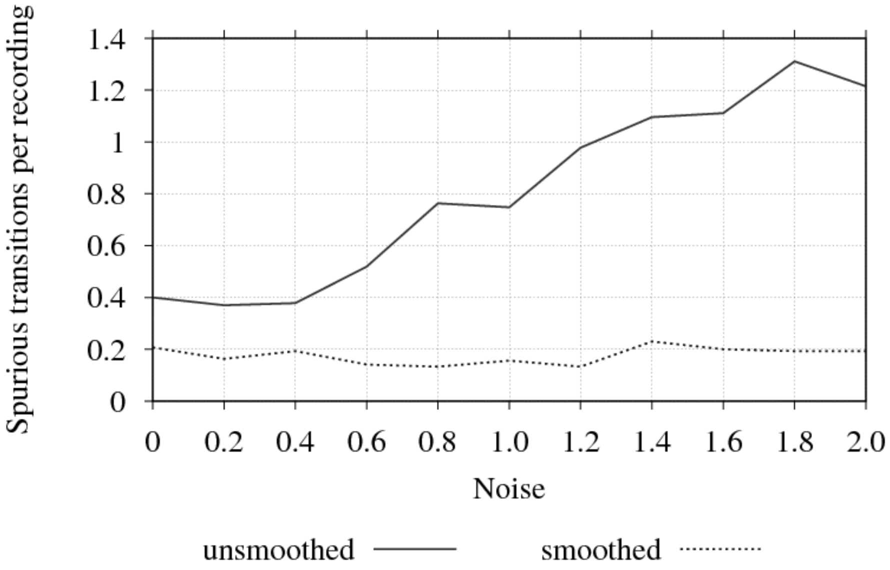

Figure 1. The number of spurious activity transitions with respect to the noise level before and after smoothing with HMM.

Figure 1 shows that smoothing with HMM almost eliminated spurious activity transitions. However, it reduced the classification error averaged over all the noise levels only from 6.5 % to 5.9 %. The reason why error reduction was not larger is that most of the error was caused by longer intervals of misclassified activities, which HMM cannot recognize as erroneous.

3.4. Tag placement and noise level

The number and placement of tags to be used by the Confidence system is yet to be determined. It must achieve a trade-off between usability and technical requirements – the users of course prefer as few tags as possible, but too few tags cannot ensure sufficient reconstruction and interpretation accuracy. The sensor noise of the hardware for the acquisition of tag coordinates is also not known yet, since the hardware is still under development. Therefore it makes little sense to pick a single tag placement and noise level. We instead report the classification accuracy with various tag placements and at various noise levels. To obtain these results, we proceeded in two steps.

In the first step we compared the classification accuracies achieved in the activity recognition task by all $2^{12} - 1 = 4095$ combinations of 1 to 12 tags. Three levels of noise were used: none, reference noise and reference noise × 2. For each number of tags, we selected ten tag placements with the highest average accuracy over all three noise levels for the use in further experiments. The best tag placement for each number of tags is shown in Figure 2.

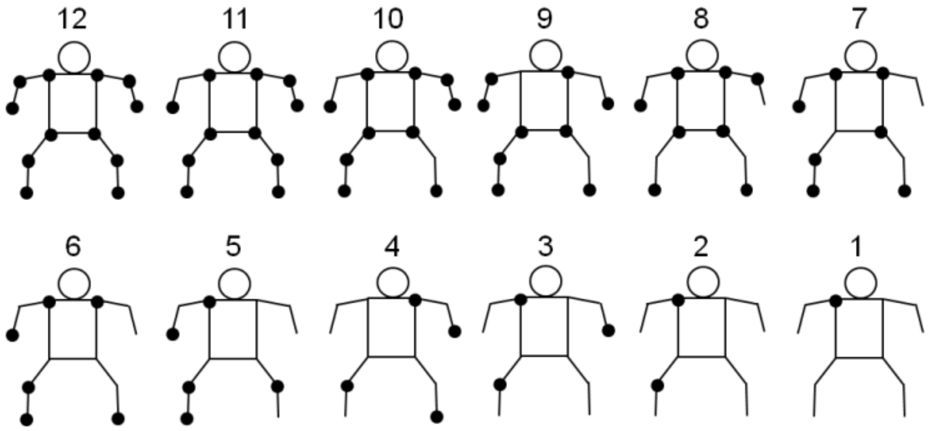

Figure 2. Best tag placement for each number of tags in for the reconstruction subsystem.

In the second step, we computed the classification accuracy for the ten best tag placements selected in the first step and every noise level from none to reference noise × 3 in increments of reference noise × 0.2. Figure 3 shows the accuracy of the best tag placement for each number of tags and each noise level.

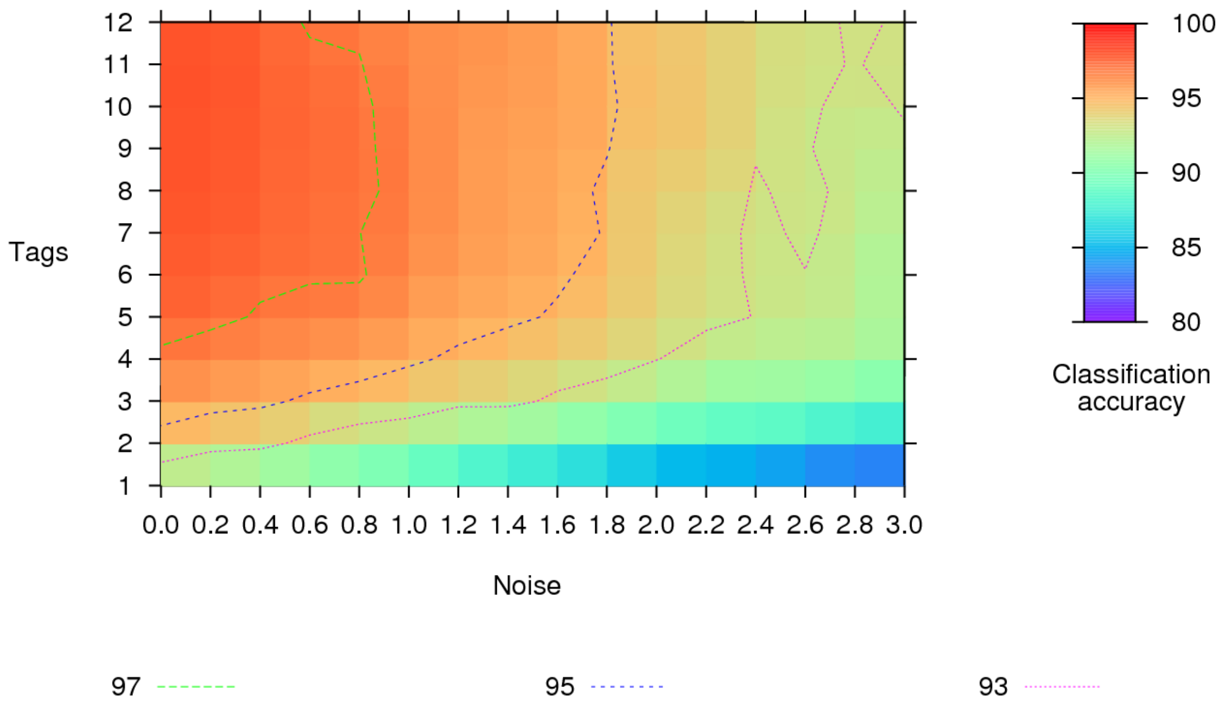

Figure 3. Classification accuracy with respect to the number of tags and noise level for the reconstruction subsystem.

The target accuracy set in the Description of Work is 95 %. As can be seen in Figure 3, at Reference noise, four tags are needed for the classification accuracy of over 95 %. Doubling the noise reduces the accuracy by only two percentage points. However, the 95 % accuracy can be achieved at reference noise × 1.8 with nine tags, at reference noise × 1.6 with six tags and at reference noise × 1.4 and 1.2 with five tags.

4. INTERPRETATION PERFORMANCE

After the reconstruction subsystem recognizes the user's activity, the interpretation subsystem takes over. Its purpose is to recognize events that warrant an alarm or warning. The scenarios in which such events occur are listed in the Deliverable D1.1 [1]. Among them are several types of falls, which are probably the most important class of alarm events – the accuracy of fall detection is reported in Subsection 4.1. As to the other scenarios, it was decided that instead of focusing on specific scenarios, we will adopt a more flexible approach, in which a number of statistics of daily living are tracked and a warning is raised if they deviate from normal. We have developed methods for the detection of abnormal walking, which are capable of detecting any problems (and more) that the rapid gait test scenario from D1.1 may signal, and some problems covered by the timed "up and go" scenario. The methods and their accuracy are described in Subsection 4.2. Methods for tracking the statistics of other activities are under development.

4.1. Falls

The recordings are interpreted as containing a fall using a simple rule: if there are three snapshots where the activity is falling, followed by one snapshot where the activity is lying, the recording contains a fall. Figure 4 shows the accuracy of fall detection for the best tag placement for each number of tags and each noise level. The best tag placement means the best for activity recognition, since fall detection is a very direct application of activity recognition.

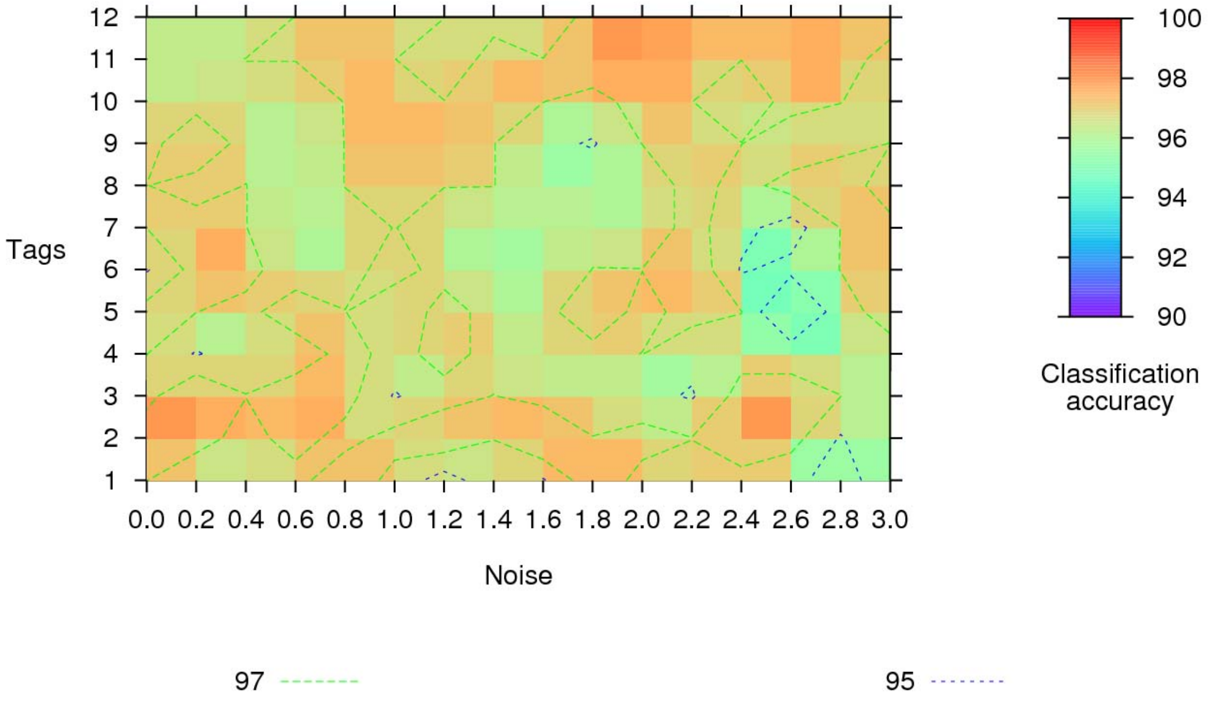

Figure 4. Classification accuracy with respect to the number of tags and noise level for the interpretation of recording as containing a fall.

According to Figure 4, the accuracy of fall detection seems independent of the tag placement and noise level. In approximately half the cases the accuracy is above 97 % and it hardly ever drops below 95 %.

4.2. Walking

The way a person walks can be characterized by a number of attributes known from medical literature [4][9]. Such attributes are used by physicians and medical researchers to describe human gait and are well suited to distinguishing normal walking from abnormal. Most attributes refer to a pair of steps, so to compute them we had to develop an algorithm for step detection.

We detect steps by observing the x and y coordinates of the user's ankles (the signal-to-noise ratio in the z coordinates is too low). For each snapshot of the user's posture, the distance in the xy plane an ankle has travelled from the previous snapshot is computed first. The snapshots are then sorted by this distance and divided into three groups: L with the lowest 30 % of the distances, H with the highest 30 % and the last group with those in between. The snapshots in the L group are considered standing still. Each period of standing still is refined by moving its boundaries to the first and last snapshot with an above-average distance for the group. The H group contains potential peaks of steps. The below-average members of the group are eliminated first. Then the middle peak between two periods of standing still is selected as the true peak. The procedure is illustrated in Figure 5.

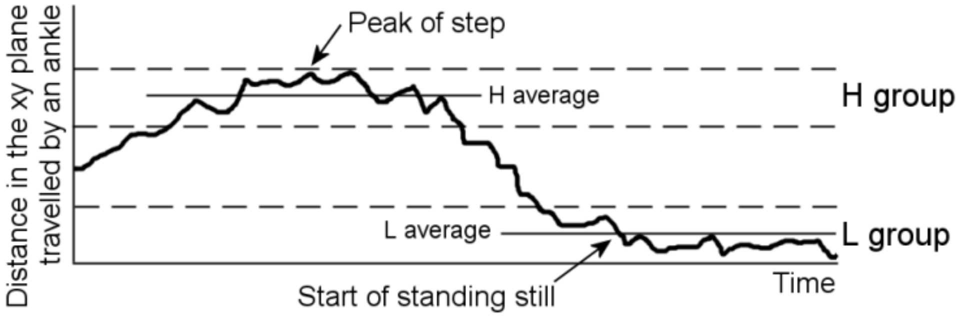

Figure 5. The working of the step detection algorithm.

4.2.1. Walking attributes

Each attribute refers to one step with each leg (two steps in all). Wherever applicable, the attributes are computed for left and right leg separately and the difference in the value for each leg is also included:

- Support (foot on the ground), swing (foot off the ground) and step (support + swing) times.
- Double support time (both feet on the ground).
- Step length and width.
- Maximal distance of the foot from the ground.
- Ankle, knee and hip angles upon touching the ground.
- Knee angle when the ankle of the leg on the ground is directly below the hip and knee angle of the opposite leg at that time.
- Minimal and maximal knee and hip angles, the angle of the torso with respect to the ground, and the range for each.
- Hip and shoulder sway (the difference between the extreme left and right deviation from the line of walking).

4.2.2. Machine learning, tag placement and noise levels

To interpret walking as normal or abnormal, machine learning was used. Two classifiers were developed. The first classified walking into one of the following classes: normally, limping, with a burden (a weight carried in one hand), dizzily, with hemiplegia and with Parkinson’s disease. The second classifier distinguished only between normal and abnormal walking. Normal walking consisted of walking completely normally and walking with a burden; abnormal walking consisted of the other four types of walking. The classifiers were trained on the recordings described in Section 2, which were labelled with the type of walking. The step detection algorithm first split the recordings into pairs of steps, after which the walking attributes were computed for each pair. This resulted in around 534 attribute vectors (depending on how many steps were detected), consisting of up to 58 attributes (depending on tag placement). These vectors were used as training data for C4.5 machine learning algorithm, which offered the best performance from a number of algorithms tested. The algorithm was implemented in Weka [15] as J4.8 and run with the default parameter values. The classification accuracy was computed with ten-fold cross validation.

We again studied the classification accuracy with respect to tag placement and noise level. Four tag placements were considered. Ankles tags were always used, since they are needed for step recognition. They were first used alone, then knee tags were added (four tags in all), then hip tags (six) and finally shoulder tags (eight). Noise level was varied from none to reference noise × 3 in increments of reference noise × 0.2. Figure 6 shows the accuracy for each tag placement and noise level for the classification into the six walking types and Figure 7 into normal/abnormal walking.

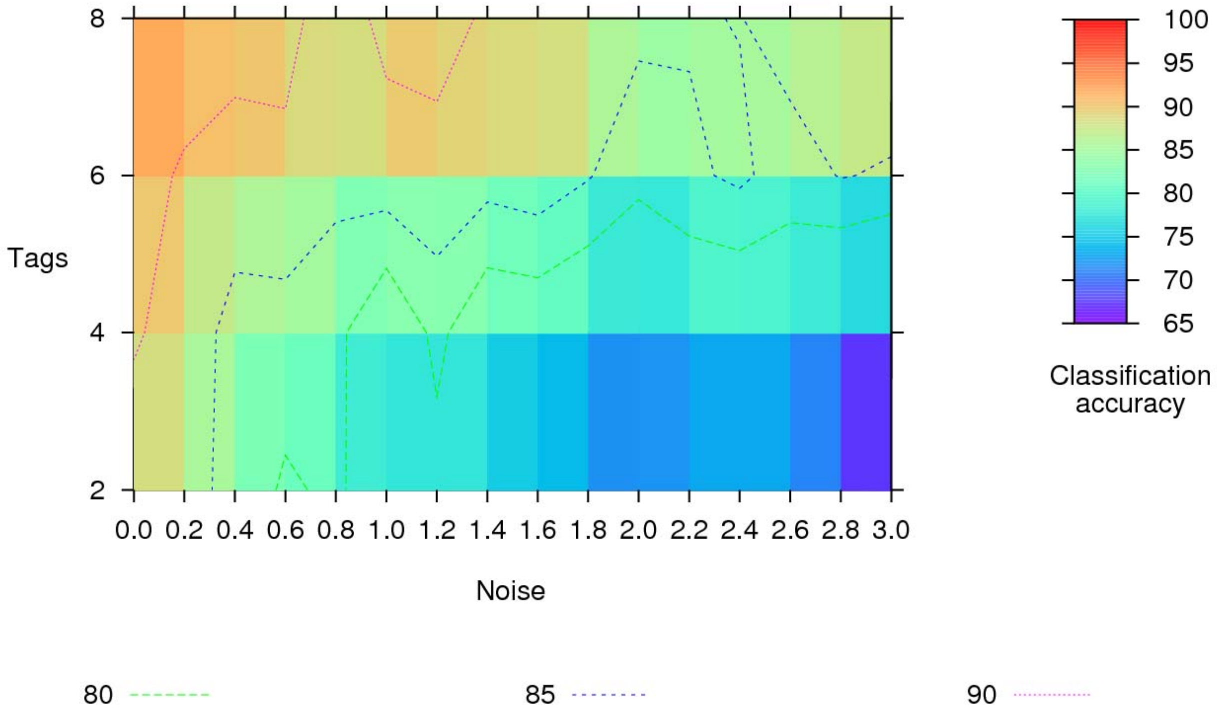

Figure 6. Classification accuracy with respect to the number of tags and noise level for the interpretation of walking as one of six types.

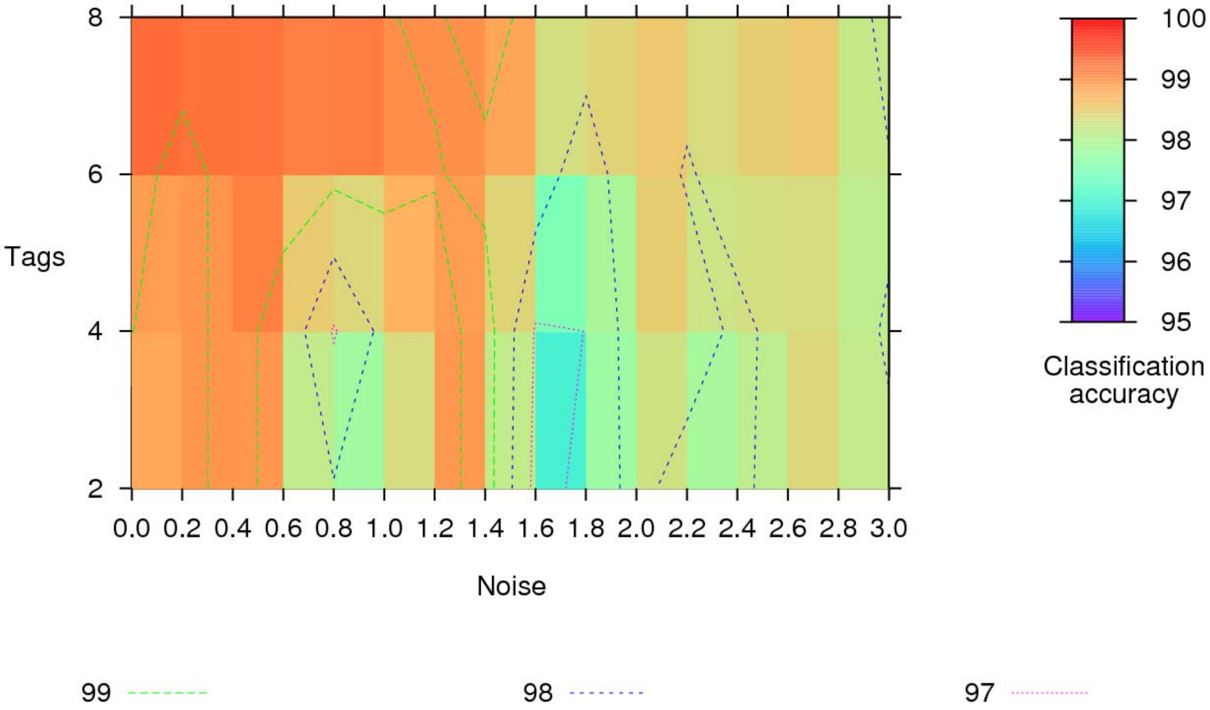

Figure 7. Classification accuracy with respect to the number of tags and noise level for the interpretation of walking as normal/abnormal.

According to Figure 6, the classification accuracy for the six walking types with the full eight tags is between 85 and 95 % and does not vary much with the level of noise. The accuracy with six tags is around 85 % and also varies little with noise level. With four and two tags, the impact of noise becomes more pronounced: the accuracy ranges from a little above 85 % to a little below 70 %. The accuracy is expected to increase if the walking attributes are averaged over a longer period of time, but at this time it does not meet the 95 % requirement stated in the Description of Work. However, for the Confidence project it is not necessary to recognize the exact type of walking. The accuracy of the recognition of abnormal walking of any type is shown in Figure 7. It rarely drops below 98 % and never below 96 %, which is well within the requirements.

5. PROTOTYPE SPECIFICATION

The architecture of the reconstruction and interpretation subsystem prototype shown in Figure 8 is based on the architecture proposed in the Deliverable D3.1 [7]. The changes are due to the practical experience gained in the six months between D3.1 and this deliverable.

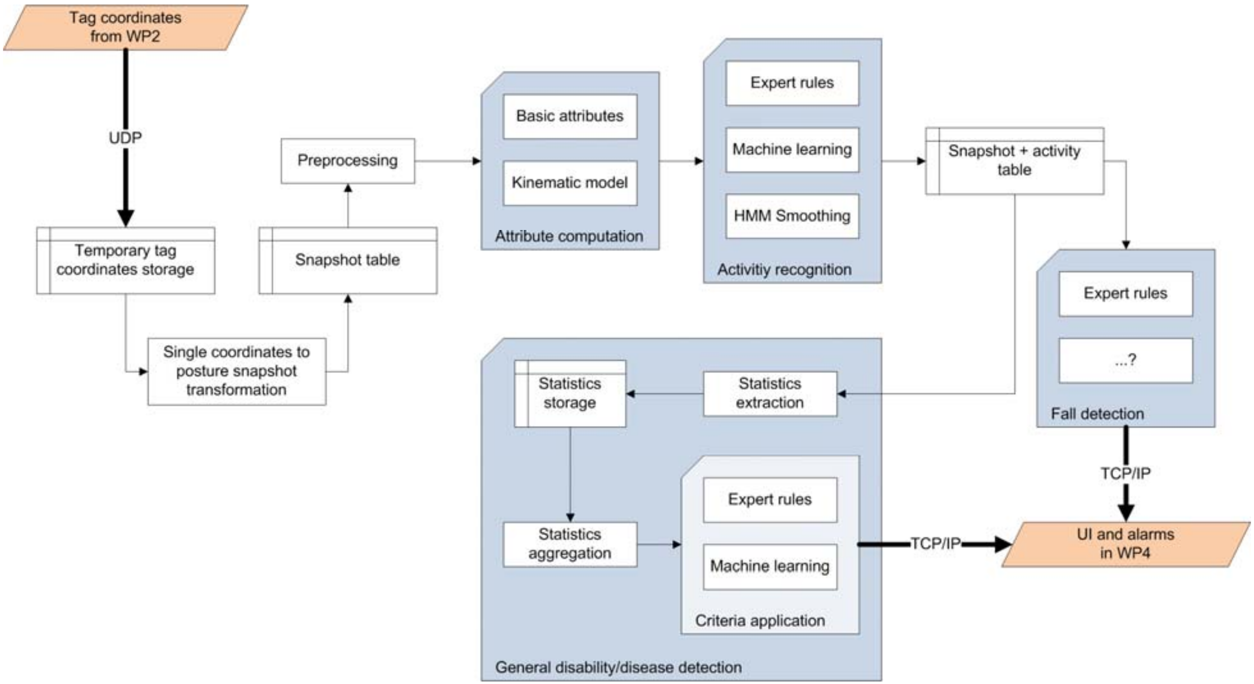

Figure 8. Architecture of the reconstruction and interpretation subsystem prototype.

The input to the reconstruction and interpretation subsystem are tag coordinates arriving from the localisation subsystem over the network using the UDP protocol. The coordinates of each tag are stored until the coordinates of all the tags for a given time interval (1/10 s) are gathered or until the next interval starts. At that point the coordinates are assembled into one snapshot of the user's posture and inserted into the snapshot table.

Each snapshot in the snapshot table is first preprocessed: the coordinates of missing tags are extrapolated and all the coordinates are filtered to reduce sensor noise. Afterwards, the attributes used by the reconstruction subsystem described in Subsection 3.1 are computed. Some of them require a kinematic model of the user's body to be built first. Attribute computation is followed by activity recognition. Activity recognition by machine learning, followed by Hidden Markov Model smoothing is described in Subsections 3.2 and 3.3. Activity recognition by means of expert rules will also be included in the prototype. When the current activity is recognized, it is included in the snapshot table to be available to the reconstruction subsystem.

Fall detection uses expert rules to infer when a fall has occurred based on the recognition of the user's activity as falling and lying. Additional methods, possibly employing machine learning, may be included in the prototype. Alarms will be passed to WP4 over the network using the TCP/IP protocol.

Disabilities and diseases in general will be detected by gathering statistics of daily living and observing when these statistics deviate from normal. The statistical attributes will first be extracted from the data in the snapshot + activity table. The attributes for walking are described in Subsection 4.2.1, but we expect to add attributes for other activities to the prototype. They will be stored and aggregated in various manners (average, standard deviation) over various periods of time. Finally, criteria for raising an alarm/warning will be applied to them. These criteria may have the form of expert rules or a classifier built by a machine learning algorithm. Such a classifier for distinguishing abnormal walking from normal is described in Subsection 4.2.2. Alarms and warnings will again be passed to WP4.

The prototype is being implemented in the Java [12] programming language and will initially be executed on a Windows PC. Java was selected for compatibility with the Weka [15] machine learning suite and for platform independence.

6. TEST PLAN

The testing will consist of two stages. The first stage will take place before the Confidence hardware for the acquisition of tag coordinates is available. It will mostly utilize the existing data described in Section 2. In the second stage, new recordings will be made with the Confidence hardware.

6.1. First stage

The first test stage has two objectives: to test the accuracy of the reconstruction and interpretation algorithms and to test the stability and performance of the prototype software.

Algorithms

These tests will measure the accuracy of reconstruction and interpretation. Two types of tests will be performed:

1. The reconstruction classifier will be trained on all the data from the first data acquisition phase and tested on the data from the second phase. It will also be trained on the recordings of all the volunteers but one and tested on the recordings of the final volunteer. We will thus check the robustness of the algorithms when the difference in training and test data is increased. A similar test will be performed for fall detection and walking interpretation.
2. Ten-fold cross validation will be performed on all the available data for all the algorithms. We will thus check the robustness of the algorithms when the variation in both training and test data is increased.

Prototype software

A program for the emulation of the Confidence data acquisition hardware will be developed. This program will transmit previously recorded tag coordinates over TCP/IP network to test whether the whole pipeline described in Section 5 is working and stable. We will also test whether the pipeline can process the 12 body tags and perform reconstruction and interpretation in real time.

6.2. Second stage

The second stage will measure the accuracy of reconstruction and interpretation with the Confidence hardware for the acquisition of tag coordinates. Behaviours similar to those described in Section 2 will be recorded with the Confidence hardware prototype. They will be performed by three to five healthy volunteers. We plan the following recordings (more may be added if necessary):

- 11 types of falls, 5 recordings per type per volunteer.
- 4 ways of lying down, 5 recordings per way per volunteer.
- Sitting down and standing up, 10 recordings per volunteer.
- Walking normally, 10 recordings per volunteer.
- 5 types of walking abnormally, 5 recordings per type per volunteer.

We will be able to test the performance of the reconstruction and interpretation algorithms in three ways:

1. Training on the existing recordings, testing on the new recordings.
2. Training and testing on the new recordings.

3. Training and testing on all the recordings.

7. CONCLUSION

The methods for reconstruction have been developed and tested. They achieved the target accuracy of over 95 % with reasonable combinations of tag placement and noise level. We can expect some small improvements with further refinements of the algorithms. We also intend to improve the confidence in our methods by adding expert rules that are easy to understand to the reconstruction subsystem. Finally, the recordings of some additional activities (standing up, the postures after some types of falls) are needed, but no modification of the algorithms will be necessary to include them.

At the moment we have a simple implementation of fall detection that is directly based on the reconstruction of activity as falling and lying. It nevertheless achieves the accuracy above the target of 95 % in almost all the cases, regardless of the level of noise. The fact that fall detection is almost independent of noise level is somewhat surprising, but quite welcome. The simplicity of the current implementation leaves some room for improvement. The Deliverable D1.1 [1] lists six indoor fall-related scenarios, which are all covered by the current fall detection (we considered 11 types of falls, which were obtained by further dividing some types of falls from D1.1).

The interpretation of walking has also been developed and tested. The recognition of abnormal walking achieved the accuracy well above the target 95 %. The recognition of specific types of abnormal walking was less accurate, but recognizing specific diseases is not really an objective of the Confidence project. In the future we will focus on the interpretation of other activities. Considering the large amount of medical literature on gait, walking is probably the most important one. The interpretation of walking covers the rapid gait test scenario from D1.1.

REFERENCES

[1] Antomarini, M., Cesaroni, F., Piangerelli, E., Sdogati, C., Moeberg, A., Larsson, K., Bergström, E, González, N., Palonen, E. M., Rosqvist, E., Vélez, I., Cortés, A., Von der Grün, T., Pietrzyk, M., Luštrek, M., Val, I., Quemada, C., Gomes, S., Gonnet, S., Tamburriello F., Amedei G., and Karlsson P. (2008). D1.1 User needs and requirements. Confidence project.

[2] Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Annals of Mathematical Statistics 41 (1), pp. 164–171.

[3] Cortés, A., Vélez, A., Val, I., Quemada, C., Gomes, S., Gonnet, S., von der Grün, T., and Pietrzyk, M. (2008). R.1.4 Interfaces for the system model and prototype. Confidence project.

[4] Heiden, T. L., Sanderson, D. J., Inglis, J. T., and Siegmund, G. P. (2006). Adaptations to Normal Human Gait on Potentially Slippery Surfaces: The Effects of Awareness and Prior Slip Experience. Gait & Posture 24, pp. 237–246.

[5] Jahmm (Java implementation of HMM related algorithms). http://www.run.montefiore.ulg.ac.be/~francois/software/jahmm, 2008-11-15.

[6] Luštrek, M., and Gams, M. (2008). Prepoznava položaja telesa s strojnim učenjem. In Proceedings of the 12th International Multiconference Information Society – IS 2008, vol. A, pp. 30–33, 2008.

[7] Luštrek, M., Nemec, B., Kaluža, B., Piltaver, R., Gams, M, Vélez, I., Larsson, K., and González, N. (2008). D3.1 Report on Indoor Reconstruction and Interpretation Techniques. Confidence project.

[8] Maybeck, P. S (1979). Stochastic models, estimation, and control. Mathematics in Science and Engineering 141.

[9] Paróczai, R., Bejek, Z., Illyés, Á., Kocsis, L., and Kiss, R. M. (2006). Gait Parameters of Healthy, Elderly People. Facta Universitatis 4(1), pp. 49–58.

[10] Rabiner, L. R. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. In Proceedings of the IEEE 77 (2), pp. 257–286.

[11] Smart motion capture system. http://www.emotion3d.com/smart/smart.html, 2008-09-15.

[12] Sun Microsystems. Java. http://java.sun.com, 2009-01-23.

[13] Ubisense. http://www.ubisense.net, 2008-09-15.

[14] Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory 13 (2), pp. 260–269.

[15] Witten, I. H., and Frank, E. (2005). Data Mining: Practical machine learning tools and techniques, 2nd Edition. Morgan Kaufmann, San Francisco, USA.