Journal of Intelligent & Fuzzy Systems 24 (2013) 215–228
DOI:10.3233/IFS-2012-0548
IOS Press

A fuzzy model for human fall detection in infrared video

Marina V. Sokolova[a], Juan Serrano-Cuerda[b], José Carlos Castillo[b] and Antonio Fernández-Caballero[b,*]
[a]*South-West State University, Kursk, Russia*
[b]*Instituto de Investigación en Informática de Albacete, Universidad de Castilla-La Mancha, Albacete, Spain*

Abstract. Fall detection, especially for elderly people, is a challenging problem which demands new products and technologies. In this paper a fuzzy model for fall detection and inactivity monitoring in infrared video is presented. The classification features proposed include geometric and kinematic parameters associated with more or less sudden changes in the tracked human-related regions of interest. A complete segmentation and tracking algorithm for infrared video as well as a fuzzy fall detection and confirmation algorithm are introduced. The proposed system is capable of identifying true and false falls, enhanced with inactivity monitoring aimed at confirming the need for medical assistance and/or care. The fall indicators used as well as their fuzzy model is explained in detail. The fuzzy model has been tested for a wide number of static and dynamic falls, demonstrating exciting initial results.

Keywords: Fall detection, video segmentation and tracking, infrared video, fuzzy system

1. Introduction

Nowadays, fall detection is still a challenging and emergent problem [1], especially for monitoring elderly people [8, 13]. The problem has mainly been provoked and induced by population ageing, showing a tendency of permanent growth in accordance with recent demographical predictions [29]. As reported, about a third part of humans aged 65 and above suffer from a fall each year, increasing up to 42 percent for elderly over 70 years. Moreover, every year approximately around 50 percent of humans who live in long-term care institutions suffer a fall, and 40 percent of them experience recurrent falls. In addition to this, elderly people are highly vulnerable, as falls are known to be one of the leading causes of injuries and death. Another important consequence of recurrent falls is the post-fall syndrome that manifests through depression, loss of autonomy, immobilization, and may result in impairments in daily activities [29].

One more reason of growing research in the field of fall detection is that governmental promotion through high public attention, and constant improvement of associated support technology in the area of elderly care, permits to carry out research and to implement new applications. This enables usage of professional up-to-date equipment and development of complex solutions based on advanced visual information analysis methods (e.g. [3, 17]). Consequently, the number of applications in this area has constantly grown. Indeed, in [31] an overview of the current situation in the field of elderly fall detection and alarm generation is presented. Firstly, the authors classify fall detection under camera-based, wearing-device-based and ambient-device-based approaches. In another review on the principles and algorithms for fall detection [21], the authors study analytical methods and machine learning techniques. Within the first group of methods, image processing and usage of sensors (horizontal

*Corresponding author. Antonio Fernández-Caballero, Instituto de Investigación en Informática de Albacete, Universidad de Castilla-La Mancha, 02071 - Albacete, Spain. Email: Antonio.Fdez@uclm.es.

inclination sensor, sensitive floor tiles, accelerometers, etc.) are included. In a second group, the authors place approaches based on machine learning for fall detection, covering all steps from observation to classification.

Within the camera-based group of methods, several approaches to fall detection have been proposed. For instance, head motion trajectory analysis is one of the indicators of a fall [23]. Many research works are dedicated to posture detection analysis with the intention not only to identify poses, but also to distinguish among true and false falls. In this way, in [28] the authors propose a 3D method from 2D pose reconstruction based on the introduction of bone, bone symmetry, and rigid body projection constraints. Another approach for fall detection, which is based on the *k*-nearest neighbor algorithm, is introduced in [15]. Moreover, 3D shape-encoded filters are described in [19]. These extract the boundary gradient information of the body image, which is then propagated by a branching particle system. In [25], the application of a kernel particle filter for 3D body tracking is introduced. In [18] a system for human behavior analysis is reported, where a human is represented with three components: human star skeleton, angles of six sticks in the star skeleton, and object motion vectors. Here the human posture classifier, based on a multi-category support vector machine, is used for pose detection.

In general terms, human fall detection starts with people monitoring and tracking and finishes with secure decision making on whether a human has fallen or not. It is a known fact that segmentation and tracking results are very sensitive to environmental conditions. The results are dependent on the quality of the image, on sudden changes in illumination (mainly in color video) and temperature (in infrared video), among others. In addition, humans have different types of body composition, which can be heavy, sturdy, slim, stocky, and so on. In 2-D images, humans can be overshadowed by other objects and be part of groups composed of two, three or more humans. Thereby, direct analysis of a human body appears to be an ambiguous and difficult task. Indeed, on the one hand, the principal anatomical proportions of human bodies are similar. But, also, personal physical characteristics of each human make it impossible to create a uniform crisp decision block. Thus, since humans usually appear warmer than the background in many environmental conditions, several methods which use thermal infrared imagery have been proposed for human detection [5, 11]. The thermal infrared imagery is specially suitable for human detection in indoor environments. These are the most usual areas of work for human fall detection. Moreover, temperature changes affecting the detection do not usually appear in indoor, since there are few external factors which can affect them (e.g., air currents and/or day/night heat variations).

There have been several approaches to infrared human detection so far [12]. For instance, a pedestrian detection method in thermal infrared imagery using the wavelet transform is proposed in [14]. The approach uses a support vector machine (SVM) classifier to establish true pedestrian regions. The regions are detected using features obtained from a conversion to the frequency domain used as an input to the SVM. In [7], a background subtraction is initially performed to identify local foreground objects. Then, gradient information from the objects and the background is combined to elaborate a contour map representing the confidence for each pixel to belong to a human's boundary. Finally, a path search is performed to complete broken segments. An approach based on histogram is introduced on [10], realizing a vertical and a horizontal adjustment on the pedestrian candidate regions. Two different cases for summer and winter are established, using the gray level values to delimit pedestrians in the winter case and the intensity changes in the human boundaries to delimit pedestrians in summer sequences. More recently, the Otsu algorithm [22] is used to binarize the image separating the warmer regions (probably belonging to humans) from the colder ones, using opening and closing morphological operations to remove small broken parts obtained from the binarization [6]. Next, objects that are too small or have a wrong aspect ratio to be humans are filtered out and a histogram of oriented gradients (HOG) method is used to extract features used to train an Adaboost classifier.

In this paper, we explain how human fall detection and inactivity monitoring in infrared video is realized within a fuzzy model. Fuzziness has been largely studied in the past for pattern recognition [9, 27]. In relation to fall detection, a method with fuzzy one class support vector machine in a smart room has been presented [30]. The fuzzy membership function represents the likelihood for the data to belong to a target class [20]. This is also our approach. The paper is organized as follows. Section 2 introduces the proposed approach towards human monitoring and fall detection, mainly explaining the required segmentation and fall detection algorithms. Then, in Section 3, data and experimental results are presented. Section 4 introduces some aspects of the accuracy of the proposed fuzzy

system and shows some comparison with other similar approaches. Finally, some final conclusions are provided in Section 5.

2. Human monitoring for fall detection

Our approach is based on the idea of combining a fall detection with an inactivity monitoring algorithm. In this case, the inactivity monitoring subsystem is thought for checking if a human continues laying inactive during some predefined time after a fall has been detected in the detection subsystem. In this paper, fall detection as such is based on the geometrical analysis of the segmented region of interest (ROI) representing a human. Roughly, it is decided that a fall has occurred through the study of the velocity of the deformation suffered by the ROI along an established time interval. This time interval is denominated the "Fall time", t_f, which typically takes values between 1 and 3 seconds. Additionally, fuzzy logic is used in order to make decisions more flexible, to avoid some limitations in the representation of human figures, and to smooth the limits of the evaluation parameters. The objectives of our fall detection and inactivity monitoring approach are the following ones:

- to detect static falls (of stationary humans) from a standing or sitting position,
- to detect dynamic falls (of walking or running humans), which may be frontal, backward and lateral (to the right or to the left) falls, and,
- to identify false falls such as kneeling and bending.

Lastly, in case of severe falls, when the human does not stand up after some pre-established inactivity monitoring time, t_m, the urgent need for medical assistance and/or care is assessed.

2.1. Segmentation for human detection

The proposed human detection algorithm is grounded on a segmentation algorithm based on a single frame. The algorithm is explained in detail in the following sections related to the different phases, namely, human candidate blobs detection, human candidate blobs refinement and human confirmation.

2.1.1. Human candidate blobs detection

The algorithm starts with the analysis of an input image $I(x, y)$, captured at time instant t by an infrared camera. Firstly, a change in scale, as shown in Equation (1) is performed in order to normalize the image. The objective of this operation is to always work on a similar scale of values, transforming $I(x, y)$ to $I^1(x, y)$. The normalization assumes a factor γ experimentally fixed as 60 and uses the mean gray level value of the current image, $\bar{I}$.

$$I^1(x, y) = \frac{I(x, y) \times \gamma}{\bar{I}} \tag{1}$$

where $I^1(x, y)$ is the normalized image. Notice that $I^1(x, y) = I(x, y)$ when $\bar{I} = \gamma$.

The algorithm uses a threshold to perform a binarization for the aim of isolating the human candidates spots. The threshold is calculated using adaptive thresholding [26] based on the standard deviation of $I(x, y)$, that is, $\sigma_{I(x,y)}$, obtaining the image areas which contain moderate heat blobs, and, therefore, belong to human candidates. Thus, the warmer zones of the image are isolated, assuming that they belong to humans. As the image has been scaled, the threshold θ_c is calculated as:

$$\theta_c = \frac{5}{4}(\gamma + \sigma_{I^1}) \tag{2}$$

where σ_{I^1} is the standard deviation of image $I^1(x, y)$. Notice that a tolerance value of a 25% above the sum of the mean image gray level value and the image gray level value standard deviation is offered.

Now, image I^1 is binarized using the obtained threshold θ_c. Pixels above the threshold are set as maximum gray level value $max = 255$ and pixels below are set as minimum gray level value $min = 0$ as shown in Equation (3).

$$I_b^1(x, y) = \begin{cases} \text{min}, & \text{if } I^1(x, y) \leq \theta_c \\ \text{max}, & \text{otherwise} \end{cases} \tag{3}$$

Next, morphological opening and closing operations are performed to eliminate isolated pixels and to unite areas split during the binarization. These operations require structuring elements that in both cases are 3×3 square matrices.

Afterwards, the blobs contained in the image are obtained. A minimum area (around $\frac{1}{16}$ of the image size) is established for consider a blob to contain a human. As a result, a list of blobs, L_B, containing human candidates in forms $b_\lambda[(x_{start}, y_{start}), (x_{end}, y_{end})]$ is generated. λ stands for the number of the human candidate blobs in image $I^1(x, y)$, whereas (x_{start}, y_{start}) and (x_{end}, y_{end}) are the upper left and lower right coordinates, respectively, of the minimum rectangle containing the blobs. At this point, there is a need to

validate the content of each blob to find out if it contains one single human candidate or more than one. Therefore, the algorithm processes each detected blob separately.

Let us define a region of interest (ROI) as the minimum rectangle containing one blob of a list L_B. A ROI may be defined as $R_\lambda = R_\lambda(i, j)$ when associated to blob $b_\lambda[(x_{start}, y_{start}), (x_{end}, y_{end})]$. Notice that $i\epsilon[1..max_i = x_{end} - x_{start} + 1]$ and $j\epsilon[1..max_j = y_{end} - y_{start} + 1]$.

2.1.2. Human candidate blobs refinement

2.1.2.1. Human vertical delimiting. The first step consists in scanning R_λ by columns, adding the gray level value corresponding to each column pixel, as shown in equation (4):

$$H_\lambda[i] = \sum_{j=1}^{max_j} R_\lambda(i, j), \forall i\epsilon[1..max_i] \tag{4}$$

This way, a histogram $H_\lambda[i]$, showing which zones of the ROI own greater heat concentrations, is obtained in order to increase the certainty of the presence and situation of human heads, as well as separate human groups (if any) into single human. For this purpose, local maxima and local minima are searched in $H_\lambda[i]$ to establish the different heat sources. To assess whether a histogram column contains a local maximum or minimum, a couple of thresholds are fixed, $\theta_{v_{max}}$, and $\theta_{v_{min}}$. The first threshold indicates the presence of the head of a human candidate on the current column, which we use as indicator of a local maximum as heads are normally warmer than the rest of the human's body covered by clothes. On the other hand, $\theta_{v_{min}}$ indicates those regions of the ROI where the sum of the heat sources are really low. These regions are supposed to belong to gaps between two humans and whose column summed gray level is below a 30% of the mean ROI gray level value. These two thresholds are calculated as shown in Equations (5) and (6) respectively.

$$\theta_{v_{max}} = (\frac{4}{3}\lambda) + \sigma_{I^1} \tag{5}$$

$$\theta_{vmin} = 0.3 \times \overline{R_\lambda} \times max_j \tag{6}$$

At the end of this process we obtain the human candidates of the scene separated into sub-ROIs, $sR_{\lambda,\alpha}$, vertically adjusted.

2.1.2.2. Human horizontal delimiting. Once the sub-ROIs have been obtained at the end of the previous section, we want to fit the height of each one of them to the real height of the human contained. This adjustment is performed by applying a new threshold, θ_h. The calculation is applied separately on each sub-ROI to avoid the influence of the rest of the image on the result. This threshold takes the value of the sub-ROI average gray level, $\theta_h = \overline{sR_{\lambda,\alpha}}$. Thus, sub-ROI $sR_{\lambda,\alpha}$ is binarized to delimit its upper and lower limits, obtaining $sR_{\lambda,\alpha,\beta}$ as shown in Equation (7) similarly to Equation (3.)

$$sR_{\lambda,\alpha,\beta}(i, j) = \begin{cases} \min, & \text{if } sR_{\lambda,\alpha}(i, j) \leq \theta_h \\ \max, & \text{otherwise} \end{cases} \tag{7}$$

After this, a closing is performed to unite spots isolated in the binarization, getting $sR_{\lambda,\alpha,\tau}$. Next, $sR_{c,\lambda,\tau}$ is scanned, searching pixels with values superior to min. The upper and lower rows of the human candidate are equal to the first and last rows, respectively, containing pixels with a value set to max. This way, we obtain a new set of sub-ROIs $sR_{\lambda,\alpha,\iota}$. These sub-ROIs are scanned separately in the next stage.

2.1.2.3. Human confirmation. Now a final stage is needed for each sub-ROI, $sR_{\lambda,\alpha,\iota}$, to confirm if the human candidate contained in it is actually a human. At this point, it is interesting to remind that every sub-ROI is defined by its coordinates (x_{start}, y_{start}) and (x_{end}, y_{end}). In first place, let us define the basic parameters needed for human confirmation for every sub-ROI. These are the "height of the sub-ROI", h_{sR}, the "width of the sub-ROI", w_{sR}, and the "height-to-width of the sub-ROI", $R_{hw_{sR}}$. The parameters are calculated as described next:

$$h_{sR} = x_{end} - x_{start} \tag{8}$$

$$w_{sR} = y_{end} - y_{start} \tag{9}$$

$$R_{hw_{sR}} = \frac{h_{sR}}{w_{sR}} \tag{10}$$

In order to confirm the presence of a human in a sub-ROI, the criterium established by Loomis [16] is used as an inspiration. Loomis studied in the 1940's the body part proportions of humans for different builds. His results are considered a standard in this area. Experimentally, and following his recommendations, we have imposed that $R_{hw_{sR}}\epsilon[0.2 \ldots 7.0]$. Finally, the blobs associated to ROIs that have not satisfied this criterium are deleted from the original blobs list, L_B.

2.2. Fuzzy-based fall detection algorithm

Our approach is based on the idea of combining a fall detection and an inactivity monitoring algorithm. In this case, the inactivity monitoring algorithm is thought for checking if a human continues laying inactive during some predefined time after a fall has been detected in the detection subsystem. In this paper, fall detection as such is based on the geometrical analysis of the blobs listed in L_B and representing humans. Roughly, it is decided that a fall has occurred through the study of the velocity of the deformation suffered by the blob along an established time interval. This time interval is denominated the "Fall time", t_f, which typically takes values between 1 and 3 seconds. Additionally, fuzzy logic is used in order to make decisions more flexible, to avoid some limitations in the representation of human figures, and to smooth the limits of the evaluation parameters.

Figure 1 illustrates the principal ideas of our approach towards fall detection and inactivity monitoring. In brief, during "Fall time", t_f, the fall detection subsystem is launched ("A" in Fig. 1). Once a fall is detected, the inactivity monitoring subsystem is activated ("B" and "C" in Fig. 1). If the fallen human stands up before a pre-established time interval, denominated "Monitoring time", t_m (typically 30 seconds), the inactivity monitoring subsystem is stopped, and the fall detection subsystem is launched again ("D" and "E" in Fig. 1).

2.2.1. Fall detection algorithm

Firstly, let us explain in more detail the proposed human fall detection algorithm. In our proposal, the general idea of fall detection is based on the joint static and kinematic analysis of the blobs corresponding to humans. The statical analysis corresponds to the study of a single (and current input) segmented infrared image, whilst kinematic analysis is based on a sequence of segmented infrared video images. During statical analysis, the geometrical parameters of each present blob are calculated, namely, the "change of the height of the blob", Δh, and the "width-to-height ratio of the blob", R_{wh}. The kinematic analysis implies the calculation of the "velocity of the change of the height of the blob", Δv, among a series of n consecutive segmented video frames. All these parameters, together with the predefined "Fall time", t_f, compose a set of fall indicators used in this research.

2.2.1.1. Calculation of the fall indicators. For the n consecutive blobs corresponding to a same human, let $y_{tr,1}$ and $y_{dl,1}$ be the top-right and down-left Y-coordinates of the blob in the first image, and let $y_{tr,n}$ and $y_{dl,n}$ be the top-right and down-left corner Y-coordinates of the blob in the last (n-th) image, respectively. In the same sense, let $x_{tr,1}$ and $x_{dl,1}$ be the top-right and down-left X-coordinates of the blob in the first image, and let $x_{tr,n}$ and $x_{dl,n}$ be the top-right and down-left corner X-coordinates of the blob in the n-th image. Let us also define x_{tr}, x_{dl}, y_{tr} and y_{ld} as the top-right and down-left coordinates of the current blob. Also, let $w = x_{tr} - x_{dl}$ and $h = y_{tr} - y_{dl}$ be the width and the height, respectively, of the current blob. The parameters are calculated as follows:

$$R_{wh} = \frac{x_{tr} - x_{dl}}{y_{tr} - y_{dl}} \tag{11}$$

$$\Delta h = \frac{y_{tr,1} - y_{dl,1}}{y_{tr,n} - y_{dl,n}} \tag{12}$$

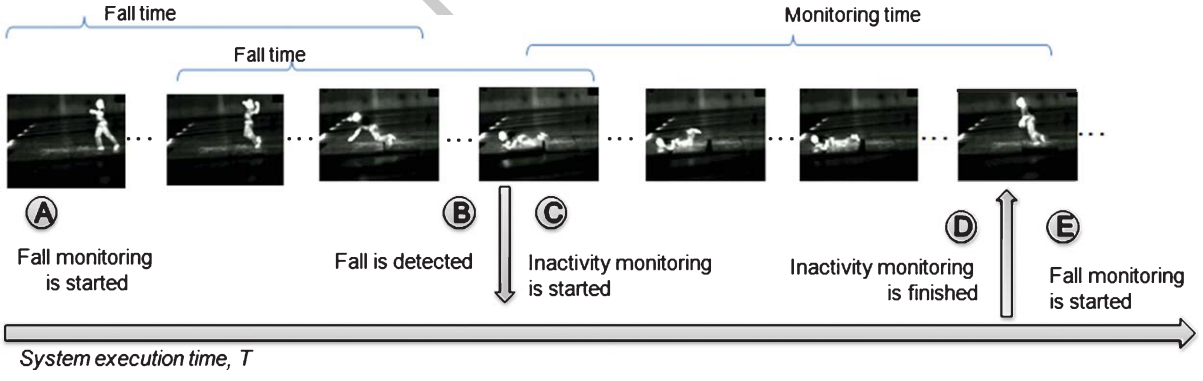

Fig. 1. Graphical representation of the fall detection and inactivity monitoring system.

$$\Delta v = \frac{\sqrt{\begin{array}{c}[(x_{tr,1} - x_{dl,1}) - (x_{tr,n} - x_{dl,n})]^2 \\ +[(y_{tr,n} - y_{dl,n}) - (y_{tr,n} - y_{dl,n})]^2\end{array}}}{\sqrt{w^2 + h^2}} \quad (13)$$

Notice that the denominator in formula (13) is the maximum possible velocity of a fall.

2.2.1.2. Fuzzy model for the fall indicators. After the previous parameters have been calculated, fuzzy logic is used in the proposed fall detection subsystem to determine the ranges for the fall indicators. Indeed, fuzziness for fall detection includes some input linguistic variables, namely "HeightChange", "VelocityChange" and "WidthToHeightRatio", directly related to Δh, Δv and R_{wh}, respectively, and an output linguistic variable called "Fall". The upper and lower bounds for the values of the fuzzy sets of the linguistic variables are real numbers within the intervals provided:

```
VelocityChange  REAL;
  (* RANGE(0 .. 1.25) *)
HeightChange    REAL;
  (* RANGE(0 .. 4) *)
WidthToHeightRatio  REAL;
  (* RANGE(0 .. 6) *)
```

The fuzzy sets for each of these input linguistic variables include three fuzzy terms, namely SMALL, MEDIUM and HIGH. Figure 2 shows the linguistic variables.

The linguistic variable "VelocityChange" is calculated as the relation of the real velocity to the maximum possible velocity through formula (13). Thus, in case of a very quick fall with velocity equal or close to the maximum, the following expression $\Delta v \approx 1.25$ is gotten. On the contrary, the change of velocity is minimal when $\Delta v \approx 0.0$. The terms for this linguistic variable have the following ranges:

SMALL$\in [0 \ldots 0.35]$
MEDIUM$\in [0.25 \ldots 0.65]$
HIGH$\in [0.6 \ldots 1.25]$

The linguistic variable "HeightChange" is calculated as the relation of the initial height to the final height of the blob. As stated in the expert work of Loomis [16], the width of a human's body is around 25 percent of his/her height. Taking into account different human builds, we have experimentally decided set up a value of 30 percent of a blob's height as the higher boundary for the HIGH term of linguistic variable "HeightChange".

The "WidthToHeightRatio" variable ranges from 0.0 to 6.0. Since a blob corresponding to a person in a strengthened position is longer in height than in width, the term SMALL $\in [0 \ldots 0.8]$, the term MEDIUM $\in [0.7 \ldots 2.5]$ stands for a situation where a blob belongs to a fallen person, or, alternatively, to a sitting or prone person. Finally, the term HIGH $\in [2.1 \ldots 6.0]$ describes a blob of a laying person.

The complete set of fuzzy rules used in this proposal to determine if a fall has occurred ("Fall is yes") is as shown in Table 1.

2.2.2. Fuzzy inference system

Figure 3 provides a schematic view of the proposed fuzzy inference system. Initially, input crisp variables are transformed into fuzzy sets within the "Fuzzification block". Next, the "Fuzzy inference engine" simulates the reasoning process by making fuzzy inference on the inputs and fuzzy "IF-THEN" rules which are stored within the "Knowledge base". The "Knowledge base" includes fuzzy rules and cases from "Fuzzy Data base". Lastly, the fuzzy set obtained is transformed by the "Fuzzy inference engine" into crisp values corresponding to output variables.

The fuzzy logic library Fuzzylite (see http://code.google.com/p/fuzzy-lite/) which provides a set of classes and methods for fuzzy interference system creation and manipulation is used. The fuzzy system is codified in agreement with the Fuzzy Control Language, which is a standard for Fuzzy Control Programming published by the International Electrotechnical Commission (IEC) (see http://www.ansi.org/). The linguistic variables describe spatial and kinetic properties

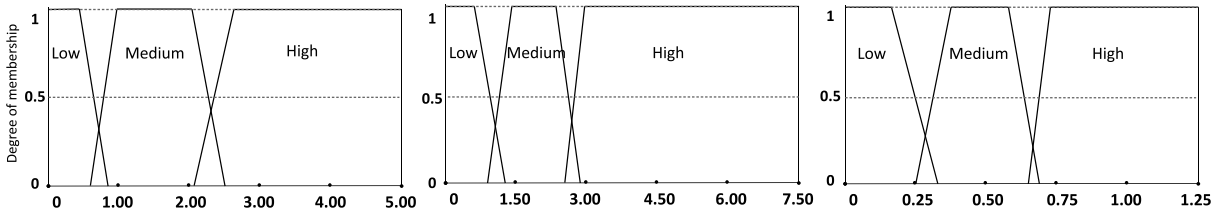

Fig. 2. From left to right, representation of the linguistic variables "WidthToHeightRatio", "VelocityChange" and "HeightChange".

Table 1
Fuzzy rules for fall detection

Rule	WidthToHeightRatio R_{wh}	HeightChange Δh	VelocityChange δv	Fall
1	HIGH	HIGH	-	YES
2	HIGH	-	HIGH	YES
3	HIGH	-	MEDIUM	YES
4	HIGH	MEDIUM	MEDIUM	YES
5	HIGH	MEDIUM	-	YES
6	MEDIUM	HIGH	HIGH	YES
7	MEDIUM	HIGH	MEDIUM	YES
8	MEDIUM	MEDIUM	HIGH	YES
9	MEDIUM	SMALL	HIGH	NO
10	MEDIUM	HIGH	SMALL	NO
11	MEDIUM	MEDIUM	SMALL	NO
12	MEDIUM	SMALL	SMALL	NO
13	MEDIUM	SMALL	MEDIUM	NO
14	MEDIUM	MEDIUM	MEDIUM	YES
15	MEDIUM	SMALL	SMALL	NO
16	SMALL	-	-	NO

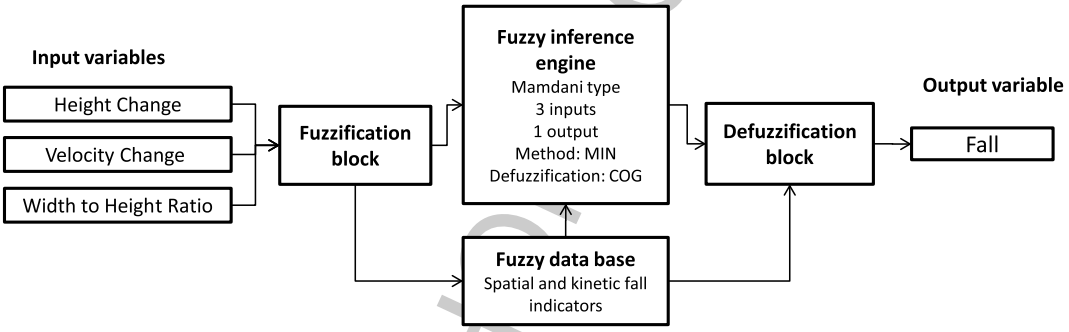

Fig. 3. Fuzzy inference system for fall patterns recognition.

of ROIs, which correspond to humans. The proposed fuzzy model for fall detection includes input linguistic variables described previously, that is, "HeightChange" (Δh), "VelocityChange" (Δv), "WidthToHeightRatio" (R_{wh}), and the output linguistic variable called "Fall".

With the purpose of creating a fuzzy model, firstly the fuzzy engine is initialized as:

```
fl::FuzzyEngine engine;
    engine.hedgeSet().
      add(new fl::HedgeNot);
    engine.hedgeSet().
      add(new fl::HedgeSomewhat);
    engine.hedgeSet().
      add(new fl::HedgeVery);
```

Then, the input variables A (which corresponds to R_{wh}), H (which stands for Δh), V (which is Δv), as well as the output variable *fall* are determined and added to a fuzzy engine.

```
fl::InputLVar* A = new fl::
  InputLVar("A");
A->addTerm(new fl::ShoulderTerm
  ("LOW", 0.6, 0.8,true));
A->addTerm(new fl::TriangularTerm
  ("MEDIUM", 0.7, 1.2));
A->addTerm(new fl::ShoulderTerm
  ("HIGH", 1.0, 1.3,false));
engine.addInputLVar(A);

fl::InputLVar* H = new fl::
  InputLVar("Hchange");
H->addTerm(new fl::ShoulderTerm
  ("LOW", 0.5, 1.2,true));
H->addTerm(new fl::TriangularTerm
  ("MEDIUM", 1.4, 2.5));
```

```
  H->addTerm(new fl::ShoulderTerm
    ("HIGH", 2.2, 3.0,false));
  engine.addInputLVar(H);

  fl::InputLVar* V = new fl::
    InputLVar("Vdelta");
  V->addTerm(new fl::ShoulderTerm
    ("LOW", 0.2, 0.35,true));
  V->addTerm(new fl::TriangularTerm
    ("MEDIUM", 0.4, 0.65));
  V->addTerm(new fl::ShoulderTerm
    ("HIGH", 0.6, 0.75, false));
  engine.addInputLVar(V);

  fl::OutputLVar* fall = new fl::
    OutputLVar("Fall");
  fall->addTerm(new fl::
    TriangularTerm("NO", 0.0, 45.0));
  fall->addTerm(new fl::
    TriangularTerm("YES", 35.0, 100.0));
  engine.addOutputLVar(fall);
```

Finally, the set of rules is created.

```
fl::RuleBlock* block = new fl::
    RuleBlock();
    block->addRule(new fl::
    MamdaniRule
   ("if A is HIGH and Hchange is
   HIGH and Vdelta is HIGH then Fall
   is YES",engine));
   ...
   engine.addRuleBlock(block);
```

To conclude, the responses of the fuzzy system for values of "HeightChange", "VelocityChange" and "WidthToHeightRatio" are calculated.

2.2.3. Inactivity monitoring algorithm

Notice that there are two main variables for final decision making. The first one is *Alarm*, which is used to indicate if a fall has been detected. The variable holds values "FallDetected" and "NoFallDetected". Initially *Alarm* is set to "NoFallDetected" until a fall is supposed in accordance with the fall detection subsystem described before.

The other important variable is *Assistance*, indicating if there is a need to assist a fallen person. The possible values for *Assistance* are "AssistanceRequired" and "NoAssistanceRequired", being initially provided value "NoAssistanceRequired". *Assistance* is only valid after a fall has been detected and the inactivity monitoring mode has been launched. Thus, at the beginning of each fall time cycle, *Alarm* and *Assistance* take values "NoFallDetected" and "NoAssistanceRequired", respectively. The inactivity monitoring subsystem behavior may be described as follows:

1. When the fuzzy model detects a fall, variables *Alarm* and *Assistance* are set to "FallDetected" and "NoAssistanceRequired", which means that a fall has been detected and the inactivity monitoring is started. During inactivity monitoring, with arrival of the new blobs, the Δh, R_{wh}, and Δv are calculated and then checked with the same fuzzy model.
2. If a person has fallen (that is, *Alarm* has value "FallDetected") and inactivity monitoring has been active for time interval t_m, *Assistance* is provided the value "AssistanceRequired". This means that medical care or assistance is needed.
3. In case the fuzzy model newly generates "NoFallDetected" for the *Alarm* variable, it is supposed that the fallen human has risen on his/her feet and the *Assistance* variable is set to "NoAssistanceRequired". In this case, the inactivity monitoring finishes and the fall detection subsystem is re-activated.

3. Data and results

The experiments described next were set up in order to validate the proposed approach. The experiments were carried out on a personal computer equipped with an Intel Core i7 processor with 3 GB of RAM. The video sequences were recorded with a FLIR A320 infrared camera at a resolution of 720×480 pixels. Five series containing a single person were used to validate our research. The experiments were carried out in accordance with the following list of objectives:

1. Detection of static falls.
2. Detection of dynamic falls.
3. Identification of false falls.

In this approach, falls are simulated by an actor. The results are offered in Table 2. The first column contains the number of the detection experiment. The following columns contain values for the calculated parameters Δh, R_{wh}, and Δv for each time fall period t_f. The *Alarm* column shows the response of the fuzzy model for the fall detection, which is "FallDetected" in case of a fall

Table 2
Fall detection results

WidthToHeightRatio R_{wh}	HeightChange Δh	VelocityChange δv	*Alarm*	*Assistance*
Static falls				
1. Frontal fall (a frontal view)				
MEDIUM (1.14)	HIGH (2.72)	MEDIUM (0.35)	FallDetected	AssistanceRequired
MEDIUM (1.53)	HIGH (3.04)	MEDIUM (0.41)	FallDetected	AssistanceRequired
MEDIUM (1.33)	MEDIUM (2.10)	SMALL (0.28)	NoFallDetected	NoAssistanceRequired
2. Backward fall from a "sitting" position				
HIGH (2.41)	HIGH (2.9)	MEDIUM (0.46)	FallDetected	AssistanceRequired
HIGH (3.07)	HIGH (3.8)	MEDIUM (0.49)	FallDetected	AssistanceRequired
HIGH (3.04)	MEDIUM (1.19)	SMALL (0.06)	NoFallDetected	NoAssistanceRequired
HIGH (2.78)	MEDIUM (1.12)	SMALL (0.03)	NoFallDetected	NoAssistanceRequired
3. Lateral fall from a "sitting" position				
MEDIUM (1.02)	SMALL (0.98)	SMALL (0.12)	NoFallDetected	NoAssistanceRequired
MEDIUM (1.48)	MEDIUM (1.27)	MEDIUM (0.37)	FallDetected	AssistanceRequired
MEDIUM (1.54)	MEDIUM (1.47)	MEDIUM (0.35)	FallDetected	AssistanceRequired
MEDIUM (1.28)	MEDIUM (0.93)	HIGH(1.48)	FallDetected	AssistanceRequired
MEDIUM (1.35)	HIGH (2.6)	MEDIUM (0.53)	FallDetected	AssistanceRequired
Dynamic falls				
1. Frontal fall (a lateral view)				
MEDIUM (1.49)	MEDIUM (1.74)	HIGH (0.85)	FallDetected	AssistanceRequired
HIGH (1.95)	MEDIUM (2.04)	HIGH (1.02)	FallDetected	AssistanceRequired
HIGH (2.8)	HIGH (2.77)	HIGH (1.13)	FallDetected	AssistanceRequired
HIGH (3.78)	HIGH (3.56)	HIGH (1.13)	FallDetected	AssistanceRequired
False fall				
2. False fall - Kneeling person				
MEDIUM (1.35)	MEDIUM (1.37)	SMALL (0.16)	NoFallDetected	NoAssistanceRequired

and "NoFallDetected" otherwise. *Assistance* is set to "AssistanceRequired" in case of inactivity after a fall and "NoAssistanceRequired" in case a person has risen.

Figures 4–6 show resulting images where falls are initially assumed in the experiment. There are red and blue bounding boxes on the images. In the paper printed version red is light gray and blue is dark gray. A red bounding box shows the rectangular borders of the ROI of the last blob, whereas a blue bounding box retains the borders corresponding to the first blob originating the fall presumption. Thus, fall indicators are calculated using the red and the blue bounding boxes. The capture of the video images was performed with an interval of 200 milliseconds, and the fall time was set to $t_f = 1.2$ seconds. Thus, fall detection is tested on every six consecutive frames. As you may notice below all cases studied in these experiments work fine. Table 2 contains values for evaluation parameters for these supposed falls.

3.1. Detection of static falls

Static falls include falling from a standing and/or sitting position. Falls from a standing position are characterized through high horizontal velocity and height change. Also, the width-to-height ratio changes significantly. On the other hand, falls from the sitting position may not have such high values of height change and width-to-height ratio change, as opposed to the previous case, but the horizontal (in case of a lateral fall) or the vertical (in case of frontal or backward fall) components of the velocity are high. Figure 4 shows fall moments for three statical falls, that is two falls from a standing position, and one fall from a sitting position.

The first static fall (shown in the two upper rows of Fig. 4), denominated "frontal fall (captured from a lateral - frontal view))" has the following characteristics at the moment of fall detection. "WidthToHeightRatio" is MEDIUM (from $R_{wh} = 1.14$), "HeightChange" is HIGH (from $\Delta h = 2.72$), and "VelocityChange" is MEDIUM (from $\Delta v = 0.35$). *Alarm* is "FallDetected" and *Assistance* is set to "AssistanceRequired". The same parameters calculated for the next image keep the same terms for their corresponding linguistic variables, that is MEDIUM (1.53), HIGH (3.04) and MEDIUM (0.41). Next, "VelocityChange" reduces to SMALL (0.28), and "HeightChange" decreases to MEDIUM (2.10) terms, which makes change values of variables

Fig. 4. Some static falls. (a) Two top rows: a frontal fall (captured from a lateral - frontal view). (b) Third row: a backward fall from a "sitting" position. (c) Two last rows: a lateral fall from a "sitting" position (captured from a lateral view).

Alarm and *Assistance* to "NoFallDetected" and "NoAssistanceRequired". See the three situations described on top of Table 2.

The second static fall is a backward fall from a "sitting position" (see Fig. 5, third row). The moment when a person falls is detected because of the abrupt increase of R_{wh} from 2.41 to 3.07 and Δh from 2.9 to 3.8. "VelocityChange" takes the MEDIUM linguistic term. Notice that Δv takes here values ranging between 0.46 and 0.49. In agreement with with the fuzzy rules block,

Fig. 5. A dynamic frontal fall (captured from a lateral view).

Fig. 6. Identification of a false fall: Kneeling.

the output variable "Fall" is set to "yes". This way, *Alarm* is set to "FallDetected". After t_m has elapsed, the variable *Assistance* is set to "AssistanceRrequired".

The last described static fall is detected from a "sitting" position (see Fig. 5, last two rows)), when "WidthToHeightRatio" is MEDIUM (related to $R_{wh} = 1.48$), "HeightChange" is MEDIUM ($\Delta h = 1.27$), and "VelocityChange" is MEDIUM ($\Delta v = 0.37$). As a response of the fuzzy model, variable *Alarm* is set to "FallDetected". Then, to the end of monitoring time t_m, "HeightChange" is increased to the term HIGH ($\Delta h = 2.6$), and *Assistance* is set to "AssistanceRequired".

3.2. Detection of dynamic falls

An example of a dynamic fall is shown in Fig. 5. It is denominated "a dynamic frontal fall (captured from a lateral view)" and represents a fall of a walking person. At the moment of the fall, the parameters "WidthToHeightRatio" and "HeightChange" are within the MEDIUM term (with $R_{wh} = 1.49$ and $\Delta h =$

1.74), whilst velocity of the fall is HIGH ($\Delta v = 0.85$). The output variable "Fall" is set to "yes", implying that variable *Alarm* has to be is set to "FallDetected". The following images are characterized with abrupt increase of the fall indicator parameters, as R_{wh} grows from 1.95 to 3.78, "HeightChange" passes from MEDIUM (2.04) to HIGH (3.56) linguistic term, and Δv obtains very high values, from 1.02 to 1.13. A rapid increase of velocity is the distinctive evidence of this sort of fall.

3.3. Identification of false falls

False falls are represented in this case study with the kneeling action. In fact, a false fall may be detected as a fall which lasts more than a fall time as well as having small values for fall detection parameters given in the last case shown in Table 2. As a rule, in case of a false fall, linguistic variable "WidthToHeightRatio" may have SMALL or MEDIUM values, and linguistic variable "VelocityChange" is rather SMALL. Figure 6 shows a moment of a false fall. In compliance with the results provided in Table 2, a fall is not detected in this case. Thus, the algorithms proposed were able to differentiate true falls from false falls. Table 2 contains values for evaluation parameters for the "kneeling person" situation with 'WidthToHeightRatio" in MEDIUM (1.35), "HeightChange" in MEDIUM (1.37), and "VelocityChange" in SMALL (0.16).

4. Discussion

During the experimentation a set of twenty one fall sequences was studied. The set includes fifteen simulated true falls and six false falls. Table 3 offers the confusion matrix used for the evaluation of the proposed fuzzy-based fall detection system.

Next, sensitivity, accuracy and specificity are calculated as:

$$Sensitivity = \frac{TP}{TP + FN} = \frac{14}{14 + 1} = 0.933 \quad (14)$$

Table 3
Confusion matrix for classification of falls

Event	Detected falls	Not detected falls
Falls	True Positive (TP)	False Positive (FP)
15	14	1
Nonfalls	False Negative (FN)	True Negative (TN)
6	1	5

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{14 + 5}{14 + 1 + 1 + 5} = 0.904 \quad (15)$$

$$Specificity = \frac{TN}{TN + FP} = \frac{5}{5 + 1} = 0.833 \quad (16)$$

Notice that the estimators demonstrate that the recognition system obtains very good results. Indeed, the value of the sensitivity, which returns a percentage of correct classification, is equal to 93.3% in our case. The accuracy of the system is also high (90.4%), and a specificity of 83.3% is more than acceptable.

Next, a comparison of our results with other similar approaches is performed. The proposal described in [24] achieves results with sensitivity and specificity of 90.7% and 90.6%, respectively. Moreover, there are 95% confidence intervals within 86.8–93.7%, and 87.0–93.4%, respectively. Another fall detection system [23] offers values for sensitivity around 88% and specificity is 87.5%. Lastly, the fuzzy-based algorithm proposed in [4] recognizes fall events with an accuracy rate varying between 74% and 100%, and the average detection performance when combining fall and non-fall events falls within the interval [80.5–84.3%], depending on the location of the sensor. Another paper [2] reports around 100% fall detection (from a set of 14 falls) with 6.0% of false negatives. Notice that our approach reaches low false positives (6.67%). This is lower that the values reported in some recent related works (for example, 16.67% in [8], and 2/17 or 11.76% in [23]).

Thus, the performance of our fall recognition system demonstrates a high performance and is competitive against similar previous approaches. The fact that the proposed fall detection system provides an adequate response solely based on the kinematic and geometrical analysis of a given ROI is also a great benefit. The limited number of fall parameters to evaluate further calculations reduces the need for great computation resources.

5. Conclusions

Falls of elderly people is a big problem which needs rapid and effective solutions. In this paper we have presented a proposal that includes a human detection algorithm in infrared video and a fuzzy-based model for fall detection and inactivity monitoring. The data supplied by the human detection algorithms are used by

the fuzzy model to detect if a true fall has occurred. The geometrical characteristics of the blob corresponding to the detected person, and velocity of the change of its region of interest serve as fall indicators. Additionally, the fuzzy model enables to avoid certain limitations in parameter evaluation, and to make smoother and more flexible decisions, on the other hand.

The proposed algorithms are incorporated into a fall detection system for the elderly, and then tested for a wide number of static and dynamic falls, including the identification of false falls. Experimental results, which have been carried out for static and dynamic falls, have demonstrated that the proposed algorithms are able to detect them and to distinguish true from false falls. The system is also designed to claim for medical assistance when the fall persists in time.

Acknowledgments

This work was partially supported by Spanish Ministerio de Economía y Competitividad/FEDER under project TIN2010-20845-C03-01, by Spanish Ministerio de Industria, Turismo y Comercio/FEDER (Plan Avanza 2) under project TSI-020100-2010-261, and by Spanish Junta de Comunidades de Castilla-La Mancha/FEDER under project PII2I09-0069-0994.

References

[1] D. Anderson, J.M. Keller, M. Skubic, X. Chen and Z. He, Recognizing falls from silhouettes, *Proceedings of the IEEE Engineering in Medicine and Biology Society Conference* (2006), 6388–6391.

[2] D. Anderson, R.H. Luke, J.M. Keller, M. Skubic, M. Rantz and M. Aud, Linguistic summarization of video for fall detection using voxel person and fuzzy logic, *Comput Vis Image Underst* **113**(1) (2009), 80–89.

[3] J. Bart and D. Rudi, Context aware inactivity recognition for visual fall detection, *Pervasive Health Conference and Workshops* (2006), 1–4.

[4] P. Boissy, S. Choquette, M. Hamel and N. Noury, User-Based Motion Sensing and Fuzzy Logic for Automated Fall Detection in Older Adults, *Telemedicine and e-Health* **13**(6) (2007), 683–694.

[5] J.C. Castillo, J. Serrano-Cuerda and A. Fernández-Caballero, Robust people segmentation by static infrared surveillance camera, *Lecture Notes in Computer Science* **6096** (2010), 348–357.

[6] S.-L. Chang, F.-T. Yang, W.-P. Wu, Y.-A. Cho and S.-W. Chen, Nighttime pedestrian detection using thermal imaging based on HOG feature, *International Conference on System Science and Engineering* (2011), 694–698.

[7] J.W. Davis and V. Sharma, Robust detection of people in thermal imagery, *Proceedings of the 17th International Conference on Pattern Recognition* **4** (2004), 713–716.

[8] C. Doukas and I. Maglogiannis, Emergency fall incidents detection in assisted living environments utilizing motion, sound, and visual perceptual components, *IEEE Transactions on Information Technology in Biomedicine* **15**(2) (2011), 277–289.

[9] A. Escobet, A. Nebot and F.E. Cellier, Fault diagnosis system based on fuzzy logic: Application to a valve actuator benchmark, *Journal of Intelligent and Fuzzy Systems* **22**(4) (2011), 155–171.

[10] Y. Fang, K. Yamada, Y. Ninomiya, B.K.P. Horn and I. Masaki, A shape-independent method for pedestrian detection with far-infrared images, *IEEE Transactions on Vehicular Technology* **53**(6) (2004), 1679–1697.

[11] A. Fernández-Caballero, J.C. Castillo, J. Martínez-Cantos and R. Martínez-Tomás, Optical flow or image subtraction in human detection from infrared camera on mobile robot, *Robotics and Autonomous Systems* **58**(12) (2010), 1273–1281.

[12] A. Fernández-Caballero, J.C. Castillo, J. Serrano-Cuerda and S. Maldonado-Bascón, Real-time human segmentation in infrared videos, *Expert Systems with Applications* **38**(3) (2011), 2577–2584.

[13] S. Khawandi, B. Daya and P. Chauvet, Implementation of a monitoring system for fall detection in elderly healthcare, *Procedia Computer Science* **3** (2011), 216–220.

[14] J. Li, W. Gong, W. Li and X. Liu, Robust pedestrian detection in thermal infrared imagery using the wavelet transform, *Infrared Physics and Technology* **53**(4) (2010), 267–273.

[15] C.-L. Liu, C.-H. Lee and P.-M. Lin, A fall detection system using k-nearest neighbor classifier, *Expert Systems with Applications* **37**(10) (2010), 7174–7181.

[16] A. Loomis, Figure Drawing for All it's Worth, Viking Adult, 1943.

[17] J. Machajdik, S. Zambanini and M. Kampel, Fusion of data from multiple cameras for fall detection, *Proceedings of the Workshop on Behaviour Monitoring and Interpretation* (2010), 1–7.

[18] H.-C. Mo, J.-J. Leou and C.-S. Lin, Human behavior analysis using multiple 2d features and multicategory support vector machine, *Proceedings of the IAPR Conference on Machine Vision Applications* (2009), 46–49.

[19] H. Moon, R. Chellappa and A. Rosenfeld, Tracking of human activities using shape-encoded particle propagation, *Processing of the International Conference on Image Processing* **1** (2001), 357–360.

[20] J. Moreno-Garcia, L. Rodriguez-Benitez, A. Fernández-Caballero and M.T. López, Video sequence motion tracking by fuzzification techniques, *Applied Soft Computing* **10**(1) (2010), 318–331.

[21] N. Noury, A. Fleury, P. Rumeau, A.K. Bourke, G.O. Laighin, V. Rialle and J.E. Lundy, Fall detection - principles and methods, *Proceedings of the 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (2007), 1663–1666.

[22] N. Otsu, A threshold selection method from gray level histograms, *IEEE Transactions on Systems, Man and Cybernetics* **9**(1) (1979), 62–66.

[23] C. Rougier and J. Meunier, Demo: Fall detection using 3d head trajectory extracted from a single camera video sequence, *Proceedings of the First International Workshop on Video Processing for Security*, 2006. Available on: http://www.computer-vision.org/4security/pdf/montreal-fall_detection.pdf

[24] H. Rimminen, J. Lindström, M. Linnavuo and R. Sepponen, Detection of falls among the elderly by a floor sensor using the

electric near field, *Trans Info Tech Biomed* **14**, 6 (November 2010), 1475–1476.
[25] J. Schmidt, J. Fritsch and B. Kwolek, Kernel particle filter for realtime 3d body tracking in monocular color images, *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition* (2006), 567–572.
[26] L.G. Shapiro and G.C. Stockman, Computer Vision, Prentice-Hall, 2002.
[27] S.C. Tan and C.P. Lim, Fuzzy ARTMAP and hybrid evolutionary programming for pattern classification, *Journal of Intelligent and Fuzzy Systems* **22**(2) (2011), 57–68.
[28] X. Wei and J. Chai, Modeling 3d human poses from uncalibrated monocular images, *Proceedings of the IEEE International Conference on Computer Vision* (2009), 1873–1880.
[29] World Health Organization. WHO global report on falls prevention in older age, WHO Press, 2007.
[30] M. Yu, A. Rhuma, S.M. Naqvi and J. Chambers, Fall detection for the elderly in a smart room by using an enhanced one class support vector machine, *Proceedings of the 17th International Conference on Digital Signal Processing* (2011), 1–6.
[31] X. Yu, X. Wang, P. Kittipanya-Ngam, H.L. Eng and L.-F. Cheong, Fall detection and alert for ageing-at-home of elderly, *Proceedings of the 7th International Conference on Smart Homes and Health Telematics: Ambient Assistive Health and Wellness Management in the Heart of the City* (2009), 209–216.